实验一：交互式 SQL 语言

李建中 刘显敏

1. 实验目的

掌握 SQL 语言的基本命令，熟练使用 SQL 语言创建关系数据库、插入数据、更改数据、编写 SQL 查询。

2. 实验环境

Windows 操作系统、SQLite 软件（或 PostgreSQL）

3. 实验任务（以 SQLite 为例）

3.1 SQLite 基本命令

1) 双击打开 sqlite3.exe，该程序为 SQLite 数据库管理系统

2) 利用.help 查看 SQLite 支持的控制台系统命令
注意系统命令结尾处没有结束符“;”

3) 阅读.help 中对.databases 命令的说明，并查看输出结果

4) 阅读.help 中对.open 命令的说明，并使用该命令创建一个数据库（名字任意）后缀名统一为“.db3”（可以没有后缀名，但不推荐）

5) 再次运行.databases 命令，与步骤 3 的输出结果对比

6) 阅读.help 中对.tables 命令的说明，并使用该命令查看当前数据库的所有表

3.2 SQL 基本命令

7) 创建满足要求的关系表（使用 create table）
表一
表名：College（存储大学的信息）
属性：cName（字符串存储的大学名字），state（字符串格式的大学所在州），enrollment（整数形式的大学入学学费）
表二
表名：Student（存储学生的信息）
属性：sID（整数形式的学号），sName（字符串形式的学生名字），GPA（小数形式的成绩），sizeHS（整数形式的所在高中规模）
表三
表名：Apply（存储学生申请学校的信息）

属性：sID（整数形式的学号），cName（字符串形式的大学名字），major（字符串形式的专业名字），decision（字符串形式的申请结果）

8) 利用.tables 查看当前数据库中的表，对比步骤 6 中的运行结果

9) 利用如下命令，将存储在 txt 文件中的元组导入数据库的关系中
```
.separator ","
.import dbcollege.txt College
.import dbstudent.txt Student
.import dbapply.txt Apply
```

10) 查看导入后的 College、Student、Apply 表中的所有内容（使用 select）

11) 找出所有分数大于 3.6 的学生的 ID、名字和分数

12) 找出所有分数大于 3.6 的学生的 ID、名字

13) 查询所有申请信息中姓名、学校的对

14) 查询所有申请信息中姓名、学校的对，去除重复元组

15) 查询所有所处高中规模小于 1000、申请了斯坦福的计算机系的学生、其分数、及申请结果

16) 查询学生申请 CS 的学校中入学费用大于 20000 的学校名字（去除重复）

17) 查询学生申请学校的所有信息（输出结果按照学生分数降序排列）

18) 查询学生申请学校的所有信息（输出结果按照学生分数降序排列，对于相同分数的学生，按照入学费用升序排列）

19) 申请了与生物相关的专业的学生 ID 及专业信息

20) 输出 Student 和 College 的所有可能组合

21) 查询学生的所有信息，并为每一个学生计算一个新的分数 newGPA（按照 GPA 同 sizeHS 的关系换算出来的新分数）
newGPA=GPA×(sizeHS÷1000)

22) 查询 GPA 相同的学生的对

23) 查询 GPA 相同的学生的对（去除类似“张三，张三”这样的组合）

24) 输出大学名字同学生名字的集合

25) 输出一个关系，其中只有一个属性 name，name 存储了大学的名字或者学生的名字

26) 将上一个步骤中的结果排序（按照字母序）

27) 输出既申请了 CS 专业又申请了 EE 专业的学生的 ID

28) 输出申请了 CS 专业但是没有申请 EE 专业的学生的 ID

29) 查询申请了 CS 专业的学生的名字与 ID（用子查询实现）

30) 查询申请了 CS 专业的学生的名字

31) 不用集合差操作，查询申请了 CS 专业但是没有申请 EE 专业的学生的 ID

32) 查询最贵的大学的名字

33) 查询分数最高的学生的名字

34) 查询分数最高的同学申请的学校的名字及其所在州

35) 查询学生名字及其选取专业的名字的信息（用连接实现）

36) 查询来自人数少于 1000 的高中的、申请了斯坦福的计算机专业的学生的名字和 GPA（用连接实现，写出至少三种不同的写法）

37) 将三个表自然连接，并输出所有属性

38) 对比如下查询结果

```
select sName, sID, cName, major
from Student inner join Apply using(sID);
=====
select sName, sID, cName, major
from Student left outer join Apply using(sID);
=====
select sName, sID, cName, major
from Student natural left outer join Apply;
```

39) 输出所有学生的平均分

40) 申请计算机系的学生的最低分

41) 求申请计算机系学生的平均分

42) 求入学费用大于 15000 的学校的个数

43) 计算申请 Cornell 大学的学生数

44) 求申请 CS 的学生平均分比没有申请 CS 的学生平均分高出多少

45) 求每个学校被申请的次数

46) 求每个州大学收费的和

47) 求每个学校、每个专业申请的最高分、最低分

48) 求每个学生申请学校的数目

49) 验证如下查询的结果，并说明其语义

```
select Student.sID, count(distinct cName)
from Student, Apply
where Student.sID = Apply.sID
group by Student.sID
union
select sID, 0
from Student
where sID not in (select sID from Apply);
```

50) 求查找申请数少于 5 的大学