

MySQL

MySQL之启动选项

The terminal window shows two files:

- config.cnf**: A configuration file for the MySQL client. It contains a section named [client] with the following settings:

```
1 [client]
2 user = root
3 password =
4 host =
5
```
- connect-mysql.sh**: A shell script. Its content is:

```
1 #####
2 # File Name: connect-mysql.sh
3 # Author: sunxiuyang
4 # mail: sunxiuyang04@gmail.com
5 # Created Time: Fri 28 Sep 2018 04:31:28 PM CST
6 #####
7#!/bin/bash
8mysql --defaults-extra-file=config.cnf
9
```

启动选项简介

当我们使用 [mysql客户端](#) 可执行文件连接mysql服务器时，需要指定IP地址、用户名及密码等信息，这些信息就是mysql客户端程序启动时的选项，通过这些选项可以连接到具体的mysql服务端上。

对于mysql服务端，在启动时可以指定同时连入的客户端数量、客户端/服务单的[通信方式](#)、表的默认存储引擎、查询缓存的大小等信息，

一般这些信息具有自己的默认值，同时连入的客户端数量默认是151，表的默认存储引擎是InnoDB，程序启动时可以修改这些默认值，在启动时指定的设置项被称为启动选项(startup option)，启动选项决定了程序启动后的行为，

mysql安装bin目录下不论是客户端还是服务端的可执行程序，在启动的时候都可以在命令行后面填写对应的启动选项，也可以在配置文件中指定。

1. 命令行中指定启动选项

命令行中指定启动选项时，需要在选项前面增加--前缀，如果选项名是多个单词组成的，则需要用短划线连接(-或者_)。

启动选项的格式

--启动选项1[=值1] --启动选项2[=值2] ... --启动选项n[=值n]

禁用客户端/服务端网络通信

例如，禁止客户端使用TCP/IP与MySQL服务端进行通信的启动选项写法如下：

mysqld --skip-networking 或者 mysqld --skip_networking，前缀用--，多个单词使用了-或_，二者是等价的。

使用了上面的命令后，mysql客户端将无法通过网络与服务端进行通信。

指定存储引擎

MySQL默认使用InnoDB作为表的存储引擎，如果默认需要使用MyISAM，可以用如下命令：

```
mysqld --default-storage-engine=MyISAM
```

启动选项注意事项

1. 每个启动选项前面都需要加上--
2. 不需要值的启动选项，只需要--名字即可
3. 需要值的启动选项，需要在=号后面加上值，选项、=号、值，三者之间不能有空白字符
4. 每个命令执行文件都可以通过命令--help形式看到所支持的启动选项，例如 mysql --help(mysqld有些特殊，需要写成 mysqld --verbose --help)

选项的长形式和短形式

为了方便，启动选项一般都有一个单词的全写和一个字母的简写，绝大多数情况下，我们都会使用简写形式，常见的长短形式如下：

长形式	短形式	含义
--host	-h	主机名
--user	-u	用户名
--password	-p	密码
--port	-P	端口
--version	-V	版本信息

短形式的注意事项如下：

1. 短形式只需要一个短划线
2. 短形式，选项名和选项值可以没有间隙，也可以用空白字符隔开(-p比较特殊，必须和密码在一起)
3. 区分大小写，-p和-P含义不同，需要注意

2. 配置文件中指定启动选项

在命令行中，每次启动都需要带上启动选项，如果选项过多是一件比较麻烦的事情，因此可以将选项写入到硬盘文件中，这样每次程序启动的时候读取配置文件中的选项就可以了。

配置文件的路径

MySQL服务端启动时，会在多个路径下寻找配置文件，这些路径是固定的，在不同的操作系统下，寻找文件的路径有所不同。

windows下的路径

windows查找路径如下：

路径名	备注
%WINDIR%\my.ini,%WINDIR%\my.cnf	
C:\my.ini,C:\my.cnf	
BASEDIR\my.ini,BASEDIR\my.cnf	
defaults-extra-file	在命令行中指定的路径
%APPDATA%\MySQL.mylogin.cnf	登录路径选项(进客户端使用)

1. 在表格的前三个路径中，配置文件可以是.ini结尾，也可以是.cnf结尾
2. %WINDIR% 是Windows目录的位置，通常是 C:\WINDOWS，可以使用 echo %WINDIR% 查看
3. BASEDIR 是MySQL的安装目录，例如: C:\Program Files\MySQL\MySQL Server5.7
4. defaults-extra-file 指的是可以在 mysqld 启动时指定配置文件路径，例如 mysqld --defaults-extra-file=配置文件路径
5. %APPDATA% 表示 Windows 应用程序数据目录的值，可以用 echo %APPDATA% 的命令查看
6. .mylogin.cnf 用于存储客户端连接服务器的选项(host、user、password、port、socket等)，仅被客户端使用，并且其并不是一个纯文本文件，而是使用 mysql_config_editor 工具创建的加密文件，该工具在安装目录的bin目录下

Linux下的路径

路径名	备注
/etc/my.cnf	
/etc/mysql/my.cnf	
SYSCONFDIR/my.cnf	
\$MYSQL_HOME/my.cnf	特定于服务器的选项(仅限服务器)
defaults-extra-file	命令行指定的额外配置文件路径
~/.my.cnf	特定于用户的选项
~/.mylogin.cnf	特定于用户的登录路径选项(仅限客户端)

1. SYSCONFDIR 表示在使用 CMake 构建 MySQL 时使用 SYSCONFDIR 选项指定的目录
2. MYSQL_HOME 是一个环境变量，该环境变量由用户自己设置，该目录下可以放置启动MySQL服务端相关的启动选项(除 ./mylogin.cnf 和 \$MYSQL_HOME/my.cnf 两个文件，其他文件既可以存放服务端相关也可以存放客户端相关的启动选项)
3. 如果使用 mysqld_safe 启动MySQL服务端，此时并没有指定 MYSQL_HOME 环境变量，那么该值将被自动设置为 MySQL 的安装目录
4. ~ 符号在Linux系统下代表当前用户的用户目录，一个系统可以有多个用户，不同的用户可以配置自己的.my.cnf和.my.login.cnf
5. defaults-extra-file 与Windows中的描述一样，启动时的命令行选项
6. .mylogin.cnf与Windows中的描述也是一致

mysqld_safe程序启动服务端程序时，会调用mysqld，如果使用 mysqld_safe 命令时带的选项，mysqld_safe 处理不了，将交给 mysqld 处理，

例如 `mysqld_safe --skip-networking`，--skip-networking 将交给 mysqld 处理。

配置文件内容

配置文件中的启动项被划分为若干个组，每个组有一个组名，用中括号[]扩起来，如下：

```
1 [server]
2 (具体的启动选项...)
3 option1          #配置项1，该选项不需要选项值
4 option2 = value2 #配置项2，该选项需要选项值
5
6 [mysqld]
7 (具体的启动选项...)
8
9 [mysql_safe]
10 (具体的启动选项...)
11
12 [client]
13 (具体的启动选项...)
14
15 [mysql]
16 (具体的启动选项...)
17
18 [mysqladmin]
19 (具体的启动选项...)
20
```

1. 在配置文件中的，选项明必须是长形式，不可以写成短形式
2. 配置文件中的启动选项不可以加--前缀，并且每行只能有一个选项
3. =等号周围可以有空白字符，命令行中是不可以有的
4. 配置文件中，可以使用#号作为注释

不同的选项做提供给不同的程序使用，如果选项组的名称与程序相同，那么组中的选项将专门用于该程序。

1. [mysqld]和[mysql]组专门应用于mysqld服务器程序和mysql客户端程序
2. [server]组与[client]组较为特别，前者的启动项用于所有的服务端程序，后者的启动项用于所有的客户端程序

程序名	类别	能读取的组
mysqld	启动服务器	[mysqld]、[server]
mysql_safe	启动服务器	[mysqld]、[server]、[mysqld_safe]
mysql.server	启动服务器	[mysqld]、[server]、[mysql.server]
mysql	启动客户端	[mysql]、[client]
mysqladmin	启动客户端	[mysqladmin]、[client]
mysqldump	启动客户端	[mysqldump]、[client]

例如，linux系统下，在 /etc/mysql/my.cnf 中添加如下内容：

```

1 [server]
2 skip-networking
3 default-storage-engine=MyISAM
4 复制代码

```

随后使用 mysqld 命令启动服务端时，虽然命令行中没有添加选项，但是MySQL也会去上面的文件中找到配置文件中的server组下的选项，使其生效，当然如果上面两个选项如果写到[client]组中，则不会生效了。

选项指定版本号

选项后面可以加上一个 -版本号，借此可以让特定版本的mysqld程序才可以读取该配置，例如：

```

1 [mysqld-5.7]
2 选项.....
3 复制代码

