# access和sql区别

【第一】

sql中select sum(case when sign=1 then cj else 0 end) CJ1

access中要写成：select sum(iif(sign=1,cj,0)) CJ1

【第二 配置文件不同】

sql这样写:

<add key="sqlcon" value ="server=.;uid=sa;pwd=sa;database=primaryTest"/>

access这样写：

<add key="accessConnection" value="Provider=Microsoft.Jet.OLEDB.4.0;Data Source=|DataDirectory|\primaryTest.mdb"/>

<!--微软特地规定 "|DataDirectory|”为 app\_data目录 物理路径的系统代称在web.config的为Data Source=|DataDirectory|\date.mdb -->

【第三】

文本 nvarchar(n)

备注 ntext

数字(长整型) int

数字(整型) smallint

数字(单精度) real

数字(双精度) float

数字(字节) tinyint

货币 money

日期 smalldatetime

布尔 bit

# case when

case when shenhezt1=0 then '审核未通过' else case when shenhezt1=1 then '审核通过' else '未审核' end end as shenhezt1

不能是：

case when shenhezt1=NULL then '未审核' else case when shenhezt1=0 then '未通过' else '通过' end end as shenhezt1

（这样只有未通过和通过之分，没有未审核之分）

【总结：判断大于两种情况时，不能先判断NULL，应将NULL放在后面判断】

# SQL日期

## 含有年月日的时间段相减

string dttiime = bond\_dropdownlist.fanhui\_string("select convert(nvarchar(10),getdate(),120)");

double yinggaigeshu = Convert.ToDateTime(dttiime).Subtract (Convert .ToDateTime ("2011-07-11")).TotalDays ;

## 截取日期

Select datepart(day,getdate()),getdate()//截取天

Select datepart(minute,getdate()),getdate()//截取分钟

//以此类推

## 时间与sql时间字段转换

select getdate()

2012-11-08 16:22:37.737

select convert (nvarchar(10),getdate(),120)

2012-11-08

select convert (nvarchar(10),'2011-1-1 01:00:02',120)

2011-1-1 0

select convert (nvarchar(10),convert(datetime,getdate()),120)

2012-11-08

Select convert (nvarchar(10),convert(datetime,'2011-1-1 01:00:02'),120)

2011-01-01

推荐用最后一种写法，进行两次转换。（因为从第三个可以看出问题，不是2011-01-01）

## 时间格式

时间：

select CONVERT(varchar(100), GetDate(), 120)用在sql语句中用于获取服务器时间【2011-06-15 08:51:08】

