**数据库学习入门数据库基础入门**

**数据库学习入门**数据库基础入门

　　-------------------

　　数据库的作用：

　　-------------------

　　数据是数据库中存储的基本对象

　　包括：文字、图形、图像、声音

　　有组织的、可共享的数据集合

　　------------------

　　常用的DBMS

　　------------------

　　MYSQL（oracle、Sun）

　　oracle（oracle）

　　SQL-Server（Microsoft)

　　DB2（IBM）

　　-----------------------------------

　　..如何学习数据库管理系统

　　------------------------------------

　　.程序员

　　1、标准语言：SQL（结构化查询语言）

　　SQL是用于访问 数据库的最常用标准化语言，他是由ANSI/ISO SQL标准定义的

　　SQL 的分类

　　DML：数据操作语言（用于检索或者修改数据）

　　DDL：数据定义语言（用于定义数据的结构，比如创建、修改或者删除数据库的对象）

　　DCL：数据控制语言（用于定义数据库用户的权限）

　　DML:

　　select:用于检索数据

　　insert:用于增加数据到数据库

　　update:用于从数据库中修改现存的数据

　　delete：用于从数据库中删除数据

　　DDL:

　　create table //创建表

　　alter //修改表

　　drop table //删除表

　　create index //创建索引

　　drop index //删除索引

　　create view //创建视图

　　drop view //删除视图

　　DCL：

　　grant //授权

　　revoke //撤销授权

　　set //设置

　　2、不同数据库的独特管理方式

　　.DBA（Database Administrator）

　　1、标准语言：SQL（结构化查询语言）

　　2、不同数据库的独特管理方式

　　3、数据库的调优

　　4、精通数据库的备份、回复机制

　　-----------------------------

　　MYSQL的安装与配置

　　----------------------------

　　安装：选择普通安装类型

　　.配置

　　1、选择配置类型：Detailed Configuration

　　2、选择服务器类型：Developer Machine

　　3、选择数据库的是使用情况：

　　--多功能数据库（Multifunctional Database）

　　--只是事务处理数据库（Transactional Database Only）

　　--只是非事务处理数据库（Non-Transactional Database Only）

　　4、InnoDB表空间文件存储的盘符，默认在c：\

　　5、数据库并发连接数配置：Decision support

　　6、MYSQL服务器端口：默认：3306

　　7、Enable Strict Mode：是否允许严格样式检查：取消

　　8、选择字符编码：手工选择：gb2312，windows下中文支持

　　9、服务器默认安装到windows系统服务中，默认服务自启动，选择自动配置环境变量

　　10、填写数据库管理系统的管理员密码

　　--在mySQL中数据库管理员的用户名为：root

　　11、是否允许使用root用户远程登录数据库

　　12、测试配置是否正确c：\>c:\mysql\bin\mysqlshow

　　------------------------

　　MYSQL 的登录

　　------------------------

　　.开始--运行--cmd

　　.输入：mysql -uroot -p，然后输入密码

　　.或者：mysql -uroot -p密码

　　.退出quit或者\q

　　--不必全在一个行内给出一个命令，较长命令可以输入到多个行中。mysql通过寻找终止分号而不是输入行的结束来决定语句在哪儿结束

　　->等待多行命令的下一行

　　'>等待下一行，等待以单引号开始的字符串结束

　　如果你决定不想执行正在输入过程中的一个命令，输入\c取消它

　　能够以大小写输入关键词，结果是等价的

　　--------------------

　　基本的SQL语句

　　--------------------

　　创建数据库---create database school；

　　查看有哪些数据库--show databases；

　　删除数据库---drop database school；

　　--------------------------

　　向数据库中存储数据

　　--------------------------

　　1、首先进入数据库 use dbname；

　　use类似quit，不需要一个分号，写也无碍

　　use必须在一个单行上给出

　　2、查看数据库中的表 show tables；

　　3、在数据库中创建表 create table student

　　（name varchar（20），age int，sex char（2））；

　　4、向表中插入数据---insert into dbname values（‘张三’，18，‘男’）；

　　5、查看表中所有的数据--select \* from student；

　　只查看姓名和年龄---select name，age from student；

　　--------------------

　　常用的数据类型

　　--------------------

　　char（M）固定长度字符

　　用来表示固定长度字符串，字符串的长度是1-255

　　VARCHAR（M）可变长度字符

　　具有灵活性的字符数据类型，范围：1-255

　　处理char类型的数据比varchar类型的数据要快，有时会快50%（char类型存储的数据的长度是声明变量时的固定长度，而不管数据的实际长度，varchar存储的是按数据的实际长度，从而减小了数据文件的大小）

　　int 整数 有符合的范围：-2147483648到2147483647，无符号范围：0到4294967295

　　--------------------------

　　其他常用的数据类型

　　--------------------------

　　DATE---日期类型，显示‘YYYY-MM-DD’

