(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 110297843 A
(43)申请公布日 2019.10.01

(21)申请号 201910591953.2

(22)申请日 2019.07.02

(71)申请人 山东浪潮通软信息科技有限公司
地址 250100 山东省济南市高新区孙村镇科航路2877号

(72)发明人 王岩恺 宋伟伟 张冬霞

(74)专利代理机构 济南信达专利事务所有限公司 37100
代理人 孙园园

(51)Int.Cl.
G06F 16/242(2019.01)
G06F 16/2455(2019.01)

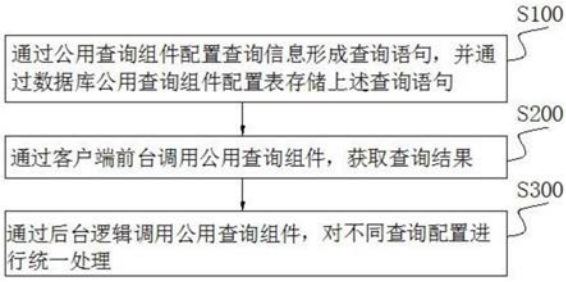
权利要求书2页 说明书18页 附图1页

(54)发明名称

用于B/S系统的数据查询方法及系统、终端

(57)摘要

本发明公开了一种用于B/S系统的数据查询方法及系统、终端，属于B/S系统查询领域，要解决的技术问题为如何实现查询操作的快速开发、部署，并在后期维护时实现查询配置的灵活修改、即时生效；方法包括：通过公用查询组件配置查询信息形成查询语句，并通过数据库公用查询组件配置表存储上述查询语句；通过客户端前台调用公用查询组件，获取查询结果；通过后台逻辑调用公用查询组件，对不同查询配置进行统一处理。系统，包括配置模块、查询模块和逻辑处理模块。终端包括处理器、输入设备、输出设备和存储器，处理器被配置用于调用所述程序指令执行用于B/S系统的数据查询方法。



CN 110297843 A

1.用于B/S系统的数据查询方法,其特征在于通过公用查询组件实现公用的查询逻辑,包括:

通过公用查询组件配置查询信息形成查询语句,并通过数据库公用查询组件配置表存储上述查询语句;

通过客户端前台调用公用查询组件,获取查询结果;

通过后台逻辑调用公用查询组件,对不同查询配置进行统一处理;

查询语句为查询SQL语句,包括但不限于查询配置编号、查询字段、表名、查询条件、排序条件、和分组条件,查询条件包括静态查询条件、动态查询条件、动态In查询条件、动态Like查询条件,上述查询语句至少一个,每个查询语句对应一个查询配置。

2.根据权利要求1所述的用于B/S系统的数据查询方法,其特征在于通过公用查询组件配置查询信息形成查询语句,并通过数据库公用查询组件配置表存储上述查询语句,包括:

基于客户端前台的动态传参方式,在不同查询配置之间进行切换;

基于客户端前台的动态传参方式,动态加载查询配置并传递动态查询条件的动态参数;

基于客户端前台的动态传参方式,对查询语句进行变更维护,包括修改查询字段、表名、查询条件、排序条件和分组条件。

3.根据权利要求1或2所述的用于B/S系统的数据查询方法,其特征在于通过公用查询组件配置查询信息形成查询语句时,查询信息中的变量用@@包裹。

4.根据权利要求1或2所述的用于B/S系统的数据查询方法,其特征在于通过公用查询组件配置查询信息形成查询语句,还包括通过静态查询条件配置与动态查询条件加载相结合的方式对查询信息中数据进行过滤,具体包括:

将不会变化、不需动态传值的查询条件配置于数据库公用查询组件配置表的静态查询条件中;

将需要动态传值的查询条件配置于数据库公用查询组件配置表的动态查询条件中;

通过客户端前台页面传参的方式对动态查询条件的参数进行参数值传递,以实现查询语句中数据的动态条件过滤。

5.根据权利要求1或2所述的用于B/S系统的数据查询方法,其特征在于通过客户端前台调用公用查询组件,获取查询结果,包括如下步骤:

通过客户端前台提供JavaScript API封装方法,客户端前台通过引入的JS文件调用公用查询组件;

客户端前台调用executePubEasySqlTable方法,并通过传递查询配置编号或动态查询条件获取查询结果。

6.根据权利要求5所述的用于B/S系统的数据查询方法,其特征在于通过传递动态查询条件获取查询结果时,动态查询条件的参数名为相应查询配置中用@@包裹的变量;

