**Module Code & Module Title**

**CC5051NA Databases System**

**50% Individual Coursework**

**Year and Semester**

**2020-21 Autumn**

**Student Name: Srijeet Sthapit**

**Group:L1C9**

**London Met ID:19031714**

**College ID:190218**

**Assignment Due Date:20th December,2020**

**Assignment Submission Date:20th December,2020**

## Contents

## 1. Introduction

### 1.1 Introduction to Islington College

Islington College, formerly known as Informatics College has been providing high-caliber human capital to the IT and Business Industry since 1996.Partnered with London Metropolitan University located in London, UK the most sought undergraduate and postgraduate programs combined with current industry practices are  available for students to compete  in employment market. The college has always been focused on driving towards quality education and inspiring students to learn and excel, and finally prepare them for the harsh competition. Along with being focused on studies, the college has excellent infrastructures, basketball ground, canteens, libraries and lots of places to hang out with friends. The college hosts various sports games such as basketball, football and programs during Christmas, Holi that creates an environment for students to socialize and improve their aspect of life other than their studies. The aim of Islington College as stated before is to provide top quality education along with a suitable environment for students to grow into a competitive individual. The objective taken into consideration to achieve the goal are as follows:

1. Develop self-esteem and feeling of self-worth.
2. Make them take responsibility of their actions.
3. Provide them with teams to improve their teamwork and communication skills.
4. Teach them to apply knowledge and skills to solve problems.

## 1.2 Current Business Activities and Organizations

1. The college records details of students and instructors in a different database along with their respective addresses.
2. College provides 3 courses: BIT and BBA and MBA.

| | Courses | | |
|---|---|---|---|
| Specification | BIT | BBA | MBA |
| | Computing | Accounting and Finance(I) | Accounting and Finance(II) |
| | Multimedia | Marketing | |
| | Network and Security | | |

Each specification has 4 modules.

3. Each student can apply for a specification that lies under one of the mentioned courses.
4. Each specification consists of either 3 or 2 modules and a single module can also belong to several specifications. For example, Java can be taught in all specifications of BIT.
5. The fees of students is based on the specification he chooses. Also, each specification has different enrollment dates.
6. Modules are taught in class which has a unique identifier class ID along with its name.
7. Instructors must be associated in a course only and teach the modules that lie under the course.

