Cost-Based Optimization

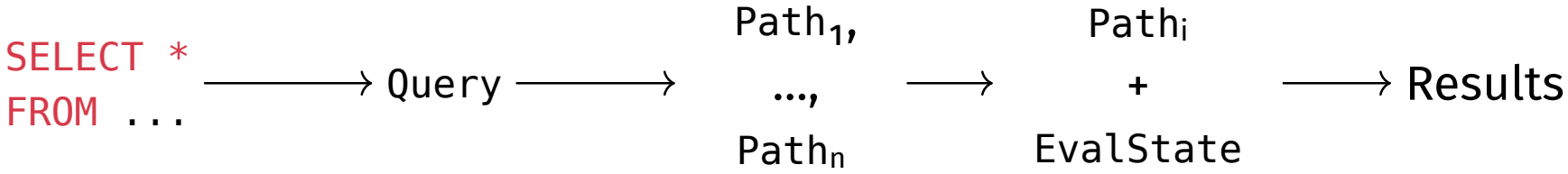
CSE 4/562: Database Systems | Lecture 11

DB. Sys.: T.C.B.: Ch. 16

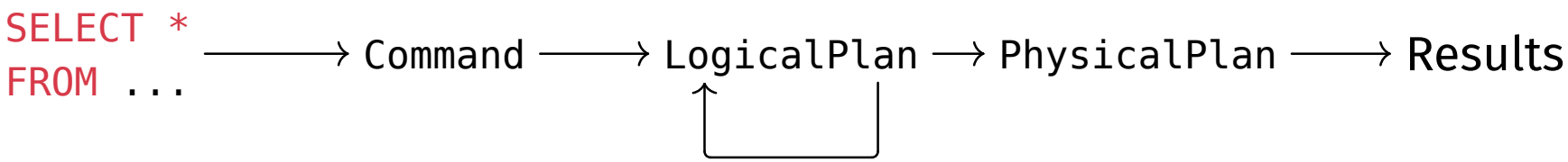
Quiz

Existing Systems

Postgresql



Apache Spark



Cost Estimation

Input

- $\mathrm{Path}_1, \ldots, \mathrm{Path}_n$
- Previously collected statistics

Output

- $\mathrm{argmin}_i \ \mathrm{Cost}(\mathrm{Path}_i)$

Cost Estimation

We need a way to estimate $\mathrm{Cost}(\mathrm{Path}_i)$

Idea 1: `Eval(`Plan_i`)`

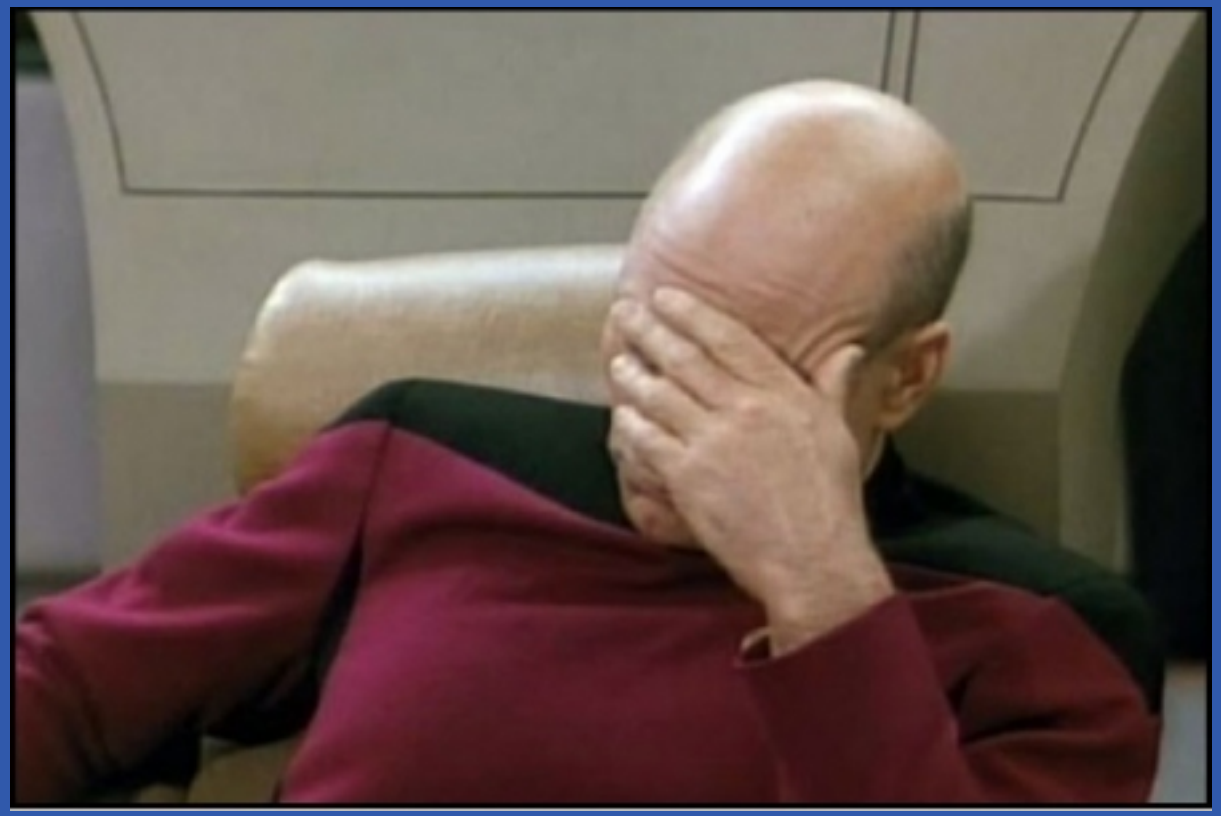


Image credit: © Paramount Pictures

Ok, no 🤦 this time

If we can’t get the exact cost of a plan, what can we do?

