CHƯƠNG 3.
KẾT NỐI CƠ SỞ DỮ LIỆU

Khoa Công nghệ Thông tin
Trường Đại học Đà Lạt

MỤC TIÊU

- Giúp sinh viên nắm được:
  - Kiến thức cơ sở về các lớp của ADO.Net
  - Cách viết một chương trình đơn giản có kết nối và đọc các dòng từ cơ sở dữ liệu.
  - Cách xây dựng ứng dụng Windows Form
  - Cách tạo kết nối đến cơ sở dữ liệu
    - SQL Server
    - Oracle
    - MS Access/ Excel
  - Cách quản lý trạng thái của kết nối
  - Cấu hình thông tin kết nối

NỘI DUNG

- ADO.Net là gì?
- Phân loại các lớp của ADO.Net
- Các lớp trong nhóm kết nối
  - Connection, Command
  - Parameter, ParameterCollection
  - DataReader, DataAdapter
  - …
- Các lớp trong nhóm không kết nối
  - DataSet, DataTable, DataView
  - DataColumn, DataRow
  - DataRelation, Constraint

3

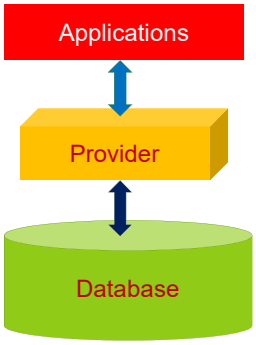
ADO.NET LÀ GÌ?

- ADO.NET là một công nghệ - công cụ có thể truy xuất dữ liệu theo 2 phương thức:
  - Kết nối: Tương tác trực tiếp với một cơ sở dữ liệu thông qua đối tượng thuộc các lớp kết nối (hay trình điều khiển được quản lý - managed provider).
  - Không kết nối: thông tin từ cơ sở dữ liệu được lưu trực tiếp trong bộ nhớ máy tính mà chương trình đang chạy
- Các đối tượng này cho phép kết nối tới cơ sở dữ liệu và thực thi các lệnh SQL khi luồng kết nối được mở.
- ADO.Net phân phối dữ liệu dựa trên chuẩn XML. Vì thế, nó có thể truyền dữ liệu từ nơi này sang nơi khác qua Internet và có thể xuyên qua các tường lửa.
- Mục đích của ADO.Net là hỗ trợ xây dựng các ứng dụng N-Tier hoặc có thể chạy trên nhiều server.

4

DATA PROVIDER

- Data Provider (hay Driver – trình điều khiển) là một thuật ngữ ám chỉ một thư viện nhị phân - được viết dưới dạng các hàm API (giao diện lập trình ứng dụng).



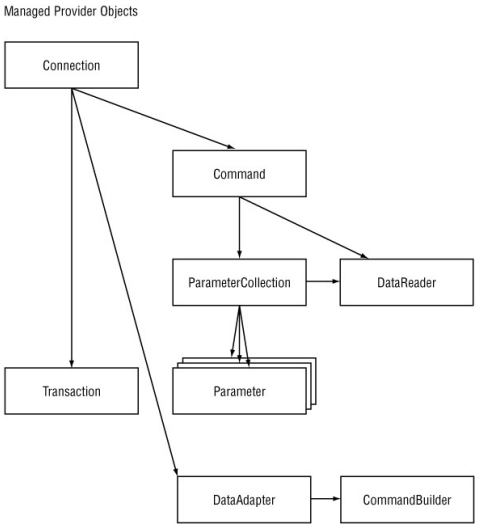
5

PHÂN LOẠI CÁC LỚP

- Các lớp kết nối (managed provider)
  - Tạo kết nối trực tiếp
  - Đồng bộ hóa dữ liệu được lưu trên máy cục bộ với csdl.
  - Đọc các dòng dữ liệu từ cơ sở dữ liệu theo một hướng.
  - Sử dụng tùy thuộc vào loại cơ sở dữ liệu muốn kết nối tới.
- Các lớp không kết nối (generic data)
  - Lưu trữ bản sao thông tin lấy được từ cơ sở dữ liệu.
  - Không phụ thuộc vào loại cơ sở dữ liệu mà bạn dùng, nghĩa là các lớp này được dùng chung cho dữ liệu được lấy từ cơ sở dữ liệu SQL Server, Access hay Oracle…
  - Các lớp này biểu diễn thông tin lấy từ cơ sở dữ liệu ở định dạng XML.

