**MySQL安装**

mysql下载地址

# 官网地址  
https://dev.mysql.com/downloads/mysql/  
# 国内的tsinghua镜像  
https://mirror.tuna.tsinghua.edu.cn/mysql/  
# 国内的huwei镜像  
https://mirrors.huaweicloud.com/mysql/Downloads/  
​  
https://mirrors.huaweicloud.com/mysql/Downloads/MySQL-8.0/mysql-8.0.24-linux-glibc2.12-x86\_64.tar.xz

Windows 系统下载的安装文件：

* 免安装方式（推荐）
  + 5.7：[mysql-5.7.28-winx64.zip](%20https:/dev.mysql.com/get/Downloads/MySQL-5.7/mysql-5.7.28-winx64.zip%20%20)
  + 8.0：[mysql-8.0.18-winx64.zip](%20)
* msi安装方式
  + 下载MSI Installer文件，下一步下一步式安装。

Linux 系统下载的安装文件：一般推荐用 **Linux-Generic** 这种二进制源格式的安装文件。

* 5.7：[mysql-5.7.28-linux-glibc2.12-x86\_64.tar.gz](https://dev.mysql.com/get/Downloads/MySQL-5.7/mysql-5.7.28-linux-glibc2.12-x86_64.tar.gz)
* 8.0：[mysql-8.0.23-linux-glibc2.12-x86\_64.tar.xz](https://dev.mysql.com/get/Downloads/MySQL-8.0/mysql-8.0.23-linux-glibc2.12-x86_64.tar.xz)

**1、windows 安装过程**

下载mysql文件并解压到 D:/mysql 目录中，添加 D:/mysql/bin 到 PATH 系统环境变量中。在根目录创建 my.ini 配置文件，内容如下：

[client]  
# 设置mysql客户端连接服务端时默认使用的端口  
port = 3306   
[mysql]  
# 设置mysql客户端默认字符集     
default-character-set=utf8   
[mysqld]  
# mysql服务端默认监听(listen on)的TCP/IP端口号  
port=3306   
# 基准路径，其他路径都相对于这个路径   
basedir="D:\mysql"   
# mysql数据库文件所在目录  
datadir="D:\mysql\data"   
# 默认字符集   
character-set-server=utf8mb4  
# 默认存储引擎  
default-storage-engine=INNODB  
# SQL模式为strict模式  
# sql-mode=STRICT\_TRANS\_TABLES,NO\_AUTO\_CREATE\_USER,NO\_ENGINE\_SUBSTITUTION   
# mysql服务器支持的最大并发连接数（用户数）。如果设置得过小而用户比较多，会经常出现“Too many connections”错误。  
max\_connections=1024  
# 禁用查询缓存，这个参数在5.7.20后已经Deprecated  
# query\_cache\_size=0   
# 内存中的每个临时表允许的最大大小。如果临时表大小超过该值，临时表将自动转为基于磁盘的表（Disk Based Table）。  
tmp\_table\_size=34M   
# 缓存的最大线程数。当客户端连接断开时，如果客户端总连接数小于该值，则处理客户端任务的线程放回缓存。在高并发情况下，如果该值设置得太小，就会有很多线程频繁创建，线程创建的开销会变大，查询效率也会下降。一般来说如果在应用端有良好的多线程处理，这个参数对性能不会有太大的提高。  
thread\_cache\_size=8   
default\_authentication\_plugin = mysql\_native\_password

以管理员身份启动 cmd , 进行数据初始化：

mysql\_install\_db 这个程序在 MySQL 5.7.6 中已经弃用，因为它的功能已经集成在 mysqld 中了，所以在安装 5.7 及以后版本时，直接使用 mysqld --initialize 或者 mysqld --initialize-insecure 就可以直接初始化。在 5.7.5 之前，mysql\_install\_db 这个程序是用 perl 脚本写的，所以需要安装 perl ，5.7.5 之后，改为用 C++ 写的，并且可以直接做为二进制文件执行。

# 不安全的初始化，默认root密码为空，需要登陆后自己设置一个密码  
mysqld --initialize-insecure --console  
# 直接初始化，为root生成一个随机密码，密码打印在控制台中  
mysqld --initialize --console  
​