Ideas 2 and 3

Idea 2: Run each plan on a small sample of the data.

Idea 3: Analytically estimate the cost of a plan.

Ideas 2 and 3

Idea 2: Run each plan on a small sample of the data.

Idea 3: Analytically estimate the cost of a plan.

Plan Cost

CPU Time

- How much time is spent processing data?

IO Cost

- How many pages are read/written?
- How much spatial locality is the workload?

Memory Cost

- What is the smallest amount of memory required (working set size)

Plan Cost

CPU Time

- How much time is spent processing data?

IO Cost

- How many pages are read/written?
- How much spatial locality is the workload?

Memory Cost

- What is the smallest amount of memory required (working set size)

Image credit: Randal Munroe (cc-by-nc)

Remember the real goals

The real goals

1. Accurately **rank** the plans.
2. Don’t spend more time optimizing than you get back.
3. Don’t pick a plan that uses more memory than you have.

Resource Consumption

- $|R|$: The number of data pages in R
- B: Pages of buffer
- K: Keys per index page

Table Scan

Table Scan

$\mathrm{IO}(R) = |R|$

Table Scan

$$\mathrm{IO}(R) = |R|$$

$$\mathrm{Mem}(R) = O(1)$$

Projection

Projection

$\mathrm{IO}(\pi R) = 0 \ (+ \ \mathrm{IO}(R))$

Projection

$\mathrm{IO}(\pi R) = 0 \ (+ \ \mathrm{IO}(R))$

$\mathrm{Mem}(\pi R) = O(1)$

Selection

Selection

$\mathrm{IO}(\sigma R) = 0$ $(+ \mathrm{IO}(R))$

Selection

$\mathrm{IO}(\sigma R) = 0 \ (+ \ \mathrm{IO}(R))$

$\mathrm{Mem}(\sigma R) = O(1)$

Union

Union

$\mathrm{IO}(R \cup S) = 0 \; (+ \; \mathrm{IO}(R) + \mathrm{IO}(S))$

Union

$\mathrm{IO}(R \cup S) = 0 \; (+ \; \mathrm{IO}(R) + \mathrm{IO}(S))$

$\mathrm{Mem}(R \cup S) = O(1)$

Sort

Sort

In Memory

Sort

In Memory

$\mathrm{IO}(\tau R) = 0 \ (+ \ \mathrm{IO}(R))$

Sort

In Memory

$\mathrm{IO}(\tau R) = 0 \ (+ \ \mathrm{IO}(R))$

$\mathrm{Mem}(\tau R) = O(|R|)$

Sort

In Memory

$\mathrm{IO}(\tau R) = 0 \ (+ \ \mathrm{IO}(R))$

$\mathrm{Mem}(\tau R) = O(|R|)$

On Disk

Sort

In Memory

$\mathrm{IO}(\tau R) = 0 \ (+ \ \mathrm{IO}(R))$

$\mathrm{Mem}(\tau R) = O(|R|)$

On Disk

$\mathrm{IO}(\tau R) = 2 \cdot \lfloor \log_B(|R|) \rfloor \ (+ \ \mathrm{IO}(R))$

Sort

In Memory

$\mathrm{IO}(\tau R) = 0 \ (+ \ \mathrm{IO}(R))$

$\mathrm{Mem}(\tau R) = O(|R|)$

On Disk

$\mathrm{IO}(\tau R) = 2 \cdot \lfloor \log_B(|R|) \rfloor \ (+ \ \mathrm{IO}(R))$

$\mathrm{Mem}(\tau R) = O(B)$

Index Scan;

B+ Tree

Index Scan;

B+ Tree

$$\mathrm{IO}(\mathrm{IndexScan}(R, \theta)) = \log_K(|R|) + |\sigma_c(R)|$$

Index Scan;

B+ Tree

$$\mathrm{IO}(\mathrm{IndexScan}(R, \theta)) = \log_K(|R|) + |\sigma_c(R)|$$

$$\mathrm{Mem}(\mathrm{IndexScan}(R, \theta)) = O(1)$$

Index Scan;

B+ Tree

$$\mathrm{IO}(\mathrm{IndexScan}(R, \theta)) = \log_K(|R|) + |\sigma_c(R)|$$

$$\mathrm{Mem}(\mathrm{IndexScan}(R, \theta)) = O(1)$$

Hash

Index Scan;

