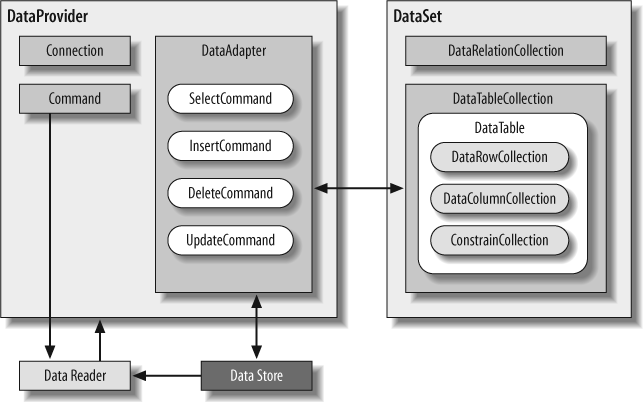
# Giới thiệu về ADO.NET

Trong phần này ta sẽ tìm hiểu về thư viện dùng để truy cập CSDL ADO.NET

ADO.NET là một tập hợp các thư viện cho phép ta tương tác với CSDL như SQL Server, My SQL Server,..

Kiến trúc cơ bản của ADO.NET gồm nhiều thành phần



Thu viện gồm nhiều lớp cấu trúc được chia ra làm 2 phần chính:

* **DataProvider** là các thư viện lớp cung cấp chức năng để kết nối đến những nguồn dữ liệu (VD: SQL Server,..) Cho phép thi hành các chức năng để thực hiện truy vấn CSDL từ câu lệnh INSERT, SELECT,…
* **DataSet** là một tập hợp các thư viện lớp cho phép tạo ra những đối tượng để quẻn lý dữ liệu và không phụ thuộc vào nguồn dữ liệu đó đến từ đâu. Nói cách khác DataSet là ánh xạ của CSDL về máy local
  + Trong DataSet tập hợp nhiều đối tượng DataTable, mỗi đối tượng DataTable tương ứng với một bảng trong CSDL mà chúng ta truy vấn

## DataProvider

Trong DataProvider gồm nhiều thành phần:

* **DbConnection**: là đối tượng cung cấp tài nguyên để kết nối đến CSDL
* **DbCommand**: là đối tượng sử dụng **DbConnection** để thực hiện các truy vấn
* **Data Reader**: là đối tượng dùng để đọc những kết quả truy vấn của **DbCommand**
* **DataAdapter:** gồm nhiều **DbCommand** (SelectCommand, InsertCommand,DeleteCommand, UpdateCommand). Đối tượng này giống như một cầu nối giữa nguồn dữ liệu (SQL Server) và **DataSet.** Khi có yêu cầu thì nó có thể lấy dữ liệu từ nguồn về **DataSet** và khi dữ liệu chỉnh sửa trong **DataSet** thông qua **DataAdapter** nó sẽ cập nhật trở lại dữ liệu nguồn (SQLServer)

Trong thư viện .NET có cung cấp DataProvider để truy cập vào SQL Server

Để cài đặt package ta sử dụng lệnh sau:

dotnet **add** package System.Data.SqlClient

# Tạo SQL Server chạy trên Docker

Dữ liệu mẫu:

<https://github.com/xuanthulabnet/learn-cs-netcore/releases/tag/exsqldocker>

Sau khi tải về và giải nén ta có thư mục

A screenshot of a computer

Description automatically generatedThư mục này chứa các config của docker

A screenshot of a computer program

Description automatically generatedNội dung file này bao gồm:

* Phiên bản đang chạy: 2017-latest
* Tên Container là: sqlserver-xtlab
* Khi SQLServer chạy nó thiết lập password cho tài khoản SA(Super Admin)
* Có tạo ra 1 volume để lưu trữ CSDL
* Cổng mặc định của SQL Server là cổng 1433

Truy cập thư mục MSSQL bằng terminal

A computer screen with white text

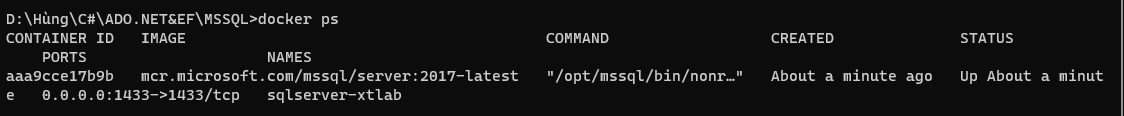
Description automatically generatedSử dụng lệnh dir để kiểm tra thành phần folder