```

只有mysql5.7服务端启动才能使用上面的选项。

配置文件优先级

MySQL根据路径的先后顺序读取配置文件，并且配置选项以后面的文件为准，即后面出现的选项会覆盖前面的选项，读取顺序见上文表格中的顺序。

同一配置文件多个组的优先级

与路径一样，相同的选项，后面出现的组会覆盖前面出现的组。

配置文件中的选项和命令行中的选项优先级

命令行中附带的选项会覆盖配置文件中的。

default-file

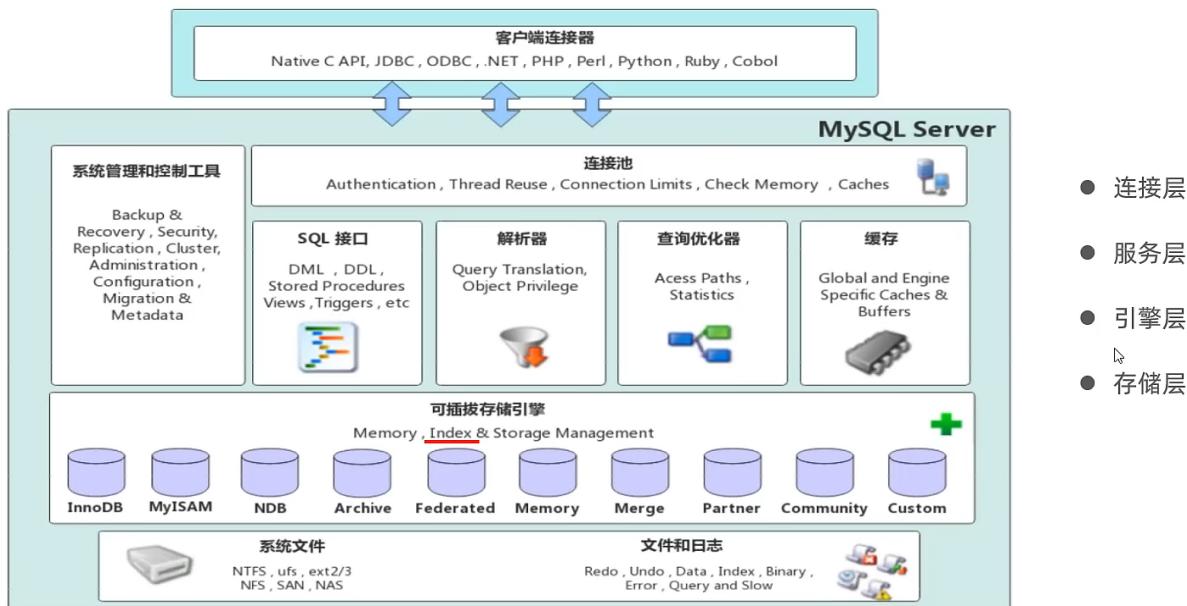
该选项可以在mysql服务端启动时指定，这样mysql就不会去上文表格中的路径中查找配置文件，而是使用该选项指定的配置文件，如果指定的文件不存在，将发生错误。

```
mysqld --default-file=/tmp/myconfig.txt
```

default-file与default-extra-file的区别是指定前者，mysql将不会去其他路径下查找配置文件，后者则会继续查找。

存储引擎

MySQL体系结构



- **连接层:** 最上层是一些客户端和链接服务，主要完成一些类似于连接处理、授权认证、及相关的安全方案。服务器也会为安全接入的每个客户端验证它所具有的操作权限。
- **服务层:** 第二层架构主要完成大多数的核心服务功能，如SQL接口，并完成缓存的查询，SQL的分析和优化，部分内置函数的执行。所有跨存储引擎的功能也在这一层实现，如 过程、函数等。
- **引擎层:** 存储引擎真正的负责了MySQL中数据的存储和提取，服务器通过AP和存储引擎进行通信。不同的存储引擎具有不同的功能，这样我们可以根据自己的需要，来选取合适的存储引擎。
- **存储层:** 主要是将数据存储在文件系统之上，并完成与存储引擎的交互。

存储引擎简介

存储引擎就是存储数据、建立索引、更新/查询数据等技术的实现方式。存储引擎是基于表而不是基于库的，所以存储引擎也可以被称为表类型。

默认存储引擎是InnoDB。

找个例子看一下：

```
#查询account表的建表语句
show create table account;
#查询结果
CREATE TABLE `account` (
  `id` int NOT NULL AUTO_INCREMENT COMMENT '主键ID',
  `name` varchar(10) COLLATE utf8mb4_general_ci DEFAULT NULL COMMENT '姓名',
  `money` int DEFAULT NULL COMMENT '余额',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci COMMENT='账户表';

ENGINE=InnoDB 可以看到未指定存储引擎，默认存储引擎是InnoDB
```

如果创建表时候想指定存储引擎，ENGINE=XXXX就可以了。

可以通过 `show engines;` 查一下支持哪些存储引擎。

存储引擎的特点

区别

特点	InnoDB	MyISAM	Memory
存储限制	64TB	有	有
事务安全	支持	-	-
锁机制	行锁	表锁	表锁
B+tree索引	支持	支持	支持
Hash索引	-	-	支持
全文索引	支持 (5.6版本之后)	支持	-
空间使用	高	低	N/A
内存使用	高	低	中等
批量插入速度	低	高	高
支持外键	支持	-	-

InnoDB

介绍: InnoDB 是一种兼顾高可靠性和高性能的通用存储引擎，在 MySQL 5.5 之后，InnoDB 是默认的 MySQL 引擎。

特点:

- DML 操作遵循 ACID 模型，支持**事务**
- 行级锁，提高并发访问性能
- 支持**外键约束**，保证数据的完整性和正确性

文件: xxx.ibd: xxx代表表名，InnoDB 引擎的每张表都会对应这样一个表空间文件，存储该表的表结构（frm、sdi）、数据和索引。

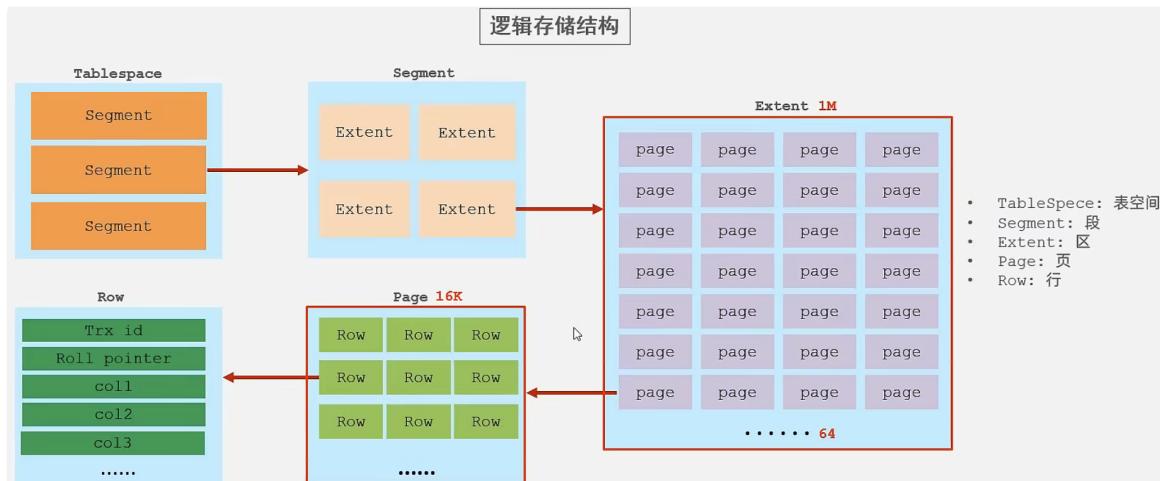
参数: innodb_file_per_table，决定多张表共享一个表空间还是每张表对应一个表空间

```
#查询表变量里的innodb_file_per_table开关
show variables like 'innodb_file_per_table';
查询结果是ON，打开的，打开就代表每张表对应一个表空间

我们可以打开mysql的数据存放目录
cd /var/lib/mysql/数据库名
看到“表名.ibd”
可以使用MySql8.0自带的工具ibd2sdi读取到ibd文件中的相关信息
如果mysql是docker安装的则先进入容器再找到指定文件: sudo docker exec -it 27e /bin/bash
```



逻辑存储结构:



MyISAM

介绍: MyISAM 是 MySQL 早期的默认存储引擎。

特点:

- 不支持事务，不支持外键
- 支持表锁，不支持行锁
- 访问速度快

文件:

- xxx.sdi: 存储表结构信息(文本文件，可以直接打开看)
 - xxx.MYD: 存储数据
 - xxx.MYI: 存储索引
- ```
#创建一张表观察一下试试
CREATE TABLE `myisam_table` (
 `id` int NOT NULL AUTO_INCREMENT COMMENT '主键ID',
 `name` varchar(10) COLLATE utf8mb4_general_ci DEFAULT NULL COMMENT '姓名',
 PRIMARY KEY (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci COMMENT='测试MyISAM'

#进入docker MySql容器
sudo docker exec -it 27e /bin/bash
```

```
#进入bian_test库文件夹下
cd /var/lib/mysql/bian_test
#可以看到如下三个文件
myisam_table.MYD myisam_table.MYI myisam_table_685.sdi
```

## Memory

**介绍:** Memory 引擎的表数据是存储在内存中的，受硬件问题、断电问题的影响，只能将这些表作为临时表或缓存使用。

**特点:**

- 存放在内存中，速度快
- hash索引 (默认)

**文件:**

- xxx.sdi: 存储表结构信息

```
#创建一张表观察一下试试
CREATE TABLE `memory_table` (
 `id` int NOT NULL AUTO_INCREMENT COMMENT '主键ID',
 `name` varchar(10) COLLATE utf8mb4_general_ci DEFAULT NULL COMMENT '姓名',
 PRIMARY KEY (`id`)
) ENGINE=Memory DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci COMMENT='测试Memory'

#进入docker MySql容器
sudo docker exec -it 27e /bin/bash

#进入bian_test库文件夹下
cd /var/lib/mysql/bian_test
#可以看到如下文件
memory_table_686.sdi
```

# 怎么选择存储引擎

在选择存储引擎时，应该根据应用系统的特点选择合适的存储引擎。对于复杂的应用系统，还可以根据实际情况选择多种存储引擎进行组合。

- InnoDB: 如果应用对事物的完整性有比较高的要求，在并发条件下要求数据的一致性，数据操作除了插入和查询之外，还包含很多的更新、删除操作，则 InnoDB 是比较合适的选择
- MyISAM: 如果应用是以读操作和插入操作为主，只有很少的更新和删除操作，并且对事务的完整性、并发性要求不高，那这个存储引擎是非常合适的。
- Memory: 将所有数据保存在内存中，访问速度快，通常用于临时表及缓存。Memory 的缺陷是对表的大小有限制，太大的表无法缓存在内存中，而且无法保障数据的安全性

电商中的足迹和评论适合使用 MyISAM 引擎，缓存适合使用 Memory 引擎。

## Linux-Ubuntu安装和配置MySQL

MySQL 是最常见的开源关系数据库管理系统 (RDBMS) 之一，它基于结构化查询语言 (SQL)，这是一种用于管理数据库中保存的数据的编程语言。

### 要求

你将需要在系统上拥有提升的权限 (root)。你可以通过执行以下命令来执行此操作。

```
sudo su
```

### 程序

本文中选择的 Ubuntu 版本是 Ubuntu 20.04 LTS，使用 MySQL Linux 版本是 5.7 版做示例。要开始安装，请按照以下步骤操作：

## 步骤 1)

确保系统中的所有软件包和存储库都是最新的。你可以通过运行以下命令来执行此操作：

```
sudo apt update
```

```
cyb@DESKTOP-UOIAA21:~$ sudo apt update
Get:1 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Hit:2 http://archive.ubuntu.com/ubuntu jammy InRelease
Get:3 http://archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Hit:4 http://archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:5 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [1502 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1059 kB]
Fetched 2790 kB in 4s (632 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
11 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

## 步骤 2)

现在，我们将通过 apt 包管理器安装 MySQL。执行下面的命令。

```
sudo apt install mysql-server
```

```

cyb@DESKTOP-UOIAA21:~$ sudo apt install mysql-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
 lib65-va-driver intel-media-va-driver libfa2-0.7.4 libfa1 libfaacs0 libfa3 libbass9 libasynccns0 libaudio2 libavcodec58 libavformat58 libavutil56
 libddplus0 libbluray2 libbs260 libcaca0 libcdio-cdda2 libcdio-paranoia2 libcdio19 libchromaprint1 libcodec2-1.0 libdav1d5 libdc0 libdv4
 libdvdnav libdvdread0 libegl-mesa0 libegl1 libenca0 libfaad2 libflac8 libgsm0 libigdmm12 libjack-jackd2-0 libldb2
 liblirc-client0 libmad0 libmfx1 libmngr2 libmp3lame0 libmpeg2-4 libmpg123-0 libnorm1 libogg0 libopenal-data libopenjp2-7 libopenmpt0
 libopus0 libpnm-5.3-0 libpostproc55 libpulse0 librabbithmq4 libsamplerate0 libssl1.2debian libsnappy1v5 libsndfile1
 libsndio7.0 libsoxr0 libspeex1 libsr1.4-gnutls libssh-gcrypt-4 libswresample3 libswscale5 libtalloc2 libtdb1 libtevent0 libtheora0 libtwolame0
 libudfread0 libva-drm2 libva-x11-2 libva2 libvpd1 libvorbis0a libvorbisenc2 libvorbisfile3 libvorbisidecl libvpx7 libwayland-server0
 libwbclient0 libwebpmux3 libx264-163 libx265-199 libxss1 libxvidcore4 libxvmc1 libzmq5 libzvbi-common libzvbi0 mesa-va-drivers mesa-vdpau-drivers
 ocl-icd-libopencl python3-lsb python3-talloc samba-libs va-driver-all vdpau-driver-all
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
 libai01 libcgi-fast-perl libfcgi-pm-perl libevent-pthreads-2.1-7 libfcgi-bin libfcgi-perl libfcgi0ldbl libhtml-template-perl libmecab2
 libprotobuf-lite23 mecab-ipadic mecab-ipadic-utf8 mecab-utils mysql-client-8.0 mysql-client-core-8.0 mysql-common mysql-server-8.0
 mysql-server-core-8.0
Suggested packages:
 libipc-sharedcache-perl mailx tinyca
The following NEW packages will be installed:
 libai01 libcgi-fast-perl libfcgi-pm-perl libevent-pthreads-2.1-7 libfcgi-bin libfcgi-perl libfcgi0ldbl libhtml-template-perl libmecab2
 libprotobuf-lite23 mecab-ipadic mecab-ipadic-utf8 mecab-utils mysql-client-8.0 mysql-client-core-8.0 mysql-common mysql-server mysql-server-8.0
 mysql-server-core-8.0
0 upgraded, 19 newly installed, 0 to remove and 11 not upgraded.
Need to get 29.1 MB of archives.
After this operation, 242 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://archive.ubuntu.com/ubuntu jammy/main amd64 mysql-common all 5.8+1.0.8 [7212 B]
Get:2 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 mysql-client-core-8.0 amd64 8.0.36-0ubuntu0.22.04.1 [2692 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 mysql-client-8.0 amd64 8.0.36-0ubuntu0.22.04.1 [22.7 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy/main amd64 libai01 amd64 0.3.112-13build1 [7176 B]
Get:5 http://archive.ubuntu.com/ubuntu jammy/main amd64 libevent-pthreads-2.1-7 amd64 2.1.12-stable-1build3 [7642 B]
Get:6 http://archive.ubuntu.com/ubuntu jammy/main amd64 libmecab2 amd64 0.996-14build9 [199 kB]
Get:7 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libprotobuf-lite23 amd64 3.12.4-1ubuntu7.22.04.1 [209 kB]
Get:8 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 mysql-server-core-8.0 amd64 8.0.36-0ubuntu0.22.04.1 [17.5 MB]
Get:9 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 mysql-server-8.0 amd64 8.0.36-0ubuntu0.22.04.1 [1437 kB]
Get:10 http://archive.ubuntu.com/ubuntu jammy/main amd64 libcgi-pm-perl all 4.54-1 [188 kB]
Get:11 http://archive.ubuntu.com/ubuntu jammy/main amd64 libfcgioldbl amd64 2.4.2-2build2 [28.0 kB]
Get:12 http://archive.ubuntu.com/ubuntu jammy/main amd64 libfcgi-perl amd64 0.82+ds-1build1 [22.8 kB]
Get:13 http://archive.ubuntu.com/ubuntu jammy/main amd64 libfcgi-fast-perl all 1.2.15-1 [10.5 kB]
Get:14 http://archive.ubuntu.com/ubuntu jammy/main amd64 libfcgi-bin amd64 2.4.2-2build2 [11.2 kB]
Get:15 http://archive.ubuntu.com/ubuntu jammy/main amd64 libhtml-template-perl all 2.97-1.1 [59.1 kB]
Get:16 http://archive.ubuntu.com/ubuntu jammy/main amd64 mecab-utils amd64 0.996-14build9 [4850 B]
Get:17 http://archive.ubuntu.com/ubuntu jammy/main amd64 mecab-ipadic all 2.7.0-20070801+main-3 [6718 kB]
Get:18 http://archive.ubuntu.com/ubuntu jammy/main amd64 mecab-ipadic-utf8 all 2.7.0-20070801+main-3 [4384 B]
Get:19 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 mysql-server all 8.0.36-0ubuntu0.22.04.1 [9460 B]
Fetched 29.1 MB in 7s (3948 kB/s)
Preconfiguring packages ...
Selecting previously unselected package mysql-common.
(Reading database ... 57269 files and directories currently installed.)
Preparing to unpack .../0=mysql-common_5.8+1.0.8_all.deb ...
Unpacking mysql-common (5.8+1.0.8) ...
Selecting previously unselected package mysql-client-core-8.0.
Preparing to unpack .../1=mysql-client-core-8.0_8.0.36-0ubuntu0.22.04.1_amd64.deb ...
Unpacking mysql-client-core-8.0 (8.0.36-0ubuntu0.22.04.1) ...
Selecting previously unselected package libai01:amd64.
Preparing to unpack .../3-libai01_0.3.112-13build1_amd64.deb ...
Unpacking libai01:amd64 (0.3.112-13build1) ...
Selecting previously unselected package libevent-pthreads-2.1-7:amd64.
Preparing to unpack .../4-libevent-pthreads-2.1-7_2.1.12-stable-1build3_amd64.deb ...
Unpacking libevent-pthreads-2.1-7:amd64 (2.1.12-stable-1build3) ...
Selecting previously unselected package libmecab2:amd64.
Preparing to unpack .../5-libmecab2_0.996-14build9_amd64.deb ...
Unpacking libmecab2:amd64 (0.996-14build9) ...
Selecting previously unselected package libprotobuf-lite23:amd64.
Preparing to unpack .../6-libprotobuf-lite23_3.12.4-1ubuntu7.22.04.1_amd64.deb ...
Unpacking libprotobuf-lite23:amd64 (3.12.4-1ubuntu7.22.04.1) ...
Selecting previously unselected package mysql-server-core-8.0.
Preparing to unpack .../7=mysql-server-core-8.0_8.0.36-0ubuntu0.22.04.1_amd64.deb ...
Unpacking mysql-server-core-8.0 (8.0.36-0ubuntu0.22.04.1) ...
Setting up mysql-common (5.8+1.0.8) ...
update-alternatives: using /etc/mysql/my.cnf.fallback to provide /etc/mysql/my.cnf (my.cnf) in auto mode
Selecting previously unselected package mysql-server-8.0.
(Reading database ... 57488 files and directories currently installed.)
Preparing to unpack .../00=mysql-server-8.0_8.0.36-0ubuntu0.22.04.1_amd64.deb ...
Unpacking mysql-server-8.0 (8.0.36-0ubuntu0.22.04.1) ...
Selecting previously unselected package libcgi-pm-perl.
Preparing to unpack .../01-libcgi-pm-perl_4.54-1_all.deb ...

```

## 步骤 3)

安装成功后，mysql-service 应该会自动启动。你可以通过执行以下命令来确认：

