



Index oder nicht Index

Johannes Ahrends
Geschäftsführer
CarajanDB GmbH

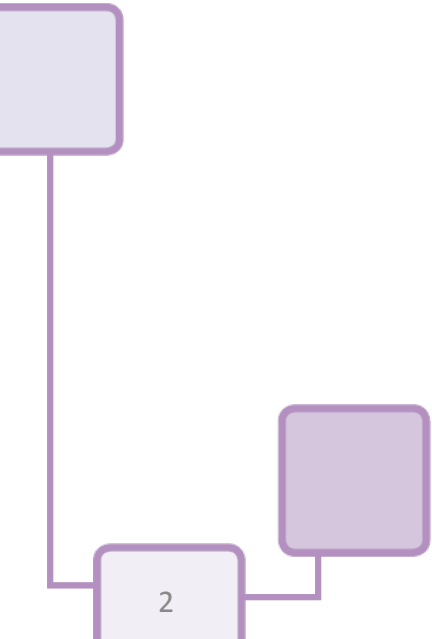


ORACLE
ACE

DOAG
Deutsche ORACLE-Anwendergruppe e.V.



- Vorstellung CarajanDB
- I
- Index oder nicht Index – das ist doch keine Frage, oder?
- Was kann der DBA tun?
- Was kann der Entwickler tun?



- **Experten mit über 30 Jahren Oracle Erfahrung**
- **Spezialisten für**
 - Backup & Recovery
 - Hochverfügbarkeit
 - Healthchecks
 - Performance Optimierung
 - Einsatz von Oracle Standard Edition
 - Oracle in virtuellen Umgebungen und in der Cloud
 - Oracle Migrationen (HW, Unicode, Konsolidierung, Standard Edition)
 - Monitoring (Grid / Cloud Control, HLMM, Foglight, Spotlight)
- **Schulung und Workshops (Oracle, Toad)**



➤ Sind diese Indizes sinnvoll?

```
CREATE INDEX idx_nachname  
  ON personen (nachname);  
CREATE INDEX idx_vorname  
  ON personen (vorname);
```

➤ Abfrage 1:

```
SELECT anrede, vorname, nachname  
  FROM tuk.personen pe  
 WHERE pe.nachname LIKE 'Wei_'  
       AND pe.vorname = 'Martin'  
       AND pe.anrede = 'Herr';
```

Bitmap Conversion

The screenshot shows a SQL IDE window titled 'TUK@JAWIN11 - Editor (0_einfacher_index.sql)'. The main editor displays a SQL query:

```
13 SELECT vorname, nachname
14 FROM tuk.personen pe
15 WHERE pe.nachname LIKE 'Wei_'
16 AND pe.vorname = 'Martin'
17 AND pe.anrede = 'Herr';
```

Below the editor is the 'Explain Plan' tab. The 'Cached Explain Plan' section shows the following plan:

```
Plan
SELECT STATEMENT ALL_ROWS
Cost: 5
8 TABLE ACCESS BY INDEX ROWID TABLE TUK.PERSONEN
Cost: 5 Bytes: 20 Cardinality: 1
7 BITMAP CONVERSION TO ROWIDS
6 BITMAP AND
2 BITMAP CONVERSION FROM ROWIDS
1 INDEX RANGE SCAN INDEX TUK.IDX_VORNAME
Cost: 1 Cardinality: 21
5 BITMAP CONVERSION FROM ROWIDS
4 SORT ORDER BY
3 INDEX RANGE SCAN INDEX TUK.IDX_NACHNAME
Cost: 3 Cardinality: 21
```

A detailed description for step 6 is provided at the bottom:

6. A logical AND operation was performed on the bitmaps from steps 2, 5.

The status bar at the bottom indicates: 17: 30, 19 msec, Row 1 of 3 total rows, TUK@JAWIN11.

