Homework Assignment 5

Due: 11:59PM, June 05, 2025

1.

1. (1 pt. each) Read the textbook Chapters 5 and 17. Fill in the blanks with the correct answers.

(a) The ( ) function gives the same rank to all tuples that are equal on the ORDER BY attributes; whereas the (DENSE_RANK()) function does not create gaps in the ordering.

(b) The use of the keyword ( ) in place of **ROWS** allows the windowing query to cover all tuples with a particular value rather than covering a specific number of typles. Thus, **ROWS CURRENT ROW** refers to eactly one tuple, while **RANGE CURRENT ROW** refers to all tuples whose value for the sort attribute is the same as that of the current tuple.

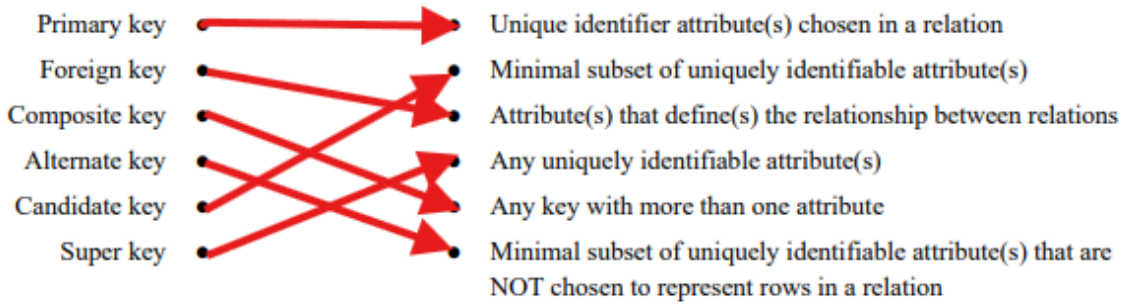
(c) ( ) refers to a collection of operations that form a single logical unit of work.

Answer 1.

a) Rank
b) Range
c) transaction

2.

2. (3 pt.) Match each of the following key types to the corresponding definition.



a. Primary Key / Unique identifier attribute(s) chosen in a relation
b. Foreign Key / Attribute(s) that define(s) the relationship between relations
c. Composite key / Any key with more than one attribute
d. Alternate key / Minimal subset of uniquely identifiable attribute(s) that are NOT chosen to represent rows in a relation
e. Candidate key / Minimal subset of uniquely identifiable attribute(s)
f. Super key / Any uniquely identifiable attribute(s)

3.

(a)According to the textbook description from Section 4.2, what is the main difference between views and named subqueries defined by WITH?

Answer:

Like subquery, the valid range of WITH is inside the clause in which the WITH clause is declared, but VIEW differs in that it can be reused in other queries without a separate DROP declaration.

(b) List the conditions that an SQL view is said to be updatable.

Answer:

1) NULL processed for properties that are not listed.
2) Only attributes for one table in one DB are specified.
3) If there is no DISTINCT specification.
4) GROUP BY, if there is no HAVING clause.

(c) According to the description from Section 17.1, explain what the ACID properties of a database are.

Answer:

ACID is a set of attributes that ensure that database systems maintain data consistency even in the context of simultaneous access by multiple users and system failures.

A: Operations within atomic/transaction operate in an indivisible unit (all or nothing), ensuring data integrity.
C: Consistency/Translation Success The DB stored after success is a property that should be consistent, meaning that integrity constraints should be met.
I : properties that are not subject to interference from other transactions when an isolation/transaction is performed
D : The nature that the information in the DB must be consistent after duration/committed

3.

(d)

(d) Discuss the differences between the following two queries in terms of the computational efficiency.

- ```
  SELECT ID, RANK() OVER (ORDER BY GPA DESC) AS s_rank
  FROM student_grades;
  ```

- ```
  SELECT ID, (1 + (
      SELECT COUNT(*)
      FROM student_grades B
      WHERE B.GPA > A.GPA)) AS s_rank
  FROM student_grades A
  ORDER BY s_rank;
  ```

Answer:

The difference is that the first query is an alignment in which the GPA is ordered in the order of high students, and the order after the equalizer is an alignment in which the number of equalizers is crossed, whereas in the second case, the result is similar to the consecutive numbers (DENSE RANK) after the equalizer.

There is also a difference in execution time. In the case of the first query, the time complexity (TC) is in the range of O(N) to O(Nlg N), but in the case of the second using Subquery, the difference in operation time increases if the total table is compared by the time of O(N^2).

4.

Question 4(a),

(a) (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.

