数据库的基本操作

1. 显示已有的数据库

show databases

1. 创建数据库

create database db\_name

1. 删除数据库

drop database db\_name

1. 表三种类型
2. 3.22及以前版本中缺省为ISAM表
3. 3.23版本中缺省为MyISAM表
4. 指定表的类型经常用于创建一个HEAP表

CREATE TABLE fast(id int,articles TEXT) TYPE=HEAP;

1. 1) 创建表（创建表必须同时创建字段，声明字段类型和是否能为空）

create table tab\_name(col1 type, col2 type,col3 type…)

2) 创建表（select语句创建）

create table tab\_name(a int not null auto\_increment,primary key(a),key(visitor)) select visitor from guestbook

意思就是：创建一个表，其中a是新增的字段，visitor是从guestbook表中提取出来的字段，都设置为主键, 注意如果在拷贝数据进表时发生任何错误，表将自动被删除。

3) 可以通过选择一个表的全部内容（无 WHERE 子句）来拷贝一个表，或利用一个总是失败的 WHERE 子句来创建一个空表

create table tab1 select \* from table\_name(复制table\_name表中全部内容)

create table tab2 select \* from table\_name where 0;(创建一个带有table\_name表中全部字段的空数据表)

1. 修饰表结构（有时你可能需要改变一下现有表的结构，那么alter Table语句将是你的合适选择）
2. 增加列

alter table tab\_name add col\_name type

1. 删除列

alter table tab\_name drop col\_name

1. 改变列
2. alter table tab\_name modify col\_name type(只改变列类型)
3. alter table tab\_name change old\_col\_name new\_col\_name type(改变列名和类型)
4. 给表更名

alter table tab\_name rename new\_tab\_name

1. 删除数据表
2. drop table if exists tab\_name,tab2\_name,tab3\_name…
3. drop table database\_name.tab\_name(指定数据库下的表)

使用关键词if exists 避免不存在表的一个错误发生

1. 向数据表插入新数据
2. 向表中插入一条数据

insert into tab\_name (col\_name,col\_name)values(value1,value2)

insert into tab\_name set col\_name=value, col2\_name=value2….

1. 一次性向表中插入多条数据

insert into tab\_name(col\_name,col2\_name)values(val1,val2), (val3,val4),…..

3) 插入从其他表选择的行

insert into tab\_name (col1,col2) select col3,col4 from tab2\_name(指定字段)

insert into tab\_name select col1,col2,col3 from tab2\_name(全部字段)

1. replace，replace……select 语句插入

replace功能与insert完全一样，如果在表中的一个老记录具有在一个唯一索引上的新记录有相同的值，在新记录被插入之前，老记录被删除；insert语句的表现是产生一个错误。

如果要新增的数据在表中本来就存在，则会被替换成新的数据

由于REPLACE语句可能改变原有的记录，因此使用时要小心。

1. txt批量录入数据

注意数据字段之间用tab键隔开

1. 手动导入txt文本数据到新表

新建一个txt文本文件，例如file.txt

id name age xuehao

1 xiaosan 20 10101

2 xiaosi 21 10102

3 xiaowu 22 10103

4 xiaoliu 23 10104

手动导入即可，选择“.txt”文件

1. 用MySQL语法导入txt文本数据到新表

例如：创建一个txt文本

1 xiaosan 20 10101

2 xiaosi 21 10102

3 xiaowu 22 10103

4 xiaoliu 23 10104

在MySQL数据库中创建一个新表

create table file(

id int not null,

name text not null,

age int,

xuehao text,

PRIMARY key(id)（可要可不要）

)

将txt文本数据导入file新表中

load data local infile 'C:/Users/USER/Desktop/file.txt' into table file;

