Inventory Management System

Database Management Project

Name	***Roll Number***
Bhavya Jain	102003676
Ishita Nagpal	102003683
Swati Saran	102003672
Sakshi	102183023

Submitted to

Dr Geeta Kasana

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

MAY 2022

INDEX

Problem Statement

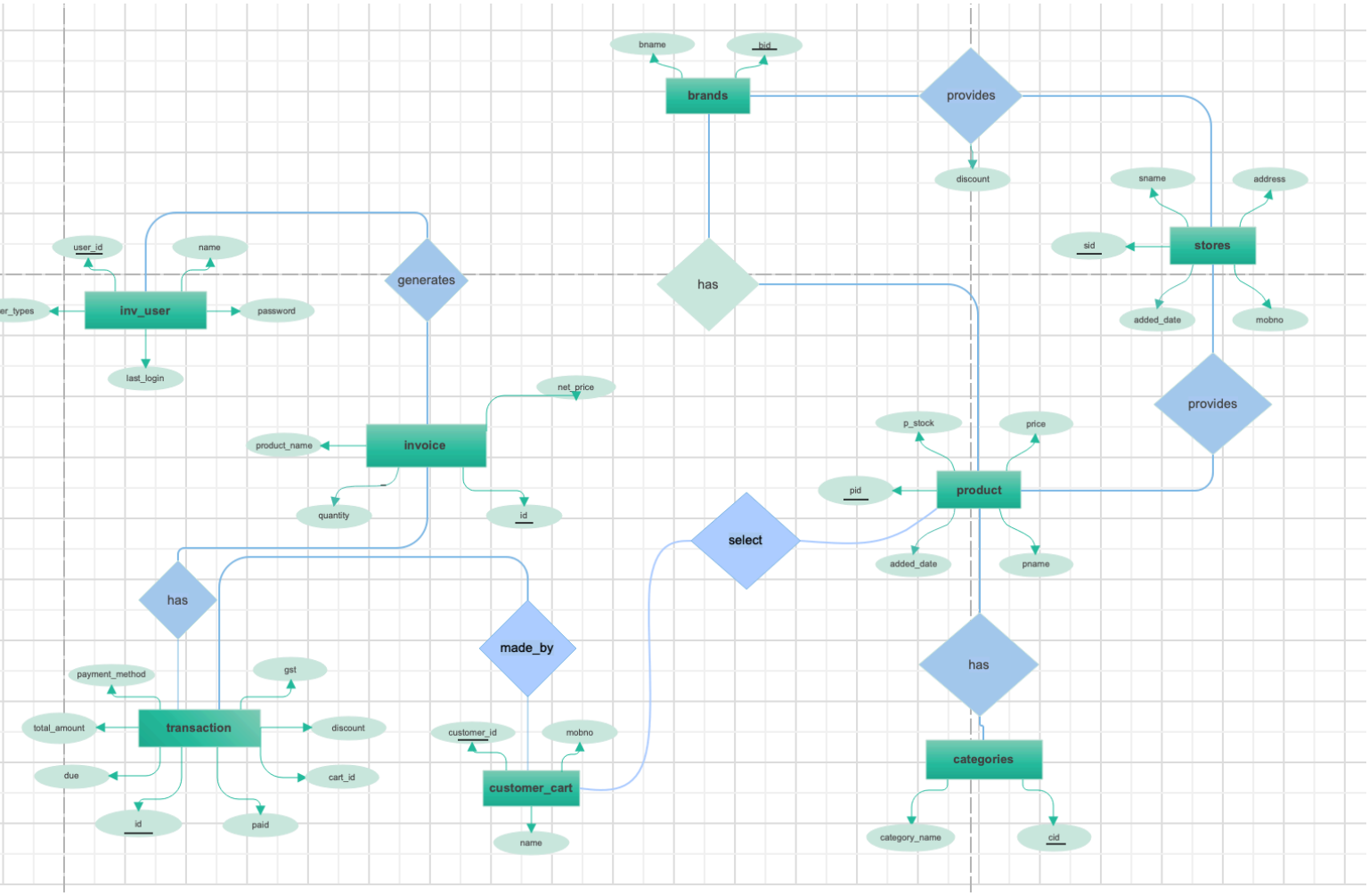
The Inventory Management system provides information to efficiently manage the flow of materials, effectively utilise people and equipment, coordinate internal activities and communicate with customers. Inventory Management does not make decisions or manage operations they provide the information to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

