数据类型

1. 整数类型

数据类型	字节数	无符号数的取值范围	有符号数的取值范围
TINYINT	1	0~255	-128~127
SMALLINT	2	0~65535	-32768~32768
MEDIUMINT	3	0~16777215	-8388608~8388608
INT	4	0~4294967295	-2147483648~ 2147483648
BIGINT	8	0~18446744073709551615	-9223372036854775808~9223372036854775808

2. 浮点数和定点数类型

数据类型	字节数	有符号的取值范围	无符号的取值范围
FLOAT	4	-3.402823466E+38~-1.175494351E-38	0和1.175494351E-38~3.402823466E+38
DOUBLE	8	-1.7976931348623157E+308~2.2250738585072014E-308	0和2.2250738585072014E-308~1.7976931348623157E+308
DECIMAL(M,D)	M+2	-1.7976931348623157E+308~2.2250738585072014E-308	0和2.2250738585072014E-308~1.7976931348623157E+308

从上图中可以看出：DECIMAL类型的取值范围与DOUBLE类型相同。

但是，请注意：DECIMAL类型的有效取值范围是由M和D决定的。

其中，M表示的是数据的长 度，D表示的是小数点后的长度。

比如，将数据类型为DECIMAL(6,2)的数据6.5243 插人数据库后显示的结果为6.52

3. 字符串类型

在MySQL中常用CHAR 和 VARCHAR 表示字符串。两者不同的是：VARCHAR存储可变长度的字符串。
当数据为CHAR(M)类型时，不管插入值的长度是实际是多少它所占用的存储空间都是M个字节；而VARCHAR(M)所对应的数据所占用的字节数为实际长度加1

4. 文本类型

文本类型用于表示大文本数据，例如，文章内容、评论、详情等

数据类型	储存范围
TINYTEXT	0~255字节
TEXT	0~65535字节
MEDIUMTEXT	0~16777215字节
LONGTEXT	0~4294967295字节

5. 日期与时间类型

数据类型	字节数	取值范围	日期格式	零值
YEAR	1	1901~2155	YYYY	0000
DATE	4	1000-01-01~9999-12-31	YYYY-MM-DD	0000-00-00
TIME	3	-838：59：59~ 838：59：59	HH:MM:SS	00:00:00
DATETIME	8	1000-01-01 00:00:00~9999-12-31 23:59:59	YYYY-MM-DD HH:MM:SS	0000-00-00 00:00:00
TIMESTAMP	4	1970-01-01 00:00:01~2038-01-19 03:14:07	YYYY-MM-DD HH:MM:SS	0000-00-00 00:00:00

TIME:使用CURRENT_TIME或NOW()输人当前系统时间。

DATETIME：

```
1 DATETIME类型用于表示日期和时间，它的显示形式为'YYYY-MM-DD HH: MM:SS'，其中，YYYY表示年，MM表示月，DD表示日，HH表示小时，MM表示分，SS 表示秒。在MySQL中，可以使用以下4种格式指定DATETIME类型的值。
2 □以'YYYY-MM-DD HH:MM:SS'或者'YYYYMMDDHHMMSS'字符串格式表示的日期和时间，取值范围为'1000-01-01 00:00:00'-'9999-12-3 23:59:59'。例如，输人'2019-01-22 09:01:23'或 '20140122_0_90123'插人数据库中的 DATETIME 值都为 2019-01-22 09:01:23。
3
4 1、以'YY-MM-DD HH:MM:SS'或者'YYMMDDHHMMSS'字符串格式表示的日期和时间，其中YY表示年，取值范围为'00'-'99'。与DATE类型中的YY相同，'00'- '69'范围的值会被转换为2000-2069范围的值，'70'-'99'范围的值会被转换为1970-1999范围的值。
5 2、以YYYYMMDDHHMMSS或者YYMMDDHHMMSS数字格式表示的日期 和时间。例如，插入20190122090123或者190122090123,插人数据库中的DATETIME值都 为 2019-01-22 09:01:23。
6 3、使用NOW来输人当前系统的日期和时间。
```

TIMESTAMP

```
TIMESTAMP类型用于表示日期和时间，它的显示形式与DATETIME相同但取值范围比DATETIME小。在此，介绍几种TIMESTAMP类型与DATATIME类型不同的形式：
1、使用CURRENT_TIMESTAMP输入系统当前日期和时间。
2、输入NULL时系统会输入系统当前日期和时间。
3、无任何输入时系统会输入系统当前日期和时间。
```

6. 二进制类型

在MySQL中常用BLOB存储二进制类型的数据，例如：图片、PDF文档等。

数据类型	储存范围
TINYBLOB	0~255字节
BLOB	0~65535字节
MEDIUMBLOB	0~16777215字节
LONGBLOB	0~4294967295字节

基本操作

1. 数据库基本操作

```
create database (数据库名称)
drop database (数据库名称)

查询出MySQL中所有的数据库MySQL命令：
show databases;

将数据库的字符集修改为gbk MySQL命令：
alter database db1 character set gbk;

切换数据库 MySQL命令：
use db1;

查看当前使用的数据库 MySQL命令：
select database();
```

