Alternative Fakten

Das Unbeeinflussbare beeinflussen

Stefan Winkler
- Geschäftsführer -
merlin.zwo InfoDesign GmbH & Co. KG
76228 Karlsruhe

Warum das Ganze?

merlin2wo InfoDesign

Weil …

Agenda

Teil 1: ***Beeinflussen des „Wie“*** **– schlechte Pläne**

- Überlisten des Metadaten-Katalogs
- Partitionieren ohne Partitionierung
- Fälschen von Ausführungsplänen

Teil II: ***Beeinflussen des „Was“*****– schlechtes SQL**

- Das Oracle SQL Translation Framework

Spitzenleistung heißt, sich auf seine Stärken zu konzentrieren.

merlin.zwo

Wir machen Oracle - nur Oracle.

Aus gutem Grund.

www.merlin-zwo.de

ORACLE Platinum Partner

Wir kümmern uns!

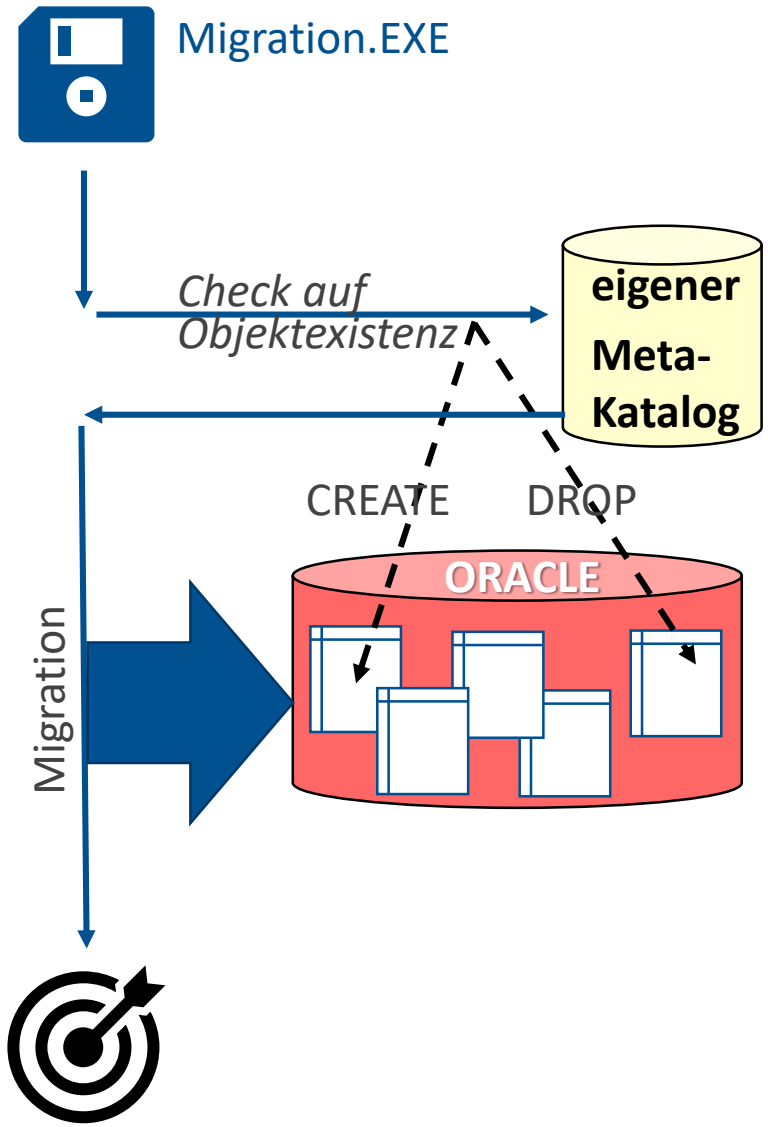
Überlisten des Metadaten-Katalogs

Migration Standard-Software mit riesigen Datenmengen (Finance Branche)

Fakten:

- Mig-Laufzeit > Wochende
- Analyse Probe-Migration offenbart fehlende Indizes bei UPDATEs/DELETEs

- Softwarehersteller
  - lehnt eigene Code-Änderungen ab
  - sichert „seine“ Objekte mit Checks in eigenem Meta-Datenkatalog…
    - Fehlendes wird generiert
    - Zusätzliches wird gedropt

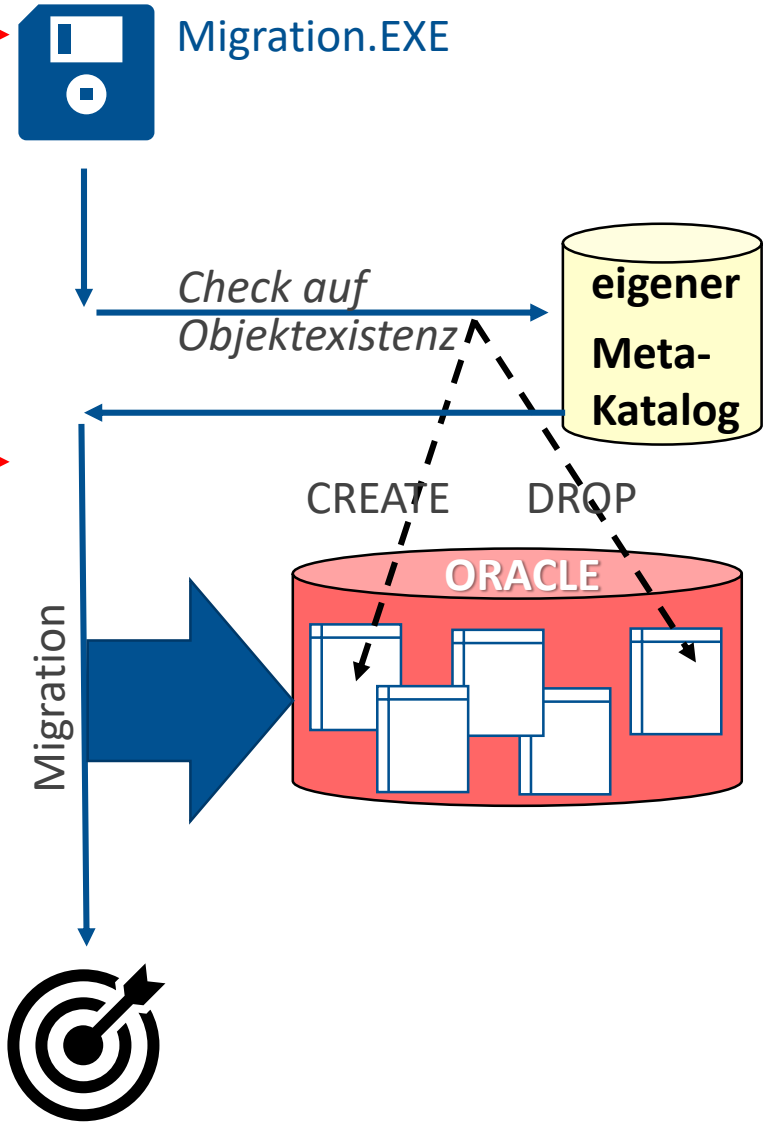


Überlisten des Metadaten-Katalogs

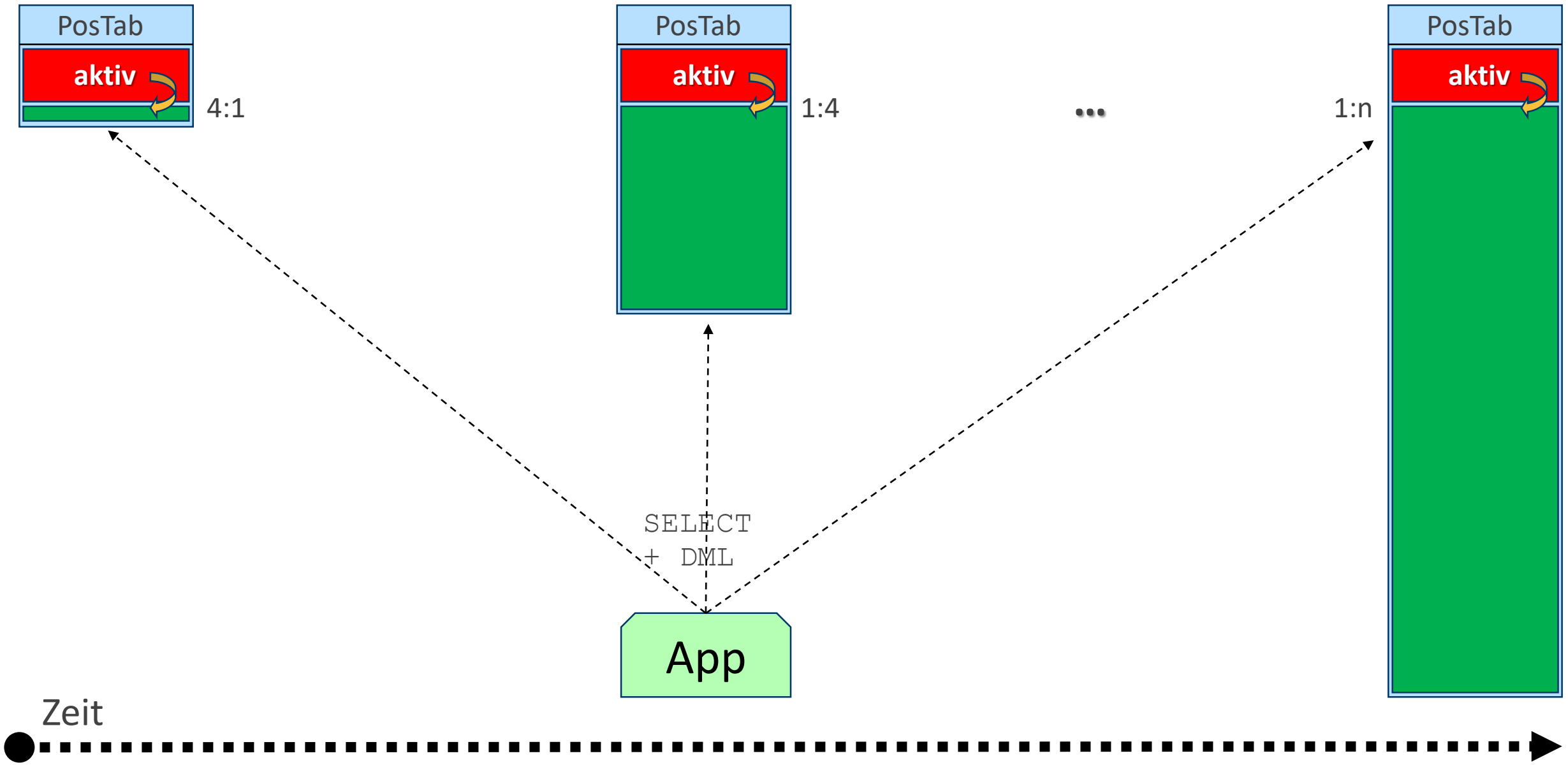
① `SYSTEM> CREATE INDEX SYSTEM.migspezial_insupd_idx ON applownerschema.tablename (columns…) INVISIBLE;`

