经典SQL语句

查所有加入域的人的信息

```
SELECT
    sAMAccountname AS AccountName,
    GivenName AS [First Name],
    SN AS [Last Name],
    mail AS [E-mail Address],
    TelephoneNumber AS [Telephone Number],
    l AS Office,
    PostalAddress AS [Postal Address],
    PostalCode,
    StreetAddress AS [Street Address],
    wWWHomePage AS [Web Page],
    distinguishedname AS DN
FROM
    OPENQUERY(
        ADSI,
        'SELECT GivenName, SN, company, department, TelephoneNumber, mail, distinguishedname,
title, l, manager, mobile, name, PhysicalDeliveryOfficeName, PostalAddress, PostalCode,
sAMAccountname, StreetAddress, wWWHomePage FROM ''LDAP://itg.net/ OU=itg ,DC=itg,DC=net''
WHERE objectCategory = ''Person'' AND objectClass = ''user'' ORDER BY Name'
    ) Rowset_1

--where sAMAccountname ='clw' --查个人信息



--查部门信息

SELECT
*
FROM
    OPENQUERY(
        ADSI,
        'SELECT GivenName, SN, company, department, TelephoneNumber, mail, distinguishedname,
title, l, manager, mobile, name, PhysicalDeliveryOfficeName, PostalAddress, PostalCode,
sAMAccountname, StreetAddress, wWWHomePage FROM ''LDAP://itg.net/ OU=itg ,DC=itg,DC=net''
WHERE objectCategory = ''Person'' AND objectClass = ''user'' ORDER BY Name'
    ) Rowset_1

where distinguishedname like '%OU=财务部,%'  --查 IT 部门的域信息
```

将整型数字转换为大写汉字的自定义函数,如转换为'壹贰叁零捌肆'

```
Create  FUNCTION [dbo].[f_digit_chn](@num bigint)
RETURNS varchar(20)
AS
BEGIN
-- 调用 select dbo.[f_digit_chn](12345678)
  DECLARE @result varchar(20),@symbol varchar(2)
  IF @num<0
    SELECT @symbol='负',@result='',@num=ABS(@num)
  ELSE
    SELECT @symbol='',@result=''
  WHILE @num<>0
    SELECT @result=SUBSTRING('零壹贰叁肆伍陆柒捌玖拾',@num%10+1,1)+@result,@num=@num/10
  RETURN @symbol+@result
END
```

存储过程如下:

```
CREATE PROC sp_PageView
@tbname    sysname,            --要分页显示的表名
@FieldKey  sysname,            --用于定位记录的主键(惟一键)字段,只能是单个字段
@PageCurrent int=1,              --要显示的页码
@PageSize  int=10,            --每页的大小(记录数)
@FieldShow  nvarchar(1000)='',  --以逗号分隔的要显示的字段列表,如果不指定,则显示所有字段
@FieldOrder  nvarchar(1000)='',  --以逗号分隔的排序字段列表,可以指定在字段后面指定 DESC/ASC 用于
指定排序顺序
@Where    nvarchar(1000)='',  --查询条件
@PageCount  int OUTPUT       --总页数
AS
DECLARE @sql nvarchar(4000)
SET NOCOUNT ON
--检查对象是否有效
IF OBJECT_ID(@tbname) IS NULL
BEGIN
RAISERROR(N'对象"%s"不存在',1,16,@tbname)
RETURN
END
IF OBJECTPROPERTY(OBJECT_ID(@tbname),N'IsTable')=0
AND OBJECTPROPERTY(OBJECT_ID(@tbname),N'IsView')=0
AND OBJECTPROPERTY(OBJECT_ID(@tbname),N'IsTableFunction')=0
```

```
BEGIN
RAISERROR(N'"%s"不是表、视图或者表值函数',1,16,@tbname)
RETURN
END


--分页字段检查
IF ISNULL(@FieldKey,N'')=''
BEGIN
RAISERROR(N'分页处理需要主键（或者惟一键）',1,16)
RETURN
END


--其他参数检查及规范
IF ISNULL(@PageCurrent,0) <1 SET @PageCurrent=1
IF ISNULL(@PageSize,0) <1 SET @PageSize=10
IF ISNULL(@FieldShow,N'')=N'' SET @FieldShow=N'*'
IF ISNULL(@FieldOrder,N'')=N''
SET @FieldOrder=N''
ELSE
SET @FieldOrder=N'ORDER BY '+LTRIM(@FieldOrder)
IF ISNULL(@Where,N'')=N''
SET @Where=N''
ELSE
SET @Where=N'WHERE ('+@Where+N')'


--如果@PageCount 为 NULL 值,则计算总页数(这样设计可以只在第一次计算总页数,以后调用时,把总页数传回给存储过程,避免再次计算总页数,对于不想计算总页数的处理而言,可以给@PageCount 赋值)
IF @PageCount IS NULL
BEGIN
SET @sql=N'SELECT @PageCount=COUNT(*)'
  +N' FROM '+@Where
EXEC sp_executesql @sql,N'@PageCount int OUTPUT',@PageCount OUTPUT
SET @PageCount=(@PageCount+@PageSize-1)/@PageSize
END


--计算分页显示的 TOPN 值
DECLARE @TopN varchar(20),@TopN1 varchar(20)
SELECT @TopN=@PageSize,
@TopN1=@PageCurrent*@PageSize
```

```
--第一页直接显示
IF @PageCurrent=1
EXEC(N'SELECT TOP '+@FieldShow
  +N' FROM '+@Where
  +N' '+@TopN
  +N' THEN @s+N'','''+QUOTENAME(RTRIM(CAST('+@tbname
  +N' '+@FieldOrder
SET ROWCOUNT @PageCurrent
EXEC sp_executesql @sql,
  '+@FieldShow
  +N' FROM '+@TopN
  +N' '+@tbname
  +N' WHERE '+@sql
  +N') '+@FieldOrder)
END
END
```

将整型数字转换为大写汉字的自定义函数,如转换为'壹贰叁零捌肆'

调用 select dbo.[f_digit_chn](12345678)

```
Create  FUNCTION [dbo].[f_digit_chn](@num bigint)
RETURNS varchar(20)
AS
BEGIN
  DECLARE @result varchar(20),@symbol varchar(2)
  IF @num<0
    SELECT @symbol='负',@result='',@num=ABS(@num)
  ELSE
    SELECT @symbol='',@result=''
  WHILE @num<>0
    SELECT @result=SUBSTRING('零壹贰叁肆伍陆柒捌玖拾',@num%10+1,1)+@result,@num=@num/10
  RETURN @symbol+@result
END
```

将身份证的位号码升级为位

```
-- --调用函数
-- update
--     表
-- set
--     身份证号= dbo.ID15TO18(身份证号)
-- where
--    LEN(身份证号) = 15
Create FUNCTION [dbo].[ID15TO18] (@id15 char(15))
RETURNS CHAR(18)
AS
BEGIN
   DECLARE @ID18 CHAR(18)

   DECLARE @S1 AS INTEGER
   DECLARE @S2 AS INTEGER
   DECLARE @S3 AS INTEGER
   DECLARE @S4 AS INTEGER
   DECLARE @S5 AS INTEGER
   DECLARE @S6 AS INTEGER
   DECLARE @S7 AS INTEGER
   DECLARE @S8 AS INTEGER
   DECLARE @S9 AS INTEGER
   DECLARE @S10 AS INTEGER
   DECLARE @S11 AS INTEGER
   DECLARE @S12 AS INTEGER
   DECLARE @S13 AS INTEGER
   DECLARE @S14 AS INTEGER
   DECLARE @S15 AS INTEGER
   DECLARE @S16 AS INTEGER
   DECLARE @S17 AS INTEGER
   DECLARE @S18 AS INTEGER

   SET @S1=SUBSTRING(@ID15,1,1)
   SET @S2=SUBSTRING(@ID15,2,1)
   SET @S3=SUBSTRING(@ID15,3,1)
   SET @S4=SUBSTRING(@ID15,4,1)
   SET @S5=SUBSTRING(@ID15,5,1)
   SET @S6=SUBSTRING(@ID15,6,1)
   SET @S7=1
   SET @S8=9
   SET @S9=SUBSTRING(@ID15,7,1)
   SET @S10=SUBSTRING(@ID15,8,1)
   SET @S11=SUBSTRING(@ID15,9,1)
   SET @S12=SUBSTRING(@ID15,10,1)
   SET @S13=SUBSTRING(@ID15,11,1)
   SET @S14=SUBSTRING(@ID15,12,1)
   SET @S15=SUBSTRING(@ID15,13,1)
   SET @S16=SUBSTRING(@ID15,14,1)
   SET @S17=SUBSTRING(@ID15,15,1)
```

```
    SET @S18=((@S1*7)+(@S2*9)+(@S3*10)+(@S4*5)+(@S5*8)+(@S6*4)+(@S7*2)+(@S8*1)
        +(@S9*6)+(@S10*3)+(@S11*7)+(@S12*9)+(@S13*10)+(@S14*5)+(@S15*8)+(@S16
        *4)+(@S17*2))%11

    SET @ID18=SUBSTRING(@ID15,1,6)+'19'+SUBSTRING(@ID15,7,9)
        +CASE WHEN @S18=0 THEN '1'
              WHEN @S18=1 THEN '0'
              WHEN @S18=2 THEN 'X'
              WHEN @S18=3 THEN '9'
              WHEN @S18=4 THEN '8'
              WHEN @S18=5 THEN '7'
              WHEN @S18=6 THEN '6'
              WHEN @S18=7 THEN '5'
              WHEN @S18=8 THEN '4'
              WHEN @S18=9 THEN '3'
              WHEN @S18=10 THEN '2'
         END

    RETURN @ID18
END
```

