《数据库系统原理》课程实训报告

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **课程名称** | 数据库系统原理 | **指导教师** | 张肖霞 | **实训日期** | 第5-8周星期二三四节 |
| **班 级** | 20软件2班 | **姓 名** | 李佳骏 | **学 号** | B20190103224 |
| **实训总学时** | 8 | **实训成绩** | |  | |
| **教师评语** |  | | | | |

报告正文

|  |
| --- |
| **一、实训项目名称**  关系数据库设计及数据库标准语言SQL |
| **二、实训目的及要求**   1. 掌握MySQL数据库安装配置。 2. 掌握根据需求说明设计概念模型的方法； 3. 掌握将概念模型转化为关系模型的方法； 4. 掌握在MySQL数据库中创建数据库和表的方法。掌握MySQL数据库中创建数据库和表的方法； 5. 掌握关系数据库标准语言SQL的数据查询、数据更新的方法。 6. 掌握视图和索引的定义及使用。 |
| **三、实验环境**  硬件： 电脑  软件：MYSQL+NAVICATE |
| **四、实验步骤及结果**  **第一部分：熟悉MySQL的实验环境及完成相关数据库的设计。**  **一、MySQL的启动和登录MySQL服务**  **1.启动MySQL服务的两种方式**  第一种方式：CMD控制台关闭/启动MySQL服务   * 第一步：在DOS命令窗口中使用net stop命令关闭mySQL服务，如下所示：   C:\Users\Administrator\AppData\Roaming\Tencent\Users\52499915\QQ\WinTemp\RichOle\9A(JZ~O4_]R2}}9~KR{724G.png   * 第二步：打开Navicate连接mySQL数据库，当服务关闭时，会出现如下图所示的连接错误：   C:\Users\Administrator\AppData\Roaming\Tencent\Users\52499915\QQ\WinTemp\RichOle\W6[IMIZ_HB``~Q[YJT`K}SY.png   * 第三步：在DOS命令窗口中通过net start命令启动mySQL服务。如下图所示：   C:\Users\Administrator\AppData\Roaming\Tencent\Users\52499915\QQ\WinTemp\RichOle\YU}MEWQT$}RLLUX}R@8)3VW.png   * 第四步：打开Navicate连接mySQL数据库，当服务启动时，会出现如下图所示的连接成功信息：   C:\Users\Administrator\AppData\Roaming\Tencent\Users\52499915\QQ\WinTemp\RichOle\$~ZV88TAT~3JM0V7(VI24_Q.pngC:\Users\Administrator\AppData\Roaming\Tencent\Users\52499915\QQ\WinTemp\RichOle\G040S1XL{GM2{CXKYUSQ2FC.png  第二种方式：手动启动/停止mySQL服务。  如下所示，在服务中启动或关闭MYSQL服务。  D:\Documents\Tencent Files\52499915\Image\ImageEditor\20210301150234.png  **2.登录MySQL服务（以下三种方式均可实现登录MySQL服务）**  （1）在DOS窗口中登录MySQL数据库。  在DOS窗口中输入mysql登录命令,如下图所示：  C:\Users\Administrator\AppData\Roaming\Tencent\Users\52499915\QQ\WinTemp\RichOle\M$_6A)M{UHUZCQWH58C)~P4.png  （2）使用mySQL客户端登录数据库。  C:\Users\Administrator\AppData\Roaming\Tencent\Users\52499915\QQ\WinTemp\RichOle\NF03A21EABII~}MS{$KEUMM.png  （3）使用navicate新建连接登录mysql数据库  **二、了解MySQL的基本语法**  1.mySQL字符集  （1）在命令窗口查看MySQL的字符集。  C:\Users\Administrator\AppData\Roaming\Tencent\Users\52499915\QQ\WinTemp\RichOle\H8RCHF5RI6W8`6T[{DOU71D.png  （2）查看mySQL字符序  C:\Users\Administrator\AppData\Roaming\Tencent\Users\52499915\QQ\WinTemp\RichOle\M}M$RS`HB@F][2]VCWM_@V4.png  （3）如何查询以utf开头的字符集对应的字符序。并将查询语句和结果进行截图。  C:\Users\Administrator\AppData\Roaming\Tencent\Users\52499915\QQ\WinTemp\RichOle\Q(@}9RPJRJJKE60]84@[}6Q.pngC:\Users\Administrator\AppData\Roaming\Tencent\Users\52499915\QQ\WinTemp\RichOle\M1A[A4PR2_4{)C(`~R)V3P7.png  2、mySQL的系统变量  （1）查看字符v开头的系统变量，如下图所示：  BGF`XNM$XTS}N]H]@HOU%SD  **（2）用系统变量查看MySQL服务器的版本和当前系统日期（常量current\_date）。（注意：系统全局变量必须在变量名称前加两个@@符号）。**  I6[EJ3`0_0EEAXZBLTW]K}G  **三、按要求完成大学数据库的相关设计**  假设要根据某大学的业务规则设计一个数据库，这个大学的业务规则如下：  1、该大学有多个学院，每一个学院有一个院长，且每一位院长只能主管一个学院。学院包括学院编号、学院名称、学院地址等信息。院长包括院长编号、姓名、电话号码等信息。  2、每一个学院有多个系。每一个系仅属于一个学院。系包括系代号、系名称、电话号码和系地址等信息。  