

一 文档约定及名词解释	7
(一)文档约定	7
[左顶格]	7
(二)名词解释	7
单元格	7
合并单元格	7
扩展单元格	7
顶子格	7
左顶格	7
上顶格	8
当前单元格	8
指定单元格	8
单元格的扩展坐标	8
数据集	8
数据源	8
单元格的数据值	8
单元格的公式值	8
报表格式文件	9
报表参数	9
报表宏	9
二 快速入门	10
(一)e表特色	10
e表介绍	10
e表特点	11
纯.NET写的web报表工具	12
类Excel报表设计界面	13
报表绘制方式的演变	14
扩展单元格 → 报表	14
导入/导出Excel文件	15
web打印/套打	16
查询条件输入表单	18
(二)实战练习	19
三 数学模型	23
e表的数学模型没有采用常规的由一个数据集的循环为主来得到报表的思路，而是采用在一个类excel的表格基础上，通过单元格的扩展而得到整个报表的思路。	23
(一)数据模型	23
多数据集支持	23
(二)扩展模型	23
可扩展单元格	23
横向扩展	23
纵向扩展	24
不可扩展	24
默认	24
(三)顶格及子格	24
上顶格认定规则	24
左顶格认定规则	24
默认规则	25
扩展规则	25
运算规则	25
(四) 单元格表达式的规则	25
(五)单元格的扩展坐标及条件汇总	25

绝对坐标	26
相对坐标	26
条件表达式	26
几个特别用法	26
四 详细功能说明	27
(一) e表设计器	27
e表设计器的三个区域	27
第一行工具栏（系统功能区）	30
第二行工具栏(单元格的功能区)	32
(二) 打开运行报表	33
打开报表	33
运行报表	35
报表运行工具栏	35
自定义报表运行工具栏的按钮	36
自定义报表运行工具栏的样式	36
(三) 单元格	37
单元格属性	37
单元格公式	41
(四) 报表属性及打印	42
报表属性及打印	42
套打	44
(五) 数据源	45
(六) 数据集	46
生成SQL向导	48
(七) 参数和宏	57
报表参数定义	57
报表参数表单设计器	58
宏	59
(八) 报表参数表单	60
表单设计器的常用功能	60
表单设计器的控件功能	66
e表中新加的功能	78
(九) 统计图	81
(十) 子报表	82
(十一) 其它	84
选项	84
导入excel文件	84
(十二) 条形码	85
五 函数说明	87
(一) 数据集函数	87
get	87
group	88
getall	89
groupall	90
getl	90
sum	91
count	91
avg	91
max	91
min	92
last	92
first	92

cols	92
field	93
scope	93
(二)单元格函数	93
sum	93
count	94
avg	94
max	94
min	94
row	95
col	95
displayvalue	95
maxwidth	95
(三)常用函数	96
list	96
map	96
sort	96
if	96
ifnull	97
case	97
sql	97
dbsql	98
call	98
dbcall	98
ds	99
scope	99
eval	99
getmaindir	100
(四)字符串函数	100
fill	100
indexof	101
left	101
length	101
likestr	101
lower	102
ltrim	102
repstr	102
right	102
rtrim	103
substr	103
trim	103
upper	103
(五)数学函数	104
abs	104
cos	104
ceil	104
int	104
log	105
log10	105
random	105
round	105
sin	106
sqrt	106
tan	106
(六)日期函数	106

afterdays	106
afterseconds	107
date	107
datetime	107
day	107
days2date	108
daysmonth	108
daysyear	108
formatdatetime	108
hour	109
minute	109
month	109
monthbegin	109
monthend	109
now	110
prevday	110
prevmonth	110
prevyear	110
quarterbegin	111
quarterend	111
second	111
second2date	111
time	112
week	112
weekbegin	112
weekend	112
weeknum	112
year	113
(七)转换函数	113
asc	113
bigmoney	113
char	113
integer	114
isdate	114
isnumber	114
istime	114
tochinese	115
todouble	115
tolong	115
tonumber	115
tostring	116
(八)操作符	116
(九)运算优先级	118
(十)关键字	119
true	119
false	119
null	119
@value	119
@@result	119
{page_cn}	119
{page}	120
{pages}	120
{pages_cn}	120
六 学习扩展模型	121
(一)了解扩展	121

第一张扩展报表	121
横向扩展	123
(二) 数据集相关的扩展	123
简单数据列表	124
分组报表	127
交叉报表	130
多层扩展	130
(三) 扩展后的单元格	133
E3=D3+1 模式	133
E3= D3[B3:-1] 模式	136
D4=sum(D3 {D3>4000}) 模式	137
'0 模式	139
& 模式	140
\$ 模式	141
(四) 数据集公式中引用单元格	144
D3=ds2.get1(已发货,单位名称=b3)	144
(五) 经常出现的错误	144
扩展方向设置错误	144
单元格引用错误	145
七 使用技巧	147
(一) 尽量利用当前行来取值	147
(二) 利用当前组	147
(三) 权衡SQL语句的写法	147
(四) 尽量在sql 里进行group	148
(五) 尽量不用select * from	148
(六) 尽量在sql 里排序	148
(七) 尽量在sql 里过滤	148
(八) 大数据量可以采用存储过程	148
(九) or 操作符	149
(十) and操作符	149
(十一) 巧用空值判断函数ifnull	149
(十二) 空白单元格的应用	149
(十三) 慎用合并单元格	149
(十四) 慎用隐藏行列	150

一 文档约定及名词解释

(一)文档约定

[左顶格]

用来引用计算机键盘上的特殊键或引用用户界面上的某个部分，如菜单，按钮等等

(二)名词解释

单元格

由行列整齐的格子组成，这些格子我们称为单元格，所有的单元格组成了报表。和excel软件中的单元格一样。

合并单元格

我们选中一片连续的单元格区域，点击**[合并单元格]**，就把这些被选中的单元格合并成了一个单元格，这个合并出来的单元格称为合并单元格。

扩展单元格

如果单元格的值是通过返回集合值的函数表达式计算出来的，那么此单元格是可以扩展的。它就被称为扩展单元格。

顶子格

报表扩展时，如果单元格需要跟着另一个单元格的扩展而复制，那么那个扩展单元格就被称为此单元格的顶格，而此单元格被称为顶格的子格。顶格必须是可扩展单元格。

左顶格

当顶格的扩展方向为纵向时，称之为左顶格。一个单元格的默认左顶格为它的左边离它最近的、在同一行上的纵向扩展单元格。如果单元格的左顶格与默认左顶格不一致，那么需要在它的属性的**[左顶格]**属性值中指定。左顶格扩展时，其子格被纵向同步复制，且与左顶格的相对位置保持不变，子格本身的子格也将被复制。复制时，复制出来的新单元格的所有属性都引用被复制单元格的属性。

上顶格

当顶格的扩展方向为横向时，称之为上顶格。一个单元格的默认上顶格为它的上边离它最近的、在同一列上的横向扩展单元格。如果单元格的上顶格与默认上顶格不一致，那么需要在它的属性的**[上顶格]**属性值中指定。上顶格扩展时，它的子格被横向同步复制，且与上顶格的相对位置保持不变，子格本身的子格也将被复制。复制时，复制出来的新单元格的所有属性都引用被复制单元格的属性。

当前单元格

表示当前写表达式所在的单元格。

指定单元格

当表达式中含有**[]**来决定的单元格的扩展坐标时，写在**[]**前面的单元格叫做指定单元格。例如：**B5[]**中的B5叫指定单元格。

单元格的扩展坐标

因为e表是通过单元格的扩展而得到整个报表的，在报表设计时单元格是尚未扩展的，所以在报表设计时常常需要对扩展出来的单元格集进行定位，这就是单元格的扩展坐标，它分为两类：1 是绝对坐标，2 是相对坐标。

数据集

数据集其实就是定义报表数据来源的一个二维表，一般是一个SQL语句的运行结果。如果报表的数据来自文件，那么在设计时数据集是一个假想的二维表。数据集通常只保存数据，而没有样式控制等信息。

数据源

数据源是与数据库的连接，用于在报表设计器中创建数据集时选择数据库中的数据表、字段等，以及预览设计好的报表时决定从哪个数据库中读取数据。

单元格的数据值

它是指一个单元格的实际内容，单元格的数据值有以下几种类型：普通文本，图片字段，HTML内容，统计图，子报表等等。

单元格的公式值

由它进行运算后得到一个或多个单元格的数据值。

报表格式文件

在报表设计器中设计保存的文件，它定义了报表的样式，扩展公式及数据来源等，运行报表格式文件才能得到真正的报表。

报表参数

报表往往有参数，例如，日报表，当我们希望生成一张日报表时，首先需要向报表传递日期参数，然后报表才会根据我们传递进去的日期参数，生成该日的日报表。

参数有以下几个特征：

- 1、参数有数据类型
- 2、参数有缺省值

参数可以在数据集和表达式中被引用，可以在表达式中直接写参数名引用，

报表宏

宏代表没有数据类型的一个字符串，在报表运算时将用真正的宏值替换宏变量所占的位置。使用宏能使一张报表格式文件得到多张报表格式文件的运算效果。

可以在报表中任何位置引用宏变量，如单元格值、表达式、数据集定义、单元格属性表达式中等，引用方法为\${宏变量名}。

二 快速入门

(一)e表特色

e表介绍

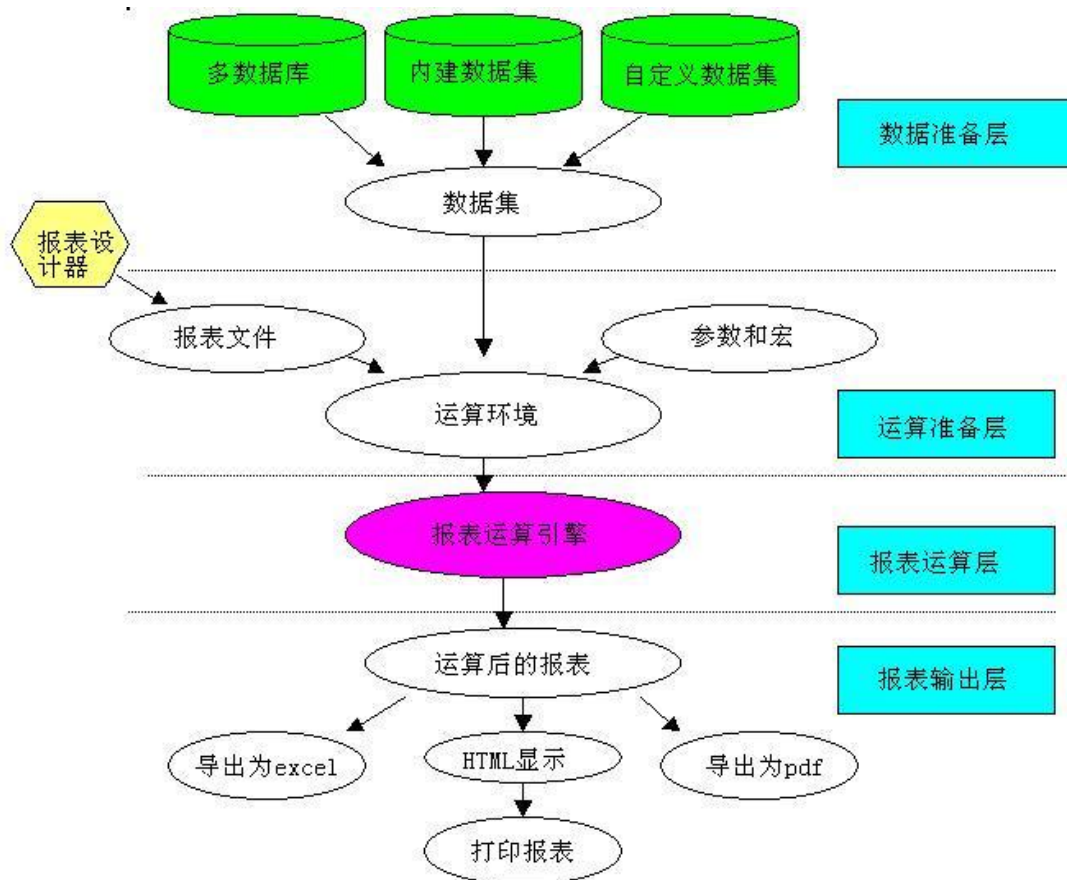
e表是一个功能强大的Web报表工具，提供了高效的报表设计方案、强大的Web报表展现能力、灵活的部署机制。使用e表可使复杂报表的设计简单化，以往难以实现的报表可以轻松实现，避免了大量的复杂SQL编写以及编程来准备数据，报表设计的效率大大提高。

e表是纯 .NET的报表工具，在.NET环境下可以无缝嵌入应用系统，因此在.NET应用中比其他非.NET的报表工具有明显优势。

用户可以通过报表设计器直接设计各种复杂格式的报表，在运行时通过报表服务器自动提取数据生成报表，可以通过Web方式展现、打印、导出。

e表内置了一个强大的表单设计器，使用它可以可视化地设计报表的查询条件的输入界面。e表的报表设计器也是在IE浏览器中直接运行的。这样用户可以很方便地将报表设计器嵌入到自己的产品中。

下面是e表系统结构图：



e表特点

e表特点

部署与集成(.NET环境)	
应用集成	纯.net应用程序,可直接嵌入用户的应用程序中
负载均衡	可以直接使用应用服务器的负载均衡体系
数据库连接	可以直接使用应用服务器的连接池
报表设计器的集成	可以将报表设计器作为一个模块直接集成到用户的应用中
activex控件	只有要精确控制客户端的打印机处要用到一个activex控件(如不需要精确打印的话,则此控件可不下载.而且此控件很小,仅75K)
Excel软件的安装	客户端和服务端都无须安装
报表设计	
报表设计器	在IE浏览器中直接运行报表设计器
编辑操作	与EXCEL处理方法类似
Excel导入	支持, 可以直接使用原来绘制好的EXCEL表格
查询条件界面	内置一个功能强大的可视化的表单设计器,可以很方便地设计功能复杂的查询条件输入表单
报表格式文件的格式	在报表设计器中设计的报表文件采用html语言的格式,这样可以充分利用上html的强大的样式功能,因为html语法众所周知,所以也大大地降低了学习成本
报表格式文件保存到数据库中	支持
自定义数据集	支持
多数据源多数据集	支持
内建数据集	支持
利用宏替换来扩充报表格式文件	支持
支持的报表种类	
简单的数据列表	支持
分组报表	支持
交叉表	支持, 且与其他报表类型是一致的, 可以混合和进行各种变化
超级链接	支持
公式计算	支持

根据条件改变格式	支持
斜线	支持
统计图表	支持
图片字段展现	支持
横向展开	支持
不规则分组	支持，在报表中即可轻松实现
多表数据	支持
主从报表	无须子表概念即可完成
子报表	支持，多层任意，格线可对齐可缩放，横纵向均可自动摆位
跨行组运算	支持，提供了强有力的扩展坐标，可以任意进行计算
不规则的分片报表	支持，上下格式可不一致，固定变动混合，直接运算性能高
可重复划分	支持
跨表取数	支持
报表展现	
HTML方式展现	支持
精确打印	支持
导出至PDF	支持
导出到Excel	支持,而且在客户端和服务端都无须安装Excel软件的情况下实现数据和样式不失真地导出到Excel
套打	底图描绘方式，轻松简单
多栏	支持
打印分页	横纵向均可，强制分页,按公式条件分页
其他打印控制	一纸多页，补空行

纯.NET写的web报表工具

目前，.net环境下能够使用的web报表工具主要有以下两类：

1 水晶报表，它集成在VS.NET开发环境下，成为最普及的报表工具。但水晶报表仍有这样或哪样的不尽如意的地方。在此就不详述了。

2 activeX控件式，它最典型的代表有cell插件和如意报表。

以上两类都能应用在 .net程序中，但都不是用纯 c# 或VB.net等语言写的。这样在和.net的web应用程序集成及部署时会有诸多不便。

随着 .net 应用程序越来越普及，迫切需要有纯 .NET写的web报表工具。e表3.0正是这样的一个web报表工具。它的服务器端全部是用c# 语言实现的。可以自然地与.net应用程序集成。可以和用户的应用程序共用数据库连接。提供丰富的编程接口供用户全面控制每一个细节。

类Excel报表设计界面

报表绘制的方法一般有 **网格格式** 和 **控件拖拽式** 两种，前者采用和EXCEL类似方案，用网格线围出报表；而后者则是用矩形框等元素拼出报表。

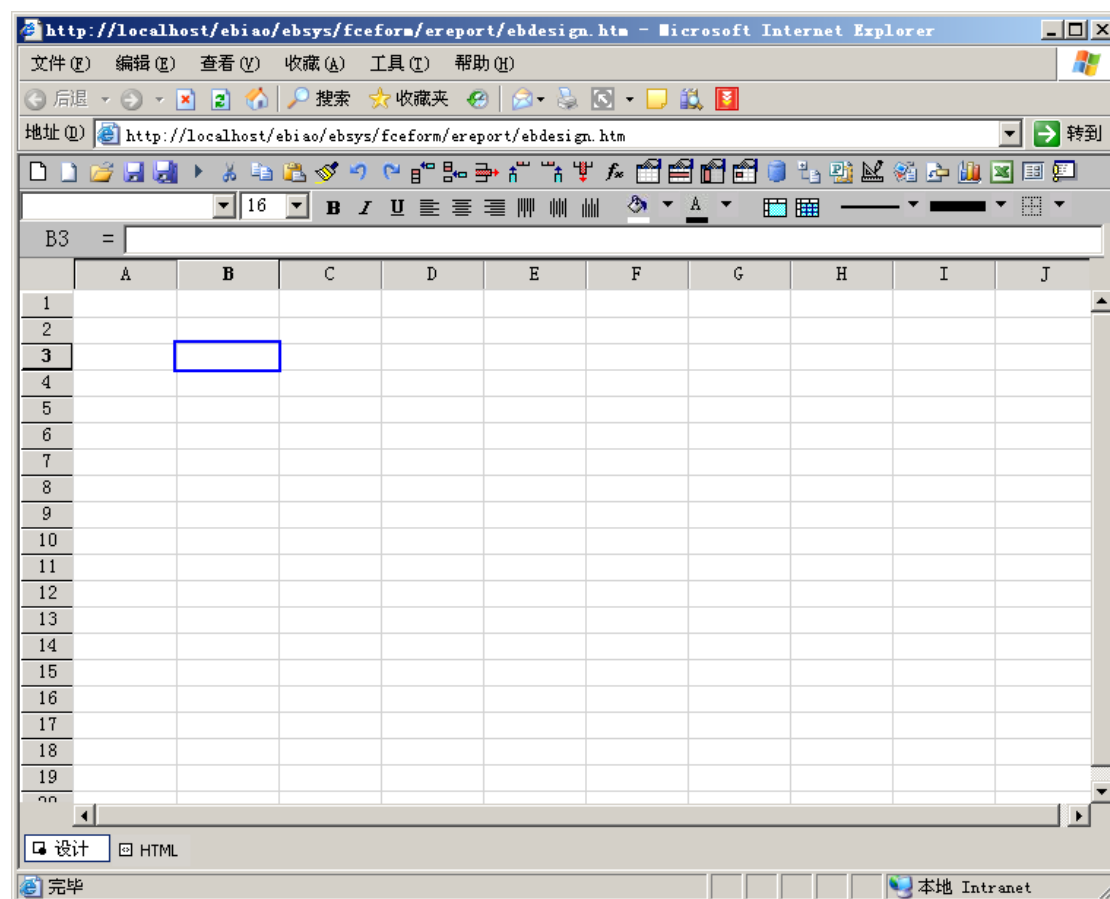
国内报表对样式要求非常复杂而且严格，绘制方案是报表工具相当重要的指标，它决定了报表样式的复杂度和绘制效率，特别是对于格线比较多的表。

传统的报表绘制，大多数是用的拖拽式，拿部件拖来拽去。表格线需要靠矩形边框重合摆放拼接出来，常见的复杂多层表头画起来非常繁琐，数据区某些纵向合并格甚至无法绘制出来(或者需要特定程序控制才能实现)，拼出的格线还与分辨率相关，屏幕上对齐的表在打印时或在WEB上显示时又可能不整齐。

网格格式在这方面有明显的优势，能够画出样式很复杂的报表，绘制效率也比控件拖拽式高出许多。由于MS Excel软件的普及，在大多数人心中画表软件就是Excel，对于Excel软件的操作也都很熟悉。这样也能节省许多学习成本。

所以，类EXCEL的方法明显优于拖拽式，或者说，画报表就应该用象Excel那样的方法。大概稍有点常识的人，都会拿Excel画表吧。所以，类Excel是必然的方向。

e表正是采用类Excel(**网格格式**)的报表绘制方案，没有采用**控件拖拽式**。



报表绘制方式的演变

因为Excel很早就已普及了。所以提及报表工具，自然首先都会想到用类Excel方式，而为什么控件拖拽式的报表工具会这么多呢？

这是因为Excel虽然具有强大的绘表能力，但数据处理能力非常弱，除非用“A2+B3=D4”这样的公式将单元格联系起来，否则单元格之间是相互独立的，没有任何关系的。如果要从数据库或数据文件中取数，需要程序员一个单元格一个单元格去定义，工作量相当繁重。需要靠程序员编程来实现动态行列报表的制作。

因为报表格式只是报表工具要解决的一部分问题。我们用报表工具的目的，是要能从数据库中读出数据，从而自动产生报表，数据才是报表要解决的核心问题。报表格式用Excel可以解决得很好，但是Excel没有良好的从数据库中进行数据汇总方案，除非编程序

往格子里填数据，基本上没法从数据库中读出数据自动产生报表。因此这种报表几乎都只能做静态报表。

而拖拽式的格式虽然解决得一般，但从数据库中取数进行数据统计能力方面要比Excel强得很多，两害相权取其轻，堆框虽费劲，慢慢堆就行了，但统计汇总不行的话那就没办法了。

报表工具软件和其它软件一样都是国外先做出来的，而国外的报表的样式非常规整，没有格线，表头非常简单，没有斜线表头、没有分层分组。一张报表提供的信息有限，如果要看所有信息，就只能几张表对照着看了。因而国外的报表工具采用控件拖拽式能弱化报表样式绘制能力弱的缺点。而国内的报表工具基本上都是学习国外的报表工具模型来设计的。而中国式报表的样式相当复杂，所以很多报表工具都被迫再辅以大量的程序代码来处理中国式报表，这也使得用户在买了报表工具后，做报表的工作量还是很大。

扩展单元格 → 报表

Excel绘表方式能解决报表样式问题，但Excel的这种基于自由单元格的模型和基于二维表的数据库物理表模型没法有机的结合起来。控件拖拽式绘表能很好地从数据库中取数，但又无法处理复杂的报表样式。因而迫切需要一种新的报表实现方式。

e表就是采用了一种新的报表实现方式。在e表中，一个报表被认为是由单元格扩展而来的。即先采用类Excel方式绘制好静态报表。当然在其中预置了一些扩展单元格的公式。而在运行报表时根据公式横向或纵向扩展单元格而得到真正的报表结果。这样就同时兼顾了报表样式和从数据库中自动取数这两个方面。

控件拖拽式报表工具除了样式绘制麻烦之外，还有它最终是由单个数据集循环展开而得到报表的致命弱点，报表的形成是由单个数据集的循环再结合报表格式文件的定义而得到真正的报表结果，它相当于只有一个循环运算点。

而e表因为可以在一个类Excel的静态网格报表上定义无限多的扩展公式，运算报表时这些扩展公式同时扩展，每个扩展公式可以基于不同的数据集。它相当于有无限多个循环运算点。因此报表的表现力大大增强，可以说是复杂统计报表的克星。

	A	B	C	D	E	F	G	H	I
1									
2				按年份汇总		已发货	已付款	=dsl.group	
3				=dsl.group	小计:			=dsl.group	小计
4	=dsl.group	=dsl.group		=dsl.sum	=sum(D4	0	0	=dsl.sum	=sum(H4
5	(sperson, f		小计:	=sum(D4	=sum(D4	0	0	=sum(H4	=sum(H4
6			合计:	=sum(D4	=sum(D4	0	0	=sum(H4	=sum(H4

运行报表后得到如下结果:

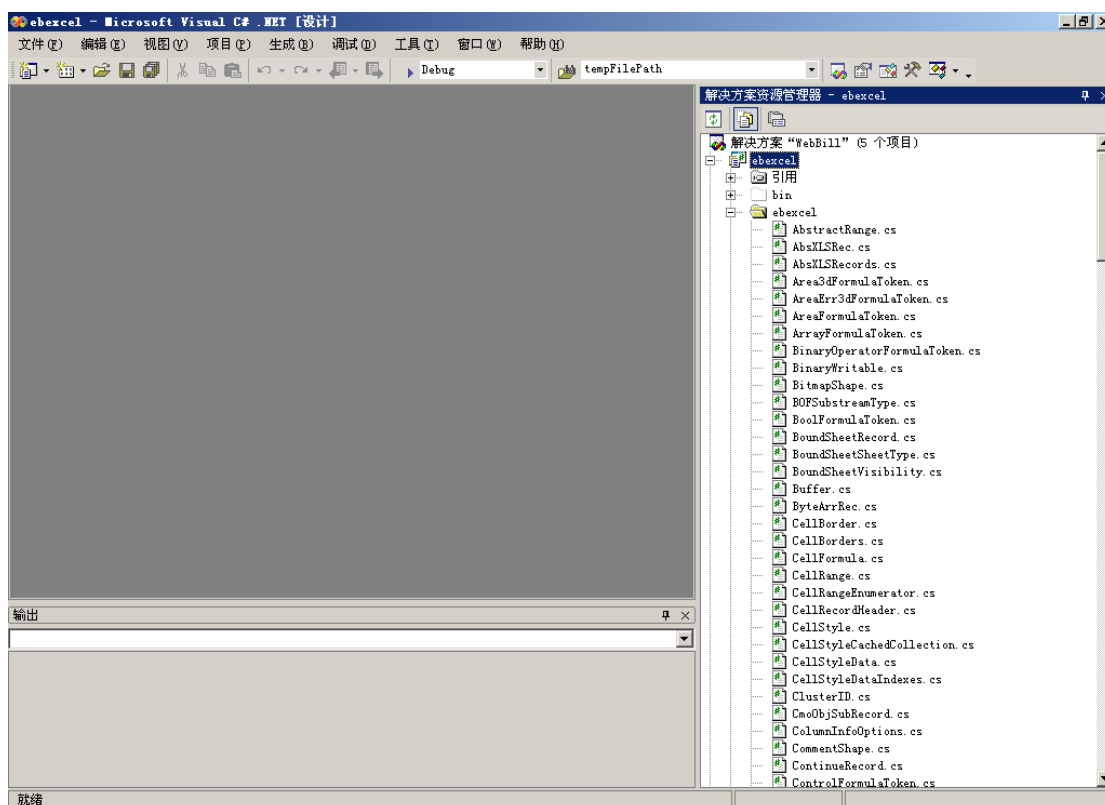
		按年份汇总			已发货	已付款	上海				北京			
		2005	2006	小计:			打印机	电脑	电视	小计	打印机	电脑	电视	小计
林黛玉	HP公司	0	900	9000	0	0	0	0	0	0	900	0	0	900
	方正集团	5100	3800	89000	0	0	0	0	0	0	0	3300	1800	5100
	小计:	5100	4700	98000	0	0	0	0	0	0	900	3300	1800	6000
薛宝钗	北京四通公司	1700	0	17000	0	0	0	0	0	0	0	0	1700	1700
	小计:	1700	0	17000	0	0	0	0	0	0	0	0	1700	1700
贾宝玉	HP公司	0	6790	67900	0	0	5800	0	5800	0	0	0	0	0
	北京方成公司	3600	3200	68000	0	0	0	0	0	0	0	3600	0	3600
	方正集团	0	1100	11000	0	0	0	0	0	0	0	0	0	0
	联想公司	4300	1200	55000	0	0	0	2300	2300	0	0	0	2000	2000
	小计:	7900	12290	201900	0	0	5800	2300	8100	0	0	3600	2000	5600
贾政	创维集团	0	1800	18000	0	0	0	0	0	0	0	0	0	0
	北京方成公司	0	5850	58500	0	0	0	1650	1650	0	0	0	0	0
	思创公司	0	3900	39000	0	0	0	0	0	0	0	0	0	0
	小计:	0	11550	115500	0	0	0	1650	1650	0	0	0	0	0
贾琏	HP公司	0	800	8000	0	800	0	0	800	0	0	0	0	0
	北京方成公司	2500	0	25000	0	0	0	2500	2500	0	0	0	0	0
	方正集团	0	9490	94900	0	1290	4900	0	6190	0	0	0	0	0
	小计:	2500	10290	127900	0	2090	4900	2500	9490	0	0	0	0	0
合计:		17200	38830	560300	0	2090	10700	6450	19240	900	6900	5500	13300	

导入/导出Excel文件

因为Excel的易用，使得用户能够自己做表，几乎所有用户的报表都有Excel文件。用户希望能将Excel的文件读入报表工具，这样就可以省去画表的工作了；同时，生成的报表，要能导出成Excel文件，这样用户就可以在其上再加工处理。因而对于一个报表工具来说，有没有完善的导入/导出Excel文件的功能是至关重要的。

在.net开发环境下，实现导出Excel文件有如下三类方式：1 只导出数据，不导出格式。生成csv的文件。2 利用Excel软件本身带的com控件来实现。这种方式的弱点是需要安装Excel软件，且可能有Excel软件的各个版本之间的兼容性问题。还有就是Excel软件中的com控件的庞大的接口需要学习。3 自己写代码实现标准Excel格式文件的二进制层面的读写。

e表就是采用第3种方式。e表系统中有一个用c#写的ebexcel.dll来实现标准Excel格式文件的二进制层面的读写。这样做的好处是在客户端和服务端都不用装excel软件的情况下实现Excel文件的不失真读写。



web打印/套打

自从开发web程序以来，web打印就一直是困扰大家的大问题。基于web的打印，难度在于要将浏览器中呈现的html，精确地打印到票据中，而且能够实现对分页位置的控制。下面就IE浏览器所能采用的打印解决方案，来个汇总分析。

一、浏览器的打印功能菜单
这种方案的优势是不需要对浏览器作任何扩充，是最简单的办法，但问题也最多，如：

1 不能精确分页。

浏览器一般是根据用户设置的页面大小，web页面的内容多少，来自行决定分页位置，程序员很难控制。会有页脚页眉干扰。

2 不能准确地控制边距及打印文字。

3 不能解决连续打印。

比如，不是仅打印一张票据，而是连续一次打印若干个票据。

二、使用webbrowser控件+ javascript
这实际上，是浏览器打印功能菜单的一种程序调用，与打印功能菜单没什么两样。分页的问题仍然存在，只不过，可以让用户不用去点菜单，直接在网页中的一个按钮，或一个链接里面调用罢了。

三、使用print css

这是一种最理想的实现web套打的方法。这种方法通过在html文档中，嵌入打印相关的css样式，来实现对html文档输出打印的控制，比如设置纸张大小，纸张纵横方向，打印边距，分页等。显而易见，这种方式成本小，不需要下载任何插件，而且跨平台性非常好。print css推出已经有些时日，但遗憾的是，至今没有一个厂商的浏览器很好地实现了这些标准，这使得程序员目前还不能利用print css进行实际的开发。关于打印css，参见：

<http://css-discuss.incutio.com/?page=PrintStylesheets>

四、使用pdf文件
用这种方式，就是从服务器端下载一个pdf文件流，在IE中用adobe插件打开，然后用adobe的打印菜单进行打印，虽然这种方案，也能实现精确套打，但需要下载adobe插件。这是国外报表工具经常推荐的一种打印方法，但在pdf不那么普及的中国，这种方案不是最

好选择。

五、采用纯ActiveX

这种方案就是下载一个控件，票据的数据不再以html方式呈现，而是呈现在ActiveX中。这种方案的优点是打印的精确度高，分页的可控性好，但缺点也是很明显的，嵌入ActiveX控件破坏了web应用的整体html风格，且这样的控件比较大（一般超过1M，下载颇费时间）。

六、

采用Applet方式

采用Applet方式，分页或精确打印，都可以做到完美，但缺点也很明显，表现在：

- 1 安装 Applet 成本巨大。需要下载十几 M 的文件。Applet 本身可能并不大，但运行 Applet 所需的 jre 一般至少 10 几 M（jre1.4.2，15.45M）。用户需要极大的耐心，来进行打印。
- 2 打印报表时，需要重新向服务器检索数据，效率低。因为 Applet 方案，一般采用 html 方式呈现数据，打印时 Applet 必须向服务器检索同一张票据的数据，看上去，是打印了当前页的票据，实际上，Applet 根本不会用当前 html 页的数据来打印，而是向服务器下载数据到 Applet 中来打印。也就是说，打印的话，必须两次请求，一次 html 呈现，一次用来打印。

七、轻量级的ActiveX打印方式

轻量级的ActiveX打印方案的优点是：

1. 效率 高；
可以直接打印指定的html文档，不必向后台再次提起请求。
2. 能做到精确打印，分页；
3. 功能 丰富，
可以利用该控件，实现对当前票据文档的打印，预览，及多个html文档的连续打印。
4. 下载量小，ActiveX只有75K。

e表就是采用第七种方式来实现报表的打印。e表内置了一个用c++写的轻量级的ActiveX控件来精确地控制客户端的打印机。一举解决web打印问题。

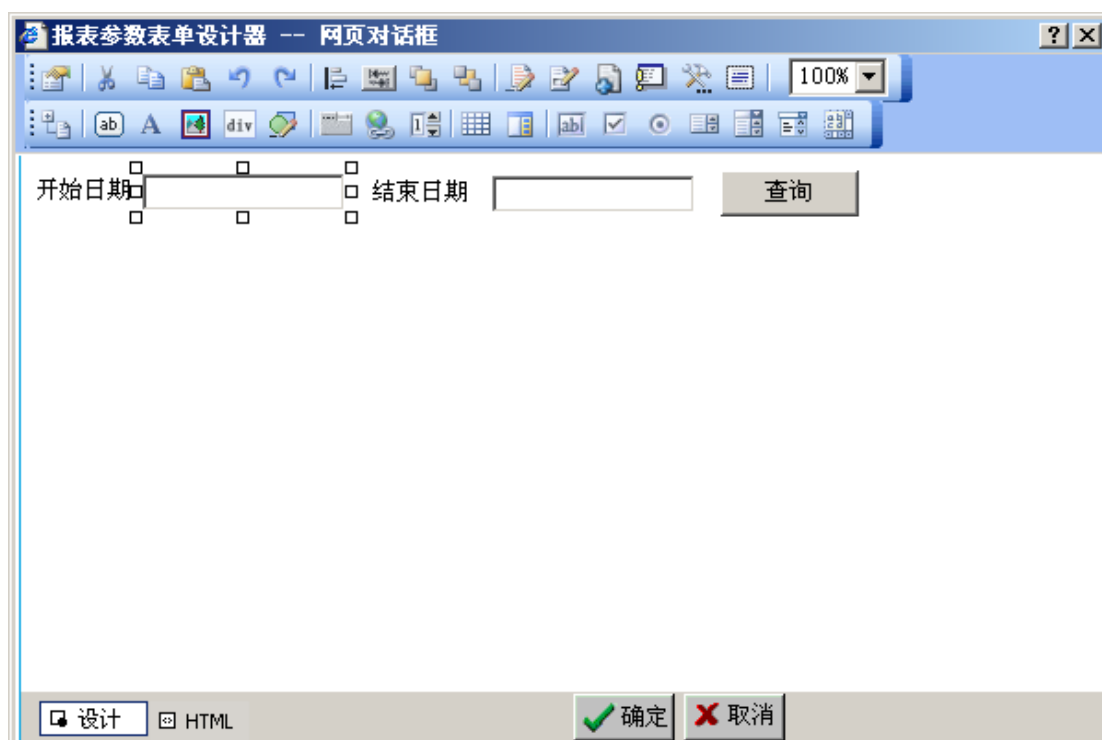
方成公司eprint打印预览演示版					
打印P 页面A 1 共 1 100% 帮助H 关于E 关闭C					
职员内码	职员姓名	性别	出生日期	email	电话
ZCA00000106	刘欣娜	女	1985-05-12	zhangfei@hotmail.com	0312-4365760
ZCA00000041	刘欣好	女	1985-05-12	zhangfei@hotmail.com	0312-4365760
ZCA00000042	系统管理员	女	1996-06-08	emailqqq	2452345234
ZCA00000043	姜和		1923-01-02	lkcd@hotmail.com	010-82746545
ZCA00000044	米丹丹		2002-06-08	23@sian.com	877788
ZCA00000045	洪成海	女	1978-07-13	goodworker@hotmail.com	098766555
ZCA00000046	赵朝阳		1973-07-13	goodworker@hotmail.com	98766555
ZCA00000049	徐下下	女	1978-07-13	goodworker@hotmail.com	098766556
ZCA00000060	郭晶	男	1974-05-21	Richard@hotmail.com	0754-2584525
ZCA00000061	万成	男	1975-04-11	ELlen@sowant.com	021-2546525
ZCA00000062	王要	男	1975-10-21	xinmian7510@sina.com	091-25345255
ZCA00000063	李东雪	女	1976-02-21	dongxue76@sohu.com	010-85345255
ZCA00000067	白小丽	男	1978-10-05	xiaoli78@sohu.com	010-85287955
ZCA00000069	夏修宜	男	1976-07-13	xiuyi76@Micro.com	010-85241955
ZCA00000070	上官平	男	1977-08-26	pinger76@Micro.com	010-85264955
ZCA00000054	韩平				
ZCA00000088	刘欣	男	1985-05-12	zhangfei@hotmail.com	0312-43651760
ZCA00000104	刘欣娜a	女	1985-05-12	zhangfei@hotmail.com	0312-4365760

查询条件输入表单

一般来说，大多数的报表都需要根据输入的报表参数而得到不同的结果。例如：日报表需要根据输入的日期来得到到底是要哪一天的日报表。这时就需要日期的输入界面，它被称为是查询条件的输入表单。

