Assignment #3

CS 348 - Fall 2022

Due : 11:59 p.m., Thur, Nov 24, 2022

Appeal Deadline: One week after return

(Total weight 10%)

Submission Instruction

This assignment will be submitted through Crowdmark. See the website for more detailed instructions. In particular, do not forget to submit one file per question to make the lives of TAs easier.

This part consists of 6 questions on physical data organization, indexing, query processing, query optimization, and transactions (lectures 11-15). These first 5 questions have a total of 100 marks (10% of the final grade), and the bonus question is of 10 marks (a bonus of 1% to the final grade).

Question 1.

[25 marks in total] Consider the following BCNF schema for a portion of a simple corporate database (type information is not relevant to this question and is omitted):

> Emp (id, name, addr, salary, age, year, deptid)
> Dept (deptid, deptname, floor, budget)

Suppose you know that the following queries are the six most common queries in the workload for this corporation and that all six are roughly equivalent in frequency and importance:

1. List the id, name, and address of employees in a user-specified age range.
2. List the id, name, and address of employees who work in the department with a user-specified department name.
3. List the id and address of employees with a user-specified employee name.
4. List the overall average salary for employees.
5. List the average salary for employees of each age; that is, for each age in the database, list the age and the corresponding average salary.
6. List all the department information, ordered by department floor numbers.

Given this information, and assuming that these queries are more important than any updates, design a physical schema for the corporate database that will give good performance for the expected workload. In particular, decide which attributes will be indexed and whether each index will be a clustered index or an unclustered index. Assume that B+ tree indexes are the only index type supported by the DBMS and that both single- and multiple-attribute keys are permitted. Specify your physical design by identifying the attributes you recommend indexing on via **clustering** or **unclustering** B+ trees. Please briefly explain your choices.

Question 2.

[10 marks in total] Consider a table occupying 1,000,000 disk blocks. There are 501 memory blocks available for query processing. Suppose you have the following two options to improve query performance:

(1) Buy more memory to increase the number of available memory blocks to 1001.

(2) Buy a faster disk to increase the speed of I/O by 10%.

Suppose there are no indexes for this table, if the objective is to speed up the query "SELECT * FROM R WHERE R.A $>$ 100;", which option is more effective now? Briefly justify your answer.

Question 3.

[30 marks in total]

Consider the following schema for an online bookstore:

```
Cust (CustID, Name, Address, State, Zip)
Book (BookID, Title, Author, Price, Category)
Order (OrderID, CustID, BookID, ShipDate)
Inventory (BookID, Quantity, WarehouseID, ShelfLocation)
Warehouse (WarehouseID, State)
```

`Cust` and `Book` represent customers and books, respectively. When a customer buys a book, a tuple is entered into `Order`. `Inventory` records the quantity and shelf location of each book for every warehouse. `Warehouse` records the state where each warehouse is located in. `Price` is numeric. `ShipDate` is an integer representation of a date. In the following, `:today` is a constant denoting the integer representation of today's date.

(a) (10 points) Transform the following query into an equivalent query that 1) contains no cross products, and 2) performs projections and selections as early as possible. Represent your result as a relational algebra expression tree.

$$\pi_{\texttt{Title,Author}}\big(\sigma_{(\texttt{State}=\texttt{"NC"})\texttt{and}(\texttt{AuthorLIKE"\%Kondo"})\texttt{and}(\texttt{ShipDate}>\texttt{:today}-60)}\big(\\ \sigma_{(\texttt{Cust.CustID}=\texttt{Order.CustID})\texttt{and}(\texttt{Book.BookID}=\texttt{Order.BookID})}\big(\\ (\texttt{Cust} \times \texttt{Order} \times \texttt{Book}))))$$

(b) Suppose we have the following statistics:

- $|\texttt{Cust}| = 3,000$; $|\pi_{\texttt{State}}\texttt{Cust}| = 50$;
- $|\texttt{Book}| = 1,000$; $|\pi_{\texttt{Category}}\texttt{Book}| = 10$;
- $|\texttt{Order}| = 60,000$; $|\pi_{\texttt{BookID}}\texttt{Order}| = 1,000$; $|\pi_{\texttt{CustID}}\texttt{Order}| = 3,000$; $|\pi_{\texttt{ShipDate}}\texttt{Order}| = 1,000$;
- $|\texttt{Inventory}| = 40,000$;$|\pi_{\texttt{BookID}}\texttt{Inventory}| = 1,000$;$|\pi_{\texttt{WarehouseID}}\texttt{Inventory}| = 50$;
- $|\texttt{Warehouse}| = 50$;$|\pi_{\texttt{State}}\texttt{Warehouse}| = 50$.