　　DATETIME--日期和时间的组合，显示‘YYYY-MM-DD HH:MM:SS'

　　TEXT/BLOB 文本和大对象

　　TEXT可以保存字符串的长度在255-65535字节内。BLOB是一个能保存二进制数据的大对象，区别是TEXT不区分大小写，而BLOB区分大小写。

　　SQL语句的导入

　　1、编写SQL脚本（.sql）

　　2、导入mysql-uusername -p < \*\*\*.sql

　　&练习：

　　&老师表：teacher

　　&姓名，性别，年龄，课程

　　导出sql脚本---mysqldump stu -uroot -p>abc.sql

　　查看表的结构------desc tablename

　　修改表中的数据---update tablename set colname=value where条件

　　删除数据-----delete from teacher where age=？

　　删除表中所有数据---delete from teacher；

　　删除表---drop table tablename

　　根据条件进行过滤查找---select \*from tablename where 条件

　　查找出版社为“清华出版社”的所有书籍---select \*from books where pub=‘清华出版社’；

　　查找出库存大于50的所有书籍---select \*from books where store>50；

　　查找出“西游记”的库存量---select title，store from books where title=‘西游记’；

　　SQL运算符

　　>大于 <小于 >=大于等于 <=小于等于 !=,<>不等于

　　查看数据库的信息----\s

　　-------------------

　　mysql常用函数

　　-------------------

　　1、查看数据库版本---select versin（）；

　　计算机的时间是存在BIOS（）

　　2、查看当前数据库的日期----select current\_date（）；

　　3、查看当前数据库时间----select now（）；

　　4、查看当前连接数据库的用户---select user（）；localhost：代表是本机

　　create table user(id int,name varchar(20),bir date,dea datetime);

　　insert into user values(1,'zhangsan',now(),now());

　　or和and查询

　　or（满足一个条件）

　　and（都需要满足）

　　查找学生EQ为80分或90分的学生

　　select \* from stu where EQ=80 or EQ=90;

　　查找学生EQ为90分并且性别为女的学生

　　select \* from stu where EQ=90 and sex='女';

　　in（x，x）返回条件中的记录与or作用相似

　　select \* from tablename where 条件 in（，）；

　　between‘’ and‘’返回两者之间的记录

　　查询年龄在20-30之间的所有学生

　　select \* from stu where age between 20 and 30；

　　like与通配符（%）一起使用，模糊查询

　　查找出姓张的学生

　　select \* from stu where name like ‘张%’；

　　查询出使用163邮箱的所有学生

　　select \* from stu where email like ‘3.com’；

　　查询出邮箱里面含有a的所有学生

　　select \* from stu where email like ‘%a%’；

　　order by 实现排序（从小到大）--asc

　　将学生的年龄从高到低排列

　　select \* from stu order by age desc；（降序）

　　将学生的年龄从低到高排列

　　select \* from stu order by age asc；（升序）

　　数据默认为升序（从低到高）

　　select \* from stu order by age ；

　　as为查询的列起别名

　　1、查询所有学生只列出姓名，年龄，性别

　　select name as '姓名',age as '年龄',sex as '性别' from stu;

　　group by对于查询出的数据结果进行分类（分组）

　　2、将学生按性别进行分类

　　select \* from stu group by sex；

　　将学生按年龄进行分类

　　select \* from stu group by age；

　　3、having 子查询：对于where查询出的结果再次进行查询

　　查找出年龄大于20岁学生，并且在其中找出姓名等于xxx的学生

　　select \* from stu where age >20 having name='xxx'或

　　select \* from stu where age>20 and name='xxx'

