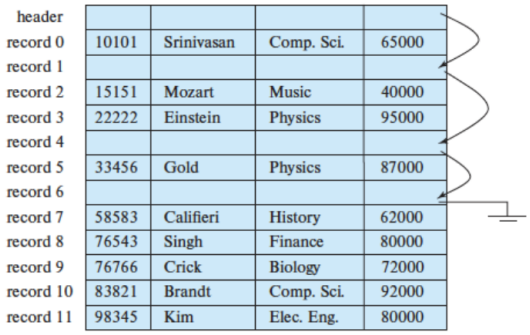
Homework 2 Databases

Jan C. Bierowiec

1. (30') Show the structure of the file in the figure below after each of the following steps:
   (a) Insert (24556, Turnamian, Finance, 98000).
   (b) Delete record 2.
   (c) Insert (34556, Thompson, Music, 67000).



header				
record 0	10101	Srinivasan	Comp. Sci.	65000
record 1				
record 2	15151	Mozart	Music	40000
record 3	22222	Einstein	Physics	95000
record 4				
record 5	33456	Gold	Physics	87000
record 6				
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000
record 11	98345	Kim	Elec. Eng.	80000

Figure: File with free list after deletion of records 1, 4, 6.

Original Figure, remade in LaTeX:

Record	ID	Name	Department	Salary
header				
record 0	10101	Srinivasan	Comp. Sci.	65000
record 1				
record 2	15151	Mozart	Music	40000
record 3	22222	Einstein	Physics	95000
record 4				
record 5	33456	Gold	Physics	87000
record 6				
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000
record 11	98345	Kim	Elec. Eng.	80000

(a) (30') Show the structure of the file in the figure below after each of the following steps:
   i. Insert (24556, Turnamian, Finance, 98000).

   Below is how the structure of the file would look like:

Record	ID	Name	Department	Salary
header				
record 0	10101	Srinivasan	Comp. Sci.	65000
record 1	24556	Turnamian	Finance	98000
record 2	15151	Mozart	Music	40000
record 3	22222	Einstein	Physics	95000
record 4				
record 5	33456	Gold	Physics	87000
record 6				
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000
record 11	98345	Kim	Elec. Eng.	80000

ii. Delete record 2.

Below is how the structure of the file would look like:

Record	ID	Name	Department	Salary
header				
record 0	10101	Srinivasan	Comp. Sci.	65000
record 1	24556	Turnamian	Finance	98000
record 2				
record 3	22222	Einstein	Physics	95000
record 4				
record 5	33456	Gold	Physics	87000
record 6				
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000
record 11	98345	Kim	Elec. Eng.	80000

iii. Insert (34556, Thompson, Music, 67000).

Below is how the structure of the file would look like:

Record	ID	Name	Department	Salary
header				
record 0	10101	Srinivasan	Comp. Sci.	65000
record 1	24556	Turnamian	Finance	98000
record 2	34556	Thompson	Music	67000
record 3	22222	Einstein	Physics	95000
record 4				
record 5	33456	Gold	Physics	87000
record 6				
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000
record 11	98345	Kim	Elec. Eng.	80000

2. (60') Consider a database with a single table T(song_id, song_name, artist_id, duration, number_of_streams), where `song_id` is the primary key, and all attributes are of the same fixed width. Suppose T has 10,000 tuples that fit into 500 pages. Ignore any additional storage overhead for the table (e.g., page headers, tuple headers). Additionally, assume:

- The DBMS does not have any additional metadata (e.g., sort order, zone maps).
- T does not have any indexes (including for primary key `song_id`).
- None of T's pages are in the buffer pool. The DBMS has an infinite buffer pool.
- Content-wise, the tuples of T will always make each query run the longest possible and do the most page accesses.
- The tuples of T can be in any order (keep this in mind when computing the minimum versus maximum number of pages that the DBMS will potentially have to read and think of all possible orderings).
- There are 100 pages per attribute.

```
SELECT MAX(number_of_streams) FROM T
    WHERE duration > 100 AND artist_id == 123321 ;
```

Part a. i. Suppose the DBMS uses the decomposition storage model (DSM) with implicit offsets. How many pages will the DBMS potentially have to read from disk to answer this query? (Should be a range between the best case and the worst case, and why.)

The range would be from 200 to 300. The DBMS would have to read potentially between 200 and 300 pages from the disk to answer this query.

