



DOAG EDITION PRAXISWISSEN

Oracle Tuning in der Praxis

Rezepte und Anleitungen für
Datenbankadministratoren und -entwickler

2. Auflage



Frank Haas

HANSER



Inhalt

Vorwort	XI
1 Oracle-Design	1
1.1 Datenhaltung in Oracle	1
1.1.1 Tabellen	1
1.1.2 Referentielle Integrität	6
1.1.3 Trigger	7
1.1.4 Views	10
1.1.5 Partitionierung	12
1.1.6 Cluster.....	14
1.1.7 Datentypen.....	17
1.1.8 Grundsätze für effektives Tabellendesign.....	20
1.2 Zugriffshilfen	21
1.2.1 Indizes.....	21
1.2.2 Index-Organisierte Tabellen (IOTs).....	23
1.2.3 Sequenzen.....	25
1.2.4 Einsatz von Indizes und Sequenzen	27
1.3 Statistiken	28
1.4 Der Zugriff auf Oracle.....	29
1.4.1 SQL*Net.....	30
1.5 SQL	37
1.5.1 Shared SQL.....	37
1.5.2 Hints, Outlines und SQL-Profile.....	38
1.5.3 Lesende Operationen	38
1.5.4 Schreibende Operationen	44
1.6 PL/SQL	52
2 Oracle-Tuning	57
2.1 Die drei Phasen einer SQL-Anweisung.....	57
2.2 Der Ausführungsplan.....	59
2.3 Der Oracle Optimizer	62
2.3.1 Der RULE-based Optimizer (RBO).....	62

2.3.2	Costbased Optimizer (CBO).....	63
2.3.3	Einstellungen für den Optimizer.....	65
2.4	Statistiken im Detail.....	68
2.4.1	Histogramme.....	75
2.4.2	Wann und wie oft soll man die Statistiken erstellen?.....	78
2.5	Row Sources.....	79
3	Das ABC des Tuning.....	95
3.1	Ratios oder Wait Interface?.....	95
3.2	Statistische Kennzahlen.....	96
3.3	V\$WAITSTAT.....	102
3.4	Ratios.....	103
3.5	Wait Events.....	105
4	Vorgehensweisen beim Tuning.....	113
4.1	Ansätze beim Tuning.....	113
4.2	Generelle Performance-Untersuchung.....	114
4.3	Spezifische Performance-Untersuchung.....	118
4.4	Wann und wo setzen Sie die verschiedenen Methoden ein?.....	124
5	Performance Tracing und Utilities.....	127
5.1	Utilities.....	127
5.2	Tuning in 10g mit den Advisories.....	129
5.3	EXPLAIN PLAN.....	135
5.4	SQL_TRACE.....	139
5.5	TKPROF.....	141
5.6	Event 10046.....	147
5.7	DBMS_MONITOR.....	148
5.8	Event 10053.....	149
5.9	AWR, ASH, Statspack und Bstat/Estat.....	152
5.10	Das Tracing von PL/SQL – DBMS_PROFILER.....	160
5.11	Performance und SQL*Net.....	162
5.11.1	SQL*Net Tracing.....	162
5.11.2	Event 10079.....	163
5.11.3	Trace Assistant.....	163
5.11.4	Trcsess Utility.....	165
5.12	Tuning mit dem Enterprise Manager.....	166
6	Physikalische Strukturen.....	173
6.1	Einleitung.....	173
6.2	Oracle im Hauptspeicher.....	173
6.3	Oracle-Systembereiche.....	182
6.4	Platzverwaltung in Tablespaces.....	196
6.5	Oracle-Systembereiche im Detail.....	198
6.6	Platzverwaltung im Segment.....	209

7	Oracle wird parallel.....	215
7.1	Parallel Query.....	215
7.2	Hauptspeicherbedarf beim Einsatz von Parallel Query	222
7.3	Parallel DML (PDML) und parallel DDL (PDDL)	225
7.4	Statistiken für Parallel Query	228
7.5	Parallele Ausführungspläne.....	231
7.6	Parallelisierung und Partitionierung	232
7.7	Parallel Tracing	234
7.8	Parallele Wait Events	235
7.9	Einsatz und Tuning paralleler Operationen	236
8	Hints, Outlines und SQL-Profile	239
8.1	Hints	239
8.1.1	Hints, die den Optimizer steuern.....	241
8.1.2	Hints für Zugriffspfade	242
8.1.3	Hints für die Transformierung von SQL-Anweisungen	244
8.1.4	Hints für Query Rewrite	245
8.1.5	Hints für die Star Transformation	246
8.1.6	Hints für Joins.....	247
8.1.7	Hints für spezielle Operationen.....	248
8.1.8	Hints für die parallele Ausführung.....	250
8.1.9	Hints und Views	251
8.2	Outlines: Stabile Optimizer-Pläne	252
8.3	SQL-Profile	256
9	Tuning über Parameter	259
9.1	Die Oracle-Parameter	259
9.2	Ausgewählte Parameter	262
10	Spezifische Einstellungen.....	275
10.1	Tuning in hochverfügbaren Umgebungen	275
10.1.1	Was ist Hochverfügbarkeit?.....	275
10.1.2	Wie viele Rechner?.....	277
10.1.3	Anforderungen an die Hardware.....	278
10.1.4	Datenbanktypen in Hochverfügbarkeitsumgebungen	280
10.1.5	Backup und Recovery.....	282
10.1.6	Applikatorische Anforderungen.....	282
10.2	Spezifische Einstellungen für das Betriebssystem.....	283
10.2.1	I/O.....	283
10.2.2	Betriebssysteme	285
11	Glossar	287
	Literatur	289
	Register.....	291



