

CS 7700-02

ADVANCED DATABASE SYSTEMS

SPRING 2015

PROFESSOR: GUOZHU DONG

PROJECT: UNIVERSITY COURSE REGISTRATION SYSTEM

Name: RAJASHEKAR SUGANDAM UID: U00750675

“I have neither given nor received aid on this assignment, nor have I observed any violation of the Honor code”

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Date: 04/23/2015 Report due date: 04/23/2015

**TABLE OF CONTENTS**

**DESCRIPTION OF PROJECT………………………………………………………..**

**DATABASE SCHEMA …………………………………………………………………**

**DATA REQUIREMENTS……………………………………………………………..**

**DESIGN CONSIDERATION………………………………………………………….**

**BUILDING ER MODEL………………………………………………………………**

**ENTITIES AND THEIR RELATIONSHIPS………………………………………..**

**ER DIAGRAM OF THE DATABASE……………………………………………….**

**TABLE STRUCTURES……………………………………………………………….**

**CONTENTS OF DATA TABLES…………………………………………………….**

**TRANSACTIONS……………………………………………………………………..**

**RELATIONSHIP VIEW IN SQL……………………………………………………**

**TRANSACTION RESULTS………………………………………………………….**

**OLAP ANALYSIS…………………………………………………………………….**

**CONSISTENCY OF THE DATABASE…………………………………………….**

**EXPERIENCE………………………………………………………………………..**

**DATABASE DESIGN………………………………………………………………..**

**IMPLEMENTATION OF PROJECT………………………………………………**

**CODE PART…………………………………………………………………………..**

**CONCLUSION………………………………………………………………………..**

**DESCRIPTION OF PROJECT**

The main objective of this project is UNIVERSITY COURSE REGISTRATION MANAGEMENT SYSTEM where it keeps track of all the Student details, Professor Details and the Courses offered by the university. It also stores the information of the courses registered by an each student and the courses taught by each professor for a particular semester. It also stores the registration transactions which are helpful for future analysis. This Course Registration System is aimed to make easier and more convenient the Course Registration Process, a bother through which every students goes through every semester.

Course Registration System provides a way to search for courses without opening the course catalog by logging into his own student account provided by the university where student can view for possible schedules and finally can register for the courses he is interested in which assures privacy for student. In this system each professor can view the list of students and their performance who has registered for his particular course.

The main purpose of this project is to design a database for real time problem based on ER schema model where we insert some sample tuples in the records of all tables and embedding some SQL queries in java programming language to carry out the transaction. At the end we need to check how well the application respond to database when there are transactions occurred.

**DATABASE SCHEMA**

**Problem Statement:** In this design the database system should keep track of users, student, professor, courses, and their course registration details. In order to remind our project consists of only two types of Users (Students, Professors). So every time when a new user is registered by entering his Login details and also specifying his position will be stored to User’s table and then it will be updated to the concerned table (Student or Professor). So when a particular User is logged in it will be redirected to the concerned table depending upon the position stored in the User table then there the personal information of that user will be displayed and the actions he can perform which is depend on the type of user he is. So we need to develop a java application in order to check the transaction occur when a particular user is logged in.

**DATA REQUIREMENTS**

The Software should capture the following information:

1. Each User who is newly registered should be linked to type of User concerned tables i.e. either Student Table or Professor Table.
2. Each User has personal information stored in its table which is distinguished from other User by his User Identification number (UID).
3. Each Student who is logged in will be displayed with his personal information and the Courses registered for that particular semester.
4. Each Student can contain more than one course registered for a particular semester.
5. Each Student can have a chance of adding or dropping of courses.
6. Each Professor when logged in will be redirected to their home page with personal details displaying and in advance we can make visible the list of courses he is teaching this semester but as we are talking about course registration system we need not to consider

**DESIGN CONSIDERATION**

After looking at the project requirements, I decided Relational Database Management System (RDBMS) would be the best solution for my implementation. MYSQL has been chosen to develop the backend database system as it is cost effective and is most widely used RDBMS. And my front end implementation is done using Java Programming Language. I have chosen some transaction to be executed using front end.

