01 MySQL数据库初识

MySQL**数据库初识**

MySQL数据库

本节目录

一 数据库概述

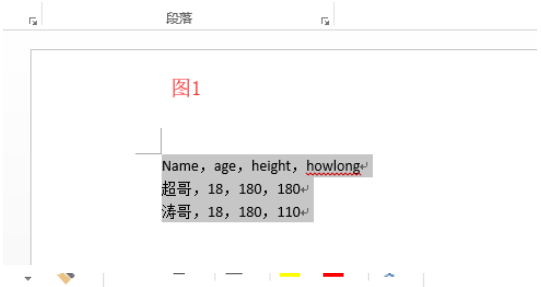
1. 数据库？？？

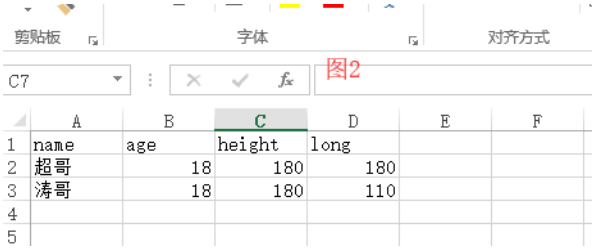
什么是数据库呢?

先来看看百度怎么说的

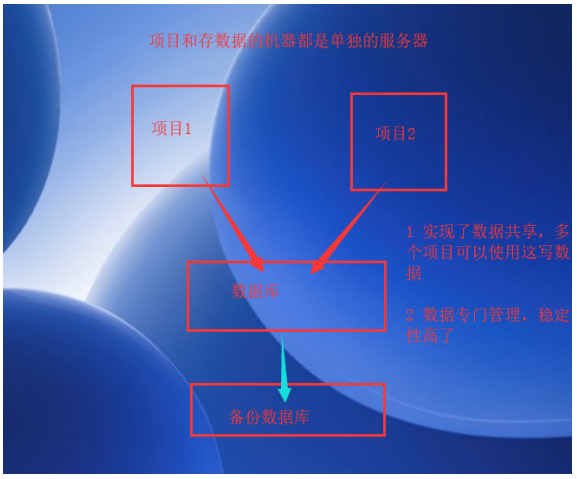
> 数据库，简而言之可视为电子化的文件柜——存储电子文件的处所，用户可以对文件中的数据运行新增、截取、更新、删除等操作。所谓“数据库”系以一定方式储存在一起、能予多个用户共享、具有尽可能小的冗余度、与应用程序彼此独立的数据集合。

百度的貌似不好理解啊，让我说啊，数据库是存储数据的地方，超哥，你这不是废话么？这位同学，你你你你你说的对，哈哈，存数据的地方是存在哪里呢，存在硬盘上，为什么不是存在内存里面，因为内存无法永久保存。之前我们存数据都是使用的文件，在一个word文档里面写一些羞羞的网址，然后保存，就存储到硬盘上了。有同学就会说了，超哥，我这通过文件不是也将数据保存上了吗？是的，没毛病，但是你想，通过文件来操作数据，效率是不是很低，首先打开关闭就比较慢，其次是我们操作起来也比较麻烦，对不对，如果我想记录一条关于我个人信息的数据，我使用文档来存，是不是很不友好，并且我们要查数据的时候，看图1：图1是一个word里面记录的信息，如果我想查询出所有人的名字，这个操作是不是就很难搞定了，来来来，配合起来~~，你应该说是的，那我就接着说，有同学可能就会说了，老师我用excel啊，看图2，一列就搞定了，没毛病，但是你想打开操作excel效率低不低。并且通过你自己写的程序来操作这些文件是不是很麻烦，就你们学的open函数。其实效率低的原因是因为我们知道文件都是保存在硬盘上的，硬盘的效率本身就低，所以没办法。





所以，为了方便的管理这些数据，又能提高对数据的管理效率，各个公司就开始想办法了，喊出了口号：我们要写一套软件，专门管理数据！！，让应用程序或者说项目程序不直接和硬盘打交道了，让我们自己写的管理数据的软件来操作数据，并且我们写的软件要解决下面几个问题：1.直接操作硬盘上的文件效率低 。 2. 通过操作文件来读写数据很麻烦。 3. 我们自己的电脑上写的程序和我们自己电脑上存储的数据都在这一台电脑上了，想和别人共享一个数据或者一个文档也是比较麻烦的，并且如果和别人共享，那么可能造成自己电脑的安全性变低了，但是公司内部的项目可能就会使用一些共同的数据啊，这样共享起来就很麻烦。 4. 项目和数据如果都在一个电脑上，例如京东，如果有一天，京东的项目出问题了，或者部署这个项目的电脑（服务器）崩了，你的数据是不是就麻烦了，嗯，稳定性和安全性都不够，我们要把数据和项目分开管理，一般公司都会这么做，看下图：



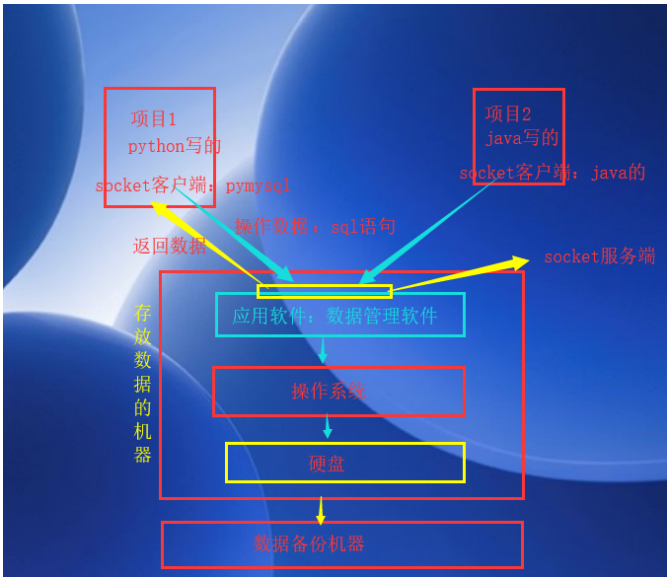
通过上面这个图里面的思想，貌似解决了共享数据和数据稳定性的问题，但是专门管理数据的机器里面还是将数据存到硬盘上啊，如果项目直接和硬盘打交道，效率还是差啊，并且虽然数据共享起来了，我们通过前面的学习知道，数据共享起来就会出现数据不安全的问题和数据混乱的问题啊，怎么办，加锁！而且很多项目或者人都可能过来连接中间这个存放数据的机器啊，我们不能让所有人都能连接这个机器啊，怎么办，加连接验证！并且要做好备份，因为现在数据多重要啊，所以还要支持做数据备份的工作！多个人连接使用这个机器中的数据的时候，还要支持并发啊，每个人的感觉都是自己单独的在操作这个机器啊，怎么办，支持并发！

哎，想要好好的玩数据，还真是tm的麻烦，不过各个公司都不怕艰难，都在努力的搞事情，搞一套可以高效管理数据的工具，称之为数据库管理软件/系统（应用软件）

又出来一个新词啊，数据库管理系统，来，先看看百度怎么解释的

> 数据库管理系统（英语：Database Management System，简称DBMS）是为管理数据库而设计的电脑软件系统，一般具有存储、

貌似又是不好理解，来吧，通俗解释--> 这个工具负责来和硬盘打交道(当然中间还隔着操作系统)，高效的管理数据，并且还支持对外通信，网络通信都是基于的socket，也就是说它还相当于一个socket服务端，那么想来到这个存数据的机器上来操作数据的人或者项目都可以连接到这个工具，并通过这个工具来管理数据，那么我们就可以通过下面这个图来看看这工具的工作方式：



上面这个图是不是就一目了然了，这个数据管理系统我们称之为DBMS，DB（database）就是数据库的意义，M（manage）就是管理的意思，S（system）就是系统的意思，其实就是英文名的首字符缩写。市场上冒出了很多优秀的数据库管理系统，例如：mysql、oracle、db2等等，人家开发好了，你使用就行了，既然是使用别人写好的数据库管理系统，那么我们在操作的时候，就要按照人家的规范来操作，这个规范叫做sql，我们通过这个系统来操作数据的语句叫做sql语句。那么过程就是这个样子的：首先下载安装人家的数据管理系统，然后启动系统，我们的项目如果想通过这个系统来操作数据，那么就需要你的项目中字节写一个socket客户端，要满足人家这个系统的服务端的消息格式要求，然后就按照人家规定好的sql语句写好操作数据的命令，使用你的写好的客户端通过网络发送给这个存放数据的机器上的数据管理系统的服务端，服务端街道这个命令之后，解析，然后产生对应的数据操作，你要是查询数据，就将查询数来的数据原路返回给你，如果你要是修改数据，那么我服务端就在系统上修改对应的数据。这就是整个操作流程了，其实这个系统就是一个基于socket编写的C/S架构的软件。

