Project 2 Description

SQL_CODE

Q1)

--Droping and creating sequence

```
DROP SEQUENCE pur#;
CREATE SEQUENCE pur# START WITH 100001 INCREMENT BY 1;
DROP SEQUENCE log#;
CREATE SEQUENCE log# START WITH 1001 INCREMENT BY 1;
```

Q2)

--**in this function I have declared the varibles and assigned the varibles using ref_cursor for exception I have used error_message which will report exception and same thing is done in all other tables.**

Table 1)

```
set serveroutput on
declare
v_cid customers.CID%type;
v_name customers.name%type;
v_telephone# customers.telephone#%type;
v_vists_made customers.VISITS_MADE%type;
v_LAST_VISIT_DATE customers.LAST_VISIT_DATE%type;
v_refcursor sys_refcursor;
v_error_message varchar2(30);
procedure show_customers(e_refcursor out sys_refcursor,error_message OUT VARCHAR2)
is
begin
open e_refcursor for
select * from customers;
EXCEPTION
        WHEN NO_DATA_FOUND THEN
            error_message := 'No data found in table.';
    WHEN OTHERS THEN
                error_message := 'Unexpected error';
    RAISE;
end;
begin
show_customers(v_refcursor,v_error_message);
loop
fetch v_refcursor into v_cid,v_name,v_telephone#,v_vists_made,v_LAST_VISIT_DATE;
exit when v_refcursor%notfound;
dbms_output.put_line(v_cid||' '||v_name||' '||v_telephone#||' '||v_vists_made||'
'||v_LAST_VISIT_DATE);
end loop;
close v_refcursor;
end;
/
```

```
show errors
Table2)
set serveroutput on
declare
v_eid employees.eid%type;
v_name employees.name%type;
v_telephone# employees.telephone#%type;
v_email employees.email%type;
v_refcursor sys_refcursor;
v_error_message varchar2(30);
procedure show_employees(e_refcursor out sys_refcursor,error_message OUT VARCHAR2)
is
begin
open e_refcursor for
select * from employees;
EXCEPTION
        WHEN NO_DATA_FOUND THEN
            error_message := 'No data found in table.';
    WHEN OTHERS THEN
                error_message := 'Unexpected error';
    RAISE;
end;
begin
show_employees(v_refcursor,v_error_message);
loop
fetch v_refcursor into v_eid,v_name,v_telephone#,v_email;
exit when v_refcursor%notfound;
dbms_output.put_line(v_eid||' '||v_name||' '||v_telephone#||' '||v_email);
end loop;
close v_refcursor;
end;
/
show errors

Table 3)
set serveroutput on
declare
v_log# logs.LOG#%type;
v_name logs.USER_NAME%type;
v_OPERATION logs.OPERATION%type;
v_OP_TIME logs.OP_TIME%type;
v_TABLE_NAME logs.TABLE_NAME%type;
v_TUPLE_PKEY logs.TUPLE_PKEY%type;
v_refcursor sys_refcursor;
```

```
v_error_message varchar2(30);
procedure show_logs(e_refcursor out sys_refcursor,error_message OUT VARCHAR2)
is
begin
open e_refcursor for
select * from logs;
EXCEPTION
        WHEN NO_DATA_FOUND THEN
            dbms_output.put_line('no data found');
    WHEN OTHERS THEN
                error_message := 'Unexpected error';
    RAISE;
end;
begin
show_logs(v_refcursor,v_error_message);
loop
fetch v_refcursor into v_log#,v_name,v_OPERATION,v_OP_TIME,v_TABLE_NAME,v_TUPLE_PKEY;
exit when v_refcursor%notfound;
dbms_output.put_line(v_log#||' '||v_name||' '||v_OPERATION||' '||v_OP_TIME||'
'||v_TABLE_NAME||' '||v_TUPLE_PKEY);
end loop;
close v_refcursor;
end;
/
show errors
```

