

Mục tiêu

- Giúp sinh viên nắm rõ các vấn đề sau
 - Thực thi thao tác truy vấn dữ liệu và đọc kết quả trả về
 - Lưu trữ dữ liệu dùng Dataset
 - Trích lọc, sắp xếp dữ liệu với Dataview

Khoa CNTT - Đại học Đà Lạt

2

Truy vấn CSDL và đọc kết quả trả về

- Lớp SqlCommand
- Thực thi lệnh Select
- Thực thi các lệnh làm thay đổi CSDL
- Thực thi các giao dịch trong CSDL
- Truyền tham số vào các lệnh
- Gọi các thủ tục SQL Server

Khoa CNTT - Đại học Đà Lạt

3

Lớp SqlCommand

- Đối tượng SqlCommand được sử dụng để thực thi một lệnh truy vấn nào đó tới cơ sở dữ liệu SQL Server
- Các thuộc tính
 - String **CommandText** { get; set; }
 - Int **CommandTimeout** { get; set; } default: 30s
 - Bool **DesignTimeVisible** { get; set; }
 - SqlConnection **Connection** { get; }
 - CommandType **CommandType** { get; set; }
 - SqlTransaction **Transaction** { get; set; }
 - SqlParameterCollection **Parameters** { get; }
 - UpdateRowSource **UpdateRowSource** { get; set; }

Khoa CNTT - Đại học Đà Lạt

4

Lớp SqlCommand

- Phương thức
 - Void **Cancel** ()
 - Int **ExecuteNonQuery** ()
 - Object **ExecuteScalar** ()
 - Void **Prepare** ()
 - Void **ResetCommandTimeout** ()
 - SqlParameter **CreateParameter** ()
 - SqlDataReader **ExecuteReader** ()
 - XmlReader **ExecuteXmlReader** ()

Tạo đối tượng Command

- Cách 1: Dùng phương thức **CreateCommand** của đối tượng Connection

```
SqlConnection mySqlConnection = new SqlConnection();
mySqlConnection.ConnectionString = "DataSource=localhost;
    Initial Catalog=Northwind;Integrated Security=SSPI";
```

```
SqlCommand mySqlCommand =
    mySqlConnection.CreateCommand();
```

Tạo đối tượng Command

- Cách 2: Dùng phương thức tạo lập của lớp SqlCommand
 - `SqlCommand ()`
 - `SqlCommand (string commandText)`
 - `SqlCommand (string commandText, SqlConnection mySqlConnection)`
 - `SqlCommand (string commandText, SqlConnection mySqlConnection, SqlTransaction mySqlTransaction)`
- Trong đó
 - `commandText` Lệnh truy vấn hoặc tên thủ tục
 - `mySqlConnection` Đối tượng kết nối
 - `mySqlTransaction` Đối tượng giao dịch cơ sở dữ liệu
- Trong trường hợp `CommandText` là tên thủ tục, cần phải thiết lập giá trị cho thuộc tính `CommandType` là `CommandType.StoredProcedure`

Khoa CNTT - Đại học Đà Lạt

7

Tạo đối tượng Command

```
SqlConnection mySqlConnection = new SqlConnection();
mySqlConnection.ConnectionString = "DataSource=localhost; Initial
Catalog=Northwind;Integrated Security=SSPI";
```

- Ví dụ 1
 - `SqlCommand mySqlCommand = new SqlCommand();`
 - `mySqlCommand.Connection = mySqlConnection;`
 - `mySqlCommand.CommandText = "SELECT TOP 10 CustomerID, CompanyName, ContactName, Address" + "FROM Customers ORDER BY CustomerID";`
- Ví dụ 2
 - `SqlCommand mySqlCommand = new SqlCommand("SELECT TOP 5 CustomerID, CompanyName, ContactName, Address" + "FROM Customers ORDER BY CustomerID", mySqlConnection);`

Khoa CNTT - Đại học Đà Lạt

8

Thực thi lệnh SELECT và TableDirect

- Lệnh TableDirect \Leftrightarrow lệnh SELECT trả về tất cả các dòng, các cột của bảng.
- Có 3 phương thức để thực thi một câu lệnh SELECT hay TableDirect.
 - Object `ExecuteScalar ()`
 - SqlDataReader `ExecuteReader ()`
 - XmlReader `ExecuteXmlReader ()`

