



**ORACLE®**  
**DATA WAREHOUSE**

**SOFTWARE. HARDWARE. COMPLETE.**

# Oracle Data Warehouse – Datenbank basierte ETL-Prozesse

Reduzieren Sie Ihre Ladezeiten auf ¼!

**ORACLE®**  
**DATA WAREHOUSE**

# Das Thema



## Datenbank-basierte ETL-Prozesse

- Verlagerung der ETL-Prozesse in die Datenbank
- Datenbank fungiert als ETL-Server
- Daten werden möglichst wenig bewegt
- Kurze Ladezeiten

# Das große Klagen

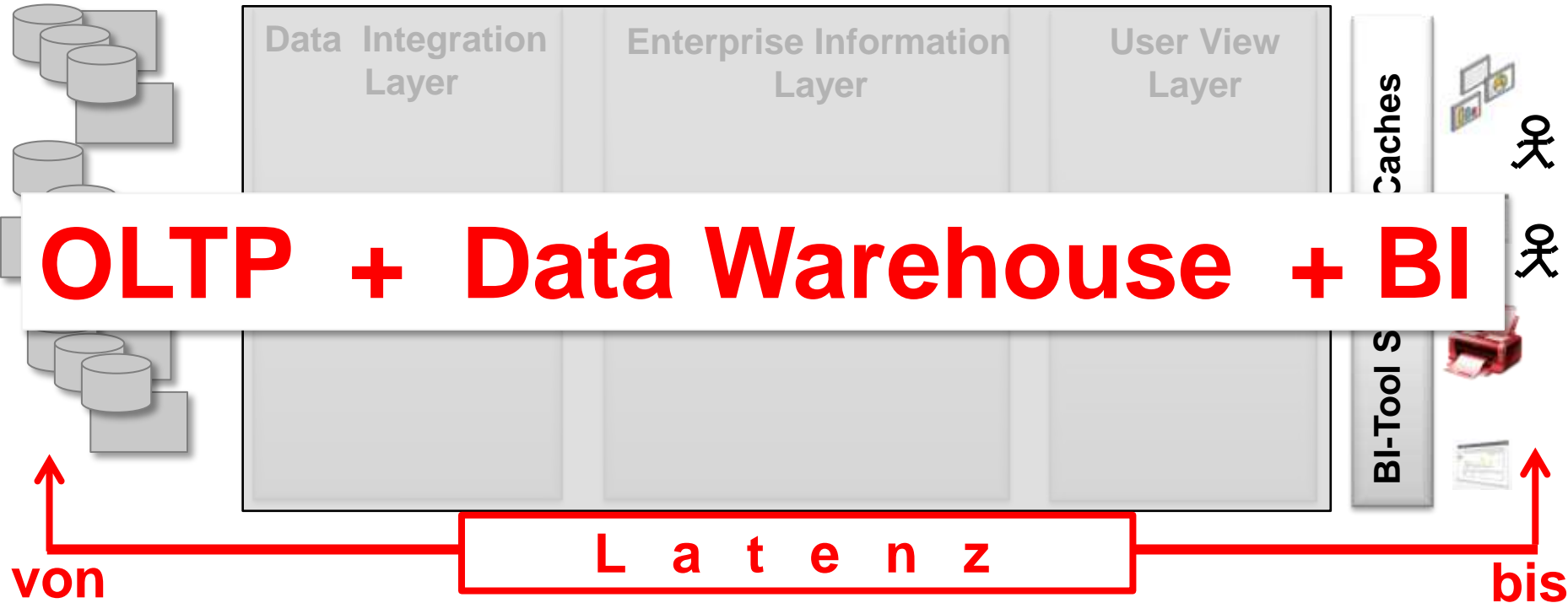
## Bzgl. System-Nutzen

- Lieferzeiten der Daten zu lange (Latenzen)
- Zu schwerfällig bei Änderungen
- Informationen mehrfach vorhanden
- Fehlende unternehmensweite Sichten
- Nicht die richtigen Informationen für die Anwender
- Anwender haben zu wenig unmittelbaren Einfluss auf die Daten

## Bzgl. Maintenance und Technik

- Immer teurer
- Maintenance-Aufwand zu hoch / Personal
- Explodierende Datenmengen -> Storage- / Ladezeitthematik

# Was definieren wir als ETL-Prozesse?



**Reduzieren Sie Ihre  
Ladezeiten auf  $\frac{1}{4}$ !**

# Ziele eines effizienten ETL-Prozesses

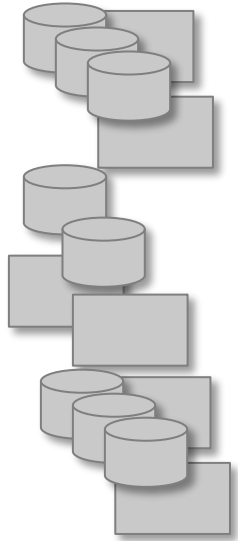
- **Ressource-schonend**
  - Rechenzeit, Storage
- **Schnell änderbar und pflegbar**
- **Kurze Laufzeiten**
- **Erzeugen von stimmigen Abfrage-Ergebnissen**
- **Erleichterung für BI-Tools**

....