四舍五入加千分位

```
create function GetFormatString(@dec decimal(28,8), @n int)
returns varchar(32) as
begin
    declare @str varchar(32), @len int, @left varchar(32), @right varchar(32),@end
varchar(32)
    if @n!='0'
  BEGIN

    set @str= round(@dec,@n)
    select @left=left(@str,charindex('.',@str)-1),@len=len(@left)-2
    while @len>1
    begin
        select @left=stuff(@left,@len,0,','), @len=@len-3
    end
    select @right=left(stuff(@str,1,charindex('.',@str),''), @n),@len=4
    while @len <=len(@right)
    begin
        select @right=stuff(@right,@len,0,','), @len=@len+4
    end
    set @end= @left+'.'+@right
```

```
end
   else
BEGIN

   set @str= round(@dec,@n)
   select @left=left(@str,charindex('.',@str)-1),@len=len(@left)-2
   while @len>1
   begin
      select @left=stuff(@left,@len,0,','), @len=@len-3
   end
   select @right=left(stuff(@str,1,charindex('.',@str),''), @n),@len=4
   while @len <=len(@right)
   begin
      select @right=stuff(@right,@len,0,','), @len=@len+4
   end
   set @end= @left
end
return @end
end

select dbo.GetFormatString(123645.7889, 2)+'±'+'MON'
union all select dbo.GetFormatString(123645.7889, 3)+'±'+'MON'
union all select dbo.GetFormatString(123645.7889, 4)+'±'+'MON'
```

SQL 2005 当中绘画日历

```
Set Nocount On
Declare
   @Date datetime,
   @StartDate datetime,
   @EndDate datetime,
   @FirstIndex int
Set @Date ='20080222' --输入一个日期，即可算出当月的日历
Select
   @StartDate=Convert(char(6),@Date,112)+'01',
   @EndDate=Dateadd(month,1,@StartDate)-1,
   @FirstIndex=Datediff(day,-1,@StartDate)%7
;With t As
(
   Select Date=Convert(int,1),Row=(@FirstIndex)/7,Col=@FirstIndex
   Union All
   Select Date=Date+1,Row=(@FirstIndex+Date)/7,Col=(Date+@FirstIndex)%7
   From t
   Where Date<=Datediff(day,@StartDate,@EndDate)
)
Select
    [日]=Isnull(Convert(char(2),[0]),''),
    [一]=Isnull(Convert(char(2),[1]),''),
    [二]=Isnull(Convert(char(2),[2]),''),
    [三]=Isnull(Convert(char(2),[3]),''),
    [四]=Isnull(Convert(char(2),[4]),''),
    [五]=Isnull(Convert(char(2),[5]),''),
    [六]=Isnull(Convert(char(2),[6]),'')
From t
Pivot (Max(Date) For col In([0],[1],[2],[3],[4],[5],[6])) b
```

```
2008年月份
------------
日   一   二  三  四  五  六
---- ---- ---- ---- ---- ---- ----
                         1     2
3    4    5    6    7    8    9
10   11   12   13   14   15   16
17   18   19   20   21   22   23
24   25   26   27   28   29
```

功能：当中绘画日历

```
DECLARE @Year nvarchar(4)
DECLARE @YearMonth nvarchar(7)    --月份
DECLARE @strTop nvarchar(200)
DECLARE @ForI INT,@ForYear INT ,@MaxDay INT
DECLARE @RowX INT --行位置
DECLARE @strWeekDayList nvarchar(20)
DECLARE @strPrint nvarchar(300)

-- =======================================
SET @Year='2008'    --请在这里输入年份
-- =======================================
SET  @strTop=  ' 日 '+char(9)+' 一 '+char(9)+' 二 '  +char(9)+' 三 '++char(9)+' 四 '++char(9)+' 五 '++char(9)+'六' +char(13)+
        '____________________________'
SET @strWeekDayList='日一二三四五六'
SET @ForYear=1
WHILE @ForYear<=12  --1 月份至月份
 BEGIN
    SET @YearMonth=@Year + '-' +CAST( @ForYear AS nvarchar(2))
    SET @MaxDay=DAY(DATEADD(Day,-1,DATEADD(Month,1,@YearMonth+'-01')))
    SET @RowX=CHARINDEX(RIGHT(DATENAME(WeekDay,@YearMonth+'-01'),1),@strWeekDayList)-1
    SET @strPrint=''
    SET @ForI=1
    WHILE @ForI<=@RowX    --构造号的位置
        BEGIN
       SET @strPrint=@strPrint+CHAR(9)
       SET @ForI=@ForI+1
        END
    SET @ForI=1
    WHILE @ForI<=@MaxDay    --构造号到月底的位置
        BEGIN
       SET @strPrint=@strPrint+CAST(@ForI AS nvarchar(2)) +Char(9)
       SET @RowX=@RowX+1
       SET @ForI=@ForI+1
       IF (@RowX%7=0)
          BEGIN
           SET @RowX=0
           SET @strPrint=@strPrint+CHAR(13)
            END
      END
    SET @ForYear=@ForYear+1
    -- 打印输出一个月的结果
    PRINT '──────────────────────────────────────────────'
    PRINT +Char(9)++Char(9)+'    '+@YearMonth+CHAR(10)
    PRINT @strTop
    PRINT @strPrint +CHAR(10)
  END
```

获取 M$ SQL Server 用户表的字段信息

```
USE database1
SELECT
  表名  = CASE a.colorder WHEN 1 THEN c.name ELSE '' END,
  序    = a.colorder,
  字段名= a.name,
  标识  = CASE COLUMNPROPERTY(a.id,a.name,'IsIdentity') WHEN 1 THEN '√' ELSE '' END,
  主键  = CASE
    WHEN EXISTS (
      SELECT *
      FROM sysobjects
      WHERE xtype='PK' AND name IN (
        SELECT name
        FROM sysindexes
        WHERE id=a.id AND indid IN (
          SELECT indid
          FROM sysindexkeys
          WHERE id=a.id AND colid IN (
            SELECT colid
            FROM syscolumns
            WHERE id=a.id AND name=a.name
          )
        )
      )
    )
    THEN '√'
    ELSE ''
  END,
  类型  = b.name,
  字节数= a.length,
  长度  = COLUMNPROPERTY(a.id,a.name,'Precision'),
  小数  = CASE ISNULL(COLUMNPROPERTY(a.id,a.name,'Scale'),0)
    WHEN 0 THEN ''
    ELSE CAST(COLUMNPROPERTY(a.id,a.name,'Scale') AS VARCHAR)
  END,
  允许空= CASE a.isnullable WHEN 1 THEN '√' ELSE '' END,
  默认值= ISNULL(d.[text],''),
  说明  = ISNULL(e.[value],'')
FROM syscolumns a
  LEFT  JOIN systypes      b ON a.xtype=b.xusertype
  INNER JOIN sysobjects    c ON a.id=c.id AND c.xtype='U' AND c.name<>'dtproperties'
  LEFT  JOIN syscomments   d ON a.cdefault=d.id
  LEFT  JOIN sysproperties e ON a.id=e.id AND a.colid=e.smallid
ORDER BY c.name, a.colorder
```

