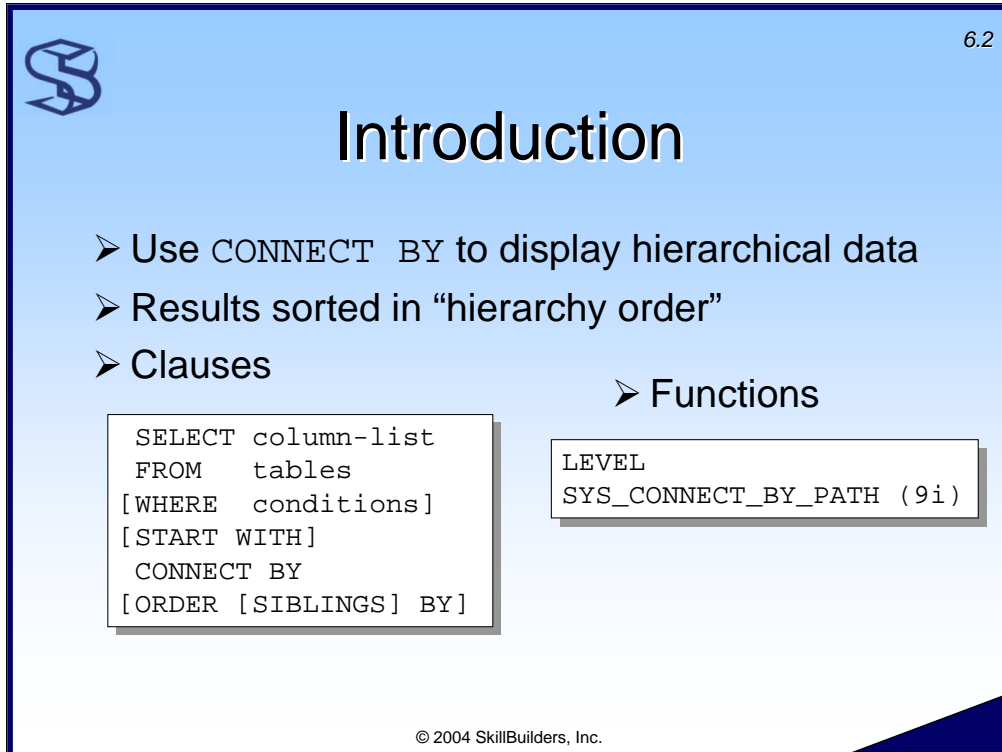


6. CONNECT BY

Creating Hierarchical Queries
LEVEL Function
SIBLINGS Sorts
SYS_CONNECT_AS_PATH Function
Joins

SKILLBUILDERS



6.2

Introduction

- Use `CONNECT BY` to display hierarchical data
- Results sorted in “hierarchy order”
- Clauses
 - SELECT column-list
 - FROM tables
 - [WHERE conditions]
 - [START WITH]
 - CONNECT BY
 - [ORDER [SIBLINGS] BY]
- Functions
 - LEVEL
 - SYS_CONNECT_BY_PATH (9i)

© 2004 SkillBuilders, Inc.

Often we store hierarchies inside a relational table. For example:

- Management organization. An employee table may include a manager column to indicate who the employee reports to.
- Parts (explosion) lists. Some parts may come together become another part. We can keep track of the parent part and all the children by adding a “parent_part” column to the parts table.

The `CONNECT BY` clause allows us to extract the data *sorted by the hierarchy*. There are several clauses related to hierarchical queries:

- `START WITH` – Optional. Code a condition that defines the root row(s) of the hierarchy. E.g. “`START WITH emp_no = 2`”
- `CONNECT BY` – Code a condition that defines the relationship between the parent and child rows. The `CONNECT BY` clause is used to determine which child rows belong to a given parent row. The condition must use the `PRIOR` keyword to identify the parent row. The following two conditions are equivalent:

```
connect by prior emp_no = mgr
connect by mgr = prior emp_no
```


Notes continue on the next page...

- `ORDER SIBLINGS BY` – This clause provides the ability to sort like-level children by one or more keys.

Two related functions are also available:

- `LEVEL` – Returns an integer, starting a 1, that reveals the level of the hierarchy the row is at.
- `SYS_CONNECT_BY_PATH` – Returns the hierarchy path. E.g. For a parts list, it might return something like this: “/2342/535/34/4” indicating that part 2342 is at level 1; part 535 is at level 2 and is a child to parent part 2342, part 34 is at level 3 and is part of 535, etc.

