

第 7 章 数据窗口对象

教学提示：本章介绍 PowerBuilder 提供的开发数据库应用的强有力的工具——数据窗口，数据窗口包括数据窗口控件和数据窗口对象，数据窗口对象检索、表现、删除、更新相关数据库或其他数据源中的数据提供了非常方便的手段。开发人员可以为数据指定显示格式、编辑风格及有效性规则等数据属性。另外，在数据窗口中还可以添加各种对象、制表的附加信息、统计图及生成报表等。

教学要求：通过本章学习，读者应该掌握以下内容：根据需要设计数据窗口对象，熟悉数据窗口画板及其基本操作，理解不同的数据源和显示风格，设置数据窗口的各种对象和属性，数据窗口的打印设置，统计图表的使用。

7.1 创建数据窗口对象

PowerBuilder 提供了创建数据窗口对象的向导，利用它可以方便、快捷地创建出数据窗口对象。具体使用步骤如下。

(1) 单击工具栏【New】按钮，接着弹出【New】对话框，选择对话框中的【DataWindow】标签页，如图 7.1 所示。

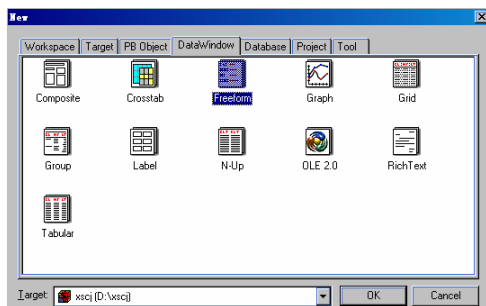


图 7.1 选择数据窗口的显示风格

(2) 【DataWindow】标签页中列出了 11 种数据窗口的样式。每种样式代表了一种独特的显示风格，将在 7.3 节中介绍。例如，选择【Tabular】显示样式，单击【OK】按钮，弹出选择数据源对话框。

(3) 选择数据源对话框如图 7.2 所示。PowerBuilder 提供了 5 种类型的数据源，分别是 Quick Select、SQL Select、Query、External 和 Store Procedure 类型。有关数据源的内容将在随后的 7.2 数据源一节中介绍。在该对话框底部有一个【Retrieve on Preview】复选框，如果在预览数据窗口时不需要检索数据，则可以不选中它。例如，选择【Quick Select】类型的数据源，选中预览时检索数据的复选框，单击【Next】命令按钮，弹出【Quick Select】

数据源对话框。

(4) 【Quick Select】数据源对话框主要完成对数据库中的表以及表中要显示字段的选择,如图7.3所示。图中左边的【Tables】列表框中列出了当前连接的数据库中的所有表名,单击某一个表名时,会在右边的【Columns】列表框中列出该表的全部字段。单击某个字段名时,该字段就会变为蓝色显示,同时在【Quick Select】数据源对话框底部的描述框中加入该字段。如果需要取消某个已选中的字段,只需再次单击该字段既可。如果需要显示所有字段,只需单击右边的【Add All】命令按钮。完成字段选择后,单击【OK】按钮,接着弹出【Select Color and Border Setting】颜色和边框设置对话框。

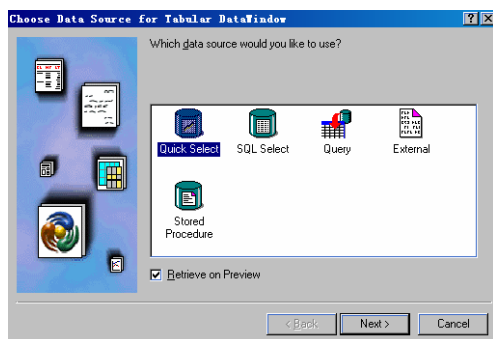


图 7.2 选择数据源类型

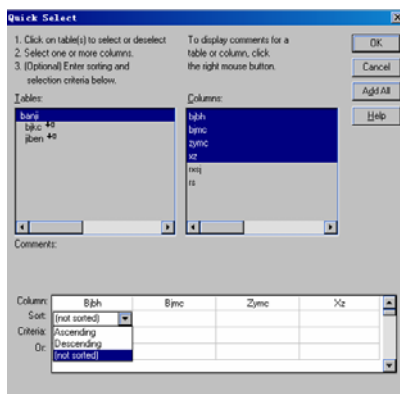


图 7.3 设置显示字段

(5) 在【Select Color and Border Setting】颜色和边框设置对话框中,对数据窗口的背景颜色、字段标签的颜色和边框类型以及字段的颜色和边框类型进行设置,如图7.4所示。单击【Next】命令按钮,弹出【Ready to Create Freeform DataWindow】对话框。

(6) 【Ready to Create Freeform Datawindow】对话框显示了关于新建数据窗口对象属性的列表,供设计者检查、确定,如果有问题,随时可以返回上一步操作重新选择和设置数据窗口对象的属性,如图7.5所示。单击【Finish】按钮,创建数据窗口对象的工作即告初步完成,转入数据窗口画板,如图7.6所示。

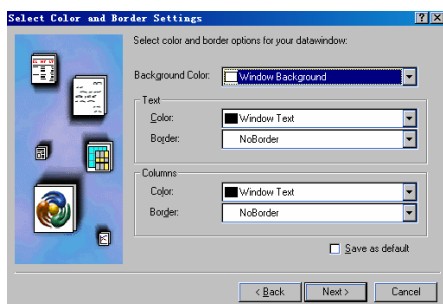


图 7.4 设置数据窗口背景和字体颜色

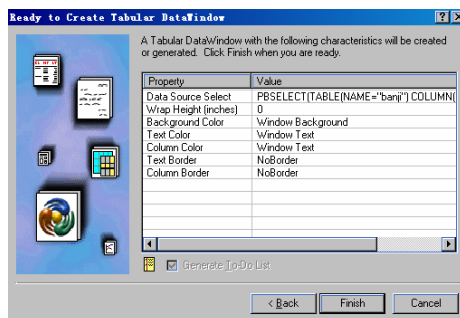


图 7.5 新建数据窗口对象属性的列表

(7) 在数据窗口画板中,可以对数据窗口对象做进一步的设计修改。例如,可以随意拖动字段,改变字段的位置和大小,改变字体的大小、类型和颜色,改变背景的颜色,改

变字段边框的显示效果, 改变字段名称的文本等。

(8) 单击工具栏上的【Save】保存按钮, 这时会弹出保存数据窗口对象对话框, 如图 7.7 所示。在第一行编辑中为新建的数据窗口对象命名, 在【Comment】注释区中可以编写一小段对新建的数据窗口对象的说明文本。单击【OK】命令按钮, 保存该数据窗口对象。

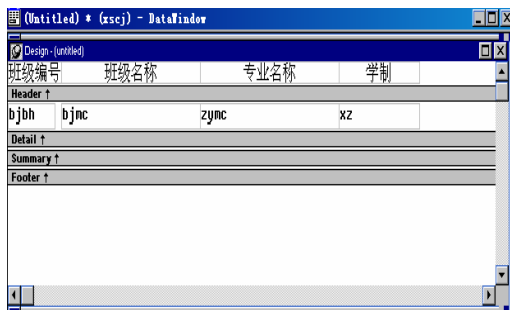


图 7.6 新建的数据窗口对象

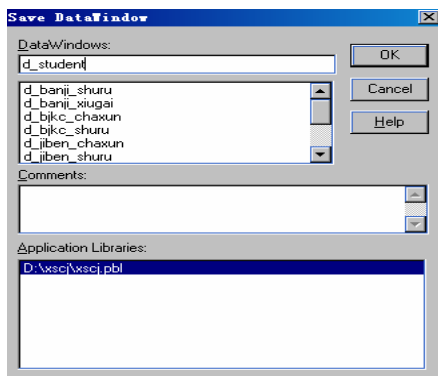


图 7.7 保存数据窗口对象

创建数据窗口对象时, 主要包括两方面的内容: 数据和显示风格。下面将分别加以介绍。

7.2 数据窗口的数据源

数据源就是数据窗口对象的数据来源。定义数据源决定了数据窗口对象获得数据的方式, 即数据窗口对象从什么地方得到数据, 如何得到数据以及怎样得到数据。在上一节建立数据窗口对象的步骤 3 中已经看到了 PowerBuilder 提供的 5 种数据源。下面进行具体说明。

7.2.1 Quick Select (快速选择数据源)

Quick Select(快速选择数据源)是最简单也是最常用的一种数据源形式。它能够创建简单的 SQL Select 语句, 主要用于从一个表或由外部的多个表中选择数据列, 但不能生成计算列。Quick Select 数据源定义出一条简单的 Select 语句, 这条语句可以指定选择的列、查询条件及排序方式, 但不支持分组 Group、计算机 Computed、提取参数 Having 等复杂的 SQL Select 功能。定义快速选择数据源的基本步骤在上一节建立数据窗口对象的步骤 4 中已经介绍。需要进一步说明的是, 在【Quick Select】数据源对话框底部的描述框中显示的就是当前设计的数据窗口, 如图 7.8 所示。其中, 【Column】行列出了选中各列的标题; 【Sort】行用于指定按哪些列排序以及排序方式。如果希望查询结果按某列排序, 那么单击该列下的【Sort】行, 然后从下拉列表框中选择所需排序方式, 其中: 【Ascending】为升序, 【Descending】为降序, 【Not Sorted】为不排序。通过拖曳操作可以改变各列的位置; 【Criteria】行及其下面的行用于指定查询条件。查询条件中能够使用任何 SQL 关系

操作字符, 包括 =、>、>=、<、<=、<>、LIKE、IN 等。如果只输入了一个值而未指定操作符, 系统就假定操作符为 =(等于)。另外, 可以使用逻辑操作符 AND、OR 来连接表达式。如果输入了多个表达式而没有逻辑运算符, 系统就使用下述规则添加上逻辑运算符: 同行使用与操作符“AND”, 不同行使用或运算符“OR”。

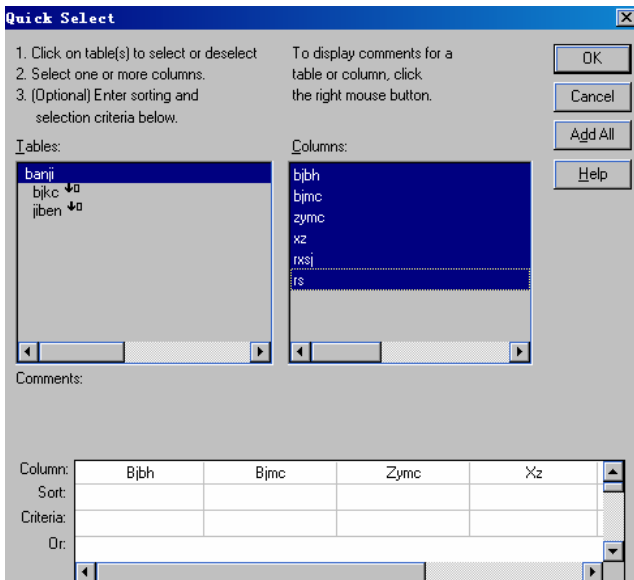


图 7.8 为【Quick Select】数据源选择表和列

7.2.2 SQL Select(SQL 选择数据源)

SQL Select(SQL 选择数据源)是一种功能全面的数据源。“SQL 选择”以可视化的方式建立。SQL Select 语句的所有细节均能通过该界面创建, 主要是用于从一个或多个表中建立复杂的 SQL Select 语句, 当然也能生成各种各样的计算列。SQL Select 数据源能够从多个表中选择列、指定查询条件、对数据排序、分组、增加计算列、定义提取参数等。下面对 SQL 选择数据源的实现过程进行介绍。

1. 定义 SQL Select 数据源

(1) 在本章 7.1 节建立数据窗口对象的步骤 3 的数据源对话框中, 选择【SQL Select】数据源后, 单击【OK】按钮, 系统显示如图 7.9 所示的【Select Tables】对话框。

(2) 【Select Tables】对话框通过单击选择数据窗口中要使用的一个或多个表, 选择之后单击【OK】按钮, 进入 SQL 画板工作区。

(3) SQL 画板工作区以图形方式显示所选表, 当打开了多个表且表之间存在外部键时, SQL 画板自动建立外部键之间的连接, 如图 7.10 所示。

(4) 在列出的表中选择所需要的列, 选定的列被加亮, 同时也出现在【Selection List】后面, 其次序就是各列出现在 Select 语句中的次序, 通过拖放操作能够改变这一排列次序, 如图 7.11 所示。

(5) 在 SQL 画板工作区的下部有一组标签, 如图 7.12 所示, 这是该画板的检索条件定义区, 用于定义 Select 语句的各种子句(后文介绍定义方法)。

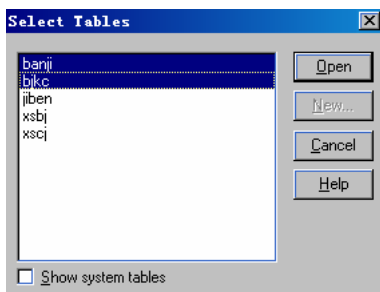


图 7.9 选择数据表

banji Label	Type
bjbh 班级编号:	char(6)
bjmc 班级名称:	varchar(20)
zymc 专业名称:	varchar(20)
xz 学期:	numeric(1,0)
rxsj 入学时间:	date
rs 人数:	numeric(3,0)

bjkc Label	Type
bjbh 班级编号:	char(6)
xq 学期:	char(9)
kcmc 课程名称:	varchar(20)
xs 学时:	numeric(3,0)
jsxm 教师姓名:	char(12)

图 7.10 图形方式打开的数据表

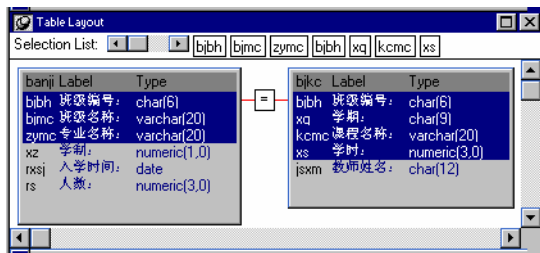


图 7.11 选择显示字段

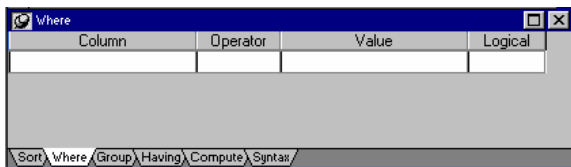


图 7.12 【Where】标签页

(6) 完成列的选择以及指定各种条件后(即使遗漏了某部分也没关系,因为在以后能够重新进入该状态并做相应修改),单击画板工具栏上的 SQL 图标,系统进入数据窗口画板工作区。定义检索条件在 SQL 画板工作区下方【Where】标签页中。

2. 定义 SQL Select 数据源的检索条件

单击【Where】标签,系统显示如图 7.12 所示的【Where】标签页,随后可以进行下述操作。

(1) 单击【Column】标签下的第一个空白行,【Column】标签右边出现黑色小三角▼,单击小三角▼,系统显示一个列名下拉列表框。也可以直接在【Column】行的右侧单击,直接展开显示列名的下拉列表框。从下拉列表框中选择一个列名,见图 7.13。

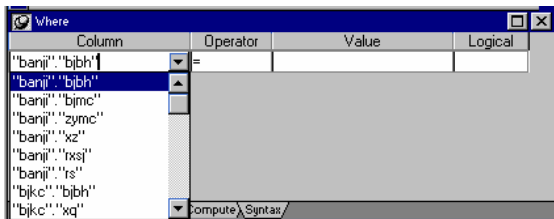


图 7.13 选择【Where】子句中的字段名

(2) 单击【Operator】标签下的第一行，系统显示一个运算符下拉列表框，从中选择所需的运算符，如图 7.14 所示。

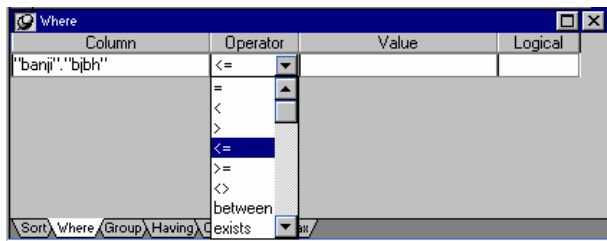


图 7.14 【Where】子句中的运算符

(3) 单击【Value】标签下的第一行，输入一个表达式，表达式由列名、数据库管理系统支持的函数、开发人员定义的检索参数(见后面检索参数的定义方法)、常量数值或子查询组成。

更常用的方法是在【Value】行内右击，系统显示如图 7.15 所示的弹出菜单，其中，选择【Columns】字段、【Functions】函数、【Arguments】变量、【Value】数值等菜单项后，系统打开相应的对话框。通过单击选择合适的选项后，单击【Paste】按钮，所选项粘贴到该单元。选择【Select】菜单项后，系统打开另一个 SQL 画板工作区，在这个工作区中定义子查询，这样就可以把子查询的返回值作为条件表达式的一部分，从而构造出更复杂的条件。

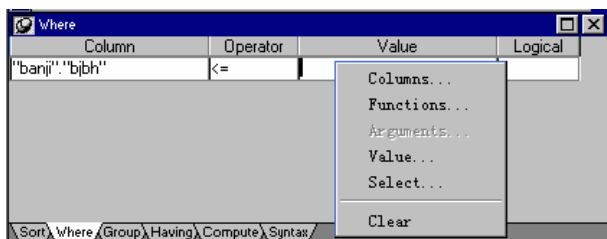


图 7.15 在【Value】标签页下的空白栏右击弹出菜单

(4) 需要多个条件时，单击【Logical】标签下的第一行，根据需要选择【And】|【or】逻辑运算符后，在下一行继续重复上面介绍的步骤。例如可以定义查询条件：

`("banji"."bjbh">="010218"AND ("banji"."bjbh"<"010222"))`。

3. 定义检索参数

在定义检索条件时，如果条件中的值在运行时才能确定，那么需要使用检索参数。例如，需要按照学生的姓名检索学生的情况，那么，学生的姓名就是检索参数，它只有在应用程序运行时，由用户输入后才能确定。检索参数是在【Where】子句中使用的参数，其定义方法如下。

(1) 选择【Design】|【Retrieval Arguments】命令，弹出如图 7.16 所示的【Specify Retrieval Arguments】对话框。

(2) 在【Name】列键入参数名称。

- (3) 在【Type】列选择参数类型，例如 String。
- (4) 需要添加多个参数时，单击【Add】按钮，然后键入参数名称并指定参数类型。
- (5) 需要在当前参数前插入一个参数时，单击【Insert】按钮，然后键入参数名称并指定参数类型。
- (6) 需要删除某个参数时，通过单击该参数的名称选择该参数后单击【Delete】按钮。
- (7) 单击【OK】按钮关闭对话框。定义了检索参数后，就可以使用检索参数构造【Value】列上的表达式了。在表达式中使用检索参数时，需要在参数前放上个冒号(:)，例如写上: ParaName，以告诉 PowerBuilder 这是个检索参数，而不是列名。在条件中使用参数后，应用程序就能够根据运行情况动态地检索数据了。

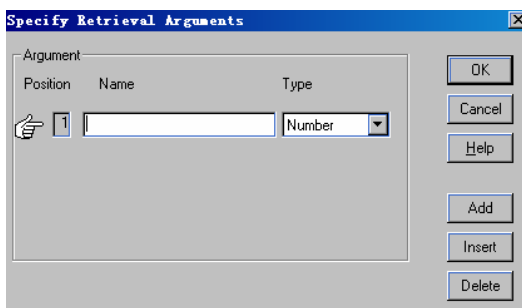


图 7.16 【Specify Retrieval Arguments】对话框

4. 指定排序方式

当希望检索出的数据按某些列进行排序时，应该定义排序方式，步骤如下。

- (1) 单击【SQL】选项卡中的【Sort】标签，系统显示如图 7.17 所示的标签页。
- (2) 把希望按其排序的列用鼠标从左边的列表框中拖曳到右边的列表框中，此时将按该列升序排序。如果想按该列降序排序，那么通过单击使【Ascending】复选框成为未选中状态。
- (3) 选择其他要排序的列。例如，指定按"banji"."bjbh"列进行升序排序，它对应于 Select 语句中的子句：

```
ORDER BY "banji"."bjbh"ASC
```

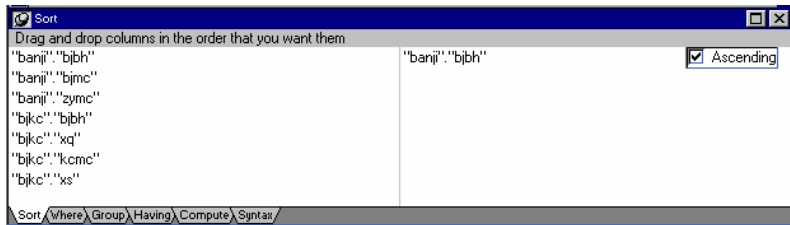


图 7.17 【SQL】选项卡中的【Sort】标签页

5. 定义计算列

计算列不是数据库表中的原始列，而是通过表达式运算得到的列。例如，表中有“数学成绩”和“英语成绩”两个数值型列，则“数学成绩”+“英语成绩”形成的列就是一

个计算列。定义计算列的步骤如下。

- (1) 单击【SQL】工具栏中的【Compute】标签，显示【Computer】标签页。
- (2) 在第一行中键入组成计算列的表达式。构造表达式时，也可以使用工具。方法是：在该行右击，显示如图 7.18 所示的弹出菜单，其中，选择【Columns】、【Functions】、【Arguments】命令后，系统打开相应的对话框，通过单击选择合适的选项后，单击【Paste】按钮，所选项被粘贴到插入点位置。
- (3) 需要多个计算列时，通过单击将插入点移动到下一行，按上述方法构造组成计算列的表达式。

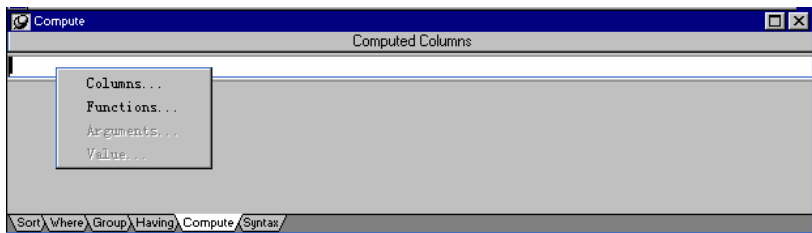


图 7.18 【Compute】选项页和单击右键的弹出菜单

6. 定义分组

在 SQL Select 数据源中，开发人员可根据应用程序的需要定义分组，方法如下。

- (1) 单击【SQL】工具栏中的【Group】标签，系统显示如图 7.19 所示的选项页。
- (2) 选择分组所依据的第一列，用鼠标把它拖到右边的列表中。
- (3) 如有必要，选择分组所依据的其他列。