安装系统服务

mysqld -install MySQL --defaults-file="D:\mysql\my.ini"

启动系统服务

net start mysql

登陆并设置密码

mysql -u root -p  
use mysql;  
update user set authentication\_string=PASSWORD("123456AWS!@#") where user="root";  
flush privileges;  
quit;

默认地，MySQL 会创建如下用户：

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
​  
mysql> select user,host from mysql.user;  
+------------------+-----------+  
| user             | host     |  
+------------------+-----------+  
| mysql.infoschema | localhost |  
| mysql.session   | localhost |  
| mysql.sys       | localhost |  
| root             | localhost |  
+------------------+-----------+  
4 rows in set (0.05 sec)  
​  
mysql>  
​  
​

**用户管理**

我们在创建数据库用户的时候都会指定host，即一个完整的用户可描述为 'username'@'host' 。

创建用户时不显式指定host则默认为%，%代表所有 ip 段都可以使用这个用户，我们也可以指定host为某个ip或ip段，这样会仅允许在指定的ip主机使用该数据库用户。

不过你也应该明白 'username'@'%' 和 'username'@'192.168.6.%' 是两个毫无关联的用户，这两个用户可以有不同的密码和权限，这里不建议创建多个同名不同host的用户，还有不要轻易更改用户的host。

**2、Linux 安装过程**

**2.1 二进制安装**

**环境**

* GNU/Linux-x86\_64
* gcc 运行时环境

**安装规范**

|  |  |
| --- | --- |
| 配置项 | 配置值 |
| basedir | /usr/local/mysql/ |
| socket | /data/mysql/mysql.socks |
| log\_error | /data/msql/mysql.err |
| pid\_file | /data/mysql/mysql.pid |
| datadir | /data/mysql/data/ |
|  |  |

**用户和数据目录创建**

$ groupadd mysql  
$ useradd -r -g mysql -s /bin/false mysql  
$ mkdir -p /data/mysql/{data,binlog}   
$ chown -R mysql:mysql /data/mysql

**创建 /etc/my.cnf文件**

这是最基本的配置文件，更详细的配置，可以参考相关mysql优化文档

[client]

port = 3306

socket = /data/mysql/mysql.sock

#default-character-set　=　utf8mb4

[mysql]

no-auto-rehash

default-character-set = utf8mb4

[mysqld]

#skip-grant-tables

log\_timestamps=SYSTEM

user = mysql

port = 3306

basedir = /usr/local/mysql

log\_error = /data/mysql/mysql.error

datadir = /data/mysql/data/

socket = /data/mysql/mysql.sock

pid-file = /data/mysql/db.pid

character-set-server = utf8mb4

collation-server = utf8mb4\_general\_ci

default\_authentication\_plugin = mysql\_native\_password

skip\_name\_resolve = 1

open\_files\_limit = 65535

back\_log = 1024

default\_time\_zone='+8:00'

# innodb\_buffer\_pool\_size=32G

innodb\_buffer\_pool\_instances=8

innodb\_log\_file\_size=4G

innodb\_log\_files\_in\_group=4

# binlog过期时间3天

binlog\_expire\_logs\_seconds=259200

wait\_timeout = 9000

interactive\_timeout = 9000

net\_write\_timeout=6000

net\_read\_timeout=6000

thread\_cache\_size=10

sync\_binlog=800

innodb\_autoextend\_increment=64

open\_files\_limit=10200

#sort\_buffer\_size=256M

**下载**

$ cd /tmp/

$ wget https://mirrors.huaweicloud.com/mysql/Downloads/MySQL-8.0/mysql-8.0.23-linux-glibc2.12-x86\_64.tar.xz

