**oneproxy-monitor使用手册**

|  |  |  |  |
| --- | --- | --- | --- |
| 编 写 | 黄辉 | 编写 时间 | 2016-12-29 |
| 审 批 | 审批者姓名（及其职务） | 审批 时间 |  |
| 版本 | V1.0 | | |

**平民软件有限公司版权所有**

目录

[1.引言](#_Toc472080505)

[1.1编写目的](#_Toc472080506)

[1.2 项目背景](#_Toc472080507)

[1.3 定义](#_Toc472080508)

[1.4 使用范围](#_Toc472080509)

[2. 软件概述](#_Toc472080510)

[2.1 目标](#_Toc472080511)

[2.2 功能](#_Toc472080512)

[2.2.1 监控功能](#_Toc472080513)

[2.2.2 多端口功能](#_Toc472080514)

[2.2.3 reuseport功能](#_Toc472080515)

[2.2.4 前后端密码分离](#_Toc472080516)

[2.2.5 连接池](#_Toc472080517)

[2.2.6 读写分离](#_Toc472080518)

[2.2.7 连接清理](#_Toc472080519)

[2.2.8 负载均衡](#_Toc472080520)

[2.2.9 数据库混合搭配](#_Toc472080521)

[2.2.10 显示交互数据](#_Toc472080522)

[2.2.11 显示前端操作的sql语句](#_Toc472080523)

[2.2.12 在线更新配置](#_Toc472080524)

[2.2.13 keepalive功能](#_Toc472080525)

[2.2.14 虚拟ip(vip)功能](#_Toc472080526)

[2.2.15 审计功能](#_Toc472080527)

[3. 安装](#_Toc472080528)

[4. 获取帮助](#_Toc472080529)

[5配置文件说明](#_Toc472080530)

[5.1 [oneproxy]节的配置](#_Toc472080531)

[5.2 [database\_xxxx]节的配置](#_Toc472080532)

[5.3 数据库组节的配置](#_Toc472080533)

[6. 附录](#_Toc472080534)

[6.1 混合数据库的配置样例](#_Toc472080535)

[6.2 sql server数据库配置样例](#_Toc472080536)

[6.3 postgresql 数据库配置样例](#_Toc472080537)

[6.4 直接转发时的配置样例](#_Toc472080538)

1.引言

1.1编写目的

编写本文档是为了帮助用户了解本软件的安装与使用、了解软件的功能、性能等特性。

读者对象：使用本软件的所有用户。

1.2 项目背景

在当前大数据量的情况下，很多场景中需要把数据进行实时复制，采用一对多的复制方式。其中一份数据用于查询以及备份，另外一份数据用于修改等，在这种场景中需要用到读写分离的功能特性。

很多场景中前端应用操作数据库都是采用短连接的方式，而在短连接中建立连接的过程占用了很大比例的时间。如果能够加快连接的建立过程，则可以大大的提供数据库的性能。为了解决这个问题，目前的解决方案是使用连接池的功能，在连接池中针对建立的连接进行维护和管理。本项目中提供了一种稍微有点差异的连接池，既可以有效的维护连接，又可以动态调整连接池的大小。

现在市场上面的数据库的种类繁多，并且每个数据库都有自己的优点和缺点。同一家公司中可能使用了多种数据库。随着数据库种类的增加，管理也特别的困难。前端应用程序需要在不同的应用程序中写入不同数据库的个性化数据。本项目可以同时管理多种数据库，方便前端应用程序对数据库的使用。

1.3 定义

oneproxy-monitor: 平民软件提供的数据库中间件

1.4 使用范围

由于oneproxy-for-postgresql, oneproxy-for-sqlserver都是在oneproxy-monitor框架下开发的，故本配置也适合于上述两个程序的配置。

oneproxy-monitor使用的范围为：

1. 如果只是转发，不进行协议解析。则不受限制
2. 进行协议解析：

i)sqlserver支持2008及以后的版本。

ii) postgresql支持9.4及以上的版本。

iii)长连接不能有效的使用连接池。

2. 软件概述

2.1 目标

1) 提供监控页面，在监控页面中提供登录信息、sql语句信息、事务信息、数据库使用量信息等。

2) 提供多个端口功能，并且提供从不同端口接收到的数据转发到不同数据库，或者同一个数据库的功能。

3) 提供负载均衡的功能，可以在同一个数据库组中的不同数据库间进行负载均衡操作。

4) 提供连接池功能，使用连接池来管理连接

5) 提供前后端密码分离功能，加强安全性管理

6) 提供读写分离功能。从slave数据库读取数据，写数据到master数据库。

2.2 功能

2.2.1 监控功能

oneproxy-monitor提供了部分监控的功能，通过监控功能可以查看到应用程序的登录和查询数据的情况，以及操作了哪些表和操作方式。可以在配置文件中通过httpserver\_addr和httpserver\_port两个参数来指定监控页面的地址和端口信息。下面是默认的配置：

httpserver\_addr = 0.0.0.0

httpserver\_port = 8080

下面详细介绍所有的监控功能：

#### 2.2.1.1 监控首页

在监控首页上面提供了当前主机的系统信息，部分查询的sql语句，部分登录的应用程序信息，以及oneproxy-monitor处理任务的负载信息（见图2.2.1.1.1）。

