Written Assignment 1

Assigned: Feb 17, 2022

Part I. DBMS Architecture & Storage Devices

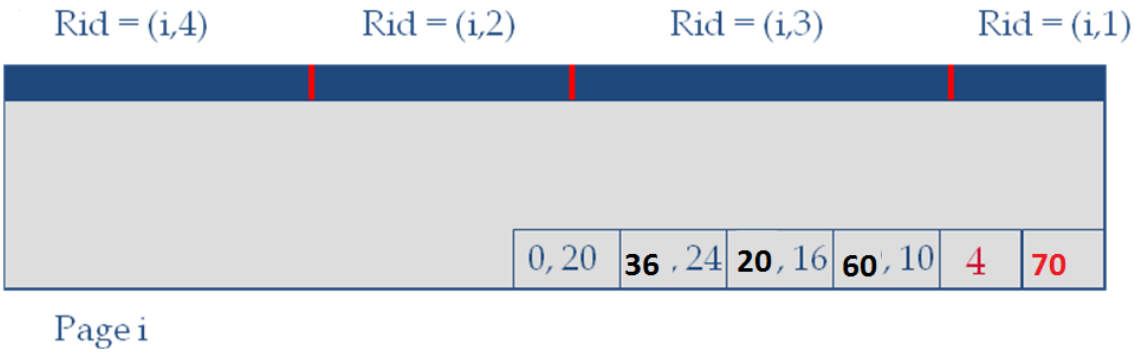
1. Briefly explain (logical and physical) data independence in your own words. Give examples where logical and physical data independence (one example for each) is useful.

2. What are main differences between embedded and client-server DBMS interfaces? Explain how could a DMBS deliver query results larger than main-memory?

3. Explain why the write performance of an SSD will degrade over time.

Part II. Data Organization & Buffer Manager

1. Suppose we store the pointer (or starting address of a record) in a page by using the offset to the beginning of the page (i.e., the first byte in a page has an offset of 0, the 2nd byte in a page has an offset of 1, and so on and so forth). Each slot entry is of the format: (record offset, record length). Consider the variable-length record page organization instance below:



(1) Show the content of the page after the deletion of Rid=(i,3)

(2) Now, after step (1) has been completed, show the content of the page after the insertion of a new record to this page with 30 bytes. And what's the Rid of this new record after the insertion is done?

1) Show the content of the page after the deletion of Rid=(i,3)

2) Now, after step 1) has been completed, show the content of the page after the insertion of a new record to this page with 20 bytes. And what's the Rid of this new record after the insertion is done?

2. In this exercise, you will compare the LRU (least recent used) and Clock eviction policy for a given workload. Here are the assumptions you must make:

- Assume there are four page slots your buffer manager must manage: P1, P2, P3, and P4.

LRU					
Step	Page Read	Buffer Frame			
		P1	P2	P3	P4
1	G				
2	F				
3	E				
4	D				
5	G				
6	C				
7	F				
8	E				
9	D				
10	C				
11	D				
12	B				
13	A				
14	D				
15	F				
No. of Pages Evicted:					

- All four slots are empty to start. The clock hand for clock algorithm starts from P1 and go forward.
- When the buffer pool has unused slots (such as at the beginning, when all four slots are empty), it will put newly read data in the leftmost empty slot (e.g., if slots 2 and 3 are free, it will use slot 2).
- The pages to be read from disk are labeled A through G.
- For each access the page is pinned, and then immediately unpinned.

Below are two tables for describing the contents of the buffer pool at each step. A page is read at the beginning of each step and unpinned at the end. You should record, in the table, the contents of each Buffer Page after the new page has been read in.

Part III. Indexes

1. Suppose that the page size is 1K bytes, and the each (fixed-sized) record size is 100 bytes. You can ignore meta information like page header and bitmaps in the following calculation. Given 1 million records in a table, answering the following questions:

(1) Suppose the table is stored in a heapfile, how many pages are in this file?

(2) We will build an unclustered alternative 2 B+ tree on this table over an attribute A, how many index entries are contained in the leaf level?

(3) Suppose A is integer type of 4 bytes, what's the size of an index entry (assuming a rid/pid is 4 bytes)?

Clock					
Time	Page Read	Buffer Frame			
		P1	P2	P3	P4
1	G				
2	F				
3	E				
4	D				
5	G				
6	C				
7	F				
8	E				
9	D				
10	C				
11	D				
12	B				
13	A				
14	D				
15	F				
No. of Pages Evicted:					

(4) Suppose each leaf level page is 70% filled, how many leaf level pages are needed?

(5) What is the height of the B+ tree (suppose each index level page is also 70% filled)?

(6) Given a query: find all index entries with 100<A<500, suppose there are 300 matching data entries for this query, what's the IO cost of this query with the B+-Tree described as (5)?

(7) Given a query: find all records with 100<A<500, suppose there are 300 matching records for this query, what's the IO cost of this query with the B+-Tree described as (5)?

(8) With this unclustered index, what's the IO cost of inserting and deleting a record?

(9) Answer (3)-(8) again suppose now we are using a clustered alternative 1 index over A.

2. Suppose that we are using extensible hashing on a file that contains records with the following search-key values: (2369, 3760, 4692, 4871, 5659, 1821, 1074, 7115, 1620, 2428, 3943, 4750, 6975, 4981, 9208).

Load these records into a file in the given order using extensible hashing. Assume that every bucket can store up to four values. Show the structure of the directory every 3 insertions, and the global and local depths. Use the hash function: h(k) = k mod 128 and then apply the extensible hashing index.

3. Suppose we are using linear hashing as discussed in class. Assume two records per bucket and the hash function $h(x) = x$. The intermediate hash functions $h_i(x)$, are given by the last i bits of $h(x)$. Assume a split is initiated whenever an overflow bucket is created. Starting with an empty hash file with the hash function $h_1(x)$ and two buckets show the significant intermediate steps (overflows and splits) when keys are inserted in the order given below:

$$(1,3,2,8,6,10,9,7,5,11)$$

Part IV. Relational Algebra

1. Consider a database with the following schema (when more than one attribute is underlined, the combination of them is the primary key):

Customer (cname, address)
Dealership (dname, address)
Car (model, manufacturer)
Visits (cname, dname, times_a_year)
Likes (cname, model)
Serves (dname, model, price)

Write the following queries in **relational algebra**:

(1) Find all customers who has visited "Ford Provo" ("Ford Provo is a dealership name).

(2) Find all dealerships that serve both Toyota and Ford.

(3) Find all dealerships that serve at least one of the car models John likes for no more than $10,000.

(4) For each dealership, find all car models served at this dealership that are liked by none of the customers who visit that dealership.

(5) Find all customers who visit *only* those dealerships that serve some car models they like.

(6) Find all customers who have visited *every* dealership that serves some car models they like.

(7) Find those customers who like all cars Mike likes.

(8) Find those customers who like exactly the same set of cars as Mike does.

(9) For each car model, find the dealership(s) that serves it at the lowest price.

(10) Find the dealership name(s) visited by all customers from SLC (SLC is an address).

2. There is an alternative relational algebra operator division ($/$). Consider two relation R and S whose schemas are $(A_1, \cdots, A_m, B_1, \cdots, B_n)$ and $(B_1, \cdots, B_n)$ respectively. R/S returns a relation of schema $(A_1, \cdots, A_m)$ where each record p is associated with all tuple s of S in R (i.e., you can always find (p, s) in R). Use the relational algebra operator we discuss in class to express R/S.