National College of Ireland

Introduction to Databases

Continuous Assessment (40%)

Student Name(s): Aurimas Jurgutis

Student ID(s): x20182104

Date of Submission: 26th of March

Lecturer: Tania Malik

Table of Contents

Abstract/Introduction

The aim of this continuous assessment, as per the assessment guidelines, is to:

1. Design a database and identify all the relationships, attributes, primary keys, and entity types.
2. Explain the relations in detail.
3. Draw an ERD to show all the functions mentioned in Part 1.
4. Convert all the entities into a logical dataset and ensure that the designed tables are in 3rd Normal Form.
5. Create, write, populate and test run the database with the desired data to a minimum of 10 tuples per relation.
6. Complete the required modifications outlined in Task Three of the CA guidelines.

To summarize, all the tasks (1-3) specified in the CA guidelines will be completed to achieve the aims of this report.

In this report, we will focus on a college database system accessible to college office clerks that keeps student records only and would be accessible to offices like student services or the exams office, or people within the university with the authority to access these records.

Task 1: Database Design

Part 1

First, we must discuss the entities (tables) and their relations. Note that these tables have not yet been derived to 3rd Normal Form (additional tables will be discussed in Part 5).

- **Departments**

The Department table describes the university department the student is a part of. If a student is enrolled in an engineering course, they will be assigned to the Department of Engineering and Built Environment. If they are enrolled in an Arts course, then they are a part of the Department of Arts & Design. This table is directly related to the student and the faculty, some departments can facilitate multiple faculties, and these can facilitate a vast number of students, i.e., the Department of Engineering and Built Environment can have a faculty of Mechanical, Automation and Design Engineering and another separate faculty for Chemical and Biopharmaceutical Engineering.

- **Courses**

This table describes the courses the students are enrolled in. Students enrol in courses as per normal university standards. There can be a variety of different courses students can enrol in. This table is related to the student, as the student can only apply to one course (one-to-one), a single faculty (many-to-one), where multiple courses are hosted, the multiple different modules (subjects) which the student needs to complete (many-to-one) and the course timetables (one-to-one).

- **Modules**

The module is where the subject of the course is taught to students. For example, an engineering course can have several different modules (one-to-many) and a student can be a part of many different modules (one-to-many).

- **Faculties**

A faculty of a university houses the students that take college courses. Students are assigned to different faculties that are part of an educational department. For the faculty table, the faculty is related to a department, since there can be one or many faculties within a department (many-to-one), the student (one-to-many), and the courses the faculties facilitate (one-to-many).

- **Semesters**

Semesters are half-school-year terms that last for several weeks. The only relation that the semester has is the student (one-to-one) since all students in our case are subject to fixed semester periods.

Part 1(cont.)

- **Timetables**

Timetables are charts that students use to check the attendance periods for their modules. The relations that this table has is the module (one-to-many) since multiple modules are displayed on a timetable, and course (one-to-one) since the timetable needs to display only one course to the student.

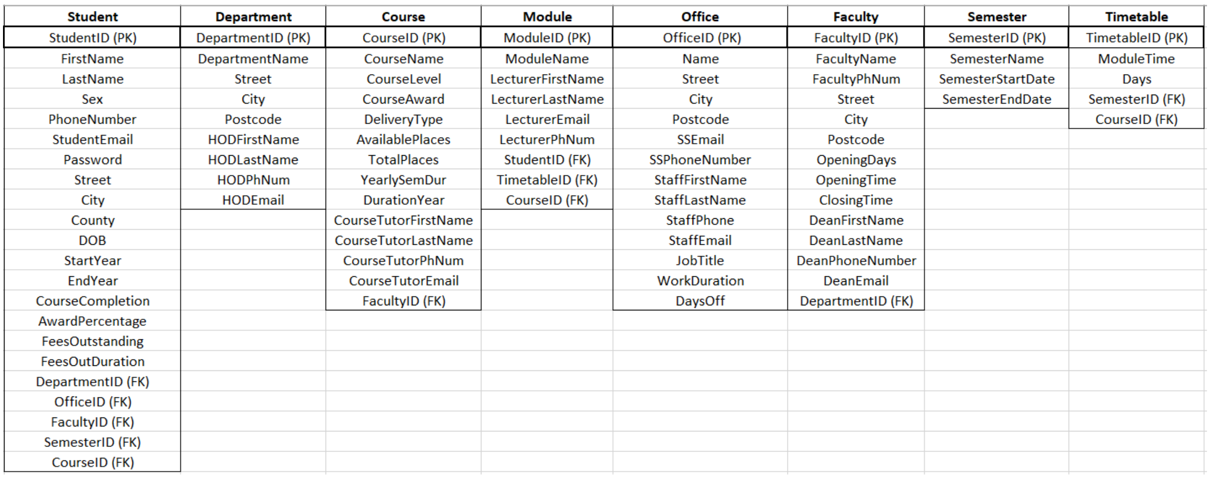
- **Students**

The student is the core part and the most important table in this database. The student is an integral part of a university, a person that pursues a degree by doing a course of their choice. The student is related to all the other 7 tables.

- **College Offices & Staff**

The college office (i.e Student Services) has will oversee storing and modifying information about the students. The office has its own table for storing information about its address, phone number, staff information etc. The college must have at least one office for each faculty.

Part 2



Student	Department	Course	Module	Office	Faculty	Semester	Timetable
StudentID (PK)	DepartmentID (PK)	CourseID (PK)	ModuleID (PK)	OfficeID (PK)	FacultyID (PK)	SemesterID (PK)	TimetableID (PK)
FirstName	DepartmentName	CourseName	ModuleName	Name	FacultyName	SemesterName	ModuleTime
LastName	Street	CourseLevel	LecturerFirstName	Street	FacultyPhNum	SemesterStartDate	Days
Sex	City	CourseAward	LecturerLastName	City	Street	SemesterEndDate	SemesterID (FK)
PhoneNumber	Postcode	DeliveryType	LecturerEmail	Postcode	City		CourseID (FK)
StudentEmail	HODFirstName	AvailablePlaces	LecturerPhNum	SSEmail	Postcode		
Password	HODLastName	TotalPlaces	StudentID (FK)	SSPhoneNumber	OpeningDays		
Street	HODPhNum	YearlySemDur	TimetableID (FK)	StaffFirstName	OpeningTime		
City	HODEmail	DurationYear	CourseID (FK)	StaffLastName	ClosingTime		
County		CourseTutorFirstName		StaffPhone	DeanFirstName		
DOB		CourseTutorLastName		StaffEmail	DeanLastName		
StartYear		CourseTutorPhNum		JobTitle	DeanPhoneNumber		
EndYear		CourseTutorEmail		WorkDuration	DeanEmail		
CourseCompletion		FacultyID (FK)		DaysOff	DepartmentID (FK)		
AwardPercentage							
FeesOutstanding							
FeesOutDuration							
DepartmentID (FK)							
OfficeID (FK)							
FacultyID (FK)							
SemesterID (FK)							
CourseID (FK)							

*Taken from Microsoft Excel

(PK) = Primary Key (FK) = Foreign Key

Student: Strong entity type

Department: Weak entity type

Course: Weak entity type

Module: Weak entity type

Office: Strong entity type

Faculty: Weak entity type

Semester: Weak entity type

Timetable: Weak entity type

Part 3

- **Relationships, Cardinalities and Participations with the Student Table**

The student has relationships with every table in the database. Starting with Semester, the relationship this table has is mapped as one-to-one. The reason for this is that each individual student will be a part of a single semester during the academic year. When SemesterID = 1, this means that the student is in the first semester running from September to February. When SemesterID = 2, this means that the student is in the second semester running from February to May. The participation from the side of the student to the semester is total, because again each student is assigned to a single semester. The semester has partial participation because all students are assigned to one semester.