# Allgemeine Regeln

- **Schichtenmodell als Orientierung nutzen**
- **Auf die spezifischen Anforderungen und Situationen in den jeweiligen Schichten reagieren**
- **Transformationen so früh wie möglich im Verlauf des Schichtenmodells**
- **Alle Schichten innerhalb derselben Datenbank**
- **Daten nur dann bewegen, wenn sich qualitativ etwas verändert.**
- **Nur diejenigen Daten laden, die wirklich benötigt werden**
  
- **Prüfungen an wenigen Stellen konzentrieren**
- **Bei Data Marts prüfen, ob sie permanent bereit gehalten werden**

# Angemessen in den Situationen agieren



## Data Integration Layer

**Keine  
Daten**

## Enterprise Information Layer

**Referenzdaten  
Stammdaten  
Bewegungsdaten  
(granulare  
Transaktionsdaten)**

## User View Layer

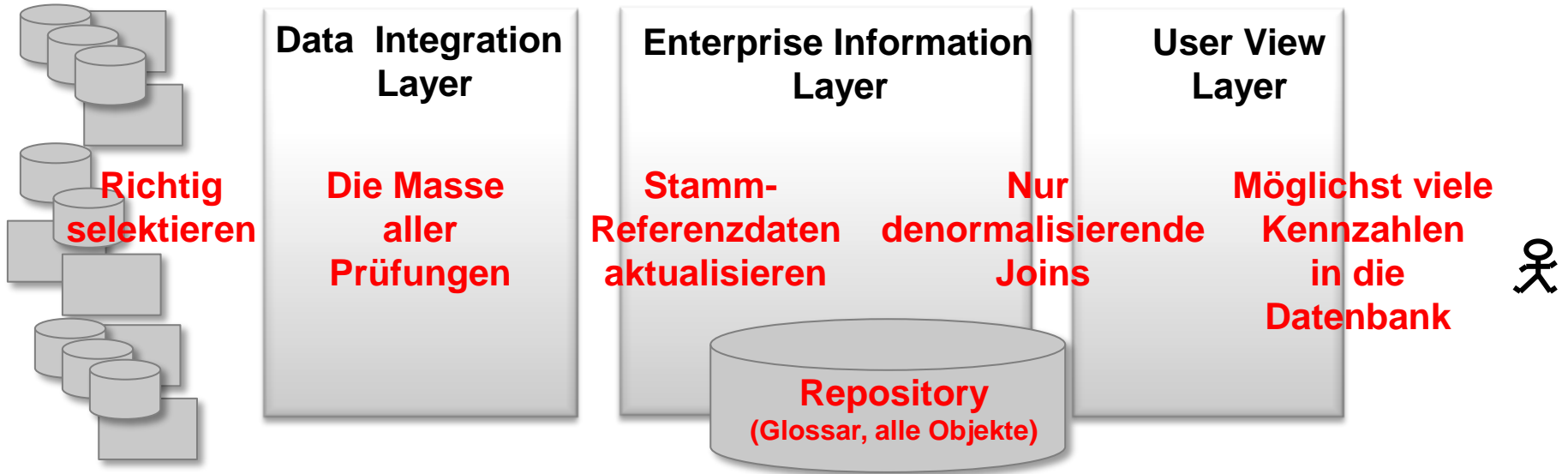
**Dimensionen  
Fakten  
Vor-  
berechnete  
Kennzahlen**



R: Referenztabellen  
T: Transfertabellen  
S: Stammdaten  
B: Bewegungsdaten  
D: Dimensionen  
F: Fakten

