A Practical activity Report submitted for

UCS310

Database Management System

Sumbitted by


**Hardik Verma** **102283044**

**Namit Nayyar** **102103831**

**Komal** **102283042**

**Bhaavya Bangotra** **102103783**


**Submitted to**
**Ms. Simran**

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

**THAPAR INSTITUTE OF ENGINEERING AND TECHNOLOGY, (A DEEMED TO BE UNIVERSITY), PATIALA, PUNJAB**

# TABLE OF CONTENTS

# INTRODUCTION

In this modern era of online shopping, no seller wants to be left behind, moreover due to its simplicity the shift from an offline selling model to an online selling model is witnessing rampant growth.
Therefore, as an engineer, our job is to ease the path of this transition for the seller. Amongst many things that an online site requires the most important is a database system. Hence in this project, we are planning to design a database where small clothing sellers can sell their products online.
The basic function of our database is to digitally store information that can be captured, retrieved, and distributed easily at a later time. This also gives businesses the ability to analyze and track information about the products, sales, and customers that have been input into the database.
One of the biggest benefits of using a database for e-commerce is the addition of structure to vast amounts of shop data. No matter how big or small your online store is, an infrastructure is needed for all the gathered information to make sense and provide useful insights. When the data is structured, it can be accessed more efficiently by the e-commerce application.
Myntra offers a well-curated comprehensive selection of clothes from different brands. It is a one-stop shop for all the fashion and lifestyle needs. Myntra aims at providing a hassle-free and enjoyable shopping experience to shoppers across the country with the widest range of brands and products on its portal. It serves consumers through its retail websites with a focus on selection, price, and convenience. In this project, we are making a conscious effort to bring the power of

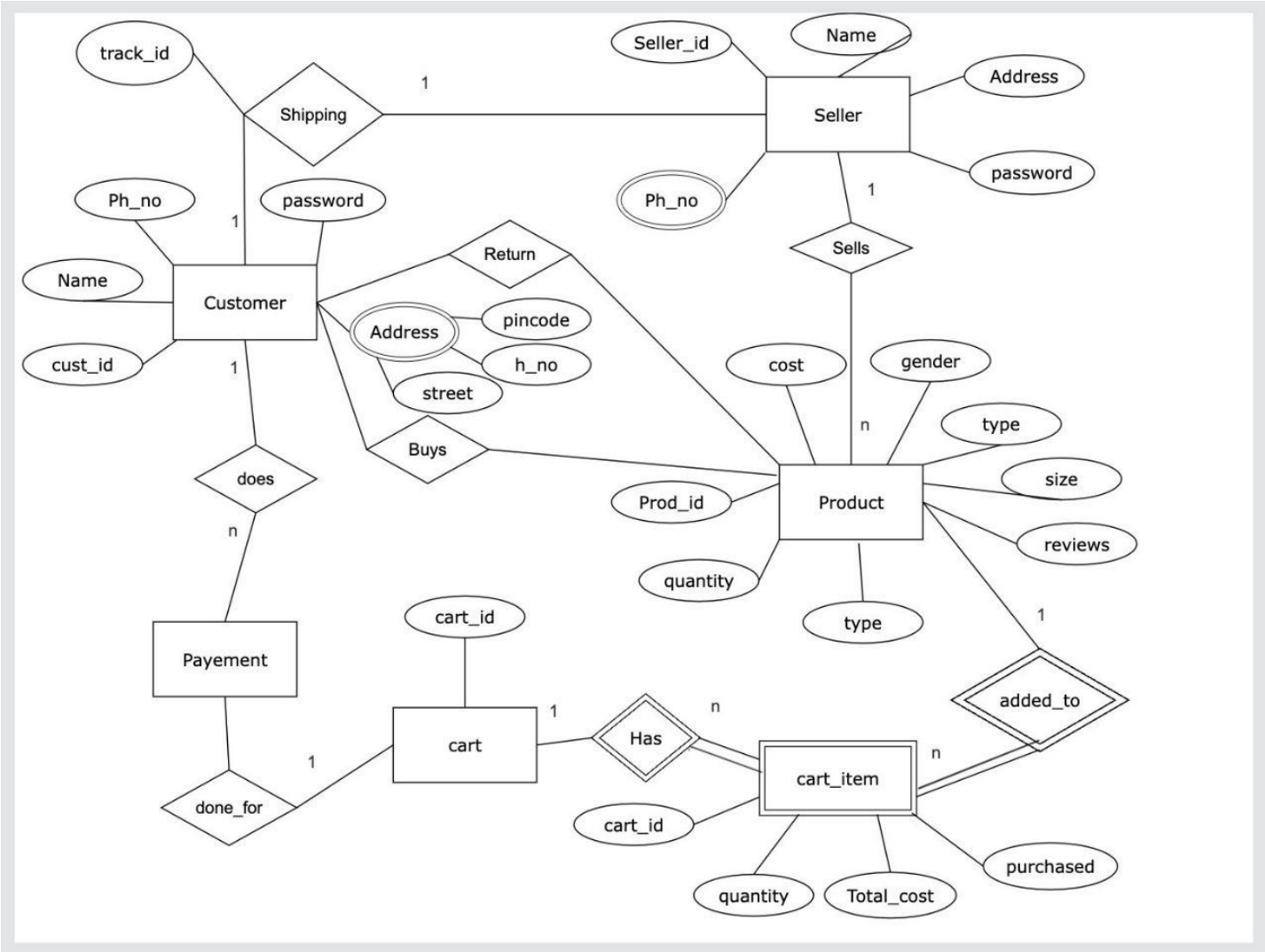
fashion to shoppers with an array of the latest and trendiest products available in the country.

## ER DIAGRAM DESCRIPTION

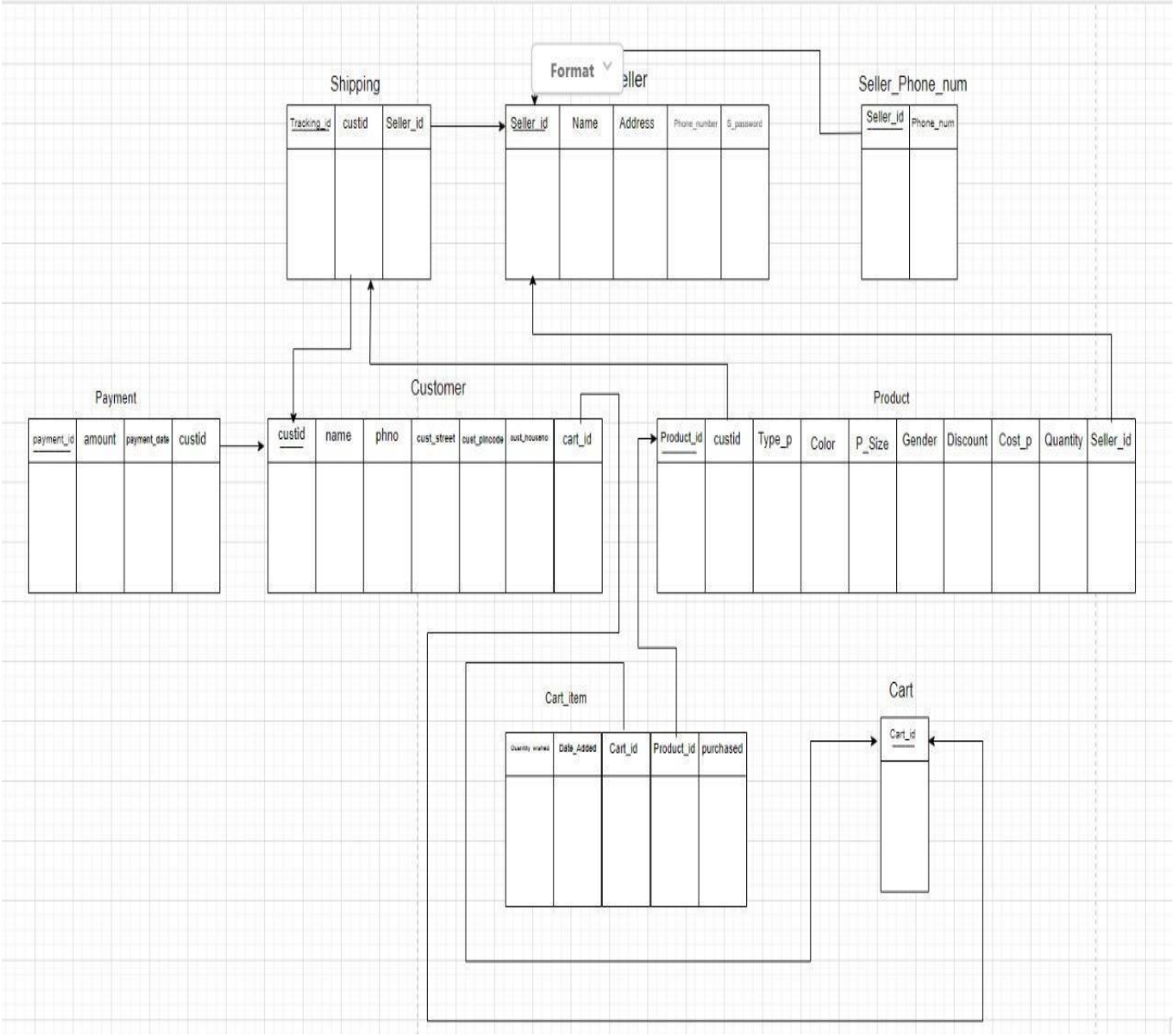
These shopping websites enable many sellers to sell their products to customers at various locations. Each seller has the following attribute: seller_id,s_pass,Name_s and address. Each product has the following attributes: Product_id, Type_p, color,p_size, gender, discount,cost_p and quantity. The customer makes the payment for the cart items added to the cart. Each customer has a cart with its unique cart_id. Cart items have the following attributes: Quantity_wished, purchased,date_added.