```
sudo systemctl mysql-server
```

你应该得到与下图类似的输出。

```
cyb@DESKTOP-UOIAA21:~$ sudo mysql_secure_installation
Securing the MySQL server deployment.

Connecting to MySQL using a blank password.

VALIDATE PASSWORD COMPONENT can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD component?

Press y|Y for Yes, any other key for No:

Skipping password set for root as authentication with auth_socket is used by default.
If you would like to use password authentication instead, this can be done with the "ALTER_USER" command.
See https://dev.mysql.com/doc/refman/8.0/en/alter-user.html#alter-user-password-management for more information.

By default, a MySQL installation has an anonymous user,
allowing anyone to log into MySQL without having to have
a user account created for them. This is intended only for
testing, and to make the installation go a bit smoother.
You should remove them before moving into a production
environment.

Remove anonymous users? (Press y|Y for Yes, any other key for No) : Y
Success.

Normally, root should only be allowed to connect from
'localhost'. This ensures that someone cannot guess at
the root password from the network.

Disallow root login remotely? (Press y|Y for Yes, any other key for No) : Y
Success.

By default, MySQL comes with a database named 'test' that
anyone can access. This is also intended only for testing,
and should be removed before moving into a production
environment.

Remove test database and access to it? (Press y|Y for Yes, any other key for No) : Y
- Dropping test database...
Success.

- Removing privileges on test database...
Success.

Reloading the privilege tables will ensure that all changes
made so far will take effect immediately.

Reload privilege tables now? (Press y|Y for Yes, any other key for No) : Y
Success.

All done!
```

在服务未运行的任何情况下，请执行以下命令：

```
sudo /etc/init.d/mysql start
```

## 验证 MySQL 安装

你可以通过运行以下命令来验证安装结果，该命令将输出系统中所安装的 MySQL 版本和发行版。

```
mysql --version
```

```
cyb@DESKTOP-UOIAA21:~$ mysql --version
mysql Ver 8.0.36-0ubuntu0.22.04.1 for Linux on x86_64 ((Ubuntu))
```

## 保护MySQL数据库

既然 MySQL 数据库安装成功了，需要设置一些参数来保证以后配置的服务器和数据库的安全。

在其他情况下，MySQL 数据库包安装完成后，mysql-secure-installation 实用程序将自动启动。但是，如果你没有自动启动，可执行以下命令：

```
sudo mysql_secure_installation
```

你将看到一个提示，询问你是否验证密码插件。它通过检查用户密码的强度来增强 MySQL 数据库的安全性，允许用户仅设置强密码。按 Y 接受 VALIDATION 或按 RETURN 键跳过。

```
cyb@DESKTOP-UOIAA21:~$ sudo mysql_secure_installation
Securing the MySQL server deployment.

Connecting to MySQL using a blank password.

VALIDATE PASSWORD COMPONENT can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD component?

Press y|Y for Yes, any other key for No:

Skipping password set for root as authentication with auth_socket is used by default.
If you would like to use password authentication instead, this can be done with the "ALTER_USER" command.
See https://dev.mysql.com/doc/refman/8.0/en/alter-user.html#alter-user-password-management for more information.

By default, a MySQL installation has an anonymous user,
allowing anyone to log into MySQL without having to have
a user account created for them. This is intended only for
testing, and to make the installation go a bit smoother.
You should remove them before moving into a production
environment.

Remove anonymous users? (Press y|Y for Yes, any other key for No) : Y
Success.

Normally, root should only be allowed to connect from
'localhost'. This ensures that someone cannot guess at
the root password from the network.

Disallow root login remotely? (Press y|Y for Yes, any other key for No) : Y
Success.

By default, MySQL comes with a database named 'test' that
anyone can access. This is also intended only for testing,
and should be removed before moving into a production
environment.

Remove test database and access to it? (Press y|Y for Yes, any other key for No) : Y
 - Dropping test database...
Success.

 - Removing privileges on test database...
Success.

Reloading the privilege tables will ensure that all changes
made so far will take effect immediately.

Reload privilege tables now? (Press y|Y for Yes, any other key for No) : Y
Success.

All done!
```

接下来，会看到设置 root 密码的提示。输入密码并按回车键。注意，出于安全原因，在控制台中键入的任何内容都不会显示。

接下来，会看到一个提示，询问你是否删除所有匿名用户，输入 Y 表示是。对于此处的任何其他提示，输入 Y 表示是。

# 以root身份登录并调整用户身份验证

MySQL 数据库带有一个客户端实用程序，允许你从 Linux 终端访问数据库并与之交互。

通常，在未执行任何配置的情况下在 Ubuntu 上全新安装 MySQL 后，访问服务器的用户将使用身份验证套接字 (auth\_socket) 插件进行身份验证。

auth\_socket 的使用会阻碍服务器使用密码对用户进行身份验证。它不仅会引起安全问题，而且还会使用户无法使用外部程序（如 phpMyAdmin）访问数据库。我们需要将身份验证方法从 auth\_socket 更改为使用 mysql\_native\_password。

为此需要打开 MySQL 控制台，并在 Linux 终端上运行以下命令。

```
sudo mysql
```

```
cyb@DESKTOP-U0IAA21:~$ sudo mysql
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.36-0ubuntu0.22.04.1 (Ubuntu)

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

现在，我们需要检查数据库对不同用户使用的身份验证方法。你可以通过运行以下命令来执行此操作。

```
SELECT user,authentication_string,plugin,host FROM
mysql.user;
```

```
mysql> SELECT user,authentication_string,plugin,host FROM mysql.user;
+-----+-----+-----+-----+
| user | authentication_string | plugin | host |
+-----+-----+-----+-----+
X BK YL~mK0d34C/G5GK7ns100N9HrfgSz00xNVC91Kj0szk4YB | caching_sha2_password | localhost |
mysql.infoschema	A005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBEUSED	caching_sha2_password	localhost
mysql.session	A005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBEUSED	caching_sha2_password	localhost
mysql.sys	A005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBEUSED	caching_sha2_password	localhost
root		auth_socket	localhost
+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

从上图中，我们可以确认 root 用户确实使用 auth\_socket 插件进行了身份验证。我们需要使用下面的“ALTER USER”命令切换到“密码验证”的使用。确保使用安全密码（应超过 8 个字符，结合数字、字符串和特殊符号），因为它将替换你在执行上述命令“sudo mysql\_secure\_installation”时设置的密码。运行以下命令。