② `SYSTEM> ALTER INDEX SYSTEM.migspezial_insupd_idx VISIBLE;`

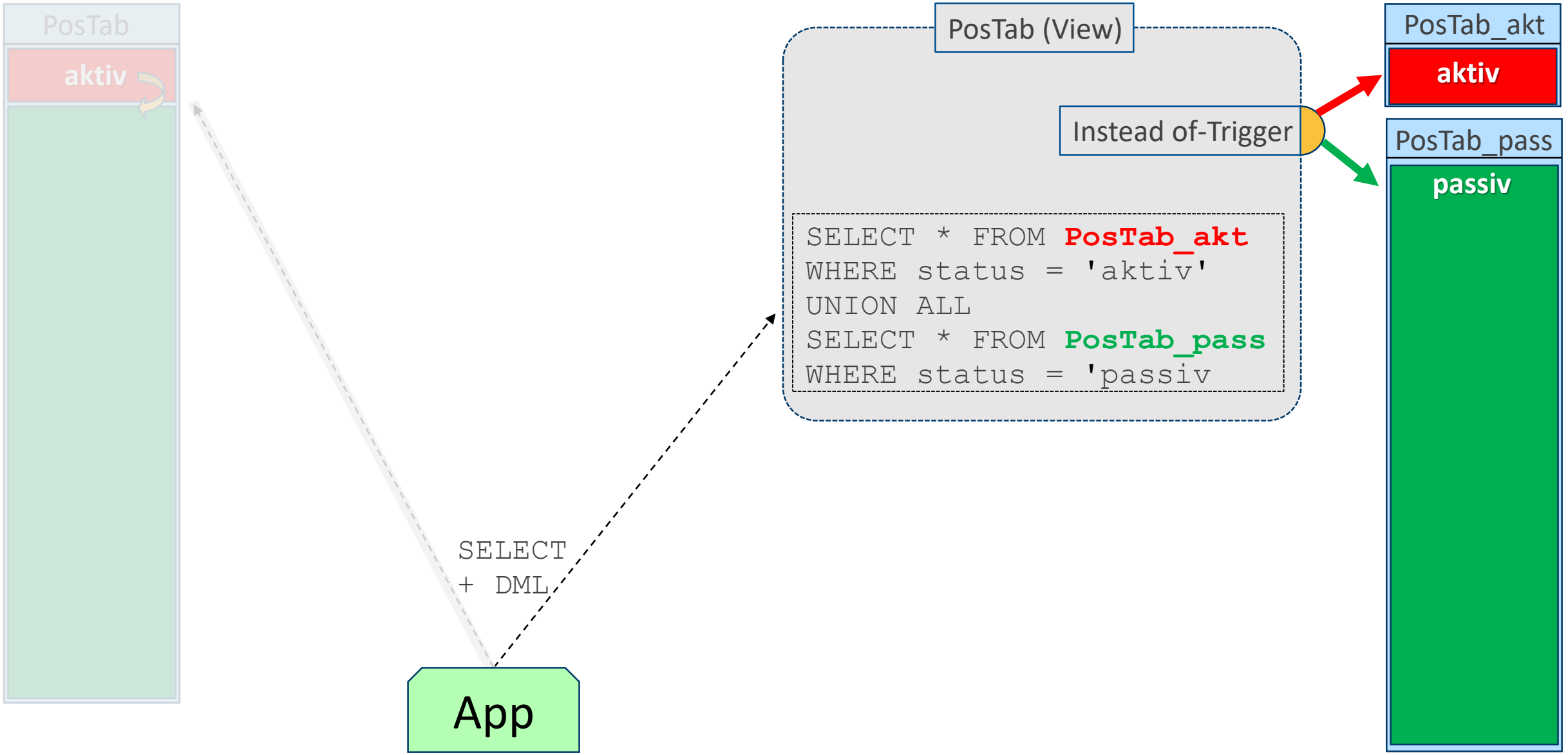
③ `SYSTEM> DROP INDEX SYSTEM.migspezial_insupd_idx;`



Überlisten von SELECTs und DML

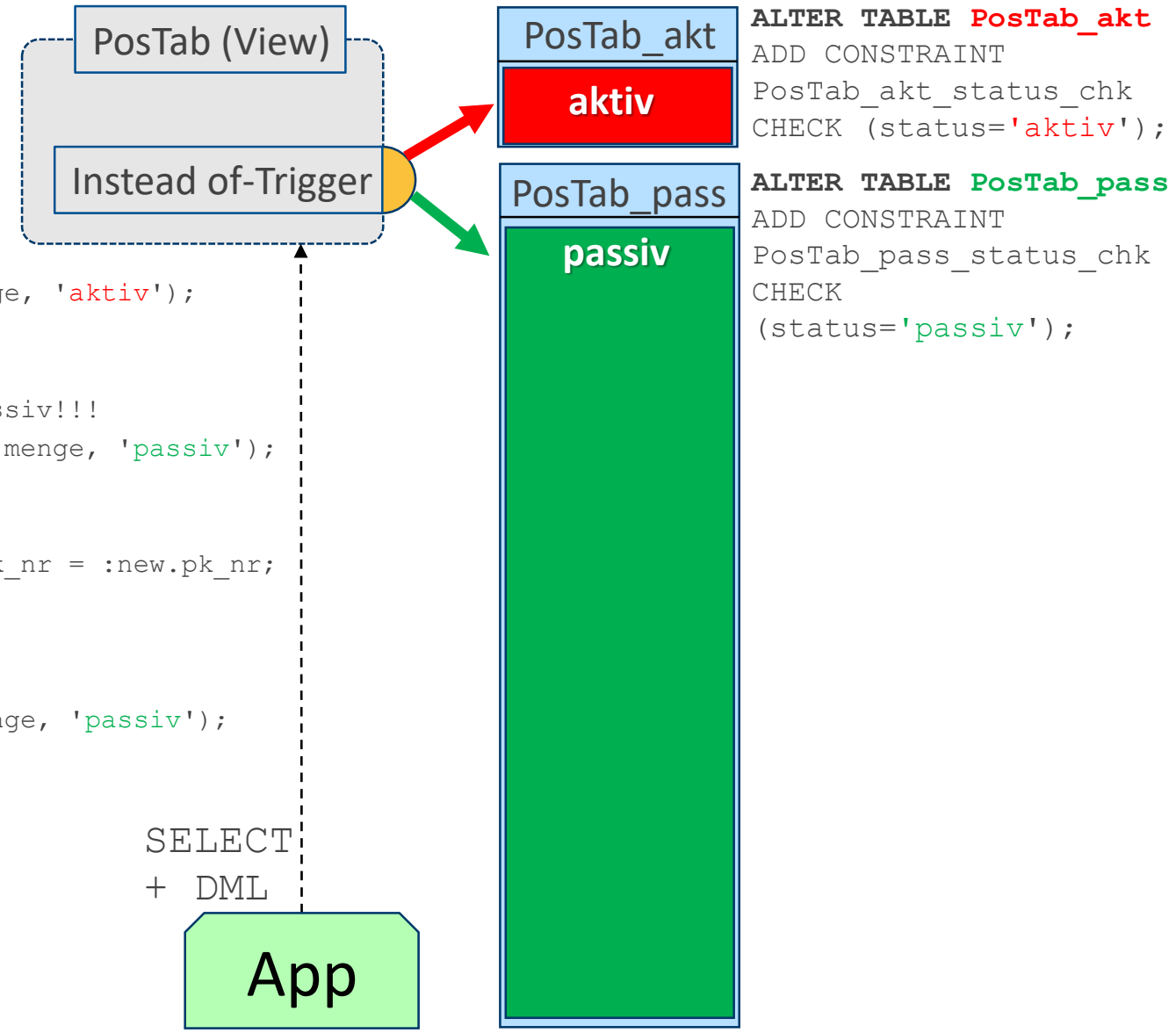


Überlisten von SELECTs und DML



Überlisten von SELECTs und DML

```
CREATE OR REPLACE TRIGGER PosTab_Trig
INSTEAD OF insert or update or delete ON PosTab
FOR EACH ROW
BEGIN
   IF inserting then
      insert into PosTab_akt values (:new.pk_nr, :new.menge, 'aktiv');
   END IF;
   IF updating then
      IF :new.status = 'passiv' then -- Wechsel aktiv=>passiv!!!
         insert into PosTab_pass values (:old.pk_nr, :old.menge, 'passiv');
         delete from PosTab_akt where pk_nr = :old.pk_nr;
      ELSE
         update PosTab_akt set menge = :new.menge where pk_nr = :new.pk_nr;
      END IF;
   END IF;
   IF deleting then
      insert into PosTab_pass values (:old.pk_nr, :old.menge, 'passiv');
      delete from PosTab_akt where pk_nr = :old.pk_nr;
   END IF;
END;
```