　　4、distinct 过滤查询的重复型记录，只显示唯一的记录

　　将学生性别过滤

　　select distinct(sex) from stu;

　　count 查看表中有几条数据

　　select count(\*) from stu;

　　select count(distinct(sex)) from stu;

　　limit 限制查询结果的输出数量 同时也可以实现数据的分页

　　查询EQ前三名的学生

　　select \* from stu order by EQ desc limit 3;

　　实现查询记录的分页

　　select \* from stu limit 0,3;

　　select \* from stu limit 3,3;

　　约束----定义了表级的强制规则、数据的完整性

　　非空约束（not null） create table test(id int not null); insert into test values();

　　innodb 会报错 ，myISAM 会整形默认以0填充

　　唯一约束（UNIQUE) 不允许列中的数据重复

　　create table test(id int,unique(id));

　　insert into test values(1);

　　默认约束(default) create table test(id int not null default'111111'); insert into test values();

　　主键约束(primary key)是一个字段或一组字段（组合键），用于唯一标识表中的记录，它可以 确保每个记录是唯一的。

　　create table test(id int,primary key(id));

　　insert into test valuse(1);

　　id主键自增,减

　　create table test (id int，name varchar（20），primary key（id）)；

　　insert into test values(1,'张安')；

　　auto\_increment的特点-------自增值 当删除某一值时，他不会自动填充，而是继续自增1

　　create table test(id int not null auto\_increment,name varchar(20),primary key(id));

　　insert into test(name)values('xxx')；

　　创建一个有合理约束的表

　　create table people(id int not null auto\_increment,name varchar(20) not null,age int not null,sex char(2) not null,pcode varchar(50),tel varchar(50),email varchar(50),primary key(id));

　　insert into people(name,age,sex,email) values('张三'，‘28’，‘男’，'abc@163.com');

　　查询出所有学生的档案信息

　　select \* from student,school where student.daih=school.id;或

　　select \* from student as a,school as b where a.daih=b.id;

　　查询所有学生档案信息 只需显示：学生姓名，年龄。毕业学校，学校地址

　　select student.name,student.age,school.name,school.addres s from student, school where student.daih=school.id;或

　　select a.name,a.age,b.name,b.address from student as a, school as b where a.daih=b.id;

　　外键约束( foreign key)

　　如何创建外键

　　foreign key:定义子中的列为外键

　　references:标记外键所要参考的父表和列

　　on delete cascade:允许在删除父表的列的同时，删除子表的列 //在InnoDB中支持

　　create datebase fordb;

　　use fordb;

　　创建父表

　　create table school(id int not null auto\_increment,name varchar(20),primary key(id))engine=innodb;

　　insert into school(name) values('紫琅学院');

　　创建 子表

　　create table student(id int not null auto\_increment,name varchar(20),schoolid int,primary key(id),foreign key(schoolid) references school(id) on delete cascade);

　　insert into student(name,schoolid) values('张三',1);

　　E-R模型

　　概述：设计数据库时，通常采用"实体关系模型“--E-R模型

　　软件开发流程：

　　需求调研（设计功能，收集数据）

　　与最终用户进行确定

　　数据库的设计

　　控制多余数据

　　那么在最后，数据库设计者确定表、不同表之间的关系以及每个表之间的关系，通常使用”E-R模型“，它将整个系统看作彼此相关的实体组成

　　实体：通常用于表示 能够被清晰识别的人、地点、事物、对象、事件

　　实体关系图：如果需要 基于‘E-R’模型建立数据库模型，需要标识实体，实体的属性、以及实体之间的关系。那么通常用‘E-R’图来表示

　　实体之间的三种类型：1：1 1：N 或N :1 M:N

　　数据库设计员确定的实体被转换为表，而其属性则成为相应 表中的字段（列）

　　如何控制冗余数据：

　　一般来说通过数据库的范式理论

　　设计数据库的范式来 控制冗余

　　共有五个范式，一般达到第三范式即可

　　第一范式：对于表中的每一行，必须且仅仅有唯一的行值，在一行中的每一列仅有唯一的值并且具有原子性

　　第二范式 ：要求 非主键列是主键的子集，非 主键列活动必须完全依赖整个主键。主键必须有唯一性的元素，一个主键可以由一个或更多的组成唯一值的列组成。一旦创建 ，主键 无法改变，外键关联一个表的主键。主外键关联意味着一对多的关系。