Every enterprise needs inventory for the smooth running of its activities. It serves as a link between the production and distribution process. The unforeseen fluctuation in demand and supply of goods also necessitates the need for inventory. It also provides a cushion for future price fluctuations. The purpose of inventory management is to ensure the availability of materials of insufficient quality and quantity as and when required and also to minimize investment in inventories. Thus it is very essential to have proper control and management of inventory. Inventories play a vital role in the operation of the real estate industry. The inventory ensures operational smoothness. In almost all the organizations a substantial part of capital is invested in inventories. Inventory refers to the stock of products of a firm that is for sale and also the components that make up the product.

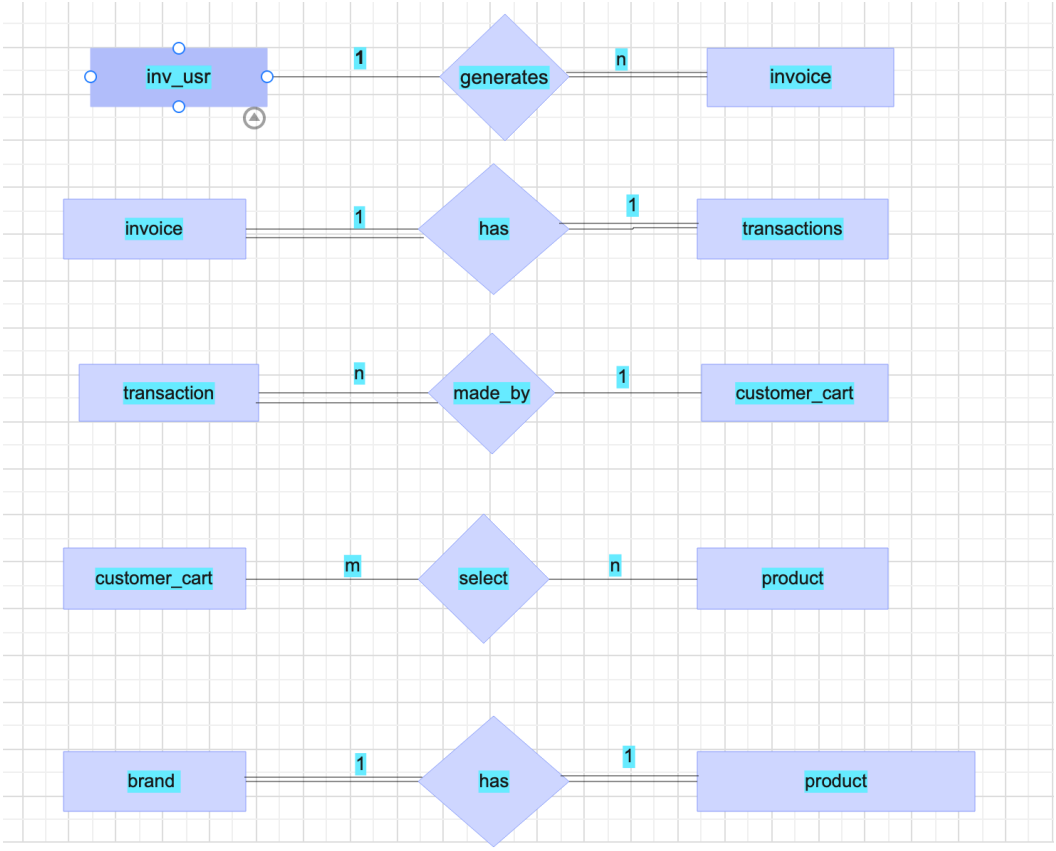
OBJECTIVES OF INVENTORY MANAGEMENT:

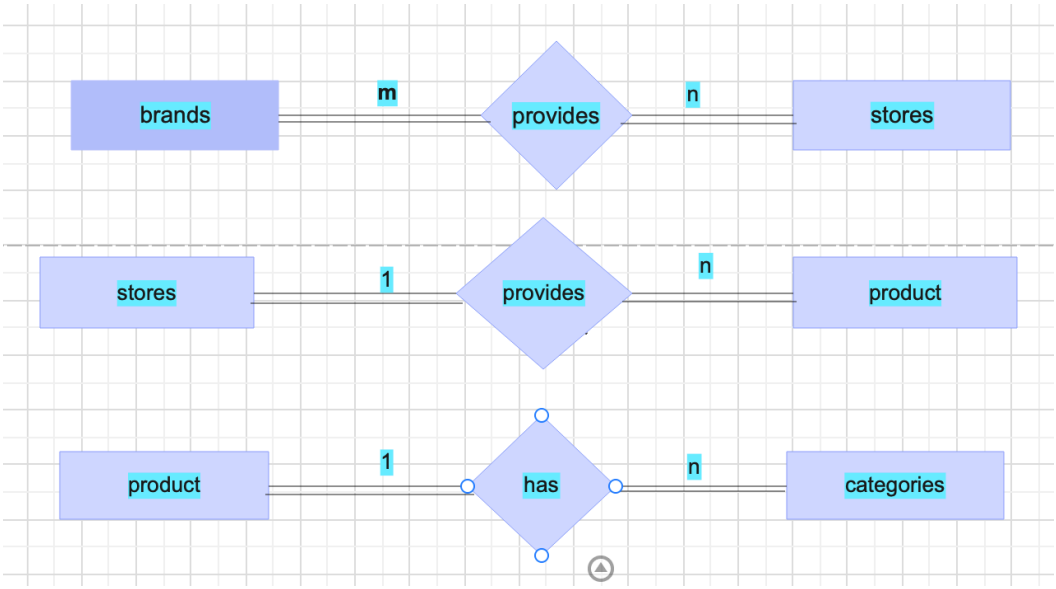
- To have stock available as and when required.
- To maintain adequate accounting and understanding of inventory.
- To avoid both overstocking and understocking of inventory.
- To ensure the right quality goods at a reasonable price.
- To design proper organization for inventory management.
- To bring down the inventory carrying cost.
- To facilitate furnishing of data for short term and long term planning and control of inventory.
- To avoid duplication in ordering or replenishing stocks.
- To facilitate purchasing economies.
- To decide which item to stock and which item to procure on demand.

ER DIAGRAM :


bname
bid
brands
provides
discount
sname
address
sid
stores
added_date
mobno
user_id
name
generates
has
inv_user
password
last_login
net_price
product_name
invoice
quantity
id
provides
p_stock
price
pid
product
select
added_date
pname
has
made_by
has
payment_method
gst
total_amount
transaction
discount
due
cart_id
id
paid
customer_id
mobno
customer_cart
name
categories
category_name
cid

CARDINALITY:





Above are the cardinalities and relationship where
1:1 -One to One relationship
1:n- One to many relationship
n:1 - Many to One relationship
m:n - Many to One relationship
Double line- total relationship
Single line - partial relationship

Er to tables:

- inv_user(user_id,name,password,last_login,user_type)
- invoice(**id**,**user_id**,quantity,net_price,product_name)
- transaction(id,**cust_id**,payment_method,total_amount,due,paid,gst,discount,cart_id)
- customer_cart(customer_id,mobno,name)
- select(**pid**,**cust_id**)
- product(pid,**bid**,**sid**,added_date,pname,price,p_stock)
- brands(bid,baname)
- provides(**bid**,**sid**,discount)
- categories(cid,**pid**,category_name)

Normalized Tables:

- brands(bid,bname)
- inv_user(user_id, name, password, last_login,user_type)
- categories(cid,category_name)
- product(**cid**,**bid**,**sid**,pid,pname,p_stock,price,add_date)
- stores(sid,sname,mobno,address)
- provides(**bid**,**sid**,discount)
- customer_cart(customer_id,name,mobno)
- select_product(**cust_id**,**pid**,quantity)
- transaction(id,total_amount,paid,due,discount,gst,payment_method,**cart_id**)
- invoice(item_no,product_name,quantity,net_price,**transaction_id**)

In the above, normalised tables underlined attributes refer to primary keys whereas the bold attributes refer to foreign keys.

