产品名称

Vast Data Studio详细设计说明书

版本： <5.0.0>

|  |  |
| --- | --- |
| 编号 | HLSJ\_Vastbase-M100-VDS\_V5.0\_XXSJ\_01 |
| 密级 |  |
| 作者 | 陈康柳 |
| 编制/修订日期 | 2022/09月/20 |
| 审核 | 李洁 |
| 批准 | 李洁 |
| 质量会签 |  |
| 标准化会签 |  |

改版记录

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **版本** | **发布日期** | **描述** | **作者** | **复审** | **批准** |
| 1.0.0 | 2022-08-10 | vds 1.0 版本 | 陈康柳 | 李洁 | 李洁 |

变更记录

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **变更编号** | **日期** | **变更项** | **描述** | **基线版本** | **变更请求编号** |
| 1 | 2022-09-20 |  | vds 5.0.0 | 1.0.0 |  |

目录

1. 范围 6

1.1 标识 6

1.2 系统概述 6

1.3 文档概述 6

2. 术语定义、首字母缩写词和缩略语 6

3. 引用文档 6

4. 系统级设计决策 6

5. 系统体系结构设计 6

5.1 数据库连接管理 6

5.1.1 功能简述 6

5.1.2 系统部件 7

5.1.3 执行方案 9

5.2 数据库对象管理 13

5.2.1 功能简述 13

5.2.2 系统部件 14

5.2.3 执行方案 15

5.3 模式对象管理 19

5.3.1 功能简述 20

5.3.2 系统部件 20

5.3.3 执行方案 20

5.4 函数/存储过程对象管理 23

5.4.1 功能简述 23

5.4.2 系统部件 24

5.4.3 执行方案 24

5.5 表对象管理 26

5.5.1 功能简述 27

5.5.2 系统部件 28

5.5.3 执行方案 29

5.6 列对象管理 37

5.6.1 功能简述 37

5.6.2 系统部件 37

5.6.3 执行方案 38

5.7 约束对象管理 40

5.7.1 功能简述 41

5.7.2 系统部件 41

5.7.3 执行方案 42

5.8 索引对象管理 44

5.8.1 功能简述 44

5.8.2 系统部件 45

5.8.3 执行方案 45

5.9 视图对象管理 47

5.9.1 功能简述 48

5.9.2 系统部件 48

5.9.3 执行方案 48

5.10 序列对象管理 51

5.10.1 功能简述 52

5.10.2 系统部件 52

5.10.3 执行方案 53

5.11 模式对象管理 56

5.11.1 功能简述 56

5.11.2 系统部件 56

5.11.3 执行方案 57

5.12 同义词对象管理 59

5.12.1 功能简述 59

5.12.2 系统部件 60

5.12.3 执行方案 60

5.13 触发器对象管理 62

5.13.1 功能简述 62

5.13.2 系统部件 63

5.13.3 执行方案 63

5.14 表空间对象管理 65

5.14.1 功能简述 65

5.14.2 系统部件 66

5.14.3 执行方案 66

5.15 用户/角色对象管理 68

5.15.1 功能简述 69

5.15.2 系统部件 69

5.15.3 执行方案 70

5.16 SQL终端编辑器 73

5.16.1 功能简述 73

5.16.2 系统部件 73

5.16.3 执行方案 73

5.17 对象搜索 76

5.17.1 功能简述 76

5.17.2 系统部件 76

5.17.3 执行方案 76

5.18 运行/调试 78

5.18.1 功能简述 78

5.18.2 系统部件 78

5.18.3 执行方案 78

详细设计说明书

（Vast Data Studio）

# 范围

## 标识

名称：Vast Data Studio

缩略名：VDS

版本号：5.0.0

## 系统概述

VDS（Vast Data Studio）的设计主要是用于海量数据所有自研数据库的管理与开发，以图形化界面形式提供数据库管理能力与开发能力，帮助数据库管理人员与开发人员便捷地对数据库对象进行管理以及构建应用程序，数据库开发人员仅需掌握少量的编程知识，即可使用该工具进行数据库对象操作，提高海量数据自研数据库的易用性。

## 文档概述

本文档用于Vast Data Studio的技术详细设计方案落地的实现。

方案设计覆盖数据库连接、对象管理、函数/存储过程调试和SQL终端编辑功能点，描述功能实现的边界、交互逻辑和关键点。

# 术语定义、首字母缩写词和缩略语

* **SSL**

Secure Socket Layer，安全网络传输协议，对传输数据进行加密；

* **Jdbc**

java数据库驱动，用于连接数据库，传输客户端指令；

* **对象管理**

对象涵盖了数据库对象、表对象、函数对象、约束对象等一系列数据库角色；

# 引用文档

《GB/T 9385-2008 计算机软件需求规格说明规范》

参考vastdata官网资料

https://docs.vastdata.com.cn/zh/

# 系统级设计决策

参考概要设计文档《VDS架构设计说明书.docx》

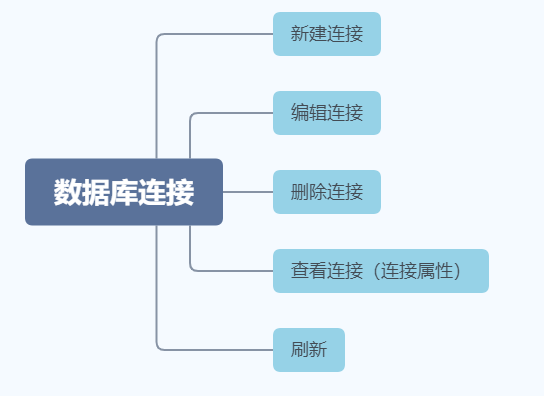
# 系统体系结构设计

## 数据库连接管理

### **功能简述**

客户端工具通过IP、端口、用户名、密码登录数据库，与数据库建立远程连接。

连接管理功能点，如下图所示



* **新建/删除连接**

客户端通过可视化界面，填写IP、端口、用户名、密码等信息，建立与数据库的连接。

* **导入/导出配置文件**

客户端与数据库建立连接后，允许把连接信息导出生成本地文件保存；后续允许通过导入本地文件自动填充连接信息。

### **系统部件**

#### **主要程序文件**

#### **关键数据结构说明**

* **连接信息字典**

|  |  |  |  |
| --- | --- | --- | --- |
| **字典** | **说明** | **类型** | **是否必填** |
| 数据库类型 | 连接的数据库类型，目前支持Vastbase M100、Vastvase G100 | 基本信息 | 是 |
| 名称 | 连接实例自定义名称 | 基本信息 | 是 |
| 主机 | 数据库服务器IP地址 | 基本信息 | 是 |
| 端口号 | 数据库服务器连接端口号 | 基本信息 | 是 |
| 数据库 | 数据库实例名称 | 基本信息 | 是 |
| 用户名 | 数据库登录用户名 | 基本信息 | 是 |
| 密码 | 数据库登录密码 | 基本信息 | 是 |
| 客户端SSL证书 | 客户端SSL证书本地路径 | SSL | 否 |
| 客户端SSL密钥 | 客户端SSL密钥本地路径 | SSL | 否 |
| 根证书 | 根证书本地路径 | SSL | 否 |
| SSL密码 | SSL密码，生成证书时指定 | SSL | 否 |
| SSL模式 | verify/require/verify-ca/allow | SSL | 否 |
| 加载选项 | 对象浏览树展示内容筛选展示，包括：模式、加载内容、加载上限数量 | 配置项 | 否 |

### **执行方案**

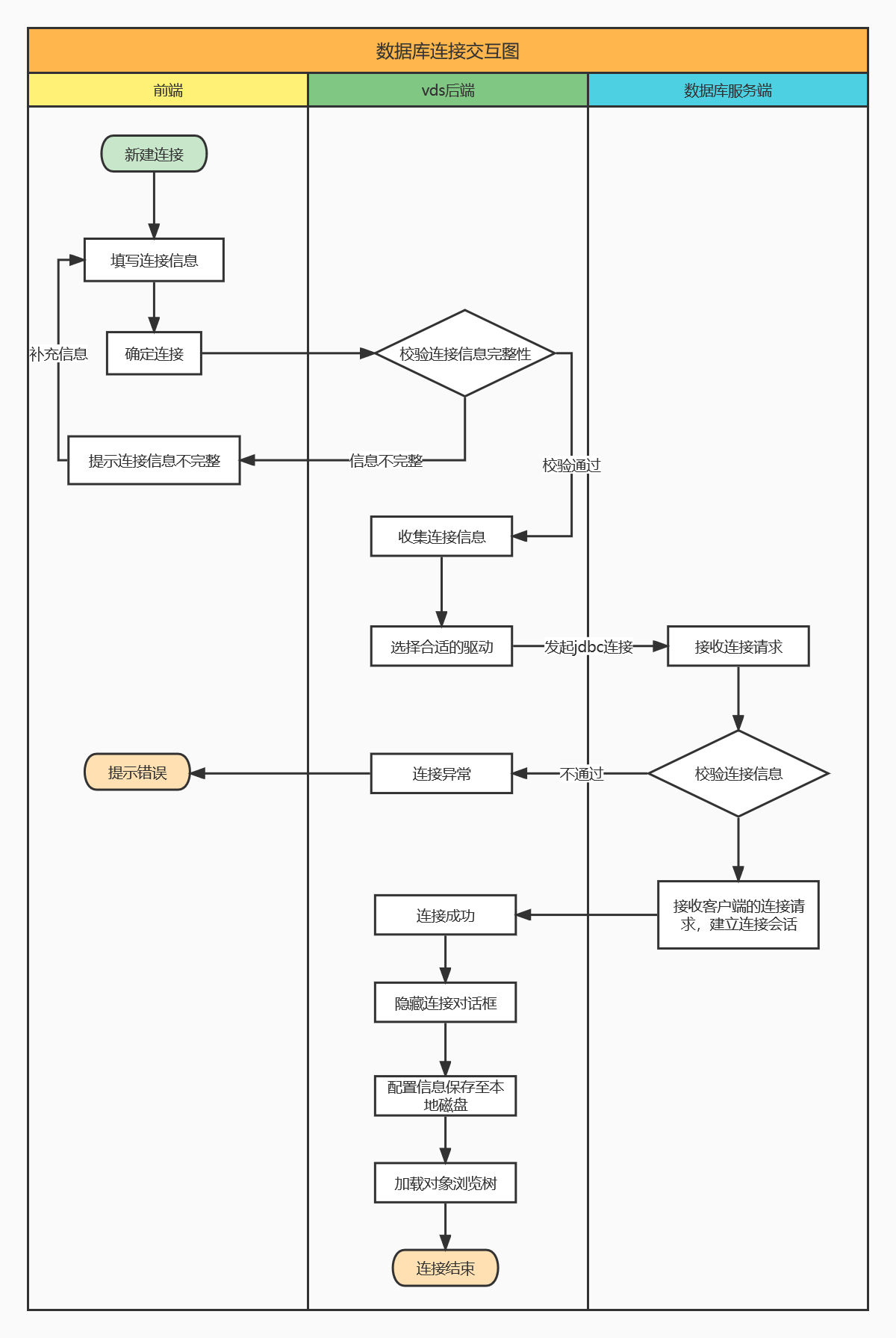
#### **数据连接对象管理**

##### **实现逻辑描述**

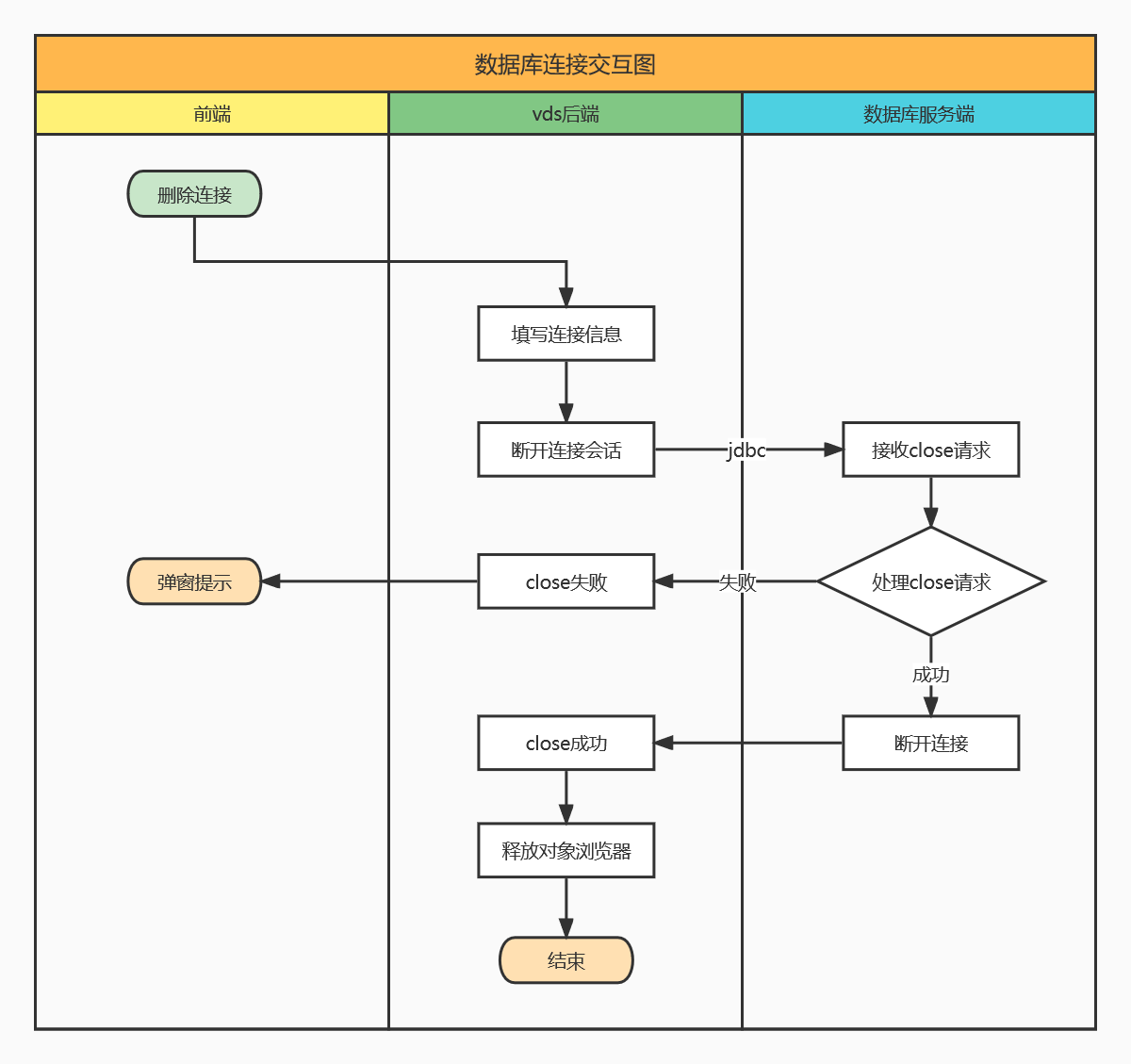
1. 根据用户选择的数据库类型，选择对应jdbc驱动，使用jdbc api发起数据库连接；
2. 建立完成数据库连接后，对连接进行实例化，连接实例信息保存至客户端的本地磁盘工作空间；
3. 删除本地连接时，后台对连接会话，调起jdbc close接口，关闭所有的连接会话；

##### **流程/交互设计**

* **新建连接交互如图所示**

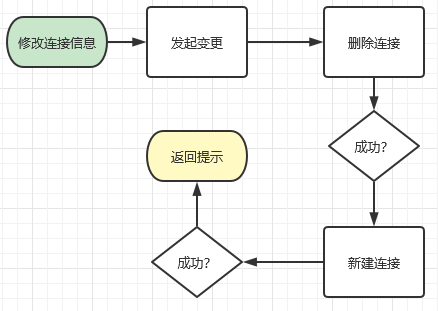


* **删除连接交互如图所示**

****

* **编辑连接**

编辑连接的流程，结合了新建连接和删除连接;



* **属性查看**

读取展示内存对象属性；

* **连接刷新**

对连接下的jdbc会话进行close，并重新连接。

##### **异常场景**

* 连接信息错误：基本信息错误，ssl证书错误；
* 网络异常：客户端与数据库服务器网络不通，或者，数据库服务禁止目标机器访问；
* 数据库异常：数据库未启动；

##### **其它**

* **关于连接实例和会话**

连接实例并不等同于jdbc会话，两者的关系可以简单描述为:

