选择题

1.SQL Server数据文件的存储结构

从SQL Server 2016数据库的物理架构上来看，SQL Server用于存储数据的基本单位是`页`，每页容量为`8KB`。SQL Server 2016在执行底层的磁盘I/O时也是以页级为单位的。SQL Server将`8`个物理上连续的页组成一个`区`，以此可以更加有效地管理数据页。

> - 数据页 SQL Server将8KB的数据划分为一页。即在SQL Server 数据库中的`1MB数据中包含128页`。包括数据页、索引页、文本/图像页等8种。每个页的开头为`96`字节的系统信息。数据区占有`8060`个字节，页尾的行偏移数组占有36个字节。
> - 扩展盘区是SQL Server数据库读写数据的基本单位，扩展盘区就是管理存储空间的基本单位。一个扩展盘区由8个物理上连续的页（`64 KB`）组成。即SQL Server数据库中`每1MB包含16个区`。
> - 为了`提高空间利用率`，SQL Server2016在为数据库中的某个数据表分配存储区时采取两种不同的策略。
>
> ①将扩展盘区中所有`8`个存储页全部分配给一个数据库对象（例如数据表），采用这种方法分配的区也被称为“`统一区`”。统一区中的所有8个存储页`只能供所属对象使用`。
>
> ②允许扩展盘区中的存储页由`1~8`个数据对象共同使用。这种分区方式也被称为“混合区”。采用这种方式的分区，区中的`每1页（共8页）都可由不同的对象拥有`。

2.数据库文件的后缀

- 主要数据文件：包含数据库的启动信息，并指向数据库中的其他文件。文件扩展名是`.mdf`
- 次要文件：将数据分散到多个磁盘上，文件扩展名是`.ndf`
- 事务日志文件：保存用于恢复数据库的日志信息，文件扩展名是`.ldf`

3.管理数据库相关命令

3.1创建数据库db1

```
create database db1;
```

3.2删除数据库db1

```
drop database db1
```

3.3修改数据库db1的名字为db2

```
alter database db1 modify name=db2
```

4.能够在服务器之间迁移数据库的操作是什么

`分离`和`附加`，主要是分离，比如说你在这边服务器下把数据库整个的搬到另外一个服务器，那先可以在这边做分离工作，然后到另外一边做附加，这样把整个数据库就迁过去了

> 在SQL Server 2016中，除了系统数据库外，其他数据库都可以从服务器的管理中进行分离，以脱离服务器的管理，同时保持数据文件与日志文件的完整性和一致性。而分离出来的数据库可以附加到其他SQL Server服务器上，构成完整的数据库。分离和附加是系统开发过程中的重要操作。

5.局部变量标识符的定义

- 标识符可以以字母开头，也可以符号`@(`表示局部变量)、`#`(表示临时变量)或者下划线`_`开头，后续字符可以是字母、数字和下划线（_）。
- 标识符不能是Transact-SQL的保留字。
- 标识符中不允许嵌入空格或特殊字符。

```
DECLARE @r INT
SET @r = 5
```

6.了解系统函数的使用

> `数学`

Transact-SQL 中的常用数学函数

函　数	功能描述
ABS	返回表达式的绝对值
ACOS	反余弦函数,返回以弧度表示的角度值
ASIN	反正弦函数，返回以弧度表示的角度值
ATAN	反正切函数，返回以弧度表示的角度值
CEILING	返回大于或等于指定数值表达式的最小整数
COS	返回以以弧度为单位的角度的余弦值
DEGREE	弧度值转换为角度值
EXP	返回给定表达式为指数的 e 值
FLOOR	返回小于或等于指定数值表达式的最大整数
LOG	返回给定表达式的自然对数
LOG10	返回给定表达式的以 10 为底的对数
PI	常量，圆周率
POWER	返回给定表达式的指定次方的值
RADIANS	角度值转换为弧度值
RAND	返回 0~1 之间的随机 float 数
ROUND	返回指定小数的位数的表达式的值
SIN	返回以以弧度为单位的角度的正弦值
SQUARE	返回给定表达式的平方
SQRT	返回给定表达式的平方根
TAN	返回以以弧度为单位的角度的正切值