B+ Tree

$$\mathrm{IO}(\mathrm{IndexScan}(R, \theta)) = \log_K(|R|) + |\sigma_c(R)|$$

$$\mathrm{Mem}(\mathrm{IndexScan}(R, \theta)) = O(1)$$

Hash

$$\mathrm{IO}(\mathrm{IndexScan}(R, \theta)) = 1$$

Index Scan;

B+ Tree

$$\mathrm{IO}(\mathrm{IndexScan}(R, \theta)) = \log_K(|R|) + |\sigma_c(R)|$$

$$\mathrm{Mem}(\mathrm{IndexScan}(R, \theta)) = O(1)$$

Hash

$$\mathrm{IO}(\mathrm{IndexScan}(R, \theta)) = 1$$

$$\mathrm{Mem}(\mathrm{IndexScan}(R, \theta)) = O(1)$$

Block Nested Loop Join

Buffer S in Memory

Block Nested Loop Join

Buffer S in Memory

$\mathrm{IO}(R \times S) = 0 \; (+ \; \mathrm{IO}(R) + \mathrm{IO}(S))$

Block Nested Loop Join

Buffer S in Memory

$\mathrm{IO}(R \times S) = 0 \; (+ \; \mathrm{IO}(R) + \mathrm{IO}(S))$

$\mathrm{Mem}(R \times S) = O(|S|)$

Block Nested Loop Join

Buffer S in Memory

$\mathrm{IO}(R \times S) = 0 \; (+ \; \mathrm{IO}(R) + \mathrm{IO}(S))$

$\mathrm{Mem}(R \times S) = O(|S|)$

Buffer S on Disk

Block Nested Loop Join

Buffer S in Memory

$$\mathrm{IO}(R \times S) = 0 \; (+ \; \mathrm{IO}(R) + \mathrm{IO}(S))$$

$$\mathrm{Mem}(R \times S) = O(|S|)$$

Buffer S on Disk

$$\mathrm{IO}(R \times S) = \left(1 + \frac{|R|}{B}\right) \cdot |S| \; (+ \; \mathrm{IO}(R) + \mathrm{IO}(S))$$

Block Nested Loop Join

Buffer S in Memory

$\mathrm{IO}(R \times S) = 0 \; (+ \; \mathrm{IO}(R) + \mathrm{IO}(S))$

$\mathrm{Mem}(R \times S) = O(|S|)$

Buffer S on Disk

$\mathrm{IO}(R \times S) = \left(1 + \frac{|R|}{B}\right) \cdot |S| \; (+ \; \mathrm{IO}(R) + \mathrm{IO}(S))$

$\mathrm{Mem}(R \times S) = O(B)$

Block Nested Loop Join

Buffer S in Memory

$\mathrm{IO}(R \times S) = 0 \ (+ \ \mathrm{IO}(R) + \mathrm{IO}(S))$

$\mathrm{Mem}(R \times S) = O(|S|)$

Buffer S on Disk

$\mathrm{IO}(R \times S) = \left(1 + \frac{|R|}{B}\right) \cdot |S| \ (+ \ \mathrm{IO}(R) + \mathrm{IO}(S))$

$\mathrm{Mem}(R \times S) = O(B)$

Rewind S

Block Nested Loop Join

Buffer S in Memory

$$\mathrm{IO}(R \times S) = 0 \ (+ \ \mathrm{IO}(R) + \mathrm{IO}(S))$$

$$\mathrm{Mem}(R \times S) = O(|S|)$$

Buffer S on Disk

$$\mathrm{IO}(R \times S) = \left(1 + \frac{|R|}{B}\right) \cdot |S| \ (+ \ \mathrm{IO}(R) + \mathrm{IO}(S))$$

$$\mathrm{Mem}(R \times S) = O(B)$$

Rewind S

$$\mathrm{IO}(R \times S) = \frac{|R|}{B} \cdot \mathrm{IO}(S) \ (+ \ \mathrm{IO}(R))$$

Block Nested Loop Join

Buffer S in Memory

$$\mathrm{IO}(R \times S) = 0 \ (+ \ \mathrm{IO}(R) + \mathrm{IO}(S))$$

$$\mathrm{Mem}(R \times S) = O(|S|)$$

Buffer S on Disk

$$\mathrm{IO}(R \times S) = \left(1 + \frac{|R|}{B}\right) \cdot |S| \ (+ \ \mathrm{IO}(R) + \mathrm{IO}(S))$$

$$\mathrm{Mem}(R \times S) = O(B)$$

Rewind S

$$\mathrm{IO}(R \times S) = \frac{|R|}{B} \cdot \mathrm{IO}(S) \ (+ \ \mathrm{IO}(R))$$

$$\mathrm{Mem}(R \times S) = O(B)$$

Hash Join

1-Pass (In-Memory)

Hash Join

1-Pass (In-Memory)

$\mathrm{IO}(R \bowtie S) = 0 \; (+ \; \mathrm{IO}(R) + \mathrm{IO}(S))$

Hash Join

1-Pass (In-Memory)

$\mathrm{IO}(R \bowtie S) = 0 \; (+ \; \mathrm{IO}(R) + \mathrm{IO}(S))$

$\mathrm{Mem}(R \bowtie S) = O(|S|)$

Hash Join

1-Pass (In-Memory)