连接实例的作用是限制会话的连接信息、权限，但不限制基于该连接信息、权限下，构建多少个jdbc会话。

* **关于session\_timeout**

由于数据库默认的session\_timeout只有60s，出于用户体验考虑，vds对于jdbc连接会清除session\_timeout的限制，该操作只对vds当前连接生效。

#### **数据库对象管理**

##### **主要算法描述**

简要说明算法实现思路。

##### **流程/交互设计**

##### **异常处理**

异常处理说明。

##### **其它**

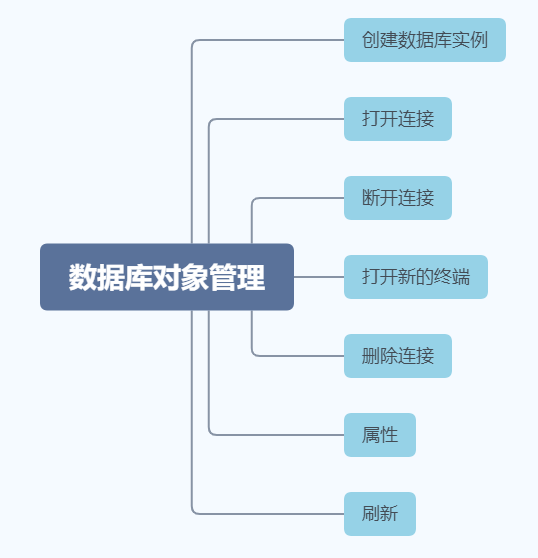
无

## 数据库对象管理

### **功能简述**

新建连接成功后，在对象浏览器树展示该连接下的所有数据库实例（在用户权限范围内）。

用户可以通过右键菜单的方式来管理数据库实例连接。



### **系统部件**

#### **主要程序文件**

#### **关键数据结构说明**

* **创建数据库**

|  |  |  |  |
| --- | --- | --- | --- |
| **字典** | **说明** | **类型** | **是否必填** |
| 数据库名称 | 输入数据库实例的名称 | 基本信息 | 是 |
| 字符编码 | UTF-8、GBK、LATIN1、SQL\_ASCII | 基本信息 | 是 |

### **执行方案**

#### **数据连接对象管理**

##### **实现逻辑描述**

* **创建数据库实例**

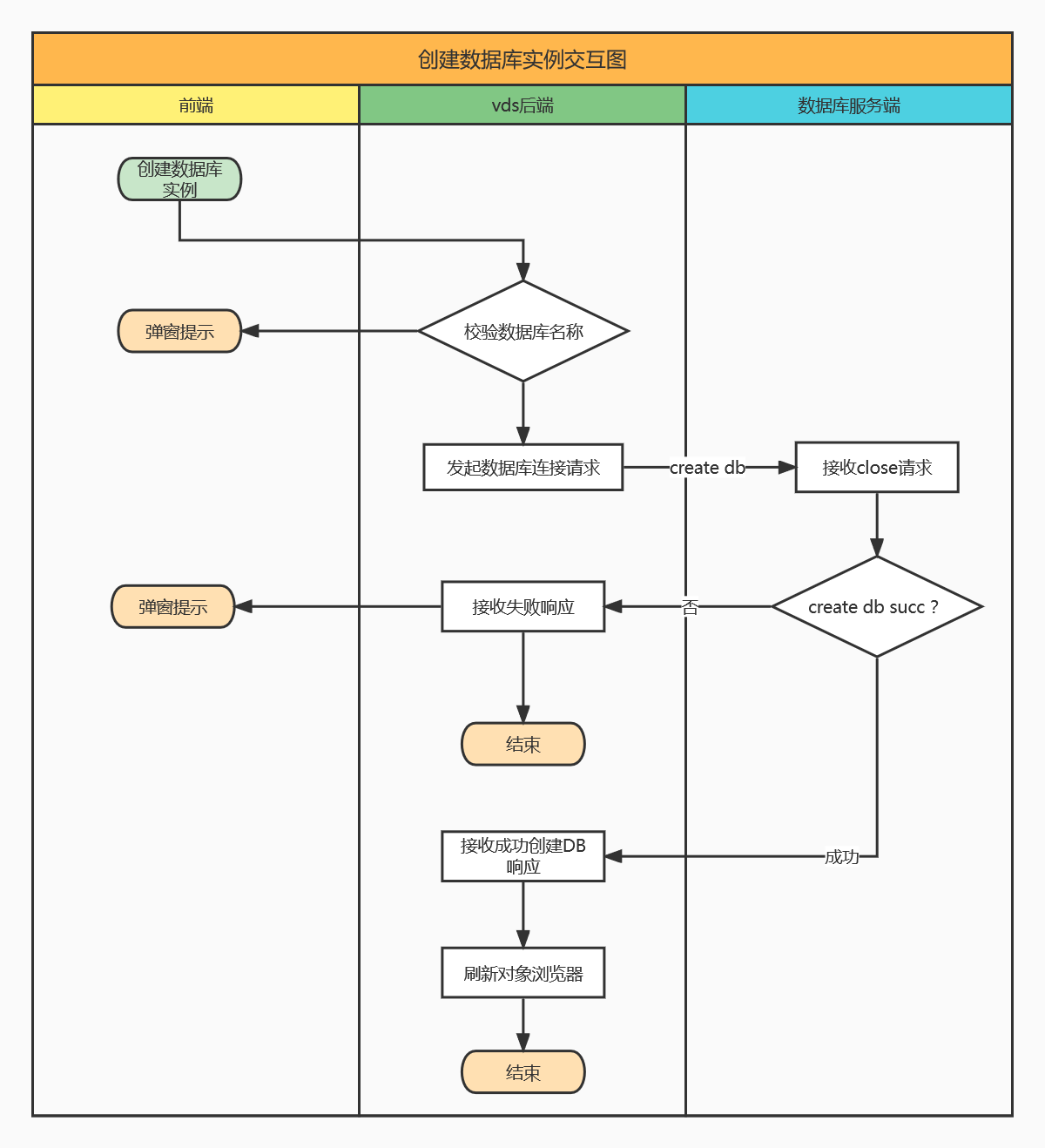
通过数据库连接，调起jdbc接口，发起数据库实例创建请求。创建成功后，重新加载数据库组展示。

* **打开连接/删除连接/刷新连接**

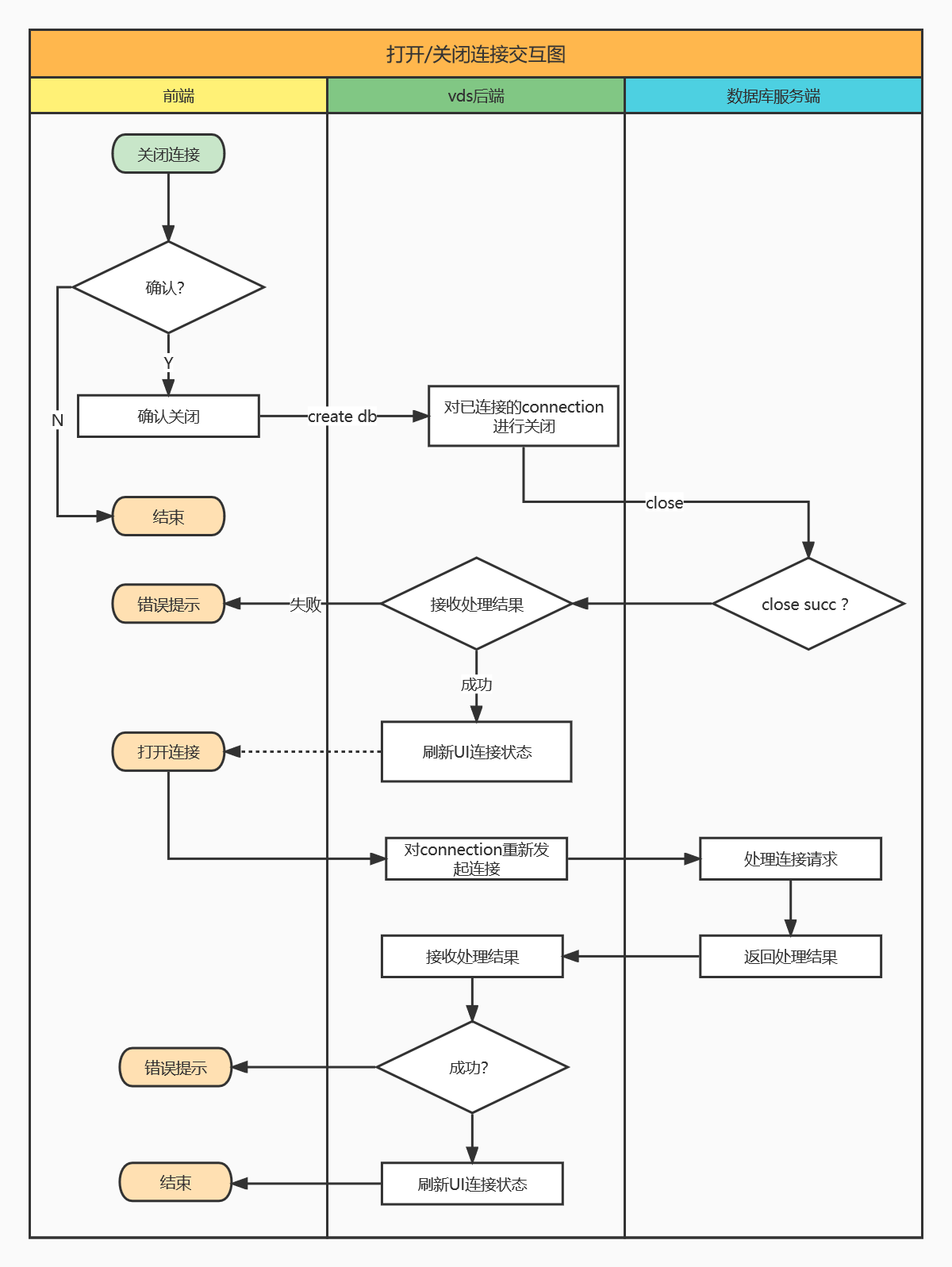
对jdbc连接进行close、connect。

##### **流程/交互设计**

* **创建数据库实例**

****

* **打开/关闭连接**



* **连接属性**

从关键表pg\_database中读取属性信息。

Select oid, pg\_encoding, datconnlimit, datcollate, datatype,

(select spcname from pg\_tablespace where oid = dattablespace) as default\_tablespace

from pg\_database;

* **刷新连接**

对数据库下面的connection进行关闭和重新打开。

##### **关键算法/SQL**

* 关键SQL

Create database dbname

encoding “UTF-8”

template = ‘template0’;

##### **其它**

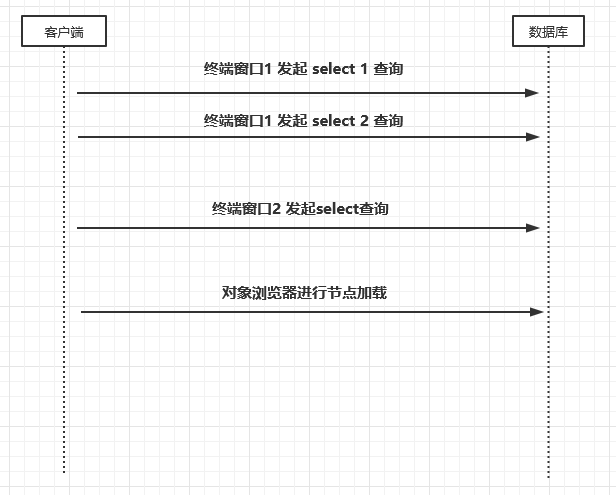
* **关于连接会话管理**

为了避免jdbc会话间的相互影响，故根据使用情况，会适当的单独创建专属会话。

例如：终端管理窗口独占一个jdbc会话。

浏览器窗口独占一个jdbc会话。

如此一来，当终端窗口执行大数据SQL，也不会影响对象浏览器的刷新或加载。

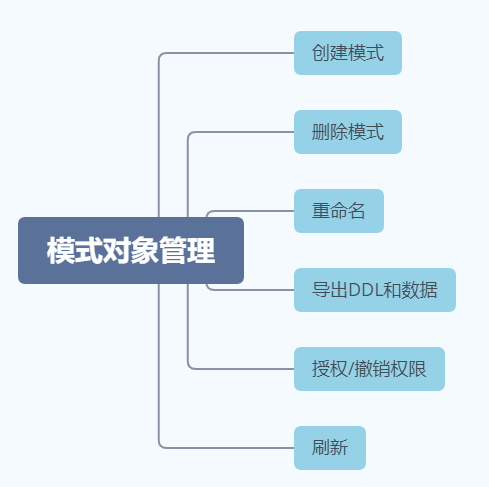


如上图所示，三个不同线程同时向数据库发起jdbc请求。其中终端窗口1会接连执行select1和select2语句。 处理方式采用：终端窗口1公用一个jdbc会话，select2必须等待select1结束才能执行。