动态查询条件的参数有多个值时况,所述多个值用逗号分隔,动态查询条件有多个参数时,所述多个参数用分号分隔。

7.根据权利要求1或2所述的用于B/S系统的数据查询方法,其特征在于通过后台逻辑调用公用查询组件,对不同查询配置进行统一处理,包括:

获取查询配置;

识别查询信息的参数；

对查询信息的参数进行包装；

绑定查询信息的参数中的变量；

基于查询语句执行查询，返回查询结果。

8.用于B/S系统的数据查询系统，其特征在于包括：

配置模块，所述配置模块用于通过公用查询组件配置查询信息形成查询语句，并用于调用数据库公用查询组件配置表存储上述查询语句；

查询模块，所述查询模块用于通过客户端前台调用公用查询组件，获取查询结果；

逻辑处理模块，所述逻辑处理模块用于通过后台逻辑调用公用查询组件，对不同查询配置进行统一处理；

通过配置模块配置的查询语句为查询SQL语句，包括但不限于查询配置编号、查询字段、表名、查询条件、排序条件、和分组条件，查询条件包括静态查询条件、动态查询条件、动态In查询条件、动态Like查询条件，上述查询语句至少一个，每个查询语句对应一个查询配置。

9.一种终端，其特征在于包括处理器、输入设备、输出设备和存储器，处理器、输入设备、输出设备和存储器相互连接，存储器用于存储计算机程序，所述计算机程序包括程序指令，所述处理器被配置用于调用所述程序指令执行如权利要求1-7任一项所述的用于B/S系统的数据查询方法。

用于B/S系统的数据查询方法及系统、终端

技术领域

[0001] 本发明涉及B/S系统查询领域，具体地说是一种用于B/S系统的数据查询方法及系统、终端。

背景技术

[0002] 目前，传统的web应用开发伴有大量的前后台数据交互操作，其中，以查询为主的请求数目占领了较高比重，这些查询操作不仅带来了大量重复的编码工作，甚至会造成代码的安全性问题，同时对后期的维护工作造成了一定的难度，浪费了大量的时间，而且对项目成本造成了很大的负担。

[0003] 基于上述分析，如何实现查询操作的快速开发、部署，并在后期维护时实现查询配置的灵活修改、即时生效，是需要解决的技术问题。

发明内容

[0004] 本发明的技术任务是针对以上不足，提供一种用于B/S系统的数据查询方法及系统、终端，来解决如何实现查询操作的快速开发、部署，并在后期维护时实现查询配置的灵活修改、即时生效的问题。

[0005] 第一方面，本发明提供一种用于B/S系统的数据查询方法，通过公用查询组件实现公用的查询逻辑，包括：

[0006] 通过公用查询组件配置查询信息形成查询语句，并通过数据库公用查询组件配置表存储上述查询语句；

[0007] 通过客户端前台调用公用查询组件，获取查询结果；

[0008] 通过后台逻辑调用公用查询组件，对不同查询配置进行统一处理；

[0009] 查询语句为查询SQL语句，包括但不限于查询配置编号、查询字段、表名、查询条件、排序条件、和分组条件，查询条件包括静态查询条件、动态查询条件、动态In查询条件、动态Like查询条件，上述查询语句至少一个，每个查询语句对应一个查询配置。

[0010] 上述实施方式中，通过公用查询组件将查询语句配置在数据库公用查询组件配置表中，实现了规则的灵活修改和维护，获取查询信息支持静态、动态等多种参数配置，对数据查询操作的安全性提供了有力保障。

