PowerBuilder 下动态报表技术的实现

李文继 （山东财政学院经济信息管理系 250014）

摘要：本文介绍了一个通用动态报表窗口对象在 PowerBuilder 应用开发环境下的实现，说明了这一对象所具有的通用方便的使用特征。分析了在实现这一通用动态报表窗口对象时所要解决的若干主要技术问题，并给出了解决这些主要技术问题的实现细节。

关键词：动态报表 Powerbuilder Datawindow

在 PowerBuilder 下我们需要借助 Datawindow 实现对数据库中表的数据操纵、显示及打印等操作。而 Datawindow 是我们事先用 Datawindow 画板制做好的，也即是说在应用中数据窗口是固定的。但许多应用中需要动态地提取一些数据，如人事管理中，经常会因工作需要动态从人事档案中选择某些项进行打印输出。本文介绍的就是这样一个动态报表技术的实现。

一、重点技术问题

要想实现动态报表功能，需要解决如下几个问题：

1. 如何动态在 datawindow 中增加、删除列

首先 datawindow 可采用 grid 风格，因为在 grid 风格的 datawindow 中增加或删除一列，datawindow 会自动调整插入、删除列后面有关列的位置，相应地后移或前移。而若采用其他风格，则需编写程序脚本将 datawindow 中的对象进行移动。

删除 datawindow 中的对象我们可以利用 modify() 函数，比如删除一名为'bh'这样的一列可用这样的语句：dw1.modify('destroy bhtdestroy bh-t')。但是在这里介绍的动态报表窗口对象中并不是采用这种方式，而是用 modify() 函数将该列对象的 X 属性（水平位置属性）及 Width 属性（宽度属性）分别置为零以使该对象不可见（有关脚本见后），其目的在于我们不想真正在 datawindow 中将其删除，以便可以通过"增加列"的功能将其再恢复回来。

在一开始调用动态报表窗口对象时，此 datawindow 中包含了尽可能多的列。而经过一系列的删除操作之后，如上所述，许多列被隐含了起来。要想将其恢复，就

应用另外的一个窗口将所有这些隐含的对象以列表的形式显示出来，让用户选择要增加(恢复)的列。因此，关键的问题是如何搜索 datawindow 中的对象，找到这些隐含的列对象。查阅 datawindow 技术手册，可以知道 datawindow 有一属性是 objects，它以 Tab 键(t)为分隔记录了 datawindow 下的所有对象名；同时我们知道，在构造 datawindow 时，PowerBuilder 对每个表示列名的 Text 文字对象，其取名规律为'列名'加'-t'。有了这两点后，获取 datawindow 中的隐含列也就不难了，具体程序脚本可见下面的"有关程序脚本"部分。

2. 如何动态调整报表标题位置

第一个问题是如何填加标题。在 grid 风格的 datawindow 的 header band 中，插入一个 Static text 对象时，该文字对象会被安排得与某列同宽，而报表标题应该是与整个报表同宽的。改变的方法是将文字对象的 layer 属性修改为'foreground'(在 Properties 下的 Position 属性组下)，这样该文字对象即可任意调整其位置及大小了。

第二个问题是如何动态调整报表标题宽度。当我们用"增加列"、"删除列"功能进行增加、删除列的操作时，整个报表的宽度发生的变化，这样必须在发生增加、删除列事件的同时调整标题的 width 属性使之等于整个报表的宽度(标题文字对象的对齐方式应为居中)。为了计算报表宽度，就必须搜索 datawindow 中的所有对象，找到其 X 属性值与 Width 属性值之和最大的值，该值即为整个报表的宽度。

3. 如何将设定好的 datawindow 予以保存，以便再次调用

当用户将一个 datawindow 中的列予以增删调整好之后，就可选择动态报表窗口对象的保存功能将该 datawindow 的定义语法保存起来，以备将来长期使用。这样需要解决两个问题：一是语法的保存。要将 datawindow 语法予以永久性的保存，必须借助数据库表，这样需建立一个如下结构的表：

表名：sys-dwsyntax

字段列名	字段类型	字段描述
Dwid	Smallint	Datawindow 编号
Dwname	Char(20)	dw 名称，用做提取 dw 的标识
Dwdesc	Char(40)	Dw 描述，通常用做选择提示
Dwsyntax	Text	dw 语法

由于 PowerBuilder 的 update( )函数对表中 text 类型列的直接存取操作有长度限制，因此语法的保存、调用必须用嵌入式 SQL 语句完成，而提取 datawindow 语法及动态生成 datawidow 可用如下两个语句完成：

```
dwsyntax = dw1.describe("datawindow.syntax")
/* 获取 dw1 的语法，并将其保存到变量 dwsyntax 中 */
dw-1.create(dwsyntax, errmess)
/* 根据 dwsyntax 所提供的语法动态生成 datawindow */
```