3、每一个系聘用多名教师，但每一位教师只属于一个系。教师包括教工号、姓名、学历、职称、工资、电话号码等信息。  4、一位教师可能讲多门课，每一门课可由多位教师讲授。课程包括课程号、课程名称、学时数和学分等信息，教师讲授课程时有授课时间和授课地点。  5、每一个系有多名学生，且每一名学生只能属于一个系。学生包括学号、姓名、性别、出生年月、专业、电话号码等信息。  6、每一名学生可选修多门课，且一门课有多名学生选修，学生选修课程有选修成绩。  实验内容及步骤：  **1．根据需求说明设计某大学数据库的概念模型（请将概念模型粘贴于此处）。**  **图片1**  **2.设计该大学数据库的关系模型，要求给出每个关系的主键及外键。**  图片2  **3．创建大学数据库及其相关的数据表。（不需要使用sql语句，定义表中的主键及外键）**  **图片3**  **4．创建大学数据库关系图。**  **图片4**  **第一部分结束。**  **第二部分 关系数据库标准语言SQL——单表查询**  **（1）在navicate中使用SQL语句完成test\_4数据库的创建，指定其默认字符集为 utf8，默认校对规则为utf8\_general\_ci,参考语法如下所示：**  6CM6V~`~OIXDA`01O0~VEHV  **（2）分别使用SQL语句完成student,course,score,teacher,teach\_course五张表的创建。五张表的结构分别如下图所示：**  Student表   |  |  |  |  | | --- | --- | --- | --- | | 字段名称 | 数据类型 | 备注 | 约束 | | studentNo | char(12) | 学号 | 主键 | | sname | char(8) | 姓名 |  | | sex | char(2) | 性别 |  | | birthdate | date | 出生年月日 |  | | entrance | int | 总分数 |  | | phone | char(11) | 电话号码 |  | | email | varchar(20) | 电子邮件 |  | | departmentName | varchar(20) | 所在系 |  |   course表   |  |  |  |  | | --- | --- | --- | --- | | 字段名称 | 数据类型 | 备注 | 约束 | | courseno | char(6) | 课程号 | 主键 | | cname | varchar(20) | 课程名 |  | | type | char(8) | 课程类型 |  | | cpno | char(6) | 先行课 |  | | period | int | 总学时 |  | | exp | int | 实验学时 |  | | term | int | 开课学期 |  |   score表   |  |  |  |  | | --- | --- | --- | --- | | 字段名称 | 数据类型 | 备注 | 约束 | | studentno | char(12) | 学号 | 主键，外键 | | courseno | char(6) | 课程号 | 主键，外键 | | daily | float(3,1) | 平时成绩 |  | | final | float(3,1) | 期末成绩 |  |   teacher表   |  |  |  |  | | --- | --- | --- | --- | | 字段名称 | 数据类型 | 备注 | 约束 | | teacherno | char(12) | 教工号 | 主键 | | tname | varchar(6) | 教师姓名 |  | | major | varchar(20) | 专业 |  | | prof | Char(10) | 职称 |  | | department | Varchar(20) | 部门 |  |   teach\_course表   |  |  |  |  | | --- | --- | --- | --- | | 字段名称 | 数据类型 | 备注 | 约束 | | teacherno | char(6) | 教工号 | 主键，外键 | | courseno | char(6) | 课程号 | 外键 | | teachtime | datetime | 授课时间 | 主键 |   **注意定义主键和外键约束，创建之后的物理模型图如下所示：**    在创建的过程中  请思考以下问题，并将答案写在下面。   1. 五张表的创建顺序是否有要求？   答: 有要求,对于没有外键关联的表之间可任意顺序, 对于有外键关联的表,其被关联的键所在的表必须先创建. 即先创建 主键没有外键参照的表，最后创建有外键关联的表.  在本实验报告中顺序可为:Student表 -> course表 -> score 表 -> teacher 表 -> teach\_course 表   1. 五张表的关系是如何创建的？   答: 外键使用外键约束语句如: foreign key (student\_no) references student(studentNO)  主键约束可使用primary key(主键名1,主键名2 , ...)  建表sql语句如下:  create table student(  studentNO char(12) primary key not null, -- 学号  sname char(8), -- 姓名  sex char(2), -- 性别  birthdate date, -- 出生年月日  phone char(11), -- 电话号码  email varchar(20), -- 电子邮箱  departmentName varchar(20) -- 所在系  );  create table course(  courseNo char(6) primary key not null, -- 课程号  cname varchar(20), -- 课程名  type char(8), -- 课程类型  cpno char(6), -- 先行课  period int, -- 总学时  exp int, -- 实验学时  term int -- 开课学期  );  create table score(  student\_no char(12), -- 