Datenbank Architektur für Fortgeschrittene

Ausarbeitung 1: Anfrageverarbeitung

Daniel Gürber
Stefan Eggenschwiler

02.05.2013

Inhaltsverzeichnis

1 Vorbereitung

1.1 Einleitung

Diese Ausarbeitung behandelt die Übung „SQL Tuning“. Sie wird Schritt für Schritt gelöst. Gezeigt werden das SQL-Statement und den dazugehörigen Ausführungsplan, der von der Oracle Datenbank generiert wird. Bei nennenswerten Erkenntnissen werden diese unterhalb des Ausführungsplan in einer kurzen Reflexion behandelt. Die Nummerierung im Dokument entspricht dabei der Nummern des Übungsblattes. Um die Übung auszuführen haben wir folgende Verbindungsdaten verwendet:

Connection Name hades11gdbarc03
Username dbarc03
Password ¡YouKnowIt¿
Role default
Connection Type Basic
Hostname hades.imvs.technik.fhnw.ch
Port 1521
SID ¡kein Eintrag¿
Service Name hades11g.hades.fhnw.ch

1.2 Einrichten Datenbasis

```
CREATE TABLE regions
AS SELECT *
  FROM dbarc00.regions;

CREATE TABLE nations
AS SELECT *
  FROM dbarc00.nations;

CREATE TABLE parts
AS SELECT *
  FROM dbarc00.parts;

CREATE TABLE customers
AS SELECT *
  FROM dbarc00.customers;

CREATE TABLE suppliers
AS SELECT *
  FROM dbarc00.suppliers;

CREATE TABLE orders
AS SELECT *
  FROM dbarc00.orders;

CREATE TABLE partsupps
AS SELECT *
  FROM dbarc00.partsupps;

CREATE TABLE lineitems
AS SELECT *
  FROM dbarc00.lineitems;
```

2 Statistiken erheben

```
BEGIN
  DBMS_STATS.GATHER_TABLE_STATS('dbarc00','parts');
END;
```

Tabelle	Anzahl Zeilen	Grösse in Bytes	Anzahl Blöcke	Anzahl Extents
CUSTOMERS	150000	23850000	3494	43
LINEITEMS	6001215	750151875	109217	186
NATIONS	25	2675	4	1
ORDER	1500000	166500000	24284	95
PARTS	200000	26400000	3859	46
PARTSUPPS	800000	114400000	16650	88
REGIONS	5	480	4	1
SUPPLIERS	10000	1440000	220	17

3 Ausführungsplan

```
EXPLAIN PLAN FOR
SELECT *
FROM parts;
```

```
SELECT plan_table_output
FROM TABLE(DBMS_XPLAN.DISPLAY('plan_table',null,'serial'));
```

```
---------------------------------------------------------------------------
| Id  | Operation         | Name  | Rows  | Bytes | Cost (%CPU)| Time     |
---------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |       |   200K|    25M|  1051   (1)| 00:00:13 |
|   1 |  TABLE ACCESS FULL| PARTS |   200K|    25M|  1051   (1)| 00:00:13 |
---------------------------------------------------------------------------
```

Reflexion
BLABLA

4 Versuche ohne Index

4.1 Projektion

```
EXPLAIN PLAN FOR
SELECT *
FROM ORDERS;
```

```
----------------------------------------------------------------------------
| Id  | Operation         | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
----------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |        |  1500K|   158M|  6610   (1)| 00:01:20 |
|   1 |  TABLE ACCESS FULL| ORDERS |  1500K|   158M|  6610   (1)| 00:01:20 |
----------------------------------------------------------------------------
```

Reflexion
BLABLA

```
EXPLAIN PLAN FOR
SELECT o_clerk
FROM ORDERS;
```

```
----------------------------------------------------------------------------
| Id  | Operation         | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
----------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |        |  1500K|    22M|  6607   (1)| 00:01:20 |
|   1 |  TABLE ACCESS FULL| ORDERS |  1500K|    22M|  6607   (1)| 00:01:20 |
----------------------------------------------------------------------------
```

