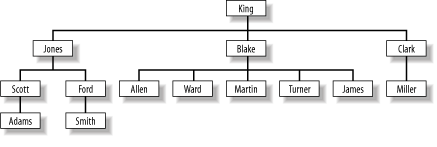
**级联查询(Hierarchical Queries)**

**一、级联数据的表示：**

首先我们来看一张关于组织架构的结构图



这张图是一个典型的“树型结构图”，只有一个根节点(King)，其下有若干个分支节点，每个分支节点下又有若干个子节点或树叶节点。假如我们要把这些关系信息映射到数据库中，表结构应当如何表示呢？

C:\E9A2D725\C64BFF96-48C2-48CD-8043-90F558F38D48.files\image002.gif

CREATE TABLE EMPLOYEE (

EMP\_ID          **NUMBER** (**4**) **CONSTRAINT EMP\_PK PRIMARY KEY,**

FNAME           **VARCHAR2** (**15**)NOT NULL,

LNAME           **VARCHAR2** (**15**)NOT NULL,

DEPT\_ID         **NUMBER** (**2**)NOT NULL,

**MANAGER\_EMP\_ID** **NUMBER** (**4**) **CONSTRAINT EMP\_FK REFERENCES EMPLOYEE(EMP\_ID),**

SALARY          **NUMBER** (**7**,**2**)NOT NULL,

HIRE\_DATE     DATE NOT NULL,

JOB\_ID          **NUMBER** (**3**));

请注意这里红色粗体部分，字段MANAGER\_EMP\_ID的值引用了字段EMP\_ID的值，我们称这种引用为“自引用”。它规定了经理人员的ID必须是来自表中存在的员工ID。

**二、Oracle 中的start with...connect by:**

[[START WITH condition1]  CONNECT BY condition2]

START WITH condition1

指定级联数据的根记录(一条或多条)，所有满足条件1的记录都将被当成是根纪录，假如我们不给定START WITH子句，所有的纪录都会被当成是根纪录，通常这不是我们想要的结果。condition1可以是一个子查询。

CONNECT BY condition2

指定级联数据中父纪录和子纪录之间的关系，这里的关系被表示成一个表达式，当前纪录的字段会和对应的父纪录的某个字段进行比较。condition2必须跟着一个PRIOR操作符，该操作符用于标明父纪录的字段。condtion2不能包含子查询

PRIOR是Oracle的一个内建操作符，仅用于级联查询。当我们在级联查询的CONNECT BY条件中使用了PRIOR操作符时，位于其后的表达式被当成是当前纪录的父纪录进行比较。

**三、实例比较：**

下面我们通过2条SQL语句来演示如何进行级联查询，以及PRIOR在不同位置时带来的不同结果。

C:\E9A2D725\C64BFF96-48C2-48CD-8043-90F558F38D48.files\image002.gif

SQL> select \* from employee;

        ID EMP\_NAME             MANAGER\_ID

---------- -------------------- ----------

**1** king

**2** mark                         **1**

**3** bob                           **1**

**4** tom                           **2**

**5** paul                          **2**

**6** jack                          **3**

**7** ben                           **4**

C:\E9A2D725\C64BFF96-48C2-48CD-8043-90F558F38D48.files\image002.gif

**7** rows selected.

需求：我们要找出员工ID为2的人及其所有下属(包括直接和间接下属)

C:\E9A2D725\C64BFF96-48C2-48CD-8043-90F558F38D48.files\image002.gif

SQL> select \* from employee start with id = **2** connect by prior id = manager\_id order by id;

        ID EMP\_NAME             MANAGER\_ID

---------- -------------------- ----------

**2** mark                          **1**

**4** tom                           **2**

**5** paul                          **2**

**7** ben                           **4**

请注意PRIOR操作符被放置在字段ID前面。查询结果中ID为7的员工ben，虽然其对应的经理ID为4，但是因为员工号为4的tom，其对应的经理ID为2，所以ben是属于mark的间接下属而符合查询条件。

我们已经知道PRIOR放在那一侧，那一侧的字段就会被当成父记录的字段而被用于和当前记录的字段(另一侧的表达式)进行比较，那么假如我们把PRIOR放在manager\_id一侧，结果会有什么不同吗？请看下面的SQL执行结果。

C:\E9A2D725\C64BFF96-48C2-48CD-8043-90F558F38D48.files\image002.gif

SQL> select \* from employee start with id = **2** connect by id = prior manager\_id order by id;

        ID EMP\_NAME             MANAGER\_ID

---------- -------------------- ----------

**1** king

**2** mark                          **1**

很明显结果完全不同，那么是什么造成了两次查询的结果完全不同呢？说到这里我们还要再回到SQL语言本身，我用一种比较直白的方式来讲解不同位置的PRIROR所带来的不同意义。

【1】第一个查询：connect by prior id = manager\_id，意思是从当前根记录开始，查找所有符合条件的记录：他们的manager\_id必须等于当前记录的id。也就是说查找所有manager\_id=2的记录及其子记录，很明显manager\_id=2的记录只有tom和paul，但是由于ben的直接领导tom是mark的下属，所以ben也是mark的下属，只不过是间接关系而已。

【2】第二个查询：connect by id = prior manager\_id，意思是从当前根记录开始，查找所有符合条件的记录：他们的id必须等于当前记录的manager\_id。也就是说查找所以id=1的记录，那么很明显id=1的记录只有king。

**总结：Prior放在那里，那一侧就是被比较的一方(父方)，另一侧就是发起比较的一方(子方)。语义上可以这样翻译：xxx字段的值必须等于当前记录XXX字段的值(prior一方)**

select level,a.\* from emp a start with mgr is null connect by prior empno = mgr;

--empno--子孙

--mgr --祖宗

--prior--在左子孙这边，查找子孙，，在右祖宗这边，查找祖宗

--connect by 递归