

数据库实验一之 SQL 练习

说明

设有如下关系模式：

- Student(**SNO**, NAME, GENDER, AGE, DEPART) SNO为学生学号，DEPART为系号
- Teacher(**TNO**, NAME, GENDER, BIRTHDAY, POSITION, DEPART) TNO为教师工号，POSITION为职称，DEPART为系别
- Course(**CNO**, NAME, CPNO, CCREDIT, TNO) CNO为课程编号，CPNO为先行课编号，CCREDIT为学分
- Score(**SNO**, **CNO**, DEGREE, SEMESTER) SNO为学号，CNO为课程号，DEGREE 为成绩，SEMESTER为开课学期

PS:斜体加粗的表示该字段为主键

任务一

根据所给 Student.csv、Teacher.csv、Course.csv、Score.csv 表中的数据信息，在数据库中创建对应的关系表并将数据录入到数据库中。可能涉及到的数据类型：varchar, char, int, float, datetime。（也需要截图在报告中展示！）

任务二

依顺序写出实现以下各题功能的sql语句（注意：有些题目会受到之前题目的影响）

修改基本表

1. 在学生表中增加一个新的属性列BIRTHYEAR（出生年份可为INT类型）
2. 计算每个学生的出生年份（使用当前年份减去年龄）注意，此操作可能需要关闭安全更新模式；提示：可以使用YEAR(NOW())获取当前年份
3. 将每个学生的年龄减去2，并将年龄的类型从int改成char
4. 将BIRTHYEAR列删除
5. 创建一个学生选课课程数量表：student_course(SNO,NUM_COURSE)，两个属性分别表示授课学生学号，选课课程数量，其中 SNO 是主键。

6. 用一条语句，结合表 score 记录，为表 student 中所有学生，在表 student_course 添加对应选课数量记录（用到查询，而不是手动数 NUM 插入）
7. 删除student_course中选修1门或者3门课的学生，然后删除student_course表
8. 在Student表中添加学生姓名必须取唯一值的约束条件，然后再将这个约束条件删除
9. 删除Score表的复合主键，然后建立新的主键（SNO,CNO,SEMESTER）
10. 修改Student表中DEPART列的列名为DEPARTMENT
11. 在score表中删除DEPARTMENT为229的学生的课程记录，然后在score表中为每个学生删除其最低分的课程记录
12. 在成绩表（Score）中增加一个新的属性列GRADE（等级，CHAR类型），并根据成绩（Score）填充等级：90以上为'A'，80-89为'B'，70-79为'C'，60-69为'D'，60以下为'F'。提示，可以使用CASE WHEN语句
13. 在Student表中插入你自己的个人数据（包含你的学号，姓名缩写，性别，年龄，系号），注意：为了防止学号撞车我们将学号扩充了一位，所以你在插入的时候也可以将自己的学号扩充一位（比如加一个0）

查询（所有题目限用一条语句完成）

1. 查询和你属于同一个系的学生学号和姓名(包括你本人)。
2. 查询和你属于同一个系的学生学号和姓名(不包括你本人)。
3. 查询和YH同学不属于同一个系的学生学号和姓名。
4. 查询和你以及YH同学都不在一个系的学生学号和姓名。
5. 查询229系老师的工号和姓名。
6. 查询 11 系和 229 系教师的总人数。
7. 查询年龄最大的学生的学号、姓名和年龄。
8. 查询你的系中年龄最小的学生的学号、姓名和年龄。
9. 查询选修 DB_Design 课程(不能通过课程号直接进行查询)且成绩在 80 分及以下的学生的学号、姓名和分数
10. 查询选修过“ZDH”老师课程(不能通过课程号直接进行查询)的学生学号和姓名（去掉重复行）。
11. 查询选过某课程的学生学号和分数，并按分数降序展示。（某课程是指 course 表中的某一课程名 NAME，你自行选择；分数指的是选的这门课的成绩，不包括选这门课的同学的其他成绩）。
12. 查询每门课的平均成绩，其中每行包含课程号、课程名和平均成绩（包括平均成绩为NULL，即该课没有成绩）。
13. 查询学分大于 3(不包括3) 的课程的平均成绩，其中每行包含课程号、课程名和平均成绩（包括平均成绩为 NULL，即该课没有成绩）
14. 查询至少选修了工号TNO="TA90023"的老师（ZDH 老师）开设的所有课程的学生学号
15. 查询每门课程的最高分和最低分，并计算其分数差。其中每行包含课程号、课程名和最高分、最低分和分数差。（课程无成绩的不用包括）。
16. 查询存在考试成绩低于 78 分的学生学号，以及每个学生低于 78 分的课程的数量。
17. 查询所教过的课程中有学生考试成绩低于 72 分的教师的工号和姓名（去掉重复行）。
18. 查询选修大于 6 门课程的学生的学号、姓名。

