**Group: 19**

**Topic: Pizza Store Interface**

Group Members:

* Naman Dhanotia (19ucs128)
* Rishabh Sahu (19ucs129)
* Manit Mehta (19ucs147)
* Aditya Khandelwal (19ucs163)

**Overview:**

This project aims to create a platform for a pizza store wherein you can enter, view, or update the details of a customer ordering pizza or employees working in the store.

We have mainly used MySQL as DBMS and Java as programming language. The java file connects to the mysql database by specifying parameters like sql server url ,and password from separate connector.properties file (which we have included in the submission zip folder for reference).

**MySQL Database Script:**

create database pizza;

use pizza;

create table Products(PID int primary key auto\_increment, Name varchar(30), Cost float, Rating int );

create table Customers(CID int primary key auto\_increment, Address varchar (30), Name varchar(30), Phone\_number bigint);

create table Employees(EID int primary key auto\_increment,Name varchar(30),Phone\_number bigint);

create table orders(OrderID int primary key auto\_increment,Date\_of\_order date, Time\_of\_order time, CID int, PID int, Cost float,foreign key (CID) REFERENCES CUSTOMERS(CID),foreign key (PID) references products(PID));

insert into Products(Name, Cost, Rating)

values ('Cheese Pizza', 124.75, 9),

('Veggie Pizza', 154.25, 7),

('Pepperoni Pizza', 300.00, 9),

('Margherita Pizza', 145.75, 6),

('Cheese Veggie Pizza', 199.95, 8),

('Onion Tomato Corn Pizza', 250.50, 8),

('Hawaiian Pizza', 225.00, 7),

('Italy Special Pizza', 175.75, 10),

('New-York Style Pizza', 99.99, 8),

('Cicilian Pizza', 100.00, 9);

insert into Customers(Address, Name, Phone\_number)

values ('6649 Blue street', 'Naman', 9989989989),

('8W Bridge', 'Aman', 8898989898),

('25E near church', 'Man', 7869785869),

('New Horizon' ,'Ana', 8324543212),

('Dangi Street' , 'Namish', 8342567896),

('Sabji Bazar', 'Anish', 8123456789),

('GokulDham Society', 'Satish', 8876567543),

('Powder Gully' , 'Sujal', 8232435465),

('Andheri East', 'Sanyam', 9876567883),

('Andheri West', 'Gautam', 7865676783);

insert into Employees(Name, Phone\_Number)

values ('Tipendra Gada', 9595959595),

('Jethalal Sharma', 9534959595),

('Champaklal Chouhan', 8895959595),

('Jayantilal Washington', 9565959595),

('Sonu Bhide', 9595959533),

('Madhavi Verma', 9595944595),

('Roshan Chandan', 9595669595),

('Rita Lal', 9595959225),

('Abdul Khan', 9595911595),

('Salman Firoz', 9598859595);

insert into orders(Date\_of\_order, Time\_of\_order, CID, PID, Cost)

values ('2020-08-10', '13:23:44', 5, 3, 543.65),

('2020-08-11', '13:23:44', 10, 4, 55),

('2020-08-12', '14:23:44', 4, 5, 365),

('2020-08-13', '15:23:44', 3, 6, 435),

('2020-08-14', '16:23:44', 2, 7, 565),

('2020-08-15', '17:23:44', 1, 8, 65),

('2020-08-16', '18:23:44', 9, 9, 554.65),

('2020-08-17', '19:23:44', 8, 10, 673.65),

('2020-08-18', '14:23:44', 7, 1, 444.65),

('2020-08-19', '16:23:44', 6, 2, 53.65);

#drop database pizza;

#select\* from orders;

#update orders set PID=1 where OrderID=5;

**Pizza.java Source Code**

import java.sql.\*;

import java.sql.Date;

import java.util.\*;

import java.io.FileInputStream;

import java.io.IOException;

