Training Resources

Futurecast with Oracle Model Clause

COLLABORATE 10

Technology and Applications Forum for the Oracle Community

IOUG OAUG Quest

Presented by: John Jay King

King Training Resources - john@kingtraining.com

Download this paper from: http://www.kingtraining.com

Session Objectives

- Learn how to use the SQL Model clause
- Be ready to use various options of Model to represent query data in a “spreadsheet”
- Use Model to create predictions of future values

Model Clause

- The SQL Model clause is a powerful extension of the SELECT statement (new with Oracle 10g)
- Model provides the ability to present the output of a SELECT in the form of multi-dimensional arrays (like a spreadsheet) and apply formulas to the array (cell) values
- The Model clause defines a multidimensional array by mapping the columns of a query into three groups: partitioning, dimension, and measure columns
  - Partitions
  - Dimensions
  - Measures

Partitions, Dimensions, Measures

- Partitions define logical blocks of the result set
  - Similar to partitions with analytical functions
  - Each partition used by formulas as an independent array; Model rules are applied the cells of each partition
- Dimensions identify Measure cells within Partitions
- Each Measure column identifies characteristics such as date, region and product name (similar to measures in a star schema fact table)
  - Measures normally contain numeric values such as sales units or cost
  - Each cell is accessed within its partition by specifying its full combination of dimensions

Model Clause Syntax

```
MODEL  [<global reference options>]
[<reference models>]
[MAIN <main-name>]
[PARTITION BY (<cols>)]
DIMENSION BY (<cols>)
MEASURES (<cols>)
[<reference options>]
[RULES] <rule options>
(<rule>, <rule>,.., <rule>)
<global reference options> ::= <reference options> <ret-opt>
<ret-opt> ::= RETURN {ALL|UPDATED} ROWS
<reference options> ::=[IGNORE NAV | [KEEP NAV]
[UNIQUE DIMENSION | UNIQUE SINGLE REFERENCE]
<rule options> ::=
[UPSERT | UPDATE]
[AUTOMATIC ORDER | SEQUENTIAL ORDER]
[ITERATE (<number>) [UNTIL <condition>]]
<reference models> ::= REFERENCE ON <ref-name> ON (<query>)
DIMENSION BY (<cols>) MEASURES (<cols>) <reference options>
```

How Model Fits in SQL

- Model is evaluated after all clauses except: SELECT DISTINCT and ORDER BY
- When using Model the SELECT and ORDER BY may not contain aggregates or analytic functions
- Aggregates and analytic functions may be specified in PARTITION, DIMENSION, and MEASURES lists but must be given alias names; the alias names may be used in SELECT or ORDER BY
- Only columns listed as MEASURES may be updated

Model and Subqueries

- Subqueries are not allowed in RULES (except in FOR constructs, see below); however subqueries may be included in MEASURES if given an alias name
- Subqueries may be used as part of the FOR construct on the left-hand side of RULES provided that:
  - Subquery returns less than 10,000 rows
  - Subquery is not correlated
  - Subquery may not be defined using WITH
  - Subquery cursor must be shareable

Oracle Goodies

- The Oracle Database Data Warehousing Guide provides many explanations and examples of the Model clause and its use
- The examples in these notes were based upon the Oracle-supplied “SH” schema’s data
- The example on the next two pages uses a “sales_view” definition from the Oracle manual
- On a later page is a Materialized View definition used for the code examples in this paper

Model Example

```
SELECT substr(country,1,20) country,substr(prod,1,15) prod,
      year,sales  FROM sales_view
  WHERE country IN ('Canada','Germany')
  MODEL RETURN UPDATED ROWS
     PARTITION BY (country)
     DIMENSION BY (prod, year)
     MEASURES (sale sales)
     RULES (sales['ZooperT',2002] = sales['ZooperT',2001]
                                    + sales['ZooperT',2000],
           sales['HulaWhirl',2002] = sales['HulaWhirl',2001],
           sales['HulaZoop Pkg',2002] = sales['ZooperT',2002]
                                    + sales['HulaWhirl',2002])

COUNTRY              PROD                 YEAR          SALES
-------------------- ---------------      ----     ----------
Canada               HulaZoop Pkg         2002       92613.16
Canada               Zoop Tube            2002        9299.08
Canada               Hula Twirl           2002       83314.08
Germany              HulaZoop Pkg         2002      103816.60
Germany              Zoop Tube            2002       11631.13
Germany              Hula Twirl           2002       92185.47
```

