

Database System Concepts

数据库系统概论

实验指导书

(SQL Server 版)

隆承志 编著

计算机科学与工程学院



华南理工大学

South China University of Technology

目录

概述.....	1
实验 1：服务器的启动、暂停和停止.....	2
实验名称和性质	2
实验目的.....	2
实验内容.....	2
实验 2：创建和管理数据库.....	4
实验名称和性质	4
实验目的.....	4
实验内容.....	4
实验 3：表建立与数据插入.....	10
实验名称和性质	10
实验目的.....	10
实验内容.....	10
验证性实验.....	10
设计性实验.....	12
实验 4：单表查询	14
实验名称和性质	14
实验目的.....	14
知识准备	14
实验内容.....	15
验证性实验.....	15
设计性实验.....	17
实验 5：多表查询	18
实验名称和性质	18
实验目的.....	18
知识准备	18
实验内容.....	19
验证性实验.....	19
设计性实验.....	22
实验 6：数据操作与索引.....	23
实验名称和性质	23
实验目的.....	23
知识准备	23
实验内容.....	24
验证性实验.....	24
设计性实验.....	27
实验 7：数据完整性约束.....	28

实验名称和性质	28
实验目的	28
知识准备	28
实验内容	29
验证性实验	29
设计性实验	31
实验 8: SQL 编程及存储过程	33
实验名称和性质	33
实验目的	33
知识准备	33
实验内容	35
验证性实验	35
设计性实验	37
实验 9: 事务处理和触发器	38
实验名称和性质	38
实验目的	38
知识准备	38
实验内容	39
验证性实验	39
设计性实验	42
实验 10: 数据库安全与数据库恢复	43
实验名称和性质	43
实验目的	43
实验内容	43
验证性实验	43
实验 11: 数据库设计	46
实验名称和性质	46
实验目的	46
知识准备	46
实验内容	47
综合性实验	47

概述

课程名称：数据库系统概论实验

英文名称：Database System Concepts

总学时：64 **学 分：**3 **实验学时：**16 **上机学时：**16

课程类别：实验

课程性质：必修

适用专业：计算机各类专业

授课实验室：

实验（上机）教学目的与基本要求：

《数据库系统概论》是计算机学科各专业的一门重要专业基础课程，是计算机科学与技术专业、网络工程四年制本科的必修课程。本课程主要学习关系数据库系统，并以大型关系数据库管理系统 sql server 为实例。通过本课程的学习，了解数据库系统的基本概念、基本理论，掌握关系数据库的相关知识和技术，初步掌握数据库设计方法，并能用数据库系统建立数据库及简单的应用；通过实际的上机操作，熟悉 sql server 操作环境，掌握关系数据库标准语言 SQL、规范化理论。能采用高级语言进行简单应用系统的实现，能进行用户的授权与管理。

实验（上机）教学方式与考核方式

- 教学方式：上机操作
- 考核方式：上机操作 + 实验报告

实验（上机）指导书和参考书：参见自编的《数据库系统概论实验指导书》

主要仪器设备：安装数据库的个人电脑或者实验室电脑。

实验 1：服务器的启动、暂停和停止

实验名称和性质

实验名称	服务器的启动、暂停和停止
实验学时	1
实验性质	<input checked="" type="checkbox"/> 验证 <input type="checkbox"/> 综合 <input checked="" type="checkbox"/> 设计
必做/选做	<input checked="" type="checkbox"/> 必做 <input type="checkbox"/> 选做

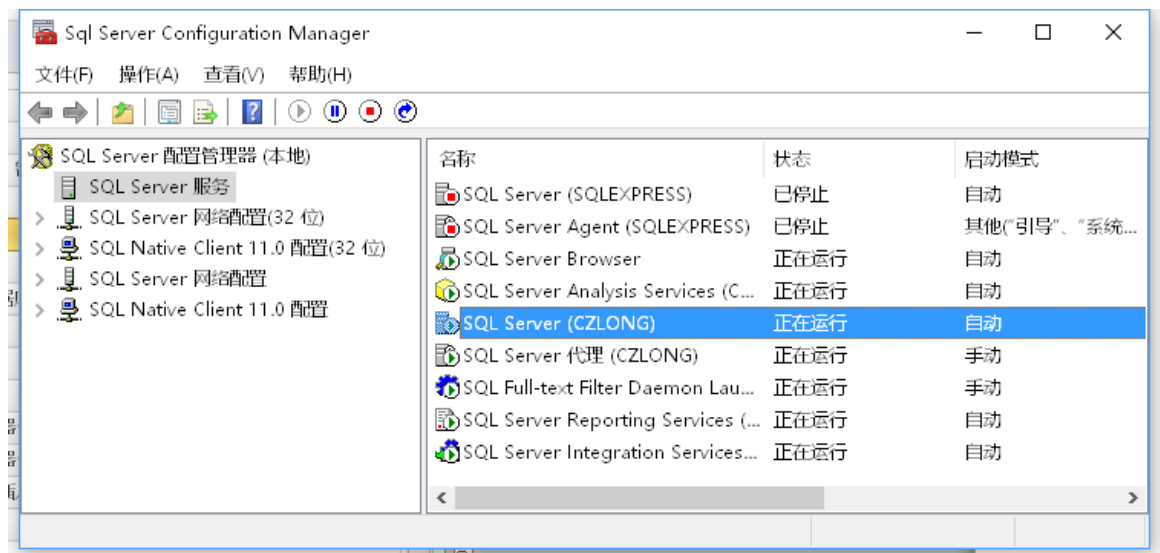
实验目的

- 1) 熟悉 SQL Server 2012 （或者更高版本）配置管理器。
- 2) 掌握服务器的启动、暂停、停止方法。

实验内容

启动，暂停和停止服务的方法很多，这里主要介绍 SQL Server 配置管理器完成这些操作，其操作步骤如下：

单击“开始” --- “Microsoft SQL Server 2012”，选择“SQL Server 配置管理器”，如下图：



SQL Server 服务中，在右边的对话框里可以看到本地所有的 SQL Server 服务，包括



不同实例的服务，启动你应用实例的 SQL Server 服务，使其状态为“正在运行”。

实验 2：创建和管理数据库

实验名称和性质

实验名称	创建和管理数据库
实验学时	1
实验性质	<input checked="" type="checkbox"/> 验证 <input type="checkbox"/> 综合 <input type="checkbox"/> 设计
必做/选做	<input checked="" type="checkbox"/> 必做 <input type="checkbox"/> 选做

