MODULE - 2

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 IDs and 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.

10

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

a. Create a new course "CS-001", titled "Weekly Seminar", with 0 credits.

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

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

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

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

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

10

3.18 List two reasons why null values might be introduced into the database.

2

3.19 Show that, in SQL, <> **all** is identical to **not in**.

5

employee (*employee_name*, *street*, *city*)
works (*employee_name*, *company_name*, *salary*)
company (*company_name*, *city*)
manages (*employee_name*, *manager_name*)

Figure 3.20 Employee database for Exercises 3.9, 3.10, 3.16, 3.17, and 3.20.

3.20 Give an SQL schema definition for the employee database of Figure 3.20. Choose an appropriate domain for each attribute and an appropriate primary key for each relation schema. 10

3.22 Rewrite the **where** clause

where unique (**select** *title* **from** *course*)

without using the **unique** construct.

member(*memb_no*, *name*, *age*)
book(*isbn*, *title*, *authors*, *publisher*)
borrowed(*memb_no*, *isbn*, *date*)

Figure 3.21 Library database for Exercise 3.21.

5

member(<u>*memb_no*</u>, *name*, *age*)
book(<u>*isbn*</u>, *title*, *authors*, *publisher*)
borrowed(<u>*memb_no*</u>, <u>*isbn*</u>, *date*)

Figure 3.21 Library database for Exercise 3.21.

3.21 Consider the library database of Figure 3.21. Write the following queries in SQL.

a. Print the names of members who have borrowed any book published by "McGraw-Hill".

b. Print the names of members who have borrowed all books published by "McGraw-Hill".

c. For each publisher, print the names of members who have borrowed more than five books of that publisher.

d. Print the average number of books borrowed per member. Take into account that if an member does not borrow any books, then that member does not appear in the *borrowed* relation at all.

10

3.23 Consider the query:

```
select course_id, semester, year, sec_id, avg (tot_cred)
from takes natural join student
where year = 2009
group by course_id, semester, year, sec_id
having count (ID) >= 2;
```

Explain why joining *section* as well in the **from** clause would not change the result.

5

3.24 Consider the query:

```
with dept_total (dept_name, value) as
     (select dept_name, sum(salary)
     from instructor
     group by dept_name),
dept_total_avg(value) as
     (select avg(value)
     from dept_total)
select dept_name
from dept_total, dept_total_avg
where dept_total.value >= dept_total_avg.value;
```

Rewrite this query without using the **with** construct.

5

4.2 Outer join expressions can be computed in SQL without using the SQL **outer join** operation. To illustrate this fact, show how to rewrite each of the following SQL queries without using the **outer join** expression.

a. **select* from** *student* **natural left outer join** *takes*

b. **select* from** *student* **natural full outer join** *takes*

5

4.3 Suppose we have three relations $r(A, B)$, $s(B, C)$, and $t(B, D)$, with all attributes declared as **not null**. Consider the expressions

- r **natural left outer join** (s **natural left outer join** t), and
- (r **natural left outer join** s) **natural left outer join** t

a. Give instances of relations r, s and t such that in the result of the second expression, attribute C has a null value but attribute D has a non-null value.

b. Is the above pattern, with C null and D not null possible in the result of the first expression? Explain why or why not.

10

4.9 SQL allows a foreign-key dependency to refer to the same relation, as in the following example:

```
create table manager
   (employee_name    varchar(20)  not null
    manager_name     varchar(20)  not null,
    primary key employee_name,
    foreign key (manager_name) references manager
                                  on delete cascade )
```

Here, *employee_name* is a key to the table *manager*, meaning that each employee has at most one manager. The foreign-key clause requires that every manager also be an employee. Explain exactly what happens when a tuple in the relation *manager* is deleted.

10

4.13 Under what circumstances would the query

```
select *
from student natural full outer join takes natural full outer join course
```

include tuples with null values for the *title* attribute?

5

4.14 Show how to define a view *tot_credits* (*year*, *num_credits*), giving the total number of credits taken by students in each year.

5

4.17 Explain why, when a manager, say Satoshi, grants an authorization, the grant should be done by the manager role, rather than by the user Satoshi.

5

4.18 Suppose user *A*, who has all authorizations on a relation *r*, grants select on relation *r* to **public** with grant option. Suppose user *B* then grants select on *r* to *A*. Does this cause a cycle in the authorization graph? Explain why.

5

4.19 Database systems that store each relation in a separate operating-system file may use the operating system's authorization scheme, instead of defining a special scheme themselves. Discuss an advantage and a disadvantage of such an approach.

5

5.1 Describe the circumstances in which you would choose to use embedded SQL rather than SQL alone or only a general-purpose programming language.

5

5.2 Write a Java function using JDBC metadata features that takes a `ResultSet` as an input parameter, and prints out the result in tabular form, with appropriate names as column headings.

5

5.3 Write a Java function using JDBC metadata features that prints a list of all relations in the database, displaying for each relation the names and types of its attributes.

5

5.4 Show how to enforce the constraint "an instructor cannot teach in two different classrooms in a semester in the same time slot." using a trigger (remember that the constraint can be violated by changes to the *teaches* relation as well as to the *section* relation).

5

5.6 To maintain the *tot_cred* attribute of the *student* relation, carry out the following:

a. Modify the trigger on updates of *takes*, to handle all updates that can affect the value of *tot_cred*.

b. Write a trigger to handle inserts to the *takes* relation.

c. Under what assumptions is it reasonable not to create triggers on the *course* relation?

5

5.9 Show how to express **group by cube**(a, b, c, d) using **rollup**; your answer should have only one **group by** clause.

