


访问 MySQL 数据库

一、安装 MySQL 数据库服务器

1. 下载 MySQL

<https://www.mysql.com/downloads/>



The world's most popular open source database

MySQL.COM DOWNLOADS DOCUMENTATION DEVELOPER ZONE

Contact MySQL Sales | Login | Register

f t in y

Enterprise Community Yum Repository APT Repository SUSE Repository Windows Archives

Contact Sales


USA: +1-866-221-0634
Canada: +1-866-221-0634

Germany: +49 89 143 01280
France: +33 1 57 60 83 57
Italy: +39 02 249 59 120
UK: +44 207 553 8447

Japan: 0120-065556
China: 10800-811-0823
India: 008001005870

[More Countries »](#)

[Contact Us Online »](#)



NoSQL + SQL = MySQL

[Learn More »](#)

MySQL Downloads

Oracle MySQL Cloud Service (commercial)

Oracle MySQL Cloud Service is built on MySQL Enterprise Edition and powered by Oracle Cloud, providing an enterprise-grade MySQL database service.

[Learn More »](#)

MySQL Enterprise Edition (commercial)

MySQL Enterprise Edition includes the most comprehensive set of advanced features and management tools for MySQL.

- MySQL Database
- MySQL Storage Engines (InnoDB, MyISAM, etc.)
- MySQL Connectors (JDBC, ODBC, .Net, etc.)
- MySQL Replication
- MySQL Partitioning
- MySQL Utilities
- MySQL Workbench
- MySQL Enterprise Backup
- MySQL Enterprise Monitor
- MySQL Enterprise HA
- MySQL Enterprise Security
- MySQL Enterprise Transparent Data Encryption (TDE)
- MySQL Enterprise Firewall
- MySQL Enterprise Encryption
- MySQL Enterprise Audit

[Learn More »](#)

[Customer Download »](#) (Select Patches & Updates Tab, Product Search)

[Trial Download »](#) (Note - Select Product Pack: MySQL Database)

MySQL Cluster CGE (commercial)

MySQL Cluster is a real-time open source transactional database designed for fast, always-on access to data under high throughput conditions.

- MySQL Cluster
- MySQL Cluster Manager
- Plus, everything in MySQL Enterprise Edition


[Learn More »](#)

[Customer Download »](#) (Select Patches & Updates Tab, Product Search)

[Trial Download »](#) (Note - Select Product Pack: MySQL Database)


MySQL Community Edition (GPL)

[Community \(GPL\) Downloads »](#)

 **Contact MySQL Sales**
USA/Canada: +1-866-221-0634 ([More Countries »](#))

f t in y

PRODUCTS	SERVICES	DOWNLOADS	DOCUMENTATION	ABOUT MYSQL
Oracle MySQL Cloud Service	Training	MySQL Community Server	MySQL Reference Manual	Contact Us
MySQL Enterprise Edition	Certification	MySQL NDB Cluster	MySQL Workbench	How to Buy
MySQL Standard Edition	Consulting	MySQL Shell	MySQL NDB Cluster	Partners
MySQL Classic Edition	Support	MySQL Router	MySQL Connectors	Job Opportunities
MySQL Cluster CGE		MySQL Workbench	Topic Guides	Site Map
MySQL Embedded (OEM/ISV)				

 English (Deutsch | Français | Italiana | 日本 | 中文)

ORACLE © 2019, Oracle Corporation and/or its affiliates

[Legal Policies](#) | [Your Privacy Rights](#) | [Terms of Use](#) | [Trademark Policy](#) | [Contributor Agreement](#) | [Cookie Preferences](#)

选择版本: MySQLCommunityEdition(GPL)

MySQL Community Edition (GPL)

Community (GPL) Downloads »

选择版本: MySQLCommunityServer(GPL)

MySQLCommunityServeristheworld'smostpopularopensourcedatabase.

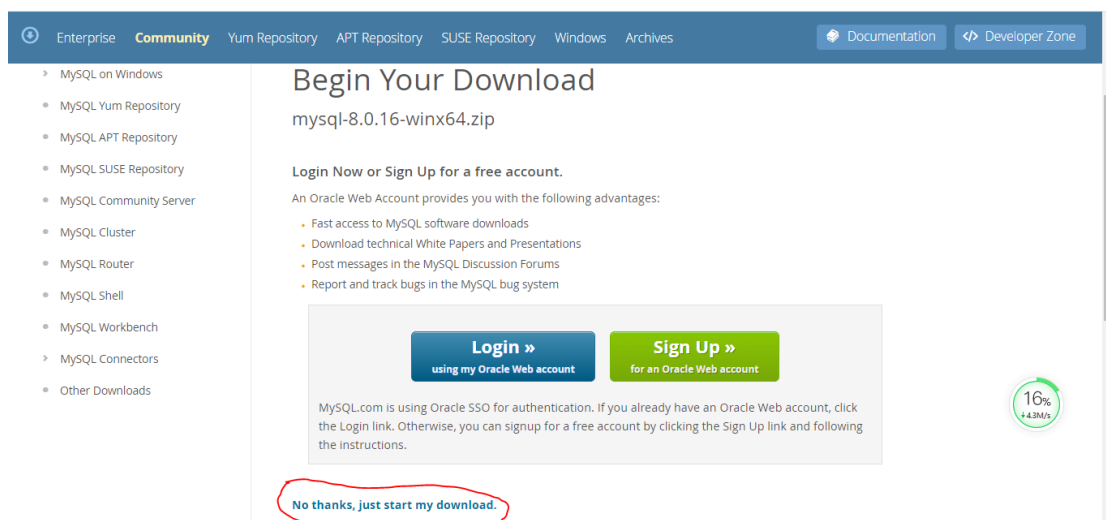
The screenshot shows the MySQL Community Downloads page. The navigation bar includes links for MySQL.COM, DOWNLOADS, DOCUMENTATION, and DEVELOPER ZONE. The main content area is titled 'MySQL Community Downloads'. Under this title, there are two main sections: 'MySQL Community Server (GPL)' and 'MySQL Cluster (GPL)'. The 'MySQL Community Server (GPL)' section is circled in red and includes the text 'MySQL Community Server is the world's most popular open source database.' and a 'DOWNLOAD' link. The 'MySQL Cluster (GPL)' section includes the text 'MySQL Cluster is a real-time, open source transactional database.' and a 'DOWNLOAD' link. To the right, there is a section for 'MySQL Enterprise Edition (commercial)' with a '14% OFF' badge and a 'Download from Oracle eDelivery' link. At the bottom, there is a link for 'MySQL Cluster CGE (commercial)'.

选择版本: Windows(x86,64-bit),ZIPArchive

The screenshot shows the MySQL Installer for Windows download page. At the top, there is a 'Select Operating System:' dropdown menu set to 'Microsoft Windows'. To the right, there is a link 'Looking for previous GA versions?'. Below this, the 'Recommended Download' section features a large image of the MySQL Installer for Windows and a 'Go to Download Page >' button. Underneath, the 'Other Downloads' section contains a table with two rows of download links. The first row is for 'Windows (x86, 64-bit), ZIP Archive' (8.0.16, 228.9M) with a 'Download' button circled in red. The second row is for 'Windows (x86, 64-bit), ZIP Archive Debug Binaries & Test Suite' (8.0.16, 336.2M) with a 'Download' button. At the bottom, there is a note: 'We suggest that you use the MD5 checksums and GnuPG signatures to verify the integrity of the packages you download.'

Download Link	Version	Size	Action
Windows (x86, 64-bit), ZIP Archive (mysql-8.0.16-winx64.zip)	8.0.16	228.9M	Download
Windows (x86, 64-bit), ZIP Archive Debug Binaries & Test Suite (mysql-8.0.16-winx64-debug-test.zip)	8.0.16	336.2M	Download

不必登陆直接下载：[Nothanks,juststartmydownload.](#)



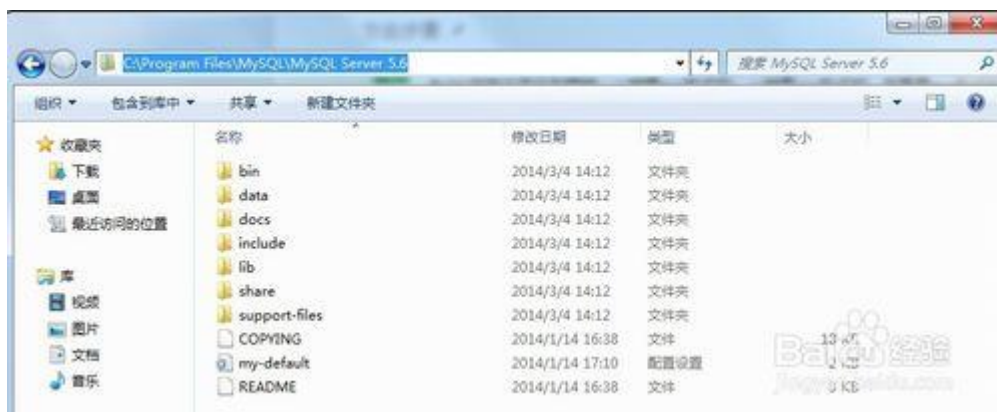
2. 安装配置 **mySQL**

参考如下：

MySQL 是一个小巧玲珑但功能强大的数据库，目前十分流行。但是官网给出的安装包有两种格式，一个是 msi 格式，一个是 zip 格式的。很多人下了 zip 格式的解压发现没有 setup.exe，面对一堆文件一头雾水，不知如何安装。下面笔者将介绍如何解决此情况下安装过程中的各种问题

方法/步骤

- (1) MySQL 安装文件分为两种，一种是 msi 格式的，一种是 zip 格式的。如果是 msi 格式的可以直接点击安装，按照它给出的安装提示进行安装（相信大家的英文可以看懂英文提示），一般 MySQL 将会安装在 C:\ProgramFiles\MySQL\MySQLServer5.6 该目录中；zip 格式是自己解压，解压缩之后其实 MySQL 就可以使用了，但是要进行配置。
- (2) 解压之后可以将该文件夹改名，放到合适的位置，个人建议把文件夹改名为 MySQLServer5.6，放到 C:\ProgramFiles\MySQL 路径中。当然你也可以放到自己想放的任意位置。



- (3) 完成上述步骤之后，很多用户开始使用 MySQL，但会出现图示的错误。这是因为没有配置环境变量所致。配置环境变量很简单：我的电脑->属性->高级->环境变量，选择 PATH,在其后面添加:你的 mysqlbin 文件夹的路径(如:C:\ProgramFiles\MySQL\MySQLServer5.6\bin)
- PATH=.....;C:\ProgramFiles\MySQL\MySQLServer5.6\bin(注意是追加,不是覆盖)



- (4) 配置完环境变量之后先别忙着启动 mysql，我们还需要修改一下配置文件（如果没有配置，之后启动的时候就会出现图中的错误哦！:错误 2 系统找不到文件），mysql-5.6.1X 默认的配置文件的在 C:\ProgramFiles\MySQL\MySQLServer5.6\my-default.ini，或者自己建立一个 my.ini 文件，在其中修改或添加配置（如图）：

[mysqld]

basedir=C:\ProgramFiles\MySQL\MySQLServer5.6 (mysql 所在目录)

datadir=C:\ProgramFiles\MySQL\MySQLServer5.6\data (mysql 所在目录\data)

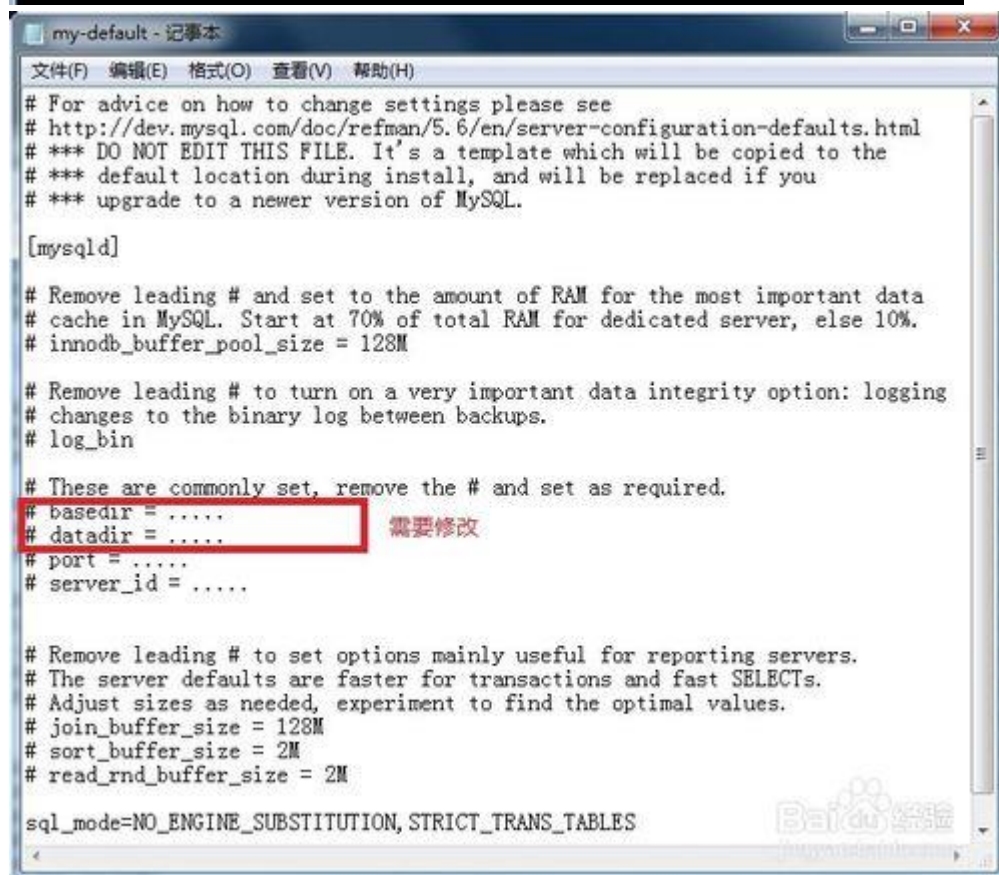
```
C:\Windows\system32>net start mysql
```