The office table is related to the student because the office would need to access the student's records and each student in our case is assigned to an office for the faculty where they attend their course. This relation is mapped as one-to-many. The office has total participation with the student because of the offices' ability to access student records; the students cannot do that, therefore they have only a partial participation with the office.

Module represents the modules in the course the student is enrolled in. The Module has a one-to-many relationship with the student since a single student can be enrolled in multiple different modules their course has to offer. The student has total participation with the module, but the module only has partial participation because some students can have the option of deferring from a module if they meet certain criteria to do so.

The course is where students are enrolled to study in their assigned field. The course has a one-to-one relationship with the student since he/she can only be enrolled in one course. The student has a total participation with the course since the student is required to be enrolled to a course to register on the college system. The course has a partial participation because the student can drop out or defer from a course, and the student does not attend every course the college offers.

A department is a division of the college where a specific set of courses are being taught to students, i.e. Department of Engineering, Department of Arts & Humanities etc. The relationship the student has with the department is many-to-one, since one or more students can be assigned to the same department. The student has total participation with the department since the student technically must be part of a department, but the department has a partial participation since there are many departments throughout the college.

The faculty is a subdivision of the department. This is where the courses are facilitated, and it is where the students study and complete the courses. The student has a many-to-one relationship with the faculty, since there can be many students situated in a single faculty where many courses are being facilitated. The student has total participation with the faculty, but the faculty has partial participation, like the department.

The timetable is a list of modules/lectures with times and room assigned to the students. The student has a one-to-one relationship with the student since every student is assigned a unique timetable. The student has a total participation with the timetable and the timetable has a partial participation.

Part 3 (cont.)

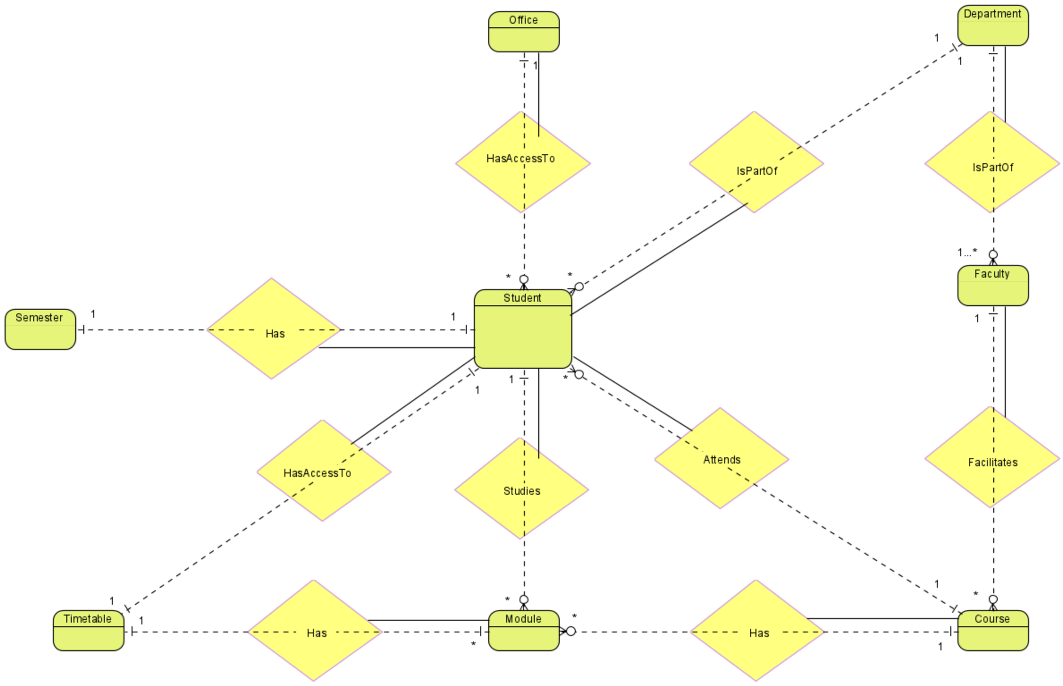
- **Relationships, Cardinalities and Participations outside the Student Table**

The faculty is a subdivision of the department. The department can have one or more faculties in our example, therefore the relationship with the department to the faculty is one-to-many. The department has total participation to the faculty whereas the faculty can have partial participation, since multiple faculties within a department have different but similar functions, i.e. Department of Engineering can have a Faculty of Mechanical and Civil Engineering and a Faculty of Chemical and Biopharmaceutical Engineering.

As discussed previously, the faculty can facilitate many courses relating to the pertained field. The relationship the faculty shares with the course is one-to-many. The faculty has a total participation with the course, but the course only has a partial participation, because a faculty manages all the courses, but the course cannot change anything related to the faculty.

The courses have modules that are subjects taught to students. The relationship between the course and the module is many-to-one since there can be more than one module per course. The course has a total participation, and the module has a partial participation since a single course needs to facilitate at least two modules, and a module is not dependent on the course.

Part 4

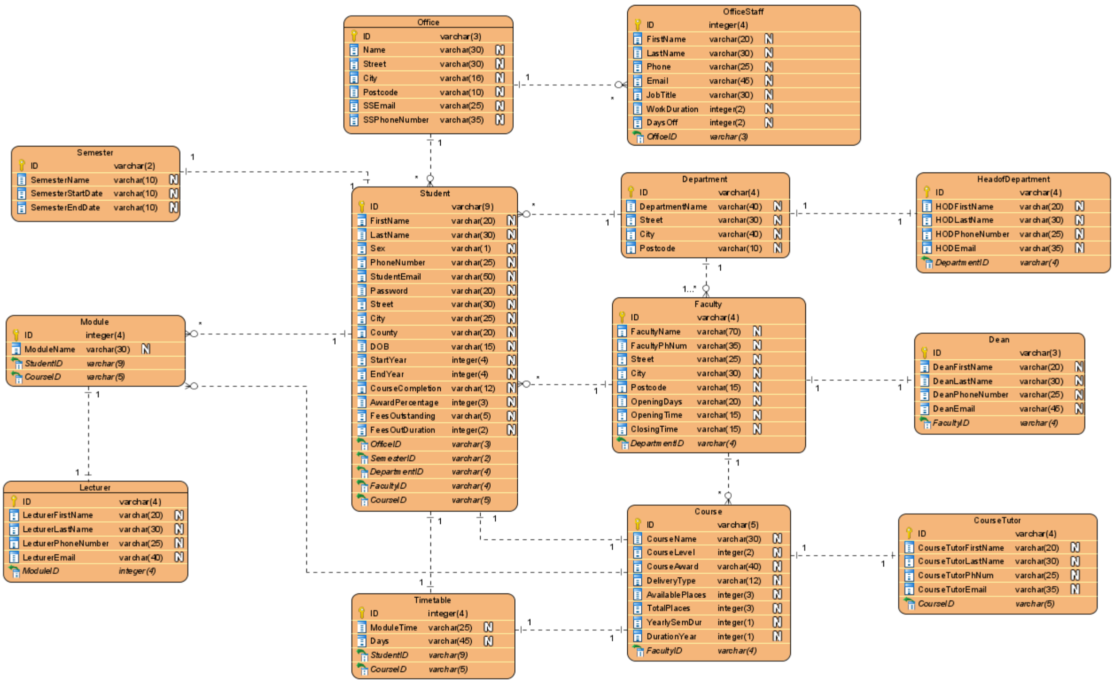


Legend: = **One to One**

= **One to Many**

Note: A dashed and normal line represents total participation.