学号  course\_no char(6), -- 课程号  daily float(3, 1), -- 平时成绩  final float(3, 1), -- 期末成绩  foreign key (student\_no) references student(studentNO),  foreign key (course\_no) references course(courseNo),  primary key (student\_no, course\_no)  );  create table teacher(  teacherNo char(12) primary key not null, -- 教工号  tname varchar(6), -- 教师姓名  major varchar(20), -- 专业  prof char(10), -- 职称  department varchar(20) -- 部门  );  create table teach\_course(  teacher\_no char(6), -- 教工号  course\_no char(6), -- 课程号  teachtime datetime, -- 授课时间  foreign key (teacher\_no) references teacher(teacherNo),  foreign key (course\_no) references course(courseNo),  primary key (teacher\_no, teachtime)  );  （3）请使用SQL语句分别完成下面五张表的数据的插入，其数据参考如下  /\*学生表数据插入\*/  Insert into student values  ('18122221324','何白露','女','2000/12/4',879,'13786112345','heyy@sina.com','机电院'),  ('18125111109','敬横江','男','2000/3/1',789,'15678945623','jing@sina.com','计数院'),  ('18125121107','王伟','男','1999/9/12',790,'13786188347','3267888@QQ.com','计数院'),  ('18137156732','吴英','女','2000/10/19',888,'13055568618','17865@QQ.com','经管院');  /\*课程表数据插入\*/  ('c05103','电子技术','选修',null,48,16,2),  ('c05107','程序设计基础','必修',null,48,16,4),  ('c05109','C语言','必修','c05108',48,16,4),  ('c05127','Java语言','必修','c05109',64,16,4),  ('c05138','软件工程','必修','c05109',48,8,5),  ('c05124','经济学','必修',null,64,16,4),  ('c05108','数据库原理','必修','c05107',48,16,3),  ('c05222','会计实务','必修',null,48,8,2),  ('c05223','UML\_SOFT','选修','c05224',48,8,2),  ('c05224','UML设计','选修','c05138',64,8,2);  /\*成绩表数据插入\*/  ('18122221324','c05103',87.0,92.0),  ('18122221324','c05109',85.0,90.0),  ('18122221324','c05127',95.0,93.0),  ('18125111109','c05127',91.0,88.0),  ('18125111109','c05138',80.0,80.0),  ('18125111109','c05108',76.0,78.0),  ('18125121107','c05103',88.0,72.0),  ('18125121107','c05127',81.0,76.0),  ('18125121107','c05138',75.0,74.0),  ('18137156732','c05124',89.0,88.0),  ('18137156732','c05222',81.0,79.0);  /\*教师表数据插入\*/  ('t05001','苏超然','软件工程','教授','计数院'),  ('t05002','常斌','会计学','助教','经管院'),  ('t05003','孙石安','网络安全','教授','计数院'),  ('t05011','卢先','软件工程','副教授','计数院'),  ('t05013','李明','机械制造','讲师','机电院'),  ('t05014','王石','软件工程',NULL,'计数院');  /\*教师授课表数据插入\*/  ('t05001','c05109','2021-03-08 10:00:00'),  ('t05001','c05127','2021-03-09 8:00:00'),  ('t05001','c05109','2021-03-10 14:00:00'),  ('t05003','c05124','2021-03-11 16:00:00'),  ('t05011','c05127','2021-03-13 16:00:00'),  ('t05011','c05127','2021-03-15 10:00:00');  SELECT \* FROM student;  SELECT \* FROM course;  SELECT \* FROM score;  SELECT \* FROM teacher;  SELECT \* FROM teach\_course;  （4）按要求完成以下1-20题的查询，在每道题目的下面粘贴好SQL语句及运行效果截图。完成其中的12道题及格，16道题良好，20道题优秀。  /\*1、在student表中查询出生日期在2003年以前的学生的学号、姓名和年龄\*/  select studentNO, sname, TIMESTAMPDIFF(YEAR, student.birthdate, CURDATE()) age from student where birthdate < '2003-01-01';  截图_选择区域_20220329105351  /\*2、查询student表中女生的人数，以别名“女生人数”命名\*/  select count('女') '女生人数' from student where sex = '女';    /\*3、查询teacher表中每一位教授的教师号，姓名和专业名称\*/  select teacherNo, tname, major from teacher;    /\*4、查询teacher表中没有职称的教师的教工号，姓名\*/  select teacherNo, tname from teacher where prof is null;    /\*5、查询选修课程号为c05103的学生的学号和期末成绩，并要求平时成绩在85到100分之间\*/  select student\_no, final from score where course\_no = 'c05103' and daily between 85 and 100;    /\*6、按性别统计student表中学生的平均年龄，并输出各性别下学生的平均年龄的值（注意以平均年龄来命名）\*/  select sex , round(avg(TIMESTAMPDIFF(YEAR, student.birthdate, CURDATE())), 0) 平均年龄 from student group by sex;    /\*7、统计选课门数超过3门以上的学生的学号及其选课门数\*/  select score.student\_no, count(student\_no) cnt from score group by student\_no having cnt >= 3;    /\*8、统计每个学生所选课程总评成绩的平均分，其中总评成绩=daily\*0.2+final\*0.8，平均分保留小数位1位\*/  select student\_no, format(avg(daily \* 0.2 + final \* 0.8), 1) '总评成绩平均分' from score group by student\_no;  /\*9、输出student表中年龄最大的男生的所有信息\*/  select \* from student where sex = '男' order by birthdate limit 1;    /\*10、创建新表stu\_new，该新表中包含学号、课程号和总评成绩。其中总评成绩=daily\*0.2+final\*0.8\*,查看该新表的结构\*/  create table stu\_new(  studentNo char(12),  course\_no varchar(20),  total double,  foreign key (course\_no) references course(courseNo),  primary key (studentNo, course\_no)  );  select student\_no, course\_no, format(daily \* 0.2 + final \* 0.8, 1) '总评成绩平均分' from score;  insert into stu\_new values ('18122221324','c05103', 91.0),  ('18122221324','c05109', 89.0),  ('18122221324','c05127', 93.4),  ('18125111109','c05108', 77.6),  ('18125111109','c05127', 88.6),  ('18125111109','c05138', 80.0),  ('18125121107','c05103', 75.2),  ('18125121107','c05127', 77.0),  ('18125121107','c05138', 74.2),  ('18137156732','c05124', 88.2),  ('18137156732','c05222', 79.4);  select \* from stu\_new;    /\*11、显示所有姓王学生的姓名、出生日期、EMAIL\*/  select sname, birthdate, email from student where sname like '王%';    /\*12、在score表中查询总评成绩大于85分的学生的学号、课程号和总评成绩，并按照课程号的升序，再按照总评成绩的降序排列\*/  select student\_no, course\_no, 0.2 \* daily + 0.8 \* final total from score where 0.2 \* daily + 0.8 \* final > 85 order by course\_no , total desc;    /\*13、查询选课在2门以上且各门课程期末成绩均高于85分的学生的学号及其总成绩，查询结果按照总成绩降序列出\*/  select student\_no 学号, sum(final) 总成绩 from score where final > 85 group by student\_no having count(student\_no) > 2 order by '总成绩' desc;    /\*14、查询所选课程期末平均成绩最高的前三个学生的学号,并按期末平均成绩降序输出\*/  select student\_no 学号, avg(final) 平均成绩 from score group by student\_no order by '平均成绩' desc limit 3;    /\*15、查询score表中学生的期末总成绩大于250分的学生的学号、期末总成绩和期末平均成绩\*/  select student\_no 学号, sum(final) 总成绩, avg(final) 平均成绩 from score group by student\_no having 总成绩 > 250;    /\*16、查询选修课程号为c05103的学生的期末最高分、最低分及之间相差的分数\*/  select max(final) - min(final) 相差分数 from score where course\_no = 'c05103';    /\*17、查询以"UML\_"开头，且倒数第3个字符为O的课程的详细情况。