1 relational algebra:

Relational Algebra

Notation:

- $\mathcal{A}(e)$ attributes of the tuples produces by e
- $\mathcal{F}(e)$ free variables of the expression e
- binary operators $e_1 \theta e_2$ usually require $\mathcal{A}(e_1) = \mathcal{A}(e_2)$

$e_1 \cup e_2$	union, $\{x \mid x \in e_1 \vee x \in e_2\}$
$e_1 \cap e_2$	intersection, $\{x \mid x \in e_1 \wedge x \in e_2\}$
$e_1 \setminus e_2$	difference, $\{x \mid x \in e_1 \wedge x \notin e_2\}$
$\rho_{a \to b}(e)$	rename, $\{x \circ (b : x.a) \setminus (a : x.a) \mid x \in e\}$
$\Pi_A(e)$	projection, $\{\circ_{a \in A}(a : x.a) \mid x \in e\}$
$e_1 \times e_2$	product, $\{x \circ y \mid x \in e_1 \wedge y \in e_2\}$
$\sigma_p(e)$	selection, $\{x \mid x \in e \wedge p(x)\}$
$e_1 \bowtie_p e_2$	join, $\{x \circ y \mid x \in e_1 \wedge y \in e_2 \wedge p(x \circ y)\}$

per definition set oriented. Similar operators also used bag oriented (no implicit duplicate removal).

Additional (derived) operators are often useful:

$e_1 \bowtie e_2$	natural join, $\{x \circ y_{\mid \mathcal{A}(e_2) \setminus \mathcal{A}(e_1)} \mid x \in e_1 \wedge y \in e_2 \wedge x =_{\mid \mathcal{A}(e_1) \cap \mathcal{A}(e_2)} y\}$
$e_1 \div e_2$	division, $\{x_{\mid \mathcal{A}(e_1) \setminus \mathcal{A}(e_2)} \mid x \in e_1 \wedge \forall y \in e_2 \exists z \in e_1 : y =_{\mid \mathcal{A}(e_2)} z \wedge x =_{\mid \mathcal{A}(e_1) \setminus \mathcal{A}(e_2)} z\}$
$e_1 \ltimes_p e_2$	semi-join, $\{x \mid x \in e_1 \wedge \exists y \in e_2 : p(x \circ y)\}$
$e_1 \triangleright_p e_2$	anti-join, $\{x \mid x \in e_1 \wedge \nexists y \in e_2 : p(x \circ y)\}$
$e_1 ⟕_p e_2$	outer-join, $(e_1 \bowtie_p e_2) \cup \{x \circ \circ_{a \in \mathcal{A}(e_2)}(a : null) \mid x \in (e_1 \triangleright_p e_2)\}$
$e_1 ⟗_p e_2$	full outer-join, $(e_1 ⟕_p e_2) \cup (e_2 ⟕_p e_1)$

The algebra needs some extensions for real queries:

- map/function evaluation
  $\chi_{a:f}(e) = \{x \circ (a : f(x)) | x \in e\}$
- group by/aggregation
  $\Gamma_{A;a:f}(e) = \{x \circ (a : f(y)) | x \in \Pi_A(e) \wedge y = \{z | z \in e \wedge \forall a \in A : x.a = z.a\}\}$
- dependent join (djoin). Requires $\mathcal{F}(e_2) \subseteq \mathcal{A}(e_1)$
  $e_1 \ltimes_p e_2 = \{x \circ y | x \in e_1 \wedge y \in e_2(x) \wedge p(x \circ y)\}$

2 equivalence:

$$
\begin{aligned}
\sigma_{p_1 \wedge p_2}(e) &\equiv \sigma_{p_1}(\sigma_{p_2}(e)) \\
\sigma_{p_1}(\sigma_{p_2}(e)) &\equiv \sigma_{p_2}(\sigma_{p_1}(e)) \\
\Pi_{A_1}(\Pi_{A_2}(e)) &\equiv \Pi_{A_1}(e) \\
&\quad \text{if } A_1 \subseteq A_2 \\
\sigma_p(\Pi_A(e)) &\equiv \Pi_A(\sigma_p(e)) \\
&\quad \text{if } \mathcal{F}(p) \subseteq A \\
\sigma_p(e_1 \cup e_2) &\equiv \sigma_p(e_1) \cup \sigma_p(e_2) \\
\sigma_p(e_1 \cap e_2) &\equiv \sigma_p(e_1) \cap \sigma_p(e_2) \\
\sigma_p(e_1 \setminus e_2) &\equiv \sigma_p(e_1) \setminus \sigma_p(e_2) \\
\Pi_A(e_1 \cup e_2) &\equiv \Pi_A(e_1) \cup \Pi_A(e_2)
\end{aligned}
$$

