[Mysql基础笔记](https://www.cnblogs.com/lzhlearn/p/5877113.html)

连接Mysql服务器

0-1360675084.png

  注意 ：-h指的是host地址 ； -u指的是user ， 之所以打到-p回车再输入密码，是为了防止密码泄露；如果-h 不写则默认是localhost

★服务器->数据库 database -> 多张或一张表 table（建表的同时就是声明列）

★基本的语句

(注意分号结束)

1、查看当前服务器有什么数据库：show databases

2、创建数据库    ：create database

3、删除数据库    ：drop database 数据库名

      删除表              ：drop table 表名

4、选择要操作的数据库    ： use 数据库名

5、查一个数据库有什么表 ： show tables；

6、修改表名 ： rename table 旧表名 to 新表名    （不能修改数据库的名字）

7、**查看一个表的结构**      ： desc 表名   （仅是查看列的结构，不含数据）

8、**创建表并添加列**     ：  自定义表名，同时下面设置列名和类型

               Create  table 表名(

                  Num int ，

                  Name char(10)

                  Age int ) ;

以上是实例 ， 注意每个属性用逗号进行隔开，最后一分号结束

             Ps ：age int unsigned zerofill à zerofill 表示0填充并且默认unsigned

**插入数据的语句 ：**

　　　　　Insert  into 表名

            （num,name,age）values(1,’张三’,22)，（2,’王四’,25）;

  注意 ：如果插入全部属性则前者可以省略 , 如果要选择插入某些属性则修改前者即可values 有个 s, 且必须有

注意values的拼写

**插入新列的语句 ：**

         Alter table 表名 add 列名 类型 限制 after 列名

         限制可以是 ：（１）unsigned 无符号

         After 是为了声明在哪一列后面，默认在最后一列，如果要放在第一则替换为first

**修改表之修改列 ：**

　　　　 Alter table 表名 change 旧列名 新列名 新列的类型 限制

          如果要修改原来列的属性可把新列名也写为旧列名即可

**修改表之删除列 :**

       Alter table 表名 drop 列名

**修改表之增加列主键 ：**

　　　 Alter table 表名 add  primary key (列名)

             如 ：　令id列为主键

          Alter table 表名 add  primary key (id)

