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\_comments);

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,AVGWAITTIME into v\_RESTAURANTNAME,v\_RADDRESS,v\_CURRENT\_STATUS,p\_AVGREVIEWSCORE,v\_RZIPCODE,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;