



(12) 发明专利申请

(10) 申请公布号 CN 102739654 A

(43) 申请公布日 2012. 10. 17

(21) 申请号 201210187560. 3

(22) 申请日 2012. 06. 08

(71) 申请人 北京久其软件股份有限公司

地址 100176 北京市北京经济技术开发区西
环中路 6 号

(72) 发明人 李纪洲 王星宇 林敏

(74) 专利代理机构 北京三友知识产权代理有限
公司 11127

代理人 任默闻

(51) Int. Cl.

H04L 29/06 (2006. 01)

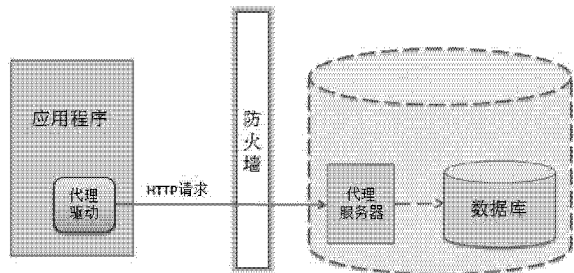
权利要求书 1 页 说明书 7 页 附图 6 页

(54) 发明名称

一种实现应用程序访问数据库的方法

(57) 摘要

一种实现应用程序访问数据库的方法,包括:在防火墙的数据库侧将代理服务器的 war 包部署到中间件中;在防火墙的应用程序侧通过配置所述代理服务器的 url 加入用于访问所述代理服务器的代理 JDBC 驱动;所述的代理 JDBC 驱动调用 webservice 接口向所述的代理服务器发送数据库访问请求;所述的代理服务器接收所述的数据库访问请求,并将所述的数据库访问请求转发给所述的数据库;所述的代理服务器接收所述数据库根据所述数据库访问请求生成的返回结果,并将所述的返回结果输出给所述的代理 JDBC 驱动;所述的代理 JDBC 驱动接收所述的返回结果。通过本发明可以在不进行额外的程序修改的条件下,实现应用程序对数据库的访问,完成防火墙安全穿越。



1. 一种实现应用程序访问数据库的方法,其特征在于,所述的方法包括:
在防火墙的数据库侧将代理服务器的 war 包部署到中间件中;
在防火墙的应用程序侧通过配置所述代理服务器的 url 加入用于访问所述代理服务器的代理 JDBC 驱动;
所述的代理 JDBC 驱动调用 webservice 接口向所述的代理服务器发送数据库访问请求;
所述的代理服务器接收所述的数据库访问请求,并将所述的数据库访问请求转发给所述的数据库;
所述的代理服务器接收所述数据库根据所述数据库访问请求生成的返回结果,并将所述的返回结果输出给所述的代理 JDBC 驱动;
所述的代理 JDBC 驱动接收所述的返回结果。
2. 根据权利要求 1 所述的方法,其特征在于,所述的将代理服务器的 war 包部署到中间件中,包括:设定所述代理服务的访问名,设定所述中间件启动的端口号及配置所述代理服务器的数据源。
3. 根据权利要求 1 所述的方法,其特征在于,所述的数据库访问请求包括:执行命令及查询命令。
4. 根据权利要求 3 所述的方法,其特征在于,所述的代理服务器将所述的返回结果输出给所述的代理 JDBC 驱动,包括:对于执行命令,所述的代理服务器将所述的返回结果直接输出给所述的代理 JDBC 驱动;对于查询命令,所述的代理服务器将所述的返回结果序列化后输出给所述的代理 JDBC 驱动。
5. 根据权利要求 4 所述的方法,其特征在于,将所述的返回结果序列化包括:将所述的返回结果转化为二进制数据。
6. 根据权利要求 5 所述的方法,其特征在于,对于查询命令,所述的代理服务器将所述的返回结果序列化的过程中,根据所述 url 的大小对所述的返回结果进行分页,所述的代理服务器将所述的返回结果以页为单位输出给所述的代理 JDBC 驱动。
7. 根据权利要求 6 所述的方法,其特征在于,对于查询命令,所述的代理 JDBC 驱动接收所述的返回结果的方式为分页接收。
8. 根据权利要求 7 所述的方法,其特征在于,对于查询命令,所述的代理 JDBC 驱动接收所述的返回结果时,对接收到的返回结果进行反序列化,并遍历反序列化后的所有记录,当遍历到最后一条记录时,向所述的代理服务器发送后续数据获取请求。
9. 根据权利要求 7 所述的方法,其特征在于,当所述的代理服务器接收到后续数据获取请求时,向所述的代理 JDBC 驱动发送下一页返回结果,如果所有的返回结果发送完毕,所述的代理服务器向所述的代理 JDBC 驱动返回空。

一种实现应用程序访问数据库的方法

技术领域

[0001] 本发明是关于网络通信技术,特别是关于一种实现应用程序访问数据库的方法。

背景技术

[0002] 应用程序经常要与数据库进行交互,一般情况下,数据库服务器与应用服务器在同一网络环境中,二者可直接进行连接。但在某些情况下,数据库服务器与应用服务器不在同一网络环境,两者之间存在防火墙的阻隔。