    增加主键的时候如果出现duplicate entry错误， 就是因为主键有重复的值

 注意：

PRIMARY KEY

主键约束，每张数据表只能存在一个主键，主键保证记录的唯一性，主键自动为NOT NULL。

AUTO\_INCREMENT(自动增长)必须和主键一起使用，但是主键不是必须和AUTO\_INCREMENT一起使用

**修改表中某列的值 ：**

          Update 表名 set  name  =  ‘李四’ , age = 3  Where Num = 11

    注意 ： where 设置 修改的 条件

**删除表中某行的数据  ：**

      Delete from 表名 where  num = 1；

     注 :　　如果要删除多行，则需要用or

**查询语句**

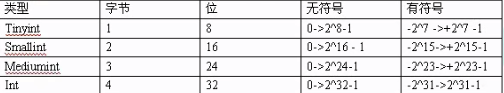
    查询全部数据   ：  Select \* from 表名 ；

    查询部分列的数据   ： Select num，name  from 表名 ；

    条件查询 ： select num,name from 表名 where num>1

★★★所谓的建表就是 声明列的过程

★★★插入的时候整型类型讨论



无符号和有符号分别表示不同的 能够取到的 取值范围 ，为了开发的速度合适的类型，默认是有符号存储范围也是在有符号的范围

★★属性进行初始化的赋值时 ，如果没有进行赋值则默认为 null  ，一般创建时可以用语句

          Name char(10)  not  null  default 自定义的默认值 ， null不好比较一般都是自定义

★★浮点型 float（M,D） M代表精度即是总长度 ； D代表标度即是小数点

如　：price float(6,2) à  9999.99 ~ -9999.99

★★字符型 (注意该类型的数据需要添加单引号)：

          **Char（M）  ：** 定长字符串 ， 如果存的数据小于M也实占M个字符，内容不足部分补0 ，M代表可容纳的字符数  ， 0<=M<=255  ， 如 ：nihao à 五个字符

                            好   à 一个字符（三个字节）

           **Varchar（M） ：**不定长字符串，如果存的数据小于Ｍ个字符，则占数据实际的大小

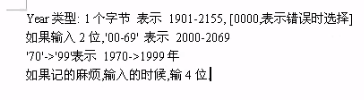
0<=M<=65535 (ascii)，但是它会有１到2个字节被用于记录长度



 速度上char比较快，varechar比较省资源

★★★**decimal ：**高精度浮点数，常用于金融运算，不会出现浮点数计算的误差

★★★year 类型



★★★Date 类型

8-1705298086.png



★★★实战建立106宿舍资料

create table ss106(

id int primary key auto\_increment,    // 主键，递增

name char(4) not null default '',

age tinyint unsigned not null default 0,

email varchar(30) not null default '',

tel char(11) not null default '',             //电话号码设置为char

salary decimal(7,2) not null default 0,      //金钱的问题设置为decimal类型

intro varchar(1000) not null default ''     //此处结尾不能添加逗号！！

)charset utf8;                          //设置为utf8类型

    Ps ：编写以上代码时 ， 单词有的打错了 ， 导致修改了几次才能运行！！！所以编写代码的时候一定要认真拼写，脑中一定要有印象

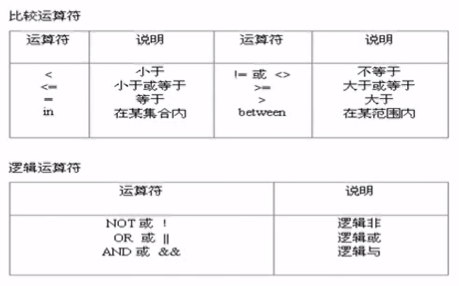
★★★使用delect的时候，有时候id不存在但也能够执行语句

★★★where :　 理解为使表达式是否成立，成立则为真

               如 ：取出数学比语文高20分的同学成绩

                Select \* from students Where math-china>20

**★★select中where常用运算符**



★  **in   ：** 表示在某个集合内进行取值,

     如 取出学号为2,3,5的学生成绩

   Select \* from students where id in(2,3,5);

★**between**： 表示在某个范围内进行取值 ，可以取边界值

     如 取出学号为2~5的学生成绩

      Select \* from students where id between 2 and 5 ；

       这句等于

      Select \* from students where id >= 2 and id <= 5 ；

★**not**：Select \* from students where id != 2 ；

Select \* from students where not in (2,3)；

★**like ： 模糊查询**

 （1）取出以某文字开头的 ：文字%

      如 ：取出姓张的成绩

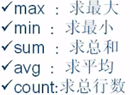
Select \* from students where name like ‘张%’；

 Ps :不姓张则 not like

 （2）取出限定字数的 ：　文字＿　，一个下划线一个字

  Select \* from students where name like ‘张\_’；

★**★★select 之 group ： 使用于统计，一般配合统计函数用**



★   **Max ：**

如 ：找出最高成绩

Select max(grade) from students ；

**Sum ：**

如 ：求出所有商品库存总和

Select sum(goods\_num) from goods ;

**Avg ：**

如 ：求出所有手机价格的平均价格

Select avg(price) from goods ;

**Conut ：**

如 ：求出行数

Select count(\*) from goods ;

**★group by ：**按照group by的字段进行分组查找和按此列排序，并在分组中寻找和要求的 ，经常和集函数一起使用 ， 如果不和集函数使用也是取出分组的排在第一位置的

如 ：查找出每个种类（type\_id）中最高价格的

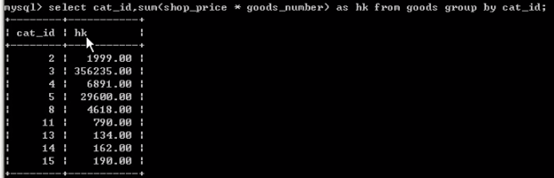
Select cat\_id , max(price) from goods group by type\_id;

Ps ：添加其他的选项没意义

★把列名当做变量名进行运算 (注：显示出来的栏目统计函数显示的时候会自己占列)



 Ps : 显示的名字可以用as 令命名



**★★ having 语句 ：**

Where 语句不能够 对查询的结果进行操作，需要进一步的操作必须受使用 having

如 ：查询市场的手机价格和店铺手机价格之间差价为100元的手机

(1) 错误的

Select good\_id,good\_name,market\_price-shop\_price as chajia from goods where chajia>100;

(2)正确的

Select good\_id,good\_name,market\_price-shop\_price as chajia from goods where market\_price-shop\_price >100;

但计算了两次

(3)用having语句 ：

Select good\_id,good\_name,market\_price-shop\_price as chajia from goods having chajia>100;

