数据访问服务运维

用户使用手册

**汉威科技**

**2017年8月**

**目录**

[前言 iv](#_Toc517274889)

[1. 主界面 1](#_Toc517274890)

[2. 管理信息 2](#_Toc517274891)

[2.1服务管理 2](#_Toc517274892)

[2.2 数据源管理 7](#_Toc517274893)

[2.3 授权管理 8](#_Toc517274894)

[2.4 用户管理 10](#_Toc517274895)

[2.5 日志查询 11](#_Toc517274896)

[3. 系统设置 12](#_Toc517274897)

[3.1数据库设置 12](#_Toc517274898)

[3.2数据库初始化 14](#_Toc517274899)

[3.3清理配置缓存 14](#_Toc517274900)

[3.4清理数据缓存 15](#_Toc517274901)

[3.5数据库认证设置 17](#_Toc517274902)

[3.6Token认证设置 18](#_Toc517274903)

[附录-sql语句xml描述说明 19](#_Toc517274904)

[1. Mapper 19](#_Toc517274905)

[2. DataSource 20](#_Toc517274906)

[3. Select 20](#_Toc517274907)

[4. Scalar 21](#_Toc517274908)

[5. Insert 21](#_Toc517274909)

[6. Update 22](#_Toc517274910)

[7. Delete 22](#_Toc517274911)

[8. Save 23](#_Toc517274912)

[9. procedure 24](#_Toc517274913)

[10. resultMap 24](#_Toc517274914)

[11. sql 25](#_Toc517274915)

[12. check 25](#_Toc517274916)

[13. Text 26](#_Toc517274917)

[14. Include 26](#_Toc517274918)

[15. Tablename 27](#_Toc517274919)

[16. Where 27](#_Toc517274920)

[17. Set 28](#_Toc517274921)

[18. If 29](#_Toc517274922)

[19. Foreach 30](#_Toc517274923)

[20. For 31](#_Toc517274924)

[21. parameter 31](#_Toc517274925)

[附录-sql描述示例1-数据源 32](#_Toc517274926)

[附录-sql描述示例2-语句 34](#_Toc517274927)

[附录-sql描述示例3-表名 38](#_Toc517274928)

# 前言

数据是企业管理之根本，为生产管理决策提供了重要依据，而信息化工具起到了举足轻重的作用，但随着应用程序数量及复杂性的增加，出现了许多难以协调的问题。主要表现在在工作过程中产生了大量的独立数据源和大量的数据格式，便应用软件之间需要进行数据转换和加载才能调用各自产生的成果数据，浪费了研究人员大量时间，同时研究成果难以在应用软件之间深层次共享。

* 数据源类型多，SQL语法不同，编写复杂，制约应用开发效率；
* 业务模型变更或是数据源改造，会导致相关应用大面积重构；
* 原有数据访问层弊端日益明显，已经不能满足云技术发展需要；
* 数据服务缺乏统一的标准规范，接口协议种类多，应用成本高；
* 缺乏有效的数据安全控制规范，无法实现数据的统一安全管理。

