

Chương 2:

Các thành phần của ADO.NET



Nội dung

1. Đối tượng Connection

2. Đối tượng Command

3. Đối tượng Parameter

4. Đối tượng DataReader

5. Đối tượng DataAdapter

6. DataSet

2. Đối tượng Command

2. Đối tượng Command

Tạo CSDL BanHang

MTEmployee

	Column Name	Condensed Type	Nullable
🔑	empid	nvarchar(10)	No
	empname	nvarchar(50)	Yes
	tell	nvarchar(50)	Yes
	idcard	nvarchar(15)	Yes
	address	nvarchar(100)	Yes
	positionid	nvarchar(2)	Yes
	active	bit	Yes
	username	nvarchar(20)	Yes
	modifieddate	datetime	Yes



MTPosition

	Column Name	Condensed Type	Nullable
🔑	positionid	nvarchar(2)	No
	positionname	nvarchar(50)	Yes
	active	bit	Yes

MTUsers

	Column Name	Condensed Type	Nullable
🔑	username	nvarchar(20)	No
	password	nvarchar(100)	Yes
	fullname	nvarchar(50)	Yes
	modifieddate	datetime	Yes
	active	bit	Yes
	role	nvarchar(10)	Yes

2. Đối tượng Command

Tạo Procedure trong CSDL **BanHang** thông tin như Sau:

Tên Procedure: **SP_DisplayUser**

Nội dung: **Hiển thị danh sách tất cả user có trong bảng MTUsers**

Thực thi thủ tục: **exec SP_Displayuser**

2. Đối tượng Command

2.1 Tạo đối tượng Command

2.2 Một số thuộc tính thường dùng của Command

2.3 Các phương thức thường dùng của Command

2. Đối tượng Command

2.1 Tạo đối tượng Command

❖ Đối tượng Command:

Là đối tượng **cho phép truy cập CSDL** và **thực thi phát biểu SQL** (Select, insert, delete, update,) hay **thủ tục Store Procedure** của CSDL, **truyền tham số** và **trả về dữ liệu**. Các lệnh *chỉ thực thi sau khi đối tượng Connection được thiết lập thành công* (tức là đã kết nối đến CSDL).

2. Đối tượng Command

❖ Khai báo đối tượng Command:

```
SqlCommand cmd;
```

❖ Khởi tạo: Có thể khởi tạo đối tượng cmd bằng 4 phương thức khởi tạo (constructor).

- Constructor không tham số:

```
cmd = new SqlCommand();
```

- Constructor một tham số:

```
cmd = new SqlCommand(<lệnh SQL>)
```

- Constructor hai tham số:

```
cmd = new SqlCommand(<lệnh SQL>, <đối tượng kết nối>)
```


2. Đối tượng Command

❖ Tạo đối tượng Command trực tiếp từ đối tượng Connection

C1: Tạo đối tượng Command bằng *phương thức khởi tạo không tham số*, sau đó thiết lập giá trị cho các thuộc tính của đối tượng Command sau khi đã khai báo

Ví dụ:

```
String connStr="DataSource=.; Database=HoaDon;
```

```
Integrated Security=true;";
```

```
SqlConnection conn = new SqlConnection(connStr);
```

```
conn.Open();
```

```
SqlCommand cmd = new SqlCommand();
```

```
cmd.Connection = conn;
```

2. Đối tượng Command

C2: Tạo đối tượng Command trực tiếp thông qua phương thức **CreateCommand** của đối tượng Connection

Ví dụ:

```
String connStr="DataSource=.; Database=HoaDon;  
                Integrated Security=true;";  
SqlConnection conn = new SqlConnection(connStr);  
conn.Open();  
SqlCommand cmd = conn.CreateCommand();
```

2. Đối tượng Command

C3: Tạo đối tượng Command từ phương *thức khởi tạo hai tham số* của lớp **SqlCommand** (constructor)

```
SqlCommand cmd=new SqlCommand(<lệnh SQL>, <connStr>)
```

Ví dụ:

```
string connStr="DataSource=.; Database=HoaDon;  
                Integrated Security=true;";  
string sql = "select*from MTUsers";  
SqlConnection conn = new SqlConnection(connStr);  
conn.Open();  
SqlCommand cmd=new SqlCommand(sql, conn);
```

2. Đối tượng Command

2.2 Một số thuộc tính thường dùng của Command

Sau khi tạo xong đối tượng cmd, cmd sẽ *có các thuộc tính sau*:

- A) cmd.**Connection**
- B) cmd.**CommandText**
- C) cmd.**CommandType**
- D) cmd.**CommandTimeout**
- E) cmd.**Parameters**

2. Đối tượng Command

B) **Connection**: Mỗi đối tượng **SqlCommand** phải đi kèm với 1 đối tượng **SqlConnection**.