[0011] 作为优选，通过公用查询组件配置查询信息形成查询语句，并通过数据库公用查询组件配置表存储上述查询语句，包括：

[0012] 基于客户端前台的动态传参方式，在不同查询配置之间进行切换；

[0013] 基于客户端前台的动态传参方式，动态加载查询配置并传递动态查询条件的动态参数；

[0014] 基于客户端前台的动态传参方式，对查询语句进行变更维护，包括修改查询字段、表名、查询条件、排序条件和分组条件。

[0015] 作为优选，通过公用查询组件配置查询信息形成查询语句时，查询信息中的变量

用@@包裹。

[0016] 更优的，通过公用查询组件配置查询信息形成查询语句，还包括通过静态查询条件配置与动态查询条件加载相结合的方式对查询信息中数据进行过滤，具体包括：

[0017] 将不会变化、不需动态传值的查询条件配置于数据库公用查询组件配置表的静态查询条件中；

[0018] 将需要动态传值的查询条件配置于数据库公用查询组件配置表的动态查询条件中；

[0019] 通过客户端前台页面传参的方式对动态查询条件的参数进行参数值传递，以实现查询语句中数据的动态条件过滤。

[0020] 作为优选，通过客户端前台调用公用查询组件，获取查询结果，包括如下步骤：

[0021] 通过客户端前台提供JavaScript API封装方法，客户端前台通过引入的JS文件调用公用查询组件；

[0022] 客户端前台调用executePubEasySqlTable方法，并通过传递查询配置编号或动态查询条件获取查询结果。

[0023] 作为优选，通过传递动态查询条件获取查询结果时，动态查询条件的参数名为相应查询配置中用@@包裹的变量，；

[0024] 动态查询条件的参数有多个值时况，所述多个值用逗号分隔，动态查询条件有多个参数时，所述多个参数用分号分隔。

[0025] 作为优选，通过后台逻辑调用公用查询组件，对不同查询配置进行统一处理，包括：

[0026] 获取查询配置；

[0027] 识别查询信息的参数；

[0028] 对查询信息的参数进行包装；

[0029] 绑定查询信息的参数中的变量；

[0030] 基于查询语句执行查询，返回查询结果。

[0031] 第二方面，本发明提供一种用于B/S系统的数据查询系统，包括：

[0032] 配置模块，所述配置模块用于通过公用查询组件配置查询信息形成查询语句，并用于调用数据库公用查询组件配置表存储上述查询语句；

[0033] 查询模块，所述查询模块用于通过客户端前台调用公用查询组件，获取查询结果；

[0034] 逻辑处理模块，所述逻辑处理模块用于通过后台逻辑调用公用查询组件，对不同查询配置进行统一处理；

[0035] 通过配置模块配置的查询语句为查询SQL语句，包括但不限于查询配置编号、查询字段、表名、查询条件、排序条件、和分组条件，查询条件包括静态查询条件、动态查询条件、动态In查询条件、动态Like查询条件，上述查询语句至少一个，每个查询语句对应一个查询配置。

[0036] 第三方面，本发明提供一种终端，包括处理器、输入设备、输出设备和存储器，处理器、输入设备、输出设备和存储器相互连接，存储器用于存储计算机程序，所述计算机程序包括程序指令，所述处理器被配置用于调用所述程序指令执行如第一方面任一项所述的用于B/S系统的数据查询方法。

[0037] 本发明的用于B/S系统的数据查询方法及系统、终端具有以下优点：

[0038] 1、通过公用查询组件将查询语句配置在数据库公用查询组件配置表中，实现了规则的灵活修改和维护，获取查询信息支持静态、动态等多种参数配置，对数据查询操作的安全性提供了有力保障；

[0039] 2、可通过数据库配置所要查询数据的分组、排序条件，可通过数据库配置静态查询条件、动态查询条件以及复杂查询条件，并可通过数据库配置查询规则支持不停机生效，便于功能的维护；

[0040] 3、支持静态查询条件配置，不需要动态获取值的查询条件，支持动态查询条件配置，支持同一动态条件可匹配多个值的配置，其中值的个数可根据需求进行调整；

[0041] 4、在支持动态条件参数数值动态获取的情况下，解决了注入SQL注入等安全问题，保证参数值在仅为字符串的条件下进行比对，不会对SQL进行影响，更不会对原始SQL进行拆分改变，保证了无关数据不会泄露，使用户数据得到了极大的安全保证。

附图说明

[0042] 为了更清楚地说明本发明实施例中的技术方案，下面将对实施例中描述中所需要使用的附图作简要介绍，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，对于本领域的普通技术人员来讲，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。

[0043] 下面结合附图对本发明进一步说明。

[0044] 附图1为实施例1用于B/S系统的数据查询方法的流程框图。

具体实施方式

[0045] 下面结合附图和具体实施例对本发明作进一步说明，以使本领域的技术人员可以更好地理解本发明并能予以实施，但所举实施例不作为对本发明的限定，在不冲突的情况下，本发明实施例以及实施例中的技术特征可以相互结合。