//Table Schema creation by Naman Dhanotia 19ucs128

//table entry part and sql connection part by Rishabh Sahu 19ucs129

//table viewing part by Aditya Khandelwal 19ucs163

//table update part by Manit Mehta 19ucs147

class SQL\_connector

{

public static Connection getConnection()throws SQLException{

//Connection parameters to the sql server like password and url are specified in the connector.properties file from where this class works

Connection connect=null;

try (FileInputStream f = new FileInputStream("connector.properties"))

{

// load the connector.properties file see this webpage for more info https://www.mysqltutorial.org/connecting-to-mysql-using-jdbc-driver/

Properties fetch\_database = new Properties();

fetch\_database.load(f);

// assign database connection parameters from connector.properties file containing the password to sql server

String link = fetch\_database.getProperty("url");

String user = fetch\_database.getProperty("user");

String password = fetch\_database.getProperty("password");

// creating a connection to the database

connect = DriverManager.getConnection(link, user, password);

}

catch (IOException e)

{

System.out.println(e.getMessage());

}

return connect;

}

}

// This program allows user to manage the tables and their data for a pizza store database which is already created in sql local host server

class Pizza\_store

{

static void Enter\_values()

{

Scanner sc1 = new Scanner(System.in);

//If ENTER option is selected ask whether to enter data in orders table ,Products table ,Customers table ,Employees table

System.out.println("Please specify the table in which you want to enter the data \n Type\n\t 1 for orders \n\t2 for products \n\t3 for Customers\n\t4 for Employees");

int enter\_option = sc1.nextInt();

switch (enter\_option)

{

//orders

case (1):

{

ResultSet rs1 = null;

int OrderID\_point = 0;

String entry = "INSERT INTO orders(Date\_of\_order,Time\_of\_order,CID,PID,Cost) " + "VALUES(?,?,?,?,?)";

String str;

String str2;

Date Order\_date;//date object

Time Order\_time;//time object

int Order\_Cid;

int Order\_Pid;

float Order\_cost;

try (Connection GATE = SQL\_connector.getConnection(); PreparedStatement order\_record = GATE.prepareStatement(entry, Statement.RETURN\_GENERATED\_KEYS);) {

//order id is autoincrement in database schema of orders table so no need to enter manually

System.out.println("Enter the values of Order date in YYYY-MM-DD format,Order time in 24 Hour HH-MM-SS format,Customer Id,Product Id,Cost ,respectively\n");

System.out.println("Enter Order date in YYYY-MM-DD format :\n");

Scanner sc = new Scanner(System.in);

str = sc.nextLine();

Order\_date = Date.valueOf(str);

order\_record.setDate(1, Order\_date);//convert to date object

System.out.println("\nEnter time in HH:MM:SS format :\t");

Scanner sc2 = new Scanner(System.in);

str2 = sc2.nextLine();

Order\_time = Time.valueOf(str2);//convert to time object

order\_record.setTime(2, Order\_time);

System.out.println("\nEnter the customer ID:\t");

Order\_Cid = sc.nextInt();

//Will soon add the condition to check if this customerID exists already or not , and if not then we will add a new customer record also

order\_record.setInt(3, Order\_Cid);

System.out.println("\nEnter the Product ID:\t");

Order\_Pid = sc.nextInt();

order\_record.setInt(4, Order\_Pid);

//Need to choose whether to calculate cost from product automatically using sql query , or manually enter

System.out.println("\nEnter the value of Order\_cost");

Order\_cost = sc.nextFloat();

order\_record.setFloat(5, Order\_cost);

int insert\_result;

insert\_result = order\_record.executeUpdate();//add this record to the sql database

if (insert\_result == 1) {

// get last entered order id ,

rs1 = order\_record.getGeneratedKeys();

if (rs1.next())

OrderID\_point = rs1.getInt(1);//return the first attribute of currently entered result set which is the auto generated order id

}

} catch (SQLException ex) {

System.out.println(ex.getMessage());

} finally {

try {

if (rs1 != null)

rs1.close();

} catch (SQLException e) {

System.out.println(e.getMessage());

}

}

}

break;

//products

case (2):

{

ResultSet rs2 = null;

int ProductID\_point = 0;

