用 SQL 形成父子结点树结构的妙法

A Clever Idea for the Use of SQL Forming the Father – child node Tree's Construction

夏明伟　刘星沙　（中南大学 现代教育技术中心 湖南长沙 410075）

摘要：在实际应用中经常会有些反映树型关系的数据，它们以父子结点的形式存储在库表中，为了能够直观地展示它们的带线树结构关系往往需要通过编程来实现。本文通过对 Oracle 的 SQL 语句的妙用，提出用一句 Oracle 的 SQL 语句实现带线树结构展示的妙想，此法在树型字典数据的选择中得到实际应用，简便实用大大提高了程序的可靠性，也为树型关系数据展示其带线树结构提供一种新颖的方法。

关键词：SQL　父子结点　树结构　展示

1　问题的提出

在应用程序开发过程中经常会对字典表数据进行各种操作，特别对树型结构组织的字典表进行数据选择时人们总是希望以树型结构的方式一次展示出来供挑选，而不是以其他方式展示。当然把树型结构的字典表以树型结构的方式展示并非难事，通过程序解析树型字典表结构然后形成树型结构展示即可。我们现在提出的问题就是能否不需要特别的程序解析处理，几乎仍然使用原先的列表型字典表的程序处理逻辑实现对树型结构组织的字典表的数据选择并且给用户的感觉与已经进行了程序解析一样。

带着这个问题笔者通过一系列实验最终找到了一种简单可行的方法，在此与大家一起探讨一下。

2　问题的分析与解决

首先假定我们有表 1 所示的树型结构组织的字典表，其中 id 为树结点编号，pid 为此树结点的父结点编号，name 为结点名称，end 备用。而对于列表型字典表可能只有 id 和 name 两个关键字段，那么如何使得针对列表型字典表的处理程序在处理树型结构组织的字典表时给用户的感觉是树型结构呢？

假定我们可以把表 1 所示的树型结构组织的字典表用一句 SQL 查询语句可动态生成表 2 所示的结果，其中 id 和 name 就相当于列表型字典表中的两个关键字段，那么对表 2 进行列表型字典表的下拉选择时是

表 1　树型结构组织的字典表内容

id	pid	name	end
aa	0	资金	
ba	aa	其他应收款	
fa	ba	工程往来	
fb	ba	四平分厂	
fc	fb	拨付资产	
gg	fb	拨付资金	
ha	gg	拨付流动资金	
hb	gg	拨付技改资金	
bc	aa	在途材料	
bf	bc	原料	
bg	bc	材料	
bd	aa	原料	
ea	bd	烟叶	
eb	bd	烟叶费用	
ec	eb	保险费	
ed	eb	其他	
ee	bd	再造烟叶	
ef	bd	芙蓉王选叶差异	
be	aa	主要材料	
eg	be	总厂主要材料	
ab	0	负债	
ca	ab	营业外收入	
eh	ca	处理盘盈固定资产收入	
cb	ab	其他	

不是感觉是对树型字典数据进行选择了。

表 2　树型结构组织字典表的列表形式

id	name
aa	●资金
ba	| ├其他应收款
fa	| | ├工程往来
fb	| | └四平分厂
fc	| | ├拨付资产
gg	| | └拨付资金
ha	| | └拨付流动资金
bc	| ├在途材料
bf	| | ├原料
bg	| | └材料
bd	| ├原料
ea	| | ├烟叶
eb	| | ├烟叶费用
ec	| | | ├保险费
ed	| | | └其他
ee	| | ├再造烟叶
ef	| | └芙蓉王选叶差异
be	| ├主要材料
eg	| | └总厂主要材料
hb	| └拨付技改资金
ab	●负债
ca	├营业外收入
eh	| └处理盘盈固定资产收入
cb	└其他

那么用一句 SQL 查询语句把表 1 转化为表 2 有解吗？答案是肯定的，下面我们就来寻找这个答案。首先我们发现表 2 中的树有个特征即兄弟中最小的弟弟前要显示 └而不是 ├，兄弟之间如果要留出行显示哥哥的子孙还需要以 | 链接，而最小的弟弟不可能还有弟弟因此在他的下面无须加 | 线。

此时我们想到表 1 中提到的 end 备用字段，其实它就是用来标识此结点是否为幺弟（根据排序字段把在兄弟之间排在最后的设为幺弟）的，因此我们在维护树型字典数据时把幺弟的 end 字段置为 1，非幺弟的 end 字段置为非 1。假如我们以前根本没有考虑维护幺弟标志，那么现在只要在库表中增设一个幺弟标志字段，然后每次增删记录后用下面的语句重设幺弟标志值即可：

```
update xmw_tree set end = '0';
update xmw_tree set end = '1' where id in (select max(a.id) from xmw_tree a,xmw_tree b where a.pid =b.id(+) group by a.pid);
```

