VIT®

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

DATABASE MANAGEMENT AND SYSTEMS -1003

Project: E-commerce management system

REVIEW-1

SLOT-L41+42

Submitting to : Prof: BIMAL KUMAR RAY

Submitted by:

1.K.Maheswara Reddy - 18BIT0132

2.M.Likith – 18BIT0039

3.RVR.Srninivas -18BIT0007

Introduction :

Our DBMS project is based E-commerce management. E-commerce is fast gaining ground as an accepted and used business paradigm. More and more business houses are implementing web sites providing functionality for performing commercial transactions over the web. It is reasonable to say that the process of shopping on the web is becoming commonplace. The objective of this project is to develop a general purpose e-commerce store where product like clothes can be bought from the comfort of home through the Internet. However, for implementation purposes, this paper will deal with an online shopping for clothes.

Data Requirements:

Data Description

This database consists of

- Users : User and Admin information is added to database with Unique ID based¬ on their roles.
- Shopping : Complete products information is stored in this table. Orders : Customer ordered products, status and delivery information is stored in¬ this table.

Data Objects

- User : ID, UserName, Password, Email, Role.
- Shopping : ID, Product, Product ID, Cost, Category, Image, Description.
- Orders : ID, Client, Product, Quantity, Price, Date, OrderShipped.

Relationships:-

Customer Selects the Product and does the Payment (1-n):-

A customer can select any number of products and can move to payment portal. So this is 1-n relationship.

Payment done for the selected items i.e which are in cart (n-1):-

Multiple products can be stored in cart and can make a single payment. So this is n-1 relationship.

Customer selects and item and saves to cart (1-1):-

A single item can be stored in a cart at a time so you can't store multiple items at a single time in cart. So this is 1-1 relationship.

Seller sells the Products (1-n):-

A single seller can sell multiple items. So this is a 1-n relationship.

A Cart is made of multiple cart item (1-n):-

A customer can add the multiple items in to cart. So this is 1-n relationship.

Product can be added to cart (1-n):-

Multiple products can be added into a cart. So this is 1-n relationship.

Functional Requirements

User roles & profiles

- The customer. The person that connects to the on-line e-commerce front-end and browses the product catalog or places an order.

- The sales manager. The person is the company that looks after the sales orders and ensures that all the sales operations work correctly.

Business process definition

- Search the product catalog.
- Browse the product catalog.
- Making an order using an on-line front-end.
- Notifying the customer that is not enough stock.
- Generate an invoice for an order that has been completed.

User stories

1. A customer connects to the e-commerce front-end looking for a product:

1. The customer starts at the e-commerce front-end main page.
2. The customer searches in the products catalog for a specific product.
3. The customer gets information about the product.
4. The customer decides if she wants to proceed with an order.

2. A customer connects to the e-commerce front-end for browsing the products catalog:

1. The customer starts at the e-commerce front-end main page.
2. The customer browser hierarchically the products catalog.
3. The customer gets information about the product.
4. The customer decides if she wants to proceed with an order.

3. A customer places successfully an order into the system:

1. The customer registers or logins into the system.
2. The customer places an order within the system.
3. System checks that there is enough stock.
4. Completes the order successfully.
5. Decides if she wants an invoice send by mail or shown to her.

4. A customer tries to place an order into the system but the product is out of stock:

1. The customer registers or logins into the system.
2. The customer places an order within the system.
3. System checks that there is enough stock.
4. The user is informed of the shortage of stock.

5. A customer tracks the status of her orders:

1. The customer logins into the system.
2. Browses her historical list of sales orders.
3. Selects the order for which wants more detail and its status.
4. Detail of the order is shown.

Functional requirements based on business processes

Customer management

- It's a common scenario that a new user registers using the e-commerce front-end to be able to perform a commercial transaction later.

Product catalog:-

A product catalog contains all the products that a user can view.

1.It should be possible for a user to perform the following actions:

- Browse the product catalog hierarchically sorted by alphabetically by product name.
- Browse the product catalog hierarchically by product category.
- Search the product catalog.
- Get all the details of a product.

2.Every product object can contain at least the following details:

- Product name.
- Product description.
- Product category.
- Product attributes *(weight, color, size, etc)*.
- Product price and tax.

It should be possible for users to query product's inventory availability. This can be displayed when the user is viewing the product information.

Sales order:-

It should be possible for a user to:

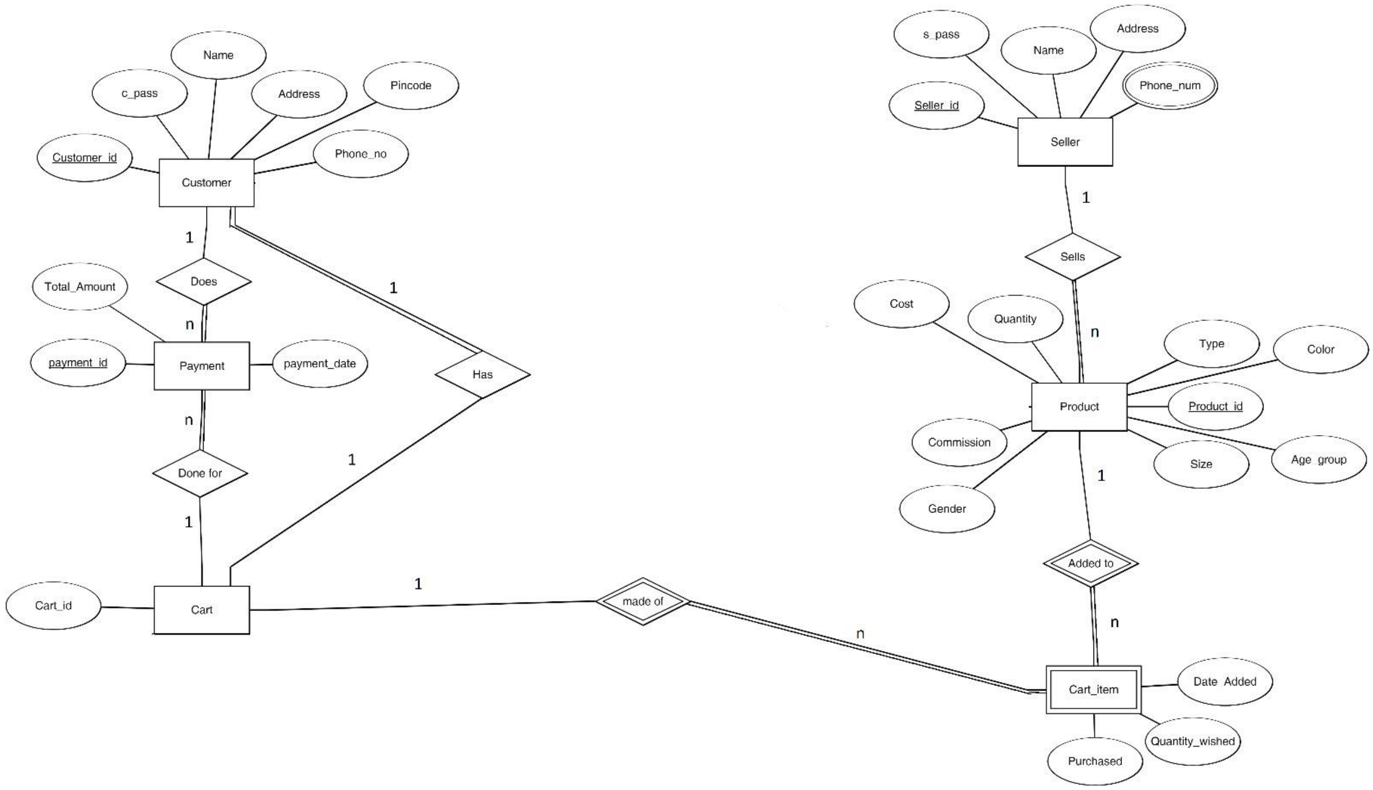
- Perform a full sales order.
- Modify or cancel an order once it has been created in the system.
- Get a list of the products that has bought.
- Track the status of his orders pending to be send.

Invoicing:-

It should be possible for the e-commerce system:

- Get access to the full invoice object to render an invoice by itself.
- Get a PDF version of an invoice using the default server report.

ER MODEL


Name
c_pass
Address
Pincode
Customer_id
Customer
Phone_no
1
Does
1
Total_Amount
n
payment_id
Payment
payment_date
n
Has
Done for
1
1
Cart_id
Cart
1
made of
n
s_pass
Name
Address
Seller_id
Phone_num
Seller
1
Sells
Cost
Quantity
n
Type
Color
Product
Product_id
Commission
Size
Age_group
Gender
1
Added to
n
Cart_item
Date Added
Quantity_wished
Purchased

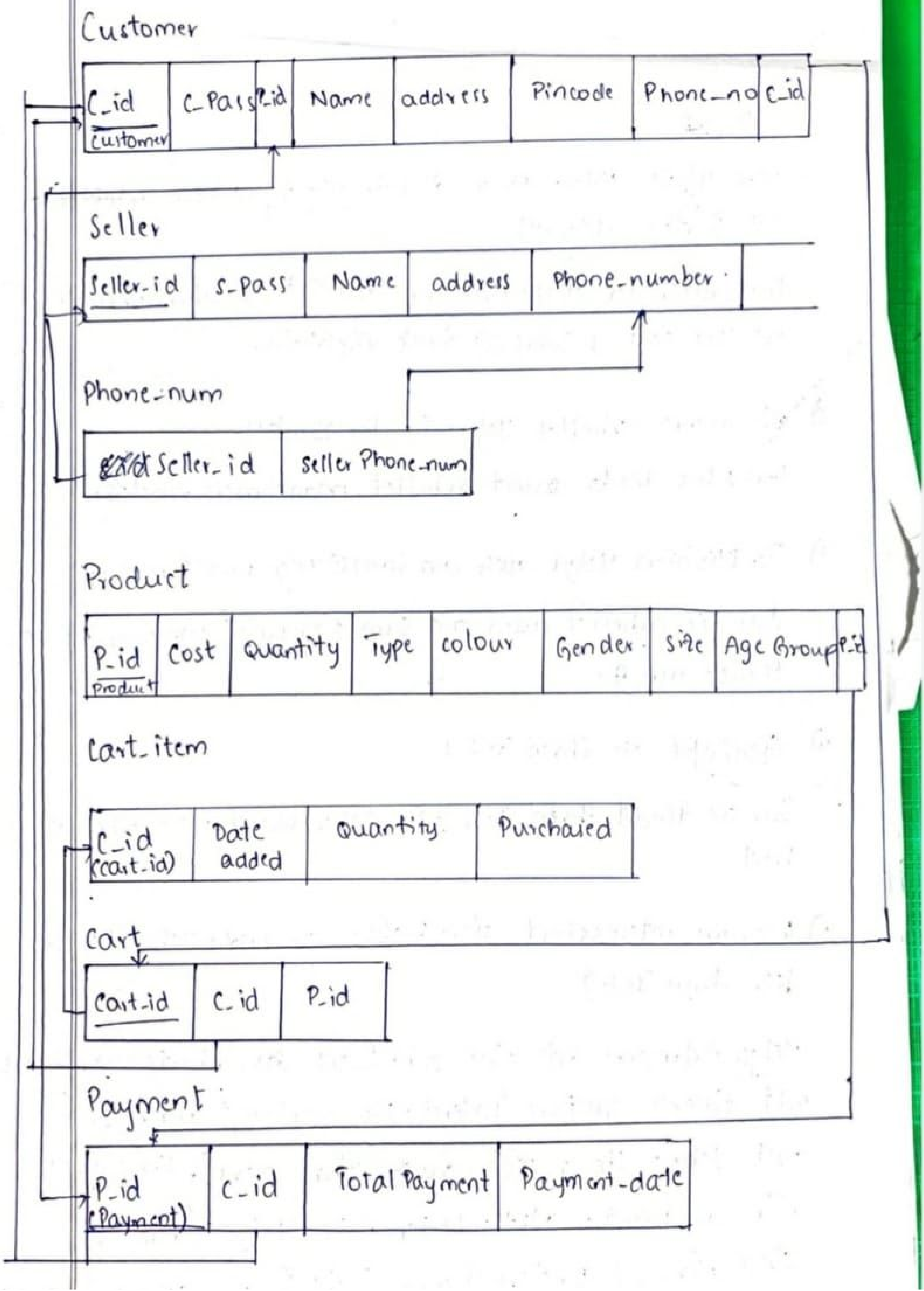
DATABASE MANAGEMENT SYSTEMS (ITE1003)