➤ Ist dieser Index sinnvoll?

```
CREATE INDEX idx_name  
ON personen (anrede, vorname, nachname);
```

➤ Abfrage 1:

```
SELECT anrede, vorname, nachname  
FROM tuk.personen pe  
WHERE pe.nachname LIKE 'Wei_'  
AND pe.vorname = 'Martin'  
AND pe.anrede = 'Herr';
```

Index Range Scan

The screenshot shows a SQL IDE window titled "TUK@JAWIN11 - Editor (select_nls.sql *)". The editor contains the following SQL query:

```
1 SELECT anrede, vorname, nachname
2 FROM tuk.personen pe
3 WHERE pe.nachname LIKE 'Wei_'
4 AND pe.vorname = 'Martin'
5 AND pe.anrede = 'Herr';
```

Below the editor is the "Explain Plan" tab. It shows the execution plan for the query:

Plan

- SELECT STATEMENT ALL_ROWS
Cost: 3 Bytes: 20 Cardinality: 1
 - 1 INDEX RANGE SCAN INDEX TUK.IDX_NAME
Cost: 3 Bytes: 20 Cardinality: 1

At the bottom of the explain plan, it states: "2. Rows were returned by the SELECT statement."

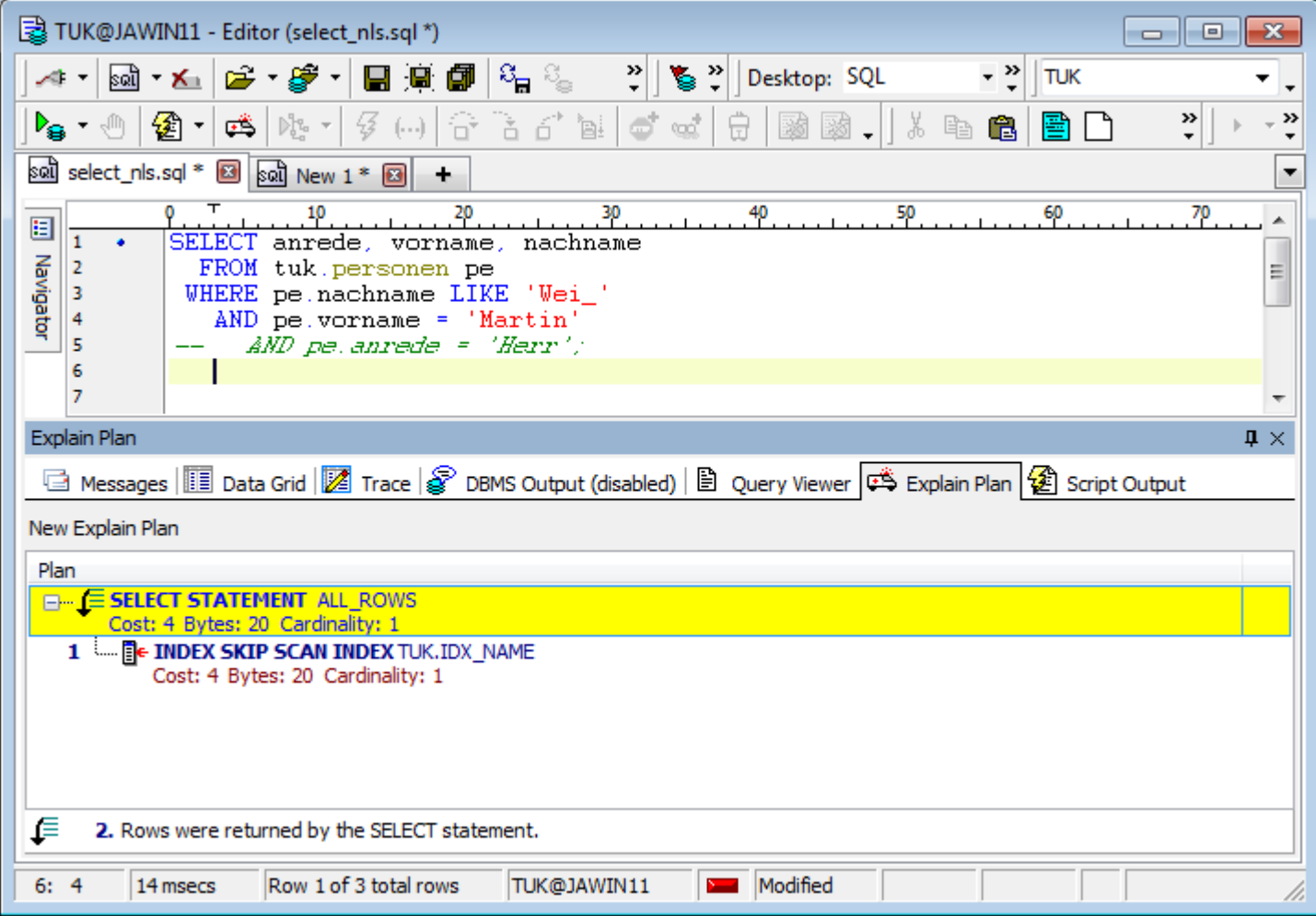
The status bar at the bottom of the IDE shows: "5: 27 14 msec Row 1 of 3 total rows TUK@JAWIN11 Modified".