**BUILDING ER MODEL**

An Entity-Relationship model (ER Model) is a data model for describing the data or the information aspects of its process requirements. It is essentially meant for human comprehension and creating a logical schema of the database. It is used in Conceptual Modelling of the Database schema. This is a high-level database design without the implementation details and it is also DBMS independent. This shows how different parts of the data model interact with one another. ER Diagrams provide a very useful framework for creating and manipulating databases and also can be transformed into relational tables which can be used to quickly build databases. ER Model diagrams uses symbols to mainly represent three different types of information.

1. Boxes which are commonly used to represent entities.
2. Diamonds are used to represent relationships.
3. Ovals are used to represent attributes present in entities and relationships.

In a Database recall that we have to define all the attributes first then the relation schemas, then the relations and then the Primary key. These all should be in order.

**ENTITIES AND THEIR RELATIONSHIPS:**

1. **Register:**

COURSES

STUDENT

Registered

1 N

Students and Courses entities hold One to Many relationship. It means one student can register for more than one course and also one course can be registered by more than one student where it holds Many to One relationship.

1. **Teaches:**

PROFESSOR

Teaches

COURSE

1 1

Course and Professor Entities hold One to One relationship. It means one professor can teach only one course.

PROFESSOR

COURSES

Teaches

1 N

Professor and Courses Entities hold One to Many relationship. It means one professor can teaches more one course.

1. **BelongsTo:**

DEPARTMENT

STUDENTS

BelongsTo

N 1

Students and Department holds Many to One relationship. Because Many students can belongs to One department.

1. **Employed:**

PROFESSORS

Employed

DEPARTMENT

1 N

Department and Professor holds One to Many relationship. It means each department employees multiple professors.

1. **Offered:**

Offered

COURSES

DEPARTMENT

1 1 N

Department and Courses holds One to Many relationship. It means each department offers multiple courses.

**ER DIAGRAM OF THE DATABASE:**

USERS

Belongs To

T

(1, 1) (1, 1)

Belongs To

T

(1, 1) (1, 1)

Employed

T

PROFESSOR

STUDENT

Teaches

T

Registered

T

COURSES

DEPARTMENT

(1, N) (1, N) (1, 1)

(1, N) (1, 1) (1, N)

(1, 1)

Offered

T

(1, N)

**TABLE STRUCTURES:**

**Users:**

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Comments |
| uid | INT (15) | Primary key, Unique for every User |
| email | VARCHAR (50) | Email of the users |
| password | VARCHAR (15) | Password of the user |
| name | TEXT | Name of the User |
| DOB | DATE | Date of birth of the user |
| position | TEXT | Position of the user |

**Students:**

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Comments |
| uid | INT (10) | Primary key, Unique for student |
| name | VARCHAR (20) | Name of the student |
| Deptno | INT (15) | Foreign key Reference students (uid) |
| email | VARCHAR (40) | Email of student |
| courses | BLOB | Courses registered by student |
| CrHrsRegistered | INT (10) | Total number of course hours registered by student |

**Professors:**

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Comments |
| uid | INT (10) | Primary key, Unique for professor |
| name | VARCHAR (20) | Name of the professor |
| Dno | INT (15) | Foreign key Reference department(Dno) |
| email | VARCHAR (20) | Email of the professor |
| Cid | INT (10) | Foreign key Reference courses(Cid) |
| CrHrsUndertook | INT (10) | Total number of course hours teaching by professor |
| salary | INT (25) | Total salary he earn |
| state | VARCHAR (15) | Name of the state he belongs |