1 Oracle-Design

Thema des ersten Kapitels ist, welche Möglichkeiten Oracle Ihnen für ein Design, das eine gute Performance gewährleistet, zur Verfügung stellt.

Performance ist idealerweise nichts, was man nachträglich auf die Datenbank aufpfropft, sondern sie sollte von Anfang an im Design berücksichtigt werden. Das setzt natürlich voraus, dass Sie bereits beim Design der Applikation mit den Möglichkeiten, die das jeweilige Datenbanksystem anbietet, vertraut sind.

Unabhängig davon, welchen Typ von Objekten Sie verwenden – wobei es sich zum großen Teil um Tabellen und Indizes handeln wird, müssen Sie diese Objekte physikalisch abspeichern. Mit der physikalischen Organisation treffen Sie auch Entscheidungen, die sich auf die Performance auswirken. Dieses Thema wird noch in einem separaten Kapitel im Detail besprochen.

Während sich der Rest des Buches vornehmlich der Untersuchung und Analyse bestehender Applikationen widmet, behandelt dieses Kapitel, wie Sie Applikationen bauen, die bereits im Design Performance berücksichtigen. Dabei liegt der Schwerpunkt auf der Datenbank und SQL. Gleichzeitig ist es wohl auch das Kapitel des Buches, das sich am stärksten an den Entwickler richtet.

1.1 Datenhaltung in Oracle

1.1.1 Tabellen

Für die Speicherung der applikatorischen Daten bietet Oracle verschiedene Möglichkeiten an. Im Regelfall werden Sie ganz normale relationale Tabellen verwenden. Es existieren zwar auch Objekttypen, aber diese sollten Sie nicht zur Speicherung verwenden, weil dort zuviel hinter der Bühne passiert. Wenn Sie beispielsweise eine Nested Table oder eine Object Table anlegen, dann generiert Oracle im Hintergrund versteckte Spalten, die Sie der Tabelle erst mal nicht ansehen. Für die Applikationsentwicklung sind diese Typen exzel-

lent geeignet, nicht aber für die Persistenz. Im Regelfall sollten Sie ganz normale relationale Tabellen verwenden. Die einfache CREATE TABLE-Anweisung erzeugt eine relationale Tabelle, genauer gesagt, eine Heap-Tabelle. Ein Heap ist in der konventionellen Programmierung eine bekannte Struktur. Eine Heap-Tabelle hat keine eingebaute Reihenfolge der Einträge, letztere können dynamisch hinzugefügt und gelöscht werden. 95% oder mehr der Tabellen in Ihrer Applikation werden solche Tabellen sein. Theoretisch könnten Sie auch beim Anlegen der Tabelle angeben, dass es sich um eine Heap-Tabelle handelt: CREATE TABLE .. ORGANIZATION HEAP. Aber da das ohnehin die Voreinstellung ist, macht (und braucht) das niemand. Weitere Details finden Sie in der offiziellen Oracle-Dokumentation in [OraCon 2005] und [OraSql 2005].

