FINAL EXAM
CS 564 Introduction to Database Management Systems
Department of Computer Sciences
University of Wisconsin, Madison

Exam Rules:

1) Open book and notes, 60 minutes

2) Please write down your name and student ID number NOW.

3) Please wait until being told to start reading and working on the exam.

4) Calculators are allowed.
1. **(5 points)** What is the difference between logical and physical operators? Give an example of each.

2. **(15 points)** Given two relations R and S with size $B(R) = 10000$ and $B(S) = 8000$:
   (a) **(5 points)** Suppose the amount of memory available is $M = 2001$. Compute the disk IO cost of a nested-loop join for R and S.
(b) **(10 points)** Suppose the amount of memory available is $M=6000$. Is it possible to perform a nested-loop join for $R$ and $S$ with cost no more than 25000 disk IO? Explain your answer.
3. (30 points; 6 points each) Consider the two relations:  
\[ \text{Company}(\text{cname}, \text{city}, \text{president}) \]  
\[ \text{Product}(\text{pname}, \text{maker}, \text{price}) \].

Assume B(Company) = B(Product) = 10000 blocks. Assume further that each block can accommodate 5 records, and that the memory has 101 buffers.

Consider the SQL query:  
\[ \text{SELECT} \ \text{Company.cname, Company.president} \]  
\[ \text{FROM} \ \text{Company, Product} \]  
\[ \text{WHERE} \ (\text{cname} = \text{maker}) \ \text{AND} \ (\text{price} > 100) \ \text{AND} \ (\text{city} = "\text{Urbana}")) \].

(a) Draw a logical query plan tree for the above SQL query. This plan tree should join Company and Product, then perform a selection on price and city, then project out cname and president.

```
π_{cname, president} 
\( \quad \land \quad \) (price > 100) \land (city = "Urbana")
-
∞_{cname = maker}
```

(b) For the logical query plan that you draw in Part a, consider the following physical query plan. First, perform a nested loop join of Company and Product, with Company being the outer relation. Next, pipeline the result of the join into the selection operation. Finally, pipeline the result of the selection operation into the projection operation. Compute the total cost of this physical query plan.

This is put the cost of the nested loop join, which is

\[ \sqrt{\frac{10000}{101-2}} \times 10000 + 10000 \]
(c) Assume that there is an index on attribute cname of relation Company, and that cname is a key for that relation. Can we replace the nested loop join operation in the physical query plan mentioned in Part b with an index based join operation? If so, which relation should be the outer relation in this index based join operation, and what should be the cost of this new physical query plan (which uses the index based join operation)?

Cost is the cost of the index based join. Assuming a hash index, then for each product, we need on average 1.2 page I/O to use the index to look up the address of the page on which the matching Company tuple resides. So the total cost is

\[ 10000 + (10000 \times 5) \times (1.2 + 1) \]

(d) Consider the logical query plan that you have drawn in Part a. Draw a new logical query plan that pushes selections and projections down as far as possible.
(e) Outline at least one scenario where pushing selections down is not desirable. Explain why.

Suppose we join `Product` with `Company` up an index on `Chrome`. Then pushup city = Urbana to below the join is not OK, because we cannot use the index on `Company` to do the join.
4. **(20 points)** Compute the optimal plan for R □ S □ T □ U using the dynamic programming algorithm, assuming the following:

B(R) = 500, B(S) = 200, B(T) = 600, B(U) = 400

The size of a join is estimated as: B(A □ B) = 0.01 * B(A) * B(B)

The cost of a join is estimated to be the cost of the subplans plus the size of the intermediate results (the same as the cost model we have covered in the lecture).

Please draw the table for dynamic programming, to show how you compute the optimal plan.
5. (10 points) Briefly describe the ACID properties.

See notes.
6. (20 points) A database has four elements A, B, C, and D. It has just crashed. Assume we maintain an undo log whose content after the crash is

```
<start T1>
<T1,A,3>
<start T2>
<T2,B,1>
<start ckpt(T1,T2)>
<T1,A,4>
<start T3>
<commit T1>
<T2,B,2>
<T3,C,5>
<commit T2>
<end ckpt>
<start T4>
<T4,D,8>
<start ckpt(T3,T4)>
<start T5>
<T4,D,9>
<T5,A,10>
<commit T5>
<start T6>
<T6,B,12>
```

Assume further that after the crash the four elements have value: A=1, B=2, C=3, and D=4. Recover the database. Clearly indicate which portion of the log you would need to inspect, and which transactions have to be executed again. Show the values of A, B, C, and D after the recovery.
FINAL EXAM
CS 564 Introduction to Database Management Systems
Department of Computer Sciences
University of Wisconsin, Madison

Exam Rules:

1) Closed book and notes, 90 minutes

2) Please write down your name and student ID number NOW.

3) Please wait until being told to start reading and working on the exam.
1. Relational Operators (25 Points)
   a. (5 Points) Briefly describe two alternatives for evaluating a projection with duplicate elimination, such as \( \pi_{\alpha}(R) \). Name one factor that the DBMS makes in choosing between these two options (one or two sentences).

   b. (10 Points) Suppose that you are given relations R with 4000 Pages and S with 20000 pages. Suppose the amount of memory available is \( M = 2001 \). Compute the disk IO cost of a nested-loop join for R and S.
c. (10 Points) Suppose that you are given relations R with 4000 Pages and S with 20000 pages. Suppose the amount of memory available is M = 2001. Compute the disk IO cost of a sort merge join for R and S.

