



東北大學 悉尼智能科技学院
Sydney Smart Technology College
NORTHEASTERN UNIVERSITY

Database System

Concepts

(数据库系统原理)

Experiment Report

学 院	悉尼智能科技学院
专业名称	计算机科学与技术
班级序号	CST2201
学 号	202219102
学生姓名	况小媛颖

2024 年 05 月 17 日

目 录

1	数据库的建立	1
1.1	实验内容	1
2	创建并管理表	4
2.1	实验内容	4
3	数据更新	9
3.1	实验内容	9
4	数据查询	12
4.1	实验内容	12
5	索引与视图	18
5.1	实验内容	18

1 数据库的建立

1.1 实验内容

1、在查询分析器中创建一个数据库，要求如下：

- (1) 数据库名称 Test2。
- (2) 主要数据文件：逻辑文件名为 Test2_data1，物理文件名为 Test2_data1.mdf，初始容量为 10MB，最大容量为 100MB，增幅为 10MB。
- (3) 次要数据文件：逻辑文件名为 Test2_data2，物理文件名为 Test2_data2.ndf，初始容量为 10MB，最大容量为 10MB，增幅为 15%。
- (4) 事务日志文件：逻辑文件名为 Test2_log1，物理文件名为 Test2_log1.ldf，初始容量为 10MB，最大容量为 50MB，增幅为 20MB。

解答：

```
CREATE DATABASE Test2
```

```
ON
```

```
PRIMARY
```

```
(
```

```
    NAME = 'Test2_data1',
```

```
    FILENAME = 'Test2_data1.mdf',
```

```
    SIZE = 10MB,
```

```
    MAXSIZE = 100MB,
```

```
    FILEGROWTH = 10MB
```

```
),
```

```
FILEGROUP Test2_Secondary
```

```
(
```

```
    NAME = 'Test2_data2',
```

```
FILENAME = 'Test2_data2.ndf',  
SIZE = 10MB,  
MAXSIZE = 10MB,  
FILEGROWTH = 15%  
)  
LOG ON  
(  
    NAME = 'Test2_log1',  
    FILENAME = 'Test2_log1.ldf',  
    SIZE = 10MB,  
    MAXSIZE = 50MB,  
    FILEGROWTH = 20MB  
);
```

2、在查询分析器中按照下列要求修改第 3 题中创建的数据库 test2

- (1) 主要数据文件的容量为 50MB，最大容量为 200MB，增幅为 20MB。
- (2) 次要数据文件的容量为 50MB，最大容量为 300MB，增幅为 20MB。
- (3) 事务日志文件的容量为 30MB，最大容量为 100MB，增幅为 10%。

解答：

(1)

```
ALTER DATABASE Test2
```

```
MODIFY FILE
```

```
(NAME = Test2_data1, SIZE = 50MB, MAXSIZE = 200MB, FILEGROWTH = 20MB);
```

(2)

```
ALTER DATABASE Test2
```

```
MODIFY FILE
```

```
(NAME = Test2_data2, SIZE = 50MB, MAXSIZE = 300MB, FILEGROWTH = 20MB);
```

(3)

ALTER DATABASE Test2

MODIFY FILE

(NAME = Test2_log1, SIZE = 30MB, MAXSIZE = 100MB, FILEGROWTH = 10%);

3、数据库更名：把 test1 数据库更名为 new_test1

解答：

ALTER DATABASE test1 MODIFY NAME = new_test1;

4、在企业管理器中删除 new_test1 数据库，在查询分析器中删除 test2 数据库。

解答：

DROP DATABASE new_test1;

DROP DATABASE Test2;

2 创建并管理表

2.1 实验内容

1、创建数据库 studentInfo，包含如下表，创建这些表并按要求定义约束：

表 2.1 student（学生表）结构

字段名	说明	数据类型	约束说明
Student_id	学号	字符串，长度为 10	主键
Student_name	姓名	字符串，长度为 10	非空
sex	性别	字符串，长度为 1	非空值，取 ‘F’ 或 ‘M’
age	年龄	整数	允许空值
department	所在系名	字符串，长度为 15	默认值为 ‘computer’

表 2.2 course（课程表）结构

字段名	说明	数据类型	约束说明
Course_id	课程号	字符串，长度为 6	主键
Course_name	课程名	字符串，长度为 20	非空值
PreCouId	先修课程号	字符串，长度为 6	允许空值
Credits	学分	十进制数，精度 3，小数位 1	非空值

