COL362/632 Introduction to Database Management Systems

Plan Cost Estimation / Cardinality Estimation

Kaustubh Beedkar

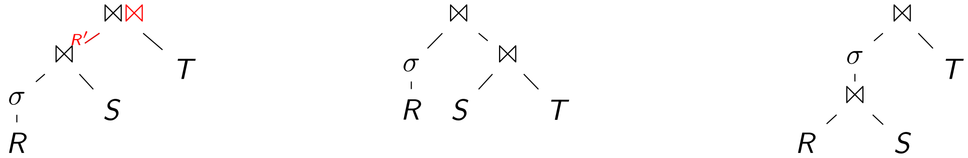
Department of Computer Science and Engineering
Indian Institute of Technology Delhi

So far ...

- Declarative queries are parsed into expression trees
- Physical operator implementations for different operators
  - Algorithms
  - Cost analysis
- Lets now look at **cost estimation**

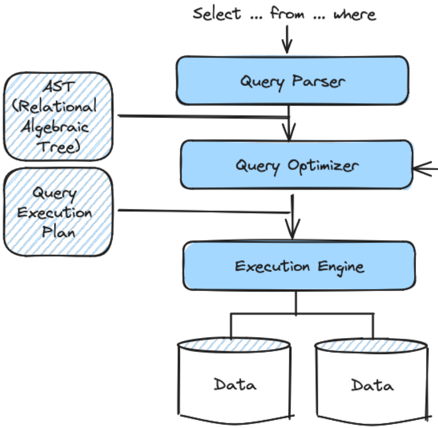
Cost Estimation Problem

- The goal of the query optimizer is to find a physical plan that has least cost
- For this, we need to **estimate** the total cost
  - Estimate cost of each operation in plan tree
  - We've already discussed these for various operators
    - Depends on input cardinalities
- Consider three (equivalent) plans //we'll discuss equivalence later



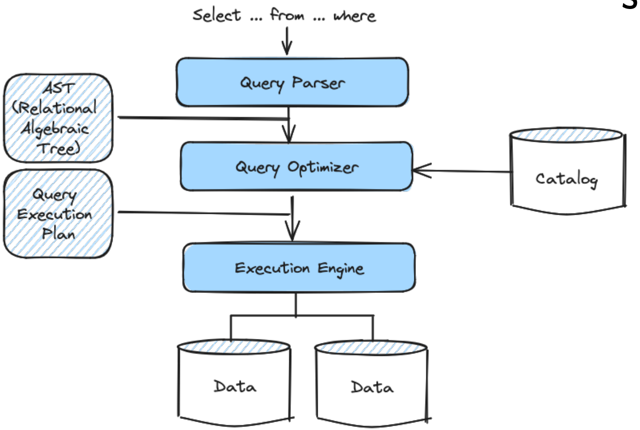
- We know how to compute the cost if we know the cardinalities
  - $\text{cost}(R' \bowtie T) = 3(b_{R'} + b_T)$ //e.g., hash join
  - $b_{R'} = \frac{n_{R'}}{b}$ // b is block size
  - $n_{R'} = n_{\sigma(R)\bowtie S}$
- Cardinality estimation problem: e.g., **estimate** $n_{\sigma(R)\bowtie S}$

Catalogue



- **Collect** statistical summaries of stored data
- **Estimate size** (cardinalities) in a bottom-up fashion
  - This is the most difficult part!
- **Estimate cost** using the estimated size
  - Formulas, heuristics (recall operator implementation)

Catalogue



Statics are collected periodically, using sampling

Statistics stored in Catalogue

- n_R //no. of tuples in R
- b_R //no. of pages
- ℓ_R //size of $r \in R$
- $V(R, A)$ // no. of distinct values
- $\min(R, A)$, $\max(R, A)$
- Indexes and metadata (e.g., height, keys)
- Histograms, Sketches,...

Cardinality Estimation Problem

- Consider a SPJ query Q

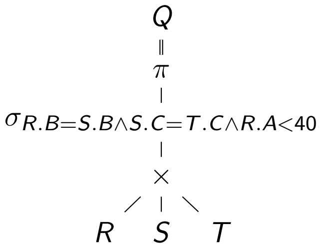
  $Q \equiv$ `select ... from R_1,...,R_n where cond_1 and ... cond_k`

- Given $n_{R_1}, n_{R_2}, \ldots, n_{R_n}$
- Estimate n_Q
- Note: this does not have to be exact! A **good approximate** is good enough
- Observe that $n_Q \leq n_{R_1} \times n_{R_2} \times \cdots \times n_{R_n}$
- **Key idea:** Each condition reduces the size n_Q by selectivity factor