银行卡信息

```
USE MASTER--连接系统数据库
IF EXISTS(select 1 from master..sysdatabases where name='bankDB')
    DROP DATABASE bankDB
GO
------------------------------------------------------------建库-----------------------
--打开外围服务器
EXEC sp_configure 'show advanced options', 1
GO
RECONFIGURE
GO
EXEC sp_configure 'xp_cmdshell', 1
GO
RECONFIGURE
GO
--新建文件夹
```

```
exec xp_cmdshell 'MD e:\数据库'

CREATE DATABASE bankDB
ON
(
    NAME ='bankDB',
    FILENAME='e:\数据库\bankDB.mdf',
    SIZE = 10,
    MAXSIZE=500,
    FILEGROWTH=15%
)
GO
--------------------------------------------------建表-----------------
USE bankDB
--用户信息表
IF EXISTS(select 1 from bankDB..sysobjects where name='userInfo')
    DROP TABLE userInfo
GO
CREATE TABLE userInfo
(
    customerID          int identity(1,1) PRIMARY KEY ,    --顾客编号(自动增长主键)
    customerName       varchar(20) not null,           --开户名
    PID                 varchar(20)UNIQUE not null,       --身份证(18-15 位数唯一约束)
    telephone          varchar(20) not null,           --联系电话(****-********或手机号位数)
    [address]          ntext                           --联系地址
)
--银行卡信息表
IF EXISTS(select 1 from bankDB..sysobjects where name='cardInfo')
    DROP TABLE cardInfo
GO
CREATE TABLE cardInfo
(
    cardID              varchar(20) primary key ,          --卡号  (格式为 3576 **** ***(*
部分是随机产生))
    curType              varchar(10) default('RMB') not null,--货币种类(默认为 RMB)
    savingType          varchar(10),                       --存款类型(活期/定活两便/定期)
    openDate          datetime default(getdate()) not null,--开户日期(默认为当前时间)
    openMoney           money check(openMoney<1) not null,    --开户金额(不能低于元)
    dalance               money check(dalance<1) not null,    --余额(不能低于元否则将销户)
    pass              varchar(20) default(888888) not null,--密码(6 位数开户时默认为个)
    IsReportLoss       bit default(0) not null,            --是否过失(是/否默认为否 1 是 0 否)
    customerID           int not null                        --顾客编号(外键该卡号对应的顾客编号一
个用户可办多张卡)
)
```

```
--交易信息表
IF EXISTS(select 1 from bankDB..sysobjects where name='transInfo')
   DROP TABLE transInfo
GO
CREATE TABLE transInfo
(
   transDate          datetime not null,       --交易日期(默认为当前时间)
   cardID              varchar(20) not null,    --卡号(外键可重复索引)
   transType          varchar(10) not null,    --交易类型(只能是存入/支取)
   transMoney          money check(transMoney>0) not null,    --交易金额(大于)
   remark             ntext                 --备注(其它说明)
)
------------------------------约束------------------------------
--约束电话和手机号码
if(object_id('uq_pid') is not null)
begin
   alter table userinfo
      drop constraint uq_pid
end
if(object_id('ck_PID') is not null)
begin
   alter table userinfo
      drop constraint ck_PID
end
if(object_id('ck_telephone') is not null)
begin
   alter table userinfo
      drop constraint ck_telephone
end
alter table userInfo
add constraint ck_PID check(len(pid) in (15,18)),
   constraint uq_PID unique(pid),
   constraint       ck_telephone       check(telephone                                like
'1[35][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'--约束手机号码
                          or
                             telephone                                                like
'[0-9][0-9][0-9]-[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'--010-12345678
                          or
                             telephone                                                like
'[0-9][0-9][0-9][0-9]-[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'--0719-12345678
                          or
                             telephone                                                like
'[0-9][0-9][0-9][0-9]-[0-9][0-9][0-9][0-9][0-9][0-9][0-9]')--0719-1234567
```

```
---------------------------------------------- 设 置 卡 号 为 随 机 数 ( 方
法)---------------------------
declare @r decimal(10,8)
declare @time varchar(25)
set @time=convert(varchar,getdate(),120)+convert(varchar,datepart(ms,getdate()))
set @time=replace(@time,' ','')
set @time=replace(@time,':','')
set @time=replace(@time,'-','')

set @time=substring(@time,8,len(@time)-1)
--select @time--查看获得的随机数

select @r=rand(convert(bigint,@time))
set @time=cast(@r as varchar)
--select @time
set @time=substring(@time,3,len(@time)-1)
print @time
print '1001'+convert(varchar(10),@time)

---------------------------------------------- 设 置 卡 号 为 随 机 数 ( 方 法 用 了 存 储 过
程)---------------------------
create proc proc_randCardID
@cardid varchar(19) output
as
declare @r numeric(8,8)
set @cardid='1010 3657 '
while(1=1)
begin
   set
@r=rand(datepart(mm,getdate())*100000+datepart(ss,getdate())*1000+datepart(ms,getdate(
)))
   declare @temp char(8)
   set @temp=substring(convert(varchar,@r),3,8)
   set @cardid=@cardid+substring(@temp,1,4)+' '+substring(@temp,5,4)
   if not exists(select 1 from cardinfo where cardid=@cardid)
      break
end
--测试(调用存储过程)
declare @card varchar(19)
exec proc_randCardID @card output
print @card
*/
```

功能概述：显示某一表的结构

```
DECLARE @tableName nvarchar(100)
SET @tableName ='brand'

--mssql2005
SELECT  (
       CASE WHEN a.colorder=1 THEN d.name ELSE '' END)表名,
       a.colorder 字段序号,
       a.name 字段名,
       (CASE WHEN COLUMNPROPERTY( a.id,a.name,'IsIdentity')=1 THEN '√' ELSE '' END) 标识,
       (CASE WHEN (
          SELECT COUNT(*)
          FROM sysobjects
          WHERE (name IN
                  (SELECT name
                  FROM sysindexes
                  WHERE (id = a.id) AND (indid IN
                          (SELECT indid
                          FROM sysindexkeys
                        WHERE (id = a.id) AND (colid IN
                                (SELECT colid
                                FROM syscolumns
                                WHERE (id = a.id) AND (name = a.name))))))) AND
                (xtype = 'PK'))>0 THEN '√' ELSE '' END) 主键,
       b.name 类型,
       a.length 占用字节数,
       COLUMNPROPERTY(a.id,a.name,'PRECISION') AS 长度,
       ISNULL(COLUMNPROPERTY(a.id,a.name,'Scale'),0) AS 小数位数,
       (CASE WHEN a.isnullable=1 THEN '√' ELSE '' END) 允许空,
       ISNULL(e.text,'') 默认值,
       ISNULL(g.[value],'') AS 字段说明
FROM syscolumns a
   LEFT JOIN systypes b ON a.xtype=b.xusertype
   INNER JOIN sysobjects d ON a.id=d.id AND d.xtype='U' AND d.name <>'dtproperties'
   LEFT JOIN syscomments e ON a.cdefault=e.id
   LEFT JOIN sys.extended_properties g  ON a.id=g.major_id AND a.colid = g.major_id
WHERE d.name=@tableName
ORDER BY a.id,a.colorder
```

功能概述:重新编译一个表上的所有索引

```
alter index all on order_master rebuild
```

主要掌握 WHERE 子句限制输出行

使用比较条件

使用 BETWEEN，IN，AND,NOT,OR 操作

最后掌握使用 ORDER BY 子句排序输出行

所有操作在数据库 pubs 里进行

```
select * from employee

use master    /*显示表结构*/
exec sp_helpdb
GO

use pubs
exec sp_help
GO

use pubs
exec sp_help employee
GO

select lname,job_lvl
from employee
where job_lvl<=50

select lname,job_lvl/*使用 BETWEEN 条件*/
from employee
where job_lvl BETWEEN 100 AND 200

select lname,job_id,job_lvl/*使用 IN 成员条件测试在列表中的值*/
from employee
where job_lvl IN (100,170,200)

select emp_id,lname,job_id,job_lvl/*使用 NOT 操作*/
from employee
where job_lvl NOT IN(100,170,200)

select title_id,title,type,price
from titles
where price IS NULL

select lname,job_lvl,job_lvl/*优先规则 AND 在前*/
from employee
where job_lvl=5
```

```
OR job_lvl=7
AND job_lvl>160

select lname,job_lvl,job_lvl/*用圆括号强制优先权*/
from employee
where (job_lvl=5
OR job_lvl=7)
AND job_lvl>160

select lname,job_id,job_lvl,hire_date/*用 ORDER BY 子句进行排序(升序排列)*/
from employee
ORDER BY hire_date

select lname,job_id,job_lvl,hire_date/*用 DESC 降序排列*/
from employee
ORDER BY hire_date DESC

select emp_id,lname,job_lvl*2 "Double job_lvl"/*用列别名排序*/
from employee
ORDER BY "Double job_lvl"

select lname,job_id,job_lvl/*多列排序*/
from employee
ORDER BY job_id,job_lvl DESC
```

掌握用函数对数据的计算

知识点：SQL 函数主要分为两中类型 1.单行函数(也叫标量函数) 2.多行函数

标量函数：对单一值操作，返回单一值。只要表达式有效即可使用标量函数。

我们把标量函数的分类列出来：

配置函数	返回当前配置信息
游标函数	返回游标信息
日期和时间函数	对日期和时间输入值进行操作，返回一个字符串、数字或日期和时间值
数学函数	对作为函数参数提供的输入值进行计算，返回一个数字值
元数据函数	返回有关数据库和数据库对象的信息
安全函数	返回有关用户和角色的信息
字符串函数	对字符串(char 或 varchar)输入值执行操作,返回一个字符串或数字值

系统函数　　执行操作并返回有关 Microsoft SQL Server 中的值、对象和设置的信息

系统统计函数 返回系统的统计信息

文本和图象函数 对文本或图象进行操作并返回有关信息

聚合函数是对一组值进行操作，跟标量函数不同.

行集函数是可以向 SQL 语句中表引用一样使用，具体看文档了....

