ORACLE

Analysemöglichkeiten mit SQL:
Mehr als SUM und GROUP BY

Carsten Czarski
ORACLE Deutschland B.V. & Co KG

Analysemöglichkeiten mit SQL:
Mehr als SUM und GROUP BY

- Aggregatsfunktionen: Klassisch
- Analytische Funktionen
- Eigene Aggregatsfunktionen
- SQL-Funktionen für Spezialaufgaben
  - Reguläre Ausdrücke
  - Pivoting
  - SQL Model Clause (Spreadsheet Formeln)
- Bonus Track: Ausführungspläne

Klassische Aggregatsfunktionen
GROUP BY-Klausel

- Wird überall und ständig verwendet
- Aber: noch alle Möglichkeiten bekannt ...?
  HAVING, GROUP BY ROLLUP | CUBE | GROUPING SETS

```
select deptno, sum(sal) from emp
group by deptno
having sum(sal) > 1000
```

Übrigens:
HAVING ist gut zum Ausfiltern von Dubletten ...

GROUP BY Erweiterungen

- Berechnen zusätzlicher Aggregate entlang der Dimensionen von GROUP BY

```
select job, deptno, sum(sal), avg(sal), min(sal), max(sal)
     from emp group by rollup (job, deptno)

JOB       DEPTNO SUM(SAL) AVG(SAL) MIN(SAL) MAX(SAL)
--------- ------ -------- -------- -------- --------
CLERK         10     1300     1300     1300     1300
CLERK         20     1900      950      800     1100
CLERK         30      950      950      950      950
CLERK                4150     1038      800     1300
:
ANALYST       20     6000     3000     3000     3000
ANALYST              6000     3000     3000     3000
:               :       :        :        :        :
                    29025     2073      800     5000
```

VARCHAR2 zusammenfassen
LISTAGG

- Aggregatsfunktion für Zeichenketten
- Ab Oracle 11.2 verfügbar
- Auch als analytische Variante

```
select
 deptno,
listagg(ename, ':') within group (order by ename) ename_list
from emp
group by deptno

    DEPTNO ENAME_LIST
---------- ----------------------------------------
        10 CLARK:KING:MILLER
        20 ADAMS:FORD:JONES:SCOTT:SMITH
        30 ALLEN:BLAKE:JAMES:MARTIN:TURNER:WARD
```