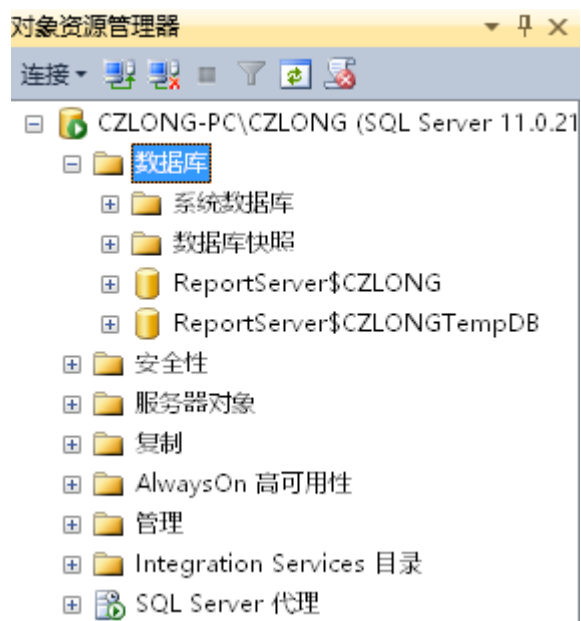
实验目的

- 1) 熟悉 SQL Server Management Studio 窗口。
- 2) 掌握创建数据库的方法。
- 3) 掌握管理数据库的方法。

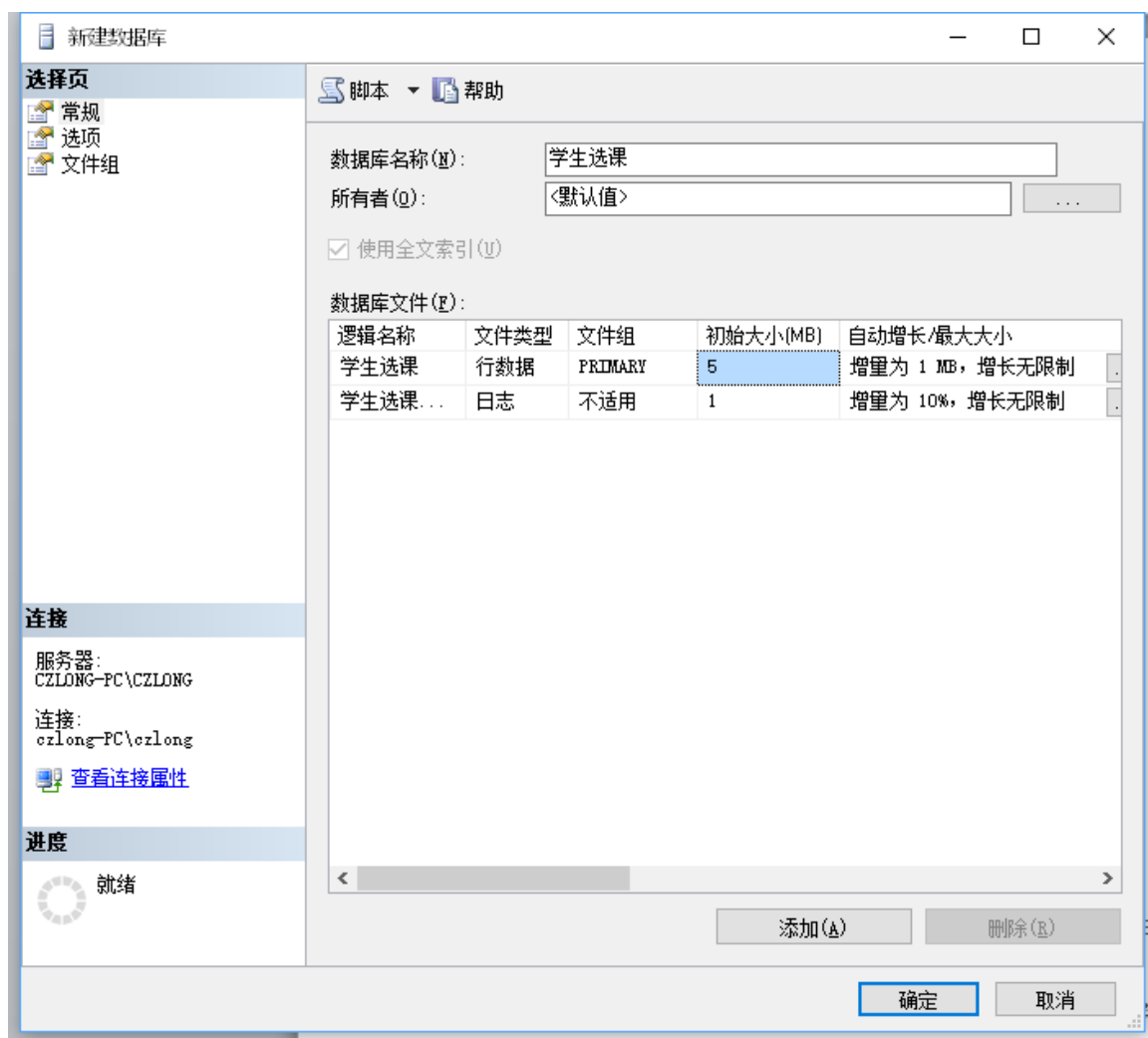
实验内容

(一) 企业管理器使用

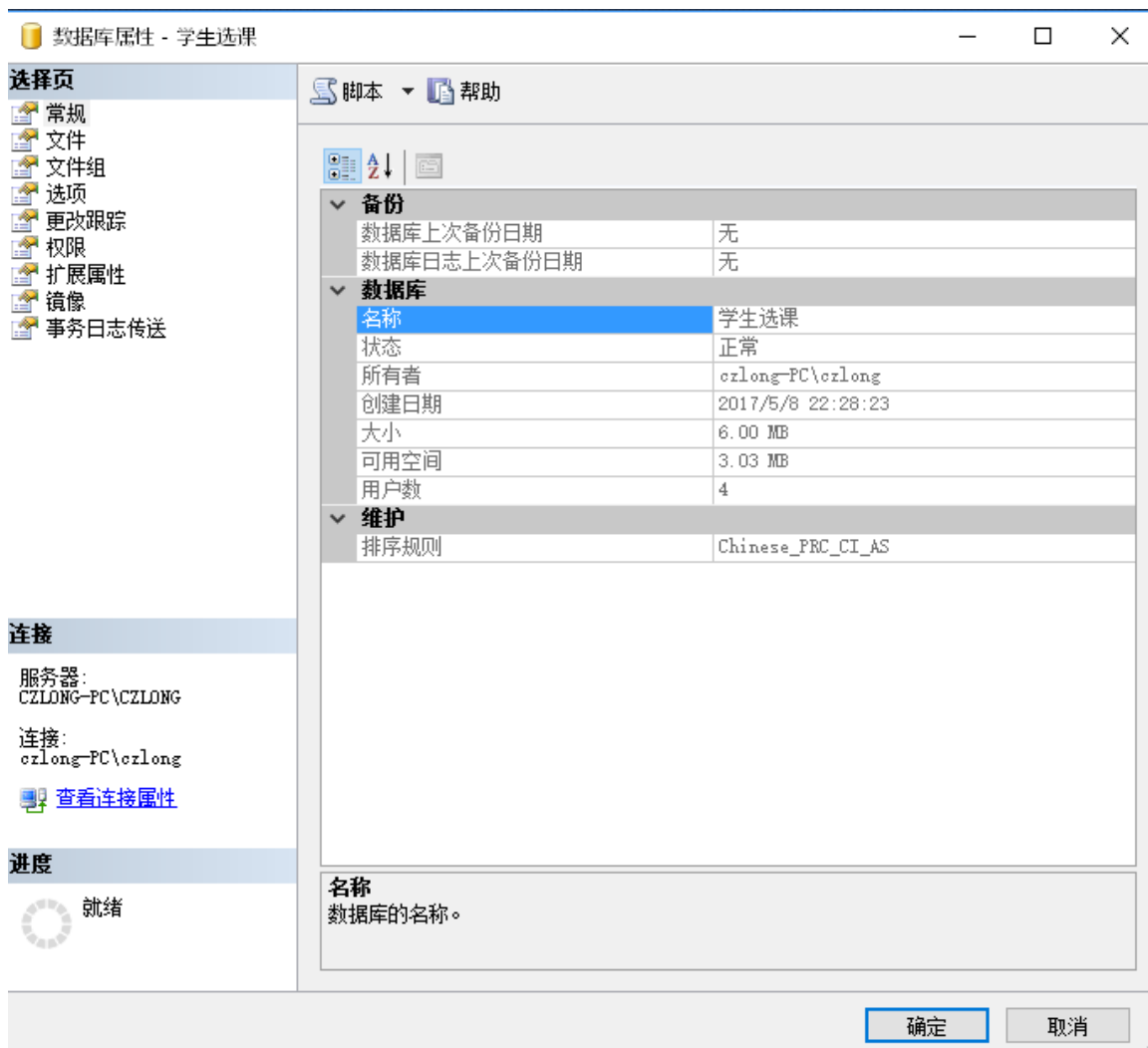
- 1) 打开“SQL Server Management Studio”窗口，在“对象资源管理器”中展开服务器，鼠标右键单击“数据库”节点，单击“新建数据库”命令，



2) 会出现“新建数据库”对话框。

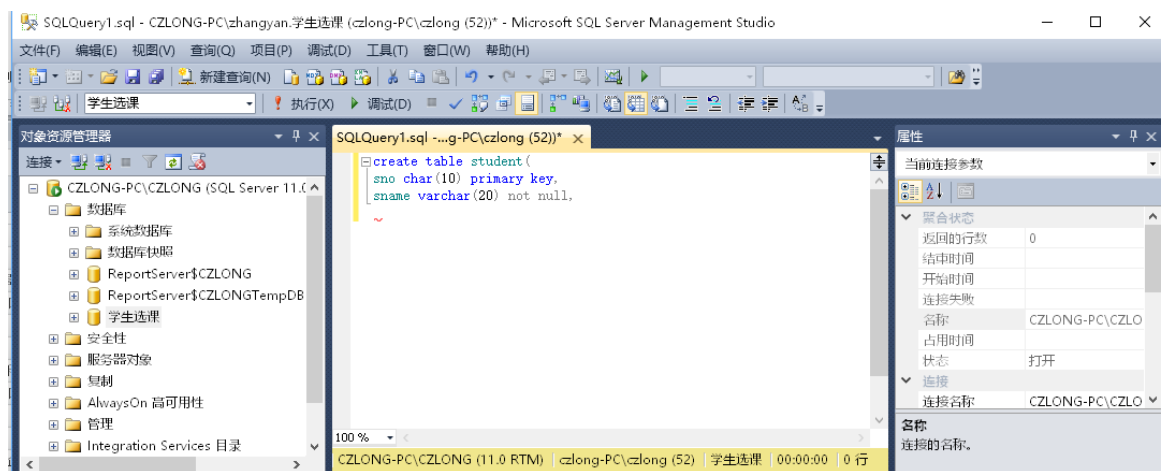



- 3) 在对话框的“数据库名称”框内输入数据库名“学生选课”后，单击“确定”按钮即可创建默认大小的数据库。
- 4) 鼠标右键单击“学生选课”数据库，在弹出的快捷菜单中选择“属性”命令，会出现“数据库属性”对话框；在对话框中单击“文件”选项卡，可以增加或删除数据库文件，单击“确定”按钮即可完成数据库的修改。

