19

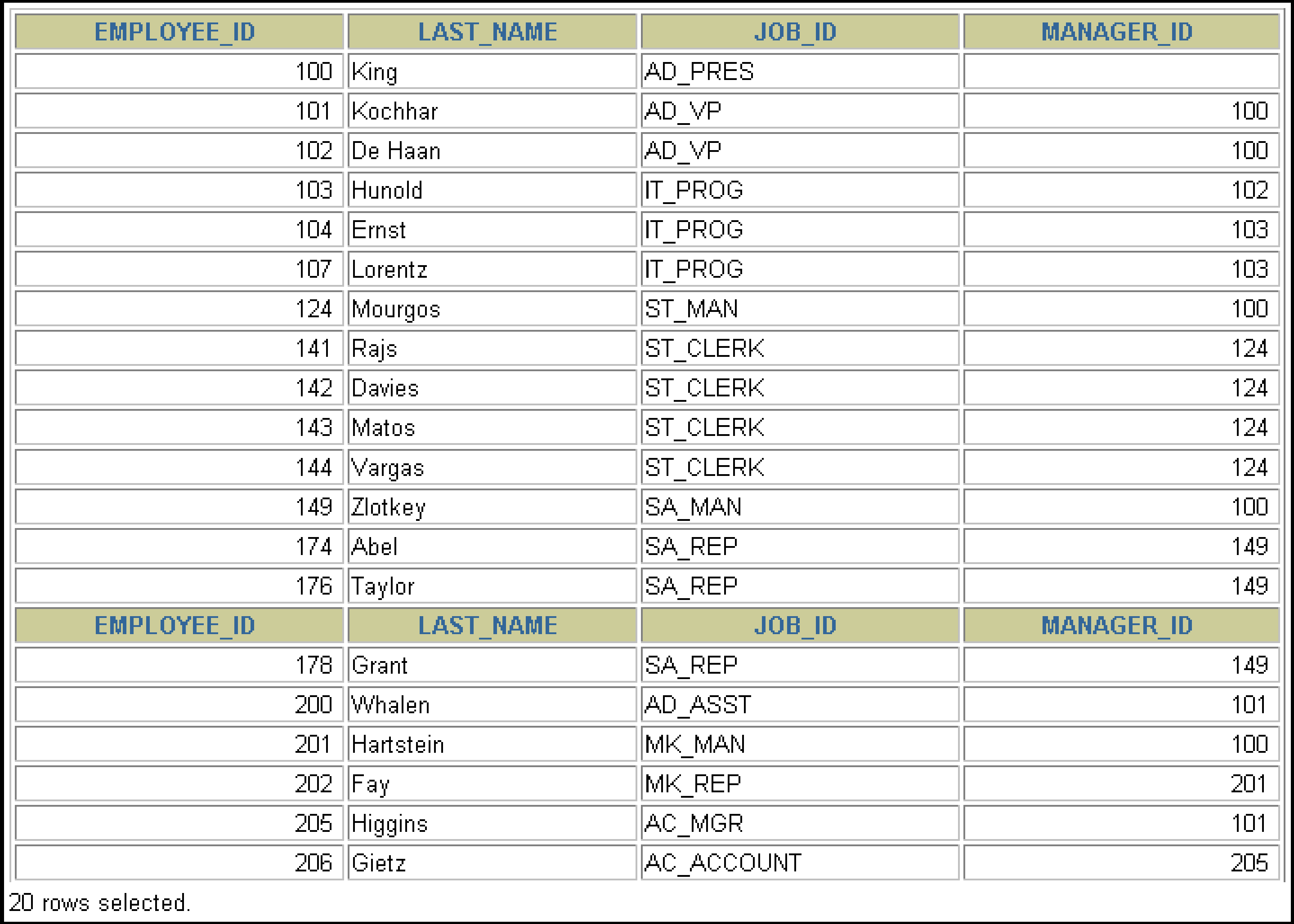
Hierarchical Retrieval

Objectives

After completing this lesson, you should be able to do the following:

- Interpret the concept of a hierarchical query
- Create a tree-structured report
- Format hierarchical data
- Exclude branches from the tree structure

Sample Data from the EMPLOYEES Table



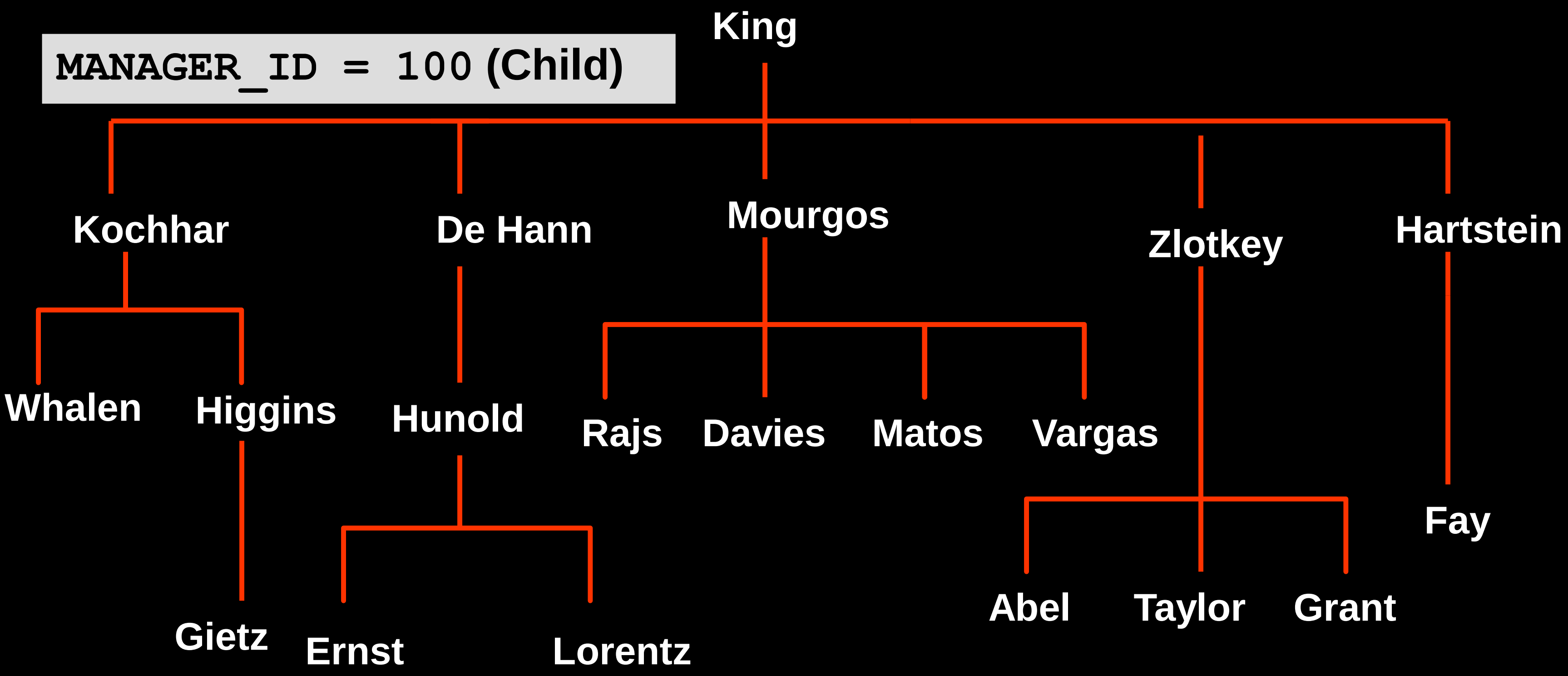
EMPLOYEE_ID	LAST_NAME	JOB_ID	MANAGER_ID
100	King	AD_PRES	
101	Kochhar	AD_VP	100
102	De Haan	AD_VP	100
103	Hunold	IT_PROG	102
104	Ernst	IT_PROG	103
107	Lorentz	IT_PROG	103
124	Mourgos	ST_MAN	100
141	Rajs	ST_CLERK	124
142	Davies	ST_CLERK	124
143	Matos	ST_CLERK	124
144	Vargas	ST_CLERK	124
149	Zlotkey	SA_MAN	100
174	Abel	SA_REP	149
176	Taylor	SA_REP	149
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	Gietz	AC_ACCOUNT	205

20 rows selected.