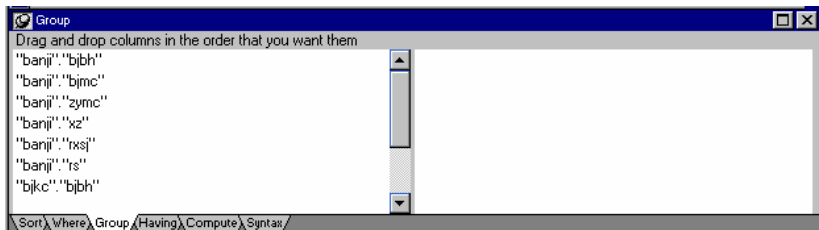


图 7.19 【Group】标签页

7. 定义 having 子句

定义了分组条件后，开发人员还可以定义 Having 子句，以对分组进行过滤，只检索那些满足的分组。定义 Having 子句的步骤如下。

- (1) 单击【SQL】工具栏中的【Having】标签，系统显示如图 7.20 所示的标签页。
- (2) 定义【Having】子句的条件表达式，方法与定义【Where】条件相同。

8. 显示当前定义条件下的 SELECT 语句

在定义 SQL Select 数据源的过程中，随时都可以查看当前定义条件下的 SELECT 语句，方法是：单击【SQL】工具栏中的【Syntax】标签，相应的 Select 语句显示在该标签页中。在这个标签页中，虽然不能直接编辑、修改 SELECT 语句，但可以通过拖曳操作选中部分

或全部语句后，按【Ctrl+C】组合键将其复制到系统剪贴板上。之后，可以按【Ctrl+V】组合键将剪贴板的内容粘贴到任何需要的地方，例如粘贴到代码编辑器中。

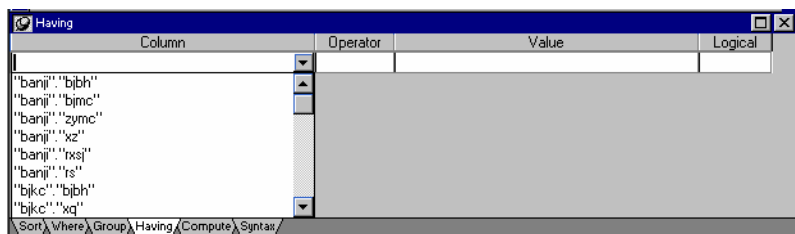


图 7.20 【Having】标签页

9. 直接键入 SELECT 语句

如果十分熟悉 SQL 语句，或图形方式构造的 SELECT 语句不能满足应用程序的需要，那么可以直接键入或编辑 SELECT 语句，完成 SQL Select 数据源的定义。方法如下。

(1) 从数据源画板上选择【Design】|【Convert to Syntax】命令，系统打开一个文本编辑窗口，如图 7.21 所示。

(2) 键入或编辑 SELECT 语句。

(3) 编写完 SELECT 语句后，选择【Design】|【Convert to Graphics】命令就返回到图形方式(如果 SQL Select 语句中包含了某些数据库专有函数，则可能无法转换到图形方式)，或单击【Data Source】图标进入数据窗口画板。

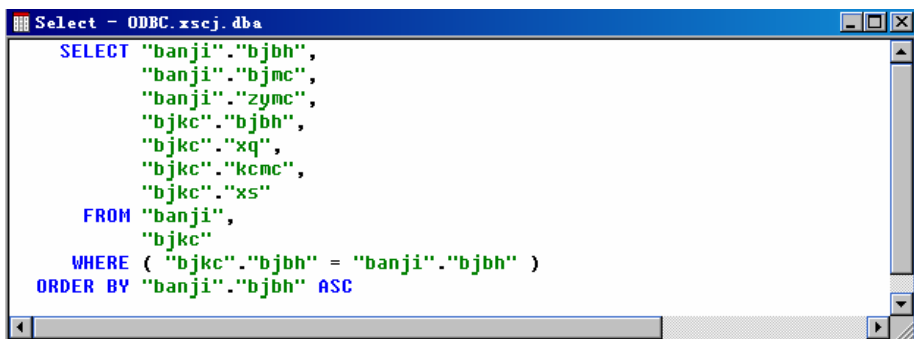


图 7.21 SELECT 语句编辑窗口

7.2.3 Query(查询数据源)

Query(查询数据源)是将以前创建的 Query 对象作为数据窗口的数据来源。Query 数据源选取 Query 对象作为数据源，Query 对象实际上就是保存在应用程序库中的 SELECT 语句，使用时，可以对 Query 对象提供的 SQL 语句进行修改。定义 Query 对象的目的是为了在多个数据窗口中重复使用或相近的 SELECT 语句而避免反复定义。与定义 SQL Select 数据源相似，在 Query 对象中可以定义检索参数、指定排序方式和分组方式、定义检索条件等。

1. 创建 Query 对象

在定义 Query 数据源之前, 需要使用 Query 画板首先创建 Query 对象, 步骤如下。

(1) 单击工具栏上【New】图标按钮, 打开【New】对话框, 选择【Database】标签页, 如图 7.22 所示。

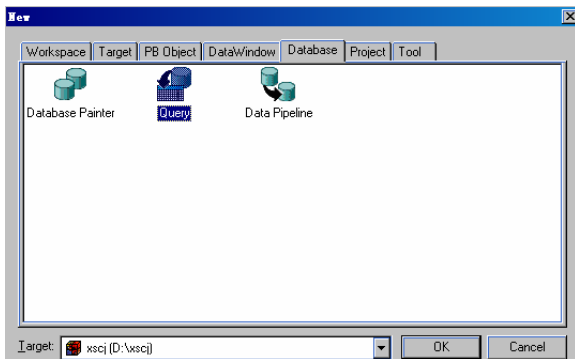


图 7.22 【New】对话框中的【Database】标签

(2) 双击【Query】图标, 进入 Query 画板, 并弹出【Select Tables】对话框, 如图 7.23 所示。

(3) 选择要使用的表后单击【Open】按钮, 进入 Query 画板工作区。

(4) 定义所需要的 Select 语句, 方法与定义 SQL Select 数据源的方法相似。

(5) 单击工具栏上的【Close】图标, 弹出询问是否需要保存的对话框, 保存 Query 对象后关闭 Query 画板。

2. 定义 Query 数据源

有了 Query 对象后, 定义 Query 数据源的方法如下。

(1) 单击工具栏【New】图标按钮, 选择【DataWindow】标签页, 选择数据窗口风格后, 单击【OK】键进入选择数据源对话框。

(2) 在选择数据源对话框中, 选择【Query】数据源后, 单击【OK】按钮, 系统显示如图 7.24 所示的【Select Query】对话框。

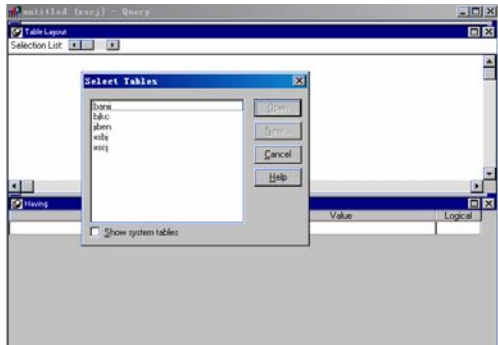


图 7.23 Query 画板中的【Select Tables】对话框

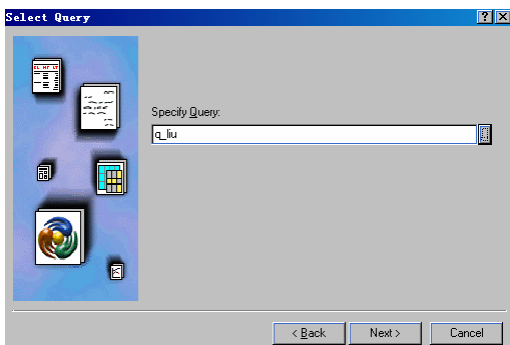


图 7.24 【Select Query】对话框

(3) 单击【Specify Query】栏右边的【...】按钮，选择需要的 Query 对象。

(4) 单击【Next】按钮进入边框设置对话框以及属性小结对话框，确定后进入数据窗口画板工作区。

7.2.4 External(外部数据源)

External(外部数据源)用于让数据窗口访问数据库之外的数据。比如文本文件，用户输入 INI 文件或其他非 DBMS 数据库来源的数据，同时在用户界面上能够充分发挥数据窗口的长处，避免复杂编程。External 数据源从外部文件(比如文本文件)中提取数据，它是数据窗口唯一不需要连接数据库的数据源，其数据或由应用程序生成或由用户输入。定义外部数据源的数据窗口时，必须定义它的每一列及其数据类型。

定义 External 数据源的步骤如下。

(1) 单击工具栏【New】图标按钮，选择【DataWindow】标签页，选择数据窗口风格后，单击【Next】键进入选择数据源对话框。

(2) 在选择数据源对话框中，选择【External】数据源后，单击【Next】按钮，系统显示如图 7.25 所示的【Defint Result Set】对话框。

(3) 指定数据窗口中所需要的列以及相应的类型和长度。

(4) 使用按钮【Add】、【Insert】、【Delete】分别增加、插入、删除数据列。

(5) 单击【Next】按钮进入边框设置对话框以及属性小结对话框，确定后进入数据窗口画板工作区。

上述操作过程只定义了 External 数据源的数据类型，并没有定义操作的数据。实际编程时，对这种数据源的数据窗口，还需要在程序代码中使用 Import 簇函数(比如 ImportFile、ImportString 簇函数等)向数据窗口中装入数据，或使用数据窗口控件的对象函数操作数据。例如，对应用程序中的数据，可以直接使用 SetItem 函数；对于从文件中读取数据，可以使用 File 簇函数或 Import 簇函数。需要注意的是，External 数据源不能用于 Crosstab 风格显示样式的数据窗口对象。Crosstab 风格显示样式将在下面的章节中介绍。

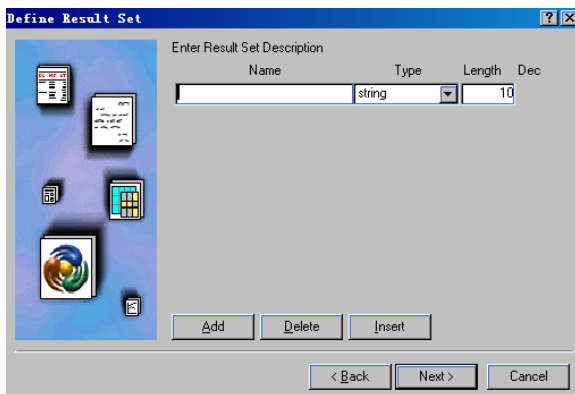


图 7.25 【Define Result Set】对话框

7.2.5 Stored Procedure(存储过程数据源)

Stored Procedure(存储过程数据源)则直接利用保存数据库中的存储过程作为数据

源, 这个数据源只有当前连接的数据库支持存储过程时才有效, 否则系统会自动隐藏该选项。Stored Procedure 数据源就是把存储过程作为数据源。存储过程(Stored Procedure)是一组保存在数据库中的、经过预先编译和优化的、执行数据库操作的 SQL 语句。与其他 SQL 语句相比, 存储过程的执行效率更高(省掉了每次执行时的编译与优化时间)。在数据窗口的五种数据源中, 存储过程与使用的具体数据库有关, 只有该数据库支持存储过程时, 数据源中才会有这个选项。使用存储过程有两个好处: 一、减少网络通信量; 二、提高查询速度, 原因在于使用存储过程时数据库管理系统避免了重复的语法分析与优化。

定义 Stored Procedure 数据源的步骤如下。

(1) 单击工具栏【New】图标按钮, 选择【DataWindow】标签页, 选择数据窗口风格后, 单击【Next】按钮, 进入选择数据源对话框。

(2) 在选择数据对话框中, 选择【Stored Procedure】数据源后, 单击【Next】按钮, 系统显示如图 7.26 所示的【Select Stored Procedure】对话框。

(3) 列表框中选择所需的存储过程。如果想在列表框中显示系统存储过程, 那么选中复选框【System Procedure】。

(4) 如果想让 PowerBuilder 自动生成结果集, 那么不要选中复选框【Manual Result Set】, 然后单击【Next】按钮进入边框设置对话框以及属性小结对话框, 确定后进入数据窗口画板工作区。

(5) 如果想自己定义结果集, 那么选中复选框【Manual Result Set】, 然后单击【Next】按钮, 系统显示如图 7.27 所示的【Define Store Procedure Result Set】对话框。

(6) 定义列及其类型、宽度。定义完所有列后, 单击【Next】按钮, 进入边框设置对话框以及属性小结对话框, 确定后进入数据窗口画板工作区。

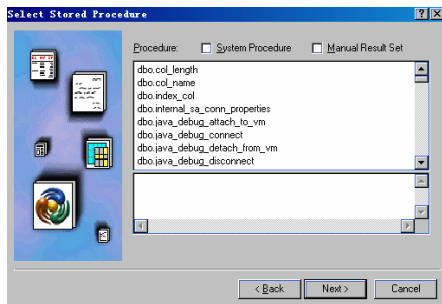


图 7.26 【Select Stored Procedure】对话框

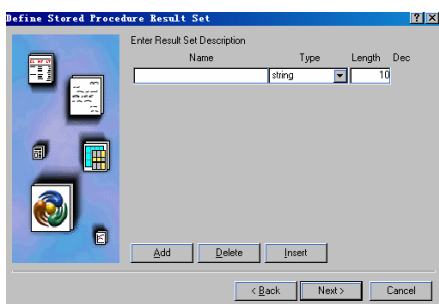


图 7.27 【Define Store Procedure Result Set】对话框

7.3 数据窗口的显示风格

数据窗口的魅力之一就在于它能够以 Presentationstyle(多种多样的显示风格)展现数据、表现数据, 过去需要大量编程才能完成的显示任务在数据窗口中也许只需要简单的选择。

显示风格决定了数据窗口以何种方式展开、表现数据。Powerbuilder 的数据窗口提供了 11 种显示风格: Tabular(列表)、Grid(表格)、Freeform(自由格式)、Label(标签)、N-up(分

栏)、Group(分组)、Crosstab(交叉列表)、Graph(统计图)、OLE2.0(外部文本)、RichText(超文本)、Composite(复合)风格。每种风格都有其独特的外观,并且上述风格只是定义了数据窗口的基本显示样式,通过设置数据窗口对象以及它所包含的其他对象的属性,就能够构造出风格各异的显示界面来。另外,在数据窗口对象内部,还能够校验、过滤数据、进行对其中的数据排序,并随时查看设计效果。下面详细介绍各种显示风格。

7.3.1 Freeform 显示风格

Freeform 格式为自由格式,这种格式通常一页只能显示一条记录,每一列数据都有标签并且数据窗口中分行垂直排列。在设计数据窗口对象时,可以随安排列标题和数据项的位置,进行任意排列。这种格式一般用于应用程序的数据输入界面,如图 7.28 所示。

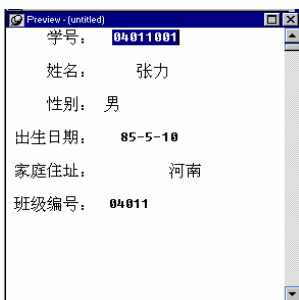


Figure 7.28 shows a preview window titled "Preview - (untitled)" displaying a form with the following data:

学号:	04011001
姓名:	张力
性别:	男
出生日期:	85-5-10
家庭住址:	河南
班级编号:	04011

图 7.28 【Freeform】显示风格

7.3.2 Grid 显示风格

Grid 格式即网格或普通表格格式,每一条记录占一行,列标题在所有列的顶端构成栏目名,一页可以显示多行记录,这种格式和 Microsoft Windows 系统的 Excel 电子表格很相似。

在设计 Grid 格式的数据窗口对象时,不能移动列和列标题,但在程序运行时,通过拖动网格线来改变列的宽度,还可以移动每一列来改变列的排列顺序。如图 7.29,显示了一个 Grid 格式的数据窗口对象。



Figure 7.29 shows a preview window titled "Preview - (untitled)" displaying a table with the following data:

学号	学期	课程名称	成绩
04011001	200420051	高等数学1	90.0
04011002	200420051	高等数学1	89.0
04011003	200420051	高等数学1	67.0
04011004	200420051	高等数学1	87.0
04011005	200420051	高等数学1	55.0
04011006	200420051	高等数学1	78.0

图 7.29 【Grid】显示风格

7.3.3 Graph 显示风格

理解 PowerBuilder 的图形表达方式对于设计图形风格的数据窗口对象是很重要的。PowerBuilder 图形表示的方法是将数据组织成 3 种元素: Series(系列)、Categories(类)

和 Values(值), 其中, 系列(Series)是一组数据点的集合, 每个系列都有不同的颜色、图案和符号。类(Categories)是数据的主体分割, 代表独立的变量。Values 就是依赖于变量的数据点的值。如图 7.30 所示可以形象地说明 Series(系列)、Categories(类)和 Values(值)的意义。

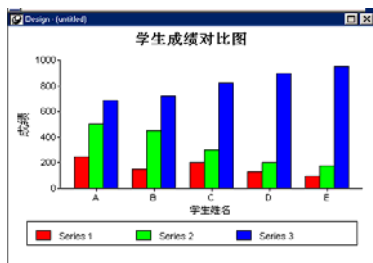


图 7.30 【Graph】显示风格的数据窗口

选择了图形风格后会弹出定义各坐标轴的对话框, 如图 7.31 所示。单击【Next】按钮, 弹出如图 7.32 所示的对话框, 输入标题并选择图形类型。单击【Next】按钮显示选择属性的小结对话框, 单击【Finish】按钮完成图形风格数据窗口对象的初步设计, 进入数据窗口画板。在数据窗口画板中, 可以对已经创建的数据窗口对象进行修改。

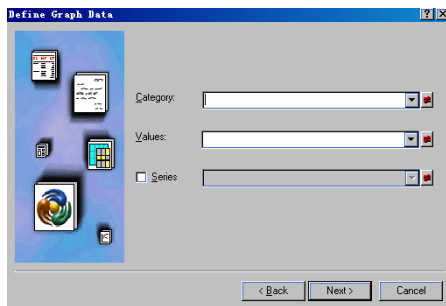


图 7.31 定义【Graph】坐标轴对话框

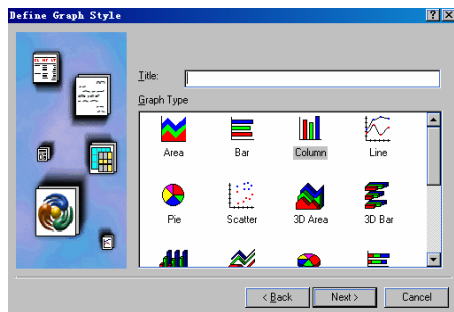


图 7.32 选择图表类型对话框

7.3.4 Composite 数据窗口

Composite 格式是组合已有的数据窗口对象, 所以在创建向导中会弹出选择数据窗口对象的对话框, 如图 7.33 所示。在数据窗口对象列表选择一个或多个数据窗口对象, 即可创建出组合式数据窗口对象。

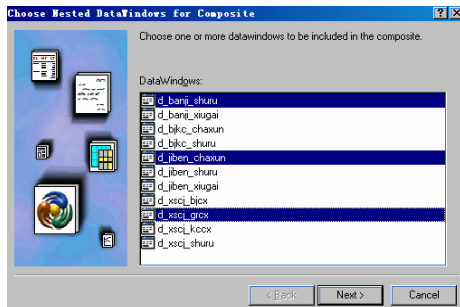


图 7.33 选择数据窗口对象对话框

Composite 格式数据窗口的特点是：

- (1) 不需要创建新的数据源。
- (2) 选择的数据窗口对象在组合样式中不能被修改。

7.3.5 Crosstab 显示风格

Crosstab(交叉列表)实际上就是常用的二维数据表。如图 7.34 所示为交叉列表的参数定义，还可以加上对每行、每列以及全部数据的统计分析。PowerBuilder 的交叉列表可以很方便地实现这些功能。理解了二维表的定义再去设置 Crosstab 的参数就不难了。另外，需要统计总的数量，所以交叉列表的值“Value”为字段“xscj_cj”的总和，即 Sum(xscj_cj for crosstab)。如果需要其他计算结果，可以双击列或行或值，弹出修改计算表达式对话框，如图 7.35 所示。可以利用 PowerBuilder 提供的函数 Functions、逻辑运算符修改计算表达式，或选择其他字段。

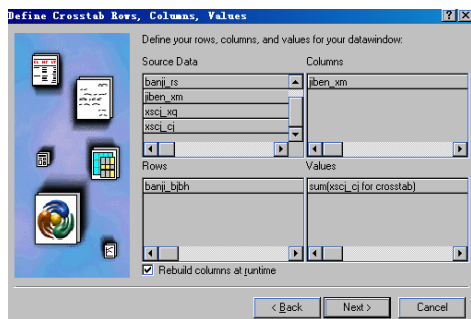


图 7.34 定义【Crosstab】参数

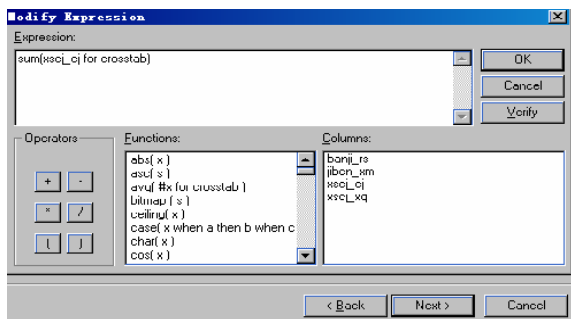


图 7.35 修改计算表达式对话框

7.3.6 Group 显示风格

Group 格式是分组显示数据，它的作用是使数据条理清晰。例如，按班级分组显示学生数据，就可以很快地找到某个学生的数据。

选择了【Group】格式后，会弹出分组定义对话框，将决定分组条件的字段从左边【Source Data】窗口中拖动到右边【Columns】窗口中，如图 7.36 所示。单击【Next】按钮弹出分组页属性设置对话框，如图 7.37 所示。