### 1.3 Business Rules

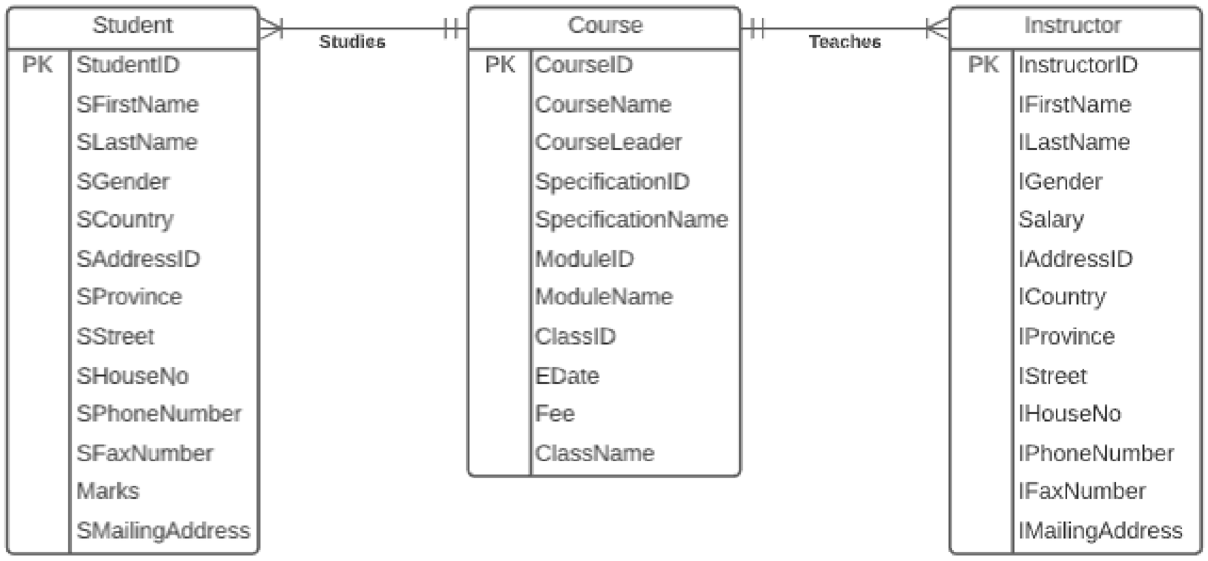
1. Records of students and instructors are stored in a separate database. Each of these records can contain one or more addresses out of which one is a mailing address.
2. Address of any individual should have country, province, city, street and house number as their attributes.
3. A person must have at least one of their phone numbers stored in the database whereas a person can have zero or multiple fax numbers.
4. A course can have many specifications but a specification belongs to one course.
5. A specification can have many modules and a single module can also belong to several specifications.
6. A student can enroll for a course and a course can have many students.
7. A student belongs to one specification and a specification has many students.
8. An instructor belongs to a course and a course can have many instructors.
9. Each course has a course leader and an instructor can be course leader of only one course.
10. Course leader are the module leader of all the module that lies in the course. Course leader monitors all the modules and teaches only one module.
11. An instructor can be associated in different modules and each module can be taught by multiple instructors.
12. A module is taught in any particular class but multiple modules are taught in each class.

### 1.4 Creation of Entities and Attributes

Entity can be a real-world object, either animate or inanimate, that is easily identifiable and attributes are properties that properly defines an entity (Tutorialspoint, 2020). In simple words, entity is an item that exists such as thing, person, place or object and attributes define the entity. In case of university/colleges, the entities may be Students, Courses and Instructors. Similarly properties such as name, age, course taken are attributes of students.

| **Entity** | **Attributes** |
|---|---|
| Student | **StudentID**, SFirstName, SLastName, SGender, SAddressID, SCountry, SProvince, SStreet, SHouseNo, SMailingAddress, SPhoneNumber, SFaxNumber, Marks |
| Course | **CourseID**, CourseName, CourseLeader, SpecificationID, SpecificationName, EDate, Fee, ModuleID, ModuleName, ClassID, ClassName |
| Instructor | **InstructorID**, IFirstName, ILastName, IGender, Salary, IAddressID, ICountry, IProvince, IStreet, IHouseNo, IMailingAddress, IPhoneNumber, IFaxNumber |

### 1.5 Initial ER Diagram



# 2. Database Design

## 2.1 Assumptions

1. The marks of students are provided as an average of all the modules they have taken.
2. Enrollment Date of different specifications are on different dates.
3. Fees are based on the specification the student chooses.
4. Salaries of teachers are fixed when contracts are signed.
5. InstructorID is a combination of string and numbers. For example I100 represents that the instructor is course leader of BIT and I101 represents that the instructor belongs to BIT and is not a course leader.
6. Address ID of students begins with A100 and that of instructor starts with A200.
7. Each specification has different enrollment dates.

## 2.2 Normalization

Normalization is a technique of organizing redundant data and improving data integrity in the database. Normalization is used to minimize the redundancy from a relation or set of relations and eliminate undesirable characteristics like Insertion, Update and Delete Anomalies. (JavaTPoint, 2020)

### 2.2.1 UNF

**Scenario**

1. Each course has its id and name.
2. Each course has multiple specification and each specification has multiple modules.
3. Each specification and module also has its own id and name as well.
4. Each module is taught in a class.
5. Each student and teacher has their personal details stored and either one or multiple address details stored.
6. Each address can contain multiple phone numbers and none or more than one fax numbers.
7. Each same details such as firstname and lastname has either I (instructor) or S (Student).