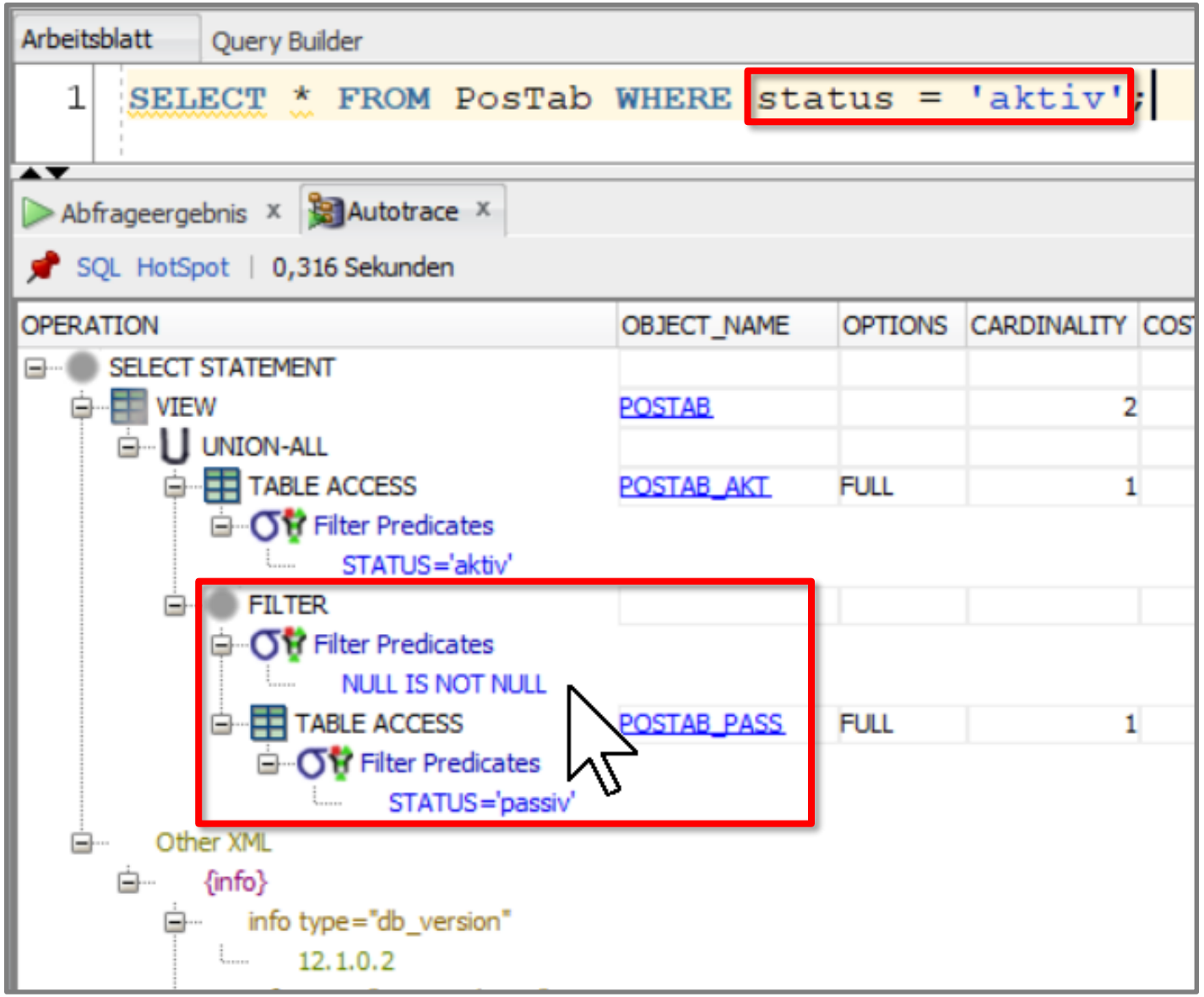



```
ALTER TABLE PosTab_akt
ADD CONSTRAINT
PosTab_akt_status_chk
CHECK (status='aktiv');
```

```
ALTER TABLE PosTab_pass
ADD CONSTRAINT
PosTab_pass_status_chk
CHECK
(status='passiv');
```

Überlisten von SELECTs und DML

CBO erkennt Prädikat „status“ richtig und filtert nicht passende Tabelle mit “NULL IS NOT NULL“ weg



Austausch der Execution Pläne

Austausch der Execution Pläne

merlin2wo InfoDesign

Oracle Mechanismen zur direkten Manipulation von Execution Plänen

merlin2 InfoDesign

SQL Profiles

- vom Tuning Advisor vorgeschlagen
- basieren meist auf Anpassung der Kardinalitäten an die Realität (dbms_sqltune)

Enterprise Edition
Tuning Pack
Diagnostic Pack

SQL Plan Baselines

- mehrere Plan-Varianten möglich
- automatisierbares System zur Akzeptanz und Deployment (dbms_spm)

Enterprise Edition

SQL Patches

- Teil des Repair Advisors (dbms_diag)
- fügen hints zu SQL-Befehlen

Enterprise Edition

merlin2wo InfoDesign

SQL Profiles

- vom Tuning Advisor vorgeschlagen
- basieren meist auf Anpassung der Kardinalitäten an die Realität

Enterprise Edition

Tuning Pack

Diagnostic Pack

Beispielaufgabe: "*Ersetze den Execution Plan für einen Befehl X mit einer getunten Variante*" – 100% transparent für Anwendung und Softwareanbieter.

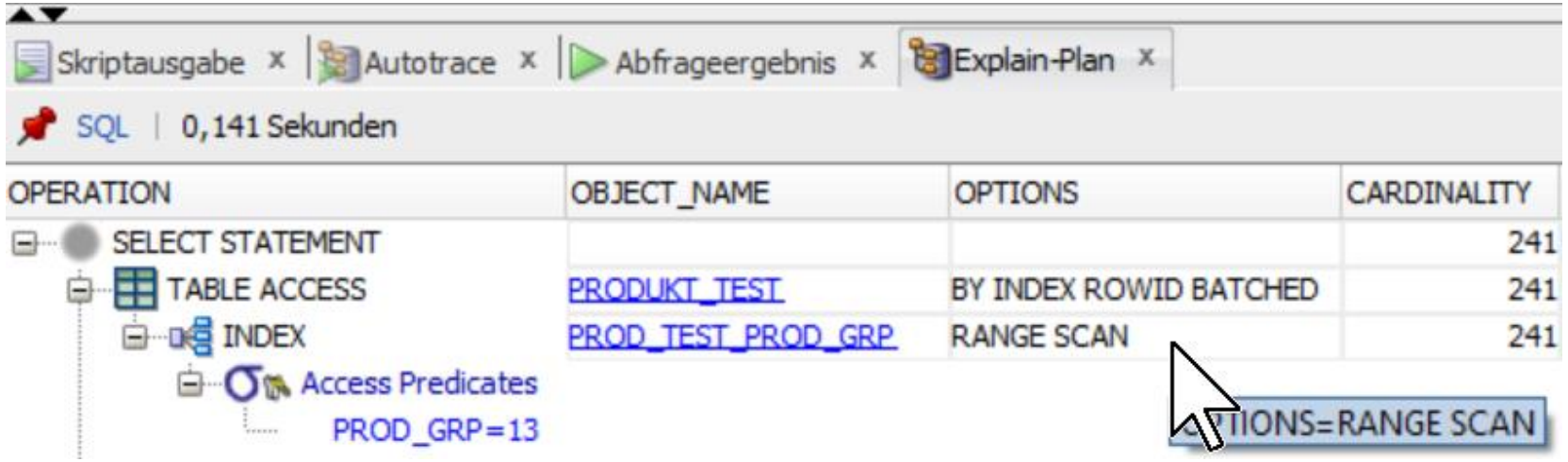
Beispielcode:

```
CREATE table produkt_test AS
SELECT 1000000+rownum                              AS prod_nr,
       sysdate-trunc(dbms_random.value(0,90)) AS datum,
       trunc(dbms_random.value(0,25))          AS personal_nr,
       trunc(dbms_random.value(0,75))          AS prod_grp,
       'Dies ist ein besonderer Kommentar'     AS kommentar
  FROM dual
CONNECT BY level <= 1000000;
CREATE INDEX prod_test_prod_grp on produkt_test (prod_grp);
```

Austausch von Execution Plänen mit dbms_sqltune

Ausgangssituation:

```
SELECT * FROM produkt_test WHERE prod_grp = 13;
```



Ziel:

Austausch von Execution Plänen mit dbms_sqltune

Vorbereitung:

```
-- SQL-ID ermitteln. Wird gebraucht, um sie per SQL Profil auszutauschen
SELECT *
  FROM v$sql
WHERE sql_fulltext LIKE '%FROM produkt_test WHERE prod_grp = 13';
-- Ergebnis: SQL_ID=4x9an5guyavpk

-- getuntes SQL:
SELECT /*+ full(x) */ * FROM produkt_test x WHERE prod_grp = 13;

SELECT *
  FROM v$sql
WHERE sql_fulltext LIKE '%FROM produkt_test x WHERE prod_grp = 13%';
-- Ergebnis: SQL_ID=85m0ryw0dfjyw

SELECT * FROM TABLE(dbms_xplan.display_cursor('85m0ryw0dfjyw', 0, 'outline'));
```

Skriptausgabe | Autotrace | Explain-Plan | Abfrageergebnis

SQL | Alle Zeilen abgerufen: 32 in 0,156 Sekunden

```
PLAN_TABLE_OUTPUT
SQL_ID  85m0ryw0dfjyw, child number 0
-------------------------------------
SELECT /*+ full(x) */ * FROM produkt_test x WHERE prod_grp = 13

Plan hash value: 3973325831

-------------------------------------------------------------------------------------
| Id  | Operation         | Name         | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |              |       |       |  2533 (100)|          |
|*  1 |  TABLE ACCESS FULL| PRODUKT_TEST |   241 | 14701 |  2533   (1)| 00:00:01 |
-------------------------------------------------------------------------------------

Outline Data
-------------

  /*+
      BEGIN_OUTLINE_DATA
      IGNORE_OPTIM_EMBEDDED_HINTS
      OPTIMIZER_FEATURES_ENABLE('12.1.0.2')
      DB_VERSION('12.1.0.2')
      ALL_ROWS
      OUTLINE_LEAF(@"SEL$1")
      FULL(@"SEL$1" "X"@"SEL$1")
      END_OUTLINE_DATA
  */

Predicate Information (identified by operation id):
---------------------------------------------------

   1 - filter("PROD_GRP"=13)
```

Austausch von Execution Plänen mit dbms_sqltune

Durchführung:

```
1  -- Als DBA mit Rechten für die dbms_sqltune-Package nun dem DELETE das SQL-Profile unterschieben...
2  DECLARE
3      v_sql_id     VARCHAR2(30) := '4x9an5guyavpk'; -- SQL-ID des Originals
4      v_clsql_text CLOB;
5  BEGIN                                    das SQL-Profile unterschieben...
6      SELECT sql_fulltext
7        INTO v_clsql_text
8        FROM v$sqlarea
9       WHERE sql_id = v_sql_id;
10
11     dbms_sqltune.import_sql_profile(
12         sql_text    => v_clsql_text,
13         profile     => sqlprof_attr('FULL(@SEL$1 X@SEL$1)'),
14         name        => 'PROFILE_' || v_sql_id,
15         force_match => true
16     );
17 END;
18 /
19
20 -- Nachschauen, ob es korrekt angelegt wurde
21 SELECT * FROM dba_sql_profiles;
22
```