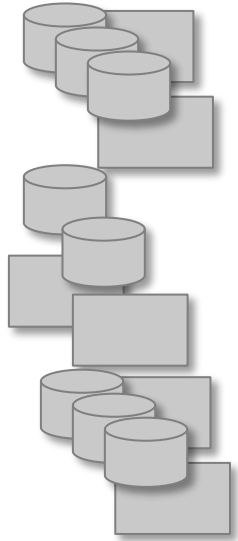


# Die Organisation des ETL-Prozesses



Für alle Aktionen den frühest möglichen Punkt finden

# Angemessen in den Situationen agieren



## Data Integration Layer

**Temporäre Daten**

## Enterprise Information Layer

**20% Volumen f. viele  
Kleine Tabellen**

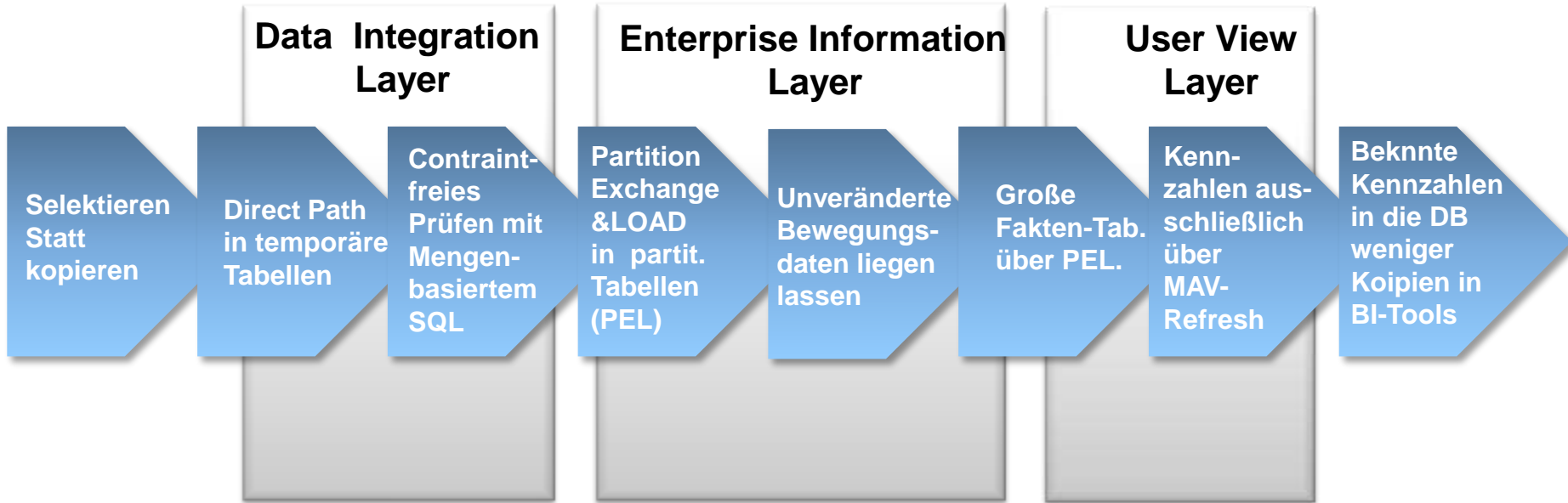
**80% Volumen f.  
wenige  
große Tabellen  
=> partitioniert**

## User View Layer

**Wieder  
herstellbare  
Daten**



# Verfahren für schnelles ETL in der Datenbank



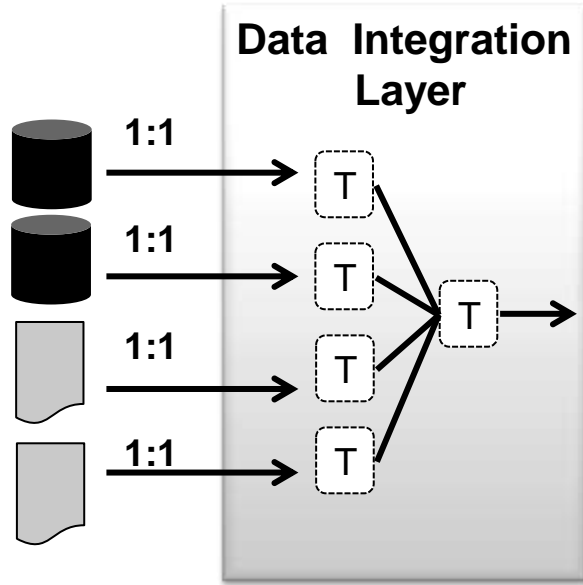
# Die Quellen



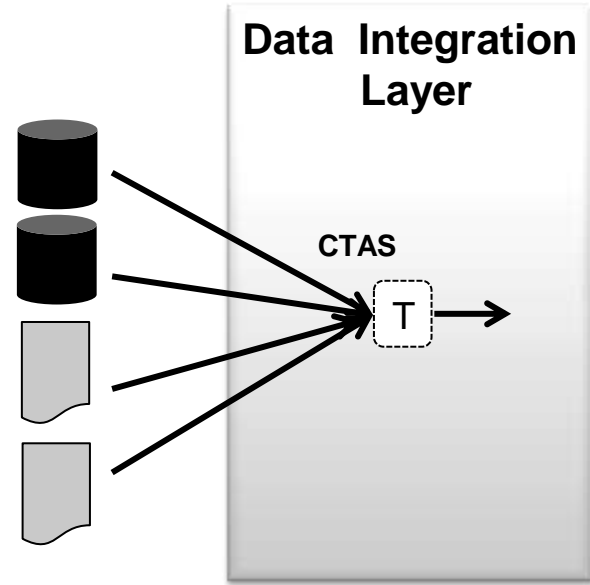
**Selektieren  
statt  
kopieren**

- Nur die Daten laden, die gebraucht werden
- Keine 1:1 Kopien in die Integrations-Schicht
- Nach Möglichkeit vor Eintritt in die Datenbank Prüfungen durchführen
- Bereits geprüfte Daten nicht mehr prüfen

# Nicht so ....sondern



Warum?



