PARAGON DATA

DOAG Datenbank 2014 - Braucht das Flagschiff Exadata tuning?

Malthe Griesel, Konstantin Balzer, Düsseldorf, 03.06.2014

Agenda

1. Kurzprofil Paragon Data
2. Exadata Architektur
3. Exadata Vorteile
4. Exadata - Smartscan
5. Exadata und Indizes
6. Exadata Smartscan Operationen
7. Exadata Smartscan – Wurde ein Smartscan ausgeführt?
8. Storageindex
9. Exadata Features ein- bzw. ausschalten
10. Exadata, Offloading und Funktionen
11. Smartscan vermessen
12. Beispiel
13. Weiteres Beispiel: AWR Bericht
14. Q &A

Kurzprofil Paragon Data

- Unternehmen der DBH/ Hugendubel
- 57 Mitarbeiter
- Schwerpunkt Buchhandel:
  DBH, Hugendubel, Weltbild, Weiland etc.
- IT-Dienstleister, Hoster, Filialservice, Hotline
- Hosting und Betrieb von Oracle-Datenbanken
- Hochsicherheitsrechenzentrum
- Oracle-Partner
- Oracle Engineered Systems (Oracle Exadata)

http://www.paragon-data.de

Kurzprofil Paragon Data – Auszug Kundenübersicht

Hugendubel
Die Welt der Bücher

A&M

S.FISCHER

Cornelsen

deugro

WOHLTHAT
Cultur Handlung

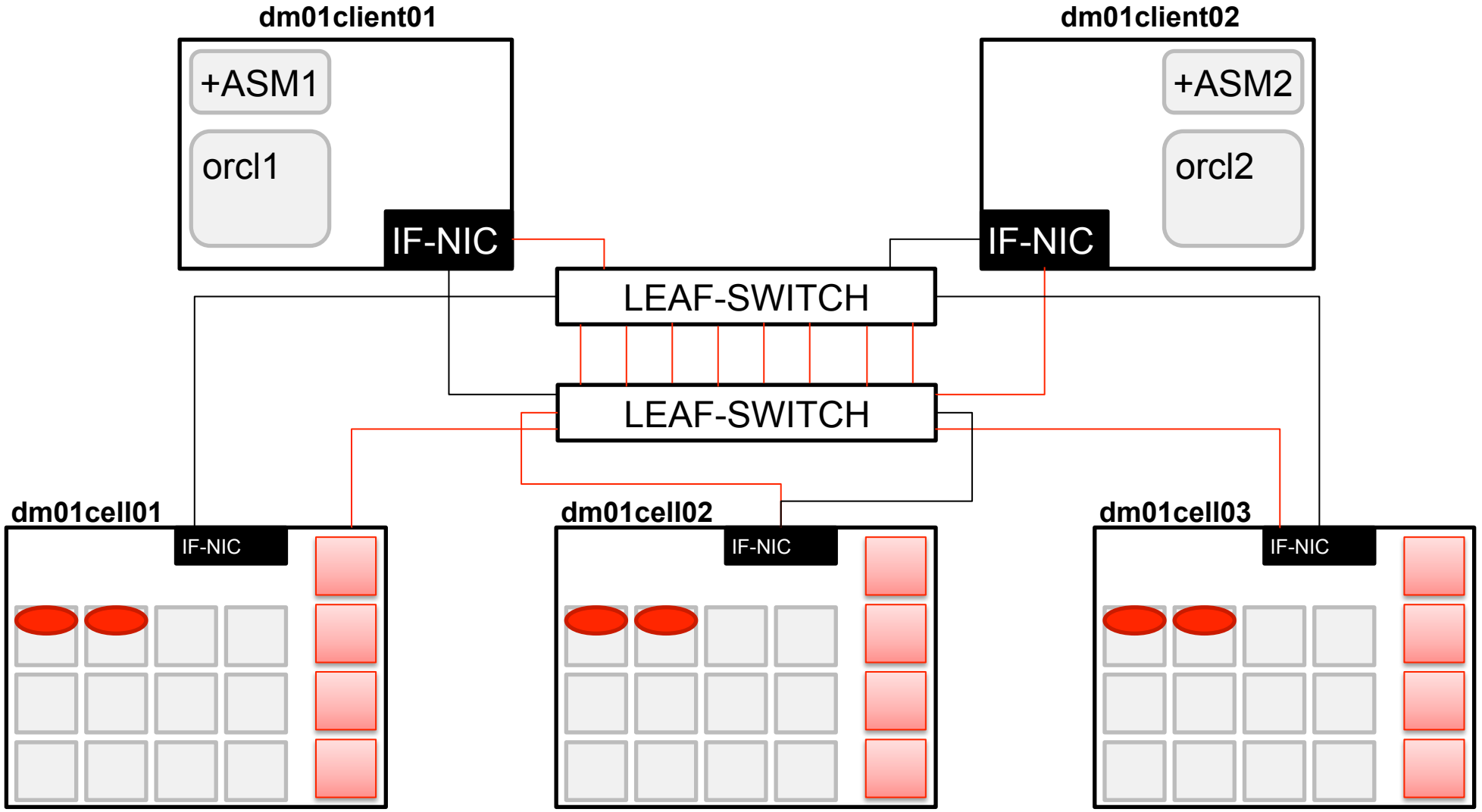
Jokers
Gute Bücher restlos günstig!