　　第三范式：要求非主键列互不依赖

　　第四范式：禁止主键列和非主键列一对多关系不受 约束

　　第五 范式 ：将表分割成尽可能 小的块，为了排除在表中所有的冗余

　　MYSQL的聚合函数

　　1、最大值

　　找出EQ最高的学生

　　select name ,eq from student where eq=(select max（EQ）from student)；

　　SELECT MAX(article) AS article FROM shop;

　　2、最小值

　　找出EQ最低的学生

　　select name，min（EQ）from student；

　　count（）//统计 查询出的记录总数

　　3、查询出学生的总条数

　　select count（name）from student；

　　avg（）//求平均值

　　4、查询学生EQ的平均值

　　select avg（EQ） from student；

　　sum（）//求和

　　5、查询学生EQ的总和

　　select sum（EQ） from stu；

　　修改字段类型

　　alter table student modify sex char（5）；

　　增加列

　　alter table student add address varchar（50）；

　　删除列

　　alter table student drop address；

　　修改列的名称

　　alter table student change name names varchar（20）；

　　修改表的名字

　　rename table student to stu；

　　MYSQL 之表连接

　　内链接(inner join)：又为等值连接，因为他将两个表中的公共字段有相等的值（记录）连接起来，这是最常用的连接。一个表引用还被称为

　　查询显示：订单编号，顾客姓名，物品名称

　　select cases.id,user.name,goods.name from cases,user,goods where cases.user\_id=user.id and cases.goods\_id=goods.id;或

　　select cases.id,user.name,goods.name from cases inner join(user,goods) on(user.id=cases.user\_id and goods.id=cases.goods\_id);或

　　select cases.id,user.name,goods.name from (cases inner join user on cases.user\_id=user.id) inner join goods on cases.goods\_id=goods.id;或

　　select cases.id,user.name,goods.name from user inner join(cases inner join goods on cases.goods\_id=goods.id) on user.id=cases.user\_id);

　　左连接:显示sql语句中left join 左边表中的所有记录，即使在left join 右边的表中没有满足连接条件的数据也被显示。当满足连接条件时，left join 右边的表中的相应的记录与left join左边表中的相应记录结合显示。

　　查询出：学生的编号，学生姓名 ，学生学校

　　select student.name,student.id,school.name from student left join school on student.school\_id=school.id;

　　右连接：与左连接相对应，他显示sql语句中right join 右边表中的记录，即使在right join 左边没有的记录也被显示。当满足条件时。right join左边表中相应记录将与right join右边的表中相应记录进行结合显示。通常左连接和右连接显示的内容是一样的。

　　select student.name,student.id,school.name from student right join school on student.school\_id=school.id;

　　left join 以左边的表查询为主

　　right join 以右边的表查询为主

　　示例：

　　student left join school

　　那么student就为左表。school就为由表

　　MYSQL之视图

　　视图：它可以访问一个或多个表中的数据，是从一个或多个表中派生出的数据对象（虚表）

　　视图的特点：

　　1、将复杂的查询简单化

　　2、提供安全机制，它保证用户只可以看得到的数据，系统中真实的表是不可被存取的（现都支持更改数据）

　　创建视图：

　　create view case\_view as select cases.id as '订单编号',user.name as '顾客姓名',goods.name as '物品名称' from user inner join(cases inner join goods on cases.goods\_id=goods.id) on user.id=cases.user\_id;

　　注意点：

　　视图属于数据库，在默认情况下，将在当前数据库创建新视图。要想在给定数据库中明确创建视图。创建时，应将名称指定为

　　db\_name.view\_name

　　mysql>create view test.v as select \*from t;

　　视图必须具有唯一的列名，不得有重复，就像基表那样。

　　修改视图:alter view viewname as SQL;

　　删除视图：drop view viewname;

　　alter view case\_view as select cases.id as '编号',user.name as '姓名',goods.name as '名称' from user inner join(cases inner join goods on cases.goods\_id=goods.id) on user.id=cases.user\_id;

　　数据字典

　　数据库的数据字典---实际上是一个数据库

　　它记录着数据库管理系统中的其他数据库的操作

　　mysql数据库字典：information\_schema

　　use information\_schema

　　select \* from views;