select CONVERT(varchar(100), GetDate(), 102) 格式为：【2011.06.15】

select CONVERT(varchar(100), GetDate(), 111) 格式为：【2011/06/15】

Convert.ToDateTime(System.DateTime.Now).ToString("yyyy-mm-dddd")获取本机时间

SQL时间转换

Select CONVERT(varchar(100), GETDATE(), 0): 05 16 2006 10:57AM

Select CONVERT(varchar(100), GETDATE(), 1): 05/16/06

Select CONVERT(varchar(100), GETDATE(), 2): 06.05.16

Select CONVERT(varchar(100), GETDATE(), 3): 16/05/06

Select CONVERT(varchar(100), GETDATE(), 4): 16.05.06

Select CONVERT(varchar(100), GETDATE(), 5): 16-05-06

Select CONVERT(varchar(100), GETDATE(), 6): 16 05 06

Select CONVERT(varchar(100), GETDATE(), 7): 05 16, 06

Select CONVERT(varchar(100), GETDATE(), 8): 10:57:46

Select CONVERT(varchar(100), GETDATE(), 9): 05 16 2006 10:57:46:827AM

Select CONVERT(varchar(100), GETDATE(), 10): 05-16-06

Select CONVERT(varchar(100), GETDATE(), 11): 06/05/16

Select CONVERT(varchar(100), GETDATE(), 12): 060516

Select CONVERT(varchar(100), GETDATE(), 13): 16 05 2006 10:57:46:937

Select CONVERT(varchar(100), GETDATE(), 14): 10:57:46:967

Select CONVERT(varchar(100), GETDATE(), 20): 2006-05-16 10:57:47

Select CONVERT(varchar(100), GETDATE(), 21): 2006-05-16 10:57:47.157

Select CONVERT(varchar(100), GETDATE(), 22): 05/16/06 10:57:47 AM

Select CONVERT(varchar(100), GETDATE(), 23): 2006-05-16

Select CONVERT(varchar(100), GETDATE(), 24): 10:57:47

Select CONVERT(varchar(100), GETDATE(), 25): 2006-05-16 10:57:47.250

Select CONVERT(varchar(100), GETDATE(), 100): 05 16 2006 10:57AM

Select CONVERT(varchar(100), GETDATE(), 101): 05/16/2006

Select CONVERT(varchar(100), GETDATE(), 102): 2006.05.16

Select CONVERT(varchar(100), GETDATE(), 103): 16/05/2006

Select CONVERT(varchar(100), GETDATE(), 104): 16.05.2006

Select CONVERT(varchar(100), GETDATE(), 105): 16-05-2006

Select CONVERT(varchar(100), GETDATE(), 106): 16 05 2006

Select CONVERT(varchar(100), GETDATE(), 107): 05 16, 2006

Select CONVERT(varchar(100), GETDATE(), 108): 10:57:49

Select CONVERT(varchar(100), GETDATE(), 109): 05 16 2006 10:57:49:437AM

Select CONVERT(varchar(100), GETDATE(), 110): 05-16-2006

Select CONVERT(varchar(100), GETDATE(), 111): 2006/05/16

Select CONVERT(varchar(100), GETDATE(), 112): 20060516

Select CONVERT(varchar(100), GETDATE(), 113): 16 05 2006 10:57:49:513

Select CONVERT(varchar(100), GETDATE(), 114): 10:57:49:547

Select CONVERT(varchar(100), GETDATE(), 120): 2006-05-16 10:57:49

Select CONVERT(varchar(100), GETDATE(), 121): 2006-05-16 10:57:49.700

Select CONVERT(varchar(100), GETDATE(), 126): 2006-05-16T10:57:49.827

Select CONVERT(varchar(100), GETDATE(), 130): 18 ???? ?????? 1427 10:57:49:907AM

Select CONVERT(varchar(100), GETDATE(), 131): 18/04/1427 10:57:49:920AM

## 判断是否闰年

Select case day(dateadd(mm, 2,

dateadd(ms,-3,DATEADD(yy, DATEDIFF(yy,0,getdate()),

0)))) when 28 then '平年' else '闰年' end

或者select case

datediff(day,datename(year,getdate())+'-02-01',dateadd(mm,1,datename(year,getdate())+'-02-01')) when 28 then '平年' else '闰年' end

## 查询天数

//按照指定的年份和月份查询该月天数

declare @m int

set @m=2 --月份

select datediff(day,'2011-'+cast(@m as

varchar)+'-15' ,'2011-'+cast(@m+1 as

varchar)+'-15')

//查询当前月的天数

select datediff(day,cast(month(GetDate()) as

varchar)+'-'+cast(month(GetDate()) as varchar)+'-15'

,cast(month(GetDate()) as

varchar)+'-'+cast(month(GetDate())+1 as

varchar)+'-15')

//或者使用计算本月的最后一天的脚本，然后用DAY函数区最后一天

Select Day(dateadd(ms,-3,DATEADD(mm,DATEDIFF(m,0,getdate())+1, 0)))

## 去掉时分秒

declare @ datetime

set @ = getdate() --'2003-7-1 10:00:00'

Select @,DATEADD(day, DATEDIFF(day,0,@), 0)

## 显示星期几

select datename(weekday,getdate())

