

MySQL数据库(二)

数据类型

MySQL中的数据类型有很多，主要分为三类：数值类型、字符串类型、日期时间类型

数据类型名称	大小	描述
INT/INTEGER (int/integer)	4bytes	普通大小的整数，-2147483648到2147483647，0到4294967295
DOUBLE[(M,D)] (double)	8bytes	普通大小(双精度)浮点数，允许值-1.7976931348623157E+380到-2.2250738585072014E-308,0和2.2250738585072014E-38到1.7976931348623157E+308.这些是理论限制，基于IEEE标准。实际的范围根据硬件或操作系统的不同可能稍微小些
DATE (date)		日期，支持的范围为‘1000-01-01’到‘9999-12-31’，MySQL以‘YYYY-MM-DD’格式显示DATE值，但允许使用字符串（‘20230721’/‘2023-07-21’）或数字(20230721)为DATE列分配值
DATETIME (datetime)		日期和时间的组合。支持的范围是‘上面加上00:00:00’到‘上面第二个加上23:59:59’.MySQL以‘YYYY-MM-DD HH:MM:SS’格式显示DATETIME值，但允许使用字符串或数字为DATETIME列分配值
TIMESTAMP (timestamp)		时间戳，范围是‘1970-01-01 00:00:00’到2037年
TIME (time)		时间，范围是‘-838: 59: 59’到‘838: 59: 59’.MySQL以‘HH:MM:SS’格式显示TIME值，但允许使用字符串或数字为TIME列分配值
YEAR (year)		两位或四位格式的年。默认是四位格式。在四位格式中，允许的值是1901到2155和0000.在两位格式中，允许的值是70-99和00-69，表示从1970到2069年。MySQL以yyyy格式显示YEAR值，但允许使用字符串或数字为YEAR列分配值
CHAR(M) (char)	0-255bytes	固定长度字符串，当保存时在右侧填充空格以达到指定长度。M表示列长度。M的范围是0到255个字符
VARCHAR(M) (varchar)	0-65535bytes	变长字符串。M表示最大列长度。M的范围是0到65535.（VARCHAR的最大实际长度由最长的行的大小和使用的字符集确定。最大有效长度是65535字节）

数据类型名称	大小	描述
...

例如：

1	数值类型示例：
2	1). 年龄字段 - 不会出现负数，而且人的年龄不会太大
3	age tinyint unsigned (0-255)
4	
5	2). 分数 - 总分100分，最多出现一位小数
6	score double(4,1)
7	
8	日期时间示例：
9	1). 生日字段 birthday
10	birthday date
11	
12	2). 创建时间 createtime
13	createtime datetime
14	
15	字符类型示例：
16	1). 用户名 username -----> 长度不定，最长不会超过50
17	username varchar(50) - > 2 2
18	
19	2). 性别 gender -----> 存储值，不是男,就是女
20	gender char(10) - > 1 10
21	
22	3). 手机号 phone -----> 固定长度为11
23	phone char(11)

✧ 创建表

- i char 与 varchar 都可以描述字符串，char是定长字符串，指定长度多长，就占用多少个字符，
和字段值的长度无关。而varchar是变长字符串，指定的长度为最大占用长度。

```

1 create table [if not exists] tab_name(
2     col_name datatype [comment '注释'],
3     col_name datatype
4 );
5
6 # 创建一个 - 学生表: 学生编号、学生姓名、出生日期
7 create table if not exists student (id int comment '编号', name
  varchar(50) comment '姓名', birth date comment '出生日期');

```

✧ 表操作-修改 DDL

- 添加列

```
1 alter table 表名 add [column] 字段名 类型;
```

- 修改列的类型

```
1 alter table 表名 modify [column] 字段名 类型;
```

- 修改列名称和类型

```
1 alter table 表名 change [column] 原字段名 新字段名 新列类型
  [comment 注释] [约束];
```

- 删除列

```
1 alter table 表名 drop [column] 字段名;
```

- 修改表名

```
1 rename table 旧名称 to 新名称;
2 -- 或者
3 alter table 表名称 rename to 新名称;
```

