[玩转 MySQL 之二]MySQL 连接机制浅析及运维

作者：bangbang
原文链接：https://ld246.com/article/1536248398959
来源网站：链滴

前言

使用MySQL数据库的第一步必然是建立连接登录，然后在上面执行SQL命令。无论是通过mysql的客端，还是通过C-API，JDBC标准接口连接数据库，这个过程一定少不了。那么就不经有几个疑问？**1. 可以通过哪几种方式连接?**

- 当C和S都在同一台机器上时，那他们之间的通信方式是否和进程间的通信差不多?
- 当C和S不在一台服务器上时候，是采用tcp来通信？还是使用http?
- 当C和S不在一台服务器上时，不论是采用tcp或者http通信，都会有安全风险，是否有加密措施?

2. MySQL的连接有没有区别?

- 如果MySQL的连接有区别，分为哪几种？他们之间的区别是什么

3. MySQL服务器是如何管理客户端连接请求的?

- 连接的时候，客户端每法送一个请求，MySQL服务器就重新创建一个连接么？如果不是，那么服务端是如何处理的?
- MySQL服务器的连接数有没有限制?
- 连接数限制能否精确到用户?
- C/S之间的连接是否有时间限制?

4. MySQL服务器相关的连接配置参数有哪些?

5. MySQL服务器中连接相关信息如何查看?

6. 客户端连接MySQL服务器会出现哪些错误以及解决办法?

一、MySQL连接方式

到MySQL5.7为止，总共有五种连接方式，分别是TCP/IP，TLS/SSL，Unix Sockets，Shared Memory，Named pipes，具体区别如下：

连接方式	默认是否开启	支持系统	只支持本机	何开启	参数配置
TCP/IP	是	所有系统	否	–skip-networking=yes/no	–port和 –bind-address
TLS/SSL	是	所有系统（基于TCP/IP)之上	否	ssl=yes/no	–ssl-* options
Unix Sockets	是	类Unix系统	是	置–socket=来关闭.	–socket=socket path
Shared Memory	否	Windows系统	是	shared-memory=on/off	–shared-memory-base-name=
Named pipes	否	Windows系统	否	enable-named-pipe=on/off	–socket=

使用教程

由于小编使用的是Mac OS系统，下面只简单介绍非Windows系统下的三种连接方式。**1.Unix Sockets**

```
mysql -uroot
```

如果在MySQL本机使用这种方式连接MySQL数据库，默认使用Unix Sockets。

2.TCP/IP

```
mysql --protocol=tcp -uroot
mysql -P3306 -h127.0.0.1 -uroot
```

连接的时候指定连接协议，或者指定相应的IP及端口，则连接方式就变成了TCP/IP方式。

3.TLS/SSL

```
mysql --protocol=tcp -uroot --ssl=on
mysql -P3306 -h127.0.0.1 -uroot --ssl=on
```

TLS/SSL是基于TCP/IP的，所以只需再指定打开ssl配置即可。然后我们通过以下语句来查询目前数据的连接情况：

```
SELECT DISTINCT connection_type from performance_schema.threads where connection_type
s not null
```

如何选择连接方式呢？1. 如果程序和数据库在同一台机子(类Unix系统)上，
荐使用Unix Sockets，因为它效率更高；2. 若数据库分布
不同的机子上，且能确保连接安全或者安全性要求不是那么高，推荐使用TCP/IP，反之使用TLS/SSL；

二、 MySQL连接分类

当数据库服务器和客户端位于不同的主机时，就需要通过建立网络连接来进行通信。客户端必须使用
据库连接来发送命令和接收应答、数据。根据服务器的处理机制分为短连接和长连接。

1.短连接

短连接是指程序和数据库通信时需要建立连接，执行操作后，连接关闭。短连接简单来说就是每一次
作数据库，都要打开和关闭数据库连接，基本步骤是：连接→数据传输→关闭连接。

在慢速网络下使用短连接，连接的开销会很大；在生产繁忙的系统中，连接也可能会受到系统端口数
限制，如果要每秒建立几千个连接，那么连接断开后，端口不会被马上回收利用，必须经历一个“FI
”阶段的等待，直到可被回收利用为止，这样就可能会导致端口资源不够用。在Linux上，可以通过
整/proc/sys/net/ipv4/ip_local_
ort_range来扩大端口的使用范围；调整/proc/sys/net/ipv4/tcp_
in_timeout来减少回收延期（如果想在应用服务器上调整这个参数，一定要慎重！）。