After the product is bought the sellers ship product to customers. The customer has the following attributes: name,phone_no.,custid and address. If the customer doesn't like the product he may request a return of the product.

# ER diagram


track_id
Shipping
1
Seller_id
Name
Seller
Address
password
Ph_no
1
Ph_no
password
1
Sells
Return
Name
Customer
Address
pincode
h_no
street
cust_id
1
cost
gender
n
type
Buys
does
Prod_id
Product
size
reviews
n
quantity
type
1
cart_id
Payement
added_to
1
n
cart
Has
n
1
cart_item
done_for
cart_id
purchased
quantity
Total_cost

# ER to Table


Shipping
Tracking_id
custid
Seller_id
Seller
Seller_id
Name
Address
Phone_number
S_password
Seller_Phone_num
Seller_id
Phone_num
Payment
payment_id
amount
payment_date
custid
Customer
custid
name
phno
cust_street
cust_pincode
cust_houseno
cart_id
Product
Product_id
custid
Type_p
Color
P_Size
Gender
Discount
Cost_p
Quantity
Seller_id
Cart_item
Date_Added
Cart_id
Product_id
purchased
Cart
Cart_id

# Normalization

**1NF:** Since our er has ph_no as multivalued attribute so we have made a separate table for ph_no.

**2NF:** Since there is no non-primary key attribute in the tables that is partially dependent on the primary key, they are normalized according to 2NF.

**3NF:** Since no non-key attribute is transitively dependent on the primary key in the tables, they are normalized according to 3NF.

**BCNF:** Since every determinant is the primary key in the tables, they are normalized according to BCNF.

## CREATE TABLE COMMANDS:

```
CREATE TABLE
  Seller (
     Seller_id VARCHAR(6) ,
     s_pass VARCHAR(10) ,
     Name_s VARCHAR(20) ,
     Shop_number NUMBER,
     Street VARCHAR(30) ,
     Pincode NUMBER,
     PRIMARY KEY
     (Seller_id)
  );
```

```
  CREATE TABLE
  Seller_Phone_num (
     Phone_num NUMBER(10),

     Seller_id VARCHAR(6)REFERENCES Seller(Seller_id) ON
DELETE CASCADE,
     PRIMARY KEY (Phone_num, Seller_id)
  );
```

```
CREATE TABLE
  Cart (
     Cart_id VARCHAR(7) NOT NULL,
     CONSTRAINT pk_cart PRIMARY
     KEY(Cart_id)
  );
```

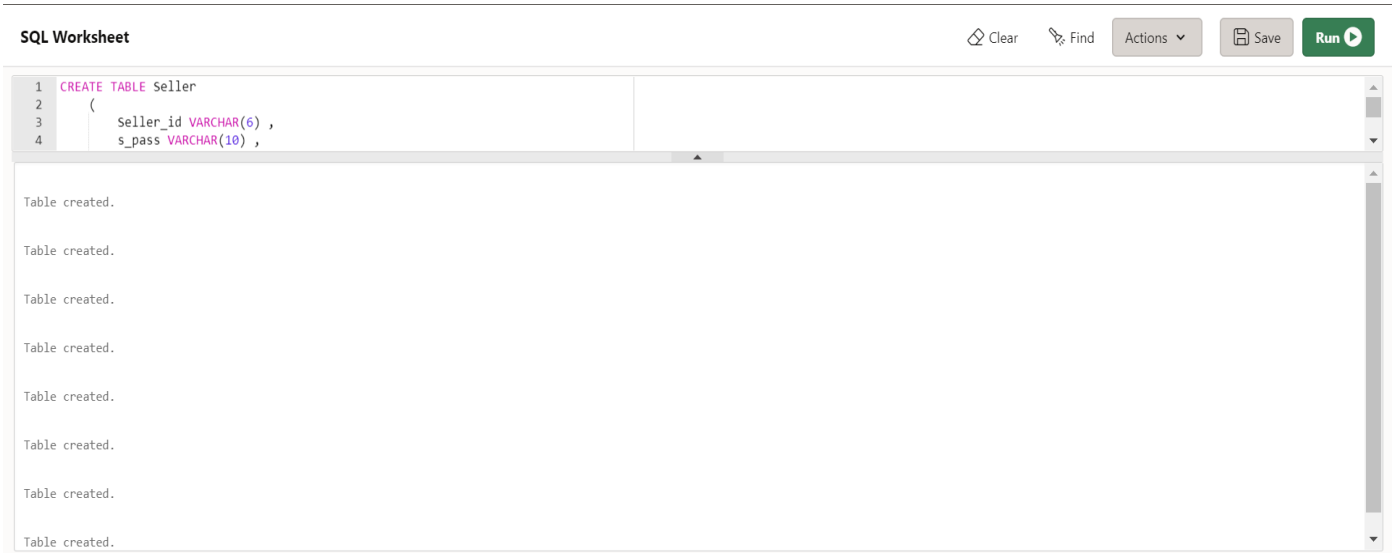
```
CREATE TABLE customer(
name VARCHAR(20),
phno NUMBER(10),
custid VARCHAR(12) PRIMARY KEY,
cust_houseno
NUMBER(10),
cust_street
VARCHAR(40),
cust_pincode
NUMBER(10),
cart_id VARCHAR(7) REFERENCES cart(cart_id));
```

```
CREATE TABLE
    SHIPPING (
      Tracking_id VARCHAR (13) PRIMARY KEY,
      custid VARCHAR(12) REFERENCES Customer(custid),
      Seller_id VARCHAR(6)REFERENCES Seller(Seller_id) ON DELETE
SET NULL
      );
```

```
CREATE TABLE payment(
payment_id VARCHAR(20) PRIMARY KEY,
amount NUMBER(10),
payment_date DATE,
custid VARCHAR(12) REFERENCES customer(custid));
```

```
CREATE TABLE Product
  (
     Product_id VARCHAR(7) NOT NULL,
     custid VARCHAR(12),
     Type_p VARCHAR(7) NOT NULL,
     Color VARCHAR(15) NOT NULL,
     P_Size VARCHAR(2) NOT NULL,
     Gender CHAR(1) NOT NULL,
     Discount NUMBER(2) NOT NULL,
     Cost_p NUMBER(5) NOT NULL,
     Quantity NUMBER(2) NOT NULL,
     Seller_id VARCHAR(6),
     PRIMARY KEY (Product_id),
     FOREIGN KEY (Seller_id) REFERENCES Seller(Seller_id) ON
DELETE SET NULL,
     FOREIGN KEY (custid) REFERENCES customer(custid)
     ON DELETE SET NULL
  );
```

```
CREATE TABLE Cart_item
 (
    Quantity_wished NUMBER(1) NOT NULL,
    Date_Added DATE NOT NULL,
    Cart_id VARCHAR(7) NOT NULL,
    Product_id VARCHAR(7) NOT NULL,
    purchased VARCHAR(5),
    CONSTRAINT fk_cart FOREIGN KEY (Cart_id) REFERENCES
Cart(Cart_id),
    CONSTRAINT fk_product FOREIGN KEY (Product_id)
REFERENCES Product(Product_id),
    CONSTRAINT pk_cart_item Primary key(Cart_id,Product_id)
 );
```