Logik so früh wie möglich

# Der „Einstieg“



## Direct Path Load in temporäre Tabellen

- **Direct Path Load**
  - Ohne Undo und Spacemanagement
  - Create Table As Select ..... (CTAS)
  - Insert /\*+ APPEND \*/ .....
  - SQLLoader oder External Table
- **Temporäre Tabellen**
  - Helfen bei der Organisation
  - können beliebig modifiziert werden
  - können beliebig gelöscht werden

# Direct Path



Direct Path in temporäre  
Tabellen

- **Direct Path Load**
  - Ohne Undo und Spacemanagement
  - Create Table As Select ..... (CTAS)
  - Insert /\*+ APPEND \*/ .....
  - SQLLoader oder External Table
- **Temporäre Tabellen werden neu aufgebaut**
  - können beliebig modifiziert werden
  - können beliebig gelöscht werden

# Es gibt 6 Prüf-Kategorien



## Attribut-bezogene Regeln

- A** {
1. Not Null / Pflichtfelder
  2. Formatangaben
    - a) numeric
    - b) Alphanumerisch
    - c) Date
    - d) Masken
  3. Div. Check Constraint
  4. Wertbereiche
    - Ober-/Untergrenzen / Wertelisten

## Satz-bezogene Regeln

- B** {
5. Abhängigkeiten von Werten in anderen Attributen desselben Satzes

## Satz-übergreifende Regeln

- C** {
6. Primary Key / Eindeutigkeit
  7. Aggregat – Bedingungen
    - a) Ober- Untergrenzen von Summen
    - b) Anzahl Sätze pro Intervall usw.
  8. Rekursive Zusammenhänge
    - Verweise auf andere Sätze derselben Tabelle (Relation)

## Tabellen-über greifende Regeln

- D** {
9. Foreign Key
    - a) Child-Parent (Orphan)
    - b) Parent-Child
  10. Aggregat – Bedingungen
    - a) Ober- Untergrenzen von Summen
    - b) Anzahl Sätze pro Intervall usw.
  11. Referenz-Zusammenhänge
    - Verweise auf Sätze einer anderen Tabelle (Relation)

## Zeit/ Zusammenhang-bezogene Regeln

- E** {
12. Zeitinvariante Inhalte (z. B. Anz. Bundesländer)
  13. Zeitabhängige Veränderungen
  14. Über die Zeit mit anderen Daten korrelierende Feldinhalte

## Verteilungs-/Mengen-bezogene Regeln

- F** {
15. Verteilung
    - a) Arithmetische Mittel
    - b) Varianz / Standardabweichungen
  16. Qualitätsmerkmale und Mengen



# Mengen-basierte Prüfungen mit SQL

## Attribut-bezogene Regeln

- A {
1. Not Null / Pflichtfelder
  2. Formatangaben
    - a) numeric
    - b) Alphanumerisch
    - c) Date
    - d) Masken
  3. Div. Check Constraint
  4. Wertbereiche
    - Ober-/Untergrenzen / Wertelisten

```
select
  bestellnr,
  case
    when -- wenn Feld BESTELLNR nicht numerisch
      REGEXP_LIKE(BESTELLNR, '[^[:digit:]]')
    then 1
    else 0
  End Num_Check_bestellnr
from bestellung;
```

```
select
CASE
  WHEN (F1 = 3 and F2 = F3 + F4)
  then 1
  ELSE 0
end
from fx
```

## Satz-bezogene Regeln

- B {
5. Abhängigkeiten von Werten in anderen Attributen desselben Satzes

## Satz-übergreifende Regeln

- C {
6. Primary Key / Eindeutigkeit
  7. Aggregat – Bedingungen
    - a) Ober- Untergrenzen von Summen
    - b) Anzahl Sätze pro Intervall usw.
  8. Rekursive Zusammenhänge
    - Verweise auf andere Sätze derselben Tabelle (Relation)

```
insert /*+ APPEND */ into err_non_unique_bestellung
select bestellnr from
  (select
    count(bestellnr) n,
    bestellnr
  from bestellung
  group by bestellnr)
where n > 1;
```

