Relational Databases with MySQL Week 4 Coding Assignment

Points possible: 70

Category	Criteria	% of Grade
Functionality	Does the code work?	25
Organization	Is the code clean and organized? Proper use of white space, syntax, and consistency are utilized. Names and comments are concise and clear.	25
Creativity	Student solved the problems presented in the assignment using creativity and out of the box thinking.	25
Completeness	All requirements of the assignment are complete.	25

Instructions: In Eclipse, or an IDE of your choice, write the code that accomplishes the objectives listed below. Ensure that the code compiles and runs as directed. Take screenshots of the code and of the running program (make sure to get screenshots of all required functionality) and paste them in this document where instructed below. Create a new repository on GitHub for this week's assignments and push this document to the repository. Additionally, push an .sql file with all your queries and your Java project code to the same repository. Add the URL for this week's repository to this document where instructed and submit this document to your instructor when complete.

Coding Steps:

In this week's coding activity, you will create a menu driven application backed by a MySQL database.

To start, choose one item that you like. It could be vehicles, sports, foods, etc....

Create a new Java project in Eclipse.

Create a SQL script in the project to create a database with one table. The table should be the item you picked.

Write a Java menu driven application that allows you to perform all four CRUD operations on your table.

Tips:

The application does not need to be as complex as the example in the video curriculum.

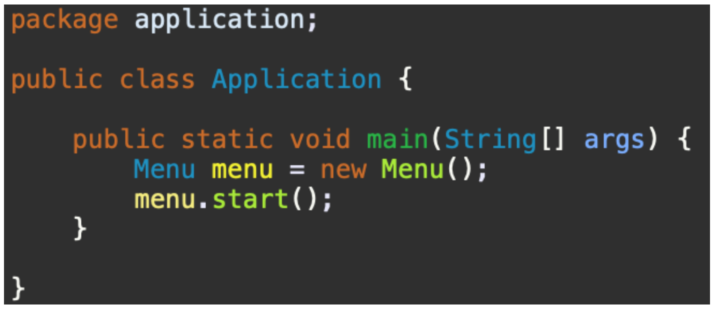
You need an option for each of the CRUD operations (Create, Read, Update, and Delete).

Remember that PreparedStatment.executeQuery() is only for Reading data and .executeUpdate() is used for Creating, Updating, and Deleting data.

Remember that both parameters on PreparedStatements and the ResultSet columns are based on indexes that start with 1, not 0.

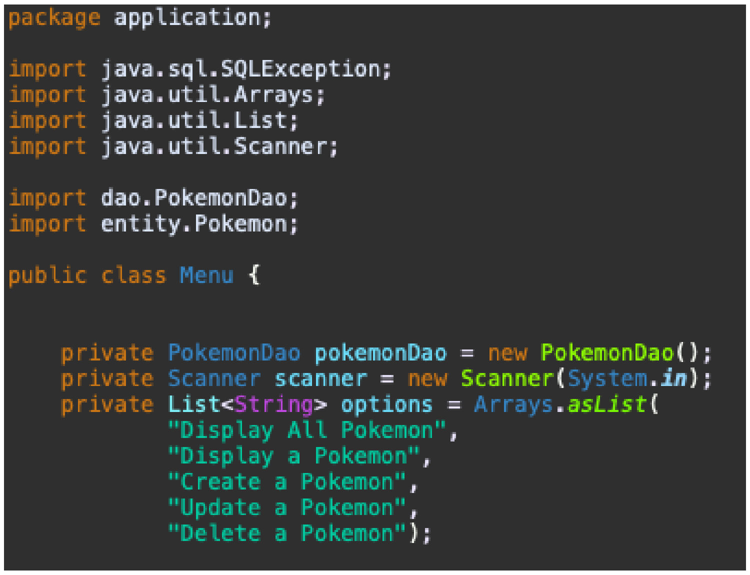
Screenshots of Code:

Application.java

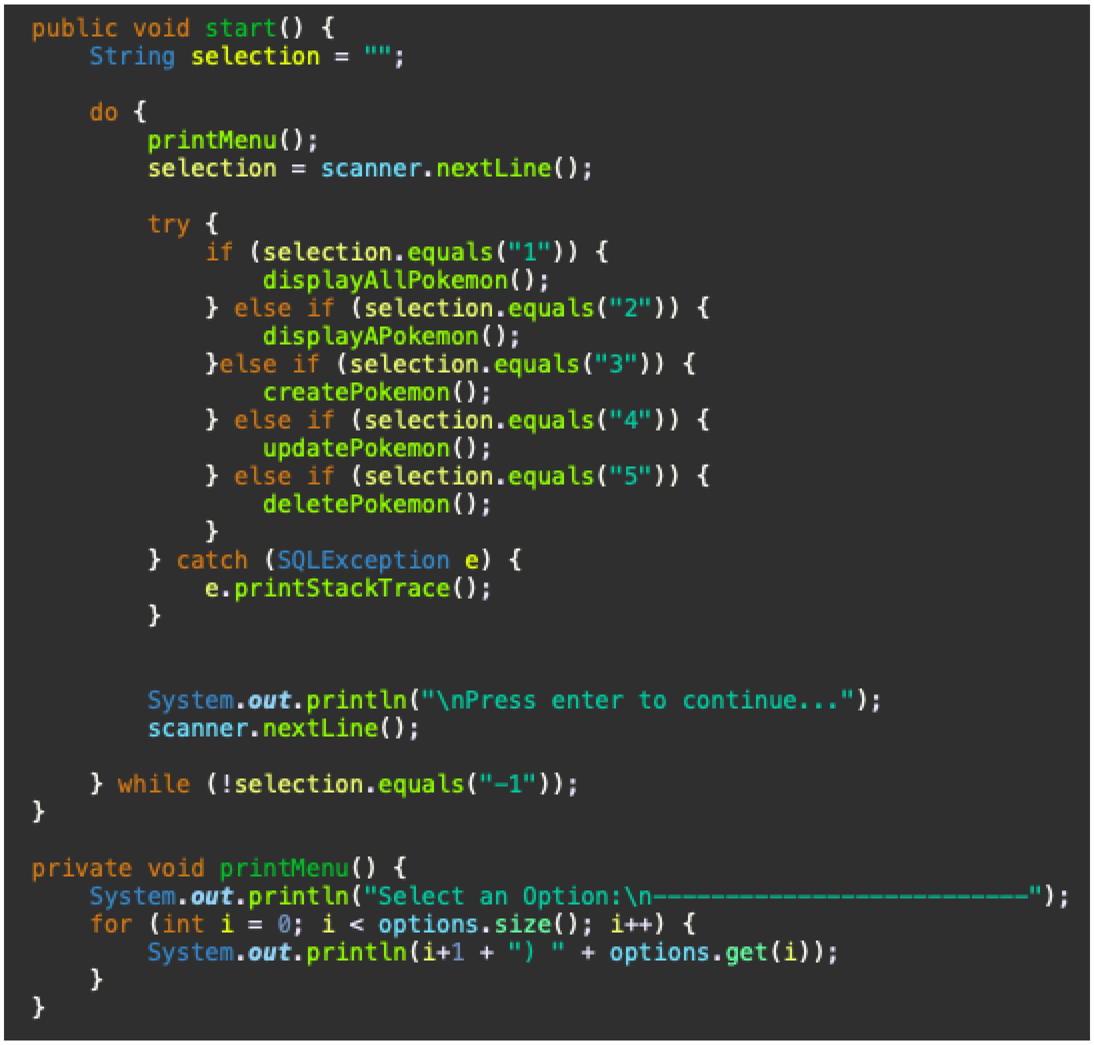


```java
package application;

public class Application {

    public static void main(String[] args) {
        Menu menu = new Menu();
        menu.start();
    }

}
```