Skriptausgabe | Abfrageergebnis

SQL | Alle Zeilen abgerufen: 1 in 0,047 Sekunden

	NAME	CATEGORY	SIGNATURE	SQL_TEXT
1	PROFILE_4x9an5guyavpk	DEFAULT	2485911664915078956	SELECT * FROM produkt_test WHERE prod_grp = 13

Kontrolle:

```
41 /* Nun test, ob der Fake klappt... */
42 SELECT * FROM produkt_test WHERE prod_grp = 13;
43
```

Skriptausgabe | Autotrace | Abfrageergebnis | Explain-Plan

SQL | 0,063 Sekunden

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY
SELECT STATEMENT			241
TABLE ACCESS	PRODUKT_TEST	FULL	241

- Filter Predicates
  - PROD_GRP=13
- Other XML
  - {info}
    - info type="db_version"
      - 12.1.0.2
    - info type="parse_schema"
      - "BLMGR"
    - info type="plan_hash_full"
      - 1034621722
    - info type="plan_hash"
      - 3973325831
    - info type="plan_hash_2"
      - 1034621722
    - info type="sql_profile" note="y"
      - "PROFILE_4x9an5guyavpk"
    - {hint}
      - FULL(@"SEL$1" "PRODUKT_TEST"@"SEL$1")
      - OUTLINE_LEAF(@"SEL$1")
      - ALL_ROWS
      - DB_VERSION('12.1.0.2')

Tipps zum Umgang mit dbms_sqltune

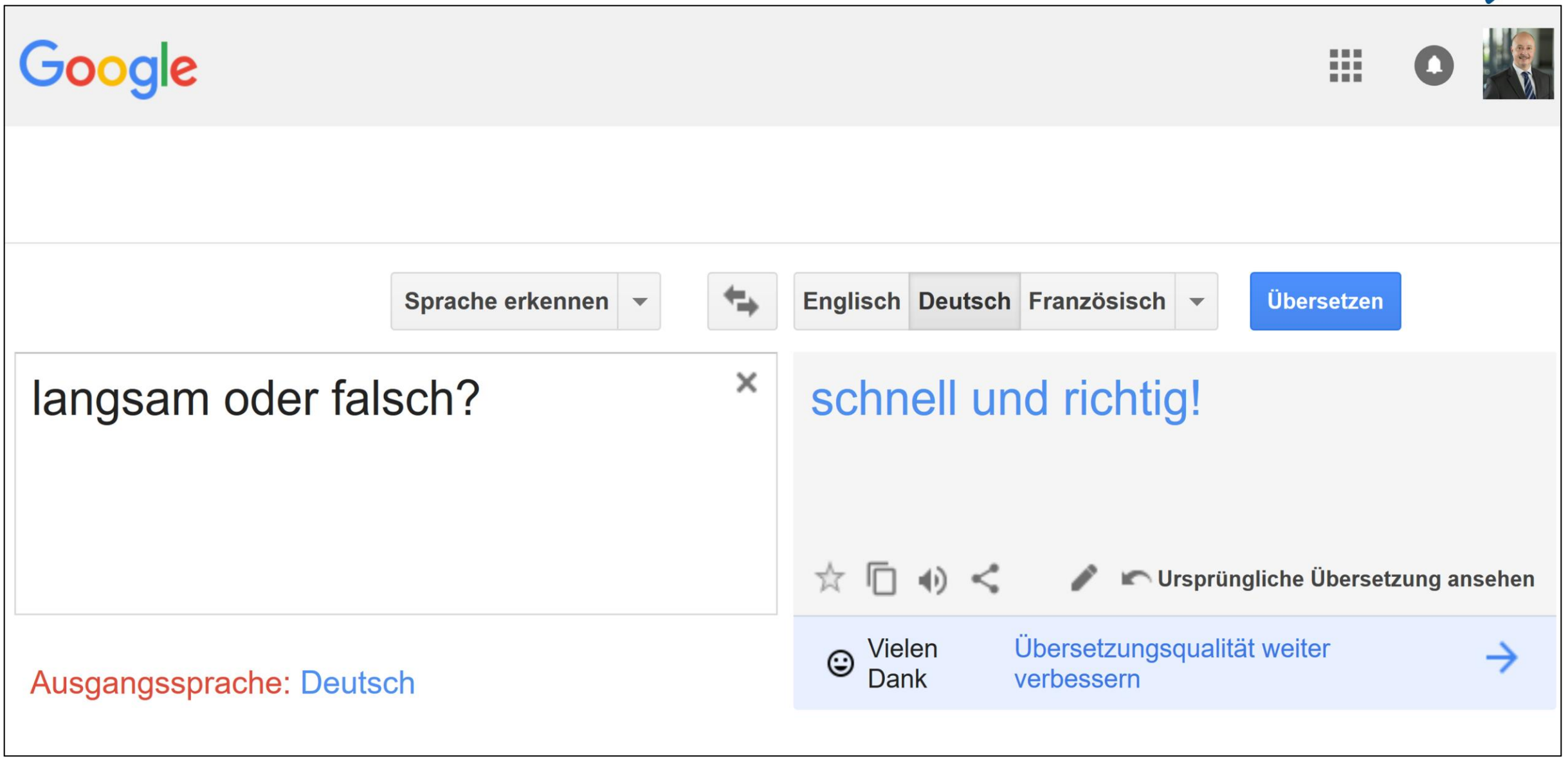
1.) Literale in Abfragen?

```
force_match  - If TRUE this causes SQL Profiles
               to target all SQL statements which have the same
               text after normalizing all literal values into
               bind variables. (Note that if a combination of
               literal values and bind values is used in a
               SQL statement, no bind transformation occurs.)
               This is analogous to the matching algorithm
               used by the "FORCE" option of the
               CURSOR_SHARING parameter.  If FALSE, literals are
               not transformed.  This is analogous to the
               matching algorithm used by the "EXACT" option of
               the CURSOR_SHARING parameter.
```

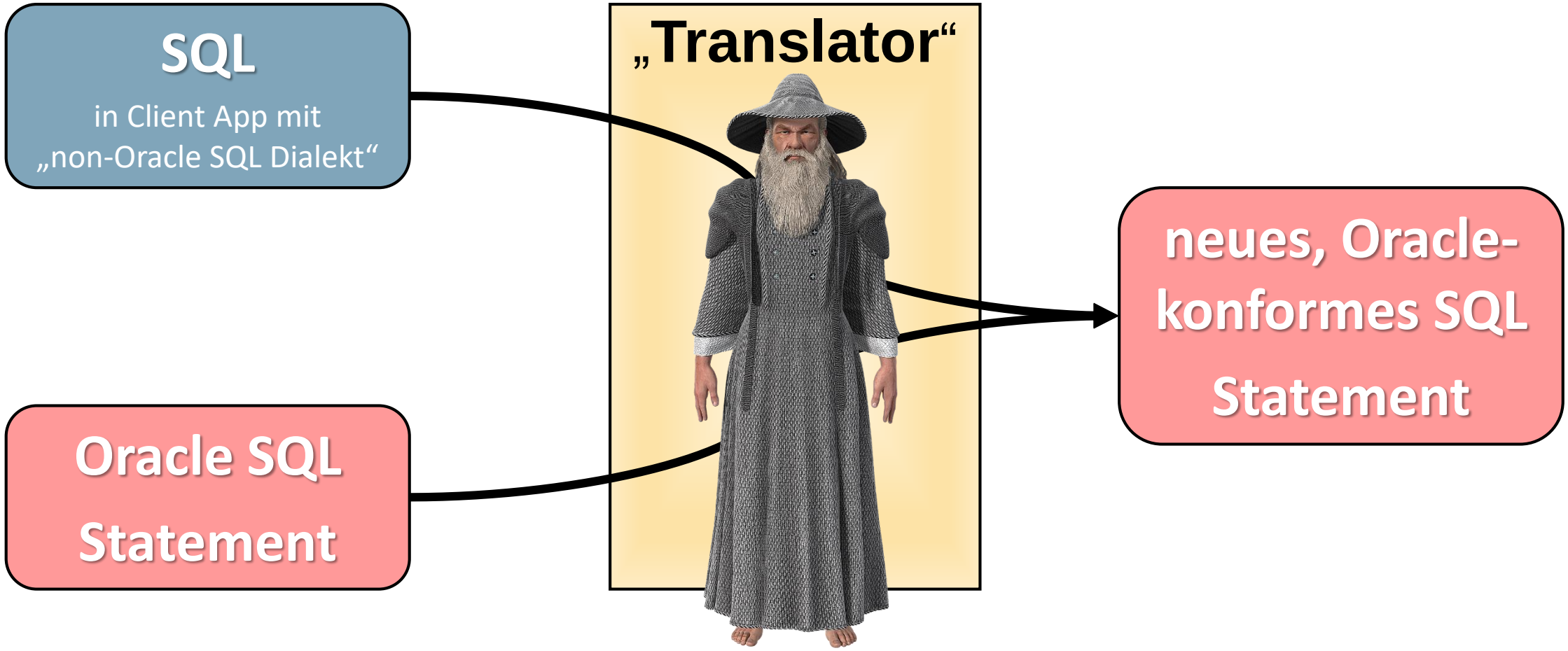
```
PROCEDURE import_sql_profile(
                    sql_text      IN CLOB,
                    profile       IN sqlprof_attr,
                    name          IN VARCHAR2 := NULL,
                    description   IN VARCHAR2 := NULL,
                    category      IN VARCHAR2 := NULL,
                    validate      IN BOOLEAN  := TRUE,
                    replace       IN BOOLEAN  := FALSE,
                    force_match   IN BOOLEAN  := FALSE);
```

2.) MOS / SQLT-Scripts: DocID 215187.1 => `coe_xfr_sql_profile.sql`-Utility

```
SQL> @coe_xfr_sql_profile.sql <SQL_ID> <PLAN HASH VALUE>
```


