**西安电子科技大学**

**《数据库原理》 课程实验报告**

**实验名称 SQL实践**

姓名 学号

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1. 实验目的   熟练掌握 SQL 的数据定义功能，数据查询功能，数据操作功能，包括定义表，定义主码和外码，修改表(增加属性，删除属性，修改属性类型)，删除表，向表中添加数据记录，查询表中内容等。   1. 实验环境   MySQL 8.0   1. 实验基本原理及步骤   实验原理：SQL基本语法  实验步骤：根据实验题目要求，写出对应的SQL语句实现对应操作   1. 实验数据记录(SQL语句，运行过程及结果） 2. 实验任务：   使用 SQL 语句建立高考志愿数据库 college\_data(包括以下 3 个表)，添加主码信息(其中标红的属性为主码)。其中，sID, enrollment, sizeHS 类型是整型，GPA 类型是浮点数，decision 类型是文本，其他属性类型是变长字符串。  学院表 College (CName,state, enrollment) 属性列表示:学院名称，所在州，招生情况  学生表 Student (sID，sName，GPA，sizeHS) 属性列表示:学生学号，学生姓名，绩点，高中规模  申请表 Apply (sID，cName，major, decision) 属性列表示:学生学号，申请大学名称，申请专业，是否被录取  实验结果：   |  | | --- | | **CREATE** **TABLE** College(  cName **VARCHAR**(50) **NOT** **NULL**,  state **VARCHAR**(50) **NOT** **NULL**,  enrollment **INTEGER** **NOT** **NULL**,  **PRIMARY** **KEY** (cName)  );    **CREATE** **TABLE** Student(  sID **INTEGER** **NOT** **NULL**,  sName **VARCHAR**(50) **NOT** **NULL**,  GPA **FLOAT** **NOT** **NULL**,  sizeHS **INTEGER** **NOT** **NULL**,  **PRIMARY** **KEY** (sID)  );    **CREATE** **TABLE** Apply(  sID **INTEGER** **NOT** **NULL**,  cName **VARCHAR**(50) **NOT** **NULL**,  major **VARCHAR**(50) **NOT** **NULL**,  decision **VARCHAR**(50) **NOT** **NULL**,  **PRIMARY** **KEY** (sID, cName, major)  ); |   代码运行前：    代码运行后：           1. 实验任务：   对各表添加数据  实验结果：   |  | | --- | | **INSERT** **INTO** College **VALUES**('Berkeley', 'CA', 36000);  **INSERT** **INTO** College **VALUES**('Cornell', 'NY', 21000);  **INSERT** **INTO** College **VALUES**('MIT', 'MA', 10000);  **INSERT** **INTO** College **VALUES**('Stanford', 'CA', 15000);    **INSERT** **INTO** Student **VALUES**(123, 'Amy', 3.9, 1000);  **INSERT** **INTO** Student **VALUES**(234, 'Bob', 3.6, 1500);  **INSERT** **INTO** Student **VALUES**(345, 'Craig', 3.5, 500);  **INSERT** **INTO** Student **VALUES**(456, 'Doris', 3.9, 1000);  **INSERT** **INTO** Student **VALUES**(543, 'Craig', 3.4, 2000);  **INSERT** **INTO** Student **VALUES**(567, 'Edward', 2.9, 2000);  **INSERT** **INTO** Student **VALUES**(654, 'Amy', 3.9, 1000);  **INSERT** **INTO** Student **VALUES**(678, 'Fay', 3.8, 200);  **INSERT** **INTO** Student **VALUES**(765, 'Jay', 2.9, 1500);  **INSERT** **INTO** Student **VALUES**(789, 'Gary', 3.4, 800);  **INSERT** **INTO** Student **VALUES**(876, 'Irene', 3.9, 400);  **INSERT** **INTO** Student **VALUES**(987, 'Helen', 3.7, 800);    **INSERT** **INTO** Apply **VALUES**(123, 'Berkeley', 'CS', 'Y');  **INSERT** **INTO** Apply **VALUES**(123, 'Cornell', 'EE', 