下面具体看例子了

```
select emp_id,UPPER(lname),job_id
from employee
where lname='chang'
/*大小写处理函数*/
select emp_id,LOWER(lname),job_id
from employee
where lname='chang'

select emp_id,fname+lname NAME,  /*使用字符处理函数*/
job_id,LEN(lname) Length,
CHARINDEX('a',lname)"Contains 'a'?"
from employee
where SUBSTRING(emp_id,1,2)='MA'
/*
运算符+ :连接值在一起
substring :选取给定位置和长度的子字符串
len  :以数字值显示一个字符串的长度
charIndex  :找到一个给定字符的数字位置
Replicate  :用给定的字符替换给定长度的字符串
Ltrim 和 Rtrim  :从一个字符串中去除头或尾的字符
*/

select ROUND(2008.092023,2),/*数字函数 ROUND 四舍五入*/
ROUND(2008.092023,0),
ROUND(2008.092023,-1)
/*注意奇数偶数的进位不同它也可以用语日期函数处理*/

select lname,job_lvl,job_lvl%50/*使用%(模)函数经常用与确定一个值是奇数还是偶数*/
from employee
where job_lvl=5

select lname,hire_date/*日期的使用*/
from employee
where lname like 'A%'
```

```
select lname,hire_date+365 AS "One Year"/*用日期做算术运算该例显示所有 job_id=10 的雇员的名字和
其受雇满一年的日期*/
from employee
where job_id=10

select lname,  /*CONVERT 函数用于日期转换*/
CONVERT(varchar(10),hire_date,20)
AS HIREDATE
from employee

select SUBSTRING(title,1,30) AS Title,ytd_sales/*对数字使用 CAST 函数*/
from titles
where CAST(ytd_sales AS char(20)) LIKE '3%'
/*
检索书名：这些图书的截止当前销售额的第一数字为，并将这些图书的 ytd_sales 转换为 char(20)
CAST(number,数据类型)
*/

select CONVERT(varchar(10),price,0) PRICE/*对数字使用 CONVERT 函数*/
from titles
where title_id='BU1111'

select CAST('2008-09-20 23:57' AS datetime)
/*使用 CAST 或 CONVERT 函数转换字符串到日期*/
select CONVERT(datetime,'2008-09-20 23:59')

select title,ytd_sales/*使用带有 LIKE 子句的 CAST*/
from titles
where
CAST(ytd_sales AS char(20)) LIKE '15%'
AND type='trad_cook'

select title,ISNULL(price,0) Price,/*使用 ISNULL 函数*/
ISNULL(price*12,0) 'Ten Price'
from titles
/*为了计算所有图书的价格，在进行算术运算之前必须转换空值为一个数，ISNULL 函数用来转换空值为零*/


select                  /*使用 CASE 表达式*/
CASE
when price IS NULL THEN 'Not yet priced'
when price <10 THEN 'Very Reasonable Title'
when price>=10 and price <20 THEN 'Coffee Table Title'
else 'Expensive book!'
```

```
END AS 'Price Category',  /*这个逗号不敢忘*/
CONVERT(varchar(20),title) AS "shortтened Title"
from pubs.dbo.titles
ORDER BY price
/*
使用 CASE 表达式使得 if-then-else 条件判断很容易实现
*/



select        /*使用 CASE 处理条件数据*/
CASE type
when 'popular_comp' then 'Popular Computing'
when 'mod_cook' then 'Modern Coking'
      when 'business' then 'Business'
when 'psychology' then 'Psychology'
when 'trad_cook' then 'Traditional Cooking'
else 'Not yet categorized'
END AS Category,
CONVERT(varchar(30),titles) AS "Shortened Title",
Price AS price
from titles
where price IS NOT NULL
ORDER BY 1
```

日期分表查询(version 1.0)

说明：以日期或月份分表进行多日、多月查询连表查询的有关写法

```
Create Procedure  Search
@date1 datetime, --起始时间
@date2 datetime  --终止时间
as
declare @date1New datetime,
 @rq varchar(6),
 @i int , --记录循环次数
 @s varchar(5000) --根据表多少可以扩大它
set @rq=convert(varchar,@date1,12) --把时间格式化比如-06-20 变成了


set @s='select * from sensor'+@rq --初始化@s='select * from sensor080620' 这样第一个表就有
了

set @i=datediff(d,@date1,@date2) --日期相减
```

```
while @i>0
begin
 --如果是非常重要的系统可以在这里加上该物理表是否存在的语句
--if  exists(select  *  from  dbo.sysobjects  where
--id  =  object_id(N'c')  and  OBJECTPROPERTY(id,N'IsUserTable')=1)
--begin
--SELECT '存在'
--end

 set @date1New= dateadd(day,@i,@date1)
 set @rq=convert(varchar,@date1New,12)
 --如果是按月进行查询就修改上面
 set @s= @s+' union all select * from sensor'+@rq
 set @i=@i-1 --累加
end

select @s
--exec(@S)
go
exec Search '2008-06-20 00:11:11.000','2008-07-25 00:11:11.000'
  drop Procedure  Search
```

查询节点的函数

```
create table tb(id varchar(3) , pid varchar(3) , name varchar(10))
insert into tb values('001' , null  , '广东省')
insert into tb values('002' , '001' , '广州市')
insert into tb values('003' , '001' , '深圳市')
insert into tb values('004' , '002' , '天河区')
insert into tb values('005' , '003' , '罗湖区')
insert into tb values('006' , '003' , '福田区')
insert into tb values('007' , '003' , '宝安区')
insert into tb values('008' , '007' , '西乡镇')
insert into tb values('009' , '007' , '龙华镇')
insert into tb values('010' , '007' , '松岗镇')
go
```

查询指定节点及其所有子节点的函数

```
create function f_cid(@ID varchar(3)) returns @t_level table(id varchar(3) , level int)
as
begin
  declare @level int
  set @level = 1
  insert into @t_level select @id , @level
  while @@ROWCOUNT > 0
  begin
    set @level = @level + 1
    insert into @t_level select a.id , @level
    from tb a , @t_Level b
    where a.pid = b.id and b.level = @level - 1
  end
  return
end
go
```
调用函数查询(广东省)及其所有子节点

```
select a.* from tb a , f_cid('001') b where a.id = b.id order by a.id
```
```
id   pid   name
---- ---- ----------
001   NULL 广东省
002   001  广州市
003   001  深圳市
004   002  天河区
005   003  罗湖区
006   003  福田区
007   003  宝安区
008   007  西乡镇
009   007  龙华镇
010   007  松岗镇
```
(所影响的行数为 10 行)

调用函数查询(广州市)及其所有子节点

```
select a.* from tb a , f_cid('002') b where a.id = b.id order by a.id
```
```
id   pid  name
---- ---- ----------
002   001  广州市
004   002  天河区
```
(所影响的行数为 2 行)

调用函数查询(深圳市)及其所有子节点

```
select a.* from tb a , f_cid('003') b where a.id = b.id order by a.id
```
```
id   pid  name
---- ---- ----------
003   001  深圳市
005   003  罗湖区
006   003  福田区
007   003  宝安区
008   007  西乡镇
009   007  龙华镇
```

```
010   007  松岗镇
（所影响的行数为7 行）
```

```
drop table tb
drop function f_cid
```

论坛会员积分判断升级

```
set ANSI_NULLS ON
set QUOTED_IDENTIFIER ON
go


ALTER trigger [tri_updatesalary_Mem_Type]
 on [dbo].[SalaryIncrease]
after  insert
as
 declare @i int
 set @i=@@identity

 update m
 set m.Mem_Type=case when s.SMark>500000 then '退休会员'--500000
               when s.SMark>400000 then '钻石五级'--400000
               when s.SMark>300000 then '钻石四级'--300000
               when s.SMark>200000 then '钻石三级'--200000
               when s.SMark>100000 then '钻石二级'--100000
               when s.SMark>50000 then '钻石一级'--50000
               when s.SMark>40000 then '五星级'--40000
               when s.SMark>30000 then '四星级'--30000
               when s.SMark>20000 then '三星级'--20000
               when s.SMark>10000 then '二星级'--10000
               when s.SMark>5000 then '一星级'--5000
               else '一般 VIP 会员'
            end
from MemberInfo m
  join (select s.SCardName,sum(s.SMark) as SMark
        from  SalaryIncrease  s  join  inserted  i  on  s.SCardName=i.SCardName  group  by
s.SCardName) s
    on m.Mem_Num=s.SCardName

 --if  exists(select  *  from  MemberInfo  m  join  inserted  i  on  m.Mem_Num=i.SCardName  and
m.Mem_Mark>100 )
 if exists (select  m.SCardName,sum(m.ShopMark)as mark from(select m.SCardName,m.ShopMark
from ShoppingMark m join inserted i on m.SCardName=i.SCardName
where year(m.SDate)=year(getdate()) and month(m.SDate)=month(getdate())) m
 group by m.SCardName  having sum(m.ShopMark)>100)
```

最新的

```
select * from ShoppingMark SM join (select m.SCardName,sum(m.ShopMark) as Mark from ShoppingMark m join inserted i on m.SCardName=i.SCardName group by m.SCardName) s

 on SM.SCardName=s.ScardName where month(SDate)=month(getdate()) and year(SDate)=year(getdate()) and s.Mark>100
```

```
begin
 update s
   set s.SIncease=case
                  --积分大于就是星级.所以不用判断是否是星级
                when m.Mem_Type<>'一般 VIP 会员' then s.SMark*0.3
                else case
                      when s.SMark>4000 then
                          s.SMark*0.3
                    --  when s.SMark>4000 then
                        --  200*0.2+200*0.23+200*0.25+200*0.28+(s.SMark-800)*0.3
                      when s.SMark>3000 then
                          1000*0.2+1000*0.23+1000*0.25+(s.SMark-600)*0.28
                      when s.SMark>2000 then
                          1000*0.2+100*0.23+(s.SMark-400)*0.25
                      when s.SMark>1000 then
                          (s.SMark-200)*.023+1000*0.2
                      else  s.SMark*0.2
                   end
               end
 from SalaryIncrease as s
 join inserted i
   on s.SCardName=i.SCardName
  join MemberInfo m
   on (i.SCardName=m.Mem_Num and s.SID=@i) or (i.SCardName=m.Mem_Num and s.SIncease=0)
   end
--go
```

bom 结构，查找节点下所有子节点：

```
create table os(id int,parentid int,desn varchar(10))
insert into os select 1,0,'体育用品'
insert into os select 2,0,'户外运动'
insert into os select 3,1,'篮球'
insert into os select 4,1,'足球'
insert into os select 5,2,'帐篷'
insert into os select 6,2,'登山鞋'
insert into os select 7,0,'男士用品'
insert into os select 8,7,'刮胡刀'
insert into os select 9,3,'大号篮球'
```
求个节点下所有子节点：

```
create function f_cid(@id int)
returns varchar(500)
as
begin
    declare @t table(id int,parentid int,desn varchar(10),lev int)
    declare @lev int
    set @lev=1
    insert into @t select *,@lev from  os where id=@id
    while(@@rowcount>0)
    begin
        set @lev=@lev+1
        insert into @t select a.*,@lev from os a,@t b
        where a.parentid=b.id and b.lev=@lev-1
    end
    declare @cids varchar(500)
    select @cids=isnull(@cids+',','')+ltrim(id) from @t order by lev
    return @cids
end
go
```
调用函数

```
select *,ids=dbo.f_cid(id) from os
```
得到每个节点路径：

```
create proc wsp2
@id int
as
select *,cast(' ' as varchar(10)) fullpath  into #os from os
DECLARE @i int,@j int
set @i=0
set @j=1
select @i=max(parentid) from #os
update #os set fullpath=id
while @j<=@i
begin
      update #os set fullpath=a.fullpath+','+ltrim(#os.id)
          from #os inner join #os a on #os.parentid=a.id
      where #os.parentid=@j
      set @j=@j+1
end
select * from #os
go
```