Reflexion
BLABLA

```
EXPLAIN PLAN FOR
SELECT DISTINCT o_clerk
FROM ORDERS;
```

```
--------------------------------------------------------------------------------
| Id  | Operation           | Name    | Rows   | Bytes | Cost (%CPU)| Time     |
--------------------------------------------------------------------------------
|   0 | SELECT STATEMENT    |         |  1000  | 16000 |  6676   (2)| 00:01:21 |
|   1 |  HASH UNIQUE        |         |  1000  | 16000 |  6676   (2)| 00:01:21 |
|   2 |   TABLE ACCESS FULL| ORDERS  |  1500K|    22M|  6607   (1)| 00:01:20 |
--------------------------------------------------------------------------------
```

Reflexion
BLABLA

4.2 Selektion

Exact Point Query

```
EXPLAIN PLAN FOR
SELECT *
FROM orders
WHERE o_orderkey=44444;
```

```
-------------------------------------------------------------------------------
| Id  | Operation          | Name    | Rows   | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |         |      1 |   111 |  6602   (1)| 00:01:20 |
|*  1 |  TABLE ACCESS FULL| ORDERS |      1 |   111 |  6602   (1)| 00:01:20 |
-------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   1 - filter("O_ORDERKEY"=44444)
```

Reflexion
BLABLA

Partial Point Query

```
EXPLAIN PLAN FOR
SELECT *
FROM orders
WHERE o_orderkey=44444 OR o_clerk='Clerk#000000286';
```

```
-------------------------------------------------------------------------------
| Id  | Operation          | Name    | Rows   | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |         |  1501  |   162K|  6629   (1)| 00:01:20 |
|*  1 |  TABLE ACCESS FULL| ORDERS |  1501  |   162K|  6629   (1)| 00:01:20 |
-------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   1 - filter("O_CLERK"='Clerk#000000286' OR "O_ORDERKEY"=44444)
```

Reflexion
BLABLA

```
EXPLAIN PLAN FOR
SELECT *
FROM orders
WHERE o_orderkey=44444 AND o_clerk='Clerk#000000286';
```

```
-----------------------------------------------------------------------------
| Id  | Operation         | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |        |     1 |   111 |  6611   (1)| 00:01:20 |
|*  1 |  TABLE ACCESS FULL| ORDERS |     1 |   111 |  6611   (1)| 00:01:20 |
-----------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   1 - filter("O_ORDERKEY"=44444 AND "O_CLERK"=’Clerk#000000286’)
```

Reflexion

BLABLA

```sql
EXPLAIN PLAN FOR
SELECT *
FROM orders
WHERE o_orderkey*2=44444 AND o_clerk=’Clerk#000000286’;
```

```
-----------------------------------------------------------------------------
| Id  | Operation         | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |        |    15 |  1665 |  6615   (1)| 00:01:20 |
|*  1 |  TABLE ACCESS FULL| ORDERS |    15 |  1665 |  6615   (1)| 00:01:20 |
-----------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------
   1 - filter("O_ORDERKEY"*2=44444 AND "O_CLERK"=’Clerk#000000286’)
```

Reflexion

BLABLA

Range Query

```sql
EXPLAIN PLAN FOR
SELECT *
FROM orders
WHERE o_orderkey BETWEEN 111111 AND 222222
```

```
-----------------------------------------------------------------------------
| Id  | Operation         | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |        | 27780 |  3011K|  6603   (1)| 00:01:20 |
|*  1 |  TABLE ACCESS FULL| ORDERS | 27780 |  3011K|  6603   (1)| 00:01:20 |
-----------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   1 - filter("O_ORDERKEY"<=222222 AND "O_ORDERKEY">=111111)
```

Reflexion

Die Intervallgrösse spielt in diesem Beispiel keine Rolle.

Partial Range Query

```sql
EXPLAIN PLAN FOR
SELECT *
FROM orders
WHERE o_orderkey BETWEEN 44444 AND 55555
AND   o_clerk BETWEEN ’Clerk#000000130’ AND ’Clerk#000000139’
```

```
-----------------------------------------------------------------------------
| Id  | Operation         | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |        |     6 |   666 |  6611   (1)| 00:01:20 |
|*  1 |  TABLE ACCESS FULL| ORDERS |     6 |   666 |  6611   (1)| 00:01:20 |
-----------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   1 - filter("O_ORDERKEY"<=55555 AND "O_CLERK"<=’Clerk#000000139’ AND
              "O_ORDERKEY">=44444 AND "O_CLERK">=’Clerk#000000130’)
```