```
ALTER USER 'root'@'localhost' IDENTIFIED WITH
mysql_native_password BY 'your_password';
```

```
mysql> ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY '19990208cyb';
Query OK, 0 rows affected (0.01 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.00 sec)
```

注意，上图中突出显示的文本是你输入安全密码的地方。将它包含在单个标签之间。现在，我们需要重新加载授权表并将更改更新到 MySQL 数据库。通过执行以下命令来执行此操作。刷新权限：

```
FLUSH PRIVILEGES;
```

完成后，我们需要确认 root 用户不再使用 auth\_socket 进行身份验证。通过再次运行以下命令来执行此操作。

```
SELECT user,authentication_string,plugin,host FROM
mysql.user;
```

```
mysql> SELECT user,authentication_string,plugin,host FROM mysql.user;
+-----+-----+-----+-----+
| user | authentication_string | plugin | host |
+-----+-----+-----+-----+
X BK YL~mk0d34yC/G5GK7ns100N9hHrfGSz00xNVC91KjoszK4YB	caching_sha2_password	localhost	
mysql.infoschema	A005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBEUSED	caching_sha2_password	localhost
mysql.session	A005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBEUSED	caching_sha2_password	localhost
mysql.sys	A005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBEUSED	caching_sha2_password	localhost
root	*0896E53F5F118C1DB87F53B4257B0D602488985E	mysql_native_password	localhost
+-----+-----+-----+-----+
5 rows in set (0.00 sec)

mysql> exit
Bye
```

从上图中，我们看到 root 身份验证方法已从“auth\_socket”更改为“password”。

由于我们更改了 root 的身份验证方法，因此我们无法使用之前使用的相同命令打开 MySQL 控制台。即“sudo mysql”。我们需要包括用户名和密码参数，如下所示。

```
mysql -u root -p
```

“-u”表示用户，在我们的例子中是“root”，“-p”代表“password”，一旦你按下 Enter 键，服务器就会提示你输入。

```
cyb@DESKTOP-UOIAA21:~$ sudo mysql
ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: NO)
cyb@DESKTOP-UOIAA21:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 8.0.36-0ubuntu0.22.04.1 (Ubuntu)

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

## 创建新用户

一切都设置好后，你可以创建一个新用户，你将授予该用户适当的权限。我们将创建一个用户 'PyDataStudio' 并分配对所有数据库表的权限以及更改、删除和添加用户权限的权限。逐行执行下面的命令。

```
CREATE USER 'PyDataStudio'@'localhost' IDENTIFIED BY
'strong_password';

GRANT ALL PRIVILEGES ON *.* TO 'PyDataStudio'@'localhost'
WITH GRANT OPTION;
```

第一个命令将创建新用户，第二个命令分配所需的权限。

```
mysql> create user 'cyb'@'localhost' identified by '123';
Query OK, 0 rows affected (0.01 sec)

mysql> grant all privileges on *.* to 'cyb'@'localhost' with grant option;
Query OK, 0 rows affected (0.02 sec)

mysql> exit
Bye
```

## 一键建用户加赋权官方已经弃用了

我们现在可以通过运行以下命令来测试我们的新用户。

```
mysql -u PyDataStudio -p
```

```
cyb@DESKTOP-UOIAA21:~$ mysql -u cyb -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 17
Server version: 8.0.36-0ubuntu0.22.04.1 (Ubuntu)

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

# 在 Ubuntu 上安装 MySQL-Server

在 Ubuntu 服务器上安装 MySQL-server 与上述步骤没有太大区别。但是，由于服务器是远程访问的，我们还需要为服务器启用远程访问。

要安装数据库并配置安全选项，只需在终端上逐行运行以下命令。

```
sudo apt update
sudo apt install mysql-server
sudo mysql_secure_installation
```

```
cyb@DESKTOP-U0IAA21:~$ sudo apt update
[sudo] password for cyb:
Sorry, try again.
[sudo] password for cyb:
Hit:1 http://archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Hit:4 http://archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:5 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [1502 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1059 kB]
Fetched 2790 kB in 5s (586 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
11 packages can be upgraded. Run 'apt list --upgradable' to see them.
cyb@DESKTOP-U0IAA21:~$ sudo apt install mysql-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
mysql-server is already the newest version (8.0.36-0ubuntu0.22.04.1).
The following packages were automatically installed and are no longer required:
i965-v-a-driver intel-media-va-driver libfa2-0.7.4 libfaa1 libfaacs0 libfaam3 libfaass9 libfauncs0 libfaudio2 libfawcodec58 libfawformat58 libfawutil56
libfdpplus0 libfburay2 libfs2b0 libfcaca0 libfdio-cdda2 libfdio-paranoia2 libfdio19 libfchromaprint1 libfcodec2-1.0 libfdavid5 libfdca0 libfdv4
libfdvdnav4 libfdvdread1 libfegl-mesa0 libfegl1 libfenc0 libfaad2 libflac8 libfgbm1 libfsm1 libfildgmm12 libfjack-jackd2-0 libfdbs2
libflirc-client0 libfdm0 libfdmx1 libfim3e0 libfmpeg2 libfmpeg3ame0 libfmpeg2-0 libfmppg123-0 libfnorm1 libfogg0 libfopenal-data libfopenjp2-7 libfopenmp0
libfopus0 libfpgm-5.3-0 libfpostproc55 libfpulse0 libfribbitm4 libfamplerate0 libfsl1.2debian libfshine3 libfmbclient libfnappy1v5 libfndfile1
libfndnfo7.0 libfsoxr1 libfspex1 libfsrt1.4-gnutls libfssh-gcrypt-4 libfresample3 libfwscale5 libfalloc2 libfdb1 libfevent0 libftheora0 libftwolame0
libfudfread0 libfva-drm2 libfva-x11-2 libfva2 libfdpaul libfvorbis0a libfvorbisenc2 libfvorbisfile3 libfvorbisidec1 libfpvx7 libfwayland-server0
libfwbcclient0 libfwbpmux3 libfx264-163 libfx265-199 libfxss1 libfxvidcore4 libfxvnc1 libfzmq5 libfzbvi-common libfzbvi0 mesa-va-drivers mesa-vdpau-drivers
ocl-icd-libfopencl python3-lldb python3-talloc samba-libs va-driver-all vdpau-driver-all
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 11 not upgraded.
cyb@DESKTOP-U0IAA21:~$ sudo mysql_secure_installation

Securing the MySQL server deployment.

Enter password for user root:

VALIDATE PASSWORD COMPONENT can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD component?

Press y|Y for Yes, any other key for No:
Using existing password for root.
Change the password for root ? ((Press y|Y for Yes, any other key for No) : No

... skipping.
By default, a MySQL installation has an anonymous user,
allowing anyone to log into MySQL without having to have
a user account created for them. This is intended only for
testing, and to make the installation go a bit smoother.
You should remove them before moving into a production
environment.

Remove anonymous users? (Press y|Y for Yes, any other key for No) : Y
Success.

Normally, root should only be allowed to connect from
'localhost'. This ensures that someone cannot guess at
the root password from the network.

Disallow root login remotely? (Press y|Y for Yes, any other key for No) : Y
Success.

By default, MySQL comes with a database named 'test' that
anyone can access. This is also intended only for testing,
and should be removed before moving into a production
environment.

Remove test database and access to it? (Press y|Y for Yes, any other key for No) : Y
- Dropping test database...
Success.

- Removing privileges on test database...
Success.

Reloading the privilege tables will ensure that all changes
made so far will take effect immediately.

Reload privilege tables now? (Press y|Y for Yes, any other key for No) : Y
Success.

All done!
```

安装成功后，需要启用远程访问。从逻辑上讲，我们需要在 Ubuntu 服务器防火墙上打开一个端口，以便 MySQL 数据库进行通信。默认情况下，MySQL 服务在 3306 端口上运行。执行以下命令。启用远程访问：

```
sudo ufw enable
sudo ufw allow mysql
```

```
cyb@DESKTOP-UOIAA21:~$ sudo ufw enable
Firewall is active and enabled on system startup
cyb@DESKTOP-UOIAA21:~$ sudo ufw allow mysql
Rule added
Rule added (v6)
```

为了增强 MySQL 数据库的可靠性和可访问性，可以将 MySQL-server 服务配置为在启动时开始运行。执行以下命令。在启动时启用 MySQL Server：

```
sudo systemctl enable mysql
```

```
cyb@DESKTOP-UOIAA21:~$ sudo systemctl enable mysql
Synchronizing state of mysql.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable mysql
```

现在需要配置服务器的接口，从而服务器能够侦听远程可访问的接口。我们需要编辑“mysqld.cnf”文件。运行以下命令。

```
sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf
```

mysql配置文件已有：

```
cyb@DESKTOP-UOIAA21:/etc/mysql/mysql.conf.d$ ls
mysql.cnf mysqld.cnf
```

mysqld.cnf 配置文件内容：

```

The MySQL database server configuration file.

