



**ORACLE®**

## **Für Querdenker – Was ODI anders macht als OWB und umgekehrt**

**Bianca Stolz, Negib Marhoul**  
Systemberater ORACLE Deutschland B.V. & Co. KG

2011  
**DOAG**  
Konferenz + Ausstellung

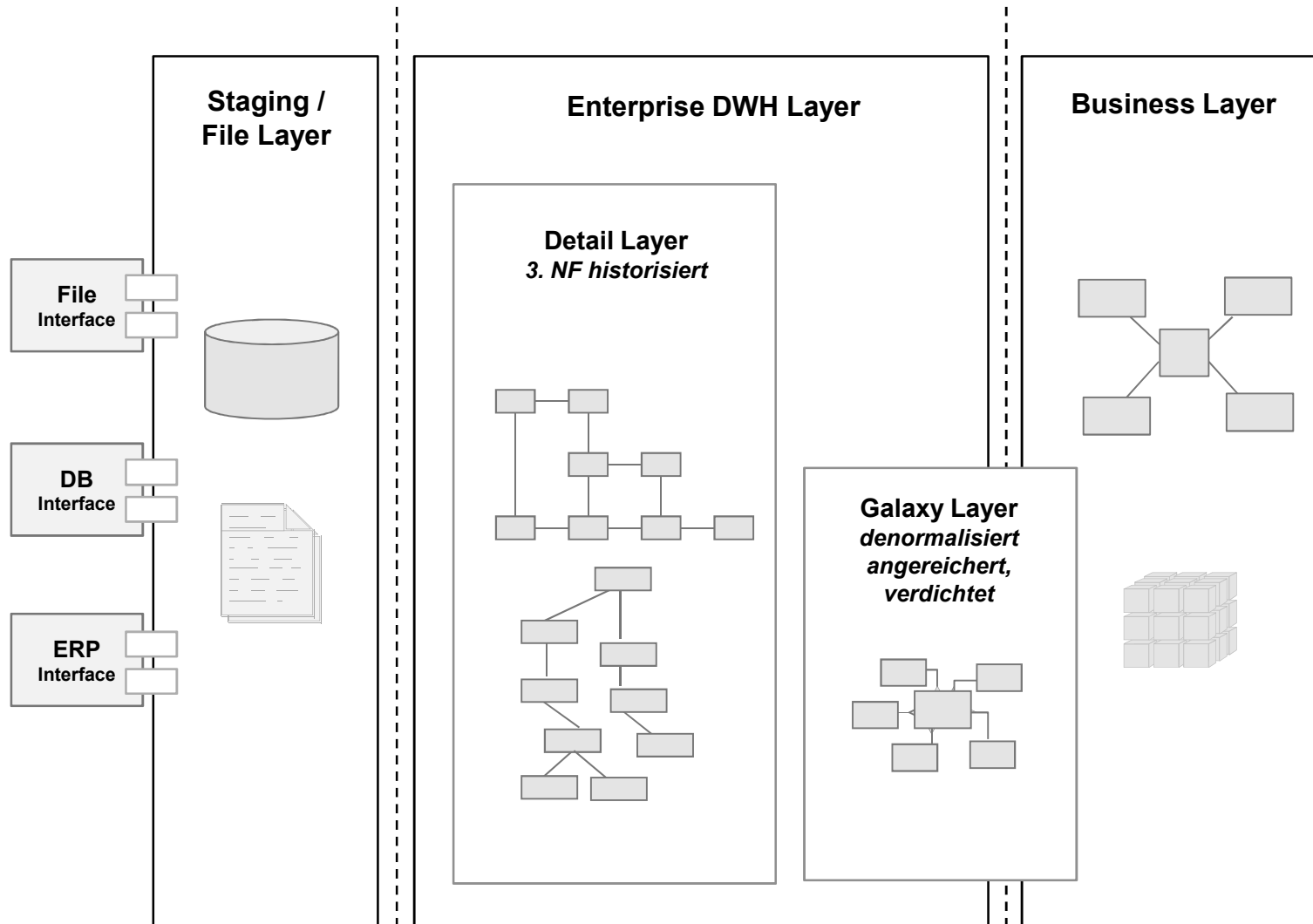
DOAG Konferenz Nürnberg, 17.November 2011



