620 Project

Feature 1:

```
drop sequence customer_seq;
create sequence customer_seq
start with 6
increment by 1;
--------
set serveroutput on;
create or replace procedure new_customer (v_cname in varchar2,
v_address in varchar2,v_state in varchar2, v_zipcode in number, v_email in varchar2)
is
v_count number;
v_credit number:=0;
begin
select count(*) into v_count from customer where email=v_email;
if (v_count=1) then
dbms_output.put_line ('The client already exists');
update customer
set address=v_address,state=v_state,zipcode=v_zipcode
where email=v_email;
else
insert into customer(CUSTOMERID,CNAME,ADDRESS,ZIPCODE,STATE,EMAIL,CREDIT)
values(customer_seq.nextval,v_cname,v_address,v_zipcode,v_state,v_email,v_credit);
dbms_output.put_line('New Customer ID.');
end if;
end;
/
BEGIN
new_customer('Mary','Hazlett Ave','MD',21229,'mary@gmail.com');
END;
/
Output: The client already exists
BEGIN
new_customer('Rakesh','Hazlett Ave','MD',21229,'rakesh.devagalla@gmail.com');
END;
/
Output: New Customer ID.
Select  * from customer;
```

Feature 2:

```
set serveroutput on;
create or replace procedure check_customer(v_email in varchar2)
is
v_count number;
V_CNAME varchar2(100);
v_address varchar2(100);
v_state varchar2(100);
v_zip number;
v_credit number;
v_ordercount number;
```

```
v_totalcost number;
begin
select count(*) into v_count from customer where email=v_email;
if v_count=0 then
dbms_output.put_line('No Such Customer');
else
select cname, address,state,zipcode,STATE,CREDIT,count(order_status),sum(totalcost) INTO
V_CNAME,v_address,v_state,v_zip,v_state,v_credit,v_ordercount,v_totalcost from customer c
join order_table o on c.customerid=o.customerid
where order_status='delivered' and o.ordertime>=ADD_MONTHS(sysdate, -6) and
email=v_email group by cname, address,state,zipcode,STATE,EMAIL,CREDIT;
dbms_output.put_line('Customer Name: '||v_cname||chr(10)||'Address: '
||v_address||chr(10)||'State: '||v_state ||chr(10)||'Zip: '|| v_zip ||chr(10)||'Mail id :
'||v_email||chr(10)||'Credit: ' ||v_credit ||chr(10)||'Total order in last 6 months: '||
v_ordercount||chr(10)||'Total cost for the orders in past 6 months: ' || v_totalcost);
end if;
end;
/
BEGIN
check_customer('mary@gmail.com');
END;
/
Output: Prints existed customer details
BEGIN
check_customer('gurleen@gmail.com');
END;
/
Output: No Such Customer
Select * from customer;
```

Feature 3:

```
set serveroutput on;
create or replace procedure restaurant_by_category (v_categoryname in varchar)
is
v_restaurantname varchar2(50);
v_avgreviewscore decimal(4,2);
v_avgwaitime decimal(4,2);
v_zipcode number;
begin
select restaurantname,avgreviewscore,avgwaittime,rzipcode into
v_restaurantname,v_avgreviewscore,v_avgwaitime,v_zipcode
from restaurants r join rcategory rc on rc.restaurantid=r.restaurantid
where lower(substr(category,0,4)) = v_categoryname and lower(current_status)='open';
dbms_output.put_line('Restaurant name: '||v_restaurantname||' Avgreviewscore:
'||v_avgreviewscore||' Avgwaitime: '||v_avgwaitime||' Zipcode: '||v_zipcode);
end;
/
begin
restaurant_by_category('seaf');
end;
/
```

```
Output: Restaurant name: '||v_restaurantname||' Avgreviewscore: '||v_avgreviewscore||'
Avgwaitime: '||v_avgwaitime||' Zipcode: '||v_zipcode
```
Feature 4:
```
create or replace procedure dishes_restaurant (v_restaurantid in number)
is
v_count number;
begin
select count(*) into v_count from restaurants where restaurantid=v_restaurantid;
if v_count=0 then
dbms_output.put_line('No such Restaurant');
else
for i in(select dishanme,price from dishes join restaurants on
dishes.restaurantid=restaurants.restaurantid where restaurants.restaurantid=v_restaurantid)
loop
dbms_output.put_line('Dish name: '|| i.dishanme||' Price: '||i.price||'$');
end loop;
end if;
end;
/
begin
dishes_restaurant(101);
end;
/
```
Output:

Feature 5:
```
set serveroutput on;
create or replace procedure check_dishes_cart
(v_cartid in number) as
v_count number;
v_dishname varchar2(20);
v_price decimal(4,2);
v_quantity number;
begin
select count(*) into v_count from cartdish where cartid=v_cartid;
if v_count=0 then
dbms_output.put_line('Invalid Cart ID');
else
select dishanme,price,quantity into v_dishname,v_price,v_quantity from dishes join cartdish on
dishes.dishid=cartdish.dishid
where cartid=v_cartid;
dbms_output.put_line('Dish name: '|| v_dishname||chr(10)||'Price: '||v_price||chr(10)||'Quantity: '
||v_quantity);
end if;
end;
/
begin
check_dishes_cart(8);
End;
/
```
Output:

Feature 6:

```
set serveroutput on;
create or replace procedure remove_dish_cart (v_dishid in number,
v_cartid in number) as
v_count number;
v_quantity number;
begin
select count(*) into v_count from cartdish where cartid=v_cartid and dishid=v_dishid;
if v_count=0 then
dbms_output.put_line('Invalid input');
else
select quantity into v_quantity from cartdish where cartid=v_cartid and dishid=v_dishid;
if (v_quantity>1)then
update cartdish
set quantity=quantity-1
where cartid=v_cartid and dishid=v_dishid;
dbms_output.put_line('quantity reduced');
elsif (v_quantity=1) then
delete from cartdish where cartid=v_cartid and dishid=v_dishid;
dbms_output.put_line('Dish removed');
end if;
end if;
end;
/
begin
remove_dish_cart(41,7);
end;
select * from cartdish;
rollback;
```

Feature 7:

```
create or replace procedure upstatus(orid number , status number , intime timestamp)
as
cnt number;
custid number;
ordstatus varchar(100);
totcost number;
pmethod varchar2(50);
begin
select count(*) into cnt
from order_table
where orderid=orid;
if cnt=0 then
dbms_output.put_line('Invalid order id');
else
select customerid,order_status,totalcost into custid , ordstatus , totcost
from order_table where orderid=orid;
select paymentmethod into pmethod
from payments where orderid=orid;
if status = 2 then
update order_table
```

```
set order_status = 'delivered'
where orderid = orid;
insert into message values ( messageseq.nextval , custid , intime , 'Your order' ||orid||' has been
delivered.');
elsif status = 3 then
update order_table
set order_status = 'cancelled'
where orderid = orid;
insert into payments values ( paymentsseq.nextval , custid , intime , orid , -totcost , pmethod);
insert into message values ( messageseq.nextval, custid , intime , 'Your order'|| orid|| ' has been
cancelled and refund issued');
else
dbms_output.put_line(' No additional changes required');
end if;
end if;
end;
/
execute upstatus ( 666 , 1 , timestamp '2022-02-08 02:20:00.00');
output : Invalid order id
execute upstatus ( 456 , 1 , timestamp '2022-02-08 02:20:00.00');
output : No additional changes required.
execute upstatus ( 579 , 2 , timestamp '2022-02-08 02:20:00.00');
select * from order_table;
select * from message;
execute upstatus ( 579 , 3 , timestamp '2022-02-08 02:20:00.00');
select * from order_table;
select * from message;
select * from payments;
```

Feature 8:

```
create or replace procedure p_review (v_customerid in number,
                                v_restaurantid in number,
                                v_reviewdate in date,
                                v_reviewscore in number,
                                v_comments in varchar2)
is
v_count number;
v_reviewid number :=seq_reviewid.nextval;
begin
select count(*) into v_count from review where customerid=v_customerid;
if v_count=0 then
dbms_output.put_line('Invalid Customer ID');
else
select count(*) into v_count from review where restaurantid=v_restaurantid;
if v_count=0 then
dbms_output.put_line('Invalid Restaurant ID');
else
insert into review
values(seq_reviewid.nextval,v_customerid,v_restaurantid,v_reviewdate,v_reviewscore,v_comm
ents);
update restaurants
```

```
set avgreviewscore= (select sum(reviewscore)/count(reviewscore) from review where
restaurantid=v_restaurantid)
where restaurants.restaurantid=v_restaurantid;
end if;
end if;
end;
/
—-Sequence creation—
drop sequence seq_reviewid;
create sequence seq_reviewid
start with 65
increment by 1;
--Test cases—
EXECUTE p_review(3,101
,'13-FEB-21',10,'too good');
select * from review;
select * from restaurants;
------------------------------
```

