(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 118092897 A
(43)申请公布日 2024.05.28

(21)申请号 202410223122.0

(22)申请日 2024.02.28

(71)申请人 中国工商银行股份有限公司
地址 100140 北京市西城区复兴门内大街55号

(72)发明人 朱婷婷

(74)专利代理机构 北京三友知识产权代理有限公司 11127
专利代理师 王天尧

(51)Int.Cl.
G06F 8/34(2018.01)
G06F 8/36(2018.01)
G06F 8/38(2018.01)
G06F 16/242(2019.01)
G06F 16/25(2019.01)

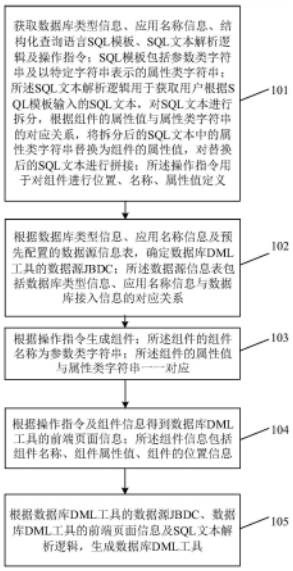
权利要求书2页 说明书10页 附图6页

(54)发明名称

数据库DML工具的低代码生成方法及装置

(57)摘要

本发明公开了一种数据库DML工具的低代码生成方法及装置,涉及金融或其他技术领域,该方法包括:获取数据库类型信息、应用名称信息、结构化查询语言SQL模板、SQL文本解析逻辑及操作指令;根据数据库类型信息确定DML工具的数据源JBDC;根据操作指令生成组件;得到数据库DML工具的前端页面信息;根据数据库DML工具的数据源JBDC、数据库DML工具的前端页面信息及SQL文本解析逻辑,生成数据库DML工具,本发明可以节约数据库DML工具的开发人力成本,避免由于开发人员认知偏差而产生错误,提高开发效率。



CN 118092897 A

1.一种数据库数据操作语言DML工具的低代码生成方法,其特征在于,包括:

获取数据库类型信息、应用名称信息、结构化查询语言SQL模板、SQL文本解析逻辑及操作指令;所述SQL模板包括参数类字符串及以特定字符串表示的属性类字符串;所述SQL文本解析逻辑用于获取用户根据SQL模板输入的SQL文本,对SQL文本进行拆分,根据组件的属性值与属性类字符串的对应关系,将拆分后的SQL文本中的属性类字符串替换为组件的属性值,对替换后的SQL文本进行拼接;所述操作指令用于对组件进行位置、名称、属性值定义;

根据数据库类型信息、应用名称信息及预先配置的数据源信息表,确定数据库DML工具的数据源JBDC;所述数据源信息表包括数据库类型信息、应用名称信息与数据库接入信息的对应关系;

根据操作指令生成组件;所述组件的组件名称为参数类字符串;所述组件的属性值与属性类字符串一一对应;

根据操作指令及组件信息,得到数据库DML工具的前端页面信息;所述组件信息包括组件名称、组件属性值、组件的位置信息;

根据数据库DML工具的数据源JBDC、数据库DML工具的前端页面信息及SQL文本解析逻辑,生成数据库DML工具。

2.如权利要求1所述的方法,其特征在于,根据数据库类型信息、应用名称信息及预先配置的数据源信息表,确定数据库DML工具的数据源JBDC,包括:

根据数据库类型信息确定数据源Java数据库连接JBDC模板;数据源JBDC模板包括待客户化字段;

根据数据库类型信息、应用名称信息及预先配置的数据源信息表,对数据源JBDC模板中的待客户化字段进行客户化,得到数据库DML工具的数据源JBDC。

3.如权利要求2所述的方法,其特征在于,根据数据库类型信息、应用名称信息及预先配置的数据源信息表,对数据源JBDC模板中的待客户化字段进行客户化,得到数据库DML工具的数据源JBDC,包括:

根据数据库类型信息、应用名称信息及预先配置的数据源信息表,确定数据库接入信息;所述数据库接入信息包括数据库ip地址信息、数据库端口信息、数据库实例名信息的其中一种或任意组合;

根据数据库接入信息对数据源JBDC模板中的待客户化字段进行客户化,得到数据库DML工具的数据源JBDC。

4.如权利要求3所述的方法,其特征在于,根据数据库接入信息对数据源JBDC模板中的待客户化字段进行客户化,包括:

将数据源JBDC模板中的待客户化字段替换为数据库接入信息。

5.如权利要求1所述的方法,其特征在于,还包括:

根据操作指令生成SQL文本组件;

根据数据库DML工具的数据源JBDC、数据库DML工具的前端页面信息及SQL文本解析逻辑,生成数据库DML工具,包括:

利用SQL文本解析逻辑获取用户通过SQL文本组件输入的SQL文本,对SQL文本的逻辑进行解析,对SQL文本按照字符串类型进行拆分,根据组件的属性值与属性类字符串的对应关

系,将拆分后的SQL文本中的属性类字符串替换为组件的属性值,对替换处理后的SQL文本按照解析出的SQL文本的逻辑进行拼接;所述字符串类型包括语法类字符串、参数类字符串、属性类字符串。

6.如权利要求5所述的方法,其特征在于,对SQL文本按照字符串类型进行拆分,根据组件的属性值与属性类字符串的对应关系,将拆分后的SQL文本中的属性类字符串替换为组件的属性值,包括:

利用SQL文本解析逻辑及正则表达式,对SQL文本按照字符串类型进行拆分;

利用SQL文本解析逻辑对拆分后的SQL文本中的属性类字符串进行标记,根据标记对SQL文本中的属性类字符串进行查找,根据组件的属性值与属性类字符串的对应关系,将查找到的属性类字符串替换为组件的属性值。

7.如权利要求1所述的方法,其特征在于,根据操作指令生成组件,包括:

根据操作指令确定组件的位置、名称及属性值;

根据组件的位置、名称及属性值生成组件。

8.一种数据库DML工具的低代码生成装置,其特征在于,包括:

获取模块,用于获取数据库类型信息、应用名称信息、结构化查询语言SQL模板、SQL文本解析逻辑及操作指令;所述SQL模板包括参数类字符串及以特定字符串表示的属性类字符串;所述SQL文本解析逻辑用于获取用户根据SQL模板输入的SQL文本,对SQL文本进行拆分,根据组件的属性值与属性类字符串的对应关系,将拆分后的SQL文本中的属性类字符串替换为组件的属性值,对替换后的SQL文本进行拼接;所述操作指令用于对组件进行位置、名称、属性值定义;

客户化模块,用于根据数据库类型信息、应用名称信息及预先配置的数据源信息表,确定数据库数据操作语言DML工具的数据源JBDC;所述数据源信息表包括数据库类型信息、应用名称信息与数据库接入信息的对应关系;

组件生成模块,用于根据操作指令生成组件;所述组件的组件名称为参数类字符串;所述组件的属性值与属性类字符串一一对应;

前端页面信息生成模块,用于根据操作指令及组件信息,得到数据库DML工具的前端页面信息;所述组件信息包括组件名称、组件属性值、组件的位置信息;