$\mathrm{IO}(R \bowtie S) = 0 \; (+ \; \mathrm{IO}(R) + \mathrm{IO}(S))$

$\mathrm{Mem}(R \bowtie S) = O(|S|)$

2-Pass (On Disk)

Hash Join

1-Pass (In-Memory)

$$\mathrm{IO}(R \bowtie S) = 0 \ (+ \ \mathrm{IO}(R) + \mathrm{IO}(S))$$

$$\mathrm{Mem}(R \bowtie S) = O(|S|)$$

2-Pass (On Disk)

$$\mathrm{IO}(R \bowtie S) = 2|R| + 2|S| \ (+ \ \mathrm{IO}(R) + \mathrm{IO}(S))$$

Hash Join

1-Pass (In-Memory)

$\mathrm{IO}(R \bowtie S) = 0 \ (+ \ \mathrm{IO}(R) + \mathrm{IO}(S))$

$\mathrm{Mem}(R \bowtie S) = O(|S|)$

2-Pass (On Disk)

$\mathrm{IO}(R \bowtie S) = 2|R| + 2|S| \ (+ \ \mathrm{IO}(R) + \mathrm{IO}(S))$

$\mathrm{Mem}(R \bowtie S) = O\left(\frac{|R| + |S|}{\#\mathrm{Buckets}}\right)$

Sort Merge Join

Sort Merge Join

$\mathrm{IO}(R \bowtie S) =$ [As Sort]

$\mathrm{Mem}(R \bowtie S) =$ [As Sort]

Aggregate

In Memory

Aggregate

In Memory

$\mathrm{IO}\left(\sum_A R\right) = 0 \; (+ \; \mathrm{IO}(R))$

Aggregate

In Memory

$\mathrm{IO}\left(\sum_A R\right) = 0 \; (+ \; \mathrm{IO}(R))$

$\mathrm{Mem}\left(\sum_A R\right) = O(\mathrm{domain}(A))$

Aggregate

In Memory

$\mathrm{IO}\left(\sum_A R\right) = 0 \; (+ \; \mathrm{IO}(R))$

$\mathrm{Mem}\left(\sum_A R\right) = O(\mathrm{domain}(A))$

Sort / Merge

Aggregate

In Memory

$\mathrm{IO}\left(\sum_A R\right) = 0 \ (+ \ \mathrm{IO}(R))$

$\mathrm{Mem}\left(\sum_A R\right) = O(\mathrm{domain}(A))$

Sort / Merge

$\mathrm{IO}\left(\sum_A R\right) =$ [As Sort]

$\mathrm{Mem}\left(\sum_A R\right) =$ [As Sort]

Resource Consumption

$|R|$: **Size of** R

B: **Pages of Buffer**

K: **Keys per Index Page**

$\mathrm{domain}(A)$: **Number of distinct values of** A

Resource Consumption

$|R|$**: Size of** R

- Precomputed if R is a table
- ??? otherwise

B**: Pages of Buffer**

K**: Keys per Index Page**

$\mathrm{domain}(A)$**: Number of distinct values of** A

Resource Consumption

$|R|$**: Size of** R

- Precomputed if R is a table
- ??? otherwise

B**: Pages of Buffer**

- Configurable Parameter

K**: Keys per Index Page**

$\mathrm{domain}(A)$**: Number of distinct values of** A

Resource Consumption

$|R|$**: Size of** R

- Precomputed if R is a table
- ??? otherwise

B**: Pages of Buffer**

- Configurable Parameter

K**: Keys per Index Page**

- Given Parameter

$\mathrm{domain}(A)$**: Number of distinct values of** A

Resource Consumption

$|R|$**: Size of** R

- Precomputed if R is a table
- ??? otherwise

B**: Pages of Buffer**

- Configurable Parameter

K**: Keys per Index Page**

- Given Parameter

$\mathrm{domain}(A)$**: Number of distinct values of** A

- Can be precomputed ($|\delta_{A(R)}|$)

Estimating Resources

Estimating IO requires being able to estimate $|Q|$ and $|\delta_A(Q)|$

Unlike estimating IOs, cardinality estimation doesn't care about the algorithm.

We can work with simple RA.

Cardinality Estimation

Operator	RA	Estimated Size
Table	R	
Projection	$\pi(Q)$	
Union	$Q_1 \cup Q_2$	
Cross Product	$Q_1 \times Q_2$	
Sort	$\tau(Q)$	
Limit	$L_N(Q)$	
Selection	$\sigma_\theta(Q)$	
Join	$Q_1 \underset{\theta}{\bowtie} Q_2$	
Distinct	$\delta_A(Q)$	
Aggregate	$\sum_{A,\ \mathrm{agg}}(Q)$	

Cardinality Estimation

Operator	RA	Estimated Size
Table	R	$\lvert R \rvert$
Projection	$\pi(Q)$	
Union	$Q_1 \cup Q_2$	
Cross Product	$Q_1 \times Q_2$	
Sort	$\tau(Q)$	
Limit	$L_N(Q)$	
Selection	$\sigma_\theta(Q)$	
Join	$Q_1 \underset{\theta}{\bowtie} Q_2$	
Distinct	$\delta_A(Q)$	
Aggregate	$\sum_{A,\ \text{agg}}(Q)$	