Natural Tree Structure

`EMPLOYEE_ID = 100` (Parent)

`MANAGER_ID = 100` (Child)



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
```

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

Walking the Tree

```
CONNECT BY PRIOR column1 = column2
```

Walk from the top down, using the `EMPLOYEES` table.

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

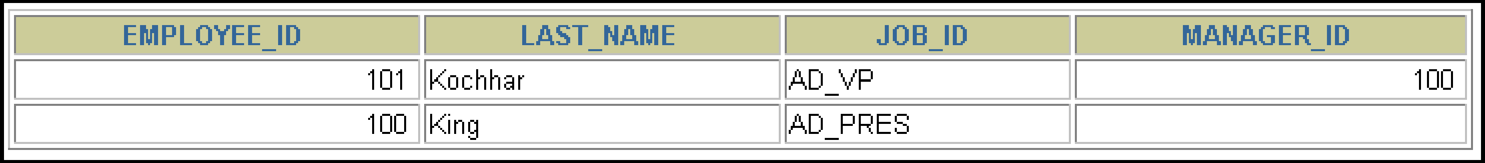
Direction

Top down	$\longrightarrow$	Column1 = Parent Key
		Column2 = Child Key

Bottom up	$\longrightarrow$	Column1 = Child Key
		Column2 = Parent Key

Walking the Tree: From the Bottom Up

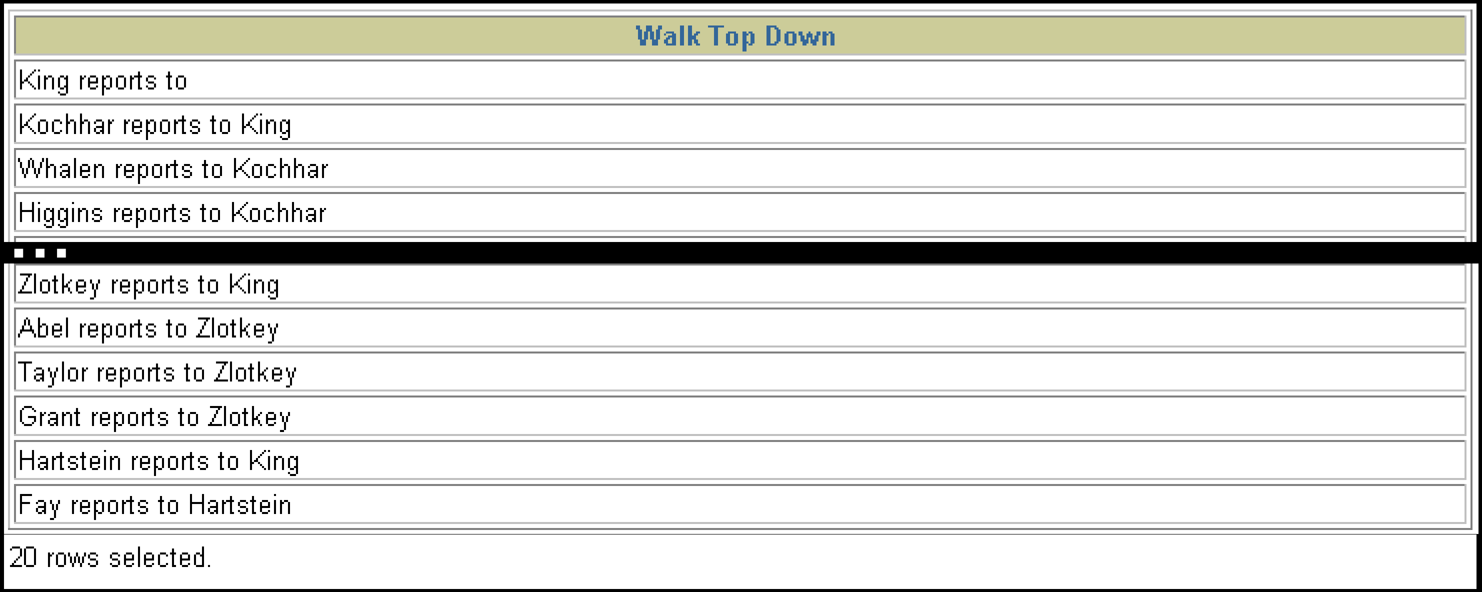
```
SELECT employee_id, last_name, job_id, manager_id
FROM   employees
START  WITH  employee_id = 101