Table 4)

```
set serveroutput on
declare
v_pid products.PID%type;
v_name products.name%type;
v_QOH products.QOH%type;
v_QOH_THRESHOLD products.QOH_THRESHOLD%type;
v_REGULAR_PRICE products.REGULAR_PRICE%type;
v_DISCNT_RATE products.DISCNT_RATE%type;
v_refcursor sys_refcursor;
v_error_message varchar2(30);
procedure show_products(e_refcursor out sys_refcursor,error_message OUT VARCHAR2)
is
begin
open e_refcursor for
select * from products;
EXCEPTION
        WHEN NO_DATA_FOUND THEN
```

```
            error_message := 'No data found in table.';
    WHEN OTHERS THEN
                error_message := 'Unexpected error';
    RAISE;
end;
begin
show_products(v_refcursor,v_error_message);
loop
fetch v_refcursor into v_pid,v_name,v_QOH,v_QOH_THRESHOLD,v_REGULAR_PRICE,v_DISCNT_RATE;
exit when v_refcursor%notfound;
dbms_output.put_line(v_pid||' '||v_name||' '||v_QOH||' '||v_QOH_THRESHOLD||'
'||v_REGULAR_PRICE||' '||v_DISCNT_RATE);
end loop;
close v_refcursor;
end;
/
show_errors
```

Table 5)

```
set serveroutput on;
declare
v_PUR# purchases.PUR#%type;
v_EID purchases.EID%type;
v_pid purchases.pid%type;
v_cid purchases.cid%type;
v_PUR_DATE purchases.PUR_DATE%type;
v_QTY purchases.QTY%type;
v_UNIT_PRICE purchases.UNIT_PRICE%type;
v_TOTAL purchases.TOTAL%type;
v_SAVING purchases.SAVING%type;
v_refcursor sys_refcursor;
v_error_message varchar2(30);
procedure show_purchases(e_refcursor out sys_refcursor,error_message OUT VARCHAR2)
is
begin
open e_refcursor for
select * from purchases;
EXCEPTION
        WHEN NO_DATA_FOUND THEN
            error_message := 'No data found in table.';
    WHEN OTHERS THEN
                error_message := 'Unexpected error';
    RAISE;
end;
```

```
begin
show_purchases(v_refcursor,v_error_message);
loop
fetch v_refcursor into v_PUR#,v_EID,v_pid,v_cid,v_PUR_DATE,v_QTY,v_UNIT_PRICE,v_TOTAL,v_SAVING;
exit when v_refcursor%notfound;
dbms_output.put_line(v_PUR#||' '||v_EID||' '||v_pid||' '||v_cid||' '||v_PUR_DATE||' '||v_QTY||'
'||v_UNIT_PRICE||' '||v_TOTAL||' '||v_SAVING);
end loop;
close v_refcursor;
end;
/
show errors
```

Q3)

--I have made join b/w tables purchases and customers to fetch every purchases made by given cid for this I have used cursor.

```
set serveroutput on
declare
      v_name varchar(20);
      v_pid purchases.pid%type;
      v_pur_date purchases.pur_date%type;
      v_qty purchases.qty%type;
      v_unit_price purchases.unit_price%type;
      v_total purchases.total%type;
      v_refcursor sys_refcursor;
      procedure purchases_made(e_refcursor out sys_refcursor, mycid purchases.cid%type)
      is
      begin
      open e_refcursor for
      select name,pid,pur_date,qty,unit_price,total from purchases join customers on
purchases.cid=customers.cid where purchases.cid=mycid;
      end;
      begin
purchases_made(v_refcursor,'c001');
loop
fetch v_refcursor into  v_name,v_pid, v_pur_date, v_qty,v_unit_price,v_total;
exit when v_refcursor%notfound;
dbms_output.put_line(v_name||' '||v_pid||' '||v_pur_date||' '||v_qty);
end loop;
close v_refcursor;
end;
/
```

Q4)

--in this function I have creted a var mycount in which I am going to insert no of customers who have purchased the products given pid and used count() function for it and returned the value.

```
set serveroutput on
declare
mycount numeric(5);
function no_of_customers(
mypid in purchases.pid%type) return number is
num_of_customers number;
begin
select distinct count(*) into num_of_customers from purchases where pid=mypid;
return(num_of_customers);
end;
begin
mycount := no_of_customers('p004');
dbms_output.put_line('Number of customers who have visited at least ' || mycount);
end;
/
```

Q5)

--I have added the customer

```
CREATE OR REPLACE procedure add_customer(c_id IN customers.cid%type, c_name in
customers.name%type, c_telephone# in customers.TELEPHONE#%type)
is
begin
insert into customers values (c_id, c_name, c_telephone#, 1, to_char(SYSDATE, 'DD-MON-YYYY'));
end;
/
```

Q6)

