

JSAAS 中的报表开发指导

1. 报表管理

平台提供了上传报表模块，在线设计其数据源、动态查询参数，结合 JasperReport 的强大的报表解析引擎，实现列表、交叉表、图表等各种数据展示方式，以提供给用户的直观的数据决策功能。

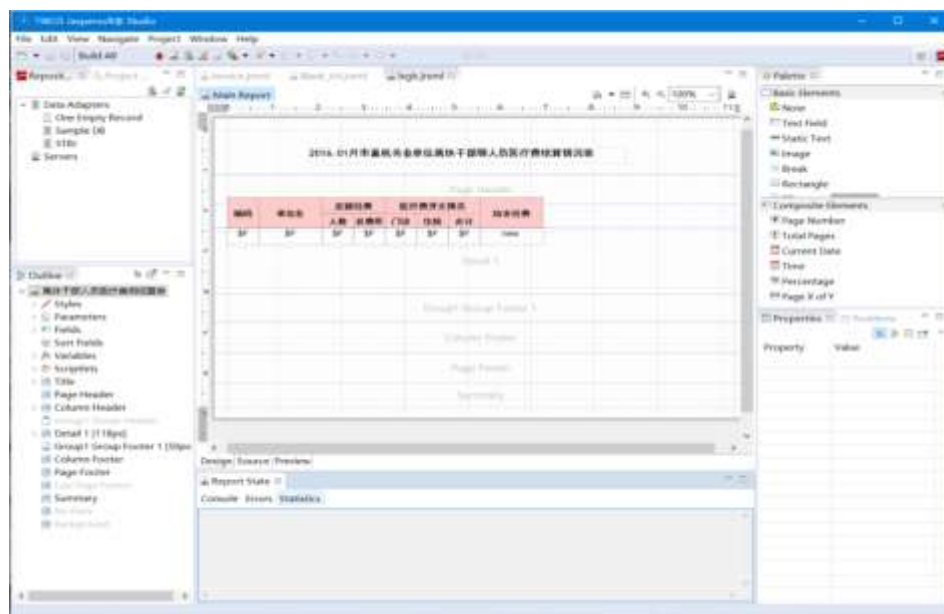
1.1. JasperReort 报表引擎

JasperReport 是一个强大、灵活的报表生成工具，能够展示丰富的页面内容，并将之转换成 PDF，HTML，或者 XML 格式。该库完全由 Java 写成，可以用于在各种 Java 应用程序，包括 J2EE，Web 应用程序中生成动态内容。

1.2. 报表设计工具使用

Jaspersoft Studio 设计器，下载地址为：

<http://community.jaspersoft.com/project/jaspersoft-studio>



Jaspersoft Studio 是一个可视化的报表设计工具，使用该软件可以方便地对报表进行可视化的设计，设计结果为格式.jrxml 的 XML 文件，并且可以把.jrxml 文件编译成.jasper 格式文件方便 JasperReport 报表引擎解析、显示。JasperReport 主要分成三个部分，包括：

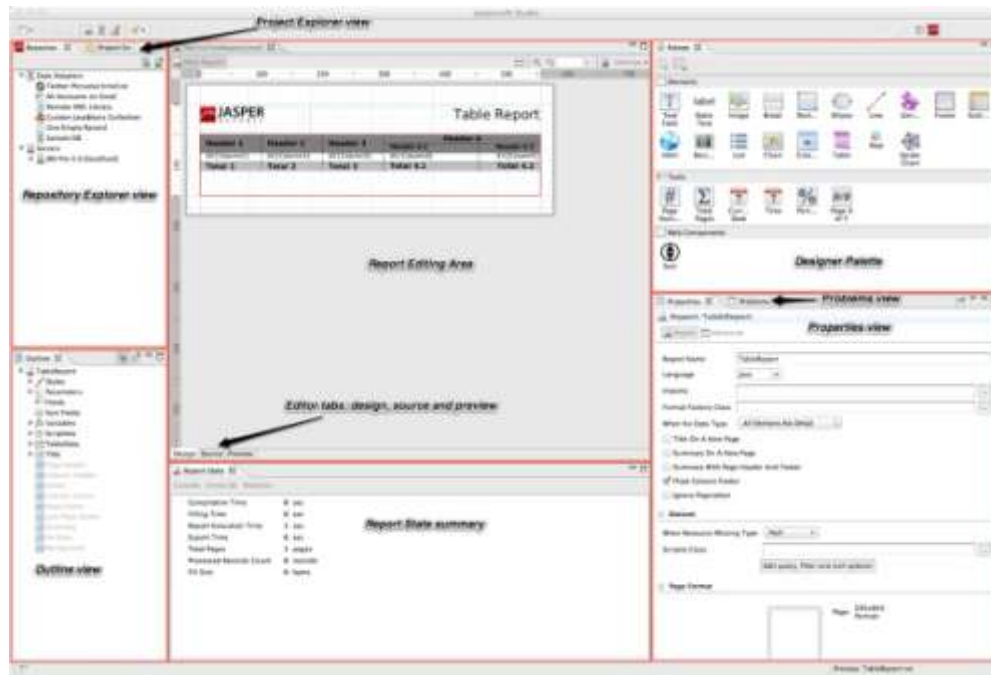
- 报表设计
- 数据填充
- 报表展示导出

一般在 Jaspersoft Studio 中进行报表的设计，导出为.jrxml 或.jasper 格式，然后在 java 中进行数据填充和报表展示导出。

1.2.1. 开发工具介绍

Jaspersoft Studio 是一个专门为 JasperReport 报表引擎而开发的报表设计器。

打开 Jaspersoft Studio 工具，进入如下界面：



- **Report editing area (主编辑区域)** 中，您直观地通过拖动，定位，对齐和通过 **Designer palette (设计器调色板)** 对报表元素调整大小。
- JasperSoft Studio 有一个多标签编辑器，**Design, Source** 和 **Preview**：
- **Design tab**：当您打开一个报告文件，它允许您以图形方式创建报表选中
- **Source tab**：包含用于报表的 JRXML 源代码。
- **Preview tab**：允许在选择数据源和输出格式后，运行报表预览。

很多页面可以查看数据：

- **Repository Explorer view**：包含 JasperServer 生成的连接和可用的数据适配器列表
- **Project Explorer view**：包含 JasperReports 的工程项目清单
- **Outline view**：在大纲视图中显示了一个树的形式的方式报告的完整结构。
- **Properties view**：通常是任何基于 Eclipse 的产品/插件的基础之一。

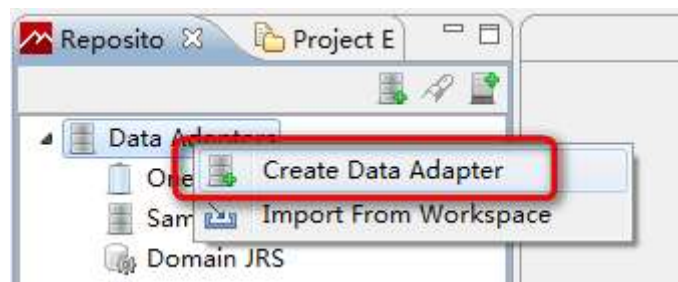
它通常被填充与实际所选元素的属性的信息。这就是这样，当你从主设计区域(即：一个文本字段)选择一个报表元素或从大纲，视图显示了它的信息。其中一些属性可以是只读的，但大部分都是可编辑的，对其进行修改，通常会通知更改绘制的元素（如：元素的宽度或高度）。

- **Problems view**：显示的问题和错误，例如可以阻断报告的正确编译。
- **Report state summary** 提供了有关在报表编译/填充/执行统计用户有用的信息。错误会显示在这里。

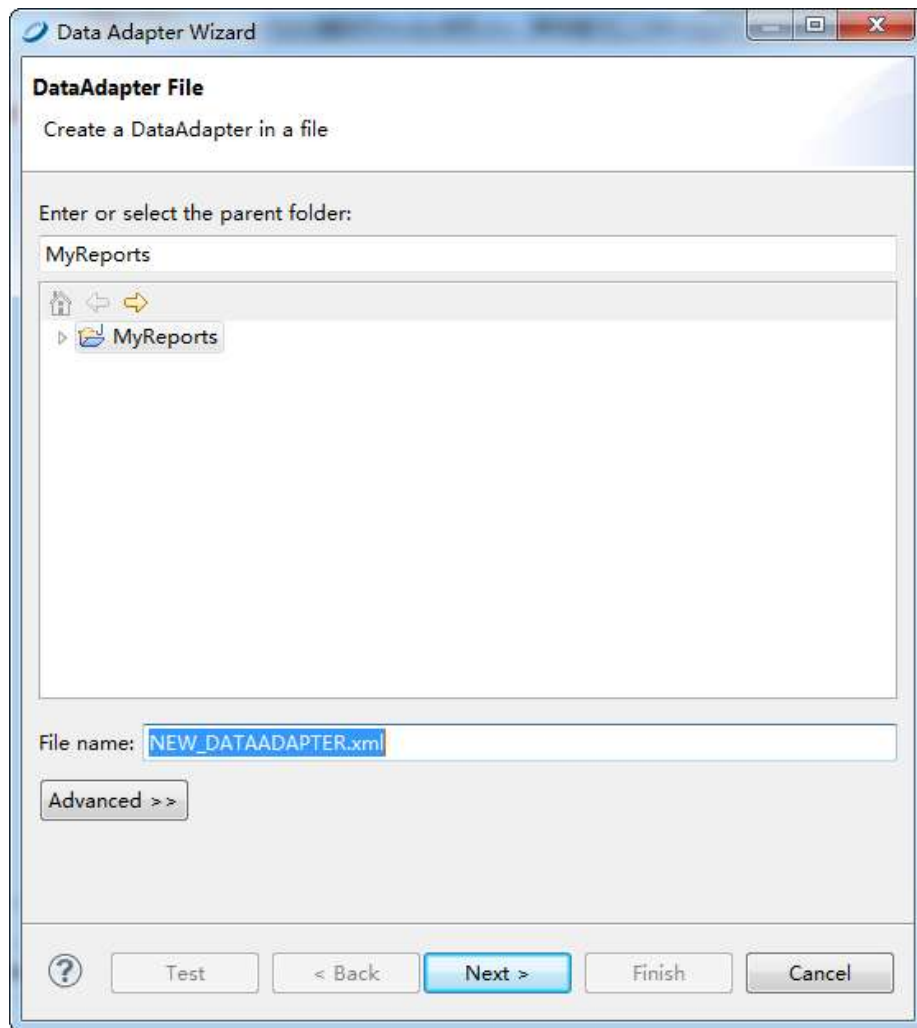
1.2.2. 基本使用

1.2.2.1. 配置数据连接（数据适配器）

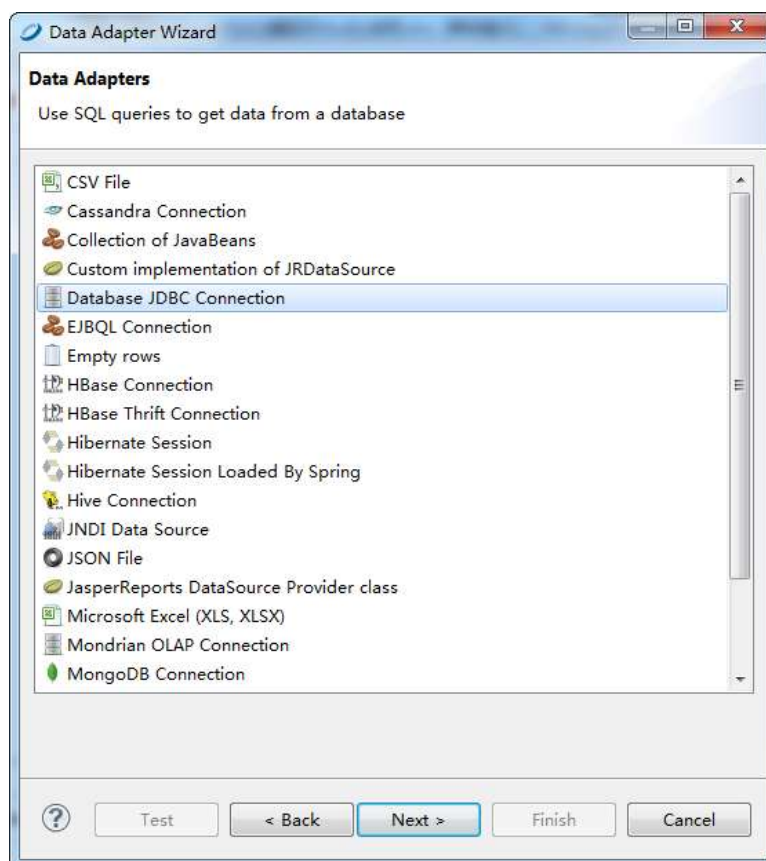
报表是模板+数据结合的展示体，因此需要有数据的来源。在左侧 Repository Explorer 区中，在“Data Adapter”上点击鼠标右键，选择“Create Data Adapter”，创建我们自己的“数据适配器”。



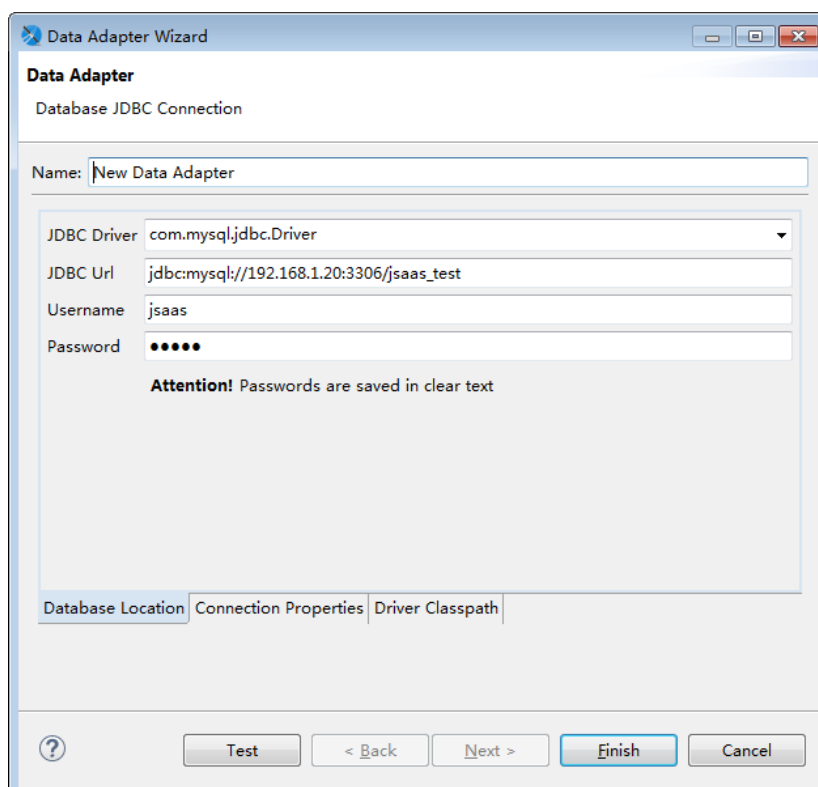
系统显示如下界面：



此处可以对数据适配器配置文件进行重命名，然后点击“Next”按钮。



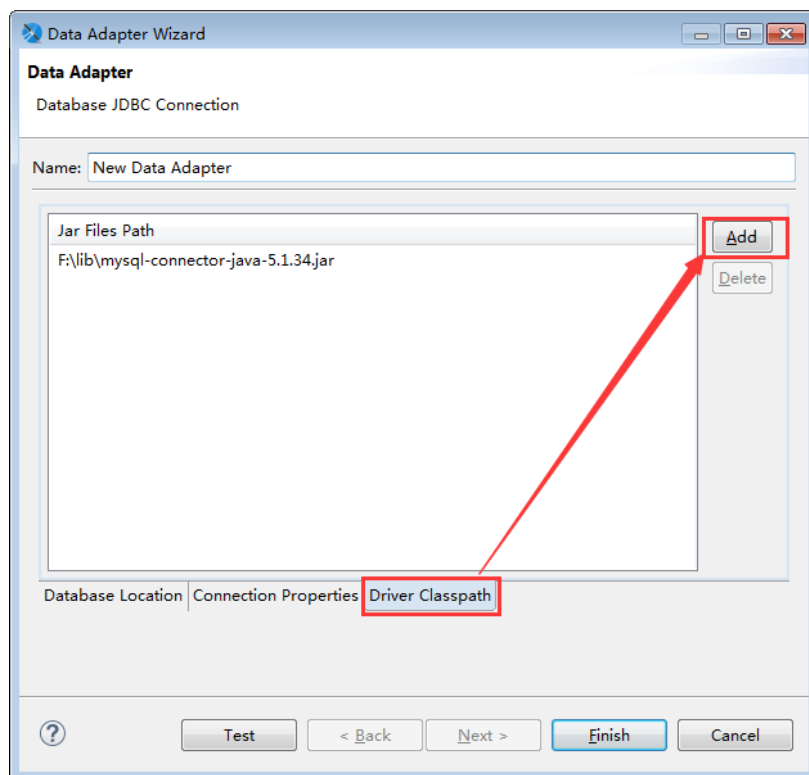
这一步，选择数据源的类型。如选择“Database JDBC Connection”。然后点击“Next”按钮，即进入配置数据库连接属性的界面。