Reflexion
BLABLA

4.3 Join

```
EXPLAIN PLAN FOR
SELECT *
FROM orders, customers
WHERE o_custkey = c_custkey
AND   o_orderkey < 100;
```

```
--------------------------------------------------------------------------------
| Id  | Operation          | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
--------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |           |    25 |  6750 |  7555   (1)| 00:01:31 |
|*  1 |  HASH JOIN         |           |    25 |  6750 |  7555   (1)| 00:01:31 |
|*  2 |   TABLE ACCESS FULL| ORDERS    |    25 |  2775 |  6602   (1)| 00:01:20 |
|   3 |   TABLE ACCESS FULL| CUSTOMERS |   150K|    22M|   951   (1)| 00:00:12 |
--------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   1 - access("O_CUSTKEY"="C_CUSTKEY")
   2 - filter("O_ORDERKEY"<100)
```

Reflexion
Spielen Varianten in der Formulierung eine Rolle? //NO DIFFERENCE FOR INNER JOIN

5 Versuch mit Index

```
CREATE INDEX o_orderkey_ix ON orders(o_orderkey);
```

```
CREATE INDEX o_clerk_ix ON orders(o_clerk);
```

Index Name	Index Grösse	Tabellengrösse in Bytes
o_orderkey_ix	60817408	166500000
o_clerik_ix	96468992	166500000

5.1 Projektion

```
EXPLAIN PLAN FOR
SELECT DISTINCT o_clerk
FROM ORDERS;
```

```
------------------------------------------------------------------------------------
| Id  | Operation             | Name       | Rows  | Bytes | Cost (%CPU)| Time     |
------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |            |  1000 | 16000 |  1615   (5)| 00:00:20 |
|   1 |  HASH UNIQUE          |            |  1000 | 16000 |  1615   (5)| 00:00:20 |
|   2 |   INDEX FAST FULL SCAN| O_CLERK_IX |  1500K|    22M|  1546   (1)| 00:00:19 |
------------------------------------------------------------------------------------
```

Reflexion
BLABLA

5.2 Selektion

Exact Point Query

```
EXPLAIN PLAN FOR
SELECT *
FROM ORDERS
WHERE o_orderkey=44444
```

```
----------------------------------------------------------------------------------------------
| Id  | Operation                   | Name           | Rows  | Bytes | Cost (%CPU)| Time     |
----------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT            |                |     1 |   111 |     4   (0)| 00:00:01 |
|   1 |  TABLE ACCESS BY INDEX ROWID| ORDERS         |     1 |   111 |     4   (0)| 00:00:01 |
|*  2 |   INDEX RANGE SCAN          | O_ORDERKEY_IX  |     1 |       |     3   (0)| 00:00:01 |
----------------------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   2 - access("O_ORDERKEY"=44444)
```

```
EXPLAIN PLAN FOR
SELECT /*+ FULL(orders) */ *
FROM ORDERS
WHERE o_orderkey=44444
```

```
----------------------------------------------------------------------------
| Id  | Operation         | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
----------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |        |     1 |   111 |  6602   (1)| 00:01:20 |
|*  1 |  TABLE ACCESS FULL| ORDERS |     1 |   111 |  6602   (1)| 00:01:20 |
----------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   1 - filter("O_ORDERKEY"=44444)
```

Reflexion
BLABLA

Partial Point Query

```
EXPLAIN PLAN FOR
SELECT *
FROM orders
WHERE o_orderkey=44444 OR o_clerk='Clerk#000000286';
```

```
---------------------------------------------------------------------------------------------------
| Id  | Operation                        | Name           | Rows  | Bytes | Cost (%CPU)| Time     |
---------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                 |                |  1501 |   162K|   336   (0)| 00:00:05 |
|   1 |  TABLE ACCESS BY INDEX ROWID     | ORDERS         |  1501 |   162K|   336   (0)| 00:00:05 |
|   2 |   BITMAP CONVERSION TO ROWIDS    |                |       |       |            |          |
|   3 |    BITMAP OR                     |                |       |       |            |          |
|   4 |     BITMAP CONVERSION FROM ROWIDS|                |       |       |            |          |
|*  5 |      INDEX RANGE SCAN            | O_CLERK_IX     |       |       |     8   (0)| 00:00:01 |
|   6 |     BITMAP CONVERSION FROM ROWIDS|                |       |       |            |          |
|*  7 |      INDEX RANGE SCAN            | O_ORDERKEY_IX  |       |       |     3   (0)| 00:00:01 |
---------------------------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   5 - access("O_CLERK"='Clerk#000000286')
   7 - access("O_ORDERKEY"=44444)
```