\*/  select \* from course where cname like 'UML\\_%O\_\_';    /\*18、统计姓名中含有三个汉字的学生的数量\*/  -- 一个中文3字节,英文数字都是1字节, 以上均为1个字符  select count(studentNo) 数量 from student where length(sname) = 9 and char\_length(sname) = 3; -- 前面字节数, 后面字符数    /\*19、查询近七日内同一个老师上同一门课程超过两次的所有的老师的授课信息。\*/  select  t1.\*  from  teach\_course t1,  (select teacher\_no, course\_no  from teach\_course  where day(now()) - day(teachtime) between 0 and 7  group by teacher\_no, course\_no  having count(teacher\_no) >= 2) t2  where  t1.teacher\_no = t2.teacher\_no and t1.course\_no = t2.course\_no;    /\*20、查询每个老师所上的课程的总课时，要求输出教师号，课程号以及以上课时数（一次课是2个课时）\*/  select teacher\_no 教师号, course\_no 课程号, count(teacher\_no) \* 2 总课时数 from teach\_course group by teacher\_no, course\_no;  **图片11**  **第二部分结束**  **第三部分 关系数据库标准语言SQL——多表查询**  1、按要求完成以下1-20题的查询，在每道题目的下面粘贴好SQL语句及运行效果截图。完成其中的12道题及格，16道题良好，20道题优秀。  /\*1、找出至少一门课程的期末成绩在90分以上的女学生的姓名。\*/  (**select** studentno, **count**(studentno) cnt **from** score **where** score*.*final > 90 **group by** studentno) t2    /\*2、查询选修课程名为Java语言课程且期末成绩在90分以上的学生的学号。\*/  **select** studentNo **from** score t1, course t2 **where** t1*.*courseno = t2*.*courseno **and** t2*.*cname = 'Java语言' **and** t1*.*final > 90;    /\*3、查询选修课程号为c05103的学生的学号、姓名和期末成绩\*/  **select** t1*.*studentNo 学号, t1*.*sname 姓名, t2*.*final 期末成绩 **from** student t1,  score t2 **where** t1*.*studentNo = t2*.*studentno  **and** t2*.*courseno = 'c05103';    /\*4、统计选课门数超过两门的学生的学号及姓名\*/  **select** studentNo 学号, sname 姓名 **from** student **where** studentNo **in** (**select** studentno **from** score t **group by** t*.*studentno **having count**(t*.*studentno) > 2);    /\*5、查询选修了姓“苏”的老师的课程的学生的学号。\*/  **select distinct** studentno 学号 **from** score **where** courseno **in** (**select distinct** courseno  **from** teach\_course t  **where** t*.*teacherno **in** (**select** teacherno **from** teacher t **where** t*.*tname **like** '苏%')); *-- 或* **select distinct** t1*.*studentno 学号 **from** score t1,  teach\_course t2,  teacher t3 **where** t1*.*courseno = t2*.*courseno  **and** t3*.*teacherno = t2*.*teacherno  **and** t3*.*tname **like** '苏%';    /\*6、求每个学生所选修课程的期末平均分，要求显示学生的学号，姓名，及所选修课程的期末平均成绩（期末平均成绩作为别名显示，且运用round函数使期末平均成绩保留一位小数），并按期末平均成绩从高到低排序。\*/  **select** t1*.*studentNo 学号, t1*.*sname 姓名, t2*.*courseno 选修课程, (t2*.*final + t2*.*daily) / 2 平均成绩 **from** student t1,  score t2 **where** t1*.*studentNo = t2*.*studentno **order by** 平均成绩 **desc**;    /\*7、查询19级学生的学号、姓名、课程名、期末成绩及学分（要求以别名学分显示所求学分，其中每门课程的学分等于该课程的总学时除以16）\*/  **select** t1*.*studentNo 学号, t1*.*sname 姓名, t2*.*courseno 课程名, t2*.*final 期末成绩, t3*.*period + t3*.*exp 学时 **from** student t1,  score t2,  course t3 **where** t2*.