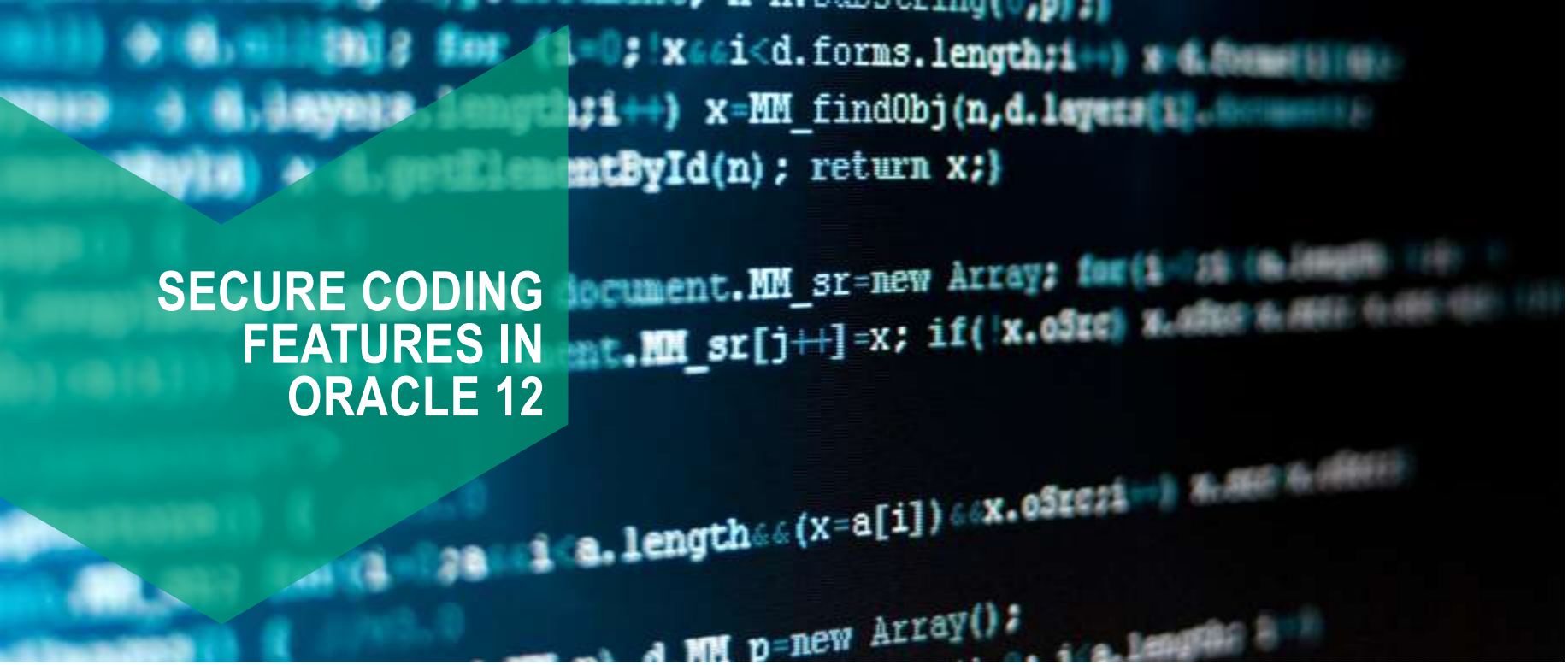


SECURE CODING FEATURES IN ORACLE 12

MARKUS LANGLOTZ & ROGER TROLLER

finnova

MARKUS LANGLOTZ

Ist seit Ende 2011 bei Finnova, Abteilung Technologie, im Bereich Data Management & Performance tätig

18 Jahre Oracle-Erfahrung mit Schwerpunkt auf Performance-Tuning

finnova

ROGER TROLLER

Ist seit Oktober 2016 in der Abteilung Consulting Dataintegration für Finnova tätig.

Davor 19 Jahre als Principal Consultant und Trainer im Bereich SQL & PL/SQL für die Trivadis AG im Einsatz.

Beta-Tester ORACLE 12.1 und ORACLE 12.2.