建设完善的数据服务体系，为应用提供安全、高效、标准化的数据服务接口，具体包括以下4个方面：

1. 搭建完善的数据服务体系架构

梳理数据服务接口的申请、审核、创建、发布、应用、监控等业务流程，建立完善的数据服务体系架构，满足各种数据服务需求的应用要求。

1. 提供稳定、简洁的数据服务管控

提供界面简洁、易操作数据服务运维管理功能，方便数据管理人员管理的同时，严格控制数据操作权限，确保数据安全。

1. 配套可靠的数据集成和数据处理

配套完善的多数据来源的数据集成功能，实现复杂的数据转换处理，支持数据源的垂直或是水平分库、分表策略。

1. 构建数据服务标准规范

搭建Rest风格化的标准服务接口，提供适应性强、安全性高、标准化的数据服务，满足业务应用跨平台的数据访问需要。

# 主界面

登录**<http://10.68.239.202:8080/Manage/login/index.html>**

您将看到如下界面：



图1-1数据访问服务登录界面

用户输入用户名与密码后，单击登录按钮，即可打开数据访问服务管理系统主界面。如图1-2所示：

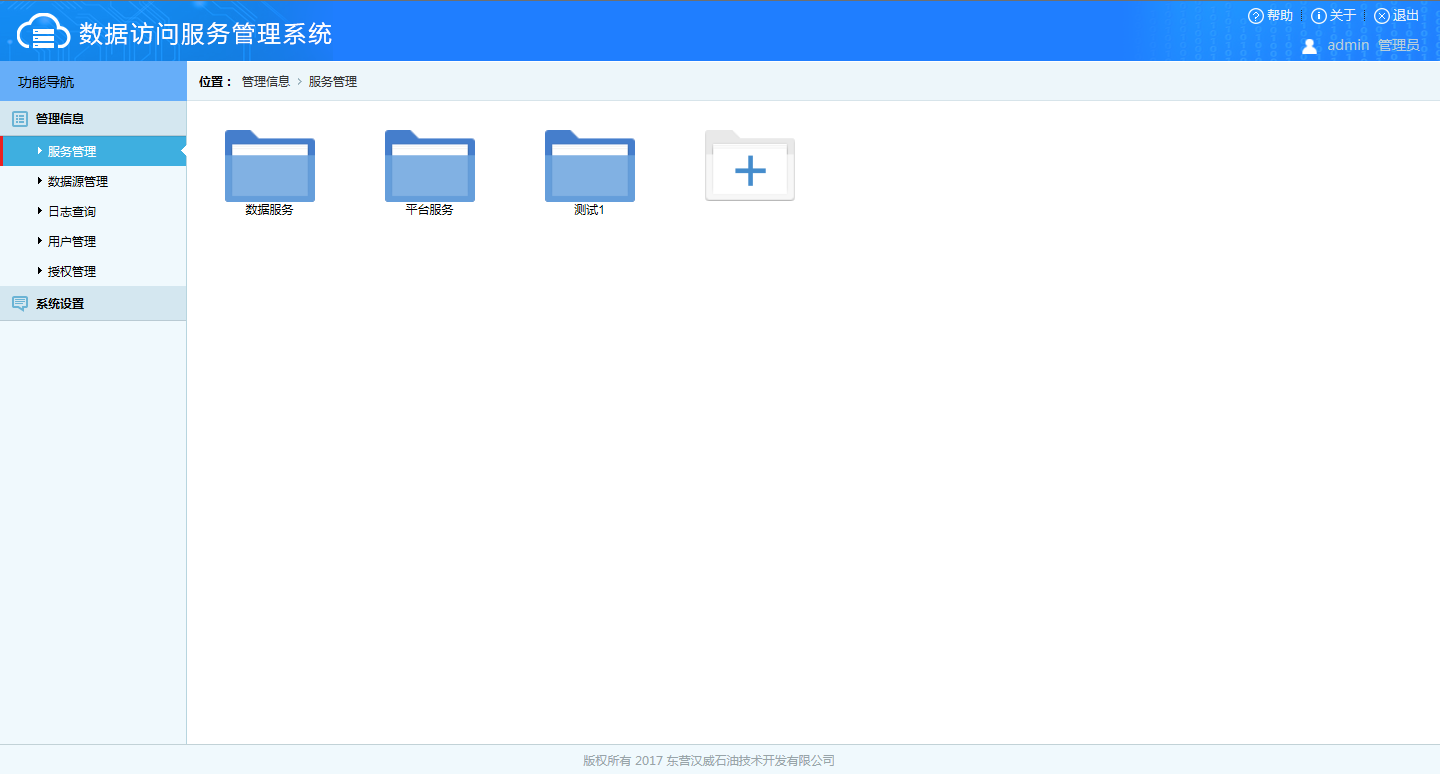


图1-2 主界面

左侧功能导航部分包含管理信息和系统设置：

管理信息模块分为：服务管理、数据源管理、日志查询、用户管理和授权管理；

系统设置模块分为：数据库设置、清理缓存、认证设置；

# 管理信息

## 2.1服务管理

服务管理模块用于添加、编辑和删除服务分类、服务资源以及服务接口方法。

如下图2-1-1所示：单击加号图标，打开添加对话框，录入完成单击确定按钮，即可看到添加的服务第一级分类名。

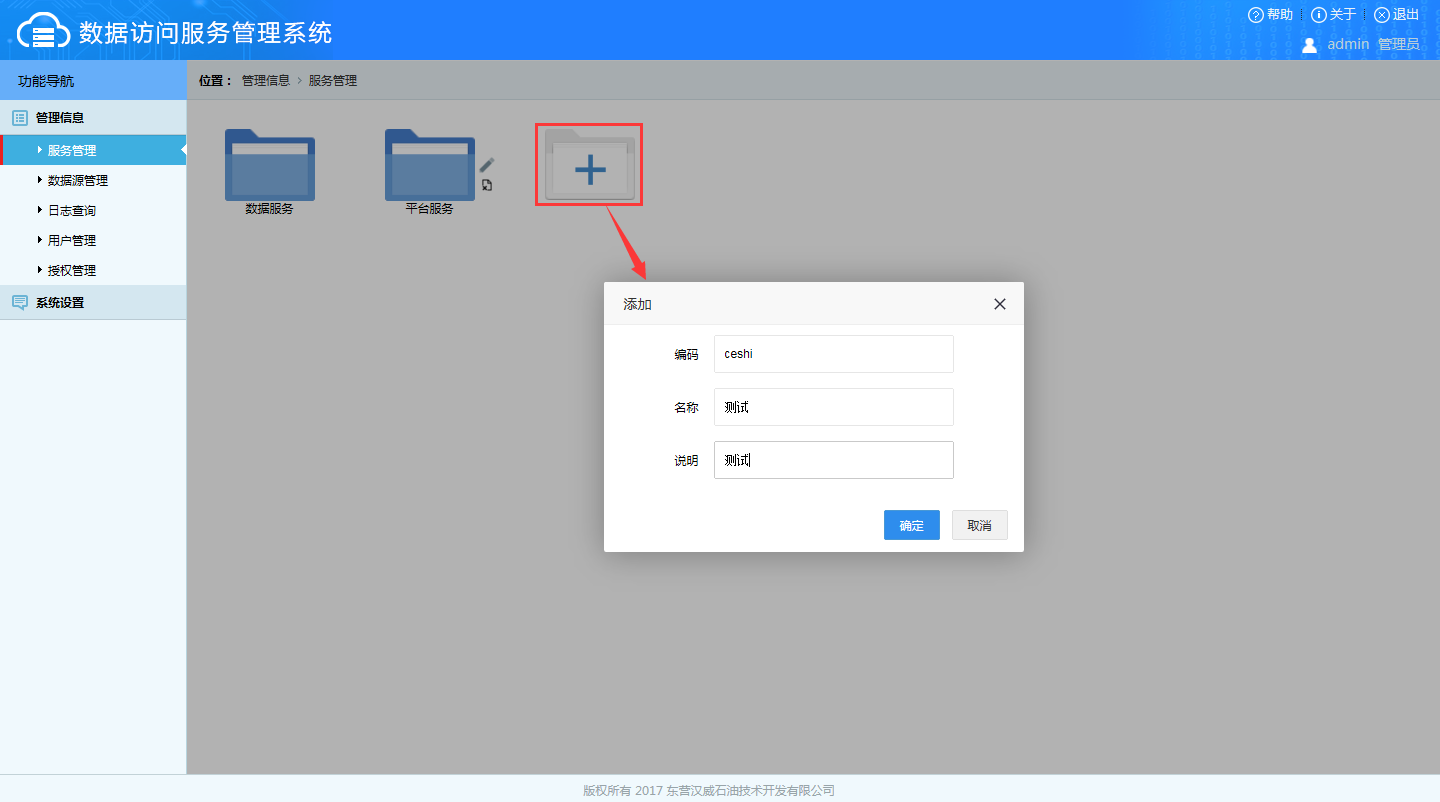