聚合函数

Transact-SQL 中的聚合函数

函数	功 能 描 述
AVG	返回组中数据的平均值,忽略 NULL 值
COUNT	返回组中项目的数量
MAX	返回多个数据比较的最大值，忽略 NULL 值
MIN	返回多个数据比较的最小值，忽略 NULL 值
SUM	返回组中数据的和，忽略 NULL 值
STDEV	返回给定表达式中所有值的标准偏差
VAR	返回给定表达式中所有值的方差

时间日期函数

Transact-SQL 中的日期时间函数

函 数 名	功 能 描 述
GETDATE	获取当前系统的日期和时间
DATEADD(unit,n,date)	在 date 的基础上添加 n(天/小时/年)后的日期
DATEDIFF(unit,date1,date2)	以 unit 为单位计算日期 1 与日期 2 之间的差值
DATENAME(part,date)	返回指定日期的指定部分(如年/月/日)的字符串形式表示
DATEPART(part,date)	返回指定日期的指定部分(如年/月/日)的整数形式
DAY	获取指定日期的天的日期部分整数
MONTH	获取指定日期的月份的日期部分整数
YEAR	获取指定日期的年份的日期部分整数
GETUTCDATE	获取格林威治的标准时间 datetime 值

常见日期时间函数中的缩写与参数范围

日 期	缩 写	范围
Year(年)	Yy	1753～9999
Quarter(季度)	Qq	1～4
Month(月)	Mm	1～12
Day of Year(一年中的第几天)	Dy	1～366
Day(一月的第几号)	Dd	1～31
Week(一年的第几周)	Wk	1～53
Weekday(一周的星期几)	Dw	1～7（Sunday－Saturday）
Hour(小时)	Hh	0～23
Minute(分钟)	Mi	0～59
Second(秒)	SS	0～59
millisecond	Ms	0～999

转换函数

```
【例6.13】日期和时间函数的使用示例。
PRINT '今天的日期是' + CONVERT(VARCHAR(12), GETDATE(),101)
PRINT '今年是'+CONVERT(VARCHAR(12),Year(Getdate()))
PRINT '本月是'+CONVERT(VARCHAR(12),Month(Getdate()))+'月'
PRINT '今天是'+CONVERT(VARCHAR(12),day(Getdate()))+'号'
PRINT '后天是'+CONVERT(VARCHAR(12),DATEADD(Dy,2,getdate()),101)
PRINT '与2021年6月07号还
差'+CONVERT(VARCHAR(12),DATEDIFF(DAy,getdate(),'06/07/2021'))+'天'
PRINT '现在是星期'+CONVERT(VARCHAR(12),DATEPART(Dw,getdate())-1)
```

字符串函数

常见字符串函数

函数名称	功能描述
ASCII	返回字符表达式最左端字符的 ASCII 代码值
CHAR	将 ASCII 代码转换为字符的字符串函数。
CHARINDEX	返回字符串中指定表达式的起始位置
DIFFERENCE	以整数返回两个字符表达式的 SOUNDEX 值之差。
LEFT	返回从字符串左边开始指定个数的字符
LEN	返回给定字符串 表达式的字符个数，其中不包含尾随空格
LOWER	将大写字符数据转换为小写字符数据后返回字符表达式
LTRIM	删除起始空格后返回字符表达式
NCHAR	根据 Unicode 标准所进行的定义，用给定整数代码返回 Unicode 字符
PATINDEX	返回指定表达式中某模式第一次出现的起始位置；如果在全部有效的文本和字符数据类型中没有找到该模式，则返回 0

常见字符串函数

函数名称	功能描述
QUOTENAME	返回带有分隔符的 Unicode 串
REPLACE	用第 3 个表达式替换第一个字符串表达式中，出现的所有第 2 个给定字符串表达式
REPLICATE	以指定的次数重复字符表达式
REVERSE	返回字符表达式的反转
RIGHT	返回从字符串右边开始指定个数的字符
RTRIM	截断所有尾随空格后返回一个字符串
SOUNDEX	返回由 4 个字符组成的代码（SOUNDEX），以评估两个字符串的相似性。
SPACE	返回由重复的空格组成的字符串。
STR	返回由数字数据转换来的字符数据
STUFF	删除指定长度的字符并在指定的起始点插入另一组字符
SUBSTRING	求子串函数
UNICODE	按照 Unicode 标准的定义，返回输入表达式的第一个字符的整数值
UPPER	返回将小写字符数据转换为大写的字符表达式