ORACLE ACE

finnova

MEETUP 1

Heute präsentieren wir vier ORACLE 12 Features, welche

- Fehlerquellen eliminieren
- Lesbarkeit unseres Codes erhöhen
- Komplexität unseres Codes reduzieren
- Unseren Code robuster machen

Oder ganz einfach:

- Uns das Leben vereinfachen können

AGENDA

- **Native Top-N**
- Validate Conversion
- On Conversion Error
- LISTAGG
- Gut zu wissen
- Fragen / Diskussion

finnova

ROW_LIMITING_CLAUSE

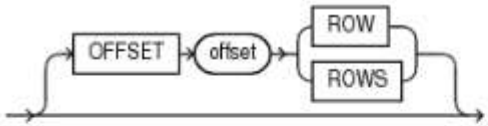
ORACLE 12 kennt als Erweiterung des Select Befehls die ROW_LIMITING_CLAUSE, welche folgende Funktionalitäten umfasst:

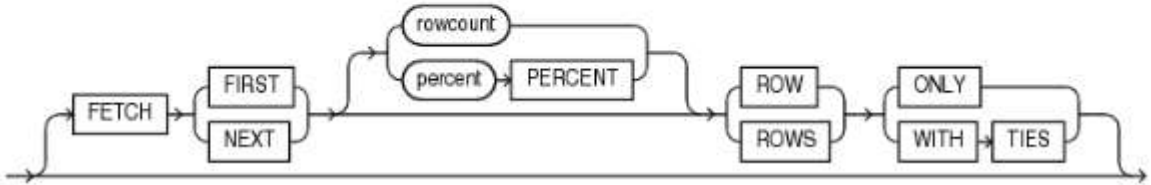
- Absolute Top-N Einschränkung
- Prozentuale Top-N Einschränkung
- Absolute Offset Deklaration (Skip Rows)
- Behandlung gleicher Werte (Ties)

Die ROW_LIMITING_CLAUSE sollte in Zukunft bisherige Konstrukte ersetzen.

ROW_LIMITING_CLAUSE (12.1)

ROW_LIMITING_CLAUSE





DARUM STATT HEUTE SO:

```
SELECT ...
  FROM (SELECT ...
             , ROWNUM as row_num
          FROM (SELECT ...
                  FROM IBWKT_BUCH_DET
                 ORDER BY ... )
       )
 WHERE row_num BETWEEN (NVL(:page,1) -1) * :page_size + 1
                   AND NVL(:page,1) * :page_size;
```

IN ZUKUNFT BITTE SO

```
SELECT ...
  FROM IBWKT_BUCH_DET
 ORDER BY ...
 OFFSET (NVL(:page,1) - 1) * :page_size ROWS
   FETCH NEXT :page_size ROWS ONLY;
```

DAS IST EIGENTLICH HEUTE SCHON FALSCH…

```
SELECT ROWID
      ,cs_msg_in_lnr
      ,c_ReadyforApplikation
      ,0
      ,COALESCE(dbms_lob.getlength("CS_MSG_TEXT"),0)
  FROM ...
 WHERE ...
   AND ROWNUM <= g_pp_cspackagermaxbulk -- maximale Anzahl
 ORDER BY 5 DESC; -- grösste Meldungen zuerst
```

DAS IST EIGENTLICH HEUTE SCHON FALSCH…

```
SELECT *
  FROM kd_stamm
 WHERE userbk_nr = 949
   AND mut_von > DATE '2017-08-01'
   AND ROWNUM <= 5
 ORDER BY mut_von desc;


-------------------------------------------------------------------------------
| Id  | Operation           | Name     | Rows  | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT    |          |       |   749 (100)|          |
|   1 |  SORT ORDER BY      |          |     1 |   749   (6)| 00:00:02 |
|*  2 |   COUNT STOPKEY     |          |       |            |          |
|*  3 |    TABLE ACCESS FULL| KD_STAMM |     1 |   748   (6)| 00:00:02 |
-------------------------------------------------------------------------------

 2 - filter(ROWNUM<=5)
 3 - filter(("MUT_VON">TO_DATE(' 2017-08-01 00:00:00', 'syyyy-mm-dd
              hh24:mi:ss') AND "USERBK_NR"=949 AND
DECODE("USERBK_NR ","DDQVPD00"."F_UBV2"(),1,
DECODE("DDQVPD00"."F_INBKLISTV2"("USERBK_NR"),1,1,0))=1))
```