Model Example Explained

- The statement on the preceding page calculates sales values for two products and defines sales for a new product based upon the other two products
  - Statement partitions data by country
    - Formulas are applied to one country at a time
    - Sales fact data ends with 2001, any rules defining values for 2002 or later will insert new “Updated” cells
  - First rule defines sales of “Zoop Tube” game in 2002 as the sum of its sales in 2000 and 2001
  - The second rule defines sales for “Hula Twirl” in 2002 to be the same value they were for 2001
  - Third rule defines "HulaZoop Pkg" that is the sum of the Zoop Tube and Hula Twirl values for 2002 -- the rules for Zoop Tube and Hula Twirl must be executed before the HulaZoop Pkg rule

Rules may perform calculations and/or call functions

Creating the MView

```
CREATE materialized VIEW sales_mview AS
   SELECT  substr(country_name,1,20) country
          ,substr(prod_name,1,15)    product
          ,calendar_year             year
          ,SUM(amount_sold)          tot_amt
          ,SUM(quantity_sold)        tot_qty
          ,COUNT(amount_sold)        tot_sales
        FROM sh.sales join sh.times
                        on sales.time_id = times.time_id
                    join sh.products
                        on sales.prod_id = products.prod_id
                    join sh.customers
                        on sales.cust_id = customers.cust_id
                    join sh.countries
                        on customers.country_id = countries.country_id
   GROUP BY country_name
                ,prod_name
                ,calendar_year
        ORDER BY country
                ,product
                ,year
```

Query Existing Data

- The query below retrieves data from two products and two countries for all years in the existing data

```
select country
        ,product
        ,year
        ,round(tot_sales,0) sales
  from sales_mview
  where country in ('Australia','Canada')
    and product in ('Mouse Pad','Deluxe Mouse')
  order by country
          ,product
          ,year
```

Existing Data

```
COUNTRY              PRODUCT                YEAR      SALES
-------------------- --------------- ---------- ----------
Australia            Deluxe Mouse           1998         86
Australia            Deluxe Mouse           1999        140
Australia            Deluxe Mouse           2000         78
Australia            Deluxe Mouse           2001        211
Australia            Mouse Pad              1998        195
Australia            Mouse Pad              1999        311
Australia            Mouse Pad              2000        264
Australia            Mouse Pad              2001        332
Canada               Deluxe Mouse           1998         65
Canada               Deluxe Mouse           1999        109
Canada               Deluxe Mouse           2000         38
Canada               Deluxe Mouse           2001         61
Canada               Mouse Pad              1998         93
Canada               Mouse Pad              1999        174
Canada               Mouse Pad              2000        144
Canada               Mouse Pad              2001        186
```

Predicting the Future

- The existing data goes through 2001, what if we want to project future sales?
- The Model clause allows the creation of a “spreadsheet” layout where each result represents a “cell”
- The rules sub-clause allows us to establish a set of rules for treating cell-values and even predict future values

First Model

```
model return all rows
   partition by (country)
   dimension by (product,year)
   measures (tot_sales sales)
   rules (
    sales['Mouse Pad',2002] =
               sales['Mouse Pad',2001] * 1.1
   ,sales['Deluxe Mouse',2002] =
               sales['Deluxe Mouse',2001] * 1.3
   )
```

- Return all rows (both existing and new)
- Group (partition) rows by country
- Define product and year as dimension values
- Define tot_sales as the value of each cell, rename “sales”
- Use rules to set sales for 2002 (a new year) to 2001’s value times some multiplier (I made them up…)

Model Output Rows

- The Model clause may specify how the output of the Model is to be displayed

– RETURN UPDATED ROWS	Statement output includes only rows created by Model
– RETURN ALL ROWS (default)	Statement output includes all rows from query plus rows created by Model