Temporäre Tabellen

Es gibt auch temporäre Tabellen, die zur Speicherung von Zwischenresultaten verwendet werden können, diese werden mit CREATE GLOBAL TEMPORARY TABLE erzeugt. Diese Tabelle ist dann für alle Sessions, also global innerhalb der Datenbank, sichtbar. Im Bereich Performance ist der Unterschied zwischen temporären und regulären Tabellen aber oft nicht so signifikant. Betrachten Sie temporäre Tabellen also vor allem als applikatorische Möglichkeit, nicht als Mittel zur Steigerung der Performance. Zwar generiert eine temporäre Tabelle kein Redo. Das ist gut für die Performance. Bei mehrstufigen Abfragen – also Query A ist Input für Query B und B ist potenziell wieder Input für eine neue Query etc. – sind sie sicher eine sehr naheliegende Lösung. Aber Sie haben – teilweise versionsbedingt – auch keine Statistiken, was wiederum unter Umständen einen schlechten Zugriffsplan verursacht. Allerdings lässt sich dieses Problem dann durch manuelles Setzen der Statistiken über DBMS_STATS oder das dynamische Sammeln der Statistiken (ab 9.2) lösen. Bei einer temporären Tabelle müssen Sie angeben, ob die Daten nur für die Transaktion oder für die ganze Session gültig sein sollen. Dies erfolgt über die ON COMMIT-Klausel: ON COMMIT DELETE ROWS löscht die Daten nach jedem COMMIT, das ist auch die Voreinstellung. Bei ON COMMIT PRESERVE ROWS bleiben die Daten während der ganzen Session erhalten. Temporäre Tabellen können indiziert werden, aber Fremdschlüssel sind nicht erlaubt. Temporäre Tabellen sind immer relationale Tabellen. Temporäre Tabellen können auch nicht parallelisiert werden. Sehr gut geeignet sind temporäre Tabellen für die Speicherung von Zwischenergebnissen oder Abfragen auf die dynamischen Performance Views, die V\$-Views. Die V\$-Views werden uns später noch öfters begegnen. Das sind alle Views, deren Name mit V\$ beginnt. Sie enthalten die aktuellen Betriebsdaten der Datenbank. So erfahren Sie über V\$SESSION, welche Benutzer gerade in der Datenbank angemeldet sind, V\$PROCESS zeigt Ihnen die Prozesse der Datenbank, V\$INSTANCE zeigt Ihnen, seit wann die Datenbank läuft, und so weiter und so fort. Diese V\$-Views sind aber keine „normalen“ Views, sondern Strukturen, die in der Datenbank eingebaut sind. Das kann dazu führen, dass Abfragen auf diese Views, die Sie mit Abfragen auf normale Tabellen joinen, ungültige oder falsche Ergebnisse bringen. Um dieses Problem zu vermeiden, müssen Sie zuerst die Daten der V\$-View in eine Zwischentabelle übertragen.

Externe Tabellen

Externe Tabellen wurden mit Oracle 9i eingeführt. Eine externe Tabelle ist eine Tabelle, die in Form einer Datei auf dem Betriebssystem vorliegt. Die Datei ist dann nur lesbar. Das spart zum einen Platz, weil man die Tabelle dann ja nicht noch auch in die Oracle-Datenbank laden muss. Zum andern spart man auch Zeit, die Zeit für das Laden der Daten entfällt ja auch. In Oracle-Versionen vor 9i können Sie dasselbe mit BFILEs erreichen. Dabei folgt das Format der Dateien den Formaten, wie sie im SQL*Loader verwendet werden:

```
CREATE TABLE dept_external
(deptno number(2), dname varchar2(14), loc varchar2(13))
ORGANIZATION EXTERNAL
(TYPE oracle_loader
DEFAULT DIRECTORY DATA_DIR
ACCESS PARAMETERS (
records delimited by newline character set WE8ISO8859P1
badfile ' dept_external.bad'
logfile 'dept_external.log'
fields terminated by "," optionally enclosed by '"' ldrtrim
reject rows with all null fields (
deptno char(25),
dname char(25),
loc char(25))
LOCATION ('dept_externalctl'))
REJECT LIMIT UNLIMITED;
```

Berücksichtigen Sie externe Tabellen besonders im Data Warehouse-Bereich, wenn verschiedene Daten auf und von unterschiedlichen Systemen miteinander abgeglichen werden müssen.

Seit Version 10g können externe Tabellen auch geschrieben werden, das erfolgt über den ORACLE_DATAPUMP-Treiber. Die entstandene Datei ist aber nicht im ASCII-Format; auch hier ein Beispiel:

```
CREATE TABLE dept_external_unload 0
ORGANIZATION EXTERNAL
(TYPE oracle_datapump
DEFAULT DIRECTORY TEMP_DIR
LOCATION ('dept_external_unload.ctl'))
AS Select * from dept;
```

Primärschlüssel

Ein Primärschlüssel definiert eindeutig jeden Datensatz in einer Tabelle und verhindert somit Duplikate. Sie können als Primärschlüssel eigentlich alles verwenden, was die Row eindeutig definiert. Dabei kann das Schlüsselfeld natürlich vorkommende Werte annehmen – sofern es solche in der Tabelle gibt, oder man nimmt ein eigens erzeugtes Schlüsselfeld. Letzteres ist vorteilhafter, da es die Applikationslogik von der technischen Implementierung abkoppelt. Sie vermeiden dadurch Probleme, falls aus irgendwelchen Gründen das Schlüsselfeld verändert werden muss. Dies sollte zwar in der Theorie nicht vorkommen, tut es in der Praxis dann aber doch. Mit einem technischen Schlüssel ist das und manches andere leichter. Steht die Tabelle in einer hierarchischen Beziehung, sind auch zusammengesetzte Primärschlüssel interessant. Nehmen wir als Beispiel die zwei Tabellen Rech-

nungskopf und Rechnungsdaten. Im Rechnungskopf sind dann die allgemeinen (einmaligen) Daten wie Kunde, Zahlungsziel, Gesamtsumme, Kontoverbindung etc. abgespeichert, während in den Rechnungsdaten die zugeordneten Rechnungsposten abgespeichert werden. Als Schlüssel im Rechnungskopf wird eine fortlaufende eindeutige Rechnungsnummer verwendet. Das Schlüsselfeld in der Tabelle Rechnungsdaten wird dann aus dem Schlüsselfeld im Rechnungskopf und einem weiteren Feld, das den zugeordneten Rechnungsposten eindeutig identifiziert, gebildet.