The remainder of this module will provide detailed examples of `CONNECT BY` and the related clauses and functions.



6.4

CONNECT BY Example

➤ Show management hierarchy

```
select emp_no, lastname, mgr, level
from employee
start with emp_no = 2
connect by prior emp_no = mgr;
```

EMP_NO	LASTNAME	MGR	LEVEL
2	Anderson		1
4	Washington	2	2
5	Doright	2	2
6	Wells	5	3
7	Perry	5	3
10	Barbee	5	3
8	Roger	2	2
9	Hall	2	2

© 2004 SkillBuilders, Inc.

CONNECT BY queries display data sorted by the hierarchy specified in the CONNECT BY columns. The keyword “PRIOR” identifies the column from the parent. The column “MGR” is from the child row.

The optional START WITH clause defines where to start the hierarchy display. I.e. START defines the root of the hierarchy.


Given that the EMPLOYEE table contains a column called MGR that contains the employee number of the employees manager (use the SQL*Plus DESCRIBE command to refresh your memory of the EMPLOYEE table), in this example we see employees listed in management-chain order:

- Employee Anderson is at the top of the chain.
- Employee’s Washington and Doright report to Anderson.
- Employee’s Wells, Perry and Barbee report to Doright, so they follow Doright in the report.

Additional Notes

CONNECT BY cannot contain a subquery.

The CONNECT BY clause can precede the START WITH clause.

6.5

LEVEL with LPAD...

➤ LPAD function to indent lower-level siblings

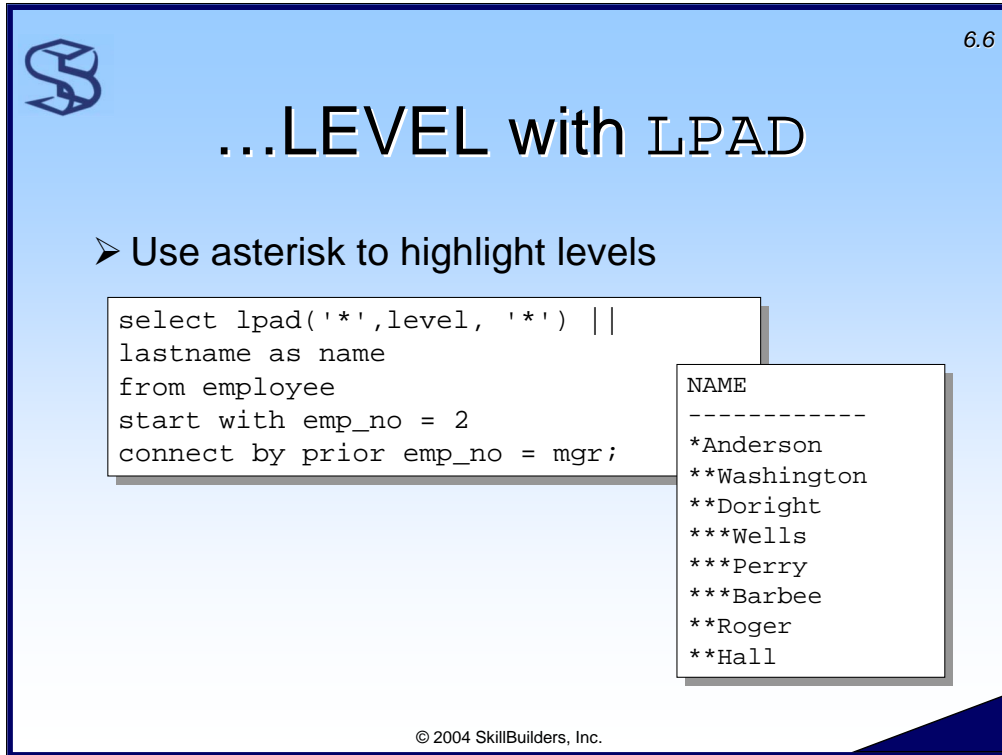
```
select lpad(' ',2*(level-1)) ||  
       lastname as name  
from employee  
start with emp_no = 2  
connect by prior emp_no = mgr;
```

NAME
Anderson
Washington
Doright
Wells
Perry
Barbee
Roger
Hall

© 2004 SkillBuilders, Inc.

