AHMEDABAD UNIVERSITY

Program Name	**: B.Tech - ICT**
Semester	**: 4th**
Course Name	**: Database Management System Lab**
Project Title	**: Online Shopping Management System**
Group Members	**: Dhara Vora - 1741046**
	Krushna Shah - 1741086

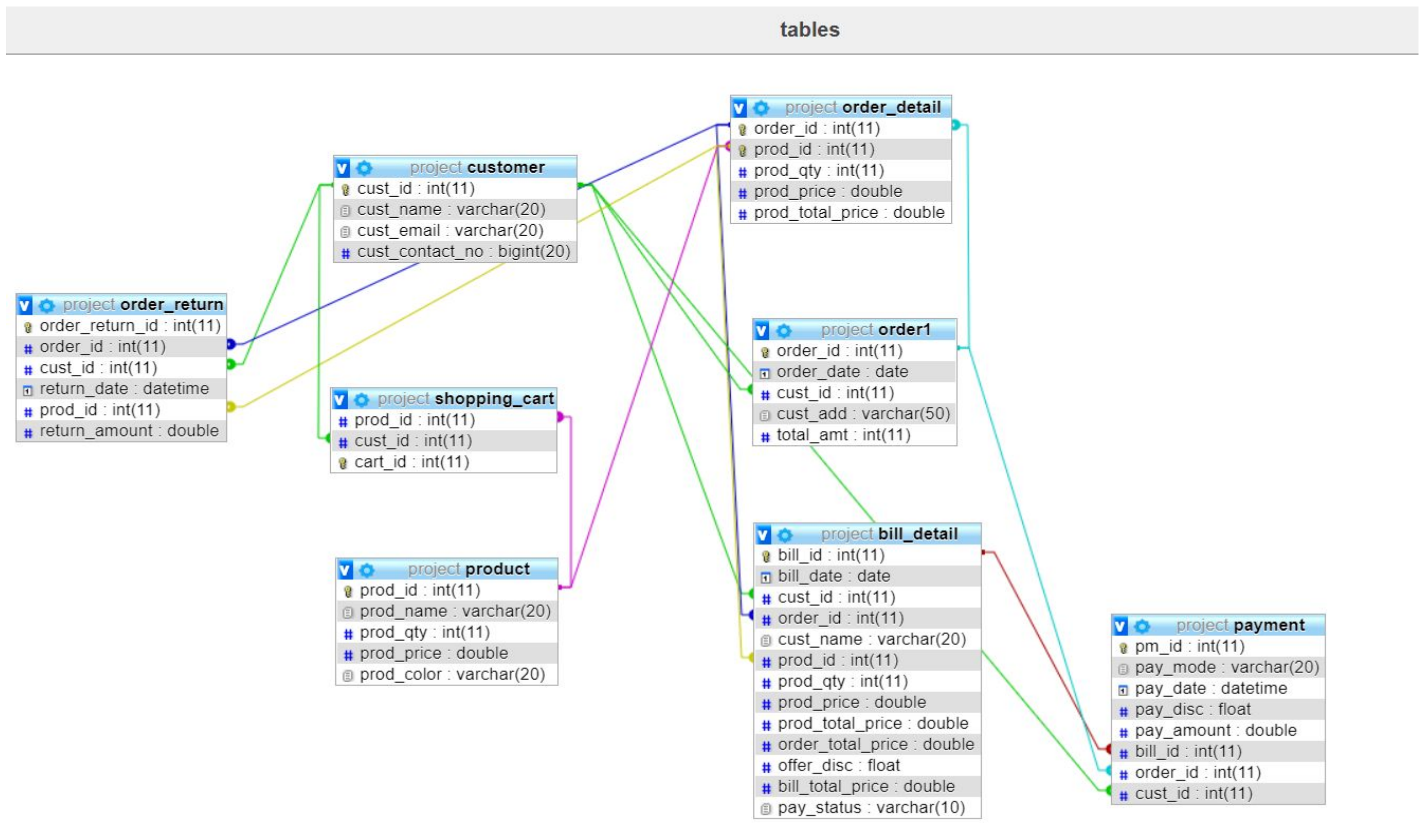
Description of Project :

This is a small project for Online Shopping System. The basic idea is that the customer’s can buy products using online. And the administrator can enter the name and generate the receipt of the purchased product and the administrator can also view the yearly,monthly and daily reports of the products.