而终端窗口2和对象浏览器刷新则各自有独立的jdbc会话，彼此间不受影响。

## 模式对象管理

### **功能简述**



### **系统部件**

#### **主要程序文件**

#### **关键数据结构说明**

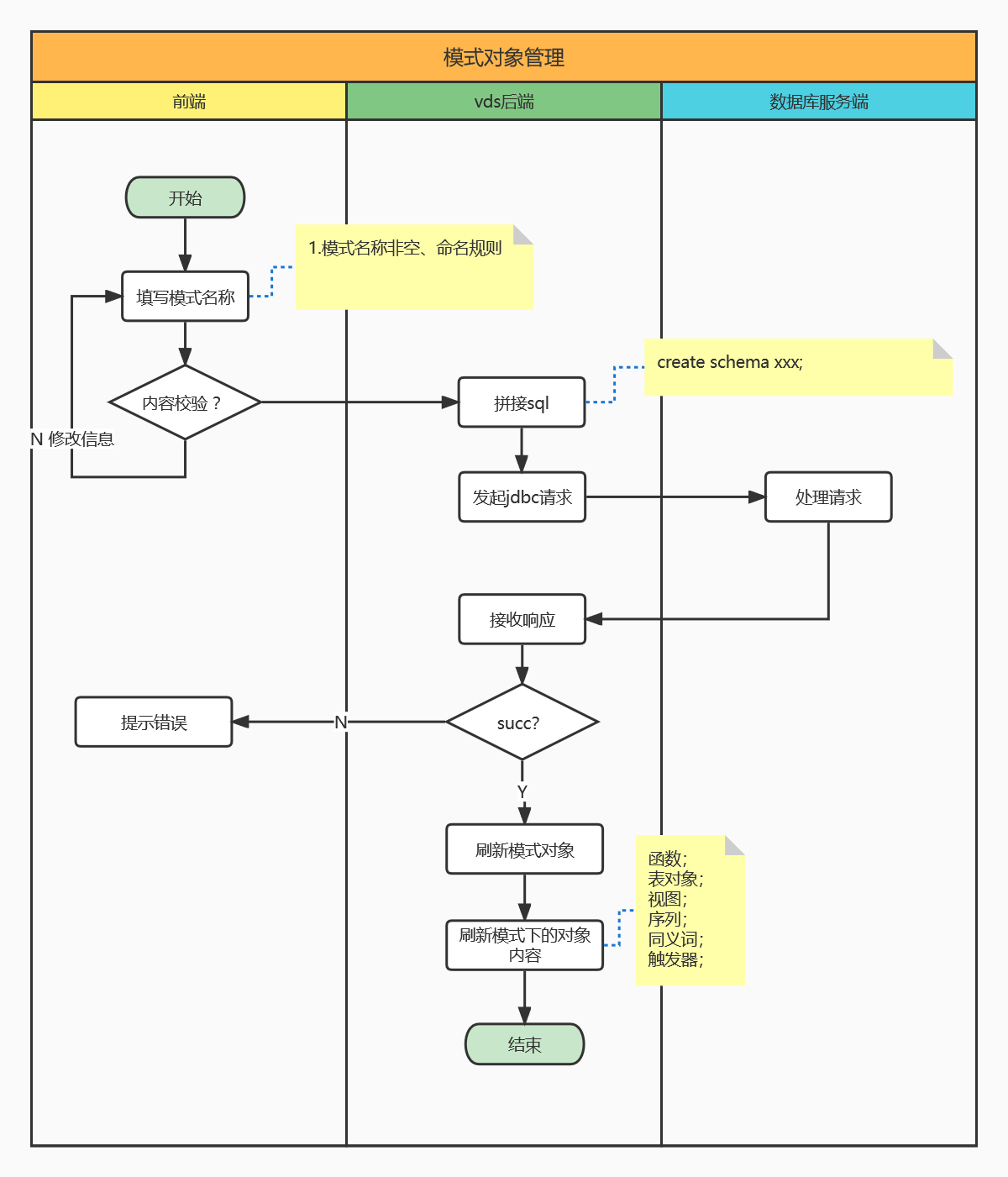
### **执行方案**

#### **数据连接对象管理**

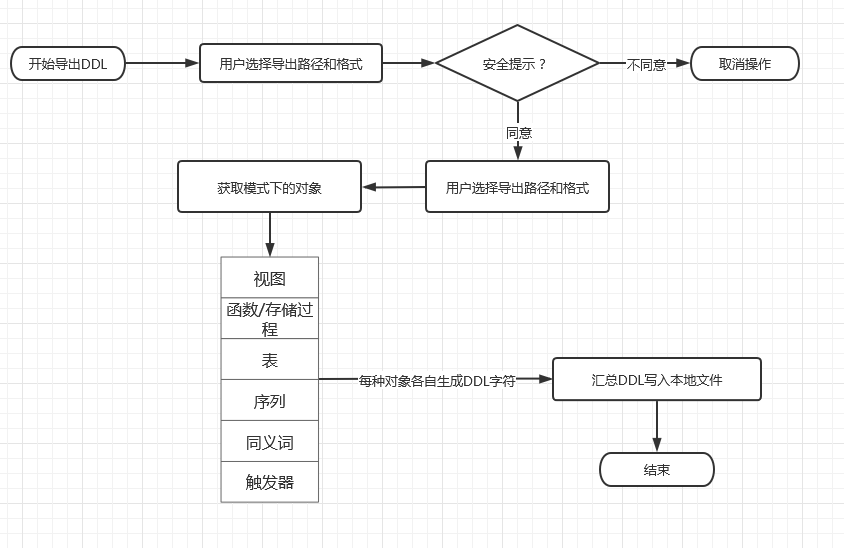
##### **实现逻辑描述**

##### **流程/交互设计**

* **创建模式交互图**



* **导出DDL流程**



##### **关键算法/SQL**

* **创建schema**

CREATE SCHEMA XXXX;

* **重命名schema**

ALTER SCHEMA schema\_name RENAME TO new\_name;

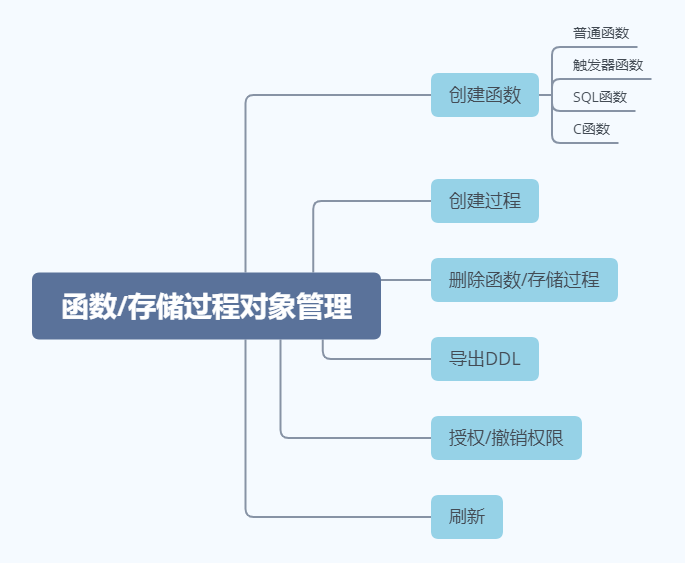
* **删除schema**

DROP SCHEMA [ IF EXISTS ] schema\_name [, ...] [ CASCADE | RESTRICT ];

##### **其它**

## 函数/存储过程对象管理

### **功能简述**



### **系统部件**

#### **主要程序文件**

#### **关键数据结构说明**

|  |  |  |  |
| --- | --- | --- | --- |
| **字典** | **说明** | **类型** | **是否必填** |
| 模式 | 函数/存储过程所在的模式 | 基本信息 | 是 |
| 名称 | 自定义名称 | 基本信息 | 是 |
| 区分大小写 | 名称是否大小写敏感 | 基本信息 | 否 |
| 语言 | PLPGSQL、SQL、C | 基本信息 | 是 |
| 返回值类型 | 常见数据类型 | 基本信息 | 是 |
| 参数列表 |  | 基本信息 | 否 |

### **执行方案**

#### **数据连接对象管理**

##### **实现逻辑描述**

* **创建函数/存储过程**

1. 用户填写函数名称、类型、大小写敏感、出参入参；
2. 后台校验函数信息，失败则提示；否则进入下一步；
3. 后台拼接sql，获取jdbc连接，发起请求；
4. 展示响应结果，成功则刷新函数组；

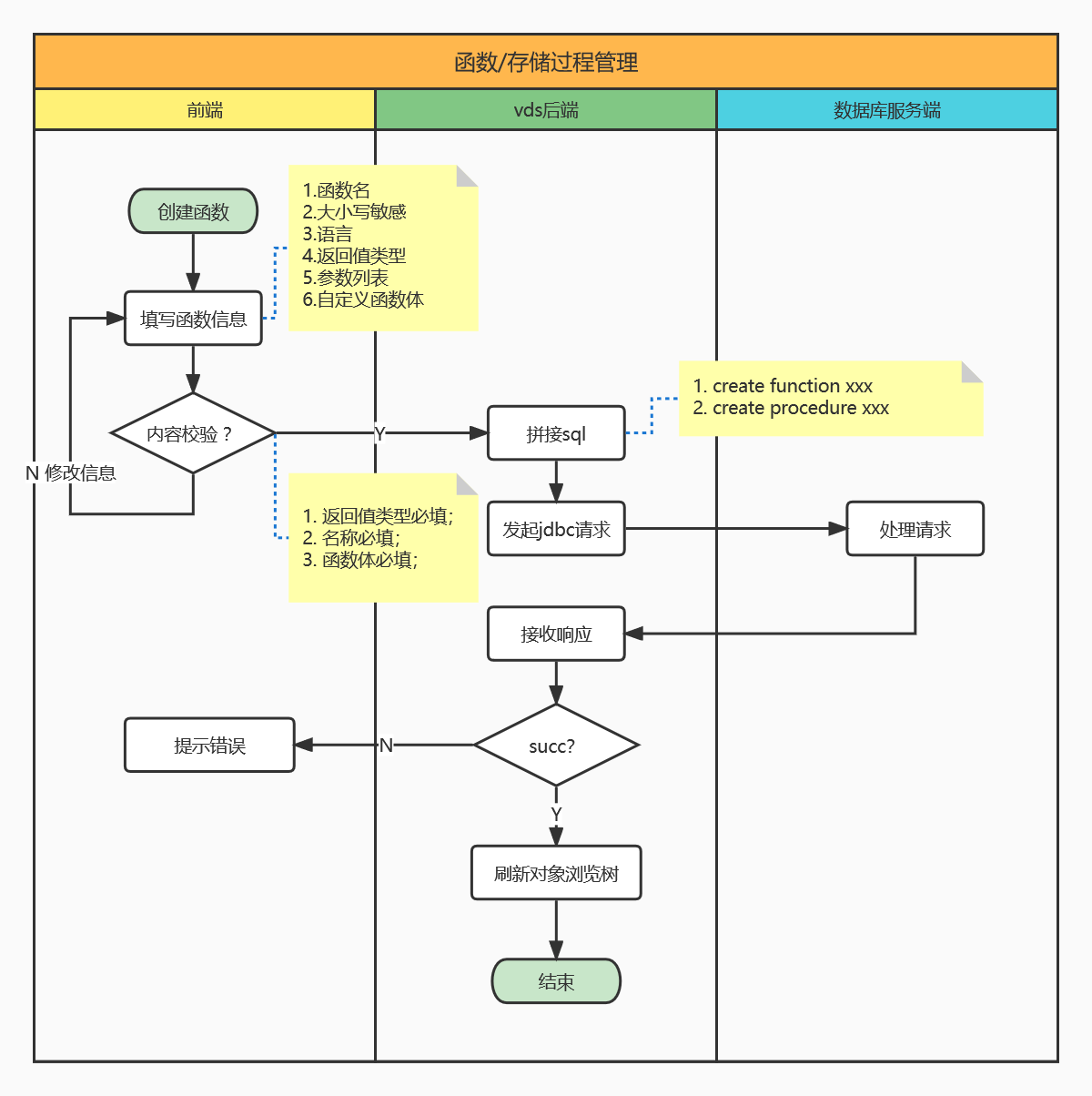
* **删除函数/存储过程**

1）根据schema + 函数名称，jdbc调起drop请求；

* **导出DDL**

使用pg\_get\_functiondef（oid）获取函数/过程的DDL；

##### **流程/交互设计**

****

##### **关键算法/SQL**

* **创建函数**

**CREATE OR REPLACE FUNCTION FUN\_NAME\_XXX(**

**ARG1 IN CHAR,**

**ARG2 OUT CHAR**

**)**

**RETURN varchar**

**LANGUAGE PLPGSQL**

**AS $$**

**DECLARE**

**/\*DECLARATION\_SECTION\*/**

**BEGIN**

**/\*EXECUTABLE\_SECTION\*/**

**END;$$**

**/**

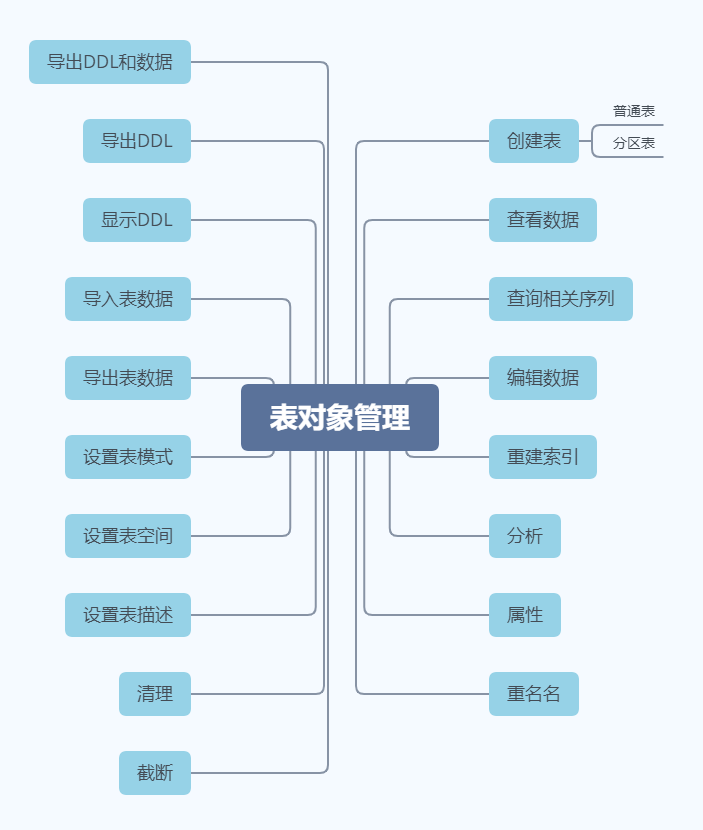
* **删除函数/存储过程**

**DROP FUNCTION/PROCEDURE XXX;**

##### **其它**

## 表对象管理

### **功能简述**



### **系统部件**

#### **主要程序文件**

#### **关键数据结构说明**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字典** | **说明** | **类型** | **父节点** | **必填** |
| 表名 | 表名称 | 基本信息 |  | 是 |
| 模式 | Schema | 基本信息 |  | 是 |
| 表空间 | 指定Tablespace | 基本信息 |  | 是 |
| 表类型 | 普通表/UNLOGGED | 基本信息 |  | 是 |
| 表存储方式 | ROW/COLUMN | 基本信息 |  | 是 |
| 填充因子 |  | 基本信息 |  | 否 |
| 表描述 | Comments | 基本信息 |  | 否 |
| 列字段列表 |  | 列表 |  | 是 |
| 列对象 |  | 对象 | 字段列表 | 是 |
| 列名 | 字段名称 | 列信息 | 列对象 | 是 |
| 阵列维度 |  | 列信息 | 列对象 | 是 |
| 数据类型 |  | 列信息 | 列对象 | 是 |
| 精度/大小 | 针对decimal等要求精度/大小的类型 | 列信息 | 列对象 | 否 |
| 列约束 | 非空、唯一、缺省 | 列信息 | 列对象 | 否 |
| 列描述 | 描述 | 列信息 | 列对象 | 否 |
| 数据分布 | 默认、REPLICATION、HASH | 数据分布 |  | 是 |
| 约束名称 |  | 约束信息 |  | 是 |
| 约束类型 | CHECK/UNIQUE/PRIMARY KEY/FOREIGN KEY | 约束信息 |  | 是 |
| 约束目标列 |  | 约束信息 |  | 是 |
| 检查表达式 |  | 约束信息 |  | 是 |
| 索引名 |  | 索引信息 |  | 是 |
| 访问方式 | btree、cbtree、cgin、gin、gist、hash、psort、spgist、ubtree | 索引信息 |  | 否 |
| 表空间 |  | 索引信息 |  | 否 |
| 填充因子 |  | 索引信息 |  | 是 |
| 索引目标列 |  | 索引信息 |  | 是 |
| WHERE表达式 |  | 索引信息 |  | 是 |

### **执行方案**

* **创建表**

1) 根据用户填写信息，通过jdbc调起create table操作。

* **删除表**

Jdbc执行Drop table。

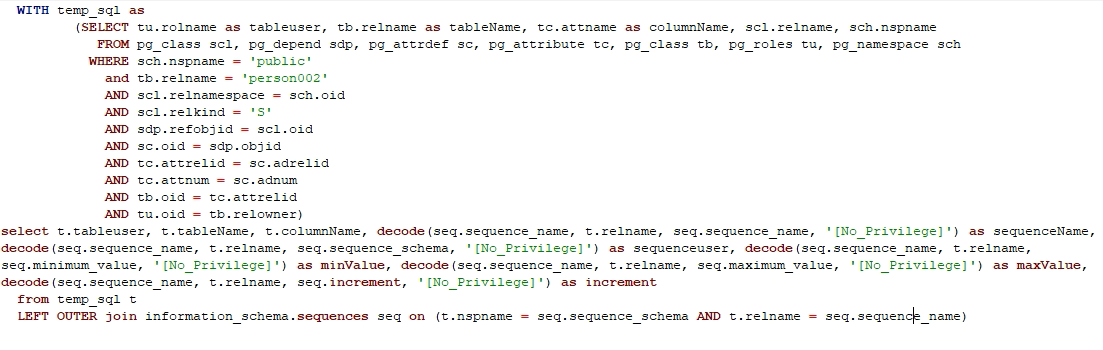
* **查看数据**

根据表拼装select语法，交由终端执行。

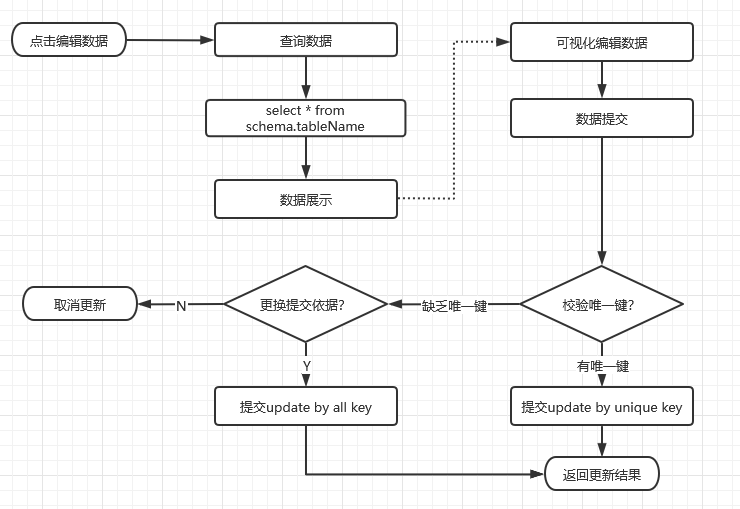
如查看表A，则拼接sql：select \* from schema.A ，交由终端执行并展示结果。

