字段类型

整数

- int

java里面的byte,short,int对应数据库里都可以用int

- bigint

Java里面的long对应数据库里的bigint

小数（浮点数）

- decimal

长度： 是整数部分和小数部分长度和，比如长度为4，那么整数位数加上小数位数不超过4

小数位数：四舍五入保留指定的小数位数

字符

- varchar

可变长度，如果实际保存的内容小于字段的长度，存储在硬盘上按实际长度分配硬盘空间

长度指的是字符（包含）数量

- char

固定长度，如果实际保存的内容小于长度，存储在硬盘上也会按字段长度占用硬盘空间

布尔型

在数据库里面可以用int, bit（0, 1）,char来保存

日期

- date

  只保存日期
- time

  只保存时间
- datetime

  1, 可以保存的时间范围更大，1000年到9999年

  2，保存的绝对时间（任何时区读出来都是一样的）
- timestamp(时间戳)

  1，可以保存时间范围小，1970~2038

  2，保存的时区对应的时间（不同的时区，读出来显示当前时区的时间）

大文本类型

- text（CLOB）

保存一篇文章，甚至一本小说。不用设置长度

二进制（图片，音频，视频）

- BLOB

  保存二进制文件。不用设置长度

备注：一般不要把二进制文件存入数据库，而是存到硬盘目录，在数据库只用varchar的字段来保存它的路径和文件名

SQL语句

新增（插入）数据：insert into

- 所有字段都插入

这种情况可以省略字段名，按默认顺序插入相应的值

```
insert into emp values(3,'小李',22);
```

- 除自动增长的主键外，其它字段全部插入

```
insert into emp(emp_name, emp_age) values('小张', 23);
```

- 只插入部分字段

可以省略不用插入的字段类型有：

1. 自动增长的主键
2. 允许为null的字段
3. 不允许为null, 但是有默认值

```
insert into emp(emp_name) values('小陈');
```

查询语句

- 全表查询（查询所有列，所有字段）

```
select * from emp;
```

- 按条件查询

```
select * from student where stu_name = '邓智友'; -- 查询姓名是邓智友的学生

select * from student where stu_sex != 1; -- 查询性别不等于1的学生（女生）
select * from student where stu_sex <> 1; -- 不等于的另一种写法

select * from student where stu_age > 20;  -- 查询年龄大于20（不包含）的学生
select * from student where stu_age >= 20; -- 查询年龄大于等于20的学生
```

```
select * from student where stu_name like '%邓%'; -- 模糊查询，用like，  内容用%代替模糊内容
select * from student where stu_name like '邓%';
select * from student where stu_name like '%友';

select * from student where stu_birthday is null;   -- 查询生日为null的学生
select * from student where stu_birthday is not null;  -- 查询生日不为null的学生

select * from student where stu_id in(2,4,5);  -- 查询编号为2，4，5的学生
select * from student where stu_id not in(2,4,5); -- 查询编号不在列表中的学生（不是2，4，5）
```

- 多条件组合查询（and ，or）

```
select * from student where stu_age>18 and stu_sex=1;  -- 查询18岁以上的男生
select * from student where stu_id =2 or stu_id=4 or stu_id=5; -- 查询学生编号等于2，或者等于4，  或者等于5
select * from student where (stu_id =2 or stu_id=4 or stu_id=5) and stu_age>=18;
-- 如果多个条件，关系不清楚时通过小括号使添加更加清晰
```

- 用limit实现分页（limit不是所有数据库都支持）

```
/*
每页（page）显示两行(rows)：
第一页：1，2，  开始0 = （page-1)*rows
第二页：3，4，  开始2 = (2-1)*2
第三页：5，6，  开始4 = (3-1)*2
*/
select * from student limit 0,2; -- limt第一个参数是起始行（0开始），第二个参数是查询多少行
select * from student limit 2,2; -- 查询第二页
select * from student limit 4,2; -- 查询第三页
```