Question 4(a) - 1, Give an example of a schedule shown the lost update anomaly.

Answer a-1)

If two transactions simultaneously approach a common resource X and perform the following operation, the operation result of T1 is completely excluded. (Final storage X = 120, the result of T2)

[Example]

T1: read(X)
T2: read(X)
T1: X = X + 50
T1: write(X)
T2: X = X + 20
T2: write(X)
T1: commit
T2: commit

Question 4(a) - 2, Give an example schedule to show that the lost update anomaly is possible with the read committed
isolation level.

Answer a-2)

Each transaction reads only the committed data (T2 does not read T1's interim calculation results), and T2 overwrites after T1 commits. Since the "Read Committed" level does not prevent other transactions from changing the data I read, T1's update is lost as T2 overwrites the value it read before.
(In other words, the problem arises because the modification of other transactions cannot be controlled after commit.)

[Example]

T1: read(X)
T2: read(X)
T1: X = X + 50
T1: write(X)
T1: commit
T2: X = X + 20
T2: write(X)
T2: commit

Question 4(a) - 3, Explain why the lost update anomaly is not possible with the repeatable read isolation level.
Answer a-3)

When changing read (multiple) / write data, it can be modified and edited only when it is authorized to read or modify data through lock,

exemplified

T1: read(X)
T2: read(X)
T1: X = X + 50
T1: write(X) // error! correction rights locked by exclusive lock. (update, invalidate itself)
T2: X = X + 20
T2: write(X)
T1: commit
T2: commit

In the same case, T2's read diary authority is not returned, so T1's UPDATE is rejected, preventing a situation in which data loss occurs.

Question 4(b),
Given a relation nyse(year, month, day, shares_traded, dollar_volume) with trading data from the New York Stock Exchange, write a query that lists each trading day in order of number of shared traded, and show each day's rank.

Answer b)

```
WITH RankedTrades AS (
SELECT
year,
month,
day,
COUNT (shares_traded) AS num_shared_trade,
DENSE_RANK() OVER (ORDER BY shares_traded DESC) AS day_rank -- shares_traded
FROM
nyse
GROUP BY year, month, day

)
SELECT
year,
month,
day,
num_shared_trade,
day_rank
FROM
RankedTrades
ORDER BY
day_rank ASC;
```

Question 4(c),
Consider the nyse relation from Problem 4(b). For each month of each year, write a query that shows 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.)

Answer c)

```
WITH MonthlyVolume AS (
SELECT
year,
month,
SUM(dollar_volume) AS total_monthly_volume
FROM
nyse
GROUP BY
year,
month
)
SELECT year, month, total_monthly_volume,
AVG(total_monthly_volume) OVER (
PARTITION BY year
ORDER BY month
ROWS BETWEEN 2 PRECEDING AND CURRENT ROW
) AS moving_avg
FROM MonthlyVolume;
```

Question 4(d),
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.

Answer d)

```
WITH StudentTotalMarks AS (
SELECT
student,
SUM(marks) AS total_marks
FROM
S
GROUP BY
student
),
StudentRanks AS (
SELECT
student,
total_marks,
DENSE_RANK() OVER (ORDER BY total_marks DESC) AS student_rank
FROM
StudentTotalMarks
)
SELECT * FROM StudentRanks
WHERE student_rank <= 10;
```

5.

More query exercises. Launch and access the MySQL databases distributed with the class virtual machine. Below uses the "sakila" database (DVD rental database), which consists of 16 tables regarding movie inventory, actors, customers, rental history, payment information, etc. For each of the following queries, evaluate and report the results. You may type in and execute each query to find the solution. Make sure you understand why you obtained the reported results.

Question 5(a),

```
(a)  SELECT inventory_id,
       ROW_NUMBER() OVER (PARTITION BY inventory_id ORDER BY rental_date) AS ROW_NO,
       rental_id, customer_id, rental_date,
       LAG(rental_date, 1) OVER (PARTITION BY inventory_id ORDER BY rental_date) AS PREV_RENTAL
     FROM rental
     WHERE YEAR(rental_date) = 2005 AND MONTH(rental_date) = 8 AND inventory_id <= 5;
```

Answer 5(a)

In the lental table, the target is rental records with lental_date of August 2005 and inventory_id of 5 or less. Group the data by each inventory_id and sort them in ascending order by lental_date within the group. In this sorted order, assign a unique number (ROW_NO) to each row, and within the same inventory_id, show the previous rental_date (PREV_RENTAL) together. Finally, output the inventory_id, ROW_NO, lental_id, customer_id, lental_date, PREV_RENTAL columns.