Nếu ta *không khởi tạo thuộc tính này khi tạo đối tượng (thông qua constructor)*, thì *bắt buộc phải gán trực tiếp đối tượng SqlConnection vào đối tượng SqlCommand*.

Ví dụ: **string** strConn= //...chuỗi kết nối.....

```
SqlConnection conn=new SqlConnection(strConn);
```

```
SqlCommand cmd = new SqlCommand(conn);
```

hoặc

```
SqlCommand cmd = new SqlCommand();
```

```
cmd.Connection = conn;
```

2. Đối tượng Command

A) **CommandText** = *câu lệnh SQL*, *tên bảng* hoặc *tên stored procedure* muốn thực hiện trên nguồn dữ liệu.

Ví dụ:

```
SqlCommand cmd;  
cmd = new SqlCommand();  
cmd.CommandText = "select*from MTUsers";
```

```
SqlCommand cmd;  
cmd = new SqlCommand();  
string strSQL = "select*from MTUsers";  
cmd.CommandText = strSQL;
```

2. Đối tượng Command

C) **CommandType**=  **CommandType.Text** (mặc định)
CommandType.StoredProcedure
CommandType.TableDirect

Phụ thuộc vào **CommandText** mà xác định được ta cần thiết lập gì ở **CommandType**

2. Đối tượng Command

- **CommandType** = **CommandType.Text**;
Thì giá trị truyền vào cho **CommandText** là một câu lệnh SQL.

Ví dụ:

```
conn.open();
```

```
SqlCommand cmd = new SqlCommand();
```

```
cmd.CommandText = "select*from MTUsers";
```

```
cmd.CommandType = CommandType.Text;
```

```
cmd.Connection=conn; //conn là đối tượng SqlConnection
```

```
object KQ = cmd.ExecuteScalar(); //Lấy giá trị đầu tiên
```


2. Đối tượng Command

- **CommandType** = **CommandType.StoredProcedure**;
Thì giá trị truyền vào cho **CommandText** là một thủ tục (procedure) của SQL Server)

Ví dụ:

```
conn.open();
```

```
SqlCommand cmd = new SqlCommand();
```

```
cmd.CommandText = "SP_DisplayUser";
```

```
cmd.CommandType = CommandType.StoredProcedure;
```

```
cmd.Connection=conn; //conn là đối tượng SqlConnection
```

```
object KQ = cmd.ExecuteScalar(); //Lấy giá trị đầu tiên
```

2. Đối tượng Command

- `CommandType = CommandType.TableDirect;`

Thì giá trị truyền vào cho `CommandText` là một bảng

Ví dụ:

```
OleDbCommand cmd = new OleDbCommand();  
cmd.CommandText = "tblnhavien";  
cmd.CommandType = CommandType.TableDirect;  
cmd.Connection=conn; //conn là đối tượng SqlConnection
```

Note: đối tượng Command sử dụng thuộc tính `TableDirect` chỉ sử dụng được trong trường hợp `DataProvider` là `OleDb`.
Như vậy đối tượng Command phải thuộc lớp: `OleDbCommand`

2. Đối tượng Command

D) CommandTimeout=thời gian thực thi tính bằng giây (mặc định 30 giây).

Ví dụ: Giới hạn thời gian thực thi thủ tục trong vòng 60 giây

```
conn.Open();  
SqlCommand cmd = new SqlCommand();  
cmd.CommandText = "SP_DisplayUser";  
cmd.CommandType = CommandType.StoredProcedure;  
cmd.CommandTimeout = 60;  
cmd.Connection = conn;  
object KQ = cmd.ExecuteScalar();
```

2. Đối tượng Command

E) **Parameters** = Tập hợp *các tham số dùng trong Command*.

Thuộc tính Parameters là thuộc tính quan trọng của đối tượng Command.