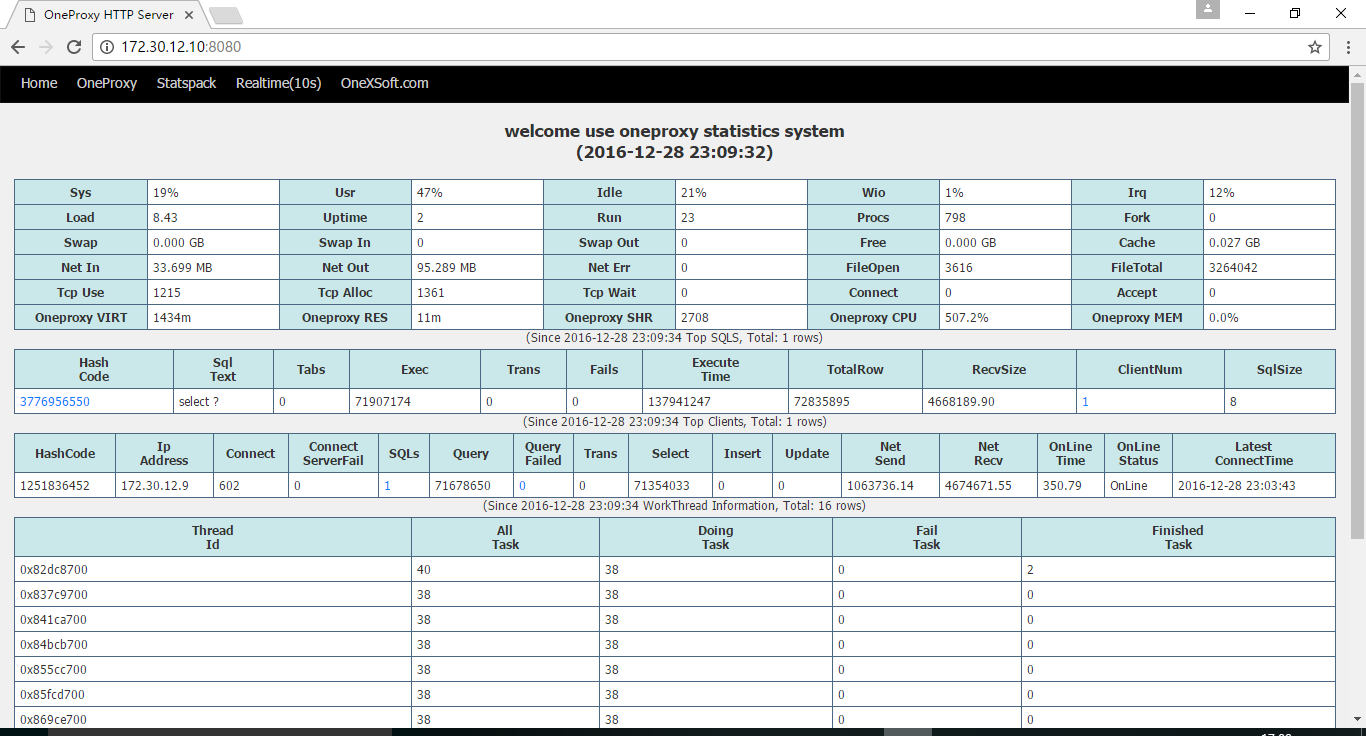


图2.2.1.1.1

从上图可以看到，整个首页分成了4部分，从上往下依次是：主机信息、sql语句信息、客户端操作信息、oneproxy-monitor任务负载信息。

针对表格中title的解释可以直接把鼠标停留在title上面，即可查看（见图2.2.1.1.2）。

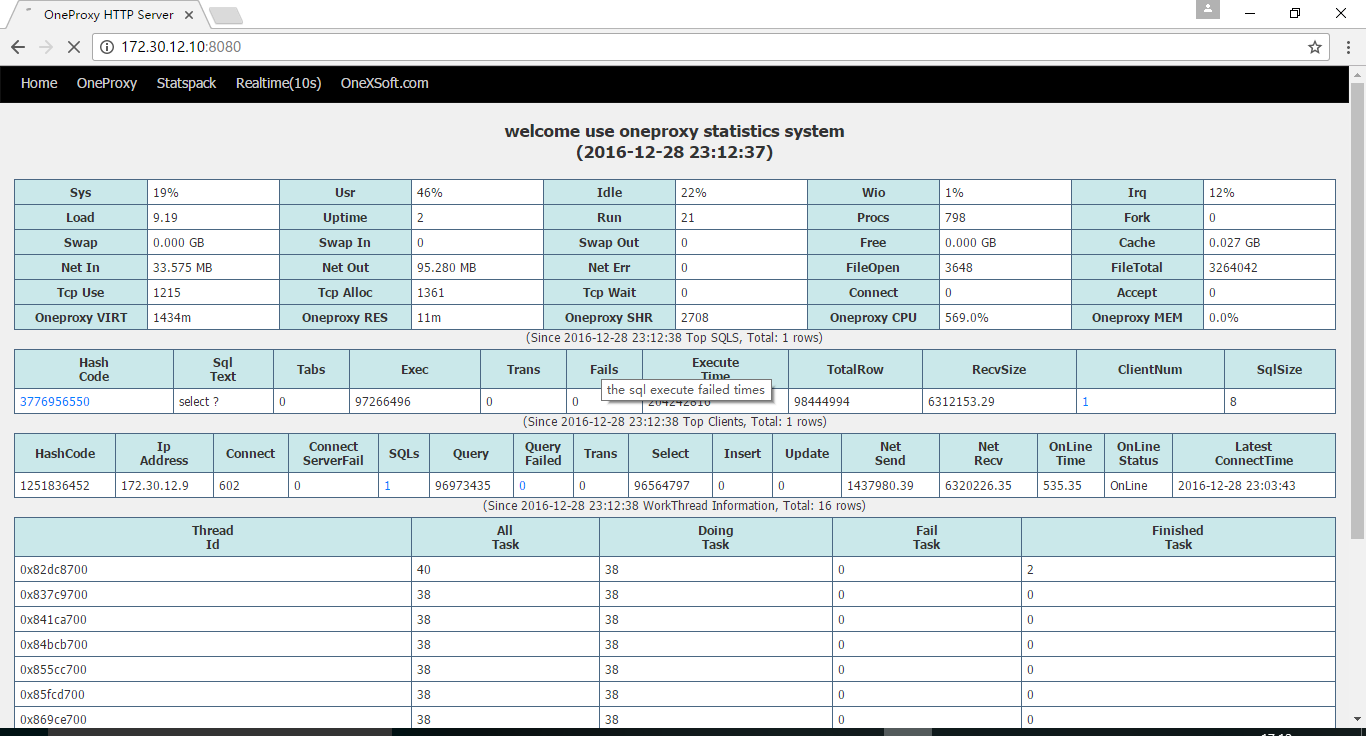


图2.2.1.1.2

上图中，把鼠标停留在了第二部分的Fails表格上面，则就弹出了提示信息。这个提示信息就是针对Fails的解释。

首页的显示情况也可以进行个性化配置的，目前支持选择是否显示sql语句、是否显示客户端操作信息、是否显示oneproxy-monitor任务负载情况。这个配置可以通过Home->Setting来配置（见图2.2.1.1.3）