8. MailingAddress is stored as yes or no and denotes if that addressed is mailing address or not.
9. House number will be unique with in a province but may repeat in other provinces.

**Showing Repeating Groups**

**Course** (**<u>CourseID</u>**, CourseName, CourseLeader, {SpecificationID, SpecificationName, EDate, Fee, {ModuleID, ModuleName, ClassID, ClassName}},
{StudentID, SFirstName, SLastName, SGender, {SAddressID, SCountry, SProvince, SStreet, SHouseNo, SMailingAddress
{SPhoneNumber},
{SFaxNumber}}},
{InstructorID, IFirstName, ILastName, IGender, Salary, {IAddressID, ICountry, IProvince, IStreet, IHouseNo, IMailingAddress
{IPhoneNumber},
{IFaxNumber}}},
)

### 2.2.2 1NF

A relation is in 1NF if it contains an atomic value. (JavaTPoint, 2020). There should be no repeated group of attributes and each and each record needs to be unique.

**Action Carried out**

Repeated groups has been separated into different tables with primary keys and linked with foreign keys.

**Entitites in 1NF:**

**Course** -1(<u>CourseID</u> , CourseName, CourseLeader)

**Specification** -1(<u>SpecificationID</u>, SpecificationName, EDate, Fee, <u>CourseID*</u>)

**Module** -1 (<u>ModuleID</u>, ModuleName, ClassID, <u>SpecificationID*</u>)

**Student** -1 (<u>StudentID</u>, SFirstName, SLastName, SGender, Marks, <u>CourseID*</u>, <u>SpecificationID*</u>)

**SAddress** -1 (SAddressID, SCountry, SProvince, SStreet, SHouseNo, SMailingAddress, StudentID*)

**SContactInfo** -1 ( SPhoneNumber*, SAddressID*)

**SFaxInfo** -1 ( SFaxNumber*, SAddressID*)

**Instructor** -1(InstructorID, IFirstName, ILastName, IGender, Salary, CourseID*, ModuleID*)

**IAddress** -1 (IAddressID, ICountry, IProvince, IStreet, IHouseNo, IMailingAddress, InstructorID*)

**IContactInfo** -1 ( IPhoneNumber,IAddressID*)

**IFaxInfo -**1 ( IFaxNumber, IAddressID*)

### 2.2.3 2NF

A relation will be in 2NF if it is in 1NF and all non-key attributes are fully functional dependent on the primary key. (JavaTPoint, 2020). In other words the non-key attributes in the table must be dependent on knowing all of the primary or composite key.

Primary key is a single column value that uniquely identifies a database record. They cannot be null and must be unique.

Similarly composite key is a primary key composed of multiple columns used to identify a record uniquely.

Foreign keys connect different tables with reference to primary key of another table. They do not have to be unique and can be null as well.

**Actions Carried out**

The tables with one attribute as unique keys cannot have partial dependency. So the table Course is already in 2NF. Also since SContactInfo, SFaxInfo, IContactInfo and IFaxInfo have no non keys, they are also already in 2NF. So, the second normalization of remaining tables are:

**Specification**

SpecificationID >> SpecificationName, EDate, Fee

(Partial Dependency exists so this needs to be in a separate table)

CourseID >>

(No attribute exist for CourseID so no need to have this as new Relation)

SpecificationID, CourseID >>

(No attribute exist for this Composite Unique Identifier but as this is from the existing Relation, it need to be included)

**Module**

ModuleID >> ModuleName

(Partial Dependency exists so this needs to be in a separate table)

SpecificationID >>

(No attribute exist for SpecificationID so no need to have this as new Relation)

ModuleID, SpecificationID >> ClassID, ClassName

(Partial Dependency exists so this needs to be in a separate table)

