INVENTORY DATABASE MANAGEMENT SYSTEM

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 never the out-of-stock stage for any inventory.

SCOPE:

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

WORKING:

- This application will have different front ends for different kinds of users. The person who is sitting on the billing counter will have access to only modify the quantity of any product i.e. he/she can either generate an invoice for any sold product or can generate a return note for any returns from any customer.

- The manager will have the access to modify the rates if there exist any dynamic price inventory. The owner of the firm will have the access to generate the final report which will be consisting 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 & Material Requirement Planning

GOALS OF PROPOSED SYSTEM:

- **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 retrieval of information as well as its storage.
- **Accuracy:** The level of accuracy in the proposed system will be higher. All operations would
- **Reliability: -** The reliability of the proposed system will be high due to the above stated reasons. The reason for the increased reliability of the system is that now there would be proper storage of information.
- **No Redundancy: -** In the proposed system utmost care would be that no information is repeated anywhere, in storage or otherwise. This would assure economic use of storage space and consistency in the data stored.

- **Immediate retrieval of information: -** The main objective of proposed system is to provide for a quick and efficient retrieval of information.
- **Immediate storage of information: -** In manual system there are many problems to store the largest amount of information.
- **Easy to Operate**: **-** The system should be easy to operate and should be such that it can be developed within a short period of time and fit in the limited budget of the user.

BACKGROUND:

- This application is nowadays a basic use of any company, firm, shop or departmental store because stock maintenance, stock forecasting are some things which are very essential these days for earning great profits. In ancient times we need to maintain the complete inventory in paper pen method. The ancient method is quite un-easy, uncomfortable and some times inaccurate. For overcoming this problem we came with a solution of inventory management system. From this system we can generate invoice for each and every purchase. In addition to this we can have the employee details, customer details in this system. In short we can call this as all in one system..!!

USER CHARACTERISTICS:

- Every user should be:
- Comfortable with computer.
- Should have knowledge about web browser.
- Having a basic knowledge of English language is must.

TECHNICAL FEASIBILITY:

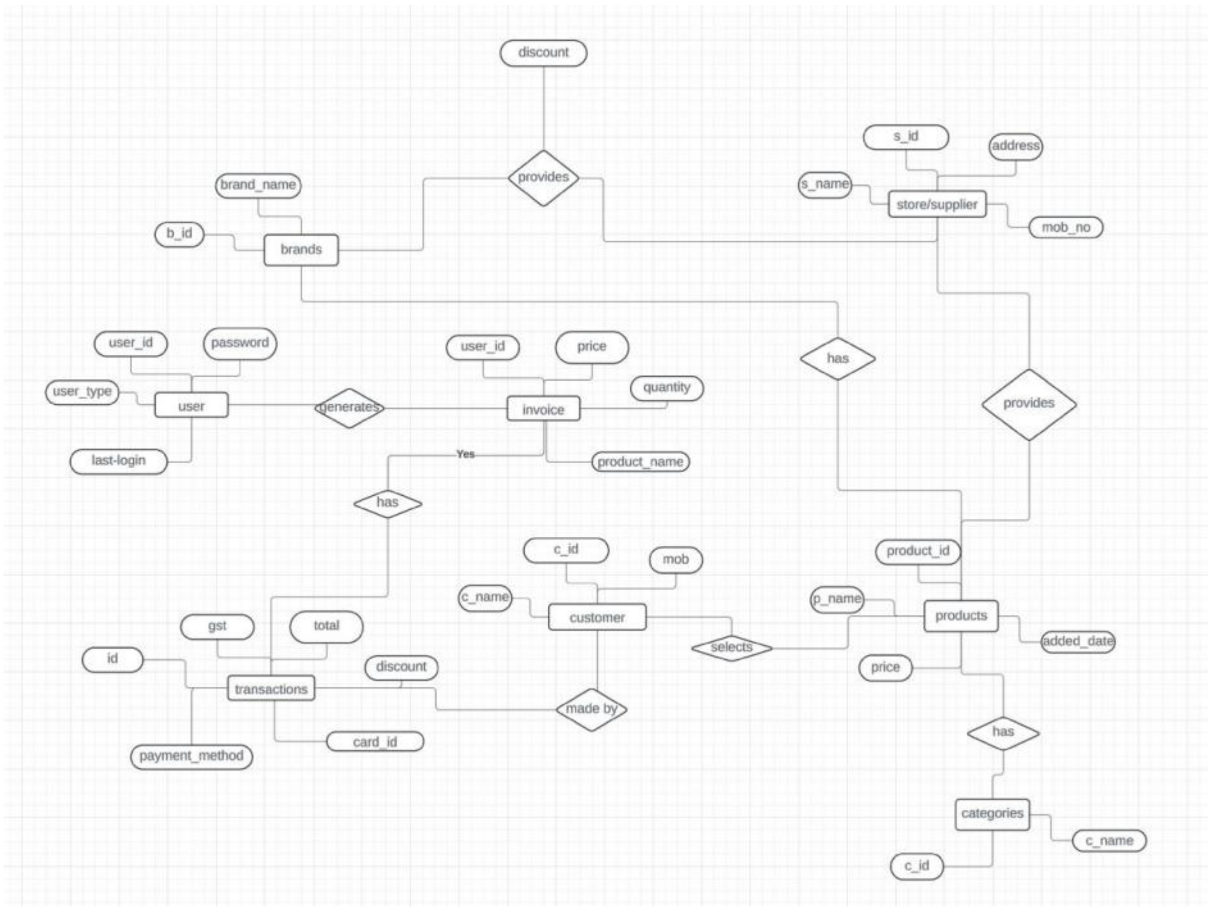
- In this project we've only implemented the back end of the system which is designed on "SQL Plus".
- On this sequence query language, we created 10 tables, whose name you will find in the sql code attached.

