Course Rank System

Dharma Teja Bankuru  Sai Kiran Katuri  Sairam Bade
130050049  130050051  130050079

October 19, 2015

Detailed Design Document
Database and Information Systems
Dept. of Computer Science
IIT BOMBAY
1 Introduction

• Course Rank is an interface where students can review courses and plan their academic programme by accessing official university information and previous years’ grading statistics.

• Students can comment under a course asking for some more information about the course and instructors or other students who did the course can reply to it.

• Students who already took the course can rate the course and review it. A professor teaching a course can update or modify its course content accordingly during his teaching period.

• In this report we will give the detailed explanation of how will we approach to build the desired application. Next we will show the ER diagram of the database we are using.

• Then we will try to explain the tables, integrity constraints etc., and also display the schema diagram of our database which we got from ER diagram. This will also give you the idea regarding the integrity constraints, tables being used in schema and authentication provided to different roles.

2 Approach for building the application and tools

We first started out by discussing upon different roles and screens of the application. Then we started drawing the ER diagram followed by schema diagram of database. Then we started thinking about different test cases and wrote sql scripts for the same. Please find the scripts for creating tables and dropping them, and inserting departments, admins etc., in the following links.

http://www.cse.iitb.ac.in/~sai.kiran/CourseRank/

In the earlier software requirement specification document we mentioned that we are planning to use some tools. No changes are made to them and we started coding using the same set of tools. Hence, We are using postgresql database for storing the data. We are using JDBC to make this application. We using eclipse IDE that we learnt in our labs.
3 Planned data for testing and Expected reports

The main entity of Course Rank is Course. Hence, we are planning to use the IIT Bombay courses database itself for testing itself. It contains about 6000 courses in total. Hence, it is also a challenge for us to manage such a huge data because in our case each course contains a lot of attributes. Please find the courses and their description at IIT Bombay in the link below.
http://iitb.ac.in/asc/Courses

For users, of IIT Bombay, we are incorporating LDAP login into our system. We are using the existent LDAP access jsp code for querying IIT Bombay Ldap server. Once, the user logs in with his Ldap login credentials, we store his details in people table of our database. We guarantee that we neither store his password in our database nor access it. Even instructors can also login using their Ldap credentials, and in this case instructors details will be added to people table and instructor table. For spectators, we are planning to ask our friends to sign-up. We will put the application online(locally on some private hostel IP), and ask our friends to access the application and put comments on different courses. Since, our application involves a lot of data with about 6000 courses and a lot of comments taking place per second, we are planning to report the performance of database by putting different indexes intuitively. We want to measure improvements in performance at end with these indexes. Being a huge database system, we are expecting to get good results to document. We will put this in the final report submission.

4 Indexes

We used indexing to quickly locate data without having to search every row in a database table every time a database table is accessed. The bottleneck to our application is the transactions of comments, reviews, and courses. In case of courses we always access them by department names. In case of reviews and comments we will access them by time they are created. Hence, we created indexes in these as follows:-

\[
\begin{align*}
\text{CREATE INDEX course_index ON course (course_id, dept_name);} \\
\text{CREATE INDEX comment_index ON comment (comment_id, course_id, created_at);} \\
\text{CREATE INDEX reply_index ON reply (comment_id, created_at);} \\
\text{CREATE INDEX review_index ON review (course_id, given_at);} \\
\end{align*}
\]
5 Access Controls

There are four types of users to this application namely Admin, Instructors, Students and Spectators. These people are given access to different tables in the database through the application based on their designations.

ADMIN:
Admin can add new courses or delete existing courses from the course structure and approves the requests of students or instructors. He also updates the timetable of the institute every semester based on instructor assignment to courses for teaching and registration done by students. Thus, indirectly he has access to all the tables in the database.

INSTRUCTOR:
Instructor teaches some courses (stored in teaches table of database updated by admin of institute) and he can modify the content of his courses and can also reply to the queries posted by students under the course. Thus he has access to mod_content table under some constraints. He don’t have any access to course and org_content tables.

STUDENT:
Students can view the courses and post their queries regarding courses below the course description page. He/She can also review and rate the course after completing the course in previous offerings.

SPECTATOR:
Spectators are people not from institute who has to sign up in order to view courses. After signing up, they can view the contents of the course and comment on them if needed. He don’t have access to course, content tables in the database. All except admins will not have any access to people table except signing up to add to it.

