7

階層式擷取

目標

上完本章節之後，您應該能夠：

- 詮釋階層式查詢的概念
- 建立樹狀結構報表
- 爲階層式資料建立格式
- 修剪樹狀結構的分支

目標

在本章節中，您會學到如何使用階層式查詢來建立樹狀結構報表。

EMPLOYEES 表格的範例資料

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
105	Austin	IT_PROG	103
106	Pataballa	IT_PROG	103
107	Lorentz	IT_PROG	103
108	Greenberg	FI_MGR	101

...

EMPLOYEE_ID	LAST_NAME	JOB_ID	MANAGER_ID
196	Walsh	SH_CLERK	124
197	Feeney	SH_CLERK	124
198	OConnell	SH_CLERK	124
199	Grant	SH_CLERK	124
200	Whalen	AD_ASST	101
201	Hartstein	MK_MAN	100
202	Fay	MK_REP	201
203	Mavris	HR_REP	101
204	Baer	PR_REP	101
205	Higgins	AC_MGR	101
206	Gietz	AC_ACCOUNT	205

107 rows selected.

ORACLE

EMPLOYEES 表格的範例資料

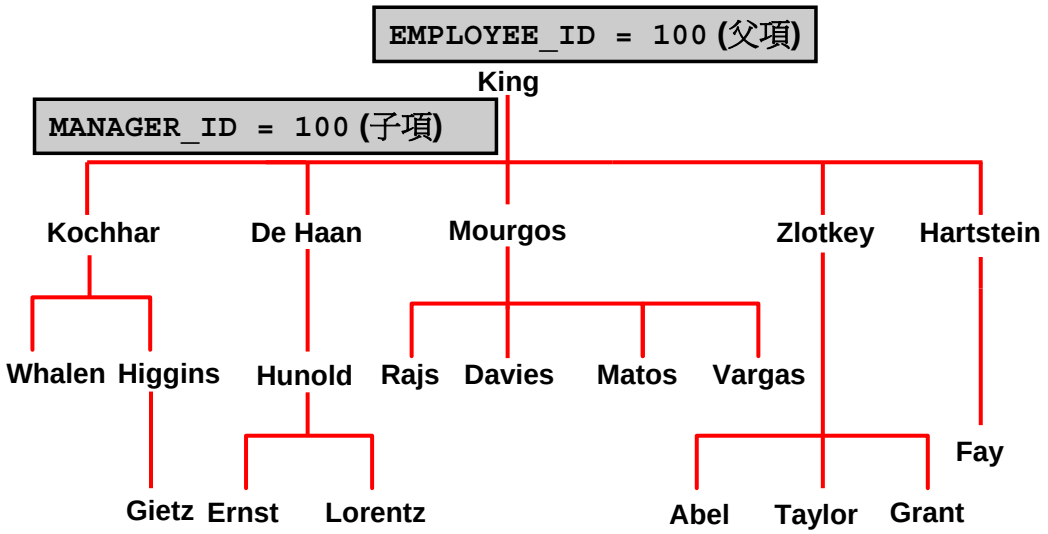
您可以使用階層式查詢，根據表格中各資料列之間的自然階層關係來擷取資料。關聯式資料庫並不會以階層的方式來儲存記錄。然而在單一表格的資料列中存在階層關係時，您可以使用一種稱爲**樹狀結構巡覽 (tree walking)** 的處理程序來建構階層。階層式查詢是一種形成報表的方法，其樹狀結構中的分支會按照一定順序。

請想像一個家族的樹狀結構圖，其中年紀最長的家族成員是位於樹狀結構的根部或樹幹，而最年輕的成員則代表樹狀結構的分支。而分支還可以有自己的分支，依此類推。

當表格中的資料列之間存在關係時，就可以使用階層式查詢。舉例來說，在投影片上，您可以看到工作 ID 爲 AD_VP、ST_MAN、SA_MAN 以及 MK_MAN 的員工都是直屬於公司總裁。我們之所以知道，是因爲這些記錄的 MANAGER_ID 資料欄都有員工 ID 100，而此 ID 爲總裁所有 (AD_PRES)。

注意：階層式樹狀結構可用於各種不同的領域，如人類族譜 (家譜)、家畜 (繁殖用途)、公司管理 (管理階層)、製造 (產品裝配)、演化研究 (物種演進) 以及科學研究。