Reflexion
BLABLA

```
EXPLAIN PLAN FOR
SELECT *
FROM orders
WHERE o_orderkey=44444 AND o_clerk='Clerk#000000286';
```

```
-----------------------------------------------------------------------------------------
| Id  | Operation                   | Name           | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT            |                |     1 |   111 |     4   (0)| 00:00:01 |
|*  1 |  TABLE ACCESS BY INDEX ROWID| ORDERS         |     1 |   111 |     4   (0)| 00:00:01 |
|*  2 |   INDEX RANGE SCAN          | O_ORDERKEY_IX |     1 |       |     3   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   1 - filter("O_CLERK"='Clerk#000000286')
   2 - access("O_ORDERKEY"=44444)
```

Reflexion
BLABLA

```
EXPLAIN PLAN FOR
SELECT *
FROM orders
WHERE o_orderkey*2=44444 AND o_clerk='Clerk#000000286';
```

```
--------------------------------------------------------------------------------------
| Id  | Operation                   | Name        | Rows  | Bytes | Cost (%CPU)| Time     |
--------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT            |             |    15 |  1665 |  1464   (1)| 00:00:18 |
|*  1 |  TABLE ACCESS BY INDEX ROWID| ORDERS      |    15 |  1665 |  1464   (1)| 00:00:18 |
|*  2 |   INDEX RANGE SCAN          | O_CLERK_IX |  1500 |       |     8   (0)| 00:00:01 |
--------------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   1 - filter("O_ORDERKEY"*2=44444)
   2 - access("O_CLERK"='Clerk#000000286')
```

Reflexion
BLABLA

Range Query

```
EXPLAIN PLAN FOR
SELECT *
FROM orders
WHERE o_orderkey BETWEEN 111111 AND 222222
```

```
-----------------------------------------------------------------------------------------
| Id  | Operation                   | Name           | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT            |                | 27780 |  3011K|   932   (1)| 00:00:12 |
|   1 |  TABLE ACCESS BY INDEX ROWID| ORDERS         | 27780 |  3011K|   932   (1)| 00:00:12 |
|*  2 |   INDEX RANGE SCAN          | O_ORDERKEY_IX | 27780 |       |    68   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   2 - access("O_ORDERKEY">=111111 AND "O_ORDERKEY"<=222222)
```

Reflexion
Spielt der Intervall eine Rolle?

Partial Range Query

```
EXPLAIN PLAN FOR
SELECT *
FROM orders
WHERE o_orderkey BETWEEN 44444 AND 55555
AND   o_clerk BETWEEN 'Clerk#000000130' AND 'Clerk#000000139'
```

```
----------------------------------------------------------------------------------------------
| Id  | Operation                        | Name           | Rows  | Bytes | Cost (%CPU)| Time     |
----------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                 |                |     6 |   666 |    27  (12)| 00:00:01 |
|   1 |  TABLE ACCESS BY INDEX ROWID     | ORDERS         |     6 |   666 |    27  (12)| 00:00:01 |
|   2 |   BITMAP CONVERSION TO ROWIDS    |                |       |       |            |          |
|   3 |    BITMAP AND                    |               |       |       |            |          |
|   4 |     BITMAP CONVERSION FROM ROWIDS|               |       |       |            |          |
|   5 |      SORT ORDER BY               |               |       |       |            |          |
|*  6 |       INDEX RANGE SCAN           | O_ORDERKEY_IX |  2780 |       |     9   (0)| 00:00:01 |
|   7 |     BITMAP CONVERSION FROM ROWIDS|               |       |       |            |          |
|   8 |      SORT ORDER BY               |               |       |       |            |          |
|*  9 |       INDEX RANGE SCAN           | O_CLERK_IX    |  2780 |       |    14   (0)| 00:00:01 |
--------------------------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   6 - access("O_ORDERKEY">=44444 AND "O_ORDERKEY"<=55555)
   9 - access("O_CLERK">='Clerk#000000130' AND "O_CLERK"<='Clerk#000000139')
```