7.在游标的执行过程中，@@FETCH_STATUS变量返回值

返回值	描述
0	FETCH 语句成功。
-1	FETCH 语句失败或行数据超出游标数据结果集范围。
-2	提取的行数据不存在。
-9	游标未执行提取操作。

8.两种特殊的表：INSERTED表和DELETED表的相关内容

- 插入表（INSERTED）里存放的是 更新后的数据

> 对于插入记录操作来说，插入表里存放的是要插入的数据；
>
> 对于更新记录操作来说，插入表里存放的是要更新的记录。

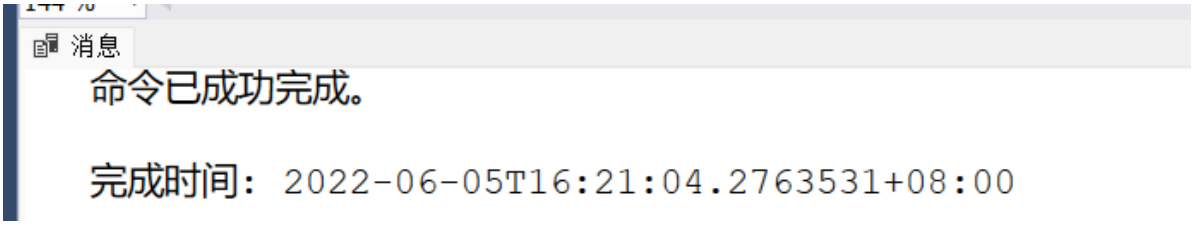
- 删除表（DELETED）里存放的是`更新前的数据`

> 对于更新记录操作来说，删除表里存放的是更新前的记录（更新完后即被删除）
>
> 对于删除记录操作来说，删除表里存入的是被删除的旧记录。

9.如何创建数据库用户名

```
-- 创建登录名和创建用户可以一起写，这里先创建了一个名为Alice的用户，登录密码是：henry626626
CREATE LOGIN Alice
WITH PASSWORD = 'henry626626' /*, DEFAULT_DATABASE = STUDENT; 这里不指定的话，默认为
master数据库*/
GO
-- 给刚刚创建的登录名Alice创建一个数据库用户Alice
CREATE USER Alice FOR LOGIN Alice;
GO
-- 创建没有登录名的用户。不能登录，但可以被授予权限
CREATE USER Mark WITHOUT LOGIN;
```

可以创建与登录名同名的用户名！



10.备份语句

10.1完全备份

```
USE master
GO
BACKUP DATABASE DB_TeachingMS
TO DISK='D:\TeachingMS_Bak\TeachingMS.bak'
GO
```

10.2差异备份

```
USE master
BACKUP DATABASE DB_TeachingMS TO TS_Bak_Device WITH DIFFERENTIAL
GO
```

10.3日志备份

```
USE master
BACKUP LOG DB_TeachingMS TO TS_Bak_Device
GO
```

10.4文件组备份

```
--第一步: 创建数据库
CREATE DATABASE [FileGroupTest]
go
USE [FileGroupTest]

--第二步:创建文件组
ALTER DATABASE [FileGroupTest] ADD FILEGROUP [FG_Test_Id_01]
ALTER DATABASE [FileGroupTest] ADD FILEGROUP [FG_Test_Id_02]

--第三步:创建文件添加到文件组
ALTER DATABASE [FileGroupTest] ADD FILE
(NAME = N'FG_TestUnique_Id_01_data',FILENAME =
N'D:DataFG_TestUnique_Id_01_data.ndf',SIZE = 1MB, FILEGROWTH = 1MB )
TO FILEGROUP [FG_Test_Id_01]

ALTER DATABASE [FileGroupTest] ADD FILE
(NAME = N'FG_TestUnique_Id_02_data',FILENAME =
N'D:DataFG_TestUnique_Id_02_data.ndf',SIZE = 1MB, FILEGROWTH = 1MB )
TO FILEGROUP [FG_Test_Id_02]

--第四步创建表存放在不同文件上
CREATE TABLE  Student(ID INT,Name varchar(50),[Address] varchar(100)) ON
[FG_Test_Id_01]
CREATE TABLE  Teacher(ID INT,Name varchar(50),[Address] varchar(100)) ON
[FG_Test_Id_02]
CREATE TABLE  School(ID INT,Name varchar(50),[Address] varchar(100)) ON [PRIMARY]
```