String entry = "INSERT INTO products(Name,Cost,Rating) " + "VALUES(?,?,?)";//product id generates automatically

String str;

float Order\_cost;

try (Connection GATE = SQL\_connector.getConnection(); PreparedStatement product\_record = GATE.prepareStatement(entry, Statement.RETURN\_GENERATED\_KEYS);) {

//order id is autoincrement in database schema of orders table so no need to enter manually

System.out.println("Enter the Product Name for this pizza\n");

Scanner sc = new Scanner(System.in);

str = sc.nextLine();

product\_record.setString(1, str);

int Product\_rating;

System.out.println("\nEnter the Rating between 1 to 10 both inclusive ID:\t");

Product\_rating = sc.nextInt();

//Will soon add the condition to check if this customerID exists already or not , and if not then we will add a new customer record also

product\_record.setInt(2, Product\_rating);

float Product\_cost;

System.out.println("\nEnter the Product Cost:\t");

Product\_cost= sc.nextFloat();

product\_record.setFloat(3, Product\_cost);

int insert\_result;

insert\_result = product\_record.executeUpdate();//add this record to the sql database

if (insert\_result == 1) {

// get last entered order id ,

rs2 = product\_record.getGeneratedKeys();

if (rs2.next())

ProductID\_point = rs2.getInt(1);//return the first attribute of currently entered result set which is the auto generated order id

}

} catch (SQLException ex) {

System.out.println(ex.getMessage());

} finally {

try {

if (rs2 != null)

rs2.close();

} catch (SQLException e) {

System.out.println(e.getMessage());

}

}

}

break;

//Customers

case (3): {

ResultSet rs3 = null;

int CustomerID\_point = 0;

String entry = "INSERT INTO customers(Address,Name,Phone\_number) " + "VALUES(?,?,?)";//Customer id generates automatically

String str;

String str2;

float Order\_cost;

try (Connection GATE = SQL\_connector.getConnection(); PreparedStatement customer\_record = GATE.prepareStatement(entry, Statement.RETURN\_GENERATED\_KEYS);) {

//order id is autoincrement in database schema of orders table so no need to enter manually

System.out.println("Enter the Customer Name \n");

Scanner sc = new Scanner(System.in);

str = sc.nextLine();

customer\_record.setString(2, str);

System.out.println("Enter the Customer's Address \n");

str2 = sc.nextLine();

customer\_record.setString(1, str2);

int Phone\_number;

System.out.println("\nEnter the Phone number of the customer:\t");

Phone\_number = sc.nextInt();

//Will soon add the condition to check if this customerID exists already or not , and if not then we will add a new customer record also

customer\_record.setInt(3,Phone\_number );

int insert\_result;

insert\_result = customer\_record.executeUpdate();//add this record to the sql database

if (insert\_result == 1) {

// get last entered order id ,

rs3 = customer\_record.getGeneratedKeys();

if (rs3.next())

CustomerID\_point = rs3.getInt(1);//return the first attribute of currently entered result set which is the auto generated order id

}

} catch (SQLException ex) {

System.out.println(ex.getMessage());

} finally {

try {

if (rs3 != null)

rs3.close();

} catch (SQLException e) {

System.out.println(e.getMessage());

}

}

}

break;

//Employees

case (4): {

ResultSet rs4 = null;

int EmployeeID\_point = 0;

String entry = "INSERT INTO Employees(Name,Phone\_number) " + "VALUES(?,?)";//Employee id generates automatically

String str;

String str2;

float Order\_cost;

try (Connection GATE = SQL\_connector.getConnection(); PreparedStatement Employee\_record = GATE.prepareStatement(entry, Statement.RETURN\_GENERATED\_KEYS);) {

//order id is autoincrement in database schema of orders table so no need to enter manually

System.out.println("Enter the Employee Name \n");

Scanner sc = new Scanner(System.in);

str = sc.nextLine();

Employee\_record.setString(1, str);

int Phone\_number;

System.out.println("\nEnter the Phone number of the Employee:\t");