The LEVEL function resolves into an integer, starting at 1, that represents the level of the hierarchy.

Using LEVEL within the LPAD function can create a report with indentations that visually help see the hierarchy.



6.6

...LEVEL with LPAD

➤ Use asterisk to highlight levels

```
select lpad('*',level, '*') ||
lastname as name
from employee
start with emp_no = 2
connect by prior emp_no = mgr;
```

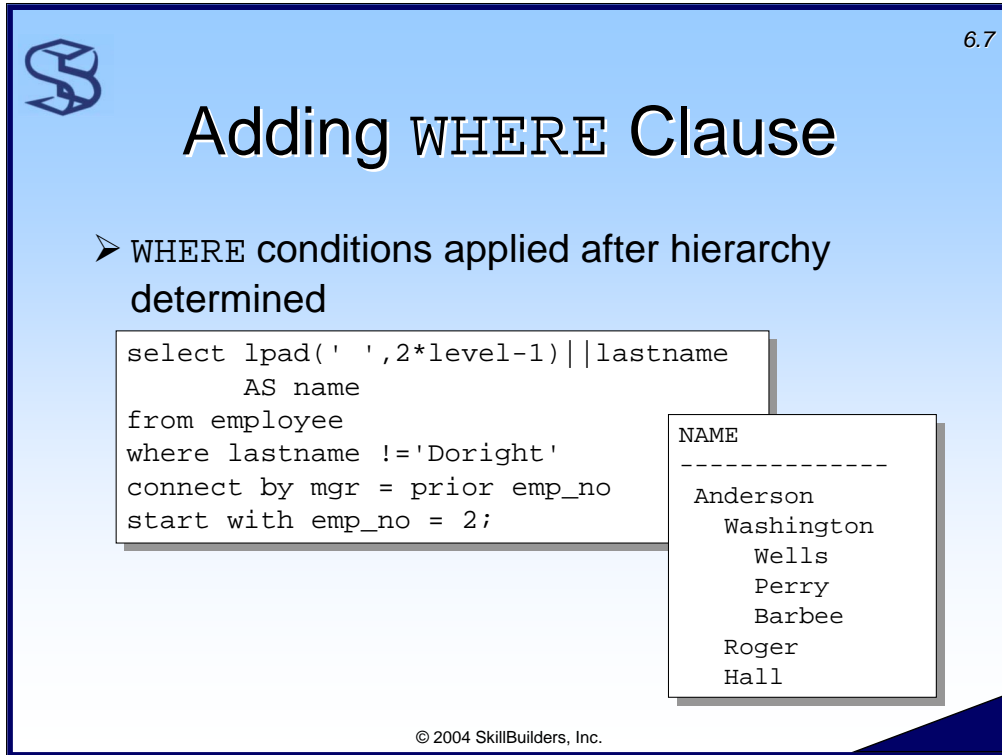
NAME

*Anderson
**Washington
**Doright
***Wells
***Perry
***Barbee
**Roger
**Hall

© 2004 SkillBuilders, Inc.

Using an asterisk or some other character in the `LPAD` function can:

- Increase the visual readability of the report
- Be used programmatically to determine the level a row is at, e.g. one asterisk indicates the row is at level 1 in the hierarchy.



6.7

Adding WHERE Clause

➤ WHERE conditions applied after hierarchy determined

```
select lpad(' ',2*level-1)||lastname
       AS name
from employee
where lastname != 'Doright'
connect by mgr = prior emp_no
start with emp_no = 2;
```