## 计算本年最后一天脚本

Select dateadd(ms,-3,DATEADD(yy, DATEDIFF(yy,0,getdate())+1, 0))

## 计算本年最后一天脚本

Select dateadd(ms,-3,DATEADD(yy, DATEDIFF(yy,0,getdate())+1, 0))

## 计算本月最后一天的SQL脚本

Select dateadd(ms,-3,DATEADD(mm, DATEDIFF(m,0,getdate())+1, 0))

## 去年的最后一天

//你需要在今年的第一天上减去3毫秒。

Select dateadd(ms,-3,DATEADD(yy, DATEDIFF(yy,0,getdate()), 0))

## 计算哪一天是本周的星期一

Select DATEADD(wk, DATEDIFF(wk,0,getdate()), 0)

## 计算这个月的第一个星期一

select DATEADD(wk, DATEDIFF(wk,0,dateadd(dd,6-datepart(day,getdate()),getdate())), 0)

## 计算本季度天数

declare @m tinyint,@time smalldatetime

select @m=month(getdate())

select @m=case when @m between 1 and 3 then 1

when @m between 4 and 6 then 4

when @m between 7 and 9 then 7

else 10 end

select @time=datename(year,getdate())+'-'+convert(varchar(10),@m)+'-01'

select datediff(day,@time,dateadd(mm,3,@time))

## 本季度的第一天

Select DATEADD(qq, DATEDIFF(qq,0,getdate()), 0)

## 显示这一年的第一天

Select DATEADD(yy, DATEDIFF(yy,0,getdate()), 0)

# DataTable操作

## 对dt表添加新行

decimal wanchenglv = 0;//不能定义string类型，因为他不能进行排序

DataTable dt = new DataTable();//自定义表

dt.Columns.Add("完成率");

dt.Columns["完成率"].DataType = System.Type.GetType("System.Decimal");//这个要有

if (a > 0)

{

jiaofeirenshu += a;

wanchenglv = (Math.Round(Convert.ToDecimal(jiaofeirenshu) \* 100 / Convert.ToDecimal(zhibiao), 2));

}

DataRow dr = dt.NewRow();

dr["姓名"] = renming;

dr["指标"] = zhibiao;

dr["交费人数"] = jiaofeirenshu;

dr["完成率"] = wanchenglv;

dt.Rows.Add(dr);

dt.DefaultView.Sort = "交费人数 DESC";

DataTable tem = dt.DefaultView.ToTable();//转换回表

GridView1.DataSource = tem;

GridView1 .DataBind ();

## dt表修改内容

for (int i = 0; i < dtt.Rows.Count;i++ )//编辑dtt表中的内容

{

DataRow dr1 = dtt.Rows[i];

dr1.BeginEdit();

dr1[1] = "设为管理员";

dr1[2] = "删除用户";

dr1.EndEdit();

}

## dt中添加Selec语句

DataTable dtt = dt.Clone(); //从dt表中克隆dtt

【1】 DataRow[] drr = dt.Select(" name='薛长城'"); //这个地方是写where语句，where不能有

for (int i = 0; i < drr.Length ; i++)

{

DataRow dr=dtt.NewRow ();

dr["ssn"] = drr[i]["ssn"];

dr["name"]=drr[i]["name"];

dr["userid"] = drr[i]["userid"];

dtt.Rows.Add(dr );

}

GridView.DataSource=dtt;

【2】或者用RowFilter

如 DataView dw = dt.DefaultView;

dw.RowFilter = "name='薛长城'";

GridView.DataSource=dw;

【3】如果自定义生成的表（dt）有冗余值。则通过一下方式去掉冗余

DataTable dtnew = dt.DefaultView.ToTable(true, new string[] {"姓名", "电话沟通", "面谈", "家访", "办公室咨询", "交报名费", "参观", "缴费"});//新建dtnew 表，将dt表中数据赋值给dtnew 表。注意第二个参数是dt中所有的字段名

GridView1.DataSource = dtnew;

