day04_多表查询

1.多表查询[重点]

准备数据

> 请创建一个home_work数据库,准备以下数据

```
# 准备数据
# 创建hero表
CREATE TABLE hero
(
    hid       INT PRIMARY KEY,
    hname     VARCHAR(255),
    kongfu_id INT
);

# 创建kongfu表
CREATE TABLE kongfu
(
    kid   INT PRIMARY KEY,
    kname VARCHAR(255)
);
# 插入hero数据
INSERT INTO hero VALUES(1, '鸠摩智', 9),(3, '乔峰', 1),(4, '虚竹', 4),(5, '段誉',
12);
# 插入kongfu数据
INSERT INTO kongfu VALUES(1, '降龙十八掌'),(2, '乾坤大挪移'),(3, '猴子偷桃'),(4, '天山
折梅手');
```

内连接练习

```
# 演示内连接查询
# 需求: 查询有对应功夫的英雄,要求展示英雄名和他的功夫
# 方式1:显式内连接格式: select 字段名 from 左表 inner join 右表 on 左右表关联条件;


# 方式2:隐式内连接格式: select 字段名 from 左表,右表 where 左右表关联条件;
```

外连接练习

```
# 需求: 查询所有英雄对应的功夫,注意:没有对应功夫的用null补全
# 分析:  如果使用左外连接,左表: 英雄表



# 需求: 查询所有功夫,并且只展示有对应功夫的英雄
# 分析:  如果使用左外连接,左表: 功夫表
```

内外连接实战

准备工作

```
# 内外连接查询实战练习
use day04;

# 由于之前已经建过category和products,并且加了外界约束,需要删除重新建表并插入新数据
# 先删除从表
drop table if exists products;
# 再删除主表
drop table if exists category;


# 准备实战数据
CREATE TABLE category (
  cid VARCHAR(32) PRIMARY KEY ,
  cname VARCHAR(50)
);
CREATE TABLE products(
  pid VARCHAR(32) 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 VARCHAR(32),
  CONSTRAINT products_fk_test FOREIGN KEY (category_id) REFERENCES category
(cid)
);

# 分类表插入数据
INSERT INTO category(cid,cname) VALUES('c001','家电');
INSERT INTO category(cid,cname) VALUES('c002','服饰');
INSERT INTO category(cid,cname) VALUES('c003','化妆品');
INSERT INTO category(cid,cname) VALUES('c004','奢侈品');

#商品表插入数据
INSERT INTO products(pid, pname,price,flag,category_id) VALUES('p001','联
想',5000,'1','c001');
INSERT INTO products(pid, pname,price,flag,category_id) VALUES('p002','海
尔',3000,'1','c001');
INSERT INTO products(pid, pname,price,flag,category_id) VALUES('p003','雷
神',5000,'1','c001');
INSERT INTO products (pid, pname,price,flag,category_id) VALUES('p004','JACK
JONES',800,'1','c002');
INSERT INTO products (pid, pname,price,flag,category_id) VALUES('p005','真维
斯',200,'1','c002');
INSERT INTO products (pid, pname,price,flag,category_id) VALUES('p006','花花公
子',440,'1','c002');
INSERT INTO products (pid, pname,price,flag,category_id) VALUES('p007','劲
霸',2000,'1','c002');
INSERT INTO products (pid, pname,price,flag,category_id) VALUES('p008','香奈
儿',800,'1','c003');
INSERT INTO products (pid, pname,price,flag,category_id) VALUES('p009','相宜本
草',200,'1','c003');
```

实战需求

需求1 : 查询哪些分类的商品已经上架， 要求:只展示商品分类名称且要去重

需求2: 查询每个分类下商品的个数， 要求:展示商品分类名称和对应个数

需求3: 查询所有分类下商品的个数， 要求:展示各商品分类名称和对应个数 ==即使分类下没有商品也要展示0==

==注意: count(*)不会忽略null值,count(字段名)会忽略null值==

实战练习

```
# 需求1 : 查询哪些分类的商品已经上架， 要求:只展示商品分类名称
# 分析: 先需要两表连接,再判断哪些商品已经上架,最后确定展示的结果


# 需求2: 查询各个分类的商品个数, 要求:展示各商品分类名称和对应个数
# 分析: 先需求两表连接,再根据分类名称分组,最后统计个数


# 需求3: 查询各个分类的商品个数, 要求:展示各商品分类名称和对应个数
# 要求:所有分类都要展示出来(即使分类下没有商品也要展示0)
```