表 2.3 score（选课表）结构

字段名	说明	数据类型	约束说明
Student_id	学号	字符串，长度为 10	外键，参照 student 的主键
Course_id	课程号	字符串，长度为 6	外键，参照 course 的主键
Grade	成绩	十进制数，精度 3，小数位 1	允许空值，要求 >0 <100
联合主键：（Student_id，Course_id）			

以下为各个表的数据：

Students 表数据

Student_id	Student_name	sex	age	department
------------	--------------	-----	-----	------------

20010101	Jone	M	19	Computer
20010102	Sue	F	20	Computer
20010103	Smith	M	19	Math
20030101	Allen	M	18	Automation
20030102	deepa	F	21	Art
20010104	Stefen	F	20	Computer

Course 表数据

Course_id	Course_name	PreCouId	Credits
C1	English		4
C2	Math	C1	2
C3	Cprogram	C2	2
C4	database	C2	2

Score 表数据

Student_id	Course_id	Grade
20010101	C1	90
20010102	C1	87
20010103	C1	88
20010102	C2	90
20010104	C2	94
20010102	C3	62
20030101	C3	80
20010103	C4	77

解答:

```
CREATE DATABASE studentInfo;
```

```
USE studentInfo;
```

```
CREATE TABLE student (  
    Student_id VARCHAR (10) PRIMARY KEY ,  
    Student_name VARCHAR (10) NOT NULL ,  
    sex CHAR(1) NOT NULL CHECK (sex IN ('F', 'M')),  
    age INT,  
    department VARCHAR(15) DEFAULT 'computer'  
) ;
```

```
CREATE TABLE (  
    Course_id VARCHAR(6) PRIMARY KEY,  
    Course_name VARCHAR(20) NOT NULL,  
    PreCould VARCHAR(6),  
    Credits DECIMAL(3,1) NOT NULL,  
    FOREIGN KEY (PreCould) REFERENCES course(Course_id)  
);
```

```
CREATE TABLE score (  
    Student_id VARCHAR(10),  
    Course_id VARCHAR(6),  
    Grade DECIMAL(3,1),  
    PRIMARY KEY (Student_id, Course_id),  
    FOREIGN KEY (Student_id) REFERENCES student(Student_id),  
    FOREIGN KEY (Course_id) REFERENCES course(Course_id),  
    CHECK (Grade > 0 AND Grade < 100)
```


);

INSERT INTO student (Student_id, Student_name, sex, age, department) VALUES

('20010101', 'Jone', 'M', 19, 'Computer'),

('20010102', 'Sue', 'F', 20, 'Computer'),

('20010103', 'Smith', 'M', 19, 'Math'),

('20030101', 'Allen', 'M', 18, 'Automation'),

('20030102', 'deepa', 'F', 21, 'Art'),

('20010104', 'Stefen', 'F', 20, 'Computer');

INSERT INTO course (Course_id, Course_name, PreCouId, Credits) VALUES

('C1', 'English', NULL, 4),

('C2', 'Math', 'C1', 2),

('C3', 'Cprogram', 'C2', 2),

('C4', 'database', 'C2', 2);

INSERT INTO score (Student_id, Course_id, Grade) VALUES

('20010101', 'C1', 90),

('20010102', 'C1', 87),

('20010103', 'C1', 88),

('20010102', 'C2', 90),

('20010104', 'C2', 94),

('20010102', 'C3', 62),

('20030101', 'C3', 80),

('20010103', 'C4', 77);

2、增加、修改、删除字段，要求：

- (1) 为表 student 增加一个 memo（备注）字段，类型为 varchar（200）。
- (2) 将 memo 字段的数据类型更改为 varchar（300）。
- (3) 删除 memo 字段