REVIEW 2

BY:

K.Maheswara reddy	18BIT0132
M.Likith chowdary	18BIT0039
RVR.Srinivas	18BIT0007

QUESTION 4:



Customer

C_id customer	C_Pass	P_id	Name	address	Pincode	Phone_no	C_id

Seller

Seller_id	S_pass	Name	address	Phone_number	

Phone_num

Seller_id	seller Phone_num

Product

P_id Product	Cost	Quantity	Type	colour	Gender	size	Age Group	P_id

Cart_item

C_id (cart_id)	Date added	Quantity	Purchased

Cart

Cart_id	C_id	P_id

Payment

P_id (Payment)	C_id	Total Payment	Payment_date

```
);
```

QUESTION 5:

TABLES

- Cart
- Customer
- Seller
- Seller_Phone_num
- Payment
- Product
- Cart_item

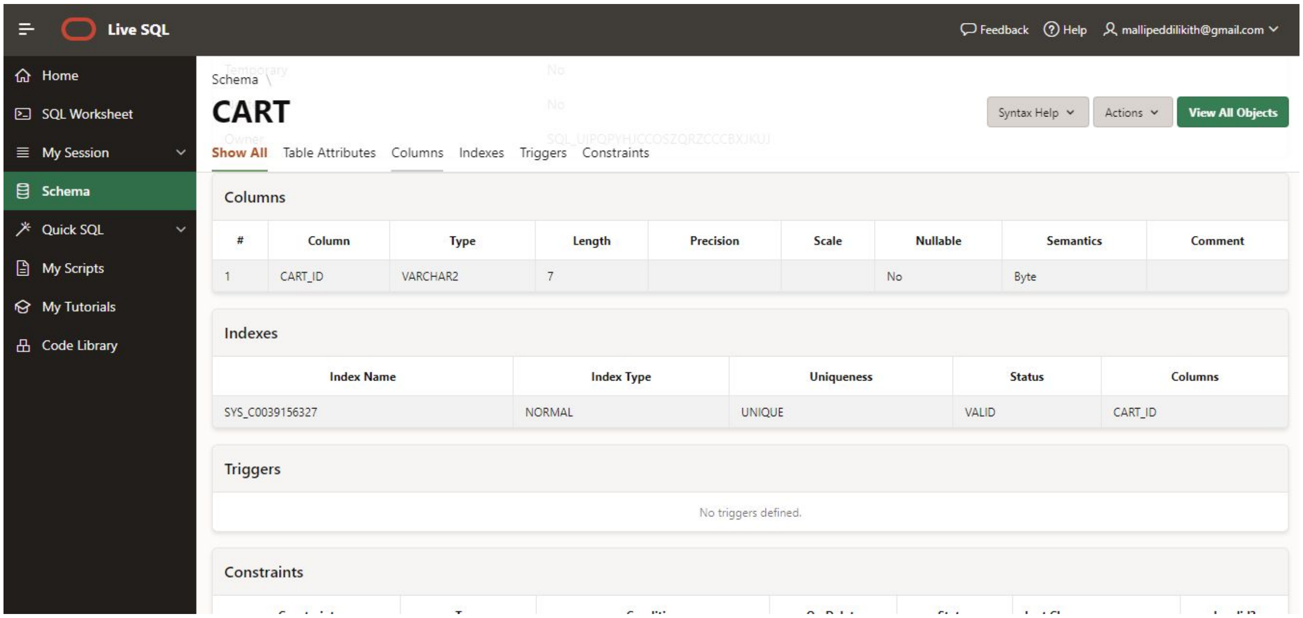
CODE TO CREATE TABLES :-

```
    CREATE TABLE Cart
(
Cart_id VARCHAR(7) NOT NULL,
PRIMARY KEY(Cart_id)
);
CREATE TABLE Customer
(
Customer_id VARCHAR(6) NOT NULL,
c_pass VARCHAR(10) NOT NULL,
Name VARCHAR(20) NOT NULL,
Address VARCHAR(20) NOT NULL,
Pincode NUMBER(6) NOT NULL,
Phone_number_s number(10) NOT NULL,
PRIMARY KEY (Customer_id),
Cart_id VARCHAR(7) NOT NULL,
FOREIGN KEY(Cart_id) REFERENCES cart(Cart_id)
);
CREATE TABLE Seller
```

```
(
Seller_id VARCHAR(6) NOT NULL,
s_pass VARCHAR(10) NOT NULL,
Name VARCHAR(20) NOT NULL,
Address VARCHAR(10) NOT NULL,
PRIMARY KEY (Seller_id)
);
CREATE TABLE Seller_Phone_num
(
Phone_num NUMBER(10) NOT NULL,
Seller_id VARCHAR(6) NOT NULL,
PRIMARY KEY (Phone_num, Seller_id),
FOREIGN KEY (Seller_id) REFERENCES Seller(Seller_id)
ON DELETE CASCADE
);
CREATE TABLE Payment
(
payment_id VARCHAR(7) NOT NULL,
payment_date DATE NOT NULL,
Payment_type VARCHAR(10) NOT NULL,
Customer_id VARCHAR(6) NOT NULL,
Cart_id VARCHAR(7) NOT NULL,
PRIMARY KEY (payment_id),
FOREIGN KEY (Customer_id) REFERENCES Customer(Customer_id),
FOREIGN KEY (Cart_id) REFERENCES Cart(Cart_id),
total_amount numeric(6)
);
CREATE TABLE Product
(
Product_id VARCHAR(7) NOT NULL,
Type VARCHAR(7) NOT NULL,
Color VARCHAR(15) NOT NULL,
P_Size VARCHAR(2) NOT NULL,
Gender CHAR(1) NOT NULL,
Commission NUMBER(2) NOT NULL,
Cost 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
);
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,
FOREIGN KEY (Cart_id) REFERENCES Cart(Cart_id),
FOREIGN KEY (Product_id) REFERENCES Product(Product_id),
Primary key(Cart_id,Product_id)
);
alter table Cart_item add purchased varchar(3) default 'NO';
```

TABLE CART:-

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



CART

Columns

#	Column	Type	Length	Precision	Scale	Nullable	Semantics	Comment
1	CART_ID	VARCHAR2	7			No	Byte	

Indexes

Index Name	Index Type	Uniqueness	Status	Columns
SYS_C0039156327	NORMAL	UNIQUE	VALID	CART_ID

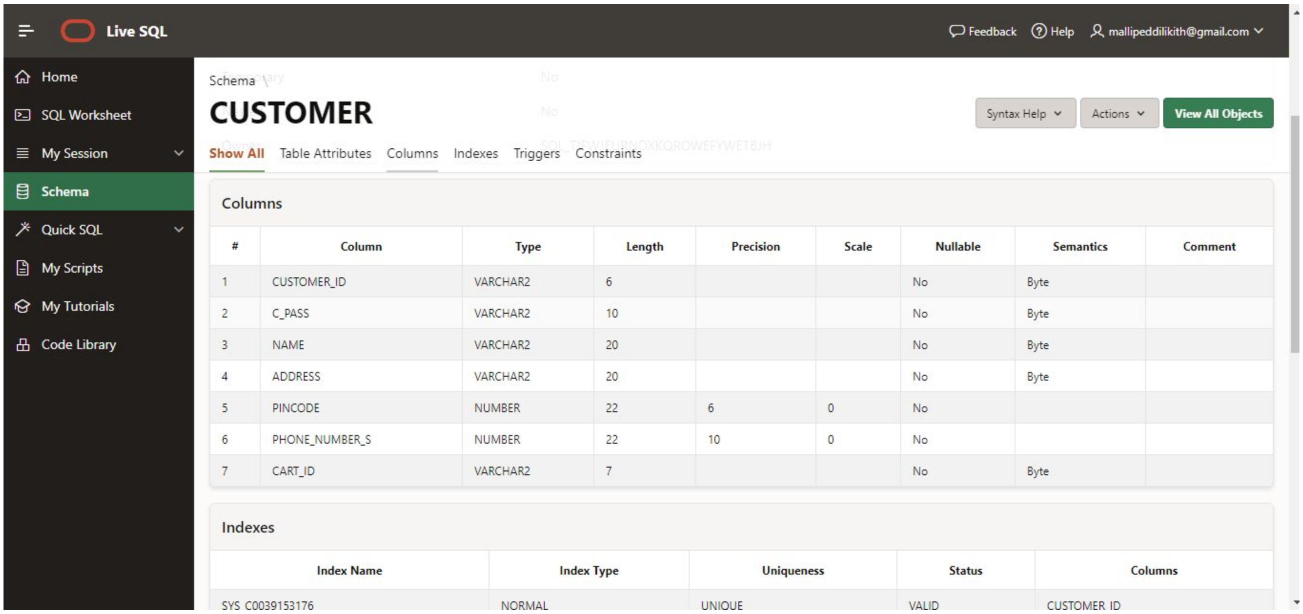
Triggers

No triggers defined.