Part 5



Legend: -|- - - - - - - - - - - -|- **= One to One**

-|- -○< **= One to Many**

N **= Nullable Value**

- **New Tables**

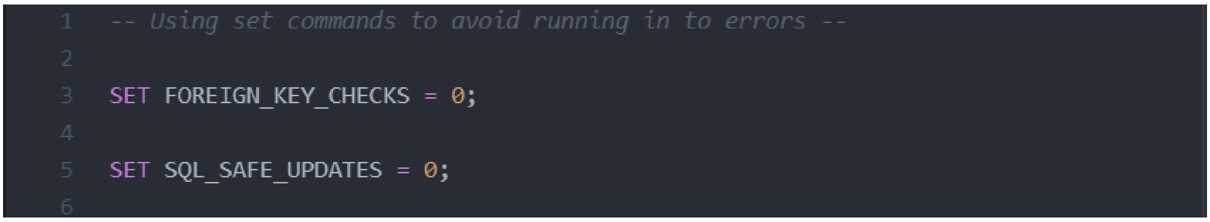
As seen from the diagram, for the tables to be in 3rd Normal Form, five new tables have been created to accommodate this requirement. Separate tables for the lecturer of the module, course tutor, dean, head of department, and office staff are created. The relationships for each of these tables is one-to-one, except for the office-office staff relation, where it is many-to-one. The participation constraints follow the same methodology as previously explained, but the order of the implementation is reversed, i.e. the Course tutor has total participation but the course has limited/partial participation, since the course tutor is the only one who can make changes to the course.

Task 2: Creating and Populating the Database

Part 1

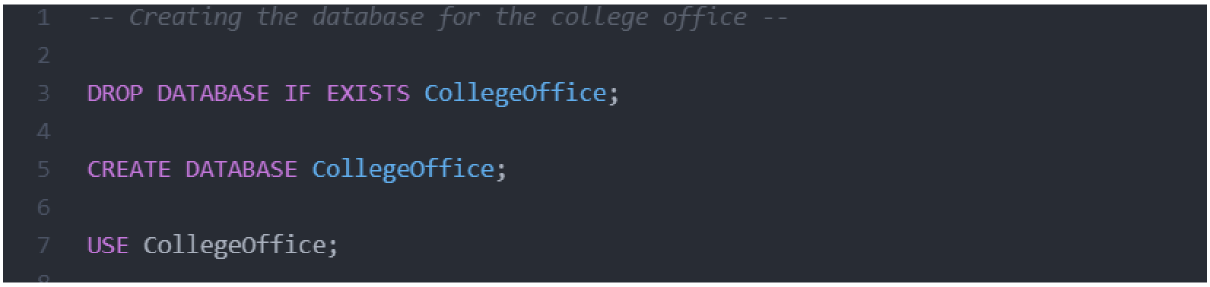
Note: The code was written using Atom text editor from my own PC and tested on MySQL Workbench and the SQL Prompt through Citrix.

1. Set parameters to avoid running in to irrelevant errors:



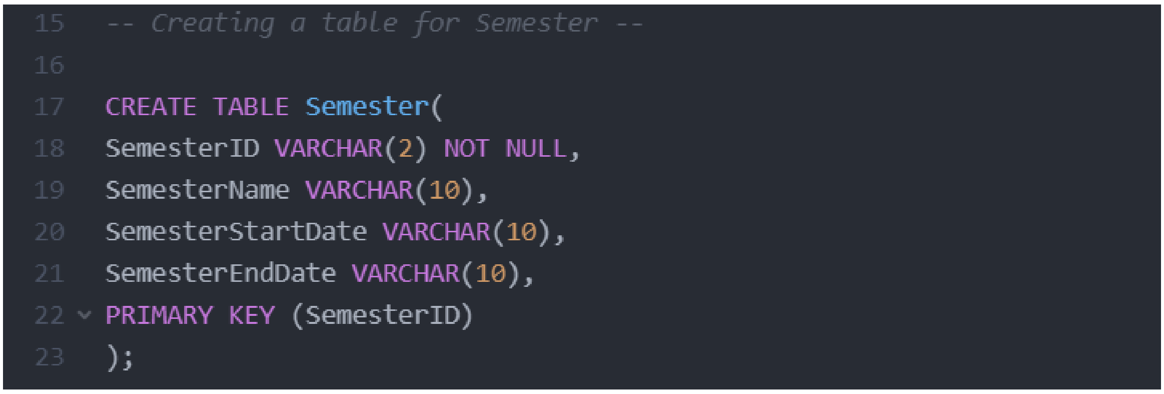
```
-- Using set commands to avoid running in to errors --

SET FOREIGN_KEY_CHECKS = 0;

SET SQL_SAFE_UPDATES = 0;

```

2. Creating a database:



```
-- Creating the database for the college office --

DROP DATABASE IF EXISTS CollegeOffice;

CREATE DATABASE CollegeOffice;

USE CollegeOffice;
```

3. Command line to create each table (Semester example):



```
-- Creating a table for Semester --

CREATE TABLE Semester(
SemesterID VARCHAR(2) NOT NULL,
SemesterName VARCHAR(10),
SemesterStartDate VARCHAR(10),
SemesterEndDate VARCHAR(10),
PRIMARY KEY (SemesterID)
);
```

Code format for creating a table:

CREATE TABLE Example(

X DATATYPE (x) **or** X DATATYPE (x) NOT NULL (**if its a non-nullable value**),

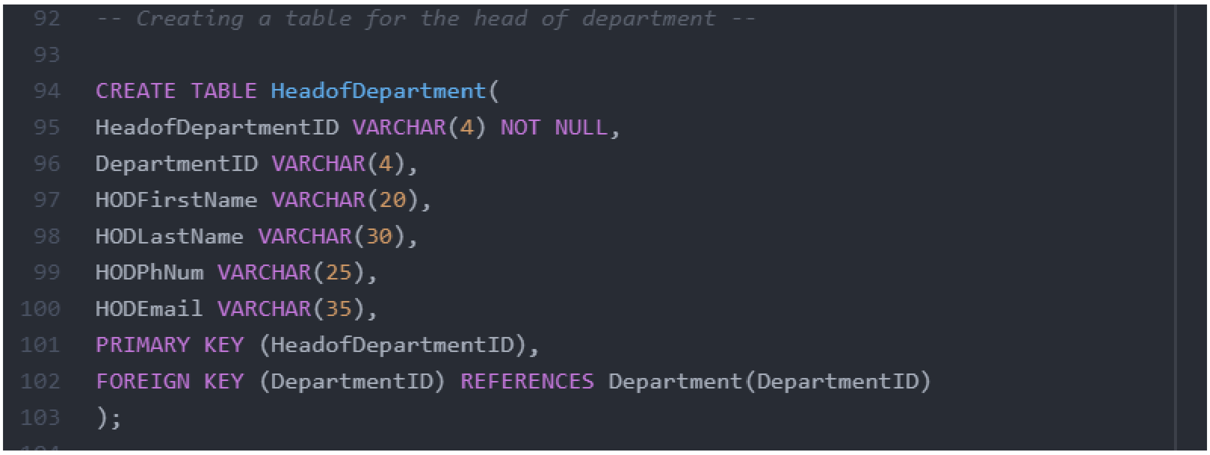
PRIMARY KEY (x)

);

This step was repeated for each table.