- 删除表

```
1 drop table [if exists] 表名;
```

- 截断表

```
1 truncate table 表名;
2 # 删除表之后会创建一个空表
```



删除表的时候，表中的全部数据也都会被删除

- 创建和某表结构一样的表

```
1 create table 表名 like 要复制的表;
2
3 # 当要复制的表不在当前数据库时
4 create table 表名 like 数据库名.要复制的表;
```

✧ 数据管理

1 插入数据 DML

```
1 insert into 表名[(字段名称1, 字段名称2, 字段名称3)] value(value
  1, value 2, value 3);
2
3 insert into 表名[(字段名称2, 字段名称21, 字段名称3)]
  values(value 2, value 1, value 3);
4
5 # 如果前面有字段按照前面的字段顺序添加, 没有按照默认的顺序添加
6 # 插入多条数据
7 insert into 表名 values (), (), ();
```

2 查询数据 DQL

```
1 # 无条件查询
2 select *(字段1, 字段2, 字段3, ... ) from 表名;
3
4 # 等值查询
5 select * from 表名 where 字段名 = 值;
6
7 # 设置列别名
8 select 字段名 [as] '别名' from 表名 WHERE 条件;
9
10 # 剔除重复行
11 # 只是在显示的时候不显示重复的数据, 数据未删除
12 select distinct * from 表名;
```

3 修改数据

```
1 update 表名 set 字段名 = 值 where 条件;
2 # 修改满足条件的字段值
```

4 删除数据

```
1 delete from 表名 where 条件;
2 # 删除满足条件的行
```

✧ 数据备份

mysqldump [选项] 数据库名 [表名] > 地址

选项说明:

参数名	缩写	含义
--host	-h	主机地址
--port	-P	服务器端口号
--user	-u	MySQL用户名
--password	-p	MySQL密码
--databases	-B	指定要备份的数据库
--all-databases	-A	备份MySQL服务器上的所有数据
--no-data	-d	不备份数据，默认为备份数据
--comments	-i	是否有信息备注，默认为打开，使用--skip-comments关闭

1 备份表结构

```

1  mysqldump -u root -p 数据库名 表1 表2 > 地址
2
3  # -d 只备份表结构
4  mysqldump -u root -p -d kfm07 dept > E:/kfm/dept.sql
5  # 数据+表结构一起备份
6  mysqldump -u root -p kfm07 > e:/kfm/kfm07.sql
7  # 关闭信息备注
8  mysqldump -u root -p -d --skip-comments store cart >
   E:/cart.sql

```

2 备份多个数据库

```

1  mysqldump -u root -p -d --databases 数据库1 数据库2 > 地址
2
3  mysqldump -u root -p -d -B kfm07 kfm007 book bank >
   E:/test.sql
4
5  # --all-databases 备份所有数据库
6  mysqldump -u root -p -d --all-databases 或 -A >
   E:/allbackupfile.sql

```

3 备份数据和结构

```

1  # 备份命令去掉 -d
2  mysqldump -u root -p --databases 数据库1 数据库2 > 地址

```

4 将查询的结果集保存为文件

```

1  mysql -u root -p -e "select * from 数据库.表名" > 地址
2
3  mysql -u root -p -e "select * from store.goods" >
   E:/result.txt
4  mysql -u root -p -e "select * from goods" store >
   E:/result.csv

```

5 还原数据库结构和数据

```

1  # 登录选中数据库之后执行，将数据还原到该数据库
2  source 地址; # 地址里是有SQL语句的文件
3
4  # 在服务器外面使用mysql命令还原
5  `新数据库需要提前建好`
6  mysql -u root -p 新数据库名 < 地址
7
8  mysql -u root -p kfm < E:/cart.sql

```

✧ 条件查询

1 条件过滤

```

1  SELECT * FROM 表名;
2  select * from 表名;
3
4  SELECT * FROM 表名 WHERE 字段名 = 值;
5  select * from 表名 where 字段名 = 值;
6
7  # WHERE 后面表示查询的条件,两个值之间的内容全部查出来
8  # [value1, value2]
9  SELECT * FROM 表名 WHERE 字段名 BETWEEN value1 AND value2;
10 select * from 表名 where 字段名 between value1 and value2;
11
12 select * from dept where dept_no between 1 and 3;
13
14 # 字段在[value1, value2 ....]任意一个就可以
15 SELECT * FROM 表名 WHERE 字段名 IN (value1, value2,
   value3);
16 select * from 表名 where 字段名 in (value1, value2,
   value3);
17
18 # 字段不在[value1,value2,value3]中,除了里面的值之外全部查出来
19 SELECT * FROM 表名 WHERE 字段名 NOT IN(value1, value2,
   value3);