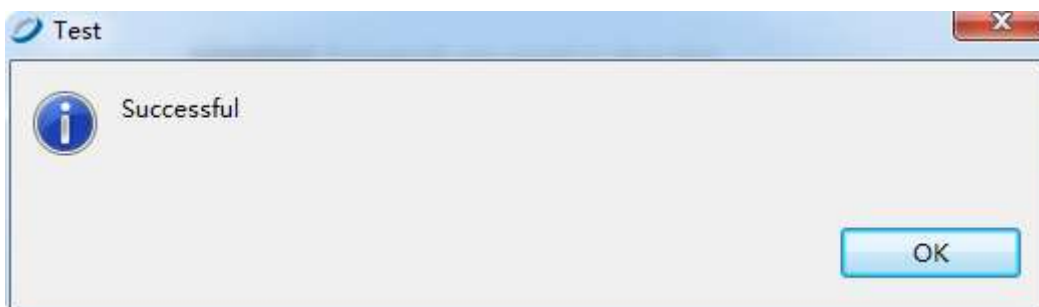
这一步，需要：

- (1) 给创建的这个数据连接起个名字；
- (2) 根据数据库选择驱动类型；

Jaspersoft Studio 已经内置了很多常用数据库的驱动，使用的时候直接选就可以了，若没有对应的数据库驱动，可在如下界面添加 JDBC 驱动包（jar）



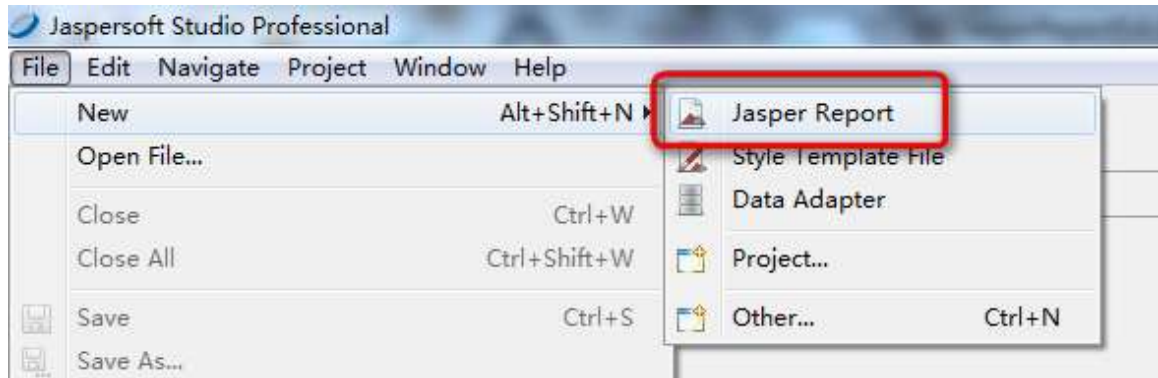
- (3) 修改一下数据连接 URL，要符合你实际的数据库 IP、服务名等的；
- (4) 输入用户名、密码；
- (5) 可以点 “Test”按钮测试一下数据库是否连接成功。



- (6) 完成以上工作后，点 “finish”按钮即可。

1.2.2.2. 创建新报表

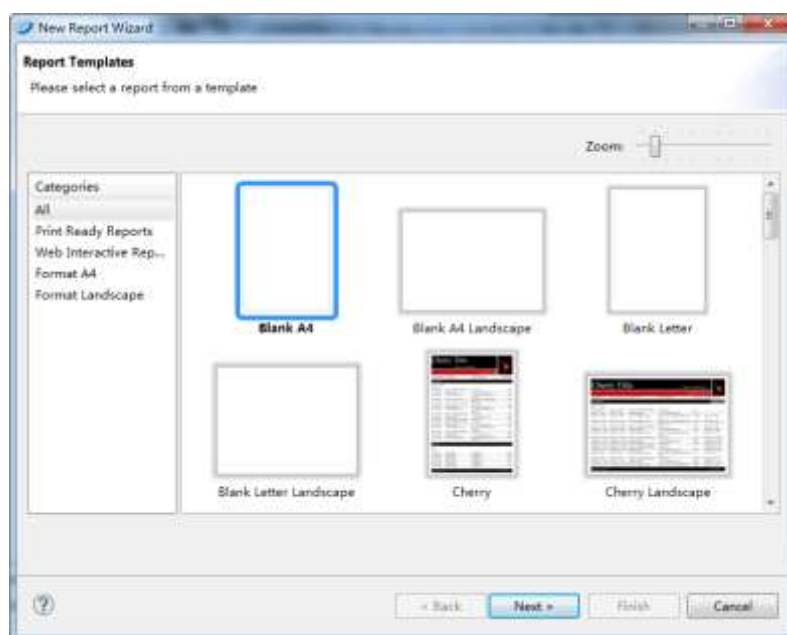
点击菜单中的“File” — “New”，系统会创建一个新的空白报表模板。



或者点击工具栏上的图标。

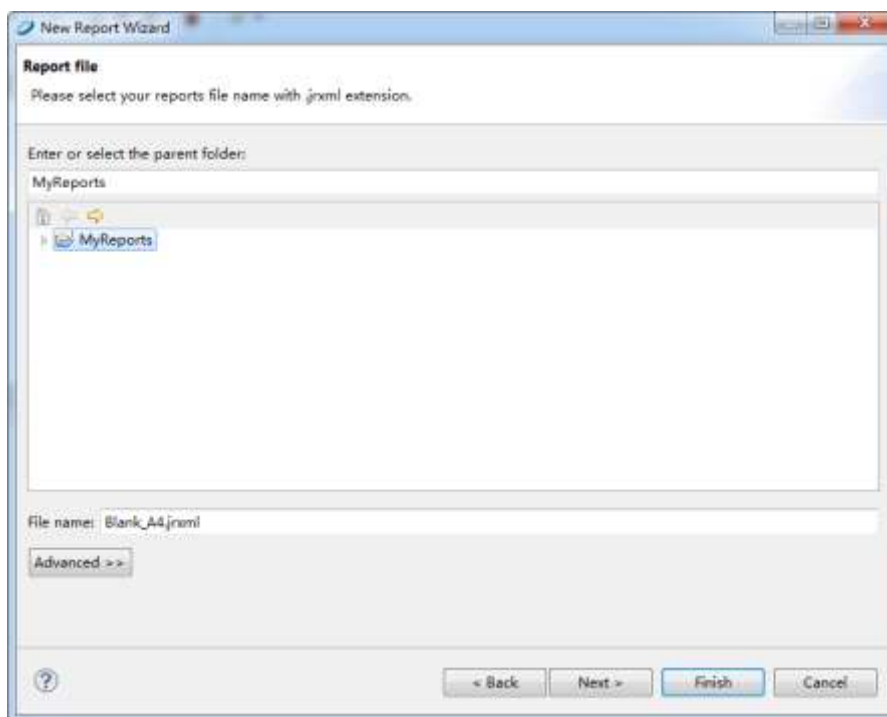


Jaspersoft Studio 提供了丰富多样的报表和图表，几乎可以满足日常工作的各种需要。如我们选择最标准的 A4 竖向空白报表。

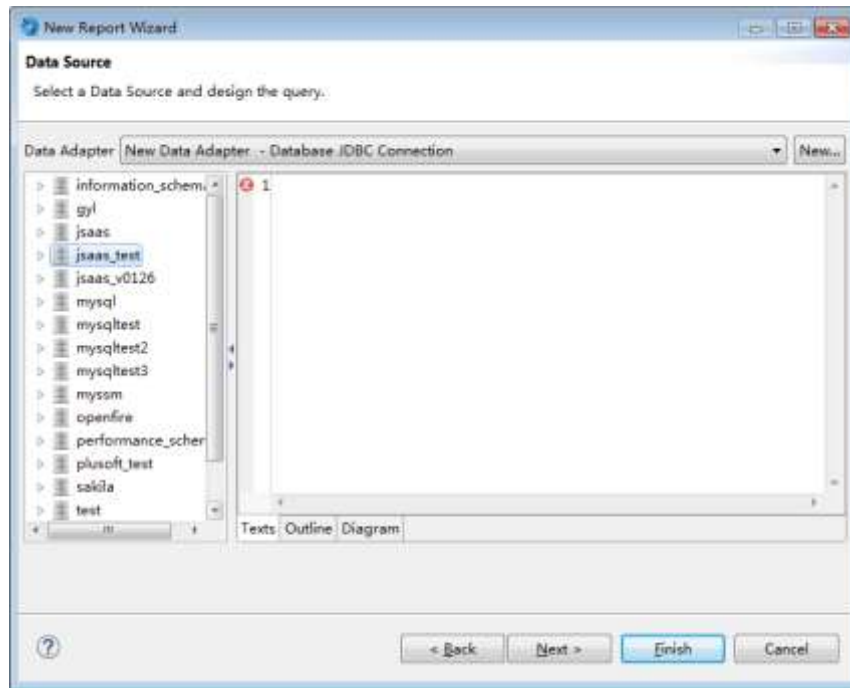


选择完毕，点击“Next”按钮。

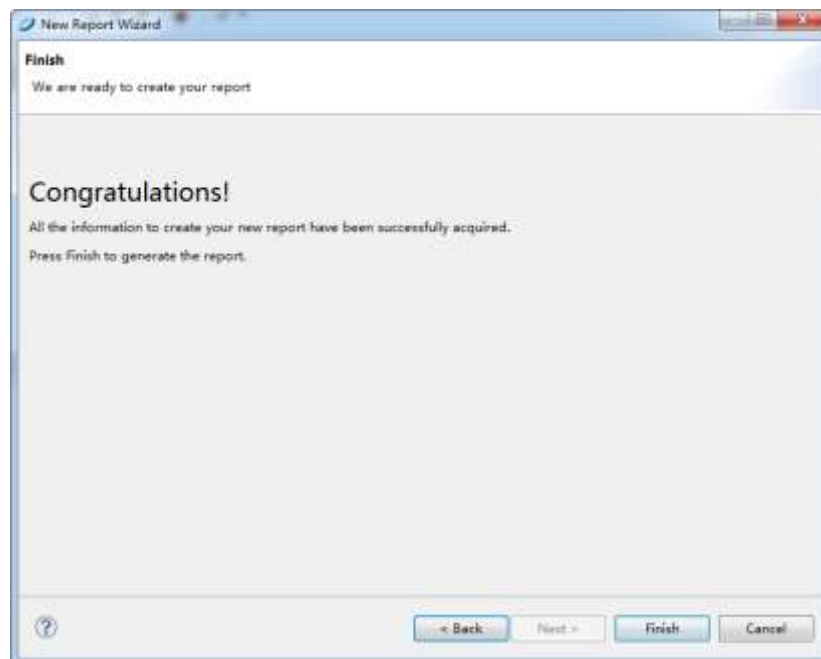
系统显示“名称和位置”界面，可以在这里设置报表的名称，以及报表文件存储的位置。



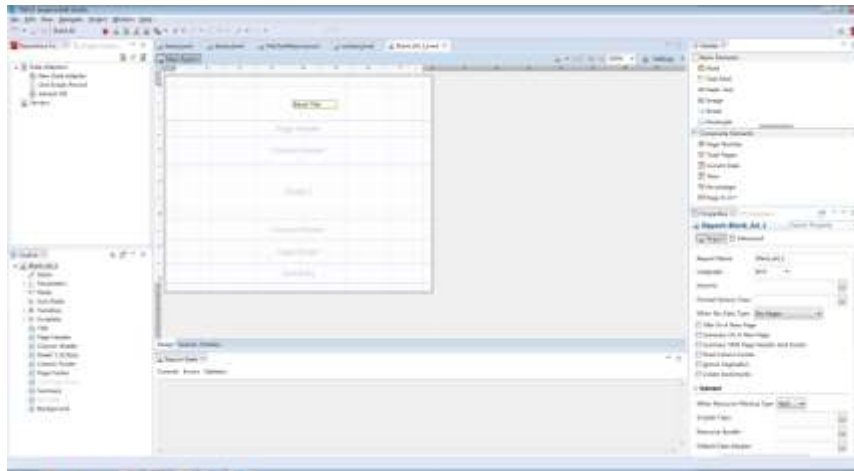
设置完毕，点击“Next”，系统进入“数据源”界面。可以选择已经配置在系统中的数据源，或者点击“New”按钮，创建新的数据源连接。可以在此处设置查询语句，查询需要编制报表的原始数据。



设置完毕，点击“Next”，系统进入“完成”界面。点击“Finish”按钮，即完成了报表的生成。

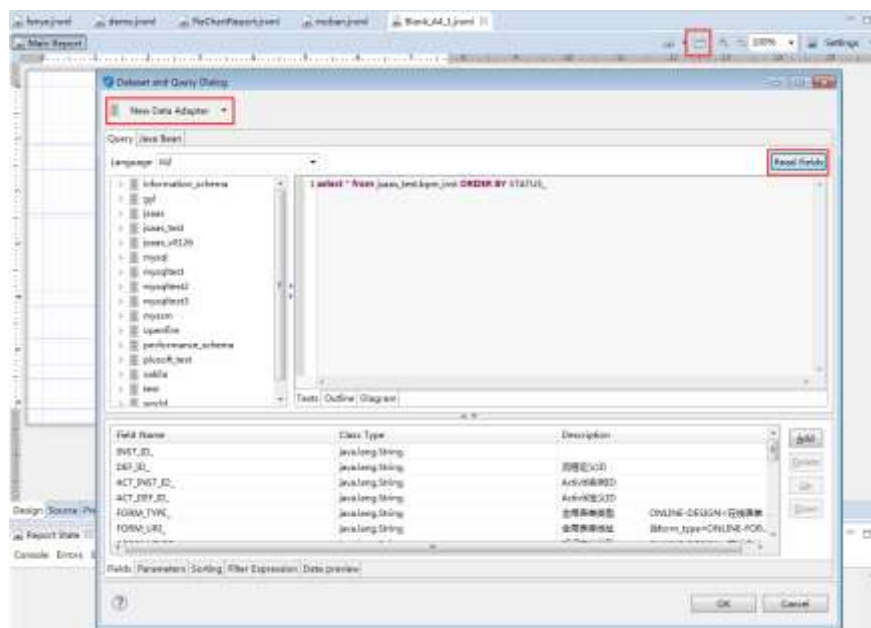


空白报表的界面如下。



1.2.2.3. 设置查询语句

数据库连接设定完毕后，就可以通过 SQL 查询语句，查询出指定的数据，用以设置、填充报表了。点击报表设计工作区中，工具栏上的“DataSet and Query editor dialog”按钮，如下图，系统会显示“DataSet and Query Dialog”界面。



用户可以在 SQL 查询语句输入窗口中，输入需要查询数据的查询语句，点击右上角的“Read Fields”按钮，界面下方的字段列表中，就会显示此查询语句中所涵盖的所有字段的列表。在后面的报表设计中，我们就可以直接使用这些字段

了。输入完毕后，点击“OK”按钮，系统即会把查询语句保存在报表模板中。

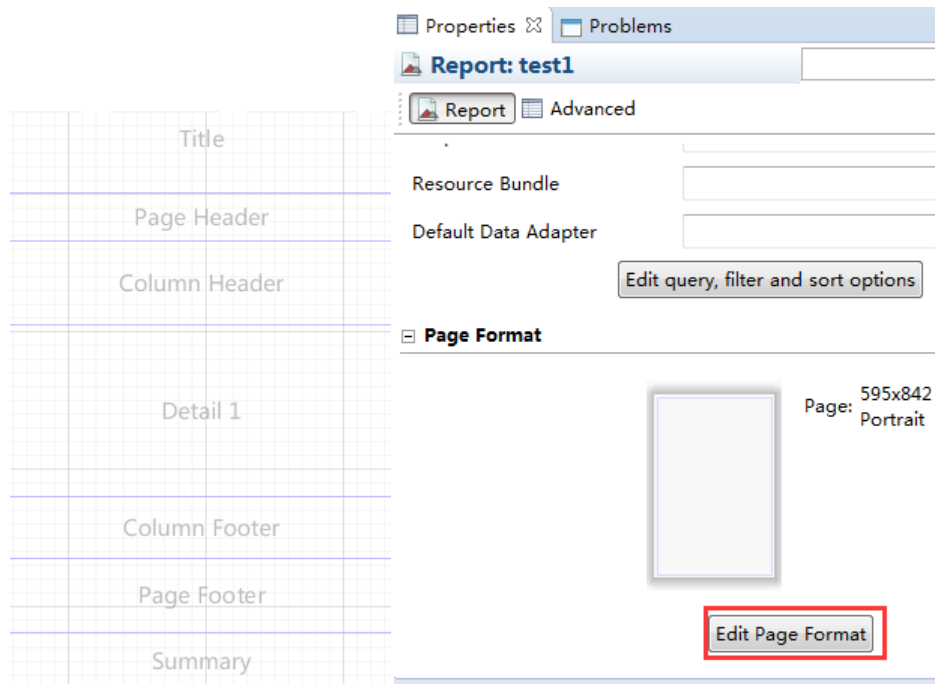
【注意】：

在“Fields”列表中，只保留报表中使用的字段，其他用不到的字段最好用“Delete”删掉，防止由于数据表变化，导致报表模板中的字段设置与数据表对应不上，导致报表报错。