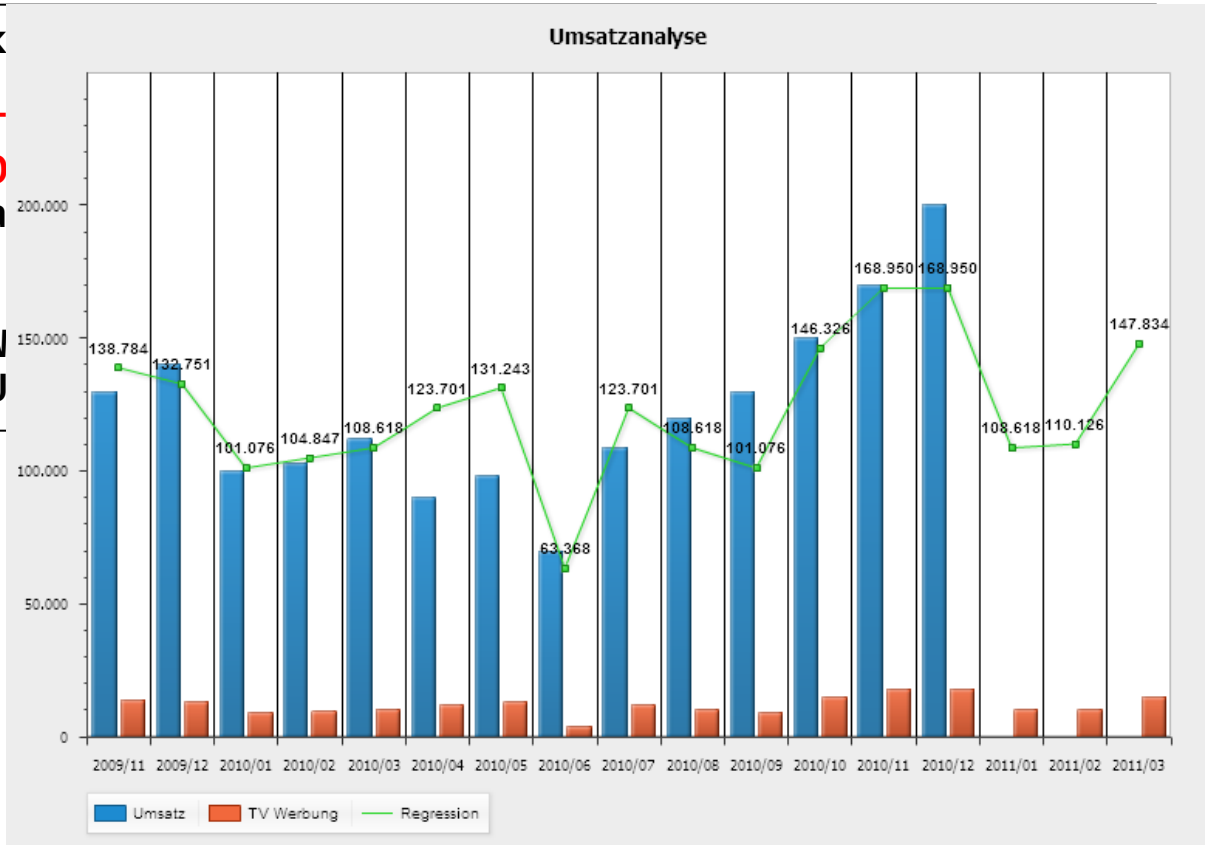
Statistische Funktionen der Datenbank

- Ranking-Funktionen
  - **rank, dense_rank, cume_dist, percent_rank, ntile**
- Analytische Funktionen (Query Window)
  - **Avg, sum, min, max, count, variance, stddev, first_value, last_value**
- Analytische Funktionen (Aggregate)
  - **Sum, avg, min, max, variance, stddev, count, ratio_to_report**
- Interrow-Berechnungen
  - **Lag, lead**
- Statistische Aggregate
  - **Korrelation, Lineare Regression, Kovarianz**

Beispiel: Einfache lineare Regression

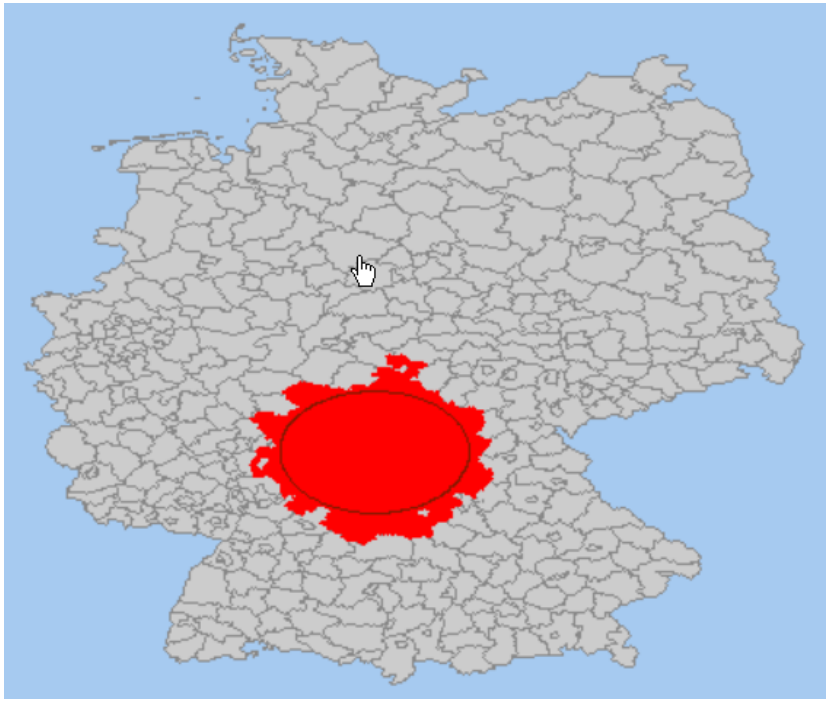
- Prognose "UMSATZ" anhand "TV-Werbung"

```
with regr_k
 select
   REGR_INT
   REGR_SLO
 from verka
)
select
  a + b * W
from VERKAU
```



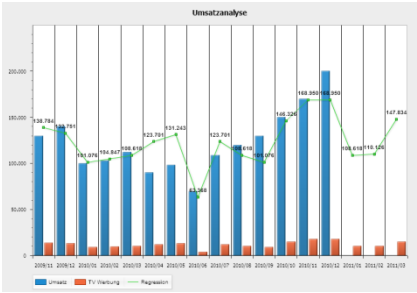
Aggregatsfunktionen und Geodaten
Räumliche Daten aggregrieren ...