2. Đối tượng Command

Xét ví dụ:

```
string sql= " select fullname  
            from mtusers  
            where fullname like '%Duy%'"
```

Nếu sử dụng câu truy vấn trên vào để truy xuất trong chương trình:

```
conn.Open();  
SqlCommand cmd= new SqlCommand();  
cmd.CommandText=sql;  
cmd.CommandType= CommandType.Text;  
cmd.Connection = conn;  
object KQ = cmd.ExecuteScalar();
```

2. Đối tượng Command

❑ Giải quyết vấn đề bằng cách truyền tham số vào câu lệnh truy vấn Sql

Ví dụ: Viết form có giao diện như hình gồm: 1 textbox để nhập dữ liệu tìm kiếm. Nhấn nút tìm kiếm có dữ liệu thì hiển thị kết quả là họ tên đầy đủ tìm được

The image shows a Windows application window titled "Form1". Inside the window, there is a text input field containing the text "Duy". To the right of the text field are two buttons: "Tìm kiếm" (Search) and "Thoát" (Exit). Below these elements is a label "Kết quả" (Result) and a text area displaying the text "Có tên Nguyễn Đức Duy trong CSDL". Two blue callout boxes with white text are present: one pointing to the text input field with the text "Textbox đặt tên txtTimKiem", and another pointing to the "Kết quả" label with the text "Label đặt tên lblKQ".

2. Đối tượng Command

```
SqlConnection conn;  
conn=new SqlConnection(" Data Source=.; Database=BanHang;  
                        Integrated Security=true ");  
conn.Open();  
string sql= @"select fullname from MtUsers  
            where fullname like N'%" + txtTimKiem.Text.Trim() + "%";  
SqlCommand cmd = new SqlCommand();  
cmd.Connection = conn;  
cmd.CommandText = sql;  
cmd.CommandType = CommandType.Text;  
object KQ = cmd.ExecuteScalar();  
if (KQ == null)  
    lblKQ.Text = "Không tìm thấy tên này";  
else  
    lblKQ.Text = "Có tên " + KQ.ToString() + " trong CSDL";
```

2. Đối tượng Command

❑ *Sử dụng thuộc tính Parameters:*

```
SqlConnection conn;  
conn=new SqlConnection(" Data Source=.; Database=BanHang;  
                        Integrated Security=true ");  
conn.Open();  
string sql= @"Select fullname  
             from MtUsers  
             where fullname like '%' + @TenNV + '%";  
SqlCommand cmd = new SqlCommand();  
cmd.Connection = conn;  
cmd.CommandText = sql;  
cmd.CommandType = CommandType.Text;  
cmd.Parameters.Add ("@ TenNV ", SqlDbType.NVarChar).Value =  
txtTimKiem.Text.Trim();  
object KQ = cmd.ExecuteScalar();
```


2. Đối tượng Command

2.3 Các phương thức thường dùng của đối tượng Command

Tạo đối tượng thuộc lớp SqlCommand:

```
SqlCommand cmd = new SqlCommand();
```

Sau khi tạo xong đối tượng cmd, cmd sẽ có các phương thức sau:

- ✓ cmd.**ExecuteScaler();**
- ✓ cmd.**ExecuteNonQuery();**
- ✓ cmd.**ExecuteReader();**
- ✓ cmd.**ExecuteXmlReader();**

2. Đối tượng Command

✓ ExecuteScaler():

Phương thức này thực hiện lệnh của Command và chỉ **trả về giá trị của cột đầu tiên và dòng đầu tiên** (ô đầu tiên trong bảng). Chúng ta thường sử dụng phương thức này khi muốn Command **thực hiện các hàm tính toán thống kê như SUM, COUNT, AVG, MAX, MIN,...** trên nguồn dữ liệu ngay lúc thực thi.

Ví dụ: `select count(*) from mtusers`

	(No column name)
1	8

2. Đối tượng Command

Ví dụ: Hiển thị số record trong bảng MTUsers

```
String connStr="DataSource=.; Database=BanHang;  
Integrated Security=true;";  
SqlConnection conn = new SqlConnection(connStr);  
conn.Open();  
SqlCommand cmd = new SqlCommand();  
cmd.Connection = conn;  
cmd.CommandText = "select count(*) from MTUsers ";  
object number=cmd.ExecuteScalar();  
MessageBox.Show("So record " + number.ToString());
```

2. Đối tượng Command

Ngoài ra cũng có thể sử dụng phương thức `ExecuteScalar()` để lấy giá trị của ô có vị trí (**dòng đầu tiên, cột chỉ định**)

Ví dụ: select `ten` from nhanvien

	ten
1	Bích
2	Phát
3	Oanh
4	Ngọc
5	Thắng

Sử dụng `ExecuteScalar()` sẽ lấy được giá trị “Bích”