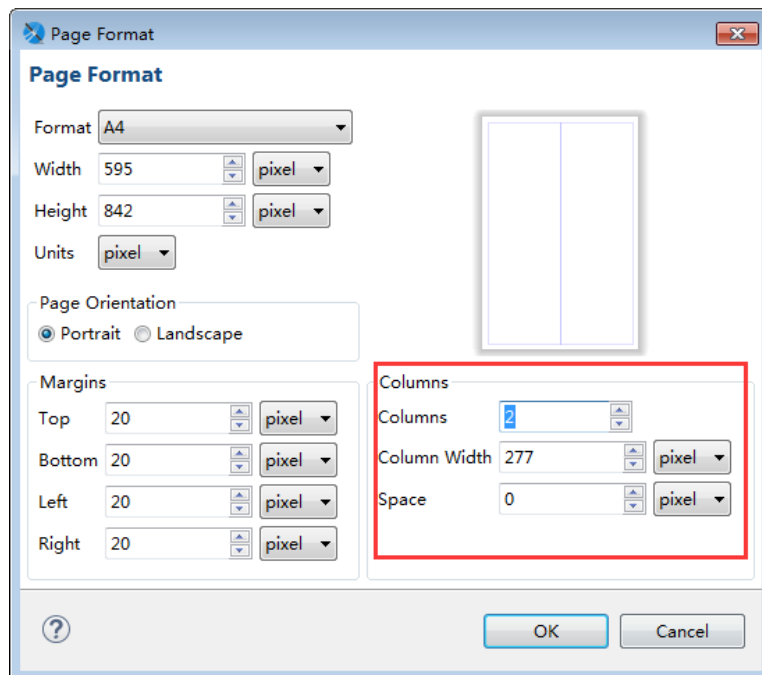
1.2.2.4. 报表各部分组成

下面，就可以正式开始制作报表模板了。首先，需要了解一下报表各部分的组成：

- Title：报表的表头。只在首页打印一次。
- PageHeader：报表的页首。每页都打印。
- ColumnHeader：报表的行首，通常用来定义行的字段名称。每页都打印。
- Detail：报表的内容。根据数据数量，自动循环输出。
- ColumnFooter：报表的行尾，可用来告知报表的一些参数，诸如页码等。每页都打印。
- PageFooter：报表的页尾。每页都打印。
- LastPageFooter：最后一页的页尾。只在最后一页打印一次。
- Summary：计算用。



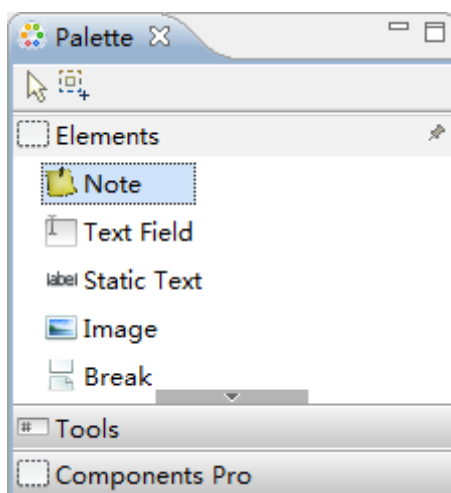
点击报表的 Properties→Report→Edit Page Format ,可以更改报表的高宽等 ,还可以选择报表每页分多少列



1.2.3. 报表明细

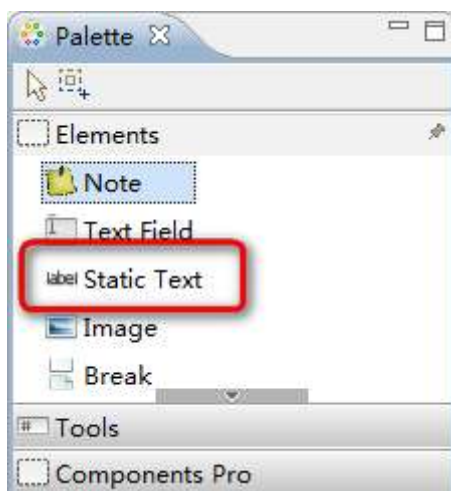
1.2.3.1. 添加表头

我们以制作明细表式的报表为例。我们可以在右侧“Elements”中找到需要的所有组件。



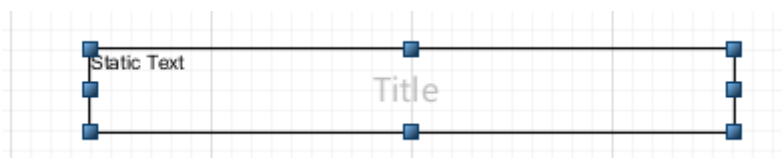
首先需要设定表头。表头中，一般都是报表的名称，所以添加不可修改的文字即可。如果表头只在第一页输出，则需要将表头文字放在“Title”区；如果表头每页都要输出，则应该放在“pageHeader”，即“页首”区。

点击“Elements”上的“Static Text”组件，如下图。



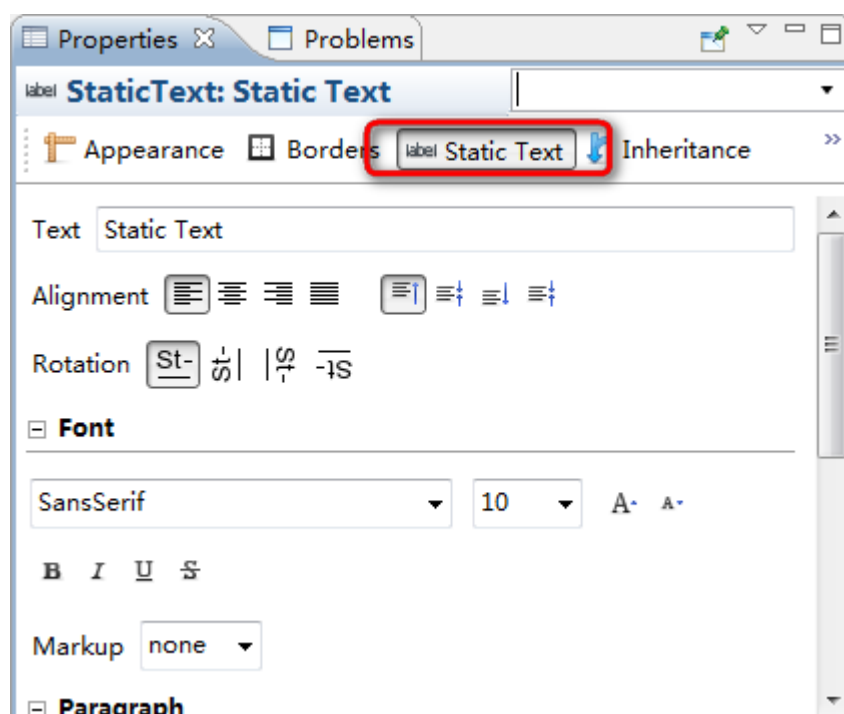
然后将组件拖拽到“Title”区或“pageHeader”区中，即可将组件加入到此

区域中。



在此文字输入区域中输入表头名称即可。

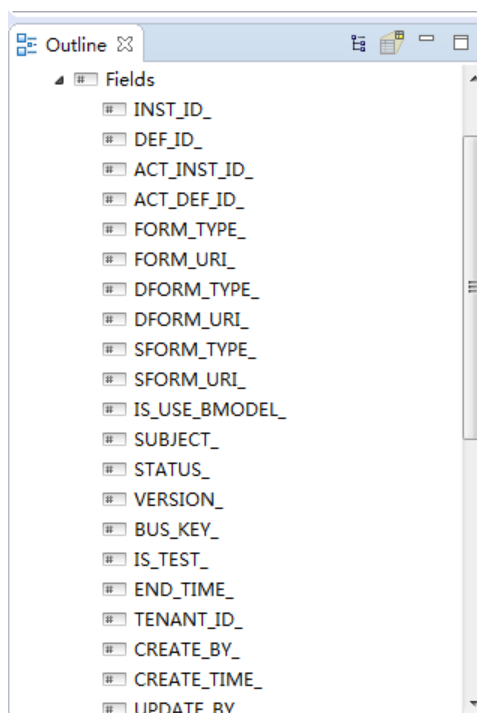
输入完毕，需要设置文字属性，对文字进行排版。在右下角的“Properties”区中，可以设置静态文本框的位置、边框、文字属性等等。点击相应的页签，即可进行对应的设置。



1.2.3.2. 添加字段

设置完表头，即可设置报表需要输出的内容。首先可以设定需要输出的字段。

在左侧“Outline”区中，展开“Fields”项，即可看到此时报表模板中所能访问的所有字段。



如果没有输入 SQL 语句,或者 SQL 语句中包含参数,此处不会自动连接数据库更新字段。此时可以在“Fields”节点上点击鼠标右键,点击弹出菜单中的“Create Field”手工添加字段。但是要注意,字段名称一定要和数据库中的一致,否则会产生错误。



选中一个需要打印的字段,然后拖动至报表中,放在“Detail”区中。拖动所有需要输出的字段,在“Detail”区中排好位置。

系统会自动在“Column Header”区中放置与字段对应的“列头文本”,我们只需调整位置、修改文字、设置格式即可。

测试报表 Title		
Page Header		
INST_ID_	DEF_ID_	SUBJECT_
Column Header		
\$F{INST_ID_}	\$F{DEF_ID_}	\$F{SUBJECT_}
Detail 1		

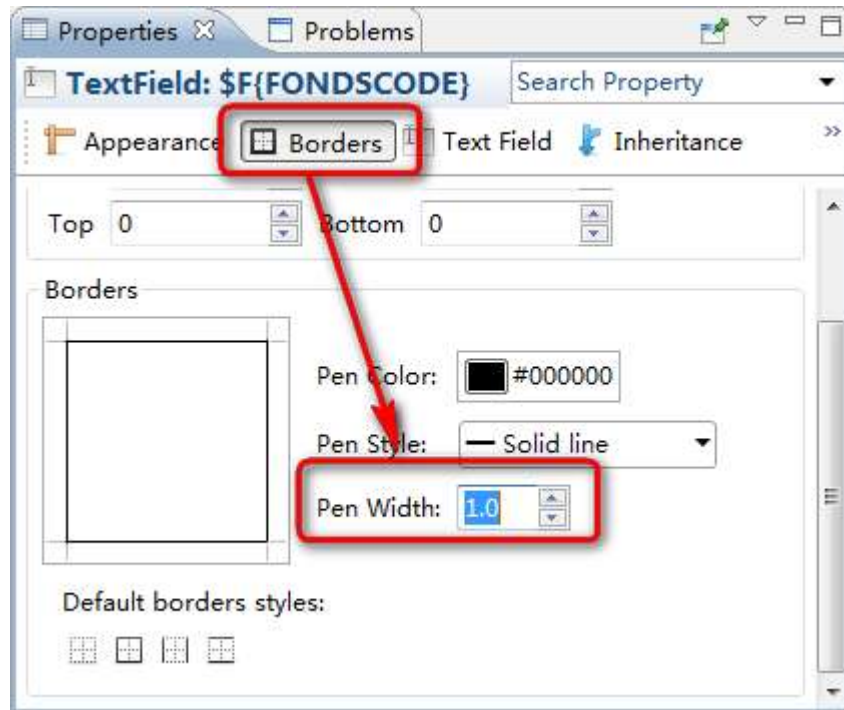
Tips:

如果使用 Excel 作为设置的模板，Excel 中，行高的高度与 Studio 中设置基本相同，而列宽单位与行高单位不同，1 单位（列宽）=6.48 磅（行高），可以根据此公式进行换算。

1.2.3.3. 设定边框

报表中可以设定每个输出域的边框。一般我们会选择四边均有边框的模式。

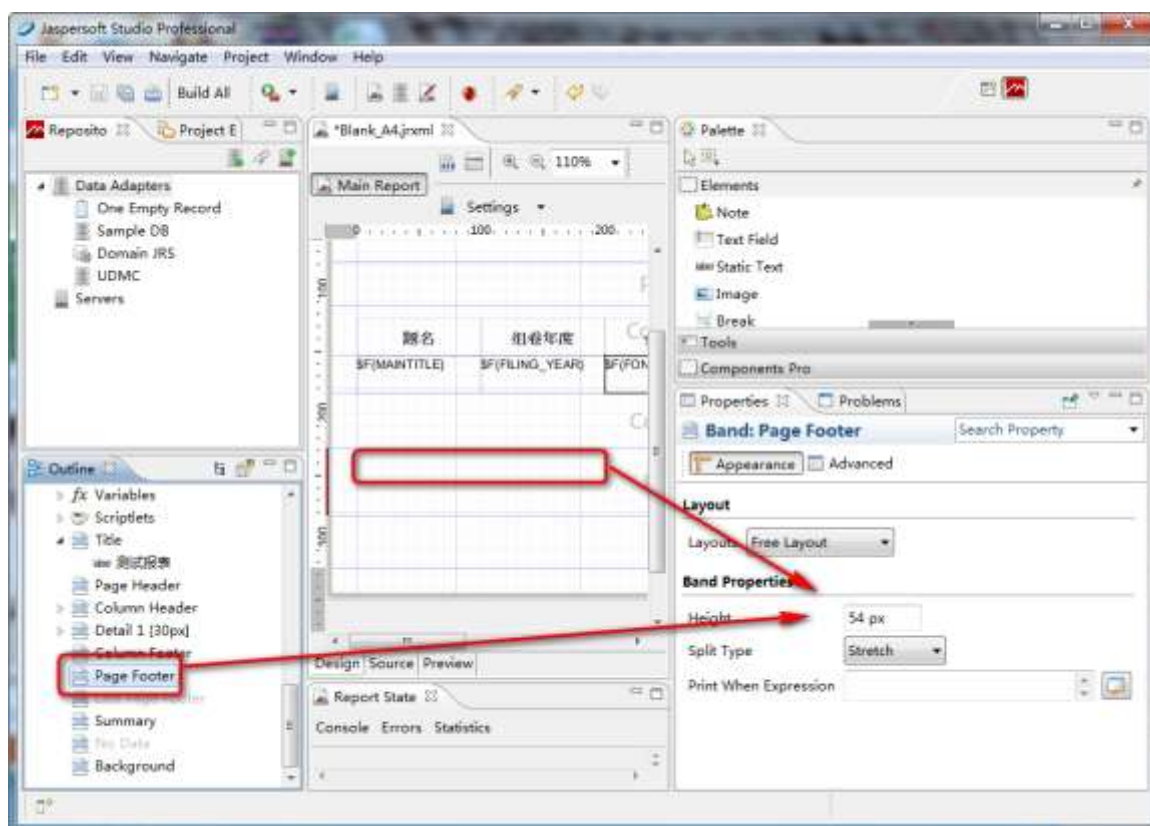
点击需要设定边框的字符或者字段域，在右侧的“Properties”区，选择“Borders”页，修改“Pen Width”（笔宽度），例如设置为“1”，即可显示此域的边框。



1.2.3.4. 设定栏高

为了让报表输出的更美观，我们可以设定报表各个栏宽度。

点击左侧树形图中各个栏的名称，或者直接点击工作区中栏的区域，在右侧“Properties”区中，即可修改栏的高度。



可以在此处设置各栏的高度，不需要的栏，可以将高度设置为 0。

为了使 Detail 区中输出的报表内容的表格可以相连，需要将 Detail 区的高度设为与字段域高度相同。本例中，字段域高度为 30，则 Detail 域的高度也为 30。

1.2.3.5. 设置字段域

在打印报表时，有的时候会遇到字段中的内容很长，标准的表格打印不下的情况。此时，用户就会希望表格能够自动适应内容的多少而自动增加行高。在 JasperReport 中可以通过设置实现。

选中所有需要设置的字段域，在右侧“Properties”区中可以进行如下设置。

Font	
Bold	<input type="checkbox"/> <INHERITED>
Font Name	SansSerif
Font Size	<INHERITED>
Italic	<input type="checkbox"/> <INHERITED>
PDF Embedded	<input checked="" type="checkbox"/> true
PDF Encoding	UniGB-UCS2-H (Chinese Simplified)
PDF Font Name	STSong-Light
Strike Through	<input type="checkbox"/> <INHERITED>
Underline	<input type="checkbox"/> <INHERITED>
Graphic	
Hyperlink	
Location	
Misc	
Anchor Name Expression	
Bookmark Level	0
Key	
Print In First Whole Band	<input type="checkbox"/> false
Print Repeated Values	<input checked="" type="checkbox"/> true
Property Expressions	[Properties: 0]
Remove Line When Blank	<input type="checkbox"/> false
Transparent	<input type="checkbox"/>
Print When	
Detail Overflows	<input checked="" type="checkbox"/> true
Group Changes	
Print When Expression	
Size	
Height	30
Stretch Type	Relative To Tallest Object
Width	100
TextField Properties	
Blank When NULL	<input checked="" type="checkbox"/> true
Evaluation Group	<NULL>
Evaluation Time	Now
Expression	\${FONDSCODE}
Pattern	
Pattern Expression	
Stretch With Overflow	<input checked="" type="checkbox"/> true
Text Properties	

- 1、“Font” - “PDF embedded”：选择 “true”，可以将报表输出到 PDF 中。
- 2、“Font” - “PDF Encoding”：选择 “UniGB-UCS2-H (Chinese Simplified)”，保证 PDF 中的中文可以正确编码。
- 3、“Font” - “PDF Font Name”：选择 “STSong-Light”，保证 PDF 报表中的中文可以正确显示。
- 4、“Misc” - “Print repeated values”：选择 “true”。在打印时，可以输出相同的值。
- 5、“Print When” - “Detail Overflow”：选择 “true”。在数据当页没有打印

完毕，需要在第二页打印时，可以将表格的内容，包括边框，在第二页中输出。

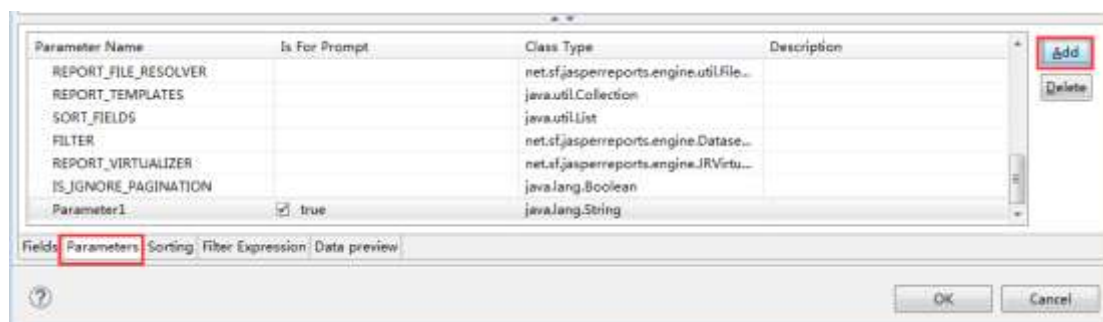
- 6、“Size” - “Stretch Type”：选择 “Relative to Tallest Object”。则所有被选中的字段域可以自动适应条目拉伸。
- 7、“TextField Properties” - “Blank when null”：选择 “true”，是当字段的值是 Null 时，输出空白格。
- 8、“TextField Properties” - “Stretch with overflow”：选择 “true”，当文本域的内容不能完全被显示在模板定义的区域时，允许文本域拉伸。

