ÔN TẬP LẬP TRÌNH CSDL

// Class kết nối

// Cách gọi hàm: <Tên_class>.<Tên hàm>(<Đối số>,...);

```
class clsKetnoi

    {
        public SqlConnection conn;
        // hàm tạo kết nối CSDL
        public void ketnoi()
        {
            try
            {
                conn = new SqlConnection(@"Data Source=<TEN SEVER>;Initial Catalog=<Tên
CSDL>;Persist Security Info=True;User ID=<Tên ddangw nhập>;Password=system");
                conn.Open();
            }
            catch
            {
                MessageBox.Show("Kết nối CSDL thất bại!");
            }
        }
        public void dongketnoi()
        {
            if(conn.State == ConnectionState.Open)
            {
                conn.Close();
            }
        }
        // hàm đổ dữ liệu vào DataGridView
        public static bool Napdulieu(DataGridView dg, string sql)
        {
            clsKetnoi kn = new clsKetnoi();
            kn.ketnoi();
            SqlCommand cm = new SqlCommand(sql, kn.conn);
            SqlDataAdapter da = new SqlDataAdapter(cm);
            DataTable dt = new DataTable();
            da.Fill(dt);
            kn.dongketnoi();
            if (dt.Rows.Count != 0)
            {
                dg.DataSource = dt;
                return true;
            }
            else
                return false;
        }
       // hàm lấy bảng dữ liệu từ CSDL
        public static DataTable Laybang(string sql)
        {
            clsKetnoi kn = new clsKetnoi();
            kn.ketnoi();
            DataTable dt = new DataTable();
            try
            {
                SqlDataAdapter da = new SqlDataAdapter(new SqlCommand(sql, kn.conn));
                da.Fill(dt);
            }
```

```
    catch
    {
        dt = null;
    }
    return dt;
}
// hàm Insert, Update, Delete dữ liệu từ CSDL
public static bool Dieuchinh(string sql)
{
    clsKetnoi kn = new clsKetnoi();
    kn.ketnoi();
    try
    {
        if (new SqlCommand(sql, kn.conn).ExecuteNonQuery() > 0)
            return true;
        else
            return false;
    }
    catch
    {
        return false;
    }
    finally
    {
        kn.dongketnoi();
    }
}
// Hàm đọc dữ liệu từ CSDL (dùng DataReader)
public static SqlDataReader Doctungdongdl(string chuoi)
{
    clsKetnoi kn = new clsKetnoi();
    kn.ketnoi();
    SqlDataReader dr;
    SqlCommand cm = new SqlCommand(chuoi, kn.conn);
    try
    {
        dr = cm.ExecuteReader();
        return dr;
    }
    catch (Exception)
    {
        return null;
    }
}
// Đưa dữ liệu vào Combobox
public static void Truyendl(ComboBox cb, string chuoi, string vl, string dip)
{
    clsKetnoi kn = new clsKetnoi();
    kn.ketnoi();
    try
    {
        kn.ketnoi();
        SqlDataAdapter da = new SqlDataAdapter(chuoi, kn.conn);
        DataSet ds = new DataSet();
        da.Fill(ds);
        cb.DataSource = ds.Tables[0];
        cb.ValueMember = vl;
        cb.DisplayMember = dip;
    }
    catch (Exception)
    {

    }
    finally
    {
```

```
                kn.dongketnoi();
            }
        }
        // Cách dùng DataReader
        //SqlDataReader dr = clsKetnoi.Doctungdongdl("Exec Laynhanvien");
        //while (dr.Read())
        //{
            //dgNhanvien.Rows.Add(dr[0], dr[1], dr[2]);
        //}
    }
```

//

*** Lưu ý:

// lấy giá trị được chọn trong combobox

```
<Tên name Combobox>.SelectedValue.ToString();
```

// Tạo combobox thủ công

```
            comboBox.DisplayMember = "Text";
            comboBox.ValueMember = "Value";
            comboBox.Items.Add(new { Text = "report AT", Value = "reportA" });
            comboBox.Items.Add(new { Text = "report BT", Value = "reportB" });
            comboBox.Items.Add(new { Text = "report CT", Value = "reportC" });
            comboBox.Items.Add(new { Text = "report DT", Value = "reportD" });
            comboBox.Items.Add(new { Text = "report ET", Value = "reportE" });
```