➤ Abfrage 2:

```
SELECT anrede, vorname, nachname  
  FROM tuk.personen pe  
 WHERE pe.nachname LIKE 'Wei_'  
       AND pe.vorname = 'Martin'  
--    AND pe.anrede = 'Herr';
```

1b_concat_index.bat

Index Skip Scan



The screenshot shows a SQL IDE window titled "TUK@JAWIN11 - Editor (select_nls.sql *)". The editor contains the following SQL query:

```
1 SELECT anrede, vorname, nachname
2 FROM tuk.personen pe
3 WHERE pe.nachname LIKE 'Wei_'
4 AND pe.vorname = 'Martin'
5 -- AND pe.anrede = 'Herr';
6
7
```

Below the editor is the "Explain Plan" tab. The "New Explain Plan" section shows the following plan:

Plan
SELECT STATEMENT ALL_ROWS Cost: 4 Bytes: 20 Cardinality: 1
1 INDEX SKIP SCAN INDEX TUK.IDX_NAME Cost: 4 Bytes: 20 Cardinality: 1
2. Rows were returned by the SELECT statement.

The status bar at the bottom indicates: 6: 4 | 14 msec | Row 1 of 3 total rows | TUK@JAWIN11 | Modified

Index Benutzung oder nicht?

```
SQL> SELECT anrede, vorname, nachname  
2      FROM tuk.personen pe  
3      WHERE pe.nachname LIKE 'wei_'  
4      AND pe.vorname LIKE 'Martin';
```

ANRED	VORNAME	NACHNAME
Herr	Martin	Weiß
Herr	Martin	Weis
Herr	Martin	Weiz

3 Zeilen ausgewählt.

Warum kein Index?

```
PLAN_TABLE_OUTPUT
```

```
-----  
Plan hash value: 1826680655
```

```
-----  
| Id  | Operation                      | Name      |  
-----  
|  0  | SELECT STATEMENT                |           |  
|  1  |   SORT ORDER BY                 |           |  
|  2  |     TABLE ACCESS FULL          | PERSONEN  |  
-----
```

... oder doch ein Index?

The screenshot shows the Oracle SQL Developer interface. The main window is titled 'TUK@JAWIN11 - Editor (2b_abfrage_ls.sql)'. It contains a SQL query in a text editor:

```
1 SELECT persid, anrede, vorname, nachname
2 FROM tuk.personen pe
3 WHERE pe.nachname LIKE 'wei_'
4 AND pe.vorname LIKE 'Martin'
5 ORDER BY pe.nachname, pe.vorname;
```

Below the editor is the 'Explain Plan' tab. It shows the 'Cached Explain Plan' for the query. The plan is as follows:

- Plan
SELECT STATEMENT ALL_ROWS
Cost: 6
3 SORT ORDER BY
Cost: 6 Bytes: 25 Cardinality: 1
2 TABLE ACCESS BY INDEX ROWID TABLE TUK.PERSONEN
Cost: 5 Bytes: 25 Cardinality: 1
1 INDEX SKIP SCAN INDEX TUK.IDX_NAME
Cost: 4 Cardinality: 1

At the bottom of the plan, it states: '4. Rows were returned by the SELECT statement.'

The status bar at the bottom of the window shows: '5: 35 14 msecs Row 0 of 0 total rows TUK@JAWIN11 No'.

... den hatten wir doch schon einmal!

The screenshot shows a SQL editor window titled "TUK@JAWIN11 - Editor (select_nls.sql *)". The editor contains the following SQL query:

```
1 SELECT anrede, vorname, nachname
2 FROM tuk.personen pe
3 WHERE pe.nachname LIKE 'Wei_'
4 AND pe.vorname = 'Martin'
5 -- AND pe.anrede = 'Herr';
6
7
```

Below the editor is the "Explain Plan" section. It includes a toolbar with "Messages", "Data Grid", "Trace", "DBMS Output (disabled)", "Query Viewer", "Explain Plan", and "Script Output". The "New Explain Plan" section shows the following plan:

```
Plan
1 SELECT STATEMENT ALL_ROWS
  Cost: 4 Bytes: 20 Cardinality: 1
  1 INDEX SKIP SCAN INDEX TUK.IDX_NAME
    Cost: 4 Bytes: 20 Cardinality: 1
  2 Rows were returned by the SELECT statement.
```

The status bar at the bottom indicates "6: 4", "14 msec", "Row 1 of 3 total rows", "TUK@JAWIN11", and "Modified".