Generell sollte also jede Tabelle in der Datenbank zumindest einen Primärschlüssel haben. Oracle realisiert den Primärschlüssel dann über einen eindeutigen B*-Baum-Index. Sie sind nicht verpflichtet, einen Primärschlüssel festzulegen. Manchmal scheint es auch nicht unbedingt notwendig. Extra einen Index anlegen für eine Tabelle mit nur 100 Datensätzen? Legen Sie aber trotzdem einen Primärschlüssel an. Das ist eine Investition in die Zukunft. Angenommen, in einem Jahr wird beschlossen, dass noch eine Standby-Datenbank neben der Produktion aufgebaut werden soll. So weit, so schön, aber: Tabellen, die in Standby-Umgebungen repliziert werden sollen, müssen einen Primärschlüssel haben. Bedeutet also konkret, dass Sie Ihre Applikation einem Redesign unterziehen müssen. Das wäre aber nicht notwendig gewesen, wenn die Tabellen gleich von Anfang an mit Primärschlüsseln versehen worden wären. Abgesehen davon geben Sie dem Optimizer damit mehr Informationen, und je mehr Informationen der Optimizer hat, desto bessere Zugriffspläne kann er erzeugen.

Bei künstlichen Schlüsselfeldern empfehlen sich fortlaufende Nummern. Mit Sequenzen und Triggern lassen sich die Schlüsselfelder dann fast automatisch nachführen, das sehen wir später dann noch im Detail. Beherzigen Sie die Empfehlungen, die im Kapitel 6 von [OraAppDev 2005] für Primärschlüssel gegeben werden:

- Verwenden Sie eine Spalte, deren Werte eineindeutig sind. Ein Primärschlüssel dient dazu, eine Zeile eindeutig zu identifizieren.
- Verwenden Sie eine Spalte, deren Wert später nicht mehr verändert wird. Der Primärschlüssel sollte lediglich dazu dienen, den Datensatz eindeutig zu kennzeichnen. Er sollte keine andere (zusätzliche) Bedeutung haben.
- Verwenden Sie eine Spalte, die als NOT NULL deklariert werden kann. NULL-Werte sind in Primärschlüsseln nicht zulässig.
- Verwenden Sie einen numerischen Datentyp, dann können die Werte für den Primärschlüssel über eine Sequenz bereitgestellt werden.
- Vermeiden Sie Primärschlüssel, die aus mehreren Feldern zusammengesetzt werden; die Wartung ist in diesem Fall wesentlich aufwändiger.

Bei der Spezifikation der Primärschlüsselspalte ist NOT NULL nicht zwingend, aber sehr zu empfehlen, wie das folgende Beispiel anschaulich zeigt:

```
SQL> create table pk_example (x number primary key);
Table created.

SQL> insert into pk_example values (1);
1 row created.
```



```

SQL> insert into pk_example values (null);
insert into pk_example values (null)
                                *
ERROR at line 1:
ORA-01400: cannot insert NULL into ("SYSTEM"."PK_EXAMPLE"."X")
SQL> commit;
Commit complete.
SQL> select * from pk_example;
-----X
-----
1

```

Die Implementation des Primärschlüssels erfolgt hier über einen eindeutigen Index. Das ist aber nicht zwingend. Existieren bereits Indizes, schaut Oracle zuerst, ob der Primärschlüssel über diese bestehenden Indizes realisiert werden kann. Diese Indizes müssen nicht einmal eindeutig sein.

Wie Sie hier sehen, können Sie einen NULL-Wert nicht eingeben; also ist es auch nicht sinnvoll die Spalte so zu deklarieren, es gibt ja einen Fehler, wenn Sie einen NULL-Wert einfügen wollen.

Das ist übrigens nicht das Gleiche wie ein eindeutiger Index auf dieser Spalte, wie man hier sieht:

```

SQL> create table unique_example (x number);
Table created.
SQL> create unique index uniq_ix on unique_example (x);
Index created.
SQL> insert into unique_example values (1);
1 row created.
SQL> insert into unique_example values (null);
1 row created.
SQL> commit;
Commit complete.

```

Sie können also in diesem Fall trotz des eindeutigen Index NULL-Werte eingeben. Es kommt aber noch besser, das lässt sich auch (beliebig oft) wiederholen:

```

SQL> insert into unique_example values (null);
1 row created.
SQL> commit;
Commit complete.
SQL> select * from unique_example;
-----X
-----
1

3 rows selected.