# Agenda

- Charakteristika von OWB und ODI
- Datenanbindung
  - Physical / Logical Schema und Context
  - ODI Staging Area
- Datenqualität
  - Static Control und Flow Control
  - Einfluss der Knowledge Modules
- Beispiel: Prüfung auf Eindeutigkeit in OWB und ODI
- Laufzeitinformationen und Verwendung von PL/SQL
- Resümee und Handout

# Das 3-Schichten-Referenzmodell



# Charakteristika von OWB und ODI

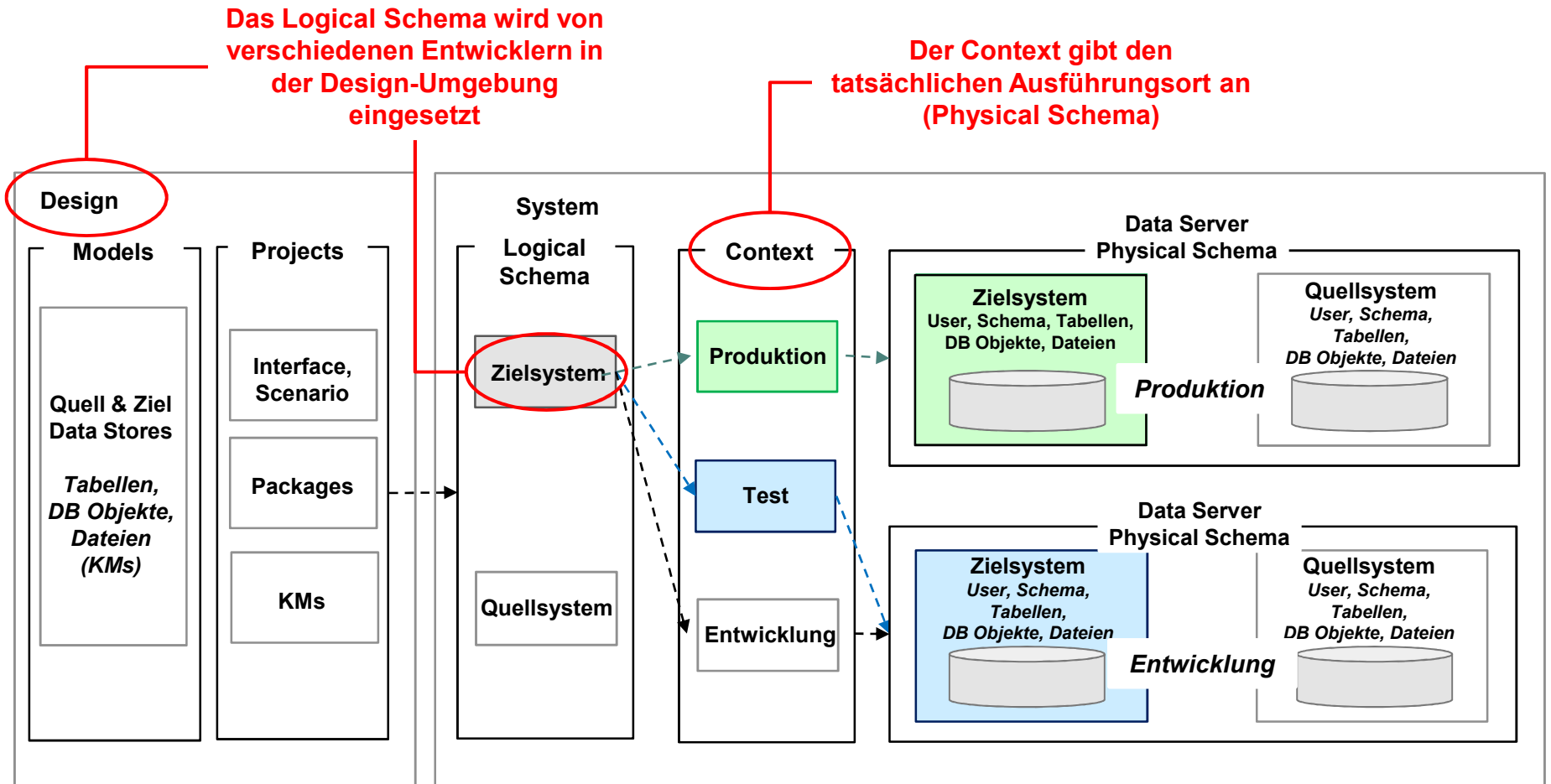
Oracle Warehouse Builder	Oracle Data Integrator
<b><i>In-Database ELT</i></b> Alle Transformationen werden möglichst nah am endgültigen Datenziel ausgeführt – am besten im DWH selbst	<b><i>„Pluggable“ ELT-Architektur</i></b> Der Ausführungsort von Transformationen kann beliebig bestimmt werden – auf Datenquelle oder -ziel
<b><i>„Oracle Optimized“</i></b> Sämtliche Operationen sind besonders auf die verfügbaren Funktionen in der Oracle Datenbank abgestimmt	<b><i>Nativer Zugriff: Knowledge Modules</i></b> Knowledge Modules kapseln die Logik, um auf „Datenserver“ oder „Data Stores“ unterschiedlicher Technologien zuzugreifen (non-Oracle)
<b><i>Datenmodellierung</i></b> Der Datenfluss wird grafisch mit diversen Operatoren feingranular modelliert, so dass der Verlauf der Daten nachvollziehbar ist	<b><i>Convention over Configuration</i></b> Immer gleichbleibende Schritte im Fluss der Datenbewegung werden gekapselt und nur spezielle Bedingungen manuell konfiguriert
<b><i>PL/SQL Code</i></b> Die ETL-Operationen werden in solche Anweisungen umgesetzt, die eine Oracle Datenbank effizient verarbeiten kann	<b><i>SQL mit Erweiterungen</i></b> Auszuführender Code wird in Standard-SQL erzeugt, per Knowledge Module kommen Erweiterungen für jeweilige Technologie hinzu



