MySQL

数据库分类

层次数据库

网状数据库

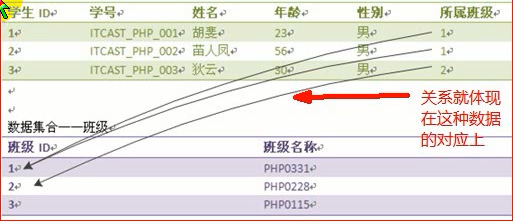
关系数据库

主流数据库

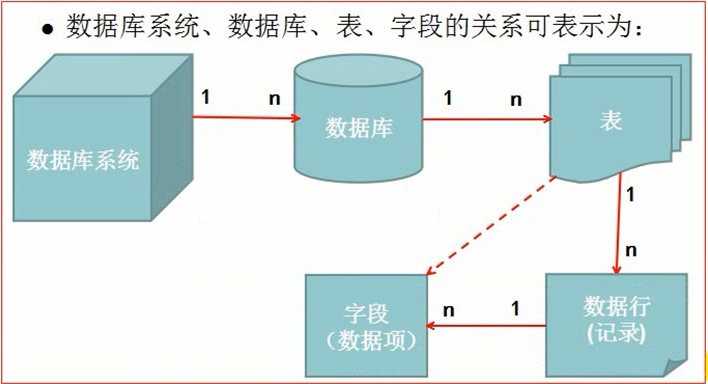


Sql：这是关系数据库的“语言标准”，STRUCT QUERY LANGUAGE（结构化查询语言）

关系数据库特点：表跟表之间可以建立



MySQL数据管理层次



数据data

数据库database：用于存储一个项目所使用的各种数据的一个逻辑单位

数据库管理系统 dbms：其实就是指一个具体数据库软件产品：database，management，system

表 table：用于存储一种数据的“结构形式”，基本就是“行列对齐”的“外观样子”。

字段 field 列column一个数据表中的一个纵列，其有一个名字，又叫“字段”

行row 记录record

数据存储的表现：

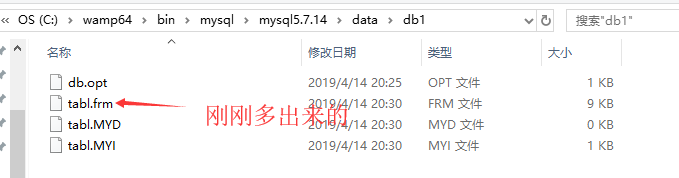
我们通过代码（语句），可以操作各种数据进行各种操作，但其其实，在文件表现，其实只是几个文件名而已，具体在数据库的存储目录中：

1. 每个数据库，会对应一个文件夹
2. 每个数据表，会对应一个或几个文件

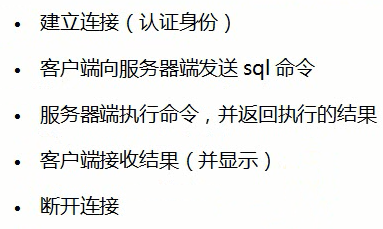


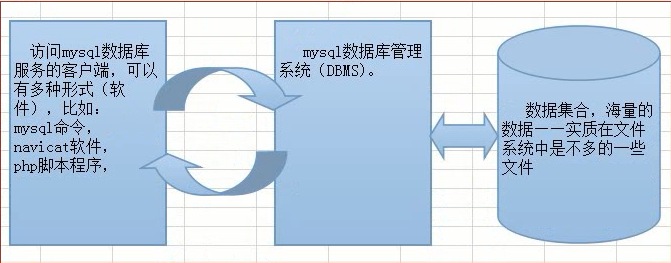
则对应数据库的存放目录中，就会多出一个文件夹

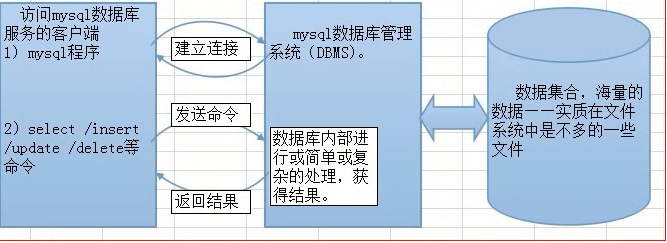




关系数据库的基本结构和流程







**Mysql数据库的系统级操作及基本语法规定**

**启动/停止**

**Cmd（管理员身份）**

* **启动 net start mysql**
* **停止 net stop mysql**

**也可在管理工具中服务中操作**

**Mysql登录退出**

登录1 mysql -h服务器地址 -u登录名 -p

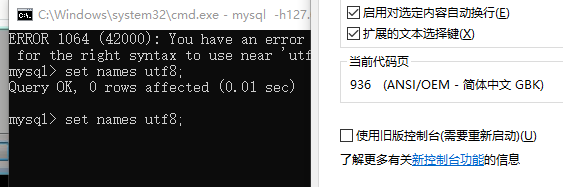
登录2 mysql –-host=服务器地址 –user用户名 –port=端口 –password