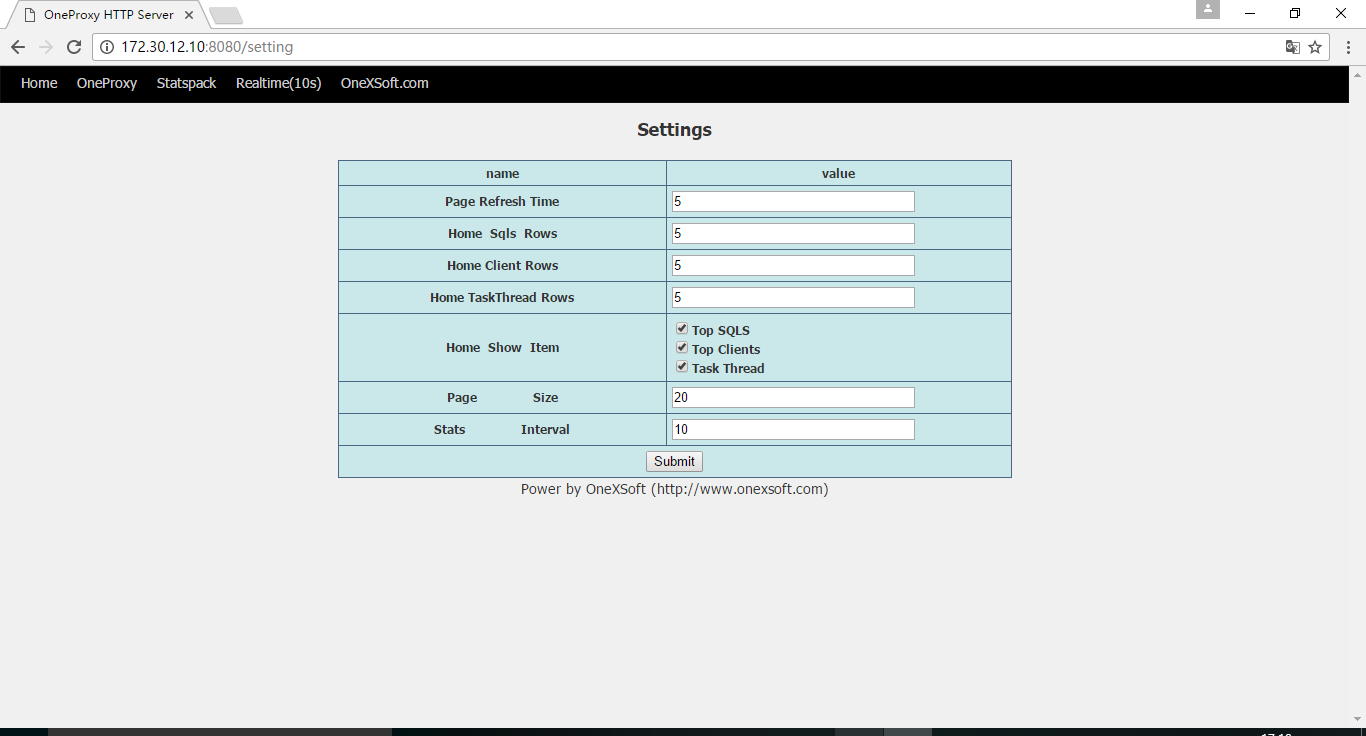


图2.2.1.1.3

上图为setting页面的情况，可以通过选择Home Show Item中的不同选项来决定首页的显示情况。

#### 2.2.1.2 数据库信息

在OneProxy->DataBase页面提供了数据信息情况（见图2.2.1.2.1）。在这个页面中显示了每个数据库的地址、端口、权值、连接数情况、以及是否为active。

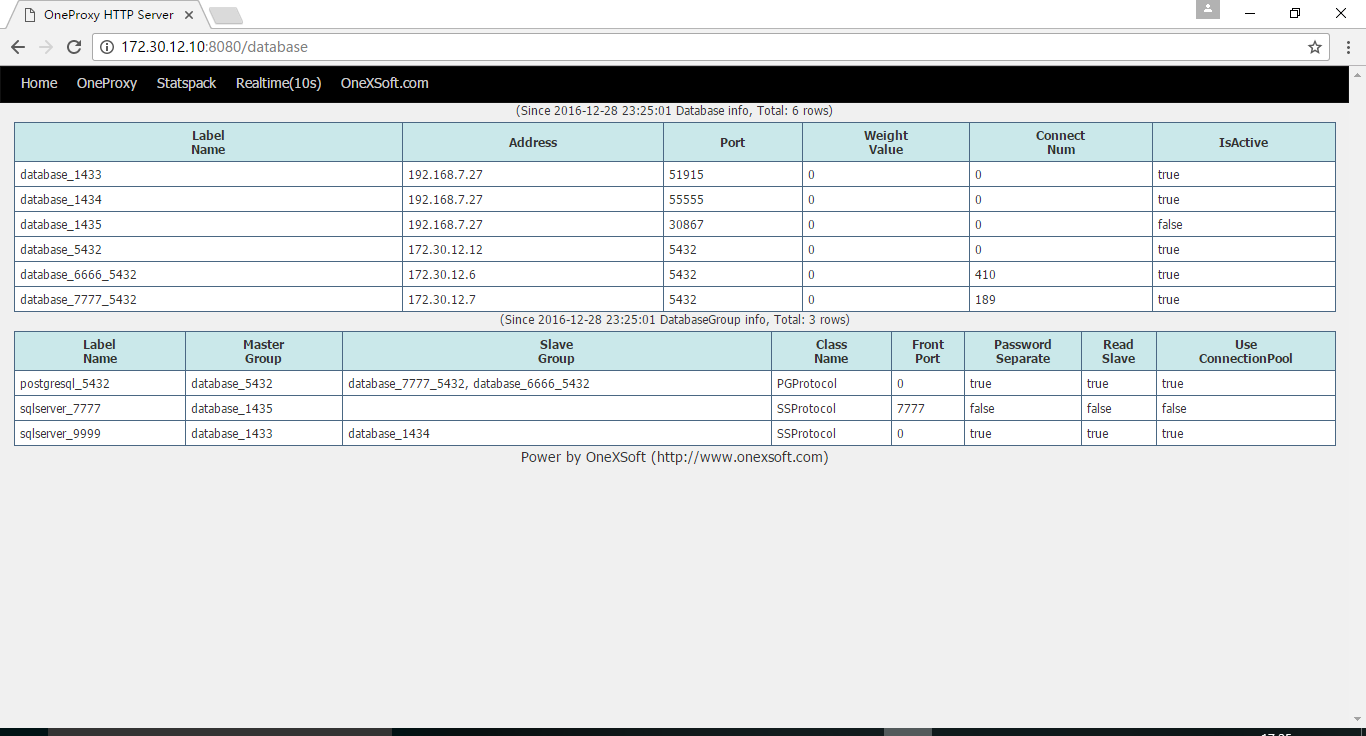


图 2.2.1.2.1

上图页面被分成了两个部分，第一部分是单个数据库的情况。在单个数据库中LabelName是单个数据库的别名，是在配置文件中指定的。这个别名不能够重复，并且必须以database\_开头。

第二部分是数据库组情况，前端只能够连接到对应的数据库组上面，其中数据库组的LabelName可以是任意的，但是不能是database\_开头。数据库组需要指定master数据库，slave数据库的labelName。如果有多个master数据库或者多个slave数据库则以逗号进行分隔，如上图所示。同时需要指定使用的处理类，目前只提供了三个处理类，sqlserver使用SSProtocol. postgresql数据库使用PGProtocol。其他的使用FakeProtocol。数据库组中的FrontPort字段指定处理前端哪个端口来的数据，如果为0表示处理所有前端端口的数据。