# Datenquellen und -ziele anbinden

- Im OWB werden sämtliche „Datenserver“ oder „Data Stores“ über Locations definiert
  - Müssen für jedes Project extra eingebunden werden
  - Über Multi-Cofiguration kann zwischen verschiedenen „Grundeinstellungen“ gewechselt werden
- ODI unterscheidet Physical und Logical Schema
  - Physical = Reale Verbindung zum Datenschema
    - Name, User, Passwort zum echten Datenbankschema
  - Logical = „Virtuelle“ Verbindung zum Datenschema
    - Name für ein Datenschema, universell beim Design verwendbar
  - Context bestimmt den endgültigen Ausführungsort
  - Datenserver lassen sich Project-übergreifend verwenden

# Physical/Logical Schema und der Context





# Datenflüsse modellieren

- Im OWB werden die Daten in einem Mapping kontinuierlich in einem Strom weiterbewegt
  - Vorgefertigte Operatoren für sämtliche SQL-Operationen
  - Sicht auf alle Einzelschritte eines kompletten Mappings
- Beim ODI Interface ist das Mapping stark gekapselt
  - Wenige vordefinierte Operatoren
  - SQL-Operationen lassen sich als Expressions hinzufügen
  - Nur eine Zieltabelle je Mapping möglich
- Design Best Practice: Mappings übersichtlich halten
  - Komplexität des Mappings begrenzen
  - Bei unerwartetem Abbruch eines Mappings ist die Fehlersuche einfacher



# ODI Staging Area (Work Schema)

- Eine dedizierte Staging Area im Rahmen des 3-Schichten-Modells kann im OWB z.B. über ein eigenes Project umgesetzt werden
- Der ODI verwendet eine separate „Staging Area“
  - Bezeichnung für den Ablageort der temporären Tabellen während der Verarbeitung der ETL-Prozesse
  - Physische Festlegung der ODI Staging Area über das sog. Work Schema
  - Nicht unbedingt identisch mit der Staging Area im Referenzmodell (!)
- Die ODI Staging Area kann sich auf einem beliebigen System befinden
  - Voraussetzung: Festlegung auf einem RDBMS