MySQL默认3306

**前两个语法，可以使用的前提都是，设置了变量环境**

**退出 quit 或 exit –是指已经登录（进入）了mysql后**

* 注意：登录数据库系统后，需要使用“**set names 编码名；**”来设定当前**连接数据库的“环境编码名”**，即当前跟数据库打交道的“客户端”本身的编码。通常来说：
  + cmd客户端中是固定的gbk编码，
  + 而php网页中，是该网页文件的编码（现在主流都是utf8）。

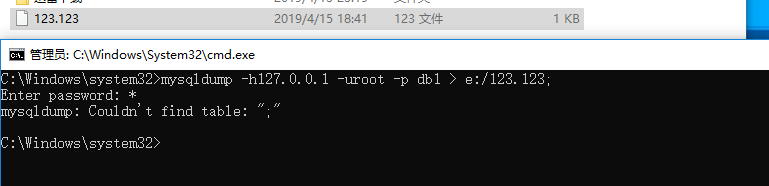


在cmd窗口中，set name只有这一个值gbk可用，而且，强烈建议：每次登录就执行该命令

**数据库备份和恢复**

备份：就是将一个数据库，完整“转换成”一个可以随时“携带和传送”的文件；

备份语法：mysqldump -h服务器地址 -u登录名 -p 数据库名>文件名



恢复：就是将一个备份的数据库文件，完整还原为一个可以使用的数据库

语法：mysql -h服务器地址 -u登录名 -p 数据库名<文件名

注意：

1. 这两个命令，都是在“没有登录进入”的时候使用。
2. 其中mysqldump命令还要求为管理员身份
3. 通常，恢复，就是指恢复原来数据库中的所有表数据信息及其他信息，而数据库名可以是原来的名字或新的名字。

# 基础语法规定

## 注释：

有如下：3种

1. 当行注释 ： #注释内容
2. 当行注释： -- 注释内容（--后面有一个空格）
3. 多行注释： /\* 注释内容 \*/

## 语句行：

默认情况下，以分号作为一条语法结束；

而且，常规的操作中，都是“一次执行一条语句”；

但：

Mysql中，也可以人为设定语句结束符 做法如下

语句： delimiter 新的结束符

此行之后，就可以使用新的结束符；

### 大小写问题

1. mysql语言内部本身不区分大小写
2. 但，mysql的某些命令执行会生成文件夹，此时他们就可能会区分大小写

2.1：在区分文件名称区分大小写的系统中，这些名字也会区分大小写，比如Unix，linux

2.2：在区分文件名称不区分大小写的系统中，他们同样不区分大小写，比如Windows

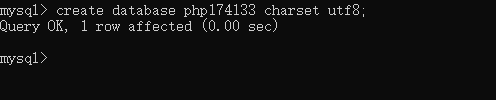
### 命名规则问题

* 可以自己命名的名字，称为标识符，包括：数据库名， 表名，字段名，视图名，函数名，过程名，变量名，用户名，，等等。
* 可以命名标识符的字符比常规的语言多，但特别建议只用：字母数字和下划线，并不用数字开头。
* 非常规字符或系统关键字虽然可以作为标识符使用，但最好要包在反引号（数字1左边那个反撇 ` ）中，并且不推荐。
* 对数据库名，表名，和视图名，在window系统中不区分大小写，而其他系统中区分，建议全使用小写，并采用下划线分割法。
* 对其他自己命名的标识符（字段名，函数名，过程名），不区分大小写，但也建议全使用小写，并采用下划线分割法

### 数据库定义语句

**创建数据库**

形式 create database 数据库名 charset 字符编码 collate 排序规定



说明：

1，字符编码名称是用于设定当前数据库中存储的字符内容以什么编码来存储。

2，collate排序规则用于设定其中的字符内容的“大小关系”（先后顺序）：

对于英文，基本没有任何问题，比如：

“abc” “abd”： 后者大；

“d”; “abc”： 前者大；

.....