Khoa CNTT - Đại học Đà Lạt

9

Phương thức ExecuteReader

- Thực thi một lệnh SELECT và kết quả trả về được lưu trong một đối tượng DataReader


```
mySqlConnection.Open();
SqlDataReader mySqlDataReader = mySqlCommand.ExecuteReader();
```
- Để đọc lần lượt các dòng từ DataReader, sử dụng phương thức **Read ()**. Phương này trả về giá trị:
 - **True** cho biết còn dòng khác chưa được đọc
 - **False** nghĩa là tất cả các dòng đã được đọc
- Lấy giá trị trên một cột từ DataReader theo một trong các cách sau
 - **Object** `variable = myDataReader [columnIndex];`
 - **Object** `variable = myDataReader [columnName];`

Khoa CNTT - Đại học Đà Lạt

10

Phương thức ExecuteReader

```

SqlConnection mySqlConnection = new SqlConnection(
    "server=localhost;database=Northwind;uid=sa;pwd=sa" );
SqlCommand mySqlCommand = mySqlConnection.CreateCommand();

mySqlCommand.CommandText =
    "SELECT TOP 5 CustomerID, CompanyName, ContactName, Address FROM Customers
    ORDER BY CustomerID";

mySqlConnection.Open();
SqlDataReader mySqlDataReader = mySqlCommand.ExecuteReader();

while ( mySqlDataReader.Read() )
{
    Console.WriteLine( mySqlDataReader["CustomerID"] );
    Console.WriteLine( mySqlDataReader["CompanyName"] );
    Console.WriteLine( mySqlDataReader["ContactName"] );
    Console.WriteLine( mySqlDataReader["Address"] );
}

```

Khoa CNTT - Đại học Đà Lạt

11

Phương thức ExecuteReader

- Phương thức *ExecuteReader* chấp nhận một tham số (tùy chọn) để điều khiển cách thức xử lý lệnh và kết quả.

VALUE	DESCRIPTION
CloseConnection	Chỉ ra rằng, khi đối tượng DataReader bị đóng bởi phương thức Close(), đối tượng kết nối (Connection) mà Command sử dụng cũng bị đóng theo.
Default	Có nghĩa là đối tượng Command có thể trả về nhiều tập kết quả.
KeyInfo	Yêu cầu đối tượng Command trả về thông tin của các cột khóa chính trong tập kết quả.
SchemaOnly	Cho biết đối tượng Command chỉ trả về thông tin của các cột.
SequentialAccess	Cho phép đối tượng DataReader đọc các dòng có các cột chứa dữ liệu nhị phân kích thước lớn. SequentialAccess buộc DataReader đọc dữ liệu ở dạng stream. Để đọc tiếp dữ liệu từ Stream này, ta dùng phương thức GetBytes() hay GetChars() của DataReader.
SingleResult	Cho biết đối tượng Command chỉ trả về một tập kết quả.
SingleRow	Cho biết đối tượng Command chỉ trả về một dòng.

Khoa CNTT - Đại học Đà Lạt

12

Phương thức ExecuteReader

- Phương thức ExecuteReader cũng được dùng để thực thi lệnh TableDirect
- Lệnh TableDirect dùng để lấy tất các dòng, các cột của một bảng
 - `SELECT * FROM Tên_Bảng`
- Để thực thi lệnh TableDirect, ta gán giá trị cho các thuộc tính của đối tượng Command như sau
 - **CommandText** Tên bảng muốn lấy dữ liệu
 - **CommandType** CommandType.TableDirect
- SqlCommand không cho phép thực thi lệnh TableDirect. Thay vào đó, ta dùng OleDbCommand.

Khoa CNTT - Đại học Đà Lạt

13

Phương thức ExecuteReader

- **OleDbConnection** myOleDbConnection = **new OleDbConnection** ("Provider=SQLOLEDB; server=localhost; database=Northwind; uid=sa; pwd=sa");
- **OleDbCommand** myOleDbCommand = myOleDbConnection.CreateCommand();
- myOleDbCommand.CommandType = **CommandType.TableDirect**;
- myOleDbCommand.CommandText = **"Products"**;
- myOleDbConnection.Open();
- **OleDbDataReader** myOleDbDataReader = myOleDbCommand.**ExecuteReader**();

Khoa CNTT - Đại học Đà Lạt

14

Phương thức ExecuteScalar

- Dùng để thực thi một lệnh SQL SELECT và **chỉ trả về một giá trị**, những giá trị khác được loại bỏ.
- Giá trị được lấy nằm ở hàng đầu tiên, cột đầu tiên trong bảng kết quả và kiểu dữ liệu trả về là **object**.
- Ví dụ
 - Đếm số lượng mẫu tin (dùng hàm COUNT)


```
mySqlCommand.CommandText = "SELECT COUNT(*) FROM Products";
int returnValue = ( int ) mySqlCommand.ExecuteScalar();
```
 - Tìm điểm trung bình cao nhất (dùng hàm MAX)
 - ...