发生系统错误 2。

很纠结的问题，需要修

系统找不到指定的文件。

改配置文件



```
my-default - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

# For advice on how to change settings please see
# http://dev.mysql.com/doc/refman/5.6/en/server-configuration-defaults.html
# *** DO NOT EDIT THIS FILE. It's a template which will be copied to the
# *** default location during install, and will be replaced if you
# *** upgrade to a newer version of MySQL.

[mysqld]

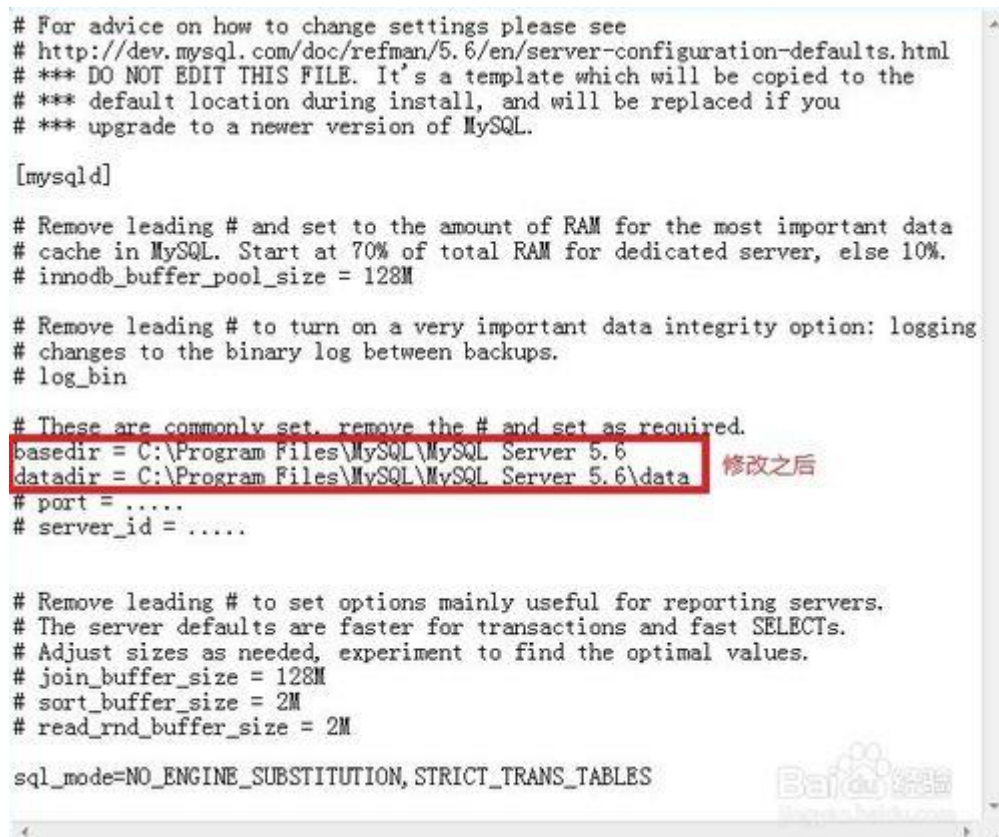
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M

# Remove leading # to turn on a very important data integrity option: logging
# changes to the binary log between backups.
# log_bin

# These are commonly set, remove the # and set as required.
# basedir = .....
# datadir = .....
# port = .....
# server_id = .....

# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M

sql_mode=NO_ENGINE_SUBSTITUTION,STRICT_TRANS_TABLES
```



```
my-default - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

# For advice on how to change settings please see
# http://dev.mysql.com/doc/refman/5.6/en/server-configuration-defaults.html
# *** DO NOT EDIT THIS FILE. It's a template which will be copied to the
# *** default location during install, and will be replaced if you
# *** upgrade to a newer version of MySQL.

[mysqld]

# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M

# Remove leading # to turn on a very important data integrity option: logging
# changes to the binary log between backups.
# log_bin

# These are commonly set, remove the # and set as required.
basedir = C:\Program Files\MySQL\MySQL Server 5.6
datadir = C:\Program Files\MySQL\MySQL Server 5.6\data
# port = .....
# server_id = .....

# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M

sql_mode=NO_ENGINE_SUBSTITUTION,STRICT_TRANS_TABLES
```

(5) 以管理员身份运行 cmd (一定要用管理员身份运行，不然权限不够)，

输入: `cd C:\ProgramFiles\MySQL\MySQLServer5.6\bin` 进入 mysql 的 bin 文件夹(不管有没有配置过环境变量, 也要进入 bin 文件夹, 否则之后启动服务仍然会报错误 2)

输入 **mysql -install**(如果不用管理员身份运行, 将会因为权限不够而出现错误: Install/RemoveoftheServiceDenied!)

安装成功



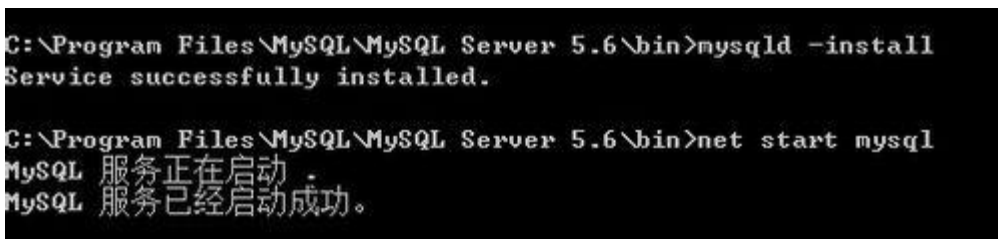
(6) 安装成功后就要启动服务了, 继续在 cmd 中输入:**net start mysql** (如图), 服务启动成功!

此时很多人会出现错误, 请看注意:

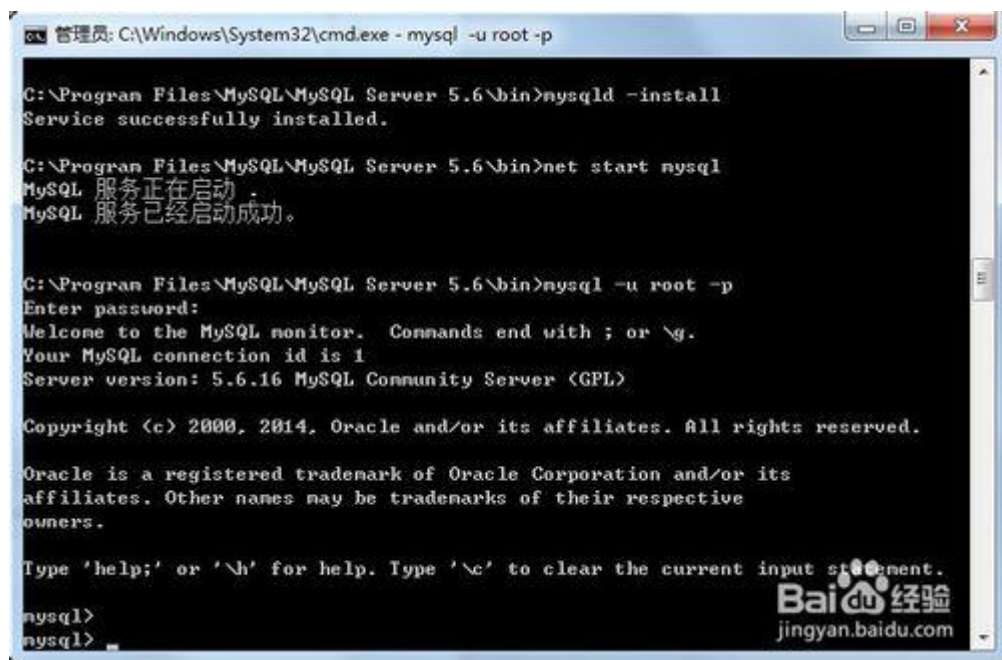
注意: 这个时候经常会出现错误 2 和错误 1067。

如果出现“错误 2 系统找不到文件”, 检查一下是否修改过配置文件或者是否进入在 bin 目录下操作, 如果配置文件修改正确并且进入了 bin 文件夹, 需要先删除 mysql (输入 `mysql -remove`) 再重新安装 (输入 `mysql -install`);

如果出现错误 1067, 那就是配置文件修改错误, 确认一下配置文件是否正确。



(7) 服务启动成功之后, 就可以登录了, 如图, 输入 `mysql -u root -p` (第一次登录没有密码, 直接按回车过), 登录成功!



END

注意事项

- my.ini 文件的编码必须是英文编码（如 windows 中的 ANSI），不能是 UTF-8 或 GBK 等。

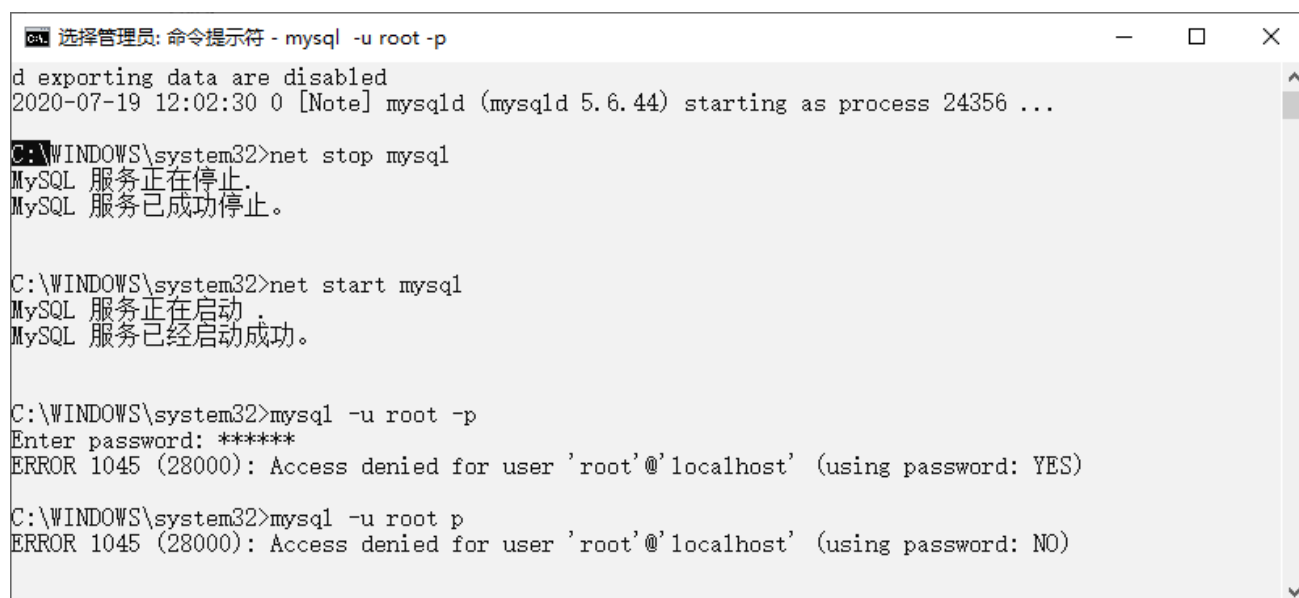
3. mysql 服务管理

(1) 启停服务

系统管理员身份运行 cmd

启动 mysql 服务：**net start mysql**

停止 mysql 服务：**net stop mysql**



(2) 修改密码

格式: **mysqladmin -u 用户名 -p password**

例: 给 root 用户设置密码 1234。

```
C:\WINDOWS\system32>mysqladmin -u root -p password
```

```
Enter password:
```

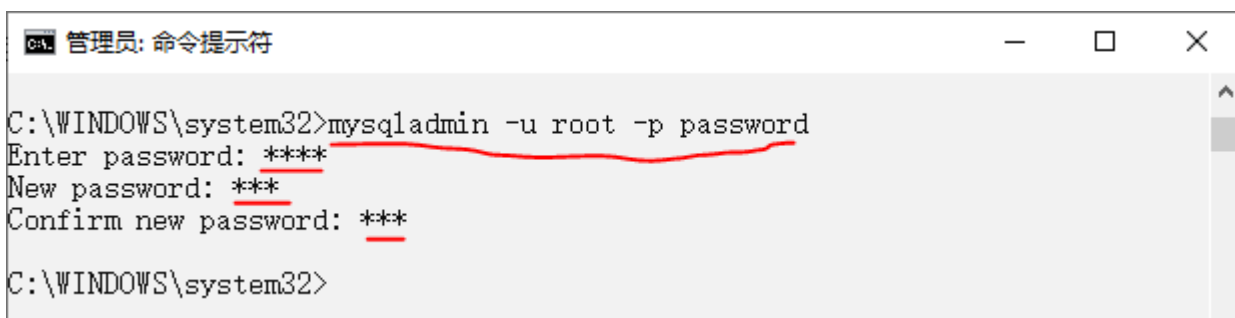
```
New password: ****
```

```
Confirm new password: ****
```

注: 因为开始时 root 没有密码, 所以旧密码一项就可以省略了。

例: 再将 root 的密码改为 123

mysqladmin -u root -p password



```
C:\WINDOWS\system32>mysqladmin -u root -p password
Enter password: ****
New password: ***
Confirm new password: ***
C:\WINDOWS\system32>
```

```
C:\WINDOWS\system32>mysqladmin -u root -p password
```

```
Enter password: ****
```

```
New password: ***
```

```
Confirm new password: ***
```

(3) 进入和退出 mysql 数据库控制台

进入 mysql 数据库控制台: **mysql -u root -p**

我的 mysql 账户: root 密码 123


```
管理员: 命令提示符 - mysql -u root -p
ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: NO)

C:\WINDOWS\system32>mysql -u root -p
Enter password: ***
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3
Server version: 5.6.44 MySQL Community Server (GPL)

Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.

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.

mysql>
```

List of all MySQL commands:

Note that all text commands must be first on line and end with ';'.

