**Homework Assignment 5**

Due: 11:59PM, Sunday, June 11, 2023

**1. (3 pt. each) Join operations**

1. (Exercise 4.1) Consider the following SQL query that seeks to find a list of titles of all courses taught in Spring 2017 along with the name of the instructor.

SELECT name, title

FROM instructor

NATURAL JOIN teaches

NATURAL JOIN section

NATURAL JOIN course

WHERE semester = ‘Spring’ AND year = 2017;

What is wrong with this query?

* We need to remember that natural join works among all of the same column names. Seem like the purpose of the problem is do the join with 'course\_id' and 'sec\_id'. But through the rules of the natural join, the join will be based on the 'year' additionally, and it might let the result get loss.

1. (Exercise 4.16) Write an SQL query using the university schema to find the ID of each student who has never taken a course at the university. Do this using no subqueries and no set operations (use an outer join).

* SELECT DISTINCT s.ID FROM student s LEFT JOIN takes t ON s.ID = t.ID

WHERE t. semester IS NULL;

1. (Exercise 4.17) Express the following query in SQL using no subqueries and no set operations.

SELECT ID

FROM student

EXCEPT

SELECT s\_id

FROM advisor

WHERE i\_ID IS NOT NULL;

* SELECT \* FROM student s

LEFT JOIN advisor a ON s.ID = a.s\_ID

WHERE a.s\_ID IS NULL AND a.i\_ID IS NOT NULL;

1. (Exercise 4.20) Show how to define a view tot\_credits(year, num\_credits), giving the total number of credits taken in each year.

* CREATE VIEW tot\_credits(year, num\_credits) AS

SELECT year, sum(credits)

FROM takes t JOIN course c ON t.course\_id = c.course\_id

GROUP BY year

ORDER BY year;

1. (Exercise 4.21) For the view that you have defined in the previous problem (Problem 1(d)), explain why the database system would not allow a tuple to be inserted into the database through this view.

* we need to provide values for both the year and num\_credits columns to insert tuple in the view. but the num\_credits column is calculated from the subquery, so we cannot provide a value for it directly.

**2. Answer the following questions that are from the textbook exercise problem sets. You may refer to the Internet as well as the textbook for assistance; however, your solution should contain your own ideas in your own language.**

1. (3 pt.; Exercise 5.9) Given a relation nyse(year, month, day, shares\_traded, dollar\_volume) with trading data from the New York Stock Exchange, list each trading day in order of number of shared traded, and show each day’s rank.

* **SELECT** year, month, day,

**RANK**() **OVER** (**ORDER** **BY** shares\_traded **DESC**) **AS** rank

**FROM** nyse;

1. (3 pt.; Exercise 5.23) Consider the relation from Problem 2(a). For each month of each year, show the total monthly dollar volume and the average monthly dollar volume for that month and the two prior months. (You may want to use the hint suggested by the textbook.)

* SELECT year, month,

SUM(dollar\_volume) AS total\_monthly\_volume,

ROUND(AVG(dollar\_volume), 2) AS average\_monthly\_volume

FROM nyse

GROUP BY year, month

ORDER BY year, month;

1. (3 pt.; Exercise 5.8) Given a relation S(student, subject, marks), write a query to find the top 10 students by total marks, by using SQL ranking. Include all students tied for the final spot in the ranking, even if that results in more than 10 total students.

* SELECT student, SUM(marks) AS total\_marks,

DENSE\_RANK() OVER (PARTITION BY student ORDER BY total\_marks DESC) AS rank

FROM S

GROUP BY student

ORDER BY total\_marks DESC

LIMIT 10;

1. (4 pt.; Exercise 5.6) Consider the bank database of Figure 5.21.

텍스트, 스크린샷, 폰트, 라인이(가) 표시된 사진

자동 생성된 설명

Let us define a view branch\_cust as follows. Suppose that the view is materialized; that is, the view is computed and stored. Write triggers to maintain the view, that is, to keep it up-to-date on insertions to depositor or account. It is not necessary to handle deletions or updates. Note that, for simplicity, we have not required the elimination of duplicates.

**CREATE VIEW** branch\_cust **AS**

**SELECT** branch\_name, customer\_name

**FROM** depositor, account

**WHERE** depositor.account\_number = account.account\_number;

* **CREATE** **TRIGGER** branch\_cust\_insert\_trigger

**AFTER** **INSERT** **ON** depositor

**FOR** **EACH** **ROW**

**BEGIN**

-- Update the view branch\_cust

**INSERT** **INTO** branch\_cust (branch\_name, customer\_name)

**VALUES** (NEW.branch\_name, NEW.customer\_name);

**END**;

**CREATE TRIGGER** branch\_cust\_insert\_trigger

**AFTER INSERT ON** account

**FOR EACH ROW**

**BEGIN**

**INSERT** **INTO** branch\_cust (branch\_name, customer\_name)

**VALUES** (NEW.branch\_name, NEW.customer\_name);

**END**;

1. (3 pt.; Exercise 5.7) Consider the bank database of Figure 5.21. Write an SQL trigger to carry out the following action: On **DELETE** of an account, for each customer-owner of the account, check if the owner has any remaining accounts, and if she does not, delete her from the depositor relation.