图2-1-1

如下图2-1-2所示：鼠标指向文件夹图标时，右侧会显示编辑和删除图标，可对其进行编辑和删除操作。

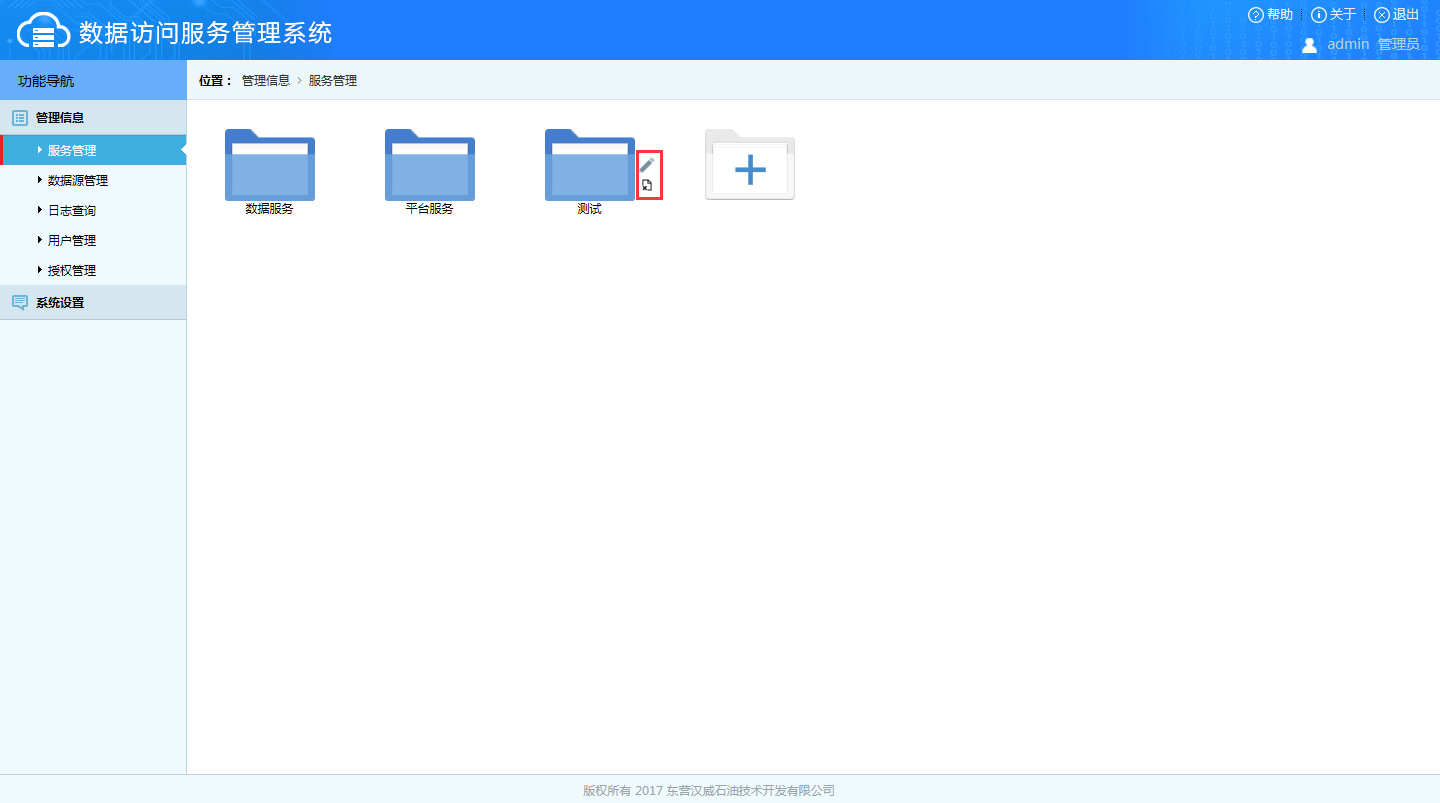


图2-1-2

鼠标单击文件夹弹出数据源管理和子分类管理页面，如图2-1-3所示：

数据源管理用于本大类中的数据访问服务用到的数据库连接信息管理，在sql语句xml编辑中会用到这里的数据访问连接编码信息。

选择数据源节点，单击添加按钮编辑数据链接信息然后单击保存，

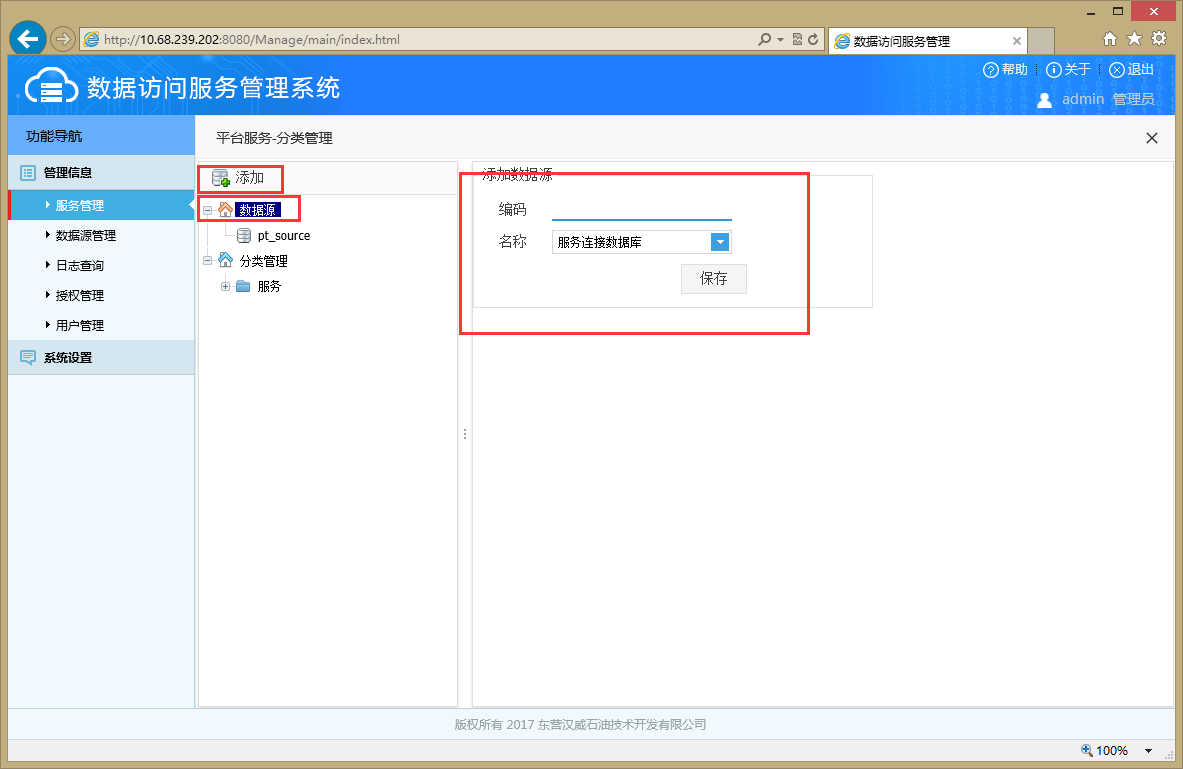


图2-1-3

注：数据源名称采用选择方式，来源于数据源管理，这里只选择对应的数据源名称。

选择分类管理节点，单击添加按钮编辑类别信息然后单击保存，即可在左侧看到新加的服务类别。

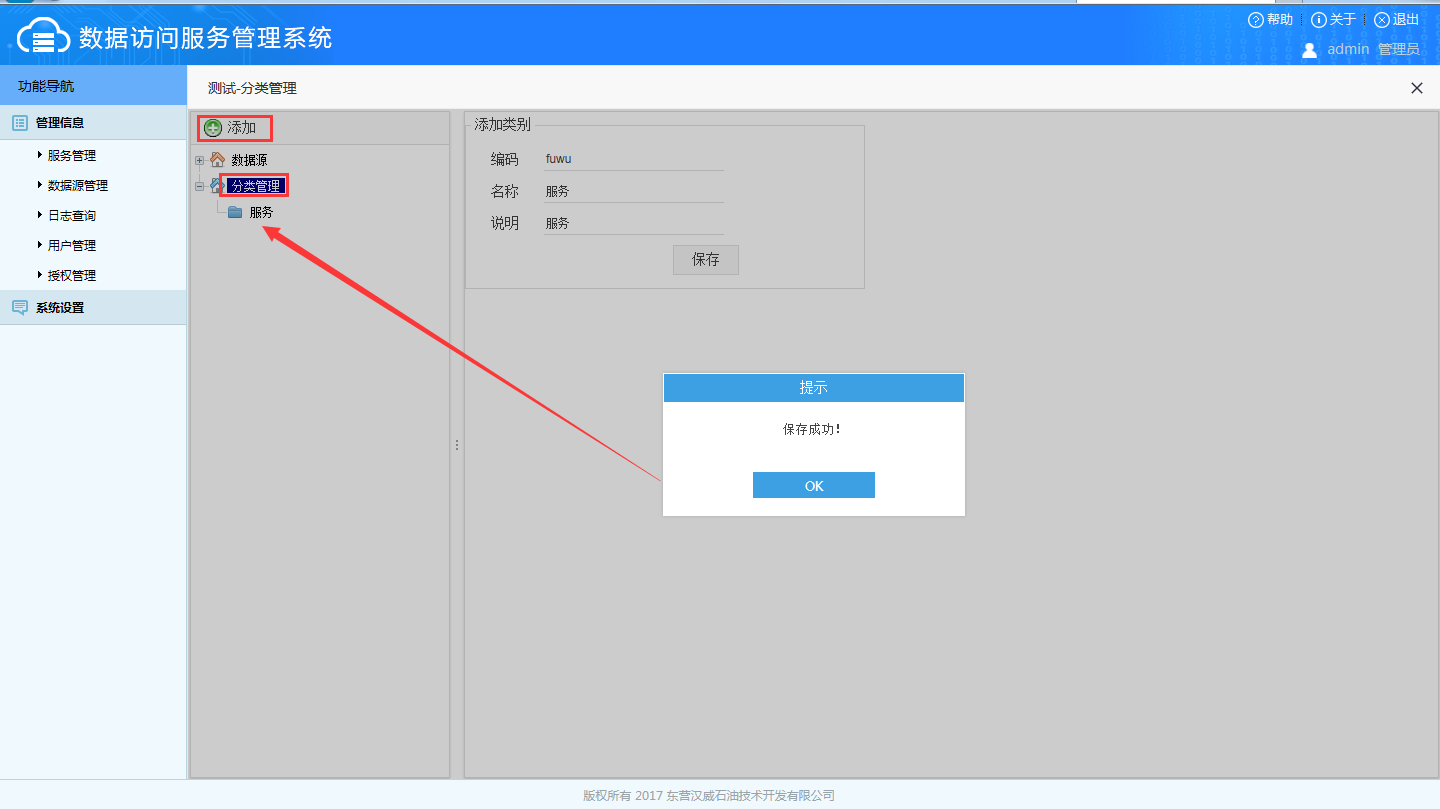


图2-1-3

如图2-1-4所示：选中服务类别，单击添加列表中的资源项，编辑完信息单击保存，即可在左侧看到新加的资源项，并以黄色高亮显示该文件夹。

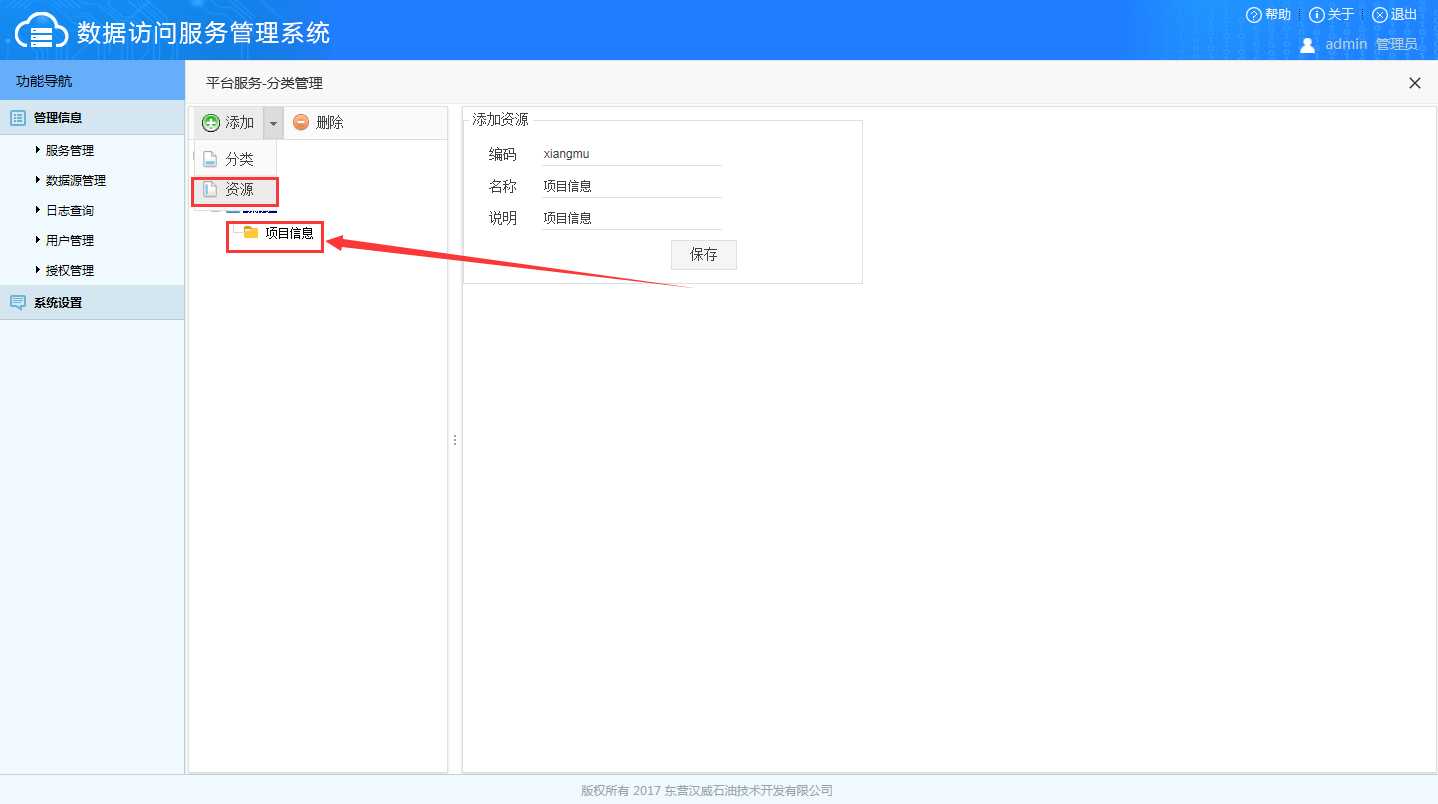


图2-1-4

如图2-1-5：选中项目信息，然后选择添加列表项中的方法名（以查询方法为例）。

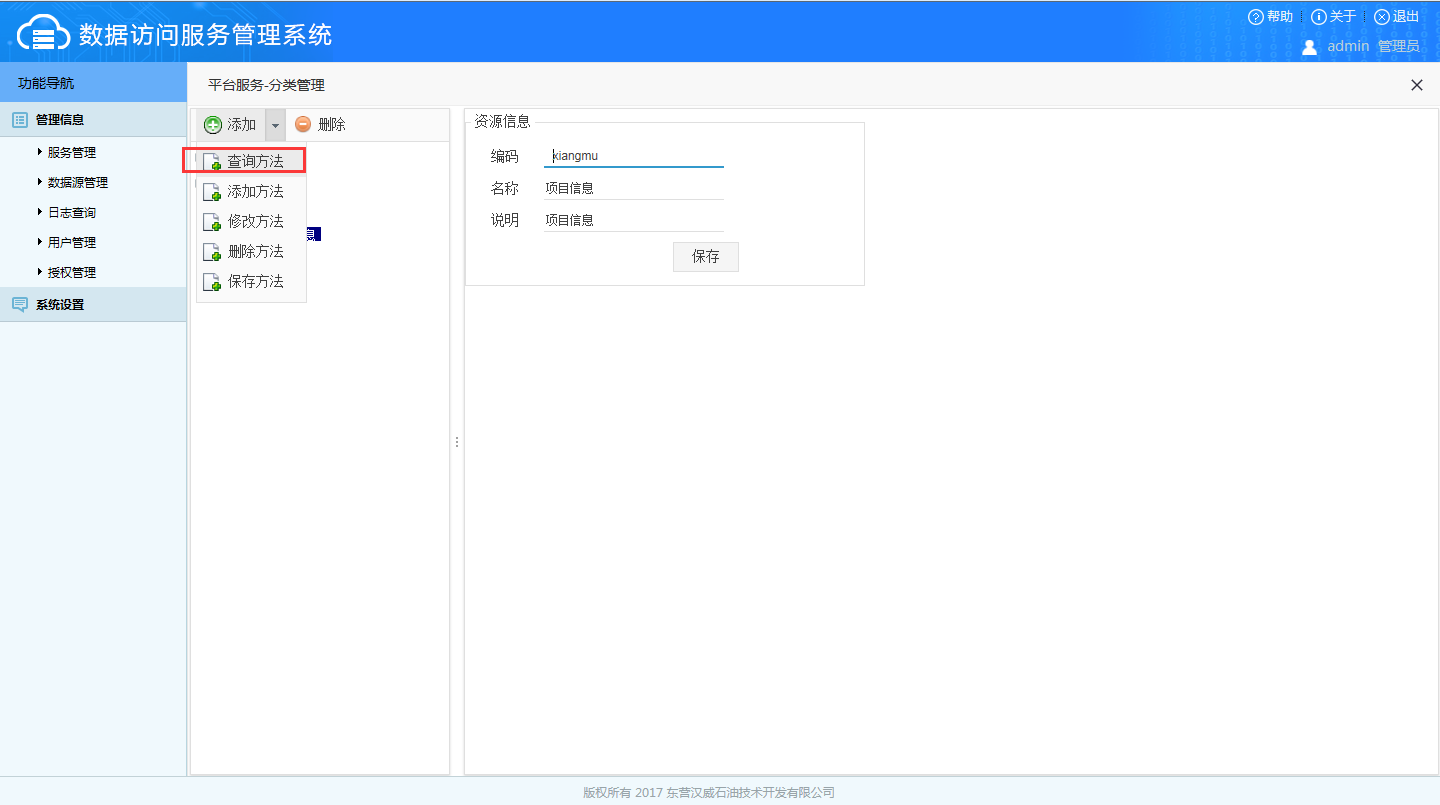


图2-1-5

如图2-1-6：选择左侧查询方法，单击新增按钮，在添加参数框中输入参数信息，然后单击提交按钮。

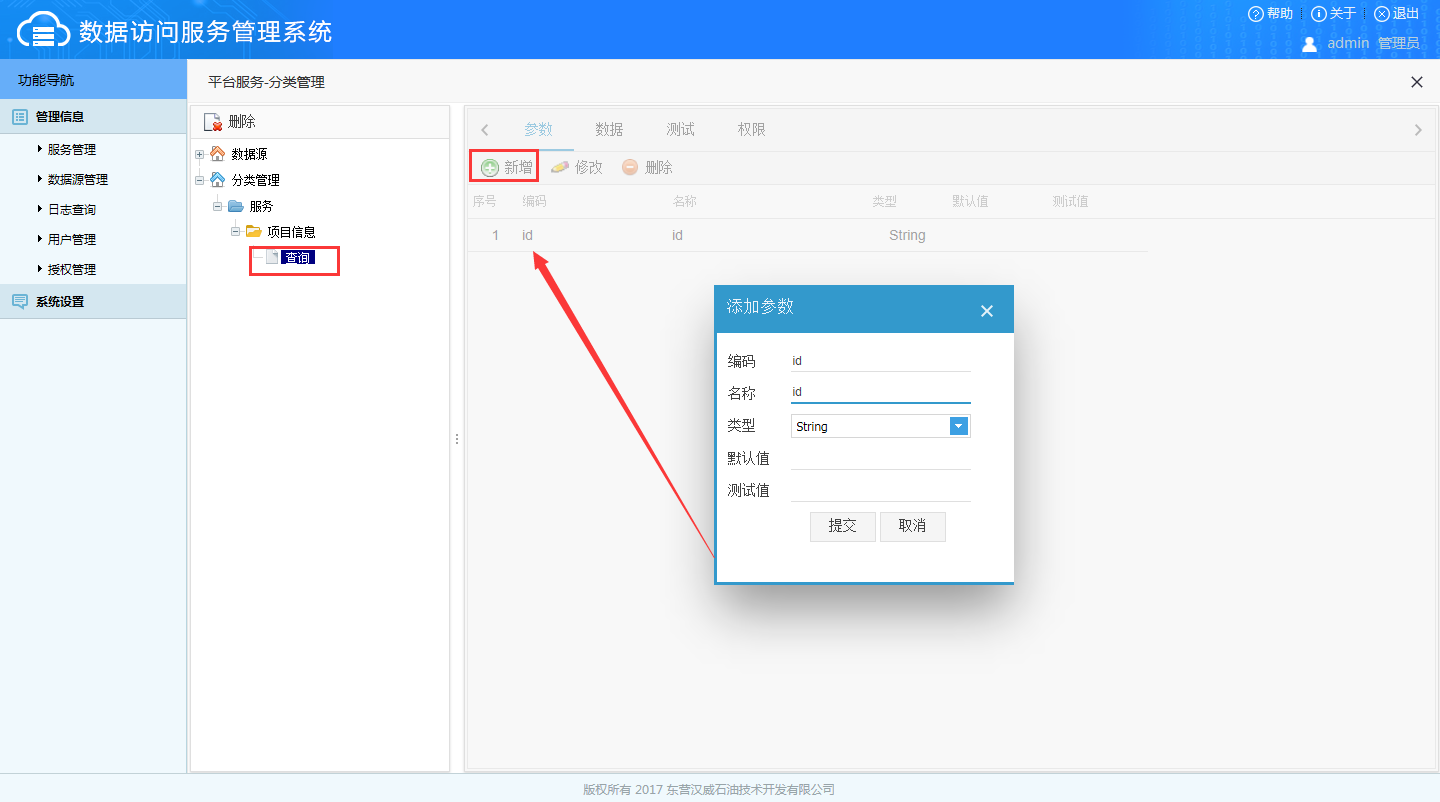


图2-1-6