Khoa CNTT - Đại học Đà Lạt

15

Phương thức ExecuteNonQuery

- Được dùng để thực thi các lệnh SQL **không yêu cầu dữ liệu trả về** từ cơ sở dữ liệu
- Các lệnh thường được thực thi
 - INSERT
 - UPDATE
 - DELETE
 - Các lệnh DDL (Data Definition Language)
 - Lệnh gọi thủ tục không trả về dữ liệu.
- Giá trị trả về của hàm này chính là **số dòng bị ảnh hưởng** khi thực thi truy vấn.

Khoa CNTT - Đại học Đà Lạt

16

Phương thức ExecuteNonQuery

- Thực thi lệnh INSERT

```
SqlCommand mySqlCommand =
    mySqlConnection.CreateCommand();
mySqlCommand.CommandText =
    "INSERT INTO Customers ( CustomerID, CompanyName ) " +
    "VALUES ( 'J2COM', 'Jason Price Corporation' )";

int numberOfRows = mySqlCommand.ExecuteNonQuery();
```

- Kết quả trả về: numberOfRows = 1 (số dòng được thêm vào)

Khoa CNTT - Đại học Đà Lạt

17

Phương thức ExecuteNonQuery

- Thực thi lệnh UPDATE

```
mySqlCommand.CommandText =
    "UPDATE Customers " +
    "SET CompanyName = 'New Company' " +
    "WHERE CustomerID = 'J2COM'";

numberOfRows = mySqlCommand.ExecuteNonQuery();
```

- Thực thi lệnh DELETE

```
mySqlCommand.CommandText =
    "DELETE FROM Customers " +
    "WHERE CustomerID = 'J2COM'";

numberOfRows = mySqlCommand.ExecuteNonQuery();
```

Khoa CNTT - Đại học Đà Lạt

18

Phương thức ExecuteNonQuery

- Thực thi lệnh DDL

```
mySqlCommand.CommandText =
    "CREATE TABLE MyPersons (" +
    " PersonID int CONSTRAINT PK_Persons PRIMARY KEY," +
    " FirstName nvarchar(15) NOT NULL," +
    " LastName nvarchar(15) NOT NULL," +
    " DateOfBirth datetime )";

int result = mySqlCommand.ExecuteNonQuery();
```
- Kết quả trả về: **result = -1** vì lệnh CREATE TABLE không ảnh hưởng lên dòng nào của bảng

Thực thi giao dịch cơ sở dữ liệu

- Giao dịch cơ sở dữ liệu (**transaction**) là một nhóm các lệnh SQL.
- Các giao dịch hoặc được thực hiện thành công (**commit**) hoặc bị gỡ bỏ (**roll back**) như một lệnh thông thường.
- Để biểu diễn các giao dịch trong ADO.NET, dùng một trong các lớp Transaction sau
 - **SqlTransaction**
 - **OleDbTransaction**
 - **OdbcTransaction**
- Theo **mặc định**, các giao dịch sẽ bị gỡ bỏ (**roll back**). Vì thế, bạn luôn phải gọi phương thức **Commit** hoặc **Rollback** để chỉ rõ bạn muốn xác nhận hay hủy bỏ các giao dịch.

Thực thi giao dịch cơ sở dữ liệu

- Các bước thực hiện
 - Tạo một đối tượng **Transaction** và bắt đầu giao dịch bằng cách gọi phương thức **BeginTransaction** của đối tượng **Connection**.
 - Tạo đối tượng **Command** để thực thi truy vấn SQL.
 - Gán đối tượng **Transaction** đã tạo ở bước 1 cho thuộc tính **Transaction** của đối tượng **Command**.
 - Gán lệnh SQL cho thuộc tính **CommandText** của đối tượng **Command**.
 - Thực thi lệnh truy vấn bằng cách gọi một trong các phương thức **Execute...**
 - Xác nhận giao dịch bằng cách gọi phương thức **Commit** của đối tượng **Transaction**. Hoặc hủy bỏ giao dịch bằng cách gọi phương thức **Rollback**.