#### 2.2.1.3 任务信息

Oneproxy->Task页面中显示的是oneproxy-monitor当前的任务情况（见图2.2.1.3.1）。

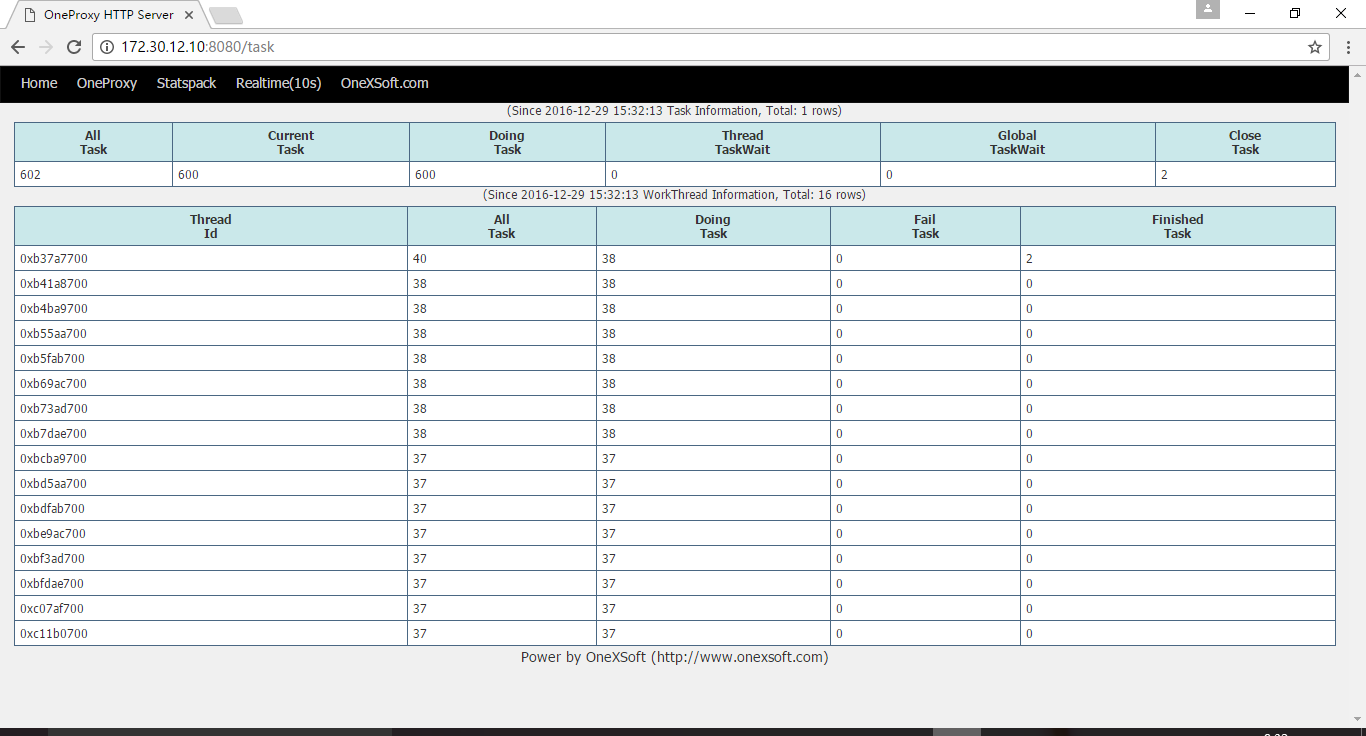


图2.2.1.3.1

Task页面也分成了两部分：第一部分是从oneproxy-monitor启动以来执行任务的情况。第二部分是每个工作线程执行任务的情况。每个部分都有总的处理任务的数量（AllTask），正在处理的任务数量(DoingTask)。在第一部分中还有Thread TaskWait表示任务已经分配给线程，但是线程在处理的过程中前端还没有获取到后端数据库连接的任务数量。GlobalTaskWait表示还没有分配给工作线程的任务的数量。如果连接过多，已经超过配置中maxconnectnum指定的任务时，则会进去到GlobalTaskWait中进行排队处理。如果前端连接没有超过maxconnectnum,则不会进去全局排队队列，直接分配给了工作线程。

第一部分的closeTask是第二部分的所有FinishedTask之和。第二部分的FailTask是当前工作线程处理失败的任务的数量。

#### 2.2.1.4 sql统计

Statspack->Top SQLS页面展示从启动或者重置开始到现在的sql执行情况（见图2.2.1.4.1）。

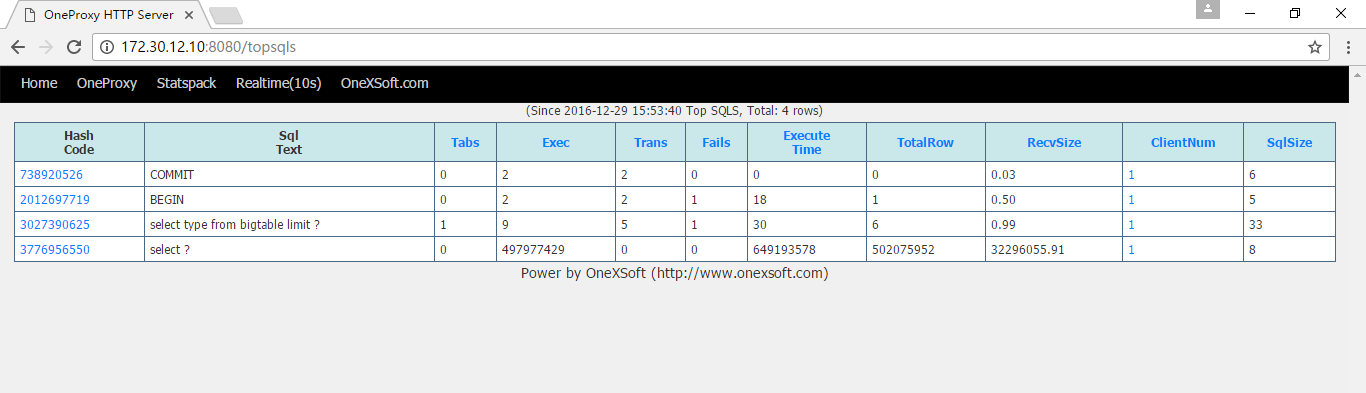


图2.2.1.4.1

可以通过点击每条sql语句前的HashCode查看完整的信息。clientNum字段显示了有多少个客户端执行了此条sql语句，也可以通过点击表格的title来按照相应的字段进行排序。每个字段的含义可以通过把鼠标停留在title上面进行查看，在此就不过多进行解释了。

#### 2.2.1.5 client统计

Statspack->Top Clients页面展示的是客户端的信息（见图2.2.1.5.1），客户端的信息是以ip地址为维度进行统计的，前面的hashcode是针对IP地址进行计算得到的。

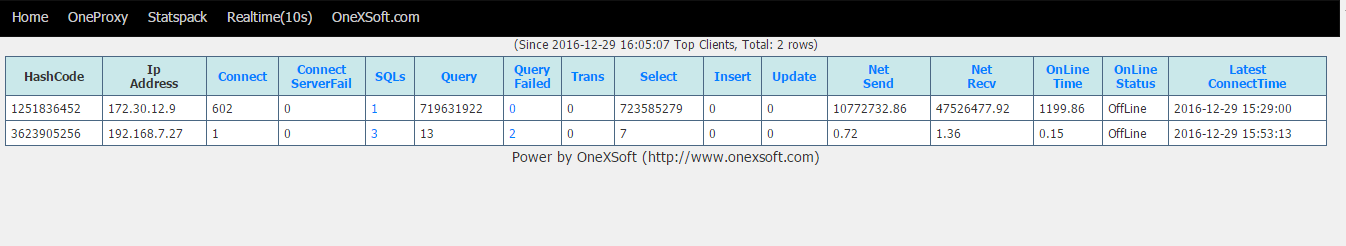


图2.2.1.5.1

下面表格是部分字段的解释：

|  |  |
| --- | --- |
| Connect | 对应的客户端建立连接的次数或者登录的次数 |
| SQLs | 当前客户端执行sql语句的条数，通过点击对应的数字 可以查看具体的sql语句 |
| QueryFailed | 是在sql语句执行的过程中得到ERROR包的次数 |
| Trans | 客户端执行事务的数量 |
| OnLineTime | 在线的累计时间 |
| OnLineStatus | 当前是否在线 |
| LatestConnectTime | 最后一次登录的时间 |

