Oracle Connect By Queries

# Hierarchical Queries

Oracle’s Connect By clause is used to traverse hierarchical data.

Description of hierarchical_query_clause.gif follows

Figure - CONNECT BY clause

Example:

SELECT employee\_id, last\_name, manager\_id

FROM employees

CONNECT BY PRIOR employee\_id = manager\_id;

EMPLOYEE\_ID LAST\_NAME MANAGER\_ID

----------- ------------------------- ----------

101 Kochhar 100

108 Greenberg 101

109 Faviet 108

110 Chen 108

111 Sciarra 108

112 Urman 108

113 Popp 108

200 Whalen 101

This query uses the CONNECT BY clause to define the relationship between employees and managers. In this example, we are specifying that the parent row is found by connecting the employee\_id column to the manager\_id column.

# Advanced Queries using Connect By

## Using START WITH to specify a root

By using START WITH we can begin our traversal of the data at any level. In the following example, we will start our query with a certain employee, and return everyone that reports to that employee:

SELECT last\_name, employee\_id, manager\_id, LEVEL

FROM employees

START WITH employee\_id = 100

CONNECT BY PRIOR employee\_id = manager\_id

ORDER SIBLINGS BY last\_name;

LAST\_NAME EMPLOYEE\_ID MANAGER\_ID LEVEL

------------------------- ----------- ---------- ----------

King 100 1

Cambrault 148 100 2

Bates 172 148 3

Bloom 169 148 3

Fox 170 148 3

Kumar 173 148 3

Ozer 168 148 3

Smith 171 148 3

De Haan 102 100 2

Hunold 103 102 3

Austin 105 103 4

Ernst 104 103 4

Lorentz 107 103 4

Pataballa 106 103 4

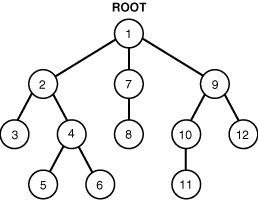
Errazuriz 147 100 2

Ande 166 147 3

Banda 167 147 3

## Using LEVEL to see where you are in the hierarchy

The keyword LEVEL specifies how far down you are in the hierarchy, with the top being 1, and everything connected to that is 2, and so forth.



In the diagram above Root is at level 1, 2/7/9 are at level 2, and 3/4/5/10/12 are at level 3.

Level can also be used to generate sequential series.

Consider the following:

SELECT level

FROM dual

CONNECT BY level <= 10

This will return a list of numbers from one to ten.

With a slight modification:

SELECT trunc(sysdate) + level – 1 as the\_days

FROM dual

CONNECT BY level <= 10

This will give us a list of the next 10 days from today.

# Important Keywords for use with CONNECT BY

### NOCYCLE

This prevents infinite loops in your queries. Often in hierarchical data, the top level connects to itself. This would cause an infinite loop if allowed when trying to query this data. To prevent this, use the NOCYCLE keyword

SELECT employee\_id, last\_name, manager\_id

FROM employees

CONNECT BY NOCYCLE PRIOR employee\_id = manager\_id;

### ORDER SIBLINGS BY

This can be used to order siblings (or leaves) in your query without messing up the order of the results.

SELECT employee\_id, last\_name, manager\_id

FROM employees

CONNECT BY NOCYCLE PRIOR employee\_id = manager\_id

Order siblings by last\_name;

### CONNECT\_BY\_ROOT and SYS\_CONNECT\_BY\_PATH

select LPAD('.',LEVEL\*3-3,'.') || name as the\_name, -- Pad the name depending on the level

id,

connect\_by\_root name as top\_dog, -- Get the name of the top person in the hierarchy

sys\_connect\_by\_path(id,'->'), -- show how the person connects to the top via S#

sys\_connect\_by\_path(name,'->'), -- show how the person connects to the top via name

level -- The level of the hierarchy.

from employee

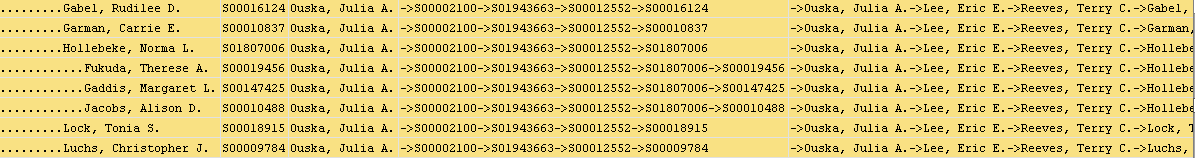
where position\_status = 'A'

start with person\_uid = 2135

connect by prior person\_uid = supervisor\_uid

order siblings by name

;



These two pseudo columns show the managers name and their place in the hierarchy.

### CONNECT\_BY\_ISCYCLE and CONNECT\_BY\_ISLEAF

CONNECT\_BY\_ISCYCLE returns 1 if this row connects to itself. Useful for finding out which rows will cause a loop.

CONNECT\_BY\_ISLEAF returns 0 if there are more rows beneath this row or a 1 if it is the end (i.e. a leaf).

# References

<http://psoug.org/reference/connectby.html>

<http://docs.oracle.com/cd/B19306_01/server.102/b14200/queries003.htm#i2069380>