**INSERTION COMMANDS:**

```
insert into Seller values('1ABQ36','#123','Namit',1470,'Rajamata Street',110003);
insert into Seller values('1CFQ36','#456','Hardik',2134,'Ram Laal Chowk',110005);
insert into Seller values('26KL74','#ABCD','Komal',73,'Munshi Chowk',400004);
insert into Seller values('17KLP7','#R$23','Bhavyal',677,'Sahibzada Nagar',140307);
insert into Seller values('23EFD5','a345d','Hardik',523,'Nawa kot Nagar',132103);
```

```
insert into Seller_Phone_num values(9873654321,'26KL74');
insert into Seller_Phone_num values(9813732561,'23EFD5');
insert into Seller_Phone_num values(9765310987,'26KL74');
insert into Seller_Phone_num values(9349845100,'1ABQ36');
insert into Seller_Phone_num values(8453877652,'26KL74');
insert into Seller_Phone_num values(8873654321,'1CFQ36');
insert into Seller_Phone_num values(7765654321,'17KLP7');
```

```
insert into Cart values('1ADF34');
insert into Cart values('2FDF54');
insert into Cart values('1QRT89');
insert into Cart values('5MPL24');
insert into Cart values('1HDF98');
```

insert into customer
values('Mansi',8862473245,'765FTYGUHUHB',1432,'Gandhi Nagar Street',147001,'2FDF54');
insert into customer
values('Ajay',9834567754,'765DBHBABDBB',234,'Sabzi Mandi Street',110864,'1ADF34');
insert into customer
values('Jitender',9865467356,'565FTYHYHUHB',75,'Ranjit Nagar',110086,'1QRT89');
insert into customer
values('Dhruv',7865473240,'765FTYGUHUUI',101,'Virat Nagar',147345,'5MPL24');
insert into customer
values('Arsh',9855327255,'865FTYGRHIHB',3465,'Model Town',132103,'1HDF98');

insert into SHIPPING values('7YIUGF','765FTYGUHUHB','26KL74'); insert into SHIPPING values('87HHKL','765DBHBABDBB','1ABQ36'); insert into SHIPPING values('34VHJF','565FTYHYHUHB','23EFD5'); insert into SHIPPING values('45GVCW','765FTYGUHUUI','1CFQ36'); insert into SHIPPING values('87JFKJ','865FTYGRHIHB','17KLP7');

```
insert into payment values('TEJBKD524VVR',5000,'23-DEC-
2023','765FTYGUHUHB');
insert into payment values('TYRFWD524VVR',500,'25-DEC-
2022','765DBHBABDBB');
insert into payment values('YSJBKD524VVR',2730,'12-SEP-
2020','565FTYHYHUHB');
insert into payment values('RFGBKD524VVR',420,'02-AUG-
2018','765FTYGUHUUI');
insert into payment values('TEYPOC524VVR',1190,'06-JUN-
2019','865FTYGRHIHB');
insert into payment values('TEJBKD545VVR',750,'25-OCT-
2015','565FTYHYHUHB');
```

```
insert into
Productvalues('1267543','765FTYGUHUHB','Sandal','Blue','6','F',20,
420,1,'26KL74');
insert into Product
values('2345676','765DBHBABDBB','Coat','Black','32','M',30,2730,1,
'23EFD5');
insert into Product
values('3647833','565FTYHYHUHB','Shoes','Green','9','M',25,5000,
2,'1ABQ36');
insert into Product
values('7384903','765FTYGUHUUI','Shirt','White','12','F',50,500,1,
'1CFQ36');
insert into Product
values('4567839','865FTYGRHIHB','Jeans','Blue','36','F',0,1190,1,'1
7KLP7');
```

```
insert into Cart_item values(1,'03-JUN-
2022','1ADF34','1267543','yes');
insert into Cart_item values(1,'23-AUG-
2021','2FDF54','2345676','no');
insert into Cart_item values(1,'06-JAN-
2020','1QRT89','3647833','yes');
insert into Cart_item values(1,'27-MAY-
2022','5MPL24','7384903','no');
insert into Cart_item values(1,'15-DEC-
2021','1HDF98','4567839','yes');
insert into Cart_item values(1,'17-DEC-
2021','1HDF98','2345676','no');
```

## QUERIES:

1) **Alter name_s as Firstname, length of firstname as 15 and add one more field of Last_name**

   Alter table Seller
   rename column Name_s to FirstName ;
   Alter table Seller
   modify FirstName varchar(15);
   Alter table Seller
   add LastName varchar(15);

**SQL Worksheet**

```
Alter table Seller
rename  column Name_s to FirstName ;
Alter table Seller
modify FirstName varchar(15);
Alter table Seller
add LastName varchar(15);
```

```
Table altered.
```

SQL Worksheet

```
1
2  desc seller;
```

TABLE SELLER

| Column | Null? | Type |
|---|---|---|
| SELLER_ID | NOT NULL | VARCHAR2(6) |
| S_PASS | - | VARCHAR2(10) |
| FIRSTNAME | - | VARCHAR2(15) |
| SHOP_NUMBER | - | NUMBER |
| STREET | - | VARCHAR2(30) |
| PINCODE | - | NUMBER |
| LASTNAME | - | VARCHAR2(15) |

Download CSV
7 rows selected.

2) **If a seller wants update his phone number**

update Seller_Phone_num set Phone_num=9765310988 where seller_id='26KL74' and Phone_num=9765310987;

Live SQL

SQL Worksheet

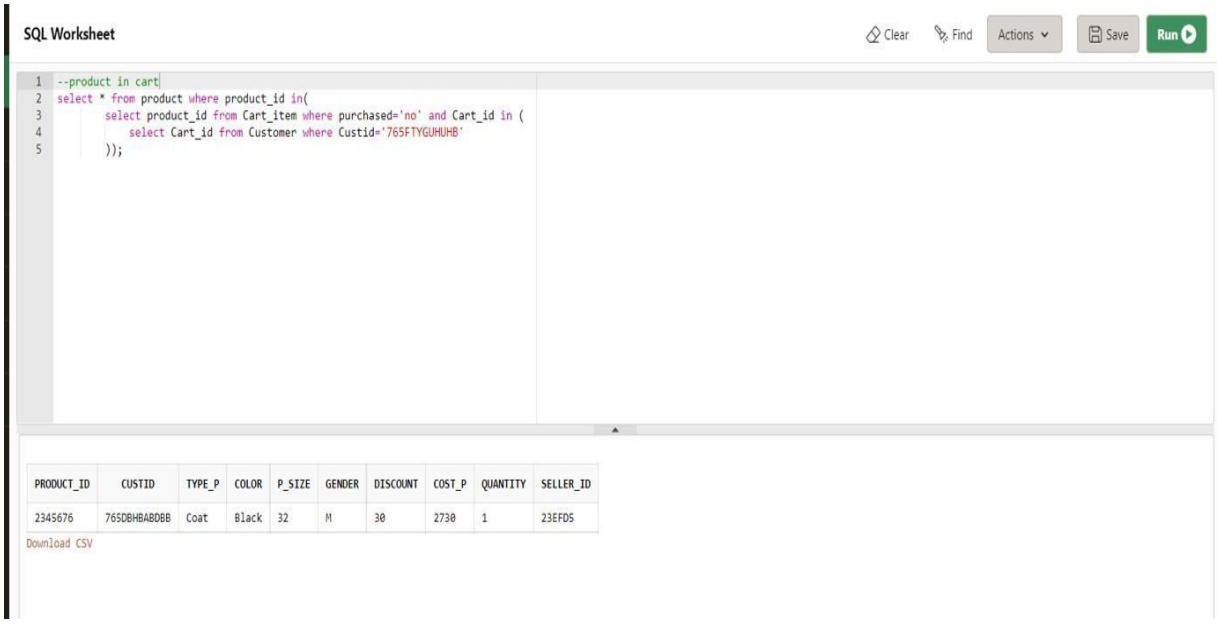
```
update Seller_Phone_num set Phone_num=9765310988 where seller_id='26KL74' and  Phone_num=9765310987;
select * from Seller_Phone_num;
```

1 row(s) updated.

| PHONE_NUM | SELLER_ID |
|---|---|
| 7765654321 | 17KLP7 |
| 8453877652 | 26KL74 |
| 8873654321 | 1CFQ36 |
| 9349845100 | 1ABQ36 |
| 9765310988 | 26KL74 |
| 9813732561 | 23EFD5 |
| 9873654321 | 26KL74 |

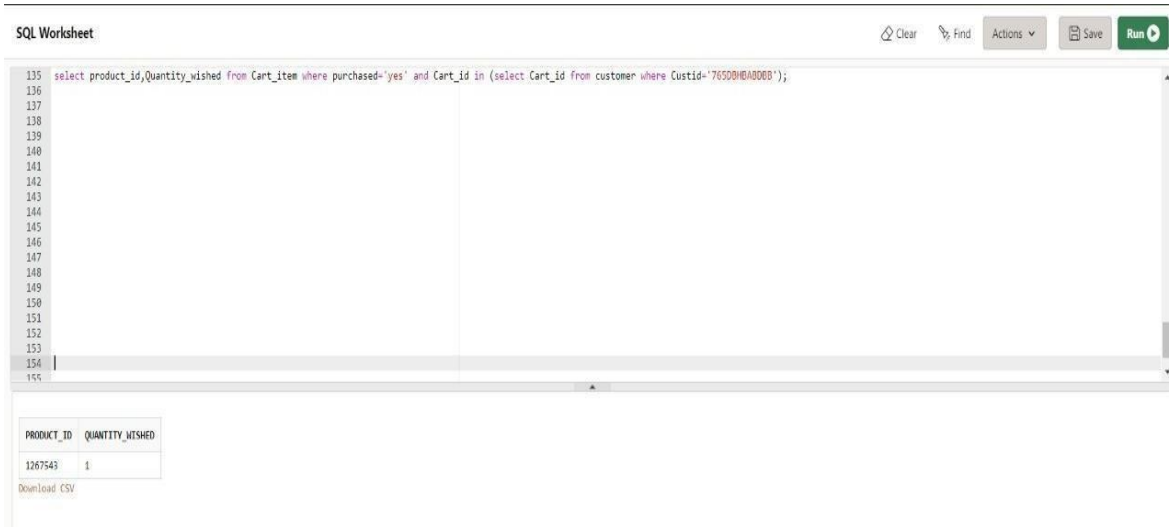
3) **If the customer wants to see details of the product present in the cart**

```
select * from product where product_id in(
    select product_id from Cart_item where purchased='no' and Cart_id in(
        Select Cart_id from Customer where custid='765FTYGUHUHB'
));
```