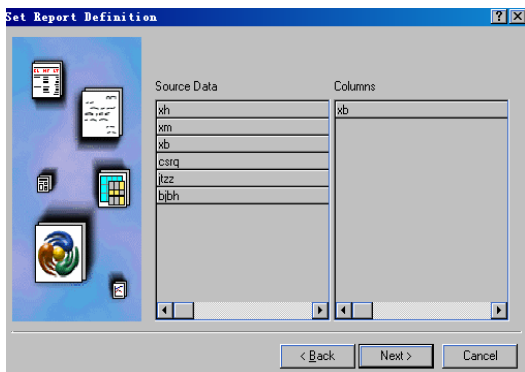


图 7.36 分组定义对话框

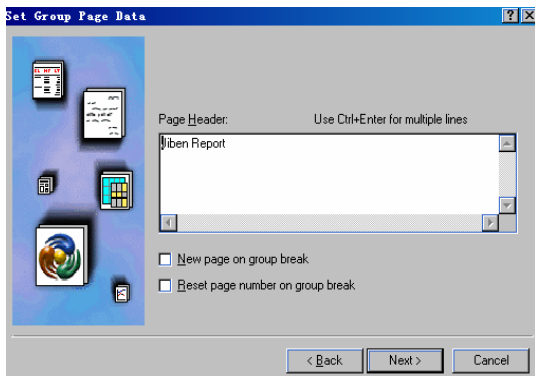


图 7.37 分组页属性设置定义对话框

7.3.7 Label 显示风格

在数据窗口对象创建向导中选择了【Label】格式后，会弹出选择预定义标签对话框，如图 7.38 所示。PowerBuilder 在下拉列表框中提供了很多尺寸。随后，弹出标签设置对话框，如图 7.39 所示。对标签的大小、布局以及排列方式作进一步设计。

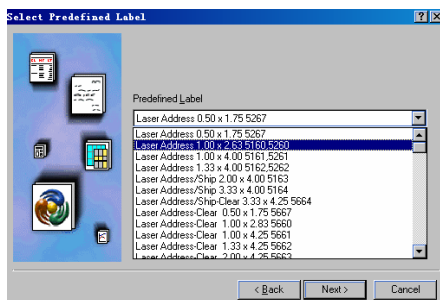


图 7.38 选择预定义标签对话框

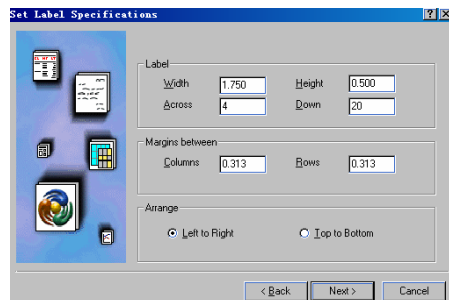


图 7.39 设置标签属性对话框

然后，弹出标签页属性设置对话框，用于设置页边距以及决定标签纸是连续页还是单页，如图 7.40 所示。

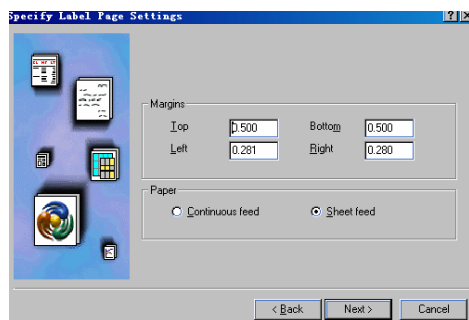


图 7.40 标签页的设置

7.3.8 N-up 显示风格

N-UP 格式以多列的形式显示数据。选择了【N-UP】格式后，需要指定显示的列数，在创建向导中会弹出分栏数目输入对话框，如图 7.41 所示，输入分栏即可。

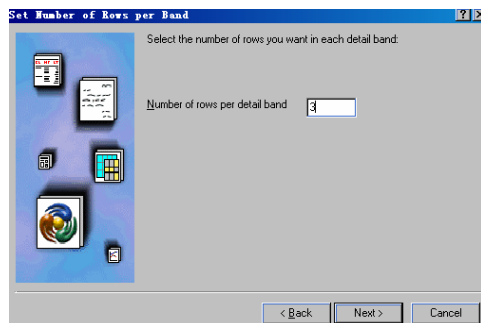


图 7.41 【N-Up】格式选择分栏数目

7.3.9 OLE2.0 显示风格

OLE2.0 格式能将数据源获得的数据与 OLE 服务器结合起来。

选择了【OLE2.0】格式后，会弹出【Choose Data Source for OLE DataWindow】对话框，如图 7.42 所示，进行数据源的设置。设置完数据源后，弹出【Specify OLE data】对话框，如图 7.43 所示，用途是确定用户有 OLE 数据窗口对象中使用的目标字段和用以分组的字段，从左边的【Source Data】列表框中将有关字段拖动到目标数据框【Target Data】中；如果需要指定分组，将分组字段拖动到分组数据框【Group by】中。单击【Next】按钮，弹出【Ready to Create OLE2.0 DataWindow】信息对话框，单击【Finish】按钮，弹出【Insert Object】对话框，如图 7.44 所示。该对话框有 3 个标签页，分别用来指定不同类型的 OLE 对象。其中，【Create New】标签页中有一个在 Windows 操作系统中注册过的服务器应用程序的列表框，选择其中一项，用来创建一个新的 OLE 对象；【Create From File】标签页用来指定一个已经存在的文件，创建 OLE 对象；【Insert Control】标签页是通过插入一个 OLE 控件来创建 OLE 对象。

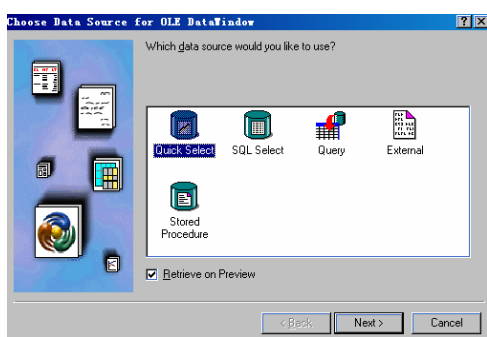


图 7.42 选择数据源对话框

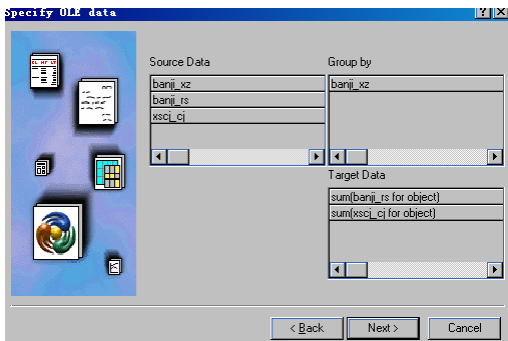


图 7.43 选择 OLE 数据对话框

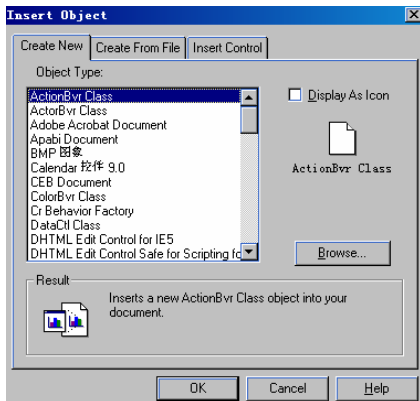


图 7.44 选择插入对象对话框

在数据窗口画板中，通过属性的编辑，可以对 OLE 数据窗口对象的名称、链接或嵌入方式、显示方式、激活方式以及链接激活方式等进行设置。

7.3.10 RichText 显示风格

选择了【RichText】格式后, 会弹出【Specify RichText Settings】对话框, 如图 7.45 所示。其中包括是否需要系统提供 Tool(工具栏)、Tab(栏)、Rule(标尺)、Header/Footer(题头和页脚)、Pop-up Menu(弹出菜单)以及 Display Only(只读属性)和 Background Color(背景颜色)等对 RichText 环境的设置。

单击【Next】按钮, 进入【Select Color and Border Settings】边界类型和文字、数据列颜色的设置对话框。再单击【Next】按钮, 弹出【Ready to Create RichText DataWindow】对话框, 在列表框中小结了新建 RichText 数据窗口的特性。单击【Finish】按钮, 进入数据窗口画板, 可以进一步对 RichText 数据窗口的属性进行具体的修改和设置。在数据窗口画板中, 单击鼠标右键, 会弹出【Rich Text Object】对话框, 如图 7.46 所示, 可以对 Rich Text 对象的属性进行修改。

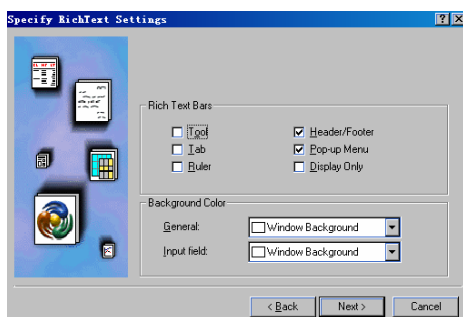


图 7.45 【Rich Text】设置对话框

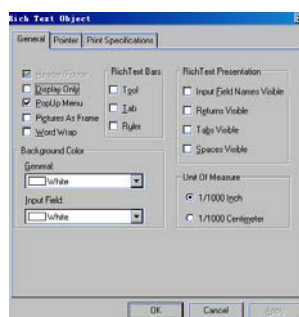


图 7.46 【Rich Text Object】对话框

7.3.11 Tabular 显示风格

Tabular 格式即列表格式, 它与 Grid 格式很相似。主要区别体现在 Tabular 格式没有网格线, 在运行时不可以改变列的宽度和位置, 但在设计时具有很大的灵活性, 可以随意交换和移动列的位置, 可以随意修改标题的内容, 还可以把几列放在一个标题下。这种显示风格在报表和数据输入中特别有用, 图 7.47 显示了一个 Tabular 风格的数据窗口对象。

学号	姓名	性别	出生日期
04011001	张力	男	85-5-10
04011002	孙丽丽	女	86-2-1
04011003	赵祥	男	85-12-10
04011004	王天力	男	86-3-15
04011005	张云莎	女	86-7-8
04011006	白银	男	85-10-5
04011007	李鹏飞	男	85-11-2

图 7.47 Tabular 显示风格

7.4 数据窗口画板

定义了数据源，选择了显示风格并根据需要提供了其他信息后，就进入了数据窗口画板。下面将介绍数据窗口画板的组成以及用途。

7.4.1 数据窗口的几个区域

数据窗口画板的外观如图 7.48 所示。实际见到的数据窗口画板与此并不一定相同，因为各个子窗口可以选择打开和关闭，位置也可以调整。数据窗口画板有 6 个子窗口，各个子窗口的名称和用途见表 7-1。

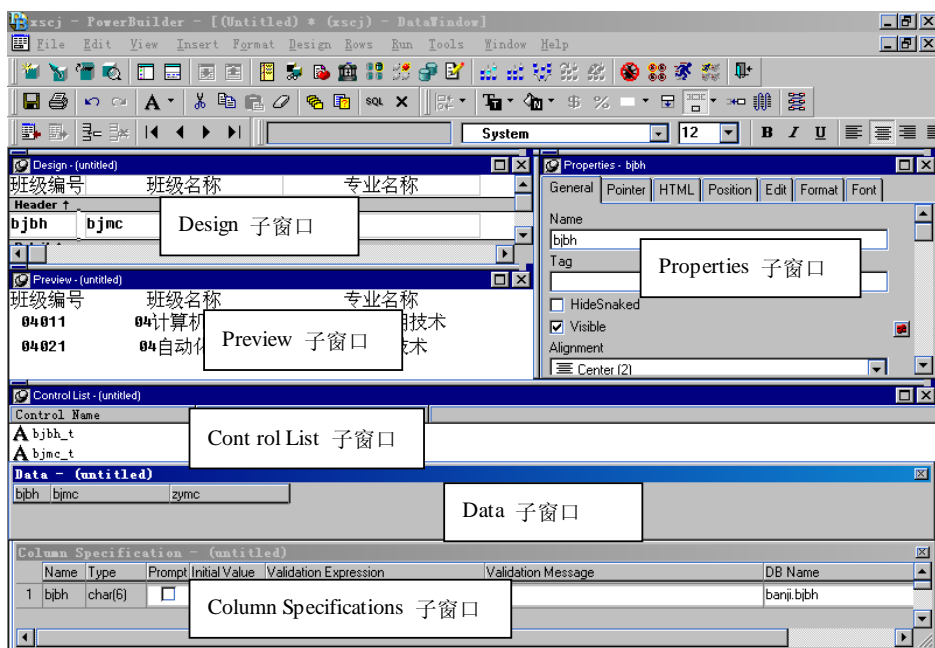


图 7.48 数据窗口画板的外观

表 7-1 数据窗口画板各子窗口的功能

子窗口的名称	功能与用途
Design 子窗口	用于调整和设计数据窗口的布局，并通过控件属性的调整来设置数据窗口的外观
Preview 子窗口	用于观察数据窗口在运行时的显示效果
Properties 子窗口	用于设置数据窗口对象或数据窗口对象中被选中的控件的属性
Control List 子窗口	显示数据窗口对象上的所有控件对象的列表，选中列表中的某个控件对象可以在 Design 子窗口中定位到该控件对象
Data 子窗口	显示数据窗口对象中检索到的数据，可以通过拖拉字段标题调整字段顺序
Column Specifications 子窗口	显示在数据源中选择的字段的列表，可以添加、修改或删除字段的初始值，也可以指定字段的检验规则以及检验提示信息，还可以通过拖拉字段来添加在数据源中定义的字段

在数据窗口画板中, 比较重要的是【Design】子窗口和【Properties】子窗口, 下面分别对这 6 个子窗口做进一步介绍。

1. 【Design】子窗口

Design 子窗口内有 6 个区域, 各个区域的名称和用途见表 7-2。

最常用的有【Header】(页眉/标题)、【Detail】(数据/细节)、【Summary】(汇总)和【Footer】(页脚)等 4 个区域, 如图 7.49 所示。其中标识带的标签旁有一个向上的箭头, 说明在标识带上方是相应的区域, 可以用鼠标拖动来改变这些区域的大小。

表 7-2 【Design】子窗口内的区域和用途

区域	位置	用途
页眉区	在 Header 带的上面, 一般在 Design 子窗口的最上部	显示字段标签或报表标题, 也可以添加修饰性对象, 例如文本对象、位图对象等
组标题区	在 Header 带和 Header Group 带之间, 只有 Group 样式或创建了组之后才会出现组标题区	主要用于分组报表, 使报表的条理清晰, 例如在报表中添加组标志符, 创建计算列, 显示分组的汇总信息等
细节区	在 Header 带和 Detail 带之间	用于显示检索数据的结果集, 可以对字段的位置、尺寸进行调整
组结尾区	在 Detail 带和 Trailer Group 带之间, 与组标题区对应	用于显示一个分组结束时关于该分组的统计计算和汇总信息
汇总区	在 Trailer Group 带或 Detail 带和 Summary 带之间, 出现在所有检索出的数据的最后	用于显示所有数据的汇总信息, 例如, 计算显示记录的总数, 满足一定条件的某字段和汇总值或显示备注信息
页脚区	在 Summary 带和 Footer 带之间, 一般在 Design 子窗口的最下部	用于显示页码、总页数或脚注等信息

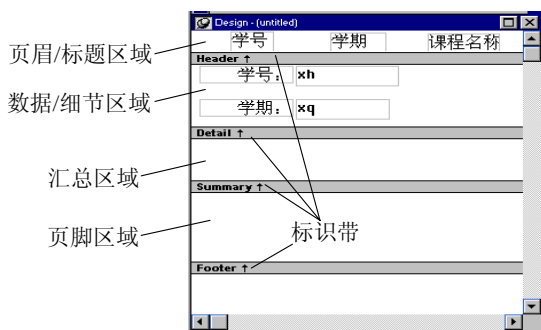


图 7.49 【Design】子窗口的四个区域

2. 【Properties】子窗口

【Properties】窗口用来设置【Design】窗口中各部分或数据窗口对象各部分的属性。根据在【Design】窗口中选项中的部分, Properties 窗口中显示相应的属性内容, 包括不同的属性选项卡。在【Properties】窗口中对属性值的改变, 可以直接反映在【Design】窗口中。

数据窗口对象的属性标签页共有 5 页, 其中【HTML Table】标签页和【HTML Generation】

标签页用于生成数据窗口的【HTML】窗体，需要时可以参考有关书籍。这里重点介绍其余 3 个属性页。

数据窗口对象的【General】标签页如图 7.50 所示，它用于指定数据窗口对象使用的计量单位、内部定时器的时间间隔、背景颜色和是否生成【HTML】窗体。其中，数据窗口对象属性使用的计量单位有 4 种选择，见表 7-3。一般可以使用缺省的选择 PowerBuilder(0)，即使用 PBU 单位，其优点是用它设计出的应用程序在不同的监视器(EGA、VGA、SVGA 等)和不同的平台上运行时外观保持一致。

数据窗口对象的【Pointer】标签页如图 7.51 所示，该页用于指定光标在数据窗口内时的图形。单击【Pointer】下拉列表框右边的▼按钮，可以选择系统提供的光标图形，也可以单击旁边的【...】按钮，选择其他光标图形。

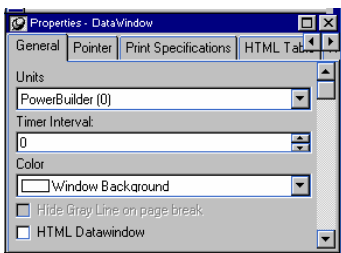


图 7.50 数据窗口对象的【General】标签页

表 7-3 数据窗口对象的计量单位

Units 属性	计量单位
PowerBuilder(0)	PowerBuilder 的单位 PBU
Pixels(1)	像素单位
1/1000 Inch(2)	千分之一英寸
1/1000 Centimeter(3)	千分之一厘米

数据窗口对象的【Print Specifications】标签页如图 7.52 所示，该页用于设置数据窗口对象的打印参数，各参数的含义见表 7-4。

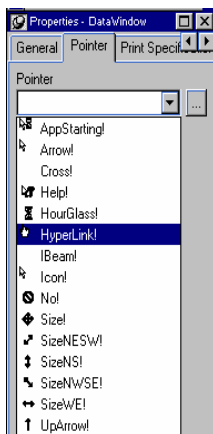


图 7.51 数据窗口对象的【Pointer】标签页

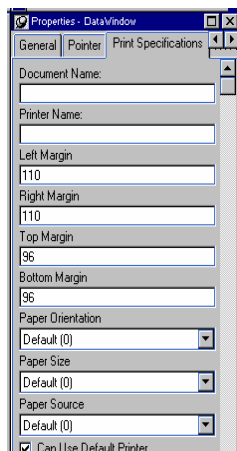


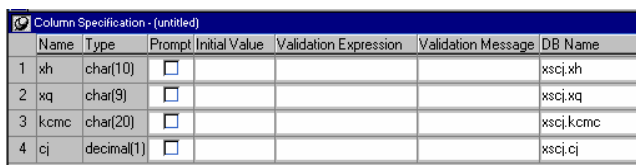
图 7.52 【Print Specifications】标签页

表 7-4 数据窗口对象的打印参数

打印参数	含义
Document Name	打印数据窗口对象时在打印队列中显示的文档名称
Left Margin, Right Margin Top Margin, Bottom Margin	分别为打印时在左边、右边、上边和下边留出的空隙长度
Paper Orientation	选择打印方向
Paper Size	指定打印纸的大小
Paper Source	指定打印时的送纸方式
Prompt before printing	在打印输出前是否显示打印设置对话框
Display Button-Print Preview	在预览时显示数据窗口对象上的按钮对象
Display Button-Print	打印数据窗口对象上的按钮对象
Newspaper Columns Across	指定每页打印的列数
Newspaper Columns Width	指定每列的宽度

3. 【Column Specification】子窗口

【Column Specification】子窗口用来对数据窗口对象中的数据列进行说明，可以说明初始值、合法性校验规则和有关提示信息等，如图 7.53 所示为【Column Specification】子窗口。



	Name	Type	Prompt	Initial Value	Validation Expression	Validation Message	DB Name
1	xh	char(10)	<input type="checkbox"/>				xscj.xh
2	xq	char(9)	<input type="checkbox"/>				xscj.xq
3	kcmc	char(20)	<input type="checkbox"/>				xscj.kcmc
4	cj	decimal(1)	<input type="checkbox"/>				xscj.cj

图 7.53 数据窗口画板的【Column Specification】子窗口

4. 【Control List】子窗口

【Control List】子窗口中给出了数据窗口对象中各种控件的列表清单，通过该窗口可以快速选择【Design】窗口中的控件，有些初学者会经常遇到这样的问题。由于把某个控件的【Visible】属性设置为 False 后，在【Design】窗口中就无法选中该控件，这时就可以通过该【Control List】窗口将其选中，并在【Properties】窗口中将其【Visible】属性设置为 True，如图 7.54 所示为【Control List】子窗口。

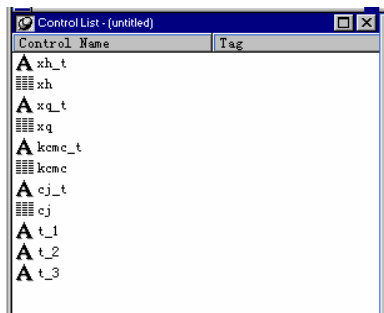


图 7.54 数据窗口画板的【Control List】子窗口

5. 【Preview】子窗口

【Preview】窗口可以在设计数据窗口对象时，随时观察数据窗口对象运行时的效果，以便在【Design】窗口中对其进行调整。在【Preview】窗口中还可以对数据库表进行插入、删除等操作，如图 7.55 为【Preview】子窗口。

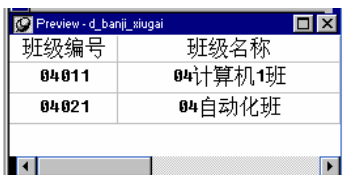


图 7.55 数据库画板的【Preview】子窗口

6. 【Data】子窗口

【Data】子窗口用来显示通过数据窗口对象检索到的数据，其作用和【Preview】窗口非常相似。

7.4.2 设置数据窗口对象的属性

在建立数据窗口对象时，选择数据窗口的显示风格和数据源后，PowerBuilder 就会自动生成一个具有默认属性和特征的数据窗口对象，接着在数据窗口画板中可以进行下一步的设计。