➤ Warum Full-Table-Scan?

Program	Machine	OSUser	SerialNum	Server	Status
SPOTLIGHT.EXE					
SQLPLUS.EXE					
TUK	CARAJANDB\CARAJANDB01	CARAJANDB01\Joh	283	DEDICATED	INACTIVE
TOAD.EXE					
SYSTEM	CARAJANDB\CARAJANDB01	CARAJANDB01\Joh	121	DEDICATED	ACTIVE
SYSTEM	CARAJANDB\CARAJANDB01	CARAJANDB01\Joh	77	DEDICATED	INACTIVE
TUK	CARAJANDB\CARAJANDB01	CARAJANDB01\Joh	423	DEDICATED	INACTIVE
TUK	CARAJANDB\CARAJANDB01	CARAJANDB01\Joh	33	DEDICATED	INACTIVE
TUK	CARAJANDB\CARAJANDB01	CARAJANDB01\Joh	105	DEDICATED	INACTIVE

Session	Process	IO	Waits	Current Statement	Open Cursors	Access	Locks	RBS Usage	Long Ops	Statistics
---------	---------	----	-------	-------------------	--------------	--------	-------	-----------	----------	------------

Cached Explain Plan

Plan

SELECT STATEMENT ALL_ROWS
Cost: 274

1 **TABLE ACCESS FULL TABLE TUK.PERSONEN**
Cost: 274 Bytes: 20 Cardinality: 1

➤ Versuch einer Analyse:

```
SQL> SELECT anrede, vorname, nachname  
2      FROM tuk.personen pe  
3      WHERE pe.nachname LIKE 'wei_'  
4      AND pe.vorname LIKE 'Martin';
```

ANRED	VORNAME	NACHNAME
Herr	Martin	Weiß
Herr	Martin	Weis
Herr	Martin	Weiz

3 Zeilen ausgewählt.

➤ Versuch einer Analyse:

PLAN_TABLE_OUTPUT

Plan hash value: 1826680655

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time	

0	SELECT STATEMENT		1	25	275 (2)	00:00:04	
1	SORT ORDER BY		1	25	275 (2)	00:00:04	
* 2	TABLE ACCESS FULL	PERSONEN	1	25	274 (1)	00:00:04	

Predicate Information (identified by operation id):

2 - filter("PE"."NACHNAME" LIKE 'wei_' AND
NLSSORT("PE"."VORNAME",'nls_sort='BINARY_CI') =HEXTORAW('6D617274696E00'))

- **ALTER SESSION SET nls_sort=binary_ci;**
 - Sortierung ist unabhängig von Groß- / Kleinschreibung aber abhängig von Akzenten
 - Alternativen:
 - binary_ai → Case und Akzent insensitiv
 - german_ai → Deutsche Sortierung, Case und Akzent insensitiv
 - ...
- **ALTER SESSION SET nls_comp=linguistic;**
 - Filter verwenden die gleiche Funktion wie NLS_SORT, d.h. in diesem Fall ist die WHERE-Clausel unabhängig von Groß- / Kleinschreibung und von Akzenten
 - Alternativen:
 - BINARY oder ANSI

- Entweder Linguistische Suche ausschalten
- ... oder Index auf Linguistische Suche

```
CREATE INDEX idx_name2  
  ON personen (NLSSORT (vorname, 'nls_sort=binary_ci'),  
              NLSSORT (nachname, 'nls_sort=binary_ci'));
```

- Aber jetzt sind z.B. keine Bitmapped Indizes auf den Spalten möglich

Index oder nicht?

```
SELECT anrede, vorname, nachname, geburtstag  
FROM tuk.personen pe  
WHERE pe.nachname like 'Weisse%'  
and pe.anrede = 'Herr';
```

PLAN_TABLE_OUTPUT

Plan hash value: 1706350694

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time
0	SELECT STATEMENT		26	728	405 (1)	00:00:05
1	TABLE ACCESS BY INDEX ROWID	PERSONEN	26	728	405 (1)	00:00:05
* 2	INDEX RANGE SCAN	IDX_NAME	26		379 (1)	00:00:05

Predicate Information (identified by operation id):

```
2 - access("PE"."ANREDE"='Herr' AND "PE"."NACHNAME" LIKE 'Weisse%')
    filter("PE"."NACHNAME" LIKE 'Weisse%')
```

... oder doch nicht?