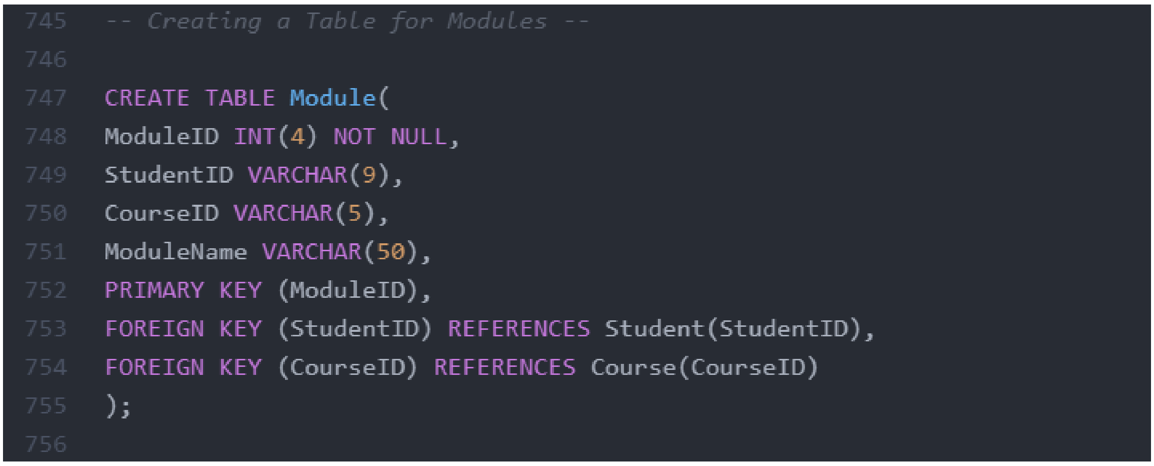
Part 1(cont.)

4. How foreign keys would be written inside a table:



```
-- Creating a table for the head of department --

CREATE TABLE HeadofDepartment(
HeadofDepartmentID VARCHAR(4) NOT NULL,
DepartmentID VARCHAR(4),
HODFirstName VARCHAR(20),
HODLastName VARCHAR(30),
HODPhNum VARCHAR(25),
HODEmail VARCHAR(35),
PRIMARY KEY (HeadofDepartmentID),
FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID)
);
```

5. For adding multiple foreign keys:

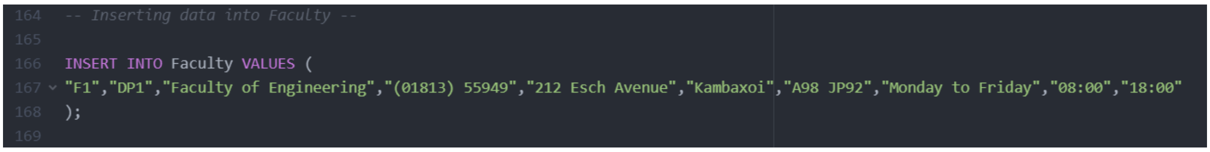


```
-- Creating a Table for Modules --

CREATE TABLE Module(
ModuleID INT(4) NOT NULL,
StudentID VARCHAR(9),
CourseID VARCHAR(5),
ModuleName VARCHAR(50),
PRIMARY KEY (ModuleID),
FOREIGN KEY (StudentID) REFERENCES Student(StudentID),
FOREIGN KEY (CourseID) REFERENCES Course(CourseID)
);
```

Part 2

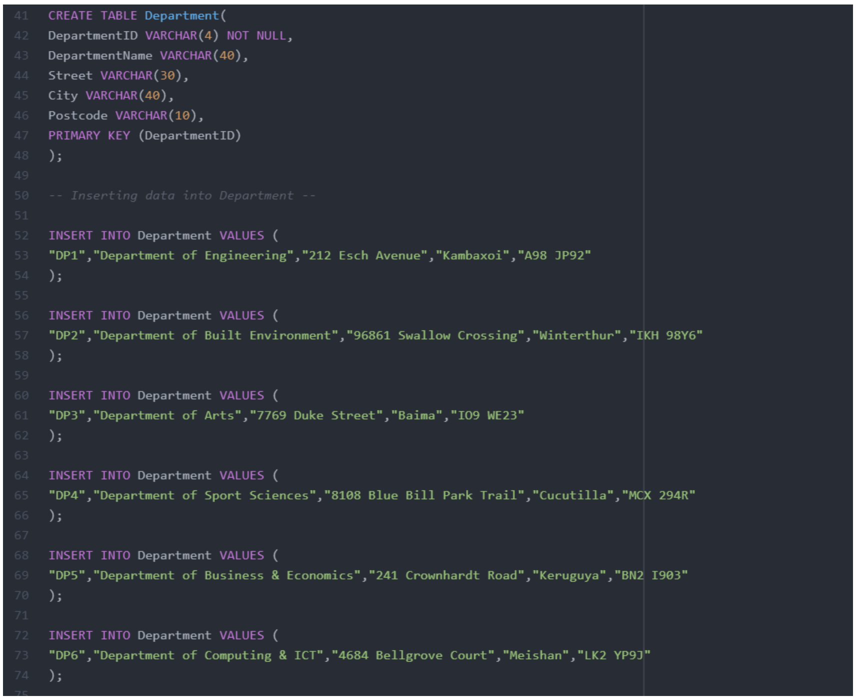
Note: To populate most of the tables I used Mockaroo, a website which generates random information for a specified category.

1. Inserting values into a table:

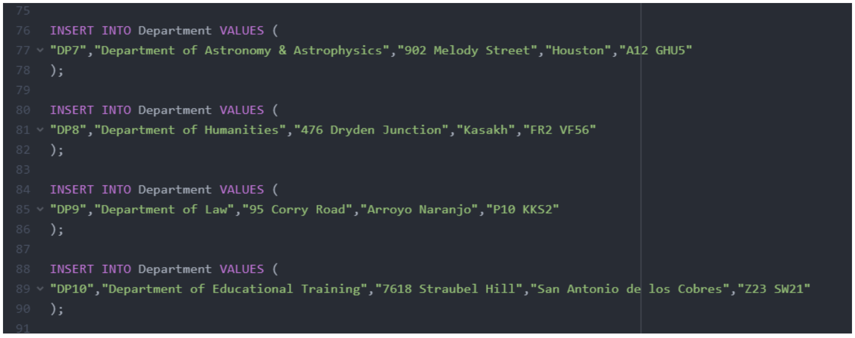


```
-- Inserting data into Faculty --

INSERT INTO Faculty VALUES (
"F1","DP1","Faculty of Engineering","(01813) 55949","212 Esch Avenue","Kambaxoi","A98 JP92","Monday to Friday","08:00","18:00"
);
```

When the code for each table was written out, the insert into x values statement was used to populate each table like this:



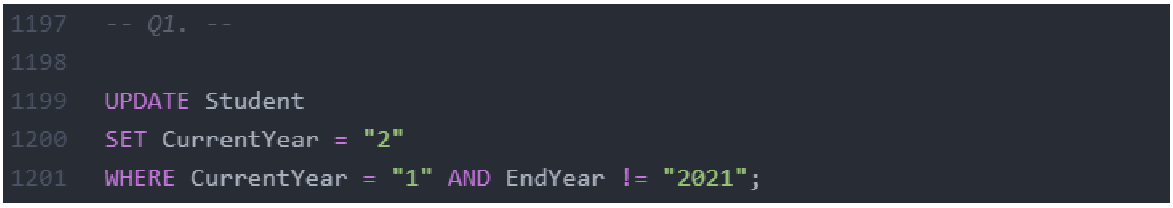
```
CREATE TABLE Department(
DepartmentID VARCHAR(4) NOT NULL,
DepartmentName VARCHAR(40),
Street VARCHAR(30),
City VARCHAR(40),
Postcode VARCHAR(10),
PRIMARY KEY (DepartmentID)
);

-- Inserting data into Department --

INSERT INTO Department VALUES (
"DP1","Department of Engineering","212 Esch Avenue","Kambaxoi","A98 JP92"
);

INSERT INTO Department VALUES (
"DP2","Department of Built Environment","96861 Swallow Crossing","Winterthur","IKH 98Y6"
);

INSERT INTO Department VALUES (
"DP3","Department of Arts","7769 Duke Street","Baima","I09 WE23"
);

INSERT INTO Department VALUES (
"DP4","Department of Sport Sciences","8108 Blue Bill Park Trail","Cucutilla","MCX 294R"
);

INSERT INTO Department VALUES (
"DP5","Department of Business & Economics","241 Crownhardt Road","Keruguya","BN2 I903"
);

INSERT INTO Department VALUES (
"DP6","Department of Computing & ICT","4684 Bellgrove Court","Meishan","LK2 YP9J"
);
```



```
INSERT INTO Department VALUES (
"DP7","Department of Astronomy & Astrophysics","902 Melody Street","Houston","A12 GHU5"
);

INSERT INTO Department VALUES (
"DP8","Department of Humanities","476 Dryden Junction","Kasakh","FR2 VF56"
);

INSERT INTO Department VALUES (
"DP9","Department of Law","95 Corry Road","Arroyo Naranjo","P10 KKS2"
);

INSERT INTO Department VALUES (
"DP10","Department of Educational Training","7618 Straubel Hill","San Antonio de los Cobres","Z23 SW21"
);
```

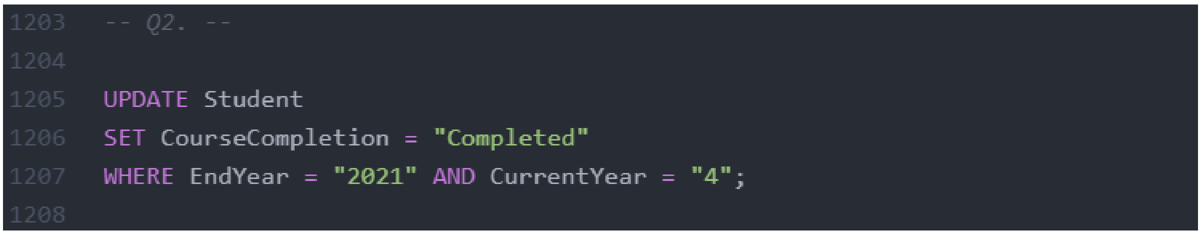
Task 3: Querying the Database

- Question 1



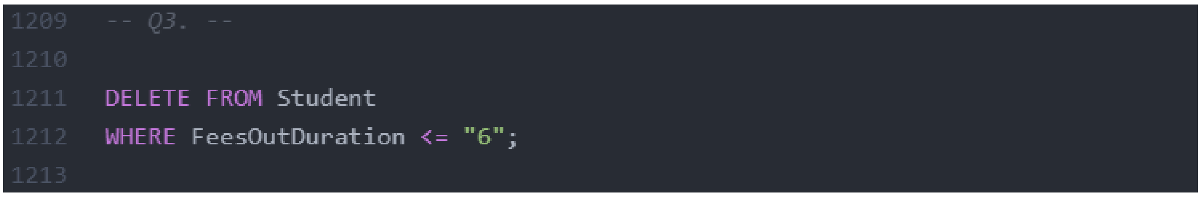
```sql
-- Q1. --

UPDATE Student
SET CurrentYear = "2"
WHERE CurrentYear = "1" AND EndYear != "2021";
```

- Question 2



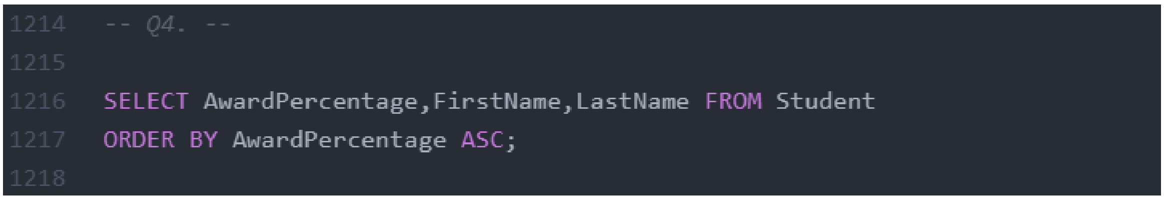
```sql
-- Q2. --

UPDATE Student
SET CourseCompletion = "Completed"
WHERE EndYear = "2021" AND CurrentYear = "4";
```

- Question 3



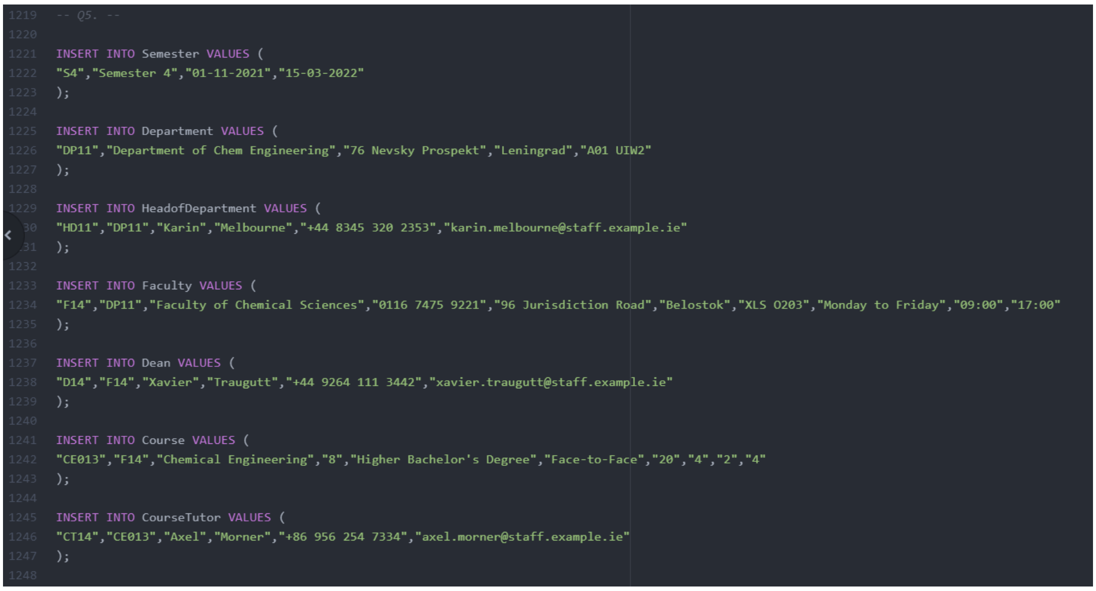
```sql
-- Q3. --

DELETE FROM Student
WHERE FeesOutDuration <= "6";
```