　　---------------------------

　　MYSQL的事务与引擎

　　---------------------------

　　什么是 事务（transaction）

　　所谓事务，它是一个操作序列。这些操作要么都执行，要么都不执行，他是一个不可分割的工作单位

　　事务是数据库维护数据一致性的单位，在每个事务结束时，都能保持数据一致性

　　mysql表类型：

　　在mysql中有多种表的类型，但是分为两类 :

　　事务类型：InnoDB、BDB

　　非事务类型：MYISAM、MERGE、MEMORY（HEAP）

　　InnoDB和BDB存储引擎提供事务安全表。BDB被包含在mac支持它的操作系统发布的mysql-max二制分发版里。InnoDB也默认被包括在所有mysql5.1二进制分发版里

　　InnoDB是为处理巨大数据量时的最大性能设计。它的cpu效率可能是任何其他基于磁盘的关系数据库引擎所有不能匹敌的。

　　MYISAM管理非事务表。他提供高速存储和检索，以及全文搜索能力。

　　ISAM数据表是mysql最原始的数据表，有三个文件，分别是：

　　.frm，存放数据表的结构定义

　　.isd，数据文件，存放数据表中的各个数据行的内容

　　.ism,索引文件，存放数据表的所有索引信息。

　　MYISAM是数据表是ISAM数据表的继承者，也是三个文件，分别是：

　　.frm,结构定义文件；

　　.MYD，数据文件；

　　.MYI，索引文件。

　　BDB数据表用两个文件 来表示，分别是;

　　.frm，结构定义文件

　　.DB，数据表数据和索引文件

　　INNODB:由于采用表空间的概念来管理数据表，所以他只有一个与数据表对应.frm文件，同一目录下的其他文件表示为空间，存储数据表的数据和索引

　　HEAP数据表是一个存在与内存中的表，所以他的数据和索引都存在于内存中，文件系统中只有一个.frm文件，及定义结构，当mysql关闭后数据都将消失。

　　定义表的的类型

　　create table test（id int）engine=heap；

　　事务表币非事务表有几大优势：

　　更安全。即使mysql崩溃或遇到硬件问题 ，要么自动恢复，要么从备份加事务日志恢复，你可以取回数据

　　你可以合并许多语句，并用commit语句同时接受她们全部（如果autocommit被禁止掉）

　　你可以执行rollback来忽略你的改变（如果autocommit被禁止掉）

　　如果更新失败，你的所有改变都变回原来。（用非事务安全表，所有发生的改变都是永久的）

　　事务安全存储引擎可以给那些当前用 读得到许多更新的表提供 更好的部署。

　　非事务表自身有几大优点：

　　非事务表自身有几大优点，因为没有事务开支，所有优点都能出现：

　　更快

　　需要更更少的磁盘空间

　　执行更新需要更少的内存

　　你可以在同一个语句中合并事务安全和非事务安全表来获得两者最好的情况。尽管如此，在autocommit被禁止掉的事务里，变换到非事务安全表依旧即使提交并且不会被回滚

　　如何在innodb中实现事务：

　　begin://告知服务器我要开始一个事物

　　rollback://如果事务发生异常，那么数据回滚

　　commit://事务执行成功，进行数据提交

　　mysql> begin;

　　Query OK, 0 rows affected (0.00 sec)

　　mysql> insert into test11 values(1,'zhangs');

　　Query OK, 1 row affected (0.00 sec)

　　mysql> select \* from test11;

　　+------+--------+

　　| id | name |

　　+------+--------+

　　| 1 | zhangs |

　　+------+--------+

　　1 row in set (0.00 sec)

　　mysql> end;

　　ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server

　　version for the right syntax to use near 'end' at line 1

　　mysql> rollback;

　　Query OK, 0 rows affected (0.00 sec)

　　mysql> select \* from test11;

　　Empty set (0.00 sec)

　　mysql> commit;

　　Query OK, 0 rows affected (0.00 sec)

　　mysql> select \* from test11;

　　Empty set (0.00 sec)

