# 多表

### 一、多表概述

#### 1. 多表简介

实际开发中，一个项目通常需要很多张表才能完成。例如：一个商城项目就需要分类表(category)、商品表(product)、订单表(orders)等多张表。且这些表的数据之间存在一定的关系，接下来我们将在单表的基础上，一起学习多表方面的知识。

#### 2. 多表创建流程

* 设计表：设计相关的多个表；
* 创建表并设置主外键关系：
  + 方式一：创建表时设置多张表之间的关系；
  + 方式二：创建表之后，再设置表之间的关系

～

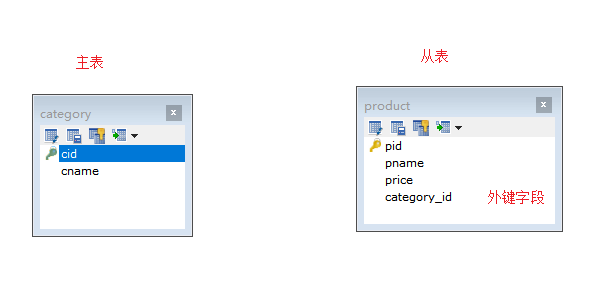
#### 3. 多表关系分类

* 一**对多关系**：
  + 常见实例：客户和订单，分类和商品，部门和员工.
* **多对多关系**：
  + 常见实例：学生和课程、用户和角色
* 一对一关系：(了解)
  + 在实际的开发中应用不多.因为一对一可以创建成一张表.常见如QQ号跟QQ信息详情

### 二、多表入门案例

#### 1. 案例说明

* 本案例以一对多关系为例
* 创建两张表：
  + 主表：分类表category
  + 从表：商品表product



* 设置外键关系

外键： 通常用来描述两张表之间的关系，表达的是一种参照完整性。

如会在员工表中添加一个字段depId，用来表示其所属部门，这个字段与部门表的主键对应。

此时就可以把depId设置为员工表的外键

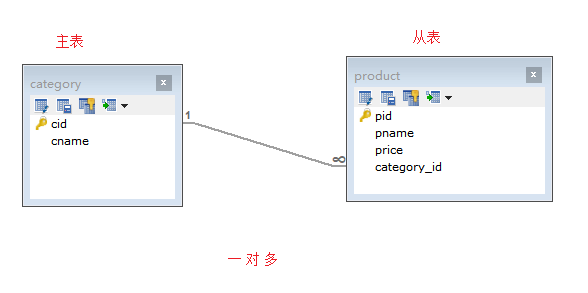
语法1：FOREIGN KEY (自己列名) REFERENCES 主表名 (主表列);

在创建从表的时候 直接在建表语句中添加上述格式

语法2：alter table 从表 add [constraint] [外键名称] foreign key (从表外键字段名) references 主表 (主表的主键);

[外键名称] 用于删除外键约束的，一般建议“\_fk”结尾

alter table 从表 drop foreign key 外键名称



#### 2. 创建分类表category

|  |
| --- |
| ###创建分类表  CREATE TABLE category(  cid INT(32) PRIMARY KEY ,  cname VARCHAR(100) #分类名称  ); |

#### 创建商品表

|  |
| --- |
| # 商品表  CREATE TABLE product (  pid INT PRIMARY KEY ,  pname VARCHAR(40) ,  price DOUBLE ,  category\_id INT # 外键字段  ); |

#### 4. 设置外键

|  |
| --- |
| #添加约束  -- alter table 从表 add [constraint] [外键名称] foreign key (从表外键字段名) references 主表 (主表的主键);  ALTER TABLE product ADD CONSTRAINT product\_fk FOREIGN KEY (category\_id) REFERENCES category (cid); |

添加了外键之后,如果再向商品表中插入数据,要注意数据的categroy\_id字段值要跟category表中的cid值相对应,如果插入的category\_id,在category表中的cid没有对应则无法插入成功.