4) **If a customer wants to see order history**

select product_id,Quantity_wished from Cart_item where purchased='yes' and Cart_id in (select Cart_id from customer where Custid='765DBHBABDBB');



5) **Customer wants to see filtered products on the basis of size,gender,type**

select product_id, Color, Cost_p, Seller_id from product where Type_p='jeans' and P_size='36' and Gender='F' and Quantity>0);


6) **If customer wants to modify the cart**

Select * from Cart_item;

delete from cart_item where purchased='no'and product_id='2345676' and Cart_id in (select cart_id from Customer where Custid='865FTYGRHIHB'

Select * from Cart_item

SQL Worksheet

Clear

```
--modify cart
select * from Cart_item;
delete from Cart_item where purchased='no' and product_id='2345676' and Cart_id in (select cart_id from Customer where Custid='865FTYGRHIHB');
Select * from Cart_item;
```

| QUANTITY_WISHED | DATE_ADDED | CART_ID | PRODUCT_ID | PURCHASED |
|---|---|---|---|---|
| 1 | 17-DEC-21 | 1HDF98 | 2345676 | no |
| 1 | 03-JUN-22 | 1ADF34 | 1267543 | yes |
| 1 | 23-AUG-21 | 2FDF54 | 2345676 | no |
| 1 | 06-JAN-20 | 1QRT89 | 3647833 | yes |
| 1 | 27-MAY-22 | 5MPL24 | 7384903 | no |
| 1 | 15-DEC-21 | 1HDF98 | 4567839 | yes |

Download CSV
6 rows selected.

1 row(s) deleted.

| QUANTITY_WISHED | DATE_ADDED | CART_ID | PRODUCT_ID | PURCHASED |
|---|---|---|---|---|
| 1 | 03-JUN-22 | 1ADF34 | 1267543 | yes |
| 1 | 23-AUG-21 | 2FDF54 | 2345676 | no |
| 1 | 06-JAN-20 | 1QRT89 | 3647833 | yes |
| 1 | 27-MAY-22 | 5MPL24 | 7384903 | no |
| 1 | 15-DEC-21 | 1HDF98 | 4567839 | yes |

Download CSV
5 rows selected.

);

7) **If a seller stops selling his product**

delete from seller where seller_id = '26KL74';

update product set quantity = 00 where seller_id is NULL;

SQL Worksheet

Clear Find Actions Save Run

```
delete  from seller where seller_id = '26KL74';
    update product set quantity = 00 where seller_id is NULL;
```

1 row(s) deleted.

1 row(s) updated.

SQL Worksheet

```
--delete seller details
--delete  from seller where seller_id = '26KL74';
--update product set quantity = 00 where seller_id is NULL;
select * from seller;
select * from product;
```

| SELLER_ID | S_PASS | NAME_S | SHOP_NUMBER | STREET | PINCODE |
|---|---|---|---|---|---|
| 1ABQ36 | #123 | Ajitpal | 1470 | Rajamata Street | 110003 |
| 1CFQ36 | #456 | Raman | 2134 | Ram Laal Chowk | 110005 |
| 17KLP7 | #R$23 | Gopal | 677 | Sahibzada Nagar | 140307 |
| 23EFD5 | a345d | Sumita | 523 | Nawa kot Nagar | 132103 |

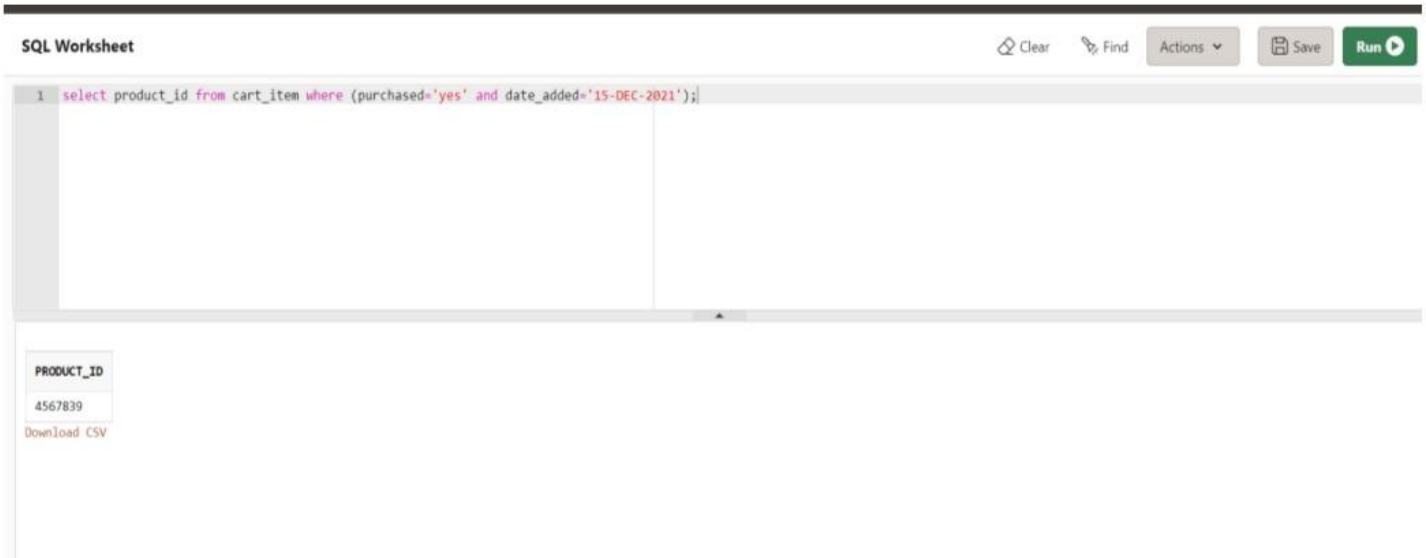
Download CSV
4 rows selected.

| PRODUCT_ID | CUSTID | TYPE_P | COLOR | P_SIZE | GENDER | DISCOUNT | COST_P | QUANTITY | SELLER_ID |
|---|---|---|---|---|---|---|---|---|---|
| 1267543 | 765FTYGUHUHB | Sandal | Blue | 6 | F | 20 | 420 | 0 | - |
| 2345676 | 765DBHBABDBB | Coat | Black | 32 | M | 30 | 2730 | 1 | 23EFD5 |
| 3647833 | 565FTYHYHUHB | Shoes | Green | 9 | M | 25 | 5000 | 2 | 1ABQ36 |
| 7384903 | 765FTYGUHUUI | Shirt | White | 12 | F | 50 | 500 | 1 | 1CFQ36 |
| 4567839 | 865FTYGRHIHB | Jeans | Blue | 36 | F | 0 | 1190 | 1 | 17KLP7 |

Download CSV
5 rows selected.

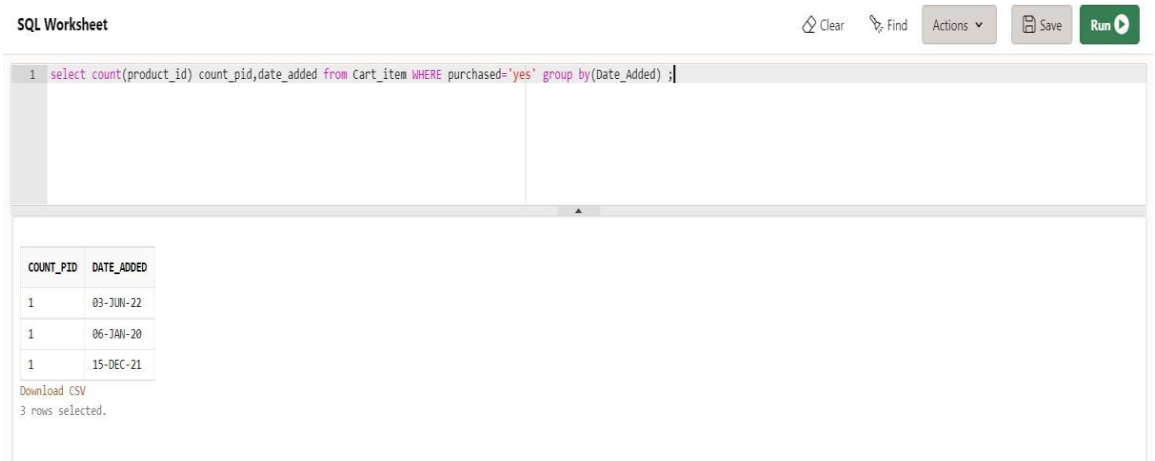
8) **If the admin wants to see what are the product purchased on the particular date**

select product_id from cart_item where (purchased='yes' and date_added='15-DEC-2021');