Example

- Consider the following query Q on relations $R(A, B)$, $S(B, C)$, and $T(C, D)$

  and $n_R = 30K$, $n_S = 200K$, and $n_T = 10K$



- Let us assume that selectivity of
  - $R.B = S.B$ is $1/3$
  - $S.C = T.C$ is $1/10$
  - $R.A < 40$ is $1/2$
- **Estimate** the size n_Q of the query Q
- $n_Q = 30K \times 200K \times 10K \times \frac{1}{3} \times \frac{1}{10} \times \frac{1}{2} = 10^{12}$
- Note This assumes that attributes are independent! //more on this later

Selectivity Estimation

- $A = c$ $\quad // \ \sigma_{A=c}(R)$
  – selectivity $= \frac{1}{V(R,A)}$

- $A < c$ $\quad // \sigma_{A<c}(R)$
  – selectivity $= \frac{c - min(R,A)}{max(R,A) - min(R,A)}$

- $A = B$ $\quad // R \bowtie_{R.A=S.B} S$
  – selectivity $= \frac{1}{max(V(R,A), V(S,B))}$

Assumptions

- **Containment of values:** if $V(R, A) \leq V(S, B)$, then all values of $R.A$ occur in $S.B$
  - Note: this always holds when there is a PK-FK relationship

- **Preservation of values:** for any other attribute C, we have $V(R \bowtie_{A=B} S, C) = V(R, C)$ or $V(S, C)$
  - Note: We don't need this to estimate the size of the join, but we need it in estimating the next operator

Join Size Estimation

- Consider relations $R(A, ...)$ and $S(B, ...)$ and $Q \equiv R \bowtie_{R.A=S.B} S$
- Assume $V(R, A) \leq V(S, B)$
  - A tuple $r \in R$ joins with $n_S/V(S, B)$ tuple(s) in S
  - Hence, $n_Q = \frac{n_R * n_S}{V(S,B)}$
- size of join $n_Q = \frac{n_R * n_S}{max(V(R,A),V(S,B))}$

Example

- $n_R = 10K$, $n_S = 20K$
- $V(R, A) = 100$, $V(S, A) = 200$
- Q: how large is $R \bowtie_{A=B} S$?

Computing Plan Cost

- Estimate cardinality in a bottom-up fashion
  - Cardinality is the size n_R of a relation R
  - Compute size of all intermediate relations in a plan
- Estimate cost by using the estimated cardinalities

Computing Plan cost (Example)

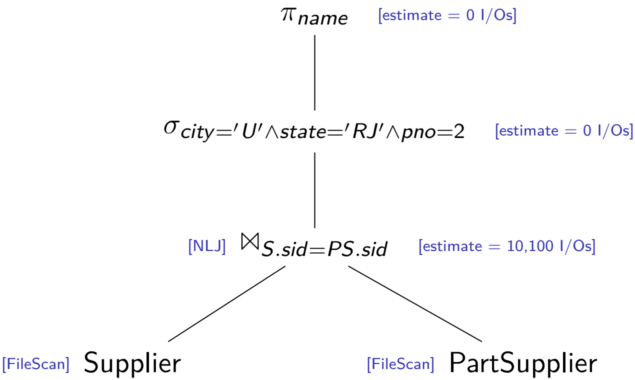
Supplier(sid, name, city, state)
PartSupplier(sid,pno,quantity)

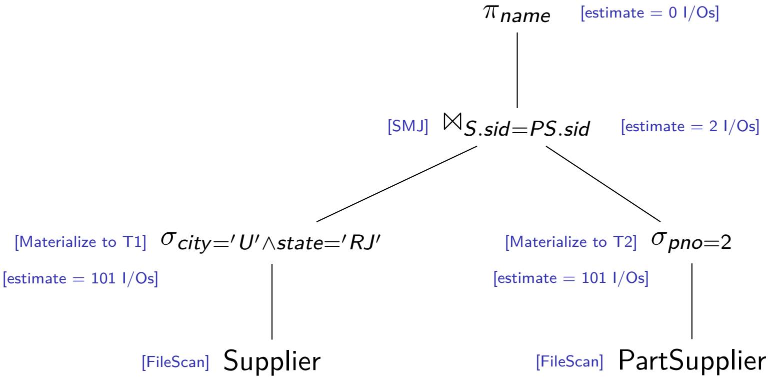
```
select name from Supplier S, PartSupplier PS where S.sid = PS.sid and
PS.pno=2 and S.city = 'Udaipur' and S.state='RJ'
```

Catalogue

- $n_S = 1000$, $n_{PS} = 10,000$
- $b_S = 100$, $b_{PS} = 100$
- $V(S, city) = 20$, $V(S, state) = 10$, $V(PS, pno) = 2500$
- $B = 11$