* **CREATE TRIGGER** delete\_account\_trigger

**AFTER DELETE ON** account

**FOR EACH ROW**

**BEGIN**

**DECLARE** customer\_name **VARCHAR**(255);

**SELECT** customer\_name **INTO** customer\_name

**FROM** depositor

**WHERE** account\_number = OLD.account\_number;

**DECLARE** remaining\_accounts **INT**;

**SELECT** **COUNT**(\*) **INTO** remaining\_accounts

**FROM** account

**WHERE** customer\_name = customer\_name;

**IF** remaining\_accounts = 0 **THEN**

**DELETE** **FROM** depositor

**WHERE** customer\_name = customer\_name;

**END IF;**

**END**;

1. (Exercise 17.8) The lost update anomaly is said to occur if a transaction Tj reads a data item, then another transaction Tk writes the data item (possibly based on a previous read), after which Tj writes the data item. The update performed by Tk has been lost, since the update done by Tj ignored the value written by Tk.
2. (3 pt.) Give an example of a schedule shown the lost update anomaly.

* Transaction T1:

1. Read the value of the data item x = 10.

2. Increase the value of x by 1.

3. Write the new value of x = 11 to the database.

* Transaction T2:

1. Read the value of the data item x = 10.

2. Decrease the value of x by 1.

3. Write the new value of x = 9 to the database.

1. (3 pt.) Give an example schedule to show that the lost update anomaly is possible with the **read committed** isolation level.

* Transaction T1:

1. Start a transaction and obtain a read committed lock on the data item x.

2. Read the value of the data item x = 10.

3. Increase the value of x by 1.

4. Write the new value of x = 11 to the database.

* Transaction T2:

1. Start a transaction and obtain a read committed lock on the data item x.

2. Read the value of the data item x = 10.

3. Decrease the value of x by 1.

4. Write the new value of x = 9 to the database.

1. (3 pt.) Explain why the lost update anomaly is not possible with the **repeatable read** isolation level.

* Transaction T1:

1. Start a transaction and obtain a shared lock on the data item x.

2. Read the value of the data item x = 10.

3. Increase the value of x by 1.

4. Write the new value of x = 11 to the database.

* Transaction T2:

1. Start a transaction and obtain a shared lock on the data item x.

2. Read the value of the data item x = 10.

3. Decrease the value of x by 1.

4. Write the new value of x = 9 to the database.

**3. (10 pt.) Consider the following timelines where two transactions are intervening each other. The two vertical downward arrows represent the progression of time. The horizontal arrows represent the dataflow between transaction and storage.**

텍스트, 도표, 평면도, 평행이(가) 표시된 사진

자동 생성된 설명

Assuming three isolation levels, REPEATABLE READ, READ COMMITTED, and READ UNCOMMITTED, what name would be returned after executing "SELECT name FROM roaster WHERE id = 8;"?

|  |  |  |  |
| --- | --- | --- | --- |
|  | REPEATABLE READ | READ COMMITTED | READ UNCOMMITTED |
| (a) | null | null | null |
| (b) | null | null | Bob |
| (c) | null | null | null |

텍스트, 도표, 평면도, 평행이(가) 표시된 사진

자동 생성된 설명

Assuming three isolation levels, REPEATABLE READ, READ COMMITTED, and READ UNCOMMITTED, what name would be returned after executing "SELECT name FROM roaster WHERE id = 7;"?

|  |  |  |  |
| --- | --- | --- | --- |
|  | REPEATABLE READ | READ COMMITTED | READ UNCOMMITTED |
| (d) | Jane | Jane | Jane |
| (e) | Jane | Jane | Mary |
| (f) | Jane | Mary | Amy |
| (g) | Jane | Mary | Mary |

**4. Views, procedures, and functions. Consider the following relations:**

텍스트, 스크린샷, 번호, 폰트이(가) 표시된 사진

자동 생성된 설명

1. (3 pt.) Assume that you have a view that has been created using the following query:

CREATE VIEW BIGBIO(name, funding) AS

SELECT name, total\_funding FROM company

JOIN working\_areas wa on company.ID = wa.company\_id

WHERE area = 'Biotech' AND total\_funding > 100000;

What is the result of the next query?