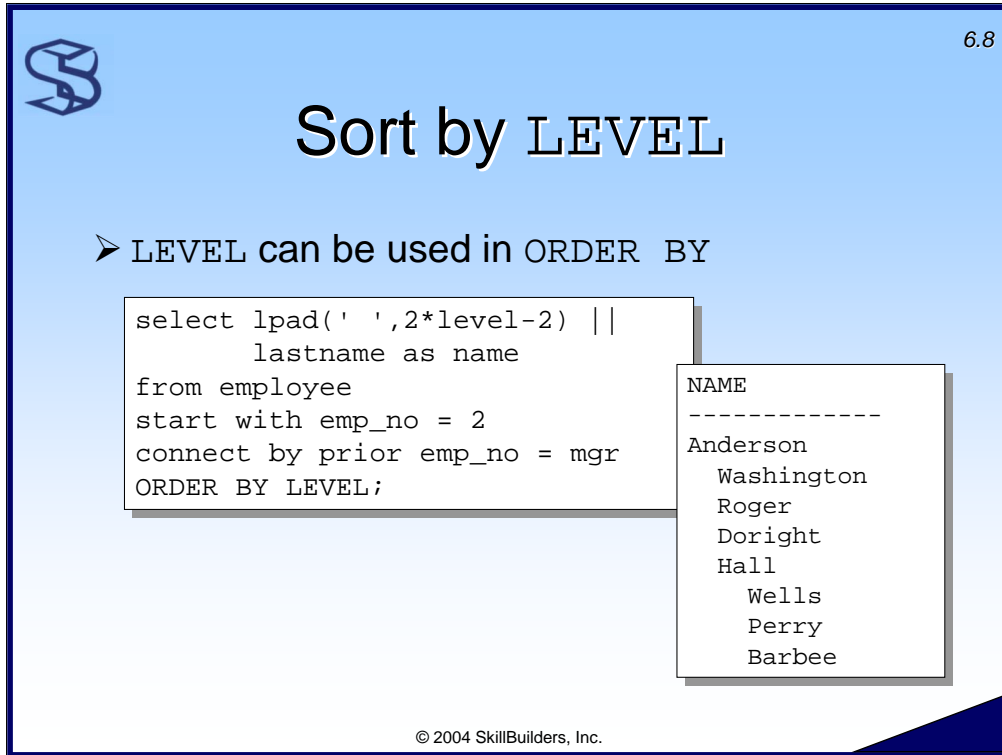
NAME
Anderson
Washington
Wells
Perry
Barbee
Roger
Hall

© 2004 SkillBuilders, Inc.

It is important to understand that WHERE clause conditions are applied after the hierarchy is constructed. (Exception: In Oracle9i, CONNECT BY supports join operations. Join conditions in the WHERE clause are processed before the CONNECT BY.)

Compare this query and result to the example on the previous page. It is the same query except that in this case I added “where lastname != ‘Doright’”. But the result shows that Oracle constructed the hierarchy first, then applied the WHERE clause and removed employee ‘Doright’. If it were processed the other way around, Oracle would not know to put employee ‘Wells’ at level 3, below Washington, since ‘Wells’ points to ‘Doright’ and ‘Doright’ is missing. In fact, it would not know at what level to put ‘Wells’ at all.

Caution: One might infer from looking at this report that employee ‘Wells’ reports to ‘Washington’. This is not case; Wells reports to ‘Doright’



The slide features a blue background with a white box containing a SQL query and its output. The query is: `select lpad(' ',2*level-2) ||
 lastname as name
from employee
start with emp_no = 2
connect by prior emp_no = mgr
ORDER BY LEVEL;` The output is a list of names: Anderson, Washington, Roger, Doright, Hall, Wells, Perry, Barbee. The slide also includes a logo in the top left and the number 6.8 in the top right.

6.8

Sort by LEVEL

➤ LEVEL can be used in ORDER BY

```
select lpad(' ',2*level-2) ||  
  lastname as name  
from employee  
start with emp_no = 2  
connect by prior emp_no = mgr  
ORDER BY LEVEL;
```


NAME

Anderson
Washington
Roger
Doright
Hall
Wells
Perry
Barbee

© 2004 SkillBuilders, Inc.

Using `ORDER BY` changes the “natural” sort order of a `CONNECT BY` query, which is to display the rows in hierarchical order. This is not a common operation because, as the Oracle manual states, “In a hierarchical query, do not specify either `ORDER BY` or `GROUP BY`, as they will destroy the hierarchical order of the `CONNECT BY` results. If you want to order rows of siblings of the same parent, then use the `ORDER SIBLINGS BY` clause.” (In this module, we’ll look at the `SIBLINGS` feature next.)

However, sorting a `CONNECT BY` query with `ORDER BY` can be done. This example uses `LEVEL` to group the group the managers at the same managerial level together in the report. However, one might infer from looking at this report that employee ‘Wells’ reports to ‘Hall’. This is not case; ‘Wells’ reports to ‘Doright’.

