

一、在学生管理数据库中，有如下三个表

学生信息表：S (S#, SNAME, AGE, SEX) ,

课程表：C(C#, CNAME, TEACHER)

选课表：SC(S#, C#, GRADE)。

请使用 SQL 回答下列问题：

- 1) 检索学号为 003 号同学所学课程的课程名和教师名
- 2) 检索选择程军老师所授课程的男同学姓名
- 3) 检索刘丽同学不学的课程的课程名
- 4) 检索平均成绩在 60 分以下的学生的学号和姓名
- 5) 检索至少选修 3 门课程的学生的学号,按照从小到大排列。

二、涉及二战中的大型舰船。它由以下几个关系组成（红色标记主键，蓝色标记外键）：

Classes(class, type, country, numGuns, bore, displacement)

Ships(name, class, launched)

Battles(name, date)

Outcomes(ship, battle, result)

舰船类表 Classes 由舰船类名 (class)、型号 (type, 其中 bb 代表主力舰, bc 代表巡洋舰)、生产国家 (country)、火炮的门数 (numGuns)、火炮口径 (bore, 单位是英寸) 和排水量(displacement, 单位是吨)组成。

舰船表 Ships 由舰船名 (name)、舰船类名 (class)、开始服役的日期 (launched) 组成。

战役表 Battles 由战役名 (name) 和舰船参加战役的时间 (date) 组成。

参战结果表 Outcomes 由舰船名 (ship)、参加的战役名 (battle) 和各个舰船在各场战役中的结果 (result, 分为沉没、受伤或完好) 组成。

class	type	country	numGuns	bore	displacement
Bismarck	bb	Germany	8	15	42000
Iowa	bb	USA	9	16	46000
Kongo	bc	Japan	8	14	32000
North Carolina	bb	USA	9	16	37000
Renown	bc	Gt.Britain	6	15	32000
Revenge	bb	Gt.Britain	9	15	29000
Tennessee	bb	USA	12	14	32000
Yamato	bb	Japan	9	18	65000

(a) 关系 Classes 的数据取样

name	class	launched
California	Tennessee	1921
Karuna	Kongo	1915
Hiei	Kongo	1914
Iowa	Iowa	1943
Kirishima	Kongo	1915
Kongo	Kongo	1913
Missouri	Iowa	1944
Musashi	Yamato	1942
New Jersey	Iowa	1943
North Carolina	North Carolina	1941

Ramillies	Revenge	1917
Renown	Renown	1916
Repulse	Renown	1916
Resolution	Revenge	1916
Revenge	Revenge	1916
Royal Oak	Revenge	1916
Royal Sovereign	Revenge	1916
Tennessee	Tennessee	1920
Washington	North Carolina	1941
Wisconsin	Iowa	1944
Yamato	Yamato	1941

(b) 关系 Ships 的数据取样

name	date
North Atlantic	5/24-27/41
Guadalcanal	11/15/42
North Cape	12/26/43
Surigao Strait	10/25/44

(c) 关系 Battles 的数据取样

ship	battle	result
Bismarck	North Atlantic	sunk
California	Surigao Strait	ok
Duke of York	North Cape	ok
Fuso	Surigao Strait	sunk
Hood	North Atlantic	sunk
King George V	North Atlantic	ok
Kirishima	Guadalcanal	sunk
Prince of Wales	North Atlantic	damaged
Rodney	North Atlantic	ok
Scharnhorst	North Cape	sunk
South Dakota	Guadalcanal	damaged
Tennessee	Surigao Strait	ok
Washington	Guadalcanal	ok
West Virginia	Surigao Strait	ok
Yamashiro	Surigao Strait	sunk

(d) 关系 Outcomes 的数据取样

按要求写出下面的 sql 语句：

- 1) 在数据库中创建 classes 和 ships 这两个关系，并说明在插入两者的数据时，需要注意什么。
- 2) a. 两艘 Nelson 类型的英国主力舰 Nelson 和 Rodney 在 1927 年下水。两者都具有 16 英寸口径的火炮 9 门，排水量为 34000 吨。把这条信息加入到数据库中。
  - b. 删除所有在战役中沉没的船只。
  - c. 更新 Classes 关系使得火炮口径使用厘米作为单位（1 英寸=2.5 厘米），排水量使用公制吨（1 公制吨=1.1 吨）。