```



Register

- ⋮
- :recursive calls 98

-
- _MEMORY_BROKER_SHRINK_HEAPS 182
- _OLD_CONNECT_BY_ENABLED 92
- _OPTIMIZER_MODE_FORCE 65
- _px_trace Event 234
- _SQLEXEC_PROGRESSION_COST 272
- _TRACE_FILES_PUBLIC 260

- A**
- Active Session History 123, 159
 - siehe auch* ASH
- ADDM 129, 287
- AIX 286
- ALL_ROWS 65
- ALL_ROWS (Hint) 241
- analytische Funktionen 39
- ANALYZE 28, 64, 68, 76
- Antijoin 44
- Anzahl Block Header in Abhängigkeit von
 - Blockgröße 198
- Anzahl der Datensätze pro Block festlegen 211
- APPEND (Hint) 248
- ASH 159, 171, 236, 287
- ASM 191
- ASM_DISKSTRING 193
- ASM_POWER_LIMIT 192, 193

- ASMM 181, 287
- ASSM 197, 283, 287
- Asynchronus I/O 285
- Aufbau eines Oracle-Blocks 209
- AUM 204, 288
- AUTOALLOCATE 196
- AUTOEXTEND 191, 217
- Automatic Segment Space Management 197
 - siehe auch* ASSM
- Automatic Shared Memory Management 181
 - siehe auch* ASMM
- Automatic Undo Management 204
 - siehe auch* AUM
- Automatic Workload Repository 157
 - siehe auch* AWR
- AUTOTRACE 59, 137
- Autotuning SGA 181 *siehe auch* ASMM
- AUX_STAT\$ 73
- AVG_ROW_LEN 69, 211
- AWR 111, 157, 288
- AWR-Einstellungen 117, 125, 168

- B**
- B*-Baum Index 21
- Baseline (Enterprise Manager) 167
- Bequeath 33
- Bind Peeking 37
- Bind Variable 37, 178, 263
- Bitmap (Ausführungsplan) 93

Bitmap Index 22
BLEVEL 70
BLOCKS 69
Bstat/Estat 152
Bucket 76
buffer busy waits 145, 227
BUFFER_POOL 176
BUFFER_POOL_KEEP 262
BUFFER_POOL_RECYCLE 263
Bulk Bind 54
Bulk Collect 54

C

Cache (Festplatte) 187
CACHE 175, 176
CACHE (Hint) 249
CHAIN_COUNT 69
Chained Rows 211
Checkpointing 201
CHOOSE 65
CHOOSE (Hint) 241
CHUNK 18
Cluster (Performance) 15
CLUSTERING_FACTOR 70
COMMAND (in V\$SESSION) 122
COMPATIBLE 263
COMPRESS 213
Concatenation 92, 184
CONNECT BY 91, 136
consistent changes 97
consistent gets 97, 99, 100, 103
Consistent Read für LOB-Segment 19
consistent reads 99, 100, 142
COUNT 87 *siehe auch* ROWNUM
cpuspeed 72
CREATE_FILE() 131, 133
CREATE_SQLWKLD() 132
CREATE_STORED_OUTLINES 254
CREATE_TUNING_TASK() 130
Cross Join 80 *siehe auch* kartesisches Produkt
CURSOR_SHARING 263
CURSOR_SHARING_EXACT (Hint) 250
CURSOR_SPACE_FOR_TIME 263

D

DATABASE LINK 31
DATAPUMP 50
Datenkonvertierung (implizit) 19
db block changes 97
db block gets 97
db file parallel write 109
db file scattered read 109, 145
db file sequential read 109, 145
db file single write 109, 145
DB_BLOCK_BUFFERS 174, 264
DB_BLOCK_CHECKING 264
DB_BLOCK_CHECKSUM 264
DB_BLOCK_LRU_LATCHES 264
DB_BLOCK_MAX_DIRTY_TARGET 264
DB_BLOCK_SIZE 265
DB_CACHE_SIZE 174, 181, 265
DB_FILE_DIRECT_IO_COUNT 265
DB_FILE_MULTIBLOCK_READ_COUNT
73, 145, 265
DB_FLASHBACK_RETENTION_TARGET 280
DB_HANDLES_CACHED 272
DB_KEEP_CACHE_SIZE 262
DB_RECOVERY_FILE_DEST 280
DB_RECYCLE_CACHE_SIZE 263
DB_WRITER_PROCESSES 266
DBA_HIST_FILESTATXS 101, 118
DBA_HIST_WR_CONTROL 117
DBMS_ADVISOR 130, 132
DBMS_METADATA 122
DBMS_MONITOR 101
DBMS_PROFILER 160
DBMS_SHARED_POOL 8, 26, 53, 179
DBMS_SPACE_ADMIN 197
DBMS_SQLTUNE 130, 132, 256
DBMS_STATS 28, 64, 77
DBMS_XPLAN 136
DBWR_IO_SLAVES 266
Deadlock 45
Deadlock (Bitmap Index) 22
DEGREE 220
Dictionary Managed Tablespace 45
Direct I/O 285

Direct Load 216
 Direct path read 109
 Direct path write 109
 Direct-Path INSERT 46
 Direct-Path Load (SQL*Loader) 47
 DISABLE RELY NOVALIDATE 48
 DISABLE VALIDATE 47
 DISABLE_OOB 36
 DISK_ASYNCH_IO 266
 DISTINCT 88
 DML_LOCKS 266
 DOP 217, 218
 DRIVING_SITE (Hint) 249
 DROP TABLE (in READ ONLY Tablespace)
 208
 DYNAMIC_SAMPLING (Hint) 250