[0046] 本发明实施例提供用于B/S系统的数据查询方法及系统、终端，用于解决如何实现查询操作的快速开发、部署，并在后期维护时实现查询配置的灵活修改、即时生效的技术问题。

[0047] 实施例1：

[0048] 如附图1所示，本发明的一种用于B/S系统的数据查询方法，通过公用查询组件实现公用的查询逻辑，包括：

[0049] S100、通过公用查询组件配置查询信息形成查询语句，并通过数据库公用查询组件配置表存储上述查询语句；

[0050] S200、通过客户端前台调用公用查询组件，获取查询结果；

[0051] S300、通过后台逻辑调用公用查询组件，对不同查询配置进行统一处理。

[0052] 其中，查询语句为查询SQL语句，包括但不限于查询配置编号、查询字段、表名、查询条件、排序条件、和分组条件，查询条件包括静态查询条件、动态查询条件、动态In查询条件、动态Like查询条件，上述查询语句至少一个，每个查询语句对应一个查询配置。

[0053] 其中，通过公用查询组件配置查询信息形成查询语句，并通过数据库公用查询组

件配置表存储上述查询语句,包括如下模式:

[0054] 模式一、基于客户端前台的动态传参方式,在不同查询配置之间进行切换;

[0055] 模式二、基于客户端前台的动态传参方式,动态加载查询配置并传递动态查询条件的动态参数;

[0056] 模式三、基于客户端前台的动态传参方式,对查询语句进行变更维护,包括修改查询字段、表名、查询条件、排序条件和分组条件,支持公用查询组件技术的灵活性。

[0057] 步骤S100中公用查询组件将开发所使用的SQL统一配置在数据库的公用查询组件配置表中,用于查询的统一维护及集中管理。公用查询组件配置表结构如下表所示:

[0058]

字段	类型	说明
EASY_SQL_ID	VARCHAR	主键
EASY_SQL_TYPE	CHAR	SQL 类型,3: 查询
EASY_SQL_COLNAME	VARCHAR	列名。查询:查询字段

[0059]

EASY_SQL_FROM	VARCHAR	FROM 表名
EASY_SQL_WHERE	VARCHAR	查询条件
EASY_SQL_WHERE_VAR	VARCHAR	动态查询条件
EASY_SQL_LIKEWHERE	VARCHAR	动态查询 LIKE 条件
EASY_SQL_INWHERE	VARCHAR	动态查询 IN 条件
EASY_SQL_ORDER	VARCHAR	排序条件
EASY_SQL_GROUP	VARCHAR	分组条件
EASY_SQL_HAVING	VARCHAR	分组过滤条件
EASY_SQL_BZ	VARCHAR	备注

[0060] 以一个业务的简单查询需求为例,开发原始查询SQL如下:

```
SELECT USER_ID,USER_NAME
FROM
  PUB_USERS
WHERE ACCOUNT_STATUS = '11'
  AND IS_SYS = ?
  AND SECURITY_LEVEL = ?
  AND CREATE_TIME LIKE CONCAT("%", ? , "%")
  AND USER_NAME IN (?,?,?,?)
  AND USER_ID IN (?,?,?,?)
ORDER BY USER_ID DESC,
  USER_NAME ASC
```

[0061]

[0062] 一个查询业务需要明确查询字段、表名,动态查询、静态查询、排序、分组条件等。以示例查询语句所示,查询字段为USER_ID,USER_NAME,表名为PUB_USERS,静态查询条件为ACCOUNT_STATUS＝'11',动态查询条件为AND IS_SYS＝?AND SECURITY_LEVEL＝?AND CREATE_TIME LIKE CONCAT("%",?,"%")AND USER_NAME IN(?,?,?,?)AND USER_ID IN (?,?,?,?),排序条件为ORDER BY USER_ID DESC,分组条件为ORDER BY USER_ID DESC, USER_NAME ASC。其中,静态查询条件不会随业务变动而变更,动态查询条件需要与用户交互动态传入条件值,动态查询还包括IN的多值匹配。

[0063] 公用查询组件需要将开发原始查询SQL语句配置到数据库中。MySql数据库配置示例如下,根据配置表的配置说明,对原始SQL语句的SQL类型、查询列名、查询表名、静态查询条件、动态查询条件、动态In查询条件、动态like查询条件、排序、分组、分组过滤条件进行配置,每个完整的配置代表一个完整的查询SQL语句。