Weltbild plus

Börsenverein des Deutschen Buchhandels

PARAGON DATA

Exadata Architektur



Exadata Vorteile

- **Standard Hardware + Software**
  - X86-64 + OL UEK + Oracle RDBMS
- **Hohe Bandbreite**
  - Infiniband, 40 GB/s
  - PQ
- **Intelligenter Storage/ Storage Software**
  - Storageserver, Cellserver, Cell
- **Exadata Operationen**
  - Smart Scan, Offload
- **Exadata Software Optionen**
  - Column Projection
  - Predicate Filtering
  - Storage Indexes
  - Function Offloading
  - Smart Flash Cache
  - HCC

Exadata - Smartscan

Teil einer Abfrage, die auf dem Storageserver ausgeführt wird.

Wann wird ein Smartscan ausgeführt?

- Fulltablescan/ Indexfastfullscan
- Direct Path Read

Direct Path Read

- Physical Read
- Die Blöcke werden in der PGA gespeichert und nicht in der SGA/ Buffer Cache
- Kein Lesen von Blöcken aus dem Buffercache
- MOS Note 793845.1, Entscheidung zu Direct Path Read ist bei 11g abhängig von der Größe der zu lesenden Tabelle und der Größe des Buffer Cache
- Je größe der Cache, desto weniger Smartscans! (_serial_direct_read=TRUE)
- Direct Path Read vermeidet Latches!
- Direct Path Read hat nichts mit dem Optimizer zu tun!

Exadata und Indizes

Gerücht: Auf einer Exadata können alle Indizes gelöscht werden!

Wahrheit:

- **Primary Keys werden immer gebraucht, Einzelsatzzugriff!**
- **Joins!**
- **Der Gebrauch Indizes muss mit dem Gebrauch von Smartscans verglichen werden!**
- **Auch die Ressourcen einer Exadata sind endlich! Auch eine Exadata geht irgendwann einmal in die Knie!**
- **Weniger Indizes → Vorteile für DML!**

Exadata Smartscan Operationen

Exadata Smartscan Operationen:

→ **Column Projection**
- → Nur die referenzierten Spalten der select-Liste werden zurückgegeben!

→ **Predicate Filtering**
- → Es werden nur die Ergebniszeilen der Abfrage zurückgegeben. Es müssen trotzdem alle Daten gelesen werden!

→ **Storage Index Access**
- → Aufgrund der Where-Bedingungen werden dynamische In-Memory Indizes zur Beschleunigung erstellt.