Reflexion
BLABLA

5.3 Join

```sql
EXPLAIN PLAN FOR
SELECT *
FROM orders, customers
WHERE o_custkey = c_custkey;
```

```
----------------------------------------------------------------------------------------
| Id  | Operation          | Name      | Rows  | Bytes |TempSpc| Cost (%CPU)| Time     |
----------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |           |  1500K|   386M|       | 17514   (1)| 00:03:31 |
|*  1 |  HASH JOIN         |           |  1500K|   386M|    24M| 17514   (1)| 00:03:31 |
|   2 |   TABLE ACCESS FULL| CUSTOMERS |   150K|    22M|       |   951   (1)| 00:00:12 |
|   3 |   TABLE ACCESS FULL| ORDERS    |  1500K|   158M|       |  6610   (1)| 00:01:20 |
----------------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   1 - access("O_CUSTKEY"="C_CUSTKEY")
```

Reflexion
BLABLA

```sql
EXPLAIN PLAN FOR
SELECT *
FROM orders, customers
WHERE o_custkey = c_custkey
AND   o_orderkey < 100;
```

```
----------------------------------------------------------------------------------------------
| Id  | Operation                    | Name          | Rows  | Bytes | Cost (%CPU)| Time     |
----------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |               |    25 |  6750 |   957   (1)| 00:00:12 |
|*  1 |  HASH JOIN                   |               |    25 |  6750 |   957   (1)| 00:00:12 |
|   2 |   TABLE ACCESS BY INDEX ROWID| ORDERS        |    25 |  2775 |     4   (0)| 00:00:01 |
|*  3 |    INDEX RANGE SCAN          | O_ORDERKEY_IX |    25 |       |     3   (0)| 00:00:01 |
|   4 |   TABLE ACCESS FULL          | CUSTOMERS     |   150K|    22M|   951   (1)| 00:00:12 |
----------------------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   1 - access("O_CUSTKEY"="C_CUSTKEY")
   3 - access("O_ORDERKEY"<100)
```

Reflexion
BLABLA

```sql
CREATE INDEX c_custkey_ix ON customers(c_custkey);
```

```sql
EXPLAIN PLAN FOR
SELECT *
FROM orders, customers
WHERE o_custkey = c_custkey;
```

```
----------------------------------------------------------------------------------------
| Id  | Operation          | Name      | Rows  | Bytes |TempSpc| Cost (%CPU)| Time     |
----------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |           |  1500K|   386M|       | 17514   (1)| 00:03:31 |
|*  1 |  HASH JOIN         |           |  1500K|   386M|    24M| 17514   (1)| 00:03:31 |
|   2 |   TABLE ACCESS FULL| CUSTOMERS |   150K|    22M|       |   951   (1)| 00:00:12 |
|   3 |   TABLE ACCESS FULL| ORDERS    |  1500K|   158M|       |  6610   (1)| 00:01:20 |
----------------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   1 - access("O_CUSTKEY"="C_CUSTKEY")
```

Reflexion
BLABLA

Erzwingen eines Nested Loop Joins:

```
EXPLAIN PLAN FOR
SELECT /*+ USE_NL (o c) */ *
FROM orders o, customers c
WHERE o_custkey = c_custkey;
```

```
---------------------------------------------------------------------------------------------
| Id  | Operation                    | Name          | Rows  | Bytes | Cost (%CPU)| Time     |
---------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |               |  1500K|   386M|  3007K  (1)| 10:01:34 |
|   1 |  NESTED LOOPS                |               |       |       |            |          |
|   2 |   NESTED LOOPS               |               |  1500K|   386M|  3007K  (1)| 10:01:34 |
|   3 |    TABLE ACCESS FULL         | ORDERS        |  1500K|   158M|  6610   (1)| 00:01:20 |
|*  4 |    INDEX RANGE SCAN          | C_CUSTKEY_IX  |     1 |       |     1   (0)| 00:00:01 |
|   5 |   TABLE ACCESS BY INDEX ROWID| CUSTOMERS     |     1 |   159 |     2   (0)| 00:00:01 |
---------------------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   4 - access("O_CUSTKEY"="C_CUSTKEY")
```

