Objectives

 Kendi kendine birleştirme (self-join) kullanarak bir tabloyu kendisiyle birleştiren SELECT ifadesi oluşturmak ve çalıştırmak

 Hiyerarşik sorgu (hierarchical query) kavramını yorumlamak

 Ağaç yapısında bir rapor oluşturmak

 Hiyerarşik verileri biçimlendirmek

 Ağaç yapısında bazı dalları hariç tutmak

Vocabulary

Her tanım için doğru terimi yazınız:

1. Bir tabloyu kendisiyle birleştirir  
   ➤ **Self Join**
2. Bir tablodaki satırlar arasında doğal bir hiyerarşik ilişkiye dayanarak veri getirir  
   ➤ **Hierarchical Query**
3. Hiyerarşik sorgunun başlangıç satırından itibaren kaç adım aşağıya inileceğini belirler  
   ➤ **LEVEL**
4. Hiyerarşik sorgunun başlangıç satırını tanımlar  
   ➤ **START WITH**
5. Hiyerarşik sorguda ebeveyn ve çocuk satırlar arasındaki ilişkiyi belirtir  
   ➤ **CONNECT BY**

Try It / Solve It

1. Çalışanın soyadı ve çalışan numarasını, ayrıca yöneticisinin soyadı ve yönetici numarasıyla birlikte görüntüle. Sütun adları sırasıyla: **Employee**, **Emp#**, **Manager**, **Mgr#** olsun.

SELECT

e.last\_name AS "Employee",

e.employee\_id AS "Emp#",

m.last\_name AS "Manager",

m.employee\_id AS "Mgr#"

FROM employees e

JOIN

employees m ON e.manager\_id = m.employee\_id;

1. Soru 1'i değiştirerek, **yöneticisi olmayan çalışanları da** dahil et. Listeyi çalışanın soyadına göre alfabetik sıraya göre sırala.

SELECT

e.last\_name AS "Employee",

e.employee\_id AS "Emp#",

m.last\_name AS "Manager",

m.employee\_id AS "Mgr#"

FROM employees e

LEFT JOIN employees m ON e.manager\_id = m.employee\_id

ORDER BY e.last\_name;

1. Yöneticisinden **önce işe alınmış** tüm çalışanların adlarını ve işe alınma tarihlerini, ayrıca yöneticilerinin adlarını ve işe alınma tarihlerini göster. Sütun adları sırasıyla: **Employee**, **Emp Hired**, **Manager**, **Mgr Hired** olsun.

SELECT e.last\_name AS "Employee",e.hire\_date AS "Emp Hired", m.last\_name AS "Manager",m.hire\_date AS "Mgr Hired"

FROM employees e

JOIN employees m ON e.manager\_id = m.employee\_id

WHERE e.hire\_date < m.hire\_date;

1. Lex De Haan’ın çalıştığı departman için bir **hiyerarşi raporu** yaz. Raporda çalışanın **soyadı**, **maaşı** ve **departman numarası** yer alsın.

SELECT

LPAD(' ', LEVEL \* 2, '-') || last\_name AS "Employee",

salary,

department\_id

FROM employees

WHERE

department\_id = (

SELECT department\_id FROM employees WHERE last\_name = 'De Haan'

)

START WITH last\_name = 'De Haan'

CONNECT BY PRIOR employee\_id = manager\_id;

1. Aşağıdaki SQL ifadesinde ne yanlış var?

SELECT last\_name, department\_id, salary

FROM employees

START WITH last\_name = 'King'

CONNECT BY PRIOR manager\_id = employee\_id;

1. Tüm **çalışan tablosu** için bir organizasyon şeması (organization chart) raporu oluştur. Her seviye (level) için çalışan isminin önüne **2 boşluk** eklensin. Ancak Oracle Application Express boşlukları düzgün göstermediği için boşluk yerine **- (tire)** karakterini kullan.

SELECT

LPAD('-', LEVEL \* 2, '-') || last\_name AS "Employee"

FROM employees

START WITH manager\_id IS NULL

CONNECT BY PRIOR employee\_id = manager\_id;

1. sorudaki raporu tekrar yaz, ancak bu kez **De Haan** ve **ona bağlı tüm çalışanlar** hariç tutulsun.

SELECT LPAD('-', LEVEL \* 2, '-') || last\_name AS "Employee"

FROM employees

START WITH manager\_id IS NULL

CONNECT BY PRIOR employee\_id = manager\_id

AND

employee\_id NOT IN (

SELECT employee\_id

FROM employees

START WITH last\_name = 'De Haan'

CONNECT BY PRIOR employee\_id = manager\_id

);