Physical Query Plan 1



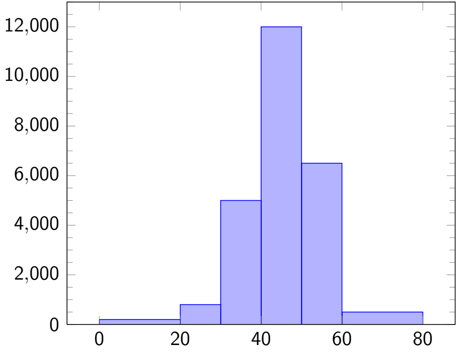

π_{name}
[estimate = 0 I/Os]
[SMJ]
$\bowtie_{S.sid=PS.sid}$
[estimate = 2 I/Os]
[Materialize to T1]
$\sigma_{city='U' \wedge state='RJ'}$
[estimate = 101 I/Os]
[Materialize to T2]
$\sigma_{pno=2}$
[estimate = 101 I/Os]
[FileScan]
Supplier
[FileScan]
PartSupplier

Histograms

A histogram is an approximate data synopsis that discretizes a value collection into different bins and stores the frequency of the values that fall into each bin

- Maintain an occurrence of count per value (or range of values)
- Makes cardinality estimation more accurate
  (hence cost estimations are more accurate)

Histograms



age	0–19	20–29	30–39	40–49	50–59	60–80
#Tuples	200	800	5000	12000	6500	500

- Employee(id, name, age)
- $n_E = 25000,\ V(E, age) = 50$
  $min(age) = 19,\ max(age) = 68$

- **Estimate** size of $\sigma_{age=48}(E)$
  – 1200

- **Estimate** size of $\sigma_{age>28 \wedge age<35}(E)$
  – $1 \times 80 + 5 \times 500 = 2580$

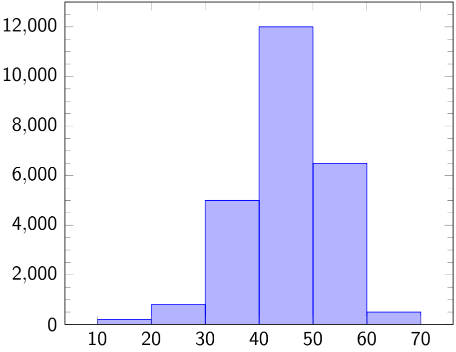
Histogram Types

Bucket boundaries have huge impact of estimations. How to determine bucket boundaries?

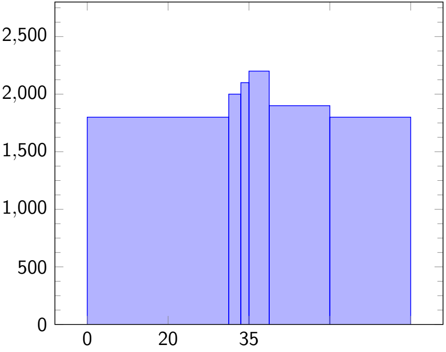
Types of histograms

- Equi-width
- Equi-depth
- Compressed
- V-Optimal

Equi-width vs Equi-depth Histograms



- all buckets have same width



- vary the width of buckets so that total number of tuples for each bucket is roughly the same
- “frequent zones” get their own bucket

V-optimal Histograms

- Define bucket boundaries in an optimal way, to minimize the error of overall point queries
- Computed rather expensively, using dynamic programming
- Modern database systems use V-optimal or some variations

Summary

- Compute selectivity for basic conditions
  - Formula-based
  - Histogram based (some databases also use sketches)
- Assumption – Uniform distribution
- Assumption – Independent conditions (attributes)
  - selectivity of AND = prod. of selectivity
  - selectivity of OR = sum of selectivity - prod. of selectivity
  - selectivity of NOT = 1 - selectivity

Quiz (Ungraded)

CarDekho Example

- Consider a relation `cars (make, model, year, price)`

Q Estimate the cardinality of

```
SELECT * FROM Cars WHERE make = 'Honda' AND model = 'Civic';
```

- Catalog
  - $n_R = 100,000$
  - Frequency of Honda is 10%
  - Frequency of Civic is 5%
- If we assume independence, cardinality $= 100,000 \times \frac{1}{10} \times \frac{1}{20} = 500$
- But this is wrong!
- In reality, **only Honda makes Civic**
  - So, every row where `Model = 'Civic'` must also have `Make = 'Honda'`
  - Correct estimated cardinality $= 100,000 \times \frac{1}{20} = 5000$

Key Takeways

- Independency assumption fails because `Make` and `Model` are strongly correlated
- Assuming independence underestimates selectivities
- Advanced selectivity estimation
  - Multidimentional histograms
  - Bayesian Networks
  - Machine Leaning
- This beyond COL362, check out Advance Data Management or Spl. Topics in Data(base) Systems