**Student**

StudentID, CourseID, SpecificationID >>

(No attribute exist for this Composite Unique Identifier but as this is from the existing Relation, it need to be included)

StudentID >> SFirstName, SLastName, SDob, SGender, Marks

(Partial Dependency exists so this needs to be in a separate table)

CourseID >>

(No attribute exist for CourseID so no need to have this as new Relation)

SpecificationID >>

(No attribute exist for SpecificationID so no need to have this as new Relation)

StudentID, CourseID >>

(No attribute exist for this Composite Unique Identifier so no need to have this as new Relation)

StudentID, SpecificationID >>

(No attribute exist for this Composite Unique Identifier so no need to have this as new Relation)

CourseID, SpecificationID >>

(No attribute exist for this Composite Unique Identifier so no need to have this as new Relation)

**SAddress**

SAddressID >>SCountry, SProvince, SStreet, SHouseNo, SMailingAddress

(Partial Dependency exists so this needs to be in a separate table)

StudentID >>

(No attribute exist for StudentID so no need to have this as new Relation)

SAddressID, StudentID >>

(No attribute exist for this Composite Unique Identifier but as this is from the existing Relation, it need to be included)

**Instructor**

InstructorID, CourseID, ModuleID >>

(No attribute exist for this Composite Unique Identifier but as this is from the existing Relation, it need to be included)

InstructorID>> IFirstName, ILastName, IDob, IGender, Salary

(Partial Dependency exists so this needs to be in a separate table)

CourseID >>

(No attribute exist for CourseID so no need to have this as new Relation)

ModuleID >>

(No attribute exist for ModuleID so no need to have this as new Relation)

InstructorID, CourseID >>

(No attribute exist for this Composite Unique Identifier so no need to have this as new Relation)

InstructorID, ModuleID >>

(No attribute exist for this Composite Unique Identifier so no need to have this as new Relation)

CourseID, ModuleID >>

(No attribute exist for this Composite Unique Identifier so no need to have this as new Relation)

**IAddress**

IAddressID >>ICountry, IProvince, IStreet, ISHouseNo, IMailingAddress

(Partial Dependency exists so this needs to be in a separate table)

InstructorID>>

(No attribute exist for InstructorID so no need to have this as new Relation)

IAddressID, InstructorID>>

(No attribute exist for this Composite Unique Identifier but as this is from the existing Relation, it need to be included)

**Final 2NF**

**Course -2** (CourseID , CourseName, CourseLeader)
**Specification -2** (SpecificationID, SpecificationName, EDate, Fee)
**SpecificationList -2** (SpecificationID*, CourseID*)
**Module -2** (ModuleID, ModuleName)
**ModuleList -2** (ModuleID*, SpecificationID*, ClassID, ClassName)
**Student -2** (StudentID, SFirstName, SLastName, SGender, Marks)
**StudentDetails -2** (StudentID*, CourseID*, SpecificationID*)
**Instructor -2** (InstructorID, IFirstName, ILastName, IGender, Salary)
**InstructorDetails -2** (InstructorID*, CourseID*, ModuleID*)
**SAddress -2** (SAddressID, SCountry, SProvince, SStreet, SHouseNo, SMailingAddress)
**SAddressDetails -2** (SAddressID*, StudentID*)
**IAddress -2** (IAddressID, ICountry, IProvince, IStreet, ISHouseNo, IMailingAddress)
**IAddressDetails -2** (IAddressID*, InstructorID*)
**SContactInfo -2** (SPhoneNumber,  SAddressID*)
**SFaxInfo -2** (SFaxNumber, SAddressID*)
**IContactInfo -2** (IPhoneNumber, I AddressID*)
**IFaxInfo -2** (IFaxNumber,  IAddressID*)

### 2.2.4 3NF

A relation will be in 3NF if it is in 2NF and no transition dependency exists. (JavaTPoint, 2020).Transitive dependency occurs if a non-key of a table depends on another Non-key of the same table.

