robotron®

DOAG 2011 Konferenz

Praxisbericht DWH mit 5 Milliarden Fakten

ORACLE Certified Professional
Oracle Database 10g Administrator

ORACLE Certified Expert
Oracle Real Application Clusters 10g Administrator

Marco Mischke, 15.11.2011

Agenda

robotron®

- Anforderungen
- Mengengerüst
- Datenmodell

- Evolution des Datenmodells
- „exotische“ Oracle Features

- Betrieb
- ETL Prozesse
- Ausblick

Problem DWH 5.000.000.000

robotron®

- Analyse von Fertigungs-Prozessdaten
- Hunderte Schritte vom Rohmaterial bis zum Fertigprodukt
- Ca. 200 Maschinen
  - pro Maschine werden Daten generiert, pro Los (25 Wafer), pro Wafer, und pro Arbeitsschritt
  - Prozessdaten (z.B. Tool Parameter, Einstellungen, etc.)
  - Logistik Daten (Produkt, Material, etc.)
  - Messdaten (z.B. Schichtdicke an 49 Messpunkten pro Wafer)
- Keine sinnvolle Aggregation (a la OLAP Cube) möglich

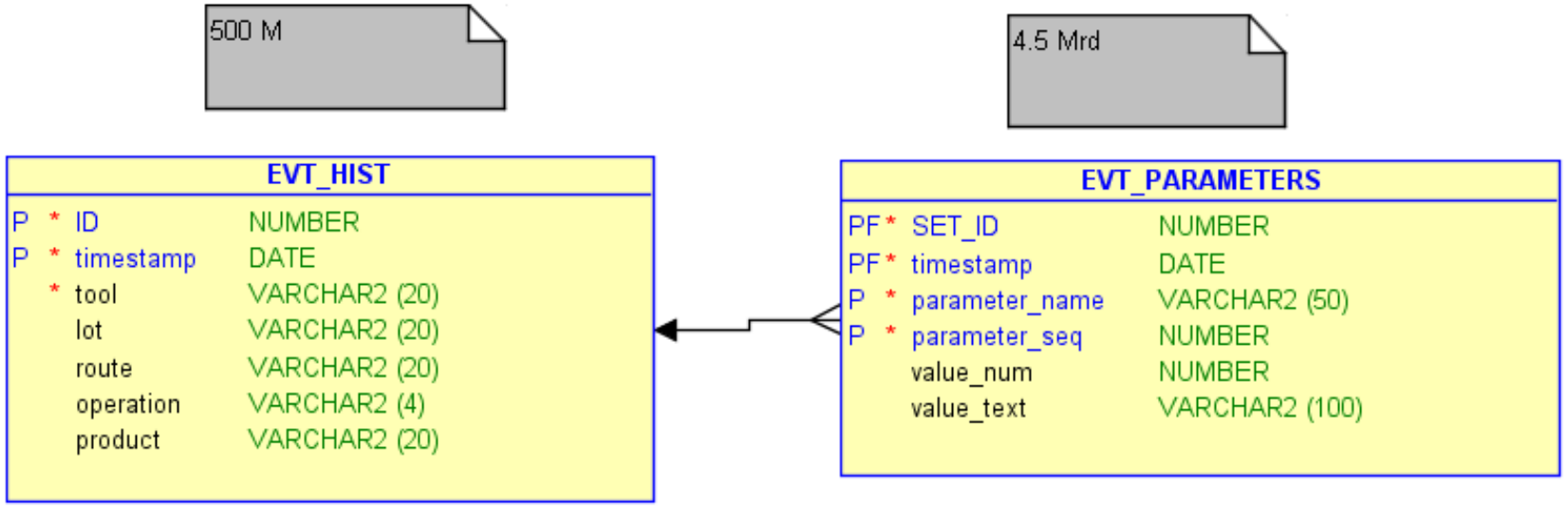
Mengengerüst

robotron®

- 200 Tools x 2.000 Losbewegungen x 25 Wafer x 50 Werte x 45 Tage
  - 4.500.000.000 Einzelwerte
- Zeitnahe Auswertung
  - 5-15 Minuten nach Entstehen der Daten
  - Rückwirkend über mehrere Wochen
- Data Lifecycle
  - Löschen von alten Daten
    - Kosten (Storage), Performance, Verfügbarkeit (Backup/Restore)