* **查询相关序列**

查询跟目标表关联的序列，sql参考：



* **编辑数据**



* **重建索引**

目标表执行sql：

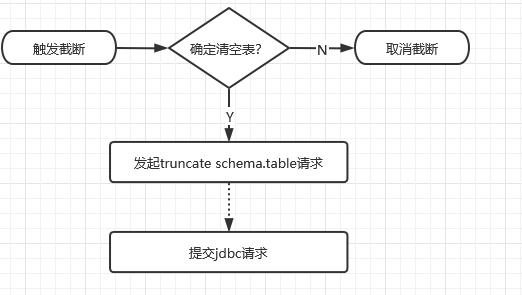
Reindex table xxx；

* **分析**

目标表执行sql：

Analyze xxx ；

* **截断**



通过jdbc执行truncate schema.table 操作;

* **清理**

通过jdbc执行

VACUUM tableName ;

* **设置表描述**

通过jdbc执行

COMMENT ON TABLE XXX IS COMMENT\_DESC;

* **设置表空间**

通过jdbc执行

ALTER TABLE ONLY TABLENAME SET TABLESPACE TPNAME;

* **设置表模式**

通过jdbc执行

ALTER TABLE ONLY TABLENAME SET SCHEMA SCHEMANAME;

* **导出表数据**
* **导入表数据**
* **显示DDL**

Select \* from pg\_get\_tabledef;

* **导出DDL和数据**
* **重命名**
* **删除**

DTOR TABLE XXX

* **属性**
* **刷新**

重新加载表对象：列信息、约束信息、索引信息

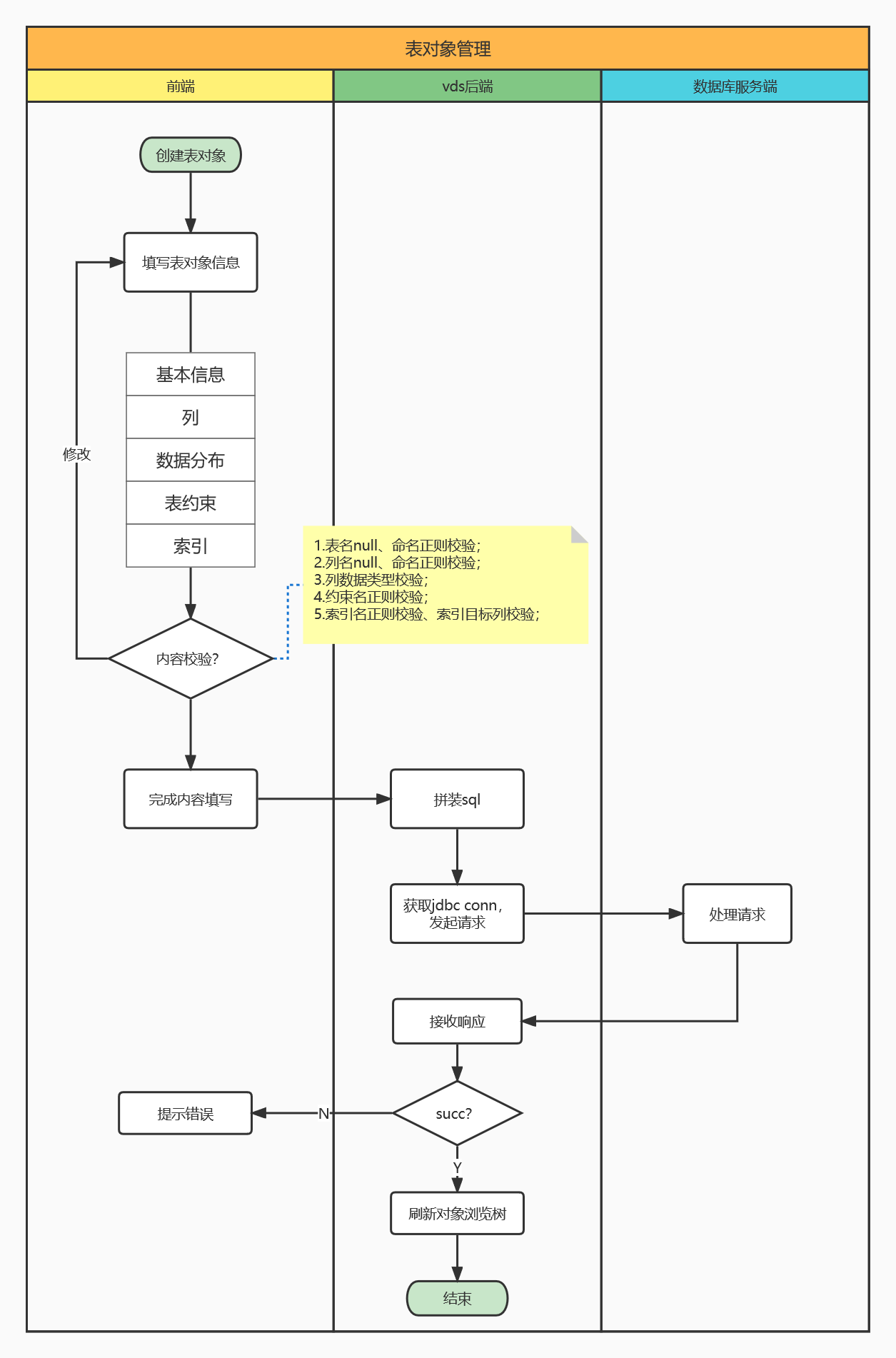
|  |  |
| --- | --- |
| pg\_attribute | 列信息 |
| pg\_constraint | 约束信息 |
| pg\_indexes | 索引 |

#### **表对象管理**

##### **实现逻辑描述**

##### **流程/交互设计**

* **创建表交互流程**



##### **关键算法/SQL**

关于表元数据

select tbl.oid as oid, tbl.relname relname, tbl.relnamespace relnamespace,

ts.spcname as reltablespace,tbl.relpersistence relpersistence, d.description as desc,

xctbl.nodeoids as nodes ,tbl.reloptions as reloptions

from pg\_class tbl left join (select d.description, d.objoid from pg\_description d

where d.objsubid=0) d on (tbl.oid = d.objoid) left join pgxc\_class xctbl

on (tbl.oid = xctbl.pcrelid) left join pg\_tablespace ts on ts.oid = tbl.reltablespace

where tbl.relkind = 'r' and tbl.parttype in ('n','p') and tbl.oid = %d

关于表定义

Select \* from pg\_get\_tabledef;

关于表删除

Drop table xxx;

关于表清理

VACUM xxx；

关于表分析

ANLYZE xxx；

重建索引

REINEX TABLE XXX;

表清理

TRUNCATE TABLE ONLY XXX;

修改表空间

Alter table only xxx set tablespace xxx；

表注释

Comment on table xxx；

表模式修改

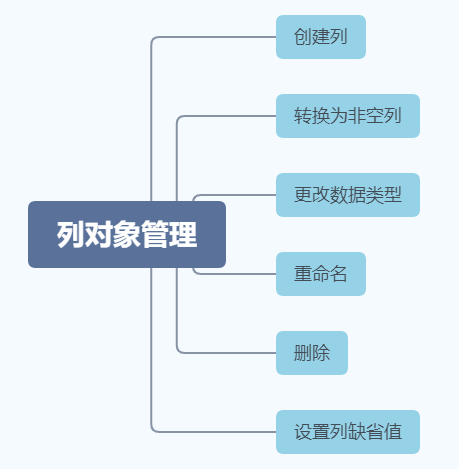
Alter table only xxx set schema xxx；

##### **其它**

mot表现阶段不支持，在2.2.10中由于无法关闭增量检查点，故无法进行mot表的创建。

## 列对象管理

### **功能简述**



### **系统部件**

#### **主要程序文件**

#### **关键数据结构说明**

|  |  |  |  |
| --- | --- | --- | --- |
| **字典** | **说明** | **类型** | **是否必填** |
| 列名 | 列名称 | 基本信息 | 是 |
| 数据类型 | 列数据类型，参考pg\_types | 基本信息 | 是 |
| 精度/大小 | 针对decimal等数字类型设定精度； | 基本信息 | 否 |
| 列约束 | 非空、唯一、缺省 | 基本信息 | 否 |
| 列描述 | Comment | 基本信息 | 否 |

### **执行方案**

#### **数据连接对象管理**

##### **实现逻辑描述**

* **创建列**

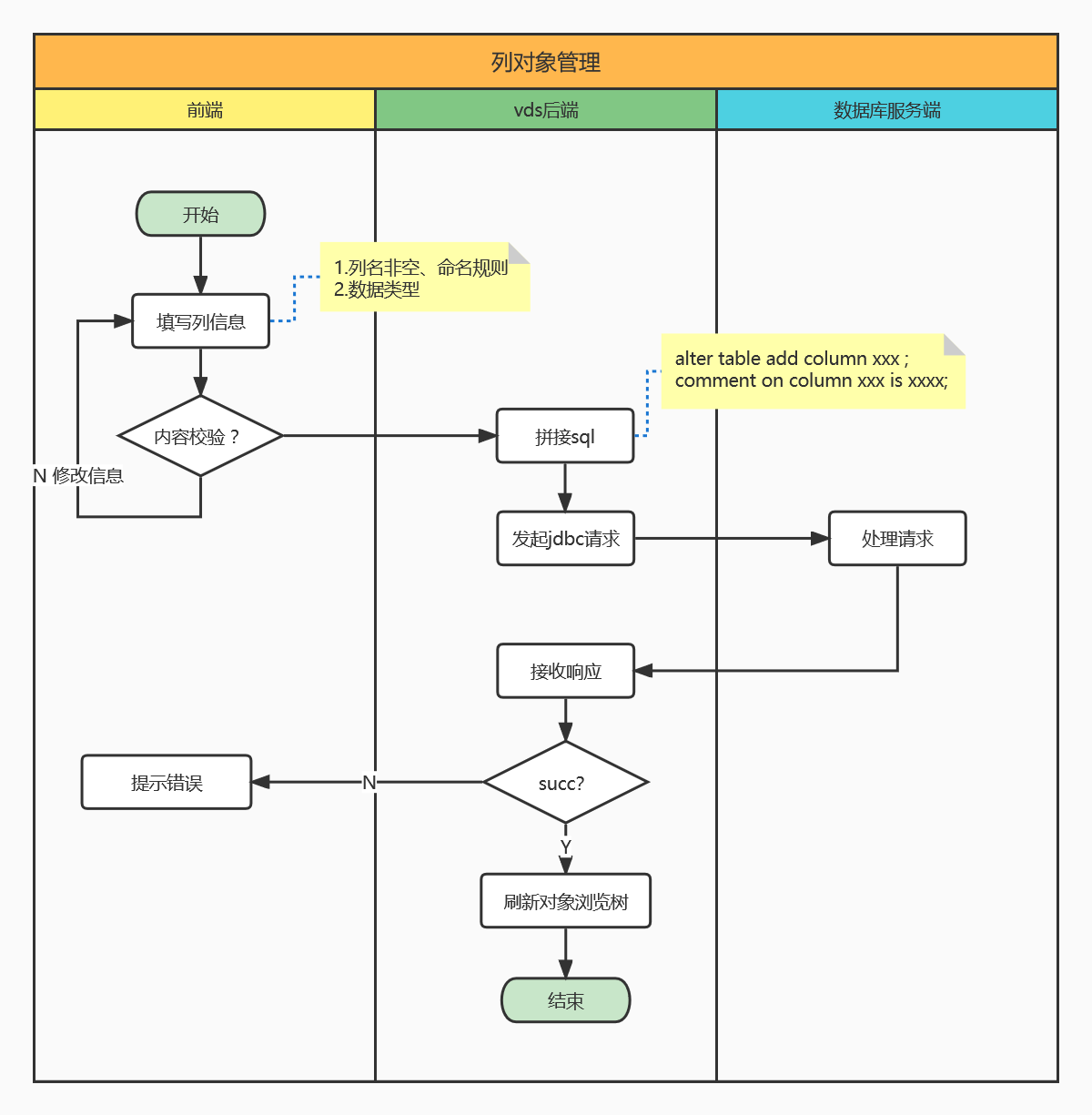
1. 用户填写列信息；
2. 后台校验信息合法性；
3. 校验通过后，后台jdbc发起alter table add column执行语法；

* **更换非空、设置缺省、更改数据类型**

通过jdbc执行alter 语法

##### **流程/交互设计**

* **创建列对象交互流程**



##### **关键算法/SQL**

* 添加列

Alter table tableName add column column not null default defaultValue；

-- 列注释

Coment on column xxx is commentDesc；

* 删除列

Drop column xxx；

* 重命名

Alter table tableName rename column xxx TO newname；

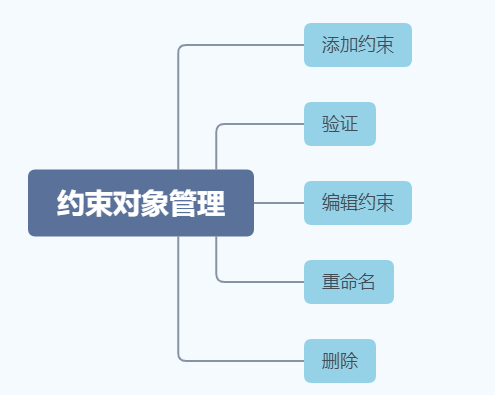
* 修改数据类型

Alter table tableName alter column type newType;

##### **其它**

## 约束对象管理

### **功能简述**



### **系统部件**

#### **主要程序文件**

#### **关键数据结构说明**

|  |  |  |  |
| --- | --- | --- | --- |
| **字典** | **说明** | **类型** | **是否必填** |
| 约束对象 | 约束目标，这里指表对象 | 基本信息 | 是 |
| 约束名称 | 约束名称 | 基本信息 | 是 |
| 约束类型 | 检查、唯一、主键、外键 | 基本信息 | 是 |
| 表达式 | 表达式 | 基本信息 | 否 |