　　修改表类型

　　sql语句：alter table tablename engine=innodb；

　　注意：不可以将mysql系统表比如‘user’或‘host’转换成innodb类型。系统表必须为myisam类型

　　对myisam进行事务处理---锁

　　lock table tablename（read，write）

　　lock table 可以对表进行加锁控制，以保证用户并发访问时非事务表的数据一致性

　　unlock tables 释放锁

　　mysql 之index（索引）

　　什么是索引？

　　索引被用来快速找出在一个列上用一特定值的行。没有索引，mysql不得不首先以第一条记录开始并然后读完整个表直到他找出相关的行，表越大，花费时间越多。如果表对于查询的列有一个索引，mysql能快速到达一个位置去搜寻到数据文件的中间，没有必要考虑所有数据。

　　索引的作用：

　　快速找出匹配一个where子句的行

　　当执行连接时，从其他表检索行

　　对特定的索引列找出max（）或min（）值

　　mysql中的索引：

　　索引实际上是一个分离的列表，具有一个指向全表的指针

　　myisam表，索引被存储为分离文件

　　innodb，索引存储为表空间的一部分

　　mysql有四种类型的索引：主键（primary key）、唯一索引（unique）、全文索引（full index）、普通索引（index）

　　创建index

　　1、create index indexname on tablename（columnname）；

　　indexname（索引名称）

　　tablename（表名称）

　　columnname（要创建索引的字段名称）

　　2、alter table tablename add index indexname（columnname）；

　　查看索引----show keys from tablename；

　　删除index--alter table tablename drop index indexname；

　　full index（myisam支持）

　　create table chat（id int，chat1 text，chat2 text）；

　　create fulltext index index\_text on tablename（col1，col2）；

　　alter table tablename add fulltext（col1，col2）；

　　高效使用索引：

　　太多的索引，占用空间

　　每次进行插入或更新时，索引都必须针对变化

　　重新排序，会导致 很多额外负担

　　何处使用索引？

　　1、where从句中条件匹配的行

　　select name from student where comment like“a%”；

　　comment域的索引就起到了作用

　　Explain 分析索引

　　语法： explain select age from test12;

　　Explain 分析mysql中的索引：

　　table：查询中涉及的表

　　type：显示连接中使用了何种类型

　　const（最好），eq\_ref，fef，range，index和all possible\_keys：可能用到的索引

　　key：实际使用的索引，可以在select语句中使用use index（indexname）来强制使用索引，用ignore index（indexname）来强制忽略索引key\_len：使用的索引长度，越短越好；

　　ref(references)：显示索引哪一列被使用了

　　rows：mysql必须检查用来返请求数据的行数

　　extra：using temporary（需要临时表来存储结果，通常发生在对不同的列集进行order by），usig filesort 意即mysql根本 不能使用索引

　　索引的算法：btree bitmap

　　使用索引注意的事项;

　　1、合理创建索引，反之会降低数据库的查询效率，反之创建的索引会失去意义。

　　mysql的备份方式：backup、 restore 、copy//冷备份、mysqldump、select into

　　backup备份myisam表的

　　示例：

　　>use test;

　　>backup table chat to 'c:\\db\_backup';

　　只生成 .frm 和 .myd,可以在数据库恢复后重建索引

　　>drop table chat;

　　>restore table chat from 'c:\\db\_backup';

　　copy备份

　　1、停止mysql服务器一避免 在备份的时候有用户进行数据库的访问

　　1、复制数据库的文件夹

　　用mysqldump备份

　　备份库

　　1、备份

　　#mysqldump -uroot -p dbname

　　>xxxname.sql

　　再打开查看（创建表备份的sql语句）

　　2、恢复

　　#mysql -uroot -p dbname <>

　　备份某个表

　　1、备份

　　#mysqldump -uroot -p dbname tablename

　　>xxxname.sql

　　再打开文件进行查看（创建表备份的sql语句）

　　2、恢复

　　#mysql -uroot -p dbname <>

　　用select into 备份

　　备份：mysql>select \*from tablename into outfile 'c:\\db\_backup\table.dat'

　　删除表：

　　mysql>delete from tablename;

　　恢复：

　　mysql>load data infile 'c:\\db\_backup\table.dat' into table tablename;