- Gebiete (Polygone) zusammenfassen
  - Ermitteln von Vertriebsgebieten
  - SDO_AGGR_UNION
- Gemeinsames Zentrum
  - SDO_AGGR_CENTROID



Analytische Funktionen: Das Konzept

- Bessere Abfrage-Performance
  - Berichte
  - OLAP
  - BI-Applikationen
- Enge Integration in den Datenbankkern
  - Verarbeitung ist effizient und skalierbar (Optimizer)
  - Code ist einfacher und flexibler
  - Reduzierte Ausführungen auf der Clien



Beispiel: Gleitender Durchschnitt

```
SELECT manager_id id, last_name, hire_date, salary, AVG(salary)
  OVER (
    PARTITION BY manager_id ORDER BY hire_date
    ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING
  ) s_avg
FROM employees;
```

ID	LAST_NAME	HIRE_DATE	SALARY	S_AVG
100	Kochhar	21-SEP-89	17000	17000
100	De Haan	13-JAN-93	17000	15000
100	Raphaely	07-DEC-94	11000	11966.6667
100	Kaufling	01-MAY-95	7900	
100	Hartstein	17-FEB-96	13000	
100	Weiss	18-JUL-96	8000	
100	Russell	01-OCT-96	14000	
100	Partners	05-JAN-97	13500	
100	Errazuriz	10-MAR-97	12000	
100	Fripp	10-APR-97	8200	
200				

Analytische Funktionen
NTH-Value

- Liefert den "n-ten" Wert einer sortierten Reihe
- Beispiel anhand der Tabelle EMP

  Gib mir zu jedem "Employee" das zweitbeste Gehalt der Abteilung

```
select
  ename, deptno, sal,
  nth_value(sal, 2) over (
    partition by deptno
    order by sal desc
    range between unbounded preceding and unbounded following
  ) "2ND_BEST"
from emp
```

Analytische Funktionen
NTH-Value

- Beispiel anhand der Tabelle EMP
  Gib mir zu jedem "Employee" das zweitbeste Gehalt der Abteilung

ENAME	DEPTNO	SAL	2ND_BEST
KING	10	5000	2450
CLARK	10	2450	2450
MILLER	10	1300	2450
SCOTT	20	3000	3000
FORD	20	3000	3000
JONES	20	2975	3000
ADAMS	20	1100	3000
SMITH	20	800	3000
BLAKE	30	2850	1600
ALLEN	30	1600	1600
TURNER	30	1500	1600

Eigene Aggregatsfunktionen

- Objekttyp erzeugen – CREATE TYPE
- Interface im Type Body ausprogrammieren
  - ODCIAggregateStart
  - ODCIAggregateMerge
- Aggregatsfunktion erzeugen
- Nutzbar ...
  - Mit GROUP BY
  - Als analytische Funktion


SQL-Aggregatsfunktion "PRODUCT" fehlt ...? Kein Problem - wir bauen sie selbst!

Auf einer Veranstaltung in München, während es um das Thema *Analytische Funktionen* ging, fragte mich jemand nach einer Aggregatsfunktion für die Bildung eines Produktes - also nicht das Aufsummieren der Werte wie bei SUM, sondern das multiplizieren - und in der Tat: eine solche Funktion gibt es in Oracle nicht.

Nun kann man das mit PL/SQL natürlich auf einfachste Weise nachprogrammieren - aber eine solche Funktion lässt sich nicht in SQL-Abfragen mit GROUP BY nutzen. Berechnungen für Datengruppen in einer Tabelle müssen dann mühsam mit PL/SQL nachgebildet werden.