- Question 4

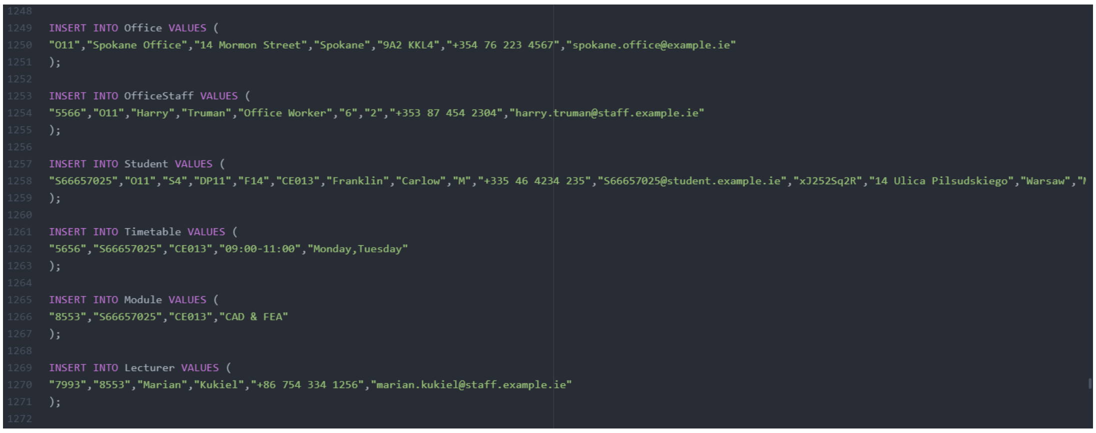


```sql
-- Q4. --

SELECT AwardPercentage,FirstName,LastName FROM Student
ORDER BY AwardPercentage ASC;
```

- Question 5

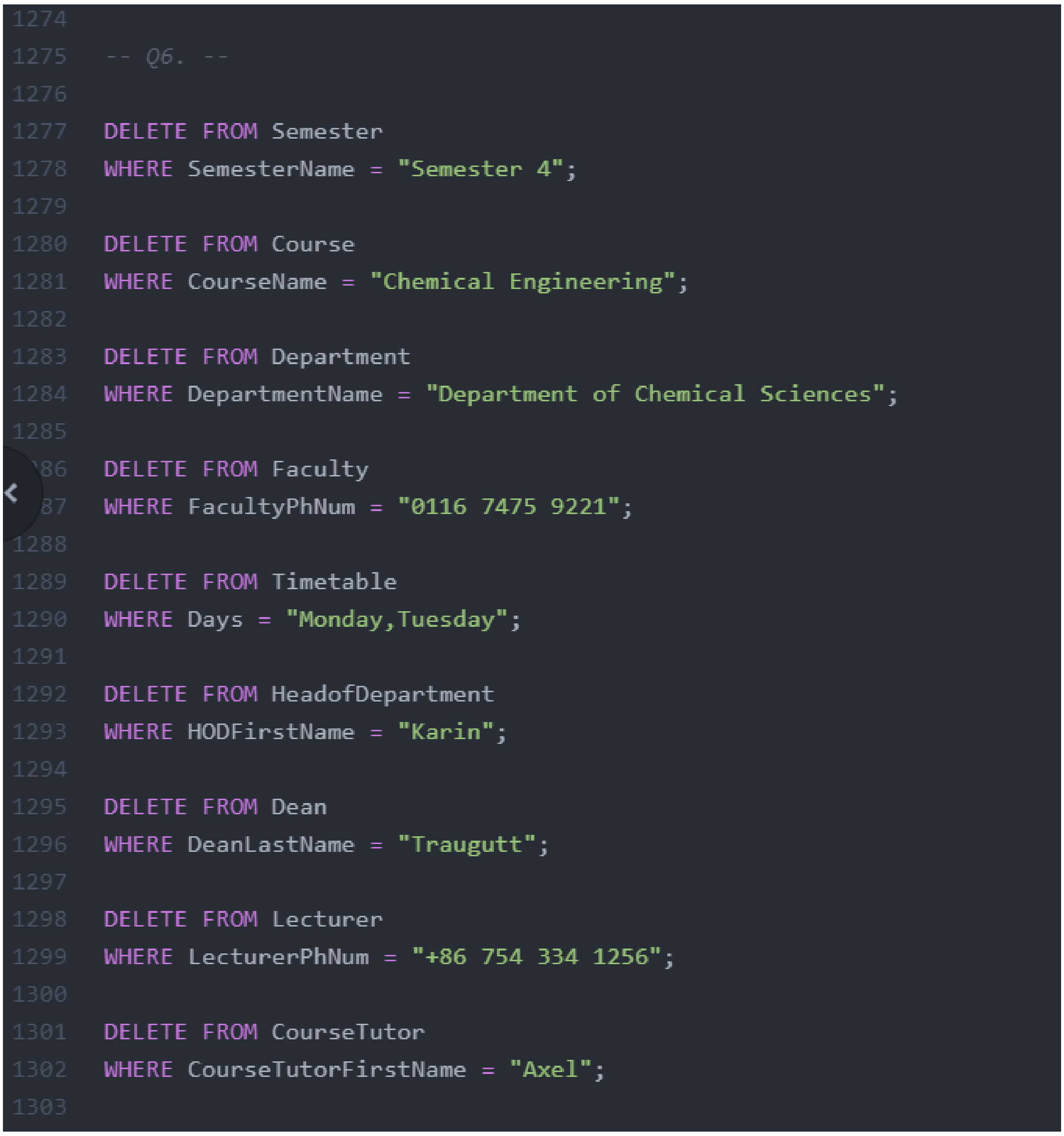


```
-- Q5. --

INSERT INTO Semester VALUES (
"S4","Semester 4","01-11-2021","15-03-2022"
);

INSERT INTO Department VALUES (
"DP11","Department of Chem Engineering","76 Nevsky Prospekt","Leningrad","A01 UIW2"
);

INSERT INTO HeadofDepartment VALUES (
"HD11","DP11","Karin","Melbourne","+44 8345 320 2353","karin.melbourne@staff.example.ie"
);

INSERT INTO Faculty VALUES (
"F14","DP11","Faculty of Chemical Sciences","0116 7475 9221","96 Jurisdiction Road","Belostok","XLS 0203","Monday to Friday","09:00","17:00"
);

INSERT INTO Dean VALUES (
"D14","F14","Xavier","Traugutt","+44 9264 111 3442","xavier.traugutt@staff.example.ie"
);

INSERT INTO Course VALUES (
"CE013","F14","Chemical Engineering","8","Higher Bachelor's Degree","Face-to-Face","20","4","2","4"
);

INSERT INTO CourseTutor VALUES (
"CT14","CE013","Axel","Morner","+86 956 254 7334","axel.morner@staff.example.ie"
);
```