Phone\_number = sc.nextInt();

//Will soon add the condition to check if this EmployeeID exists already or not , and if not then we will add a new Employee record also

Employee\_record.setInt(2,Phone\_number );

int insert\_result;

insert\_result = Employee\_record.executeUpdate();//add this record to the sql database

if (insert\_result == 1) {

// get last entered order id ,

rs4 = Employee\_record.getGeneratedKeys();

if (rs4.next())

EmployeeID\_point = rs4.getInt(1);//return the first attribute of currently entered result set which is the auto generated order id

}

} catch (SQLException ex) {

System.out.println(ex.getMessage());

} finally {

try {

if (rs4 != null)

rs4.close();

} catch (SQLException e) {

System.out.println(e.getMessage());

}

}

}

break;

default:

System.out.println("Didn't match with any of the existing option ,Please try again\n ");

}

}

public static void View\_data() {

System.out.println("Please specify the table whose data you want to view-(1)orders,(2)products (3)Customers ,(4)Employees");

Scanner sc = new Scanner(System.in);

int choice = sc.nextInt();

switch (choice)

{

//orders

case (1):

{

int data\_option = 0;

System.out.println("Enter how you want to see data by want the entire data(press 1) or do you want to search(press 2) with specific criteria");

data\_option = sc.nextInt();

if (data\_option == 1) //entire data of orders

{

try (Connection GATE = SQL\_connector.getConnection(); Statement stmt = GATE.createStatement(); ResultSet rs = stmt.executeQuery("SELECT \*" + " from orders")) {

System.out.println("Contents of the table are: ");

int i=1;

while (rs.next()) {

System.out.println("Order number +"+i+"\n");

System.out.println("\t Date of ordering"+rs.getString("Date\_Of\_Order"));

System.out.println("\t Time of ordering"+rs.getString("Time\_Of\_Order"));

System.out.println("\t Cost of the order"+rs.getString("Cost"));

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

i++;

}

} catch (SQLException ex) {

System.out.println(ex.getMessage());

}

} else //searching data from user specification

{

System.out.println("You can search by date");

try (Connection GATE = SQL\_connector.getConnection(); Statement smt = GATE.createStatement()) {

System.out.print("Enter date: in YYYY-MM-DD pattern");

String date = sc.nextLine();

String q = "Select \* from orders where Date\_Of\_Order='" + date + "'";

ResultSet rs = smt.executeQuery(q);

//to print the resultset on console

if (rs.next()) {

do {

System.out.println("OrderID:"+rs.getString(1)+"\nOrder date:" + rs.getString(2) + "\nOrder time:" + rs.getString(3) + "\nCustomer ID who ordered:" + rs.getString(4) + "\nProduct Id of pizza which was ordered" + rs.getString(5) + "\nCost of the order:" + rs.getString(4)+"\n");

}

while (rs.next());

} else {

System.out.println("Record Not Found...");

}

} catch (Exception e) {

System.out.println(e);

}

}

}

break;

//Products

case (2): {

try (Connection GATE = SQL\_connector.getConnection(); Statement stmt = GATE.createStatement();) {

//Creating a Statement

int entire\_option = 0;

System.out.println("Enter how you want to see data by-By searching(type 2) or do want to see the entire data(type 1) then press enter");

entire\_option = sc.nextInt();

if (entire\_option == 1) {

//Retrieving the data

ResultSet rs = stmt.executeQuery("SELECT \*" + " from products");

int i=1;

System.out.println("Contents of the table are: ");

while (rs.next()) {

System.out.println("Product number :"+i+"\n");

System.out.println("\t Pizza name: "+rs.getString("Name"));

System.out.println("\t Pizza MRP cost: "+rs.getString("Cost"));

System.out.println("\t Pizza Rating: "+rs.getString("Rating"));

i++;

}

}

else

{

int option = 0;

System.out.println("You can search by id(type 1) or by using Name(type 2) then press enter");

option = sc.nextInt();

if(option == 1)

{

//creating object of Scanner

Scanner sc2 = new Scanner(System.in);