**★having和where 同时使用**

如 ：查询栏目3的，市场的手机价格和店铺手机价格之间差价为100元的手机

Select good\_id,good\_name,market\_price-shop\_price as chajia from goods where good\_id = 3 having chajia>100;

◇实战 ：计算挂科数目在２科以上的同学的平均成绩

Name  subject  score

Select name,sum(score<60) as shumu,avg(score) as pj from students group by name haing shumu>=2 ;

**★★★order by ：**改变排序的依据，一般默认升序 ， 如果要改为降序则要添加desc

      Order by 列名1[desc/asc],列名2[desc/asc]；

      如 ：按照价格进行降序排列，在价格排序好之后，如果价格一样的再按照id排

 select id,price from goods order by price,id desc

**★limit  ：** 配合order by 使用，限制选出来的条数和从哪条开始取

   limit 哪条后开始取，取出的条数

如 ：按照价格进行降序排列，在价格排序好之后，如果价格一样的再按照id排，且只取出4,5,6行的数据

 select id,price from goods order by price,id desc limit 3,3；

★★★ 特殊的错误 ：

★★如果出现这个情况，分号在前面，就会使任何的语句都不能执行，需要输入’;进行结束（即是一个分号和冒号）

5-1490903358.png

  ★★版本不支持中文

52-551002359.png

  ★★我擦，用一个insert语句，错了这么多个地方，第一个地方就版本不支持中文，第二个坑爹的地方就是 输入的 字符中含有 中文的字符，特别是标点的问题 ， 唉……

★★★where , group by , having , order by , limit 多个时，按照这个顺序进行抒写

★★复制一张表（只是复制表结构） ： create  table 表名 like 被复制的表名

★★复制某张表的数据 ：insert  into 表名 from被复制的表名(后能加order by等 )

★★清空一张表的数据 ：truncated 表名

**★★★巧妙的运用**

★★查询（select）的结果一般都是一个表结构，所以可以把结果当做表使用，用括号区别

  如 ：select \* from (select id,cat\_id,goods\_name,shop\_price from goods order by cat\_id) as temp group by shop\_price;

      括号内的查询结果当做一个已有的表进行处理，\*表示括号内有的列，把括号内的表称为temp

★★把查询结果当做条件使用 ：

   如 ：select goods\_id , goods\_name from goods where

goods\_id = (select max(goods\_id ) from goods) ;

          ★★where 带入行中看是否成立来理解，列看做变量来理解可以运算，查询的结果当做临时表来理解

         ★★★where 型子查询 ：把内层的查询结果作为外层查询的比较条件，用 in 来表示在里面

                 如 ：取出每个栏目中价格最贵的商品

　　　　　　　Select id,cat\_id,shop \_price from goods where shop\_price in (select max(shop\_price) from goods group by cat\_id);

                注意 ：这里巧妙地把一个查询的结果组作为　外层的查询条件；内层不用cat\_id也可以，因为那只是表示显示出来的栏目而已

★★★From 型子查询 ： 把内层的查询结果当成临时表，供外层Sql再次查询

注意 ：使用from型的子查询必须 添加别名 ， 即是as

否则出现这个错误

5-1756930039.png

           如 ：取出每个栏目下最新的商品

Select \* from (select id,cat\_id,goods\_name from goods order by cat\_id asc,id des) as temp group by cat\_id ;

              先排序好，再按栏目进行取出来

★★★不允许这样操作

0-1132385147.png

★★★找出挂科两门以上同学的平均成绩

  select name,avg(score) from students where name in (select name from(select name,count(\*) as num from students where score < 60 group by name having num>= 2) as temp) group by name;

      把题目进行分解  两个以上，不及格，平均成绩；所以先找出两科以上，不及格的学生

★★★**exists 子查询**：判断是否存在，返回值True或False，把外层的查询结果，拿到内层，看内层的查询是否成立

如 ： 获取 商品类型的表 中，有商品的栏目；

Select cat\_id,good\_name from category where exists (select \* from goods where goods.cat\_id=category.cat\_id);

英国是岛国之一，有海底隧道

**★★★Union ：** 连接查询 ， 把两次或者以上的查询结果一起显示，要求两次或多次查询的结果列数一样,类型可以不一样

                     如 ： select user\_name,msg\_content from feedback where id=1

                          Union select name, content from conmumicate where id=1

                           注意 ： 此时 列名以第一个为准

练习：有两个表A和B， 都有两列 分别为 id num，有部分的id 是相同的，要求把两个表显示出来，如果id相同则把 sum 相加

Select id,sum(num) from (select \* from A union select \* from B) as temp group by id;