通过上述讨论解决了在实现动态报表打印中的有关技术问题，下一部分将给出实现中的一些重要程序脚本，供读者参考。

二、有关程序脚本

1. 在报表中删除一列

```
/* 变量 colname 为要删除列的对象名称 */
dw-1.modify(colname + '.x = 0') //将列的水平位置变为最左端
dw-1.modify(colname + '-t.x = 0') //将列标题的水平位置也变为最左端
/* 将列移到最左端的目的是为了使其不影响计算整个报表宽度 */
dw-1.modify(colname + '.width = 0') //将该列宽度变为零，即隐藏此列
dw-1.modify(colname + '-t.width = 0') //将列标题宽度变为零，隐藏列标题
fn-chgtitlewidth( ) //调整报表标题宽度，使其与整个报表同宽
```

2. 在报表中插入一列

```
dwobjects = dw-1.describe('datawindow.objects')
/* dwobjects 记录了 dw-1 中所有的对象名称(对象名称之间是用t分隔的) */
objlist = fn-analysis(dwobjects)
/* 函数 fn-analysis 用来分析 dwobjects，找出所有隐含的列对象。返回时变量 objlist 中就形成了'汉字列名 1t 表列名 1/汉字列名 2t 表列 名 2..../汉字列名 nt 表列名 n'这样一个用 Tab 和'/'分隔的一个关 于隐含列名列表的字符串。 */
if objlist<>'' then //若有隐含列则执行下面的恢复操作
open(w-columnlist) //w-columnlist 用于让用户选择
```

要恢复的隐含列

```
dw-1.modify(colname + '.x = ' + objx) //colname 为用户选择的列名
dw-1.modify(colname + '.width = ' + objwidth)
/* 上述两语句用于将原隐含的列予以恢复 */
fn-chgtitlewidth() //调整报表标题宽度,使其与整个报表同宽
end if
```

3. 计算报表宽度

```
/* 在调整报表标题宽度之前需先计算出报表宽度。本函数的输入 是一个 datawindow 类型的变量 dw1,返回值为给定 datawindow 的宽度值 */
maxx = 0
dwobjects = dw1.describe('datawindow.objects')
l1 = 1
do
  l2 = pos(dwobjects, 't', l1)
  if l2 = 0 then
    objname = mid(dwobjects, l1)
  else
    objname = mid(dwobjects, l1, l2 - l1)
  end if
  if pos(objname, '-t') > 0 then
    objx = dw-1.describe(objname + '.x')
    objwidth = dw-1.describe(objname + '.width')
    if long(objx) + long(objwidth) > maxx then
      maxx = long(objx) + long(objwidth) + 6
    end if
  end if
  l1 = l2 + 1
loop while l2 > 0
return maxx //返回报表宽度
```

4. 保存报表语法

```
dwsyntax = dw1.describe("datawindow.syntax")
/* 获取 dw1 的语法,并将其保存到变量 dwsyntax 中 */
UPDATE sys-dwsyntax
  SET dwsyntax = :dwsyntax
    WHERE dwname = :dwname ;
/* 将 datawindow 的语法保存到表 sys-dwsyntax 中 */
```

5. 调入报表语法并生成 datawindow

```
/* 下面的语句用于将 datawidow 语法由表 sys-dwsyntax 中读到 变量 dwsyntax 中 */
SELECT dwsyntax INTO :dwsyntax FROM sys-dwsyntax
  WHERE dwname = :dwname ;
if isnull(dwsyntax) or trim(dwsyntax) = '' then
  messagebox('错误提示','数据窗口语法错误')
  return
end if
/* 下面的语句使用 dwsyntax 变量中记录的 datawindow 语法来动态生成 datawindow */
if dw1.create(dwsyntax, errmess) <> 1 then
  messagebox('报表格式错误', errmess)
end if
```

三、使用方法

上面介绍了通用动态报表窗口对象实现中的技术问题,其引用方法是:在调用此窗口对象的 open()函数中将一个 datawindow 类型的变量通过 message 对象传递给它。用户可通过该窗口提供的一系列功能选项,包括增加列、删除列、转入、保存等,对给定报表 datawindow 进行一系列的增删修改操作。对于已经设定好的 datawindow,可调用动态报表窗口对象所提供的"保存"功能将其语法保存到表 sys-dwsyntax 中。

要调用已设定好的 datawindow 进行数据输出,可使用一个窗口列出所有用户设定好的 datawindow(即列出 sys-dwsyntax 表中的 dwdesc 列),由用户选定所要使用的 datawindow,然后调入报表语法并生成 datawindow;再从有关数据库表中提取数据予以输出。

四、总结

在许多应用系统中,都有随机从数据库表中提取有关数据项的需求,也即是说用户对查询及报表功能的需求是变化的、增加的。本文所介绍的动态报表窗口对象技术正是为了适合这种需求而实现的。由于它采用了建立窗口对象的实现方式,因此具有较好的通用性,可在各种不同的应用系统中引用这一对象来实现动态报表功能。

(来稿时间:1998 年 3 月)