#### 2.2.1.6 table统计

Statspack->Top Tables页面是针对数据库表的使用统计（图2.2.1.6.1）。

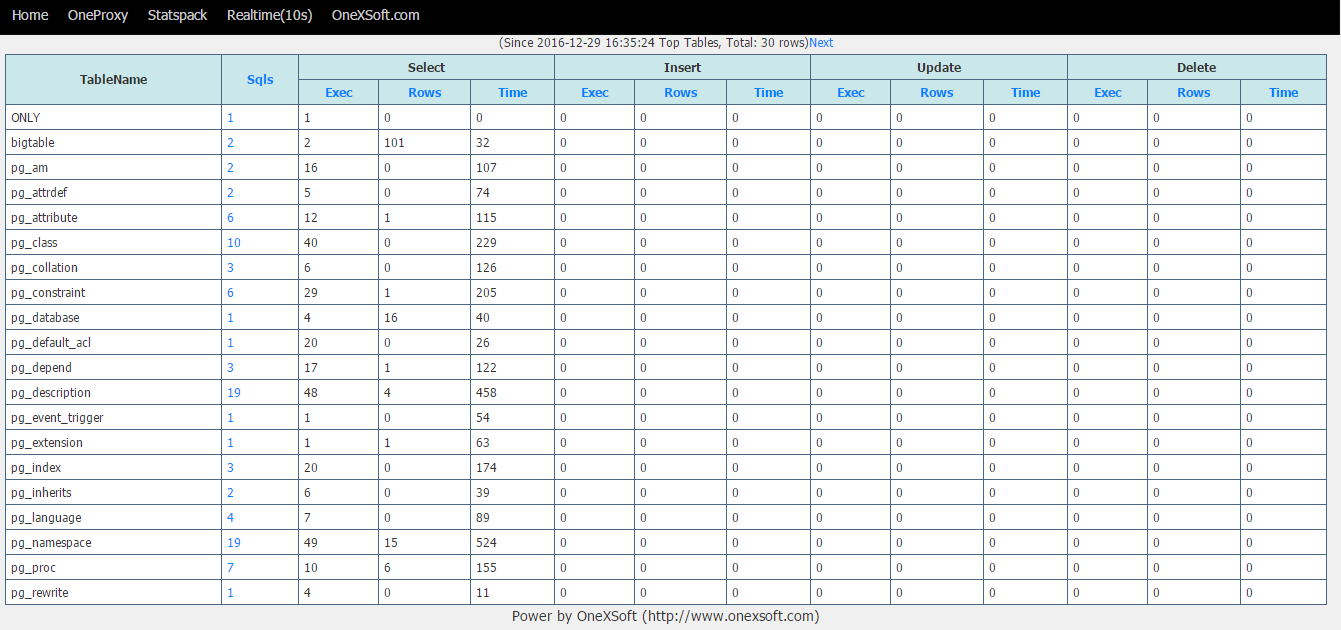


图2.2.1.6.1

在表的统计中，是按照select, insert, update, delete维度来进行统计的。可以通过sqls来查看有多少个sql语句包含了此表。

#### 2.2.1.7 表和SQL的映射关系统计

Statspack->Top TablesMap页面展示的是表与sql语句的关系（图2.2.1.7.1）。



图2.2.1.7.1

从图中可以看到一组表可能在多个sql语句中被使用。通过这个统计可以查看出sql语句的复杂度。也可以看到具有强复杂度的sql语句的数量。

2.2.2 多端口功能

oneproxy-monitor在读取客户端连接的时候是系统调用，如果有大量的并发请求，则可能导致listen得队列满而导致部分客户端无法连接的问题。而从listen队列中读取客户端连接是accept函数进行系统调用，每次只能读取一个连接。当有大量并发请求时，此处有瓶颈问题。oneproxy-monitor提供了注册多个监听端口的解决方案，这样可以把不同的应用程序连接到不同的端口，或者不同的数据库使用不同的端口。具体的配置方法如下：

设置config标签下的listen\_port的值，如果是多个端口，则以逗号分隔，例如：

listen\_port = 9999,7777,6666

这样就开启多个监听端口了。如果需要把不同端口得到的数据转发到不同的数据库组，则还需要配置数据库组下的frontport的值为对应的监控端口号，例如：

[sqlserver\_7777]

dbmastergroup = database\_1435

classname = SSProtocol

frontport = 7777

sqlserver\_7777数据库组就只会处理从7777监听端口读取到的数据。如果想让sqlserver\_7777处理所有前端监听端口的数据，则可以配置frontport为0即可。

2.2.3 reuseport功能

Linux在2.6.32-573及以上版本中提供了reuseport功能。同一个进程中可以使用多个线程针对同一个端口进行监听。Linux自动进行负载均衡，把进行了三次握手后的连接分配到不同的accept调用线程中。Linux通过这种方式来解决大量并发的问题，oneproxy-monitor目前也已经支持这个特性，可以通过配置config标签下的acceptthreadnum来决定使用多少个线程进行监听。默认配置为2，同时受cpu core 的数量的限制。如果只有一个cpu core，oneproxy-monitor把acceptthreadnum设置为1。

2.2.4 前后端密码分离

oneproxy-monitor提供了密码分离功能，这样前端应用程序中可以不用根据不同的数据库使用不同的密码了。如果需要使用此功能，则需要进行如下配置：

1. 在config标签下的clientusername设置客户端使用的用户名
2. 在config标签下的clientpassword设置客户端使用的密码
3. 在数据库组的标签下设置passwordseparate = true（默认配置为true）。如果需要设置数据库组不使用读写分离，则可以设置passwordseparate = false。例如：

[sqlserver\_7777]

dbmastergroup = database\_1435

classname = SSProtocol

frontport = 7777

passwordseparate = false

1. 在数据库标签下设置对应的数据库的密码，即登录到数据库需要的用户名和密码。例如：

[database\_1435]

host = 192.168.7.27

port = 30867

username = sa

password = 0000

1. 如果是postgresql数据库，则需要在数据库组下增加clientauthmethod为md5(默认为md5)。指定前端的验证方式为md5的方式。

通过上面五步的配置后，则数据库组就开启了读写分离功能，前端需要使用config标签下指定的用户名和密码进行登录。oneproxy-monitor则采用数据库标签下对应的用户名和密码登录到数据库。

2.2.5 连接池

为了加快客户端建立连接的速度，oneproxy-monitor提供了连接池功能。oneproxy-monitor提供的连接池功能不主动创建连接，只对连接进行管理。oneproxy-monitor提供的连接池的原理如下：首先客户端发起连接，oneproxy-monitor检查连接池中是否有连接，如果没有，则向对应的数据库发起连接请求，建立连接。并且把建立后的连接分配给客户端。当客户端完毕后，把连接到数据库的连接保存到连接池中。连接池会隔一段时间就检查连接是否已经空闲超时，如果超时了还没有被使用，那么就是否此连接。

从oneproxy-monitor连接池的原理来看，如果客户端向oneproxy-monitor发起请求的频率是恒定的，则连接池中的连接能够有效的被复用。如果客户端暴涨暴跌的发起连接，oneproxy-monitor也能够不受限制的发起到数据库的连接。连接池也能够及时的释放空闲的连接，从而降低数据库的连接数。