2. Đối tượng Command

Ví dụ: Hiển thị giá trị tại record đầu tiên và ở cột Fullname trong bảng MTUsers:

```
String connStr="DataSource=.; Database=BanHang;  
                Integrated Security=true;";  
SqlConnection conn = new SqlConnection(connStr);  
conn.Open();  
SqlCommand cmd = new SqlCommand();  
cmd.Connection = conn;  
cmd.CommandText = " select fullname from MTUsers ";  
object number=cmd.ExecuteScalar();  
MessageBox.Show("Gia tri:" + number.ToString());
```

2. Đối tượng Command

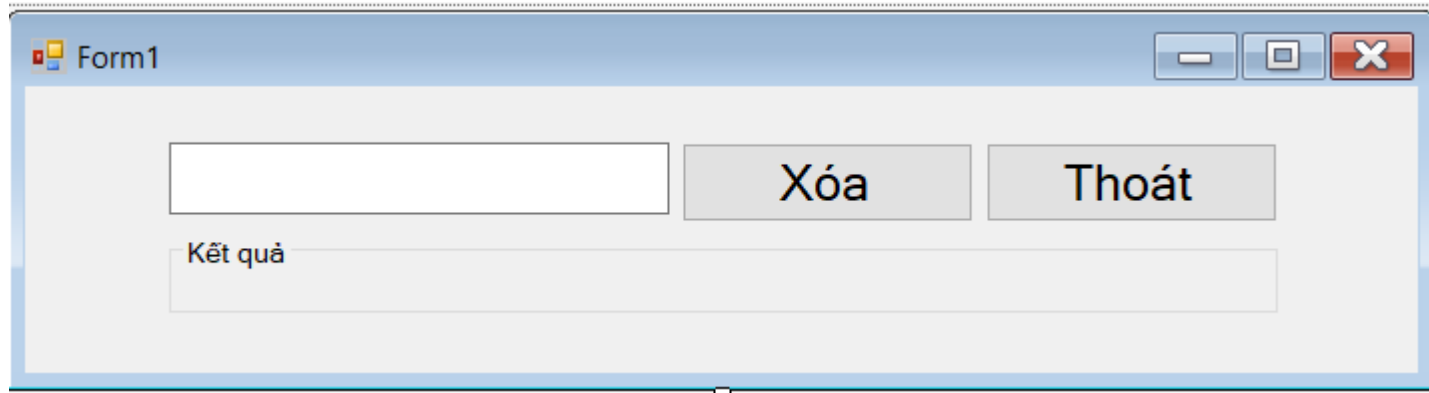
✓ ExecuteNonQuery():

Phương thức thực thi các phát biểu SQL (*delete, update, insert*) và các thủ tục của SQL Server. Phương thức này trả về số dòng bị tác động và sẽ trả về 0 nếu gặp lỗi.

	username	password	fullname	modifieddate	active	role
1	admin	admin@123	Quan tri vien	2021-01-01 00:00:00.000	1	A
2	duy.nguyennguc	duy@123	Nguyễn Đức Duy	2021-01-01 00:00:00.000	1	S
3	hoang.nguyenanh	hoang@123	Nguyễn Hoàng Anh	2021-02-02 00:00:00.000	1	S
4	huy.nguyenhoang	huy@123	Nguyễn Hoàng Huy	2021-01-01 00:00:00.000	1	S
5	nam.nguyenvan	nam@123	Nguyễn Văn Nam	2021-01-01 00:00:00.000	1	S
6	phuc.tranngoc	phuc@123	Trần Ngọc Phúc	2021-01-01 00:00:00.000	1	S
7	quang.lamxuan	quan@123	Lâm Xuân Quang	2021-01-01 00:00:00.000	1	S
8	tu.duong	tu@123	Dương Tử	2021-01-01 00:00:00.000	1	S

2. Đối tượng Command

Ví dụ: Xây dựng form như hình dưới, viết chương trình ***nhập vào username => nhấn nút Xóa, nếu tìm thấy username sẽ xóa dòng tìm thấy trong CSDL, không tìm thấy thì hiển thị “Không tìm thấy”***



The image shows a screenshot of a Windows application window titled "Form1". The window has a standard Windows XP-style title bar with minimize, maximize, and close buttons. Inside the window, there is a search interface. It consists of a text input field on the left, followed by two buttons labeled "Xóa" (Delete) and "Thoát" (Exit). Below the input field and buttons is a label "Kết quả" (Result) followed by a large, empty text area for displaying the search results.