$$
\begin{aligned}
e_1 \times e_2 &\equiv e_2 \times e_1 \\
e_1 \bowtie_p e_2 &\equiv e_2 \bowtie_p e_1 \\
(e_1 \times e_2) \times e_3 &\equiv e_1 \times (e_2 \times e_3) \\
(e_1 \bowtie_{p_1} e_2) \bowtie_{p_2} e_3 &\equiv e_1 \bowtie_{p_1} (e_2 \bowtie_{p_2} e_3) \\
\sigma_p(e_1 \times e_2) &\equiv e_1 \bowtie_p e_2 \\
\sigma_p(e_1 \times e_2) &\equiv \sigma_p(e_1) \times e_2 \\
&\quad \text{if } \mathcal{F}(p) \subseteq \mathcal{A}(e_1) \\
\sigma_{p_1}(e_1 \bowtie_{p_2} e_2) &\equiv \sigma_{p_1}(e_1) \bowtie_{p_2} e_2 \\
&\quad \text{if } \mathcal{F}(p_1) \subseteq \mathcal{A}(e_1) \\
\Pi_A(e_1 \times e_2) &\equiv \Pi_{A_1}(e_1) \times \Pi_{A_2}(e_2) \\
&\quad \text{if } A = A_1 \cup A_2, A_1 \subseteq \mathcal{A}(e_1), A_2 \subseteq \mathcal{A}(e_2)
\end{aligned}
$$

3 Join Ordering Basics

- Selectivity f_R of a selection $\sigma(R)$

$$f_R = \frac{|\sigma(R)|}{|R|}$$

- Selectivity $f_{1,2}$ of a join $R_1 \bowtie R_2$

$$f_{1,2} = \frac{|R_1 \bowtie R_2|}{|R_1 \times R_2|} = \frac{|R_1 \bowtie R_2|}{|R_1| \cdot |R_2|}$$

selectivity estimation of a predicate:

We know $|R_1|$, $|R_2|$, domains of $R_1.x$, $R_2.y$, (that is, $|R_1.x|$, $|R_2.y|$), and whether x and y are keys or not.
The selectivity of $\sigma_{R_1.x=c}$ is...

- if x is the key: $\frac{1}{|R_1|}$
- if x is not the key: $\frac{1}{|R_1.x|}$

selectivity estimation of joins:

We know $|R_1|$, $|R_2|$, $|R_1.x|$, $|R_2.y|$, and whether x and y are keys or not.
First, the size of $R_1 \times R_2$ is $|R_1||R_2|$
The selectivity of $\bowtie_{R_1.x=R_2.y}$ is...

- if both x and y are the keys: $\frac{1}{max(|R_1|,|R_2|)}$
- if only x is the key: $\frac{1}{|R_1|}$
- if both x and y are not the keys: $\frac{1}{max(|R_1.x|,|R_2.y|)}$

selectivity estimation of range predicates:

We know $|R_1|$, $max(R_1.x)$, $min(R_1.x)$, $R_1.x$ is arithmetic.

The selectivity of $\sigma_{R_1.x>c}$ is $\frac{max(R_1.x)-c}{max(R_1.x)-min(R_1.x)}$

The selectivity of $\sigma_{c_1<R_1.x<c_2}$ is $\frac{c_2-c_1}{max(R_1.x)-min(R_1.x)}$

output cardinality of a join tree T:

Given a join tree T, the result cardinality $|T|$ can be computed recursively as

$$|T| = \begin{cases} |R_i| & \text{if } T \text{ is a leaf } R_i \\ (\prod_{R_i \in T_1, R_j \in T_2} f_{i,j})|T_1||T_2| & \text{if } T = T_1 \bowtie T_2 \end{cases}$$

basic cost function:

- $C_{\text{out}}(T) = \begin{cases} 0 & \text{if } T \text{ is a leaf } R_i \\ |T| + C_{\text{out}}(T_1) + C_{\text{out}}(T_2) & \text{if } T = T_1 \bowtie T_2 \end{cases}$
- $C_{NL}(T_1 \bowtie T_2) = |T_1||T_2|$
- $C_{HJ}(T_1 \bowtie T_2) = 1.2|T_1|$
- $C_{SMJ}(T_1 \bowtie T_2) = |T_1|log(|T_1|) + |T_2|log(|T_2|)$