[0003] 为解决应用程序与数据库不在同一网络环境时二者的连接问题,现有技术中通常采用下面几种做法:

[0004] 1 采取在防火墙开放端口,允许应用程序直接连接到数据库的方法。

[0005] 如图 1 所示,应用程序访问数据库服务时,一个常规的结构可以划分为若干层,最终通过数据通信层实现应用程序与数据库服务的通信。对于在不同网络环境的应用程序和数据库服务,最直接的连接方式就是通过开发防火墙的端口(如图 2 所示),使得应用程序有能力直接连接到数据库服务。这种方式不需要修改应用程序以及数据库服务的结构。首先,双方需要约定访问协议以及访问数据库的端口号,然后数据库服务器一方根据约定好的协议以及端口号在防火墙上设置开放权限以及网络内部的 IP 映射等设置。最后应用程序方根据开发的端口修改应用程序中的配置,达到连接数据库服务的目的。

[0006] 该方案是跨网络环境访问数据库的相对比较直观的实现方式。但是该方案普遍存在以下几个问题:

[0007] 修改防火墙配置的工作复杂,在复杂的网络环境中,打开特定的端口是一个非常复杂的协调和管理配置的工程。修改网络防火墙的配置增加了开放的协议和端口,可能引起因配置不当而带来的额外的安全隐患。同时将数据库暴露在应用程序的同时,也暴露给其他可能的恶意程序,如果安全配置做的不够完善,则可能引入新的漏洞。

[0008] 2 双方约定数据的中间格式,应用程序并不直接与数据库服务进行通信,而是由数据库方提供接口,再由应用程序方对接口进行调用,如图 3 所示。此方案比较常见的实现方式是双方约定数据格式,提供临时存储空间,由数据库服务方将数据按照约定的数据格式提供到临时存储空间,再有应用程序访问临时存储空间,将数据读取下来。

[0009] 此种方法虽然避免了与数据库服务直接连接,但是需要在数据库服务和应用程序服务端额外的开发应用进行吐数和取数,使得开发量增加,并且不具备可扩展性,可移植性。

[0010] 3 数据库方提供数据访问接口,应用程序方针对数据访问接口进行定制开发,主动取数,如图 4 所示。此种方法不具备通用性,应用程序需要针对不同的接口进行不同的开发,涉及到的程序修改范围大,开发工作量大。当需求变更的时候需要重新开发。

发明内容

[0011] 本发明提供一种实现应用程序访问数据库的方法,使得当数据库服务器与应用程

序服务器不在同一网络时实现应用程序对数据库的访问。

[0012] 为了实现上述目的,本发明提供一种实现应用程序访问数据库的方法,该方法包括:在防火墙的数据库侧将代理服务器的 war 包部署到中间件中;在防火墙的应用程序侧通过配置所述代理服务器的 url 加入用于访问所述代理服务器的代理 JDBC 驱动;所述的代理 JDBC 驱动调用 webservice 接口向所述的代理服务器发送数据库访问请求;所述的代理服务器接收所述的数据库访问请求,并将所述的数据库访问请求转发给所述的数据库;所述的代理服务器接收所述数据库根据所述数据库访问请求生成的返回结果,并将所述的返回结果输出给所述的代理 JDBC 驱动;所述的代理 JDBC 驱动接收所述的返回结果。

[0013] 进一步地,所述的将代理服务器的 war 包部署到中间件中,包括:设定所述代理服务的访问名,设定所述中间件启动的端口号及配置所述代理服务器的数据源。

[0014] 进一步地,所述的数据库访问请求包括:执行命令及查询命令。

[0015] 进一步地,所述的代理服务器将所述的返回结果输出给所述的代理 JDBC 驱动,包括:对于执行命令,所述的代理服务器将所述的返回结果直接输出给所述的代理 JDBC 驱动;对于查询命令,所述的代理服务器将所述的返回结果序列化后输出给所述的代理 JDBC 驱动。

[0016] 进一步地,将所述的返回结果序列化包括:将所述的返回结果转化为二进制数据。

[0017] 进一步地,对于查询命令,所述的代理服务器将所述的返回结果序列化的过程中,根据所述 url 的大小对所述的返回结果进行分页,所述的代理服务器将所述的返回结果以页为单位输出给所述的代理 JDBC 驱动。

[0018] 进一步地,对于查询命令,所述的代理 JDBC 驱动接收所述的返回结果的方式为分页接收。

[0019] 进一步地,对于查询命令,所述的代理 JDBC 驱动接收所述的返回结果时,对接收到的返回结果进行反序列化,并遍历反序列化后的所有记录,当遍历到最后一条记录时,向所述的代理服务器发送后续数据获取请求。

[0020] 进一步地,当所述的代理服务器接收到后续数据获取请求时,向所述的代理 JDBC 驱动发送下一页返回结果,如果所有的返回结果发送完毕,所述的代理服务器向所述的代理 JDBC 驱动返回空。