2. Đối tượng Command

```
string strconn = "Data Source=.;Initial Catalog=BanHang;  
                Integrated Security=True";  
conn = new SqlConnection(strconn);  
conn.Open();  
SqlCommand cmd = conn.CreateCommand();  
cmd.CommandText = @"delete from mtusers  
                  where username = @username";  
cmd.CommandType = CommandType.Text;  
cmd.Parameters.Add("@username", SqlDbType.NVarChar).Value =  
                txtTimKiem.Text.Trim();  
int KQ = cmd.ExecuteNonQuery();  
if (KQ == 0)  
    lblKQ.Text = "Không tìm thấy tên này";  
else  
    lblKQ.Text = "Đã xóa " + txtTimKiem.Text + " trong CSDL";
```


2. Đối tượng Command

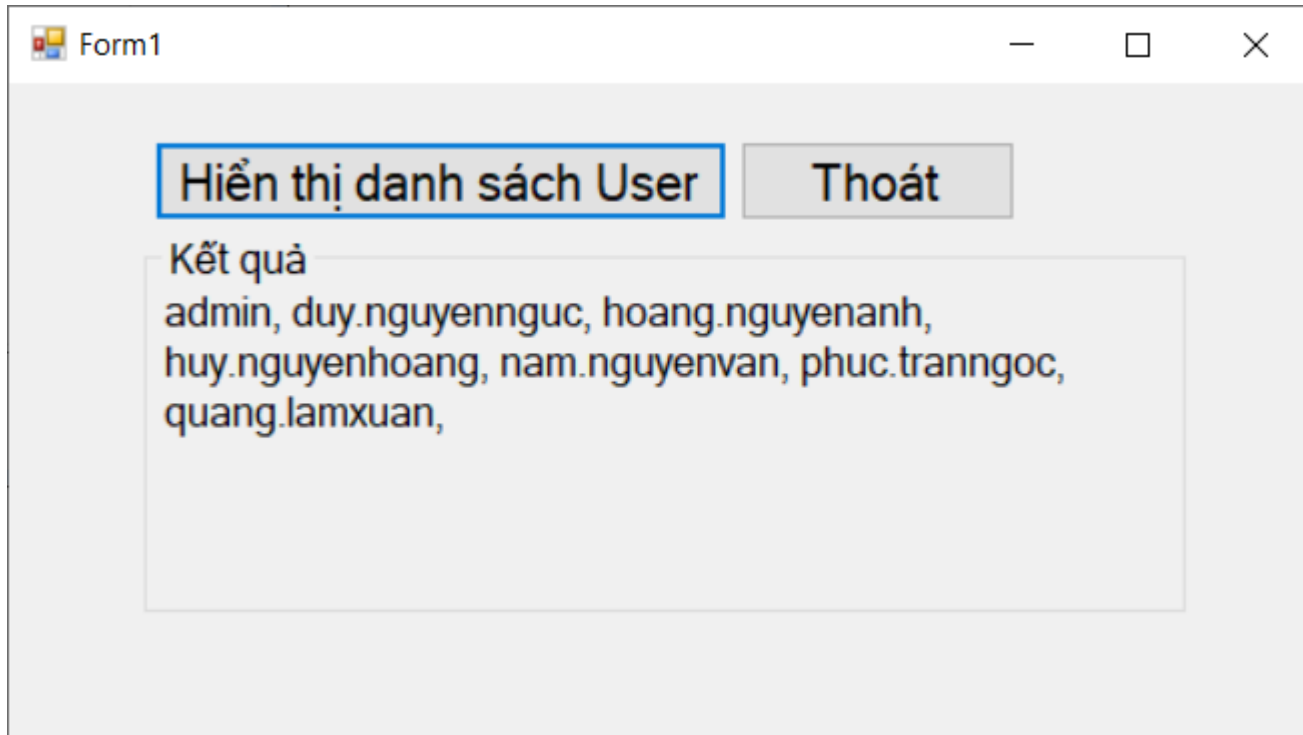
✓ ExecuteReader():

Phương thức ExecuteReader trả về hàng dữ liệu, và sử dụng đối tượng SqlDataReader để nắm hàng dữ liệu đó

	username	password	fullname	modifieddate	active	role
1	admin	admin@123	Quan tri vien	2021-01-01 00:00:00.000	1	A
2	duy.nguyennguc	duy@123	Nguyễn Đức Duy	2021-01-01 00:00:00.000	1	S
3	hoang.nguyenanh	hoang@123	Nguyễn Hoàng Anh	2021-02-02 00:00:00.000	1	S
4	huy.nguyenhoang	huy@123	Nguyễn Hoàng Huy	2021-01-01 00:00:00.000	1	S
5	nam.nguyenvan	nam@123	Nguyễn Văn Nam	2021-01-01 00:00:00.000	1	S
6	phuc.tranngoc	phuc@123	Trần Ngọc Phúc	2021-01-01 00:00:00.000	1	S
7	quang.lamxuan	quan@123	Lâm Xuân Quang	2021-01-01 00:00:00.000	1	S
8	tu.duong	tu@123	Dương Tử	2021-01-01 00:00:00.000	1	S