# Daten nachträglich prüfen (Static Control)

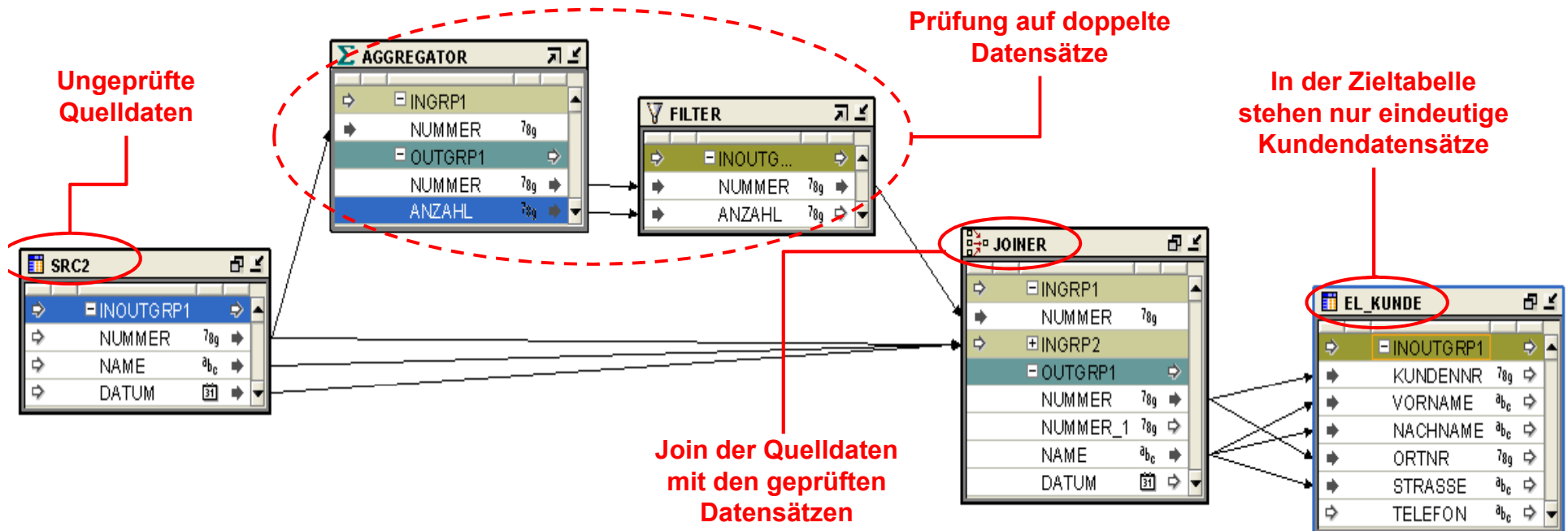
- Die Lage der ODI Staging Area hat Einfluss darauf, welche Funktionen des verwendeten Check Knowledge Module (CKM) zur Verfügung stehen
- Zur Überprüfung von Constraints auf die geladenen Daten wird der Parameter „Static Control“ auf TRUE gesetzt
  - Prüfung eines zuvor definierten Constraints nach dem Laden
  - Als Prüfkriterium kommen nicht nur Constraints in Frage, die in der Datenbank existieren, sondern auch solche, die auf ODI-Level definiert wurden
- Static Control ist auf einzelne Tabellen sowie auf einen kompletten Data Store anwendbar



# Daten unterwegs validieren (Flow Control)

- Befindet sich die ODI Staging Area auf demselben System wie das Datenziel, können Daten über Flow Control während des Ladevorgangs geprüft werden
  - Diese Funktion verhindert, dass falsche Daten überhaupt erst geladen werden
- Die Verwendung von Flow Control bedingt die Definition eines Update Keys
  - Definition über eine oder mehrere Spalten zur Eindeutigkeit
  - Kann auch nur für ein spezielles Interface festgelegt werden
- Flow Control entspricht einer Art Data Profiling und dient dem Einhalten von Business Rules in den Daten

# Beispiel: Prüfung auf Eindeutigkeit (OWB)



```
INSERT INTO el_kunde (kundenr,vorname,nachname,ortnr,strasse,telefon)
SELECT src2.nummer,src2.name,src2.name,src2.nummer,src2.name,src2.nummer
FROM SRC2,(SELECT nummer FROM
            (SELECT count(nummer) n, nummer
             FROM src2 group by nummer)
            WHERE n = 1) eindeutig
WHERE src2.nummer = eindeutig.nummer;
```