Khoa CNTT - Đại học Đà Lạt

21

Thực thi giao dịch cơ sở dữ liệu

- `mySqlConnection.Open();`
- `SqlTransaction mySqlTransaction = mySqlConnection.BeginTransaction();`
- `SqlCommand mySqlCommand = mySqlConnection.CreateCommand();`
- `mySqlCommand.Transaction = mySqlTransaction;`
- `mySqlCommand.CommandText =
"INSERT INTO Customers (CustomerID, CompanyName) "
"VALUES ('J3COM' , 'Jason Price Corporation')";`
- `mySqlCommand.ExecuteNonQuery();`

Khoa CNTT - Đại học Đà Lạt

22

Thực thi giao dịch cơ sở dữ liệu

- `mySqlCommand.CommandText = "INSERT INTO Orders (CustomerID) VALUES ('J3COM')";`
- `mySqlCommand.ExecuteNonQuery();`
- `mySqlTransaction.Commit();`
- `mySqlConnection.Close();`
- Nếu muốn hủy bỏ các lệnh đã tạo trong giao dịch, ta gọi phương thức **Rollback** thay vì **Commit**.

Truyền tham số vào lệnh

- Nhận xét
 - Giá trị trong các truy vấn được gán cố định
 - Phải viết nhiều lệnh hoặc chạy chương trình nhiều lần
 - Rắc rối & Không hiệu quả
- Giải pháp
 - Sử dụng một truy vấn có tham số
 - Truyền các giá trị khác nhau cho các tham số
- Cách thực hiện
 - Tạo đối tượng Command chứa lệnh SQL chứa điểm đánh dấu tham số
 - Thêm các tham số vào đối tượng Command
 - Truyền giá trị cho các tham số
 - Thực thi lệnh

Truyền tham số vào lệnh

- Tạo đối tượng Command có chứa tham số
 - Thay giá trị thực sự bằng một điểm đánh dấu tham số.
 - Cú pháp sử dụng điểm đánh dấu tùy thuộc vào cơ sở dữ liệu đang sử dụng.
 - Với SQL Server, điểm đánh dấu được ký hiệu bởi một tên, bắt đầu bằng dấu @
 - Điểm đánh dấu thay thế cho giá trị của một cột bất kỳ trong các lệnh SELECT, INSERT, UPDATE hay DELETE.

- Ví dụ

```
INSERT INTO Customers ( CustomerID, CompanyName, ContactName ) VALUES (
    @CustomerID, @CompanyName, @ContactName )
```

```
SELECT * FROM Customers WHERE CustomerID = @CustomerID
```

```
UPDATE Customers SET CompanyName = @CompanyName WHERE CustomerID
    = @CustomerID
```

```
DELETE FROM Customers WHERE CustomerID = @CustomerID
```

25

Truyền tham số vào lệnh

- Thêm tham số vào đối tượng Command
 - Gọi phương thức Add qua thuộc tính Parameters của đối tượng Command
 - Phương thức này có nhiều dạng với các tham số sau
 - Tên tham số = tên điểm đánh dấu tham số
 - Kiểu dữ liệu của tham số
 - Kích thước của giá trị mà tham số chứa (dùng cho giá trị kiểu chuỗi)

- Ví dụ:

```
SqlCommand mySqlCommand = mySqlConnection.CreateCommand();
mySqlCommand.CommandText = "INSERT INTO Customers ( CustomerID,
    CompanyName, ContactName ) VALUES ( @CustomerID, @CompanyName,
    @ContactName )";
mySqlCommand.Parameters.Add ("@CustomerID", SqlDbType.NChar, 5);
mySqlCommand.Parameters.Add ("@CompanyName", SqlDbType.NVarChar, 40);
mySqlCommand.Parameters.Add ("@ContactName", SqlDbType.NVarChar, 30);
```

Khoa CNTT - Đại học Đà Lạt

26

Truyền tham số vào lệnh

- Truyền giá trị cho tham số
 - Giá trị của tham số được gán qua thuộc tính Value theo cú pháp
`mySqlCommand.Parameters["@Tên_tham_số"].Value = Biểu_Thức;`
 - Hoặc
`mySqlCommand.Parameters.Add("@Tên_TS", KDL, KThước).Value = Biểu_Thức;`
- Gán giá trị NULL cho tham số
`mySqlCommand.Parameters["@Tên_TS"].IsNull = true;`
- Hoặc
`mySqlCommand.Parameters["@Tên_TS"].Value = DBNull.Value;`
- Ví dụ
`mySqlCommand.Parameters.Add("@CustomerID", SqlDbType.NChar, 5).Value = "J4COM";`
`mySqlCommand.Parameters["@CustomerID"].Value = "J4COM";`

Khoa CNTT - Đại học Đà Lạt

27

Truyền tham số vào lệnh

- Thực thi lệnh truy vấn
 - **ExecuteNonQuery** : thực thi các lệnh INSERT, UPDATE hay DELETE.
 - Nếu muốn thực thi lệnh SELECT, dùng các phương thức **ExecuteReader**, **ExecuteScalar** hoặc **ExecuteXmlReader**.