- 指定要查询的字段名。 还可以给字段名起别名

```
select stu_name,stu_age from student; -- 按指定字段名查询
select stu_name as 姓名, stu_age as 年龄 from student; -- 给字段名起个别名
select stu_name 姓名, stu_age 年龄 from student; -- 别名的简写方式（省略as）
```

- 统计表的记录总数

```
select count(*) from student; -- 统计表的总记录
```

- 去掉重复的记录 distinct

```
select distinct stu_age from student; -- distinct 去掉重复记录
```

排序

- 排序关键字是order by, 默认是升序（asc）, 降序是desc

- 排序字段可以有多个，每个字段可以分别设置升序或降序，先按前面的字段排序，如果顺序相同再以此看后面的字段
- 如果跟where条件一起使用，order by 是要放在where条件后面

```
select * from student order by stu_age; -- order by ,排序,默认是升序
select * from student order by stu_age desc; -- 降序
/*
多个字段排序，首先按前面字段排序，如果相同再按后面一个字段排序
每个字段可以单独设置升序或降序。升序是asc， 不指明升序还是降序情况默认用的asc
*/
select * from student order by stu_age desc, stu_name asc;
select * from student where stu_sex=1 ORDER BY stu_age; -- 如果排序和条件查询一起使用，order by 要放到where 后面
```

分组查询

```
select stu_sex, count(*) from student group by stu_sex; -- 按性别分组，查询每种性别的数量
select stu_sex, max(stu_age) from student group by stu_sex; -- 按性别分组，查询每种性别最大的年龄
select stu_sex, min(stu_age) from student group by stu_sex; -- 按性别分组，查询每种性别最大的年龄
select stu_sex, avg(stu_age) from student group by stu_sex; -- 按性别分组，查询每种性别平均年龄
select stu_sex, SUM(stu_age) from student group by stu_sex; -- 按性别分组，查询每种性别求和

select stu_sex, SUM(stu_age) from student group by stu_sex order by stu_sex; -- 分组和order by 结合使用
select stu_sex, SUM(stu_age) from student where 1=1 group by stu_sex ; -- 分组和where条件结合使用，where也要放到group by 前面
select stu_sex, avg(stu_age) avg_age from student where 1=1 group by stu_sex
HAVING avg_age>20; -- 对分组的结果再条件筛选用having
```

MySql常用函数

> 不同数据库服务器函数不一样

```
-- 数字相关函数
select ROUND(dt_float) from datatype  -- 取整,四舍五入
select round(3.14);

-- 字符串相关函数
select REPLACE('123456789','45','四五'); -- 替换字符串
select trim('  fdjsak   ');  -- 去掉字符串前后空格
select SUBSTR('1234567890', 4, 2); -- 截取字符串
select RIGHT('1234567890',3); -- 截取字符串右边几位
select LEFT('1234567890',3);  -- 截取字符串左边几位
select LOCATE('56','1234567890'); -- 查找一个子字符串在另一个字符串中的位置
select INSTR('1234567890','56'); -- 查找一个子字符串在另一个字符串中的位置
select LENGTH('1234567890中文'); -- 返回字符串的长度，中文字符占3个长度
select CONCAT('Hello ','HQYJ ' , '.'); -- 字符串拼接

-- 日期相关函数
select DATE_ADD(now(), INTERVAL 1 DAY); -- 日期加上一个段时间（年，月， 日，时，分，秒）
```

```sql
select DATE_ADD(now(), INTERVAL -1 DAY); -- 加一个负值，就相当于是减
select DATEDIFF(now(), '2020-1-1'); -- 两个日期相减，得到相差的天数
select year(now());
select MONTH(now());
select day(now());
select HOUR(now());
select MINUTE(now());
select SECOND(now());
select LAST_DAY(now()); -- 返回指定日期的最后一天
select DAYOFYEAR(now()); -- 返回指定日期在一年中是第几天
select DAYOFWEEK(now()); -- 返回星期几
select DAYOFMONTH(now());  -- 返回指定日期是当月的第几天
```