E

eindeutiger Index 5
 Einschränkungen (parallel DML) 226
 EMPTY_BLOCKS 64, 69
 Enqueue 109, 227
 ENQUEUE_RESOURCES 267
 Equijoin 43, 80
 ETL 216
 Event (Syntax) 119
 Event 10520 55
 execute count 97, 179, 272
 Execute-Phase 58
 EXISTS 44
 externe Tabelle 3

F

FACT (Hint) 246
 FAST_START_IO_TARGET 267
 Fetch-Phase 58
 FILESYSTEMIO_OPTIONS 285
 FILTER 82
 FIRST_ROWS 65
 FIRST_ROWS (Hint) 241
 FIRST_ROWS_n 65
 FIRST_ROWS_n (Hint) 241
 Flashback 280

FORALL 54
 free buffer waits 109, 145
 FREELIST GROUPS 46, 227, 282
 FREELISTS 25, 46, 197, 227
 FTS 38 *siehe auch* Full Table Scan
 FULL (Hint) 242
 Full Table Scan 38, 40, 90
 funktionsbasierte Indizes 23

G

GATHER_STATS_JOB 28, 79
 GET_DDL() 122
 GROUP BY 88

H

Hard Parse 178
 Hash Cluster 15
 Hash Join 81
 HASH_AREA_SIZE 81, 222, 237, 267
 HASH_JOIN_ENABLED 81, 267
 HASH_MULTIBLOCK_IO_COUNT 222
 HASHKEYS 15
 heißer Block 100
 Highwatermark 46, 58
 Hit Ratio 103
 HPUX 286

I

I/O Waits 195
 Idle Event 106, 235
 implizite Datenkonvertierung 40
 INDEX (Hint) 242
 Index Full Scan 40
 Index Komprimierung 22
 Index Monitoring 23
 Index Range Scan 40
 Index Scan 89
 Index Skip Scan 89
 INDEX_COMBINE (Hint) 243
 INDEX_FFS (Hint) 244
 INDEX_HISTOGRAM 78
 INDEX_JOIN (Hint) 244
 INDEX_SS (Hint) 243

init.ora 260
 INITRANS 212, 227
 INLIST ITERATOR 91
 Inner Join 80
 INTERSECT 42, 84
 Intra-Partition Parallelisierung 225
 IPC (SQL*Net) 32

J

JAVA_POOL_SIZE 181, 267
 JOB_QUEUE_INTERVAL 268
 JOB_QUEUE_PROCESSES 268

K

Kardinalität 66
 kartesisches Produkt 42, 80
 KEEP 176
 KEEP() 179

L

langsamer Zugriff auf leere Tabelle 58
 LARGE_POOL_SIZE 181, 268
 Latch free 109
 LEADING (Hint) 247
 Linux 286
 LMT 288
 LOB 17
 Locally Managed Tablespace 45, 196
 log buffer space 204
 log file switch (checkpoint incomplete) 201
 log file switch(archiving needed) 200
 LOG_ARCHIVE_MAX_PROCESSES 200
 LOG_ARCHIVE_START 200
 LOG_BUFFER 200, 268
 LOG_CHECKPOINT_INTERVAL 202, 268
 LOG_CHECKPOINT_TIMEOUT 202, 268
 LOG_PARALLELISM 200, 236
 Logical Reads 97
 LONG 17
 LRU 175

M

MAX_DUMP_FILE_SIZE 139
 maximale Größe der SGA 180
 MAXLOGHISTORY 199
 MAXVALUE (Partitionierung) 12
 mbrc 72, 74
 MERGE (Hint) 245
 MINIMIZE_RECORDS_PER_BLOCK 211
 MINUS 42, 84
 MODIFY_SNAPSHOT_SETTINGS() 117
 mreadtim 72, 74
 MRU 175, 176
 Multiplexing 199, 279
 Multitable INSERT 46
 Multi-Versioning 99

N

NAS 188
 Native Compilation 55
 Natural Join 43, 80
 NESTED LOOPS 83, 144, 247
 Network Attached Storage siehe auch NAS 188
 nichtindizierte Fremdschlüssel 7
 NO_EXPAND (Hint) 245
 NO_FACT (Hint) 246
 NO_INDEX (Hint) 244
 NO_MERGE (Hint) 245
 NO_PARALLEL (Hint) 251
 NO_PARALLEL_INDEX (Hint) 251
 NO_REWRITE (Hint) 246
 NO_STAR_TRANSFORMATION (Hint) 246
 NOAPPEND (Hint) 249
 NOCOPY 55
 NOLOGGING 50, 206
 NOT EXISTS 44
 NOT NULL 53
 NTILE() 78

O

OCIAttrSet() 62
 ON-LOGON-Trigger (SQL_TRACE aktivieren) 119
 OPEN_CURSORS 269
 OPEN_LINKS 269