*studentno **in** (**select** studentNo **from** student **where** studentNo **like** '19%')  **and** t2*.*studentno = t1*.*studentNo  **and** t2*.*courseno = t3*.*courseno;    /\*8、查询与“王伟”在同一个系学习的学生的基本信息\*/  **select** \* **from** student **where** student*.*departmentname **in** (**select** departmentname  **from** student  **where** sname = '王伟');    /\*9、查询选修了“数据库原理"课程的学生的学号和姓名\*/  **select** studentNo, sname **from** student **where exists**(  **select** \*  **from** score t1,  course t2  **where** t1*.*courseno = t2*.*courseno  **and** t2*.*cname = '数据库原理'  **and** student*.*studentNo = t1*.*studentno  );      /\*10、查询期末成绩大于等于90分、总评成绩高于85分的学生的学号、课程号和总评成绩（要求以别名总评成绩显示总评成绩，其中总评成绩=平时成绩\*30%+期末成绩\*70%)\*/  **select** studentno, courseno, 0.8 \* score*.*final + score*.*daily \* 0.2 总评成绩 **from** score **where** final > 90  **and** final \* 0.8 + score*.*daily \* 0.2 > 85;    /\*11、查询期末成绩比选修课程平均期末成绩低的学生的姓名、课程号和期末成绩\*/  **select** t2*.*sname, courseno, t1*.*final 期末成绩 **from** score t1,  student t2 **where** final < (t1*.*final + t1*.*daily) / 2  **and** t1*.*studentno = t2*.*studentNo;    /\*12、查询期末成绩中含有高于90分的学生的学号、姓名、电话及课程名\*/  **select** t1*.*studentNo, sname, phone, cname **from** student t1,  score t2,  course t3 **where** t2*.*final > 90  **and** t1*.*studentNo = t2*.*studentno  **and** t2*.*courseno = t3*.*courseno;    /\*13、查找score表中所有比c05103课程期末成绩都高的学生的学号、姓名和期末成绩\*/  **select** t1*.*studentNo, t1*.*sname, t2*.*final **from** student t1,  score t2 **where** t2*.*final > **all** (**select** final **from** score **where** score*.*courseno = 'c05103')  **and** t1*.*studentNo = t2*.*studentno;    /\*14、查询每一课程的间接先行课(即先行课的先行课)。\*/  **select** t1*.*courseno, t1*.*cname, t2*.*cpno **from** course t1  **left join** course t2  **on** t1*.*cpno = t2*.*courseno;    /\*15、找出每个学生期末成绩超过他选修课程期末平均成绩的学生的学号及课程号\*/  **select** studentno, courseno **from** score **where** final > (score*.*final + score*.*daily) / 2;    /\*16、查询年龄低于所有计数院学生的学生姓名、所在系、年龄。\*/  **select** sname, departmentname, **year**(now()) - **year**(birthdate) age **from** student **where year**(now()) - **year**(birthdate) < **all** (**select year**(now()) - **year**(birthdate) **from** student **where** departmentname = '计数院');    /\*17、查询选修了全部课程的学生姓名。\*/  *-- 没有一门课是他不选的* **select** sname **from** student t1 **where not exists**(  **select** \*  **from** score t2  **where not exists**(  **select** \*  **from** score t3  **where** t3*.*studentno = t1*.*studentNo  **and** t3*.*courseno = t2*.*courseno  )  );    /\*18、查询至少选修了学生18125111109选修的全部课程的学生学号,注意查询结果中不应该含有18125111109学生\*/  **select distinct** studentno **from** score t1 **where not exists**(  **select** \*  **from** score t2  **where** t2*.*studentno = '18125111109'  **and not exists**(  **select** \*  **from** score t3  **where** t3*.*courseno = t2*.*courseno  **and** t3*.