Google
Sprache erkennen
Englisch
Deutsch
Französisch
Übersetzen
langsam oder falsch?
schnell und richtig!
Ursprüngliche Übersetzung ansehen
Vielen Dank
Übersetzungsqualität weiter verbessern
Ausgangssprache: Deutsch

Das Oracle SQL Translation Framework – 2 Grundideen



SQL Translation Framework - Nutzen

schlechtes bzw. falsches SQL korrigieren

Migration von Client Anwendungen aus non DBs

Ausführung von Nicht-Oracle SQL- Apps auf einer Oracle-DB

Anwendungscode vor Produktivnahme testen

SQL Translation Framework - Komponenten

merlin2wo InfoDesign

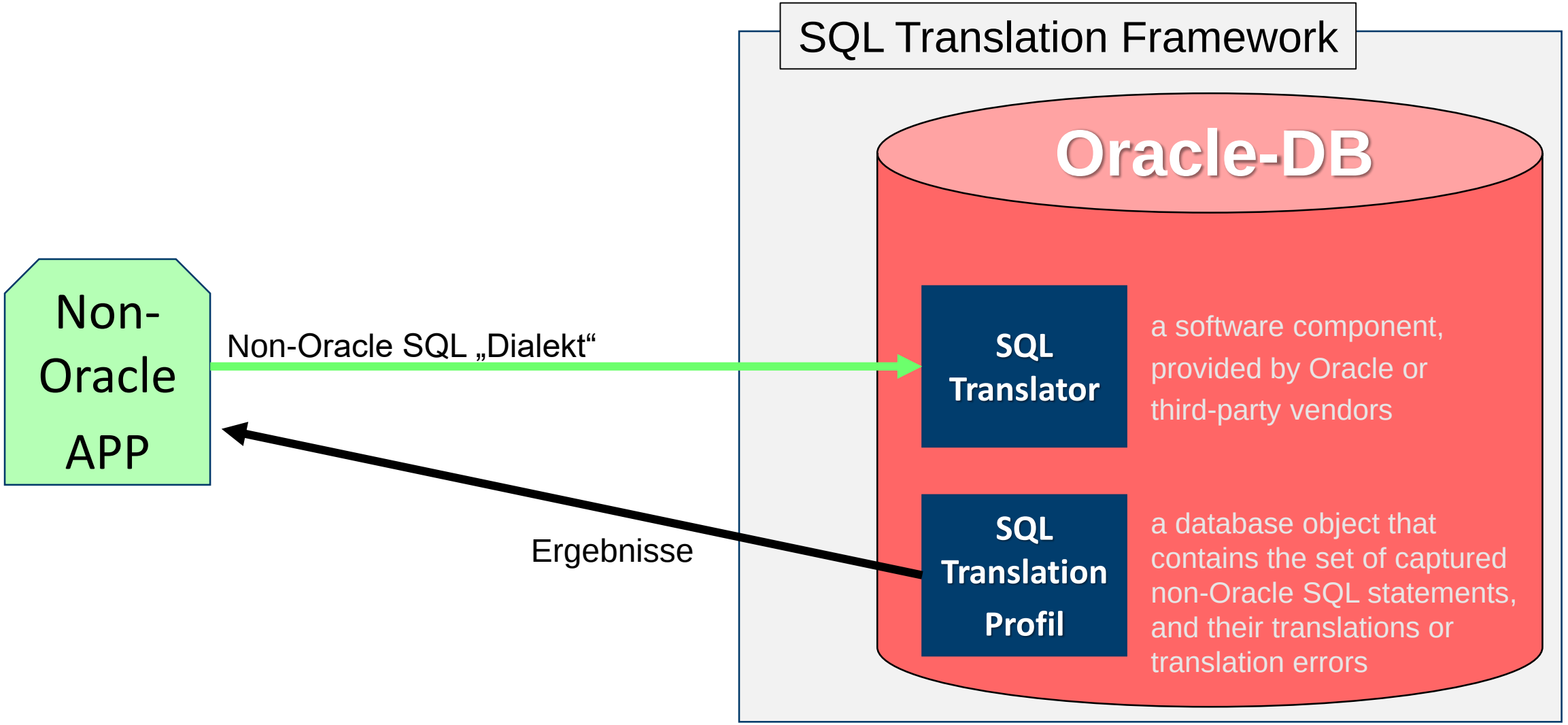
SQL Translator

Engine,
die das SQL
übersetzt

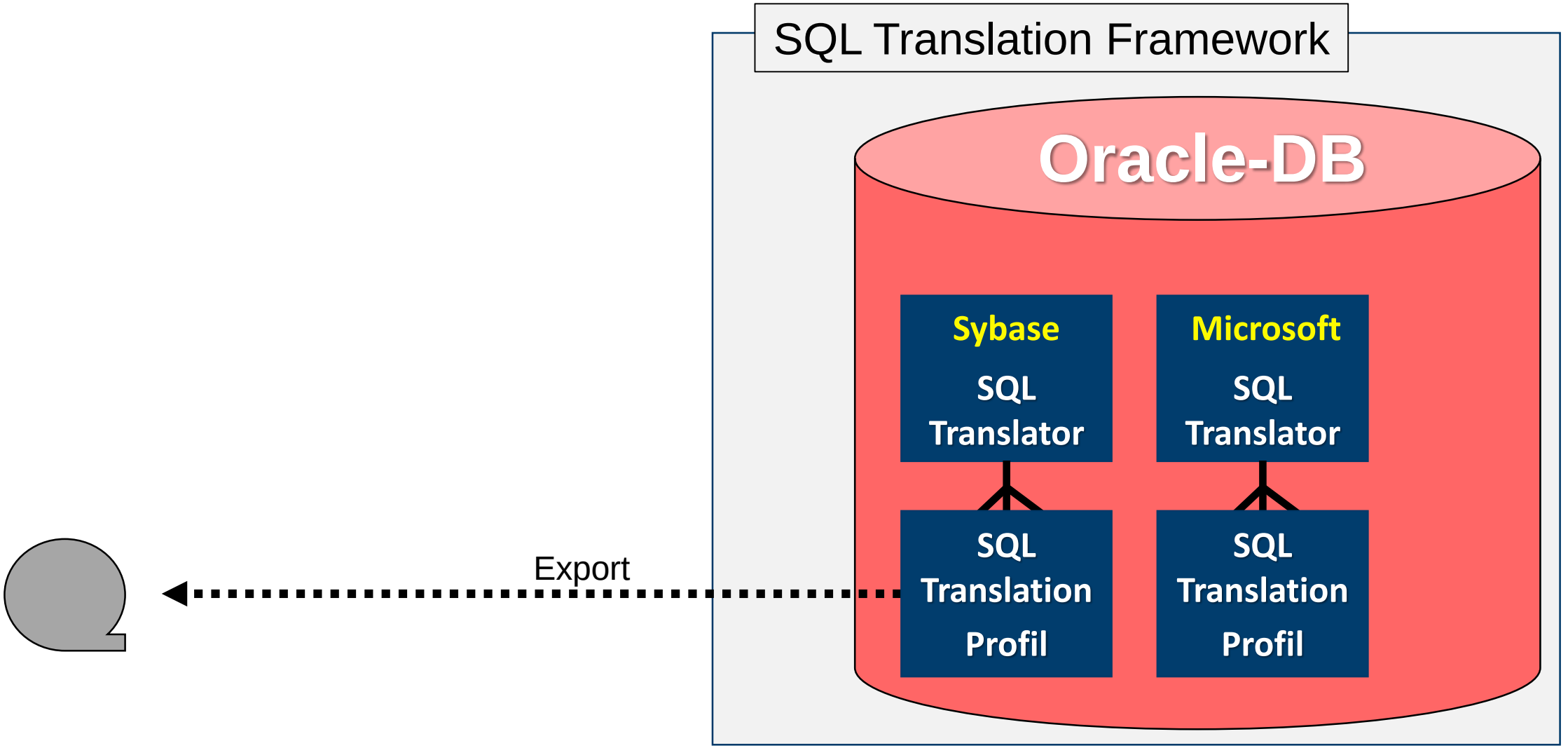
SQL Translation Profile

Sammlung
übersetzter
Statements