Cardinality Estimation

Operator	RA	Estimated Size
Table	R	$\lvert R \rvert$
Projection	$\pi(Q)$	$\lvert Q \rvert$
Union	$Q_1 \cup Q_2$	
Cross Product	$Q_1 \times Q_2$	
Sort	$\tau(Q)$	
Limit	$L_N(Q)$	
Selection	$\sigma_\theta(Q)$	
Join	$Q_1 \underset{\theta}{\bowtie} Q_2$	
Distinct	$\delta_A(Q)$	
Aggregate	$\sum_{A,\ \mathrm{agg}}(Q)$	

Cardinality Estimation

Operator	RA	Estimated Size
Table	R	$\lvert R \rvert$
Projection	$\pi(Q)$	$\lvert Q \rvert$
Union	$Q_1 \cup Q_2$	$\lvert Q_1 \rvert + \lvert Q_2 \rvert$
Cross Product	$Q_1 \times Q_2$	
Sort	$\tau(Q)$	
Limit	$L_N(Q)$	
Selection	$\sigma_\theta(Q)$	
Join	$Q_1 \underset{\theta}{\bowtie} Q_2$	
Distinct	$\delta_A(Q)$	
Aggregate	$\sum_{A,\ \mathrm{agg}}(Q)$	

Cardinality Estimation

Operator	RA	Estimated Size
Table	R	$\lvert R \rvert$
Projection	$\pi(Q)$	$\lvert Q \rvert$
Union	$Q_1 \cup Q_2$	$\lvert Q_1 \rvert + \lvert Q_2 \rvert$
Cross Product	$Q_1 \times Q_2$	$\lvert Q_1 \rvert \times \lvert Q_2 \rvert$
Sort	$\tau(Q)$	
Limit	$L_N(Q)$	
Selection	$\sigma_\theta(Q)$	
Join	$Q_1 \underset{\theta}{\bowtie} Q_2$	
Distinct	$\delta_A(Q)$	
Aggregate	$\sum_{A,\ \mathrm{agg}}(Q)$	

Operator	RA	Estimated Size
Table	R	$\lvert R \rvert$
Projection	$\pi(Q)$	$\lvert Q \rvert$
Union	$Q_1 \cup Q_2$	$\lvert Q_1 \rvert + \lvert Q_2 \rvert$
Cross Product	$Q_1 \times Q_2$	$\lvert Q_1 \rvert \times \lvert Q_2 \rvert$
Sort	$\tau(Q)$	$\lvert Q \rvert$
Limit	$L_N(Q)$	
Selection	$\sigma_\theta(Q)$	
Join	$Q_1 \underset{\theta}{\bowtie} Q_2$	
Distinct	$\delta_A(Q)$	
Aggregate	$\sum_{A,\ \mathrm{agg}}(Q)$	

Cardinality Estimation

Operator	RA	Estimated Size
Table	R	$\lvert R\rvert$
Projection	$\pi(Q)$	$\lvert Q\rvert$
Union	$Q_1 \cup Q_2$	$\lvert Q_1\rvert + \lvert Q_2\rvert$
Cross Product	$Q_1 \times Q_2$	$\lvert Q_1\rvert \times \lvert Q_2\rvert$
Sort	$\tau(Q)$	$\lvert Q\rvert$
Limit	$L_N(Q)$	$\min(N, \lvert Q\rvert)$
Selection	$\sigma_\theta(Q)$	
Join	$Q_1 \underset{\theta}{\bowtie} Q_2$	
Distinct	$\delta_A(Q)$	
Aggregate	$\sum_{A,\ \mathrm{agg}}(Q)$	

Cardinality Estimation

Operator	RA	Estimated Size
Table	R	$\lvert R\rvert$
Projection	$\pi(Q)$	$\lvert Q\rvert$
Union	$Q_1 \cup Q_2$	$\lvert Q_1\rvert + \lvert Q_2\rvert$
Cross Product	$Q_1 \times Q_2$	$\lvert Q_1\rvert \times \lvert Q_2\rvert$
Sort	$\tau(Q)$	$\lvert Q\rvert$
Limit	$L_N(Q)$	$\min(N, \lvert Q\rvert)$
Selection	$\sigma_\theta(Q)$	$\lvert Q\rvert \times \mathrm{SEL}(\theta, Q)$
Join	$Q_1 \underset{\theta}{\bowtie} Q_2$	
Distinct	$\delta_A(Q)$	
Aggregate	$\sum_{A,\ \mathrm{agg}}(Q)$	