RETURN ALL ROWS

```
select country
      ,product
      ,year
      ,round(sales,0) sales
from sales_mview
where country in ('Australia','Canada')
  and product in ('Mouse Pad','Deluxe Mouse')
model return all rows
   partition by (country)
   dimension by (product,year)
   measures (tot_sales sales)
   rules (    sales['Mouse Pad',2002] =
                           sales['Mouse Pad',2001] * 1.1
              ,sales['Deluxe Mouse',2002] =
                           sales['Deluxe Mouse',2001] * 1.3 )
order by country,product,year
```

ALL ROWS Model Results

```
COUNTRY              PRODUCT               YEAR      SALES
-------------------- --------------- ---------- ----------
Australia            Deluxe Mouse          1998         86
Australia            Deluxe Mouse          1999        140
Australia            Deluxe Mouse          2000         78
Australia            Deluxe Mouse          2001        211
Australia            Deluxe Mouse          2002        274
Australia            Mouse Pad             1998        195
Australia            Mouse Pad             1999        311
Australia            Mouse Pad             2000        264
Australia            Mouse Pad             2001        332
Australia            Mouse Pad             2002        365
Canada               Deluxe Mouse          1998         65
Canada               Deluxe Mouse          1999        109
Canada               Deluxe Mouse          2000         38
Canada               Deluxe Mouse          2001         61
Canada               Deluxe Mouse          2002         79
Canada               Mouse Pad             1998         93
Canada               Mouse Pad             1999        174
Canada               Mouse Pad             2000        144
Canada               Mouse Pad             2001        186
Canada               Mouse Pad             2002        205
```

UPDATED ROWS only

```
select country
      ,product
      ,year
      ,round(units_sold,0) units_sold
from sales_mview
where country in ('Australia','Canada')
  and product in ('Mouse Pad','Deluxe Mouse')
model return updated rows
  partition by (country)
  dimension by (product,year)
  measures (tot_sales units_sold)
  rules (    units_sold['Mouse Pad',2002] =
                   units_sold['Mouse Pad',2001] * 1.1
            ,units_sold['Deluxe Mouse',2002] =
                   units_sold['Deluxe Mouse',2001] * 1.3 )
order by country,product,year
```

Updated Rows

```
COUNTRY                  PRODUCT                YEAR UNITS_SOLD
-------------------- --------------- ---------- ----------
Australia            Deluxe Mouse          2002        274
Australia            Mouse Pad             2002        365
Canada               Deluxe Mouse          2002         79
Canada               Mouse Pad             2002        205
```

Model-Related SQL Functions

• CV	Current value of dimension in Model clause
• ITERATION_NUMBER	Returns iteration number in Model clause rules
• PRESENTNNV	Present Value of cell in Model clause (nulls converted)
• PRESENTV	Present Value of cell in Model clause
• PREVIOUS	Returns cell value at beginning of Model clause iteration

Note: These functions are invalid anywhere in SQL except as part of a Model clause

CV

- The CV function provides the current value of a cell and may only be used on the right-hand side of a Model clause Rule

– CV()	Returns the current value of the dimension column in the same position Rule’s left-hand side
– CV(dimcol)	Returns the current value of the named dimension column

ITERATION_NUMBER

- ITERATION_NUMBER may be used only when ITERATE(number) is used in a Model clause Rule
- ITERATION_NUMBER has no parameters/arguments
- ITERATION_NUMBER returns the integer value of the last completed iteration through the Model Rules (returns 0 in first iteration)
- It returns an integer representing the last completed iteration through the Model Rules (current iteration plus 1); ITERATION_NUMBER returns 0 during the first iteration

PRESENTNNV & PRESENTV

- PRESENTV and PRESENTNNV may be used only on the right-hand side of a Model Rule to
- Both PRESENTV and PRESENTNNV use the same syntax:

```
PRESENTV(cellref,cell_exists,cell_doesnotexist)
PRESENTNNV(cellref,cell_exists,cell_doesnotexist)
```

  – PRESENTV tests if the referenced cell existed prior to Model clause execution
    - If so the first expression (cell_exists above) is executed
    - If the referenced cell did not exist prior to Model clause execution the second expression (cell_doesnotexist above) is executed