判断题

1.SQL Server 数据库包含哪些操作系统文件

主文件、辅助数据文件（非必要）、日志文件

2.文件组的作用

文件组是指将数据库相关的一组 `磁盘文件组成的集合` 。SQL Server 2016在创建数据库时会自动创建一个的主文件组，用户也可根据自己的需要自定义一个文件组。

必要性

1. 文件组可以帮助数据库管理人员执行相应的数据布局，以及某些管理任务。
2. 通过创建用户文件组，可以将数据文件集合起来，以便于管理、数据分配和放置。

3.数据库快照snapshot可以由谁创建

`任何能创建数据库的用户` 都可以创建数据库快照。

4.如何对多个变量赋值

	SELECT	SET
同时对多个变量同时赋值时	支持	不支持
表达式返回多个值时	将返回的最后一个值赋给变量	出错
表达式未返回值时	变量保持原值	变量被赋null值

5.简单CASE语句和搜索CASE语句

- 简单 Case 语句将某个表达式与一组简单表达式进行比较，以返回特定的值。

```
Simple Case Statement
CASE [input_expression]
WHEN when_expression THEN when_true_result_expression
[...n]
[ELSE else_result_expression]
END
CASE sex
WHEN '1' THEN '男'
WHEN '2' THEN '女'
ELSE '其他' END
```

- 搜索 Case 语句计算一组布尔表达式，以返回特定的值。

```
Search Case Statement
CASE
WHEN Boolean_expression THEN when_true_result_expression
[...n]
[ELSE else_result_expression]
END
CASE
WHEN sex = '1' THEN '男'
WHEN sex = '2' THEN '女'
ELSE '其他' END
```

6.各种函数返回的值

1. 标量值自定义函数

> **标量值函数返回一个确定类型的标量值，其函数值类型为系统数据类型（除text、ntext、image、 cursor、timestamp、table类型外）。函数体语句定义在BEGIN...END语句内。**

2. 内嵌表值自定义函数

> **内嵌表值函数返回的函数值为一个表。内嵌表值函数的函数体不使用BEGIN...END语句，其返回的表 是RETURN子句中的SELECT命令查询的结果集，其功能相当于一个提供参数化的视图。**

3. 多语句表值自定义函数

> **多语句表值函数可以看作标量函数和内嵌表值函数的结合体。其函数的返回值也是一个表，但函数 体也用BEGIN...END语句定义，返回值的表中的数据由函数体中的语句插入。因此，多语句表值函数 可以进行多次查询，弥补了内嵌表值自定义函数的不足。**

7.输出参数

- 自定义函数不支持输出参数
- 存储过程参数中，添加 OUTPUT 标识符

8.登录名与数据库用户的关系

- 登录名是用来登录SQL Server服务器的登录账户，而数据库用户是登录SQL Server服务器后用来访问具体某个数据库的用户账户
- 一般一个登录名总是与一个或多个数据库用户相关联，这样才能访问对应的数据库
- 要访问特定的数据库还必须具有对应的数据库用户名，而用户名在特定的数据库内创建时，必须关联一个登录名。

9.权限管理相关语句

- 授权

```
GRANT <权限> [, <权限>]
[ON <对象类型> <对象名>]
TO <用户> [, <用户>]
[WITH GRANT OPTION][AS 用户]
```

- 回收权限

```
REVOKE [GRANT OPTION FOR] <权限> [, <权限>]
[ON <对象类型> <对象名>]
FROM <用户> [, <用户>]
[CASCADE] [AS 用户]
```

