# 题目1

1、学校图书馆借书信息管理系统建立三个表：

学生信息表：student

|  |  |  |
| --- | --- | --- |
| **字段名称** | **数据类型** | **说明** |
| stuID | char(10) | 学生编号，主键 |
| stuName | Varchar(10) | 学生名称 |
| major | Varchar(50) | 专业 |

图书表：book

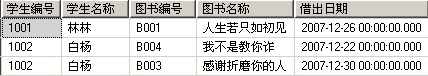
|  |  |  |
| --- | --- | --- |
| **字段名称** | **数据类型** | **说明** |
| BID | char(10) | 图书编号，主键 |
| title | char(50) | 书名 |
| author | char(20) | 作者 |

借书信息表：borrow

|  |  |  |
| --- | --- | --- |
| **字段名称** | **数据类型** | **说明** |
| borrowID | char(10) | 借书编号，主键 |
| stuID | char(10) | 学生编号，外键 |
| BID | char(10) | 图书编号，外键 |
| T\_time | datetime | 借书日期 |
| B\_time | datetime | 还书日期 |

请编写SQL语句完成以下的功能：

1. 查询“计算机”专业学生在“2007-12-15”至“2008-1-8”时间段内借书的学生编号、学生名**称、图书编号、图书名称、借出日期；参考查询结果如**下图所示：



1. 查询所有借过图书的学生编号、学生名称、专业；参考查询结果如下图所示：



1. 查询借过作者为“安意如”的图书的学生姓名、图书名称、借出日期、归还日期；参考查询结果如下图所示：



1. 查询目前借书但未归还图书的学生名称及未还图书数量；参考查询结果如下图所示：



附加：建表语句:

|  |
| --- |
| USE master  GO  /\*$$$$$$$$$$$$$建库$$$$$$$$$$$$$$$$$$$$$$$$\*/  --检验数据库是否存在，如果为真，删除此数据库--  IF exists(SELECT \* FROM sysdatabases WHERE name='BOOK')  DROP DATABASE BOOK  GO  CREATE DATABASE BOOK  GO  --建数据表--  USE BOOK  GO  CREATE TABLE student --学生信息表  (  stuID CHAR(10) primary key, --学生编号  stuName CHAR(10) NOT NULL , --学生名称  major CHAR(50) NOT NULL --专业  )  GO  CREATE TABLE book --图书表  (  BID CHAR(10) primary key, --图书编号  title CHAR(50) NOT NULL, --书名  author CHAR(20) NOT NULL, --作者  )  GO  CREATE TABLE borrow --借书表  (  borrowID CHAR(10) primary key, --借书编号  stuID CHAR(10) foreign key(stuID) references student(stuID), --学生编号  BID CHAR(10) foreign key(BID) references book(BID),--图书编号  T\_time datetime NOT NULL, --借出日期  B\_time datetime --归还日期  )  GO  --学生信息表中插入数据--  INSERT INTO student(stuID,stuName,major)VALUES('1001','林林','计算机')  INSERT INTO student(stuID,stuName,major)VALUES('1002','白杨','计算机')  INSERT INTO student(stuID,stuName,major)VALUES('1003','虎子','英语')  INSERT INTO student(stuID,stuName,major)VALUES('1004','北漂的雪','工商管理')  INSERT INTO student(stuID,stuName,major)VALUES('1005','五月','数学')  --图书信息表中插入数据--  INSERT INTO book(BID,title,author)VALUES('B001','人生若只如初见','安意如')  INSERT INTO book(BID,title,author)VALUES('B002','入学那天遇见你','晴空')  INSERT INTO book(BID,title,author)VALUES('B003','感谢折磨你的人','如娜')  INSERT INTO book(BID,title,author)VALUES('B004','我不是教你诈','刘庸')  INSERT INTO book(BID,title,author)VALUES('B005','英语四级','白雪')  --借书信息表中插入数据--  INSERT INTO borrow(borrowID,stuID,BID,T\_time,B\_time)VALUES('T001','1001','B001','2007-12-26',null)  INSERT INTO borrow(borrowID,stuID,BID,T\_time,B\_time)VALUES('T002','1004','B003','2008-1-5',null)  INSERT INTO borrow(borrowID,stuID,BID,T\_time,B\_time)VALUES('T003','1005','B001','2007-10-8','2007-12-25')  INSERT INTO borrow(borrowID,stuID,BID,T\_time,B\_time)VALUES('T004','1005','B002','2007-12-16','2008-1-7')  INSERT INTO borrow(borrowID,stuID,BID,T\_time,B\_time)VALUES('T005','1002','B004','2007-12-22',null)  INSERT INTO borrow(borrowID,stuID,BID,T\_time,B\_time)VALUES('T006','1005','B005','2008-1-6',null)  INSERT INTO borrow(borrowID,stuID,BID,T\_time,B\_time)VALUES('T007','1002','B001','2007-9-11',null)  INSERT INTO borrow(borrowID,stuID,BID,T\_time,B\_time)VALUES('T008','1005','B004','2007-12-10',null)  INSERT INTO borrow(borrowID,stuID,BID,T\_time,B\_time)VALUES('T009','1004','B005','2007-10-16','2007-12-18')  INSERT INTO borrow(borrowID,stuID,BID,T\_time,B\_time)VALUES('T010','1002','B002','2007-9-15','2008-1-5')  INSERT INTO borrow(borrowID,stuID,BID,T\_time,B\_time)VALUES('T011','1004','B003','2007-12-28',null)  INSERT INTO borrow(borrowID,stuID,BID,T\_time,B\_time)VALUES('T012','1002','B003','2007-12-30',null) |