One can use all long options that the program supports.
Run program with --help to get a list of available options and with
--print-defaults to see which it would actually understand and use.

For explanations see
http://dev.mysql.com/doc/mysql/en/server-system-variables.html

Here is entries for some specific programs
The following values assume you have at least 32M ram

[mysqld]

* Basic Settings

user = mysql
pid-file = /var/run/mysqld/mysqld.pid
socket = /var/run/mysqld/mysqld.sock
port = 3306
datadir = /var/lib/mysql

If MySQL is running as a replication slave, this should be
changed. Ref https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#
tmpdir = /tmp

Instead of skip-networking the default is now to listen only on
localhost which is more compatible and is not less secure.
bind-address = 127.0.0.1
mysqlx-bind-address = 127.0.0.1

* Fine Tuning

key_buffer_size = 16M
max_allowed_packet = 64M
thread_stack = 256K

thread_cache_size = -1

This replaces the startup script and checks MyISAM tables if needed
the first time they are touched
myisam-recover-options = BACKUP

max_connections = 151
table_open_cache = 4000

* Logging and Replication

Both location gets rotated by the cronjob.

Log all queries
Be aware that this log type is a performance killer.
general_log_file = /var/log/mysql/query.log
general_log = 1

Error log - should be very few entries.

log_error = /var/log/mysql/error.log

Here you can see queries with especially long duration
slow_query_log = 1
slow_query_log_file = /var/log/mysql/mysql-slow.log
long_query_time = 2
log-queries-not-using-indexes

The following can be used as easy to replay backup logs or for replication.
note: if you are setting up a replication slave, see README.Debian about
other settings you may need to change.
server-id = 1
log_bin = /var/log/mysql/mysql-bin.log
binlog_expire_logs_seconds = 2592000
```

```
binlog_expire_logs_seconds = 2592000
5 max_binlog_size = 100M
7 # binlog_do_db = include_database_name
8 # binlog_ignore_db = include_database_name
```

现在需要配置服务器的接口，从而服务器能够侦听远程可访问的接口。我们需要编辑“mysqld.cnf”文件。运行以下命令。

```
sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf
```

```
Instead of skip-networking the default is now to listen only on
localhost which is more compatible and is not less secure.
bind-address = 127.0.0.1
#
* Fine Tuning
#
```

配置绑定地址

默认情况下，绑定地址为“127.0.0.1”。为公网接口添加绑定地址，为服务网络接口添加另一个绑定地址。你可以将所有IP地址的绑定地址配置为“0.0.0.0”。

## 简明教程 | Linux中UFW的使用



小蘑菇糖糖

web三脚猫 / 萌新摄影 / Ingress Res

59 人赞同了该文章

这几天部署了一些服务，有用到UFW来管理防火墙规则，十分好用，下面大致记录一下其安装及使用方法，方便后续可以快速应用。

UFW，或称Uncomplicated Firewall，是iptables的一个接口，为不熟悉防火墙概念的初学者提供了易于使用的界面，同时支持IPv4和IPv6，广受欢迎。

### 先决条件

- 任意一台安装有Linux发行版的主机
- root权限（以下命令默认在root账户下运行，如果是其他账户，使用sudo也是一样的）

Systemctl是一个systemd工具，主要负责控制systemd系统和服务管理器。

Systemd是一个系统管理 [守护进程](#)、工具和库的集合，用于取代System V初始进程。Systemd的功能是用于集中管理和配置类UNIX系统。

在Linux生态系统中，Systemd被部署到了大多数的标准 [Linux发行版](#) 中，只有为数不多的几个发行版尚未部署。Systemd通常是所有其它守护进程的父进程，但并非总是如此。

本文旨在阐明在运行systemd的系统上“如何控制系统和服务”。



## 一、**systemctl**理解

Linux 服务管理两种方式service和systemctl

systemd是Linux系统最新的初始化系统(init),作用是提高系统的启动速度，尽可能启动较少的进程，尽可能更多进程并发启动。

systemd对应的进程管理命令是systemctl

### 1. systemctl命令兼容了service

即systemctl也会去/etc/init.d目录下，查看，执行相关程序

1.  
    systemctl redis start
- 2.
3.  
    systemctl redis stop
- 4.
5.  
    # 开机自启动
- 6.
7.  
    systemctl enable redis

## 2.管理服务(unit)

systemctl 提供了一组子命令来管理单个的 unit，其命令格式为：

```
systemctl [command] [unit]
```

command 主要有：

start：立刻启动后面接的 unit。

stop：立刻关闭后面接的 unit。

restart：立刻关闭后启动后面接的 unit，亦即执行 stop 再 start 的意思。

reload：不关闭 unit 的情况下，重新载入配置文件，让设置生效。

enable：设置下次开机时，后面接的 unit 会被启动。

disable：设置下次开机时，后面接的 unit 不会被启动。

status：目前后面接的这个 unit 的状态，会列出有没有正在执行、开机时是否启动等信息。

is-active：目前有没有正在运行中。

is-enable：开机时有没有默认要启用这个 unit。

kill：不要被 kill 这个名字吓着了，它其实是向运行 unit 的进程发送信号。

show：列出 unit 的配置。

mask：注销 unit，注销后你就无法启动这个 unit 了。

unmask：取消对 unit 的注销。

# MySQL语法

## 创建、删除、查询数据库

### 创建数据库

我们可以在登陆 MySQL 服务后，使用 **create** 命令创建数据库，语法如下：

```
CREATE DATABASE 数据库名；
```

以下命令简单的演示了创建数据库的过程，数据名为 RUNOOB：

```
[root@host]# mysql -u root -p
Enter password:***** # 登录后进入终端

mysql> create DATABASE RUNOOB;
```

建数据库的基本语法如下：

```
CREATE DATABASE [IF NOT EXISTS] database_name
[CHARACTER SET charset_name]
[COLLATE collation_name];
```

如果你希望在创建数据库时指定一些选项，可以使用 CREATE DATABASE 语句的其他参数，例如，你可以指定字符集和排序规则：

实例

```
CREATE DATABASE mydatabase
CHARACTER SET utf8mb4
COLLATE utf8mb4_general_ci;
```

如果数据库已经存在，执行 CREATE DATABASE 将导致错误。

为了避免这种情况，你可以在 CREATE DATABASE 语句中添加 IF NOT EXISTS 子句：

实例

```
CREATE DATABASE IF NOT EXISTS mydatabase;
```

## 查询数据库

要查看当前 MySQL 服务器上的数据库列表，使用以下命令：

```
show databases;
```

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
4 rows in set (0.00 sec)
```

上述命令的执行结果如上图所示，在所示的数据库列表中，information\_schema 和 mysql 为系统数据库，test 为测试数据库，bookstore 为刚刚创建的数据库。information\_schema 是信息数据库，其中保存着 MySQL 服务器所维护的所有其他数据库的信息。mysql 数据库存

储了 MySQL 的账户信息以及 MySQL 账户的访问权限，进而实现 MySQL 的账户的身份认证和权限验证，确保数据库安全。test 数据库则是安装时创建的一个测试数据库，是一个空数据库，其中没有任何表，可以删除

## 显示数据库

数据库创建好之后，可以使用以下 MySQL 命令来查看数据库的相关信息，如默认字符集等：

```
show create database bookstore;
```

## 选择当前数据库

在进行数据库操作前，必须指定操作的是哪个数据库，即需要指定哪一个数据库为当前数据库。在使用 CREATE DATABASE 命令创建新的数据库后，新数据库并不会自动的成为当前数据库。使用以下命令进行指定：

```
use bookstore;
```

## 删除数据库

- 如果要删除某个指定的数据库，如 bookstore 数据库，则使用如下命令：

```
drop database bookstore;
```

# 创建、删除、查询数据表

## 创建数据表

创建 MySQL 数据表需要以下信息：

- 表名
- 表字段名
- 定义每个表字段的数据类型

### 语法

以下为创建 MySQL 数据表的 SQL 通用语法：

```
CREATE TABLE table_name (
 column1 datatype,
 column2 datatype,
 ...
);
```

## 参数说明：

- `table_name` 是你要创建的表的名称。
- `column1, column2, ...` 是表中的列名。
- `datatype` 是每个列的数据类型。