2. 数据表基本操作

```
create table 表名(
          字段1 字段类型,
          字段2 字段类型,
          …
          字段n 字段类型
);
```

```
查看当前数据库中所有表 MySQL命令：
show tables;

查表的基本信息 MySQL命令：
show create table student;

查看表的字段信息 MySQL命令：
desc student;
```

```
修改表名 MySQL命令：
alter table student rename to stu;

修改字段名 MySQL命令：
alter table stu change name sname varchar(10);

修改字段数据类型 MySQL命令：
alter table stu modify sname int;

增加字段 MySQL命令：
alter table stu add address varchar(50);

删除字段 MySQL命令：
alter table stu drop address;
```

```
删除数据表
drop table 表名;
```

约束

约束条件	说明
PRIMARY KEY	主键约束用于唯一标识对应的记录
FOREIGN KEY	外键约束
NOT NULL	非空约束
UNIQUE	唯一性约束
DEFAULT	默认值约束，用于设置字段的默认值

1. 主键约束

非空且唯一

```
create table student(
id int primary key,
name varchar(20)
);
```

```
create table student01(
id int
name varchar(20),
primary key(id)
);
```

2. 非空约束

```
create table student02(
id int
name varchar(20) not null
);
```

3. 默认值约束

```
字段名 数据类型 DEFAULT 默认值;
```

```
create table student03(
id int,
name varchar(20),
gender varchar(10) default 'male'
);
```

4. 唯一约束

```
字段名 数据类型 UNIQUE;
```

```
create table student04(
id int,
name varchar(20) unique
);
```

5. 外键约束

```
-- 在创建数据表时语法如下：
CONSTRAINT 外键名 FOREIGN KEY (从表外键字段) REFERENCES 主表 (主键字段)
-- 将创建数据表创号后语法如下：
ALTER TABLE 从表名 ADD CONSTRAINT 外键名 FOREIGN KEY (从表外键字段)
REFERENCES 主表 (主键字段);
```

```
create table student05(
id int primary key,
name varchar(20)
);
```

```
create table class(
classid int primary key,
studentid int
);
```

学生表作为主表，班级表作为副表设置外键

```
alter table class add constraint fk_class_studentid foreign
key(studentid) references student05(id);
```

删除外键

```
alter table 从表名 drop foreign key 外键名;

alter table class drop foreign key fk_class_studentid;
```

DML-操作语言

1. 为所有字段插入数据

每个字段与其值是严格一一对应的。也就是说：每个值、值的顺序、值的类型必须与对应的字段相匹配。但是，各字段也无须与其在表中定义的顺序一致，它们只要与 VALUES中值的顺序一致即可。

```
INSERT INTO 表名（字段名1,字段名2,...) VALUES (值 1,值 2,...);

insert into student (id,name,age,gender) values (1,'bob',16,'male');
```

2. 同时插入多条数据

```
INSERT INTO 表名 [(字段名1,字段名2,...)]VALUES (值 1,值 2,…),(值 1,值 2,…),...;

insert into student (id,name,age,gender) values (2,'lucy',17,'female'),(3,'jack',19,'male'),(4,'tom',18,'male');
```

3. 更新数据

```
UPDATE 表名 SET 字段名1=值1[,字段名2 =值2,…] [WHERE 条件表达式];
```

```
update student set age=20,gender='female' where name='tom';
```

4. 删除数据

```
DELETE FROM 表名 [WHERE 条件表达式];
```

```
delete from student where age=14;
```

DQL-查询语句

1. 基本语法

1.1查询所有字段

```
select * from (表名);
```

1.2 查询指定字段

```
select sid,sname from student;
```

1.3 常数查询

用于进行标记（暂不常用）

```
select sid,sname,'2021-03-02' from student;
```

1.4 过滤重复数据

在使用DISTINCT 时需要注意：
在SELECT查询语句中DISTINCT关键字只能用在第一个所查列名之前。

```
select distinct gender from student;
```

1.5 算数运算符

在SELECT查询语句中还可以使用加减乘除运算符。

```
select sname,age+10 from student;
```

2. 函数

2.1 count()

```
select count(*) from student;
```

2.2 max()/min()

```
select max(age) from student;
```

2.3 sum()

计算指定列的数值和，如果指定列类型不是数值类型则计算结果为0

```
select sum(age) from student;
```

2.4 avg()

```
select avg(age) from student;
```

3. 条件查询

3.1 关系运算符

关系运算符	说明
=	等于
<>	不等于
!=	不等于
<	小于
<=	小于等于
>	大于
>=	大于等于

3.2 IN

IN关键字用于判断某个字段的值是否在指定集合中。如果字段的值恰好在指定的集合中，则将字段所在的记录将查询出来。

```
查询sid为S_1002和S_1003的学生信息
select * from student where sid in ('S_1002','S_1003');
```

```
查询sid为S_1001以外的学生的信息
select * from student where sid not in ('S_1001');
```