GridView1.DataBind();

## 额外知识点

(1).DataTable dtnew= dt.Select("").CopyToDataTable();//将datarow 对象复制到制定的datatable中

(2).DataRow[] dr = ds.Tables["tbcalicoinfo"].Select("country is not null", "country asc");//通过表获取每一行

foreach(DataRow datarow in dr)

{

｝

# ExecuteNonQuery和ExecuteScalar方法

【1】命令对象的ExecuteNonQuery()方法用于执行任何不从数据库返回结果集的SQL操作命令，包括INSERT INTO、UPDATE、DELETE语句、没有返回数值的存储过程、CREATE TABLE和CREATEINDEX之类的DDL语句。ExecuteNonQuery方法还可用来执行目录操作，例如查询数据库的结构或创建诸如表等的 数据库对象。ExecuteNonQuery方法对于UPDATE、INSERT INTO和DELETE这些操作语句，其返回值为该命令所影响的行数。对于所有其他类型的语句，返回值为-1。如果发生回滚，返回值也为-1。

同时，

返回一个int类型变量，如果SQL语句是对数据库的记录进行操作（如记录的增加、删除和更新），那么方法将返回操作所影响的记录条数。

如：cmd.CommandText = “DELETE FROM Table1 WHERE [id]>1”;

int count = cmd.ExecuteNonQuery();

【2】----ExecuteScalar()：如果SQL语句是SELECT查询，则仅仅返回查询结果集中的第1行第1列，而忽略其他的行 和列。该方法所返回的结果为object类型，在使用之前必须强制转换为所需的类型。如果SQL语句不是SELECT查询，则返回结果没有任何作用。

同时，它可以将添加和查询同时使用如：

string sqlstr = "insert into zsgl\_jd\_canguan(daiduiren, daoxiaoshijian) values ('" + lbl\_daiduiren .Text .Trim () + "','" + txt\_comTime.Text + ")select @@IDENTITY as jdID";//@@IDENTITY as jdID返回新插入记录的ID号

string IID = bond\_dropdownlist.fanhui\_string(sqlstr);

public static string fanhui\_string(string sql)

{

string abc = "";

SqlConnection con = new SqlConnection(ConfigurationManager.AppSettings["sqlcon"]);

con.Open();

SqlCommand cmd = new SqlCommand(sql, con);

object result = cmd.ExecuteScalar();//注意这个地方，要赋给object对象才行，否则

if (result == null) //若 if (cmd.ExecuteScalar() == null) 则会插入两条相同记录

abc = "0";

else

abc = result.ToString();

con.Close();

return abc;

}

# 获取一个字段中指定的字符个数

SELECT sum(charindex('W',region)) AS 查找个数 FROM Employees a

语法 ：CHARINDEX ( expression1 , expression2 [ , start\_location ] )

expression1 　　一个表达式，其中包含要寻找的字符的次序。expression1 是一个短字符数据类型分类的表达式。

expression2 　　一个表达式，通常是一个用于搜索指定序列的列。expression2 属于字符串数据类型分类。

start\_location 　　在 expression2 中搜索 expression1 时的起始字符位置。如果没有给定 start\_location，而是一个负数或零，则将从 expression2 的起始位置开始搜索。

# Sql截取和拼凑

## 截取指定个数

(1).第一种。sql中substring从1开始，如substring( xieyibianhao,1,1)

另外：select distinct substring(monthandday,CHARINDEX('月',monthandday)-1,1) from [statistical \_data] 这个是用来切分用的。例如里面有6月1日-6月30日，7月1日-7月30日，那么只返回6，7两条数据。（monthandday是日期字段名）

(2)第二种。Sql从左边开始截取两位，如select distinct left(muOpt,2) from SysRoleMenu

## sql拼凑

如：省-市-县

select sheng+case when shi is null then '' else '-'+shi end +case when xian is null then '' else '-'+xian end from zsgl\_xieyi\_congxieyi where xieyibianhao='" + xieyibianhao + "'//这个要记住