Constraints

TABLE CUSTOMER:-

```
CREATE TABLE Customer
(
Customer_id VARCHAR(6) NOT NULL,
c_pass VARCHAR(10) NOT NULL,
Name VARCHAR(20) NOT NULL,
Address VARCHAR(20) NOT NULL,
Pincode NUMBER(6) NOT NULL,
Phone_number_s number(10) NOT NULL,
PRIMARY KEY (Customer_id),
Cart_id VARCHAR(7) NOT NULL,
FOREIGN KEY(Cart_id) REFERENCES cart(Cart_id)
);
```



CUSTOMER

Columns

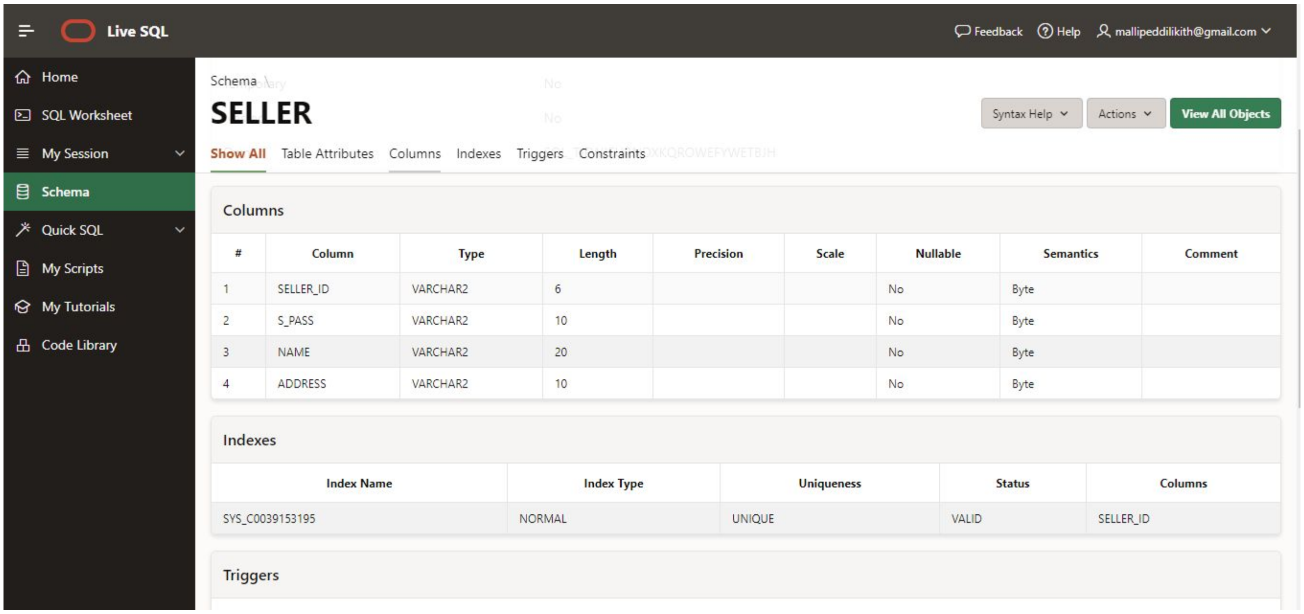
#	Column	Type	Length	Precision	Scale	Nullable	Semantics	Comment
1	CUSTOMER_ID	VARCHAR2	6			No	Byte	
2	C_PASS	VARCHAR2	10			No	Byte	
3	NAME	VARCHAR2	20			No	Byte	
4	ADDRESS	VARCHAR2	20			No	Byte	
5	PINCODE	NUMBER	22	6	0	No		
6	PHONE_NUMBER_S	NUMBER	22	10	0	No		
7	CART_ID	VARCHAR2	7			No	Byte	

Indexes

Index Name	Index Type	Uniqueness	Status	Columns
SYS_C0039153176	NORMAL	UNIQUE	VALID	CUSTOMER_ID

TABLE_ SELLER:-

```
CREATE TABLE Seller
(
Seller_id VARCHAR(6) NOT NULL,
s_pass VARCHAR(10) NOT NULL,
Name VARCHAR(20) NOT NULL,
Address VARCHAR(10) NOT NULL,
PRIMARY KEY (Seller_id)
);
```



Schema

SELLER

Columns

#	Column	Type	Length	Precision	Scale	Nullable	Semantics	Comment
1	SELLER_ID	VARCHAR2	6			No	Byte	
2	S_PASS	VARCHAR2	10			No	Byte	
3	NAME	VARCHAR2	20			No	Byte	
4	ADDRESS	VARCHAR2	10			No	Byte	

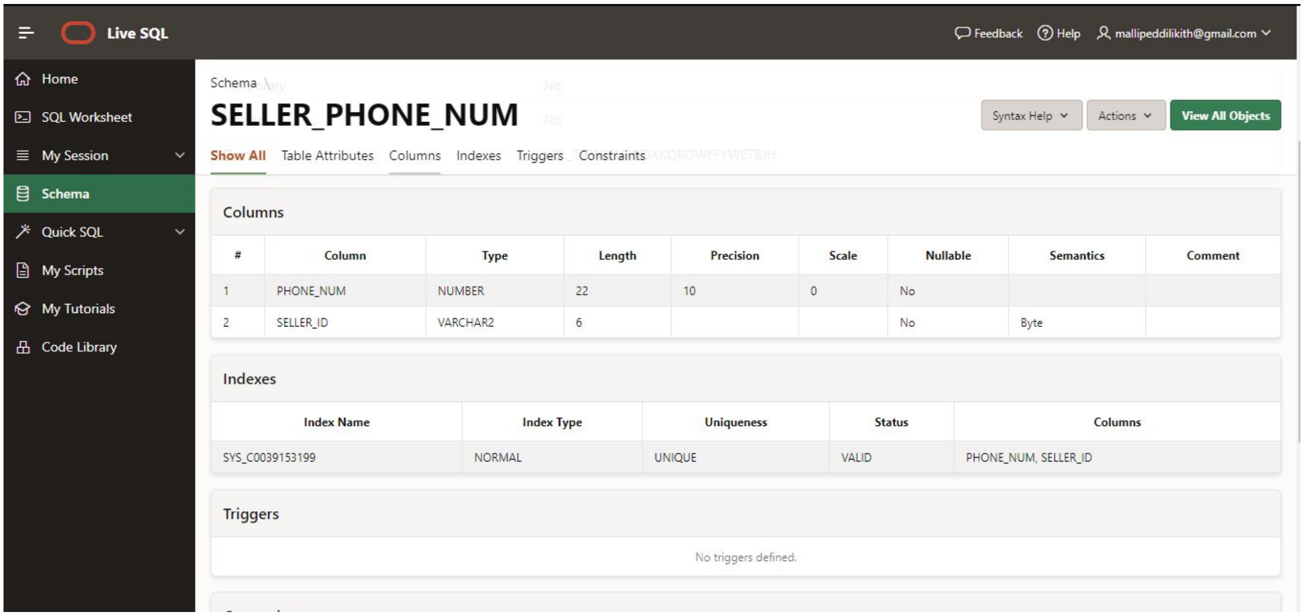
Indexes

Index Name	Index Type	Uniqueness	Status	Columns
SYS_C0039153195	NORMAL	UNIQUE	VALID	SELLER_ID

Triggers

TABLE Seller_Phone_num :-

```
CREATE TABLE Seller_Phone_num
 (
 Phone_num NUMBER(10) NOT NULL,
 Seller_id VARCHAR(6) NOT NULL,
 PRIMARY KEY (Phone_num, Seller_id),
 FOREIGN KEY (Seller_id) REFERENCES Seller(Seller_id)
 ON DELETE CASCADE
 );
```



Schema

SELLER_PHONE_NUM

Columns

#	Column	Type	Length	Precision	Scale	Nullable	Semantics	Comment
1	PHONE_NUM	NUMBER	22	10	0	No		
2	SELLER_ID	VARCHAR2	6			No	Byte	

Indexes

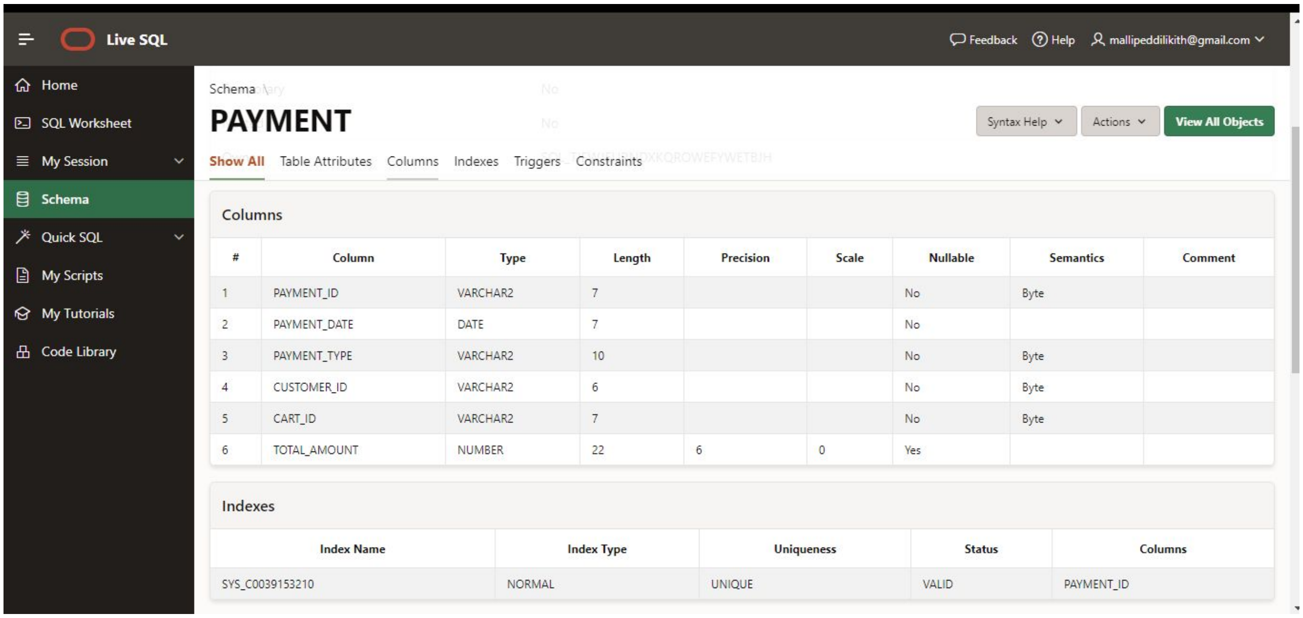
Index Name	Index Type	Uniqueness	Status	Columns
SYS_C0039153199	NORMAL	UNIQUE	VALID	PHONE_NUM, SELLER_ID

Triggers

No triggers defined.

