CS411 Database Systems

Spring 2015, Prof. Chang

Department of Computer Science
University of Illinois at Urbana-Champaign

Final Examination

May 8, 2015
Time Limit: 180 minutes

- Print your name and NetID below. In addition, print your NetID in the upper right corner of every page.

Name: _____ **NetID:** _____

- Including this cover page, this exam booklet contains **13** pages. Check if you have missing pages.
- The exam is closed book and closed notes. You are allowed to use non-programmable calculators. No other electronic devices are permitted. Any form of cheating on the examination will result in a zero grade.
- Please write your solutions in the spaces provided on the exam. You may use the blank areas and backs of the exam pages for scratch work.
- Please make your answers clear and succinct; you will lose credit for verbose, convoluted, or confusing answers. *Simplicity does count!*
- Each problem has different weight, as listed below– So, plan your time accordingly.

Problem	1	2	3	4	5	6	Total
Points	54	26	12	15	18	25	150
Score							

Problem 1 (*54 points*) Misc. Concepts

For each of the following statements:

- for true/false choices, indicate whether it is *TRUE* or *FALSE* by **circling** your choice, and provide an **explanation** to justify;
- for short answer questions, provide a brief **answer** showing your work.

You will get *3 points* for each correct answer with correct explanations, and ***no penalty* (of negative points) for wrong answers**. However, for a question requiring explanation, if you give us incorrect/missing explanation, you will get 0 point even when the answer is correct!

Note: Questions 1-3 use the following two tables $R(A, B, C)$ and $S(C, D)$.

Table R

A	B	C
1	2	8
1	4	7
2	5	5
2	4	3

Table S

C	D
2	(1, 2)
3	(5, 6)
5	(6, 9)
7	(5, 11)

(1) Answer: *True* *False*

$\{A, B\}$ can be a key for table R.

Answer: If answering True, explain they do not have duplicates. If answering False, explain we can not infer key by observing data.

(2) Write down the output of $(\pi_A R) \bowtie_{R.A=S.C} S$.

Answer: ($A = 2$, $C = 2$, $D = (1, 2)$)

(3) For a natural join between the tables R and S, i.e., $R \bowtie S$, what is the number of rows and columns of the resulting table?

Answer: Rows - 3, Columns - 4.

(4) Answer: *True* *False*

In a linear hash table, the number of buckets n must be a power of 2.

Answer: False because buckets are added one by one

(5) Assume join is a commutative binary operator, i.e., $A \bowtie B$ is equivalent to $B \bowtie A$. How many distinct query trees are there for query $A \bowtie B \bowtie C$?

Answer: 3, i.e., (A JOIN B) JOIN C, (A JOIN C) JOIN B, (B JOIN C) JOIN A.

(6) Answer: *True* *False*

A left-deep query tree can support not only materialization but also pipelining in query processing.

Answer: True. Left-deep and right-deep tree enable the opportunity to reuse the intermediate result, which is called pipelining.

(7) Answer: *True* *False*

Rule-based query optimization does not always give us the optimal query plan; in contrast, cost-based query optimization always gives us the optimal query plan.

Answer: False because both approaches are only estimations.

(8) Answer: *True* *False*

If we execute all possible query plans, we are able to make a correct query plan decision because we know the actual cost for each plan. So, this approach is more practical than rule-based query optimization.

Answer: False because time for executing all possible plans is too expensive, which is not as practical as rule-based approaches

(9) Answer: *True* *False*

A REDO logging system must write a log entry after each update of a database value on disk.

Answer: False: Regardless of UNDO or REDO, a logging system must write the corresponding log entry **before** any updates of database values on disk.

(10) Among the four "ACID" properties achieved by transaction management, which two properties are guaranteed by failure recovery?

Answer: Atomicity and Durability. Answers like "A" and "D" will get -1

(11) Answer: *True* *False*

To perform a non-quiescent checkpoint, the database needs to stop accepting new connections.

Answer: False: Since non-quiescent checkpoint does not require to wait till the active transactions have written a COMMIT or an ABORT to the log, non-quiescent checkpoint can be done with out accepting new connections.

(12) Answer: *True* *False*

In case of UNDO logging, $\langle$T1, A, 1$\rangle$ means that transaction T1 has changed the database element A, and its new value is 1.

Answer: False: In case of UNDO logging, $\langle$T1, A, 1$\rangle$ means that transaction T1 has changed the database element A, and its **former** value is 1.

(13) Answer: *True* *False*

A relation with two attributes, e.g., $R(A, B)$, with an unknown set of functional dependencies, must necessarily be in 3NF.

