Inventory Management System

Project

Database Management System

(Using PostgreSQL)

Overview

- Project is related to Inventory Management System
- The project maintains three levels of users:
  - Billing Counter Level
  - Manager Level
  - Owner Level
- Main facilities available in this project are:
  - We can forecast the sales by analyzing the previous sales statistics.
  - We can get an idea that when we need to order new inventory.
  - We can reduce the chances of any kind of frauds done by the staff members in the inventory.
  - Customer details can be added.
  - Invoice generation.
  - We can keep a track of transactions received through different payment methods.

Introduction

An inventory management system is the combination of technology (hardware and software) and processes and procedures that oversee the monitoring and maintenance of stocked products, whether those products are company assets, raw materials and supplies, or finished products ready to be sent to vendors or end consumers. This system can widely be used by normal shops, departmental stores, or MNCs for keeping a proper track of the stock. It also consists of information like manager details, customer details, etc. With the help of this system, we can fix a minimum quantity of any inventory below which we need to place an order for that inventory. This will help us in good sales results and avoid the out-of-stock stage for any inventory.

Description

SCOPE :

- This will help us in maintaining the exact count of any product.
- Can help us set the minimum quantity of any product below which we need to order the product from the manufacturer.
- Can reduce duplicate entries.

WORKING :

This application will have different front ends for different kinds of users. The person who is sitting at the billing counter will have access only to modify the quantity of any product; they can either generate an invoice for any sold product or generate a return note for any returns from any customer. The manager will have access to modify the rates if there exist any dynamic price inventories. The owner of the firm will have access to generate the final report, which will consist of sales done on any particular day, the total sales on any particular counter, or by any salesperson.

PURPOSE :

INVENTORY MANAGEMENT must tie together the following objectives, to ensure that there is continuity between functions:

– Company's Strategic Goals

– Sales Forecasting

– Sales & Operations Planning

– Production & Materials Requirement Planning

Inventory Management must be designed to meet the dictates of the marketplace and support the company's Strategic Plan. The many changes in market demand, new opportunities due to worldwide marketing, global sourcing of materials, and new manufacturing technology mean many companies need to change their Inventory Management approach and adjust the process for Inventory Control.

Inventory Management systems provide information to efficiently manage the flow of materials, effectively utilize people and equipment, coordinate internal activities, and communicate with customers. Inventory Management does not make decisions or manage operations; it provides the information to managers who make more accurate and timely decisions to manage their operations.

INVENTORY is defined as the blocked Working Capital of an organization in the form of materials. As this is the blocked Working Capital of the organization, ideally it should be zero. However, we maintain Inventory to account for fluctuations in demand and lead time. In some cases, it is maintained to manage increasing price tendencies of commodities or rebates from bulk buying.

BACKGROUND :

This application is nowadays a basic use of any company, firm, shop, or departmental store because stock maintenance and stock forecasting are essential for earning great profits.

In ancient times, inventory was maintained using paper and pen methods. These methods were quite cumbersome, uncomfortable, and sometimes inaccurate. To overcome these problems, we developed an inventory management system. This system allows us to generate invoices for each purchase and includes employee and customer details.

In short, we can call this an all-in-one system..!!

USER CHARACTERISTICS :

Every user should be:

– Comfortable with computers.

– Knowledgeable in using internet browsers.

– Possess basic knowledge of English.

Goals of Proposed System :

1. **Planned approach towards working:** The working in the organization will be well-planned and organized. The data will be stored properly in data stores, which will help in the retrieval of information as well as its storage.
2. **Accuracy:** The level of accuracy in the proposed system will be higher. All operations will be done correctly, ensuring that whatever information is coming from the center is accurate.
3. **Reliability:** The reliability of the proposed system will be high due to the above-stated reasons. The increased reliability is due to proper storage of information.
4. **No Redundancy:** In the proposed system, utmost care will be taken to ensure that no information is repeated anywhere, in storage or otherwise. This will assure the economic use of storage space and consistency in the data stored.
5. **Immediate retrieval of information:** The main objective of the proposed system is to provide quick and efficient retrieval of information.
6. **Immediate storage of information:** In manual systems, there are many problems associated with storing large amounts of information.
7. **Easy to Operate:** The system should be easy to operate, developed within a short period of time, and fit within the limited budget of the user.