Datenmodell (logisch)

robotron®



Agenda

robotron®

- Anforderungen
- Mengengerüst
- Datenmodell

- Evolution des Datenmodells
- „exotische“ Oracle Features

- Betrieb
- ETL Prozesse
- Ausblick

Partitionierung EVT_HIST

robotron®

- Alle Abfragen haben eine Einschränkung nach Zeit + Tool (ggf. mehrere)
- Timestamp, subpartition by hash(tool), 8 Sub-Partitionen
- Oracle Partition Pruning selektiert nur noch relevante Partitionen
- Datenkompression (seit Oracle 8)
  - Daten der Vorwoche werden jeweils komprimiert
  - `create table … pctfree 0 compress as select … from EVT_HIST order by tool, timestamp`
  - `alter table exchange partition including indexes without validation`
  - 60% Storage Einsparung

    Aktuelle Woche: ca. 3GB

    Ältere Wochen ca. 1GB pro Woche

Partitionierung EVT_HIST

robotron®

	Tool1 Tool9 Tool17	Tool2 Tool10 Tool18	Tool3 … …	…	Tool200 … …
KW18	Part1	Part2	Part3	…..	…..
KW19	….				
KW20	….				
…	….				
KW53	….				Part N

```
select … from EVT_HIST
where timestamp between '01-Jun-2010'
                    and '04-Jun-2010'
  and tool = 'Tool2'
```