[0021] 本发明实施例的有益效果在于,当数据库服务器与应用程序服务器不在同一网络时实现应用程序对数据库的访问,完成防火墙安全穿越。基于 JDBC 标准实现的驱动,使得应用程序无缝集成,不需要做额外的程序修改。代理 JDBC 驱动向服务器发送的请求经过加密,保证请求的保密性。自定义的可序列化结果集,使得传输过程中安全高效。传输结果集可根据网络状况设置结果集的压缩级别。传输结果集不是一次性传送到客户端代理驱动,而是分批次传输。每次传输的结果集的大小可以根据实际情况进行设置。

附图说明

[0022] 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一简单地介绍,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性的前提下,还可以根据这些附图获得其他的附图。

- [0023] 图 1 为现有技术中应用程序访问数据库的常规结构示意图；
- [0024] 图 2 为现有技术一中数据库访问的结构示意图；
- [0025] 图 3 为现有技术二中数据库访问的结构示意图；
- [0026] 图 4 为现有技术三中数据库访问的结构示意图；
- [0027] 图 5 为常规模式下应用程序对数据库的访问示意图；
- [0028] 图 6 为本发明应用程序访问数据库的结构示意图；
- [0029] 图 7 为本发明实施例实现应用程序访问数据库的方法流程图；
- [0030] 图 8 为本发明实施例代理服务器部署成功时的响应示意图；
- [0031] 图 9 为本发明实施例结果集的自定义格式示意图；
- [0032] 图 10 为本发明实施例布尔类型 / 字节类型的结果集中的数值结构示意图；
- [0033] 图 11 为本发明实施例整数类型 / 浮点类型的结果集中的数值结构示意图；
- [0034] 图 12 为本发明实施例长整数类型 / 双浮点类型的结果集中的数值结构示意图；
- [0035] 图 13 为本发明实施例字符串类型 / 字节数组类型的结果集中的数值结构示意图；
- [0036] 图 14 为本发明实施例分页接收结果集的流程图。

具体实施方式

[0037] 下面将结合本发明实施例中的附图,对本发明实施例中的技术方案进行清楚、完整地描述,显然,所描述的实施例仅仅是本发明一部分实施例,而不是全部的实施例。基于本发明中的实施例,本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例,都属于本发明保护的范围。

[0038] 图 5 为常规模式下应用程序对数据库的访问示意图,如图 5 所示,当数据库服务器与应用程序服务器在同一网络环境时,应用程序服务器可直接通过 JDBC 驱动连接数据库,但是当数据库服务器与应用程序服务器不在同一网络环境时,应用程序服务器与数据库将被防火墙隔开。

[0039] 本发明提供一种实现应用程序访问数据库的方法,用代理 JDBC 驱动代替图 5 中的 JDBC 驱动,并配置代理服务器的相关地址即可。代理服务器就可以被认为应用程序是所要连接的数据库,而对于代理服务器以外的其他网络部署环境则不需要理会,实现了对应用程序完全透明的访问方式。

[0040] 在使用 JAVA 开发的应用程序中,与数据库进行通信的任务通常交给基于 JDBC 标准实现的数据库驱动来完成的。应用程序不需要针对某种数据库做特定的开发,而只是调用 JDBC 接口规范中提供的方法进行数据库操作即可。

[0041] 本发明正是针对此特性,开发一套基于 JDBC 标准的数据库驱动(代理 JDBC 驱动),用于访问跨防火墙数据库。代理 JDBC 驱动通过修改于数据库的通信协议,使用 webservices 技术进行通信,使得代理 JDBC 驱动具有穿过防火墙的能力,并且不需要在应用程序端做过多的修改。

[0042] 如图 6 所示应用程序只需要更换使用的数据库驱动(将 JDBC 驱动替换为代理 JDBC 驱动)而不需要做额外的操作,就具备了连接跨防火墙数据库的能力。在防火墙的服务器端,提供一个代理服务器,用于接收应用程序通过代理 JDBC 驱动发送的请求并转发给实际

的数据库服务。代理 JDBC 驱动与代理服务器之间的通信通过基于 HTTP 协议的 webservices 请求,可以轻松的穿过防火墙(因为 HTTP 协议对于大部分防火墙而言都是被允许的,即使防火墙禁用了此协议,但由于 HTTP 协议本身的安全性,也可以通过简单的协商沟通而开放防火墙对此协议的限制,并且不会因此而带来额外的风险)。对于应用程序来说,代理服务器则可以认为是要连接的数据库,在应用程序中仅配置代理服务器的相关 url 即可,而不需要关系代理服务器如何与实际的数据库进行连接。

[0043] 如图 7 所示,本实施例提供一种实现应用程序访问数据库的方法,该方法包括:

[0044] 步骤 S701:在防火墙的数据库侧将代理服务器的 war 包部署到中间件中。

[0045] 在应用程序访问数据库之前,首先需要确定防火墙的 HTTP 协议以及端口(通常是 80 端口)是否打开,如果没有打开,则需要修改防火墙配置,开启 HTTP 协议及相应端口。

[0046] 在数据库一侧部署代理服务器,包括:设定所述代理服务的访问名,设定所述中间件启动的端口号及配置所述代理服务器的数据源。首先修改代理服务器中 war 包的用户配置文件 user.xml,登陆用户配置为 user,登陆密码配置为 password。代理服务器需要依赖于中间件的运行,比如 tomcat,weblogic 等。代理服务器的访问名例如可以设定为 teapoy,中间件启动的端口号例如可以设定为 80 端口。需要配置代理服务器的数据源在这里就是实际要连接的数据库。不同的中间件配置数据源的方式不同。这里我们命名数据源的名称为 ds,数据源的连接信息则是实际要连接的数据库的连接信息。配置好后并测试通过,则可以启动代理服务器。

