CSC2001F, 2020 - Assignment 5 - DATABASES

CHTTEN011; MFNLIN003; RGNVER001; WNGJIA001

## Additional Information

Besides the students matric and first year final year marks we have also created attributes to store students' medical conditions, whether they get financial aid and their accommodation situation as we feel that these factors can contribute to a student's academic performance. Some of the relations and their proposed attributes before refinement are listed below:

**Student:**

1.StudentNumber(var)- Primary key

2.Degree Code-foreign key

3. Name(text)

4. Surname(text)

5.Gender

6.Nationality(var)

7.IdentityNumber/PassportNumber(var)

8.Address(text)

9.On Campus Accommodation(boolean)

10.GPA for 1st year university

12.GPA for matric

13.1st Year Results

14.Fee Payer(Financial Aid or guardian pays or self-paying by working)

**School Leaving Results:**

1.StudentNumber(var)- foreign key

2.Subject name

3.Final Mark

**1st Year Results:**

1.StudentNumber(var)- foreign key

2.Course Code- foreign key

3.Final mark

**Lecturer:**

1.Lecturer Name(text)

2.Highest Level of Education(text)

3.Faculty(text)

4.Course(text)

5.Course Code(var)-foreign key

6.Pre-Service(var)

7.Years of Experience(int)

8.Achivements(text)

9.Attendance(int)

10.Staff Number(var)

**Faculty:**

1.Faculty Name-primary key

2.DegreeCode-foreign key

3.Number of Employees

**Degree:**

1.Degree Code-primary key

2.Course Code-foreign key

3.Faculty Name-foreign key

4.Degree Name

5.1st Year Pass Rate

**Course:**

1.Course Code-primary key

2.Course Name

3.Number of registered students

4.Lecture Recorded(boolean)

5.DegreeCode

6.First year pass rate

**Residence Status:**

1.res name

2.Location

3.Population

4.Library(Boolean)

5.ComputerLab(Boolean)

## ER Model

Person

Student

Course

Requires

Lecturer

Lectures

Takes

Degree

Enrols

Has

Has

First Year Results

School-leaving Results

## Assumptions & Motivations

Degree.First Year Pass Rate was assumed to get overwritten every year.

Initially the design included a Faculty entity, but we decided to remove it since it doesn’t store much necessary information in our case.

An ISA relationship was used between Person and Student/Lecturer but was later replaced with the use of PeopleSoft ID as a foreign key in the Student/Lecturer entity for a better design.

## Relationships

A student must have a school-leaving result / A matric final result must be had by 1 student (Weak Entity Set)

A student must have a first-year result / A first-year result must be had by 1 student (Weak Entity Set)

A student must enrol for 1…N degrees / A degree can have 0…N students

A degree must require 1…N courses / A course can be required by 0…N degrees.

A student must take 1…N courses / A course can be taken by 0…N students

A lecturer can lecture 0…N courses / A course must be taught by 1…N lecturers

## a) Relation Schema

Primary keys are underlined, and foreign keys are in italics.

***Student*** (Student Number, *PeopleSoft ID*, Financial Aid, Residence Status, Health/Disability Record)

***School-leaving Results*** (Student Number, Subject Name, Final Mark) [Weak Entity]

***First Year Results*** (Student Number, *Course Code*, Final Mark) [Weak Entity]

***Person*** (PeopleSoft ID, Date of Birth, Gender, Address, Nationality, First Name, Middle Name, Surname)

***Lecturer*** (Staff Number, *PeopleSoft ID*, Attendance, Highest Level of Education)

***Lectures*** (Staff Number, Course Code, Years of Experience)

***Course*** (Course Code, Course Name, Lectures Recorded)

***Pre-requisite*** (Course Code, *Pre-requisite Course Code*)

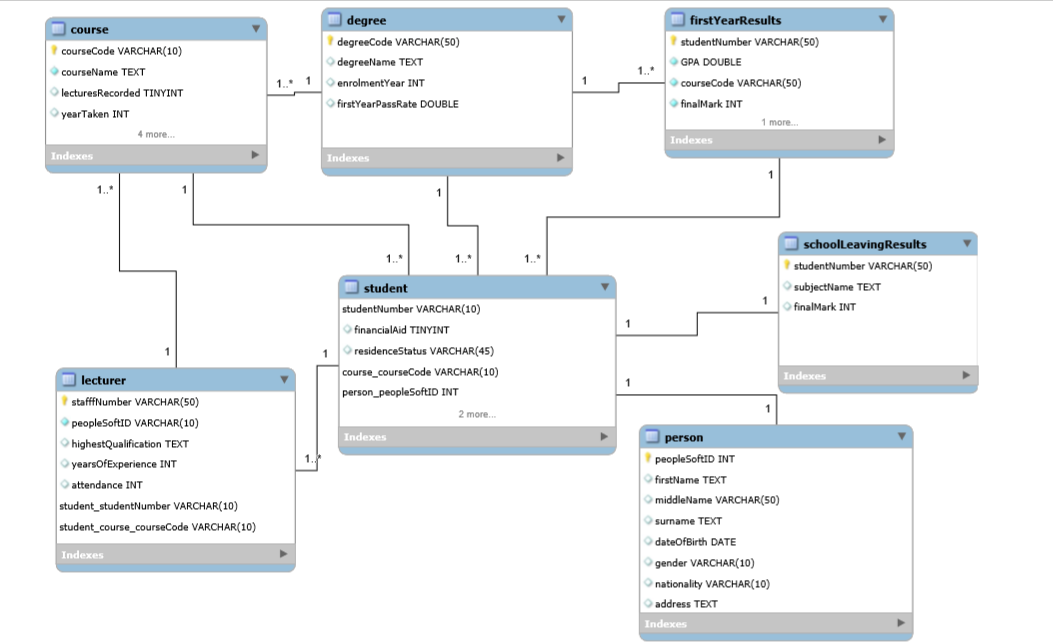
***Takes*** ( Student Number, Course Code, Year)