2. Đối tượng Command

Ví dụ: Xây dựng form như hình dưới, Khi nhấn nút “”Hiển thị danh sách User”” thì sẽ hiển thị được danh sách các username trong bảng MTUsers



The screenshot shows a Windows application window titled "Form1". Inside the window, there are two buttons at the top: "Hiển thị danh sách User" (highlighted with a blue border) and "Thoát". Below these buttons is a text box containing the following text:

Kết quả
admin, duy.nguyennguc, hoang.nguyenanh,
huy.nguyenhoang, nam.nguyenvan, phuc.tranngoc,
quang.lamxuan,

2. Đối tượng Command

Ví dụ:

```
String connStr="DataSource=.; Database=BanHang;  
                Integrated Security=true;";  
SqlConnection conn = new SqlConnection(connStr);  
conn.Open();  
SqlCommand cmd = new SqlCommand();  
cmd.Connection = conn;  
cmd.CommandText = "select * from MTUsers ";  
SqlDataReader Reader = cmd.ExecuteReader();  
string userlist = "";  
while (Reader.Read())  
{  
    userlist = userlist + Reader[0].ToString() +", ";  
}  
Reader.Close(); //lưu ý phải Close() SqlDataReader
```

2. Đối tượng Command

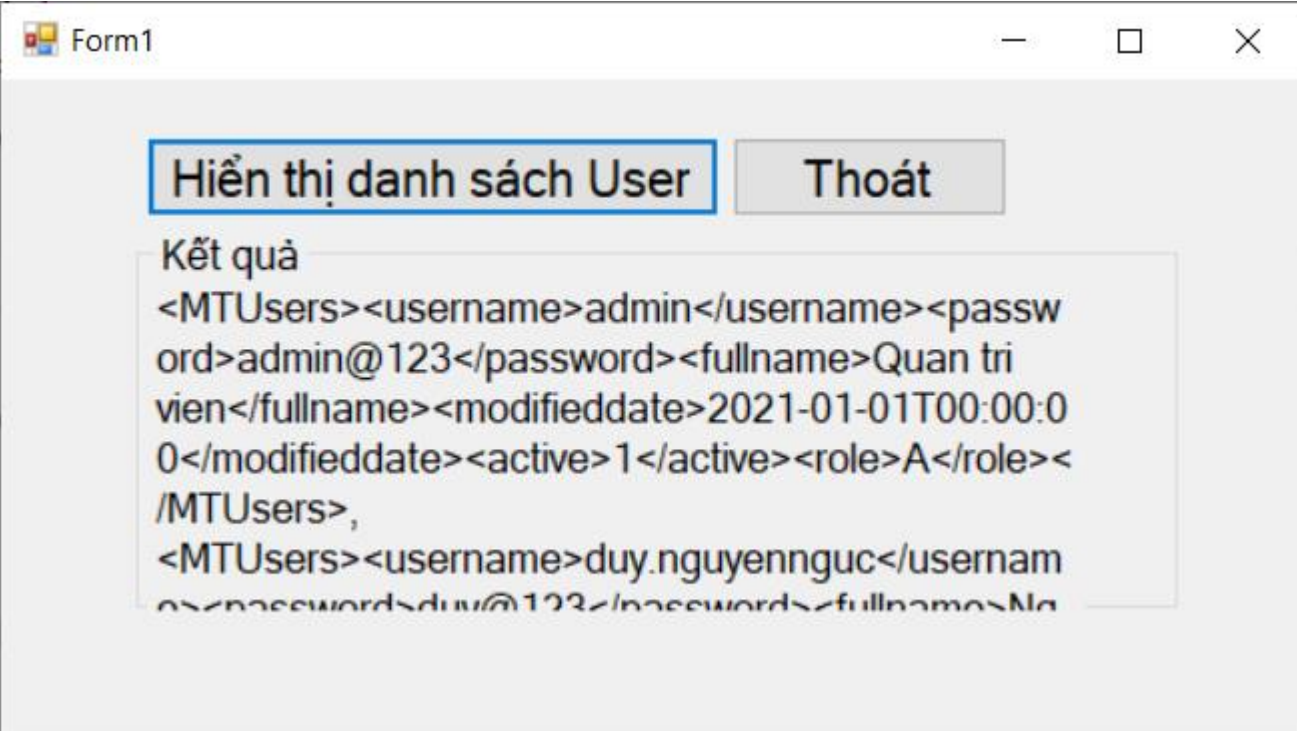
✓ ExecuteXmlReader():

