Databases and information systems laboratory CS313

IIT Dharwad

Handout 4
30 – 08 – 2023

Consider a University database which we have discussed in the theory class (Primary keys are underlined):

- `classroom (` **building varchar(15), room_number varchar(7)**`, capacity numeric(4,0) )`
- `department (` **dept_name varchar(20)**`, building varchar(15), budget numeric(12,2) )`
  Here Budget should be non-negative
- `course (` **course_id varchar(8)**`, title varchar(50), dept_name varchar(20), credits numeric(2,0) )`
  Here credits should be greater than 0, dept_name is a foreign key that references department and it should be set to null if the source row is deleted.
- `instructor (` **ID varchar(5)**`, name varchar(20) not null, dept_name varchar(20), salary numeric(8,2),`
  `)`
  Here salary should be at least 29000 and dept_name is a foreign key that references department and it should be set to null if the source row is deleted.
- `section (` **course_id varchar(8), sec_id varchar(8), semester varchar(6)**`, year numeric(4,0), building varchar(15), room_number varchar(7), time_slot_id varchar(4) )`
  Here semester should be one of the values ('Fall', 'Winter', 'Spring', 'Summer') ; year should be between 1701 and 2100; course_id is a foreign key that references course and it should be deleted if the source

is deleted; (building, room_number) references classroom and it should be set to `null` if the source is deleted

- `teaches (` **ID varchar(5), course_id varchar(8), sec_id varchar(8), semester varchar(6), year numeric(4,0)** `)`
  Here (course_id, sec_id, semester, year) references section and ID) references instructor (ID); futher both references should be deleted if the source rows are deleted

- `student (` **ID varchar(5)**`, name varchar(20), dept_name varchar(20), tot_cred numeric(3,0) )`
  Here name should not be `null`; (dept_name) references department and should be set to `null` if the source rows are deleted

- `takes (` **ID varchar(5), course_id varchar(8), sec_id varchar(8), semester varchar(6), year numeric(4,0)**`, grade varchar(2) )`
  Here (course_id, sec_id, semester, year) references section and ID references student; futher both should be deleted if the source rows are deleted

- `advisor (`**s_ID varchar(5)**`, i_ID varchar(5) )`
  Here (i_ID) references instructor (ID) and (s_ID) references student (ID); futher both should be deleted if the source rows are deleted

- `time_slot (` **time_slot_id varchar(4)**`, day varchar(1), start_hr numeric(2), start_min numeric(2), end_hr numeric(2), end_min numeric(2) )`
  Here start_hr and end_hr should be between 0 and 24; start_min and end_min should be between 0 and 60

- `prereq (` **course_id varchar(8), prereq_id varchar(8)** `)`
  Here both course_id and prereq_id references course (course_id); futher both should be deleted if the source rows are deleted

- Create a new database called `universitydb`
- Execute `create-tables.sql` and `populate.sql` to create tables and populate them respectively.

Write queries for the following (Add appropriate data to check your queries if it produces empty result)

1. Find the maximum and minimum number of enrollments across all sections (ignore the sections that do not have any enrollment)
2. Find all sections that had the maximum enrollment (along with the enrollment), using a subquery.
3. Repeat Q1, but now also include sections with no students taking them; the enrollment for such sections should be treated as 0. Do this in two different ways.
   (a) Using a scalar subquery
   (b) Using aggregation on a left outer join (use the SQL natural left outer join syntax)
4. Find all courses whose identifier starts with the string "CS-1"
5. Find instructors who have taught all the above courses (obtained as answer for previous question)
   (a) Using the "not exists ... except ..." structure
   (b) Using matching of counts which we covered in class
6. Insert each instructor as a student, with tot_creds = 0, in the same department to which the instructor belongs
7. Delete all the newly added "students" in the previous query (Note: already existing students who happened to have tot_creds = 0 should not get deleted)
8. Update the salary of each instructor to 10000 times the number of course sections they have taught.
9. Create a view that lists information about all fail grades (grade = F in takes table) of all students along with their id, course_id, sec_id, semester, year and grade (the view should contain all attributes from the takes relation).

10. Using the view defined in the previous query, find all students who have 2 or more F grades and list the student ids.
11. Grades are mapped to a grade point as follows: A:10, B:8, C:6, D:4 and F:0. Create a new table to store these mappings.
12. Using the table created previously (and other relavent tables), write a query to find the Cumulative Grade Point Average of each student, using this table.
13. Find all rooms that have been assigned to more than one section at the same time. Display the rooms along with the assigned sections.
14. Create a view faculty showing only the ID, name, and department of instructors.
15. Create a view CSinstructors, showing all information about instructors from the Comp. Sci. department.
16. Insert appropriate tuple into each of the views faculty and CSinstructors, to see what updates your database allows on views. (In the submission you should give all tuples you have tried to insert and the corresponding reaction from the database.)
17. Create a new user with some name. Grant permission to this new user to view all data in the student relation.