For sequences of join operators $s = s_1 \bowtie \ldots \bowtie s_n$:

$$\begin{aligned} C_{nlj}(s) &= \sum_{i=2}^{n} |s_1 \bowtie \ldots \bowtie s_{i-1}||s_i| \\ C_{hj}(s) &= \sum_{i=2}^{n} 1.2|s_1 \bowtie \ldots \bowtie s_{i-1}| \\ C_{smj}(s) &= \sum_{i=2}^{n} |s_1 \bowtie \ldots \bowtie s_{i-1}| \log(|s_1 \bowtie \ldots \bowtie s_{i-1}|) + \sum_{i=2}^{n} |s_i| \log(|s_i|) \end{aligned}$$

4 Search Space

Catalan Numbers: the number of binary trees with n leaf nodes is given by C(n-1).

$$C(n) = \frac{1}{n+1}\binom{2n}{n}$$

Number of Join Trees

	cliques/with cross products	chains, no cross products	stars, no cross products
left-deep	$n!$	2^{n-1}	$2 \cdot (n-1)!$
zigzag	$n! \cdot 2^{n-2}$	$2^{n-1} \cdot 2^{n-2}$	$2 \cdot (n-1)! \cdot 2^{n-2}$
bushy	$n!C(n-1) = \frac{(2n-2)!}{(n-1)!}$	$2^{n-1} \cdot C(n-1)$	$2 \cdot (n-1)! \cdot 2^{n-2}$

5 Join Algorithms

5.1 Problem Complexity according to query shape

query graph	join tree	cross products	cost function	complexity
general	left-deep	no	ASI	NP-hard
tree/star/chain	left-deep	no	ASI, 1 joint.	P
star	left-deep	no	NLJ+SMJ	NP-hard
general/tree/star	left-deep	yes	ASI	NP-hard
chain	left-deep	yes	-	open
general	bushy	no	ASI	NP-hard
tree	bushy	no	-	open
star	bushy	no	ASI	P
chain	bushy	no	any	P
general	bushy	yes	ASI	NP-hard
tree/star/chain	bushy	yes	ASI	NP-hard

5.2 IKKBZ: keep the n-s-C-T-rank table!!!

For every relation R_i we keep

- cardinality n_i
- selectivity s_i — the selectivity of the incoming edge from the parent of R_i
- cost $C(R_i) = n_i s_i$ (or 0, if R_i is the root)
- rank $r_i = \frac{T(r_i)-1}{C(r_i)} = \frac{n_i s_i - 1}{n_i s_i}$

Moreover,

- $C(S_1 S_2) = C(S_1) + T(S_1)C(S_2)$
- $T(S) = \prod_{R_i \in S}(s_i n_i)$
- rank of a sequence $r(S) = \frac{T(S)-1}{C(S)}$

5.3 MVP: Weighted Directed Join Graph

- physical edge: $w_{u,v} = \frac{|\bowtie_u|}{|u \sqcap v|}$
- virtual edge: $w_{u,v} = 1$
- node: $w(p_{i,j}, S) = \frac{|\bowtie^S_{p_{i,j}}|}{|R_i \bowtie_{p_{i,j}} R_j|}$

5.4 Non-inner Joins – Reordering Constraints, Compatibility Matrix

row: 1st join
column: 2nd join

	⋈	⟕	⟗	▷	⋉	▷Γ	...
⋈	+	+	-	+	+	+	...
⟕	-	+	-	-	-	-	...
⟗	-	+	+	-	-	-	...
▷	-	-	-	-	-	-	...
⋉	-	-	-	-	-	-	...
▷Γ	-	-	-	-	-	-	...
...							

some extra for right outer join:
$(A \bowtie B) ⟖ C \neq A \bowtie (B ⟖ C)$
$(A ⟖ B) \bowtie C \equiv A ⟖ (B \bowtie C)$

5.5 size of DP-Table for different query graph structures (for DPccp):

The approach aims to enumerate the number of connected subsets and csg-cmp-pairs, avoiding duplicates. There are some formulas which can be used wit n relations:

- Chain queries:
  - $\texttt{csg}(n) = \frac{n(n+1)}{2}$;
  - $\texttt{ccp}(n) = \frac{(n+1)^3-(n+1)^2+2(n+1)}{3}$;
- Cycle queries:
  - $\texttt{csg}(n) = n^2 - n + 1$;
  - $\texttt{ccp}(n) = n^3 - 2n^2 + n$;
- Star queries:
  - $\texttt{csg}(n) = 2^{n-1} + n - 1$;
  - $\texttt{ccp}(n) = (n-1)2^{n-2}$;
- Clique queries:
  - $\texttt{csg}(n) = 2^n - 1$;
  - $\texttt{ccp}(n) = 3^n - 2^{n+1} + 1$.