子查询练习

```
# 需求4: 查询化妆品分类下的所有商品详情
# 分析: 先在分类表中查询化妆品分类id,再根据这个分类id去商品表中找到对应所有商品


# 需求5: 查询化妆品和家电两个分类的所有商品详情
# 分析: 先查询化妆品和家电两个分类的id,再根据分类id去商品表中查询对应的商品
```

自连接练习

准备数据

> 执行资料中的areas.sql脚本文件,创建areas表并插入数据

练习

```
#  演示自连接查询
# 需求1: 查询湖南省所有的城市
# 分析: 省市区三级都在一个表中,那么就可以使用自连接
# 子查询方式: 先查询湖南省的id,再根据湖南省id查询下面所有的城市


# 自连接方式: 给areas表起两个别名分别当成省表和市表使用




# 需求2: 查询株洲市下面所有的区县
```

```
# 子查询方式：先查询株洲的id,再根据株洲id查询下面所有的区县


# 自连接方式：给areas表起两个别名分别当成市表和区县表使用
```

题目（选做题，提交不需要这题）

数据准备

```
-- 创建数据库
create database db_hw4;
use db_hw4;

-- 创建user表
CREATE TABLE USER(
    id INT PRIMARY KEY AUTO_INCREMENT,  -- 用户id
    NAME VARCHAR(20),           -- 用户姓名
    age INT                                 -- 用户年龄
);
-- 添加数据
INSERT INTO USER VALUES (1,'张三',23);
INSERT INTO USER VALUES (2,'李四',24);
INSERT INTO USER VALUES (3,'王五',25);
INSERT INTO USER VALUES (4,'赵六',26);


-- 订单表
CREATE TABLE orderlist(
    id INT PRIMARY KEY AUTO_INCREMENT,  -- 订单id
    number VARCHAR(30),         -- 订单编号
    uid INT,    -- 外键字段
    CONSTRAINT ou_fk1 FOREIGN KEY (uid) REFERENCES USER(id)
);
-- 添加数据
INSERT INTO orderlist VALUES (1,'hm001',1);
INSERT INTO orderlist VALUES (2,'hm002',1);
INSERT INTO orderlist VALUES (3,'hm003',2);
INSERT INTO orderlist VALUES (4,'hm004',2);
INSERT INTO orderlist VALUES (5,'hm005',3);
INSERT INTO orderlist VALUES (6,'hm006',3);
INSERT INTO orderlist VALUES (7,'hm007',NULL);


-- 商品分类表
CREATE TABLE category(
    id INT PRIMARY KEY AUTO_INCREMENT,  -- 商品分类id
    NAME VARCHAR(10)                    -- 商品分类名称
);
-- 添加数据
INSERT INTO category VALUES (1,'手机数码');
INSERT INTO category VALUES (2,'电脑办公');
INSERT INTO category VALUES (3,'烟酒茶糖');
INSERT INTO category VALUES (4,'鞋靴箱包');
```

```sql
-- 商品表
CREATE TABLE product(
    id INT PRIMARY KEY AUTO_INCREMENT,   -- 商品id
    NAME VARCHAR(30),                    -- 商品名称
    cid INT, -- 外键字段
    CONSTRAINT cp_fk1 FOREIGN KEY (cid) REFERENCES category(id)
);
-- 添加数据
INSERT INTO product VALUES (1,'华为手机',1);
INSERT INTO product VALUES (2,'小米手机',1);
INSERT INTO product VALUES (3,'联想电脑',2);
INSERT INTO product VALUES (4,'苹果电脑',2);
INSERT INTO product VALUES (5,'中华香烟',3);
INSERT INTO product VALUES (6,'玉溪香烟',3);
INSERT INTO product VALUES (7,'计生用品',NULL);


-- 中间表
CREATE TABLE us_pro(
    upid INT PRIMARY KEY AUTO_INCREMENT,  -- 中间表id
    uid INT, -- 外键字段。需要和用户表的主键产生关联
    pid INT, -- 外键字段。需要和商品表的主键产生关联
    CONSTRAINT up_fk1 FOREIGN KEY (uid) REFERENCES USER(id),
    CONSTRAINT up_fk2 FOREIGN KEY (pid) REFERENCES product(id)
);
-- 添加数据
INSERT INTO us_pro VALUES (NULL,1,1);
INSERT INTO us_pro VALUES (NULL,1,2);
INSERT INTO us_pro VALUES (NULL,1,3);
INSERT INTO us_pro VALUES (NULL,1,4);
INSERT INTO us_pro VALUES (NULL,1,5);
INSERT INTO us_pro VALUES (NULL,1,6);
INSERT INTO us_pro VALUES (NULL,1,7);
INSERT INTO us_pro VALUES (NULL,2,1);
INSERT INTO us_pro VALUES (NULL,2,2);
INSERT INTO us_pro VALUES (NULL,2,3);
INSERT INTO us_pro VALUES (NULL,2,4);
INSERT INTO us_pro VALUES (NULL,2,5);
INSERT INTO us_pro VALUES (NULL,2,6);
INSERT INTO us_pro VALUES (NULL,2,7);
INSERT INTO us_pro VALUES (NULL,3,1);
INSERT INTO us_pro VALUES (NULL,3,2);
INSERT INTO us_pro VALUES (NULL,3,3);
INSERT INTO us_pro VALUES (NULL,3,4);
INSERT INTO us_pro VALUES (NULL,3,5);
INSERT INTO us_pro VALUES (NULL,3,6);
INSERT INTO us_pro VALUES (NULL,3,7);
INSERT INTO us_pro VALUES (NULL,4,1);
INSERT INTO us_pro VALUES (NULL,4,2);
INSERT INTO us_pro VALUES (NULL,4,3);
INSERT INTO us_pro VALUES (NULL,4,4);
INSERT INTO us_pro VALUES (NULL,4,5);
INSERT INTO us_pro VALUES (NULL,4,6);
INSERT INTO us_pro VALUES (NULL,4,7);
```