// Procedure

```
Create proc  Xuatsach
      @masach varchar(6), @soluong int
      as
            Declare @slhienco int = (select soluong from sach where masach = @masach)
            if @slhienco >= @soluong
            begin
                  set @slhienco = @slhienco-@soluong
                  Update sach
                  Set
                        soluong = @slhienco
                  Where
                        masach = @masach
            end
            else
            return 0
      go

Exec Xuatsach 'S1', '20'
```

// Trigger

```
-- Tự động tăng số lần mượn của sách
Create trigger Capnhatsolanmuon
on phieumuon
for insert
as
      Begin
            DECLARE @masach varchar(6) = (SELECT masach FROM inserted)
            Declare @cu int = (Select solanmuon from sach where masach = @masach)
            Declare @moi int = (Select soluong from inserted where masach = @masach)
```

```
            Set @moi = @moi + @cu
            Update dbo.sach
                  Set
                        solanmuon = @moi
                  Where
                        masach = @masach
      End
go -- đã duyệt

-- khi trả sách thì cập nhật lại số lượng sách đang có
Create trigger Capnhatsoluongsach
On phieumuon
for update
as
      Begin
            if UPDATE(trangthai)
            begin
            -- Thuc thi
            Declare @masach varchar(6) = (Select i.masach from inserted i)
            Declare @sotra int = (Select i.soluong from inserted i)
            Declare @sohienco int = (Select s.soluong from sach s where masach =
@masach)
            Set @sohienco = @sohienco + @sotra
            Update sach
                  Set
                        soluong = @sohienco
                  where
                        masach = @masach
            end
      End
go -- đã duyệt
```

//

1. Form

```
// Kiểm tra tồn tại Form
private Form KiemTraTonTai(Type ftype)
{
    foreach (Form f in this.MdiChildren)
    {
         if (f.GetType() == ftype)
             return f;
     }
         return null;
}
```

```
// Hiển thị Form con

Form frm = this.KiemTraTonTai(typeof(frmCon));
if (frm != null)
{
     frm.Activate();
}
else
{
     frmCon a = new frmCon();
     a.MdiParent = this;
     a.Show();
 }
```

2. Chuỗi:

string str ,str1,str2.
str.Length: lấy chiều dài.
String.Compare(str1,str2,true) == 0, <0,>0 so sánh hai chuỗi không phân biệt hoa thường.(str1 bằng, nhỏ hơn, lớn hơn str2)
Ví dụ:
Xem chuỗi 1 có bằng chuỗi 2 không.
if(String.Compare(str1,str2,true)==0)
Console.Write("Bằng nhau, không phân biệt hoa thường");
else if(String.Compare(str1,str2,true) < 0)
Console.Write("str1 nhỏ hơn str2, không phân biệt hoa thường");(cái kia tương tự)
String.Compare(str1,str2,false) y nhu trên. nhưng phân biệt hoa thường.
Str1.Contains(Str2) :Kiểm tra trong chuỗi Str1 có chuỗi Str2 hay không?
Str1.IndexOf(„t“):Vị trí xuất hiện đầu tiên của ký tự t trong Str1 .Trả về -1 nếu trong Str1 không có ký tự t
Str1.LastIndexOf(„t“):Vị trí xuất hiện cuối cùng của ký tự t trong Str1 Trả về -1 nếu trong Str1 không có ký tự t
Str1.Remove(1):Xóa chuỗi Str1 từ vị trí 1 đến hết
Str.Remove(1,5) :Xóa 1 chuỗi con trong Str1 có chiều dài là 5
Str1.StartsWith(Str2):Kiểm tra xem chuỗi Str1 có bắt đầu bằng chuỗi Str2 không?
Str = Str.Replace(„,“,“.“):Thay thế dấu , bằng dấu . trong chuỗi Str
Str = Str.Replace(“xx”,”yy”):Thay thế chuỗi xx bằng chuỗi yy trong chuỗi Str
Str1 = Str.SubString(2):Tạo chuỗi con từ chuỗi Str bắt đầu từ vị trí 2 đến hết
Str1 = Str.ToLower() :Chuyển chuỗi sang chữ thường
Str1 = Str.ToUpper() Chuyển chuỗi sang chữ hoa
Str = Str.Trim() Cắt hết khoảng trắng ở đầu và cuối chuỗi
Str = Str.TrimLeft() Cắt hết khoảng trắng ở đầu chuỗi
Str = Str.TrimRight() Cắt hết khoảng trắng ở cuối chuỗi