Answer: True: Any functional dependencies between two attributes cannot violate 3NF for the table composed of the two attributes.

(14) Answer: *True* *False*

Functional dependencies can be inferred by observing data.

Answer: False: Observations from the data can be coincidental. One should rely on domain knowledge to obtain functional dependencies.

(15) Answer: *True* *False*

In ER modeling, every entity must own a minimal set of uniquely identifying attributes, which is called the entity's primary key.

Answer: False, weak entities do not own all attributes of its primary key. The exact phrase on the Wikipedia page about ER model: "Every entity (unless it is a weak entity) must have a minimal set of uniquely identifying attributes, which is called the entity's primary key."

(16) Consider two relations about people who have lived in Champaign and Urbana: *ChampaignLog*(SSN, name, occupation, info), *UrbanaLog*(SSN, name, occupation, info). Apply algebraic laws on the following relational algebra expression to optimize the cost. Make sure selection and projection operations are executed as soon and in as small scope as possible.

$\pi_{SSN,occupation,name}(\sigma_{occupation='professor'}(ChampaignLog - UrbanaLog))$

Answer:

$\sigma_{occupation='professor'}(\pi_{SSN,occupation,name}(ChampaignLog)) - \pi_{SSN,occupation,name}(UrbanaLog)$
or
$\pi_{SSN,occupation,name}(\sigma_{occupation='professor'}(ChampaignLog) - \sigma_{occupation='professor'}(UrbanaLog))$
A few other variants are acceptable too.

(17) Convert the E/R diagram below to a relational database schema, using one of the following approaches: E/R, OO and Nulls. You only need to choose one of the methods, but you must indicate which method you use. Failure to do so will lead to a zero score for this sub question.

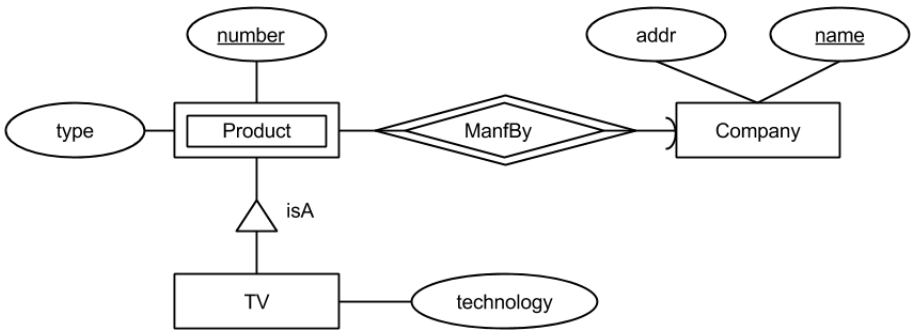


Figure 1: E/R diagram

Answer: E/R approach:
Company(name, addr)
Product(companyName, number, type)
TV(companyName, number, technology)

OO approach:
Company(<u>name</u>, addr)
OtherProduct(<u>companyName</u>, <u>number</u>, type)
TV(<u>companyName</u>, <u>number</u>, type, technology)

Null approach:
Company(<u>name</u>, addr)
Product(<u>companyName</u>, <u>number</u>, type, technology)

(18) Consider two relations of UIUC student/enrollment records: *Student*(<u>netid</u>, department, name, state) and *Enrollment*(<u>netid</u>, <u>courseid</u>, grade). There are 40,000 tuples in *Student* from 100 departments and 50 states, which include every student at UIUC. There are 200,000 tuples in *Enrollment*. Estimate the size, in terms of number of tuples, for

$$(\sigma_{department = \text{“}CS\text{”}} Student) \bowtie Enrollment.$$

Answer:
Solution 1: Since *Enrollment.netid* and *Student.netid* is a foreign-key relationship, every *Enrollentment* tuple will join with exactly one tuple in *Student*. Among them, 1/100 will be selected for the "CS" condition. Thus, $200,000 \times 1 \times \frac{1}{100} = 2000$.
Solution 2: $T(CSStudent \bowtie_{netid} Enrollment)$
$= T(CSStudent) * T(Enrollment)/max(V(CSStudent, netid), V(Enrollment, netid))$
$= (40000/100) * 200000/max((40000/100), 40000)$
$= 2000$

Problem 2 (*22 points*) Query Languages

(1) Answer the following questions based on the database schema provided below for a class registration system.
Class(ClassID, ClassName, InstructorID, Time, Location)
Instructor(InstructorID, Name)
Student(StudentID, Name, Major, GPA, TotalHours)
Register(StudentID,ClassID, PointGrade, CreditHours)

(a) Write a query, in *relational algebra*, to return the names of students with highest GPA. There may be more than one student with the highest GPA; in such case, return all such students. (*4 points*)
Answer: $\pi_{Name}(Student) - \pi_{Student1.Name}(Student1 \bowtie_C Student2)$, C= $Student1.GPA < Student1.GPA$

(b) Write a query, in *SQL*, to return the names of students who have registered for class taught by 'Angrave, L.'. (*4 points*)
Answer: SELECT S.name
FROM Student as S, Register as R, Class as C, Instructor as I
WHERE S.StudentID = R.StudentID
AND R.ClassID = C.ClassID
AND C.InstructorID = I.InstructorID
AND I.Name = 'Angrave, L.'