# 查询所有表名或列名

## 查询指定数据库中所有的表名

select name from sysobjects where xtype='u'

## 读取指定表的所有列名

select name from syscolumns where id=(select max(id) from sysobjects where xtype='u' and name='表名')

# 获取刚插入的记录ID

第一种方法：

insert into zsgl\_jd\_canguan(daiduiren) values ('陈杰|j1000011')select SCOPE\_IDENTITY()

第二种方法：（最好用第二种方法）

insert into zsgl\_jd\_canguan(daiduiren) values ('陈杰|j1000011')select @@IDENTITY as jdID

# sql基本操作

## 复制表到另一个表

(1).全部复制：

insert into table1 select\*from table22

(2).如果是有选择性的复制数据表达如下：

insert into table1(column1,column2,column3...) select column1,column2,colunm3... from table23、

## 删除表中记录

删除表中所有行：TRUNCATE TABLE

删除表中的行：DELETE 语句

删除表：DROP TABLE

## 最基础的语句

选择：select \* from table1 where 范围

插入：insert into table1(field1,field2) values(value1,value2)

删除：delete from table1 where 范围

更新：update table1 set field1=value1 where 范围

查找：select \* from table1 where field1 like ’%value1%’ （所有包含‘value1’这个模式的字符串）---like的语法很精妙，查资料!

排序：select \* from table1 order by field1,field2 [desc]

总数：select count(\*) as totalcount from table1

求和：select sum(field1) as sumvalue from table1

平均：select avg(field1) as avgvalue from table1

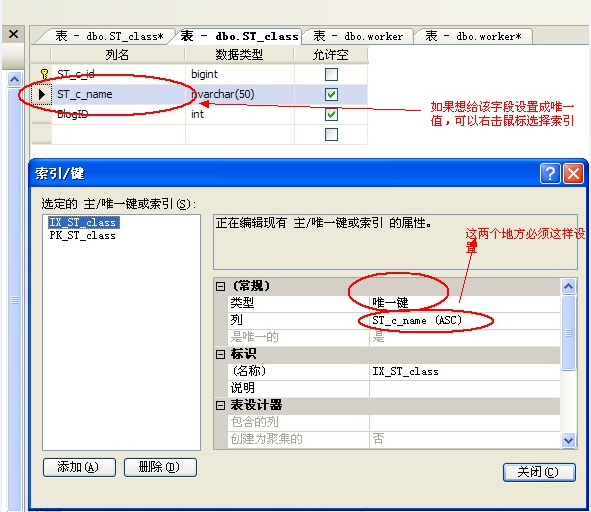
最大：select max(field1) as maxvalue from table1

最小：select min(field1) as minvalue from table1[separator]

## Exists注意事项

SELECT birthdate FROM Employees a WHERE exists(SELECT birthdate FROM employees b WHERE b.birthdate != '1952-02-19 0:00:00' and a.birthdate=b.birthdate)（注意最后一句，这样exists才能和in一样起作用）

# 设置某一字段为唯一值



# 判断库，表，存储过程等是否存在

## 判断临时表

If Object\_id( 'TempDB..临时表名 ') Is Not Null

Drop Table 临时表名

if exists(select \* from master..sysdatabases where name=N'库名')

print 'exists'

else

print 'not exists'

## 判断表名

if exists (select \* from dbo.sysobjects where id = object\_id(N'[dbo].[表名]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)

-- 删除表

drop table [dbo].[表名]

GO

## 判断存储过程名

if exists (select \* from dbo.sysobjects where id = object\_id(N'[dbo].[存储过程名]') and OBJECTPROPERTY(id, N'IsProcedure') = 1)

-- 删除存储过程

drop procedure [dbo].[存储过程名]

GO

## 判断视图名

if exists (select \* from dbo.sysobjects where id = object\_id(N'[dbo].[视图名]') and OBJECTPROPERTY(id, N'IsView') = 1)