TABLE_PAYMENT :-

```
CREATE TABLE Payment
 (
 payment_id VARCHAR(7) NOT NULL,
 payment_date DATE NOT NULL,
 Payment_type VARCHAR(10) NOT NULL,
 Customer_id VARCHAR(6) NOT NULL,
 Cart_id VARCHAR(7) NOT NULL,
 PRIMARY KEY (payment_id),
 FOREIGN KEY (Customer_id) REFERENCES Customer(Customer_id),
 FOREIGN KEY (Cart_id) REFERENCES Cart(Cart_id),
 total_amount numeric(6)
 );
```



Schema

PAYMENT

Columns

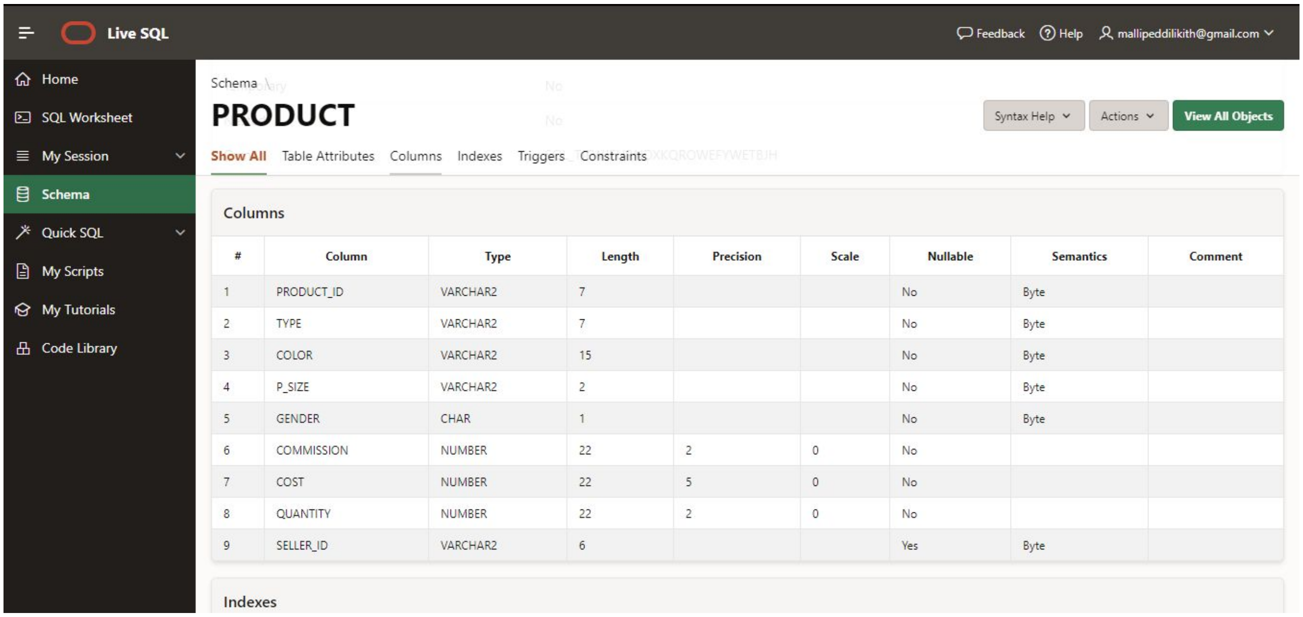
#	Column	Type	Length	Precision	Scale	Nullable	Semantics	Comment
1	PAYMENT_ID	VARCHAR2	7			No	Byte	
2	PAYMENT_DATE	DATE	7			No		
3	PAYMENT_TYPE	VARCHAR2	10			No	Byte	
4	CUSTOMER_ID	VARCHAR2	6			No	Byte	
5	CART_ID	VARCHAR2	7			No	Byte	
6	TOTAL_AMOUNT	NUMBER	22	6	0	Yes		

Indexes

Index Name	Index Type	Uniqueness	Status	Columns
SYS_C0039153210	NORMAL	UNIQUE	VALID	PAYMENT_ID

TABLE PRODUCT:-

```
CREATE TABLE Product
 (
 Product_id VARCHAR(7) NOT NULL,
 Type VARCHAR(7) NOT NULL,
 Color VARCHAR(15) NOT NULL,
 P_Size VARCHAR(2) NOT NULL,
 Gender CHAR(1) NOT NULL,
 Commission NUMBER(2) NOT NULL,
 Cost 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
 );
```



Schema

PRODUCT

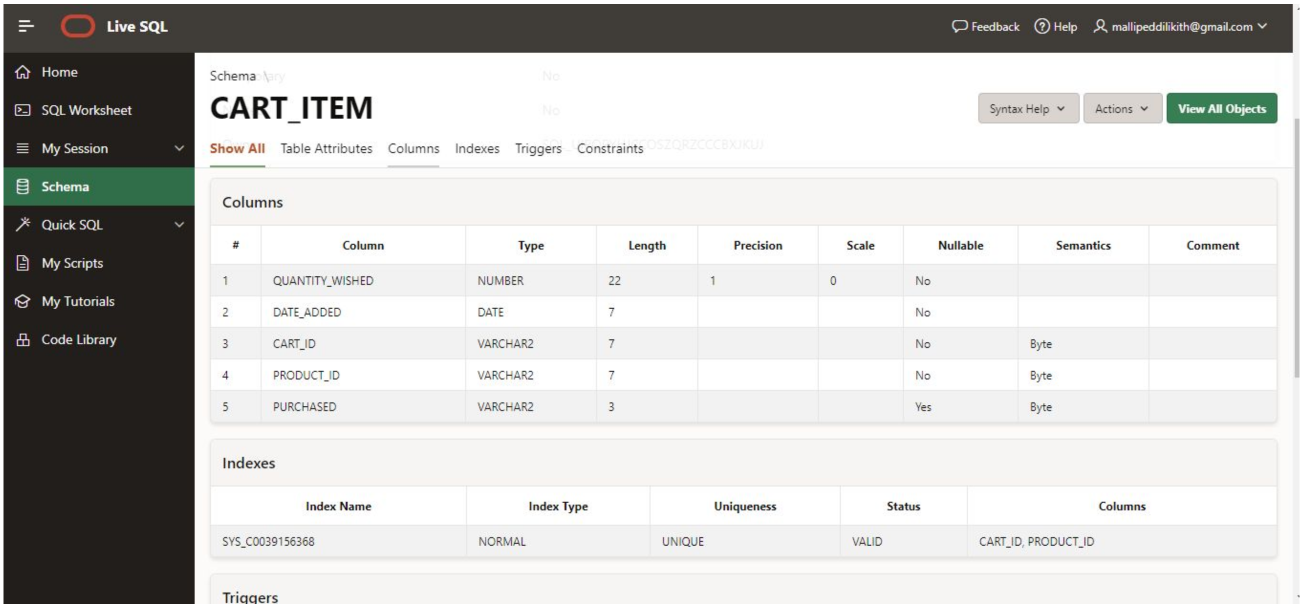
Columns

#	Column	Type	Length	Precision	Scale	Nullable	Semantics	Comment
1	PRODUCT_ID	VARCHAR2	7			No	Byte	
2	TYPE	VARCHAR2	7			No	Byte	
3	COLOR	VARCHAR2	15			No	Byte	
4	P_SIZE	VARCHAR2	2			No	Byte	
5	GENDER	CHAR	1			No	Byte	
6	COMMISSION	NUMBER	22	2	0	No		
7	COST	NUMBER	22	5	0	No		
8	QUANTITY	NUMBER	22	2	0	No		
9	SELLER_ID	VARCHAR2	6			Yes	Byte	

Indexes

TABLE Cart_item :-

```
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,
 FOREIGN KEY (Cart_id) REFERENCES Cart(Cart_id),
 FOREIGN KEY (Product_id) REFERENCES Product(Product_id),
 Primary key(Cart_id,Product_id)
 );
 alter table Cart_item add purchased varchar(3) default 'NO';
```



CART_ITEM

Columns

#	Column	Type	Length	Precision	Scale	Nullable	Semantics	Comment
1	QUANTITY_WISHED	NUMBER	22	1	0	No		
2	DATE_ADDED	DATE	7			No		
3	CART_ID	VARCHAR2	7			No	Byte	
4	PRODUCT_ID	VARCHAR2	7			No	Byte	
5	PURCHASED	VARCHAR2	3			Yes	Byte	

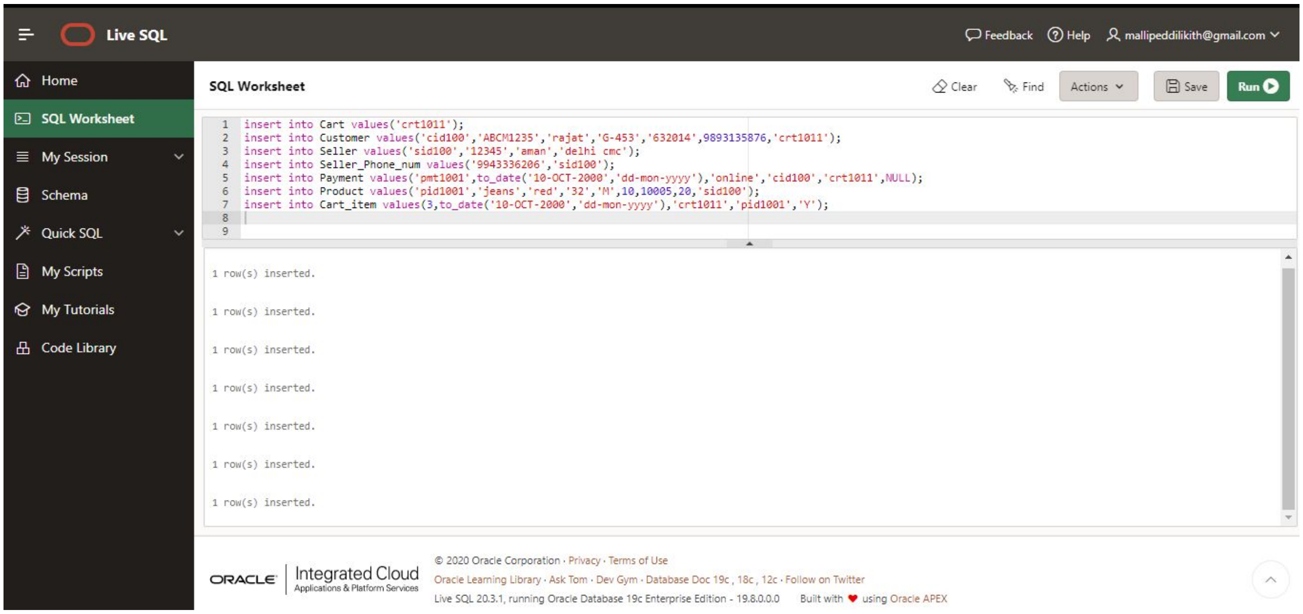
Indexes

Index Name	Index Type	Uniqueness	Status	Columns
SYS_C0039156368	NORMAL	UNIQUE	VALID	CART_ID, PRODUCT_ID