→ **Function Offloading**
- → Bestimmte Funktionen werden auf den Storageservern ausgeführt!

→ **...**

Alle Operationen geschehen auf den Storageservern!

Exadata Smartscan – Wurde ein Smartscan ausgeführt?

Wurde ein Smartscan ausgeführt?

→ GV$SQL.IO_CELL_OFFLOAD_ELIGIBLE_BYTES > 0

```
select last_load_time, sql_id, child_number,
decode (IO_CELL_OFFLOAD_ELIGIBLE_BYTES,0,'No','Yes')
Offloadable
from v$sql
where sql_text like '%&sql_text%'
order by last_load_time desc;
```

Storageindex

Wurde ein Storageindex benutzt?

Session Level Statistik

"cell physical IO bytes saved by storage index"

```
select s.name, m.value
from v$statname s, v$mystat m
where name like ('%storage%')
and s.statistic# = m.statistic#
```

Exadata Features ein- bzw. ausschalten

- **Column projection/ predicate filtering**
  - alter session set cell_offload_processing = true;
  - alter session set cell_offload_processing = false;

- **Storage Indexes**
  - alter session set "_kcfis_storageidx_disabled"=true;
  - alter session set "_kcfis_storageidx_disabled"=false;

Smartscan vermessen

```
SQL> select name as stat_name,
       value/1024/1024 as value
from v$mystat s, v$statname n
where s.statistic# = n.statistic#
and n.name in ('cell physical IO bytes saved by storage index',
               'cell physical IO interconnect bytes returned by smart scan');

STAT_NAME                                                        VALUE
---------------------------------------------------------------- ----------------
cell physical IO bytes saved by storage index                    0
cell physical IO interconnect bytes returned by smart scan       332.033638

SQL>
```

Exadata, Offloading und Funktionen

- **Nicht jede Funktion kann Offloading nutzen!**
- **Keine selstdefinierten Fuktionen (PL/SQL)!**
-
```
SQL> select OFFLOADABLE, count(*)
     from v$sqlfn_metadata group by OFFLOADABLE;

OFF    COUNT(*)
---    ----------
NO     530
YES    393
```

- **Ist Funktion nicht zum Offloading geeignet, findet kein Smartscan statt!**
- **Funktionen in der Where Clause werden für jeden Satz ausgeführt!**

Beispiel

```
SQL> select count(*) ANZ
 2   from buch.dim_artikel_bestand_dbh where aenderung_kz like 'Bestellung%';

       ANZ
  --------
  65017788

Elapsed: 00:00:03.21
```

```
SQL> select sum(bytes)/1024/1024 MB
 2 from dba_segments where owner='BUCH'
 3 and segment_name='DIM_ARTIKEL_BESTAND_DBH';

        MB
----------
11689.8125
```

Ist das gut?

104 Partitionen!
diverse Indizes!

SQL_ID: 4nd1rqtxscx42

Beispiel - Wird Smartscan genutzt?

```
SQL> select last_load_time,
       sql_id,
       child_number,
       decode (IO_CELL_OFFLOAD_ELIGIBLE_BYTES,0,'No','Yes') Offloadable
from v$sql
where sql_id='4nd1rqtxscx42'
order by last_load_time desc;

LAST_LOAD_TIME           SQL_ID          CHILD_NUMBER     OFFLOADABLE
-------------------      -------------   ------------     -----------
2014-05-31/12:49:53      4nd1rqtxscx42         1          Yes
```

Beispiel – Wird der Storageindex genutzt?

```
SQL> select s.name, m.value
from v$statname s, v$mystat m
where name like ('%storage%')
and s.statistic# = m.statistic;

NAME                                                                  VALUE
---------------------------------------------------------------- ----------
cell physical IO bytes saved by storage index                              0

SQL>
```