'Y');  **INSERT** **INTO** Apply **VALUES**(123, 'Stanford', 'CS', 'Y');  **INSERT** **INTO** Apply **VALUES**(123, 'Stanford', 'EE', 'N');  **INSERT** **INTO** Apply **VALUES**(234, 'Berkeley', 'biology', 'N');  **INSERT** **INTO** Apply **VALUES**(345, 'Cornell', 'bioengineering', 'N');  **INSERT** **INTO** Apply **VALUES**(345, 'Cornell', 'CS', 'Y');  **INSERT** **INTO** Apply **VALUES**(345, 'Cornell', 'EE', 'N');  **INSERT** **INTO** Apply **VALUES**(345, 'MIT', 'bioengineering', 'Y');  **INSERT** **INTO** Apply **VALUES**(543, 'MIT', 'CS', 'N');  **INSERT** **INTO** Apply **VALUES**(678, 'Stanford', 'history', 'Y');  **INSERT** **INTO** Apply **VALUES**(765, 'Cornell', 'history', 'N');  **INSERT** **INTO** Apply **VALUES**(765, 'Cornell', 'psychology', 'Y');  **INSERT** **INTO** Apply **VALUES**(765, 'Stanford', 'history', 'Y');  **INSERT** **INTO** Apply **VALUES**(876, 'MIT', 'biology', 'Y');  **INSERT** **INTO** Apply **VALUES**(876, 'MIT', 'marine biology', 'N');  **INSERT** **INTO** Apply **VALUES**(876, 'Stanford', 'CS', 'N');  **INSERT** **INTO** Apply **VALUES**(987, 'Berkeley', 'CS', 'Y');  **INSERT** **INTO** Apply **VALUES**(987, 'Stanford', 'CS', 'Y'); |   数据插入前：    数据插入后：         1. 实验任务：进行查询操作   (1)查询GPA大于3.6的学生学号和姓名   |  | | --- | | **SELECT** **DISTINCT** sID, sName **FROM** Student **WHERE** GPA > 3.6; |     (2)查询所有学生姓名及申请专业   |  | | --- | | **SELECT** **DISTINCT** sName, major **FROM** Student **JOIN** Apply  **WHERE** Student.sID = Apply.sID; |     (3)查询所在高中规模不到1000，申请了斯坦福大学CS专业的学生姓名、GPA和申请结果   |  | | --- | | **SELECT** **DISTINCT** sName, GPA, decision **FROM** Student **JOIN** Apply  **WHERE** Student.sID = Apply.sID  **AND** sizeHS < 1000  **AND** cName = 'Stanford'  **AND** major = 'CS'; |     (4)查询具有CS专业、规模在2000人以上的学校名称   |  | | --- | | **SELECT** **DISTINCT** cName **FROM** College  **WHERE** enrollment >= 20000  **AND** cName **IN** (**SELECT** cName **FROM** Apply **WHERE** major = 'CS'); |     (5)查询学生学号、姓名、绩点、申请学校，申请学校规模(按照申请学校分组，组内按绩点降序、学校规模升序排序)   |  | | --- | | **SELECT** **DISTINCT** Student.sID, Student.sName, Student.GPA, Apply.cName, Student.sizeHS  **FROM** Student  **JOIN** Apply **ON** Student.sID = Apply.sID  **GROUP** **BY** Apply.cName, Student.sID, Student.sName, Student.GPA, Student.sizeHS  **ORDER** **BY** Apply.cName, Student.GPA **DESC**, Student.sizeHS; |     (6)查询申请专业含有‘bio’字符的学生学号和申请专业   |  | | --- | | **SELECT** sID, major **FROM** Apply **WHERE** major **LIKE** '%bio%'; |     (7)查询具有相同绩点的学生信息对，输出他们的学号、姓名、GPA(S1.sID, S1.sName, S1.GPA, S2.sID, S2.sName, S2.GPA )   |  | | --- | | **SELECT** **DISTINCT** s1.sID, s1.sName, s1.GPA, s2.sID, s2.sName, s2.GPA  **FROM** Student s1 **JOIN** Student s2  **WHERE** s1.GPA = s2.GPA **AND** s1.sID < s2.sID; |     (8)查询同时申请了CS和EE专业的学生学号   |  | | --- | | **SELECT** **DISTINCT** sID **FROM** Apply **AS** A1  **WHERE** major = 'CS' **AND** **EXISTS**(  **SELECT** \* **FROM** Apply **AS** A2  **WHERE** A1.sID = A2.sID **AND** A2.major = 'EE'  ); |     (9)查询申请了CS专业但是没有申请EE专业的学生学号的姓名   |  | | --- | | **SELECT** **DISTINCT** sID, sName **FROM** Student  **WHERE** **EXISTS**(  **SELECT** \* **FROM** Apply  **WHERE** Student.sID = Apply.sID **AND** major = 'CS'  )**AND** **NOT** **EXISTS**(  **SELECT** \* **FROM** Apply  **WHERE** Student.sID = Apply.sID **AND** major = 'EE'  ); |     (10)查询申请了CS专业的学生学号的姓名   |  | | --- | | **SELECT** **DISTINCT** sID, sName **FROM** Student  **WHERE** **EXISTS**(  **SELECT** \* **FROM** Apply  **WHERE** Student.sID = Apply.sID **AND** major = 'CS'  ); |     (11)查询所在州有其他学校的学校名称和所在州   |  | | --- | | **SELECT** **DISTINCT** cName, state **FROM** College  **WHERE** state **IN**(  **SELECT** state **FROM** College  **GROUP** **BY** state **HAVING** **COUNT**(\*) > 1  ); |     (12)查询规模最大的学校的名称   |  | | --- | | **SELECT** cName **FROM** College  **WHERE** enrollment = (  **SELECT** **MAX**(enrollment) **FROM** College  ); |     (13)查询绩点最高的学生姓名和GPA（不用子查询）   |  | | --- | | **SELECT** sName, GPA **FROM** Student  **ORDER** **BY** GPA **DESC**  **LIMIT** 1; |     (14)查询不是来自规模最小的高中的学生学号、姓名和高中规模   |  | | --- | | **SELECT** **DISTINCT** sID, sName, sizeHS **FROM** Student  **WHERE** sizeHS <> (  **SELECT** **MIN**(sizeHS) **FROM** Student  ); |     (15)查询每个学校的学校名称、所在州、以及申请者中GPA最高的学生的绩点   |  | | --- | | **SELECT** C.cName, C.state, **MAX**(S.GPA) **AS** max\_GPA  **FROM** College C  **JOIN** Apply A **ON** C.cName = A.cName  **JOIN** Student S **ON** A.sID = S.sID  **GROUP** **BY** C.cName, C.state |     (16)查询学生姓名和他们申请的专业   |  | | --- | | **SELECT** **DISTINCT** S.sName, A.major **FROM** Student S  **JOIN** Apply A **ON** S.sID = A.sID; |     (17)查询学生姓名和成绩绩点（条件是申请了Stanford大学的CS专业，所在高中人数少于1000）   |  | | --- | | **SELECT** **DISTINCT** sName, GPA **FROM** Student  **WHERE** sizeHS < 1000  **AND** **EXISTS**(  **SELECT** \* **FROM** Apply  **WHERE** major = 'CS' **AND** Student.sID = Apply.sID  ); |     (18)查询申请CS专业的学生的最低成绩绩点   |  | | --- | | **SELECT** **MIN**(GPA) **AS** min\_GPA **FROM** Student S  **JOIN** Apply A **ON** S.sID = A.sID  **WHERE** A.major = 'CS'; |     (19)查询申请CS专业的学生的平均成绩绩点（不管他们申请多少次）   |  | | --- | | **SELECT** avg(**DISTINCT** GPA) **AS** avg\_GPA **FROM** Student S  **JOIN** Apply A **ON** S.sID = A.sID  **WHERE** A.major = 'CS'; |     (20)查询申请Cornell大学的申请数（申请不同专业的同一个学生算一次申请）   |  | | --- | | **SELECT** **COUNT**(**DISTINCT** A.sID) **AS** application\_num  **FROM** Apply A **JOIN** College C **ON** A.cName = C.cName  **WHERE** C.cName = 'Cornell'; |     (21)查询学生信息（条件是与他具有相同GPA的学生人数等于与他所在高中规模相同的学生人数）   |  | | --- | | **SELECT** S.sID, S.sName, S.GPA, S.sizeHS **FROM** Student S  **JOIN**(  **SELECT** GPA, **COUNT**(**DISTINCT** sID) **AS** cnt1  **FROM** Student  **GROUP** **BY** GPA  )T1 **ON** S.GPA = T1.GPA  **JOIN**(  **SELECT** sizeHS, **COUNT**(**DISTINCT** sID) **AS** cnt2  **FROM** Student  **GROUP** **BY** sizeHS  )T2 **ON** S.sizeHS = T2.sizeHS  **WHERE** T1.cnt1 = T2.cnt2; |     (22)查询申请CS专业的学生的平均绩点和申请非CS专业的学生的平均绩点的差值   |  | | --- | | **SELECT** avg(**CASE** **WHEN** major = 'CS' **THEN** GPA **ELSE** **NULL** **END**) **AS** avg\_cs,  avg(**CASE** **WHEN** major <> 'CS' **THEN** GPA **ELSE** **NULL** **END**) **AS** avg\_no\_cs,  avg(**CASE** **WHEN** major = 'CS' **THEN** GPA **ELSE** **NULL** **END**)-  avg(**CASE** **WHEN** major <> 'CS' **THEN** GPA **ELSE** **NULL** **END**) **AS** diff  **FROM** Apply  **JOIN** Student **ON** Apply.sID = Student.sID; |     (23)查询每个大学的申请人数（同一个学生申请不同专业按照不同申请对待）   |  | | --- | | **SELECT** cName, **COUNT**(sID) **AS** application\_num  **FROM** Apply  **GROUP** **BY** cName; |     (24)查询所有大学每个专业申请人的最低GPA和最高GPA的最大差值   |  | | --- | | **SELECT** A.cName, A.major, **TRUNCATE**((T.maxGPA - T.minGPA),1) **AS** maxDiffGPA  **FROM** Apply A **JOIN**(  **SELECT** cName, major, **MIN**(GPA) **AS** minGPA, **MAX**(GPA) **AS** maxGPA  **FROM** Student S **JOIN** Apply A **ON** S.sID = A.sID  **GROUP** **BY** cName, major  ) T **ON** A.cName = T.cName **AND** A.major = T.major  **GROUP** **BY** A.cName, A.major  **ORDER** **BY** A.cName; |     (25)查询每个学生申请的学校个数（包括没有申请任何学习哦啊的学生，输出学生学号和申请学校个数）   |  | | --- | | **SELECT** S.sID, **COUNT**(A.cName) **AS** application\_num  **FROM** Student S **LEFT** **JOIN** Apply A **ON** S.sID = A.sID  **GROUP** **BY** S.sID; |     (26)查询申请者少于5的大学名称（申请者，不是申请数）   |  | | --- | | **SELECT** A.cName **FROM** Apply A  **GROUP** **BY** A.cName  **HAVING** **COUNT**(**DISTINCT** A.sID) < 5; |     (27)查询申请者最高GPA低于所有学生平均GPA的专业   |  | | --- | | **SELECT** A.major  **FROM**(  **SELECT** avg(GPA) **AS** avg\_GPA  **FROM** Student  ) **AS** S,(  **SELECT** major, **MAX**(GPA) **AS** max\_GPA  **FROM** Apply, Student  **WHERE** Apply.sID = Student.sID  **GROUP** **BY** major  ) **AS** A  **WHERE** A.max\_GPA < S.avg\_GPA; |      1. 