[0047] 代理服务器启动完毕后,在防火墙外侧,通过浏览器访问代理服务器的 services 路径,假设代理服务器所在物理机器的 IP 为 192.168.1.1,则访问全路径为 http://192.168.1.1:80/teapoy/services。如果收到图 8 中的响应,则说明代理服务器部署成功并且在防火墙外部可访问。

[0048] 步骤 S702:在防火墙的应用程序侧通过配置所述代理服务器的 url 加入用于访问所述代理服务器的代理 JDBC 驱动。

[0049] 代理 JDBC 驱动是按照 JDBC 规范实现的一套数据库访问驱动程序。常规模式下,应用程序通过 JDBC 驱动访问数据库(如图 5 所示),当使用此方法访问跨防火墙数据库服务时,需要使用代理 JDBC 驱动来实现。

[0050] 代理 JDBC 驱动的实现原理主要是通过修改底层通信协议以达到跨防火墙访问的目的。常规的 JDBC 驱动,通常使用如 TCP/IP 协议直接访问,但大部分情况下,防火墙会禁止此协议的访问。代理 JDBC 驱动,改用 webservices 技术,通过 HTTP 协议访问服务端。而防火墙对于 HTTP 协议,通常都是启用的。

[0051] 应用程序中需要加入访问代理服务器的代理 JDBC 驱动的全名为 com.jiuqi.teapoy.jdbc.TeapoyDriver。然后拼写访问代理服务器的 url。代理 JDBC 驱动的 url 格式为:

[0052]

```
jdbc:jiuqi:teapoy://[server]:[port];DataSource=[sourcename];
```

[0053] 其中 server 表示代理服务器地址。代理 JDBC 驱动中不需要写协议名称,仅需要代理服务器的机器名(或 IP),默认的服务名称为 teapoy,可以省略;如果服务名称更改,则

需要在服务器地址上加上“/ 服务名称”，例如 :127. 0. 0. 1/jdbc:teapoy。Port 代表服务端口号，默认为 80，可以省略。Sourcename 代表服务器配置的数据源名称。客户端访问的实际数据库。

[0054] 根据上面代理服务器部署的例子，代理 JDBC 驱动的 url 写法应该为：

[0055]

```
jdbc:jiuqi:teapoy://192.168.1.1:80/teapoy;DataSource=ds;
```

[0056] 如果需要指定其他参数，如返回结果（结果集）分页大小，结果集压缩级别等，则在 url 后面追加属性。

[0057]

```
jdbc:jiuqi:teapoy://192.168.1.1:80/teapoy;DataSource=ds;buffer=4096;ziplevel=9
```

[0058] 一个标准的最基本的通过 JDBC 访问数据库的方式为：

[0059]

[0060]

```
String className = ""; // 要加载的数据库驱动类名称
String url = ""; // 要连接的url
String username = ""; // 登陆用户
String password = ""; // 登陆密码

try {
    Class.forName(className);

    Connection conn = DriverManager.getConnection(url, username,
        password);

    // ...得到连接后的后续操作

    conn.close();
} catch (Exception e) {
    e.printStackTrace();
}
```

[0061] 只要将 className, url, username, password 设置为相应的值，应用程序就可以获取到代理服务器的连接。

[0062] 在应用程序中对连接的获取和管理方式有很多种，比如通过数据源配置连接池。但无论哪种方式，只要设置了上面提到的几个属性，即可获取到代理服务器的连接。

[0063] 步骤 S703：所述的代理 JDBC 驱动调用 webservice 接口向所述的代理服务器发送数据库访问请求。

[0064] 数据库访问请求包括：执行命令、查询命令、查询元数据、获取数据库中的连接信息等 JDBC 支持的函数。本发明仅以执行命令及查询命令进行详细说明。

[0065] 应用程序要进行执行及查询操作，需要调用执行命令及查询命令，在应用程序调用代理 JDBC 驱动执行命令及查询命令时，代理 JDBC 驱动接收到命令后，可以先将命令加密，然后调用 webservises 接口发送到代理服务器端。

[0066] 步骤 S704：所述的代理服务器接收所述的数据库访问请求，并将所述的数据库访

问请求转发给所述的数据库。

[0067] 当代理服务器接收到代理 JDBC 驱动发送的请求后,首先会对请求进行解密,得到 SQL 脚本,并且将 SQL 脚本交由实际连接的数据库执行。

[0068] 步骤 S705:所述的代理服务器接收所述数据库根据所述数据库访问请求生成的返回结果,并将所述的返回结果输出给所述的代理 JDBC 驱动。

[0069] 对于执行命令,代理服务器将返回结果直接输出给所述的代理 JDBC 驱动;对于查询命令,代理服务器将返回结果序列化后输出给代理 JDBC 驱动。

[0070] 对于查询命令,代理服务器将所述的返回结果序列化的过程中,根据创建的代理 JDBC 驱动与代理服务器之间连接时 URL 指定的返回结果(结果集)大小对结果集进行分页。代理服务器首先将查询的结果记录依序列化到结果集中,并且判断结果集大小是否达到设置的结果集容量。当达到设置的容量时,代理服务器将序列化结果集发送到代理驱动中(即代理服务器将结果集以页为单位输出给代理 JDBC 驱动),并等待代理驱动获取下批数据的请求。

