

Recuperación Jerárquica

Objetivos

Después de completar este capítulo conocerás lo siguiente:

- Interpretar el concepto de consultas jerárquicas
- Crear un reporte con estructura de árbol
- El formato de datos jerárquicos
- Excluir ramas de una estructura de árbol

Sample Data from the EMPLOYEES Table

EMPLOYEE_ID	LAST_NAME	JOB_ID	MANAGER_ID
100	King	AD_PRES	
101	Kochhar	AD_VP	100
102	De Haan	AD_VP	100
103	Hunold	IT_PROG	103
104	Ernst	IT_PROG	103
107	Lorentz	IT_PROG	103
124	Mourgos	ST_MAN	100
141	Rajs	ST_CLERK	124
142	Davies	ST_CLERK	124
143	Matos	ST_CLERK	124
144	Vargas	ST_CLERK	124
149	Zlotkey	SA_MAN	100
174	Abel	SA_REP	149
176	Taylor	SA_REP	149
EMPLOYEE_ID	LAST_NAME	JOB_ID	MANAGER_ID
178	Grant	SA_REP	149
200	Whalen	AD_ASST	101
201	Hartstein	MK_MAN	100
202	Fay	MK_REP	201
205	Higgins	AC_MGR	101
206	Bletz	AC_ACCOUNT	205

20 rows selected.

ORACLE

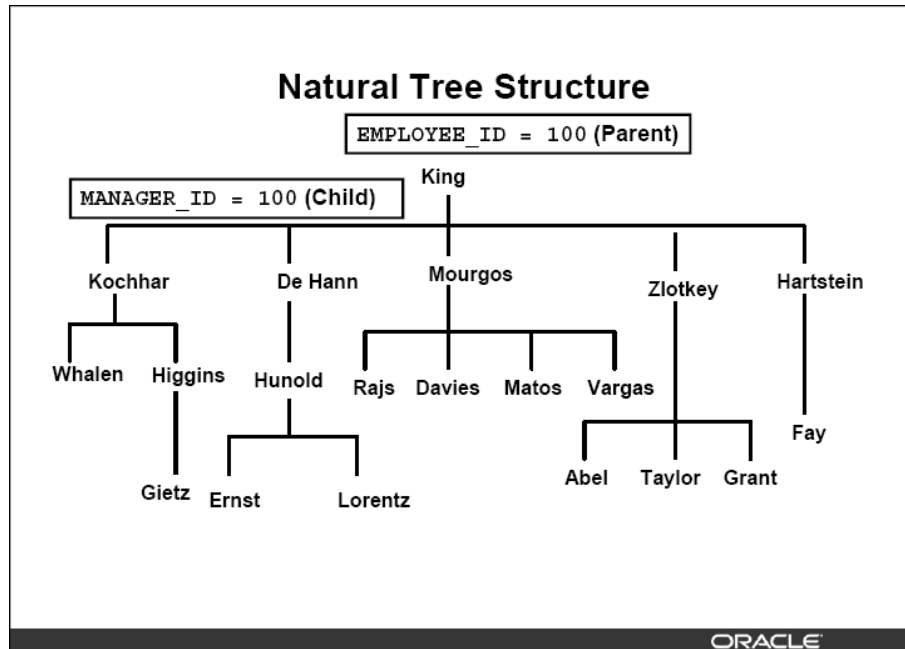
Datos de muestra de la tabla EMPLOYEES

Usando consultas jerárquicas, se pueden recuperar datos basados en una natural relación jerárquica entre filas en una tabla. Una base de datos relacional no almacena registros en un camino jerárquico. Sin embargo, donde una relación jerárquica exista entre filas de una tabla sencilla, un proceso llamado *recorrido del árbol (tree walking)* habilita la jerarquía que ha sido construida. Una consulta jerárquica es un método de reportar, en orden, las ramas de un árbol.

Imagínese que un árbol genealógico con los miembros mayores de la familia cerca de la base o tronco del árbol y los miembros mas jóvenes que representan ramas del árbol. Las ramas pueden tener sus propias ramas, etcétera.

