Lifetime of a Query

1. **SQL Query**
2. **Scanner/Parser**
3. **Query In Intermediate Form**
4. **Logical Query Plan**
5. **Code Generator**
6. **Query Execution Code**
7. **Query Processor**
8. **Query Result**
Example(s):

```sql
select sname
from s, p, spj
where s.sno = spj.sno
  and spj.pno = p.pno
  and status <> 0
  and pname = 'Nut';
```
Rule-based Optimization (RBO)

Example(s):

\[ \pi_{\text{sname}} \]
\[ \sigma_{\text{status} < > 0 \land \text{pname} = 'Nut'} \]
\[ \bowtie P\# \]
\[ \bowtie S\# \]
\[ P \]
\[ S \]
\[ \text{SPJ} \]
Algorithms for Project and Select
Join Algorithms (1 / 5)

#1: Nested-Loops Join (NLJ)

Join Algorithms (2 / 5)

#2: Sort-Merge Join (SMJ)

The Algorithm:

Make the first tuple of relation r the current tuple
Loop until one of the relations is exhausted:
    Make a set of all tuples from relation r that have the same join values as the current tuple
    For each tuple from relation s that has a join match:
        Output marriages of it to all set members
    The next tuple from relation r that is not a member of the set becomes the new current tuple
#2: Sort-Merge Join (cont.)

Example(s):

#3: Hash Join

The Algorithm:

Hash each relation using the same hash function on the join attributes of the tuples.

For each of the M corresponding pairs of buckets:
   Build an in-memory hash index on the contents of the first bucket in the pair
   For each tuple in the second bucket:
      Probe the index with its join attribute
      For each matching tuple found:
         Output the marriage
Examining Query Plans (1 / 2)

In Oracle 11: Prefix query with `explain plan for`.

    explain plan for
    select sname
    from s, spj, p
    where s.sno = spj.sno
    and spj.pno = p.pno
    and status <> 0
    and pname = 'Nut';

And then issue the following to see the generated plan:

    set linesize 100
    set pagesize 0
    select plan_table_output
         from table(dbms_xplan.display('plan_table',null,'serial'));
Examining Query Plans (2 / 2)

The (slightly abbreviated) output:

```
Explained.
Plan hash value: 3135777751

<table>
<thead>
<tr>
<th>Id</th>
<th>Operation</th>
<th>Name</th>
<th>Rows</th>
<th>Bytes</th>
<th>Cost (%CPU)</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SELECT STATEMENT</td>
<td></td>
<td>4</td>
<td>104</td>
<td>8 (25)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>1</td>
<td>HASH JOIN</td>
<td></td>
<td>4</td>
<td>104</td>
<td>8 (25)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>2</td>
<td>MERGE JOIN</td>
<td></td>
<td>4</td>
<td>56</td>
<td>4 (25)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>3</td>
<td>TABLE ACCESS BY INDEX ROWID</td>
<td>P</td>
<td>1</td>
<td>8</td>
<td>2 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>4</td>
<td>INDEX FULL SCAN</td>
<td>SYS_C0011137</td>
<td>6</td>
<td>1</td>
<td>1 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>5</td>
<td>SORT JOIN</td>
<td></td>
<td>24</td>
<td>144</td>
<td>2 (50)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>6</td>
<td>INDEX FULL SCAN</td>
<td>SYS_C0011150</td>
<td>24</td>
<td>144</td>
<td>1 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>7</td>
<td>TABLE ACCESS FULL</td>
<td></td>
<td>5</td>
<td>60</td>
<td>3 (0)</td>
<td>00:00:01</td>
</tr>
</tbody>
</table>
```

“Vacuuming” a Database (1 / 3)

How to wipe your DBMS’ mind:

In Oracle 11:
```
exec DBMS_STATS.DELETE_SCHEMA_STATS('mccann');
select count(*) from user_histograms;  ==> 0
```

In Postgres:
```
vacuum;
```
“Vacuuming” a Database (2 / 3)

Without its statistics, Oracle generates a different plan:

```
Plan hash value: 2693896614
```

<table>
<thead>
<tr>
<th>Id</th>
<th>Operation</th>
<th>Name</th>
<th>Rows</th>
<th>Bytes</th>
<th>Cost (%CPU)</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SELECT STATEMENT</td>
<td></td>
<td>4</td>
<td>220</td>
<td>9 (12)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>* 1</td>
<td>HASH JOIN</td>
<td></td>
<td>4</td>
<td>220</td>
<td>9 (12)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>2</td>
<td>MERGE JOIN CARTESIAN</td>
<td></td>
<td>5</td>
<td>230</td>
<td>6 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>* 3</td>
<td>TABLE ACCESS FULL</td>
<td>P</td>
<td>1</td>
<td>17</td>
<td>3 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>4</td>
<td>BUFFER SORT</td>
<td></td>
<td>5</td>
<td>145</td>
<td>3 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>* 5</td>
<td>TABLE ACCESS FULL</td>
<td>S</td>
<td>5</td>
<td>145</td>
<td>3 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>6</td>
<td>INDEX FAST FULL SCAN</td>
<td>SYS_C0011150</td>
<td>24</td>
<td>216</td>
<td>2 (0)</td>
<td>00:00:01</td>
</tr>
</tbody>
</table>

```
```

Query Optimization – CSc 460 v1.0 (McCann) – p. 17

“Vacuuming” a Database (3 / 3)

We can command Oracle to collect new stats:

```
exec DBMS_STATS.GATHER_SCHEMA_STATS('mccann');
```

And try again:

```
select count(*) from user_histograms;  ==> 180
```

```
```

And try again:

```
select count(*) from user_histograms;  ==> 180
```

```
```

Back to the original plan!
One way to nudge the DBMS: Rewrite your queries.