Technical Feasibility: Back End

In this project, we've only implemented the back end of the system, which is designed on **PostgreSQL**.
On this structured query language, we created 10 tables named:

1. Brands
2. inv_user
3. Categories
4. Products
5. Stores
6. Providers
7. Customer_cart
8. Select_product
9. Transaction
10. Invoice

ADVANTAGES :

1. **Inventory Balance.** Good inventory management helps you figure out exactly how much inventory you need. This makes it easier to prevent product shortages and keep just enough inventory on hand without having too much.
2. **Inventory Turnover.** You need to maintain a high inventory turnover ratio to ensure your products aren't spoiling, becoming obsolete, or sucking up your working capital. Calculate how many times your inventory sells in a year and identify areas where you can make better use of your resources.
3. **Repeat Customers.** Good inventory management leads to repeat customers. You want your hard-earned customers to keep returning to your business. Ensuring you have what they're looking for every time they visit is key to achieving this.
4. **Accurate Planning.** Smart inventory management allows you to stay ahead of the demand curve, maintain the right amount of products on hand, and plan ahead for seasonal changes, ensuring customer satisfaction year-round.
5. **Warehouse Organization.** Knowing your top-selling products and the combinations often ordered together helps optimize warehouse setup. Placing these products in accessible locations speeds up picking, packing, and shipping processes.
6. **Employee Efficiency.** Empowering employees with barcode scanners, inventory management software, and other tools helps them manage inventory better, improving both human and technological resource utilization.
7. **Inventory Orders.** Keeping track of inventory allows smarter decisions about when and what to order. Inventory management software speeds up this process by enabling product barcode scanning and quick order generation.
8. **Inventory Tracking.** For businesses with multiple locations, inventory management becomes critical in coordinating supplies at each location based on demand and other factors.
9. **Time Saving.** Inventory management saves time by keeping accurate records of products on hand and on order, reducing the need for time-consuming recounts.
10. **Cost Cutting.** Efficient inventory management helps avoid wasting money on slow-moving products, enabling better use of resources in other business areas.

SUMMARY :

In this project, we developed a complete back-end software solution that allows us to update stock, modify stock, forecast stock, and generate invoices.

From this application, we can receive updates when a particular inventory or stock falls below a pre-set quantity, making it easier for the manager or owner to reorder products from suppliers and avoid the "Out of Stock" stage.

Additionally, this software helps manage warehouses by adding new ones as needed, which can be a valuable feature. It stores complete customer details, helping to retrieve order histories for regular customers.

Furthermore, the system keeps track of transactions performed by different customers or clients and provides insight into how much revenue has been generated through various payment methods.

This application will help maintain a high inventory turnover ratio, ensuring that products don't spoil or become obsolete, and making better use of our working capital. It also assists in calculating how many times the inventory sells in a year, enabling us to optimize our resource utilization.