--In this procedure I have added tuple to purchases table furthermore before adding tuple I have accessed Regular_price and QOH to compare whether the given requirement will fulfilled by the previous quantity or not.If it is met then insert and else print the message.

```
set serveroutput on
declare
calculatesaving number(8,2);
v_qoh number(4);
v_refcursor sys_refcursor;
procedure add_purchase(e_id in purchases.EID%type, p_id in purchases.pid%type, c_id in
purchases.cid%type
, pur_qty in purchases.QTY%type, pur_unit_price in purchases.UNIT_PRICE%type,ref_cursor out
sys_refcursor)
is
```

```
begin
open ref_cursor for
select REGULAR_PRICE,QOH from products where pid=p_id;
fetch v_refcursor into calculatesaving,v_qoh;
if(pur_qty>v_qoh) then
dbms_output.put_line('Insufficient quantity in stock.');
else
insert into purchases values (pur#.nextval, e_id, p_id, c_id,  to_char(SYSDATE, 'DD-MON-YYYY'), pur_qty,
pur_unit_price, pur_unit_price*pur_qty
,(pur_unit_price*pur_qty)-calculatesaving);
end if;
end;
begin
loop
add_purchase('e03', 'p008', 'c003',6,211.65,v_refcursor);
fetch v_refcursor into calculatesaving,v_qoh;
exit when v_refcursor%notfound;
dbms_output.put_line(calculatesaving||' '||v_qoh );
end loop;
close v_refcursor;
end;
/
```

Part6 Trigger1)

```
--trigger will be called after insert in purchases
create or replace trigger QOH_update after insert on purchases for each row
declare
new_qoh number(4);
qoh_products number(4);
QOH_THRESHOLD_products number(4);
v_refcursor sys_refcursor;
last_PUR_DATE date;
new_date date;
begin
--accesing QTY from purchases
new_qoh:= :new.QTY;
dbms_output.put_line(new_qoh||' '||:new.pid);
--fetching qoh and QOH_threshold into var
open v_refcursor for
select qoh,QOH_THRESHOLD from products where pid=:new.pid;
fetch v_refcursor into qoh_products,QOH_THRESHOLD_products;
close v_refcursor;
dbms_output.put_line(qoh_products);
```

```
--checking condtion if QOH_threshold exceeds then qoh will be qoh_threshold+10 else will be reduced --
--by given quantity
if((qoh_products-new_qoh)<QOH_THRESHOLD_products) then
dbms_output.put_line('The current qoh of the product is below the required threshold and new supply
is required'||' '||QOH_THRESHOLD_products);
update products set qoh=(QOH_THRESHOLD_products+10) where pid= :new.pid;
new_qoh:=(QOH_THRESHOLD_products+10);
dbms_output.put_line('the new value of the qoh of the product after new supply'||''||new_qoh);
else
update products set qoh=qoh_products-new_qoh where pid= :new.pid;
end if;
last_PUR_DATE:=:old.PUR_DATE;
new_date:=:new.PUR_DATE;
dbms_output.put_line('purchases'||''||last_PUR_DATE||''||new_date);
if(last_PUR_DATE!=new_date) then
dbms_output.put_line('purchases');
update customers set VISITS_MADE=(VISITS_MADE+1),LAST_VISIT_DATE=:old.PUR_DATE where
cid=:new.cid;
end if;
end;
```

Part 6 Trigger2)

--The below trigger will be called before insertion in purchases table as we need to select the last_purchase date. So sfter selecting the last purchases date we are comparing the current date and last purchases date if both are different then we are updating the customers

```
create or replace trigger VISIT_made before insert on purchases for each row
declare
last_PUR_DATE date;
new_date date;
refcursor sys_refcursor;
begin
open refcursor for
select *from(select PUR_DATE from purchases   where cid=:new.cid order by PUR_DATE desc) where
rownum<=1;
fetch refcursor into last_PUR_DATE;
close refcursor;
new_date:= :new.PUR_DATE;
dbms_output.put_line('purchases'||''||last_PUR_DATE||''||new_date);
if(last_PUR_DATE!=new_date) then
dbms_output.put_line('purchases');
update customers set VISITS_MADE=(VISITS_MADE+1),LAST_VISIT_DATE=last_PUR_DATE where
cid=:new.cid;
end if;
```

```
end;
```

Q7)