This project is an attempt to provide the advantages of online shopping to customers of a real shop. It helps buying the products in the shop anywhere through internet by using an android device. Thus the customer will get the service of online shopping and home delivery from his favorite shop. This system can be implemented to any shop in the locality or to multinational branded shops having retail outlet chains.

The central concept of the application is to allow the customer to shop virtually using the Internet and allow customers to buy the items and articles of their desire from the store. The information pertaining to the products are stores on the MySQL database. The Server process the customers and the items are shipped to the address submitted by them.
The system was designed into two modules, first is administrator who maintains and updates the information of the product. And the second is customer who wish to buy products. Order which are placed by the customer, will store into the database and according to the order detail, bill will be generated and the payment will be paid by the customer. According to our system, administrator can view the different records of the products, orders, bill details and the payments. Like, year wise order details , day wise placed orders , maximum pay mode used by customers , over all order details , etc.

Relational Diagram :



Stored Procedures :

(1). This procedure will insert the product details into the 'order_detail' table. Only those customer can add the data who has placed the order.

```
Delimiter $
drop procedure insert_order_detail$
create procedure insert_order_detail(in order_id int,in pid int,in qty int)
        begin
        declare prod_total_price,prod_final_price,price, order_total_price
double;
        declare discount float;
        declare id int;
        declare b int;

        declare cur1 cursor for select prod_id, prod_price from product where
prod_id = pid;
        declare continue handler for not found set b = 1;
        set order_total_price = 0;
        open cur1;
        set b = 0;

        fetch cur1 into id, price;
        while b = 0 do

                set prod_total_price = price*qty;

                insert into order_detail values(order_id, pid, qty, price,
prod_total_price);
                fetch cur1 into id,price;
```

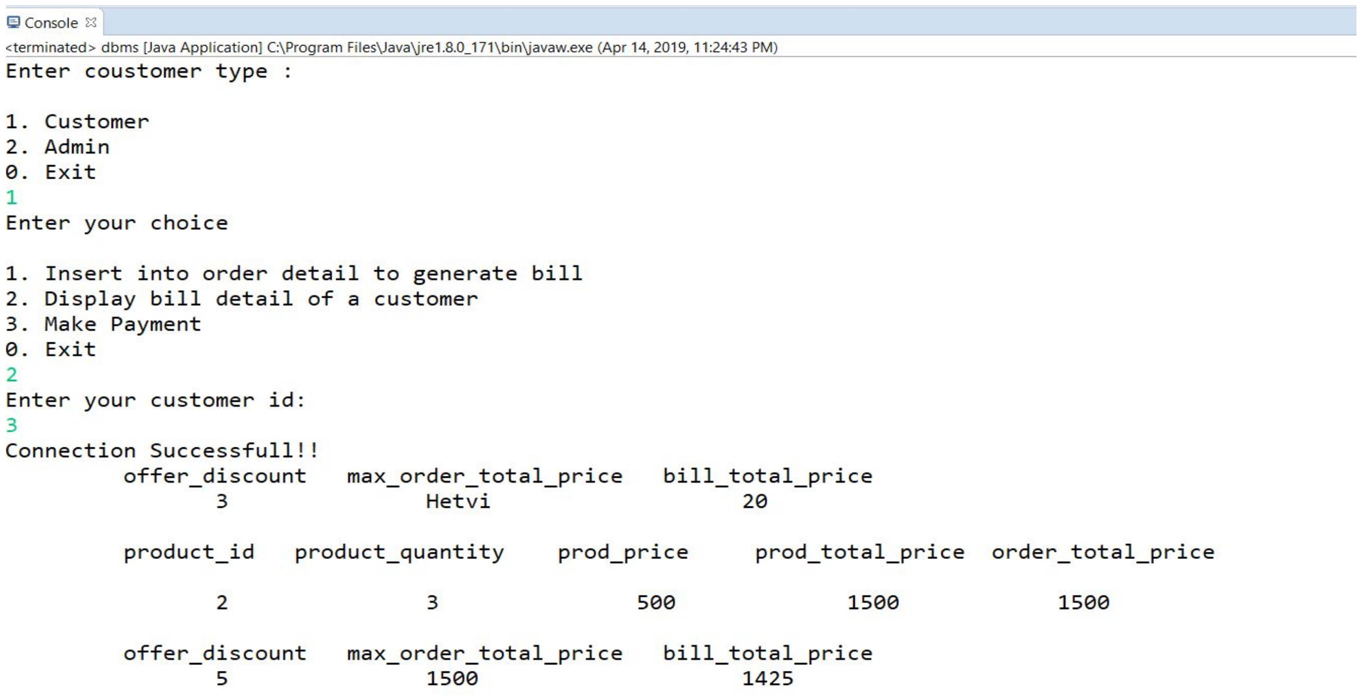
```
          end while;

          close cur1;

      end$

      call insert_order_detail(19, 2, 2)$
```

