**目录**

[1. 概述 2](#_Toc24830373)

[1.1. OLAP与OLTP 2](#_Toc24830374)

[1.2. OLAP基本概念 3](#_Toc24830375)

[1.3. 典型维度说明 4](#_Toc24830376)

[1.3.1. 时间维度 4](#_Toc24830377)

[1.3.2. 设备维度 5](#_Toc24830378)

[1.3.3. 地域维度 5](#_Toc24830379)

[1.4. Mordrian Schema 5](#_Toc24830380)

[1.4.1. Cube 5](#_Toc24830381)

[1.4.2. Dimension 6](#_Toc24830382)

[1.4.3. Hierarchy 6](#_Toc24830383)

[1.4.4. Level 7](#_Toc24830384)

[1.4.5. Join 8](#_Toc24830385)

[1.4.6. Measure 8](#_Toc24830386)

[2. BI设计 10](#_Toc24830387)

[2.1. 初步方案 10](#_Toc24830388)

[2.1.1. 前言 10](#_Toc24830389)

[2.1.2. 分析 10](#_Toc24830390)

[2.1.3. 思路 11](#_Toc24830391)

[2.2. 数据表设计 11](#_Toc24830392)

[2.3. page\_report字段设计 12](#_Toc24830393)

[2.4. 创建合理的视图（视图命名规范v\_） 12](#_Toc24830394)

[2.4.1. 通过BI和视图创建报表 13](#_Toc24830395)

[2.4.2. 经济类型 13](#_Toc24830396)

[2.4.3. 生产现状 13](#_Toc24830397)

[2.4.4. 开采方式 14](#_Toc24830398)

[2.4.5. 矿山规模 14](#_Toc24830399)

[2.4.6. 矿类统计 14](#_Toc24830400)

[2.5. BI钻取的实现 15](#_Toc24830401)

[2.5.1. 思路引导 15](#_Toc24830402)

[2.5.2. 示示例 15](#_Toc24830403)

[3. BI使用说明 17](#_Toc24830404)

[3.1. 维表管理 18](#_Toc24830405)

[3.2. 主题管理 18](#_Toc24830406)

[3.2.1. 主题管理（事实表） 18](#_Toc24830407)

[3.2.2. 主题字段管理 19](#_Toc24830408)

[3.3. 报表管理 20](#_Toc24830409)

[3.4. 页面管理 21](#_Toc24830410)

[3.5. 屏幕管理 21](#_Toc24830411)

[3.6. 展示页面 22](#_Toc24830412)

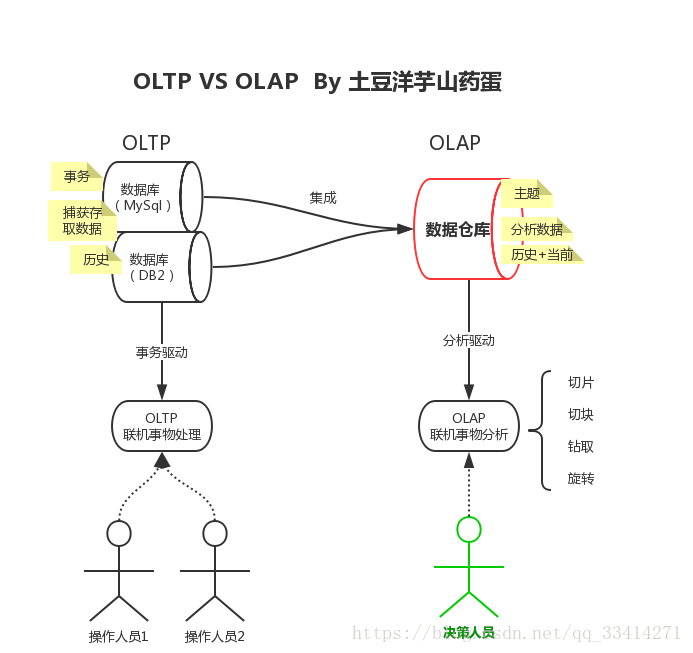
[3.6.1. 监控大屏展示页面 22](#_Toc24830413)

[3.6.2. 流布局展示页面 22](#_Toc24830414)

[3.6.3. 多维表格展示页面 23](#_Toc24830415)

[3.6.4. 多维图表展示页面 23](#_Toc24830416)

1. 概述
   1. OLAP与OLTP



* 1. OLAP基本概念

1） 维

维(Dimension)：人们观察事物的视角，如时间、地理位置、年龄和性别等，是单一角度概念。维的层次(Lever of Dimension)：表示维度概念基础上进一步的细分，如时间可以细分为年、季度、月三个层次。维成员(Member of Dimension)：表示维不可再细分的原子取值，如时间维的成员可以是2019年1月10日。度量(Measure)：表示在这个维成员上的取值。