自然樹狀結構



自然樹狀結構

EMPLOYEES 表格有一個樹狀結構，來代表管理上的從屬關係。透過觀察 EMPLOYEE_ID 與 MANAGER_ID 資料欄中相對值之間的關係，就可以建立階層。也可以透過將表格結合到表格本身來探索此關係。MANAGER_ID 資料欄包含員工所屬經理的員工編號。

樹狀結構中的父-子關係讓您可以控制：

- 巡覽階層的方向
- 階層內的起點

注意：此投影片顯示 EMPLOYEES 表格中，員工管理階層的反向樹狀結構。

階層式查詢

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

此處的*條件*：

```
expr comparison_operator expr
```

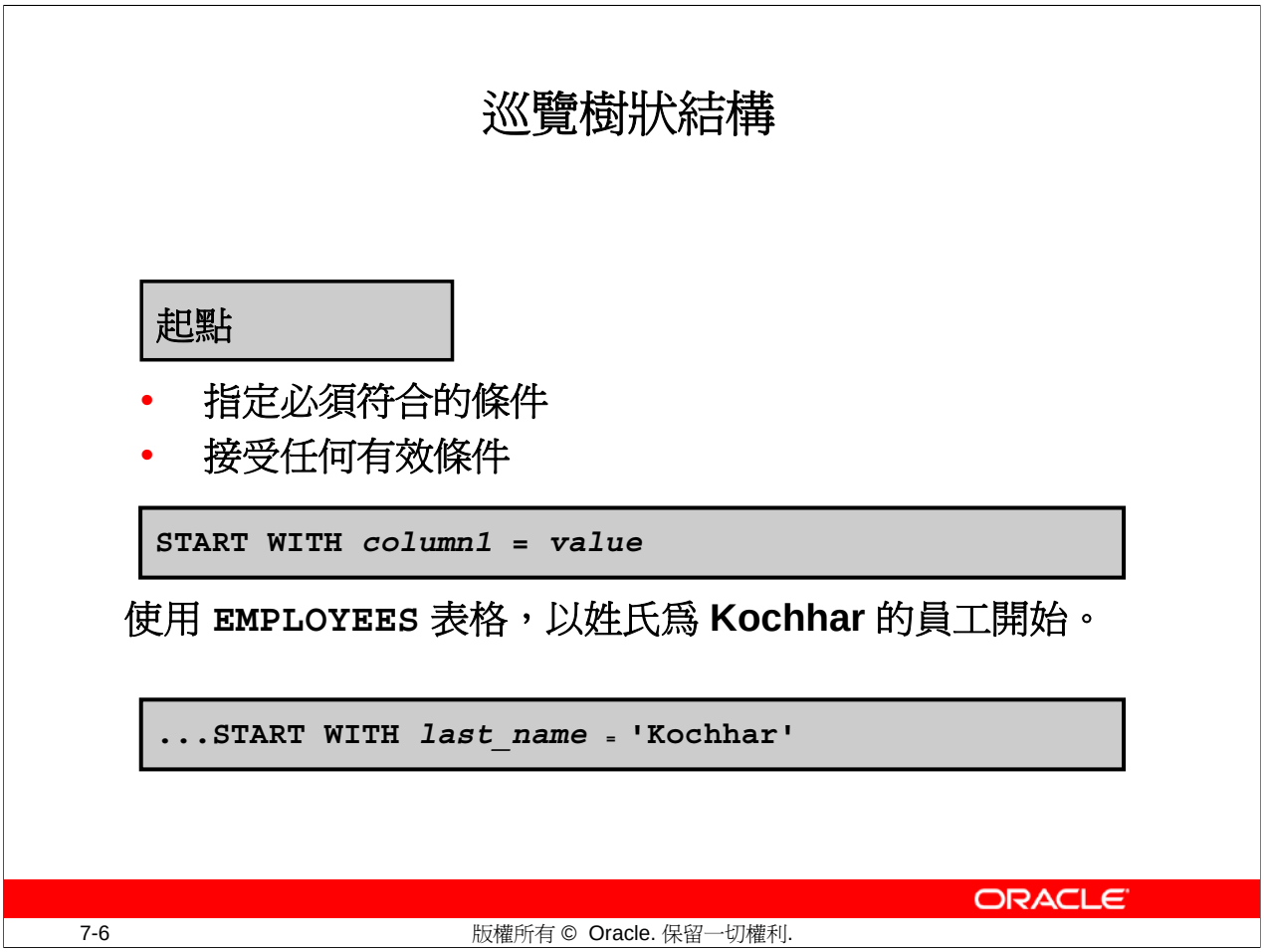

階層式查詢

您可以透過檢查是否存在 CONNECT BY 與 START WITH 子句，來識別階層式查詢。

在此語法中：

SELECT	爲標準的 SELECT 子句
LEVEL	針對一個階層式查詢所傳回的每個資料列，LEVEL 虛擬直欄會傳回根資料列爲 1，根資料列的子資料列爲 2，依此類推。
FROM *table*	指定包含資料欄的表格、視觀表或快照。您只能從一個表格中選擇。
WHERE	限制查詢所傳回的資料列，而不影響階層中的其他資料列
condition	是擁有表示式的比較
START WITH	指定階層的根資料列 (開始的位置)。這是真實階層式查詢必要的子句。
CONNECT BY	指定資料欄，其中在父與子 PRIOR 資料列之間存在關係。這是階層式查詢必要的子句。

SELECT 敘述句不能包含結合或來自有結合的視觀表之查詢。



巡覽樹狀結構

要用來當成樹狀結構根部的資料列是由 START WITH 子句來決定。START WITH 可以與任何有效條件一起結合使用。

範例

使用 EMPLOYEES 表格，從公司總裁 King 開始。

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

使用 EMPLOYEES 表格，從員工 Kochhar. A 開始。START WITH 條件可以包含一個子查詢。

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

如果是略過 START WITH 子句，則樹狀結構的巡覽會將表格中的所有資料列當成根資料列來開始。如果使用 WHERE 子句，則巡覽會從所有滿足 WHERE 條件的資料列開始。這樣一來就無法再反映真實的階層。

注意：CONNECT BY PRIOR 與 START WITH 子句並非 ANSI SQL 標準。

巡覽樹狀結構

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

使用 **EMPLOYEES** 表格，由上往下巡覽。