### **执行方案**

* **添加约束**

1）用户填写约束信息；

2）后台校验信息合法性；

3）校验通过后，后台jdbc发起alter table add constraint执行语法；

* **验证**

通过jdbc执行alter 语法；

* **编辑约束**

Jdbc执行alter 语法；

* **重命名**

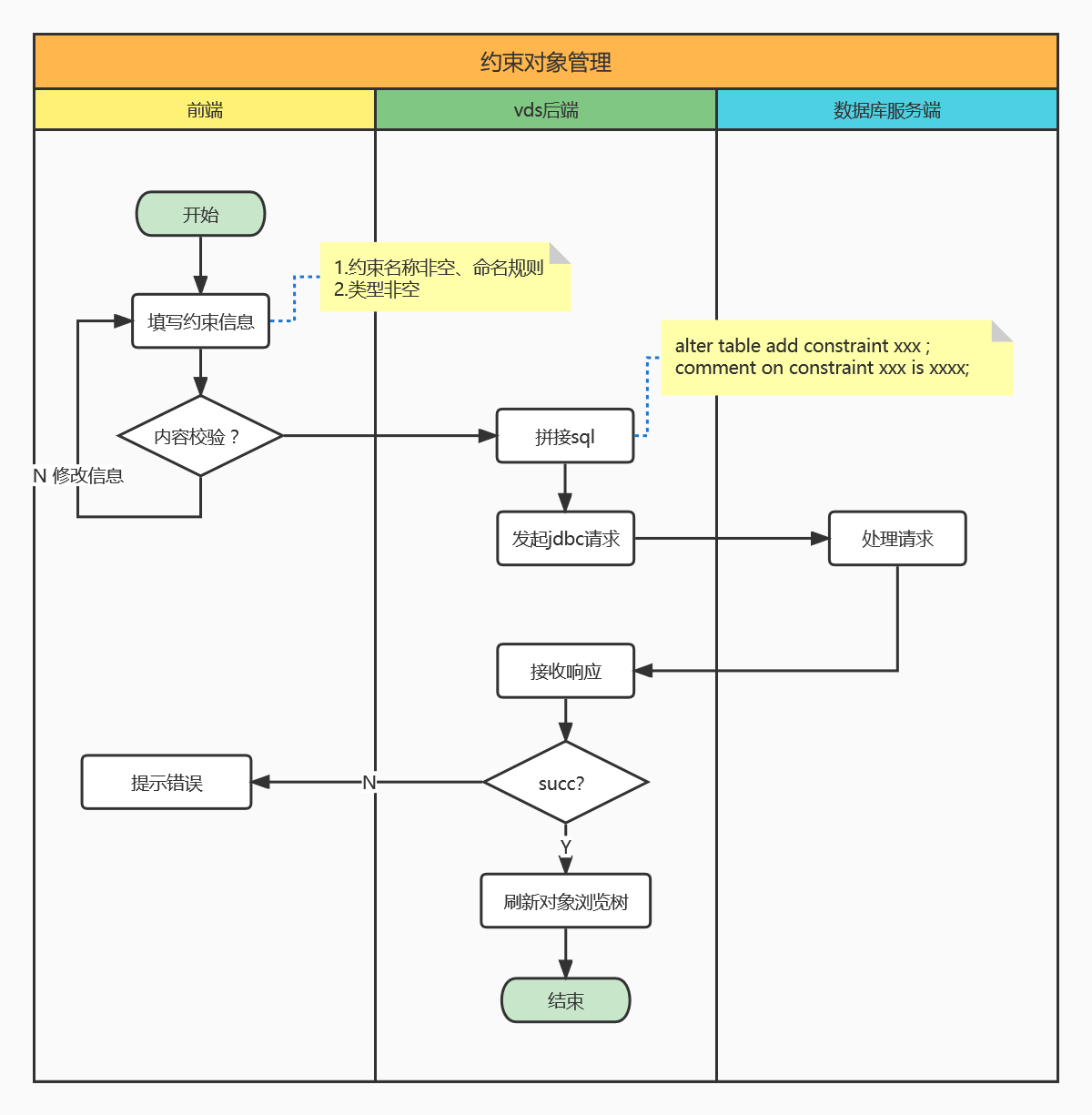
Jdbc执行Alter rename；

#### **约束对象管理**

##### **实现逻辑描述**

##### **流程/交互设计**

* **添加约束**



##### **关键算法/SQL**

* **添加约束**

**Alter table add**

**CHECK / UNIQUE / PRIMARY KEY / FOREIGN KEY / PARTIAL CLUSTER KEY /**

**EXCULSION KEY**

**[WITH OPTIONS ]**

**constraintName;**

* **启动/禁用约束**

**Alter table tableName modify constraint constraintName enable/disable;**

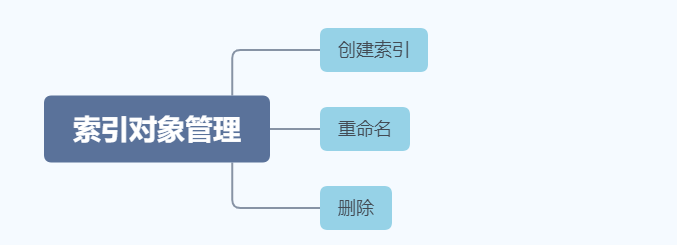
* **删除**

**Alter table tablename Drop constraint name;**

##### **其它**

## 索引对象管理

### **功能简述**



### **系统部件**

#### **主要程序文件**

#### **关键数据结构说明**

|  |  |  |  |
| --- | --- | --- | --- |
| **字典** | **说明** | **类型** | **是否必填** |
| 索引名 | 索引名 | 基本信息 | 是 |
| 访问方式 | Btree、cbtree、cgin、gin、gist、  Hash、psort、spgist、ubtree | 基本信息 | 否 |
| 表空间 |  | 基本信息 | 否 |
| 填充因子 |  | 基本信息 | 否 |
| 表达式 |  | 基本信息 | 否 |
| 索引目标 |  | 基本信息 | 是 |

### **执行方案**

#### **数据连接对象管理**

##### **实现逻辑描述**

* **添加索引**

1）用户填写索引信息；

2）后台校验信息合法性；

3）校验通过后，后台jdbc发起creaet index执行语法；

* **重命名**

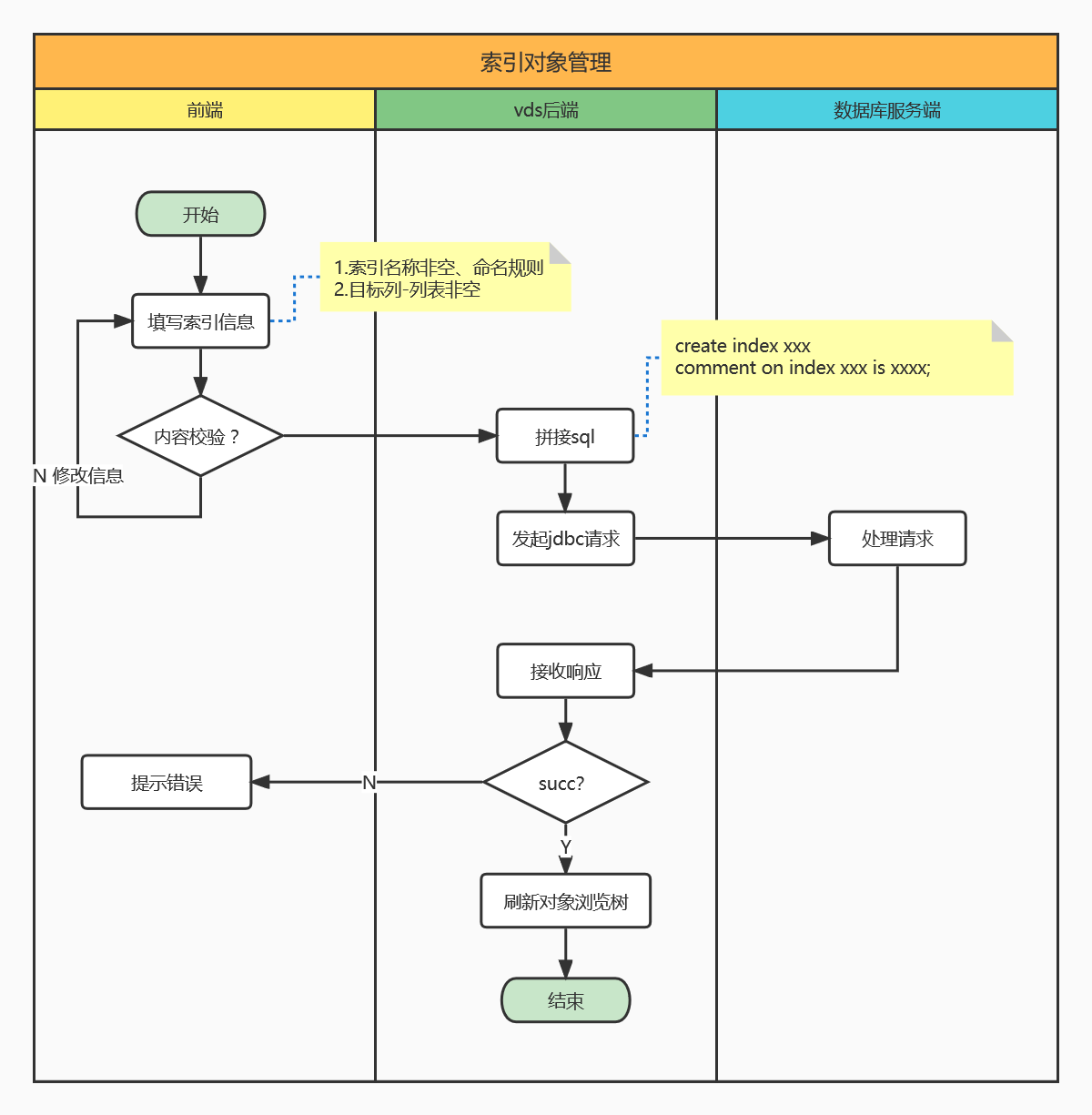
Jdbc执行Alter rename；

* **删除**

Jdbc执行drop index；

##### **流程/交互设计**

* **创建索引**



##### **关键算法/SQL**

* **创建索引**

Create [unique] INDEX schemaName.indexName ON schema.tableName(columnName,columnName2...)

With (fillFactor = xxx);

* **重命名**

ALTER INDEX oldName RENAME TO newName;

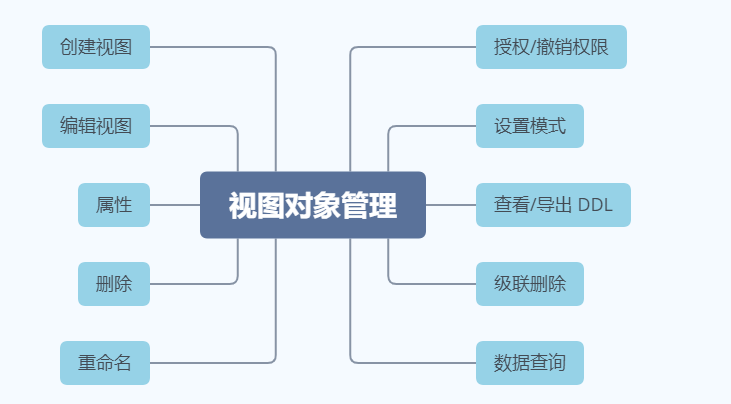
* **删除**

Drop index xxx；

##### **其它**

## 视图对象管理

### **功能简述**



### **系统部件**

#### **主要程序文件**

#### **关键数据结构说明**

|  |  |  |  |
| --- | --- | --- | --- |
| **字典** | **说明** | **类型** | **是否必填** |
| 模式名 | Namespace名称 | 基本信息 | 是 |
| 视图类型 | 普通视图、物化视图 | 基本信息 | 是 |
| 视图名称 |  | 基本信息 | 是 |
| 视图SQL | 表、列的查询SQL | 基本信息 | 是 |

### **执行方案**

#### **视图对象管理**

##### **实现逻辑描述**

* **创建**

1）用户填写视图信息；

2）后台校验信息合法性；

3）校验通过后，后台jdbc发起create view执行语法；

* **编辑视图**

编辑视图采用先drop 后 create方式，故修改的内容不限制；

* **设置模式**

Jdbc执行alter view set schema语法

* **删除**

Jdbc执行drop view语法

* **授权/撤销**

Jdbc执行grant/revoke

* **查看数据**

终端执行select \* from view；

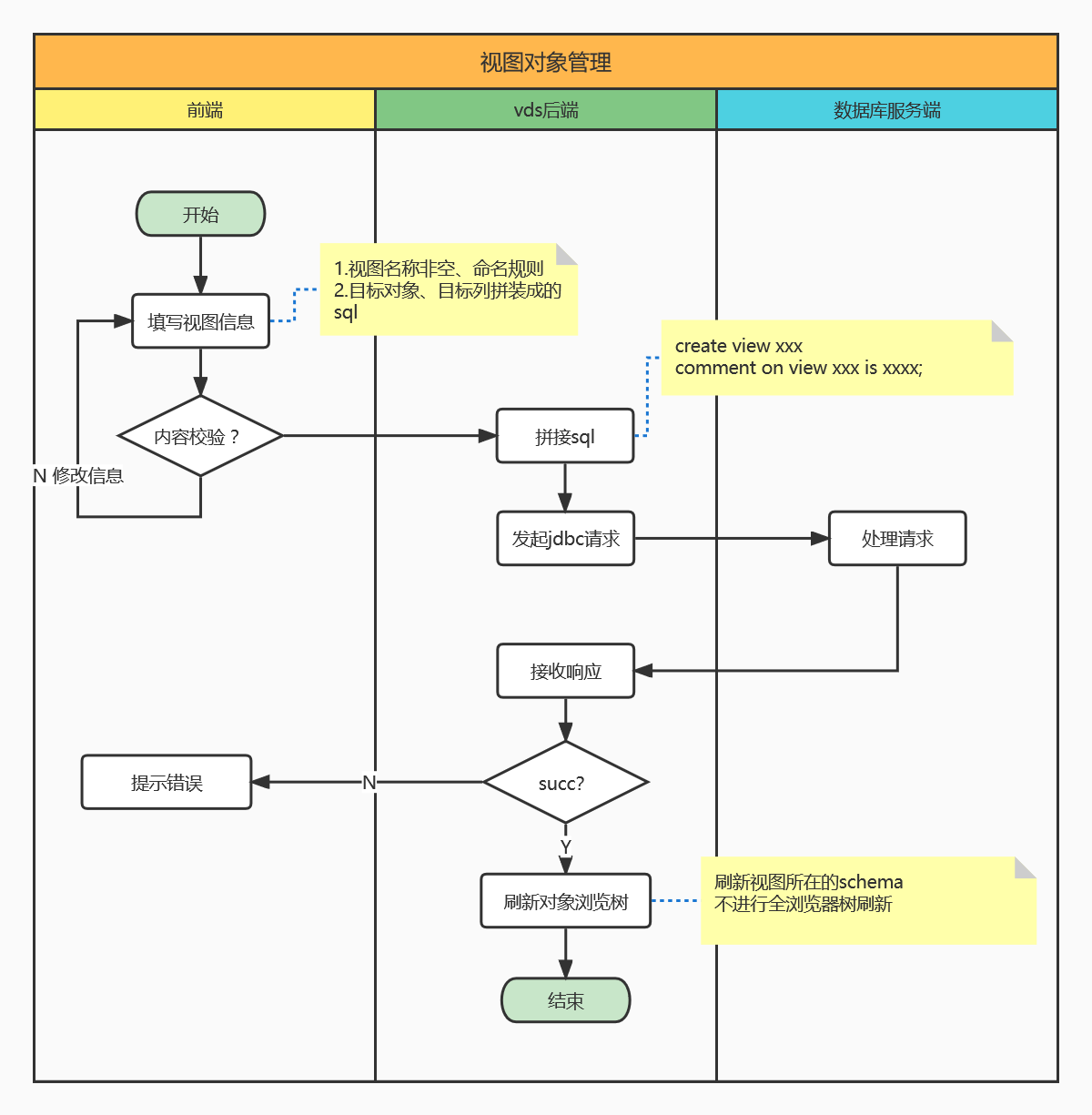
* **重命名**

Jdbc执行Alter rename；

* **删除**

Jdbc执行drop index；

##### **流程/交互设计**



##### **关键逻辑/SQL**

* **创建视图**

**Create or replace [MATERIALIZED] view schemaName.viewName**

**AS**

**SELECT ...**

* **DDL**

Select \* from pg\_get\_viewdef(‘viewName’);

* **列展示**

Select \* from pg\_attribute

Left join pg\_type on pg\_type.oid = pg\_attribute.atttypid;

* **重命名**

Alter view viewName rename TO newname;

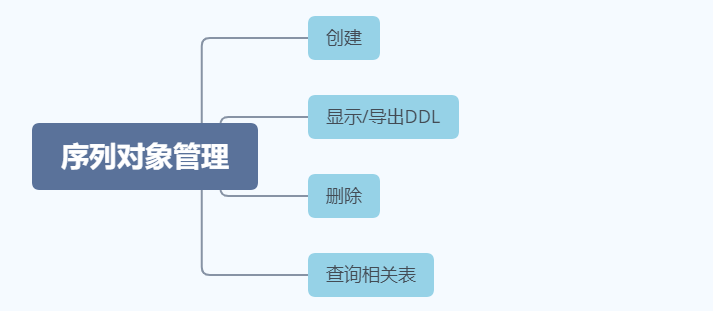
* **设置模式**

ALTER VIEW [ IF EXISTS ] view\_name SET SCHEMA new\_schema;

##### **其它**

## 序列对象管理

### **功能简述**



### **系统部件**

#### **主要程序文件**

#### **关键数据结构说明**

|  |  |  |  |
| --- | --- | --- | --- |
| **字典** | **说明** | **类型** | **是否必填** |
| 序列名称 |  | 基本信息 | 是 |
| 最小值 |  | 基本信息 | 否 |
| 最大值 |  | 基本信息 | 否 |
| 增量 |  | 基本信息 | 否 |
| 初始值 |  | 基本信息 |  |
| 缓存数值 |  | 基本信息 |  |
| 是否循环 |  | 基本信息 |  |
| 模式 | 序列目标所在模式 | 基本信息 |  |
| 表 | 序列目标所在表对象 | 基本信息 |  |
| 列 | 序列目标列 | 基本信息 |  |