**Courses:**

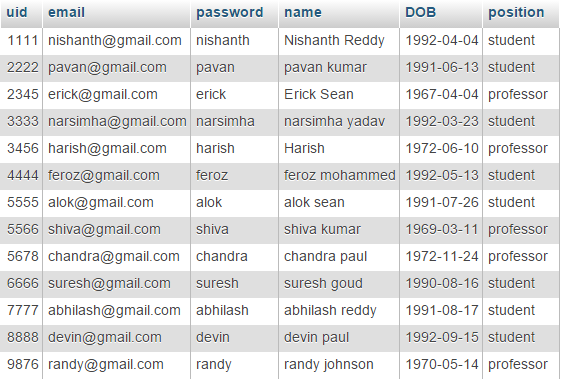
|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Comments |
| Cid | INT (15) | Primary key, Unique for course |
| Cname | VARCHAR (30) | Name of the course |
| PID | INT (20) | Foreign key Reference professors(uid) |
| Description | TEXT | Description of the course |
| CrHrs | INT (15) | Credit hours for a course |
| Term | VARCHAR (25) | Term to which it belongs |

**Departments:**

|  |  |  |
| --- | --- | --- |
| Column Name | Data Type | Comments |
| Dno | INT (10) | Primary key, Unique for each department |
| Dname | TEXT | Name of the department |
| Doffice | VARCHAR (25) | Department Office |
| DHead | TEXT | Name of head of the department |

**CONTENTS OF DATA TABLES:**

**Users:** (Only few records have been displayed)



**Students:** (Only few records have been displayed)

****

**BLOB:** (Binary Language object) here when a student registers for multiple courses we need to create a registration table storing all the courses registered by every student which will lead Data Overriding. In order to avoid that we use BLOB type where it can store multiple values in it. Implementation part should be taken care in code which is as follows,

**Code for BLOB:**

Blob customObject = null;

CoursesRegistered obj=courses;

Object customObjectdata = courses;

/\* creating an object\*/

ByteArrayOutputStream baos = new ByteArrayOutputStream();

ObjectOutputStream objOstream = new ObjectOutputStream(baos);

objOstream.writeObject(customObjectdata);

/\* passing the selected courses into the list\*/

objOstream.flush();

objOstream.close();

byte[] bArray = baos.toByteArray();

ObjectInputStream ois = null;

ByteArrayInputStream baip = null;

ByteArrayInputStream bais = new ByteArrayInputStream(bArray);

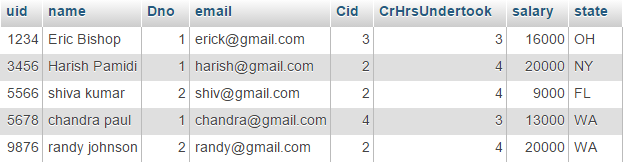
/\* Storing the entered courses in an ArrayList\*/

pstmt.setBinaryStream(1, bais, bArray.length);

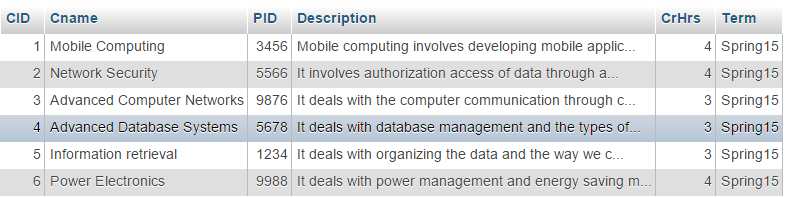
pstmt.setInt(2, totalCreditHours);

pstmt.setInt(3, uid);

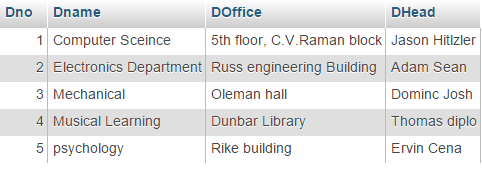
**Professors:** (Only few records have been displayed)



**Courses:** (Only few records have been displayed)



**Departments:** (Only few records have been displayed)

****

**TRANSACTIONS:**

In this application we use MySQL Database as backend and Java as the Programming Language to process all the Transactions done.

1. **Inserting Values to the Artist Table:**

**Use:** This transaction inserts new tuples into the respective Users table of the database.

**Functionality:** In a Particular University Users (Students, Professors) keeps on add day by day. In order to store their information they will be registered as New Users by providing all the necessary information in the Registration form and then will be inserted into User table there by updated to concerned position (Students, Professors) tables.