Una consulta jerárquica es posible cuando una relación existe entre las filas de una tabla. Por ejemplo, en los datos anteriores, se ve que los empleados con el puesto de *AD_VP*, *ST_MAN*, *SA_MAN* y *MK_MAN* se reportan directamente al presidente de la compañía. Nosotros conocemos esto debido a que la columna *MANAGER_ID* de esos registros contiene el valor 100, que es el *EMPLOYEE_ID* del presidente (*AD_PRES*).

Nota: los árboles jerárquicos son usados en varios campos como la genealogía humana (árboles genealógicos), existencia de animales (para propósitos reproductivos), administración corporativa (organigrama), manufactura (ensamble de productos), investigación evolutiva (desarrollo de especies) e investigación científica.



Estructura natural de árbol

La tabla EMPLOYEES tiene una estructura de árbol representando la línea de reporte administrativo. La jerarquía puede ser creada por la relación entre valores equivalentes de las columnas EMPLOYEE_ID y MANAGER_ID. Esta relación puede ser explotada por la relación de la tabla con si misma. La columna MANAGER_ID contiene el número de empleado de los empleados administrativos.

La relación padre – hijo de la estructura de árbol habilita el control:

- De la dirección en el cuál la jerarquía es recorrida
- El punto de inicio de la jerarquía

Nota: El diagrama muestra una estructura de árbol invertido de la administración jerárquica de los empleados en la tabla EMPLOYEES.

Hierarchical Queries

```

SELECT [LEVEL], column, expr...
FROM table
[WHERE condition(s)]
[START WITH condition(s)]
[CONNECT BY PRIOR condition(s)];

```

WHERE condition:

```

expr comparison_operator expr

```

ORACLE

Consultas Jerárquicas

Palabras reservadas y cláusulas

Las consultas jerárquicas pueden ser identificadas por la presencia de las cláusulas CONNECT BY y START WITH.

Donde:

SELECT	Es el estándar de la cláusula SELECT
LEVEL	Para cada fila obtenida por la consulta jerárquica, la pseudocolumna LEVEL obtiene 1 para la fila raíz, 2 para los hijos de la raíz, etc.
FROM <i>table</i>	Especifica la tabla, vista o snapshot conteniendo las columnas. Se puede seleccionar de una sola tabla.
WHERE	Restringe las filas obtenidas por la consulta sin afectar otras filas de la jerarquía
<i>condition</i>	Es una comparación con expresiones
START WITH	Especifica la fila raíz de la jerarquía (donde inicia). Esta cláusula es requerida para una verdadera consulta jerárquica
CONNECT BY	
PRIOR	Especifica las columnas en la cual la relación entre filas padre e hijo existe. Esta cláusula es requerida para consultas jerárquicas

La sentencia SELECT no puede contener un *JOIN* o consulta de una vista que contenga un *JOIN*.

Walking the Tree

Starting Point

- Specifies the condition that must be met
- Accepts any valid condition

START WITH *column1* = *value*

Using the EMPLOYEES table, start with the employee whose last name is Kochhar.

...START WITH *last_name* = 'Kochhar'

ORACLE

Recorriendo el árbol

La fila o filas usadas como raíz del árbol se determinan con la cláusula START WITH. La cláusula START WITH puede ser usada en conjunto con alguna condición válida.

Ejemplos

Usando la tabla EMPLOYEES, inicie con “King”, el presidente de la compañía.

```
... START WITH manager_id IS NULL
```

Usando la tabla EMPLOYEES, inicie con el empleado “Kochhar”. Una condición START WITH puede contener una sub consulta.

```
... START WITH employee_id = (SELECT employee_id
                               FROM   employees
                               WHERE  last_name = 'Kochhar')
```

Si la cláusula START WITH es omitida, el recorrido del árbol es iniciado con todas las filas en la tabla como filas raíz. Si una cláusula WHERE es usada, El recorrido es iniciado con todas las filas que satisfacen la condición WHERE. Esto no es un gran reflejo en una verdadera jerarquía.

Nota: Las cláusulas CONNECT BY PRIOR y START WITH no son un estándar de ANSI SQL.