Aber das muss nicht sein. Es ist kaum bekannt, aber man kann eigene Aggregatsfunktionen in der Datenbank hinterlegen. Das habe ich in einem sehr frühen Blog-Posting schonmal genutzt. Mit der dort beschriebenen **LIST**-Funktion kann man VARCHAR2-Spalten auch von vor Oracle11g zusammenfassen (in Oracle11g gibt es mit **LISTAGG** ja eine eingebaute Funktion). Um ein User Defined Aggregate zu bauen, muss man eine Schnittstelle ausprogrammieren. Und hier ist der Code für **AGG_PRODUCT**.

```
drop function agg_product;
drop type agg_product_t;

create type agg_product_t as object(
  v_agg_product number,

  static function ODCIAggregateInitialize(
    sctx IN OUT agg_product_t
  ) return number,
  member function ODCIAggregateIterate(
    self IN OUT agg_product_t, value IN number
  ) return number,
  member function ODCIAggregateTerminate(
```

http://sql-plsql-de.blogspot.co.uk/2011/02/sql-aggregatsfunktion-product-fehlt.html

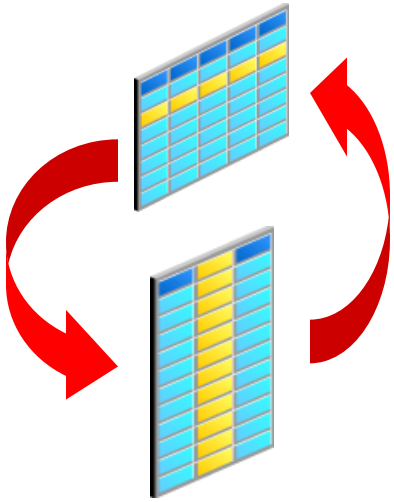
Eigene Aggregatsfunktionen
Konkretes Beispiel

```
create type agg_product_t as object(
  v_agg_product number,
  static function ODCIAggregateInitialize
  member function ODCIAggregateIterate
  member function ODCIAggregateTerminate
  member function ODCIAggregateMerge
);

create or replace type body agg_product_t is
:
end agg_product;

CREATE or replace FUNCTION agg_product (
  input number
) RETURN number
PARALLEL_ENABLE AGGREGATE USING agg_product_t;
```

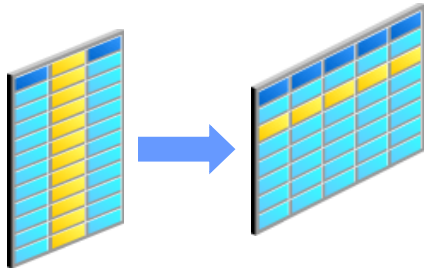
Kreuztabellen ganz einfach!
... mit neuen SQL Operationen



- Mit SQL-Klauseln **PIVOT** und **UNPIVOT**
  - PIVOT wandelt Zeilen in Spalten um
  - UNPIVOT wandelt Spalten in Zeilen um
- Vorteile:
  - Komplexe, eigene Logik entfällt
  - Optimiert für große Datenmengen
  - Profitiert von Abfrageoptimierung durch Statistiken, Partitionierung, Indizes usw.
- Implementierung:

```
[FROM table] PIVOT [pivot_for][pivot_in]
[FROM table] UNPIVOT [unpivot_for][unpivot_in]
```