数据窗口对象有很多属性，在设计修改一个数据窗口对象时，可以在画板中设置它的一些属性，以增强它的功能、满足应用的需要。

如果在【Design】视图中没有选中任何一个对象，那么在【Properties】视图中显示的是当前数据窗口对象的属性，如图 7.56 所示。

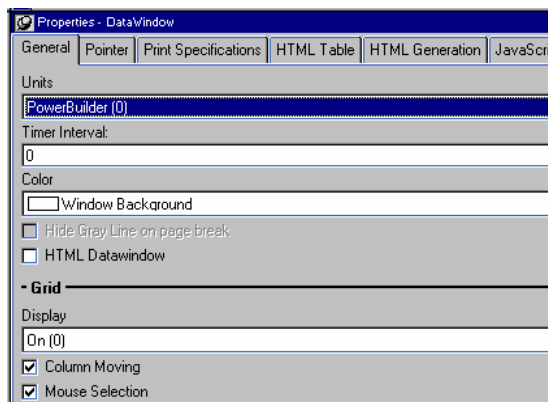


图 7.56 数据窗口对象的属性视图

1. 【General】标签页

【General】标签页用来设置数据窗口对象所用的测量单位、计时器时间间隔和背景颜色等内容。不同显示风格的数据窗口对象的【General】标签页的内容稍有一些差别。

1) 【Units】列表框

【Units】下拉列表框是用来指定数据窗口对象的测量单位,在该列表框中有以下4个选项。

- 【PowerBuilder(0)】选项:表示使用的是 PowerBuilder 单位(PBU)。
- 【Pixels(1)】选项:使用像素为单位。
- 【1/1 000 Inch(2)】选项:使用的测量单位是 1/1 000 英寸。
- 【1/1 000 Centimeter(3)】选项:使用的测量单位是 1/1 000 厘米。

如果开发人员希望在程序运行时打印数据窗口对象的内容,那么选择英寸或者厘米作为测量单位,可以比较容易地定义边距尺寸。

2) 【Timer Interval】编辑框

【Timer Interval】编辑框是用来指定该数据窗口对象内部定时触发事件的间隔(以毫秒为单位)。只有在该数据窗口对象中使用了时间域(如在数据窗口的 Header 区域显示时间),该属性值才有效,它决定了 PowerBuilder 系统更新时间域的时间间隔。例如,如果该值设置为 1 000,那么 PowerBuilder 每隔一秒钟更新一次时间域。

3) 【Color】列表框

【Color】下拉列表用来指定该数据窗口对象的背景颜色,用户可以从下拉列表中选择所需要的颜色。

4) 【Grid】窗口

对于 Grid 显示风格的数据窗口对象,在【General】标签页中还会多出一些设置网格的属性。【Display】下拉列表框是用来指定网格的显示时机,它有以下选项。

- 【On(0)】选项:总是显示网格。
- 【Off(1)】选项:总是不显示网格。
- 【Display Only(2)】选项:仅在显示时显示网格。
- 【Print Only(3)】选项:仅在打印时显示网格。
- 【Column Moving】复选框:用来决定列是否可以在运行时被移动。
- 【Mouse Selection】复选框:用来决定运行时是否可以用鼠标选中列。
- 【Row Resize】复选框:用来决定运行时是否允许改变行的尺寸。

2. 【Pointer】标签页

根据数据窗口对象中各个不同区域的不同用途,可以将鼠标设置为不同的形状,这个属性是在【Pointer】标签页中设置的。

【Pointer】标签页中只有一个【Pointer】下拉列表框,可以从该下拉列表框中选择一个系统预定义的鼠标指针类型,也可以单击该列表框右边的【Browse】按钮,从其他指针文件(CUR 文件)中定义鼠标的形状,这时需要指定鼠标指针文件的名称。

3. 【Print Specifications】标签页

在 PowerBuilder 中,数据窗口对象中的内容还可以直接被打印出来,在数据窗口对象的属性视图的【Print Specifications】标签页中可以设置一些与打印有关的属性。

- 【Document Name】编辑框:在该编辑框中指定文件名,用于在打印队列中标识打印任务或打印的报表。

- **【Printer Name】** 列表框：使用的打印机名称。
- **【Left Margin】** (左边距)、**【Right Margin】** (右边距)、**【Top Margin】** (顶部边距)、**【Bottom Margin】** (底部边距)列表框：按指定的度量单位定义页边距。
- **【Paper Orientation】** 列表框：指定打印方向，即横向打印或纵向打印。
- **【Paper Size】** 列表框：指定打印纸的尺寸或规格。
- **【Paper Source】** 列表框：指定打印纸来源，是自动送纸还是手工送纸。
- **【Prompt Before Printing】** 列表框：指定打印之前是否给出提示信息。
- **【Can Use Default Printer】**：该复选框用来确定当指定的打印机名称无效时，是否默认使用系统打印机。
- **【Display Buttons-Print】** 复选框和 **【Display Buttons-Print Preview】** 复选框：这两个复选框分别用来设置是否显示 **【Print】** 按钮和 **【Print Preview】** 按钮。
- **【Newspaper Columns Across】** 编辑框和 **【Newspaper Columns Width】** 编辑框用来设置数据窗口对象的 **【Newspaper】** 列的属性。其中 **【Newspaper Columns Across】** 编辑框用来指定在一行内可以排几列，**【Newspaper Columns Width】** 编辑框用来指定列的宽度。

4. 【HTML Table】标签页

当把数据窗口转换成 HTML 表格时，该标签页用于设置 HTML 表格的属性。包括表格的边框、单元格时间的空隙、单元格填充字符、宽度等。

5. 【HTML Generation】标签页

该标签页主要用来设置转换后的 HTML 文件的格式，设置的属性包括每页的行数、浏览器的型号、HTML 语言的版本、对象的名称、自连接以及自链接的参数信息。

7.4.3 向数据窗口中添加对象

构成数据窗口对象的每一个元素都是一个独立的对象。由 PowerBuilder 初始生成的数据窗口(如 Grid、Tabular 风格等)一般包括两类对象：一类是可以看作静态文本控件的标签或栏目名；另一类是可以看作编辑框控件的数据列。在数据窗口画板的 **【Control List】** 窗口中可以看到这些对象的列表，在数据窗口画板中可以管理修饰这些对象，如果需要还可以插入新的对象。

在数据窗口中添加对象的方法是：从 **【Insert】|【Control】** 菜单项中选择要插入的控件名称，然后在数据窗口的适当区域中单击鼠标，这样该控件就被放置到数据窗口中，最后对新添加的控件进行设计和设置属性等。下面介绍可以在数据窗口对象中插入的各种控件和作用及其设计方法等。

1. 添加文本

用户可以在数据窗口的任何区域中添加文本，作为提示性说明。

向数据窗口对象中添加文本的步骤如下。

- (1) 选择 **【Insert】|【Control】|【Text】** 菜单。
- (2) 单击数据窗口中要放置文本的地方，就自动在鼠标单击的地方添加了一个静态文

本控件。

- (3) 输入文本信息。
- (4) 使用【StyleBar】菜单中的工具为文本设置字体、大小、风格以及对齐方式等属性。

2. 添加命令按钮

在数据窗口对象中还可以添加一些命令按钮来完成一些特殊的命令功能。数据窗口中的命令按钮不同于一般窗口中的命令按钮控件，它一般有固定的功能，所以不需要用户为命令按钮编写程序。

在数据窗口对象中添加命令按钮的方法如下。

- (1) 选择【Insert】|【Control】|【Button】命令。
- (2) 单击数据窗口中要放置文本的地方，就自动在鼠标单击的地方添加了一个命令按钮。
- (3) 在【Properties】窗口的【General】标签页中的【Text】编辑框中，输入按钮的提示文本。
- (4) 从【Action】下拉列表框中为按钮指定功能。
 - 【Retrieve】按钮：从数据库中检索数据。
 - 【Page Next】按钮：翻到下一页。
 - 【Page Prior】按钮：翻到前一页。
 - 【Sort】按钮：显示【Sort】对话框并指定排序的列，然后数据窗口按指定的方式排序。
 - 【Filter】按钮：显示【Filter】对话框并指定过滤条件，然后数据窗口将过滤不满足条件的行。
 - 【Delete Row】按钮：从数据窗口中删除当前行。
 - 【Append Row】按钮：在数据窗口尾部添加一新行。
 - 【Insert Row】按钮：在数据窗口中插入一新行。
 - 【Update】按钮：将修改的内容写到数据库中。
 - 【SaveRowsAs】按钮：显示【Save As】对话框，将记录另存为指定的格式。
 - 【Print】按钮：打印数据窗口对象。
- (5) 如果希望使用图片命令按钮，可以选中【Active Default Picture】复选框，为按钮指定默认的图标，也可以通过【Picture File】编辑框指定图片文件。

3. 添加新列

当开发人员发现某些数据库中的列没有被包含在数据窗口对象中时，可以向数据窗口对象中添加新的列。分两种情况：一种是在数据窗口的数据源中已包含了这些列，但在数据窗口对象中不包括这些列；另外一种情况是在数据窗口的数据源中就不包含这些列。

针对第一种情况添加新的列方法如下。

- (1) 选择【Insert】|【Control】|【Column】命令。
 - (2) 单击要放置该列的地方，弹出【Select Column】对话框，显示出数据窗口对象数据源的所有列。
 - (3) 选择要添加的列，然后单击【OK】按钮。
- 而针对第二种情况，则需要修改数据窗口的数据源，方法是单击【Painter Bar】菜单

上的【DataSource】按钮,或选择【Design】|【Data Source】命令,打开【SQL Select】画板,然后在图形界面添加新的字段,最后单击【Return】按钮,返回到数据窗口画板。

4. 添加图片

有时还可以向数据窗口对象中添加图片来美化数据窗口对象的外观。如果图片添加到数据窗口对象的【Header】、【Summary】或【Footer】区域中,那么程序运行时只显示一次图片;如果将图片添加到【Detail】区域,那么会在【Detail】区域的每行都显示图片。

在数据窗口对象中添加图片的方法如下。

(1) 选择【Insert】|【Control】|【Picture】命令。

(2) 在要显示图片的地方单击鼠标后,弹出【Select Picture】对话框。

(3) 使用【Browse】按钮选择要添加的图片文件名称,或者直接在【File Name】编辑框内输入文件名(文件类型可以是 BMP、JPG、JIF、RLE 或 WMF 类型),然后单击【Open】按钮。

(4) 如果要求按原始大小显示图片,则可以右击新插入的图片,然后从快捷菜单中选择【Original Size】菜单。

(5) 在【Properties】窗口中设置图片的其他属性。

5. 添加绘图控件

为了装饰数据窗口,还可以向数据窗口对象中添加以下绘图控件。

(1) 矩形(Rectangle)。

(2) 圆角矩形(Round Rectangle)。

(3) 线(Line)。

(4) 椭圆(Oval)。

这些控件插入到数据窗口对象后,可以改变它们的位置、大小,可以改变填充颜色和线条宽度等。

在数据窗口对象中除了添加以上 5 种控件外,还可以添加计算列或计算域,这也是数据窗口对象中非常重要的一项内容。

6. 添加计算域

有时候用户需要在报表中进行数据统计(如计算每个学生的平均成绩等)、显示系统日期、时间、显示页数等,可以利用计算域这个对象。在数据窗口对象中添加一个计算域的方法如下。

(1) 选择【Insert】|【Control】|【Computed Field】菜单命令,或者单击工具栏上下拉按钮中的【Computed Field】按钮。

(2) 在数据窗口中要放置计算域的地方单击鼠标,这时会弹出【Modify Expression】对话框,如图 7.57 所示。

(3) 通过选择【Functions】列表框中的系统函数、【Columns】列表框中的数据列以及该对话框左下方的运算符按钮,在【Expression】编辑框输入计算域的表达式。

(4) 单击【Verify】按钮对输入的表达式进行合法性检查。

(5) 最后单击【OK】按钮即可。

下面介绍几种常用的计算域的添加方法。

1) 统计数据

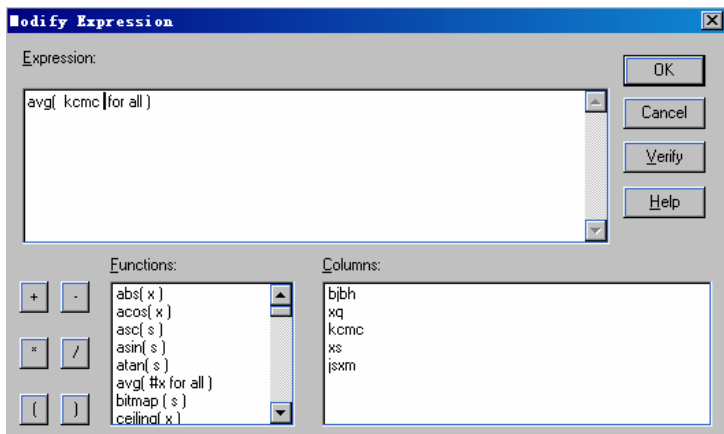


图 7.57 输入计算域的表达式

在【Summary】区域中，利用系统提供的汇总对象可以很方便地实现数据的汇总。在数据窗口对象中，可以进行的汇总统计如下。

- (1) 求平均值(Average)。
- (2) 计数(Count)。
- (3) 求和(Sum)。

要添加一个汇总计算域的方法如下。

- (1) 数据窗口对象中选择要进行汇总的列。
- (2) 选择【Insert】|【Control】|【Average】(或【Count】、【Sum】)命令，PowerBuilder 将自动添加一个计算域到数据窗口对象中去。

2) 添加页码

可以在数据窗口的标题区域或页脚区域添加页码，添加一个页码计算域的方法如下。

- (1) 选择【Insert】|【Control】|【Page n of n】命令。
- (2) 在要添加页码的区域单击鼠标，就添加了一个计算域，此计算域的默认表达式为：

```
'page'+page()+ ' of '+pagecount()
```

- (3) 用户可以将这个计算域的表达式改成自己习惯的形式，例如，可以在【General】标签页的【Compute Expression】编辑框中输入下面的表达式：

```
'第'+page()+ '页,共'+pagecount()+ '页'
```

- (4) 当地调整计算域的大小，以完全显示页码。

3) 添加日期

和添加页码一样，可以在数据窗口对象的标题区域或页脚区域中添加日期。一般如果数据窗口用于打印的话，习惯在标题区域中添加一个日期计算域。添加日期计算域的方法如下。

- (1) 选择【Insert】|【Control】|【Today】命令。

(2) 在要添加日期的区域单击鼠标，添加一个日期计算域。

(3) 如果用户要修改日期的显示格式，可以在【Format】标签页的【Format】编辑框中输入用户习惯的格式，例如输入如下格式：

yyyy 年 m 月 d 日

(4) 适当调整计算域的大小，以便能完全显示日期。

7.4.4 设置数据窗口中对象的属性

数据窗口中对象的属性包括文本(标题、选项卡等)和数据对象(包括列、计算域、汇总信息等)等的属性。

1. 文本框的属性

文本框的属性窗口如图 7.58 所示。

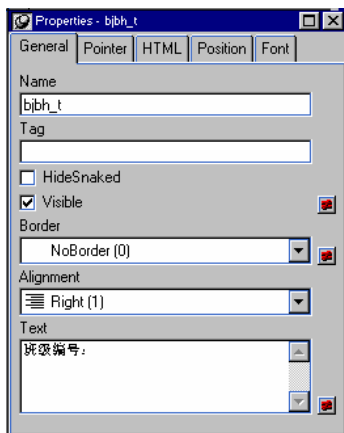


图 7.58 数据窗口对象中文本框的属性窗口

- 【General】标签页主要用来设置文本框的名字(Name)、标记(Tag)、边框的格式(Border)、对齐方式(Alignment)和显示的文本(Text)等。
- 【Pointer】标签页用来设置当鼠标光标位于该文本框上方时的光标形状。
- 【HTML】标签页用于设置超链接情况，如链接目标和链接参数等。
- 【Position】标签页用于设置文本框的位置信息，如坐标、宽度以及能否移动、重置大小等信息。
- 【Font】标签页用于设置文本框的字体属性，如字体的大小、颜色、背景色等。

2. 数据对象的属性

选中一个数据对象，如数据窗口中的某一列，打开它的属性对话框。数据对象由于直接与数据库中的数据相联系，在数据对象的属性窗口中增加了两个与数据格式有关的标签页：【Format】标签页和【Edit】标签页，详细设置参照 7.6 节的介绍。其他选项卡和文本框的标签页相同，此处不再赘述。

3. 设置数据窗口中对象的 Tab 顺序

当 PowerBuilder 生成一个数据窗口对象时, 为所有列分配一个 Tab 值, 这些 Tab 值决定了在程序运行期间使用【Tab】键在数据窗口中各个对象之间进行切换的顺序。各个对象的默认 Tab 值是按对象从左到右、从上到下的顺序以 10 为单位依次递增的。合法的【Tab】值为 0~9 999 之间的任意一个整数。Tab 值为 0 表示该对象不能通过 Tab 键获得光标。每个对象的【Tab】值的大小决定了按 Tab 键时获得光标的顺序, Tab 值越小的对象越先得到光标。

注意: (1) 在修改 Tab 值时, 可以看到红色的 Tab 值只出现在结果集中的字段上方, 文本对象、图形对象、计算字段等都没有 Tab 值。

(2) 在窗口画板中, 如果某个控件的 Tab 值为 0, 表示使用 Tab 值跳过该控件, 但是用鼠标依然能够选中该控件。而在数据窗口对象中, 如果某个对象的 Tab 值为 0, 即使用户使用鼠标也无法选中这个对象, 使用这种方法可以完全禁止某一行, 用来保护此列的内容不被修改。

(3) 数据窗口对象中包含了多个表, 那么所有列的 Tab 值都默认为 0, 这是由于在数据窗口中一次只能修改一个表中的数据。

4. 设置列的边框

PowerBuilder 提供了 7 种边框形式, 通过给数据窗口对象指定各种边框, 增强它们的显示效果。要给数据窗口对象中的某一个对象指定边框形式, 先选中该对象, 然后在该对象的属性视图中的【General】标签页的【Border】下拉列表框中指定, 还可以利用工具栏上的按钮来设置, 如图 7.59 所示。如果用户选中【ResizeBorder】选项形式, 那么在程序运行过程中, 可以改变对象的边框大小。

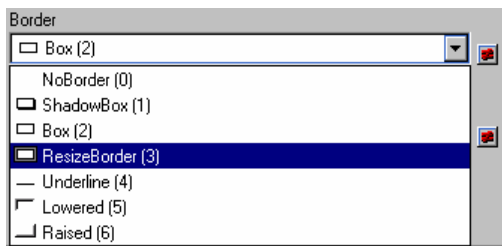


图 7.59 数据窗口对象中边框属性

7.5 预览数据窗口

创建了数据窗口, 用户总是希望在 PowerBuilder 中预览数据窗口。使用数据窗口对象的 Preview(预览)窗口可以在设计时预先浏览数据窗口对象运行的效果, 并可以通过该窗口完成对数据库的一些数据操作。

7.5.1 预览数据窗口

【Preview】窗口是在数据窗口画板中默认打开的,如果在当前画板中看不到【Preview】窗口,可以通过选择菜单【View】|【Preview】命令打开 Preview 窗口。在打开【Preview】窗口时,PowerBuilder 自动完成数据的检索(如果有参数,系统会提示用户输入参数值)。如果数据窗口对象使用了外部数据源,那么预览数据窗口将不检索数据,这时可以选择【Rows】|【Import】菜单,从外部文件向数据窗口对象导入数据。

在数据窗口画板中有 3 个【PainterBar】按钮,其中一个用来进行【Preview】窗口的操作。如图 7.60 所示,各图标的功能从左到右依次是:

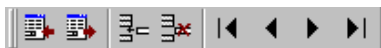


图 7.60 【Preview】窗口的【PainterBar】图标

- 【Retrieve】图标: 从数据库中重新检索数据。
- 【Save Changes】图标: 保存对数据库所做的修改。
- 【Insert Row】图标: 在数据窗口中插入一行。
- 【Delete Row】图标: 删除数据窗口中的当前行。
- 【First】图标: 指针移动到第一条记录。
- 【Prior】图标: 指针移动到前一条记录。
- 【Next】图标: 指针移动到下一条记录。
- 【Last】图标: 指针移动到最后一条记录。

7.5.2 数据排序

在定义 SQL Select 数据源时,SELECT 语句中可以事先定义好排序语句,当数据被取到数据窗口对象时,它们是已排好序的。然而,有时需要在数据窗口中实现排序功能,这就要重新定义。

对数据窗口中的数据进行排序的步骤如下。

(1) 从菜单中选择【Rows】|【Sort】命令,则出现如图 7.61 所示的对话框。

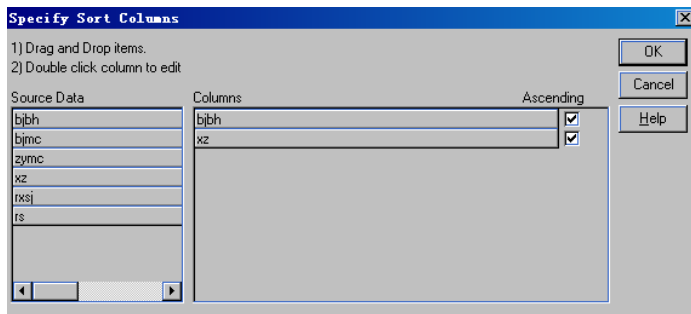


图 7.61 【Specify Sort Columns】对话框

(2) 把要排序的列从【Source Data】所属的框拖到【Columns】所属的框。如果要排序的列的【Ascending】标题下的复选框带有“√”,表明这个列是按升序排列的,反之就

是降序排列。

(3) 需要说明的一点是【Columns】框中显示的列顺序决定了排序的先后顺序。例如：在上面的对话框中，行首先按“bjbh”这一列排序，对“bjbh”列相同的行，再按“xz”排序。如果要改变排序的顺序，只需选中某列，上下拖动即可。

(4) 可以按指定的表达式排序。例如：对于一列数据类型是 Number 型的 NO 列和 NUM 列，可以用 NO+NUM 来排序。如果要定义表达式，可以双击【Columns】框中的某列，此时将弹出如图 7.62 所示的对话框，在对话框中定义表达式，并单击【OK】按钮，就完成了排序定义。