Cardinality Estimation

Operator	RA	Estimated Size
Table	R	$\lvert R\rvert$
Projection	$\pi(Q)$	$\lvert Q\rvert$
Union	$Q_1 \cup Q_2$	$\lvert Q_1\rvert + \lvert Q_2\rvert$
Cross Product	$Q_1 \times Q_2$	$\lvert Q_1\rvert \times \lvert Q_2\rvert$
Sort	$\tau(Q)$	$\lvert Q\rvert$
Limit	$L_N(Q)$	$\min(N, \lvert Q\rvert)$
Selection	$\sigma_\theta(Q)$	$\lvert Q\rvert \times \mathrm{SEL}(\theta, Q)$
Join	$Q_1 \underset{\theta}{\bowtie} Q_2$	$(\lvert Q_1\rvert \times \lvert Q_2\rvert) \times \mathrm{SEL}(\theta, Q_1 \times Q_2)$
Distinct	$\delta_A(Q)$	
Aggregate	$\sum_{A,\ \mathrm{agg}}(Q)$	

Cardinality Estimation

Operator	RA	Estimated Size
Table	R	$\lvert R\rvert$
Projection	$\pi(Q)$	$\lvert Q\rvert$
Union	$Q_1 \cup Q_2$	$\lvert Q_1\rvert + \lvert Q_2\rvert$
Cross Product	$Q_1 \times Q_2$	$\lvert Q_1\rvert \times \lvert Q_2\rvert$
Sort	$\tau(Q)$	$\lvert Q\rvert$
Limit	$L_N(Q)$	$\min(N, \lvert Q\rvert)$
Selection	$\sigma_\theta(Q)$	$\lvert Q\rvert \times \mathrm{SEL}(\theta, Q)$
Join	$Q_1 \bowtie_\theta Q_2$	$(\lvert Q_1\rvert \times \lvert Q_2\rvert) \times \mathrm{SEL}(\theta, Q_1 \times Q_2)$
Distinct	$\delta_A(Q)$	$\mathrm{UNIQ}(A, Q)$
Aggregate	$\sum_{A,\ \mathrm{agg}}(Q)$	

Cardinality Estimation

Operator	RA	Estimated Size
Table	R	$\lvert R\rvert$
Projection	$\pi(Q)$	$\lvert Q\rvert$
Union	$Q_1 \cup Q_2$	$\lvert Q_1\rvert + \lvert Q_2\rvert$
Cross Product	$Q_1 \times Q_2$	$\lvert Q_1\rvert \times \lvert Q_2\rvert$
Sort	$\tau(Q)$	$\lvert Q\rvert$
Limit	$L_N(Q)$	$\min(N, \lvert Q\rvert)$
Selection	$\sigma_\theta(Q)$	$\lvert Q\rvert \times \mathrm{SEL}(\theta, Q)$
Join	$Q_1 \underset{\theta}{\bowtie} Q_2$	$(\lvert Q_1\rvert \times \lvert Q_2\rvert) \times \mathrm{SEL}(\theta, Q_1 \times Q_2)$
Distinct	$\delta_A(Q)$	$\mathrm{UNIQ}(A, Q)$
Aggregate	$\sum_{A,\ \mathrm{agg}}(Q)$	$\mathrm{UNIQ}(A, Q)$

Cardinality Estimation: The Hard Parts

1. $\mathrm{SEL}(\theta, Q)$: The selectivity of c on Q = $\frac{|\sigma_\theta(Q)|}{|Q|}$
2. $\mathrm{UNIQ}(A, Q)$: The number of distinct values of Q.A

Cardinality Estimation - Idea 1

Idea 1: Assume each selection filters down to 10% of the data.

