数据库——第三章

陈奕培 22920202202879

1. 试述SQL的特点。

（1）综合统一：SQL语言集数据定义语言、数据操纵语言、数据控制语言的功能于一体。在关系模型中实体和实体集之间的联系均用关系表示，这种数据结构的单一性带来了操作符的统一，客服了非关系系统由于信息表示方式的多样性带来的操作复杂性。

（2）高度非过程化：用SQL语言进行数据操作，只需提出“做什么“，不必在乎“怎么做”，无需了解存放路径。存放路径的选择和SQL的操作过程由系统自动完成。提高数据独立性。

（3）面向集合的操作方式：非关系数据模型的操作对象是一条数据，SQL采用集合操作方式，操作对象和查找结果且插入、删除、更新操作的对象都可以是元组的集合。

（4）以同一种语法结构提供两种使用方式：SQL不仅是独立的语言，还是嵌入式语言，可以在终端键盘上直接输入SQL命令，也可以通过高级语言使用，且语法结构基本一致。

（5）语言简洁，易学易用。

1. 说明在DROP TABLE时，RESTRICT和CASCADE的区别。

若选择RESTRICT，则删除该表时有限制条件；

若选择CASCADE，则删除时没有限制条件；

默认是RESTRICT。

1. 写出下列查询等价的SQL表达式：
2. SELECT \* FROM S WHERE A=10;
3. SELECT A,B FREOM S;
4. SELECT \* FROM S INNER JOIN T ON S.C=T.C AND S.D=T.D;
5. SELECT \* FROM S INNER JOIN T ON S.C=T.C;
6. SELECT \* FROM S INNER JOIN T ON S.A<T.E;
7. SELECT \* FROM S,T WHERE S.C=T.C AND T.D=S.D;
8. 实现四个表，并建立查询。

注：使用MySQL实现，版本8.0

建表：

CREATE TABLE S(

SNO CHAR(2) UNIQUE,

SNAME VARCHAR(5),

STATUS CHAR(2),

CITY VARCHAR(5)

);

CREATE TABLE P(

PNO CHAR(2) UNIQUE,

PNAME VARCHAR(5),

COLOR VARCHAR(2),

WEIGHT SMALLINT

);

CREATE TABLE J(

JNO CHAR(2) UNIQUE,

JNAME VARCHAR(5),

CITY VARCHAR(5)

);

CREATE TABLE SPJ(

SNO CHAR(2),

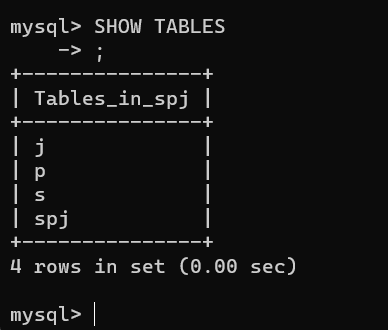
PNO CHAR(2),

JNO CHAR(2),

QTY SMALLINT

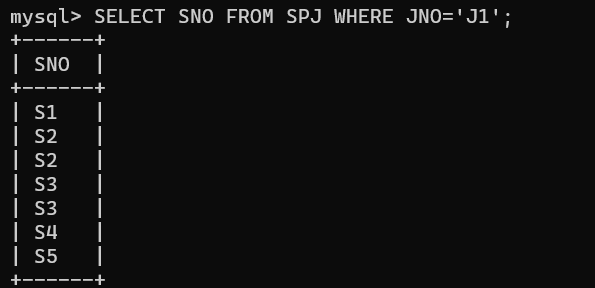
);