所有的对于英文字符的比较，本质上都是“单个字符”的比较。

但，对于中文，就成问题了，比如：

“中国”， “印度”： 谁大？

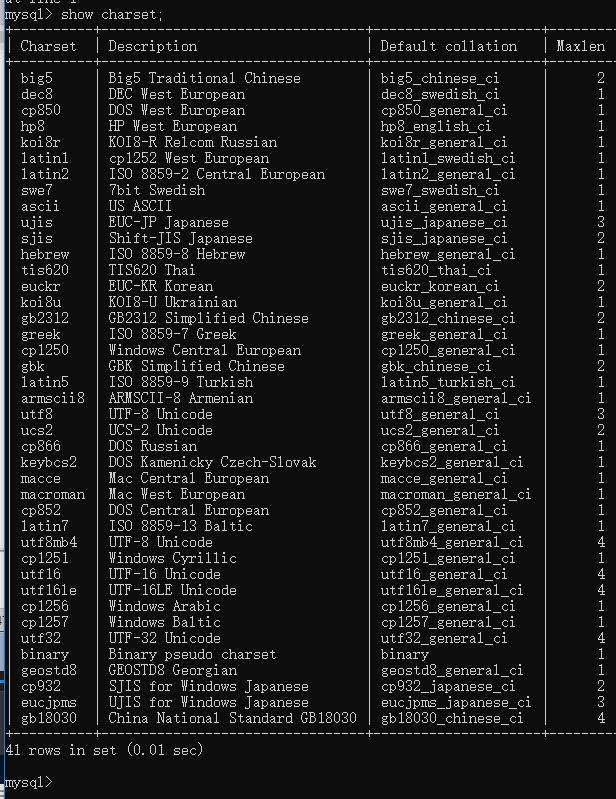
“中”， “美”： 谁大？

排序规则，就是用于设定类似这种字符的大小关系或先后顺序的一种规定！

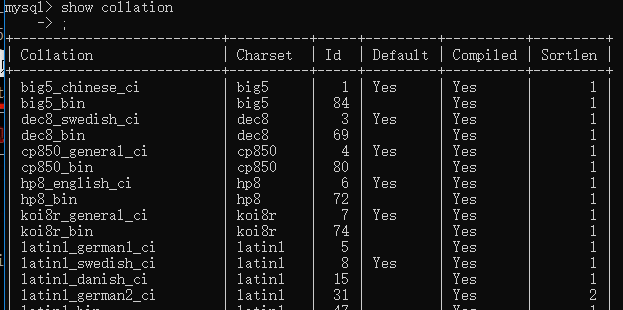
实际我们的代码中（应用级别），只是一个名字：排序规则名

而且，通常每种字符编码（字符集），都有一个默认的排序规则，所以，通常都不写的。

显示mysql中的所有可用的字符编码（供共41种）：



显示MySQL中所有可用排序规则：222种

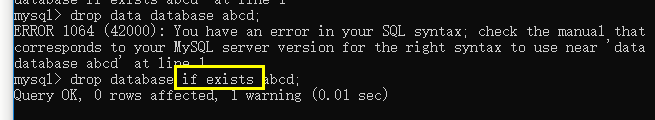


实际应用中，我们使用某个字符集（字符编码），然后可用的与之对应的排序规则其实可选项很少，通常只有两个，当然，我们

**删除数据库**

形式：

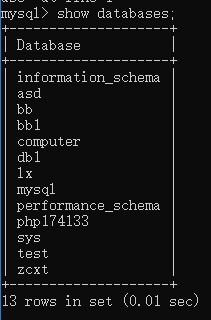
Drop database 数据库名；



修改数据库（字符编码）

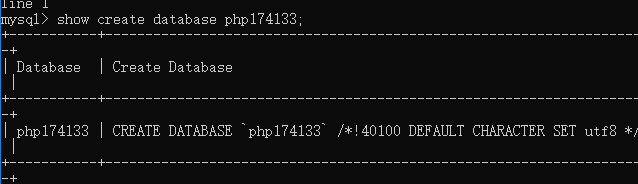
**显示所有数据库**

Show databases；注意 后面要加s



显示一个数据库的创建语句

Show crease database 数据库名



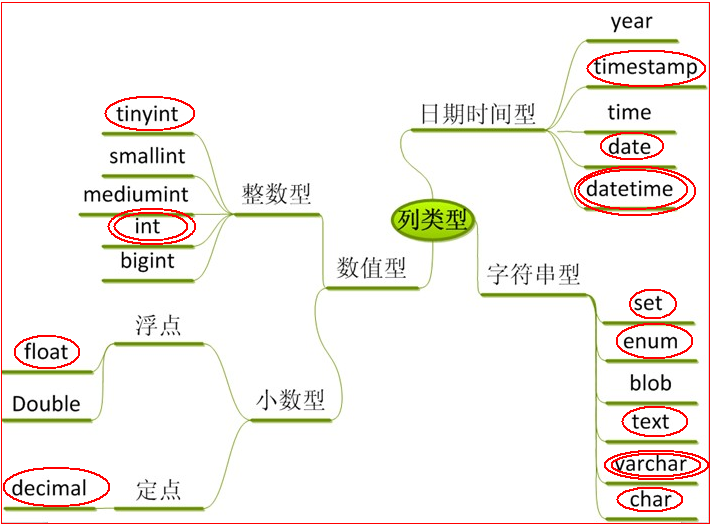
**进入（选择）某个数据库**

Use 数据库名

# 字段类型（数据类型）

### 总览：

Mysql中，数据类型主要分3大类，数字型，字符型，时间型；



数据类型，在mysql中，主要用于创建一个表的时候，设定字段的类型

Create table 表名（id int f1 float,f2 double f3 time f4 text）

**整数类型**

主要有：int tinyint smallint mediumint bigint

**整数类型所占空间（字节）**

Int 占4个字节 即32位

Tinyint 占1个字节，即8位（最多存储256个数字，默认范围是-128到127）