CODE :

CREATION OF TABLES:-

```
create table brands(
 bid number(5),
 bname varchar(20)
 );

alter table brands add primary key(bid);
```

```
Table BRANDS created.

Table BRANDS altered.
```

```
create table inv_user(
user_id varchar(20),
name varchar(20),
password varchar(20),
last_login timestamp,
user_type varchar(10)
);
alter table inv_user add primary key(user_id);
```

```
Table INV_USER created.

Table INV_USER altered.
```

```
create table categories(
cid number(5),
category_name varchar(20)
);
```

```
alter table categories add primary key(cid);
```

```
Table CATEGORIES created.

Table CATEGORIES altered.
```

```
create table product(
pid number(5) primary key,
cid number(5) references categories(cid),
bid number(5) references brands(bid),
sid number(5),
pname varchar(20),
p_stock number(5),
price number(5),
added_date date);
```

```
alter table product
 add foreign key(sid)references stores(sid);
```

```
Table PRODUCT created.

Table PRODUCT altered.
```

```
create table stores(
 sid number(5),
 sname varchar(20),
 address varchar(20),
 mobno number(10)
 );
```

```
alter table stores add primary key(sid);
```

```
Table STORES created.

Table STORES altered.
```

```
create table provides(
 bid number(5)references brands(bid),
 sid number(5)references stores(sid),
 discount number(5));
```

```
Table PROVIDES created.
```

```
create table customer_cart(
 cust_id number(5) primary key,
 name varchar(20),
 mobno number(10)
 );
```

```
Table CUSTOMER_CART created.
```

```
create table select_product(
 cust_id number(5) references customer_cart(cust_id),
 pid number(5)references product(pid),
 quantity number(4)
 );
```

```
Table SELECT_PRODUCT created.
```

```
create table transaction(
 id number(5) primary key,
 total_amount number(5),
 paid number(5),
 due number(5),
 gst number(3),
 discount number(5),
 payment_method varchar(10),
 cart_id number(5) references customer_cart(cust_id)
 );
```

```
Table TRANSACTION created.
```

```
create table invoice(
item_no number(5),
product_name varchar(20),
quantity number(5),
net_price number(5),
transaction_id number(5)references transaction(id));
```

```
Table INVOICE created.
```

INSERTION INTO TABLES:-

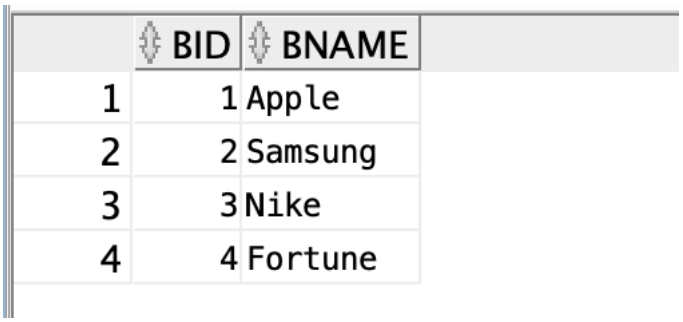
insert into brands values('&bid' ,'&bname');

insert into brands values(2,'Samsung');

insert into brands values(3,'Nike');

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

select * from brands;



	BID	BNAME
1	1	Apple
2	2	Samsung
3	3	Nike
4	4	Fortune

insert into inv_user values('&user_id', '&name', '&password','&last_login', '&user_type');

insert into inv_user values('harsh@gmail.com','Harsh Khanelwal','1111','30-oct-18 10:20','Manager');

insert into inv_user values('prashant@gmail.com','Prashant','0011','29-oct-1810:20','Accountant');

select * from inv_user;