# Umgang mit SQL und PL/SQL im DB-ETL

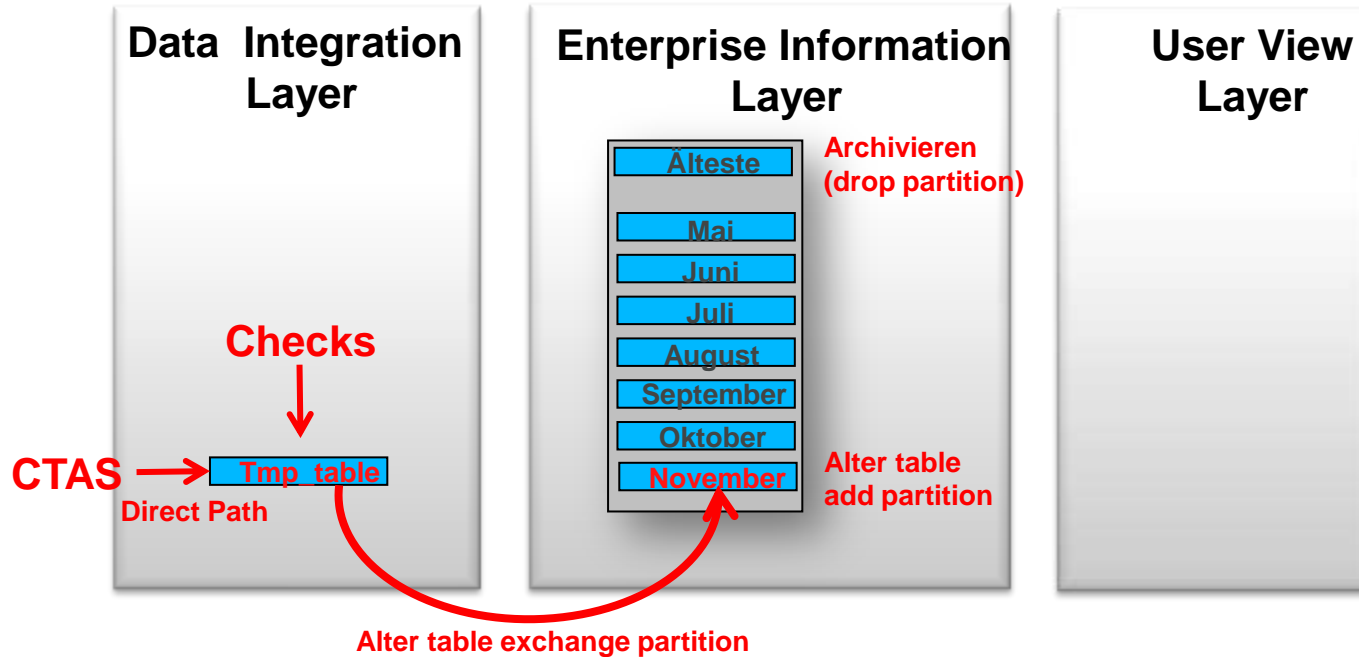
## So nicht ...

```
Create or replace procedure Proc_A
V1      number;
V2      number;
V3      varchar2;
V4      varchar2;
....
Cursor CS as select s1,s2 from tab_src;
Begin
  open CS;
  loop
    fetch CS into v1,v2,...;
    select f1 into v3 from tab1;
    select f1 into v4 from tab2;
    insert into Ziel _tab s1,s2,s3,s4
      values(v1,v2,v3,v4);
  end;
end;
```

## Aber z. B. so ...

```
insert into ziel
  select f1, f2, f3, f4 from
  (with CS as
    select s1 v1,s2 v2 from tab_src
  Select tab1.f1 f1 ,tab2.f2 f2, CS.s1 f3,CS.s2 f4
    from tab1,tab2,CS
  Where...
);
```

# Exchange Partition



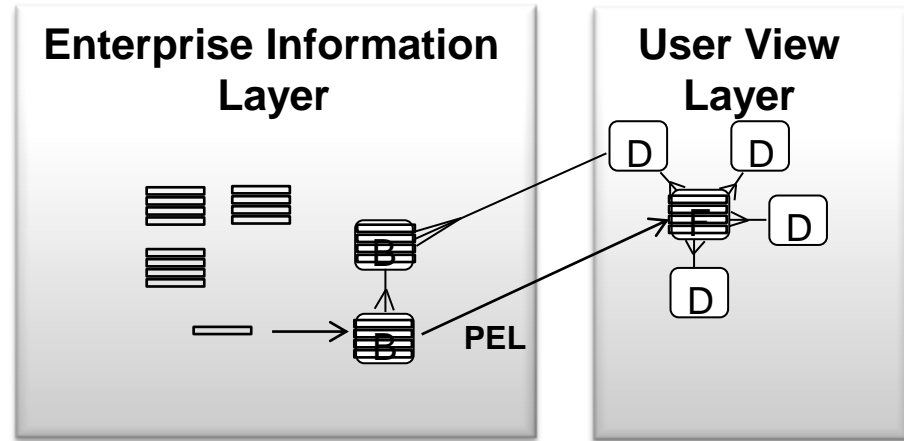
# Minimales Bewegen



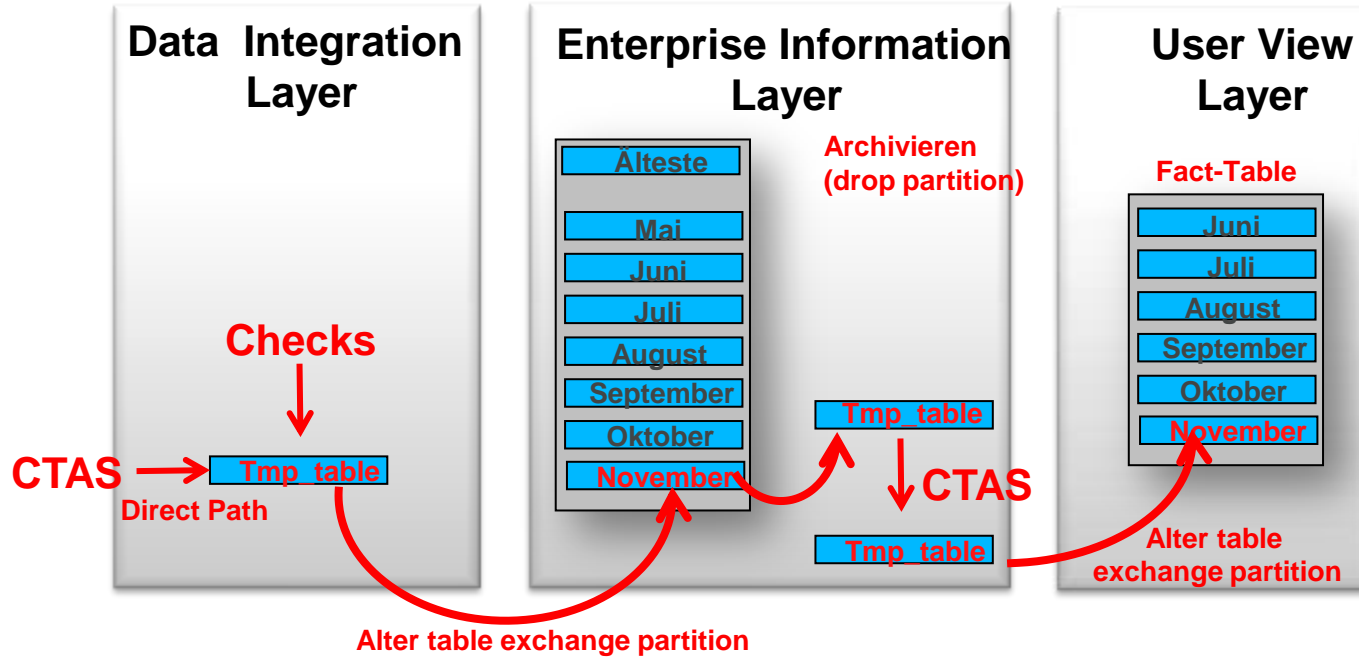
**Große unveränderte  
Tabellen liegen lassen**