SELECT \* FROM BIGBIO WHERE name LIKE 'M%';

* \*I found that working\_areas table do not have company\_id column, so I consider companu\_id as CID. If there were no errors in the problem, the answer is NO RESULT.

|  |  |
| --- | --- |
| name | funding |
| Moderna | 2700000000 |
| Medtronic | 367000000 |

1. (3 pt.) Consider a stored procedure given below:

DELIMITER //

CREATE PROCEDURE count\_cpn\_proc(IN area VARCHAR(250), OUT c\_count INTEGER)

BEGIN

SELECT COUNT(\*) INTO c\_count

FROM (

SELECT name, total\_funding FROM company

JOIN working\_areas wa on company.ID = wa.CID

WHERE wa.area = area) AS JOINED;

END//

How does one invoke the above procedure and display the number of companies working in ‘Genetics’ using query SELECT @tmp? Write a single line of query that calls procedure count\_cpn\_proc() for area = 'Genetics'.

* **SET** @tmp = **NULL**;

**CALL** count\_cpn\_proc('Genetics', @tmp);

**SELECT** @tmp;

1. (3 pt.) Assume that you have given a function defined as below:

DELIMITER //

CREATE FUNCTION area\_count(cname VARCHAR(250))

RETURNS INTEGER DETERMINISTIC

BEGIN

DECLARE a\_count INTEGER;

SELECT COUNT(\*) INTO a\_count

FROM company

JOIN working\_areas wa on company.ID = wa.CID

WHERE name = cname;

RETURN a\_count;

END //

What would be the result of the next query?

SELECT name

FROM company

WHERE area\_count(name) > 2;

|  |
| --- |
| Name |
| Acme Corporation |
| XYZ company |

**5. Indexes. Answer the following questions that are from the textbook exercise problem sets. You may refer to the Internet as well as the textbook for assistance; however, your solution should contain your own ideas in your own language.**

1. (4 pt. each; Exercise 14.3) Construct a B+tree for the following set of key values: (2, 3, 5, 7, 11, 17, 19, 23, 29, 31)

Assume that the tree is initially empty and values are added in ascending order. Construct B+trees for the vases where the number of pointers that will fit in one node is as follows:

1. Four

텍스트, 도표, 라인, 평면도이(가) 표시된 사진

자동 생성된 설명

1. Six

텍스트, 도표, 라인, 평면도이(가) 표시된 사진

자동 생성된 설명

1. Eight

텍스트, 도표, 라인, 평면도이(가) 표시된 사진

자동 생성된 설명

1. (2 pt. each; Exercise 14.18) For each B+tree of Exercise 14.3a (not b and c), show the steps involved in the following queries:

텍스트, 도표, 라인, 평면도이(가) 표시된 사진

자동 생성된 설명

1. Find records with a search-key value of 11.
   1. Find 11 at the root node.
   2. Not found, and 11 is less than the root node's key value (19), so we follow the left link.
   3. Find 11 at the arrived node. We found it, so we follow the corresponding right LINK.
   4. Find the value.
2. Find records with a search-key value between 7 and 17, inclusive.
   1. Find the values in the range (7-17) at the root node. We don't find it, so we follow the link to the left.
   2. When you arrive, look for the values in the range. We found 11, so we go to the child node associated with 11.
   3. Repeat the process to get the corresponding data.
3. (5 pt. each; Exercise 14.4) For each B+tree of Exercise 14.3, show the form of the tree after each of the following series of operations:
4. Insert 9.

|  |
| --- |
| 텍스트, 도표, 라인, 스크린샷이(가) 표시된 사진  자동 생성된 설명 |
| 텍스트, 도표, 라인, 스크린샷이(가) 표시된 사진  자동 생성된 설명 |
| 텍스트, 도표, 라인, 스크린샷이(가) 표시된 사진  자동 생성된 설명 |

1. Insert 10.

|  |
| --- |
| 텍스트, 도표, 라인, 폰트이(가) 표시된 사진  자동 생성된 설명 |
| 텍스트, 도표, 라인, 폰트이(가) 표시된 사진  자동 생성된 설명 |
| 텍스트, 도표, 라인, 폰트이(가) 표시된 사진  자동 생성된 설명 |

1. Insert 8.

|  |
| --- |
| 텍스트, 도표, 라인, 폰트이(가) 표시된 사진  자동 생성된 설명 |
| 텍스트, 도표, 라인, 폰트이(가) 표시된 사진  자동 생성된 설명 |
| 텍스트, 도표, 라인, 폰트이(가) 표시된 사진  자동 생성된 설명 |