Bigint 占8个字节，即64位



整数类型字段的设定形式：

类型名【(M)】【unsigned】 【zerofill】

说明：

1，M表示设定该整数的“显示长度”，即select输出的时候，123可能显示为000123（假设M为6)

2，unsigned用于设定该整数为“无符号数”，其实就是没有负数。

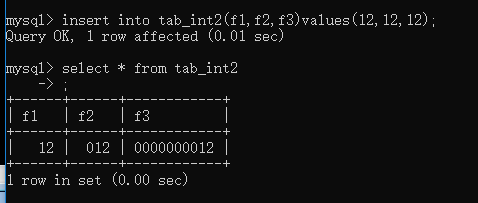
3，zerofill用于设定是否填充“0”到一个数字的左边，此时，需与设定的长度M配合使用。

4，如果设置了zerofill，则自动也就表示同时具备了unsigned修饰

举例：







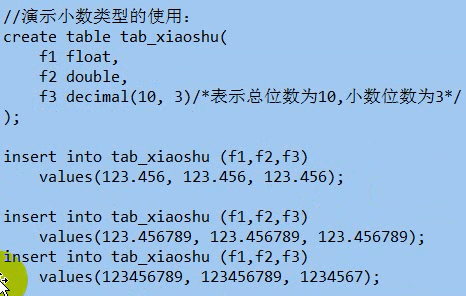
## 小数类型

有3中：float， double，decimal。

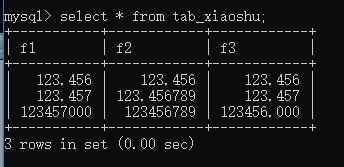
float：单精度浮点型，使用4个字节存储数据，其精度大约只有6-7个有效数字数位；

double：双精度浮点型，使用8个字节存储数据，其精度大约只有15个有效数字数位；

decimal：定点小数类型，整数部分最长可以有65位，小数部分最长可以有30位。一般设置格式为：decimal(总位数，小数部分位数）



输出



**其中，f1字段的值，精度损失明显**

**P**hp中，操作MySQL数据库基本函数复习：

通常，php网页中完成有关数据库的操作，首先，需要如下代码

$link=Mysql\_connect（“数据库地址”，“用户名”，“密码”）//连接数据库系统

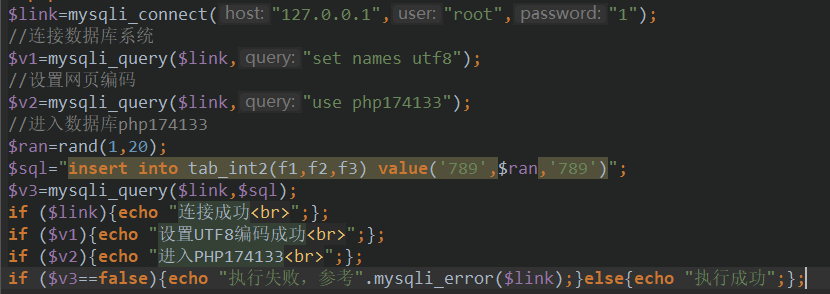
Mysql\_query(“set names 网页文件编码名”)//指定连接编码

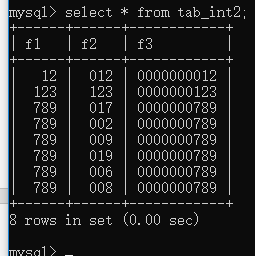
Mysql\_set\_charset（“网页文件编码名”）

Mysql\_query(“use 数据库名”) //选定要使用的数据库

然后，才开始正式执行要完成数据库操作任务；

$result = mysql\_query(“select/” delete /update/insert…….





