## *Software Systems Components A*

## *Exercise 2: A Database Application for a University*

| | |
|---|---|
| **Deadline** | You must have *submitted your code to Canvas* for this exercise by **Thursday 29th October 2015, at 12:00**<br>No submissions will be accepted after this date, except in cases where we have explicit permission to grant an extension from the welfare team - Students with extensions granted should also notify the course lecturer by email.<br>Vivas will take place in the lab sessions on Thursday 29th & Friday 30th. Vivas will be scheduled by a timetable, which will be published on the module web page.<br>The viva **must** demonstrate your code working on the lab. machines and using the school DB server |
| **Marking scheme** | This exercise is worth 5% of your total mark for SSC.<br>Your mark will be determined by the amount of the exercise you have completed *and* the quality of your solution. |
| **Marking format** | Marks will be awarded by viva and may be altered by supplementary tests, including plagiarism detection, which we perform on your electronic submission. You must submit your code electronically before your viva. You should also fill out a viva form *before* your viva so as not to delay the demonstration. The code that you demonstrate in your viva *must* be the code that you submitted electronically. |
| **Printable sheets** | Viva form is available here [PDF] |

## *Introduction*

For this exercise you are will set up and manipulate a simple data base to record and manipulate information for a University. The data base is a simplified system which does not capture all of the intricacies that would be necessary in a real system.

The exercise has three parts:

1. Completing the design of the database
2. Setting up and populating the data base
3. Creating an interface to the DB using Java's JDBC package. You will use this interface to create a database management program with a number of different functions.

## *Part 1: Data Definition (10%)*

A University has a database that records all of the information about their students including their personal tutors. An outline of a subset of the data base is described below:

- Student (studentID, titleID, foreName, familyName, dateOfBirth)
- Lecturer (lecturerID, titleID, foreName, familyName)

- StudentRegistration(studentID, yearOfStudy, registrationTypeID)
- StudentContact (studentID, eMailAddress, postalAddress)
- NextOfKinContact(studentID, name, eMailAddress, postalAddress)
- LecturerContact(lecturerID, Office, eMailAddress)
- Tutor(studentID, lecturerID)
- Titles (titleID, titleString)
- RegistrationType(registrationTypeID, description)

Where:

- Titles is a table for titles: Professor, Dr, Mr, etc.,
- RegistrationType represents the type of registration (e.g. normal, repeat, external)
- yearOfStudy represents the year of the course that the student is studying (e.g. 1,2,3,4,5)

There are some additional constraints which are not made explicit here and which you will need to make explicit.

#### Part 1.1 (10%)

List the additional constraints which you have identified.

## *Part 2: Creating and populating the Data Base (30%)*

#### Part 2.1 (10%)

For the data base in part 1 you should set up the data base tables. Make sure that you include all the appropriate constraints.

#### Part 2.2 (20%)

Populate the data base with a set of test data that is, at least, adequate to test and demonstrate the functionality in part 3. You should do this using a separate java program which will set up a clean database for use during development and testing.

As a guideline, you should have:

1. At least 5 lecturers
2. At least 5 Titles
3. At least 100 students
4. Sufficient real data to demonstrate the later part of the exercise. For instance, you should have at least 1 lecturer with a realistic set of tutees.

Note: You can do this by:

1. Explicitly creating a small amount of this data
2. Using loops to create the rest as synthetic data eg. Generating students such as Mr Firstname1 Lastname1 …..

***Show your data set to the demonstrator in the viva and be prepared to justify its adequacy.***

## *Part 3: An interface using JDBC (60%)*

Along with its database, the University also needs an interface for its admin team to access the database. Build a Java program (or programs) that will allow a user to perform the tasks below.

Note:

1. This exercise is *not* about the UI. You can produce a sophisticated interface but it is also perfectly okay to have a very simple text based interface with no error checking (infact, see point 2 below)
2. You need to be able to show that your underlying database and java code will detect and correctly handle errors (e.g. inserting a student who is already in the database). So, bear this in mind when you design the UI

#### Part 3.1 (10%)

Register a new student

The operation should check for errors and handle them appropriately.

#### Part 3.2 (10%)

Add a new tutor for a student

You should, again, check for any errors and deal with them appropriately.

#### Part 3.3 (20%)

Produce a report for a student. Your report should list:

- Student title, first and family name
- Date of birth
- StudentID
- Year of study
- Registration type
- Email address and postal address
- Emergency contact: Name, email address and postal address
- Personal tutor

### Part 3.4 (20%)

Produce a report for a lecturer, showing:

- For each year of study:
    - A list of tutees:
        - Title, forename, family name
        - StudentID
        - Registration type
        - Date of Birth
        - Email & Postal address
        - Emergency Contact details

## *Marking scheme*

The mark scheme is documented on the viva form. You should download, print out and fill in the form prior to asking a demonstrator to give you your viva.