注意命令以分号结尾，例如: `mysql>exit;`

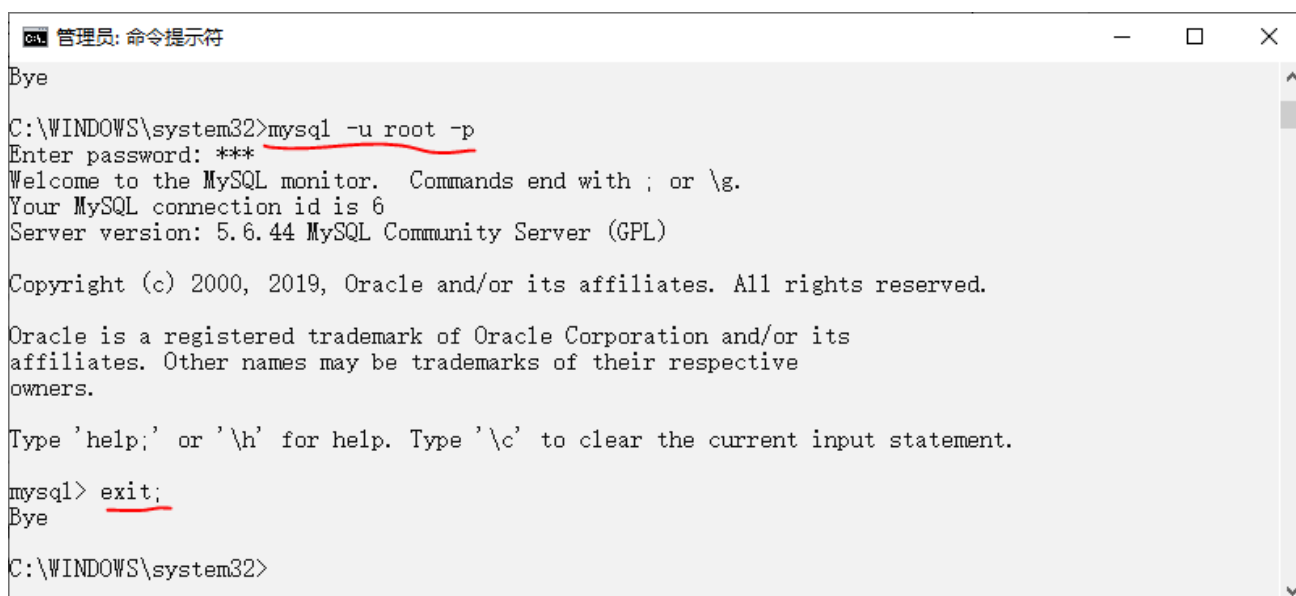
<code>?</code>	<code>(\?)</code> Synonym for <code>'help'</code> .
<code>clear</code>	<code>(\c)</code> Clear the current input statement.
<code>connect</code>	<code>(\r)</code> Reconnect to the server. Optional arguments are db and host.
<code>delimiter</code>	<code>(\d)</code> Set statement delimiter.
<code>ego</code>	<code>(\G)</code> Send command to mysql server, display result vertically.
<code>exit</code>	<code>(\q)</code> Exit mysql. Same as quit.
<code>go</code>	<code>(\g)</code> Send command to mysql server.
<code>help</code>	<code>(\h)</code> Display this help.
<code>notee</code>	<code>(\t)</code> Don't write into outfile.
<code>print</code>	<code>(\p)</code> Print current command.
<code>prompt</code>	<code>(\R)</code> Change your mysql prompt.
<code>quit</code>	<code>(\q)</code> Quit mysql.
<code>rehash</code>	<code>(\#)</code> Rebuild completion hash.
<code>source</code>	<code>(\.)</code> Execute an SQL script file. Takes a file name as an argument.
<code>status</code>	<code>(\s)</code> Get status information from the server.
<code>tee</code>	<code>(\T)</code> Set outfile [to_outfile]. Append everything into given outfile.
<code>use</code>	<code>(\u)</code> Use another database. Takes database name as argument.
<code>charset</code>	<code>(\C)</code> Switch to another charset. Might be needed for processing binlog with multi-byte charsets.
<code>warnings</code>	<code>(\W)</code> Show warnings after every statement.
<code>nowarning</code>	<code>(\w)</code> Don't show warnings after every statement.

For server side help, type `'help contents'`

退出 mysql 数据库控制台：**mysql>exit;**

或者：**mysql>quit;**

或者：**mysql>\q**



```
管理员: 命令提示符
Bye
C:\WINDOWS\system32>mysql -u root -p
Enter password: ***
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 6
Server version: 5.6.44 MySQL Community Server (GPL)

Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.

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

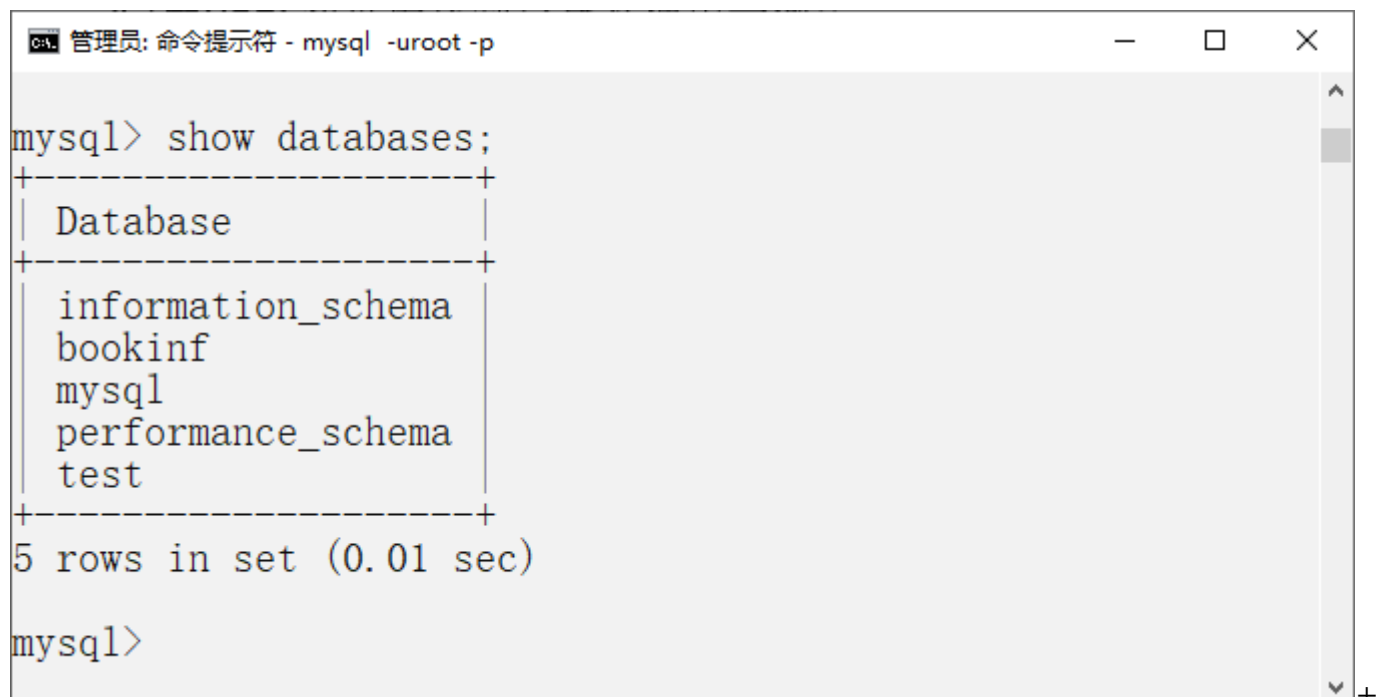
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> exit;
Bye
C:\WINDOWS\system32>
```

4. mysql 数据库控制台命令使用示例

(1) 列出数据库

mysql> **show databases;**



```
管理员: 命令提示符 - mysql -uroot -p

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| bookinf          |
| mysql            |
| performance_schema |
| test              |
+-----+
5 rows in set (0.01 sec)

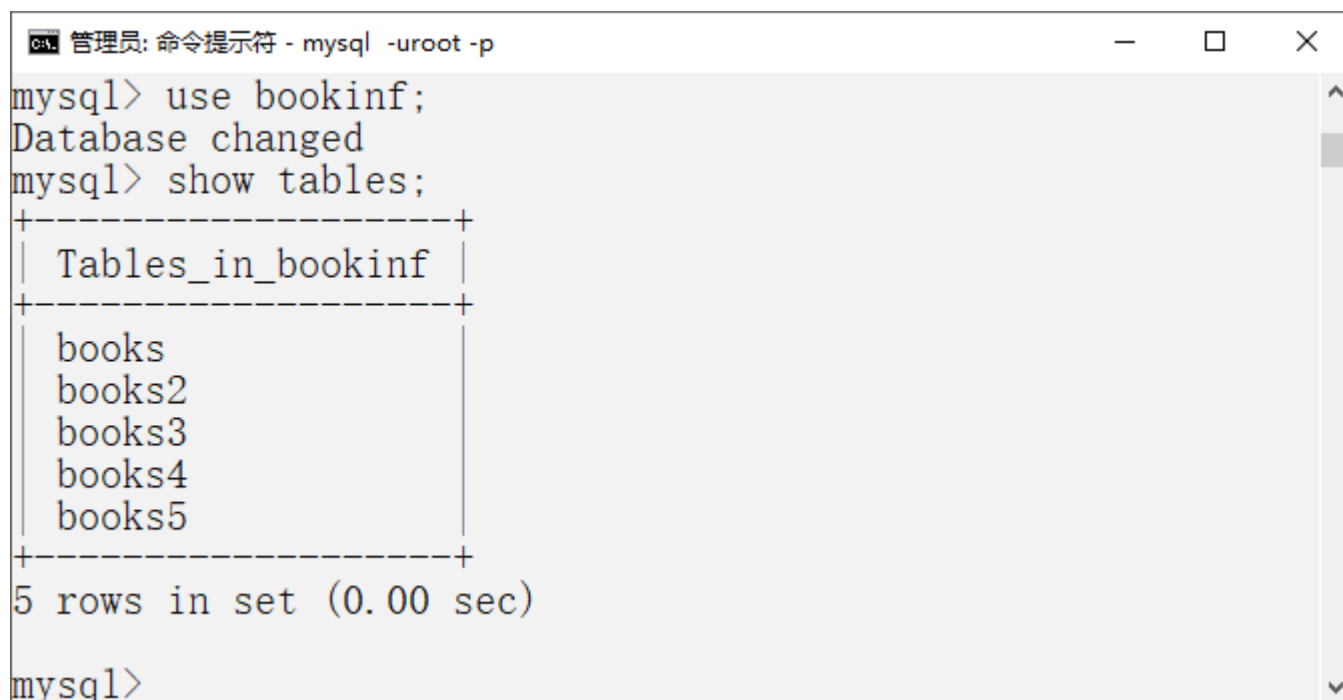
mysql>
```

