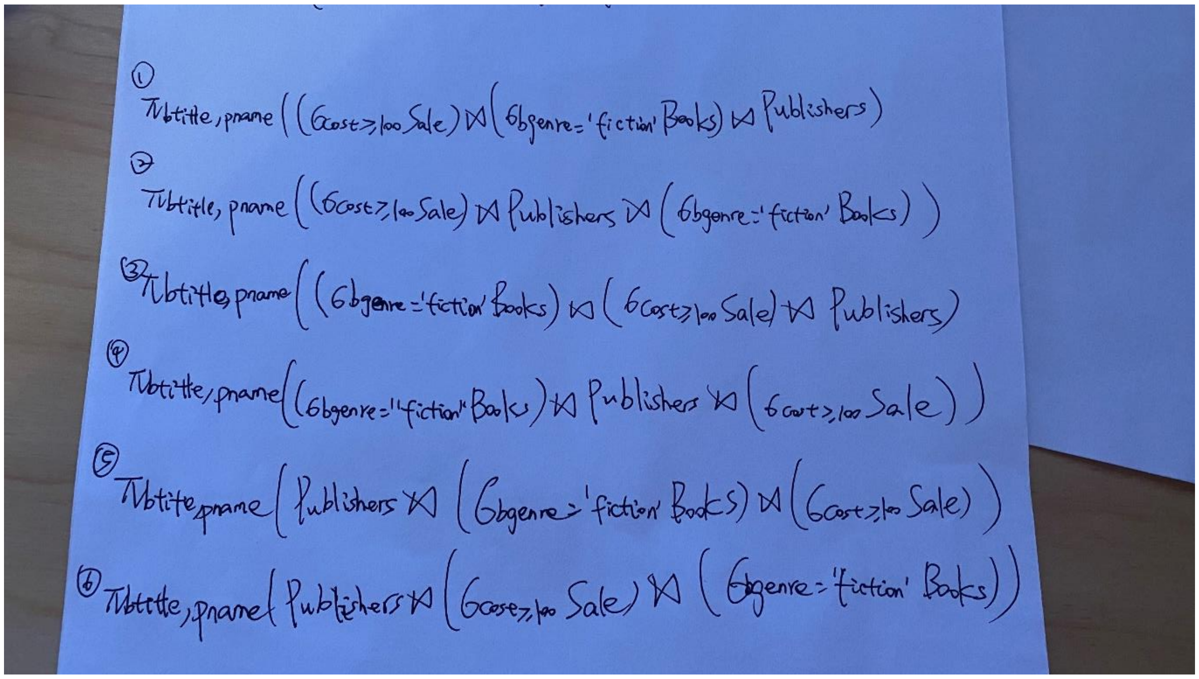
Dayu Liu
0028023243
CS448 Homework 3

Part 1:
# of pages: 10,000,000 records * (128/2048) = 625000 pages

1. # of passes: $\lceil \log_2 625000 \rceil$ + 1 = 20 +1 = 21 passes
   Cost: 2 * 625000 * (# of passes) = 2 * 625000 * 21 = 26250000
2. Number of passes: 1 + $\lceil \log_{20-1} \lceil 625000/20 \rceil \rceil$ = 1 + $\lceil \log_{19} \lceil 31250 \rceil \rceil$ = 1 + 4 = 5 passes
   Cost: 2 * 625000 * (# of passes) = 2 * 625000 * 5 = 6250000
3. Assume we are using alternative 1, Cost:
   Traverse from root to the left-most leaf + all leaf = 3 + 625000 = 625003

Part 2:

a) Assume all data are equally distributed:
   Books: 10000/20 = 500 books
   Sales: 200000*(1000-100/1000) = 180000 sales
   Publishers: 100 publishers



① $\pi_{btitle, pname}((\sigma_{cost \geq 100} Sale) \bowtie (\sigma_{bgenre='fiction'} Books) \bowtie Publishers)$

② $\pi_{btitle, pname}((\sigma_{cost \geq 100} Sale) \bowtie Publishers \bowtie (\sigma_{bgenre='fiction'} Books))$