Reflexion
BLABLA

Erzwingen eines Nicht-Hash Joins:

```
EXPLAIN PLAN FOR
SELECT /*+ NO_USE_HASH (o c) */ *
FROM orders o, customers c
WHERE o_custkey = c_custkey;
```

```
-----------------------------------------------------------------------------------------
| Id  | Operation           | Name      | Rows  | Bytes |TempSpc| Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT    |           |  1500K|   386M|       | 50568   (1)| 00:10:07 |
|   1 |  MERGE JOIN         |           |  1500K|   386M|       | 50568   (1)| 00:10:07 |
|   2 |   SORT JOIN         |           |   150K|    22M|    52M|  6202   (1)| 00:01:15 |
|   3 |    TABLE ACCESS FULL| CUSTOMERS |   150K|    22M|       |   951   (1)| 00:00:12 |
|*  4 |   SORT JOIN         |           |  1500K|   158M|   390M| 44366   (1)| 00:08:53 |
|   5 |    TABLE ACCESS FULL| ORDERS    |  1500K|   158M|       |  6610   (1)| 00:01:20 |
-----------------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   4 - access("O_CUSTKEY"="C_CUSTKEY")
       filter("O_CUSTKEY"="C_CUSTKEY")
```

Reflexion
BLABLA

6 Quiz

```
EXPLAIN PLAN FOR
SELECT count(*)
FROM parts, partsupps, lineitems
WHERE p_partkey=ps_partkey
AND ps_partkey=l_partkey
AND ps_suppkey=l_suppkey
AND ((ps_partkey = 5 AND p_type = 'MEDIUM ANODIZED BRASS')
OR (ps_partkey = 5 AND p_type = 'MEDIUM BRUSHED COPPER'));
```

Ausgangslage:

```
--------------------------------------------------------------------------------
| Id  | Operation            | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
--------------------------------------------------------------------------------
|   0 | SELECT STATEMENT     |           |     1 |    45 | 35577   (2)| 00:07:07 |
|   1 |  SORT AGGREGATE      |           |     1 |    45 |            |          |
|*  2 |   HASH JOIN          |           |     4 |   180 | 35577   (2)| 00:07:07 |
|*  3 |    HASH JOIN         |           |     4 |   144 |  5872   (6)| 00:01:11 |
|*  4 |     TABLE ACCESS FULL| PARTSUPPS |     4 |    36 |  4525   (1)| 00:00:55 |
|*  5 |     TABLE ACCESS FULL| PARTS     |  2667 | 72009 |  1052   (1)| 00:00:13 |
|   6 |    TABLE ACCESS FULL | LINEITEMS |  6001K|    51M| 29675   (1)| 00:05:57 |
--------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   2 - access("PS_PARTKEY"="L_PARTKEY" AND "PS_SUPPKEY"="L_SUPPKEY")
   3 - access("P_PARTKEY"="PS_PARTKEY")
       filter("PS_PARTKEY"=5 AND "P_TYPE"='MEDIUM ANODIZED BRASS' OR
              "PS_PARTKEY"=5 AND "P_TYPE"='MEDIUM BRUSHED COPPER')
   4 - filter("PS_PARTKEY"=5)
```

ps_partkey ist restriktiv:

```
CREATE INDEX ps_partkey_ix ON partsupps(ps_partkey);
```

```
----------------------------------------------------------------------------------------------
| Id  | Operation                     | Name          | Rows  | Bytes | Cost (%CPU)| Time     |
----------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |               |     1 |    45 | 31055   (2)| 00:06:13 |
|   1 |  SORT AGGREGATE               |               |     1 |    45 |            |          |
|*  2 |   HASH JOIN                   |               |     4 |   180 | 31055   (2)| 00:06:13 |
|*  3 |    HASH JOIN                  |               |     4 |   144 |  1350  (23)| 00:00:17 |
|   4 |     TABLE ACCESS BY INDEX ROWID| PARTSUPPS    |     4 |    36 |     4   (0)| 00:00:01 |
|*  5 |      INDEX RANGE SCAN         | PS_PARTKEY_IX |     4 |       |     3   (0)| 00:00:01 |
|*  6 |     TABLE ACCESS FULL         | PARTS         |  2667 | 72009 |  1052   (1)| 00:00:13 |
|   7 |    TABLE ACCESS FULL          | LINEITEMS     |  6001K|    51M| 29675   (1)| 00:05:57 |
----------------------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   2 - access("PS_PARTKEY"="L_PARTKEY" AND "PS_SUPPKEY"="L_SUPPKEY")
   3 - access("P_PARTKEY"="PS_PARTKEY")
       filter("PS_PARTKEY"=5 AND "P_TYPE"='MEDIUM ANODIZED BRASS' OR "PS_PARTKEY"=5 AND
              "P_TYPE"='MEDIUM BRUSHED COPPER')
```