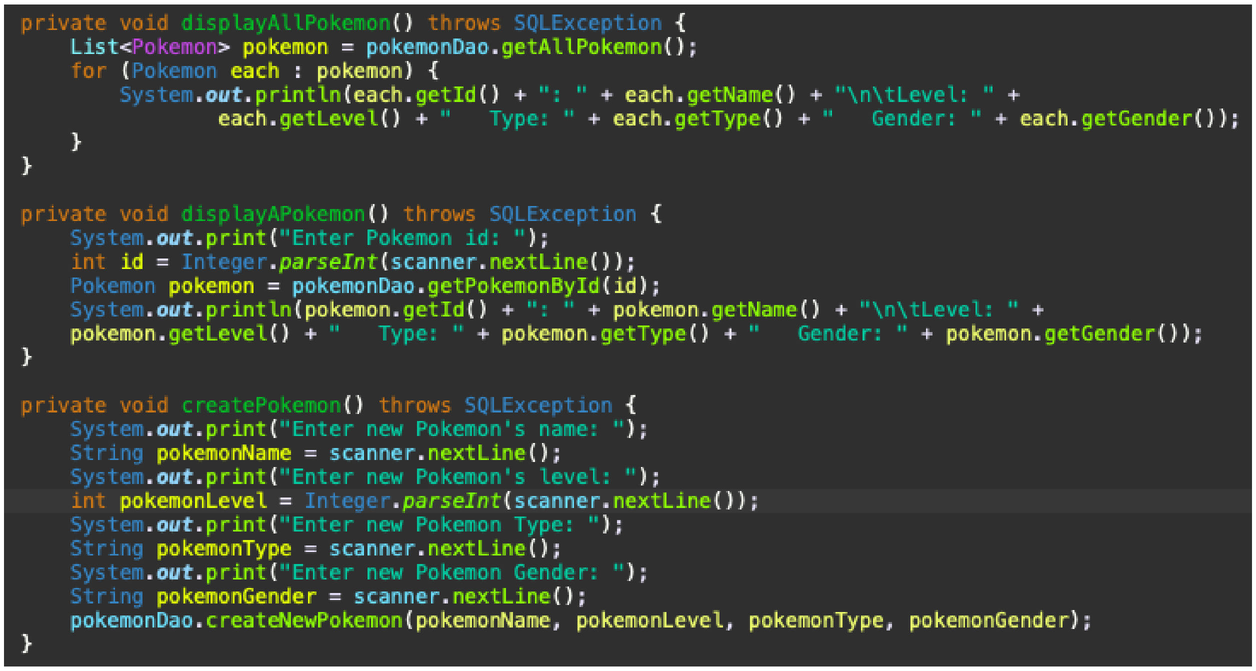
Menu.java



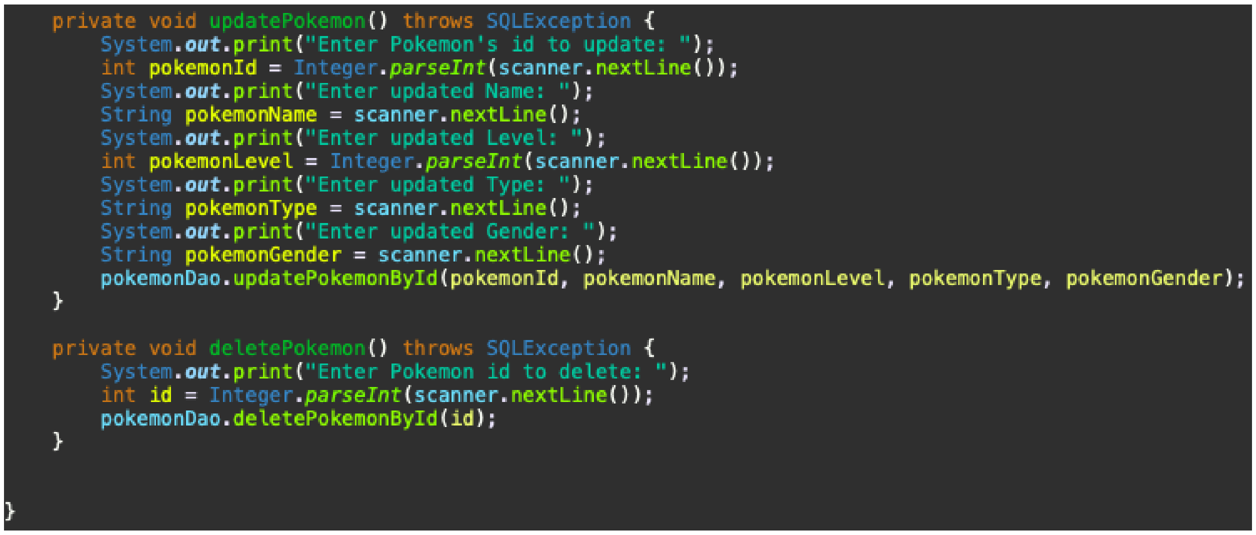
```java
package application;

import java.sql.SQLException;
import java.util.Arrays;
import java.util.List;
import java.util.Scanner;

import dao.PokemonDao;
import entity.Pokemon;

public class Menu {


    private PokemonDao pokemonDao = new PokemonDao();
    private Scanner scanner = new Scanner(System.in);
    private List<String> options = Arrays.asList(
            "Display All Pokemon",
            "Display a Pokemon",
            "Create a Pokemon",
            "Update a Pokemon",
            "Delete a Pokemon");
```



```java
public void start() {
    String selection = "";

    do {
        printMenu();
        selection = scanner.nextLine();

        try {
            if (selection.equals("1")) {
                displayAllPokemon();
            } else if (selection.equals("2")) {
                displayAPokemon();
            }else if (selection.equals("3")) {
                createPokemon();
            } else if (selection.equals("4")) {
                updatePokemon();
            } else if (selection.equals("5")) {
                deletePokemon();
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }


        System.out.println("\nPress enter to continue...");
        scanner.nextLine();

    } while (!selection.equals("-1"));
}

private void printMenu() {
    System.out.println("Select an Option:\n--------------------------");
    for (int i = 0; i < options.size(); i++) {
        System.out.println(i+1 + ") " + options.get(i));
    }
}
```