删除外键:

ALTER TABLE product DROP FOREIGN KEY product\_fk

删除之后,product表 就没有外键约束了

#### 5. 常见操作

#1 向分类表中添加数据

INSERT INTO category (cid ,cname) VALUES(1,'服装');

#2 向商品表添加普通数据,没有外键数据，默认为null

INSERT INTO product (pid,pname) VALUES(1,'商品名称');

#3 向商品表添加普通数据，含有外键信息(数据存放在)

INSERT INTO product (pid ,pname ,category\_id) VALUES(2,'商品名称2', 1);

#4 向商品表添加普通数据，含有外键信息(数据不存在) -- 不能异常

INSERT INTO product (pid ,pname ,category\_id) VALUES(3,'商品名称2',9);

#5 删除指定分类(分类被商品使用) -- 执行异常

DELETE FROM category WHERE cid = 1;

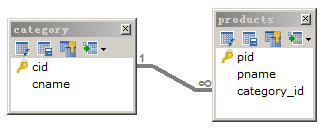
### 三、多表查询

#### 1. 案例说明

本案例以一对多关系的分类和商品表为例，进行常见的几种查询

* 交叉连接查询
  + - 语法：select \* from A,B
* 内连接查询(使用的关键字 inner join -- inner可以省略)
  + 隐式内连接：select \* from A,B where 条件;
  + 显示内连接：select \* from A inner join B on 条件;
* 外连接查询(使用的关键字 outer join -- outer可以省略)
  + 左外连接：left outer join
    - select \* from A left outer join B on 条件;
  + 右外连接：right outer join
    - select \* from A right outer join B on 条件;

#### 数据准备



CREATE TABLE category (

cid int PRIMARY KEY ,

cname VARCHAR(50)

);

CREATE TABLE product(

pid int PRIMARY KEY ,

pname VARCHAR(50),

price INT,

flag VARCHAR(2), #是否上架标记为：1表示上架、0表示下架

category\_id int,

CONSTRAINT product\_fk FOREIGN KEY (category\_id) REFERENCES category (cid)

);

#分类

INSERT INTO category(cid,cname) VALUES('1','家电');

INSERT INTO category(cid,cname) VALUES('2','服饰');

INSERT INTO category(cid,cname) VALUES('3','化妆品');

#商品

INSERT INTO product(pid, pname,price,flag,category\_id) VALUES('1','联想',5000,'1',1);

INSERT INTO product(pid, pname,price,flag,category\_id) VALUES('2','海尔',3000,'1',1);

INSERT INTO product(pid, pname,price,flag,category\_id) VALUES('3','雷神',5000,'1',1);

INSERT INTO product (pid, pname,price,flag,category\_id) VALUES('4','JACK JONES',800,'1',2);

INSERT INTO product (pid, pname,price,flag,category\_id) VALUES('5','真维斯',200,'1',2);

INSERT INTO product (pid, pname,price,flag,category\_id) VALUES('6','花花公子',440,'1',2);

INSERT INTO product (pid, pname,price,flag,category\_id) VALUES('7','劲霸',2000,'1',2);

INSERT INTO product (pid, pname,price,flag,category\_id) VALUES('8','香奈儿',800,'1',3);

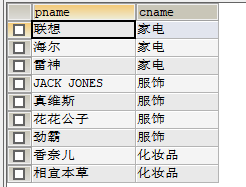
INSERT INTO product (pid, pname,price,flag,category\_id) VALUES('9','相宜本草',200,'1',3);

#### 基本查询

* #需求1. 显示所有商品的名字和其对应的分类

# 格式： SELECT 列名1,列名2 FROM 表1 ,表2 WHERE 表1.外键 = 表2.主键

期望效果：



语句实现：

# 格式： SELECT 列名1,列名2 FROM 表1 ,表2 WHERE 表1.外键 = 表2.主键

SELECT pname,cname FROM product ,category WHERE product.category\_id = category.cid

# 等效写法 ： 这种写法叫 显示 内连接 ， 上面的写法叫隐式内连接

SELECT pname,cname FROM product INNER JOIN category ON product.category\_id = category.cid

* #需求2.显示所有的家电的对应商品名字

格式：# 格式： SELECT 列名1,列名2 FROM 表1 ,表2 WHERE 表1.外键 = 表2.主键 and 条件2

期望效果：



实现：

SELECT cname,pname FROM product ,category WHERE product.category\_id = category.cid AND cname='家电'

内连接



#### 左外连接: left outer join

**左外链接列出左边关系中所有的元组**

**显示结果通常是 左表全部+右表有关联关系的部分，未关联部分 null占位**

语法： select \* from A left outer join B on 条件;

数据准备：

向 分类表中添加两条数据

INSERT INTO category (cid,cname) VALUES(4,'蔬菜'), (5,'水果')