(2) . This procedure will display the year wise order details to the administrator.

```
Delimiter $
drop procedure year_wise_order_detail$
create procedure year_wise_order_detail(year year)
begin
          select year;
          #select * from order1 where year = extract(year from
order1.order_date);
          select o.order_id, o.order_date, o.cust_id, o.cust_add, od.prod_id,
od.prod_qty, od.prod_price,
                    od.prod_total_price, o.total_amt from order1 o inner join
order_detail od
                     on o.order_id = od.order_id and year = extract(year from
o.order_date) ;
end$

call year_wise_order_detail(2000)$
```

(3). This procedure will display the customer id wise bill detail to the customer.

```
Delimiter $
drop procedure cust_wise_bill_detail$
create procedure cust_wise_bill_detail(in c_id int)
begin
        select distinct cust_id,cust_name,order_id from bill_detail where c_id
= cust_id;
        select prod_id, prod_qty, prod_price, prod_total_price,
                order_total_price from bill_detail where c_id = cust_id;
        select offer_disc,max(order_total_price),bill_total_price from
bill_detail where c_id = cust_id;



end$

call cust_wise_bill_detail(3)$
```

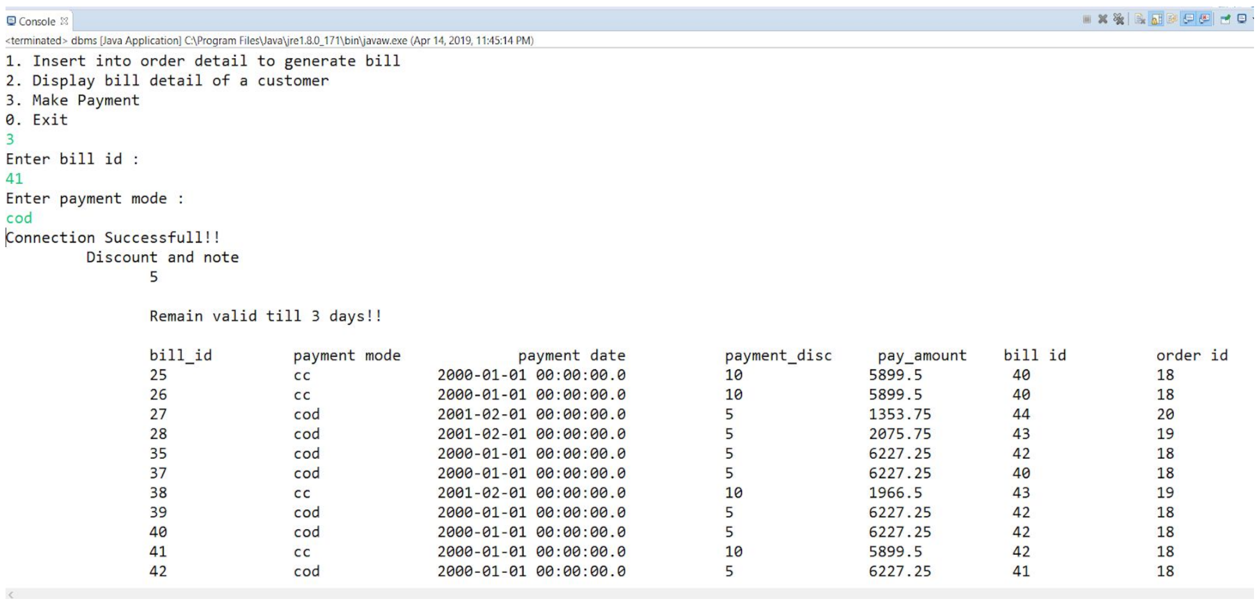


```
Console ☒
<terminated> dbms [Java Application] C:\Program Files\Java\jre1.8.0_171\bin\javaw.exe (Apr 14, 2019, 11:24:43 PM)
Enter coustomer type :

1. Customer
2. Admin
0. Exit
1
Enter your choice

1. Insert into order detail to generate bill
2. Display bill detail of a customer
3. Make Payment
0. Exit
2
Enter your customer id:
3
Connection Successfull!!
          offer_discount   max_order_total_price   bill_total_price
                3               Hetvi                     20

          product_id   product_quantity    prod_price     prod_total_price  order_total_price

                2               3               500             1500            1500

          offer_discount   max_order_total_price   bill_total_price
                5               1500                     1425
```