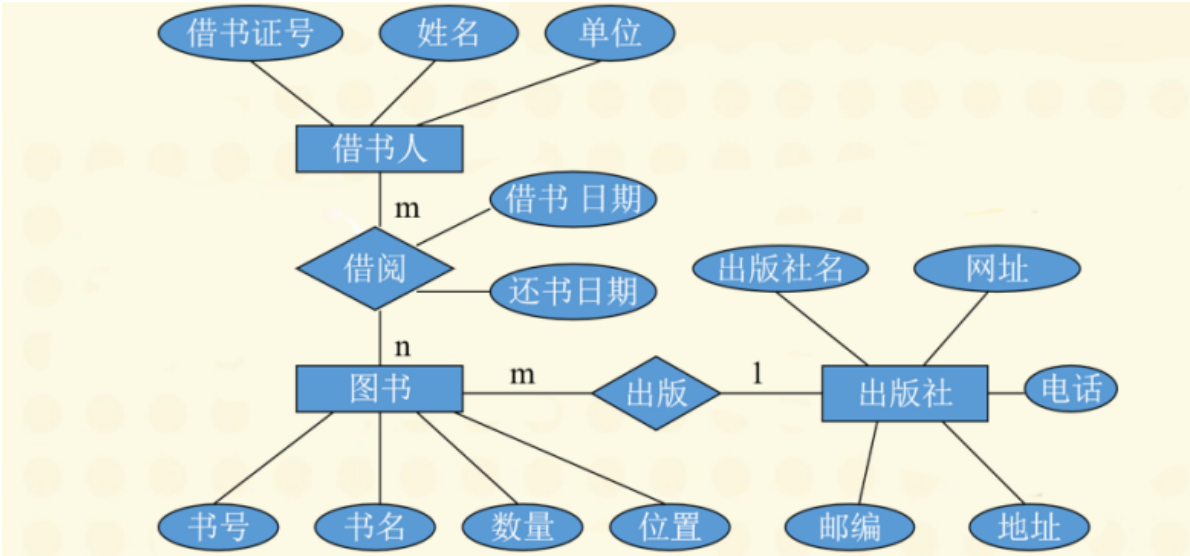
- 拒绝

```
DENY <权限> [, <权限>]
[ON <对象类型> <对象名>]
TO <用户> [, <用户>]
```

10.临时备份与永久备份的差异

- 临时备份：直接指定路径进行备份
- 永久备份：通过逻辑备份设备进行备份

设计题



借书人（借书证号，姓名，单位 ）

图书（书号，书名，数量，位置，出版社名），出版社名是外键

出版社（出版社名，网址，电话，邮编，地址）

借阅（借书证号，书号，借书日期，还书日期），借书证号、书号是外键

简答题

1.改表

- 增加字段

```
ALTER TABLE <表名> Add <字段名> <字段类型>

增加字段Address，数据类型为varchar(60)
ALTER TABLE TB_Teacher ADD Address varchar(60)
```

- 更改字段类型

```
ALTER TABLE <表名> ALTER COLUMN <字段> <字段类型>
alter table 表名 modify 表中字段名 新类型;

将字段“Sex”的字段类型改为char(2)
ALTER TABLE TB_Teacher ALTER COLUMN Sex char(2)
```

- 修改表中字段名

```
alter table 表名 change 旧字段名 新字段名 新字段名类型;
```

- 删除表中字段

```
alter table 表名 drop 字段名;
```

- 修改表名

```
rename table 旧表名 to 新表名;
```

- 向表中插入字段

```
alter table 表名 add 新字段 新字段类型 after 字段名; 将新字段插入到指定字段之后
```

- 创建唯一性约束

```
ALTER TABLE <表名> ADD CONSTRAINT <约束名> UNIQUE (<字段名>)

创建TeacherName为唯一性约束
ALTER TABLE TB_Teacher ADD CONSTRAINT TN UNIQUE (TeacherName)
```

- 设置默认值

```
ALTER TABLE <表名> ADD CONSTRAINT <约束名> DEFAULT <默认值> For <字段名>

添加TitleID的默认值为“T3”
ALTER TABLE TB_Teacher ADD CONSTRAINT DF DEFAULT 'T3' FOR TitleID
```