5

5.10 Given a relation $S(student, subject, marks)$, write a query to find the top n students by total marks, by using ranking.

5

5.15 Consider an employee database with two relations

employee (<u>*employee_name*</u>, *street*, *city*)
works (<u>*employee_name*</u>, *company_name*, *salary*)

where the primary keys are underlined. Write a query to find companies whose employees earn a higher salary, on average, than the average salary at "First Bank Corporation".

a. Using SQL functions as appropriate.

b. Without using SQL functions.

5

6.1 Write the following queries in relational algebra, using the university schema.

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

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

c. Find the highest salary of any instructor.

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

employee (*person_name*, *street*, *city*)
works (*person_name*, *company_name*, *salary*)
company (*company_name*, *city*)
manages (*person_name*, *manager_name*)

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

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

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

6.3 The natural outer-join operations extend the natural-join operation so that tuples from the participating relations are not lost in the result of the join. Describe how the theta-join operation can be extended so that tuples from the left, right, or both relations are not lost from the result of a theta join.

6.5 Let the following relation schemas be given:

$$R = (A, B, C)$$
$$S = (D, E, F)$$

Let relations $r(R)$ and $s(S)$ be given. Give an expression in the tuple relational calculus that is equivalent to each of the following:

a. $\Pi_A(r)$

b. $\sigma_{B=17}(r)$

c. $r \times s$

d. $\Pi_{A,F}(\sigma_{C=D}(r \times s))$

5

6.6 Let $R = (A, B, C)$, and let r_1 and r_2 both be relations on schema R. Give an expression in the domain relational calculus that is equivalent to each of the following:

a. $\Pi_A(r_1)$

b. $\sigma_{B=17}(r_1)$

c. $r_1 \cup r_2$

d. $r_1 \cap r_2$

e. $r_1 - r_2$

f. $\Pi_{A,B}(r_1) \bowtie \Pi_{B,C}(r_2)$

5

6.7 Let $R = (A, B)$ and $S = (A, C)$, and let $r(R)$ and $s(S)$ be relations. Write expressions in relational algebra for each of the following queries:

a. $\{< a > \mid \exists b\, (< a, b > \in r \land b = 7)\}$

b. $\{< a, b, c > \mid < a, b > \in r \land < a, c > \in s\}$

c. $\{< a > \mid \exists c\, (< a, c > \in s \land \exists b_1, b_2\, (< a, b_1 > \in r \land < c, b_2 > \in r \land b_1 > b_2))\}$

5

employee (<u>*person_name*</u>, *street*, *city*)
works (<u>*person_name*</u>, *company_name*, *salary*)
company (<u>*company_name*</u>, *city*)
manages (<u>*person_name*</u>, *manager_name*)

6.8 Consider the relational database of Figure 6.22 where the primary keys are underlined. Give an expression in tuple relational calculus for each of the following queries:

a. Find all employees who work directly for "Jones."

b. Find all cities of residence of all employees who work directly for "Jones."

c. Find the name of the manager of the manager of "Jones."

d. Find those employees who earn more than all employees living in the city "Mumbai."

10

6.10 Write the following queries in relational algebra, using the university schema.

a. Find the names of all students who have taken at least one Comp. Sci. course.

b. Find the IDs and 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.

10

6.11 Consider the relational database of Figure 6.22, where the primary keys are underlined. Give an expression in the relational algebra to express each of the following queries:

a. Find the names of all employees who work for "First Bank Corporation".

b. Find the names and cities of residence of all employees who work for "First Bank Corporation".

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

d. Find the names of all employees in this database who live in the same city as the company for which they work.

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

6.14 Consider the following relational schema for a library:

member(<u>*memb_no*</u>, *name*, *dob*)
books(<u>*isbn*</u>, *title*, *authors*, *publisher*)
borrowed(<u>*memb_no*</u>, <u>*isbn*</u>, *date*)

Write the following queries in relational algebra.

a. Find the names of members who have borrowed any book published by "McGraw-Hill".

b. Find the name of members who have borrowed all books published by "McGraw-Hill".

c. Find the name and membership number of members who have borrowed more than five different books published by "McGraw-Hill".

d. For each publisher, find the name and membership number of members who have borrowed more than five books of that publisher.

e. Find the average number of books borrowed per member. Take into account that if an member does not borrow any books, then that member does not appear in the *borrowed* relation at all.

10

6.16 Let $R = (A, B)$ and $S = (A, C)$, and let $r(R)$ and $s(S)$ be relations. Write relational-algebra expressions equivalent to the following domain-relational-calculus expressions:

a. $\{< a > \mid \exists\, b\, (< a, b > \in r \ \wedge\ b = 17)\}$

b. $\{< a, b, c > \ \mid\ < a, b > \in r \wedge\ < a, c > \in s\}$

c. $\{< a > \mid \exists\, b\, (< a, b > \in r) \vee \forall\, c\, (\exists\, d\, (< d, c > \in s) \Rightarrow\ < a, c > \in s)\}$

d. $\{< a > \ \mid \exists\, c\, (< a, c > \in s \ \wedge\ \exists\, b_1, b_2\, (< a, b_1 > \in r \wedge\ < c, b_2 > \in r \ \wedge\ b_1 > b_2))\}$

5

6.18 Let $R = (A, B)$ and $S = (A, C)$, and let $r(R)$ and $s(S)$ be relations. Using the special constant *null*, write tuple-relational-calculus expressions equivalent to each of the following:

a. r ⟕ s

b. r ⟗ s

c. r ⟖ s 5

6.19 Give a tuple-relational-calculus expression to find the maximum value in relation $r(A)$. 5