19. 查询至少选修了学号为SNO="PB230000002"的同学（ZY同学）所选全部课程的学生学号
20. 查询student表中各个学生姓名与相应的平均成绩。
21. 查询每个系的学生人数和每个系的平均分，其中每行包含系号、系的人数和平均成绩。（计算人数的时候需要包括那些没有成绩的同学，但是计算成绩的时候不需要包括这些同学）
22. 查询所有未选修 DB_Design 课程或者 Data_Mining 课程的学生的学生姓名（去掉重复行）
23. 查询各个课程的课程名及选该课的学生们的平均年龄。（包括没有人选的课）
24. 查询选修了课程名中包含"Computer"课程的学生们的学号和姓名。
25. 设课程平均成绩为 x，查询各个课程成绩处于[x-5, x+5]区间的同学的成绩表，即包含 SNO、CNO、DEGREE
26. 查询每门课程的课程号、课程名以及在这门课程上GRADE为A的学生人数，结果按人数降序排列。
27. 查询包含 "_"（下划线）且位 "_" 于后面的字符串包含"r"的所有课程的课程名。
28. 查询每个老师教的所有课程中最高平均成绩，并展示出对应的老师工号和课程号。

索引

1. 用 create 语句在 Student 表的名称 NAME 上建立普通索引 NAME_INDEX。
2. 用 create 语句在 Course 表的课程号 CNO 上建立唯一索引 CNO_INDEX。
3. 用create 语句在 Score表中按学号升序，DEGREE成绩降序建立索引（即先按学号升序，然后按成绩降序）
4. 用一条语句查询表 score 的索引。
5. 删除Course 表的课程号 CNO 上建立的唯一索引 CNO_INDEX。

视图

1. 建立 229 系的学生视图（db_229_student），属性与 student 表一样，并要求对该视图进行修改和插入操作时仍需保证该视图只有 229 系的学生。
2. 将 229 系学生视图（db_229_student）中学号为"PB210000031"的学生姓名和学号改为你的姓名和学号。
3. 在 229 系学生视图（db_229_student）中找出年龄小于 21 岁的学生，包含 SNO、NAME、GENDER。
4. 向 student 表中插入一名"学号 SA242290001，姓名 QWE，性别male，229系，年龄22"的学生。然后查询视图 db_229_student 的所有学生，验证其是否更新。
5. 删除视图 db_229_student。

触发器

1. 创建关系表： teacher_salary(TNO, SAL)，其中 TNO 是教师工号（主键），SAL 是教师工资（类型 float）。

2. 定义一个 BEFORE 行级触发器，为关系表 teacher_salary 定义完整性规则：“表中出现的工号必须也出现在 teacher 表中，否则报错”。注：该规则实际上就是外键约束；MySQL 中可使用 SIGNAL 抛出错误；需要为 INSERT 和 UPDATE 分别定义触发器。请展示成功创建触发器和测试抛出错误信息的截图。
3. 定义一个 BEFORE 行级触发器，为关系表 teacher_salary 定义完整性规则：“Instructor/Associate Professor/Professor 的工资不能低于4000/7000/10000，且不能高于 7000/10000/13000，如果低于则改为 4000/7000/10000”，如果高于则改成 7000/10000/13000。
注意：需要为 INSERT 和 UPDATE 分别定义触发器。并检验触发器是否工作：为 teacher_salary 构造 INSERT 和 UPDATE 语句并激活所定义过的触发器，将过程截图展示。
4. 删除刚刚创建的所有触发器。

空值

1. 将 score 表中的 DB_Design 课程成绩设为空值，然后在 score 表查询选修此课学生学号和分数，并按分数升序展示。观察 NULL 在 MySQL 中的大小是怎样的？

开放题

1. 请自己设计两个题目并给出 sql 代码实现