**Zugriff auf beide  
Schichten**

**Security mit BORDMittel  
anstatt durch Kopieren  
lösen**



# Aufbau Fakten-Tabellen





# Der Weg in die BI-Tools

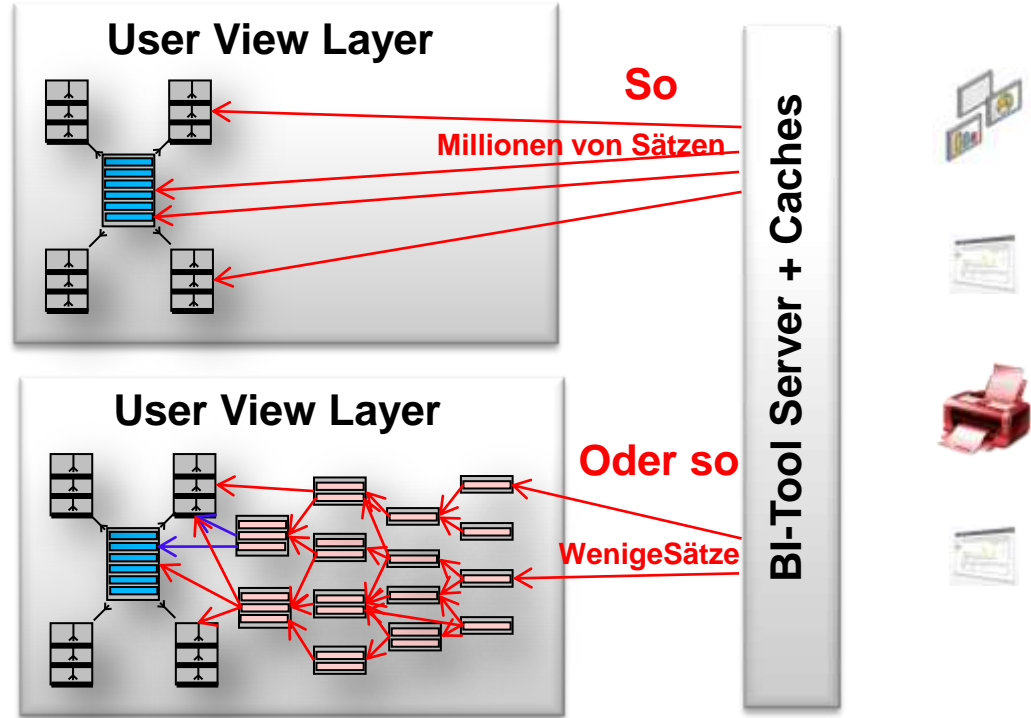


So viel wie möglich in der DB vorbereiten

Verhindert unnötiges Kopieren

Keine Verlagerung von Pseudo-ETL in die BI-Tools

Standardisierte Kennzahlen



# Weitere Einflussfaktoren und Techniken

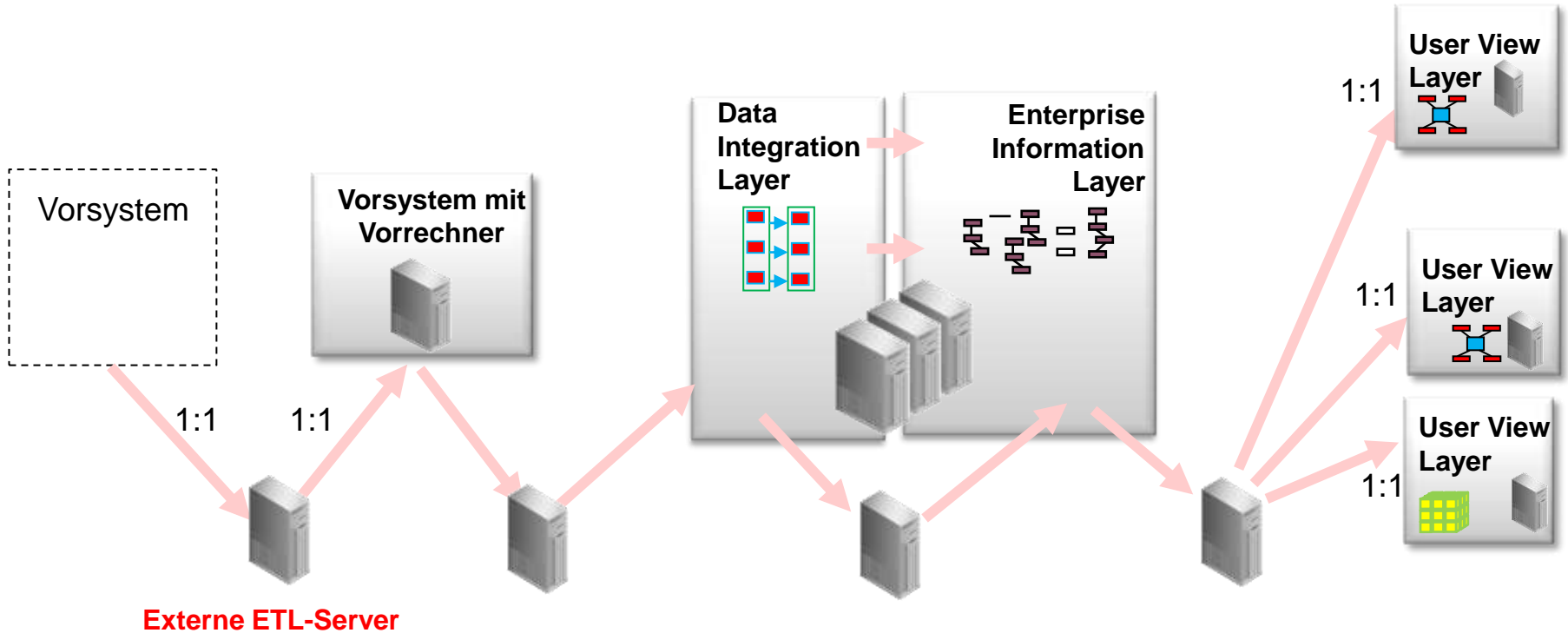