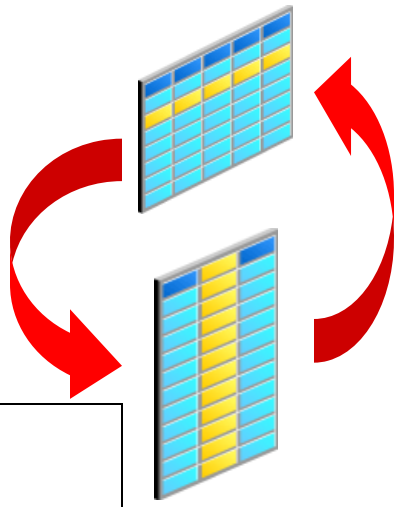
SQL Pivot und Unpivot
Am Beispiel



	COUNTRY	PROD	YEAR	SALE	CNT
1	Japan	Adventures with Numbers	1999	2653,73	196
2	United Kingdom	Adventures with Numbers	2000	3093,79	240
3	Australia	Adventures with Numbers	2000	1429,25	111
4	Spain	Adventures with Numbers	2001	997,18	77
5	France	Adventures with Numbers	2001	1778,42	137
6	Denmark	Adventures with Numbers	1999	899,46	66
7	Argentina	Adventures with Numbers	1999	45,67	3
8	Italy	Adventures with Numbers	2000	1925,02	150
9	Japan	Adventures with Numbers	2000	2907,54	227
10	Denmark	Adventures with Numbers	200		
11	United States ...	Adventures with Numbers	199		
12	Italy	Adventures with Numbers	199		
13	France	Adventures with Numbers	199		
14	Singapore	Adventures with Numbers	199		
15	United Kingdom	Adventures with Numbers	199		
16	Spain	Adventures with Numbers	200		
17	France	Adventures with Numbers	199		
18	Denmark	Adventures with Numbers	200		
19	United Kingdom	Adventures with Numbers	200		
20	Australia	Adventures with Numbers	200		
21	Japan	Adventures with Numbers	199		
22	Australia	Adventures with Numbers	199		
23	Denmark	Adventures with Numbers	199		
24	Turkey	Adventures with Numbers	199		
25	Italy	Adventures with Numbers	199		
26	Germany	Adventures with Numbers	1999	4113,57	288

	PROD	J_1998	J_1999	J_2000	J_2001
1	Adventures with Numbers	35771,66	41459,76	44832,21	53500,29
2	Bounce	(null)	51241,47	89730,94	103623,24
3	CD-R Mini Discs	91540,88	97693,58	105875,74	89443,42
4	CD-R, Professional Grade, Pack of 10	40338,17	42709,61	44657,1	42565,25
5	CD-R with Jewel Cases, pACK OF 12	39241,12	47838,75	42805,3	40520,59
6	CD-RW, High Speed, Pack of 10	10505,43	32282,47	35780,7	27899,81
7	CD-RW, High Speed Pack of 5	56622,64	59080,74	62234,62	52295,35
8	Comic Book Heroes	(null)	32296,23	32592,31	36326,06
9	Deluxe Mouse	70288,37	111552,79	53224,73	142334,42
10	DVD-R Disc with Jewel Case, 4.7 GB	27333,37	47654,82	242031,17	260401,26
11	DVD-R Discs, 4.7GB, Pack of 5	163865,43	211207,92	209852,9	319610,95
12	DVD-RAM Jewel Case, Double-Sided, 9.4G	(null)	40372,42	43167,62	36855,77
13	DVD-RW Discs, 4.7GB, Pack of 3	624,82	106440,58	93274,76	111696,38
14	Endurance Racing	97259,84	128439,01	124192,35	156539,41
15	Envoy Ambassador	5477218,04	2502740,15	3578027,71	3453656,62

Pivot und Unpivot
Noch mehr Funktionalität



- Mehrere Spalten:

```
... PIVOT (sum(quantity_sold)
    FOR (channel, quarter) IN ...
```

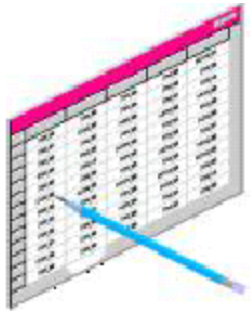
- Mehrere Aggregatsfunktionen

```
... PIVOT (SUM(amount_sold) AS sum1,
           SUM(quantity_sold) AS sum2
    FOR channel IN ...
```