*studentno = t1*.*studentno  )  )  **and** studentno != '18125111109';    /\*19、查询2021年3月份苏姓老师的授课信息，包括教工号，教师姓名，课程名，授课时间。\*/  **select** t1*.*teacherno, tname, cname, teachtime **from** teacher t1,  teach\_course t2,  course t3 **where** t1*.*teacherno = t2*.*teacherno  **and** t2*.*courseno = t3*.*courseno  **and** tname **like** '苏%'  **and** date\_format(teachtime, '%Y-%m') = '2021-03';    /\*20、查询没有学生选修的课程的课程号\*/  **select** courseno, cname **from** course **where** courseno **not in** (**select** courseno **from** score);    **第三部分结束**  **第四部分 视图、索引及数据库完整性**  /\***一、视图和索引的定义**\*/  /\*1、创建一个名为S\_1的视图，显示“计数院”学生的学号，姓名。\*/  **create view** S\_1 **as select** studentNo, sname, departmentname **from** student **where** departmentname = '计数院';    /\*2、创建一个名为S\_2的视图，显示“计数院”的所有老师的工号，姓名和所在系部,要求视图中包含WITH CHECK OPTION子句\*/  OR REPLACE：表示替换已有视图  **CREATE OR REPLACE VIEW** S\_2 **AS SELECT** teacherno, tname, department **FROM** teacher **WHERE** department = '计数院' **WITH CHECK OPTION**;    /\*3、创建一个名为S\_3的视图，要求显示学生学号、姓名、课程名及期末成绩，要求视图中包含WITH CHECK OPTION子句\*/  **CREATE OR REPLACE VIEW** S\_3 **AS SELECT** t1*.*studentNo, sname, cname, final **FROM** student t1,  course t2,  score t3 **WHERE** t1*.*studentNo = t3*.*studentno  **AND** t2*.*courseno = t3*.*courseno **WITH CHECK OPTION**;    /\*4、创建一个名为S\_4的视图,显示课程号，选修该门课程的期末最高成绩，及该期末成绩最高的学生的学号，姓名。\*/  **CREATE OR REPLACE VIEW** S\_4 **AS SELECT** t1*.*studentNo, sname, t2*.*courseno, maxx **FROM** student t1,  (**SELECT** max(final) maxx, courseno **FROM** score **GROUP BY** courseno) t2,  score t3 **WHERE** t1*.*studentNo = t3*.*studentno  **AND** t2*.*courseno = t3*.*courseno  **AND** t2*.*maxx = t3*.*final **WITH CHECK OPTION**;    /\*5、定义一个视图S\_5,该视图中包括教师姓名，所在院系，所授课程名称，授课时间\*/  **CREATE OR REPLACE VIEW** S\_5 **AS select** tname, department, cname, teachtime **from** teacher t1,  teach\_course t2,  course t3 **where** t1*.*teacherno = t2*.*teacherno  **and** t2*.*courseno = t3*.*courseno **WITH CHECK OPTION**;    /\*6、定义一个视图S\_6，该视图中包含了选修了苏超然老师C语言课程的所有学生的学号和姓名\*/  **create or replace view** S\_6 **as select** studentno, sname **from** student **where exists**(  **select** \* **from** score **where** courseno = 'c05109' **and** student*.*studentNo = score*.*studentno  ) **WITH CHECK OPTION**;    /\*7、为学生表的电话号码字段创建唯一索引\*/  **CREATE UNIQUE INDEX** phone\_idx **ON** student (phone);    /\***二、视图的使用**\*/  /\*1、在视图S\_1上，分别插入以下两条记录,并通过查看S\_1和student表分析所出现的结果:  学号为“18125111110”,姓名为“王明”,系部为“计数院"  学号为“18125111111”,姓名为“张三”,系部为“电信院" \*/  **insert into** S\_1 **values** (18125111110, '王明', '计数院'); **insert into** S\_1 **values** (18125111111, '张三', '电信院'); *-- 结果: 因为S\_1在创建时未做ＣＨＥＣＫ　检查　，所以对其插入只要符合基本表的约束即可插入成功， -- 但是不会更新视图，　而基本表会更新插入的两条数据，　因为对视图的修改最终为对基本表的修改*  /\*2、使用SQL语句将视图S\_2上姓名为"王石"的教师的系部更改为电信院,  通过查看S\_2和teacher表分析所出现的结果: \*/  **update** S\_2 **set** department = '电信院' **where** tname = '王石'; *-- 修改失败，　创建视图时设置了检查　所以错误，　基本表也修改失败*  /\*3、针对视图S\_3，完成以下操作：  （1）使用SQL语句在该视图下插入以下数据：  学号：18122221324，姓名：张一，课程名：高等数学，分数：90，观察插入是否成功，分析原因。