	USER_ID	NAME	PASSWORD	LAST_LOGIN	USER_TYPE
1	ishita@gmail.com	ishita	1234	31-10-22 12:40:00.000000000 PM	admin
2	harsh@gmail.com	Harsh Khanelwal	1111	30-10-18 10:20:00.000000000 AM	Manager
3	prashant@gmail.com	Prashant	0011	29-10-18 10:20:00.000000000 AM	Accountant

insert into categories values('&cid','&category_name');

insert into categories values(2,'Clothing');

insert into categories values(3,'Grocey');

select * from categories;

	CID	CATEGORY_NAME
1	1	Electronics
2	2	Clothing
3	3	Grocey

insert into stores values('&sid','&sname','&address', '&mobno');

insert into stores values(2,'Rakesh kumar','chennai',8888555541);

insert into stores values(3,'Suraj','Haryana',7777555541);

select * from stores;

	SID	SNAME	ADDRESS	MOBNO
1	1	Ram	Chandigarh	9999999999
2	2	Rakesh kumar	chennai	8888555541
3	3	Suraj	Haryana	7777555541

- insert into product values('&pid', '&cid', '&bid', '&sid', '&pname', '&p_stock', '&price','&added_date');
- insert into product values(2,1,1,1,'Airpods',3,19000,'27-oct18');
- insert into product values(3,1,1,1,'Smart Watch',3,19000,'27-oct-18');
- insert into product values(4,2,3,2,'Air Max',6,7000,'27-oct-18');
- insert into product values(5,3,4,3,'REFINED OIL',6,750,'25-oct-18');
- select * from product;

	PID	CID	BID	SID	PNAME	P_STOCK	PRICE	ADDED_DATE
1	1	1	1	1	IPhone	4	45000	31-10-22
2	2	1	1	1	Airpods	3	19000	27-10-18
3	3	1	1	1	Smart Watch	3	19000	27-10-18
4	4	2	3	2	Air Max	6	7000	27-10-18
5	5	3	4	3	REFINED OIL	6	750	25-10-18

insert into provides values(1,1,12);

insert into provides values(2,2,7);

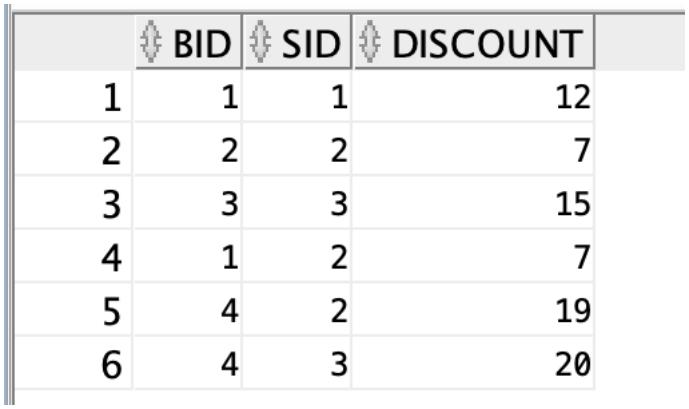
insert into provides values(3,3,15);

insert into provides values(1,2,7);

insert into provides values(4,2,19);

insert into provides values(4,3,20);

select * from provides;



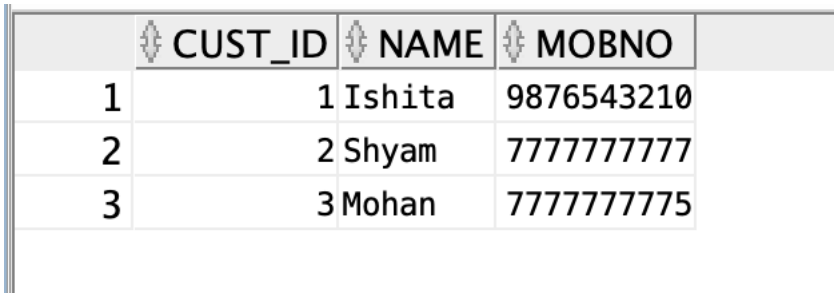
	BID	SID	DISCOUNT
1	1	1	12
2	2	2	7
3	3	3	15
4	1	2	7
5	4	2	19
6	4	3	20

insert into customer_cart values(‘&cust_id','&name','&mobno');

insert into customer_cart values(2,'Shyam',7777777777);

insert into customer_cart values(3,'Mohan',7777777775);

select * from customer_cart;