这里假定 id 同时又作为兄弟之间的排序字段，如果我们把 name 字段作为兄弟之间的排序字段那么重设幺弟标志值的语句改为如下语句即可：

```
update xmw_tree set end = '0';
update xmw_tree set end = '1' where name||pid in (select max(a.name)||a.pid from xmw_tree a,xmw_tree b where a.pid =b.id(+) group by a.pid);
```

有了幺弟标志下面就要解决表 2 中的 name 值以及作为普通列表的排序字的问题。由于树型结构的层数总是有限的，下面我们考虑最大层数为 5 层的树的 sql 实现，最大层数为 n 层的树的 sql 实现可照此方法类推。

对于最大为 5 层的树，表 2 中某一行的 name 值最大可有五段每段的形成可如下考虑：

第一段，若存在曾曾祖父并且曾曾祖父在其兄弟排行中不为幺弟则显示'| '，若为幺弟则显示'└'，否则不显示任何内容。

第二段，若存在曾祖父并且曾祖父在其兄弟排行中不为幺弟则显示'| '，若为幺弟则显示'└'，否则不显示任何内容。

第三段，若存在祖父并且祖父在其兄弟排行中不为幺弟则显示'| '，若为幺弟则显示'└'，否则不显示任何内容。

第四段，若存在父亲并且父亲在其兄弟排行中不为幺弟则显示'| '，若为幺弟则显示'└'，否则不显示任何内容。

第五段，若存在父亲而自己在兄弟排行中不为幺弟则显示'├'，若为幺弟则显示'└'，否则显示'●'。

至于排序字若以 id 作为兄弟之间的排序字段，那么曾曾祖父的 id + 曾祖父的 id + 祖父的 id + 父亲的 id + 自己的 id 即为此树结点的全局排序字。若以 name 作为兄弟之间的排序字段，那么曾曾祖父的 name + 曾

祖父的 name + 祖父的 name + 父亲的 name + 自己的 name 即为此树结点的全局排序字。如此一来下面的语句就不难理解了，问题也就到此基本解决了，最后只要把曾曾祖父、曾祖父、祖父、父亲和本人的关系设条件联系起来即可，其中下面语句中 from 后的 a、b、c、d、e 对应的表可看作 5 本家谱，where 条件把五本家谱通过父子关系级联起来，级联要使用外链接，否则只能得到五代同在的记录。下面就是分别以 id 作为兄弟之间排序字段和 name 作为兄弟之间排序字段的 SQL 语句，请参考。

(1) 以 id 作为兄弟之间排序字段的树型结构表的 SQL 查询语句

```
select a.ID,
    decode(e.id,null,'',decode(e.end,1,'','|  '))||
    decode(d.id,null,'',decode(d.end,1,'','|  '))||
    decode(c.id,null,'',decode(c.end,1,'','|  '))||
    decode(b.id,null,'',decode(b.end,1,'','|  '))||
    decode(a.pid,0,'●',decode(a.end,1,'└','├'))
||trim(a.name)
    from xmw_tree a,xmw_tree b,xmw_tree c,xmw_
tree d,xmw_tree e
    where a.pid = b.ID(+) and b.pid = c.ID(+)
and c.pid = d.ID(+) and d.pid = e.ID(+)
    order by e.id||d.id||c.id||b.id||a.id;
```

(2) 以 name 作为兄弟之间排序字段的树型结构表的 SQL 查询语句

```
select a.ID,
    decode(e.id,null,'',decode(e.end,1,'','|  '))||
    decode(d.id,null,'',decode(d.end,1,'','|  '))||
    decode(c.id,null,'',decode(c.end,1,'','|  '))||
    decode(b.id,null,'',decode(b.end,1,'','|  '))||
    decode(a.pid,0,'●',decode(a.end,1,'└','├'))
||trim(a.name)
    from xmw_tree a,xmw_tree b,xmw_tree c,xmw_
tree d,xmw_tree e
    where a.pid = b.ID(+) and b.pid = c.ID(+)
and c.pid = d.ID(+) and d.pid = e.ID(+)
    order by e.name||d.name||c.name||b.name||
a.name;
```

3 结束语

以上论述了用 ORACLE 的 SQL 语句实现带线树结构展示的方法，希望本文的内容对大家今后使用 ORACLE SQL 语句有所启示。

参考文献

1 [美] Christopher Allen 著，Oracle PL/SQL 程序设计基础教程 - 北京，机械工业出版社，2001.5.