数据库DML工具生成模块,用于根据数据库DML工具的数据源JBDC、数据库DML工具的前端页面信息及SQL文本解析逻辑,生成数据库DML工具。

9.一种计算机设备,包括存储器、处理器及存储在存储器上并可在处理器上运行的计算机程序,其特征在于,所述处理器执行所述计算机程序时实现权利要求1至7任一所述方法。

10.一种计算机可读存储介质,其特征在于,所述计算机可读存储介质存储有计算机程序,所述计算机程序被处理器执行时实现权利要求1至7任一所述方法。

11.一种计算机程序产品,其特征在于,所述计算机程序产品包括计算机程序,所述计算机程序被处理器执行时实现权利要求1至7任一所述方法。

数据库DML工具的低代码生成方法及装置

技术领域

[0001] 本发明涉及金融技术领域,尤其涉及一种数据库DML工具的低代码生成方法及装置。

背景技术

[0002] 本部分旨在为本发明实施例提供背景或上下文。此处的描述不因为包括在本部分中就承认是现有技术。

[0003] 低代码开发是一种通过可视化进行应用程序开发的方法,使具有不同经验水平的开发人员可以通过图形化的用户界面,使用拖拽组件和模型驱动的逻辑来创建网页和移动应用程序。低代码开发平台使非技术开发人员可不必编写代码,而是将传统IT架构抽象化来支持专业开发人员。业务部门和IT部门的开发人员可以共同创建、迭代和发布应用程序,花费的时间比传统方式更少,这类称为APAAS(Application Platform as a Service,应用程序平台即服务)。

[0004] 数字经济已经逐渐成为社会经济发展过程中不可或缺的一部分。而数据作为金融行业的关键生产要素,在开发及测试工作中都离不开对数据库的操作。通常对数据库的操作有增、删、改、查,又被称为数据操纵语言(Data Manipulation Language,DML)。DML是对数据库其中的对象和数据运行访问工作的编程语句,通常是数据库专用编程语言之中的一个子集,例如在信息软件产业通行标准的SQL(Structured Query Language,结构化查询语言)语言中,以INSERT、UPDATE、DELETE三种指令为核心,分别代表插入(意指新增或创建)、更新(修改)与删除(销毁)。

[0005] 但由于金融行业中数据作为内部保密材料,数据库的DML权限不能直接对所有工作人员开放,需以固化数据库变更SQL并将其中部分字段客户化的方式,也就是开发成固定模式的数据库DML工具的方式达到控制数据库操作权限的目的,也因此在开发和测试的过程中难免带来阻碍。而面对数据库DML变更需求的日益增长,如果公司IT人员采用传统研发模式进行DML工具开发,将需要更多的开发人员以解决对大量创新性服务的需要,但由于高昂的人力资源成本,且由于开发人员认知偏差而产生错误,会使得公司更加缺乏竞争力。

发明内容

[0006] 本发明实施例提供一种数据库DML工具的低代码生成方法,用以节约数据库DML工具的开发人力成本,避免由于开发人员认知偏差而产生错误,提高开发效率,该方法包括:

[0007] 获取数据库类型信息、应用名称信息、结构化查询语言SQL模板、SQL文本解析逻辑及操作指令;所述SQL模板包括参数类字符串及以特定字符串表示的属性类字符串;所述SQL文本解析逻辑用于获取用户根据SQL模板输入的SQL文本,对SQL文本进行拆分,根据组件的属性值与属性类字符串的对应关系,将拆分后的SQL文本中的属性类字符串替换为组件的属性值,对替换后的SQL文本进行拼接;所述操作指令用于对组件进行位置、名称、属性值定义;

[0008] 根据数据库类型信息、应用名称信息及预先配置的数据源信息表,确定数据库DML工具的数据源JBDC;所述数据源信息表包括数据库类型信息、应用名称信息与数据库接入信息的对应关系;

[0009] 根据操作指令生成组件;所述组件的组件名称为参数类字符串;所述组件的属性值与属性类字符串一一对应;

[0010] 根据操作指令及组件信息,得到数据库DML工具的前端页面信息;所述组件信息包括组件名称、组件属性值、组件的位置信息;

[0011] 根据数据库DML工具的数据源JBDC(Java Database Connectivity,Java数据库连接)、数据库DML工具的前端页面信息及SQL文本解析逻辑,生成数据库DML工具。

[0012] 本发明实施例还提供一种数据库DML工具的低代码生成装置,用以节约数据库DML工具的开发人力成本,避免由于开发人员认知偏差而产生错误,提高开发效率,该装置包括:

[0013] 获取模块,用于获取数据库类型信息、应用名称信息、结构化查询语言SQL模板、SQL文本解析逻辑及操作指令;所述SQL模板包括参数类字符串及以特定字符串表示的属性类字符串;所述SQL文本解析逻辑用于获取用户根据SQL模板输入的SQL文本,对SQL文本进行拆分,根据组件的属性值与属性类字符串的对应关系,将拆分后的SQL文本中的属性类字符串替换为组件的属性值,对替换后的SQL文本进行拼接;所述操作指令用于对组件进行位置、名称、属性值定义;

[0014] 客户化模块,用于根据数据库类型信息、应用名称信息及预先配置的数据源信息表,确定数据库数据操作语言DML工具的数据源JBDC;所述数据源信息表包括数据库类型信息、应用名称信息与数据库接入信息的对应关系;

[0015] 组件生成模块,用于根据操作指令生成组件;所述组件的组件名称为参数类字符串;所述组件的属性值与属性类字符串一一对应;

[0016] 前端页面信息生成模块,用于根据操作指令及组件信息,得到数据库DML工具的前端页面信息;所述组件信息包括组件名称、组件属性值、组件的位置信息;

[0017] 数据库DML工具生成模块,用于根据数据库DML工具的数据源JBDC、数据库DML工具的前端页面信息及SQL文本解析逻辑,生成数据库DML工具。

[0018] 本发明实施例与现有技术中采用传统研发模式进行DML工具开发的技术方案相比,通过获取数据库类型信息、应用名称信息、结构化查询语言SQL模板、SQL文本解析逻辑及操作指令;所述SQL模板包括参数类字符串及以特定字符串表示的属性类字符串;所述SQL文本解析逻辑用于获取用户根据SQL模板输入的SQL文本,对SQL文本进行拆分,根据组件的属性值与属性类字符串的对应关系,将拆分后的SQL文本中的属性类字符串替换为组件的属性值,对替换后的SQL文本进行拼接;所述操作指令用于对组件进行位置、名称、属性值定义;根据数据库类型信息、应用名称信息及预先配置的数据源信息表,确定数据库DML工具的数据源JBDC;所述数据源信息表包括数据库类型信息、应用名称信息与数据库接入信息的对应关系;根据操作指令生成组件;所述组件的组件名称为参数类字符串;所述组件的属性值与属性类字符串一一对应;根据操作指令及组件信息,得到数据库DML工具的前端页面信息;所述组件信息包括组件名称、组件属性值、组件的位置信息;根据数据库DML工具的数据源JBDC、数据库DML工具的前端页面信息及SQL文本解析逻辑,生成数据库DML工具,

可以节约数据库DML工具的开发人力成本,避免由于开发人员认知偏差而产生错误,提高开发效率。

附图说明

[0019] 为了更清楚地说明本发明实施例或现有技术中的技术方案,下面将对实施例或现有技术描述中所需要使用的附图作简单地介绍,显而易见地,下面描述中的附图仅仅是本发明的一些实施例,对于本领域普通技术人员来讲,在不付出创造性劳动的前提下,还可以根据这些附图获得其他的附图。在附图中:

[0020] 图1为本发明实施例中数据库DML工具的低代码生成方法的流程图;

[0021] 图2为本发明实施例中数据库DML工具的低代码生成系统的框架图;

[0022] 图3为本发明实施例中数据库DML工具的低代码生成系统的技术框架图;