(2) 选择某个数据库

mysql> **use bookinf;**

(3) 列出数据库中的表

mysql> **show tables;**

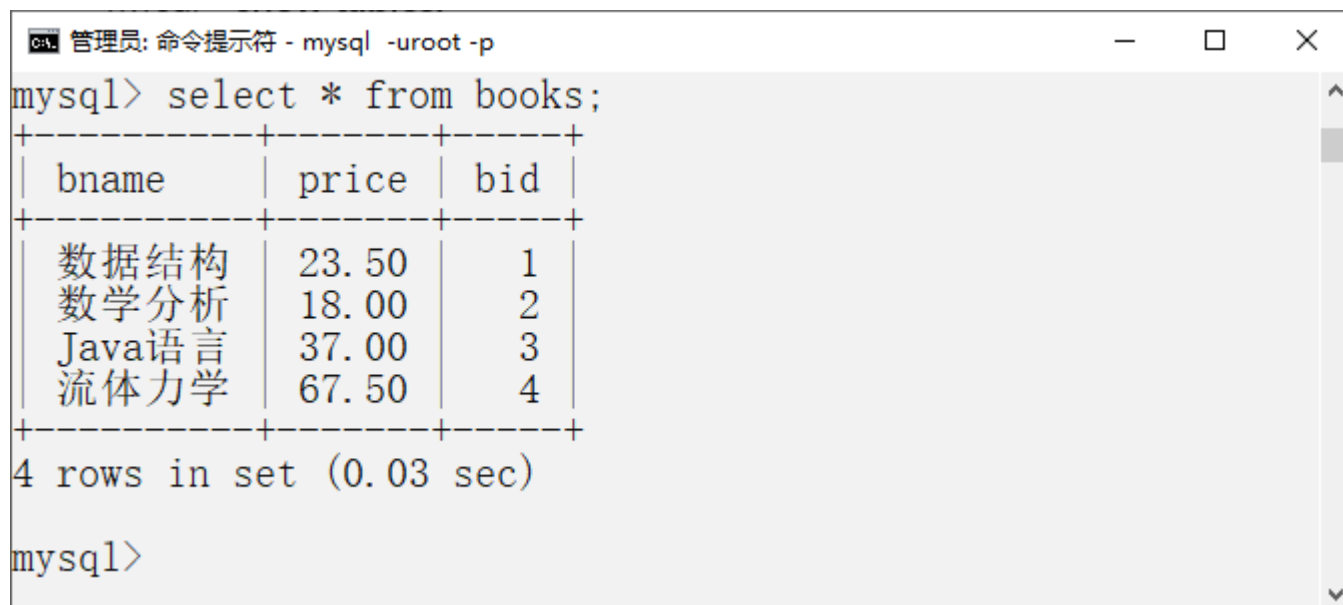


```
管理员: 命令提示符 - mysql -uroot -p
mysql> use bookinf;
Database changed
mysql> show tables;
+-----+
| Tables_in_bookinf |
+-----+
| books              |
| books2             |
| books3             |
| books4             |
| books5             |
+-----+
5 rows in set (0.00 sec)

mysql>
```

(4) 查询表中数据

mysql> **select * from books;**



```
管理员: 命令提示符 - mysql -uroot -p
mysql> select * from books;
+-----+-----+-----+
| bname   | price | bid |
+-----+-----+-----+
| 数据结构 | 23.50 | 1   |
| 数学分析 | 18.00 | 2   |
| Java语言 | 37.00 | 3   |
| 流体力学 | 67.50 | 4   |
+-----+-----+-----+
4 rows in set (0.03 sec)

mysql>
```

(5) 向表中插入记录

mysql> **insert into books(bid,bname,price) values(5,'Python',49);**

```
管理员: 命令提示符 - mysql -uroot -p
mysql> insert into books (bid, bname, price) values (5, 'Python', 49);
Query OK, 1 row affected (0.01 sec)

mysql> select * from books;
+-----+-----+-----+
| bname   | price | bid |
+-----+-----+-----+
| 数据结构 | 23.50 | 1   |
| 数学分析 | 18.00 | 2   |
| Java语言 | 37.00 | 3   |
| 流体力学 | 67.50 | 4   |
| Python  | 49.00 | 5   |
+-----+-----+-----+
5 rows in set (0.00 sec)

mysql>
```

(6) 更新表中某些记录

mysql> **update books set price=price*0.5 where price>40;**

```
管理员: 命令提示符 - mysql -uroot -p
mysql> update books set price=price*0.5 where price>40;
Query OK, 2 rows affected (0.02 sec)
Rows matched: 2  Changed: 2  Warnings: 0

mysql> select * from books;
+-----+-----+-----+
| bname   | price | bid |
+-----+-----+-----+
| 数据结构 | 23.50 | 1   |
| 数学分析 | 18.00 | 2   |
| Java语言 | 37.00 | 3   |
| 流体力学 | 33.75 | 4   |
| Python  | 24.50 | 5   |
+-----+-----+-----+
5 rows in set (0.00 sec)
```

(6) 删除表中指定记录

mysql> **delete from books where bname like '%力学';**

```
管理员: 命令提示符 - mysql -uroot -p

+-----+-----+-----+
| bname | price | bid |
+-----+-----+-----+
| 数据结构 | 23.50 | 1 |
| 数学分析 | 18.00 | 2 |
| Java语言 | 37.00 | 3 |
| 流体力学 | 33.75 | 4 |
| Python | 24.50 | 5 |
+-----+-----+-----+
5 rows in set (0.00 sec)

mysql> delete from books where bname like '%力学';
Query OK, 1 row affected (0.00 sec)

mysql> select * from books;
+-----+-----+-----+
| bname | price | bid |
+-----+-----+-----+
| 数据结构 | 23.50 | 1 |
| 数学分析 | 18.00 | 2 |
| Java语言 | 37.00 | 3 |
| Python | 24.50 | 5 |
+-----+-----+-----+
4 rows in set (0.00 sec)
```

(7) 查询排序

```
mysql> select * from books order by price;
```

```
管理员: 命令提示符 - mysql -uroot -p

+-----+-----+-----+
| bname  | price | bid  |
+-----+-----+-----+
| 数据结构 | 23.50 | 1    |
| 数学分析 | 18.00 | 2    |
| Java语言 | 37.00 | 3    |
| Python  | 24.50 | 5    |
+-----+-----+-----+
4 rows in set (0.00 sec)

mysql> select * from books order by price;

+-----+-----+-----+
| bname  | price | bid  |
+-----+-----+-----+
| 数学分析 | 18.00 | 2    |
| 数据结构 | 23.50 | 1    |
| Python  | 24.50 | 5    |
| Java语言 | 37.00 | 3    |
+-----+-----+-----+
4 rows in set (0.00 sec)

mysql>
```

(8) 统计查询

mysql>**select count(*)as recordCount,max(price),min(price),avg(price) from books;**

```
管理员: 命令提示符 - mysql -uroot -p

mysql> select count(*)as recordCount,max(price),min(price),avg(price) from books;

+-----+-----+-----+-----+
| recordCount | max(price) | min(price) | avg(price) |
+-----+-----+-----+-----+
| 4           | 37.00      | 18.00      | 25.750000  |
+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql>
```

