

CS386, Spring 2005, FINAL EXAM

Please answer the 7 questions on separate pieces of paper, answer the parts of each question in order and number your answers. Put your name on each sheet. Work alone.

1. *Operator Implementation* (15 Points): Some query operations can be implemented using only an index, without accessing the actual data in a table.

Consider the `agent` relation from the spy database:

`agent(agent_id, first, middle, last, address, city, country, salary, clearance_id)`

Assume `agent` has a dense B-tree index `I-sal` on `salary`. (Assume that `I-sal` contains data entries pointing to the actual data records.)

(a) Explain how to use `I-sal` to evaluate aggregations on `salary` for `agent`. That is, how can you use `I-sal` to evaluate a query such as

```
SELECT max(salary)
FROM agent
```

without looking at the data records themselves. (Also explain how to handle min, sum, count and average.)

(b) Which of these aggregations on `salary` could be computed from `I-sal` if it were a sparse index?

2. *Algebraic Equivalences* (20 points): For each of the algebraic equivalences below, say whether it

- Always holds
- Holds with conditions
- Does not hold in general

If your answer is “Holds with conditions,” state what the appropriate conditions are. If your answer is “Does not hold in general,” give a counterexample. Assume `q`, `r` and `s` are relations with no duplicates, where `r` and `s` are union compatible, and that `X` is a set of attributes. The symbol `|×|` is natural join, and `P` is a selection condition.

(a) $\pi_X(r - s) \equiv \pi_X(r) - \pi_X(s)$

(b) $\pi_X(r \mid \times \mid q) \equiv \pi_X(r) \mid \times \mid q$

(c) $r \cup r \equiv r$

(d) $(r \text{ UNION-ALL } r) \equiv r$

(e) $\sigma_P(r \cup s) \equiv \sigma_P(r) \cup \sigma_P(s)$

3. *Concurrency Control* (15 points) Consider the following three transactions with their read and write operations.

T1: R₁(A), W₁(A), R₁(C), W₁(C)

T2: R₂(B), R₂(C), W₂(B), W₂(C)

T3: R₃(A), R₃(B)

For each of the schedules below, say whether it is serializable or not. If it is serializable, give an equivalent serial schedule. If not, explain the conflict.

Schedule I

<u>T1</u>	<u>T2</u>	<u>T3</u>
R ₁ (A)		
		R ₃ (A)
W ₁ (A)		
R ₁ (C)		
W ₁ (C)		
	R ₂ (B)	
	R ₂ (C)	
	W ₂ (B)	
		R ₃ (B)
	W ₂ (C)	

Schedule II

<u>T1</u>	<u>T2</u>	<u>T3</u>
R ₁ (A)		
		R ₃ (A)
W ₁ (A)		
R ₁ (C)		
W ₁ (C)		
	R ₂ (B)	
	R ₂ (C)	
		R ₃ (B)
	W ₂ (B)	
	W ₂ (C)	

Schedule III

<u>T1</u>	<u>T2</u>	<u>T3</u>
	R ₂ (B)	
	R ₂ (C)	
	W ₂ (B)	
	W ₂ (C)	
R ₁ (A)		
W ₁ (A)		
R ₁ (C)		
W ₁ (C)		
		R ₃ (A)
		R ₃ (B)

4. *PhP, embedded SQL* (10 points)

Write the database-related code necessary to return one sailor's name from the Sailors table to your program's address space. Indicate what host language you are using. Comment your code to explain each database-related statement, to show that you know what each statement does.

5. *Security* (10 points)

5a. Would you classify "Discretionary Access Control" as a security policy or as a security mechanism?

5b. Assume that Len owns the relation Emp. Consider the following set of SQL statements:

(by Len) GRANT SELECT ON Emp TO Dave WITH GRANT OPTION

(by Len) GRANT SELECT ON Emp TO Jim

(by Dave) GRANT SELECT ON Emp TO Jim

(by Dave) REVOKE SELECT ON Emp FROM Jim CASCADE

After these statements are executed, who has SELECT permission on Emp?

6. *Storage* (15 points)

Consider a 10,000 rpm disk with a sector size of 512 bytes, 2000 tracks per surface, 64 sectors per track, five double-sided platters, average seek time of 10ms and track-to-track seek time of 1ms. Suppose a block size of 8K bytes is used in a file system stored on that disk. In this file system, each record must be stored within a single block.

A file, containing 320,000 records of 100 bytes each, is stored in that file system, in such a way as to minimize the time to read the file sequentially.

What time is required to read the file sequentially? Explain your calculation.

7. *Physical database design* (15 points)

Consider this schema for a relational database:

Prof(SSN, pname, age, specialty, dept_did)

Dept(did, dname, budget, num_majors)

Prof has 100 pages and 10 records per page, Dept has 10 pages and 20 records per page.

Here dept_did is a foreign key referencing the primary key of Dept. The meaning of the remaining fields should be clear. You may assume, as usual, that attribute values are distributed uniformly.

Assume that the workload for this database consists of these three queries, each executed once per day.

- Given a specialty, list the SSNs of professors with that specialty. (Assume there are 100 different specialties among all the professors.)
- Let D be the average budget of all departments, and assume that every department has a different budget amount. List the dids of all departments with budgets less than D.
- Find the SSN and name of all professors who belong to departments with more than 100 majors and who are less than 25 years old. Assume the number of majors per department ranges from 0 to 1000 and the ages of professors range from 20 to 70.

Thus there are no updates in the workload and there is plenty of disc space, so indexes are free.

Design a physical schema (in this case, a set of indexes) for this logical schema that will give the best performance each day for this workload. Be sure to indicate for each index if it is hash or tree, clustered or not, and which of the three alternatives for data entries is used. Also explain how each of the queries will be executed using your physical schema.