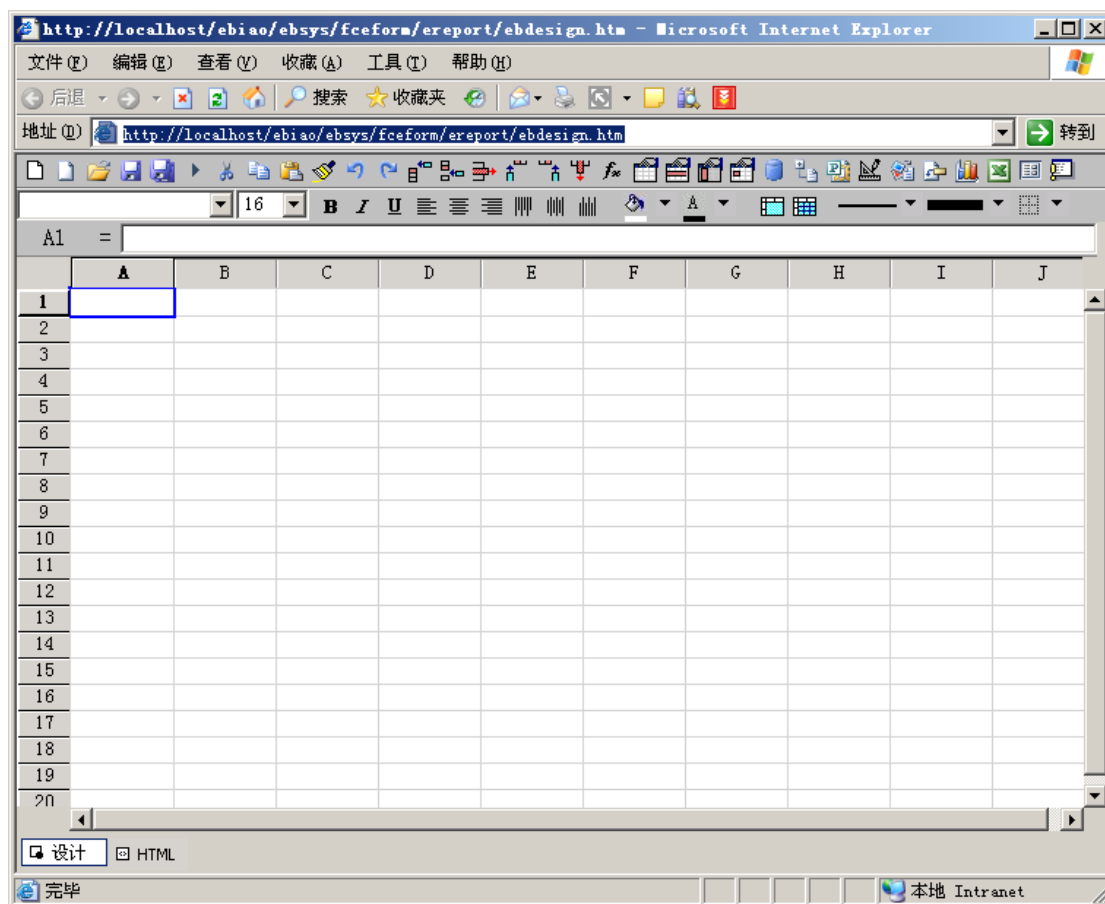
查询条件的输入界面是千变万化，没有一定之规的。很多报表工具只是提供一个自动生成这个界面的功能。可想而知，自动生成的界面往往无法满足用户的多样化的需求。甚至于有的报表工具就没有此功能，由用户用程序写去。而查询条件输入表单是报表密不可分且又不可或缺的一部分，如果查询条件输入表单没做好的话，往往会造成用户在使用了报表工具后还是需要大量的编码工作。

e表内置了强大的eform可视化自定义web表单工具的主要部分，由它来解决查询条件输入表单的制作问题。它提供了下拉列表，页签控件等丰富的页面控件，完整的数据验证机制，也可以通过自定义事件来实现复杂的控制。通过将输入控件和报表参数进行绑定来将查询条件的输入表单和报表联系起来。它采用可视化的设计界面，既可以通过简单的控件拖拉绑定实现简单的查询条件输入表单，又可以通过写事件的代码来实现复杂的查询条件输入表单。一举解决了查询条件输入表单的多样化问题。



(二)实战练习

1 在IE的地址栏上输入 <http://localhost/ebiao/ebsys/fceform/ereport/ebdesign.htm> 即可进入如下图所示的编辑界面：



2 在工具栏上点[数据集设置]图标进入数据集设置界面：



在此界面上点[添加]按钮，增加一个数据集，在 SQL语句中填写如下内容：

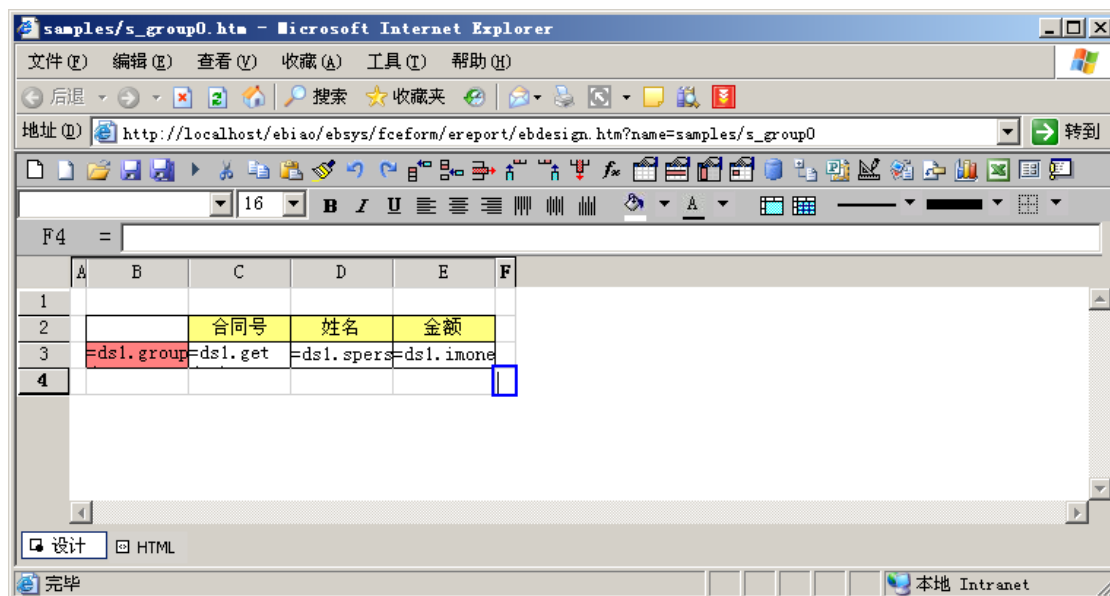
`select sarea,id,sperson,imoney from e_group`

得到如下的界面图：



然后点[确定]按钮返回。

3 直接在单元格中填写内容：即 C2单元格中填写：合同号，D2 = 姓名，E2 = 金额，B3 = `=ds1.group(sarea,false)`，C3 = `=ds1.get(id)`，D3 = `=ds1.sperson`，E3 = `=ds1.imoney`，然后设置一下这些单元格的对齐方式，背景色等样式。得到如下的界面图：



4 在工具栏上点[保存]按钮，输入一个报表文件名，如 test，就将报表格式文件制作好了。

5在工具栏上点[运行报表]按钮，可以看如下图的运行结果：

	合同号	姓名	金额
上海	12	贾璘	4900
	5	贾宝玉	2300
	9	贾政	1650
	7	贾璘	1290
	17	贾宝玉	5800
	16	贾璘	800
	4	贾璘	2500
北京	18	林黛玉	900
	11	林黛玉	3300
	10	贾宝玉	3600
	3	薛宝钗	1700
	2	林黛玉	1800
	1	贾宝玉	2000
天津	20	贾璘	3300
山东	6	贾宝玉	1200
	8	贾政	4200
	15	贾政	1800
江西	14	林黛玉	3800
	19	贾宝玉	1100
	13	贾政	3900
河北	22	贾宝玉	990
	21	贾宝玉	3200

5 这样就得到了一个以 地区字段分组的统计报表。要运行此报表，可以通过在IE地址栏上输入

<http://localhost/ebiao/ebsys/fceform/ereport/ebrun.htm?file=/ebiao/ebsys/ebfile/test.htm>

来运行，其中 test 为报表文件名。

三 数学模型

e表的数学模型没有采用常规的由一个数据集的循环为主来得到报表的思路，而是采用在一个类excel的表格基础上，通过单元格的扩展而得到整个报表的思路。

(一)数据模型

多数据集支持

往往同一张报表的数据来自多个物理表，甚至是多个物理库；而且报表的上下或左右或同时被分成多片，每片规则不一致，且又可能同时参与运算，因此决定了报表的数据来源必须是多个数据集，这多个数据集同时在一个报表上按某种规则扩展，而且多个数据集分别直接用来参与报表运算，而不是合并成一个以后再参与运算。

(二)扩展模型

可扩展单元格

当单元格的数据值表达式包含集合函数、to操作符、: (link)操作符、数组时，该单元格默认为可扩展单元格，此时您可以把单元格的扩展方向属性设为默认、横向扩展或者纵向扩展。

如果单元格的数据值表达式不包含上述元素，则该单元格默认为不可扩展。

集合函数包括：group(), groupall(), get(), getall(), list(), sql(), call()

to 操作符举例：=1 to 10

: 操作符举例：=a1:a10

数组举例：=[1,3,5,7,9,11]

横向扩展

当单元格为可扩展单元格时，可以把该单元格的扩展方向设为横向扩展，此时该单元格会横向进行复制，复制单元格的数据值依次为表达式的结果数据值，表达式返回几个值，单元格就复制几个，这称为单元格的横向扩展。

单元格横向扩展时，此单元格的上方单元格缺省为它的上顶格，下方单元格缺省为它的子格；如果上方没有横向扩展单元格，则上顶格缺省为00格；缺省情况可以人为通过设置来改变，例如：您可以把下方单元格设置为当前单元格的上顶格。

单元格进行扩展时，主单元格或同级别单元格被拉高（纵向扩展时）或拉宽（横向扩展时），从单元格被复制；复制时，复制出来的新单元格的所有属性都引用被复制单元格的属性。

纵向扩展

当单元格为可扩展单元格时，可以把该单元格的扩展方向设为纵向扩展，此时该单元格会纵向进行复制，复制单元格的数据值依次为表达式的结果数据值，表达式返回几个值，单元格就复制几个，这称为单元格的纵向扩展。

单元格纵向扩展时，此单元格的左方单元格缺省为它的左顶格，右方单元格缺省为它的子格；如果左方没有纵向扩展单元格，则左顶格缺省为00格；缺省情况可以人为通过设置来改变，例如：您可以把右方单元格设置为当前单元格的左顶格。

单元格进行扩展时，主单元格或同级别单元格被拉高（纵向扩展时）或拉宽（横向扩展时），从单元格被复制；复制时，复制出来的新单元格的所有属性都引用被复制单元格的属性。

不可扩展

当单元格里表达式的计算结果值是单值时，即不是数组或集合值时。该单元格默认为不可扩展的单元格。不可扩展的单元格不能缺省做顶格。

默认

当单元格的[扩展方向]属性为默认时，则遵守如下规则：

- 单元格的表达式为单值表达式时，该单元格默认为不可扩展单元格
- 单元格的表达式为集合表达式时，该单元格默认为可扩展单元格
- 可扩展单元格的顶格为行首格或列首格时，该单元格默认为纵向扩展
- 可扩展单元格的上顶格横向扩展时，该单元格默认为横向扩展
- 可扩展单元格的左顶格纵向扩展时，该单元格默认为纵向扩展。

(三)顶格及子格

上顶格认定规则

- 上顶格的认定规则是向上找
- 首先检查当前单元格是否设了[上顶格]属性，如果设了，就是设定的值
- 如果没设，就检查上方单元格是否是横向扩展单元格，如果是，上方单元格就是当前格的上顶格；如果不是，就检查上上单元格是否是横向扩展单元格，如果是则为它的上顶格，否则继续按此规则往上找。
- 如向上一直找不到横向扩展单元格，则上顶格为00格。

左顶格认定规则

- 左顶格的认定规则是向左找
- 首先检查当前单元格是否设了[左顶格]属性，如果设了，就是设定的值
- 如果没设，就检查左方单元格是否是纵向扩展单元格，如果是，左方单元格就是当前格的左顶格；如果不是，就检查左左单元格是否是纵向扩展单元格，如果是则为它的左顶格，否则继续按此规则往左找。
- 如向左一直找不到纵向扩展单元格，则左顶格为00格。

默认规则

- 单元格横向扩展时，上方横向扩展单元格缺省为它的上顶格，下方单元格缺省为它的从单元格；如果上方没有横向扩展格，则上顶格缺省为00格。
- 单元格纵向扩展时，左边纵向扩展单元格缺省为它的左顶格，右边单元格缺省为它的从单元格；如果左边没有纵向扩展格，则左顶格缺省为00格。

扩展规则

单元格进行扩展时，主单元格或同级别单元格被拉高（纵向扩展时）或拉宽（横向扩展时），从单元格被复制；复制时，复制出来的新单元格的所有属性都引用被复制单元格的属性。

运算规则

单元格扩展时，从单元格可以动态取主单元格的值；随着主单元格扩展，从单元格被复制到不同的位置，对主单元格的引用也相应的变化。

例：`ds1.get(col1,col2=A2)`（其中A2为扩展顶格）

单元格扩展时，从单元格如采用统计类表达式，则缺省统计此顶格扩展出来的所有单元格的值。

(四) 单元格表达式的规则

用[]来定义单元格的扩展坐标

所有条件表达式都是用{}括起来的

条件表达式的返回值为true或false

如{}之间不写任何内容则表示没加任何限定条件,即取{}前定义的所有单元格集。

如{}之间定义了条件表达式,则表示为符合条件的单元格集

(五)单元格的扩展坐标及条件汇总

在进行报表设计时,单元格尚未进行扩展,但是单元格的表达式往往需要对扩展后的单元格进行运算。

绝对坐标

用一个绝对坐标来唯一标识扩展后的单元格。例如: **C2[A2:2, B2:1]{}** 表达式中, **A2:2**表示A2扩展后的第二个单元格,即扩展后的a7, **B2:1**表示a7分组区域内, **B2**扩展后的第一个单元格,即扩展后的b7,因此整个表达式表示c7,c8,c9三个单元格。

相对坐标

在报表运算时常常需要取上一行,上几行,上一列之类的的数据,这就要用到相对坐标。例如: **D3+E3[-1] E3[-1]**表示取E3扩展出来的上一个单元格的值。

条件表达式

在报表运算时常常需要取某个单元格扩展出来的单元格中赋合指定条件的部分单元格。例如: **avg(C1{C1<60})**表示取c1扩展出来的值小于60的所有单元格的平均值。

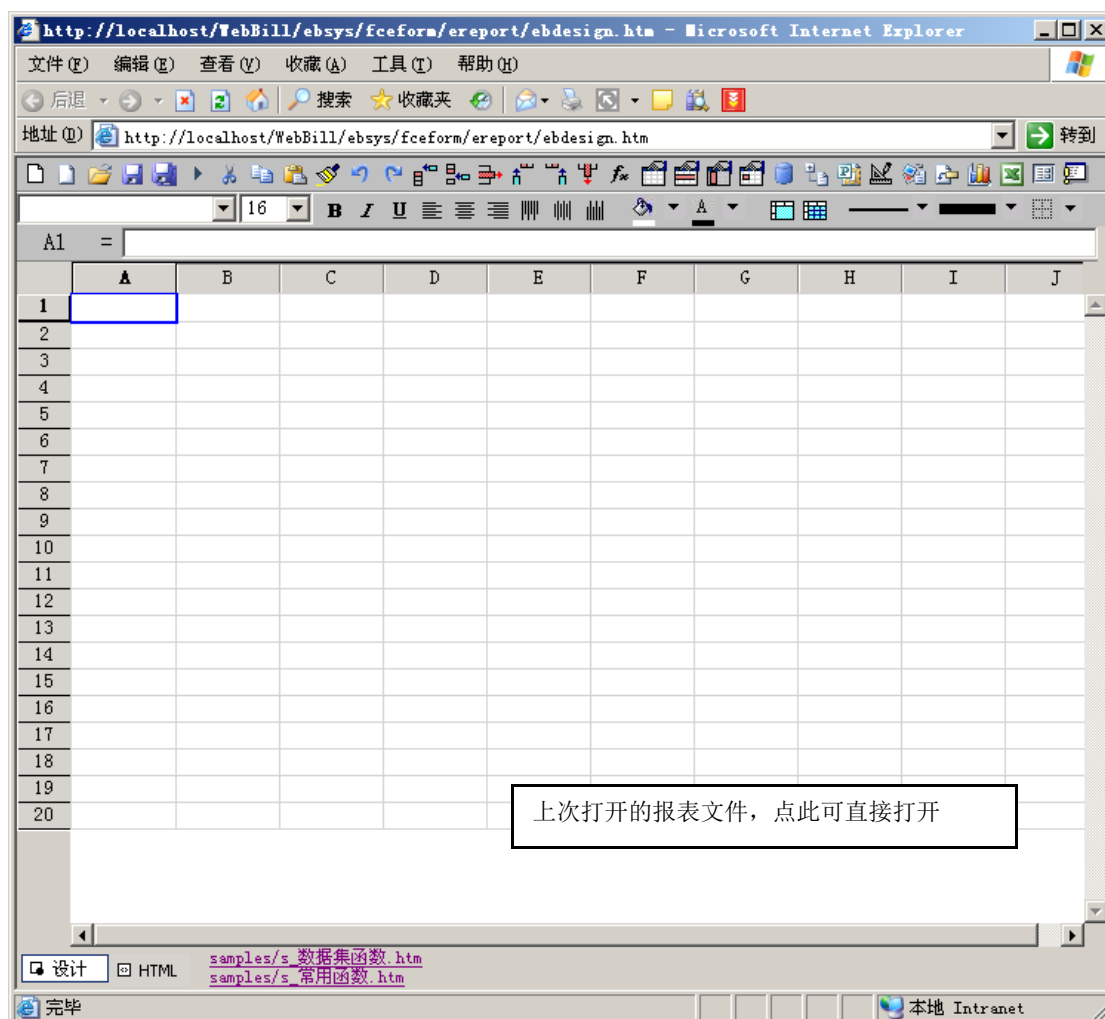
几个特别用法

- &运算符可以获得单元格位于某个顶格中的位置
- \$运算符当前单元格的指定顶格的数据值
- `0返回00格下所有指定单元格的集合

四 详细功能说明

(一)e表设计器

下图是e表设计器的界面。



e表设计器从上到下分为三个区域：

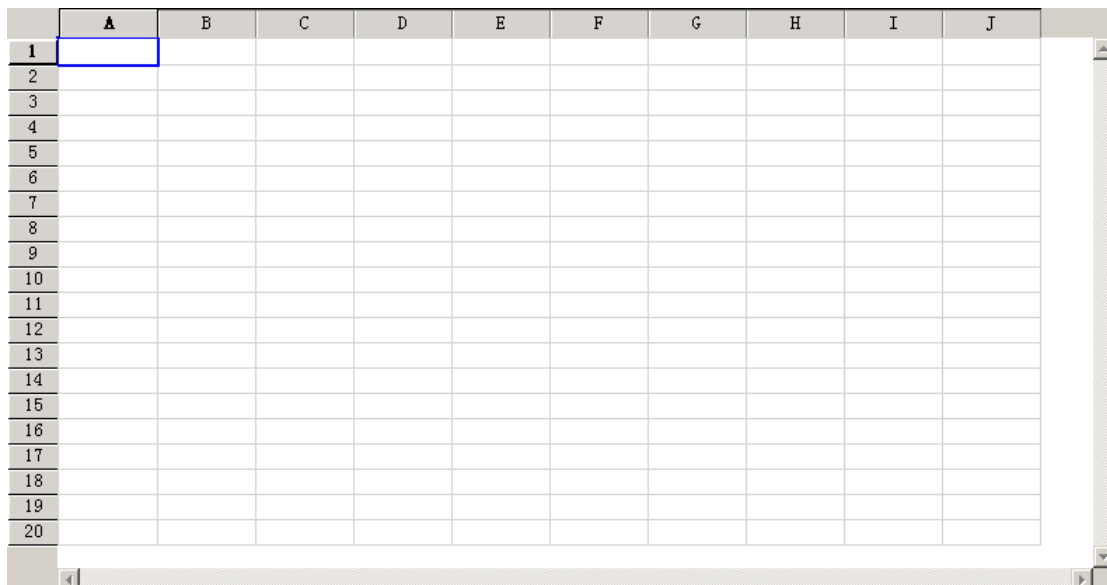
e表设计器的三个区域

- 工具栏区域

共有三行工具栏，第一行工具栏为系统功能区，如[打开报表]，[保存报表]等；第二行工具栏为设置单元格的功能区，如设置单元格的字体，字号等。第三行工具栏只是用于显示或编辑当前单元格的值。

• 内容区

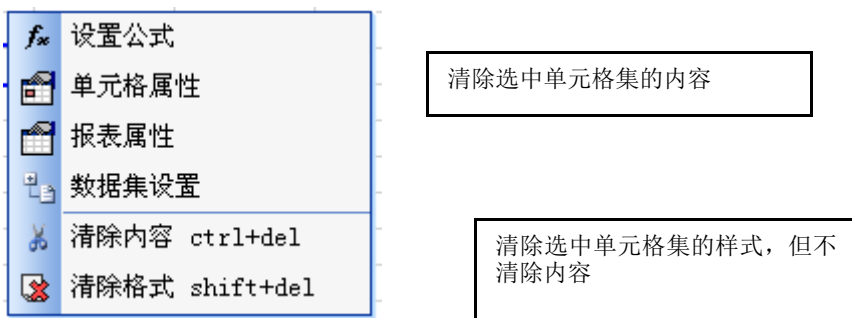
一个行用1,2,3,4...等等数字标识，列用A,B,C,D...等等字母标识的网格区域。和EXCEL类似。可以在上面直接编辑报表的内容。如下图:



单元格是通过列标+行标的方式来标识的，如上图选中的单元格为A1；当内容超过窗口大小时可以通过横向和纵向滚动条来查看内容。如要调节某列的列宽可以将鼠标移到标题行中，当鼠标变为其它形状时，拖动鼠标即可调整列宽，调整行高和此类似。

要改变单元格的内容可以直接选中单元格，然后在上输入即可。

在内容区中按鼠标右键会出现如下图的右键功能菜单：

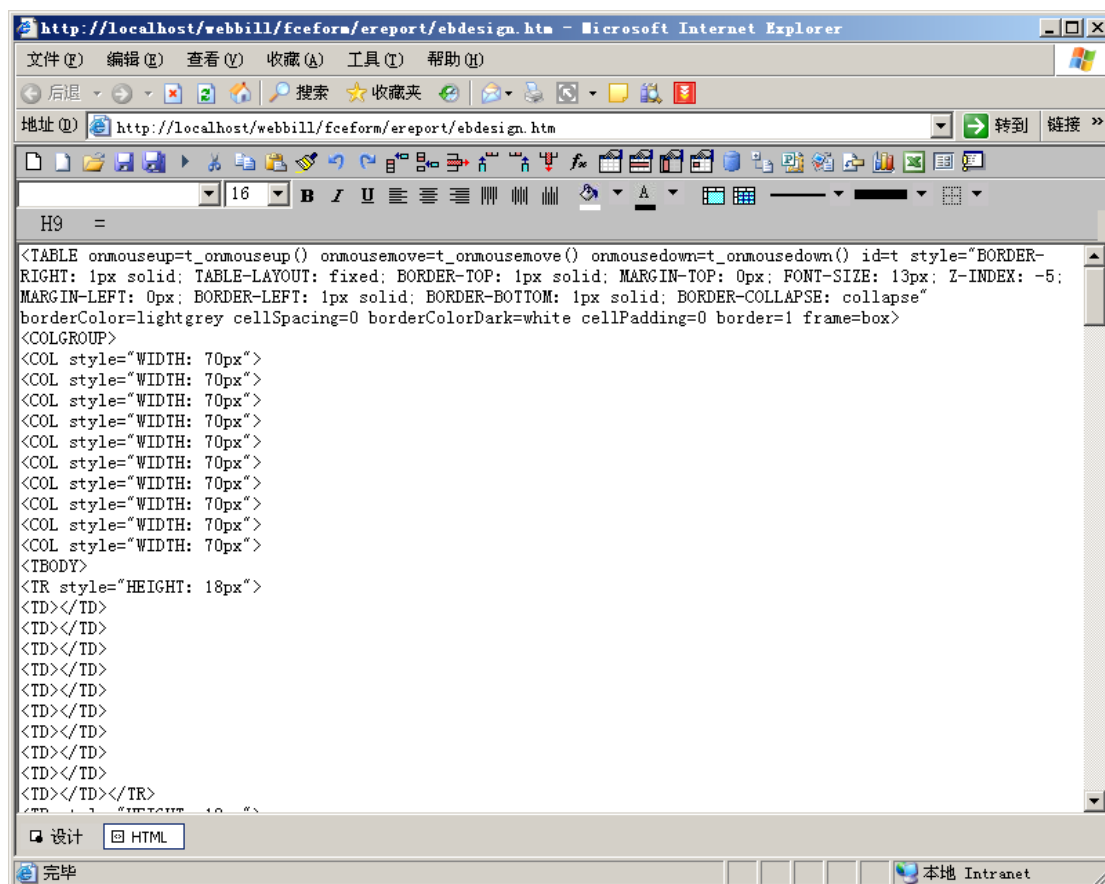


• 设计/HTML切换区

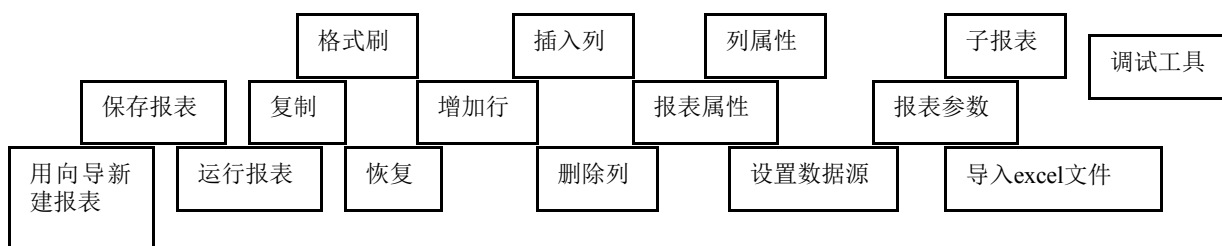


由此可以切换到HTML代码编辑状态。在HTML代码编辑状态中可以通过直接手写HTML代码来改变报表。

点击HTML代码后，如下图:



第一行工具栏（系统功能区）



- **新建空白报表**

按默认的数据新建一张空白报表，如当前报表尚未保存，则会提示是否保存当前报表。

- **用向导新建报表**

根据向导界面上的配置来自动生成报表。

- **打开报表**

通过一个选择报表的界面来选择一张报表，然后将选择的报表在报表设计器中打开。

- **保存报表**

保存当前设计器中打开的报表。

- **另存报表**

将当前设计器中打开的报表另存为另外的报表。

- **运行报表**

运行当前报表

- **剪切**

剪切当前选中的单元格集的内容


- **复制**

复制当前选中的单元格集的内容

- **粘贴**

粘贴内容

- **格式刷**

复制单元格的格式，比如要将B2单元格的格式复制到B3单元格，可先选中B2单元格，然后点工具栏上的[格式刷]，此时会看到[格式刷]变成，然后再选中B3单元格。这样就将B2的格式复制到了B3了。最后再点一下[格式刷]，以恢复到正常状态。

- **撤销**

撤销上一步的操作

- **恢复**

恢复已撤销的上一步的操作

- **插入行**

在当前选中的单元格前插入行，如当前选中1行，则插入1行，如当前选中了多行，则一次插入多行。

- **增加行**

在报表的最后增加行，如当前选中1行，则增加1行，如当前选中了多行，则一次增加多行。

- **删除行**

删除当前选中的一行或多行。

- **插入列**

在当前选中的单元格前插入列，如当前选中1列，则插入1列，如当前选中了多列，则一次插入多列。

- **增加列**

在报表的最后增加列，如当前选中1列，则增加1列，如当前选中了多列，则一次增加多列。

- **删除列**

删除当前选中的一列或多列。

- **设置公式**

通过一个界面来选择或输入当前单元格的公式内容。

- **报表属性**

弹出一个界面来设置当前报表的属性

- **行属性**

弹出一个界面来设置当前选中行的属性

- **列属性**

弹出一个界面来设置当前选中列的属性

- **单元格属性**

弹出一个界面来设置当前选中单元格的属性，如当前选中了一组单元格，则同时设置这一组单元格的属性

- **设置数据源**

弹出一个界面来配置数据源的信息

- **设置数据集**

弹出一个界面来配置数据集的信息

- **报表参数**

弹出一个界面来设置当前报表的参数

- **设置宏**

弹出一个界面来设置当前报表的宏。

- **子报表**

定义当前报表要调用的子报表。

- **统计图**

设置当前单元格的统计图属性

- **导入excel文件**

将一个excel文件转换成e表格式的报表文件

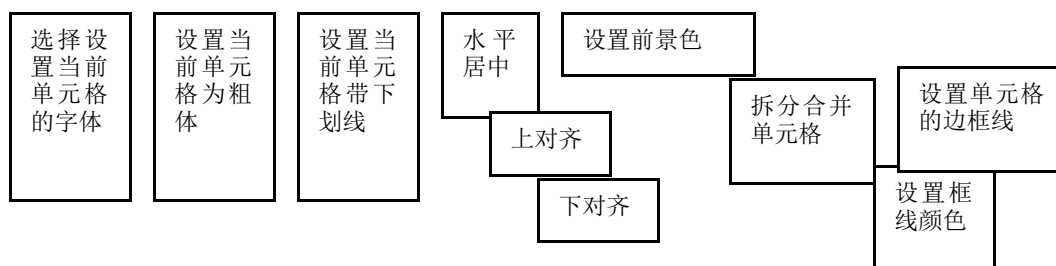
- **选项**

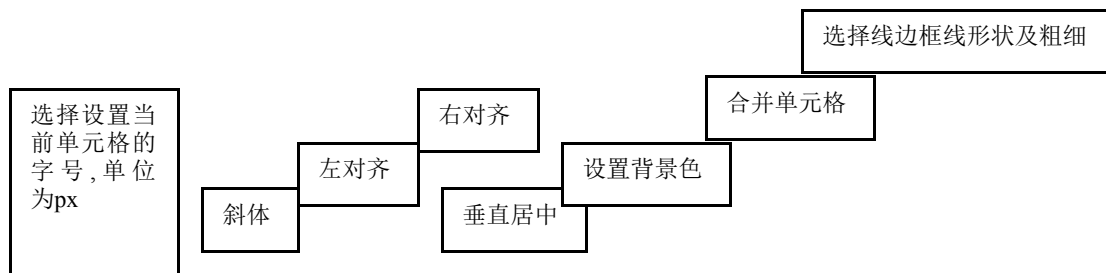
配置e表设计器的默认信息。

- **调试工具**

用脚本来调试或批量修改当前报表的html内容。

第二行工具栏(单元格的功能区)