                 注意 ： (1)如果合并的两个表中，它们都有一行所有的数据都相同的话，合并后的表会将此视为重复只保留一条（去重复功能），如果想要不去重复则要 加 all                                     如：select \* from A union all select \* from B

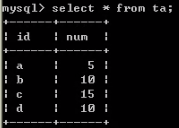
                        (2)如果union和order by ,limit 并用则要添加括号，为了分辨整体还是不部分

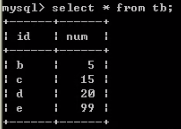
                                      (3)在两个表合并时，如果子句使用order by 必须陪着limit 使用呢，否则无效

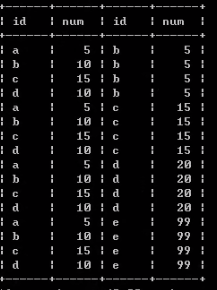
★★★Mysql中，集合相乘，就是笛卡尔积，就是两个集合的完全组合。 所以A\*B得到的积， 有M\*N个元素且不可能重复。

如 ：Ａ有９行，Ｂ有８行，两表相乘有７２行

　　　Select \* from A,B;   注意 ：用逗号进行分割表示乘，形成的过程是另一表从上到下进行匹配而成







★★左连接：以左表为准，去右表找匹配的数据，如果找不到就用null补

            所以至少有N行（即是左表的行数）

注意 ： 与列位置无关

     如 ：现在要从goods,category表中取数据，且将两张表互联

         Select

          Goods\_id,goods.cat\_id,goods\_name,shop\_price

          From

          Goods left join category

          On goods.cat\_id = category.cat\_id;

      注意 ：（1）在添加自己要的列名时，如果两个表中有相同的列名就要添加表名前缀

            （2）因为goods表在左边所以是left

            （3）第四，五部分表看做 连接 一张大表 ， 所以后面where,having就像以前写

            （4）右连接 right 替换 left ； 内连接 inner 替换 left



**★左右连接的区别 ：**

                       （1）以不同的表为准

                       （2）一般最好以左连接为好()，右连接可以转换为左连接

**★内连接** ：查询左右表都有的数据，即舍弃null的那部分，所以内连接是左右连接的交集

                         注意 ： 目前 mysql不支持外连接 outer join，但是可以用Union达到目的 ，就是把左右连接Union

练习 ：

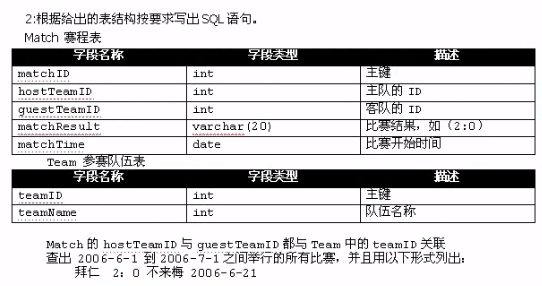


              Select goods\_id,goods.cat\_id,cat\_name,brand\_id,brand\_name,goods\_name

               from goods left join category  on goods.cat\_id = category.cat\_id  left join brand

               on goods.brand\_id = brand.brand\_id;

           解释 ：把四五部分看做一张表，再次进行左连接



       Select  t1.teamname,matchresult,t2.teamname,matchtime

       From

        Match left join team as t1

        On match.hostteamid = t1.teamid

        Left join team as t2

On match.guestteamid = t2.teamid

Where matchtime between ‘2006-6-1’and’2006--1’;

解析 ：为了区分team的不通所以用as添加了别名

★当多表时，同名如果不添加前缀，发生的错误

883-94487425.png

★★★视图 ：把查询的结果形成一张虚拟表，并没有真实的数据存在，只是与表的一种查询产生的关系 。

创建视图的语法 ：Create view 视图名 as select 语句

删除视图的语法 ：drop view 视图名

               如 ：create view goods2 as select id,goods\_name,goods\_price from goods where goods\_price >1500;

          优点 ：（1）简化视图，能把自己经常用的拿出来

                （2）能够起到约束的作用，可以选择想给别人看的列，把表的权限封锁，开放视图的权限

（3）大数据分表时会用到，当表的行数超过了200万行时，就会变慢。因此可以把一张表分为多张表。分表常用取模法：比如200万行分为4张表，就直接求余 à id%4+1 = [1,2,3,4]