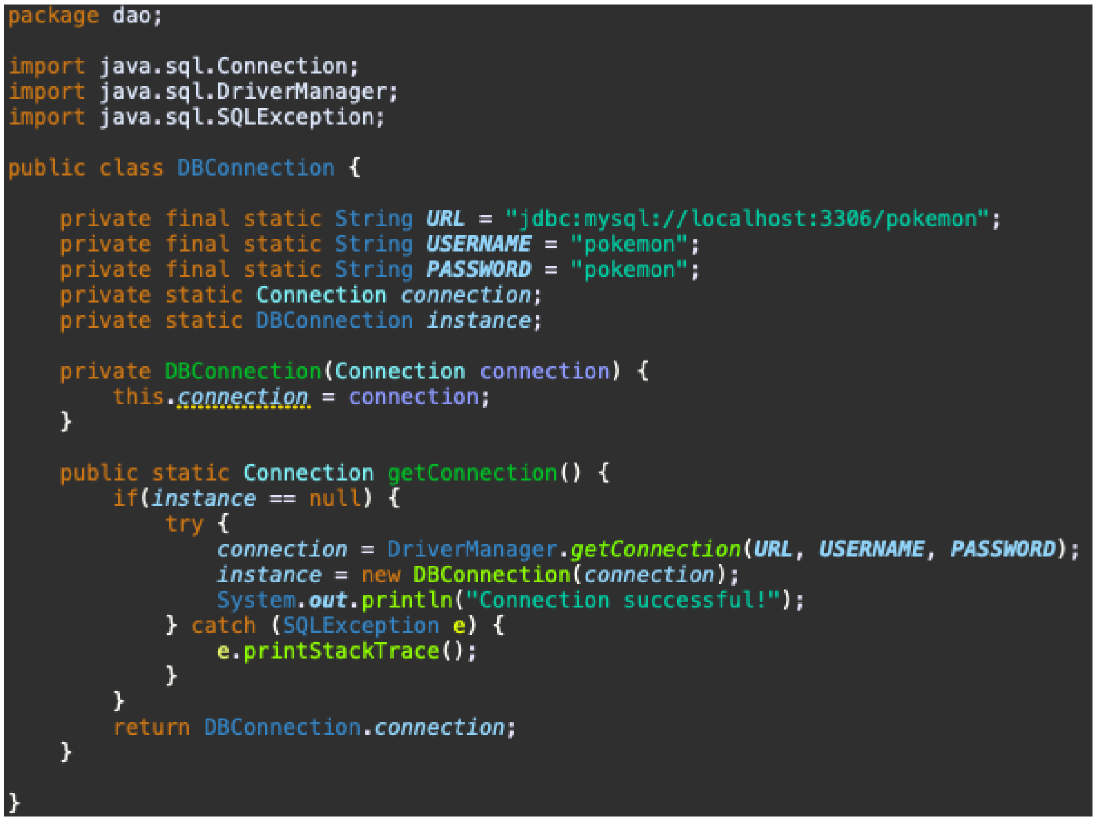


```java
private void displayAllPokemon() throws SQLException {
    List<Pokemon> pokemon = pokemonDao.getAllPokemon();
    for (Pokemon each : pokemon) {
        System.out.println(each.getId() + ": " + each.getName() + "\n\tLevel: " +
                each.getLevel() + "   Type: " + each.getType() + "   Gender: " + each.getGender());
    }
}

private void displayAPokemon() throws SQLException {
    System.out.print("Enter Pokemon id: ");
    int id = Integer.parseInt(scanner.nextLine());
    Pokemon pokemon = pokemonDao.getPokemonById(id);
    System.out.println(pokemon.getId() + ": " + pokemon.getName() + "\n\tLevel: " +
    pokemon.getLevel() + "   Type: " + pokemon.getType() + "   Gender: " + pokemon.getGender());
}

private void createPokemon() throws SQLException {
    System.out.print("Enter new Pokemon's name: ");
    String pokemonName = scanner.nextLine();
    System.out.print("Enter new Pokemon's level: ");
    int pokemonLevel = Integer.parseInt(scanner.nextLine());
    System.out.print("Enter new Pokemon Type: ");
    String pokemonType = scanner.nextLine();
    System.out.print("Enter new Pokemon Gender: ");
    String pokemonGender = scanner.nextLine();
    pokemonDao.createNewPokemon(pokemonName, pokemonLevel, pokemonType, pokemonGender);
}
```