Part1)

```
create or replace trigger customers_insert after insert on Customers for each row
declare
login_name Varchar(10);
v_refcursor sys_refcursor;
begin
open v_refcursor for
select USERNAME from user_users;
fetch v_refcursor into login_name;
close v_refcursor;
insert into logs values
(log#.nextval, login_name, 'insert',to_char(SYSDATE, 'DD-MON-YYYY'),'Customers', :new.cid);
end;
/
```

Part2)

```
create or replace trigger last_visit_date after update of last_visit_date on Customers for each row
declare
login_name Varchar(10);
v_refcursor sys_refcursor;
begin
open v_refcursor for
select USERNAME from user_users;
fetch v_refcursor into login_name;
close v_refcursor;
insert into logs values
(log#.nextval, login_name, 'update',to_char(SYSDATE, 'DD-MON-YYYY'),'Customers', :new.CID);
end;
/
```

Part3)

```
create or replace trigger visits_made after update of visits_made on Customers for each row
declare
login_name Varchar(10);
v_refcursor sys_refcursor;
begin
open v_refcursor for
select USERNAME from user_users;
fetch v_refcursor into login_name;
close v_refcursor;
```

```
insert into logs values
(log#.nextval, login_name, 'update',to_char(SYSDATE, 'DD-MON-YYYY'),'Customers', :new.CID);
end;
/

Part4)
create or replace trigger purchases_insert after insert on purchases for each row
begin
insert into logs values
(log#.nextval, user, 'insert',to_char(SYSDATE, 'DD-MON-YYYY'),'purchases', :new.pur#);
end;
/

Part5)
create or replace trigger QOH after update of QOH on products for each row
begin
insert into logs values
(log#.nextval, user, 'update',to_char(SYSDATE, 'DD-MON-YYYY'),'Productss', :new.pid);
end;
/
```