- 3) 定义一个视图，它包括所有英国舰船的类型 (class)、类别 (type)、火炮数量、口径、排水量和下水年份。
- 4) 查询 Kongo 类型舰船参加的战役名称。
- 5) 查询所有主力舰的平均火炮数量。
- 6) 查询每一类型的第一艘船下水的年份。
- 7) 列出至少有 3 艘舰船的类型中，该类型名和被击沉的舰船数目。
- 8) 列出所有数据库中提到的舰船名字。
- 9) 列出那些既有主力舰又有巡洋舰的国家。
- 10) 找出拥有火炮数量最多的船只所属的国家。
- 11) 找出至少有一艘船在战役中被击没的舰船类型。

三、设一个影片出租公司的数据库由以下几个关系组成：

Movies(MovieID, MovieName)  
Suppliers(SupplierID, SupplierName)  
Customers(CustID, LastName, FirstName)  
MovieSupplier(SupplierID, MovieID, Price)  
Orders(OrderID, SupplierID, MovieID, Copies)  
Inventory(TapeID, MovieID)  
Rentals(CustomerID, TapeID, CkoutDate, Duration)

影片表 Movies 由影片号 (MovieID)、影片名 (MovieName) 组成。

供货商表 Suppliers 由供货商号 (SupplierID)、供货商名 (SupplierName) 组成。

顾客表 Customers 由顾客号 (CustID)、顾客姓氏 (LastName) 和顾客名 (FirstName) 组成。

影片供货表 MovieSupplier 由供货商号 (SupplierID)、影片号 (MovieID) 和售价 (Price) 组成。

订单表 Orders 由订单号 (OrderID)、供货商号 (SupplierID)、影片号 (MovieID) 和拷贝光盘数量 (Copies) 组成。

库存表 Inventory 由光盘号 (TapeID)、影片号 (MovieID) 组成 (注意：同一部影片的多个拷贝光盘具有不同的编号)。

出租表 Rentals 由顾客号 (CustomerID)、光盘号 (TapeID)、出租日期 (CkoutDate)、持续时间 (Duration, 单位：天) 组成。

按要求写出下面的 sql 语句：

- 1) 在数据库中创建 Orders 和 Rentals 这两个关系。
- 2) a. 编号为 5600 的光盘被编号为 9823 的顾客在 2017 年 3 月 26 日租出，出租时长为 30 天。  
将该条信息加入到出租表 Rentals 中。
  - b. 删除出租表 Rentals 中所有出租日期早于 2000 年 1 月 1 日的记录。
  - c. 更新影片供货表 MovieSupplier，使得售价从人民币改为美元作为单位 (1 美元=6.88 人民币)。
- 3) 定义一个视图，它包括所有由名为“Joe's House of Video”的供货商供应的影片的片名和售价。
- 4) 列出库存中每个供货商的名称和该供货商提供的不同影片的总量。
- 5) 检索出租公司预订的拷贝光盘数量大于 5 的影片片名。
- 6) 检索在库存中的拷贝光盘数多于 1 的影片片名。
- 7) 检索出租给顾客的持续时间最长的影片的片名。

- 8) 检索不在库存中的所有影片片名。
- 9) 检索名为"Hacksaw Ridge "的影片给出的售价最低的供货商号及该供货商名。
- 10) 检索租用了名为"Beauty and the Beast"的影片或者租用了由名为"VWS Video"的供货商提供的影片的那些顾客的姓名。

第二次作业 3月20日（下周三）晚10点之前提交到作业邮箱，要求和之前一样。

一、在学生管理数据库中，有如下三个表

学生信息表：S (S#, SNAME, AGE, SEX) ,

课程表：C(C#, CNAME, TEACHER)