p_type ist restriktiv:

```
CREATE INDEX p_type_ix ON parts(p_type);
```

```
----------------------------------------------------------------------------------------------------
| Id  | Operation                           | Name          | Rows  | Bytes | Cost (%CPU)| Time     |
----------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                    |               |     1 |    45 | 29885   (1)| 00:05:59 |
|   1 |  SORT AGGREGATE                     |               |     1 |    45 |            |          |
|*  2 |   HASH JOIN                         |               |     4 |   180 | 29885   (1)| 00:05:59 |
|   3 |    NESTED LOOPS                     |               |       |       |            |          |
|   4 |     NESTED LOOPS                    |               |     4 |   144 |   180   (0)| 00:00:03 |
|   5 |      TABLE ACCESS BY INDEX ROWID    | PARTSUPPS     |     4 |    36 |     4   (0)| 00:00:01 |
|*  6 |       INDEX RANGE SCAN              | PS_PARTKEY_IX |     4 |       |     3   (0)| 00:00:01 |
|   7 |      BITMAP CONVERSION TO ROWIDS    |               |       |       |            |          |
|   8 |       BITMAP AND                    |               |       |       |            |          |
|   9 |        BITMAP OR                    |               |       |       |            |          |
|  10 |         BITMAP CONVERSION FROM ROWIDS|              |       |       |            |          |
|* 11 |          INDEX RANGE SCAN            | P_TYPE_IX     |       |       |     8   (0)| 00:00:01 |
|  12 |         BITMAP CONVERSION FROM ROWIDS|               |       |       |            |          |
|* 13 |          INDEX RANGE SCAN            | P_TYPE_IX     |       |       |     8   (0)| 00:00:01 |
|  14 |        BITMAP OR                     |               |       |       |            |          |
|  15 |         BITMAP CONVERSION FROM ROWIDS|               |       |       |            |          |
|* 16 |          INDEX RANGE SCAN            | P_TYPE_IX     |       |       |     8   (0)| 00:00:01 |
|  17 |         BITMAP CONVERSION FROM ROWIDS|               |       |       |            |          |
|* 18 |          INDEX RANGE SCAN            | P_TYPE_IX     |       |       |     8   (0)| 00:00:01 |
|* 19 |     TABLE ACCESS BY INDEX ROWID      | PARTS         |     1 |    27 |   180   (0)| 00:00:03 |
|  20 |    TABLE ACCESS FULL                 | LINEITEMS     |  6001K|    51M| 29675   (1)| 00:05:57 |
------------------------------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   2 - access("PS_PARTKEY"="L_PARTKEY" AND "PS_SUPPKEY"="L_SUPPKEY")
   6 - access("PS_PARTKEY"=5)
  11 - access("P_TYPE"='MEDIUM ANODIZED BRASS')
  13 - access("P_TYPE"='MEDIUM BRUSHED COPPER')
  16 - access("P_TYPE"='MEDIUM ANODIZED BRASS')
  18 - access("P_TYPE"='MEDIUM BRUSHED COPPER')
  19 - filter("P_PARTKEY"="PS_PARTKEY" AND ("PS_PARTKEY"=5 AND "P_TYPE"='MEDIUM ANODIZED
```