SQL Translation Framework: Grundkonzept SQL Translator I

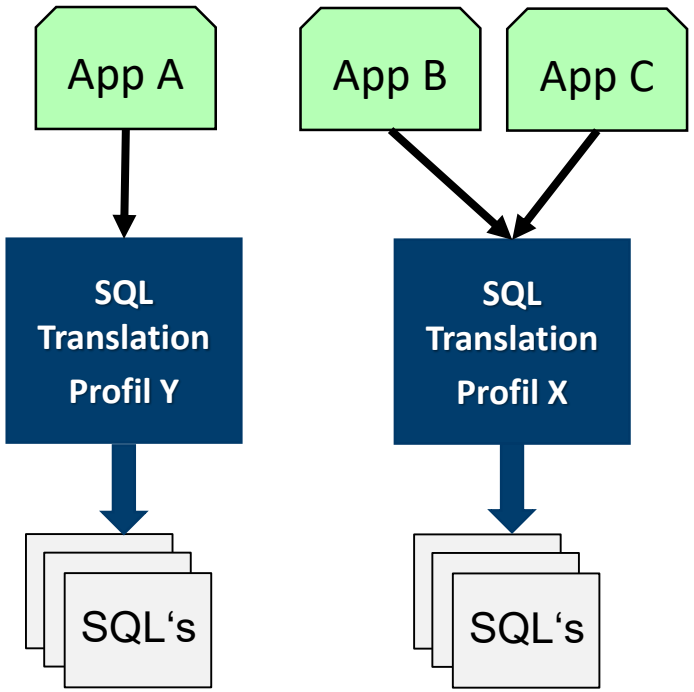


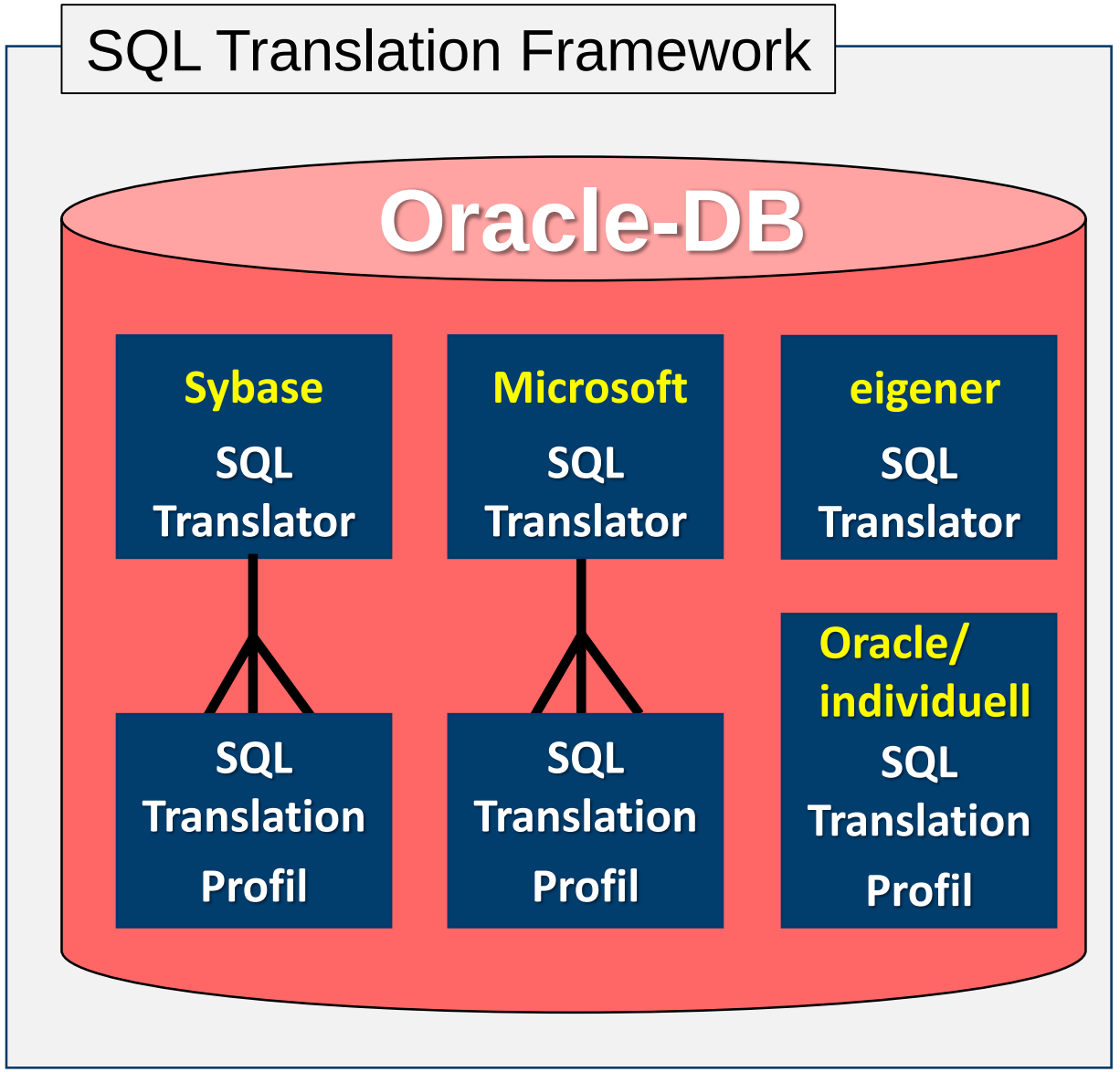
SQL Translation Framework: Grundkonzept SQL Translator II



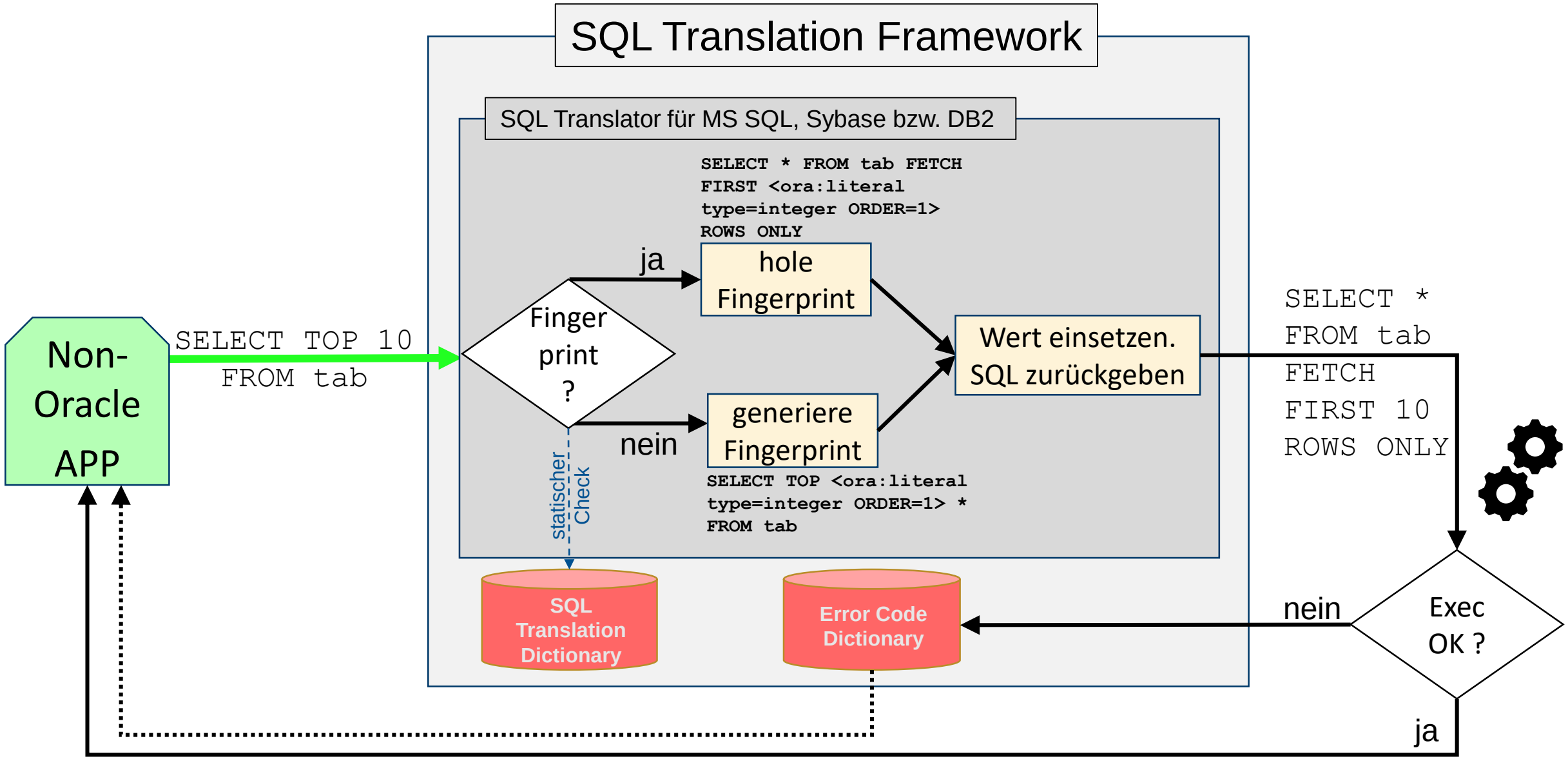
SQL Translation Framework: Grundkonzept SQL Translator III

Normalerweise gibt es ein
SQL Translation Profil pro Applikation.
Andernfalls können Anwendungen
übersetztes SQL gemeinsam nutzen:

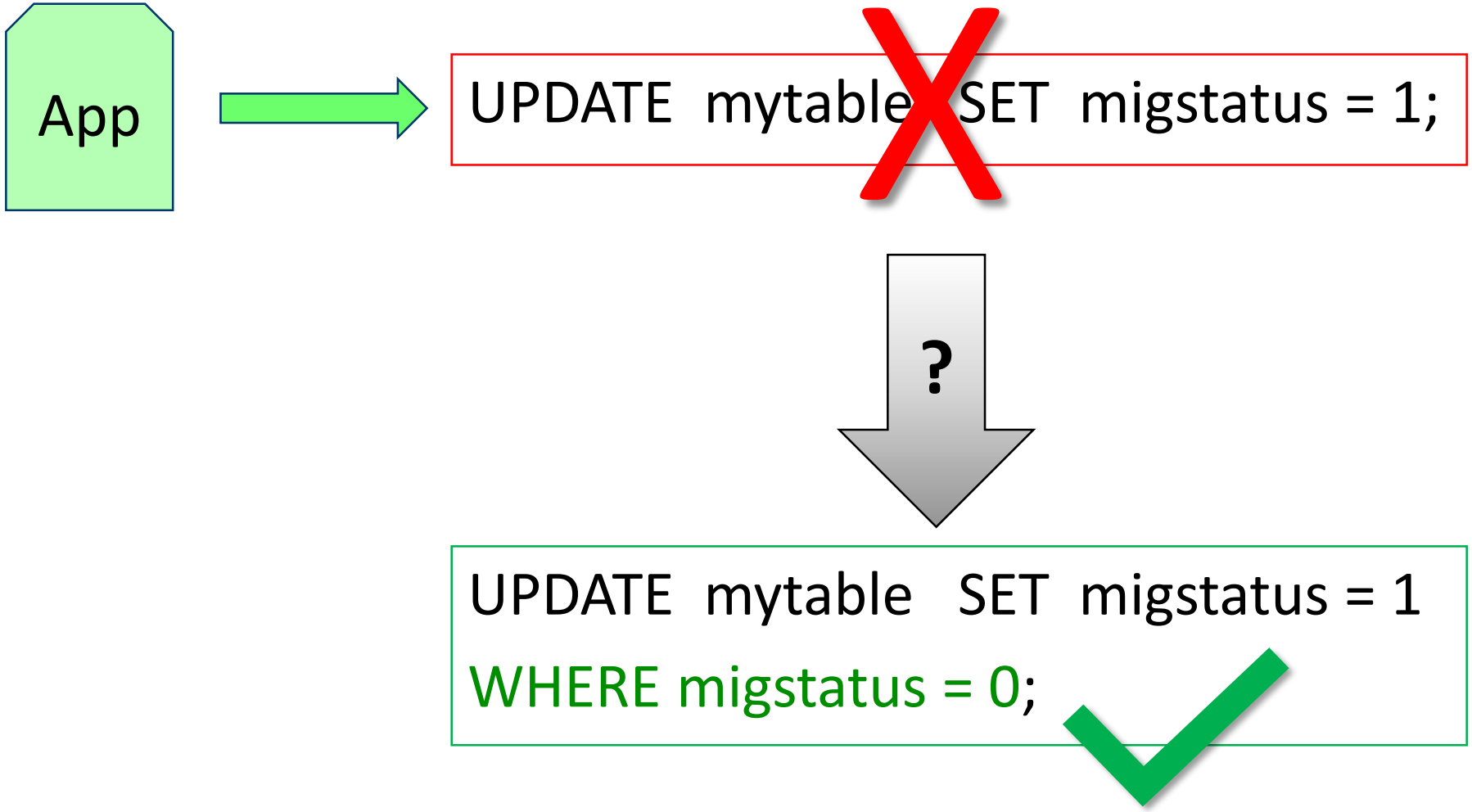




SQL Translation Framework: Grundkonzept SQL Translator IV



Die Aufgabe: direkte Manipulation von ORACLE SQL-Code…



Vorbereitungen I

1.) Rechte vergeben (als SYS/SYSTEM)

```
GRANT alter session,
      create sql translation profile TO AppOwner;
```