6.9

9i SIBLINGS Sorts

➤ Sort same level siblings by some key


```
select lpad(' ',2*level-1) ||  
       lastname as name, hiredate  
from employee  
start with emp_no = 2  
connect by prior emp_no = mgr  
ORDER SIBLINGS BY hiredate;
```

NAME	HIREDATE
Anderson	01-FEB-94
Washington	21-APR-94
Doright	02-AUG-94
Wells	02-AUG-94
Perry	15-MAR-95
Barbee	15-JAN-99
Roger	15-MAR-95
Hall	15-AUG-97

© 2004 SkillBuilders, Inc.

SIBLINGS is a new Oracle9i keyword that sorts all like-level siblings (child) rows by some key.

In this case I chose to sort by column hiredate.

6.10

9i Hierarchy Path...

➤ SYS_CONNECT_BY_PATH function shows path

```
select lpad(' ',2*level-1) ||  
       lastname as name,  
       sys_connect_by_path(lastname, '/') AS path  
from employee  
start with emp_no = 2  
connect by prior  
       emp_no = mgr;
```


NAME	PATH
Anderson	/Anderson
Washington	/Anderson/Washington
Doright	/Anderson/Doright
Wells	/Anderson/Doright/Wells
Perry	/Anderson/Doright/Perry
Barbee	/Anderson/Doright/Barbee
Roger	/Anderson/Roger
Hall	/Anderson/Hall

© 2004 SkillBuilders, Inc.

The new Oracle9i function `SYS_CONNECT_BY_PATH` reveals the hierarchy path for the specified column. It shows the entire path, from root to node.

The first argument is the column for which we want the path. The second argument is the separator.

In this example, we see the management chain starting with employee Anderson.



6.11

...9i Hierarchy Path

```

select lpad(' ',2*level-1) ||
       sys_connect_by_path(lastname, '/') as path
from employee
start with emp_no = 2
connect by prior
       emp_no = mgr;

```

PATH
/Anderson
/Anderson/Washington
/Anderson/Doright
/Anderson/Doright/Wells
/Anderson/Doright/Perry
/Anderson/Doright/Barbee
/Anderson/Roger
/Anderson/Hall

© 2004 SkillBuilders, Inc.

As this example shows, using the `SYS_CONNECT_BY_PATH` function in conjunction with the `LPAD` function creates a useful result.

Note that any column can be used in the `SYS_CONNECT_BY_PATH` function. For example:

```

1 select lpad(' ',2*level-2) || lastname as name,
2        sys_connect_by_path(hiredate, '/') AS path
3 from employee
4 start with emp_no = 2 connect by prior emp_no = mgr;

```

NAME	PATH
-----	-----
Anderson	/01-FEB-94
Washington	/01-FEB-94/21-APR-94
Doright	/01-FEB-94/02-AUG-94
Wells	/01-FEB-94/02-AUG-94/02-AUG-94
Perry	/01-FEB-94/02-AUG-94/15-MAR-95
Barbee	/01-FEB-94/02-AUG-94/15-JAN-99
Roger	/01-FEB-94/15-MAR-95
Hall	/01-FEB-94/15-AUG-97

However, some columns will provide more useful than others!

6.12

9i Supports Joins

➤ CONNECT BY and join now work!

```
select e1.lastname AS employee,  
       e2.lastname AS manager  
from employee e1, employee e2  
where e1.mgr = e2.emp_no(+)  
start with e1.emp_no = 2  
connect by prior  
       e1.emp_no = e1.mgr;
```

EMPLOYEE	MANAGER
Anderson	-----
Hall	Anderson
Roger	Anderson
Doright	Anderson
Barbee	Doright
Perry	Doright
Wells	Doright
Washington	Anderson

© 2004 SkillBuilders, Inc.

In Oracle8i and earlier, `CONNECT BY` and Join could not be used in the same query. Oracle9i now provides support for this.

In this example I incorporated a self-outer-join to the `Employee` table to pick up the lastname of the employees' manager.



6.13

CONNECT BY Workshop

➤ Using CONNECT BY

© 2004 SkillBuilders, Inc.

Run the supplied script `PARTS.SQL` to create the parts table and create the following report:

PARTS

```
-----  
**10 widgets  
****2 wOdgets  
****3 wUdgets  
*****4 weedgets  
*****5 weebles  
*****6 woobles  
*****7 werbless
```