除了维的基本概念，还有多维分析的分析操作。

2）操作

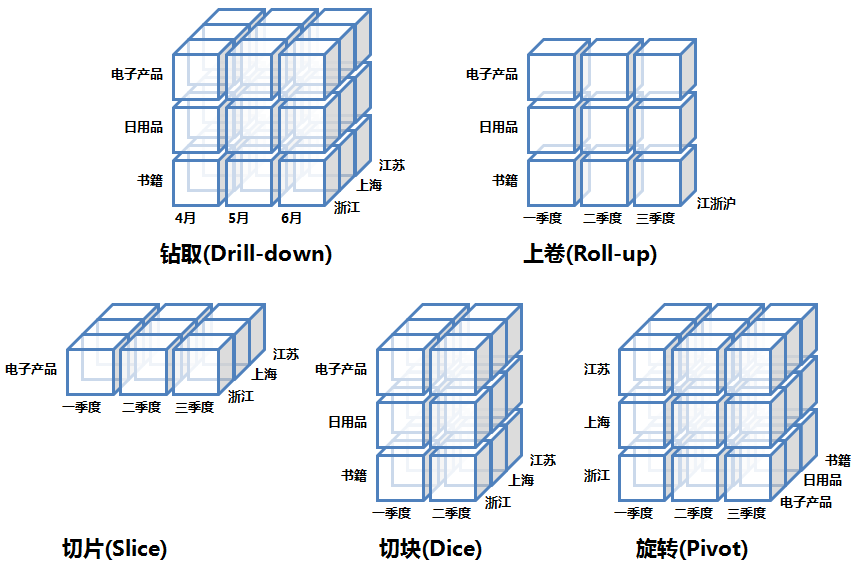
下探(Drill down)：维度是有层次的，下探表示进入维度的下一层，将汇总数据拆分到下一层所在细节数据信息，如下图从第二季度下探到看4、5、6月的明细数据。

上钻(Drill up)： 下探的反向操作，回到更高汇聚层的汇总数据。

切片(Slice)：切片可以理解成把立体按某一个维度进行切分，就可以看两维数据，如图中按电子产品切分，看到的是时间和地理位置关系的二维数据。

切块(Dice)：相对于切片是按一个点切分，切块就是按一个范围(区间)来做切分。

旋转(Pivot)：维的行列位置交换，换一个视角分析数据。



* 1. 典型维度说明
     1. 时间维度

1) 说明:该维度记录了每天的时间,粒度最高精确到日,并可分周,月,年等粒度.

2) 对应表:tbl\_dimdate

3) 对应过程: pro\_supportdw\_dimdate

4) 是否公用:是

5) 说明:此维度可建Hierarchy(层),见下图:



* + 1. 设备维度

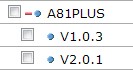
1) 说明:该维度记录了的设备信息. 并可分品牌,机型等粒度.

2) 对应表: tbl\_dimdevice

3) 对应过程: pro\_supportdw\_dimdevice

4) 是否公用:否

5) 说明:此维度可建Hierarchy(层),见下图:



* + 1. 地域维度

1) 说明:该维度记录了的地域信息. 并可分国家,省,区等粒度.

2) 对应表: tbl\_dimgeography

3) 对应过程: 无,必要时手动加入地区数据

4) 是否公用:否

5) 说明:此维度无Hierarchy(层) ,见下图:

* 1. Mordrian Schema

Schema 定义了一个多维数据库。包含了一个逻辑模型，而这个逻辑模型的目的是为了书写 MDX 语言的查询语句。这个逻辑模型实际上提供了这几个概念： Cubes （立方体）、维度（ Dimensions ）、度量 (Measure)、层次（ Hierarchies ）、级别（ Levels ）、和成员（ Members ）。而一个 schema 文件就是编辑这个 schema 的一个xml 文件。在这个文件中形成逻辑模型和数据库物理模型的对应。

* + 1. Cube

一个 Cube 是一系列维度 (Dimension) 和度量 (Measure) 的集合区域。在 Cube 中， Dimension 和 Measure 的共同地方就是共用一个事实表。 Cube 中的有以下几个属性：

|  |  |
| --- | --- |
| 属性名 | 含义 |
| name | Cube 的名字 |
| caption | 标题 , 在表示层显示的 |
| cache | 是否对 Cube 对应的实表用 mondrian 进行存储 , 默认为 true |
| enabled | 是布尔型的 , 如果是被激活 ,Cubes 就执行 , 否则就不予理睬，默认为 true |

Cube 里面有一个全局的标签定义了所用的事实表的表名

* + 1. Dimension

他是一个层次（ Hierarchies ）的集合 , 维度一般有其相对应的维度表 . 他的组成是由层次（ Hierarchies ）而层次（ Hierarchies ）又是有级别（ Level ）组成 . 其属性如下：

|  |  |
| --- | --- |
| 属性名 | 含义 |
| name | Dimension 的名称 |
| type | 类型，有两个可选的类型： StandarDimension 和 TimeDimension ，默认为 StandardDimension |
| caption | 标题 , 在表示层显示的 |
| UsagePrefix | 加前缀 , 消除歧义 |
| foreignKey | 外键，对应事实表中的一个列，它通过 <Hierarchy> 元素中的主键属性连接起来。 |

* + 1. Hierarchy

你一定要指定其中的各种关系 , 如果没有指定 , 就默认 Hierarchy 里面装的是来自立方体中的真实表 . 属性如下：

|  |  |
| --- | --- |
| 属性名 | 含义 |
| name | Hierarchy 的名称，该值可以为空，为空时表示 Hirearchy 的名字和 Dimension 的名字相同。当一个 Dimension 有多个 Hierarchy 时，注意 name 值要唯一。 |
| hasAll | 布尔型的 , 决定是否包含全部的成员 member |
| allMemberName | 所有成员的名字 , 也就是总的标题 , 例如： allMemberName= “全部产品” |
| allLevelName | 所有级别的名字，它会覆盖其下所有的 Member 的 name 和所有的 Level 的 name 属性的值。 |
| allMemberCaption | 例如 : allMemberCaption= “全部产品”这个是在表示层显示的内容 |
| PrimaryKey | 通过主键来确定成员，该主键指的是成员表中的主键，该主键同时要与 Dimension 里设置的 foreignKey 属性对应的字段形成外键对应关系 |
| primaryKeyTable | 如果成员表不只一个，而是多个表通过 join 关系形成的，那么就要通过这个属性来指明 join 的这些表中，哪一个与 Dimension 里设置的 foreignKey 属性形成外键关系。通过该属性来指明主表 |
| caption | 标题 , 在表示层显示的 |
| defaultMember |  |
| memberReaderClass | 设定一个成员读取器，默认情况下 Hierarchy 都是从关系型数据库里读取的，如果你的数据不在 RDBMS 里面的话，你可以通过自定义一个 member reader 来表现一个 Hierarchy 。 |