1.2.4. 动态报表参数

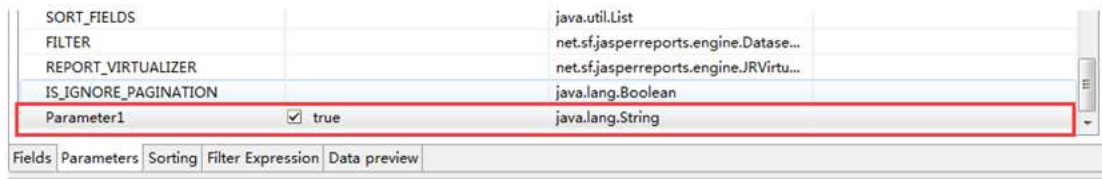
1.2.4.1. 添加参数

以上报表模板使用的 SQL 查询语句是固定的，不能随着程序的运行而改变。使用 Studio 制作报表模板，可以通过使用“参数”功能，接收程序传递过来的参数表，并将参数传递到报表模板中，从而实现根据程序运行的情况，动态组合 SQL 查询语句，动态生成报表数据。

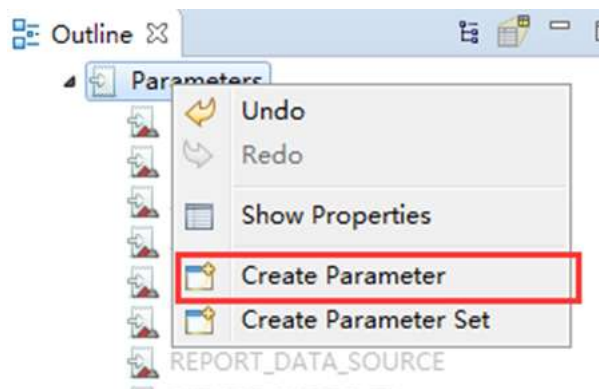
在“查询设置”界面的下方，点击“Parameters”，就可以切换到参数列表界面。



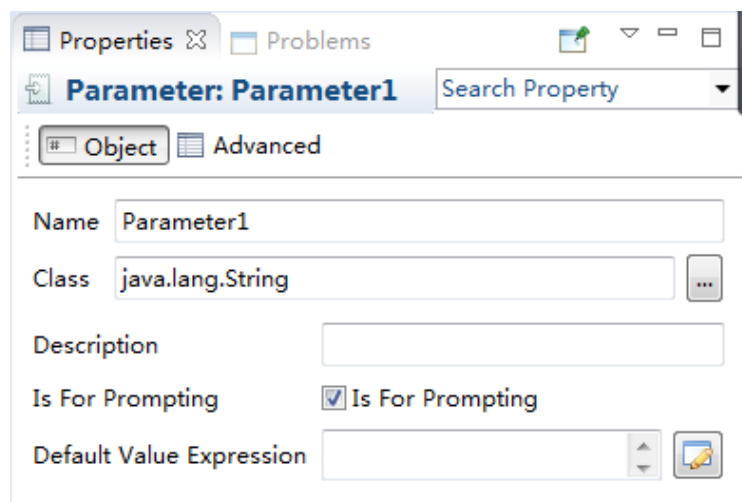
点击 “Add”，就可以添加新的参数。



在 “Parameter Name” 中添加变量名称，在 “Class Type” 选择变量的数据类型。也可以在 Outline 页面右键“Parameter”，然后选择“Create Parameter”



然后在右边的 Properties 窗口修改属性。



1.2.4.2. 程序中传递参数

添加完毕的变量，既可以在变量列表中浏览到，可以在 Java 程序中通过 Map 传递过来。

1.2.4.3. 报表模板接收参数

要想在报表模板中使用变量，需要在 SQL 查询语句中输入 “\${P{变量名}}”，如此格式进行调用。

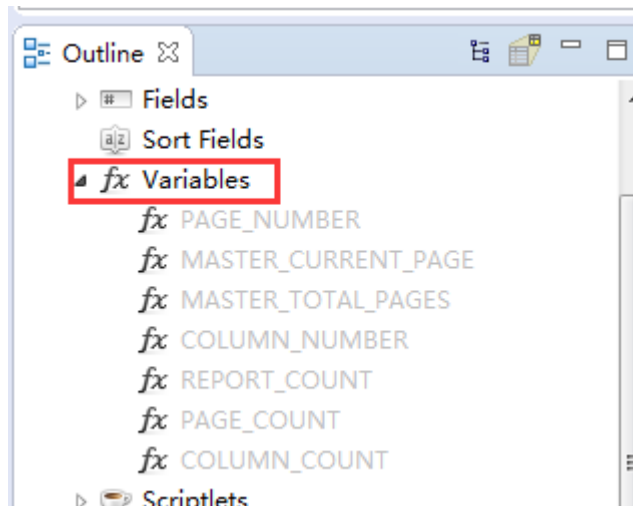
最简单的办法是，直接在参数列表中选中需要加入的参数，然后拖拽到 SQL 语句中。默认拖拽过来的参数显示为 “\${P{变量名}}”。



1.2.5. 参数变量的使用

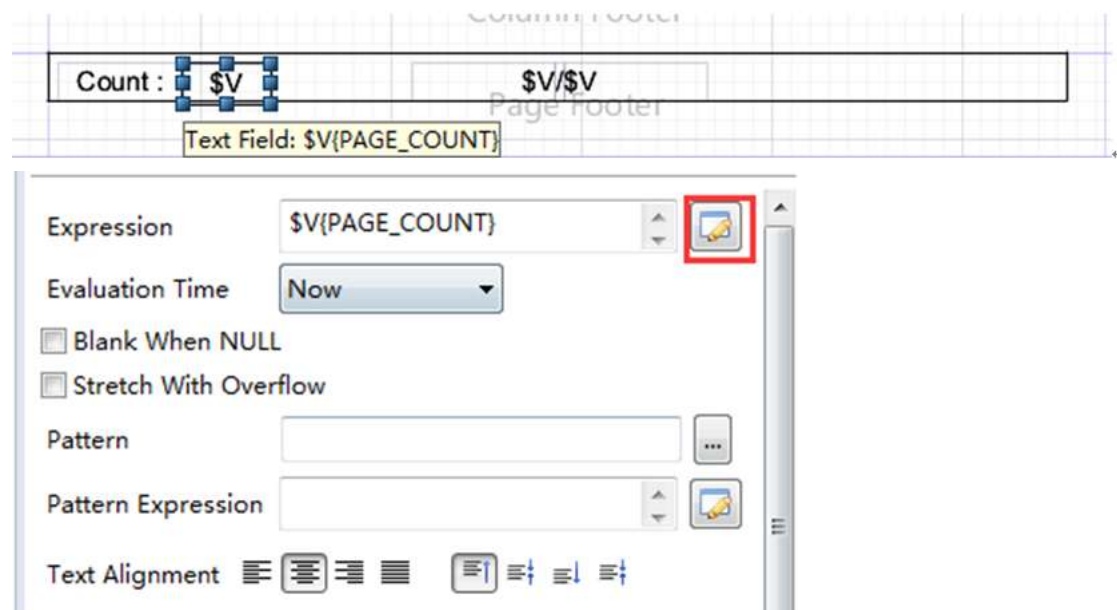
1.2.5.1. 默认变量的使用

在 Outline 页面，点击“Variables”，可以看到系统默认自带的变量，以 number 结尾的就是页数、行数，以 count 结尾的就是该页，该报表，该行所包含的 details 数量。

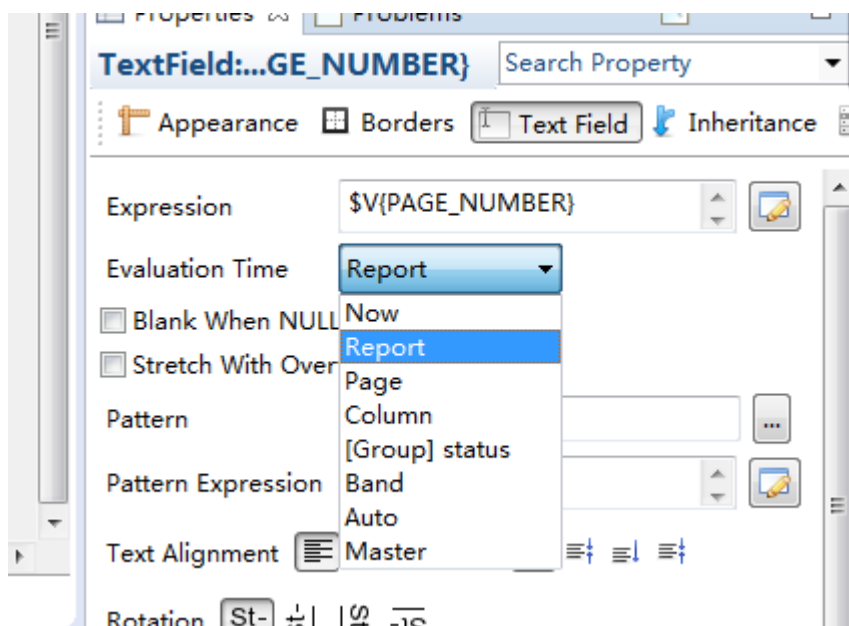


将一个 Text Field 拖到 Footer 中，然后点击“Properties”—“Text Field”—“Expression”，将表达式选为该页面的所有 detail 数量“PAGE_COUNT”。

0



其中“page_number”变量有多重表达方式：



当为 now 和 page 时，就是默认显示的当前页数

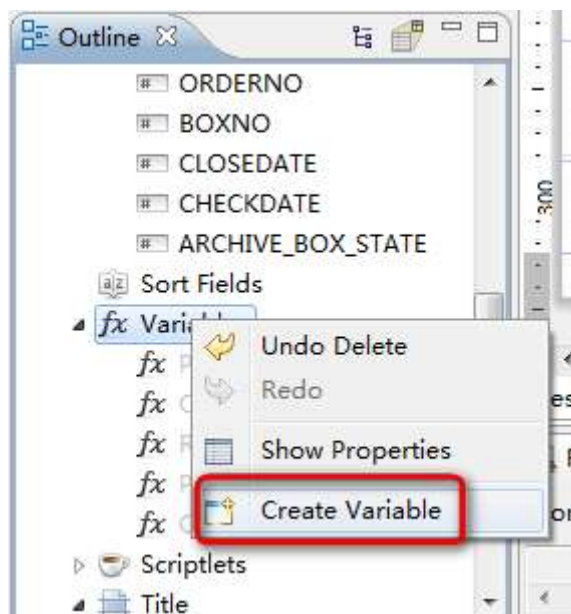
当为 report 时，则为显示所有页数

Let's see what properties can be set for a text field:

Blank when null	If set to true, this option avoids printing the text field content if the expression result is a null object that would be produce the text "null" when converted in a string.
Evaluation time	Determines in which phase of the report creation the Text field Expression has to be elaborated.
Evaluation group	The group to which the evaluation time is referred if it is set to Group.
Stretch with overflow	When it is selected, this option allows the text field to adapt vertically to the content, if the element is not sufficient to contain all the text lines.
Pattern	The pattern property allows you to set a mask to format a value. It is used only when the expression class is congruent with the pattern to apply, meaning you need a numeric value to apply a mask to format a number, or a date to use a date pattern.

1.2.5.2. 自定义变量

在“Outline”区中的“Variable”节点上点击鼠标右键，选择“Create Variable”，新增加一个变量。



在右侧的“属性”区中，可以设置新增的变量的属性。

Object Advanced

Name	statusCount
Value Class Name	java.lang.Integer
Calculation	Count
Expression	#{STATUS}
Initial Value Expression	
Increment type	None
Incrementer Factory Class Name	
Reset type	[Group] status

各属性解释：

Name：变量的名称改为“rank”。

Value Class Name：变量的数据类型。此处根据数据库的不同，字段类型的不同，选择不同的数据类型。例如，Oracle 的数值型字段均定义为 Number 型，则此处需要选择“java.math.BigDecimal”；若为 SQL Server 或 MySQL 数据库，整型为 Integer，则此处选择“java.lang.Integer”。如此类推。

Calculation：计算函数。有求和，求平均，求最大等。

Expression：变量计算公式。可以在此处输入变量统计时计算的公式，可以使用系统中的变量、字段等资源，通过调用不同的方法，实现公式的设置。

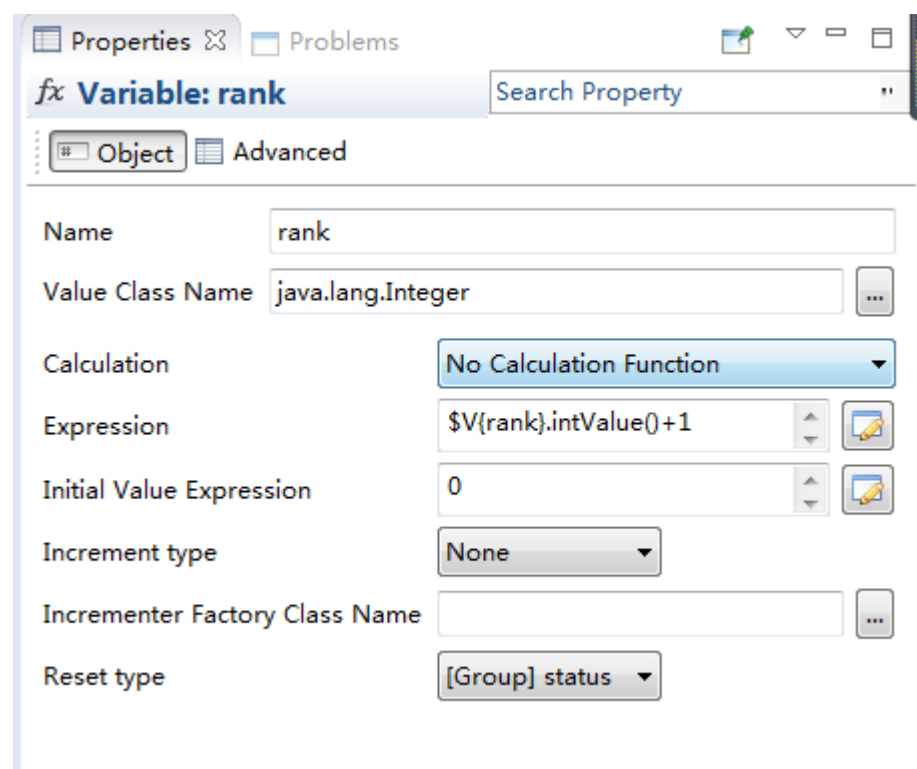
Initial Value Expression：变量重置时初始值。可以留空，系统可以自动根据字段类型赋予 0 或者空值。也可以手工赋予指定类型的数据。例如，本例中留空。

Increment type：增长类型。表示变量增长的计算范围。如果在分组报表中使用，此处选择“group”，即在分组范围内进行累加；如果在普通报表中使用，此处选择 None。

Incrementer Factory Class：增长工厂类。一般用不到。

Reset type：重置类型。表示变量在什么时候做重置操作。如果是在分组报表中使用，此处选择 Group，当分组变化时，即重新合计；如果在普通报表中使用，此处选择 Report。

上例变量的意思是计算每个 Group 中的 status 数量。



The screenshot shows the configuration window for a variable named "rank". The window has tabs for "Object" and "Advanced". The "Object" tab is selected. The configuration fields are as follows:

Property	Value
Name	rank
Value Class Name	java.lang.Integer
Calculation	No Calculation Function
Expression	$\$V\{rank\}.intValue()+1$
Initial Value Expression	0
Increment type	None
Incrementer Factory Class Name	
Reset type	[Group] status

上例变量意思为顺序号，从 1 开始，每更换一个 group 更新重新计数。

1.2.6. 分组报表

有两种情况会使用分组报表：

数据分组统计显示清晰

另一种是当数据分为两层表时，经常需要批量打印子表的数据。打印时，常常需要按照父表的外键或关联值进行自动分组，即每一条父表记录所属的子表记录打印到一组报表中，每组报表都单独计数及计算页数。

在应用中，可以通过选择需要打印的父表记录，将父表记录的 ID 传入，由报表自动进行分组。

示例：

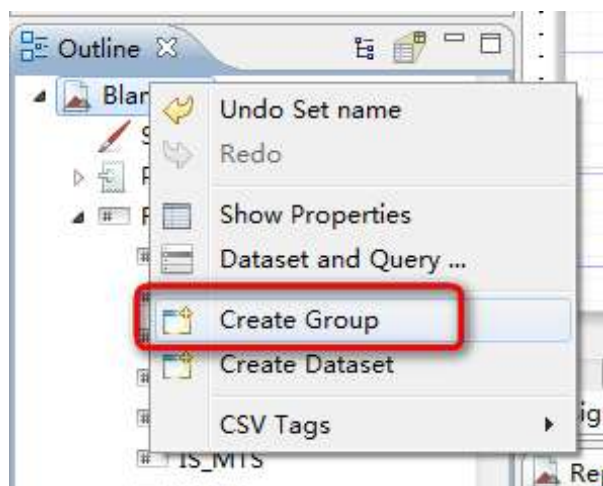
1、设置报表查询语句，输入如下 SQL 语句：

```
select
    wjh.filingcode as filingcode,
    wj.note as note
from DAT_ARCHIVE_GDWJ wj left join DAT_ARCHIVE_GDWJH wjh
    on wj.pid=wjh.id order by wj.pid
```