-- 删除视图

drop view [dbo].[视图名]

GO

## 判断函数名

if exists (select \* from dbo.sysobjects where id = object\_id(N'[dbo].[函数名]') and xtype in (N'FN', N'IF', N'TF'))

-- 删除函数

drop function [dbo].[函数名]

GO

if col\_length('表名', '列名') is null

print '不存在'

select 1 from sysobjects where id in (select id from syscolumns where name='列名') and name='表名'

# 自己做的sql个数统计

string sqlstr = "select addtime,biaoji,sfz into #temp from zsgl\_cg\_xueshengTB where 1=1";

if (ddl\_biaoshi.SelectedItem.Text != "请选择...")

{

sqlstr += " and biaoji='" + ddl\_biaoshi.SelectedItem.Text + "'";

}

if (txt\_kstime.Text != "")

{

sqlstr += " and addtime>='" + txt\_kstime.Text.Trim() + "'";

}

if (txt\_jstime.Text != "")

{

sqlstr += " and addtime<='" + txt\_jstime.Text.Trim() + " 23:59:59'";

}

sqlstr +=

@"; select count( zsgl\_cg\_xueshengTB.cardID) as cheshangrenshu,#temp.biaoji into #temp1 from zsgl\_cg\_xueshengTB inner join #temp on #temp.sfz=zsgl\_cg\_xueshengTB.sfz group by #temp.biaoji order by biaoji;

select distinct count(1) as baomingrenshu,#temp.biaoji into #temp2 from zsgl\_xueshengxinxi as a inner join zsgl\_jd\_canguanxuesheng as b on a.ID=b.mubiaoshengyuanID inner join #temp on #temp.sfz=a.shenfenID group by biaoji order by biaoji ;

select distinct count(1) as xitongrenshu,#temp.biaoji into #temp3 from #temp inner join zsgl\_xueshengxinxi as b on #temp.sfz=b.shenfenID group by #temp.biaoji order by biaoji;

select #temp.biaoji,cheshangrenshu,baomingrenshu,xitongrenshu into #tempzhong from #temp,#temp1,#temp2,#temp3 where #temp.biaoji=#temp1.biaoji and #temp1.biaoji=#temp2.biaoji and #temp2.biaoji=#temp3.biaoji

select distinct \* from #tempzhong order by biaoji";

//自己做的，经测试可以

# 大量数据导入数据库（文本文件）

## 第一种：（推荐）单线程

private void ImportData()

{

using (SqlConnection conn = new SqlConnection(ConfigurationManager.ConnectionStrings["connStringName"].ConnectionString))

{

using (SqlCommand cmd = new SqlCommand())

{

if (conn.State != ConnectionState.Open)

conn.Open();

cmd.Connection = conn;

cmd.CommandTimeout = 900;//长城设置，15分钟即超时

cmd.CommandText = "BULK INSERT ChipData FROM '" + path + "' WITH (FIELDTERMINATOR = ' ',KEEPIDENTITY,ROWTERMINATOR='\n',BATCHSIZE = 100000)";

try

{

Stopwatch timer = new Stopwatch();

timer.Start();

cmd.ExecuteNonQuery();

timer.Stop();

HDIC\_Message.ShowInfoDialog(null, "数据导入成功,用时：" + timer.ElapsedMilliseconds.ToString() + "毫秒");

conn.Close();

}

catch (System.Exception ex)

{

conn.Close();

HDIC\_Message.ShowInfoDialog(null, "导入数据库失败，原因：\r\n" + ex.ToString());

}

}

}

}

## 第二种：（单线程）

protected void F1()

{

SqlConnection conn = new SqlConnection(str);

conn.Open();

System.Diagnostics.Stopwatch timer = new Stopwatch();

using (System.Data.SqlClient.SqlBulkCopy sqlBC = new System.Data.SqlClient.SqlBulkCopy(conn))