如图2-1-7：选择上方导航条的数据项，找到要操作的表并先展开数据表，右键执行“构建select”命令，即可在脚本编辑器中自动生成SQL语句。

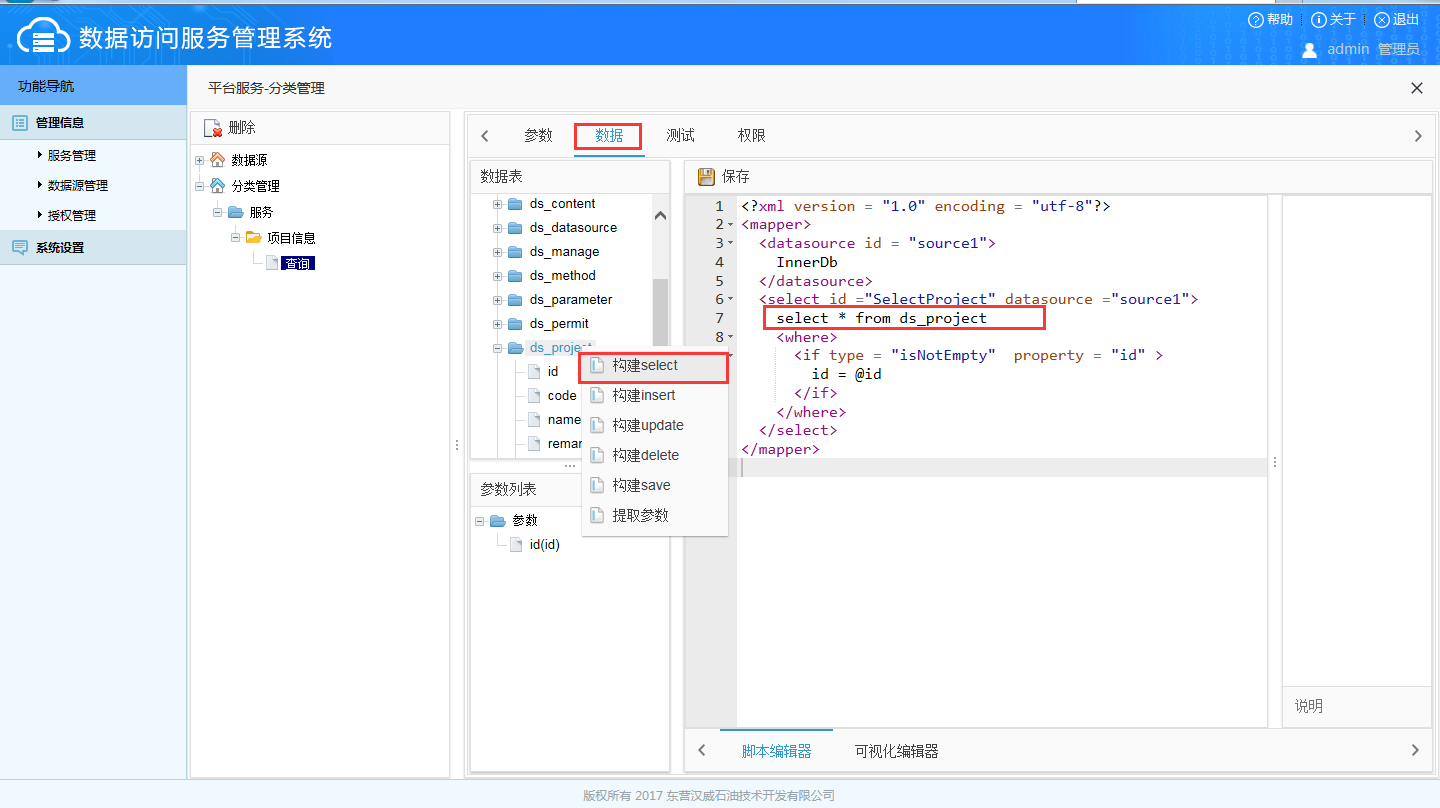


图2-1-7

如图2-1-8：选择上方导航条的测试项，单击执行按钮，查看右侧执行状态是否成功。

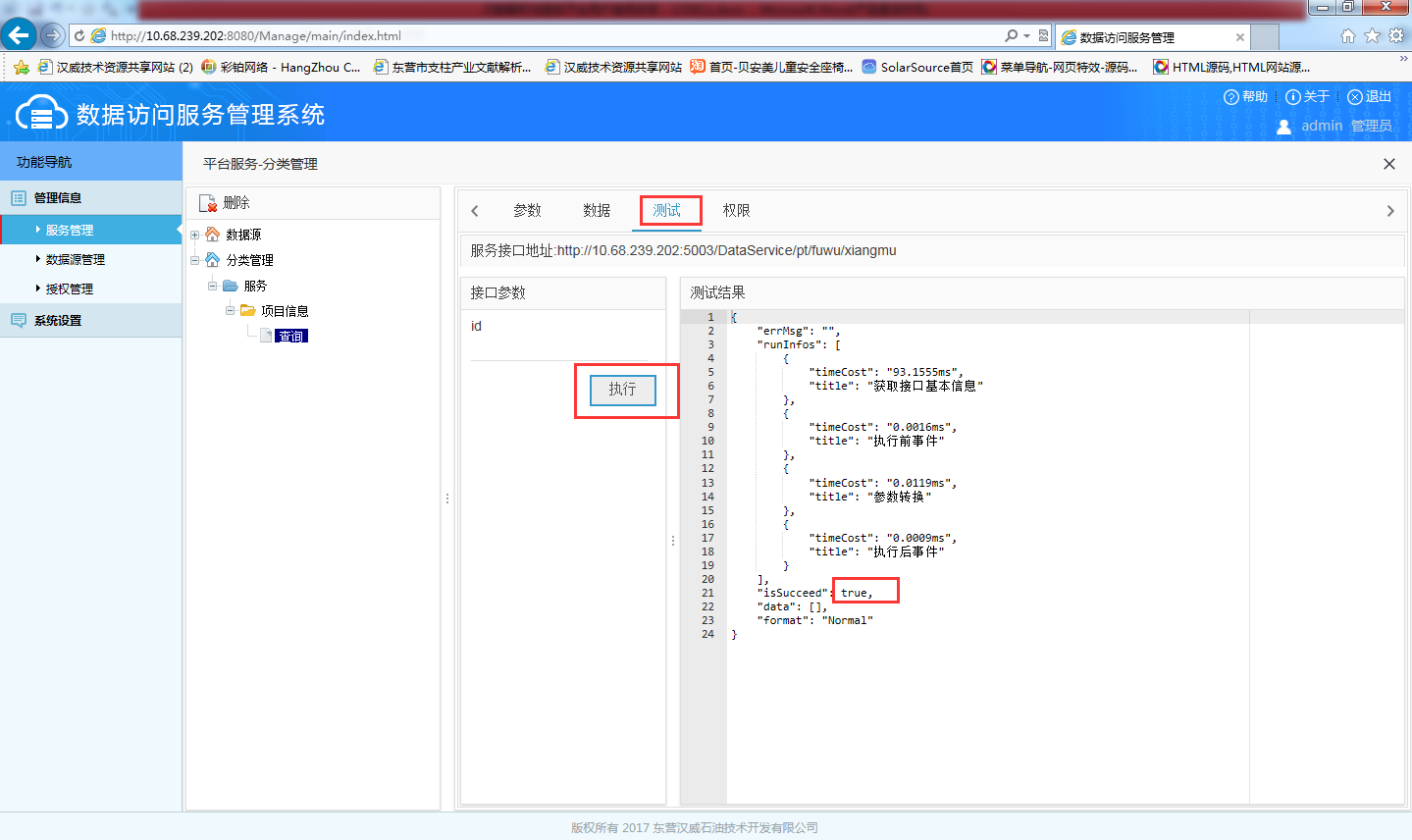


图2-1-8

如图2-1-9：选择上方导航条的权限项，勾选要设置的权限，并单击保存按钮，提示保存成功。

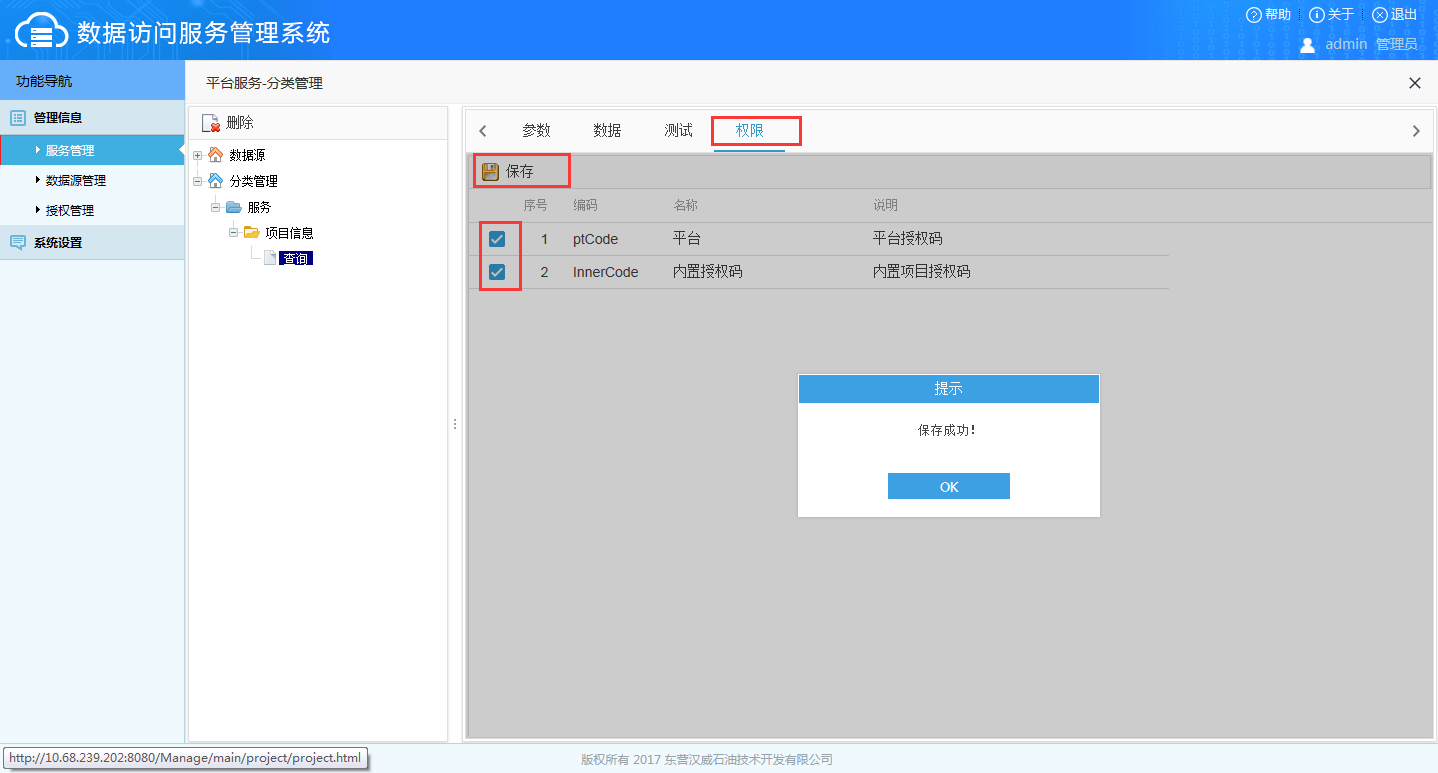


图2-1-9

## 2.2 数据源管理

如图2-2-1：选择数据源管理项，单击新增按钮，在弹出的添加数据源框中输入数据库配置信息，单击提交按钮，可看到新加的数据库信息。

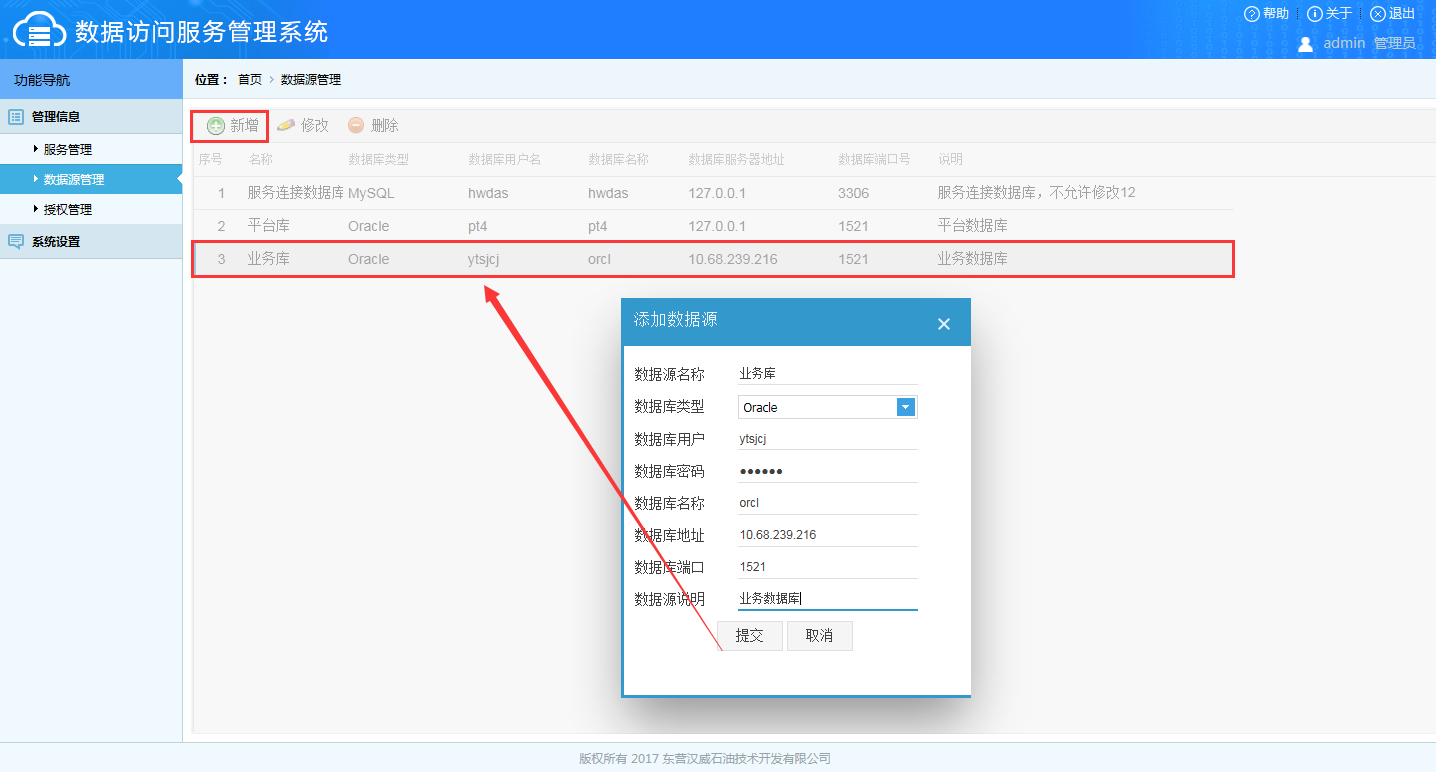


图2-2-1

如图2-2-2：选中要编辑的数据源，单击修改按钮，在弹出的编辑参数对话框中修改数据库的配置参数，然后单击提交按钮。

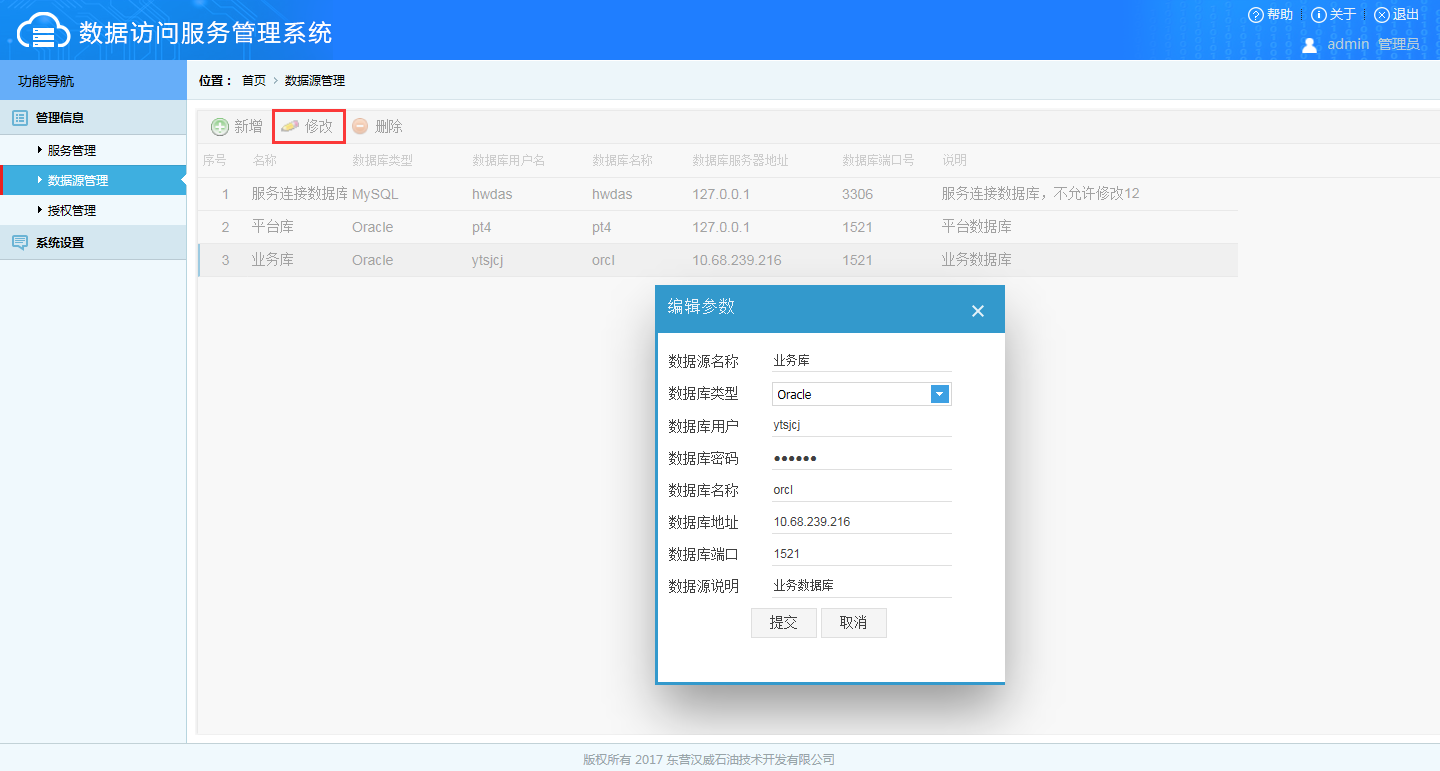


图2-2-2

如图2-2-3：选中要编辑的数据源，单击删除按钮，在提示框中单击确认即可删除。

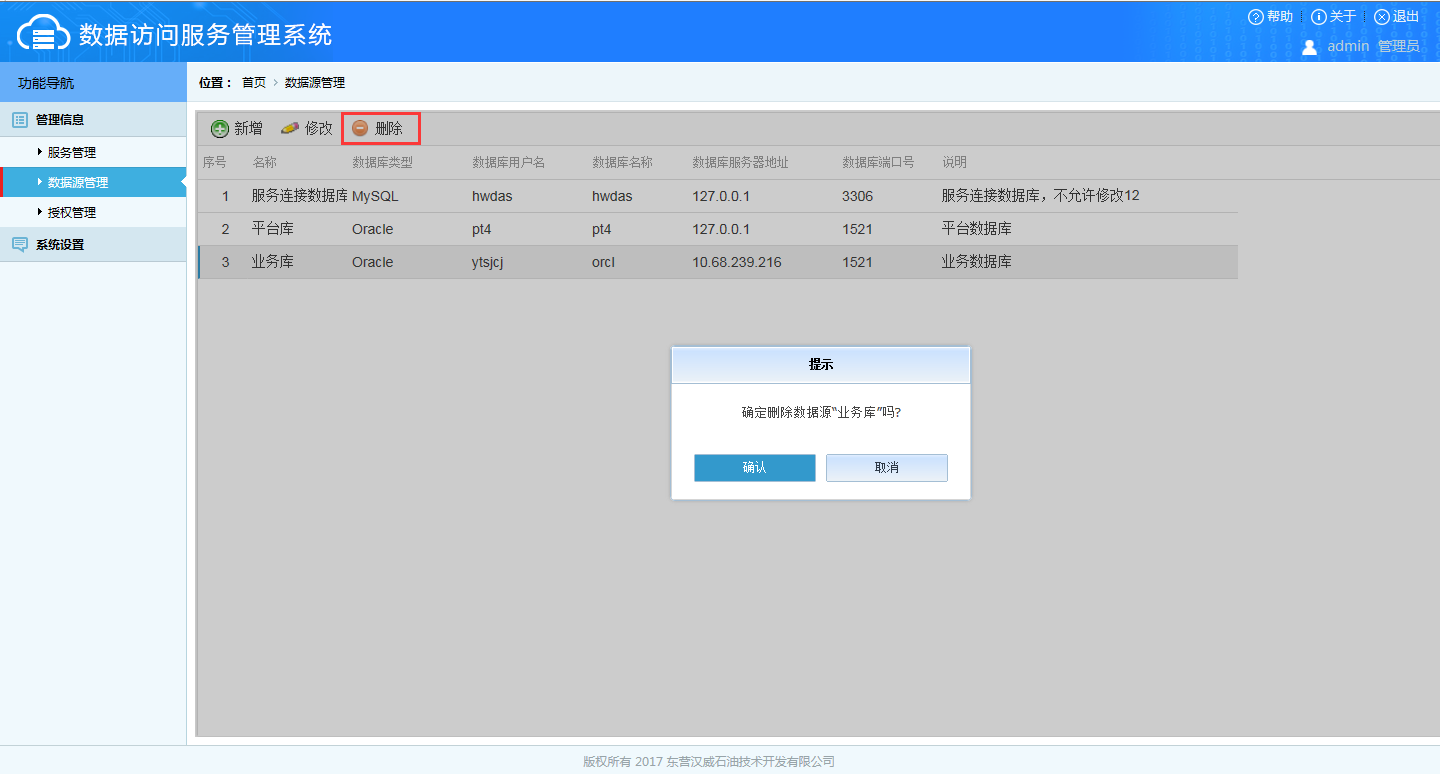


图2-2-3

## 2.3 授权管理

如图2-3-1：选中授权管理，单击新增按钮，在弹出的添加授权对象框中输入授权项，单击提交按钮，即可看到新增的授权信息。

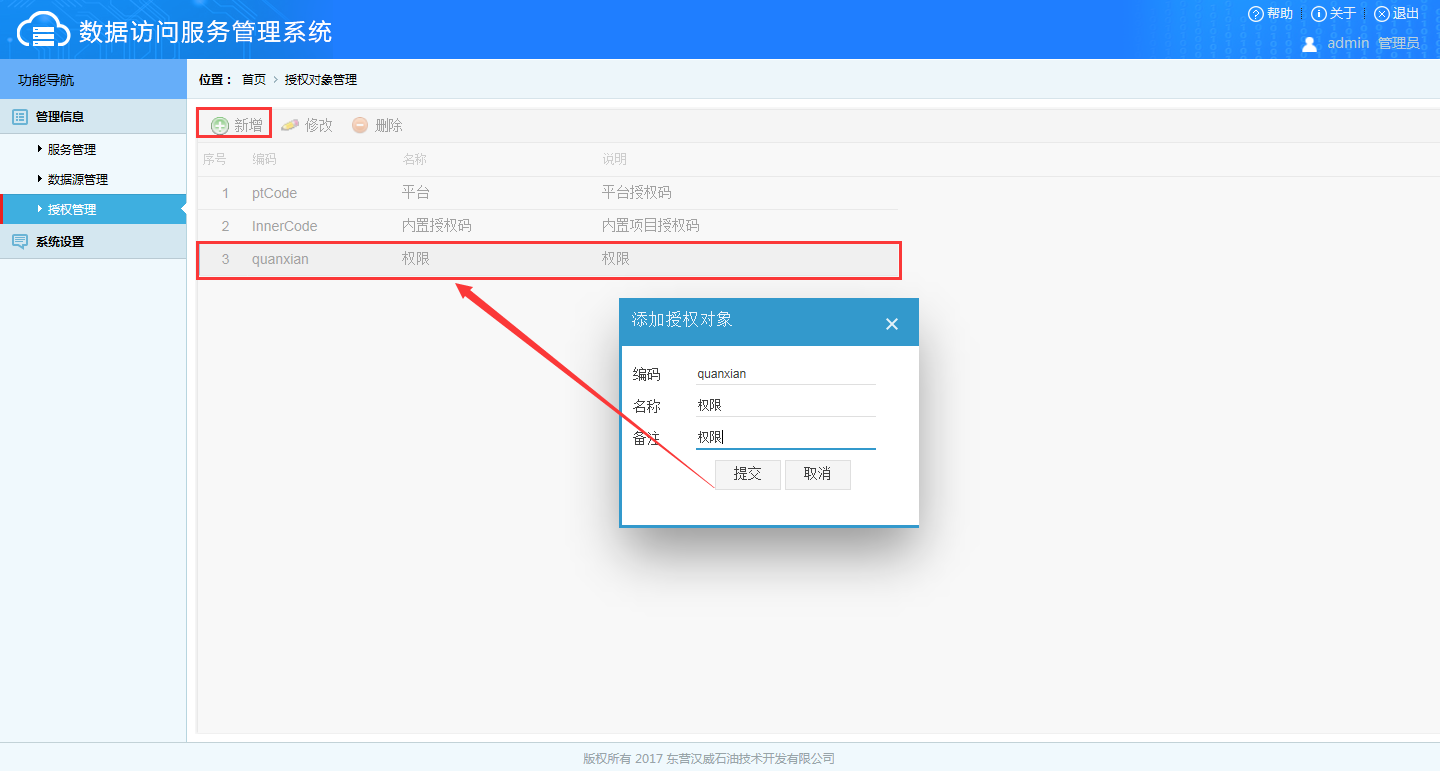


图2-3-1