使用连接池功能，则需要进行如下的配置：

1. 配置**支持前后端密码分离**。
2. 配置config标签下的poolconncheckactivetime的值。此配置是指定在多长时间对连接池中的连接进行检查是否还是active，如果不是active就释放连接。单位为秒。默认为5秒。
3. 配置config标签下的poolconntimeoutreleasetime的值。如果poolconntimeoutreleasetime时间内连接没有被使用，则连接会被连接池释放掉。单位是秒。默认为60秒。
4. 配置数据库组下的useconnectionpool的值为TRUE。默认为TRUE。

例如：

[postgresql\_5432]

dbmastergroup = database\_5432

dbslavegroup = database\_7777\_5432, database\_6666\_5432

classname = PGProtocol

frontport = 0

passwordseparate=true

useconnectionpool=true

clearbackendconnection = true

1. 配置clearbackendconnection的值为true，当连接在放回连接池时是否进行清理操作。postgresql必须配置为true。

经过上面4个步骤的配置，就可以使用连接池功能了。

**切记**：一定配置支持前后端密码分离。

**注：**为了支持prepared方式，目前只支持短连接使用连接池。如果是长连接就没有连接池的效果。

2.2.6 读写分离

为了从slave数据上面读取数据，写数据到master上面。oneproxy-monitor也提供了读写分离的功能。

配置oneproxy-monitor支持读写分离的方法为：

1. 配置oneproxy-monitor**支持前后端密码分离**
2. 在数据库组下增加readslave = true的配置即可

例如：

[sqlserver\_9999]

dbmastergroup = database\_1433

dbslavegroup = database\_1434

classname = SSProtocol

readslave = true

passwordseparate = true

**切记**：配置支持前后端密码分离。

2.2.7 连接清理

如果需要把连接放到连接池之前进行连接的清理工作，则需要配置支持连接清理的选项。可以通过在数据库组下增加clearbackendconnection=true。默认情况下已经配置为TRUE。

建议所有数据库都设置为TRUE。

2.2.8 负载均衡

oneproxy-monitor支持同一个数据库组配置多个slave的方式。当前端连接到来时，则可以通过均衡的方式把连接分配到不同的数据库上面去。为了支持负载均衡，则需要进行下列的配置：

1. 配置每个数据库的权值（weightvalue），默认为0。weightvalue的取值范围为0-100。如果都是0，则会**大概**均衡的把连接分配到不同的数据库上面。
2. 配置多个需要均衡处理连接的数据库到数据库组中。例如：

[database\_5432]

host = 172.30.12.12

port = 5432

username = db\_user

password = 0000

[database\_7777\_5432]

host = 172.30.12.7

port = 5432

username = db\_user

password = 0000

[database\_6666\_5432]

host = 172.30.12.6

port = 5432

username = db\_user

password = 0000

[postgresql\_5432]

dbmastergroup = database\_5432

dbslavegroup = database\_7777\_5432, database\_6666\_5432

classname = PGProtocol

frontport = 0

则上面的配置会把读进行负载均衡到两个slave数据库上面。

默认情况下是根据前端连接分配得到的数据库的情况进行负载均衡的，oneproxy-monitor中还提供了根据真正到数据库的连接数的情况进行负载均衡。可以通过在数据库组中增加：usedatabaseconnecttobalace来指定不同的方式。

usedatabaseconnecttobalace = true 使用正在建立到数据库的连接的情况来进行负载均衡。针对短连接比较多的情况下比较有效。

usedatabaseconnecttobalace = false 使用预分配给前端连接的数据库情况来进行负载均衡，这种情况下可能某些客户端连接并没有建立到数据库的连接。这种方式针对长连接比较有效。

#### 2.2.8.1 master提供查询服务

如果需要把master数据库也当做slave数据库作为负载均衡部分的数据库，则可以在数据库组的标签下增加：usemasterassalve=true。

例如：

[postgresql\_5432]

dbmastergroup = database\_5432

dbslavegroup = database\_7777\_5432, database\_8888\_5432

classname = PGProtocol

frontport = 0

passwordseparate=true

useconnectionpool=true

readslave = true

clearbackendconnection = true

**usemasterassalve = true**

2.2.9 数据库混合搭配

oneproxy-monitor已经支持多种数据库进行混搭的模式。目前支持sqlserver, postgresql进行统计，读写分离等功能，其他的数据库可以直接进行转发。使用的方法是：

1. 开启**多端口功能**
2. 不同的端口使用不同的处理类进行处理，目前支持的处理类有SSProtocol, PGProtocol, FakeProtocol。其中SSProtocol是针对sql server的处理类，PGProtocol是针对postgresql的处理类。其他的数据库则可以使用FakeProtocol进行直接转发处理。下面的配置是一个支持混合数据库处理的配置：

[oneproxy]

logfile = oneproxy\_log.log

pidfile = oneproxy\_pid.pid

listen\_addr = 0.0.0.0

listen\_port = 9999,7777,6666

clientusername = sa

clientpassword = 0000

[database\_1433]

host = 192.168.7.27

port = 51915

username = sa

password = 0000

[database\_1434]

host = 192.168.7.27

port = 55555

username = sa

password = abcdefuz

[database\_1435]

host = 192.168.7.27

port = 30867

username = sa

password = 0000

[database\_5432]

host = 172.30.12.12

port = 5432

username = db\_user

password = 0000

[database\_7777\_5432]

host = 172.30.12.7

port = 5432

username = db\_user

password = 0000

[database\_6666\_5432]

host = 172.30.12.6

port = 5432

username = db\_user

password = 0000

[database\_1521]

host = 172.30.12.10

port = 1521

[sqlserver\_9999]

dbmastergroup = database\_1433

dbslavegroup = database\_1434

classname = SSProtocol

useconnectionpool = true

readslave = true

passwordseparate = true

frontport=9999

[postgresql\_5432]

dbmastergroup = database\_5432

dbslavegroup = database\_7777\_5432, database\_6666\_5432

classname = PGProtocol

frontport = 7777

passwordseparate=true

useconnectionpool=true

readslave = true

clearbackendconnection = true

[other\_database]

dbmastergroup = database\_1521

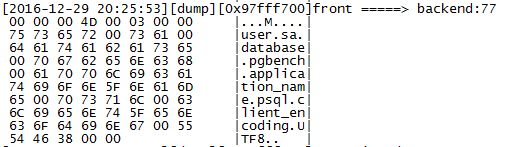
classname = FakeProtocol

frontport = 6666

上面的配置指出，客户端可以通过9999端口来处理sqlserver的请求，通过连接7777端口来处理postgresql的请求，通过连接6666连处理其他数据库的请求。

2.2.10 显示交互数据

oneproxy-monitor也可以使用作为调试工具，通过开启config下的data\_dump功能来打印前端和后端交互的数据。输出格式如下所示：



上面的数据是从前端发送到后端的数据，以及大小（77）。

2.2.11 显示前端操作的sql语句

有时为了监控客户端的操作情况，需要知道客户端进行了哪些查询。则需要输出进行操作的sql语句。目前只支持sqlserver和postgresql。则可以通过设置config标签下的log\_sql为true即可。