{

sqlBC.BatchSize = 100000;

sqlBC.BulkCopyTimeout = 60;

sqlBC.DestinationTableName = "dbo.TableB";

sqlBC.ColumnMappings.Add("valueA", "Value1");

timer.Start();

sqlBC.WriteToServer(dt);

timer.Stop();

}

conn.Dispose();

MessageBox.Show(timer.ElapsedMilliseconds.ToString());

}

private void GetData()

{

dt = new DataTable();

dt.Columns.Add("valueA", typeof(string));

for (int i = 0; i < 1000000; i++)

{

DataRow dr = dt.NewRow();

dr["valueA"] = i.ToString();

dt.Rows.Add(dr);

}

}

## 第三种（多线程）

private void button1\_Click(object sender, EventArgs e)

{

Thread thread1 = new Thread(new ThreadStart(F1));

thread1.Start();

Thread thread2 = new Thread(new ThreadStart(F1));

thread2.Start();

Thread thread3 = new Thread(new ThreadStart(F1));

thread3.Start();

Thread thread4 = new Thread(new ThreadStart(F1));

thread4.Start();

Thread thread5 = new Thread(new ThreadStart(F1));

thread5.Start();

}

protected void F1()

{

System.Diagnostics.Stopwatch timer = new Stopwatch();

timer.Start();

//虽然方法是静态的，但是该方法内部是使用实例方式实现的

SqlProvider.ExecuteNonQuery("BULK INSERT TableB FROM 'c:\\20w.txt' WITH (FIELDTERMINATOR = ',',ROWTERMINATOR='|',BATCHSIZE = 100000)");

timer.Stop();

MessageBox.Show(timer.ElapsedMilliseconds.ToString());

}

# 随机

select top 10 \* from tb\_test where testCourse='" + Session["KM"].ToString() + "' order by newid() //newid()是随机抽取而且是不重复的值

# 存储过程

## 直接定义存储过程

SqlCommand com = new SqlCommand("returnds", con);

com.CommandType = CommandType.StoredProcedure;

SqlParameter[] parms ={

new SqlParameter("@dtstart",SqlDbType.DateTime),

new SqlParameter("@dtend",SqlDbType.DateTime),

new qlParameter("@txt\_name",SqlDbType.NVarChar),

};

parms[0].Value = dt\_start.Value;

parms[1].Value = dt\_end.Value;

parms[2].Value = txt\_name.Text.Trim();

foreach (SqlParameter parm in parms)

{

com.Parameters.Add(parm);

}

try

{

// com.ExecuteNonQuery();//因为返回的是查询的table表，所以不用该语句，但是若是添加删除等，必须执行该语句

}

catch (Exception ex)

{ }

SqlDataAdapter sda = new SqlDataAdapter();

sda.SelectCommand = com;

//上面也可以这样写-------------------------

List<QuaryParameter> lst = new List<QuaryParameter>();

QuaryParameter par = new QuaryParameter();

par.DbType = DbType.DateTime;

par.Value = dtDelTime;

par.ParameterName = "@delTime";

lst.Add(par);

## 调用数据库存储过程

con1.Open();

SqlCommand cmd = new SqlCommand();

cmd.CommandType = CommandType.StoredProcedure;

cmd.CommandText = "FingerprintToCard";

cmd.Connection = con1;

cmd.Parameters.Add(new SqlParameter("@Return", SqlDbType.Int));

cmd.Parameters["@Return"].Direction = ParameterDirection.ReturnValue;

cmd.ExecuteNonQuery();

if (int.Parse(cmd.Parameters["@Return"].Value.ToString()) == 0)

{

MessageBox.Show("转换成功");