1. **Retrieve Information from User Table:**

**Use:** In this transaction it extracts the necessary information from the User table when a particular user is logged in.

**Functionality:** In here when a particular User is logged in then the data Uid, Password will be compared in user table and if is correct then will be linked to the concerned Student or Professor tables and the information will be retrieved.

1. **Deleting Information from Student Table:**

**Use:** In this transaction it deletes the courses when a student want to drop his registered courses.

**Functionality:**  In this process when a student logged in and drops his registered courses then the course data in the BLOB type of student will be deleted.

**RELATIONSHIP VIEW IN SQL:**

**Users Table**

Uid

Email

Password

Name

DOB

Position

**Students**

Uid

Name

Deptno

Email

Courses

CrHrs

**Departments**

Dno

Dname

DOffice

DHead

**Professors**

Uid

name

Dno

email

Cid

CrHrsUndertok

Salary

state

**Courses**

Cid

Cname

PID

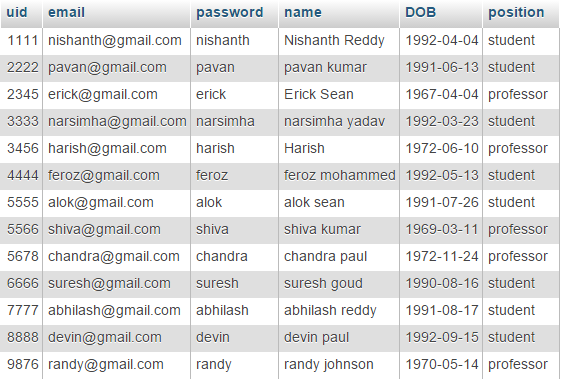
Description

CrHrs

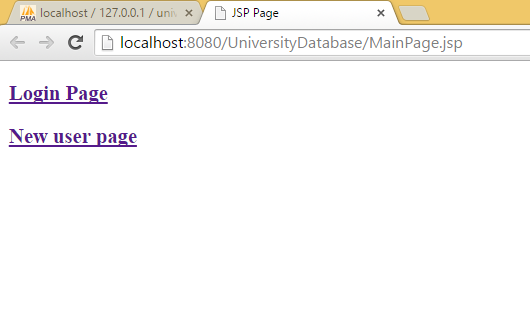
Term

**TRANSACTION RESULTS:**

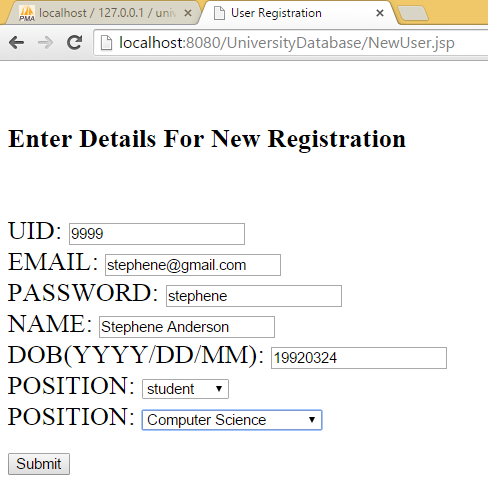
1. Initially the User table is as follows,



So, when a new User is registered then it will be stored in the User table. For Example let’s run my projects and it looks like this,

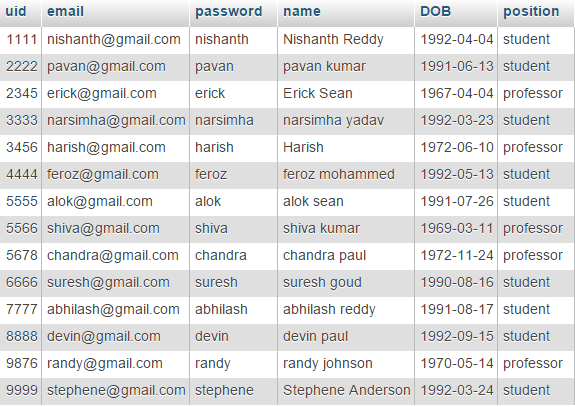
****

Now let’s select the option **New user page** then it redirects to its page and it looks like the one below,