Ta chạy lệnh để khởi tạo file

A screenshot of a computer program

Description automatically generated

Sau khi chạy lệnh ta kiểm tra bằng lệnh docker ps



Ta thấy ta đang có 1 container là sqlserver-xtlab đang chạy

Như vậy ta đang chạy SQLServer trên docker

Đối với SQL server này chúng ta có thể truy cập được từ địa chỉ localhost:1433 hoặc thông qua địa chỉ ip là 127.0.0.1 với tài khoản truy cập là “sa” password: “Password123”

Chúng ta có thể sử dụng công cụ Azure Data Studio để quản lý SQL server

A screenshot of a computer

Description automatically generatedĐây là ứng dụng Azure Data Studio, nó cho phép chúng ta kết nối đến SQL server và quản lý SQL server

Chúng ta có thể tạo ra kết nối đến SQL server đangc hạy trên Docker

**A screenshot of a computer

Description automatically generated**

Sau khi kết nối:

A screenshot of a computer

Description automatically generated

Phục hồi CSDL mẫu:

Mở terminal và chạy lệnh sau:

A black screen with white text

Description automatically generated

Chúng ta đang ở trong container chạy sql server

Trong file docker-compose khi chúng ta ánh xạ thư mục bk tương ứng là /var/opt/mssql/backup