　　作业：

　　1、唯一索引

　　2、全文索引

　　要求：理解她们的意思及用途，以及实验

　　3、select into备份

　　MYSQL其他客户端工具

　　查看数据库中有哪些数据库

　　mysqlshow -uroot -p

　　查看数据库中有哪些表

　　mysqlshow -uroot -p dbname

　　查看数据库中表的结构

　　mysqlshow -uroot -p dbname tablename

　　mysqlcheck客户端可以检查和修复myisam表，他还可以优化和分析表

　　1、mysqlcheck -uroot -p dbname //检测

　　2、mysqlcheck -uroot -p --auto-repair dbname //如碰到有问题的表自动进行修复

　　3、mysqlcheck -uroot -p --optimize //优化表

　　MYSQL 管理方式

　　连接方式----mysql -hlocalhost -uroot 通过本地主机，以root用户访问，无需密码验证

　　mysql -hlocalhost 通过本地主机，匿名用户访问，无需密码验证

　　设置密码----mysqladmin

　　外部：

　　1、修改密码--mysqladmin hlocalhost -uroot -p password "newpassword"

　　2、密码为空--mysqladmin -hlocalhost -uroot -p password ""

　　3、设置root用户远程访问密码--mysqladmin -hremote -uroot -p password ""

　　内部设置密码：

　　1、设置root用户本地登录密码

　　set password for 'root'@'localhost'=password('000000');

　　2、设置root用户远程登录密码

　　set password for 'root'@'remote'=password('newpassword');

　　加密：

　　mysql> use start

　　Database changed

　　mysql> create table users(username varchar(50),password varchar(50));

　　Query OK, 0 rows affected (0.16 sec)

　　mysql> insert into users values('abcd','123456');

　　Query OK, 1 row affected (0.34 sec)

　　mysql> insert into users values('abcdfg',password('123456'));

　　Query OK, 1 row affected (0.00 sec)

　　mysql> select \* from users; select user，host，password form user；

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

　　| username | password |

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

　　| abcd | 123456 |

　　| abcdfg | \*6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9 |

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

　　2 rows in set (0.00 sec)

　　系统中的权限表

　　user表：记录允许连接到服务器的用户帐号信息，里面的权限是全局 级的

　　db表：记录各个帐号在各个数据库上的操作权限

　　table\_priv记录数据表级的操作权限

　　columns\_priv：记录数据列级的操作权限

　　host表：配合db权限表对给定主机上数据库级操作权限作更细致的控制。这个权限表不受grant和revoke语句的影响

　　-------------------------------------------------- --------------------------------------------------- --------------------------------

　　设置密码---修改user表

　　直接修改user表

　　mysql>use mysql;

　　mysql>update user set password=password('newpassword')where user='root';

　　mysql>flush privilegs;重新加载权限表，使修改立即生效

　　添加用户

　　mysql> use mysql

　　mysql> insert into user(host,user,password)values('localhost','test', password('222222'));

　　mysql> flush privileges;

　　授权：grant all on \*.\* to 'test'@'localhost'identified by '222222'

　　grant select on stus.\* to 'test'@'localhost'identified by '121212'

　　grant privileges（columns）

　　privileges表示授予的权限，columns表示作用的列（可选）

　　on what

　　设置权限级别：全局级、数据库级、表级、列级

　　to account

　　权限授予的用户:用'user \_name'@'host\_name'这种用户名、主机名格式、identified by 'password'设置密码

　　privileges有哪些

　　alter 修改表和索引

　　create 创建数据库和表

　　delete 删除表中以有的记录

　　drop 删除数据库和表

　　inde 创建和删除索引

　　insert 向表中插入数据

　　select 检索表中的数据、

　　update 修改表中的记录

　　file 读写服务器上的数据

　　process 查看服务器中执行的线程信息或杀死线程

　　reload 重载授权表或清空日志、主机缓存或表缓存

　　shutdown 关闭服务器

　　all 所有权限

　　revoke取消授权

　　mysql>revoke privileges(columns)on what

　　from account;

　　注：revoke可删除权限，但不能删除帐号，即使帐号已经没有任何权限。所以user数据表里还会有该帐号的记录，要彻底删除帐号，需使用delete命令手工删除user表中的相关数据。

　　示例：

　　revoke all on \*.\*from'test'@'localhost';

　　delete from user where user='test';

　　细节权限的分配与撤销

　　grant select,delete,update on student,test.\*to'erich'@'localhost'identified by 'test'

　　运行结果是，erich用户只能通过‘test‘密码从本机访问student，test数据库，并且没有insert权限

　　revoke select,delete,update on student.\*from'erich'@'localhost';

　　运行结果是，将erich用户在student数据库中的select，delete，update权限撤销