[0023] 图4为本发明实施例中数据库DML工具的低代码生成系统的实现流程图;

[0024] 图5为本发明实施例中数据库DML工具的低代码生成装置的示意图;

[0025] 图6为本发明实施例中计算机设备的示意图。

具体实施方式

[0026] 为使本发明实施例的目的、技术方案和优点更加清楚明白,下面结合附图对本发明实施例做进一步详细说明。在此,本发明的示意性实施例及其说明用于解释本发明,但并不作为对本发明的限定。

[0027] 为了避免金融行业开发、测试过程中日益增长的对数据库DML工具需求带来的传统研发压力,节约数据库DML工具的开发人力成本,避免由于开发人员认知偏差而产生错误,提高开发效率,本发明提出一种基于SQL拆分的数据库DML工具的低代码开发方法。图1为本发明实施例中数据库DML工具的低代码生成方法的流程图,如图1所示,数据库DML工具的低代码生成方法可以包括:

[0028] 步骤101,获取数据库类型信息、应用名称信息、结构化查询语言SQL模板、SQL文本解析逻辑及操作指令;所述SQL模板包括参数类字符串及以特定字符串表示的属性类字符串;所述SQL文本解析逻辑用于获取用户根据SQL模板输入的SQL文本,对SQL文本进行拆分,根据组件的属性值与属性类字符串的对应关系,将拆分后的SQL文本中的属性类字符串替换为组件的属性值,对替换后的SQL文本进行拼接;所述操作指令用于对组件进行位置、名称、属性值定义;

[0029] 步骤102,根据数据库类型信息、应用名称信息及预先配置的数据源信息表,确定数据库DML工具的数据源JBDC;所述数据源信息表包括数据库类型信息、应用名称信息与数据库接入信息的对应关系;

[0030] 步骤103,根据操作指令生成组件;所述组件的组件名称为参数类字符串;所述组件的属性值与属性类字符串一一对应;

[0031] 步骤104,根据操作指令及组件信息,得到数据库DML工具的前端页面信息;所述组件信息包括组件名称、组件属性值、组件的位置信息;

[0032] 步骤105,根据数据库DML工具的数据源JBDC、数据库DML工具的前端页面信息及SQL文本解析逻辑,生成数据库DML工具。

[0033] 在一个实施例中,数据库DML工具的低代码开发方法可以基于一个系统或平台,本发明实施例提出一种基于SQL拆分的数据库DML工具的低代码开发平台,通过可视化拖拽控件的形式完成数据库DML工具的开发。

[0034] 在本实施例中,数据库DML工具的低代码开发平台一共可以包括四个模块,一是数据源自动封装模块,二是SQL语句解析模块,三是工具界面可视化设计模块,四是前后端集成模块。使用者即使无前后端开发的知识基础,亦可利用本发明实施例中的方法和平台快速开发数据库DML工具,供他人完成数据库的增删改查。节约了传统开发模式下的开发人力成本,也提升了企业数字化水平。

[0035] 图2为本发明实施例中数据库DML工具的低代码生成系统的框架图,如图2所示,数据库DML工具的低代码生成系统可以包含存储层、后端、前端,使用者可以通过浏览器以http的形式访问平台前端完成工具开发。其中,存储层可以采用MySQL数据库;后端可以采用Linux服务器或Java平台,并实现人员管理、工作台、权限管理、页面生成、审批管理、开发流程、自定义组件及日志管理功能;前端可以使用Vue框架、HTML(HyperText Markup Language,超文本标记语言)技术、JS(JavaScript,Java脚本编程语言)等完成对数据库DML工具前端页面的拖拽生成。

[0036] 图3为本发明实施例中数据库DML工具的低代码生成系统的技术框架图,如图3所示:

[0037] 数据源自动封装模块可以完成数据库类型判断、获取数据库信息和拼接数据源JDBC三部分功能。数据库DML工具的低代码生成系统支持多种数据库类型,不同类型数据库的JDBC会存在微小差异,所以在数据源自动封装模块中需要对数据库类型进行判断。各JDBC会设置固定模板,其中数据库地址、数据库端口及数据库实例名需要进行客户化,这依赖于数据库DML工具的低代码生成系统的使用者(系统使用者)提前配置。最后将实际的数据库地址、端口以及实例名完成替换后得到最终数据库DML工具使用的数据源JDBC。

[0038] SQL语句解析模块需要完成SQL文本获取、SQL语句正则拆分、SQL语句解析的功能。对于数据库DML工具的低代码生成系统的使用者利用数据库DML工具的低代码生成系统开发的页面需要实现的数据库DML操作,也即对应SQL语句,在使用者设计工具的时候就需定义。SQL语句解析模块会存储完整的SQL文本,以便后续拆分。SQL文本拆分可以通过正则实现,主要将SQL文本拆分为三个部分,一部分为语法类字符串,一部分为参数类字符串,一部分为属性类字符串。然后通过SQL语句解析将语句还原,并将定位到的属性类字符串进行标记,以便后续前后端集成。

[0039] 工具界面可视化设计模块包括组件拖拽设计、组件客户化和传递界面信息三个功能。数据库DML工具的低代码生成系统的使用者可以采用可视化的方式拖拽组件设计数据库DML工具UI界面,其中各组件需要完成客户化,以对应SQL语句解析模块中参数类字符串及属性类字符串。最后将界面信息,包括组件信息及客户化信息打包,传送到后端以便后续前后端集成。

[0040] 前后端集成模块需要完成SQL语句客户化及前后端逻辑拼接的功能。解析后的SQL语句中有属性类字符串需要与前端组件中的组件属性进行一对一拼接,这样在数据库DML工具用户在前端输入SQL客户化字段之后,就能通过该逻辑传送到后端,拼接出用户最终执行的SQL语句。与此同时也通过SQL语句中的属性类字符串与前端组件属性的对应关系,完

成前后端集成。

[0041] 图4为本发明实施例中数据库DML工具的低代码生成系统的实现流程图展示了系统的实现流程。系统使用者首先需要完成数据源配置,包括数据库类型、数据库地址、数据库端口以及数据库实例,并配置对应的关键字段应用名。这样在定义界面的时候只需要选择数据库类型及应用名,即可定位到对应的数据库信息。接下来系统使用者需要录入SQL模板,其中属性类字符串需要用特殊字符代替,这样在实现SQL解析的时候即可根据特殊字符串定位到属性类字符串。又由于SQL语句中定义字段的格式固定为“字段名=字段值”,当字段值被判定为为属性类字符串时,位于它“=”前边的字符串则被标记为参数类字符串。在组件设计的过程中,同样客户化字段属性用SQL语句中对应的特殊字符代替。当有多个客户化字段时,需要保证SQL语句属性类字符串与组件属性二者是一对一的,这样才能保证客户化的值一致且一一对应。完成后端逻辑及前端界面设计之后点击生成工具即可完成前后端逻辑集成,从而生成数据库DML工具。

[0042] 使用数据库DML工具的低代码生成系统可以解决公司开发人员数量不足的问题,和产生大量需求文档的旧技术方法一样,使用数据库DML工具的低代码生成系统业务应用能够更方便地把需求直接注入应用开发环境中,变更需求也不会由于公司IT团队认识的偏差而产生错误。