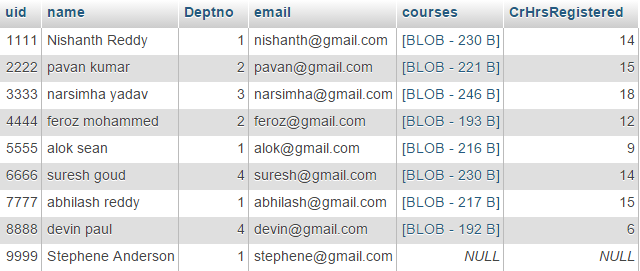


When you click on submit the registration will be completed and the new user will be **INSERTED** into user table and then will be **UPDATED** to concerned student table because the position selected is student. It looks like this,

**Inserted in User table:**  Inserted row can be seen at the bottom.

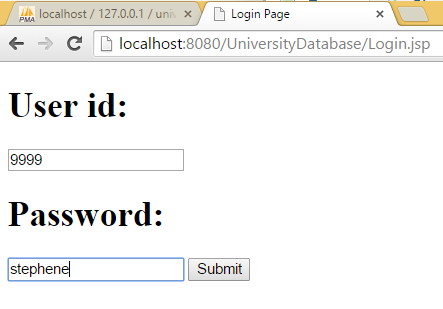


**Updated in student table:** Updated row can be seen at the bottom.

****

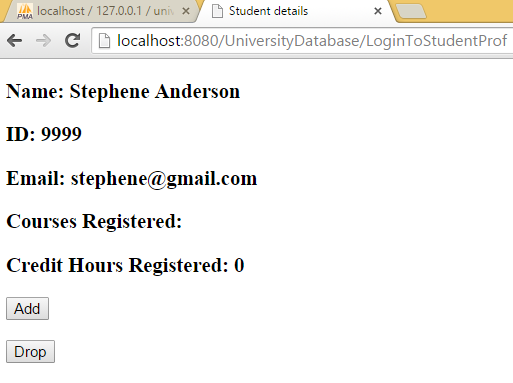
And now if the new registered user here it is student, want to register for some courses then the student has to login into the user account using UID, and PASSWORD where the student personal information will be displayed where student can VIEW, ADD, DROP. So, when clicked on login it will be directed to,

**LOGIN PAGE:**



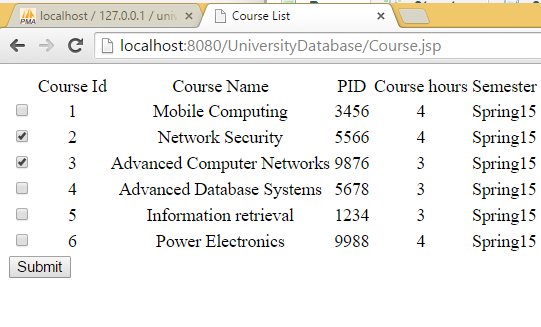
When it is submitted it will check for correctness of login details in the user table. If it is correct then we will be displayed with this home page which looks like this,

**Home Page:**

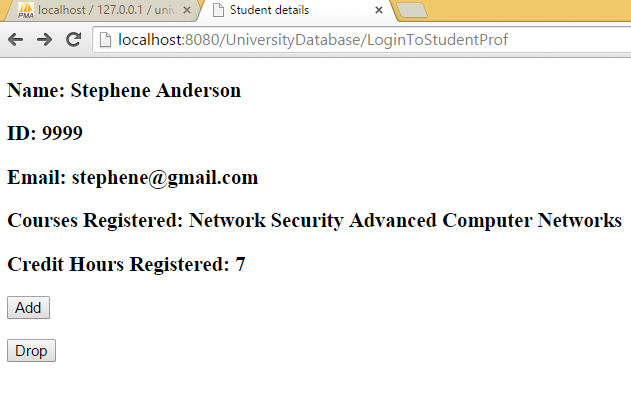


And now if the student want to add any courses click on ADD button it will be redirected to a page with all courses in a list. It looks like the one below,