此处一定要用分组的字段进行排序，否则生成的报表是错误的。

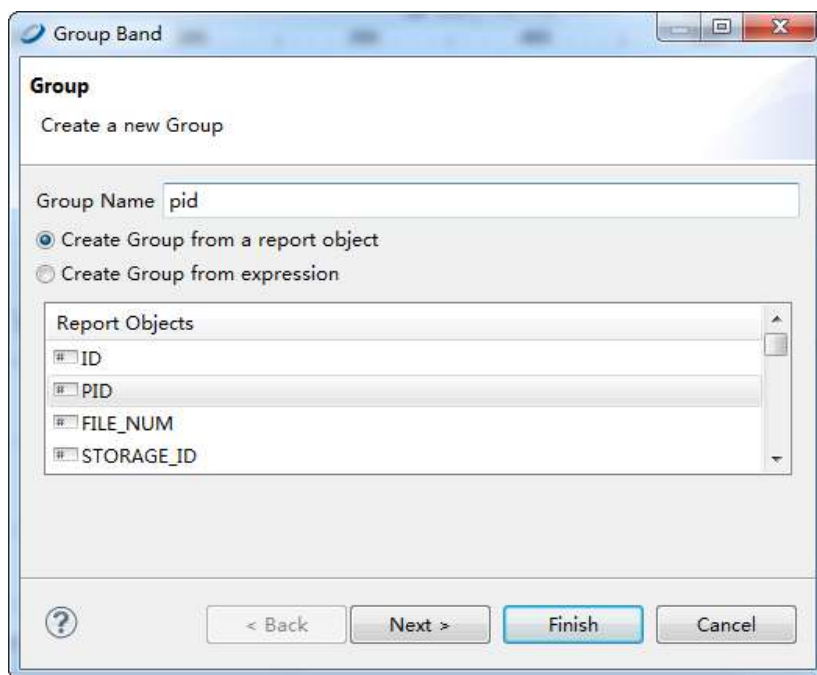
2、新建报表群组

选中报表名称点击右键，选择菜单中的“Create Group”。



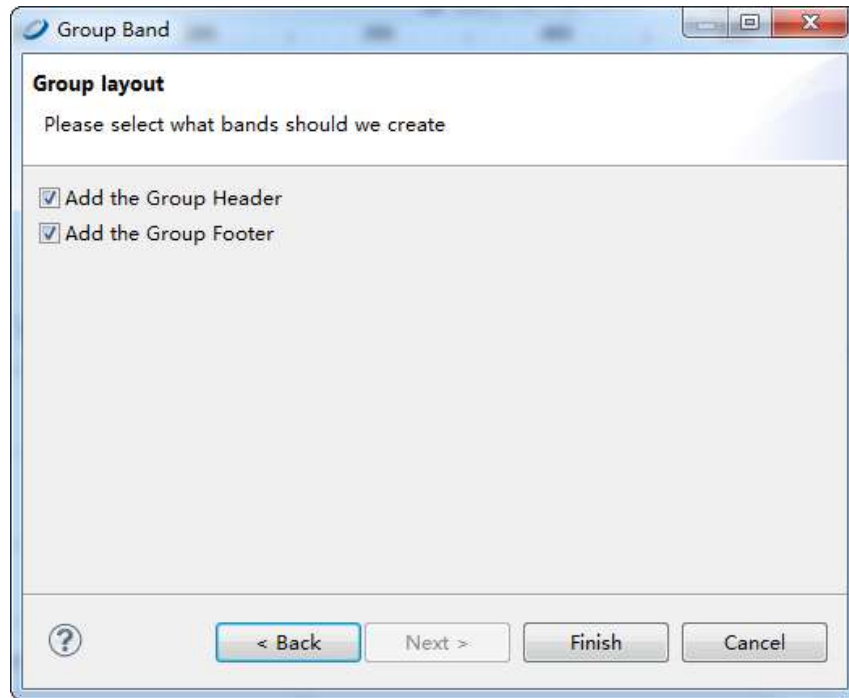
系统显示分组创建向导界面。

第一个界面中，需要设置分组的名称、分组字段。也可以设置按照指定的函数、方法处理后进行分组。



本例将“Group name”设置为“pid”，按照字段“pid”进行分组。设置完毕，点击“Next”。

系统显示细节设置界面。此处可以设置是否加入“group header”和“group footer”区。建议保持默认选中，加入这两个区域，这样可以控制在每组报表的结尾，打印相应的信息，例如统计信息或者父表所属的字段内容等。



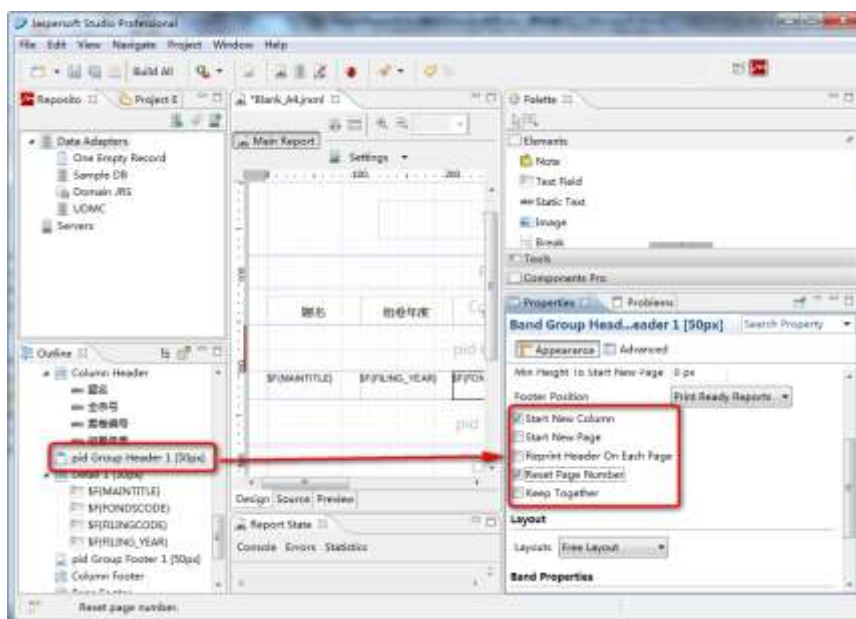
设置完毕，点击“Finish”即可。

3、放置报表数据

将需要作为表头打印的内容拖入 pid Group Header1 栏 将字段拖入 detail 栏，
将每个分组结尾需要打印的内容放入 pid group footer1 栏，将页脚需要打印的内
容放入 Page Footer 栏，如下图。

4、设置分组分页打印

选择分组表头，在右侧的属性栏中可以设置分页打印的选项，如下图。



“Start on a new page”，即在新页面打印分组头和分组尾。

“Reset page number”，即新页面中重置页码。

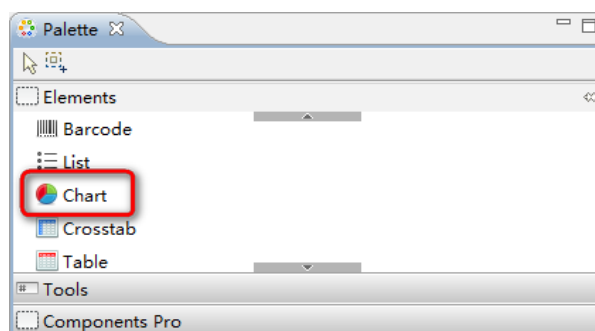
1.2.7. 图形报表

- 1、设置报表查询。例如，在查询设置界面中输入如下 SQL 语句。

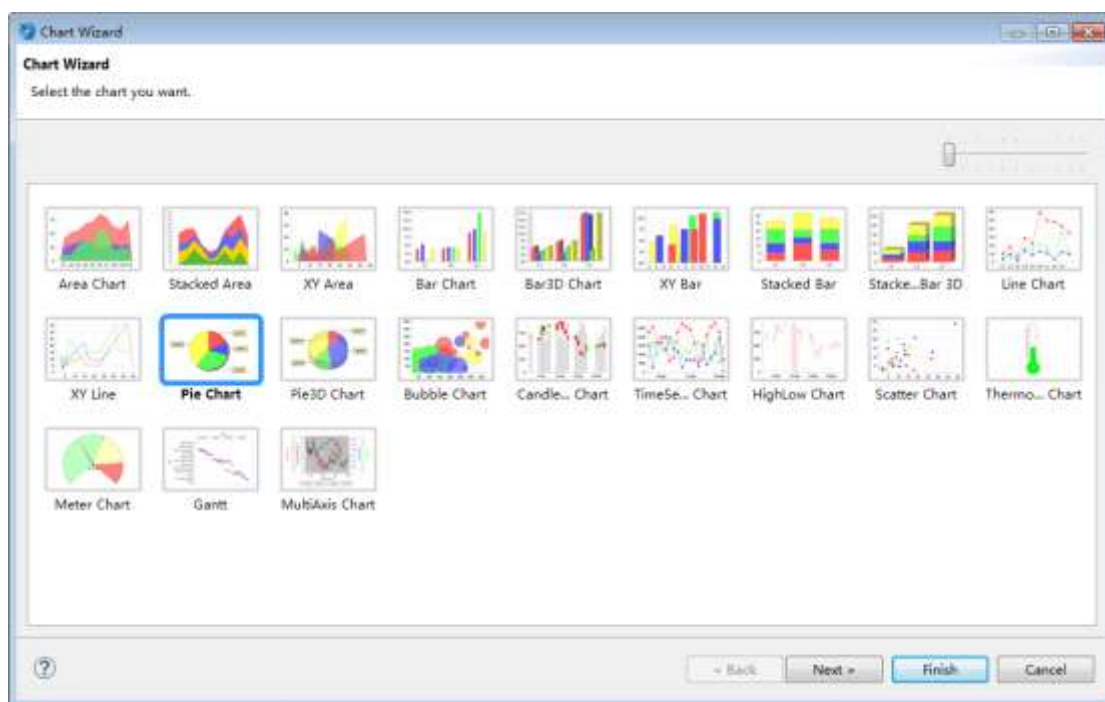
```
select * from jsaas_test.bpm_inst ORDER BY STATUS_
```

一定要分组显示

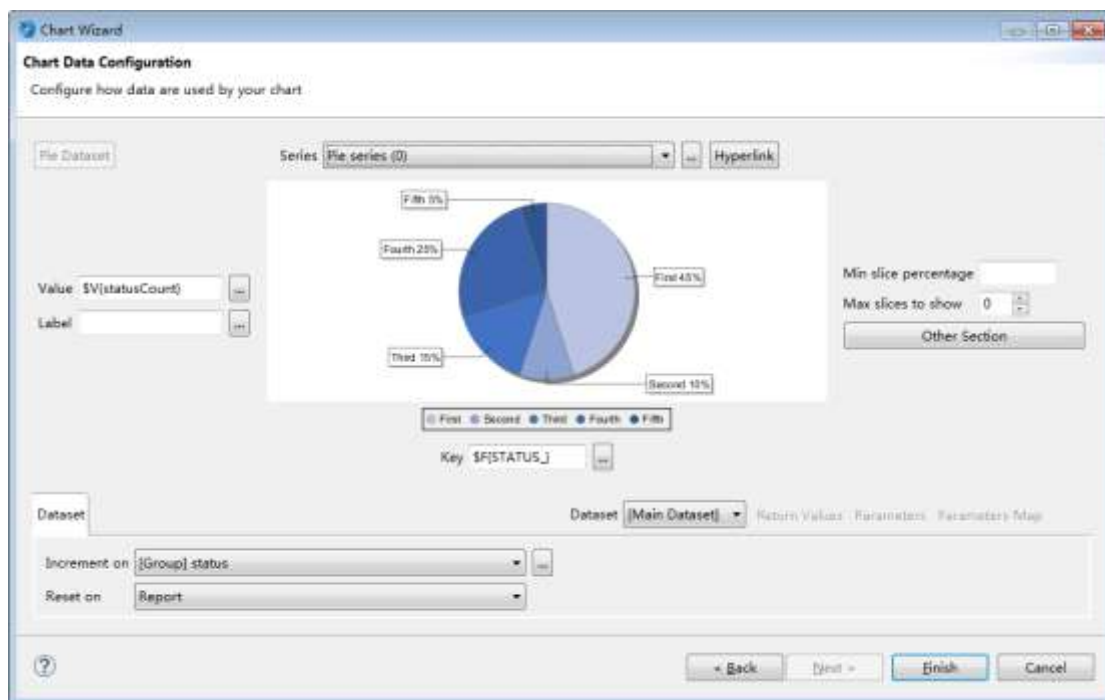
- 2、删除一些报表栏。将 columnHeader.detail.colmnFooter 等栏高度设置为 0。
- 3、增加 Summary 栏的高度，设为 400。
1. 插入饼图，放置在 Summary 栏。选择如图所示的按钮，插入图形报表。



选择三维饼图。



4、点击“Next”，可设置饼图的参数。



Key：圆饼图的内容是什么，也就是下面的 First，Second...的内容

Value：这个圆饼图的比例依据，根据 Value 属性来显示每个 Key 占的比例

Increment on：这里要选择为[Group]XXX

Figure 12-9 Chart Wizard > Chart Data Configuration

In this window you select data to use in your chart.

Depending on the dataset type that you have selected, the window's **Chart Data** tab shows the fields within the specified dataset. Detailed descriptions of the various field types and their functionality are available in the JasperReports Library Ultimate Guide.

2. The **Max slices to show** field is useful for charts with many slices. When set, your chart only shows the number of slices you specify; additional values are included in a slice labeled **Other**.

Set this field to 10.

3. On the **Dataset** tab, you can define the dataset within the context of the report.

The **Reset on** drop-downs allow you to periodically reset the dataset. This is useful, for example, when summarizing data relative to a special grouping. **Increment on** specifies the events that determine when new values must be added to the dataset. By default, each record of the dataset used to fill the chart corresponds to a value printed in the chart. This behavior can be changed, forcing the engine to collect the data for the chart at a specific time (for instance, every time the end of a group is reached).

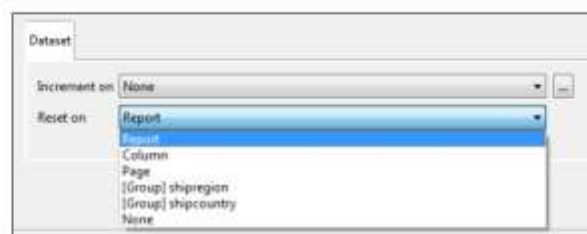
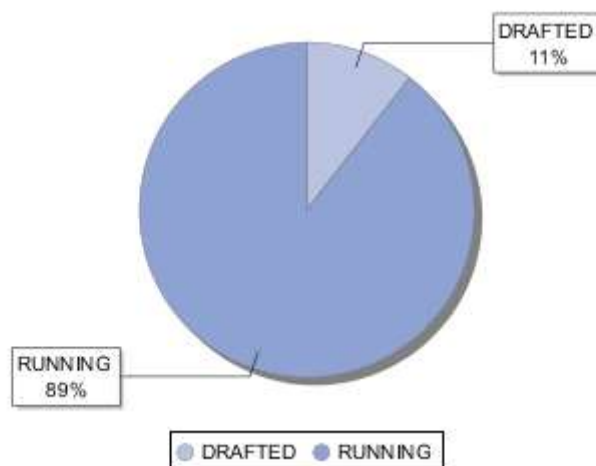


Figure 12-10 Dataset Tab

Set **Reset on** to **Report** since you don't want the data to be reset, and leave **Increment Type** set to **None** so that each record is appended to your dataset.

4. Also in the **Chart Data Configuration** dialog, enter an expression to associate with each value in the data source. For a Pie 3D chart, three expressions can be entered: *key*, *value*, and *label*.
 - **Key expression** must be a unique value to identify a slice of the pie chart. If a key value is repeated, the label and value values previously associated with that key are overwritten. A key can never be null.
 - **Value expression** specifies the numeric value associated with the key.
 - **Label expression** allows you to specify a label for each slice of the pie chart. This expression is optional, and the default value is the key value.