[0043] 在一个实施例中,由于生成数据库DML工具需要接入多种数据库类型,而不同类型数据库的JDBC会存在微小差异,因此,需要对数据库类型进行判断,并预先配置数据源信息表,以快速定位到对应的数据库信息。数据库DML工具的低代码生成方法的步骤102中,根据数据库类型信息、应用名称信息及预先配置的数据源信息表,确定数据库DML工具的数据源JBDC,可以包括:根据数据库类型信息确定数据源Java数据库连接JBDC模板;数据源JBDC模板包括待客户化字段;根据数据库类型信息、应用名称信息及预先配置的数据源信息表,对数据源JBDC模板中的待客户化字段进行客户化,得到数据库DML工具的数据源JBDC。例如,开发人员需要开发一个针对MySQL数据库的数据库DML工具,根据数据库类型信息,即MySQL,确定数据源JBDC模板,再根据数据库类型信息:MySQL、应用名称信息:XXX1,对数据源JBDC模板中的待客户化字段进行客户化。

[0044] 在本实施例中,为了更简单、快捷、高效地完成数据库SQL工具的数据源配置,根据数据库类型信息、应用名称信息及预先配置的数据源信息表,对数据源JBDC模板中的待客户化字段进行客户化,得到数据库DML工具的数据源JBDC,可以包括:根据数据库类型信息、应用名称信息及预先配置的数据源信息表,确定数据库接入信息;所述数据库接入信息包括数据库IP地址信息、数据库端口信息、数据库实例名信息的其中一种或任意组合;根据数据库接入信息对数据源JBDC模板中的待客户化字段进行客户化,得到数据库DML工具的数据源JBDC。例如,开发人员需要开发一个针对MySQL数据库的数据库DML工具,输入数据库类型信息为MySQL、应用名称信息为XXX1,系统在预先配置的数据源信息表中查找出对应的数据库IP地址信息为XX.XX.XX.XX、数据库端口信息为端口1、数据库实例名信息为XXX。再根据数据库类型信息为MySQL确定针对MySQL数据库的数据源JBDC模板,利用数据库ip地址信息(XX.XX.XX.XX)、数据库端口信息(端口1)、数据库实例名信息(XXX)对数据源JBDC模板中的待客户化字段进行客户化,得到数据库DML工具的数据源JBDC。

[0045] 在本实施例中,为实现低代码进行数据源客户化,根据数据库接入信息对数据源

JBDC模板中的待客户化字段进行客户化,可以包括:将数据源JBDC模板中的待客户化字段替换为数据库接入信息。例如,将数据源JBDC模板中的待客户化字段替换为数据库ip地址信息(XX.XX.XX.XX)、数据库端口信息(端口1)、数据库实例名信息(XXX)即可。

[0046] 在一个实施例中,为了使开发人员即使无前后端开发的知识基础,也可快速开发基于SQL文本的数据库DML工具,数据库DML工具的低代码生成方法还可以包括:根据操作指令生成SQL文本组件;根据数据库DML工具的数据源JBDC、数据库DML工具的前端页面信息及SQL文本解析逻辑,生成数据库DML工具,可以包括:利用SQL文本解析逻辑获取用户通过SQL文本组件输入的SQL文本,对SQL文本的逻辑进行解析,对SQL文本按照字符串类型进行拆分,根据组件的属性值与属性类字符串的对应关系,将拆分后的SQL文本中的属性类字符串替换为组件的属性值,对替换处理后的SQL文本按照解析出的SQL文本的逻辑进行拼接;所述字符串类型包括语法类字符串、参数类字符串、属性类字符串。例如,假设开发人员想开发一个具有插入功能的MySQL数据库工具,可以在前端页面拖拽生成SQL文本组件,根据已经配置好的SQL文本解析逻辑,可以接收用户通过SQL文本组件输入的SQL文本,插入语句的通用格式为“INSERT INTO表名(字段名1,字段名2,……)VALUES(变量名1,变量名2,……)”,模板实例为“INSERT INTO fn_test(status,creator,……)VALUES(num1,num2,……)”。对SQL文本的逻辑进行解析,其中num1、num2即为特殊字符串,对应属性类字符串,则status、creator即为客户化字段,后端逻辑中标记num1、num2需要替换为前端组件上送的具体属性值。利用SQL文本解析逻辑对SQL文本按照字符串类型进行拆分,对拆分后的SQL文本中的属性类字符串进行标记。利用SQL文本解析逻辑根据组件的属性值与属性类字符串的对应关系,对拆分后的SQL文本中的属性类字符串进行替换;利用SQL文本解析逻辑对替换处理后的SQL文本按照解析出的SQL文本的逻辑进行拼接。下一步进行界面可视化设计的时候,会定义各组件属性值,假设status(状态)与creator(创造者)均为文本,则设计两个文本框,其中文本框的字段名属性分别定义为num1、num2,这样就能保证工具用户在输入该两个字段后前端组件可将值对应传送到后端,并替换掉SQL语句中的num1、num2,完成SQL语句的客户化。

[0047] 在本实施例中,为了使SQL文本拆分及替换更便捷,对SQL文本按照字符串类型进行拆分,根据组件的属性值与属性类字符串的对应关系,将拆分后的SQL文本中的属性类字符串替换为组件的属性值,可以包括:利用SQL文本解析逻辑及正则表达式,对SQL文本按照字符串类型进行拆分;利用SQL文本解析逻辑对拆分后的SQL文本中的属性类字符串进行标记,根据标记对SQL文本中的属性类字符串进行查找,根据组件的属性值与属性类字符串的对应关系,将查找到的属性类字符串替换为组件的属性值。例如,利用SQL文本解析逻辑及正则表达式,对“INSERT INTO fn_test(status,creator,……)VALUES(num1,num2,……)”按照语法类字符串、参数类字符串、属性类字符串进行拆分;利用SQL文本解析逻辑对拆分后的SQL文本中的属性类字符串进行标记,即num1、num2为特殊字符串,对应属性类字符串,将num1、num2进行标记,根据标记对SQL文本中的属性类字符串进行查找,根据组件的属性值与属性类字符串的对应关系,将查找到的属性类字符串替换为组件的属性值。

[0048] 在一个实施例中,根据操作指令生成组件,可以包括:根据操作指令确定组件的位置、名称及属性值;根据组件的位置、名称及属性值生成组件。例如,开发人员输入拖拽操作指令确定组件的位置,输入名称及属性值输入的操作指令确定组件的名称及属性值;输入

确定操作指令,依托平台根据组件的位置、名称及属性值生成组件。这样,开发人员可以通过简单的操作指令就可可视化拖拽组件,定义组件信息,完成数据库DML工具的开发。

[0049] 在一个实施例中,数据库DML工具的低代码生成方法生成的数据库DML工具支持数据库表的增加,修改,删除,(字典)查询功能。字典查询是指:技术可以配置查询页面时指定一些字段的枚举类型。例如:sex 1-男,0-女。配置后查询结果可以自动翻译。

[0050] 为了便于对本发明实施例的理解,数据库DML工具的低代码生成方法的开发实例如下:

[0051] 首先系统使用者配置数据源信息如下表1所示,在进行开发时通过输入应用名及数据库类型即可定位对应ip、端口及实例名信息:

[0052] 表1

[0053]

应用名	数据库类型	数据库ip地址	数据库端口	数据库实例名
XXX1	MySQL	XX.XX.XX.XX	3306	XXX
XXX2	DB2	……	……	……
XXX3	Oracle	……	……	……
XXX4	GaussDB	……	……	……