JAVA_CODE

```
// usage:  1. compile: javac -cp /usr/lib/oracle/18.3/client64/lib/ojdbc8.jar jdbcdemo2.java
//         2. execute: java -cp /usr/lib/oracle/18.3/client64/lib/ojdbc8.jar jdbcdemo2.java

//Illustrate call stored procedure
import java.sql.*;
import oracle.jdbc.*;
import java.math.*;
import java.io.*;
import java.awt.*;
import oracle.jdbc.pool.OracleDataSource;
import java.util.*;
import java.sql.Types;

public class jdbcdemo2 {

  public static void main (String args []) throws SQLException {
   try
   {
```

```
        //Connection to Oracle server. Need to replace username and
        //password by your username and your password. For security
        //consideration, it's better to read them in from keyboard.
        OracleDataSource ds = new oracle.jdbc.pool.OracleDataSource();
        ds.setURL("jdbc:oracle:thin:@castor.cc.binghamton.edu:1521:ACAD111");
        Connection conn = ds.getConnection("bsehgal1", "9212367502");

//created switch statement for implementing diffrent cases
Scanner sc=new Scanner(System.in);
boolean b=true;
while(b){
System.out.println("enter 1 for displaying employees");
System.out.println("enter 2 for displaying customers");
System.out.println("enter 3 for displaying products");
System.out.println("enter 4 for displaying purchases");
System.out.println("enter 5 for displaying logs");
System.out.println("enter 6 for displaying name name of the customer as well as every purchase the
the customer has made for given cid");
System.out.println("enter 7 for displaying the number of customers who have purchased the product
identified by the pid");
System.out.println("enter 8 for inserting in the customer table");
System.out.println("enter 9 for adding tuple to the Purchases table");
System.out.println("enter 10 to exit");
int n=sc.nextInt();
switch(n){
case 1:
display_employees(conn );
break;
case 2:
display_customers(conn);
break;
case 3:
display_products(conn);
break;
case 4:
display_purchases(conn);
break;
case 5:
display_logs(conn);
break;
case 6:
purchases_made(conn);
break;
case 7:
```

```
number_customers(conn);
break;
case 8:
add_customer(conn);
break;
case 9:
add_purchase(conn);
break;
case 10:
b=false;
break;
default:
System.out.println("please enter right input");
}
}

      // Input sid from keyboard
//       BufferedReader  readKeyBoard;
  //     String         sid;
    //    readKeyBoard = new BufferedReader(new InputStreamReader(System.in));
     //  System.out.print("Please Enter SID:");
      // sid = readKeyBoard.readLine();

       //Prepare to call stored procedure:
        // create or replace procedure show_status (sid_in in students2.sid%type,
        //                  status_out out students2.status%type) is
        // begin select status into status_out from students2 where sid = sid_in;
        // end;
      conn.close();
  }
  catch (SQLException ex) { System.out.println ("\n*** SQLException caught ***\n" + ex.getMessage());}
  catch (Exception e) {System.out.println ("\n*** other Exception caught ***\n");}
 }

static void display_employees(Connection conn) throws SQLException{
try
   {
//calling the show_employees procedure
CallableStatement cs = conn.prepareCall("call show_employees(?)");

       //set the in parameter (the first parameter)
      // cs.setString(1, sid);

       //register the out parameter (the second parameter)
```

```
       cs.registerOutParameter(1, OracleTypes.CURSOR);

       //execute the stored procedure
       cs.execute();

     ResultSet rs = null;
     rs = (ResultSet)cs.getObject(1);
    if(rs != null){

                                System.out.println("\n\neid" + "\t\t" + "name" + "\t" + "\t" + "telephone#" +
"\t\t"  + "\t\t" + "EMAIL" + "\t\t\t");
                                System.out.println("----------------------------------------------------------------------------
------------------------------------------------------------");

}
    while (rs.next()) {
           System.out.println(rs.getString(1) + "\t" + rs.getString(2) + "\t" + rs.getString(3) +rs.getString(4));
           }

       //close the result set, statement, and the connection
       cs.close();

}
catch (SQLException ex) { System.out.println ("\n*** SQLException caught ***\n" + ex.getMessage());}
  catch (Exception e) {System.out.println ("\n*** other Exception caught ***\n");}
 }


static void display_customers(Connection conn) throws SQLException{
try
   {

CallableStatement cs = conn.prepareCall("call show_customers(?)");

       //set the in parameter (the first parameter)
      // cs.setString(1, sid);

       //register the out parameter (the second parameter)
       cs.registerOutParameter(1, OracleTypes.CURSOR);

       //execute the stored procedure
       cs.execute();

     ResultSet rs = null;
```

```
     rs = (ResultSet)cs.getObject(1);
   if(rs != null){

                                System.out.println("\n\ncid" + "\t\t" + "name" + "\t" + "telephone#" + "\t" +
"visit_made" + "\t\t" + "last_visit_date");
                                System.out.println("-----------------------------------------------------------------------------
-------------------------------------------------------------");

}
    while (rs.next()) {
          System.out.println(rs.getString(1) + "\t" + rs.getString(2) + "\t" + rs.getString(3)
+"\t"+rs.getString(4)+"\t"+rs.getString(5));
          }

      //close the result set, statement, and the connection
      cs.close();

}