Indizes EVT_HIST

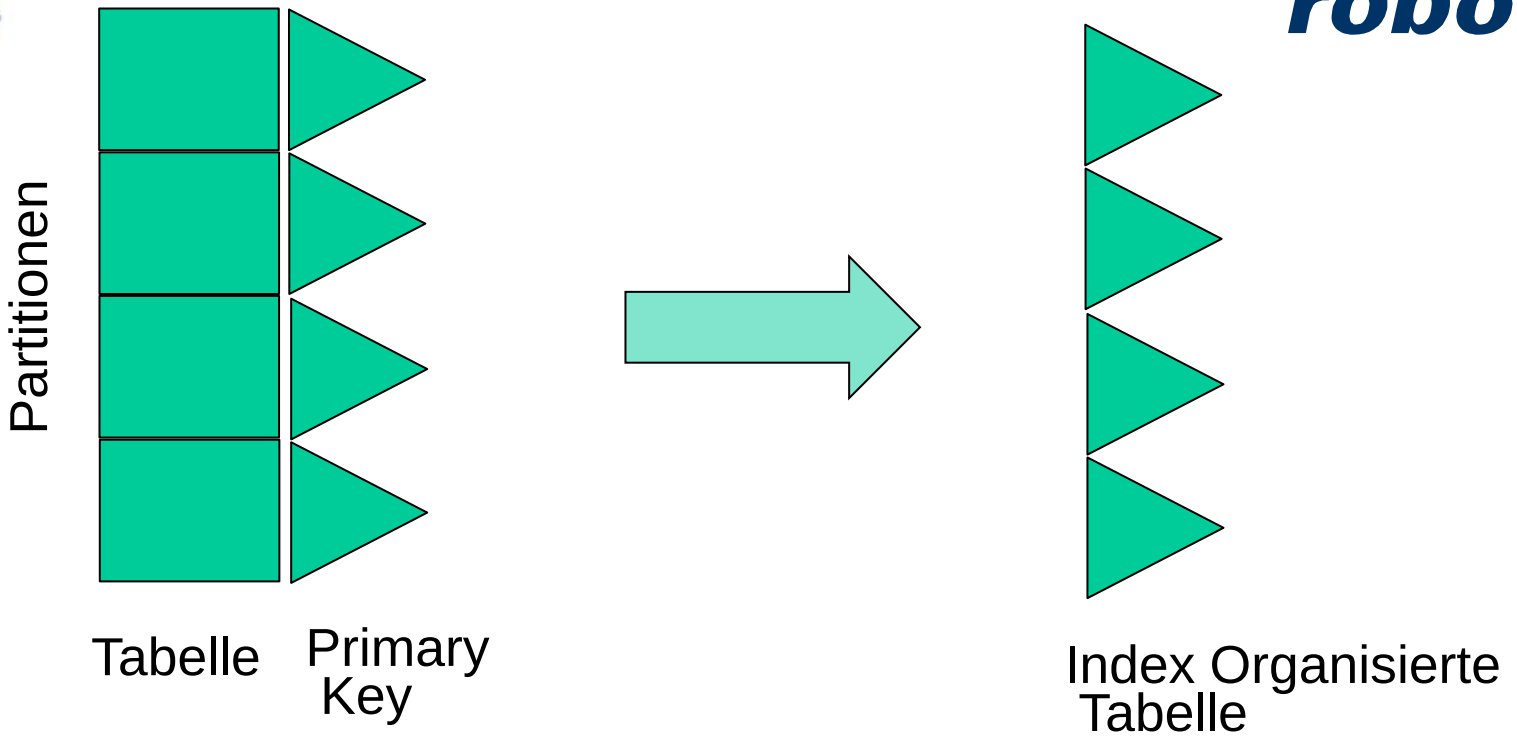
robotron®

- Keine globalen Indizes
  - `Create index …` **`local`**
  - Effiziente Partition Maintenance (drop, exchange)
- Bitmap Index auf allen Spalten
  - Inclusive Timestamp (!)
  - Abfragemuster
    - *Immer:* Einschränkung nach Zeit
    - *Meistens:* Tool, bzw. Toolgruppen
    - *Oft:* logistische Daten
      (Los, Produkt, Operation etc. in beliebigen Kombinationen)
  - `Alter table` **`minimize_records_per_block`**
    - Optimierung der Bitmap Indizes (-20% Storage pro Index)
    - ORA-14643: Hakan factor mismatch for tables in ALTER TABLE EXCHANGE PARTITION

EVT_PARAMETERS
Optimierung: Schritt 1

robotron®

- Primary Key besteht aus 4 Spalten (Tabelle hat 6 Spalten)
- Nur ein sinnvoller Zugriffspfad, nur 1 Index
  - Join mit EVT_HIST Faktentabelle
  - **Index Organized Table (IOT)**
  - Partitioniert nach Zeitstempel (Tagespartitionen)
  - Index Rebuild **pctfree=0** nach Tagesabschluss



-50% Storage

-Daten sind vorsortiert,
gemeinsam abgefragte
Daten liegen physikalisch nebeneinander
→ 70% weniger I/O für typische Reports

EVT_PARAMETERS
Optimierung: Schritt 2

robotron®

- Pro Event (Timestamp, ID) gibt es hunderte Parameter
  - In der Parameter Tabelle gibt es viele Duplikate von (Timestamp, SET_ID)
- **Index Prefix Compression**

robotron®

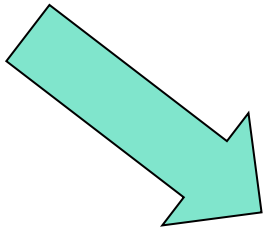
Timestamp, ID, Parameter_Name, Parameter_SEQ, Value

2010-03-27 10:00:00, 10, Schichtdicke , 1, 300

2010-03-27 10:00:00, 10, Schichtdicke , 2, 310

2010-03-27 10:00:00, 10, Temp, 1, 78.5

… etc



-30% Storage

(zusammen mit Schritt 1: 85%)

PREFIX- Tabelle pro Block

1 = 2010-03-27 10:00:00, 10

Prefix, Parameter_Name , Parameter_SEQ, Value

1, Schichtdicke, 1, 300

1, Schichtdicke, 2, 310

1, Temp, 1, 78.5

… etc

EVT_PARAMETERS
Optimierung: Schritt 3

robotron®

- Parameter Namen sind Texte (durchschnittl. 20 Zeichen)
- Nur ca. 20.000 verschiedene Parameternamen
- → Auslagern in Lookup Tabelle
  - Ersetzen durch numerisches Feld
- Alle Reports brauchen Parameter Namen
  (Join mit Lookup Tabelle, Tausende Lookups pro Report)
  - Index Zugriff ist zu teuer (ca. 3 logische I/O pro Lookup)

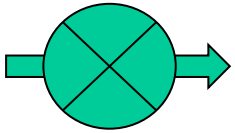
  → Oracle Single Table Hash Cluster (nur 1 I/O pro Lookup)

Single Table Hash Cluster

robotron®

```
CREATE CLUSTER name_cluster
(id number)
 SIZE 30 SINGLE TABLE HASHKEYS 40000;

CREATE TABLE parameter_names
(id number,
 parameter_name varchar2(50))
 cluster name_cluster(id);
```