### **执行方案**

#### **序列对象管理**

##### **实现逻辑描述**

* **创建**

1）用户填写序列信息；

2）后台校验信息合法性；

3）校验通过后，后台jdbc发起create sequence执行语法；

* **删除**

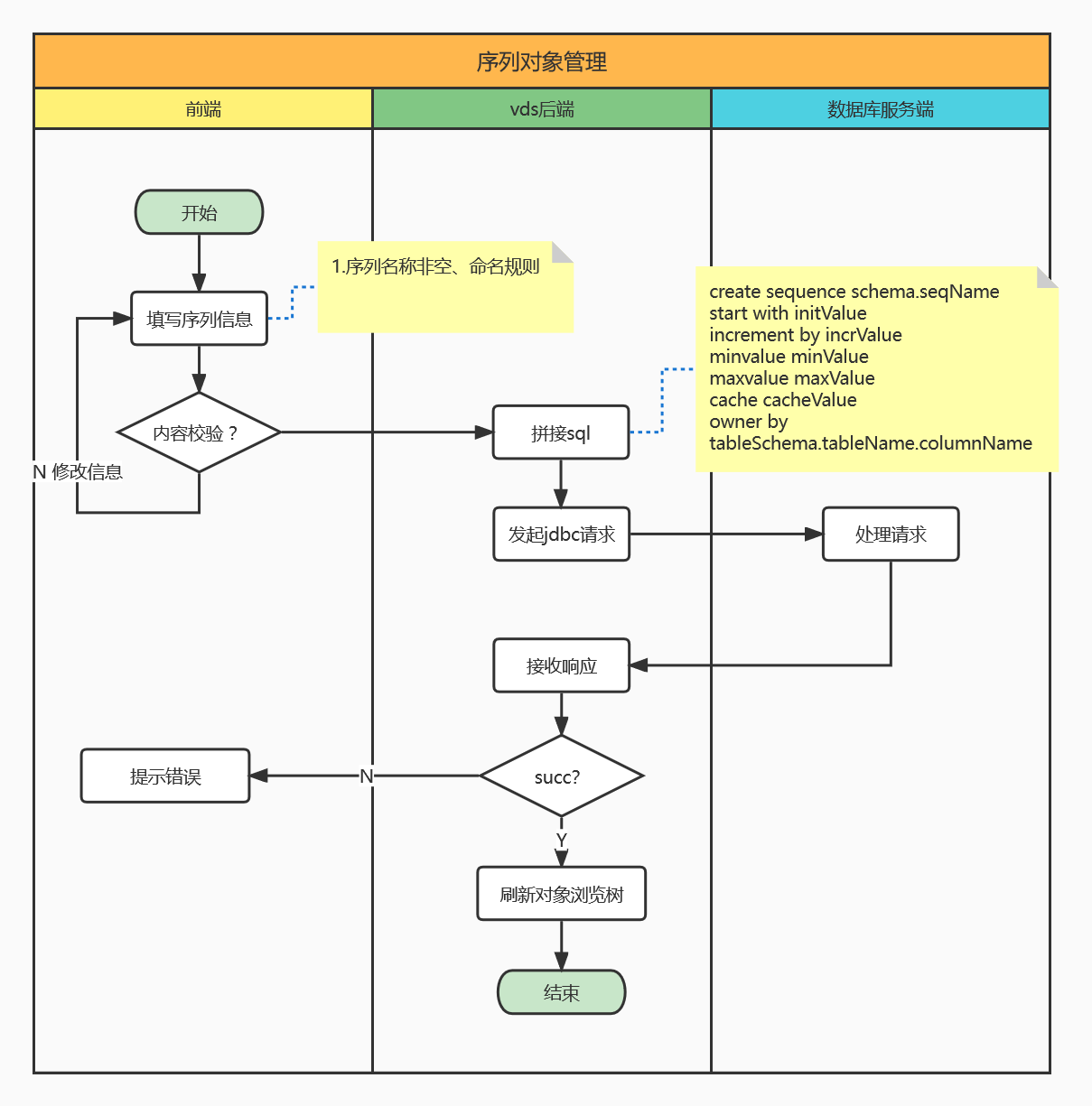
Jdbc执行drop sequence操作；

* **查询相关表**

拼装sql交由终端查询；

##### **流程/交互设计**

* **创建序列交互图**

****

##### **关键算法/SQL**

* **创建序列**

Create sequence xxx

Start with startValue

Increment by incrValue

Minvalue minValue

Cache cacheValue

Cycle cycleValue

Owner by owner；

* **DDL**

SELECT sequence\_name,start\_value, increment\_by,

CASE WHEN increment\_by > 0 AND max\_value = %d THEN NULL

WHEN increment\_by < 0 AND max\_value = -1 THEN NULL ELSE max\_value END AS max\_value,

CASE WHEN increment\_by > 0 AND min\_value = 1 THEN NULL WHEN increment\_by < 0 AND min\_value = %s

THEN NULL ELSE min\_value END AS min\_value, cache\_value, is\_cycled FROM %s

* **删除**

Drop sequence if exists seqName；

* **查询相关表**

SELECT seq.sequence\_name as sequenceName, seq.sequence\_schema as sequenceuser,

seq.minimum\_value as minValue,seq.maximum\_value as maxValue , seq.increment as increment, tc.attname as columnName, tu.rolname as tableuser

, tb.relname as tableName FROM information\_schema.sequences seq, pg\_namespace sch, pg\_class scl, pg\_depend sdp

, pg\_attrdef sc, pg\_attribute tc, pg\_class tb, pg\_roles tu

WHERE seq.sequence\_schema = ? AND seq.sequence\_name = ? AND sch.nspname = seq.sequence\_schema

AND scl.relnamespace = sch.oid AND scl.relname = seq.sequence\_name AND scl.relkind = 'S' AND sdp.refobjid = scl.oid

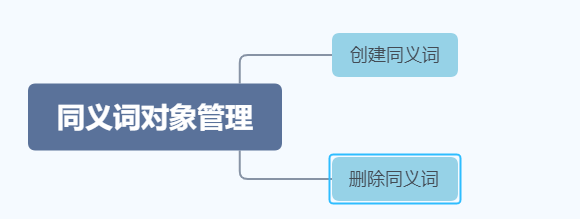
AND sc.oid = sdp.objid AND tc.attrelid = sc.adrelid AND tc.attnum = sc.adnum AND tb.oid = tc.attrelid

AND tu.oid = tb.relowner;

##### **其它**

## 同义词对象管理

### **功能简述**



### **系统部件**

#### **主要程序文件**

#### **关键数据结构说明**

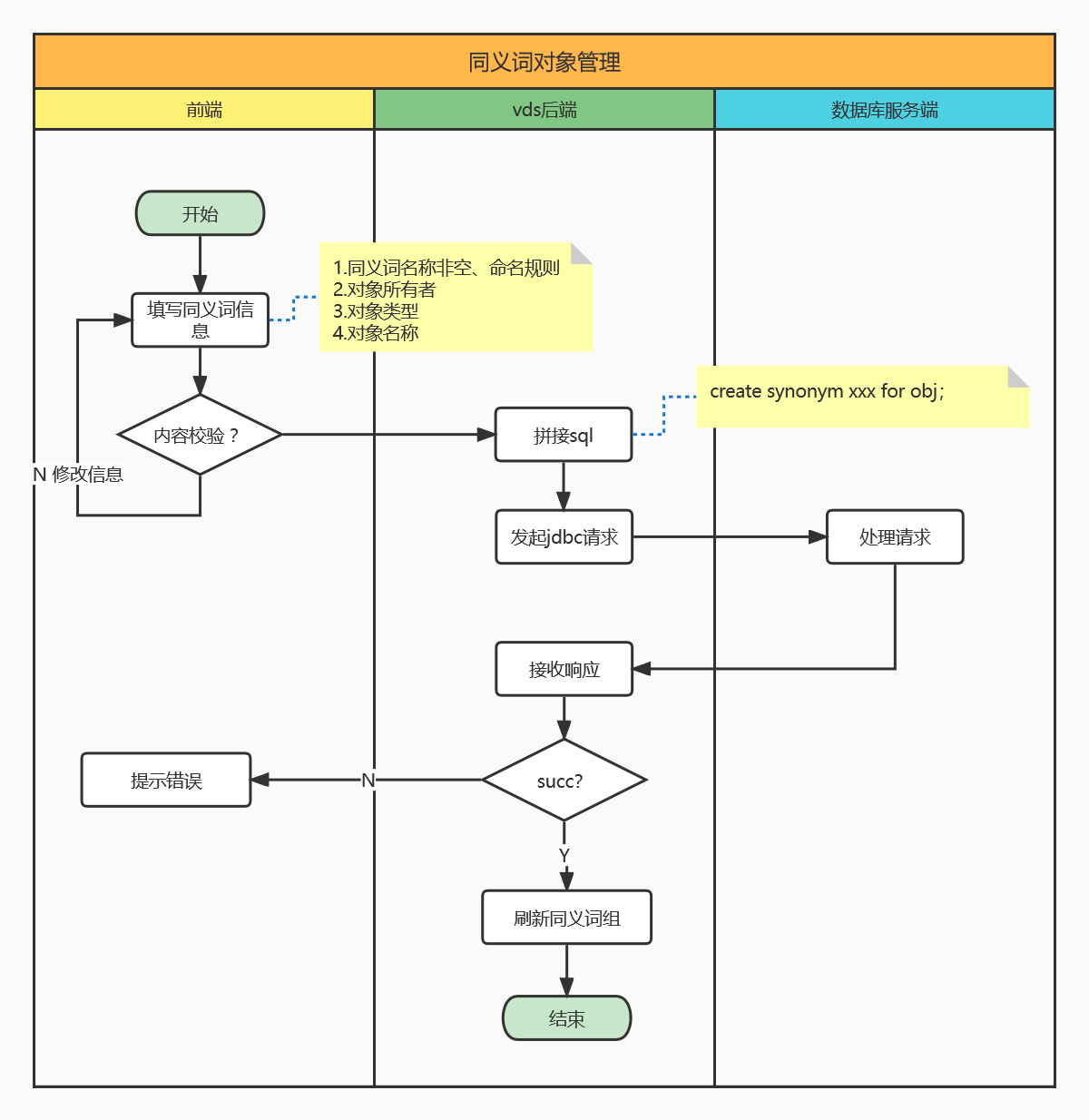
|  |  |  |  |
| --- | --- | --- | --- |
| **字典** | **说明** | **类型** | **是否必填** |
| 同义词名称 |  | 基本信息 | 是 |
| 对象所有者 | 对象所在的schema | 基本信息 | 是 |
| 对象类型 | 对象类型：函数/过程、视图、普通表 | 基本信息 | 是 |
| 对象名称 |  | 基本信息 | 是 |

### **执行方案**

#### **数据连接对象管理**

##### **实现逻辑描述**

##### **流程/交互设计**

****

##### **关键算法/SQL**

* **创建同义词**

**Create or replace synonym xxx**

**Owner.synonymName**

**For obj;**

* **同义词查询**

**select pgs.synname as synonym\_name, pgr.rolname as owner,**

**pgn.nspname as schema\_name, pgs.synobjname**

**as table\_name from pg\_synonym pgs, pg\_namespace pgn,pg\_roles pgr where**

**pgn.oid=pgs.synnamespace and pgs.synowner=pgr.oid**

* **删除**

Drop synonym if exists xxx；

##### **其它**

## 触发器对象管理

### **功能简述**



### **系统部件**

#### **主要程序文件**

#### **关键数据结构说明**

|  |  |  |  |
| --- | --- | --- | --- |
| **字典** | **说明** | **类型** | **是否必填** |
| 触发器名称 |  | 基本信息 | 是 |
| 触发器类型 |  | 基本信息 | 是 |
| 事件 | INSERT、DELETE、TRUNCATE、UPDATE | 基本信息 | 是 |
| 模式 |  | 基本信息 | 是 |
| 触发表/视图名 |  | 基本信息 | 是 |
| 触发条件 |  | 基本信息 | 否 |
| 触发级别 | 语句级触发器、行级触发器 | 基本信息 | 是 |
| 触发函数 |  | 基本信息 | 是 |
| 参数列表 |  | 基本信息 | 否 |

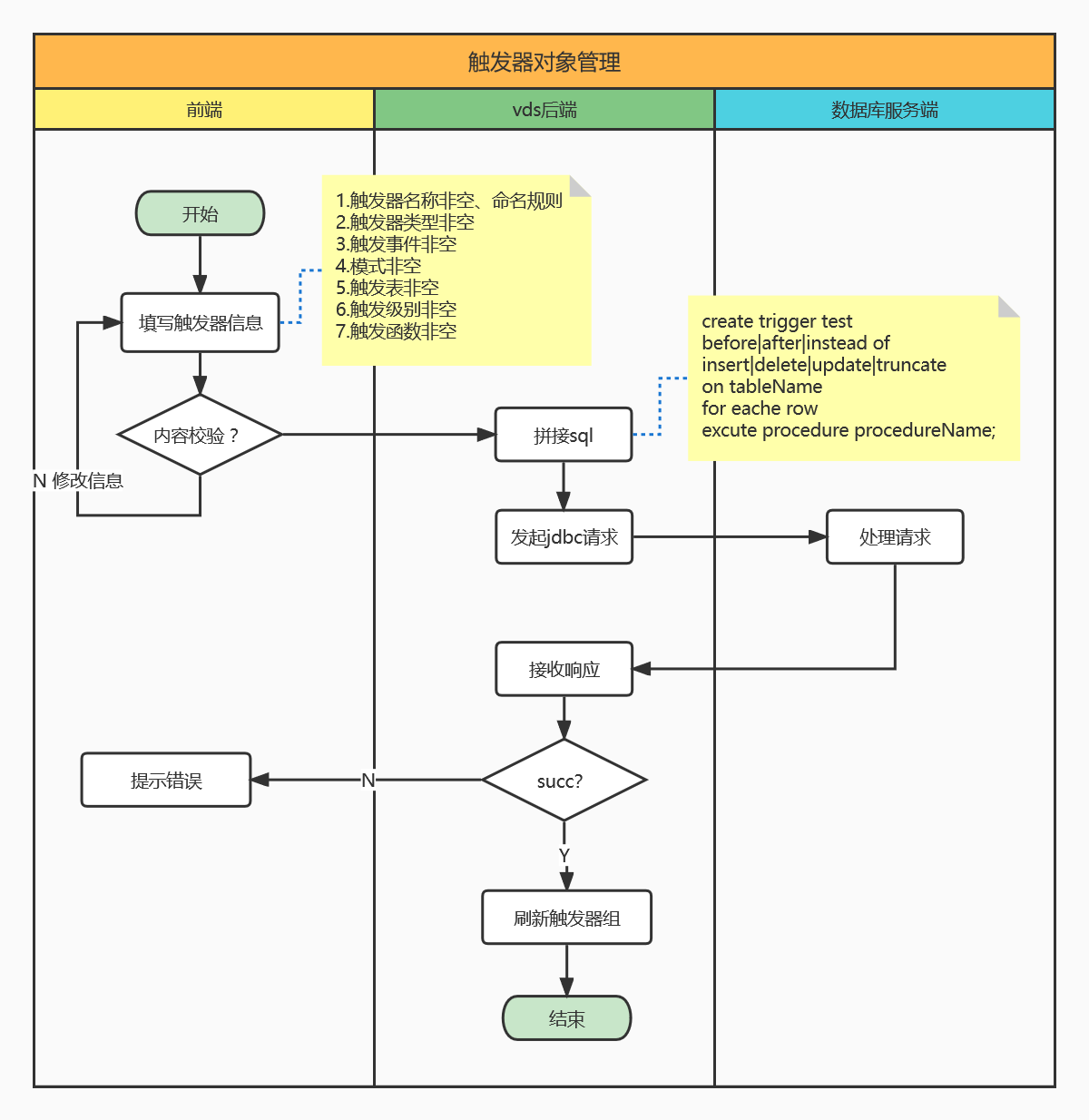
### **执行方案**

#### **数据连接对象管理**

##### **实现逻辑描述**

##### **流程/交互设计**

* **创建触发器**

****

##### **关键算法/SQL**

* **删除触发器**

Drop trigger xxx on schema.tableName ;

* **启动/禁用触发器**

Alter table schema.tableName ENABLE/DISABLE trigger triggerName ;