PLAN_TABLE_OUTPUT

Plan hash value: 2673218719

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time
0	SELECT STATEMENT		26	728	274 (1)	00:00:04
* 1	TABLE ACCESS FULL	PERSONEN	26	728	274 (1)	00:00:04

Predicate Information (identified by operation id):

1 - filter("PE"."ANREDE"='Herr' AND "PE"."NACHNAME" LIKE 'Weisse%')

➤ **db_file_multiblock_read_count**

- Anzahl Blöcke die mit einem I/O beim table oder index scan gelesen werden
- Default 128 (ev. 64)
- Je höher der Wert umso wahrscheinlicher ein Full-Table-Scan

➤ **optimizer_index_cost_adj**

- Verhältnis zwischen Full-Table-Scan und Index Benutzung
- Default 100
- Je geringer der Wert umso wahrscheinlicher ein Index Zugriff
- Schlecht bei hohen Clustering Faktoren!

Index oder nicht?

```
SELECT p.persid,  
       p.vorname,  
       p.nachname,  
       m.email  
FROM   personen p, mail m  
WHERE  p.persid = m.persid  
       AND p.persid = :l_persid;
```

- **Beide Tabellen persid als Primary Key!**

Warum kein Index Zugriff?

TUK3@JAWIN11 - Editor (New 1 *)

Desktop: SQL

extended_stat1.sql | umlaute_tabellenname.sql * | New 1 * | create_mailtable.sql

```
1 SELECT p.persid,  
2       p.vorname,  
3       p.nachname,  
4       m.email  
5 FROM personen p, mail m  
6 WHERE p.persid = m.persid  
7       AND p.persid = :l_persid;
```

Explain Plan

Messages | Data Grid | Trace | DBMS Output (disabled) | Query Viewer | Explain Plan | Script Output

New Explain Plan

Plan

- SELECT STATEMENT ALL_ROWS
Cost: 21 Bytes: 54 Cardinality: 1
 - 4 NESTED LOOPS
Cost: 21 Bytes: 54 Cardinality: 1
 - 2 TABLE ACCESS BY INDEX ROWID TABLE TUK3.PERSONEN
Cost: 2 Bytes: 20 Cardinality: 1
 - 1 INDEX UNIQUE SCAN INDEX (UNIQUE) TUK3.PK_PERSONEN
Cost: 1 Cardinality: 1
 - 3 TABLE ACCESS FULL TABLE TUK3.MAIL
Cost: 19 Bytes: 34 Cardinality: 1

5. Rows were returned by the SELECT statement.

5: 24 | 79 msec | TUK3@JAWIN11 | Modified

Warum kein Index Zugriff?

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time
0	SELECT STATEMENT		1	54	21 (0)	00:00:01
1	NESTED LOOPS		1	54	21 (0)	00:00:01
2	TABLE ACCESS BY INDEX ROWID	PERSONEN	1	20	2 (0)	00:00:01
* 3	INDEX UNIQUE SCAN	PK_PERSONEN	1		1 (0)	00:00:01
* 4	TABLE ACCESS FULL	MAIL	1	34	19 (0)	00:00:01

Predicate Information (identified by operation id):

3 - access("P"."PERSID"=TO_NUMBER(:L_PERSID))
4 - filter(TO_NUMBER("M"."PERSID")=TO_NUMBER(:L_PERSID))



```
CREATE TABLE MAIL  
(  
    PERSID          VARCHAR2 (10 BYTE) ,  
    EMAIL            VARCHAR2 (50 BYTE) ,  
    BEMERKUNG        VARCHAR2 (200 BYTE)  
)  
TABLESPACE USERS;
```

```
CREATE INDEX idx_ort  
  ON adressen (plz,ort);
```

```
CREATE INDEX idx_name  
  ON personen(nachname,vorname);
```

```
SELECT pe.anrede, pe.vorname, pe.nachname,  
       ad.plz, ad.ort, ad.strasse  
  FROM adressen ad, personen pe  
 WHERE pe.persid = ad.persid  
       AND pe.nachname = :nachname  
       AND ad.plz = :plz  
       AND ad.ort = :ort;
```