\*/  **insert into** S\_3 **values** ('18122221324', '张一', '高等数学', 90); *-- 插入失败　因为课程表中并无高等数学*  /\*(2)使用SQL语句将学号为"19137156732"所选修的课程名为“经济学"的学生的期末成绩修改为87.观察更新是否成功，分析原因。\*/  **update** S\_3 **set** final = 87 **where** studentNo = '19137156732'  **and** cname = '经济学'; *-- 更新成功，　因为在ｖｉｅｗ表中能查到该条数据，无其他限制，修改成功*  /\*(3)使用SQL语句将学号为19137156732的学生的姓名更新为“吴英新”，选修课程的期末成绩均更新为80。观察更新是否成功，分析原因\*/  **update** S\_3 **set** final = 80,  sname = '吴英新' **where** studentNo = '19137156732'; *-- 更新失败*  /\***三、完整性约束定义的检查**\*/  （1）编写sql语句验证主键约束，观察主键重复和为null的插入情况并将运行结果截图。（任意一张表即可）  主键重复  **insert into** teacher **values** ('t05001', '主键重复测试', 'test', 'test', 'test');    主键为ｎｕｌｌ  **insert into** teacher **values** (**null**, '主键null测试', 'test', 'test', 'test');     1. 编写sql语句验证外键约束，观察外键值对数据插入情况的影响并将运行结果截图。   实验表结构为：  截图_选择区域_20220412103541  删除主表测试  **drop table** test\_4*.*student;    或  插入主表不存在的数据  **insert into** score **values** ('123456', '123456', 100, 100);    \*/  **加分思考题：**  /\*（1）、在视图S\_4的基础上统计每个学生所获得的期末最高成绩的门数，按照门数从高到低排列，未获得一门的用0来表示，比如学号为19125121107的学生的期末最高成绩的门数应该为0\*/  **select** t1*.*studentNo 学号, sname 姓名, ifnull(cnt, 0) 门数 **from** student t1  **left join** (**select** studentNo, **count**(\*) cnt **from** S\_4 **group by** studentNo) t2 **on** t1*.*studentNo = t2*.*studentNo **order by** 门数 **desc**;     1. 定义一个视图S\_7，该视图中包含了选修了刘老师所授任意课程的学生姓名。\*/   **create or replace view** S\_7 **as select distinct** sname **from** student t1,  score t2,  teach\_course t3,  teacher t4 **where** t1*.*studentNo = t2*.*studentno  **and** t2*.*courseno = t3*.*courseno  **and** t3*.*teacherno = t4*.*teacherno  **and** t4*.*tname **like** '苏%';    **第四部分结束** |
| 实训总结  GROUP BY语句通常会和聚合函数一起使用，按照一个或者多个列对结果进行分组  HAVING子句允许用户通过一个简单的语法完成原本需要通过子查询才能对GROUP BY语句产生的分组进行条件过滤的任务  聚合函数 count（），求数据表的行数  聚合函数 max（），求某列的最大数值  聚合函数min（）,求某列的最小值  聚合函数sum（）,对数据表的某列进行求和操做  avg(字段名） from 数据表  表是建立关系数据库的基本结构，用来存储数据具有已定义的属性，在表的操作过程中，有查看表信息、查看表属性、修改表中的数据、删除表中的数据及修改表和删除表的操作。从实训中让我更明白一些知识，表是数据最重要的一个数据对象，表的创建好坏直接关系到数数据库的成败，表的内容是越具体越好，但是也不能太繁琐，以后在实际应用中多使用表，对表的规划和理解就会越深刻。  查询某个月的数据(查询21年3月份数据)  select \* from table where date\_format(start\_time,'%Y-%m') = '2021-03'  sql中没有全称量词，所以当遇到全部问题时，可以转化为否定再否定（两层exists）  多表查询要注意理清各表之间的联系，防止出现无用数据和重复数据查询某个月的数据(查询21年3月份数据)  select \* from table where date\_format(start\_time,'%Y-%m') = '2021-03'  sql中没有全称量词，所以当遇到全部问题时，可以转化为否定再否定（两层exists）  多表查询要注意理清各表之间的联系，防止出现无用数据和重复数据  视图（view）是一种虚拟存在的表，是一个逻辑表，本身并不包含数据。作为一个select语句保存在数据字典中的。  　　通过视图，可以展现基表的部分数据；视图数据来自定义视图的查询中使用的表，使用视图动态生成。  基表：用来创建视图的表叫做基表base table  因为视图本身没有数据，因此对视图进行的dml操作最终都体现在基表中 |