ID  Physikalische Block Adresse

Hash Funktion

Dateninhalte = „Index“
schnellst-möglicher Zugriffspfad (neben ROWID)

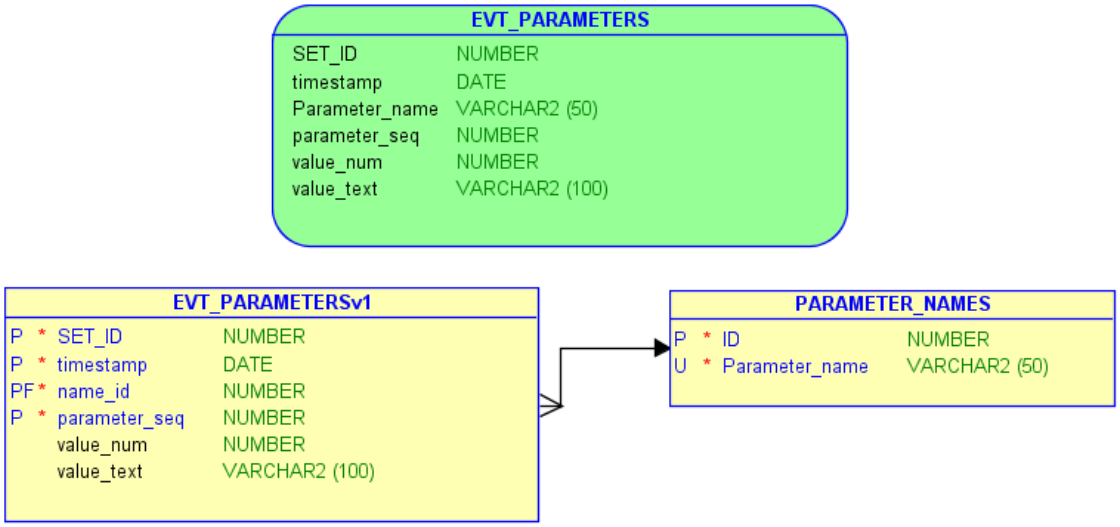
Nutzen Hash Cluster

robotron®

- Storage: -40% (ca. 15 Bytes weniger pro Zeile = 62GB)
  - Insgesamt (Schritt 1-3) -94% gegenüber trivialem Design
- Komplexeres Reporting SQL durch View vermieden
  - Keine Anpassung der Reports notwendig
- Keine merkbaren Performance Einbußen
  - Single Table Hash Access
  - Aufwand für Join wird durch weniger I/O kompensiert
- Einbindung in ETL Prozess
  - Pflege der Lookup Tabelle

Neues Datenmodell EVT_PARAMETERS

robotron®



Join wird in View „versteckt“

Zwischenfazit

robotron®

- Massive Einsparungen Storage
  - Initiale Schätzung: >> 1 TB
  - Nach Optimierung: ca. 300GB
  - → Gute I/O Performance durch Caching (SAN, Oracle)
- Report Performance zufriedenstellend
  - Durchschnitt < 10 Sekunden
    - Parameter Report für 1 Event: < 0.5 Sekunden
  - Hohe Akzeptanz (durch gute Performance)
  - Interaktives Arbeiten ist gängige Praxis
    - Reports mit Laufzeiten > 5 Minuten werden kaum genutzt
- Vorberechnung von Reports nicht sinnvoll
  - < 1% der Datenmenge wird überhaupt ausgewertet
  - Daten werden 15 Minuten nach Entstehen ausgewertet