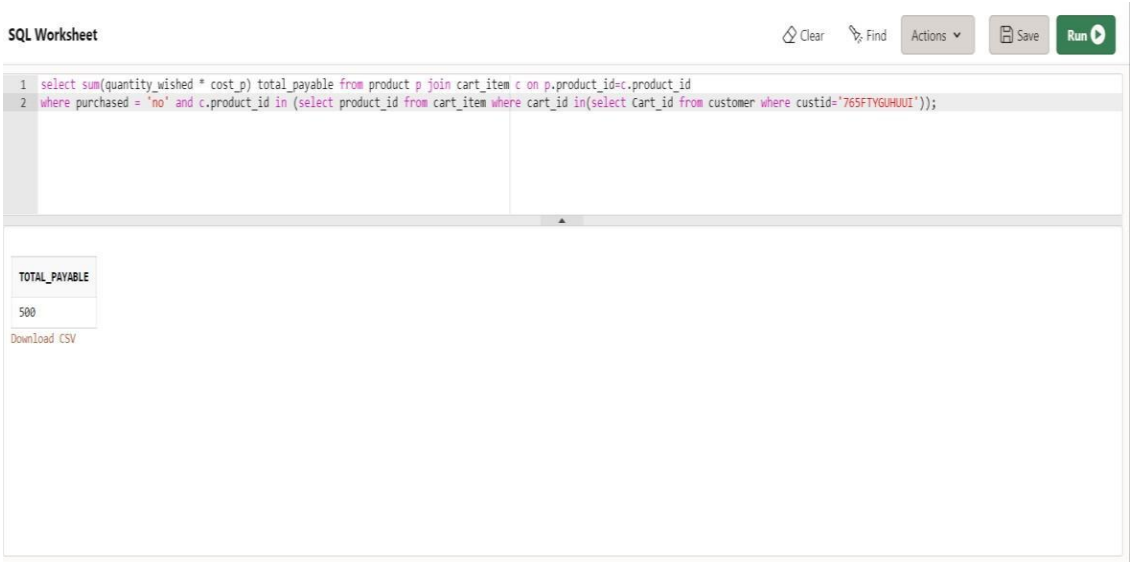

9) **How much product sold on the particular date**

select count(product_id) count_pid,date_added from Cart_item WHERE purchased='yes' group by(Date_Added) ;



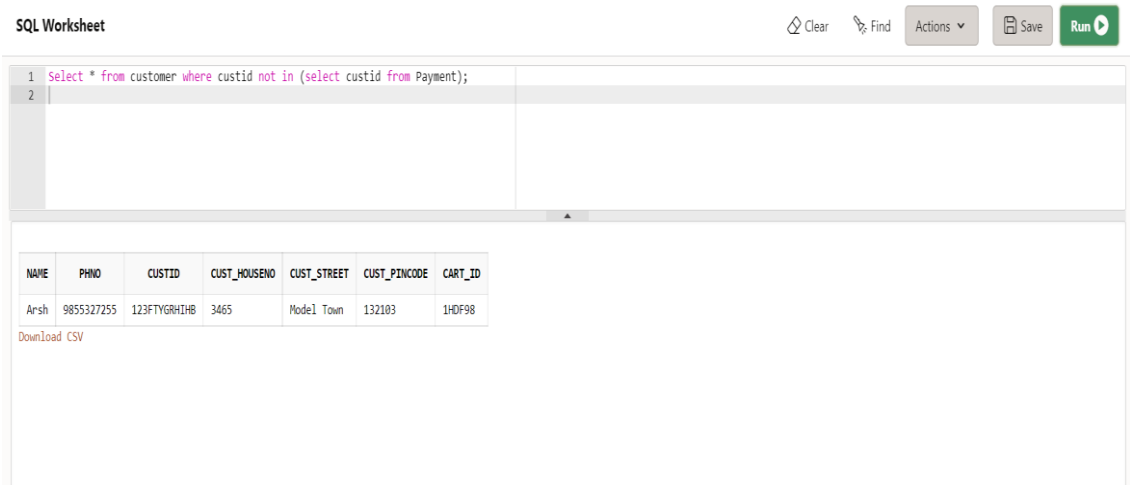

10) **If a customer want to know the total price present in the cart**

select sum(quantity_wished * cost_p) total_payable from product p join cart_item c on p.product_id=c.product_id
where purchased = 'no' and c.product_id in (select product_id from cart_item where cart_id in(select Cart_id from customer where custid='765FTYGUHUUI'));




11) **Show the details of the customer who has not purchased any thing**

Select * from customer where custid not in (select custid from Payment);




12) **Find total sales of a seller..**

```
select sum(quantity_wished * cost_p ) total_profit,seller_id
from product p join cart_item c on p.product_id=c.product_id
WHERE purchased='yes' group by seller_id;
```

SQL Worksheet

Clear Find Actions Save Run

```
select sum(quantity_wished * cost_p  ) total_profit,seller_id from product p join cart_item c on p.product_id=c.product_id WHERE purchased='yes' group by seller_id;
```

| TOTAL_PROFIT | SELLER_ID |
|---|---|
| 420 | 26KL74 |
| 5000 | 1ABQ36 |
| 1190 | 17KLP7 |

Download CSV

3 rows selected.

13) **Revenue generated by each seller on the particular date**

select name_s, seller_id, revenue from seller,

(select seller_id as s_id, sum(Cost_p*Quantity) as revenue from product group by seller_id)

where seller.seller_id = s_id

14) **Create a view to display the payment details for a particular date**

```
CREATE VIEW payment_details AS
SELECT payment_id, amount
FROM payment
WHERE payment_date = '23-DEC-2022';

select * from payment_details
```

**SQL Worksheet**

```
21 CREATE VIEW payment_details AS
22 SELECT payment_id, amount
23 FROM payment
24 WHERE payment_date = '23-DEC-2022';
25
26 select * from payment_details
```

| PAYMENT_ID | AMOUNT |
|---|---|
| TEJBKD524VVR | 5000 |

Download CSV

15) **Display the total purchase by each customer**

```
select name, custid, cost from customer,
(select custid as c_id, sum(Cost_p*Quantity) as cost from product group by custid)
where customer.custid = c_id
```

SQL Worksheet

Clear Find Actions Save Run

```
select name, custid, cost from customer,
(select custid as c_id, sum(Cost_p*Quantity) as cost from product group by custid)
where customer.custid = c_id
```

| NAME | CUSTID | COST |
|---|---|---|
| Dhruv | 765FTYGUHUUI | 500 |
| Mansi | 765FTYGUHUHB | 420 |
| Ajay | 765DBHBABDBB | 2730 |
| Jitender | 565FTYHYHUHB | 10000 |
| Arsh | 865FTYGRHIHB | 1190 |

Download CSV

5 rows selected.

16) **Create a view to display the payment details for today**

```
CREATE VIEW payment_details_today AS
SELECT payment_id, amount
 FROM payment
WHERE payment_date = sysdate;

select * from payment_details_today
```

SQL Worksheet

Clear Find Actions Save Run

```
CREATE VIEW payment_details_today AS
            SELECT payment_id, amount
            FROM payment
            WHERE payment_date = sysdate;

select * from payment_details_today
```

no data found

17) **Displaying all contact details for each seller**

```
SELECT Seller.NAME_S,Seller_Phone_num.PHONE_NUM
FROM Seller , Seller_Phone_num
WHERE Seller.SELLER_ID = Seller_Phone_num.SELLER_ID;
```

18) **Display details of seller and products sold by him** select * from seller,product where seller.Seller_id=Product.Seller_id;

SQL Worksheet

```
select * from seller,product where seller.Seller_id=Product.Seller_id;
```

| SELLER_ID | S_PASS | NAME_S | SHOP_NUMBER | STREET | PINCODE | PRODUCT_ID | CUSTID | TYPE_P | COLOR | P_SIZE | GENDER | DISCOUNT | COST_P | QUANTITY | SELLER_ID |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 26KL74 | #ABCD | Amrit | 73 | Munshi Chowk | 400004 | 1267543 | 765FTYGUHUHB | Sandal | Blue | 6 | F | 20 | 420 | 1 | 26KL74 |
| 23EFD5 | a345d | Sumita | 523 | Nawa kot Nagar | 132103 | 2345676 | 765DBHBABDBB | Coat | Black | 32 | M | 30 | 2730 | 1 | 23EFD5 |
| 1ABQ36 | #123 | Ajitpal | 1470 | Rajamata Street | 110003 | 3647833 | 565FTYHYHUHB | Shoes | Green | 9 | M | 25 | 5000 | 2 | 1ABQ36 |
| 1CFQ36 | #456 | Raman | 2134 | Ram Laal Chowk | 110005 | 7384903 | 765FTYGUHUUI | Shirt | White | 12 | F | 50 | 500 | 1 | 1CFQ36 |
| 17KLP7 | #R$23 | Gopal | 677 | Sahibzada Nagar | 140307 | 4567839 | 865FTYGRHIHB | Jeans | Blue | 36 | F | 0 | 1190 | 1 | 17KLP7 |

Download CSV

5 rows selected.