Beispiel – Wieviel wird gespart?

```
SQL> select sql_id,
decode(IO_CELL_OFFLOAD_ELIGIBLE_BYTES,0,'No','Yes') Offloaded,
decode(IO_CELL_OFFLOAD_ELIGIBLE_BYTES,0,0,
100*
(IO_CELL_OFFLOAD_ELIGIBLE_BYTES/IO_INTERCONNECT_BYTES)/
IO_CELL_OFFLOAD_ELIGIBLE_BYTES) as "IO_SAVED"
from v$sql
where sql_id = '4nd1rqtxscx42';

SQL_ID            OFF      IO_SAVED
-------------     ---      ----------
4nd1rqtxscx42     Yes      9.7806E-07
```

Warum so wenig Ersparnis?

Beispiel – HCC Compression

```
SQL> select dbms_compression.get_compression_type('BUCH',
                                                  'DIM_ARTIKEL_BESTAND_DBH',
                                                  'AACYMGAD0AAAA1zAAI') Comp
from dual;

COMP
----
   4
```

HCC, Compress for Query High

Beispiel HCC Compression

```
SQL> @get_compression_ratio

Compression Advisor self-check validation successful. select count(*) on both
Uncompressed and EHCC Compressed format = 1000001 rows
Estimated Compression Ratio: 8.1
Blocks used by compressed sample: 1807
Blocks used by uncompressed sample: 14711

PL/SQL procedure successfully completed.

SQL>
```

Beispiel von Uwe Hesse
http://uhesse.com/2011/09/12/dbms_compression-example/

Weiteres Beispiel: AWR Bericht

Top Timed Events

- Instance '*' - cluster wide summary
- '*' Waits, %Timeouts, Wait Time Total(s) : Cluster-wide total for the wait event
- '*' 'Wait Time Avg (ms)' : Cluster-wide average computed as (Wait Time Total / Event Waits) in ms
- '*' Summary 'Avg Wait Time (ms)' : Per-instance 'Wait Time Avg (ms)' used to compute the following statistics
- '*' [Avg/Min/Max/Std Dev] : average/minimum/maximum/standard deviation of per-instance 'Wait Time Avg(ms)'
- '*' Cnt : count of instances with wait times for the event

	Wait		Event		Wait Time			Summary Avg Wait Time (ms)				
I#	Class	Event	Waits	%Timeouts	Total(s)	Avg(ms)	%DB time	Avg	Min	Max	Std Dev	Cnt
*		DB CPU			459,560.00		80.06					2
	User I/O	cell single block physical read	297,599,163	0.00	82,335.94	0.28	14.34	0.28	0.27	0.28	0.01	2
	System I/O	db file parallel write	2,616,487	0.00	13,577.67	5.19	2.37	5.04	4.60	5.47	0.62	2
	Network	SQL*Net message from dblink	27,940,644	0.00	12,555.58	0.45	2.19	0.49	0.36	0.62	0.18	2
	User I/O	direct path read temp	803,809	0.00	7,746.84	9.64	1.35	9.66	9.28	10.04	0.54	2
	User I/O	cell multiblock physical read	5,651,051	0.00	6,191.98	1.10	1.08	1.09	1.07	1.12	0.04	2
	System I/O	log file sequential read	420,572	0.00	4,102.56	9.75	0.71	9.54	8.99	10.08	0.78	2
	Other	PX Deq: reap credit	433,913,762	100.03	3,976.59	0.01	0.69	0.01	0.01	0.01	0.00	2
	Network	SQL*Net more data from dblink	3,008,044	0.00	3,363.87	1.12	0.59	1.11	1.06	1.15	0.07	2
	Cluster	gc cr multi block request	4,821,462	0.00	2,944.03	0.61	0.51	0.61	0.58	0.65	0.05	2

Fragen?

Q & A

Vielen Dank!

Malthe Griesel, M.Griesel@paragon-data.de, Konstantin Balzer, K.Balzer@paragon-data.de