Tương tự phương thức ExecuteReader nhưng giá trị trả về là tập dữ liệu định dạng XML

	username	password	fullname	modifieddate	active	role
1	admin	admin@123	Quan tri vien	2021-01-01 00:00:00.000	1	A
2	duy.nguyennguc	duy@123	Nguyễn Đức Duy	2021-01-01 00:00:00.000	1	S
3	hoang.nguyenanh	hoang@123	Nguyễn Hoàng Anh	2021-02-02 00:00:00.000	1	S
4	huy.nguyenhoang	huy@123	Nguyễn Hoàng Huy	2021-01-01 00:00:00.000	1	S
5	nam.nguyenvan	nam@123	Nguyễn Văn Nam	2021-01-01 00:00:00.000	1	S
6	phuc.tranngoc	phuc@123	Trần Ngọc Phúc	2021-01-01 00:00:00.000	1	S
7	quang.lamxuan	quan@123	Lâm Xuân Quang	2021-01-01 00:00:00.000	1	S
8	tu.duong	tu@123	Dương Tử	2021-01-01 00:00:00.000	1	S

2. Đối tượng Command

Ví dụ: Xây dựng form như hình dưới, Khi nhấn nút “”Hiển thị danh sách User”” thì sẽ hiển thị được danh sách các username trong bảng MTUsers dưới dạng XML



The screenshot shows a Windows application window titled "Form1". Inside the window, there are two buttons at the top: "Hiển thị danh sách User" (highlighted with a blue border) and "Thoát". Below these buttons is a text area labeled "Kết quả" (Result). The text area contains XML data representing user information from a table named MTUsers.

```
<MTUsers><username>admin</username><password>admin@123</password><fullname>Quan tri vien</fullname><modifieddate>2021-01-01T00:00:00</modifieddate><active>1</active><role>A</role></MTUsers>,<MTUsers><username>duy.nguyennguc</username><password>duy@123</password><fullname>Ng
```

2. Đối tượng Command

```
String connStr="DataSource=.; Database=BanHang;  
                Integrated Security=true;";  
SqlConnection conn = new SqlConnection(connStr);  
conn.Open();  
SqlCommand cmd = new SqlCommand();  
cmd.Connection = conn;  
cmd.CommandText = @"select * from MTUsers  
                    FOR XML AUTO, elements";  
XmlReader reader=cmd.ExecuteXmlReader();  
string str = "";  
while (!reader.EOF)  
{  
    str += reader.ReadOuterXml() + ", ";  
};  
reader.Close();  
lblKQ.Text = str;
```

3. Đối tượng SqlParameter

3. Đối tượng SqlParameter

Parameter là gì ?

Và tại sao phải sử dụng parameter ?

3. Đối tượng SqlParameter

- Parameter là các tham số trong câu lệnh *Sql* hoặc thủ tục *Sql*.
- Do đó có thể sử dụng đối tượng **SqlParameter** để tạo ra các tham số và truyền vào giá trị cho các tham số trong các câu truy vấn *Sql* hoặc cho các thủ tục *Sql*.

2. Đối tượng Command

❑ *Chương trình tìm kiếm nhân viên theo tên*

```
SqlConnection conn;  
conn=new SqlConnection(" Data Source=.; Database=BanHang;  
                        Integrated Security=true ");  
conn.Open();  
string sql= @"Select fullname from MtUsers  
            where fullname like '%' + @TenNV + '%";  
SqlCommand cmd = new SqlCommand();  
cmd.Connection = conn;  
cmd.CommandText = sql;  
cmd.CommandType = CommandType.Text;  
cmd.Parameters.Add ("@TenNV ", SqlDbType.NVarChar).Value =  
                                txtTimKiem.Text.Trim();  
object KQ = cmd.ExecuteScalar();
```

2. Đối tượng Command

❑ *Chương trình tìm kiếm nhân viên theo tên*

```
SqlConnection conn;  
conn=new SqlConnection(" Data Source=.; Database=BanHang;  
                        Integrated Security=true ");  
conn.Open();  
string sql= @"Select fullname from MtUsers  
            where fullname like '%' + @TenNV + '%";  
SqlCommand cmd = new SqlCommand();  
cmd.Connection = conn;  
cmd.CommandText = sql;  
cmd.CommandType = CommandType.Text;  
SqlParameter parameter = new SqlParameter();  
parameter.ParameterName = "@TenNV";  
Parameter.SqlValue = txtTimKiem.Text.Trim();  
cmd.Parameters.Add (Parameter);
```