(c) Write a query, in *SQL*, to return the names of instructors who teach more than 3 classes. The ouput should be in descending order by number of classes.(*4 points*)
Answer: SELECT I.Name, S.cnt
FROM Instructor as I,
(SELECT InstructorID, COUNT(*) as cnt
FROM Class
GROUP BY InstructorID) AS S
WHERE I.InstructorID = S.InstructorID
AND S.cnt > 3
ORDER BY S.cnt DESC

(d) In *SQL*, define a foreign key constraint to ensure whenever a student record is deleted from the *Student* table, his/her records are deleted from the *Register* table as well.(*5 points*)
Answer: ALTER TABLE Register
ADD CONSTRAINT fk_student
FOREIGN KEY (StudentID)
REFERENCES Student(StudentID)
ON DELETE CASCADE;
oracle language is correct too
It is also acceptable if you mention that during constructing the table, you have
FOREIGN KEY (StudentID)

REFERENCES Student(StudentID)
ON DELETE CASCADE;

(e) Write a trigger, in *SQL*, that updates GPA for a student whenever there is an update on *Register* table. GPA is calculated using formula below: (*5 points*)

$$GPA = \sum_{\text{class} \in \text{ all classes taken}} \frac{PointGrade(class) * CreditHours(class)}{TotalHours}$$

Answer: CREATE Trigger updateGrade
AFTER UPDATE on Register
FOR EACH row
UPDATE Student
SET GPA = (GPA*TotalHours + new.PointGrade*new.CreditHours)/(TotalHours + new.CreditHours);
It is also correct if you let GPA = (GPA*TotalHours + new.PointGrade*new.CreditHours)/TotalHours.
There are multiple ways to define this trigger. You can get full credits as long as you use the correct syntax and formula.

Problem 3 (*16 points*) Indexing

(1) Consider the following B+Tree of order 4 (i.e., n=4, each index can hold n keys and n + 1 pointers), shown in the figure below:

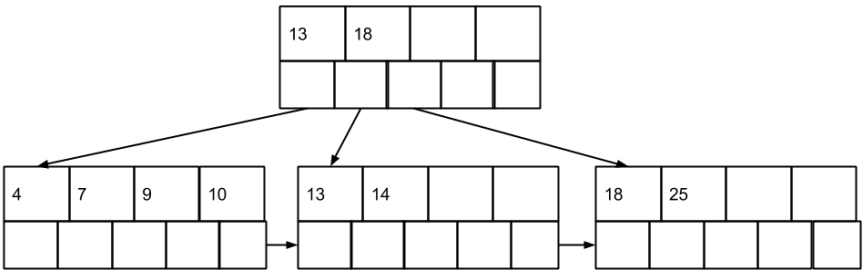
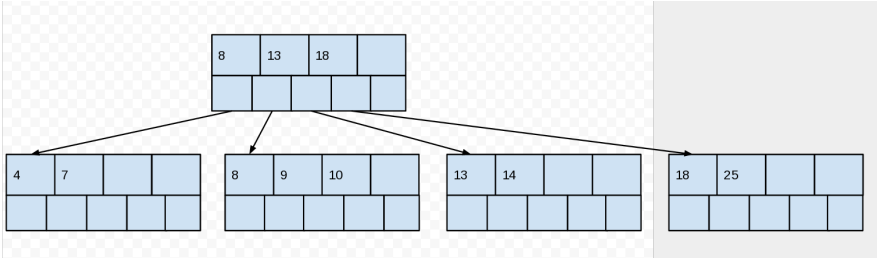


Figure 2: B+Tree

For the following two questions, **please draw the full tree**. Any incomplete drawing will results in **0** points.

