

Performante Verarbeitung großer Datenbanken am praktischem Beispiel



Thomas Lehmann – 08.09.2015, Dresden

Agenda

1. Technische Rahmenbedingungen
2. Theoretische Grundlagen
3. Verschiedene Probleme am praktischen Beispiel

Allgemeine Anmerkungen

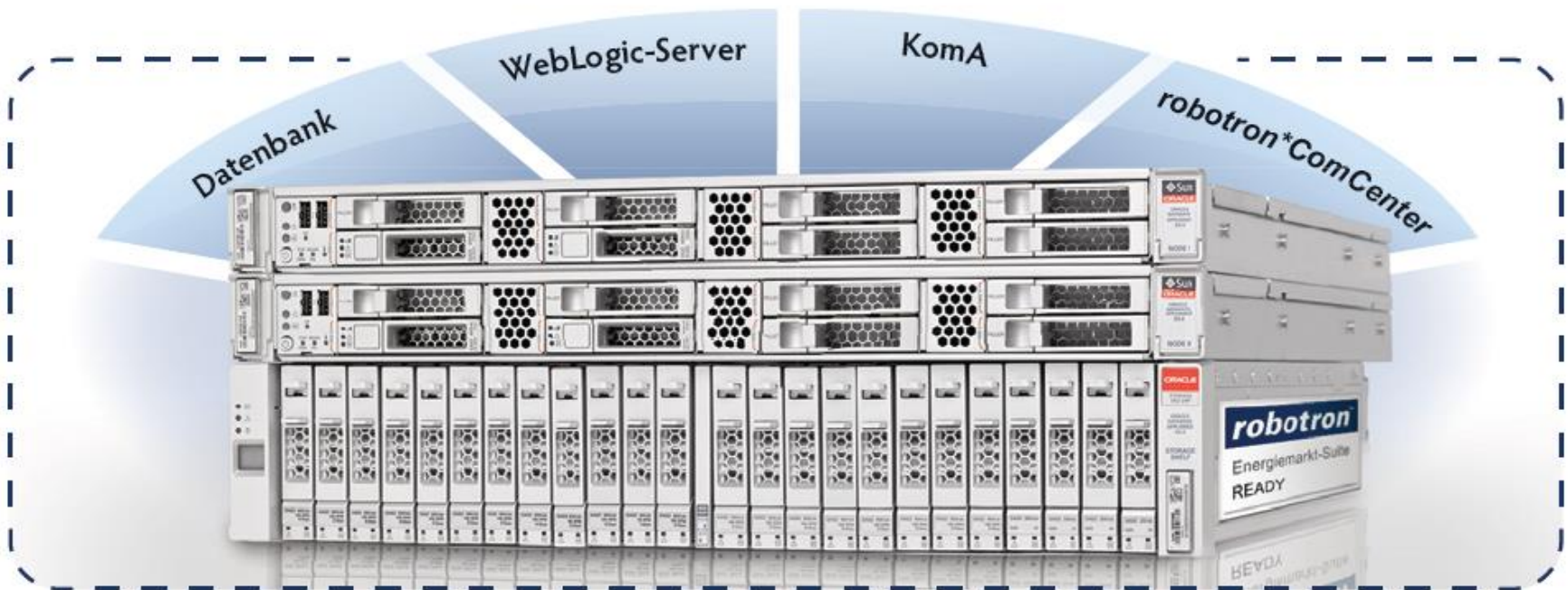
- ▶ Keine vorgefertigten Universallösungen
- ▶ Möglichkeiten der Oracle Datenbank aufzeigen
- ▶ Individuelle Entscheidung / Abstimmung notwendig
- ▶ Seiteneffekte im Auge behalten

Technische Rahmenbedingungen

- ▶ Programmlogik in der Datenbank (PL/SQL-Code)
- ▶ Serielle Programmabarbeitung
- ▶ Hohes Datenvolumen
- ▶ Wiederkehrende Prozesse

- ▶ Leistungsstarke Serverhardware (CPU, IO, Memory)
- ▶ Optimale Ressourcennutzung ???

Oracle Database Appliance → Robotron EDM Appliance



Zertifizierte Hard- und Software-
Komplettlösung für den Betrieb der Robotron-Energemarkt-Suite

Theroretische Grundlagen

- ▶ Wie gelangen Daten in die Datenbank ?
 - Tools
 - Tabellenlayout
 - Statistiken
- ▶ Wie können DML-Operationen auf großen Datenmengen durchgeführt werden ?
 - DML / CTAS ?
 - Row / Set-Based ?