- 增删改查

```
insert into 表名(字段名1,…,字段N) values(value1,…,valueN);
select * from 表名; 其中星号*表示表中所有字段
update 表名 set 要修改的字段名=value;
delete from 表名 where 条件;
```

2.查询

- 单表查询

```
查询学生姓名包含“丽”这个字的学生的所有信息，并按学号排序 %任意字符 _单个字符 DESC降序排列
select * from TB_Student where StuName like '%丽%' order by StuID

查询1985年后出生的男生信息
select * from TB_Student where year(Birthday)>= '1985' and Sex = 'M'

其他关键字的使用 count() max() min() avg() between..and..
group by having..

查询成绩表中总评平均成绩高于75分的班级编号，列出班级编号及平均分，按总评平均成绩降序排列。
select ClassID round(avg(TotalScore),1) as 平均分 from TB_Grade group by
ClassID
having avg(TotalScore)>75 order by avg(TotalScore) desc
```

- 多表查询

```
根据课程班和任课教师对学生平均分进行分类汇总，并显示平均成绩大于75分的信息，要求显示标题为：
课程班
号、教师姓名、平均分。
select b.CourseClassID 课程班号, a.TeacherName 教师姓名,avg(b.TotalScore) 平均分
from
TB_Teacher a,TB_Grade b,TB_CourseClass c where b.CourseClassID =
c.CourseClassID
and c.TeacherID = a.TeacherID group by
b.CourseClassID,a.TeacherID,a.TeacherName
having avg(b.TotalScore) > 75;

左外连接(左边的表不加限制) 返回包括左表中的所有记录和右表中连接字段相等的记录,如果右表没有
则为null
```

```
右外连接(右边的表不加限制)

统计每个学生的选课数（包括没有选课的学生），列出学号、姓名、选课门数。
select a.StuID,a.StuName,count(CourseClassID) 选课门数 from TB_Student a left join
TB_SelectCourse b on a.StuID = b.StuID group by a.StuID,a.StuName;

找出没有选修任何课程的学生的学号、姓名及所在系名
select StuID,StuName,DeptName from TB_Student a,TB_Dept b
where a.DeptID = b.DeptID and StuID not in
(select distinct StuID from TB_SelectCourse)
```

3.权限管理

- 授权

```
GRANT <权限> [, <权限>]
[ON <对象类型> <对象名>]
TO <用户> [, <用户>]
[WITH GRANT OPTION][AS 用户]

GRANT SELECT ON TB_Student TO Student
GRANT Update ON TB_CourseClass TO Teachers
```

- 回收权限

```
REVOKE [GRANT OPTION FOR] <权限> [, <权限>]
[ON <对象类型> <对象名>]
FROM <用户> [, <用户>]
[CASCADE] [AS 用户]

REVOKE SELECT ON TB_Student TO Student
REVOKE Update ON TB_CourseClass TO Teachers
```

- 拒绝

```
DENY <权限> [, <权限>]
[ON <对象类型> <对象名>]
TO <用户> [, <用户>]

DENY SELECT ON TB_Student TO Student
DENY Update ON TB_CourseClass TO Teachers
```

4.备份

- 创建备份设备

```
EXEC sp_addumpdevice 'DISK', 'TS_Bak_Device',
'D:\TS_Bak_Device\TeachingMS.bak'
```