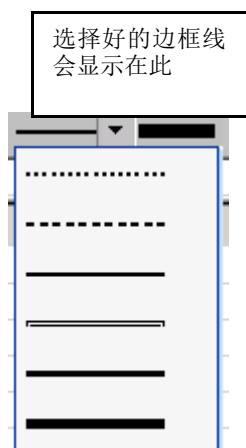


点击此则直接设置下边的颜色



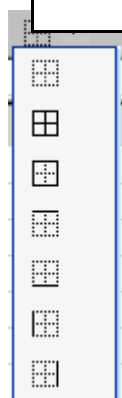
点此弹出一个系统颜色选取窗口

当前选中的颜色



点击此则出现下面的选择菜单

选择好的边框模式会显示在此,点此则直接应用它

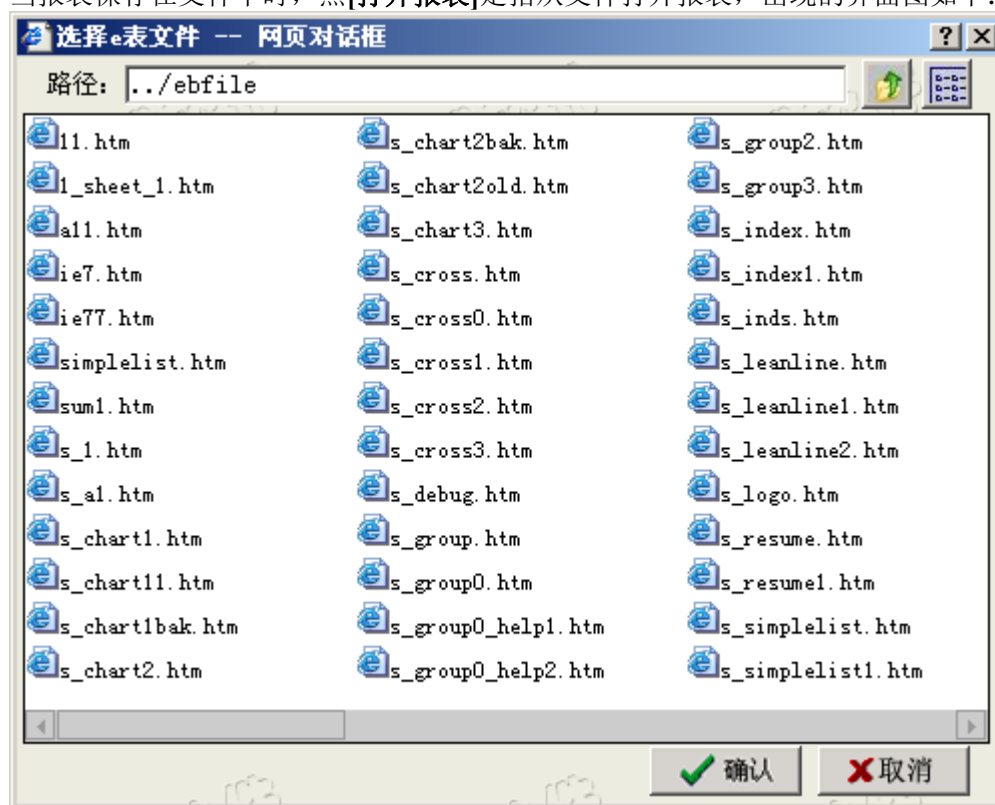


点击此则出现下面的选择菜单

(二)打开运行报表

打开报表

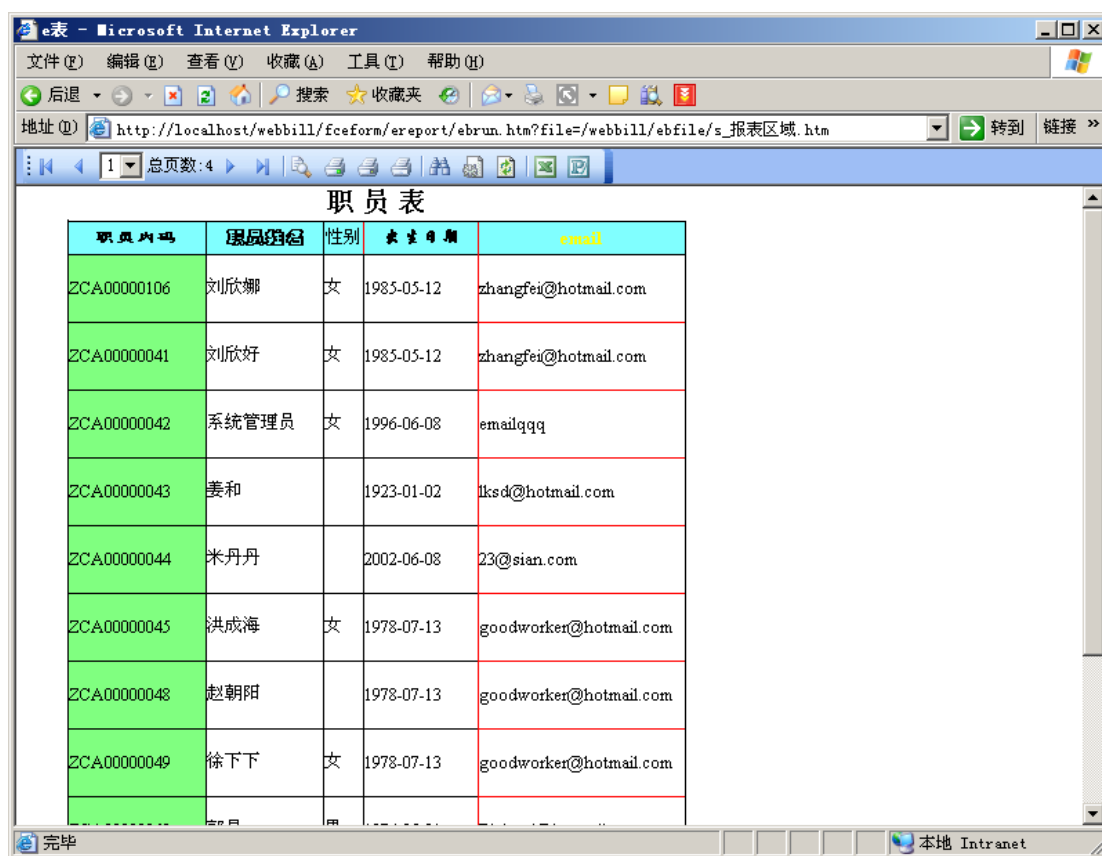
当报表保存在文件中时，点[打开报表]是指从文件打开报表，出现的界面图如下：



当报表保存在数据库中时，点[打开报表]是指从数据库中打开报表，出现的界面图如下：

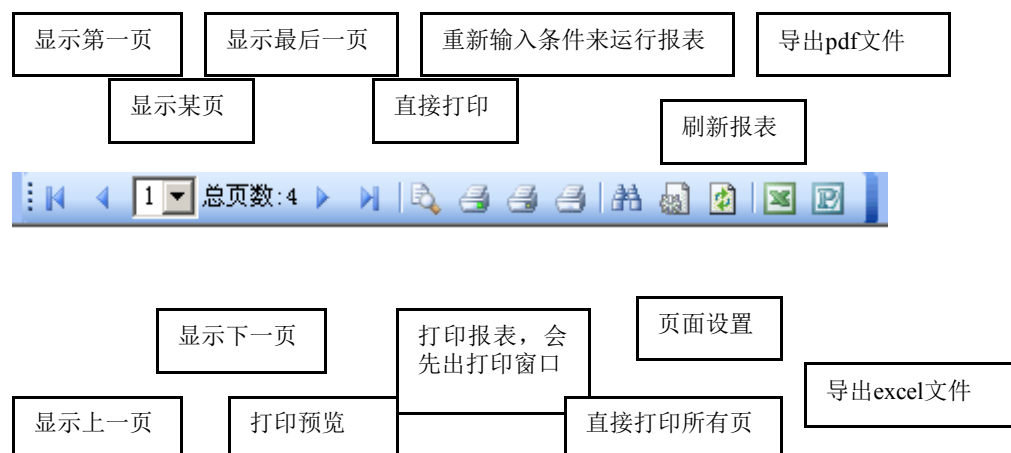


运行报表



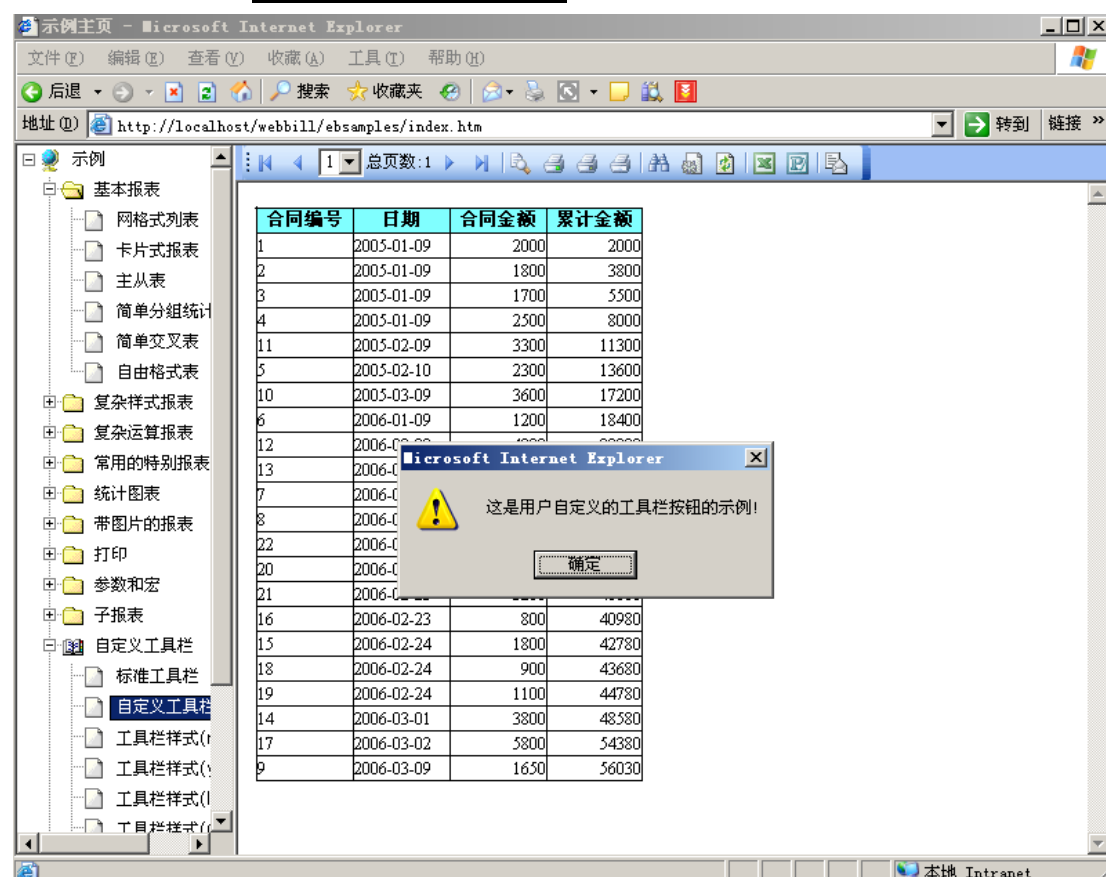
运行报表的界面是上面为一个工具栏，下面是报表的内容。

报表运行工具栏



自定义报表运行工具栏的按钮

这个是自定义的功能按钮



自定义报表运行工具栏的样式



(三)单元格

报表是由许多单元格组成的，下面来看看单元格的属性：

单元格属性

选择下面的输入框的内容
是真正的值还是公式

弹出一个窗口来
选择生成公式

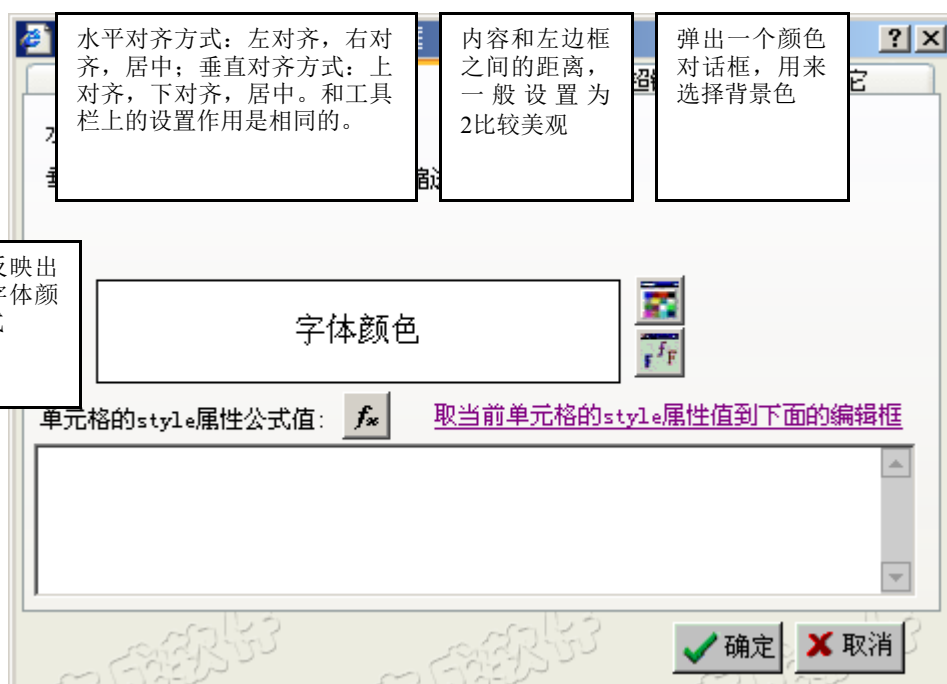
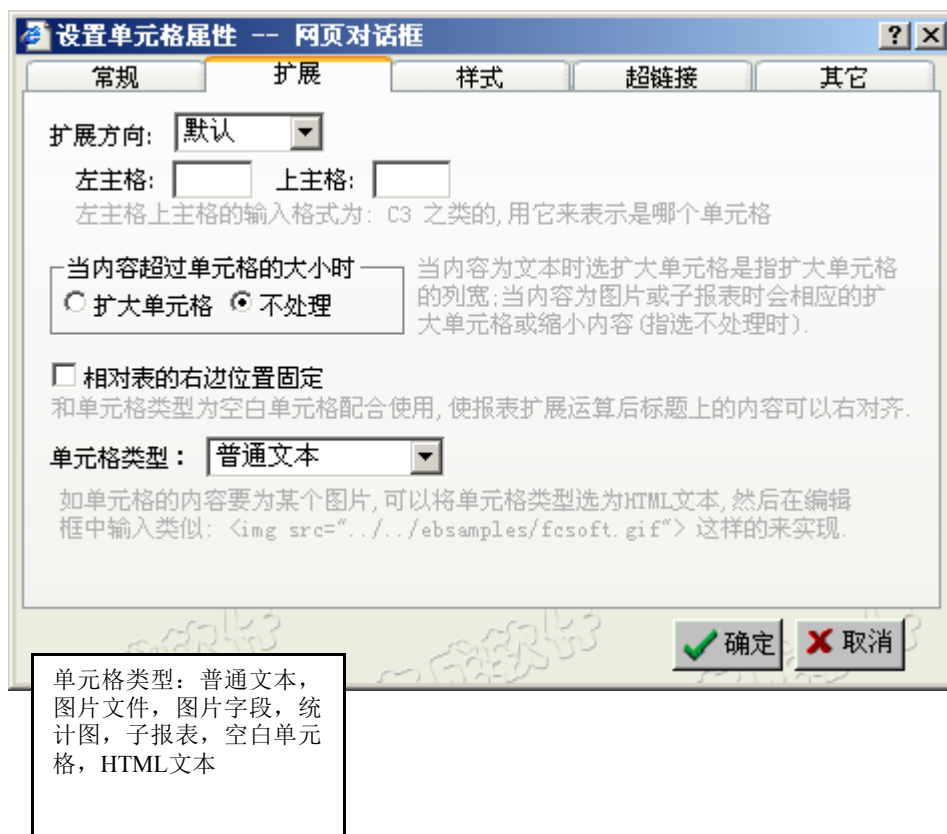
选择下面的输入框的内容是真
正的单元格显示值还是公式



显示格式仅仅用来对单元格的数据值进行格式化，例如：单元格的数据值如果是12345，显示格式是"#,##0.00"，那么该单元格就显示成"12,345.00"

分类为：数值，日期，时间，日期时间，货币，分数

扩展方向为：默认、横向扩展、纵向扩展、不可扩展



用公式来定义单元格的style属性，这样单元格的样式就都可以随着公式而动态变化。比如实现值为负数时用红色表示。

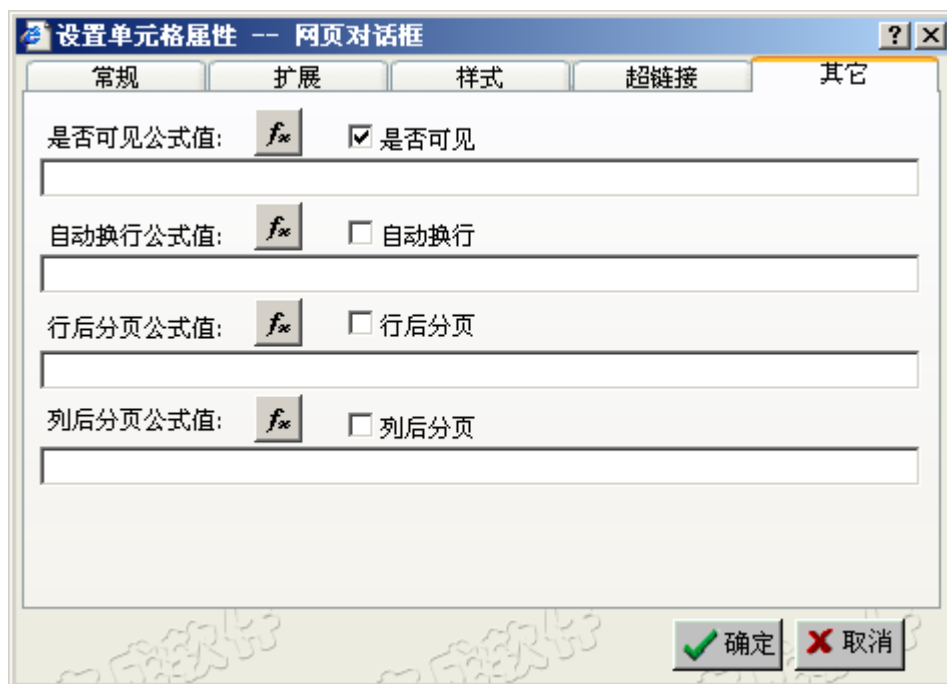
弹出一个窗口，它可以设置字体，字大小，粗体/斜体/下划线，字的前景色等。

如要一个常数的地址，则用引号引起来，如：“http://www.fcsoft.com.cn”。

打开的窗口，即target属性，值有：本窗口，新窗口，父窗口，顶层窗口。



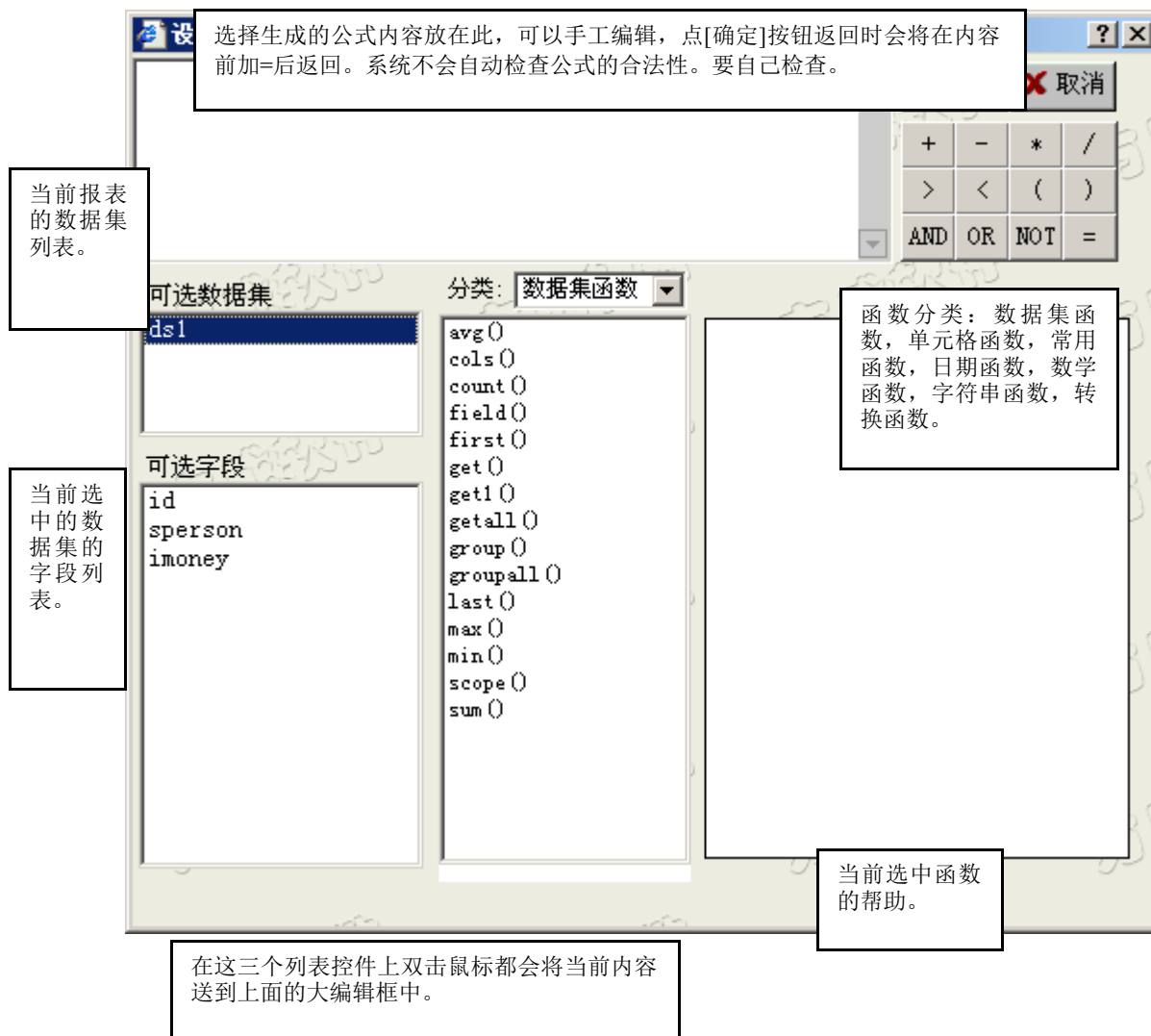
当下面的编辑框为空时，即没有公式值时，表示用常数值。如是否可见下的编辑框为空时，表示没有是否可见的公式值，只取是否可见的常数值，如下图时，是否可见已勾上，则表示是否可见的值是true，当有公式值时，则是否可见勾没勾上都是一样的，因为它取公式值，即由公式的运算结果来定。



单元格公式

可以设置为公式的单元格的属性有：数据值，显示值，style，超链接地址，是否可见，自动换行，行后分页，列后分页。

公式可以直接手工输入，也可以通过下面的窗口来选择输入。



(四)报表属性及打印

报表属性及打印

报表属性是用来设置当前报表的总体属性。它的界面图如下：



在此设置分栏时的分栏数。

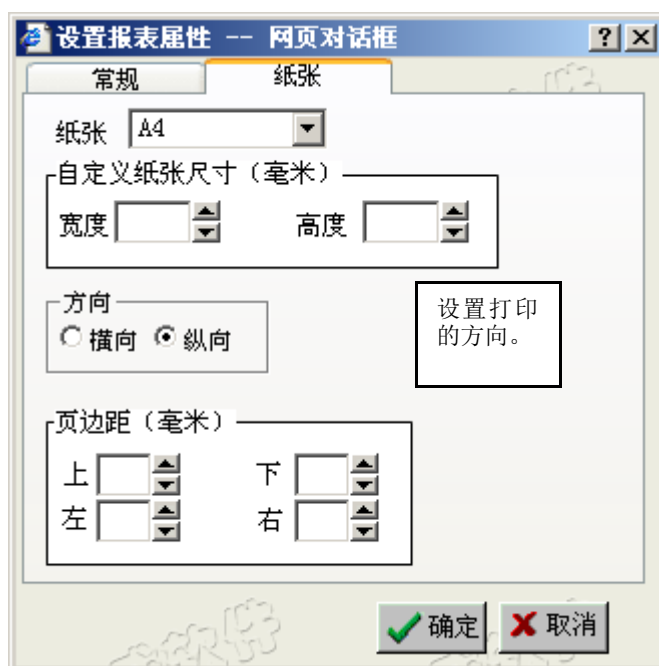
分页方式：按纸张尺寸分页；按数据行数分页。

当按数据行数分页时可在此设置每页显示的行数。

点此可以出一个窗口来选择图片文件。

设置打印的纸张大小。

当纸张为自定义时在此自定义纸张的宽高。



设置打印的方向。

设置上下左右的页边距。

打印纸张：

纸张规格	宽度（毫米）	高度（毫米）
A0	841	1189
A1	594	841

A2	420	594
A3	297	420
A4	210	297
A5	148	210
B0	1000	1414
B1	707	1000
B2	500	707
B3	353	500
B4	250	353
B5	176	250

尺寸换算公式：

$$(1\text{毫米} \times 72) / 25.4 = 1\text{像素}$$

$$72\text{像素} = 1\text{英寸}$$

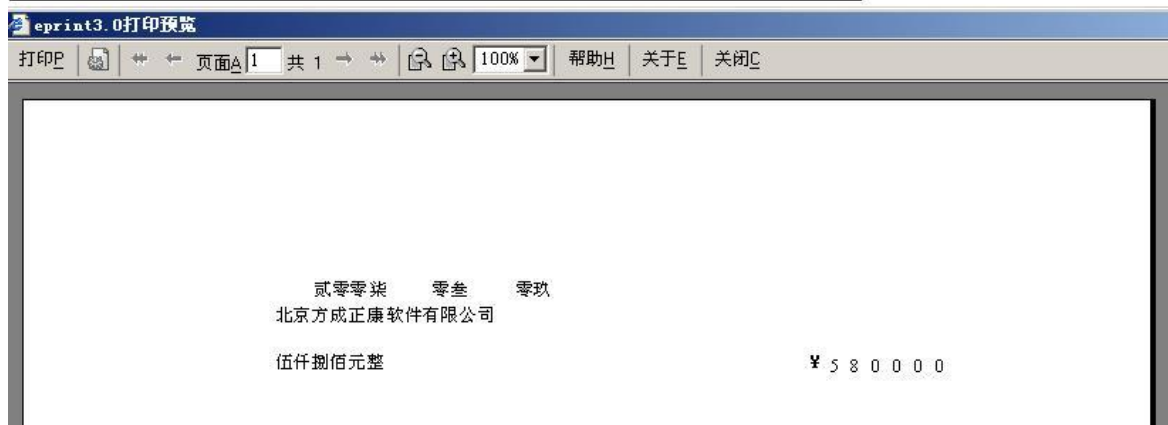
注意：当使用运行报表工具栏上的[直接打印所有页]时，要打印的页面会先临时保存在\ebssys\ebtmpfile\printtemp\目录下。打印完后这些临时文件就没有用了，为了怕累积太多，可以定时清除这个目录的文件。

套打

要实现套打第一步应扫描底图，扫描底图的时候最好按照分辨率72来扫描，因为打印机是按照72分辨率打印的，不同的分辨率就会导致不同的尺寸。然后将这个图片设置为报表的背景图。

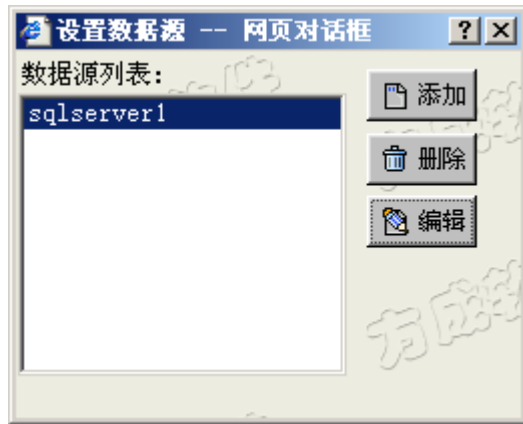
然后在含背景图的报表中设计报表就可以了。

如下图：

(五)数据源

e表有一个默认的数据源，其名称叫default，其配置信息默认在web.config文件中配置。
e表支持从多个数据源取数据。下面是多数数据源的配置界面：



添加一个新的数据源

删除当前选中的数据源

修改当前选中的数据源

当数据源名称列表

编辑数据源的界面如下图：

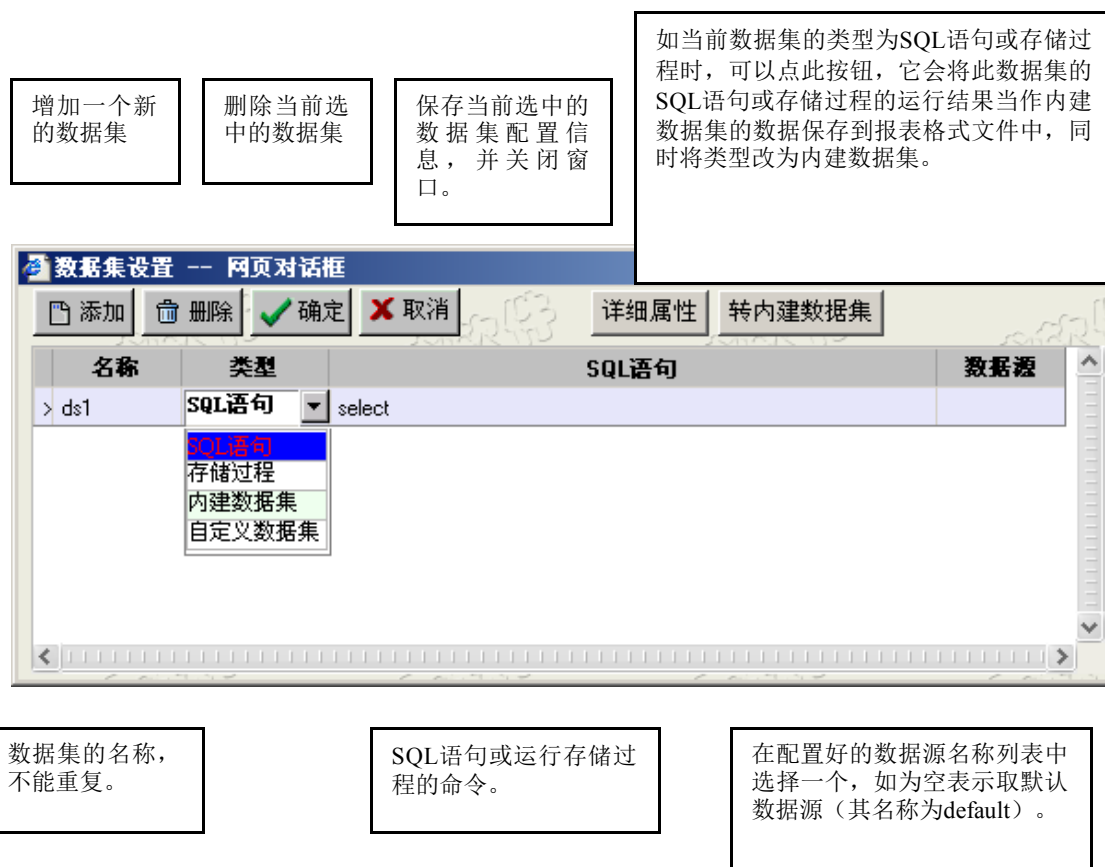
数据源名称，不能重复，也不能叫default



数据库类型如：oracle, sqlserver, sybase, sqlany, Informix, foxpro, access, foxbase, db2, mysql

(六)数据集

数据集的设置界面图如下：



当数据集的类型为SQL语句或存储过程时点[详细属性]按钮时,将会出现如下的界面图:

运行下面编辑框中sql语句,并将运行结果以表格形式显示



用一个向导界面来自动生成sql语句到下面的编辑框中

写SQL语句，如数据集的类型为存储过程，则写法为：{call procName(?,?)}，如没有参数，则写为：{call procName}。

执行存储过程的sql和常规方式没有区别，特殊的地方在于：如果存储过程是通过输出参数来返回结果集的，那么该输出参数照样用问号表示，但是在参数tab页中相对应的参数表达式用@@result来表示。

生成SQL向导

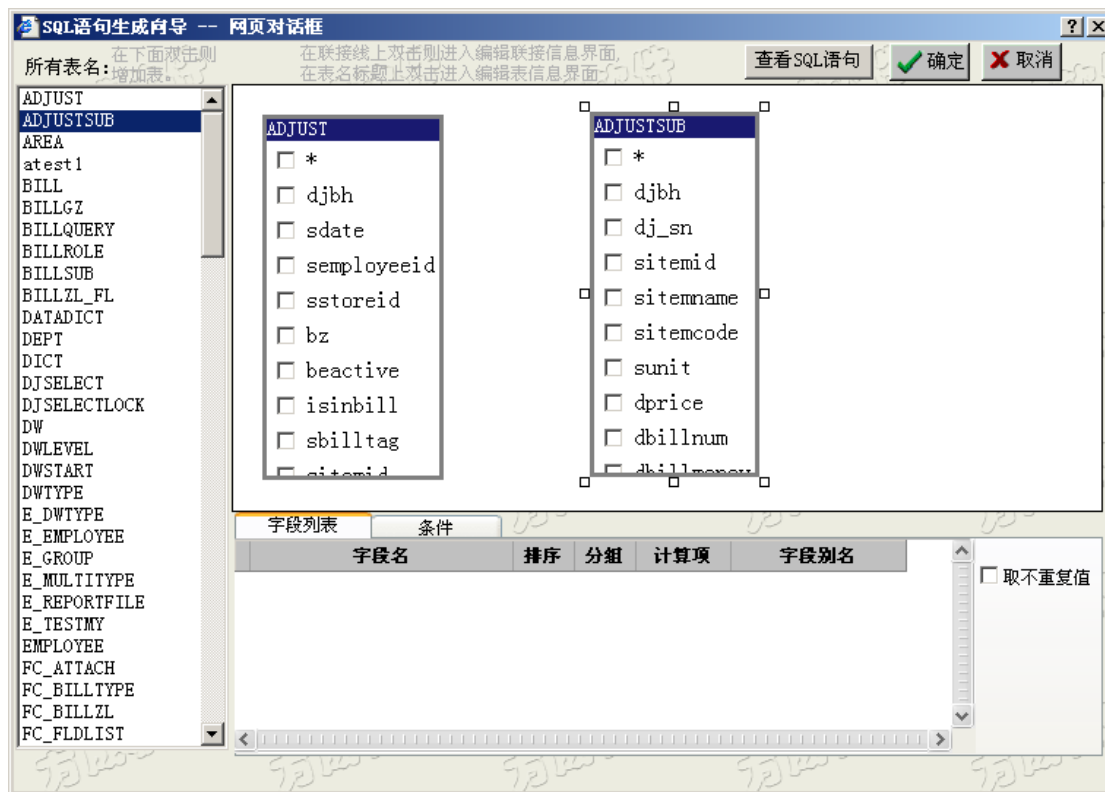
当点[生成sql向导]按钮时，得到如下的界面图：



左边是数据库中的可有的物理表名。右边上部为可拖动的表关联的编辑区，下部为字段列表及条件的编辑。

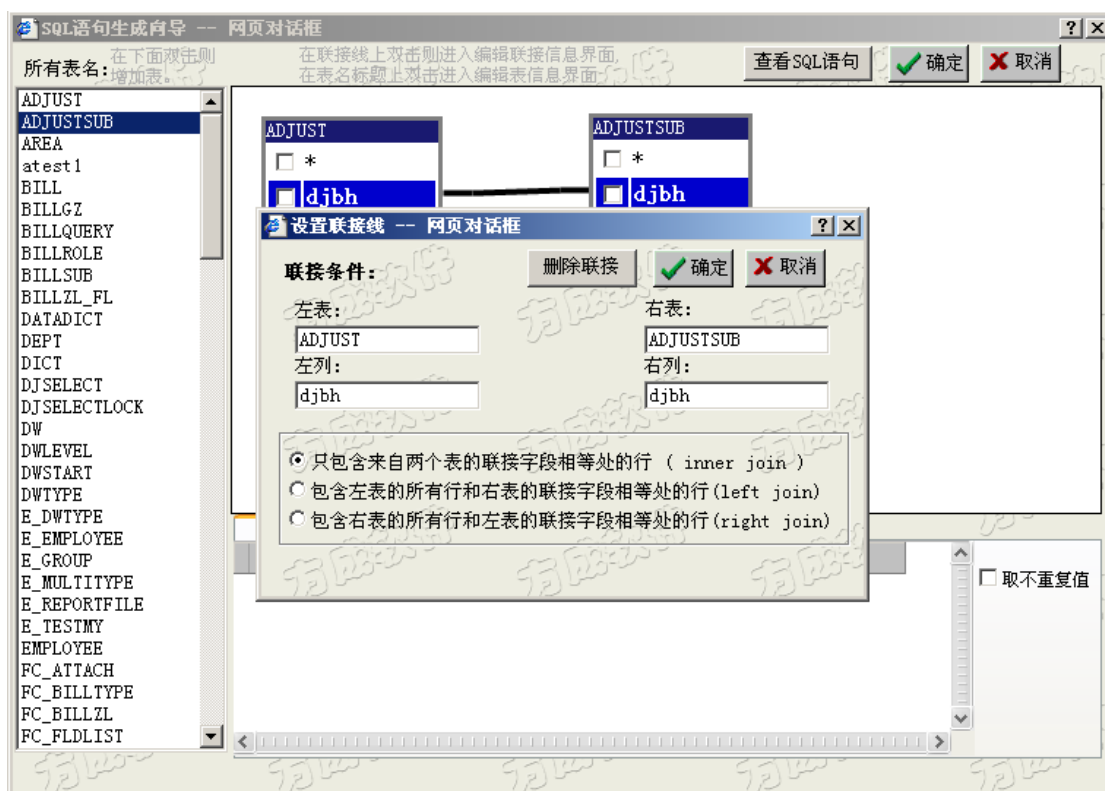
此界面的常规操作如下：

先在左边的要查询的表名上双击，如在 ADJUST和ADJUSTSUB 这两个表上双击后，再在左上部将 ADJUSTSUB 子窗口向右边拖开一些，拖动操作是先用鼠标在 ADJUSTSUB 子窗口的标题上按住鼠标左键不放，拖动鼠标即可。得到如下图所示的界面：

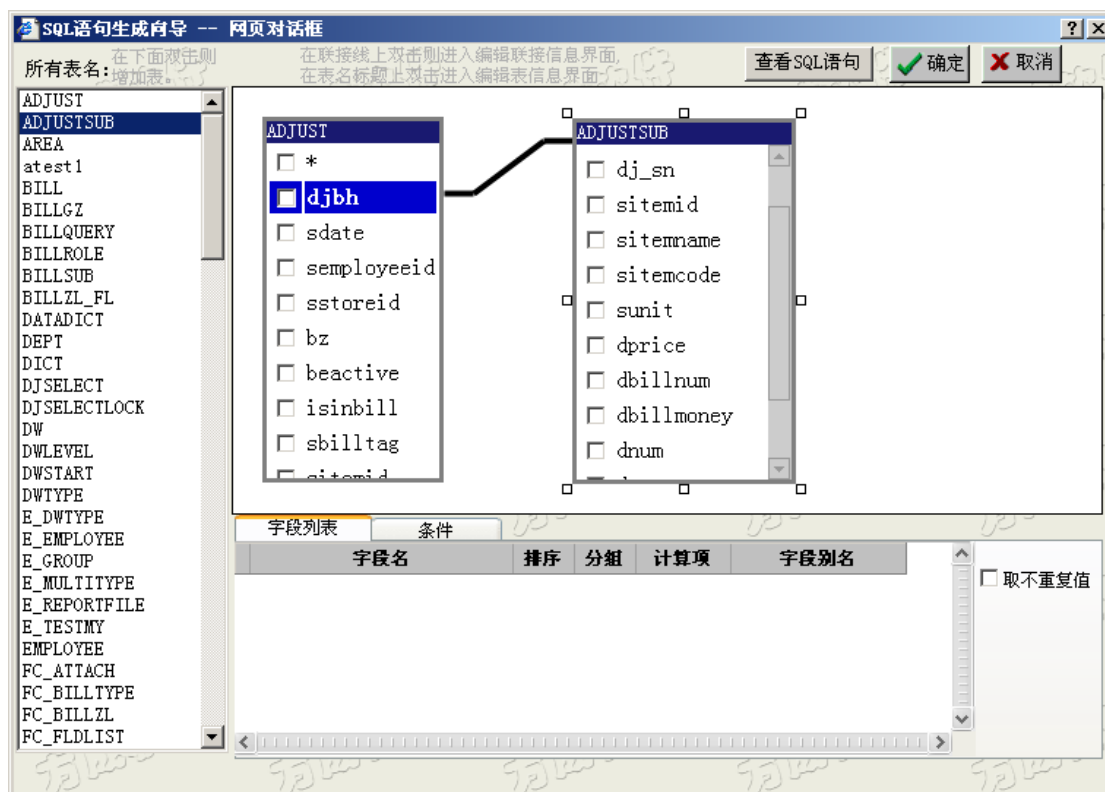


因ADJUST和ADJUSTSUB这两个表是通过djbh字段来关联的，建立关联的操作如下，先用鼠标在 ADJUST的djbh字段上点一下，以使它的底色变成蓝色，表示选中了此字段的意思。然后在ADJUST的djbh字段上按下鼠标左键不放，拖动鼠标到ADJUSTSUB的djbh字段上时再松开鼠标。此时就能看到在两个djbh字段之间画了一条线。这样关联就建好了。

如果要删除关联或改变关联的属性，可以用鼠标在关联线上点击，此时会弹出一个窗口。界面图如下：

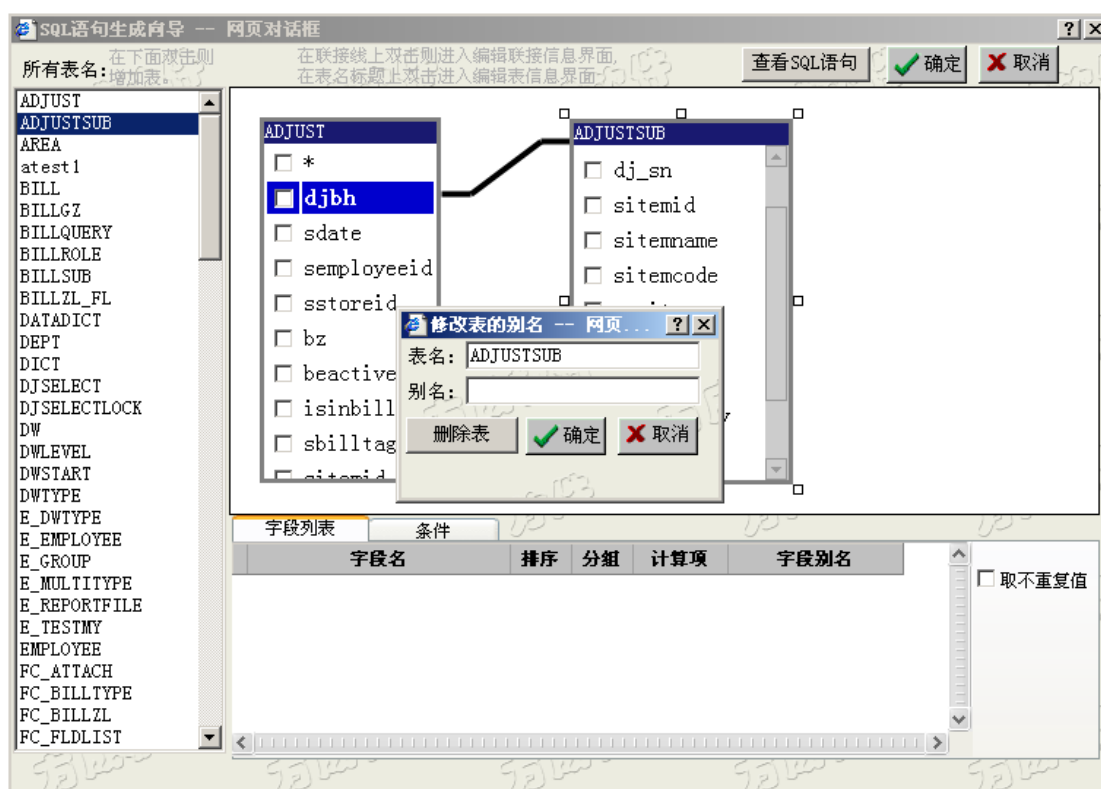


当一个表的字段数比较多时，窗口显示不下时，可以拖宽窗口，此时会看到右边有一个滚动条，可以用鼠标上下滚动，以显示其它字段。界面图如下图所示：



在表的子窗口的标题上双击鼠标，则弹出一个窗口，要其中可以修改表的别名，也可以在

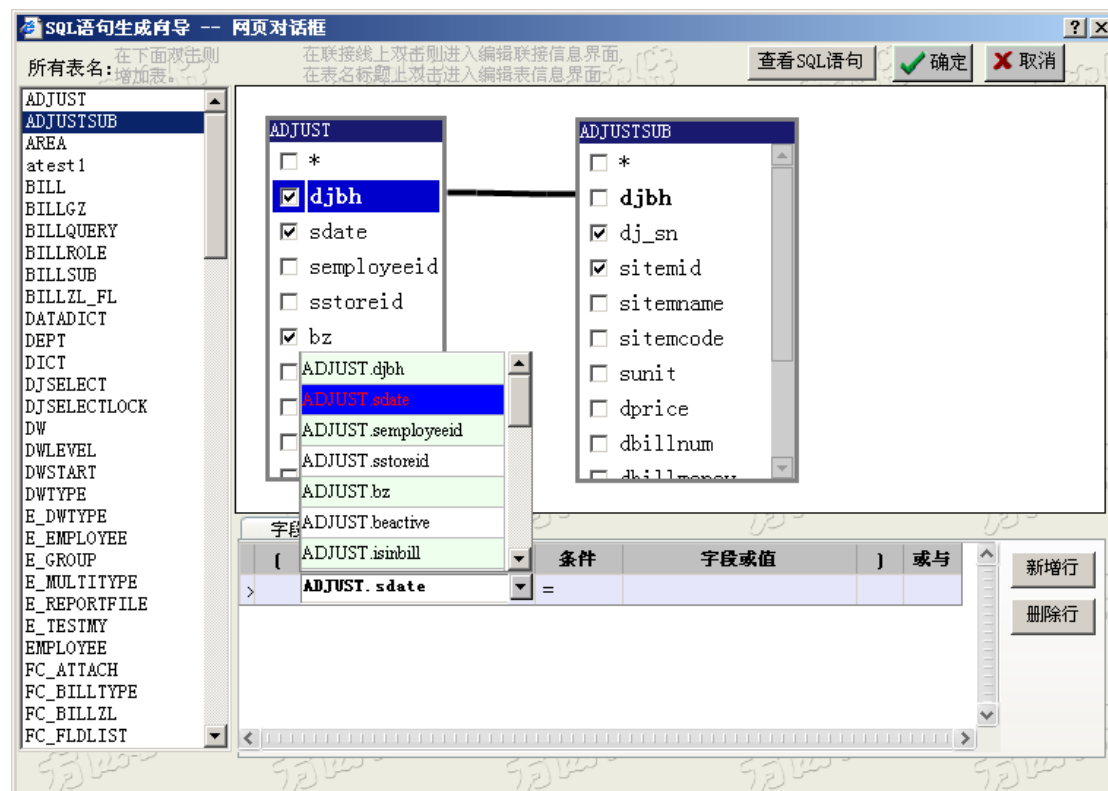
编辑区中删除当前表。界面图如下所示：



在表的子窗口的字段左边的复选框中点击选中，则表示将此字段加入下边的字段列表中，即SQL语句的select 部分，在下面的表格中可以选择是否排序，分组，以及计算字段，字段别名等信息。界面图如下图所示：



