CS 186 Fall 2024

Introduction to Database Systems

Alvin Cheung

DIS 7

1 Selectivity Estimation

Consider two relations R(a, b, c) and S(a), with 1000 tuples and 500 tuples respectively.
We have:

- An index on R.a with 50 unique integer values uniformly distributed in the range [1, 50]
- An index on R.b with 100 unique float values uniformly distributed in the range [1, 100]
- An index on S.a with 25 unique integer values uniformly distributed in the range [1, 25].

Assume that columns are independent (R.a, R.b, and R.c are independent).
Use selectivity estimation to estimate the number of tuples produced by the following queries.

1. SELECT * FROM R
2. SELECT * FROM R WHERE a = 42
3. SELECT * FROM R WHERE c = 42
4. SELECT * FROM R WHERE a <= 25
5. SELECT * FROM R WHERE b <= 25
6. SELECT * FROM R WHERE c <= 25
7. SELECT * FROM R WHERE a <= 25 AND b <= 25
8. SELECT * FROM R WHERE a <= 25 AND c <= 25
9. SELECT * FROM R WHERE a <= 25 AND a > 10
10. SELECT * FROM R WHERE a <= 25 OR b <= 25
11. SELECT * FROM R WHERE a = c
12. SELECT * FROM R, S WHERE R.a = S.a

For the rest of this worksheet we will try to optimize the following query:

```
SELECT *
FROM R, S, T
WHERE R.b = S.b AND S.c = T.c
AND R.a <= 50
```

We have 3 relations: R(a,b), S(b,c), and T(c,d).

- R has 1,000 data pages and 10,000 records
- S has 2,000 data pages and 40,000 records
- T has 3,000 data pages and 30,000 records

2 Single Table Access Plans

Assume we have the following indexes:

- Alt 2 unclustered index on R.a with 50 leaf pages
- Alt 2 clustered index on R.b with 100 leaf pages
- Alt 2 clustered indexes on S.b, T.c, and T.d (leaf page counts aren't relevant)

Also assume that it takes 2 IOs to reach the level above a leaf node and that no index or data pages are ever cached. All indexes have keys in the range [1, 100] with 100 distinct values.

1. How many IOs does a full scan on R take?

2. How many IOs does an index scan on R.a take?

3. How many IOs does an index scan on R.b take?

4. How many pages from R will advance to the next stage for all of these access plans?

Now assume that the other potential single table access plans have the following IO costs:

- Full scan on S: 2000 IOs
- Index scan on S.b: 2500 IOs
- Full scan on T: 3000 IOs
- Index scan on T.c: 3500 IOs
- Index scan on T.d: 3500 IOs

5. Which single table access plans advance to the next stage?

3 Multi-Table Plans

1. Assume we have 52 buffer pages. Remember, when the query optimizer calculates the cost of joining 2 tables for a given query, it must take into account the single table access plan for each table respectively. Consider the following joins:

a. R $\textit{BNLJ}_{R.b=S.b}$ S
   i. Which single table access plans for R and S will minimize the cost of this join?

   ii. What is the I/O cost for performing this join?

b. R $\textit{SMJ}_{R.b=S.b}$ S
   i. Which single table access plans for R and S will minimize the cost of this join?

   ii. What is the I/O cost for performing this join?

Assume all of the joins our database could do are as follows:

1. R BNLJ S: 3.a
2. R SMJ S: 3.b
3. S BNLJ R: 18,000 IOs
4. S SMJ R: 3,000 IOs
5. R BNLJ T: 30,000 IOs
6. R SMJ T: 40,000 IOs
7. T BNLJ R: 35,000 IOs
8. T SMJ R: 20,000 IOs
9. S BNLJ T: 15,000 IOs
10. S SMJ T: 10,000 IOs
11. T BNLJ S: 25,000 IOs
12. T SMJ S: 30,000 IOs

2. Which of these joins will actually be considered by the query optimizer on pass 2?

3. Which of these joins will advance to the next pass of the query optimizer?

4. Will any of these joins produce an interesting order?

5. How could we modify the query so that the S SMJ R produces an interesting order?

6. Will the query plan: T BNLJ (S SMJ R) be considered by the final pass of the query optimizer?