标准答案:

|  |
| --- |
| -- 1)查询“计算机”专业学生在“2007-12-15”至“2008-1-8”时间段内借书的学生编号、学生名称、图书编号、图书名称、借出日期—  select 学生编号=stuID,学生名称=(select stuName from student where stuID=borrow.stuID),图书编号=BID,图书名称=(select title from book where BID=borrow.BID),借出日期=T\_time from borrow where stuID in (select stuID from student where major='计算机') and T\_time>'2007-12-15' and T\_time<'2008-1-8'  -- 2)查询所有借过图书的学生编号、学生名称、专业--  select 学生编号=stuID,学生名称=stuName,专业=major from student where stuID in (select stuID from borrow)  -- 3)查询借过作者为“安意如”的图书的学生姓名、图书名称、借出日期--  select 学生名称=(select stuName from student where stuID=borrow.stuID),图书名称=(select title from book where BID=borrow.BID),借出日期=T\_time,归还日期=B\_time from borrow where BID in (select BID from book where author='安意如')  -- 4)查询目前借书但未归还图书的学生名称及未还图书数量--  select 学生名称=(select stuName from student where stuID=borrow.stuID),借书数量=count(\*) from borrow where B\_time is null group by stuID |

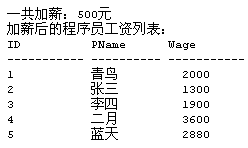
# 题目2

程序员工资表：ProWage

|  |  |  |
| --- | --- | --- |
| **字段名称** | **数据类型** | **说明** |
| ID | int | 自动编号，主键 |
| PName | Char(10) | 程序员姓名 |
| Wage | int | 工资 |

创建一个存储过程，对程序员的工资进行分析，月薪1500到10000不等，如果有百分之五十的人薪水不到2000元，给所有人加薪，每次加100，再进行分析，直到有一半以上的人大于2000元为止，存储过程执行完后，最终加了多少钱？

例如：如果有百分之五十的人薪水不到2000，给所有人加薪，每次加100元，直到有一半以上的人工资大于2000元，调用存储过程后的结果如图：



请编写T-SQL来实现如下功能：

1. 创建存储过程，查询是否有一半程序员的工资在2200、3000、3500、4000、5000或6000元之上，如果不到分别每次给每个程序员加薪100元，至之一半程序员的工资达到2200，3000，3500，4000，5000或6000元。
2. 创建存储过程，查询程序员平均工资在4500元，如果不到则每个程序员每次加200元，至到所有程序员平均工资达到4500元。

建表语句

|  |
| --- |
| USE master  GO  /\*$$$$$$$$$$$$$建库$$$$$$$$$$$$$$$$$$$$$$$$\*/  --检验数据库是否存在，如果为真，删除此数据库--  IF exists(SELECT \* FROM sysdatabases WHERE name='Wage')  DROP DATABASE Wage  GO  CREATE DATABASE Wage  GO  --建数据表--  USE Wage  GO  CREATE TABLE ProWage --程序员工资表  (  ID int identity(1,1) primary key, --工资编号  PName CHAR(10) NOT NULL , --程序员姓名  Wage int NOT NULL --工资  )  GO  --插入数据--  INSERT INTO ProWage(PName,Wage)VALUES('青鸟',1900)  INSERT INTO ProWage(PName,Wage)VALUES('张三',1200)  INSERT INTO ProWage(PName,Wage)VALUES('李四',1800)  INSERT INTO ProWage(PName,Wage)VALUES('二月',3500)  INSERT INTO ProWage(PName,Wage)VALUES('蓝天',2780) |