选课表：SC(S#, C#, GRADE)。

请使用 SQL 回答下列问题：

- 1) 检索学号为 003 号同学所学课程的课程名和教师名
- 2) 检索选择程军老师所授课程的男同学姓名
- 3) 检索刘丽同学不学的课程的课程名
- 4) 检索平均成绩在 60 分以下的学生的学号和姓名
- 5) 检索至少选修 3 门课程的学生的学号,按照从小到大排列。

1) Select CNAME, TEACHER

From SC, C

Where S# = "003" and SC.C# = C.C#

2) Select SNAME

From C, S, SC

Where TEACHER = "程军" and

SEX = "男" and

S.S# = SC.S# and

C.C# = SC.C#

3) Select CNAME

From C

Where C# not in

(Select C#

From S, SC

Where SNAME = "刘丽" and S.S# = SC.S#)

4) Select S#, SNAME

From S

Where S# in

(Select S#

From SC

Group by S# Having avg(GRADE) < 60)

Select S.S#, SNAME

From S, SC

Where S.S# = SC.S#

Group by S.S#, SNAME

Having avg(GRADE) < 60 ;

5) Select S#

From SC

Group by S# Having Count(\*) >= 3

Order by S# ASC

二、涉及二战中的大型舰船。它由以下几个关系组成（红色标记主键，蓝色标记外键）：

Classes(class, type, country, numGuns, bore, displacement)

Ships(name, class, launched)

Battles(name, date)

Outcomes(ship, battle, result)

1) Create table Classes

( class varchar(20) NOT NULL,  
type char(2),  
country varchar(20),  
numGuns SmallINT,  
bore Float,  
displacement INTEGER,  
Primary key (class),  
check(type in ('bb', 'bc')) )

Create table Ships

( name varchar(20) NOT NULL,  
class varchar(20),  
launched INTEGER,  
Primary key (name),  
Foreign key (class) reference Classes(class))

向两表插入数据时，DBMS会自动检查是否违反完整性约束，所以向classes表插入数据时，注意class属性不能为空值且在该属性上新元组的值不能与其它元组重复；插入ships时，除了上述的实体完整性，还要注意外键class的值必须在主键中出现或取空值。

- 2) a. <sup>①</sup> Insert  
 Into Classes  
 Values ('Nelson', 'bb', 'Gt. Britain', 9, 16, 34000)
- <sup>②</sup> Insert  
 Into Ships  
 Values ('Nelson', 'Nelson', 1927)
- <sup>③</sup> Insert  
 Into Ships  
 Values ('Rodney', 'Nelson', 1927)
- b. Delete  
 From Ships  
 Where name in  
 (Select ship  
 From Outcomes  
 Where result = "沉没")
- C. Update Classes  
 Set bore = bore \* 2.5, displacement = displacement / 1.1  
 Create View Britain\_Ships(class, type, numGuns, bore,  
 displacement, launched)  
 AS  
 Select Classes.class, type, numGuns, bore,  
 displacement, launched  
 From Classes, Ships  
 Where Ships.class = Classes.class  
 Select battle  
 From Outcomes  
 Where ship in  
 (Select name  
 From Ships  
 Where class = "Kongo")
- Select battle  
 From ships, outcomes  
 where class = "Kongo" and  
 ships.name = outcomes.ship

- 5) Select avg(bore) X  
From Classes  
Where type = "bb"
- 6) Select class, min(launched)  
From Ships  
Group by class
- 7) Select class, count(\*)  
From Ships, Outcomes  
Where class in  
| Select class  
From Ships  
Group by class Having count(\*) ≥ 3 and  
name = ship and  
result = "沉没"  
Group by class
- 8) Select name  
From Ships  
Union  
Select Distinct ship name  
From Outcomes
- 9) Select Distinct country  
From Classes  
Where type = "bb"  
Intersect  
Select Distinct country  
From Classes  
Where type = "bc"
- Select avg(numGuns)  
From classes, ships  
where type = "bb" and  
classes.class = ships.class

10) Select country

From Classes

Where bore =

( Select mas(bore)  
From Classes )