```

```

20 select * from 表名 where 字段名 not in(value1, value2,
    value3);
21
22 -- & 条件1 和 条件2 都为 true 的结果显示出来
23 SELECT * FROM 表名 WHERE 条件1 AND 条件2;
24
25 -- | 满足条件1 和 满足条件2的结果都可以显示出来
26 SELECT * FROM 表名 WHERE 条件1 OR 条件2;

```

2 算数运算

```

1 # 可以对查询出来的结果进行算数运算
2 SELECT 字段名1 + 字段名2 FROM 表名;
3 select 字段名1 + 字段名2 from 表名;
4
5 # 修改的时候也可以进行算数运算
6 UPDATE 表名 SET 字段名 = 字段名 + 值 WHERE 条件;
7 update 表名 set 字段名 = 字段名 + 值 where 条件;

```

3 NULL值查询

```

1 # NULL 值只能无法通过等值操作查询
2 # NULL值指的是未填值，注意跟空字符串做区分
3 SELECT * FROM 表名 WHERE 字段名 IS NULL;
4 select * from 表名 where 字段名 is NULL;
5
6 SELECT * FROM 表名 WHERE 字段名 IS NOT NULL;
7 select * from 表名 where 字段名 is not NULL;

```

4 模糊查询

```

1 # LIKE 模糊查询 _ 表示一个任意字符 %表示零个或多个任意字符
2 # xa xb
3 SELECT * FROM 表名 WHERE 字段名 LIKE 'x_';
4 select * from 表名 where 字段名 like 'x_';
5
6 # xa x xbs
7 SELECT * FROM 表名 WHERE 字段名 LIKE 'x%';
8 select * from 表名 where 字段名 like 'x%';
9
10 # xa ax axb axbc
11 SELECT * FROM 表名 WHERE 字段名 LIKE '%x%';
12 select * from 表名 where 字段名 like '%x%';
13
14 # ax axb axbc
15 SELECT * FROM 表名 WHERE 字段名 LIKE '_x%';
16 select * from 表名 where 字段名 like '_x%';

```

5 逻辑运算


```
1 # WHERE 条件语句里面可以写 > < = != 等
2 SELECT * FROM 表名 WHERE 字段名 > 值;
3 select * from 表名 where 字段名 > 值;
```

6 排序

```
1 # ASC 升序 (默认)
2 SELECT * FROM 表名 ORDER BY 字段名 ASC;
3 # DESC 降序
4 SELECT * FROM 表名 ORDER BY 字段名 DESC;
5
6 # ORDER BY 对结果集排序 DESC 降序,ASC 升序 (默认)
7 SELECT * FROM 表名 ORDER BY 字段名1 DESC, 字段名2 ASC;
8 select * from 表名 order by 字段名1 desc, 字段名2 asc;
9
10 # 先按照dept_no 降序排序, 如果dept_no字段相同, 按照d_name降序排
    序
11 select * from dept ORDER BY dept_no desc, d_name desc;
```

7 分页查询

```
1 # LIMIT 对结果集分页 参数1: 起始行; 参数2: 显示条数
2 # 页码 n 页大小 s LIMIT (n-1) * s, s
3
4 # 从第0行开始显示两条数据
5 SELECT * FROM 表名 LIMIT 0,2;
6 select * from 表名 limit 0,2;
```

8 单行函数

```
1 # length 计算长度
2 SELECT length("123");
3 select length(字段名) from 表名;
4
5 # upper/lower 大、小写转换
6 SELECT UPPER("a"), LOWER('A');
7 select upper(字段名), lower(字段名) from 表名;
8
9 # concat 字符串拼接
10 SELECT CONCAT(upper('smith'), 'john');
11 select concat(字段名1, 字段名2) from 表名;
```