# $ wget https://dev.mysql.com/get/Downloads/MySQL-5.7/mysql-5.7.24-linux-glibc2.12-x86\_64.tar.gz

**解压并安装**

$ tar -zxvf /tmp/mysql-8.0.23-linux-glibc2.12-x86\_64.tar.xz -C /usr/local

$ mv /usr/local/mysql-8.0.23-linux-glibc2.12-x86\_64 /usr/local/mysql

$ cd /usr/local/mysql

$ ./bin/mysqld --initialize-insecure \

--basedir=/usr/local/mysql \

--datadir=/data/mysql/data \

--user=mysql \

--pid-file=/data/mysql/mysql.pid

$ ./support-files/mysql.server start

$ ./support-files/mysql.server status

5.6与5.7在初始化的时候有一些区别，在 5.7.6 以前都用 mysql\_install\_db 来初始化数据库，在这之后的版本，由于 mysqld 程序已经集成了初始化数据库功能， mysql\_install\_db 这个功能在未来的版本中可能会被去掉。所以建议直接使用 mysqld --initialize-insecure 这样的方法来进行数据初始化。

**添加到系统服务和开机自启**

$ cp /usr/local/mysql/support-files/mysql.server /etc/init.d/mysql

$ chkconfig --add mysql

$ chkconfig mysql on

**设置root密码，并开启任意IP登陆**

/usr/local/mysql/bin/mysql -u root -p

alter user root@'localhost' IDENTIFIED by 'Sj36668182' ;

-- UPDATE mysql.user SET authentication\_string=PASSWORD("QH@123456") WHERE user='root' ;

grant all privileges on \*.\* to 'root' @'%' identified by '123456';

flush privileges;

**设置 mysql 环境变量**

将 mysql 安装路径的 bin 目录加到 Linux 的 PATH 环境，方便以后在任意目录下都可以都可以执行 mysql 的相关命令。

echo "PATH=$PATH:/usr/local/mysql/bin" >> /etc/profile

# echo "PATH=$PATH:/usr/local/mysql/bin" >> ~/.bashrc

source /etc/profile

**mysql服务管理**

$ systemctl status mysql

$ systemctl start mysql

$ systemctl stop mysql

$ systemctl restart mysql

**2.2 包管理器安装**

包管理器也是一种简单的安装方式，但是自由性没有那么好，自定义程度也没有那么好。

常用的 Linux 包管理器有 yum 和 apt 。分别适用于 rhel 系和 debian 系的 Linux 发行版。

**2.2.1、yum 安装**

添加 MySQL Yum 仓库到系统仓库列表中

$ wget -P /tmp https://dev.mysql.com/get/mysql80-community-release-el7-1.noarch.rpm

$ wget -P /tmp https://dev.mysql.com/get/mysql80-community-release-el6-1.noarch.rpm

$ rpm -Uvh platform-and-version-specific-package-name.rpm

1. 选择版本

默认地，Yum 包安装的是最新的 GA 版本（标准发布版），如果这正是你需要的，可以跳过这一步。

在 MySQL Yum 仓库中，不同的发行版在不同的子仓库中，默认当前是当前最新的 GA 版本（8.0），5.7 系列默认是被禁用的，使用下面的命令来查看所有 MySQL 子仓库，看看哪些被启用，哪些被禁用

$ yum repolist all | grep mysql

$ dnf repolist all | grep mysql

如果要安装 5.7 系列的，需要禁用最新的 GA 版本子仓库，启用 5.7 系列版本子仓库。

$ sudo yum-config-manager --disable mysql80-community

$ sudo yum-config-manager --enable mysql57-community

这两行命令，其实就是修改 /etc/yum.repos.d/mysql-community.repo 这个文件里面的内容，通过 enabled 字段来控制启用和禁用。

[mysql80-community]

name=MySQL 8.0 Community Server

baseurl=http://repo.mysql.com/yum/mysql-8.0-community/el/6/$basearch/

enabled=0

gpgcheck=1

gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-mysql

# Enable to use MySQL 5.7

[mysql57-community]

name=MySQL 5.7 Community Server

baseurl=http://repo.mysql.com/yum/mysql-5.7-community/el/6/$basearch/

enabled=1

gpgcheck=1

gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-mysql

验证当前子仓库的可用性

$ yum repolist enabled | grep mysql

1. 安装 MySQL

$ sudo yum install mysql-community-server

1. 启动 MySQL

$ sudo service mysqld start

$ sudo systemctl start mysqld.service

$ sudo service mysqld status

$ sudo service mysqld status

对于 5.7 ，root账号会随机生成一个密码存在日志文件中，通过 sudo grep 'temporary password' /var/log/mysqld.log 命令来查看。