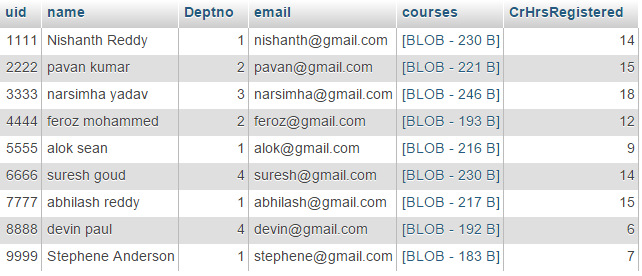
**Course Page:**

****

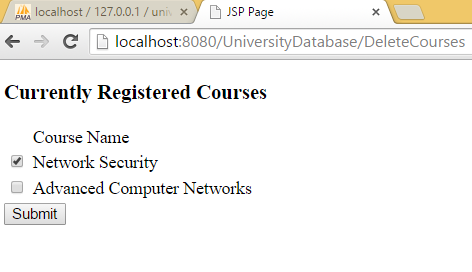
Now student selected courses with Course Id 2 & 3 then the student course table will be INSERTED with the selected course which is as follows,



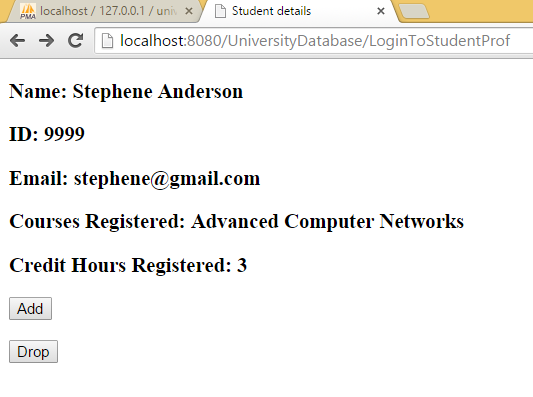
**Student Table:**



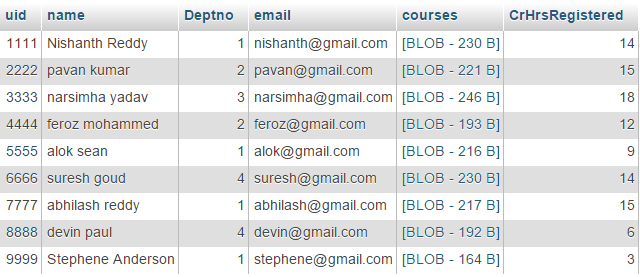
For suppose if the student want to drop any course then student will click on drop button then it directs to the following page,



Then the Network security course will be DELETED from course list of this student then the student row will be updated as follows,

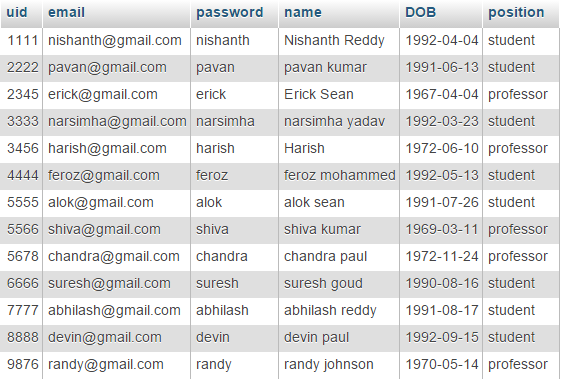


**Student Table:** (After deleting the course with Credit hours updation)

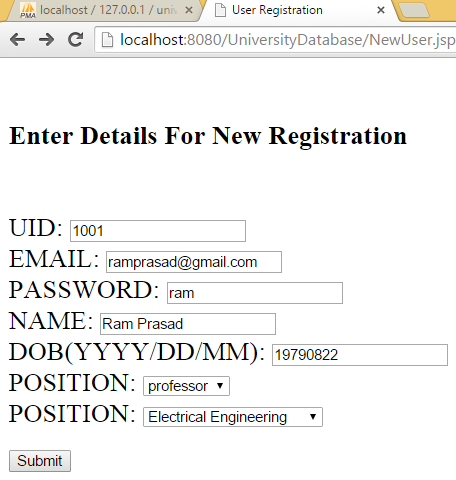
****

Now in case of professor when a new professor registered which is as follows, Initial User table is