[0064] 通过公用查询组件配置查询信息形成查询语句时,查询信息中的变量用@@包裹。即SECURITY_LEVEL＝?的?部分)用@@包裹,如:@SECURITYLEVEL@;多值匹配变量只写一个,如:USER_NAME IN(@USERNAME@)。代码如下:

[0065]

```
INSERT INTO `PUB_EASY_SQL` (
  `EASY_SQL_ID`,`EASY_SQL_TYPE`,`EASY_SQL_COLNAME`,
  `EASY_SQL_FROM`,`EASY_SQL_WHERE`,`EASY_SQL_WHERE_VAR`,
  `EASY_SQL_LIKEWHERE`,`EASY_SQL_INWHERE`,`EASY_SQL_ORDER`,
  `EASY_SQL_GROUP`,`EASY_SQL_HAVING`,`EASY_SQL_BZ`
)
VALUES
  (
'DEMO_QUERY','3','USER_ID,
USER_NAME','PUB_USERS','AND ACCOUNT_STATUS = \'11\'',
    'AND IS_SYS = @ISSYS@   AND SECURITY_LEVEL = @SECURITYLEVEL@
AND CREATE_TIME LIKE CONCAT(\"%\",@CREATETIME@,\"%\")',
    NULL,
```

[0066]

```
    'AND USER_NAME IN(@USERNAME@) AND USER_ID IN (@USERID@)   AND
USER_NAME IN(@USERNAME@)',
    'USER_ID DESC,USER_NAME ASC',
    '','','查询测试案例'
  ) .
```

[0067] 步骤S200中,通过客户端前台调用公用查询组件,获取查询结果,包括如下步骤:

[0068] S210、通过客户端前台提供JavaScript API封装方法,客户端前台通过引入的JS文件调用公用查询组件;

[0069] S220、客户端前台调用executePubEasySqlTable方法,并通过传递查询配置编号或动态查询条件等获取查询结果,动态查询条件的参数名为相应配置中用@@包裹的变量,例如配置中为@SECURITYLEVEL@,则前台传递参数名需使用SECURITYLEVEL＝XXXX,针对一个参数有多个值的情况,需将多个值用‘,’逗号分隔,多个参数间用‘;’分号分割。

[0070] 步骤S210的核心代码如下:

[0071]

```
forPubSql()
// 公用查询组件总入口
function forPubSql(){
    // 定义配置编号
  var pubEasySqlId = 'demo_query';
  // 定义拼接动态查询参数值
  var params = 'CREATETIME=2005;SECURITYLEVEL=9;ISSYS=1';
    // 定义动态 In 查询参数值
  var iparams = 'USERNAME=CPUBLIC, 高 级 管 理 员,SYSADMIN;USERID=CPUBLIC,GPUBLIC,SUPERADMIN,SYSADMIN';
```

[0072]

```
        // 对参数进行编码操作
        var pubEasySqlIdStr=encodePubEasySqlParam(pubEasySqlId);
      var paramsStr=encodePubEasySqlParam(params);
      var iparamsStr=encodePubEasySqlParam(iparams);
      // 拼接请求所需的URL
      var url="pubEasySql.cmd?pubSqlId="+pubEasySqlIdStr
                                  +"&params="+paramsStr
                                  +"&iparams="+iparamsStr;
        // 执行公用查询组件的查询方法
      var arr=executePubEasySqlTable(url);
      if(arr==-1){
         alert("没有数据");
        return;
      }
        // 存在数据，遍历取出
      for(var i=0;i<arr.length;i++){
         for(var j=0;j<arr[i].length;j++){
          alert("v["+i+"]["+j+"]="+arr[i][j]);
          console.log('>>>>>>>>>>>>>>>>>>>>>>RESULT          DETIAL:
'+"v["+i+"]["+j+"]="+arr[i][j]);
      }
      }
    }
    // 对入参进行编码并返回
    function encodePubEasySqlParam(sql){
```

```
    return encodeURIComponent(encodeURIComponent(sql));
}
// 获取单个请求值
function executePubEasySql(url) {
  if (url == null) {
     alert("传递的URL不能为空！");
     return;
  }
  ;
  try {
     var xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
     if (xmlhttp == null) {
[0073]          alert("创建XMLHTTP对象失败!" + ex.description);
return;
     }
     ;
     xmlhttp.open("POST", url, false);
     xmlhttp.send();
     return xmlhttp.responseText;
  } catch (ex) {
     alert("XMLHTTP对象传递数据失败!" + ex.description);
     return "";
  }
  ;
};
```

```
// 获取多个请求值
function executePubEasySqlTable(url) {
  var values = executePubEasySql(url);
  if (values.indexOf("没有记录") > -1) {
     return -1;
  }
  ;
  var arr = new Array();
  var obs = values.split("@|@");
  for ( var i = 0; i < obs.length; i++) {
     obsv = obs[i].split("@,@");
     arr[i] = obsv;
  }
  ;
  return arr;
}.
```