5.6 Graph Simplification: calculation of *orderingBenefit*($\bowtie_1, \bowtie_2$)

the benefit to put $\bowtie_2$ in front of $\bowtie_1$:

$$\text{orderingBenefit}(X \bowtie_1 R_1, X \bowtie_2 R_2) = \frac{C((X \bowtie_1 R_1) \bowtie_2 R_2)}{C((X \bowtie_2 R_2) \bowtie_1 R_1)}$$

Be careful in calculation of updating the orderingBenefit after the hyperedge is added. TRICKY to make mistakes!

5.7 Randomized Approaches: counting paths for the random bushy trees in a dyck word

The number of different paths from $(0,0)$ to (i,j) can be computed by

$$p(i,j) = \frac{j+1}{i+1} \binom{i+1}{\frac{1}{2}(i+j)+1}$$

These numbers are the *Ballot numbers*.
The number of paths from (i,j) to $(2n,0)$ can thus be computed as:

$$q(i,j) = p(2n-i,j)$$

Note the special case $q(0,0) = p(2n,0) = C(n)$.

The number of paths can also be computed intuitively from the end to start.

5.8 Order Preserving Joins: the equivalences

$$\begin{array}{rcll}
\sigma^L_{p_1}(\sigma^L_{p_2}(e)) & \equiv & \sigma^L_{p_2}(\sigma^L_{p_1}(e)) & \\
\sigma^L_{p_1}(e_1 \bowtie^L_{p_2} e_2) & \equiv & \sigma^L_{p_1}(e_1) \bowtie^L_{p_2} e_2) & \text{if } \mathcal{F}(p_1) \subseteq \mathcal{A}(e_1) \\
\sigma^L_{p_2}(e_1 \bowtie^L_{p_2} e_2) & \equiv & e_1 \bowtie^L_{p_2} \sigma^L_{p_1}(e_2) & \text{if } \mathcal{F}(p_1) \subseteq \mathcal{A}(e_2) \\
e_1 \bowtie^L_{p_1}(e_2 \bowtie^L_{p_2} e_3) & \equiv & (e_1 \bowtie^L_{p_1} e_2) \bowtie^L_{p_2} e_3) & \text{if } \mathcal{F}(p_i) \subseteq \mathcal{A}(e_i) \cup \mathcal{A}(e_{i+1})
\end{array}$$

6 Accessing the Data

6.1 seektime estimation

A good approximation of the **seek time** among d cylinders is:

$$seektime(d) = \begin{cases} c_1 + c_2\sqrt{d} & d \leq c_0 \\ c_3 + c_4 d & d > c_0 \end{cases}$$

6.2 Simplistic Cost Model

Model 2004		
Parameter	Value	Abbreviated Name
capacity	180 GB	D_{cap}
average latency time	5 ms	D_{lat}
sustained read rate	100 MB/s	D_{srr}
sustained write rate	100 MB/s	D_{swr}

The time a disk needs to read and transfer n bytes is then approximated by

$$T = D_{lat} + \frac{n}{D_{srr}}$$

D_{lat}: average latency time(seek + rotational delay)
D_{srr}: sustained read rate

Different access types:

- Sequential I/O: $T = 1 \cdot D_{lat} + \frac{n}{D_{srr}}$
- Random I/O: $T = (D_{lat} + \frac{|\text{page size}|}{D_{srr}}) \cdot \#\text{pages}$

6.3 counting the number of accesses

6.3.1 Parameters

N	$\|R\|$	number of tuples in the relation R
m	$\|\|R\|\|$	number of pages on which tuples of R are stored
B	N/m	number of tuples per page
k		number of (distinct) TIDs for which tuples have to be retrieved

6.3.2 Yao's formula (direct, uniform, distinct)

Considering m buckets with n items, then there is a total of $N = nm$ items. Randomly selecting k **distinct** items give a number of qualifying buckets which is:

$$\bar{y}_n^{N,m}(k) = m * y_n^N(k)$$

$$y_n^N(k) = \begin{cases} [1-p] & k \leq N-n \\ 1 & k > N-n \end{cases}$$

$y_n^N(k)$ is the probability that bucket n contains at least one tuple, and p is the probability that a bucket contains none of the k items.

$$p = \frac{\binom{N-n}{k}}{\binom{N}{k}} = \prod_{i=0}^{k-1} \frac{N-n-i}{N-i} = \prod_{i=0}^{n-1} \frac{N-k-i}{N-i}$$

6.3.3 Waters's approximation

$$p \approx \left(1 - \frac{k}{N}\right)^n$$

Then still needs to follow the rest of Yao's formula.