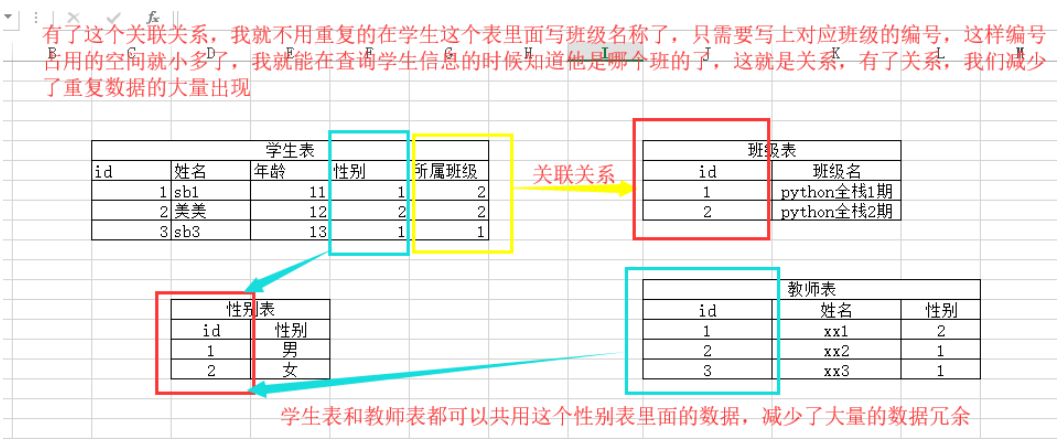
有人称这个数据管理系统为数据库，有人称这个存放数据的机器叫做数据库，有人称里面的一个存放数据的库叫做数据库，（存放数据的库，其实就是将数据分开管理，例如：你有两个项目来使用这个数据管理系统，那么我们两个项目的数据肯定是不能掺和在一块的对不对，那么就需要分开管理，分开管理的是可以给每个项目单独创建一个库，每个库存放自己项目对应的数据，有人称这个库为数据库），但是不管怎么称呼，你结合他当时说话的场景，就能够理解他指的是什么了，反正大意也都差不多，这个知道就好了。那么我们来看看有哪些NB的数据库管理系统啊，看下节的分类（还是先安装一下mysql搞一下看看在看分类吧！直接看第三大节）。

2. 数据库分类

目前的数据库可以分为两个大类：关系型数据库和非关系型数据库

2.1 关系型数据库（RDBMS）

解释：关系型数据库模型是把复杂的数据结构归结为简单的二元关系（即二维表格形式，不是excel，但是和excel的形式很像），结合下图来看一下，



这个表看着好乱啊，但是你细看一下，这几个表之间建立了某种关系，共享着双方的数据。这就是关系。关

系型数据库里面存数据的时候就类似这个样子的。有个大概了解了吗~~~

操作关系型数据库的命令，我们称之为SQL，看解释

结构化查询语言(Structured Query Language)简称SQL(发音：/'es kju: 'el/ "S-Q-L")，是一种特殊目的的编程语言，是一种数据库查
结构化查询语言是高级的非过程化编程语言，允许用户在高层数据结构上工作。它不要求用户指定对数据的存放方法，也不需要用户了
1986年10月，美国国家标准协会对SQL进行规范后，以此作为关系式数据库管理系统的标准语言（ANSI X3.135-1986），1987年

其中最后一句挺重要的：**不同数据库系统之间的**SQL**不能完全相互通用**

常用的关系型数据库：

2.1.1 oracle数据库

Oracle前身叫SDL、由Larry Ellison和两个变成任意在1977创办，他们开发了主机的拳头产品，在市场上大量销售。
Oracle公司是目前全球最大的数据库软件公司，也是近年业务增长极为迅速的软件提供与服务商
主要应用范围：传统大企业、大公司、政府、金融、证券等。
版本升级：oracle8i，oracle9i，oracle10g，oracle11g，oracle12c

2.1.2 MySQL

MySQL被广泛的应用在Internet上的大中小型网站中。由于体积小、速度快、总体拥有成本低，开放源代码

2.1.3 MariaDB数据库

MAriaDB数据库管理系统是MySQL数据库的一个分支，主要由开元社区维护，采用GPL授权许可。开发这个MariaDB

MariaDB基于事务的Maria存储引擎，替换了MySQL的MyISAM的存储引擎，它使用了Percona的XtraDB（InnoDB的

2.1.4 SQL Server数据库

Microsoft SQL Server是微软公司开发的大型关系数据库系统。SQL Server的功能比较全面，效率高，可以作为中型

2.1.5 Access数据库

Access是入门级小型桌面数据库，性能安全性都很一般，可供个人管理或小型企业只用
Access不是数据库语言，只是一个数据库程序，目前最新版本为Office 2007，其特点主要如下：
（1）完善地管理各种数据库对象，具有强大的数据组织，用户管理、安全检查等功能
（2）强大的数据处理功能，在一个工作组级别的网络环境中，使用Access开发的多用户管理系统具有传统的XSASE
（3）可以方便地生成各种数据对象，利用存储的数据建立窗体和报表
（4）作为Office套件的一部分，可以与Office集成，实现无缝连接
（5）能够利用Web检索和发布数据，实现与Internet的连接，Access主要适用于中小企业应用系统，或作为客户机/

2.1.6 其他不常用关系型数据库

DB2，PostgreSQL，Informix，Sybase等。这些关系型数据库逐步的淡化了普通运维的实现，特别是互联网公司几

2.2 **非关系型数据库**

非关系型数据库也被成为NoSQL**数据库，**NOSQL**的本意是**“Not Olnly SQL”
指的是非关系型数据库，而不是“No SQL”的意思，因此，NoSQL的产生并不是要彻底地否定关系型数据库，而是作为传统关系型数据库的一个有效补充。NOSQL数据库在特定的场景下可以发挥出难以想象的高效率和高性能。

随着互联网Web2.0网站的星期，传统的关系型数据库在应付web2,0网站，特别是对于规模日益扩大的海量数

据，超大规模和高并发的微博、微信、SNS类型的web2.0纯动态网站已经显得力不从心，暴露了很多难以克服的问题。

例如：传统的关系型数据库IO瓶颈、性能瓶颈都难以有效突破，于是出现了大批针对特定场景，以高性能和使用便利为目的功能特异化的数据库产品。NOSQL（非关系型）类的数据就是在这样的情景下诞生并得到了非常迅速的发展

高性能、高并发、对数据一致性要求不高
开源的NoSQL体系，如Facebook的Cassandra，Apache的HBase，也得到了广泛认同，Redis，mongb也逐渐越来越受到各类大中小型公司的欢迎和追捧

NOSQL非关系型数据库小结：
1、NOSQL不是否定关系数据库，而是作为关系数据库的一个重要补充
2、NOSQL为了高性能、高并发而生，忽略影响高性能，高并发的功能
3、NOSQL典型产品memcached （纯内存），redis（持久化缓存），mongodb（文档的数据库）

非关系型数据库又分为以下4种：

（1）键值（Key-Value）存储数据库

键值数据库就类似传统语言中使用哈希表，可以通过key来添加、查询或删除数据，因为使用key主键访问，所以会获得很高的性
键值（Key-Value）数据库主要是使用一个哈希表，这个表中有一个特定的键和一个指针指向特定的数据。Key/value模型对于IT
k1—>数据
k2—>数据
典型产品：Memcached、Redis、MemcacheDB、BerkeleyDB

（2）列存储（Column-oriedted）数据库 =======>了解即可，一般公司用不到
这部分数据库通常用来分布式存储的海量数据，键仍然存在，但是他们的特点是指向了多个列。
典型产品：Cassandra,HBase

（3）面向文档（Document-Oriented）数据库
面向文档数据库会将以文档的形式存储。每个文档都是自包含的数据单元，是一系列数据项的集合。每个数据项都有一个名称与对
典型产品：MorgoDB、CouchDB

（4）图形（Graph）数据库

常见的非关系型数据库

2.2.1 memcached（key-value）

Memcaced是一个开源的、高性能的、具有分布式内存对象的缓存系统。通过它可以减轻数据库负载，加速动态的web应用，最初
缓存一般用来保存一些进程被存取的对象或数据，通过缓存来存取对象或数据要比在磁盘上存取块很多，前者是内存，后者是磁盘、M
官网：http://memcached.org/
由于memcached为纯内存缓存软件，一旦重启所有数据都会丢失，因此，新浪网基于Memcached开发了一个开源项目Memcach

Memcached小结：
1、key-value行数据库
2、纯内存数据库
3、持久化memcachedb（sina）

2.2.2 Redis（key-value）

和Memcached类似，redis也是一个key-value型存储系统。但redis支持的存储value类型相对更多，包括string（字符串）、lis

redis是一个高性能的key-value数据库。redis的出现、很大程度补偿了memcached这类key/value存储的不足，在部分场合可以

官方：http://www.redis.io/documentation
redis特点：
1）支持内存缓存，这个功能相当于memcached
2）支持持久化存储，这个功能相当于memcachedb，ttserver
3）数据库类型更丰富。比其他key-value库功能更强
4）支持主从集群、分布式
5）支持队列等特殊功能
应用：缓存从存取memcached更改存取redis

2.2.3 MongoDB（Document-oriented）

MongoDB是一个介于关系型数据库和非关系型数据库之间的产品，是非关系型数据库当中功能最丰富，最像关系数据库的。他支持

特点：
高性能、易部署、易使用、存储数据非常方便
主要功能特性：

1.面向集合存储，易存储对象类型的数据
2.“面向集合”（Collenction-Orented）意思是数据库被分组存储在数据集中，被称为一个集合（Collenction）每个 集合在数
3.模式自由
模式自由（schema-free）意为着存储在mongodb数据库中的文件，我们不需要知道它的任何结构定义。
4.支持动态查询
5.支持完全索引，包含内部对象
6.支持查询
7.支持复制和故障恢复
8.使用高效的二进制数据存储，包括大型对象
9.自动处理碎片、以支持云计算层次的扩展性

2.2.4 Cassandra（Column-oriented）

Apache Cassndra是一套开源分布式Key-Value存储系统。它最初由Facebook开发，用于存储特别大的数据。Facebook目前在使
主要特点：

1.分布式
2.基于column的结构化
3.高伸展性
4.Cassandra的主要特点就是它不是一个数据库，而是由一堆数据库节点共同构成一个分布式网络服务，对Cassandra的一个写操
Cassandir是一个混合型的非关系的数据库，类似于Google的BigTable。其主要功能比Dynomie（分布式的key-value存储系统）

2.2.5 其他不常用非关系型数据库

HBase、MemcacheDB、BerkeleyDB、Tokyo Cabinet\Tokyo Tyrant（ttserver）
ttserver 持久化输出，缺点存储2千万条 性能下降（由日本人发明）

看了这么多的数据库，我们主要讲的是MySQL，这个公司里面非常常用的又非常nb的关系型数据库，后面还会将一些非关系型数据库的使用，来吧，我们好好认识一下MySQL，看下一节介绍！