	CUST_ID	NAME	MOBNO
1	1	Ishita	9876543210
2	2	Shyam	7777777777
3	3	Mohan	7777777775

insert into select_product values('&cust_id','&pid','&quantity');

insert into select_product values(1,3,1);

insert into select_product values(2,3,3);

insert into select_product values(3,2,1);

select * from select_product;

	CUST_ID	PID	QUANTITY
1	1	2	2
2	1	3	1
3	2	3	3
4	3	2	1

- insert into transaction values('&id','&total_amount','&paid','&due','&gst','&discount', '&payment_method','&cart_id');
- insert into transaction values(2,57000,57000,0,570,570,'cash',2);
- insert into transaction values(3,19000,17000,2000,190,190,'cash',3);
- select * from transaction;

	ID	TOTAL_AMOUNT	PAID	DUE	GST	DISCOUNT	PAYMENT_METHOD	CART_ID
1	1	57000	2000	5000	350	350	card	1
2	2	57000	57000	0	570	570	cash	2
3	3	19000	17000	2000	190	190	cash	3

FUNCTION IMPLEMENTATION WITH PREDEFINED EXCEPTION :-

In this function we are implementing to view the due amount of a particular customer by entering the customer id from user

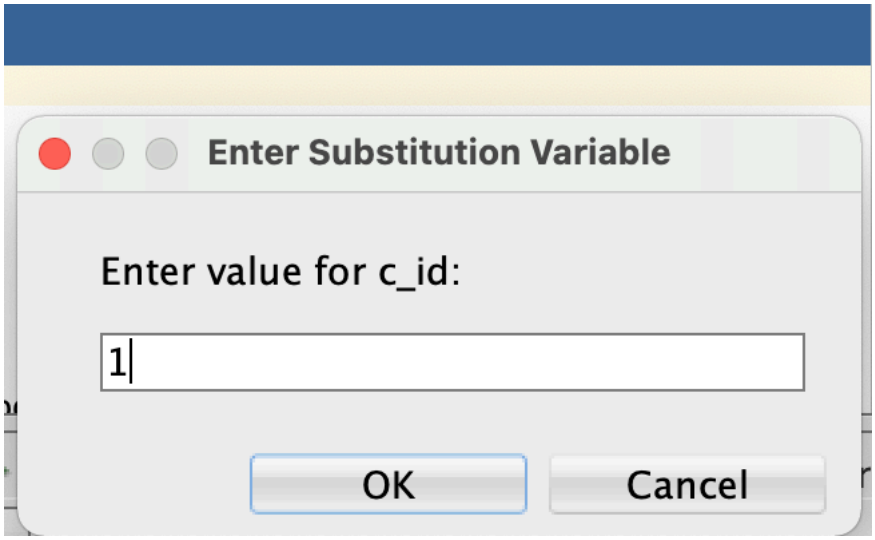
```
--FUNCTION IMPLEMENTATION
declare
due1 number(7);
```

```
cart_id1 number(7);
function get_cart(c_id number)return number is
due_final transaction.due%type;
begin
select due into due_final from transaction where   cart_id=c_id;
return (due_final);
end;
begin
cart_id1:=&c_id;
due1:=get_cart(cart_id1);
dbms_output.put_line('Due amount is '||due1||' for the customer id
'||cart_id1);
EXCEPTION
WHEN NO_DATA_FOUND THEN
dbms_output.put_line('No Record found with the given pid');
WHEN VALUE_ERROR THEN
dbms_output.put_line('invalid input as conversion of character
string to number fails');
WHEN TOO_MANY_ROWS THEN
dbms_output.put_line('select into statement returns too many
rows');
WHEN ZERO_DIVIDE THEN
dbms_output.put_line('An attempt is made to divide a number by
0');
```