- 完全备份

```
BACKUP DATABASE DB_TeachingMS TO TS_Bak_Device
```

- 差异备份

```
BACKUP DATABASE DB_TeachingMS TO TS_Bak_Device WITH DIFFERENTIAL
```

- 日志备份

```
BACKUP LOG DB_TeachingMS TO TS_Bak_Device
```

5.存储过程

```
CREATE PROCEDURE SP_GradeProc @CourseClassID CHAR(10)
AS
------定义课程考试比例系数变量并获取相应的值-------
DECLARE @CPart REAL,@MPart REAL,@LPart REAL
SELECT @CPart=CommonPart,@MPart=MiddlePart,@LPart=LastPart
FROM TB_CourseClass
WHERE CourseClassID=@CourseClassID
------定义用来存放平时、期中、期末、总评成绩变量-------
DECLARE @CScore REAL,@MScore REAL,@LScore REAL,@TotalScore REAL
------------------声明游标------------------
DECLARE CUR_GradeProc CURSOR FOR
SELECT CommonScore,MiddleScore,LastScore FROM TB_Grade
WHERE CourseClassID=@CourseClassID ORDER BY StuID
------------------打开游标------------------
OPEN CUR_GradeProc
-----------循环提取游标中成绩并处理------------
FETCH NEXT FROM CUR_GradeProc INTO @CScore,@MScore,@LScore
WHILE @@FETCH_STATUS = 0
BEGIN
SET @TotalScore = ROUND((@CScore*@CPart+@MScore*@MPart+@LScore*@LPart)/100,0)
UPDATE TB_Grade SET TotalScore=@TotalScore
WHERE CURRENT OF CUR_GradeProc
FETCH NEXT FROM CUR_GradeProc INTO @CScore,@MScore,@LScore
END
----------------关闭释放游标----------------
CLOSE CUR_GradeProc
DEALLOCATE CUR_GradeProc
```

6.函数

```
编一函数，要求输入学生姓名，返回该学生的选课门数
create function CourseCount(@sname char(8))
returns int
as
begin
Return(select count(*) from TB_SelectCourse TBSC join TB_Student TBS on
TBSC.StuID=TBS.StuID
where StuName=@sname)
End


执行用户自定义函数
[database_name.]owner_name.function_name ([argument_expr] [, …] )
```

```
SELECT dbo.DatetoQuarter ('2021-3-2')
```

7.AFTER触发器

- 实现选课人数自减功能

```
USE DB_TeachingMS
GO
CREATE TRIGGER TR_SelectCourse_addNum
ON TB_SelectCourse AFTER DELETE
AS
UPDATE TB_CourseClass SET SelectedNumber = SelectedNumber-1
WHERE CourseClassID = (SELECT CourseClassID FROM DELETED)
```

- 设计一触发器，要求在TB_Student中加入学生时，自动将对应班级表中的ClassNumber加1

```
CREATE TRIGGER TR_AddStudent
ON TB_Student AFTER INSERT
AS
UPDATE TB_Class SET ClassNumber = ClassNumber+1
WHERE ClassID = (SELECT ClassID FROM INSERTED)
```

- 当向TB_Grade表插入记录后，如果成绩非空则在该学生的TotalGrade中自动加上该门课程的得分。

```
CREATE TRIGGER TR_AddStuScore
ON TB_Grade AFTER INSERT
AS
UPDATE TB_Student SET TotalGrade = TotalGrade +INSERTED.TotalScore
FROM TB_Student TS,INSERTED
WHERE TS.StuID = INSERTED.StuID
```

- 在TB_Grade中插入成绩时，如果成绩大于等于60分则在该学生的TotalCredit中自动加上相应课程的学分

```
create Trigger SumCredit on TB_Grade after insert
as
declare @credit real,@grade real
set @grade=(select TotalScore from inserted)
if (@grade>=60)
begin
set @credit=(select CourseGrade from TB_Course where CourseID=(select
CourseID from INSERTED))
update TB_Student set TotalCredit=TotalCredit+@credit where StuID=(select
StuID from INSERTED)
end
```

其他

1.编写一个程序，实现输出圆的周长和圆的面积，其中圆的半径为5

```
DECLARE @r INT
SET @r = 5
PRINT 2*3.14*@r
PRINT 3.14*@r*@r
```

2.输出学生表中当前学生的总人数，并据总人数进行判定

2.1 IF语句

```
DECLARE @total INT
SELECT @total=COUNT(*) FROM TB_Student
IF @total>100
BEGIN
    PRINT '学生过多，请缩减招生'
END
ELSE IF @total<10
BEGIN
    PRINT '学生过少，请扩大招生'
END
ELSE
BEGIN
    print @total
END
```

2.2 CASE语句

```
DECLARE @total INT
SELECT @total=COUNT(*) FROM TB_Student
PRINT
CASE
    WHEN @total>100 THEN '学生过多，请缩减招生'
    WHEN @total<10 THEN '学生过少，请扩大招生'
    ELSE CAST(@total AS VARCHAR(20))
END
```