19) **Select ids of seller and customer who belong to same place** select seller_id,custid from Seller,customer where seller.pincode=customer.cust_pincode;

## SQL Worksheet

```
1  select seller_id,custid from Seller,customer where seller.pincode=customer.cust_pincode;
```

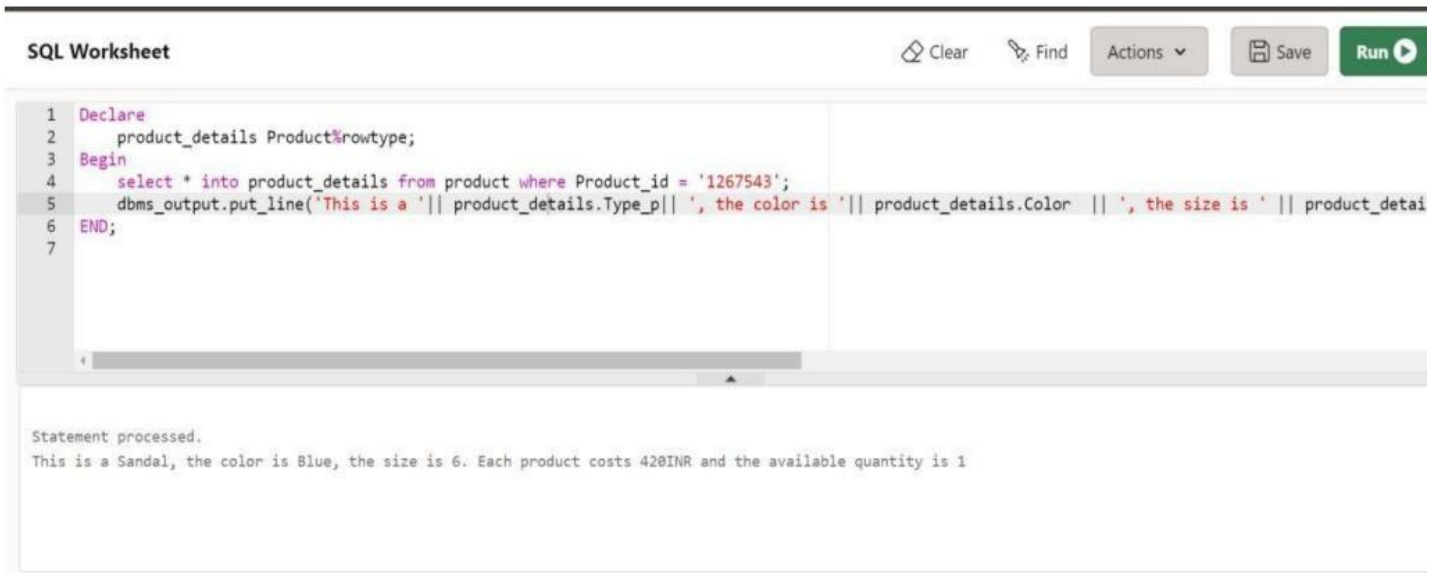
| SELLER_ID | CUSTID |
|---|---|
| 23EFD5 | 865FTYGRHIHB |

Download CSV

## PL/SQL:

1) **Displaying a detailed view of each product of a particular product**

```
Declare
        product_details Product%rowtype;
Begin
        select * into product_details from product where
Product_id = '1267543';
        dbms_output.put_line('This is a '||
product_details.Type_p|| ', the color is '|| product_details.Color
|| ', the size is ' || product_details.P_Size || '. Each product
costs '|| product_details.Cost_p || 'INR and the available
quantity is '|| product_details.Quantity);
END;
```




2) **Error Messages displayed when too many rows or no data is selected (Inbuilt Exception Handling)**

```
DECLARE
   product_size VARCHAR(2);
BEGIN
    select P_Size from product into product_size where Type_p
= 'Shoes';
EXCEPTION
    WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('TOO MANY ROWS');
    WHEN NO_DATA_FOUND THEN
         DBMS_OUTPUT.PUT_LINE('NO DATA FOUND');
END;
```

```
DECLARE
   product_size VARCHAR(2);
BEGIN
    select P_Size into product_size from product where Type_p
= 'Pant';
EXCEPTION
    WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('TOO MANY ROWS');
    WHEN NO_DATA_FOUND THEN
         DBMS_OUTPUT.PUT_LINE('NO DATA FOUND');
END;
```

SQL Worksheet

Clear Find Actions Save Run

```
DECLARE
    product_size VARCHAR(2);
BEGIN
     select P_Size into product_size from product  where Type_p  = 'Shoes';
EXCEPTION
     WHEN TOO_MANY_ROWS THEN
          DBMS_OUTPUT.PUT_LINE('TOO MANY ROWS');
     WHEN NO_DATA_FOUND THEN
           DBMS_OUTPUT.PUT_LINE('NO DATA FOUND');
END;
```

```
Statement processed.
```

SQL Worksheet

Clear Find Actions Save Run

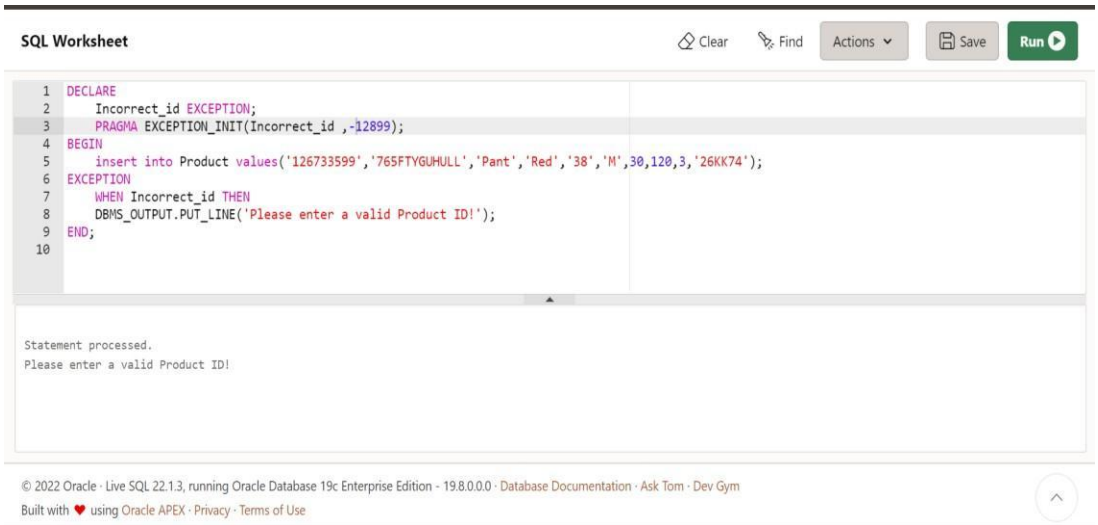
```

DECLARE
    product_size VARCHAR(2);
BEGIN
     select P_Size into product_size from product where Type_p  = 'Pant';
EXCEPTION
     WHEN TOO_MANY_ROWS THEN
          DBMS_OUTPUT.PUT_LINE('TOO MANY ROWS');
     WHEN NO_DATA_FOUND THEN
           DBMS_OUTPUT.PUT_LINE('NO DATA FOUND');
END;
```

```
Statement processed.
NO DATA FOUND
```

3) **Error Messages displayed when a wrong product id with too many characters is inserted (Non- predefined Exception Handling)**

```
DECLARE
   Incorrect_id EXCEPTION;
   PRAGMA EXCEPTION_INIT(Incorrect_id ,-12899);
BEGIN
   insert into Product
values('126733599','765FTYGUHULL','Pant','Red','38','M',30,1
20,3,'26KK74');
EXCEPTION
   WHEN Incorrect_id THEN
   DBMS_OUTPUT.PUT_LINE('Please enter a valid Product
   ID!');
END
```




**4) Display all the products that are available in the shop**

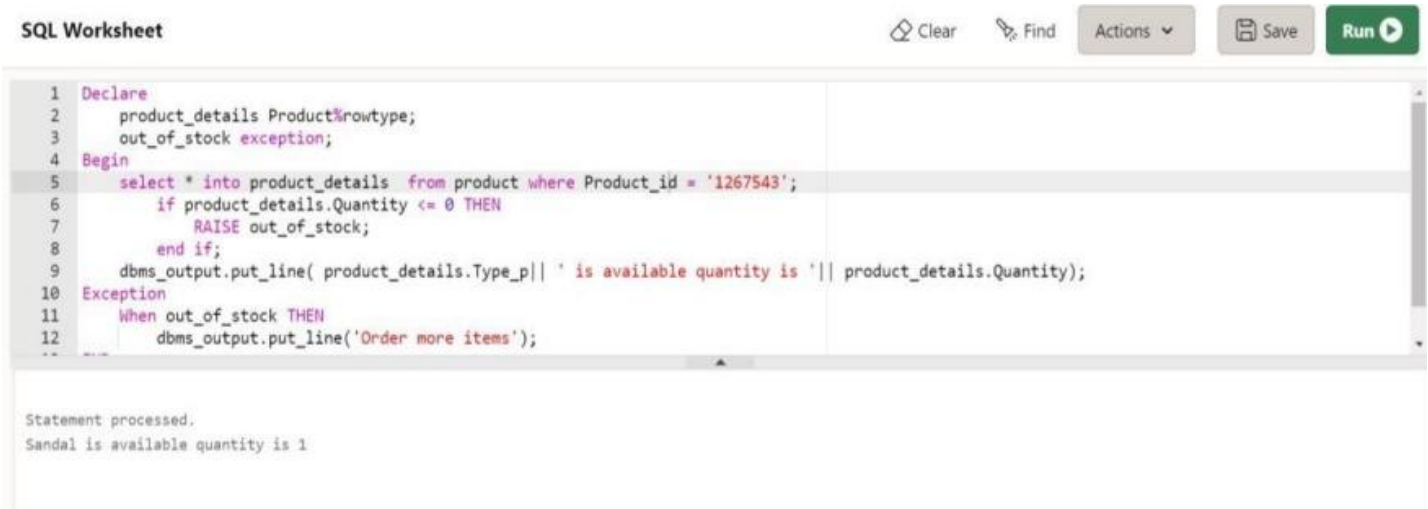
```
DECLARE
cs product.Cost_p%type;
    ty product.Type_p%type;
    id product.product_id%type;
    cursor cf is
    select product_id,Cost_p,Type_p from product;
BEGIN
    open cf;
    loop
```

```
    fetch cf into id,cs,ty;
    exit when cf%notfound;
    dbms_output.put_line('Product' || id || 'has cost ' || cs || '
and the type is ' || ty);
    end loop;
   close cf;
EXCEPTION
    when no_data_found then
    dbms_output.put_line('No products available');
END;
```