二 MySQL介绍

1.mysql**版本**
双授权版本：社区版（完全免费，功能也够nb了）和商业版（更好，功能更多更强大一些，但是收费，VIP，有售后服务，也会参考和吸收社区版的一些nb的功能，安全性和稳定性都是最好的，大几十万），一般NB的开源软件都是双授权的
每个版本又分四个版本依次进行发布：
Alpha版：一般只在开发公司内部使用，不对外公开，测试、自我检查的版本
Beta版：一般是开发完也测试完的版本，一般不会出现比较大的性能bug（一般我们不用，阿里、去哪儿会使用这个版本，有些新功能，内部有高手能调，也能评估新功能的性能）

RC版：根据Beta版测试之后收集到一些bug等在进行完善的一个版本

GA版：软件正式发布的版本，没有特别的需求一定要使用GA版，有些公司追求新功能会使用Beta版，这是个例。

2.MySQL**的产品线：**（mysql是C++写的，oracle 9i版本之前是C语言写的，之后主要是java）

最早期，mysql是按照3.x--4.x--5.x等来开发的，但是为了提高MySQL的竞争优势，以及提高性能、降低开发维护成本等原因，同时，更方便企业用户更精准的选择合适的版本产品用于自己的企业生产环境中，MySQL在发展到5.1系列版本之后，重新规划为三条产品线。

第一条：5.0.xx 到 5.1.xx产品线系列介绍

第二条：5.4.xx 到 5.7.xx产品线系列介绍(主流：5.5和5.6)

第三条：6.0.xx 到 7.1.xx产品线系列介绍

3. MySQL**数据库软件命名介绍**

以mysql-5.6.42.tar.gz的版本号来解释：

1.第一个数字5是主版本号，描述了文件格式。所有版本5发行都有相同的文件格式。

2.第二个数字6是发行级别。主版本号和发行级别组合到一起便构成了发行序列号。

3.第三个数据42是在此发行系列的版本号，随每个新发布版递增。通常你需要已经选择发行的最新版本，每次更新后，版本字符串的最后一个数字会递增。如果增加了一些新功能或者微小的不兼容性，版本字符串的第二个数字会递增。如果文件格式改变，第一个数字会递增。

一般有的版本也会加上上面我们说的4个版本的后缀，beta、alpha、rc版、ga版等等，我们举得这个例子是不带后缀的，就相当于GA版

4.1 MySQL**版本选择建议**

1.稳定版：选择开源的社区版的稳定版GA版本

2.产品线：可以选择5.1、5.5、5.6，互联网公司主流5.5和5.6，其次是5.1.

3.选择MySQL数据库GA版发布后6个月以上的GA版本。

4.要选择前后几个月没有大的BUG修复的版本，而不是大量修复BUG的集中版本

5.最好向后较长时间没有更新发布的版本。

6.开发、测试、运维、DBA进行自己本地测试的时候，最好要和线上的版本一致，最差也要兼容，所以作为开发，你要清楚公司用的哪个版本的数据库

7.作为内部开发测试数据库环境，跑大概3-6个月的时间。

8.优先企业非核心业务采用新版本的数据库GA版本的软件。

10.想DBA高手请教，或者在技术分为好的群里和大家一起交流，使用真正高手用过的好用的GA版本产品

经过上述工序后，若没有重要的功能BUG或者性能瓶颈，则可以开始考虑作为任何业务数据服务的后端数据库软件。

好了，同志们，铺垫了这么多，我们要开始学习实战内容啦，来看第四大节，修改密码~~

三 MySQL**的下载安装、简单应用及目录介绍**

1.**下载安装**

这个下载安装内容我专门整理了一个博客，专门针对的windows10的：https://www.cnblogs.com/clschao/articles/9916971.html，大家参考其中的第二种下载安装方法，我们后面的学习就是根据第二种方法来的。

linux系统下MySQL数据库企业生产中常用的4中安装方法介绍(将来如果需要你一个开发人员来安装数据库或者自己想研究数据库的时候再自行去研究吧~~~这里就简单给大家提一下~~~)

1.yum/rpm方式安装MySQL

2.常规方式编译安装MySQL

3.采用cmake方式编译安装MySQL

4.采用二进制方式免编译安装MySQL

以上的安装方法都可以，性能上也不会有太多的差距，只是不同层次的人的安装习惯不同，了解一下就可以了，多数的运维人员习惯使用cmake编译方式安装，如果数据库服务器很多，而且对定制化有要求，可以选择通过源码定制rpm包，搭建yum参数的方式安装，但是需要你有一定的能力，还有好多专业的DBA选择二进制免编译安装的方式进行安装。

将来大家更多的是面对linux下的mysql安装和使用（没有包含上述的所有安装方法），我也给大家整理了一篇

博客（centos7.1下安装mysql5.6）：https://www.cnblogs.com/clschao/articles/6736840.html

如果我上面这篇博客安装的时候有问题，并且自己解决不了的话，删除你已经安装的mysql，然后看另外一篇博客进行安装（centos7下安装mysql5.6），其实道理都差不多，我的那一篇详细一些，还有一些关于防火墙的设置，话不多说。这一篇博客也挺好的，转载过来给大家看
看：https://blog.csdn.net/qq_17776287/article/details/53536761

如果你们公司用的是红帽或者是Ubuntu，可以自行百度教程来尝试安装，这里我就不给大家演示啦。

这位同学，如果你mac本，我只能说超哥一直以来都很穷。。。没有一台自己的mac本，所以不是很熟悉，但是给你找了两篇博客，可以参考一下，我大致看了一下，过程是差不多的：

mac 安装mysql5.6 ：https://blog.csdn.net/mike694439716/article/details/48218239?utm_source=blogxgwz0

mac 安装tar.gz版MySQL5.6：https://blog.csdn.net/zmx729618/article/details/72769840

如果Mac本安装完mysql之后，登陆以后，不管运行什么指令，总是提示这个：mac mysql error You must reset your password using ALTER USER statement before executing this statement.

你需要做的就是修改密码就行了，默认安装完成之后，mysql应该是安装在了/usr/local/mysql这个目录下，里面的目录结构和windows的是一样的。

mysql安装的简单总结：

```
#1、下载：MySQL Community Server 5.7.16
http://dev.mysql.com/downloads/mysql/

#2、解压
如果想要让MySQL安装在指定目录，那么就将解压后的文件夹移动到指定目录，如：C:\mysql-5.7.16-winx64

#3、添加环境变量
【右键计算机】--》【属性】--》【高级系统设置】--》【高级】--》【环境变量】--》【在第二个内容框中找到 变量名为Path 的-

#4、初始化
mysqld --initialize-insecure

#5、启动MySQL服务
mysqld # 启动MySQL服务

#6、启动MySQL客户端并连接MySQL服务
mysql -u root -p # 连接MySQL服务器

#7、将mysql添加系统服务
   注意：--install前，必须用mysql启动命令的绝对路径
   # 制作MySQL的Windows服务，在终端执行此命令：
   "c:\mysql-5.7.16-winx64\bin\mysqld" --install

   # 移除MySQL的Windows服务，在终端执行此命令：
   "c:\mysql-5.7.16-winx64\bin\mysqld" --remove

   注册成服务之后，以后再启动和关闭MySQL服务时，仅需执行如下命令：
   # 启动MySQL服务
   net start mysql

   # 关闭MySQL服务
   net stop mysql
```

1.解压tar包
cd /software #cd到一个自己创建的文件夹中
tar -xzvf mysql-5.6.21-linux-glibc2.5-x86_64.tar.gz #解压下载下来的mysql文件，如果没在这个文件夹中，记得把文件移动到
mv mysql-5.6.21-linux-glibc2.5-x86_64 mysql-5.6.21 #通过mv指令给这个解压出来的文件改了个名字

2.添加用户与组
groupadd mysql #添加用户组
useradd -r -g mysql mysql #创建mysql用户，并添加到mysql用户组
chown -R mysql:mysql mysql-5.6.21 #这是mysql用户和mysql用户组的归属
chmod +x -Rf /usr/local/mysql #赐予可执行权限

3.安装数据库
su mysql
cd mysql-5.6.21/scripts
./mysql_install_db --user=mysql --basedir=/software/mysql-5.6.21 --datadir=/software/mysql-5.6.21/data #使用mys

4.配置文件
cd /software/mysql-5.6.21/support-files #配置文件在这个目录下
cp my-default.cnf /etc/my.cnf #copy一份my-default.cnf文件到etc目录下，并起名为my.cnf文件
cp mysql.server /etc/init.d/mysql #copy一份mysql.server文件，到etc的init.d的mysql文件夹中，启动加载的初始配置文件会有
vim /etc/init.d/mysql #若mysql的安装目录是/usr/local/mysql,则可省略此步
修改文件中的两个变更值
basedir=/software/mysql-5.6.21 #基础目录
datadir=/software/mysql-5.6.21/data #数据目录

5.配置环境变量
vim /etc/profile #环境变量的配置文件，添加下面两行
export MYSQL_HOME="/software/mysql-5.6.21"
export PATH="$PATH:$MYSQL_HOME/bin"
#使配置生效，通过source指令
source /etc/profile

6.添加自启动服务
chkconfig --add mysql
chkconfig mysql on

7.启动mysql
service mysql start

8.登录mysql及改密码与配置远程访问
mysqladmin -u root password 'your_password' #修改root用户密码
mysql -u root -p #登录mysql,需要输入密码
mysql>GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY 'your_password' WITH GRANT OPTION; #允许root用户
mysql>FLUSH PRIVILEGES; #刷新权限

9. 一些必要的初始配置（除了下面这些，将来玩mysql的时候还有很多很多的配置）
1）修改字符集为UTF8
vi /etc/my.cnf
在[client]下面添加 default-character-set = utf8
在[mysqld]下面添加 character_set_server = utf8
2）增加错误日志
vi /etc/my.cnf
在[mysqld]下面添加:
log-error = /usr/local/mysql/log/error.log
general-log-file = /usr/local/mysql/log/mysql.log
3) 设置为不区分大小写，linux下默认会区分大小写。
vi /etc/my.cnf
在[mysqld]下面添加:
lower_case_table_name=1