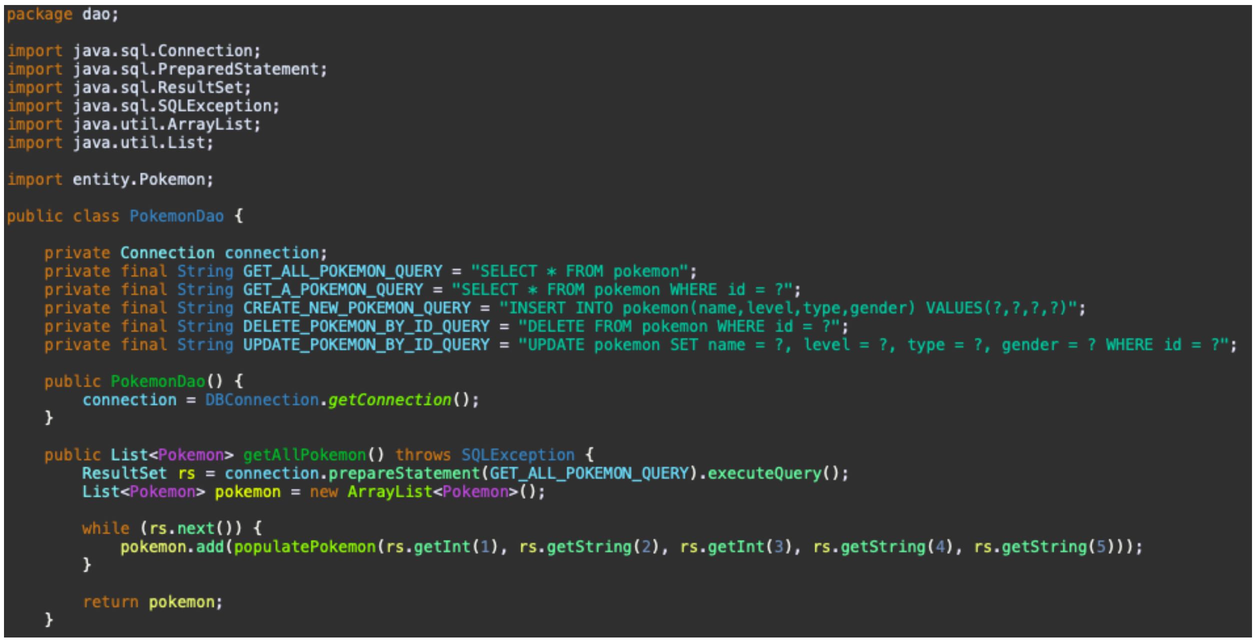
```java
    private void updatePokemon() throws SQLException {
        System.out.print("Enter Pokemon's id to update: ");
        int pokemonId = Integer.parseInt(scanner.nextLine());
        System.out.print("Enter updated Name: ");
        String pokemonName = scanner.nextLine();
        System.out.print("Enter updated Level: ");
        int pokemonLevel = Integer.parseInt(scanner.nextLine());
        System.out.print("Enter updated Type: ");
        String pokemonType = scanner.nextLine();
        System.out.print("Enter updated Gender: ");
        String pokemonGender = scanner.nextLine();
        pokemonDao.updatePokemonById(pokemonId, pokemonName, pokemonLevel, pokemonType, pokemonGender);
    }

    private void deletePokemon() throws SQLException {
        System.out.print("Enter Pokemon id to delete: ");
        int id = Integer.parseInt(scanner.nextLine());
        pokemonDao.deletePokemonById(id);
    }


}
```