- **Parallelisierung** -> abhängig von Hardware  
-> direktes Steuern über Hints
- **Aktuelle Statistiken** -> Source Tabellen  
-> auch während des ETL-Laufes
- **Ausnutzen des Cache-Effektes**  
-> Organisieren der Abarbeitungsreihenfolge  
-> Eventuell Query-Result-Cache nutzen
- **Vermeiden von großen Join-Tabellen**  
-> eventuell kleine Join-Tabelle mit wenigen Spalten



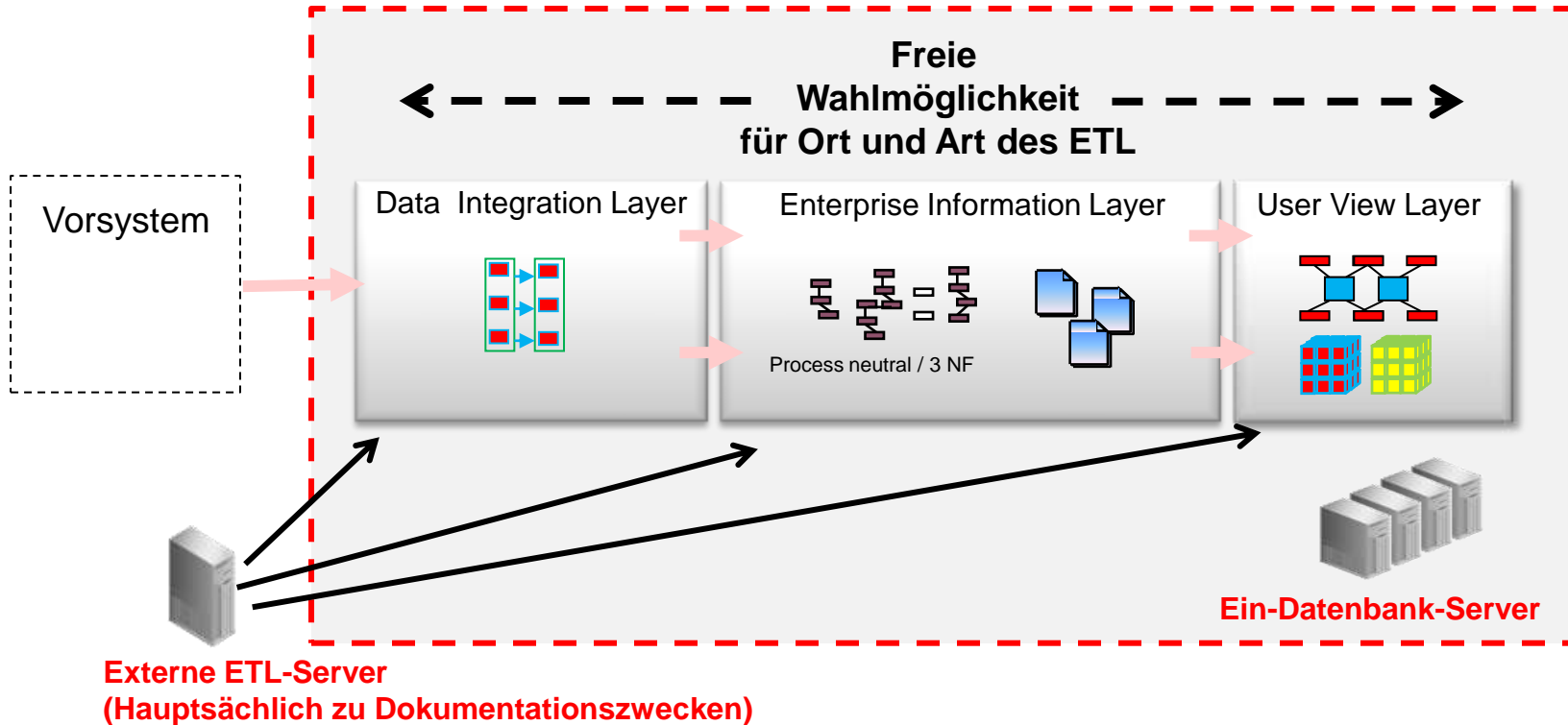
# Weitere Einflussfaktoren und Techniken