[0054] (1)假设系统使用者需要开发一个具有插入功能的MySQL数据库工具。

[0055] 插入语句的通用格式为“INSERT INTO表名(字段名1,字段名2,……)VALUES(变量名1,变量名2,……)”,在配置SQL模板时需保证每个客户化字段都对应一个需要输入的变量。因此模板实例为“INSERT INTO fn_test(status,creator,……)VALUES(num1,num2,……)”,其中num1、num2即为特殊字符串,对应属性类字符串,则status、creator即为客户化字段,后端逻辑中标记num1、num2需要替换为前端组件上送的具体属性值。下一步进行界面可视化设计的时候,会定义各组件属性值,假设status与creator均为文本,则设计两个文本框,其中文本框的字段名属性分别定义为num1、num2,这样就能保证工具用户在输入该两个字段后前端组件可将值对应传送到后端,并替换掉SQL语句中的num1、num2,完成SQL语句的客户化。最后将前后端逻辑组装,并生成工具界面,即完成数据库INSERT工具的开发。

[0056] (2)假设系统使用者需要开发一个具有变更功能的MySQL数据库工具。

[0057] 变更语句的通用格式为“UPDATE表名SET字段1=变量1,字段2=变量2where字段3=变量3”,则实例为“UPDATE fn_test SET status=num1,creator=num2 where id=num3”。其中num1、num2、num3即为特殊字符串,对应属性类字符串,则status、creator、id即为客户化字段,后端逻辑中会标记num1、num2、num3需要替换为前端组件上送的具体属性值。则UI界面需要对应拖拽三个组件,并将组件属性定义为num1、num2、num3,分别向后端传送num1、num2、num3的对应值。

[0058] 在本实施例中,删除及查询语句以此类推。

[0059] 综上所述,本发明实施例与现有技术中采用传统研发模式进行DML工具开发的技术方案相比,通过获取数据库类型信息、应用名称信息、结构化查询语言SQL模板、SQL文本解析逻辑及操作指令;所述SQL模板包括参数类字符串及以特定字符串表示的属性类字符串;所述SQL文本解析逻辑用于获取用户根据SQL模板输入的SQL文本,对SQL文本进行拆分,根据组件的属性值与属性类字符串的对应关系,将拆分后的SQL文本中的属性类字符串替

换为组件的属性值,对替换后的SQL文本进行拼接;所述操作指令用于对组件进行位置、名称、属性值定义;根据数据库类型信息确定数据源Java数据库连接JBDC模板;数据源JBDC模板包括待客户化字段;根据数据库类型信息、应用名称信息及预先配置的数据源信息表,对数据源JBDC模板中的待客户化字段进行客户化,得到数据库DML工具的数据源JBDC;所述数据源信息表包括数据库类型信息、应用名称信息与数据库接入信息的对应关系;根据操作指令生成组件;所述组件的组件名称为参数类字符串;所述组件的属性值与属性类字符串一一对应;根据操作指令及组件信息,得到数据库DML工具的前端页面信息;所述组件信息包括组件名称、组件属性值、组件的位置信息;根据数据库DML工具的数据源JBDC、数据库DML工具的前端页面信息及SQL文本解析逻辑,生成数据库DML工具,可以节约数据库DML工具的开发人力成本,避免由于开发人员认知偏差而产生错误,提高开发效率。

[0060] 本发明实施例中还提供了一种数据库DML工具的低代码生成装置,如下面的实施例所述。由于该装置解决问题的原理与数据库DML工具的低代码生成方法相似,因此该装置的实施可以参见数据库DML工具的低代码生成方法的实施,重复之处不再赘述。

[0061] 图5为本发明实施例中数据库DML工具的低代码生成装置的示意图,包括:

[0062] 获取模块501,用于获取数据库类型信息、应用名称信息、结构化查询语言SQL模板、SQL文本解析逻辑及操作指令;所述SQL模板包括参数类字符串及以特定字符串表示的属性类字符串;所述SQL文本解析逻辑用于获取用户根据SQL模板输入的SQL文本,对SQL文本进行拆分,根据组件的属性值与属性类字符串的对应关系,将拆分后的SQL文本中的属性类字符串替换为组件的属性值,对替换后的SQL文本进行拼接;所述操作指令用于对组件进行位置、名称、属性值定义;

[0063] 客户化模块502,用于根据数据库类型信息、应用名称信息及预先配置的数据源信息表,确定数据库数据操作语言DML工具的数据源JBDC;所述数据源信息表包括数据库类型信息、应用名称信息与数据库接入信息的对应关系;

[0064] 组件生成模块503,用于根据操作指令生成组件;所述组件的组件名称为参数类字符串;所述组件的属性值与属性类字符串一一对应;

[0065] 前端页面信息生成模块504,用于根据操作指令及组件信息,得到数据库DML工具的前端页面信息;所述组件信息包括组件名称、组件属性值、组件的位置信息;

[0066] 数据库DML工具生成模块505,用于根据数据库DML工具的数据源JBDC、数据库DML工具的前端页面信息及SQL文本解析逻辑,生成数据库DML工具。

[0067] 在一个实施例中,客户化模块502,具体用于根据数据库类型信息、应用名称信息及预先配置的数据源信息表,确定数据库DML工具的数据源JBDC,包括:

[0068] 根据数据库类型信息确定数据源Java数据库连接JBDC模板;数据源JBDC模板包括待客户化字段;

[0069] 根据数据库类型信息、应用名称信息及预先配置的数据源信息表,对数据源JBDC模板中的待客户化字段进行客户化,得到数据库DML工具的数据源JBDC。

[0070] 在本实施例中,客户化模块502,具体用于:

[0071] 根据数据库类型信息、应用名称信息及预先配置的数据源信息表,确定数据库接入信息;所述数据库接入信息包括数据库ip地址信息、数据库端口信息、数据库实例名信息的其中一种或任意组合;

[0072] 根据数据库接入信息对数据源JBDC模板中的待客户化字段进行客户化,得到数据库DML工具的数据源JBDC。

[0073] 在本实施例中,客户化模块502,具体用于:

[0074] 将数据源JBDC模板中的待客户化字段替换为数据库接入信息。

[0075] 在一个实施例中,数据库DML工具的低代码生成装置还包括:SQL文本组件生成模块,用于:根据操作指令生成SQL文本组件;

[0076] 数据库DML工具生成模块505,具体用于:

[0077] 利用SQL文本解析逻辑获取用户通过SQL文本组件输入的SQL文本,对SQL文本的逻辑进行解析,对SQL文本按照字符串类型进行拆分,根据组件的属性值与属性类字符串的对应关系,将拆分后的SQL文本中的属性类字符串替换为组件的属性值,对替换处理后的SQL文本按照解析出的SQL文本的逻辑进行拼接;所述字符串类型包括语法类字符串、参数类字符串、属性类字符串。

[0078] 在本实施例中,数据库DML工具生成模块505,具体用于:

[0079] 利用SQL文本解析逻辑及正则表达式,对SQL文本按照字符串类型进行拆分;

[0080] 利用SQL文本解析逻辑对拆分后的SQL文本中的属性类字符串进行标记,根据标记对SQL文本中的属性类字符串进行查找,根据组件的属性值与属性类字符串的对应关系,将查找到的属性类字符串替换为组件的属性值。

[0081] 在一个实施例中,组件生成模块503,具体用于:

[0082] 根据操作指令确定组件的位置、名称及属性值;