**重命名**

Alter trigger triggerName

On schema.tableName

Rename to newName ;

##### **其它**

## 表空间对象管理

### **功能简述**



### **系统部件**

#### **主要程序文件**

#### **关键数据结构说明**

|  |  |  |  |
| --- | --- | --- | --- |
| **字典** | **说明** | **类型** | **是否必填** |
| 表空间名称 |  | 基本信息 | 是 |
| 相对路径 |  | 基本信息 | 是 |
| 位置 |  | 基本信息 | 是 |
| 大小限制 |  | 基本信息 | 是 |
| 存储容量 | 单位：KB、MB、GB、TB、PB | 基本信息 | 是 |
| 顺序磁盘页面开销 |  | 基本信息 | 是 |
| 非顺序磁盘页面开销 |  | 基本信息 | 是 |

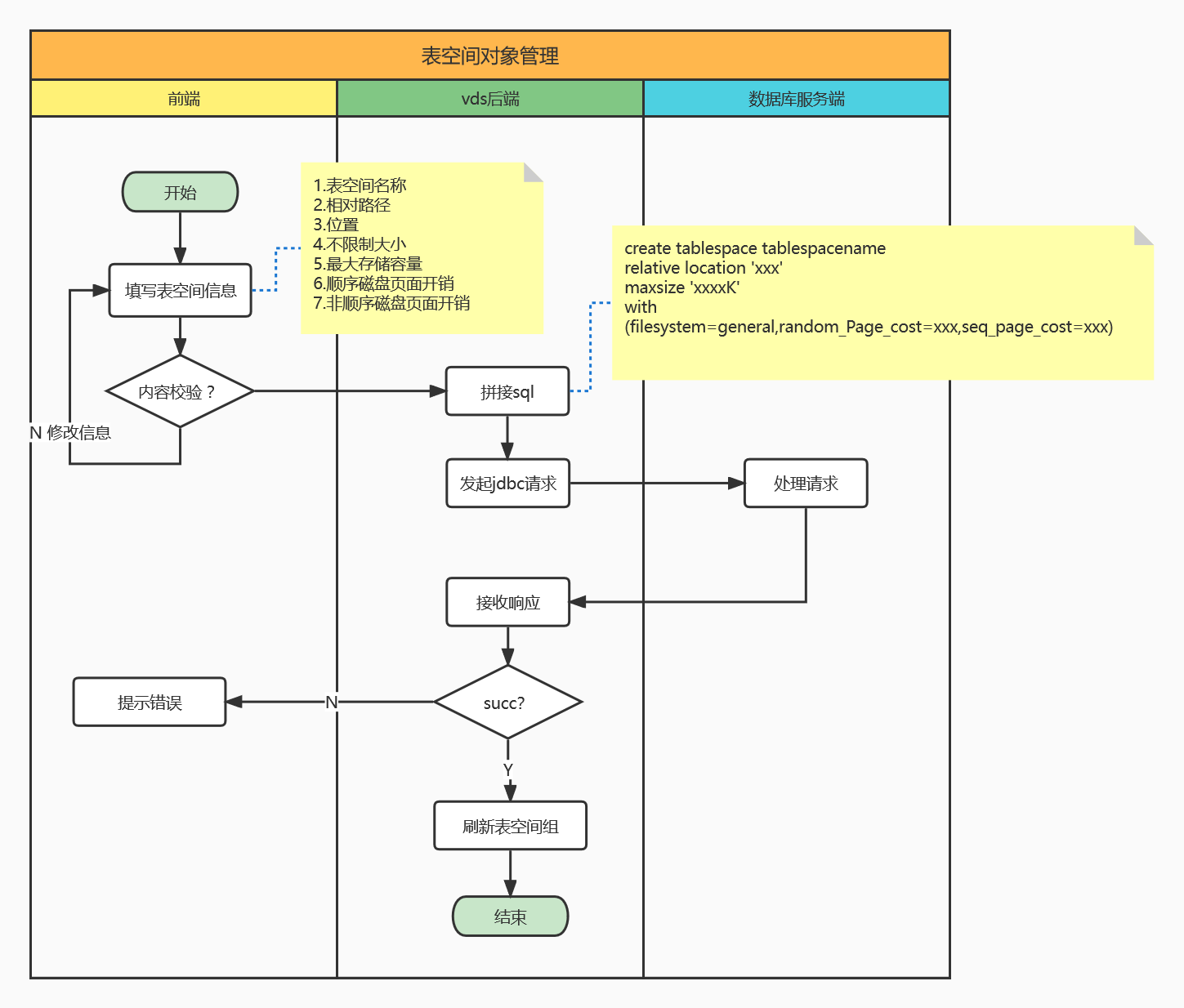
### **执行方案**

#### **表空间对象管理**

##### **实现逻辑描述**

##### **流程/交互设计**

* **创建表空间**



##### **关键算法/SQL**

* **创建表空间**

-- 设置是否绝对路径

Set enable\_absolute\_tablespace=on|off;

-- create tablespace sql

create tablespace tablespacename

relative location 'xxx'

maxsize 'xxxxK'

with (filesystem=general,random\_Page\_cost=xxx,seq\_page\_cost=xxx)

* **重命名**

Alter tablespace xxx rename to newName；

* **设置大小**

Alter tablespace xxx resize maxsize xxx；

* **设置选项**

Alter tablespace xxx set (random\_page\_cost=newValue);

Alter tablespace xxx set (seq\_page\_cost=newValue);

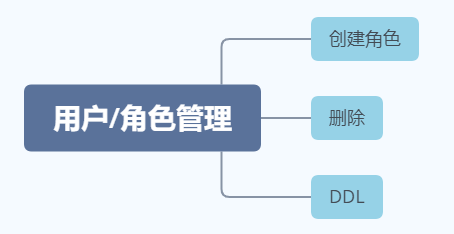
* **删除**

Drop tablespace if exists tpName;

##### **其它**

## 用户/角色对象管理

### **功能简述**



### **系统部件**

#### **主要程序文件**

#### **关键数据结构说明**

|  |  |  |  |
| --- | --- | --- | --- |
| **字典** | **说明** | **类型** | **是否必填** |
| 名称 |  | 基本信息 | 是 |
| 密码 |  | 基本信息 | 是 |
| 权限 | 所有、继承、创建角色、审计管理员、登录、复制、创建数据库、系统管理员 | 基本信息 | 是 |
| 描述 |  | 基本信息 | 否 |
| 连接限制 |  | 基本信息 | 否 |
| 开始日期 |  | 基本信息 | 否 |
| 结束日期 |  | 基本信息 | 否 |
| 资源池 |  | 基本信息 | 否 |
| 角色组 |  | 基本信息 | 否 |
| 管理员组 |  | 基本信息 | 否 |

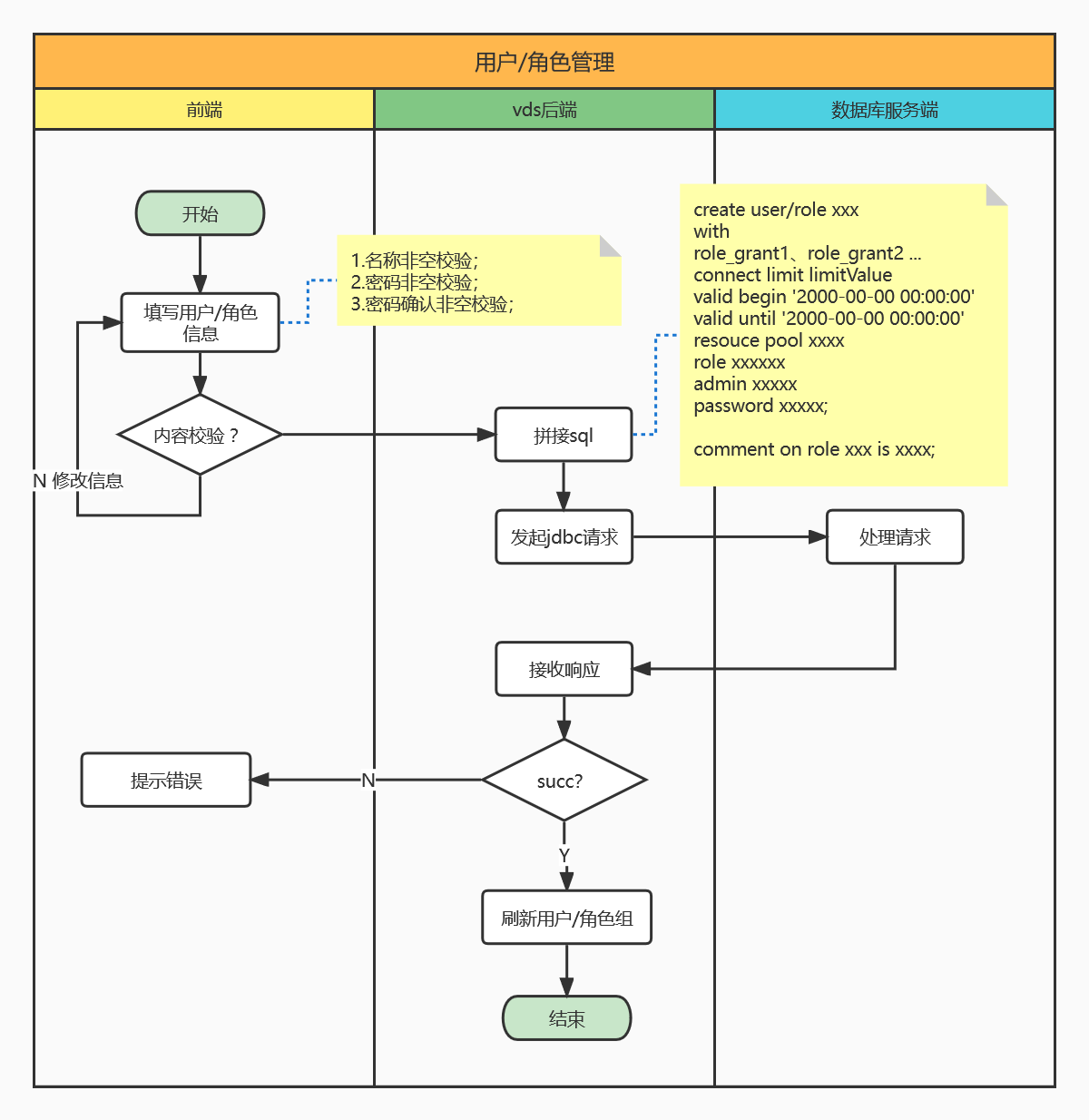
### **执行方案**

#### **数据连接对象管理**

##### **实现逻辑描述**

##### **流程/交互设计**

* **创建用户/角色交互流程**

****

##### **关键算法/SQL**

* **创建用户/角色**

create user/role xxx

with

role\_grant1、role\_grant2 ...

connect limit limitValue

valid begin '2000-00-00 00:00:00'

valid until '2000-00-00 00:00:00'

resouce pool xxxx

role xxxxxx

admin xxxxx

password xxxxx;

-- comment for role

comment on role xxx is xxxx;

* **权限关键词**
* **SYSADMIN**
* **AUDITADMIN**
* **CRAETEDB**
* **CRAETEROLE**
* **INHERIT**
* **LOGIN**
* **REPLICATION**
* **锁定/解锁**

Alter user|role xxx account unlock|lock ;

**PS：管理员才有控制是否锁定的能力**

* **重命名**

Alter user|role oldName TO newName;

##### **其它**

## SQL终端编辑器

### **功能简述**

终端SQL编辑器

* 批量执行sql；
* 历史执行sql记录；(序号/锁定状态/Sql语句/获取记录数/开始时间/执行时间/数据库/状态;