另外一个办法是主机使用多个IP地址。端口数的限制其实是基于同一个IP:PORT的，如果主机增加了I
，MySQL就可以监听多个IP地址，客户端也可以选择连接某个IP:PORT，这样就增加了端口资源。

2.长连接

长连接是指程序之间的连接在建立之后，就一直打开，被后续程序重用。使用长连接的初衷是减少连

的开销，尽管MySQL的连接比其他数据库要快得多。

以PHP程序为例，当收到一个永久连接的请求时，PHP将检查是否已经存在一个（前面已经开启了的相同的永久连接。如果存在，则将直接使用这个连接；如果不存在，则建立一个新的连接。所谓“相”的连接是指用相同的用户名和密码到相同主机的连接。

从客户端的角度来说，使用长连接有一个好处，可以不用每次创建新连接，若客户端对MySQL服务的连接请求很频繁，永久连接将更加高效。对于高并发业务，如果可能会碰到连接的冲击，推荐使用连接或连接池。

从服务器的角度来看，情况则略有不同，它可以节省创建连接的开销，但维持连接也是需要内存的。果滥用长连接的话，可能会使用过多的MySQL服务器连接。现代的操作系统可以拥有几千个MySQL接，但很有可能绝大部分都是睡眠（sleep）状态的，这样的工作方式不够高效，而且连接占据内存也会导致内存的浪费。

对于扩展性好的站点来说，其实大部分的访问并不需要连接数据库。如果用户需要频繁访问数据库，么可能会在流量增大的时候产生性能问题，此时长短连接都是无法解决问题的，所以应该进行合理的计和优化来避免性能问题。

如果客户端和MySQL数据库之间有连接池或Proxy代理，一般在客户端推荐使用短连接。对于长连接使用一定要慎重，不可滥用。如果没有每秒几百、上千的新连接请求，就不一定需要长连接，也无法长连接中得到太多好处。在Java语言中，由于有连接池，如果控制得当，则不会对数据库有较大的冲，但PHP的长连接可能导致数据库的连接数超过限制，或者占用过多的内存。

对此，研发工程师、系统运维工程师、DBA需要保持沟通，确定合理的连接策略，千万不要不假思索采用长连接。

3.连接池

由于一些数据库创建和销毁连接的开销很大，或者相对于所执行的具体数据操作，连接所耗的资源过，此时就可能需要添加连接池来改进性能。

数据库连接池是一些网络代理服务或应用服务器实现的特性，如J2EE服务器，它实现了一个持久连接“池”，允许其他程序、客户端来连接，这个连接池将被所有连接的客户端共享使用，连接池可以加连接，也可以减少数据库连接，降低数据库服务器的负载。

4.持久连接和连接池的区别

长连接是一些驱动、驱动框架、ORM工具的特性，由驱动来保持连接句柄的打开，以便后续的数据操作可以重用连接，从而减少数据库的连接开销。而连接池是应用服务器的组件，它可以通过参数来置连接数、连接检测、连接的生命周期等。

如果连接池或长连接使用的连接数很多，有可能会超过数据库实例的限制，那么就需要留意连接相关设置了，比如连接池的最小、最大连接数设置，以及php-fpm的进程个数等，否则程序将不能申请新连接。

三、MySQL服务器端如何管理客户端连接

MySQL连接管理器线程负责处理服务器侦听的网络接口上的客户端连接请求。连接管理器线程将每客户端连接与专用于它的线程相关联，以处理该连接的身份验证和请求处理。

连接管理线程在必要时创建一个新线程，但是会先尝试避免这样做，MySQL连接管理器线程首先会看线程缓存是否包含空闲可用于连接的线程，如果有，则从线程缓存中选取一个空闲的线程分配给客端，如果没有，则重新创建一个线程。当连接结束时，如果缓存未满，则其线程返回到线程缓存中，于下次使用。

在这种连接线□□程模型中，存在与当前连接的客户端一样多的线程，这在服务器工作负载高还必须以理大量的连接时具有一些缺点，例如，线程创建和处理变得昂贵。此外，每个线程都需要消耗服务器内核资源，如堆栈空间。所以为了适应大量的并发连接，每个线程的堆栈大小必须保持较小，否则会于太大导致消耗服务器大量内存的情况，也可能耗尽其他资源，调度开销也会变得很大。