This cost is the cost of sort R + cost of sort S + cost of merge R ∩ S

The cost of sort R is \( \left( \left\lceil \log_{2000} \left( \frac{4000}{2001} \right) \right\rceil + 1 \right) \times 8000 \)

The cost of sort S is \( \left( \left\lceil \log_{2000} \left( \frac{20000}{2001} \right) \right\rceil + 1 \right) \times 40000 \)

The cost of merge the sorted R and S varies. In the worst-case scenario, it is 4000 x 20000. In the best-case scenario, it is (4000 + 20000). [This does not assume some optimization trick such as finding out that the smallest value of R is greater than the biggest value of S.]
2. (15 Points) **Normalization**
   a. (10 Points). Given the relation R(A,B,C,D,E) with functional dependencies A -> BC, C -> D, decompose it into BCNF. Please show the decomposition process.
b. (5 Points) Given $R(A,B,C)$ with constraint $A \rightarrow C$, give an example of a decomposition of $R$ that is lossless and a decomposition of $R$ that is lossy. Does BCNF ensure the lossless join property (that is, the lossless decomposition)?
3. **Relational algebra (15 points).** Consider the two relations

**Company**(cname, city, president)

**Product**(pname, maker, price)

where **cname** is the name of the company, and **maker** is the name of the company that makes the product.

(a) (10 points) Write a relational algebra expression that returns the names of all expensive products made by companies in Madison, where a product is expensive if there is at least one company that makes that product at the price of at least $100.

\[ \Pi_{\text{pname}} (\sigma_{\text{price}>100} \land \text{city=Madison}) \quad \text{Product} \bowtie \text{Company} \quad \text{maker=cname} \]

(b) (5 points) Write a relational algebra expression that returns the names of all products that are made by at least two different companies.

\[ \Pi_{\text{pname}} (\sigma_{\text{maker=maker1} \land \#1 \neq \#2} \land \sigma_{\text{maker=maker2} \land \#5 \neq \#2}) \quad \text{Product} \bowtie \text{Product} \]

\[ \text{renaming operation} \]

\[ \Pi_{\#1} (\sigma_{\#2 \neq \#5} \quad \text{Product} \bowtie \text{Product}) \]

\[ \text{refers to position of attributes} \]
4. Query optimization (30 points).
(a) (15 points) Suppose a SQL query must join three relations A, B, and C. A join order specifies how to join the three relations. For example, the join order (A join B) join C specifies that first we join A and B (with A being the outer join), and then we join the result with C in a pipeline fashion (with C being the inner join). Current RDBMSs consider only certain join orders, not all possible ones. For the above three relations, write down all join orders that current RDBMSs would consider. Explain briefly why they consider only those and not the remaining join orders.
(b) (15 points) Consider again the two relations \( \text{Company}(\text{ename, city, president}) \)
\( \text{Product}(\text{pname, maker, price}) \)

Suppose that the size of relation \( \text{Company} \) is 20,000 pages, with 100 tuples per page, and that the size of relation \( \text{Product} \) is 100,000 pages, with 200 tuples per page. Suppose further that each company makes no more than 20 products, and that \( \text{ename} \) is a key in \( \text{Company} \). Finally, suppose that \( \text{price} \) is distributed uniformly between 1 and 100 in relation \( \text{Product} \).

Consider the following plan: first we do a selection on \( \text{Product} \) with condition \( \text{price} \geq 20 \) AND \( \text{price} \leq 60 \), that is, price is between 20 and 60. We materialize the output of this selection on disk, then join this output with relation \( \text{Company} \) using a sort-merge join, on the condition \( \text{ename} = \text{maker} \). Then we perform on the fly the selection \( \text{city} = \text{Madison} \) followed by the projection of attributes \( \text{ename} \) and \( \text{president} \).

Assume the memory size is 15,000 pages. Compute the cost of the above plan. Showing the formula for the cost (instead of computing the actual number) is fine.

Price is distributed uniformly between 1 and 100, so the output size after the selection on \( \text{Product} \) is \( \frac{100,000 \times (60 - 20)}{100} = 40,000 \).

So the total cost of the plan is:

- The cost of reading \( \text{Product} \): 100,000
- The cost to write the output of \( \text{Product} \) to disk: 40,000
- The cost of sorting more 40,000 pages
- The cost of sorting \( \text{Company} \)
- The cost of merging the two sorted relations. Here, for each tuple in \( \text{Company} \), we will join with at most 20 tuples in \( \text{Product} \). The output of \( \text{Product} \) (under a page of data). So the total cost of merging is 20,000 + 40,000.
5. **Transaction management (15 points).**

(a) (10 points) Briefly describe the ACID property of transaction. Who (the programmer or the system) is responsible for ensuring which property?
(b) (5 points) Briefly explain the idea of checkpointing. Does the database have to freeze (that is, stop accepting new transactions) during checkpointing?