ADVANTAGES:

- **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.
- **Inventory Turnover:** Need to keep 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 see where you can make better use of your resources.
- **Warehouse Organization:** If we know which products are your top sellers and what combinations of products your customers often order together, you can optimize your warehouse setup by putting those products close together and in easily accessible places. This speeds up the picking, packing and shipping processes.
- **Repeat Customers:** Good inventory management leads to what every business owner wants – repeat customers. You want your hard-earned customers to keep coming back to your business to meet their needs. One way to do this is to make sure you have what they're looking for every time they come.
- **Accurate Planning:** Using smart inventory management, you can stay ahead of the demand curve, keep the right amount of products on hand and plan ahead for seasonal changes. This goes back to keeping your customers happy all year long.

- **Inventory Orders:** If you’ve done a good job keeping track of how much inventory you have on hand, you can make smarter decisions about when and what to order. Inventory management software lets you speed up the ordering process. You can simply scan a product barcode and type in some information to place an order and generate an invoice.
- **Inventory Tracking:** If you have multiple locations, then inventory management becomes even more important because you need to coordinate your supplies at each location depending on differences in demand and other factors.
- **Employee Efficiency:** We can empower your employees to help you manage inventory. Training employees to use barcode scanners, inventory management software and other tools helps them make better use of their time, and it helps your business make better use of its resources, both human and technological.

ER MODEL:



SUMMARY:

In this project we developed a complete back-end software in which we can update the stock, modify stock, we can forecast the stock, generate invoice. From this application we can get an update that if a particular inventory or stock is less than the sum pre-fixed quantity then it'll be easy for the manager/owner to reorder the product from supplier to overcome the "Out of Stock" stage. In addition to this it can also help us to manage the warehouses, add warehouses which can be proved as very useful feature. We can have complete customer details which can help us to retrieve the order details of regular customers. From this program we can also keep a track of transactions performed by different customers/clients. We can also get an idea that how much fund we received from different payment methodologies.

This application will keep a high inventory turnover ratio to ensure our products aren't spoiling, becoming obsolete for our working capital. It'll help us to calculate how many times inventory sells in a year and see where we can make better use of our resources.