Triggers

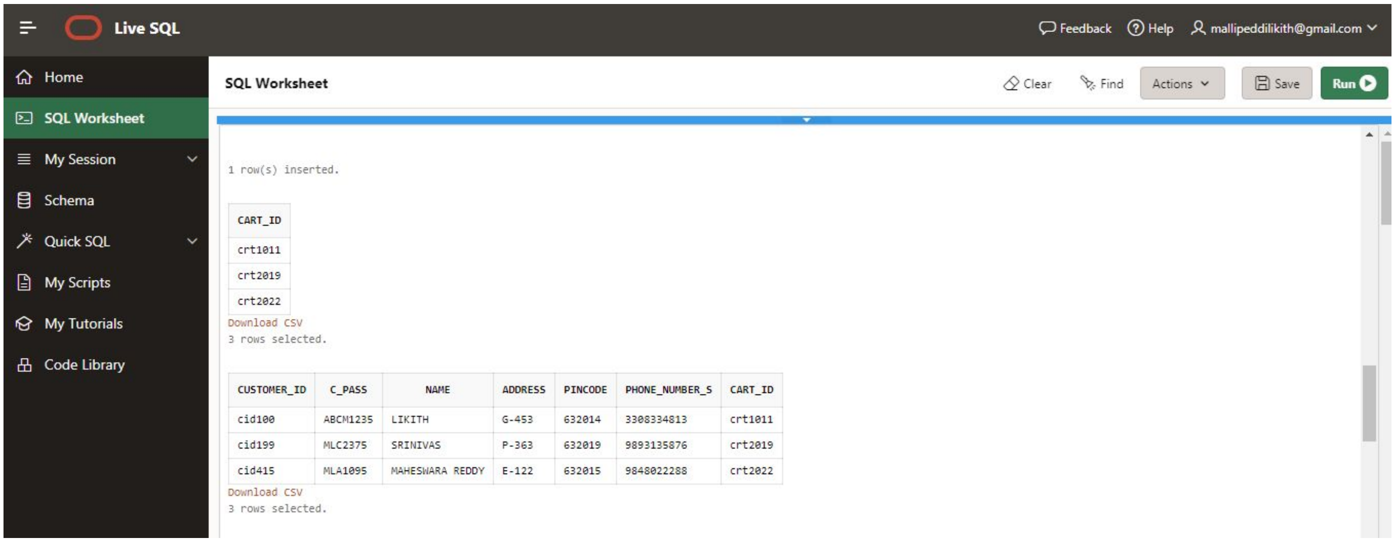
CODE TO INSERT VALUES TO TABLE:-

```
insert into Cart values('crt1011');
Insert into Customer
values('cid100','ABCM1235','rajat','G-453','632014',9893135876, 'crt1011');
insert into Seller values('sid100','12345','aman','delhi cmc');
insert into Seller_Phone_num values('9943336206','sid100');
insert into Payment
values('pmt1001',to_date('10-OCT-2000','dd-mon-yyyy'),'online','cid100','crt1011',N
ULL);
insert into Product values('pid1001','jeans','red','32','M',10,10005,20,'sid100');
insert into Cart_item
values(3,to_date('10-OCT-2000','dd-mon-yyyy'),'crt1011','pid1001','Y');
```


Live SQL
Feedback
Help
mallipeddilikith@gmail.com
Home
SQL Worksheet
My Session
Schema
Quick SQL
My Scripts
My Tutorials
Code Library
SQL Worksheet
Clear
Find
Actions
Save
Run
ORACLE
Integrated Cloud
Applications & Platform Services
© 2020 Oracle Corporation · Privacy · Terms of Use
Oracle Learning Library · Ask Tom · Dev Gym · Database Doc 19c , 18c , 12c · Follow on Twitter
Live SQL 20.3.1, running Oracle Database 19c Enterprise Edition - 19.8.0.0.0 Built with ♥ using Oracle APEX

```
insert into Cart values('crt1011');
insert into Customer values('cid100','ABCM1235','rajat','G-453','632014',9893135876,'crt1011');
insert into Seller values('sid100','12345','aman','delhi cmc');
insert into Seller_Phone_num values('9943336206','sid100');
insert into Payment values('pmt1001',to_date('10-OCT-2000','dd-mon-yyyy'),'online','cid100','crt1011',NULL);
insert into Product values('pid1001','jeans','red','32','M',10,10005,20,'sid100');
insert into Cart_item values(3,to_date('10-OCT-2000','dd-mon-yyyy'),'crt1011','pid1001','Y');
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```



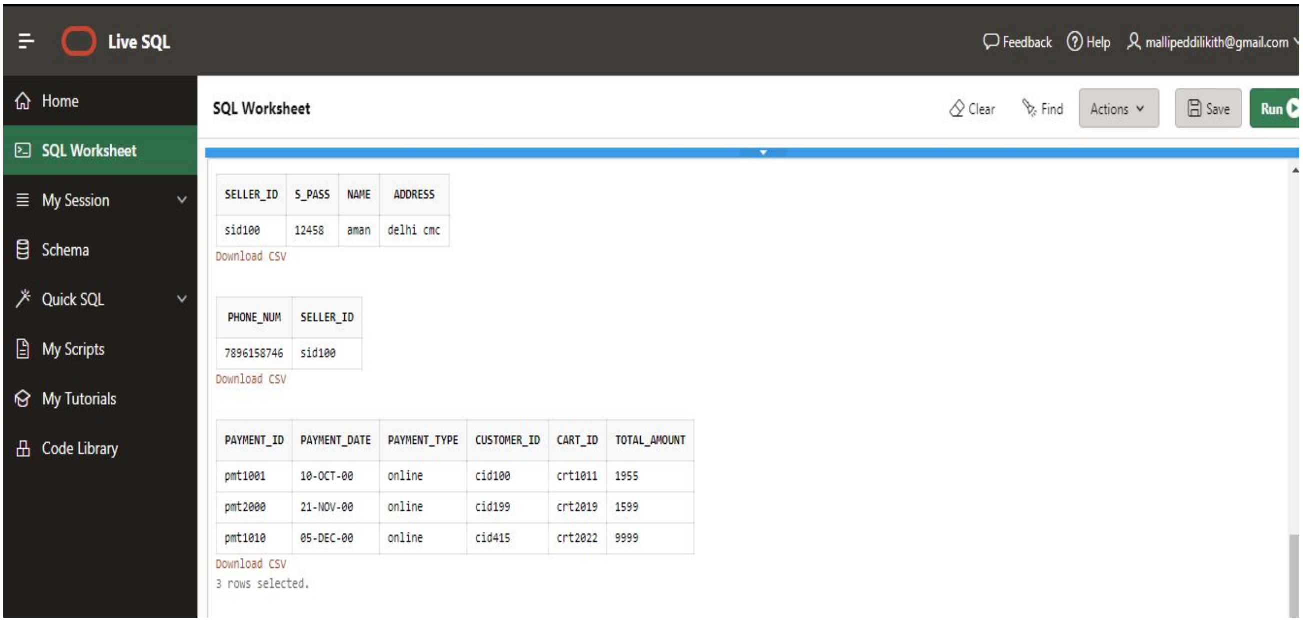
```
1 row(s) inserted.
```

CART_ID
crt1011
crt2019
crt2022

Download CSV
3 rows selected.

CUSTOMER_ID	C_PASS	NAME	ADDRESS	PINCODE	PHONE_NUMBER_S	CART_ID
cid100	ABCM1235	LIKITH	G-453	632014	3308334813	crt1011
cid199	MLC2375	SRINIVAS	P-363	632019	9893135876	crt2019
cid415	MLA1095	MAHESWARA REDDY	E-122	632015	9848022288	crt2022

Download CSV
3 rows selected.



SELLER_ID	S_PASS	NAME	ADDRESS
sid100	12458	aman	delhi cmc

Download CSV

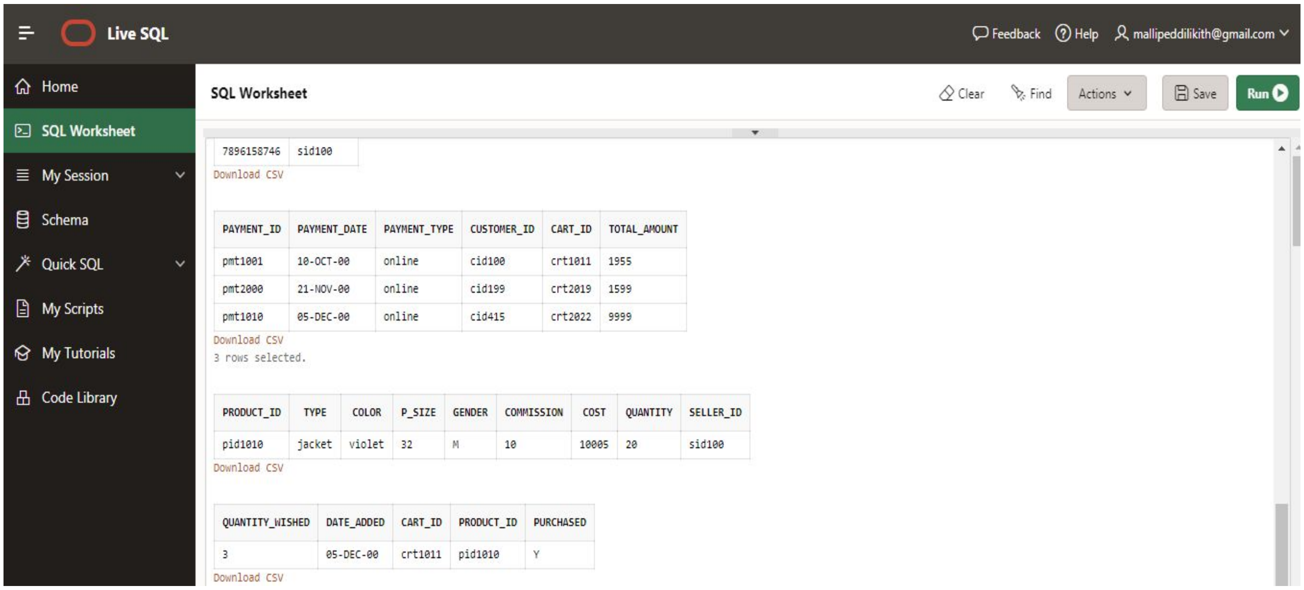
PHONE_NUM	SELLER_ID
7896158746	sid100

Download CSV

PAYMENT_ID	PAYMENT_DATE	PAYMENT_TYPE	CUSTOMER_ID	CART_ID	TOTAL_AMOUNT
pmt1001	10-OCT-00	online	cid100	crt1011	1955
pmt2000	21-NOV-00	online	cid199	crt2019	1599
pmt1010	05-DEC-00	online	cid415	crt2022	9999

Download CSV

3 rows selected.



7896158746	sid100

Download CSV

PAYMENT_ID	PAYMENT_DATE	PAYMENT_TYPE	CUSTOMER_ID	CART_ID	TOTAL_AMOUNT
pmt1001	10-OCT-00	online	cid100	crt1011	1955
pmt2000	21-NOV-00	online	cid199	crt2019	1599
pmt1010	05-DEC-00	online	cid415	crt2022	9999

Download CSV

3 rows selected.

PRODUCT_ID	TYPE	COLOR	P_SIZE	GENDER	COMMISSION	COST	QUANTITY	SELLER_ID
pid1010	jacket	violet	32	M	10	10005	20	sid100

Download CSV

QUANTITY_WISHED	DATE_ADDED	CART_ID	PRODUCT_ID	PURCHASED
3	05-DEC-00	crt1011	pid1010	Y

Download CSV

DATABASE MANAGEMENT SYSTEMS
(ITE1003)

REVIEW 3

BY :

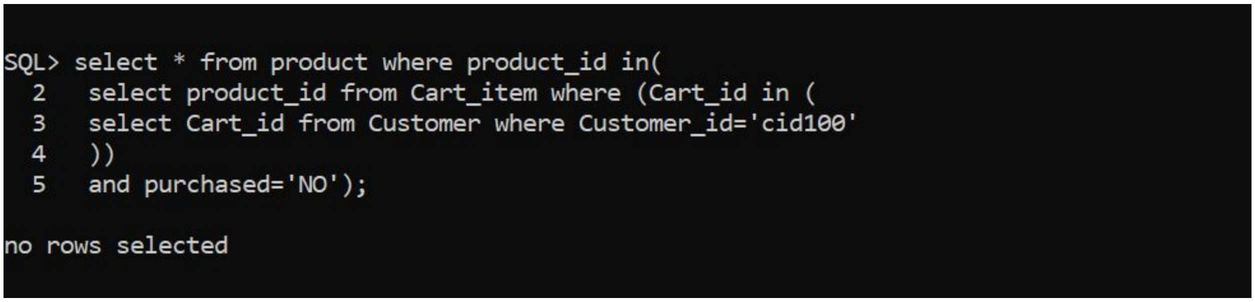
M.LIKITH CHOWDARY	18BIT0039
RVR.SRINIVAS	18BIT0007
C.MAHESWARA REDDY	18BIT0132

6. Write down the necessary SQL statements for implementation of functional requirements (refer to 2) through SQL select, delete and update statement. You may have to modify functional requirements to enable you write complex SQL statements. The SQL statements must include one query showing the usage of nvl function and nullif function, one join query involving order by clause, one uncorrelated nested query, one correlated nested query, one query involving one of the set operators, one query involving group by, having and where clause and one query involving (left or right or full) outer join.

Queries :-

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

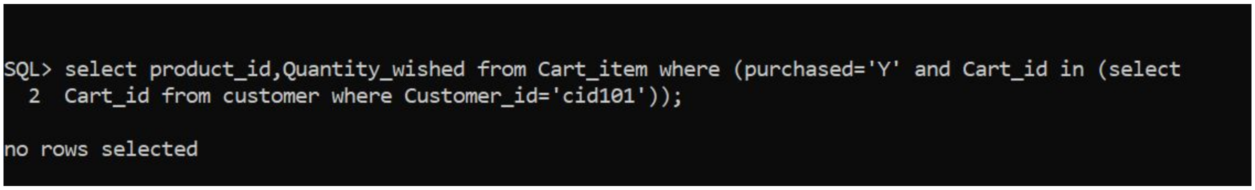
```
select * from product where product_id in(
        select product_id from Cart_item where (Cart_id in (
            select Cart_id from Customer where Customer_id='cid100'
        ))