调用存储过程

```
exec wsp2 1
```

库存先进先出简单例子：

```
create table t(
id int identity(1,1),
name varchar(50),--商品名称
j int,        --入库数量
c int,        --出库数量
jdate datetime --入库时间
)
insert into t(name,j,c,jdate) select  'A',100,0,'2007-12-01'
insert into t(name,j,c,jdate) select  'A',200,0,'2008-01-07'
insert into t(name,j,c,jdate) select  'B',320,0,'2007-12-21'
insert into t(name,j,c,jdate) select  'A',100,0,'2008-01-15'
insert into t(name,j,c,jdate) select  'B',90,0,'2008-02-03'
insert into t(name,j,c,jdate) select  'A',460,0,'2008-02-01'
insert into t(name,j,c,jdate) select  'A',510,0,'2008-03-01'
go




create proc wsp
@name varchar(50),--商品名称
@cost int         --销售量
as
--先得出该货物的库存是否够
declare @spare float --剩余库存
select @spare=sum(j)-sum(c) from t where name=@name
if(@spare>=@cost)
begin
    --根据入库日期采用先进先出原则对货物的库存进行处理
    update t set c=
    case when (select @cost-isnull(sum(j),0)+isnull(sum(c),0) from t where name=@name and
jdate<=a.jdate and j!=c)>=0
    then a.j
    else
        case when (select @cost-isnull(sum(j),0)+isnull(sum(c),0) from t where name=@name
and jdate<a.jdate and j!=c)<0 then 0
        else (select @cost-isnull(sum(j),0)+isnull(sum(c),0)+a.c from t where name=@name and
jdate<a.jdate and j!=c)
        end
    end
```

```
   from t a where name=@name and j!=c
end
else
   raiserror('库存不足',16,1)
   return
go
```
测试：

```
exec wsp @name='A',@cost=180
select * from t

--drop table t
--drop proc wsp
```

返程问题,找出雇员从本地出发后直接返回的情况

```
create  table  trav(name  nvarchar(10),date  datetime,comefrom  nvarchar(10),destin
nvarchar(10),id int)
insert into trav select '张三','2007-01-01','上海','广州',1
insert into trav select '李四','2007-01-01','上海','广州',2
insert into trav select '李四','2007-02-01','上海','成都',3
insert into trav select '张三','2007-01-15','广州','上海',4
insert into trav select '张三','2007-02-06','上海','广州',5
insert into trav select '张三','2007-02-18','广州','上海',6
go
select a.name,a.date,a.comefrom,a.destin,b.date,b.comefrom,b.destin
 from  trav  a  inner  join  trav  b  on  a.name=b.name  and  a.comefrom=b.destin  and
a.destin=b.comefrom where a.id<b.id
 and not exists(select 1 from trav where comefrom=b.comefrom and date<b.date and date>a.date)
go
drop table trav
name     date       comefrom       destin                 date        comefrom      destin
---- -------- --------- ----- ---------- ------------------------ ---------- ----------
张三  2007-01-01 00:00:00.000 上海      广州      2007-01-15 00:00:00.000 广州      上海
张三  2007-02-06 00:00:00.000 上海      广州      2007-02-18 00:00:00.000 广州      上海
 (2 行受影响)
```

分解字符串并查询相关数据

说明：通过使用函数等方法分解字符串查询相关数据。

问题：通过分解一个带某种符号分隔的字符串在数据库中查找相关数据。

例如:
@str = '1,2,3',查询下表得到记录,4,5,6

```
ID   TypeID
1    1,2,3,4,5,6,7,8,9,10,11,12
2    2,3
3    3,7,8,9
4    2,6
5    4,5
6    6,7
-----------------------------
```

```sql
create table tb (ID int , TypeID varchar(30))
insert into tb values(1 , '1,2,3,4,5,6,7,8,9,10,11,12')
insert into tb values(2 , '2,3')
insert into tb values(3 , '3,7,8,9')
insert into tb values(4 , '2,6')
insert into tb values(5 , '4,5')
insert into tb values(6 , '6,7')
go
```

如果仅仅是一个,如@str = '1'.

```sql
declare @str as varchar(30)
set @str = '1'
select * from tb where charindex(',' + @str + ',' , ',' + TypeID + ',') > 0
select * from tb where ',' + TypeID + ',' like '%,' + @str + ',%'
```

```
ID          TypeID
----------- ------------------------------
1           1,2,3,4,5,6,7,8,9,10,11,12
```

(所影响的行数为 1 行)

如果包含两个,如@str = '1,2'.

```sql
declare @str as varchar(30)
set @str = '1,2'
select * from tb where charindex(',' + left(@str , charindex(',' , @str) - 1) + ',' , ','
+ typeid + ',') > 0 or
  charindex(',' + substring(@str , charindex(',' , @str) + 1 , len(@str)) + ',' , ',' + typeid
+ ',') > 0
select * from tb where ',' + typeid + ',' like '%,' + left(@str , charindex(',' , @str)
- 1) + ',%' or
  ',' + typeid + ',' like '%,' + substring(@str , charindex(',' , @str) + 1 , len(@str))
+ ',%'
```

```
ID          TypeID
----------- ------------------------------
1           1,2,3,4,5,6,7,8,9,10,11,12
2           2,3
4           2,6
```

（所影响的行数为 3 行）

如果包含三个或四个，用 PARSENAME 函数来处理.

```
declare @str as varchar(30)
set @str = '1,2,3,4'
select * from tb where
  charindex(',' + parsename(replace(@str , ',' , '.') , 4) + ',' , ',' + typeid + ',') >
0 or
  charindex(',' + parsename(replace(@str , ',' , '.') , 3) + ',' , ',' + typeid + ',') >
0 or
  charindex(',' + parsename(replace(@str , ',' , '.') , 2) + ',' , ',' + typeid + ',') >
0 or
  charindex(',' + parsename(replace(@str , ',' , '.') , 1) + ',' , ',' + typeid + ',') >
0
select * from tb where
  ',' + typeid + ',' like '%,' + parsename(replace(@str , ',' , '.') , 4) + ',%' or
  ',' + typeid + ',' like '%,' + parsename(replace(@str , ',' , '.') , 3) + ',%' or
  ',' + typeid + ',' like '%,' + parsename(replace(@str , ',' , '.') , 2) + ',%' or
  ',' + typeid + ',' like '%,' + parsename(replace(@str , ',' , '.') , 1) + ',%'
```

```
ID          TypeID
----------- ------------------------------
1           1,2,3,4,5,6,7,8,9,10,11,12
2           2,3
3           3,7,8,9
4           2,6
5           4,5
（所影响的行数为 5 行）
```

如果超过四个，则只能使用函数或动态 SQL 来分解并查询数据。

名称：fn_split 函数.

功能：实现字符串分隔功能的函数

```
create function dbo.fn_split(@inputstr varchar(8000), @seprator varchar(10))
returns @temp table (a varchar(200))
as
begin
  declare @i int
  set @inputstr = rtrim(ltrim(@inputstr))
  set @i = charindex(@seprator , @inputstr)
  while @i >= 1
  begin
    insert @temp values(left(@inputstr , @i - 1))
    set @inputstr = substring(@inputstr , @i + 1 , len(@inputstr) - @i)
    set @i = charindex(@seprator , @inputstr)
  end
  if @inputstr <> '\'
  insert @temp values(@inputstr)
  return
end
go

--调用
declare @str as varchar(30)
set @str = '1,2,3,4,5'

select distinct m.* from tb m,
(select * from dbo.fn_split(@str,',')) n
where charindex(',' + n.a + ',' , ',' + m.typeid + ',') > 0

drop table tb
drop function dbo.fn_split
```

```
ID          TypeID
----------- ------------------------------
1           1,2,3,4,5,6,7,8,9,10,11,12
2           2,3
3           3,7,8,9
4           2,6
5           4,5
（所影响的行数为 5 行）
```

使用动态 SQL 的语句。

```
declare @str varchar(200)
declare @sql as varchar(1000)
set @str = '1,2,3,4,5'
set @sql = 'select ''' + replace(@str , ',' , ''' as id union all select ''')
set @sql = @sql + ''''
set @sql = 'select distinct a.* from tb a , (' + @sql + ') b where charindex(' + ''',''
+ b.id + ' + ''',''' + ' , ' + ''',''' + a.typeid + ' + ''',''' + ') > 0 '
exec (@sql)
```

```
ID          TypeID
----------- ------------------------------
1           1,2,3,4,5,6,7,8,9,10,11,12
2           2,3
3           3,7,8,9
4           2,6
5           4,5
```

