day02_表和表中记录操作

一.表中字段的操作

> 字段/列的英文单词: ==column==
>
> 操作字段/列的本质就是在修改表

知识点:

```
在表中添加一列: alter table 表名 add  字段名 字段类型;

在表中删除一列: alter table 表名 drop  字段名;

在表中修改列名和类型: alter table 表名 change 旧字段名 新字段名 新字段类型;
在表中只修改类型: alter table 表名 modify  旧字段名 新字段类型;
注意: 建议大家以后都用change,因为change既能单独修改名也能单独修改类型而且还可以同时修改名称和类型,而modify只能改类型

查看指定表中字段信息: desc 表名;
```

示例:

```
# 二.演示表中字段的操作
# 在表中添加一列: alter table 表名 add [column] 字段名 字段类型;
# 需求1:添加一列class,类型为字符串
alter table stu1 add column class varchar(100);
# -- 注意: column可以省略
alter table stu1 add  hight double;

# 在表中删除一列: alter table 表名 drop [column] 字段名;
# 需求2:删除age这一列
alter table stu1 drop column age;
# -- 注意: column可以省略
alter table stu1 drop hight;

# 在表中修改列名和类型: alter table 表名 change [column] 旧字段名 新字段名 新字段类型;
# 需求3:修改id名为sid,同时修改类型为字符串
alter table stu1 change column id sid varchar(100);
-- 注意: change也可以单独修改类型,注意名称即使不修改也要写两遍
alter table stu1 change column sid sid int;

# 在表中只修改类型: alter table 表名 modify [column] 旧字段名 新字段类型;
# 需求4:修改class字段类型为int
alter table stu1 modify column class int;
# 注意: 建议大家以后都用change,因为change既能单独修改名也能单独修改类型而且还可以同时修改

# 查看指定表的结构: desc 表名;
desc stu1;
```

二.表中记录的操作

知识点:

```
往表中插入记录:
    插入一条: insert into 表名 (字段1名,字段2名, ...) values (值1,值2, ...);
    插入多条: insert into 表名 (字段1名,字段2名, ...) values (值1,值2, ...) , (值1,值
2, ...) , (值1,值2, ...);
    注意: 如果插入的记录是包含所有字段,那么表名后面可以省略字段,默认就是所有字段

在表中修改记录:
    修改部分记录: update 表名 set 字段名=值 wehre 条件;
    修改所有记录(慎用): update 表名 set 字段名=值;

从表中删除记录:
    删除部分记录: delete from 表名 where 条件;
    删除所有记录方式1: delete from 表名;
    删除所有记录方式2: truncate [table] 表名;
```

示例:

```
# 三.演示表中记录的操作
# 1.插入操作
# 操作记录的前提要有表
# 需求: 往stu1表中一次插入1条数据
insert into stu1 (sid, name, class) values (1,'张三',52);
insert into stu1 (sid, name) values (1,'张三');
-- 注意: 如果插入的数据是所有字段,那么可以不指定字段插入
insert into stu1 values (2,'李四',52);

# 需求: 往stu1表中一次插入多条数据
insert into stu1 (sid, name, class) values (3,'王五',52),(4,'赵六',52),(5,'田
七',52);
-- 注意: 如果插入的数据是所有字段,那么可以不指定字段插入
insert into stu1 values (6,'王五2',52),(7,'赵六2',52),(8,'田七2',52);



# 2.修改操作
# 需求: 修改李四的班级为53期
update stu1 set class=53 where name = '李四';
# 需求: 修改所有学生班级改为54
update stu1 set class=54; -- 慎用! 不加条件会有警告,如果执意运行,还会再次提醒两个选择,建议
选择excute



# 3.删除操作
# 需求: 删除李四这条记录
delete from stu1 where name = '李四';
# 需求: 使用delete删除所有学生信息
delete from stu1; -- 慎用! 不加条件会有警告,如果执意运行,还会再次提醒两个选择,建议选择
excute

# 为了演示truncate清空所有需要再次插入数据
insert into stu1 (sid, name, class) values (3,'王五',52),(4,'赵六',52),(5,'田
七',52);
```

```
# 需求: 使用truncate删除所有数据
truncate table stu1;
truncate stu1; -- table可以省略
# 注意: delete和truncate都能删除所有数据,具体有什么区别呢? 后续讲到主键自增的时候补充
```