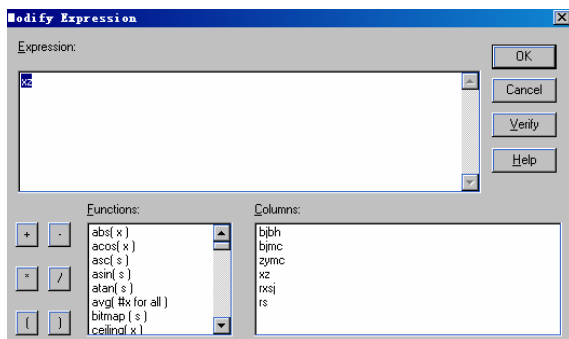


图 7.62 【Modify Expression】对话框

7.5.3 数据过滤

在创建数据窗口过程中定义数据源时，通过 Where、Having 子句以及检索参数可以达到过滤数据库中数据的目的。在数据窗口画板中，还可以进一步进行过滤，并可以通过改变过滤条件，实现灵活多变的数据窗口对象。

在数据窗口对象画板中设置过滤条件的方法如下。

(1) 选择【RowS】|【Filter】命令，将出现如图 7.63 所示的对话框。

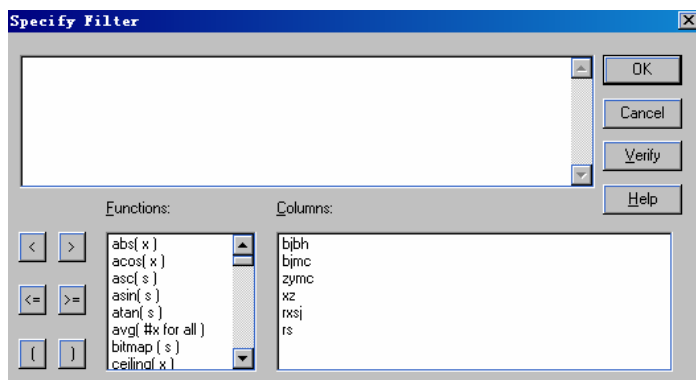


图 7.63 Specify Filter 对话框

(2) 输入一个布尔型过滤表达式，PowerBuilder 将对每行进行测试。如果表达式值为 TRUE，该行将被显示，可以在表达式中粘贴 PowerScript 函数、列和运算符。

(3) 单击【OK】按钮。则 PowerBuilder 过滤数据，只显示出满足过滤准则的行。

要删除一个过滤准则，可以从图 7.63 中选中要删除的过滤表达式，按【OK】按钮删除就可以了。

7.5.4 数据的分组

在应用中经常需要按某一列或几列的值对数据进行分组统计，在数据窗口对象中实现这种分组统计非常地方便。

一般定义分组有两种方法。第一是建立一个 Group 显示风格的数据窗口对象，第二是对一个现有的 Tabular 风格的数据窗口对象定义分组属性。在前面我们已经简述了 Group 显示风格的数据窗口对象的创建，所以在这里只讲述第 2 种方法。

为某个已经创建的数据窗口对象定义分组属性的方法如下。

(1) 选中并打开要分组的数据窗口对象。

(2) 在数据窗口画板中，选择【Rows】|【Create Group】命令，这时进入了【Specify Group Columns】对话框，如图 7.64 所示。

(3) 分组所依据的列从左边的列表框中拖入到右边的列表框中，用户也可以定义分组的表达式。方法是双击要分组的列的列名，弹出【Modify Expression】对话框后，在其中定义表达式即可。

(4) 如果用户要对组内的数据再进行分组，可以定义子分组。方法是选择【Rows】|【Create Group】菜单项，定义子分组表达式。

(5) 定义完分组之后，单击【OK】按钮，这时就会在数据窗口对象增加两个新的区域：Group Header(组标题区)和 Group Tailer(组尾区)。在这两个区域中，可以添加任何对象，如文本，计算域和按钮等。

(6) 组风格数据窗口对象的一个优点就是能够统计每组记录的信息，这也是用户创建分组数据窗口的一个目的，为此，可以分别为每一组增加计算域。增加计算域的方法跟前面讲的一样。

(7) 在有些情况下，需要按照某个表达式来排序，这时就需要设置组标题区的 Group Sort 属性了。方法是将鼠标指针移动到标题区的标题带上，此时鼠标指针变为双箭头，单击鼠标右键，在弹出的菜单中选择【Properties】属性，此时就进入了【Properties】视图(见图 7.65)。然后在【Properties】视图的【General】标签页选择【Group Sort】编辑框，用户

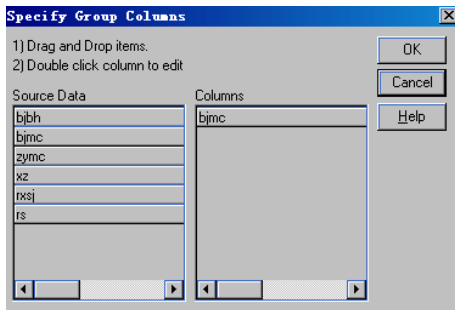


图 7.64 【Specify Group Columns】对话框

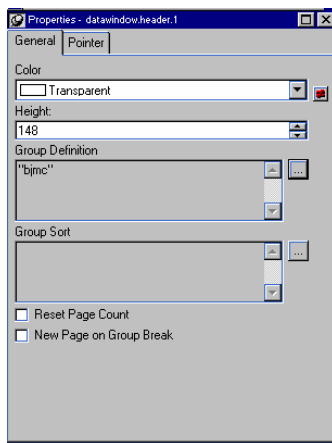


图 7.65 组标题区的属性视图

可以看到这个编辑框是灰色的，表明直接不能在这个框架中编辑表达式，如果要编辑表达式，需要单击右边按钮【…】，这时会弹出一个【Specify Sort Columns】对话框，在这个对话框中定义排序即可。

7.5.5 去掉重复值

当用户采用 Tabular 选项或者是 Grid 选项显示风格时，有可能在数据窗口的字段中出现重复的数据，那么，当用户看到这些重复的数据时，可能觉得有些冗余。例如：在班级基本信息的数据窗口中，学生所属的班级编号、专业名称就有可能出现重复，如图 7.66 所示。



班级编号	专业名称	学号	姓名
04011	计算机应用技术	04011013	何庆
04011	计算机应用技术	04011014	王凯
04011	计算机应用技术	04011015	赵飞
04011	计算机应用技术	04011016	赵童
04011	计算机应用技术	04011017	苗清华
04011	计算机应用技术	04011018	王北方
04011	计算机应用技术	04011019	查贺年
04011	计算机应用技术	04011020	白丽萍
04011	计算机应用技术	04011021	孙强
04011	计算机应用技术	04011030	毛凡玺
04011	计算机应用技术	04011022	朱大鹏
04011	计算机应用技术	04011023	王发

图 7.66 窗口显示重复数值

从图中可以看到，班级编号、专业名称重复显示在多个记录中。为了美观，可以把相同的值屏蔽掉，让数据窗口对象只显示该列中头一次出现的那个值。

在数据窗口中禁止重复数据的步骤如下。

(1) 在数据窗口画板中，选择【Rows】|【Suppress Repeating Values】命令，这时会打开【Specify Repeating Value Suppression List】对话框，如图 7.67 所示。

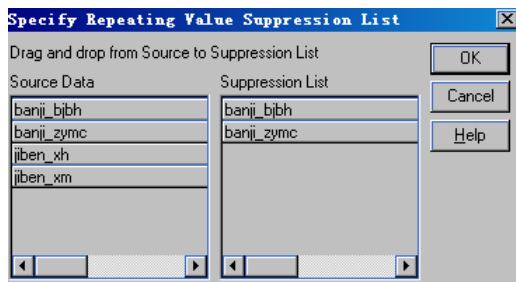


图 7.67 屏蔽重复值的对话框

(2) 该对话框中左边的列表框显示了当前数据窗口对象所选择的所有列，右边的列表框显示的是当前屏蔽的所有列。将要屏蔽重复值的列从【Source Data】列表框架中拖入到【Suppression List】列表框架中，如图 7.68，屏蔽了班级编号“banji_bjbh”和专业名称“banji_zymc”两列。

(3) 重复步骤(2)直到所有的要屏蔽的列都拖入到右边的列表框架中了，单击【OK】按钮，返回到数据窗口画板中即可。

图 7.68 显示了屏蔽班级编号和专业名称两列的数据窗口对象的预览情况,可以看到此时数据窗口对象明显要比没有屏蔽重复值的数据窗口对象美观一些了。



班级编号	专业名称	学号	姓名
04011	计算机应用技术	04011013	何庆
		04011014	王凯
		04011015	赵飞
		04011016	赵章
		04011017	苗清华
		04011018	王北方
		04011019	查贺年
		04011020	白丽萍

图 7.68 屏蔽重复值的数据窗口对象

7.6 数据的显示格式

当预览 DataWindow 对象时,有一些列具有不同的格式(例如:下拉列表框、单选按钮、编辑屏蔽)。除了以不同的方式显示数据外,当输入不正确的数据时,某些列还能显示出错错误消息。

下面介绍如何使用已有的显示格式、编辑风格和有效性规则,如何改变它们以及如何给它们创建新的形式。

7.6.1 列的显示格式

要使用显示格式,可打开【Column Object】属性表,然后选择【Format】标签页面,单击后面的小三角,就会打开一个对话框。在属性对话框中选择【Format】选项卡,如图 7.69 所示。

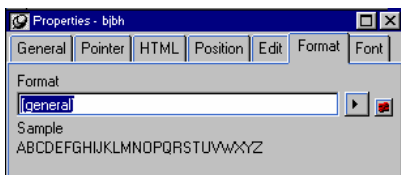



图 7.69 【Format】选项卡

在【Format】编辑框中输入格式掩码,或单击  按钮,显示当前所有可用的格式的菜单,从中选择所需要的格式即可。在【Sample】标题下面的静态文本框中将显示当前的格式示例。当用户改变【Format】编辑框中的格式后,【Sample】标题下面的格式示例会随之改变。

如果此列的数据类型的显示格式已经存在(例如:String),它们便会显示在对话框底部的列表框中。要应用一个已存在的格式,先单击此格式名,然后再单击【OK】按钮。如果用户想看看一个值在所提供的显示格式下会是什么样子,可以在【Test Value】单行编辑框中输入一个正确的数据类型值,然后再单击【Test】按钮。

7.6.2 列的编辑风格

与显示格式一样,编辑风格也能改变数据在用户面前显现的方式,但和显示风格不同的是,当列获得焦点时,编辑样式并不会消失。编辑风格影响了用户与数据交互作用的方式。要选择一个编辑风格,打开【Column Object】属性表,然后单击【Edit】标签页面。在该标签页中有一个【Style Type】下拉列表框,有6个样式可供选择:【Edit】(编辑框)、【CheckBox】(复选框)、【DropDownDW】(下拉数据窗口)、【DropDownListBox】(下拉列表框)、【EditMask】(编辑屏蔽框)和【RadioButtons】(单选按钮),如图7.70所示。

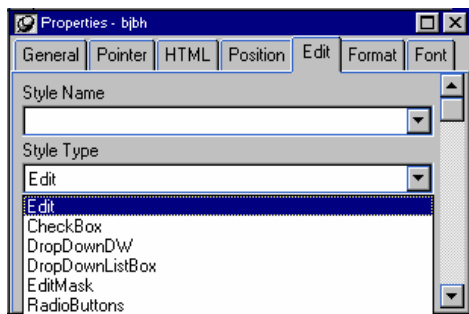


图 7.70 【Style Type】下拉列表框

1. 【Edit】(编辑框)

编辑框是默认的样式,它的功能同窗口画板中的单行或多行编辑框一样,在前面已经介绍过。下面介绍选择其他样式的【Edit】选项卡的设置方法。

2. 【CheckBox】(复选框)

如果某一列的值可以从几个状态中进行选择,则应该考虑使用【CheckBox】复选框。

【CheckBox】复选框属性如图7.70所示。

在【CheckBox】样式中主要有下列属性需要设置:【3D Look】用来设置复选框是否有3D效果;【Text】编辑框用来输入显示在复选框后面的文本,如果要定义加速键,只需在表示该加速键字母的前面输入一个“&”号。【Data Value for On(Off)】编辑框用来设置在复选框中(不选中)时该数据列的值。当选中【3 States】复选框时,在图7.71中出现最后一行【Other State】编辑框,它用来表示区别于选中(不选中)的第三种状态的值。

3. 【DropDownDW】(下拉数据窗口)

【DropDownDW】样式中的数据来源于数据库,因此是动态的。这种编辑方式便于数据库数据的维护。【DropDownDW】样式的属性如图7.72所示。

在 DropDownDW 样式中主要设置如下属性。

- 【AutoRetrieve】复选框:用于设置是否自动检索下拉数据窗口的数据。
- 【Lines In DropDown】编辑框:用来设置下拉数据窗口列表中显示的数据项数。
- 【Width of DropDown】编辑框:用来设置下拉数据项的宽度(用相对于该列编辑框的宽度的百分比表示)。
- 【Data Window】编辑框:用来选择表示数据来源的数据窗口。

- **【Display Column】** 列表框：用于设置列表中要显示的列。
- **【Data Column】** 列表框：用来设置该字段要存储的值的列。

为了便于数据库的维护和节省存储空间，往往对一些数据编码存储，例如在学生成绩表中存储课程编号，而课程编号和课程名的对应关系存储在课程编号表中。在输入学生成绩时，为了直观起见，课程列显示课程名，但存储课程编号。

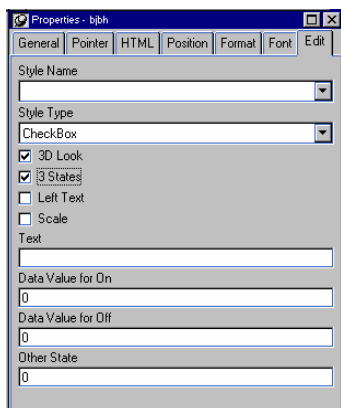


图 7.71 【CheckBox】样式

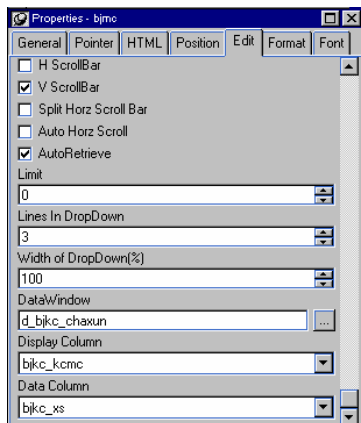


图 7.72 【DropDownDW】样式

4. 【DropDownListBox】(下拉列表框)

一般用来处理某列数据值个数为有限个数的情况，以列表的形式提供给用户选择，避免输入错误。同时也可以处理像**【DropDownListBox】**样式的代码表形式的数据，但**【DropDownListBox】**样式的数据是静态的。**【DropDownListBox】**样式的属性如图 7.73 所示。

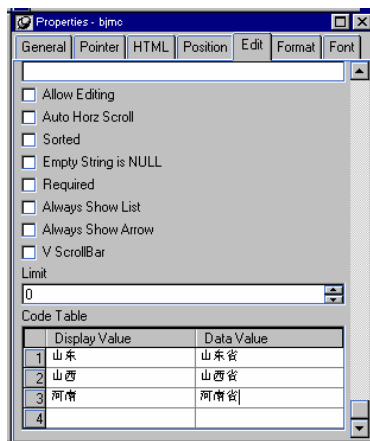


图 7.73 【DropDownListBox】样式

DropDownListBox 样式主要设置如下属性：选中**【Allow Editing】**复选框表示可以在下拉列表框中的编辑区域输入数据，否则，用户只能从提供的选项中进行选择；**【Sorted】**复选框用来指定是否将下拉列表框的数据按字母数字顺序排序；选中**【Always Show List】**

复选框会使下拉列表框的选项随焦点的下移而下移；选中【Always Show Arrow】复选框会使下拉列表框的向下箭头总是显示在区域的尾部。【Code Table】标题为向表中输入代码表，【Display Value】标题为显示值，【Data Value】标题表示与左边该行显示值对应的数据值。

5. 【EditMask】(编辑屏蔽框)

编辑屏蔽框样式用于设置数据输入的固定格式，避免输入错误，该格式通常用于日期、时间等字段。例如可定义日期的输入格式为：

yyy/mm/dd,

表示年/月/日。在输入数据时，光标会随用户的键入而向后覆盖掉屏蔽字，如图 7.74 所示。选中【AutoSkip】复选框时该数据项输入完毕，光标自动跳转到下一列。

6. 【RadioButtons】(单选按钮)

单选按钮样式用来在几个选项中做出唯一的选择，如图 7.75 所示。该图表示在学生基本情况录入数据窗口对象中的性别字段的设置。【Columns Across】编辑框用来设置在一行中显示的单选按钮个数，【Code Table】编辑框用来输入单选按钮代码表。

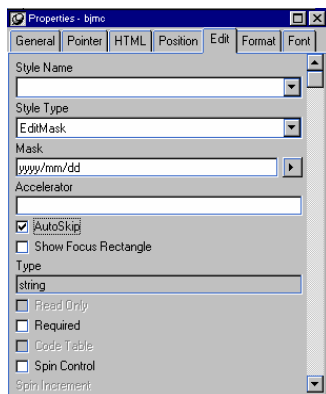


图 7.74 【EditMask】样式

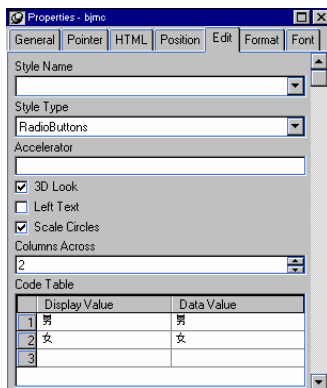


图 7.75 【RadioButtons】样式

7.6.3 数据校验

除了在数据库画板中定义有效性检验规则外，在数据窗口对象画板中，也可以对有效性规则进行设置和修改。方法是选择菜单【View】|【Column Specifications】命令，出现【Column Specification】子窗口，如图 7.76 所示。在【Validation Expression】栏中输入有

Name	Type	Prompt	Initial Value	Validation Expression	Validation Message	DB Name
1 bjbh	char(6)	<input type="checkbox"/>				banji.bjbh
2 bjmc	char(20)	<input type="checkbox"/>				banji.bjmc
3 zymc	char(20)	<input type="checkbox"/>				banji.zymc
4 xz	decimal(0)	<input type="checkbox"/>				banji.xz
5 rxsj	date	<input type="checkbox"/>				banji.rxsj
6 rs	decimal(0)	<input type="checkbox"/>				banji.rs

图 7.76 【Column Specification】子窗口

效性检验规则；也可以在该栏中单击鼠标右键，选择弹出菜单【Expression】命令，弹出【Modify Expression】对话框，用图形方式设置有效性检验规则。在【Validation Message】栏中输入错误提示信息。

7.7 设置更新属性

数据窗口对象的功能非常强大的原因就是它能修改数据库。当用户修改了数据窗口对象内的数据时，只需调用 Update()函数就能够更新数据库中的数据。

在创建数据窗口对象时，PowerBuilder 会自动设置默认的数据库更新属性来保存所做的修改。在对单个表进行操作的情况下，默认的更新属性能够满足用户的需要，可是在对多个表操作的情况下，应该更新设置、更新属性，以保证数据的完整性、一致性和安全性。

7.7.1 数据窗口的更新属性

对于定义了更新属性的数据窗口对象，可以通过修改数据窗口中的数据而实现对数据库的更新。所以数据窗口对象的更新属性实际上决定了能否更新它所对应的某个表，或者能否更新某个表中的某些列。

当一个数据窗口对象被创建时，它的更新属性取决于两个标准：一是看这个数据窗口对象是对应一个表还是对应多个表；另一个是看表的主键是否被数据窗口对象选中。

因为数据窗口对象只能更新一个表，如果数据窗口对象只对应一个表，那么这个表中的所有列都是可以更新的，并且此时所有列的 Tab 值均不为 0，以使用户能修改数据。但是如果数据窗口对象对应多个表时，则 PowerBuilder 定义表的所有列都不能更新，并且所有列的 Tab 值都为 0 使用户不能修改数据，这时必须手工修改数据窗口对象的更新属性。

要定义一个数据窗口对象的更新属性，选择【Rows】|【Update Properties】菜单命令，弹出【Specify Update Properties】对话框，如图 7.77 所示。

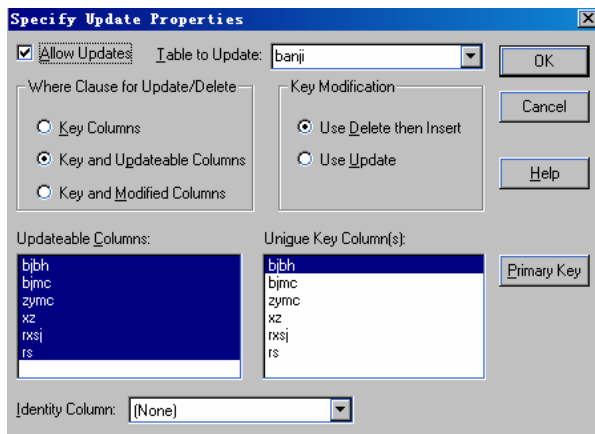


图 7.77 设置数据窗口对象的更新属性

- 【Allow Updates】复选框：该复选框决定是否允许数据窗口更新。如果这个复选框

没有选中,表明不能更新,而且其他选项都是不可选的。如果数据窗口对应多个表,就必须从下拉列表框【Table to Update】中选择一个表。

- **【Where Clause for Update/Delete】**分组框:该分组框内的3个单选按钮构造了 Where 子句的方法,不同的 Where 子句可以提供不同级别的数据库完整性保护。
- **【Key Columns】**分组框:表示让 Where 子句中只包含主键列。这些主键在下面的 **【Updateable Columns】**列表框中指定。在这种情况下,PowerBuilder 只把原来检索出来的关键列的值与数据库中关键列的值进行比较,如果这些值匹配,则能够更新数据库。
- **【Key and Updateable Columns】**按钮:该单选按钮是让 Where 子句中只包括主键列和可更新列。主键列在 **【Unique Key Columns】**列表框中,所有可更新的列在 **【Updateable Columns】**列表框中选择。在这种情况下,PowerBuilder 把原来检索出来的关键列和可更新列的值同数据库中的这些列值进行比较,如果匹配,则能够更新数据库。
- **【Key and Modified Columns】**按钮:该按钮是让 Where 子句中包括主键列和已作过修改的列。在这种情况下,PowerBuilder 把原来检索出来的关键列和已经更新的列的值同数据库中的这些列的值进行比较,如果自上次检索以后,这些列中任意一列的值与数据库中的列的值不匹配,则不能更新数据库。
- **【Key Modification】**分组框:该分组框内的两个单选按钮是用来决定当关键字被更新时的更新方式。有两种方式:
 - ◆ **【Use Delete then Insert】**按钮为先使用 Delete 语句删除,再使用 Insert 语句插入。
 - ◆ **【Use Update】**按钮为直接使用 Update 语句对主键列进行更新。
- **【Updateable Columns】**列表框:该列表框中列出了指定表的所有列。在这个列表框中选择哪些列可以更新,哪些列不可以更新。用鼠标单击加亮后,表示该列被选中。
- **【Unique key Column】**列表框:该列表框列出了指定表的所有列。从这个列表框选择该表的主键,用鼠标单击选中或取消选中。