(4). This procedure will take bill id and the payment mode input from the customer and according to that insert the data into the payment table and according to the bill total amount , offer discount will be generated and display the final amount.

```
Delimiter $
drop procedure payment$
create procedure payment(in b_id int, in pay_mode varchar(20))
begin
        declare cid,oid int;
        declare btp double;
        declare disc float;
        declare note varchar(40);
        declare bdate date;
        declare pm_amount double;


        select bill_total_price into btp from bill_detail where b_id = bill_id;

        select cust_id into cid from bill_detail where b_id = bill_id;

        select order_id into oid from bill_detail where b_id = bill_id;

        select bill_date into bdate from bill_detail where b_id = bill_id;

        if btp >1000 then
                if pay_mode = 'cc' then
                        set disc = 10;
                        set note = 'Remain valid till 5 days!!';
                        select disc 'Discount' ;
                        select note 'Note';
```

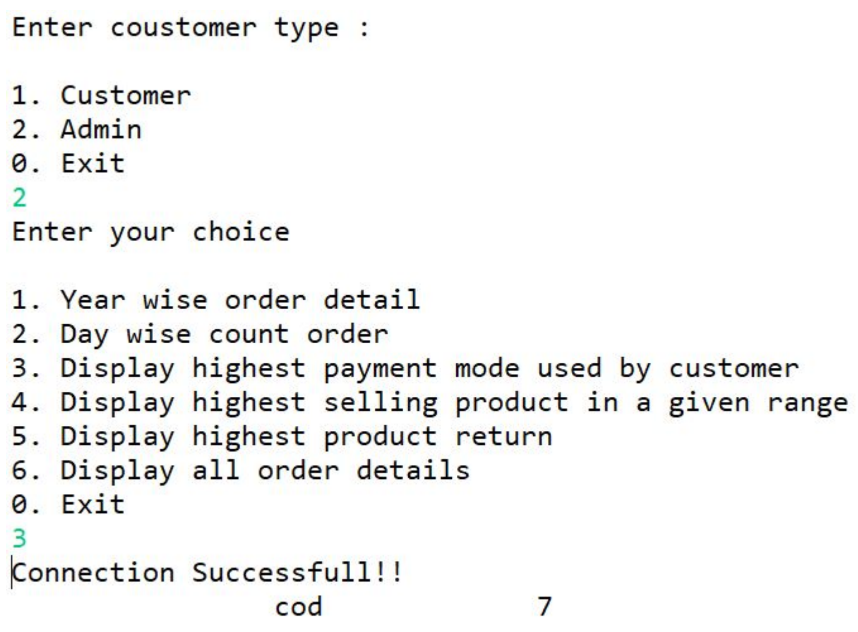
```
		elseif pay_mode = 'cod' then
			set disc = 5;
			set note = 'Remain valid till 3 days!!';
			select disc 'Discount' ;
			select note 'Note';

		else
			set disc = 0;
			set note = 'No Discount';

		end if;
	end if;

	set pm_amount = btp - (btp*disc/100);

	insert into
payment(pay_mode,pay_date,pay_disc,pay_amount,bill_id,order_id,cust_id)
values(pay_mode,bdate,disc,pm_amount,b_id,oid,cid);
end$

call payment(23, 'cc')$
```



```
Console
<terminated> dbms [Java Application] C:\Program Files\Java\jre1.8.0_171\bin\javaw.exe (Apr 14, 2019, 11:45:14 PM)
1. Insert into order detail to generate bill
2. Display bill detail of a customer
3. Make Payment
0. Exit
3
Enter bill id :
41
Enter payment mode :
cod
Connection Successfull!!
         Discount and note
                5

                Remain valid till 3 days!!

                bill_id         payment mode             payment date            payment_disc     pay_amount     bill id          order id
                25              cc              2000-01-01 00:00:00.0            10               5899.5          40               18
                26              cc              2000-01-01 00:00:00.0            10               5899.5          40               18
                27              cod             2001-02-01 00:00:00.0            5                1353.75         44               20
                28              cod             2001-02-01 00:00:00.0            5                2075.75         43               19
                35              cod             2000-01-01 00:00:00.0            5                6227.25         42               18
                37              cod             2000-01-01 00:00:00.0            5                6227.25         40               18
                38              cc              2001-02-01 00:00:00.0            10               1966.5          43               19
                39              cod             2000-01-01 00:00:00.0            5                6227.25         42               18
                40              cod             2000-01-01 00:00:00.0            5                6227.25         42               18
                41              cc              2000-01-01 00:00:00.0            10               5899.5          42               18
                42              cod             2000-01-01 00:00:00.0            5                6227.25         41               18
```