System.out.print("Enter id: ");

int id = sc2.nextInt();

String q = "Select \* from Products where PID= " + id;

ResultSet rs = stmt.executeQuery(q);

//to print the resultset on console

if (rs.next())

{

do

{

System.out.println("Product Id: "+rs.getString(1)+"\n Pizza Name " + ":" + rs.getString(2)+"\n Cost:" + "," + rs.getString(3) + "\nRating:" + rs.getString(4));

}

while (rs.next());

}

else

{

System.out.println("Record Not Found...");

}

}

else

{

//creating object of Scanner

Scanner sc2 = new Scanner(System.in);

System.out.print("Enter name: ");

String name = sc2.nextLine();

String q = "Select \* from Products where Name = " + name;

ResultSet rs = stmt.executeQuery(q);

//to print the resultset on console

if (rs.next())

{

do

{

System.out.println("Product Id: "+rs.getString(1)+"\n Pizza Name " + ":" + rs.getString(2)+"\n Cost:" + "," + rs.getString(3) + "\nRating:" + rs.getString(4));

}

while (rs.next());

}

else

{

System.out.println("Record Not Found...");

}

}

}

}

catch (SQLException e)

{

System.out.println(e);

}

}

break;

//Customers

case (3):

{

int entire\_choice = 0;

System.out.println("Do u want to display entire data (type 1 then press enter) OR search particular record(type 2 then press enter)");

entire\_choice = sc.nextInt();

if (entire\_choice == 1) {

try (Connection GATE = SQL\_connector.getConnection(); Statement stmt = GATE.createStatement())

{

//Retrieving the data

ResultSet rs = stmt.executeQuery("SELECT \*" + " from customers");

System.out.println("Contents of the table are: ");

while (rs.next()) {

System.out.println(rs.getString("Name"));

System.out.println(rs.getString("Phone\_Number"));

System.out.println(rs.getString("Address"));

}

}

catch (SQLException e) {

System.out.println(e);

}

}

else

{

int option = 0;

System.out.println("You can search by id(press 1) or by using Name(press 2)");

option = sc.nextInt();

if (option == 1)

{

try (Connection GATE = SQL\_connector.getConnection(); Statement smt = GATE.createStatement()) {

//creating object of Scanner

Scanner sc2 = new Scanner(System.in);

System.out.print("Enter id: \n");

int id = sc2.nextInt();

String q = "Select \* from Customers where id = " + Integer.toString(id);

ResultSet rs2 = smt.executeQuery(q);

//to print the resultset on console

if (rs2.next()) {

do {

System.out.println(rs2.getString(1) + "," + rs2.getString(2) + "," + rs2.getString(3) + "," + rs2.getString(4));

}

while (rs2.next());

} else {

System.out.println("Record Not Found...");

}

} catch (Exception e) {

System.out.println(e);

}

}

else

{

try (Connection GATE = SQL\_connector.getConnection(); Statement smt = GATE.createStatement())

{

//creating object of Scanner

Scanner sc2 = new Scanner(System.in);

System.out.print("Enter name: ");

String name = sc2.nextLine();

String q = "Select \* from Customers where Name =" + name;

ResultSet rs3 = smt.executeQuery(q);

//to print the resultset on console

if (rs3.next()) {

do {

System.out.println(rs3.getString(1) + "," + rs3.getString(2) + "," + rs3.getString(3) + "," + rs3.getString(4));

}

while (rs3.next());

} else {

System.out.println("Record Not Found...");

}

}

catch (Exception e)

{

System.out.println(e);

}

}

}

}

break;

//Employees table query

case (4):

{

try (Connection GATE = SQL\_connector.getConnection(); Statement stmt = GATE.createStatement()) {

int option = 0;

System.out.println("You can search by id or by using Name(press 1) or display entire info(press 0)");

option = sc.nextInt();

if (option == 0) {

//Retrieving the data

ResultSet rs = stmt.executeQuery("SELECT Name,Phone\_Number" + " from Employees");

