

JDBC Assignments1

Objective: At the end of the assignments, participants will be able to perform insert(Create) operations out of CRUD operations on database tables using JDBC.

- 1) Create a dao class named EmployeeDao which should have
addEmployee(Employee employee) method which would be using JDBC
to insert the values from the employee object into the Employee table.
You also need to create a test class called TestEmployeeDao which would
test the methods of EmployeeDao using the main method.

For solving the above problem, you need to first create the following table and classes:

Create the Employee table in the database with naming convention TBL_<Your
Employee Id>_EMPLOYEE.

The table should have the following columns:

EMPLOYEE_ID integer (Primary Key)

EMPLOYEE_NAME varchar2(256)

GENDER varchar2(256)

DESIGNATION varchar2(256)

EMAIL varchar2(256)

Create Employee class with the following attributes:

employeeId

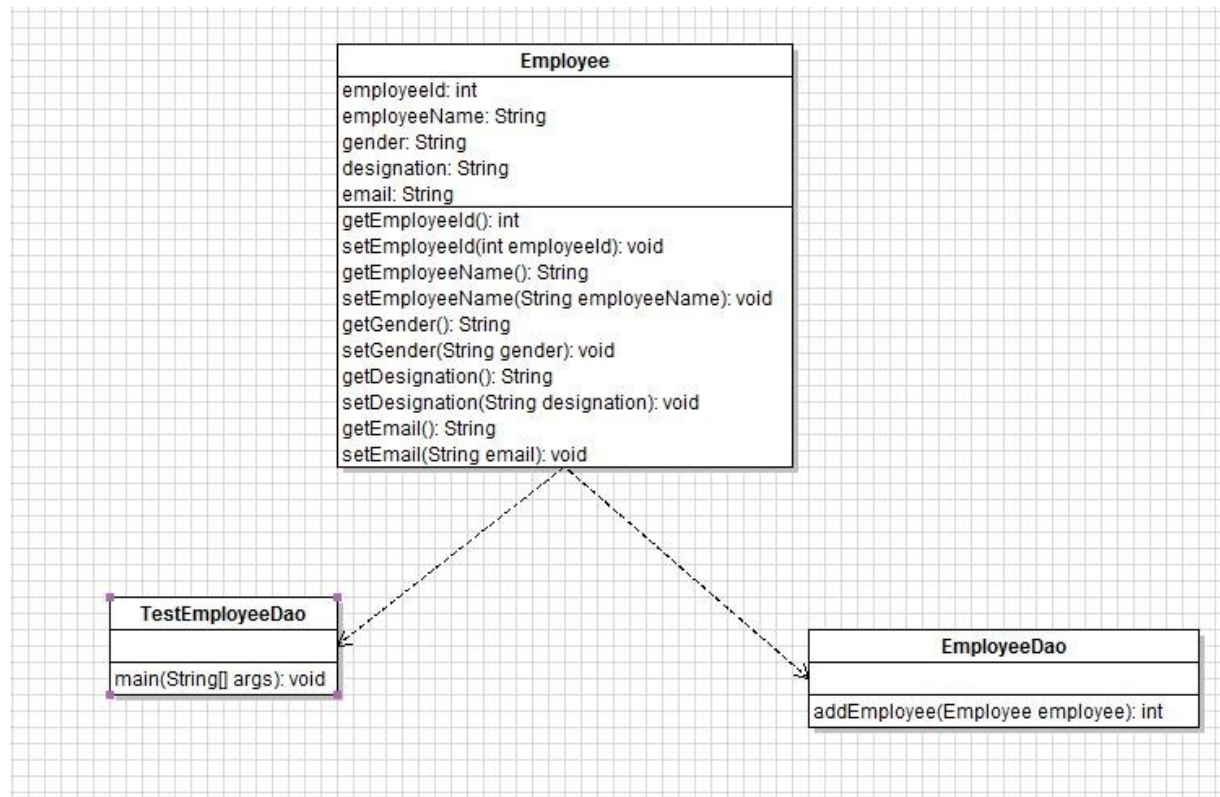
employeeName

gender

designation

email

You may take help from the following class diagram to create these classes:



Solution:

We create the table using the following query:

In the below table name, please replace 387105 with your employee Id.

```
CREATE TABLE TBL_387105_EMPLOYEE
(
  EMPLOYEE_ID      INTEGER NOT NULL,
  EMPLOYEE_NAME     VARCHAR2 (256),
  GENDER            VARCHAR2 (256),
  DESIGNATION       VARCHAR2 (256),
  EMAIL             VARCHAR2 (256),
  PRIMARY KEY (EMPLOYEE_ID)
)
```

Employee.java

```
package com.tcs.ilp.crud.bean;
public class Employee {
    private int employeeId;
    private String employeeName;
    private String gender;
    private String designation;
    private String email;
    public int getEmployeeId() {
        return employeeId;
    }
    public void setEmployeeId(int employeeId) {
        this.employeeId = employeeId;
    }
    public String getEmployeeName() {
        return employeeName;
    }
    public void setEmployeeName(String employeeName) {
        this.employeeName = employeeName;
    }
    public String getGender() {
        return gender;
    }
    public void setGender(String gender) {
        this.gender = gender;
    }
    public String getDesignation() {
        return designation;
    }
    public void setDesignation(String designation) {
        this.designation = designation;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
}
```

EmployeeDao.java

```
package com.tcs.ilp.crud.dao;

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

import com.tcs.ilp.crud.bean.Employee;

public class EmployeeDao {
    private String driver = "oracle.jdbc.driver.OracleDriver";
    private String dbURL = "jdbc:oracle:thin:@172.26.132.40:1521:orclilp";
    private String dbUserName = "a88b";
    private String dbPassword = "a88b";
    private Connection con = null;

    public int addEmployee(Employee employee) {
        int numberOfEmployeesAdded = 0;
        try {
            Class.forName(driver);
            con = DriverManager.getConnection(dbURL, dbUserName, dbPassword);
            PreparedStatement pst = con
                .prepareStatement("insert into TBL_387105_EMPLOYEE values (?, ?, ?, ?, ?)");
            pst.setInt(1, employee.getEmployeeId());
            pst.setString(2, employee.getEmployeeName());
            pst.setString(3, employee.getGender());
            pst.setString(4, employee.getDesignation());
            pst.setString(5, employee.getEmail());
            numberOfEmployeesAdded = pst.executeUpdate();
            con.commit();
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            if (con != null) {
                try {
                    con.close();
                } catch (SQLException e) {
                    e.printStackTrace();
                }
            }
        }
        return numberOfEmployeesAdded;
    }
}
```

TestEmployeeDao.java

```
import com.tcs.ilp.crud.bean.Employee;
import com.tcs.ilp.crud.dao.EmployeeDao;

public class TestEmployeeDao {
    public static void main(String[] args) {
        EmployeeDao employeeDao = new EmployeeDao();

        Employee employee = new Employee();
        employee.setEmployeeId(172785);
        employee.setEmployeeName("Ram Kumar");
        employee.setGender("Male");
        employee.setDesignation("Manager");
        employee.setEmail("ram.k@gmail.com");
        employeeDao.addEmployee(employee);
    }
}
```

After running this test program, the data provided through main method must be inserted into the database. And the following screen shows that this data is inserted into the table:

	EMPLOYEE_ID	EMPLOYEE_NAME	GENDER	DESIGNATION	EMAIL
	NUMBER (38)	VARCHAR2 (256)	VARCHAR2 (256)	VARCHAR2 (256)	VARCHAR2 (256)
1	172,785	Ram Kumar	Male	Manager	ram.k@gmail.com.com

Each trainee is supposed to write above programs and try out even if they are aware about JDBC. Along with learning JDBC, below points are very important to practice and applicable throughout this ILP training as well as most important for any software code.