5、查看结果。



1.2.8. 子报表

1.2.8.1. 制作父报表

首先制作父报表，就是调用子报表的一个基础报表。

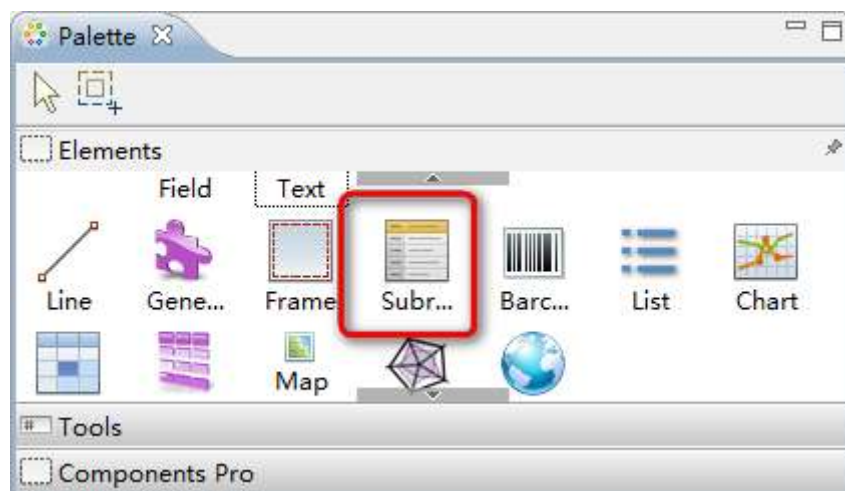
如果父报表中需要显示数据，在 SQL 编辑器中输入 SQL 语句，查询出需要的字段，将字段加入到报表中即可。

如果父报表不需要显示任何数据，只是作为子报表的载体，那么也需要在 SQL 编辑器中输入语句，并且需要能查询出记录，系统才会在生成报表时显示父报表，否则，系统会显示一个空白报表。

本系统中，建议输入 “select id from sys_user where user_name='admin'”，只查询出一条记录即可使报表正常生成。

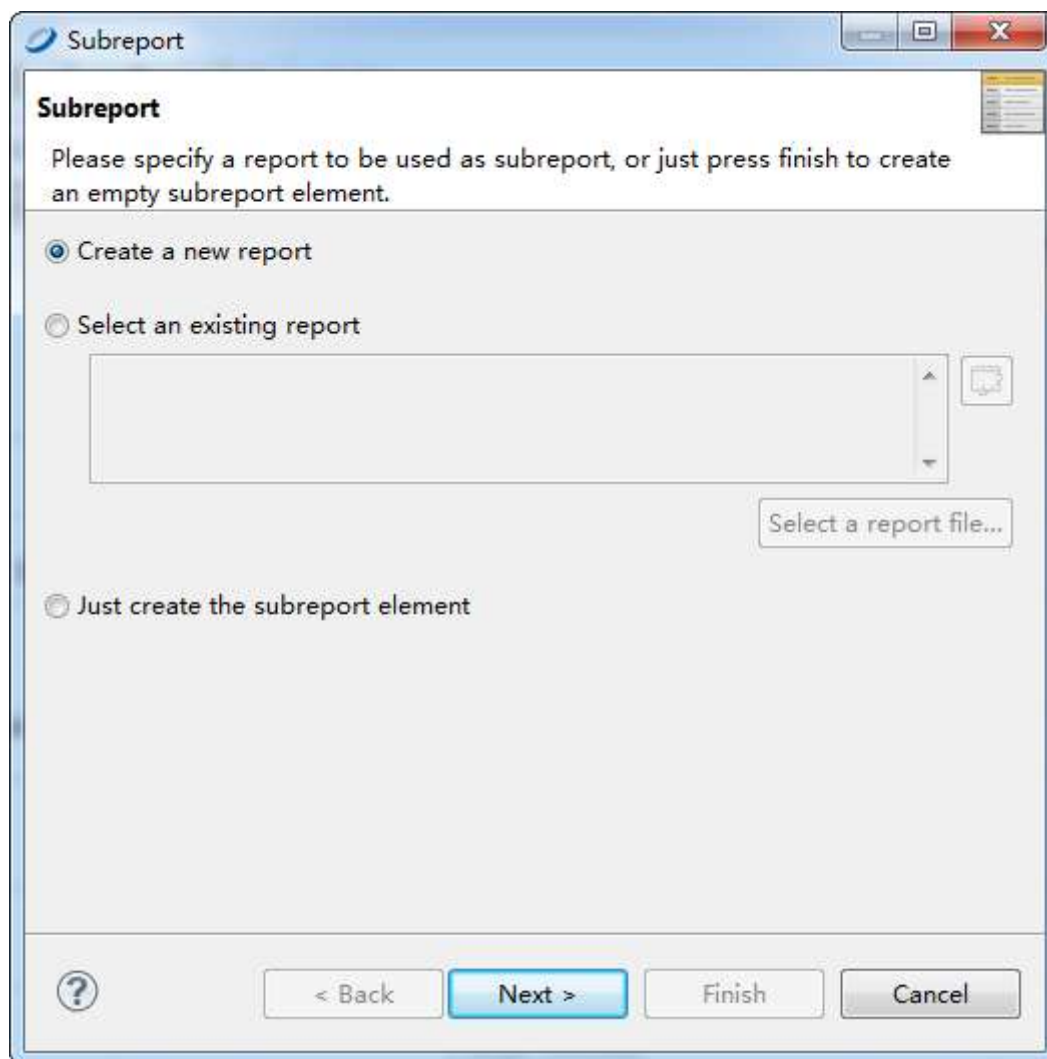
1.2.8.2. 制作子报表

点击组件面板上的 “Subreport” 按钮，拖动到报表工作区上。



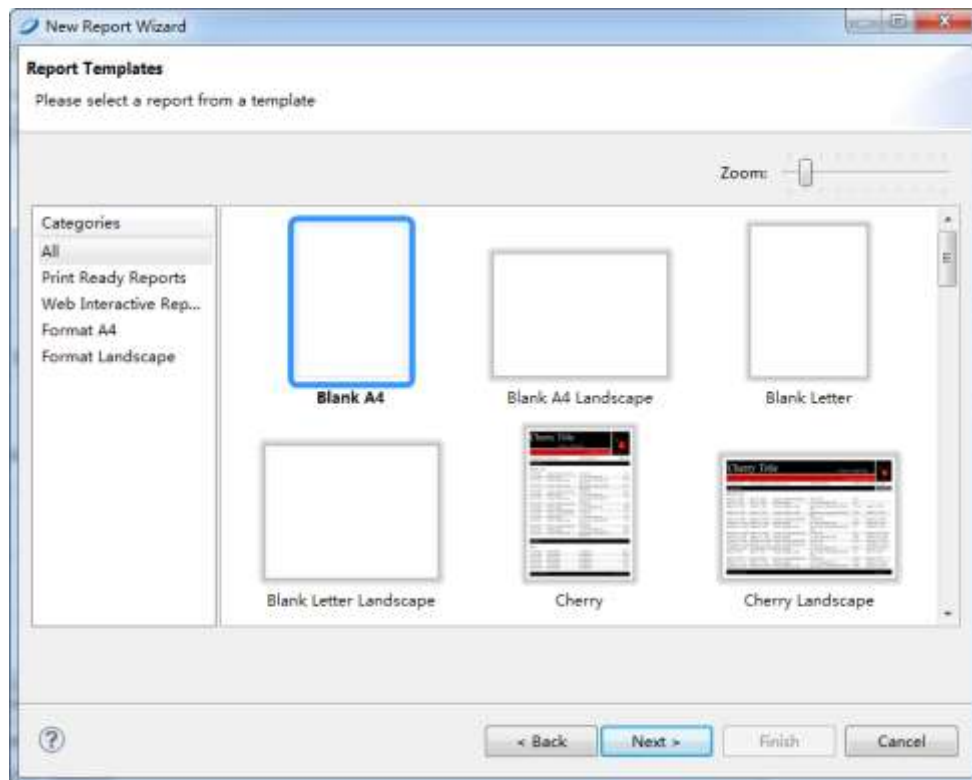
系统会自动弹出子报表选择窗口。可以选择创建一个新报表，还是使用一个

已有的报表作为子报表。

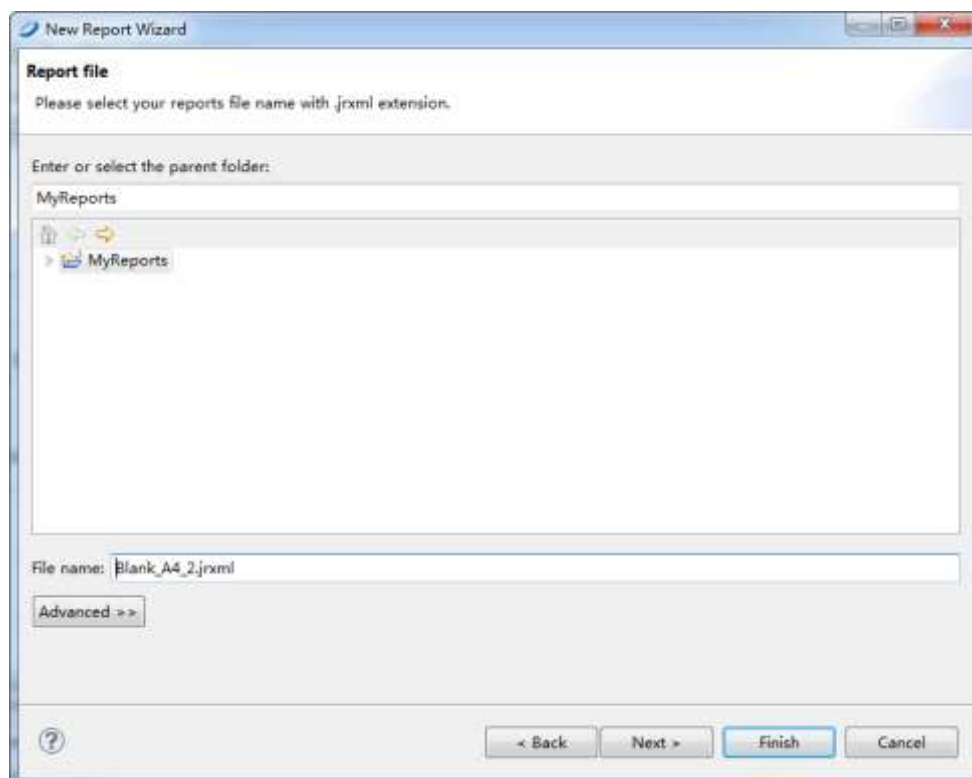


选择“Create a new report”，可以立即制作新的子报表；如果选择“Select an existing report”，则可以调用已有的报表作为子报表；如果选择“Just create the subreport element”，系统会生成一个子报表区，可以在之后挂接需要的子报表。