6 Tables in CourseRank database

<table>
<thead>
<tr>
<th>TableName(Attribute1, Attribute2, ...)</th>
</tr>
</thead>
<tbody>
<tr>
<td>people(ID, password, name, address, admin_ins_std_spec,email)</td>
</tr>
<tr>
<td>admin:(ID, qualification)</td>
</tr>
<tr>
<td>student(ID, dept_name, tot_credits)</td>
</tr>
</tbody>
</table>
instructor(ID, dept_name, salary)
spectator(ID, institute_name, degree)
department(dept_name, building, budget)
course(course_id, name, dept_name, credits, course_type, half_full_sem)
prereq(course_id, prereq_id)
takes(ID, course_id, sec_id, semester, year, grade)
teaches(ID, sec_id, course_id, semester, year)
section(sec_id, semester, course_id, year, building, room_number, time_slot_id, day, start_time)
classroom(building, room_number, capacity)
time_slot(time_slot_id, day, start_time, end_time)
org_content(course_id, org_lec_credits, org_tut_credits, org_practical_credits, 
          org_self_study_credits, org_text_ref, org_description)
mod_content(course_id, mod_lec_credits, mod_tut_credits, mod_practical_credits, 
            mod_self_study_credits, mod_text_ref, mod_description, ID, modified_at)
statistics(course_id, sec_id, semester, year, tot_students, grade_stats)
review(ID, course_id, review, rating, difficulty_level, given_at)
comment(comment_id, ID, course_id, comment, created_at)
reply(comment_id, ID, course_id, comment, created_at)

7 Database Design and integrity constraints

people:
people(ID, password, name, address, admin_ins_std_spec, email)
   Everybody is considered as people including students, Admin, Instructors and students from other institutions. Students from our institute and instructors are given ID and password by the Admin while students from other institutes called spectators need to signup. ID is the primary key and so it is unique. Every other attribute should have some value i.e. they should be not null.

admin:
admin(ID, qualification)
   These table contains all the admins along with their qualifications. Every record in Admin table is also a record in table people. So every ID present in the admin table should be present in people table.
student:
    student(ID, dept_name, tot_credits)
    This table contains all the students from this institution added by admin. They are given ID’s and password by admin. Every ID present in this table should be present in table people. dept_name should be one of departments present in the institute.

instructor:
    instructor(ID, dept_name, salary)
    This table contains all the instructors from this institution added by admin. salary and dept_name is also stored along with it. dept_name should be one of departments present in the institute. Every ID present in this table should be present in table people.

spectator:
    spectator(ID, institute_name, degree)
    This table contains people not belonging to this institute along with their institute_name and degree. Spectators need to sign up by filling their details like email, address, ID and password. ID is stored along with institute_name and degree. Every ID present in this table should be present in table people.

department:
    department(dept_name, building, budget)
    This table contains all the departments present in the institute along with department building and it’s budget.

course:
    course(course_id, name, dept_name, credits, course_type, half_full_sem)
    This table contains all the courses offered in the institute along with number of credits, the type of course(either theory or lab) and whether the course is offered full semester or half semester. dept_name present in this table should be one of the departments offered in the institute.

prereq:
    prereq(course_id, prereq_id)
    This table contains all the prerequisites of a given course.

time_slot:
    time_slot(time_slot_id, day, start_time, end_time)
    This table contains different time slots along with their starting and end
time.day attribute present in the table should be one of weekdays. The triplet time_slot_id, day, start_time acts as primary key.

classroom:
    classroom(building, room_number, capacity)
    This table contains all the classrooms present in different buildings in institute.

section:
    section(sec_id, course_id, semester, year, building, room_number, time_slot_id, day, start_time)
    This table contains all the sections with courses which are offered in a particular semester along with the class where it is offered and time_slot when it is offered. The triplet time_slot_id, day, start_time should be present in table time_slot. semester should one of autumn, spring, summer.

takes:
    takes(ID, sec_id, course_id, semester, year, grade)
    This table gives the information about student taking a course in a semester and year along with the grade he/she obtained in that course. The course_id along sec_id, semester and year should be present in section table. ID in this table should be present in student table.

teaches:
    teaches(ID, sec_id, course_id, semester, year)
    This table gives information about all the courses an instructor is teaching in a given semester and year. The course_id along sec_id, semester and year should be present in section table. ID in this table should be present in instructor table.

org_content:
    org_content(course_id, org_lec_credits, org_tut_credits, org_practical_credits, org_self_study_credits, org_text_ref, org_description)
    This contains original information of a given course.

mod_content:
    mod_content(course_id, mod_lec_credits, mod_tut_credits, mod_practical_credits, mod_self_study_credits, mod_text_ref, mod_description, ID, modified_at)
    This contains all the modified or updated information about a course and ID of the person who modified it and time at last it is modified. The default value of all the attributes present in this table is null except from ID which
should be one of the ID’s from instructor table. From the table tuple with earliest time is given to be posted on the course page.