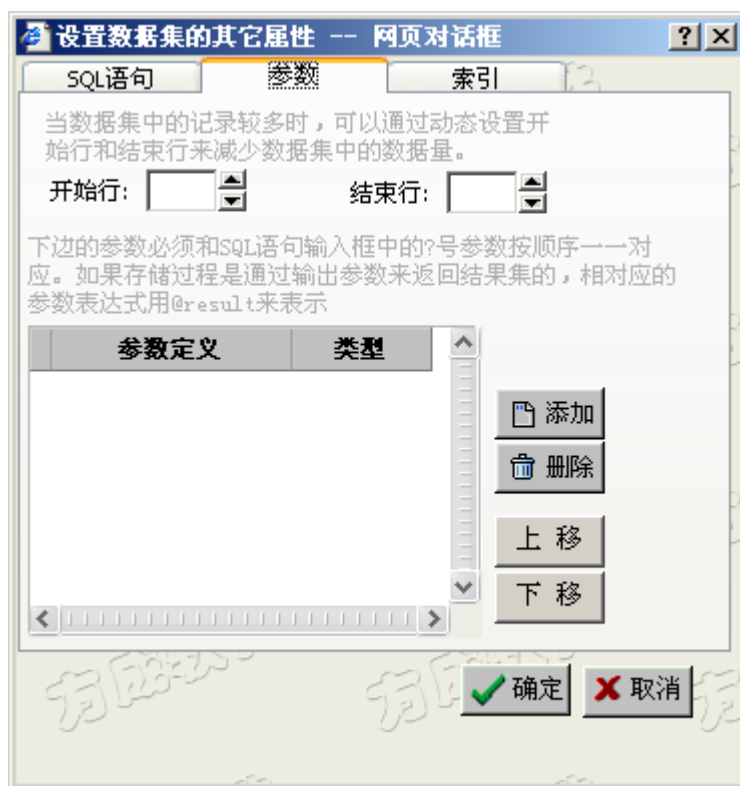
而SQL语句的where部分则通过点[条件]页签，然后在下面的表格中输入条件。界面图如下所示：



设置好SQL语句后，可以点[查看SQL语句]按钮，出现如下的界面图：



在这里可以检验一下SQL语句是否能正确运行，也能看到运行结果。



增加一个参数。

删除当前选中的参数。

上移或下移当前选中的参数。

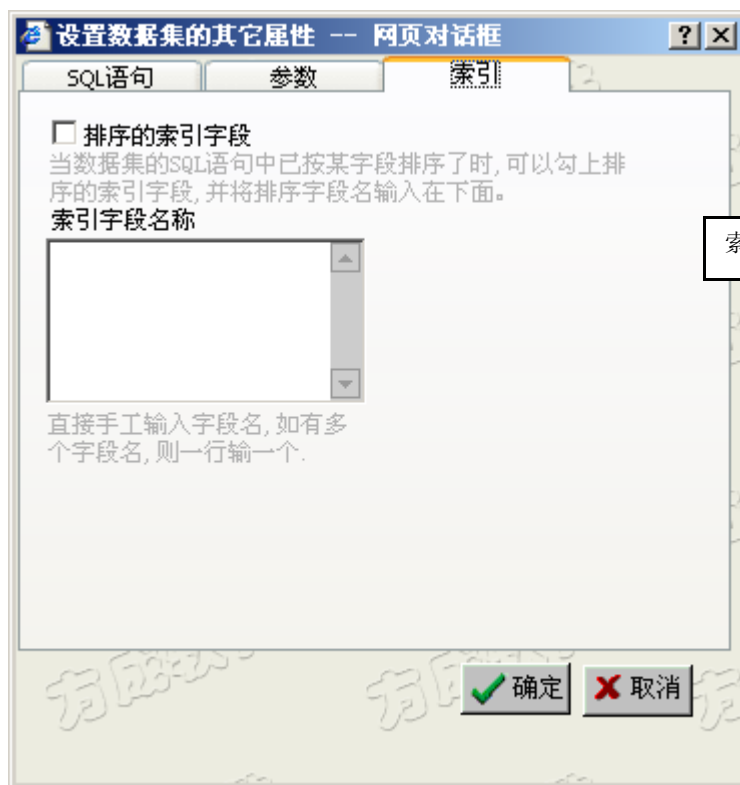
参数列表，其个数应和上面SQL语句中的?参数一致。

参数必须和上边输入框中（即SQL语句中）的?号参数按顺序一一对应。如果存储过程是通过输出参数来返回结果集的，相对应的参数表达式用@result来表示。

参数的类型有：

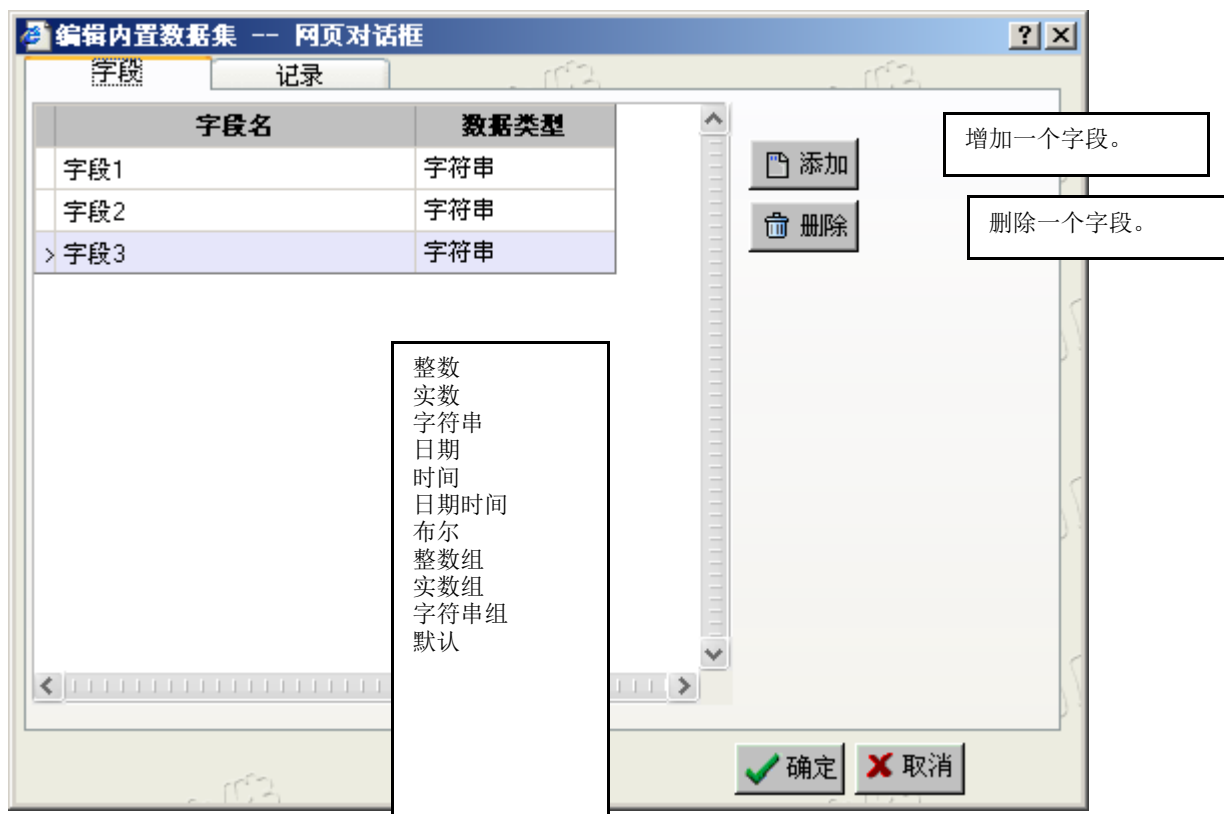
整数，实数，字符串，日期，时间，日期时间，布尔，整数组，实数组，字符串组，默认

当点索引页签时，出现如下的界面图：

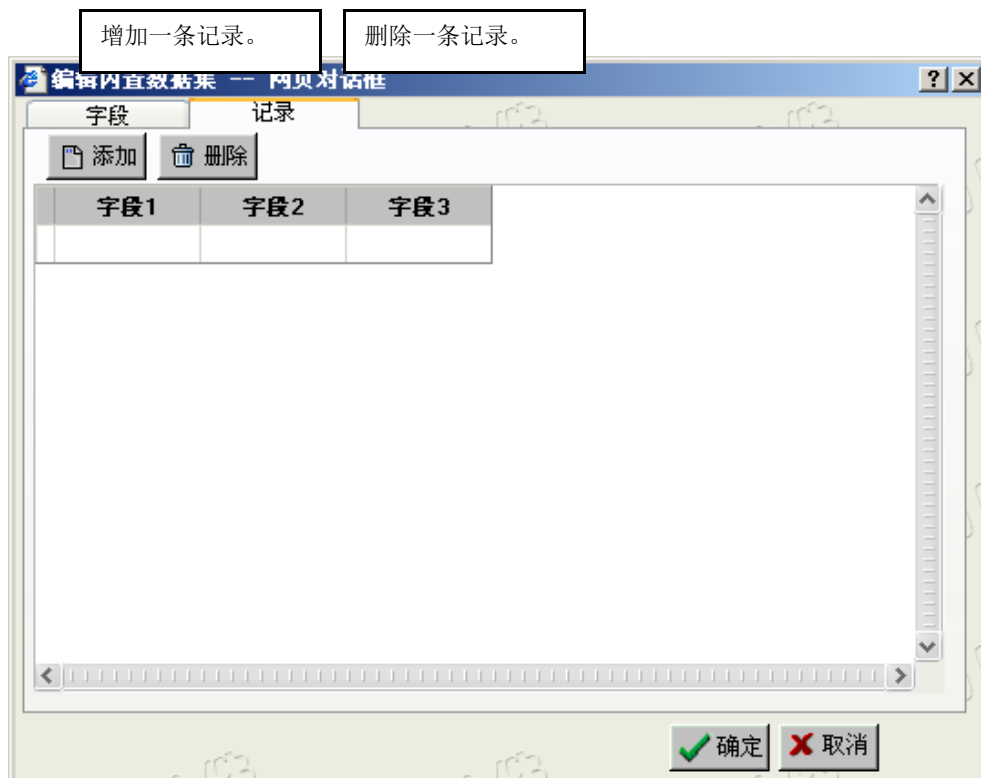


当数据集的SQL语句中已按某字段排序了时,可以勾上排序的索引字段,并将排序字段名输入。使用了索引可以提高报表运行的速度。
当数据集中的记录较多时, 可以通过动态设置开始行和结束行来减少数据集中的数据量。以减少内存的占用量。

当数据集的类型为内建数据集时点[详细属性]按钮时,将会出现如下的界面图:



在这个界面完成字段的增删改。相当于表结构的设计。
点[记录]页签，将会看到如下的界面图：



在这个界面完成记录的增删改。

当数据集的类型为自定义数据集时点[详细属性]按钮时,将会出现如下的界面图:



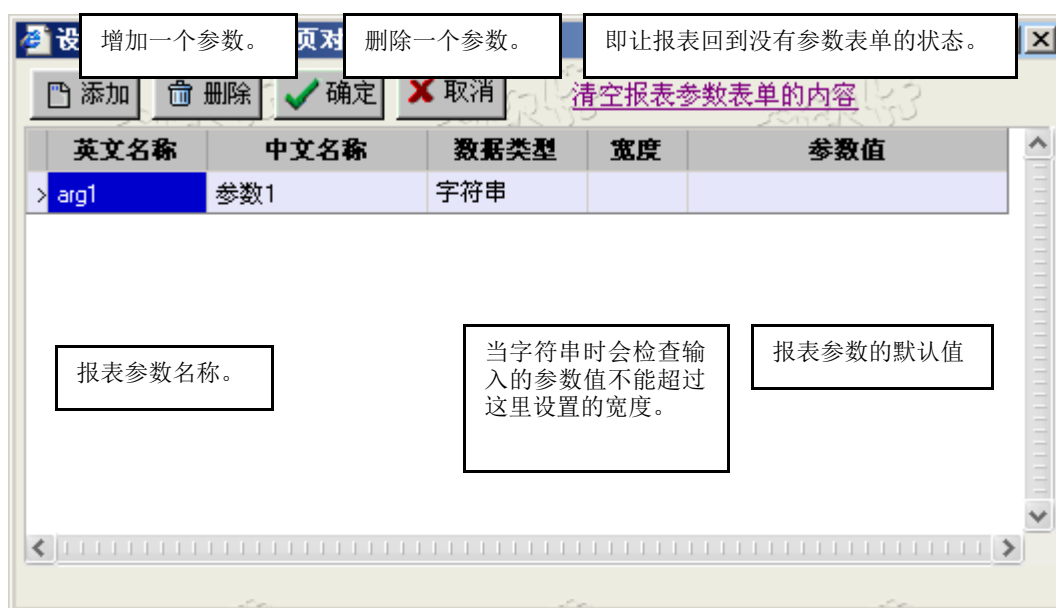
(七)参数和宏

一般来说，报表都需要根据用户输入条件，然后才运行报表，来得到赋合用户条件的结果。这时就需要给报表设置参数。

参数是一个有数据类型有缺省值的变量，在表达式中使用参数时，应该当成变量来使用。参数可以在数据集和表达式中被引用，表达式中直接写参数名引用。

报表参数定义

下面是定义报表参数的界面图：



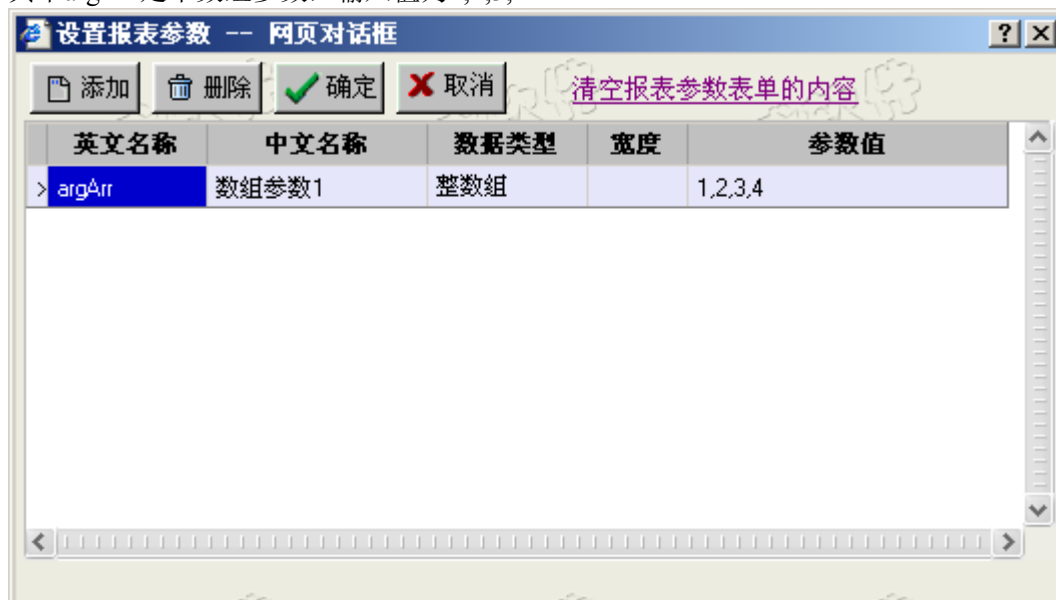
参数的类型有：

整数，实数，字符串，日期，时间，日期时间，布尔，整数组，实数组，字符串组，默认

参数的数据类型里有整数组、实数组、字符串组，这些类型的参数都是数组参数，如果选择这些类型，这些参数在报表模板里应该被当成数组来使用，用法和一般数组的用法完全一致。数组型参数录入时，多个值之间用英文逗号分隔，例如：1,2,3,4等。获得数组型参数的元素个数：count(argName)，其中argName是个数组参数，获得数组型参数中第二个元素：argName[2]，数组型参数由于是个数组集合，因此可以象集合表达式一样直接写入单元格进行扩展，例如：

B2	=	=argArr
	A	B
1		
2		=argArr

其中argArr是个数组参数，输入值为1,2,3,4

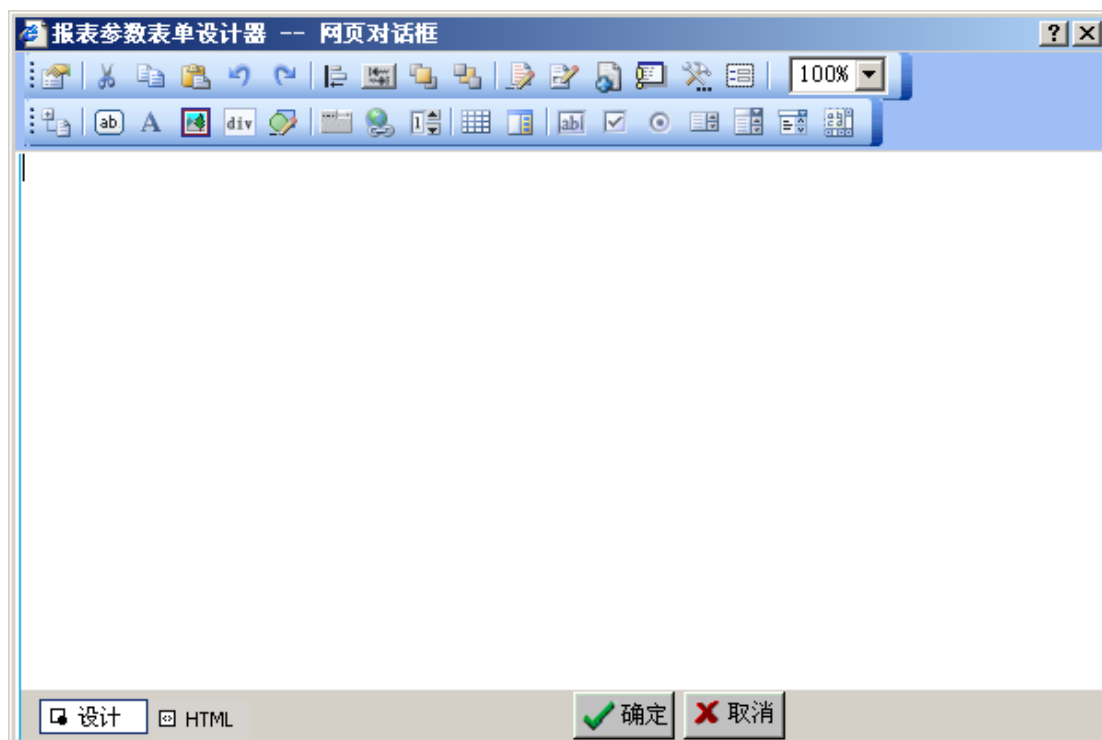


则运行结果为：

1
2
3
4

报表参数表单设计器

在至少设置了一个报表参数后，便能进入报表参数表单设计器了，下面是报表参数表单设计器的界面图：



这在后一节详细说明。

宏

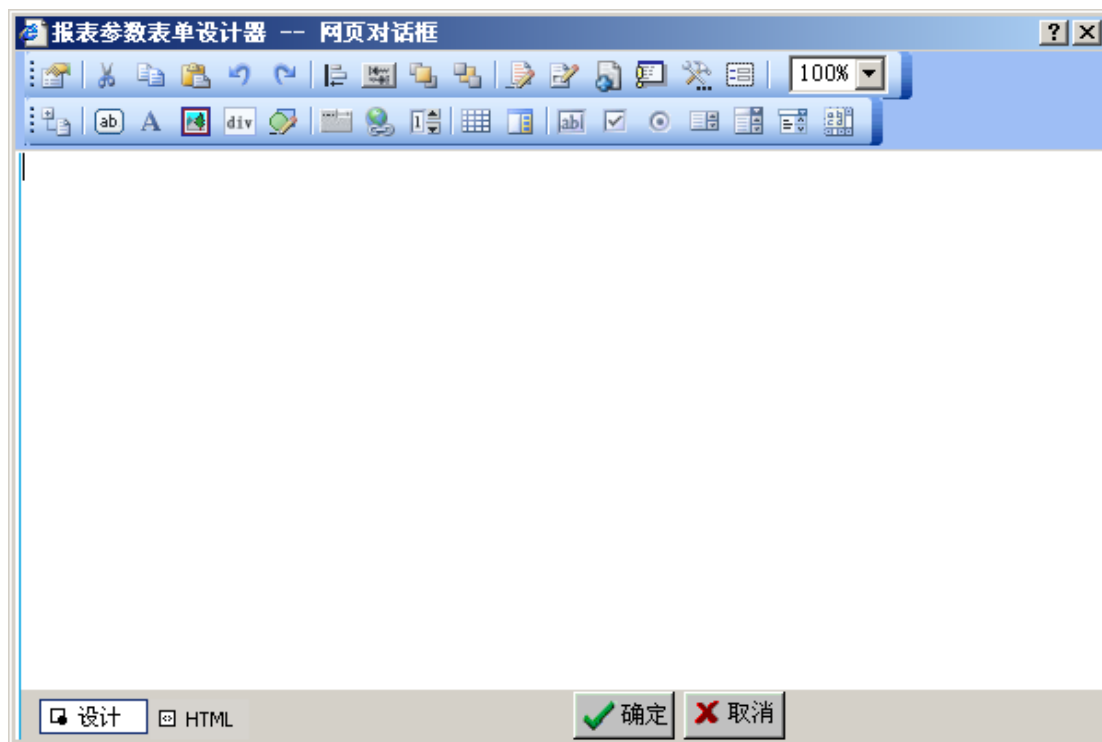
宏是一个没有数据类型的字符串标识，在报表运算之前，系统会全面搜索整张报表定义，将所有的宏名替换成宏值。可以在报表中任何位置引用宏，如单元格值、表达式、数据集定义、单元格属性表达式中等，引用方法为`${宏名}`。

使用宏可以使一类报表格式文件只需要画一张，然后通过不同的宏值来得到不同的报表格式文件的运行效果。

下面是宏定义的界面图：




(八)报表参数表单




表单设计器的常用功能


1. 复制

- 将表单内选中的内容复制，一次可复制一个或多个控件，若要复制多个控件，必须使用Ctrl键依次点击将控件选中，然后在点击“”按钮即可。可使用快捷键“Ctrl+C”。


2. 粘贴

- 粘贴已复制或剪切的内容，只须控件复制或剪切后点击“”按钮即可，如果您在操作完成后未发现粘贴的控件，可拖动原复制控件，原因是粘贴控件与原复制控件重叠所致。可使用快捷键“Ctrl+V”。

3. 剪切

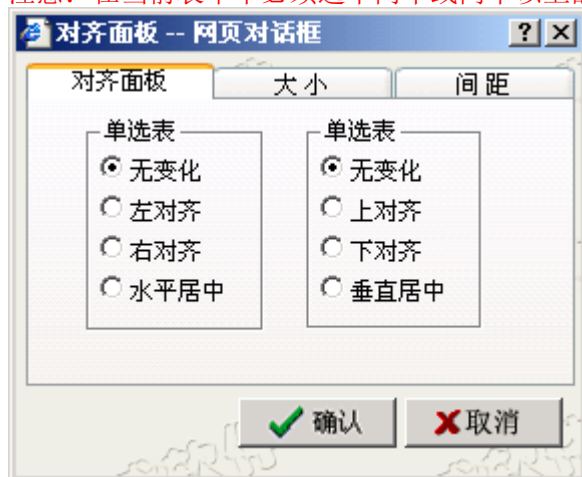
- 将表单内选中的内容剪切，一次可剪切一个或多个控件，若要剪切多个控件，必须使用Ctrl键依次点击将控件选中，然后在点击“”按钮即可。可使用快捷键“Ctrl+X”。

4. 对齐面板

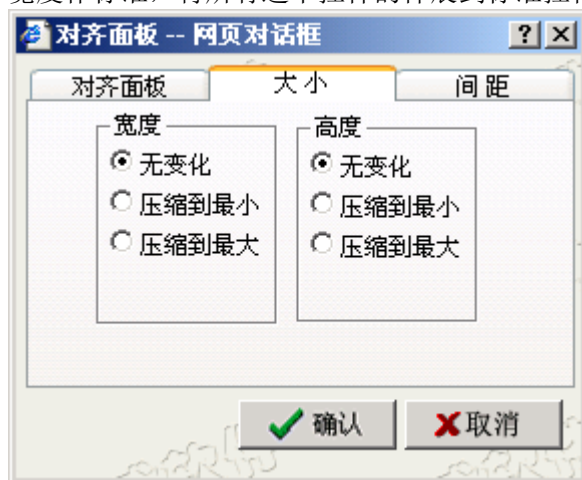
- 点击“”按钮，打开对齐面板属性页面，对齐面板水平方向对齐和垂直方向对齐，水平方向分为无变化、左对齐、右对齐和水平居中，左对齐是以第一次选中

的控件左坐标为标准对齐，右对齐是以第一次选中的控件右坐标为标准对齐，水平居中是以长度最大控件的中间坐标为标准对齐；垂直方向分为无变化、上对齐、下对齐和垂直居中，上对齐是以第一次选中的控件上坐标为标准对齐，下对齐是以第一次选中的控件的下坐标为标准对齐，垂直居中是以高度最大控件的中间坐标为标准对齐。如图：

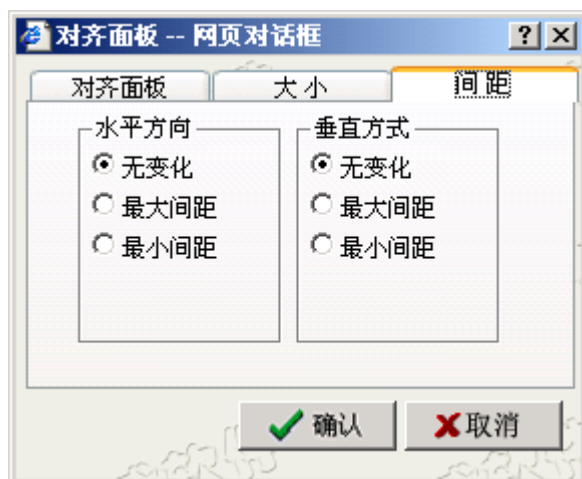
注意：在当前表单中必须选中两个或两个以上的控件才能使用。




- 设置控件大小对齐方式，可分为宽度对齐和高度对齐，宽度对齐和高度对齐又分为无变化、压缩到最小、伸展到最大。压缩到最小是比较选中的控件的高度和宽度的大小，以最小控件的高度和宽度为标准，将所有选中控件压缩为标准控件的大小。伸展到最大是比较选中控件的高度和宽度的大小，以最大的控件的高度和宽度作标准，将所有选中控件的伸展到标准控件的大小，如图：



- 设置控件与控件之间的间距大小，可分为水平方向和垂直方向两种对齐方式，水平方向和垂直方向分为无变化、最小间距、最大间距。最小间距是比较选中的控件与控件之间的水平和垂直间距的大小，以最小的间距为标准，将所有选中控件之间的间距调节为标准间距，最大间距以最大的间距作标准，将所有选中控件之间的间距调节为标准间距，如图：




5. 焦点次序


- 调整当前表单内所有控件的焦点次序。点击“”按钮，打开焦点次序属性页面，点击上下箭头按钮，调动列表内选中控件ID的先后顺序，即可调节表单内控件的焦点次序, 如图：




6. 前置

- 当两个或多个控件重叠时，将指定控件在前面显示。将指定在前面显示的控件选中，再点击“”按钮即可。

7. 后置


当两个或多个控件重叠时，将指定控件在后面显示。将指定在后面显示的控件选中后，再点击“”按钮即可。

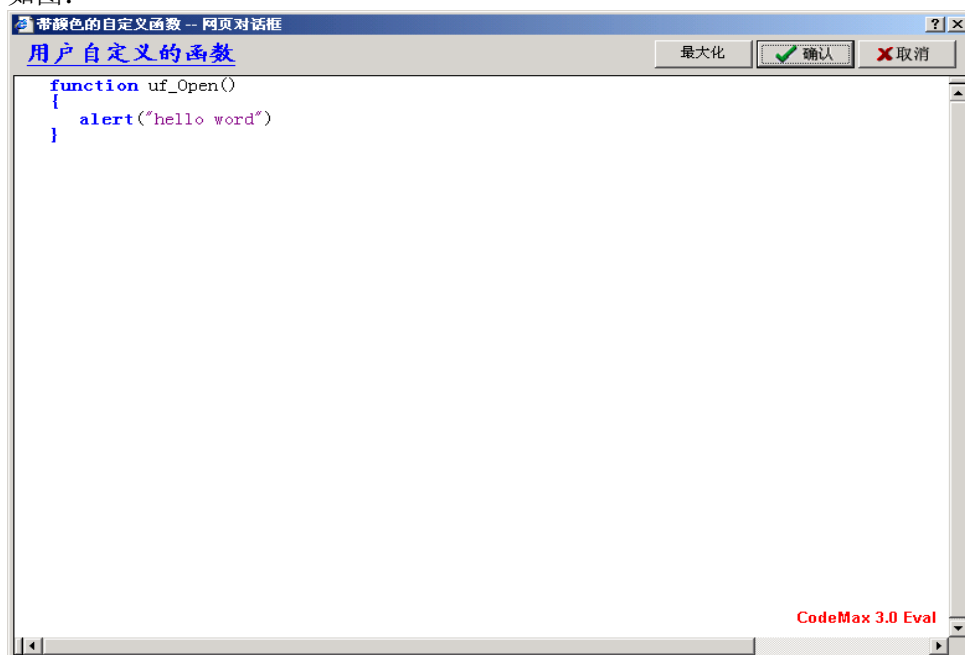
8. 自定义函数

- 点击“”按钮，打开自定义函数,此页面对话框支持javascript脚本，如图：



9. 能变色的自定义函数

- 点击 “” 按钮，打开能变色的自定义函数，此页面对话框支持javascript脚本，如图：




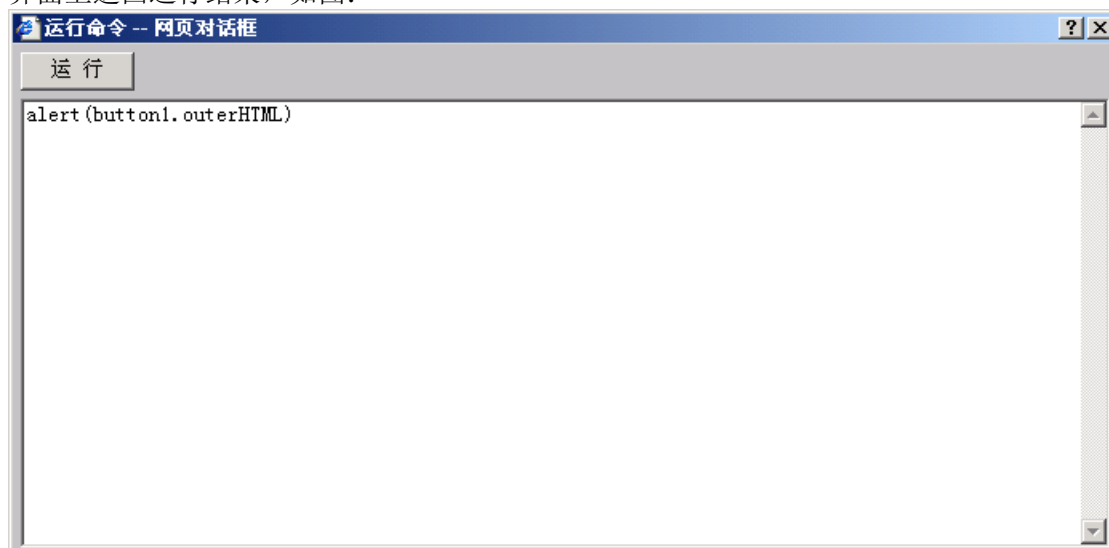
10. 附加页面元素

- 点击“”按钮，打开附加页面，附加页面元素就是在原有的表单基础上添加标准的HTML元素，如图：

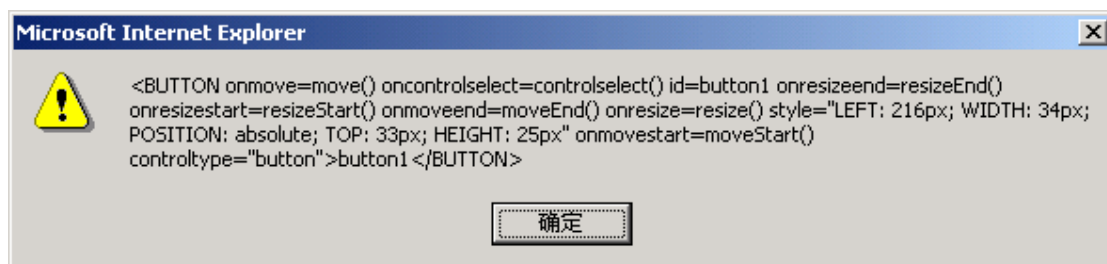


11. 直接运行命令


- 点击“”按钮，打开运行命令，在此对话框内直接输入运行命令，并在设计界面上返回运行结果，如图：




运行页面如图：

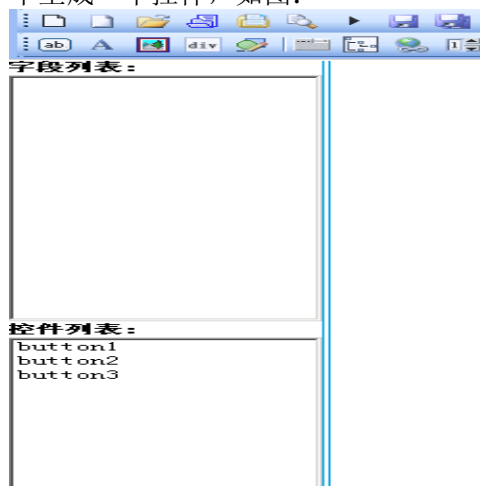


12. 切换定位类型

点击“”按钮，可设置当前表单的所有要新增元素的定位类型。

13. 显示/隐藏控件列表

- 点击“”按钮，可看到在设计编辑区分出一栏表，将设计编辑区上的控件ID用列表方式显示或隐藏出来，双击控件列表中的控件，打开控件属性页面，dataset数据集所有的字段列表，双击选中的dataset数据集中的字段，在设计界面中生成一个控件，如图：





14. 放大和缩小

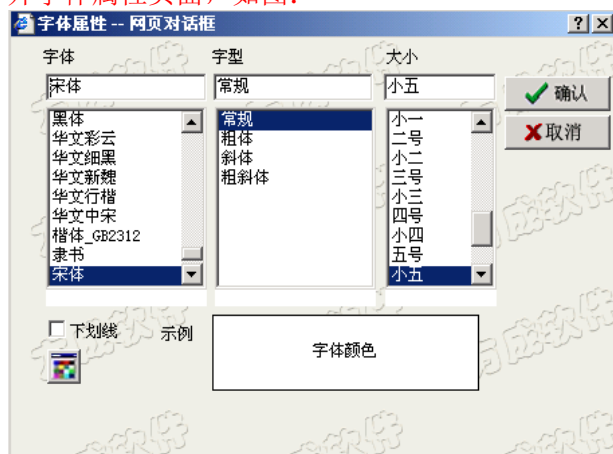
- 设置显示比例是指设置设计编辑区的整体的比例，本平台提供10% 25% 50% 75% 100% 150% 200% 300% 500%，用户可根据需要设置。