结果，省略数据的插入

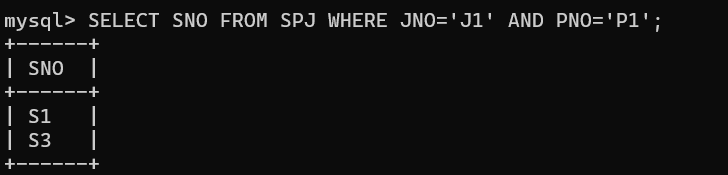


查询：

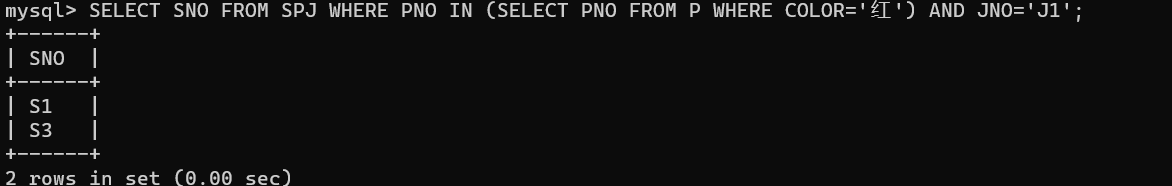
(1)SELECT SNO FROM SPJ WHERE JNO=’J1’;



(2)SELECT SNO FROM SPJ WHERE JNO='J1' AND PNO='P1';



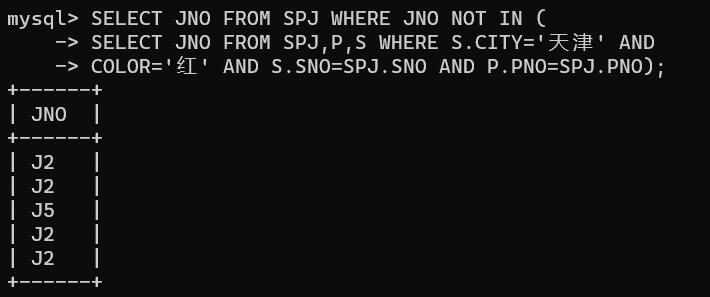
(3)SELECT SNO FROM SPJ WHERE PNO IN (SELECT PNO FROM P WHERE COLOR='红') AND JNO='J1';



(4) SELECT JNO FROM SPJ WHERE JNO NOT IN (

-> SELECT JNO FROM SPJ,P,S WHERE S.CITY='天津' AND

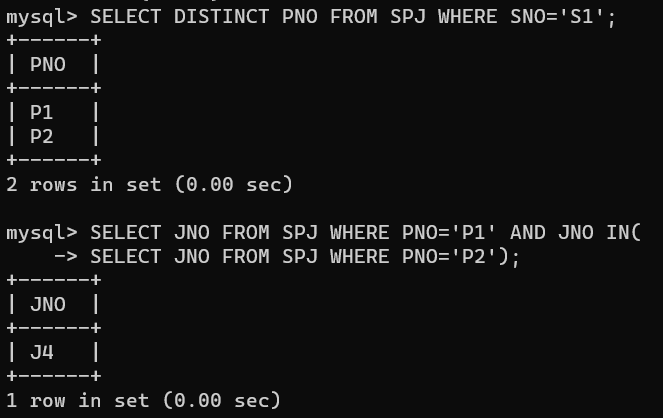
-> COLOR='红' AND S.SNO=SPJ.SNO AND P.PNO=SPJ.PNO);



(5) SELECT DISTINCT PNO FROM SPJ WHERE SNO='S1';

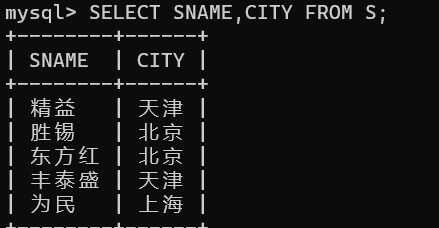
SELECT JNO FROM SPJ WHERE PNO='P1' AND JNO IN(

-> SELECT JNO FROM SPJ WHERE PNO='P2');

