CS109A Notes for Lecture 3/17/95

Algebra of Relations

- Operand = relation (including attributes) or a variable representing a relation.
- Operators = union ($\cup$), difference ($-$), selection (σ), projection (π), product ($\times$).
  - ☐ Also, important operators intersection ($\cap$) and join ($\bowtie$) defined in terms of these.

Why Relational Algebra?

- Very expressive language with operators that "do a lot," e.g., $R \cup S$ implies a complex algorithm with lots of details we don't have to specify.
- Like all algebras, the algebraic laws let us "optimize" expressions into equivalent forms that are cheaper to evaluate.
  - ☐ For relations, where data is large and operators powerful, this ability makes orders of magnitude difference in running time.

Union, Intersection, Difference

- As for sets.
  - ☐ But schemes must agree (or rename the attributes).
  - ☐ Result has same scheme as operands.
- Note intersection in terms of difference: $R \cap S \equiv R - (R - S)$.

Selection

$\sigma_C(R)$ = relation of all tuples of R that satisfy condition C.

- ☐ C refers to attributes, representing components of the tuples.
- ☐ Result has same scheme as R.

Example: $\sigma_{\text{Weight} \geq 400000}(\text{Classes})$ = "find all those classes displacing at least 40,000 tons."

Class	Weight	Guns	Caliber	Type	Country
Hood	41000	8	15	BC	Gt. Br.
Iowa	46000	9	16	BB	USA
Yamato	65000	9	18	BB	Japan

Projection

$\pi_S(R)$ = take from each tuple of relation R those components for the attributes in list S.

- Scheme = attributes of S.

Example: $\pi_{\text{Guns, Caliber}}\big(\sigma_{\text{Country = "USA"}}(Classes)\big)$ = "List the number of guns and calibers of the US capital ships."

Guns	Caliber
9	12
8	16
9	16
10	14
12	14
12	12

Cartesian Product

$R \times S$ = take each tuple of R and pair it with each tuple of S.

- Scheme = attributes of R, then attributes of S.
  - ☐ If attribute A appears in both schemes, use $A.R$ and $A.S$ in result scheme.
- Not commonly used; generally appears in a "join" = product followed by selection.

Natural Join

$R \bowtie S$ =

1. Take $R \times S$.
2. Select for equality between each pair of attributes with the same name.
3. Project out one of each pair of equated attributes.

Example: *Ships* $\bowtie$ *Classes* extends the Ships tuples with all the information about its class.

- Example tuples:

Name	Lnchd	Class	Wt.	Guns	Cal.	Type	Cntry
Alabama	1942	S. Dakota	37000	9	16	BB	USA
Alaska	1944	Alaska	28000	9	12	BC	USA

Join Algorithms

- Very expensive operation — number of tuples can be product of number in the two operand relations.
  - □ If almost all tuples agree on the shared attributes.
- Methods: Index-join and Sort-Join.

Index-Join

Compute $R \bowtie S$ by:

```
for (r ∈ R) {
    find tuples s ∈ S matching
        r in shared attributes;
    produce tuple from r and s;
}
```

- Helps greatly if there is an index for S on one of the shared attributes.
- If no index, create temporary hash table (often called *hash-join*).
- Note that natural join is "sort of" commutative — the result scheme has a different order, but the information in $R \bowtie S$ is the same as in $S \bowtie R$.
  - □ Thus, an index for R on a shared attribute is as useful as one on S.
- With maximum-efficiency index, time on relations of size n is $O(n)$ plus big-oh of output size (possibly much larger than $O(n)$).

Sort-Join

- Sort R and S on their common attributes.

- Run through the sorted lists to group tuples from both relations that have the same values for shared attributes.
- Time is $O(n \log n)$ plus big-oh of output size.

Query Optimization

- Many algebraic equivalences.
- Major efficiency gains obtained by doing size-reducing operations (selection and projection) as early as possible.
- "Pushing selections down." $\sigma_C(R \bowtie S) \equiv \sigma_C(R) \bowtie S$, provided condition C refers only to attributes present in the scheme of S.
  - Similar push to S if attributes of C are there.
- "Splitting selections." $\sigma_{C \text{ and } D}(R) \equiv \sigma_C(\sigma_D(R))$.

Example: "What ships launched after 1940 had guns of less than 15-inch caliber?" In SQL, the compiler would interpret it as the algebraic expression

$$\pi_{\text{Name}}\big(\sigma_{\text{Launched}>1940 \text{ and } \text{Caliber}<15}(Ships \bowtie Classes)\big)$$

- Requires the join of two large relations.
- Split selection and push each part down the side where it makes sense:

$$\pi_{\text{Name}}\big(\sigma_{\text{Launched}>1940}(Ships) \bowtie \sigma_{\text{Caliber}<15}(Classes)\big)$$

- Selections produce subsets of each of the two large relations.
- Answer to either query = {Alaska, Anson, Duke of York, Guam, Howe, Prince of Wales}, a 1-ary relation with scheme Name.