表单设计器的控件功能

1. 控件的公共属性：

- 拖动控件位置：选中当前控件，移动鼠标拖动控件位置，也可选中控件后，使用Ctrl键+箭头键拖动控件位置。
- 调整控件的大小，选中当前控件，移动鼠标调整控件的大小，也可以选中控件后，使用Shift键+箭头键调整控件的大小。
- 活动：设置或检索控件是否被禁用，选中表示启用，反之则为被禁用。
- 可见：设置或检索控件在运行时是否显示，选中表示显示，反之则被隐藏。
- 透明：设置控件的背景色是否透明，选中表示透明，反之则不透明。
- 左：设置控件与表单左边缘的位置。
- 上：设置控件与表单上边缘的位置。

- 宽：设置控件的宽度
- 高：设置控件的高度。
- 字体颜色设置：点击“”设置控件的背景色（不透明的情况），点击打开字体属性页面，如图：



设置控件的字体类型、字体大小、字体颜色、是否带有下划线。

- 组件边界风格：设置控件的是否有边框线，左边线：选中表示控件有左边线，反之则没有，上边线：选中表示控件有上边线，反之则没有，下边线：选中表示控件有下边线，反之则没有，右边线：选中表示控件有右边线，反之则没有。
- 双击打开选中控件,打开选中控件的属性页面。

2. 右键支持

- 表单设计界面右键，支持剪切、复制、粘贴、删除、属性，其中复制和粘贴与第一节和第二节讲到的复制和粘贴相同，属性是指控件属性，如有控件选中则指其属性，如没有则为最后添加的控件属性，如没有控件则为表单属性，如图：



3. 快捷键支持


- 剪切： Ctrl + x
- 复制： Ctrl + c
- 粘贴： Ctrl + v
- 删除： Delete

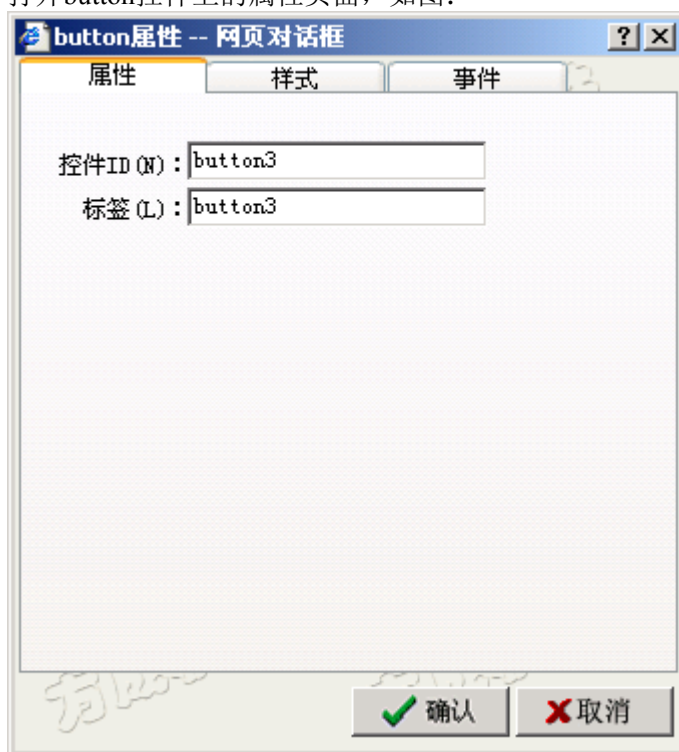
4. 事件

注：事件代码（不支持双引号, 点击“”按钮，选择系统函数）

- 单击：鼠标单击时触发该事件。
- 双击：鼠标双击时触发该事件。
- 进入：鼠标进入时触发该事件。
- 退出：鼠标退出时触发该事件。
- 按键：按键时触发该事件。
- 改变：内容发生改变时该事件。

5. button控件


- 在自定义表单的工具栏中点击“”按钮，增加button控件，双击button控件，打开button控件上的属性页面，如图：



设置button控件的属性页分为：“属性”、“样式”和“事件”三个页签

- 1) 设置button控件的“属性”页签中的控件ID：表示唯一标识，可以修改，标签：是控件显示的内容。

6. label控件

- 在设计界面的工具栏中点击“”按钮增加label控件，双击打开label控件的属性页面，如图：



设置label控件的属性页分为：“属性”和“样式”二个页签

1) 设置控件的ID和标签，请参考button控件的设置方法。

7. img控件


- 在工具栏上点击“”按钮，创建img控件，双击img控件，打开img控件的属性页面，如图：

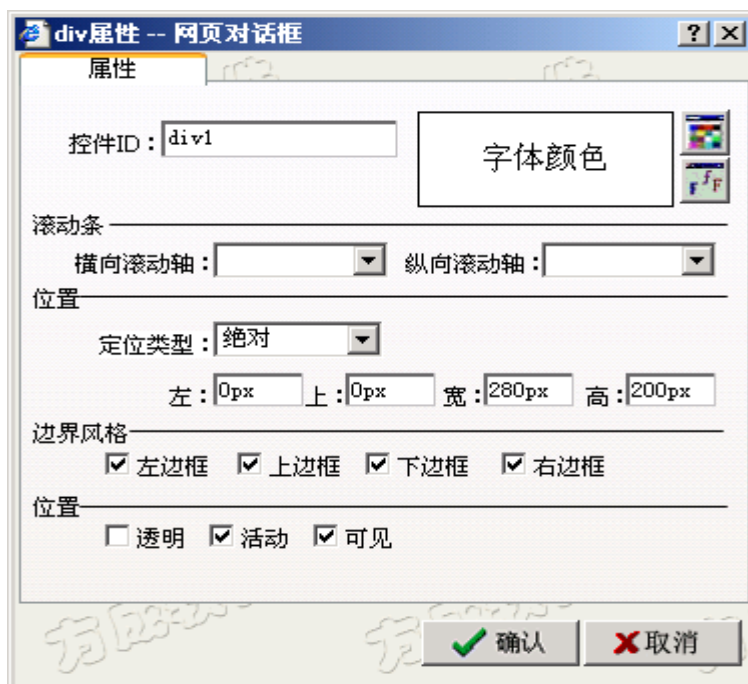


设置img控件的属性页分为：“属性”和“样式”二个页签

- 1) 设置img控件的ID，请参考上面控件的设置方法，
- 2) 设置选择图片：选择要显示的图片；
- 3) 设置图片位置：图片存放的路径及图片名称。


8. div控件

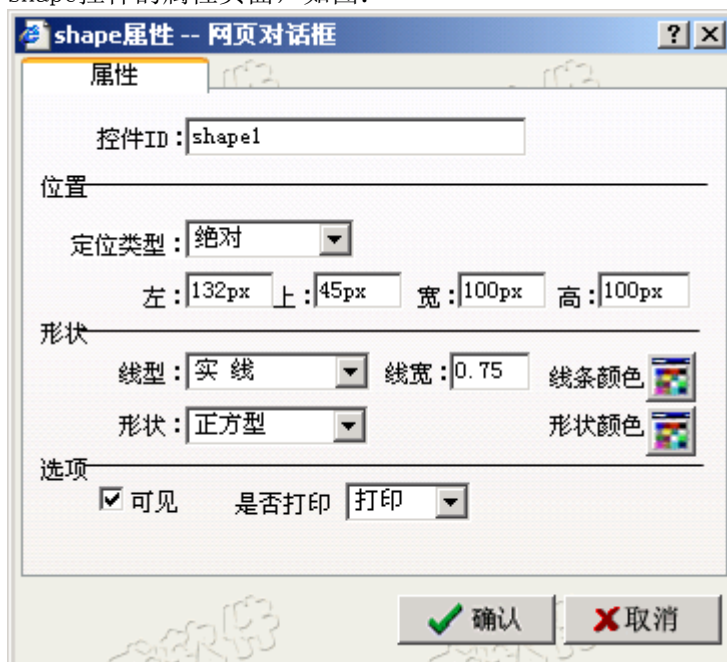
- 点击工具栏中的“”按钮，增加div控件，双击div控件，打开div控件的属性页面，如图：



- 1) 设置控件ID，请参考上面的控件ID。
- 2) 设置滚动轴：设置或检索控件的滚动轴，自动：根据Div的内容自动调节是否显示滚动轴；隐藏：隐藏滚动轴；显示：显示滚动轴。


9. shape控件

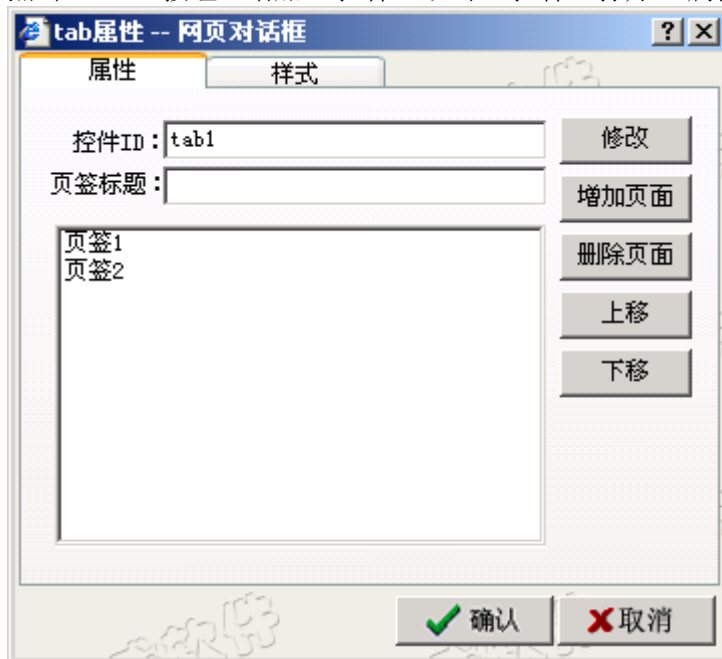
- shape为图形控件，点击工具栏上的“”按钮，增加shape控件, 双击打开shape控件的属性页面，如图：



- 1) 设置控件的ID，请参考上面控件的设置方法；
- 2) 设置控件的线型：shape图形的边框线类型，分为实线、点虚线、破折线、破折点破折线、破折点破折线。
- 3) 设置控件的形状：图形形状，分为圆形、椭圆形、正方形、长方形、圆角正方形、圆角长方形、横线、竖线、左斜线、右斜线、上箭头、下箭头、左箭头、右箭头、左上斜线、左下斜线、右上斜线、右下斜线。


10. tab控件

- 点击“”按钮，增加tab控件，双击tab控件，打开tab属性页面，如图：



- 1) 设置tab控件的ID，请参考上面控件的设置方法。
- 2) 设置页签标题：当选中下面列表的某个选项时，修改标题文本框内的值，点击“修改”按钮，修改标题。
- 3) 增加页面：点击“增加页面”按钮，即可以增加页面。
- 4) 删除页面：选中列表内的某个选项，点击“删除页面”即可删除当前选中页面。
- 5) 上下移按钮，选中页签标题，移动页签的位置。

11. a控件

- 点击“”按钮，增加a控件，双击打开a控件的属性页面，如图：

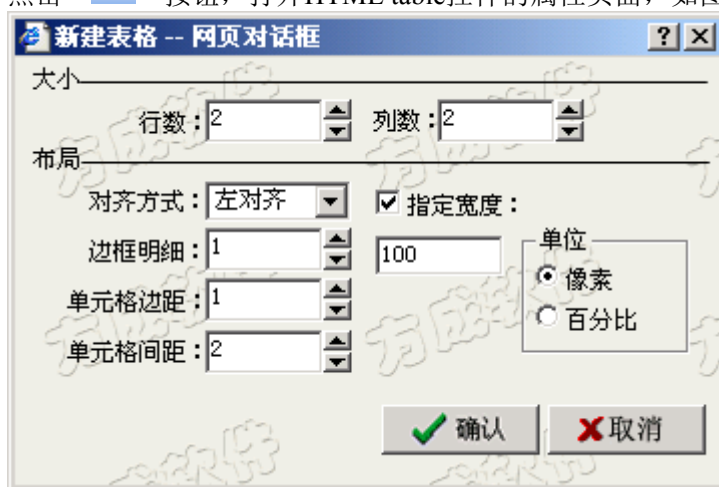


- 1) 设置控件ID，请参考上面控件的设置方法。
- 2) 设置连接地址：输入连接到的地址，示例：<http://www.fcsoft.com.cn>;
- 3) 设置显示标题：控件显示的问题。

4) 设置打开位置：设置打开的连接的位置。


13. HTML table控件

- 点击“”按钮，打开HTML table控件的属性页面，如图：



- 1) 设置新建表格大小：行数：表示新建表格的总行数；列数：表示新建表格的总列数。
- 2) 设置表格布局
 - a) 设置对齐方式：新建表格的单元格部分的显示内容的对齐方式；本平台提供左对齐（align=left）、右对齐(align=right)、水平居中(align=center)。
 - b) 设置边框明细：新建表格的边框线的粗细程度；单位为像素。
 - c) 设置单元格边距：新建表格的边框线与单元格的边距；单位为像素。
 - d) 设置单元格间距：新建表格的边框线与单元格的间距；单位为像素。
 - e) 指定宽度：选中表示指定宽度，反之表示不指定宽度，按默认宽度走。本平台提供像素、百分比。无论选择哪个指定宽度，都是编辑框中输入阿拉伯数字。

14. 版式和表格向导

- 点击“”按钮，打开版式和表格向导属性页面，如图：

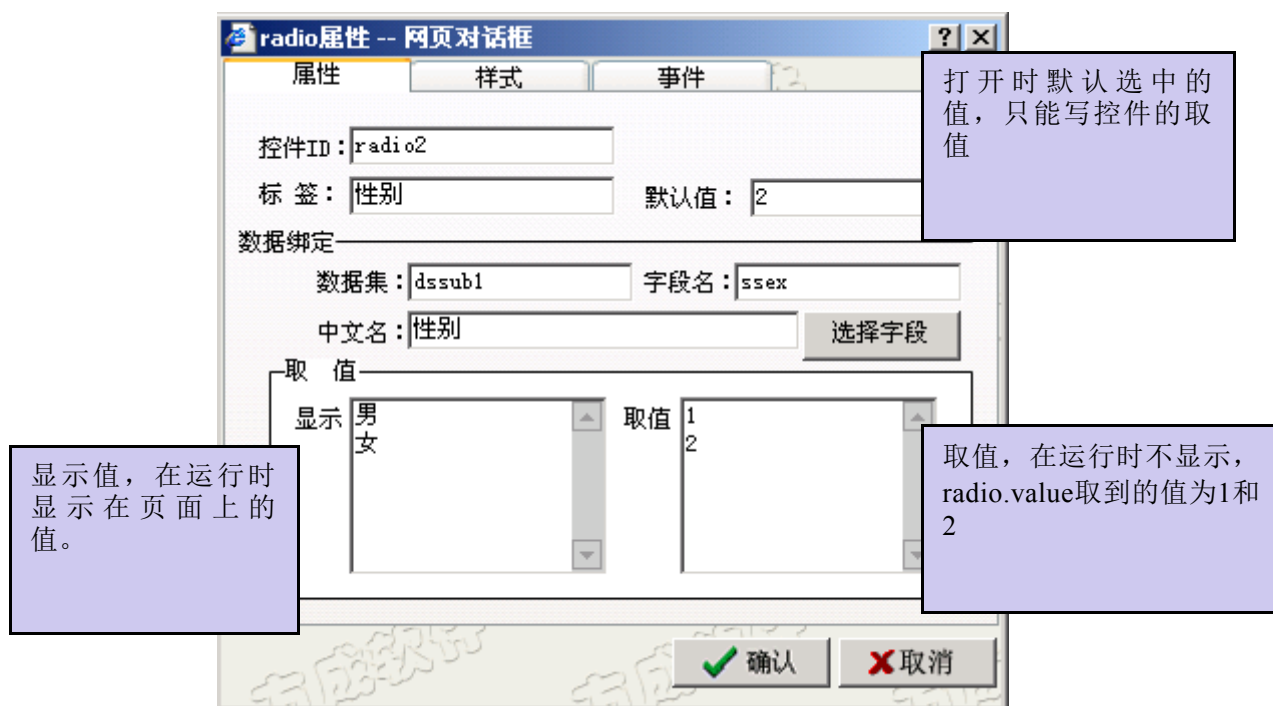


1) 设置插入版式表格样式

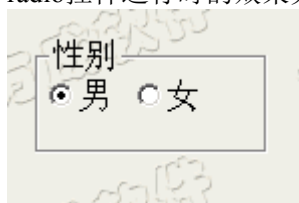
- a) 一列表格：列数只有一列的表格。当选中任意一种表格，下面的表格样式图呈现的样式图就是当前选中的表格样式。
- b) 两列表格：列数只有两列的表格。当选中任意一种表格，下面的表格样式图呈现的样式图就是当前选中的表格样式。
- c) 三列表格：列数只有三列的表格。当选中任意一种表格，下面的表格样式图呈现的样式图就是当前选中的表格样式。
- d) 自定义表格：本平台提供普通表格、表格1，表格2，表格3，表格4。当选中任意一种表格，下面的表格样式图呈现的样式图就是当前选中的表格样式。

15. radio控件

- 以“ssex”字段为例，选择“ssex”字段的编辑格式为radio控件,设置radio控件的ID、标签、默认值和数据绑定，请参考上面text控件的设置方法，设置radio控件的显示值和取值，如图：

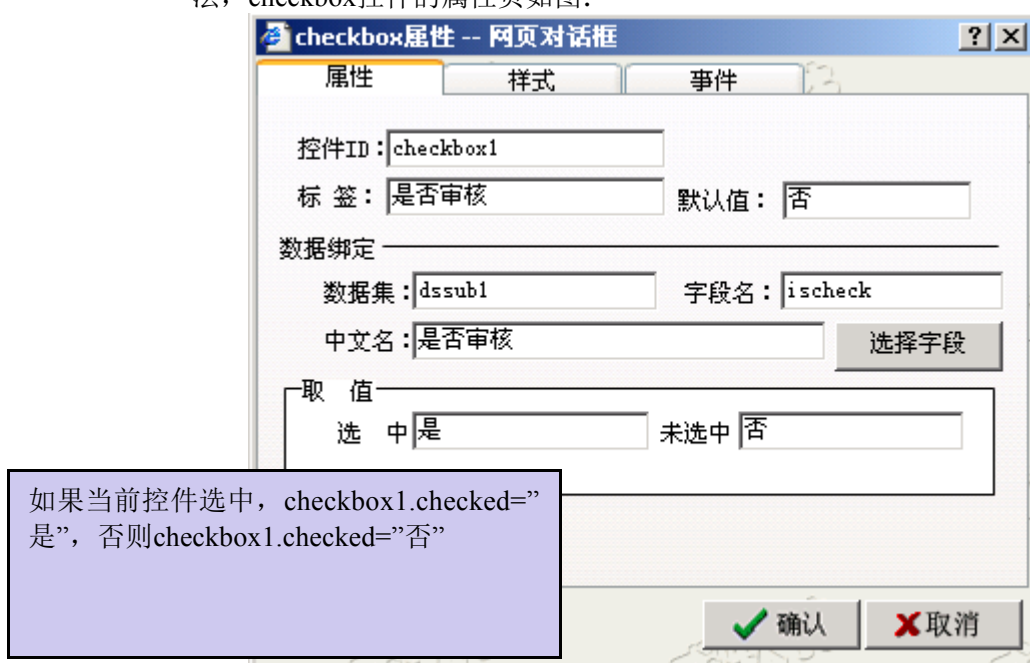


- radio控件运行时的效果如图：



16. checkbox控件

- 以“ischeck”字段为例，选择“ischeck”字段的输入格式为checkbox控件，设置checkbox控件的ID、标签、取值、默认值和数据绑定请参考radio控件的设置方法，checkbox控件的属性页如图：



- checkbox控件运行时的效果如图：

☐ 是否审核

17. combobox控件

- 以“sculture”字段为例，选择“sculture”字段的编辑格式为combobox控件，设置combobox控件的数据绑定、取值方式和默认值；数据绑定和默认值请参考radio控件的设置方法；combobox控件属性页如图：

标准格式是由常数组成的，SQL语句是从数据库中查询的数据

combobox属性 -- 网页对话框

属性 样式 事件

控件ID: combobox1

数据绑定

数据集: dssub1 字段名: sculture

中文名: 学历 选择字段

取值方式

☒ 标准格式 ☐ SQL语句 ? 默认值:

显示 高中 取值 3

职高 4

大专 5

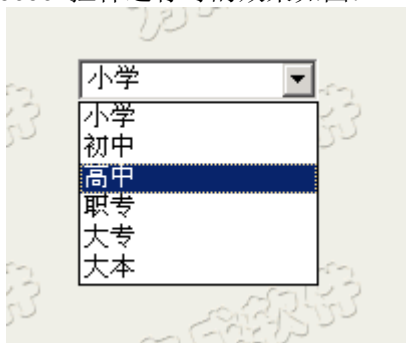
大本 6

博士 7

硕士 8

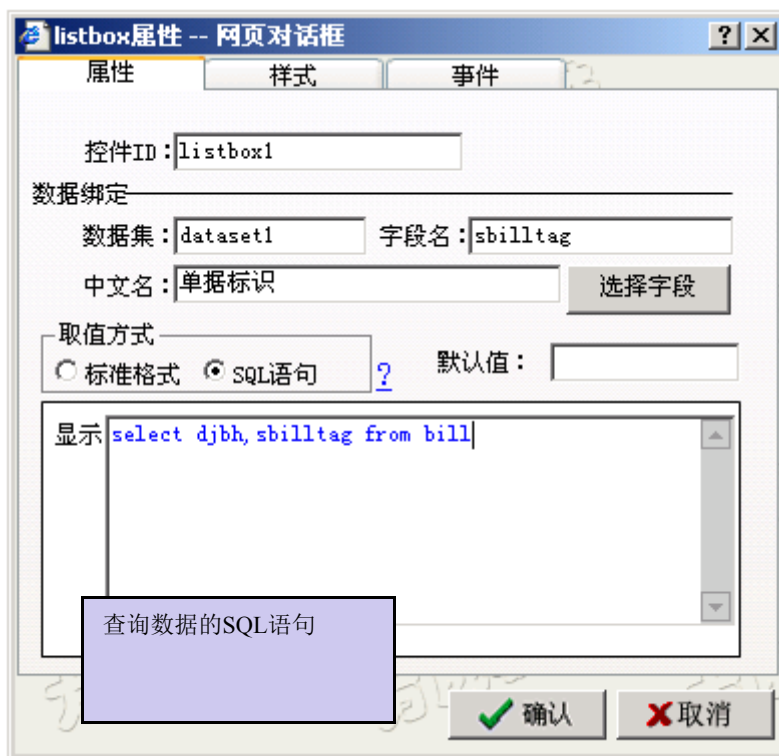
确认 取消

- combobox控件运行时的效果如图：

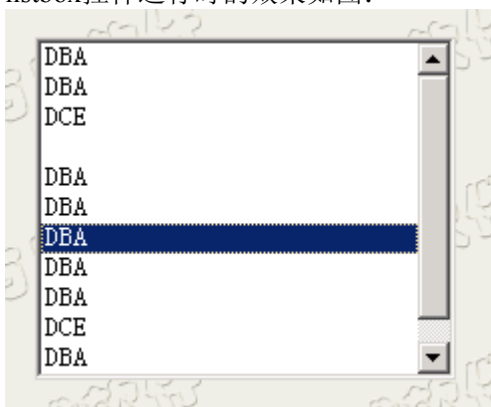


18. listbox控件

- 以“sbilltag”字段为例，选择“sbilltag”字段的输入格式为listbox控件，设计listbox控件的ID、数据绑定、默认值和取值方式，设置控件的数据绑定、ID、默认值、标签请参考combobox控件的设置方法，listbox控件的属性页如图：



- listbox控件运行时的效果如图：



19. dropdownlist控件

- 以“sdwcode”和“sdwname”两个字段为例，选择“”和“”字段的编辑格式为dropdownlist控件，设置控件的ID、标题、选项、数据绑定和取值方式，请参考上面控件的设置方法，dropdownlist控件的属性页如图：

dropdownlist属性 -- 网页对话框

属性 样式 事件

控件ID: dropdownlist1

标题: |单位编号 | 单位名称!

数据绑定

数据集: dssub1 字段名: sdwna

中文名: 单位名称

选项

☒ 新增 ☐ 多选 ☐ 空行 ☒ 直接输入 ?

取值方式

☐ 标准格式 ☒ SQL语句 ?

显示

select sdwid, sdwcode, sdwname from dw

确认 取消

有多个标题用“|”分开，采用同名字段复制方法

空行：控制下拉列表中是否有空行，只针对取值方式为SQL语句。直接输入：控制文本列表是否能直接输入。

新增：控制打开一张表单，对编辑框赋值；
多选：控制在下拉列表中有没有有一列checkbox控件，可以同时选择多个值。

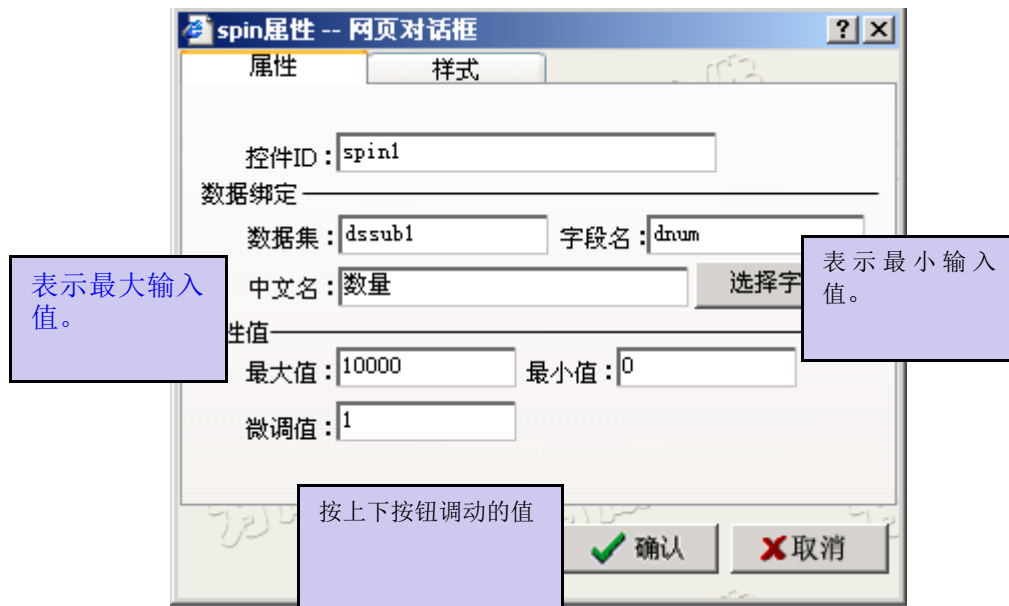
- dropdownlist控件运行时的效果如图：

单位编号	单位名称
...+...	...+...
B00007	咸阳市秦都向群中西经营部
A00001	美国IBM公司中国北京分理处
C001	河南制药三分公司公司
001	联想公司
B00002	杭州前江
B000071	咸阳市秦都
W09008	特富丽药业有限公司
AB992745927323	中科

“...+...”表示新增行，点击当前行，打开一张表单

20. spin控件

- 以“dnum”字段为例，选择“dnum”字段的编辑格式为spin控件，设置控件的ID和数据绑定请参考上面控件的设置方法，设置控件的属性值，spin控件的属性页如图：



- spin控件运行时的效果如图:



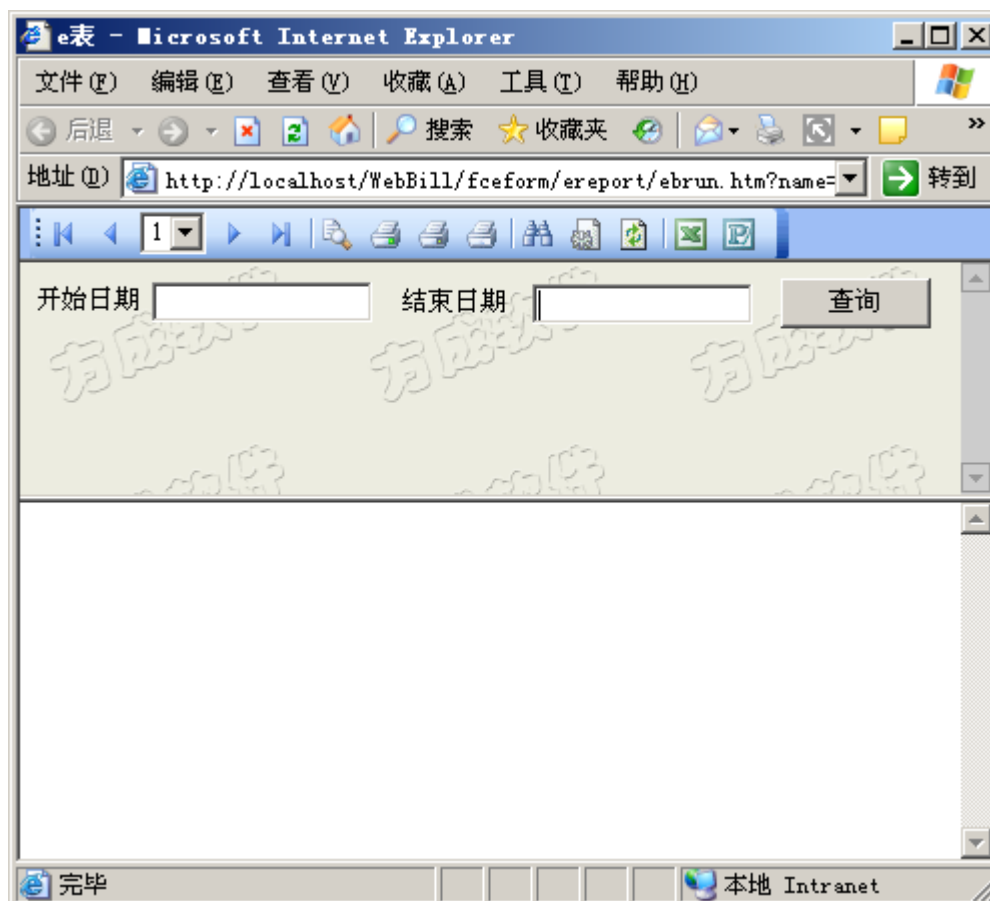
e表中新加的功能

1. 表单属性

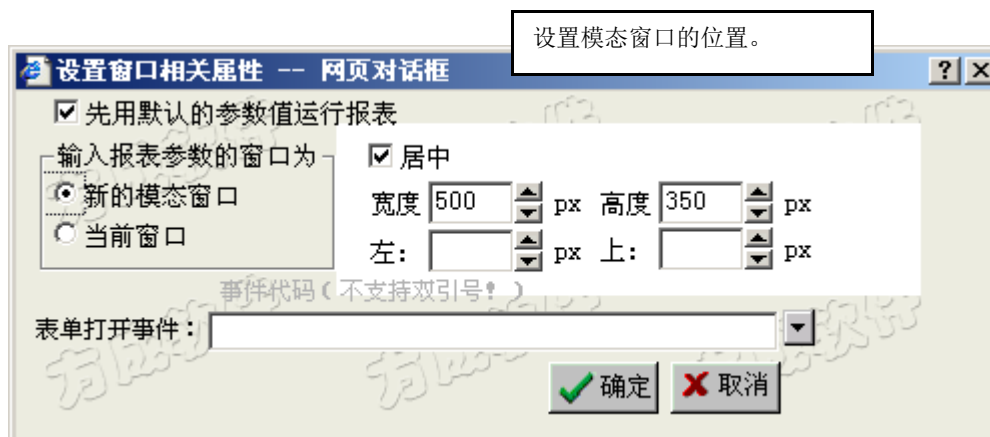
在这里用来设置输入报表参数表单的窗口是用一个模态窗口显示，如下图的样子：



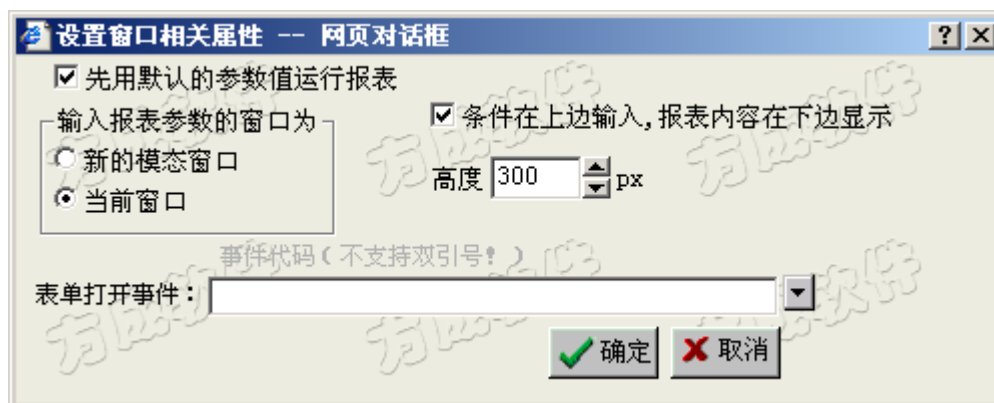
还是在当前窗口显示，如下图的样子：



模态窗口时的界面图如下：



当前窗口时的界面图如下：



2. 报表参数绑定

将可输入的控件（如text,textarea,radio,checkbox,combobox,listbox,dropdownlist,spin）绑定到报表参数后，则控件上的值会自动作报表参数的值。
一个报表参数只能绑定到一个控件上。



点此会将当前控件ID绑定到当前行的报表参数上。

(九)统计图

下面是统计图的界面：

默认的统计图类型，即当系列中图形类型为空就用这个统计图类型。

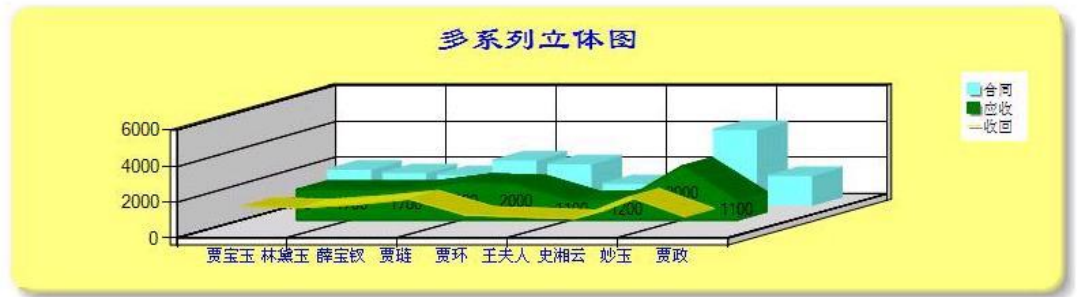


点[样式]页签的界面图如下:



下图为统计图的运行界面：

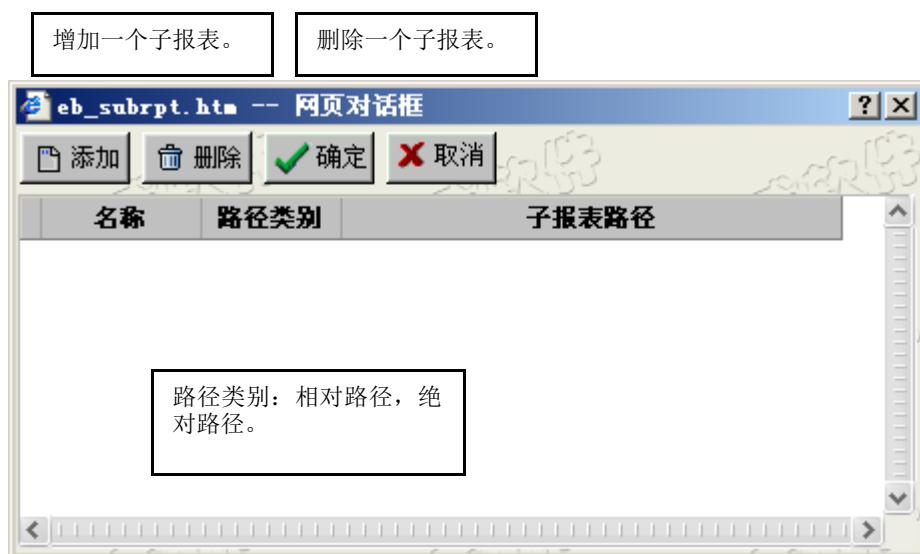
合同编号	经手人	合同金额	应收金额	收回金额
1	贾宝玉	2000	1700	1600
2	林黛玉	1800	1700	1500
3	薛宝钗	1700	1700	1700
4	贾琏	2500	2100	2000
5	贾环	2300	2000	1100
6	王夫人	1200	1100	1000
7	史湘云	1290	1200	900
8	妙玉	4200	3000	2100
9	贾政	1650	1100	1000



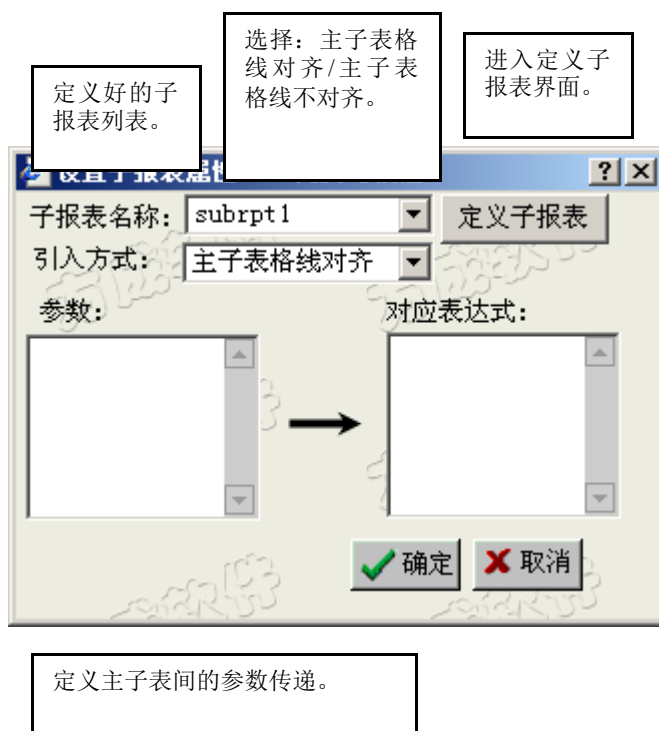
(十)子报表

在业务逻辑中，常常会出现一张报表由多张子报表组成，这多张子报表与母报表之间可能有业务联系，也有可能完全没有联系。子报表总是位于母报表的某个单元格中，该单元格可以是合并格。子报表本身也可以再包含子报表，因此，子报表可以一层一层的嵌套下去。