如图2-3-2：选中要修改的授权对象，单击修改按钮，在弹框中编辑授权对象信息，然后单击提交按钮。

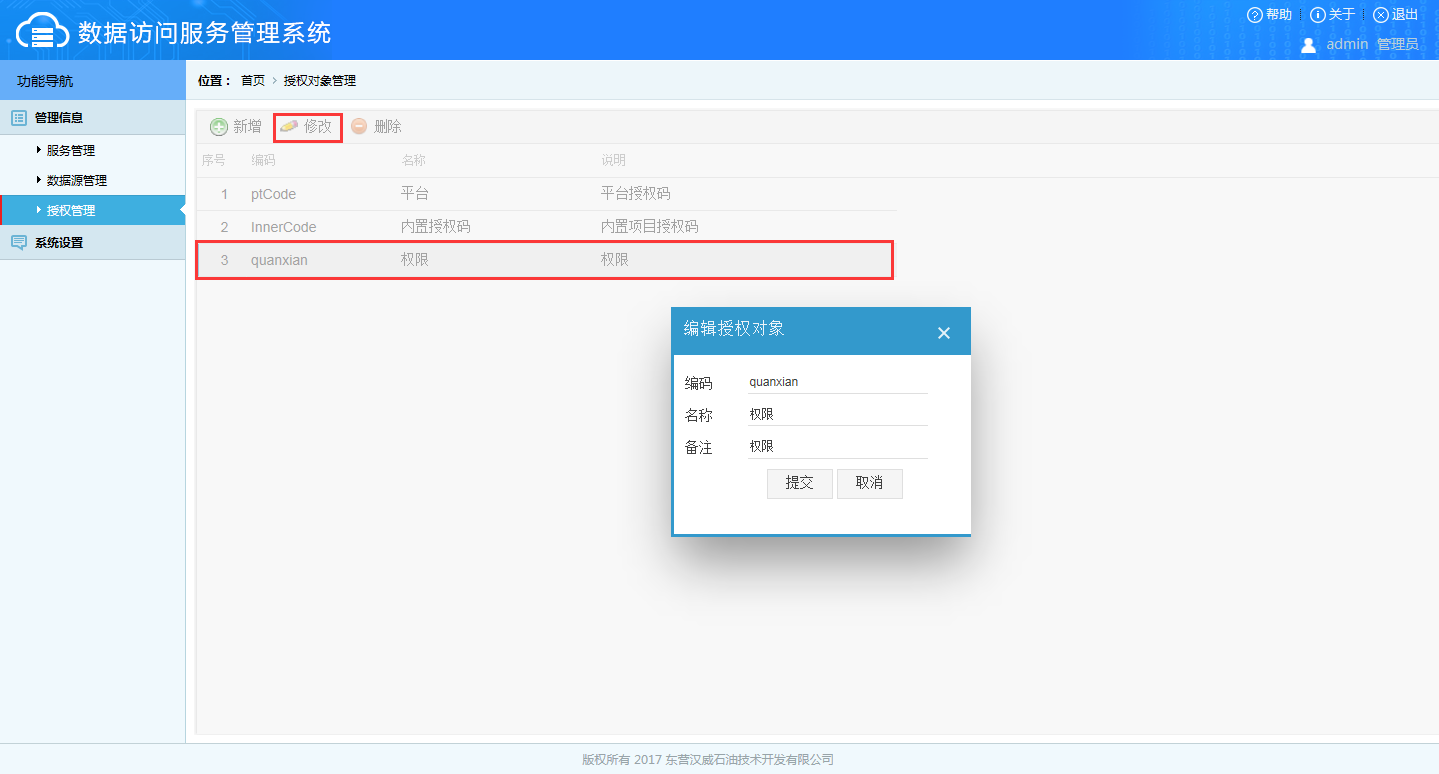


图2-3-2

如图2-3-3：选中要删除的授权对象，单击删除按钮，在弹出的提示框中单击确认按钮。

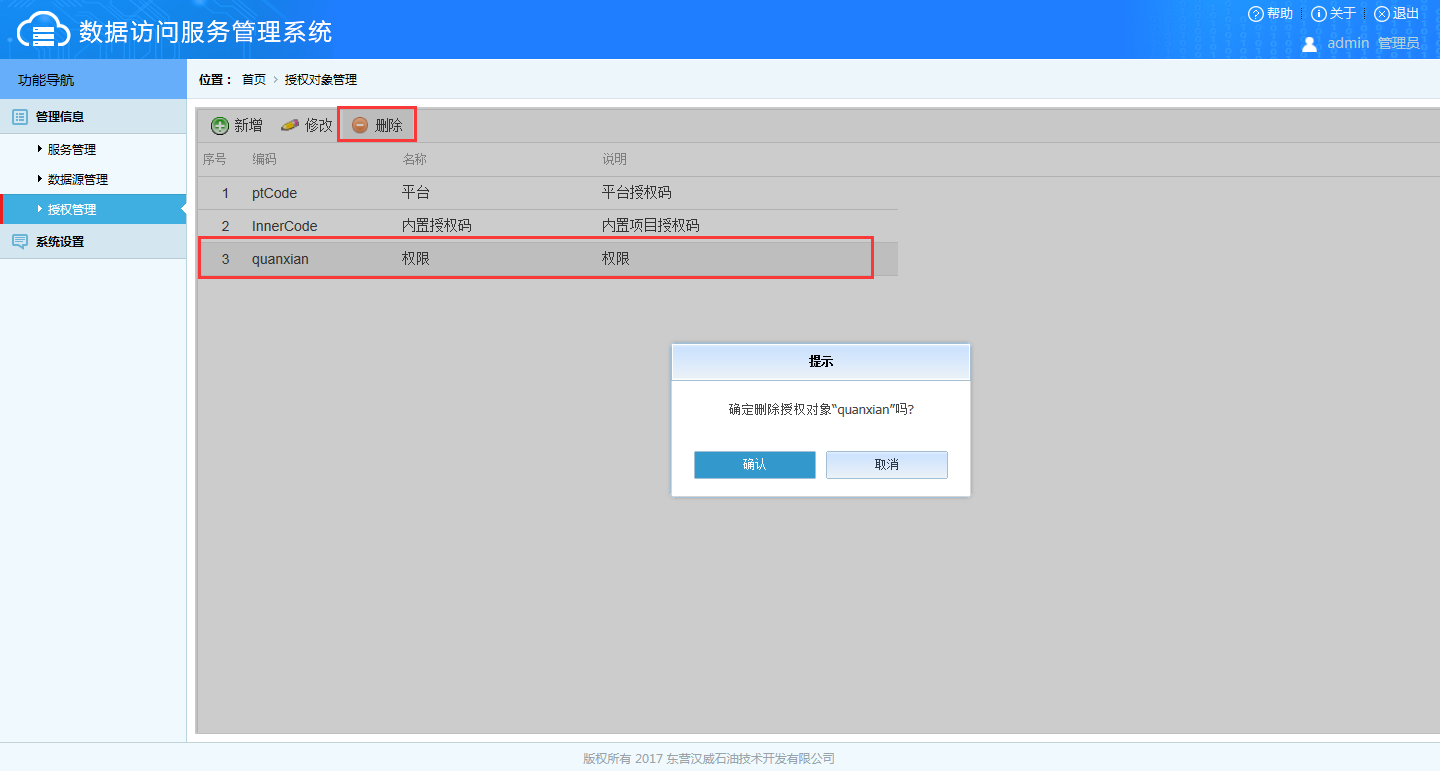


图2-3-3

## 2.4 用户管理

如图2-4-1：选中用户管理，单击新增按钮，在弹出的添加用户对象框中输入用户项，单击提交按钮，即可看到新增的用户信息。

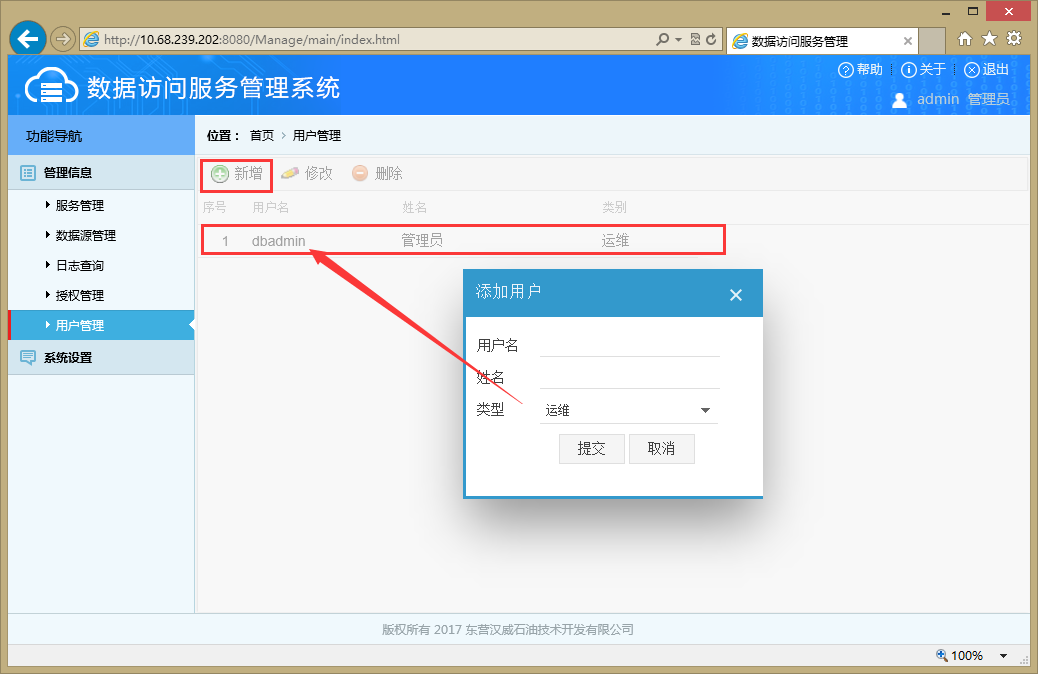


图2-4-1

如图2-4-2：选中要修改的用户对象，单击修改按钮，在弹框中编辑用户对象信息，然后单击提交按钮。



图2-4-2

如图2-4-3：选中要删除的用户对象，单击删除按钮，在弹出的提示框中单击确认按钮。

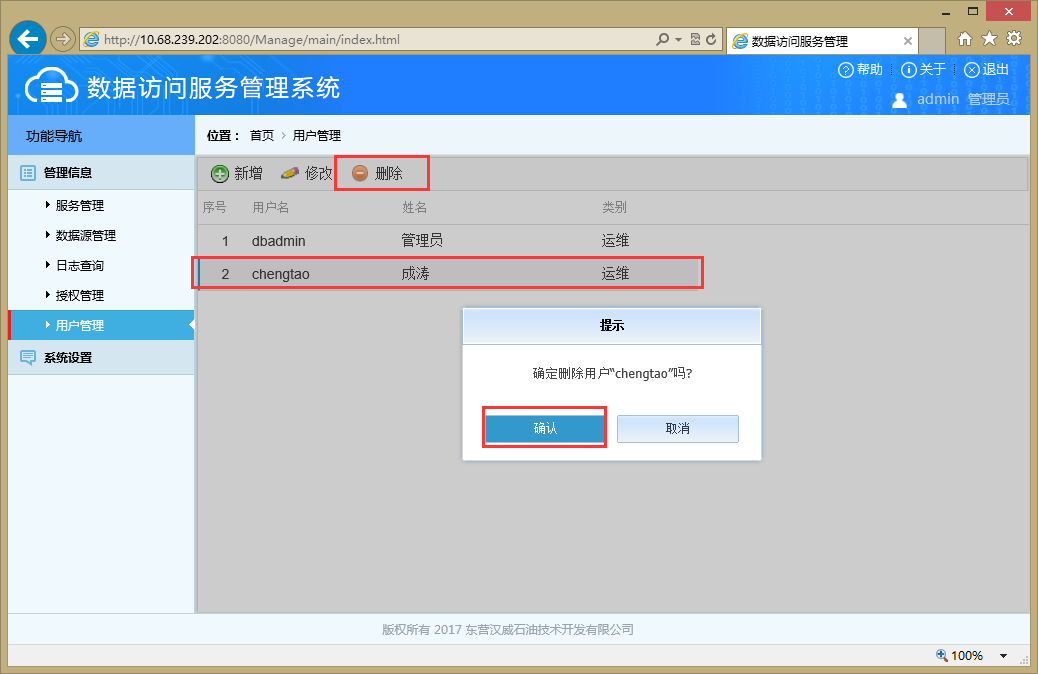


图2-4-3

## 2.5 日志查询

如图2-5-1：选中日志查询，选择开始和结束日志，单击查询按钮，即可查询时间段内的日志信息。

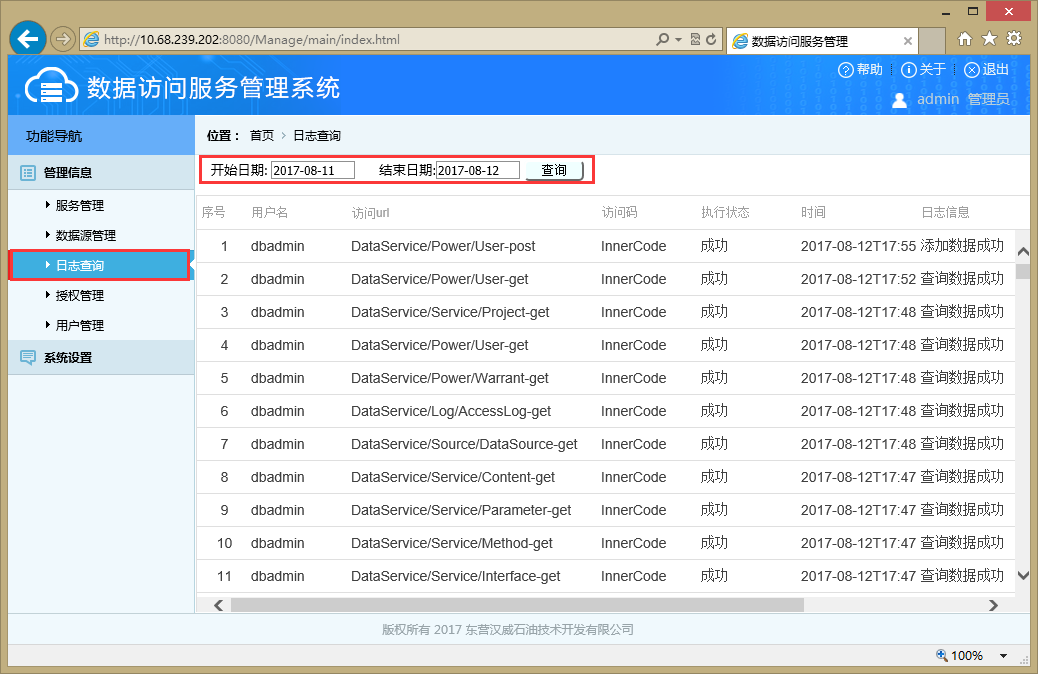


图2-5-1

# 系统设置

## 3.1数据库设置

数据库设置用户维护数据访问服务连接的数据库链接信息。

如下图3-1-1所示：单击左侧系统设置->数据库设置->数据源设置 打开配置界面,根据信息内容填写相应的数据。

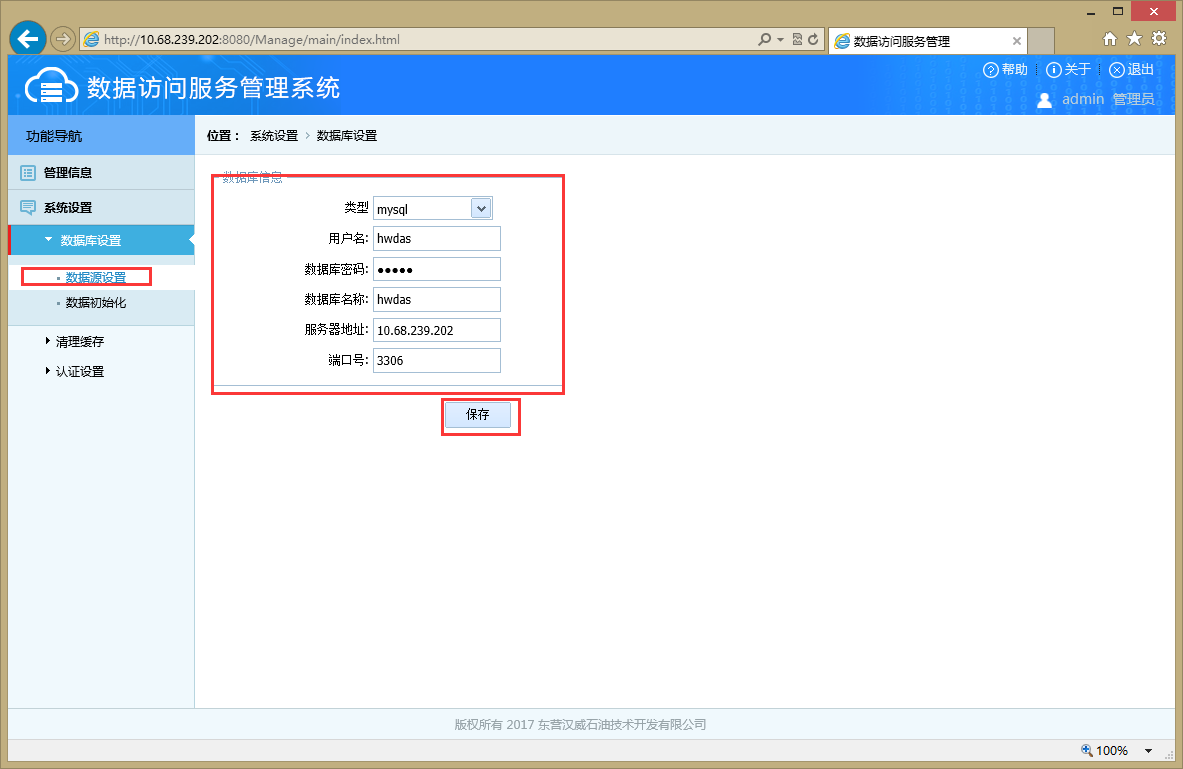


图3-1-1

填写完成后,点击”保存”按钮，进行数据保存，如果保存成功，会有提示，如下图3-1-2所示。