6.3.4 Bernstein's approximation

$$\bar{y}_n^{N,m}(k) \approx \begin{cases} k & k < \frac{m}{2} \\ \frac{k+m}{3} & \frac{m}{2} \leq k \leq 2m \\ m & 2m \leq k \end{cases}$$

6.3.5 Cheung's formula (direct, uniform, non-distinct)

The number of multiset with cardinality k containing only elements from a set S with $|S| = N$ is:

$$\binom{N+k-1}{k}$$

The number of qualifying buckets, if we randomly select k not necessarily distinct items:

$$\overline{Cheung}_n^{N,m}(k) = m * Cheung_n^N$$

$$Cheung_n^N(k) = [1-\tilde{p}]$$

$$\tilde{p} = \frac{\binom{N+k-1-n}{k}}{\binom{N+k-1}{k}} = \prod_{i=0}^{k-1} \frac{N-n+i}{N+i} = \prod_{i=0}^{n-1} \frac{N-1-i}{N-1+k-i}$$

6.3.6 Cardenas's approximation

$$\tilde{p} \approx \left(1 - \frac{n}{N}\right)^k$$

Then still needs to follow the rest of Cheung's formula.

6.3.7 Estimate of number of distinct values in a multiset, Corollary

The number of **distinct** values in a k-multiset of cardinality N with uniform distribution is:

$$D(n,k) = \frac{Nk}{N+k-1}$$

6.4 The cost of accessing these pages calculated by Yao/Cheung etc.

6.4.1 Bitvector Model

The total number of pages of a relation R is: bitvector length $\longrightarrow$ B
Pages to be read (computed by Yao/Cheung's formula): number of 1s $\longrightarrow$ b
Pages to be skipped: number of 0s

- Estimate the distribution of distance between two qualifying pages
- Bitvector B, b bits are set to 1
- First, the distribution of the number of j zeros
  - before first 1
  - between two consecutive 1s
  - after last 1
- Bitvectors having a 1 at position i followed by j zeros: $\binom{B-j-2}{b-2}$
- $B-j-1$ positions for i
- every bitvector has $b-1$ sequences of a form $10\ldots01$
- $\mathcal{B}_b^B(j) = \frac{(B-j-1)\binom{B-j-2}{b-2}}{(b-1)\binom{B}{b}} = \frac{\binom{B-j-1}{b-1}}{\binom{B}{b}}$
- now, the expected number of 0s between two 1s: $\bar{\mathcal{B}}_b^B = \frac{B-b}{b+1}$
- then, the expected number of bits from the start to the last 1: $\bar{\mathcal{B}}_{tot}(B,b) = \frac{Bb+b}{b+1}$
- finally, the expected number of bits between first and last 1: $\bar{\mathcal{B}}_{\text{1-span}}(B,b) = \frac{Bb-B+2b}{b+1}$

6.5 selectivity estimations

6.5.1 Heuristic Estimations

$|D(A)|$: the cardinality of A

Some commonly used selectivity estimations:

predicate	selectivity	requirement
$A = c$	$1/\lvert D(A)\rvert$	if index on A
	$1/10$	otherwise
$A > c$	$(\max(A) - c)/(\max(A) - \min(A))$	if index on A, interpol.
	$1/3$	otherwise
$A_1 = A_2$	$1/\max(\lvert D(A_1)\rvert, \lvert(D(A_2)\rvert)$	if index on A_1 and A_2
	$1/\lvert D(A_1)\rvert$	if index on A_1 only
	$1/\lvert D(A_2)\rvert$	if index on A_2 only
	$1/10$	otherwise

6.5.2 Histograms

Given a histogram, we can approximate the selectivities as follows:

$$A = c \qquad \frac{\sum_{b\in B:c\in b} H_A(b)}{\sum_{b\in B} H_A(b)}$$

$$A > c \qquad \frac{\sum_{b\in B:c\in b} \frac{\max(b)-c}{\max(b)-\min(b)} H_A(b)+\sum_{b\in B:\min(b)>c} H_A(b)}{\sum_{b\in B} H_A(b)}$$

$$A_1 = A_2 \qquad \frac{\sum_{b_1\in B_1, b_2\in B_2, b'=b_1\cap b_2 : b'\neq\emptyset} \frac{\max(b')-\min(b')}{\max(b_1)-\min(b_1)} H_{A_1}(b_1) \frac{\max(b')-\min(b')}{\max(b_2)-\min(b_2)} H_{A_2}(b_2)}{\sum_{b_1\in B_1} H_{A_1}(b_1) \sum_{b_2\in B_2} H_{A_2}(b_2)}$$