Departments

Name - up to 100 characters
Subject Abbreviation - up to 4 characters.
The subject abbreviation must be unique.

```
Create table Departments(
Name varchar(100) not null,
SubjectAbr varchar(4),
Primary key(SubjectAbr)
);
```

Courses

Name - up to 100 characters
Number - a four-digit number (you do not need to restrict the number of digits to 4, but it must be capable of holding that many).
The combination of the Course number and the Department which it belongs to must be unique.

```
Create table Courses(
Name varchar(100) not null,
Number int unsigned not null,
SubjectAbr varchar(4),
CourseID int unsigned auto_increment,
Primary key (CourseID),
Foreign key (SubjectAbr) references Departments(SubjectAbr),
unique(SubjectAbr, Number)
);
```

Classes

Semester - must be represented as a year (number) plus a season. The seasons can be one of "Spring", "Fall", or "Summer". Due to a limitation in a tool we will use in the future, do not use the SQL YEAR type, represent the year as an unsigned int.
Location - up to 100 characters
Start time - a time
End time - a time
The combination of the semester plus the Course to which the Class belongs must be unique.

We will prevent classes from overlapping in location and time using web server logic, not with table constraints.

```
Create table Classes(
Season enum(“Spring”, “Summer”, “Fall”) not null,
Year int unsigned no null,
```

```
Location varchar(100) not null,
StartTime time not null,
EndTime time not null,
ClassID int unsigned auto_increment,
CourseID int unsigned not null,
Professor varchar(8) not null,
Foreign key (CourseID) references Courses(CourseID),
Foreign key(Professor) references Professors(uID),
Unique(Season, Year, CourseID),
Primary key(ClassID)
);
```

Students, Professors, Administrators

All three of these types are users. The recommended translation is to create a separate table for each of them, and no table for the abstract Users type.

uID - a 'u' followed by seven digits, as in "u1234567".
First name - up to 100 characters
Last name - up to 100 characters
Date of birth - a date
The uID must be unique.

```
Create table Students (
uID varchar(8) not null,
fName varchar (100) not null,
lName varchar(100) not null,
DoB DATE not null,
Major varchar(4) NOT NULL,
Primary key (uID),
Foreign key (Major) references Departments(SubjectAbr)
);

Create table Professors (
uID varchar(8) not null,
fName varchar (100) not null,
lName varchar(100) not null,
DoB DATE not null,
Department varchar(4) NOT NULL,
Primary key (uID),
Foreign key (Department) references Departments(SubjectAbr)
);

Create table Administrators (
uID varchar(8) not null,
```

```
fName varchar (100) not null,
lName varchar(100) not null,
DoB DATE not null,
Primary key (uID)
);
```

Assignment Categories

Grading weight - an unsigned integer representing a percentage (0 - 100)
Name - up to 100 characters
The combination of the Name plus the Class to which the category belongs must be unique.

We recommend adding an extra ID column as a primary key for this table.

```
Create table AssignmentCategories(
AssCatID int unsigned auto_increment,
GradeWeight int unsigned not null,
Name varchar(100) not null,
ClassID int unsigned not null,
Primary key (AssCatID),
Foreign key (ClassID) references Classes(ClassID),
unique(Name, ClassID),
Constraint checkRegNumber CHECK (GradeWeight<=100 AND GradeWeight > 0)
);
```

Assignments

Name - up to 100 characters
Maximum point value - an unsigned integer
Contents - plain text containing html, up to 8192 characters
Due date/time - a date and time (hint: mysql has a type for this)

```
Create table Assignments (
AssName varchar (100) not null,
AssCatID int unsigned not null,
AssID int unsigned auto_increment,
MaxPt int unsigned not null,
Primary key (AssID),
unique(AssName, AssCatID),
Foreign key (AssCatID) references AssignmentCategories(AssCatID)
);
```

Submission

Date and time - the exact date and time of a submission should be tracked
Score - an unsigned integer
Contents - a text submission up to 8192 characters

```
Create table AssignmentSubmission(
SubmissionTime datetime not null,
Score float,
Contents text (8192),
AssID int unsigned not null,
uID varchar(8) not null,
Foreign key (uID) references Students(uID),
Foreign key (AssID) references Assignments(AssID),
Primary key (uID, AssID)
);
```

Enrollment Grade

The grade a student earns for a class enrollment should be up to two characters (such as "A" or "A-")

Enrollment:

```
Create table Enrollment(
uID varchar(8) not null,
ClassID int unsigned not null,
Grade varchar(2),
Foreign key (uID) references Students(uID),
Foreign key (ClassID) references Classes(ClassID),
Primary key(uID, ClassID)
);
```

Hints

Consult the ER to RM algorithm discussed in class.
Some of these tables may have additional columns based on their relationships with other entities.

Expect to make mistakes! You should expect to alter tables, or even drop and re-create tables multiple times throughout this assignment.
Do not add any test data to your tables until you are satisfied they are finalized. This would just make it more difficult to correct your tables if you have to change them.