l_partkey ist restriktiv:

```sql
CREATE INDEX l_partkey_ix ON lineitems(l_partkey);
```

```
-------------------------------------------------------------------------------------------------------
| Id  | Operation                             | Name          | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                      |               |     1 |    45 |   308   (0)| 00:00:04 |
|   1 |  SORT AGGREGATE                       |               |     1 |    45 |            |          |
|   2 |   NESTED LOOPS                        |               |       |       |            |          |
|   3 |    NESTED LOOPS                       |               |     4 |   180 |   308   (0)| 00:00:04 |
|   4 |     NESTED LOOPS                      |               |     4 |   144 |   180   (0)| 00:00:03 |
|   5 |      TABLE ACCESS BY INDEX ROWID      | PARTSUPPS     |     4 |    36 |     4   (0)| 00:00:01 |
|*  6 |       INDEX RANGE SCAN                | PS_PARTKEY_IX |     4 |       |     3   (0)| 00:00:01 |
|*  7 |      TABLE ACCESS BY INDEX ROWID      | PARTS         |     1 |    27 |   180   (0)| 00:00:03 |
|   8 |       BITMAP CONVERSION TO ROWIDS     |               |       |       |            |          |
|   9 |        BITMAP AND                     |               |       |       |            |          |
|  10 |         BITMAP OR                     |               |       |       |            |          |
|  11 |          BITMAP CONVERSION FROM ROWIDS|               |       |       |            |          |
|* 12 |           INDEX RANGE SCAN            | P_TYPE_IX     |       |       |     8   (0)| 00:00:01 |
|  13 |          BITMAP CONVERSION FROM ROWIDS|               |       |       |            |          |
|* 14 |           INDEX RANGE SCAN            | P_TYPE_IX     |       |       |     8   (0)| 00:00:01 |
|  15 |         BITMAP OR                     |               |       |       |            |          |
|  16 |          BITMAP CONVERSION FROM ROWIDS|               |       |       |            |          |
|* 17 |           INDEX RANGE SCAN            | P_TYPE_IX     |       |       |     8   (0)| 00:00:01 |
|  18 |          BITMAP CONVERSION FROM ROWIDS|               |       |       |            |          |
|* 19 |           INDEX RANGE SCAN            | P_TYPE_IX     |       |       |     8   (0)| 00:00:01 |
|* 20 |     INDEX RANGE SCAN                  | L_PARTKEY_IX  |    30 |       |     2   (0)| 00:00:01 |
|* 21 |    TABLE ACCESS BY INDEX ROWID        | LINEITEMS     |     1 |     9 |    32   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   6 - access("PS_PARTKEY"=5)
   7 - filter("P_PARTKEY"="PS_PARTKEY" AND ("PS_PARTKEY"=5 AND "P_TYPE"='MEDIUM ANODIZED
              BRASS' OR "PS_PARTKEY"=5 AND "P_TYPE"='MEDIUM BRUSHED COPPER'))
  12 - access("P_TYPE"='MEDIUM ANODIZED BRASS')
  14 - access("P_TYPE"='MEDIUM BRUSHED COPPER')
  17 - access("P_TYPE"='MEDIUM ANODIZED BRASS')
  19 - access("P_TYPE"='MEDIUM BRUSHED COPPER')
  20 - access("PS_PARTKEY"="L_PARTKEY")
  21 - filter("PS_SUPPKEY"="L_SUPPKEY")
```

Optimierte Abfrage:

```sql
EXPLAIN PLAN FOR
SELECT count(*)
FROM parts, partsupps, lineitems, orders
WHERE p_partkey=ps_partkey
AND ps_partkey=l_partkey
AND ps_suppkey=l_suppkey
AND l_orderkey=o_orderkey
```

```
-------------------------------------------------------------------------------------------
| Id  | Operation             | Name      | Rows  | Bytes |TempSpc| Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT      |           |     1 |    35 |       | 54346   (1)| 00:10:53 |
|   1 |  SORT AGGREGATE       |           |     1 |    35 |       |            |          |
|*  2 |   HASH JOIN           |           |   803K|    26M|    25M| 54346   (1)| 00:10:53 |
|   3 |    TABLE ACCESS FULL  | ORDERS    |  1500K|  8789K|       |  6599   (1)| 00:01:20 |
|*  4 |    HASH JOIN          |           |   792K|    21M|    19M| 44915   (1)| 00:08:59 |
|*  5 |     HASH JOIN         |           |   792K|    10M|  3328K|  6540   (1)| 00:01:19 |
|   6 |      TABLE ACCESS FULL| PARTS     |   200K|   976K|       |  1050   (1)| 00:00:13 |
|   7 |      TABLE ACCESS FULL| PARTSUPPS |   800K|  7031K|       |  4523   (1)| 00:00:55 |
|   8 |     TABLE ACCESS FULL | LINEITEMS |  6001K|    85M|       | 29675   (1)| 00:05:57 |
-------------------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------
   2 - access("L_ORDERKEY"="O_ORDERKEY")
   4 - access("PS_PARTKEY"="L_PARTKEY" AND "PS_SUPPKEY"="L_SUPPKEY")
   5 - access("P_PARTKEY"="PS_PARTKEY")
```

Reflexion
BLABLA

7 Deep Left Join

8 Eigene SQL-Abfragen