* + 1. Level

级别 , 他是组成 Hierarchy 的部分。属性很多，并且是 schema 编写的关键，使用它可以构成一个结构树， Level的先后顺序决定了 Level 在这棵树上的的位置，最顶层的 Level 位于树的第一级，依次类推。 Level 的属性如下：

|  |  |
| --- | --- |
| 属性名 | 含义 |
| name | 名称 |
| table | 该 Level 要使用的表名 |
| column | 用上面指定的表中某一列作为该 Level 的关键字 |
| nameColumn | 用来显示的时候使用，如果不定义，那么就采用上面的 column 的值来进行显示。 |
| oridinalColumn | 定义该 Level 上的成员的显示顺序，如果不指定，那么采用 column 的值。 |
| parentColumn | 在一个有父 - 子关系的 Hierarchy 当中，当前 Level 引用的是其父成员的列名。好比是一张部门表，在一张表里表现部门的上下级关系，一个是主键，肯定还有一个字段为连接到该主键的外键的列名，这里的 parentColumn 指的就是这个列名。 |
| nullParentValue | 如果当前的 Level 是有上下级关系（设置了 parentColumn 属性），如果该 Level 又处于顶级，我们需要将顶级的数据取出来，这里指的是位于顶级的父成员的值，有些数据库不支持 null, 那么也可以使用 ’0’ 或 ’-1’ 等，这就表示顶级的成员的父 ID 为 ’0’ 或为 ’-1’ 。 |
| type | 数据类型，默认值为 string 。当然还可以是 Numeric 、 Integer 、 Boolean 、 Date 等。 |
| uniqueMembers | 该属性用于优化产生的 SQL ，如果你知道这个级别和其父级别交叉后的值或者是维度表中给定的级别所有的值是唯一的，那么就可以设置该值为 true ，否则为 false 。 |
| levelType | 该 Level 的类型，默认为 regular （正常的），如果你在其 Dimension 属性 type 里选择了 TimeDimension 那么这里就可以选择 TimeYears 、 TimeQuarters 、 TimeMonth 、 TimeWeeds 、 TimeDays 。 |
| hideMemberIf | 在什么时候不隐藏该成员，可选的值有三个： Never 、 IfBlankName 、 IfParentName |
| approxRowCount | 该属性可以用来提高性能，可以通过指定一个数值以减少判断级别、层次、维度基数的时间，该属性在通过使用 XMLA 连接 Mondrian 很有用处。 |
| caption | 标题 , 在表示层显示的 |
| captionColumn | 用来显示标题的列 |
| formatter | 该属性定义了 Member.getCaption() 方法返回的动作值，这里需要是一个实现了 mondrian.olap.MemberFormatter 接口的类，用来对 Caption 地值进行格式化。 |

* + 1. Join

对于一个 Hierarchy 来说，有两种方式为其指定：一种是直接通过一个 Table 标签指定；一种是通过 Join 将若干张表连接起来指定。一旦采用 Join 的话，那么就要在 Hierarchy 里的 primaryKeyTable 属性指定主表。

* + 1. Measure

Measure 就是我们要计算的数值，操作的核心。它的属性如下：

|  |  |
| --- | --- |
| 属性名 | 含义 |
| name | 名称 |
| aggregator | 要采用的计算函数 |
| column | 要计算的列名 |
| formatString | 计算结果的显示格式。 |
| visible | 是否可见 |
| datatype | 数据类型，默认为 Numeric |
| formatter | 采用类来对该 Measure 的值进行格式，具体参考 Level 的 formatter 属性。 |
| caption | 标题，用来显示时使用 |



1. BI设计
   1. 初步方案
      1. 前言

矿山信息展示系统分为两部分，一部分为客户端，另一部分为web端，由我们来做web端，web端数据可视化，采用图表的形式进行展示，使用酷炫界面风格来进行布局。需要的数据根据客户需求方的设计思想为：统计数据的展示和详细的信息展示，根据数据源的不同进行展示不同信息。

针对于北京的这个项目我认为还不是一个传统的BI，是一个比传统BI层次要高的一个系统，因为他是依赖于地图，并且还得可以详细搜索，想看什么就得通过搜索得到之后接着进行利用BI的数据可视化的方式展示出各种数据。

* + 1. 分析

客户的意愿：想看什么可以快速准确的调出要看的数据并且进行图表展示，并且界面需要详细的统计信息，概括的图标展示，并且配有信息文字统计展示信息。

数据展示的准确性、实时性、有效性必须全部体现，例如：不同状态的矿山需要用不同的颜色进行展示，问题越多的需要要警示的颜色进行提示。经过多维度的展示矿山的信息，统计矿山的变化。