Khoa CNTT - Đại học Đà Lạt

28

Gọi thủ tục SQL Server

- Có 2 cách gọi thủ tục SQL trong ADO.Net
 - Gán tên thủ tục cho thuộc tính **CommandText**, dạng thực thi lệnh cho thuộc tính **CommandType** của đối tượng Command
 - myCommand. **CommandText** = "Tên_Thủ_Tục";
 - myCommand. **CommandType** = CommandType. **StoredProcedure**;
 - Sử dụng lệnh T-SQL: **EXECUTE**
 - myCommand. **CommandText** = "EXECUTE Tên_Thủ_Tục (@Danh_Sách_TS)"
 - Có thể lấy dữ liệu trả về bởi lệnh **RETURN** từ thủ tục → **Hiệu quả hơn**
- Việc thực thi thủ tục phụ thuộc vào việc thủ tục có trả về tập dữ liệu hay không.

Khoa CNTT - Đại học Đà Lạt

29

Gọi thủ tục SQL Server

- Thực thi thủ tục không trả về tập dữ liệu
 - Tạo đối tượng Command với lệnh T-SQL: EXECUTE
 - Thêm các tham số vào đối tượng Command
 - Tham số OUTPUT phải được gán giá trị cho thuộc tính Direction là ParameterDirection.Output
- ```
myCommand. Parameters ["@Tên_TS"]. Direction = ParameterDirection.Output
```
- Truyền giá trị cho các tham số
- Thực thi lệnh bởi phương thức ExecuteNonQuery
  - Đọc giá trị trả về từ các tham số OUTPUT
- ```
Object returnedValue = mySqlCommand. Parameters [ "@Tên_TS" ]. Value;
```

Khoa CNTT - Đại học Đà Lạt

30

Gọi thủ tục SQL Server

```
CREATE PROCEDURE AddProduct2 ( @MyProductName nvarchar(40),
    @MySupplierID int, @MyCategoryID int, @MyQuantityPerUnit nvarchar(20),
    @MyUnitPrice money, @MyUnitsInStock smallint, @MyUnitsOnOrder smallint,
    @MyReorderLevel smallint, @MyDiscontinued bit )
AS
DECLARE @MyProductID int

INSERT INTO Products ( ProductName, SupplierID, CategoryID, QuantityPerUnit,
    UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, Discontinued )
VALUES ( @MyProductName, @MySupplierID, @MyCategoryID, @MyQuantityPerUnit,
    @MyUnitPrice, @MyUnitsInStock, @MyUnitsOnOrder, @MyReorderLevel,
    @MyDiscontinued )

SET @MyProductID = SCOPE_IDENTITY(); --@@@IDENTITY

RETURN @MyProductID
```

Khoa CNTT - Đại học Đà Lạt

31

Gọi thủ tục SQL Server

- Lấy dữ liệu trả về bởi lệnh RETURN

```
mySqlCommand.CommandText =
    "EXECUTE @MyProductID = AddProduct2 @MyProductName, " +
    "@MySupplierID, @MyCategoryID, @MyQuantityPerUnit, " +
    "@MyUnitPrice, @MyUnitsInStock, @MyUnitsOnOrder, " +
    "@MyReorderLevel, @MyDiscontinued";

mySqlCommand.Parameters.Add("@MyProductID", SqlDbType.Int);
mySqlCommand.Parameters["@MyProductID"].Direction = ParameterDirection.Output;

mySqlCommand.Parameters.Add( "@MySupplierID", SqlDbType.Int ).Value = 1;
...

mySqlCommand.ExecuteNonQuery();
Console.WriteLine("New ProductID = " +
    mySqlCommand.Parameters["@MyProductID"].Value);
```

32

Gọi thủ tục SQL Server

- Thực thi một thủ tục có trả về tập dữ liệu
 - Tạo đối tượng Command và gán lệnh EXECUTE cho thuộc tính CommandText
 - Thêm các tham số vào đối tượng Command
 - Lưu ý các tham số OUTPUT
 - Truyền giá trị vào các tham số
 - Thực thi lệnh dùng phương thức ExecuteReader
 - Đọc các dòng dữ liệu bởi phương thức Read
 - Đóng đối tượng DataReader bằng phương thức Close
 - Đọc giá trị trả về từ các tham số OUTPUT