Question 5(b),

```
(a)  SELECT inventory_id,
       ROW_NUMBER() OVER (PARTITION BY inventory_id ORDER BY rental_date) AS ROW_NO,
       rental_id, customer_id, rental_date,
       LAG(rental_date, 1) OVER (PARTITION BY inventory_id ORDER BY rental_date) AS PREV_RENTAL
     FROM rental
     WHERE YEAR(rental_date) = 2005 AND MONTH(rental_date) = 8 AND inventory_id <= 5;
```

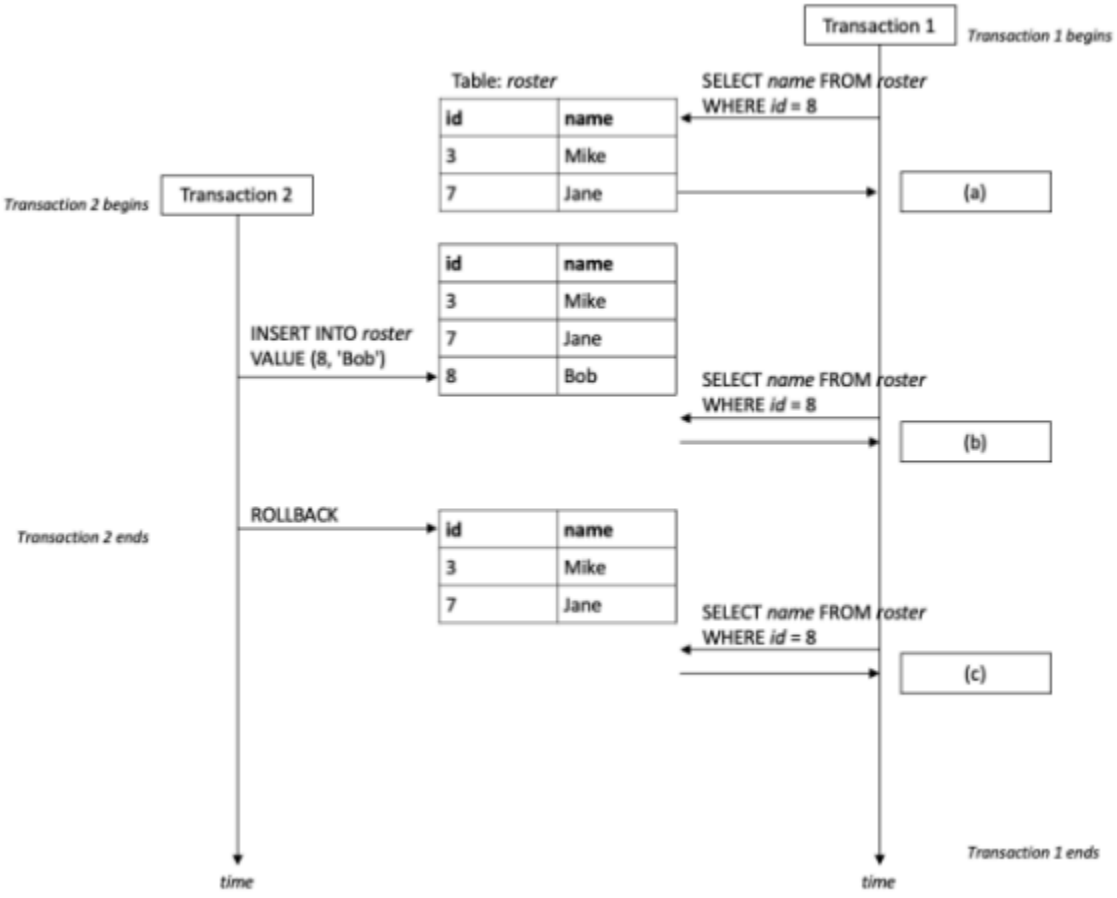
Answer 5(b)

In the payment table, payment records with payment_date of May 2005 and customer_id of 5 or less are targeted. For each payment, the sums of all payments made between the previous 12 hours (INTERVAL 0.5 DAY PRECEDING) and the following 12 hours (INTERVAL 0.5 DAY FOLLOWING) are added and shown as DAILY_SUM. Finally, output the payment_id, customer_id, lental_id, payment_date, account_date, amount, and DAILY_SUM columns.

6.

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.