catch (SQLException ex) { System.out.println ("\n*** SQLException caught ***\n" + ex.getMessage());}
  catch (Exception e) {System.out.println ("\n*** other Exception caught ***\n");}
 }

static void display_products(Connection conn) throws SQLException{
try
   {

CallableStatement cs = conn.prepareCall("call show_products(?)");

      //set the in parameter (the first parameter)
     // cs.setString(1, sid);

      //register the out parameter (the second parameter)
      cs.registerOutParameter(1, OracleTypes.CURSOR);

      //execute the stored procedure
      cs.execute();

    ResultSet rs = null;
    rs = (ResultSet)cs.getObject(1);
   if(rs != null){

                                System.out.println("\n\npid" + "\t\t" + "name" + "\t" + "qoh" + "\t" +
"qoh_threshold" + "\t\t" + "reg_price_rate"+"\t"+"last_visit_date");
```

```
                                System.out.println("-----------------------------------------------------------------------------
-------------------------------------------------------");

}
    while (rs.next()) {
            System.out.println(rs.getString(1) + "\t" + rs.getString(2) + "\t" + rs.getString(3)
+"\t"+rs.getString(4)+"\t"+rs.getString(5)+"\t"+rs.getString(6));
            }

       //close the result set, statement, and the connection
       cs.close();

}
catch (SQLException ex) { System.out.println ("\n*** SQLException caught ***\n" + ex.getMessage());}
  catch (Exception e) {System.out.println ("\n*** other Exception caught ***\n");}
 }

static void display_purchases(Connection conn) throws SQLException{
try
   {

CallableStatement cs = conn.prepareCall("call show_purchases(?)");

       //set the in parameter (the first parameter)
      // cs.setString(1, sid);

       //register the out parameter (the second parameter)
       cs.registerOutParameter(1, OracleTypes.CURSOR);

       //execute the stored procedure
       cs.execute();

     ResultSet rs = null;
     rs = (ResultSet)cs.getObject(1);
   if(rs != null){

                                System.out.println("\n\npur#" + "\t\t" + "eid" + "\t" + "pid" + "\t" + "cid" + "\t\t"
+ "pur_date"+"\t"+"qty"+"\t"+"unit_price"+"\t"+"total"+"\t"+"saving");
                                System.out.println("-----------------------------------------------------------------------------
-------------------------------------------------------");

}
    while (rs.next()) {
```

```
            System.out.println(rs.getString(1) + "\t" + rs.getString(2) + "\t" + rs.getString(3)
+"\t"+rs.getString(4)+"\t"+rs.getString(5)+"\t"+rs.getString(6)+"\t"+rs.getString(7)+"\t"+rs.getString(8)+"
\t"+rs.getString(9));
            }

       //close the result set, statement, and the connection
       cs.close();

}
catch (SQLException ex) { System.out.println ("\n*** SQLException caught ***\n" + ex.getMessage());}
  catch (Exception e) {System.out.println ("\n*** other Exception caught ***\n");}
 }

static void display_logs(Connection conn) throws SQLException{
try
   {

CallableStatement cs = conn.prepareCall("call show_logs(?)");

       //set the in parameter (the first parameter)
      // cs.setString(1, sid);

       //register the out parameter (the second parameter)
       cs.registerOutParameter(1, OracleTypes.CURSOR);

       //execute the stored procedure
       cs.execute();

     ResultSet rs = null;
     rs = (ResultSet)cs.getObject(1);
   if(rs != null){

                                System.out.println("\n\nlog#" + "\t\t" + "user_name" + "\t" + "operation" + "\t"
+ "op_time" + "\t\t" + "table_name"+"\t"+"tuple_pkey");
                                System.out.println("----------------------------------------------------------------------------
------------------------------------------------------------");

}
    while (rs.next()) {
            System.out.println(rs.getString(1) + "\t" + rs.getString(2) + "\t" + rs.getString(3)
+"\t"+rs.getString(4)+"\t"+rs.getString(5)+"\t"+rs.getString(6));
            }
```

```
      //close the result set, statement, and the connection
      cs.close();

}
catch (SQLException ex) { System.out.println ("\n*** SQLException caught ***\n" + ex.getMessage());}
  catch (Exception e) {System.out.println ("\n*** other Exception caught ***\n");}
 }

static void purchases_made(Connection conn) throws SQLException{
try
   {

CallableStatement cs = conn.prepareCall("call purchases_made(?,?)");
System.out.println("enter the cid of customers");
Scanner sc=new Scanner(System.in);
String cid=sc.next();
      //set the in parameter (the first parameter)
       cs.setString(2,cid);

      //register the out parameter (the second parameter)
      cs.registerOutParameter(1, OracleTypes.CURSOR);

      //execute the stored procedure
      cs.execute();

    ResultSet rs = null;
    rs = (ResultSet)cs.getObject(1);
  if(rs != null){

                              System.out.println("\n\n name" + "\t\t" + "pid" + "\t" + "pur_date" + "\t" +
"Qty" + "\t\t" + "unit_price"+"\t"+"total");
                              System.out.println("----------------------------------------------------------------------------------
----------------------------------------------------------");

}
while (rs.next()) {
          System.out.println(rs.getString(1) + "\t" + rs.getString(2) + "\t" + rs.getString(3)
+"\t"+rs.getString(4)+"\t"+rs.getString(5)+"\t"+rs.getString(6));
          }

      //close the result set, statement, and the connection
      cs.close();

}
```

```