3. Đối tượng SqlParameter

❖ Truyền tham số theo câu truy vấn

```
// Khai báo ra parameter
```

```
string sql= "select *from nhanvien where MaNV = @MS";
```

```
SqlCommand cmd = new SqlCommand( sql, conn);
```

```
// định nghĩa ra parameter
```

```
SqlParameter param = new SqlParameter();
```

```
param.ParameterName = "@MS";
```

```
param.SqlValue = 11 // đưa giá trị vào biến @city
```

```
cmd.Parameters.Add(param);
```

3. Đối tượng SqlParameter

❖ Truyền tham số theo Store Procedure

//Tạo procedure trong SQL Server

```
CREATE PROC SP_DisplayUserbyUserName @Username varchar(20)
```

As

```
SELECT * FROM MTUsers WHERE username=@Username;
```

// Viết code trong C#

```
SqlCommand cmd = new SqlCommand();
```

```
cmd.Connection = conn;
```

```
cmd.CommandText = "SP_DisplayUserbyUserName";
```

```
cmd.CommandType = CommandType.StoredProcedure;
```

```
SqlParameter para = new SqlParameter();
```

```
para.ParameterName = "@Username";
```

```
para.SqlValue = "admin";
```

```
cmd.Parameters.Add(para);
```

3. Đối tượng SqlParameter

Dùng Parameter có rất nhiều điều lợi:

- ❖ Không cần lo lắng type của tham số có hợp hay không
- ❖ Không sợ SQL-Injection

3. Đối tượng SqlParameter

❖ Không cần lo lắng type của tham số có hợp hay không

Thí dụ:

```
SELECT * FROM tblTable
WHERE
    (ColA BETWEEN 01.01.2009 AND 01.04.2009)
AND
    ColB = 'whatever'
```

ColA là DateTime, khi đưa tham số vào ta phải chú ý định dạng hợp lệ

ColB là varchar, khi đưa vào ta phải nhớ bọc tham số lại với '

3. Đối tượng SqlParameter

Nếu dùng Parameters không cần phải chú ý điểm này:

```
string sql = "SELECT * FROM tblTable  
            WHERE (ColA BETWEEN @dt1 AND @dt2)  
            AND ColB = @name";
```


3. Đối tượng SqlParameter

❖ Không sợ SQL - Injection

SQL-Injection là gì ???

3. Đối tượng SqlParameter

-SQL injection là **một kỹ thuật** cho phép những kẻ tấn công **lợi dụng lỗ hổng** của **việc kiểm tra dữ liệu đầu vào trong các ứng dụng** và **các thông báo lỗi** của hệ quản trị cơ sở dữ liệu trả về **để inject (tiêm vào) và thi hành các câu lệnh SQL bất hợp pháp**

- Sql injection có thể cho phép những kẻ tấn công thực hiện các thao tác, delete, insert, update,... trên cơ sở dữ liệu của ứng dụng

3. Đối tượng SqlParameter

- Ví dụ:

```
string sql= "SELECT* FROM data WHERE id = " + textbox.Text
```

Nhận thấy câu lệnh SQL trên sẽ hiển thị dữ liệu có id bằng với giá trị trong biến (là 1 số). Tuy nhiên thanh thì nhập 1 số, thì người dùng có thể nhập vào textbox 1 chuỗi như sau:

1; DROP TABLE user

Như vậy câu lệnh SQL hoàn chỉnh như sau:

```
SELECT * FROM data WHERE id=1; DROP TABLE user
```

3. Đối tượng SqlParameter

Do đó *thay vì truyền thẳng textbox.text vào câu lệnh truy vấn SQL, thì ta có thể sử dụng tham số* như sau:

```
string sql= "SELECT* FROM data WHERE id = @id"  
SqlParameter para=new SqlParameter();  
para.ParameterName = "@id";  
para.SqlValue = textbox.Text;  
cmd.Parameters.Add(para);
```

Và lúc này ta **có thể truyền giá trị cho @id thông qua Parameter**. Vì vậy việc *hack database cũng khó mà thành công, bởi vì dữ liệu đầu vào đã qua hệ thống kiểm tra của SqlCommand.Parameters* và sẽ thấy exception ra nếu data không hợp lệ

QUESTION ???