<u>SQL CODE IMPLEMENTATION:</u>

CREATE TABLE :

```
SQL> create table brands(
2 bid number(5),
3 bname varchar(20)
4 );
Table created.
SQL> alter table brands
2 add primary
key(bid); Table altered.
SQL> create table inv_user(
2 user_id varchar(20),
3 name varchar(20),
4 password varchar(20),
5 last_login timestamp,
6 user_type varchar(10)
7 );
Table created.
SQL> create table categories(
2 cid number(5),
3 category_name varchar(20)
4 );
Table created.
```

```
SQL> alter table categories
2 add primary
key(cid); Table altered.
SQL> alter table inv_user
2 add primary key(user_id);
Table altered.
SQL> create table product(
2 pid number(5) primary key,
3 cid number(5) references categories(cid),
4 bid number(5) references brands(bid),
5 sid number(5),
6 pname varchar(20),
7 p_stock number(5),
8 price number(5),
9 added_date date);
Table created.
SQL> create table stores(
2 sid number(5),
3 sname varchar(20),
4 address varchar(20),
5 mobno number(10)
6 );
Table created.
SQL> alter table stores
2 add primary key(sid);
Table altered.
SQL> alter table product
```

```
2 add foreign key(sid)references stores(sid);
Table altered.
SQL> create table provides(
2 bid number(5)references brands(bid),
3 sid number(5)references stores(sid),
4 discount number(5));
Table created.
SQL> create table customer_cart(
2 cust_id number(5) primary key,
3 name varchar(20),
4 mobno number(10)
5 );
Table created.
SQL> create table select_product(
2 cust_id number(5) references customer_cart(cust_id),
3 pid number(5)references product(pid),
4 quantity number(4)
5 );
Table created.
SQL> create table transaction(
2 id number(5) primary key,
3 total_amount number(5),
4 paid number(5),
5 due number(5),
6 gst number(3),
7 discount number(5),
8 payment_method varchar(10),
9 cart_id number(5) references customer_cart(cust_id)
10 );
Table created.
SQL> create table invoice(
2 item_no number(5),
3 product_name varchar(20),
4 quantity number(5),
5 net_price number(5),
6 transaction_id number(5)references transaction(id)
7 );
INSERTION:
INSERT INTO BRANDS:
SQL> insert into brands values(
2 '&bid'
3 ,
4 ’&bname’);
Enter value for bid: 1
old 2: '&bid'
new 2: '1'
Enter value for bname: Apple
old 4: '&bname')
new 4: 'Apple')
1 row created.
1 row created.
SQL> insert into brands values(2,’Samsung’);
1 row created.
SQL> insert into brands values(3,’Nike’);
```

```
1 row created.

SQL> insert into brands values(4,'Fortune');

1 row created.

INSERT INTO INV_USER:

SQL> insert into inv_user values(

2 '&user_id',

3 '&name',

4 '&password',

5 '&last_login',

6 '&user_type');

Enter value for user_id: vidit@gmail.com

old 2: '&user_id',

new 2: 'vidit@gmail.com',

Enter value for name: vidit

old 3: '&name',

new 3: 'vidit',

Enter value for password: 1234

old 4: '&password',

new 4: '1234',

Enter value for last_login: 31-oct-18 12:40

old 5: '&last_login',

new 5: '31-oct-18 12:40',

Enter value for user_type: admin

old 6: '&user_type')

new 6: 'admin')

1 row created.

SQL> insert into inv_user values('harsh@gmail.com','Harsh
Khanelwal','1111','30-oct
```

```
18 10:20’,’Manager’);

1 row created.

SQL> insert into inv_user values('prashant@gmail.com','Prashant','0011','29-
oct-18

10:20’,’Accountant’);

1 row created.
```

INSERT INTO CATEGORIES:

```
SQL> insert into categories values(

2 '&cid',

3 ’&category_name’);

Enter value for cid: 1

old 2: '&cid',

new  2: '1',

Enter value for category_name: Electroincs

old  3: '&category_name')

new  3: 'Electroincs')

1 row created.

SQL> insert into categories values(2,’Clothing’);

1 row created.

SQL> insert into categories values(3,’Grocey’);

1 row created.
```

INSERT INTO STORE

```
SQL> insert into stores values(

2 '&sid',

3 '&sname',

4 '&address',

5 ’&mobno’);

Enter value for sid:
```

```
1 old 2: '&sid',
new  2: '1',
Enter value for sname: Ram kumar
old  3: '&sname',
new  3: 'Ram kumar',
Enter value for address: Katpadi vellore
old  4: '&address',
new  4: 'Katpadi vellore',
Enter value for mobno: 9999999999
old  5: '&mobno')
new 5: '9999999999')
1 row created.
SQL> insert into stores values(2,’Rakesh kumar’,’chennai’,8888555541);
1 row created.
SQL> insert into stores values(3,’Suraj’,’Haryana’,7777555541);
1 row created.
INSERT INTO PRODUCT:
SQL> insert into product values(
2 '&pid',
3 '&cid',
4 '&bid',
5 '&sid',
6 '&pname',
7 '&p_stock',
8 '&price',
9 ’&added_date’);
Enter value for pid: 1
```

```
old 2: '&pid',
new  2: '1',
Enter value for cid: 1
old  3: '&cid',
new  3: '1',
Enter value for bid: 1
old  4: '&bid',
new  4: '1',
Enter value for sid: 1
old  5: '&sid',
new  5: '1',
Enter value for pname: IPHONE
old  6: '&pname',
new  6: 'IPHONE',
Enter value for p_stock: 4
old  7: '&p_stock',
new  7: '4',
Enter value for price: 45000
old  8: '&price',
new  8: '45000',
Enter value for added_date: 31-oct-18
old  9: '&added_date')
new 9: '31-oct-18')
1 row created.
SQL> insert into product values(2,1,1,1,'Airpods',3,19000,'27-oct
18’); 1 row created.
SQL> insert into product values(3,1,1,1,'Smart Watch',3,19000,'27-oct-18’);
```

```
1 row created.
SQL> insert into product values(4,2,3,2,'Air Max',6,7000,'27-oct-18’);
1 row created.
SQL> insert into product values(5,3,4,3,'REFINED OIL',6,750,'25-oct-18’);
1 row created.
INSERT INTO PROVIDES:
SQL> insert into provides values(1,1,12);
1 row created.
SQL> insert into provides values(2,2,7);
1 row created.
SQL> insert into provides values(3,3,15);
1 row created.
SQL> insert into provides values(1,2,7);
1 row created.
SQL> insert into provides values(4,2,19);
1 row created.
SQL> insert into provides values(4,3,20);
1 row created.
INSERT INTO CUSTOMER_CART:
SQL> insert into customer_cart values(
2 '&cust_id',
3 '&name',
4 ’&mobno’);
Enter value for cust_id: 1
old 2: '&cust_id',
new 2: '1',
Enter value for name: Ram
```

```
old 3: '&name',
new 3: 'Ram',
Enter value for mobno: 9876543210
old 4: '&mobno')
new 4: '9876543210')
1 row created.
SQL> insert into customer_cart values(2,’Shyam’,7777777777);
1 row created.
SQL> insert into customer_cart values(3,’Mohan’,7777777775);
1 row created.
INSERT INTO SELECT_PRODUCT:
SQL> insert into select_product values(
2 '&cust_id',
3 '&pid',
4 ’&quantity’);
Enter value for cust_id: 1
old 2: '&cust_id',
new 2: '1',
Enter value for pid: 2
old 3: '&pid',
new 3: '2',
Enter value for quantity: 2
old 4: '&quantity')
new 4: '2')
1 row created.
SQL> insert into select_product values(1,3,1);
1 row created.
```

```
SQL> insert into select_product values(2,3,3);
1 row created.
SQL> insert into select_product values(3,2,1);
1 row created.
```