选择“Create a new report”的，点击“Next”会出现“选择纸型”界面。本例中选择纵向的 A4 纸，即“Blank A4”。

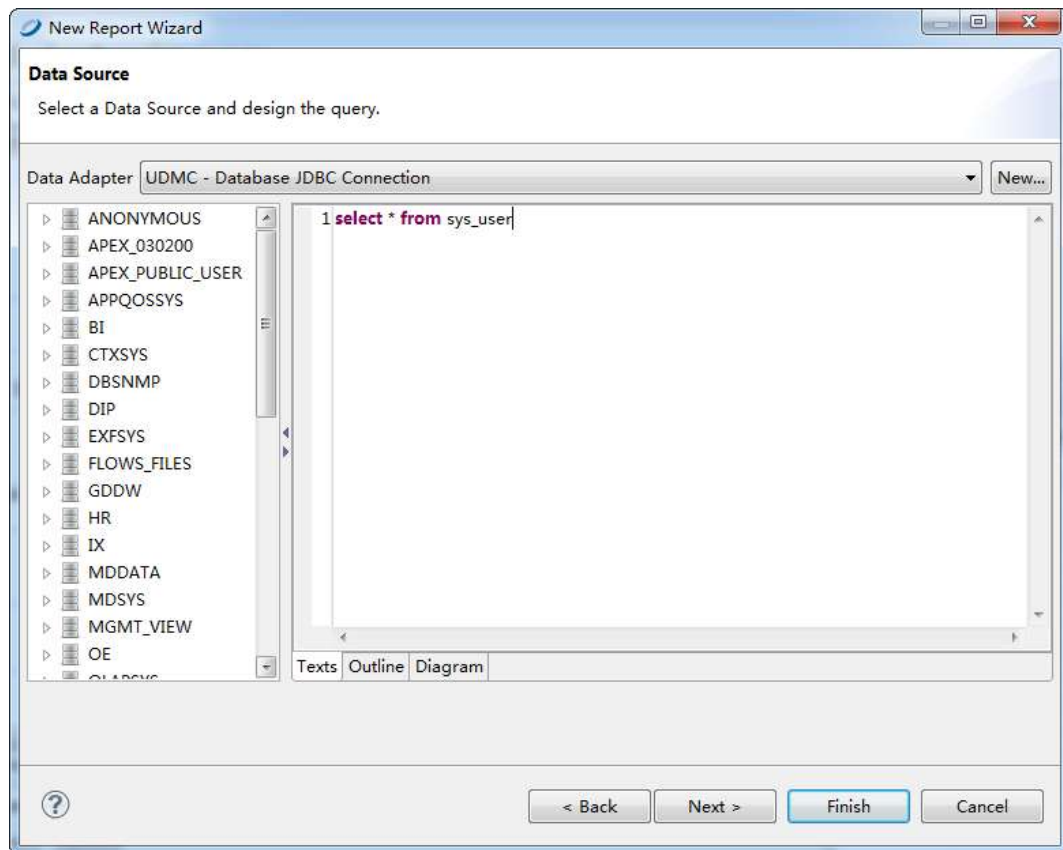


选择完毕，点击“Next”，系统显示文件存储设置页面，在此处设置子报表文件存放在何处。

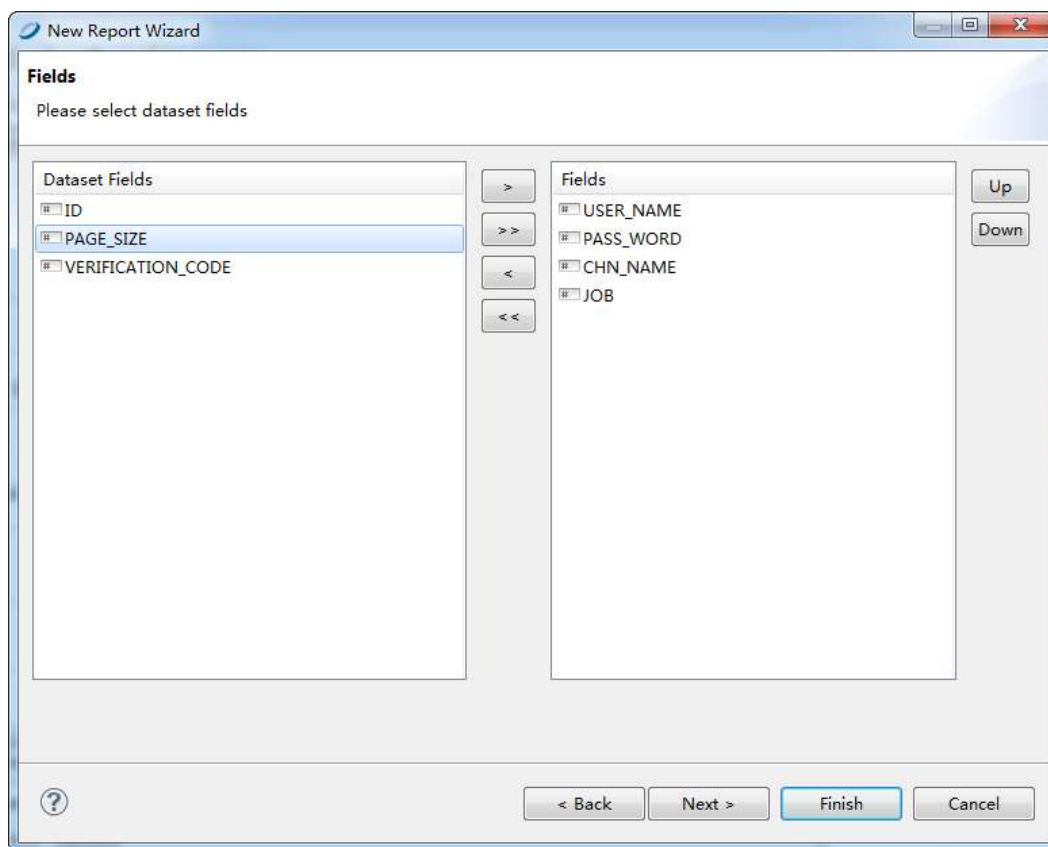


选择完毕，点击“Next”，系统显示“Data Source”界面。需要在此处选择数

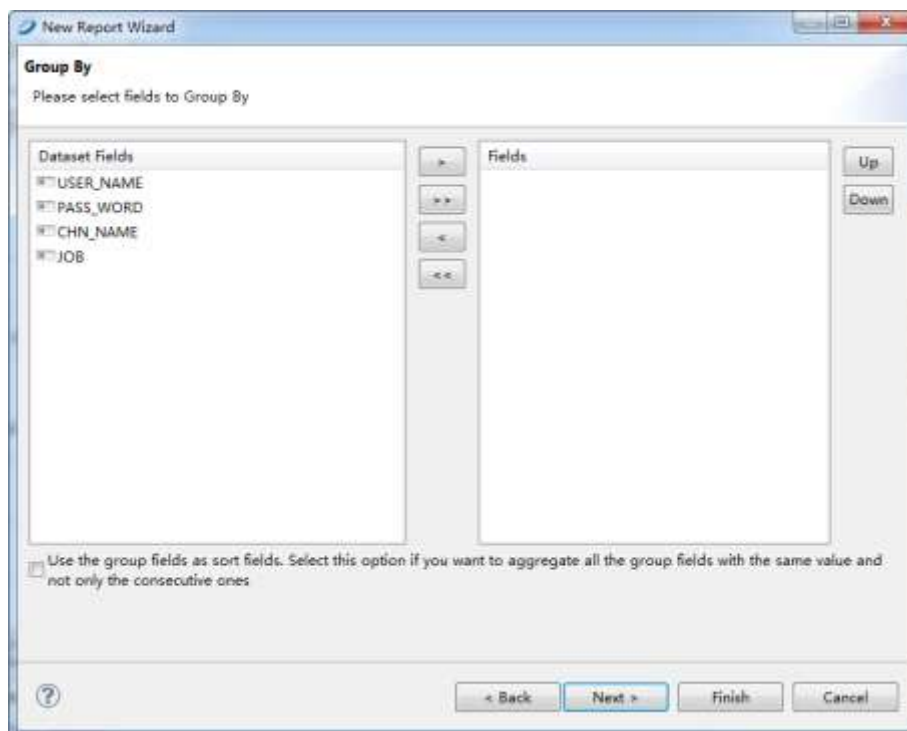
据连接，并输入 SQL 语句，来控制子报表数据的查询范围。



点击“Next”按钮，系统显示“Fields”界面。可以在此处选择子报表中显示的字段。

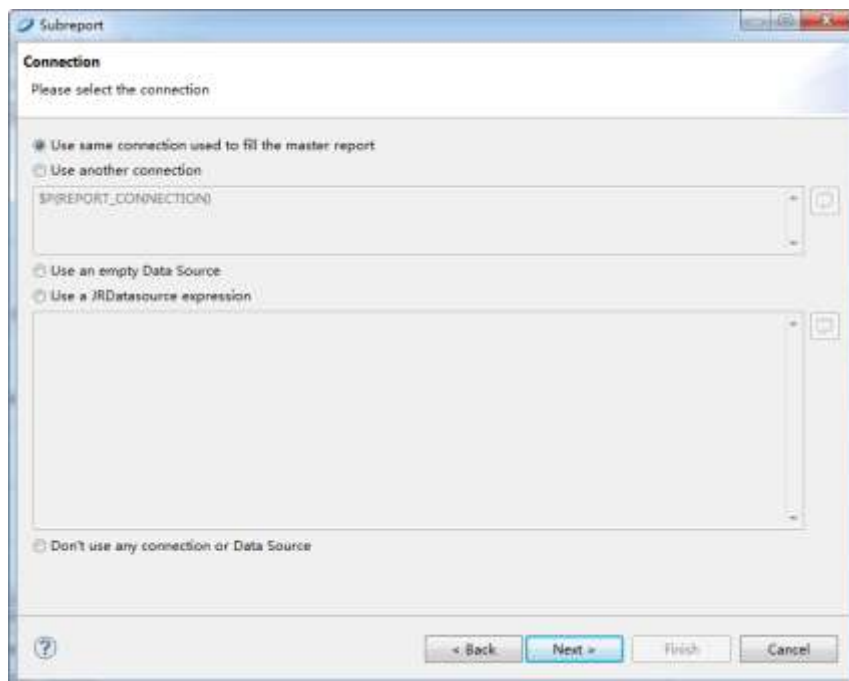


点击“Next”按钮，系统显示“Group By”界面。可以在此处设置子表的分组字段。



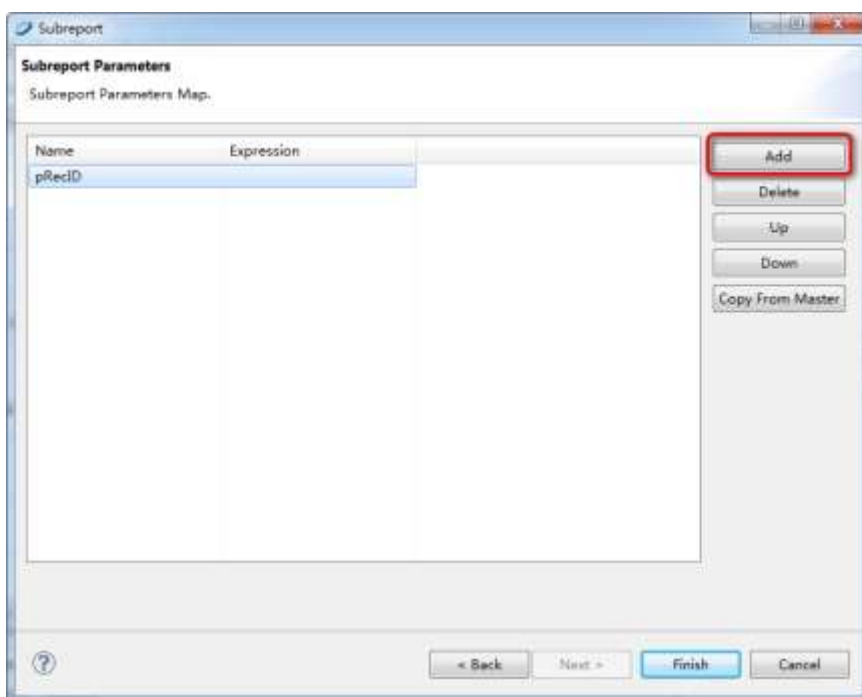
点击“Next”按钮，系统显示“Connection”界面。默认使用与父报表相同

的数据连接，也可以选择使用其他指定的数据连接。



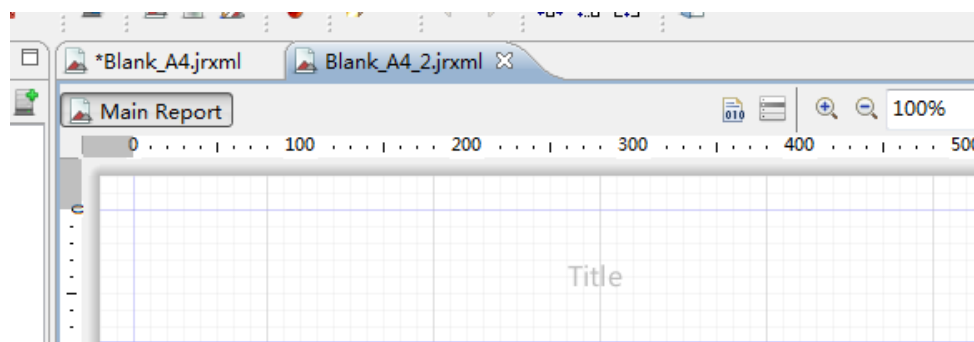
点击“Next”按钮，系统显示“Subreport Parameters”界面，在此处可以设置从父表需要传入子表的变量，一般都是关联字段的值。

子报表如果想与父报表联动工作，需要通过参数传递，将父报表中的值传递给子报表。



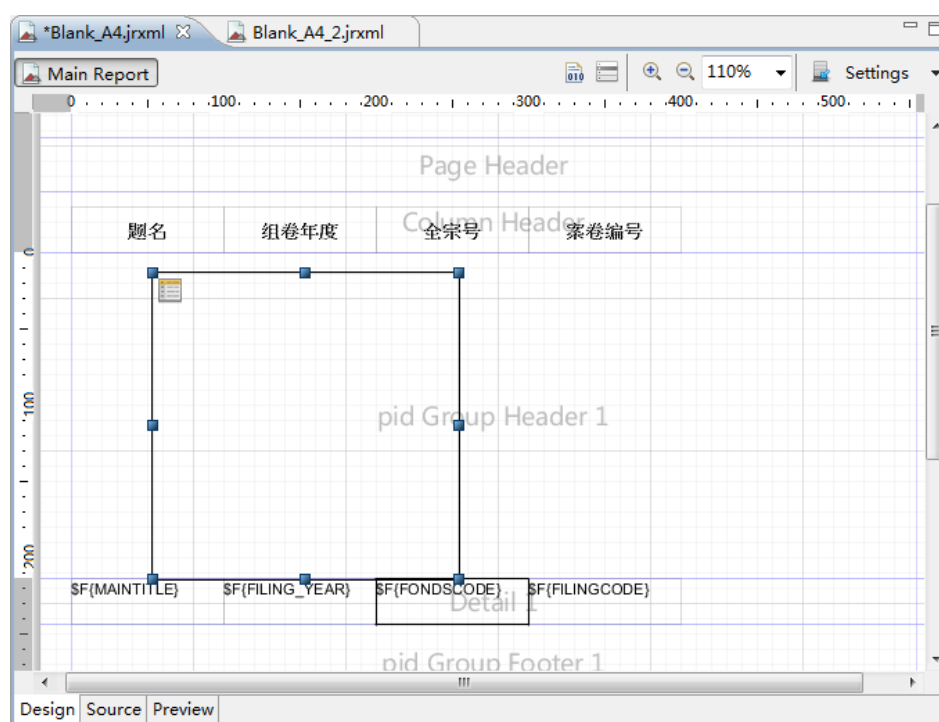
在“Name”中输入参数的名字，在“Expression”中输入参数的公式。公式可以使用系统的公式生成器自动生成。设定好后。点击“Finish”按钮即可。

系统自动显示子报表界面，如下所示。



可以在此处修改子报表的格式。修改完毕后，保存即可。

设置完毕后，主报表显示如下。



1.2.8.3. 子报表自适应行高

在准备放置子报表的区域，先放置一个 Field 域，删除关联的字段值，即放置一个空白域。设置此域为自动扩展行高。设置如下。

其中，“Stretch Type”选项，选择“Relative to Band Height”。

子报表的 field 域设置为自动扩展行高。

父报表中的子报表域设置为取当前行最高值，即“Stretch Type”选项，选择“Relative to Tallest Object”。如此设置即可。

1.3. 平台 JasperReport 引擎整合

1.3.1. 加入依赖包

```
<dependency>
  <groupId>net.sf.jasperreports</groupId>
  <artifactId>jasperreports</artifactId>
  <version>6.2.0</version>
  <exclusions>
    <exclusion>
      <artifactId>olap4j</artifactId>
      <groupId>org.olap4j</groupId>
    </exclusion>
    <exclusion>
      <artifactId>itext</artifactId>
      <groupId>com.lowagie</groupId>
    </exclusion>
  </exclusions>
</dependency>
```

另外，生成 PDF 文件，需要导出时引用以下的依赖包：

```
<dependency>
  <groupId>com.lowagie</groupId>
  <artifactId>itext</artifactId>
  <version>2.1.7</version>
</dependency>
<dependency>
  <groupId>com.lowagie</groupId>
  <artifactId>itextasian</artifactId>
  <version>2.1.7</version>
</dependency>
```

1.3.2. 实现类

报表 的展示及参数的解析的实现均由以下类进行实现：

com.redxun.sys.core.controller.SysReportController

```

SysReportController 6912 16-11-17 下午7:14 cjk
├── sysReportManager : SysReportManager
├── sysDatasourceManager : SysDatasourceManager
├── sysTreeManager : SysTreeManager
├── insNewsManager : InsNewsManager
├── convertService : JasperJrxmlConvertService
├── getQueryFilter(HttpServletRequest) : QueryFilter
├── del(HttpServletRequest, HttpServletResponse) : JsonResult
├── get(HttpServletRequest, HttpServletResponse) : ModelAndView
├── edit(HttpServletRequest, HttpServletResponse) : ModelAndView
├── listBlank(HttpServletRequest, HttpServletResponse) : List<ModelAndView>
├── preview(HttpServletRequest, HttpServletResponse) : ModelAndView
├── designerDlg(HttpServletRequest, HttpServletResponse) : ModelAndView
├── show(HttpServletRequest, HttpServletResponse) : void
├── getBaseManager() : BaseManager
├── export(HttpServletRequest, HttpServletResponse) : void
├── publish(HttpServletRequest, HttpServletResponse) : ModelAndView

```

报表模板的处理及动态参数的解析则由以下类进行处理：

com.redxun.sys.core.service.JasperJrxmlConvertService

1.3.3. 报表的展示解析

使用 Jaspersoft Studio 创建完报表模板后，我们就可以在 Java 中编译解析这个模板了，如果是 jrxml 格式的模板，需要先解析成 jasper 格式（或者说 JasperReport 格式）：

```

input = new FileInputStream(new File(filePath)); //filePath是jrxml文件的路径
JasperDesign design = JRXmlLoader.load(input); //读取jrxml文件
JasperReport report = JasperCompileManager.compileReport(design); //将jrxml文件编译成jasper文件，或者说JasperReport类
JasperPrint print = JasperFillManager.fillReport(report, parameters, connection); //编译成print格式，这种格式可以导出和展示
connection.close();

```

如果是 jasper 格式的模板，直接编译成 print 格式即可：

```

input = new FileInputStream(new File(filePath)); //filePath是jasper文件的路径
JasperPrint print = JasperFillManager.fillReport(input, parameters, connection);

```

得到 print 格式后，就可以展示或者导出，在 html 上的展示如下：

```

response.setContentType("text/html;charset=UTF-8");
// 获得输出流, 这里不能这样response.getOutputStream()
PrintWriter printWriter = response.getWriter();
// 创建HtmlExporter对象
HtmlExporter htmlExporter = new HtmlExporter();
request.getSession().setAttribute(ImageServlet.DEFAULT_JASPER_PRINT_SESSION_ATTRIBUTE, print); // 显示图片需要的设置
SimpleHtmlExporterOutput out = new SimpleHtmlExporterOutput(printWriter);
out.setImageHandler(new WebHtmlResourceHandler(request.getContextPath() + "/image?image={0}")); // 显示图片需要的设置
SimpleHtmlReportConfiguration configuration = new SimpleHtmlReportConfiguration();
configuration.setPageIndex(pageIndex); // 显示特定的某一页, 如果没有指定将一页显示所有页面
htmlExporter.setConfiguration(configuration);
// configuration.setRemoveEmptySpaceBetweenRows(true);
htmlExporter.setExporterInput(new SimpleExporterInput(print));
htmlExporter.setExporterOutput(out);

htmlExporter.exportReport();

```

上文中的 Parameter 是一个 Map 类变量，用于传递变量到报表中

```
SELECT * FROM jsaas_test.ins_news where `SUBJECT` = ${subject}
```

`${subject}`是一个参数，需要从 Java 中传递过去赋值，如果没有获取到正确的值，则报表会出错，所以需要用一个 Map 变量传递过去。如下：

```

Map<String, Object> parameters = new HashMap<String, Object>(); // 该Map用于传参数到报表模板
parameters.put("subject", "xxx主题名字");

```

1.3.4. 报表的导出

报表的导出跟展示在获得 print 之前的操作是一样的，不同的是，在导出时需要设定 `response.setHeader` 才能为下载模式。

同时，需要根据不同的导出类型，调用不同的 jasper 解析类，并且设置 input，output，ContentType，Configuration（可设可不设）即可。

```

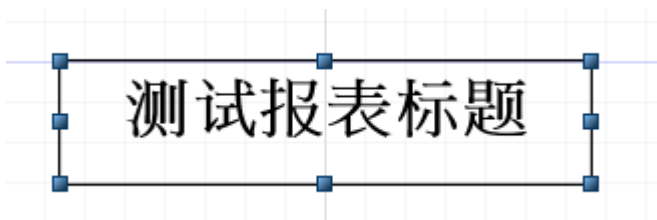
JRAbstractExporter exporter = null;
HttpServletResponse response = (HttpServletResponse) map.get("response");
OutputStream out = response.getOutputStream();
if ("pdf".equalsIgnoreCase(ext)) {
    exporter = new JRPdfExporter();
    exporter.setExporterInput(new SimpleExporterInput(jasperPrint));
    exporter.setExporterOutput(new SimpleOutputStreamExporterOutput(out));
    SimplePdfExporterConfiguration configuration = new SimplePdfExporterConfiguration();
    configuration.setCreatingBatchModeBookmarks(true);
    exporter.setConfiguration(configuration);
    response.setContentType("application/pdf");
} else if ("xls".equalsIgnoreCase(ext)) { //不分页显示
    exporter = new JRXlsExporter();
    response.setContentType("application/xls");
    exporter.setExporterInput(new SimpleExporterInput(jasperPrint));
    exporter.setExporterOutput(new SimpleOutputStreamExporterOutput(out));
    SimpleXlsReportConfiguration configuration = new SimpleXlsReportConfiguration();
    configuration.setOnePagePerSheet(false);
    exporter.setConfiguration(configuration);
} else if ("pagexls".equalsIgnoreCase(ext)) { //分页显示
    exporter = new JRXlsExporter();
    response.setContentType("application/xls");
    exporter.setExporterInput(new SimpleExporterInput(jasperPrint));
    exporter.setExporterOutput(new SimpleOutputStreamExporterOutput(out));
    SimpleXlsReportConfiguration configuration = new SimpleXlsReportConfiguration();
    configuration.setOnePagePerSheet(true);
    exporter.setConfiguration(configuration);
} else if ("doc".equalsIgnoreCase(ext)) {
    exporter = new JRDocxExporter();
    response.setContentType("application/vnd.openxmlformats-officedocument.wordprocessingml.document");
    exporter.setExporterInput(new SimpleExporterInput(jasperPrint));
    exporter.setExporterOutput(new SimpleOutputStreamExporterOutput(out));
}

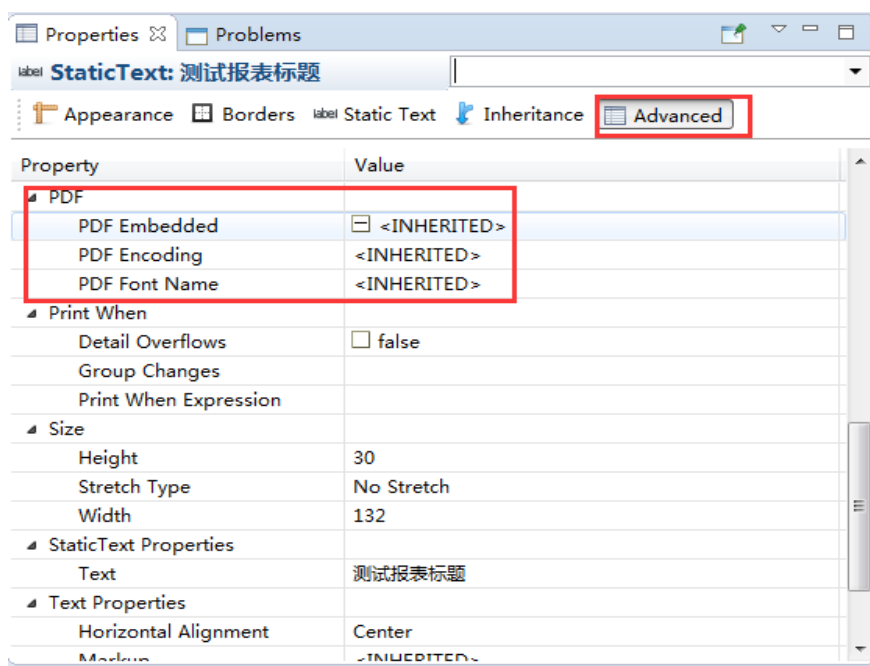
try {
    exporter.exportReport();
}