External tables

- ▶ Voraussetzung: strukturierte Daten
- ▶ Einfach einzurichten aber statisch
- ▶ Empfehlenswert bei Initialbeladungen / Entladungen oder wiederkehrenden Prozessen
- ▶ Zugriff auf DB-Server
- ▶ Steuerung über DDL-Statement

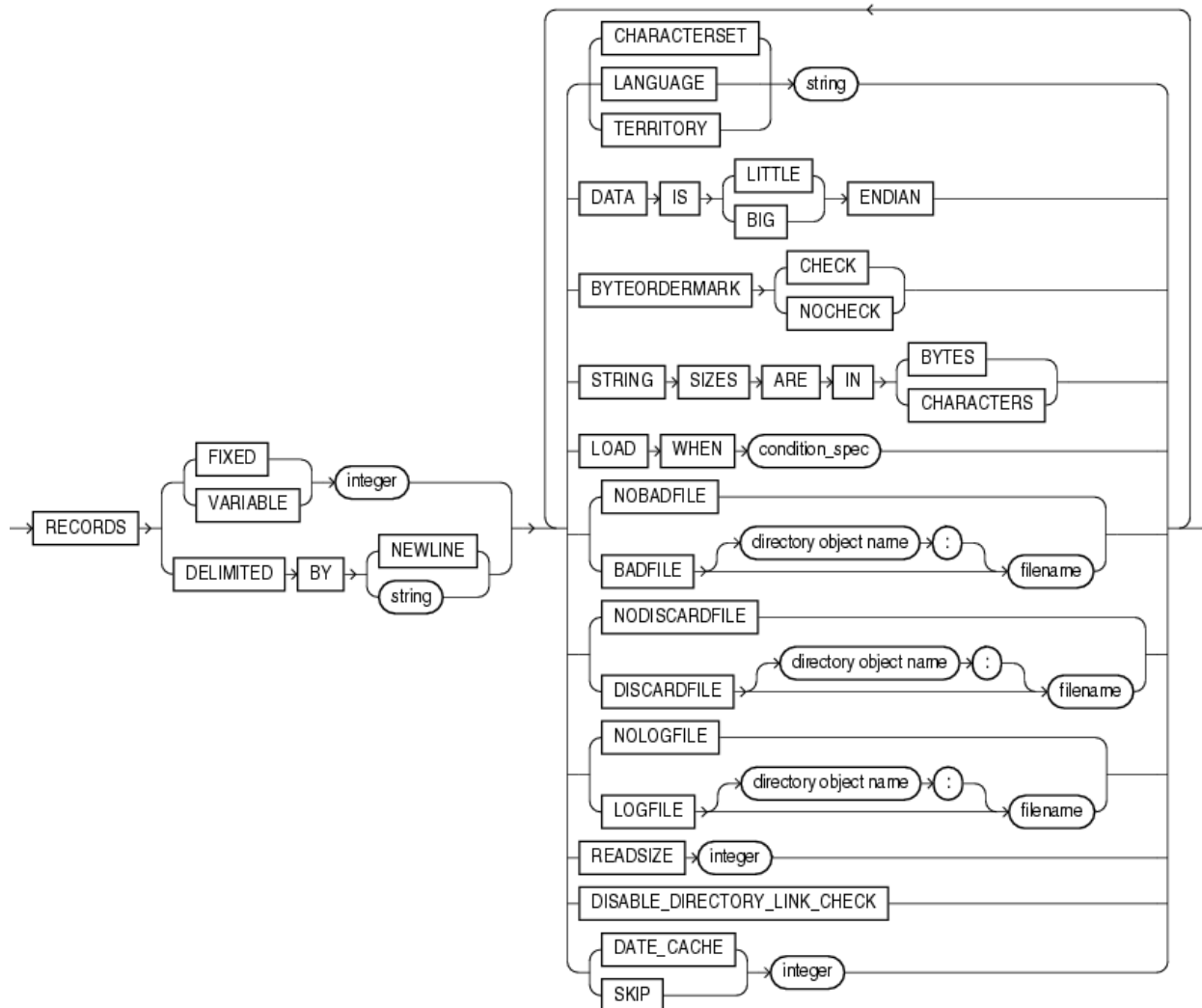
- ▶ Beispiel

External tables

```
CREATE TABLE imp_std_csv_data (  
  INT_ID VARCHAR(10),  
  ID VARCHAR(10),  
  name VARCHAR(200),  
  name2 VARCHAR(200),  
  plz VARCHAR(10),  
  ort VARCHAR(200),  
  strasse VARCHAR(200),  
  region VARCHAR(10),  
  met_code VARCHAR(50)  
)
```

```
ORGANIZATION EXTERNAL (TYPE ORACLE_LOADER DEFAULT DIRECTORY extern_data_dir  
  ACCESS PARAMETERS (FIELDS TERMINATED BY ';')  
  LOCATION ('data01.csv'));
```


External tables



SQL Loader

- ▶ Voraussetzung: strukturierte Daten
- ▶ Einfach einzurichten
- ▶ Keine Festen Dateinamen
- ▶ Empfehlenswert bei wiederkehrenden Prozessen
- ▶ Zugriff über SQL Net
- ▶ Steuerung über Control File

- ▶ Beispiel

External Tables vs. SQL Loader

- ▶ External Tables:
 - Datentransformation beim Laden
 - Parallelisierung
 - NLS-Einstellung der Datenbank
 - Performanter

- ▶ SQL Loader:
 - Remotezugriff
 - Zusätzliche Indizierung der Staging Table
 - NLS-Einstellung des Clients

Statistiken & Indizes

- ▶ Massenbeladung wenn Möglich ohne Indizes / Trigger
- ▶ Anlegen der Indizes nach Beladeprozess