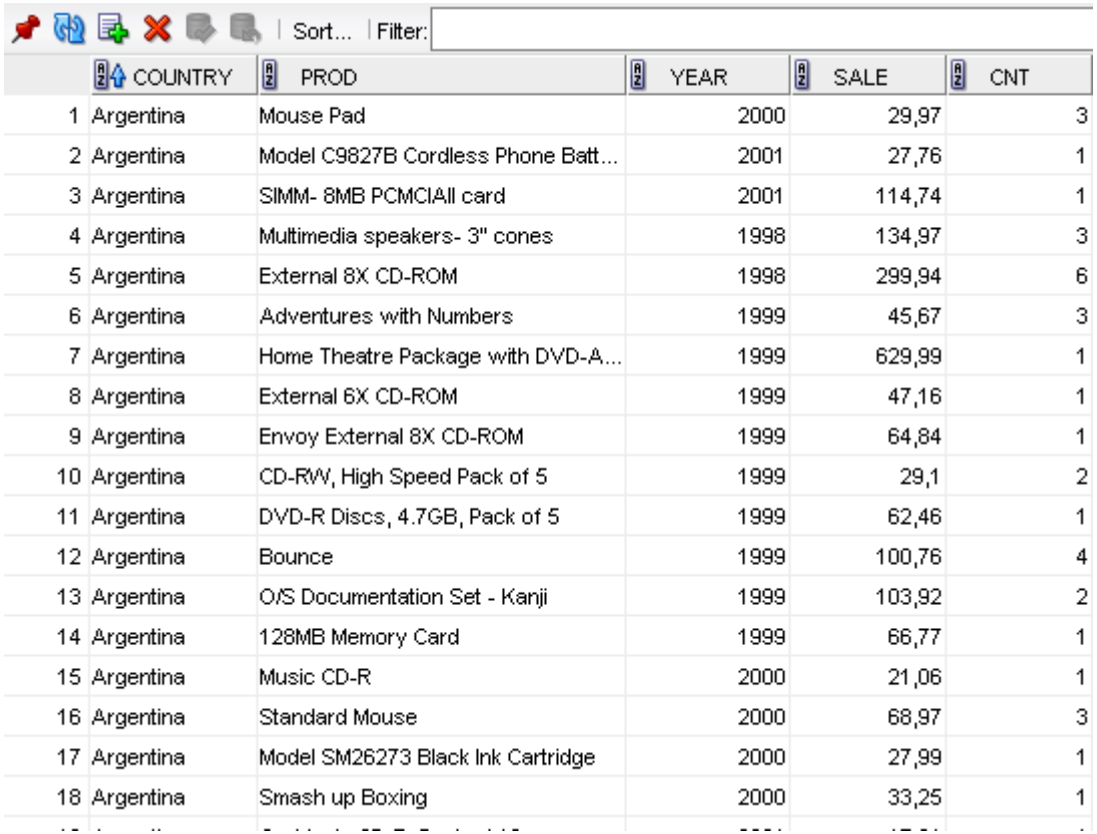
- Subqueries bzw. Wildcards für die IN-Klausel möglich (Ergebnis wird im XML Format ausgeliefert!)

SQL MODEL Klausel

- Nutzbar für
  - Komplexe Berechnungen
  - in SQL und PL/SQL
- Ersetzt komplizierte ...
  - ... JOIN- und UNION-Operationen
  - ... Berechnungen in Tabellenkalk.-Programmen
- Beispiele
  - Marktanteile,
  - Budgetberechnungen
  - Umsatzprognosen



MODEL-Berechnungen: Ein Beispiel



Sort... | Filter:

	COUNTRY	PROD	YEAR	SALE	CNT
1	Argentina	Mouse Pad	2000	29,97	3
2	Argentina	Model C9827B Cordless Phone Batt...	2001	27,76	1
3	Argentina	SIMM- 8MB PCMCIAII card	2001	114,74	1
4	Argentina	Multimedia speakers- 3" cones	1998	134,97	3
5	Argentina	External 8X CD-ROM	1998	299,94	6
6	Argentina	Adventures with Numbers	1999	45,67	3
7	Argentina	Home Theatre Package with DVD-A...	1999	629,99	1
8	Argentina	External 6X CD-ROM	1999	47,16	1
9	Argentina	Envoy External 8X CD-ROM	1999	64,84	1
10	Argentina	CD-RW, High Speed Pack of 5	1999	29,1	2
11	Argentina	DVD-R Discs, 4.7GB, Pack of 5	1999	62,46	1
12	Argentina	Bounce	1999	100,76	4
13	Argentina	O/S Documentation Set - Kanji	1999	103,92	2
14	Argentina	128MB Memory Card	1999	66,77	1
15	Argentina	Music CD-R	2000	21,06	1
16	Argentina	Standard Mouse	2000	68,97	3
17	Argentina	Model SM26273 Black Ink Cartridge	2000	27,99	1
18	Argentina	Smash up Boxing	2000	33,25	1