(a) Show the resulting tree after inserting key 8. (*4 points*)

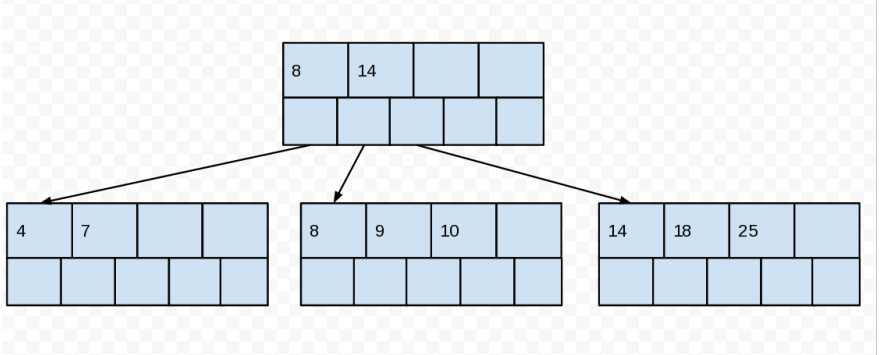
Answer:



There are different ways to do this. You can get full credits as long you maintain B+ tree properties.

(b) Based on the original tree, show the resulting tree after deleting 13. (*4 points*)

Answer:



There are different ways to do this. You can get full credits as long you maintain B+ tree properties.

(2) Consider indexing the following key values using an extensible hash table. Suppose that we insert the keys in the order of: 1, 15, 21, 35.
The hash function h(n) for key n is h(n) = n mod 16; i.e., the hash function is the remainder after the key value is divided by 16. Thus, the hash value is a 4-bit value. Assume that each bucket can hold 2 data items.
You have performed this indexing correctly, and have come up with the following table, as shown in figure below:

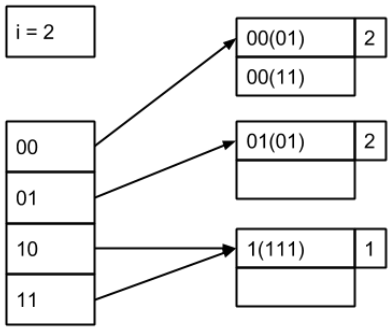
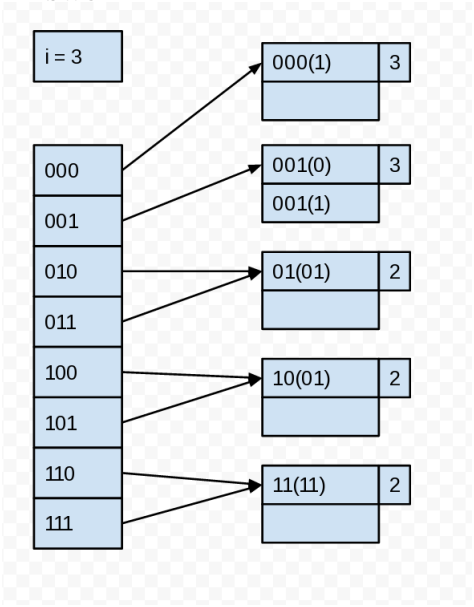


Figure 3: Extensible Hashing Table

Now insert 18 and 41, in this order, into the hash table into the table above. Redraw the table to reflect the new values. Be sure to indicate the number of bits in the hash value that are used in the array. Also, indicate the "nub" value of each block. Again, you can just draw the final table after inserting all the six keys. (*8 points*)
Answer:



Is is also acceptable if you put 1001 and 1111 in one bucket.

Problem 4 (*15 points*) Query Processing

Given relations R and S, assume the size of main memory is M blocks, the size of relation R is $B(R)$ blocks, and the size of relation S is $B(S)$ blocks.

(1) If we are using Block-based nested-loop join algorithm for R JOIN S, what is the proper I/O cost if $B(R) = 8$, $B(S) = 15$, $M = 6$? Feel free to directly apply the corresponding formula.(*4 points*)
Answer: B(R) + B(R) * B(S) / (M - 1) = 32

(2) We are using Block-based nested-loop join algorithm for R JOIN S with settings $B(R) = 8$, $B(S) = 15$, $M = 10$. Note that the memory size is different now. Does this change affect the formula in **sub-question 1**? If yes, please write a new proper formula; if no, use the same formula. Please write down the formula and the calculated I/O cost. (*4 points*)
Answer: B(R) + B(S) = 23; Note that we may store the entire B(R) in the memory. For the detail explaination, please refer the textbook.