  – PRESENTNNV also tests if the referenced cell existed prior to Model clause execution but also checks to see if the existing cell had a null value;
    - If the referenced cell existed and was not null prior to the Model clause the first expression (cell_exists above) is executed
    - If the referenced cell did not exist prior to Model clause execution or did exist and was null the second expression (cell_doesnotexist above) is executed

PREVIOUS

- PREVIOUS may only be used as part of ITERATE...UNTIL in a Model Rule
- PREVIOUS returns the value of the referenced cell at the beginning of the iteration

```
PREVIOUS(cell-reference)
```

Model-specific Conditions

- Two conditions have been added to SQL that are allowed only in a Model clause
- dimension_column IS ANY or ANY are used to include all values from a dimension column including NULLs (always TRUE)
- cell_reference IS PRESENT returns TRUE if the referenced cell is present before the Model clause is executed; FALSE if it is not

Reference Options

- The Model clause provides several keywords that may be specified at a GLOBAL level or at a LOCAL level
  - IGNORE NAV
  - KEEP NAV
  - UNIQUE DIMENSION
  - UNIQUE SINGLE REFERENCE

IGNORE NAV & KEEP NAV

- IGNORE NAV and KEEP NAV control whether or not cells not provided by the query result set are treated as zero by calculations

– KEEP NAV	Unavailable cell values are not changed (default)
– IGNORE NAV	Unavailable numeric cell values are treated as:

    - 0 for numeric data
    - Empty string for character data
    - '01-JAN-2001' for date data
    - NULL for other data types

Unique Cells

- UNIQUE DIMENSION and UNIQUE SINGLE REFERENCE control the checking for cell uniqueness
  - UNIQUE DIMENSION (default) Requires that the combination of PARTITION & DIMENSION columns must uniquely identify each cell in the model
  - UNIQUE SINGLE REFERENCE Requires that the PARTITION & DIMENSION columns must uniquely identify single cells on the right-hand side of Model Rules

Rule Options

- The Model clause provides several Rule keywords that may be specified at a GLOBAL level or at a LOCAL level
  - UPDATE
  - UPSERT
  - UPSERT ALL
  - AUTOMATIC ORDER
  - SEQUENTIAL ORDER

UPSERT & UPDATE

- UPSERT, UPSERT ALL, and UPDATE control whether updates will occur if the cell on the left-hand side of a Rule does not exist or if the cell reference is not positional

– UPSERT (default)	Updates cell values if the cell exists and creates the cell if it does not exist and uses positional (non-symbolic) cell notation
– UPSERT ALL	Like UPSERT but allows creation of new cells with use of ANY
– UPDATE	Updates existing cells only

Order of RULE Execution

- AUTOMATIC ORDER and SEQUENTIAL ORDER control the order of Rule execution

– AUTOMATIC ORDER	Oracle decides sequence of Rule execution
– SEQUENTIAL ORDER (default)	Rules execute in order specified by Model clause

Recapping Options

- SEQUENTIAL ORDER rules are defined in the order they appear in the rules sub-clause
- AUTOMATIC ORDER rules are considered according to dependencies
- IGNORE NAV treats missing values as:
  - 0 for numeric data
  - Empty string for character data
  - ’01-JAN-2001’ for date data
  - NULL for other data types
- KEEP NAV treats nulls normally
- UNIQUE DIMENSION (default), PARTITION BY and DIMENSION BY columns must uniquely identify each and every cell
- UNIQUE SINGLE REFERENCE, PARTITION BY and DIMENSION BY uniquely identify single point references on the right-hand side of the rules
- ITERATE (n) iterates rules specified number of times, ITERATION_NUMBER returns current value (starts with 0)

Projecting into the Future

- The Model clause includes a special FOR construct allowing the modification and/or creation of many new rows (called UPSERT)
  - Values may range as desired
  - Increment and/or decrement value
  - Use cv() function to use a cell's current value
  - UPSERT is limited to 10,000 rows

```
FOR  dimension_value
          FROM lowval TO hival

         INCREMENT | DECREMENT incrval
```