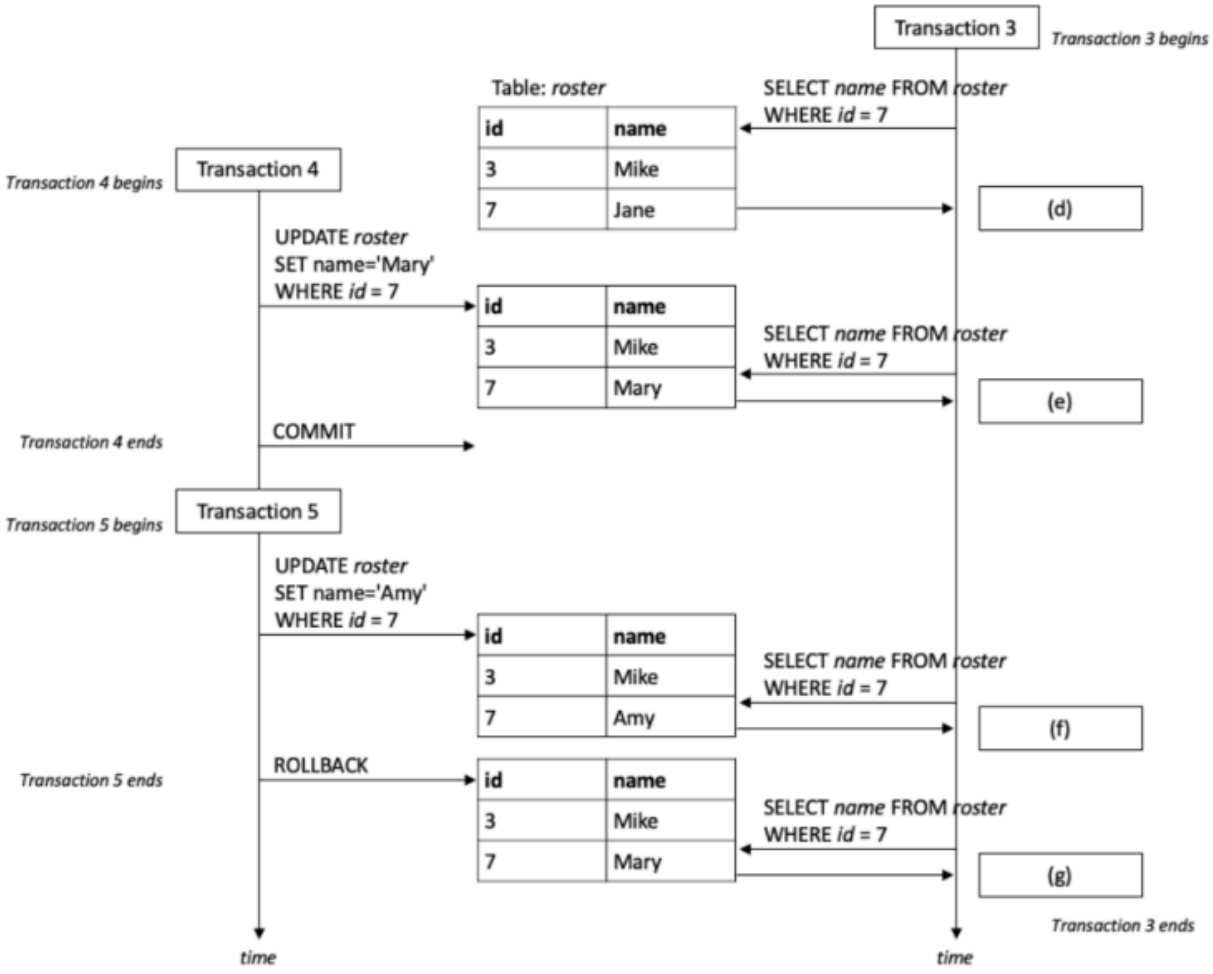
Question 6(a),
Assuming three isolation levels, REPEATABLE READ, READ COMMITTED, and READ UNCOMMITTED, what name would be returned after executing "SELECT name FROM roaster WHERE id = 8;"?



Answer 6(a)

	REPEATABLE READ	READ COMMITTED	READ UNCOMMITTED
(a)	NULL	NULL	NULL
(b)	NULL	NULL	Mary
(c)	NULL	NULL	NULL

Question 6(b),
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;"?



Answer 6(b)

	REPEATABLE READ	READ COMMITTED	READ UNCOMMITTED
(a)	Jane	Jane	NULL
(b)	Jane	Jane	Mary
(c)	Jane	Mary	Amy
(d)	Jane	Mary	Mary