System.out.println("Contents of the table are: ");

while (rs.next()) {

System.out.println(rs.getString("Name"));

System.out.println(rs.getString("Phone\_Number"));

}

}

else

{

int sub\_option = 0;

System.out.println("You can search by id(press 1) or by using Name(press 2)");

sub\_option = sc.nextInt();

if (option == 1)//search by id

{

Scanner sc2 = new Scanner(System.in);

System.out.print("Enter id: ");

int id = sc2.nextInt();

String q = "Select \* from Employees where id = " + Integer.toString(id);

ResultSet rs = stmt.executeQuery(q);

//to print the resultset on console

if (rs.next()) {

do {

System.out.println(rs.getString(1) + "," + rs.getString(2) + "," + rs.getString(3) + "," + rs.getString(4));

}

while (rs.next());

}

else {

System.out.println("Record Not Found...");

}

}

else //search by name

{

Scanner sc2 = new Scanner(System.in);

System.out.print("Enter name: ");

String name = sc2.nextLine();

String q = "Select \* from Employees where Name = " + name;

ResultSet rs = stmt.executeQuery(q);

//to print the resultset on console

if (rs.next()) {

do {

System.out.println(rs.getString(1) + "," + rs.getString(2) + "," + rs.getString(3) + "," + rs.getString(4));

}

while (rs.next());

} else {

System.out.println("Record Not Found...");

}

}

}

}

catch (SQLException ex)

{

System.out.println(ex);

}

}

break;

default:

{

System.out.println("You had to choose among (1-4)!! ");

}

}

}

public static void update\_data() {

System.out.println("Please specify the table whose data you want to update\n(Type 1 then press enter for)ORDERS\n(Type 2 then press enter for)PRODUCTS\n(Type 3 then press enter for)CUSTOMERS\n(Type 4 then press enter for)EMPLOYEES\n");

Scanner sc = new Scanner(System.in);

int table\_select = sc.nextInt();

switch (table\_select) {

//update orders table

case (1): {

String sqlUpdate = "UPDATE orders "

+ " SET PID = ? "

+ " where OrderID = ? ";

try (Connection GATE = SQL\_connector.getConnection(); PreparedStatement pstmt = GATE.prepareStatement(sqlUpdate);) {

System.out.print("enter OrderID whose record is to be updated");

int OrderID = sc.nextInt();

System.out.print("enter productID to be updated in this record");

int newproduct = sc.nextInt();

int PID = newproduct;

pstmt.setInt(1, PID);

pstmt.setInt(2, OrderID);

int rowAffected = pstmt.executeUpdate();

} catch (SQLException ex) {

System.out.println(ex.getMessage());

}

}

break;

//Update products table

case (2): {

System.out.print("enter ProductID whose record is to be updated");

int Product\_ID = sc.nextInt();

System.out.print("Please select: (a)update product name,(b)update product price");

char u = sc.next().charAt(0);

switch (u) {

case ('a'): {

String sqlUpdate = " UPDATE products "

+ " SET NAME = ? "

+ " where ProductID= ? ";

try (Connection Gate2 = SQL\_connector.getConnection();PreparedStatement pstmt = Gate2.prepareStatement(sqlUpdate);) {

System.out.print("enter new product name");

String newname = sc.nextLine();

String NAME = newname;

pstmt.setString(1, NAME);

pstmt.setInt(2, Product\_ID);

int rowAffected = pstmt.executeUpdate();

} catch (SQLException ex) {

System.out.println(ex.getMessage());

}

}

break;

case ('b'): {

String sqlUpdate = "UPDATE products "

+ " SET Cost = ? "

+ " where PID = ? ";

try (Connection Gate = SQL\_connector.getConnection();PreparedStatement pstmt = Gate.prepareStatement(sqlUpdate);) {

System.out.print("enter new product cost");

float newcost = sc.nextFloat();

float cost = newcost;

pstmt.setFloat(1, cost);

pstmt.setInt(2, Product\_ID);

int rowAffected = pstmt.executeUpdate();

} catch (SQLException ex) {

System.out.println(ex.getMessage());