- opened cursors cumulative 97, 104
OPS 281
OPTIMIZER_DYNAMIC_SAMPLING 67
OPTIMIZER_FEATURES_ENABLE 66, 269
OPTIMIZER_INDEX_ADJ 269
OPTIMIZER_INDEX_CACHING 269
OPTIMIZER_MAX_PERMUTATIONS 58
OPTIMIZER_MODE 269
OPTIMIZER_PERCENT_PARALLEL 270
ORA-1555 205
ORA-4031 264, 272
ORA-12827 222
ORA-14552 130
ORACLE_SID 31
ORADEBUG 121
ORDERED (Hint) 247
Outer Join 43, 81
Outline (privat) 255
OVERFLOW Segment (IOT) 24
- P**
- PARALLEL (Hint) 250
PARALLEL_ADAPTIVE_MULTI_USER
219, 221
PARALLEL_AUTOMATIC_TUNING
225, 270
PARALLEL_ENABLE 53, 225
PARALLEL_INDEX (Hint) 251
PARALLEL_MAX_SERVERS 230, 232, 237
PARALLEL_MIN_PERCENT 221
Parallelisierung (mögliche Operationen) 220
Parallelisierungsgrad 217
parse count (hard) 97
parse count (total) 97, 179
parse time CPU 98
parse time elapsed 98
Parse-Phase 57
Partition Change Tracking (PCT) 131
Partitionsstatistiken 68
PCT 131
PCTFREE 25, 210
PCTTHRESHOLD 24
PCTUSED 25, 46, 197, 210
- PGA_AGGREGATE_TARGET 223, 237, 273
physical reads 98, 103
physical reads direct 81, 98
physical writes 98
physical writes direct 81, 98
Pinging (OPS/RAC) 281
Pinnen von Triggern 8
PLAN_TABLE 135
Platzbedarf für archivierte Redo Logs 203
Primärschlüssel 3
primary key 4
Pstart 233
Pstop 233
- Q**
- QUERY REWRITE 11
QUERY_REWRITE_ENABLED 11, 245, 270
QUICK_TUNE() 130
- R**
- RAC 281
RAID 186
RAID 5 187
RAID 10 186
Raw Device 283
READ ONLY 207
RECOVERY_PARALLELISM 236, 271
RECV_BUF_SIZE 35
RECYCLE 176
Redo Log 198
redo log space requests 98, 204
redo size 98, 268
Redundancy (ASM) 193
referentielle Integrität und Trigger 9
REMOTE 86
Retention (Statistiken) 75
REWRITE (Hint) 245
REWRITE_OR_ERROR (Hint) 246
rollende Views 10
Row Level Locking 44
ROW_LOCKING 226
ROWID 41, 212
Row-Migration 210

- ROWNUM 87
- RULE 65
- RULE (Hint) 241
- S**
- SAME 186
- SAN 188
- Schnappschuss 169
 - siehe auch* Snapshot (AWR)
- SCN 99, 201
- Script
 - Maxima aus TKRPOF-Dateien ermitteln 116
 - TKPROF über alle Dateien 115
 - Zeiten aus TKPROF-Dateien ermitteln 115
- SDU 34, 143, 165
- SELECT FOR UPDATE 41, 45, 87
- Selektivität 66
- Selfjoin 43
- Semijoin 44, 82
- SEND_BUF_SIZE 34
- SEQUENCE() 26
- Sequenz 86, 282
- Sequenz (automatischer Primärschlüsselwert) 8
- session logical reads 98
- session pga memory max 228
- SESSION_CACHED_CURSORS 271
- SESSION_TRACE_ENABLE() 121
- SET_BOOL_PARAM_IN_SESSION() 120
- SET_EV() 120
- SET_INT_PARAM_IN_SESSION() 120
- setAutoCommit 30
- SGA 173, 288
- SGA_MAX_SIZE 181, 271
- SGA_TARGET 181, 271
- Share Row Exclusive Table Lock 7
- Shared SQL 178
- SHARED_POOL_SIZE 177, 181, 271
- SID 288
- SMJ 84 *siehe auch* SORT/MERGE JOIN
- Snapshot (AWR) 169
- Soft Parse 178
- Solaris 285
- SORT AGGREGATE 88
- SORT ORDER BY 87
- SORT_AREA_MULTIBLOCK_READ_COUNT 272
- SORT_AREA_SIZE 237, 272
- SORT_MULTIBLOCK_READ_COUNT 222
- SORT/MERGE JOIN 84, 247
- Sorted Hash Cluster (Performance) 16
- sorts (disk) 98, 272
- sorts (memory) 98
- sorts (rows) 98
- spfile 261
- SQL
 - Anzeige aller Parameter 260
 - ASH für bestimmte SID ausführen 123
 - buffer busy waits Segmente anzeigen 227
 - CACHE Tabellen beim Startup laden 175
 - COMMAND in V\$SESSION uebersetzen 122
 - DDL für Tabelle von der Datenbank erzeugen lassen 122
 - Einstellungen STATISTICS_LEVEL 273
 - Empfehlungen des SQL Access Advisor 133
 - Enqueue Waits anzeigen 227
 - freien Platz im Shared Pool ermitteln 271
 - Highwatermarks (DML Locks) 266
 - Highwatermarks (Enqueue Resources) 267
 - Histogramme in Tabelle ermitteln 78
 - höchste SNAP_ID (AWR) ermitteln 132
 - KEEP Tabellen beim Startup laden 177
 - Objekt aus DBA_EXTENTS ermitteln 145
 - Outline beim LOGON aktivieren 256
 - Pagesize Redo ermitteln 268
 - Parallel Query Session Statistiken 229
 - Parallel Query System Statistiken 229
 - PL/SQL Profiler RUNID ermitteln 161
 - PL/SQL Profiling an-/abschalten 161
 - Session Cursor Cache ermitteln 271
 - SID für eigene Session ermitteln 122
 - SQL Text aus SQL Tuning Set anzeigen 132
 - Trace-Datei (Namen ermitteln) 121
 - Undo Advisor (Grösse bei Migration) 205
- SQL Access Advisor 133
- SQL Tuning Set 132
- SQL Workload 132
- SQL*Loader 47, 216