(5). This procedure will display the payment_mode which was more Used by the customers.

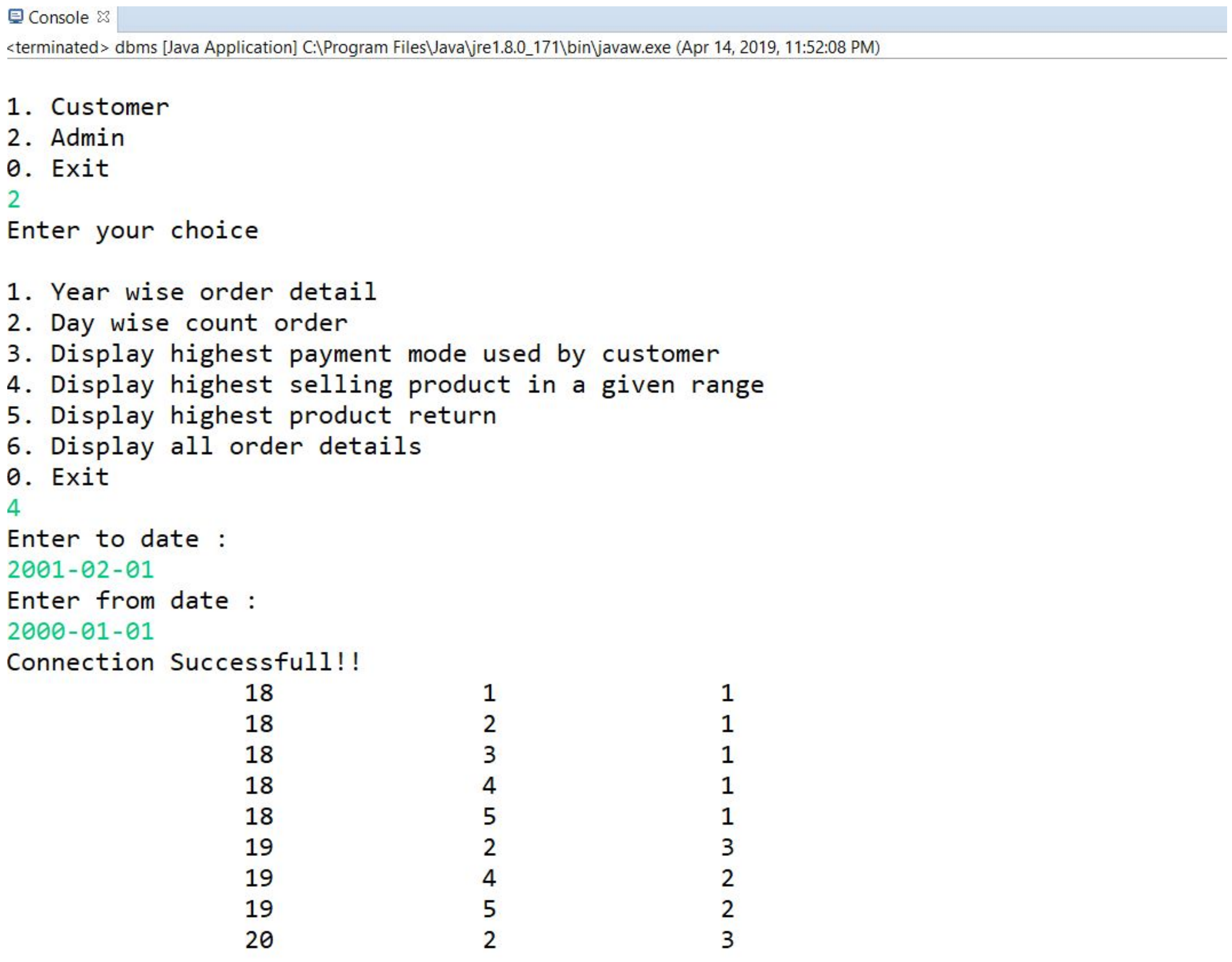
```
Delimiter $
drop procedure pay_mode$
create procedure pay_mode()
    begin
    declare cnt int;

    select count(pay_mode) into cnt from payment group by pay_mode order
by count(pay_mode) desc limit 1;

    select pay_mode, count(pay_mode) from payment group by pay_mode
having count(pay_mode) = cnt order by count(pay_mode);
end$

call pay_mode$
```



```
Enter coustomer type :

1. Customer
2. Admin
0. Exit
2
Enter your choice

1. Year wise order detail
2. Day wise count order
3. Display highest payment mode used by customer
4. Display highest selling product in a given range
5. Display highest product return
6. Display all order details
0. Exit
3
Connection Successfull!!
                cod             7
```