（所影响的行数为 5 行）

分拆列值

有表 tb, 如下:

```
id           value
----------- -----------
1           aa,bb
2           aaa,bbb,ccc
```

欲按 id,分拆 value 列, 分拆后结果如下:

```
id           value

----------- --------
1           aa
1           bb
2           aaa
2           bbb
2           ccc
```

1. 旧的解决方法(sql server 2000)

```
SELECT TOP 8000 id = IDENTITY(int, 1, 1) INTO # FROM syscolumns a, syscolumns b

SELECT A.id, SUBSTRING(A.[values], B.id, CHARINDEX(',', A.[values] + ',', B.id) - B.id)
FROM tb A, # B
WHERE SUBSTRING(',' + A.[values], B.id, 1) = ','

DROP TABLE #
```

2. 新的解决方法(sql server 2005)

```
create table tb(id int,value varchar(30))
insert into tb values(1,'aa,bb')
insert into tb values(2,'aaa,bbb,ccc')
go
SELECT A.id, B.value
FROM(
   SELECT id, [value] = CONVERT(xml,' <root> <v>' + REPLACE([value], ',', ' </v> <v>') +
' </v> </root>') FROM tb
)A
OUTER APPLY(
   SELECT value = N.v.value('.', 'varchar(100)') FROM A.[value].nodes('/root/v') N(v)
)B

DROP TABLE tb
```

```
id              value
----------- ------------------------------
1               aa
1               bb
2               aaa
2               bbb
2               ccc
 (5 行受影响)
```

合并列值

表结构，数据如下：

```
id     value
----- ------
1       aa
1       bb
2       aaa
2       bbb
2       ccc
```

需要得到结果：

```
id       values
------ -----------
1        aa,bb
2        aaa,bbb,ccc
```

即：group by id, 求value 的和（字符串相加）

1. 旧的解决方法(在sql server 2000中只能用函数解决。)

1. 创建处理函数

```
create table tb(id int, value varchar(10))
insert into tb values(1, 'aa')
insert into tb values(1, 'bb')
insert into tb values(2, 'aaa')
insert into tb values(2, 'bbb')
insert into tb values(2, 'ccc')
go

CREATE FUNCTION dbo.f_str(@id int)
RETURNS varchar(8000)
AS
BEGIN
    DECLARE @r varchar(8000)
    SET @r = ''
    SELECT @r = @r + ',' + value FROM tb WHERE id=@id
```

```
   RETURN STUFF(@r, 1, 1, '')
END
GO

-- 调用函数
SELECt id, value = dbo.f_str(id) FROM tb GROUP BY id

drop table tb
drop function dbo.f_str
```

```
id             value
----------- -----------
1             aa,bb
2             aaa,bbb,ccc
（所影响的行数为 2 行）
```

2、另外一种函数.

```
create table tb(id int, value varchar(10))
insert into tb values(1, 'aa')
insert into tb values(1, 'bb')
insert into tb values(2, 'aaa')
insert into tb values(2, 'bbb')
insert into tb values(2, 'ccc')
go

--创建一个合并的函数
create function f_hb(@id int)
returns varchar(8000)
as
begin
  declare @str varchar(8000)
  set @str = ''
  select @str = @str + ',' + cast(value as varchar) from tb where id = @id
  set @str = right(@str , len(@str) - 1)
  return(@str)
End
go

--调用自定义函数得到结果:
select distinct id ,dbo.f_hb(id) as value from tb

drop table tb
drop function dbo.f_hb
```

```
id             value

----------- -----------
```

```
1            aa,bb
2            aaa,bbb,ccc
```

（所影响的行数为 2 行）

2. 新的解决方法(在 sql server 2005 中用 OUTER APPLY 等解决。)

```
create table tb(id int, value varchar(10))
insert into tb values(1, 'aa')
insert into tb values(1, 'bb')
insert into tb values(2, 'aaa')
insert into tb values(2, 'bbb')
insert into tb values(2, 'ccc')
go
-- 查询处理
SELECT * FROM(SELECT DISTINCT id FROM tb)A OUTER APPLY(
       SELECT [values]= STUFF(REPLACE(REPLACE(
           (
              SELECT value FROM tb N
              WHERE id = A.id
              FOR XML AUTO
           ), ' <N value="', ','), '"/>', ''), 1, 1, '')
)N
drop table tb
```

```
id             values
----------- -----------
1            aa,bb
2            aaa,bbb,ccc
```

(2 行受影响)

SQL2005中的方法

```
create table tb(id int, value varchar(10))
insert into tb values(1, 'aa')
insert into tb values(1, 'bb')
insert into tb values(2, 'aaa')
insert into tb values(2, 'bbb')
insert into tb values(2, 'ccc')
go

select id, [values]=stuff((select ','+[value] from tb t where id=tb.id for xml path('')),
1, 1, '')
from tb
group by id
drop table tb
```

```
id              values
----------- --------------------
```

```
1             aa,bb
2             aaa,bbb,ccc
(2 行受影响)
```

按某一字段分组取最大(小)值所在行的数据

数据如下：

```
name val memo
a     2   a2(a的第二个值)
a     1   a1--a的第一个值
a     3   a3:a的第三个值
b     1   b1--b的第一个值
b     3   b3:b的第三个值
b     2   b2b2b2b2
b     4   b4b4
b     5   b5b5b5b5b5
```

```
--创建表并插入数据：
create table tb(name varchar(10),val int,memo varchar(20))
insert into tb values('a',    2,   'a2(a 的第二个值)')
insert into tb values('a',    1,   'a1--a 的第一个值')
insert into tb values('a',    3,   'a3:a 的第三个值')
insert into tb values('b',    1,   'b1--b 的第一个值')
insert into tb values('b',    3,   'b3:b 的第三个值')
insert into tb values('b',    2,   'b2b2b2b2')
insert into tb values('b',    4,   'b4b4')
insert into tb values('b',    5,   'b5b5b5b5b5')
go
```

一、按 name 分组取 val 最大的值所在行的数据。

方法：

```
select a.* from tb a where val = (select max(val) from tb where name = a.name) order by
a.name
```

方法：

```
select a.* from tb a where not exists(select 1 from tb where name = a.name and val > a.val)
```

方法：

```
select a.* from tb a,(select name,max(val) val from tb group by name) b where a.name = b.name
and a.val = b.val order by a.name
```

方法：

```
select a.* from tb a inner join (select name , max(val) val from tb group by name) b on
a.name = b.name and a.val = b.val order by a.name
```
方法

```
select a.* from tb a where 1 > (select count(*) from tb where name = a.name and val > a.val )
order by a.name
```

```
name            val          memo
---------- ----------- --------------------
a                 3           a3:a的第三个值
b                 5           b5b5b5b5b5
```
二、按 name 分组取 val 最小的值所在行的数据。

方法:

```
select a.* from tb a where val = (select min(val) from tb where name = a.name) order by
a.name
```
方法:

```
select a.* from tb a where not exists(select 1 from tb where name = a.name and val < a.val)
```
方法:

```
select a.* from tb a,(select name,min(val) val from tb group by name) b where a.name = b.name
and a.val = b.val order by a.name
```
方法:

```
select a.* from tb a inner join (select name , min(val) val from tb group by name) b on
a.name = b.name and a.val = b.val order by a.name
```
方法

```
select a.* from tb a where 1 > (select count(*) from tb where name = a.name and val < a.val)
order by a.name
```
```
name         val           memo
---------- ----------- --------------------
a             1             a1--a的第一个值
b             1             b1--b的第一个值
```
三、按 name 分组取第一次出现的行所在的数据。

```
select a.* from tb a where val = (select top 1 val from tb where name = a.name) order by
a.name
```
```
name          val         memo
---------- ----------- --------------------
a              2          a2(a的第二个值)
b              1          b1--b的第一个值
```
四、按 name 分组随机取一条数据。

```
select a.* from tb a where val = (select top 1 val from tb where name = a.name order by
newid()) order by a.name
```

```
name          val        memo
---------- ----------- --------------------
a              1          a1--a的第一个值
b              5          b5b5b5b5b5
```

五、按 name 分组取最小的两个(N 个)val

```
select a.* from tb a where 2 > (select count(*) from tb where name = a.name and val < a.val )
order by a.name,a.val
select a.* from tb a where val in (select top 2 val from tb where name=a.name order by val)
order by a.name,a.val
select a.* from tb a where exists (select count(*) from tb where name = a.name and val <
a.val having Count(*) < 2) order by a.name
```

```
name          val        memo
---------- ----------- --------------------
a             1          a1--a的第一个值
a             2          a2(a的第二个值)
b             1          b1--b的第一个值
b             2          b2b2b2b2
```

六、按 name 分组取最大的两个(N 个)val

```
select a.* from tb a where 2 > (select count(*) from tb where name = a.name and val > a.val )
order by a.name,a.val
select a.* from tb a where val in (select top 2 val from tb where name=a.name order by val
desc) order by a.name,a.val
select a.* from tb a where exists (select count(*) from tb where name = a.name and val >
a.val having Count(*) < 2) order by a.name
```

```
name          val        memo
---------- ----------- --------------------
a             2          a2(a的第二个值)
a             3          a3:a的第三个值
b             4          b4b4
b             5          b5b5b5b5b5
```