catch (SQLException ex) { System.out.println ("\n*** SQLException caught ***\n" + ex.getMessage());}
  catch (Exception e) {System.out.println ("\n*** other Exception caught ***\n");}
 }

static void  number_customers(Connection conn) throws SQLException{
try
   {

CallableStatement cs = conn.prepareCall("call number_customers(?,?)");
System.out.println("enter the pid of customers");
Scanner sc=new Scanner(System.in);
String pid=sc.next();
      //set the in parameter (the first parameter)
       cs.setString(2,pid);
//register ouut parameter
cs.registerOutParameter(1, OracleTypes.CURSOR);
      //execute the stored procedure
      cs.execute();

    ResultSet rs = null;
    rs = (ResultSet)cs.getObject(1);
   if(rs != null){

                                System.out.println("\n\npurchaseno");
                                System.out.println("----------------------------------------------------------------------------
------------------------------------------------------------");

}
while (rs.next()) {
           System.out.println(rs.getString(1));
           }

      //close the result set, statement, and the connection
      cs.close();

}
catch (SQLException ex) { System.out.println ("\n*** SQLException caught ***\n" + ex.getMessage());}
  catch (Exception e) {System.out.println ("\n*** other Exception caught ***\n");}
 }

static void  add_customer(Connection conn) throws SQLException{
try
   {
```

```
CallableStatement cs = conn.prepareCall("call add_customer(?,?,?)");
System.out.println("enter the cid of customers");
Scanner sc=new Scanner(System.in);
String pid=sc.next();
      //set the in parameter (the first parameter)
      cs.setString(1,pid);
System.out.println("enter the customername of customers");
String name=sc.next();
cs.setString(2,name);
System.out.println("enter the customername of customers");
String telephoneno=sc.next();
cs.setString(3,telephoneno);

      //execute the stored procedure
      cs.executeUpdate();

   //  ResultSet rs = null;
    // rs = (ResultSet)cs.getObject(1);
//             if(rs != null){

//                              System.out.println("\n\npurchaseno");
  //                            System.out.println("-----------------------------------------------------------------------------------
-------------------------------------------------------");
//
//         }
//         while (rs.next()) {
        //             System.out.println(rs.getString(1));
           //}

      //close the result set, statement, and the connection
      cs.close();

}
catch (SQLException ex) { System.out.println ("\n*** SQLException caught ***\n" + ex.getMessage());}
   catch (Exception e) {System.out.println ("\n*** other Exception caught ***\n");}
  }

static void  add_purchase(Connection conn) throws SQLException{
try
   {

CallableStatement cs = conn.prepareCall("call add_purchase(?,?,?,?,?)");
System.out.println("enter the eid:");
Scanner sc=new Scanner(System.in);
```

```
String e_id=sc.next();
	//set the in parameter (the first parameter)
	cs.setString(1,e_id);
System.out.println("enter the pid:");
String p_id=sc.next();
cs.setString(2,p_id);
System.out.println("enter the cid:");
String c_id=sc.next();
cs.setString(3,c_id);
System.out.println("enter the pur_qty:");
int pur_qty=sc.nextInt();
cs.setInt(4,pur_qty);
System.out.println("enter the unit_price:");
float unit_price=sc.nextFloat();
cs.setFloat(5,unit_price );

	//execute the stored procedure
	cs.executeUpdate();
String query="SELECT qoh,pid,qoh_threshold FROM products";
//PreparedStatement ps = conn.prepareStatement(query);
//ps.setString(1,p_id);
Statement stmt = conn.createStatement ();
	ResultSet rset  = null;
	rset = stmt.executeQuery (query);
int my_qoh=0,my_qoh_threshold=0;
	//rs = (ResultSet)cs.getObject(1);
//		if(rs != null){

//				System.out.println("\n\npurchaseno");
 //				System.out.println("-----------------------------------------------------------------------------
-------------------------------------------------------");
//
//	}
//getting out the whole data from products table then comparing with p_id.
while (rset.next()) {
String s3=rset.getString(2);
if(p_id.equals(s3)){
	my_qoh=Integer.parseInt(rset.getString(1));
	my_qoh_threshold=Integer.parseInt(rset.getString(3));
}
}
//comparing threshold with qantity required and quantityin hand.
if((my_qoh-pur_qty)<my_qoh_threshold){
```

```
System.out.println("The current qoh of the product is below the required threshold and new supply is
required");
System.out.println("The new qoh of product is"+(my_qoh_threshold+10));
}
      //close the result set, statement, and the connection
rset.close();
      stmt.close();
      cs.close();

}
catch (SQLException ex) { System.out.println ("\n*** SQLException caught ***\n" + ex.getMessage());}
  catch (Exception e) {System.out.println ("\n*** other Exception caught ***\n");}
 }

}
```

“I have done this assignment completely on my own. I have not copied it, nor have I given my solution to anyone else. I understand that if I am involved in plagiarism or cheating I will have to sign an official form that I have cheated and that this form will be stored in my official university record. I also understand that I will receive a grade of 0 for the involved assignment and my grade will be reduced by one level (e.g., from A to A- or from B+ to B) for my first offense, and that I will receive a grade of “F” for the course for any additional offense of any kind.”
Bhawesh