6

NHÓM LỚP KẾT NỐI



7

NHÓM LỚP KẾT NỐI

- SQL Server Managed Provider Classes: được dùng để kết nối tới một cơ sở dữ liệu SQL Server.
- OLE DB Managed Provider Classes: được dùng để kết nối tới bất kỳ cơ sở dữ liệu nào hỗ trợ chuẩn OLE DB (*Object Linking and Embedding for Databases*) như Access hay Oracle.
- ODBC Managed Provider Classes: được dùng để kết nối tới các cơ sở dữ liệu có hỗ trợ chuẩn ODBC (*Open Database Connectivity*).
  - Tốc độ truy xuất chậm → chỉ nên dùng nhóm lớp này khi không còn cách nào khác để thay thế nhóm OLE DB.

8

NHÓM LỚP KẾT NỐI

- Tất cả các nhóm lớp kết nối này đều được xây dựng cùng một tập chức năng, nghĩa là có cùng các hàm xử lý.
- Tên của các chuẩn này được đặt trước tên của các lớp kết nối để cho biết nhóm lớp nào đang được dùng.
- Chẳng hạn, nếu tên lớp kết nối bắt đầu bằng ***Sql*** thì cơ sở dữ liệu đang dùng được quản lý bởi SQL Server.
- Vì thế, để đơn giản, ta chỉ gọi tên chung của chúng. Chẳng hạn, lớp Connection, lớp Command…

9

LỚP CONNECTION

- Muốn giao tiếp với một cơ sở dữ liệu, trước hết phải thực hiện một kết nối tới cơ sở dữ liệu thông qua lớp Connection.
- Có 3 lớp kết nối:
  - SqlConnection
  - OledbConnection
  - OdbcConnection
- Mở kết nối trước khi thực thi lệnh bằng phương thức Open() và đóng kết nối bằng phương thức Close()

10

LỚP COMMAND

- Đối tượng Command được dùng để
  - thực thi một lệnh SQL như SELECT, INSERT, UPDATE hay DELETE.
  - Gọi một thủ tục hay lấy các dòng dữ liệu từ một bảng cụ thể.
- Đối tượng Command sử dụng một đối tượng Connection và yêu cầu kết nối này phải được mở trước khi thực thi truy vấn.
- Có 3 lớp Command
  - SqlCommand
  - OledbCommand
  - OdbcCommand

11

LỚP PARAMETER

- Đối tượng Parameter được dùng để gửi một tham số vào trong đối tượng *Command*.
- Nó được dùng để lưu giá trị tham số được gửi vào trong một thủ tục (Stored Procedure) hay một lệnh SQL.
- Khi có nhiều tham số, chúng được lưu và truyền vào đối tượng Command thông qua đối tượng ParameterCollection.
- Có 3 lớp tham số:
  - SqlParameter
  - OledbParameter
  - OdbcParameter

12

LỚP DATAREADER

- Đối tượng thuộc lớp này này dùng để dọc các dòng dữ lấy được từ cơ sở dữ liệu bởi đối tượng Command.
- Đối tượng DataReader chỉ được dùng để đọc dữ liệu theo một chiều từ đầu tới cuối và không thể cập nhật dữ liệu cho các dòng trong cơ sở dữ liệu.
- DataReader có thể dùng để thay thế cho DataSet và việc đọc dữ liệu từ DataReader thường nhanh hơn đọc dữ liệu từ DataSet.
- Có 3 lớp DataReader:
  - SqlDataReader
  - OleDbDataReader
  - OdbcDataReader

13

LỚP DATAADAPTER

- Đối tượng DataAdapter được dùng để
  - chuyển các dòng trong một cơ sở dữ liệu vào một DataSet
  - đồng bộ dữ liệu đã thay đổi ở máy cục bộ tới cơ sở dữ liệu.
  - Việc đồng bộ được thực hiện qua một đối tượng Connection.
- Có thể đọc các dòng trong cơ sở dữ liệu vào một DataSet qua một DataAdapter, thay đổi dữ liệu đó. Sau đó, cập nhật các thay đổi lên cơ sở dữ liệu thông qua đối tượng Connection.
- Có 3 lớp DataAdapter:
  - SqlDataAdapter
  - OleDbDataAdapter
  - OdbcDataAdapter