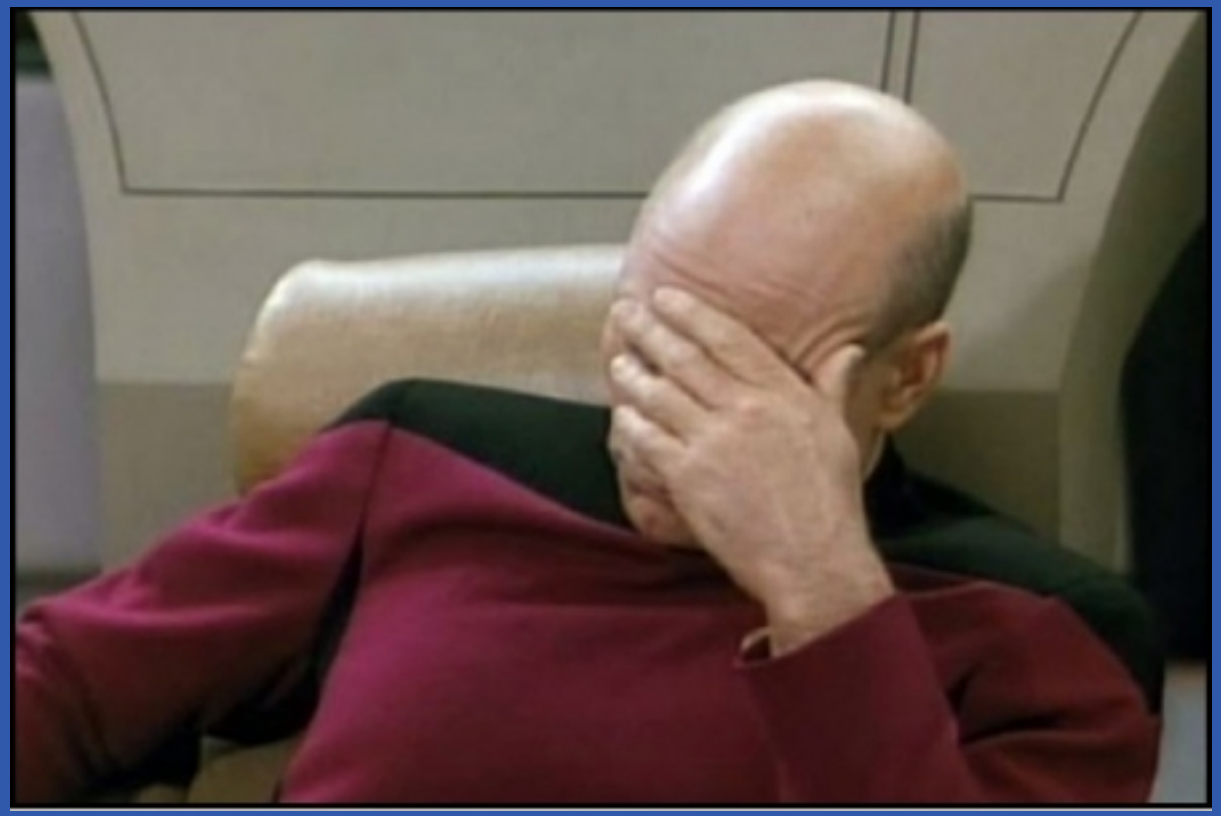


Image credit: © Paramount Pictures

Cardinality Estimation - Version 1

No... really.

(Postgres and many other databases do this)

Cardinality Estimation - Idea 1

There are Problems

Cardinality Estimation - Idea 1

There are Problems

- Inconsistent estimation: $|\sigma_{\theta_1}\left(\sigma_{\theta_2}(R)\right)| = |\sigma_{\theta_1 \wedge \theta_2}(R)|$

Cardinality Estimation - Idea 1

There are Problems

- Inconsistent estimation: $|\sigma_{\theta_1}\left(\sigma_{\theta_2}(R)\right)| = |\sigma_{\theta_1 \wedge \theta_2}(R)|$
- Too consistent estimation: $|\sigma_{\text{id}=1}(\text{STUDENTS})| = |\sigma_{\text{residence} = \text{NY}}(\text{STUDENTS})|$

Cardinality Estimation - Idea 1

There are Problems

- Inconsistent estimation: $|\sigma_{\theta_1}\left(\sigma_{\theta_2}(R)\right)| = |\sigma_{\theta_1 \wedge \theta_2}(R)|$
- Too consistent estimation: $|\sigma_{\text{id}=1}(\text{STUDENTS})| = |\sigma_{\text{residence} = \text{NY}}(\text{STUDENTS})|$

... remember that all we need is to **rank** plans.

Cardinality Estimation - Idea 1

Many major databases (Oracle, Postgres, Teradata, etc..) use something like the 10% rule if they have nothing better.

e.g. Teradata uses 10% for the first AND clause, 75% for every subsequetn clause.

Estimation Techniques

The 10% rule

- Rules of thumb if you have no other options

Uniform Prior

- Use basic statistics to make a rough guess

Sampling / History

- Small, quick sampling runs (or prior executions of the query)

Histograms

- More detailed statistics for improved guesses

Constraints

- Rules about the data for improved guesses

Uniform Prior

Assume that for $\sigma_\theta(Q)$ or $\delta_A(Q)$...

1. Basic statistics are known about Q:
   - `COUNT(*)`
   - `COUNT(DISTINCT A)` (for each `A`)
   - `MIN(A)`, `MAX(A)` (for each numeric `A`)
2. Attribute values are uniformly distributed.
3. No inter-attribute correlations.

Uniform Prior

- If necessary statistics aren't available, fall back to the 10% rule.
- If we make assumptions that are not perfectly correct, it will still likely be more accurate than the 10% rule.

COUNT(DISTINCT A)

COUNT(DISTINCT A)

$$\mathrm{UNIQ}\big(A, \pi_{A,\ldots}(Q)\big) = \mathrm{UNIQ}(A, Q)$$

COUNT(DISTINCT A)

$$\mathrm{UNIQ}\big(A, \pi_{A,\ldots}(Q)\big) = \mathrm{UNIQ}(A, Q)$$

$$\mathrm{UNIQ}(A, \sigma(Q)) = \mathrm{UNIQ}(A, Q)$$

COUNT(DISTINCT A)

$$\mathrm{UNIQ}\big(A, \pi_{A,\ldots}(Q)\big) = \mathrm{UNIQ}(A, Q)$$

$$\mathrm{UNIQ}(A, \sigma(Q)) = \mathrm{UNIQ}(A, Q)$$

$$\mathrm{UNIQ}(A, Q_1 \times Q_2) = \mathrm{UNIQ}(A, Q_1) \text{ or } \mathrm{UNIQ}(A, Q_2)$$