二、使用 Navicat 管理数据库

1. 下载 Navicat

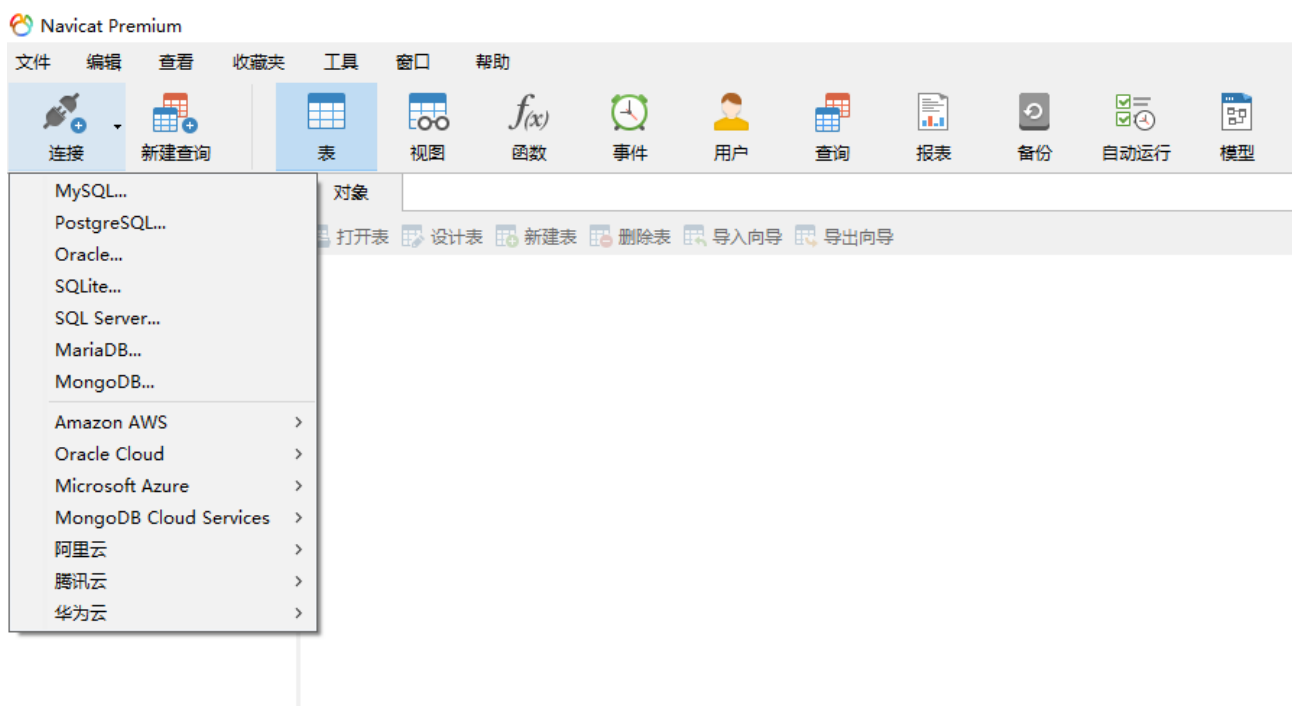


www.navicat.com.cn/download/navicat-for-mysql



<https://www.navicat.com.cn/download/navicat-premium>

navicat-premium 功能更加强大，可以连接管理各种数据库！



下面以 navicat-for-mysql 为例，说明 navicat 的安装是使用。



Windows

Navicat for MySQL 版本 12.1



macOS

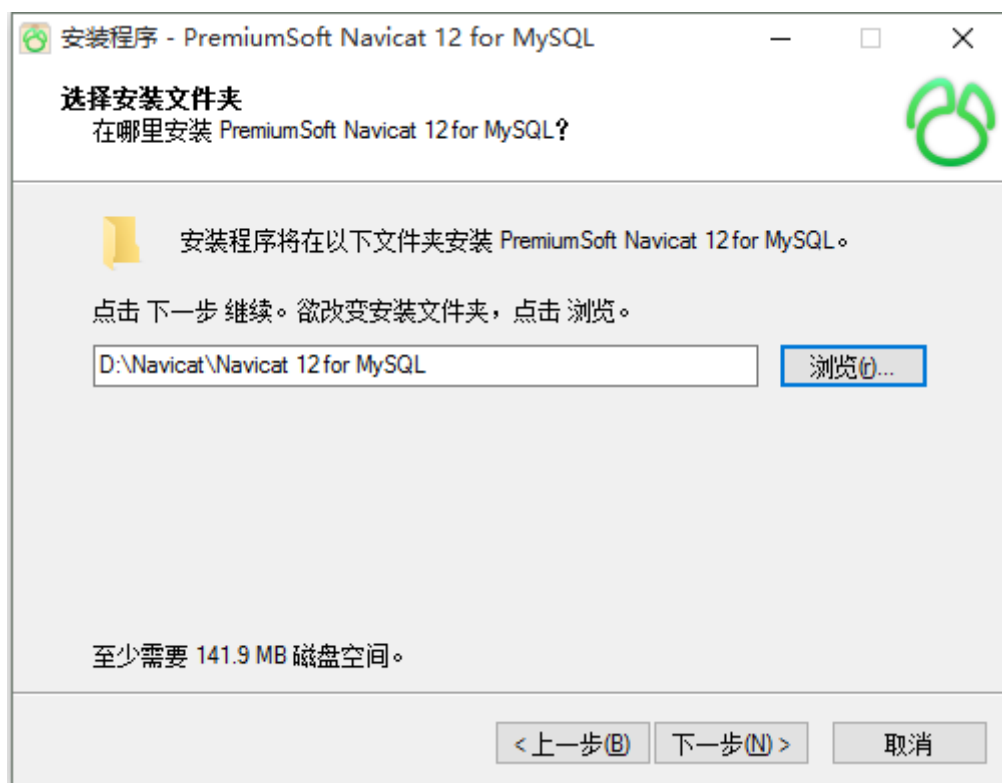
Navicat for MySQL 版本 12.1



2. 安装 Navicat



下一步~选择路径~

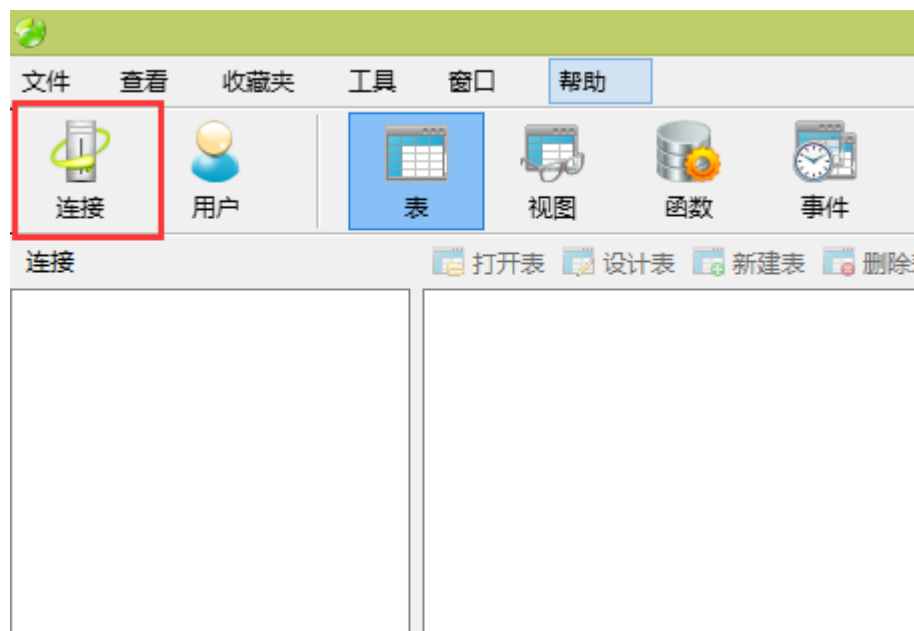


之后不用管直接下一步直接完成~

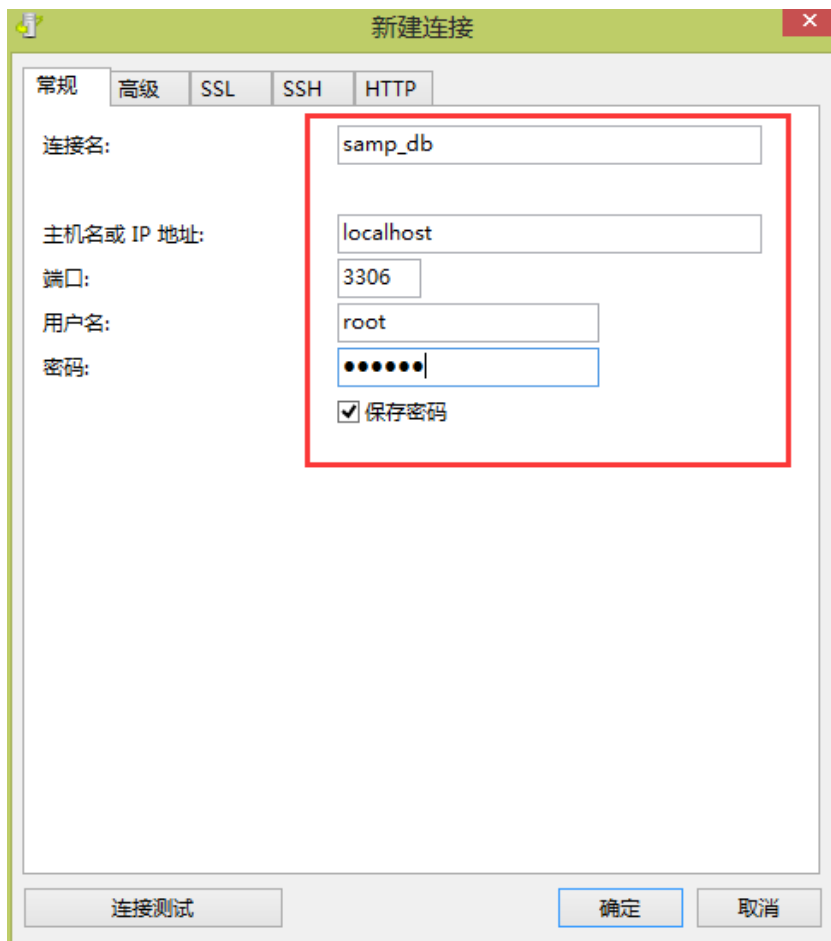
3. 运行并使用 Navicat

确保已经安装 mysql 数据库和 navicatformysql，打开该程序；

进入程序后点击连接



弹出窗口，创建新连接，填写连接数据库的相关参数，连接名随意，输入 root 帐号的密码，并勾选保存密码，点击连接测试，测试通过后点击确定：



确定后，双击窗口的连接栏的该连接：

端口号默认为 3306，必须与 mysql 的端口号一直，mysql 端口号在 my.ini 配置文件中设置，默认端口号也是 3306。

```
# For advice on how to change settings please see
# http://dev.mysql.com/doc/refman/5.6/en/server-configuration-defaults.html
# *** DO NOT EDIT THIS FILE. It's a template which will be copied to the
# *** default location during install, and will be replaced if you
# *** upgrade to a newer version of MySQL.

[client] #设置 mysql 客户端连接的默认字符集（若 my.ini 中没有[client]字段，添加上即可）
default_character_set=utf8
#character_set_client=utf8
#character_set_connection=utf8
#character_set_results=utf8

# 设置 mysql 客户端连接服务端时默认使用的端口
port=3306

[mysql] #（若 my.ini 中没有[mysql]字段，添加上即可）
# 设置 mysql 客户端默认字符集
default_character_set=utf8

[mysqld]
```

```
# Remove leading # and set to the amount of RAM for the most important data
# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.
# innodb_buffer_pool_size = 128M

# Remove leading # to turn on a very important data integrity option: logging
# changes to the binary log between backups.
# log_bin

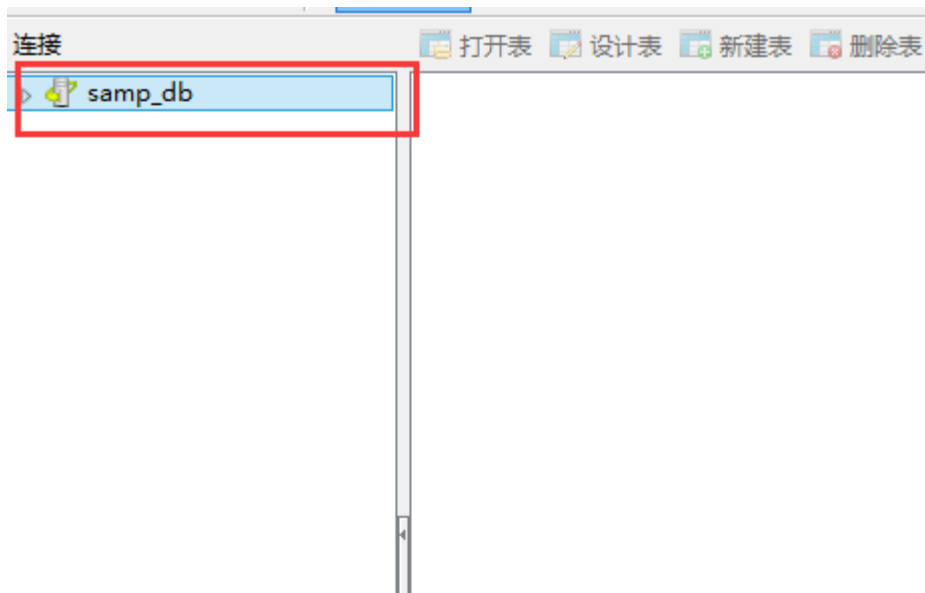
# These are commonly set, remove the # and set as required.
basedir = C:\\mysql-5.6.44
datadir = C:\\mysql-5.6.44\\data
port = 3306
# server_id = .....
# 服务端使用的字符集默认为 UTF8
character_set_server = utf8
#default_character_set=utf8
#character_set_database=utf8

default_storage_engine=INNODB # 创建新表时将使用的默认存储引擎

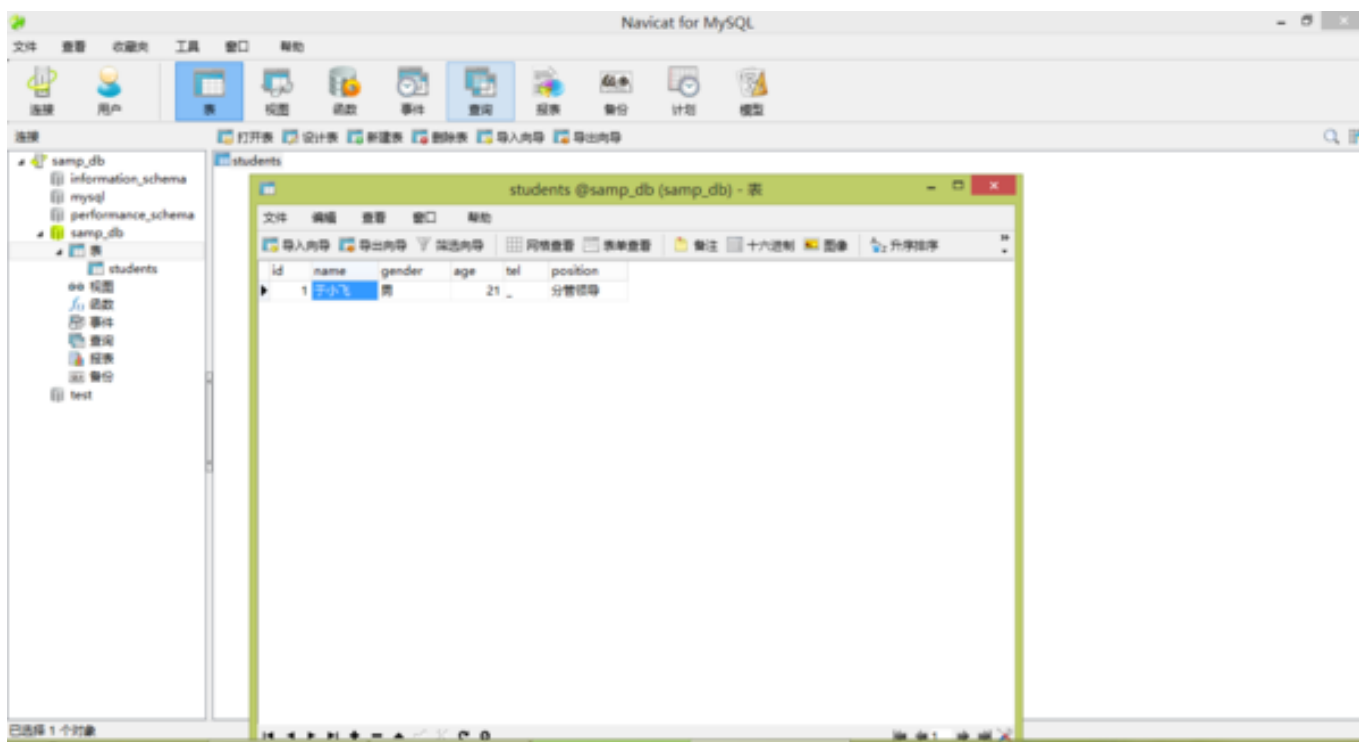
# 允许最大连接数
max_connections=200
# 允许连接失败的次数。这是为了防止有人从该主机试图攻击数据库系统
max_connect_errors=10

# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M

# SQL 模式为 strict 模式
sql-mode="STRICT_TRANS_TABLES,NO_AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION"
#sql_mode=NO_ENGINE_SUBSTITUTION,STRICT_TRANS_TABLES
```