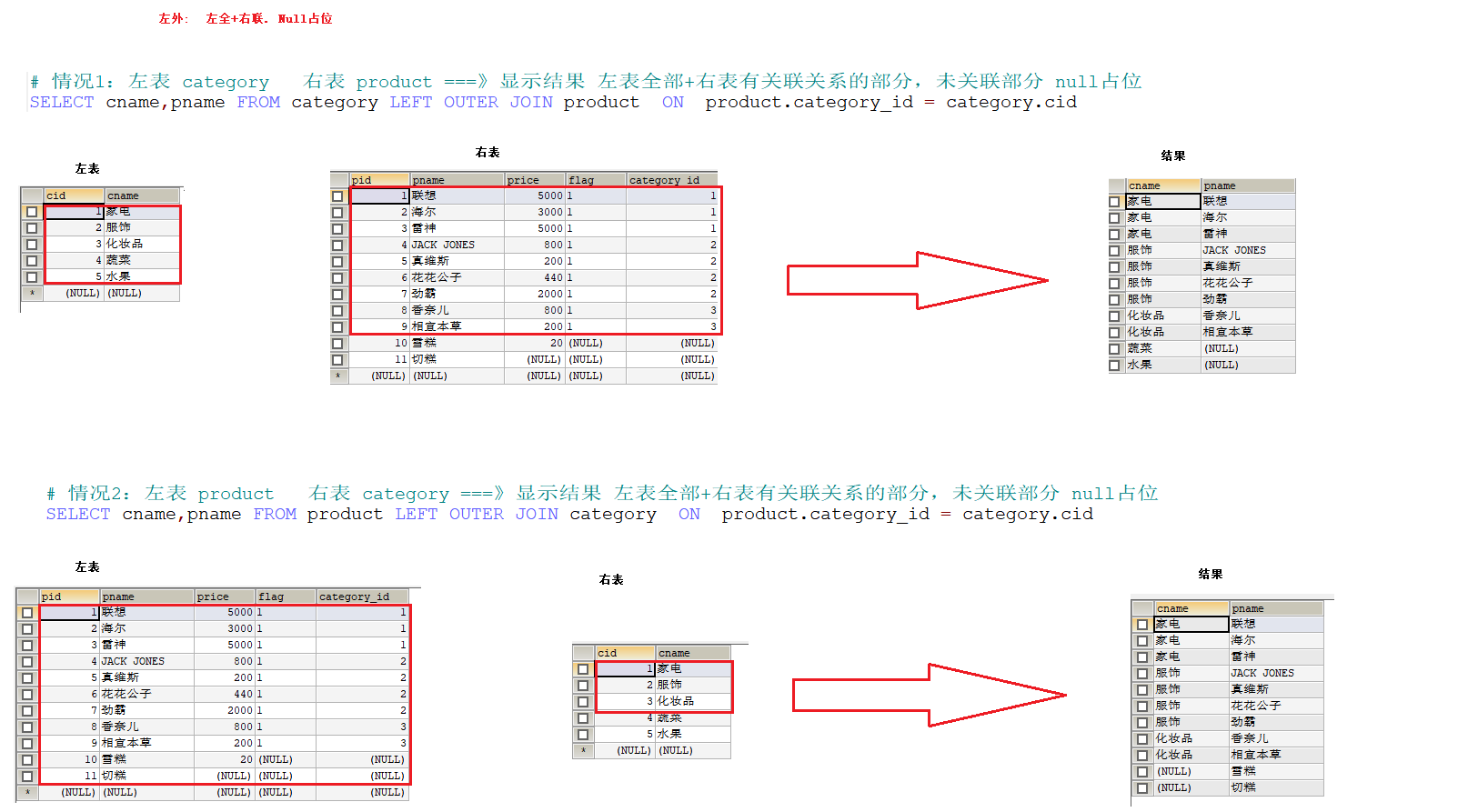
向 商品表中添加两条数据

INSERT INTO product (pid,pname) VALUE(10,'雪糕'),(11,'切糕')