**5) Raise error if product being ordered has a quantity of 0 (User defined Exception Handling)**

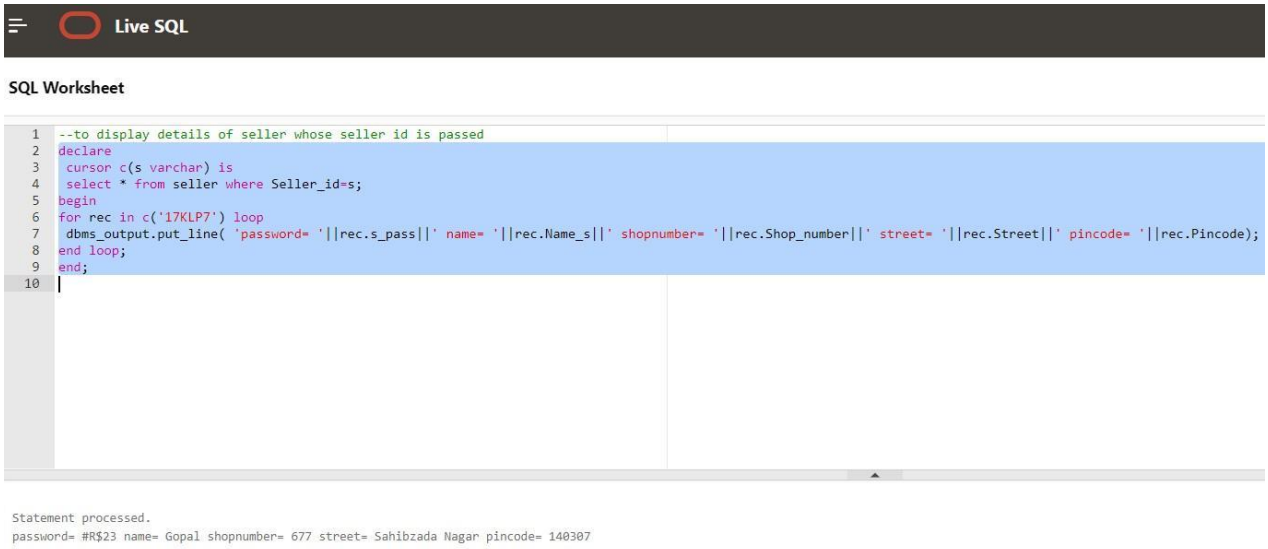
```
Declare
        product_details Product%rowtype;
```

```
                                        out_of_stock exception;
Begin
                                        select * into product_details from
                                        product where
Product_id = '1267543';
          if product_details.Quantity <= 0 THEN
             RAISE out_of_stock;
          end if;
       dbms_output.put_line( product_details.Type_p|| ' is
available quantity is '|| product_details.Quantity);
Exception
   When out_of_stock THEN
      dbms_output.put_line('Order more items');
END;
```




**6) Cursor for loop for finding details of a seller with given seller id**

```
declare
 cursor c(s varchar) is
 select * from seller where Seller_id=s;
begin
for rec in c('17KLP7') loop
 dbms_output.put_line( 'password= '||rec.s_pass||' name=
'||rec.Name_s||' shopnumber= '||rec.Shop_number||' street=
'||rec.Street||' pincode= '||rec.Pincode);
end loop;
End;
```




## 7) Apply 10% discount on all items using cursor

```
   declare
cursor c is select * from Product;
d number
p number;
```

```
begin
for rec in c loop
p:=rec.cost_p-0.1*rec.cost_p;
d:=rec.Discount+10;
update Product set cost_p=p,Discount=d where
Product_id=rec.Product_id;
end loop;
end;
```

Live SQL

SQL Worksheet

```
--to reduce price of products by 10%

declare
cursor c is select * from Product;
d number;
p number;
begin
for rec in c loop
 p:=rec.cost_p-0.1*rec.cost_p;
 d:=rec.Discount+10;
 update Product set cost_p=p,Discount=d where Product_id=rec.Product_id;
end loop;
end;
```

```
Statement processed.
```

Live SQL

SQL Worksheet

```
declare
cursor c is select * from Product;
d number;
p number;
begin
for rec in c loop
 p:=rec.cost_p-0.1*rec.cost_p;
 d:=rec.Discount+10;
 update Product set cost_p=p,Discount=d where Product_id=rec.Product_id;
end loop;
end;

select * from Product;
```

| PRODUCT_ID | CUSTID | TYPE_P | COLOR | P_SIZE | GENDER | DISCOUNT | COST_P | QUANTITY | SELLER_ID |
|---|---|---|---|---|---|---|---|---|---|
| 1267543 | 765FTYGUHUHB | Sandal | Blue | 6 | F | 30 | 378 | 1 | 26KL74 |
| 2345676 | 765DBHBABDBB | Coat | Black | 32 | M | 40 | 2457 | 1 | 23EFD5 |
| 3647833 | 565FTYHYHUHB | Shoes | Green | 9 | M | 35 | 4500 | 2 | 1ABQ36 |
| 7384903 | 765FTYGUHUUI | Shirt | White | 12 | F | 60 | 450 | 1 | 1CFQ36 |
| 4567839 | 865FTYGRHIHB | Jeans | Blue | 36 | F | 10 | 1071 | 1 | 17KLP7 |

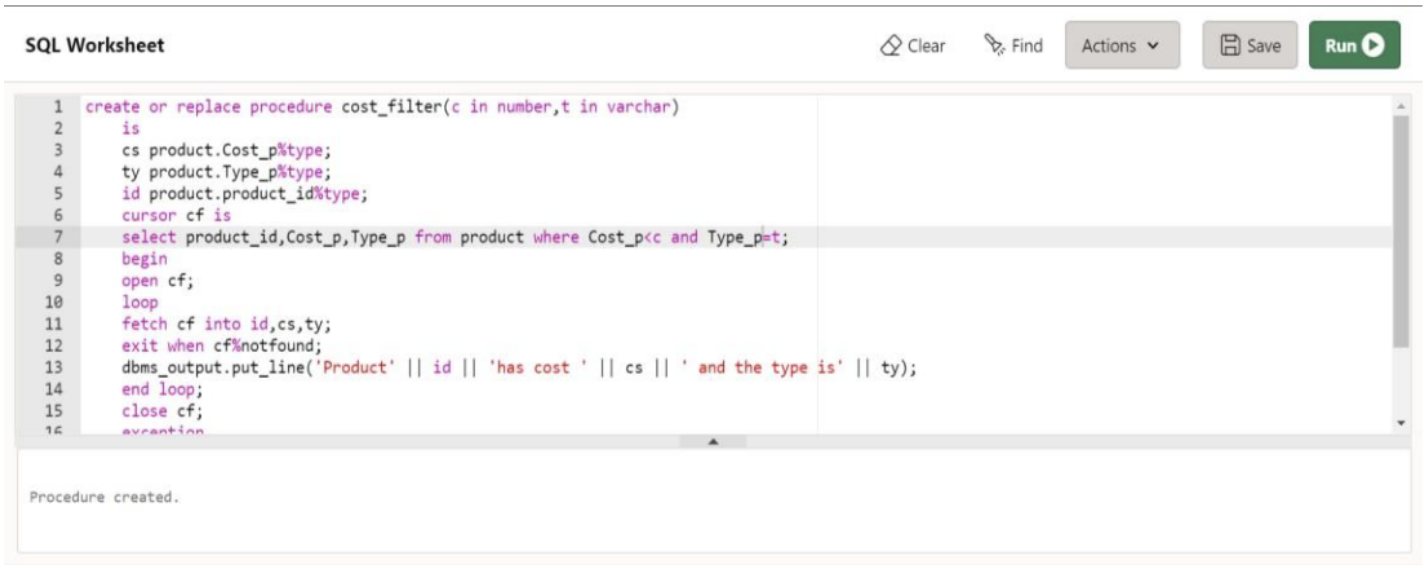
Download CSV

5 rows selected.