[0083] 根据组件的位置、名称及属性值生成组件。

[0084] 本发明实施例提供一种基于SQL拆分的数据库DML工具的低代码开发方法、装置及平台,通过系统的数据源自动封装模块、SQL语句解析模块、工具界面可视化设计模块、前后端集成模块四个模块的协同作用按照数据库DML工具的低代码开发方法的步骤,实现数据库DML工具的低代码开发。能让系统使用者(开发人员)通过简单的配置数据源、配置SQL模板、可视化界面设计三个步骤即可完成数据库DML工具的开发,避免传统研发模式下的人力成本损耗。即使开发人员没有开发基础,也能通过本发明提供的方法、系统、装置完成开发任务,简单、快捷、高效。开发的工具还能供用户使用,实现数据库的增、删、改、查,以提高金融行业开发、测试效率。

[0085] 本发明实施例还提供一种计算机设备,图6为本发明实施例中计算机设备的示意图,所述计算机设备600包括存储器610、处理器620及存储在存储器610上并可在处理器620上运行的计算机程序630,所述处理器620执行所述计算机程序630时实现上述数据库DML工具的低代码生成方法。

[0086] 本发明实施例还提供一种计算机可读存储介质,所述计算机可读存储介质存储有计算机程序,所述计算机程序被处理器执行时实现上述数据库DML工具的低代码生成方法。

[0087] 本发明实施例还提供一种计算机程序产品,所述计算机程序产品包括计算机程序,所述计算机程序被处理器执行时实现上述数据库DML工具的低代码生成方法。

[0088] 本领域内的技术人员应明白,本发明的实施例可提供为方法、系统、或计算机程序产品。因此,本发明可采用完全硬件实施例、完全软件实施例、或结合软件和硬件方面的实施例的形式。而且,本发明可采用在一个或多个其中包含有计算机可用程序代码的计算机

可用存储介质(包括但不限于磁盘存储器、CD-ROM、光学存储器等)上实施的计算机程序产品的形式。

[0089] 本发明是参照根据本发明实施例的方法、设备(系统)、和计算机程序产品的流程图和/或方框图来描述的。应理解可由计算机程序指令实现流程图和/或方框图中的每一流程和/或方框、以及流程图和/或方框图中的流程和/或方框的结合。可提供这些计算机程序指令到通用计算机、专用计算机、嵌入式处理机或其他可编程数据处理设备的处理器以产生一个机器,使得通过计算机或其他可编程数据处理设备的处理器执行的指令产生用于实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能的装置。

[0090] 这些计算机程序指令也可存储在能引导计算机或其他可编程数据处理设备以特定方式工作的计算机可读存储器中,使得存储在该计算机可读存储器中的指令产生包括指令装置的制造品,该指令装置实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能。

[0091] 这些计算机程序指令也可装载到计算机或其他可编程数据处理设备上,使得在计算机或其他可编程设备上执行一系列操作步骤以产生计算机实现的处理,从而在计算机或其他可编程设备上执行的指令提供用于实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能的步骤。

[0092] 以上所述的具体实施例,对本发明的目的、技术方案和有益效果进行了进一步详细说明,所应理解的是,以上所述仅为本发明的具体实施例而已,并不用于限定本发明的保护范围,凡在本发明的精神和原则之内,所做的任何修改、等同替换、改进等,均应包含在本发明的保护范围之内。

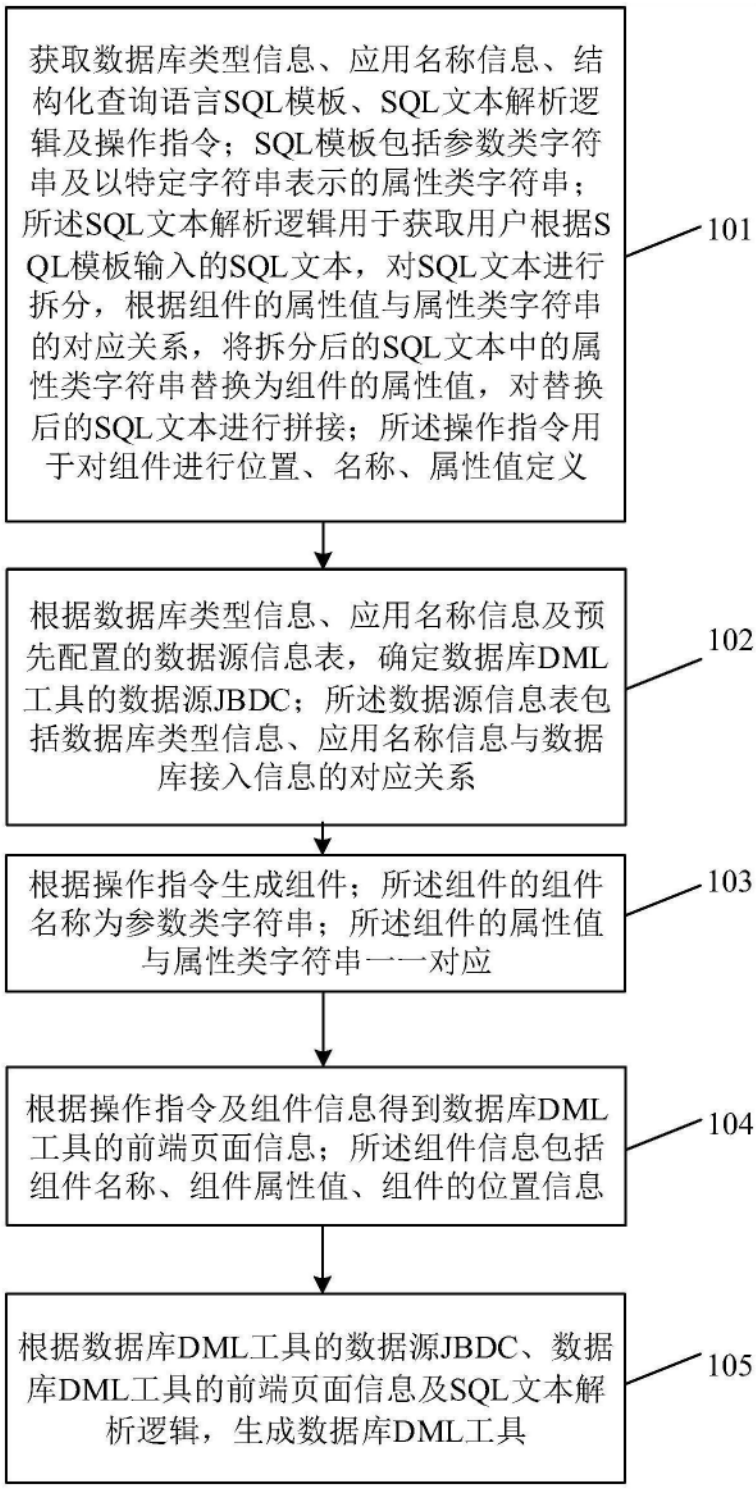

获取数据库类型信息、应用名称信息、结构化查询语言SQL模板、SQL文本解析逻辑及操作指令；SQL模板包括参数类字符串及以特定字符串表示的属性类字符串；所述SQL文本解析逻辑用于获取用户根据SQL模板输入的SQL文本，对SQL文本进行拆分，根据组件的属性值与属性类字符串的对应关系，将拆分后的SQL文本中的属性类字符串替换为组件的属性值，对替换后的SQL文本进行拼接；所述操作指令用于对组件进行位置、名称、属性值定义
101
根据数据库类型信息、应用名称信息及预先配置的数据源信息表，确定数据库DML工具的数据源JBDC；所述数据源信息表包括数据库类型信息、应用名称信息与数据库接入信息的对应关系
102
根据操作指令生成组件；所述组件的组件名称为参数类字符串；所述组件的属性值与属性类字符串一一对应
103
根据操作指令及组件信息得到数据库DML工具的前端页面信息；所述组件信息包括组件名称、组件属性值、组件的位置信息
104
根据数据库DML工具的数据源JBDC、数据库DML工具的前端页面信息及SQL文本解析逻辑，生成数据库DML工具
105

