Homework Assignment 5

Maximum earnable: 90 pt.
Due: 11:59PM, Thursday, Jun 9, 2023

- Read the assignment carefully. *You will need to **write and execute several SQL queries**; and **submit the results of your queries**.*
- You are **allowed to re-use any of the queries from the lecture slides** while developing solutions to the problems.
- This is an individual work; Please be clear with HGU CSEE Standard:
  - Submitting assignments or program codes written by others or acquired from the internet without explicit approval of the professor is regarded as cheating.
  - Showing or lending one's own homework to other student is also considered cheating that disturbs fair evaluation and hinders the academic achievement of the other student.
  - It is regarded as cheating if two or more students conduct their homework together and submit it individually when the homework is not a group assignment.
- Posting any of the assignment on the Internet and asking is prohibited.
- When finished, submit your work to *LMS.*

1. (3 pt. each) Join operations.

(a) (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?

(b) (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).

(c) (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**;

(d) (Exercise 4.20) Show how to define a view *tot_credits*(*year, num_credits*), giving the total number of credits taken in each year.

(e) (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.

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.

(a) (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.

(b) (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.)

(c) (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.

(d) (4 pt.; Exercise 5.6) Consider the bank database of Figure 5.21. Let us define a view *branch_cust* as follows:

```
CREATE VIEW branch_cust AS
    SELECT branch_name, customer_name
    FROM depositor, account
    WHERE depositor.account_number = account.account_number
```

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.

(e) (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.

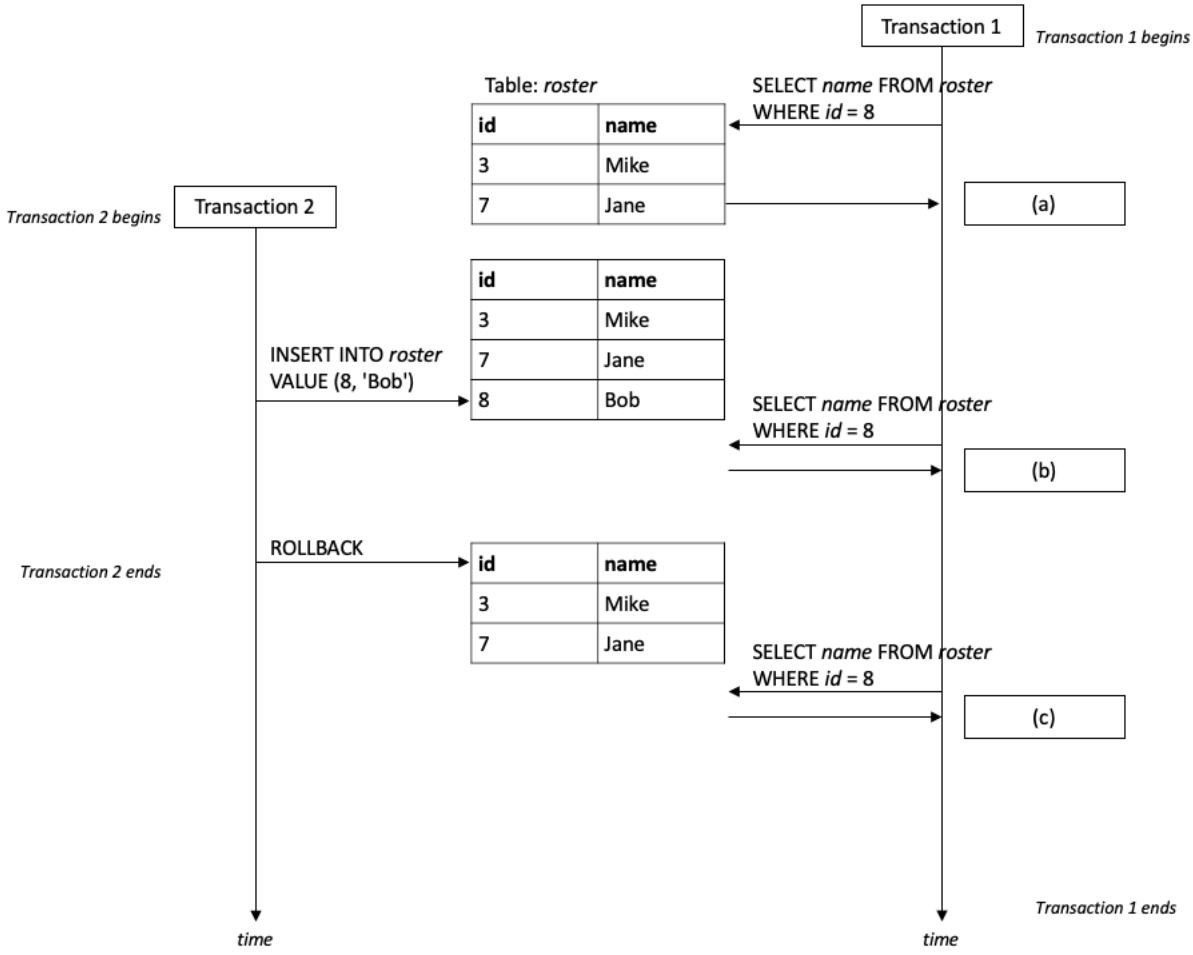
(f) (Exercise 17.8) The *lost update* anomaly is said to occur if a transaction T_j reads a data item, then another transaction T_k writes the data item (possibly based on a previous read), after which T_j writes the data item. The update performed by T_k has been lost, since the update done by T_j ignored the value written by T_k.

a. (3 pt.) Give an example of a schedule shown the lost update anomaly.