2.2.12 在线更新配置

oneproxy-monitor支持在线更新配置，从而保证在不中断前端连接的情况下实现oneproxy-monitor的在线更新。操作的方法为：

1. 更新配置文件config.ini或者可执行程序。
2. 像第一次启动程序一样启动可执行程序(例如：./oneproxy-monitor)

经过上面的两步，就可以完成oneproxy-monitor的在线更新配置。

**注意点：**

1. 需要配置配置文件中oneproxy节下的pidfile。例如：

[oneproxy]

logfile = oneproxy\_log.log

pidfile = oneproxy\_pid.pid

1. 在线更新配置使用了pid文件，如果没有pid文件，将无法让原来的oneproxy-monitor退出。故启动后生成的pid文件不能人为删除，程序退出时会自动删除自身生成的pid文件。
2. 原来的oneproxy-monitor在新的oneproxy-monitor启动后就停止接收客户端的连接。但是不会马上退出，会等待已经连接的程序执行完毕后才退出。故如果存在长连接的话，会导致oneproxy-monitor很难退出。如果长连接一直在工作（有数据交换），原来的oneproxy-monitor不会退出，如果长连接没有工作（没有数据交换），则在指定的时间后被强制关闭。通过在oneproxy节下的connecttimeout来指定超时时间，单位为second。这个超时时间是指没有数据交换后的超时时间，如果一直有数据交换，此时间不影响长连接的操作。默认为：86400秒。

2.2.13 keepalive功能

oneproxy-monitor提供了keepalive功能，这样当程序在意外退出情况下，可以被父进程快速启动，从而不影响向前端提供服务。需要注意的是当oneproxy-monitor运行在前台时，可以通过ctrl+c的方式来退出程序。

配置方法：

oneproxy节下面增加keepalive=1。例如：(见黑体)

[oneproxy]

logfile = oneproxy\_log.log

pidfile = oneproxy\_pid.pid

listen\_addr = 0.0.0.0

listen\_port = 9999,7777,6666

httpserver\_addr = 0.0.0.0

httpserver\_port = 8080

log\_level = error

data\_dump = false

log\_sql = false

clientusername = sa

clientpassword = 0000

**keepalive = 1**

acceptthreadnum = 1

maxconnectnum = 40000

在启动oneproxy-monitor，通过ps可以看到有两个oneproxy-monitor的进程存在，一个是父进程，一个是子进程。

2.2.14 虚拟ip(vip)功能

oneproxy-monitor提供了vip功能，可以通过在配置文件中的oneproxy节下面增加

vip\_ifname和vip\_address配置来使用。例如：(见黑色部分)

[oneproxy]

logfile = oneproxy\_log.log

pidfile = oneproxy\_pid.pid

listen\_addr = 0.0.0.0

listen\_port = 9999,7777,6666

httpserver\_addr = 0.0.0.0

httpserver\_port = 8080

log\_level = error

data\_dump = false

log\_sql = false

clientusername = sa

clientpassword = 0000

keepalive = 1

acceptthreadnum = 1

maxconnectnum = 40000

**vip\_ifname = em1:0**

**vip\_address = 172.30.12.22**

上面的配置中em1是本主机的网卡名称。em1:0的含义是在已经存在的物理网卡em1上面建立虚拟网卡em1:0。虚拟IP地址为172.30.12.22。前端应用程序则可以通过172.30.12.22来访问oneproxy-monitor了。

oneproxy-monitor会在后台不停的检测当前的vip地址是否可以ping通，如果不通时，则会把vip设置在自身的物理网卡上面。从而达到局域网内多个oneproxy-monitor的协同工作。

2.2.15 审计功能

为了查看客户端进行了哪些查询，以及在查询时使用的条件等。oneproxy-monitor提供了把prepared中的动态变量回填显示的功能。

可以通过在配置文件中[config]标签下增加logclientsql=true来开启此功能，默认情况下是关闭的。

输出的sql语句是回填值后的sql语句，输出到日志文件中。下面是输出的示例：

[2017-01-12 20:14:08][0x2fc1700][192.168.7.58:43564] select dbtype from dbo.bigtable where id = 10

[2017-01-12 20:14:08][0x2fc1700][192.168.7.58:43564] set implicit\_transactions on

[2017-01-12 20:14:08][0x2fc1700][192.168.7.58:43564] select dbtype from dbo.bigtable where id = 10

[2017-01-12 20:14:08][0x2fc1700][192.168.7.58:43564] IF @@TRANCOUNT > 0 COMMIT TRAN

[2017-01-12 20:14:08][0x2fc1700][192.168.7.58:43564] IF @@TRANCOUNT > 0 COMMIT TRAN set implicit\_transactions off

[2017-01-12 20:14:08][0x2fc1700][192.168.7.58:43564] select dbtype from dbo.bigtable where id = 10

[2017-01-12 20:14:08][0x2fc1700][192.168.7.58:43564] begin tran

[2017-01-12 20:14:08][0x2fc1700][192.168.7.58:43564] select dbtype from dbo.bigtable where id = 10

[2017-01-12 20:14:08][0x2fc1700][192.168.7.58:43564] commit

[2017-01-12 20:14:08][0x2fc1700][192.168.7.58:43564] select dbtype from dbo.bigtable where id = 10

[2017-01-12 20:14:08][0x2fc1700][192.168.7.58:43564] set implicit\_transactions on

[2017-01-12 20:14:08][0x2fc1700][192.168.7.58:43564] select id, name, age, dbtype from dbo.bigtable where id = 10

[2017-01-12 20:14:08][0x2fc1700][192.168.7.58:43564] IF @@TRANCOUNT > 0 COMMIT TRAN

[2017-01-12 20:14:08][0x2fc1700][192.168.7.58:43564] select id, name, age, dbtype from dbo.bigtable where id < 10

3. 安装

1) 下载可执行程序和配置文件

<https://github.com/onexsoft/oneproxy-monitor/tree/master/bin>

配置文件叫做config.ini。配置文件只要是以.ini结尾即可。

1. 把配置文件和可执行程序放在同一个目录下
2. 直接执行可执行程序即可（Windows下双击可执行程序，Linux下使用命令行执行）。或者使用-f参数指定配置文件来启动可执行程序。

