Relational Algebra and Query Processing
One View of Basic Query Processing Steps

query -> parser and translator -> relational algebra expression

query output -> evaluation engine

evaluation engine -> execution plan

data

statistics about data
## Relational Algebra

<table>
<thead>
<tr>
<th>Relational Algebra Operation</th>
<th>Relational Algebra Operation Name</th>
<th>Equivalent SQL Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Pi_{A_1, A_2, \ldots, A_k}(r)$</td>
<td>Project</td>
<td>select $A_1, A_2, \ldots, A_k$ from $r$</td>
</tr>
<tr>
<td>$\sigma_p(r)$</td>
<td>Select</td>
<td>… from $r$ where $p$</td>
</tr>
<tr>
<td>$r \times s$</td>
<td>Cartesian-Product</td>
<td>… from $r, s$</td>
</tr>
<tr>
<td>$\sigma_{A=C}(r \times s)$</td>
<td>Equijoin</td>
<td>… from $r, s$ where $A = C$</td>
</tr>
<tr>
<td>$\sigma_{\theta}(r \times s)$</td>
<td>Theta Join</td>
<td>… from $r, s$ where $A \theta C$</td>
</tr>
<tr>
<td>$r \bowtie s$</td>
<td>Natural Join</td>
<td>… from $r$ natural join $s$</td>
</tr>
<tr>
<td>$\rho_X(E)$</td>
<td>Rename (Alias)</td>
<td>… from $E \times \ldots$</td>
</tr>
<tr>
<td>$\rho_X(A_1, A_2, A_n)(E)$</td>
<td>Rename</td>
<td>select $x.x A_1, x.y A_2, x.z A_n$ from $E \times$</td>
</tr>
<tr>
<td>table ←</td>
<td>Assignment</td>
<td>create table tmp as (select …)</td>
</tr>
</tbody>
</table>
Selection and Projection Rules

• Break complex selection into simpler ones:
  \[ \sigma_{\text{Cond}_1 \land \text{Cond}_2}(R) \equiv \sigma_{\text{Cond}_1}(\sigma_{\text{Cond}_2}(R)) \]

• Break projection into stages:
  \[ \pi_{\text{attr}}(R) \equiv \pi_{\text{attr}}(\pi_{\text{attr}'}(R)), \text{ if } \text{attr} \subseteq \text{attr}' \]

• Commute projection and selection:
  \[ \pi_{\text{attr}}(\sigma_{\text{Cond}}(R)) \equiv \sigma_{\text{Cond}}(\pi_{\text{attr}}(R)), \quad \text{if } \text{attr} \subseteq \text{all attributes in Cond} \]
Commutativity and Associativity of Join

• Join commutativity: \( R \bowtie S \equiv S \bowtie R \)
  – used to reduce cost of nested loop evaluation strategies (smaller relation should be in outer loop)

• Join associativity: \( R \bowtie (S \bowtie T) \equiv (R \bowtie S) \bowtie T \)
  – used to reduce the size of intermediate relations in computation of multi-relational join – first compute the join that yields smaller intermediate result

• N-way join has \( T(N) \times N! \) different evaluation plans
  – \( T(N) \) is the number of parenthesized expressions
  – \( N! \) is the number of permutations

• Query optimizer cannot look at all plans (might take longer to find an optimal plan than to compute query brute-force). Hence it does not necessarily produce optimal plan
Relational Algebra Rules

Pushing Selections and Projections

- $\sigma_{Cond} (R \times S) \equiv R \times_{Cond} S$
  - $Cond$ relates attributes of both $R$ and $S$
  - Reduces size of intermediate relation since rows can be discarded sooner

- $\sigma_{Cond} (R \times S) \equiv \sigma_{Cond} (R) \times S$
  - $Cond$ involves only the attributes of $R$
  - Reduces size of intermediate relation since rows of $R$ are discarded sooner

- $\pi_{attr}(R \times S) \equiv \pi_{attr}(\pi_{attr'} (R) \times S)$,
  if $attributes(R) \subseteq attr' \subseteq attr$
  - reduces the size of an operand of product
Oracle Join Methods

Before showing query processing examples, we need to discuss some Oracle Join Methods.

Nested loops join
The nested loop iterates over all rows of the outer table. If there are conditions in the where clause of the SQL statement that apply to the outer table only, it checks whether those apply. If they do, the corresponding rows (from the where condition) in the joined inner table are searched. These rows from the inner table are either found using an index (if a suitable index exists) or by doing a full table scan.

Hash join
Hash joins are used when the joining large tables. The optimizer uses the smaller of the 2 tables to build a hash table in memory and the scans the large table and compares the hash value (of rows from large table) with this hash table to find the joined rows.

Merge join (also called sort merge join)
Sort merge join is used to join two independent data sources. They perform better than nested loop joins when the volume of data is big in tables but not as good as hash joins in general.

They perform better than hash join when the join condition columns are already sorted or there is no sorting required.
Query Processing Example – SQL, Rational Algebra, and Query Tree

```
select * from s_dept d
join s_region r on d.region_id = r.id
join s_warehouse w on r.id = w.region_id;

\[ \prod_* (\sigma_{r.id=w.region_id} (\sigma_{d.region_id=r.id} (\rho_d (s_dept) \times \rho_r (s_region)) \times \rho_w (s_warehouse))) \]
```

Diagram:
```
\( \prod_* \)

\( \sigma_{r.id=w.region_id} \)

\( \times \)

\( \sigma_{d.region_id=r.id} \)

\( \times \)

\( \rho_d (s_dept) \)

\( \times \)

\( \rho_r (s_region) \)

\( \times \)

\( \rho_w (s_warehouse) \)
```
Query Processing Example

```sql
alter session set OPTIMIZER_FEATURES_ENABLE = '11.2.0.1';

select * from s_dept d join s_region r on d.region_id = r.id join s_warehouse w on r.id = w.region_id;
```

Run this first (it disables Adaptive Execution Plans). Then get the Execution Plan for the query.
Cost-Based Oracle Fundamentals

“The insights that Jonathan provides into the workings of the cost-based optimizer will make a DBA a better designer and a developer a better SQL coder. Both groups will become better troubleshooters.”

—Thomas Kyte

Jonathan Lewis

Foreword by Thomas Kyte
Vice President (Public Sector), Oracle Corporation

Apress®
SQL, Rational Algebra, Query Tree, and Optimized Query Tree

```
select * from s_dept
don f.s_region r on d.region_id = r.id
join s_warehouse w on r.id = w.region_id where w.country = 'US';

\[\prod_{d, r, w} (\sigma_{w.county='US'} (\sigma_{r.id=w.region_id} (\sigma_{d.region_id=r.id} (\rho_d(s_dept) \times \rho_r(s_region)) \times \rho_w(s_warehouse))))\]
```

Query Tree

```
\[\prod_{d, r, w} (\sigma_{w.county='US'} (\sigma_{r.id=w.region_id} (\sigma_{d.region_id=r.id} (\rho_d(s_dept) \times \rho_r(s_region)) \times \rho_w(s_warehouse))))\]
```

Optimized Query Tree
Query Processing Example

```
select * from s_dept d join s_region r on d.region_id = r.id join s_warehouse w on r.id = w.region_id where w.country = 'US';
```