以下是一个具体的实例，创建一个用户表 `users`：

## 实例

```
CREATE TABLE users (
 id INT AUTO_INCREMENT PRIMARY KEY,
 username VARCHAR(50) NOT NULL,
 email VARCHAR(100) NOT NULL,
 birthdate DATE,
 is_active BOOLEAN DEFAULT TRUE
);
```

## 实例解析：

- `id`: 用户 `id`, 整数类型, 自增长, 作为主键。
- `username`: 用户名, 变长字符串, 不允许为空。
- `email`: 用户邮箱, 变长字符串, 不允许为空。
- `birthdate`: 用户的生日, 日期类型。
- `is_active`: 用户是否已经激活, 布尔类型, 默认值为 `true`。

以上只是一个简单的实例，用到了一些常见的数据类型包括 `INT`, `VARCHAR`, `DATE`, `BOOLEAN`，你可以根据实际需要选择不同的数据类型。`AUTO_INCREMENT` 关键字用于创建一个自增长的列，`PRIMARY KEY` 用于定义主键。

如果你希望在创建表时指定数据引擎, 字符集和排序规则等, 可以使用 `CHARACTER SET` 和 `COLLATE` 子句：

## 实例

```
CREATE TABLE mytable (
 id INT PRIMARY KEY,
 name VARCHAR(50)
) CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
```

以上代码创建一个使用 utf8mb4 字符集和 utf8mb4\_general\_ci 排序规则的表。

以下例子中我们将在 RUNOOB 数据库中创建数据表 runoob\_tbl：

## 实例

```
CREATE TABLE IF NOT EXISTS runoob_tbl(
 runoob_id INT UNSIGNED AUTO_INCREMENT,
 runoob_title VARCHAR(100) NOT NULL,
 runoob_author VARCHAR(40) NOT NULL,
 submission_date DATE,
 PRIMARY KEY (runoob_id)
)ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

实例解析：

- 如果你不希望字段为空可以设置字段的属性为 **NOT NULL**，如上实例中的 runoob\_title 与 runoob\_author 字段，在操作数据库时如果输入该字段的数据为空，就会报错。
- **AUTO\_INCREMENT** 定义列为自增的属性，一般用于主键，数值会自动加 1。
- **PRIMARY KEY** 关键字用于定义列为主键。您可以使用多列来定义主键，列间以逗号，分隔。
- **ENGINE** 设置存储引擎，**CHARSET** 设置编码。

## 删除数据表

MySQL 中删除数据表是非常容易操作的，但是你在进行删除表操作时要非常小心，因为执行删除命令后所有数据都会消失。

## 语法

以下为删除 MySQL 数据表的通用语法：

```
DROP TABLE table_name ; -- 直接删除表，不检查是否存在
或
DROP TABLE [IF EXISTS] table_name;
```

## 参数说明：

- `table_name` 是要删除的表的名称。
- `IF EXISTS` 是一个可选的子句，表示如果表存在才执行删除操作，避免因为表不存在而引发错误。

```
-- 删除表，如果存在的话
DROP TABLE IF EXISTS mytable;

-- 直接删除表，不检查是否存在
DROP TABLE mytable;
```

请替换 **mytable** 为你要删除的表的名称。

在命令行中，你也可以使用 **mysqladmin** 工具来删除表。

以下是使用 **mysqladmin** 删除表的命令：

```
mysqladmin -u your_username -p drop your_table
```

- `your_username` 是 MySQL 用户名。
- `your_table` 是要删除的表的名称。

执行此命令后，系统会提示输入密码，输入密码后按 Enter 键即可删除表。

**注意：** 在执行删除表操作时，请确保你确实想要删除表及其所有数据，因为该操作是不可逆的。为了避免误操作，通常建议在执行删除之前备份表。

## 实例

以下实例删除了数据表 runoob\_tbl:

## 实例

```
root@host# mysql -u root -p
Enter password:*****
mysql> **USE** RUNOOB;
DATABASE changed
mysql> **DROP** **TABLE** runoob_tb1;
Query OK, 0 **ROWS** affected (0.8 sec)
mysql>
```

## 查看表

里的查看表指的是查看表的结构。

可以使用**DESCRIBE 表名**进行查看

我们还可以通过语句**SHOW CREATE TABLE 表名**的方法将展示表的详细信息

```
SHOW TABLES #查看已创建的表格
SHOW DATABASES #查看已创建的数据库
```

## 插入数据

MySQL 表中使用 **INSERT INTO** 语句来插入数据。

你可以通过 **mysql>** 命令提示窗口中向数据表中插入数据，或者通过PHP脚本来插入数据。

## 语法

以下为向MySQL数据表插入数据通用的 **INSERT INTO** SQL语法：

```
INSERT INTO table_name (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...);
```

### 参数说明：

- `table_name` 是你要插入数据的表的名称
- `column1, column2, column3, ...` 是表中的列名
- `value1, value2, value3, ...` 是要插入的具体数值

如果数据是字符型，必须使用单引号 ' 或者双引号 "，如： 'value1'， "value1"。

一个简单的实例，插入了一行数据到名为 users 的表中：

```
INSERT INTO users (username, email, birthdate, is_active)
VALUES ('test', 'test@runoob.com', '1990-01-01', true);
```

- `username`: 用户名，字符串类型
- `email`: 邮箱地址，字符串类型
- `birthdate`: 用户生日，日期类型
- `is_active`: 是否已激活，布尔类型

如果你要插入所有列的数据，可以省略列名：

```
INSERT INTO users
VALUES (NULL, 'test', 'test@runoob.com', '1990-01-01',
true);
```

这里，**NULL** 是用于自增长列的占位符，表示系统将为 **id** 列生成一个唯一的值。

如果你要插入多行数据，可以在 VALUES 子句中指定多组数值：

```
INSERT INTO users (username, email, birthdate, is_active)
VALUES
 ('test1', 'test1@runoob.com', '1985-07-10', true),
 ('test2', 'test2@runoob.com', '1988-11-25', false),
 ('test3', 'test3@runoob.com', '1993-05-03', true);
```

## 查询数据

MySQL 数据库使用 **SELECT** 语句来查询数据。

你可以通过 **mysql>** 命令提示窗口中在数据库中查询数据，或者通过 PHP 脚本来查询数据。

# 语法

以下为在 MySQL 数据库中查询数据通用的 SELECT 语法：

```
SELECT column1, column2, ...
FROM table_name
[WHERE condition]
[ORDER BY column_name [ASC | DESC]]
[LIMIT number];
```

## 参数说明：

- `column1, column2, ...` 是你想要选择的列的名称，如果使用 `*` 表示选择所有列。
- `table_name` 是你要从中查询数据的表的名称。
- `WHERE condition` 是一个可选的子句，用于指定过滤条件，只返回符合条件的行。
- `ORDER BY column_name [ASC | DESC]` 是一个可选的子句，用于指定结果集的排序顺序，默认是升序 (ASC)。
- `LIMIT number` 是一个可选的子句，用于限制返回的行数。

MySQL SELECT 语句简单的应用实例：

## 实例

```
-- 选择所有列的所有行
SELECT * FROM users;

-- 选择特定列的所有行
SELECT username, email FROM users;

-- 添加 WHERE 子句，选择满足条件的行
SELECT * FROM users WHERE is_active = TRUE;

-- 添加 ORDER BY 子句，按照某列的升序排序
SELECT * FROM users ORDER BY birthdate;

-- 添加 ORDER BY 子句，按照某列的降序排序
SELECT * FROM users ORDER BY birthdate DESC;
```

```
-- 添加 LIMIT 子句，限制返回的行数
SELECT * FROM users LIMIT 10;
```

SELECT 语句可以是灵活的，我们可以根据实际需求组合和使用这些子句，比如同时使用 WHERE 和 ORDER BY 子句，或者使用 LIMIT 控制返回的行数。

在 WHERE 子句中，你可以使用各种条件运算符（如 =, <, >, <=, >=, !=），逻辑运算符（如 AND, OR, NOT）以及通配符（如 %）等。

以下是一些进阶的 SELECT 语句实例：

### 实例

```
-- 使用 AND 运算符和通配符
SELECT * FROM users WHERE username LIKE 'j%' AND is_active
= TRUE;

-- 使用 OR 运算符
SELECT * FROM users WHERE is_active = TRUE OR birthdate <
'1990-01-01';

-- 使用 IN 子句
SELECT * FROM users WHERE birthdate IN ('1990-01-01',
'1992-03-15', '1993-05-03');
```

## 更新数据

如果我们需要修改或更新 MySQL 中的数据，我们可以使用 UPDATE 命令来操作。

以下是 UPDATE 命令修改 MySQL 数据表数据的通用 SQL 语法：

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

### 参数说明：

- table\_name 是你要更新数据的表的名称

- `column1, column2, ...` 是你要更新的列的名称
- `value1, value2, ...` 是新的值，用于替换旧的值
- `WHERE condition` 是一个可选的子句，用于指定更新的行。如果省略 `WHERE` 子句，将更新表中的所有行

## 更多说明：

- 你可以同时更新一个或多个字段
- 你可以在 `WHERE` 子句中指定任何条件
- 你可以在一个单独表中同时更新数据

当你需要更新数据表中指定行的数据时 `WHERE` 子句是非常有用的

## 实例

以下实例演示了如何使用 `UPDATE` 语句。

### 1. 更新单个列的值