3
2
1

IN ZUKUNFT BITTE SO:

```
SELECT ROWID
       ,cs_msg_in_lnr
       ,c_ReadyforApplikation
       ,0
       ,COALESCE(dbms_lob.getlength("CS_MSG_TEXT"),0)
   FROM ...
  WHERE ...
  ORDER BY 5 DESC
  FETCH FIRST g_pp_cspackagermaxbulk ROWS ONLY;
```

UND DER KLASSIKER

```
SELECT ...
  FROM (SELECT ...
              ,ROW_NUMBER() OVER (ORDER BY ...) rn
          FROM ...
         WHERE ...)
 WHERE rn = 1;
```

IN ZUKUNFT BITTE SO:

```
SELECT ...
  FROM ...
 WHERE ...
  ORDER BY ...
  FETCH FIRST 1 ROWS ONLY;
```

WAS KANN DAS FEATURE SONST NOCH?

- FETCH FIRST n ROWS WITH TIES;
  - Top-N mit Erkennung von Duplikaten
- FETCH FIRST n PERCENT ROWS ONLY
  - Prozentuales Top-N
- FETCH FIRST n PERCENT ROWS WITH TIES;
  - Prozentuales Top-N mit Erkennung von Duplikaten

VORTEILE GEGENÜBER ROWNUM / ROW_NUMBER

- bessere Lesbarkeit
- bessere Wartbarkeit
- einfacheres Konstrukt und dadurch
  - geringere Gefahr einer falschen Implementierung (WHERE VOR ORDER)
- weiterreichende Funktionalität

WAS LÖST DIE ROW_LIMITING_CLAUSE NICHT?

- Top-N pro Gruppe
- Empfehlung hier:
  - Weiterhin mit analytischen Funktionen ROW_NUMBER (RANK ODER DENSE_RANK) in einer Inline-View arbeiten.

```
SELECT ...
  FROM (SELECT ROW_NUMBER() OVER
                 (PARTITION BY ...
                      ORDER BY ...) AS rn
              ,...
         FROM ...
        WHERE ... )
 WHERE rn = 1;
```

AGENDA

Native Top-N

Validate Conversion

On Conversion Error

LISTAGG

Gut zu wissen

Fragen / Diskussion

finnova

DATENKONVERSION

«…darum prüfe, wer konvertieren will…»

Aber wie?

- Selbstgestrickte isNumber oder isDate Funktion?
- Konvertierungsblock mit Exception Handler?
- Built-In Funktionalität mit ORACLE 12.2

WAS IST IN UNSEREM CODE ZU FINDEN?

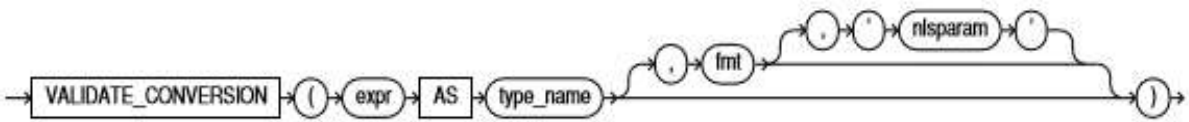
```
FUNCTION F_IsNumber(p_Value IN VARCHAR2) RETURN NUMBER IS
   n NUMBER;
BEGIN
   n := p_Value;
   RETURN n;
EXCEPTION
   WHEN OTHERS THEN RETURN 0;
END F_IsNumber;
```

WAS IST IN UNSEREM CODE ZU FINDEN?

```
FUNCTION isDate(P_Datumstring IN VARCHAR2) RETURN BOOLEAN IS
   L_TempDate DATE;
BEGIN
   L_TempDate := TO_DATE (P_Datumstring,'YYYY-MM-DD');
   RETURN TRUE;
EXCEPTION
   WHEN OTHERS THEN
      RETURN FALSE;
END isDate;
```

DATA CONVERSION ERWEITERUNGEN (12.2)

- Neue Funktion VALIDATE_CONVERSION
  - Prüfen ob Konversionen möglich sind (Funktionsresultat 0 = nicht möglich oder 1 = möglich)
  - Verwendbar in SQL und PL/SQL