[0074]

[0075] 步骤S300中,通过后台逻辑调用公用查询组件,对不同查询配置进行统一处理,包括如下内容:

[0076] 获取查询配置;

[0077] 识别查询信息的参数;

[0078] 对查询信息的参数进行包装;

[0079] 绑定查询信息的参数中的变量;

[0080] 基于查询语句执行查询,返回查询结果。

[0081] 上述步骤S300的核心代码为:

[0082] /**

[0083]

```
        * 执行公用查询操作
        */
       protected String execute(HttpServletRequest req,
HttpServletResponse res,
               IErrorHandler error, IMessageHandler message,
ViewHelper helper)
               throws Exception {
           //解码并获取参数
           String pubSqlId = req.getParameter("pubSqlId");
           if(pubSqlId!=null){

  pubSqlId=java.net.URLDecoder.decode(pubSqlId,"UTF-8");
           }
           String params = req.getParameter("params");
           if(params!=null){
              params=java.net.URLDecoder.decode(params,"UTF-8");
           }
           String lparams = req.getParameter("lparams");
           if(lparams!=null){
              lparams=java.net.URLDecoder.decode(lparams,"UTF-8");
           }
           String iparams = req.getParameter("iparams");
           if(iparams!=null){
              iparams=java.net.URLDecoder.decode(iparams,"UTF-8");
           }
```

[0084]

```
            // 查询公用 sql 配置
            PubEasySqlView
pubEasySqlView=getPubEasySqlService().getPubEasySqlConfigInfo(pubSql
Id);

    // 组装公用查询所需参数

            Map<String,Object>                              parameter=new
HashMap<String,Object>();
            parameter.put("pubEasySqlView", pubEasySqlView);
            parameter.put("params", params);
            parameter.put("lparams", lparams);
            parameter.put("iparams", iparams);

    // 执行真正的查询方法
            String                                  Sql_Object_Values=
getPubEasySqlService().doExecutePubEasySql(parameter);
            req.setAttribute("Sql_Object_Values",
Sql_Object_Values);

            return "EasySqlView.success";
        }
```

后台逻辑处理动态参数值核心代码如下：

```
/*
         * 结束开始对 where_var 动态查询条件进行处理
         */
```

[0085]

```
/*
 * 开始对 InWhere 动态查询条件进行处理
 */
if (iparams != null
      && !"".equals(iparams)
      && !"null".equals(iparams)
      && !"undefined".equals(iparams)) {
   // 分号分割变量组,获取全部 In 参数集合
   String[] iparamList = iparams.split(";");

   /**
    * 遍历 iparamList 中所有参数，将所有在 easySqlWhereVar
中配置的变量与单个 iparam 的 key 值进行检索。将匹配的值及其所在的索引位
置等信息保存在 list 中
    *
    */
   for (int i = 0; i < iparamList.length; i++) {
      //根据等号，分隔每个参数，得到当前的参数集合。
      String[] iparamListCur = iparamList[i].split("=");
      // 获取当前用于参数替换的 key 部分
      String              iparamListCurKey              =
"@"+iparamListCur[0].trim()+"@";
      // 获取当前 value 部分
      String iparamListCurValue = "";
      // 存储 in 的参数值集合数组
```

[0086]

```
String[] iparamListCurValueArray;
// 存储 In 动态传入参数的绑定变量字符串
String iparamReplaceStr = "";
// 存储 In 动态传入参数值的个数
int iparamReplaceStrNum=0;
if(iparamListCur.length==2){
   iparamListCurValue=iparamListCur[1].trim();

iparamListCurValueArray=iparamListCurValue.split(",");

StringBuffer iparaReplaceStr = new StringBuffer();
for(int cv = 0; cv < iparamListCurValueArray.length; cv++){
   if(cv<iparamListCurValueArray.length-1){
      iparaReplaceStr.append("?").append(",");
   }else{
      iparaReplaceStr.append(" ? ");
   }
   iparamReplaceStrNum++;
}
if(iparamListCurValueArray.length>0){
   iparamReplaceStr=iparaReplaceStr.toString();
}else{
   iparamReplaceStr = " ? ";
```

