Student Management System

by

Fuzzy Logic

Group Details

Team Name	Fuzzy Logic	
Team Members	**Talha Vawda**	218023210
	Luqmaan Haffejee	218008477
	Azhar Mohamed	218006491
	Ahmad Jawaad Shah	218029400
SQL Server Login Details	Username	ist2hw
	Password	ufjufh
Application Login Details	<u>Administrator</u>	
	Username	1234567890
	Password	jkane123
	<u>Student 1 (enrolled 2019; registered)</u>	
	Username	2019000001
	Password	collins98243
	<u>Student 2 (enrolled 2019; not registered)</u>	
	Username	2019360606
	Password	brett22
	<u>Student 3 (enrolled 2017; registered for 2019)</u>	
	Username	2017000001
	Password	yolo1
	<u>Student 4 (enrolled 2017; not registered for 2019)</u>	
	Username	2017000006
	Password	harvarduni4me

<u>Some Notes:</u>
These are login details you can use to test the application for different types of users. However should you wish to test the application with other users, their username (Student/Admin number) and password can be obtained from the Student and Admin tables or you can add (enrol) your own student(s).

When viewing reports/statistics on an admin account, we advise you to view the Computer Science discipline as this is the discipline that we have registered majority of the students for (for testing purposes) and other disciplines may not have any student doing them as yet.
2017 is the first year that we have enrolled students so when viewing a Level 1 module set year to 2017, Level 2: set year to 2018 etc. for optimal results.

Those students that have already registered for 2019 (the current academic year) have their marks for Semester 1 2019 captured already.
We have not registered all students for 2019 so that you can test out the registration using them.

Project Background

Fuzzy Logic has undertaken the task of researching, designing, and developing an Information System for Imperial College that allows them to manage the personal and registration details of their student cohort. We have thus aptly named this project Student Management System.

Fuzzy Logic was approached by Imperial College, a relatively new institution, because they needed a bespoke application tailored for them as the proprietary solutions they had tried to implement did not seem to fulfil their requirements.

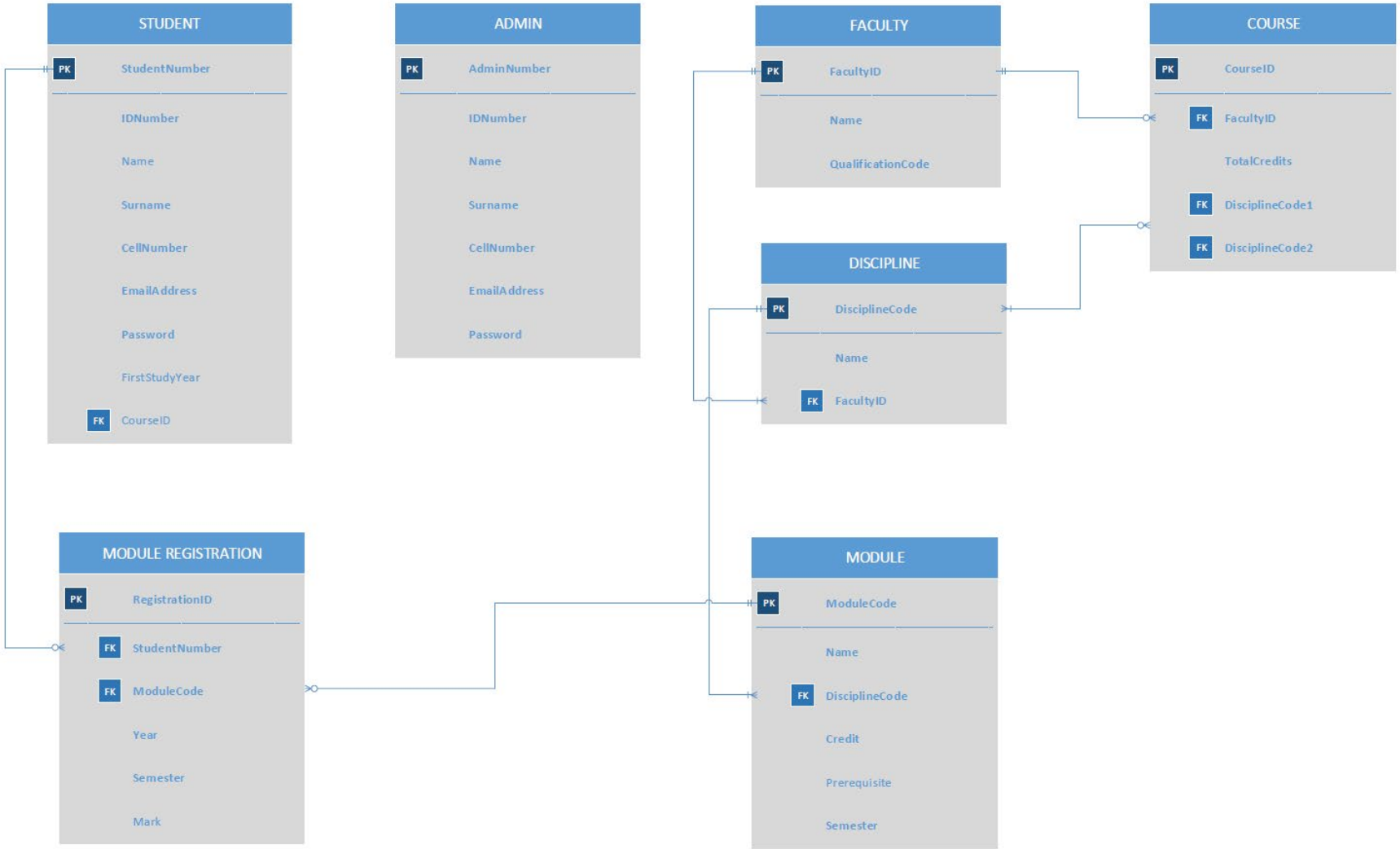
The system was developed using Microsoft Visual Studio 2015 and Microsoft SQL Server Management Studio 2014.