INSERT INTO TRANSACTIONS:

```
SQL> insert into transaction values(
2 '&id',
3 '&total_amount',
4 '&paid',
5 '&due',
6 '&gst',
7 '&discount',
8 '&payment_method',
9 ’&cart_id’);
Enter value for id: 1
old 2: '&id',
new 2: '1',
Enter value for total_amount: 57000
old 3: '&total_amount',
new 3: '25000',
Enter value for paid: 2000
old 4: '&paid',
new 4: '20000',
Enter value for due: 5000
old 5: '&due',
new 5: '5000',
Enter value for gst: 350
```

```
old 6: '&gst',
new 6: '350',
Enter value for discount: 350
old 7: '&discount',
new 7: '350',
Enter value for payment_method: card
old 8: '&payment_method',
new 8: 'card',
Enter value for cart_id: 1
old 9: '&cart_id')
new 9: '1')
1 row created.
insert into transaction values(2,57000,57000,0,570,570,’cash’,2);
SQL> insert into transaction values(3,19000,17000,2000,190,190,’cash’,3);
1 row created. SQL> insert into transaction
values(3,19000,17000,2000,190,190,’cash’,3);
1 row created.
```

PL/SQL

Functions:

```
SQL> declare
2 due1 number(7);
3 cart_id1 number(7);
4 function get_cart(c_id number)return number is
5 begin
6 return (c_id);
7 end;
8 begin
```

```
9 cart_id1:=get_cart(’&c_id’);
10 select due into due1 from transaction where cart_id=cart_id1;
11 dbms_output.put_line(due1);
12 end;
13 /
Enter value for c_id: 1
old 9: cart_id1:=get_cart(’&c_id’);
new 9: cart_id1:=get_cart(’1’);
5000
PL/SQL procedure successfully completed.
```

Cursors:

```
SQL> DECLARE
2 p_id product.pid%type;
3 p_name product.pname%type;
4 p_stock product.p_stock%type;
5 cursor p_product is
6 select pid,pname ,p_stock from product;
7 begin
8 open p_product;
9 loop
10 fetch p_product into p_id,p_name,p_stock;
11 exit when p_product%notfound;
12 dbms_output.put_line(p_id||’ ’||p_name||’ ’||p_stock);
13 end loop;
14 close p_product;
15 end;
16 /
```

```
1 IPHONE 4
2 Airpods 3
3 Smart Watch 3
4 Air Max 6
5 REFINED OIL 6
PL/SQL procedure successfully completed.
```

Procedure:

```
SQL> DECLARE
2 a number;
3 b number;
4 PROCEDURE check_stock(x IN number) IS
5 BEGIN
6 IF x < 2 THEN
7
dbms_output.put_line(’Stock is Less’);
8 ELSE
9
dbms_output.put_line('Enough
Stock’); 10 END IF;
11 END;
12 BEGIN
13   b:=’&b’;
14 select p_stock into a from product where pid=b;
15 check_stock(a);
16 END;
17 /
Enter value for b: 2
```

```
old 13: b:=’&b’;
new 13: b:=’2’;
Enough Stock
PL/SQL procedure successfully completed.
```