Ausführungsplan 1

Ausführungsplan

Plan hash value: 2938594640

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time
0	SELECT STATEMENT		2	140	5 (0)	00:00:
1	NESTED LOOPS					
2	NESTED LOOPS		2	140	5 (0)	00:00:
3	TABLE ACCESS BY INDEX ROWID	ADRESSEN	2	76	3 (0)	00:00:
* 4	INDEX RANGE SCAN	IDX_ORT	2		1 (0)	00:00:
* 5	INDEX UNIQUE SCAN	PK_PERSONEN	1		0 (0)	00:00:
* 6	TABLE ACCESS BY INDEX ROWID	PERSONEN	1	32	1 (0)	00:00:

Predicate Information (identified by operation id):

- 4 - access("AD"."PLZ"=:L_PLZ AND "AD"."ORT"=:L_ORT)
- 5 - access("PE"."PERSID"="AD"."PERSID")
- 6 - filter("PE"."NACHNAME"=:L_NACHNAME)

Statistiken

```
1 recursive calls
0 db block gets
474 consistent gets
1 physical reads
0 redo size
922 bytes sent via SQL*Net to client
524 bytes received via SQL*Net from client
2 SQL*Net roundtrips to/from client
0 sorts (memory)
0 sorts (disk)
1 rows processed
```

➤ Erstellen von erweiterten Statistiken

```
SELECT dbms_stats.create_extended_stats (
    OWNNAME      => 'TUK3',
    TABNAME      => 'ADRESSEN',
    EXTENSION    => ' (PLZ,ORT) ' )
FROM DUAL;

BEGIN
    dbms_stats.gather_table_stats (
        OWNNAME      => 'TUK3',
        TABNAME      => 'ADRESSEN'
        ,ESTIMATE_PERCENT => 100
        ,METHOD_OPT    => 'FOR ALL COLUMNS SIZE SKEWONLY'
        ,DEGREE        => NULL
        ,CASCADE        => TRUE
        ,NO_INVALIDATE  => FALSE) ;
END;
```

Ausführungsplan 2

Ausführungsplan

Plan hash value: 3657448784

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time
0	SELECT STATEMENT		4	280	75 (2)	00:00:01
* 1	HASH JOIN		4	280	75 (2)	00:00:01
2	TABLE ACCESS BY INDEX ROWID	PERSONEN	4	128	6 (0)	00:00:01
* 3	INDEX RANGE SCAN	IDX_NAME	4		2 (0)	00:00:01
* 4	TABLE ACCESS FULL	ADRESSEN	206	7828	68 (0)	00:00:01

Predicate Information (identified by operation id):

- 1 - access("PE"."PERSID"="AD"."PERSID")
- 3 - access("PE"."NACHNAME"=:L_NACHNAME)
- 4 - filter("AD"."PLZ"=:L_PLZ AND "AD"."ORT"=:L_ORT)

Statistiken

```
      8 recursive calls
      0 db block gets
196 consistent gets
0 physical reads
      0 redo size
    922 bytes sent via SQL*Net to client
    524 bytes received via SQL*Net from client
      2 SQL*Net roundtrips to/from client
      0 sorts (memory)
      0 sorts (disk)
      1 rows processed
```


➤ Löschen von erweiterten Statistiken

```
BEGIN
  dbms_stats.drop_extended_stats (
    OWNNAME    => 'TUK2',
    TABNAME     => 'ADRESSEN',
    EXTENSION  => '(PLZ,ORT)');
END;
```

Delete eines Satzes

```
SQL> DELETE FROM mitarbeiter  
      2  WHERE mit_id=102075;
```

1 Zeile wurde gelöscht.

Plan hash value: 3730998038

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time		

0	DELETE STATEMENT		1	26	1 (0)	00:00:01		
1	DELETE	MITARBEITER						
* 2	INDEX UNIQUE SCAN	PK_MIT_ID	1	26	1 (0)	00:00:01		

Predicate Information (identified by operation id):

2 - access("MIT_ID"=102075)

... und die Statistiken?