The best case is 200 pages. The reason for this is because, there are 100 pages per attribute. 100 pages is needed to find duration and another 100 to find artist.id for all the tuples. The best case would thus be 200 pages needed to be read.

The worst case is 300 pages. The worst case scenario has an additional 100, because for T's content, the number of streams for all tuples must be accessed as well. In other words, meaning, another 100 pages must be read. The worst case would thus be 300 pages needed to be read.

ii. Suppose the DBMS uses the N-ary storage model (NSM). How many pages will the DBMS potentially have to read from disk to answer this query? (Should be a range between the best case and the worst case, and why.)

The range would be from 300 to 500. The DBMS would have to read potentially between 300 and 500 pages.

For the best case it would read 300 pages, because it would read through the 100 pages of the duration, artist.id as well as the number of streams for all tuples accessed as well, which totals. 300.

For the worst case scenario, the DBMS would have to read all 500 pages from the disk to answer this query. This is because the query needs to find the duration as well as the artist.id for all tuples, which means that, all pages must be accessed.

```
SELECT song_name, artist_id FROM T
    WHERE song_id = 15445 OR song_id = 15645 OR song_id = 15721;
```

Part b. i. Suppose the DBMS uses the decomposition storage model (DSM) with implicit offsets. What is the minimum and maximum number of pages that the DBMS will potentially have to read from disk to answer this query?

The range would be from 3 to 106.

The best case would be 3 pages. This was concluded based on the supposition that

all three primary keys appear on the first page. Due to the fact that all attributes are of the same fixed width, that would mean that each attribute of song id=15445 and song id=15645 will also appear on the same page. What this means is that 1 page will be needed to be read to find the three primary keys and 2 additional pages need to be read to access the song_name, artist.id as well as the other corresponding offsets.

The worst case would be 106 pages. This is because there are 100 pages per attribute. This means that, in the worst case, we scan through all 100 pages to find the three primary keys. In the worst case, the primary keys will be located on different pages. Since all attributes are of the same fixed width, each attribute of song.id=15445, song.id=15645 and song.id=15721 will also appear on different pages. Hence we must read 3 pages to access each attribute at their corresponding offsets. Thus, we read 6 pages in total to access song_name and the artist.id.

ii. Suppose the DBMS uses the N-ary storage model (NSM). What is the minimum and maximum number of pages that the DBMS will potentially have to read from disk to answer this query?

The range would be from 1 to 500.

For the best case, it would only be 1 page. This is because if the tuples of all three primary keys are found to be on the first page, then there is no need to look in other pages since all the attributes are stored together.

For the worst case, it would be all 500 pages. This is because if at least one tuple with a matching primary key is located on the last page. We must therefore scan through every page.

```
SELECT song_id, number_of_streams FROM T
   WHERE duration = (SELECT MIN(duration) FROM T);
```

Part c. i. Suppose the DBMS uses the decomposition storage model (DSM) with implicit offsets. What is the minimum and maximum number of pages that the DBMS will potentially have to read from disk to answer this query?

The range would only be one number. It would be 300 pages for the minimum and maximum.

It would be 300 pages for the minimum and maximum. 100 pages would be needed for the inner select, then an additional 100 pages to get the song.id and number of streams since the buffer pool will have the duration pages from the inner select. The tuples, content-wise make the queries always run for the longest time, so the only consideration can only be the different orderings of the tuples.

ii. Suppose the DBMS uses the N-ary storage model (NSM). What is the minimum and maximum number of pages that the DBMS will potentially have to read from disk to answer this query?

The range would only be one number. It would be 500 pages for the minimum and maximum.

It would be 500 pages for the minimum and maximum. All the pages would need to be scanned to find the minimum duration, and there is a potential for the worst case scenario, which is a part of scanning all 500 pages, to needing to scan all again to fetch song.id as well as the number of streams).

3. (10') Consider the following relation **Cars**:

Brand	Type	Color	Risk
Opel	Corsa	Grey	Low
Opel	Corsa	Red	Medium
Peugeot	206	Black	Medium
BMW	A	Black	High

Construct a bitmap index for the attributes `Brand` and `Color` for this table.

Bitmap index for the attributes `Brand`:

Opel	Peugeot	BMW
1	0	0
1	0	0
0	1	0
0	0	1

Bitmap index for the attributes `Color`:

Grey	Red	Black
1	0	0
0	1	0
0	0	1
0	0	1