三.SQL约束

主键约束: 非空唯一

```
主键约束关键字: primary key

主键约束特点: 非空唯一(限制主键列的数据不能为空,不能重复)
```

```
# 四.演示约束
# 1.主键约束特点: 限制数据非空唯一
create table stu2(
    sid int PRIMARY KEY ,  -- 建议: 建表的时候添加主键约束
    name varchar(100),
    class int
);


123678
# 验证主键约束的限制作用
# 限制了主键不能重复
insert into stu2 values(1,'张三',52); -- 第一条正常插入
insert into stu2 values(1,'李四',53); -- 插入失败,因为有主键限制不能重复
# 限制了主键不能为空
insert into stu2 values(null,'李四',53); -- 插入失败,因为有主键限制不能为空
```

主键自增约束

```
主键自增关键字: PRIMARY KEY auto_increment

主键自增特点: 每次自动加1

注意: 设置了自增后null和0就是一个占位符,代表使用自增  或者  插入数据的时候不指定主键,默认自增
```

```
# 演示主键自增
create table stu4(
    sid int PRIMARY KEY auto_increment ,  -- 建议: 建表的时候添加主键自增约束
    name varchar(100),
    class int
);
# 查看表结构
desc stu4;

# 演示主键自增特点: 自动加1
# 注意: 如果设置了主键自增,null和0都代表自动使用自增
# 没有指定字段插入数据,默认所有字段,null和0其实就是一个占位代表自增
insert into stu4 values(null,'李四',53);
insert into stu4 values(0,'王五',53);
# 指定字段的时候不指定id字段,默认自增
insert into stu4(name,class) values('赵六',54);
```

delete和truncate区别

> 共同点: 都能删除所有数据
> 不同点:
> delete方式删除所有数据,但是自增顺序会保留(再次插入数据,继续自增)
> truncate方式删除所有数据,自增顺序也会被重置(再次插入数据,从1开始重新自增)

```
# 演示 delete方式删除所有数据,但是自增顺序会保留(再次插入数据,继续自增)
delete from stu4; -- 有警告,不建议使用此种方式删除所有数据
# 再次插入数据,自增顺序从删除前最后一次开始继续自增
insert into stu4(name,class) values('张三',52),('李四',53),('王五',54);

# 演示truncate方式删除所有数据,自增顺序也会被重置(再次插入数据,从1开始重新自增)
truncate stu4; -- 如果要删除所有数据,使用此种方式
# 再次插入数据,自增顺序重置,从1开始重新自增
insert into stu4(name,class) values('张三',52),('李四',53),('王五',54);
```

非空约束: 不能为空

```
非空约束关键字: not null

非空约束特点: 修饰的字段对应数据不能为空
```

```
# 演示非空约束
# 非空约束关键字: not null
# 非空约束特点: 修饰的字段对应数据不能为空
create table stu5(
    sid int PRIMARY KEY auto_increment ,
    name varchar(100) not null,
    class int not null
);
# 查看表结构
desc stu5;
# 演示非空约束的特点: 限制对应的数据不能为空
insert into stu5(name,class) values(null,52); -- 插入失败,因为有限制
insert into stu5(name,class) values('张三',null); -- 插入失败,因为有限制
insert into stu5(name,class) values('张三',52); -- 插入成功
```

唯一约束: 不能重复

```
唯一约束关键字: unique

唯一约束特点: 修饰的字段对应数据不能重复
```

```
# 演示唯一约束
create table stu7(
    sid int PRIMARY KEY auto_increment ,
    name varchar(100) unique ,
    class int
);
# 查看表结构
desc stu7;
# 演示唯一约束的特点: 限制数据不能重复
insert into stu7(name,class) values('张三',null); -- 第一次插入成功
insert into stu7(name,class) values('张三',null); -- 插入失败,因为有限制
```

默认约束: 设置默认值

```
默认约束关键字: default

默认约束特点: 可以提前为某列设置默认值
```

```
# 5.默认约束
create table stu9(
    sid int PRIMARY KEY auto_increment ,
    name varchar(100),
    class int default 52
);
# 查看表结构
desc stu9;

# 演示默认约束的特点: 插入数据的时候不指定,使用默认值
insert into stu9(name) values('张三');
insert into stu9(name) values('李四');
# -- 注意: 如果你插入数据的时候指定了具体值,以你的为主
insert into stu9(name,class) values('李四',53);
```