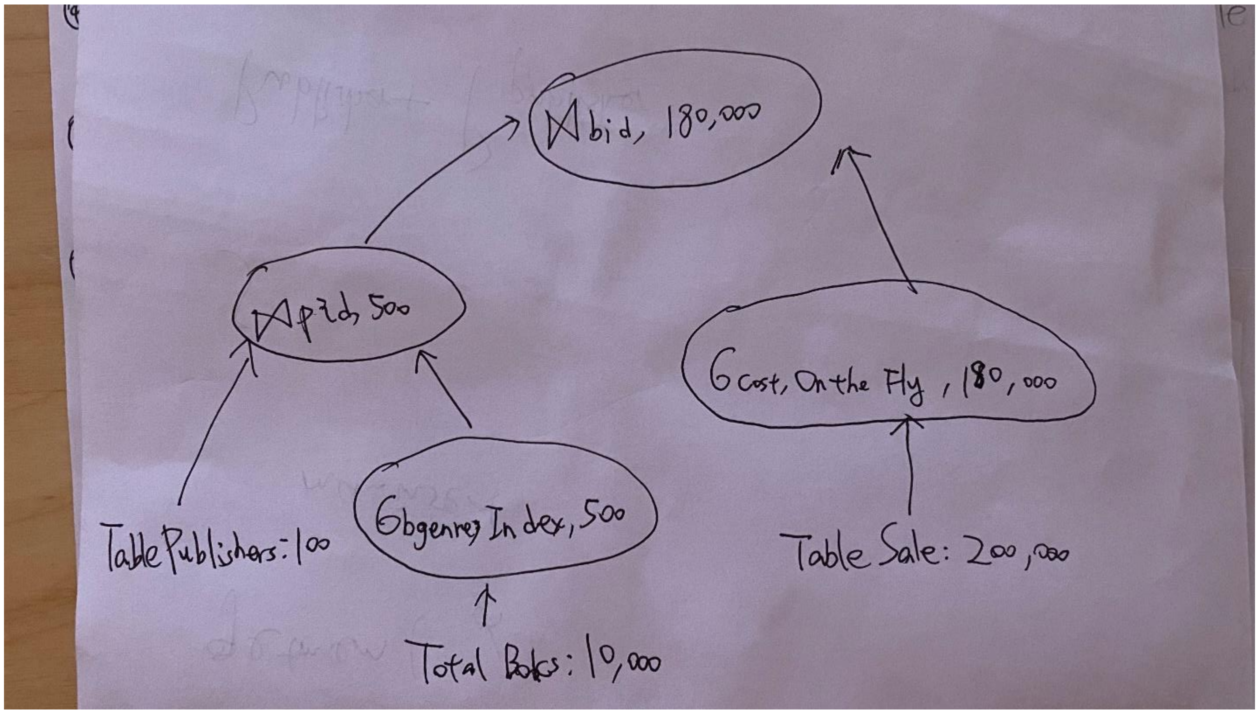
③ $\pi_{btitle, pname}((\sigma_{bgenre='fiction'} Books) \bowtie (\sigma_{cost \geq 100} Sale) \bowtie Publishers)$

④ $\pi_{btitle, pname}((\sigma_{bgenre="fiction"} Books) \bowtie Publishers \bowtie (\sigma_{cost \geq 100} Sale))$

⑤ $\pi_{btite, pname}(Publishers \bowtie (\sigma_{bgenre='fiction'} Books) \bowtie (\sigma_{cost \geq 100} Sale))$

⑥ $\pi_{btitle, pname}(Publishers \bowtie (\sigma_{cost \geq 100} Sale) \bowtie (\sigma_{bgenre='fiction'} Books))$

b)



c)

I/Os by each operator:

Assume on average there will be 1.2 I/Os to get data entry in index, plus 1 I/O to get (the exactly one) matching tuple. So the cost of finding matching S tuples is 2.2 I/Os each tuple

Select b.genre:

    # of tree height + # of scan for each qualifying tuple = 3 + 500 = 503 I/Os

Select s.cost:

    # of total tuples/cost of base table scan = 200,000/20 = 10000 I/Os

Join on pid:

    # of scan for outer table + # of tuples in each page of outer table * # of pages of outer table * cost of finding matching S tuples
    = 100/20 + 20 *100/20 * 2.2 = 225 I/Os

Join on bid:

    # of scan for outer table + # of tuples in each page of outer table * # of pages of outer table * cost of finding matching S tuples

= 500/20 + 20 *500/20 * 2.2 = 1125 I/Os

Cost of writing out the final result: 180000 / 20 = 9000 I/Os

Total I/Os
503 + 10000 + 225 + 1125 + 9000 = 20853 I/Os

Part 3:

1) Plan 1: 500 + 500 * 2000 + 300 + 300 * 100 = 1030800 I/Os
   Plan 2: 500 + 500 * 100 * 2000 = 100000500 I/Os
2) Plan2 with temporary table:
   Cost of I/Os:
   s*p (Table Student joins with temporary table) + p (save temporary table to disk) +
   g*c (iterate all grades, courses tuples to construct temporary table)
   = 500*100 + 100 + 2000 * 100 = 250100 I/Os
   The cost of this optimized plan is much cheaper than original right-deep plan because we do not need to iterate to get every join permutation between grades and courses for every student.
   It is also cheaper than original left-deep plan because the choice of joining students and grades is a bad choice that cost a huge amount of I/Os.
3) The cheapest query evaluation plan that I can come is: $((c \bowtie g) \bowtie s)$ with left-deep join.
   Courses will be the first table because it has the least pages. $(c \bowtie g)$ is the first join operator because it produces the least pages as result.
4) I think the most important factor is the (estimated) number of pages produced by each join predicate. To reduce the cost, we would like to join tables which has the highest selectivity first, this way we end up joining smaller tables overall and save tons of I/Os.