修改完重启：#service mysql restart

在linux上使用mysql的时候，一定要注意的就是权限问题，linux恶心的地方就是权限问题。

2.mysql**的简单使用演示**

这里只是给大家演示一下mysql是个什么样子，具体怎么管理数据，后面我们会学，这里只做演示用，因为演示一下之后，你对这个东西就有了一些简单的认识和了解，再进行后面的学的时候，你就不会摸着黑听理论了，所谓的瞎听了，哈哈。注意，我们下面会输入一些指令进行操作数据库，数据库里面的指令必须要用；分号结尾，然后才能执行，切记。

1.开启服务端，mysqld\ net start mysql

2.使用mysql自带的客户端进行连接，cmd下输入mysql -u root -p，然后回车，会提示你输入密码，此时初始的root用户还没有密码，所以还是直接回车就可以连接上了

3.show databases；先不讲里面的内容，说一下这是几个库，每个项目可以有自己单独的一个库，里面放这个项目的所有数据表

4.创建一个库：create database CRM；然后show databases；查看一下就有了这个crm库，不分大小写，统一会变成小写，对照着我们mysql安装目录下的data文件夹里面的内容看一下，库就是对应的文件夹。

5.我们目前在所有数据库之上，想在我们自己项目的库里面操作数据，就需要切换到我们自己这个crm项目的库里面进行数据的操作，切换数据库使用use + 库名，例如：use crm；就提示你切换成功了。

6.我们说过，库里面维护的数据就像一张一张的数据表，类似excel，对不对，那我们创建一张表看一下，命令：

```
create table student(
    id int,
    name char(10),
    age int
);
```

7.再执行show tables;就可以看到有了一个student表

8.查看一下这个表里的数据select * from student；发现什么数据也没有

9.插入几条数据,写几条数据：insert into student values(1,'d',18),(2,'x',11),(3,'d',10),(4,'k',9);然后回车，就执行了这条指令，然后我们再查看一下这个表里面有没有数据了，执行上一条指令，select * from student；发现里面就有数据了：

结果：

```
mysql> select * from student;
+------+------+------+
| id | name | age |
+------+------+------+
| 1 | d | 18 |
| 2 | x | 11 |
| 3 | d | 10 |
| 4 | k | 9 |
+------+------+------+
4 rows in set (0.00 sec)
```

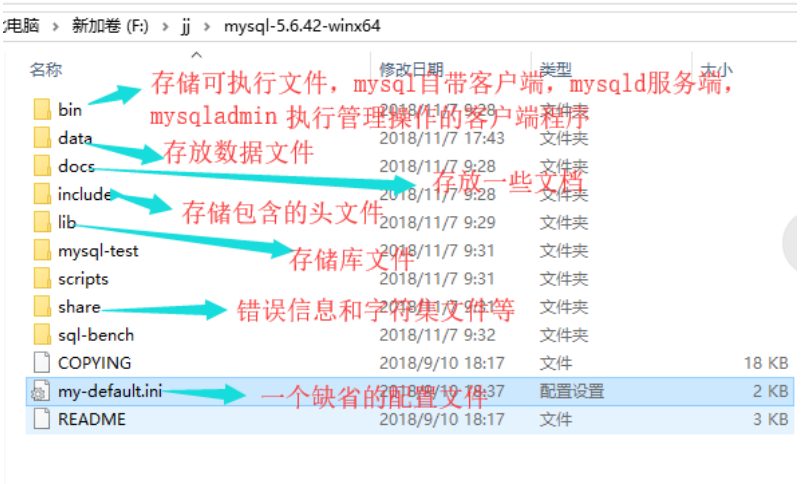
分析一下：上面这些就是mysql数据帮我们保存的数据，以表格的外貌展示，第一行为表头，从第二行开始都是对应的数据，每列都是自己这一列规定的内容，比如id这一列是你插入的这几条数据的id，我们这个insert就是插入数据，select就是查看数据，这就是我们通过MySQL自带的客户端来操作的MySQL服务端来进行数据的操作。MySQL服务端和操作系统及硬盘打交道，快速的帮你实现数据的操作，其他的语言开发的客户端就是通过这种形式来操作数据库里面的数据的，将来我们使用python操作数据库的时候，会使用一个叫做pymysql的工具来搞，到时候会给你们讲，他就是一个咱们MySQL服务器的客户端，连接上服务端就可以操作服务端的保存的你的项目的数据了。例如用户要查看自己的信息，就通过你写的程序接受到客户的请求，通过自己的mysql客户端去MySQL服务端查看对应的信息，然后mysql服务端将这些信息发送给你的py程序客户端，你通过程序再将数据返回给你的用户，你的用户就看到了自己的信息，就是这么个过程，大家理解了吗。

真正的数据库维护优化等高级数据库的技术一般都是由公司的DBA来做，或者由比较懂数据库的运维来做，一般不会让开发来搞，除非你开发人员的数据库能力很强，这些NB的技术包括：数据库优化，数据库BUG解决，数据库备份（冷备、热备），保证数据不丢失，集群，高可用等等保证项目的稳定性和可用性及高并发用（很多的用户都来操作数据，你要并发），数据库各项配置参数的调优，慢sql语句的提炼和调优，数据库开发、数据库更新，数据迁移，数据恢复，分库分表等等，这些是数据库的高端技术，而针对开发人员，一般你需要学习这些：基本的开发环境使用的数据库搭

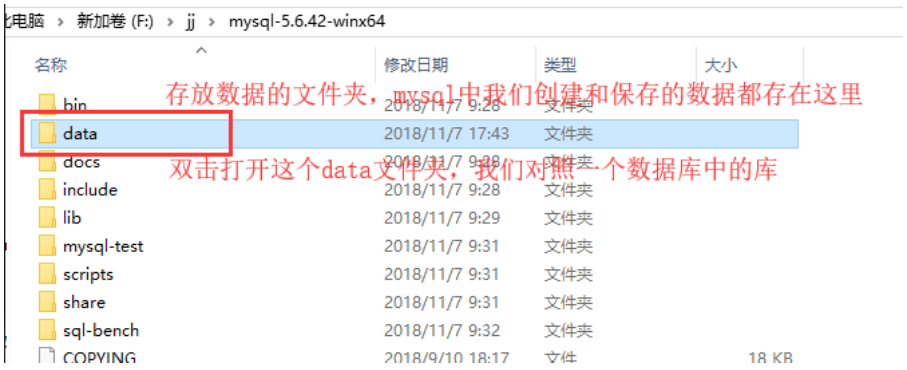
建，然后增删改查就差不多了。当然如果想提升自己的能力和水平（还有薪资水平），数据库是你必须要学好的内容，但是那是你做开发之后的事情了，而且有好多同学目前已经在数据库或者运维方面很NB了，但是开发还不行啊，对不对，哈哈，好好学python，数据库不是你学习python的重点，但是必须要会一些基本的内容，懂得越多越好，ok吗，同志们~~~

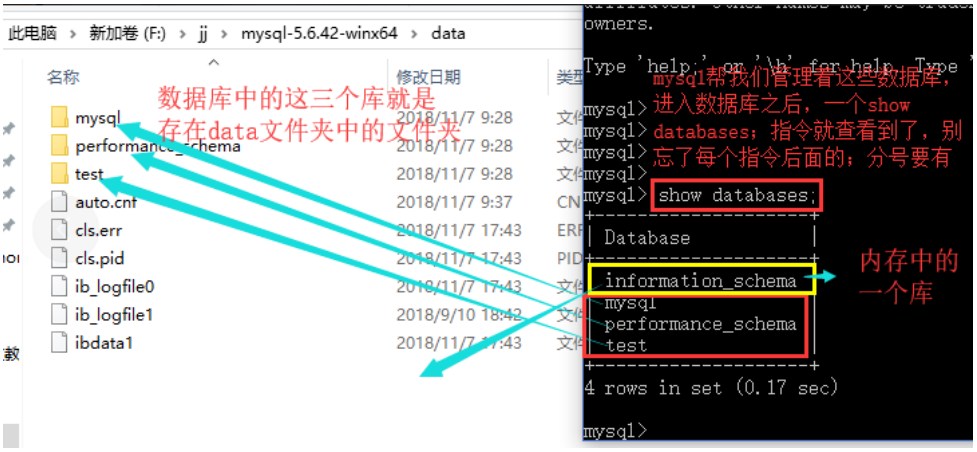
3.MySQL安装目录介绍

看图：



其中，我们重点看一下data文件夹：如果你找不到自己建立的库或者表的文件，可能不在这个data文件夹下面，连接上mysql之后，输入 `show global variables like "%datadir%"` ;来查看数据文件存储路径，找到路径之后，到对应路径下如果找不到这个文件夹，那么可能是隐藏的，把





关于数据库中的这4个初始的库的详细介绍，有兴趣的同学可以来看看我的这篇文章（目前作为了解用）：https://www.cnblogs.com/clschao/articles/9928223.html

其中mysql这个库我需要提一下：

mysql：这个是mysql整个服务的核心数据库，类似于sql server中的master表，主要负责存储数据库的用

户、权限设置、关键字等mysql自己需要使用的控制和管理信息。不可以删除，如果对mysql不是很了解，也不要轻易修改这个数据库里面的表信息。

总结：其实这些库就是我们电脑上对应的文件夹，在mysql**中显示为对应的库，来方便我们管理数据，而文件或者文件夹这种与硬盘打交道的事情就交给**mysql**了，我们只需要对**mysql**库中的数据进行操作就可以了，你可以看到，我们刚才简单使用的时候创建的一个**crm**库，也就是在**data**目录下的生成了一个**crm**文件夹。**