其他个别补充：

Mysql1\_error（）

获取mysql执行失败时错误信息，通常应用于如下场合：

$result = mysql\_query(“select/delete/update/insert/create/dropt…所有SQL语句)

If（$result==false）{echo “语句执行失败.mysql\_error”}

Else{//成功}

对于mysql\_query这个函数，可以执行“几乎任何sql语句”，在应用中，通常需要区分两种情况：

情况一：

执行没有数据返回的语句，比如 insert update delete create table create database drop….

这种语句，使用mysql\_query 执行结果只有true和false

True表示执行成功 false表示执行失败

情况二：

执行有数据返回的语句：select show tables show database desc 表名

此时，使用mysql\_query 返回的结果为：

如果失败，返回false；

如果成功，返回的是“结果集”—“数据集”---就是一行一行数据，就是有字段的一个“表结构”。他需要我们进一步去“取出”数据，才能使用其中的数据；

$result=mysql\_query（“”）

While($rec = mysql\_fetch\_array($result)){

// mysql\_fetch\_array($result)会取出该结果集中的“一行”数据，并返回数据后赋值给rec中

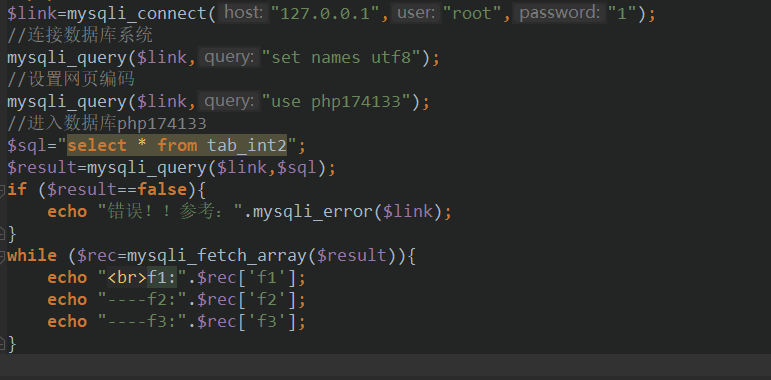
//rec就是一个数组，其下标就是字段名

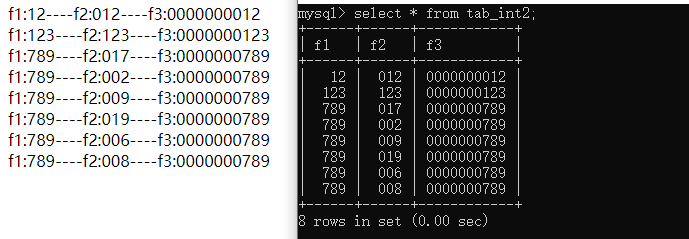
//在此while循环中，mysql\_fetch\_array会一次次取出结果集的所有数据

//处理rec

}

示例如下：





# “fetch函数”的3种形式的辨析：

假设mysql\_query(“select id, age, name ”)执行的结果集为$result，其该表中的数据类似这样：

|  |  |  |
| --- | --- | --- |
| id | name | age |
| 1 | user1 | 18 |
| 2 | 张三 | 28 |
| 4 | user4 | 38 |

$result = mysql\_query(“select id, age, name ”);

mysql\_fetch\_assoc($result):得到的数组类似这样：

array(“id” => 1, “name” => “user1”, “age”=>18);

mysql\_fetch\_row($result):得到的数组类似这样：

array(0=> 1, 1=> “user1”, 2=>18);

mysql\_fetch\_array($result):得到的数组类似这样：

array(“id” => 1, “name” => “user1”, “age”=>18, 0=> 1, 1=> “user1”, 2=>18);

## 扩展php中操作mysql数据的几个函数：

$n1 = mysql\_num\_rows(结果集); //获得该结果集的数据行数；

$n2 = mysql\_num\_fields(结果集); //获得该结果集的数据列数；

$name = mysql\_field\_name(结果集, $i ); //获得结果集的第i个字段的名字！i从0开始算起