# 2017年2月16日

# 基本语句

注:[]内容为选填

//连接数据

mysql -h localhos -u root -p [password];

//显示当前所有数据库

show databases;

//创建数据库 cmsdemo是自定义数据库名称

create database cmsdemo;

//使用cmsdemo数据库

use cmsdemo;

//查看所有表

SHOW TABLES;

//查看表内容

select \* from 表名称

//查看表结构

DESC users;

//查看数据库状态

\s

//退出编辑

\c

# 一、创建数据表的SQL语句模型

规则

[]为可选填 WIN不区分大小写 LINUX区分大小写

CREATE TABLE [IF NOT EXISTS]表名称(

字段名1 列类型 [属性][索引]

字段名2 列类型 [属性][索引]

)[表类型][表字符集utf-8等];

# 二、列类型；

①整形 ②字符串型 ③日期型

-> 细分都是按空间大小来区分的

## 整形INT

### TINYINT非常小的整形

1字节 1字节=8位 0 0000000 有符号-128-127 无符号0-255

### SMALLINT较小的整形

2字节 2字节=16位 0 0000000 000000

### MEDIUMINT中等大小的整形

3字节

### INT标准的整数型

4字节

### BIGINT大整数型

8字节

CREATE TABLE tab1（id tinyint）; //创建tab1 字段为id 类型为TINYINT

insert into tab1(id) values(100-->设置id数值100);

select \* from tab1; //选择tab1表

CREATE TABLE tab2 (id tinyint UNSIGNED-->定义为无符号)；

insert into tab2(id) values(90);

insert into tab2(id) values(900);

insert into tab2(id) values(-90);

## 浮点型

### float(M,D)

4字节 M个数保留小数D位 四舍五入

double(M,D) 8字节

定点数

### Decimal 定点数

M+2字节 以字符串保存

CREATE TABLE tab3(price FLOAT(5,2)); //price为字段名 FLOAT(5,2)表示一共五位两位是小数

insert into tab3(price) values(12345678.123); ->999.99

insert into tab3(price) values(123.33); ->123.12

浮点数是近似数,有误差，对于敏感的如货币数值用 定点数保存；

定点数精确，不能直接运算，影响效率

CREATE TABLE tab4(one FLOAT(8,1),two decimal(8,1));

insert into tab4(one,two) values(12.3,12.3);

insert into tab4(one,two) values(12.38,12.38); -》12.4 12.4

## 字符型

### char(m)固定长度

255字符->最大存储空间255字符 m指定多长字符

固定长度 char(4) 占4个字符 少了用4字符 多了截断

存储4个字符

**m定义的是字符**

### varchar(m)可变长度

mysql单行最大长度65535字节

可变长度 var(4) 4字符

①ASCII 英文一字节，汉字两字节。

②UTF-8，英文字符一个字节，中文（含繁体）等于三个字节。

③Unicode英文两个字节，中文（含繁体）等于两个字节。

文本符号：英文标点占一个字节，中文标点占两个字节。

④UTF-16英文或汉字2个字节（Unicode扩展区的一些汉字存储需要4个字节）。

⑤UTF-32任何字符都需要4个字节。

### Text

文本数据，文章、日记 长度2^16-1

MEDIUMTEXT 长度2^24-1

LONGTEXT 长度2^32-1

### Blob

保存二进制数据，照片、文本、压缩包 长度2^16-1

MEDIUMBLOB 长度2^24-1

LONGBLOB 长度2^32-1

一般很少保存二进制数据的，一般是保存二进制数据的文件名

SQL语句无法存图片要用PHP、JAVA等程序进行读取二进制数据获取字符串然后用语句插入字符串

### ENUM枚举

枚举类型 1或2字符 通过下标取值

ENUM("one","two","three","four") ---MAX 65535成员 一次只能存一个值

1 2 3 4

例如：

create table tab9(one enum("a","b","c","d"),two set("a","b","c","d"));

insert into tab9 values('a','b') -one-》a two-》b

insert into tab9 values('a,b','a,b') -one-》a two-》ab

### SET 集合

集合类型 1，2，3，4，8字节 ---MAX 64成员 一次能存多个值，用逗号隔开

-------------------------varchar char----------------------------------

CREATE TABLE tab6(name char(10),desn varchar(10));

insert into tab6 values('abc','abc');

select \* from tab6;