SQL*Net roundtrips 98
 SQL*Net Tuning 36
 SQL_TRACE 273
 sreadtim 72, 74
 STALE (Statistiken) 71
 Standby-Datenbank 280
 Star Query 42
 STAR_TRANSFORMATION (Hint) 246
 STAR_TRANSFORMATION_ENABLED 273
 STATISTICS_LEVEL 66, 168, 273
 Statistiken
 selbst setzen 74
 transferieren 74
 verdichten 101
 Statistiken (Index) 70
 Statistiken (Tabelle) 69
 STAT\$FILESTATX 118
 STAT\$IDLE_EVENT 235
 STATSPACK 154
 STORAGE 196
 Storage Area Network 188 *siehe auch* SAN
 Striping 184
 SWAP_JOIN_INPUTS() 81
 Systemstatistiken 72

T

table fetch by rowid 98
 table fetch continued row 98
 Table Monitoring 71
 table scans 99
 TCP/IP (SQL*Net) 33
 Template (ASM) 194
 temporäre Tabelle 2
 TIMED_OS_STATISTICS 273
 TIMED_STATISTICS 108, 274
 Tkprof 61
 Trace-Datei (Namensformat) 61
 Trace-Dateien (Speicherort) 139
 TRACFILE_IDENTIFIER 139
 Transportable Tablespace 49
 TTS 288
 TWO_TASK 31

U

uffer busy waits 109
 ufs 285
 UNDO_MANAGEMENT 205
 UNDO_RETENTION 205
 UNDO_SUPPRESS_ERRORS 205
 UNIFORM 196
 UNION 41, 85
 UNION ALL 85
 unique index 5
 USE_CONCAT (Hint) 245
 USE_HASH (Hint) 248
 USE_MERGE (Hint) 248
 USE_NL (Hint) 248
 USE_STORED_OUTLINES 254
 user commits 99
 user rollbacks 99
 USER_DUMP_DEST 139
 Utlbstat 125
 utlxplp 136, 231
 utlxpls 136

V

V\$ACTIVE_SESSION_HISTORY 110
 V\$ASM_CLIENT 193
 V\$ASM_DISK 193
 V\$ASM_OPERATION 193
 V\$ASM_TEMPLATES 194
 V\$DB_CACHE_ADVICE 174
 V\$EVENT_HISTOGRAM 112
 V\$EVENT_NAME 105
 V\$FILESTAT 101, 118
 V\$FIXED_TABLE 96
 V\$INSTANCE_RECOVERY 202
 V\$LATCH_MISSES 263
 V\$LATCHNAME 110
 V\$LIBRARYCACHE 104
 V\$METRICNAME 105
 V\$MTTR_TARGET_ADVICE 203
 V\$OBJECT_USAGE 23
 V\$PGA_TARGET_ADVICE 224
 V\$PQ_SESSTAT 229
 V\$PQ_SYSSTAT 229

V\$ROWCACHE 104
V\$SEGSTAT 100
V\$SESS_IO 100
V\$SESS_TIME_MODEL 111
V\$SESSION 171
V\$SESSION_EVENT 105
V\$SESSION_WAIT 105, 108, 152
V\$SESSION_WAIT_CLASS 112
V\$SESSTAT 97
V\$SGA_RESIZE_OPS 181
V\$SHARED_POOL_ADVICE 180
V\$SQL_BIND_CAPTURE 128
V\$SQL_BIND_DATA 128
V\$SQL_PLAN 60
V\$STATISTICS_LEVEL 273

V\$SYS_TIME_MODEL 110
V\$SYSSTAT 97, 203, 219, 268, 272
V\$SYSTEM_EVENT 105, 124
V\$TEMPFILE 206
V\$TRANSPORTABLE_PLATFORM 49
V\$UNDOSTAT 205
V\$WAITSTAT 197
VW_NSO_1 91
vxfs 285

W

Waits (in 10046 Traces) 144
WIDTH_BUCKET() 78
WORKAREA_SIZE_POLICY 224
WRITEABLE (Materialized View) 11