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

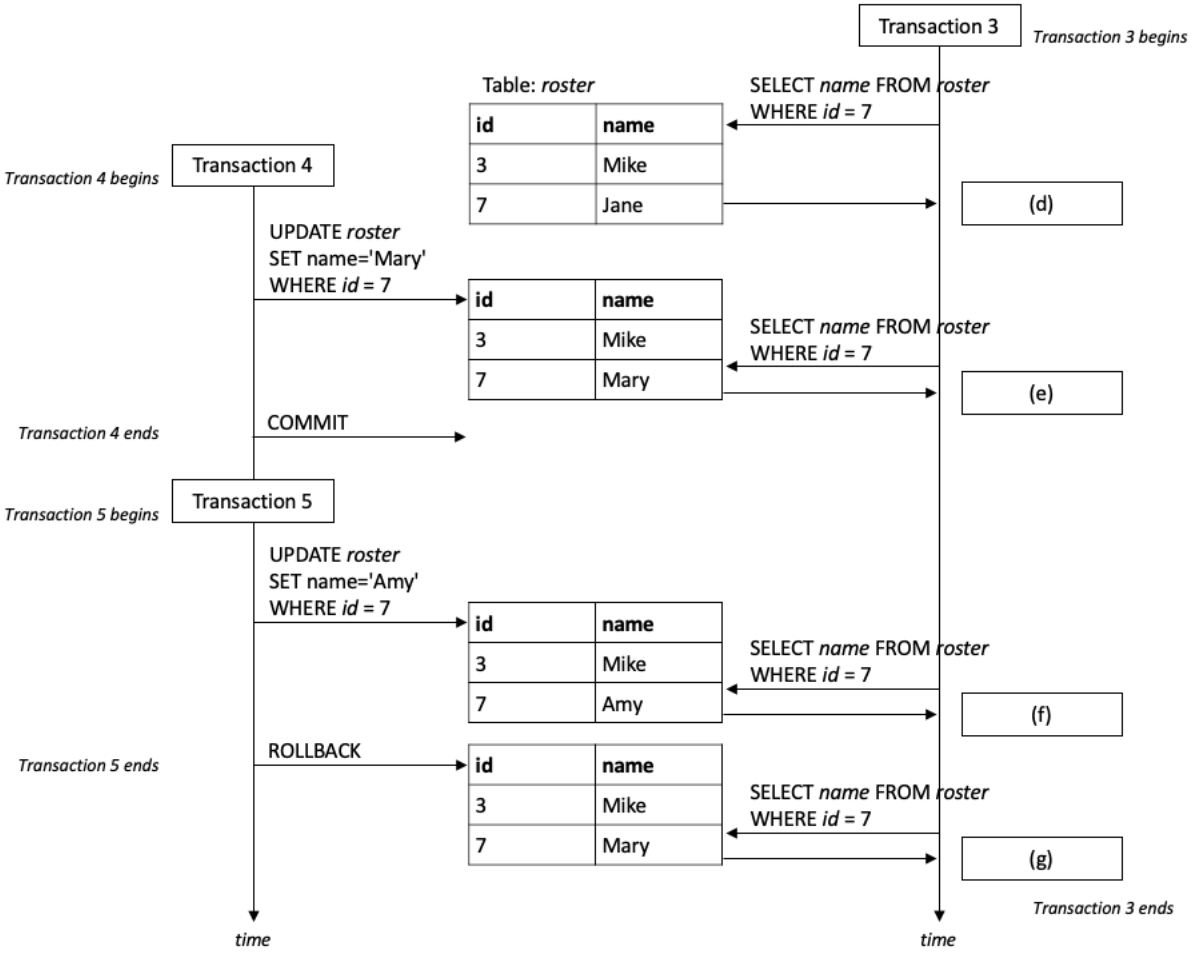
c. (3 pt.) Explain why the lost update anomaly is not possible with the **repeatable read** isolation level.

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.



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)			
(b)			
(c)			



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)			
(e)			
(f)			
(g)			

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

* Table: *company*

ID	*name*	*total_funding*	*location*
1001	Moderna	2700000000	Cambridge, MA
1002	Illumina	28000000	San Diego, CA
1003	Pacific Biosciences	1300000000	Menlo Park, CA
1004	Medtronic	367000000	Minneapolis, MN
1005	DaVita	750000000	Denver, CO
1006	Editas Medicine	656600000	Cambridge, MA
1007	Teladoc Health	172900000	Purchase, NY
1008	Align Technology	26000000	San Jose, CA
1009	Vertex Pharmaceuticals	606200000	Boston, MA

* Table: *working_areas*

CID	*area*
1001	Biotech
1001	Genetics
1001	Pharmaceutical
1002	Biotech
1002	Genetics
1003	Biotech
1003	Genetics
1003	Life Science
1004	Biotech
1004	Health Diagnostics
1004	Medical Device
1004	Manufacturing
1006	Biotech
1006	Genetics
1007	Hospital
1007	mHealth
1008	Manufacturing
1009	Biotech
1009	Pharmaceutical

(a) (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%';
```

(b) (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'`.

(c) (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;
```

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.

(a) (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:

a. Four
b. Six
c. Eight

(b) (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:

a. Find records with a search-key value of 11.
b. Find records with a search-key value between 7 and 17, inclusive.

(c) (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:

a. Insert 9.
b. Insert 10.
c. Insert 8.