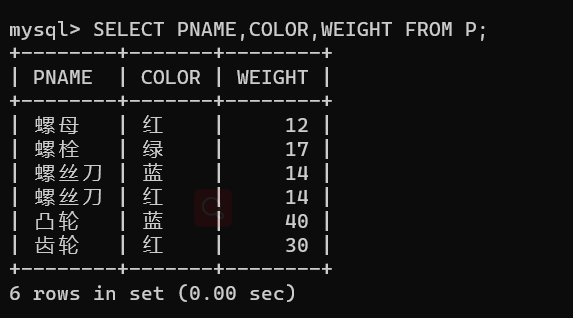


1. 完成以下操作

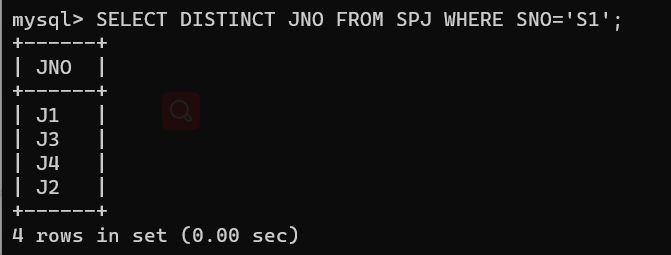
（1）



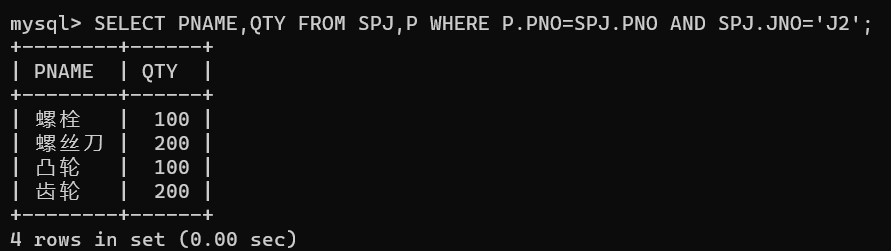
(2)



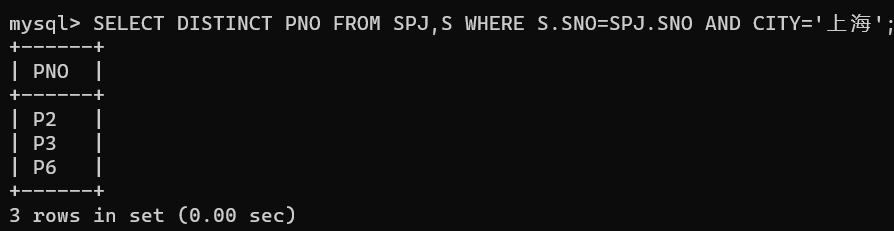
(3)



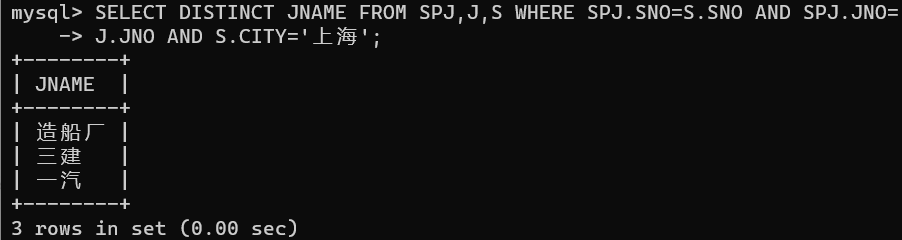
(4)



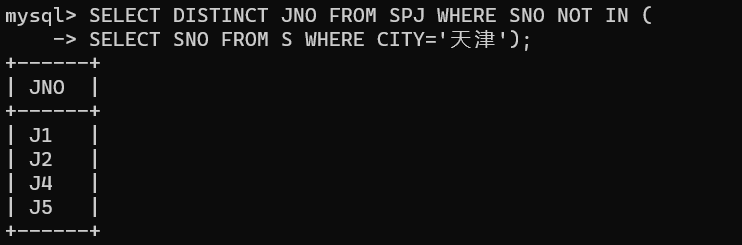
(5)



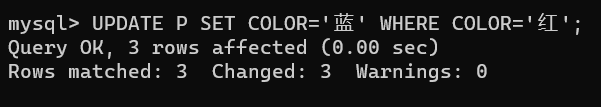
(6)

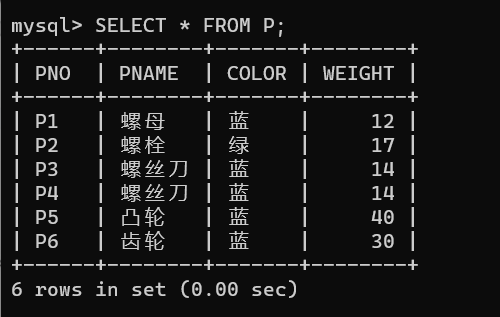


(7)