VIELE «EIGENKOMPOSITIONEN» WERDEN DADURCH UNNÖTIG

Beispiel:

```
WITH DATA (str) AS (SELECT '17'    FROM DUAL UNION ALL
                    SELECT '1E5'   FROM DUAL UNION ALL
                    SELECT NULL    FROM DUAL UNION ALL
                    SELECT 'E1'    FROM DUAL UNION ALL
                    SELECT '17.25' FROM DUAL UNION ALL
                    SELECT '17,25' FROM DUAL)
 SELECT str
      , VALIDATE_CONVERSION(str AS NUMBER) AS RESULTAT
      , VALIDATE_CONVERSION(str AS NUMBER, '999999D99'
                    ?➡ ,q'{NLS_NUMERIC_CHARACTERS = ''',''}') AS RES_FORMATTED
   FROM DATA;
```

VIELE «EIGENKOMPOSITIONEN» WERDEN DADURCH UNNÖTIG

Resultat

```
STR      RESULTAT RES_FORMATTED
-----  ---------- -------------
17              1             1
1E5             1             0
                1             1
E1              0             0
17.25           1             0
17,25           0             1
```

AGENDA

Native Top-N

Validate Conversion

On Conversion Error

LISTAGG

Gut zu wissen

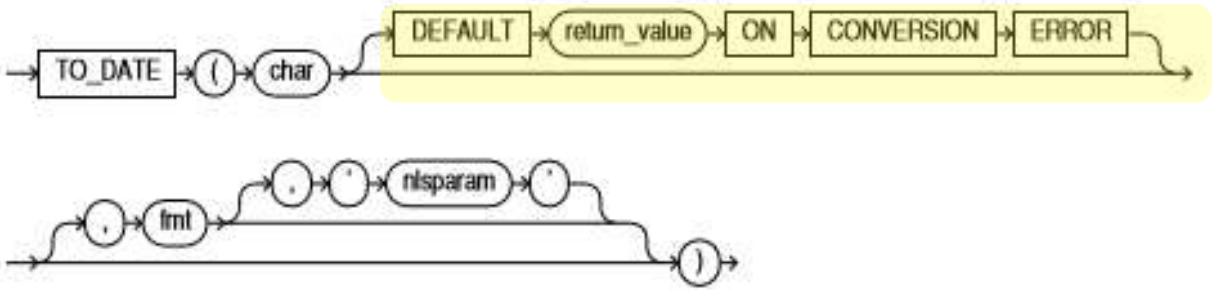
Fragen / Diskussion

finnova

DATA CONVERSION ERWEITERUNGEN (12.2)

Ebenfalls kann neu definiert werden, was im Falle einer nicht erfolgreichen Konvertierung zu tun ist.

- Erweiterung der Syntax bei Konvertierungsfunktionen (TO_DATE, TO_NUMBER, CAST, etc.)
  - Ersatzwert, falls Konversion fehl schlägt (NULL, DEFAULT)



DATA CONVERSION ERWEITERUNGEN

Beispiele (mehrstufige Datumskonvertierung):

```
WITH data (str) AS (SELECT '15-03-2016'  FROM dual UNION ALL
                    SELECT '2015.08.27'  FROM dual UNION ALL
                    SELECT '15-MAR-99'   FROM dual UNION ALL
                    SELECT '06.10.1582'  FROM dual UNION ALL
                    SELECT '23. März 99' FROM dual UNION ALL
                    SELECT 'Test'        FROM dual)
SELECT str
     , COALESCE(TO_DATE(str DEFAULT NULL ON CONVERSION ERROR,'DD-MON-RR')
               ,TO_DATE(str DEFAULT NULL ON CONVERSION ERROR,'DD. Month RR'
                        ,'NLS_DATE_LANGUAGE=GERMAN')
               ,TO_DATE(str DEFAULT NULL ON CONVERSION ERROR,'DD.MM.YYYY')
               ,TO_DATE(str DEFAULT NULL ON CONVERSION ERROR,'YYYY.MM.DD')
               ) AS conversion
 FROM data;
```

DATA CONVERSION ERWEITERUNGEN

Beispiele (mehrstufige Datumskonvertierung) Resultat:

```
STR          CONVERSION
------------ --------------------
15-03-2016   15.03.2016 00:00:00
2015.08.27   27.08.2015 00:00:00
15-MAR-99    15.03.1999 00:00:00
06.10.1582   15.10.1582 00:00:00
23. März 99 23.03.1999 00:00:00
Test
```

AGENDA

Native Top-N

Validate Conversion

On Conversion Error

LISTAGG

Gut zu wissen

Fragen / Diskussion

finnova

LISTAGG ERWEITERUNGEN

Seit der Version 11.2 kennen wir die LISTAGG Funktion – und eventuell auch ORA-01489.