解答：

(1)

```
ALTER TABLE student
```

```
ADD COLUMN memo VARCHAR(200);
```

(2)

```
ALTER TABLE student
```

```
ALTER COLUMN memo varchar(300);
```

(3)

```
ALTER TABLE student
```

```
DROP COLUMN memo;
```

4. 分别使用企业管理器和查询分析器删除表

解答：

```
DROP TABLE table_name;
```

3 数据更新

3.1 实验内容

在已经建立的 studentinfo 数据库和 3 个 students、courses、score 基础上完成下列操作：

- 1、向 students 表添加一个学生记录，学号为 20010112，性别为男，姓名为 stefen，年龄 25 岁，所在系为艺术系 art。
- 2、向 score 表添加一个选课记录，学生学号为 20010112，所选课程号为 C2。
- 3、建立表 tempstudent，结构与 students 结构相同，其记录均从 student 表获取
- 4、将所有学生的成绩加 5 分
- 5、将姓名为 sue 的学生所在系改为电子信息系
- 6、将选课为 database 的学生成绩加 10 分
- 7、删除所有成绩为空的选修记录（delete form 表名 where 属性列 is null）
- 8、删除学生姓名为 deepa 的学生记录
- 9、删除计算机系选修成绩不及格的学生的选修记录。

解答：

1、

INSERT

INTO student (Student_id, Student_name, sex, age, department)

VALUES ('20010112', 'Stefen', 'M', 25, 'Art');

2、

INSERT

INTO score (Student_id, Course_id)

VALUES ('20010112', 'C2');

3、

```
CREATE TABLE tempstudent
```

```
AS
```

```
SELECT * FROM student;
```

4、

```
UPDATE score
```

```
SET Grade = Grade + 5;
```

5、

```
UPDATE student
```

```
SET department = 'Electronic Information'
```

```
WHERE Student_name = 'Sue';
```

6、

```
UPDATE score
```

```
SET Grade = Grade + 10
```

```
WHERE Course_id = 'C4'
```

7、

```
DELETE
```

```
FROM score
```

```
WHERE Grade IS NULL;
```

8、

```
DELETE
```

```
FROM student
```

WHERE Student_name = 'deepa';

9、

DELETE

FROM score

WHERE Student_id IN

(SELECT Student_id FROM student WHERE department = 'computer')

AND Grade < 60;

4 数据查询

4.1 实验内容

在已经建立好的 studentInfo 数据库中使用已存在的 3 个数据表 student、course、score 基础上完成下列查询实验：

一、简单查询实验

- (1) 查询全体学生的学号、姓名、所在系，并为结果集的各列设置中文名称。
- (2) 查询全体学生的选课情况，并为所有成绩加 5 分。
- (3) 显示所有选课学生的学号，去掉重复行。
- (4) 查询选课成绩大于 80 分的学生。
- (5) 查询年龄在 20 到 30 之间的学生学号，姓名，所在系
- (6) 查询数学系、计算机系、艺术系的学生学号，姓名。
- (7) 查询姓名第二个字符为 u 并且只有 3 个字符的学生学号，姓名
- (8) 查询所有以 S 开头的学生。
- (9) 查询姓名不以 S、D、或 J 开头的学生
- (10) 查询没有考试成绩的学生和相应课程号（成绩值为空）is null
- (11) 求年龄大于 19 岁的学生的总人数
- (12) 分别求选修了 c 语言课程的学生平均成绩、最高分、最低分学生。
- (13) 求学号为 20010102 的学生总成绩
- (14) 求每个选课学生的学号，姓名，总成绩
- (15) 求课程号及相应课程的所有的选课人数
- (16) 查询选修了 3 门以上课程的学生姓名学号

解答：

1、

```
SELECT Student_id AS "学号", Student_name AS "姓名", department AS "所在系"
FROM student;
```

2、

UPDATE score

SET Grade = Grade + 5;

3、

SELECT DISTINCT Student_id

FROM score;

4、

SELECT *

FROM score

WHERE Grade > 80;

5、

SELECT Student_id, Student_name, department

FROM student

WHERE age BETWEEN 20 AND 30;

6、

SELECT Student_id, Student_name

FROM student

WHERE department IN ('Math', 'Computer', 'Art');

7、

SELECT Student_id, Student_name

FROM student

WHERE Student_name LIKE '_u_' AND LENGTH(Student_name) = 3;

8、

```
SELECT *  
FROM student  
WHERE Student_name LIKE 'S%';
```

9、

```
SELECT *  
FROM student  
WHERE Student_name NOT LIKE '[SDJ]%';
```

10、

```
SELECT Student_id, Course_id  
FROM score  
WHERE Grade IS NULL;
```

11、

```
SELECT COUNT(*) AS "总人数"  
FROM student  
WHERE age > 19;
```

12、

```
SELECT AVG(Grade) AS "平均成绩", MAX(Grade) AS "最高分", MIN(Grade) AS "最低分"  
FROM score  
WHERE Course_id = 'C1';
```