### **系统部件**

#### **主要程序文件**

#### **关键数据结构说明**

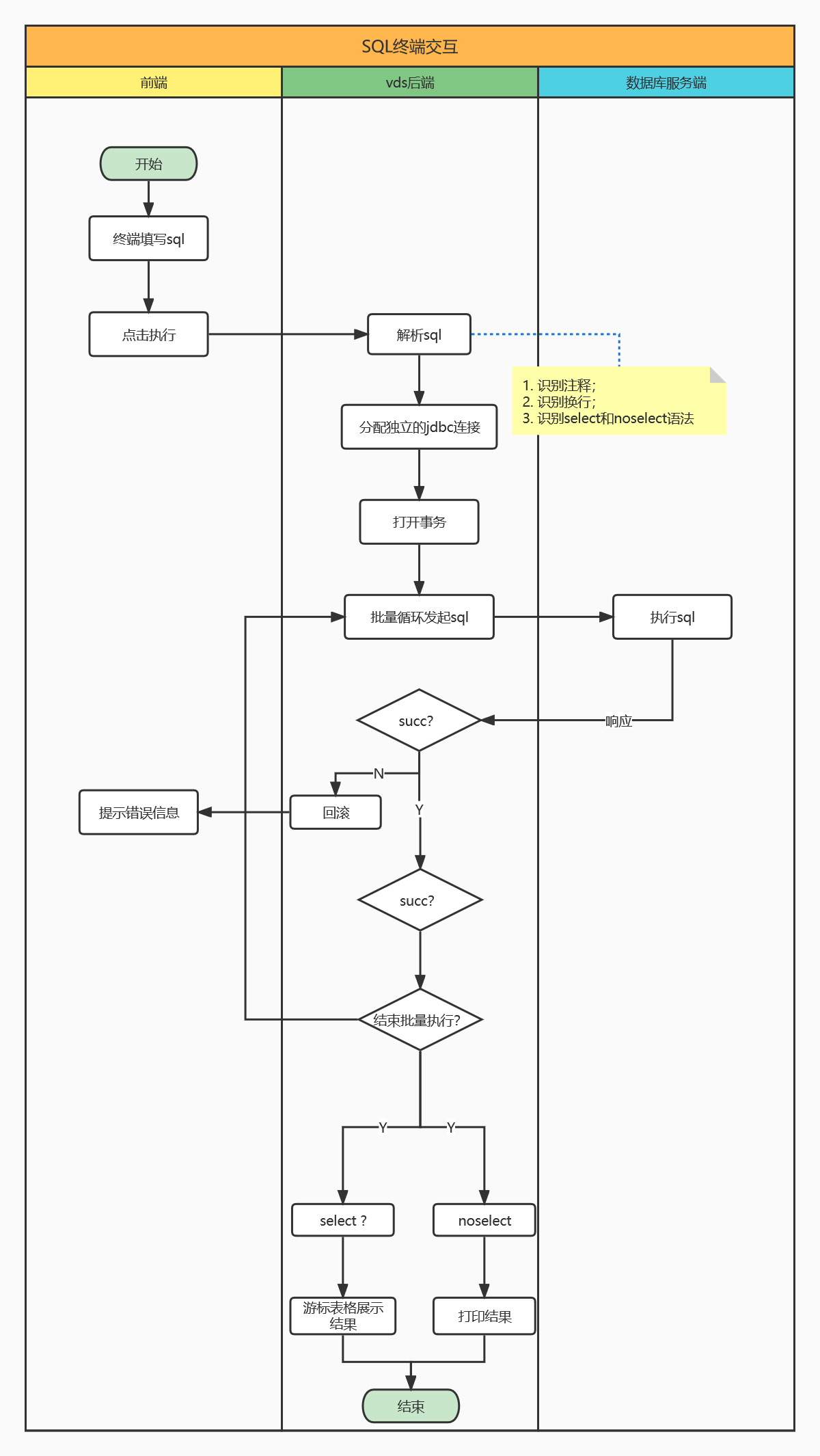
### **执行方案**

#### **SQL终端实现方案**

##### **实现逻辑描述**

##### **流程/交互设计**

* **终端执行交互**

****

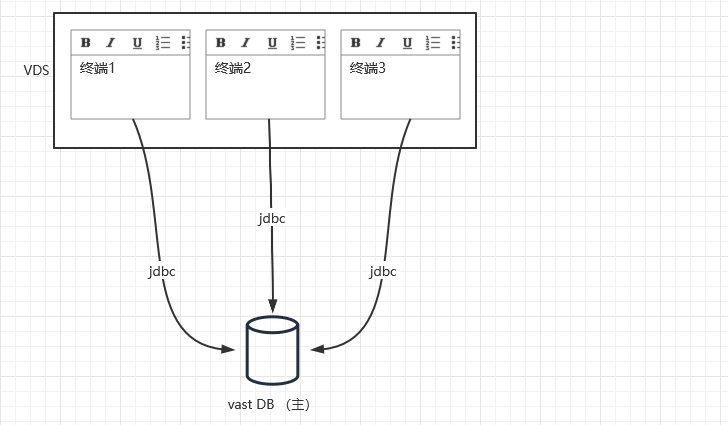
* **窗口和jdbc连接**

为了避免不同终端间执行sql的相互影响，每个终端窗口会分配独立的jdbc连接。

多个终端间的查询互不打扰。

如：终端1在执行大数据的select，导致长时间等待；

此时终端2不受影响去执行自身的sql任务；



##### **关键逻辑/SQL**

* **关于sql格式化**

数据格式化的实现借助jsqlparser工具，对sql进行分行展示；

* **历史sql保存方式**

Sql缓存至内存，不序列化至磁盘；

##### **其它**

## 对象搜索

### **功能简述**

### **系统部件**

#### **主要程序文件**

#### **关键数据结构说明**

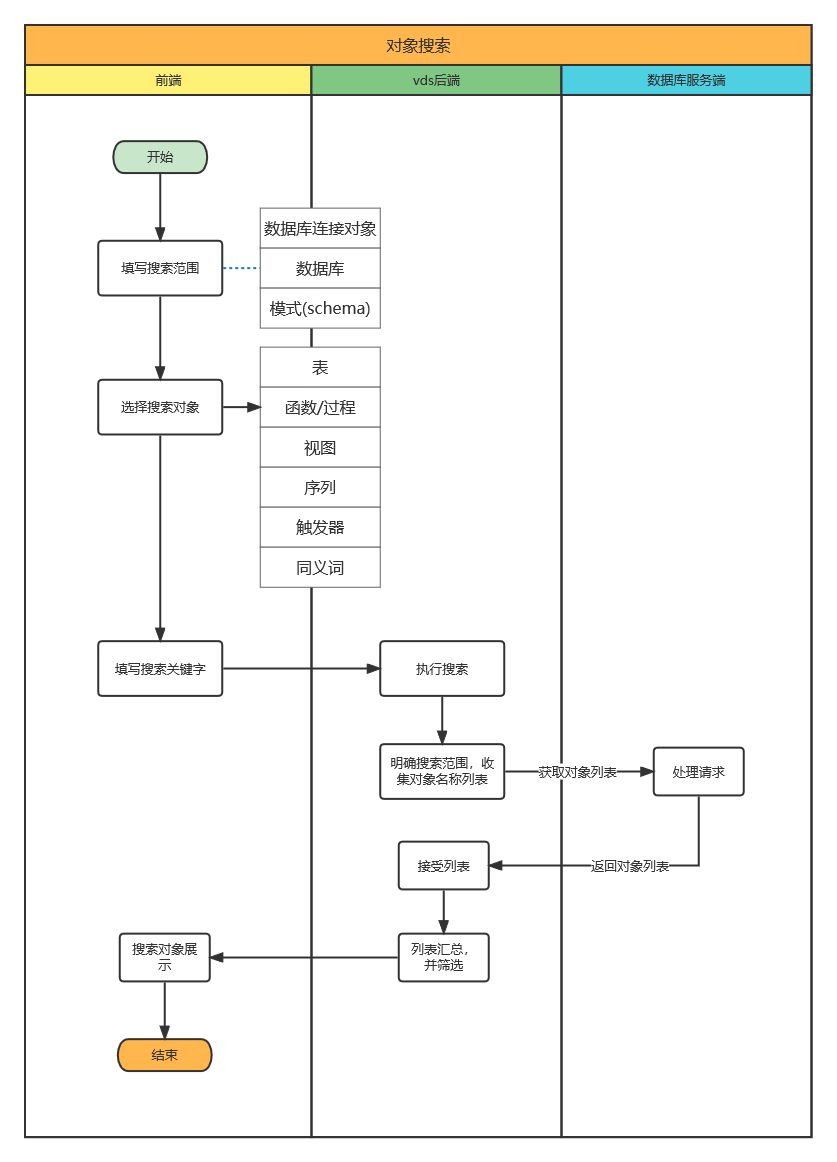
### **执行方案**

#### **数据连接对象管理**

##### **实现逻辑描述**

##### **流程/交互设计**

* **对象搜索交互流程**



##### **关键算法/SQL**

##### **其它**

## 运行/调试

### **功能简述**

### **系统部件**

#### **主要程序文件**

#### **关键数据结构说明**

### **执行方案**

#### **调试实现方案**

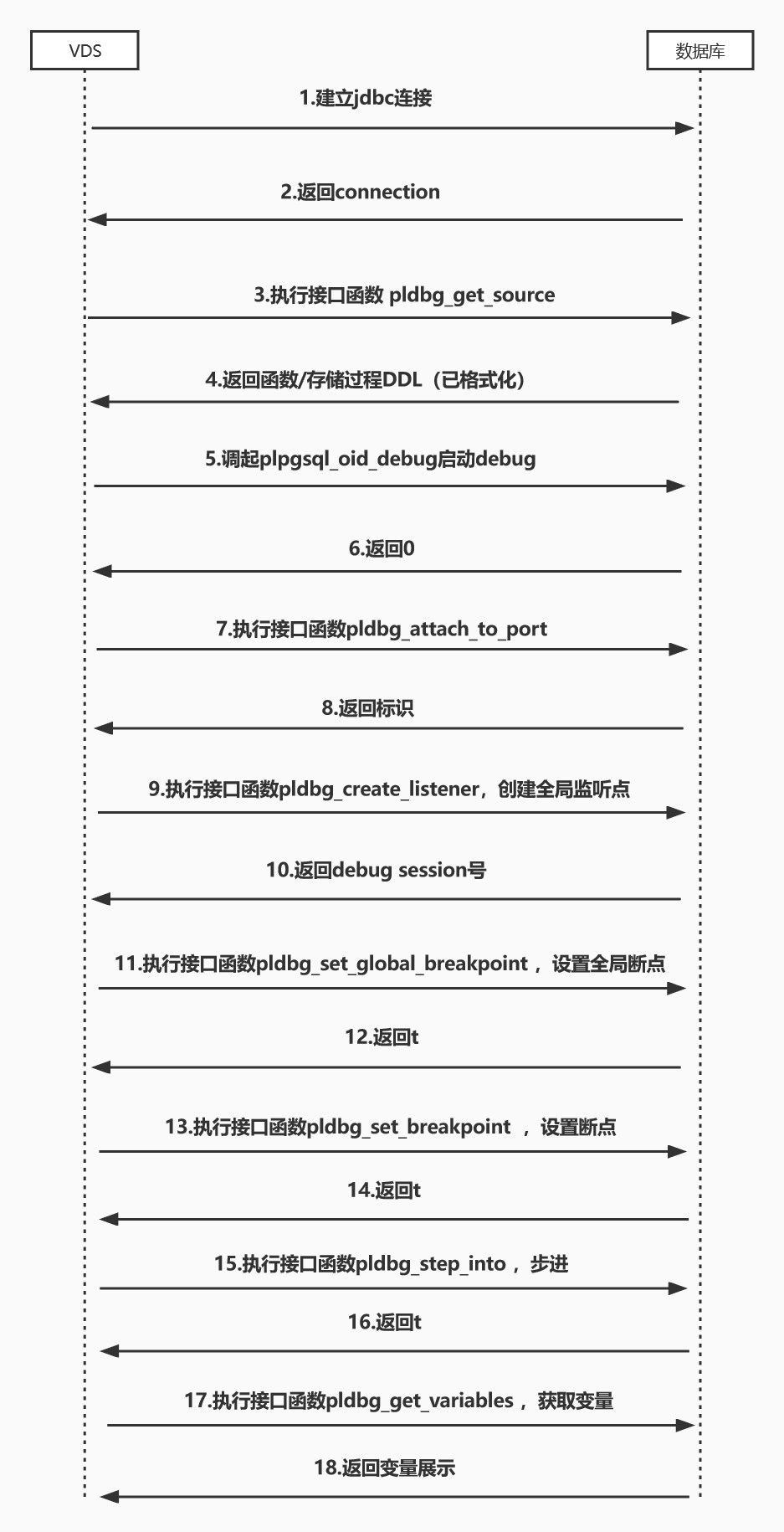
##### **实现逻辑描述**

函数/存储过程调试依赖于调试插件提供的接口函数

|  |  |
| --- | --- |
| **接口函数** | **简介** |
| [plpgsql\_oid\_debug 函数](https://blog.csdn.net/arthemis_14/article/details/126600940#plpgsql-oid-debug) | 创建局部监听 |
| [pldbg\_attach\_to\_port 函数](https://blog.csdn.net/arthemis_14/article/details/126600940#pldbg-attach-to-port) | 连接目标后端 |
| [pldbg\_create\_listener 函数](https://blog.csdn.net/arthemis_14/article/details/126600940#pldbg-create-listener) | 创建全局监听 |
| [pldbg\_set\_global\_breakpoint 函数](https://blog.csdn.net/arthemis_14/article/details/126600940#pldbg-set-global-breakpoint) | 设置全局断点 |
| [pldbg\_wait\_for\_target 函数](https://blog.csdn.net/arthemis_14/article/details/126600940#pldbg-wait-for-target) | 等待连接 |
| [pldbg\_set\_breakpoint 函数](https://blog.csdn.net/arthemis_14/article/details/126600940#pldbg-set-breakpoint) | 设置断点 |
| [pldbg\_get\_breakpoints 函数](https://blog.csdn.net/arthemis_14/article/details/126600940#pldbg-get-breakpoints) | 查询断点 |
| [pldbg\_drop\_breakpoint 函数](https://blog.csdn.net/arthemis_14/article/details/126600940#pldbg-drop-breakpoint) | 删除断点 |
| [pldbg\_wait\_for\_breakpoint 函数](https://blog.csdn.net/arthemis_14/article/details/126600940#pldbg-wait-for-breakpoint) | 返回建立连接时断点所在行的信息 |
| [pldbg\_step\_into 函数](https://blog.csdn.net/arthemis_14/article/details/126600940#pldbg-step-into) | 单步进入（step into） |
| [pldbg\_step\_over 函数](https://blog.csdn.net/arthemis_14/article/details/126600940#pldbg-step-over) | 单步跳过（step over） |
| [pldbg\_continue 函数](https://blog.csdn.net/arthemis_14/article/details/126600940#pldbg-continue) | 继续执行（continue） |
| [pldbg\_abort\_target 函数](https://blog.csdn.net/arthemis_14/article/details/126600940#pldbg-abort-target) | 中止执行 |
| [pldbg\_get\_variables 函数](https://blog.csdn.net/arthemis_14/article/details/126600940#pldbg-get-variables) | 查看变量 |
| [pldbg\_deposit\_value 函数](https://blog.csdn.net/arthemis_14/article/details/126600940#pldbg-deposit-value) | 修改变量 |
| [pldbg\_get\_stack 函数](https://blog.csdn.net/arthemis_14/article/details/126600940#pldbg-get-stack) | 查看堆栈信息 |
| [pldbg\_select\_frame 函数](https://blog.csdn.net/arthemis_14/article/details/126600940#pldbg-select-frame) | 追溯堆栈信息 |
| [pldbg\_get\_source 函数](https://blog.csdn.net/arthemis_14/article/details/126600940#pldbg-get-source) | 查询源代码信息 |
| [pldbg\_get\_proxy\_info 函数](https://blog.csdn.net/arthemis_14/article/details/126600940#pldbg-get-proxy-info) | 查询当前系统版本信息 |

##### **流程/交互设计**

* **调式的时序图**

****

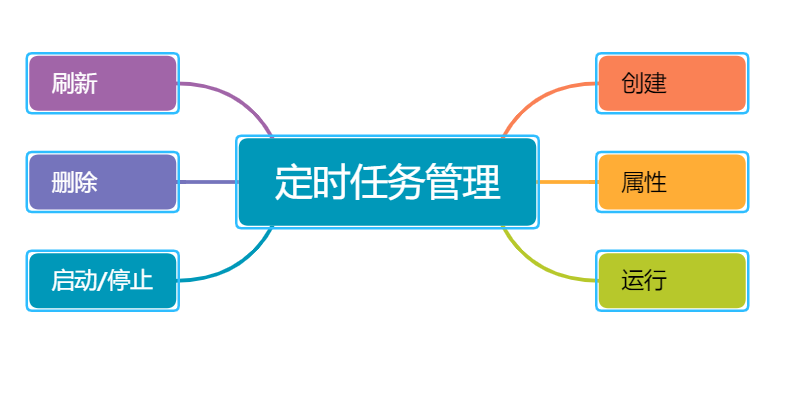
##### **异常场景**

调式过程中，调起接口函数返回异常，则中断调式

##### **其它**

## 定时任务管理

### **功能简述**



### **系统部件**

#### **主要程序文件**

#### **关键数据结构说明**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字典** | **说明** | **类型** | **父节点** | **必填** |
| 定时任务编号 | 定时任务编号是定时任务的唯一标识 | 基本信息 |  | 是 |
| 定时任务内容 | 定时任务的执行内容 | 基本信息 |  | 是 |
| 下次运行时间 | 定时任务下一次的运行时间 | 基本信息 |  | 是 |
| 间隔 | 定时任务的执行间隔，频次 | 基本信息 |  | 是 |
| 状态 | 定时任务的当前状态(启动、停止) | 基本信息 |  | 是 |
| 创建者 | 创建定时任务的人 | 基本信息 |  | 否 |
| 执行者 | 执行定时任务的人 | 基本信息 |  | 否 |
| 上一次运行结束时间 | 上一次定时任务运行结束时间 | 列表 |  | 否 |

### **执行方案**

* **创建定时任务**

1) 根据用户填写定时任务信息后，后台发起select pkg\_service.job\_submit(xx)操作。

* **停止/启用定时任务**

用户填写定时任务信息后，后台发起select pkg\_service.job\_finish(任务编号,true/false)操作；

* **查看定时任务（属性）**

根据系统表pg\_job查询指定任务编号的定时任务，获取相关详情。

如查看定时任务编号100，则拼接sql：select \* from pg\_job .where job\_id = 100 ，交由终端执行并展示结果。

* **运行定时任务**

执行select pkg\_service.sql\_run(任务编号)

* **删除定时任务**

select pkg\_service.job\_cancel(任务编号)

* **刷新**

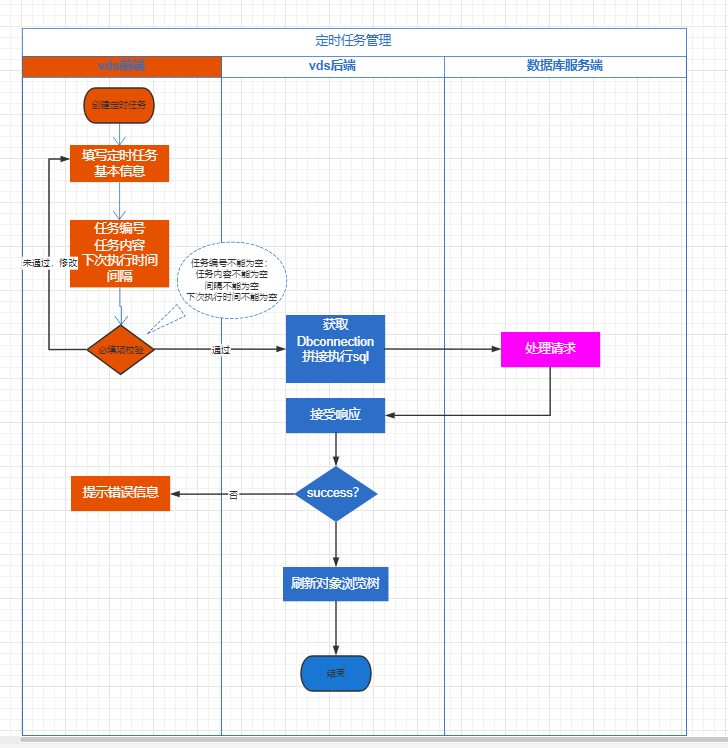
重新加载定时任务树节点：

#### **定时任务管理**

##### **实现逻辑描述**

##### **流程/交互设计**

* **创建定时任务交互流程**



##### **关键算法/SQL**

关于定时任务元数据

select \* from pg\_job

where job\_id = %

##### **其它**