### VARCHAR与CHAR检索方式不同

CREATE TABLE tab7 (v varchar(4),c char(4));

select concat(v,"#">,concat(c,"#") from tab7; //连接字符#

v-》ab # c-》ab#

char 因为固定长度所以速度比varchar快 变化不大的用char 如：性别

## 日期型

### DATE YYYY-MM-DD

### TIME hh:mm:ss

### DATETIME YYYY-MM:SS hh:mm:ss

### TIMESTAMP YYYYMMDDhh:mm:ss 时间戳

### YEAR YYYY

创建表时最好不要使用这些中的时间格式(PHP中时间戳 1970-1-1 0：0：0)

用整数保存时间 time(); 转整数传到数据库 INT保存

## NULL

示例：

CREATE TABLE IF NOT EXISTS users(

id INT,

\c-》退出编辑 没有创建成功

有'号的情况下要退不出要闭合'方可退出

判断users 表是否存在才创建

CREATE TABLE IF NOT EXISTS users(

id INT,

name CHAR(40) ->最后一个不用加逗号， 40为字符

);

# 三、数据字段属性

### 1.unsigned去除负数

去除负数，整数空间增加一倍 -128-127 0-255

用在数值型

字段名1 列类型 [属性][索引]

CREATE TABLE t1 (id int unsigned);

insert into t1 values(100);

### 2.zerofill前导0

只能用在数值型的整形无符号

自动应用无符号unsigend

字段名1 列类型 [属性][索引]

CREATE TABLE t2 (num int(5) zerofill,price float(7,2) zerofill,name varchar(10));

int 限制不住长度 float varchar限制长度

00001

### 3.AUTO\_INCREMENT 数据每增加一条就会自动增1

只能是整数，设置主键的字段值是不允许重复

NULL 0 留空 亦+1

每个表都最好有一个ID字段，设置为自动增长 auto\_increment

CREATE TABLE t3 (id int auto\_increment primary key,name char(10));

|

索引

delete from t3 where id>1 and id <10 //删除

//delete frome t3 //全部删除

select\* from t3 order by id; //排序

### 4.NULL 和 NOT NULL

默认是空

NULL值

建议：在创建表时每个字段都不要插入NULL NULL转为PHP不一定是自己想要的 如NULL -> 0 空字符串-》""

CREATE TABLE t3 (id int not null,name char(10) not null);

### 5.default 缺省值

CREATE TABLE t3 (id int not null default 0,name char(10) not null default '',price double not null default "0.00");

default 留空时自动转换为相对应的字符

用户创建示例：

CREATE TABLE users(

id INT UNSIGNED NOT NULL AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(30) NOT NULL DEFAULT '',

height DOUBLE(10,2) NOT NULL DEFAULT 0.00,

age INT NOT NULL DEFAULT 0,

sex CHAR(4) NOT NULL DEFAULT '男'

);

# 四、创建索引

### 1.主键索引 primary key ->防止创建重复的值

一表一主键、主键的值不能为空、内容不能重复

建议每个表都设置主键

写法一：

CREATE TABLE t7(

id int not null auto\_increment primary key,

name varchar(10),

age int not null default ''

);

写法二：

CREATE TABLE t7(

id int not null auto\_increment,

name varchar(10),

age int not null default ''，

primary key(id)

);

desc t7 //查看表 t7

### 2.唯一索引 unique->防止创建重复的值

一表可多唯一索引、内容不能重复

示例：

CREATE TABLE t7(

id int not null auto\_increment,

name varchar(10),

age int not null default '' unique， //防止重复

primary key(id)

);

### 3.常规索引 index key 同义

可多列

### 写法一：独立

CREAT index index\_name on t7(name,age); //创建索引

drop index index\_name on users; //删除索引

### 写法二：

CREATE TABLE t7(

id int not null auto\_increment,

name varchar(10),

age int not null default '' unique， //防止重复

uid int not null

sid int not null

primary key(id),

key index\_name(uid), 写法1

index index\_name(sid), 写法2

index index\_name(uid,sid) 写法3

);

drop index index\_name on t7 //写法3方便管理

### 4.全文索引 fulltext

MyISAM表类型使用、在VARCHAI CHAR TEXT文本字符串是使用

可创建多个字段索引

示例:

CREATE TABLE books(

id int,

bookname varchar(30),

price double,

detail text not null,

fulltext(detail,bookname), //detail ,bookname均为字符串所以可以用全文索引

index text\_name(price) //text\_name为索引名称price为数值不能用全文索引

primay key(id)

匹配语句 select\* from books where bookname like '%php%'; 匹配php 速度慢

全文索引 select bookname,price frome books where MATCH(detail) AGAINST('php');

select match(detail) against('php') from books;