(4)还可以把多张表合成一张表

 create view new select from n1  union select from n2

（5）视图的数据来自于表，如果表中的数据改变了则视图的数据也会改变；

     当视图的数据与表的数据一一对应时，此时可以进行修改。对于视图insert还应注意，视图必须包含表中没有默认值的列，否则视图不知道如何处理该列

      **★★视图的algorithm  （以下是它的取值）**

          （1）merge ：不写默认，当引用视图时，引用视图的语句与定义视图的语句合并。即是把查询视图的语句与创建视图的语句合并，分析之后形成select语句

                例如 ：新生成一个表goods2，进行查询表goods2小于3000元的手机

                       create view goods2 as select id,goods\_name,goods\_price from goods where goods\_price >1500;

                       select \* from goods2 where shop\_price ＜3000;

                   在执行第二句进行查询的时候，实际上执行的是where shop\_price >1500 and shop\_price ＜3000 ,第一句执行之后实际上是没有数据的只是空表

最终执行的是       select id,goods\_name,goods\_price from goods where goods\_price >1500 and goods\_price <3000’

                        查询的对象是goods表

　　　　　　　　　解析 ：在查询的时候，实际上还是回到了goods表中进行查询

          （2）temptable ：当引用视图时，根据视图的创建语句建立一个临时表

               例如 ：  create algorithm=temptable view goods2 as select id,goods\_name,goods\_price from goods where goods\_price >1500;

                           select \* from goods2 where shop\_price ＜3000;