14

LỚP COMMANDBUILDER VÀ TRANSACTION

- Có 3 lớp CommandBuilder:
  - SqlCommandBuilder, OledbCommandBuilder, OdbcCommandBuilder
- Đối tượng CommandBuilder tự động sinh ra các lệnh INSERT, UPDATE, DELETE trên một bảng và đồng bộ dữ liệu bị thay đổi trong DataSet lên cơ sở dữ liệu.
- Đối tượng Transaction được dùng để biểu diễn một giao dịch cơ sở dữ liệu (database transaction).
- Một giao dịch cơ sở dữ liệu là một nhóm các lệnh làm thay đổi các dòng trong cơ sở dữ liệu.
- Các lệnh lệnh này được xem như một đơn vị công việc về mặt logic.
- Có 3 lớp giao dịch (*transaction*):
  - SqlTransaction, OledbTransaction và OdbcTransaction.

15

NAMESPACE CHỨA CÁC LỚP KẾT NỐI

- Các lớp kết nối sử dụng cho SQL Server được khai báo trong namespace *System.Data.SqlClient*.
- Các lớp kết nối cho cơ sở dữ liệu hỗ trợ OLE DB được khai báo trong namespace *System.Data.OleDb*.
- Các lớp kết nối cho cơ sở dữ liệu hỗ trợ ODBC được khai báo trong namespace *System.Data.Odbc*.

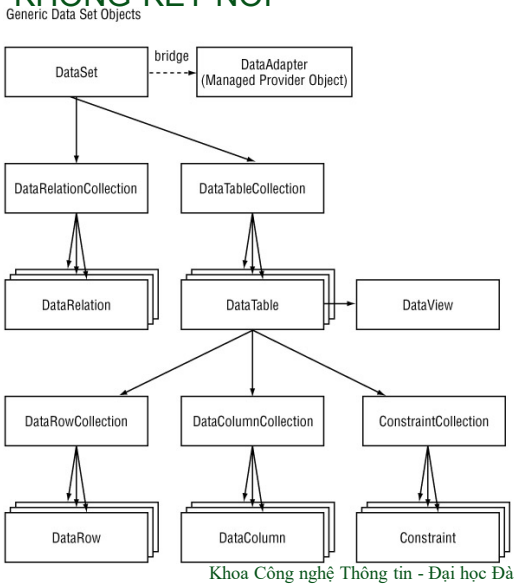
16

CÁC LỚP KHÔNG KẾT NỐI

- Các đối tượng không kết nối (hay các đối tượng chứa dữ liệu) cho phép
  - Lưu trữ một bản sao thông tin lấy từ cơ sở dữ liệu.
  - Khi đã ngắt kết nối tới cơ sở dữ liệu.
  - Đọc các dòng theo thứ tự bất kỳ
  - Tìm kiếm, sắp xếp hay trích lọc các dòng một cách linh hoạt.
  - Tạo ra các thay đổi trên dữ liệu, sau đó đồng bộ (cập nhật) các thay đổi này vào cơ sở dữ liệu.
- Cầu nối giữa các lớp kết nối và các lớp không kết nối là DataAdapter.

17

CÁC LỚP KHÔNG KẾT NỐI



18

LỚP DATASET

- Đối tượng của lớp DataSet được dùng để biểu diễn một bản sao thông tin được lưu trong cơ sở dữ liệu.
- Một DataSet có thể biểu diễn các cấu trúc như một cơ sở dữ liệu bao gồm các bảng, các dòng và cột, các ràng buộc như tính duy nhất (***unique***) hoặc rằng buộc về khóa ngoại.
- Đối tượng DataSet cũng được dùng để chứa dữ liệu có định dạng XML.

19

LỚP DATATABLE

- Đối tượng của lớp *DataTable* dùng để biểu diễn một bảng.
- Một *DataSet* có thể chứa nhiều bảng hay nhiều đối tượng *DataTable*.
- Những bảng này được truy xuất qua thuộc tính ***Tables*** của lớp DataSet. Thuộc tính này có kiểu ***DataTableCollection***.
- Một *DataTable* có thể chứa nhiều dòng, nhiều cột (tương ứng là DataRow và DataColumn).