WHEN OTHERS THEN

dbms_output.put_line('some error occured');

end;




```
end;
Due amount is 5000 for the customer id 1
```

CURSOR IMPLEMENTATION:-

--METHOD 1

DECLARE

p_id product.pid%type;

p_name product.pname%type;

p_stock product.p_stock%type;

cursor p_product is

select pid,pname ,p_stock from product;

begin

open p_product;

```
dbms_output.put_line('The productid,product_name and quantity
of products present in stock are :');
loop
fetch p_product into p_id,p_name,p_stock;
exit when p_product%notfound;
dbms_output.put_line(p_id||' '||p_name||' '||p_stock);
end loop;
close p_product;
end;
```

--METHOD 2

```
DECLARE
cursor c1 is select pid,pname,p_stock from product;
BEGIN
dbms_output.put_line('The productid,product name and quantity
of products present in stock are :');
for rec in c1 loop
dbms_output.put_line(rec.pid||' '||rec.pname||' '||rec.p_stock);
end loop;
end;
```

```
The product id , product name and quantity of products present in stock are :
1 IPhone 4
2 Airpods 3
3 Smart Watch 3
4 Air Max 6
5 REFINED OIL 6
```

PROCEDURE IMPLEMENTATION:-

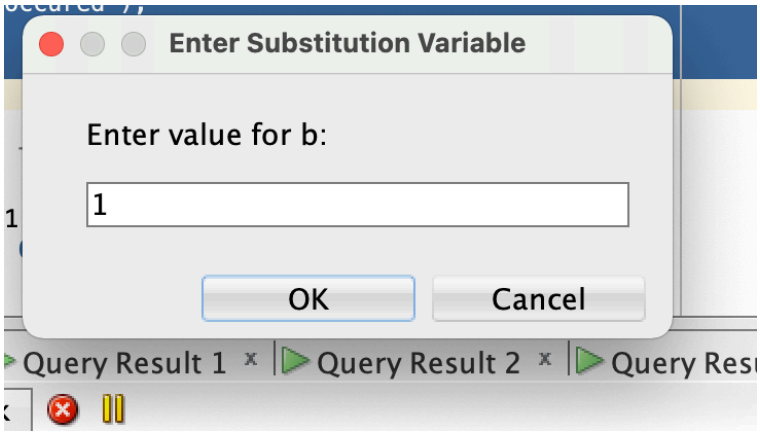
The procedure checks that the whether the availabity of

A certain product is enough or not.

PROCEDURE IMPLEMENTATION WITH EXCEPTION

```
DECLARE
a number;
b product.pid%type;
LARGE_VALUE exception;
pragma exception_init(LARGE_VALUE,-01438);
PROCEDURE check_stock(x IN number) IS
BEGIN
IF x < 2 THEN
dbms_output.put_line('Stock is Less as the quantity in stock is'||
x);
ELSE
dbms_output.put_line('Enough Stock as the quantity in stock is
'||.x);
END IF;
END;
BEGIN
b:='&b';
select p_stock into a from product where pid=b;
check_stock(a);
EXCEPTION
```

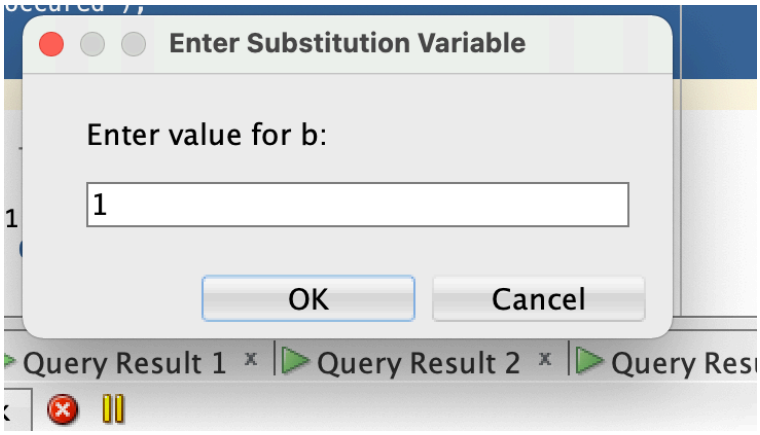
```
    WHEN LARGE_VALUE THEN
    dbms_output.put_line('input is too large for the column');
    WHEN NO_DATA_FOUND THEN
    dbms_output.put_line('No Record found with the given pid');
    WHEN VALUE_ERROR THEN
    dbms_output.put_line('invalid input as conversion of character string to number fails');
    WHEN OTHERS THEN
    dbms_output.put_line('some error occured');
    END;
```




```
END;
Enough Stock as the quantity in stock is 4
```