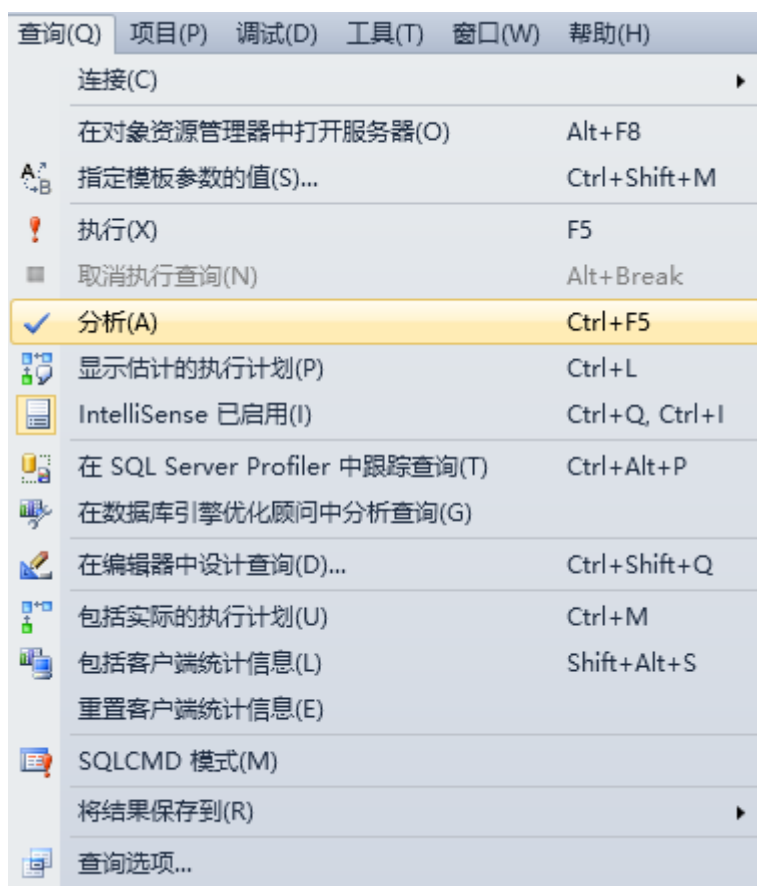


(二) 查询分析器使用

- 1) SQL Server 管理平台的对象资源管理器窗口中，选择“学生选课”数据库，单击鼠标右键，在弹出快捷菜单中选择“新建查询”命令，或者选择数据库后，单击工具栏的“新建查询”命令，打开查询分析器窗口



- 2) 在查询分析器的编辑窗口中，输入 SQL 代码。
- 3) 选择 SQL Server 管理平台的“查询”→“分析”菜单项，或者单击常用工具栏上  按钮，查询分析器将对输入的代码进行语法分析，并由消息窗给出分析结果报告。



- 4) 选择 SQL Server 管理平台的“查询”→“执行”菜单项，或者单击常用工具栏上“执行”按钮，SQL Server 将编译窗口中的代码，并在结果窗格中显示查询结果。

根据上述步骤执行如下操作：

CREATE DATABASE LMS

在建立一个应用系统的关系表之前，最好创建属于应用自己的数据库。

选中该语句并执行出现：命令已成功完成。表示数据库成功建立。可以在“对象资源管理器”中的服务器名称上右击鼠标，并单击“刷新”，在服务器中可以看到 LMS 数据库。

实验 3：表建立与数据插入

实验名称和性质

实验名称	数据表的建立与数据的插入
实验学时	2
实验性质	<input checked="" type="checkbox"/> 验证 <input type="checkbox"/> 综合 <input checked="" type="checkbox"/> 设计
必做/选做	<input checked="" type="checkbox"/> 必做 <input type="checkbox"/> 选做

实验目的

- 1) 掌握数据表的建立方法
- 2) 掌握数据表中数据的插入方法。

实验内容

- 1) 在企业管理器中和查询分析器中创建数据表；
- 2) 企业管理器中和查询分析器中数据表中数据的插入。

验证性实验

(一) 企业管理器使用

1) 打开上一章创建的“LMS”数据库，并在“对象资源管理器”窗口的右边窗口中用鼠标右键点击“表”对象，选择“新建表”命令，打开 SQL Server 的表编辑器窗口创建如下几张表：

表1 学生表

列名	数据类型	长度	是否允许为空
学号	Char	6	N
姓名	Char	8	N
性别	Bit	1	N
出生日期	smalldatetime	4	N

专业名	Char	10	N
所在系	Char	10	N
联系电话	char	11	Y

表2 课程表

列名	数据类型	长度	是否允许为空
课程号	Char	3	N
课程名	Char	20	N
学时	Tinyint	1	N
学分	Tinyint	1	N

表3 选课表

列名	数据类型	长度	是否允许为空
学号	Char	6	N
课程号	Char	3	N
成绩	Tinyint	1	Y

(二) 查询分析器使用

用 T-SQL 语句在查询分析器中创建数据表 Student:

```
USE LMS

CREATE TABLE Student (

SNO VARCHAR(10) NOT NULL,

SNAME VARCHAR(20) NOT NULL,

SEX CHAR(2),

BIRTHDAY DATETIME)
```

选中上述语句并执行，出现命令已成功完成，表示数据表已经建立好。可以在“对象浏览器”中的 LMS 数据库上右击鼠标，并单击“刷新”，然后展开“用户表”，可以看到 Student 数据表。

4) 向 Student 数据表中插入数据记录。

```
INSERT INTO Student

VALUES ('200512', 'Josephine', 'F', '1980-12-20')
```

选中上述语句并执行，出现“1 行受影响”，表示数据已经插入到数据表中。

设计性实验

- 1) 删除验证性试验中创建的 LMS 数据库,
- 2) 再创建 LMS 数据库。
- 3) 在 LMS 数据库中创建三个数据表:

```
S (SNO VARCHAR (10) NOT NULL ,SNAME VARCHAR (20) ,DEPT VARCHAR (20) ,  
AGE INT, SEX CHAR (4) )
```

```
C (CNO VARCHAR (10) NOT NULL, CNAME VARCHAR (20) ,cpno  
varchar(10),credit int)
```

```
SC (SNO VARCHAR (10) NOT NULL, CNO VARCHAR (10) NOT NULL, SCORE  
DECIMAL)
```

- 3) 在 3 个表中输入如下数据。

表 4 学生表-S

学号SNO	姓名SName	系部Dept	年龄Age	性别Sex
200512	李勇	计算机系	20	男
200518	刘晨	计算机系	19	女
200018	王敏	数学系	18	女
200511	杨扬	物理系	20	男
200510	张立	信息系	19	男
200513	张立	物理系	19	男
200514	王点点	信息系	19	女
200012	欧阳雨	数学系	20	男
200515	刘依依	数学系	23	女

表 5 课程表-C

课程号CNo	课程名CName	前修课程Cpno	学分Credit
c01	数据库原理	c03	4
c02	信息系统	C01	4
c03	数据结构	null	6
c04	DB_设计	co1	3

表 6 成绩表-SC

学号SNo	课程号CNo	成绩Score
200512	c02	70
200512	c03	null
200512	c04	null
200515	c01	80
200518	c03	95

200018	c01	80
200518	c02	45
200511	c02	78
200511	c01	45
200511	c03	89
200514	c02	78
200514	c01	45
200514	c03	89
200514	c04	78

实验 4：单表查询

实验名称和性质

实验名称	单表查询
实验学时	2
实验性质	<input checked="" type="checkbox"/> 验证 <input type="checkbox"/> 综合 <input checked="" type="checkbox"/> 设计
必做/选做	<input checked="" type="checkbox"/> 必做 <input type="checkbox"/> 选做

实验目的

- 1) 掌握数据的基本检索方法；
- 2) 掌握数据查询的 Group by 和 Order by 子句的使用；
- 3) 掌握聚集函数的使用方法。

知识准备

数据检索的语句格式：

SELECT [ALL|DISTINCT] <目标列表表达式列表>

FROM <表名或视图名列表>

[WHERE <条件表达式>]

[GROUP BY <列名 1> [HAVING <条件表达式>]]

[ORDER BY <列名 2> [ASC|DESC]]

其中，SELECT 后的目标列表表达式可以是列名、表达式或函数。

GROUP BY 子句：对查询结果按指定列的值分组，该属性列值相等的元组为一个组。通常会在每组中作用集函数。

HAVING 短语：筛选出只有满足指定条件的组

ORDER BY 子句：对查询结果表按指定列值的升序或降序排序

注意：T-SQL 语句中关键字不区分大小写

实验内容

- 1) 对数据表进行简单检索;
- 2) 对数据表进行排序检索;
- 3) 在检索中使用聚集函数;

验证性实验

先选择要操作的数据库，用 T-SQL 命令：**USE LMS**；或在工具栏的当前数据库中选择 **LMS**。

以下查询要求在逐个语句执行，执行后将执行结果记录下来：

- 1) 无条件查询全部数据

```
USE LMS
SELECT * FROM S
```

- 2) 在 SELECT 关键字后指明要检索的列名

- ① 查询S表的学生学号和姓名

```
SELECT SNO, SNAME FROM S
```

- ② 查询S表中的系部名

```
SELECT DEPT FROM S
```

- ③ 查询S表中不重复的系部名

```
SELECT DISTINCT DEPT FROM S
```

- 3) 改变列标题的检索

- ① 使用 空格 形式： 列名 新标题

```
SELECT SNO 学号 , SNAME 姓名 FROM S
```

- ② 使用 “AS” 形式，列名 AS 新标题

```
SELECT SNO AS 学号, SNAME AS 姓名 FROM S
```

- 4) 有条件选择的查询

- ① 在S表检索 “信息系” 的学生信息

```
SELECT * FROM S WHERE DEPT='信息系'
```

- ② 在S表中检索姓 “王” 的学生信息

```
SELECT * FROM S WHERE sname like '王%'
```

③ 在SC表检索'C01'选修课成绩为空的选课记录

```
SELECT SNO, CNO FROM SC WHERE cno='C01' and SCORE is null
```

④ 检索年龄为21, 18, 22的学生学号、姓名

```
SELECT SNO, SNAME FROM S WHERE age in (21,18,22)
```

Age IN {21, 18, 22} 表示某条记录的AGE字段值是否是集合{21, 18, 22}中的元素, 如是, 则选择。它等价于下面语句:

```
SELECT SNO, SNAME FROM S WHERE age=18 or age=21 or age=22
```