20

LỚP DATACOLUMN VÀ DATAROW

- Đối tượng có kiểu *DataRow* dùng để biểu diễn một hàng (một dòng trong DataTable).
- Đối tượng thuộc lớp *DataColumn* được dùng để biểu diễn một cột của bảng.
- Một *DataTable* có thể chứa nhiều *DataRow* và được truy xuất thông qua thuộc tính *Rows* của lớp DataTable. Thuộc tính này có kiểu *DataRowCollection*.
- Một *DataTable* cũng có thể chứa nhiều *DataColumn* và được truy xuất thông qua thuộc tính *Columns* của lớp DataTable. Thuộc tính này có kiểu *DataColumnCollection*.

21

LỚP CONSTRAINT

- Đối tượng thuộc kiểu *Constraint* dùng để biểu diễn một ràng buộc cơ sở dữ liệu áp dụng cho một hoặc nhiều đối tượng *DataColumn* của một bảng (*DataTable*).
- Một DataTable có thể lưu trữ nhiều đối tượng *Constraint* và được truy xuất thông qua thuộc tính *Constraints* của lớp *DataTable*. Thuộc tính này có kiểu là *ConstraintCollection*.
- Có 2 loại ràng buộc:
  - UniqueConstraint:
  - ForeignKeyConstraint:

22

LỚP DATAVIEW VÀ DATARELATION

- Đối tượng có kiểu *DataView* được dùng để xem nội dung dữ liệu của các dòng trong một *DataTable* bằng cách dùng một bộ lọc với điều kiện nào đó.
- Đối tượng *DataRelation* dùng để biểu diễn một quan hệ giữa hai đối tượng *DataTable*.
- Có thể dùng một đối tượng *DataRelation* làm quan hệ cha-con giữa hai bảng như trong cơ sở dữ liệu.
- Một *DataSet* có thể chứa nhiều đối tượng *DataRelation*. Các quan hệ này được truy xuất qua thuộc tính *Relations* của lớp *DataSet*. Thuộc tính này có kiểu *DataRelationCollection*.

23

NAMESPACE CHỨA CÁC LỚP KHÔNG KẾT NỐI

- Các lớp *DataSet, DataTable, DataRow, DataColumn, DataRelation, Constraint* và *DataView* được khai báo trong namespace *System.Data*.

24

CÁC BƯỚC THỰC HIỆN MỘT TRUY VẤN TRONG ADO.NET

- Tạo chuỗi kết nối
- Tạo đối tượng kết nối tới cơ sở dữ liệu (Connection)
- Tạo lệnh truy vấn
- Tạo đối tượng thực thi lệnh (Command)
- Thiết lập lệnh truy vấn
- Tạo đối tượng điều phối dữ liệu (DataAdapter)
- Tạo đối tượng lưu trữ (DataSet / DataTable)
- Mở kết nối tới cơ sở dữ liệu
- Gọi phương thức lấy dữ liệu
- Đóng kết nối
- Lấy dữ liệu từ DataSet / DataTable
- Hiển thị dữ liệu

25

DEMO

- Console Application
- Windows Application

26

KẾT NỐI CƠ SỞ DỮ LIỆU

- Cách kết nối cơ sở dữ liệu
  - SQL Server
  - Access/ Excel
  - Oracle
- Kết nối dùng chung (Connection Pooling)
- Quản lý trạng thái kết nối
- Quản lý các sự kiện kết nối
  - StateChanged
  - InfoMessage
- Tạo & cấu hình đối tượng Connection bằng giao diện

27

LỚP SQLCONNECTION

- Đối tượng thuộc lớp này quản lý và điều khiển việc liên lạc giữa cơ sở dữ liệu và chương trình ứng dụng.
- Lớp SqlConnection chỉ được dùng cho SQL Server.
- Hầu hết các thuộc tính, phương thức và sự kiện trong lớp này cũng giống như trong các lớp OledbConnection và OdbcConnection.

28

LỚP SQLCONNECTION

- Các thuộc tính
  - String ConnectionString { get; set; }
  - Int ConnectionTimeout { get; }
  - String Database { get; }
  - String DataSource { get; }
  - Int PacketSize { get; }
  - String ServerVersion { get; }
  - ConnectionState State { get; }
    - Broken, Closed, Connecting, Executing, Fetching hay Open
  - String WorkstationId { get; }

29

LỚP SQLCONNECTION

- Phương thức
  - SqlTransaction BeginTransaction()
  - Void ChangeDatabase()
  - Void Close()
  - Void Open()
  - SqlCommand CreateCommand()
- Sự kiện
  - StateChangeEventHandler StateChange
  - SqlInfoMessageEventHandler InfoMessage