**Actions Carried Out**

The tables with 1 or less than one non key cannot have transitive dependency. So the tables that are already in 3NF are:

1. SpecificationList
2. Module
3. StudentDetails
4. InstructorDetails
5. SAddressDetails
6. IAddressDetails
7. SContactInfo
8. SFaxInfo
9. IContactInfo
10. IFaxInfo

**Third normalization of remaining entities are as follows:**

**Course -2** (CourseID , CourseName, CourseLeader)

CourseID >> CourseName

CourseName >> CourseLeader

(Transitive dependency exists so different tables must be made for both relations)

**Specification -2** (SpecificationID, SpecificationName, EnrollmentDate, Fee)

SpecificationID >> SpecificationName

SpecificationName >> EDate, Fee

(Transitive dependency exists so different tables must be made for both relations)

**ModuleList -2** (ModuleID*, SpecificationID*, ClassID, ClassName)

ModuleID,SpecificationID >> ClassID

ClassID >> ClassName

(Transitive dependency exists so different tables must be made for both relations)

**Student -2** (StudentID, SFirstName, SLastName, SGender, Marks)

StudentID >> SFirstName, SLastName, SDob, SGender, Marks

(Transitive dependency does not exist)

**Instructor -2** (InstructorID, IFirstName, ILastName, IGender, Salary)

InstructorID >>IFirstName, ILastName, IDob, IGender, Salary

(Transitive dependency does not exist)

**SAddress -2** (SAddressID, SCountry, SProvince, SStreet, SHouseNo, SMailingAddress)

SAddressID >> SProvince, SStreet, SHouseNo, SMailingAddress

SProvince >> SCountry

(Transitive dependency exists so different tables must be made for both relations)

**IAddress -2** (IAddressID, ICountry, IProvince, IStreet, ISHouseNo, IMailingAddress)

IAddressID >> IProvince, IStreet, IHouseNo, IMailingAddress

IProvince >> ICountry

(Transitive dependency exists so different tables must be made for both relations)