图1

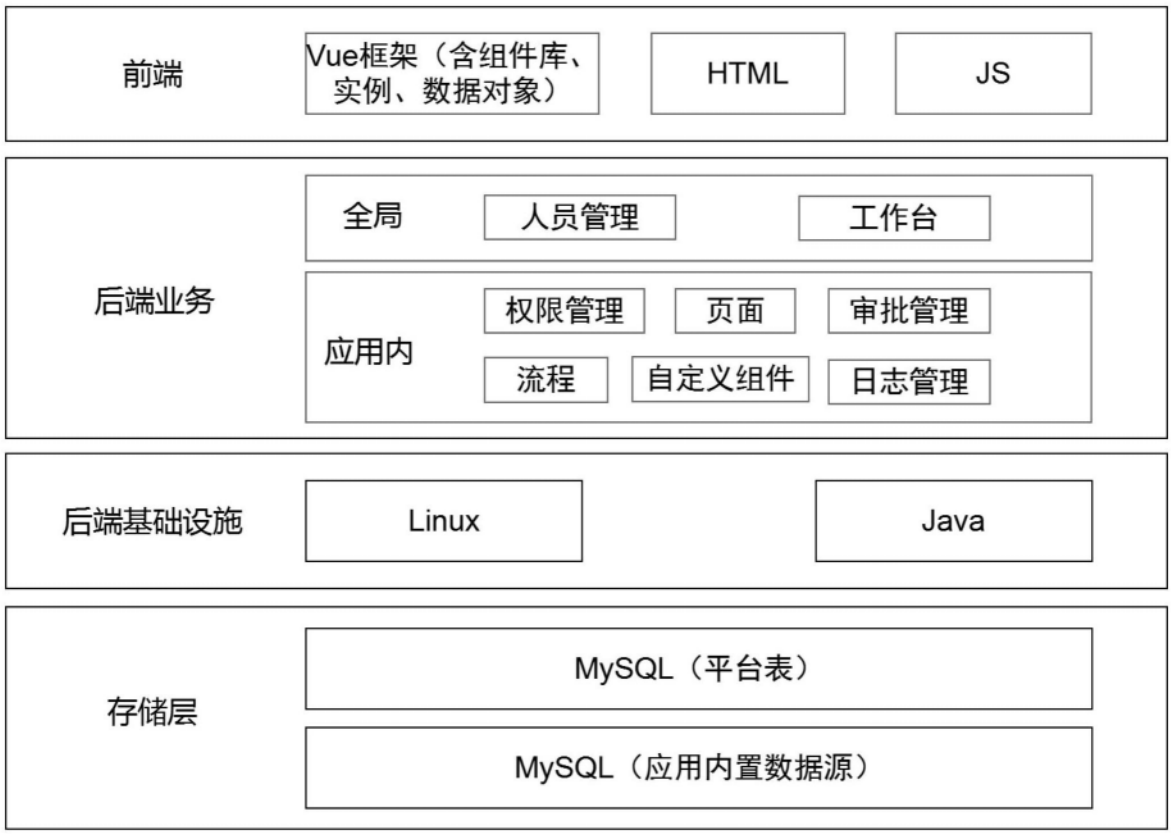

前端
Vue框架（含组件库、实例、数据对象）
HTML
JS
后端业务
全局
人员管理
工作台
应用内
权限管理
页面
审批管理
流程
自定义组件
日志管理
后端基础设施
Linux
Java
存储层
MySQL（平台表）
MySQL（应用内置数据源）

图2

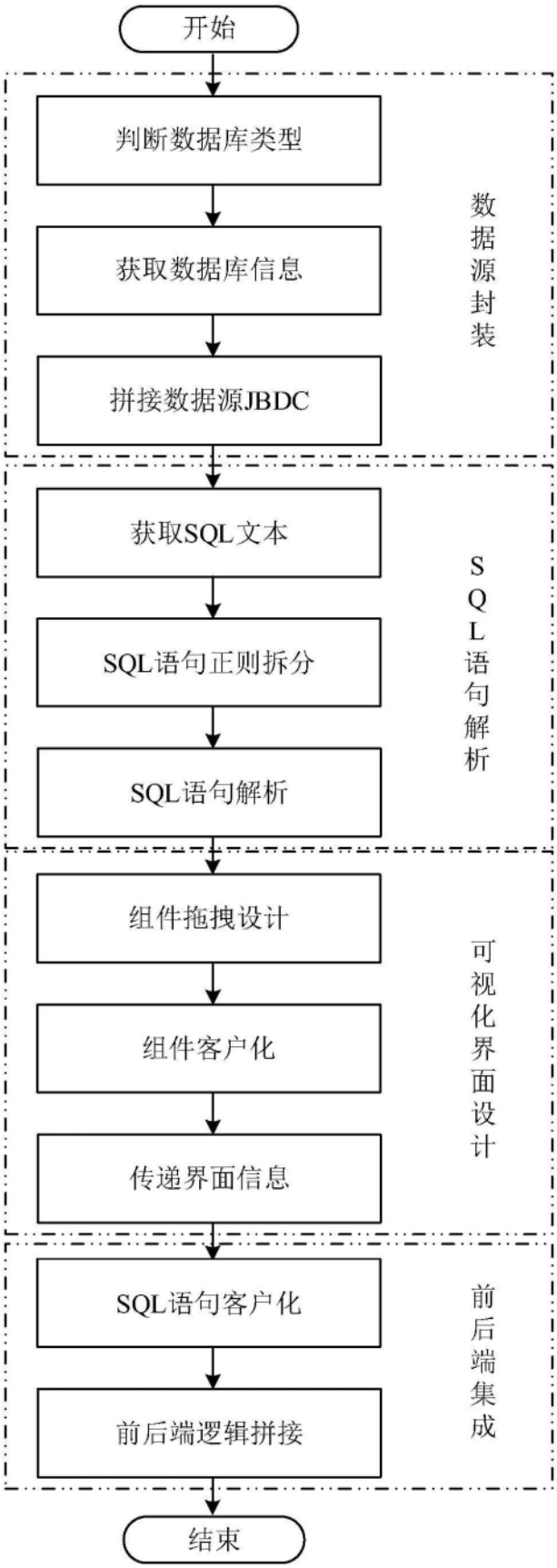

开始
判断数据库类型
获取数据库信息
拼接数据源JBDC
数据源封装
获取SQL文本
SQL语句正则拆分
SQL语句解析
SQL语句解析
组件拖拽设计
组件客户化
传递界面信息
可视化界面设计
SQL语句客户化
前后端逻辑拼接
前后端集成
结束