Khoa CNTT - Đại học Đà Lạt

33

Gọi thủ tục SQL Server

```
CREATE PROCEDURE AddProduct3 ( @MyProductName nvarchar(40),
    @MySupplierID int, @MyCategoryID int, @MyQuantityPerUnit nvarchar(20),
    @MyUnitPrice money, @MyUnitsInStock smallint, @MyUnitsOnOrder smallint,
    @MyReorderLevel smallint, @MyDiscontinued bit )
AS DECLARE @MyProductID int

INSERT INTO Products ( ProductName, SupplierID, CategoryID, QuantityPerUnit,
    UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, Discontinued )
VALUES ( @MyProductName, @MySupplierID, @MyCategoryID, @MyQuantityPerUnit,
    @MyUnitPrice, @MyUnitsInStock, @MyUnitsOnOrder, @MyReorderLevel,
    @MyDiscontinued )

SET @MyProductID = SCOPE_IDENTITY(); @@IDENTITY
SELECT ProductName, UnitPrice FROM Products WHERE ProductID = @MyProductID

RETURN @MyProductID
```

Khoa CNTT - Đại học Đà Lạt

34

Gọi thủ tục SQL Server

```

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

mySqlConnection.Open();
SqlCommand mySqlCommand = mySqlConnection.CreateCommand();

mySqlCommand.CommandText = "EXECUTE @MyProductID = AddProduct3
@MyProductName, @MySupplierID, @MyCategoryID,
@MyQuantityPerUnit, @MyUnitPrice, @MyUnitsInStock,
@MyUnitsOnOrder, @MyReorderLevel, @MyDiscontinued";

mySqlCommand.Parameters.Add ("@MyProductID", SqlDbType.Int);
mySqlCommand.Parameters ["@MyProductID"]. Direction =
ParameterDirection.Output;

... // Thêm các tham số khác

```

Khoa CNTT - Đại học Đà Lạt

35

Gọi thủ tục SQL Server

```

SqlDataReader mySqlDataReader = mySqlCommand.ExecuteReader();

while (mySqlDataReader. Read () )
{
    Console.WriteLine ( mySqlDataReader["ProductName"] );
    Console.WriteLine ( mySqlDataReader["UnitPrice"] );
}

mySqlDataReader.Close();

Console.WriteLine("New ProductID = " +
mySqlCommand.Parameters["@MyProductID"].Value);

mySqlConnection.Close();

```

Khoa CNTT - Đại học Đà Lạt

36

Tạo đối tượng Command bằng IDE

- Tạo đối tượng Connection
- Cấu hình thông tin kết nối cơ sở dữ liệu
- Tạo đối tượng Command
- Cấu hình lệnh truy vấn
- Thực thi lệnh và hiển thị kết quả

Khoa CNTT - Đại học Đà Lạt

37

Thực hành

- Làm việc theo nhóm
- Thực hiện tiếp các yêu cầu theo từng đề tài
- Tập trung thực hiện các công việc sau
 - Viết hàm thực thi các lệnh truy vấn (trực tiếp không có tham số, có tham số) hoặc gọi thủ tục SQL Server
 - Nhập - Xuất kết quả lên màn hình (Console & Windows App)

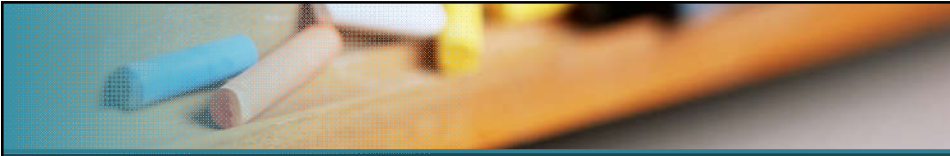
Khoa CNTT - Đại học Đà Lạt

38



Bài tập nhóm

- Thiết kế cơ sở dữ liệu
- Tạo khung nhìn (nếu cần)
- Tạo các thủ tục (stored procedure): ứng với mỗi bảng
 - Thêm
 - Xóa
 - Cập nhật
 - Tìm theo điều kiện
 - Lấy tất cả
 - Lấy mẫu tin theo mã
- Tạo hàm (nếu cần)
- Tạo trigger (nếu cần)



Khoa CNTT - Đại học Đà Lạt

40