需求： # 左外 连接 显示 分类名和商品名

情况1：左表 category 右表 product名

情况2：左表 product 右表 category



#### 右外连接：right outer join

**右外链接列出右边关系中所有的元组**

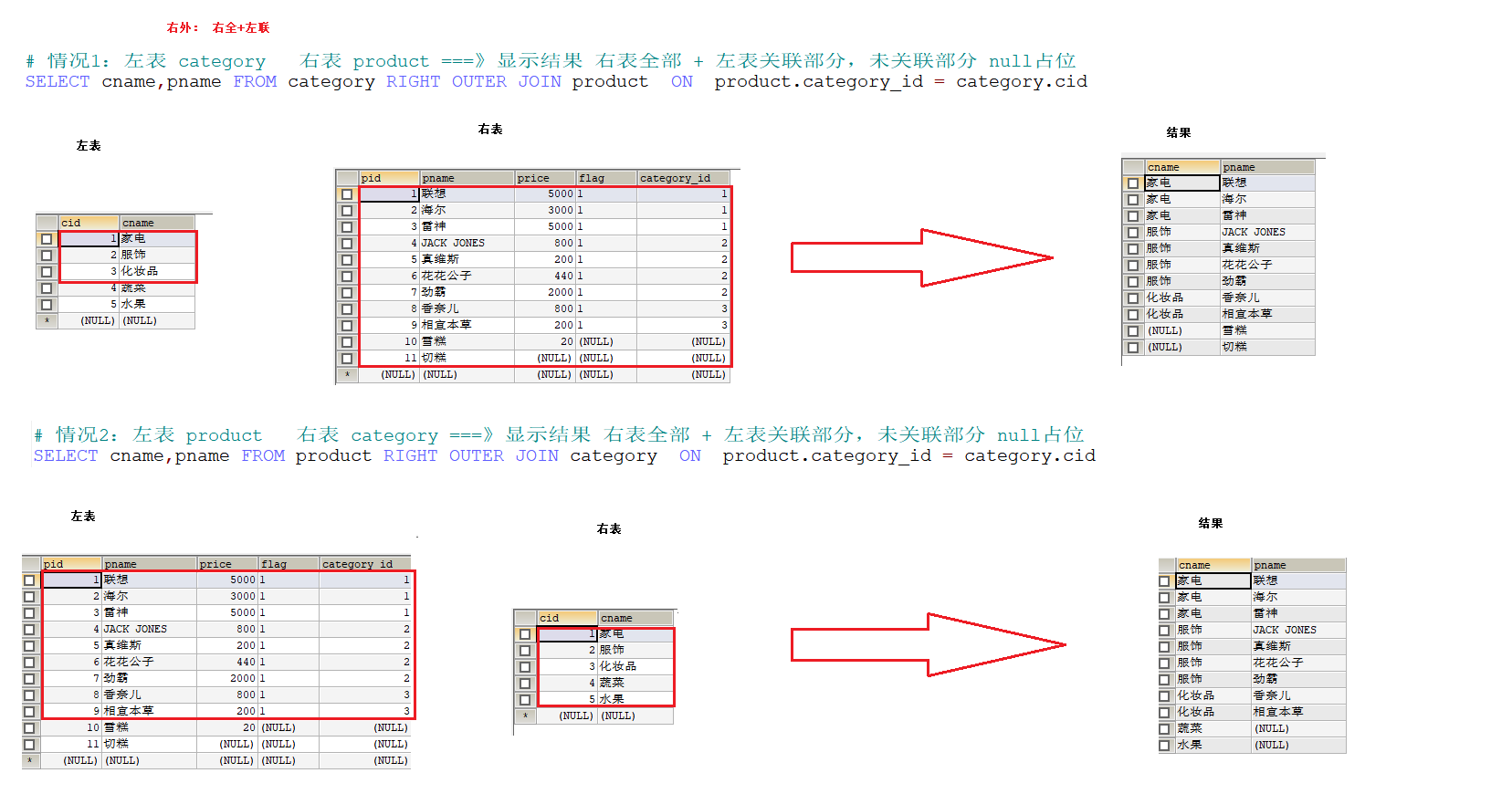
**显示结果通常是 右表全部+左表有关联关系的部分，未关联部分 null占位**

语法：select \* from A right outer join B on 条件;