* + 1. 思路

维度：区域、类型、规模、时间、数量等多维度可配置

数据来源：矿山信息采集数据库、多数据源可配置

搜索查询：多条件、自定义、查询历史

展示方式：饼图、柱状图、折线图、GIS地图

现在北京客户最主要的是进行把所有录入的数据进行展示出来，展示的过程中也需要数据进行清洗，结合北京所有录入的数据和开的几次会议来说，我认为应该这样去做：

开发通用性的BI展示，类似刚开始做的ZUP中制作的商业智能的模块一样，通过抽取数据表中的字段，来进行数据清洗。还可以通过模板的方式对客户的具体需求来进行定制化配置信息。

数据源：刚开始进行的是数据源的选择， 数据源选择之后接着是进入首页面，首页面展示一些统计信息，不同区域的不同类型的矿山总量等等，根据不同的区域、不同的类型、特征进行统计数据，进而用友好的界面图表的形式展示出来。

生成报告：通过对现有数据进行处理、挖掘，得出处理结果，最终导出报告,报告的形式有:excel\pdf\word,具体导出什么报告根据用户选择情况而定。

权限配置：每一个用户根据不同的区域来展示不同信矿山和信息。需要进行统一的规划 每一个用户的角色,做到配置一个地方,起初我是这样想的,把用户的角色所在区域ID和区域类型放入seesion,每一个用户登录之后只需要去session获取,直接获取对应数据就可以了.这样做得方式减少了原有光伏那种模式每一个页面都需要书写代码，复杂，难维护。

界面展示：两种方式：一种以地图信息展示，另一种是统计信息图表展示

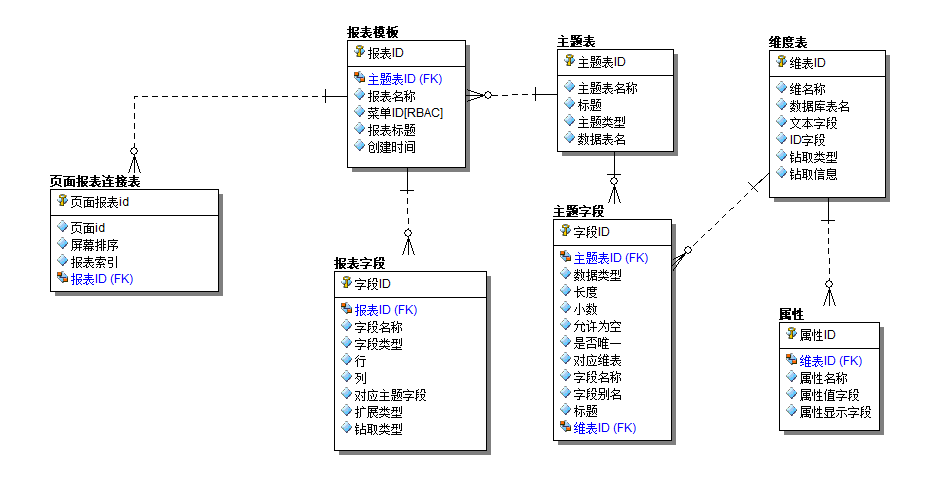
多端展示：平板电脑、手机（在电脑端设计页面都需要显示什么）、电脑。

* 1. 数据表设计

建立一张中间关系表，记录报表需要在页面中6个方块中，哪一个方块进行展示，这相当于是页面中的报表的索引位置， 当前是一个页面， 哪一个页面上的哪一个主题（主题是通过滑屏方式切换）。

先进行配置中间关系表，中间关系表配置后，我就可以打开这个页面之后，我就能把这个页面所有的报表进行查询到，页面所有报表拿到之后按照第一屏、、、第N屏排序，当用户看完第一屏（第一个主题之后），右键点击下一屏，就能看到第二屏的展示数据。

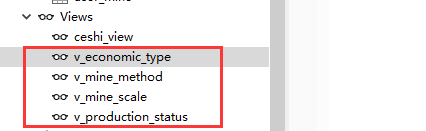
获取到一个屏幕有6块数据，这样再按照数据的索引位置进行排序展示。



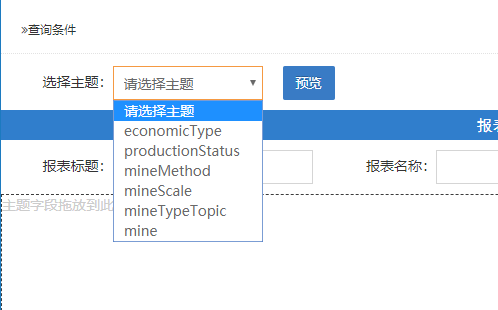
* 1. page\_report字段设计

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 字段名称 | 字段类型 | 字段说明 |
| PAGE\_REPORT\_ID | 页面报表id | Integer | 主键id |
| MENU\_ID | 菜单id | Integer | 菜单id，也就是页面标识（外键） |
| SCREEN\_SORT | 屏幕排序 | Integer | 滑屏的排序 |
| SCREEN\_SORT | 屏幕名称 | String | 每一个屏幕的名称 |
| REPORT\_INDEX | 报表索引 | Integer | 当前页中的排序 |
| REPORT\_ID | 报表id | Integer | 报表id（外键） |