**User table:**

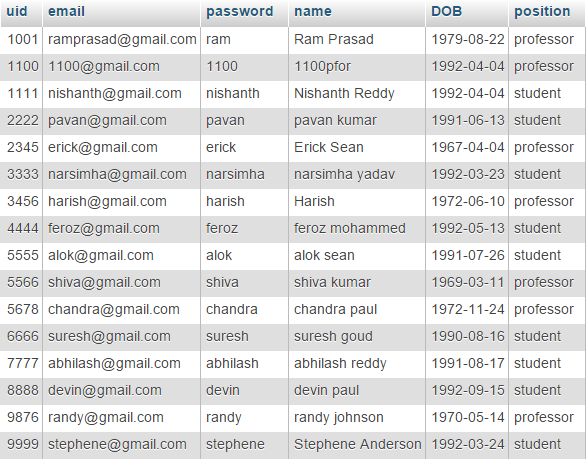


**Registration Form:**

****

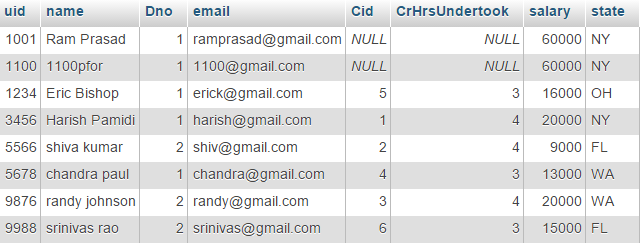
It will be **Inserted** into User table as show below,

**User table:**

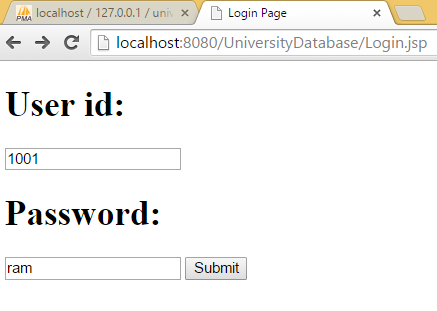


And the professor table will be UPDATED as follows,

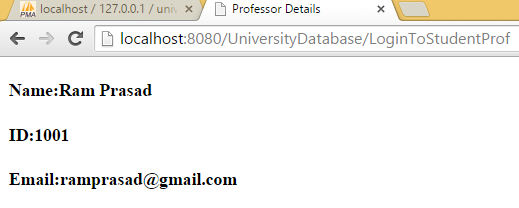
**Professor table:**



So when professor login as shown,



**Professor Home Page:**



**OLAP ANALYSIS:**



**Query 1: (RollUp)**

SELECT S.uid,U.name,D.Dno,COUNT(S.name) from students S, departments D, users U where S.CrHrsRegistered>12 and S.uid = U.uid and S.Deptno = D.Dno group by D.Dno,S.uid,U.name with rollup

union

SELECT S.uid,U.name,D.Dno,COUNT(S.name) from students S, departments D, users U where S.CrHrsRegistered>12 and S.uid = U.uid and S.Deptno = D.Dno group by S.uid,D.Dno,U.name with rollup