说到这里，大家对数据库有个基本的了解了呢。那么我们返回去看一下mysql的一些其他知识（提高逼格的内容~~），回到数据库分类及mysql介绍~~~~~

四 root用户密码设置及忘记密码的解决方法

再怎么说我们的root用户密码也不能为空啊对不对，所以需要设置一个密码，看下面设置密码的方法，我给了三种方法：

方法1： 用SET PASSWORD**命令**

首先登录MySQL，使用mysql自带的那个客户端连接上mysql。

格式：mysql> set password for 用户名@localhost = password('新密码');

例子：mysql> set password for root@localhost = password('123');

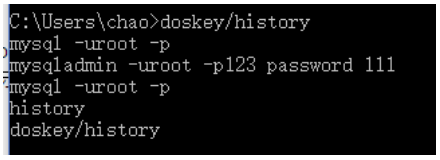
方法2：用mysqladmin （因为我们将bin已经添加到环境变量了，这个mysqladmin也在bin目录下，所以可以直接使用这个mysqladmin功能，使用它来修改密码）

关于mysqladmin的介绍：是一个执行管理操作的客户端程序。它可以用来检查服务器的配置和当前状态、创建和删除数据库、修改用户密码等等的功能，虽然mysqladmin的很多功能通过使用MySQL自带的mysql客户端可以搞定，但是有时候使用mysqladmin操作会比较简单。

格式：mysqladmin -u用户名 -p旧密码 password 新密码

例子：mysqladmin -uroot -p123456 password 123

只用mysqladmin的时候，会出现一个warning警告信息：Warning: Using a password on the command line interface can be insecure.，这个没关系，是提示你，你直接在cmd下使用明文设置密码的时候，是不安全的，因为别人可以通过翻看你输入指令的历史记录来查看到你设置的密码，所以提示你一下，不信你按上下键，可以看到自己之前输入的命令，或者输入下面这个指令也可以看到：



所以我们最好连接进入到mysql里面之后，在进行密码的修改和设置。

方法3：用UPDATE**直接编辑那个自动的**mysql**库中的**user**表**

首先登录MySQL，连接上mysql服务端。

mysql> use mysql； use mysql的意思是切换到mysql这个库，这个库是所有的用户表和权限相关的表都在这个库里面，我们进入到这个库才能修改这个库里面的表。

mysql> update user set password=password('123') where user='root' and host='localhost'； 其中password=password('123') 前面的password是变量，后面的password是mysql提供的给密码加密用的，我们最好不要明文的存密码，对吧，其中user是一个表，存着所有的mysql用户的信息。

mysql> flush privileges；刷新权限，让其生效，否则不生效，修改不成功。

在忘记root**密码的时候，可以这样（注意：**root**密码最好不要忘记，找地方记录下来，不然如果是工作中你们使用的数据库（不管是测试的还是线上的，都是比较麻烦的事情，数据库轻易不会让你重启的，不过作为一个开发来讲，你应该是无法用**root**用户的~~~））**

以windows**为例：**

1. 关闭正在运行的MySQL服务，net stop mysql（这个mysql是你添加的mysqld到系统服务时的服务名）。

2. 打开DOS窗口，转到mysql\bin目录。

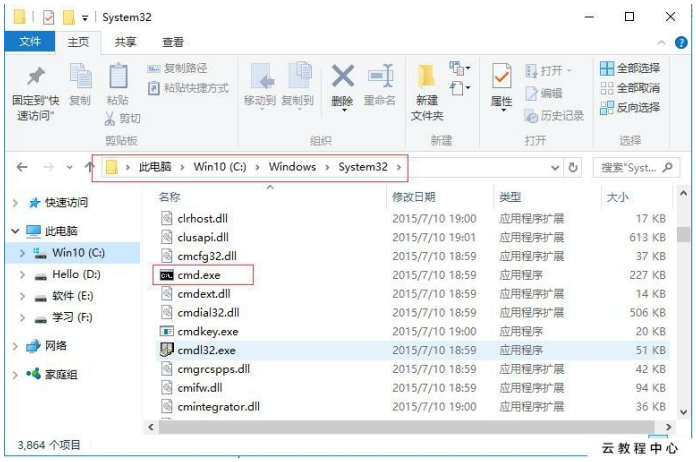
3. 输入mysqld --skip-grant-tables 回车。--skip-grant-tables 的意思是启动MySQL服务的时候跳过权限表认证，因为之所以mysql启动之后，客户端连接的时候需要登陆认证，输入密码什么的，是因为mysql服务端启动的时候，加载了自己内部的一些权限相关信息的授权表、权限认证表什么的，这样就要求客户端必须有认证，如果启动的时没有加载这些表和设置，那么我们客户端再进行登陆的时候，就不需要认证了，那么就可以登陆上了，登陆之后，我们到mysql这个存有所有用户信息的表中去修改root用户或者别的用户的密码了，还是比较6的，但是这样搞需要关闭服务端，在实际工作中想关闭mysql服务？？你觉得可能吗？？记住这个问题，我后面给大家解决。

注意一个问题，如果我们直接使用的上面这个指令，也即是mysqld --skip-grant-tables，也就是直接通过mysqld启动的mysql服务的话，我们就不能通过net stop mysql的方式来关闭mysql服务了。但是可以通过别的方式来关闭，我在安装mysql的那篇博客里面写到了，杀进程的方式，知道你肯定忘记了，再给你写一下（win10）：tasklist |findstr mysqld找到这个mysqld服务的端口号，然后taskkill /F /PID 端口号来杀死这个mysql服务的进行，以后就可以使用net start/stop mysql的方式来启动和关闭了。

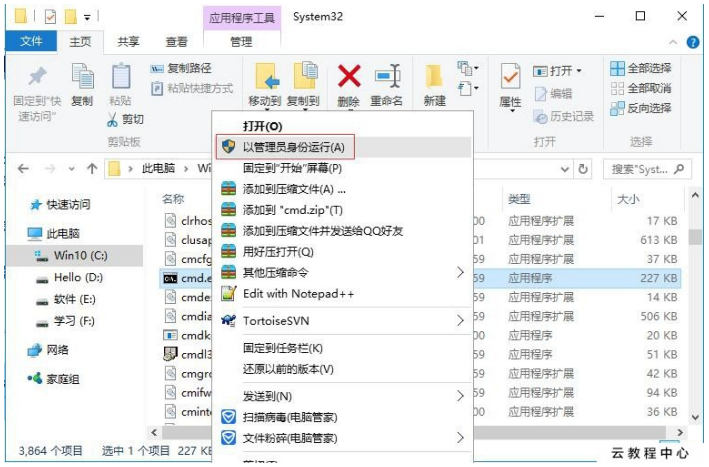
4. 再开一个DOS窗口（因为刚才那个DOS窗口已经不能动了），转到mysql\bin目录。
5. 输入mysql回车，如果成功，将出现MySQL提示符 >。
6. 连接权限数据库： use mysql; 。
6. 改密码：update user set password=password("123") where user="root";（别忘了最后加分号） 。
7. 刷新权限（必须步骤）：flush privileges; 。凡是涉及到密码修改或者后面我们会学到的权限修改，修改完之后全部要再执行一下这一句。
8. 退出 quit。
9. 注销系统，再进入，使用用户名root和刚才设置的新密码123登录。

注意：我们在使用cmd的时候，经常需要使用管理员身份来运行cmd窗口，每次都需要自己右键选择管理员身份运行，很麻烦，所有有永久解决的方法，看我下面的操作：

1、去"C:/Windows/System32"目录找到"cmd.exe":



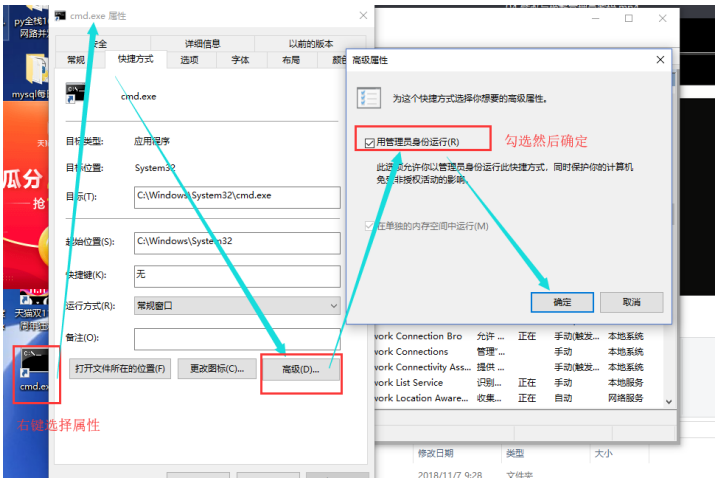
2、右击属性,选择"以管理员身份运行":



永久解决办法：

1、创建"cmd.exe"快捷方式:

2、右击选择“属性”,选择“快捷方式”,再选择“高级”,在选择“以管理员身份运行”,再单击“确定”。



以后只要打开快捷方式就可以以管理员的身份运行cmd了!