为了控制和监视服务器如何管理客户端连接的线程，有几个系统和状态变量可以查看。

线程缓存由thread_cache_size系统
量确定其大小，默认值为0（无缓存），这将导致为每个新连接设置一个线程，并在连接终止时进行理。设置thread_cache_size为N，启用N个
活动连接线程被缓存。thread_cache_
ize可以在服务器启动时设置或在服务器运行时更□□改，连接线程在与其关联的客户端连接终止时变为活动。

通过MySQL状态变量Threads_cached和Threads_
reated可以监控高速缓存中的线程数和已经创建了多少个线程。另外你也可以通过max_connections量控制可以同时连接的最大客户端数。

但当线程堆栈太小时，会限制了服务器可以处理的SQL语句的复杂性，存储过程的递归深度和其他内消耗的操作。

四、 MySQL服务器相关的连接配置参数

1.skip-networking

开启该选项可以彻底关闭MySQL的TCP/IP连接方式，如果WEB服务器是以远程连接的方式访问MySQ数据库服务器则不要开启该选项！否则将无法正常连接！

2.skip-name-resolve

禁止MySQL对外部连接进行DNS解析，使用这一选项可以消除MySQL进行DNS解析的时间。但需要意，如果开启该选项，则所有远程主机连接授权都要使用IP地址方式，否则MySQL将无法正常处理连请求！

3.max_connections

max_connections是指MySql的最大连接数，如果服务器的并发连接请求量比较大，建议调高此值，增加并行连接数量，当然这建立在服务器能支撑的情况下，因为如果连接数越多，由于MySQL会为个连接提供连接缓冲区，就会开销越多的内存，所以要适当调整该值，不能盲目提高设值。

4.max_user_connections

max_user_connections是指每个数据库用户的最大连接针对某一个账号的所有
户端并行连接到MYSQL服务的最大并行连接数。简单说是指同一个账号能够同时连接到mysql服务的大连接数。设置为0表示不限制。

顺便介绍下show global status中的Max_used_
onnections:它是指从这次mysql服务启动到现在，同一时刻并行连接数的最大值。它不是指当前的连情况，而是一个比较值。如果在过去某一个时刻，MYSQL服务同时有1000个请求连接过来，而之后也没有出现这么大的并发请求时，则Max_used_
onnections=1000.请注意与show variables 里的max_
ser_connections的区别。默认为0表示无限大。

5. max_connect_errors

设置每个主机的连接请求异常中断的最大次数，当超过该次数，MySQL服务器将禁止host的连接请，直到mysql服务器重启或通过flush hosts命令清空此host的相关信息。

6.thread_handing

Mysql 服务中线程处理模式包括两种：

- no-threads(单线程处理，多用于debug)
- one- thread-per-connection（每个请求对应一个线程，目前被作为默认值)；

7.thread_cache_size

这个值表示可以重新利用保存在缓存中线程的数量,当断开连接时如果缓存中还有空间,那么客户端的程将被放到缓存中,如果线程重新被请求，那么请求将从缓存中读取,如果缓存中是空的或者是新的请，那么这个线程将被重新创建,如果有很多新的线程，增加这个值可以改善系统性能.因为当应用发起个对数据库的操作时，在整个应用中是一个不小的开销，从建立连接之初，CPU 要给它划分一定的thread stack，然后进行用户身份认证，建立上下文信息，最后请求完成，关闭连接，同时释放资源，可称的上是秒级的过程，在高并发的情况下，将给系统带来巨大的压力更不能保证性能。所以，采用线重用，减小这部分的消耗。通过比较 Connections 和 Threads_created 状态的变量，可以看到这个量的作用。

```
mysql> show status like 'thread%';
mysql> show status like 'thread%';
+-------------------+-------+
| Variable_name     | Value |
+-------------------+-------+
| Threads_cached    | 649   |  <—当前被缓存的空闲线程的数量
| Threads_connected | 93    | <—正在使用（处于连接状态）的线程
| Threads_created   | 742   |  <—服务启动以来，创建了多少个线程
| Threads_running   | 5     | <—正在忙的线程（正在查询数据，传输数据等等操作）
+-------------------+-------+```
查看开机起来数据库被连接了多少次?
```bash
mysql> show status like '%connections%';
mysql> show global status like '%connections%';
+---------------------------------+--------------------+
| Variable_name                   | Value              |
+---------------------------------+--------------------+
| Connection_errors_max_connections | 0                |
| Connections                     | 101395055          |   <–服务启动以来，历史连接数
| Max_used_connections          | 742                |
| Max_used_connections_time     | 2018-08-21 15:42:38 |
+---------------------------------+--------------------+
```

通过连接线程池的命中率来判断设置值是否合适？命中率超过90%以上,设定合理。(Connections - Threads_created) / Connections * 100 %

8. thread_stack

每个连接线程被创建时，MySQL给它分配的内存大小。当MySQL创建一个新的连接线程时，需要给分配一定大小的内存堆栈空间，以便存放客户端的请求的Query及自身的各种状态和处理信息。当然果遇到下面的错误提示就应该考虑增加这个值了。mysql-debug: Thread stack overrun 如:

```
java.sql.SQLException: Thread stack overrun: 5456 bytes used of a 131072 byte stack, and 128
00 bytes needed. Use 'mysqld --thread_stack=#' to specify a bigger stack.
```

9.Connect_Timeout
```

字面上看意思是连接超时，指的是MySQL连接过程中握手的超时时间，在5.0.52以后默认为10秒，前版本默认是5秒。

connect_timeout: The number of seconds that the mysqld server waits for a connect packet before responding with Bad handshake. The default value is 10 seconds as of MySQL 5.0.52 and 5 seconds before that

mysql的基本原理是有个监听线程循环接收请求，当有请求来时，创建线程（或者从线程池中取）来理这个请求。由于mysql连接采用TCP协议，那么之前势必是需要进行TCP三次握手的。TCP三次握成功之后，客户端会进入阻塞，等待服务端的消息。服务端这个时候会创建一个线程(或者从线程池中一个线程)来处理请求，主要验证部分包括host和用户名密码验证。host验证我们比较熟悉，因为在用rant命令授权用户的时候是有指定host的。用户名密码认证则是服务端先生成一个随机数发送给客户，客户端用该随机数和密码进行多次sha1加密后发送给服务端验证。如果通过，整个连接握手过程完。（具体握手过程后续找到资料再分析）

由此可见，整个连接握手可能会有各种可能出错。所以这个connect_timeout值就是指这个超时时间。可以简单测试下，运行下面的telnet命令会发现客户端会在10秒后超时返回。

```
$time telnet mysql\_ip\_addr port
$ time telnet 127.0.0.1 5051
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^\]'.
?
Connection closed by foreign
host.
real    0m5.005s #这里的5秒即mysql默认的连接超时
user    0m0.000s
sys 0m0.000s
```

Telnet未退出前通过show processlist查看各线程状态可见，当前该连接处于授权认证阶段，此时的户为“unauthenticated user”。

```
256 | unauthenticated user | localhost:60595 | NULL | Connect | NULL | Reading from net | NU
L
```

10.wait_timeout&interactive_timeout

等待超时，那mysql等什么呢？确切的说是mysql在等用户的请求(query)，如果发现一个线程已经sleep的时间超过wait_timeout了那么这个线程将被清理掉，从文档上来看wait_imeout和interactive_timeout都是指不活跃的连接超时时间，而interactive_imeout针对交互式连接，wait_timeout针对非交互式连接。MySQL接线程启动的时候wait_timeout会根据是交互模式还是非交互模式被设置为这个值中的一个。如果我们运行mysql -uroot -p命令登陆到mysql，wait_imeout就会被设置为interactive_timeout的值。如果我们在wait_tieout时间内没有进行任何操作，那么再次操作的时候就会提示超时，这是mysql client会重新连接。于两者的区别，请参考云小秘博客MySQL之wait_timeout和interactive_imeout参数

11. net_read_timeout

数据库从客户端读取数据超时时间。在终止读之前，从一个连接获得数据而等待的时间秒数；当服务在从客户端读取数据时，net_read_imeout控制何时超时。即客户端执行数据读取，等待多少秒仍未执行成功时自动断开连接。 可能的