3. Regex:

- Trong thực tế chúng ta thường thấy các trang Web dùng Regex để kiểm tra định dạng của 1 chuỗi Input xem có hợp lệ hay ko, thường là khi đăng ký account nào đó, nếu định dạng ko khớp với định dạng cho trc thì hệ thống sẽ thông báo, như email chẳng hạn, nếu chúng ta quy định là email phải có @….Com, thì hiển nhiên khi đăng ký mà ko có chữ @ hệ thống sẽ báo lỗi ngay.
- VD dau đây để so khớp định dạng chuỗi Email :
String pattern = "^(\w+@\w+\.\w+)$";
- Biểu thức chính quy được xây dựng dựa vào ký tự đại diện và những nguyên tắc sau :
Ký tự đại diện
[0-9] hay \d: đại diện 1 ký tự số từ 0 – 9
[0-9a-zA-Z_] hay \w: Đại diện ký tự aphabet hay ký tự gạch chân(hoa thường).
.(dấu chấm): đại diện ký tự bất kỳ.
\s : đại diện ký tự trắng (\r\n\t\f)
[xyz] đại diện 1 ký tự x,y hặc z.
\D: đại diện ký tự không thuộc \d
\W : đại diện ký tự không thuộc \w
[^xyz]: đại diện ký tự không thuộc xyz.
^: chỉ ra ký tự bắt đầu
$: chỉ ra ký tự kết thúc
\\,\.,\$,\^: đại diện ký tự ‘\,’.’,’$’,’^’.
Số lần xuất hiện :
{n,m}: ít nhất n lần, nhiều nhất m lần
{,m}: nhiều nhất m lần
{n,}: ít nhất n lần

{n}: chính xác n lần
?: tương đương {0,1}
*** : tương dương {0,vô cùng}**
+ : tương đương {1,vô cùng}
KHÔNG CHỈ ĐỊNH: 1 lần

Ví dụ :

```
// Số chứng minh nhân dân 9 ký tự
String cmnd = "[0-9]{9}";
// số điện thoại 10 số hay 11 số
// bắt đầu bằng 0
String PhoneNumber = "0\d{9,10}";
// Số xe máy Sài Gòn : VD như 51-Z8-111.11
String MotoNumber = "5\d-[A-Z]\d-\d{3}.\d{2}";
// Địa chỉ Email
String Email = "\w+.@\w+.\w{2,4}";
```

- Khi đã hiểu được cách lập biểu thức chính qui chúng ta có thể kiểm tra một chuỗi bất kỳ xem có đúng định dạng hay không, cách đơn giản nhất là dùm phương thức IsMach().

```
bool Regex.IsMach(String input,String Pattern);
```

Vd:

```
1.  if (String.IsNullOrEmpty(txtHo.Text) || String.IsNullOrEmpty(txtTen.Text))
2.                  lblWarning.Text = "khong de trong ho va ten";
3.              else if ((DateTime.Now.Year - dtpNgaySinh.Value.Year) < 20)
4.                  lblWarning.Text = "tuoi nhan vien ko duoc duoi 20";
5.              else if (double.Parse(txtLuong.Text) < 9000000)
6.                  lblWarning.Text = "Luong toi thieu la 9tr";
7.              else if (!Regex.IsMatch(txtSoCMND.Text, @"^([0-9]{9})$"))
8.                  lblWarning.Text = "So CMND ko hop le";
9.              else if (!Regex.IsMatch(txtEmail.Text, @"^(\w+@\w+\.\w+)$"))
10.                 lblWarning.Text = "Email ko hop le";
11.             else if (!Regex.IsMatch(txtMobile.Text, @"^(0\d{9,10})$"))
12.                 lblWarning.Text = "So Phone ko hop le";
13.             else if (!Regex.IsMatch(txtSoXeMay.Text, @"^(5\d-[A-Z]\d-\d{3}\.\d{2})$"))
14.                 lblWarning.Text = "So Xe ko hop le";
```