Feature 9:

```
create or replace procedure p_review_restaurant(v_restaurantid in number)
is
v_count number;
cursor c_restaurant is select
restaurantname,review.reviewscore,review.comments,review.reviewdate
from restaurants join review on review.restaurantid = restaurants.restaurantid where
restaurants.restaurantid=v_restaurantid;
begin
select count(*) into v_count from review where restaurantid=v_restaurantid;
if (v_count=0) then
dbms_output.put_line('Invalid restaurant ID');
else
for i in c_restaurant loop
dbms_output.put_line('The ' || i.restaurantname|| ' has rated '|| i.reviewscore ||' with comments '
|| i.comments|| ' on ' || i.reviewdate);
end loop;
end if;
end;
/
-–Test cases—
exec p_review_restaurant(102);
exec p_review_restaurant(10);
—---------------------------
```

Group Features

Feature 10:

```
    drop sequence cartdishvalues;
    create sequence cartdishvalues
    start with 65
    increment by 1;

    set serveroutput on;
    create or replace procedure dish_shopping_cart
    (v_customerid in number,
    v_restaurantid in number,
    v_dishid in number)
    is
    v_count number;
    d_count number;
    c_count number;
    cd_count number;
    v_quantity number;
    v_cartid number;
    v_current_status varchar2(20);
    begin
    select count(*) into v_count from customer
    where customerid=v_customerid;
    if v_count=0 then
    dbms_output.put_line('No such Customer');
    else
    select current_status into v_current_status from restaurants
    where restaurantid=v_restaurantid;
        if v_current_status='Closed' then
        dbms_output.put_line('The restaurant is closed');
        else
            select count(*) into d_count from dishes d join restaurants r
            on d.restaurantid=r.restaurantid
            where d.dishid=v_dishid and d.restaurantid=v_restaurantid;
                if d_count=0 then
                    dbms_output.put_line('Invalid Dish ID');
                else
                select count(*) into c_count from cart where customerid=v_customerid;
                    if c_count=0 then
                insert into cart(cartid,customerid,restaurantid)
values(cartvalues.nextval,v_customerid,v_restaurantid);
                else
                select count(*) into cd_count from dishes d join cartdish cd
                on d.dishid=cd.dishid where d.dishid=v_dishid;
                if cd_count=0 then
                insert into cartdish values(cartdishvalues.nextval,v_cartid,v_dishid,1);
                else
                select quantity into v_quantity from dishes d join cartdish cd
                on d.dishid=cd.dishid where d.dishid=v_dishid;
                update cartdish
                set quantity=v_quantity+1
                where dishid=v_dishid;
                end if;
            end if;
        end if;
    end if;
    end if;
    exception
    when no_data_found then
    dbms_output.put_line(' No such restaurant');
    end;
/
begin
DISH_SHOPPING_CART(1,104,43);
end;
/
begin
DISH_SHOPPING_CART(10,101,40);
end;
/
begin
DISH_SHOPPING_CART(3,110,42);
end;
/
begin
DISH_SHOPPING_CART(5,102,50);
end;
/
```

Feature 11:

```
set serveroutput on;
create or replace procedure p_cart_check(v_cartid number ,
v_checkout_time timestamp,
v_deliverymethod number) as
deliveryfee number;
order_amount number;
tax_amount number;
totalamount number;
v_count number;
cart_price number;
czip number;
rzip number;
rtax number;
distyp varchar2(100);
damount number;
dis_amt number;
BEGIN
SELECT count(*) INTO v_count
FROM cart
WHERE cartid = v_cartid;
IF v_count = 0 THEN
dbms_output.put_line('invalid customer id');
ELSE
select SUM(cd.quantity * d.price) into cart_price
from cart c join cartdish cd
on c.cartid = cd.cartid join dishes d
on c.restaurantid = d.restaurantid
and cd.dishid = d.dishid
where c.cartid = v_cartid
group by c.cartid;
```

```
dbms_output.put_line(cart_price);
select customer.zipcode, r.rzipcode, d.discount_type, d.DISCOUNTAMOUNT, t.rate
into czip, rzip, distyp, damount, rtax
from cart c join customer
on c.customerid = customer.customerid join restaurants r
on c.restaurantid = r.restaurantid
left join storediscounts sd
on sd.customerid = c.customerid join discounts d
on d.discountid = sd.discountid join tax_table t
on t.state = r.rstate
where c.cartid = v_cartid
and sd.dstartdate <= trunc(v_checkout_time) and sd.denddate >= trunc(v_checkout_time);
IF czip <> rzip
then deliveryfee := 5;
ELSE
deliveryfee := 2;
END IF;
IF v_deliverymethod = 2
then deliveryfee := 0;
END IF;
IF distyp = 1
then deliveryfee := 0;
dis_amt := 0;
ELSIF distyp = 2 then
dis_amt := cart_price * damount;
ELSE
dis_amt := damount;
END IF;
order_amount := cart_price - dis_amt;
tax_amount := order_amount * rtax/100;
totalamount := order_amount + deliveryfee + tax_amount;
dbms_output.put_line('Order amount: '|| order_amount || ' Total tax:  ' ||tax_amount || ' Total tax '
||totalamount);
END IF;
exception
when no_data_found then
dbms_output.put_line('Invalid Inputs');
END;
/
select * from storediscounts;
begin
p_cart_check(6,'13-JAN-20',1);
end;
/
begin
p_cart_check(8,'10-MAY-22',3);
end;
/
```