<u>DBConnection.java</u>

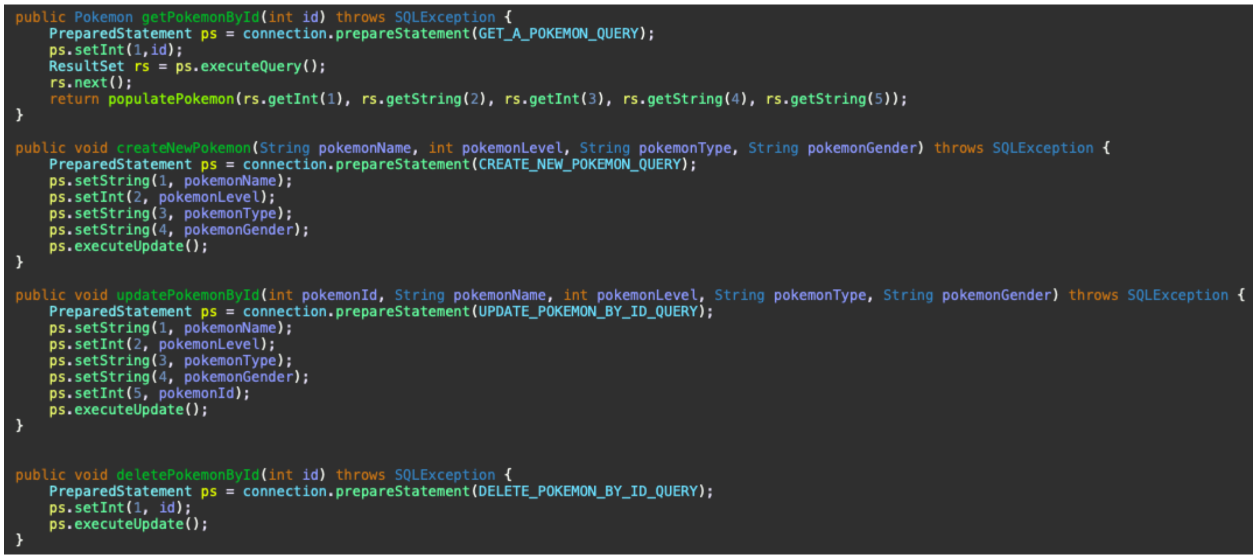


```java
package dao;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class DBConnection {

    private final static String URL = "jdbc:mysql://localhost:3306/pokemon";
    private final static String USERNAME = "pokemon";
    private final static String PASSWORD = "pokemon";
    private static Connection connection;
    private static DBConnection instance;

    private DBConnection(Connection connection) {
        this.connection = connection;
    }

    public static Connection getConnection() {
        if(instance == null) {
            try {
                connection = DriverManager.getConnection(URL, USERNAME, PASSWORD);
                instance = new DBConnection(connection);
                System.out.println("Connection successful!");
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
        return DBConnection.connection;
    }

}
```

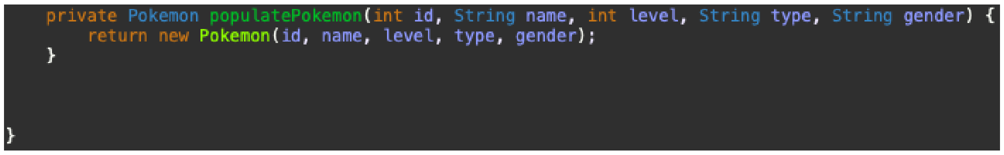
PokemonDao.java



```java
package dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

import entity.Pokemon;

public class PokemonDao {

    private Connection connection;
    private final String GET_ALL_POKEMON_QUERY = "SELECT * FROM pokemon";
    private final String GET_A_POKEMON_QUERY = "SELECT * FROM pokemon WHERE id = ?";
    private final String CREATE_NEW_POKEMON_QUERY = "INSERT INTO pokemon(name,level,type,gender) VALUES(?,?,?,?)";
    private final String DELETE_POKEMON_BY_ID_QUERY = "DELETE FROM pokemon WHERE id = ?";
    private final String UPDATE_POKEMON_BY_ID_QUERY = "UPDATE pokemon SET name = ?, level = ?, type = ?, gender = ? WHERE id = ?";

    public PokemonDao() {
        connection = DBConnection.getConnection();
    }

    public List<Pokemon> getAllPokemon() throws SQLException {
        ResultSet rs = connection.prepareStatement(GET_ALL_POKEMON_QUERY).executeQuery();
        List<Pokemon> pokemon = new ArrayList<Pokemon>();

        while (rs.next()) {
            pokemon.add(populatePokemon(rs.getInt(1), rs.getString(2), rs.getInt(3), rs.getString(4), rs.getString(5)));
        }

        return pokemon;
    }
```