[0071] 在序列化结果集时,代理服务器会对结果集进行压缩。在创建连接的时候,可以通过 JDBC URL 指定压缩的级别,代理服务器会根据这个级别对结果集进行压缩。

[0072] 序列化结果集是代理服务器获取到数据库返回的基于 JDBC 标准的 ResultSet 对象后,对其进行转化为二进制数据以满足在网络中传输的目的的结果集。

[0073] 如图 9 所示,结果集的格式为自定义格式,首先在结果集中的开头部分,使用一个字节记录数据集的版本号,以后如果数据集格式升级,则通过这个版本号进行区分。后面紧跟一个头信息数据段,用于附加一个结果集的额外信息。其中头四个字节记录了头信息数据段的长度,根据这个长度往后读取相应的字节数作为头信息的数据。

[0074] 结果集中的数值,不同类型的数据将占用不同的字节数。并且用一个字节表示当前字段是否为空。不同类型的数据介绍如下:

[0075] 如图 10 所示,布尔类型/字节类型的数值,使用两个字节表示。第一个字节表示是否为空。如果是 0 即为空,则该数据段结束,进入下一个数据段,以下的数据类型均有相同约定,不再赘述。如果不为空,读取下一个字节。

[0076] 如图 11 所示,整数类型/浮点类型的数值,使用五个字节表示。如果不为空,则读取后四个字节。

[0077] 如图 12 所示,长整数类型/双浮点类型的数值,使用九个字节表示。如果不为空,则读取后八个字节。

[0078] 如图 13 所示,字符串类型/字节数组类型的数据,对于字符串类型的数据,首先按 UTF-8 字符编码转化为字节数据后进行存储。使用第一个字节表示是否为空,如果为空数据段结束。后四个字节表示字符串的字节长度 Length。后面 Length 个字节为数据。

[0079] 步骤 S706:所述的代理 JDBC 驱动接收所述的返回结果。

[0080] 对于查询命令,一次查询的结果集的记录数可能很多,不可能通过一次传输把所有的结果传输到代理驱动中,因此代理 JDBC 驱动接收结果集的方式为分页接收。

[0081] 图 14 为本发明实施例分页接收结果集的流程。代理 JDBC 驱动接收所述的返回结果时,对接收到的当前页的结果集进行反序列化,并遍历当前页反序列化后的所有记录,当遍历到当前页的最后一条记录时,向代理服务器发送后续数据获取请求。代理服务器接

收到代理 JDBC 驱动发来的后续数据获取请求后,向代理 JDBC 驱动发送下一页数据。此时如果代理服务器返回另一批(下一页)结果集时,则表示还有后续数据。当所有数据全部发送完毕后,代理驱动再次发送获取数据的请求时,代理服务器向代理 JDBC 驱动返回空,表示此次查询结果发送完毕。

[0082] 本发明实现应用程序访问数据库的方法是利用基于 webservices 技术和 JDBC 标准开发的代理驱动,为了解决在不同网络环境下跨防火墙访问数据库的一种通用实现方法,这种方法是可移植的,并且无需编码,可适用于任何环境。该方法避免了为了访问数据库而带来的复杂的网络配置,减少了沟通成本、实施成本,也减少了可能带来的安全隐患。同时,该方法不需要做额外的程序开发,只需要简单的部署即可实现,减少了因此而带来的开发成本。对于使用该方法的应用程序,仅仅需要替换一下连接数据库使用的 JDBC 驱动,并且配置代理服务器的相关地址即可。代理服务器就可以被认为应用程序是所要连接的数据库,而对于代理服务器以外的其他网络部署环境则不需要理会,实现了对应用程序完全透明的访问方式。

[0083] 采用标准技术 webservices 和 jdbc,可以最大限度减少对现有应用的修改和维护。

[0084] 当数据库服务器与应用程序服务器不在同一网络时实现应用程序对数据库的访问,完成防火墙安全穿越。基于 JDBC 标准实现的驱动,使得应用程序无缝集成,不需要做额外的程序修改。代理 JDBC 驱动向服务器发送的请求经过加密,保证请求的保密性。自定义的可序列化结果集,使得传输过程中安全高效。传输结果集可根据网络状况设置结果集的压缩级别。传输结果集不是一次性传送到客户端代理驱动,而是分批次传输。每次传输的结果集的大小可以根据实际情况进行设置。该方法可以无缝的溶于用户现有的 IT 基础设施中,对现有的网络配置策略,信息安全策略,系统管理策略没有任何影响。

[0085] 以上所述的具体实施方式,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方式而已,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。