3.3 BETWEEN AND

BETWEEN AND用于判断某个字段的值是否在指定的范围之内。如果字段的值在指定范围内，则将所在的记录将查询出来

```
查询15到18岁的学生信息
select * from student where age between 15 and 18;
```

```
查询不是15到18岁的学生信息
select * from student where age not between 15 and 18;
```

3.4 null

```
select * from student where sname is not null;
```

3.5 AND

在MySQL中可使用AND关键字可以连接两个或者多个查询条件。

```
select * from student where age>15 and gender='male';
```

3.6 OR

在使用SELECT语句查询数据时可使用OR关键字连接多个查询条件。在使用OR关键字时，只要记录满足其中任意一个条件就会被查询出来

```
查询年纪大于15或者性别为male的学生信息
select * from student where age>15 or gender='male';
```

3.7 LIKE

MySQL中可使用LIKE关键字可以判断两个字符串是否相匹配

%用于匹配任意长度的字符串。例如，字符串“a%”匹配以字符a开始任意长度的字符串

```
查询学生姓名以li开始的记录
select * from student where sname like 'li%';
```

```
查询学生姓名以g结尾的记录
select * from student where sname like '%g';
```

```
查询学生姓名包含s的记录
select * from student where sname like '%s%';
```

下划线通配符只匹配单个字符，如果要匹配多个字符，需要连续使用多个下划线通配符。例如，字符串“ab_”匹配以字符串“ab”开始长度为3的字符串，如abc、abp等等；字符串“a__d”匹配在字符“a”和“d”之间包含两个字符的字符串，如"abcd"、"atud"等等。

```
查询学生姓名以zx开头且长度为4的记录
select * from student where sname like 'zx__';
```

```
查询学生姓名以g结尾且长度为4的记录
select * from student where sname like '___g';
```

3.8 LIMIT

当执行查询数据时可能会返回很多条记录，而用户需要的数据可能只是其中的一条或者几条

```
查询学生表中年纪最小的3位同学
select * from student order by age asc limit 3;
```

3.9 GROUP BY-分组查询

```
统计各部门员工个数
select count(*), departmentnumber from employee group by
departmentnumber;
```

```
统计部门编号大于1001的各部门员工个数
select count(*), departmentnumber from employee where
departmentnumber>1001 group by departmentnumber;
```

```
统计工资总和大于8000的部门 MySQL命令
select sum(salary),departmentnumber from employee group by
departmentnumber having sum(salary)>8000;
```

> 注：having与group by联合使用，效果与where类似

3.10 ORDER BY

```
SELECT 字段名1,字段名2,…
FROM 表名
ORDER BY 字段名1 [ASC ｜ DESC],字段名2 [ASC | DESC];
```

asc-升序 desc-降序

通常情况下，ORDER BY子句位于整个SELECT语句的末尾。

```
select * from student order by age asc;
```

多表查询

1. 内连接查询

INNER JOIN用于连接两个表，ON来指定连接条件；其中INNER可以省略。

```
SELECT 查询字段1,查询字段2, ...
FROM 表1 [INNER] JOIN 表2
ON 表1.关系字段=表2.关系字段
```

2. 外连接查询

在使用内连接查询时我们发现：返回的结果只包含符合查询条件和连接条件的数据。但是，有时还需要在返回查询结果中不仅包含符合条件的数据，而且还包括左表、右表或两个表中的所有数据，此时我们就需要使用外连接查询。外连接又分为左(外)连接和右(外)连接。

```
SELECT 查询字段1,查询字段2, ...
FROM 表1 LEFT | RIGHT [OUTER] JOIN 表2
ON 表1.关系字段=表2.关系字段
WHERE 条件
```

1. LEFT [OUTER] JOIN 左(外)连接：返回包括左表中的所有记录和右表中符合连接条件的记录。
2. RIGHT [OUTER] JOIN 右(外)连接：返回包括右表中的所有记录和左表中符合连接条件的记录

子查询

子查询是指一个查询语句嵌套在另一个查询语句内部的查询；该查询语句可以嵌套在一个SELECT、SELECT...INTO、INSERT...INTO等语句中。在执行查询时，首先会执行子查询中的语句，再将返回的结果作为外层查询的过滤条件。在子查询中通常可以使用比较运算符和IN、EXISTS、ANY、ALL等关键字。

```
select * from class where cid=(select classid from student where sname='张三');
```

1. EXISTS

EXISTS关键字后面的参数可以是任意一个子查询， 它不产生任何数据只返回TRUE或FALSE。当返回值为TRUE时外层查询才会执行

```
select * from class where exists (select * from student where sname='王五');
```

2. ANY

ANY关键字表示满足其中任意一个条件就返回一个结果作为外层查询条件。

```
select * from class where cid > any (select classid from student);
```

3. ALL

ALL关键字与ANY有点类似，只不过带ALL关键字的子查询返回的结果需同时满足所有内层查询条件。

```
select * from class where cid > all (select classid from student);
```

分页查询

```
SELECT
  *
FROM
  TABLE
LIMIT (Page - 1) * PageSize, PageSize ;
```

> LIMIT (当前页数-1) * 每页条数，每页条数