11) Select Distinct class

From Ships, Outcomes

Where name = ship and  
result = "沉没"

select class

From ships, outcomes

where result = "沉没" and

ships.name = outcomes.ship

Group by class

Having count(\*) >=

三、设一个影片出租公司的数据库由以下几个关系组成：

Movies(MovieID, MovieName)

Suppliers(SupplierID, SupplierName)

Customers(CustID, LastName, FirstName)

MovieSupplier(SupplierID, MovieID, Price)

Orders(OrderID, SupplierID, MovieID, Copies)

Inventory(TapeID, MovieID) 库存

Rentals(CustomerID, TapeID, CheckOutDate, Duration)

1) Create table Orders

( OrderID char(14) Primary key,  
SupplierID char(4),  
MovieID char(4),  
Copies Int,

Foreign key(SupplierID) Reference Suppliers(SupplierID),

Foreign key(MovieID) Reference Movies(MovieID),

Create table Rentals

( CustomerID char(4),  
TapeID char(4),

CKoutDate Date, / primary key (customerID, tapeID)

Duration Int,           

Foreign key (CustomerID) reference Customers (CustID))

2) a. Insert Foreign key (tapeID) reference — )

Into Rentals

Values (9823, 5600, 2017-03-26, 30)

b. Delete

From Rentals

Where CKoutDate < 2000-01-01

c. Update MovieSupplier

Set Price = Price / 6.88

3) Create View V(MovieName, Price)

AS

Select MovieName, Price

From Movies, Suppliers, MovieSupplier

Where MovieSupplier.SupplierID = Suppliers.SupplierID and

MovieSupplier.MovieID = Movies.MovieID and

SupplierName = "Joe's House of Video"

4) Select SupplierName, count(\*)

From Suppliers, MovieSupplier

Where Suppliers.SupplierID = MovieSupplier.SupplierID

Group by SupplierName

5) 这里考虑电影同名的情况 (同样适用于不同名的情况)

Select MovieName

如果不同名

From Movies

Select MovieName

Where MovieID in

From Movies, orders

| Select MovieID

where Movies.MovieID = orders.MovieID

From Orders

Group by MovieName Having —

Group by MovieID Having —

Sum(Copies) > 5 )

- 6) Select MovieName  
From Movies  
Where MovieID in  
  | Select MovieID  
    From Inventory  
    Group by MovieID Having count(\*) > 1)
- 7) Select MovieName  
From Movies, Inventory, Rentals  
Where Movies.MovieID = Inventory.MovieID and  
Inventory.TapeID = Rentals.TapeID and  
Duration >= ALL  
  | Select Distinct Duration  
    From Rentals)
- 8) Select MovieName  
From Movies  
Where MovieID NOT ~~Exist~~ <sup>in</sup>  
  | Select Distinct MovieID  
    From Inventory)
- 9) Select Suppliers.SupplierID, SupplierName  
From Suppliers, MovieSupplier, Movies  
Where Suppliers.SupplierID = MovieSupplier.SupplierID and  
Movies.MovieID = MovieSupplier.MovieID and  
MovieName = "Hacksaw Ridge" and  
Price <= ALL  
  | Select Distinct Price  
    From Movies, MovieSupplier  
    Where Movies.MovieID = MovieSupplier.MovieID  
      and MovieName = "Hacksaw Ridge")  
  
也可以先找到最低售价, 再找供货商.

10) Select FirstName, LastName  
From Movies, Customers, Inventory, Rentals  
Where Movies.MovieID = Inventory.MovieID and  
Inventory.TapeID = Rentals.TapeID and  
Customers.CUSTID = Rentals.customerID and  
MovieName = "Beauty and the Beast"  
Union  
Select FirstName, LastName  
From Suppliers, MovieSupplier, Inventory, Rentals, customers  
Where Suppliers.SupplierName = "VWS Video" and  
Suppliers.SupplierID = MovieSupplier.SupplierID and  
MovieSupplier.MovieID = Inventory.MovieID and  
Inventory.TapeID = Rentals.TapeID and  
Rentals.CustomerID = Customers.CUSTID