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

方向

由上往下 ⟶ **資料欄 1 = 父鍵**
資料欄 2 = 子鍵

由下至上 ⟶ **資料欄 1 = 子鍵**
資料欄 2 = 父鍵

巡覽樹狀結構 (續)

查詢的方向，不論是從父到子或子到父，是由 CONNECT BY PRIOR 資料欄位置所決定的。PRIOR 運算子會參照父資料列。Oracle 伺服器爲了要找出父資料列中的子資料列，會評估父資料列的 PRIOR 表示式，以及表格中每個資料列的其他表示式。其條件爲真的資料列就是父資料列的子資料列。Oracle 伺服器總是會考量目前的父資料列，然後透過評估 CONNECT BY 條件來選擇子資料列。

範例

使用 EMPLOYEES 表格，由上至下進行巡覽。定義一個階層關係，其中父資料列的 EMPLOYEE_ID 值與子資料列中的 MANAGER_ID 相同。

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

使用 EMPLOYEES 表格，由下至上進行巡覽。

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

PRIOR 運算子並不需要在 CONNECT BY 後面立即被編碼。因此，下列的 CONNECT BY PRIOR 子句就可以提供與前面範例相同的結果。

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

注意：CONNECT BY 子句不能包含一個子查詢。

巡覽樹狀結構：由下至上

```
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	

巡覽樹狀結構：由下至上

投影片中的範例顯示了一份經理清單，而從員工 ID 爲 101 的員工開始。

範例

在下列範例中，會評估父資料列的 EMPLOYEE_ID 值，還會評估子資料列的 MANAGER_ID 與 SALARY 值。PRIOR 運算子只會套用到 EMPLOYEE_ID 值。

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

如果要符合子資料列的資格，一個資料列必須要有與父資料列 EMPLOYEE_ID 的值相同的 MANAGER_ID 值，而且必須要有比 $15,000 還要高的 SALARY 值。

巡覽樹狀結構：由上至下

```
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
King reports to
Kochhar reports to King
Greenberg reports to Kochhar
Faviet reports to Greenberg
Chen reports to Greenberg