单击 **【Primary Key】**按钮,可以将指定表的所有主键选中。

- **【Identity Column】**(标识列)下拉列表框:

许多数据库管理系统支持这样一种列。当向表中插入一条新记录时,此列的值有数据库管理系统自动指定,称之为标识列。不同的数据库管理系统支持的标识列的类型也不同。例如,Adaptive Server Anywhere 允许定义其值自动增加的列。在**【Specify Update Properties】**对话框中,可以指定一个标识列。这样,当在数据窗口中新插入一列并提交后,PowerBuilder 会自动把此列的值带回并显示出来。否则,需要重新检索才能看到此列的值。

7.7.2 多表数据窗口的更新

一般情况下,含有多表数据的数据窗口只能用于数据检索,而不能用于更新数据,所以,通常多表的更新操作使用多个数据窗口来实现(在脚本中要做数据库的相关完整性检查)。但有时也会遇到在一个数据窗口中更新多个表数据的情况。如果需要在单数据窗口中做多表更新,只能采用如下特殊的手段来解决:第一,通过修改数据窗口的更新属性实现多表更新;第二,用隐含数据窗口(或数据存储)实现多表更新。

其中第二种方法可将一个数据窗口对象中的不同表的数据先导入到不同的隐含数据窗口中，每一个表的数据通过不同的隐含数据窗口更新，所以其实质还是用多个数据窗口实现数据更新。下面介绍第一种方法的步骤如下。

1. 创建数据窗口

- (1) 选用合适的数据源和显示风格，保证数据窗口具有更新能力。
- (2) 选取多表中的所需数据项(各表的主键和非空列一定要包含进去)。
- (3) 修改所需数据项的【TabOrder】选项，保证其在数据窗口中是可修改的(TabOrder 值不为 0)。
- (4) 选取菜单【Rows】|【Update Properties】命令，定义数据窗口更新属性(只定义一个表的列为可更新列)。

2. 保存操作

- (1) 先对多表的一个单表进行更新,使用 Modify()函数将其余表中的数据项的【Update】属性改为 False。

- (2) 选取单表的主键及表中其他数据项，设置此单表如下。

```
"DataWindow.Table.UpdateTable=表名" //设置可更新的表名  
"表的主键.Key=Yes" //设置可更新的主键
```

使用 Modify()函数将要更新表中的数据项【Update】属性改为 True。

- (3) 用 Update()函数，对所选中的单表进行更新。
- (4) 选取数据窗口中的其余表及其数据项，重复以上操作来更新多表数据。

7.8 数据窗口的编程

数据窗口对象定义好以后，一定要与窗口界面上的数据窗口控件相关联。用户通过数据窗口控件提供的方法、事件和函数，可以编写 PowerScript 脚本代码来操作数据窗口对象，使得用户在应用程序窗口中可以方便地对数据库中的数据进行各种操作。

7.8.1 事务对象

1. 什么是事务对象

在应用程序中访问数据库是通过一个名为事务对象的不可见对象来进行的。与 Power Builder 中的其他对象(例如：窗口对象，菜单对象)不同的是，事务对象不是在画板中创建的，它是不可见的。实际上它是 PowerBuilder 程序与数据库之间传递信息的一个类似于结构变量的对象，它驻留在内存中，存储着用于连接数据库及从数据库得到反馈的所有信息。

事务对象中定义了数据库与应用程序连接的参数。在访问数据库之前，必须先建立一个事务对象，然后给这个事务对象的属性赋予相应的值，才能通过这个事务对象与数据库连接，完成所需要的数据库操作。

一个应用程序开始运行时，PowerBuilder 会自动地创建一个名为 SQLCA(SQL Communica

考虑 tions Area)的默认全局变量。可以在应用程序中使用这个默认全局变量,也可以用户自定义事务对象变量。

2. 事务对象的属性

事务对象共有 15 个属性,可以分为两类。第一类有 10 个,用于连接数据库的信息;第二类有 5 个,用来接受有关数据库或最近执行 SQL 命令的状态信息。下面分别说明这些属性。

- **AutoCommit**: 布尔型,用来指定是否将数据库设置成自动提交所有事务。当它为 True 时,由系统自动提交所有事务,为 False 时,必须有用户在程序中进行事务管理和向数据库提交事务,这个属性只用于 SQL Server,它的默认值为 False。
- **Database**: 字符串类型,指定要连接的数据库的名称。
- **DBMS**: 字符串类型,指定应用程序所连接数据库管理系统的名称。例如 Sybase、Oracle 等。
- **DBParm**: 字符串类型,用于向数据库传输特殊的信息。
- **DBPass**: 字符串类型,它包含了用户连接数据库的密码。
- **Lock**: 字符串类型,用于连接数据库的隔离层,是数据库的保护级别,为支持锁值和隔离层的 DBMS 提供的,一般不必给出这个值。
- **LogID**: 字符串类型,指定登陆数据库服务器的用户名或用户 ID。
- **LogPass**: 字符串类型,指定登陆数据库服务器的用户的口令。
- **ServerName**: 字符串类型,指定所连接的数据库服务器的名称。
- **SQLCode**: 长整型(long),存储最近一次对数据库操作成功与否的返回代码,有 3 个取值:

0: 表示操作成功。

100: 表示操作成功,但没有返回数据。


-1: 表示操作失败,用户可以从 SQLDBCode 和 SQLErrText 中得到详细的错误信息。

- **SQLDBCode**: 长整型(long),包含数据库错误代码,不同的数据库错误代码不同。但是一般数据库厂商都用“0”表示成功,“100”表示成功但没有找到数据,负数表示失败。
- **SQLErrText**: 字符串类型,返回对数据库操作的错误信息。
- **SQLNRows**: 长整型(long),表示最近一次 SQL 操作影响的行数。不同的数据库系统,它的含义也不同。
- **SQLReturnData**: 字符串类型,包含了数据库返回给用户的附加信息。不同的数据库其值不同。
- **UserID**: 字符串类型,指定了连接数据库的用户名或用户 ID。

设置事务对象的方法如下。

```
SQLCA.DBMS= "ODBC"  
SQLCA.Database= "mydatabase"  
SQLCA.LogID= "DBA"  
SQLCA.LogPass= "sql"  
SQLCA.ServerName= "myserver"  
SQLCA.Autocommit=FALSE
```

在应用程序中，用户不需要一行一行地输入这些语句，可以由系统自动生成这些语句，方法如下。

(1) 单击工具栏上的【DB Profile】按钮，在弹出的【Database Profiles】对话框中选择相应的 Profile 文件，然后单击【Edit】按钮，如图 7.78 所示。

(2) 这时弹出【Database Profile Setup-ODBC】对话框，选择 Preview 标签，在这个标签页的【Database Connection Syntax】多行编辑框中显示了系统生成的连接数据库的脚本语句。

(3) 单击【Copy】按钮，复制这段代码，在应用程序需要连接数据库的地方粘贴这段代码。如图 7.79 所示中显示了与 PowerBuilder 9.0 自带的数据库 Adaptive Server AnyWhere 8.0 连接的语句，将这段程序复制到程序中。如果要连接数据库，输入下面的语句：

```
Connect using SQLCA;
```

这样，在应用程序中就可以访问数据库了。

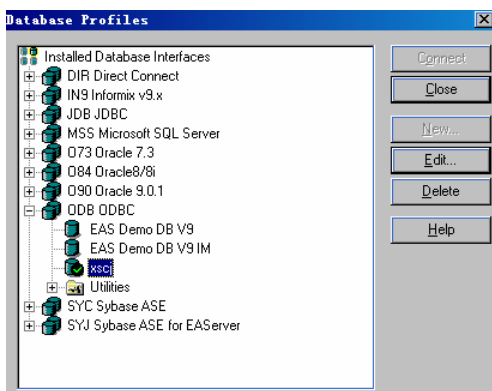


图 7.78 【Database Profiles】对话框

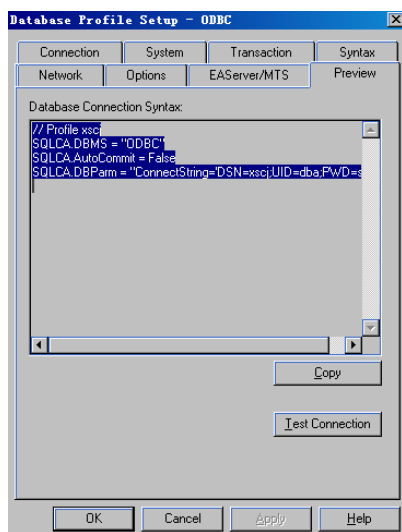


图 7.79 【Database Profile Setup-ODBC】对话框

3. 事务对象语句

PowerScript 有 4 条管理事务的语句，它们是：

- CONNECT
- DISCONNECT
- COMMIT
- ROLLBACK

下面一一介绍这几条语句。

1) CONNECT 语句

CONNECT 语句用于建立到数据库的连接，同时标志着一个事务的开始。它的语法格式是：

```
CONNECT [USING 事务对象名];
```

如果程序使用的是系统默认的事务对象 **SQLCA**，那么方括号中的内容可以省略；如果使用的是用户定义的事务对象，则方括号中的内容不能省略。例如：

```
CONNECT;  
或 CONNECT USING SQLCA;
```

这两条语句是等价的。

2) DISCONNECT 语句

DISCONNECT 语句用来断开与数据库的连接。它的语法格式为：

```
DISCONNECT [USING 事务对象名];
```

该语句的可选项 **USING** 事务对象名的使用方法与 **CONNECT** 相同。

3) COMMIT 语句

COMMIT 语句的功能是向数据库提交事务，即确定当前事务对数据库所做的修改，同时标志着一个事务的开始。它的语法格式为：

```
COMMIT;
```

4) ROLLBACK 语句

ROLLBACK 语句的作用是撤销自上次提交事务以来对数据库所做的修改工作，使数据库恢复到上一次提交后的状态。撤销一个旧的事务，也标志着一个新的事务的开始。

7.8.2 为数据窗口设置事务对象

假设现在已经有有了一个可用的事务对象，下面将简述一下将一个事务对象与数据窗口控件相连接的过程。

1. 设置事务对象的属性

一般来说事务对象的属性是在应用程序的 **Open** 事件中设置的，因为开始了一个应用程序，就有可能需要同数据库连接。设置一个事务对象的属性，可以按前面讲的方法，即在图 7.80 中的 **Database Profile Setup-ODBC** 对话框中，复制设置事务对象的程序代码。这儿讲述另外一种方法，就是利用 **PowerBuilder** 提供的几个函数和配置文件 **Profile**。

PowerBuilder 提供了 3 个函数来访问配置文件 **Profile**。

- (1) **ProfileString** (filename, section, key, default)。
- (2) **ProfileInt** (filename, section, key, default)。
- (3) **SetProfile** (filename, section, key, value)。

其中前两个函数用来获取 **Profile** 文件的信息，而最后一个函数用来设置 **Profile** 文件的信息。**filename** 是 **Profile** 文件的路径和名称，**section** 指的是 **Profile** 文件中的节，**key** 指的是 **Profile** 文件中的关键字，**default** 是在没有取到值的情况下的默认值。

在 **PowerBuilder** 中，**Profile** 文件一般是与文件 “**pb.exe**” 在同一目录下的 “**pb.ini**” 文件。下面是从 **Profile** 文件 **pb.ini** 文件中读取参数给事务对象的例子。

```
SQLCA.DBMS=ProfileString("pb.ini", "Database", "DBMS", " ")  
SQLCA.Database= ProfileString("pb.ini", "Database", "DataBase", " ")
```



```

SQLCA.LogID= ProfileString("pb,ini", "Database", "LogID", " ")
SQLCA.LogPass= ProfileString("pb,ini", "Database", "LogPassword", " ")
SQLCA.ServerName= ProfileString("pb,ini", "Database", "ServerName",
" ")
SQLCA.UserID= ProfileString("pb,ini", "Database", "UserID", " ")
SQLCA.DBPass= ProfileString("pb,ini", "Database", "DatabasePassword",
" ")
SQLCA.Lock= ProfileString("pb,ini", "Database", "Lock", " ")
SQLCA.DBParm= ProfileString("pb,ini", "Database", "DbParm", " ")

```

2. 连接数据库

在给事务对象设置完属性之后，就应该连接数据库了，一般在事务对象属性的设置语句后面用 **CONNECT** 语句来连接数据库。例如：

```

CONNECT USING SQLCA;
//判断连接数据库是否成功，使用 SQLCode 的值进行判断
IF SQLCA.SQLCode<>0 then
    MessageBox("数据库连接失败","不能连接到数据库,是因为"+SQLCA.SQLErrText)
    RETURN
END IF

```

3. 设置事务对象关联

连接了数据库后，要使用一个具体的数据窗口来访问数据库，还必须把事务对象与数据窗口控件关联起来。可以使用 PowerBuilder 提供的两个函数 **SetTrans()** 和 **SetTransObject()** 来为数据窗口控件设置事务对象。

1) SetTransObject()

使用这个函数的语句格式如下：

```
DataWindowControlname.SetTransObject(TransactionObjectName)
```

一般在用户要设置事务对象的数据窗口控件的窗口的 **Open** 事件中使用上面的一条语句，因为这个数据窗口控件直到窗口打开时才存在。

例如：下面的语句使数据窗口控件 **dw_control** 与默认的事务对象 **SQLCA** 连接起来，

```
dw_control.SetTransObject(SQLCA)
```

2) SetTrans()

这个函数的使用格式与上面一样，**SetTrans** 函数将一个特定的事务对象拷贝到数据窗口控件和内部事务对象中。当在程序中使用该函数时，数据窗口控件使用自己内部事务对象并且根据需要自动执行连接与断开连接的操作，如果出现错误，则自动撤销已做的所有更新操作，恢复到事务开始时的状态。

当数据窗口对象访问数据库时，如执行一个 **Retrieve** 或 **Update** 函数时，数据窗口会发出一个内部的 **CONNECT** 命令，处理相应的数据访问，然后再发出一个内部的 **DISCONNECT** 命令，断开与数据库的连接。所以在程序中使用 **SetTrans** 函数时，不需要另外书写 **CONNECT** 语句和 **DISCONNECT** 语句。

每次访问数据库时，**SetTrans** 都是自动地与数据库连接，访问完毕后再断开与数据库的连接。一般来说，应该避免使用 **SetTrans** 函数。因为 **CONNECT** 会占用相当一部分系统

资源且用户不能对事务处理进行管理。

提示: (1) 在脚本程序中, 可能会经常使用数据窗口控件而不是数据窗口对象, 用户可以把数据窗口控件想像成为数据窗口对象在窗口中的代表。当调用 `SettransObject()`、`SetTrans()`、`Update()`或 `Retreive()`这些函数时, 用户使用的是数据窗口控件而不是数据窗口对象。

(2) `SettransObject()`与 `SetTrans()`的不同之处在于 `SetTrans` 不要求使用 `CONNECT` 语句和 `DISCONNECT` 语句, `PowerBuilder` 会替用户自动完成这些功能。每次向数据库发送这样的命令语句时都会这么做, 因此当用户要求数据窗口对象检索数据时, 它做一次 `CONNECT`、`Retrieve`、`Disconnect`。当要求它更新数据库时, 它做一次 `CONNECT`、`Update`、`Disconnect`。这样每次与数据库交互, 应用程序的性能会大大降低。所以, 在一般情况, 用户还是使用 `SetTransObject()`函数来自己管理数据库的连接与断开。

7.8.3 用数据窗口连接数据库

数据窗口控件只有和数据窗口对象关联起来, 然后为数据窗口对象设置事务对象, 才能实现对数据库的连接与操作。有两种方式可以将数据窗口对象和数据窗口控件连接起来。一种是在窗口画板中给数据窗口设置 `DataObject` 属性, 这是一种最常见的方式。一种是利用脚本语句建立数据窗口控件和某个数据窗口对象的动态联系, 这样可以使一个数据窗口控件在不同的情况下连接不同的数据窗口对象, 以适应不同的情况。

1. 静态关联

静态关联就是上面所说的第一种方法, 实现的具体步骤如下。

(1) 首先要进入窗口画板, 并且保证打开的是要关联数据窗口对象的窗口, 在这个窗口上选择要关联的数据窗口控件。如果没有数据窗口控件, 就需要创建一个新的数据窗口控件, 方法在前面已经介绍过了。

(2) 用右键单击数据窗口控件, 选择弹出菜单中的【**Properties**】属性命令, 或者选择【**Views**】|【**Properties**】命令, 进入如图 7.80 所示的属性视窗。

(3) 在属性视窗的【**General**】标签页的【**DataObject**】编辑框中输入所要关联据窗口对象的名称; 或者单击右边的【**Browse**】按钮, 这时会弹出【**Select Object**】对话框, 如图 7.81 所示。

(4) 在这个对话框中, `DataWindows` 编辑框中是要关联的数据窗口对象的名称。它下面的列表框中列出当前选中的库的所有数据窗口对象, 可以从这个列表框中单击要关联的数据窗口对象的名称。

- 【**Comments**】多行编辑框显示当前选中的数据窗口对象的注释。

- 【**Application Libraries**】列表框列出了当前应用程序中的所有库文件。

如果当前库中的数据窗口对象太多, 可以利用【**Browse**】按钮来模糊查找所需要的数据窗口对象。

(5) 单击【**OK**】按钮, 返回到窗口画板中, 并保存所做的修改。

完成上述步骤之后, 就将一个数据窗口对象与窗口中的一个数据窗口控件连接起来了。

如果选择的数据窗口对象中存储了数据，那么现在在数据窗口控件将会看到数据。否则数据窗口控件只能看到数据窗口对象的标题信息。

在以后应用程序的设计中，如果用户想修改数据窗口控件所关联的数据窗口，也可以按照上面的方法在窗口画板中修改。

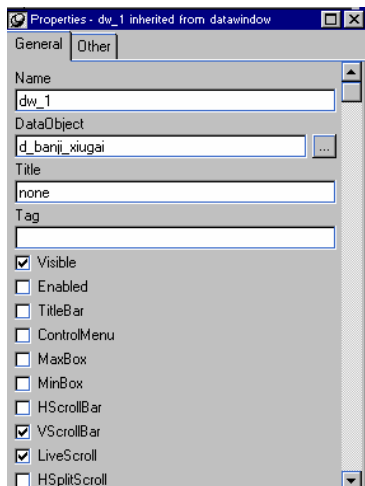


图 7.80 数据窗口控件的属性视图

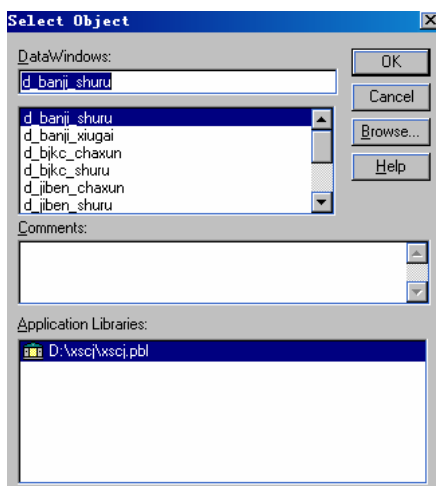


图 7.81 【Select Object】对话框

2. 动态关联

不过，在窗口画板中进行数据窗口控件和数据窗口对象的连接并不是唯一的方法，还可以在程序中编写一段程序，让数据窗口控件和数据窗口对象在运行的过程中动态连接起来。

从上面的设置属性中知道，数据控件和数据窗口关联是利用了数据窗口控件的一个称为 **DataObject** 的属性。这个属性的值是一个包含数据窗口对象名称的字符串。在程序中，可以动态地修改这个属性的值来动态地修改数据窗口对象，在要修改所关联的数据窗口对象的事件中输入下面的语句：

```
dw_controlname.dataobject= "d_objectname"
```

例如，语句：

```
dw_employee.dataobject= "d_employee"
```

是将数据窗口控件 **dw_employee** 中的数据窗口对象改名为 **d_employee** 的数据窗口对象。其中 **dw_employee** 已经在数据窗口画板中创建好了，并且保存在应用程序的可搜索路径下的库文件中。

在设置完一个数据窗口控件和一个数据窗口对象的关联之后，在数据窗口控件中只会显示数据窗口的一些标题信息，不会显示数据结果，原因是数据窗口控件还没有和数据库之间建立连接。

当用户在数据窗口画板中预览数据窗口对象时，会看到显示在数据库中的数据。此时用户会想，自己没有亲自建立起与数据库的连接，怎么就会显示数据了呢？原因是，在预览时，**PowerBuilder** 自动为数据窗口对象建立起了与数据库的连接，从而显示了数据，但

是当用户把应用程序编译成独立的可执行文件时,就必须亲自创建应用程序与数据库之间的连接。应用程序与数据库之间的连接是通过事务对象完成的。

7.8.4 数据的缓冲区

通过数据窗口控件在窗口上显示数据,进行插入、删除记录等维护操作是从表面上看到的数据记录的表现和数据窗口控件的行为。实际上,数据窗口控件从数据库中检索到的数据被放置在客户的一片专门的内存中,这片内存称为缓冲区。这片内存被分成了 3 个区域,亦即 3 个缓冲区,分别如下。

- (1) 主缓冲区(Primary Buffer)。
- (2) 删除缓冲区>Delete Buffer)。
- (3) 过滤缓冲区(Filter Buffer)。

其中,主缓冲区也称之为当前缓冲区。该缓冲区中的数据即为用户界面中可见的数据窗口部分,其他两个缓冲区都是不可见的,但用户可以对其进行有关的操作。

下面分别对这 3 个缓冲区做简要介绍。

1. 主缓冲区

主缓冲区存放的是没有被删除或者过滤掉的数据,这些数据直接显示在数据窗口控件中。当使用数据窗口的对象函数 `Retrieve()`检索数据时,检索到的数据就被放置到主缓冲区中;当插入新的数据行时,插入的数据也被存放到主缓冲区。各行和各列的当前状态用于数据窗口的数据保存时,将生成相应的 SQL 语句。主缓冲区的记录行数可以通过数据窗口的对象函数 `RowCount()`得到。