Statistiken

14	recursive calls
10	db block gets
1197	consistent gets
0	physical reads
816	redo size
839	bytes sent via SQL*Net to client
800	bytes received via SQL*Net from client
3	SQL*Net roundtrips to/from client
1	sorts (memory)
0	sorts (disk)
1	rows processed

- Kein Index:
 - Table Lock auf die Detail Tabelle, wenn Update auf Primary Key der Master Tabelle
- Updates auf den Primary Key sind selten
 - ➔ Index auf Foreign Key nicht erforderlich

Aber:

- DELETE FROM <Master Tabelle>
 - ➔ FULL TABLE SCAN auf die <Detail Tabelle>

Index auf Foreign Key

Spotlight on Oracle - ORAP orap1

File View Favorites Tools Help

ORAP orap1

SQL - [Full Screen]

SQL Details SQL Plan

SQL text:

```
/* delete */  
DELETE FROM Partner  
WHERE id = :1 AND version = :2
```

Live connections

ORAP

ORAP ora...

SQL Time Breakdown (Last refreshed at 13:36:13)

Category	Value	Per Exec	Per Row	Pct Elapsed
Execution Statistics				
Parse calls	380,00	1,00	1,00	
Executions	379,00	1,00	1,00	
Fetches	0,00	0,00	0,00	
Rows	379,00	1,00	1,00	
Buffer Gets	48.945.568,00	129.143,98	129.143,98	
Direct Writes	0,00	0,00	0,00	
Disk Reads	1.411.728,00	3.724,88	3.724,88	
Sorts	0,00	0,00	0,00	
Time breakdown (ms)				
CPU Time	512.970,00	1.353,48	1.353,48	32,72
Cluster Time	108,07	0,29	0,29	0,01
Concurrency Time	192,18	0,51	0,51	0,01
Application Time	63.340,10	167,12	167,12	4,04
IO Time	579.693,26	1.529,53	1.529,53	36,98
PLSQL Time	0,00	0,00	0,00	0,00
Java Time	0,00	0,00	0,00	0,00
Other Time	411.374,09	1.085,42	1.085,42	26,24
Elapsed Time	1.567.677,70	4.136,35	4.136,35	100,00

Time Breakdown (ms)

Time Breakdown (ms)

I/O

CPU

Other

Application

Concurrency

Cluster

Java

PL/SQL

ms

Zusammengesetzter Index

TUK@JAWIN11 - Schema Browser (TUK.PERSONEN)

PERSONEN: Created: 09.13.2012 12:27:45 Last DDL: 02.23.2013 12:53:36
Primary Key: PERSID

Used By: Columns, Indexes, Constraints, Triggers, Data, Script, Policies: Grants, Synonyms, Partitions, Subpartitions, Auditing: Stats/Size, Referential

☐ Include Size Info with Index Stats

Img	Index Name	Unique	Logging	Degree	Column Name	Order	Position	Index Owner
▶	IDX_NAME	N	YES	1	ANREDE	Asc	1	TUK
	IDX_NAME	N	YES	1	VORNAME	Asc	2	TUK
	IDX_NAME	N	YES	1	NACHNAME	Asc	3	TUK
	PK_PERSONEN	Y	YES	1	PERSID	Asc	1	TUK
▲	SYS_IL0000103570C00006\$\$	Y	YES	0	BILD	N/A	1	TUK

Parameter Value

Index Name	IDX_NAME
Index Type	NORMAL
Uniqueness	NONUNIQUE
Status	VALID
Table	TUK.PERSONEN
Table Type	TABLE
Tablespace	USERS
Buffer Pool	DEFAULT
Partitioned	No
Temporary	No
Initial Transactions	2
Max Transactions	255
Initial Extent Size	64 KB
Next Extent Size	1 MB
Minimum Extents	1
Maximum Extents	2,147,483,645
Percent Free	10
Degree	1

9_concat_iindex.bat

... und trotzdem Full Table Scan!?

The screenshot shows the Oracle SQL Developer interface. The main window is titled 'TUK@JAWIN11 - Editor (9_create_index.sql)'. It contains a SQL editor with the following query:

```
7 SELECT vorname, nachname
8 FROM tuk.personen pe
9 WHERE pe.nachname LIKE 'Wei_'
10 AND pe.vorname = 'Martin'
11 AND pe.anrede = 'Herr';
```

Below the editor is the 'Explain Plan' tab. It shows the execution plan for the query:

```
Plan
1 SELECT STATEMENT ALL_ROWS
  Cost: 274 Bytes: 20 Cardinality: 1
  1 TABLE ACCESS FULL TABLE TUK.PERSONEN
    Cost: 274 Bytes: 20 Cardinality: 1
```

The status bar at the bottom indicates '11: 31' and '210 msec'.