MODEL-Berechnungen: Beispiel

- P für **Partition**
- D für **Dimension**
- M für Measure

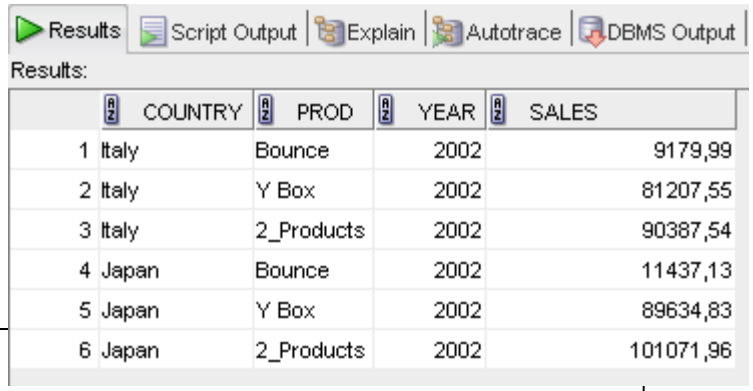
P	D	D	M	
	COUNTRY	PROD	YEAR	SALE	CNT
1	Argentina	Mouse Pad	2000	29,97	3
2	Argentina	Model C9827B Cordless Phone Batt...	2001	27,76	1
3	Argentina	SIMM- 8MB PCMCIAll card	2001	114,74	1
4	Argentina	Multimedia speakers- 3" cones	1998	134,97	3
5	Argentina	External 8X CD-ROM	1998	299,94	6
6	Argentina	Adventures with Numbers	1999	45,67	3
7	Argentina	Home Theatre Package with DVD-A...	1999	629,99	1
8	Argentina	External 6X CD-ROM	1999	47,16	1
9	Argentina	Envoy External 8X CD-ROM	1999	64,84	1
10	Argentina	CD-RW, High Speed Pack of 5	1999	29,1	2
11	Argentina	DVD-R Discs, 4.7GB, Pack of 5	1999	62,46	1
12	Argentina	Bounce	1999	100,76	4
13	Argentina	O/S Documentation Set - Kanji	1999	103,92	2
14	Argentina	128MB Memory Card	1999	66,77	1
15	Argentina	Music CD-R	2000	21,06	1
16	Argentina	Standard Mouse	2000	68,97	3
17	Argentina	Model SM26273 Black Ink Cartridge	2000	27,99	1
18	Argentina	Smash up Boxing	2000	33,25	1
19	Argentina	OraMusic CD-R, Pack of 10	2001	17,21	1
20	Argentina	Music CD-R	2001	18,67	1
21	Argentina	O/S Documentation Set - English	1998	134,97	3
22	Argentina	Model K8822S Cordless Phone Batt...	1999	215,66	6
23	Argentina	Mouse Pad	1999	21,37	2

MODEL-Berechnungen: Syntax

- Spalten einer Tabelle werden in 3 Gruppen aufgeteilt:
  - PARTITION: logischer Bereich der Resultatmenge
    - Beispiel: COUNTRY
  - DIMENSION: Identifiziert Zelle
    - Beispiel: PROD, YEAR
  - MEASURE: Werte, die berechnet werden
    - Beispiel: SALE
- Regeln werden folgendermaßen angegeben

```
sale['Bottle',2004]     =
   sale[prod='Bottle',year=2001] + sale[prod='Bottle',year=2000]
sale['Hair Dryer',2004] =
   sale[prod='Hair Dryer',year=2000]
```

MODEL-Berechnungen
Ein Beispiel


Results | Script Output | Explain | Autotrace | DBMS Output

Results:

	COUNTRY	PROD	YEAR	SALES
1	Italy	Bounce	2002	9179,99
2	Italy	Y Box	2002	81207,55
3	Italy	2_Products	2002	90387,54
4	Japan	Bounce	2002	11437,13
5	Japan	Y Box	2002	89634,83
6	Japan	2_Products	2002	101071,96

```
SELECT country, prod, year, sales
FROM sales_view
WHERE country IN ('Italy','Japan')
MODEL RETURN UPDATED ROWS
      PARTITION BY (country)
      DIMENSION BY (prod, year)
      MEASURES (sale sales)
RULES (
sales['Bounce',2002]     = sales['Bounce',2001]+sales['Bounce',2000],
sales['Y Box',2002]      = sales['Y Box',2001],
sales['2_Products',2002] = sales['Bounce',2002]+sales['Y Box',2002]
)
ORDER BY country, prod, year;
```

MODEL Berechnungen
Weitere Beispiele für Regeln

- Regel für ganze Bereiche ...

```
sales[FOR prod in('Mouse Pad','Bounce','Y Box'),2005] =
  1.3 * sales[cv(prod), 2001]
```