2.) Profil anlegen (als AppOwner)

```
dbms_sql_translator.create_profile(
                profile_name => 'MIGRATION_MYAPP');
```

merlin2nd InfoDesign

3.) Übersetzung eintragen

```
dbms_sql_translator.register_sql_translation(
    profile_name     => 'MIGRATION_MYAPP',
    sql_text         => 'UPDATE mytable SET migstatus = 1',
    translated_text => 'UPDATE mytable SET migstatus = 1 WHERE migstatus = 0',
    enable           => TRUE);
```

4.) Profil aktivieren

```
ALTER SESSION set sql_translation_profile = MIGRATION_MYAPP;
ALTER SESSION set events ='10601 trace name context forever, level 32';
```

5.) Profil aktivieren (alternativ)

```
CREATE OR REPLACE TRIGGER sql_translation_trg
  AFTER LOGON ON DATABASE
BEGIN
  IF USER IN ('APPOWNER') THEN
     execute immediate
             'ALTER SESSION set sql_translation_profile = MIGRATION_MYAPP';
     execute immediate
             'ALTER SESSION set events =''10601 trace name context forever,
                                         level 32''';
   END IF;
END;
/
```

Anwendungstest I: „Beeinflussung des SQL UPDATE“

Ausgangssituation:

Arbeitsblatt | Query Builder

```
1  SELECT * FROM mytable;
```

Abfrageergebnis

SQL | Alle Zeilen abgerufen: 2 in 0,016 Sekunden

	IRGENDEIN_INHALT	MIGSTATUS	LZT_AENDERUNG
1	unmigriert	0	(null)
2	migriert	1	(null)

Faking aktivieren…

```
ALTER SESSION set sql_translation_profile=MIGRATION_MYAPP;
ALTER SESSION set events ='10601 trace name context forever, level 32';
```

```
SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(null,null,'ALLSTATS LAST')) ;
```

Abfrageergebnis

SQL | Alle Zeilen abgerufen: 25 in 0,094 Sekunden

PLAN_TABLE_OUTPUT

```
SQL_ID  gabspxwdxy2dk, child number 1
-------------------------------------
UPDATE mytable SET migstatus = 1 WHERE migstatus = 0

Plan hash value: 3342251486

-----------------------------------------------
| Id  | Operation          | Name    | E-Rows |
-----------------------------------------------
|   0 | UPDATE STATEMENT   |         |        |
|   1 |  UPDATE            | MYTABLE |        |
|*  2 |   TABLE ACCESS FULL| MYTABLE |      1 |
-----------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   2 - filter("MIGSTATUS"=0)
```

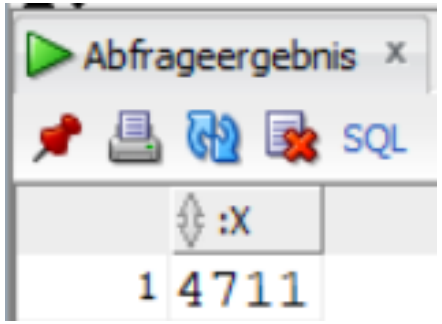
```
UPDATE mytable SET migstatus = 1;
```

2 Zeilen aktualisiert.

1 Zeile aktualisiert.

Handling von Bindevariablen und massive „Beeinflussung“ (andere Objektzugriffe)

```
DBMS_SQL_TRANSLATOR.REGISTER_SQL_TRANSLATION(
   profile_name     => 'MIGRATION_MYAPP',
   sql_text         => 'SELECT COUNT(*) FROM mytable WHERE migstatus = :X',
   translated_text => 'SELECT :X FROM dual');
```




```
1  variable X NUMBER
2  exec :X := 4711
3  SELECT COUNT(*) FROM mytable WHERE migstatus = :X;
4  SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(null,null,'ALLSTATS LAST')) ;
```

Abfrageergebnis 1 | SQL | Alle Zeilen abgerufen: 19 in 0,047 Sekunden

```
PLAN_TABLE_OUTPUT
1  SQL_ID  duq3ycb3d6bak, child number 0
2  -------------------------------------
3  SELECT :X FROM dual
4
5  Plan hash value: 1388734953
6
7  ----------------------------------------
8  | Id  | Operation         | Name | E-Rows |
9  ----------------------------------------
10 |   0 | SELECT STATEMENT  |      |        |
11 |   1 |  FAST DUAL        |      |      1 |
12 ----------------------------------------
```

merlin2nd InfoDesign

Statische Informationen

VIEW_NAME	COLUMNS…
USER_ERROR_TRANSLATIONS	PROFILE_NAME, ERROR_CODE, TRANSLATED_CODE, TRANSLATED_SQLSTATE, ENABLED, REGISTRATION_TIME, COMMENTS
USER_SQL_TRANSLATION_PROFILES	PROFILE_NAME, TRANSLATOR, FOREIGN_SQL_SYNTAX, TRANSLATE_NEW_SQL, RAISE_TRANSLATION_ERROR, LOG_TRANSLATION_ERROR, TRACE_TRANSLATION
USER_SQL_TRANSLATIONS	PROFILE_NAME, SQL_TEXT, TRANSLATED_TEXT, SQL_ID, HASH_VALUE, ENABLED, REGISTRATION_TIME, CLIENT_INFO, MODULE, ACTION, PARSING_USER_ID, PARSING_SCHEMA_ID, COMMENTS

sowie analog für ALL_%, DBA_% und CDB_%-Views

```
SELECT * FROM USER_SQL_TRANSLATIONS;
```

PROFILE_NAME	SQL_TEXT	TRANSLATED_TEXT	SQL_ID	HASH_VALUE	ENABLED	REGISTRATION_TIME
MIGRATION_MYAPP	UPDATE mytable SET migstatus = 1	UPDATE mytable SET migstatus = 1 WHERE migstatus = 0	43jq2vyh3uv37	2688380007	TRUE	25.09.18 14:26:28,993647000
MIGRATION_MYAPP	SELECT COUNT(*) FROM mytable WHERE migstatus = :X	SELECT COUNT(*) FROM dual WHERE dummy <> :X	by7m30angkhdf	2834907566	TRUE	25.09.18 14:30:15,630655000

merlin2wo InfoDesign

Dynamische Informationen

`UPDATE mytable SET migstatus = 1;` und dann: `SELECT * FROM` **`V$MAPPED_SQL;`**

Spalte	Inhalt
SQL_FULLTEXT	UPDATE mytable SET migstatus = 1
SQL_ID	43jq2vyh3uv37
HASH_VALUE	2688380007
MAPPED_SQL_FULLTEXT	UPDATE mytable SET migstatus = 1 WHERE migstatus = 0
MAPPED_SQL_ID	gabspxwdxy2dk
MAPPED_HASH_VALUE	467601842
SQL_TRANSLATION_PROFILE_ID	168756

```
SELECT
  sql_id,
  sql_translation_profile_id,
  name
FROM v$session JOIN
sys."_CURRENT_EDITION_OBJ"
ON (sql_translation_
    profile_id =obj#)
WHERE sid =
 (SELECT sys_context(
             'USERENV','SID')
  FROM dual);
```

```
SELECT sql_id, sql_text, executions FROM v$sqlarea
WHERE sql_id IN ('43jq2vyh3uv37','gabspxwdxy2dk');
```

SQL_ID	SQL_TEXT	EXECUTIONS
gabspxwdxy2dk	UPDATE mytable SET migstatus = 1 WHERE migstatus = 0	1

SQL_ID	P.-ID	NAME
…	168903	MIGRATION_MYAPP

merlin2nd InfoDesign

Vorbereitung, Ergebnis/Output und Deaktivierung

1.)

```
DBMS_SQL_TRANSLATOR.SET_ATTRIBUTE(
   profile_name    => 'MIGRATION_MYAPP',
   attribute_name  => DBMS_SQL_TRANSLATOR.ATTR_TRACE_TRANSLATION,
   attribute_value => 'TRUE');
```

```
1 variable X NUMBER
2 exec :X := 4711
3 SELECT COUNT(*) FROM mytable WHERE migstatus = :X;
```

2.)

```
Trace file /opt/oracle/diag/rdbms/prod/DEMO/trace/DEMO_ora_14572.trc
*** 2018-09-27 12:18:57.327
SQL Translation Profile "APPOWNER"."MIGRATION_MYAPP": original SQL text "DECLARE SqlDevBind1Z_1
   NUMBER:=:SqlDevBind1ZInit1;  BEGIN BEGIN  SqlDevBind1Z_1 := 4711; END; :X:=SqlDevBind1Z_1;  END;"
SQL Translation Profile "APPOWNER"."MIGRATION_MYAPP": translated SQL text "DECLARE SqlDevBind1Z_1
   NUMBER:=:SqlDevBind1ZInit1;  BEGIN BEGIN  SqlDevBind1Z_1 := 4711; END; :X:=SqlDevBind1Z_1;  END;"

*** 2018-09-27 12:19:43.309
SQL Translation Profile "APPOWNER"."MIGRATION_MYAPP":
    original SQL text "SELECT COUNT(*) FROM mytable WHERE migstatus = :X"
SQL Translation Profile "APPOWNER"."MIGRATION_MYAPP":
    translated SQL text "SELECT :X FROM dual"
```

3.)

```
DBMS_SQL_TRANSLATOR.SET_ATTRIBUTE(
   profile_name    => 'MIGRATION_MYAPP',
   attribute_name  => DBMS_SQL_TRANSLATOR.ATTR_TRACE_TRANSLATION,
   attribute_value => 'FALSE');
```

Was geht?