30

TẠO ĐỐI TƯỢNG KẾT NỐI

- Dùng phương thức tạo lập
  - SqlConnection ()
  - SqlConnection (string connectionString)

31

- Các thuộc tính của chuỗi kết nối

Từ khóa	Ý nghĩa
Provider Driver	Trình cung cấp hoặc trình điều khiển cách xử lý truy cập dữ liệu.
DataSource Server Hostname	Tên hoặc địa chỉ của máy chủ chứa cơ sở dữ liệu.
Initial Catalog Database	Tên cơ sở dữ liệu hoặc đường dẫn đến file dữ liệu.
User Id UID	Tên tài khoản đăng nhập vào cơ sở dữ liệu.
Password PWD	Mật khẩu tương ứng với tên tài khoản đăng nhập vào cơ sở dữ liệu.

KẾT NỐI CƠ SỞ DỮ LIỆU SQL SERVER

- Cách 1:

```
SqlConnection mySqlConnection = new SqlConnection();
mySqlConnection.ConnectionString =
    "server=localhost;database=Northwind;uid=sa;pwd=sa";
```

32

- Cách 2:

```
string connectionString = "server=localhost;database=Northwind;uid=sa;pwd=sa";
SqlConnection mySqlConnection = SqlConnection (connectionString);
```

- Cách 3:

```
SqlConnection mySqlConnection = new SqlConnection
    ("server=localhost;database=Northwind;uid=sa;pwd=sa");
```

- Cách 4:

```
SqlConnection mySqlConnection = new SqlConnection
("server=localhost;database=Northwind;integrated security=SSPI;");
```

KẾT NỐI CƠ SỞ DỮ LIỆU ACCESS

- Cú pháp của chuỗi kết nối
  - provider= **Microsoft.ACE.OLEDB.12.0**; data source=*databaseFile*

33

- Ví dụ:

```
connectionString = "Provider=Microsoft.ACE.OLEDB.12.0;" +
"data source= D:\\Database\\RestaurantManagement.accdb;”

myOleDbConnection =  new OleDbConnection (connectionString);
```

KẾT NỐI CƠ SỞ DỮ LIỆU ORACLE

- Cú pháp của chuỗi kết nối
  - provider=MSDAORA;data source=*OracleNetServiceName*;user id=*username*; password=*password*

34

- Ví dụ

```
string connectionString =
  "provider=MSDAORA;data source=ORCL;user id=SCOTT;password=TIGER";

OleDbConnection myOleDbConnection =
  new OleDbConnection (connectionString);
```

MỞ VÀ ĐÓNG KẾT NỐI

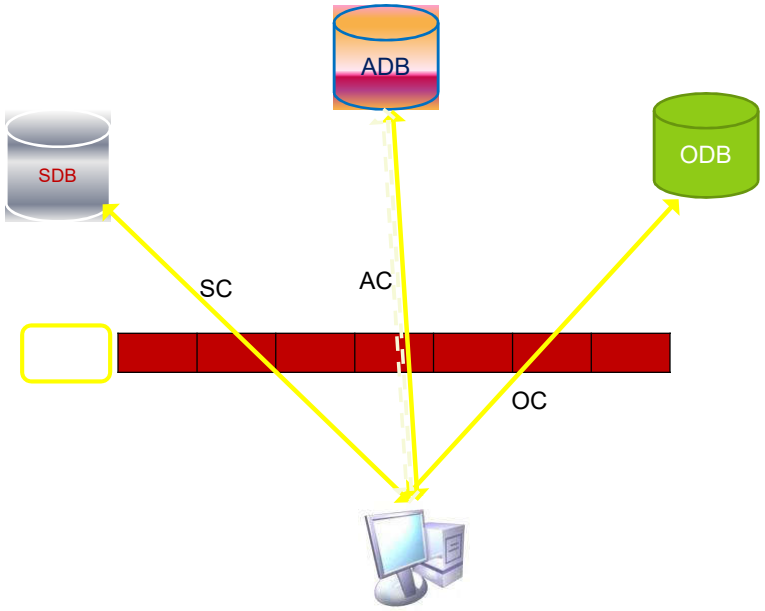
- Trước khi thực thi lệnh, phải mở kết nối bằng cách gọi hàm Open của đối tượng Connection
  - myConnection.Open()
- Sau khi thực thi lệnh, phải đóng kết nối bằng cách gọi hàm Close của đối tượng Connection
  - myConnection.Close()
- Lưu ý:
  - Khi sử dụng phương thức Fill của DataAdapter, không cần mở và đóng kết nối