Walking the Tree

CONNECT BY PRIOR *column1* = *column2*

Walk from the top down, using the EMPLOYEES table.

... CONNECT BY PRIOR *employee_id* = *manager_id*

Direction

Top down	→	Column1 = Parent Key Column2 = Child Key
Bottom up	→	Column1 = Child Key Column2 = Parent Key

ORACLE

La dirección de la consulta, si es de padre a hijo o de hijo a padre, es determinada por la posición de las columnas en la cláusula CONNECT BY PRIOR. El operador PRIOR se refiere a la fila padre. Para encontrar los hijos de la fila padre, el Servidor de Oracle evalúa la expresión PRIOR para la fila padre y las otras expresiones para cada fila en la tabla. Las filas para la cual la condición es verdadera son los hijos del padre. El servidor de Oracle siempre selecciona hijos por la evaluación de la condición CONNECT BY con respecto a la actual fila padre.

Ejemplos

Recorra de arriba - abajo usando la tabla EMPLOYEES. Defina una relación jerárquica en el cual el valor EMPLOYEE_ID de la fila padre es igual al valor MANAGER_ID de la fila hijo.

```
... CONNECT BY PRIOR employee_id = manager_id
```

Recorra de abajo – arriba usando la tabla EMPLOYEES

```
... CONNECT BY PRIOR manager_id = employee_id
```

El operador PRIOR no es necesariamente requerido después del CONNECT BY. Así, la siguiente cláusula CONNECT BY PRIOR da el mismo resultado que uno de los ejemplos anteriores.

```
... CONNECT BY employee_id = PRIOR manager_id
```

Nota: La cláusula CONNECT BY no puede contener una sub consulta.

Walking the Tree: From the Bottom Up

```
SELECT employee_id, last_name, job_id, manager_id
FROM employees
START WITH employee_id = 101
CONNECT BY PRIOR manager_id = employee_id ;
```

EMPLOYEE_ID	LAST_NAME	JOB_ID	MANAGER_ID
101	Kochhar	AD_VP	100
100	King	AD_PRES	

ORACLE

Recorriendo el árbol de abajo – arriba

En el ejemplo se despliega una lista de administradores iniciando con el empleado cuyo EMPLOYEE_ID es 101.

Ejemplo

En el siguiente ejemplo, los valores de EMPLOYEE_ID son evaluados para la fila padre y los valores de MANAGER_ID y SALARY son evaluados para las filas hijas. El operador PRIOR aplica solo para el valor EMPLOYEE_ID.

```
... CONNECT BY PRIOR employee_id = manager_id
                AND salary > 15000;
```

Para cualificar a una fila como hija, la fila debe tener un valor en MANAGER_ID igual al valor de EMPLOYEE_ID de la fila padre y debe tener un valor en SALARY mayor que \$15,000.

Walking the Tree: From the Top Down

```

SELECT last_name || ' reports to ' ||
PRIOR last_name "Walk Top Down"
FROM employees
START WITH last_name = 'King'
CONNECT BY PRIOR employee_id = manager_id ;
    
```