4. 获取帮助

1) 可以从平民软件官网获取联系方式

2) 加入群《数据库监控 521095285》

3）关注https://github.com/onexsoft/oneproxy-monitor

5配置文件说明

配置文件的格式见附录。本节主要针对配置文件的组织关系进行说明，方便读者的使用与配置。

oneproxy-monitor的配置文件被分成了3个大的部分：

第一部分：[oneproxy]节，这个节下面的配置信息是oneproxy-monitor的全局配置信息。

第二部分：[database\_xxxx]节，这个是配置后端数据库信息的节，节的名称必须为database\_开头。

第三部分：其他的节（除了[oneproxy]节,[database\_xxxx]节），是数据库组的信息，这些信息与每个连接相关。

下面介绍每个节完整的配置：

5.1 [oneproxy]节的配置

|  |  |  |  |
| --- | --- | --- | --- |
| 配置名 | 默认值 | 取值范围 | 功能说明 |
| logfile | oneproxy\_log.log | 有效文件名 |  |
| pidfile | oneproxy\_pid.pid | 有效文件名 | 见2.2.12 |
| listen\_addr | 0.0.0.0 |  |  |
| listen\_port | 9999 |  | 见2.2.2 |
| httpserver\_addr | 0.0.0.0 |  | 见2.2.1 |
| httpserver\_port | 8080 |  | 见2.2.1 |
| log\_level | error | fatal, error, warning, info,debug |  |
| data\_dump | false | false, true | 见2.2.10 |
| log\_sql | false | false, true | 见2.2.11 |
| tryconnservertimes | 3 | 如果小于1，自动设置为1 | 连接数据库的重试次数 |
| maxconnectnum | 2000 | 如果小于1， 自动设置为1 | 数据库的连接总数 |
| keepalive | false | true,false | 见2.2.13 |
| vip\_ifname |  |  | 见2.2.14 |
| vip\_address |  |  | 见2.2.14 |
| threadnum | 0 | 如果小于等于0，自动设置为cpu core数 | 工作线程数量 |
| clientusername | admin |  | 见2.2.4 |
| clientpassword | 0000 |  | 见2.2.4 |
| poolconncheckactivetime | 5 | 大于0， 单位为秒 | 见2.2.5 |
| poolconntimeoutreleasetime | 60 | 大于0， 单位为秒 | 见2.2.5 |
| connecttimeout | 86400 | 大于0， 单位为秒 | 见2.2.12 |
| acceptthreadnum | 1 | 大于0 | 见2.2.3 |
| listenbacklog | 1000 | 大于等于128 | socket的监听队列 |

5.2 [database\_xxxx]节的配置

|  |  |  |  |
| --- | --- | --- | --- |
| 配置名 | 默认值 | 取值范围 | 功能说明 |
| labelname | database\_fakedb |  | 配置中没有配置取节名称 |
| host | 127.0.0.1 |  | 数据库地址 |
| port | 1433 |  | 数据库端口 |
| weightvalue | 0 |  | 权值 |
| username | sa |  | 用户名 |
| password | 0000 |  | 密码 |
| connectnumlimit | 1000 |  | 连接数 |

5.3 数据库组节的配置

|  |  |  |  |
| --- | --- | --- | --- |
| 配置名 | 默认值 | 取值范围 | 功能说明 |
| labelname | group\_fake |  | 配置中没有配置取节名 |
| dbmastergroup | database\_fakedb |  | master数据库的节名（多个以逗号分隔） |
| dbslavegroup |  |  | slave数据库的节名(多个以逗号分隔) |
| classname | FakeProtocol | SSProtocol, PGProtocol, FakeProtocol | 处理类的名称。  SSProtocol是sql server数据库的处理类  PGProtocol是postgresql数据库的处理类  FakeProtocol是直接转发的类 |
| frontport | 0 |  | 见2.2.2 |
| passwordseparate | true |  | 见2.2.4 |
| readslave | true |  | 见2.2.6 |
| useconnectionpool | true |  | 见2.2.5 |
| clientauthmethod | md5 |  | 见2.2.4 |
| clearbackendconnection | true |  | 见2.2.5 |

6. 附录

6.1 混合数据库的配置样例

[oneproxy]

logfile = oneproxy\_log.log

pidfile = oneproxy\_pid.pid

listen\_addr = 0.0.0.0

listen\_port = 9999,7777,6666

clientusername = sa

clientpassword = 0000

[database\_1433]

host = 192.168.7.27

port = 51915

username = sa

password = 0000

[database\_1434]

host = 192.168.7.27

port = 55555

username = sa

password = abcdefuz

[database\_1435]

host = 192.168.7.27

port = 30867

username = sa

password = 0000

[database\_5432]

host = 172.30.12.12

port = 5432

username = db\_user

password = 0000

[database\_7777\_5432]

host = 172.30.12.7

port = 5432

username = db\_user

password = 0000

[database\_6666\_5432]

host = 172.30.12.6

port = 5432

username = db\_user

password = 0000

[database\_1521]

host = 172.30.12.10

port = 1521

[sqlserver\_9999]

dbmastergroup = database\_1433

dbslavegroup = database\_1434

classname = SSProtocol

useconnectionpool = true

readslave = true

passwordseparate = true

frontport=9999

[postgresql\_5432]

dbmastergroup = database\_5432

dbslavegroup = database\_7777\_5432, database\_6666\_5432

classname = PGProtocol

frontport = 7777

passwordseparate=true

useconnectionpool=true

readslave = true

clearbackendconnection = true

[other\_database]

dbmastergroup = database\_1521

classname = FakeProtocol

frontport = 6666

6.2 sql server数据库配置样例

[oneproxy]

logfile = oneproxy\_log.log

pidfile = oneproxy\_pid.pid

listen\_addr = 0.0.0.0

listen\_port = 9999,7777,6666

httpserver\_addr = 0.0.0.0

httpserver\_port = 8080

log\_level = error

data\_dump = false

log\_sql = false

clientusername = sa

clientpassword = 0000

keepalive = 0

acceptthreadnum = 1

maxconnectnum = 40000

#threadnum = 1

[database\_1433]

host = 192.168.7.27

port = 51915

username = sa

password = 0000

[database\_1434]

host = 192.168.7.27

port = 55555

username = sa

password = abcdefuz

[database\_1435]

host = 192.168.7.27

port = 30867

username = sa

password = 0000

[sql server\_5432]

dbmastergroup = database\_1433

dbslavegroup = database\_1434, database\_1435

classname = SSProtocol

frontport = 0

6.3 postgresql 数据库配置样例

[oneproxy]

logfile = oneproxy\_log.log

pidfile = oneproxy\_pid.pid

listen\_addr = 0.0.0.0

listen\_port = 9999,7777,6666

httpserver\_addr = 0.0.0.0

httpserver\_port = 8080

log\_level = error

data\_dump = false

log\_sql = false

clientusername = sa

clientpassword = 0000

keepalive = 0

acceptthreadnum = 1

maxconnectnum = 40000

#threadnum = 1

[database\_5432]

host = 172.30.12.12

port = 5432

username = db\_user

password = 0000

[database\_7777\_5432]

host = 172.30.12.7

port = 5432

username = db\_user

password = 0000

[database\_6666\_1435]

host = 172.30.12.6

port = 5432

username = db\_user

password = 0000

[sql server\_5432]

dbmastergroup = database\_5432

dbslavegroup = database\_7777\_5432, database\_6666\_1435

classname = PGProtocol

frontport = 0

6.4 直接转发时的配置样例

[oneproxy]

logfile = oneproxy\_log.log

pidfile = oneproxy\_pid.pid

listen\_addr = 0.0.0.0

listen\_port = 9999,7777,6666

httpserver\_addr = 0.0.0.0

httpserver\_port = 8080

log\_level = error

data\_dump = false

log\_sql = false

clientusername = sa

clientpassword = 0000

keepalive = 0

acceptthreadnum = 1

maxconnectnum = 40000

#threadnum = 1

[database\_5432]

host = 172.30.12.12

port = 5432

username = db\_user

password = 0000

[database\_7777\_5432]

host = 172.30.12.7

port = 5432

username = db\_user

password = 0000

[sql server\_5432]

dbmastergroup = database\_5432

dbslavegroup = database\_7777\_5432

classname = FakeProtocol

frontport = 0