13、

```
SELECT SUM(Grade) AS "总成绩"  
FROM score
```


WHERE Student_id = '20010102';

14、

SELECT s.Student_id, s.Student_name, SUM(sc.Grade) AS "总成绩"

FROM student s

JOIN score sc ON s.Student_id = sc.Student_id

GROUP BY s.Student_id, s.Student_name;

15、

SELECT Course_id, COUNT(*) AS "选课人数"

FROM score

GROUP BY Course_id;

16、

SELECT s.Student_id, s.Student_name

FROM student s

JOIN (SELECT Student_id, COUNT(Course_id) AS course_count FROM score GROUP BY Student_id) sc

ON s.Student_id = sc.Student_id

WHERE sc.course_count >= 3;

二、多表连接查询

- (1) 查询每个学生基本信息及选课情况
- (2) 查询每个学生学号姓名及选修的课程名、成绩
- (3) 求计算机系选修课程超过 2 门课的学生学号姓名、平均成绩并按平均成绩降序排列
- (4) 查询与 sue 在同一个系学习的所有学生的学号姓名

(5) 查询所有学生的选课情况, 要求包括所有选修了课程的学生和没有选课的学生, 显示他们的姓名学号课程号和成绩 (如果有)

解答:

1、

```
SELECT s.*, sc.Course_id  
FROM student s  
LEFT JOIN score sc ON s.Student_id = sc.Student_id;
```

2、

```
SELECT s.Student_id, s.Student_name, c.Course_name, sc.Grade  
FROM student s  
LEFT JOIN score sc ON s.Student_id = sc.Student_id  
LEFT JOIN course c ON sc.Course_id = c.Course_id;
```

3、

```
SELECT s.Student_id, s.Student_name, AVG(sc.Grade) AS "平均成绩"  
FROM student s  
JOIN score sc ON s.Student_id = sc.Student_id  
WHERE s.department = 'Computer'  
GROUP BY s.Student_id, s.Student_name  
HAVING COUNT(sc.Course_id) > 2  
ORDER BY AVG(sc.Grade) DESC;
```

4、

```
SELECT s.Student_id, s.Student_name  
FROM student s  
WHERE s.department = (SELECT department FROM student WHERE Student_name =  
'Sue');
```

5、

```
SELECT s.Student_id, s.Student_name, sc.Course_id, sc.Grade  
FROM student s  
LEFT JOIN score sc ON s.Student_id = sc.Student_id;
```

5 索引与视图

5.1 实验内容

5.1 在已经建立的 studentInfo 数据库的 3 个表基础上，完成下列操作：

- (1) 建立数学系的学生视图；
- (2) 建立计算机系选修了课程名为 database 的学生的视图，视图名为 compStudentview，该视图的列名为学号、姓名、成绩
- (3) . 创建一个名为 studentSumview 的视图，包含所有学生学号和总成绩
- (4) 建立一个计算机系学生选修了课程名为 database 并且成绩大于 80 分的学生视图，视图名为 CompsutdentView1，视图的列为学号姓名成绩。
- (5) 使用 sql 语句删除 compsutdentview1 视图。

解答：

(1)

```
CREATE VIEW MathStudentView AS  
SELECT * FROM student  
WHERE department = 'Math';
```

(2)

```
CREATE VIEW compStudentView AS  
SELECT s.Student_id AS "学号", s.Student_name AS "姓名", sc.Grade AS "成绩"  
FROM student s  
JOIN score sc ON s.Student_id = sc.Student_id  
JOIN course c ON sc.Course_id = c.Course_id  
WHERE s.department = 'Computer' AND c.Course_name = 'database';
```

(3)

```
CREATE VIEW studentSumView AS
SELECT Student_id, SUM(Grade) AS "总成绩"
FROM score
GROUP BY Student_id;
```

(4)

```
CREATE VIEW CompStudentView1 AS
SELECT s.Student_id AS "学号", s.Student_name AS "姓名", sc.Grade AS "成绩"
FROM student s
JOIN score sc ON s.Student_id = sc.Student_id
JOIN course c ON sc.Course_id = c.Course_id
WHERE s.department = 'Computer' AND c.Course_name = 'database' AND sc.Grade > 80;
```

(5)

```
DROP VIEW CompStudentView1;
```