****总结一下*****，到目前为止，我们已经大致了解mysql了，并且知道怎么使用自带的mysql客户端来连接mysql服务端，还知道怎么修改密码什么的了，那后面的我们是不是就应该实战了呢，哈哈，实战之前，本来想带着大家使用xshell这个客户端工具来操作一下mysql，不过学它需要一些后面的知识，后面再学吧~~~不过还需要做一件事情，那就是数据库的编码问题，我们知道自己写socket简单对话程序的时候还需要双方规定好编码方式，是gbk啊还是utf-8啊等问题，不然容易报错或者乱码，对吧，mysql也存在这个问题，这个问题搞不清楚，将来你们学习、甚至工作中都会非常头疼~~~来，我们就学一下怎么把mysql的编码调好~~~

五 修改字符集编码

使用数据库的时候要注意的字符集编码，其实主要的是中文乱码的问题，大家应该对编码比较熟悉了，双方沟通需要编码相同不然容易报错或者出现乱码的问题，在使用数据库的时候也会存在这样的问题，所以我们需要解决这个问题：

先来模拟一下这个问题：

我们将刚才的student数据表删除，我们再来创建一个student表，然后往这个表里面插入几条含有中文的数据来看一下效果：

```
1.create table student(id int,name char(10),age int);
2.insert into student value(1,'呵呵',11),(2,'老刁',12),(3,'dsb',10),(4,'你好',9);
3.select * from student;
+------+------+------+
| id | name | age |
+------+------+------+
| 1 | ?? | 11 |
| 2 | ?? | 12 |
| 3 | dsb | 10 |
| 4 | ?? | 9 |
+------+------+------+
4 rows in set (0.00 sec)
```

发现什么问题，插入的中文怎么成了??号了，乱码了。。。

什么原因呢？我们先来看看我们创建这个crm库的时候，是怎么创建的，输入查看库创建的是时候mysql内部实际执行的创建语句的指令（咱们自己写了个指令，但是mysql在执行的时候会按照自己的方式来执行这个执行，现在就看看它实际执行的时候是个什么语句）：show database crm\G；然后看结果：

```
mysql> show create database crm\G
*************************** 1. row ***************************
Database: crm
Create Database: CREATE DATABASE `crm` /*!40100 DEFAULT CHARACTER SET latin1 */
1 row in set (0.04 sec)
```

可以看到，我们创建这个库的时候，mysql默认帮我们指定了一个字符集：latin1，就是上面的后面半句DEFAULT CHARACTER SET latin1，创建库的时候默认指定了latin1的意思是，在这个库里面我们创建的数据表，只要没有给表指定字符集，那么这个库里面的所有表都将是latin1字符集的(除了库创建的时候可以指定字符集之外，创建表的

时候也可以指定字符集)，来看一下创建表的时候，查看表创建语句的指令(此时我们并没有在创建表的时候指定字符集对吧)：show create table student\G;看结果：

```
mysql> show create table student\G;
*************************** 1. row ***************************
Table: student
Create Table: CREATE TABLE `student` (
  `id` int(11) DEFAULT NULL,
  `name` char(10) DEFAULT NULL,
  `age` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1
1 row in set (0.00 sec)
```

发现创建表的时候，mysql也默认给我们指定了一个字符集，也是latin1，因为库就是latin1的。

下面我们来看一下MySQL的一些初始信息：通过指令(\s)来查看，注意只需要看里面的有关characterset的部分

```
mysql> \s
--------------
mysql Ver 14.14 Distrib 5.6.42, for Win64 (x86_64)

Connection id: 12
Current database: crm
Current user: root@localhost
SSL: Not in use
Using delimiter: ;
Server version: 5.6.42 MySQL Community Server (GPL)
Protocol version: 10
Connection: localhost via TCP/IP
Server characterset: latin1 #服务端是latin1
Db characterset: latin1 #数据库默认是latin1
Client characterset: gbk #我们现在用的客户端是gbk
Conn. characterset: gbk #双方连接也是gbk编码的，这个不用管
TCP port: 3306
Uptime: 4 days 18 hours 52 min 50 sec

Threads: 1 Questions: 88 Slow queries: 0 Opens: 70 Flush tables: 1 Open tables: 61 Queries per second avg: 0.000
--------------
```

上面看的不够清晰，那么我们再看一下各个角色的编码（客户端、服务端，数据库等等）：

在mysql中执行指令：show variables like '%char%'；看结果：

```
mysql> show variables like "%char%";
+--------------------------+----------------------------------------+
| Variable_name            | Value                                  |
+--------------------------+----------------------------------------+
| character_set_client     | gbk                                    |
| character_set_connection | gbk                                    |
| character_set_database   | latin1                                 |
| character_set_filesystem | binary                                 |
| character_set_results    | gbk                                    |
| character_set_server     | latin1                                 |
| character_set_system     | utf8                                   |
| character_sets_dir       | F:\jj\mysql-5.6.42-winx64\share\charsets\ |
+--------------------------+----------------------------------------+
8 rows in set (0.00 sec)
```

关于上面这些编码的详细解释，看我的博客：https://www.cnblogs.com/clschao/articles/9946174.html，我就不给大家详细说啦。

那为什么出现乱码问题呢？又如何解决乱码问题呢？

原因：因为客户端mysql的字符集和服务端的字符集不一样，注意一下其中的character_set_client、character_set_connection、character_set_results这三项，我们在这里可以简单称为客户端三炮，就是因为这三炮和服务端的编码不一致导致的，所以我们需要将这三项改为和服务端一致的字符集就可以了。

解决方案：

一.在插入数据之前，先执行一条指令：set names latin1；临时修改客户端三炮的字符集，让客户端

插入数据的时候按照服务端的字符集编码来插入数据，然后我们再插入一条数据，然后看效果：

```
mysql> set names latin1;
Query OK, 0 rows affected (0.14 sec)
```

然后我们再查看一下编码：

```
mysql> show variables like "%char%";
+--------------------------+----------------------------------------
-+
| Variable_name | Value |
+--------------------------+----------------------------------------
-+
| character_set_client | latin1 |
| character_set_connection | latin1 |
| character_set_database | latin1 |
| character_set_filesystem | binary |
| character_set_results | latin1 |
| character_set_server | latin1 |
| character_set_system | utf8 |
| character_sets_dir | F:\jj\mysql-5.6.42-winx64\share\charsets\ |
+--------------------------+----------------------------------------
-+
8 rows in set (0.00 sec)
```

发现客户端三炮都改为和服务端一致的latin1了，按照我们刚才说的，按理说再插入数据应该就不会乱码了。来插入一条数据试试看：

```
mysql> insert into student value(5,'不乱了吧',111);
Query OK, 1 row affected (0.09 sec)

mysql> select * from student;
+------+----------+------+
| id | name | age |
+------+----------+------+
| 1 | ?? | 11 |
| 2 | ?? | 12 |
| 3 | dsb | 10 |
| 4 | ?? | 9 |
| 5 | 不乱了吧 | 111 | #再次插入的数据就不乱吗了，但是之前乱码的内容还是乱码的内容
+------+----------+------+
5 rows in set (0.00 sec)
```

总结：在进行DQL和DML语句(关于DQL和DML的解释我们后面会讲的，你就理解为一些sql语句)之前，先执行set names latin1；

但是我们如果断开连接，退出数据库之后，在连接进来以后，插入数据时如果不执行set names latin1，还是会乱码，说明这句指令没有让字符集永久生效。

不信，我们退出一下，然后再连接进来看看：

```
mysql> quit
Bye

C:\Users\chao>mysql -uroot -p #连接进来

mysql> use crm; #切换库
Database changed
mysql> select * from student; #查看crm库中的student表中的数据
+------+-----------+------+
| id | name | age |
+------+-----------+------+
| 1 | ?? | 11 |
| 2 | ?? | 12 |
| 3 | dsb | 10 |
| 4 | ?? | 9 |
| 5 | ??????? | 111 |
+------+-----------+------+
5 rows in set (0.00 sec)
```

发现还是tm的乱码，真恶心啊，果然没有永久生效，我们在使用一下set names latin1；然后再查看一下表中的数据

```
mysql> set names latin1;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from student;
+------+----------+------+
| id | name | age |
+------+----------+------+
| 1 | ?? | 11 |
| 2 | ?? | 12 |
| 3 | dsb | 10 |
| 4 | ?? | 9 |
| 5 | 不乱了吧 | 111 |
+------+----------+------+
5 rows in set (0.00 sec)
```

发现第五条数据，也就是我们之前使用latin1插入的数据，在查看之前使用set names latin1；还是可以看到对应的不乱码的数据的，因为我们就是以latin1的字符集插入的，只是查看的时候客户端三炮还是之前的gbk的编码，所以直接查看结果的时候还是乱码的。

这样虽然可以解决乱码问题，但是没办法永久解决乱码问题，所以每次在进行sql语句输入之前都要先执行一下set names latin1；(latin1这里代指的是服务端的字符集，不一定就是latin1)，所以我们还有其他的方法来解决，修改配置文件，看第二种方法!

注意一点：如果想把之前已经乱码的数据改为不乱码，在工作中，我们需要将数据全部导出来，然后重新建库建表，再把数据导进来。

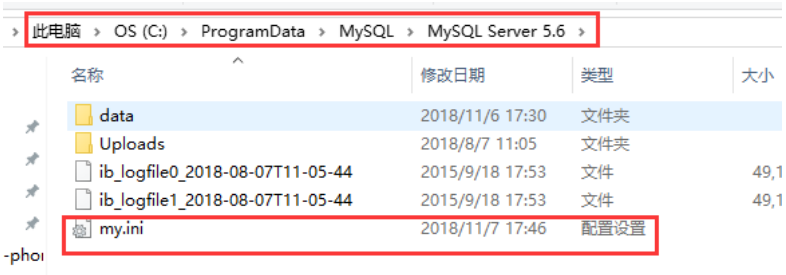
二.在配置文件里面修改客户端和服务端参数，可以实现set names latin1；**的效果，并且永久生效**

首先回答一个问题：至于为什么MySQL中的编码是latin1，是因为这是人家mysql规定好的，并写在自己的运行程序中的，只要mysql启动的时候，我们没有给人家指定一个字符集，那么它就会按照自己程序中写好的这个字符集来运行。

所以，我们如果想永久的更改mysql的字符集(不管是客户端还是服务端的)，就需要在mysql服务端启动之前给人家指定好，具体怎么指定呢，就需要看一下mysql启动的时候加载自己程序文件的过程(在并发编程的时候我们说过，任何程序运行都是通过加载自己的程序文件运行起来的)，看看加载了哪些文件，并且哪些文件是关于mysql字符集的，我们只需要将这个关于字符集的文件里面的内容修改一下，就相当于间接的告诉mysql运行加载的时候需要使用的字符集格式。那么我们就可以做到永久修改mysql的字符集了，对不对。好，针对这个思路，我们研究了一下发现，这个包含字符集配置的配置文件叫做my.ini文件(win10，unix叫做my.cnf)，这个文件是mysql启动的时候加载的一些用户自定制配置的文件，那么我们可以通过这个配置文件来改一改字符集，除了能填写字符集的配置项之外，还能填写一些你自己想要定制的其他的内容(需要研究mysql官方手册了~~)。