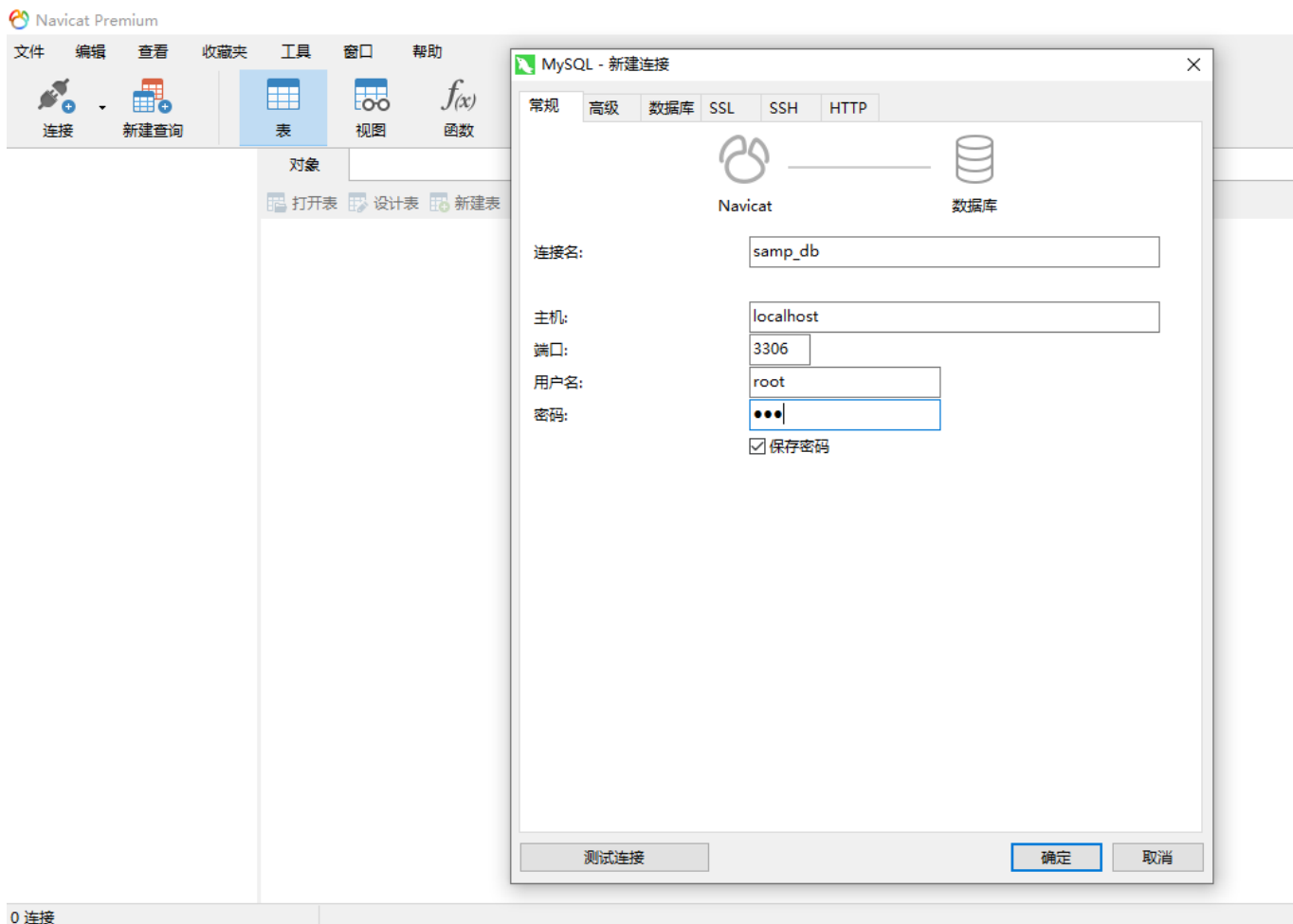


如此即可对该数据库进行各项内容管理了：

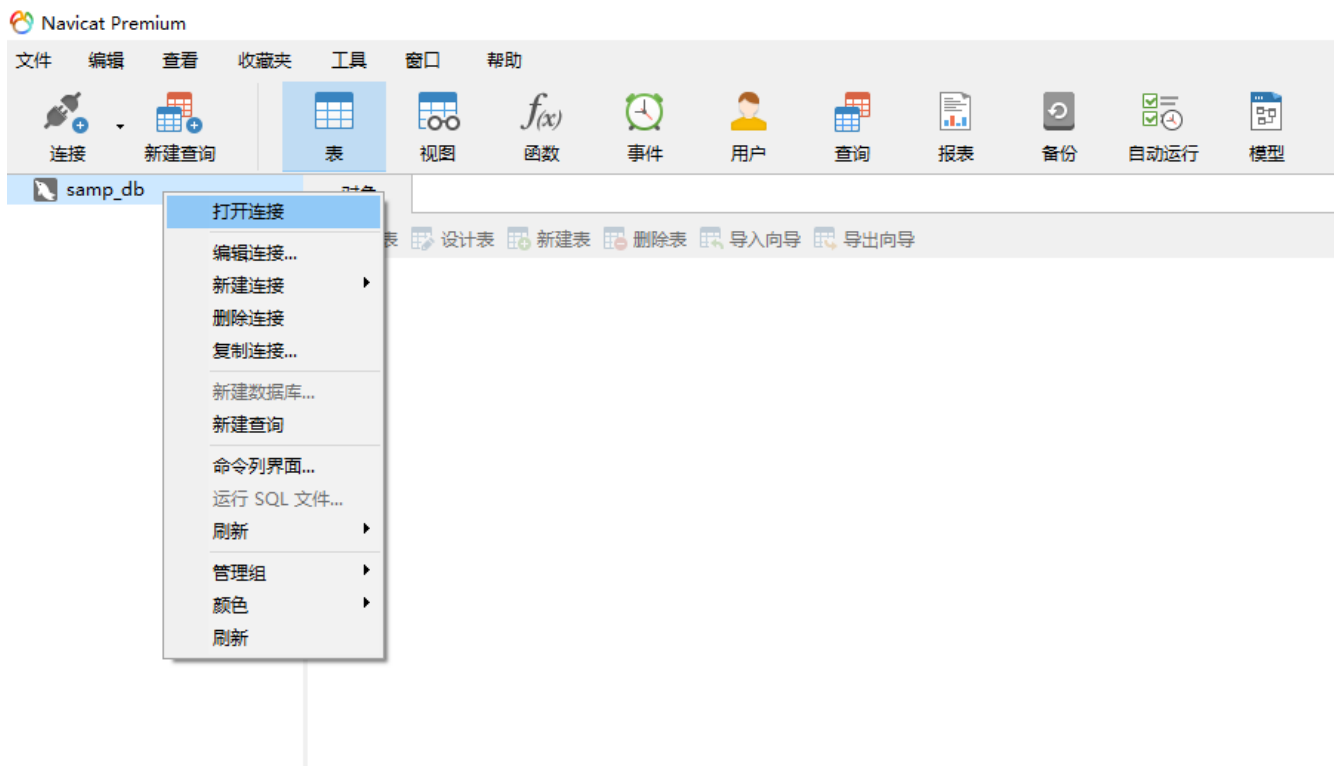


下面是 navicat-premium 连接到 MySQL 数据库的操作示图：

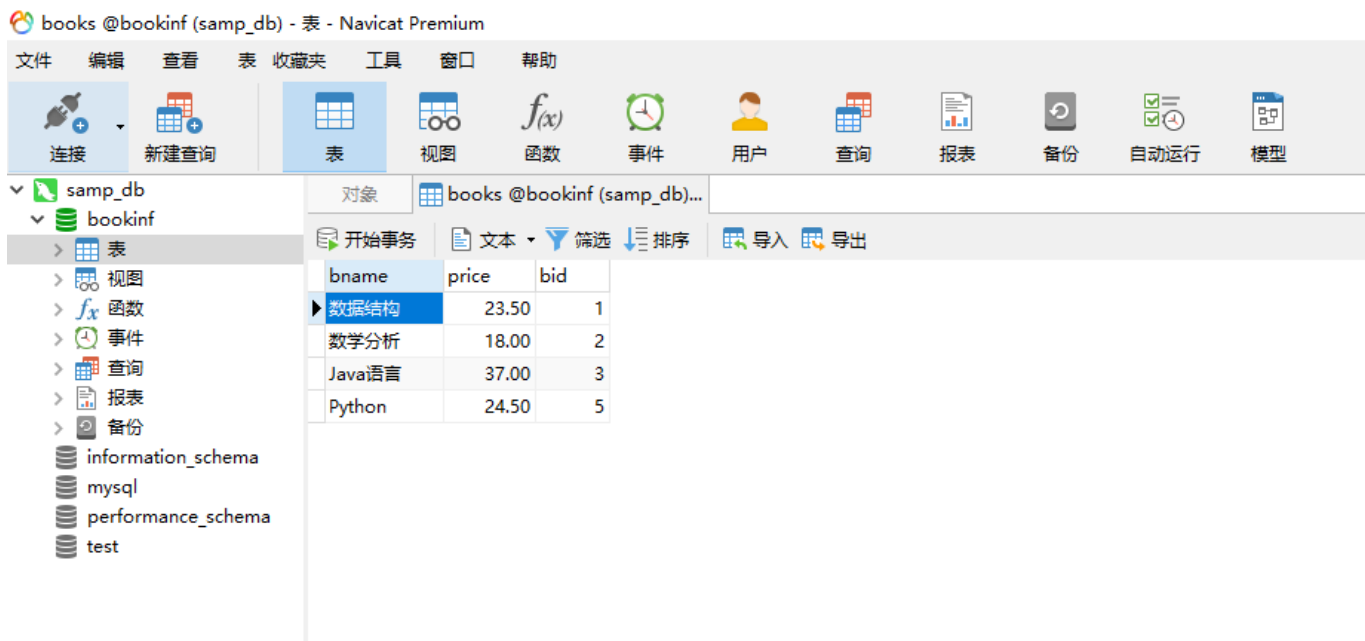
(a) 新建连接



(b) 打开连接



(c) 显示某个数据库中的某个表



三、Python 中访问 MySQL 数据库

1. mysql-connector 驱动

mysql-connector 用来连接使用 MySQL，它是 MySQL 官方提供的驱动器。

我们可以使用 **pip** 命令来安装 **mysql-connector** 库:

```
python -m pip install mysql-connector
```

A screenshot of a Windows Command Prompt window titled "管理员: 命令提示符". The window shows the execution of the command "python -m pip install mysql-connector". The output indicates that the package is being collected and downloaded from the internet. A progress bar is shown for the download of "mysql-connector-2.2.9.tar.gz (11.9 MB)", which is currently at approximately one-third completion. The download speed is 84 kB/s and the estimated time remaining is 0:01:36.

```
C:\>python -m pip install mysql-connector
```

Microsoft Windows [版本 10.0.18362.900]
(c) 2019 Microsoft Corporation。保留所有权利。

C:\WINDOWS\system32>python -m pip install mysql-connector
Collecting mysql-connector
 Downloading mysql-connector-2.2.9.tar.gz (11.9 MB)
 |■■■■■■■■■■■■■■■■■■■■| 3.9 MB 84 kB/s eta 0:01:36

```

C:\WINDOWS\system32> pip install mysql-connector
Created wheel for mysql-connector: filename=mysql_connector-2.2.9-cp38-cp38-win_amd64.whl size=247946 sha256=d69399792917f4ab9ac02f192f2f5fb974a4f3115ce053a0a8c69457a13e7d79
Stored in directory: c:\users\ruanz\appdata\local\pip\cache\wheels\57\e4\98\5feafb5c393dd2540e44b064a6f95832990d543e5b4f53ea8f
Successfully built mysql-connector
Installing collected packages: mysql-connector
Successfully installed mysql-connector-2.2.9
C:\WINDOWS\system32>

```

使用以下代码测试 `mysql-connector` 是否安装成功：

```
from mysql import connector
```

执行以上代码，如果没有产生错误，表明安装成功。

2. 创建数据库连接

可以使用以下代码来连接数据库服务器：

```
from mysql import connector
mydb = connector.connect(
    host="localhost", # 数据库主机地址
    user="root",      # 数据库用户名
    passwd="123"      # 数据库密码
)
print(mydb)

# 输出 <mysql.connector.connection.MySQLConnection object at 0x0000029ED6C49640>
```

3. 创建 `cursor`

```
mycursor = mydb.cursor()
```

4. 关闭数据库连接

```
mydb.close()
```

数据库访问完成后，应当及时关闭数据库连接，以释放珍贵的数据库资源。

5. 执行 `mysql` 命令

(1) 创建数据库

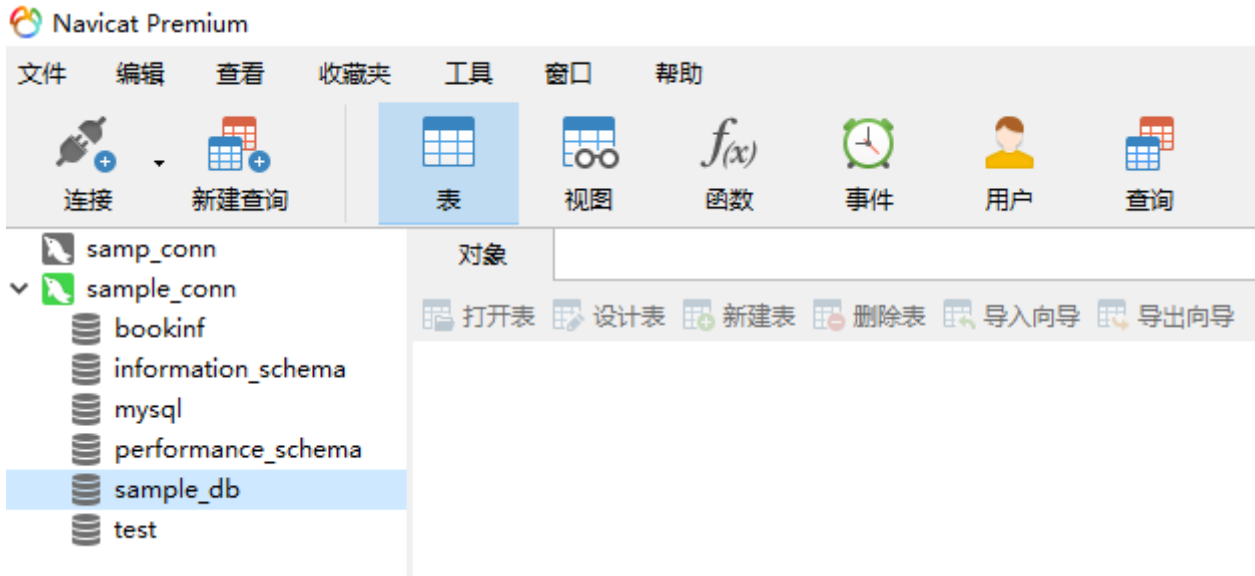
创建数据库使用 "**CREATE DATABASE**" 语句，以下创建一个名为 `samp2_db` 的数据库：

```
from mysql import connector
mydb = connector.connect(host="localhost", user="root", passwd="123"
    " ")
mycursor = mydb.cursor()
mycursor.execute("CREATE DATABASE sample_db")
mydb.close()
```