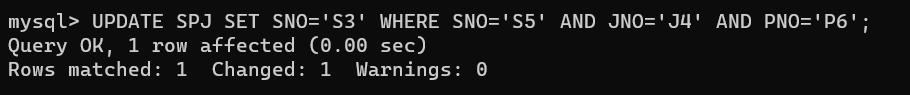


(8)

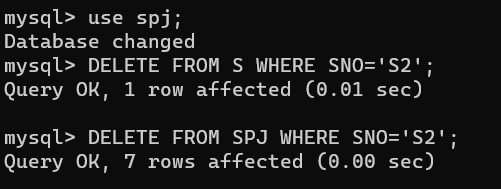




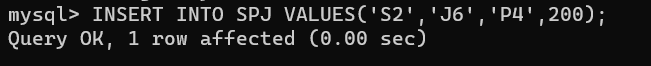
(9)



(10)



(11)



1. 什么是基本表？什么是视图？两者之间的区别和联系是什么？
2. 基本表是本身独立存在的表，在SQL中，一个关系就对应一个表。
3. 视图是从一个或者多个基本表中导出的虚表，不独立存储在数据库当中。
4. 区别和联系

区别：视图是虚表，不独立存储在数据库中，数据库只存放视图的定义，数据存放在基本表当中。

联系：视图在概念上与基本表相同，用户可以如同使用表一样使用视图，可以在视图上再定义视图。视图依据基本表产生，会随着基本的数据变化而变化。

1. 试述视图的优点

（1）视图能够简化用户的操作：视图机制使用户注意力集中在数据上，通过定义视图，可以使数据库看起来结构简单、清晰，并且简化用户的数据查询操作；

（2）视图使用户能够以多种角度看待同一数据：视图机制使不同的用户以不同的角度看待同一数据，当多个用户使用同一数据库时，这种机制十分重要；

（3）视图对重构数据库提供了一定程度的独立性：当数据库重构时，用户的应用程序不受到影响；

（4）视图对机密数据提供了安全保护：有了视图机制，在设计数据库应用系统的时候，对不同的用户定义不同的视图，使机密数据收到保护；

（5）使用视图可以更清晰的表达查询。

1. 哪些视图的可以更新的，哪些不可以，举例说明。

CREATE TABLE TEST (

A VARCHAR(2),

B INT,

C INT);

CREATE VIEW V\_A

AS

SELECT A,B FROM TEST;

CREATE VIEW V\_B

AS

SELECT A,AVG(C) FROM TEST GROUP BY A;

（1）基本表的行列子集视图是可以更新的，如V\_A

INSERT INTO V\_A VALUE(‘A1’,1);

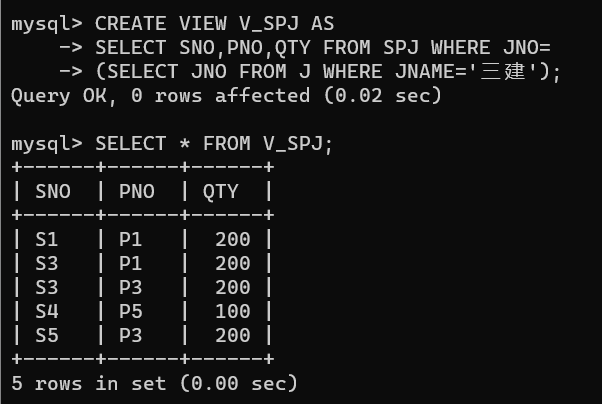
将会转变为 对TEST表的更新

（2）对于视图的属性来自集函数、表达式的视图不可更新，如V\_B

INSERT INTO V\_B VALUE(‘A2’,1);

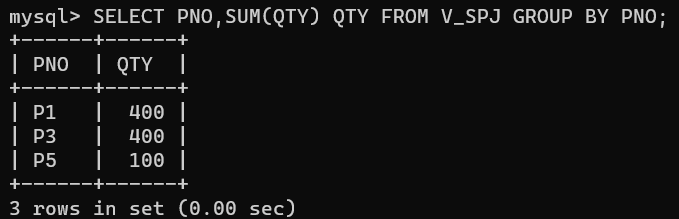
将出错

1. 设计视图



查询：

(1)



(2)