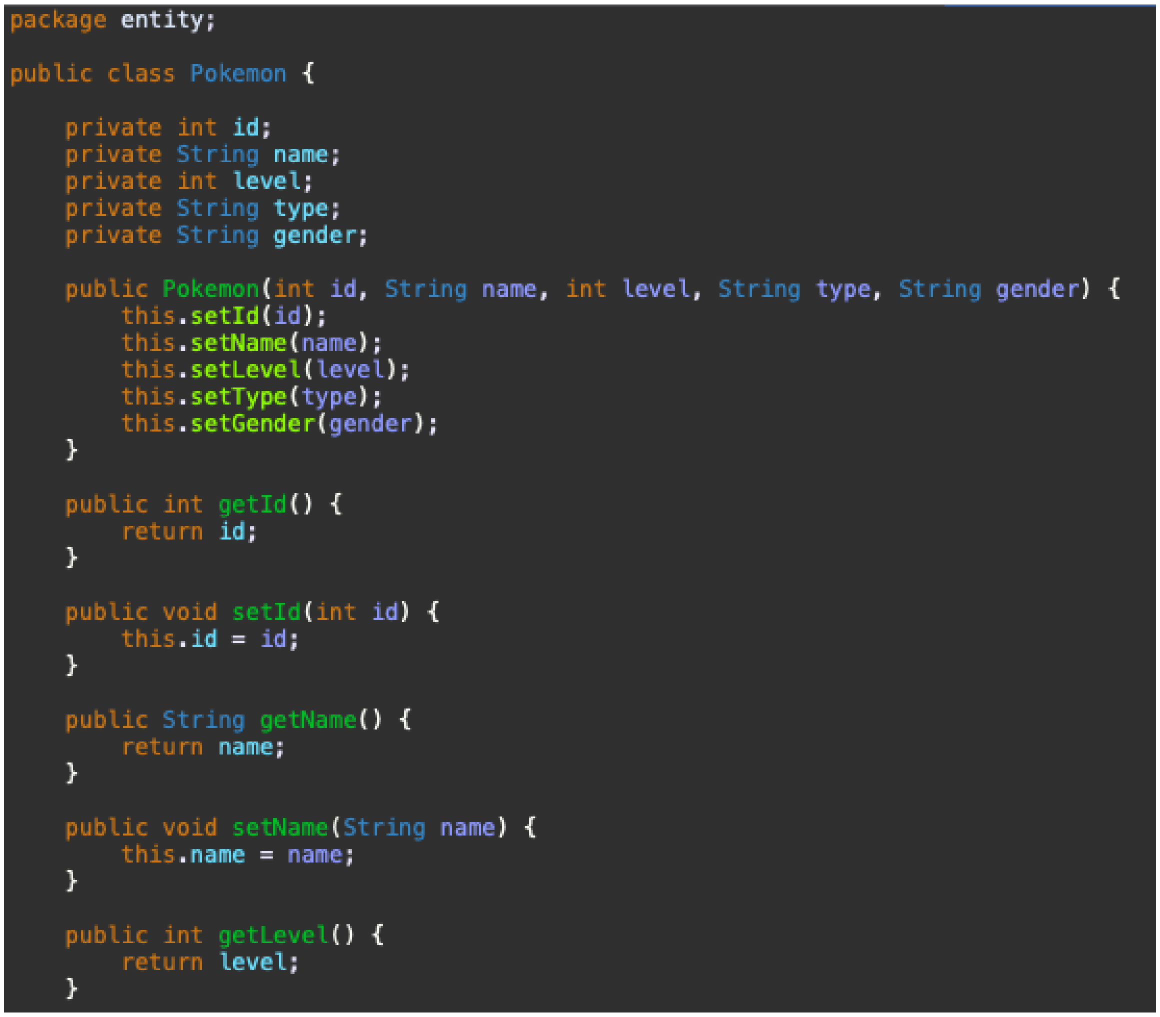


```java
public Pokemon getPokemonById(int id) throws SQLException {
    PreparedStatement ps = connection.prepareStatement(GET_A_POKEMON_QUERY);
    ps.setInt(1,id);
    ResultSet rs = ps.executeQuery();
    rs.next();
    return populatePokemon(rs.getInt(1), rs.getString(2), rs.getInt(3), rs.getString(4), rs.getString(5));
}

public void createNewPokemon(String pokemonName, int pokemonLevel, String pokemonType, String pokemonGender) throws SQLException {
    PreparedStatement ps = connection.prepareStatement(CREATE_NEW_POKEMON_QUERY);
    ps.setString(1, pokemonName);
    ps.setInt(2, pokemonLevel);
    ps.setString(3, pokemonType);
    ps.setString(4, pokemonGender);
    ps.executeUpdate();
}

public void updatePokemonById(int pokemonId, String pokemonName, int pokemonLevel, String pokemonType, String pokemonGender) throws SQLException {
    PreparedStatement ps = connection.prepareStatement(UPDATE_POKEMON_BY_ID_QUERY);
    ps.setString(1, pokemonName);
    ps.setInt(2, pokemonLevel);
    ps.setString(3, pokemonType);
    ps.setString(4, pokemonGender);
    ps.setInt(5, pokemonId);
    ps.executeUpdate();
}

public void deletePokemonById(int id) throws SQLException {
    PreparedStatement ps = connection.prepareStatement(DELETE_POKEMON_BY_ID_QUERY);
    ps.setInt(1, id);
    ps.executeUpdate();
}
```