七，如果整行数据有重复，所有的列都相同。

数据如下：

```
Name    val  memo
a       2   a2(a的第二个值)
a       1   a1--a的第一个值
a       1   a1--a的第一个值
a       3   a3:a的第三个值
a       3   a3:a的第三个值
b       1   b1--b的第一个值
b       3   b3:b的第三个值
b       2   b2b2b2b2
```

```
b       4   b4b4
b       5   b5b5b5b5b5
```

在 sql server 2000 中只能用一个临时表来解决，生成一个自增列，先对 val 取最大或最小，然后再通过自增列来取数据。

创建表并插入数据：

```
create table tb(name varchar(10),val int,memo varchar(20))
insert into tb values('a',    2,   'a2(a 的第二个值)')
insert into tb values('a',    1,   'a1--a 的第一个值')
insert into tb values('a',    1,   'a1--a 的第一个值')
insert into tb values('a',    3,   'a3:a 的第三个值')
insert into tb values('a',    3,   'a3:a 的第三个值')
insert into tb values('b',    1,   'b1--b 的第一个值')
insert into tb values('b',    3,   'b3:b 的第三个值')
insert into tb values('b',    2,   'b2b2b2b2')
insert into tb values('b',    4,   'b4b4')
insert into tb values('b',    5,   'b5b5b5b5b5')
go
select * , px = identity(int,1,1) into tmp from tb
select m.name,m.val,m.memo from
(
  select t.* from tmp t where val = (select min(val) from tmp where name = t.name)
) m where px = (select min(px) from
(
  select t.* from tmp t where val = (select min(val) from tmp where name = t.name)
) n where n.name = m.name)
drop table tb,tmp
```

```
name          val         memo
---------- ----------- --------------------
a             1          a1--a的第一个值
b             1          b1--b的第一个值

(2 行受影响)
```

在 sql server 2005 中可以使用 row_number 函数，不需要使用临时表。

创建表并插入数据：

```
create table tb(name varchar(10),val int,memo varchar(20))
insert into tb values('a',    2,   'a2(a 的第二个值)')
insert into tb values('a',    1,   'a1--a 的第一个值')
insert into tb values('a',    1,   'a1--a 的第一个值')
insert into tb values('a',    3,   'a3:a 的第三个值')
insert into tb values('a',    3,   'a3:a 的第三个值')
insert into tb values('b',    1,   'b1--b 的第一个值')
insert into tb values('b',    3,   'b3:b 的第三个值')
insert into tb values('b',    2,   'b2b2b2b2')
insert into tb values('b',    4,   'b4b4')
insert into tb values('b',    5,   'b5b5b5b5b5')
go
select m.name,m.val,m.memo from
(
  select * , px = row_number() over(order by name , val) from tb
) m where px = (select min(px) from
(
  select * , px = row_number() over(order by name , val) from tb
) n where n.name = m.name)
drop table tb
```

```
name          val        memo
---------- ----------- --------------------
a             1          a1--a的第一个值
b             1          b1--b的第一个值
 (2 行受影响)
```

普通行列转换(version 1.0)仅针对 sql server 2000 提供静态和动态写法，version 2.0 增加 sql server 2005 的有关写法。

问题：假设有张学生成绩表(tb)如下:

```
姓名    课程   分数
张三    语文   74
张三    数学   83
张三    物理   93
李四    语文   74
李四    数学   84
李四    物理   94
```

想变成(得到如下结果):

```
姓名    语文   数学   物理
---- ---- ---- ----
李四    74    84    94
张三    74    83    93
-------------------
```

```
create table tb(姓名 varchar(10) , 课程 varchar(10) , 分数 int)
insert into tb values('张三' , '语文' , 74)
insert into tb values('张三' , '数学' , 83)
insert into tb values('张三' , '物理' , 93)
insert into tb values('李四' , '语文' , 74)
insert into tb values('李四' , '数学' , 84)
insert into tb values('李四' , '物理' , 94)
go
```

```
--SQL SERVER 2000 静态 SQL,指课程只有语文、数学、物理这三门课程。(以下同)
select 姓名 as 姓名,
  max(case 课程 when '语文' then 分数 else 0 end) 语文,
  max(case 课程 when '数学' then 分数 else 0 end) 数学,
  max(case 课程 when '物理' then 分数 else 0 end) 物理
from tb
group by 姓名
--SQL SERVER 2000 动态 SQL,指课程不止语文、数学、物理这三门课程。(以下同)
declare @sql varchar(8000)
set @sql = 'select 姓名'
select @sql = @sql + ' , max(case 课程 when ''' + 课程+ ''' then 分数 else 0 end) [' + 课程+
']'
from (select distinct 课程 from tb) as a
set @sql = @sql + ' from tb group by 姓名'
exec(@sql)
--SQL SERVER 2005 静态 SQL。
select * from (select * from tb) a pivot (max(分数) for 课程 in (语文,数学,物理)) b
--SQL SERVER 2005 动态 SQL。
declare @sql varchar(8000)
select @sql = isnull(@sql + '],[' , '') + 课程 from tb group by 课程
set @sql = '[' + @sql + ']'
exec ('select * from (select * from tb) a pivot (max(分数) for 课程 in (' + @sql + ')) b')
```

```
问题：在上述结果的基础上加平均分，总分，得到如下结果：
姓名   语文  数学  物理  平均分  总分
---- ---- ---- ---- ------ ----
李四   74   84   94   84.00  252
张三   74   83   93   83.33  250
--------------------------------
--SQL SERVER 2000 静态SQL。
```

```
select 姓名姓名,
  max(case 课程 when '语文' then 分数 else 0 end) 语文,
  max(case 课程 when '数学' then 分数 else 0 end) 数学,
  max(case 课程 when '物理' then 分数 else 0 end) 物理,
  cast(avg(分数*1.0) as decimal(18,2)) 平均分,
  sum(分数) 总分
from tb
group by 姓名
--SQL SERVER 2000 动态 SQL。
declare @sql varchar(8000)
set @sql = 'select 姓名'
select @sql = @sql + ' , max(case 课程 when ''' + 课程+ ''' then 分数 else 0 end) [' + 课程+
']'
from (select distinct 课程 from tb) as a
```

```
set @sql = @sql + ' , cast(avg(分数*1.0) as decimal(18,2)) 平均分, sum(分数) 总分 from tb group
by 姓名'
exec(@sql)
--SQL SERVER 2005 静态 SQL。
select m.* , n.平均分, n.总分 from
(select * from (select * from tb) a pivot (max(分数) for 课程 in (语文,数学,物理)) b) m,
(select 姓名, cast(avg(分数*1.0) as decimal(18,2)) 平均分, sum(分数) 总分 from tb group by 姓
名) n
where m.姓名= n.姓名
--SQL SERVER 2005 动态 SQL。
declare @sql varchar(8000)
select @sql = isnull(@sql + ',' , '') + 课程 from tb group by 课程
exec ('select m.* , n.平均分, n.总分 from
(select * from (select * from tb) a pivot (max(分数) for 课程 in (' + @sql + ')) b) m ,
(select 姓名, cast(avg(分数*1.0) as decimal(18,2)) 平均分, sum(分数) 总分 from tb group by 姓
名) n
where m.姓名= n.姓名')
drop table tb
```

问题：如果上述两表互相换一下：即表结构和数据为：

```
姓名  语文  数学  物理
张三  74    83    93
李四  74    84    94
想变成(得到如下结果)：
姓名  课程    分数
---- ---- ----
李四  语文    74
李四  数学    84
李四  物理    94
张三  语文    74
张三  数学    83
张三  物理    93
--------------
```

```
create table tb(姓名 varchar(10) , 语文 int , 数学 int , 物理 int)
insert into tb values('张三',74,83,93)
insert into tb values('李四',74,84,94)
go
--SQL SERVER 2000 静态 SQL。
select * from
(
 select 姓名, 课程= '语文' , 分数= 语文 from tb
 union all
 select 姓名, 课程= '数学' , 分数= 数学 from tb
 union all
 select 姓名, 课程= '物理' , 分数= 物理 from tb
) t
order by 姓名, case 课程 when '语文' then 1 when '数学' then 2 when '物理' then 3 end
--SQL SERVER 2000 动态 SQL。
--调用系统表动态生态。
declare @sql varchar(8000)
select @sql = isnull(@sql + ' union all ' , '' ) + ' select 姓名, [课程] = ' + quotename(Name ,
'''') + ' , [分数] = ' + quotename(Name) + ' from tb'
from syscolumns
where name! = N'姓名' and ID = object_id('tb') --表名 tb,不包含列名为姓名的其它列
order by colid asc
exec(@sql + ' order by 姓名')
--SQL SERVER 2005 动态 SQL。
select 姓名, 课程, 分数 from tb unpivot (分数 for 课程 in([语文] , [数学] , [物理])) t
--SQL SERVER 2005 动态 SQL,同 SQL SERVER 2000 动态 SQL。
```

问题：在上述的结果上加个平均分，总分，得到如下结果：

```
姓名   课程   分数
---- ------ ------
李四   语文   74.00
李四   数学   84.00
李四   物理   94.00
李四   平均分  84.00
李四   总分   252.00
张三   语文   74.00
张三   数学   83.00
张三   物理   93.00
张三   平均分  83.33
张三   总分   250.00
------------------
select * from
```

```
(
 select 姓名 as 姓名, 课程= '语文' , 分数= 语文 from tb
 union all
 select 姓名 as 姓名, 课程= '数学' , 分数= 数学 from tb
 union all
 select 姓名 as 姓名, 课程= '物理' , 分数= 物理 from tb
 union all
 select 姓名 as 姓名, 课程= '平均分' , 分数= cast((语文+ 数学+ 物理)*1.0/3 as decimal(18,2)) from
tb
 union all
 select 姓名 as 姓名, 课程= '总分' , 分数= 语文+ 数学+ 物理 from tb
) t
order by 姓名, case 课程 when '语文' then 1 when '数学' then 2 when '物理' then 3 when '平均
分' then 4 when '总分' then 5 end
```

```
drop table tb
```