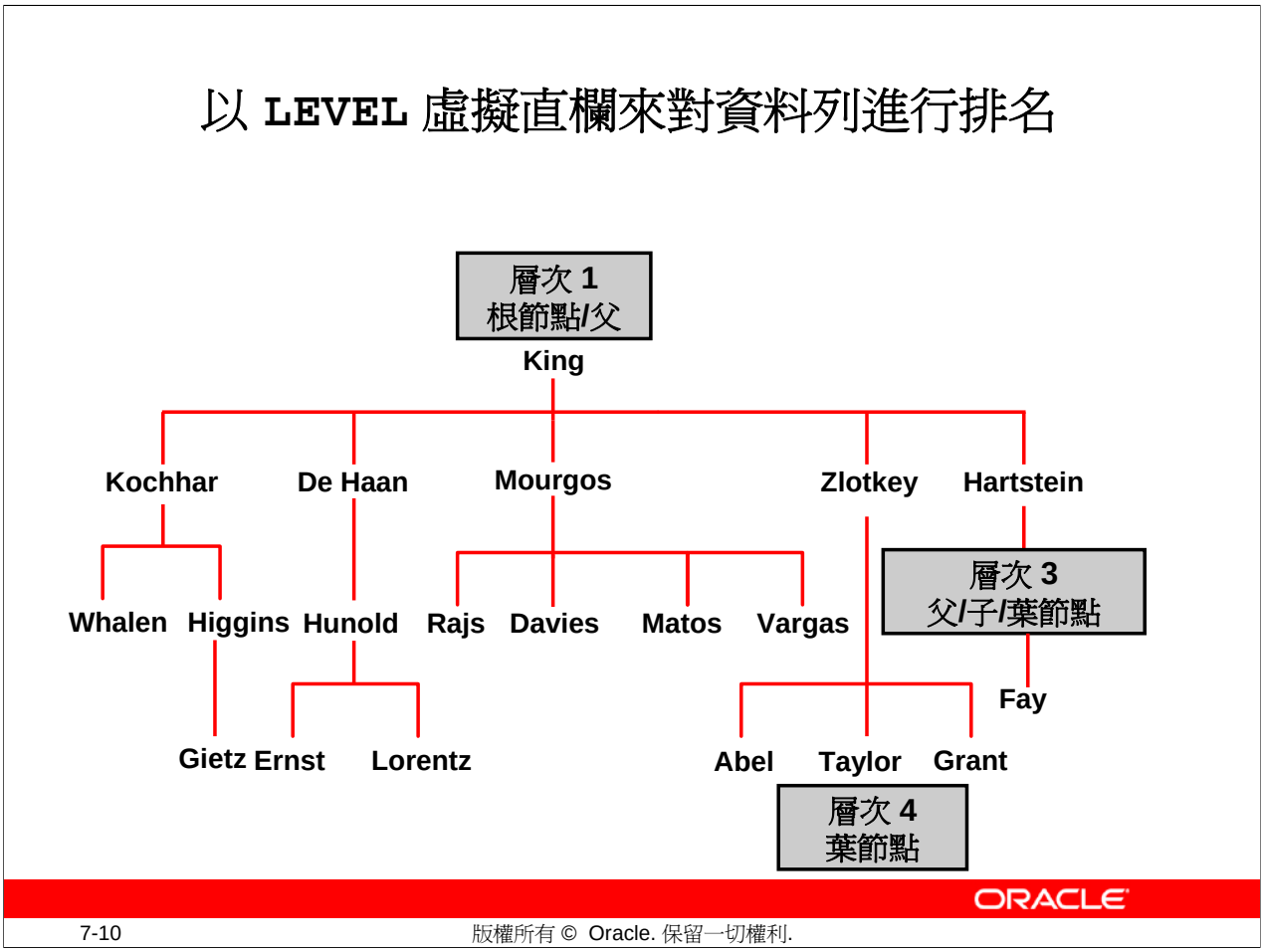
...

108 rows selected.

ORACLE

巡覽樹狀結構：由上至下

由上至下來巡覽，顯示員工的名稱和他們的經理。使用員工 King 爲起點。只列印一個資料欄。



以 LEVEL 虛擬直欄來對資料列進行排名

您可以使用 LEVEL 虛擬直欄，明確地顯示一個資料列在階層樹中的排名或等級，這樣做能讓您的報表更易於閱讀。從一個更大的分支再分出一或多個分支的分叉位置，即稱爲節點 (node)，而分支的盡頭則稱爲葉 (leaf) 或葉節點。投影片上的圖示顯示反向樹狀結構的節點以及他們的 LEVEL 值。舉例來說，員工 Higgens 是一個父節點與一個子節點，而員工 Davies 則爲一個子節點及一個葉節點。