密码要求如下：至少一个大写字母、至少一个小写字母、至少一个数字、至少一个特殊符号、字符总长度不小于8。

**3、用户管理**

**用户标识**

在 MySQL 中，用户是由 mysql.user 表中的 Host 和 User 两个字段一起来 **唯一标识** 一个用户的。

任何对用户的相关操作（赋权，修改密码等），都是通过 user@'host' 这样来标识一个用户。

MySQL 8.0 在用户管理方面增加了角色管理，默认的密码加密方式也做了调整，由之前的 SHA1 改为了 SHA2。

同时加上 MySQL 5.7 的禁用用户和用户过期的功能，MySQL 在用户管理方面的功能和安全性都较之前版本大大的增强了。

在 MySQL 8.0 中，caching\_sha2\_password 是默认的身份验证插件，而不是之前版本的 mysql\_native\_password，默认的密码加密方式是 SHA2 。

<https://www.cnblogs.com/yinzhengjie/p/10301516.html>

由于我们实际开发过程中的 navicate 等工具或各种语言的客户端，可能没有即使更新不支持 8.0 的这种认证方式，所以一般我们安装时都会设置为以前的认证方式。

mysql> show variables like 'default\_authentication\_plugin';

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

| Variable\_name | Value |

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

| default\_authentication\_plugin | caching\_sha2\_password |

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

1 row in set (0.00 sec)

mysql> select user,host,plugin from mysql.user;

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

| user | host | plugin |

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

| root | % | caching\_sha2\_password |

| mysql.infoschema | localhost | mysql\_native\_password |

| mysql.session | localhost | mysql\_native\_password |

| mysql.sys | localhost | mysql\_native\_password |

| root | localhost | caching\_sha2\_password |

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

5 rows in set (0.00 sec)

如果需要保持之前的验证方式和密码加密方式，需要在配置文件 my.cnf 中修改以下配置项并重启服务后生效。

[mysqld]

default\_authentication\_plugin = mysql\_native\_password

将 MySQL 8.0 中已有的用户的 SHA2 密码修改为 SHA1 的模式。

**用户授权和修改密码**

**MySQL8创建用户的操作已经不支持grant的同时创建用户的方式，需先创建用户再进行授权，下面这种操作将会报错**

GRANT ALL PRIVILEGES ON \*.\* TO `mike`@`%` IDENTIFIED BY '000000' WITH GRANT OPTION;

**密码过期管理**

MySQL可以给密码设置生命周期，让一个密码只在生命周期内有效，到期后强迫用户修改密码，如果不修改，则将用户锁定，无法登陆。

MySQL 从 5.6.6 开始引入密码自动过期的新功能，添加了 password\_expired 功能，它允许设置用户的过期时间。

MySQL 5.7.4 版本中改进了用户密码过期时间这个特性。可以通过一个全局变量 default\_password\_lifetime 来设置一个全局的自动密码过期策略。

# mysql5.7.10(GA)版本之后默认的 default\_password\_lifetime 默认值为 0，表示禁用自动密码过期。默认全局配置：所有用户的密码永不过期

# 如果要建立全局策略，让所有用户的密码的使用期限为六个月，可在服务端配置文件 my.cnf 中修改 default\_password\_lifetime 配置项的值为 180。

# 如果是正整数 N ，则表示允许的设置密码生存周期 为 N，单位为天 。

**密码策略**

-- 查看密码策略

show variables like 'validate\_password%';

-- 新密码不能和前面三次的密码相同

password\_history = 3 ;

-- 新密码不能和前面九十天内使用的密码相同

password\_reuse\_interval = 90 ;

-- 密码验证策略低要求(0或LOW代表低级)

validate\_password.policy=0;

-- 密码至少要包含的小写字母个数和大写字母个数

validate\_password.mixed\_case\_count=0;

-- 密码至少要包含的数字个数

validate\_password.number\_count=0;

-- 密码至少要包含的特殊字符数

validate\_password.special\_char\_count=0;

-- 密码至少长度

validate\_password.length=6;

**用户管理语句**

-- 建用户

CREATE USER 'dba'@'%' IDENTIFIED WITH 'mysql\_native\_password' by 'admin@123' ;

