# [MySQL提示“too many connections”](http://blog.chinaunix.net/uid-24426415-id-77347.html)

为了防止发生too many connections时候无法登录的问题，mysql manual有如下的说明：

mysqld actually allows max\_connections+1 clients to connect. The extra connection is reserved for use by accounts that have the SUPER privilege. By granting the SUPER privilege to administrators and not to normal users (who should not need it), an administrator can connect to the server and use SHOW PROCESSLIST to diagnose problems even if the maximum number of unprivileged clients are connected.

因此, 必须只赋予root用户的SUPER权限，同时所有数据库连接的帐户不能赋予SUPER权限。应用程序不应使用root用户。

“too many connections”的解决办法

1、把max\_connections设置为比较大的值。max\_connections默认为100，一般设置到500～1000比较合适，最大是16384。

2、把**wait\_timeout设置为较小的值。wait\_timeout默认是28800八小时。一般设置到120~1000.**

mysql> set global **wait\_timeout**=120; （这种修改方式重启失效）

3、可能是多次insert,update操作没有关闭session,需要在spring里配置transaction支持。要关闭session 和 sessionFactory。

# 手动去释放一些没用的连接

登陆到MySQL的提示符下，数据show processlist这个命令，可以得到所以连接到这个服务器上的MySQL连接：

mysql> show  processlist;  
+---------+------+---------------------+---------+---------+------+-------+-------------------+  
| Id      | User | Host                | db      | Command | Time | State | Info              |  
+---------+------+---------------------+---------+---------+------+-------+-------------------+  
| 1180421 | ur   | 202.103.96.68:49754 | test1   | Sleep   |    1 |       | NULL              |  
| 1180427 | ur   | 202.103.96.68:55079 | test2   | Sleep   |    1 |       | NULL              |  
| 1180429 | ur   | 202.103.96.68:55187 | testdba | Sleep   |    0 |       | NULL              |  
| 1180431 | ur   | 202.103.96.68:55704 | testdba | Sleep   |    0 |       | NULL              |  
| 1180437 | ur   | 202.103.96.68:32825 | test1   | Sleep   |    1 |       | NULL              |  
| 1180469 | ur   | 202.103.96.68:58073 | testdba | Sleep   |    0 |       | NULL              |  
| 1180472 | ur   | 83.136.93.131:47613 | test2   | Sleep   |    8 |       | NULL              |  
| 1180475 | root | localhost           | NULL    | Query   |    0 | NULL  | show  PROCESSLIST |  
+---------+------+---------------------+---------+---------+------+-------+-------------------+  
8 rows in set (0.00 sec)  
  
mysql>

然后，你可以看到像上面这样的MySQL数据连接列表，而且每一个都会有一个进程ID号（在上表的第一列）。我们只要输入这样的命令：

mysql> kill 1180421;  
Query OK, 0 rows affected (0.00 sec)  
  
mysql>

其中1180421为你在进程列表里找到并且要杀掉的进程号。

# [修改max\_connections 参数的方法](http://blog.itpub.net/22664653/viewspace-720598/)

开发说应用不能连接数据库,查看数据库连接信息已经超过最大允许连接数

1 数据库系统允许的最大可连接数max\_connections。该参数是可以设置的。如果不设置,默认是100。最大是16384。

2 数据库当前的连接线程数threads\_connected。该参数是动态变化的。

如果 threads\_connected == max\_connections 时,数据库系统就不能提供更多的连接数了,这时,如果程序还想新建连接线程,数据库系统就会拒绝,结果如开发所说连接不了。如何修改max\_connections呢？

方法1 在会话中修改

mysql> show variables like '%max\_connections%';

+-----------------+-------+

| Variable\_name   | Value |

+-----------------+-------+

| max\_connections | 151 |

+-----------------+-------+

1 row in set (0.00 sec)

mysql> set global max\_connections=14000;

Query OK, 0 rows affected (0.01 sec)

mysql> show variables like '%max\_connections%';

+-----------------+-------+

| Variable\_name   | Value |

+-----------------+-------+

| max\_connections | 14000 |

+-----------------+-------+

1 row in set (0.00 sec)

