Populate.java source code Snapshots

```
import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.sql.*;
import java.util.ArrayList;

public class Hw {

    static Connection connection;

    public static void query1(int vertices, String[] q1){
        try {
            String  loopPolygonCoord = q1[0] +" "+ q1[1];
            String polygonCoord = "";
            for(int i=0; i < vertices*2; i++){
                polygonCoord += q1[i] + " ";
            }
            polygonCoord = (polygonCoord + loopPolygonCoord).trim().replaceAll("(\\s[^\\s]*)\\s", "$1,");

            String rangeSql = "Select incident.Unique_incident_ID, ST_AsText(Incident_Coord), " +
                    "incident.Type_of_incident FROM incident WHERE ST_CONTAINS(ST_PolygonFromText(?), " +
                    "incident.Incident_Coord) order by Unique_incident_ID;";

            PreparedStatement prepStmt = connection.prepareStatement(rangeSql);
            String zoneString = "POLYGON((" + polygonCoord + "))";
            prepStmt.setString(1, zoneString);
            ResultSet resultSet = prepStmt.executeQuery();


                while (resultSet.next()) {
                    int incidentID = resultSet.getInt("Unique_incident_ID");
                    String[] incidentCoordArray = resultSet.getString("ST_AsText(Incident_Coord)")
                            .replaceAll(".*\\(|\\).*", "").replaceFirst("\\s", ",")
                            .split(",");
                    String incidentCoord = incidentCoordArray[1] + "," + incidentCoordArray[0];
                    String incidentType = resultSet.getString("Type_of_incident");

                    String incidentType = resultSet.getString("Type_of_incident");
                    System.out.println(incidentID + " " + incidentCoord + " " + incidentType);
                }

            connection.close();
        }catch (SQLException sqe){
            System.out.println("Functions not run correctly, please check the PrepareStatement functions");
        }
    }

    public static void query2(int incidentID, int distance){
        try {
            String findDistanceSql = "Select officer.Unique_Badge_number,ST_Distance_Sphere(incident.Incident_Coord, " +
                    "officer.Officer_location) as dist, officer.Officer_Name from officer,incident " +
                    "where incident.Unique_incident_ID = ? HAVING dist<=? order by dist";
            PreparedStatement prepStatement = connection.prepareStatement(findDistanceSql);
            prepStatement.setInt(1,incidentID);
            prepStatement.setInt(2,distance);
            ResultSet resultSet = prepStatement.executeQuery();


                while (resultSet.next()) {
                    int badgeNum = resultSet.getInt("officer.Unique_Badge_number");
                    int officerDistance = Math.round(resultSet.getFloat("dist"));
                    String officerName = resultSet.getString("officer.Officer_Name");
                    System.out.println(badgeNum + "  " + officerDistance + "m  " + officerName);
                }
            connection.close();
        }catch (SQLException sqe){
            System.out.println("Functions not run correctly, please check the PrepareStatement functions");
        }
    }

    public static void query3(int squadNum){
        try {
            String findZoneSql = "Select Zone_Name, ST_AsText(Zone_Coord) from zone where Squad_Number=?";
            PreparedStatement statement = connection.prepareStatement(findZoneSql);
```

```
        PreparedStatement statement = connection.prepareStatement(findZoneSql);
        statement.setInt(1, squadNum);
        ResultSet resultSet = statement.executeQuery();
        String zoneCoord= null;
        while(resultSet.next()) {
            String zoneName = resultSet.getString("Zone_Name");
             zoneCoord = resultSet.getString("ST_AsText(Zone_Coord)");
            System.out.println("Sqaud "+squadNum+" is now patrolling: " + zoneName);
        }

            String findOfficerSql = "Select officer.Unique_Badge_number,IF(ST_CONTAINS(ST_PolygonFromText(?), " +
                    "officer.officer_location) = 1, 'IN', 'OUT'), officer.Officer_Name from officer " +
                    "where officer.Squad_Number=?";
            PreparedStatement statement1 = connection.prepareStatement(findOfficerSql);
            statement1.setString(1,zoneCoord);
            statement1.setInt(2, squadNum);
            ResultSet resultSet1 = statement1.executeQuery();

                while (resultSet1.next()) {
                    int badgeNum = resultSet1.getInt("Unique_Badge_number");
                    String officerName = resultSet1.getString("Officer_Name");
                    String location = resultSet1.getString(2);
                    System.out.println(badgeNum + "  " + location + "  " + officerName);
                }
        connection.close();
        }catch (SQLException sqe){
        System.out.println("Functions not run correctly, please check the PrepareStatement functions");
    }
}

public static void query4(int routeNum){
    try {
        String routeCoverageSql = "Select zone.Zone_ID, zone.Zone_Name from route, zone " +
                "where route.Unique_route_number = ? AND ST_Intersects(zone.Zone_Coord, route.Route_Coord);";
        PreparedStatement preparedStatement = connection.prepareStatement(routeCoverageSql);
        preparedStatement.setInt(1,routeNum);
        ResultSet resultSet = preparedStatement.executeQuery();

        ResultSet resultSet = preparedStatement.executeQuery();


            while (resultSet.next()) {
                int zoneID = resultSet.getInt("zone.Zone_ID");
                String zoneName = resultSet.getString("zone.Zone_Name");
                System.out.println(zoneID + " " + zoneName);
            }
        connection.close();
    }catch (SQLException sqe){
        System.out.println("Functions not run correctly, please check the PrepareStatement functions");
    }
}

public static void createConnection(String fileName){
    try {
        Class.forName("com.mysql.jdbc.Driver");
        BufferedReader bufferedReader = new BufferedReader(new FileReader(fileName));
        ArrayList<String> list = new ArrayList<String>();
        String str;
        while ((str = bufferedReader.readLine()) != null) {
            list.add(str);
        }
        String host = list.get(0);
        String port = list.get(1);
        String DB = list.get(2);
        String userName = list.get(3);
        String password = list.get(4);
        connection = DriverManager.getConnection("jdbc:mysql://" + host + ":" + port + "/" + DB + "?useSSL=false", userName  ,
        password);
    } catch (
FileNotFoundException fn){
    System.out.println("File not found please verify the path for db.properties file ");
}catch(
IOException io){
    System.out.println("Error in accessing the data in file db.properties");
}catch (
```

```
    }catch (
    SQLException sqe){
        System.out.println("SqlException found");
    }catch (ClassNotFoundException cnf){
            System.out.printf("forName class not found");
        }
    }

    public static void main(String[] args) {

        try {
            createConnection(args[0]);
            String queryNum = args[1];

            if (queryNum.equals("q1")) {
                int vertices = Integer.parseInt(args[2]);
                String[] q1 = new String[vertices * 2];
                for (int j = 0, i = 3; j < 8 && i < (vertices * 2 + 3); j++, i++) {
                    q1[j] = args[i];
                }
                query1(vertices, q1);
            } else if (queryNum.equals("q2")) {
                int incidentID = Integer.parseInt(args[2]);
                int distance = Integer.parseInt(args[3]);
                query2(incidentID, distance);

            } else if (queryNum.equals("q3")) {
                int squadNum = Integer.parseInt(args[2]);
                query3(squadNum);
            } else if (queryNum.equals("q4")) {
                int routeNum = Integer.parseInt(args[2]);
                query4(routeNum);
            }

        }catch (ArrayIndexOutOfBoundsException ae){
            System.out.println("Please pass correct number of arguments");
        }
    }
}
```