Agenda

robotron®

- Anforderungen
- Mengengerüst
- Datenmodell

- Evolution des Datenmodells
- „exotische“ Oracle Features

- Betrieb
- ETL Prozesse
- Ausblick

Betriebserfahrungen

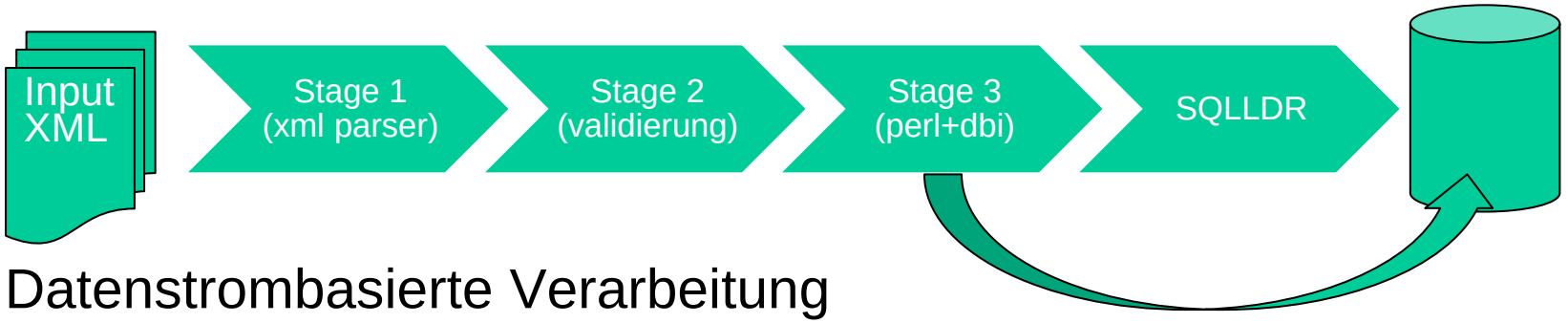
robotron®

- Produktiv seit 2002
- Iterative Entwicklung „agile Entwicklungsprozesse“
  - Kein dauerhafter Nutzen ohne ständige Weiterentwicklung
- Oracle Fehler
  - Im Einzelnen sehr zuverlässig, Ora-600 etc. bei Kombination
    z.B. List/Hash Partition+ Parallel Query
  - Upgrade Bugs: 8→9→10
- Oracle Fremdschlüssel disabled
  - „drop Partition“ – Probleme
  - Lösung: ggf. Oracle 11 Reference Partitioning
- ETL Durchsatz mindestens 2x größer als Datenaufkommen

ETL – Übersicht

robotron®

- Input: XML- ähnliche Text Files
- Mehrere Stages



- Datenstrombasierte Verarbeitung
  - Niemals gesamten Input in RAM laden (XML Parser!)
  - Kopplung der Stages über Unix STDIN/STDOUT
  - Parallele Bearbeitung (Multi Core/Multi CPU)
    - Stage N+1 kann schon arbeiten, während Stage N noch läuft
  - Für ETL: jeden logischen Record genau 1x Lesen / Schreiben
    - Zwischenergebnisse **nicht** als Datei/Tabelle materialisieren

ETL – Stage 1

robotron®

- Verwendung des SAX Parser API
  - Geringe Anforderungen an RAM
    - DOM Parser baut gesamten XML Baum im RAM auf
  - Callback Funktionen für Elemente
  - Dateigröße nicht limitiert
  - Daten können schon vor Erreichen des Dateiendes an nächste Stufe weitergegeben werden

ETL – Stage 2/3

robotron®

- Stage 2 – Validierung
  - Optionale datensatzbezogene Prüfungen (Datentypen, Plausibilität)
- Stage 3 – Ladeprozess
  - Pflege von Stammdatentabellen mittels Perl DBI Modul
    - Umsetzen der Texte in IDs (aus Lookup Tabelle)
  - Prüfen auf Duplikate
    - Entscheidung, INSERT (Load) oder UPDATE
  - Laden der Daten mittels SQLLDR über STDIN
    - Perl: open(,| sqlldr … infile=-‘);

Data Lifecycle

robotron®

- Alle Tabellen mit Bewegungsdaten sind nach Zeitstempel partitioniert
- Löschen alter Daten abgebildet über Partitionierung
  - `alter table …` **`drop partition`**
  - Keine Belastung durch DELETE Operationen
  - Praktisch kein Redo/Undo
  - Segmente/Extents werden wieder freigegeben → keine „Lücken“ in den Tabellen
  - Löschen bleibt zeitlich konstant, da ausschließlich lokale Indizes verwendet werden

Ausblick/Ideen

robotron®

- Oracle 11g New Features
  - Advanced Compression (vermutlich nicht relevant, €€€)
  - Partitioning New Features
    - Interval Partitioning
      Ablösung von selbst geschriebenen PL/SQL jobs
    - Reference Partitioning sehr vielversprechend
- Standard Reporting/BI Tools
  - Ggf. Ablösung von handgeschriebenem SQL
- Linux

Fragen

robotron®

robotron®

DANKE

marco.mischke@robotron.de
www.robotron.de