     and purchased='NO');
```



```
SQL> select * from product where product_id in(
  2   select product_id from Cart_item where (Cart_id in (
  3   select Cart_id from Customer where Customer_id='cid100'
  4   ))
  5   and purchased='NO');

no rows selected
```

If a customer wants to see order history

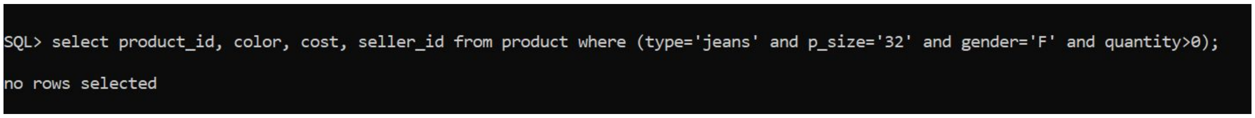
```
select product_id,Quantity_wished from Cart_item where (purchased='Y' and Cart_id in (select
Cart_id from customer where Customer_id='cid101'));
```



```
SQL> select product_id,Quantity_wished from Cart_item where (purchased='Y' and Cart_id in (select
  2  Cart_id from customer where Customer_id='cid101'));

no rows selected
```

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

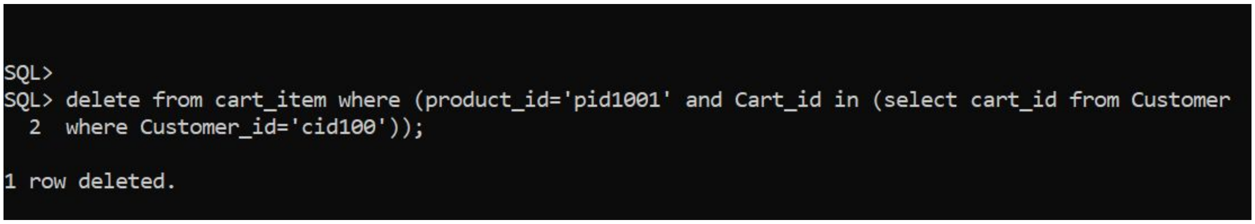
```
    select product_id, color, cost, seller_id from product where (typ e='jeans' and p_size='32'
and gender='F' and quantity>0)
```



```
SQL> select product_id, color, cost, seller_id from product where (type='jeans' and p_size='32' and gender='F' and quantity>0);

no rows selected
```

If customer wants to modify the cart

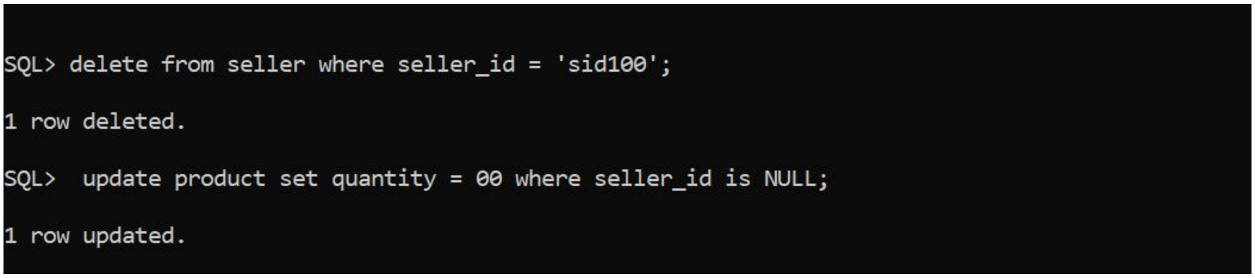
```
delete from cart_item where (product_id='pid1001' and Cart_id in (select cart_id from Customer
where Customer_id='cid100'));
```



```
SQL>
SQL> delete from cart_item where (product_id='pid1001' and Cart_id in (select cart_id from Customer
  2  where Customer_id='cid100'));

1 row deleted.
```

If a seller stops selling his product

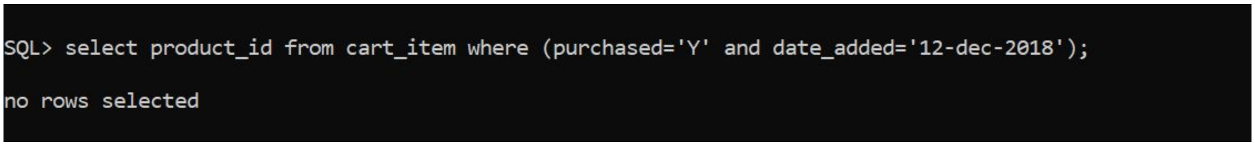
```
    delete  from seller where seller_id = 'sid100';
    update product set quantity = 00 where seller_id is NULL;
```



```
SQL> delete from seller where seller_id = 'sid100';

1 row deleted.

SQL>  update product set quantity = 00 where seller_id is NULL;

1 row updated.
```

If admin want to see what are the product purchased on the particular date

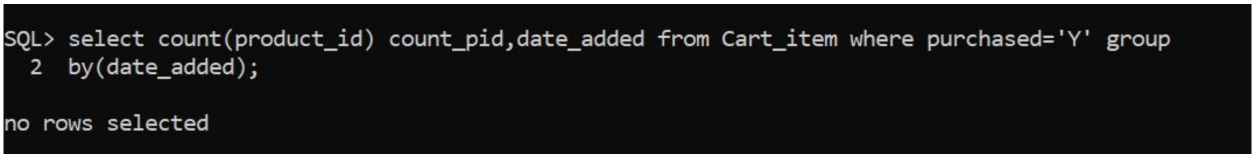
```
    select product_id from cart_item where (purchased='Y' and date_added='12-dec-2018');
```



```
SQL> select product_id from cart_item where (purchased='Y' and date_added='12-dec-2018');

no rows selected
```

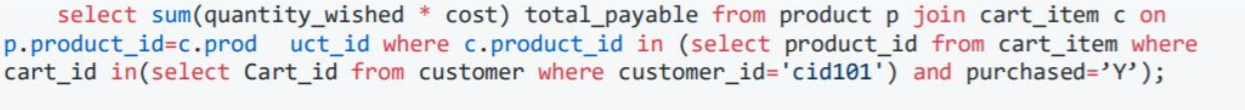
How much product sold on the particular date

```
    select count(product_id) count_pid,date_added from Cart_item where purchased='Y'  group
by(date_added);
```



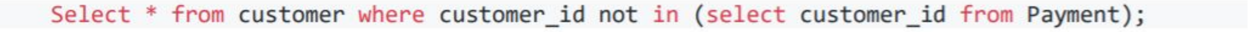
```
SQL> select count(product_id) count_pid,date_added from Cart_item where purchased='Y' group
  2  by(date_added);

no rows selected
```

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

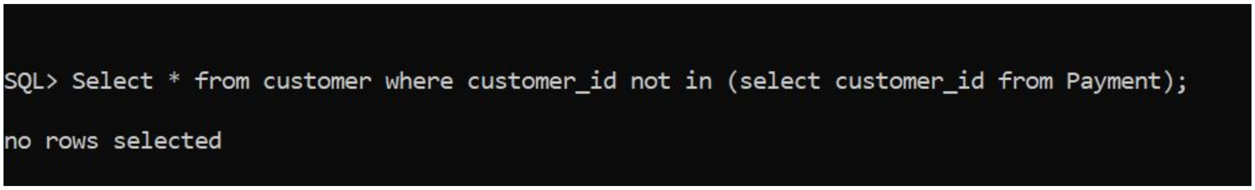