方法2 在my.conf 参数中修改，不过需要重新启动mysql 服务！不推荐

# 修改[wait\_timeout](http://blog.csdn.net/wulantian/article/details/21461363)

wait\_timeout**默认是28800八小时，**过大有弊端，其体现就是MySQL里大量的SLEEP进程无法及时释放，拖累系统性能，不过也不能把这个指设置的过小，否则你可能会遭遇到“MySQL has gone away”之类的问题，通常来说，我觉得把wait\_timeout设置为10是个不错的选择，但某些情况下可能也会出问题，比如说有一个CRON脚本，其中两次SQL查询的间隔时间大于10秒的话，那么这个设置就有问题了（当然，这也不是不能解决的问题，你可以在程序里时不时mysql\_ping一下，以便服务器知道你还活着，重新计算wait\_timeout时间）：  
  
# vi /etc/my.cnf  
  
[mysqld]  
wait\_timeout=10  
  
# /etc/init.d/mysql restart  
  
不过这个方法太生硬了，线上服务重启无论如何都应该尽可能避免，看看如何在MySQL命令行里通过[SET](http://dev.mysql.com/doc/refman/5.0/en/set-option.html)来设置：  
  
mysql> set global wait\_timeout=10;  
mysql> show global variables like 'wait\_timeout';  
+----------------------------+-------+  
| Variable\_name              | Value |  
+----------------------------+-------+  
| wait\_timeout               | 10    |  
+----------------------------+-------+  
  
这里一个容易把人搞蒙的地方是如果查询时使用的是show variables的话，会发现设置好像并没有生效，这是因为单纯使用show variables的话就等同于使用的是show session variables，查询的是会话变量，只有使用show global variables，查询的才是全局变量。  
  
网络上很多人都抱怨说他们set global之后使用show variables查询没有发现改变，原因就在于混淆了会话变量和全局变量，如果仅仅想修改会话变量的话，可以使用类似set wait\_timeout=10;或者set session wait\_timeout=10;这样的语法。  
  
另一个值得注意的是会话变量wait\_timeout初始化的问题，这一点在手册里已经明确指出了，我就直接拷贝了：  
  
On thread startup, the session wait\_timeout value is initialized from the global wait\_timeout value or from the global interactive\_timeout value, depending on the type of client (as defined by the CLIENT\_INTERACTIVE connect option to mysql\_real\_connect()).  
  
MySQL大拿[Jeremy Zawodny](http://jeremy.zawodny.com/blog/)曾在他的文章[Fixing Poor MySQL Default Configuration Values](http://jeremy.zawodny.com/blog/archives/011421.html)里面列出了几个很恶心的MySQL缺省设置，不过没包含wait\_timeout，但我觉得它也应该算一个，每次新装MySQL后最好都记得修改它。

# [MYSQL四个timeout参数的含义](http://blog.chinaunix.net/uid-27038861-id-3521727.html)

4个timeout都是为了防止某一个链接长时间占用一数据库的描述符,又不干事情,或者网络有问题没有释放这个描述符,线程阻塞种种资源浪费的情况.我们要主动的"卡"掉对端!

首先看看这4个timeout在mysql5.1手册里的英文解释:

**connect\_timeout**   
The number of seconds that the mysqld server waits for a connect packet before responding with Bad handshake.   
**interactive\_timeout**   
The number of seconds the server waits for activity on an interactive connection before closing it.   
**wait\_timeout**   
The number of seconds the server waits for activity on a noninteractive connection before closing it.   
**net\_read\_timeout**   
The number of seconds to wait for more data from a connection before aborting the read.   
**net\_write\_timeout**   
The number of seconds to wait for a block to be written to a connection before aborting the write.