- Use exactly same class names as mentioned
- Use exactly same method signature (method name, return type, method parameter type, position of each method parameter)
- Define attributes with same name and data type as given in class outline/diagram.
- Define getter setters as given in the class outline/diagram.
- Ensure attributes are private and other methods which will be called from main method, getter-setter methods and constructor is public.
- Use main method only for input and output and testing object creation and object methods.

As mentioned above, any logic which may be 100% correct is not valid if above points are not taken care. Hence, simply building logic does not certify us as project ready. Building exact and complete solution does.

2) Create a dao class named StudentDao which would have addStudent(Student student) method which would be using JDBC to insert the values from the student object into the Student table.

You also need to create a test class called TestStudentDao which would test the methods of StudentDao using the main method.

For solving the above problem, you need to first create the following table and classes:

Create the Student table in the database with naming convention TBL_<Your Employee Id>_STUDENT.

The table should have the following columns:

STUDENT_ID integer (Primary Key)

STUDENT_NAME varchar2(256)

GENDER varchar2(256)

COURSE_NAME varchar2(256)

ADDRESS varchar2(256)

Create Student class with the following attributes:

studentId

studentName

gender

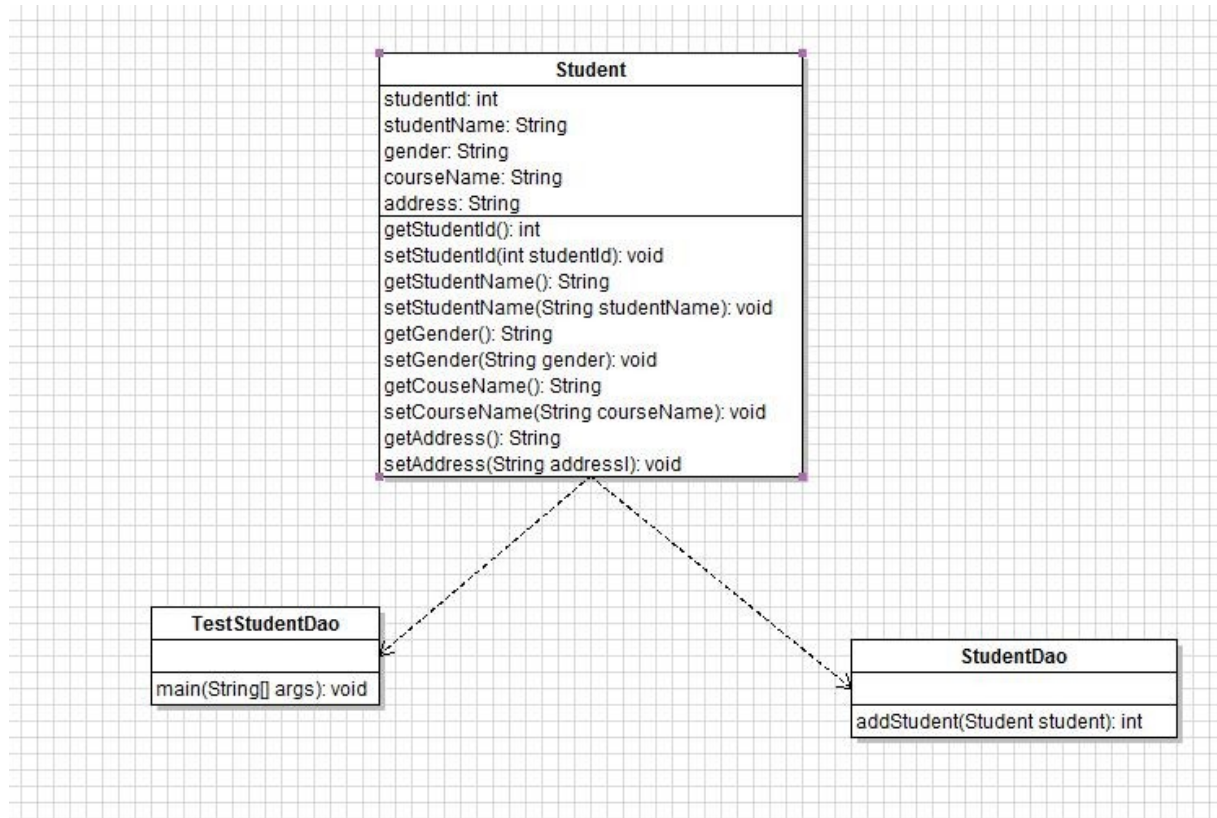
courseName

address

Now create a dao class named StudentDao which would have addStudent(Student student) method which would be using JDBC to insert the values from the student object into the Student table which was created earlier.

You also need to create a test class called TestStudentDao which would test the methods from StudentDao using the main method..

Please follow the following class diagram to create these classes:



3) Create a dao class named **CustomerDao** which would have `addCustomer(Customer customer)` method which would be using JDBC to insert the values from the customer object into the **Customer** table. You also need to create a test class called **TestCustomerDao** which would test the methods of **CustomerDao** using the `main` method.

For solving the above problem, you need to first create the following table and classes:

Create the **Customer** table in the database with naming convention `TBL_<Your Employee Id>_CUSTOMER`.

The table should have the following columns:

CUSTOMER_ID integer (Primary Key)

CUSTOMER_NAME varchar2(256)

DATE_OF_BIRTH date

ADDRESS varchar2(256)

Create Customer class with the following attributes:

customerId

customerName

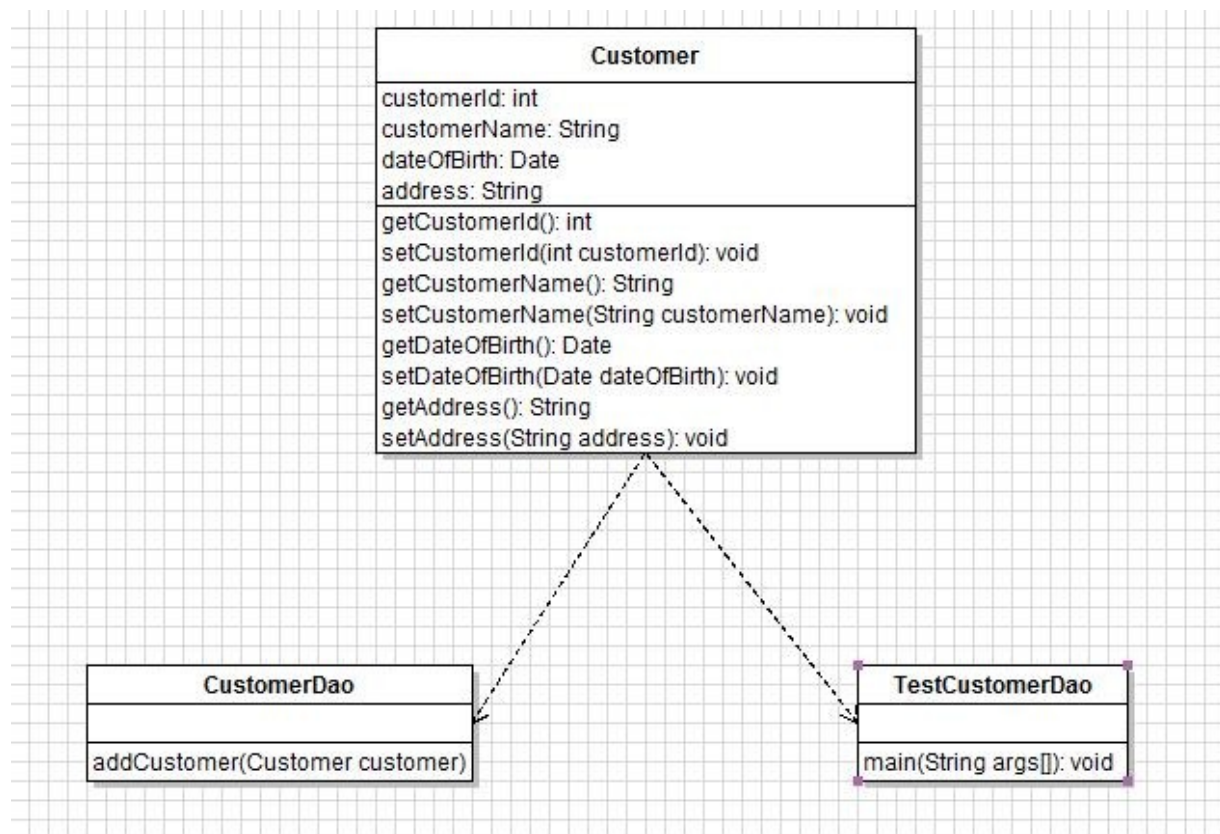
dateOfBirth

address

Now create a dao class named CustomerDao which would have addCustomer(Customer customer) method which would be using JDBC to insert the values from the customer object into the Customer table which was created earlier.

You also need to create a test class called TestCustomerDao which would test the methods of CustomerDao using the main method..

Please follow the following class diagram to create these classes:



JDBC Assignments 2

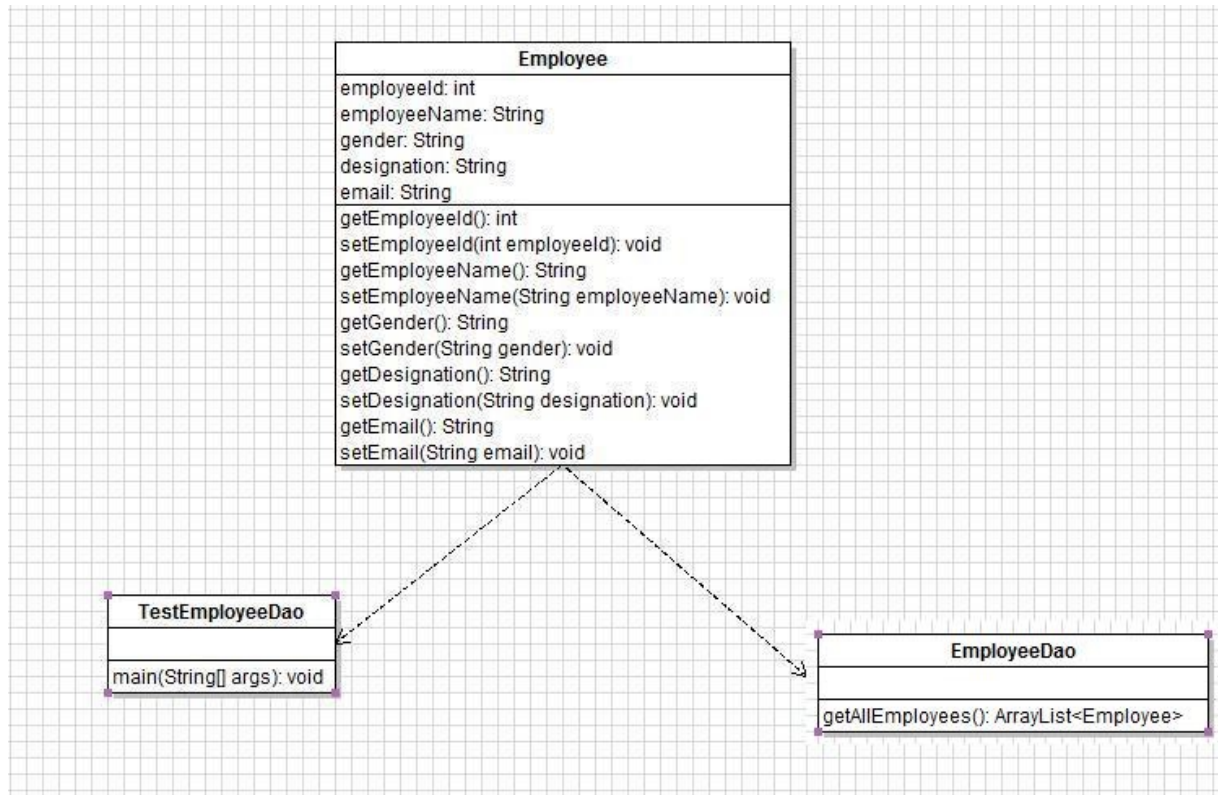
Objective: At the end of the assignments, participants will be able to perform Read operations out of CRUD operations on database tables using JDBC.

- 1) In the class EmployeeDao (*which you must have already created while completing Assignment 1*), create a method getAllEmployees() which would be using JDBC to retrieve the values of all employees that are present in the Employee table.

You also need to use the test class called TestEmployeeDao which would test the method getAllEmployees() of EmployeeDao to retrieve all the records from Employee table and display them on console using the main method.

It is assumed that the table TBL_<Your Employee Id>_EMPLOYEE and the bean Employee.java is already created.

You may refer to the following class diagram:



Solution:

Employee.java

```
package com.tcs.ilp.crud.bean;
public class Employee {
    private int employeeId;
    private String employeeName;
    private String gender;
    private String designation;
    private String email;
    public int getEmployeeId() {
        return employeeId;
    }
    public void setEmployeeId(int employeeId) {
        this.employeeId = employeeId;
    }
    public String getEmployeeName() {
        return employeeName;
    }
    public void setEmployeeName(String employeeName) {
        this.employeeName = employeeName;
    }
    public String getGender() {
        return gender;
    }
    public void setGender(String gender) {
        this.gender = gender;
    }
    public String getDesignation() {
        return designation;
    }
    public void setDesignation(String designation) {
        this.designation = designation;
    }
    public String getEmail() {
        return email;
    }
}
```