实验任务：进行数据修改操作   (1) 在 college 中插入一条数据，学校为'Carnegie Mellon', 所 在州'PA', 入学人数 11500;   |  | | --- | | **INSERT** **INTO** College **VALUES**('Carnegie Mellon', 'PA', 11500); |   插入前：    插入后：    (2)在Apply表中插入数据:没有申请任何学校的学生将申请 Carnegie Mellon 大学的 CS 专业;  将申请信息插入 apply 表中(decision 设置为空值)   |  | | --- | | **INSERT** **INTO** Apply(sID, cName, major, decision)  **SELECT** S.sID, 'Carnegie Mellon', 'CS', **NULL**  **FROM** Student S  **WHERE** **NOT** **EXISTS**(  **SELECT** \*  **FROM** Apply A  **WHERE** A.sID = S.sID  ); |   修改前：    修改后：    (3) 在 Apply 表中插入数据:允许所有申请 EE 专业被拒绝的学生 进入卡内基梅隆大学 EE 专业(decision 值为‘Y’);   |  | | --- | | **INSERT** **INTO** Apply(sID, cName, major, decision)  **SELECT** sID, 'Carnegie Mellon', 'EE', 'Y'  **FROM** Apply  **WHERE** major = 'EE' **AND** decision = 'N'; |   修改前：    修改后：    (4)从学生表中删除申请了2个以上专业的学生信息   |  | | --- | | **DELETE** **FROM** Student  **WHERE** sID **IN**(  **SELECT** sID **FROM** Apply  **GROUP** **BY** sID **HAVING** **COUNT**(**DISTINCT** major) > 2  ); |   修改前：    修改后：    (5)从college表中删除没有CS专业申请记录的大学   |  | | --- | | **DELETE** **FROM** College  **WHERE** **NOT** **EXISTS**(  **SELECT** \* **FROM** Apply  **WHERE** Apply.cName = College.cName  **AND** Apply.major = 'CS'  ); |   实验前：    实验后：    (6)更新Apply表：录取申请Carnegie Mellon 大学、GPA 小于 3.6 的学生，录取专业为 economics;   |  | | --- | | **UPDATE** Apply  **SET** decision = 'Y', major = 'economics'  **WHERE** cName = 'Carnegie Mellon'  **AND** sID **IN**(**SELECT** sID  **FROM** Student **WHERE** GPA < 3.6  ); |   修改前：    修改后：    (7) 更新 Apply 表:将申请 EE 专业具有最高 GPA 学生的专业更新为 CSE。   |  | | --- | | **SET** @maxGPA := (  **SELECT** **MAX**(s.GPA)  **FROM** Apply a  **JOIN** Student s **ON** a.sID = s.sID  **WHERE** a.major = 'EE'  );    **UPDATE** Apply a  **JOIN** Student s **ON** a.sID = s.sID  **SET** a.major = 'CSE'  **WHERE** a.major = 'EE' **AND** s.GPA = @maxGPA; |   修改前：    修改后：    问题记录：   1. 在3.5中，报错：未指定表中的列。   发现原因是对于sID这个变量，我未指定是哪个表中的。因此相同列名必须指定是哪个表中的列，否则会引起歧义   1. 在3.13,要求不使用子查询，通过资料查询可以先排序，后使用limit语句。   但在实际操作过程中，发现当多个人的GPA都为maxGPA时，无法无法判断到底有多少人，因此无法正确输出所有GPA为maxGPA的学生信息。   1. 在3.21中,由于使用cnt时未group by，报错In aggregated query without GROUP BY   查询资料发现，当聚集函数和非聚集函数出现在一起时，必须要将非聚集函数进行group by   1. 在3.24，mysql进行减法以后，无法得到完全正确的结果，会出现好几位小数   对结果进行四舍五入，取一位小数即可   1. 在4.7中,报错 You can't specify target table 'Apply' for update in FROM clause。   发现不能先select出同一表中的某些值，再update这个表(在同一语句中)。于是首先通过sql变量求出申请 EE 专业的学生的最高 GPA，保存在@maxGPA中，再进行Apply表的更新。 |