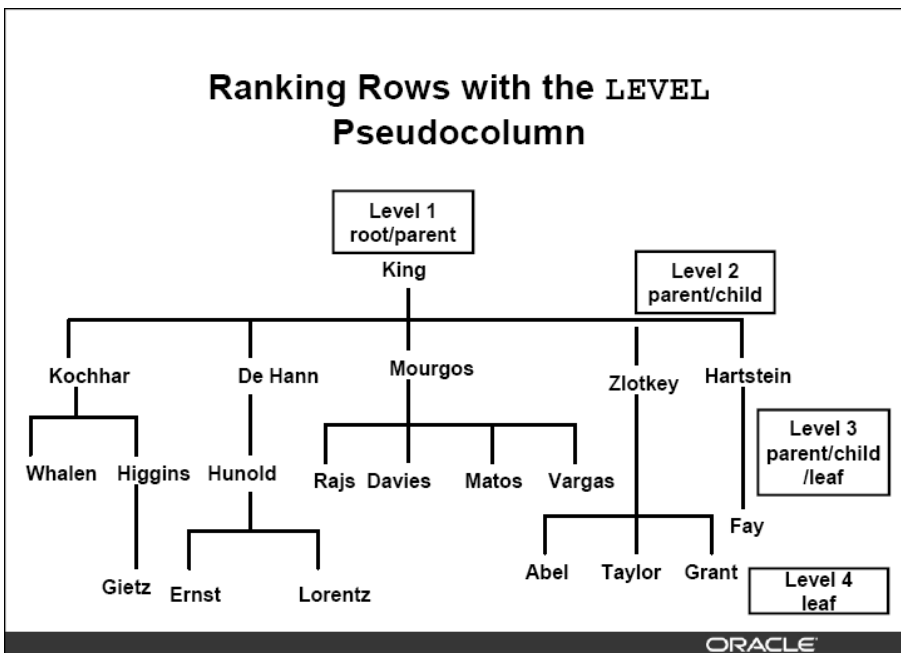
Walk Top Down
King reports to
Kochhar reports to King
Whalen reports to Kochhar
Higgins reports to Kochhar
...
Zlotkey reports to King
Abel reports to Zlotkey
Taylor reports to Zlotkey
Grant reports to Zlotkey
Hartstein reports to King
Fay reports to Hartstein

20 rows selected.

ORACLE

Recorriendo el árbol de arriba – abajo

En el ejemplo anterior se recorre el árbol de arriba-abajo, se despliegan los nombres de los empleados y sus jefes, usando al empleado KING como punto inicial, imprimiendo el resultado en una sola columna.



Jerarquizando filas con la pseudo columna LEVEL

Se puede explícitamente mostrar el nivel de una fila en la jerarquía con el uso de la pseudo columna LEVEL. Esto hace que su reporte sea de una mejor lectura. La bifurcación donde una o más ramas se dividen son llamados nodos, y el final de la rama es llamada hoja o nodo hoja. El diagrama anterior muestra los nodos de un árbol invertido con los valores de su LEVEL. Por ejemplo, el empleado Higgins es un padre y un hijo, mientras que el empleado Davies es un hijo y una hoja

La pseudo columna LEVEL

Valor	LEVEL (Nivel)
1	Un nodo raíz
2	Un hijo del nodo raíz
3	Un hijo de un hijo, etc.

Nota: un *nodo raíz* es el nodo más alto en un árbol invertido. Un *nodo hijo* es cualquier nodo que no sea la raíz. Un *nodo padre* es cualquier nodo que tenga un hijo. Un *nodo hoja* es cualquier nodo sin hijos. El número de niveles obtenidos por una consulta jerárquica puede ser limitado por la memoria disponible del usuario.

En el ejemplo, King es la raíz o padre (LEVEL = 1). Kochhar, De Hann, Mourgos, Zlotkey, Hartstein, y Hunold son hijos y padres (LEVEL = 2). Whalen, Rajs, Davies, Matos, Vargas, Gietz, Ernst, Lorente, Abel, Taylor, Grant y Fay son hijos y hojas. (LEVEL = 3 y LEVEL = 4)

Formatting Hierarchical Reports Using LEVEL and LPAD

Create a report displaying company management levels, beginning with the highest level and indenting each of the following levels.

```

COLUMN org_chart FORMAT A12
SELECT LPAD(last_name, LENGTH(last_name) + (LEVEL*2) - 2, '_')
       AS org_chart
FROM   employees
START WITH last_name='King'
CONNECT BY PRIOR employee_id=manager_id