* 1. 创建合理的视图（视图命名规范v\_）



* + 1. 通过BI和视图创建报表



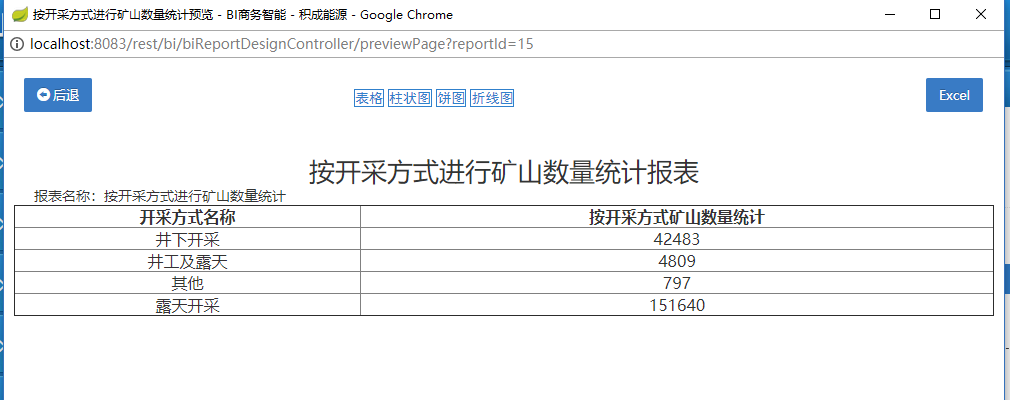
* + 1. 经济类型



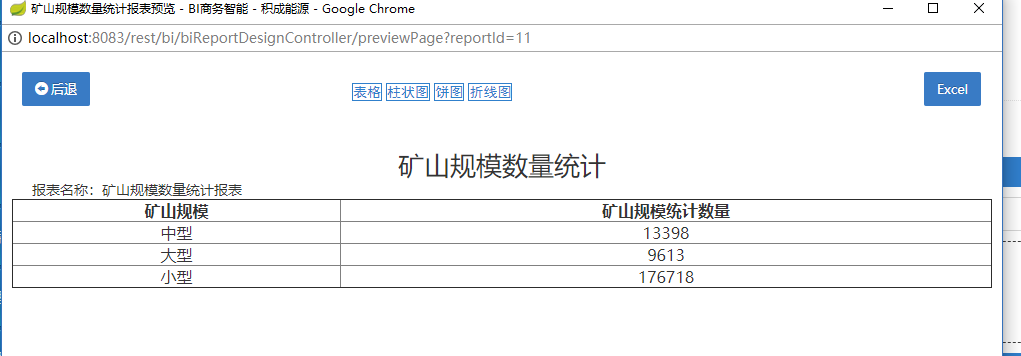
* + 1. 生产现状



* + 1. 开采方式



* + 1. 矿山规模



* + 1. 矿类统计



* 1. BI钻取的实现

目前采取了分为两步走：

一种为需要分割字段字符串进行匹配省市县，省市县作为维度也作为钻取路径的情况

另一种为不需要分割字符串进行钻取的，当省市县作为条件进行钻取，而不是维度的情况

* + 1. 思路引导

通过在BI维度字段表（bi\_dim\_attribute）中添加一个字段是否需要进行分割，每一个字端后面书写的具体数据是如果不需要分割则为0，如果需要分割字段数据的前两位就书写2，如果分割前四位就书写4，一次类推。

1）需要字符串分割

通过以上方案解决了区域采用了一个字段通过前两位代表省，前四位代表市，前六位代表区县的问题。

需要分割则在SQL语句中加入需要在SQL中加入left(需要分割字段, 分割位数)

2）不需要字符串分割

如果不需要字符串分割则为0，当为零的时候就不需要在SQL中加入left(需要分割字段, 分割位数)