}

## 存储过程加密

存储过程加密《with encryption》：代码如下：

if exists (select \* from sysobjects where name='cunchu1' and type='p')

drop procedure cunchu1

go --注意这个go要有，否则下面创建存储过程不会执行了

create procedure cunchu1 with encryption

as

begin

select \* from myfaxingssue where 人员编号='j0005973'

end

这样的好处是加密后存储过程以不可读的形式存储，用户无论是使用企业管理器还是系统存储过程sp\_helptext 都无法查看存储过程的代码

例如执行exec sp\_helptext cunchu1 显示的结果是“对象 'cunchu1' 的文本已加密”

## Declare作用

不需要Declare 定义的，是给你传进来的参数；

需要Declare 定义的是你在存储过程中将要用到的参数。注意不用declare定义的变量要放到as上面声明

## 细节注意

create procedure cunchu1

as

declare @sql nvarchar(1000)

begin

set @sql= 'select \* from [tcc].[dbo].myfaxingssue where 人员编号=''j0005973'''

end

exec(@sql)

（上面j0005973必须用双引号，同时@sql必须用单引号，注意和【2】中不一样）

同时下面也是不同的：

set @sql= 'select \* from [tcc].[dbo].myfaxingssue where 人员编号='''+@empid+''''

等于select \* from [tcc].[dbo].myfaxingssue where 人员编号=''+@empid+''

等于select \* from [tcc].[dbo].myfaxingssue where 人员编号=@empid

# 用索引提高检索速度

聚集索引

**create clustered index 索引名称 on 表名(字段1[,字段2,字段3...])**

非聚集索引

**create index 索引名称 on 表名(字段1[,字段2,字段3...])**

一般建议索引名称以 idx\_表名\_字段1\_字段2 如此命名规则。

一个表只能创建一个聚集索引

可以创建多个非聚集索引

删除索引

drop index idx\_data\_datetime on car\_data

下面是具体实例：

//用非聚集索引

create index idx\_data\_datetime on car\_data(datetime)

declare @m1 datetime

set @m1 =getdate()

SELECT \*

FROM Car\_Data with (index(idx\_data\_datetime))//用索引查询，要记住格式：用with

where datetime>'2010-1-1'

select [语句执行花费时间(毫秒)]=datediff(ms,@m1,getdate())

declare @m datetime

set @m =getdate()

SELECT \*

FROM Car\_Data with (index(0))//不用索引查询

where datetime>'2010-1-1'

select [语句执行花费时间(毫秒)]=datediff(ms,@m,getdate())

/\*注意如果想查看各个sql语句运行用的时间：在各个select语句前加：declare @d datetime

set @d=getdate()

并在select语句后加：

select [语句执行花费时间(毫秒)]=datediff(ms,@d,getdate())

\*/

# Trans方法

protected SqlConnection Connection ;

public bool ExecuteSQL(ArrayList SqlStrings)//执行多个非查询的SQL命令

{

bool success = true;

this.Open();

SqlCommand cmd = new SqlCommand();

SqlTransaction trans = Connection.BeginTransaction();

cmd.Connection = Connection;

cmd.Transaction = trans;

try

{

foreach (String str in SqlStrings)

{

cmd.CommandText = str;

cmd.ExecuteNonQuery();

}

trans.Commit();

}

catch(Exception d)

{

success = false;

trans.Rollback();

}

finally

{

this.Close();

}

return success;

}

# 代码新建表并设置唯一值和约束

create table worker

(

w\_id int identity (1000,1) not null,

w\_name Nvarchar(10) unique,

w\_age SMALLINT CONSTRAINT CK\_W\_AGE CHECK(w\_age>20 and w\_age<150),

w\_pay money DEFAULT 0,

CONSTRAINT PK\_W\_ID PRIMARY KEY(W\_ID)

)

# Sql中排序规则不一致

无法解决 equal to 操作中 "SQL\_Latin1\_General\_CP1\_CI\_AS" 和 "Chinese\_PRC\_CI\_AS" 之间

--解决办法

select \* from table1 where ID COLLATE Chinese\_PRC\_CI\_AS not in (select ID from table2 )

在查询字段连接条件后加上：COLLATE Chinese\_PRC\_CI\_AS 即可