**8) Procedure which returns the type of product with the cost less than the given cost**

```
create or replace procedure cost_filter(c in number,t in varchar)
    is
    cs product.Cost_p%type;
    ty product.Type_p%type;
    id product.product_id%type;
```

```
    cursor cf is
    select product_id,Cost_p,Type_p from product where Cost_p<c
and Type_p=t;
    begin
    open cf;
    loop
    fetch cf into id,cs,ty;
    exit when cf%notfound;
    dbms_output.put_line('Product' || id || 'has cost ' || cs || '
and the type is' || ty);
    end  loop;
    close  cf;
    exception
    when no_data_found then
    dbms_output.put_line('Sorry no such products exist');
    end;
```



**9) Function which returns total number of products which a particular seller sells**

```
create or replace function totalProducts(sId in varchar)
return number
is
total number(2):=0;
begin
select count(*) into total
from product
where seller_id=sId;
return total;
end
```


;

**<u>Function execution:</u>**

```
declare
c number(2);
begin
c:=totalProducts('1ABQ36');
```

```
dbms_output.put_line('Total products is : '|| c)
end;
```


**10) Procedure which returns the total quantity of product with the given ID (Procedure with exception handling)**

```
create or replace procedure prod_details(p_id in varchar)
 is
 quan number(2);
 begin
 select quantity into quan from product where product_id=p_id;
 exception
 when no_data_found then
 dbms_output.put_line('Sorry no such product exist !!');
 end;
```

SQL Worksheet

Clear Find Actions Save Run

```
create or replace procedure prod_details(p_id in varchar)
is
quan number(2);
begin
    select quantity into quan from product where product_id=p_id;
    exception
    when no_data_found then
    dbms_output.put_line('Sorry no such product exist!');
end;
```

```
Procedure created.
```

11)**Procedure to delete seller info based on sellerid** create or replace procedure deleteseller(sellerid varchar)as begin
delete from Seller where Seller_id=sellerid;
end;

declare
s varchar(6);
begin
s:='17KLP7';
deleteseller(s);
end;

## SQL Worksheet

```
create or replace procedure deleteseller(sellerid varchar)as
begin
delete from Seller where Seller_id=sellerid;
end;
/*declare
s varchar(6);
begin
s:='17KLP7';
deleteseller(s);
end;*/
```

```
Procedure created.
```

Live SQL

SQL Worksheet

```
create or replace procedure deleteseller(sellerid varchar)as
begin
delete from Seller where Seller_id=sellerid;
end;
declare
s varchar(6);
begin
s:='17KLP7';
deleteseller(s);
end;
```

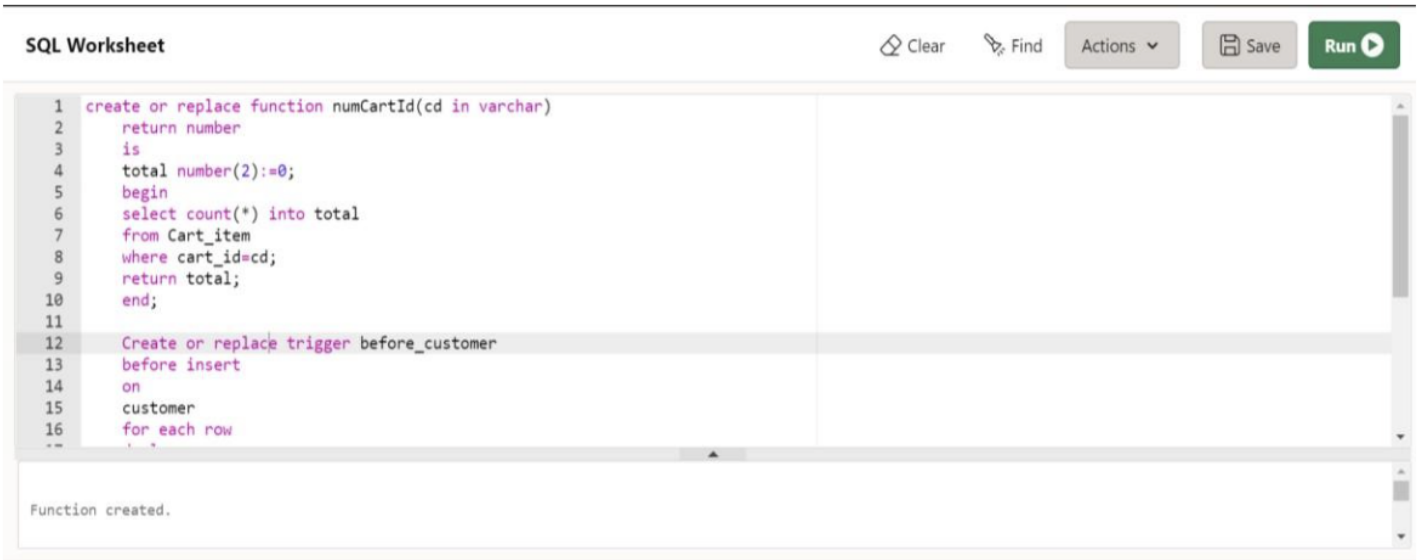
```
Statement processed.
```

## 12) Triggers:

**Trigger that will execute before inserting new customer to database and inserting a new cartId to the cart_items table (Function to count number of cart items)**

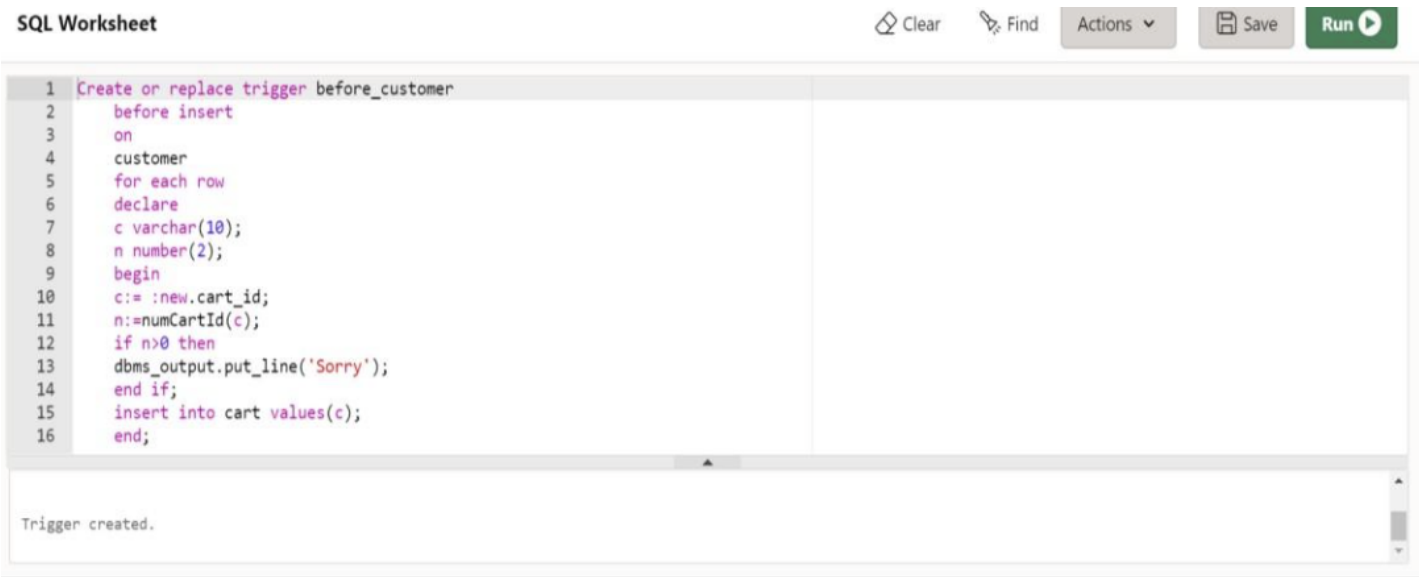
```
create or replace function numCartId(cd in varchar)
 return number
```

```
is
total number(2):=0;
begin
select count(*) into total
from cart_item
where cart_id=cd;
return total;
end;
```




```
Create or replace trigger before_customer
before insert
on
customer
for each row
declare
c varchar(10);
n number(2);
begin
c:= :new.cart_id;
n:=numCartId(c);
```

```
if n>0 then
dbms_output.put_line('Sorry');
end if;
insert into cart values(c);
end;
```




**13) Trigger to update the total amount of user everytime he adds something to payment table**

```
create or replace function total_cost(cId in varchar)
return number
is

total number(2):=0;
begin
select sum(cost_p) into total from product,cart_item where
product.product_id=cart_item.product_id and cart_id=cId;
return total;
end;
```

SQL Worksheet

Clear Find Actions Save Run

```
create or replace function total_cost(cId in varchar)
return number
is
total number(2):=0;
begin
select sum(cost_p) into total from product,cart_item where product.product_id=cart_item.product_id and cart_id=cId;
return total;
end;
```

Function created.

create or replace trigger before_pay_up before insert
on payment
for each row declare
total number(3); begin
total :=total_cost(:new.custid); insert
into payment
values(:new.payment_id,total,:new.payment_date,:new.custid); end;

SQL Worksheet

Clear Find Actions Save Run

```
create or replace trigger before_pay_up
before insert
on
payment
for each row
declare
total number(3);
begin
total :=total_cost(:new.custid);
insert into payment values(:new.payment_id,total,:new.payment_date,:new.custid);
end;
```