* + 1. 示例



不需要分割字符串的





需要分割字符串



1. BI使用说明
   1. 维表管理

选择通过清洗、或者未清洗的数据表组成一个维度表。使用说明：

选择数据库中的一张表，这里有可能是视图， 也有可能是一个物理数据表。

维名称：为新建的维表起得一个方便理解的一个名称

字段显示：是当你选择数据表的时候，会把数据表的字段都读取出来

ID字段： 是字典表的ID， 也是这个字典表和基本数据表中相关联的字段

文字字段： 文字字段是ID字段的文字

属性设置： 关系到如果有钻取路径的时候， 这里面添加的就是钻取路径的属性， 例如钻取路径是省市县， 属性设置里面就有三个字段， 省、市、县对应的字段

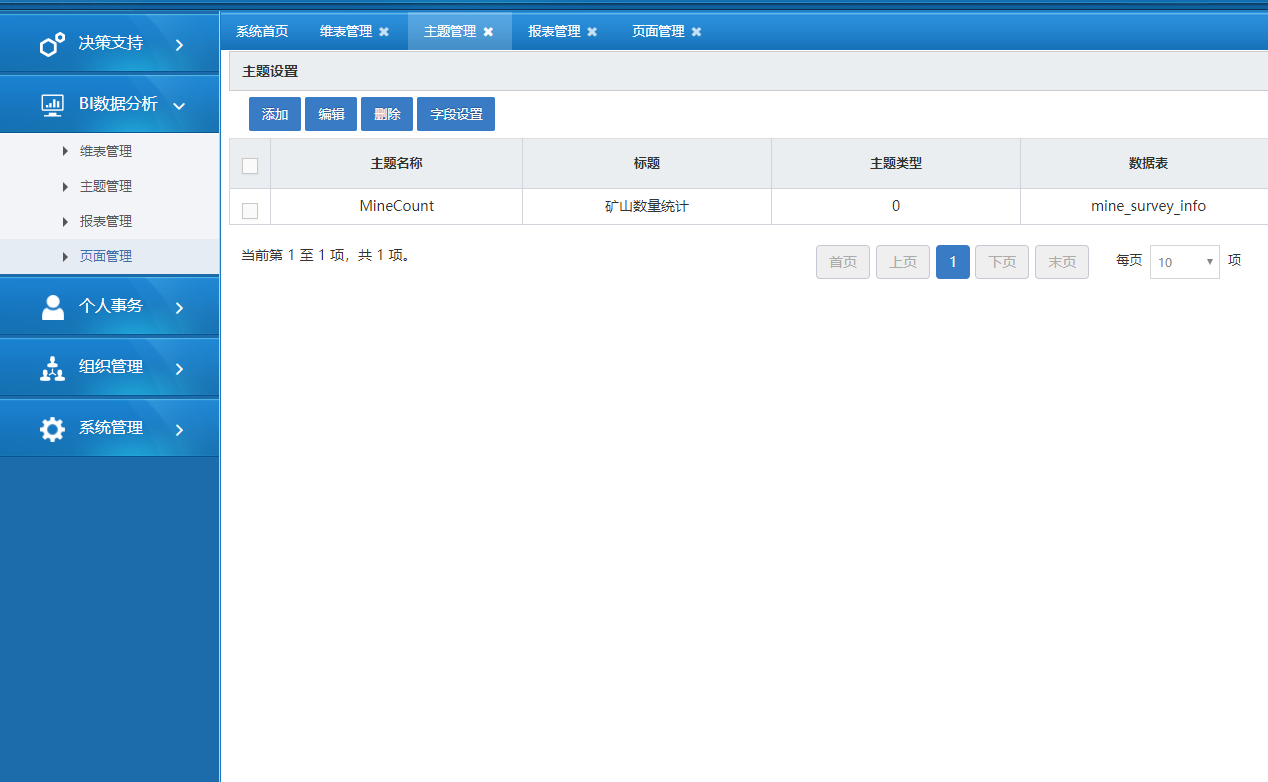
属性设置里面：三个选项， 目前只实现了第一个和第三个， 钻取路径是省市县钻取



* 1. 主题管理
     1. 主题管理（事实表）

一个主题可以理解为一个需要展示的事实表。包括主题名称、主题标题、主题类型、数据表

主题名称随意填写，主题标题也是的，主题类型暂时没有使用。



* + 1. 主题字段管理

