Schema Definitions. Overview.

Registered Users

This contains the common fields for Students and Lecturers such as his email, Password (encrypted of course),:date of birth and Registration:date.

Students

Information specifically pertaining to a student such as his professional Profile Link and Type = Student

Universities

Information pertaining to a University which acts as supplier in our case.

Lecturers

Information pertaining to a Lecturer such as his Parent University

Certificates

Information pertaining to a certificate such as Certification:date, Grade, Type of Certification, Affiliation.

Course Schedule

Course Name, Offered by which Lecturer and Start and End:date for the courses.

Certification

Specific information pertaining to a given instance of certification. Like which professor has certified which Student in which course and other certificate details which are offered by the Certificates Table.

Schemas and Normalizations.

Sechemas

registered_user(regid:integer, rname:string, dob:date, email:string, pass:string, regdate:date);

students(sid:integer, type:string, profile:string, areas);

lecturers(lid:integer, uname:string, uwebpage:string, areas:string);

certificates(certid:integer, ctype:string, cgrade:string, clevel:integer);

courseschedule(cid:integer, fromdate:date, enddate:date);

certification(certindex:integer, sid:integer, cid:integer, issuedate:date, certid:integer);

In the above Schema definitions,

i. The name for the registered user can contain multiple fields namely First Name, Middle Name, and Last Name. The advantage for having this division between the names is
   a. The communication to the user looks friendlier when he is referred to by his first name.
   b. The certification can be watermarked with the user's exact name for future authentication regarding the validity of the certification.

Thus the name is split into firstName, middleName, lastName

ii. A student can have multiple professional profiles on different sites. Hence it can be a multivalued value. Currently the scope is limited to having a maximum of three profile creating websites to be incorporated. In case of more than 3 profiles on different providers, the user has to manually upload a copy of the certification to his profile to the other profiles.

Thus profile is split into profile1, profile2, profile3.

iii. A Professor and student can have multiple areas of Expertise and interests. In our scope we are limiting this to 3 areas of interest.

Thus areas in both these tables can be split into area0, area1, area3

After 1NF Normalizations, the resulting schema looks like

registered_user(regid:integer, firstName:string, middleName:string, lastName:string, dob:date, email:string, pass:string, regdate:date);

students(sid:integer, type:string, profile0:string, profile1:string, profile2:string, area0:string, area1:string, area2:string);

lecturers(lid:integer, uname:string, uwebpage:string, area0:string, area1:string, area2:string);

certificates(certid:integer, ctype:string, cgrade:string, clevel:integer);

courseschedule(cid:integer, fromdate:date, enddate:date);

certification(certindex:integer, sid:integer, cid:integer, issuedate:date, certid:integer);

2 NF Normalizations:

Applying 2 NF normalization rules to the resulting schema, we have the following observations

i. In the Lecturers table, the schema has university name and the university webpage for every lecturers' entry. Thus a non-prime key is dependent on a part of the candidate key. Thus the table is not in 2NF form. Evidently this can lead to redundancies in the database since a given university will always have the same page. So this table is split into two tables, one for universities and one for lecturers

The resulting schema after applying 2NF normalization is

registered_user(regid:integer, firstName:string, middleName:string, lastName:string, dob:date, email:string, pass:string, regdate:date);

students(sid:integer, type:string, profile0:string, profile1:string, profile2:string, area0:string, area1:string, area2:string);

universities(uname:string, uwebpage:string)

lecturers(lid:integer, uname:string, area0:string, area1:string, area2:string);

certificates(certid:integer, ctype:string, cgrade:string, clevel:integer);

courseschedule(cid:integer, fromdate:date, enddate:date);

certification(certindex:integer, sid:integer, cid:integer, issuedate:date, certid:integer);

3 NF normalization:

All the above schemas are already in the 3NF.

i. They are in 2NF by the normalizations above.
ii. All attributes are only determined by the candidate keys in each table and not by other non-prime attributes.

SQL Statements for the Schema above

```
create table courses(cid int(20), name varchar(20), primary key(cid));

create table registered_user(regid int(20), firstname varchar(30), middlename varchar(30), lastname varchar(30), dob date, email varchar(20), pass varchar(20), regdate date, primary key(regid), unique(email));

create table students(sid int(20), type varchar(20), profile0 varchar(512), profile1 varchar(512), profile2 varchar(512), area0 varchar(40), area1 varchar(40), area2 varchar(40), primary key(sid), foreign key(sid) references registered_user(regid) on delete cascade);

create table universities(uname varchar(20) not null, uwebpage varchar(512), primary key(uname));
```

```
create table lecturers(lid int(20), type varchar(20), uname
varchar(20), area0 varchar(40), area1 varchar(40), area2 varchar(40),
primary key(lid), foreign key(lid) references registered_user(regid)
on delete cascade, foreign key(uname) references universities(uname)
on delete no action);

create table certificates(certid int(20), ctype varchar(20), cgrade
varchar(5), clevel int(20), primary key(certid));

create table courseschedule(cid int(20), fromdate date, enddate date,
foreign key(cid) references courses(cid) on delete cascade, primary
key(cid));

create table certification(certindex int(20), sid int(20), cid
int(20), issuedate date, certid int(20), foreign key(sid) references
students(sid),foreign key(cid) references courses(cid), foreign
key(certid) references certificates(certid), primary key(certindex));
```

Populating data in the database.