# Beispiel: Prüfung auf Eindeutigkeit (ODI)

## 1. Aggregation der doppelten Promotion IDs

The screenshot displays the ODI Mapping Properties window for a virtual target data store named 'AGG\_STG'. The source is a table '1 - SRC\_PROMOTION (SRC\_PROMOT)' with columns including \*PROMO\_ID, \*PROMO\_NAME, \*PROMO\_SUBCATEGORY, \*PROMO\_SUBCATEGORY\_ID, \*PROMO\_CATEGORY, \*PROMO\_CATEGORY\_ID, \*PROMO\_COST, \*PROMO\_BEGIN\_DATE, \*PROMO\_END\_DATE, \*PROMO\_TOTAL, \*PROMO\_TOTAL\_ID, and SR\_KEY. The target data store has an indicator 'CNT' with the mapping 'COUNT(SRC\_PROMOTION.PROMO\_ID)'. The source column 'PROMO\_ID' is mapped to the target column 'PROMO\_ID'. The mapping is defined as 'COUNT(SRC\_PROMOTION.PROMO\_ID)'.

**Annotations:**

- Virtueller Target Data Store:** Points to the 'AGG\_STG' data store.
- Tabelle mit Quelldaten:** Points to the '1 - SRC\_PROMOTION (SRC\_PROMOT)' source table.
- COUNT auf die vorkommenden Promotion IDs in den Quelldaten:** Points to the 'COUNT(SRC\_PROMOTION.PROMO\_ID)' mapping function.

**Mapping Properties:**

Active Mapping:

Implementation:  Technical Description:  Business Rule:

Implementation: COUNT(SRC\_PROMOTION.PROMO\_ID)

# Beispiel: Prüfung auf Eindeutigkeit (ODI)

## 2. Filter auf Promotions über Anzahl der Promotion IDs

Einbindung des eben erstellen (temporären) Interfaces

Filterkriterium

Join der eindeutigen Promotion IDs

1 - SRC\_PROMOTION (SRC\_PROMO1)

*PROMO_ID
*PROMO_NAME
*PROMO_SUBCATEGORY
*PROMO_SUBCATEGORY_ID
*PROMO_CATEGORY
*PROMO_CATEGORY_ID
*PROMO_COST
*PROMO_BEGIN_DATE
*PROMO_END_DATE
*PROMO_TOTAL
*PROMO_TOTAL_ID
SR_KEY

Filter - Property Inspector

Filter Properties

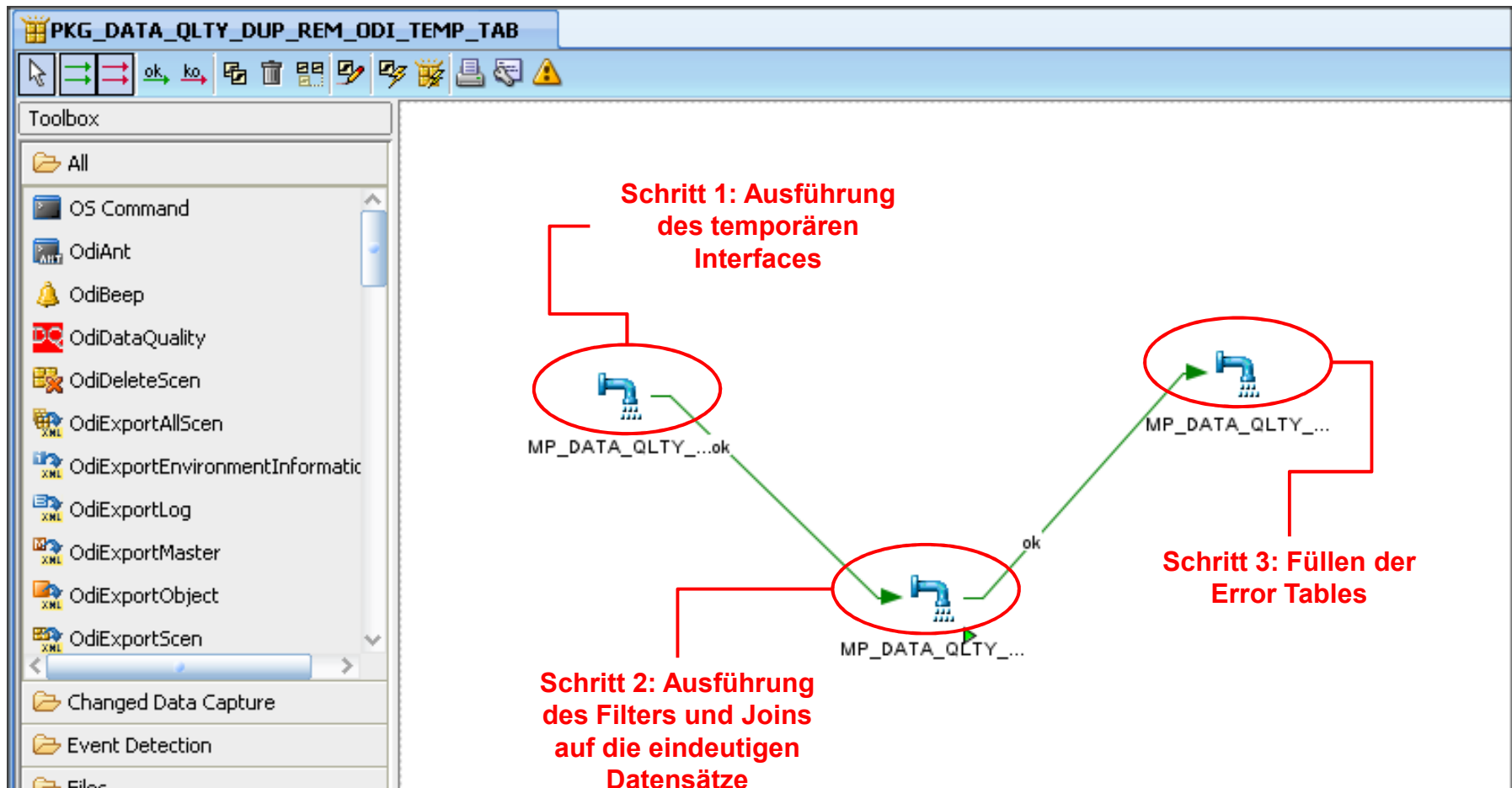
Active Filter:

Implementation Technical Description Business Rule

TEST\_TEMP.CNT = 1

# Beispiel: Prüfung auf Eindeutigkeit (ODI)

## 3. Komplettes Package für Prüfung auf Eindeutigkeit





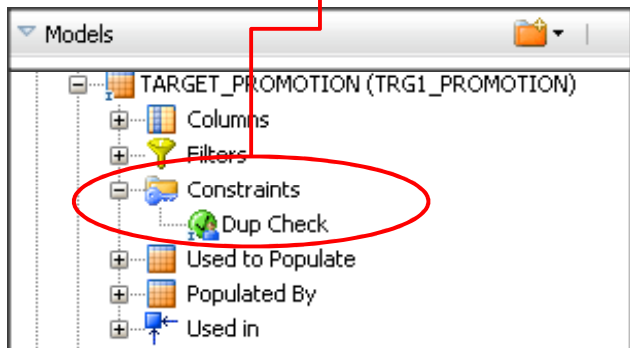
# Einfluss der Knowledge Modules

- Knowledge Modules kapseln die Logik bestimmter Funktionen im ETL-Prozess, z.B. Laden oder Integrieren der Daten
- Single-Tech Knowledge Modules
  - Werden bei Operationen verwendet, die auf der gleichen Technologie stattfinden (z.B. Oracle ↔ Oracle)
  - Sowohl Static als auch Flow Control verfügbar
- Multi-Tech Knowledge Modules
  - Für Operationen zwischen verschiedenen Technologien
  - Nur Static Control verfügbar
- Knowledge Modules können angepasst bzw. umgeschrieben werden
  - Reihenfolge der abzuarbeitenden Schritte
  - Code der Knowledge Modules

# Beispiel: Prüfung auf Eindeutigkeit (ODI)

## 1. Prüfung auf Eindeutigkeit mit Flow Control

Constraint zur Prüfung auf Eindeutigkeit  
(Project-übergreifend einsetzbar)



Parameter Flow Control  
auf TRUE gesetzt

The screenshot shows the 'Target Properties' window for 'Target (DWH - SALES)'. The 'Target Area - Property Inspector' is open, showing the 'Options' section. The 'FLOW\_CONTROL' parameter is circled in red and set to '<default>:true'. The 'IKM Selector' is set to 'IKM Oracle Incremental Update'. The 'Distinct Rows' checkbox is unchecked. The 'Options' table is as follows:

Name	Value
INSERT	<default>:true
UPDATE	false
COMMIT	<default>:true
SYNC_IRM_DELETE	<default>:true
FLOW_CONTROL	<default>:true
RECYCLE_ERRORS	<default>:false
STATIC_CONTROL	<default>:false
TRUNCATE	true

ODI Staging Area  
befindet sich auf dem  
Datenziel



# Beispiel: Prüfung auf Eindeutigkeit (ODI)

## 2. CKM für Flow Control und Vergleich mit OWB Expressions

Einbindung des Constraints in ein Interface

MP\_ODI\_DEFAULT\_DUP\_REM

Maximum number of errors allowed:   %

CKM Selector: CKM Oracle

Options:

Name	Value
DROP_ERROR_TABLE	true
DROP_CHECK_TABLE	<default>:false
CREATE_ERROR_INDEX	<default>:true
COMPATIBLE	<default>:9
VALIDATE	<default>:false
ENABLE_EDITION_SUPPORT	<default>:false
UPGRADE_ERROR_TABLE	<default>:false

Constraints:

Name	Value
Dup Check	<default>:true

Definition des Constraints in Verbindung mit den temporären Tabellen des ODI – mit fast dem gleichen Text könnte ein Expression Operator im OWB definiert werden

**Definition**

Condition [Model: First\_Model ▶ Datastore: TARGET\_PROMOTION]

Name: Dup Check

Type: Oracle Data Integrator Condition

Where (Use the table alias: TARGET\_PROMOTION):

```
TARGET_PROMOTION.PROMO_ID not IN (select PROMO_ID from |
( SELECT I$_TARGET_PROMOTION.PROMO_ID,count(I$_TARGET_PROMOTION.promo_id)
FROM I$_TARGET_PROMOTION group by I$_TARGET_PROMOTION.PROMO_ID HAVING COUNT(I$_TARGET_PROMOTION.PROMO_ID)>1))
```



# Laufzeitinformationen abfragen

- Im OWB werden Fehler u.a. in den DML Error Tables geloggt, Laufzeitinformationen finden sich in den Runtime Audit Tables (Präfix ALL\_RT\_)
- ODI schreibt sämtliche DML-Fehler für eine Zieltabelle jeweils in eine eigene Error Table (Präfix E\$)
- Alle Fehler einer Session werden in der zentralen SNP\_CHECK\_TAB Tabelle festgehalten
- Laufzeitinformationen können detailliert im ODI Operator auf unterschiedlichen Leveln abgerufen werden
  - Session, Step, Task Level
  - Nach Ausführungsdatum
  - Pro User