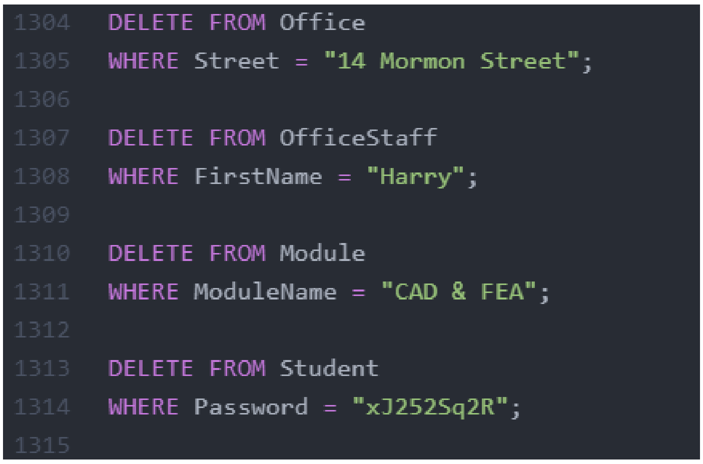


```
INSERT INTO Office VALUES (
"O11","Spokane Office","14 Mormon Street","Spokane","9A2 KKL4","+354 76 223 4567","spokane.office@example.ie"
);

INSERT INTO OfficeStaff VALUES (
"5566","O11","Harry","Truman","Office Worker","6","2","+353 87 454 2304","harry.truman@staff.example.ie"
);

INSERT INTO Student VALUES (
"S66657025","O11","S4","DP11","F14","CE013","Franklin","Carlow","M","+335 46 4234 235","S66657025@student.example.ie","xJ252Sq2R","14 Ulica Pilsudskiego","Warsaw","
);

INSERT INTO Timetable VALUES (
"5656","S66657025","CE013","09:00-11:00","Monday,Tuesday"
);

INSERT INTO Module VALUES (
"8553","S66657025","CE013","CAD & FEA"
);

INSERT INTO Lecturer VALUES (
"7993","8553","Marian","Kukiel","+86 754 334 1256","marian.kukiel@staff.example.ie"
);
```

- Question 6

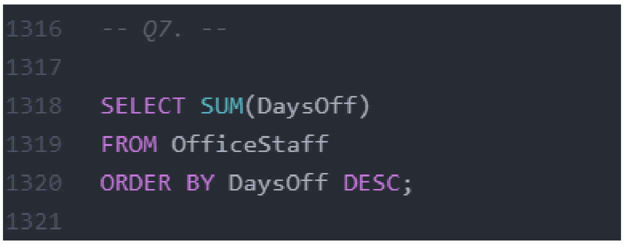


```
-- Q6. --

DELETE FROM Semester
WHERE SemesterName = "Semester 4";

DELETE FROM Course
WHERE CourseName = "Chemical Engineering";

DELETE FROM Department
WHERE DepartmentName = "Department of Chemical Sciences";

DELETE FROM Faculty
WHERE FacultyPhNum = "0116 7475 9221";

DELETE FROM Timetable
WHERE Days = "Monday,Tuesday";

DELETE FROM HeadofDepartment
WHERE HODFirstName = "Karin";

DELETE FROM Dean
WHERE DeanLastName = "Traugutt";

DELETE FROM Lecturer
WHERE LecturerPhNum = "+86 754 334 1256";

DELETE FROM CourseTutor
WHERE CourseTutorFirstName = "Axel";
```



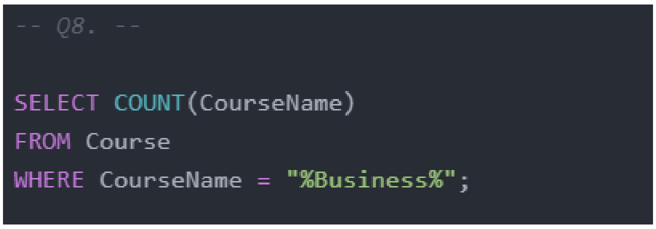
```
DELETE FROM Office
WHERE Street = "14 Mormon Street";

DELETE FROM OfficeStaff
WHERE FirstName = "Harry";

DELETE FROM Module
WHERE ModuleName = "CAD & FEA";

DELETE FROM Student
WHERE Password = "xJ252Sq2R";
```

- Question 7



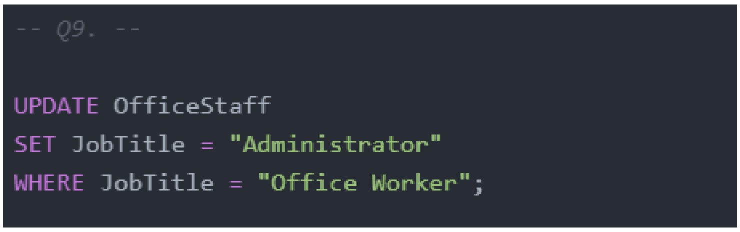
```sql
-- Q7. --

SELECT SUM(DaysOff)
FROM OfficeStaff
ORDER BY DaysOff DESC;
```

- Question 8



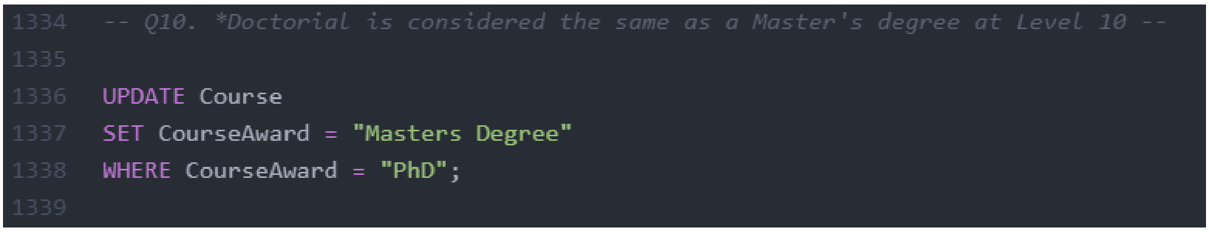
```sql
-- Q8. --

SELECT COUNT(CourseName)
FROM Course
WHERE CourseName = "%Business%";
```

- Question 9



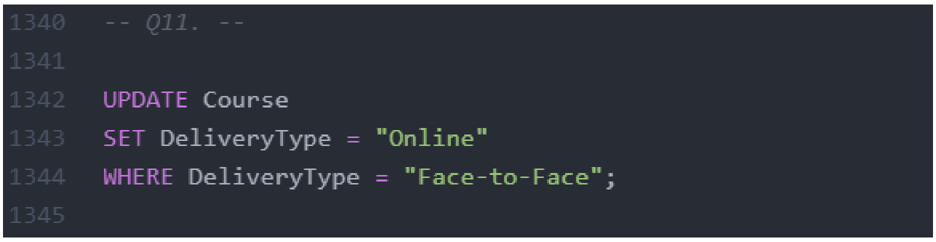
```sql
-- Q9. --

UPDATE OfficeStaff
SET JobTitle = "Administrator"
WHERE JobTitle = "Office Worker";
```

- Question 10



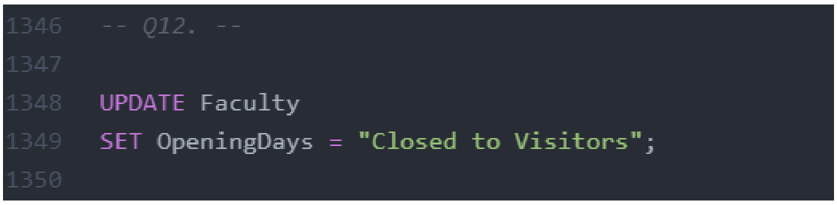
```sql
-- Q10. *Doctorial is considered the same as a Master's degree at Level 10 --

UPDATE Course
SET CourseAward = "Masters Degree"
WHERE CourseAward = "PhD";
```