CONNECT BY PRIOR manager_id = employee_id ;
```



EMPLOYEE_ID	LAST_NAME	JOB_ID	MANAGER_ID
101	Kochhar	AD_VP	100
100	King	AD_PRES	

Walking the Tree: From the Top Down

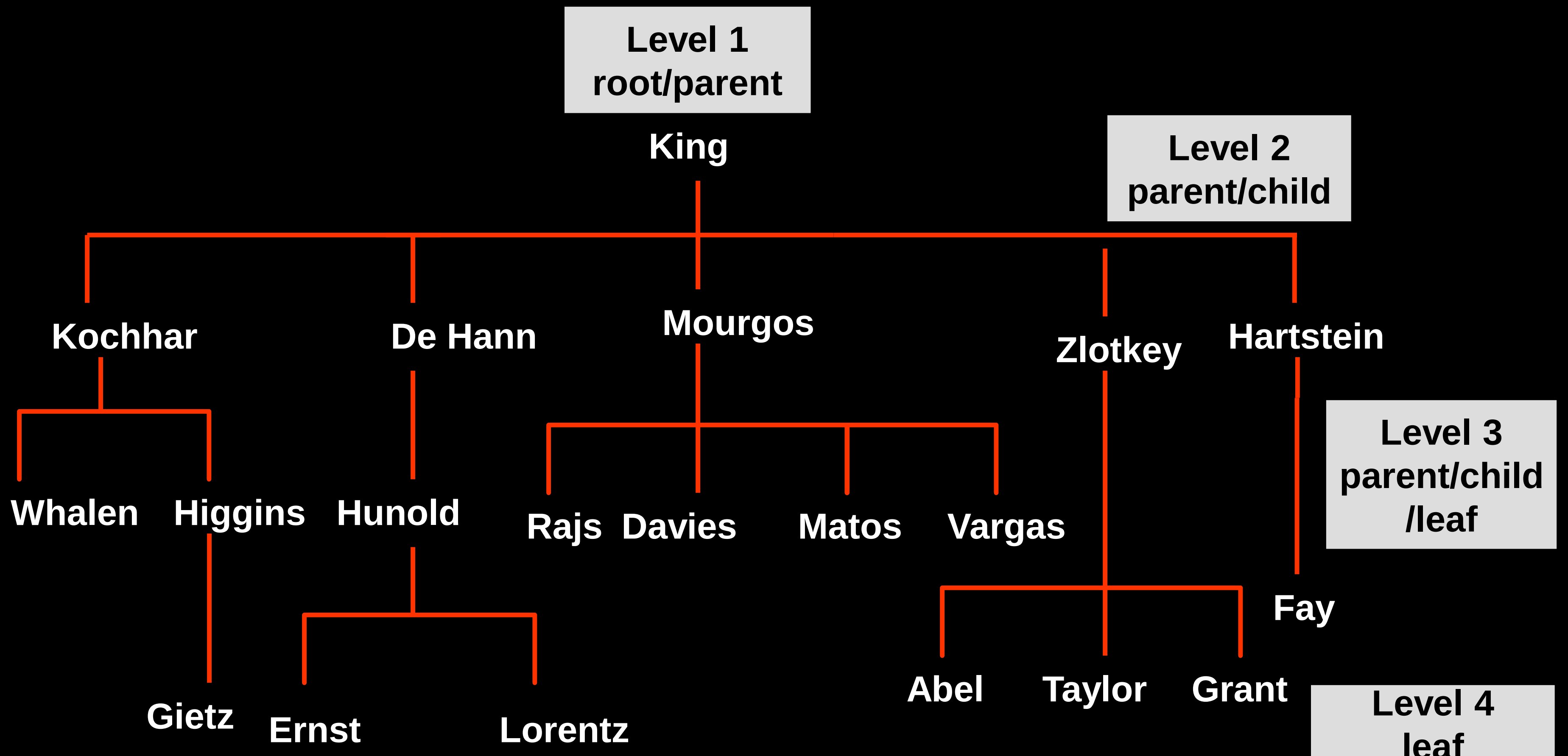
```
SELECT   last_name||' reports to '||
PRIOR    last_name "Walk Top Down"
FROM     employees
START    WITH last_name = 'King'
CONNECT BY PRIOR employee_id = manager_id ;
```



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.

Ranking Rows with the `LEVEL` Pseudocolumn



Formatting Hierarchical Reports Using LEVEL and LPAD

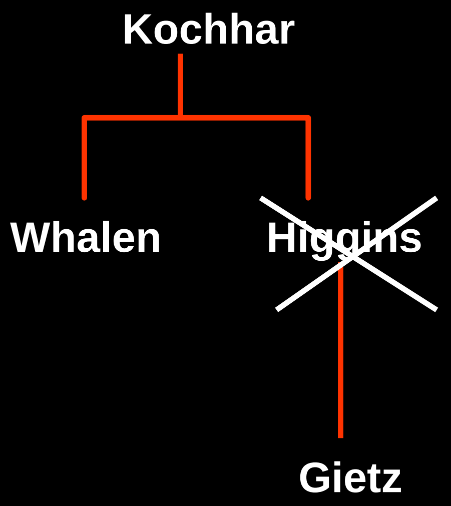
Create a report displaying company management levels, beginning with the highest level and indenting each of the following levels.