**Final 3NF**

**Course -3** ( CourseID , CourseName*)

**Leader -3** (CourseName, CourseLeader)

**Specification -3** (SpecificationID, SpecificationName*)

**SpecificationDetails -3** (SpecificationName, EDate, Fee)

**SpecificationList -3** (SpecificationID*, CourseID*)

**Module -3** (ModuleID, ModuleName)

**ModuleList -3** (ModuleID*, SpecificationID*, ClassID*)

**Class -3** (ClassID, ClassName)

**Student -3** (StudentID, SFirstName, SLastName, SDob, SGender, Marks)

**StudentDetails 3**(StudentID*, CourseID*, SpecificationID*)

**SAddress -3** (SAddressID, SProvince*, SStreet, SHouseNo, SMailingAddress)

**SCountry -3** (SProvince, SCountry)

**SAddressDetails -3** (SAddressID*, StudentID*)

**SContactInfo -3** (SPhoneNumber, SAddressID*)

**SFaxInfo -3** (SFaxNumber, SAddressID*)

**Instructor -3** (InstructorID, IFirstName, ILastName, IDob, IGender, Salary)

**InstructorDetails -3** (InstructorID*, CourseID*, ModuleID*)

**IAddress -3** (IAddressID, IProvince*, IStreet, ISHouseNo, IMailingAddress)

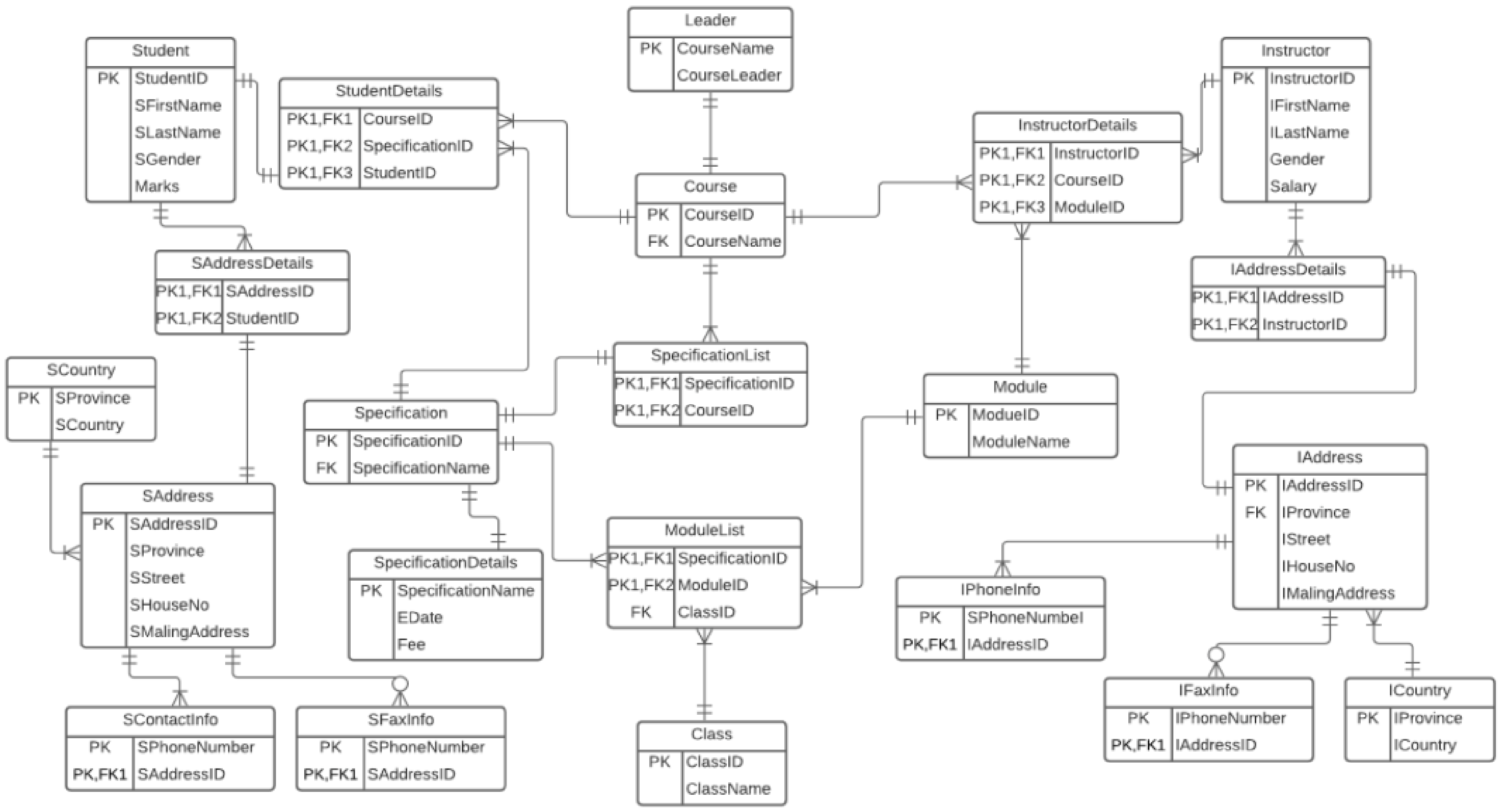
**ICountry -3** (IProvince, ICountry)

**IAddressDetails -3** (IAddressID*, InstructorID*)

**IContactInfo -3** (IPhoneNumber,I AddressID*)

**IFaxInfo -3** (IFaxNumber,  IAddressID*)

### 2.3 ER Diagram after Normalization



## 3. Implementation

### 3.1 Creation of Tables

### 3.2 Populating data into tables

## 4. Information and Transaction Queries

## 5. Conclusion