对主题添加字段，主题字段需要自己主动设置，暂时没有做到默认把主题关联到数据表的所有字段拿过来。主题字段说明：

字段名称这个是从主题中选择的数据表中加载出来的字段，选择字段之后，字段标题自动填充， 不要修改

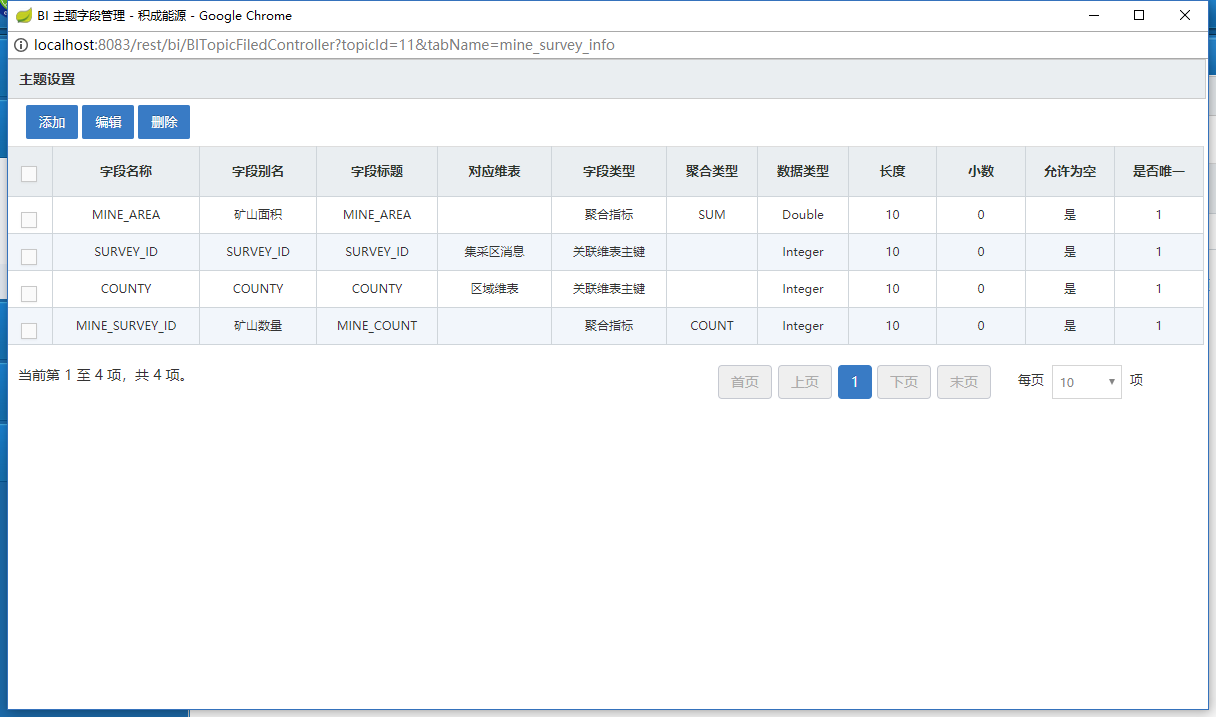
字段别名：”字段修改为字段的别名，这个是会显示在包表上面的。

字段类型： 字段类型有三种： 普通字段、关联维表字段、聚合字段

普通字段只是显示，不存在逻辑处理

关联维表字段， 就是这个统计数据表中与维表中ID字段对应的维表，选择后需要选择对应的维表，这样在查询的时候，对应的数据表中的1、2、3就能变成文字说明了

聚合字段就是指标了， 例如我要统计矿山数量就需要聚合字段然后聚合类型需要选择COUNT， 数据库中的聚合函数



* 1. 报表管理

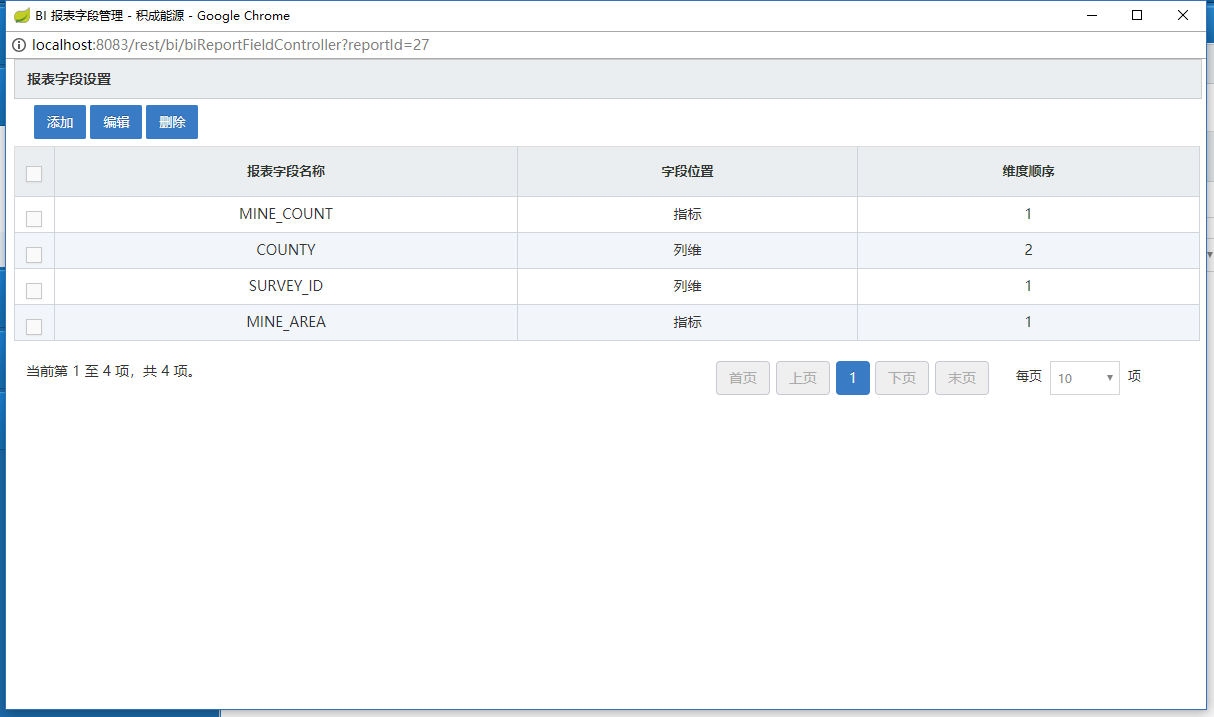
添加一个报表，报表关联一个主题。



添加报表字段，通过主题的字段进行重组成一个报表。

需要设置字段是指标还是行维、列维

这是注意的是这里需要设置维度的顺序，维度顺序就是设计到在页面展示的时候是处于列维的第几列



* 1. 页面管理