LEVEL 虛擬直欄

值	層次
1	根節點
2	根節點的子節點
3	子節點的子節點，等等

在投影片中，King 爲根節點或父節點 (LEVEL = 1)。Kochhar、De Haan、Mourgos、Zlotkey、Hartstein、Higgens 及 Hunold 爲子節點，而同時也是父節點 (LEVEL = 2)。Whalen、Rajs、 Davies、Matos、Vargas、Gietz、Ernst、Lorentz、Abel、Taylor、Grant 與 Fay 爲子節點與葉節點。(LEVEL = 3 及 LEVEL = 4)

注意：根節點 (root node) 是反向樹狀結構中最高的節點。**子節點 (child node)** 則是任何非根節點的節點。而父節點是任何擁有子節點的節點，葉節點則是沒有子節點的節點。一個階層式查詢所傳回的層次數目，受可用的使用者記憶體限制。

使用 LEVEL 與 LPAD 來建立階層式報表的格式

建立一份報表，顯示公司的管理層次，從最高層開始，然後將下列層次的每一個項目都加以縮排處理。

```
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
```


使用 LEVEL 來建立階層式報表的格式

在樹狀結構中的節點會從根節點開始被指定層次編號。請使用 LPAD 函數結合虛擬直欄 LEVEL 來將一個階層式報表顯示成縮排的樹狀結構。

在投影片中的範例：

- LPAD(*char1*,*n* [,*char2*]) 會傳回 *char1*，並以 *char2* 中的字元順序從左邊依長度 *n* 填滿。引數 *n* 是傳回值的總長度，如同它在您終端機視窗上顯示的狀態。
- LPAD(last_name, LENGTH(last_name)+(LEVEL*2)-2,'_') 會定義顯示格式。
- *char1* 爲 LAST_NAME，*n* 爲傳回值的總長度，也是 LAST_NAME +(LEVEL*2)-2 的長度，而 *char2* 爲 '_'。

換句話說，這會 SQL 使用 LAST_NAME，然後在它左邊填上 '_' 字元，直到結果字串的長度相等於 LENGTH(last_name)+(LEVEL*2)-2 所決定的值爲止。

針對 King，LEVEL = 1。所以是 (2 * 1) – 2 = 2 – 2 = 0。所以 King 並不會填上任何 '_' 字元，而且會顯示在資料欄 1 中。

針對 Kochhar，LEVEL = 2。所以是 (2 * 2) – 2 = 4 – 2 = 2。因此 Kochhar 會填上兩個 '_' 字元，並且會以縮排的方式來顯示。

EMPLOYEES 表格中的其他記錄會以同樣的方式來顯示。

使用 LEVEL 來建立階層式報表的格式 (續)

ORG_CHART
King
King
__Kochhar
____Greenber g
______Faviet
______Chen
______Sciarr a
______Urman
______Popp
____Whalen
____Mavris
____Baer
____Higgins
Gietz
...
____Kumar
__Zlotkey
____Abel
____Hutton
____Taylor
____Livingst on
____Grant
____Johnson
__Hartstein
____Fay

108 rows selected.

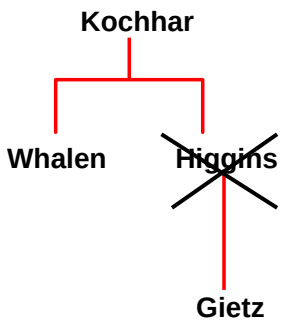
刪除分支

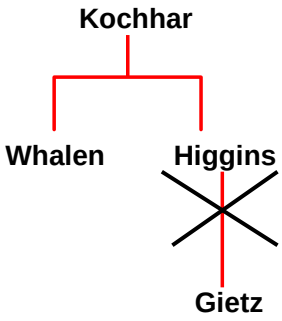
使用 **WHERE** 子句來刪除節點。

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

使用 **CONNECT BY** 子句來刪除分支。

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





ORACLE

刪除分支

您可以使用 WHERE 與 CONNECT BY 子句來修剪樹狀結構 ；也就是說，來控制要顯示哪個節點或資料列。您使用的述詞會用來當作布林值條件。

範例

從根節點開始，由上至下巡覽，然後在結果中刪除員工 Higgins，但要處理子資料列。

```
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;
```

從根節點開始，由上至下巡覽，然後刪除員工 Higgins 以及所有子資料列。

```
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';
```