35

KẾT NỐI DÙNG CHUNG

- Nhận xét
  - Việc mở và đóng kết nối tốn nhiều thời gian
- Giải pháp
  - Dùng tổ hợp kết nối (hay kết nối dùng chung)
- Đặc điểm
  - Kết nối không bị đóng hẳn khi gọi phương thức Close mà chuyển sang trạng thái chờ (not used)
  - Cải thiện thời gian một cách đáng kể khi thiết lập kết nối cơ sở dữ liệu → tăng hiệu suất

36



37

THIẾT LẬP KẾT NỐI DÙNG CHUNG

- Max Pool Size (mặc định là 100)
- Min Pool Size (mặc định là 0)

38

- Nếu cố gắng mở thêm một đối tượng SqlConnection mới trong khi tổ hợp đã đầy, yêu cầu mở sẽ phải chờ cho đến khi có một kết nối cũ bị đóng lại.
- Nếu thời gian chờ lớn hơn số giây được quy định trong thuộc tính ConnectionTimeout, một lỗi sẽ phát sinh.
- Ví dụ

```
SqlConnection mySqlConnection = new SqlConnection(
"server=localhost;database=Northwind;uid=sa;pwd=sa;" +
"max pool size=10;min pool size=5");
```

QUẢN LÝ TRẠNG THÁI KẾT NỐI

- Trạng thái của kết nối cho biết tình trạng của quá trình yêu cầu kết nối tới cơ sở dữ liệu.
- Để biết trạng thái hiện tại của một kết nối, ta dùng thuộc tính ***State*** của đối đối tượng SqlConnection
- Nên kiểm tra trạng thái trước khi gọi hàm Open để mở kết nối

39

CONSTANT NAME	DESCRIPTION
Broken	Kết nối đã bị ngắt vì một lý do nào đó. Điều này có thể xảy ra sau khi bạn mở kết nối. Để khắc phục, ta gọi hàm đóng kết nối và rồi mở lại kết nối đó.
Closed	Kết nối đã bị đóng.
Connecting	Đang thiết lập kết nối tới cơ sở dữ liệu.
Executing	Kết nối đang được dùng trong quá trình thực thi một lệnh truy vấn.
Fetching	Kết nối đang được dùng trong quá trình nhận thông tin trả về từ cơ sở dữ liệu.
Open	Kết nối đang mở.

QUẢN LÝ SỰ KIỆN TRONG QUÁ TRÌNH KẾT NỐI

- Sự kiện StateChange
  - Phát sinh khi trạng thái của kết nối bị thay đổi
  - Dùng để quản lý sự thay đổi về trạng thái của đối tượng Connection
  - Đăng ký trình xử lý sự kiện

40

```
mySqlConnection.StateChange +=
  new StateChangeEventHandler(StateChangeHandler);

public static void StateChangeHandler(
  object mySender, StateChangeEventArgs myEvent)
{
   Console.WriteLine(
     "mySqlConnection State has changed from " +
     myEvent.OriginalState + " to " +
     myEvent.CurrentState );
}
```

QUẢN LÝ SỰ KIỆN TRONG QUÁ TRÌNH KẾT NỐI

- Sự kiện InfoMessage
  - phát sinh khi cơ sở dữ liệu trả về một cảnh báo hoặc một thông điệp.
  - dùng để quản lý các thông điệp, cảnh báo hoặc lỗi
  - Để lấy thông điệp, ta đọc nội dung thuộc tính Errors của tham số SqlInfoMessageEventArgs
  - Đăng ký trình xử lý sự kiện

```
mySqlConnection.InfoMessage +=
    new SqlInfoMessageEventHandler(InfoMessageHandler);

public static void InfoMessageHandler(
  object mySender, SqlInfoMessageEventArgs myEvent)
{
   Console.WriteLine(
    "The following message was produced:\n" +
    myEvent.Errors[0] );
}
```

41

DEMO

- Tạo kết nối cơ sở dữ liệu
- Tạo trình xử lý sự kiện
  - StateChange
  - InfoMessage

42

HẾT CHƯƠNG 3