- Erstellen von Indizes zum Testen
 1. Test, ob Index tatsächlich genutzt wird
→ CREATE INDEX ... INVISIBLE
 2. Test, ob Index notwendig ist
→ ALTER INDEX ... INVISIBLE

- Was passiert, wenn ein Wert nicht gleich verteilt ist?
- Z.B. AUFTRAGSSTATUS
 - Geliefert hoffentlich über 90%
 - Storniert hoffentlich nur 1 – 2 %

```
SELECT ...  
  FROM personen p, auftraege a  
 WHERE p.persid = a.persid  
       AND a.aufstatus = 'G';
```

- Wenn l_aufstatus = 'G' ist, dann besser Full-Tables Scan
- Wenn l_aufstatus = 'S' ist, dann besser Index Range Scan

- Auch mehrere Indizes können gleichzeitig verwendet werden → Bitmap conversion
- Bei zusammengesetzten Indizes Reihenfolge der Spalten beachten (trotz „Skip Scan“)
- Funktionen setzen Indizes außer Kraft (z.B: NLS Parameter)
- Vorsicht bei der Annahme eine „ID“ oder gar ein „NR“ sei eine Nummer!
- Indizes auf Foreign Keys sind durchaus sinnvoll (DELETE)
- Invisible Indizes helfen nicht nur beim Anlegen sondern auch beim Löschen von Indizes
- Erweiterte Statistiken sind hilfreich
- Histogramme nutzen aber cursor_sharing=FORCE vermeiden
- ... und immer die Systemparameter und –statistiken beachten!

Wenn Sie Hilfe brauchen →

- Experten mit über 30 Jahren Oracle Erfahrung
- Spezialisten für
 - Backup & Recovery
 - Hochverfügbarkeit
 - Healthchecks
 - Performance Optimierung
 - Einsatz von Oracle Standard Edition
 - Oracle in virtuellen Umgebungen und in der Cloud
 - Oracle Migrationen (HW, Unicode, Konsolidierung, Standard Edition)
 - Monitoring (Grid / Cloud Control, HLMM, Foglight, Spotlight)
- Schulung und Workshops (Oracle, Toad)



Oracle Standard Edition

Funktion	Standard Edition	Enterprise Edition
Data Guard	NEIN	JA
Active Data Guard	NEIN	Option
Online Table und Index Rebuild	NEIN	JA
Parallel DML and DDL	NEIN	JA
Flashback Query	JA	JA
Flashback Table, Database, Transaction Query	NEIN	JA
Flashback Data Archive (Total Recall)	NEIN	Option
Streams (inklusive Capture)	NEIN	JA
Online und Incremental Backup and Recovery	JA	JA
Advanced Compression	NEIN	Option
Bitmapped Index und Bitmapped Join Index	NEIN	JA
Oracle Real Application Clusters	JA	Option
Partitioning	NEIN	Option
Transportable Tablespaces	NEIN	JA
AWR, ADDM, ASH	NEIN	Option

Vollständige Liste: [Oracle Database Licensing Information 11g Release 2 \(11.2\) Part Number E10594-18](#)

➤ Zitat DOAG Lizenzierung Competence Center:

- Der gesamte Cluster muss lizenziert werden! Dies ist eine grundsätzliche Regel bei Softpartitionierung, der VMWare und HyperV zugerechnet werden. Die Art der Automatisierung ist hierbei für die Lizenzierung unerheblich. Sofern in dem Cluster kein Server über mehr als zwei Prozessorsockel verfügt, kann – wenn es von der Funktionalität her reicht – die DB SE1 lizenziert werden. Dies muss allerdings für alle bestückten Prozessorsockel geschehen.

<http://www.doag.org/doag/competence-center/lizenz/fragen-und-antworten.html>



Fragen?

Johannes Ahrends
www.carajandb.com

Johannes.ahrends@carajandb.com



ORACLE
ACE

DOAG
Deutsche ORACLE-Anwendergruppe e.V.