因为网络异常或客户端/服务器端忙无法及时发送或接收处理包。

12. net_write_timeout

数据库往客户端写数据超时时间。和net_read_
imeout意义类似，在终止写之前，等待多少秒把block写到连接；当服务正在写数据到客户端时，net_
rite_timeout控制何时超时。一般在网络条件比较差的时，或者客户端处理每个数据块耗时比较
时，由于 net_write_timeout导致的连接中断很容易
生。

13. net_retry_count

如果MySQL服务端在读写数据时，出现连接中断，会重试net_
etry_count指定的次数。在系统FreeBSD中有效，Linux中只有在build的时候指定NO_
LARM参数时net_retry_count才会起作用。

connect_timeout在获取连接阶段（authenticate）起作用，interactive_
imeout和wait_timeout在连接空闲阶段（sleep）起作用，而net_
ead_timeout和net_write_timeout则是在连接繁忙阶段（query）起作
。

获取MySQL连接是多次握手的结果，除了用户名和密码的匹配校验外，还有IP->HOST->DNS->IP
证，任何一步都可能因为网络问题导致线程阻塞。为了防止线程浪费在不必要的校验等待上，超过co
nect_timeout的连接请求将会被拒绝。

即使没有网络问题，也不能允许客户端一直占用连接。对于保持sleep状态超过了wait_
imeout（或interactive_timeout，取决于CLIENT_INTERACTIVE标志）的客
端，MySQL会主动断开连接。

即使连接没有处于sleep状态，即客户端忙于计算或者存储数据，MySQL也选择了有条件的等待。在
据包的分发过程中，客户端可能来不及响应（发送、接收、或者处理数据包太慢）。为了保证连接不
浪费在无尽的等待中，MySQL也会选择有条件（net_
ead_timeout和net_write_t
meout）地主动断开连接。比如我在客户端用load data infile的方式导入很大的一个文件到数据库中
然后中途用iptables禁用掉mysql的3306端口，这个时候服务器端该连接状态是reading from net，
等待net_read_timeout后关闭该连接。同理，在程序里面查询一个很
的表时，在查询过程中同样禁用掉端口，制造网络不通的情况，这样该连接状态是writing to net，
后在net_write_timeout后关闭该连接。

五、 MySQL连接相关状态查看

查看MySQL状态命令:

```
1. show global status  # 输出所有记录的状态
2. show global status like '%xxxx%'  # 输出记录带有xxx关键字的状态
   如: show global status like   'thread%'
```

1. Aborted_clients

由于客户端没有正确关闭连接导致客户端终止而中断的连接数。 如，客户端和服务端建立连接好之后
等待wait_time时间后，客户端和服务器之间的连接自动退出，此时会发现Aborted_
lients计数器加1。

2. Aborted_connects

试图连接到MySQL服务器而失败的连接数。如:**telnet 127.0.0.1 3306**， 然后什么都不做， ctrl+]

quit退出，就会发现aborted_connects计数器加1。

3. connections

试图连接到(不管成功与否)MySQL服务器的连接数计数。

4. Bytes_received

从所有客户端接收到的字节数。

5. Bytes_sent

发送给所有客户端的字节数。

6. Threads_cached

代表当前此时此刻线程缓存中有多少空闲线程。

7.Threads_connected

代表当前已建立连接的数量，因为一个连接就需要一个线程，所以也可以看成当前被使用的线程数。

8. Threads_created

创建用来处理连接的线程数。如果Threads_created较大
则需要增加thread_cached_size的值。

9.Threads_running

代表当前激活的（非睡眠状态）线程数。

六、MySQL连接常见错误

1.ERROR 1045 (28000): Access denied for user 'testcon'@'10.24.236.231' (using password YES)

原因1: 可能是用户密码错误

原因2: 可能是用户错误

原因3: 可能是host错误

解决办法:

- 如果确认密码正确，检查mysql.user表里面的account信息(主要是user,host列)，确认连接的account符合user里面的匹配项
- 如果通过mysql.user account正常，可以尝试更改一下密码再进行测试
- 最快速简单的方法是重新创建一个账号，授予相关的权限

2.ERROR 2005 (HY000): Unknown MySQL server host 'com-mysql.coflodhn1n0y.us-west-1.rds.amazonaws.com' (110)