}

}

break;

default: {

System.out.print("enter legal value");

}

}

}

break;

//Customers table update

case (3): {

System.out.print("Please select: (a)update customer adress,(b)update customer phone number");

char parameter\_select = sc.next().charAt(0);

switch (parameter\_select) {

case ('a'): {

String sqlUpdate = "UPDATE Customers "

+ " SET Address = ? "

+ " where CID = ? ";

try (Connection Gate = SQL\_connector.getConnection(); PreparedStatement pstmt = Gate.prepareStatement(sqlUpdate);) {

System.out.print("enter new address");

String newaddress = sc.nextLine();

String address = newaddress;

int CustomerID = 2;

pstmt.setString(1, address);

pstmt.setInt(2, CustomerID);

int rowAffected = pstmt.executeUpdate();

} catch (SQLException ex) {

System.out.println(ex.getMessage());

}

}

break;

case ('b'): {

String sqlUpdate = "UPDATE products "

+ " SET Phone\_NO = ? "

+ " where CID = ? ";

try (Connection Gate = SQL\_connector.getConnection(); PreparedStatement pstmt = Gate.prepareStatement(sqlUpdate);) {

System.out.print("enter new phone number");

int newnumber = sc.nextInt();

int Phone\_NO = newnumber;

int CustomerID = 2;

pstmt.setInt(1, Phone\_NO);

pstmt.setInt(2, CustomerID);

int rowAffected = pstmt.executeUpdate();

} catch (SQLException ex) {

System.out.println(ex.getMessage());

}

}

break;

default: {

System.out.print("Value entered does not match given list of options");

}

}

}

break;

//Employees table update

case (4):

{

System.out.print("Please select: (a)update employee address,(b)update employee phone number");

char parameter\_select = sc.next().charAt(0);

switch (parameter\_select)

{

case ('a'):

{

String sqlUpdate = "UPDATE Employees "

+ " SET Address = ? "

+ " where EID = ? ";

try (Connection Gate = SQL\_connector.getConnection();PreparedStatement pstmt = Gate.prepareStatement(sqlUpdate))

{

System.out.print("enter new address");

String newaddress = sc.nextLine();

String address = newaddress;

int Employee\_ID = 2;

pstmt.setString(1, address);

pstmt.setInt(2, Employee\_ID);

int rowAffected = pstmt.executeUpdate();

}

catch (SQLException ex)

{

System.out.println(ex.getMessage());

}

}

break;

case ('b'):

{

String sqlUpdate = "UPDATE products "

+ " SET Phone\_NO= ? "

+ " where EID=?;";

try (Connection Gate = SQL\_connector.getConnection();PreparedStatement pstmt = Gate.prepareStatement(sqlUpdate))

{

System.out.print("enter new phone number");

int newnumber = sc.nextInt();

int Phone\_NO = newnumber;

int EmployeeID = 2;

pstmt.setInt(1, Phone\_NO);

pstmt.setInt(2, EmployeeID);

int rowAffected = pstmt.executeUpdate();

}

catch (SQLException ex)

{

System.out.println(ex.getMessage());

}

}

break;

default: {

System.out.print("enter legal value");

}

}

}

break;

default: {

throw new java.lang.IllegalStateException("Unexpected value: " + table\_select);

}

}

}

//}

public static void main(String[] Args)

{

//Part to test sql connection to database,before user interaction

try (Connection check\_connection = SQL\_connector.getConnection())

{

// print out a message

System.out.println(String.format("Connected to database %s "

+ "successfully.", check\_connection.getCatalog()));

}

catch (SQLException ex)

{

System.out.println(ex.getMessage());

/\*if connection is unsuccessful then we display error message and system administrator can check for problems

encountered while accessing the database\*/

return;

}

// If connection is successful then user interaction begins

int option=5;

while(option!=6) {

System.out.println("Type 1 for Entering values in existing tables OR\nType 2 for Viewing data in existing tables OR\nType 3 for Updating existing data present in tables\n Type 6 to exit program");

Scanner sk = new Scanner(System.in);

option = sk.nextInt();

switch (option) {

// Take new values from user

case (1):

Enter\_values();

break;

//view existing data in tables on user specified parameters

case (2):

View\_data();

break;

// update existing data in tables

case (3):

update\_data();

break;

case(6):{

System.out.println("Exiting program\n");

return;

}

default: {

System.out.println("You had to choose from among 1 ,2 ,3 or 6 Since you have chosen option as " + option + " the program will now terminate");

return;

}

}

}

}

}

**Project Input and their Output :**

Connected to database pizza successfully.