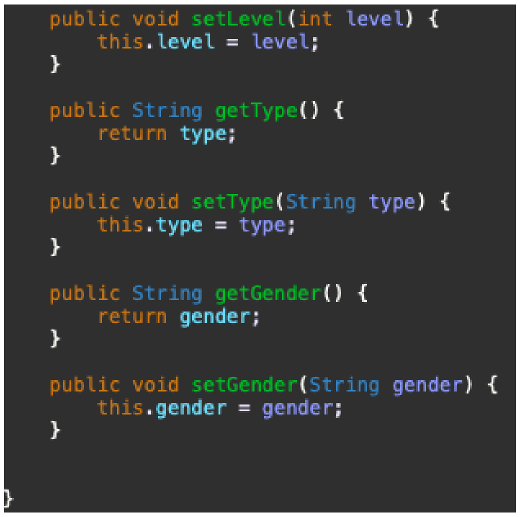


```java
    private Pokemon populatePokemon(int id, String name, int level, String type, String gender) {
        return new Pokemon(id, name, level, type, gender);
    }



}
```

Pokemon.java

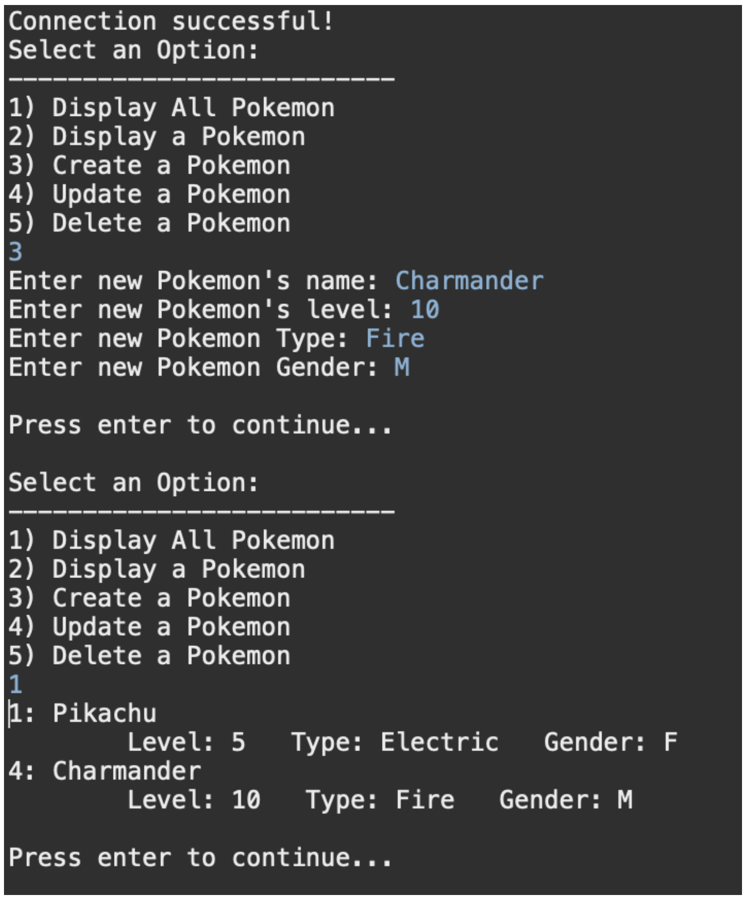


```java
package entity;

public class Pokemon {

    private int id;
    private String name;
    private int level;
    private String type;
    private String gender;

    public Pokemon(int id, String name, int level, String type, String gender) {
        this.setId(id);
        this.setName(name);
        this.setLevel(level);
        this.setType(type);
        this.setGender(gender);
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getLevel() {
        return level;
    }
```



```java
    public void setLevel(int level) {
        this.level = level;
    }

    public String getType() {
        return type;
    }

    public void setType(String type) {
        this.type = type;
    }

    public String getGender() {
        return gender;
    }

    public void setGender(String gender) {
        this.gender = gender;
    }


}
```

Screenshots of Running Application:



```
Connection successful!
Select an Option:
--------------------------
1) Display All Pokemon
2) Display a Pokemon
3) Create a Pokemon
4) Update a Pokemon
5) Delete a Pokemon
3
Enter new Pokemon's name: Charmander
Enter new Pokemon's level: 10
Enter new Pokemon Type: Fire
Enter new Pokemon Gender: M

Press enter to continue...

Select an Option:
--------------------------
1) Display All Pokemon
2) Display a Pokemon
3) Create a Pokemon
4) Update a Pokemon
5) Delete a Pokemon
1
1: Pikachu
        Level: 5   Type: Electric   Gender: F
4: Charmander
        Level: 10   Type: Fire   Gender: M

Press enter to continue...
```

URL to GitHub Repository:

https://github.com/DukesGuy/MySQL/tree/main/MySQL_Week4