Provided are the **Requirements of a Client to create a database for Firefox University.**

* The client wants a system that should keep track of all students (their names, IDs, and addresses) and professors (Names, IDs, departments). The client is not interested to keep record of the address of professors but wants to keep track of their ages.
* Furthermore, the system should help maintain details of all courses offered and details about the courses. For instance, we should be able to retrieve the attrition of a course taught by a faculty during a quarter. or What is the current enrollment? and Which department offers a specific course?
* In addition, the client is building this system as it wants to meet the university board specified constraints. The system should therefore enable the client to know when
  + A course can have ≥ 0 pre-requisites
  + Each course must be taught by a professor
  + Every student evaluates the professor teaching the course
  + Each student receives a single grade (A, B, C, D, or F) for each course registered at the end of the quarter
  + The departments are unique, and should have at least one department chair
  + A department chair is not allowed to head more than one department

**Part 1: Task -** Given the following Conceptual Model: (a) Verify the corresponding Relational Model; Ensure that you understand the relational model – especially the Entity and the Referential Integrity Constraints

STUDENT (SID)

PROFESSOR (PID)

COURSE (Course\_Num)

DEPARTMENT (Dept\_Name)

Teach

Takes

Pre-req

Belong

(1, N)

(1, N)

(1, N)

(1,1)

(1, N)

(0, N)

(1, N)

(1, N)

(1, N)

**Turn In this Sheet:**

NAME:

CWID:

|  |  |  |  |
| --- | --- | --- | --- |
| STUDENT | | | |
| SID |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| PROFESSOR | | | |
| PID |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Takes | | | |
| SID | Cour.Num |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Teach | | | |
| PID | Dept\_Name | Course\_Num |  |

|  |  |  |  |
| --- | --- | --- | --- |
| COURSE | | | |
| CourseNum | Dept\_Name |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| DEPARTMENT | | | |
| Dept\_Name |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Prereq | | | |
| Course\_Num | Dept\_Name |  |  |

(b) Write down the resultant schemas of the tables in the database. Note, list the relational schemas of **entities first**, followed by the schemas of the relationships.

DEPARTMENT(*Dept\_Name*, Dept\_Chair)

COURSE(*CourseNum*,*Dept\_Name,* …)

TAKES(SID, CourseNum, Grade,…)

**Part 2: Table Descriptions (your meta data)**

The tables are described in order of relationship described in the previous figure.

STUDENT

There is one row in the STUDENT table for each student registering for the course.

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Meaning | NULL allowed |
| SID | Varchar (10) | Student ID | No |
| Name | Varchar (45) | Last, First Name | Yes |
| Address | Varchar (45) | Address | Yes |

PROFESSOR

There is one row in the PROFESSOR table for each professor in the department

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Meaning | NULL allowed |
| PID | Varchar (10) | Professor ID | No |
| Name | Varchar (45) | Last, First Name | Yes |
| Office | Varchar (10) | Office number | Yes |
| DateofBirth | Date | Age of Professor | Yes |

COURSE

There is one row in the COURSE table for each course offered that quarter.

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Meaning | NULL allowed |
| CourseNum | Integer | Course number | No |
| DeptName | Varchar(45) | Name of Department | No |
| CourseName | Varchar(45) | Course Name | No |
| ClassRoom | DateTime | Room Number | Yes |
| Enrollment | Integer | Number of Students Enrolled | Yes |

DEPARTMENT

There is one row in the DEPARTMENTS table for each department in the University.

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Meaning | NULL allowed |
| DeptName | Varchar (45) | Name of Department | No |
| ChairmanID | Varchar (45) | Name of Chairman | Yes |

Prereq

There is one row in the Prereq table for each pre-requite for any course.

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Meaning | NULL allowed |
| CourseNum | Integer | Course Number | No |
| DeptName | Varchar (45) | Name of Department | No |
| PreReqNumber | Integer | Pre requisite Number | Yes |
| PreReqDeptName | Varchar (45) | Pre requisite Dept Name | Yes |

Teach

There is a row in the TEACH table for each course taught by a professor.

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Meaning | NULL allowed |
| PID | Varchar(10) | Professor who teaches the course | No |
| CourseNum | Integer | The course taught | No |
| DeptName | Varchar (45) | The department the course taught | No |

Takes

There is a row in the TAKE table for each course enrolled by a student.

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Meaning | NULL allowed |
| SID | Varchar (10) | ID of student taking a course | No |
| CourseNum | Integer | The course take by student | No |
| DeptName | Varchar (45) | The department of course | No |
| Grade | Decimal(4,2) | The grade obtained | Yes |
| ProfessorEval | Decimal(4,2) | The professor evaluation | Yes |

**Exercise:** (c) Use the DLL commands of SQL to create the resulting schemas

**KEY:**

create database newdbW22;

/\* drop database newdbW22;\*/

**/\* the "USE" statement: to let the workbench know which DB**

**you intend to use\*/**

use newdbW22;

**/\* Table with not null and primary key constraint \*/**

CREATE TABLE STUDENT (

SID varchar(10) not null PRIMARY KEY,

SNAME varchar (45),

SADDRESS varchar(45));

**/\* Table with not null and primary key constraint specified differently \*/**

create table PROFESSOR (

PID varchar(10) not null,

PName varchar(45),

Office varchar(45),

Age Int,

primary key (PID));

create table DEPARTMENT (

DeptName varchar(45) not null,

ChairmanID varchar(45),

primary key(`DeptName`)

);

**/\* Alter existing table \*/**

ALTER TABLE PROFESSOR CHANGE COLUMN Age DataOfBirth DATE NULL;

ALTER TABLE PROFESSOR ADD NewDate DATE;

ALTER TABLE PROFESSOR DROP NewDate;

**/\* Table with not null and more than one primary key constraints \*/**

create table COURSE (

CourseNum INTEGER not null,

DeptName varchar(45) not null,

CourseName varchar(45),

ClassRoom DateTime,

Enrollment INT,

primary key(`CourseNum`,`DeptName`)

);

**RELATIONSHIP TABLES:**

CREATE TABLE PreReq(

CourseNum Integer not null,

DeptName varchar(45),

PreReqNumber INT,

PreReqDeptName varchar(45),

primary key(`CourseNum`,`DeptName`)

);

**/\* Drop a Table \*/**

Drop table PreReq;

**/\* Tables with not null and foreign key constraint \*/**

CREATE TABLE PreReq(

CourseNum Integer not null,

DeptName varchar(45),

PreReqNumber INT,

PreReqDeptName varchar(45),

FOREIGN KEY (CourseNum) REFERENCES COURSE (CourseNum),

FOREIGN KEY (DeptName) REFERENCES DEPARTMENT (DeptName)

);

CREATE TABLE Teach (

PID VARCHAR(10),

CourseNum INT,

DeptName VARCHAR(45),

FOREIGN KEY (PID) REFERENCES PROFESSOR (PID),

FOREIGN KEY (CourseNum) REFERENCES COURSE (CourseNum),

FOREIGN KEY (DeptName) REFERENCES DEPARTMENT (DeptName)

);

CREATE TABLE Take (

SID VARCHAR(10),

CourseNum INT,

DeptName VARCHAR(45),

Grade Decimal(4,2),

ProfessorEval int,

FOREIGN KEY (SID) REFERENCES STUDENT (SID),

FOREIGN KEY (CourseNum) REFERENCES COURSE (CourseNum),

FOREIGN KEY (DeptName) REFERENCES DEPARTMENT (DeptName)

)