```
    select sum(quantity_wished * cost) total_payable from product p join cart_item c on
p.product_id=c.prod  uct_id where c.product_id in (select product_id from cart_item where
cart_id in(select Cart_id from customer where customer_id='cid101') and purchased=’Y’);
```

Show the details of the customer who has not purchased any thing

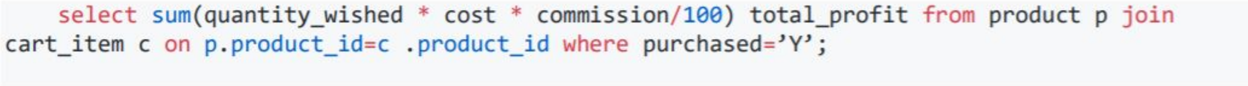


```
    Select * from customer where customer_id not in (select customer_id from Payment);
```

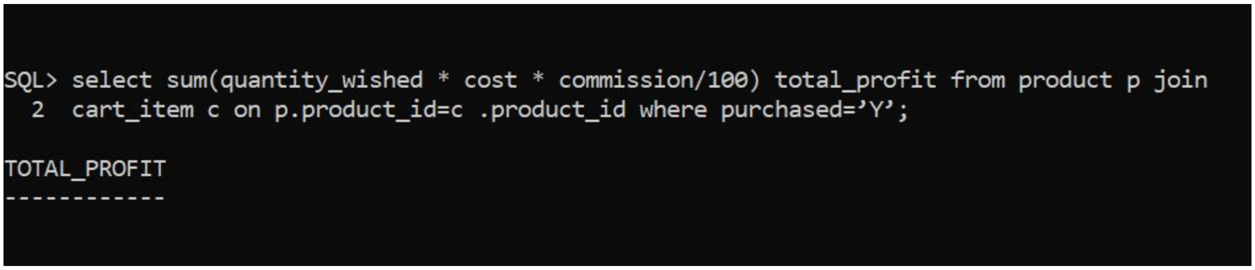


```
SQL> Select * from customer where customer_id not in (select customer_id from Payment);

no rows selected
```

Find total profit of the website from sales.



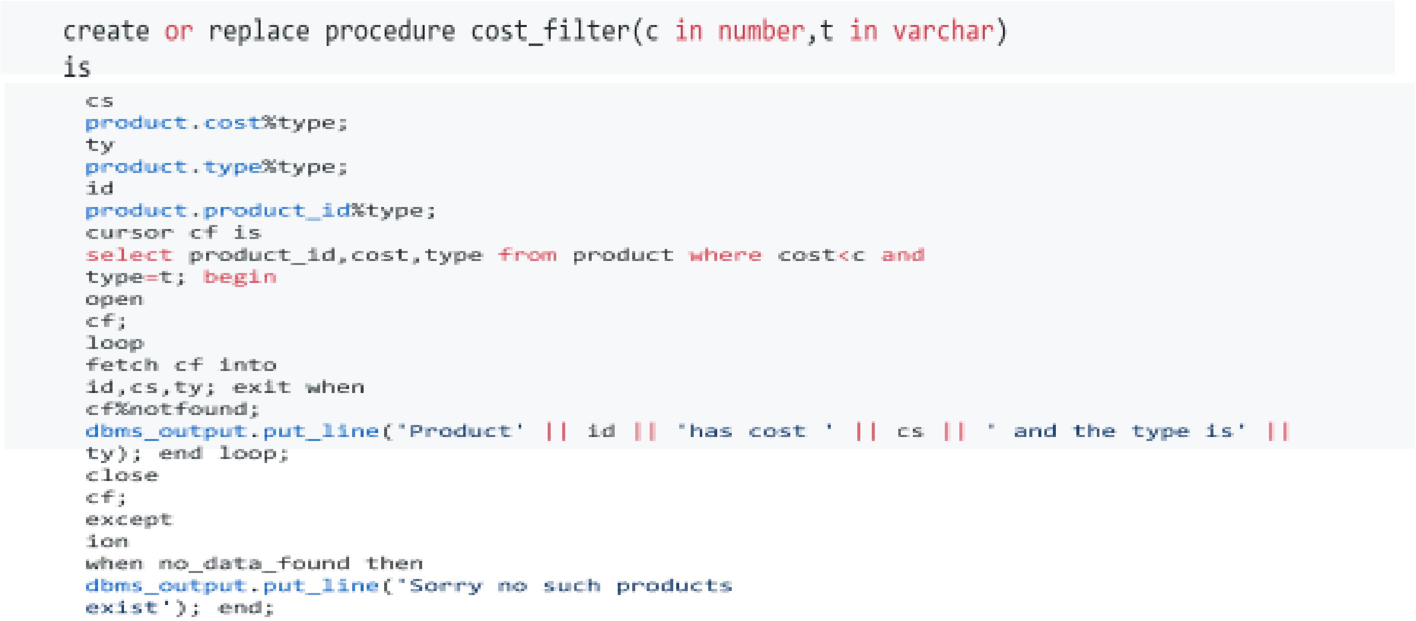
```
    select sum(quantity_wished * cost * commission/100) total_profit from product p join
cart_item c on p.product_id=c .product_id where purchased=’Y’;
```



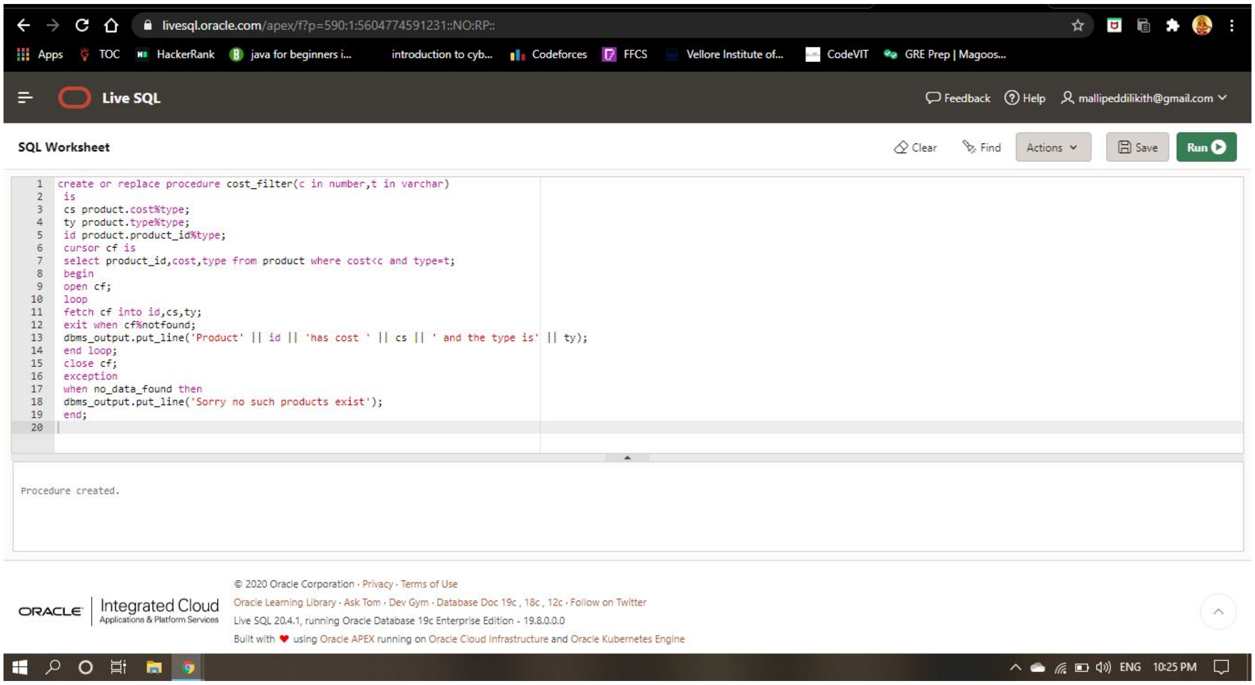
```
SQL> select sum(quantity_wished * cost * commission/100) total_profit from product p join
  2  cart_item c on p.product_id=c .product_id where purchased=’Y’;

TOTAL_PROFIT
------------
```

7. Define and implement two PL/SQL function involving cursor and two PL/SQL procedure involving cursor for the database under consideration (i. e. required for the project)

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%type;
 ty
 product.type%type;
 id
 product.product_id%type;
 cursor cf is
 select product_id,cost,type from product where cost<c and
 type=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;
 except
 ion
 when no_data_found then
 dbms_output.put_line('Sorry no such products
 exist'); end;
```


livesql.oracle.com/apex/f?p=590:1:5604774591231::NO:RP::
Apps
TOC
HackerRank
java for beginners i...
introduction to cyb...
Codeforces
FFCS
Vellore Institute of...
CodeVIT
GRE Prep | Magoos...
Live SQL
Feedback
Help
mallipeddilikith@gmail.com
SQL Worksheet
Clear
Find
Actions
Save
Run
ORACLE Integrated Cloud
Applications & Platform Services
© 2020 Oracle Corporation · Privacy · Terms of Use
Oracle Learning Library · Ask Tom · Dev Gym · Database Doc 19c , 18c , 12c · Follow on Twitter
Live SQL 20.4.1, running Oracle Database 19c Enterprise Edition - 19.8.0.0.0
Built with ♥ using Oracle APEX running on Oracle Cloud Infrastructure and Oracle Kubernetes Engine
ENG
10:25 PM

```
create or replace procedure cost_filter(c in number,t in varchar)
 is
 cs product.cost%type;
 ty product.type%type;
 id product.product_id%type;
 cursor cf is
 select product_id,cost,type from product where cost<c and type=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;
```