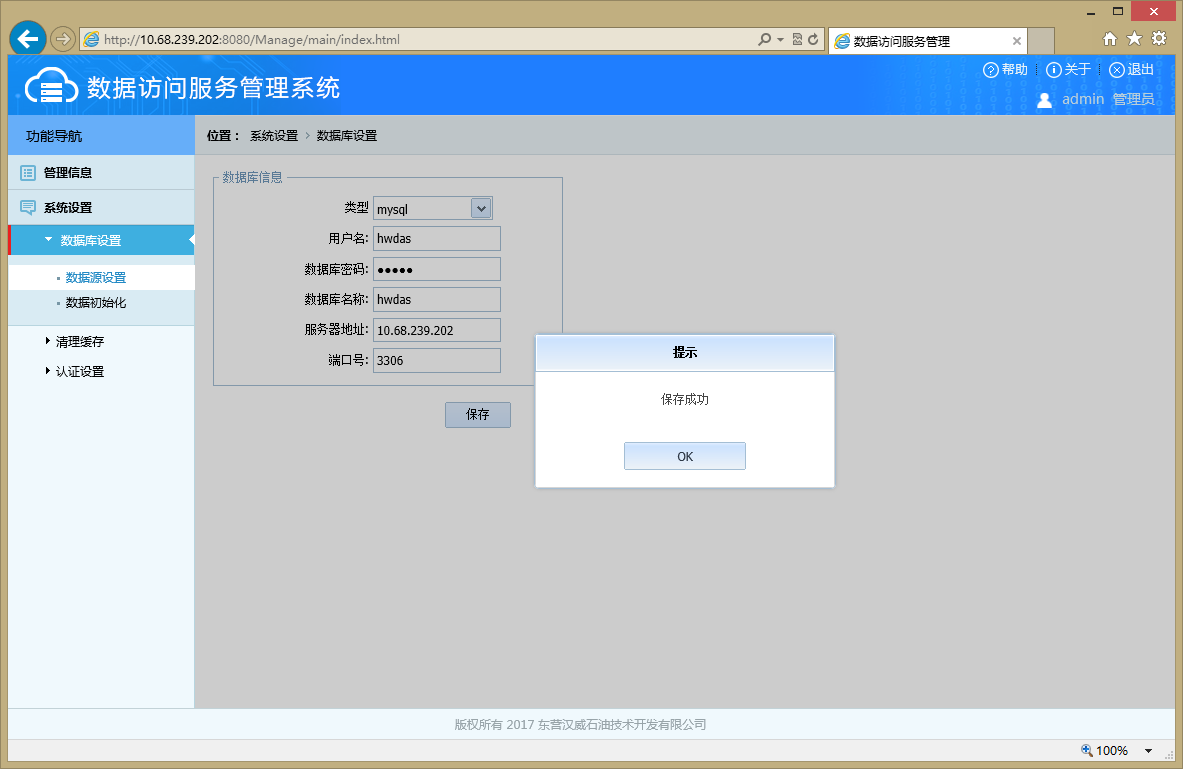


图3-1-2

## 3.2数据库初始化

数据库初始化用于初始化数据访问服务数据库，包括表结构以及初始数据。

单击“系统设置”->“数据库设置”->“数据初始化”进行数据库初始化操作,如果已经进行了初始化,则提示如下图3-2-1所示。

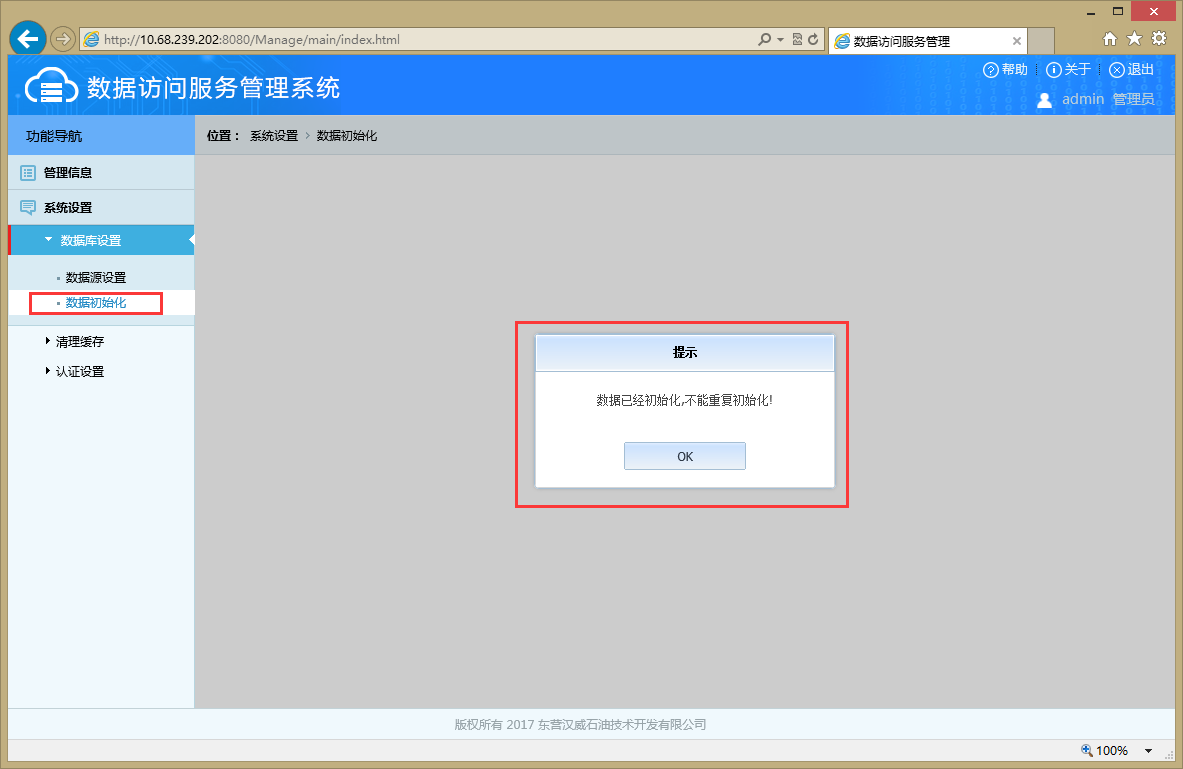


图3-2-1

## 3.3清理配置缓存

清理配置缓存用于清理数据访问服务生成的数据访问服务配置缓存,数据访问服务每次访问都会检测本地缓存,如果有就读取缓存信息,如果没有就从数据访问服务数据库读取方法信息,所以当通过数据访问服务修改服务方法后,就需要执行本操作。

单击“系统设置”->“清理缓存”->“清理配置缓存”进行缓存数据清理操作,提示如下图3-3-1所示。

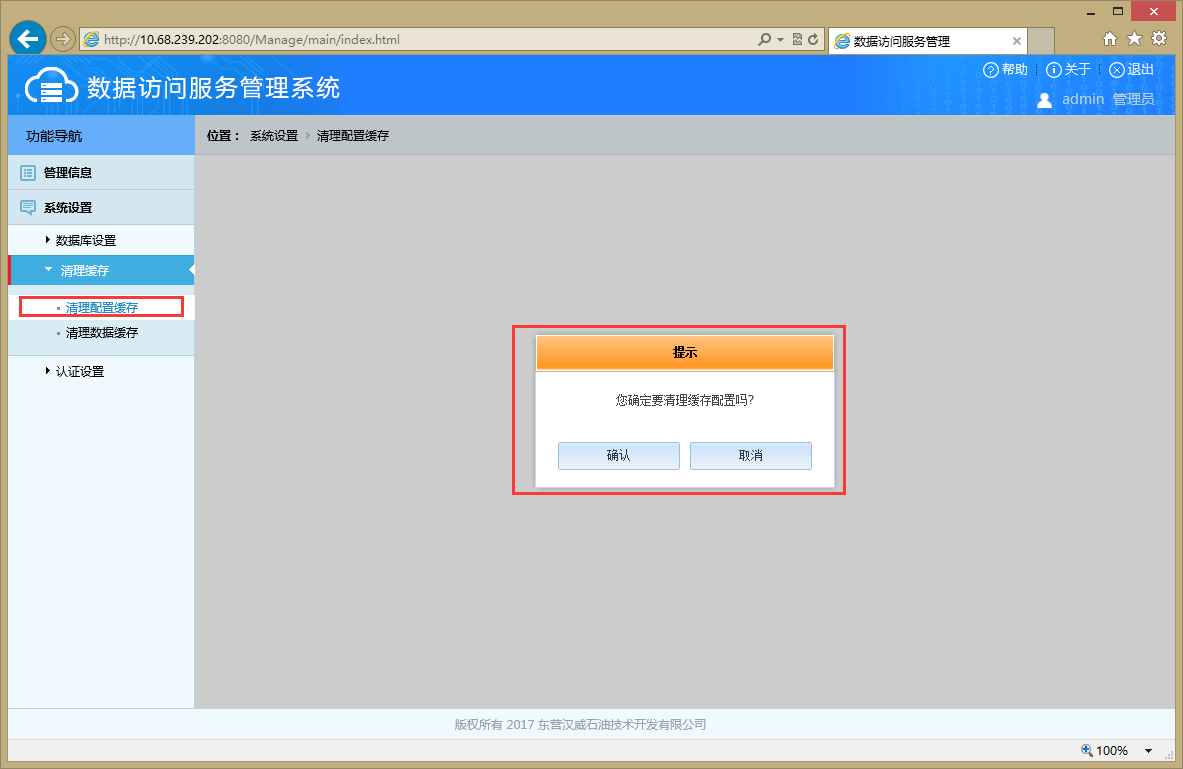


图3-3-1

点击”确认”按钮,清理缓存,如果清理成功,则提示如下图3-3-2所示.

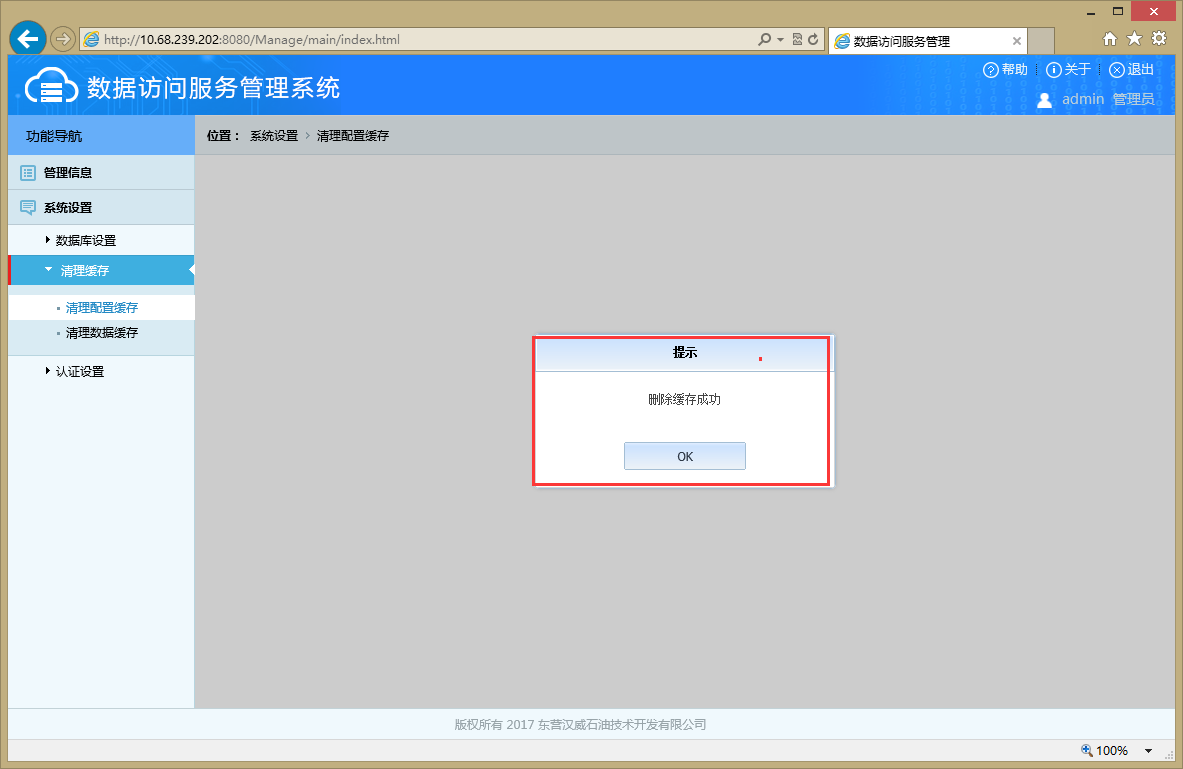


图3-3-2

## 3.4清理数据缓存

清理数据缓存用于设置了手动清理缓存数据的数据访问服务接口,清理服务器本地缓存,使其在下次访问时,从数据库读取数据。

单击“系统设置”->“清理缓存”->“清理数据缓存”进行缓存数据清理操作,提示如下图3-4-1所示。

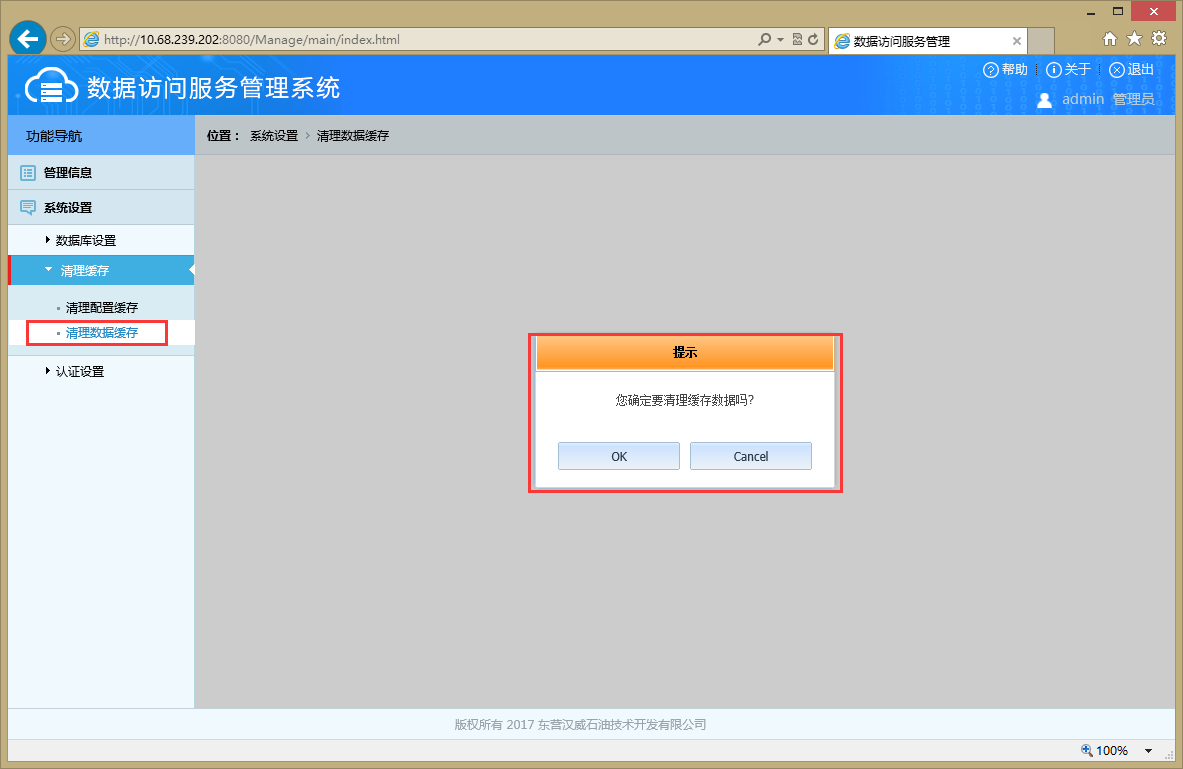


图3-4-1

点击”确认”按钮,清理缓存,如果清理成功,则提示如下图3-4-2所示.