从意思上可以理解为

connect\_timeout在获取连接阶段（authenticate）起作用，

interactive\_timeout和wait\_timeout在连接空闲阶段（sleep）起作用，

net\_read\_timeout和net\_write\_timeout在连接繁忙阶段（query）起作用。

查看timeout参数的值：

mysql> show variables like '%timeout%';

# MySQL配置文件my.ini

注意：my.ini是在ProgramData这个目录下，而不是在安装目录下。

例如MySQL的安装目录是C:\Program Files\MySQL\MySQL Server 5.6， 则my.ini配置文件是在C:\ProgramData\MySQL\MySQL Server 5.6\my.ini 。ProgramData可能是隐藏的目录。

# [MYSQL用户管理](http://www.cnblogs.com/hencehong/archive/2013/03/19/mysql_user.html)

## 用户登录

格式： mysql -h主机地址 -u用户名 -p用户密码

mysql -h110.110.110.110 -uroot -p123

本地可以直接mysql –uroot -p

## 用户退出

exit、quit

## 添加用户

**mysql.user表保存的是用户的登录信息**

1. 直接添加无权限

insert into mysql.user (host,user,password) values('%','jifei',PASSWORD('jifei'));

1. 添加并赋权

grant select on 数据库.\* to '用户名'@'登录主机' identified by '密码'

## 用户权限

1. 添加权限

grant 权限 on 数据库.表 to '用户名'@'登录主机';

权限： select ,update,delete,insert(表数据)、create,alert,drop(表结构)、references(外键)、create temporary tables(创建临时表)、index(操作索引)、create view,show view(视图)、create routine,alert routine,execute(存储过程)、all,all privileges(所有权限)

数据库：数据库名或者\*(所有数据库)

表：表名或者\*(某数据库下所有表)

主机:主机名或者%(任何其他主机)

例：grant selec,insert,update,delete on \*.\* to 'jifei'@'%';

1. 撤销权限

revoke 权限 on 数据库.表 from '用户名'@'登录主机';//将to改为from

例：revoke all on \*.\* from ‘jifei’@’%’;

1. 查看权限

show grants;//自己

show grants for dba@localhost;//指定用户指定host

## 删除用户

delete from mysql.user where user='' and host='';

## 修改密码

update mysql.user set password=PASSWORD('111111') where user='root';

## 找回密码

1. 关闭mysql服务

killall -TERM mysqld

1. 修改配置文件

vi /etc/my.cnf

在[mysqld]的段中加上一句：skip-grant-tables

例如：

[mysqld]

datadir=/var/lib/mysql

socket=/var/lib/mysql/mysql.sock

skip-grant-tables

1. 重启mysqld

service mysqld restart

1. 登录

mysql -uroot -p

1. 修改密码

update mysql.user set password=PASSWORD('111111') where user='root';

flush privileges;//刷新权限

1. 修改配置文件

 vi /etc/my.cnf

去掉之前的改动

1. 重启服务
2. 设置远程用户

## 远程用户

①     限制在指定ip登录host为ip详情请看 添加权限

②     在任意远程ip登录host为%详情请看 添加权限

1. 远程访问

mysql -h110.110.110.110 -uroot -p123;//指定h为ip详情请看 用户登录

一些标准实例：

1. mysql.user表实例：一般来说，Host字段都使用ip来限制，而不是机器名（机器名可变，不是特别靠谱）

select Host, User from user;

| 172.17.% | dev |   
| 172.17.0.% | export |   
| 172.17.0.20 | demo |   
| 172.28.0.% | dev |   
| 192.168.% | dev |   
| 110.111.126.% | demo |   
| 110.111.126.103 | helper          |   
| 110.111.127.% | webnav    |   
| localhost | backup |   
| localhost | backupdata |   
| localhost | root |   
+-----------------+-----------------+

2. 授权实例：show grants for 'helper'@'110.111.127.%'

+--------------------------------------------------------------------------------------------------------------------------------------------------+  
| GRANT USAGE ON \*.\* TO 'helper'@'110.111.127.%' IDENTIFIED BY PASSWORD 'xxxxxxxxxxxxxxxxx' WITH MAX\_USER\_CONNECTIONS 200 |   
| GRANT ALL PRIVILEGES ON `helper\_online`.\* TO 'helper'@'110.111.127.%' |   
+--------------------------------------------------------------------------------------------------------------------------------------------------+