[0087]

```
                    }

                }else{
                    iparamListCurValue="";

    iparamListCurValueArray=iparamListCurValue.split(",");
                    iparamReplaceStr = " ? ";
                    iparamReplaceStrNum=1;
                }
                // 获取在 easySqlWhereVar 中出现 key 的全部索引位置
                int iparamListCurKeyIndex =
easySqlWhereAll.indexOf(iparamListCurKey);
                while (iparamListCurKeyIndex != -1) {
                    // 判断是否有值，有则遍历值列表。否则直接添加空
值
                    if (iparamListCur.length == 2) {
                        for(int
vla=0;vla<iparamReplaceStrNum;vla++){
                            easySqlParamViewList.add(new
EasySqlParamView(iparamListCurKeyIndex,  iparamListCur[0].trim(),
iparamListCurValueArray[vla]," ? "));
                        }
                    } else {
                        easySqlParamViewList.add(new
EasySqlParamView(iparamListCurKeyIndex,  iparamListCur[0].trim(),
iparamListCurValueArray[0]," ? "));
```

[0088]

```
                    }
                    iparamListCurKeyIndex                    =
easySqlWhereAll.indexOf(iparamListCurKey,   iparamListCurKeyIndex   +
1);//从这个索引往后开始第一个出现的位置
                }

                easySqlWhereAllFinal                    =
easySqlWhereAllFinal.replaceAll(iparamListCurKey, iparamReplaceStr);
            }
```

。

[0089] 作为本实施例的进一步改进，还包括通过静态查询条件配置与动态查询条件加载相结合的方式对查询信息中数据进行过滤，包括：

[0090] 将不会变化、不需动态传值的查询条件配置于数据库公用查询组件配置表的静态查询条件中；

[0091] 将需要动态传值的查询条件配置于数据库公用查询组件配置表的动态查询条件中；

[0092] 通过客户端前台页面传参的方式对动态查询条件的参数进行参数值传递，以实现查询语句中数据的动态条件过滤。

[0093] 其中，动态条件查询还支持同一个变量多个值的过滤查询操作，还包括通过静态查询条件配置与动态查询条件加载相结合的方式对查询信息中数据进行过滤，包括：

[0094] 将不会变化、不需动态传值的查询条件配置于数据库公用查询组件配置表的静态查询条件中；

[0095] 将需要动态传值的查询条件配置于数据库公用查询组件配置表的静态查询条件中；

[0096] 通过客户端前台页面传参的方式对动态查询条件的参数进行参数值传递，以实现查询语句中数据的动态条件过滤。

[0097] 实施例2：

[0098] 本发明的用于B/S系统的数据查询系统，包括配置模块、查询模块和逻辑处理模块，配置模块用于通过公用查询组件配置查询信息形成查询语句，并用于调用数据库公用查询组件配置表存储上述查询语句；查询模块用于通过客户端前台调用公用查询组件，获取查询结果；逻辑处理模块用于通过后台逻辑调用公用查询组件，对不同查询配置进行统一处理。

[0099] 其中，通过配置模块配置的查询语句为查询SQL语句，包括但不限于查询配置编号、查询字段、表名、查询条件、排序条件、和分组条件，查询条件包括静态查询条件、动态查

询条件、动态In查询条件、动态Like查询条件，上述查询语句至少一个，每个查询语句对应一个查询配置。

[0100] 配置模块具有如下功能：通过公用查询组件配置查询信息形成查询语句，并通过数据库公用查询组件配置表存储上述查询语句，包括如下模式：

[0101] 模式一、基于客户端前台的动态传参方式，在不同查询配置之间进行切换；

[0102] 模式二、基于客户端前台的动态传参方式，动态加载查询配置并传递动态查询条件的动态参数；

[0103] 模式三、基于客户端前台的动态传参方式，对查询语句进行变更维护，包括修改查询字段、表名、查询条件、排序条件和分组条件，支持公用查询组件技术的灵活性。

[0104] 查询模块为具有如下功能的模块：

[0105] (1)通过客户端前台提供JavaScript API封装方法，客户端前台通过引入的JS文件调用公用查询组件，通过传递动态查询条件获取查询结果时，动态查询条件的参数名为相应查询配置中用@@包裹的变量，动态查询条件的参数有多个值时况，所述多个值用逗号分隔，动态查询条件有多个参数时，所述多个参数用分号分隔；