5) 使用聚集函数

① 查询选课表中最高分、平均分、最低分

```
SELECT MAX(SCORE), AVG(SCORE), MIN(SCORE) FROM SC
```

② 查询“C01”课程的最高分、平均分和最小成绩。

```
SELECT MAX(SCORE), AVG(SCORE), MIN(SCORE)
FROM SC
WHERE CNO='C01'
```

6) 对检索结果进行排序

```
SELECT * FROM SC
WHERE SCORE IS NULL
ORDER BY SNO, CNO DESC
```

7) 进行分组统计:

① 查询各学生的选课数

```
SELECT SNO, COUNT(*)
FROM SC
GROUP BY SNO
```

② 使用HAVING: “选课表”中查询选修了3 门以上课程的学生学号。

```
SELECT SNO, COUNT(*) FROM SC
GROUP BY SNO
HAVING COUNT(*)>=3
```

③ “选课表”中按学号分组汇总学生的平均分, 并按平均分降序排列。

```
SELECT SNO 学号, AVG(SCORE) 平均分 FROM SC
GROUP BY SNO
ORDER BY 平均分 DESC
```

④ 分析下面两个SELECT语句执行结果, 说明有什么不同?

```
SELECT SNO FROM SC
ORDER BY SNO
GO
SELECT SNO FROM SC
GROUP BY SNO
```

设计性实验

- 1) 查询计算机系学生的学号和姓名。
- 2) 查询选修了课程的学生学号。
- 3) 查询选修 C01 课程的学生学号和成绩，并要求对查询结果按成绩的降序排列，如果成绩相同则按学号的升序排列。
- 4) 查询每门课程的平均分。
- 5) 查询学校开设的课程总数。
- 6) 查询选修两门及两门以上课程的学生学号。
- 7) 查询年龄在 20-22 之间的男生的学号、姓名、系部。
- 8) 查询选修‘ C01’ 课程的学生人数。

实验 5：多表查询

实验名称和性质

实验名称	多表查询
实验学时	2
实验性质	<input checked="" type="checkbox"/> 验证 <input type="checkbox"/> 综合 <input checked="" type="checkbox"/> 设计
必做/选做	<input checked="" type="checkbox"/> 必做 <input type="checkbox"/> 选做

实验目的

- 1) 掌握多表连接查询方法；
- 2) 掌握 IN 子查询的嵌套查询；
- 3) 了解 EXISTS 嵌套查询方法；

知识准备

- 1) 多表的连接查询（相当于做笛卡儿乘积）

SELECT <目标列表表达式列表> FROM 表名, 表名

- 2) 多表的等值连接查询

SELECT [ALL|DISTINCT] <表名.列名>...

FROM 表 1, 表 2

WHERE 表 1.列名=表 2.列名

注意：多表连接时要注意当某列名在两张表中同时存在时，在列名前要加表名以示区分。

- 3) 表自身的连接

SELECT 别名.列名, ...

FROM 表 1 AS 别名 1, 表 1 AS 别名 2

WHERE 别名 1.列名=别名 2.列名

注意：自身连接时因为是对同一张表操作，为区分开来对该表操作的顺序，需要对表

取两个别名，以示区别。

4) IN 嵌套子查询

SELECT <目标列表表达式列表>

FROM 表名

WHERE 列名 IN

(SELECT 子句)

5) * EXISTS 嵌套子查询

—带有 EXISTS 谓词的子查询不返回任何数据，只产生逻辑真值 “true” 或逻辑假值 “false”。

- 若内层查询结果非空，则返回真值

- 若内层查询结果为空，则返回假值。

由 EXISTS 引出的子查询，其目标列表表达式通常都用*，因为带 EXISTS 的子查询只返回真值或假值，给出列名无实际意义。

所以，EXISTS 子查询中一般是相关自查询，即子查询脱离父查询后不能单独执行。

思考：如果 EXISTS 子查询中是不相关子查询，会有什么结果？

实验内容

1) 多表等值连接查询；

2) 外连接查询；

3) IN 子查询嵌套；

4) * EXISTS 嵌套查询。

验证性实验

1) 多表的连接查询（相当于做笛卡儿乘积）

SELECT * FROM C, SC

这里你可以往这两个表中追加大量数据，观察执行的效率

2) 表的等值连接查询

① 查询各学生的选课信息（包括学号、课程名、成绩）

因为学号和成绩在SC表中，而课程名在C表中，因此需要多表查询。

SELECT SNO, CNAME, SCORE

FROM C, SC

```
WHERE C.CNO=SC.CNO
```

- ② 查询学生的选课记录，显示学生的学号、姓名、课程号、成绩)

```
SELECT S.SNO, SNAME, CNO, SCORE
FROM S, SC
WHERE S.SNO=SC.SNO
```

- ③ 查询学生的选课记录，显示学生的学号、姓名、课程号、课程名、成绩

```
SELECT S.SNO, SNAME, SC.CNO, CNAME, SCORE
FROM S, SC, C
WHERE S.SNO=SC.SNO AND SC.CNO=C.CNO
```

- ④ 左外连接查询：当希望左表（第一张表）中所有记录全部显示出来时，需要用左外连接操作。

```
INSERT INTO S (SNO, SNAME) VALUES ('20000', 'ZXX')
```

```
SELECT S.SNO, SNAME, CNO, SCORE
FROM S
LEFT OUTER JOIN SC ON S.SNO=SC.SNO
```

观察与下面等值连接的执行结果有何不同？

```
SELECT S.SNO, SNAME, CNO, SCORE FROM S, SC WHERE S.SNO=SC.SNO
```

3) 表自身的连接

- ① 查询与‘李勇’同系的学生学号

将学生表S与S本身进行等值连接（系部相等），因为S与S做连接操作时不能区分，所以，对表取一个别名。然后将第二张表S中名字为‘李勇’的记录选择出就可。

```
SELECT S1.SNO
FROM S AS S1, S AS S2
WHERE S1.DEPT=S2.DEPT AND S2.SNAME='李勇'
```

- ② 显示每个学生的非最高分成绩（学生自己的选课成绩中，不是最高分的选课记录显示出来）

```
SELECT SNO, CNO, SCORE FROM SC AS SC1
WHERE SCORE < (SELECT MAX(SCORE) FROM SC AS SC2
WHERE SC2.SNO=SC1.SNO)
```

4) IN 嵌套子查询

- ① 不相关的IN 子查询：子查询可以单独执行，与被嵌套的查询无关。

如，查询与‘李勇’同系的学生学号

可以先查询出‘李勇’所在的系，然后再到S表中查询与上述结果相同的记录。

```

SELECT SNO
FROM S
WHERE DEPT IN
(SELECT DEPT FROM S WHERE SNAME='李勇')
查询'数据库原理'课程的选课人数
SELECT COUNT (*) FROM SC
WHERE CNO IN
( SELECT CNO FROM C WHERE CNAME='数据库原理')

```

② 相关的子查询：子查询中要用到父查询表的信息，子查询不能独立执行。

如，查询选修课程号为“C01”课程且成绩至少高于选修课程号为“C02”的同学的Cno、Sno和SCORE。

在子查询中，因为要查找该同学'C02'课程的成绩，所以，需要父查询表中该学生的学号信息。

```

SELECT CNO, SNO, SCORE
FROM SC AS SC1
WHERE CNO='C01' AND SCORE >(
SELECT SCORE FROM SC AS SC2
WHERE SC2.SNO=SC1.SNO AND SC2.CNO='C02')

```

5) * EXISTS 嵌套子查询

① 执行以下语句，观察显示的两个查询结果

```

SELECT SNO, SNAME
FROM S
WHERE EXISTS
(SELECT * FROM SC WHERE CNO='C03')
GO
SELECT SNO, SNAME FROM S

```

② 执行以下语句，观察显示的两个查询结果

```

INSERT INTO C VALUES ('C06', '数据库安全', NULL, 3)
SELECT SNO, SNAME
FROM S
WHERE EXISTS
(SELECT * FROM SC WHERE CNO='C06')

```

③ 查询'数据库原理'课程的选课人数

```

SELECT COUNT (*)
FROM SC

```



```
WHERE EXISTS  
( SELECT * FROM C  
  WHERE C.CNO=SC.CNO AND CNAME='数据库原理')
```

设计性实验

- 1) 查询“计算机系”学生所教课程的成绩表。
- 2) 查询成绩比该课程平均成绩低的同学的成绩表。
- 3) 查询选修 C01 课程的学生学号、课程名、成绩，并要求对查询结果按学号的降序排列，如果学号相同则按成绩的升序排列。
- 4) 查询选修两门及两门以上课程的学生学号及姓名。
- 5) 查询年龄在 20-22 之间的男生的选修的课程号。
- 6) 查询选修‘数据结构’课程的学生人数。
- 7) 查询每门选课成绩在 80 分以上的学生学号、姓名。

实验 6：数据操作与索引

实验名称和性质

实验名称	数据操作与索引
实验学时	2
实验性质	<input checked="" type="checkbox"/> 验证 <input type="checkbox"/> 综合 <input checked="" type="checkbox"/> 设计
必做/选做	<input checked="" type="checkbox"/> 必做 <input type="checkbox"/> 选做