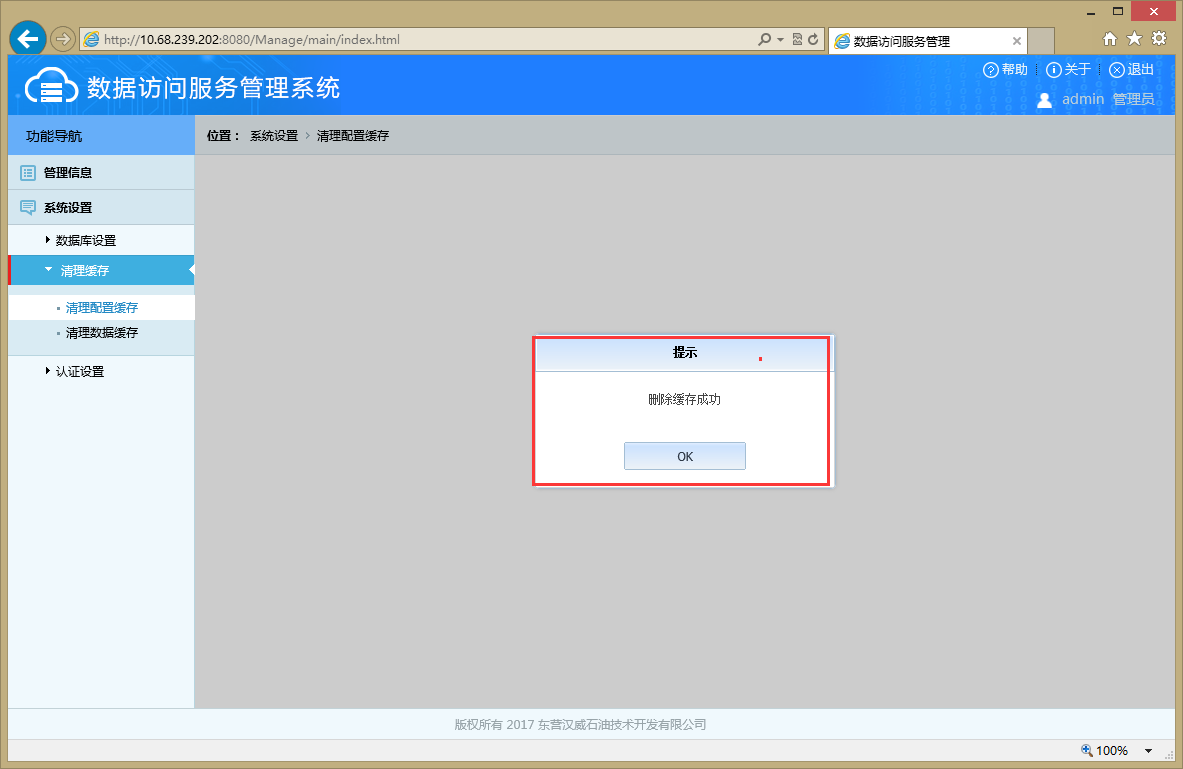


图3-4-2

## 3.5数据库认证设置

数据库认证设置用于设置数据访问服务用户名与密码认证时连接的数据库链接信息,目前一般设置为关联的平台数据库。

如下图3-5-1所示：单击左侧系统设置->认证设置->数据库认证 打开配置界面,根据信息内容填写相应的数据。

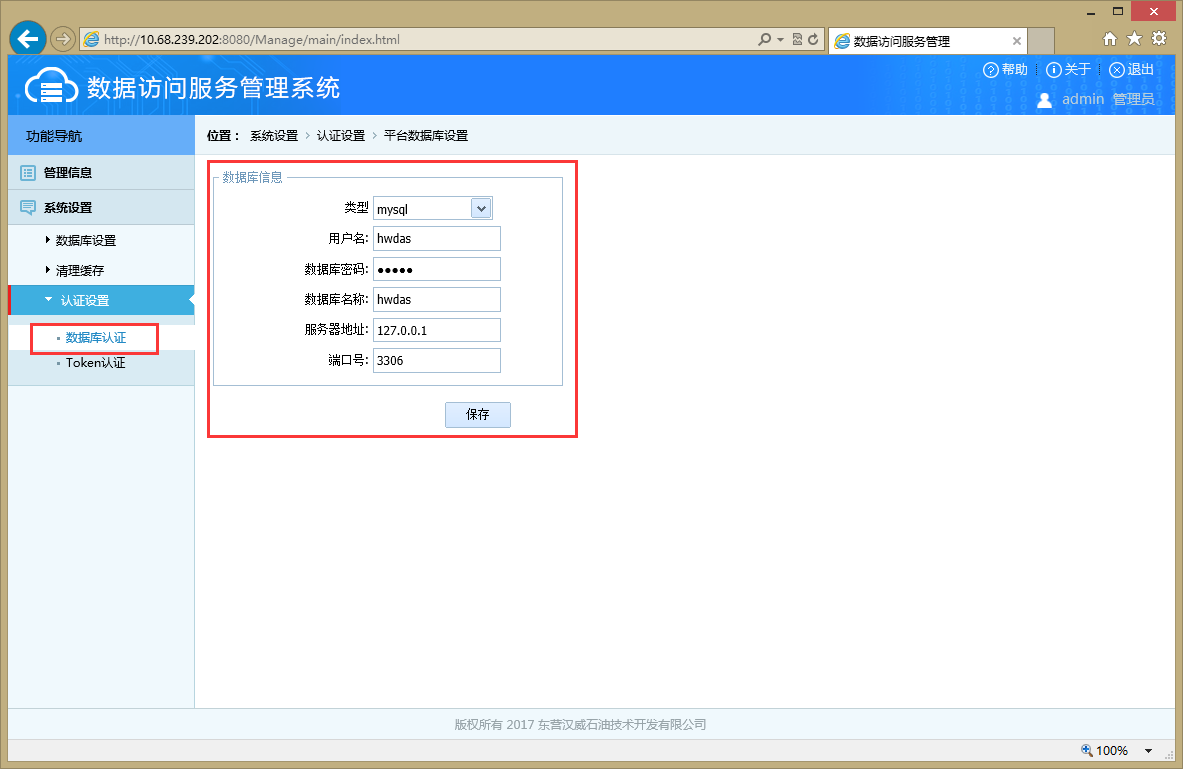


图3-5-1

填写完成后,点击”保存”按钮，进行数据保存，如果保存成功，会有提示，如下图3-5-2所示。

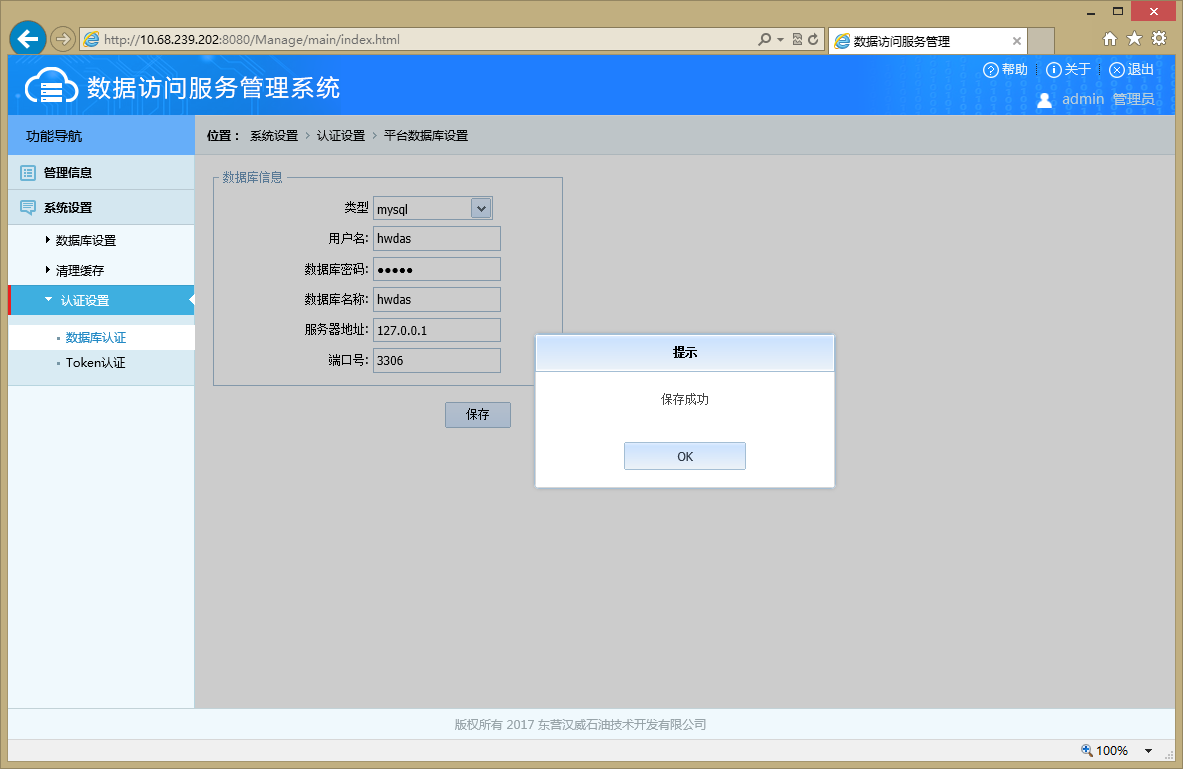


图3-5-2

## 3.6Token认证设置

Token认证配置用于集成平台的Token认证,这里要设置Token认证的地址信息,有还没有与平台集成,暂未实现。

# 附录-sql语句xml描述说明

针对在用的hwsql编辑以及数据访问层出现的种种问题，sql语句变量空条件过滤经常出错和不能预防sql语句注入攻击等问题，重新研发了数据访问层，利用xml格式描述sql语句，动态解析描述生成相应的sql语句，sql语句执行也换成了参数执行方式，解决了sql语句注入攻击，sql语句的描述与编写更灵活。

## Mapper

名称：sql语句描述

说明：Xml文件的根节点描述，是sql语句描述必须的。

属性：引用服务地址using（引用其他服务接口的sql语句到本描述中）

子节点：datasource、select、insert、update、delete、procedure、save、sql、resultMap

示例：

<?xml version="1.0" encoding="utf-8" ?>

<mapper using="url/bb/cc/dd">

<datasource id ="source1" >

source\_28

</datasource>

<datasource id ="source2" >

source\_29

</datasource>

<select id="select-Include1" datasource="source1" resultMap="guserMap">

<include refid="sql9"/>

<if test="var1==10" property="var1" >

and name = @name

</if>

<if test="var1==20" property="var1" >

and id = @id

</if>

</select>

<sql id="sql9" datasource="source1" resultMap="guserMap">

SELECT \* FROM GTUSER where unit like @unit

</sql>

</mapper>

## DataSource

名称：数据源描述

说明：整个描述文件的所有语句用到的数据源集中描述，其中sql语句及存储过程中都会应用该节点描述的数据源id。可以通过左右连接方式描述或是子节点描述动态组合数据源名称。

属性：数据源id、名称连接方式conbintype（左连接left、右连接right）、连接值combinvalue。

子节点：if

示例：

<datasource id ="source1" combinvalue="var1" conbintype="left">

-value

</datasource>

变量var1连接节点内容，如果var1=”a”，则结果为”a-value”。

## Select

名称：查询语句描述

说明：根据描述动态生成查询语句

属性：语句id、数据源datasource、字段转换关系resultMap、二进制转字符串字段byte2strattrs（byte2strattrs\_utf8或是byte2strattrs\_gb2312）、二进制转文件字段byte2fileattrs

子节点：include、tablename、if、where、text

示例：

<select id="selectuser" datasource="source1" resultMap="guserMap">

SELECT \* FROM GTUSER

</select>

## Scalar

名称：查询语句描述

说明：根据描述动态生成查询语句

属性：语句id、数据源datasource、回填变量property、

子节点：无

示例：

<scalar id="selectuser" datasource="source1" property="var1">

SELECT ‘123’ as var1 FROM dual

</scalar>

首先执行scalar 语句,回填变量值,然后再执行其他语句

## Insert

名称：插入语句描述

说明：根据描述动态生成插入语句

属性：语句id、数据源datasource、字段转换关系resultMap

子节点：include、tablename、if、text

示例：

<insert id="insertuser" datasource="source1" resultMap="guserMap">

insert GTUSER (id,login,name) values (@id,@login,@name)

</insert>

## Update

名称：修改语句描述

说明：根据描述动态生成修改语句

属性：语句id、数据源datasource、字段转换关系resultMap

子节点：include、tablename、if、where、text、set

示例：

<update >

update GTUSER

<set>

<if type="isNotEmpty" property="name">

name=@name,

</if>

<if type="isNotEmpty" property="password">

password=@password,

</if>

<if type="isNotEmpty" property="email">

email=@email,

</if>

</set>

<where>

<if type="isNotEmpty" property="id" >

id = @id

</if>

</where>

</update>

## Delete

名称：删除语句描述

说明：根据描述动态生成删除语句

属性：语句id、数据源datasource、字段转换关系resultMap

子节点：include、tablename、if、where、text

示例：

<delete id="delete-test" datasource="source1" resultMap="guserMap">

delete FROM GTUSER

<where>

<if test="var1==10" property="var1" >

and name = @name

</if>

</where>

</delete>

## Save

名称：更新数据集语句描述

说明：根据描述动态处理对应语句

属性：语句id、字段转换关系resultMap

子节点：check、insert、update、delete

示例：

<save id="select-isNotEmpty" resultMap="guserMap">

<check>

SELECT \* FROM GTUSER where a=@a

</check>

<insert >

insert GTUSER (id,login,name) values (@id,@login,@name)

</insert>

<update >

update GTUSER

<set>

<if type="isNotEmpty" property="name">

name=@name,

</if>

<if type="isNotEmpty" property="password">

password=@password,

</if>

<if type="isNotEmpty" property="email">

email=@email,

</if>

</set>

<where>

<if type="isNotEmpty" property="id" >

id = @id

</if>

</where>

</update>

<delete >

delete FROM GTUSER where id = @id

</delete>

</save>

## procedure

名称：存储过程描述

说明： 动态生成执行存储过程名称

属性：语句id、数据源datasource

子节点：parameter

示例：

<procedure id="procedure-test" datasource="source1">

procedurename

<parameter name="id" type="String" property="id">

id

</parameter>

<parameter name="login" type="String" property="login">

login

</parameter>

<parameter name="name" type="String" property="name">

name

</parameter>

</procedure>

## resultMap

名称：字段对应关系描述

说明： 数据集字段或是对象属性转换关系，包括名称和类型

属性：语句id

子节点：result

示例：

<resultMap id="guserMap">

<result property="id" column="id" />

<result property="login" column="login" />

<result property="name" column="name" />

<result property="dw" column="dw" />

</resultMap>

## sql

名称：普通sql语句描述

说明：普通sql语句描述

属性：语句id

子节点：

示例：

<sql id="sql9">

SELECT \* FROM GTUSER where unit like @unit

</sql>

## check

名称：检查语句描述

说明：检查数据是否存在的数据查询语句

属性：语句id、数据源datasource

子节点：

示例：

<check>

SELECT \* FROM GTUSER where a=@a

</check>

## Text

名称：sql部分文本内容

说明： 描述sql语句部分文本内容

属性：

子节点：

示例：

<select id="select-Foreach" datasource="source1" resultMap="guserMap">

SELECT \* FROM GTUSER

<where>

<if test="var1==10" property="var1" >

and name = @name

</if>

<if type="isNotNull" property="loginList" >

and login in

<foreach property="loginList" separator = "','" open="('" close="')">

{0}

</foreach>

</if>

</where>

<text>

order by login

</text>

</select>

## Include

名称：引用语句id

说明： 可以直接应用已定义的sql节点的语句

属性：关联sql编码refid

子节点：

示例：

<select id="select-Include2" datasource="source1" resultMap="guserMap">

<include refid="sql9"/>

<if test="var1==10" property="var1" >

and name = @name

</if>

</select>

## Tablename

名称：表名称

说明： 动态组织表名称，从而支持数据分表功能

属性：关联变量property、连接值combinvalue、连接方式conbintype（left或right）

子节点：if、where、text

示例：

<select id="tablename-combin-left" datasource="source1" resultMap="guserMap">

SELECT \* FROM

<!--左连接-->

<tablename combinvalue="var1.Substring(0,2)" conbintype="left" property="var1">

-GTUSER

</tablename>

</select>

如果变量var1=“abcd”，那么表名组合后就是“ab-GTUSER”