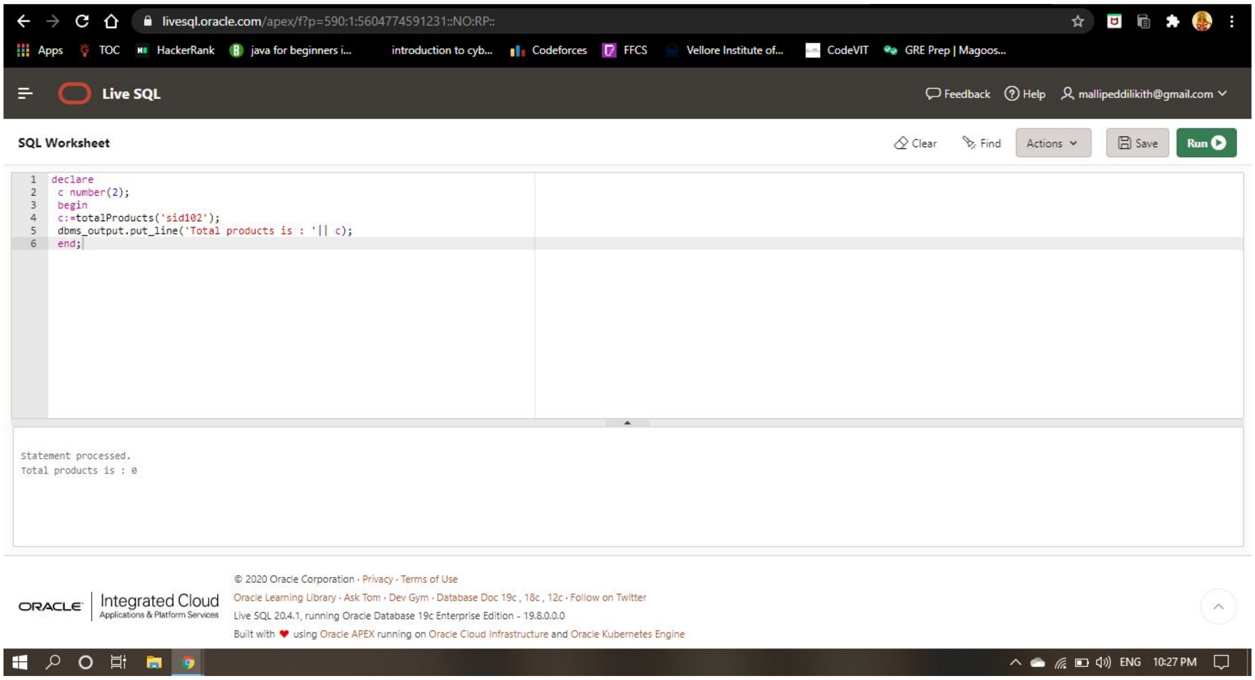
```
Procedure created.
```

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;
/
```

Function execution:

```
declare
c number(2);
begin
c:=totalProducts('sid102');
dbms_output.put_line('Total products is : '|| c);
end;
```



livesql.oracle.com/apex/f?p=590:1:5604774591231::NO:RP::
Apps
TOC
HackerRank
java for beginners i...
introduction to cyb...
Codeforces
FFCS
Vellore Institute of...
CodeVIT
GRE Prep | Magoos...
Live SQL
Feedback
Help
mallipeddilikith@gmail.com
SQL Worksheet
Clear
Find
Actions
Save
Run
ORACLE
Integrated Cloud
Applications & Platform Services
© 2020 Oracle Corporation · Privacy · Terms of Use
Oracle Learning Library · Ask Tom · Dev Gym · Database Doc 19c , 18c , 12c · Follow on Twitter
Live SQL 20.4.1, running Oracle Database 19c Enterprise Edition - 19.8.0.0.0
Built with ❤ using Oracle APEX running on Oracle Cloud Infrastructure and Oracle Kubernetes Engine
ENG
10:27 PM

```
declare
 c number(2);
 begin
 c:=totalProducts('sid102');
 dbms_output.put_line('Total products is : '|| c);
 end;
```

```
Statement processed.
Total products is : 0
```

Procedure which returns the total quantity of product with the given ID

Procedure with exception handling

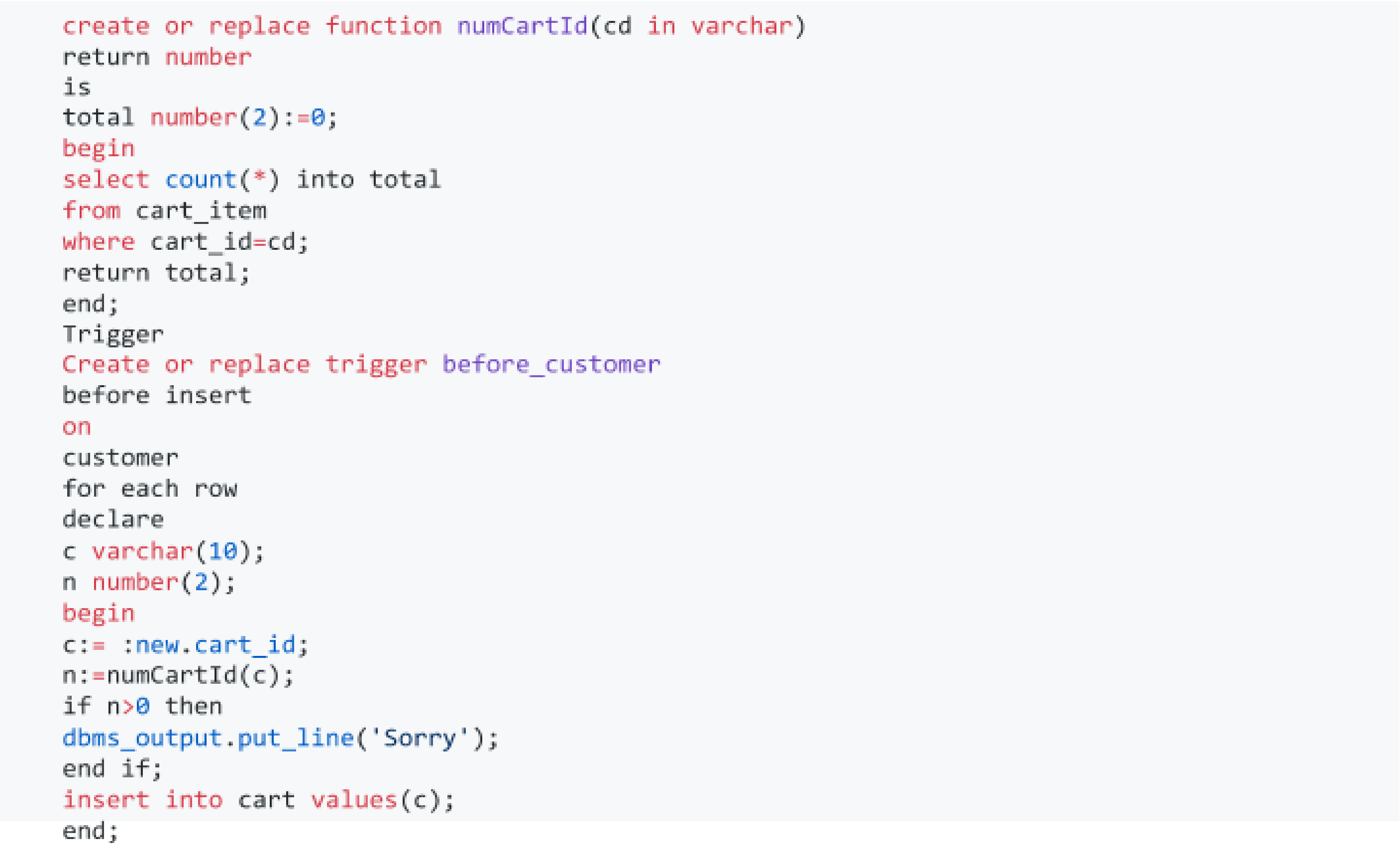
```
create or replace procedure prod_details(p_id in varchar)
is
quan number(2);
begin
select quantity into quan from product where product_id=p_id;
exception
when no_data_found then
dbms_output.put_line('Sorry no such product exist !!');
end;
/
```

8. Define three business rules appropriate for the database under consideration and implement the rules using trigger.

Trigger1

Function to count number of cart items

CODE :-

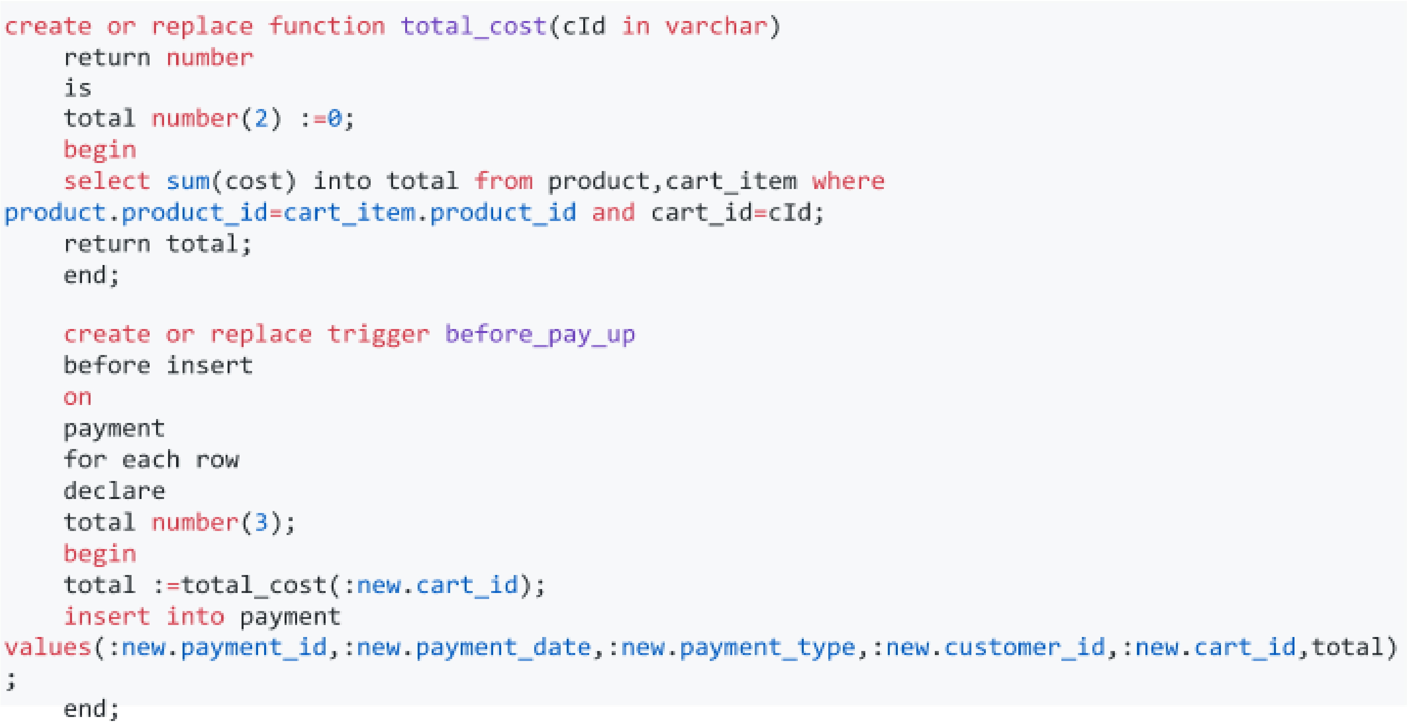


```
create or replace function numCartId(cd in varchar)
return number
is
total number(2):=0;
begin
select count(*) into total
from cart_item
where cart_id=cd;
return total;
end;
Trigger
Create or replace trigger before_customer
before insert
on
customer
for each row
declare
c varchar(10);
n number(2);
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;
```

Trigger 2 :-

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

CODE :-



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

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