如果你的安装目录里面没有这个文件并且没有在其他地方设置，那么mysql就会按照自己默认的配置参数来运行，我们可以通过写一个my.ini文件来指定，mysql运行起来时会读取这个my.ini文件中的一些配置，其中就可以配置指定字符集。

我们知道这个文件叫做my.ini文件了，但是这个文件写在哪里呢？这里我们再说一个问题：有的人的博客上说mysql在启动的时候使用的配置文件在这个目录下(以win10举例)：

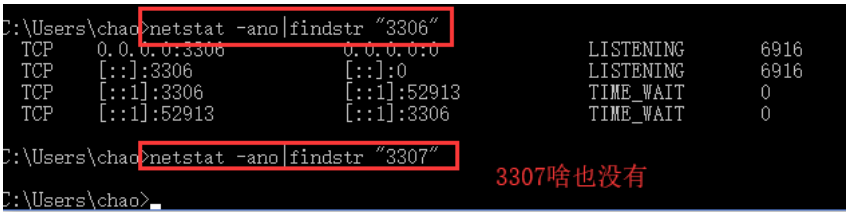


但是我查看了一下mysql启动的时候对my.ini文件的加载顺序：

```
C:\WINDOWS\system32>mysql --help|findstr my.ini
C:\WINDOWS\my.ini C:\WINDOWS\my.cnf C:\my.ini C:\my.cnf F:\jj\mysql-5.6.42-winx64\my.ini F:\jj\mysql-5.6.42-winx64\my.cnf

C:\WINDOWS\system32>
```

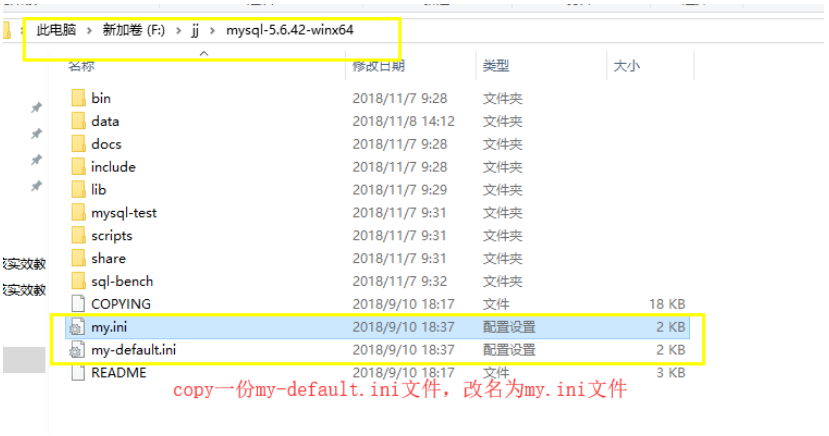
发现并没有加载上面这个文件夹中的my.ini文件，并且我测试了一下，将上面文件夹中my.ini文件中的mysqld下面的默认端口号3306改为了3307，然后我启动了mysql服务，然后我查了一下端口号，发现3307这个端口并没有被使用：



所以验证出上面的这个文件夹中的文件并没有被加载生效。所以我确定，上面这个文件夹中的配置文件只是一个参考用的，我称它为伪配置文件。并不是有些博客里面的mysql中加载的那个my.ini文件。

并且我们通过上面的查看my.ini文件的加载顺序中看到，我们自己mysql的安装目录中的my.ini文件就是其中一个加载顺序的结果，所以，看样子我们自己在自己的mysql的安装目录下写一个my.ini文件就应该能行了，来就按照这个思路搞一搞(和大家确定说一下，这样肯定是可以的~~~)

首先在安装目录下创建一个my.ini文件(copy一份my-default.ini文件，改名为my.ini文件)，使用Notepad++打开，里面写上下面的内容，来看看是不是会生效



之前我们连接mysql服务端的时候的指令是 mysql -uroot -p回车，其实我们直接输入mysql然后回车就能连接上mysql服务端，但是用户并不是root，而是mysql给我们创建一个用户，没啥用，可以忽略，工作中这个用户肯定是要被删除了，我们先来看一下这个用户，通过select user()来查看，并不是我们的root用户，root用户连接的时候是需要填密码的。

之前我们用root用户连接输入库的命令是这样的：

C:\Users\chao>mysql -uroot -p

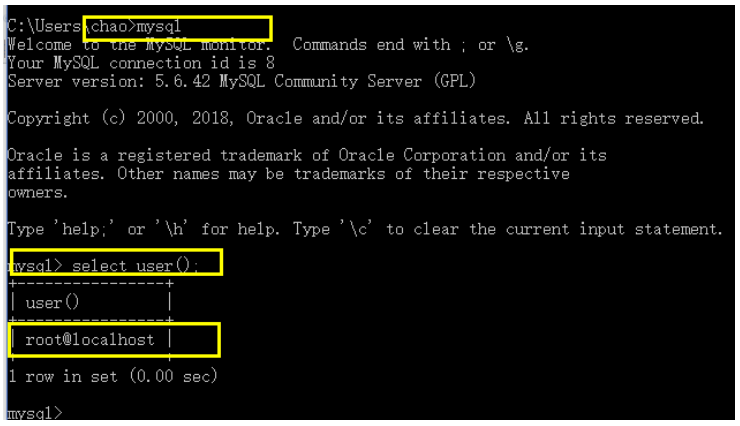
Enter password: ***

我们在我们创建的my.ini文件中写上下面几行，然后保存：

[mysql] #配置客户端连接的时候，指定一下用户名和密码，那么我们在进行mysql客户端连接的时候，直接输入mysql然后回车就可以了，并且用户是我们下面指定的root用户
user=root
password=666

重启mysql服务（一般修改配置文件，让其生效，需要重启服务，但是我测试了一下，这个用户名和密码的指定，貌似不需要重启服务就可以的）：

然后再起一个cmd客户端，直接输入mysql然后回车，再查看一下用户：



改为root用户了~~~~配置文件生效~~~~

通过上面的实验，我们知道，我们可以通过my.ini文件中的配置项，来更改mysql系统的一些服务，实现一些自定制配置，其实能够完成的配置非常多，将来深入学习mysql的时候，这个配置文件很关键，不过对于现在只做开发的你，就不必去研究那么多了，如果想看一下都可以进行哪些配置，可以参考一下我的那个centos7.1下安装mysql的博客最后面的内容，有关于其中的很多配置及解释，还可以参考上面我们提到的那个伪配置文件里面的内容来搞，但是记住一点，你写的这些配置必须是mysql能够认识的，也就是要按照人家规定的变量名称来配置，比如上面我们配置的用户名和密码，就叫做user和password，不能是username什么的，这个记住啦。

下面我们通过配置文件来搞一搞编码，终于到了这一步了(windows和linux都是这个配置)
在配置之前我们看一下各个角色的编码，还记得查看指令吗：

```
mysql> show variables like "%char%";
+--------------------------+----------------------------------------
-+
| Variable_name | Value |
+--------------------------+----------------------------------------
-+
| character_set_client | gbk |
| character_set_connection | gbk |
| character_set_database | latin1 |
| character_set_filesystem | binary |
| character_set_results | gbk |
| character_set_server | latin1 |
| character_set_system | utf8 |
| character_sets_dir | F:\jj\mysql-5.6.42-winx64\share\charsets\ |
+--------------------------+----------------------------------------
-+
8 rows in set (0.00 sec)
```

好，我们来改一改my.ini配置文件，文件中的内容写法，写完之后保存，然后重启mysql系统服务：

#强调：配置文件中的注释可以有中文，但是配置项中不能出现中文
#在mysql的解压目录下，新建my.ini,然后配置
#1. 在执行mysqld命令时，下列配置会生效，即mysql服务启动时生效
[mysqld]

character_set_server=utf8
collation-server=utf8_general_ci #就是一个校对规则，一般默认都是这个，如果不是就改成这个就可以了，所以直接写上就行了，这个规则后面我们会讲的~~~

还可以配置好多内容，比如下面的端口号，基准路径，数据文件路径：

port=3306 # mysql服务端默认监听(listen on)的TCP/IP端口

basedir="C:/Program Files/MySQL/MySQL Server 5.5/" # 基准路径，其他路径都相对于这个路径

datadir="C:/Program Files/MySQL/MySQL Server 5.5/Data" # mysql数据库文件所在目录

#2. 针对客户端命令的全局配置，当mysql客户端命令执行时，下列配置生效
[client]
default-character-set=utf8

#3. 只针对mysql这个客户端的配置，2中的是全局配置，而此处的则是只针对mysql这个命令的局部配置
[mysql]
user=root
password=666
default-character-set=utf8

#如果没有[mysql],则用户在使用mysql系统自带的mysql客户端来执行mysql命令时的配置以[client]为准

重启mysql服务，让配置文件生效：
C:\WINDOWS\system32>net stop mysql
MySQL 服务正在停止..
MySQL 服务已成功停止。

C:\WINDOWS\system32>net start mysql
MySQL 服务正在启动 .
MySQL 服务已经启动成功。

然后连接进入mysql，再次查看编码：
C:\Users\chao>mysql -uroot -p
Enter password: ***
mysql> show variables like "%char%";
+--------------------------+--
-+
| Variable_name | Value |
+--------------------------+--
-+
character_set_client	utf8
character_set_connection	utf8
character_set_database	utf8
character_set_filesystem	binary
character_set_results	utf8
character_set_server	utf8
character_set_system	utf8
character_sets_dir	F:\jj\mysql-5.6.42-winx64\share\charsets\
+--------------------------+--
-+
8 rows in set (0.00 sec)