- **Schnelle mengen-basierte Prüfungen kommen zuerst, teure Prüfungen (Feld-Prüfungen) zuletzt durchführen**
- **Möglichst viel Hauptspeicher für Join-Operationen mit großen Tabellen**
- **Sort-Area-Size hoch setzen**
- **Blocksize auf 16 bzw. 32 K**
- **PCTfree auf Null setzen**
- **Partition Change Tracking (PCT) für inkrementelles Refresh der MAVs**

# Keine unnötige Daten-Transporte ...

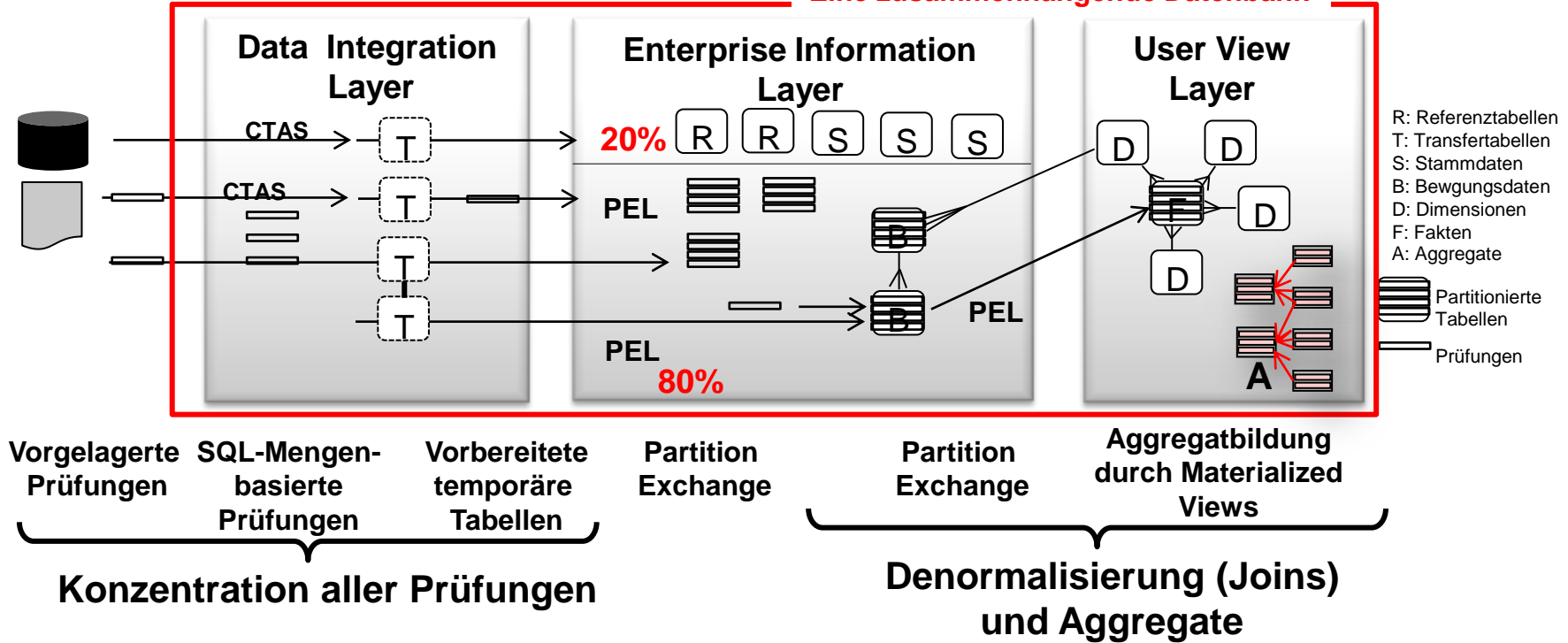


# ... sondern kurze Wege



# Zusammenfassung der wichtigsten Techniken

Eine zusammenhängende Datenbank



ORACLE®

---

**DATA WAREHOUSE**