(6) . This procedure will display the product which was highest sold.

```
Delimiter $
drop procedure highest_sell$
create procedure highest_sell(in to_date date,in from_date date)
begin
        declare o_date date;
        declare b,cnt int;
        declare cur1 cursor for select order_date from order1,order_detail
where
                order1.order_id = order_detail.order_id and order_date >=
to_date and order_date <= from_date;
        declare continue handler for not found set b = 1;
        open cur1;
        set b = 0;
        fetch cur1 into o_date;
        select order_id,prod_id,prod_qty from order_detail;

        select sum(order_detail.prod_qty) into cnt from order_detail,product
where order_detail.prod_id = product.prod_id
                group by order_detail.prod_id order by
sum(order_detail.prod_qty) desc limit 1;


        select order_detail.prod_id as 'Product ID', product.prod_name as
'Product Name',
                sum(order_detail.prod_qty) as 'Maximum quantity sold' from
order_detail,product
                where order_detail.prod_id = product.prod_id group by
order_detail.prod_id
                having sum(order_detail.prod_qty) = cnt order by
sum(order_detail.prod_qty);

        close cur1;

end$

call highest_sell('2000-02-02','2002-02-02')$
```



```
<terminated> dbms [Java Application] C:\Program Files\Java\jre1.8.0_171\bin\javaw.exe (Apr 14, 2019, 11:52:08 PM)

1. Customer
2. Admin
0. Exit
2
Enter your choice

1. Year wise order detail
2. Day wise count order
3. Display highest payment mode used by customer
4. Display highest selling product in a given range
5. Display highest product return
6. Display all order details
0. Exit
4
Enter to date :
2001-02-01
Enter from date :
2000-01-01
Connection Successfull!!
                18              1               1
                18              2               1
                18              3               1
                18              4               1
                18              5               1
                19              2               3
                19              4               2
                19              5               2
                20              2               3
```

(7). This procedure will display the product which was highest return by the different or same customers.

```
Delimiter $
drop procedure highest_prod_return$
create procedure highest_prod_return()
begin

        declare cnt int;
        select count(prod_id) into cnt from order_return group by prod_id
order by count(prod_id) desc limit 1;
        select prod_id, count(prod_id) from order_return group by prod_id
having count(prod_id) = cnt order by count(prod_id);

end$

call highest_prod_return$
```