编码都编程utf8了，verygood，编码统一了，我们再来插入一条数据看看：
首先我们将之前的crm库删除，然后再重新创建一个crm库，并在crm库里面重新创建一个

student表，为什么删除呢？因为前面我们说了，之前的数据是乱码的，没办法改，所以我们先删除吧，然后重新创建一个，之前创建的crm库是Latin1的字符集的，所以为了简单演示，我们就删除重新创建吧，这里注意，如果将来你们公司使用的数据库的编码确实有问题，并且和你们现在要使用的编码不一致，那么就需要使用第一种临时修改字符集的方式来插入和查询数据，没办法，要不然就要重塑数据库，将编码调节好。

来操作一下看看效果：

首先看一下之前的crm库的创建语句：

```
mysql> show create database crm\G
*************************** 1. row ***************************
Database: crm
Create Database: CREATE DATABASE `crm` /*!40100 DEFAULT CHARACTER SET latin1 */

1 row in set (0.08 sec)
```

还是latin1的字符集

```
mysql> drop database crm; #删除之前的crm库
Query OK, 1 row affected (0.41 sec)

mysql> show databases; #查看一下，crm库没有了
+--------------------+
| Database |
+--------------------+
| information_schema |
| mysql |
| performance_schema |
| test |
+--------------------+
4 rows in set (0.01 sec)

mysql> create database crm; #重新创建
Query OK, 1 row affected (0.00 sec)

mysql> show databases;
+--------------------+
| Database |
+--------------------+
| information_schema |
| crm |
| mysql |
| performance_schema |
| test |
+--------------------+
5 rows in set (0.00 sec)
```

查看创建时的语句：

```
mysql> show create database crm\G
*************************** 1. row ***************************
Database: crm
Create Database: CREATE DATABASE `crm` /*!40100 DEFAULT CHARACTER SET utf8 */

1 row in set (0.00 sec)
```

已经改为了utf8的字符集

然后我们插入数据，查看数据，看一下效果：

```
mysql> use crm;
Database changed
mysql> create table student(id int,name char(10),age int);
Query OK, 0 rows affected (0.47 sec)

mysql> insert into student value(1,'呵呵',11),(2,'老刁',12),(3,'dsb',10),(4,'你好',9);
Query OK, 4 rows affected (0.10 sec)
Records: 4 Duplicates: 0 Warnings: 0

mysql> select * from student;
+------+--------+------+
```

```
| id | name | age |
+------+--------+------+
| 1 | 呵呵 | 11 |
| 2 | 老刁 | 12 |
| 3 | dsb | 10 |
| 4 | 你好 | 9 |
+------+--------+------+
```

good，完全没有乱码了~~~这就是解决方案

总结：不乱码的思想：系统的编码、客户端、服务端、库、表、列，这几项的编码都要统一才不会出现乱码的情况。

附赠：

windows系统查看系统默认编码的指令：

cmd窗口情况下：windows下cmd默认的编码是GBK

想在windows下查看sqlite的utf-8中文需要先 执行chcp 65001把当前页换为utf-8编码

chcp 命令:

chcp 65001 就是换成UTF-8代码页，在命令行标题栏上点击右键，选择"属性"->"字体"，将字体修改为True Type字体"Lucida Console"，然后点击确定将属性应用到当前窗口

chcp 936 可以换回默认的GBK

chcp 437 是美国英语

linux系统查看系统默认编码的指令：

执行指令：cat sysconfig i18n

结果中有一条是：LANG="zh_CN.utf8"

六 初识sql语句

有了mysql这个数据库软件，就可以将程序员从对数据的管理中解脱出来，专注于对程序逻辑的编写。

mysql服务端软件即mysqld帮我们管理好文件夹以及文件，前提是作为使用者的我们，需要下载mysql的客户端，或者其他模块来连接到mysqld，然后使用mysql软件规定的语法格式去提交自己命令，实现对文件夹或文件的管理。该命令的语法即sql（Structured Query Language 即结构化查询语言），sql语句又分为几类，具体看我的博客：https://www.cnblogs.com/clschao/articles/9930802.html，主要分为4中，DDL、DQL、DML、DCL。

SQL语句主要是针对数据库里面三个角色进行操作，对象是：库、表、行，操作包括：增删改查。

1、库（data文件夹中的文件夹，每创建一个库，这个库的名称就是文件夹的名称，文件夹里面保存着一些这个库相关的初始信息）

增：create database db1 charset utf8; #创建一个库，可以指定字符集

查：show databases; #查看数据库中所有的库

show create database db1; #查看单独某个库db1的信息

改：alter database db1 charset latin1; #修改库的字符集，注意语句的格式（其他语句也是这么个格式），alter（修改） database（修改数据库） db1（哪个数据库） charset（字符集） latin1（改成哪个字符集）

删除: drop database db1; #删除数据库

2、表（操作文件，表是上面库文件夹里面的文件）

先切换库：use db1； #要操作表文件，要先切换到对应的库下才能操作表

查看当前所在的是哪个库：select database();

增：create table t1(id int,name char(10))；#创建表的时候，和excel一样，需要有字段啊，每个字段还需要只能一下这个字段数据的格式，这里指定的是两个字段列，id和name列，id和name是列名(字段名)，id 后面的int的意思说id这一列中的数据只能是int类型的，name后面的char的意思是，name这一列中的数据只能是char类型的(char表示定长字符串类型)，char里面的10是说这个字段的长度最长为10个字符，如果不指定这个长度，默认长度是1，注意是字符而不是字节，这些字段的内容我们后面会详解，这里知道一下就好啦。

#在创建表的时候，我们去看一下mysql安装目录里面的data文件夹里面的db1文件夹里面的文件，然后我们执行创建表的指令，看看db1文件夹里面的变化，多了两个文件，分别是：db1.frm，db1.ibd文件，创建了一张表为什么会多了两个文件呢，这两个文件都是啥呢？看解释（里面涉及到存储引擎，关于存储引擎我们后面会讲的~~）：

1.后缀名为.frm的文件：这个文件主要是用来描述数据表结构(id,name字段等)和字段长度等信息
2.后缀名为.ibd的文件：这个文件主要储存的是采用独立表储存模式时储存数据库的数据信息和索引信息；
3.后缀名为.MYD（MYData）的文件：从名字可以看出，这个是存储数据库数据信息的文件，主要是存储采用独立表储存模式时存储的
4.后缀名为.MYI的文件：这个文件主要储存的是数据库的索引信息；
5.ibdata1文件:主要作用也是储存数据信息和索引信息，这个文件在mysql安装目录的data文件夹下。
从上面可以看出，.ibd储存的是数据信息和索引信息，ibdata1文件也是存储数据信息和索引信息，.MYD和.MYI也是分别储存数据信
主要区别是再于数据库的存储引擎不一样，如果储存引擎采用的是MyISAM，则生成的数据文件为表名.frm、表名.MYD、表名的MY
在进行数据恢复的时候，如果用的是MYISAM数据引擎，那么数据很好恢复，只要将相应.frm, .MYD, .MYI文件拷贝过去即可。但是msyql人家设定的规则就是这样存储表的，使用人家的系统，就要理解人家的规则。

查：show tables； #查看当前库中所有的表

show create table t1; #查看单表的创建信息

#还可以通过下面两句来查看表信息，以表格的形式展示结果：

desc t1；

describe t1；#上下这两句是一样的结果

改：alter table t1 modify name char(3); #修改字段属性的，将name字段的char长度改为3，改完之后我们在用上面的show create table t1；desc t1；describe t1；来查看一下修改结果。

alter table t1 change name name1 char(2);

删：drop table t1;

3. 行（操作文件（表）中的内容/记录）（*****将来的重中之重）

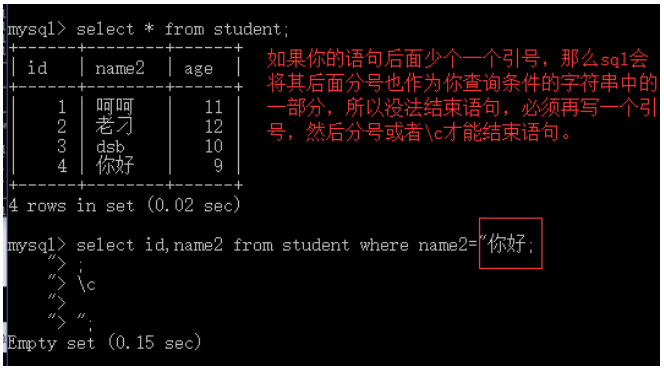
增：insert into t1 values(1,'dsb1'),(2,'dsb2'),(3,'dsb3'); #往t1表中插入三行数据，注意你插入的每行内容都要和你创建表的时候的字段个数和字段属性对应好，注意每行数据以逗号分隔。

insert后面的into可以不用写。

查：select * from t1; #查看t1表中所有字段的数据，select 字段 from 表。

select id,name from t1;#查看t1表中的id和name列的数据，其他的不看，注意格式，每个字段逗号分隔，在cmd窗口下只是展示给我们看，将来我们通过程序获取查询数据的时候，就可以这么获取，查询字段的顺序也是可以颠倒的，name,id这样也是可以的。

注意还有一个问题，看下图：当你写sql语句的时候，可能会出现下面这种情况，由于少写了一个引号，导致怎么也结束不了



改：update t1 set name='sb' where id=2; #把id为2的行（记录）中的name字段的数据改为sb；id>1;id<=1;等等都可以。后面会细讲的~~~

update t1 set name='sb',id=88 where id>2; #对两个字段进行修改

update t1 set name='sb';#如果不指定where，那么会name字段的所有数据都改成sb。

删：delete from t1 where id=1; #删除id为1的行

清空表：

delete from t1; #如果有自增id，新增的数据，仍然是以删除前的最后一样作为起始。

truncate table t1;数据量大，删除速度比上一条快，且直接从零开始，

auto_increment 表示：自增

primary key 表示：约束（不能重复且不能为空）；加速查找

至此，我们大家认识了一下简单的SQL语句，下来大家练一练吧~~~明天我们针对库、表、行的操作来一些详细的讲解，其实库的内容不多，主要是表和行，最主要是行。