原因: 可能DNS解析异常

解决方法:通过ping来观察dns的解析，并检查系统dns配置。如果ping数据库域名无法获取IP，可以判断dns配置有问题。

3. ERROR 1449 (HY000): The user specified as a definer ('testcon'@'10.24.236.231') does

not exist 或者ERROR 1045 (28000): user not exist

原因: account不存在

解决办法 :检查mysql.user表里面的account信息(主要是user,host列)，确认连接的account符合user里面的匹配项

4.ERROR 2003 (HY000): Can't connect to MySQL server on 'test.mysql.rds.aliyuncs.com' (10)

原因1: 安全限制，不允许访问，比如防火墙或者云RDS的白名单

原因2: 端口错误，如MySQL运行端口不是3306原因3: mysql服务未正确运行

解决办法:

- 执行lsof -i:3306， 检查下3306端口的进程信息
- 执行ps -ef | grep mysql，检查下进程信息
- 如果前两步没有输出，可能mysql server未在主机正确运行
- 如果前两步有输出，说明MySQL正常运行，则检查一下防火墙配置规则，或者使用telnet host 3306
- 云数据库RDS，需检查RDS的白名单中是否包含客户端服务器IP

5. ERROR 1135 (HY000): Can't create a new thread (errno 11); if you are not out of available memory, you can consult the manual for a possible OS-dependent bug

原因1: 服务器操作系统limits.conf文件对max user processes做了限制

原因2: MySQL没有可用内存

解决办法:

- 通过命令 **ulimit -a**查看max user processes配置，如果较小，修改内核参数
- 增加MySQL内存

6. ERROR 1130 (HY000): Host '192.168.1.3' is not allowed to connect to this MySQL server

原因: mysql.user表里没有匹配的host名称

解决办法:检查mysql.user表里面的account信息(主要是user,host列)，确认连接的account符合user里面的匹配项

7.ERROR 1045 (HY000): #28000ip not in whitelist

原因: rds for mysql的提示，且访问模式为高安全模式才会出现，原因是ip地址没有在rds的白名单中

解决办法检查RDS的白名单中是否包含客户端服务器IP地址

8.ERROR 5 (HY000): Out of memory (Needed 260400 bytes)

原因: MySQL没有可用内存

解决办法:（先确认内存是否已经不足）：增加mysql的内存，如果是rds，可以考虑升级RDS的实例格

9. ERROR 1129 (HY000): Host '10.24.236.231' is blocked because of many connection errors; unblock with 'mysqladmin flush-hosts'

原因: account达到了MySQL服务器设置的max_connect_errors的值

解决办法:

系统命令行下执行mysqladmin flush-hosts或者mysql命令行里执行flush hosts

10. ERROR 1226 (42000): User 'testcon' has exceeded the 'max_user_connections' resource (current value: 2)

原因: 达到了该account设定的max_user_connections大小

解决办法:

重新使用grant设置用户的max_user_connections或者调整系统变量max_user_connections，如果没有权限，也可以删除账号重新创建

11. ERROR 1226 (42000): User 'testcon' has exceeded the 'max_connections_per_hour' resource (current value: 2)

原因: 达到了该account设定的max_connections_per_hour大小

解决办法:

重新grant该账号max_connections_per_hour为0，如果没有权限，也可以删除账号重新创建

12.ERROR 1040 (HY000): Too many connections

原因: 达到了mysql系统参数max_connections的限制

解决办法:

- 检查MySQL server的CPU,IO,内存等状态的变化，是否有明显的升高现象，如果有明显的升高，实的通过show processlist获取session信息，通过获取到的session信息分析cpu，io以及内存跑高的因，综合分析（是否是遇到了阻塞或者慢查询）kill掉相关会话来解决
- 检查MySQL server的CPU,IO,内存等状态的变化，是否有明显的升高现象，如果没有明显的升高，时的通过show processlist获取session信息，通过获取到的session信息找到会话来源(看下是否是sleep连接较多)，尝试调整来源主机的应用行为
- 如果无法手动干预，尝试调整MySQL的max_connections的值，如果是rds for mysql，需要升级例规格来提升连接数

参考资料

MySQL网络协议分析

MySQL初识-架构-安装-初始化-连接-管理工具-数据文件

mysql timeout各种超时的机制以及区别

MySQL性能优化之参数配置
mysql timeout知多少mysql查看connect_timeout设置
MySQL 各种超时参数的含义
mysql timeout各种超时的机制以及区别
MySQL连接错误的十二“坑”

更多内容请关注公众号

原文链接：[玩转 MySQL 之二]MySQL 连接机制浅析及运维