```
UPDATE employees SET salary = 60000 WHERE employee_id = 101;
```

### 2. 更新多个列的值

```
UPDATE orders
SET status = 'Shipped', ship_date = '2023-03-01'
WHERE order_id = 1001;
```

### 3. 使用表达式更新值

```
UPDATE products
SET price = price * 1.1
WHERE category = 'Electronics';
```

以上 SQL 语句将每个属于 'Electronics' 类别的产品的价格都增加了 10%。

### 4. 更新符合条件的所有行

```
UPDATE students
SET status = 'Graduated';
```

以上 SQL 语句将所有学生的状态更新为 'Graduated'。

## 5. 更新使用子查询的值

```
UPDATE customers
SET total_purchases = (
 SELECT SUM(amount)
 FROM orders
 WHERE orders.customer_id = customers.customer_id
)
WHERE customer_type = 'Premium';
```

以上 SQL 语句通过子查询计算每个 'Premium' 类型客户的总购买金额，并将该值更新到 total\_purchases 列中。

**注意：**在使用 UPDATE 语句时，请确保你提供了足够的条件来确保只有你想要更新的行被修改。如果不提供 WHERE 子句，将更新表中的所有行，可能导致不可预测的结果。

## 删除数据

## 删除用户

在 MySQL 数据库中，可以使用 DROP USER 语句删除用户，也可以直接在 mysql.user 表中删除用户以及相关权限。

## 1. 使用 DROP USER 语句删除普通用户

使用 DROP USER 语句删除用户的语法格式如下：

```
DROP USER <用户1> [, <用户2>]...
```

其中，用户用来指定需要删除的用户账号。

使用 DROP USER 语句应注意以下几点：

- DROP USER 语句可用于删除一个或多个用户，并撤销其权限。
- 使用 DROP USER 语句必须拥有 mysql 数据库的 DELETE 权限或全局 CREATE USER 权限。
- 在 DROP USER 语句的使用中，若没有明确地给出账户的主机名，则该主机名默认为“%”。

**注意：**用户的删除不会影响他们之前所创建的表、索引或其他数据库对象，因为 MySQL 并不会记录是谁创建了这些对象。

### 例 1

下面使用 DROP USER 语句删除用户'test1@'localhost'。SQL 语句和执行过程如下：

```
1 | mysql> DROP USER 'test1'@'localhost';
2 | Query OK, 0 rows affected (0.00 sec)
```

在 cmd 命令行工具中，使用 test1 用户登录数据库服务器，发现登录失败，说明用户已经删除，如下所示：

```
1 | C:\Users\USER>mysql -h localhost -u test1 -p
2 | Enter password: ****
3 | ERROR 1045 (28000): Access denied for user 'test'@'localhost' (using password: YES)
```

## 2. 使用DELETE语句删除普通用户

可以使用 DELETE 语句直接删除 mysql.user 表中相应的用户信息，但必须拥有 mysql.user 表的 DELETE 权限。其基本语法格式如下：

```
DELETE FROM mysql.user WHERE Host='hostname' AND User='username';
```

Host 和 User 这两个字段都是 mysql.user 表的主键。因此，需要两个字段的值才能确定一条记录。

### 例 2

下面使用 DELETE 语句删除用户'test2'@'localhost'。SQL 语句和执行过程如下所示：

```
1 | DELETE FROM mysql.user WHERE Host='localhost' AND User='test2';
2 | Query OK, 1 rows affected (0.00 sec)
```

结果显示删除成功。可以使用 SELECT 语句查询 mysql.user 表，以确定该用户是否已经成功删除。

# 创建用户和赋权

## 正确的执行赋权

那么在MySQL8.0版本及以后，我们如何正确执行grant呢？

**先创建用户，再赋予授权。**

```
1 | mysql> create user test@'localhost' identified by '123456';
2 | Query OK, 0 rows affected (0.10 sec)
3 |
4 | mysql> grant all privileges on test.* to test@'localhost';
5 | Query OK, 0 rows affected (0.17 sec)
6 |
7 | mysql> flush privileges;
8 | Query OK, 0 rows affected (0.18 sec)
```

这个方法也适用MySQL5.7版本，所以建议大家以后使用这种方式赋权，一键建用户加赋权官方已经弃用了。

# 修改密码

## 在登陆MySQL的情况下

### 方法一：通过sql命令修改密码

命令格式：`set password for 用户名@localhost = password('新密码');`

新版本mysql 命令：`alter user 用户名@localhos identified by '新密码';`

例子：

```
1 | set password for root@localhost = password('123');
2 |
3 | alter user 'root'@'localhost' identified by '123';
```

### 方法二：用UPDATE直接修改user表

```
1 | -- 使用mysql 数据库
2 | use mysql;
3 | -- 更改user 表中指定用户的密码
4 | update user set password=password('123') where user='root' and host='localhost';
5 | -- 权限刷新
6 | flush privileges;
```

# 查看mysql已经创建的用户

## 查看MySQL已经创建的用户

MySQL是一种流行的关系型数据库管理系统，它允许用户创建并管理多个数据库和用户。用户是用于连接和管理数据库的登录凭据。在MySQL中，可以使用特定的命令和查询来查看已经创建的用户列表。本文将介绍如何使用MySQL的命令和查询来查看已经创建的用户。

### 1. 使用命令行工具登录MySQL

首先，我们需要使用命令行工具登录到MySQL服务器。打开终端或命令提示符，并输入以下命令：

```
1. mysql -u your_username -p
```

请将 `your_username` 替换为您的MySQL用户名。然后按下Enter键并输入密码。如果密码正确，您将成功登录到MySQL服务器。

### 2. 查看MySQL已经创建的用户

一旦成功登录到MySQL服务器，我们可以执行以下命令来查看已经创建的用户：

```
1. SELECT User, Host FROM mysql.user;
```

此查询将返回一个结果集，其中包含所有已经创建的用户的用户名和主机信息。每一行表示一个用户，`User` 列显示用户名，`Host` 列显示允许该用户连接的主机。

下面是一个示例输出：

| User     | Host      |
|----------|-----------|
| root     | localhost |
| admin    | 127.0.0.1 |
| testuser | %         |

这里的 `root` 是MySQL的默认超级用户，`admin` 和 `testuser` 是其他普通用户。

### 3. 使用用户名过滤结果

如果您只想查看特定用户的信息，可以使用 `WHERE` 子句来过滤结果。例如，要查看用户名为 `admin` 的用户信息，可以执行以下查询：

```
1. SELECT User, Host FROM mysql.user WHERE User = 'admin';
```

如果用户存在，将返回包含该用户信息的一行。否则，将不返回任何结果。

## 4. 查看用户权限

除了查看用户列表，我们还可以查看每个用户的权限。要查看特定用户的所有权限，可以执行以下查询：

```
1. SHOW GRANTS FOR 'username'@'host';
```

请将 `username` 和 `host` 替换为要查看权限的用户和主机。例如，要查看用户名为 `admin` 的用户在本地主机上的权限，可以执行以下查询：

```
1. SHOW GRANTS FOR 'admin'@'localhost';
```

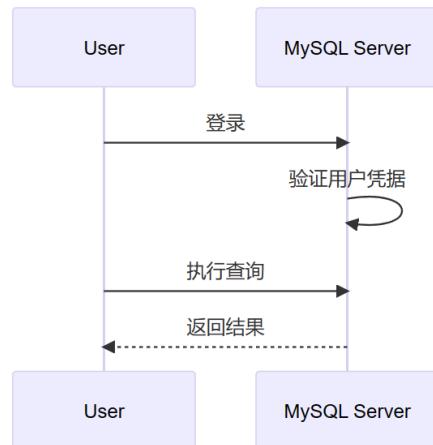
此查询将返回一个结果集，其中包含用户的权限列表。每一行表示一个权限，列出了可以执行的操作和受影响的数据库和表。例如：

```
1. GRANT ALL PRIVILEGES ON *.* TO 'admin'@'localhost' IDENTIFIED BY PASSWORD '****' WITH GRANT OPTION
```

这里的 `GRANT ALL PRIVILEGES` 表示该用户具有所有权限，在所有数据库和表上。`WITH GRANT OPTION` 表示该用户可以为其他用户授予权限。

## 序列图

下面是一个描述上述流程的序列图：



## 流程图

下面是一个描述上述流程的流程图：