练习题目

> -- 1.查询用户的编号、姓名、年龄,订单编号,注意:查询有订单的用户
> -- 2.用户的编号、姓名、年龄,订单编号.注意: 查询所有的用户包括没有订单的
> -- 3.查询所有的订单。用户的编号、姓名、年龄。订单编号
> -- 4.查询用户年龄大于23岁的信息。显示用户的编号、姓名、年龄。订单编号
> -- 5.查询张三和李四用户的信息。显示用户的编号、姓名、年龄。订单编号

```
-- 1.查询用户的编号、姓名、年龄,订单编号,注意:查询有订单的用户
select user.id, name, age, number
from user
         join orderlist on user.id = orderlist.uid;

-- 2.用户的编号、姓名、年龄,订单编号.注意: 查询所有的用户包括没有订单的
select user.id, name, age, number
from user
         left join orderlist on user.id = orderlist.uid;

-- 3.查询所有的订单。用户的编号、姓名、年龄。订单编号
select user.id, name, age, number
from orderlist
         left join user on user.id = orderlist.uid;

-- 4.查询用户年龄大于23岁的信息。显示用户的编号、姓名、年龄。订单编号
select user.id, name, age, number
from user
         left join orderlist on user.id = orderlist.uid
where user.age > 23;
-- mysql调优: 实际开发建议join之前提前过滤
select u.id, name, age, number
from (select * from user where user.age > 23) u
         left join orderlist on u.id = orderlist.uid;

-- 5.查询张三和李四用户的信息。显示用户的编号、姓名、年龄。订单编号
select user.id, name, age, number
from user
         join orderlist on user.id = orderlist.uid
where user.name in ('张三', '李四');
-- mysql调优: 实际开发建议join之前提前过滤
select u.id, name, age, number
from (select * from user where user.name in ('张三', '李四')) u
         join orderlist on u.id = orderlist.uid;
```