Projection SQL

```
select country,product,year,round(units_sold,0) units_sold
from sales_mview
where country in ('Australia','Canada')
  and product in ('Mouse Pad','Deluxe Mouse')
model return updated rows
  partition by (country)
  dimension by (product,year)
  measures (tot_sales units_sold)
  rules (
    units_sold['Mouse Pad',
        for year from 2001 to 2005 increment 1]
         = units_sold['Mouse Pad',cv(year)-1] * 1.1
   ,units_sold['Deluxe Mouse',
        for year from 2001 to 2005 increment 1]
         = units_sold['Deluxe Mouse',cv(year)-1] * 1.3
   )
order by country,product,year
```

Projection Results

```
COUNTRY              PRODUCT               YEAR UNITS_SOLD
-------------------- --------------- ---------- ----------
Australia            Deluxe Mouse          2001        101
Australia            Deluxe Mouse          2002        132
Australia            Deluxe Mouse          2003        171
Australia            Deluxe Mouse          2004        223
Australia            Deluxe Mouse          2005        290
Australia            Mouse Pad             2001        290
Australia            Mouse Pad             2002        319
Australia            Mouse Pad             2003        351
Australia            Mouse Pad             2004        387
Australia            Mouse Pad             2005        425
Canada               Deluxe Mouse          2001         49
Canada               Deluxe Mouse          2002         64
Canada               Deluxe Mouse          2003         83
Canada               Deluxe Mouse          2004        109
Canada               Deluxe Mouse          2005        141
Canada               Mouse Pad             2001        158
Canada               Mouse Pad             2002        174
Canada               Mouse Pad             2003        192
Canada               Mouse Pad             2004        211
Canada               Mouse Pad             2005        232
```

Oracle Documentation

- Oracle10g and Oracle 11g
  - Oracle Database Data Warehousing Guide
  - SQL Reference
  - PL/SQL User’s Guide and Reference
  - Application Developer's Guide - Object-Relational Features
- Lots of papers and examples:
  http://technet.oracle.com
  http://tahiti.oracle.com

Wrapping it all Up

- The Oracle 10g Model clause adds new capabilities to better serve the users of our data
- The Model clause’s ability to provide the data in a cell-by-cell “spreadsheet” makes output more readable than ever before
- Rules provide the ability to add new cells to the Model based upon calculations performed on existing data cells

RMOUG
ROCKY MOUNTAIN ORACLE USERS GROUP

Training Days 2011

Save the dates!

February 15-17 2011

ODTUG Kaleidoscope 2010

ODTUG
2010
KALEIDOSCOPE
Washington, D.C. • June 27-July 1 • Marriott Wardman Park Hotel

June 27-July 1 2010
Washington, DC

Please fill out session Evaluations

Futurecast with Oracle Model Clause

To contact the author:

John King

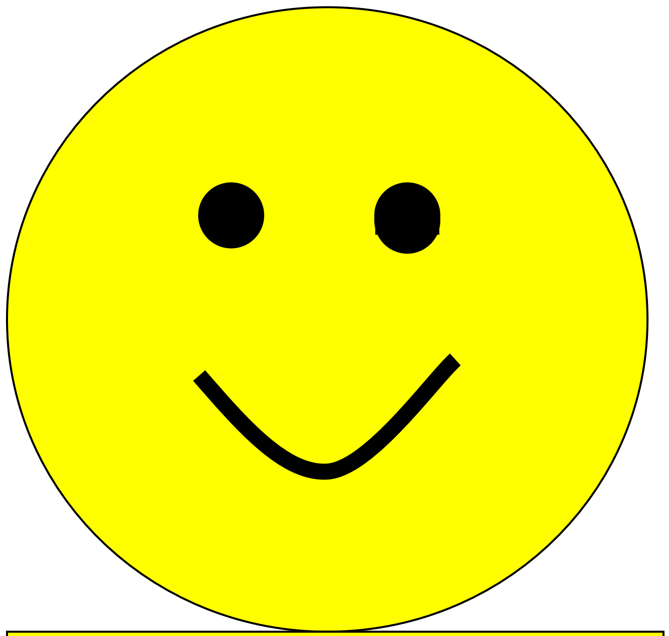
King Training Resources

6341 South Williams Street

Littleton, CO 80121-2627 USA

1.800.252.0652 - 1.303.798.5727

Email: john@kingtraining.com



Thanks for your attention!

Today's slides are on the web:

http://www.kingtraining.com