下图是子报表的定义界面：



下图是设置子报表的界面图：



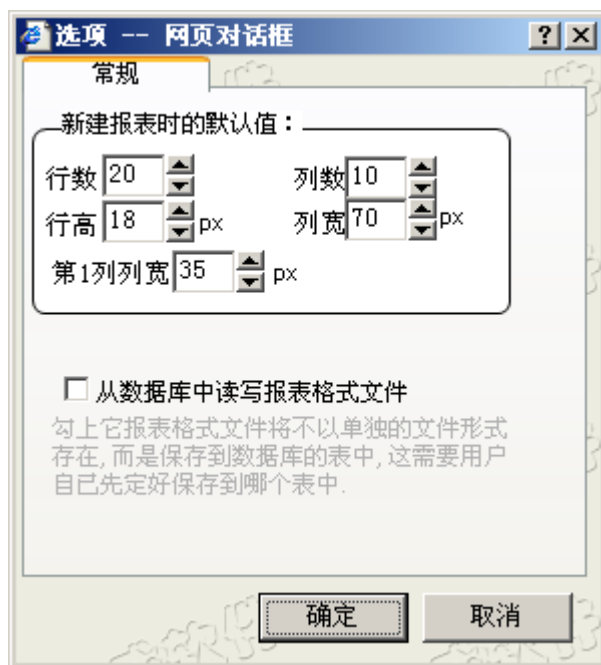
注意：

使用子报表时，要特别注意。因为主子报表共用了运行环境，所以主子报表的数据集的名称不能相同，比如主报表有一个数据集叫ds1，则子报表就不能有数据集叫ds1。还有系统的数据库连接是以主报表的为准，所以子报表不能使用额外的数据库连接。比如：主报表的数据集都是内建数据集时，子报表就不能有SQL语句类型的数据集。

(十一)其它

选项

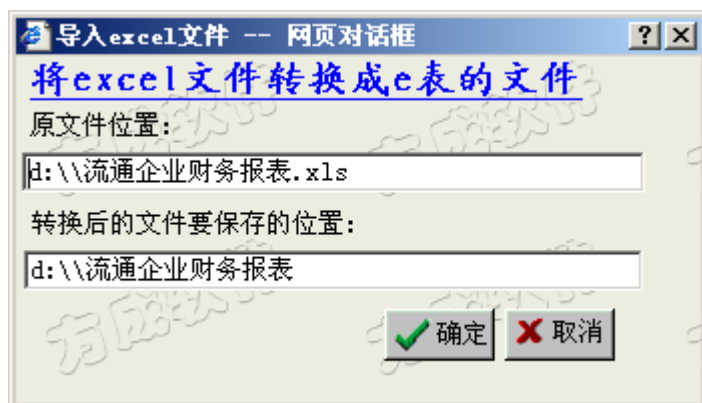
它是报表设计器的配置信息，它的界面图如下：



这些配置信息是独立于任何一个报表文件的，它保存在 fceform/ereport/econfig.xml 文件中。

导入excel文件

现实中用户可能已经有很多用excel画好的报表文件，如何需要在e表系统中重画，就太麻烦了。E表提供一个工具可以直接将excel文件生成e表格式的报表文件。使用这个转换工具和导出excel文件一样，不要求在客户端或服务器端安装excel软件。



(十二)条形码

下面是e表 for .NET版的条形码的界面：



The screenshot shows a dialog box titled "条形码属性窗口 -- 网页对话框" (Barcode Properties Window -- Web Dialog Box). The dialog is divided into two main sections: "属性" (Properties) and "数据" (Data). The "属性" section contains the following settings:

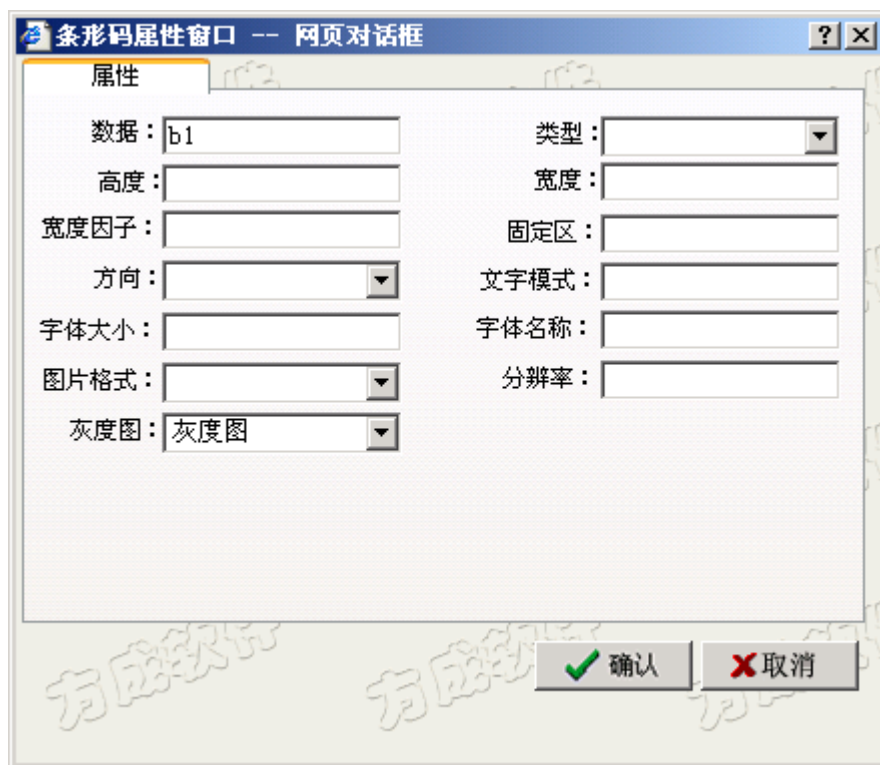
- 类型 (Type): CODE39
- 标题 (Title):
- 附加标题 (Additional Title):
- 宽度 (Width): 1
- 高度 (Height): 40
- 图片格式 (Image Format): GIF
- 方向 (Orientation): BottomFacing
- 背景色 (Background Color): White
- 前景色 (Foreground Color): #000000

The "数据" section contains the following settings:

- 标题位置 (Title Position): Above
- 附加标题位置 (Additional Title Position): Above
- 数据 (Data): a3
- 附加数据 (Additional Data): 55
- 字体名称 (Font Name): Arial
- 字体大小 (Font Size): 9
- 字体样式 (Font Style): Bold
- 无效数据操作 (Invalid Data Operation): DisplayNone

At the bottom of the dialog, there are two buttons: "确认" (Confirm) with a green checkmark icon and "取消" (Cancel) with a red X icon.

下面是e表 for Java版的条形码的界面：



使用方法主要是设置一下数据，数据一般来一个公式。再设置一下条形码的类型，高度，宽度。这样就可以产生条形码的图片。

五 函数说明

(一)数据集函数

get

函数说明:

从当前数据组(由其顶格决定)中选取符合条件的记录

语法:

```
datasetName.get( <get_exp>[, desc_exp, filter_exp, sort_exp] )
```

功能说明:

数据源从数据库中读取的记录实际上是一个行列的二维表。

get()函数从数据源记录集中选取get_exp字段列中符合过滤条件的值。

在不需要排序或数据集中已经按需要排好序时，为提高效率请如下使用此函数

```
datasetName.get( <get_exp>, , <filter_exp> )
```

参数说明:

get_exp: 要选择的字段列，可以是字段列名，也可以是以“#列序号”的形式，#0表示记录行号，#1表示数据源中的第一个字段列.....以此类推。当然也可以是表达式

desc_exp: 指定数据排序的顺序，true表示降序排列，false表示升序排列。

filter_exp: 数据过滤表达式。

sort_exp: 数据排序表达式。当此项为空时先检查desc_exp是否为空，如果为空，则不排序，否则使用get_exp排序。

函数示例:

```
ds1.get( fieldname )
```

从数据源ds1中选取fieldname字段列的所有值, 不排序

```
ds1.get( #3, true )
```

从数据源ds1中选取第3个字段列的所有值并降序排列

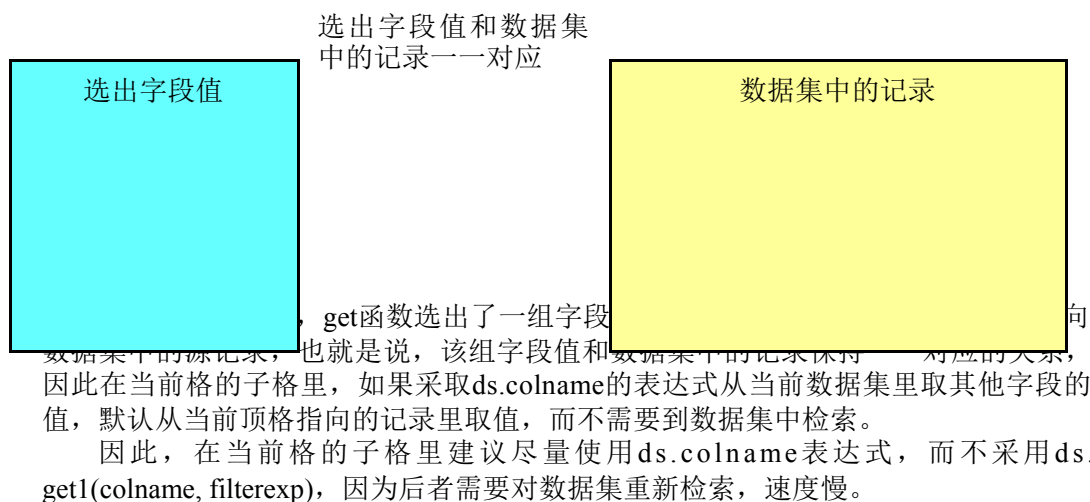
```
ds1.get( name, false, sex='1' )
```

从数据源ds1中选取性别为男性的name字段列的值并升序排列

```
ds1.get( name, true, sex='1', id )
```

从数据源ds1中选取性别为男性的name字段列的值并按id字段降序排列

图例:

**返回值：**

一组数据的集合

group**函数说明：**

根据分组表达式，从数据集中选出符合过滤条件的一组组集。

语法：

```
datasetName.group( <select_exp>[, desc_exp, filter_exp, sort_exp] )
```

功能说明：

在不需要排序或数据集中已经按需要排好序时，为提高效率请如下使用此函数

```
datasetName.group( <select_exp>, <filter_exp> )
```

参数说明：

- select_exp:** 选出的分组表达式，可以是字段列名，也可以是以“#列序号”的形式，#0表示记录行号，#1表示数据源中的第一个字段列.....以此类推。当然也可以是表达式
- desc_exp:** 指定数据排序的顺序，true表示降序排列，false表示升序排列。
- filter_exp:** 数据过滤表达式。
- sort_exp:** 数据排序表达式。当此项为空时先检查desc_exp是否为空，如果为空，则不排序，否则使用select_exp排序。

函数示例：

```
ds1.group( name )
```

从数据源ds1中选取name字段，并按照name列进行分组，取出每一组第一条记录的name字段的值, 不排序

```
ds1.group( #2, true )
```

从数据源ds1中选取第二个字段并降序排列，然后按照第二列进行分组，取出每一组的第一条记录的第二个字段值

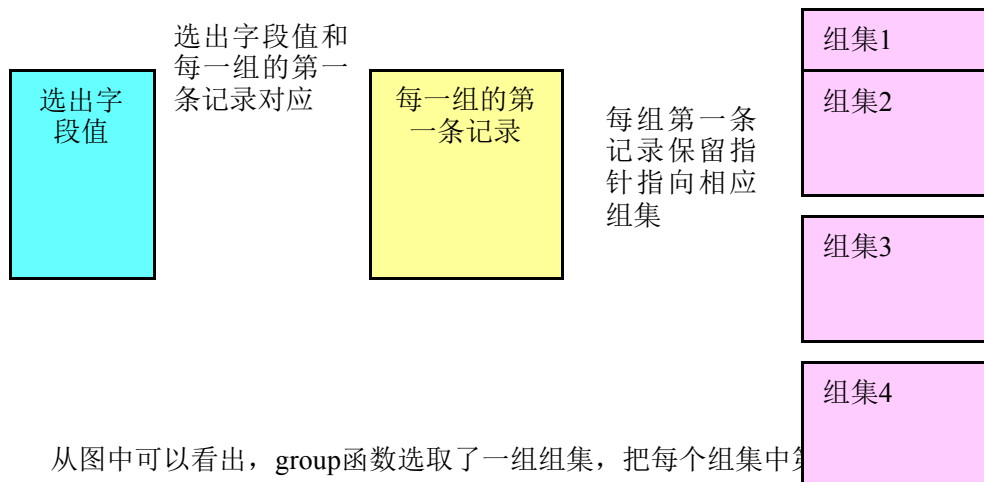
```
ds1.group( name, false, sex='1' )
```

从数据源ds1中选取性别为男性的name字段列的值并升序排列，然后按照name字段进行分组，取出每一组第一条记录的name字段的值

```
ds1.group( name, true, sex='1', id )
```

从数据源ds1中选取性别为男性的name字段列的值并按id字段降序排列，然后按照name字段进行分组，取出每一组第一条记录的name字段的值

图例：



从图中可以看出，group函数选取了一组组集，把每个组集中第一条记录的值取出来，获得该记录的selectExp字段值，该字段值保留了一个指针，指向每一组的第一条记录，而该记录保留了一个指针指向相应的组集。因此在当前格的子格里，如果采取get、group函数，则缺省从当前格指向的组集里取数；如果采取dsn.colname的表达式，则缺省从当前格指向的组集的第一条记录里取数。

因此，在当前格的子格里如果采取get/group函数，不需要再加上和顶格相关的过滤条件。

返回值：

一组数据的集合

getall

函数说明：

从根数据组(即包含数据集所有记录的组，而不是当前组集)中选取符合条件的记录，其余说明参见get函数

语法：

`datasetName.getall(<select_exp>[, desc_exp, filter_exp, sort_exp])`

参数说明：

select_exp: 要选择的字段列，可以是字段列名，也可以是以“#列序号”的形式，#0表示记录行号，#1表示数据源中的第一个字段列.....以此类推。

desc_exp: 指定数据排序的顺序，true表示降序排列，false表示升序排列。

filter_exp: 数据过滤表达式。

sort_exp: 数据排序表达式。当此项为空时先检查desc_exp是否为空，如果为空，则不排序，否则使用select_exp排序。

返回值：

一组数据的集合

groupall

函数说明：

重新取得数据集中所有数据(即从根数据组中，而不是当前组集)，并分组，其余说明请参见group()

语法：

`datasetName.groupall(<select_exp>[, desc_exp, filter_exp, sort_exp])`

功能说明：

在不需要排序或数据集中已经按需要排好序时，为提高效率请如下使用此函数
`datasetName.groupall(<select_exp>, <filter_exp>)`

参数说明：

`select_exp`: 选出的分组表达式，可以是字段列名，也可以是以“#列序号”的形式，#0表示记录行号，#1表示数据源中的第一个字段列.....以此类推。
 当然也可以是表达式
`desc_exp`: 指定数据排序的顺序，`true`表示降序排列，`false`表示升序排列。
`filter_exp`: 数据过滤表达式。
`sort_exp`: 数据排序表达式。当此项为空时先检查`desc_exp`是否为空，如果为空，则不排序，否则使用`select_exp`排序。

返回值：

一组数据的集合，该组数据由数据集根据选出字段分组而来

get1

函数说明：

从数据集中根据选出字段或表达式以及选出条件，选出一个数据

语法：

`datasetName.get1(selectExp{,filterExp})`

参数说明：

`selectExp` 选出字段或表达式
`filterExp` 过滤条件

sum

函数说明：

从数据集中，从满足条件的记录中，算出给定字段或表达式的总和

语法：

`datasetName.sum(selectExp{,filterExp})`

参数说明：

`selectExp` 需求字的字段或表达式
`filterExp` 条件表达式

count

函数说明：

计算数据集中，满足条件的记录数

语法：

`datasetName.count({filterExp})`

参数说明：

`filterExp` 条件表达式

avg

函数说明：

从数据集中，从满足条件的记录中，算出给定字段或表达式的平均值

语法：

`datasetName.avg(selectExp{,filterExp})`

参数说明：

`selectExp` 需要计算平均值的字段或表达式
`filterExp` 过滤条件表达式

max**函数说明：**

从数据集中，从满足条件的记录中，选出给定字段或表达式的最大值

语法：

`datasetName.max(selectExp{,filterExp})`

参数说明：

selectExp	需要获得最大值的字段或表达式
filterExp	过滤表达式

min**函数说明：**

从数据集中，从满足条件的记录中，选出给定字段或表达式的最小值

语法：

`datasetName.min(selectExp{,filterExp})`

参数说明：

selectExp	需要获得最小值的字段或表达式
filterExp	过滤表达式

last**函数说明：**

从数据集满足条件的记录集合中，选出最后一条记录，返回给定字段或表达式的值

语法：

`datasetName.last(selectExp{,descExp{,filterExp{,sortExp}}})`

参数说明：

selectExp	选出字段或表达式
descExp	排序的顺序，true代表逆序，false代表顺序
filterExp	过滤条件表达式
sortExp	排序依据表达式

first**函数说明：**

从数据集满足条件的记录集合中，选出第一条记录，返回给定字段或表达式的值

语法：

`datasetName.first(selectExp{,descExp{,filterExp{,sortExp}}})`

参数说明：

selectExp	选出字段或表达式
descExp	排序的顺序，true代表逆序，false代表顺序
filterExp	过滤条件表达式
sortExp	排序依据表达式

cols**函数说明：**

数据集的列数

语法：

`datasetName.cols()`

field**函数说明：**

取数据集的列

语法：

`datasetName.field(stringExp)`

`datasetName.field(intExp)`

参数说明:

stringExp 返回数据集列名的表达式
intExp 返回数据集列号的表达式

scope

函数说明:

根据完全划分进行分组

语法:

`datasetName.scope(selectExp, listExp[, filterExp][, eqExp][, ascExp])`

参数说明:

selectExp 取值表达式
ListExp 返回同valueExp数据类型相同的数组, 要求其中元素从小到大排列
filterExp 过滤表达式
eqExp 返回布尔值的表达式, 缺省为false, 表示与元素比较时不包含等于
ascExp 返回布尔值的表达式, 缺省为true, 表示listExp返回的数组按从小到大排, 否则为从大到小排

特别说明:

ascExp为true(即listExp从小到大排列)时, eqExp为true时, 与元素比较时采用<=, eqExp为false时, 采用<;

ascExp为false(即listExp从大到小排列)时, eqExp为true时, 与元素比较时采用>=, eqExp为false时, 采用>

示例:

`ds1.scope(year(now())-year(sbirthday),[15,30,40,50])`

(二)单元格函数

sum

函数说明:

对可扩展单元格或集合表达式进行求和

语法:

`sum(expression)`

参数说明:

expression 需要被求和的单元格或表达式, 一般为可扩展单元格或集合表达式

count

函数说明:

对可扩展单元格或集合表达式进行计数

语法:

`count(expression)`

参数说明:

expression 需要被计数的单元格或表达式, 一般为可扩展单元格或集合表达式

avg

函数说明:

对可扩展单元格或集合表达式求平均值

语法:

`avg(expression)`

参数说明:

expression 需要求平均值的单元格或表达式, 一般为可扩展单元格或集合

表达式

max

函数说明:

对可扩展单元格或集合表达式求最大值

语法:

`max(expression)`

参数说明:

expression 需要求最大值的单元格或表达式，一般为可扩展单元格或集合

表达式

min

函数说明:

对可扩展单元格或集合表达式求最小值

语法:

`min(expression)`

参数说明:

expression 需要求最小值的单元格或表达式，一般为可扩展单元格或集合

表达式

row

函数说明:

取得当前格所有行的行号

请注意，此函数只能在get、group等扩展函数之后使用，否则取出的行号可能不

正确

语法:

`row()`

col

函数说明:

取得当前单元格所在列的列号

请注意，此函数只能在get、group等扩展函数之后使用，否则取出的列号可能不

正确

语法:

`col()`

displayvalue

函数说明:

取单元格的显示值

语法:

`displayvalue (cell)`

参数说明:

cell 单元格

maxwidth

函数说明:

求当前列中所有非合并格的显示串的最大长度（注意：ASCII码大于255的字符长度为2）

语法说明:

`maxwidth()`

(三)常用函数

list

函数说明:

获得一个枚举的数据集合

语法:

`list(valueExp1 {,valueExp2 {,valueExp3 {,.....}}})`

参数说明:

valueExp(n) 可以是常数或表达式

返回值:

一组数据的集合

map

函数说明:

显示值对照表函数:, 从对照表中找出当前单元格对应值的显示值, 没有则返回

null

语法:

`map(valueListExp, displayListExp)`

参数说明:

valueListExp 真实值列表, 可以是可扩展单元格或结果为集合列表的表达式

displayListExp 显示值列表, 可以是可扩展单元格或结果为集合列表的表达式

真实值列表和显示值列表一一对应

sort

函数说明:

对数组进行排序

语法:

`sort(arrayExp[, boolExp])`

参数说明:

arrayExp 返回数组的表达式, 譬如group, select, list等函数

boolExp 布尔表达式, true表示对数组按升序排列, 否则降序

if

函数说明:

根据布尔表达式的不同结果, 返回不同的值

语法:

`if(boolExp, trueValueExp, falseValueExp)`

参数说明:

boolExp 结果为布尔类型的表达式

trueValueExp 布尔表达式为真时, 返回本参数的计算结果, 本参数可以是常数或表达式

falseValueExp 布尔表达式为假时, 返回本参数的计算结果, 本参数可以是常数或表达式

ifnull

函数说明:

根据第一个表达式的值是否为空, 返回其他值

语法:

`ifnull(valueExp1, valueExp2)`

参数说明:

valueExp1 需要计算的表达式, 其结果不为空时返回其值
valueExp2 需要计算的表达式, 当valueExp1结果为空时返回此值

case

函数说明:

根据布尔表达式的不同计算结果, 返回不同的值。本函数:从左到右计算, 先出现的布尔表达式先算, 如果出现为true的表达式, 则返回相应的结果, 后面的不再计算。如果没有一个布尔表达式为true,而且有缺省值表达式, 则返回缺省值, 否则返回null。

语法:

`case(boolExp1,valueExp1 {,boolExp2,valueExp2 {,boolExp3,valueExp3 {, {defaultExp}}})`

参数说明:

boolExp(n) 布尔表达式, 如果结果为真, 则返回对应的值表达式计算结果
valueExp(n) 值表达式, 和布尔表达式一一对应
defaultExp 缺省值表达式, 如果所有布尔表达式结果都为假, 则返回本表达式计算结果

sql

函数说明:

执行sql语句, 返回结果数据集合, 只能返回单列数据, 如果sql语句中有多个字段, 则返回第一个字段的结果值

语法:

`sql(sqlStatement{,arg1 {,arg2 {,arg3 {,.....}}})`

参数说明:

sqlStatement 合法的sql语句
arg(n) sql语句的参数, 可以是常数也可以是表达式

返回值:

一组数据的集合

dbsql

函数说明:

执行sql语句, 返回结果数据集合, 只能返回单列数据, 如果sql语句中有多个字段, 则返回第一个字段的结果值

语法:

`dbsql(dbname, sqlStatement{,arg1 {,arg2 {,arg3 {,.....}}})`

参数说明:

dbname 数据库逻辑名, 为null表示缺省数据库
sqlStatement 合法的sql语句
arg(n) sql语句的参数, 可以是常数也可以是表达式

返回值:

一组数据的集合

call

函数说明:

执行存储过程, 返回结果数据集合, 只能返回单列数据, 如果存储过程返回多个字段, 则返回第一个字段的结果值

语法:

`call(sqlStatement{,arg1 {,arg2 {,arg3 {,.....}}})`

参数说明:

sqlStatement 合法的调用存储过程的sql语句
arg(n) sql语句的参数, 可以是常数也可以是表达式,

如果是输出结果集的参数，对应的参数表达式写成"@result"

举例：

```
call("{call ResultSet (?,?,?)}","@result","000001","A2")
```

dbcall

函数说明：

执行存储过程，返回结果数据集合，只能返回单列数据，如果存储过程返回多个字段，则返回第一个字段的结果值

语法：

```
dbcall(dbname, sqlStatement {,arg1 {,arg2 {,arg3 {,.....}}})
```

参数说明：

dbname	数据库逻辑名，为null表示缺省数据库
sqlStatement	合法的调用存储过程的sql语句
arg(n)	sql语句的参数，可以是常数也可以是表达式， 如果是输出结果集的参数，对应的参数表达式写成"@result"

ds

函数说明：

按名称取得数据集

语法：

```
ds( stringExp )
```

参数说明：

stringExp 返回字符串的表达式

scope

函数说明：

计算一个值在一个完全划分中的位置

语法：

```
scope( valueExp, ListExp[, eqExp][, ascExp] )
```

参数说明：

valueExp	返回值的表达式，值可以是字符串、数值、日期、时间等
ListExp	返回同valueExp数据类型相同的数组，要求其中元素从小到大排列
eqExp	返回布尔值的表达式，缺省为true，为true表示与ListExp元素比较时

包含等于

ascExp	返回布尔值的表达式，缺省为true，即表示ListExp中元素从小到大排列，否则为从大到小排列
--------	---

函数示例：

```
plot( 0, [0,10,100] ) 返回1
```

```
plot( 0, list(0,10,100), true ) 返回0
```

详细说明：

划分是指将一个集合划分成几个集合，如数组[0,10,100]在eqExp返回false的情况下将整数集或实数集划分成4个集合，

依次分别是<0、<10(此集合中元素必定≥0)、<100(此集合中元素必定≥10)及其它(即≥100)；在eqExp返回true的情况

下将整数集或实数集划分成4个集合，分别是≤0、≤10(且>0)、≤100(且>10)及其它(即>100)；对于数组[100,10,0]，

eqExp为false时也划分成4个集合，分别为>100,>10(且≤100),>0(且≤10)及其它(即≤0)；eqExp为true时则为≥100，

≥10(且<100),≥0(且<10)及其它(即<0)

eval

函数说明：

计算表达式

语法:

`eval(StringExp)`
`eval(StringExp, SubRptExp)`
`eval(StringExp, DataSetExp)`

参数说明:

StringExp 返回字符串的表达式
 SubRptExp 返回嵌入式子报表的表达式
 DataSetExp 返回数据集的表达式

函数示例:

`eval("1+5")` 返回6
`ds1.count(eval("id > 1 and id < 10"))` 返回数据集ds1中id大于1且小于10的记录个数

特殊使用:

如A1为嵌入式子报表, 要取A1中子报表中B2的值加10, 则表达式为

`eval("B2+10", A1)`或`eval("B2",A1)+10`

如ds1为数据集名, 计算ds1中salary加100, 则表达式为`eval("salary+100", ds("ds1"))`, 它与`eval("ds1.salary+100")`等效

getmaindir

函数说明:

取得后台主目录的绝对路径.

语法:

`getmaindir()`

参数说明:

无

函数示例:

`getmaindir()` 返回c:\inetpub\wwwroot\webbill

(四)字符串函数

fill

函数说明:

获得n个x拼成的字符串

语法:

`fill(s, n)`

参数说明:

s 用于拼成新串的源串
 n 新串中包含源串的个数

indexof

函数说明:

查找母串中子串的位置

语法:

`indexof(s1, s2{, begin})`

参数说明:

s1 待查找子串的母串
 s2 被查找的子串
 begin 查找的起始位置, 缺省为0

left**函数说明:**

获得字符串左边的子串

语法:

`left(string, n)`

参数说明:

string 获得子串的源串

n 获得子串的长度

length**函数说明:**

计算字符串的长度

语法:

`length(s)`

参数说明:

s 待计算长度的字符串

likestr**函数说明:**

判断字符串是否匹配格式串(*匹配0个或多个字符, ?匹配单个字符)

语法:

`likestr(stringExp, formatExp[, ignoreCase])`

参数说明:

stringExp 返回字符串的表达式

formatExp 返回格式串的表达式

ignoreCase 返回布尔值的表达式, 缺省为false, 表示匹配时大小写敏感,

否则忽略大小写

函数示例:

`likestr("abc123", "abc*")` 返回true

`likestr("abc123", "abc1?3")` 返回true

`likestr("abc123", "abc*34")` 返回false

`likestr("abc123", "ABC*")` 返回false

`likestr("abc123", "ABC*", true)` 返回true

lower**函数说明:**

将字符串转成小写

语法:

`lower(s)`

参数说明:

s 待转成小写的字符串

ltrim**函数说明:**

去掉字符串左边的空格

语法:

`ltrim(s)`

参数说明:

s 准备去掉左边空格的源串

repstr**函数说明:**

替换字符串，即将源串中的某部分（例如 aa）替换为另外的值（例如 bb）。

语法：

`repstr(srcExp,findStr,repStr,bQuot)`

参数说明：

srcExp 源串
findStr 要查找的字符串
repStr 要替换的子串
bQuot true/false 引号内的字符是否也替换。

返回值：

替换后的字符串

right

函数说明：

获得字符串右边的子串

语法：

`right(s, n)`

参数说明：

s 待获得子串的源串
n 获得子串的起始位置

rtrim

函数说明：

去掉字符串右边的空串

语法：

`rtrim(s)`

参数说明：

s 待去掉右边空串的字符串

substr

函数说明：

返回字符串的子串

语法：

`substr(s, start{, end})`

参数说明：

s 待获得子串的源串
start 获得子串的起始位置
end 获得子串的结束位置，缺省为源串的长度

trim

函数说明：

去掉字符串左右的空串

语法：

`trim(s)`

参数说明：

s 待去掉左右空串的源串

upper

函数说明：

把字符串转成大写

语法：

`upper(s)`

参数说明：

s 待转成大写的源串

(五)数学函数

abs

函数说明:

计算参数的绝对值

语法:

`abs(numberExp)`

参数说明:

numberExp 待计算绝对值的数据

cos

函数说明:

计算参数的余弦值，其中参数以弧度为单位

语法:

`cos(numberExp)`

参数说明:

numberExp 待计算余弦值的弧度数

ceil

函数说明:

对参数进行取整，返回大于或等于原数据的整数。

语法:

`int(numberExp)`

参数说明:

numberExp 需要进行取整的数据

int

函数说明:

对参数进行取整，直接去掉小数部分，返回整数部分

语法:

`int(numberExp)`

参数说明:

numberExp 需要进行取整的数据

log

函数说明:

计算参数的自然对数

语法:

`log(numberExp)`

参数说明:

numberExp 需要计算自然对数的数据

log10

函数说明:

计算以10为底的对数

语法:

`log10(numberExp)`

参数说明:

numberExp 需要计算以10为底的对数的数据

random

函数说明：

取得0-1.0之间的一个随机数

语法：

random()

参数说明：

无

round

函数说明：

对参数指定位置进行四舍五入取整

语法：

round(numberExp, nExp)

参数说明：

numberExp

需要进行四舍五入取整的数据

nExp

指定位置，为正表示小数点后四舍五入，为负表示小数点前四

舍五入

sin

函数说明：

计算参数的正弦值，其中参数以弧度为单位

语法：

sin(number)

参数说明：

number

需要计算正弦值的弧度数

sqrt

函数说明：

计算平方根

语法：

sqrt(number)

参数说明：

number

需要计算平方根的数据

tan

函数说明：

计算参数的正切值，其中参数以弧度为单位

语法：

tan(number)

参数说明：

需要计算正切值的弧度数

(六)日期函数

afterdays

函数说明：

从给定的日期型数据中，算出相差n天后的新的日期数据

语法:

`afterdays(dateExp, nExp)`

参数说明:

<code>dateExp</code>	给定的起始日期表达式，其结果必须为日期或中国日期时间格式的字符串
<code>nExp</code>	整数表达式，需要求得n天后的新日期

afterseconds

函数说明:

从给定的日期时间型数据中，算出相差n秒后的新的日期时间数据

语法:

`afterseconds(datetimeExp, nExp)`

参数说明:

<code>datetimeExp</code>	给定的起始日期表达式，其结果必须为日期或中国日期时间格式的字符串
<code>nExp</code>	整数表达式，需要求得n秒后的新日期时间

date

函数说明:

将字符串转换成日期型数据

语法:

`date(stringExp)`

参数说明:

<code>stringExp</code>	字符串表达式，其结果必须返回合法日期格式的字符串
------------------------	--------------------------

datetime

函数说明:

将字符串或长整数转换成日期时间

语法:

`datetime(string)`

`datetime(long)`

参数说明:

<code>string</code>	合法的日期格式的字符串
<code>long</code>	以微秒计的长整数

day

函数说明:

从日期型数据中获得该日在本月中是几号

语法:

`day(dateExp)`

参数说明:

<code>dateExp</code>	表达式，其结果必须为日期或中国日期时间格式的字符串
----------------------	---------------------------

days2date

函数说明:

计算两个日期型数据相差几天

语法:

`days2date(dateExp1, dateExp2)`

参数说明:

<code>dateExp1</code>	起始日期表达式，其结果必须为日期或中国日期时间格式的字符串
<code>dateExp2</code>	结束日期表达式，其结果必须为日期或中国日期时间格式的字符串

daysmonth**函数说明：**

获得指定日期所在月的天数

语法：

[daysmonth\(dateExp\)](#)

参数说明：

dateExp 表达式，其结果必须为日期或中国日期时间格式的字符串

daysyear**函数说明：**

获得指定日期所在年的天数

语法：

[daysyear\(dateExp\)](#)

[daysyear\(year\)](#)

参数说明：

dateExp 表达式，其结果必须为日期或中国日期时间格式的字符串

year 返回整数的表达式

formatdatetime**函数说明：**

将字符串转换成日期时间，转换时对其进行格式化

语法：

[formatdatetime\(string, format\)](#)

参数说明：

string 合法的日期格式的字符串

format 用于格式化的格式串

hour**函数说明：**

从日期时间型数据中，获得当前时间位于一天中的第几个时辰

语法：

[hour\(datetimeExp\)](#)

参数说明：

datetimeExp 表达式，其结果必须为日期或中国日期时间格式的字符串

minute**函数说明：**

从日期时间型数据中，获得分钟的信息

语法：

[minute\(datetimeExp\)](#)

参数说明：

datetimeExp 表达式，其结果必须为日期或中国日期时间格式的字符串

month**函数说明：**

取得指定日期所在的月份

语法：

[month\(dateExp\)](#)

参数说明：

dateExp 表达式，其结果必须为日期或中国日期时间格式的字符串

monthbegin**函数说明:**

取得指定日期所在月的月首

语法:

`monthbegin(dateExp)`

参数说明:

dateExp 表达式，其结果必须为日期或中国日期时间格式的字符串

monthend**函数说明:**

取得指定日期所在月的月末

语法:

`monthend(dateExp)`

参数说明:

dateExp 表达式，其结果必须为日期或中国日期时间格式的字符串

now**函数说明:**

获得系统此刻的时间

语法:

`now()`

prevday**函数说明:**

取得指定日期的上一天。

语法:

`prevday(dateExp)`

参数说明:

dateExp 表达式，为日期或中国日期时间格式的字符串

prevmonth**函数说明:**

取得指定日期在上月的同日，若无同一日，则返回上月月末

语法:

`prevmonth(dateExp)`

参数说明:

dateExp 表达式，为日期或中国日期时间格式的字符串

prevyear**函数说明:**

取得指定日期在去年的同月同日，若无同月同日，则返回同月最后一天

语法:

`prevyear(dateExp)`

参数说明:

dateExp 表达式，为日期或中国日期时间格式的字符串

quarterbegin**函数说明:**

取得指定日期所在季度的首日

语法:

`quarterbegin(dateExp)`

参数说明:

dateExp 表达式，为日期或中国日期时间格式的字符串

quarterend

函数说明：

取得指定日期所在季度的末日

语法：

[quarterend\(dateExp \)](#)

参数说明：

dateExp 表达式，其结果必须为日期或中国日期时间格式的字符串

second

函数说明：

从日期时间型数据中，获得秒信息

语法：

[second\(datetimeExp\)](#)

参数说明：

datetimeExp 表达式，其结果必须为日期或中国日期时间格式的字符串

second2date

函数说明：

计算两个日期时间型数据相差几秒

语法：

[seconds2date\(datetimeExp1,datetimeExp2\)](#)

参数说明：

datetimeExp1 起始时间表达式，其结果必须为日期或中国日期时间格式的字符串

datetimeExp2 结束时间表达式，其结果必须为日期或中国日期时间格式的字符串

time

函数说明：

将字符串转换成时间型数据

语法：

[time\(stringExp\)](#)

参数说明：

stringExp 字符串表达式，其结果必须返回合法时间格式的字符串

week

函数说明：

从日期型数据中获得该日的星期名称

语法：

[week\(dateExp \)](#)

参数说明：

dateExp 表达式，其结果必须为日期或中国日期时间格式的字符串

weekbegin

函数说明：

获得指定日期所在星期的星期天

语法：

[weekbegin\(dateExp\)](#)

参数说明：

dateExp 表达式，其结果必须为日期或中国日期时间格式的字符串

weekend**函数说明：**

获得指定日期所在星期的星期六

语法：

`weekend(dateExp)`

参数说明：

dateExp 表达式，其结果必须为日期或中国日期时间格式的字符串

weeknum**函数说明：**

从日期型数据中，获得该日位于一个星期中的第几天

语法：

`weeknum(dateExp)`

参数说明：

dateExp 表达式，其结果必须为日期或中国日期时间格式的字符串

year**函数说明：**

从日期型数据中获得年信息

语法：

`year(dateExp)`

参数说明：

dateExp 表达式，其结果必须为日期或中国日期时间格式的字符串

(七)转换函数

asc**函数说明：**

取字符串指定位置的字符的unicode值，如果是ascii字符则返回ascii码

语法：

`asc(string[, nPos])`

参数说明：

string 给定的字符串

nPos 整数表达式

bigmoney**函数说明：**

将浮点数（只支持小数点后两位）转换人民币大写格式

语法：

`bigmoney(numberExp)`

参数说明：

numberExp 数据值表达式

char**函数说明：**

根据给定的unicode编码取得其对应的字符

语法：

`char(int)`

参数说明:

int 整数表达式, 返回unicode编码

integer**函数说明:**

将字符串或数字转换成整数

语法:

`integer(string)`
`integer(number)`

参数说明:

string 需要转换的字符串表达式
number 数字

函数示例:

`integer("100")` 返回100
`integer(100.1)` 返回100

isdate**函数说明:**

判定字符串是否具有转换成日期的合法格式

语法:

`isdate(string)`

参数说明:

string 字符串表达式

isnumber**函数说明:**

判定字符串是否具有转换成数值的合法格式

语法:

`isnumber(string)`

参数说明:

string 字符串表达式

istime**函数说明:**

判定字符串是否具有转换成时间的合法格式

语法:

`istime(string)`

参数说明:

string 字符串表达式

tochinese**函数说明:**

将一个整数转化成汉字大写

语法:

`tochinese(intExp[, abbreviateExp], uppercaseExp)`

参数说明:

intExp 整数表达式
abbreviateExp 是否采用亿万千百十的写法
uppercaseExp 为真时采用零一二三四五六七八九十百千, 否则采用壹贰叁肆伍陆柒捌玖拾佰仟

todouble**函数说明：**

将字符串或数字转换成双精度浮点数

语法：

`todouble(string)`

`todouble(number)`

参数说明：

string 需要转换的字符串表达式

number 数字

tolong**函数说明：**

将字符串或数字转换成长整数

语法：

`tolong(string)`

`tolong(number)`

参数说明：

string 需要转换的字符串表达式

number 数字

tonumber**函数说明：**

将字符串转换成相应的数值

语法：

`tonumber(string)`

参数说明：

string 需要转换的字符串表达式

tostring**函数说明：**

将对象转换成字符型，转换过程中可以进行格式化

语法：

`tostring(expression[, format])`

参数说明：

expression 需要转换成字符串的常数对象或表达式

format 转换过程中进行格式化的格式串

(八)操作符

符号	说明	左操作数	右操作数	运算结果	示例说明
+	加	数值	数值	数值	A1+A2 或 1+2
		字符串	字符串	字符串	A1+A2 或 "a"+"b"
-	减	数值	数值	数值	A1-A2
		无	数值	数值	-A1
*	乘	数值	数值	数值	A1*A2
/	除	数值	数值	数值	A1/A2
&&	与	布尔值	布尔值	布尔值	A1&&A2

或 and					
&	取位置	无	主单元 格		&A1
或 or	或	布尔值	布尔值	布尔值	A1 A2
! 或 not	非	无	布尔值	布尔值	!A1
=或 ==	等于	数值	数值	布尔值	A1==A2
		字符串	字符串		A1=A2
		日期	日期		
		布尔值	布尔值		
!=	不等于	数值	数值	布尔值	A1!=A2
		字符串	字符串		
		日期	日期		
		布尔值	布尔值		
:	到	单元格	单元格	List集 合	A1:D3 由A1和D3为两个对角点圈起的矩形 中的单元格集合，可以使用在sum，max等 函数中作参数
in	包含	数值	List集 合	布尔值	A1 in (1,2,3) 是否在集合中 左右操作数均可为单值或数组，如 [1,2,3] in [1,2,3,4] ds1.get(id) in [1,2,3,5] 1 in ds1.get(id) 1 in ds1.group(id) //此处应写成1 in ds1. get(id)可提高效率 [1,2,3] in 1 1 in 1
		字符串			
		日期			
		布尔值			
		数值	布尔值	布尔值	A1 in A2 相当于A1=A2
		字符串			
		日期			
		布尔值			
>	大于	数值	布尔值	布尔值	A1>A2
		字符串			
		日期			
		布尔值			
>=	大于等 于	数值	布尔值	布尔值	A1>=A2
		字符串			
		日期			
		布尔值			
<	小于	数值	布尔值	布尔值	A1<A2
		字符串			
		日期			
		布尔值			
<=	小于等 于	数值	布尔值	布尔值	A1<=A2
		字符串			
		日期			
		布尔值			
to	到	整数	整数	List集 合	1 to 5
%	求余	数值	数值	数值	11%3
.	对象成				ds1.get(#2)

	员				
like	象				a like “b*”
\$	绝对定位				扩展坐标的条件表达式中，表示指定单元格的顶格D5[A5:1]{b5=\$b5-3}
@	参数或变量标识				@arg1
[]	数组，取单元格，取数组元素				[1,2,3,4]表示一个元素为1,2,3,4的数组 [1,2,3,4][2]表示取元素为1,2,3,4的数组中的第2个元素，即2 D5[A5:1] 扩展坐标
{}	选择条件表达式				D5[A5:1]{b5=3}扩展坐标中的条件表达式 D5[A5:1]{}表示该扩展坐标中的所有指定单元格，代表一个集合
#	列号				ds1.#3表示数据集ds1中的第三列，ds1.#0表示1,2,3的记录顺序号
\${}	宏				\${macro1}

(九)运算优先级

优先级	计算顺序	运算符	说明
14	从左到右	., [], ()	字段访问、数组索引、函数调用和表达式分组
13	从右到左	-, !	一元运算符
12	从左到右	*, /, %	相乘、相除、求余数
11	从左到右	+, -	相加、字符串串联、相减
10	从左到右	<, <=, >, >=	小于、小于或等于、大于、大于或等于
9	从左到右	==, !=	相等、不相等
8	从左到右	&	按位“与”
7	从左到右	^	按位“异或”
6	从左到右		按位“或”
5	从左到右	&&	逻辑“与”
4	从左到右		逻辑“或”

3	从右到左	?:	条件
2	从右到左	=	赋值
1	从左到右	, (逗号)	多个计算

(十)关键字

true

布尔型，代表是

false

布尔型，代表否

null

代表空值

@value

代表当前单元格的真实值

@@result

用在数据集定义的参数表达式中，当数据集来自存储过程，且该存储过程通过输出参数返回结果集时，该输出参数在sql中用问号表示，和该问号对应的参数表达式写成@@result

存储过程可以在call函数中使用，例如：

call("{call procName (?,?,?)}", "@@result", "a01", "A1")

此时call函数写在单元格中，可以引用单元格的值作为输入参数

{page_cn}

在标题区或者页眉页脚区中使用，代表当前页号，以中文格式出现，是个宏替换的概念

{page}

在标题区或者页眉页脚区中使用，代表当前页号，是个宏替换的概念

{pages}

在标题区或者页眉页脚区中使用，代表总页数，是个宏替换的概念

{pages_cn}

在标题区或者页眉页脚区中使用，代表总页数，以中文格式出现，是个宏替换的概念

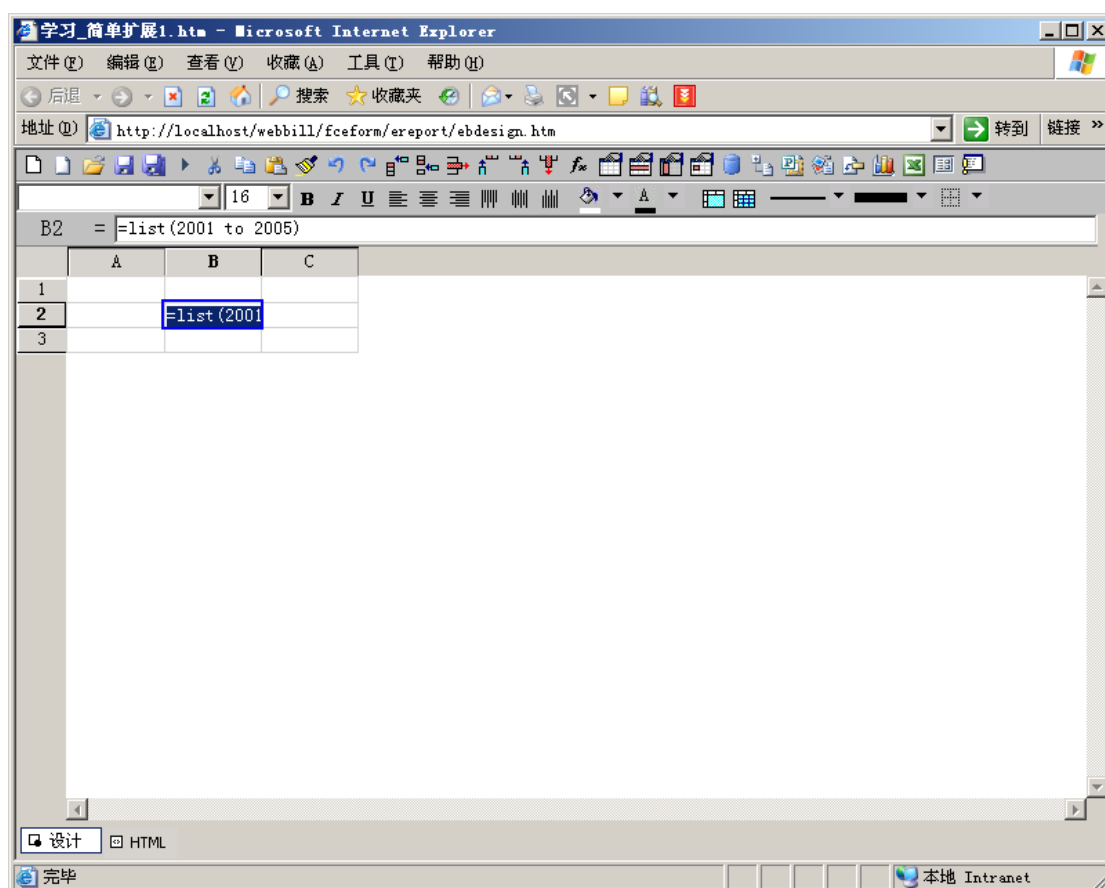
六 学习扩展模型

e表没有采用常规的由一个数据集的循环为主来得到报表的思路，而是采用在一个类excel的表格基础上，通过单元格的扩展而得到整个报表的思路。

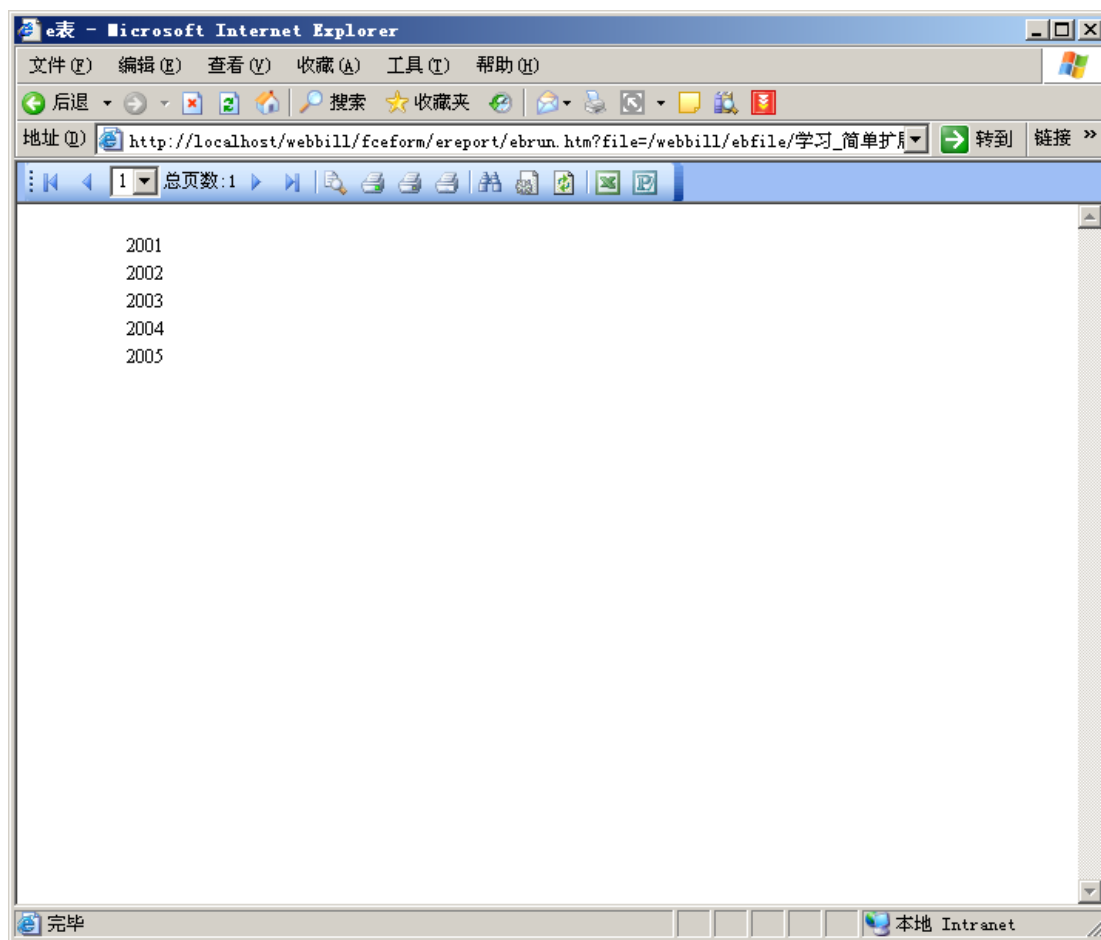
(一)了解扩展

第一张扩展报表

进入e表设计器,在B2单元格中输入 `=list(2001 to 2005)`，然后删除多余行列，点[保存报表]。在弹出的窗口中输入 `学习_简单扩展1` 来作为报表名称。如下图：



然后点[运行报表]，运行结果如下图：



可以看到单元格B2 的公式 `list(2001 to 2005)` 已运算并扩展成5个单元格，这5个单元格的值为从2001 到 2005 。而且是从上往下排列。这是因为默认情况下是往下扩展的。

下面进一步在B2单元格两边的单元格写上文字和加上颜色等属性。如下图：

	A	B	C
1			
2	同级格拉大	=list	从单元格复制
3			

运行结果如下图：

同级格拉大	2001	从单元格复制
	2002	从单元格复制
	2003	从单元格复制
	2004	从单元格复制
	2005	从单元格复制

从中可以看到，纵向扩展时左边的单元格是拉大，右边的单元格是复制。具体为：
b2为纵向扩展格，因此c2缺省附属于b2，因此当b2纵向扩展时，c2被复制；a2由于不是扩展格，因此a2虽然在b2的左边，但a2不是b2的左顶格，a2和b2的左顶格都是00格，因此a2和b2属于同一级别的单元格，所以b2扩展的时候，也会把同级别单元格a2拉大。
这个示例是符合如下规则：

- 纵向扩展时，左边纵向扩展单元格缺省为它的左顶格，右边单元格缺省为它的子格

- 如果左边没有纵向扩展格，则左顶格缺省为00格
- 单元格进行扩展时，顶格或同级别单元格被拉大，子格被复制；

横向扩展

将上面的例子调整一下，将扩展单元格的[扩展方向]属性设置为横向扩展，如下图：

	A	B	C
1			
2		同级格拉大	
3		=list(2001 to	
4		从单元格复制	

运行报表界面如下图：

同级格拉大				
2001	2002	2003	2004	2005
从单元格复制	从单元格复制	从单元格复制	从单元格复制	从单元格复制

从中可以看到，横向扩展时上边的单元格是拉大，下边的单元格是复制。具体为：
b3为横向扩展格，因此b4缺省附属于b3，因此当b3横向扩展时，b4被复制；b2由于不是扩展格，因此b2虽然在b3的上边，但b2不是b3的上顶格，b2和b3的上顶格都是00格，因此b2和b3属于同一级别的单元格，所以b3扩展的时候，也会把同级别单元格b2拉大。
这个示例是符合如下规则：

- 横向扩展时，上边横向扩展单元格缺省为它的上顶格，下边单元格缺省为它的子格
- 如果上边没有横向扩展格，则上顶格缺省为00格
- 单元格进行扩展时，顶单元格或同级别单元格被拉大，子格被复制；

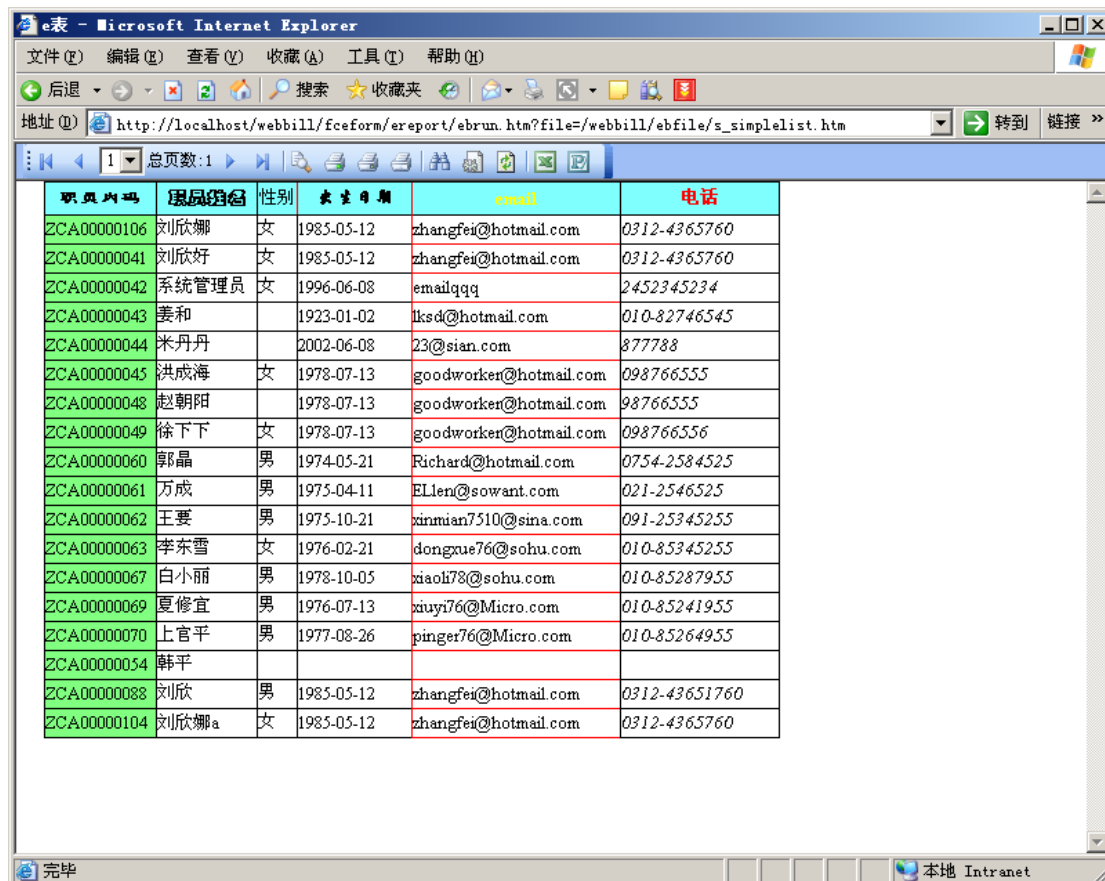
(二)数据集相关的扩展

简单数据列表

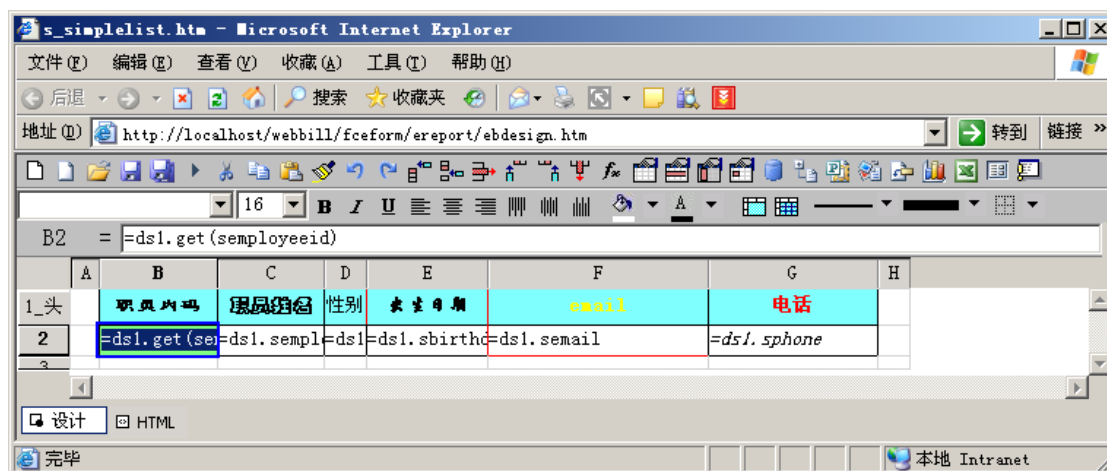
假如数据集中的数据如下图所示。为了方便演示，所以采用内建数据集的方式。直接用sql语句来产生数据集也是一样的。



要得到如下图所示的报表:



样式部分就不在这里说明，在此主要说明如何定义公式。将数据展现出来。下面是在报表设计器中的图示。



主要是B2单元格的公式为：`=ds1.get(semployeid)`

ds1是数据集的名称，semployeid是ds1的字段名，一般为主键字段或不含重复值的字段。

Get是数据集的函数，公式的意义为取ds1数据集中semployeid字段的内容列表。这个公式的计算结果是一个集合（指不是一个单值）。在默认的情况下计算结果是集合的单元格是纵向扩展的。所以会纵向复制行。同时会在此保存当前行数据的一个指针，以便能直接取到本行的其它字段的值。

C2单元格的公式为：`=ds1.semployeename`

表示从C2的左顶格中保存的当前行数据指针中取semployeename字段的值。

下面演示一个扩展过程，为了说明方便，去掉单元格的样式部分的同步：

A 准备开始计算时如下图：

职员内码	职员姓名	性别	出生日期	email	电话
=ds1.get(se	=ds1.sempl	=ds1	=ds1.sbirthc	=ds1.semail	=ds1.sphone

B 计算了 `ds1.get(semployeid)` 公式后。

职员内码	职员姓名	性别	出生日期	email	电话
ZCA00000106	=ds1.sempl	=ds1	=ds1.sbirthc	=ds1.semail	=ds1.sphone
ZCA00000041	=ds1.sempl	=ds1	=ds1.sbirthc	=ds1.semail	=ds1.sphone
ZCA00000042	=ds1.sempl	=ds1	=ds1.sbirthc	=ds1.semail	=ds1.sphone
...					

即将semployeid的值算出来了，同时复制了多行。

C 计算 `ds1.semployeename` 公式后。

职员内码	职员姓名	性别	出生日期	email	电话
ZCA00000106	刘欣娜	=ds1	=ds1.sbirthc	=ds1.semail	=ds1.sphone
ZCA00000041	刘欣好	=ds1	=ds1.sbirthc	=ds1.semail	=ds1.sphone
ZCA00000042	系统管理员	=ds1	=ds1.sbirthc	=ds1.semail	=ds1.sphone
...	...				

D 以此类推，直到整个表都计算出来。

职员内码	职员姓名	性别	出生日期	email	电话
ZCA00000106	刘欣娜	女	1985-05-12	zhangfei@hotmail.com	0312-4365760
ZCA00000041	刘欣好	女	1985-05-12	zhangfei@hotmail.com	0312-4365760
ZCA00000042	系统管理员	女	1996-06-08	emailqqq	2452345234
ZCA00000043	姜和		1923-01-02	lksd@hotmail.com	010-82746545
ZCA00000044	米丹丹		2002-06-08	23@sian.com	877788
ZCA00000045	洪成海	女	1978-07-13	goodworker@hotmail.com	098766555
ZCA00000048	赵朝阳		1978-07-13	goodworker@hotmail.com	98766555
ZCA00000049	徐下下	女	1978-07-13	goodworker@hotmail.com	098766556
ZCA00000060	郭晶	男	1974-05-21	Richard@hotmail.com	0754-2584525
ZCA00000061	万成	男	1975-04-11	ELlen@sowant.com	021-2546525
ZCA00000062	王要	男	1975-10-21	xinmian7510@sina.com	091-25345255
ZCA00000063	李东雪	女	1976-02-21	dongxue76@sohu.com	010-85345255
ZCA00000067	白小丽	男	1978-10-05	xiaoli78@sohu.com	010-85287955
ZCA00000069	夏修宜	男	1976-07-13	xiuyi76@Micro.com	010-85241955
ZCA00000070	上官平	男	1977-08-26	pinger76@Micro.com	010-85264955
ZCA00000054	韩平				
ZCA00000088	刘欣	男	1985-05-12	zhangfei@hotmail.com	0312-43651760
ZCA00000104	刘欣娜a	女	1985-05-12	zhangfei@hotmail.com	0312-4365760

分组报表

假如数据如下图所示：

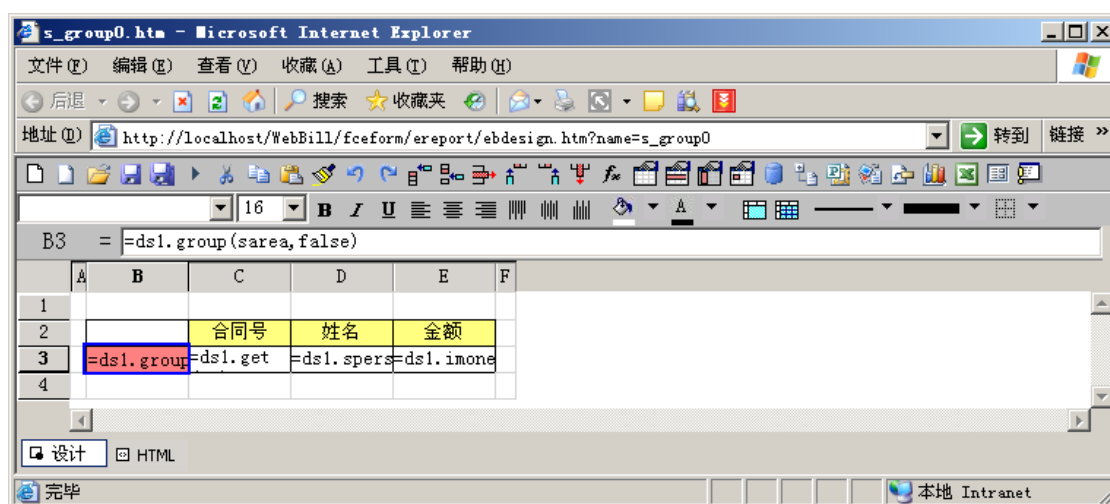
sarea	id	sperson	imoney
北京	1	贾宝玉	2000
北京	2	林黛玉	1800
北京	3	薛宝钗	1700

要计算出下图这样的报表：

	合同号	姓名	金额
上海	12	贾琏	4900
	5	贾宝玉	2300
	9	贾政	1650
	7	贾琏	1290
	17	贾宝玉	5800
	16	贾琏	800
	4	贾琏	2500
	18	林黛玉	900
北京	11	林黛玉	3300
	10	贾宝玉	3600
	3	薛宝钗	1700
	2	林黛玉	1800
	1	贾宝玉	2000
	20	贾琏	3300
天津			
山东	6	贾宝玉	1200
	8	贾政	4200
	15	贾政	1800
江西	14	林黛玉	3800
	19	贾宝玉	1100
	13	贾政	3900
河北	22	贾宝玉	990
	21	贾宝玉	3200

即要求按sarea字段分组显示。

在报表设计器中制作的界面如下图：



相比简单数据列表的示例，此表主要是多了B3单元格的公式为：ds1.group(sarea,false)

它的意思是ds1按sarea字段分组，将组的集合（**注意：不是单值**）按纵向扩展，即纵向复制多行。同时在展开的每一个B3单元格中都保存了当前组的指针。这些B3单元格的内容为当前组的首行的sarea字段的内容。

C3单元格的公式虽然和简单数据列表的示例中的相同，但因为它的左顶格也是一个扩展单元格（B3），而不是00格。所以它是指在当前组内纵向扩展。

下面演示一个扩展过程，为了说明方便，去掉单元格的样式部分的同步：

A 准备开始计算时，如下图：

	合同号	姓名	金额
=ds1.group	=ds1.get	=ds1.spers	=ds1.imone

B 计算了 ds1.group(sarea,false) 公式后。

	合同号	姓名	金额
上海	=ds1.get	=ds1.spers	=ds1.imone
北京	=ds1.get	=ds1.spers	=ds1.imone
天津	=ds1.get	=ds1.spers	=ds1.imone
...			

如图所示将B3单元格的值计算出来了，同时在B3单元格中会保留一个当前组的指针，通过它能得到整个组的数据。如第一个是上海组，即ds1数据集中sarea=上海的数据。（**注意：一个组的数据是指一个二维表的数据，不光是一行的数据**）同时也复制了多行。

C 计算一个 ds1.get(id) 公式后

	合同号	姓名	金额
上海	12	=ds1.spers	=ds1.imone
	5	=ds1.spers	=ds1.imone
	...	=ds1.spers	=ds1.imone
北京	=ds1.get	=ds1.spers	=ds1.imone
天津	=ds1.get	=ds1.spers	=ds1.imone
...			

如图所示，第一个 ds1.get(id) 就计算出来了，它的计算过程和简单数据列表的示例中的 get函数的计算过程一样，只是指取sarea=上海 这一组的合同号来扩展，纵向扩展时左边的单元格拉大。

D 以此类推，就得到了如下的报表。

	合同号	姓名	金额
上海	12	贾璉	4900
	5	贾宝玉	2300
	9	贾政	1650
	7	贾璉	1290
	17	贾宝玉	5800
	16	贾璉	800
	4	贾璉	2500
北京	18	林黛玉	900
	11	林黛玉	3300
	10	贾宝玉	3600
	3	薛宝钗	1700
	2	林黛玉	1800
	1	贾宝玉	2000
天津	20	贾璉	3300
山东	6	贾宝玉	1200
	8	贾政	4200
	15	贾政	1800
江西	14	林黛玉	3800
	19	贾宝玉	1100
	13	贾政	3900
河北	22	贾宝玉	990
	21	贾宝玉	3200

交叉报表

分组报表是纵向扩展，如设置单元格的[扩展方向]属性为横向扩展，这样就可以得到交叉报表，因为交叉报表的运算原理和分组报表一样，在此就不详细说明了。

多层扩展

随着报表越来越复杂，常常需要多层扩展，多层扩展时，扩展次序是从主到次的，即先扩展顶格，然后扩展其子格，再扩展其二级子格，依此类推。

单元格进行横向扩展时，会将其同列的上顶格拉大，把其子格复制，特别地，如果其某个上顶格不在同列上，则该上顶格不会被拉大；单元格进行纵向扩展时，会将其同行的左顶格拉大，把其附属单元格复制，特别地，如果其某个左顶格不在同行上，则该左顶格不会被拉大。

一般地，顶格能够主动进行扩展复制，子格被顶格带动着复制。由于子格同时又可能是别的单元格的顶格，本身还可以进行主动扩展复制，因此主动扩展和被动复制是相对的。既不能主动扩展复制，也不能被动复制的单元格，我们称为不可复制格，或者叫固定格。

同一报表中可能同时有纵向扩展格和横向扩展格，如果它们的子格有重叠部分，则这些子格就既有左顶格又有上顶格，在扩展时会既向下又向右复制，形成一片矩形单元格区域，从而做到交叉扩展。相应地，在这种机制下，多层交叉也不难实现。

在交叉扩展中，有的单元格有可能既被横向扩展向右复制，也被纵向扩展向下复制，可是，单元格的横向扩展与纵向扩展这两种扩展是相互独立的，既可以先进行横向扩展，也可以先进行纵向扩展，并不会影响扩展之后的结果。

下面以一个示例来说明：
假定数据如下图所示：

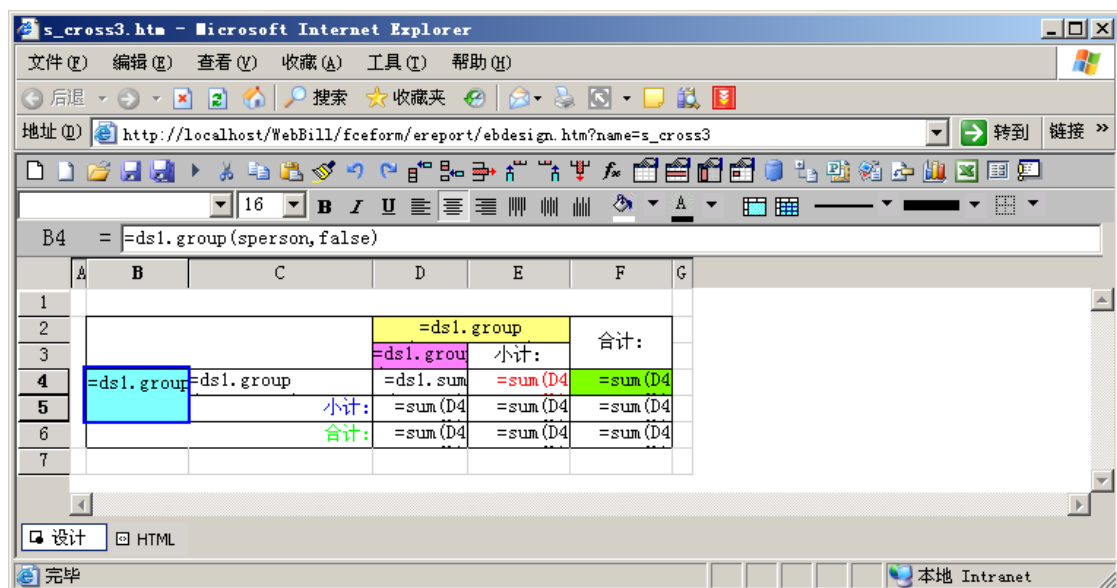
sarea	sitem	sperson	sdw	imoney
北京	电视	贾宝玉	联想公司	2000
北京	电视	林黛玉	方正集团	1800
北京	电视	薛宝钗	北京四通公司	1700
上海	电视	贾琏	北京方成公司	2500

由上述数据进行多级分组交叉要得到如下报表：

		上海				北京		
		打印机	电脑	电视	小计:	打印机	电脑	电视
林黛玉	HP公司	0	0	0	0	900	0	0
	方正集团	0	0	0	0	0	3300	1800
	小计:	0	0	0	0	900	3300	1800
薛宝钗	北京四通公司	0	0	0	0	0	0	1700
	小计:	0	0	0	0	0	0	1700
贾宝玉	HP公司	0	5800	0	5800	0	0	0
	北京方成公司	0	0	0	0	0	3600	0
	方正集团	0	0	0	0	0	0	0
	联想公司	0	0	2300	2300	0	0	2000
	小计:	0	5800	2300	8100	0	3600	2000
贾政	创维集团	0	0	0	0	0	0	0
	北京方成公司	0	0	1650	1650	0	0	0
	恩创公司	0	0	0	0	0	0	0
	小计:	0	0	1650	1650	0	0	0
贾琏	HP公司	800	0	0	800	0	0	0
	北京方成公司	0	0	2500	2500	0	0	0
	方正集团	1290	4900	0	6190	0	0	0
	小计:	2090	4900	2500	9490	0	0	0
合计:		2090	10700	6450	19240	900	6900	5500

即要按 sperson和sdw 两个字段来分组，按 sarea,sitem两个字段来交叉。下面说一下主要制作过程。

在报表设计器中的界面图如下：



主要单元格属性:

B4公式 = ds1.group(spersion, false) B4扩展方向 = 纵向扩展
 C4公式 = ds1.group(sdw, false) C4扩展方向 = 纵向扩展
 D2公式 = ds1.group(sarea, false) D2扩展方向 = 横向扩展
 D3公式 = ds1.group(sitem, false) D3扩展方向 = 横向扩展
 D4公式 = ds1.sum(imoney)
 D5, D6, E4, E5, E6, F4, F5, F6的公式都 = sum(D4) //因为D5, D6, E4, E5, E6, F4, F5, F6的左顶格或上顶格不一样, 所以汇总的值会不同

由上述设置可以看出, 在e表中设置多层分组交叉的报表只是多用几次数据集的group函数而已。

在本例中还有 D4 = ds1.sum(imoney), 这是用到了数据集的统计函数。表示汇总ds1数据集在当前组的 imoney字段的值。实际上此例中的D4单元格在运算后就扩展成了多个区域的单元格了。如下图中用红框框住的区域就是它扩展出来的。

		上海				北京				天津	
		打印机	电脑	电视	小计:	打印机	电脑	电视	小计:	电脑	小计:
林黛玉	HP公司	0	0	0	0	900	0	0	900	0	0
	方正集团	0	0	0	0	0	3300	1800	5100	0	0
	小计:	0	0	0	0	900	3300	1800	6000	0	0
薛宝钗	北京四通公司	0	0	0	0	0	0	1700	1700	0	0
	小计:	0	0	0	0	0	0	1700	1700	0	0
贾宝玉	HP公司	0	5800	0	5800	0	0	0	0	0	0
	北京方成公司	0	0	0	0	0	3600	0	3600	0	0
	方正集团	0	0	0	0	0	0	0	0	0	0
	联想公司	0	0	2300	2300	0	0	2000	2000	0	0
	小计:	0	5800	2300	8100	0	3600	2000	5600	0	0
贾政	创维集团	0	0	0	0	0	0	0	0	0	0
	北京方成公司	0	0	1650	1650	0	0	0	0	0	0
	思创公司	0	0	0	0	0	0	0	0	0	0
	小计:	0	0	1650	1650	0	0	0	0	0	0
贾琏	HP公司	800	0	0	800	0	0	0	0	0	0
	北京方成公司	0	0	2500	2500	0	0	0	0	0	0
	方正集团	1290	4900	0	6190	0	0	0	0	3300	3300
	小计:	2090	4900	2500	9490	0	0	0	0	3300	3300
合计:		2090	10700	6450	19240	900	6900	5500	13300	3300	3300