实验目的

- 1) 掌握数据操作--插入、删除、修改；
- 2) 掌握数据表的索引方法；
- 3) 掌握视图的创建机制及其上的操作。

知识准备

- 1) 数据插入的两种方式
 - 单记录插入—将数值列表所表示的记录插入数据表
INSERT INTO 表名[<目标列表>] VALUES (数值列表)
 - 多记录插入—将SELECT语句的查询结果插入到数据表
INSERT INTO 表名[<目标列表>] SELECT 语句
- 2) 数据删除的语句格式：


```
DELETE FROM 表名 [WHERE 字句]
```
- 3) 数据修改格式：


```
UPDATE 表名 SET 字段名=表达式 [WHERE 字句]
```
- 4) 索引—提高数据效率的有效方法，建立索引的方法有两种：
 - 非聚族索引：根据某些字段按次序建立索引表，但是不修改原来的数据表；1张数据表上可以建立多个非聚族索引。
 - 聚族索引：根据某些字段进行排序建立索引，同时将原表根据该字段排序过。一张表上只能建立一个聚族索引。
- 5) 根据索引看数据表

```
SELECT * FROM 表名 (INDEX=该表上的索引名)
```

实验内容

- 1) 数据的插入操作;
- 2) 数据的修改操作;
- 3) 数据的查询操作;
- 4) 聚族索引、非聚族索引的建立
- 5) 视图的建立及其上的操作

验证性实验

- 1) 数据的插入

① 单记录的插入

```
USE LMS
GO
SELECT * FROM S
INSERT INTO S(SNO, SNAME) VALUES('200300', 'Paulwen')
SELECT * FROM S
```

② 多记录的插入

```
CREATE TABLE SBAK(
SNO VARCHAR(10) NOT NULL,
SNAME VARCHAR(20),
DEPT VARCHAR(20),
AGE INT,
SEX CHAR(4))
GO
SELECT * FROM SBAK
INSERT INTO SBAK (SELECT * FROM S)
SELECT * FROM SBAK
```

- 2) 数据的删除

```
CREATE TABLE CBAK (
CNO VARCHAR(10) NOT NULL,
CNAME VARCHAR(20),
CPNO VARCHAR(10),
```

```

CREDIT INT)
GO
INSERT INTO CBAK SELECT * FROM C
DELETE FROM CBAK WHERE CREDIT<4
SELECT * FROM CBAK

```

3) 数据的修改

```

SELECT * FROM S
UPDATE S SET DEPT='CS' WHERE DEPT='计算机系'
GO
SELECT * FROM S

```

4) 索引的建立

① 建立非聚族索引

```

SELECT * FROM C
GO
CREATE INDEX ICNAME ON C (CNAME)
GO
SELECT * FROM C /*看数据表 */
SELECT * FROM C (INDEX=ICNAME) /*按索引次序看数据表，不是所有数
据库都支持 */

```

② 建立聚族索引

```

SELECT * FROM C
GO
CREATE CLUSTERED INDEX CICNO ON C (CNO)
GO
SELECT * FROM C

```

③ 建立唯一索引

```

CREATE UNIQUE INDEX UISNO ON SC (SNO)
执行后有什么结果？为什么会出现这个结果？
CREATE UNIQUE INDEX UISNO ON SC (SNO, CNO DESC)
执行后有什么结果？

```

④ 建立复合索引

```

SELECT * FROM S
GO
CREATE INDEX IAGESNO ON S (AGE DESC, SNO)
GO

```

```
SELECT * FROM S
SELECT * FROM S (INDEX =IAGESNO)
```

⑤ 查看表中的索引

```
SP_HELPINDEX 'S'
```

⑥ 删除表中的索引

```
DROP INDEX S.IAGESNO
GO
SP_HELPINDEX 'S'
```

5) 视图及其操作

① 视图的建立

```
CREATE VIEW SHOWSNO
AS
SELECT S.SNO, SNAME, AVG(SCORE) AS 平均成绩
FROM S, SC
WHERE S.SNO=SC.SNO
GROUP BY S.SNO, SNAME
GO
SELECT * FROM SHOWSNO
```

创建信息系学生的视图

```
CREATE VIEW VISDEPT
AS
SELECT * FROM S WHERE DEPT='信息系'
```

② 在视图上修改数据

```
UPDATE VISDEPT
SET DEPT='IS'
GO
SELECT * FROM VISDEPT
视图中还有数据吗？为什么会这样？
```

```
CREATE VIEW S_SUM
AS
SELECT SNO, SUM(SCORE) AS TOTALSCORE FROM SC
GROUP BY SNO
GO
```

```
UPDATE S_SUM SET TOTALSCORE=60
```

执行语句，出现什么结果？分析为什么？

③ 在视图上删除数据

```
SELECT * FROM VISDEPT
```

```
DELETE FROM VISDEPT WHERE SEX='女'
```

```
SELECT * FROM VISDEPT
```

```
SELECT * FROM S
```

设计性实验

- 1) 将 S 表系部为“计算机系”学生对应的系部改为‘CS’。
- 2) 创建一个与 SC 表相同新数据表 SCNEW，查询 SC 表中成绩为空的记录并将其复制到 SCNEW 表中。
- 3) 删除 SC 表中成绩为空的选课记录。
- 4) 根据 S 表中姓名字段建立唯一性索引。
- 5) 建立 S 表上 SNO 的聚族索引。
- 6) 建立 SC 表上根据 SNO 升序，CNO 降序的复合索引。
- 7) 建立每门课程的课程号、课程名、选课人数、平均分的视图。

实验 7：数据完整性约束

实验名称和性质

实验名称	数据完整性约束
实验学时	2
实验性质	<input checked="" type="checkbox"/> 验证 <input type="checkbox"/> 综合 <input checked="" type="checkbox"/> 设计
必做/选做	<input checked="" type="checkbox"/> 必做 <input type="checkbox"/> 选做

实验目的

- 1) 掌握数据完整性约束的类型;
- 2) 掌握 SQL SERVER 中的相关完整性约束;

知识准备

- 1) SQL SERVER 中的完整性约束
 - ① Primary key约束:利用表中的一列或多列来唯一标识一行数据. 能确保primary key 对应的数据列不为空, 且数据不重复.
 - ② default约束:处理用户不包含全部数据列的数据插入.
 - ③ check约束通过检查输入数据的值来维护数据的完整性.
 - ④ unique约束确保主键外的列数据的唯一性
 - ⑤ Foreign key主要用来维护两个表之间的数据一致性.
- 2) 创建数据表时指明完整性约束

CREATE TABLE <表名>

(<列名> <数据类型>[<列级完整性约束条件>]

[, <列名> <数据类型>[<列级完整性约束条件>]] ...

[, <表级完整性约束条件>])

<列级完整性约束条件>：涉及相应属性列的完整性约束条件

<表级完整性约束条件>：涉及一个或多个属性列的完整性约束条件

实验内容

- 1) 建立新表时增加完整性约束。
- 2) 为已有表添加完整性约束。
- 3) 为两表建立关联，实现参照完整性。

验证性实验

每小题目语句输入好后执行，观察执行后有什么结果？想想为什么？

(1) PRIMARY KEY 主键约束的建立

- 1) 建立表时加主键约束

```
CREATE TABLE ST(SNO VARCHAR(10) PRIMARY KEY ,SNAME VARCHAR(20) NOT NULL,  
DEPT VARCHAR(20), AGE INT, SEX CHAR(4))
```

```
GO
```

```
INSERT INTO ST (SNO, SNAME, DEPT) VALUES ('1001', 'ZXX', NULL)
```

```
GO
```

```
INSERT INTO ST (SNO, SNAME) VALUES ('1001', 'MID')
```

```
GO
```

```
INSERT INTO ST (SNO, SNAME) VALUES (NULL, 'ZXX')
```

- 2) 在已有的表上添加约束

```
CREATE TABLE ST2 (SNO VARCHAR(10) NOT NULL ,SNAME VARCHAR(20) NOT  
NULL, DEPT VARCHAR(20), AGE INT, SEX CHAR(4))
```

```
GO
```

```
INSERT INTO ST2 SELECT * FROM S
```

```
GO
```

```
ALTER TABLE ST2 ADD CONSTRAINT pk_Sno PRIMARY KEY (SNO)
```

```
INSERT INTO ST2 (SNO, SNAME, DEPT) VALUES ('200512', 'Beibei', NULL)
```

运行后出现什么结果？分析原因。

(2) DEFAULT 约束

```
CREATE TABLE CUST(  
    NO VARCHAR(5) PRIMARY KEY,  
    WEIGHT INT DEFAULT(10))  
GO  
INSERT INTO CUST(NO) VALUES('ZY01')  
INSERT INTO CUST(NO) VALUES('ZY03')  
INSERT INTO CUST VALUES('ZY02', 20)  
GO  
SELECT * FROM CUST  
运行后出现什么结果？分析原因。
```

(3) CHECK 约束

```
CREATE TABLE CUSTOMER(CUSTNO CHAR(4) NOT NULL CHECK (CUSTNO LIKE  
'[A-Z0-9][A-Z0-9][A-Z0-9][A-Z0-9]'), CUSTNAME VARCHAR(20))  
GO  
INSERT INTO CUSTOMER VALUES('BJ01','Beijing Grid Corp.')GO  
INSERT INTO CUSTOMER VALUES('BJ1','Beijing Grid Corp.')GO  
INSERT INTO CUSTOMER VALUES('BJ*1','Beijing Grid Corp.')运行后出现什么结果？分析原因。这里的[A-Z0-9]是代表啥意义？
```