图3

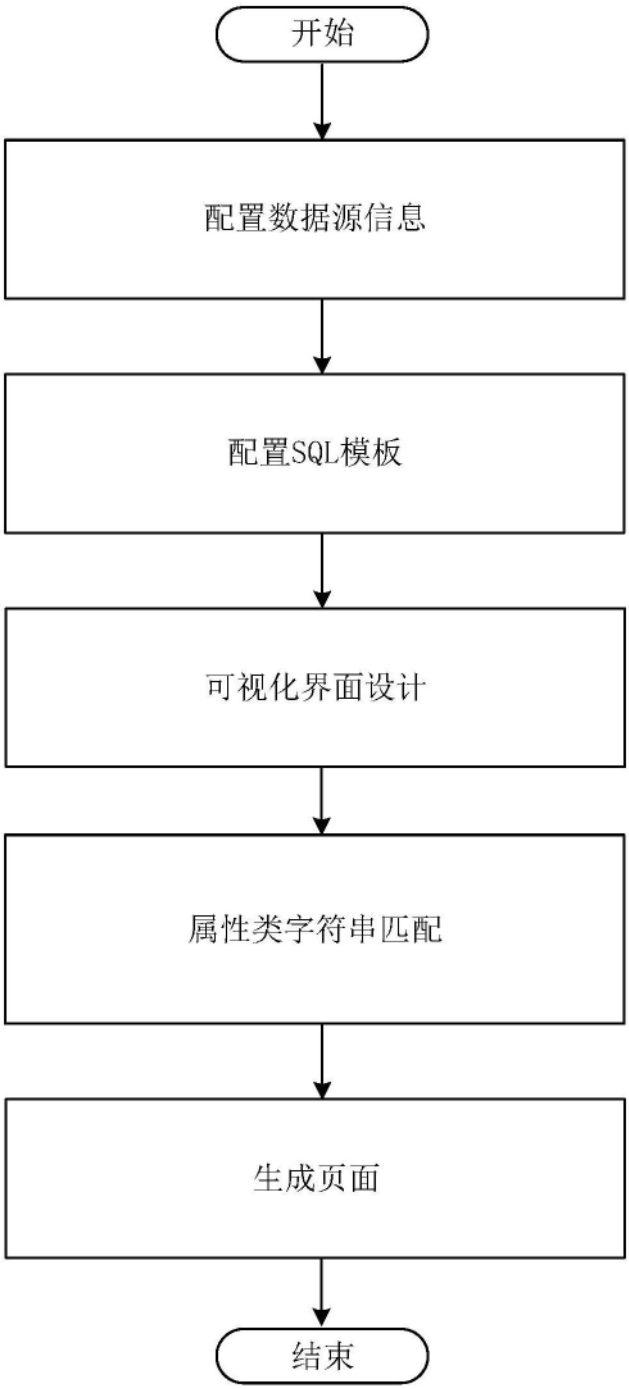


图4

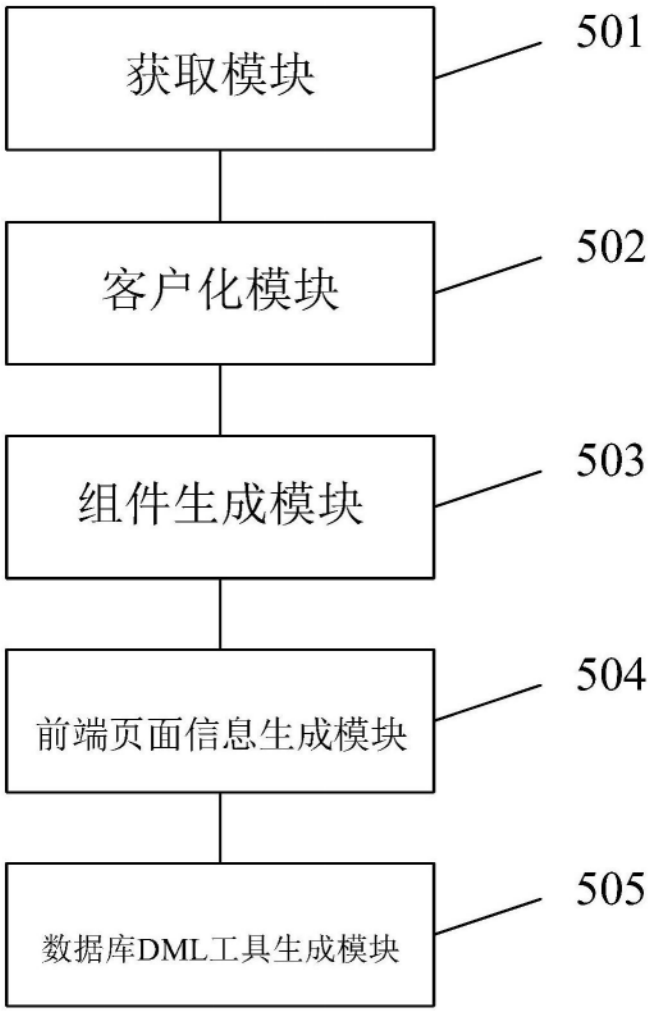

获取模块
501
客户化模块
502
组件生成模块
503
前端页面信息生成模块
504
数据库DML工具生成模块
505

图5

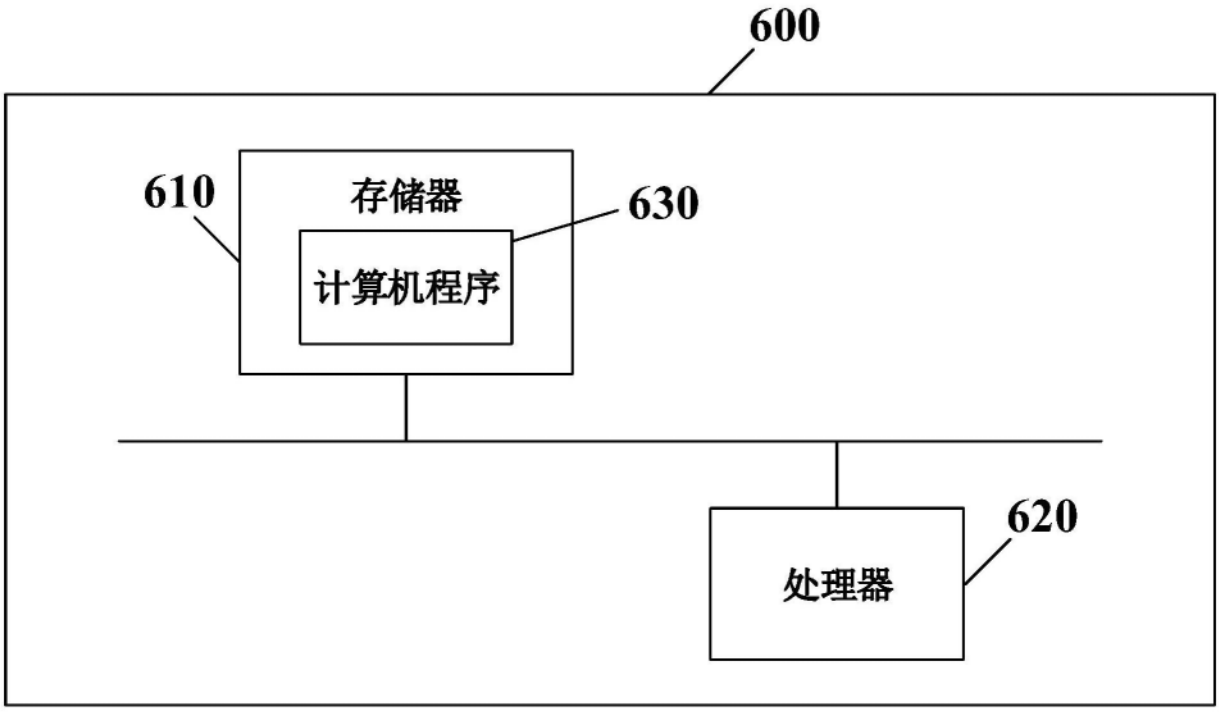


图6