For each of the following relational algebra expressions, estimate the number of tuples it produces. Note that each estimation may build on the previous ones. You may make the same assumptions as in the lecture on query optimization. If you make different or additional assumptions, please state them explicitly.

(i) (5 points) $\sigma_{\texttt{ShipDate}>\texttt{:today}-\texttt{60}}\texttt{Order}$

(ii) (5 points) $\pi_{\texttt{CustID}}(\sigma_{(\texttt{ShipDate}>\texttt{:today}-\texttt{60})}\texttt{Order})$

(c) (10 points) Suppose that

- Each disk/memory block can hold up to 10 rows (from any table);
- All tables are stored compactly on disk (10 rows per block);
- 8 memory blocks are available for query processing.

Suppose that there are no indexes available at all, and records are stored in no particular order. What is the best execution plan (in terms of number of I/O's performed) you can come up with for the query

$$\sigma_{(\texttt{ShipDate}=\texttt{:today})}(\texttt{Order} \bowtie \texttt{Inventory})?$$

Describe your plan and show the calculation of its I/O cost.

Question 4.

[20 marks in total] Given the following histories:

$$\begin{aligned}
H_1 = &\quad \{W_1(x), R_1(x), R_2(z), W_2(x), W_2(y), R_3(x), R_3(z), R_3(y)\} \\
H_2 = &\quad \{R_2(z), W_2(x), W_2(y), R_3(x), W_1(x), R_1(x), R_3(z), R_3(y)\} \\
H_3 = &\quad \{W_1(x), R_1(x), R_2(z), R_3(x), R_3(z), W_2(x), W_2(y), R_3(y)\} \\
H_4 = &\quad \{W_1(x), R_2(z), R_1(x), W_2(x), R_3(x), R_3(z), W_2(y), R_3(y)\}
\end{aligned}$$

a. (12 points) Which pairs of the above histories (6 pairs in total) are conflict equivalent and why?

b. (8 points) Which of the four histories above are serializable and why?

Question 5.

[15 marks in total] Consider the university enrollment database schema:
The meaning of these relations is straightforward; for example, Enrolled has one record per student-class pair such that the student is enrolled in the class.

Student(snum, sname, major, level, age)
Class(name, meets at, room, fid)
Enrolled(snum, cname)
Faculty(fid, fname, deptid)

For each of the following transactions, state the lowest possible SQL isolation level you would use and explain why you chose it.

a. (5 points) Enroll a student identified by her snum into the class named “Introduction to Database Systems”.

b. (5 points) Change enrollment for a student identified by her snum from one class to another class.

c. (5 points) For each class from the Class table, show the number of students enrolled in the class. Classes with no enrollment should also appear in the output with a count 0.

Question 6.

[Bonus 10 marks in total] Continuing with Question 3 (c) with the same assumptions that:

- Each disk/memory block can hold up to 10 rows (from any table);
- All tables are stored compactly on disk (10 rows per block);
- 8 memory blocks are available for query processing.

Suppose there is a B+-tree primary index on `Order(OrderID)` and a B+-tree primary index on `Inventory(BookID, WarehouseID)`, but no other indexes are available. Furthermore, assume that both B+-trees have a maximum fan-out of 100 for non-leaf nodes; each leaf stores 10 rows; and all nodes in both B+-trees are at maximum capacity except the two roots. What is the best plan for the same query in 3(c)

$$\sigma_{(\texttt{ShipDate}=:\texttt{today})}(\texttt{Order} \bowtie \texttt{Inventory})?$$

Again, describe your plan and show the calculation of its I/O cost.