We have created two user types for the application, a student and an administrator, as it is more practical and efficient for each student to enrol themselves onto the system (adding their details and selecting their majors) and registering than for an admin to add each student.

Program Functionality (CRUD)

STUDENT		ADMIN	
Add personal details	Create	Add a students' mark (final mark) on a module that they are registered for but currently has no mark recorded	Create
Select two disciplines as their majors	Create	View their own personal details	Read
Register for both semesters of the current year	Create	View statistics for a particular discipline	Read
View their own personal details	Read	View the Academic Record of a student	Read
View their current and past registrations	Read	View the current and past registrations of a student	Read
View their marks for the modules that they have registered for and completed (Academic Record)	Read	Change some of their personal details	Update
Change some of their personal details	Update	Change their password	Update
Change their password	Update	Delete the current registration of a student	Delete

Entity Relationship Diagram



*Please see the PDF version in the ERD folder for a higher resolution image

SQL Statements

a) An SQL statement that extracts and displays all data from a database table

```
SELECT Name FROM FACULTY
```

Display all the Faculties to a student when they are enrolling so that they can select which faculty they want to go into and from there they can select 2 Disciplines as their majors

b) An SQL statement that will update values of a table

```
UPDATE     STUDENT
SET         Password = @Password
WHERE     (StudentNumber = @StudentNumber) AND (IDNumber = @IDNumber)
AND        (CellNumber = @CellNumber)
```

Allow the student to change (update) their password

c) At least three SQL statements with a WHERE clause

1. Get all Student Numbers' of the students that have passed a particular module

```
SELECT StudentNumber, ModuleCode, Year, Semester, Mark, RegistrationID
FROM [MODULE REGISTRATION]
WHERE StudentNumber = @StudNum AND Mark >= 50
```

The administrator can use this for administrative purposes

2. Add the final mark for a particular module for a particular student

```
UPDATE     [MODULE REGISTRATION]
SET         Mark = @Mark
WHERE     (StudentNumber = @StudentNumber) AND (Year = @Year)
AND        (ModuleCode = @ModuleCode)
```

It is the administrator's job to enter students' final marks. This SQL query which is part of the code will add (update; -1 represents no mark) the mark for a student for a module they did in a particular year.

3. Get the course that a student has registered for:

```
SELECT CourseID  FROM STUDENT WHERE  (StudentNumber = @StudentNumber)
```

This will be used to get the 2 majors of the student and the qualification type which will be displayed in the student's details tab and also on their academic record

4. Get all modules belonging to a specific discipline

```
SELECT      ModuleCode, Name, DisciplineCode, Credit, Prerequisite, Semester
FROM          MODULE
WHERE      (DisciplineCode = @Disc)
```

This is used to display all the modules in a combo box based on a discipline selected so that the administrator can select a module to view statistics and analytics on that module

d) At least two SQL subqueries

1. Get available Semester 1 modules for a specific year based on completion of prerequisites

```
SELECT      ModuleCode, Name, DisciplineCode, Credit, Prerequisite, Semester
FROM        MODULE
WHERE       (Prerequisite IN (SELECT ModuleCode FROM [MODULE REGISTRATION]
WHERE       (StudentNumber = @StudNo) AND (Mark >= 50) AND (Year = @Year)))
AND         (DisciplineCode = @Major1 OR DisciplineCode = @Major2)
AND         (Semester = 1)
```

Display all the modules a student can register for semester 1 of the year so that they can register for that semester. This is done by selecting all the modules that are part of their majors and filtering out all the modules they cannot do (they can only do a module if they passed the prerequisite module for that module [First-year modules have no prerequisite])

2. Get available Semester 2 modules for a specific year based on completion of prerequisites

```
SELECT      ModuleCode, Name, DisciplineCode, Credit, Prerequisite, Semester
FROM        MODULE
WHERE       (Prerequisite IN (SELECT ModuleCode FROM [MODULE REGISTRATION]
WHERE       (StudentNumber = @StudNo) AND (Mark >= 50) AND (Year = @Year)))
AND         (DisciplineCode = @Major1 OR DisciplineCode = @Major2)
AND         (Semester = 2)
```

Display all the modules a student can register for semester 2 of the year so that they can register for that semester. This is done by selecting all the modules that are part of their majors and filtering out all the modules they cannot do (they can only do a module if they passed the prerequisite module for that module [First-year modules have no prerequisite])

e) **At least two aggregation queries**

1. Get number of student registered for a module

```
SELECT      COUNT(*) AS TotalStudents
FROM        [MODULE REGISTRATION]
WHERE       (ModuleCode = @Mod) AND (Year = @Year)
```

The administrator can view the number of students registered for a particular module which they can use to determine the appropriate classroom the have the lectures for that module in, and also to plan seating arrangements for exams

2. Get the highest mark for a specific module in a specific year

```
SELECT      MAX(Mark) AS Highest
FROM        [MODULE REGISTRATION]
WHERE       (ModuleCode = @Mod) AND (Year = @Year) AND (Mark <> - 1)
```

The administrator can use the highest mark for a module to evaluate performance of that module and to award Certificate of Merit to the student with the highest mark