---

**statistics:**

statistics(course_id, semester, year, tot_students, grade_stats)

This table contains the number of students who obtained a particular grade obtained in a course in a given semester and year. grade_stats is an array containing the number of students who obtained different grades.

---

**review:**

reviews(ID, course_id, review, rating, difficulty_level, given_at)

This table gives the rating and review given by a student for a course along with its difficulty level.

---

**comment:**

comment(comment_id, ID, course_id, comment, created_at)

This table contains all the comments under a course along with the time at which it is created.

---

**reply:**

reply(comment_id, ID, course_id, comment, created_at)

This table contains all the replies to a particular comment under a course along with the time at which it is created.

---

### 8 Screens in CourseRank(tentative)

**Homepage:**

This page is same for instructor, students but different for admin. There will be search fields you can view all courses offered by department till now and select necessary course from the resultant list. Along with these admin will also have two options at the top to add or delete a course. Admin can add a new course and delete an old course. Clicking on view more option next to a particular course you are taken to the course page of that particular course. Please find screen shots in Figure 1.

**Login:**

This webpage asks for username and password of the person to look into necessary details of the courses offered by every department. If he types in
wrong pair of username, password it notifies person showing invalid username or password. This page is common to all people i.e Admin, instructors, students and spectators. Please find screen shots in Figure 2.

Signup:
This html page is for people who are not registered for application i.e students not from this institution. This adds the person signing as a spectator. It has fields like email, name, address, username, password. If username is already registered it asks to change the required field. None of them can be empty or null for successful signup of the person. Please find screen shots in Figure 3.

After logging in you are directed to homepage.

Course Page:

• This page contains original course content, modified course content, an option to see reviews, option to see previous years grading statistics, comments along with the replies given to this course.

• This page is different for everyone. Along with the above mentioned parts we will be having different options for different people which are shown based on their designations.

• Admin will be having an option to delete the comments which are inappropriate.

• Instructor will have an option to modify the course content of the
Figure 2: Login page of CourseRank

Figure 3: Sign Up page of CourseRank
course he is teaching so that students can get a brief idea what will be on in the course which will be shown on the modified content part of the page.

- Student who took a course and passed it will be having an option to review the course.

- Original course content gives the details of course like course code, course name, total credits of the course, type of the course (whether it is theory or lab), lectures per week, number of tutorials per week, number of hours required for practicals per week, number of hours required for self study per week, whether the course is half or full semester, prerequisites if any, reference textbooks, short description of the course.

- Modified course content will be having course code and name same as that of the original content while the other fields being default. Modified content will be shown along with instructor name who modified it along with the timestamp at which it is changed.

- On clicking review it redirects to another page which shows all the reviews given by students who took that course in the previous offering and passed.

- On clicking grade statistics we are redirected to a page which shows grading statistics of this course in the previous offerings. Since, we had not developed this yet, we were not able to attach screen shots.

Reviews:
This page displays all the reviews summary for the course given by students of university who had passed them. User can see the reviews and he will be given an option to choose how to display different reviews (sort by time stamp or username etc..). On clicking on one review it will redirect to review view page which contains the selected review elaborately.

Reviews view page:
This page displays the selected review completely.

Statistics:
This page asks to type in year and semester for which you want to know the grade statistics. Once you click search button it gives the information like name of the instructor, number of students associated with each grade only if the course was offered in the interested year and semester. Since, we had not developed this yet, we were not able to attach screen shots.
9 Classes and JSPs(servlets)

- Objects people and course are needed to get information from database which are to be displayed on different web pages of the application. course object is needed while displaying the course page and index page of the application.

- Objects like comment, reply, review, statistics are needed to get information from database which are to be displayed on web pages accordingly when requests are made. Along with these helper objects are needed like Pair etc.

- Classes like CourseRetrieval, StudentRetrieval, FormValidations are used in JSPs to get information about courses to be displayed on index screen according to department chosen, pages to be displayed based on their designations and checking whether ID,password in the database match to that of values given on login page. These classes help us to connect to database and acts as a bridge between database and jsp.

- JSPs are used to write webpages in our application. We have many jsp pages which are used to login, signup, logout, index page which shows course offered in the institute, course page containing all the information about the course, review page, statistics page.

10 References Used

(1) https://erdplus.com/
(2) http://www.postgresql.org/
(3) http://www.latextemplates.com/