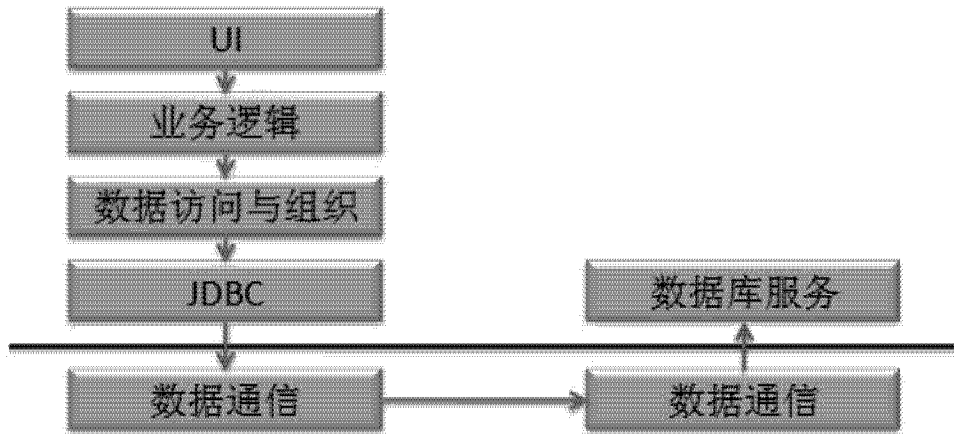


图 1

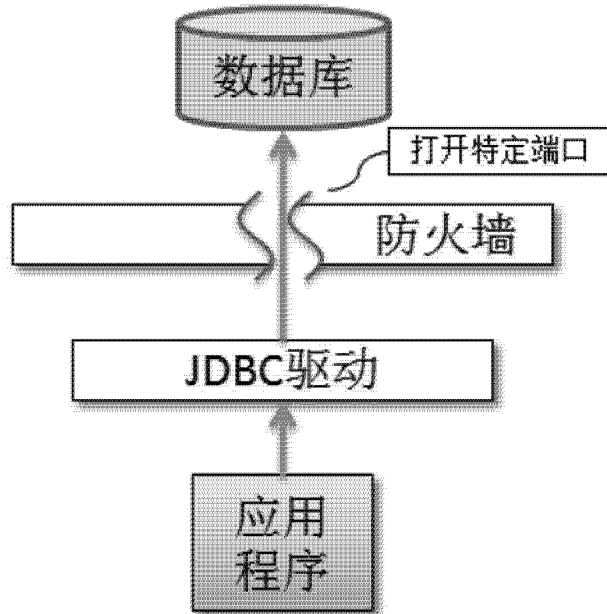


图 2

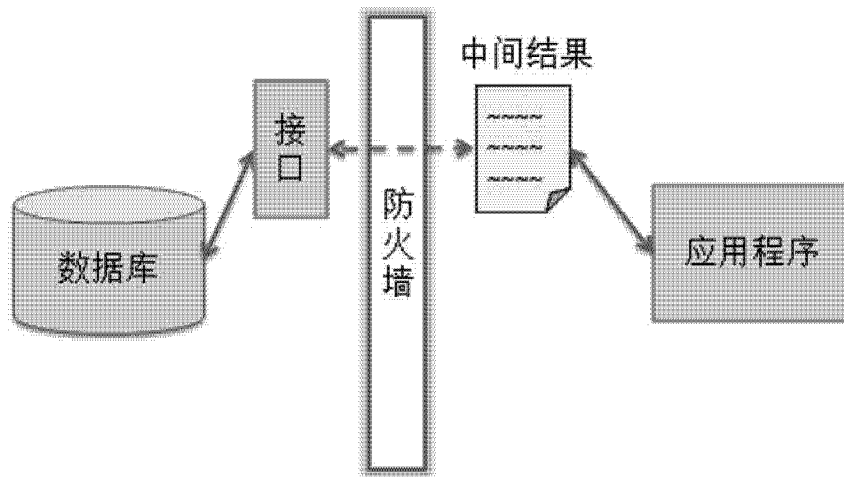


图 3

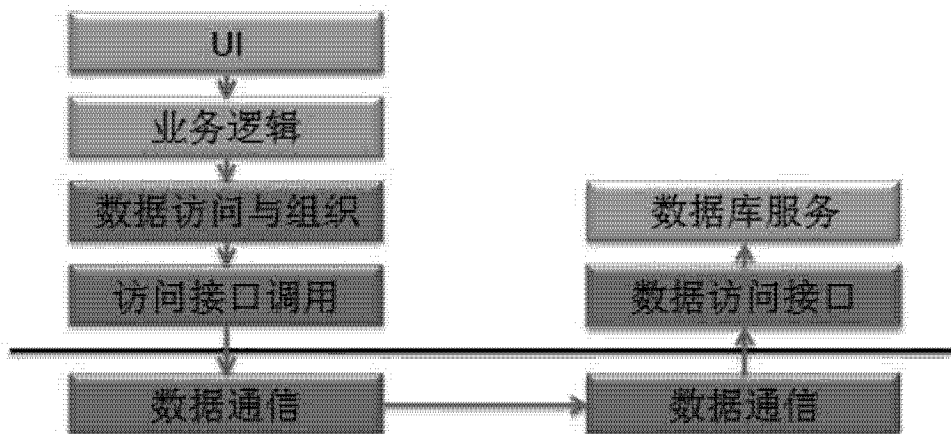


图 4