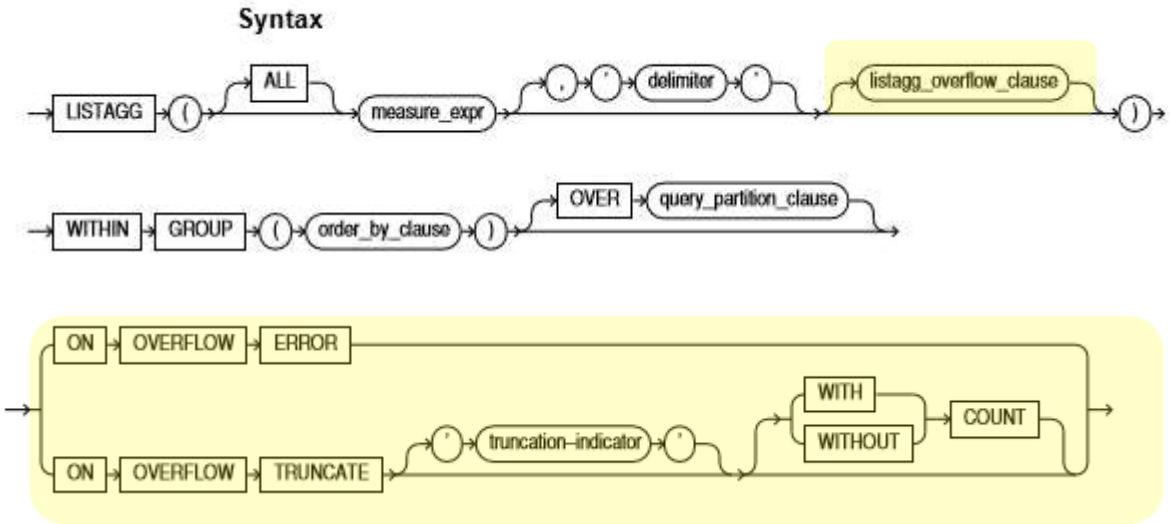
```
SELECT SUBSTR(object_name,1,2)
     , LISTAGG(object_name || '(' || object_type || ')',',')
       WITHIN GROUP (ORDER BY object_name) AS object_list
  FROM user_objects
 WHERE object_name LIKE '__\_%' ESCAPE '\'
 GROUP BY SUBSTR(object_name,1,2);

ORA-01489: result of string concatenation is too long
01489. 00000 -  "result of string concatenation is too long"
```

Limite liegt bei 4000 Bytes – und dies wird sich wohl auch nicht ändern…

LISTAGG ERWEITERUNGEN

ORACLE 12.2 erweitert die LISTAGG Funktion um eine Überlaufbehandlung.



LISTAGG ERWEITERUNGEN

- `ON OVERFLOW ERROR (Default)`
- `ON OVERFLOW TRUNCATE`
  - schneidet ganze Elemente weg und fügt die Anzahl abgeschnittener Elemente als Zähler hinzu
- `ON OVERFLOW TRUNCATE WITHOUT COUNT`
  - schneidet ganze Elemente weg ohne Zähler
- `ON OVERFLOW TRUNCATE 'truncation indicator' WITH COUNT`
  - ermöglicht einen eigenen Indikator für den Überlauf

LISTAGG ERWEITERUNGEN - BEISPIEL

Beispiel:

```
SELECT SUBSTR(object_name,1,2)
     , LISTAGG(object_name || '(' || object_type || ')',',' 
               ON OVERFLOW TRUNCATE '...' WITH COUNT)
       WITHIN GROUP (ORDER BY object_name) AS object_list
  FROM user_objects
 WHERE object_name LIKE '__\_%' ESCAPE '\'
 GROUP BY SUBSTR(object_name,1,2);
```

LISTAGG ERWEITERUNGEN

- Schliesst eine Lücke beim Erstellen von CSV-Strings
- Eliminiert eine Fehlerquelle, welche an vielen Orten (implizit) vorhanden ist.
- Löst die Problematik überlanger Strings nicht vollständig, da Abschneiden nicht unbedingt die Lösung ist.
  - Eine eigene LISTAGG_CLOB Funktion ist eventuell weiterhin erforderlich
  - Oder aber eine Aufteilung auf mehrere Zeilen (<= 4000 Byte)

UND WENN DIE LISTAGG-ERWEITERUNGEN MEIN PROBLEM NICHT LÖSEN KÖNNEN?