Änderung der Übersetzung: weniger Bindevariable …

```
DBMS_SQL_TRANSLATOR.REGISTER_SQL_TRANSLATION(
    profile_name     => 'MIGRATION_MYAPP',
    sql_text         => 'SELECT COUNT(*) FROM mytable WHERE migstatus = :X',
    translated_text => 'SELECT 4711 FROM dual');
```

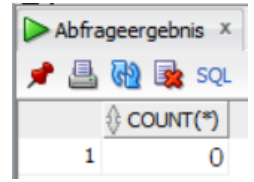
Error!

=> Meldung des SELECTs

```
ORA-01006:
bind variable
does not exist
```

Die anschließende Abfrage hat als Ergebnis 4711. Wahr oder falsch?

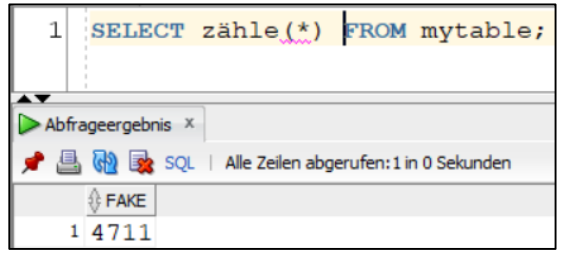
```
variable X NUMBER
exec :X := 4711
SELECT count(*) FROM mytable WHERE migstatus = :X;  -- => Ergebnis=4711???
```



Eintragen eines undefinierten Funktionsaufruf …

```
DBMS_SQL_TRANSLATOR.REGISTER_SQL_TRANSLATION(
    profile_name     => 'MIGRATION_MYAPP',
    sql_text         => 'SELECT zähle(*) FROM mytable',
    translated_text => 'SELECT 4711 AS Fake FROM dual');
```

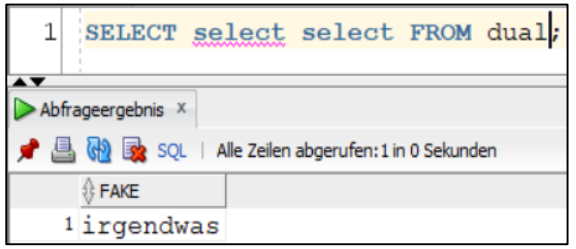
undef.?!



Eintragen eines invaliden SELECTs …

```
DBMS_SQL_TRANSLATOR.REGISTER_SQL_TRANSLATION(
    profile_name     => 'MIGRATION_MYAPP',
    sql_text         => 'SELECT select select FROM dual',
    translated_text => 'SELECT ''irgendwas'' AS Fake FROM dual');
```

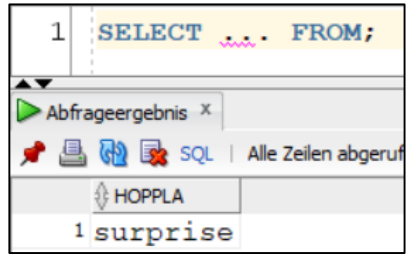
falsch?!



Eintragen eines invaliden SELECTs …

```
DBMS_SQL_TRANSLATOR.REGISTER_SQL_TRANSLATION(
    profile_name     => 'MIGRATION_MYAPP',
    sql_text         => 'SELECT ... FROM',
    translated_text => 'SELECT ''surprise'' as hoppla FROM dual');
```

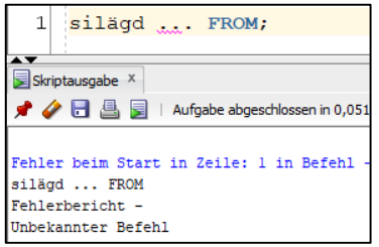
falsch?!



Eintragen eines invaliden SELECTs …

```
DBMS_SQL_TRANSLATOR.REGISTER_SQL_TRANSLATION(
    profile_name     => 'MIGRATION_MYAPP',
    sql_text         => 'silägd ... FROM',
    translated_text => 'SELECT ''auch das?'' as ooh FROM dual');
```

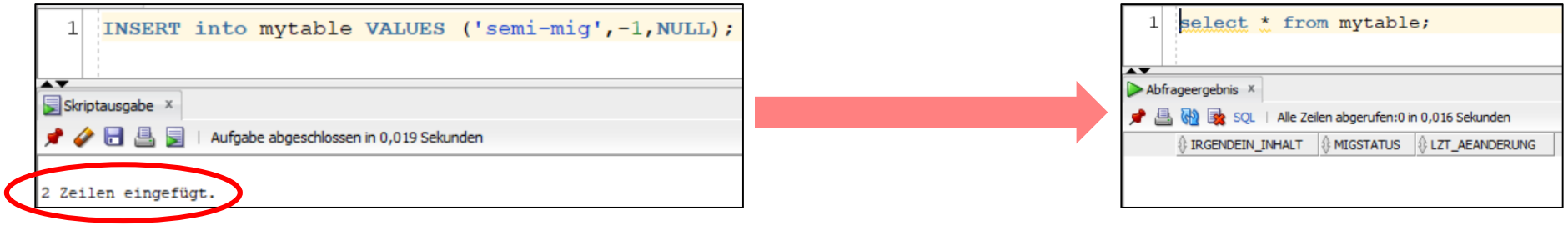
Error!



DML-Typ ändern …

I=>D

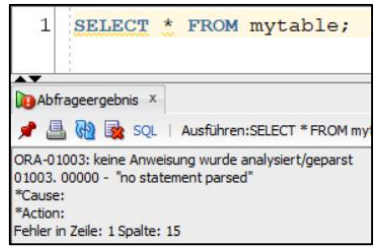
```
DBMS_SQL_TRANSLATOR.REGISTER_SQL_TRANSLATION(
    profile_name     => 'MIGRATION_MYAPP',
    sql_text         => 'INSERT into mytable VALUES (''semi-mig'',-1,NULL)',
    translated_text => 'DELETE FROM mytable');
```



Von SELECT zu DROP…

Error!

```
DBMS_SQL_TRANSLATOR.REGISTER_SQL_TRANSLATION(
    profile_name     => 'MIGRATION_MYAPP',
    sql_text         => 'SELECT * FROM mytable',
    translated_text => 'DROP TABLE mytable');
```



ALTER SESSION ändern …

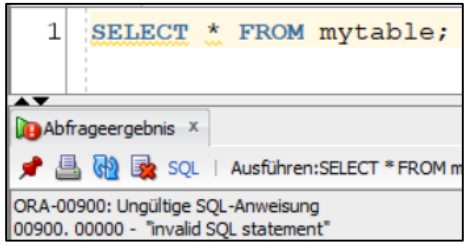
Session

```
DBMS_SQL_TRANSLATOR.REGISTER_SQL_TRANSLATION(
    profile_name     => 'MIGRATION_MYAPP',
    sql_text         => 'ALTER SESSION set optimizer_index_cost_adj=10000',
    translated_text  => 'ALTER SESSION set optimizer_index_cost_adj=1');
```

Von SELECT zu ALTER SESSION…

Error!

```
DBMS_SQL_TRANSLATOR.REGISTER_SQL_TRANSLATION(
    profile_name     => 'MIGRATION_MYAPP',
    sql_text         => 'SELECT * FROM mytable',
    translated_text  => 'ALTER SESSION set optimizer_index_cost_adj=1');
```



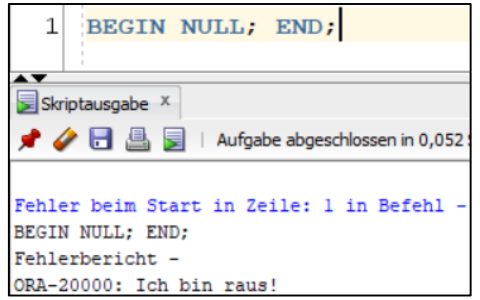
And the winner ist …

Ergänzende Informationen

`WITH, MERGE` werden auch erkannt/berücksichtigt

```
DBMS_SQL_TRANSLATOR.REGISTER_SQL_TRANSLATION(
     profile_name    => 'MIGRATION_MYAPP',
     sql_text        => 'CREATE TABLE X (x number)',
     translated_text => 'CREATE TABLE Y (y number)');
```

```
DBMS_SQL_TRANSLATOR.REGISTER_SQL_TRANSLATION(
     profile_name    => 'MIGRATION_MYAPP',
     sql_text        => 'BEGIN NULL; END;',
     translated_text => 'BEGIN raise_application_error
                  (-20000,''Ich bin raus!''); END;');
```



SQL in einem PL/SQL-Block? Nicht ersetzbar!!! Klar, weil…

Gefahren? Code Injection?

"Wow, **generalised bottom-up SQL injection**!“ Stew Ashton in Kerry Osborne's Oracle Blog

“…but quite obviously **it could be abused for attacking the database as well**. It could also be use for security "good" as well as security "bad". It could be used for instance to aid an application to use a shared "view/logon" account instead of having users use the schema to connect to, **it could be used to enable security within an existing application, it could maybe be used to enable encryption BUT the bigger issue is to make sure it is not used to bypass security**. Imagine an application that logs in and then says "select role from app_roles where username=lowest of the low" it could be changed to say "select role from app_roles where username=god“

in Pete Finnigan Blog

“Now, more than ever, **what you see on the network can be something completely different than what runs on the database**. So, you can see a statement like ‘select * from dual’ on the network but in the database it will be translated to ‘select * from credit_cards’…” in Slavik Markovich Blog

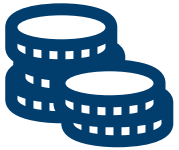
Fazit SQL Translation Framework

sehr elegant
bei falschen/
schlechten Code

schnell
& einfach
implementiert

Exp/Imp +
StandBy-
DBs

Transparent
für Betreiber
und Hersteller



keine
Option
oder Pack!

(k)ein
EE-Feature
???

Kooperation
mit DBAs
+ Security

Zeit zur
Konzeption
und Tests

Fragen?

merlin.zwo

Wir kümmern uns!

merlin.zwo InfoDesign GmbH & Co. KG

Stefan Winkler

Elsa-Brändström-Straße 14
76228 Karlsruhe
Tel. 0721 – 132 096 41

stefan.winkler@merlin-zwo.de
http://www.merlin-zwo.de