```

ORACLE

Formateando reportes jerárquicos usando LEVEL

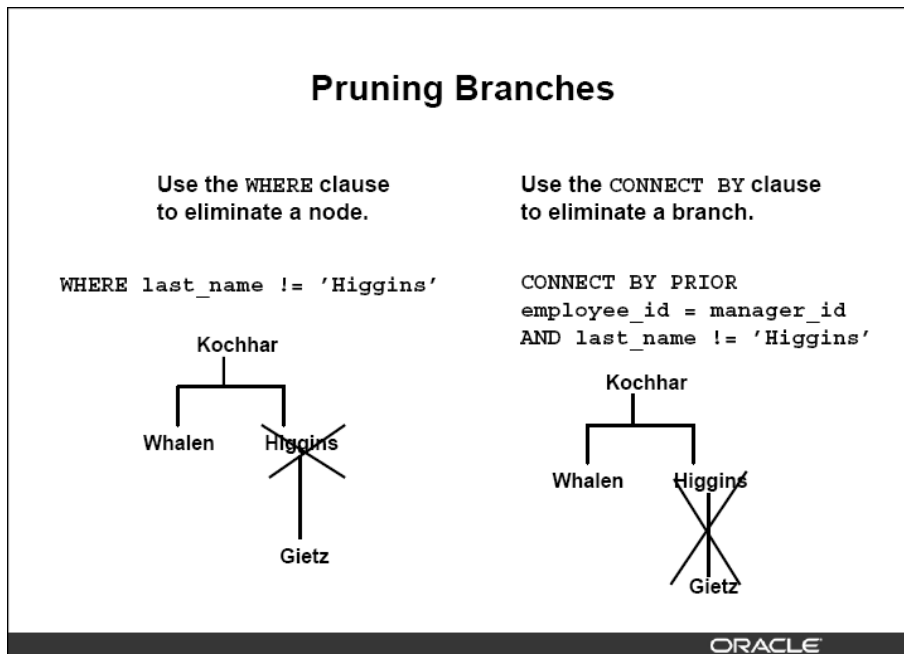
A los nodos en un árbol les es asignado un número de nivel desde la raíz. Usando la función LPAD en conjunto con la pseudo columna LEVEL para desplegar un reporte jerárquico como un árbol indentado.

En el ejemplo:

- LPAD(char1, n [, char2]) obtiene char1, rellenado a la izquierda con una longitud de n con la secuencia de caracteres en char2. El argumento n es la longitud total del valor obtenido ha ser desplegado en la pantalla.
- LPAD(last_name, LENGTH(last_name) + (LEVEL * 2) - 2, '_') define el formato a desplegar
- char1 es el LAST_NAME, n es la longitud total del valor obtenido, es la longitud de LAST_NAME + (LEVEL * 2) - 2, y char2 es '_'.

En otras palabras, esto dice a SQL que tome el LAST_NAME y lo rellene a la izquierda con caracteres '_' hasta que la longitud de la cadena resultante sea igual al valor determinado por LAST_NAME + (LEVEL * 2) - 2.

ORG_CHART
King
_Kochhar
__Whalen
___Higgins
____Gietz
_De Haan
__Hunold
___Ernst
____Lorent z
_Mourgos
__Rajs
___Davies
____Matos
____Vargas
ORG_CHART
_Zlotkey
__Abel
___Taylor
____Grant
____Hartstein
____Fay



Cortando ramas

Se pueden usar las cláusulas WHERE y CONNECT BY para podar el árbol; esto es, controlar que nodos o filas serán desplegadas. El predicado que se usa para hacerlo es una condición Booleana.

Ejemplos

Iniciando con la raíz, recorramos de arriba hacia abajo y eliminemos al empleado Higgins del resultado, pero procesemos a sus hijos.

```
SELECT department_id, employee_id, last_name, job_id, salary
FROM employees
WHERE last_name != 'Higgins'
START WITH manager_id IS NULL
CONNECT BY PRIOR employee_id = manager_id;
```

Iniciando con la raíz, recorramos de arriba hacia abajo y eliminemos al empleado Higgins y a todos sus hijos.

```
SELECT department_id, employee_id, last_name, job_id, salary
FROM employees
START WITH manager_id IS NULL
CONNECT BY PRIOR employee_id = manager_id
AND last_name != 'Higgins';
```

Resumen

En este capítulo se han revisado los siguientes temas:

- Como usar consultas jerárquicas para ver una relación jerárquica entre filas en una tabla.
- Especificar la dirección y punto de inicio de una consulta
- Eliminar nodos o ramas