Final Exam in INF-2700

Database Systems

2012-02-20

1 (40%)

Below are some database tables for a task management application. The example data show that task `wash dishess` is of category `home` and is assigned to `Ole` and `Anna` as two sub-tasks, and that `Anna` finished her part of the `wash dishes` task on `2012-02-20`.

- `People`

Pid	Name
2	Karin
6	Tom
7	Ole
9	Anna

- `Tasks`

Tid	Title	Category	Size
101	change light bulb	home	1
102	wash dishes	home	8
103	clean desk	job	3
105	bake cake	home	10
108	dance	fun	12
110	walk dog	home	5

- `DoTask`

Pid	**Tid**	SubSize	Status	UpdatedOn
6	101	1	done	2012-02-18
7	102	3	todo	2012-02-19
9	102	5	done	2012-02-20
7	103	3	todo	2012-02-20
9	105	8	done	2012-02-20
2	108	12	done	2012-02-20

The *primary keys* of the tables are in **bold** font.

Foreign keys:

- `DoTask`
  - `Pid`: references `Pid` of `People`
  - `Tid`: references `Tid` of `Tasks`

The person with `pid` has the responsibility for (maybe part of) the task with `tid`.

Write queries to find the required information.
Queries 1–5 must be formulated in *both relational algebra and SQL*.
Queries 6–10 need only be formulated in *SQL*.

1. Names of all people.

   The result for the example database is:

Name
Karin
Tom
Ole
Anna

2. Tasks (titles and sub-sizes) assigned to `Ole`.

   The result for the example database is:

Title	SubSize
wash dishes	3
clean desk	3

3. Names of people who share the `wash dishes` task.

   The result for the example database is:

Name
Ole
Anna

4. Titles of tasks that have not been assigned to anybody.

   The result for the example database is:

Title
walk dog

5. Names of people who do not share a category of tasks with any other people.

   The result for the example database is:

Name
Karin

6. List of Ole's `todo` tasks (together with sub-sizes and update date) in ascending order of updated date.

   The result for the example database is:

Title	SubSize	UpdatedOn
wash dishes	3	2012-02-12
clean desk	3	2012-02-20

7. Total sub-sizes finished on `2012-02-20`.

   The result for the example database is:

FinishedSize
25

8. Tasks not completely done (sum of finished sub-sizes, if any, is less than the size of the task).

   The result for the example database is:

Title
wash dishes
clean desk
bake cake
walk dog

9. Tasks assigned to multiple people.

   The result for the example database is:

Title
wash dishes

10. Date(s) with the most amount of finished work.

    The result for the example database is:

date	FinishedSize
2012-02-20	25

2 (20%)

We now consider the physical design of the table files. Assume that the files are organized as the following:

- `People` file uses *hash file organization* with hashing on attribute `Pid`.
- `Tasks` file uses *hash file organization* with hashing on attribute `Tid`.
- `DoTask` file uses *hash file organization* with hashing on attribute `Pid` *and* has a *hash index* on attribute `Tid`.

Assume the following about the data sizes:

- The sizes of both disk blocks and buffer pages are 4 kilo bytes.
- There are 100 buffer pages allocated to your algorithms.
- The sizes of data records of all three tables are 100 bytes.
- There 10000 people, each being assigned 400 sub-tasks.
- Every task is assigned to 4 people as 4 sub-tasks.
- You may make further assumptions.

Answer the following questions. To keep your answers focused, consider only static hashing.

1. How are `DoTask` data organized on disk? Please draw a figure to illustrate the data organization.
2. Sketch an algorithm to make a natural join of the `People` and `DoTask` tables.
3. What is the performance overhead of your algorithm for the above question?
4. Sketch an algorithm to make a natural join of the `Tasks` and `DoTask` tables.
5. What is the performance overhead of your algorithm for the above question?

3 (20%)

Answer the following questions. Try to give formal definitions of the corresponding concepts.

1. What is *functional dependency* $\texttt{P} \rightarrow \texttt{Q}$ of a relation instance r?

A	B	C
1	x	t
1	y	t
2	z	u

   Given the above relation instance, check if the following functional dependencies are satisfied:

   - $\texttt{A} \rightarrow \texttt{B}$
   - $\texttt{B} \rightarrow \texttt{C}$
   - $\texttt{A} \rightarrow \texttt{C}$
   - $\texttt{AC} \rightarrow \texttt{B}$
   - $\texttt{AB} \rightarrow \texttt{C}$

2. What is a *schema decomposition*? What is the purpose of making a schema decomposition?
3. What is a *lossless* schema decomposition?
4. What is a schema decomposition that *preserves functional dependencies*?

4 (20%)

1. What is an *ACID transaction*?
2. Describe the *two-phase locking* protocol (*2PL*), *strict two-phase locking* protocol (*S2PL*), and *rigorous two-phase locking* protocol (*R2PL*).
3. Discuss the advantages and shortcomings of these protocols.

–END–

FACULTY OF SCIENCE AND TECHNOLOGY
University of Tromsø, N-9037 Tromsø, Phone 77 64 40 01, Telefax 77 64 47 65