Feature 13:

```
set serveroutput on;
create or replace procedure advancedsearch (v_customerid in number,
                                        v_category_name in varchar,
                                        v_reviewscore in number,
                                        v_waittime in decimal)
is
v_count number;
v_RESTAURANTNAME varchar2(50);
v_RADDRESS varchar2(100);
v_CURRENT_STATUS varchar2(10);
p_AVGREVIEWSCORE number;
v_RZIPCODE number;
v_AVGWAITTIME number;
begin
select count(*) into v_count from customer where customerid=v_customerid;
if v_count=0 then
dbms_output.put_line('Invalid Customer ID');
else
select
RESTAURANTNAME,RADDRESS,CURRENT_STATUS,AVGREVIEWSCORE,RZIPCODE,AV
GWAITTIME into
v_RESTAURANTNAME,v_RADDRESS,v_CURRENT_STATUS,p_AVGREVIEWSCORE,v_RZI
PCODE,v_AVGWAITTIME from restaurants r join rcategory rc on rc.restaurantid=r.restaurantid
where category=v_category_name and avgreviewscore>=v_reviewscore and
avgwaittime<=v_waittime
and rzipcode = (select zipcode from customer where customerid=v_customerid) or
substr(rzipcode,1,4)=(select substr(zipcode,1,4) from customer where
customerid=v_customerid);
dbms_output.put_line(v_restaurantname||' ' ||v_raddress||' '||v_current_status||' '
||p_AVGREVIEWSCORE||' ' ||v_RZIPCODE||' '||v_AVGWAITTIME);
end if;
end;
/
select * from customer;
select * from rcategory;
select * from restaurants;
begin
advancedsearch (9,'pizza',6,10.2);
end;
```

Feature 14:

```
set SERVEROUTPUT on;
f_restaurantname cf_restaurantname%rowtype;
begin
select count(*) into v_count from ccreate or replace procedure
Restaurant_recommendation(v_customerid in number)
is
v_count number;
cursor c_restaurantname is select restaurantname from restaurants r join order_table o on
r.restaurantid=o.restaurantid where customerid=v_customerid;
cursor c_customerid is select customerid from restaurants r join order_table o on
r.restaurantid=o.restaurantid
```

```
where customerid not in v_customerid and restaurantname in (select restaurantname from
restaurants r join order_table o on r.restaurantid=o.restaurantid where
customerid=v_customerid);
cursor cf_restaurantname is select r.restaurantid,r.restaurantname,r.avgreviewscore,r.raddress
from restaurants r join order_table o on r.restaurantid=o.restaurantid
where customerid in (select o.customerid from restaurants r join order_table o on
r.restaurantid=o.restaurantid where customerid not in v_customerid and restaurantname in
(select restaurantname from restaurants r join order_table o on r.restaurantid=o.restaurantid
where customerid=v_customerid) and
restaurantname not in( select restaurantname from restaurants r join order_table o on
r.restaurantid=o.restaurantid where customerid=v_customerid));
cv_customerid c_customerid%rowtype;
cv_restaurantname c_restaurantname%rowtype;
ustomer where customerid=v_customerid;
if (v_count=0)then
dbms_output.put_line('Inavlid Customer ID');
else
for i in c_customerid loop
dbms_output.put_line('Customer ID: '||i.customerid);
end loop;
for j in c_restaurantname loop
dbms_output.put_line('Restaurant Name: '||j.restaurantname);
end loop;
for k in cf_restaurantname loop
dbms_output.put_line('Restaurant ID: '||k.restaurantid||'Restaurant name:
'||k.restaurantname||'Average Review Score: '||k.avgreviewscore||'Restaurant address: '||
k.raddress);
end loop;
end if;
end;
/
select * from customer;
begin
Restaurant_recommendation(1);
end;
```