SQL Code Implementation :

```sql
-- Creating brands table
CREATE TABLE brands (
    bid SERIAL PRIMARY KEY,
    bname VARCHAR(20)
);

-- Creating inv_user table
CREATE TABLE inv_user (
    user_id VARCHAR(20) PRIMARY KEY,
    name VARCHAR(20),
    password VARCHAR(20),
    last_login TIMESTAMP,
    user_type VARCHAR(10)
);

-- Creating categories table
CREATE TABLE categories (
    cid SERIAL PRIMARY KEY,
    category_name VARCHAR(20)
);

-- Creating stores table
CREATE TABLE stores (
    sid SERIAL PRIMARY KEY,
    sname VARCHAR(20),
    address VARCHAR(20),
    mobno BIGINT
);

-- Creating product table
CREATE TABLE product (
    pid SERIAL PRIMARY KEY,
    cid INT REFERENCES categories(cid),
    bid INT REFERENCES brands(bid),
    sid INT REFERENCES stores(sid),
    pname VARCHAR(20),
    p_stock INT,
    price INT,
    added_date DATE
);

-- Creating provides table
CREATE TABLE provides (
    bid INT REFERENCES brands(bid),
    sid INT REFERENCES stores(sid),
    discount INT
);

-- Creating customer_cart table
CREATE TABLE customer_cart (
    cust_id SERIAL PRIMARY KEY,
    name VARCHAR(20),
    mobno BIGINT
);

-- Creating select_product table
CREATE TABLE select_product (
    cust_id INT REFERENCES customer_cart(cust_id),
    pid INT REFERENCES product(pid),
    quantity INT
);

-- Creating transaction table
CREATE TABLE transaction (
    id SERIAL PRIMARY KEY,
    total_amount INT,
    paid INT,
    due INT,
    gst INT,
    discount INT,
    payment_method VARCHAR(10),
    cart_id INT REFERENCES customer_cart(cust_id)
);

-- Creating invoice table
CREATE TABLE invoice (
    item_no SERIAL PRIMARY KEY,
    product_name VARCHAR(20),
    quantity INT,
    net_price INT,
    transaction_id INT REFERENCES transaction(id)
);

-- Inserting into brands
INSERT INTO brands (bname) VALUES ('Apple'), ('Samsung'), ('Nike'), ('Fortune');

-- Inserting into inv_user
INSERT INTO inv_user (user_id, name, password, last_login, user_type)
VALUES
('vidit@gmail.com', 'vidit', '1234', '2018-10-31 12:40', 'admin'),
('harsh@gmail.com', 'Harsh Khanelwal', '1111', '2018-10-30 10:20', 'Manager'),
('prashant@gmail.com', 'Prashant', '0011', '2018-10-29 10:20', 'Accountant');

-- Inserting into categories
INSERT INTO categories (category_name) VALUES ('Electronics'), ('Clothing'), ('Grocery');

-- Inserting into stores
INSERT INTO stores (sname, address, mobno)
VALUES
('Ram kumar', 'Katpadi vellore', 9999999999),
('Rakesh kumar', 'Chennai', 8888555541),
('Suraj', 'Haryana', 7777555541);

-- Inserting into product
INSERT INTO product (cid, bid, sid, pname, p_stock, price, added_date)
VALUES
(1, 1, 1, 'IPHONE', 4, 45000, '2018-10-31'),
(1, 1, 1, 'Airpods', 3, 19000, '2018-10-27'),
(1, 1, 1, 'Smart Watch', 3, 19000, '2018-10-27'),
(2, 3, 2, 'Air Max', 6, 7000, '2018-10-27'),
(3, 4, 3, 'REFINED OIL', 6, 750, '2018-10-25');

-- Inserting into provides
INSERT INTO provides (bid, sid, discount)
VALUES
(1, 1, 12),
(2, 2, 7),
(3, 3, 15),
(1, 2, 7),
(4, 2, 19),
```

```sql
(4, 3, 20);

-- Inserting into customer_cart
INSERT INTO customer_cart (name, mobno)
VALUES
('Ram', 9876543210),
('Shyam', 7777777777),
('Mohan', 7777777775);

-- Inserting into select_product
INSERT INTO select_product (cust_id, pid, quantity)
VALUES
(1, 2, 2),
(1, 3, 1),
(2, 3, 3),
(3, 2, 1);

-- Inserting into transaction
INSERT INTO transaction (total_amount, paid, due, gst, discount, payment_method, cart_id)
VALUES
(57000, 20000, 5000, 350, 350, 'card', 1),
(57000, 57000, 0, 570, 570, 'cash', 2),
(19000, 17000, 2000, 190, 190, 'cash', 3);

CREATE OR REPLACE FUNCTION get_due_amount(c_id INT) RETURNS INT AS $$
DECLARE
    due1 INT;
BEGIN
    SELECT due INTO due1 FROM transaction WHERE cart_id = c_id;
    RETURN due1;
END;
$$ LANGUAGE plpgsql;

-- Example call
SELECT get_due_amount(1);

DO $$
DECLARE
    product_id INT;
    product_name VARCHAR(20);
    product_stock INT;
    p_product CURSOR FOR SELECT pid, pname, p_stock FROM product;
BEGIN
    OPEN p_product;
    LOOP
        FETCH p_product INTO product_id, product_name, product_stock;
        EXIT WHEN NOT FOUND;
        RAISE NOTICE '% % %', product_id, product_name, product_stock;
    END LOOP;
    CLOSE p_product;
END;
$$;

CREATE OR REPLACE PROCEDURE check_stock(p_id INT) LANGUAGE plpgsql AS $$
DECLARE
    stock INT;
BEGIN
    SELECT p_stock INTO stock FROM product WHERE pid = p_id;
    IF stock < 2 THEN
        RAISE NOTICE 'Stock is Less';
    ELSE
        RAISE NOTICE 'Enough Stock';
    END IF;
END;
$$;

-- Example call
CALL check_stock(2);
```

CONCLUSION :

The project successfully implements a robust backend for an Inventory Management System using PostgreSQL. By defining multiple interrelated tables such as `brands`, `categories`, `stores`, and `product`, the system ensures seamless management of inventory, customer details, and transactions. Functions and procedures, such as `get_due_amount` and `check_stock`, enhance operational efficiency by automating essential tasks like checking due payments and managing stock levels. The system is scalable and designed to handle various inventory processes, from generating invoices to monitoring stock and ensuring timely reorders, thus optimizing the business flow.

This project highlights the versatility and strength of relational databases in handling real-world applications, ensuring data integrity, efficiency, and ease of use.