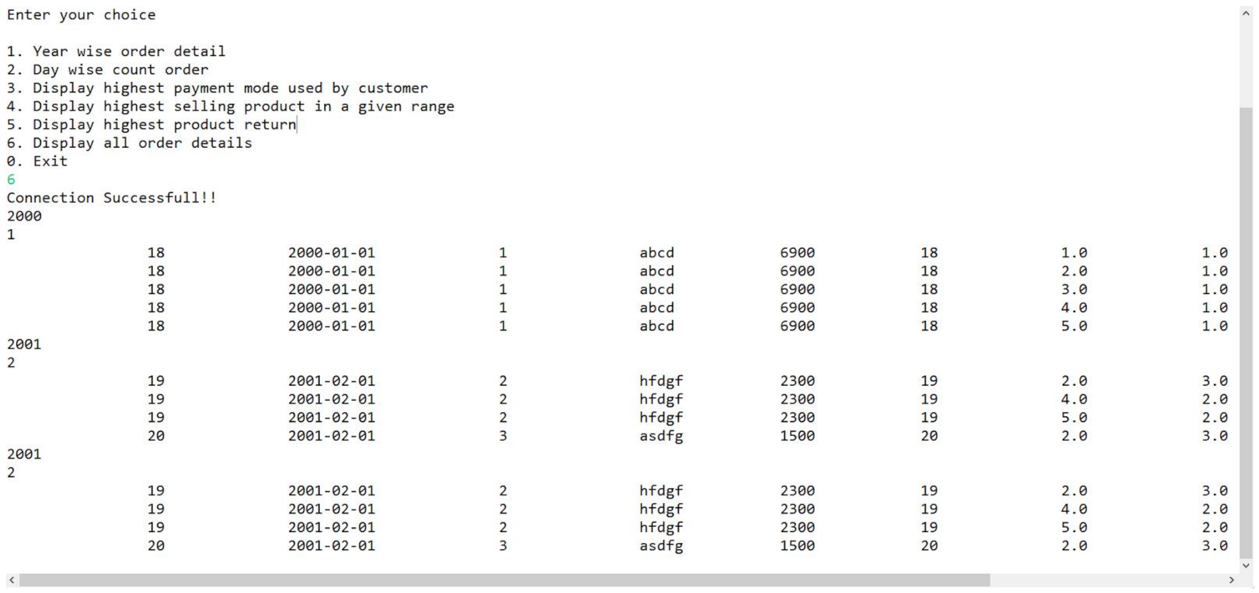
(8). This procedure will manage the products which are in the shopping Cart. As soon as the shopping cart product will add into the order detail, the product will be delete from the shopping cart.

```
Delimiter $
drop procedure manage_shopping_cart$
create procedure manage_shopping_cart(in pid int, in cid int)
begin
        delete from shopping_cart where prod_id = pid and cust_id =
cid;

end$

Call manage_shopping_cart$
```

(9). This procedure will display the overall order details based on year and month.

```
Delimiter $
drop procedure orderdetail$
create procedure orderdetail()
begin
```

```
        declare b int;
        declare odate date;
        declare cur1 cursor for select order1.order_date from order1 inner join
order_detail
                on order1.order_id= order_detail.order_id group by
order_detail.order_id order by order1.order_date;
        declare continue handler for not found set b = 1;
        open cur1;
        set b = 0;
        fetch cur1 into odate;
        while b = 0 do

                select extract(year from odate);
                select extract(month from odate);
                select * from order1 inner join order_detail on order1.order_id
= order_detail.order_id
                        where order1.order_date = odate;
                fetch cur1 into odate;
        end while;
        close cur1;
end$

call orderdetail$
```



```
Enter your choice

1. Year wise order detail
2. Day wise count order
3. Display highest payment mode used by customer
4. Display highest selling product in a given range
5. Display highest product return
6. Display all order details
0. Exit
6
Connection Successfull!!
2000
1
                18              2000-01-01              1               abcd            6900            18              1.0             1.0
                18              2000-01-01              1               abcd            6900            18              2.0             1.0
                18              2000-01-01              1               abcd            6900            18              3.0             1.0
                18              2000-01-01              1               abcd            6900            18              4.0             1.0
                18              2000-01-01              1               abcd            6900            18              5.0             1.0
2001
2
                19              2001-02-01              2               hfdgf           2300            19              2.0             3.0
                19              2001-02-01              2               hfdgf           2300            19              4.0             2.0
                19              2001-02-01              2               hfdgf           2300            19              5.0             2.0
                20              2001-02-01              3               asdfg           1500            20              2.0             3.0
2001
2
                19              2001-02-01              2               hfdgf           2300            19              2.0             3.0
                19              2001-02-01              2               hfdgf           2300            19              4.0             2.0
                19              2001-02-01              2               hfdgf           2300            19              5.0             2.0
                20              2001-02-01              3               asdfg           1500            20              2.0             3.0
```