	COUNTRY	PROD	YEAR	SALES
1	Italy	Bounce	2005	6300,19
2	Italy	Mouse Pad	2005	6172,27
3	Italy	Y Box	2005	105569,815

- Loop-Konstrukte

```
sales['Mouse Pad', FOR year FROM 2005 TO 2012 INCREMENT 2] =
  1.2 * sales[cv(prod), 2001]
```

	COUNTRY	PROD	YEAR	SALES
1	Italy	Mouse Pad	2005	5697,48
2	Italy	Mouse Pad	2007	5697,48
3	Italy	Mouse Pad	2009	5697,48
4	Italy	Mouse Pad	2011	5697,48

Reguläre Ausdrücke in SQL und PL/SQL

- REGEXP_LIKE — Existiert das Muster?
- REGEXP_INSTR — Position des Musters?
- REGEXP_SUBSTR — Muster extrahieren
- REGEXP_REPLACE — Muster ersetzen
- REGEXP_COUNT — Wie oft existiert das Muster?

```
select * from message_table
where REGEXP_LIKE(message, 'ORA-.[0-9]{5}')

select REGEXP_SUBSTR(
 'Max Mustermann, 80992 München',
 '[[:digit:]]{5}')
from dual
```

Reguläre Ausdrücke:
Weitere Beispiele

- Reduziert komplexe Logik auf eine Zeichenkette
- Email-Adresse:
  `^[a-z0-9\-_]?[a-z0-9.\-_]+[a-z0-9\-_]?@[a-z.-]+\.[a-z]{2,}$`
- URL:
  `^http[s]?://[-a-zA-Z0-9_.:]+[-a-zA-Z0-9_:@&?=+,.!/~*'%$]*$`
- KFZ-Kennzeichen
  `^[A-Z]{1,3}-[A-Z]{1,2} [0-9]{1,4}$`
- Uhrzeitformat (24h)
  `^([01]?[0-9]|2[0123])[:]?([0-5][0-9])$`

Ausführungspläne betrachten
So geht's überall ...

- Führe EXPLAIN PLAN aus

```
EXPLAIN PLAN FOR
SELECT ....
```

- Einstellungen wählen
  - BASIC, **TYPICAL (Default),** ALL, SERIAL
  - Filterkriterien möglich: Rows, Note, Projection,...

```
SELECT * FROM table
       (dbms_xplan.display())
       (dbms_xplan.display('PLAN_TABLE',null, 'BASIC'))
       (dbms_xplan.display(null,null,'BASIC +NOTE'))
```

Ausgabe von DISPLAY am Beispiel
Was kann man daraus erkennen ...?

```
EXPLAIN PLAN FOR SELECT ...

SELECT * from table(dbms_xplan.display(null,null,'BASIC+NOTE'));

-------------------------------------------
| Id  | Operation                   | Name  |
-------------------------------------------
|   0 | SELECT STATEMENT            |       |
|   1 |  TABLE ACCESS BY INDEX ROWID| COMP  |
|   2 |   INDEX RANGE SCAN          | I_ORT |
-------------------------------------------

Note
-----
   - dynamic sampling used for this statement
   - SQL plan baseline "SYS_SQL_PLAN_56415425c55874be" used for
     this Statement
```

Weitere Funktion von DBMS_XPLAN
DISPLAY_CURSOR

- Ausführungsplan ... ohne EXPLAIN PLAN
- Zeigt Informationen aus dem Cursor Cache
- Zusätzliche Memory Statistiken sind möglich
- Voraussetzung: Dictionary-Privilegien
  - Rechte auf `V$SQL_PLAN_STATISTICS_ALL`, `V$SQL` und `V$SQL_PLAN`

```
SELECT * FROM ...

select * from table(
  dbms_xplan.display_cursor()
);
```

Weitere Informationen

- Oracle Dokumentation
  - SQL Language Reference
  - Data Warehousing Guide
  - Application Developers' Guide
- Blog des Autors
  http://sql-plsql-de.blogspot.com

Carsten.Czarski@oracle.com

http://tinyurl.com/apexcommunity

http://sql-plsql-de.blogspot.com
http://oracle-text-de.blogspot.com
http://oracle-spatial.blogspot.com

http://plsqlexecoscomm.sourceforge.net
http://plsqlmailclient.sourceforge.net

Twitter: @cczarski @oraclebudb