- Wenn ich beispielsweise „alles“ sehen will?
  - Dann könnte ein nächstes Meetup Thema „Row Pattern Matching“ interessant werden…

```
WITH DATA AS (SELECT substr(object_name,1,2) AS bereich
                   , object_name || '(' || object_type || ')' AS object_name
                FROM user_objects
               WHERE object_name LIKE '__\_%' ESCAPE '\')
SELECT BEREICH, grp_no
     , LISTAGG(object_name,',')
         WITHIN GROUP (ORDER BY object_name) as object_list
  FROM data
 MATCH_RECOGNIZE (PARTITION BY bereich
                      ORDER BY object_name
                  MEASURES match_number() AS grp_no
                  ALL ROWS PER MATCH
                  AFTER MATCH SKIP PAST LAST ROW
                  PATTERN (S b+)
                  DEFINE b AS LENGTHB(S.object_name)
                         + SUM(LENGTHB(CONCAT(b.object_name,',')))
                         + LENGTHB(',') <= 4000)
 GROUP BY bereich, grp_no;
```

AGENDA

Native Top-N

Validate Conversion

On Conversion Error

LISTAGG

Gut zu wissen

Fragen / Diskussion

finnova

UND WAS ES SCHON LÄNGER GIBT… COMPILER-WARNUNGEN

Der Oracle-PL/SQL-Compiler weist uns auf einige Dinge hin, wenn wir ihn dies tun lassen und die Warnungen beachten…

Beispiele, speziell interessant für SECURE CODING:

- PLW-06025 (ab 12.2)
  - Implizite Konvertierungen zwischen NUMBER/DATE/VARCHAR Typen ohne explizite Berücksichtigung von NLS-Einstellungen
- PLW-07207 (ab 12.2)
  - BULK COLLECT ohne Angabe einer LIMIT-Klausel. Häufiges Problem, kann zu Speicherengpässen führen und damit zu Laufzeitfehlern
- PLW-7204 / (PLW-07202)
  - Diese Fehler zeigen an, dass Konvertierungen von SQL zu PL/SQL auftreten. Dies kann auf potentielle ORA-01722-Laufzeitfehler hinweisen oder auf Folgeprobleme im Bereich Performance.

UND WAS ES SCHON LÄNGER GIBT… DYNAMISCHES SQL

- Dynamisches SQL da muss und darf man nicht immer konkatenieren
- Execute Immediate (Native Dynamic SQL) funktioniert auch mit Bind Variablen
- DBMS_SQL funktioniert ebenfalls mit Bind-Variablen

TOOLUNTERSTÜTZUNG

- Wer hilft mir dabei „defensiv“ zu entwickeln und Schwachstellen aufzudecken?
  - Oracle PL/SQL Compiler – Compiler-Warnungen
  - SQL Developer – PLCop Plugin
  - PL/SQL Developer – Developer Hints
  - SonarQube PL/SQL Plugin
  - TOAD CodeXpert
  - ClearSQL / SQL Detective

LINKS UND WEITERFÜHRENDE DOKUMENTATION

- Interne 12er Seite (Feature Freigabe)
  https://dev.finnova.ch/confluence/display/ora12/Oracle+Features
- ROW LIMITING CLAUSE (Top-N Query)
  http://docs.oracle.com/database/122/SQLRF/SELECT.htm#SQLRF01702
- VALIDATE_CONVERSION
  http://docs.oracle.com/database/122/SQLRF/VALIDATE_CONVERSION.htm#SQLRF-GUID-DC485EEB-CB6D-42EF-97AA-4487884CB2CD
- ON CONVERSION ERROR
  http://docs.oracle.com/database/122/SQLRF/CAST.htm#SQLRF00613
- LISTAGG
  http://docs.oracle.com/database/122/DWHSG/sql-analysis-reporting-data-warehouses.htm#DWHSG8702

AGENDA

Native Top-N

Validate Conversion

On Conversion Error

LISTAGG

Rund um das Thema

Fragen / Diskussion

finnova

DANKE UND BIS ZUM NÄCHSTEN MAL!

finnova