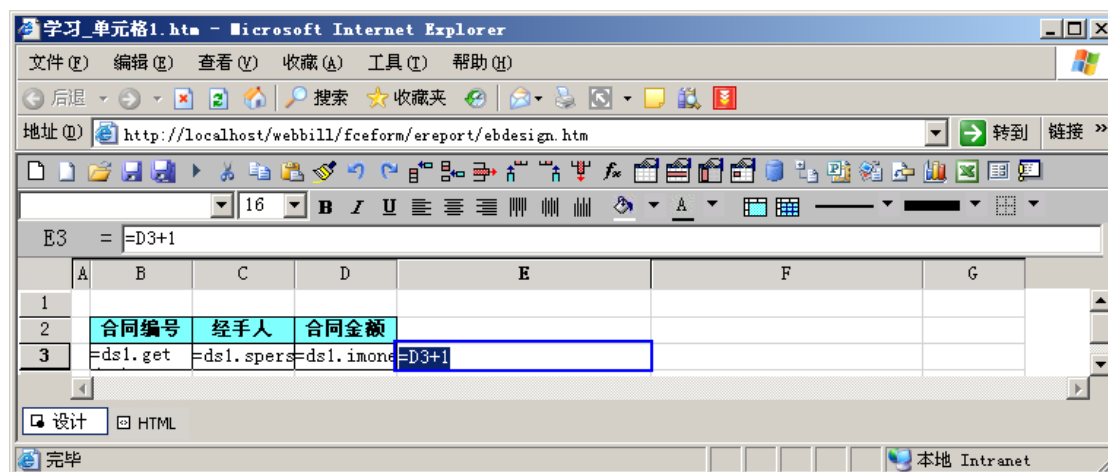
其扩展过程同分组报表中描述的一样, 只是经过多次group扩展。所以D4单元格就变成了这么多。

(三)扩展后的单元格

通过上面的多层扩展示例中可以看出，一个D4单元格扩展后竟变成了如此多的单元格。在实际中常需要对扩展后的单元格进行引用，定位，条件过滤，运算等等操作，本节就说明一下如何做这样工作。下面是一些单元格公式的常用模式：

E3=D3+1 模式

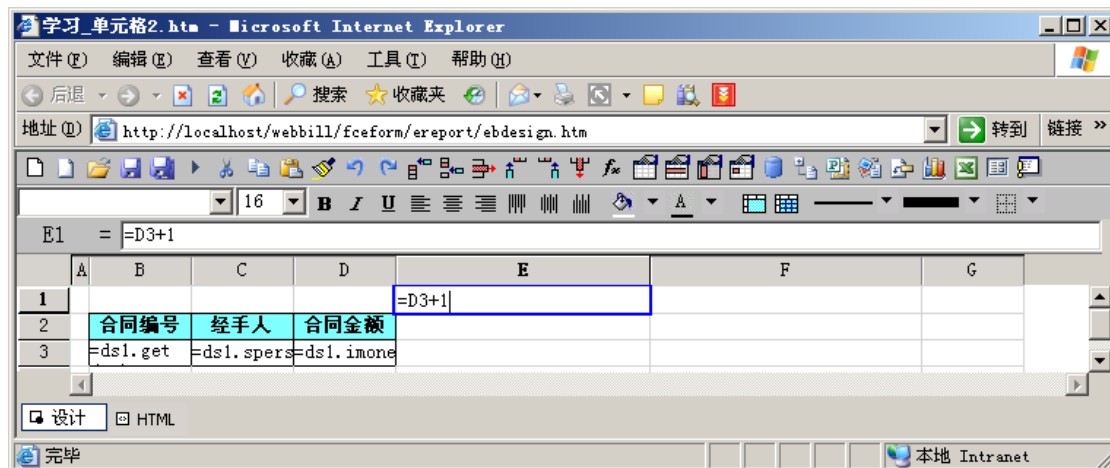
参看报表文件：学习_单元格1.htm，在文件在报表设计器中的界面为：



运行结果界面为：

合同编号	经手人	合同金额
1	贾宝玉	2000
2	林黛玉	1800
3	薛宝钗	1700
4	贾璉	2500
5	贾宝玉	2300
6	贾宝玉	1200
7	贾璉	1290
8	贾政	4200
9	贾政	1650
10	贾宝玉	3600
11	林黛玉	3300
12	贾璉	4900
13	贾政	3900
14	林黛玉	3800
15	贾政	1800
16	贾璉	800
17	贾宝玉	5800
18	林黛玉	900
19	贾宝玉	1100
20	贾璉	3300
21	贾宝玉	3200
22	贾宝玉	990

此报表前几列为数据集的扩展部分，因在前面已经说明了。在此就不说明了。
 在此主要说明E3单元格，它的公式为： $=D3+1$ ，直观的意思是： $E3=D3+1$ ，但因为D3和E3都会被动扩展出许多单元格。所以就得到如上的运行结果。值得注意的是因为D3和E3有共同的左顶格（B3）。如果将E3单元格的公式移到E1（因为E1不会扩展，为固定单元格），如报表文件：学习_单元格2.htm，设计界面如下图：



运行结果界面如下图：

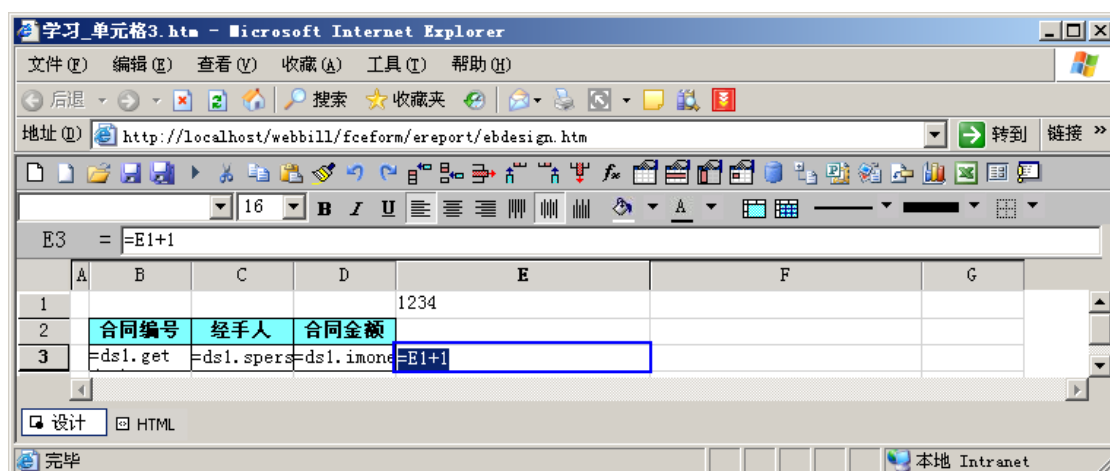
2001

合同编号	经手人	合同金额
1	贾宝玉	2000
2	林黛玉	1800
3	薛宝钗	1700
4	贾璉	2500
5	贾宝玉	2300
6	贾宝玉	1200
7	贾璉	1290
8	贾政	4200
9	贾政	1650
10	贾宝玉	3600
11	林黛玉	3300
12	贾璉	4900
13	贾政	3900
14	林黛玉	3800
15	贾政	1800
16	贾璉	800
17	贾宝玉	5800
18	林黛玉	900
19	贾宝玉	1100
20	贾璉	3300
21	贾宝玉	3200
22	贾宝玉	990

E1单元格的值为2001，为第一个D3单元格的值（2000）+1的结果。因为E1为固定格所以只有一个单元格。而D3会扩展出很多单元格，所以取第一个D3的值。

还有一种情况是当D3为固定格，而E3为扩展单元格时会是如何结果呢，我们来看看报表文

件：学习_单元格3.htm，其设计界面如下图：



运行结果界面如下图：

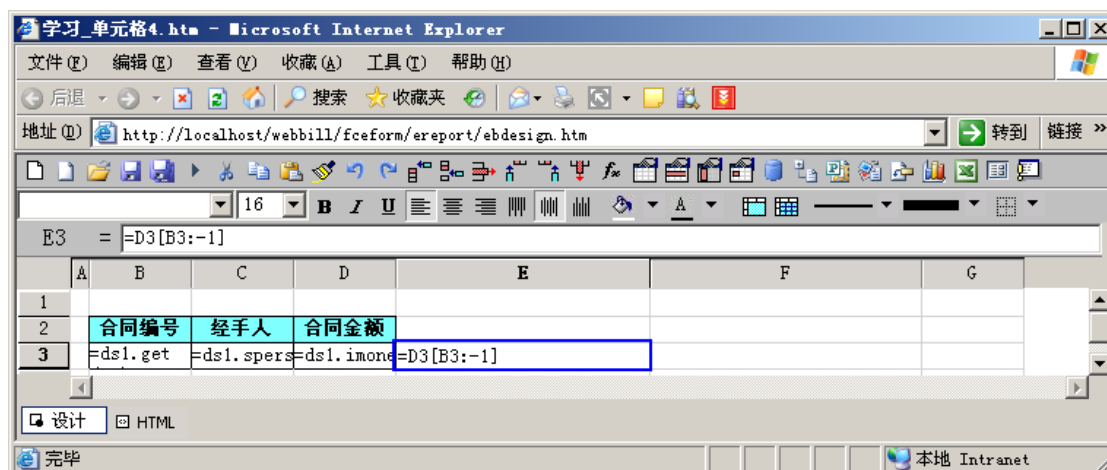
			1234
合同编号	经手人	合同金额	
1	贾宝玉	2000	1235
2	林黛玉	1800	1235
3	薛宝钗	1700	1235
4	贾璉	2500	1235
5	贾宝玉	2300	1235
6	贾宝玉	1200	1235
7	贾璉	1290	1235
8	贾政	4200	1235
9	贾政	1650	1235
10	贾宝玉	3600	1235
11	林黛玉	3300	1235
12	贾璉	4900	1235
13	贾政	3900	1235
14	林黛玉	3800	1235
15	贾政	1800	1235
16	贾璉	800	1235
17	贾宝玉	5800	1235
18	林黛玉	900	1235
19	贾宝玉	1100	1235
20	贾璉	3300	1235
21	贾宝玉	3200	1235
22	贾宝玉	990	1235

公式 $E3 = E1 + 1$ ，其中E1为固定格，E3为扩展单元格。所以E3扩展出多个单元格，其公式都为E1+1，所以值都为1235。

由此可以看出公式的运算结果是和单元格的位置密切相关的。

E3= D3[B3:-1] 模式

先来看文件：学习_单元格4.htm，其设计界面如下图：



运行结果界面如下图：

合同编号	经手人	合同金额
1	贾宝玉	2000
2	林黛玉	1800
3	薛宝钗	1700
4	贾璉	2500
5	贾宝玉	2300
6	贾宝玉	1200
7	贾璉	1290
8	贾政	4200
9	贾政	1650
10	贾宝玉	3600
11	林黛玉	3300
12	贾璉	4900
13	贾政	3900
14	林黛玉	3800
15	贾政	1800
16	贾璉	800
17	贾宝玉	5800
18	林黛玉	900
19	贾宝玉	1100
20	贾璉	3300
21	贾宝玉	3200
22	贾宝玉	990

公式 $E3=D3[B3:-1]$ 的意义是：取上一个D3的值，取随顶格B3扩展出的D3中的上一个。因为D3和E3的左顶格都是B3，所以在此B3可以省略，即写为 $E3=D3[-1]$ 也是一样的。在此例中，E3为当前单元格，D3是指引用单元格。在D3即引用单元格后面加上[-1]来描述和当前单元格E3之间的相对位置关系，这叫单元格的相对坐标。它应用非常广泛，例如报表中常常需要计算同期比、比上期之类的与时间相关的运算，而这些运算往往需要用到下一行的数据减上一行数据，后一列数据减前一列数据，等等，这种涉及到相邻n行或者n列的行间、列间的运算叫单元格的相对定位。

D4=sum(D3{D3>4000}) 模式

上述说的都是取某个单元格的值，而在实际中常常需要对符合一定条件的单元格集合进行统计。特别的，单个单元格也可以视为仅含一个单元格的格集。

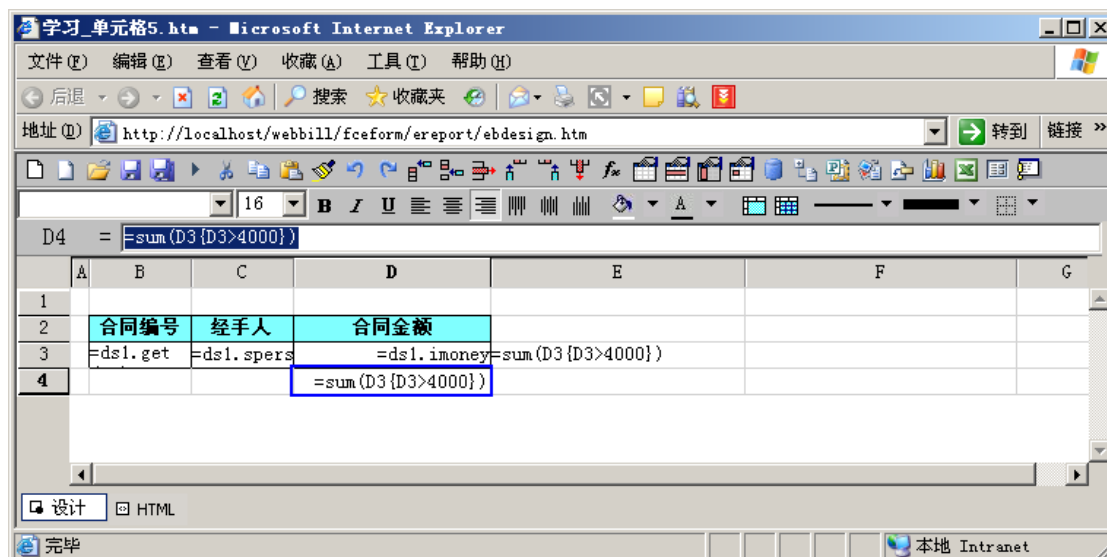
对于固定的单元格，我们可以用数组或者:来表示单元格集合:

举例:

[A1,B3,C4] 表示由A1,B3,C4 三个单元格组成的集合

A1:A10 表示从A1 到A10 的连续区域内的所有单元格组成的集合

对于扩展单元格，我们在单元格的后面加上{ }来表示单元格集合。如D3{}，下面来看看文件：学习_单元格5.htm，设置界面如下图：



运行结果界面如下图：

合同编号	经手人	合同金额
1	贾宝玉	20000
2	林黛玉	18000
3	薛宝钗	17000
4	贾琏	25000
5	贾宝玉	23000
6	贾宝玉	12000
7	贾琏	12900
8	贾政	42004200
9	贾政	16500
10	贾宝玉	36000
11	林黛玉	33000
12	贾琏	49004900
13	贾政	39000
14	林黛玉	38000
15	贾政	18000
16	贾琏	8000
17	贾宝玉	58005800
18	林黛玉	9000
19	贾宝玉	11000
20	贾琏	33000
21	贾宝玉	32000
22	贾宝玉	9900
		14900

咱们先来看 D4 单元格的公式：sum（D3{D3>4000}），因为D4的左顶格为00格，所

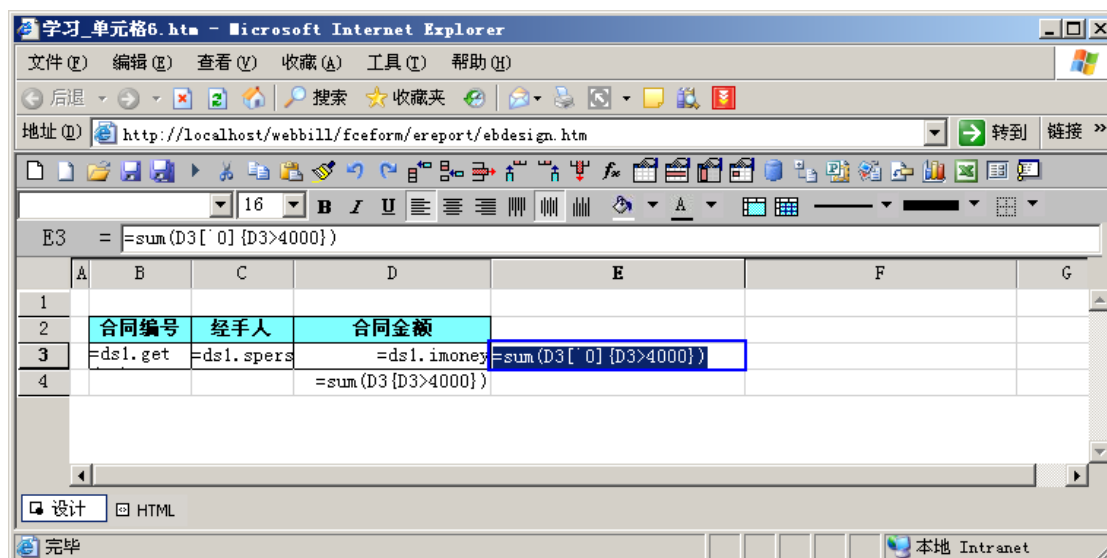
以在其中写扩展格D3是指所有的D3格，写 $D3\{\}$ 就是指所有的D3格的集合。写 $D3\{D3>4000\}$ 就是指大于4000的所有D3格的集合，再加上sum，表示求和。即大于4000的所有D3格的求和。故等于 14900。

下面再来看看E3，它的公式和D4一模一样，但计算结果不一样。主要是因为E3和D3的左顶格都是B3，所以在E3中写D3是指和当前格（E3）在同顶格（B3）下的D3这一个单元格。所以在E3中写 $\text{sum}(D3\{D3>4000\})$ 和 $D3\{D3>4000\}$ 是一样的(当然在此会有0和空的样式区别，有兴趣的可以改改试一试看看效果。)

实际上D3是D3[]的简写，所以 $E3 = \text{sum}(D3[]\{D3>4000\})$ 也是一样的。在单元格后写[]是用来定位单元格的坐标位置。而再加{}则表示符合条件的单元格集合。

`0 模式

在上例中如果希望在E3单元格中也能得到D4单元格一样的结果时，就需要用到`0。下面来看看文件：学习_单元格6.htm,设计界面如下图：



运行结果界面如下图：

合同编号	经手人	合同金额
1	贾宝玉	2000
2	林黛玉	1800
3	薛宝钗	1700
4	贾璉	2500
5	贾宝玉	2300
6	贾宝玉	1200
7	贾璉	1290
8	贾政	4200
9	贾政	1650
10	贾宝玉	3600
11	林黛玉	3300
12	贾璉	4900
13	贾政	3900
14	林黛玉	3800
15	贾政	1800
16	贾璉	800
17	贾宝玉	5800
18	林黛玉	900
19	贾宝玉	1100
20	贾璉	3300
21	贾宝玉	3200
22	贾宝玉	990
		14900

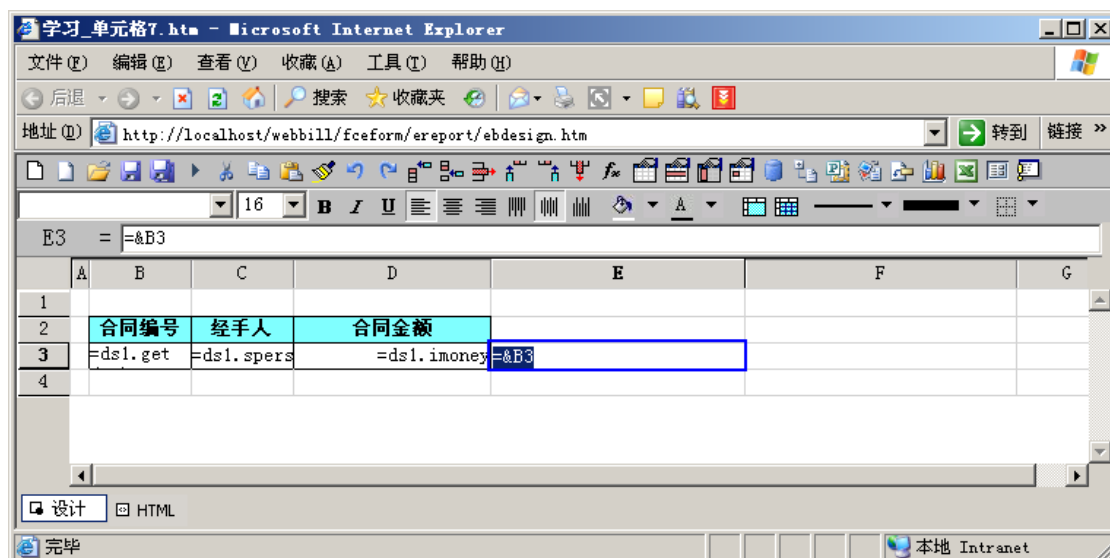
公式D3[*0]是指所有的D3单元格，而和当前单元格的位置无关。所以结果如上。
每个报表都有一个00格，即根格。报表中的扩展单元格是逐级扩展的，有顶格、子格的概念，呈树状的结构，而根格则是这棵树的根。

单元格逐级扩展后实际形成了以00格为根的一棵树，报表中存在多片独立扩展，那么根上就长出多棵树，每一个可主动扩展的格子都是树上的一个节点，最末一级的不可扩展格则是节点上的叶子。而扩展坐标相当于描述任意一个节点或者叶子到达根的路径。由于根格是客观存在的，而树上的所有节点都是由根格发展而来，

& 模式

从前面的介绍可知，单元格是可以扩展的，一个格子可以扩展出多个，那么如何知道某个扩展出来的格子在所有扩展出来的格子中排第几呢？此时我们就引入了&运算符，他可以获得当前格所属的某顶格在所有扩展出来的格子中排第几。

下面来看看文件：学习_单元格7.htm，设计界面如下图：



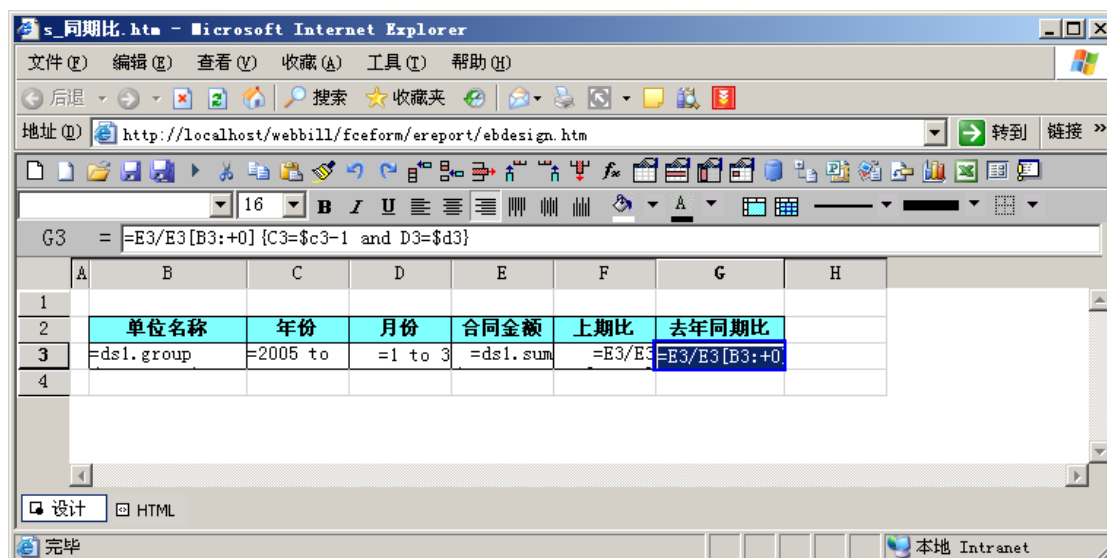
运行结果界面如下图：

合同编号	经手人	合同金额
1	贾宝玉	20001
2	林黛玉	18002
3	薛宝钗	17003
4	贾琏	25004
5	贾宝玉	23005
6	贾宝玉	12006
7	贾琏	12907
8	贾政	42008
9	贾政	16309
10	贾宝玉	360010
11	林黛玉	330011
12	贾琏	490012
13	贾政	390013
14	林黛玉	380014
15	贾政	180015
16	贾琏	80016
17	贾宝玉	580017
18	林黛玉	90018
19	贾宝玉	110019
20	贾琏	330020
21	贾宝玉	320021
22	贾宝玉	99022

要学习&模式也可看看示例中心的多级序号示例。

\$ 模式

下面我们来看看示例中心的文件：s_同期比.htm，它的设计界面图如下：



运行结果界面如下：

单位名称	年份	月份	合同金额	上期比	去年同期比
HP公司	2005	1	0		
		2	0	正无穷大	
		3	0	正无穷大	
	2006	1	0	正无穷大	正无穷大
		2	2690	正无穷大	正无穷大
		3	5800	215.61%	正无穷大
创维集团	2005	1	0		
		2	0	正无穷大	
		3	0	正无穷大	
	2006	1	0	正无穷大	正无穷大
		2	1800	正无穷大	正无穷大
		3	0	0.00%	正无穷大
北京四通公司	2005	1	1700		
		2	0	0.00%	
		3	0	正无穷大	
	2006	1	0	正无穷大	0.00%
		2	0	正无穷大	正无穷大
		3	0	正无穷大	正无穷大
北京方成公司	2005	1	2500		
		2	0	0.00%	
		3	3600	正无穷大	
	2006	1	0	0.00%	0.00%
		2	7400	正无穷大	正无穷大
		3	1650	22.30%	45.83%
恩创公司	2005	1	0		
		2	0	正无穷大	
		3	0	正无穷大	
	2006	1	0	正无穷大	正无穷大
		2	3900	正无穷大	正无穷大
		3	0	0.00%	正无穷大
方正集团	2005	1	1800		

我们重点讲一下 $G3 = E3/E3[B3:+0]\{C3=\$c3-1 \text{ and } D3=\$d3\}$ 这个公式。

设计器中只有一行的表达式单元格，但是扩展后变成了很多的行。而该表格中的难点是

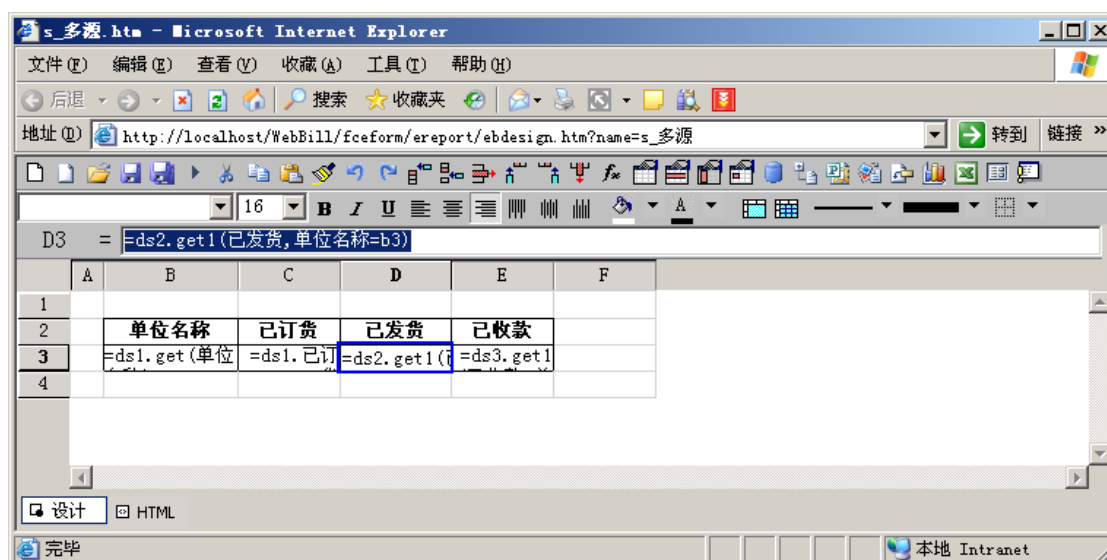
上期比，也就是说，对于2006年3月份的格子来说，需要和2005年3月份的数据进行对比运算，而月份数据可能不是标准的从1 to 3，所以无法靠相对坐标来实现。这时就需要用到\$，如\$C3是指当前单元格的C3格。

$G3 = E3/E3[B3:+0]\{C3=\$c3-1 \text{ and } D3=\$d3\}$ 公式的意思是：E3[B3:+0] 表示B3（即单位名称）相同的E3格集合，\$C3是指当前单元格（G3）所对应的C3格，直接写C3是指引用单元格（即E3[B3:+0]）所对应的C3格，即同单位的所有C3格集合。所以{C3=\$c3-1 and D3=\$d3}的意思是上一年的月份相同。E3[B3:+0]{C3=\$c3-1 and D3=\$d3}的意思是同单位的上一年的月份相同的E3格。所以 $G3 = E3/E3[B3:+0]\{C3=\$c3-1 \text{ and } D3=\$d3\}$ 就得到了去年同期的数据了。

(四)数据集公式中引用单元格

D3=ds2.get1(已发货,单位名称=b3)

打开示例中心的文件：s_多源.htm，在设计界面如下图：



运行结果界面如下图：

单位名称	已订货	已发货	已收款
单位1	1200	900	890
单位2	1345	1111	900
单位3	1586	1009	901
单位4	3122	1211	1000
单位5	1452	1311	1008
单位6	2211	890	700

其中已订货数据来自ds1数据集，已发货数据来自ds2数据集，通过单位名称关联起来。D3 = ds2.get1(已发货, 单位名称=b3) 公式的意思是在ds2数据集中取第一条符合 单位名称=b3 的已发货字段的数据。

(五)经常出现的错误

扩展方向设置错误

返回集合内容的单元格除了默认纵向扩展之外，我们允许人为地改变单元格的顶格。可以将某个单元格的左顶格设置成某个纵向扩展格、上顶格设置的某个横向扩展格，左顶格和上顶格是分别设置的。

为了符合扩展变化的规则，我们可以知道人为设置顶格需要满足一些条件：

A 左顶格必须是纵向扩展格，上顶格必须是横向扩展格，否则设置无效。

B 不允许出现循环设置的情况，即设置A的顶格是B，B的顶格是C，C的顶格又是A，出现循环设置时认为设置有误，报表无法计算。显然，在缺省的情况下是不可能出现循环设置的，而在人为设置时必须避免这种情况的出现。

C 横向扩展格不允许有左顶格，纵向扩展格不允许有上顶格。

D 人为设置时，可能发生左（上）顶格在右（下）边的情况，而且顶格也不一定和子格在同一行（列）上。

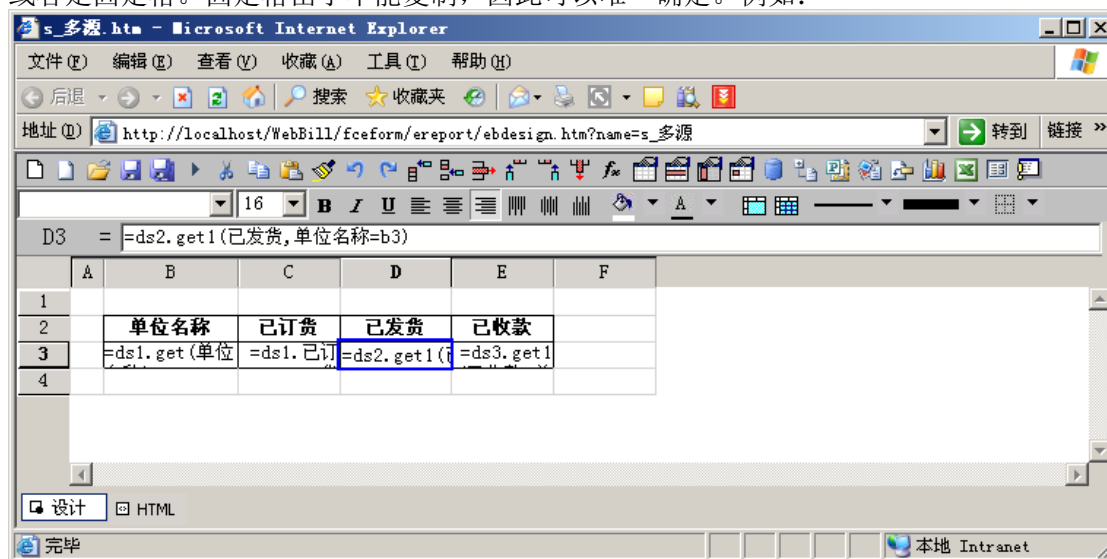
如下面两个图的设置都是错误的：

	A	B
1	横向扩展	
2	纵向扩展	

	A	B
1	纵向扩展	横向扩展
2		

单元格引用错误

一般情况下，被引用单元格往往是当前格的顶格，或者与当前格有相同的一级顶格，或者是固定格。固定格由于不能复制，因此可以唯一确定。例如：



中B3是D3的左顶格,所以在D3中就可以引用B3，反之则不行，如在B3的公式中引用了D3则会出错。

七 使用技巧

(1) 尽量利用当前行来取值

如下图:

	A	B	C	D	E	F	G	H
1_头		职员内码	职员姓名	性别	出生日期	email	电话	
2		=ds1.get	=ds1.sempl	=ds1	=ds1.sbirthc	=ds1.semail	=ds1.sphone	
3								

C2至G2单元格的左顶格都是B2,B2单元格的公式为 `ds1.get(semicolonemployeeid)`,为纵向扩展的。它在扩展时会保存当前行的记录的指针。所以C2到G2单元格可以直接用`ds1.`字段名的方式来取到当前行的值,而如果用 `ds1.get1(字段名)` 的话,系统会在整个数据集中搜索符合条件的行,再取出相应的值,使得效率大大降低。

(2) 利用当前组

`group` 函数是对数据集按照某个字段或者表达式进行分组,获得一组组的集合,然后从每组中取出一个指定字段或者表达式的值,放到单元格中进行扩展,扩展出来的每个单元格都保留了一个指针指向当前的组集,该组集称为当前组。因此在子单元格中,需要对该组集进行操作时,不需要用任何条件和顶单元格关联了,如果加设了条件,反倒画蛇添足,导致报表引擎还对组集中的记录进行遍历检索。

(3) 权衡SQL语句的写法

报表中,经常会出现多个单元格的数据来自同一个数据表,但是取出的条件不一样,或者条件相同,取出的字段或表达式不一样等等,如果你细心研究,会发现这些单元格的数据完全可以通过一个`sql`取出来,然后再通过数据集的函数算出每个单元格的值。

如果将多个`sql`合并以后,取出的数据量并没有增大非常多,那么我们建议您尽量将`sql`合并,充分利用数据集函数中的条件表达式来算每个单元格的值;

如果`sql`合并后,取出的数据量增大了非常多,那么我们建议您尽量将`sql`拆开,减少传输的数据量。

这两者是一个权衡的关系,`sql`多的时候,系统访问数据库的次数多了,导致效率低;反之,`sql`少了,可如果传输的数据量变的非常大,同样会导致效率非常低。所以需要用户根据实际情况权衡利弊。

(4) 尽量在sql 里进行group

对于汇总类型的报表，往往需要进行分组聚集运算，如果在数据库中先进行一次分组聚集，能够大大减少取到报表服务器的记录数，加快取数和报表运算的速度。

(5) 尽量不用select * from

对于初学者来说，这是一个很容易犯的错误，例如报表中只需要用到三个字段，但是数据库中实际的表有几十个字段，一些初学者习惯性的用select * from table1，这样相当于把十个字段的数据都取到报表服务器端，增加了报表服务器端的内存占用以及减慢了运算速度，正确的写法是：select col1,col2,col3 from table1，即用到哪几个字段就取哪几个，用不着的不要取

(6) 尽量在sql 里排序

报表中往往需要对数据进行排序，排序运算可以在数据库中进行，也可以在报表端进行，如果报表中的排序规则是确定的，那么建议排序操作选择在数据库端进行，因为数据库中有索引，且数据库是c 语言开发的，数据运算速度快。

(7) 尽量在sql 里过滤

报表很多时候并不需要对表中的所有记录进行操作，而是对部分满足条件的记录进行操作，因此建议过滤操作在数据库中进行，这样取到报表服务器端的记录数大大减少，既加快了取数的速度，也加快了报表的运算速度，因为报表需要处理的数据少了。

(8) 大数据量可以采用存储过程

有时候，需要用于汇总统计的原始数据量非常大，如果每次生成报表都需要现算，一方面非常慢，另一方面数据库的压力会很大，此时可以采用存储过程对数据预先进行一次压缩，生成中间表，然后再基于中间表生成报表，可以大大提高运算速度并减轻数据库的压力。

(9) or 操作符

使用or 操作符时，尽量把值为true 的可能性更大的条件表达式放在or 的前面，为true 可能性更小的条件表达式放在or 的后面，原因：or 操作符的左右两个操作数，只要有一个为true，其结果必定为true，因此只要第一个条件表达式算出来是true，后面的条件表达式就没必要算了，这样可以加快运算速度。

(10) and操作符

使用and 操作符时，尽量把值为false 的可能性更大的条件表达式放在and 的前面，为false 可能性更小的条件表达式放在and 的后面，原因：and 操作符的左右两个操作数，只要有一个为false，其结果必定为false，因此只要第一个条件表达式算出来是false，后面的条件表达式就没必要算了，这样可以加快运算速度。

(11) 巧用空值判断函数ifnull

表达式中，经常需要用到空值判断，例如在单元格的显示值属性中，判断当单元格的值为空时，显示为0，否则显示单元格的真实值，等等。一般这种情况下，用户习惯写的表达式是：

if(@value==null, 0, @value)。如果我们把@value 换成更加复杂的表达式，例如if(ds1.get1(...)=null, 0, ds1.get1(...))，大家可以看出，这种算法明显很慢，需要把ds1.get1(...)这样的复杂表达式运算两次，而如果采用ifnull()则可以避免这个问题。

(12) 空白单元格的应用

由于e表的界面是个规则的矩形，不可能在中间或者边上挖去一块，因此你会发现报表中时常会多出一些无谓的占位格。在内存中，一个单元格就要占用一块内存，因此单元格越少越好，

这种情况下可以尽量使用空白单元格。空白单元格在内存中是个null，基本不占用内存，因此对于报表边上、中间一些占位格，尽量设成空白单元格，这样既达到了占位的效果，又不会占用内存。

(13) 慎用合并单元格

合并单元格的所有属性都存在左上角的格子中，而合并区域中的其他被合并格并不保存任何属性也不占用内存，因此，做报表的过程中，不少用户习惯对于没用的单元格合并起来，减少内存的占用。殊不知，这种做法虽然减少了内存，但是由于合并格的运算牵扯到主合并格和被合并格的关系，运算比较复杂，会降低运算速度，因此，我们建议：没用的格子设为空白单元格，尽量不要合并。

(14) 慎用隐藏行列

报表中为了进行一些复杂的运算，往往需要用到隐藏行列来处理中间的运算，而这些隐藏行列中被用到的单元格往往只有一两个格子，整行整列的单元格的个数往往很多，此时没被用到的单元格会额外浪费内存，因此要记住把没用的单元格设为空白单元格。