2. 删除缓冲区

删除缓冲区中存放的是被删除的数据窗口数据。当删除数据行时,不管是使用函数 `DeleteRow()`直接删除一行,还是使用 `RowsMove()`函数在缓冲区之间移动数据行,被删除的数据行都必须将从主缓冲区移动到删除缓冲区。当使用数据窗口控件的对象函数 `Update()`把对数据窗口所作的修改发送到数据库管理系统之后,被成功删除的记录均将从删除缓冲区中清除。在保存数据时,删除缓冲区将用于生成有关的 `DELETE` 语句。删除缓冲区的记录行数可以通过数据窗口控件的对象函数 `DeletedCount()`得到。

3. 过滤缓冲区

过滤缓冲区中存放的是被过滤出去的数据窗口数据。过滤缓冲区常用于保存那些满足数据源定义(即满足有关 `SELECT` 语句中的条件),同时却不满足用户所指定过滤条件的数据行。过滤缓冲区与主缓冲区一起在更新数据时生成相应的 `INSERT` 语句或者 `UPDATE` 语句。过滤缓冲区的记录行数可以通过数据窗口控件的对象函数 `Filter Count()`得到。

在程序运行过程中,数据窗口控件只显示主缓冲区中的数据,用户所进行的所有操作(包括数据的浏览、插入、删除、更新等),并且都是针对主缓冲区进行的,各缓冲区之间可以利用 `RowsMove()`函数进行数据的相互转移。

主缓冲区和删除缓冲区都有行级和列级的状态值,此状态值为枚举类型。在提交时,由行的状态值来决定是否要产生 SQL 语句。我们利用 `GetItemStatus()`函数和 `SetItemStatus()`

函数可以对这一状态值进行操纵。可取的各枚举状态值如下。

- **NotModified!**, 数据行的值为查询所得, 并没有发生什么改变。
- **DataModified!**, 数据行的值为查询所得, 但数据发生了改变。
- **New!**, 数据行为新插入的行, 数据没有发生改变(数据为空或默认值)。
- **NewModified!**, 数据行为新插入的行, 数据发生了改变(通过用户的键盘输入或者调用 `SetItem()` 函数)。

7.8.5 数据窗口的常用函数

PowerBuilder 9.0 为数据窗口提供了大量的对象函数, 这些函数使得其他的任何控件没有数据窗口控件的功能强大。灵活运用系统提供的这些函数, 可以极大地加快应用的开发进度、提高程序的运行效率。由于数据窗口控件的对象函数多达几百个, 限于篇幅不可以一一列举。在这里, 仅选择一些比较常用的数据窗口控件函数加以说明。

1. 利用检索函数检索数据

在完成数据库的连接并将给数据窗口控件设置完事务对象之后, 接着便可以使用数据窗口控件的 `Retrieve()` 函数从数据库中检索数据, 并将其显示到数据窗口中。在使用此函数的过程中, 可根据具体的数据检索要求来确定指定检索参数。

使用 `Retrieve()` 函数的语法格式如下:

```
dwcontrol.Retrieve({argument1、argument2...})
```

其中, `dwcontrol` 为数据窗口控件的名称; 参数 `argumentN` 为可选项。如果选用这些参数, 则可将这些参数的值用作数据窗口的对象 `SQL SELECT` 语句的数据提取参数。

该函数执行成功时, 返回所检索到的数据行数(即主缓冲的数据行数); 函数执行失败时返回-1; 若任何参数的值为 `NULL`, 则函数返回 `NULL`。

使用 `Retrieve()` 函数检索出数据之后, 系统将自动执行数据窗口的过滤条件, 不满足过滤条件的数据行将被立即移动到过滤缓冲区中, 函数返回的数值不包括移动到过滤缓冲区中的行数。

`Retrieve()` 函数是数据窗口控件的常用函数之一, 下面对其使用方法做几点说明:

(1) 在执行 `Retrieve()` 函数之前, 必须使用数据窗口控件的对象函数 `SetTransObject()` 或者 `SetTrans()` 为数据窗口控件设置事务对象。若使用 `SetTransObject()` 函数, 在调用之前需要 `CONNECT` 语句建立事务对象与数据库之间的连接。

(2) 在一般情况下, 执行 `Retrieve()` 函数之后, 数据窗口中原有的数据被丢弃, 并用新的数据来取代。若想改变这种默认的处理方式, 可以通过在数据窗口控件的 `RetrieveStart` 事件中编写代码来实现。方法是在该事件的处理程序中添加语句:

```
Return 2,
```

(3) 此时检索出的数据将被增加到原有数据的后面。

(4) 若数据窗口控件中的数据窗口对象需要检索参数, 而 `Retrieve()` 函数中又没有提供这些参数的值, 则在执行该函数时, 系统将会弹出一个对话框, 要求用户输入检索参数的取值。

(5) 执行 `Retrieve()` 函数的过程中, 可能会触发数据窗口控件的 `DBError`、`RetrieveEnd`、

RetrieveRow 事件以及 RetrieveStart 事件。

2. 插入行与删除行的函数

1) InsertRow 函数

InsertRow 函数用来在数据窗口的主缓冲区中插入一条新纪录。它的语法格式为：

```
dwcontrol.InsertRow(row)
```

其中：row 是个长整型的参数，它指定行的插入位置。若 row 为 0，则表示在最后一行插入新行。

例如：

```
dw_1.InsertRow(0)
```

2) DeleteRow 函数

DeleteRow 函数用来在数据窗口的主缓冲区中删除一行。它的语法格式为：

```
dwcontrol.DeleteRow(row)
```

其中：row 表示要删除行的行号。若 row 为 0，则表示删除当前行。

例如：

```
dw_1.DeleteRow(3)
```

表示删除数据窗口控件 dw_1 中的第 3 行。

3. 滚动数据行函数

完成数据行滚动的函数有 6 个，分别是函数 Scroll、ScrollToRow、ScrollNextRow、ScrollPriorRow、ScrollNextPage 和 ScrollPriorPage。

1) Scroll 函数和 ScrollToRow 函数

Scroll 函数按参数 number 所指定的数目来滚动数据窗口上的编辑控件中的文本行。Scroll 函数的语法格式为：

```
dwcontrol.Scroll(number)
```

其中参数 number 是要滚动的行数，它是一个相对值。当 number 大于 0 时向下滚动 number 行，number 小于 0 时则向上滚动 number 行。如果该函数执行成功，那么将返回编辑控件中第一行的行号。

ScrollToRow 函数

有时数据窗口主缓冲区的大小有限，因此如果在末尾插入了一行记录，那么用户不一定能在数据窗口中看到。这时可以使用 ScrollToRow 函数将数据滚动指定的行，则用户不必借助滚动条就可看到新行。

ScrollToRow 函数的语法格式为：

```
dwcontrol.ScrollToRow(row)
```

其中参数 row 指定要滚动的行号。如果该函数执行成功，则返回 1，否则返回-1。

2) ScrollNextRow 函数和 ScrollPriorRow 函数

这两个函数分别用来滚动到下一行和前一行。如果相应行已经在数据窗口的视野中,那么这两个函数的执行并不改变数据窗口中的显示。如果相应行不在数据窗口的视野中,那么这两个函数执行后相应于 `ScrollNextRow` 函数而言,将出现在数据窗口控件的下部,而对于 `ScrollPriorRow` 函数而言,将出现在数据窗口控件的上部。

3) `ScrollNextPage` 函数和 `ScrollPriorPage` 函数

同 `ScrollNextRow` 函数和 `ScrollPriorRow` 函数类似,这两个函数分别用来滚动到下一页和前一页。所谓一页是指在数据窗口控件中可见的全部记录的集合。这两个函数都返回滚动后数据窗口中最上面一行的行号。

4. 得到与设置数据项的函数

在应用程序的脚本中可以使用数据窗口控件的函数来访问指定行或列的值。这里介绍的函数是针对数据窗口的缓冲区的操作,它们不访问数据窗口上的编辑控件中的数据。

1) `GetItem` 函数

`GetItem` 函数从指定的行或列取得数据项的值,对不同数据类型的列使用的 `GetItem` 函数也不同。对于字符串、日期、日期时间、时间、数值、`Decimal` 等各种类型都有相应的一个函数,且各函数的返回值与其函数名最后面的字符串相同。例如, `GetItemDate` 函数的返回值类型为 `Date`。`GetItem` 函数的语法格式有以下 6 种:

```
dwcontrol.GetItemstring(row,column{, dwbuffer,originalvalue})
dwcontrol.GetItemdata(row,column{, dwbuffer,originalvalue})
dwcontrol. GetItemdatatime(row,column{, dwbuffer,originalvalue})
dwcontrol. GetItemtime(row,column{, dwbuffer,originalvalue})
dwcontrol. GetItemnumber(row,column{, dwbuffer,originalvalue})
dwcontrol. GetItemdecimal(row,column{, dwbuffer,originalvalue})
```

其中,参数 `row` 是个长整型,它用来指定数据所在行的行号;`column` 指定数据所在的列,它既可以使用一个整数来指明列号,也可以使用字符串指明列的名称。注意,计算列没用列号,必须使用列名。`dwbuffer` 指定读取数据的缓冲区,`originalvalue` 是一个布尔值,其值如果为 `True`,则表示返回原始缓冲区中的值。

2) `SetItem` 函数

`SetItem` 函数为指定行或列的项赋值,该值并不进行有效性检验,而是直接放置到数据窗口的主缓冲区中。

`SetItem` 函数语法格式如下。

```
dwcontrol.SetItem(row,column,value)
```

其中参数 `row` 是个长整数,它用来指定赋值项所在的行;`column` 是个整数或字符串,代表列号或列名,它用来指明赋值项所在的列;`value` 是要为数据项设置的数值。

该函数执行成功时返回 1,失败时返回-1。

5. 可编辑控件函数

所谓可编辑控件函数是指与数据窗口控件的可编辑控件相关的函数,用户在输入数据时,实际上是在可编辑控件中输入的。当用户移动输入焦点后,数据窗口会把可编辑控件中的数据移到对应的列中。

1) AcceptText 函数

AcceptText 函数告诉数据窗口控件：如果可编辑控件中还有数据未检验，那么应该检验它并把它放入主缓冲区中。

AcceptText 函数的格式为：

```
dwcontrol.AcceptText()
```

该函数没有参数，该函数执行成功时返回 1，失败时返回-1。

2) GetText 函数

GetText 函数用来访问在当前行或列的可编辑控件中还没有传送给数据窗口的数据。

GetText 函数的格式为：

```
dwcontrol.GetText()
```

例如下面的代码将返回可编辑控件中文本：

```
string FirstName  
FirstName = dw_employee.GetText()
```

6. 和数据库有关的函数

1) Update 函数

Update 函数用来把数据窗口上数据的变化更新到数据库中。

Update 函数的格式为：

```
Dwcontrol.Update({accept,{resetflag}})
```

其中：参数 accept 是一个布尔值。如果该值为 True，则表示在更新之前调用 AcceptText 函数。参数 resetflag 也是一个布尔值，它确定数据窗口更新后是否自动重置更新标记。

2) Sort 函数

Sort 函数使用当前数据窗口的排序条件对数据控件进行排序。

Sort 函数的格式为：

```
dwcontrol.Sort()
```

只有改变了排序条件后才有必要调用该函数，可以用 SetSort 函数改变排序条件。调用格式为：

```
dwcontrol.SetSort(format)
```

其中参数 format 是一个表示排序标准的字符串。

3) SetFilter 和 Filter 函数

在应用程序的脚本中，这两个函数经常配合使用。SetFiler 函数在程序中动态修改数据窗口过滤条件。修改之后，再调用 Filter()函数过滤数据。

SetFilter 函数的语法格式为：

```
dwcontrol.SetFilter(format)
```

其中参数 format 是个字符串，其值是作为过滤条件的逻辑表达式，表达式中可以包括列名或列号，如#2 表示第二列。如果 format 的值为 NULL，PowerBuilder 将显示“Specify Filter”

对话框，让用户输入过滤条件。

该函数执行成功时返回 1，否则返回-1。

Filter()函数的语法格式为：

```
dwcontrol.Filter()
```

4) Reset 函数

Reset 函数可以清除数据窗口中的所有行。

它的语法格式为：

```
dwcontrol.Reset()
```

若该函数执行成功，则返回 1，否则返回-1。

执行 Reset 函数后再执行 Update 函数并不会清除数据库，而删除数据窗口的所有行后再执行 Update()函数则会清除数据库中的数据。

7.8.6 数据窗口的事件

数据窗口的强大功能不仅表现在数据窗口对象有丰富的显示风格和灵活的数据源上，而且表现在数据窗口控件的众多事件上。用户可以利用这些事件来编写事件处理程序，从而能进一步控制程序的执行。

对数据窗口的事件的理解，有助于用户更加明确地控制数据窗口的动作行为。对数据窗口控件来说，除了用户操作能够触发事件外，某些函数在执行过程中也将触发事件。不同事件触发的时机不同，返回值的意义不相同。只有充分理解事件的这些特性，才能根据需要灵活地控制应用程序。下面将对数据窗口的一些主要事件做一个大概的介绍。

1. Clicked 事件

当单击某个不可编辑字段或在数据窗口的字段间单击时触发。

2. Constructor 事件

在该事件窗口的 Open 事件前被触发，是数据窗口控件的构造事件。

3. Destructor 事件

该事件在窗口的 Close 事件后发生，是数据窗口控件的析构事件。

4. DBError 事件

该事件是一个错误处理事件。在调用函数 Retrieve()或 Update()时导致某种数据库错误时触发。此时，该事件的参数 sqldbcode 中包含由数据库厂商提供的特定出错代码，用户可以查看数据库厂商提供的有关资料。如果厂商未提供出错代码，则 sqldbcode 取下列值之一：

- 1——因为事务对象中未正确提供某些值而不能连接到数据库。
- 2——不能连接到数据库。
- 3——Update 或 Retrieve 中指定的键值与现有行不匹配。
- 4——向数据库中写 Blob 型数据时发生错误。

DBError 事件的默认返回值是 0，显示保存在事务对象的 SQLErrText 属性中的出错信息，如果返回 1，则表示不显示出错信息。

例如：

```
IF sqldbcode=-195 THEN
    MexxageBox("丢失信息", & "你没有提供足够的值给出" & + "所需的区域" )
END IF
```

5. Error 事件

该事件在发现数据窗口对象的数据或属性表达式有错误时发生。它有 8 个参数，分别是 errornumber、errortext、errorwindowmenu、errorobject、errorscrip、errorline、action 和 returnvalue。读者可以由名称推测出它们的含义。例如可以在 Error 事件中编写以下代码，它将显示发生的错误信息，并允许代码继续执行：

```
MessageBox("错误号" + string(errornumber)& + "已经发生", "错误  
为: "+String(errortext))  
action=ExceptionIgnore!
```

6. ItemError 事件

该事件在用户修改了某字段后移走焦点，但数据未能通过该列的有效性检查的时候发生。它的参数如下。

- row: 长整型，表示包括发生改变且没有通过有效性检查的字的段的行号。
- dwo: DWOObject，发生错误的数据库窗口对象。
- data: 字符串型，新输入的发生了改变的项的数据。

ItemError 事件的返回值如下。

- 0——默认值，拒绝输入值并显示有效性验证出错信息，同时不允许移走焦点。
- 1——拒绝输入值，但不显示出错信息，同时不允许移走焦点。
- 2——接收输入值。
- 3——拒绝输入值，但允许移走焦点。

7. ItemChanged 事件

该事件在用户完成对某一可编辑字段的修改，然后通过其他操作使当前字段失去输入焦点时(输入值通过有效性检查后)发生。如果该字段的新值与修改前的原有值相同，则不触发 ItemChanged 事件。

该事件的返回值如下。

- 0——默认值，接收当前数据值。
- 1——拒绝输入值且不允许移走焦点。
- 2——拒绝输入值但允许移走焦点。

8. ItemFocusChanged 事件

该事件发生在焦点从一个可编辑字段切换到另一个可编辑字段的时候。

参数 row 表示获得焦点的数据行，dwo 表示获得焦点的字段对象。

9. RowFocusChanged 事件

该事件发生在当前行发生变化时。

参数 `currentrow` 表示新数据行的行号。

10. PrintStart 事件

该事件在数据窗口开始打印之前发生。

参数 `pagemax` 表示打印的页数。

11. PrintPage 事件

该事件在数据窗口中的每页已经被格式化后，并且准备打印之前发生。

参数 `pagenumber` 表示要打印的页号，`copy` 表示要打印的拷贝号。

该事件的返回值的含义如下。

0 ——不跳过一页，为默认值。

1 ——跳过一页。

12. PrintEnd 事件

该事件在数据窗口打印结束后发生。

参数 `pagesprinted` 表示已经打印了的页数。

13. RetrieveStart 事件

该事件在为数据窗口检索数据前发生。

它的返回值的含义如下。

0 ——默认值，继续检索。

1 ——执行检索。

2 ——从数据库中检索数据前不重置数据及缓冲区。

14. RetrieveRow 事件

该事件在检索一条记录并送到数据窗口后的时刻发生。

它的返回值的含义如下。

0 ——默认值，继续后面的检索。

1 ——终止检索过程。

例如：可以用该事件来限制检索的行数。以下的代码表示最多只能检索出 250 行数据：

```
IF row >250 THEN
    MessageBox("检索停止", & "你已经检索出了 250 条数据，它是最大值！")
    RETURN 1
ELSE
    RETURN 0
END IF
```

15. UpdateStart 事件

该事件在调用 `Update()` 函数来修改数据库中的数据前发生。

它的返回值的含义如下。

0——默认值，继续更新过程。

1——不执行更新。

16. UpdateEnd 事件

该事件和 UpdateStart 事件相对应，发生在当数据窗口对数据库进行更新后的时刻。

它有 3 个长整型的参数，含义如下。

- rowsinserted: 表示更新操作中新插入数据库的行数。
- rowsupdated: 表示更新操作中被更新的行数。
- rowsdeleted: 表示更新操作中被删除的行数。

17. SQLPreview 事件

该事件发生在调用 Retrieve、Update 或 ReselectRow 函数后，SQL 语句被发送到 DBMS 之前。

它有 5 个参数，各参数的含义如下。

- Request: 表示要求访问数据库的函数类型。
- Sqltype: 表示发送到 DBMS 的 SQL 语句的类型。
- Sqlsyntax: 表示全部 SQL 语句的文本。
- Buffer: 表示与数据库操作有关的包含数据行的缓冲区。
- Row: 表示将要被更新、插入、选取或删除的数据行。

SQLPreview 事件的返回值的含义如下。

0——继续执行。

1——停止执行。

2——跳过当前请求，接着执行下一个请求。

7.8.7 如何标识数据窗口中数据

在数据窗口缓冲区中每一个字段都称之为一个项。在 PowerScript 中可以通过字段名或代表数据窗口中列的相关位置数字(列号)来引用每一项(列的编号指的是在 SQL SELECT 语句中的各列引用顺序，而不一定是在表中的顺序。序号从 1 开始)，把这种对项的引用称之为表达式引用；用户还可以利用函数对各项进行引用。

1. 通过表达式引用数据窗口项

利用数据窗口控件的对象属性可以访问与数据窗口控件相关联的数据窗口对象中的数据，使用表达式来引用数据窗口项可以访问数据窗口中的某一项、某一列，甚至整个数据窗口中的数据。

1) 通过列名引用数据窗口项

使用列名引用数据窗口项的一般格式如下：

```
dwcontrol.object.columnname{.buffer}{.current | .original}{[rownumber]}
```

各参数含义如下。

- **dwcontrol**: 数据窗口控件名称。
- **object**: 必选部分, 用来指定一个数据窗口对象的数据值。
- **columnname**: 必选部分, 用来指定要操作的数据窗口列名。
- **buffer**: 可选部分, 指明要从哪个缓冲区中取得数据, 用户可以在 **Primary**(主缓冲区)、**Delete**(删除缓冲区)、或 **Filter**(过滤缓冲区)中选择一个, 默认设置为 **Primary** 缓冲区。
- **.current | .original**: 用来指明是从原始数据(从数据库中检索出来的数据)还是从当前缓冲区中取得数据。**curent** 为默认设置, 指当前数据缓冲区。**original** 指原始数据。
- **rownumber**: 可选字段, 指定数据窗口的行号或某一范围。

下列语句把数据 9 952 101 赋值给主缓冲区中的第一行的 **number** 列项:

```
dw_1.object.number[1]=9952101
```

下列语句从主缓冲区中获得 **name** 列的第 3 行~第 5 行的数据并赋值给一个已经定义的数组, 然后将取出的数据写到单行编辑器中:

```
string name[]
name=dw_1.object.name.primary[3,5]
sle_2.text=name[1]+ "---" +name[2] + "---" +name[3]
```

2) 访问所有行

用法与上述的调用格式类似, 不同之处是省略了访问数据所在的行号:

```
dwcontrol.object.columnname{.buffer}{.current | .original}
```

例如, 下列语句从当前主缓冲区中获得 **name** 列的所有行的数据, 并赋值给一个已定义的数组:

```
string name[]
name=dw_1.object.name.primary
```

3) 通过列号引用数据窗口项

使用列号引用数据窗口项的一般格式如下:

```
dwcontrol.object.columnnumber{.buffer}{.current  
| .original}{[rownumber]}
```

各参数的含义与前面的含义类似, 其中 **columnnumber** 用来指定一个数据窗口列号。

提示: 通过列号引用列时, 在列号前要加#号, 例如#2 表示数据窗口对象的 SQL SELECT 语句中的第 2 列。

例如, 下列语句从过滤缓冲区中获得 **name** 列(列号为 2)的第 3 行~第 5 行的数据并赋值给一个已定义的数组:

```
string name[]
name=dw_1.object.#2.Filter[3,5]
sle_2.text=name[1]+ name[2] name[3]
```

2. 通过函数引用数据窗口项

PowerBuilder 提供了两类用来访问数据窗口项的函数: 一类函数可以取得数据窗口缓

冲区中特定行列的数据(GetItem 簇函数); 另一类函数可以设置数据窗口特定行列的数据值(SetItem 函数), 具体使用方法参考本章 7.8.5 节的介绍。

例如, 将数据窗口当前行的学号赋值给 num_no 变量, 脚本如下:

```
rownum=dw_1.getrow() //取得当前行的行号
num_no=dw_1.GetItemNumber(rownum,number) //取得当前行及 number 列的值
```