标准答案：

|  |
| --- |
| --1、创建存储过程--  if exists (select \* from sysobjects where name='Sum\_wage')  drop procedure Sum\_wage  GO  create procedure Sum\_wage  @PWage int,  @AWage int,  @total int  as  while (1=1)  begin  if (select count(\*) from ProWage)>2\*(select count(\*) from ProWage where Wage>=@PWage)  update ProWage set @total=@total+@AWage,Wage=Wage+@AWage  else  break  end  print'一共加薪：'+convert(varchar,@total)+'元'  print'加薪后的程序员工资列表：'  select \* from ProWage  --调用存储过程1--  exec Sum\_wage @PWage=2000,@AWage=100,@total=0  exec Sum\_wage @PWage=2200,@AWage=100,@total=0  exec Sum\_wage @PWage=3000,@AWage=100,@total=0  exec Sum\_wage @PWage=4000,@AWage=100,@total=0  exec Sum\_wage @PWage=5000,@AWage=100,@total=0  exec Sum\_wage @PWage=6000,@AWage=100,@total=0  --2、创建存储过程2--  if exists (select \* from sysobjects where name='Avg\_wage')  drop procedure Avg\_wage  GO  create procedure Avg\_wage  @PWage int,  @AWage int,  @total int  as  while (1=1)  begin  if ((select Avg(Wage) from ProWage)<=@PWage)  update ProWage set @total=@total+@AWage,Wage=Wage+@AWage  else  break  end  print'一共加薪：'+convert(varchar,@total)+'元'  print'加薪后的程序员工资列表：'  select \* from ProWage  --调用存储过程--  exec Avg\_wage @PWage=3000,@AWage=200,@total=0  exec Avg\_wage @PWage=4500,@AWage=200,@total=0 |

# 题目3：

学生成绩信息三个表，结构如下：

学生表：Member

|  |  |  |
| --- | --- | --- |
| **字段名称** | **数据类型** | **说明** |
| MID | Char(10) | 学生号，主键 |
| MName | Char(50) | 姓名 |

课程表：

|  |  |  |
| --- | --- | --- |
| **字段名称** | **数据类型** | **说明** |
| FID | Char(10) | 课程，主键 |
| FName | Char(50) | 课程名 |

成绩表：Score

|  |  |  |
| --- | --- | --- |
| **字段名称** | **数据类型** | **说明** |
| SID | int | 自动编号，主键，成绩记录号 |
| FID | Char(10) | 课程号，外键 |
| MID | Char(10) | 学生号，外键 |
| Score | int | 成绩 |

请编写T-SQL语句来实现如下功能：

1. 查询各个学生语文、数学、英语、历史课程成绩，例如下表：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **姓名** | **语文** | **数学** | **英语** | **历史** |
| 张萨 | 78 | 67 | 89 | 76 |
| 王强 | 89 | 67 | 84 | 96 |
| 李三 | 70 | 87 | 92 | 56 |
| 李四 | 80 | 78 | 97 | 66 |

1. 查询四门课中成绩低于70分的学生及相对应课程名和成绩。
2. 统计各个学生参加考试课程的平均分，且按平均分数由高到底排序。
3. 创建存储过程，分别查询参加1、2、3、4门考试及没有参加考试的学生名单，要求显示姓名、学号。

建表语句:

|  |
| --- |
| USE master  GO  /\*$$$$$$$$$$$$$建库$$$$$$$$$$$$$$$$$$$$$$$$\*/  --检验数据库是否存在，如果为真，删除此数据库--  IF exists(SELECT \* FROM sysdatabases WHERE name='Student')  DROP DATABASE Student  GO  CREATE DATABASE Student  GO  --建数据表--  USE Student  GO  CREATE TABLE Member --学生表  (  MID char(10) primary key, --学生号  MName CHAR(50) NOT NULL --姓名  )  GO  CREATE TABLE F --课程表  (  FID char(10) primary key, --课程号  FName CHAR(50) NOT NULL --课程名  )  GO  CREATE TABLE score --学生成绩表  (  SID int identity(1,1) primary key, --成绩记录号  FID char(10) foreign key(FID) references F(FID) , --课程号  MID char(10) foreign key(MID) references Member(MID) , --学生号  Score int NOT NULL --成绩  )  GO  --课程表中插入数据--  INSERT INTO F(FID,FName)VALUES('F001','语文')  INSERT INTO F(FID,FName)VALUES('F002','数学')  INSERT INTO F(FID,FName)VALUES('F003','英语')  INSERT INTO F(FID,FName)VALUES('F004','历史')  --学生表中插入数据--  INSERT INTO Member(MID,MName)VALUES('M001','张萨')  INSERT INTO Member(MID,MName)VALUES('M002','王强')  INSERT INTO Member(MID,MName)VALUES('M003','李三')  INSERT INTO Member(MID,MName)VALUES('M004','李四')  INSERT INTO Member(MID,MName)VALUES('M005','阳阳')  INSERT INTO Member(MID,MName)VALUES('M006','虎子')  INSERT INTO Member(MID,MName)VALUES('M007','夏雪')  INSERT INTO Member(MID,MName)VALUES('M008','璐璐')  INSERT INTO Member(MID,MName)VALUES('M009','珊珊')  INSERT INTO Member(MID,MName)VALUES('M010','香奈儿')  --成绩表中插入数据--  INSERT INTO Score(FID,MID,Score)VALUES('F001','M001',78)  INSERT INTO Score(FID,MID,Score)VALUES('F002','M001',67)  INSERT INTO Score(FID,MID,Score)VALUES('F003','M001',89)  INSERT INTO Score(FID,MID,Score)VALUES('F004','M001',76)  INSERT INTO Score(FID,MID,Score)VALUES('F001','M002',89)  INSERT INTO Score(FID,MID,Score)VALUES('F002','M002',67)  INSERT INTO Score(FID,MID,Score)VALUES('F003','M002',84)  INSERT INTO Score(FID,MID,Score)VALUES('F004','M002',96)  INSERT INTO Score(FID,MID,Score)VALUES('F001','M003',70)  INSERT INTO Score(FID,MID,Score)VALUES('F002','M003',87)  INSERT INTO Score(FID,MID,Score)VALUES('F003','M003',92)  INSERT INTO Score(FID,MID,Score)VALUES('F004','M003',56)  INSERT INTO Score(FID,MID,Score)VALUES('F001','M004',80)  INSERT INTO Score(FID,MID,Score)VALUES('F002','M004',78)  INSERT INTO Score(FID,MID,Score)VALUES('F003','M004',97)  INSERT INTO Score(FID,MID,Score)VALUES('F004','M004',66)  INSERT INTO Score(FID,MID,Score)VALUES('F001','M006',88)  INSERT INTO Score(FID,MID,Score)VALUES('F002','M006',55)  INSERT INTO Score(FID,MID,Score)VALUES('F003','M006',86)  INSERT INTO Score(FID,MID,Score)VALUES('F004','M006',79)  INSERT INTO Score(FID,MID,Score)VALUES('F002','M007',77)  INSERT INTO Score(FID,MID,Score)VALUES('F003','M008',65)  INSERT INTO Score(FID,MID,Score)VALUES('F004','M007',48)  INSERT INTO Score(FID,MID,Score)VALUES('F004','M009',75)  INSERT INTO Score(FID,MID,Score)VALUES('F002','M009',88) |

标准答案:

|  |
| --- |
| -- 1)查询各个学生语文、数学、英语、历史课程成绩--  SELECT Member.MName AS 姓名,  英语 = SUM(CASE F.FName WHEN '语文' THEN Score.Score END),  数学 = SUM(CASE F.FName WHEN '数学' THEN Score.Score END),  语文 = SUM(CASE F.FName WHEN '英语' THEN Score.Score END),  历史 = SUM(CASE F.FName WHEN '历史' THEN Score.Score END)  FROM Score, Member,F  WHERE F.FID = Score.FID AND Member.MID =Score.MID GROUP BY Member.MName  -- 2)查询四门课中成绩低于70分的学生及相对应课程名和成绩--  select 姓名=(select MName from Member where MID=Score.MID),课程名=(select FName from F where FID=Score.FID),成绩=Score from Score where Score<70  -- 3)统计各个学生四课程的平均分，且按平均分数由高到底排序--  select 姓名=(select MName from Member where MID=Score.MID),平均分=Avg(Score) from Score group by MID order by 平均分 desc  -- 4)创建存储过程--  if exists (select \* from sysobjects where name='P\_stu')  drop procedure P\_stu  GO  create procedure P\_stu  @num int  As  print'参加'+convert(varchar(5),@num)+'门课考试的学生姓名及学号：'  select 姓名=(select MName from Member where MID=Score.MID),学号=MID from Score group by MID having count(\*)=@num  --调用存储过程--  exec P\_stu @num=2 |