          最终执行的是2句话 ： 先取出数据放到临时表然后去查临时表

                                 查询的对象是临时表

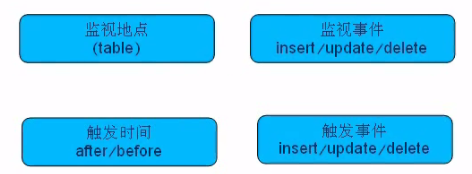
　　　　　（3）Undefind　：让系统帮你选

**★★★字符集 ：**如果某级没有设置字符集，则继承上一级的。一般设置表的即可 。一般出现乱码的问题是因为字符集不符合造成的，可以声明字符的编码进行解决

**★★★触发器 ：**监控某种情况，并触发某种操作

               如 ：商店拍下商品并减少库存　；

　　　　★触发器能够监视和触发的操作均为增删改 , 四要素



         触发的时间 ：after/before ：比如饭前洗手（before）/饭后洗手（after）

        ★★创建触发器的语法

           Create trigger 触发器名字

            After/before  insert/update/delete  on  表名

             For each row

             Begin

               Sql语句

              End$

            在表XX触发XX操作之后/之前，每一行开始触发XX操作，然后结束

注意 ：（1）sql即是需要触发的语句，可以是多句，但是也必须是那三个语句范围

（2）多个sql语句用分号隔开，并事先声明用$进行结束，语句为 delimiter $，此时分号只表隔开不是结束 ， 修改回来则是  delimiter ；

  ★续上，如何在触发器的sql语句中引用行的值 ：

（1）对于监视insert来说，新增的行 可以用new来表示，且行中增加的一列：

用  new.列名 来表示

实例 ：两张表，一张是商品表goods；另一张是订单表orde，如下

 注意 ：这里不能命名为order ，因为是关键字

当进行下单后，库存减少的触发器

进行设置触发器

当插入语句后，触发的情况。这里是的good的gid是和orde的id对应的，

（2）对于监视delete 来说 ， 删去之后，则之前的是旧行 ， 可以用old来进行表示

     且行中被删除的一列的值：用  old.列名 来表示

当删除下单后，库存增加的触发器

删除后的触发效果

（3）对于监视update来说， 更新之后 ， 改之前的是old，改之后是new

        如 ：同一样商品，下单之后，又修改了订单的数量　，则

库存数＝当前库存（未改的）＋未改订单商品数量—修改订单后商品数量

注意　：如果不是同一样商品的话则拆开计算那个语句

★删除触发器的语法 ：drop trgger 触发器的名字；

★查看触发器 ：show trigger

★after和before的区别

before案例 ：用于对订单的判断，如订单的数量超出范围，强制修改订单等等

　　　如 ：如果订单超过5，就强制把订单数改为5

★★★存储引擎 ：数据库存储同样的数据，有着不同的存储方式和管理方式，称为存储引擎　，　常用的有

Myisam ：批量插入速度快，不支持事务，锁表，支持全文索引

innoDB ：批量插入相对较慢，支持事务，锁行，支持全文索引

　　★★事务：指一组操作，要么全部都成功操作，要么都不执行à原子性（不可分割）

在所有的操作完成之前，其他会话不够看见中间的改变à隔离性

当事务完成之后，其影响是会保留下来的，不能撤销à 持久性

数据的变化，在事务的前后保持不变 à一致性

比如 ：转钱瞬间银行断电收不到钱，因为只完成了转钱扣钱的操作，转钱进的操作未完成

　　　★事务一般选用InnoDB，所以建表的时候要进行声明

     ★事务的语法 ：

（1）开启事务：start  transaction

（2）提交事务：commit  （在输入完要进行的事务语句之后，进行提交）

在没提交事务之前，对方都不会看见你的操作结果

有些语句会造成事务的隐式提交比如再来一个start transaction

（3）rollback ：取消事务

事务的原理

★★★备份 ：系统运行时，增量备份与整体备份

        比如 ： 每周日整体备份一次，周一到周六备份当天

 ★★备份文件语法：

  （1）导出库下的多张表

mysqldump –u用户名 –p密码 库名 表1 表2 > 地址/自定义备份文件名

注意 ：如果把表名去掉就能够导出一个库下的所有表

mysqldump –u用户名 –p密码 库名 > 地址/自定义备份文件

（2）以库为单位进行导出

mysqldump –u用户名 –p密码 –B 库1 库2> 地址/自定义备份文件

（3）导出所有的库

mysqldump –u用户名 –p密码 –A> 地址/自定义备份文件

★★★数据库的恢复 ：即是把备份文件里面的语句全部执行一次

　　★★登陆到命令行的状态下

（1）恢复一个库

　　　Source 地址//库名

（2）恢复一个表

   Use 库名

   　  Source 地址//表名

★★不登陆到命令行

注意 ：如果直接使用cmd的话，显示‘mysql'不是内部或外部命令,也不是可运行的程序时，需要设置环境变量。我的电脑上右键-〉属性-〉高级-〉环境变量-〉新建：变量名：path 变量值：mysql.exe所在的目录。然后重启CMD窗口

★★★索引 ：针对数据所建立的目录，加快搜索的速度

 例子 ：　select \* from goods where id=100; à 实际上在查找的时候并不一定是从id=1查到id=100，它是会按照一定的优化算法寻找的

 ★★索引的算法 ：

　　　（1）二叉树索引，把数按照大小不断分开在两边 log2N

      （2）哈希索引

　★索引的优点 ：加快查询速度

    索引的缺点 ：降低了增删改的速度，增大表文件的大小（索引文件可能大过表文件）

　　★实例　：如果要导入很多的数据，索引占内容比较大的话，就要先去掉所引，再导入，然后再统一添加索引

    所以，索引的时候必须：（1）不过度索引 （2）索引条件列选择比较频繁的比如id而不是性别 （3）索引散列的值，比如不索引为男，女添加索引

★★文件第一个是表结构文件，第二个是表数据文件（最后的Date），第三个是表索引文件（最后的index）。所以索引文件比数据文件大。

 ★★索引的分类 ：

    （1）普通索引 ： index 仅仅是加快查询的速度

    （2）唯一索引 ：unique index 行上的值不能重复

    （3）主键索引 ：primary key 不能重复，一张表只能有一个主键，但是可以有一个或者多个唯一索引

    （4）全文索引 ：fulltext index

★查看一张表上的所有索引 ： show index from 表名

            垂直排列查看    Show index from 表名 /G

★给表中的列添加索引 ： alter table 表名 add index/unique /fulltext自定索引名（列名）

　　　　　　　　　　　alter table 表名 add primary (列名)

  注 ：自定索引名可以省略则默认和列名一样，主键索引直接省略它

            注 ：全文索引是依靠相似度来匹配找出

Select \* from 表名 where match（全文搜索的列名） agains （‘搜索的内容’）

全文索引是有停止词的，就是对于常见的词比如in it，是不会添加索引的

全文索引在mysql的默认情况下，对中文的意义是不大的，因为英文单词有空格拆分成单词，进而对单词进行索引，中文则没有

★删除非主键索引 ：alter table 表名drop index 索引名

删除主键索引 ：alter table 表名 drop primar key

★★★存储过程 ：　procedure ，将一段代码封装起来，当要调用该存储过程来实现。在封装的语句体里面，可以用if，while等控制

  ★★查看现有的封装　：show procedure status

      删除封装 ：drop procedure 名字

       调用封装 ： call 名字（）；  实际调用封装的语句，即是begin到end之间的语句