4. Datetime

- **Add:** Trả về ngày giờ mới có thể tăng hoặc giảm tùy thuộc vào tham số truyền vào hàm Timespan. Bạn có thể truyền giá trị âm vào TimeSpan để lấy giá trị giảm.
- **AddYears**: Trả về ngày tháng mới mà giá trị thay đổi là Năm. Bạn có thể nhận một giá trị tăng hoặc giảm tùy vào tham số truyền vào
- **AddMonths:** Trả về ngày giờ mới mà giá trị thay đổi là Tháng và Năm . . .
- **AddDays:** Trả về ngày giờ mới mà giá trị thay đổi là Ngày, Tháng Năm . . .
- **AddHours:** Trả về ngày giờ mới mà giá trị thay đổi là Giờ, Ngày, Tháng Năm . . .
- **AddMinutes:** Trả về ngày giờ mới mà giá trị thay đổi là Phút, Giờ, Ngày, Tháng Năm . . .
- **AddSeconds:** Trả về ngày giờ mới mà giá trị thay đổi là Giây, Phút, Giờ, Ngày, Tháng Năm . . .
- **AddMilliseconds:** Tương tự như trên và thay đổi tới mức Mili giây . . .
- **AddTicks:** Trả về ngày giờ mới và thay đổi tới mức siêu nhỏ (nanoseconds). Tham số truyền vào có giá trị gấp 100 lần nanoseconds

Kiểu dữ liệu DateTime trong C# sẽ trả về ngày tháng dạng: ***09/23/2013 01:00:00 AM***

Dưới đây là các ký tự được dùng cho việc quy định hiển thị một phần giá trị của định dạng trên:
- y hiển thị tháng và năm dạng: ***Septemper 2013***
- yy hiển thị 2 số cuối của năm: ***13***
- yyy hiển hị năm 3 số nếu năm 4 số thì tự chuyển qua: ***2013***
- yyyy hiển thị năm dạng 4 số: ***2013***
- M hiển thị tháng dạng chữ cái + ngày dạng số: ***Septemper 23***
- MMM hiển thị tháng dưới dạng 3 chữ cái đầu tiên trong tên tháng: ***Sep***
- MMMM hiển thị tháng dạng chữ cái đầy đủ: ***Septemper***
- d hiển thị ngày tháng năm dạng: ***23/09/2013***
- dd hiển thị chỉ ngày dạng số: ***23***
- ddd hiển thị thứ của ngày dạng 3 chữ cái: ***Mon***
- dddd hiển thị thứ của ngày dạng đầy đủ: ***Monday***

- HH hiển thị giờ theo dạng 24h
- mm hiển thị phút dưới dạng 2 số
- ss hiển thị giây dạng 2 số
- tt hiển thị AM/PM : ***01:21:45:AM***
- N Hiển thị dấu phẩy cho dãy số. Có thể tùy chỉnh N0, N1, N2 . . .

Tuy nhiên, chúng ta có cách khác ngoài việc dùng các ký tự trong lúc in ra ngày giờ bằng cách sau:

```
DateTime today= DateTime.Now;
int year = today.Year;
int month = today.Month;
int day = today.Day;
int hour = today.Hour;
int minute = today.Minute;
int second = today.Second;
int millisecond = today.Millisecond;
```

Các biến ở trên get ra Năm, Tháng, Ngày, Giờ . . . hiện tại của ngày đã cho

```
long tick = today.Ticks;
```

Đối với biến tick mà chúng ta get ra được ở trên nó khác hoàn toàn với các biến trên nó,
bởi vì tick này chính là tổng ngày tháng, giờ....tính ra dạng ticks. Cho độ chính xác cao, từ biến này ta có thể in ra các kết quả dạng

```
//          21,338,580,000,000,000 nanoseconds
//          2,190,385,800,000,000 ticks
//          219,333,580.00 seconds
//          2,555,643.00 minutes
//..................... hour, day, month, year
```