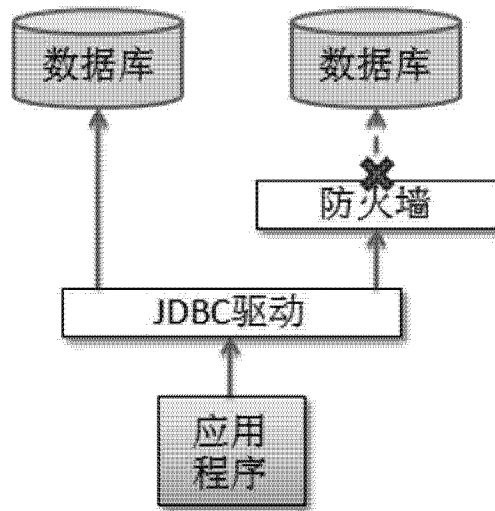


图 5

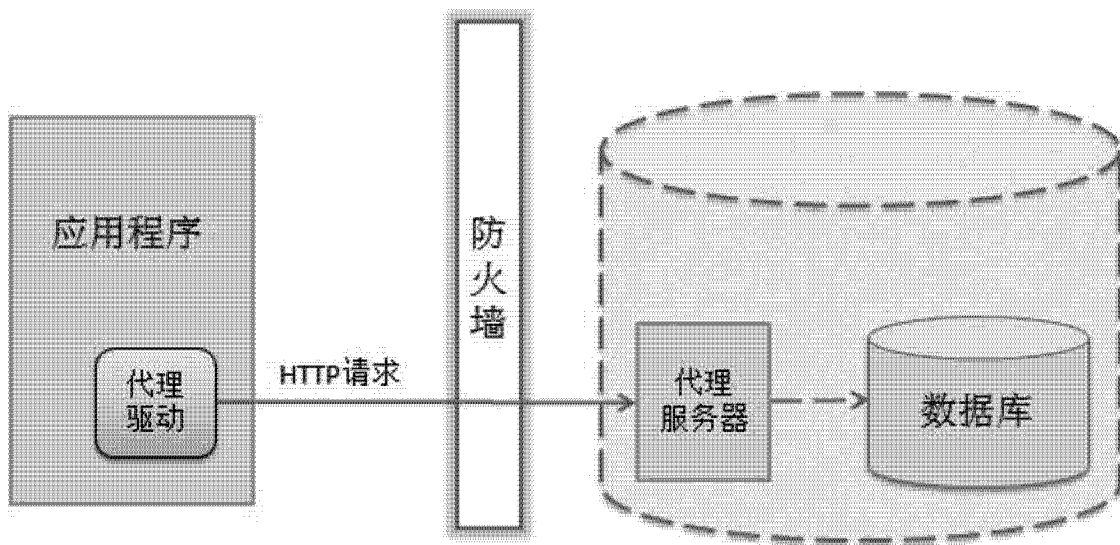


图 6

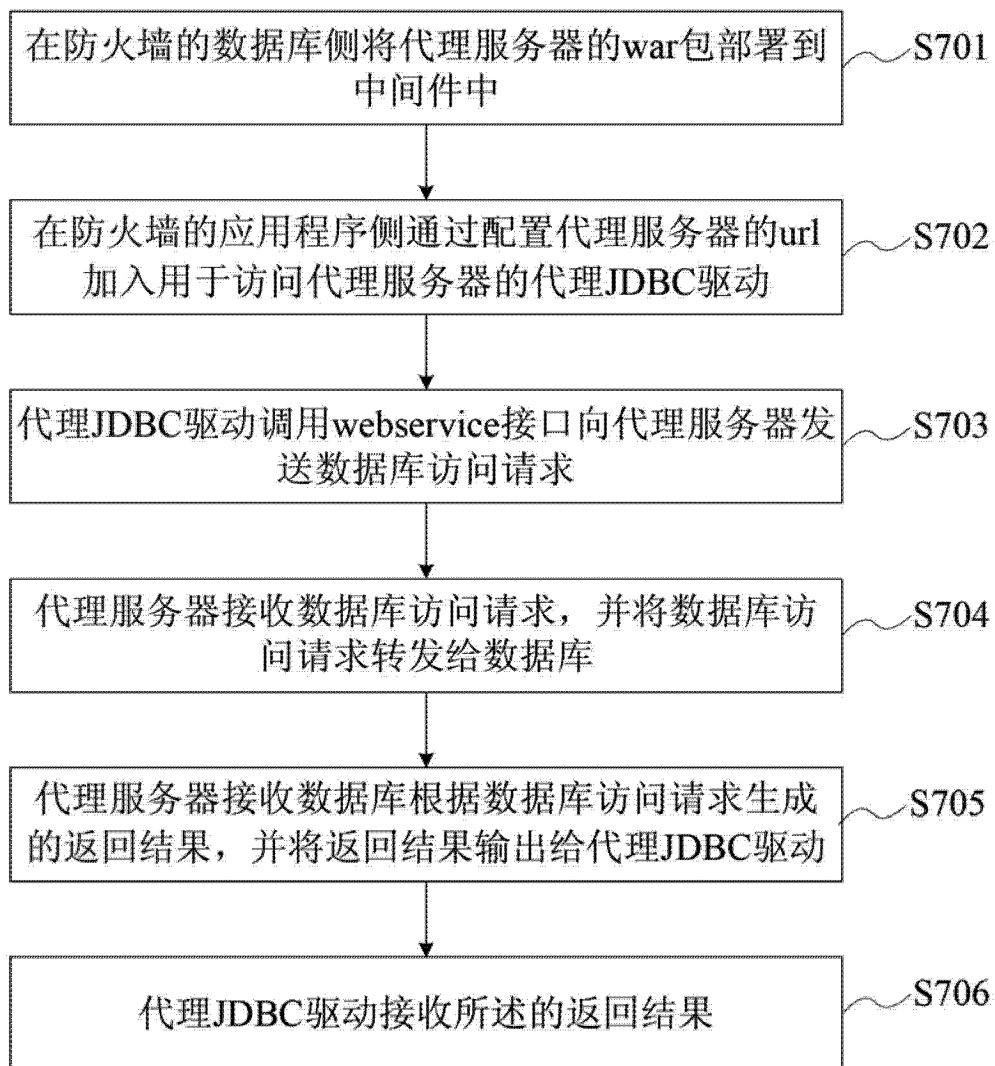


图7



图 8

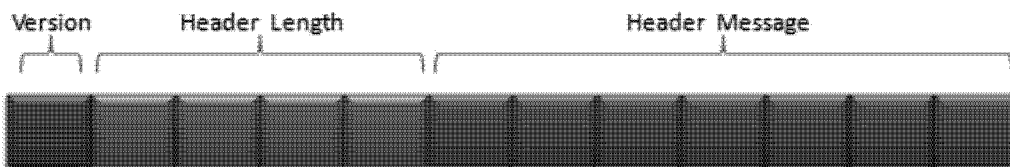


图 9

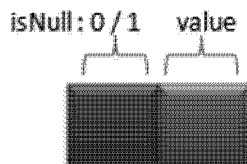