Oracle has a different scheme: HINTS
Example: **RULE**

```sql
explain plan for select /*+ RULE */ sname from s, p, spj
  where s.sno = spj.sno and spj.pno = p.pno
  and status <> 0 and pname = 'Nut';
```

Plan hash value: 1885933196

<table>
<thead>
<tr>
<th>Id</th>
<th>Operation</th>
<th>Name</th>
<th>Rows</th>
<th>Bytes</th>
<th>Cost (%CPU)</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SELECT STATEMENT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>NESTED LOOPS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>NESTED LOOPS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>NESTED LOOPS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>TABLE ACCESS FULL</td>
<td>SPJ</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>* 5</td>
<td>INDEX UNIQUE SCAN</td>
<td>SYS_C001137</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>* 6</td>
<td>INDEX UNIQUE SCAN</td>
<td>SYS_C001142</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>* 7</td>
<td>TABLE ACCESS BY INDEX ROWID</td>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note**
- rule based optimizer used (consider using.cbo)

Example: **ALL_ROWS**

```sql
explain plan for select /*+ ALL_ROWS */ sname from s, p, spj
  where s.sno = spj.sno and spj.pno = p.pno
  and status <> 0 and pname = 'Nut';
```

Plan hash value: 3135777751

<table>
<thead>
<tr>
<th>Id</th>
<th>Operation</th>
<th>Name</th>
<th>Rows</th>
<th>Bytes</th>
<th>Cost (%CPU)</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SELECT STATEMENT</td>
<td></td>
<td>4</td>
<td>104</td>
<td>8 (25)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>* 1</td>
<td>HASH JOIN</td>
<td></td>
<td>4</td>
<td>104</td>
<td>8 (25)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>2</td>
<td>MERGE JOIN</td>
<td></td>
<td>4</td>
<td>56</td>
<td>4 (25)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>* 3</td>
<td>TABLE ACCESS BY INDEX ROWID</td>
<td>P</td>
<td>1</td>
<td>8</td>
<td>2 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>4</td>
<td>INDEX FULL SCAN</td>
<td>SYS_C001137</td>
<td>6</td>
<td></td>
<td>1 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>* 5</td>
<td>SORT JOIN</td>
<td></td>
<td>24</td>
<td>144</td>
<td>2 (50)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>6</td>
<td>INDEX FULL SCAN</td>
<td>SYS_C001150</td>
<td>24</td>
<td>144</td>
<td>1 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>* 7</td>
<td>TABLE ACCESS FULL</td>
<td></td>
<td>5</td>
<td>60</td>
<td>3 (0)</td>
<td>00:00:01</td>
</tr>
</tbody>
</table>
Example: FIRST_ROWS

```sql
explain plan for select /*+ FIRST_ROWS */ sname
from s, p, spj
where s.sno = spj.sno and spj.pno = p.pno
and status <> 0 and pname = 'Nut';
```

Plan hash value: 2513296039

<table>
<thead>
<tr>
<th>Id</th>
<th>Operation</th>
<th>Name</th>
<th>Rows</th>
<th>Bytes</th>
<th>Cost (%CPU)</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SELECT STATEMENT</td>
<td></td>
<td>4</td>
<td>104</td>
<td>8 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>1</td>
<td>NESTED LOOPS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>NESTED LOOPS</td>
<td></td>
<td>4</td>
<td>104</td>
<td>8 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>3</td>
<td>NESTED LOOPS</td>
<td></td>
<td>4</td>
<td>56</td>
<td>4 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>* 4</td>
<td>TABLE ACCESS FULL</td>
<td>P</td>
<td>1</td>
<td>8</td>
<td>3 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>5</td>
<td>INDEX FULL SCAN</td>
<td>SYS_C0011150</td>
<td>4</td>
<td>24</td>
<td>1 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>* 6</td>
<td>INDEX UNIQUE SCAN</td>
<td>SYS_C0011142</td>
<td>1</td>
<td></td>
<td>0 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>* 7</td>
<td>TABLE ACCESS BY INDEX ROWID</td>
<td>S</td>
<td>1</td>
<td>12</td>
<td>1 (0)</td>
<td>00:00:01</td>
</tr>
</tbody>
</table>

Example: USE_MERGE()

```sql
explain plan for select /*+ USE_MERGE(s,p) */ sname
from p, spj, s
where s.sno = spj.sno and spj.pno = p.pno
and status <> 0 and pname = 'Nut';
```

Plan hash value: 2745782731

<table>
<thead>
<tr>
<th>Id</th>
<th>Operation</th>
<th>Name</th>
<th>Rows</th>
<th>Bytes</th>
<th>Cost (%CPU)</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SELECT STATEMENT</td>
<td></td>
<td>4</td>
<td>104</td>
<td>8 (13)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>* 1</td>
<td>HASH JOIN</td>
<td></td>
<td>4</td>
<td>104</td>
<td>8 (13)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>2</td>
<td>MERGE JOIN CARTESIAN</td>
<td>P</td>
<td>5</td>
<td>100</td>
<td>6 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>* 3</td>
<td>TABLE ACCESS FULL</td>
<td>P</td>
<td>1</td>
<td>8</td>
<td>3 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>4</td>
<td>BUFFER SORT</td>
<td></td>
<td>5</td>
<td>60</td>
<td>3 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>* 5</td>
<td>TABLE ACCESS FULL</td>
<td>S</td>
<td>5</td>
<td>60</td>
<td>3 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>6</td>
<td>INDEX FULL SCAN</td>
<td>SYS_C0011150</td>
<td>24</td>
<td>144</td>
<td>1 (0)</td>
<td>00:00:01</td>
</tr>
</tbody>
</table>