-- 赋超级权限

GRANT ALL PRIVILEGES ON \*.\* TO 'dba'@'%';

CREATE USER 'shendandan'@'192.168.%' IDENTIFIED WITH 'mysql\_native\_password' by 'admin@123' ;

-- 指定密码过期，以便用户第一次使用的时候需要修改密码

CREATE USER 'jeffrey'@'localhost' IDENTIFIED WITH 'caching\_sha2\_password' BY 'new\_password' PASSWORD EXPIRE DEFAULT;

-- 不使用加密连接

CREATE USER 'dba'@'%' IDENTIFIED WITH 'mysql\_native\_password' by 'admin@123' REQUIRE NONE ;

-- 使用加密连接

CREATE USER 'dba'@'%' IDENTIFIED WITH 'mysql\_native\_password' by 'admin@123' REQUIRE NONE ;

-- 创建用户并且设置资源闲置（如果不带资源选项，默认不限制资源）

-- 单位小时内账户被允许查询500次

-- 单位小时内账户被允许更新100次

-- 单位小时内最大连接数不限制

-- 同一时刻最大并发连接数不限制

CREATE USER 'jeffrey'@'%'

IDENTIFIED WITH 'mysql\_native\_password' by 'admin@123'

WITH

MAX\_QUERIES\_PER\_HOUR 500

MAX\_UPDATES\_PER\_HOUR 100

MAX\_CONNECTIONS\_PER\_HOUR 0

MAX\_USER\_CONNECTIONS 0;

-- 锁定用户

CREATE USER 'jeffrey'@'localhost' ACCOUNT LOCK

-- 解锁用户

ALTER USER 'jeffrey'@'localhost' ACCOUNT UNLOCK

-- 删除用户

DROP USER 'jeffrey'@'localhost';

-- 修改用户

RENAME USER 'jeffrey'@'localhost' TO 'jeff'@'127.0.0.1';

-- 修改当前用户自己账号的密码

ALTER USER USER() IDENTIFIED BY 'new\_password';

-- 修改其他用户账号的密码

ALTER USER 'jeffrey'@'localhost' IDENTIFIED BY 'new\_password';

-- 修改用户认证插件

ALTER USER 'jeffrey'@'localhost' IDENTIFIED WITH mysql\_native\_password;

-- 修改密码和插件

ALTER USER 'jeffrey'@'localhost' IDENTIFIED WITH mysql\_native\_password BY 'new\_password';

-- 详细语法

CREATE USER [IF NOT EXISTS]

user [auth\_option] [, user [auth\_option]] ...

DEFAULT ROLE role [, role ] ...

[REQUIRE {NONE | tls\_option [[AND] tls\_option] ...}]

[WITH resource\_option [resource\_option] ...]

[password\_option | lock\_option] ...

user:

(see Section 6.2.4, “Specifying Account Names”)

auth\_option: {

IDENTIFIED BY 'auth\_string'

| IDENTIFIED WITH auth\_plugin

| IDENTIFIED WITH auth\_plugin BY 'auth\_string'

| IDENTIFIED WITH auth\_plugin AS 'hash\_string'

}

tls\_option: {

SSL

| X509

| CIPHER 'cipher'

| ISSUER 'issuer'

| SUBJECT 'subject'

}

resource\_option: {

MAX\_QUERIES\_PER\_HOUR count

| MAX\_UPDATES\_PER\_HOUR count

| MAX\_CONNECTIONS\_PER\_HOUR count

| MAX\_USER\_CONNECTIONS count

}

password\_option: {

PASSWORD EXPIRE [DEFAULT | NEVER | INTERVAL N DAY]

| PASSWORD HISTORY {DEFAULT | N}

| PASSWORD REUSE INTERVAL {DEFAULT | N DAY}

| PASSWORD REQUIRE CURRENT [DEFAULT | OPTIONAL]

}

lock\_option: {

ACCOUNT LOCK

| ACCOUNT UNLOCK

}

**4、权限管理**

grant 是MySQL中的权限管理语句，所有的权限都是从 root 赋权出来的。

赋予某个权限给某个用户时，最后跟上 with grant option，表示这个用户有这个权限有赋权权限，就是它可以把这个权限赋给其他用户。

revoke 是权限收回语句