EmployeeDao.java

```
package com.tcs.ilp.crud.dao;

import java.sql.Connection;

public class EmployeeDao {
    private String driver = "oracle.jdbc.driver.OracleDriver";
    private String dbURL = "jdbc:oracle:thin:@172.26.132.40:1521:orclilp";
    private String dbUserName = "a88b";
    private String dbPassword = "a88b";
    private Connection con = null;

    public ArrayList<Employee> getAllEmployees() {
        ArrayList<Employee> employeesList = new ArrayList<Employee>();
        try {
            Class.forName(driver);
            con = DriverManager.getConnection(dbURL, dbUserName, dbPassword);
            PreparedStatement pst = con
                .prepareStatement("select * from TBL_387105_EMPLOYEE");

            ResultSet rs = pst.executeQuery();
            while (rs.next()) {
                Employee employee = new Employee();

                int employeeId = rs.getInt("EMPLOYEE_ID");
                String employeeName = rs.getString("EMPLOYEE_NAME");
                String gender = rs.getString("GENDER");
                String designation = rs.getString("DESIGNATION");
                String email = rs.getString("EMAIL");
                employee.setEmployeeId(employeeId);
                employee.setEmployeeName(employeeName);
                employee.setGender(gender);
                employee.setDesignation(designation);
                employee.setEmail(email);

                employeesList.add(employee);
            }
        } catch (Exception e) {
            System.out.println("Exception occurred: " + e);
        } finally {
            if (con != null) {
                try {
                    con.close();
                } catch (SQLException e) {
                    e.printStackTrace();
                }
            }
        }

        return employeesList;
    }
}
```

TestEmployeeDao.java

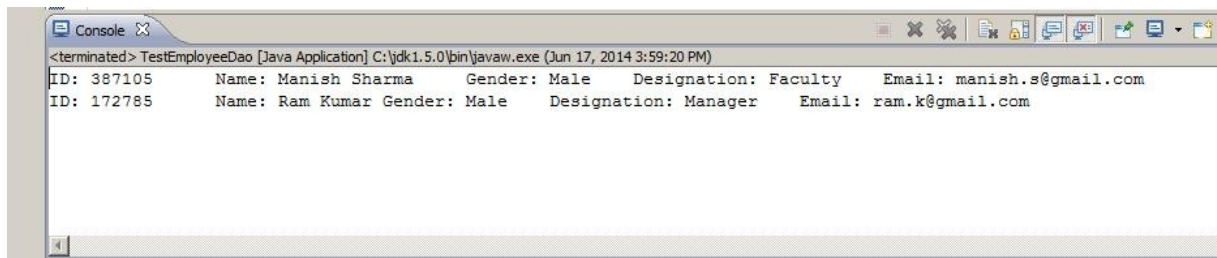
```
import java.util.ArrayList;

import com.tcs.ilp.crud.bean.Employee;
import com.tcs.ilp.crud.dao.EmployeeDao;

public class TestEmployeeDao {
    public static void main(String[] args) {
        EmployeeDao employeeDao = new EmployeeDao();

        ArrayList<Employee> employeesList= employeeDao.getAllEmployees();
        // iterate over the list to print all the records on console
        for (Employee employee : employeesList) {
            System.out.print("ID: "+employee.getEmployeeId());
            System.out.print("\t");
            System.out.print("Name: "+employee.getEmployeeName());
            System.out.print("\t");
            System.out.print("Gender: "+employee.getGender());
            System.out.print("\t");
            System.out.print("Designation: "+employee.getDesignation());
            System.out.print("\t");
            System.out.println("Email: "+employee.getEmail());
        }
    }
}
```

After running this test program, the following screen shows all the records which are there in the table:



```
<terminated> TestEmployeeDao [Java Application] C:\jdk1.5.0\bin\javaw.exe (Jun 17, 2014 3:59:20 PM)
ID: 387105      Name: Manish Sharma      Gender: Male      Designation: Faculty      Email: manish.s@gmail.com
ID: 172785      Name: Ram Kumar      Gender: Male      Designation: Manager      Email: ram.k@gmail.com
```

Each trainee is supposed to write above programs and try out even if they are aware about JDBC. Along with learning JDBC, below points are very important to practice and applicable throughout this ILP training as well as most important for any software code.

- Use exactly same class names as mentioned

- Use exactly same method signature (method name, return type, method parameter type, position of each method parameter)
- Define attributes with same name and data type as given in class outline/diagram.
- Define getter setters as given in the class outline/diagram.
- Ensure attributes are private and other methods which will be called from main method, getter-setter methods and constructor is public.
- Use main method only for input and output and testing object creation and object methods.

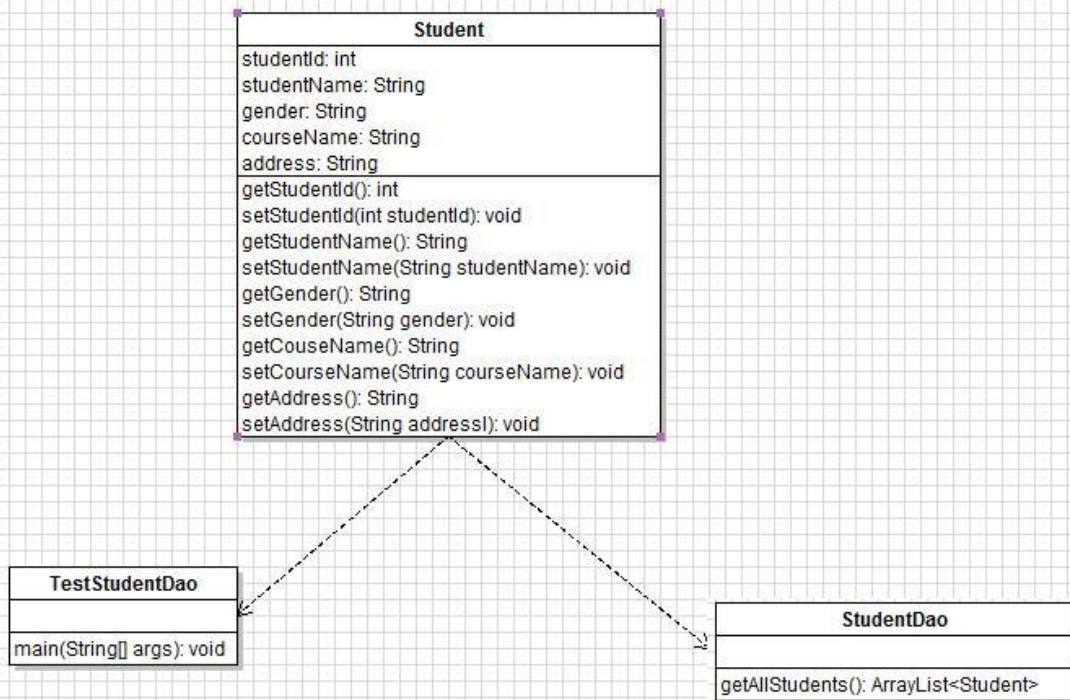
As mentioned above, any logic which may be 100% correct is not valid if above points are not taken care. Hence, simply building logic does not certify us as project ready. Building exact and complete solution does.

2) Create a dao class named StudentDao which would have getAllStudents() method which would be using JDBC to fetch all the records from the Student table.

You also need to use the test class called TestStudentDao which would test the method getAllStudents() of StudentDao to retrieve all the records from Student table and display them on console using the main method.