總結

在本章節中，您應該已經學會如何：

- 使用階層式查詢來檢視表格中資料列之間的階層關係。
- 指定查詢的方向與起點。
- 使用修剪的方式來刪除節點或分支。

總結

您可以使用階層式查詢，根據表格中資料列之間的自然階層關係來擷取資料。LEVEL 虛擬直欄會計算您在階層式樹狀結構中巡覽幾層。您可以使用 CONNECT BY PRIOR 子句來指定查詢的方向，也可以使用 START WITH 子句來指定起點，還可以使用 WHERE 及 CONNECT BY 子句來修剪樹狀結構的分支。

課堂練習 7：簡介

本課堂練習涵蓋下列主題：

- 區分階層式查詢與非階層式查詢
- 在樹狀結構中巡覽
- 使用 **LEVEL** 虛擬直欄來產生一份縮排的報表
- 修剪樹狀結構
- 對輸出結果進行排序

ORACLE

課堂練習 7：簡介

在本課堂練習中，您將獲得產生階層式報表的實際經驗。

注意：問題 1 是用紙筆作答的問題。

課堂練習 7

1. 請觀察下列的輸出範例。這些輸出範例是階層式查詢的結果嗎？請解釋爲什麼是，或者爲什麼不是。

展示 1：

EMPLOYEE_ID	LAST_NAME	MANAGER_ID	SALARY	DEPARTMENT_ID
100	King		24000	90
101	Kochhar	100	17000	90
102	De Haan	100	17000	90
201	Hartstein	100	13000	20
205	Higgins	101	12000	110
174	Abel	149	11000	80
149	Zlotkey	100	10500	80
103	Hunold	102	9000	60
...				
200	Whalen	101	4400	10
107	Lorentz	103	4200	60
141	Rajs	124	3500	50
142	Davies	124	3100	50
143	Matos	124	2600	50
144	Vargas	124	2500	50

20 rows selected.

展示 2：

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
205	Higgins	110	Accounting
206	Gietz	110	Accounting
100	King	90	Executive
101	Kochhar	90	Executive
102	De Haan	90	Executive
149	Zlotkey	80	Sales
174	Abel	80	Sales
176	Taylor	80	Sales
103	Hunold	60	IT
104	Ernst	60	IT
107	Lorentz	60	IT

11 rows selected.

課堂練習 7 (續)

展示 3：

RANK	LAST_NAME
1	King
2	Kochhar
2	De Haan
3	Hunold
4	Ernst

2. 請建立一個報表，來顯示 Mourgos 的部門的組織圖。列印姓氏、薪資及部門 ID。

LAST_NAME	SALARY	DEPARTMENT_ID
Mourgos	5800	50
Rajs	3500	50
Davies	3100	50
Matos	2600	50
Vargas	2500	50
Walsh	3100	50
Feeney	3000	50
OConnell	2600	50
Grant	2600	50

9 rows selected.

3. 建立一個報表，而此報表是顯示員工 Lorentz 的經理的階層。先顯示他的直屬經理。

LAST_NAME
Hunold
De Haan
King

課堂練習 7 (續)

4. 建立一個縮排的報表，顯示從 LAST_NAME 爲 Kochhar 的員工開始的管理階層。請列印員工的姓氏、經理 ID 以及部門 ID。並如範例輸出中所示，提供資料欄別名。

NAME	MGR	DEPTNO
Kochhar	100	90
__Greenberg	101	100
____Faviet	108	100
____Chen	108	100
____Sciarra	108	100
____Urman	108	100
____Popp	108	100
__Whalen	101	10
__Mavris	101	40
__Baer	101	70
__Higgins	101	110
____Gietz	205	110

12 rows selected.

如果還有時間，您可以試做下列練習題：

5. 建立一個顯示管理階層的公司組織圖。從最上層的人員開始，排除所有工作 ID 爲 IT_PROG 的人，再排除 De Haan 及那些直屬於 De Haan 的員工。

LAST_NAME	EMPLOYEE_ID	MANAGER_ID
King	100	
Kochhar	101	100
Greenberg	108	101
Faviet	109	108
Chen	110	108
Sciarra	111	108

...

LAST_NAME	EMPLOYEE_ID	MANAGER_ID
Livingston	177	149
Grant	178	149
Johnson	179	149
Hartstein	201	100
Fay	202	201

101 rows selected.