(3) If we are using Sort-Merge join algorithm for R JOIN S, what is the proper I/O cost if $B(R) = 8$, $B(S) = 15$, $M = 6$? Feel free to directly apply the corresponding formula. (*4 points*)
Answer: 5B(R) + 5B(S) = 115

(4) If we are using Hash-join algorithm for R JOIN S, what is the proper I/O cost if $B(R) = 8$, $B(S) = 15$, $M = 6$? Feel free to directly apply the corresponding formula. (*3 points*)
Answer: 3B(R) + 3B(S) = 69

Problem 5 (*18 points*) Dynamic Programming

(1) Assume we have 50 tuples for relation A, i.e., $T(A) = 50$, and 30 tuples for relation B. What is the possible minimum T(A JOIN B)? Explanation is not required. (*1 points*)
Answer: 0

(2) For the same question in **problem 5.1**, what is the possible maximum T(A JOIN B)? Explanation is not required. (*1 points*)
Answer: 1500

(3) Given relations A, B, C, D, assume $T(A) = 20$, $T(B) = 50$, $T(C) = 70$, $T(D) = 10$, and the size estimation heuristic: size(R1 JOIN R2) = 0.1 * T(R1) * T(R2). In order to find out the best query plan for **A JOIN B JOIN C JOIN D**, please fill in the following table. (*16 points*) **Answer:**

	Subquery	Size	Cost	Plan
1	AB	100	0	AB
2	AC	140	0	AC
3	AD	20	0	AD
4	BC	350	0	BC
5	BD	50	0	BD
6	CD	70	0	CD
7	ABC	700	100	(AB)C
8	ABD	100	20	(AD)B
9	ACD	140	20	(AD)C
10	BCD	350	50	(BD)C
11	ABCD	700	120	((AD)B)C

You get 1 point off for 1 incorrect cell value.

Problem 6 (*25 points*) Failure Recovery

Consider the following log sequence.

Log ID	**Log**
1	⟨START T1⟩
2	⟨T1, A, 1⟩
3	⟨START T2⟩
4	⟨T2, B, 2⟩
5	⟨COMMIT T2⟩
6	⟨T1, B, 2⟩
7	⟨COMMIT T1⟩
8	⟨START T3⟩
9	⟨T3, A, 3⟩
10	⟨START T4⟩
11	⟨T3, B, 4⟩
12	⟨START T5⟩
13	⟨COMMIT T3⟩
14	⟨T4, C, 5⟩
15	⟨COMMIT T4⟩
16	⟨T5, A, 6⟩
17	⟨COMMIT T5⟩
18	⟨START T6⟩
19	⟨T6, A, 8⟩
20	⟨COMMIT T6⟩

Note: For the questions below, assume the given log sequence is a **UNDO** log.

(a) Suppose we want to perform quiescent checkpointing some time after logID 2. Since quiescent checkpointing can only be perfomed when all active transactions have written a COMMIT or ABORT to the log, indicate where the earliest checkpoint record would be. (*4 points*)

Answer:

Between logID 7 and 8, ⟨CKPT⟩

(b) Suppose that we begin nonquiescent checkpointing right after logID 12. In the space below, indicate where *i*) the start checkpointing record would be, and what it would look like; and *ii*) the earliest end checkpoint record would be, and what it would look like. (*4 points*)

Answer:

1. Between logID 12 and 13, ⟨STARTCKPT(T3,T4,T5)⟩
2. Between logID 17 and 18, ⟨ENDCKPT⟩

(c) Continue from (b). Suppose the system crashes right after logID 16. What is the portion of the log we would need to inspect and which transactions need to be undone? (*5 points*)

Answer:

1. Log portion needs to be inspected: from crash point until the start of the earliest incomplete transaction.
2. Transactions need to be undone: T5

Note: For the questions below, assume the given log sequence is a **REDO** log.

(d) Suppose that we begin nonquiescent checkpointing right after logID 4. In the space below, indicate where *i*) the start checkpointing record would be, and what it would look like; and *ii*) the earliest end checkpoint record would be, and what it would look like. (*4 points*)

Answer:

1. Between logID 4 and 5, ⟨STARTCKPT(T1,T2)⟩
2. After STARTCKPT, ⟨ENDCKPT⟩

(e) Continue from (d). Suppose the system crashes right after logID 16. If ⟨ENDCKPT⟩ was written to the log, indicate the portion of the log we would need to inspect and which transactions need to be redone. (*4 points*)

Answer:

1. Log portion needs to be inspected: from logID 1 to crash point
2. Transactions need to be redone: T1,T2,T3,T4

(f) Now, suppose that we begin nonquiescent checkpointing right after logID 4 and the system crashes right after logID 6. If ⟨ENDCKPT⟩ was not written to the log, indicate the portion of the log we would need to inspect and which transactions need to be redone. (*4 points*)

Answer:

1. Log portion needs to be inspected: from logID 1 to crash point.
2. Transactions need to be redone: T2