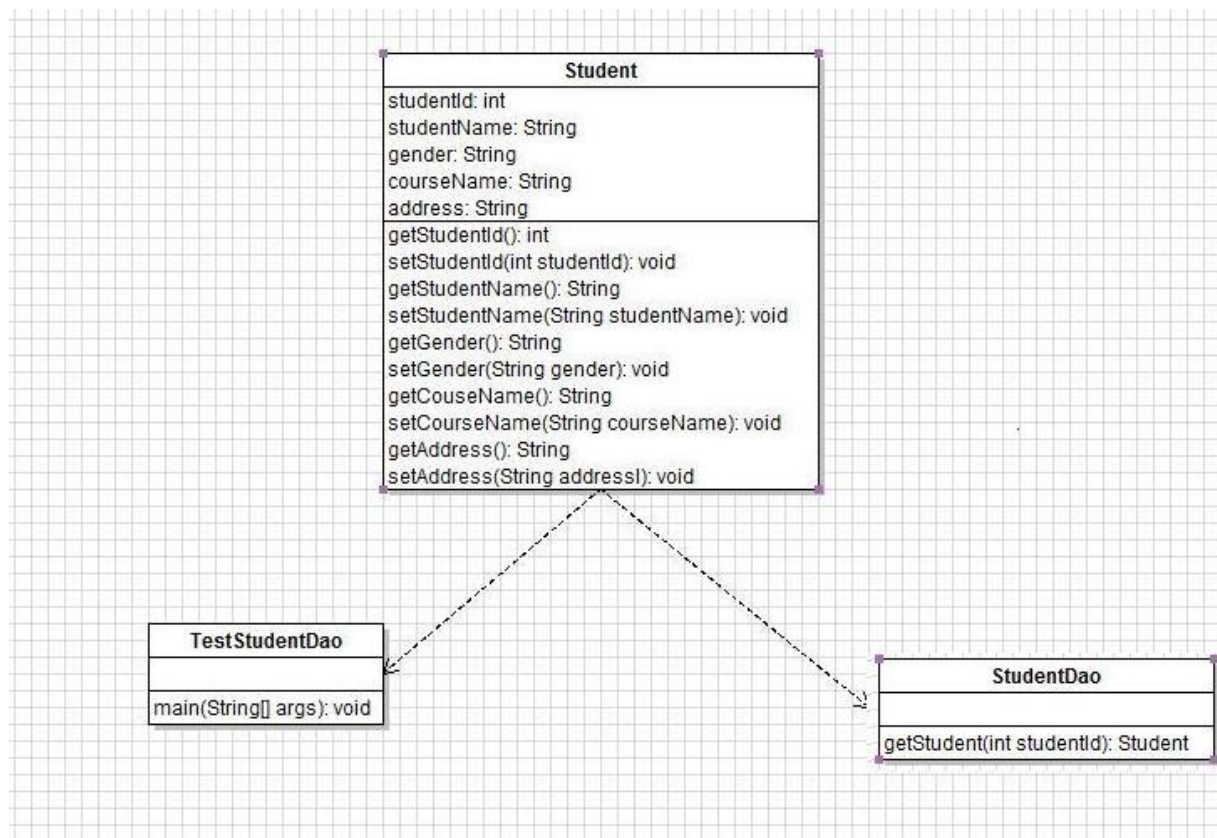
It is assumed that the table TBL_<Your Employee Id>_STUDENT and the bean Student.java is already created.

You may refer to the following class diagram:



- 3) Create a method `getStudent(int studentId)` in the `StudentDao.java` which returns a `Student` object.
- From the test class, call the `getStudent()` method by supplying the id of an existing student. And then display the details of the `Student` object returned by `getStudent()` method.
- Note: You are supposed to use the same database table as you must have used in the previous question.

Please refer to the class diagram below:



JDBC Assignments 3

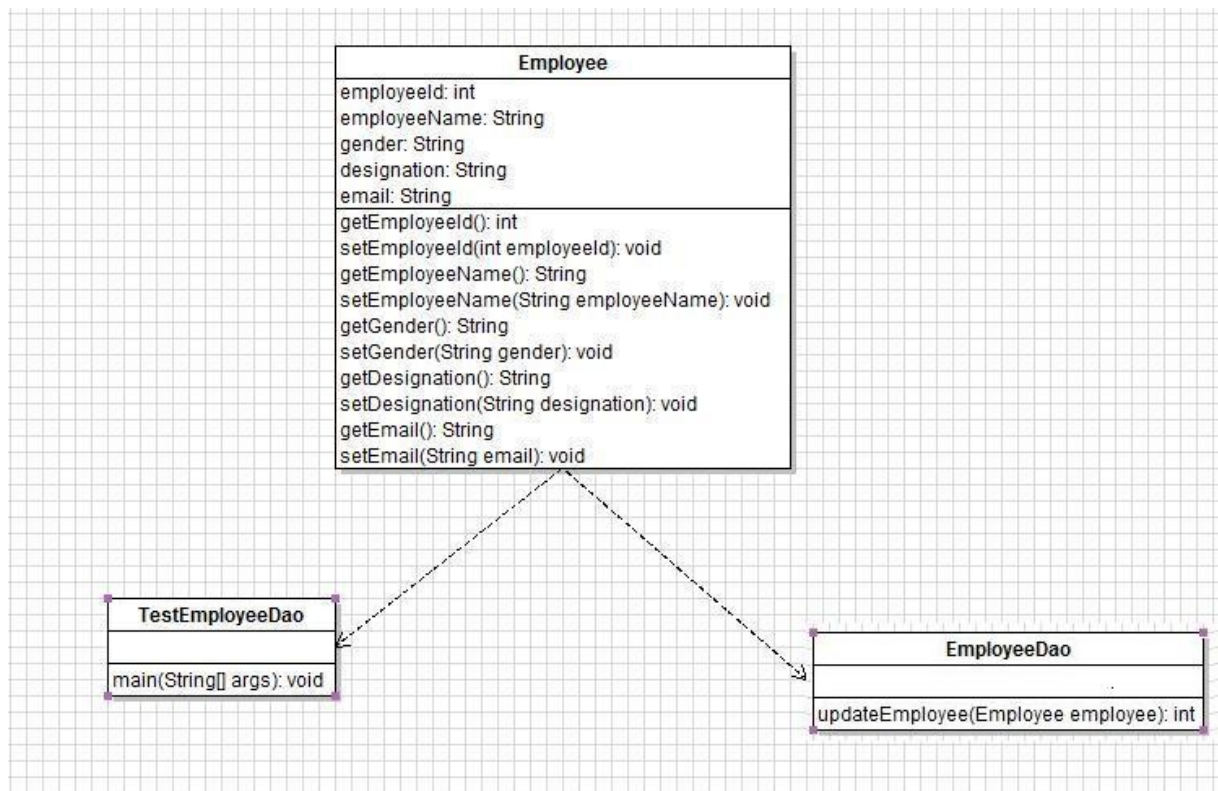
Objective: At the end of the assignments, participants will be able to perform Update operations out of CRUD operations on database tables using JDBC.

- 1) Create a method `updateEmployee(Employee employee)` in the class `EmployeeDao` (which you must have already created while completing Assignment1), which would use JDBC to update the data of an Employee that is present in the Employee table with the values that are supplied to this method as an input parameter in the form of Employee object.

You also need to use the test class called `TestEmployeeDao` which would test the method `updateEmployee()` of `EmployeeDao`.

It is assumed that the table `TBL_<Your Employee Id>_EMPLOYEE` and the bean `Employee.java` is already created.