注意local表示本地地址，不加local就表示txt文本是服务器上的地址

地址写法，是/ 不是\ ，否则会报错

1. 查询数据表中的数据

SELECT selection\_list 选择哪些列

FROM table\_list 从何处选择行

WHERE primary\_constraint 行必须满足什么条件

GROUP BY grouping\_columns 怎样对结果分组

HAVING secondary\_constraint 行必须满足的第二条件

ORDER BY sorting\_columns 怎样对结果排序

LIMIT count 结果限定

1. 进行表达式计算

select num1-num2 from tab\_name

1. MySQL允许对表达式求值而不引用任何表

select (2+3\*4.5)/2.5

1. 条件查询
2. having：是第二条件，在与其他从句配合使用，显然不能在where子句中的项目使用having。
3. where运算符



1. between and连用筛选

select col\_name…from tab\_name where

1. order by排序

select col\_name… from tab\_name order by col\_name

select col\_name… from tab\_name where condition order by

1. 查询分组和行计数

group by col\_name….

1. 计数count是统计非NULL结果的数目

select count(\*) from tab\_name

select owner,count(\*) from pet group by owner;(统计每个主人有多少只宠物)

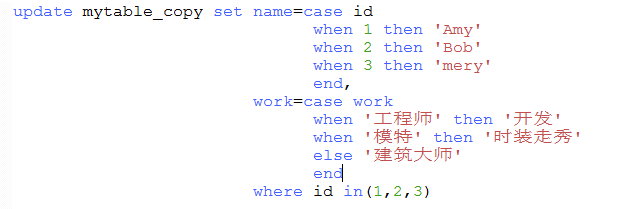
select species,count(\*) from pet group by species;(统计每个宠物的只数)

1. 查询多个表

select tab1.col1,tab2.col2… from tab1,tab2 where condition

1. 修改记录

update tab\_name set col\_name=value1,col1\_name=value2…where condition



1. 删除记录

delete from tab\_name where condition

where 表示可选，如果不加where将会删除全部记录

使用时一定注意，如果需要恢复数据，可以利用MySQL的binlog，参照文档[《binlog数据恢复》](binlog数据恢复.docx)

1. 事件函数

count(): 统计费NULL集合

sum(): 计和

avg(): 平均数

max(): 最大

min(): 最小

curdate()：返回当前日期 2017-12-14

curdate()+0：返回20171214

curtime(): 返回当前时间 09:24:13

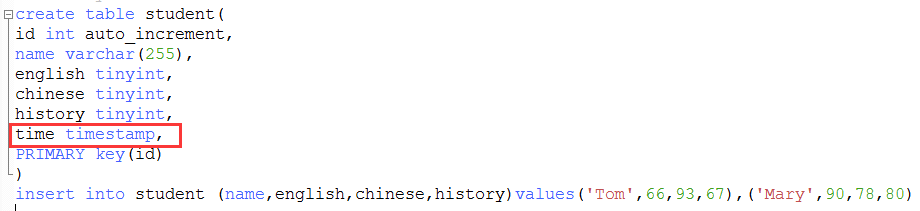
curtime()+0：返回92413

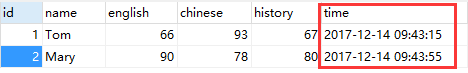
now()：返回当前准点2017-12-14 09:31:25

now()+0：返回20171214093125

1. 自动记录改变

timestamp()函数的使用如图：

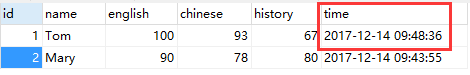


  
time字段为系统自动记录的时间

再次进行select查询操作，time信息不发生变化，还是原本insert数据时的时间

如果进行update更新操作

update student set english=100 where id=1;



可以看到id=1的time信息发生改变，因为你对该条数据进行了更改，所以系统会记录你最新操作数据的时间；

如果你想在update时不改变time的值，你可以这样操作：

update student set english=100,time=time where id=1;

此时time还是原来的值，不发生改变

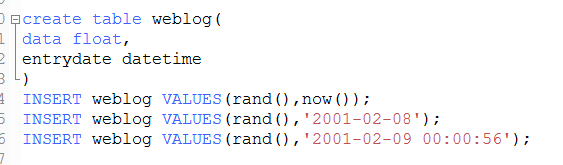
tips:

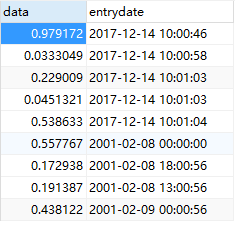
now()和timestamp()的功能类似，区别在于，now()的存储需求是8字节，timestamp()的存储需求是4字节，更节省空间。

1. 返回日期和时间范围

当你需要查看某个时间段的数据时，可以这样操作：

1. 新建一个表weblog，插入几条数据，其中rand()用于随机生成浮点数，now()当前时间，多执行几次就可以随机生成多条数据，最后加上我们的测试数据





1. 选取你想要查看的数据，比如想要查看2001-2-8这一天的数据

你有以下几种方式可选：

1. 使用关系运算符进行的是比较过程，时转换成内部的存储格式后进行的，时间的书写可以不是那么严格要求；

SELECT \* FROM weblog WHERE entrydate>="2001-02-08";

SELECT \* FROM weblog WHERE entrydate>="2001-2-8";

SELECT \* FROM weblog WHERE entrydate>="2001\*02\*08";

SELECT \* FROM weblog WHERE entrydate>="20010208";

SELECT \* FROM weblog WHERE entrydate>="2001/2/8";

1. 使用LIKE模糊匹配查询，是通过比较串值进行的，因此必须使用标准的时间书写格式，YYYY-MM-DD HH-MM-SS：

SELECT \* FROM weblog WHERE entrydate LIKE '2001-02-08%' ;

1. 比较日期和时间

to\_days()函数的用法：

给出一个日期date，返回一个天数(从0年的天数)，date可以是一个数字，也可以是一个串值，当然更可以是包含日期的时间类型。

select TO\_DAYS(960501);

select TO\_DAYS('1997-07-01');

select to\_days(now())-to\_days('20010101');

1. 标准的字符串模式匹配

select \* from pet where name like ‘\_’; 匹配任意单个字符

select \* from pet where name like ‘\_\_\_\_’; 匹配长度为4的字符

……

select \* from pet where name like ‘ab%’; 匹配以ab开头的字符

select \* from pet where name like ‘%ab’; 匹配以ab结尾的字符

select \* from pet where name like ‘%ab%’; 匹配包含ab的字符

select \* from pet where name regexp ‘[abc]’; 匹配包含a、b、c任意字符

select \* from pet where name regexp ‘^[abc]’; 匹配以a、b、c任意字符开头

select \* from pet where name regexp ‘[abc]$’; 匹配以a、b、c任意字符结尾

select \* from pet where name regexp ‘^…..$’;匹配正好五个字的字符

tips:

模式匹配中，没有!=或者<>，只能写成like（包含）或者not like （不包含）

1. 别名

as 的使用

select name as n, date as d from pet;

返回的数据列名发生改变

as查询不允许你在一个WHERE子句中引用一个别名，因为WHERE代码被执行时，列值还可能没有终结。

1. 数据去重

distinct关键字

1. NULL值

一般我们用 is null 或者 is not null来判断值是否为NULL

SELECT 1 is null,1 is not null;



其中0表示false，1表示true

也可以用函数isnull(str)来判断是否为空

tips:

NULL≠空字符串

1. 大小写敏感性
2. 数据库和表

Unix上区分大小写，win32忽略大小写，但在同一查询中请使用相同的大小写来引用给定的数据库和表。

1. 列名

在所有情况下都是忽略大小写的

1. 表的别名

是区分大小写的

1. 列的别名

不区分大小写

1. 字符串比较和模式匹配

MySQL搜索是大小写不敏感的

如果你想要使这个搜索大小写敏感，使用INDEX(col\_name, "A")=0检查一个前缀

如果列值必须确切是"A"，使用STRCMP(col\_name, "A") = 0

你想要一个列总是被当作大小写敏感的方式，声明它为binary

1. left join

select \* from pet left join student on pet.name=student.name;

左表匹配右表，如果右表没有满足的条件，左表依旧返回自己的数据

1. 对表进行修改调整

alter table语句的使用:

创建列的索引alter table student add index mark(english,Chinese,history)

查看创建的索引 show index from student

删除信息 alter table student drop id;