添加一个展示页面，页面需要选择页面模板，并且选择页面展示在哪里，如果关联菜单，则是在平台框架中打开，那就可以直接在菜单上点击打开，

如果是大屏展示，则菜单中不显示，不显示那怎么打开呢，所以就自动生成了一个连接地址，通过点击连接地址就可以打开大屏界面



* 1. 屏幕管理

设计思路是：一个页面可以显示多个屏幕，当选择第二个屏幕的时候，相当于与页面上的数据表全部重新更换数据源报表，进行刷新一次

屏幕名称：是显示在屏幕下方的，可以切换的选项

屏幕索引是:屏幕在页面中的排序

关联页面就是制定这个屏幕属于哪一个页面

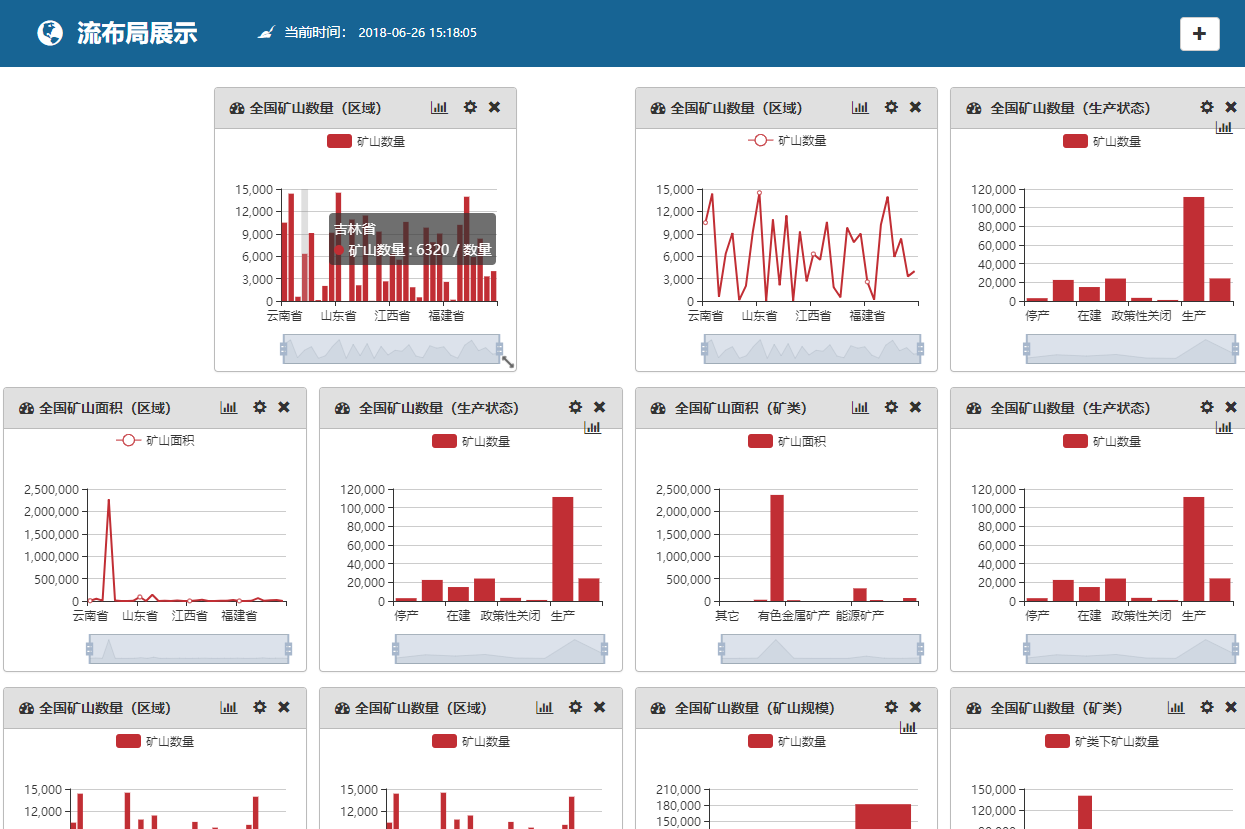


* 1. 展示页面
     1. 监控大屏展示页面

此页面标题为真实数据，图表数据为假数据，数据展示还没有书写，需要周一完成。

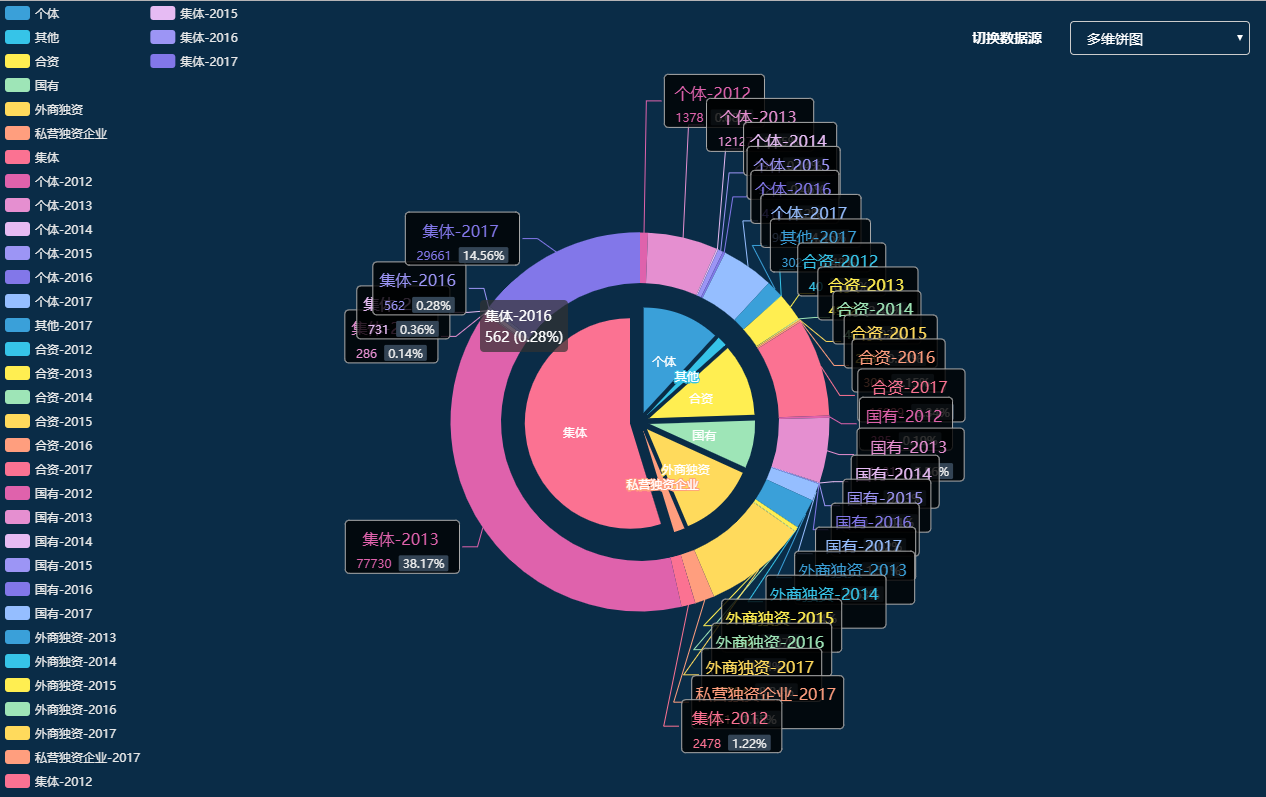
* + 1. 流布局展示页面

右上角添加按钮进行添加一个快，然后选择报表加载数据块的图表



* + 1. 多维表格展示页面

多个维度的表格展示，一个页面只允许显示一个



* + 1. 多维图表展示页面



数据可以进行钻取，导出excel