# 五、数据表类型及存储位置

show engines; //查看支持的数据表引擎

show variables like 'table\_type'; //查看默认的表类型

我们只学12个中的MYISAM 和 InnoDB 两个表类型

create table () type InnoDB; 写法一

create table () engine InnoDB; 写法二

MyISAM 表类型是默认的表类型 成熟稳定易于管理表格锁定机制用来优化多个并发的读写操作

### 1.MyISAM 表类型

OPTIMIZE TABLE 表名； 修复浪费的空间

强调快速读取操作

### 2.InnoDB 表类型

占用空间大一倍，读取不如MyISAM，不支持全文索引

支持一些MyISAM不支持的功能

示例：

\* 功能 MyISAM InnoDB

\*

\* 事务处理 不支持 支持

\* 数据行锁定 不支持 支持

\* 外键约束 不支持 支持

\* 表空间占用 相对小 相对大 最大2倍

\* 全文索引 支持 不支持

演示：

CREATE DATABASES xsphpdemo;

use xsphpdemo;

create table t1(id int) engine = InnoDB;

create table t2(id int) type = MyISAM;

create table t3(id int);

MyISAM类型有 结构FRM 数据MYD 索引MYI 3个文件

# 六、MySQL默认字符集

\s 查看库状态

ASCII 7位字节

ISO-8859-1/latin1 西欧

gb2312-1980 6000+字 不推荐使用

gb13000-1993 2.7W+字 不推荐使用

GBK -1995 gb2312 扩展且集gb13000.1 可以使用

GB18030 -2000 2字节 数据库支持还比较少见

UTF -32 4个字节字符集

USC -2 2个字节字符集 win2000 内部使用

UTF -16 2或4字节字符集

UTF-8 1到4字节编码 强烈推荐

\* 汉字 GBK占 2字节 UTF8占 3字节 数据库使用时uft8 不要带杠

\*

\* name varchar(12) 6个汉字 GBK

\* name varchar(12) 4个汉字 UTF-8

MySQL服务器， 数据库， 数据表， 字段都可以指定字符集

show character set; //查看支持的所有字符集

\* MySQL的字符集包括两个概念

\*

\*

\* 字符集： 是用来定义MySQL存储字符串的方式 36

\*

\* 校对规则：是对规则是定义了比较字符串的方式 70

\*

\* 一对多的关系： 1 个字符集可以对应多个校对规则

Show character set;

show collation like 'gbk%'; //查看gbk 校对规则

\_ci不区分大小写

\_cs 区分大小写

\_bin ……二进制比较

show variables like 'collation\_server' //查看服务器变量 collation\_server 为服务器名称 character\_set\_server

CREATE database xsdemo default character set gbk collate gbk\_chinese\_ci;

use xsdemol;

\s

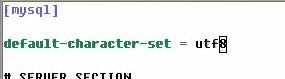
default character set gbk 设定字符集 缺省时用gbk

collate gbk\_chinese\_ci 设定校对规则

创建表时也可以指定字符集

CREATE table t1(id int) type=InnoDB default character set gbk collate gbk\_chinese\_ci;

服务器字符集修改my.ini



客户端与服务器交互时字符集->意思是读取出来放到客户端

character\_set\_client //发出内容时

character\_set\_connection //连接时

character\_set\_results //获取结果

三个字符集保持一致就不会出错了

set names 字符集 同时修改以上三个值

set names utf8;

\s

alter database character set utf8; //更改数据库字符集

\s

alter table t1 character set utf8; //更改表字符集

\s

控制台：

mysqldump -u root -p --default-character-set=utf8 -d users > c:/windows/system32/users.sql //备份数据库 users为数据名称

mysql -u root -p users <c:/windows/system32/users.sql //导入库

# 八、修改表

Alter 更改 -》 add 增加 modify 改类型 change 改类型和字段名 rename 改表名

? alter table //修改表更多语法

model；Alter table 表名称

alter database 数据库名称

Alter database character set utf8; //更改数据库字符集

alter table t1 add name varchar(30) not null default '';

增加 表字段 类型 属性

alter table t1 add sex varchar(30) not null default '' after name ; //字段name后添加字段

表名 增加 表字段 类型 属性

alter table t1 add height double first; //首行添加字段

alter table t1 modify name char(40); //modify修改name字段类型

alter table t1 change name username char(40); //change修改name字段名称 name字段名改为username

alter table t1 rename as users; //修改t1表名为users

alter table users drop sex; //更改USERS表并删除sex字段

drop table users; //删除表

drop table if exists users; //如果表USERS表存在则删除

show tables; //显示当前库所有表；