[0106] (2)客户端前台调用executePubEasySqlTable方法，并通过传递查询配置编号或动态查询条件获取查询结果。

[0107] 逻辑处理模块通过如下步骤实现通过后台逻辑调用公用查询组件，对不同查询配置进行统一处理：

[0108] (1)获取查询配置；

[0109] (2)识别查询信息的参数；

[0110] (3)对查询信息的参数进行包装；

[0111] (4)绑定查询信息的参数中的变量；

[0112] (5)基于查询语句执行查询，返回查询结果。

[0113] 其中，配置模块还具有如下功能：通过公用查询组件配置查询信息形成查询语句，还包括通过静态查询条件配置与动态查询条件加载相结合的方式对查询信息中数据进行过滤，具体包括：

[0114] 将不会变化、不需动态传值的查询条件配置于数据库公用查询组件配置表的静态查询条件中；

[0115] 将需要动态传值的查询条件配置于数据库公用查询组件配置表的静态查询条件中；

[0116] 通过客户端前台页面传参的方式对动态查询条件的参数进行参数值传递，以实现查询语句中数据的动态条件过滤。

[0117] 本发明的用于B/S系统的数据查询系统可实现实施例1公开的用于B/S系统的数据查询方法。

[0118] 实施例3：

[0119] 本发明提供一种终端，包括处理器、输入设备、输出设备和存储器，处理器、输入设备、输出设备和存储器相互连接，存储器用于存储计算机程序，该计算机程序包括程序指令，该处理器被配置用于调用所述程序指令执行如实施例1公开的用于B/S系统的数据查询方法。

[0120] 以上所述实施例仅是为充分说明本发明而所举的较佳的实施例，本发明的保护范围不限于此。本技术领域的技术人员在本发明基础上所作的等同替代或变换，均在本发明的保护范围之内。本发明的保护范围以权利要求书为准。

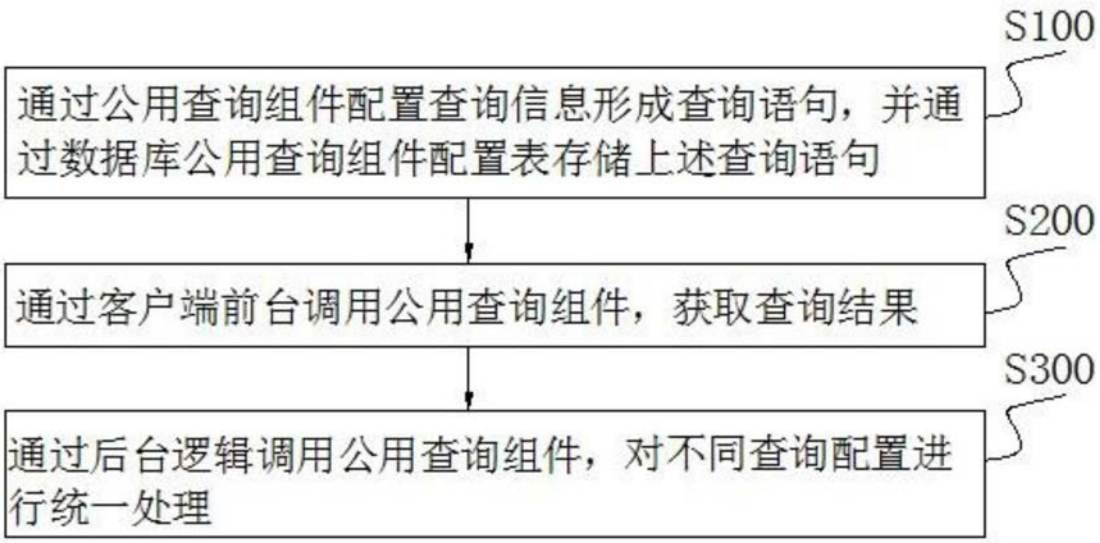

S100
通过公用查询组件配置查询信息形成查询语句，并通过数据库公用查询组件配置表存储上述查询语句
S200
通过客户端前台调用公用查询组件，获取查询结果
S300
通过后台逻辑调用公用查询组件，对不同查询配置进行统一处理

图1