需求： # 左外 连接 显示 分类名和商品名

情况1：左表 category 右表 product名

情况2：左表 product 右表 category



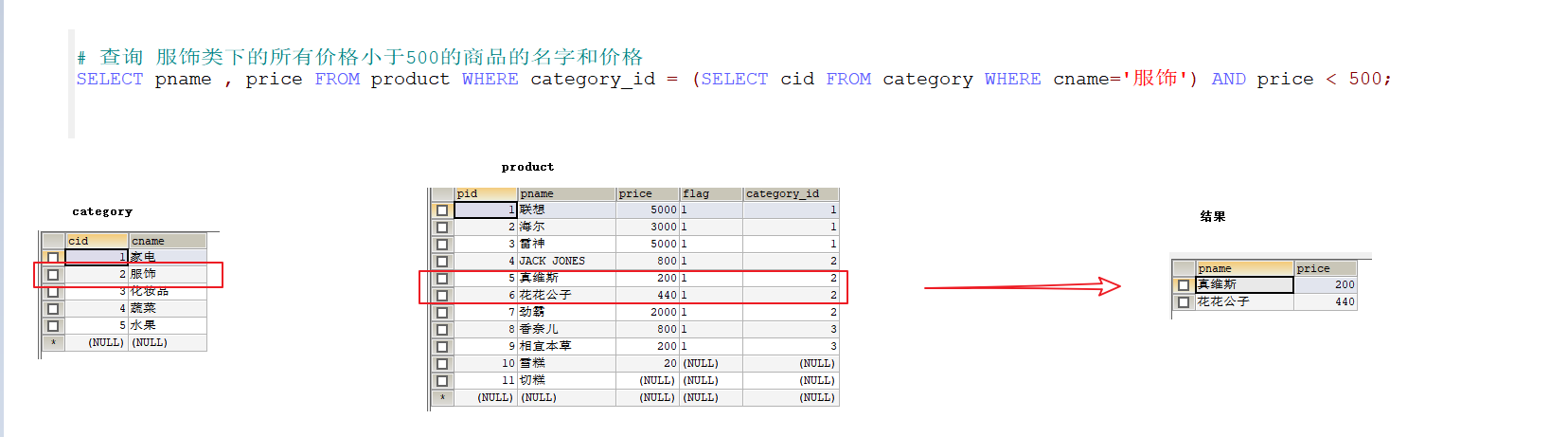
#### 子查询

要得到的结果集是以另外的结果集为基础得到的。 作用到语法上 则为

一条select语句结果作为另一条select语法一部分（查询条件，查询结果，表等）。

# 查询 服饰类下的所有价格小于500的商品的名字和价格

SELECT pname , price FROM product WHERE category\_id = (SELECT cid FROM category WHERE cname='服饰') AND price < 500;

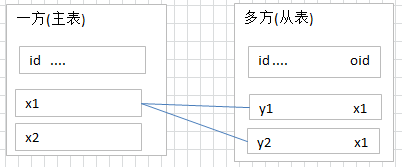


### 理论补充：表之间的关系

### 一对多关系

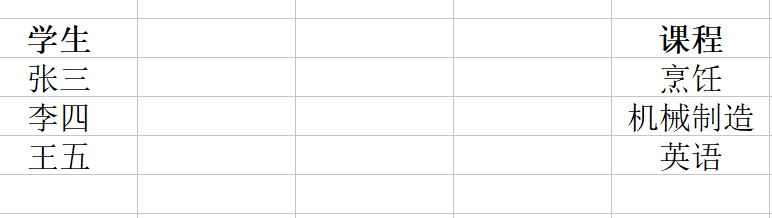
* + 常见实例：客户和订单，分类和商品，部门和员工.
  + 一对多建表原则：在从表(多方)创建一个字段，字段作为外键指向主表(一方)的主键.

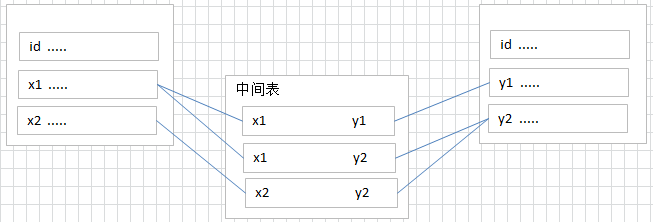




### 多对多关系：

* + 常见实例：学生和课程、用户和角色
  + 多对多关系建表原则：需要创建第三张表,中间表中至少两个字段，这两个字段分别作为外键指向各自一方的主键.





### 一对一关系：(了解)

* + 在实际的开发中应用不多.因为一对一可以创建成一张表.
  + 两种建表原则：
    - 外键唯一：主表的主键和从表的外键（唯一），形成主外键关系，外键唯一unique。
    - 外键是主键：主表的主键和从表的主键，形成
    - 主外键关系。

