**201533661 이승수’s database homework#3 date: 2016. 10. 11.**

**3.1 Write the following queries in SQL, using the university schema. (We suggest you actually run these queries on a database, using the sample data that we provide on the Web site of the book, db-book.com. Instructions for setting up a database, and loading sample data, are provided on the above Web site.)**

**a. Find the titles of courses in the Comp. Sci. department that have 3 credits.**

Select title from course where dept\_name=’Comp. Sci.’ and credits=3;

**b. Find the IDs of all students who were taught by an instructor named Einstein; make sure there are no duplicates in the result.**

Select distinct student.ID from (student join takes using(ID)) join (instructor join teaches using(ID)) using(course\_id,sec\_id,semester,year) where instructor.name=”Einstein”;

**c. Find the highest salary of any instructor.**

Select max(salary) from instructor;

**d. Find all instructors earning the highest salary (there may be more than one with the same salary).**

Select name from instructor where salary=(select max(salary) from instructor);

**e. Find the enrollment of each section that was offered in Autumn 2009.**

Select course\_id,sec\_id,count(ID) from section natural join takes where semester=”Fall” and year=2009 group by course\_id,sec\_id;

**f. Find the maximum enrollment, across all sections, in Autumn 2009.**

Select max(enrollment) from (select count(ID) as enrollment from section natural join takes where semester=”Fall” and year=2009 group by course\_id,sec\_id);

**g. Find the sections that had the maximum enrollment in Autumn 2009.**

With sec\_enrollment as (select course\_id,sec\_id,count(ID) as enrollment from section natural join takes where semester=”Fall” and year=2009 group by course\_id,sec\_id) select course\_id,sec\_id from sec\_enrollment where enrollment=(select max(enrollment) from sec\_enrollment);

**3.2 Suppose you are given a relation grade points(grade, points), which provides a conversion from letter grades in the takes relation to numeric scores; for example an “A” grade could be speciﬁed to correspond to 4points, an “A−” to 3.7 points, a “B+” to 3.3 points, a “B” to 3 points, and so on. The grade points earned by a student for a course offering (section) is deﬁned as the number of credits for the course multiplied by the numeric points for the grade that the student received. Given the above relation, and our university schema, write each of the following queries in SQL. You can assume for simplicity that no takes tuple has the null value for grade.**

**a. Find the total grade-points earned by the student with ID12345, across all courses taken by the student**.

Select tot\_cred from student where ID=’12345’;

**b. Find the grade-point average (GPA) for the above student, that is, the total grade-points divided by the total credits for the associated courses.**

Select sum(credits\*points)/sum(credits) as GPA from (takes natural join course) natural join grade\_points where ID=”12345”;

**c. Find the ID and the grade-point average of every student.**

Select ID<sum(credits\*points)/sum(credits) as GPA from (takes natural join course) natural join grade\_points group by ID;

**3.3 Write the following inserts, deletes or updates in SQL, using the university schema.**

**a. Increase the salary of each instructor in the Comp. Sci. department by 10%.**

Update instructor set salary=salary\*1.1 where dept\_name=”Comp. Sci.”;

**b. Delete all courses that have never been offered (that is, do not occur in the section relation).**

Delete from course where course\_id not in (select course\_id from section);

**c. Insert every student whose tot cred attribute is greater than 100 as an instructor in the same department, with a salary of $10,000.**

Insert into instructor select ID,name,dept\_name,100000 from student where tot\_cred>100;

**3.8 Consider the bank database of Figure3.19, where the primary keys are underlined. Construct the following SQL queries for this relational database.**

**a. Find all customers of the bank who have an account but not a loan.**

Select customer\_name from depositor except (select customer\_name from borrower);

**b. Find the names of all customers who live on the same street and in the same city as “Smith”.**

Select F.customer\_name from customer F join customer S using(customer\_street,customer\_city) where S.customer\_name=”Smith”;

**c. Find the names of all branches with customers who have an account in the bank and who live in “Harrison”.**

Select distinct branch\_name from account natural join depositor natural join customer where customer\_city=”Harrison”;

**3.9 Consider the employee database of Figure3.20, where the primary keys are underlined. Give an expression in SQL for each of the following queries.**

**a. Find the names and cities of residence of all employees who work for “First Bank Corporation”**.

Select e.employee\_name,city from employee e,works w where w.company\_name=”First Bank Corporation” and w.employee\_name=e.employee\_name;

**b. Find the names, street addresses, and cities of residence of all employees who work for First Bank Corporation and earn more than $10,000.**

Select \* from employee where employee\_name in(select employee\_ame from works where company\_name=”First Bank Corporation and salary>10000”);

**c. Find all employees in the database who do not work for First Bank Corporation.**

Select employee\_name from works where company\_name!=”FirstBank Corporation”;