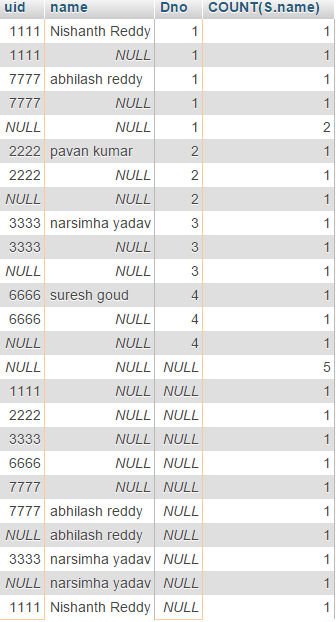
union

SELECT S.uid,U.name,D.Dno,COUNT(S.name) from students S, departments D, users U where S.CrHrsRegistered>12 and S.uid = U.uid and S.Deptno = D.Dno group by U.name,S.uid,D.Dno with rollup

union

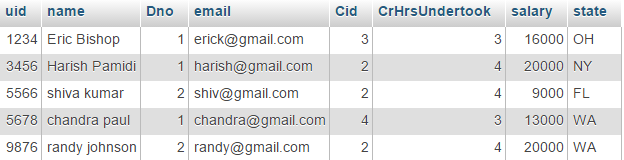
SELECT S.uid,U.name,D.Dno,COUNT(S.name) from students S, departments D, users U where S.CrHrsRegistered>12 and S.uid = U.uid and S.Deptno = D.Dno group by U.name,S.uid,D.Dno with rollup

**Result:**



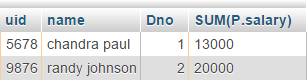
As, my version of MYSQL does not support CUBE. So, I just modified my query which is specified above where students from different departments with Total number of Course Hours registered for a particular semester greater than 12 has been filtered using GROUP BY CLAUSE with ROLLUP embedded in it. Using this result which will be useful in analyzing their result when register beyond the limit.

**Query 2:** (Slicing)

****

select P.uid, U.name, D.Dno, SUM(P.salary) from professors P, departments D, users U where P.uid = U.uid and P.Dno = D.Dno and P.salary>10000 and P.state = 'WA' group by P.uid, U.name, D.Dno

**Result:**

****

**CONSISTENCY OF THE DATABASE:**

In order to perform any Transactions on a Database, the data present in the database should satisfy the following conditions which confirms the consistency of database. They are as follows:

**First Normal Form:**

In the above Relational Schema all the attributes are single valued attributes. So every relation in the relational schema satisfies the First Normal Form.

**Second Normal Form:**

In the above Relational Schema all the attributes in every relation are dependent only on the primary key of that relation. Therefore all the relations in the relation schema satisfies the Second Normal Form.

**Third Normal Form:**

In the above Relational Schema No attribute in all the relations transitively dependent on any other attribute. Therefore all the relations present in the Relational Schema satisfies the Third Normal Form.

As the data present in the Database has satisfied all the Three Normal Forms mentioned above, it is also said that the data is also in Boyce-Codd Normal Form (BCNF). As it has satisfied all the conditions of the Normal Forms now Transactions can be performed on the Database.

**EXPERIENCE:**

While designing the database I have come across new concepts such as BLOB (Binary Language Object) which made a part of my implementation easier, OLAP (Online Analytical Processing) which is used for reporting on current data and learned a lot with respective to MYSQL.

**DATABASE DESIGN:**

According to my design, I initially started by making several modules of my project where each module performs a task such as Database creation, JDBC-ODBC connection, and coding part for the front end etc. I have spent more time in carefully implementing the Relational Database Schema because it can avoid redundancy of data. After gathering all the information which I need I followed the bottom up approach to design the relational schema by applying the Normal Forms one by one to normalize raw attribute list.

**IMPLEMNTATION OF PROJECT:**

The University Course Registration System was implemented successfully using Apache Server and MYSQL running as the backend database and JAVA used as the server side language. Performance execution time, efficiency, and memory usage are noted after implementation for comparison purpose.

**CODING PART:**

The application Java code for my project is zipped and saved separately with a Read.txt file which consists of instructions on how to run the project.

**CONCLUSION:**

The project which I implemented is efficient because the time for loading application University Database done in 5,798 ms which is quiet considerable. This implementation is easier for a student to register for his courses with privacy. In advance this project can be worked further on creating an application by considering some advanced transactions which will be very much helpful for the University in meeting the day to day operations of the users.