You may refer to the following class diagram:



Solution:

Employee.java

```
package com.tcs.ilp.crud.bean;
public class Employee {
    private int employeeId;
    private String employeeName;
    private String gender;
    private String designation;
    private String email;
    public int getEmployeeId() {
        return employeeId;
    }
    public void setEmployeeId(int employeeId) {
        this.employeeId = employeeId;
    }
    public String getEmployeeName() {
        return employeeName;
    }
    public void setEmployeeName(String employeeName) {
        this.employeeName = employeeName;
    }
    public String getGender() {
        return gender;
    }
    public void setGender(String gender) {
        this.gender = gender;
    }
    public String getDesignation() {
        return designation;
    }
    public void setDesignation(String designation) {
        this.designation = designation;
    }
    public String getEmail() {
        return email;
    }
}
```

EmployeeDao.java

```

public class EmployeeDao {
    private String driver = "oracle.jdbc.driver.OracleDriver";
    private String dbURL = "jdbc:oracle:thin:@172.26.132.40:1521:orcl1p";
    private String dbUserName = "aSSb";
    private String dbPassword = "aSSb";
    private Connection con = null;

    public int updateEmployee(Employee employee) {
        int numberOfEmployeesUpdated = 0;
        try {
            Class.forName(driver);
            con = DriverManager.getConnection(dbURL, dbUserName, dbPassword);
            PreparedStatement pst = con
                .prepareStatement("UPDATE TBL_387105_EMPLOYEE SET EMPLOYEE_NAME=?, GENDER=?, DESIGNATION=?, EMAIL=?
WHERE EMPLOYEE_ID=?");
            pst.setString(1, employee.getEmployeeName());
            pst.setString(2, employee.getGender());
            pst.setString(3, employee.getDesignation());
            pst.setString(4, employee.getEmail());
            pst.setInt(5, employee.getEmployeeId());
            numberOfEmployeesUpdated = pst.executeUpdate();
            con.commit();
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            if (con != null) {
                try {
                    con.close();
                } catch (SQLException e) {
                    e.printStackTrace();
                }
            }
        }
        return numberOfEmployeesUpdated;
    }
}

```

TestEmployeeDao.java

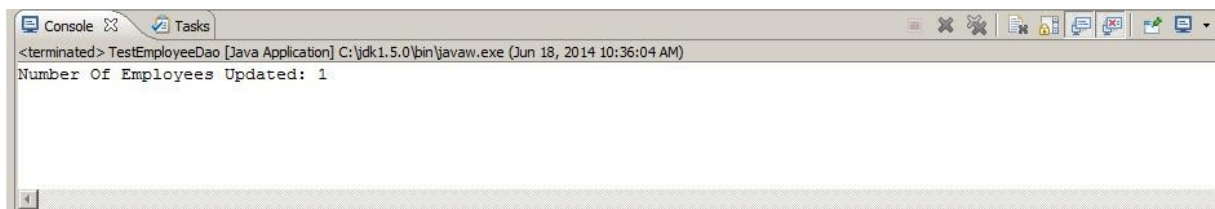
```

public class TestEmployeeDao {
    public static void main(String[] args) {
        Employee employee = new Employee();
        employee.setEmployeeId(172785);
        employee.setEmployeeName("Amrita Singh");
        employee.setGender("Female");
        employee.setDesignation("Manager");
        employee.setEmail("amrita.s@gmail.com");

        EmployeeDao employeeDao = new EmployeeDao();
        int numberOfEmployeesUpdated = employeeDao.updateEmployee(employee);
        System.out.println("Number Of Employees Updated: "+numberOfEmployeesUpdated);
    }
}

```

After running this test program, the following screen shows that one employee is updated in the database table:



Each trainee is supposed to write above programs and try out even if they are aware about JDBC. Along with learning JDBC, below points are very important to practice and applicable throughout this ILP training as well as most important for any software code.

- Use exactly same class names as mentioned
- Use exactly same method signature (method name, return type, method parameter type, position of each method parameter)
- Define attributes with same name and data type as given in class outline/diagram.
- Define getter setters as given in the class outline/diagram.
- Ensure attributes are private and other methods which will be called from main method, getter-setter methods and constructor is public.
- Use main method only for input and output and testing object creation and object methods.

As mentioned above, any logic which may be 100% correct is not valid if above points are not taken care. Hence, simply building logic does not certify us as project ready. Building exact and complete solution does.

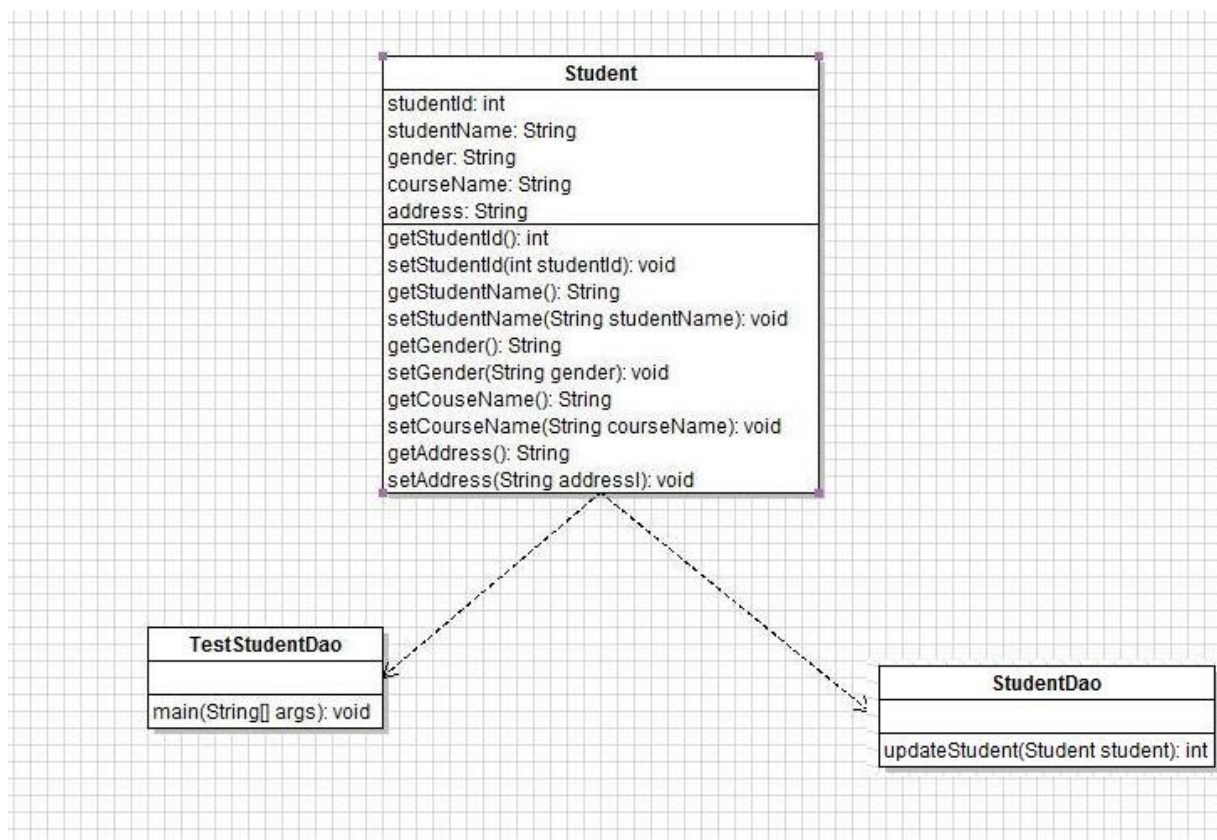
2) Create a method `updateStudent(Student student)` in the `StudentDao.java` (which you must have already created while completing Assignment 1) which would use JDBC to update an existing student record from Student table.

The new values will be supplied through a Student object supplied as a input parameter to this method.