## Where

名称：条件描述

说明： 查询、修改、删除等语句的条件

属性：是否允许为空allowemply（true 或是false）

子节点：if、text、foreach

示例：

<select id="select-isNotEmpty" datasource="source1" resultMap="guserMap">

SELECT \* FROM GTUSER

<where allowemply="false">

<if type="isNotEmpty" property="id" >

and id = @id

</if>

</where>

</select>

## Set

名称：更新语句set标识

说明：更新语句set标识

属性：

子节点：if、text、foreach

示例：

<update id="update-set" datasource="source1" resultMap="guserMap">

update GTUSER

<set>

<if type="isNotEmpty" property="name">

name=@name,

</if>

<if type="isNotEmpty" property="password">

password=@password,

</if>

<if type="isNotEmpty" property="email">

email=@email,

</if>

</set>

<where>

<if type="isNotEmpty" property="id" >

id = @id

</if>

</where>

</update>

## If

名称：条件判断

说明：条件判断

属性：关联变量property、操作类型type（值包括IsEmpty, IsNotEmpty, IsNull, IsNotNull, IsConvey, IsNotConvey, IsLessThan, IsGreaterEqual, IsGreaterThan, IsLessEqual, StartsWith, EndsWith, Contains）、操作值compareValue (不能与test一起使用)、表达式test (不能与type、compareValue一起使用)

子节点：if、text、foreach

示例：

<select id="select-isNotEmpty" datasource="source1" resultMap="guserMap">

SELECT \* FROM GTUSER

<where allowemply="false">

<if type="isNotEmpty" property="id" >

and id = @id

</if>

<if type="IsNotNull" property="login" >

and login = @login

</if>

<if test="var1==10" property="var1" >

and name like '%'|| @var1 || '%'

</if>

<if test="var2==&quot; abc&quot; " property="var1" >

and name like '%'|| @var1 || '%'

</if>

</where>

</select>

类型说明

IsEmpty :变量为空或是null

IsNotEmpty: 变量不为null也不为空

IsNull:变量为null

IsNotNull:变量不为null

IsConvey:变量是传递过来的

IsNotConvey:变量不是传递过来的

IsLessThan:变量小于compareValue设置的值

IsGreaterEqual:变量大于等于compareValue设置的值

IsGreaterThan: 变量大于compareValue设置的值

IsLessEqual: 变量小于等于compareValue设置的值

StartsWith: 变量开始位置匹配compareValue设置的值

EndsWith:变量末尾位置匹配compareValue设置的值

Contains变量包含: compareValue设置的值

## Foreach

名称：in数组处理

说明： 循环处理变量，生成语句

属性：连接字符separator、关闭符close、开始符open、变量property

子节点：

示例：

<select id="select-Foreach" datasource="source1" resultMap="guserMap">

SELECT \* FROM GTUSER

<where>

<if type="isNotNull" property="loginList" >

<foreach property="loginList" separator = "','" open="('" close="')">

login in {0}

</foreach>

</if>

</where>

</select>

如果变量loginlist为数组{“login1”，“login2”}那么sql语句为“SELECT \* FROM GTUSER where login in ('login1','login2')”

如果数组超过1000后 ,会自动处理成,处理后的结果为: “SELECT \* FROM GTUSER where （login in ('login1','login2',……) or login in ('login1000','login1001',……)）”

## For

名称：循环执行

说明： 根据数组变量循环执行内部语句

属性：循环的变量property、遍历后的元素名称itemname

子节点：insert、update、delete、procedure

示例：

<for id="insertfor" property="varList" itemname="itemName" >

<insert id="insert-test" datasource="source1" resultMap="guserMap">

insert GTUSER (id,login,name,item) values (@id,@login,@name,@itemName)

</insert>

<insert id="insert-test2" datasource="source1" resultMap="guserMap">

insert GTUSER1 (id,login,name,item) values (@id,@login,@name,@itemName)

</insert>

</for>

如果变量varList为数组{“login1”，“login2”}那么执行的sql语句为：

insert GTUSER (id,login,name,item) values (@id,@login,@name,’ login1’)

insert GTUSER1 (id,login,name,item) values (@id,@login,@name, ’ login1’)

insert GTUSER (id,login,name,item) values (@id,@login,@name,’ login2’)

insert GTUSER1 (id,login,name,item) values (@id,@login,@name, ’ login2’)

## parameter

名称：过程参数

说明： 存储过程参数信息

属性：属性property、名称name、是否返回值needreturn（true或false）

子节点：if、text、foreach

示例：

<procedure id="procedure-test" datasource="source1" resultMap="guserMap" >

procedurename

<parameter name="id" type="String" property="id">

id

</parameter>

<parameter name="login" type="String" property="loginname">

loginname

</parameter>

<parameter name="name" type="String" property="name">

name

</parameter>

</procedure>

# 附录-sql描述示例1-数据源

<?xml version="1.0" encoding="utf-8" ?>

<mapper >

<!--左连接-->

<datasource id ="source-combin-left" combinvalue="var1" conbintype="left">

-value

</datasource>

<!--右连接-->

<datasource id ="source-combin-right" combinvalue="var1" conbintype="right">

value-

</datasource>

<!--条件类型-->

<!--IsEmpty、IsNotEmpty类型-->

<datasource id ="source-if-IsEmpty" >

prefixStr-

<if property="var1" type="IsEmpty">str-IsEmpty</if>

<if property="var1" type="IsNotEmpty">str-IsNotEmpty</if>

</datasource>

<!--IsNull、IsNotNull类型-->

<datasource id ="source-if-IsNull" >

prefixStr-

<if property="var1" type="IsNull">str-IsNull</if>

<if property="var1" type="IsNotNull">str-IsNotNull</if>

</datasource>

<!--IsLessEqual、IsLessThan类型-->

<datasource id ="source-if-Less" >

prefixStr-

<if property="var1" type="IsLessThan" compareValue="10">str-IsLessThan</if>

<if property="var1" type="IsGreaterEqual" compareValue="20">str-IsLessEqual</if>

</datasource>

<!--IsGreaterThan、IsGreaterEqual类型-->

<datasource id ="source-if-Greater" >

prefixStr-

<if property="var1" type="IsGreaterThan" compareValue="15">str-IsGreaterThan</if>

<if property="var1" type="IsLessEqual" compareValue="10">str-IsGreaterEqual</if>

</datasource>

<!--StartsWith类型-->

<datasource id ="source-if-StartsWith" >

prefixStr-

<if property="var1" type="StartsWith" compareValue="start">str-StartsWith</if>

</datasource>

<!--EndsWith类型-->

<datasource id ="source-if-EndsWith" >

prefixStr-

<if property="var1" type="EndsWith" compareValue="end">str-EndsWith</if>

</datasource>

<!--Contains类型-->

<datasource id ="source-if-Contains" >

prefixStr-

<if property="var1" type="Contains" compareValue="str">str-Contains</if>

</datasource>

<!--OtherEL类型-->

<datasource id ="source-if-OtherEL" >

prefixStr-

<if property="var1" test="var1==10">str-10</if>

<if property="var1" test="var1!=10">str-other</if>

</datasource>

<datasource id ="source-Choose" >

prefixStr-

<choose>

<when type="StartsWith" property="var1" compareValue="start">

str-StartsWith

</when>

<when test="var1==&quot;10&quot;" property="var1">

str-10

</when>

<otherwise>

str-ELSE

</otherwise>

</choose>

</datasource>

</mapper>

# 附录-sql描述示例2-语句

<?xml version="1.0" encoding="utf-8" ?>

<mapper >

<datasource id ="source1" >

sourceStr

</datasource>

<!--select-isNotEmpty-->

<select id="select-isNotEmpty" datasource="source1" resultMap="guserMap">

SELECT \* FROM GTUSER

<where allowemply="false">

<if type="isNotEmpty" property="id" >

and id = @id

</if>

<if type="IsNotNull" property="login" >

and login = @login

</if>

</where>

</select>

<!--select-IsGreaterEqual-IsLessEqual-->

<select id="select-GreaterEqual" datasource="source1" resultMap="guserMap">

SELECT \* FROM

<tablename>

GTUSER

</tablename>

<where>

<if type="IsGreaterEqual" property="var1" compareValue="30" >

and id = @id

</if>

<if type="IsLessEqual" property="var1" compareValue="20" >

and login = @login

</if>

</where>

</select>

<!--select-StartsWith-Contains-->

<select id="select-StartsWith" datasource="source1" resultMap="guserMap">

SELECT \* FROM GTUSER

<where>

<if type="StartsWith" property="var1" compareValue="cs" >

and id = @id

</if>

<if type="Contains" property="var1" compareValue="test" >

and login in (@var1)

</if>

</where>

</select>

<!--select-OtherEL-if-->

<select id="select-OtherEL" datasource="source1" resultMap="guserMap">

SELECT \* FROM GTUSER

<where>

<if test="var1==10" property="var1" >

and name like '%'|| @var1 || '%'

</if>

<if test="var1==20" property="var1" >

and id = @id

</if>

</where>

</select>

<!--select-where-choose-->

<select id="select-Choose" datasource="source1" resultMap="guserMap">

SELECT \* FROM GTUSER

<where>

<choose>

<when type="isNotEmpty" property="login">

and login = @login

</when>

<when test="nameLike!=null" property="nameLike">

and name like '%'||@nameLike||'%'

</when>

<otherwise>

and unit like @unit

</otherwise>

</choose>

</where>

</select>

<!--select-Foreach-->

<select id="select-Foreach" datasource="source1" resultMap="guserMap">

SELECT \* FROM GTUSER

<where>

<if test="var1==10" property="var1" >

and name = @name

</if>

<if type="isNotNull" property="loginList" >

<foreach property="loginList" separator = "','" open="('" close="')">

and login in {0}

</foreach>

</if>

</where>

<text>

order by login

</text>

</select>

<!--select-include1-->

<select id="select-Include1" datasource="source1" resultMap="guserMap">

<include refid="sql9"/>

<if test="var1==10" property="var1" >

and name = @name

</if>

<if test="var1==20" property="var1" >

and id = @id

</if>

</select>

<!--select-include2-->

<select id="select-Include2" datasource="source1" resultMap="guserMap">

<include refid="sql9"/>

<if test="var1==10" property="var1" >

and name = @name

</if>

<if test="var1==20" property="var1" >

and id = @id

</if>

</select>

<sql id="sql9" datasource="source1" resultMap="guserMap">

SELECT \* FROM GTUSER where unit like @unit

</sql>

<!--update-set-->

<update id="update-set" datasource="source1" resultMap="guserMap">

update GTUSER

<set>

<if type="isNotEmpty" property="name">

name=@name,

</if>

<if type="isNotEmpty" property="password">

password=@password,

</if>

<if type="isNotEmpty" property="email">

email=@email,

</if>

</set>

<where>

<if type="isNotEmpty" property="id" >

id = @id

</if>

</where>

</update>

<!--insert-->

<insert id="insert-test" datasource="source1" resultMap="guserMap">

insert GTUSER (id,login,name) values (@id,@login,@name)

</insert>

<!--delete-->

<delete id="delete-test" datasource="source1" resultMap="guserMap">

delete FROM GTUSER

<where>

<if test="var1==10" property="var1" >

and name = @name

</if>

</where>

</delete>

<!--procedurename(id,login,name)-->

<procedure id="procedure-test" datasource="source1" resultMap="guserMap" >

procedurename

<parameter name="id" type="String" property="id">

id

</parameter>

<parameter name="login" type="String" property="login">

login

</parameter>

<parameter name="name" type="String" property="name">

name

</parameter>

</procedure>

<save id="select-isNotEmpty" >

<check>

SELECT \* FROM GTUSER where a=@a

</check>

<insert >

insert GTUSER (id,login,name) values (@id,@login,@name)

</insert>

<update >

update GTUSER

<set>

<if type="isNotEmpty" property="name">

name=@name,

</if>

<if type="isNotEmpty" property="password">

password=@password,

</if>

<if type="isNotEmpty" property="email">

email=@email,

</if>

</set>

<where>

<if type="isNotEmpty" property="id" >

id = @id

</if>

</where>

</update>

<delete >

delete FROM GTUSER where id = @id

</delete>

</save>

<resultMap id="guserMap">

<result property="id" column="id" />

<result property="login" column="login" />

<result property="name" column="name" />

<result property="dw" column="dw" />

</resultMap>

</mapper>

# 附录-sql描述示例3-表名

<?xml version="1.0" encoding="utf-8" ?>

<mapper>

<datasource id ="source1" >

sourceStr

</datasource>

<!--select-left- combin-->

<select id="tablename-combin-left" datasource="source1" resultMap="guserMap">

SELECT \* FROM

<!--左连接-->

<tablename combinvalue="var1.Substring(0,2)" conbintype="left" property="var1">

-GTUSER

</tablename>

</select>

<!--右连接-->

<select id="tablename-combin-right" datasource="source1" resultMap="guserMap">

SELECT \* FROM

<tablename combinvalue="var1+&quot;test&quot;" conbintype="right" property="var1">

GTUSER-

</tablename>

</select>

<!--条件类型-->

<!--IsEmpty、IsNotEmpty类型-->

<select id="tablename-if-IsEmpty" datasource="source1" resultMap="guserMap">

SELECT \* FROM

<tablename>

GTUSER-

<if property="var1" type="IsEmpty">str-IsEmpty</if>

<if property="var1" type="IsNotEmpty">str-IsNotEmpty</if>

</tablename>

</select>

<!--IsNull、IsNotNull类型-->

<select id="tablename-if-IsNull" datasource="source1" resultMap="guserMap">

SELECT \* FROM

<tablename>

GTUSER-

<if property="var1" type="IsNull">str-IsNull</if>

<if property="var1" type="IsNotNull">str-IsNotNull</if>

</tablename>

</select>

<!--IsGreaterEqual、IsLessThan类型-->

<select id="tablename-if-Greater" datasource="source1" resultMap="guserMap">

SELECT \* FROM

<tablename>

GTUSER-

<if property="var1" type="IsGreaterEqual" compareValue="10">str-IsGreaterEqual</if>

<if property="var1" type="IsLessThan" compareValue="10">str-IsLessThan</if>

</tablename>

</select>

<!--IsLessEqual、IsGreaterThan类型-->

<select id="tablename-if-Less" datasource="source1" resultMap="guserMap">

SELECT \* FROM

<tablename>

GTUSER-

<if property="var1" type="IsLessEqual" compareValue="10">str-IsLessEqual</if>

<if property="var1" type="IsGreaterThan" compareValue="10">str-IsGreaterThan</if>

</tablename>

</select>

<!--StartsWith类型-->

<select id="tablename-if-StartsWith" datasource="source1" resultMap="guserMap">

SELECT \* FROM

<tablename>

GTUSER-

<if property="var1" type="StartsWith" compareValue="start">str-StartsWith</if>

</tablename>

</select>

<!--EndsWith类型-->

<select id="tablename-if-EndsWith" datasource="source1" resultMap="guserMap">

SELECT \* FROM

<tablename>

GTUSER-

<if property="var1" type="EndsWith" compareValue="end">str-EndsWith</if>

</tablename>

</select>

<!--Contains类型-->

<select id="tablename-if-Contains" datasource="source1" resultMap="guserMap">

SELECT \* FROM

<tablename>

GTUSER-

<if property="var1" type="Contains" compareValue="str">str-Contains</if>

</tablename>

</select>

<!--OtherEL类型-->

<select id="tablename-if-OtherEL" datasource="source1" resultMap="guserMap">

SELECT \* FROM

<tablename>

GTUSER-

<if property="var1" test="var1==10">str-10</if>

<if property="var1" test="var1!=10">str-other</if>

</tablename>

</select>

<select id="tablename-Choose" datasource="source1" resultMap="guserMap">

SELECT \* FROM

<tablename>

GTUSER-

<choose>

<when type="IsNotEmpty" property="var1">

str-isNotEmpty

</when>

<when type="StartsWith" property="var1" compareValue="start">

str-isNotEmpty

</when>

<when test="var2=='10'" property="var2">

str-10

</when>

<otherwise>

str-ELSE

</otherwise>

</choose>

</tablename>

</select>

</mapper>