Type 1 for Entering values in existing tables OR

Type 2 for Viewing data in existing tables OR

Type 3 for Updating existing data present in tables

Type 6 to exit program

1

Please specify the table in which you want to enter the data

Type

1 for orders

2 for products

3 for Customers

4 for Employees

2

Enter the Product Name for this pizza

Sharma's special

Enter the Rating between 1 to 10 both inclusive ID:

9

Enter the Product Cost:

260

Type 1 for Entering values in existing tables OR

Type 2 for Viewing data in existing tables OR

Type 3 for Updating existing data present in tables

Type 6 to exit program

2

Please specify the table whose data you want to view-(1)orders,(2)products (3)Customers ,(4)Employees

2

Enter how you want to see data by-By searching(type 2) or do want to see the entire data(type 1) then press enter

1

Contents of the table are:

Product number :1

Pizza name: Cheese Pizza

Pizza MRP cost: 124.75

Pizza Rating: 9

Product number :2

Pizza name: Veggie Pizza

Pizza MRP cost: 154.25

Pizza Rating: 7

Product number :3

Pizza name: Pepperoni Pizza

Pizza MRP cost: 300.0

Pizza Rating: 9

Product number :4

Pizza name: Margherita Pizza

Pizza MRP cost: 145.75

Pizza Rating: 6

Product number :5

Pizza name: Cheese Veggie Pizza

Pizza MRP cost: 199.95

Pizza Rating: 8

Product number :6

Pizza name: Onion Tomato Corn Pizza

Pizza MRP cost: 250.5

Pizza Rating: 8

Product number :7

Pizza name: Hawaiian Pizza

Pizza MRP cost: 225.0

Pizza Rating: 7

Product number :8

Pizza name: Italy Special Pizza

Pizza MRP cost: 175.75

Pizza Rating: 10

Product number :9

Pizza name: New-York Style Pizza

Pizza MRP cost: 99.99

Pizza Rating: 8

Product number :10

Pizza name: Cicilian Pizza

Pizza MRP cost: 100.0

Pizza Rating: 9

Product number :11

Pizza name: Sharma's special

Pizza MRP cost: 9.0

Pizza Rating: 260

Type 1 for Entering values in existing tables OR

Type 2 for Viewing data in existing tables OR

Type 3 for Updating existing data present in tables

Type 6 to exit program

3

Please specify the table whose data you want to update

(Type 1 then press enter for)ORDERS

(Type 2 then press enter for)PRODUCTS

(Type 3 then press enter for)CUSTOMERS

(Type 4 then press enter for)EMPLOYEES

2

enter ProductID whose record is to be updated11

Please select: (a)update product name,(b)update product priceb

enter new product cost 200

Type 1 for Entering values in existing tables OR

Type 2 for Viewing data in existing tables OR

Type 3 for Updating existing data present in tables

Type 6 to exit program

2

Please specify the table whose data you want to view-(1)orders,(2)products (3)Customers ,(4)Employees

2

Enter how you want to see data by-By searching(type 2) or do want to see the entire data(type 1) then press enter

2

You can search by id(type 1) or by using Name(type 2) then press enter

1

Enter id: 11

Product Id: 11

Pizza Name :Sharma's special

Cost:,200.0

Rating:260

Type 1 for Entering values in existing tables OR

Type 2 for Viewing data in existing tables OR

Type 3 for Updating existing data present in tables

Type 6 to exit program

6

Exiting program

Process finished with exit code 0