RELATIONAL OPERATORS: EXERCISES

CS 564 - Spring 2025

HASH-BASED AGGREGATION

Businesses (BusinessID INTEGER, BName CHAR(30), City CHAR(20), State CHAR(2))

```
SELECT City, COUNT (BusinessID)
FROM Businesses
GROUP BY City;
```

What is the **maximum number of cities** for which it is possible to implement hash-based aggregation using a **one pass** algorithm?

- Suppose that there is no index on the Businesses relation
- The fudge factor of creating an in-memory hash table is *f* = 1.4
- A page is 8 kB (1kB = 1024B).
- Each integer is 8B and each character is 1B
- Buffer size B = 10,000

HASH-BASED AGGREGATION

- Each entry is = 20*1 B + 8 B = 28 B
- Size of hash table = 28 * C * f /(8 *1024) pages
- This must be <= B-1

SORT-MERGE JOIN

- We are given two relations: R with 30,000 pages and S with 10,000 pages.
- We are performing a key-foreign key join between R and S, where S has the foreign key attribute.
- Suppose that R is already sorted on the join attribute.
- Assume that the size of the buffer is B = 100 pages

What is the I/O cost of the Sort Merge Join algorithm that uses replacement sort to create the initial runs?

Do not count the cost of writing the join result to disk.

SORT-MERGE JOIN

- Phase 1: create initial runs for S
  - – # runs = 10,000/(2*B) = 50
  - – I/O cost = 2*10,000 = 20,000.
- Phase 2: (we have 51 runs)
  - – We can merge in one pass
  - – I/O cost = 30,000 + 10,000 = 40,000

- Total I/O cost = 60,000 I/Os

SORTING

- Sort relation R using the external sort algorithm.
- Assume that we use replacement sort during the initial pass, and we create sorted runs of size $2B$.
- the buffer pool has size $B = 11$.

Compute the **maximum size** of R (in pages) that can be sorted in **2 and 3** passes respectively.

- After the first pass, we have N/(2B) runs
- When do I need one more pass? N/(2B) <= B-1 <=> N <= 2B*(B-1).
- After the second pass. (N/(2B))/(B-1) <= B-1 <=> N <= 2B*(B-1)^2

HASH JOIN

- We are given two relations: *R* with 1,000 pages and *S* with 2,000 pages.
- We are performing a key-foreign key join of *R* and *S* wherein *S* has the foreign key attribute.

What is the **smallest size** *B* of the buffer pool for which the block nested loop join has smaller I/O cost than the hash join?

BNLJ cost = 1000+ 2000* k   Where k = [1000/(B-2)]

HJ cost = 3 *(1000 + 2000) = 9,000 but only when it runs in 2 passes

Solving for k, k <= 3. 1000/(B-2) >= 3, B 240 (B^2 > smallest relation)

QUERY OPTIMIZATION

```
SELECT COUNT (UserID)
FROM Users U, Reviews R
WHERE U.UserID = R.UserID AND R.Stars < 2 AND U.Age = 18;
```

- No indexes on any relation and no relation is sorted on any attribute.
- Assume that the values of Stars are real numbers uniformly distributed between 1 and 5 (inclusive), and the values of Age are integers uniformly distributed between 10 and 99 (inclusive).
- B = 10,000
- Users has 75,000 pages, Reviews has 500,000 pages

Propose a physical plan for the following SQL query that achieves the **smallest possible I/O cost**.

QUERY OPTIMIZATION

SELECT COUNT (UserID)

FROM Users U, Reviews R

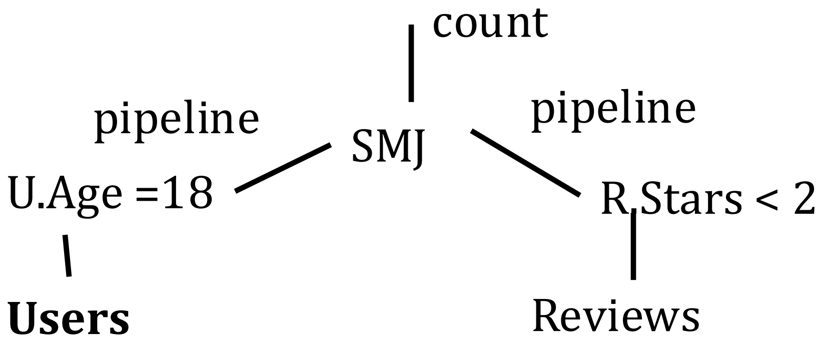
WHERE U.UserID = R.UserID AND R.Stars < 2 AND U.Age = 18;

- Assume that the values of Stars are real numbers uniformly distributed between 1 and 5 (inclusive), and the values of Age are integers uniformly distributed between 10 and 99 (inclusive).
- B = 10,000, Users has 75,000 pages, Reviews has 500,000 pages

Selection (R.Stars < 2): selectivity = 0.25 : output size 500,000/4 =125,000

Selection (U.Age=18): selectivity =1/90 : output size = 75,000/90 ~ 830 pages

75,000+
500,000 +
2 * (830 + 125,000)+
0



count

pipeline

SMJ

pipeline

U.Age =18

R.Stars < 2

Users

Reviews