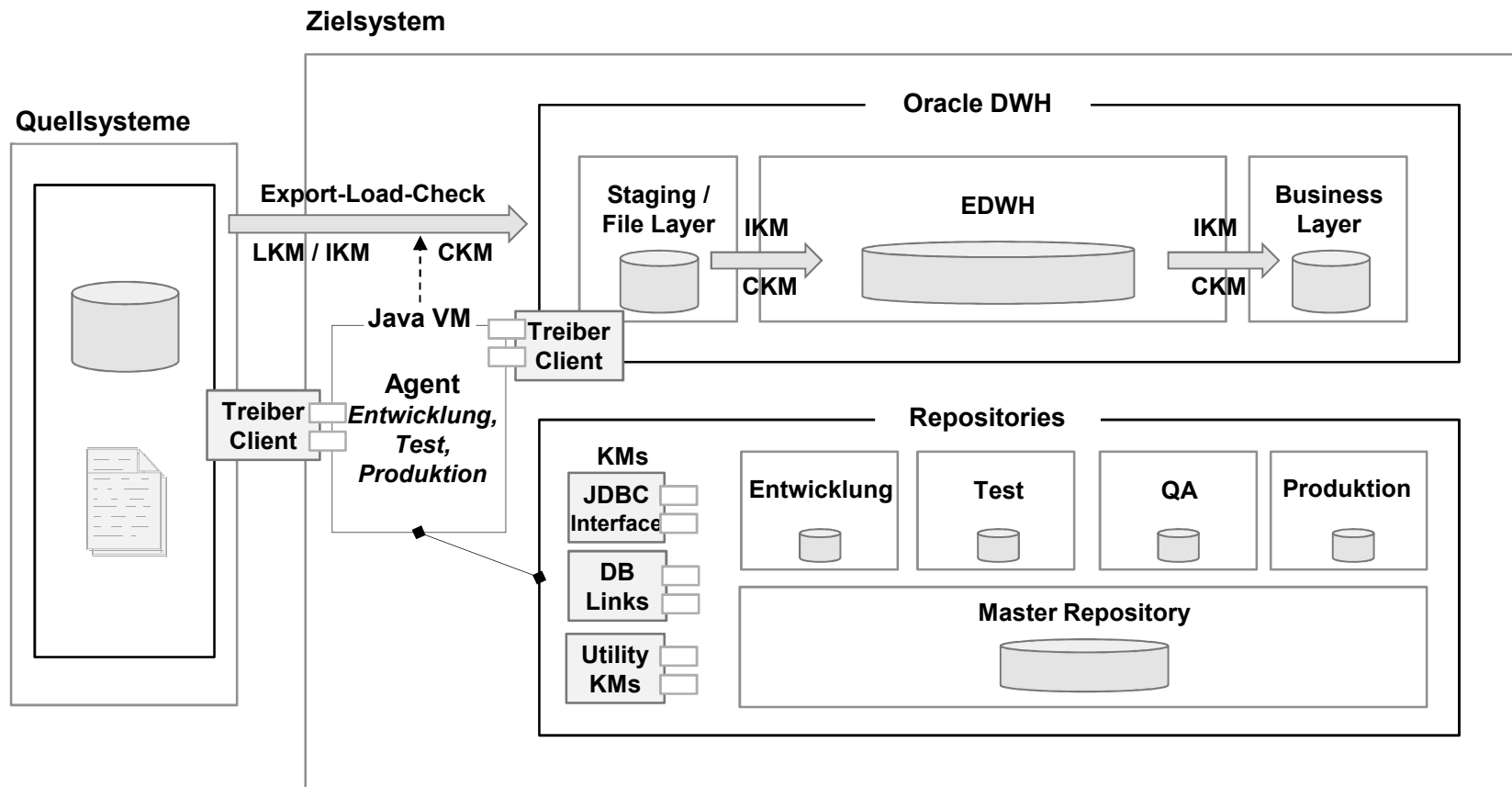


# Verwendung von PL/SQL

- OWB generiert standardmäßig alle Mappings als PL/SQL Packages
- ODI erzeugt zur Laufzeit reinen SQL Code
- Um mit ODI ebenfalls PL/SQL zu erzeugen, ist ein entsprechendes Knowledge Module erforderlich
- Die Generierung von nativem Code ermöglicht die Umsetzung der ETL-Prozesse auf verschiedensten Systemen (Technologien)

# ELT-Setup eines realen Anwendungsfalls

Umsetzung mit ODI





# Zuletzt: Resümee und Handout

- Die Entscheidung für ein Konzept (Referenzmodell) kommt vor der Entscheidung über ein ETL-Werkzeug
  - Werkzeuge unterstützen den systematischen Aufbau
- Design Best Practices gelten für beide Werkzeuge
  - Mappings bzw. Interfaces übersichtlich halten
  - Ein OWB Mapping würde im ODI möglicherweise mit mehreren Interfaces umgesetzt
- Die feingranulare Datenmodellierung im OWB steht der Kapselung gleichbleibender Operationen im ODI gegenüber
- Je nach gewünschter oder verfügbarer Funktion lässt sich mit dem ODI der Ausführungsort von Mappings, Filtern und Joins flexibel verlegen

Q&A

A 3D rendering of the letters 'Q&A' in a vibrant red color. The letters are thick and have a glossy finish, casting soft shadows on a light gray surface below them. The ampersand is positioned between the 'Q' and the 'A', and is rendered in a white color with a slight shadow, making it stand out against the red background of the letters.

**Vielen Dank für Ihre Aufmerksamkeit!**

**ORACLE®**

*Bei Fragen wenden Sie sich gern an uns:*

*[bianca.stolz@oracle.com](mailto:bianca.stolz@oracle.com)*

*[negib.marhoul@oracle.com](mailto:negib.marhoul@oracle.com)*