创建数据库前我们也可以使用 **"SHOW DATABASES"** 语句来查看数据库是否存在：

```
mycursor.execute("SHOW DATABASES")
for x in mycursor: #输出所有数据库列表
    print(x)
```

在 Navicat 中也可以看到，新增了一个数据库 sample_db:



也可以直接连接服务器上的某个数据库，如果数据库不存在，在则会输出错误信息。

```
from mysql import connector
mydb = connector.connect(
    host="localhost", # 数据库主机地址
    user="root",      # 数据库用户名
    passwd="123",     # 数据库密码
    database="bookinf" # 数据库
)
```

(2) 创建数据表

创建数据表使用 **"CREATE TABLE"** 语句，创建数据表前，需要确保数据库已存在，以下创建一个名为 **sites** 的数据表：

```
from mysql import connector
mydb = connector.connect(host="localhost", user="root", passwd="123",
    database="sample_db" )
mycursor = mydb.cursor()
mycursor.execute("CREATE TABLE sites (name VARCHAR(255), url VARCHAR(255))")
```

```
)  
mydb.close()
```

MySQL 数据类型请参见附录。

执行成功后，我们可以在 Navicat 中看到数据库创建的数据表 `sites`，字段为 `name` 和 `url`。



我们也可以使用 "SHOW TABLES" 语句来查看数据表是否已存在：

```
mycursor.execute("SHOW TABLES")  
for x in mycursor:  
    print(x)
```

执行代码，输出结果为：

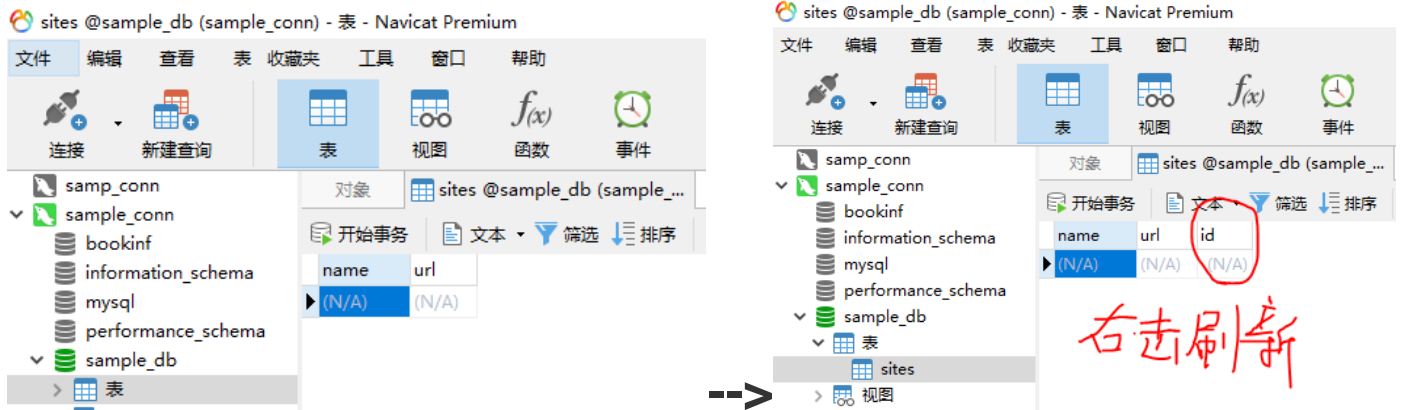
```
('sites',)
```

(3) 主键设置

创建表的时候我们一般都会设置一个主键 (PRIMARY KEY)，我们可以使用 "INT AUTO_INCREMENT PRIMARY KEY" 语句来创建一个主键，主键起始值为 1，逐步递增。

如果我们的表已经创建，我们需要使用 ALTER TABLE 来给表添加主键：

```
mycursor.execute("ALTER TABLE sites ADD COLUMN id INT AUTO_INCREMENT  
PRIMARY KEY")
```



如果尚未创建 `sites` 表，可以直接使用以下代码创建：

```
mycursor.execute("CREATE TABLE sites (id INT AUTO_INCREMENT PRIMARY KEY, name VARCHAR(255), url VARCHAR(255))")
```

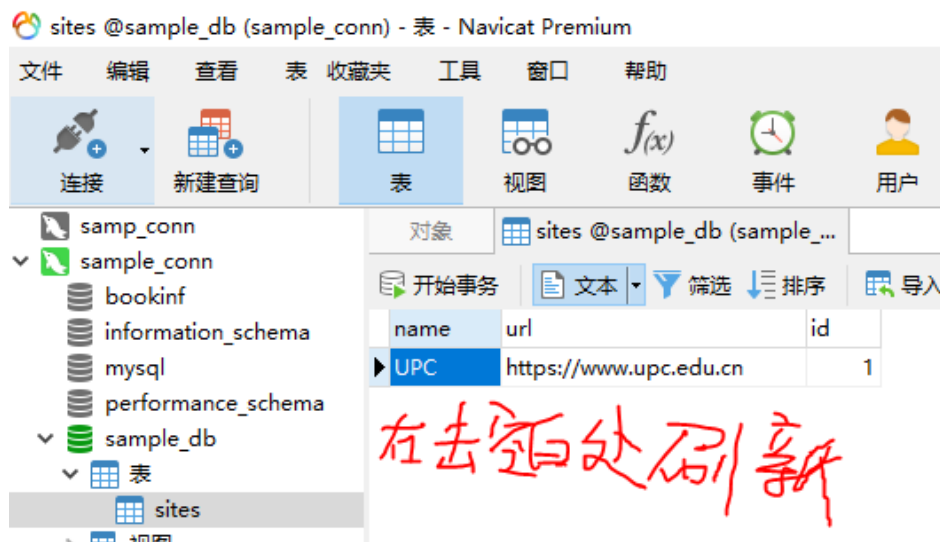
(4) 插入数据

插入数据使用 “INSERT INTO” 语句：

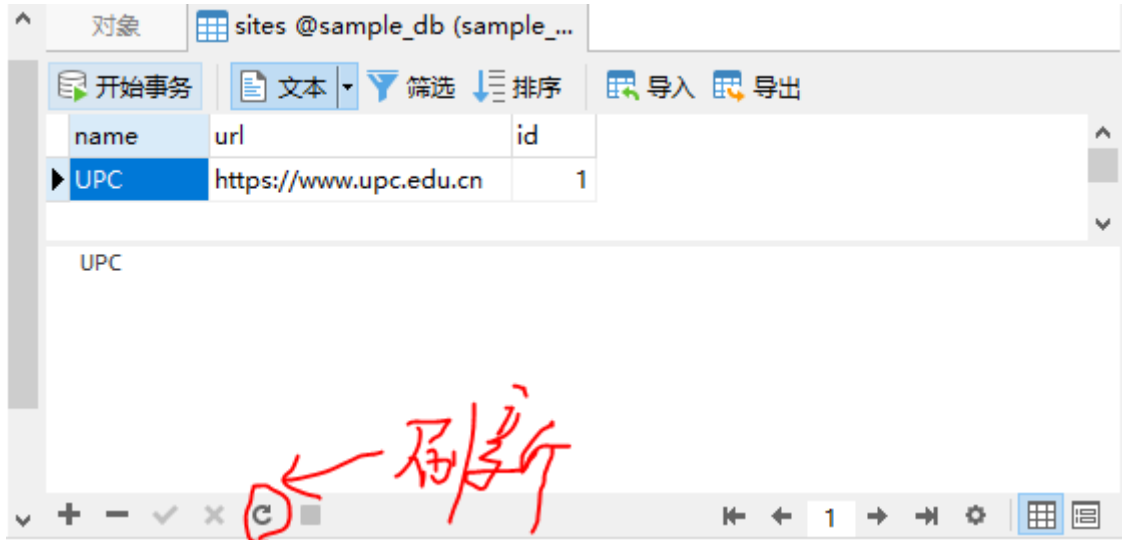
```
sql = "INSERT INTO sites (name, url) VALUES (%s, %s)"
val = ("UPC", "https://www.upc.edu.cn")
mycursor.execute(sql, val) # 给 SQL 语句传递两个参数
mydb.commit() # 数据表内容有更新，必须使用到该语句
print(mycursor.rowcount, "记录插入成功。")
```

执行代码，输出结果为：

1 记录插入成功



或者点击数据表下方的“刷新”按钮：



(5) 批量插入

批量插入使用 `executemany()` 方法，该方法的第二个参数是一个元组列表，包含了要插入的数据。

```
sql = "INSERT INTO sites (name, url) VALUES (%s, %s)"
val = [ ('Baidu', 'https://www.Baidu.com'),
        ('Github', 'https://www.github.com'),
        ('JD', 'https://www.JD.com'),
        ('Python', 'https://www.python.org/') ]

mycursor.executemany(sql, val)

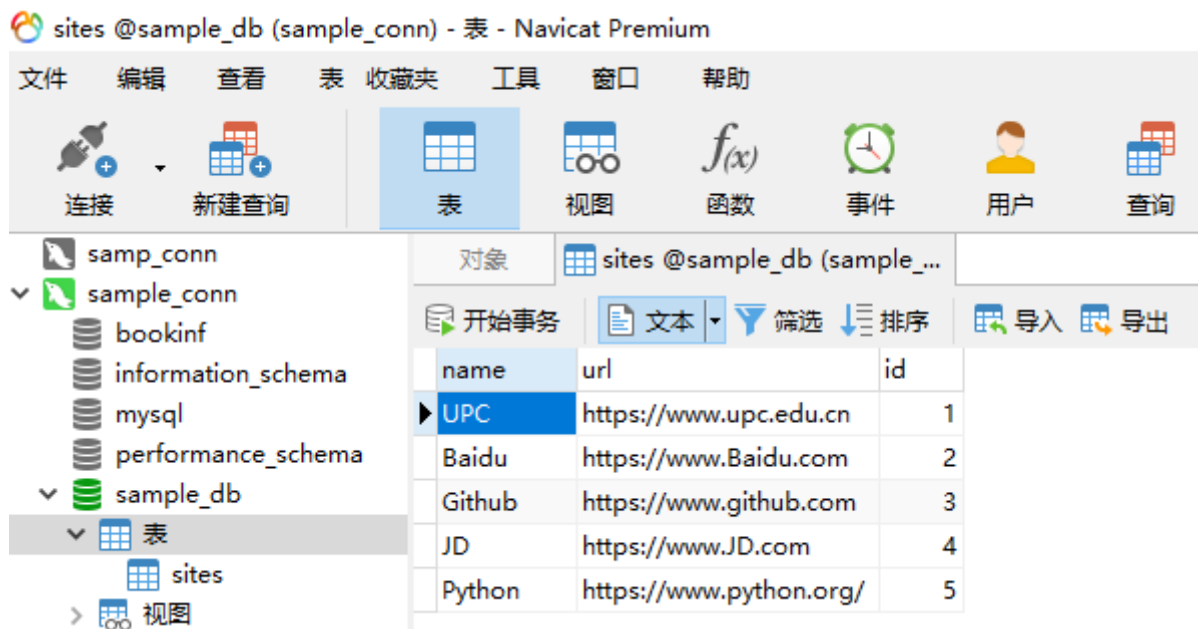
mydb.commit() # 数据表内容有更新，必须使用到该语句

print(mycursor.rowcount, "记录插入成功。")
```

执行代码，输出结果为：

4 记录插入成功。

执行以上代码后，我们可以看看数据表的记录：



如果我們想在數據記錄插入後，獲取該記錄的 ID，可以使用以下代碼：

```
sql = "INSERT INTO sites (name, url) VALUES (%s, %s)"
val = ("Zhihu", "https://www.zhihu.com")
mycursor.execute(sql, val)
mydb.commit()
print("1 條記錄已插入，ID:", mycursor.lastrowid)
```

執行代碼，輸出結果為：

1 條記錄已插入，ID: 6

(6) 查詢數據

查詢數據使用 SELECT 語句：

```
mycursor.execute("SELECT * FROM sites")
myresult = mycursor.fetchall() # fetchall() 獲取所有記錄，並以元組列表形式返回
for x in myresult:
    print(x)
```

執行代碼，輸出結果為：

```
('UPC', 'https://www.upc.edu.cn', 1)
('Baidu', 'https://www.Baidu.com', 2)
('Github', 'https://www.github.com', 3)
('JD', 'https://www.JD.com', 4)
('Python', 'https://www.python.org/', 5)
('Zhihu', 'https://www.zhihu.com', 7)
```

也可以读取指定的字段数据:

```
mycursor.execute("SELECT name, url FROM sites")

myresult = mycursor.fetchall() # fetchall() 获取所有记录, 并以元组列表形式返回

for x in myresult:

    print(x)
```

执行代码, 输出结果为:

```
('UPC', 'https://www.upc.edu.cn')
('Baidu', 'https://www.Baidu.com')
('Github', 'https://www.github.com')
('JD', 'https://www.JD.com')
('Python', 'https://www.python.org/')
('Zhihu', 'https://www.zhihu.com')
```

如果我们只想读取一条数据, 可以使用 `fetchone()` 方法:

```
mycursor.execute("SELECT * FROM sites")

myresult = mycursor.fetchone() # fetchone() 获取第一条记录, 并以元组形式返回

print(myresult)
```

执行代码, 输出结果为:

```
('UPC', 'https://www.upc.edu.cn', 1)
```

(7) where 条件语句

如果要读取指定条件的数据, 可以使用 `where` 语句:

```
mycursor.execute("SELECT * FROM sites WHERE name = 'Baidu'")

myresult = mycursor.fetchall()

for x in myresult:
```

```
print(x)
```

执行代码，输出结果为：

```
('UPC', 'https://www.upc.edu.cn', 1)
```

也可以使用通配符 % 进行模糊查询：

```
mycursor.execute("SELECT * FROM sites WHERE url LIKE '%th%'")
myresult = mycursor.fetchall()
for x in myresult:
    print(x)
```

执行代码，输出结果为：

```
('Github', 'https://www.github.com', 3)
('Python', 'https://www.python.org/', 5)
```

(8) 排序

查询结果排序可以使用 ORDER BY 语句，默认的排序方式为升序，关键字为 ASC，如果要设置降序排序，可以设置关键字 DESC。

按 name 字段字母的升序排序：

```
mycursor.execute("SELECT * FROM sites ORDER BY name")
myresult = mycursor.fetchall()
for x in myresult:
    print(x)
```

执行代码，输出结果为：

```
('Baidu', 'https://www.Baidu.com', 2)
('Github', 'https://www.github.com', 3)
('JD', 'https://www.JD.com', 4)
('Python', 'https://www.python.org/', 5)
('UPC', 'https://www.upc.edu.cn', 1)
('Zhihu', 'https://www.zhihu.com', 6)
```

按 name 字段字母的降序排序：

```
mycursor.execute("SELECT * FROM sites ORDER BY name DESC")

myresult = mycursor.fetchall()

for x in myresult:

    print(x)
```

执行代码，输出结果为：

```
('Zhihu', 'https://www.zhihu.com', 6)
('UPC', 'https://www.upc.edu.cn', 1)
('Python', 'https://www.python.org/', 5)
('JD', 'https://www.JD.com', 4)
('Github', 'https://www.github.com', 3)
('Baidu', 'https://www.Baidu.com', 2)
```

(9) Limit

如果我们要设置查询的数据量，可以通过 “LIMIT” 语句来指定。

读取前 3 条记录：

```
mycursor.execute("SELECT * FROM sites ORDER BY name LIMIT 3 ")

myresult = mycursor.fetchall()

for x in myresult:

    print(x)
```

执行代码，输出结果为：

```
('Zhihu', 'https://www.zhihu.com', 6)
('UPC', 'https://www.upc.edu.cn', 1)
('Python', 'https://www.python.org/', 5)
```

也可以指定起始位置，使用的关键字是 OFFSET。

从第二条开始读取前 3 条记录：

```
mycursor.execute("SELECT * FROM sites ORDER BY name LIMIT 3 OFFSET
1") # 0 为 第一条, 1 为第二条, 以此类推

myresult = mycursor.fetchall()

for x in myresult:

    print(x)
```

执行代码，输出结果为：

```
('UPC', 'https://www.upc.edu.cn', 1)
('Python', 'https://www.python.org/', 5)
('JD', 'https://www.JD.com', 4)
```

（10）删除记录

删除记录使用 "DELETE FROM" 语句。

删除 name 为 JD 的记录：

```
sql = "DELETE FROM sites WHERE name = 'JD'"
mycursor.execute(sql)
mydb.commit()
print(mycursor.rowcount, " 条记录删除")
```

执行代码，输出结果为：

1 条记录删除

注意：要慎重使用删除语句，删除语句要确保指定了 WHERE 条件语句，否则会导致整表数据被删除。

（11）更新表数据

数据表更新使用 "UPDATE" 语句。

将 name 为 Zhihu 的字段数据改为 ZH：

```
sql = "UPDATE sites SET name = 'ZH' WHERE name = 'Zhihu'"
mycursor.execute(sql)
mydb.commit()
print(mycursor.rowcount, " 条记录被修改")
```

执行代码，输出结果为：

1 条记录被修改

注意：UPDATE 语句要确保指定了 WHERE 条件语句，否则会导致整表数据被更新。

（12）删除表

删除表使用 "DROP TABLE" 语句， IF EXISTS 关键字是用于判断表是否存在，只有在存在的情况才删除。

```
sql = "DROP TABLE IF EXISTS sites" # 删除数据表 sites
```



```
mycursor.execute(sql)
```

6. 数据库访问综合示例：创建表、添加记录、查询记录

```
from mysql import connector
import sys

# 数据库访问信息
user = 'root'
pwd = '123'
host = '127.0.0.1'
db_name = 'bookinf'

# 连接至 mysql 数据库
conn = connector.connect(host=host, user=user, password=pwd,
database=db_name)
cursor = conn.cursor()

# 创建数据库表
sql1 = "drop table if exists person;" # python 向 mysql 传递的 SQL, 结尾分号可略去
sql2 = """
CREATE TABLE person(
    id int(10) AUTO_INCREMENT PRIMARY KEY,
    name varchar(20),
    age int(4)
)CHARACTER SET utf8;
"""

try:
    cursor.execute(sql1)
    cursor.execute(sql2)
except connector.Error as err:
    print("create table 'person' failed.")
    print(f"Error: {err.msg}")
    conn.close() # 关闭数据库连接
    sys.exit() # 退出程序

# 插入数据
sql3 = 'INSERT INTO person(name, age) VALUES (%s,%s)' # %s 为展位符
try:
```

```
cursor.execute(sql3, ('张三', 22)) # 插入一条
cursor.executemany(sql3, [('Smith', 26), ('李四', 24)]) # 插入两条
except connector.Error as err:
    print("insert table 'person' failed.")
    print(f"Error: {err.msg}")
    conn.close() # 关闭数据库连接
    sys.exit()

conn.commit() # 提交修改

# 查询数据
sql4 = "SELECT * FROM person"
try:
    cursor.execute(sql4)
    # 获取一条记录, 每条记录做为一个tuple(元组)返回
    records = cursor.fetchone()
    print(records)

    # 获取2条记录, 注意由于之前执行有了fetchone(), 所以游标已经指到第二条记录
    # 了, 也就是从第二条开始的2条记录
    records = cursor.fetchmany(2)
    print(records)

    cursor.execute(sql4)
    records = cursor.fetchall() # 获取所有结果
    print(records)

    # 获取所有结果, 并提取各字段
    cursor.execute(sql4)
    records = cursor.fetchall()
    for (ID, name, age) in records:
        print(f"ID:{ID} Name:{name} Age:{age}")

except connector.Error as err:
    print("query table 'person' failed.")
    print(f"Error: {err.msg}")
    conn.close() # 关闭数据库连接
    sys.exit()
```

```
cursor.close()  # 关闭游标
```

```
conn.close()  # 关闭数据库连接
```

四、附录：MySQL 数据类型

MySQL 中定义数据字段的类型对你数据库的优化是非常重要的。

MySQL 支持多种类型，大致可以分为三类：数值、日期/时间和字符串(字符)类型。

1. 数值类型

MySQL 支持所有标准 SQL 数值数据类型。

这些类型包括严格数值数据类型(INTEGER、SMALLINT、DECIMAL 和 NUMERIC)，以及近似数值数据类型(FLOAT、REAL 和 DOUBLE PRECISION)。

关键字 INT 是 INTEGER 的同义词，关键字 DEC 是 DECIMAL 的同义词。

BIT 数据类型保存位字段值，并且支持 MyISAM、MEMORY、InnoDB 和 BDB 表。

作为 SQL 标准的扩展，MySQL 也支持整数类型 TINYINT、MEDIUMINT 和 BIGINT。下面的表显示了需要的每个整数类型的存储和范围。

类型	大小	范围（有符号）	范围（无符号）	用途
TINYINT	1 byte	(-128, 127)	(0, 255)	小整数值
SMALLINT	2 bytes	(-32 768, 32 767)	(0, 65 535)	大整数值
MEDIUMINT	3 bytes	(-8 388 608, 8 388 607)	(0, 16 777 215)	大整数值
INT 或 INTEGER	4 bytes	(-2 147 483 648, 2 147 483 647)	(0, 4 294 967 295)	大整数值
BIGINT	8 bytes	(-9,223,372,036,854,775,808, 9 223 372 036 854 775 807)	(0, 18 446 744 073 709 551 615)	极大整数值
FLOAT	4 bytes	(-3.402 823 466 E+38, -1.175 494 351 E-38), 0, (1.175 494 823 466 E+38, 3.402 823 466 351 E+38)	0, (1.175 494 351 E-38, 3.402 823 466 E+38)	单精度浮点数值
DOUBLE	8 bytes	(-1.797 693 134 862 315 7 E+308, -2.225 073 858 507 201 4 E-308), 0, (2.225 073 858 507 201 4 E-308, 1.797 693 134 862 315 7 E+308)	0, (2.225 073 858 507 201 4 E-308, 1.797 693 134 862 315 7 E+308)	双精度浮点数值

		315 7 E+308)		
DECIMAL	对 DECIMAL(M,D) , 如果 M>D, 为 M+2 否则为 D+2	依赖于 M 和 D 的值	依赖于 M 和 D 的值	小数值

2. 日期和时间类型

表示时间值的日期和时间类型为 DATETIME、DATE、TIMESTAMP、TIME 和 YEAR。

每个时间类型有一个有效值范围和一个"零"值，当指定不合法的 MySQL 不能表示的值时使用"零"值。

TIMESTAMP 类型有专有的自动更新特性，将在后面描述。

类型	大小 (bytes)	范围	格式	用途
DATE	3	1000-01-01/9999-12-31	YYYY-MM-DD	日期值
TIME	3	'-838:59:59'/'838:59:59'	HH:MM:SS	时间值或持续时间
YEAR	1	1901/2155	YYYY	年份值
DATETIME	8	1000-01-01 00:00:00/9999-12-31 23:59:59	YYYY-MM-DD HH:MM:SS	混合日期和时间值
TIMESTAMP	4	1970-01-01 00:00:00/2038 结束时间是第 2147483647 秒，北京时间 2038-1-19 11:14:07 ，格林尼治时间 2038 年 1 月 19 日 凌晨 03:14:07	YYYYMMDD HHMMSS	混合日期和时间值， 时间戳

3. 字符串类型

字符串类型指 CHAR、VARCHAR、BINARY、VARBINARY、BLOB、TEXT、ENUM 和 SET。该节描述了这些类型如何工作以及如何在查询中使用这些类型。

类型	大小	用途
CHAR	0-255 bytes	定长字符串
VARCHAR	0-65535 bytes	变长字符串

TINYBLOB	0-255 bytes	不超过 255 个字符的二进制字符串
TINYTEXT	0-255 bytes	短文本字符串
BLOB	0-65 535 bytes	二进制形式的长文本数据
TEXT	0-65 535 bytes	长文本数据
MEDIUMBLOB	0-16 777 215 bytes	二进制形式的中等长度文本数据
MEDIUMTEXT	0-16 777 215 bytes	中等长度文本数据
LOBLOB	0-4 294 967 295 bytes	二进制形式的极大文本数据
LOBTEXT	0-4 294 967 295 bytes	极大文本数据

注意：char(n) 和 varchar(n) 中括号中 n 代表字符的个数，并不代表字节个数，比如 CHAR(30) 就可以存储 30 个字符。

CHAR 和 VARCHAR 类型类似，但它们保存和检索的方式不同。它们的最大长度和是否尾部空格被保留等方面也不同。在存储或检索过程中不进行大小写转换。

BINARY 和 VARBINARY 类似于 CHAR 和 VARCHAR，不同的是它们包含二进制字符串而不要非二进制字符串。也就是说，它们包含字节字符串而不是字符串。这说明它们没有字符集，并且排序和比较基于列值字节的数值值。

BLOB 是一个二进制大对象，可以容纳可变数量的数据。有 4 种 BLOB 类型：TINYBLOB、BLOB、MEDIUMBLOB 和 LOGBLOB。它们区别在于可容纳存储范围不同。

有 4 种 TEXT 类型：TINYTEXT、TEXT、MEDIUMTEXT 和 LONGTEXT。对应的这 4 种 BLOB 类型，可存储的最大长度不同，可根据实际情况选择。