```

1.3.5. 中文显示

Jasper 报表在导出 pdf 时，中文不显示，这是因为在用 Jaspersoft Studio 创建报表模板的时候，需要设置 pdf 的输出字体以及格式。如下例：我们需要更改这个 text 的 Advanced





具体设置如下：

- (1) PDF Embedded : True
- (2) PDF Encoding : UniGB-UCS2-H (Chinese Simplified)
- (3) PDF Font Name : STSong-Light

PDF	
PDF Embedded	<input checked="" type="checkbox"/> true
PDF Encoding	UniGB-UCS2-H (Chinese Simplified)
PDF Font Name	STSong-Light

同时，在 java 中需要引入相关 jar 包：iText.jar 和 iTextAsian.jar

1.3.6. html 上图片的显示

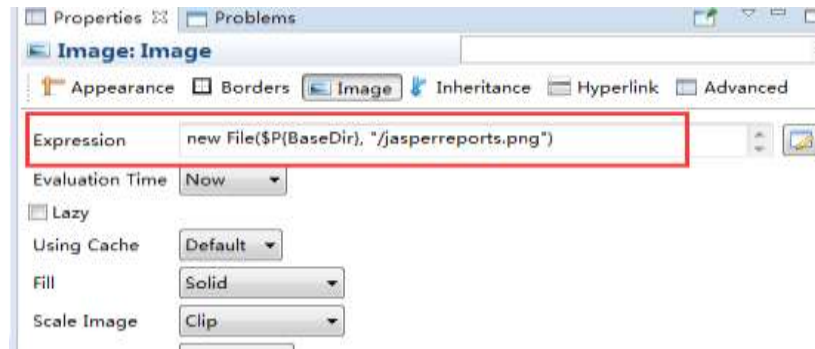
在html页面上图片的显示需要给image的地址 并且在web.xml中配好servlet

```
<!-- Jasper Begin -->
<servlet>
  <servlet-name>ImageServlet</servlet-name>
  <servlet-class>net.sf.jasperreports.j2ee.servlets.ImageServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>ImageServlet</servlet-name>
  <url-pattern>/image</url-pattern>
</servlet-mapping>
<!-- Jasper End -->
```

```
// 设置HtmlExporter
HtmlExporter htmlExporter = new HtmlExporter();
request.getSession().setAttribute(ImageServlet.DEFAULT_JASPER_PRINT_SESSION_ATTRIBUTE, print);
SimpleHtmlExporterOutput out = new SimpleHtmlExporterOutput(printWriter);
out.setImageHandler(new WebHtmlResourceHandler(request.getContextPath() + "/image?image={0}"));
```

其中setImageHandler中的地址与web.xml中配好的servlet地址要一样，此处为/image

然后还需要在模版中拖入一个image，并把其Expression修改为如下：



然后在Java中传入BaseDir，即你的图片地址：

```
String picPath = picPath.replaceAll("\\\\", "/"); // 替换图片目录
parameters.put("BaseDir", new File(WebAppUtil.getAppAbsolutePath() + "/reports/" + picPath)); // 传入图片目录
```

此时即可显示报表包含的图片了。

1.4. 在平台中使用报表模板

1.4.1. 报表系统的基本界面

报表分类：点击左边报表分类，右边将显示该报表分类下的所有报表，右键分类

目录可以对目录进行增删改操作

上传报表：点击可以上传新的报表模板

初始化参数：点击可以编辑该报表的初始化参数，一般有需要传参数到报表模板里才需要初始化参数。

预览：点击可以预览和导出报表

1.4.2. 上传报表模板

点击【上传报表】，进入上传报表页面：

上传：点击上传报表模板

上传帮助：点击后在最底部会显示上传帮助信息

数据源：点击选择这个报表所连接的数据源（MySQL..等）

标题：该报表的标题

表示 Key：一个唯一值，用于区分报表

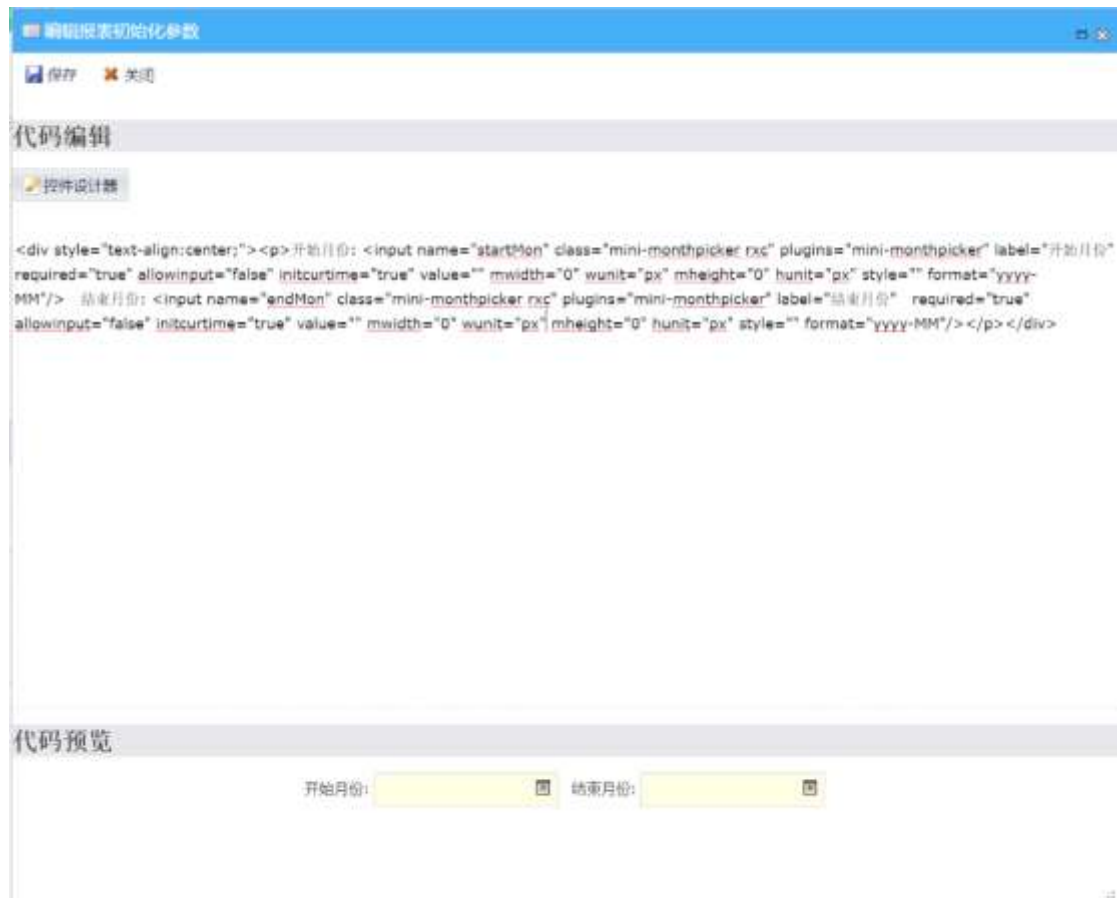
描述：对该报表的一段文字描述

上传模板文件：点击上传模板文件，目前支持 jrxml 格式，jasper 格式以及 zip 格式

报表解析引擎：选择上传的报表模板文件的类型

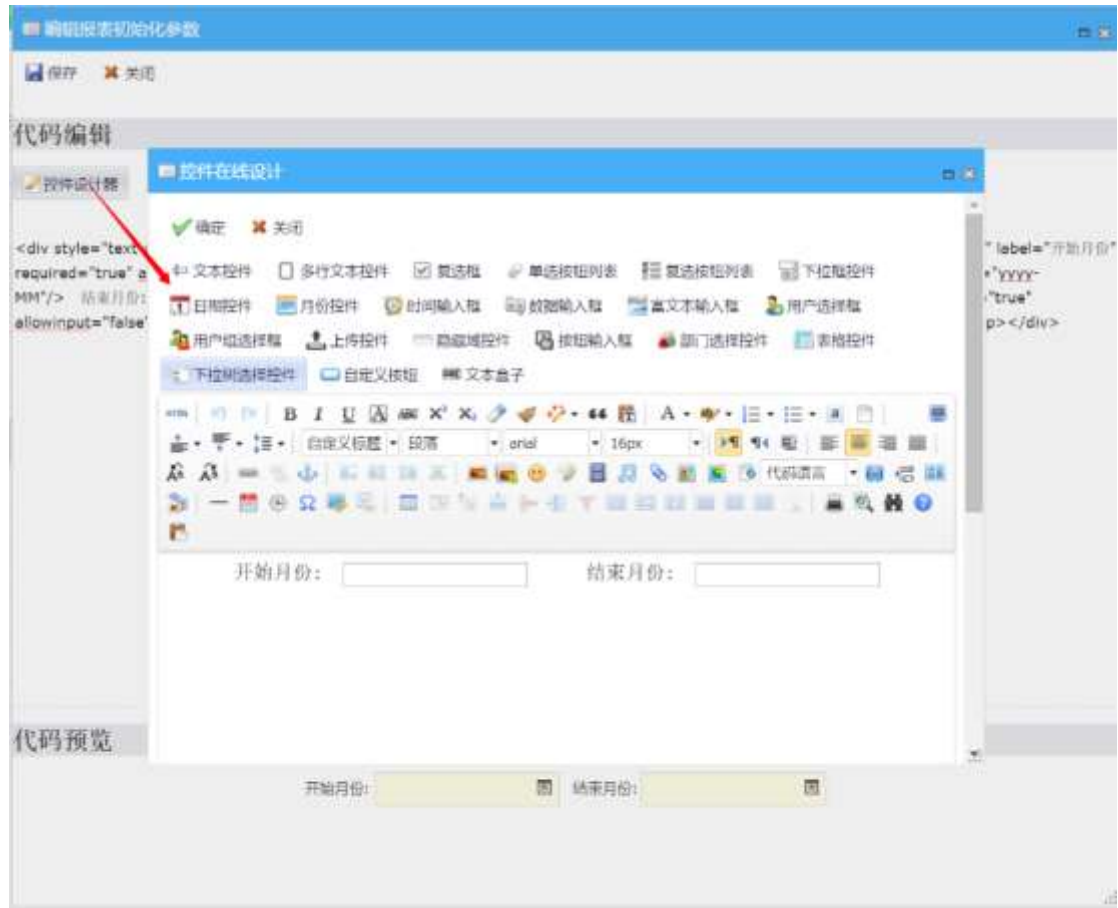
参数配置：报表的初始化参数代码，一般有参数传递的时候才需要，在基本页面中修改。

1.4.3. 初始化参数界面



控件设计器：点击打开控件设计器，其可以可视化的编辑控件，然后返回控件的代码，显示在按钮下面，即第二个红框，然后设计控件的显示预览即第三个红框。

保存：点击保存控件代码



1.4.4. 报表预览和导出

点击【预览】按钮，可以看到如下页面：

2016-08月中航关业单位离休干部等人员医疗费结算情况表

页码: 第1页 (共1页)


编码	单位名称	主要结算		医疗费开支情况			结算比例计算结果				
		人数	医疗结算	门诊	住院	合计	超支	结余	单位应付	单位应收	单位应付金额
002	离休干部办公室	2	600	22.00	418.00	440.00	93.00	0.00	25.00	8.00	25.00
003	中航关业单位	2	600	626.00	24.00	650.00	93.00	0.00	25.00	8.00	25.00
本部总计		1,300	1,300	626.00	442.00	1,300.00	190.00	0.00	40.00	0.00	40.00
总计		1,300	1,300	626.00	442.00	1,300.00	190.00	0.00	40.00	0.00	40.00

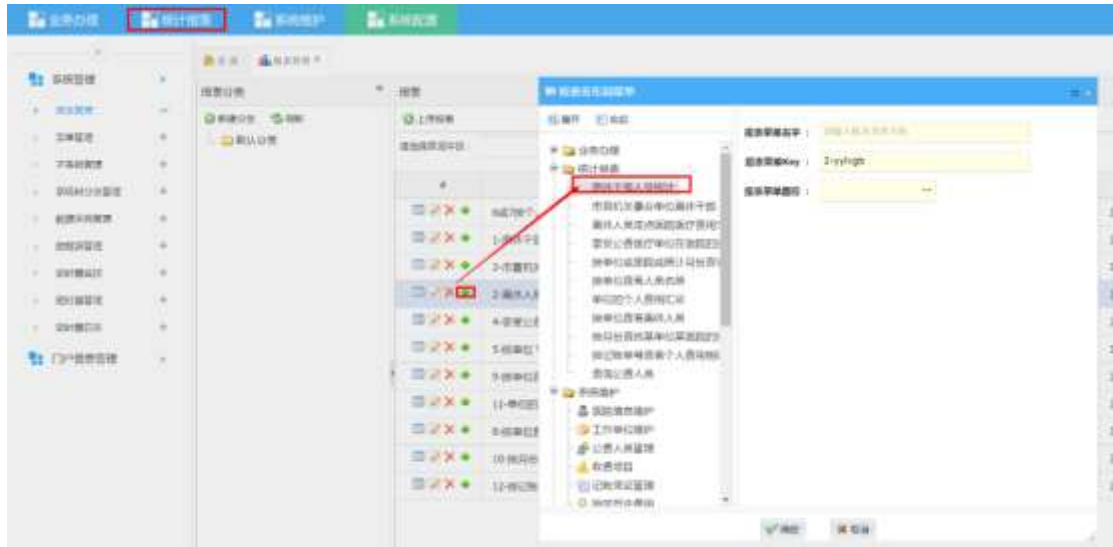
页数工具栏：显示报表页数和页数跳转功能

导出：将此报表导出成各种格式，如 pdf，excel，doc

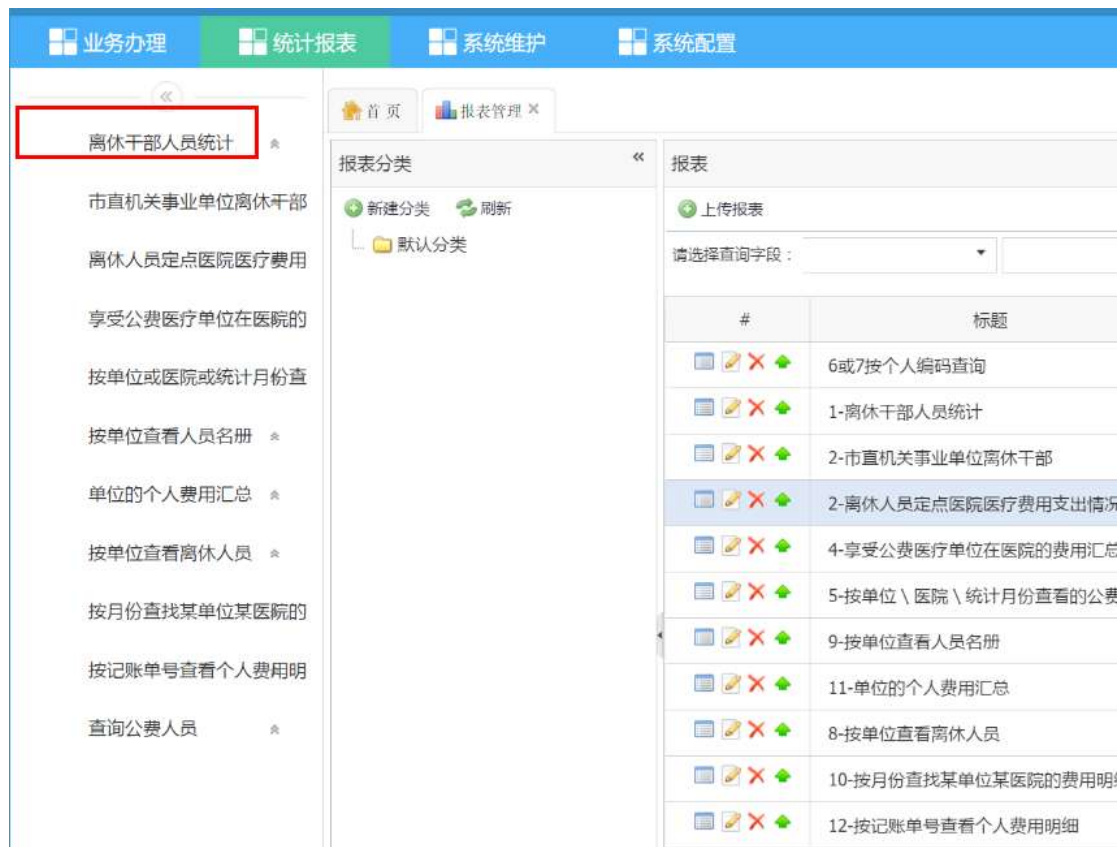
生成报表：有参数选择和传递时才会出现的按钮，和下一行的选择框配合使用，当选择完参数后，点击【生成报表】就会显示报表内容。

1.4.5. 添加报表菜单

进入系统管理模块下的报表管理模块中，在报表管理列表中，点击需要添加到菜单中的报表行的  的按钮，则弹出如下菜单：



加到完成后，可以在菜单中看到如下菜单：



或进入菜单界面中，手工添加如下菜单：

在菜单管理中配置报表地址：`/sys/core/sysReport/preview.do?key=demo`

其中 key 后面的值根据报表不同而改变