(4) UNIQUE 唯一性约束的建立

```
CREATE TABLE CUST1(  
    CUSTNO CHAR(4) PRIMARY KEY,  
    CUSTNAME VARCHAR(20) UNIQUE,  
    COUNTRY VARCHAR(10))  
GO  
INSERT INTO CUST1(CUSTNO) VALUES('BJ01')  
GO  
INSERT INTO CUST1(CUSTNO) VALUES('BJ02')  
GO  
INSERT INTO CUST1 VALUES('BJ03','Beijing Grid Corp.')GO
```

```
INSERT INTO CUST1 VALUES ('BJ04','Beijing Grid Corp.')
```

运行后出现什么结果？分析原因。

(5) FOREIGN KEY 外键约束的建立

先创建如下数据表：

```
Create table S (SNO VARCHAR (10) NOT NULL ,SNAME VARCHAR(20),DEPT  
VARCHAR(20), AGE INT, SEX CHAR(4) )
```

```
Create table C (CNO VARCHAR(10) NOT NULL, CNAME VARCHAR(20),cpno  
varchar(10),credit int)
```

```
Create table SC(SNO VARCHAR(10) NOT NULL, CNO VARCHAR(10) NOT NULL, SCORE  
DECIMAL)
```

然后执行如下语句：

```
ALTER TABLE C ADD CONSTRAINT CpriKEY PRIMARY KEY(CNO)  
GO
```

```
ALTER TABLE SC ADD CONSTRAINT scpriKEY PRIMARY KEY(SNO,CNO)
```

```
ALTER TABLE SC ADD CONSTRAINT scforKEY1 FOREIGN KEY (SNO)  
REFERENCES ST2(SNO)
```

```
ALTER TABLE SC ADD CONSTRAINT scforKEY2 FOREIGN KEY (CNO)  
REFERENCES C(CNO)
```

GO

```
INSERT INTO SC (SNO,CNO) VALUES ('890','C10')
```

```
INSERT INTO SC (SNO,CNO) VALUES ('200518','C01')
```

```
INSERT INTO SC (SNO,CNO) VALUES ('890',NULL)
```

```
INSERT INTO SC (SNO,CNO) VALUES (NULL,'C10')
```

运行后出现什么结果？分析原因。

设计性实验

- 1) 在 S 表中添加完整性约束：SNO 设置为主键，SEX 的却省值为‘女’，AGE 的有效值为 16-25。
- 2) 创建 1 张与 S 相同的表 S1，在创建的同时将 SNO 设置为主键，SEX 的却省值为‘女’，AGE 的有效值为 16-25，并将 S 表中的数据插入到 S1 中，插入不同的记录来严整设置的完整性。
- 3) 创建 1 张与 SC 相同的表 SC1，将 (SNO, CNO) 设置为主键，SNO 和 CNO 设置为外键，并将 SC 表中的数据复制到 SC1 中，插入不同的记录来严整设置的完整性。

- 4) 创建 1 张与 C 表相同的表 C1, 将 C 中数据插入到 C1 后, 在 C1 上添加完整性约束: 将 (CNO) 设置为主键, CPNO 引用 CNO, CREDIT 值为 1-6。

实验 8: SQL 编程及存储过程

实验名称和性质

实验名称	SQL 编程及存储过程
实验学时	2
实验性质	<input checked="" type="checkbox"/> 验证 <input type="checkbox"/> 综合 <input checked="" type="checkbox"/> 设计
必做/选做	<input checked="" type="checkbox"/> 必做 <input type="checkbox"/> 选做

实验目的

- 1) 了解 T-SQL 的基本数据类型、函数;
- 2) 掌握局部变量的定义和赋值;
- 3) 掌握存储过程的定义及调用

知识准备

(1) SQL 中的基本数据类型

varchar: 字符数据。

datetime: (常量使用特定格式的字符日期值来表示, 并使用单引号括起来。例如: '1976-05-28', 'May 28, 1976', '28 May, 1976', '760528', '05/28/76'。

Integer: 用一串数字来表示, 不含小数点, 不使用引号。例如, 123, 1896。

Decimal: 用一串数字来表示, 可以包含小数点, 不使用引号。例如, 1893.1209, 2.0。

float和real: 使用科学记数法表示。例如, 101.5E5, 0.5E-2。

money: 用一串数字, 可以包含或不包含小数点, 以一个货币符号 (\$) 作为前缀, 不使用等等。

(2) SQL 中的变量

SQL中变量用来存放临时数据, 变量使用前一定要先进行定义, 变量名必须以@开

头。另外，SQL Server中还提供一些系统变量，系统变量是以@@开头的。

变量的定义： declare 变量名 数据类型

如

go

Declare @name varchar(20)

set @name='王雨'

Print @name

Go

(3) SQL 中的基本函数

① 字符串函数:LEN()、UPPER()、LOWER()、RIGHT()、LEFT()、SUBSTRING()、CHARINDEX()、STR()、REPLACE()等

② 数学函数：常用的数学函数通常对作为参数提供的输入值执行计算，并返回一个数字值。如ABS()、POWER()、SQUARE()、SQRT()、ROUND()、RAND()等。

③ 日期和时间函数：日期和时间函数对日期和时间输入值执行操作，并返回一个字符串、数字值或日期和时间值。GETDATE()、DATEADD()、DATEIFF()、DATEPART()、ISDATE()、DAY()、MONTH()、YEAR()等

④ 数据类型转换函数

CAST函数

CAST(expression AS data_type)

CONVERT函数

CONVERT(data_type[(length)], expression [, style])

(4) 存储过程的创建及调用

CREATE proc 过程名

[{@变量名 数据类型}][, ...n]

AS

Sql语句

调用：EXEC 过程名 参数列表

(5) 局部变量的定义

DECLARE @变量名 数据类型

注意：SAL中的局部变量名必须以@开头。

实验内容

- 1) 基本结构编程;
- 2) 存储过程的建立;
- 3) 存储过程的调用

验证性实验

- 1) 编程实现查询与‘李勇’同系的学生学号

```
USE LMS
GO
DECLARE @DEPT VARCHAR (20)
SELECT @DEPT=DEPT FROM S WHERE SNAME='李勇'
SELECT SNO FROM S WHERE DEPT=@DEPT
```

- 2) 运行下面的程序，观察运行结果

```
DECLARE @X INT, @Y INT
SET @X=0
SELECT @Y=1
WHILE @Y<20
BEGIN
    SET @X=@X+@Y
    SELECT @Y=@Y+2
    PRINT STR(@Y)+' IN THE LOOP'
    IF @Y>14
        BREAK
END
PRINT 'OUT OF THE LOOP'
```

- 3) CASE 函数的使用

① CASE后带表达式

```
Select 性别=case sex
        when '男' then 'M'
```

```

        when '女' then 'F'
        else '输入出错'
        end
    From s

```

② CASE后不带参数

```

SELECT 性别=CASE
WHEN SEX= '男' THEN 'M'
WHEN SEX= '女' THEN 'F'
ELSE 'ERROR'
END
FROM S

```

③ 用CASE 语句进行多条件修改

```

update s set DEPT=
case DEPT
when '计算机系' then 'CS'
when '信息系' then 'IS'
when '数学系' then 'MA'
when '物理系' then 'PH'
end

```

4) 存储过程

① 不带参数的存储过程

```

CREATE PROC SCLIST
AS
SELECT SNO,CNO, SCORE FROM SC WHERE SNO IN
(SELECT SNO FROM S WHERE DEPT='计算机系')
执行存储过程:
EXEC SCLIST
观察结果

```

② 带参数的存储过程

如：传入一个学生的学号与姓名, 显示出这个学生选修的课程号, 成绩

```

CREATE PROC SC_PROC @xh varchar(10)
AS
SELECT cno , SCORE
From sc
Where sc.sno=@xh

```

执行存储过程：

```
EXEC SC_PROC '200512'
```

5) 存储过程的相关操作

① 查看存储过程：

```
Sp_help SCLIST
```

② 重命名存储过程

```
Sp_rename 'SCLIST', 'STUDENTSCORE'
```

设计性实验

- 1) 调用帮助系统来查找系统存储过程或函数来显示 SQL SERVER 的版本号及当前系统时间。

提示：到帮助中根据关键字等查询到对应的函数或存储过程来完成任务

- 2) 编写存储过程：完成 $1! + 2! + \dots + n!$ 的计算。

提示：传入参数 n，根据 N 用循环控制来求对应的值

- 3) 将 S 表中的 DEPT 字段内容改为中文：IS-信息系，CS-计算机系，PH-物理系，MA-数学系。

提示：用 UPDATE 语句来修改 DEPT 字段，DEPT 的值根据不同英文缩写要修改为不同的中文系名，用 CASE 函数来完成

- 4) 将 C 表上增加一个字段 seleNUM，并编写存储过程：传入学号和课程号后，完成在 SC 表中插入相应的选课记录，并在 C 表中对应课程的 seleNUM 加 1。

实验 9：事务处理和触发器

实验名称和性质

实验名称	事务处理和触发器
实验学时	2
实验性质	✓ <input type="checkbox"/> 验证 <input type="checkbox"/> 综合 ✓ <input type="checkbox"/> 设计
必做/选做	✓ <input type="checkbox"/> 必做 <input type="checkbox"/> 选做

实验目的

- 1) 了解事务的概念和基本特征;
- 2) 掌握事务的提交与回滚操作
- 3) 掌握触发器的基本原理;
- 4) 掌握触发器的建立语句

知识准备

(1) 事务是一段与数据库打交道的程序，它能保持企业状态和数据库状态的一致性。当某一事件影响企业状态时，事务会更新数据库状态以体现这个事件的发生。如银行里的存款事务，事件是客户向出纳员提供现金和存款单，事务则是更新数据库中客户的帐号信息以体现这次存款事件。