```
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
```

Pruning Branches

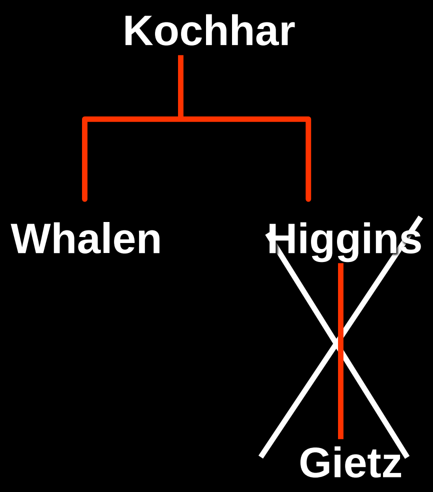
Use the `WHERE` clause
to eliminate a node.

```
WHERE last_name != 'Higgins'
```



Use the `CONNECT BY` clause
to eliminate a branch.

```
CONNECT BY PRIOR
employee_id = manager_id
AND last_name != 'Higgins'
```



Summary

In this lesson, you should have learned the following:

- You can use hierarchical queries to view a hierarchical relationship between rows in a table.
- You specify the direction and starting point of the query.
- You can eliminate nodes or branches by pruning.

Practice 19 Overview

This practice covers the following topics:

- Distinguishing hierarchical queries from nonhierarchical queries
- Walking through a tree
- Producing an indented report by using the `LEVEL` pseudocolumn
- Pruning the tree structure
- Sorting the output