图 10

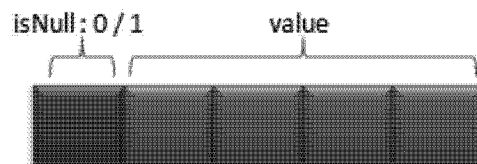


图 11

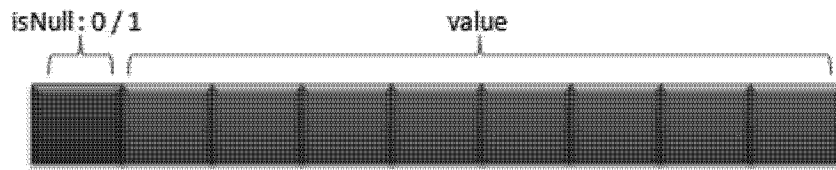


图 12

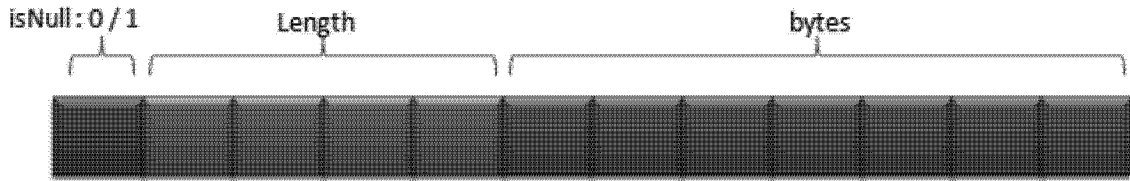


图 13

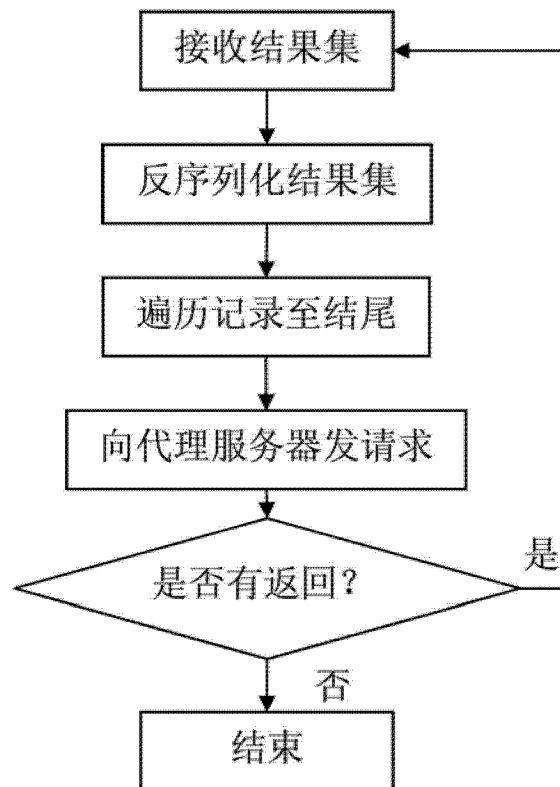


图 14