日期转换参数,值得收藏

```
select CONVERT(varchar, getdate(), 120 )
--结果
2004-09-12 11:06:08
select replace(replace(replace(CONVERT(varchar, getdate(), 120 ),'-',''),' ',''),':','')
--结果
20040912110608
select CONVERT(varchar(12) , getdate(), 111 )
--结果
2004/09/12
select CONVERT(varchar(12) , getdate(), 112 )
--结果
20040912
select CONVERT(varchar(12) , getdate(), 102 )
--结果
2004.09.12
```

其它我不常用的日期格式转换方法:

```

select CONVERT(varchar(12) , getdate(), 101 )
--结果
09/12/2004
select CONVERT(varchar(12) , getdate(), 103 )
--结果
12/09/2004
select CONVERT(varchar(12) , getdate(), 104 )
--结果
12.09.2004
select CONVERT(varchar(12) , getdate(), 105 )
--结果
12-09-2004
select CONVERT(varchar(12) , getdate(), 106 )
--结果
12 09 2004
select CONVERT(varchar(12) , getdate(), 107 )
--结果
```

```
09 12, 2004

select CONVERT(varchar(12) , getdate(), 108 )
--结果
11:06:08

select CONVERT(varchar(12) , getdate(), 109 )
--结果
09 12 2004 1

select CONVERT(varchar(12) , getdate(), 110 )
--结果
09-12-2004

select CONVERT(varchar(12) , getdate(), 113 )
--结果
12 09 2004 1

select CONVERT(varchar(12) , getdate(), 114 )
--结果
11:06:08.177
```

固定列数的行列转换

如

```
select student,
sum(decode(subject,'语文', grade,null)) "语文",
sum(decode(subject,'数学', grade,null)) "数学",
sum(decode(subject,'英语', grade,null)) "英语"
from table
group by student;
```

```
结果
student     subject    grade
--------- ---------- --------
student1      语文       80
student1      数学       70
student1      英语       60
student2      语文       90
student2      数学       80
```

```
student2       英语       100
```
……

转换为

```
         语文  数学  英语
student1 80    70   60
student2 90    80   100
```

生成 10000 笔资料，可以是任意一数字

```
CREATE TABLE #t (id int IDENTITY(1,1) PRIMARY KEY,nums int)
GO

INSERT INTO #t
SELECT abs(CHECKSUM(NEWID())%100000)
GO 10000  ----生成笔资料,可以是任意一数字

SELECT * FROM #t
GO
DROP TABLE #t
GO
```

用代码在SQLSERVER 2005上实现一个JOB

```
USE [msdb]
GO
/****** Object:  StoredProcedure [dbo].[sp_create_job]    Script Date: 02/10/2009 19:44:33
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO

create proc [dbo].[sp_create_job]

as

BEGIN TRANSACTION
  DECLARE @jobLogpath VARCHAR(1000)
  DECLARE @JobID BINARY(16)
  DECLARE @ReturnCode INT
  SELECT @ReturnCode = 0
IF (SELECT COUNT(*) FROM msdb.dbo.syscategories WHERE name = N'[Uncategorized (Local)]')
< 1
  EXECUTE msdb.dbo.sp_add_category @name = N'[Uncategorized (Local)]'

  -- Delete the job with the same name (if it exists)
  SELECT @JobID = job_id
  FROM  msdb.dbo.sysjobs
  WHERE (name = N'FiveDayOptOutList')
  IF (@JobID IS NOT NULL)
  BEGIN
  -- Check if the job is a multi-server job
  IF (EXISTS (SELECT  *
            FROM   msdb.dbo.sysjobservers
            WHERE  (job_id = @JobID) AND (server_id <> 0)))
  BEGIN
    -- There is, so abort the script
    RAISERROR (N'Unable to import job ''FiveDayOptOutList'' since there is already a
multi-server job with this name.', 16, 1)
    GOTO QuitWithRollback
  END
  ELSE
    -- Delete the [local] job
    EXECUTE msdb.dbo.sp_delete_job @job_name = N'FiveDayOptOutList'
    SELECT @JobID = NULL
  END

BEGIN
/************************************************************************************
****************/
DEClARE @ServerRole Varchar(20)
DECLARE @UserName Varchar(20)
DECLARE @Logfile_name Varchar(100)
-------------------------------------------------------------------------------------
-----------------
if object_id('msdb.dbo.temppopedom') is not null drop table msdb.dbo.temppopedom
-------------------------------------------------------------------------------------
-----------------
/***select  all users information in SqLServer **/
select SrvRole = g.name, UserName = u.name, MemberSID = u.sid
into msdb.dbo.temppopedom
from sys.server_principals u, sys.server_principals g, sys.server_role_members m
where g.principal_id = m.role_principal_id
and u.principal_id = m.member_principal_id
order by 1, 2
-------------------------------------------------------------------------------------
-----------------
/***select the user's name who using database currently **/
select @UserName = loginame  from master.dbo.sysprocesses   where  uid=user_id()
-------------------------------------------------------------------------------------
-----------------
/***select the user's role who using database currently base on @UserName **/
select @ServerRole=SrvRole
from msdb.dbo.temppopedom
where Username=@UserName
-------------------------------------------------------------------------------------
-----------------
/************************************************************************************
****************/



 -- Add the job
EXECUTE @ReturnCode = msdb.dbo.sp_add_job
    @job_id = @JobID OUTPUT ,
    @job_name = N'FiveDayOptOutList',
    @owner_login_name = @UserName,
    @description = N'Execute package: FiveDayOptOutList',
    @category_name = N'[Uncategorized (Local)]',
    @enabled = 1,
    @notify_level_email = 0,
    @notify_level_page = 0,
    @notify_level_netsend = 0,
    @notify_level_eventlog = 2,
    @delete_level= 0
  IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback
------------------------------------------------------------------------
IF @ServerRole = N'sysadmin'
 BEGIN
     -- SysAdmin role Add the job steps
    EXECUTE @ReturnCode = msdb.dbo.sp_add_jobstep
    @job_id = @JobID,
    @step_name = N'CREATE TABLE ',
    @step_id = 1,
    @cmdexec_success_code = 0,
    @on_success_action = 1,
    @on_success_step_id = 0,
    @on_fail_action = 2,
    @on_fail_step_id=0,
    @retry_attempts = 0,
    @retry_interval = 0,
    @os_run_priority=0,
    @subsystem = N'CmdExec',
    @command=N'input you commond ',
    @output_file_name = @jobLogpath,
    @server = N'',
    @database_user_name = N'',
    @flags = 4
IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback
EXEC @ReturnCode = msdb.dbo.sp_update_job @job_id = @jobId, @start_step_id = 1
IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback
EXEC       @ReturnCode       =       msdb.dbo.sp_add_jobschedule       @job_id=@jobId,
@name=N'FiveDayOptOutList',
       @enabled=1,
       @freq_type=8,
       @freq_interval=2,
       @freq_subday_type=1,
       @freq_subday_interval=0,
       @freq_relative_interval=1,
       @freq_recurrence_factor=1,
       @active_start_date=20070319,
       @active_end_date=99991231,
       @active_start_time=230000,
       @active_end_time=235959
IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback
--EXEC @ReturnCode = msdb.dbo.sp_add_jobserver @job_id = @jobId, @server_name = N'(local)'
--IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback
END
ELSE
   ---non-SysAdmin Role add job
----------------------------------
EXECUTE @ReturnCode = msdb.dbo.sp_add_jobstep
       @job_id = @JobID,
   @step_name = N'CREATE TABLE ',
   @step_id = 1,
   @cmdexec_success_code = 0,
   @on_success_action = 1,
   @on_success_step_id = 0,
   @on_fail_action = 2,
   @on_fail_step_id=0,
   @retry_attempts = 0,
   @retry_interval = 0,
   @os_run_priority=0,
   @subsystem = N'CmdExec',
   @command=N'input you commond ',
   @server = N'',
   @flags = 16
IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback
EXEC @ReturnCode = msdb.dbo.sp_update_job @job_id = @jobId, @start_step_id = 1
IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback
EXEC       @ReturnCode       =       msdb.dbo.sp_add_jobschedule       @job_id=@jobId,
@name=N'FiveDayOptOutList',
       @enabled=1,
       @freq_type=8,
       @freq_interval=2,
       @freq_subday_type=1,
       @freq_subday_interval=0,
       @freq_relative_interval=1,
       @freq_recurrence_factor=1,
       @active_start_date=20070319,
       @active_end_date=99991231,
       @active_start_time=230000,
       @active_end_time=235959
IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback
EXEC @ReturnCode = msdb.dbo.sp_add_jobserver @job_id = @jobId, @server_name = N'(local)'
IF (@@ERROR <> 0 OR @ReturnCode <> 0) GOTO QuitWithRollback
```

```
END
------------------------
COMMIT TRANSACTION
GOTO   EndSave
QuitWithRollback:
  IF (@@TRANCOUNT > 0) ROLLBACK TRANSACTION
EndSave:
```