3.编写一个程序，实现输出1-100之间的整数和

```
DECLARE @ans INT
SET @ans=0
DECLARE @i INT
SET @i=1
WHILE @i<=100
BEGIN
    SET @ans=@ans+@i
    SET @i = @i+1
END
PRINT @ans
```

4.创建一个用户自定义函数，根据给定的学生姓名返回该学生所获得的学分总和

4.1创建自定义函数

```
CREATE FUNCTION f(@name VARCHAR(255)) RETURNS INT
AS
BEGIN
    DECLARE @total INT
    SELECT @total = SUM(TB_Course.CourseGrade) FROM TB_Grade,TB_Course,TB_Student
                WHERE TB_Course.CourseID=TB_Grade.CourseID AND
TB_Student.StuID=TB_Grade.StuID AND
                      TB_Student.StuName = @name AND TB_Grade.TotalScore>=60
    RETURN @total
END
```

4.2调用函数

```
DECLARE @ans INT
SELECT @ans = dbo.f('王倩')
PRINT @ans
```

5.创建一个存储过程，根据给定的课程班编号返回课程班人数

5.1创建存储过程

```
CREATE PROCEDURE GetSelectCoursePeople @courseClassID VARCHAR(20)
AS
BEGIN
    SELECT COUNT(*) FROM TB_SelectCourse WHERE CourseClassID=@courseClassID
END
```

5.2调用存储过程

```
EXEC GetSelectCoursePeople 'T080040401'
```

6.创建一个带OUTPUT的存储过程，根据给定的课程班编号返回课程班人数

6.1创建存储过程

```
CREATE PROCEDURE GetSelectCoursePeopleOut @courseClassID VARCHAR(20),@total INT
OUTPUT
AS
BEGIN
    SELECT @total=COUNT(*) FROM TB_SelectCourse WHERE CourseClassID=@courseClassID
END
```

6.2调用存储过程

```
DECLARE @ans INT
EXEC GetSelectCoursePeopleOut 'T080040401',@ans OUTPUT
PRINT @ans
```

7.创建一个游标，用以获取一个结果集，内容为课程班“T080010401”的学生成绩（学号、姓名、课程名称、总评成绩）并通过游标将这些数据逐行提取出来显示。

```
DECLARE @StuID CHAR(8),@StuName CHAR(8),@CourseName VARCHAR(32),@TotalScore REAL

DECLARE CUR_CourseClassGrade CURSOR
FOR
SELECT
TB_Student.StuID,TB_Student.StuName,TB_Course.CourseName,TB_Grade.TotalScore
    From TB_Student,TB_Course,TB_Grade
    WHERE TB_Student.StuID=TB_Grade.StuID AND
TB_Course.CourseID=TB_Grade.CourseID AND
          TB_Grade.CourseClassID='T080010401'

OPEN CUR_CourseClassGrade
FETCH NEXT FROM CUR_CourseClassGrade
INTO @StuID,@StuName,@CourseName,@TotalScore
WHILE @@FETCH_STATUS=0
BEGIN
    PRINT @StuID+'|'+@StuName+'|'+@CourseName+'|'+CAST(@TotalScore AS
VARCHAR(30))
    FETCH NEXT FROM CUR_CourseClassGrade
    INTO @StuID,@StuName,@CourseName,@TotalScore
END
CLOSE CUR_CourseClassGrade
DEALLOCATE CUR_CourseClassGrade
```

8.创建触发器，实现选课人数自动加一

```
CREATE TRIGGER TR_SelectCourse ON TB_SelectCourse AFTER INSERT
AS
    UPDATE TB_CourseClass SET SelectedNumber = SelectedNumber+1
    WHERE CourseClassID = (SELECT CourseClassID FROM INSERTED)
```

9.请用INSTEAD OF触发器实现“TB_SelectCourse”表中的选课记录不能被更新修改这一功能

```
CREATE TRIGGER TR_UpdateSelectCourse ON TB_SelectCourse INSTEAD OF UPDATE
AS
    PRINT '学生选课信息不能被修改！'
```