***Degree*** (Degree Code, Degree Name, First Year Pass Rate)

***Enrols*** (Student Number, Degree Code*,* Enrolment Year)

***Requires*** (Degree Code, Course Code)

## b) ER Diagram



In the diagram above:

* 1\* represents a -to many relationship and 1 represents a to one relationship.
* The blue diamonds represent attributes that can not be null.

## Foreign Key

An example of Foreign Key is First Year Results.Course Code

Another example of Foreign Key is Student.PeopleSoft ID

## Functional Dependency

## Degree Name and First Year Pass Rate are both only functionally dependent on the Degree Code (which is a primary key) in the table Degree.

Degree code à Degree Name

Each degree code maps to one unique degree name. Degree name is functionally dependent on degree code. We will know the name of a degree if we are given its degree code.

Degree code à First Year Pass rate

Each degree code maps to value for First Year Pass rate. First Year Pass rate is functionally dependent on degree code. Given a degree code, we can find out what its first-year pass rate is.

## 6. 2nd Normal Form

## If we add another attribute called Financial Aid Payer to a Student table with a primary key of (Student Number, Financial Aid), and it’s given that Financial Aid à Financial Aid Payer, then the Financial Aid Payer, which is a non-prime attribute, will be partially dependent on the primary key. As only if the financial aid attribute evaluates to true will the financial aid payer be not null. The database will no longer be in 2nd normal form as an attribute will be partially dependent on the key attribute, but not fully dependent.

Shown below is the student table which would not be in 2nd normal form.

***Student*** (Student Number, Financial Aid, PeopleSoft ID, Financial Aid Payer, Residence Status, Health/Disability Record)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Student Number | *PeopleSoft ID* | Financial Aid | Financial Aid Payer | Residence Status | Health/Disability Record |
| RKNLUH001 | 1645678 | True | NED Bank | Off-Campus | none |
| LNNDEA001 | 1678930 | False | null | Off-Campus | none |
| KNHJUG007 | 1865467 | True | ABSA | On-Campus | none |

## 3rd Normal Form

3rd normal form should not have transitive dependences if we add Course Code and course name as attributes to the Degree table. Course Code is functionally depended on degree code (degree code -> course code) and course name is functionally depended on course code (course code -> course name). So, this means that course name is transitively depended on degree code. For example, the Mechatronics Engineering degree and the Electrical Engineering degree both have Computer Science (CSC1016F) as a course code. Meaning that if course code and course name were attributes then the course code and course name for Computer Science will be stored in more than one place making the database not be in 3rd normal form.

Shown below is what the Degree table would look like if it was not in 3rd normal form

***Degree*** (Degree Code, Degree Name, Course Code, Course Name, First Year Pass Rate)

Degree Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Degree Code | Degree Name | Course code | Course Name | First Year Pass Rate |
| EEE | Electrical engineering | CSC1016F | Computer science | 67 |
| MEE | Mechatronic engineering | CSC1016F | Computer science | 76 |

## View Data

CREATE VIEW DegreesAtUCT AS

## SELECT

## Degree Name, First Year Pass rate

## FROM

## Degree

## ---to view all the degree names offered at the university and their first-year pass rates. This is useful to see which degrees have very low pass rates which then identifies degrees that first year students struggle with.

## Contribution

CHTTEN011: Collaborated with WNGJIA001, MFNLIN003 on making the relation scheme and the ER Diagram representation of the scheme.

RGNVER001: contributed to the initial planning of the tables with the attributes. Added examples for 5-8.

WNGJIA001: Composed the ER model, drafted the relation schema and refined it with CHTTEN011, MFNLIN003.

MFNLIN003: Composed the initial set up of the database relations, attributes and got assistance from RGNVER001. Answered questions 5 to 7 with RGNVER001.