(2) 事务处理具有原子性(atomicity)：系统必须保证这段程序要么执行到结束，要么就一点效果也没有。如果一个事务成功执行，我们称其已经提交(commit)；一个事务没有正常完成，称其已经中止(abort)，这时候，事务监视器有责任保证事物对数据库造成的部分改变要修改回来，这就叫事务回滚(roll back)

(3) 触发器是一种特殊类型的存储过程，它不同于前面介绍过的一般的存储过程。q在使用触发器过程中，SQL Server使用到了两张特殊的临时表，分别是inserted和deleted表。

- 在deleted表中存放Update和delete影响的旧数据行。在执行Update和delete时先将数据从基本表中删除，然后被转移到deleted表中。
- 在inserted表中存放Update和insert影响的数据。当用户执行Update和insert时，将数据添加到基本表中，同时，将数据行的备份复制到inserted临时表中。

(4) 触发器的创建格式

```
CREATE TRIGGER 触发器名 ON 表名
AFTER INSERT/UPDATE/DELETE
AS
T-SQL语句组
```

实验内容

- 1) 事务的提交与回滚操作;
- 2) 触发器的建立;
- 3) 触发器的修改、删除操作;
- 4) 触发器的删除

验证性实验

(1) 用 ROLLBACK TRAN 实现事务的回滚操作

```
USE LMS
GO
BEGIN TRAN
SELECT * FROM C
INSERT INTO C VALUES ('C08', '决策支持系统', NULL, 3)
SELECT * FROM C
ROLLBACK TRAN
SELECT * FROM C
```

观察每次显示C表中数据记录条数变化，为什么有这种变化？

(2) 事务回滚点的保存

```
USE LMS
GO
BEGIN TRAN Mytran
SELECT * FROM C
INSERT INTO C VALUES ('C09', '信息分析与预测', NULL, 4)
SAVE TRAN POINT1
INSERT INTO C VALUES ('C10', '数据挖掘', NULL, 3)
SELECT * FROM C
```

```
ROLLBACK TRAN POINT1
```

```
SELECT * FROM C
```

观察最后的C表中的数据结果与前面C表显示结果的不同。

(3) 触发器的建立

- ① 在S表建立建立如下触发器

```
CREATE TRIGGER TRIGONS
```

```
ON S FOR UPDATE
```

```
AS
```

```
SELECT * FROM DELETED
```

```
SELECT * FROM INSERTED
```

选中上述触发器创建语句并执行，然后输入下面语句：

```
UPDATE S
```

```
SET DEPT = '计算机系' WHERE DEPT = 'CP'
```

- ② 在S表上建立下面触发器：

```
CREATE TRIGGER TEMPTABLE
```

```
ON S FOR INSERT
```

```
AS
```

```
SELECT * FROM INSERTED
```

```
ROLLBACK TRAN
```

执行上述触发器后在S表中插入数据，

```
INSERT INTO S(SNO, SNAME) VALUES ('10012', 'ZXX')
```

执行插入语句后打开S观察结果。

- ③ 在C表中加入一列selectNum表示选课数量，在SC表上插入一条选课记录后，将C表中对应的课程的selectNum值加1。

```
CREATE TRIGGER SELECTCOURSE
```

```
ON SC FOR INSERT
```

```
AS
```

```
IF (SELECT COUNT (*) FROM INSERTED) >1
```

```
BEGIN
```

```
PRINT 'YOU CAN INSERT ONE RECORD ONCE. SO THE RECORDS ARE NOT INSERTED  
INTO THE TABLE'
```

```
ROLLBACK TRAN
```

```

END
ELSE
BEGIN
    DECLARE @CNO VARCHAR(10)
    SELECT @CNO=CNO FROM INSERTED
    UPDATE C
    SET selectNum=selectNum+1 WHERE CNO=@CNO
    PRINT 'THE NUMBER OF STUDENTS WHO SELECTED'+@CNO +' HAS BEEN ADDED'
END

```

(4) 触发器的修改、删除

① 修改触发器

```

ALTER TRIGGER 触发器名 ON 表名
AFTER INSERT/UPDATE/DELETE
AS
T-SQL语句组

```

在调试触发器时可以用修改触发器的语句来进行。如：

```

CREATE TRIGGER DELS
ON S FOR DELETE
AS
DECLARE @DEPT VARCHAR(20)
SELECT @DEPT=DEPT FROM INSERTED
IF @DEPT='CS'
BEGIN
    PRINT 'YOU CAN NOT DELETE THE STUDENT FROM CS'
    ROLLBACK
END

```

执行上面的触发器，发现错误，则修改触发器。

```

ALTER TRIGGER DELS
ON S FOR DELETE
AS
DECLARE @DEPT VARCHAR(20)
SELECT @DEPT=DEPT FROM DELETED
IF @DEPT='CS'
BEGIN
    PRINT 'YOU CAN NOT DELETE THE STUDENT FROM CS'

```

```
ROLLBACK  
END
```

② 删除触发器

```
DROP TRIGGER 表名. 触发器名  
DROP TRIGGER S. DELS
```

设计性实验

- 1) 将 S 表的 PRIMARY KEY 完整性约束删除, 编写一个 S 表上的触发器来完成在 S 表上插入记录时实现如下目标: SNO 不能为空, 且 SNO 不能重复。
- 2) 在 C 表修改记录时, 若修改的是 CREDIT 字段, 则修改值超过 6 时提示出错信息并不将结果写出数据表。

实验 10：数据库安全与数据库恢复

实验名称和性质

实验名称	数据库安全与数据库恢复
实验学时	2
实验性质	<input checked="" type="checkbox"/> 验证 <input type="checkbox"/> 综合 <input type="checkbox"/> 设计
必做/选做	<input type="checkbox"/> 必做 <input checked="" type="checkbox"/> 选做

实验目的

- 1) 了解数据库的安全性控制方法；
- 2) 了解数据库恢复的基本原理；
- 3) 掌握数据库备份和恢复机制

实验内容

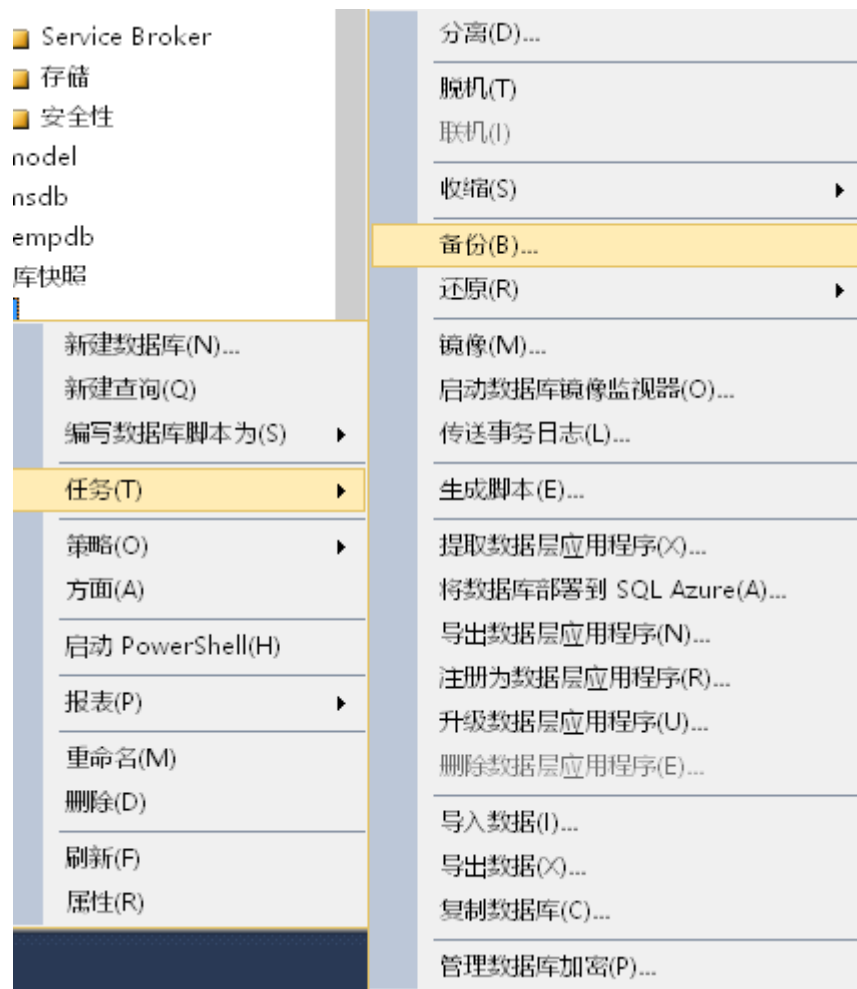
- 1) 数据库备份。
- 2) 数据库恢复。

验证性实验

(1) 数据库备份

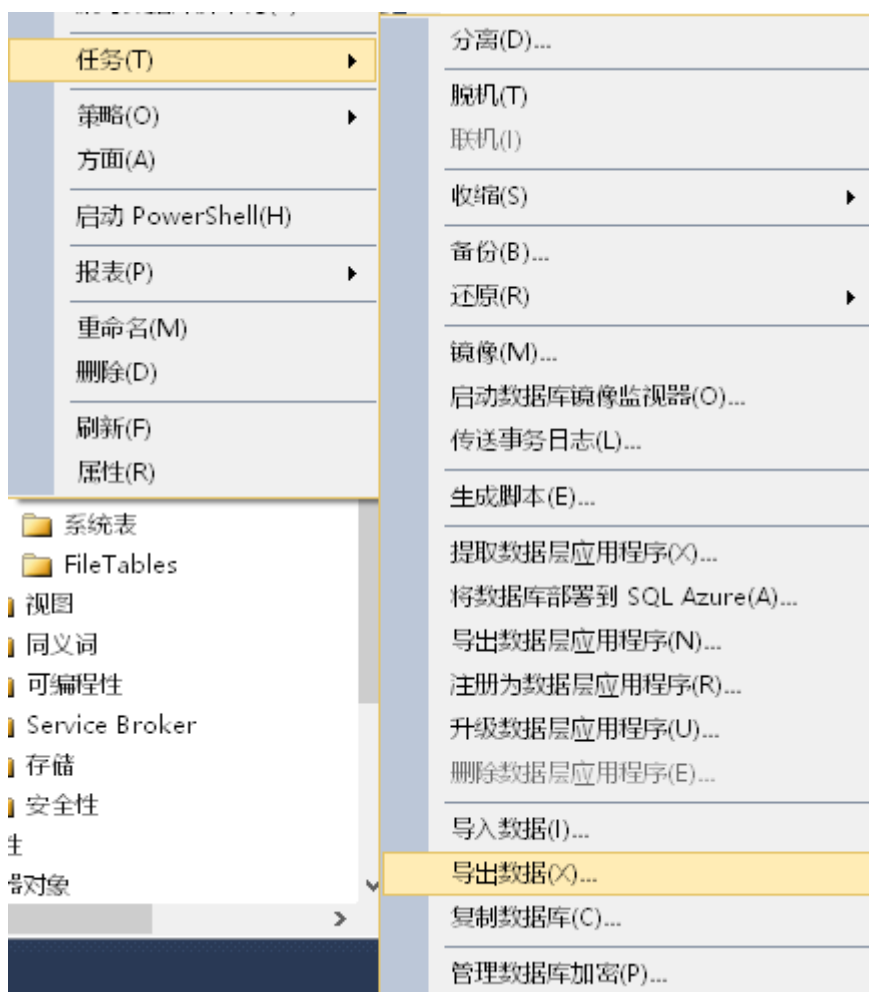
① 数据库备份

在SQL管理器中，选中对应的数据库，单击快捷菜单‘所有任务’——>‘备份数据’，在窗口中选择备份类型（完全备份、差异备份），添加备份设备（操作系统通常对数据的输入/输出操作是将数据从输入设备得来或输出到输出设备。因此，对存放数据的文件也称为文件设备。因此，添加设备实际上就是对备份文件的存储路径和文件名用一个设备名来表示），然后单击“确定”按钮，SQL SERVER将数据库备份到你设置的设备文件中。



② 数据表的导出（将数据转换成其他格式文件如 DBF，TXT、XLS 等）

在SQL管理器中，在数据库快捷菜单‘任务’——>‘导出数据’可以导出你需要的格式文件或者向其他数据库导出数据。



(2) 数据库恢复

① 数据恢复

在SQL管理器中，选中对应的数据库，单击快捷菜单‘任务’——>‘还原’，根据提示进行操作。

② 数据导入

在SQL管理器中，选中对应的数据库，单击快捷菜单‘任务’——>‘导入数据’，设置数据源（数据源提供驱动器和数据文件位置及文件名），单击“下一步”进入到“目标数据”窗口，选择目标数据驱动程序（选择SQL SERVER）和数据库名，点击“下一步”，然后根据提示进行操作即可。

实验 11：数据库设计

实验名称和性质

实验名称	数据库设计
实验学时	2
实验性质	<input type="checkbox"/> 验证 <input checked="" type="checkbox"/> 综合 <input type="checkbox"/> 设计
必做/选做	<input type="checkbox"/> 必做 <input checked="" type="checkbox"/> 选做

实验目的

- 1) 了解数据字典的描述方法；
- 2) 掌握由数据字典进行概念设计的方法；
- 3) 掌握数据库中数据表的规范化程度的判断及分解方法；

知识准备

- 1) 数据字典中包含：数据项、数据结构、数据流、数据存储、数据操作等。在实际运用中数据字典可能用数据结构、数据流图结合数据项说明等来描述。
- 2) E-R 图的组成：实体、属性、实体间联系。
- 3) 根据 E-R 图进行数据库逻辑结构设计，将实体、实体的属性和实体之间的联系转化为关系模式：
 - ① 一个实体型转换为一个关系模式。
 - ② 一个m:n联系转换为一个关系模式，该关系模式中。
 - -关系的属性：与该联系相连的各实体的码以及联系本身的属性
 - -关系的码：各实体码的组合
 - ③ 一个1:n联系
 - 若联系上有属性，将该联系转换为一个独立的关系模式，方法与 2 相同
 - 若联系上无属性，与 n 端对应的关系模式合并（在 n 端关系中加入 1 端关系的码和联系本身的属性）

- ④ 一个1:1联系可转换为一个独立的关系模式，也可以与任意一端对应的关系模式合并。
- ⑤ 三个或三个以上实体间的一个多元联系转换为一个关系模式。
 - 关系的属性：与该多元联系相连的各实体的码以及联系本身的属性
 - 关系的码：各实体码的组合
- 4) 函数依赖的概念，根据数据表或数据字典写对应关系模式中的函数依赖；
- 5) 1NF、2NF、3NF 等概念；
- 6) 模式分解方法，就是将一个大的关系模式分解为小的关系模式（即 1 张大的数据表分解为几张小的数据表，使数据表对应的关系模式具有更高的规范化程度）
 - 将关系模式分解为 3NF 且保持函数依赖的分解方法；
 - 将关系模式分解为 3NF 且保持函数依赖和无损连接的分解方法

实验内容

- 1) E-R 图的绘制；
- 2) 由 E-R 图进行数据库逻辑结构设计；
- 3) 根据数据字典、数据表写关系模式中的函数依赖集；
- 4) 关系模式规范化程度的判断及模式分解

综合性实验

在SQL 2000中有Northwind例子数据库，其中存储的是一个虚拟的从世界各地进出口食品公司的有关销售数据。

此数据库的数据字典定义如下：

- 1) Territories (城市): TerritoryID (城市编号)、TerritoryDescription (城市名)、RegionID (地区编号)；
- 2) Suppliers (供货厂商): SupplierID (供货厂商编号)、CompanyName (厂名)、ContactName (联系人名)、ContactTitle(联系人职位)、Address (地址)、City (城市名)、Region (地区)、PostalCode (邮政编码)、Country (国家)、Phone (电话)、Fax (传真)、HomePage (主页)；
- 3) Shippers (托运): ShipperID (托运人编号) CompanyName (厂名)、Phone (电话)；
- 4) Region (地区): RegionID (地区编号) RegionDescription (地区描述)；
- 5) Products (产品): ProductID (产品编号)、ProductName (品名)、SupplierID

(供货厂商编号)、CategoryID (所属种类号)、QuantityPerUnit (单位数量)、UnitPrice (单价)、UnitsInStock (库存)、UnitsOnOrder (定货数)、ReorderLevel (修订量)、Discontinued(是否进行) ;

- 6) Orders (定单): OrderID (定单编号)、CustomerID (顾客编号)、EmployeeID (职员编号)、OrderDate (定货日期)、RequiredDate (交货日期)、ShippedDate (载运日期)、ShipVia (经由数)、Freight (运费)、ShipName (船名)、ShipAddress(地址)、ShipCity (城市)、ShipRegion(地区)、PostalCode (邮政编码)、ShipCountry(国籍);
- 7) Order Details (定单详细信息): OrderID (定单编号)、ProductID (产品编号)、UnitPrice (单价)、Quantity (数量)、Discount (折扣);
- 8) EmployeeTerritories(职工所在城市): EmployeeID(职工编号)、TerritoryID(城市编号);
- 9) Employees (职工): EmployeeID (职工编号)、LastName (姓)、FirstName (名)、Title (头衔)、TitleOfCourtesy (性别) BirthDate (生日)、HireDate (受聘日期)、Address (地址)、City (城市)、Region (地区)、PostalCode (邮政编码)、Country (国籍)、HomePhone (住宅电话)、Extension (分机号)、Photo (照片)、Notes (备注)、ReportsTo (直接上级号)、Photopath (职工照片路径);
- 10) Customers (客户) :CustomerID (客户编号)、CompanyName (公司名)、ContactName (联系人名)、ContactTitle (联系人头衔)、Address (地址)、City (城市)、Region (地区)、PostalCode (邮政编码)、Country (国籍)、Phone (电话)、Fax (传真);
- 11) CustomerDemographics (客户类别说明): CustomerTypeID (客户类编号)、CustomerDesc (描绘);
- 12) CustomerCustomerDemo (客户类别): CustomerID (客户编号)、CustomerTypeID (客户类别编号);
- 13) Categories (食品类别): CategoryID (类编号)、CategoryName (类名)、Description (详细信息)、Picture (图片);

根据数据字典, 完成:

- ① 画出E-R图, 进行概念数据库设计;
- ② 查看数据表, 写出各逻辑数据库模式中的函数依赖集, 判断逻辑数据库模式中的各个关系(表)是第几范式, 如果没有达到第三范式或BC范式, 请进行规范化设计。