You also need to use the test class called `TestStudentDao` which would test the method `updateStudent(Student student)` of `StudentDao` through the main method to update a record from Student table.

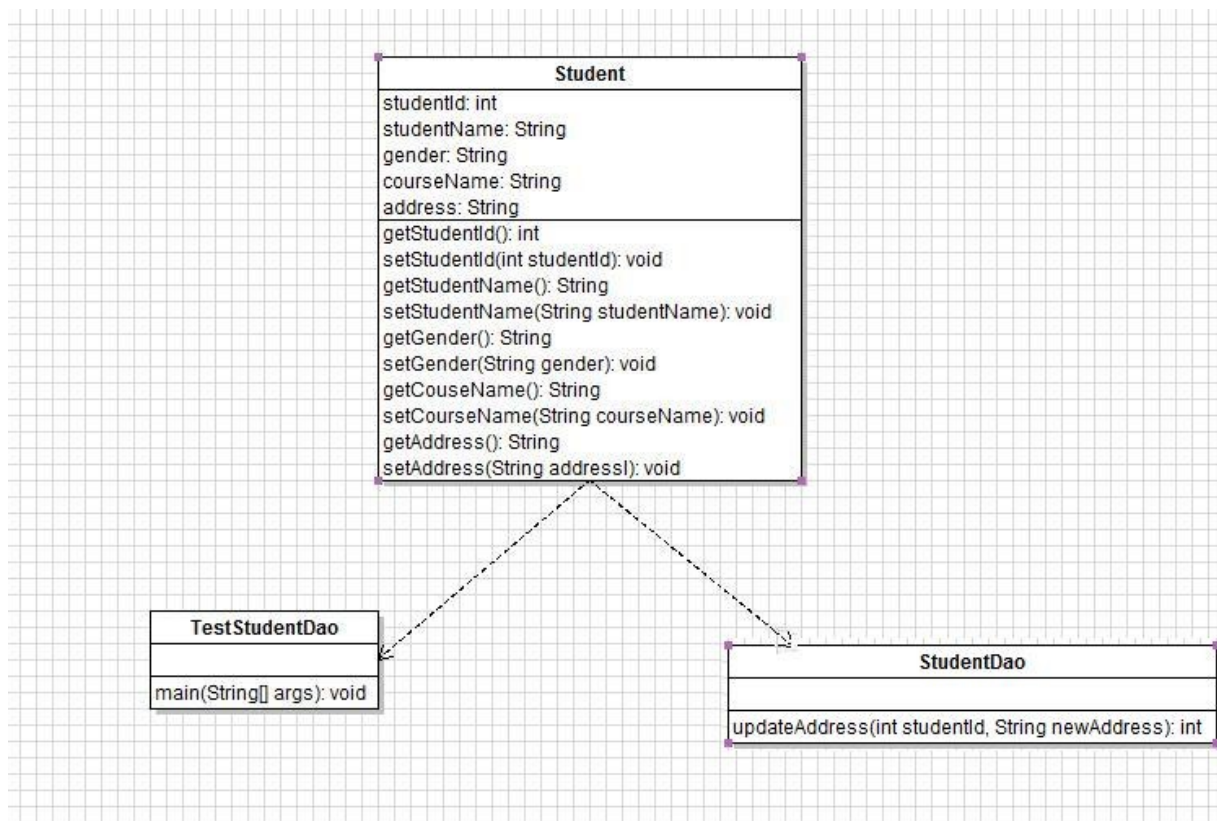
It is assumed that the table `TBL_<Your Employee Id>_STUDENT` and the bean `Student.java` is already created.

You may refer to the following class diagram:



- 3) Create a method `updateAddress(int studentId, String newAddress)` in the `StudentDao.java` which updates the address of the student to the provided new address based on the supplied `studentId`.
From the test class, call the `updateAddress()` method by supplying the id of an existing student, and new address of this student.
Note: You are supposed to use the same database table as you must have used in the previous question.

Please refer to the class diagram below:



JDBC Assignments 4

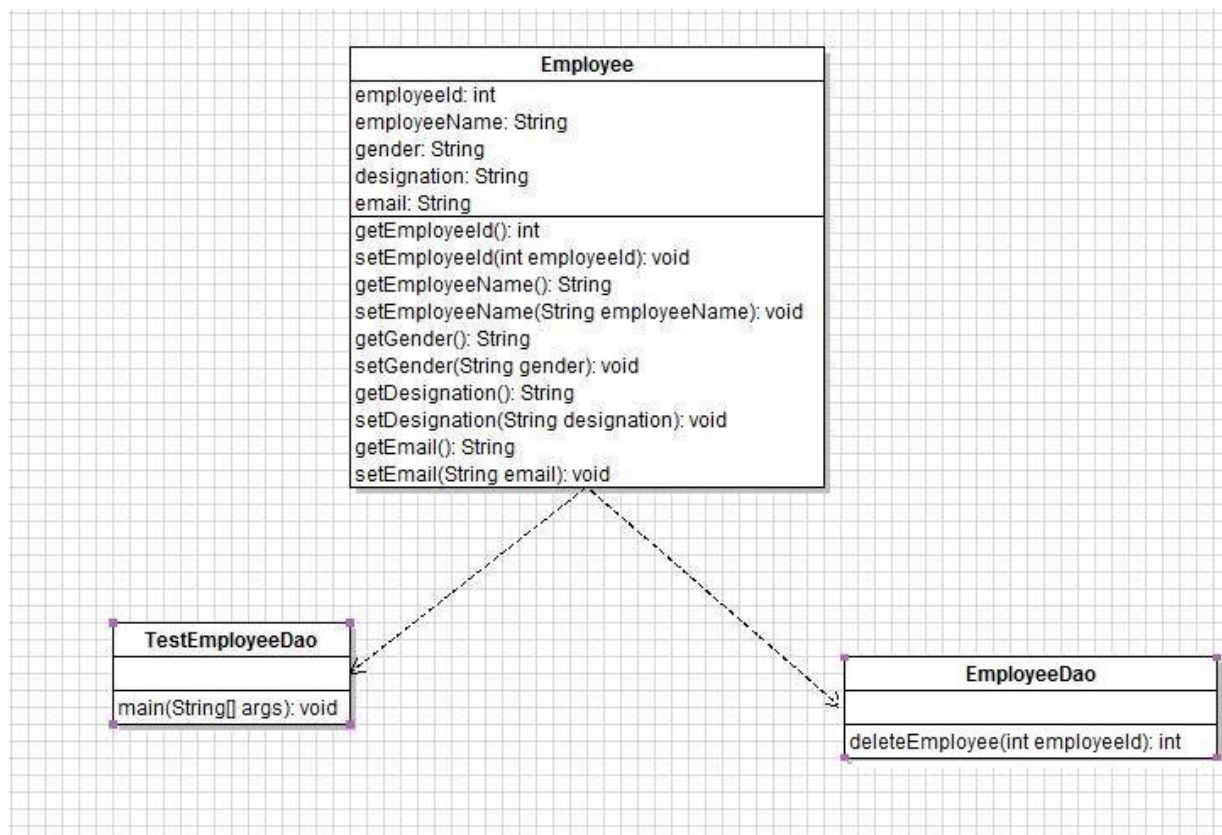
Objective: At the end of the assignments, participants will be able to perform Delete operations out of CRUD operations on database tables using JDBC.