**d. Find all employees in the database who earn more than each employee of Small Bank Corporation.**

**e. Assume that the companies may be located in several cities. Find all companies located in every city in which Small Bank Corporation is located.**

**f. Find the company that has the most employees.**

**g. Find those companies whose employees earn a higher salary, on average, than the average salary at First Bank Corporation.**

**3.10 Consider the relational database of Figure 3.20. Give an expression in SQL for each of the following queries.**

**a. Modify the database so that “Jones” now lives in “Newtown”.**

Update employee set city=”Newton” where person\_name=”Jones”;

**b. Give all managers of “First Bank Corporation” a 10 percent raise unless the salary becomes greater than $100,000; in such cases, give only a 3 percent raise.**

Update works T set T.salary=T.salary\*1.03 where T.employee\_name in (select managet\_name from manages) and T.salary\*1.1>100000 and T.company\_name=”First Bank Corporation”;

**3.11 Write the following queries in SQL, using the university schema.**

**a. Find the names of all students who have taken at least one Comp. Sci. course; make sure there are no duplicate names in the result.**

**b. Find the ID sand names of all students who have not taken any course offering before Spring 2009.**

**c. For each department, ﬁnd the maximum salary of instructors in that department. You may assume that every department has at least one instructor. d. Find the lowest, across all departments, of the per-department maximum salary computed by the preceding query.**

**3.12 Write the following queries in SQL, using the university schema.**

**a. Create a new course“CS-001”, titled “Weekly Seminar”, with 0credits.**

Insert into course(course\_id,title,credits) values(“CS-001”,”Weekly Seminar”,0);

**b. Create a section of this course in Autumn 2009, with sec id of 1.**

Insert into section(course\_id,sec\_id,semester,year) values(“CS-001”,1,”Fall”,2009);

**c. Enroll every student in the Comp. Sci. department in the above section.**

Insert into takes(ID,course\_id) values(student.ID,”CS-001”) from student where student.dept\_name=”Comp. Sci.”;

**d. Delete enrollments in the above section where the student’s name is Chavez.**

Delete from takes where course\_id=”CS-001” and ID=”23121”;

**e. Delete the course CS-001. What will happen if you run this delete statement without ﬁrst deleting offerings (sections) of this course.**

The course will not be deleted because there is any tuples.

**f. Delete all takes tuples corresponding to any section of any course with the word “database” as apart of the title; ignore case when matching the word with the title.**

**3.15 Consider the bank database of Figure3.19, where the primary keys are underlined. Construct the following SQL queries for this relational database.**

**a. Find all customers who have an account at all the branches located in “Brooklyn”.**

Select customer\_name from account,depositor,branch where depositor.account\_number= account.account\_number and account.branch\_name= branch.branch\_name and branch.branch\_city= ”Brooklyn” and count(branch)=count(account);

**b. Find out the total sum of all loan amounts in the bank.**

Select sum(amount) from loan,branch where branch.branch\_name=loan.branch\_name;

**c. Find the names of all branches that have assets greater than those of at least one branch located in “Brooklyn”.**

Select S.branch\_name from branch as S,branch as T where T.branch\_city=”Brooklyn” and S.asset>T.asset;

**3.16 Consider the employee database of Figure3.20, where the primary keys are underlined. Give an expression in SQL for each of the following queries.**

**a. Find the names of all employees who work for “First Bank Corporation”.**

Select employee\_name from works where company\_name=”First Bank Corporation”;

**b. Find all employees in the database who live in the same cities as the companies for which they work.**

Select distinct employee\_name from employee,works,company where employee.city=company.city and company.company\_name=works.company\_name;

**c. Find all employees in the database who live in the same cities and on the same streets as do their managers.**

Select E.employee\_name from employee as E, employee as M, manages where E.employee\_name=manages.employee\_name and M.demployee\_name=manages.manager\_name and E.city=M.city and E.street=M.street;

**d. Find all employees who earn more than the average salary of all employees of their company.**

Select employee\_name from works where salary>avg(salary) group by company\_name;

**e. Find the company that has the smallest payroll.**

Select company\_name from works where min(select count(employee\_name) group by company\_name);

**3.17 Consider the relational database of Figure 3.20. Give an expression in SQL for each of the following queries.**

**a. Give all employees of “First Bank Corporation” a 10 percent raise.**

Update works set salary=salary\*1.1 where company\_name=”First Bank Corporation”;

**b. Give all managers of “First Bank Corporation” a 10 percent raise.**

Update works set salary=salary\*1.1 where company\_name=”First Bank Corporation” and manages.manager\_name=works.employee\_name;

**c. Delete all tuples in the works relation for employees of “Small Bank Corporation”.**

Delete from works where company\_name=”Small Bank Corporation”;