COUNT(DISTINCT A)

$$\mathrm{UNIQ}\big(A, \pi_{A,\ldots}(Q)\big) = \mathrm{UNIQ}(A, Q)$$

$$\mathrm{UNIQ}(A, \sigma(Q)) = \mathrm{UNIQ}(A, Q)$$

$$\mathrm{UNIQ}(A, Q_1 \times Q_2) = \mathrm{UNIQ}(A, Q_1) \text{ or } \mathrm{UNIQ}(A, Q_2)$$

$$\max(\mathrm{UNIQ}(A, Q_1), \mathrm{UNIQ}(A, Q_2)) \leq \mathrm{UNIQ}(A, Q_1 \cup Q_2) \leq \mathrm{UNIQ}(A, Q_1) + \mathrm{UNIQ}(A, Q_2)$$

MIN(A), MAX(A)

$$\min_A\left(\pi_{A,\ldots}(Q)\right) = \min_A(Q)$$

$$\min_A\left(\pi_{A,\ldots}(Q)\right) = \min_A(Q)$$

$$\min_A(\sigma(Q)) \approx \min_A(Q)$$

$$\min_A\left(\pi_{A,\ldots}(Q)\right) = \min_A(Q)$$

$$\min_A(\sigma(Q)) \approx \min_A(Q)$$

$$\min_A(Q_1 \times Q_2) = \min_A(Q_1) \text{ or } \min_A(Q_2)$$

$$\min_A\left(\pi_{A,\ldots}(Q)\right) = \min_A(Q)$$

$$\min_A(\sigma(Q)) \approx \min_A(Q)$$

$$\min_A(Q_1 \times Q_2) = \min_A(Q_1) \text{ or } \min_A(Q_2)$$

$$\min_A(Q_1 \cup Q_2) = \min(\min_A(Q_1) + \min_A(Q_2))$$

Estimating $\delta_A(Q)$ requires only `COUNT(DISTINCT A)`

Estimating Selectivity

Selectivity is a probability: $\mathrm{SEL}(\theta, Q) = P[\theta(r) \mid r \in Q]$

Estimating Selectivity

Selectivity is a probability: $\mathrm{SEL}(\theta, Q) = P[\theta(r) \mid r \in Q]$

$$P(A = c_1) = \frac{1}{\mathrm{COUNT(DISTINCT\ A)}}$$

Estimating Selectivity

Selectivity is a probability: $\mathrm{SEL}(\theta, Q) = P[\theta(r) \mid r \in Q]$

$$P(A = c_1) = \frac{1}{\mathrm{COUNT(DISTINCT\ A)}}$$

$$P(A \in (c_1, ..., c_n)) = \frac{n}{\mathrm{COUNT(DISTINCT\ A)}}$$

Selectivity is a probability: $\mathrm{SEL}(\theta, Q) = P[\theta(r) \mid r \in Q]$

$$P(A = c_1) = \frac{1}{\mathrm{COUNT(DISTINCT\ A)}}$$

$$P(A \in (c_1, ..., c_n)) = \frac{n}{\mathrm{COUNT(DISTINCT\ A)}}$$

$$P(A \leq c_1) = \frac{c_1 - \mathrm{MIN}(A)}{\mathrm{MAX}(A) - \mathrm{MIN}(A)}$$

Selectivity is a probability: $\mathrm{SEL}(\theta, Q) = P[\theta(r) \mid r \in Q]$

$$P(A = c_1) = \frac{1}{\mathrm{COUNT(DISTINCT\ A)}}$$

$$P(A \in (c_1, ..., c_n)) = \frac{n}{\mathrm{COUNT(DISTINCT\ A)}}$$

$$P(A \leq c_1) = \frac{c_1 - \mathrm{MIN}(A)}{\mathrm{MAX}(A) - \mathrm{MIN}(A)}$$

$$P(c_1 \leq A \leq c_1) = \frac{c_2 - c_1}{\mathrm{MAX}(A) - \mathrm{MIN}(A)}$$

Estimating Selectivity

$$P(A = B) = \min\left(\frac{1}{\text{COUNT(DISTINCT A)}}, \frac{1}{\text{COUNT(DISTINCT B)}}\right)$$

Estimating Selectivity

$$P(A = B) = \min\left(\frac{1}{\text{COUNT(DISTINCT A)}}, \frac{1}{\text{COUNT(DISTINCT B)}}\right)$$

$$P(\theta_1 \wedge \theta_2) = P(\theta_1) \times P(\theta_2)$$

$$P(A = B) = \min\left(\frac{1}{\text{COUNT(DISTINCT A)}}, \frac{1}{\text{COUNT(DISTINCT B)}}\right)$$

$$P(\theta_1 \wedge \theta_2) = P(\theta_1) \times P(\theta_2)$$

$$P(\theta_1 \vee \theta_2) = 1 - (1 - P(\theta_1)) \times (1 - P(\theta_2))$$

We don't always have statistics for Q

$\pi_{A=(B \times C)}(R)$

We don't always have clear rules for θ

$\sigma_{\text{FitsModel(A, B, C)}}(R)$

Attribute values are not always uniformly distributed

$\sigma_{\text{major} = \text{CSE}}(\text{STUDENTS})$

Attribute values are sometimes correlated

$\sigma_{\text{major} = \text{CSE} \wedge \text{school} = \text{SEAS}}(\text{STUDENTS})$

Reminders

- Quiz 3 results posted
- 1 week to Checkpoint 2 due date
  - Don’t forget to sign up for a code review