Stored Function :

(1). This function will return the no. of orders placed in a given day.

```
Delimiter $
drop function day_wise_cnt_order$
create function day_wise_cnt_order(date date) returns int
begin
        declare order_cnt int;
        #select date;
        select count(distinct order1.order_id) into order_cnt from order1 inner
join order_detail
            on order1.order_id = order_detail.order_id and date =
order1.order_date group by order_date;
        return (order_cnt);
end$

Select day_wise_cnt_order('2001-02-02')$
```

```
Enter coustomer type :

1. Customer
2. Admin
0. Exit
2
Enter your choice

1. Year wise order detail
2. Day wise count order
3. Display highest payment mode used by customer
4. Display highest selling product in a given range
5. Display highest product return
6. Display all order details
0. Exit
2
Enter date :
2001-02-01
Connection Successfull!!
2
```

Stored Triggers :

(1). This trigger will fired after insert on the order detail table. This trigger Will update the total amount value in the order table, update the , Product quantity in the product table and insert data into the bill details.

```
delimiter $
drop trigger err_ins1$
create trigger err_ins1 after insert on order_detail
for each row
begin
        declare c_id int;
        declare c_name varchar(20);
        declare t_amt, btp double;
        declare b_date date;
        declare day, month int;
        declare disc float;

        update order1 set total_amt = total_amt + new.prod_total_price where
order_id=new.order_id;
        update product set prod_qty = prod_qty - new.prod_qty where prod_id
= new.prod_id;
        #delete from order1 where total_amt = 0;

        select cust_id into c_id from order1 where order1.order_id =
new.order_id;

        select customer.cust_name into c_name from customer, order1 where
customer.cust_id  = order1.cust_id
                and order1.order_id = new.order_id;
        select total_amt into t_amt from order1 where order1.order_id =
new.order_id;

        insert into bill_detail(cust_id, order_id, cust_name, prod_id, prod_qty,
prod_price, prod_total_price,
                order_total_price) values(c_id, new.order_id, c_name,
new.prod_id, new.prod_qty,
                new.prod_price, new.prod_total_price, t_amt);

        select order_date into b_date from order1 where order1.order_id =
new.order_id;

      update bill_detail set bill_date = b_date where order_id = new.order_id;

        set day = extract(day from(b_date));
        set month = extract(month from(b_date));

        if day = 15 and month = 8 then
                set disc = 10.0;
        elseif day > 24 and day < 28 and month = 10 then
                set disc = 15.0;
        elseif day = 14 and month = 1 then
                set disc = 20.0;
        else
                set disc = 5.0;
        end if;

        update bill_detail set offer_disc = disc where order_id = new.order_id;

        set btp = t_amt - (((t_amt)*disc)/100);
        update bill_detail set bill_total_price = btp where order_id =
new.order_id;
```

```
        call manage_shopping_cart(new.prod_id, c_id);

    end$
```

(2). This trigger will fired if the customer's desired quantity is greater than the total available quantity.

```
    delimiter $
    drop trigger err_ins2$
    create trigger err_ins2 before insert on order_detail
    for each row
    begin

        declare msg varchar(128);
        declare p_qty int;

        select distinctrow product.prod_qty into p_qty from product inner join
    order_detail on new.prod_id = product.prod_id;
        #set pid = select prod_id from product where prod_id = new.prod_id;
        if p_qty < new.prod_qty then
            set msg = 'Not enough quantity.....';
        elseif new.prod_qty < 0 then
            set msg = 'Quantity can not be negative.....';
        end if;
            signal sqlstate '45001' set message_text = msg;
    end$
```

(3). This trigger will update the payment status after the payment make by the customer.

```
Delimiter $
drop trigger pay_status$
create trigger pay_status after insert on payment
for each row
begin

        update bill_detail set pay_status = 'Paid' where bill_id = new.bill_id;

end$
```

(4). This trigger will update the product quantity in the product table after the order return.

```
delimiter $
drop trigger qty_return$
create trigger qty_return after insert on order_return
for each row
begin
        declare p_qty int;

        select prod_qty into p_qty from order_detail where
order_detail.order_id = new.order_id and order_detail.prod_id =
new.prod_id;

        update product set prod_qty = prod_qty + p_qty where prod_id =
new.prod_id;

end$
```