- ▶ Aktualisierung der Statistiken (evtl. inkrementell)
- ▶ Besonderheit der Histogramme

- ▶ Beispiele

Statistiken

► DBMS_STATS.GATHER ... STATS

```
DBMS_STATS.GATHER_TABLE_STATS (  
  ownname          VARCHAR2,  
  tabname          VARCHAR2,  
  partname         VARCHAR2 DEFAULT NULL,  
  estimate_percent NUMBER   DEFAULT to_estimate_percent_type  
                    (get_param('ESTIMATE_PERCENT')),  
  block_sample     BOOLEAN  DEFAULT FALSE,  
  method_opt       VARCHAR2 DEFAULT get_param('METHOD_OPT'),  
  degree           NUMBER   DEFAULT to_degree_type (get_param('DEGREE')),  
  granularity      VARCHAR2 DEFAULT GET_PARAM('GRANULARITY'),  
  cascade          BOOLEAN  DEFAULT to_cascade_type (get_param('CASCADE')),  
  stattab         VARCHAR2 DEFAULT NULL,  
  statid           VARCHAR2 DEFAULT NULL,  
  statown         VARCHAR2 DEFAULT NULL,  
  no_invalidate    BOOLEAN  DEFAULT to_no_invalidate_type (  
                    get_param('NO_INVALIDATE')),  
  force           BOOLEAN  DEFAULT FALSE);
```

method_opt Accepts either of the following options, or both in combination:

- FOR ALL [INDEXED | HIDDEN] COLUMNS [size_clause]
- FOR COLUMNS [size clause] column|attribute [size_clause] [,column|attribute [size_clause]...]

size_clause is defined as size_clause := SIZE {integer | REPEAT | AUTO | SKEWONLY}

column is defined as column := column_name | (extension)

- integer : Number of histogram buckets. Must be in the range [1,254].
- REPEAT : Collects histograms only on the columns that already have histograms.
- AUTO : Oracle determines the columns to collect histograms based on data distribution and the workload of the columns.
- SKEWONLY : Oracle determines the columns to collect histograms based on the data distribution of the columns.
- column_name : name of a column
- extension : can be either a column group in the format of (column_name, column_name [, ...]) or an expression

The default is FOR ALL COLUMNS SIZE AUTO. The default value can be changed using the [SET DATABASE_PREFS Procedure](#), [SET GLOBAL_PREFS Procedure](#), [SET SCHEMA_PREFS Procedure](#) and [SET TABLE_PREFS Procedure](#).

Partitionierung

- ▶ Aufteilung von Tabellen und Indizes
- ▶ Transparent für die Anwendung

- ▶ Verbesserung der Abfrageperformance
- ▶ Beschleunigung von Wartungsvorgängen

- ▶ Lizenzkosten beachten
- ▶ Evtl. zusätzliche Wartungsaufgaben

- ▶ Beispiele

Parallelisierung

- ▶ PL/SQL Code läuft auf einer CPU Core
- ▶ → keine Skalierungsfähigkeit
- ▶ → sequentielle Abarbeitung

- ▶ → langlaufende Aktionen parallelisieren

- ▶ Row-By-Row → Bulk Operations → Parallelisierung

Parallelisierung, Partitionierung und Komprimierung

- ▶ Optimale Ressourcenausnutzung durch Parallelisierung
- ▶ Achtung: Keine Überbeanspruchung
- ▶ → Customizingmöglichkeiten im Code/Layout vorsehen

- ▶ Aufteilung der Daten durch Partitionierung
- ▶ Performancegewinn vs. Maintenance

- ▶ Verringerung des Datenbestandes durch Komprimierung
- ▶ Verringerung Lesezugriffe (Tagesgeschäft, Betrieb)

- ▶ Hinweis: Lizenzkosten im Auge behalten

Row Bases vs. Set Based

- ▶ „eCount-Krankheit“: Verarbeitungsschritte Row Based
- ▶ (Nachvollziehbare Gründe: Trigger, Mandanten, komplexe Logik)
- ▶ Langsamste Verarbeitung innerhalb der Datenbank

- ▶ Oracle beherrscht JOINS - Performant und Skalierbar !

Row Bases vs. Set Based

- ▶ Problem: 1 GB große Tabelle mit ca. 3 Mio Datensätzen soll ein Update erfahren, bei dem:
 - IDs < 300.000 → Region ‚SN‘
 - IDs < 400.000 → Region ‚NRW‘
 - IDs < 500.000 → Region ‚BY‘
 - IDs >500.000 → Region ‚BL‘

Zusammenfassung

- ▶ Wachsende Datenmengen
- ▶ Wachsende Anforderungen
- ▶ Leistungsstarke Hardware

- ▶ Optimale Auslastung der Ressourcen
- ▶ Maximale Performancesteigerung durch Code / Designanpassungen
- ▶ Supportabteilung lässt sich gern in Designphasen einbinden



Thomas Lehmann
Senior Systemberater Support

Tel: +49 351 25859 2782

Fax: +49 351 25859 3696

thomas.lehmann@robotron.de