IMPLEMENTING THE ABOVE PROCEDURE WITH EXPLICIT EXCEPTIONS

```
DECLARE
out_of_stock EXCEPTION;
rec product.p_stock%type;
BEGIN
select p_stock into rec from product where pid=&pid;
if rec<2 then
RAISE out_of_stock;
END IF;
dbms_output.put_line('The quantity of item in stock is '||rec);
EXCEPTION
WHEN out_of_stock THEN
dbms_output.put_line('The item is out of stock with quantity '||
rec);
END;
```




```
END;
The quantity of item in stock is 4
```

TRIGGERS IMPLEMENTATION 1:-

--IMPLEMENTING TRIGGERS HERE WE ARE BASICALLY INSERTING

-- THE MOBILE NUMBERS OF STORES WE CAN EVEN UPDATE THE -

--EXISTING MOBILE NUMBERS

```
CREATE OR REPLACE TRIGGER trigger1
BEFORE INSERT OR UPDATE OF mobno
ON stores
FOR EACH ROW
BEGIN
   dbms_output.put_line('Old mobile number: ' || :OLD.mobno);
   dbms_output.put_line('New mobile number: ' || :NEW.mobno);

END;

--TRIGGERING trigger1 IN STORE TABLE
select * from stores;
INSERT INTO stores (sid, sname, address, mobno)
VALUES (4,'Ramesh','Patna',8362946291);
select * from stores;
```

```
Trigger TRIGGER1 compiled
```

```
Old mobile number:
New mobile number: 8362946291


1 row inserted
```

	SID	SNAME	ADDRESS	MOBNO
1	4	Ramesh	Patna	8362946291
2	5	Ramesh	Patna	8362946291

TRIGGERS IMPLEMENTATION 2:-

--UPDATING OR INSERTING PAYMENT METHOD IN TRANSACTION

--TABLE AFTER THE TRIGGERING ACTION IS PERFORMED

```
CREATE OR REPLACE TRIGGER trigger2
AFTER INSERT OR UPDATE OF  payment_method ON transaction
FOR EACH ROW
BEGIN
  dbms_output.put_line('Old payment_method: ' || :OLD.payment_method);
  dbms_output.put_line('New payment_method: '
|| :NEW.payment_method);
END;
```

--TRIGGERING trigger2 IN TRANSACTION TABLE

```
select * from transaction;
INSERT INTO transaction VALUES (4,20000,13000,2000,140,140,'card',3);
select * from transaction;
UPDATE transaction SET payment_method='PAYTM' where id=4;
select * from transaction;
```

```
Trigger TRIGGER2 compiled
```

```
Old payment_method: CARD
New payment_method: PAYTM
```

	ID	TOTAL_AMOUNT	PAID	DUE	GST	DISCOUNT	PAYMENT_MET...	CART_ID
1	2	57000	57000	0	570	570	cash	2
2	3	19000	17000	2000	190	190	cash	3
3	4	20000	13000	2000	140	140	PAYTM	3

SUMMARY :

Efficient inventory management of inventory helps the owners, but also customers, suppliers, government and society. To increase the profits it is necessary to reduce the inventory cost. The study has been undertaken with the objectives such as inventory control system, purchase procedure and evaluation of purchase procedure. Hence inventory management proves to be an asset to any organisation.

CONCLUSION :

To conclude the store's department is concerned with the receiving of materials, storing and issuing to the production department. The company using this inventory management system is consuming various categories of materials there is a balance between procurement and consumption. The inventory management system helps a particular organisation to create and sink as to what is happening in the production, purchase cell and reduce the chances of out of stock products, missing products and missing of records regarding a particular product.