- 1) Create a method `deleteEmployee(int employeeId)` , in the class `EmployeeDao` (*which you must have already created while completing Assignment1*), which would use JDBC to delete the employee record that is present in the Employee table based on the `employeeId` supplied as input parameter.

You also need to use the test class called `TestEmployeeDao` which would test the method `deleteEmployee()` of `EmployeeDao`.

It is assumed that the table `TBL_<Your Employee Id>_EMPLOYEE` and the bean `Employee.java` is already created.

You may refer to the following class diagram:



Solution:

Employee.java

```
package com.tcs.ilp.crud.bean;
public class Employee {
    private int employeeId;
    private String employeeName;
    private String gender;
    private String designation;
    private String email;
    public int getEmployeeId() {
        return employeeId;
    }
    public void setEmployeeId(int employeeId) {
        this.employeeId = employeeId;
    }
    public String getEmployeeName() {
        return employeeName;
    }
    public void setEmployeeName(String employeeName) {
        this.employeeName = employeeName;
    }
    public String getGender() {
        return gender;
    }
    public void setGender(String gender) {
        this.gender = gender;
    }
    public String getDesignation() {
        return designation;
    }
    public void setDesignation(String designation) {
        this.designation = designation;
    }
    public String getEmail() {
        return email;
    }
}
```

EmployeeDao.java

```
public int deleteEmployee(int employeeId)
{
    int numberOfEmployeesDeleted = 0;
    try{
        Class.forName(driver);
        con = DriverManager.getConnection(dbURL, dbUserName, dbPassword);
        PreparedStatement pst = con.prepareStatement("delete from TBL_387105_EMPLOYEE where EMPLOYEE_ID=?");
        pst.setInt(1, employeeId);
        numberOfEmployeesDeleted = pst.executeUpdate();
        con.commit();
    }catch(Exception e){
        System.out.println("Exception occurred: "+e);
    }finally {
        if (con != null) {
            try {
                con.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
    return numberOfEmployeesDeleted;
}
```

TestEmployeeDao.java

```
public class TestEmployeeDao {

    public static void main(String[] args) {
        int employeeId = 172785;
        EmployeeDao employeeDao = new EmployeeDao();
        int numberOfEmployeesDeleted = employeeDao.deleteEmployee(employeeId);
        System.out.println("Number of Employees Deleted: "
            + numberOfEmployeesDeleted);
    }
}
```

After running this test program, the employee with the given employeeId must be deleted from the database.

Each trainee is supposed to write above programs and try out even if they are aware about JDBC. Along with learning JDBC, below points are very important to practice and applicable throughout this ILP training as well as most important for any software code.

- Use exactly same class names as mentioned
- Use exactly same method signature (method name, return type, method parameter type, position of each method parameter)
- Define attributes with same name and data type as given in class outline/diagram.
- Define getter setters as given in the class outline/diagram.
- Ensure attributes are private and other methods which will be called from main method, getter-setter methods and constructor is public.
- Use main method only for input and output and testing object creation and object methods.

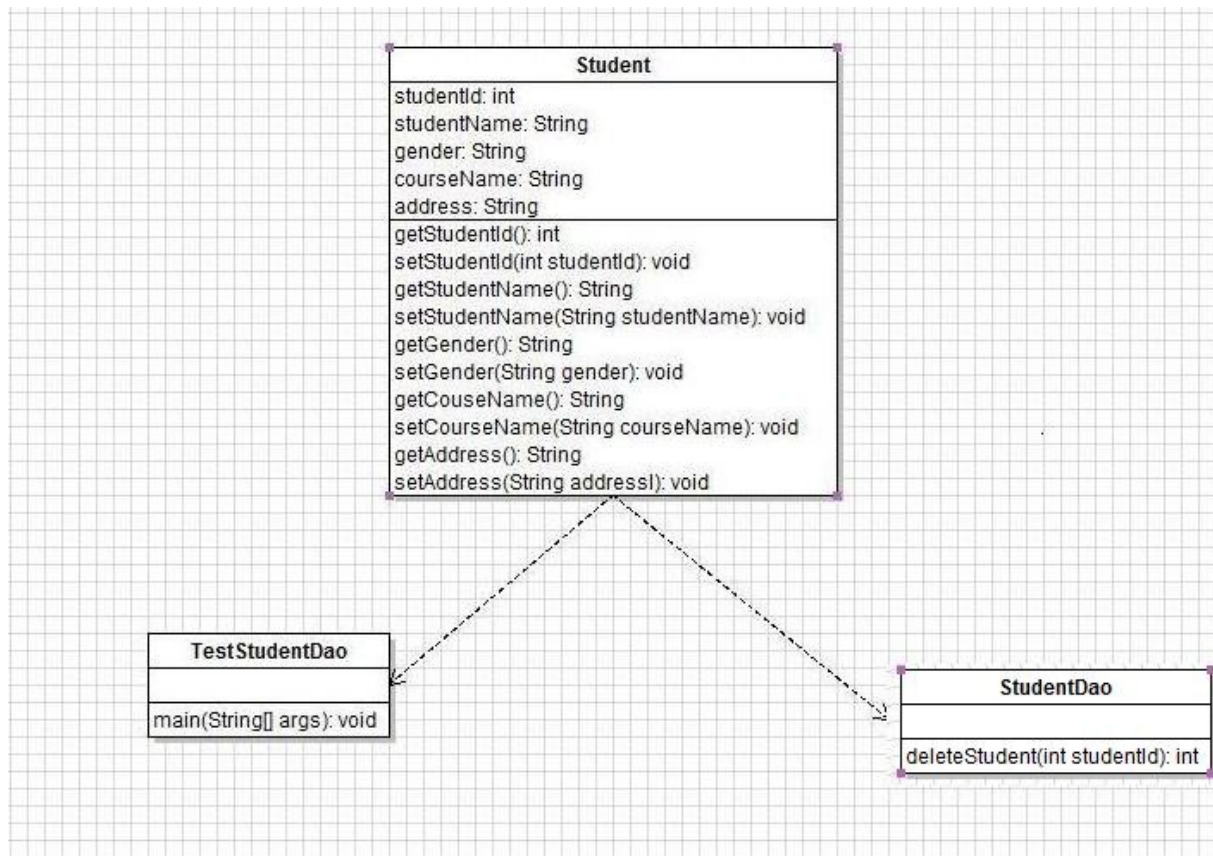
As mentioned above, any logic which may be 100% correct is not valid if above points are not taken care. Hence, simply building logic does not certify us as project ready. Building exact and complete solution does.

2) Create a method `deleteStudent(int studentId)` in the `StudentDao.java` (*which you must have already created while completing Assignment1*) which would use JDBC to delete an existing student record from Student table based on the `studentId` supplied as a input parameter to the method.

You also need to use the test class called `TestStudentDao` which would test the method `deleteStudent(int studentId)` of `StudentDao` through the main method to delete a record from Student table.

It is assumed that the table `TBL_<Your Employee Id>_STUDENT` and the bean `Student.java` is already created.

You may refer to the following class diagram:



- 3) Create a method `deleteStudent(String studentName)` in the `StudentDao.java` which should delete all the students with the given name from the `Student` table.

From the test class, call the `deleteStudent(String studentName)` method by supplying the name of existing student.

Note: You are supposed to use the same database table as you must have used in the previous question.

Please refer to the class diagram below:

