5. Exercise

Timo Bergerbusch 344408
Thomas Näveke 311392
Shu Xing 381176

17.12.2017

Exercise 5.1

Important numbers:

B: $\frac{20\,000\,000 \text{ Records}}{40\frac{\text{Records}}{\text{Page}}} = 500\,000$ Pages

R: $40\frac{\text{Records}}{\text{Page}}$

1.a) $\sigma_{ID=500}$

Answer: Use path A3 for this Equality Selection.
Explanation:
Sorted: $Dlog_2B \approx 19D$
Clustered B^+ tree: $D(1 + log_G 0.15B)$
Hash index: $2D < 19D$
If A2 is cheapest, then $D(1 + log_G 0.15B) < 2D \Rightarrow log_G 75000 < 1 \Rightarrow G > 75000$, unpractical!

1.b) $\sigma_{ID\neq500}$

Answer: Use path A1 for this scan of relation
Explanation:

type	formula	cost
sorted	$B \cdot D$	$500\,000D$
B^+ tree	$0.15BD + 1.5BD$	$825\,000D$
hash index	$BD(R + 0.125)$	$20\,062\,500D$

1.c) $\sigma_{ID>500 \wedge ID<1500}$

Answer: A2 for range selection
Explanation:
Sorted: $D(log_2B + \#matchingPages)$
Clustered B^+ tree: $D(log_G 0.15B + \#matchingPages)$
Hash index: $BD = 500\,000D$, obviously larger than the other two.

If cost(A2)<cost(A1), then $log_G 0.15B) < log_2 B \approx 19 \Rightarrow G > 75\,000^{\frac{1}{19}} \approx 1.8 \Rightarrow G \geq 2.$ So usually A2 is cheaper than A1.

2

Nest: 200 bytes/record, 200 tuples
Egg: 10 bytes/record, 5000 tuples
EggInNest: 50 bytes/ record, 100 000 tuples
2000 bytes/page, 7 buffer pages, cost(an index access)=2 I/Os

2.a)

Nest: $\frac{2000 bytes/page}{200 bytes/record} = 10$ records/page $\Rightarrow \frac{200 records}{10 records/page} = 20$ pages
Egg: $\frac{2000 bytes/page}{10 bytes/record} = 200$ records/page $\Rightarrow \frac{5000 records}{200 records/page} = 25$ pages
EggInNest: $\frac{2000 bytes/page}{50 bytes/record} = 40$ records/page $\Rightarrow \frac{100\,000 records}{40 records/page} = 2500$ pages

2.b) i.

Costs for using block-nested loop join:
M=25 I/Os, N=2500I/Os, B=7
Costs=$M + \lceil \frac{M}{B-2} \rceil * N = 25 + \lceil \frac{25}{7-2} \rceil * 2500 = 12525$ I/Os

2.b) ii.

Costs for using index-nested loop join with a clustered hash index:
There exists a hash index on eid on Egg. $\Rightarrow$ Loop of Egg is the inner loop.
M=2500 I/Os, N=25 I/Os
Costs=$M + (tuplesOfOutLoop * CostOfFindingMatchingInnerTuples) = 2500 + 100\,000 \times 2 = 202500$ I/Os

2.b) iii.

Both relations are sorted on join colum. $\Rightarrow$ The Sort-Merge Join is the best with
costs = M+N = 2500 + 25 = 2525 I/Os

Exercise 5.2

1.

Split the computing into two parts:

- Finding the first suitable records

1. through the stated average size and load we retrieve $0.15 \cdot B$ data entries
2. the height of such a tree then is given by $log_G 0.15 \cdot B$
3. Since every one has to be read we multiply it by D

$\Rightarrow D \cdot (log_G 0.15 \cdot B)$

- find first following non-suitable record We then continue reading the index structure until we encounter a record that is not suitable. Through the clustering we know that there are no suitable records following. So we read the number of suitable records = $\#matchingPages$ many records
$\Rightarrow D \cdot \#matchingPages$

$\Rightarrow D \cdot (log_G 0.15 \cdot B) + D \cdot \#matchingPages = D \cdot (log_G 0.15 \cdot B + \#matchingPages)$

2.

- read the page of the hash-group, which contains all entries that satisfy the equality $\Rightarrow D$
- read the data entry satisfying the equality $\Rightarrow D$

$\Rightarrow D + D = 2D$

3.

- the costs of retrieving the file containing the data computes as in Exercise 5.2.1 to $D \cdot (\log_G 0.15 \cdot B)$
- the costs of finding the data entry is constant D
- rewriting the index page and datafile takes $2D$

$\Rightarrow D \cdot (\log_G 0.15 \cdot B) + D + 2D = D \cdot (3 + \log_G 0.15 \cdot B)$

Exercise 5.3

- Regarding `Q1`:
Equality Selection based on its primary key sno:

type	formula	cost
heap	$\frac{1}{2}BD$	$500D$
Sorted	$D \cdot \log_2 B$	$9.97D$
Clustered Tree Index	$D \cdot (1 + \log_G 0.15B)$	$2.09D$
Unclustered Tree Index	$D \cdot (1 + \log_G 0.15B)$	$2.09D$
Unclustered Hash Index	$2D$	$2D$

$\Rightarrow$ for `Q1` a unclustered hash index on sno would be the best option. A(n) (un)clustered tree index would only be very slightly less efficient.

- Regarding `Q2`:
Range Selection based on the salary:
Obviously a tree index is the most efficient for such queries. Also having a clustered tree index would be more efficient than an unclustered tree index, since otherwise each suitable

records would cost 1 page I/O. A clustered tree index can read the records continuously which results in less page I/Os.
So a clustered tree index would be the best choice.

- Regarding `Q3`:
  maybe indexing sno?