- Question 11



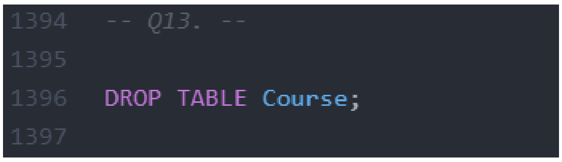
```sql
-- Q11. --

UPDATE Course
SET DeliveryType = "Online"
WHERE DeliveryType = "Face-to-Face";
```

- Question 12



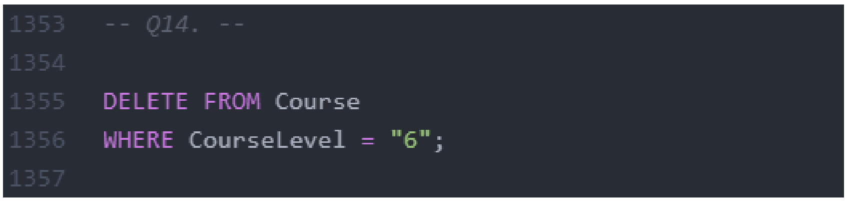
```sql
-- Q12. --

UPDATE Faculty
SET OpeningDays = "Closed to Visitors";
```

- Question 13



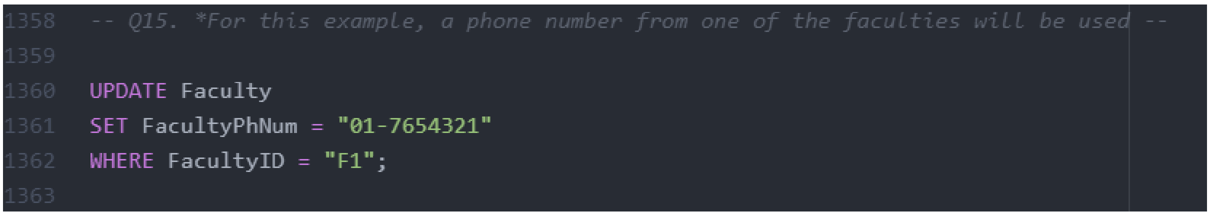
```sql
-- Q13. --

DROP TABLE Course;

```

- Question 14



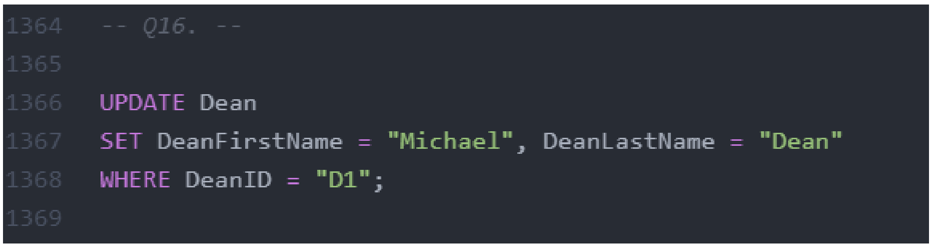
```sql
-- Q14. --

DELETE FROM Course
WHERE CourseLevel = "6";

```

- Question 15



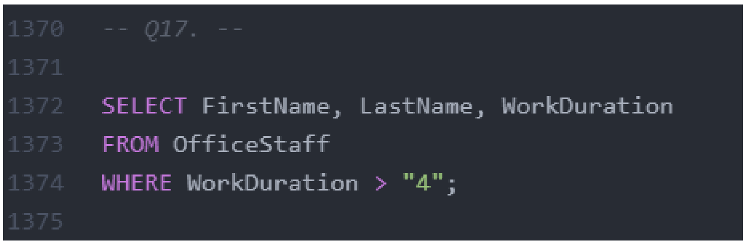
```sql
-- Q15. *For this example, a phone number from one of the faculties will be used --

UPDATE Faculty
SET FacultyPhNum = "01-7654321"
WHERE FacultyID = "F1";

```

- Question 16



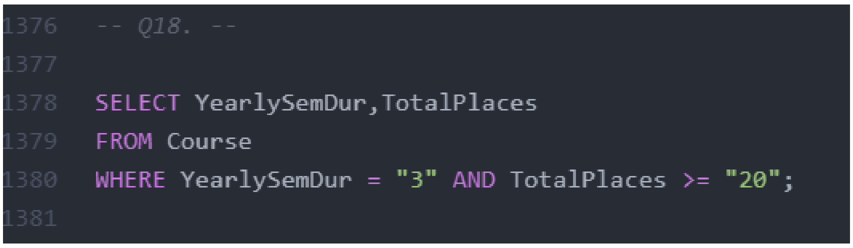
```sql
-- Q16. --

UPDATE Dean
SET DeanFirstName = "Michael", DeanLastName = "Dean"
WHERE DeanID = "D1";

```

- Question 17



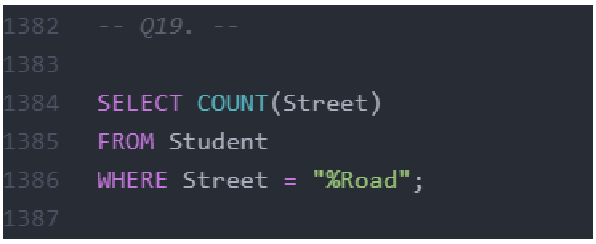
```sql
-- Q17. --

SELECT FirstName, LastName, WorkDuration
FROM OfficeStaff
WHERE WorkDuration > "4";

```

- Question 18



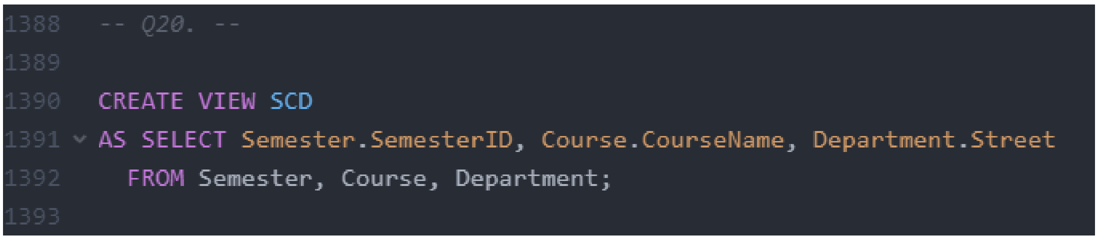
```sql
-- Q18. --

SELECT YearlySemDur,TotalPlaces
FROM Course
WHERE YearlySemDur = "3" AND TotalPlaces >= "20";

```

- Question 19



```sql
-- Q19. --

SELECT COUNT(Street)
FROM Student
WHERE Street = "%Road";

```

- Question 20



```sql
-- Q20. --

CREATE VIEW SCD
AS SELECT Semester.SemesterID, Course.CourseName, Department.Street
  FROM Semester, Course, Department;

```

Conclusions/Additional Comments

1. The set parameters had to be used to avoid running into errors 1175 and 1452, as displayed on the workbench. What seems to be causing error 1175 is that the primary keys have no where clause, but this clause would not need to be applied in our database, so SQL SET_SAFE_UPDATES = 0; was used to get rid of this error. As for error 1452, it is still unclear how this error could be caused, but it is possible that there were too many foreign keys used per table, or the constraints in some of these tables already exist.

2. The database was originally created with the aim to make it as realistic as possible, but due to time constraints it was impossible to accomplish that, so the database was kept as simple as possible.