提示: 在取得某数据项的数据时, 对于计算列没有列号, 但必须使用列名。

7.9 打印数据窗口

数据窗口不仅是屏幕上显示数据和操作数据库的界面, 还经常用它来完成打印的功能。

7.9.1 数据窗口的打印设置

数据窗口的打印属性是在 Print Specifications 标签页中设置的。具体内容参阅本章 7.4.2 节数据窗口对象属性设置部分的介绍。

7.9.2 预览与打印

1. 打印预览

设置完数据窗口的打印属性后, 可以在数据窗口的预览模式下检查所设置的参数是否起作用。打印预览操作方法如下。

- (1) 确保数据窗口的预览工作区已经打开, 如果还未打开, 可选择菜单【View】|【Preview】将其打开。
- (2) 选择菜单【File】|【Preview】命令, 数据窗口预览工作区变为打印预览模式。
- (3) 选择菜单【File】|【Preview Rulers】命令, 打印预览模式会显示标尺, 但该标尺不会打印出来。标尺的单位由数据窗口属性的【General】标签页的【Units】选项决定, 如图 7.82 所示。
- (4) 选择菜单【File】|【Preview Zoom】命令, 弹出【Zoom】对话框, 如图 7.83 所示。选择不同的比例, 然后单击【OK】按钮, 可设置打印预览模式的显示比例。该设置只是一种显示设置, 不是打印的实际尺寸。



图 7.82 数据窗口对象的打印预览模式图

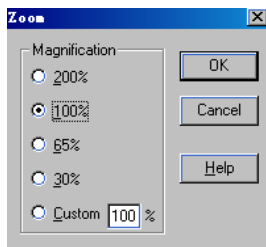


图 7.83 【Zoom】对话框

2. 数据打印

用户在对数据窗口对象进行打印预览时，还可将其中的数据打印出来。其步骤如下。

- (1) 选择菜单【File】|【Print Report】命令，弹出如图 7.84 所示的【Print】对话框。
- (2) 在对话框的【Copies】编辑框中设置报表的打印份数。

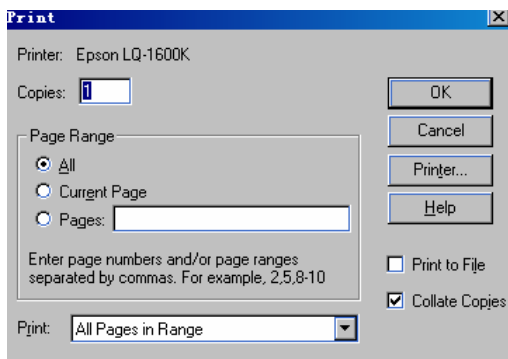


图 7.84 Print 对话框

(3) 在【Page Range】组框中指定打印的页码范围：【All】选项表示打印报表所有页；【Current Page】选项表示打印当前页；【Pages】单选按钮用于指定打印页，在后面的编辑框中输入打印的页码范围。

(4) 在【Print】下拉列表框中选择选项【All Page in Range】(打印范围内的所有页)、【Even Pages】(打印范围内的偶数页)、【Odd Pages】(打印范围内的奇数页)。

(5) 【Collate Copies】复选框用来设置副本页的排列次序(打印次序)，若选中该复选框，则报表逐份打印。若不选中该复选框，则逐页打印。只有在选择打印多份报表时，才能体现出该复选框的作用。

(6) 单击【Printer】按钮可以选择打印机。

(7) 设置完成后，单击【OK】按钮即可打印。

(8) 如果要把数据输出打印到一个文件中，则选择【Print to File】复选框，单击【OK】按钮，显示如图 7.85 所示的【Print to File】对话框。

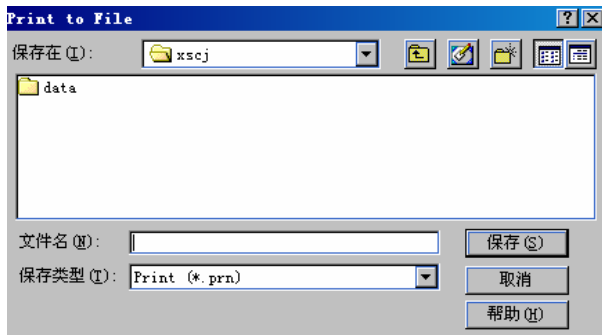


图 7.85 【Print to File】对话框

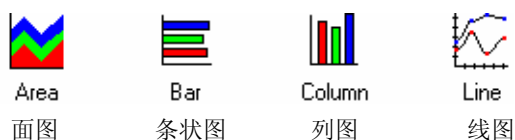
7.10 统计图的使用

统计图是一种用图形表示数据的方式,这种表现风格使得数据看起来更直观,更容易对数据进行比较、分析。在应用程序的开发过程中,统计图常用于统计、分析的场合。

7.10.1 统计图的种类

PowerBuilder 提供了 17 种标准的统计图形,可分为 9 类:面图、条状图、列图、线图、饼图、散点图、三维统计图及其他。

1. 面图、条状图、列图、线图



这几种图形的属性十分相似,只是显示数据的方式不同。如图 7.86~图 7.89 所示为这几种形式的图例。

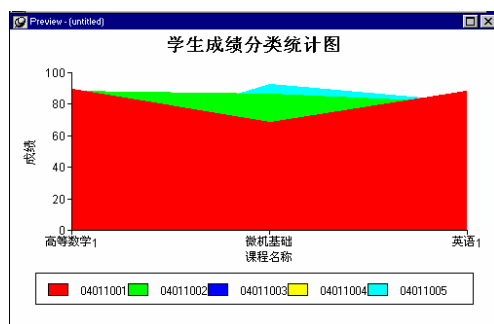


图 7.86 面图案

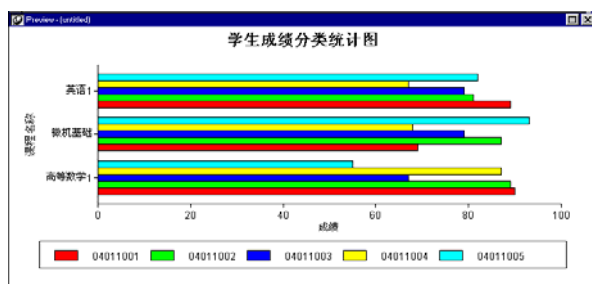


图 7.87 条状图

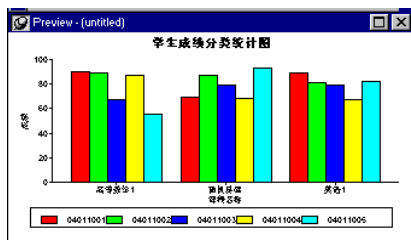


图 7.88 面图案

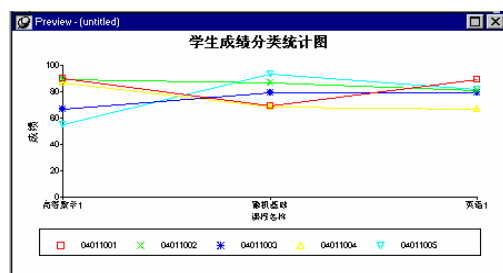


图 7.89 条状图

2. 饼图

饼图是用图形方式显示同一序列数据点中各数据在整体中所占的比例的图形方式。在饼图中,检索数据时,PowerBuilder 会自动计算每个切片的百分比,如图 7.90 所示为饼图。

04级班级人数分布图

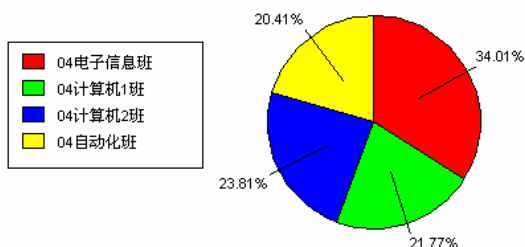


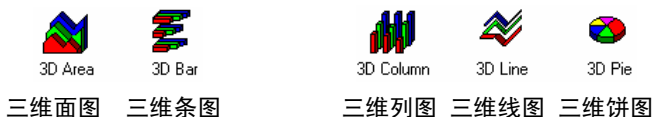
图 7.90 饼图

3. 散点图

这类统计图不使用分类(series)，它显示(xy)数据点(坐标点)。

4. 三维统计图

三维统计图包括三维面图(3D Area)、三维条图(3D Bar)、三维列图(3D Column)、三维线图(3D Line)和三维饼图(3D Pie)。



5. 堆积图

在堆积图中，每个分类用一个条或列代表。堆积图包括堆积条图(Stacked Bar)和堆积列图(Stacked)。



6. 其他

包括实心条图(Solid Bar)、实心列图(Solid Column)、实心堆积条图(Solid Stacked Bar)、实心堆积列图(Solid Stacked Column)。



7.10.2 统计图属性的定义

数据窗口中统计图对象的属性在数据窗口的属性的各标签页中定义。其【General】标签页如图 7.91 所示。

在【General】标签页中可设置如下属性。

- 在【Title】编辑框中输入统计图的标题文本。该文本将显示在统计图顶部。在文本中可以使用“~n”强制标题换行。
- 在【GraphType】下拉列表框中选择统计图类型。
- 在【Legend】下拉列表框中选择图例位置：默认为底部(Bottom)。

统计图数据窗口的【Data】标签页如图 7.92 所示，在【Data】标签页中可设置以下属性。

- 在【Category】列表框中选择分类轴列。
- 在【Value】下拉列表框中选择值轴列。
- 选中【Series】复选框则可以设置分类序列。
- 在【Series】下拉列表框中选择序列。

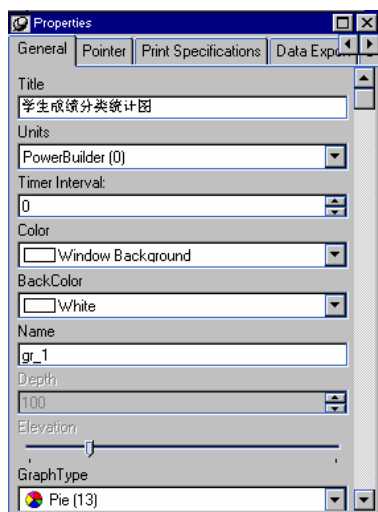


图 7.91 数据窗口属性的【General】标签页

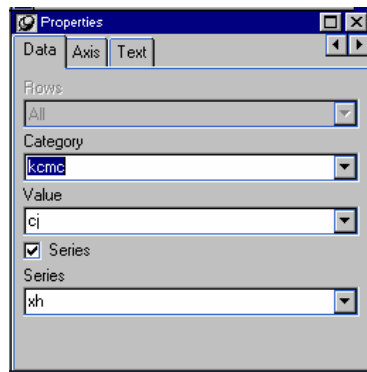


图 7.92 统计图数据窗口属性的【Data】标签页

【Axis】标签页如图 7.93 所示。

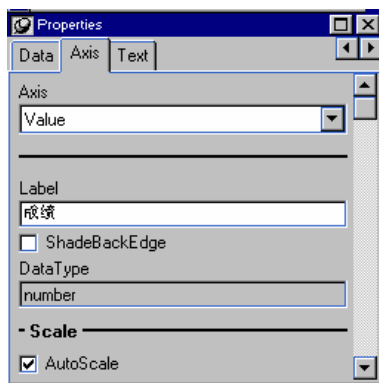


图 7.93 统计图数据窗口属性的【Axis】标签页

在【Axis】标签页中可设置统计图各轴标签文本。在【Axis】下拉列表框中选择【Value】选项后，在【Label】编辑框中输入值轴文本；在【Axis】下拉列表框中选择【Series】选项后，在【Label】编辑框中输入序列文本。

在【Text】标签页中设置各轴的标签、文本、标题、图例等。

7.11 小 结

本章介绍了数据窗口对象的基本知识，包括各种风格的数据窗口对象及其数据源的选择。通过数据窗口画板的使用，可以进一步修饰向导创建的数据窗口对象、设置和更改数据窗口对象及内部放置对象的属性、设置各种显示格式和编辑样式，以求达到最佳的效果、实现更多的功能。在数据窗口画板中还可以对数据进行排序、过滤、打印等操作，提供了对数据库中的数据进行维护操作的一种环境。可以在应用的窗口上实现对数据库中数据的各种操作。

另外，介绍了数据窗口编程方面的事务处理对象、数据窗口控件的函数、事件及与数据库的连接方法。

7.12 实 训

实训目的

- (1) 进一步加深对数据窗口的了解，熟悉数据窗口画板的使用和数据窗口对象属性的调整。
- (2) 掌握数据窗口对象的标签和字段的属性的调整方法。
- (3) 了解数据窗口对象的各种编辑样式的特点、编程和使用方法。
- (4) 掌握数据窗口各种常用操作(插入、删除、更新等)的程序设计方法。

实训内容

- (1) 复习有关数据窗口对象及数据窗口控件的内容。
- (2) 启动 PowerBuilder 9.0。
- (3) 选择已经创建的工作区 xscj.pbw。
- (4) 连接数据库到第2章中创建的 xscj.db(数据源名和 db profile 名为 xscj)。
- (5) 制作一个录入学生基本信息的数据窗口，如图 7.94 所示。该窗口中有一个反映学生基本信息的数据窗口对象 d_jiben_shuru，该数据窗口中又包含另一个班级修改的数据窗口对象 d_banji_xiugai，有 4 个前后查看记录的按钮以及对数据库进行添加、删除、保存和退出的 4 个按钮。“性别”字段采用单选按钮编辑风格。在“班级编号”字段上单击鼠标，会弹出如图 7.93 所示的下拉列表框，提供数据的选择，避免了每次输入字符串，使用非常方便。

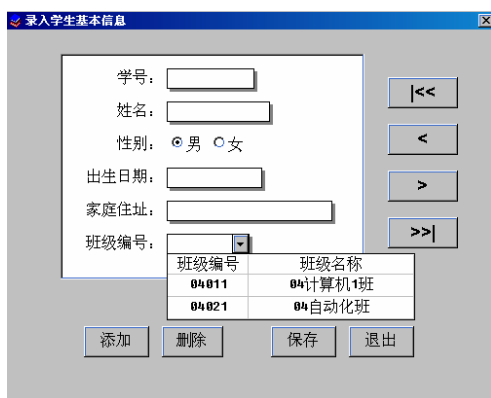


图 7.94 “录入学生基本情况”窗口

实训步骤

1. 创建数据窗口对象

首先创建 d_banji_xiugai 数据窗口对象，步骤如下。

(1) 单击【New】图标按钮，弹出【New】对话框。

(2) 选择【DataWindow】标签页，双击【Grid】图标，弹出【Choose Data Source for Grid DataWindow】对话框。

(3) 选择【Quick Select】数据源方式，单击【Next】按钮，弹出【Quick Select】对话框。

(4) 在【Tables】列表框中选择 banji 表，这时在其右边的【Columns】列表框中列出了 jiben 表的 6 个字段的名称，单击【Add All】按钮，就将这 6 个字段添加到了下面的区域中。单击【OK】按钮，弹出【Select Color and Border Settings】对话框。

(5) 单击【Next】按钮，弹出【Ready to Create Grid DataWindow】对话框，单击【Finish】按钮，进入数据窗口画板。

(6) 单击【Design】子窗口中“bjbh”字段标签，将【Properties】属性卡【General】页中的【Text】属性改为“班级编号”。类似地，将“bjmc”、“zymc”、“xz”、“rxsj”、“rs”字段标签的【Text】属性分别改为“班级名称”、“专业名称”、“学制”、“入学时间”、“人数”。

(7) 调整数据窗口对象中字段的布局。

(8) 按下【Save】按钮，保存数据窗口对象名称 d_banji_xiugai，完成数据窗口对象的创建工作。

然后创建 d_jiben_shuru 数据窗口对象，步骤如下。

(1) 单击【New】图标按钮，弹出【New】对话框。

(2) 选择【DataWindow】标签页，双击【Freeform】图标，弹出【Choose Data Source for Freeform DataWindow】对话框。

(3) 选择【Quick Select】数据源方式，单击【Next】按钮，弹出【Quick Select】对话框。

(4) 在【Table】列表框中选择 jiben 表，这时在其右边的【Columns】列表框中列出了

jiben 表的所有 6 个字段的名称,单击【Add All】按钮,就将这 6 个字段添加到了下面的区域中。单击【OK】按钮,弹出【Select Color and Border Settings】对话框。

(5) 单击【Next】按钮,弹出【Ready to Create Freeform DataWindow】对话框,单击【Finish】按钮,进入数据窗口画板。

(6) 单击【Design】子窗口中“xh:”字段标签,将【Properties】属性卡【General】页中的【Text】属性改为“学号:”。类似地,将“xm:”、“xb:”、“csrq”、“jtzz:”、“bjbh:”字段标签的【Text】属性分别改为“姓名:”、“性别:”、“出生日期:”、“家庭地址:”、“班级编号:”。

(7) 调整数据窗口对象中字段的布局。

(8) 将“xb”字段以外的所有字段选中,在【Properties】标签页的【General】页的【Border】属性的下拉列表框中选择【Shadowbox(1)】选项。

(9) 选中“xb”字段,在【Edit】页的【Style Type】属性的下拉列表框中选择【RadioButtons】选项,选中【3D Look】复选框。

(10) 在【Columns Across】页下的编辑框中输入“2”,在【Code Table】页下的【Display Value】、【Data Value】编辑框中分别输入“男”、“女”。

(11) 选中“bjbh”字段,在【Edit】页的【Style Type】属性的下拉列表框中选择【DropDownDW】选项,选中【VScrollBar】、【AutoRetrieve】两个复选框,其他属性的设置如图 3.95 所示。

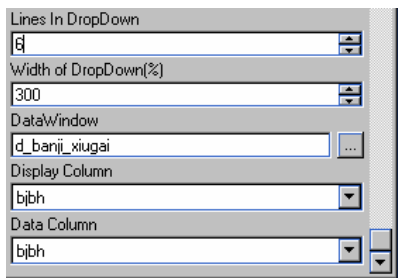


图 7.95 字段“bjbh”属性框中的编辑页面

(12) 单击【Save】按钮,保存数据窗口对象名称 d_jiben_shuru,完成第二个数据窗口对象的创建工作。

2. 创建窗口对象

创建一个新的主窗口,Title 属性改为“录入学生基本信息”,其他属性项的设置为默认值,保存窗口名称为 w_jiben_shuru。创建窗口的具体步骤参见第 5 章的有关内容。

3. 向窗口中添加控件

(1) 设置窗口中的“添加”、“删除”、“保存”和“退出”4 个命令按钮控件。

在窗口控件下拉列表工具栏图标中选择命令按钮控件,然后在窗口上单击,生成命令按钮控件。设置按钮的名称分别为 cb_1, cb_2, cb_3 和 cb_4。它们的【Text】属性分别改为“添加”、“删除”、“保存”和“退出”。选中“保存”按钮,去掉其【Enabled】复选框。

(2) 布置窗口中“|<<”、“<”、“>”和“|>>”4 个命令按钮控件。

在窗口控件下拉列表工具栏图标中选择命令按钮控件，然后在窗口上单击，生成命令按钮控件。设置按钮的名称分别为 cb_5, cb_6, cb_7 和 cb_8。它们的 Text 属性分别改为“|<<”、“<”、“>”和“|>>”。

(3) 设置一个数据窗口控件。

在窗口控件下拉列表工具栏图标中选择数据窗口控件，然后在窗口上单击，生成数据窗口控件，名称为 dw_1。单击【DataObject】栏右边的【...】按钮，选择数据窗口对象 d_jiben_shuru，调整数据窗口控件的大小，使其能够完整地显示数据窗口对象。

4. 编写脚本

(1) 窗口 W_jiben_shuru 的 Open 事件中输入以下脚本。

```
dw_1.SetTransObject(sqlca)
dw_1.InsertRow(0)
```

(2) 【添加】按钮的 Click 事件脚本如下。

```
dw_1.scrolltorow(dw_1.insertrow(0))
```

(3) 【删除】按钮的 Click 事件脚本如下。

```
dw_1.deleterow(0)
cb_3.enabled=true
```

(4) 【保存】按钮的 Click 事件脚本如下。

```
if dw_1.update(true,false)=1 then
    dw_1.resetupdate( )
    commit;
    cb_3.enabled=false
else
    rollback;
    messagebox("错误","保存数据失败!")
end if
```

(5) 【退出】按钮的 Click 事件脚本如下。

```
close(parent)
```

(6) 【|<<】按钮的 Click 事件脚本如下。

```
dw_1.scrolltorow(1)
```

(7) 【<】按钮的 Click 事件脚本如下。

```
dw_1.scrollpriorrow( )
```

(8) 【>】按钮的 Click 事件脚本如下。

```
dw_1.scrollnextrow( )
```

(9) 【>>|】按钮的 Click 事件脚本如下。

```
dw_1.scrolltorow( dw_1.rowcount( ) )
```

5. 运行程序

运行程序，输入数据，观察几种数据窗口字段风格的特点及命令按钮的操作效果。

7.13 习 题

1. PowerBuilder 提供了哪几种数据源？各自适合于什么场合？
2. 数据窗口的风格有哪些？分别适用于什么场合？
3. 数据窗口画板 DataWindow Painter 由哪些视图窗口组成，分别有什么用途？
4. 数据窗口画板 DataWindow Painter 中设计视图 Design View 可以分为哪几个区域？各个区域分别有什么用途？
5. 数据窗口对象上除可以放置列对象、文本对象外，还可以放置哪些对象？
6. 通过向导建立一个 DataWindow Object 后可以更改它的数据源的表或字段吗？如果可以如何更改？
7. 在 DataWindow Object 的 Data Source 中可以指定可计算列。在 DataWindow Object 画板中也可以添加计算域，这两者有什么区别？
8. 通过数据窗口画板的设计如何实现高亮度显示当前选择的行记录？
9. 哪些风格的数据窗口比较适合为报表使用？
10. 报表中的一些敏感数据，如不及格成绩要求显著显示，如何实现？
11. 数据窗口对象与数据窗口控件有何不同？各自的作用是什么？
12. 设计一个报表，要求统计某学期各门课程的不及格人数。
13. 设计一个统计图表，要求统计某学期某门考试课的各班平均成绩。
14. 数据源指定了 DataWindow 中数据的来源和要显示的数据项。数据的来源可以有哪些？
15. 把某个控件的浏览顺序设置为 0 可以起到什么作用？
16. 数据的有效性校验是数据库应用中必须考虑的问题，在数据窗口对象的设计中如何实现数据的有效性校验？
17. 要设计一个不需要进行数据更改的 DataWindow 对象，可以通过什么途径实现？
18. 怎样使数据窗口控件与数据窗口对象相关联？怎样为数据窗口控件分配事务对象？
19. SetTransObject 函数和 SetTrans 函数的作用是什么？二者有什么区别？