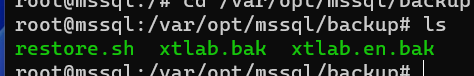
A close up of text

Description automatically generated

Tiếp tục đi đến thư mục

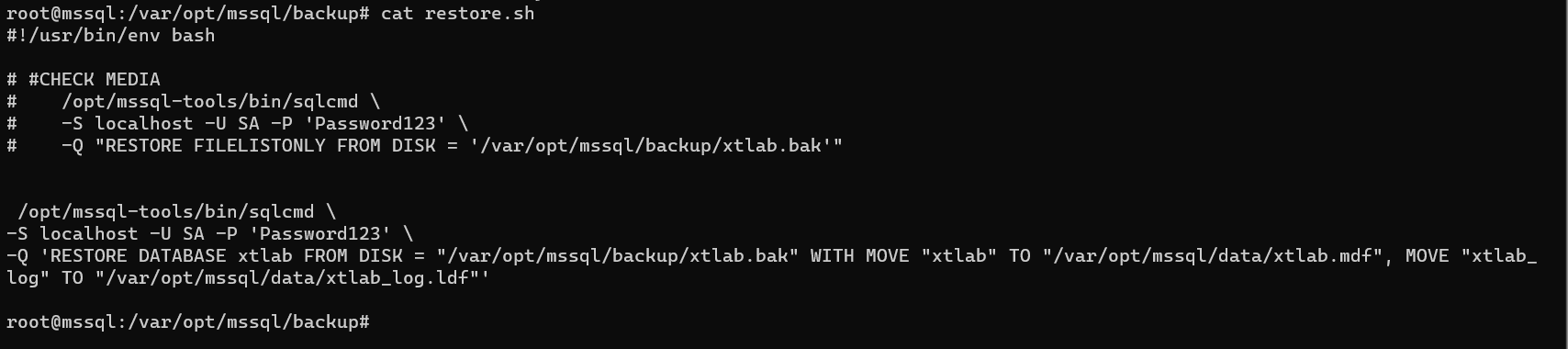
A black background with white text

Description automatically generated

Trong thư mục backup có file lưu trữ csdl mẫu là xtlab.bak

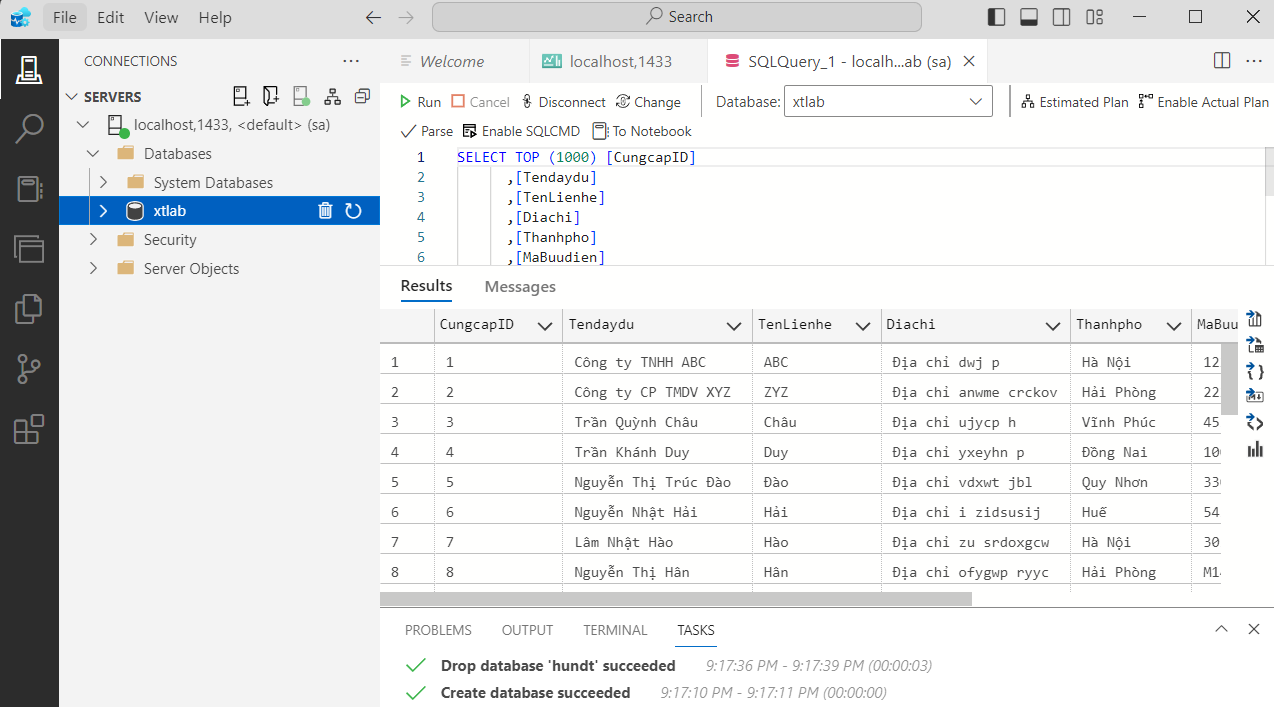
Chúng ta sẽ phục hồi CSDL này để trở thành csdl chạy trên docker

Nội dung file restore.sh



Sau khi hiển thị nội dung ta chạy file này ./restore.sh

Kết quả:

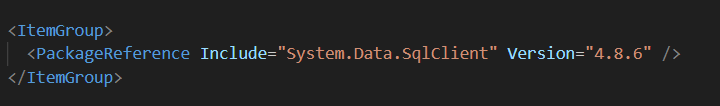


# Thực hành sử dụng SqlClient

* Trong phần này ta sẽ thực hành trên SqlClient để tạo ra 1 Connection kết nối đến CSDL trên SQL Server mà chúng ta dndag chạy trên docker
* Connection là đối tượng tài nguyên cơ bản đầu tiên mà ta cần tạo ra nếu muốn thực hiện kết nối đến SQL

Do chúng ta sử dụng SqlClient nên trong dự án cần add package **System.Data.SqlClient**

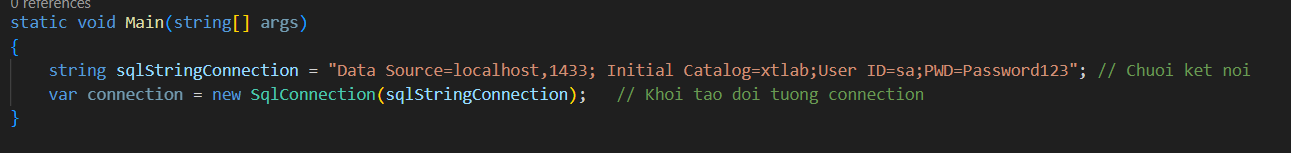




## Tạo chuỗi kết nối với SQL Server

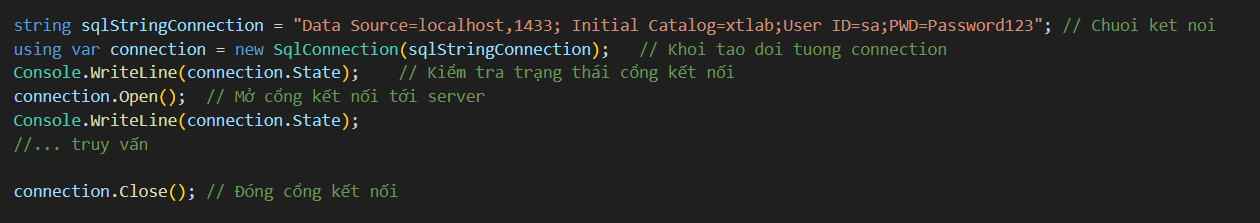
* Chuỗi kết nối có các thành phần, thông số để kết nối đến server
* Thông tin của chuỗi kết nối này được viết là các cặp key=value;

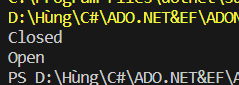




Ta sẽ sử dụng chuỗi kết nối này để làm tham số cho SqlConnection

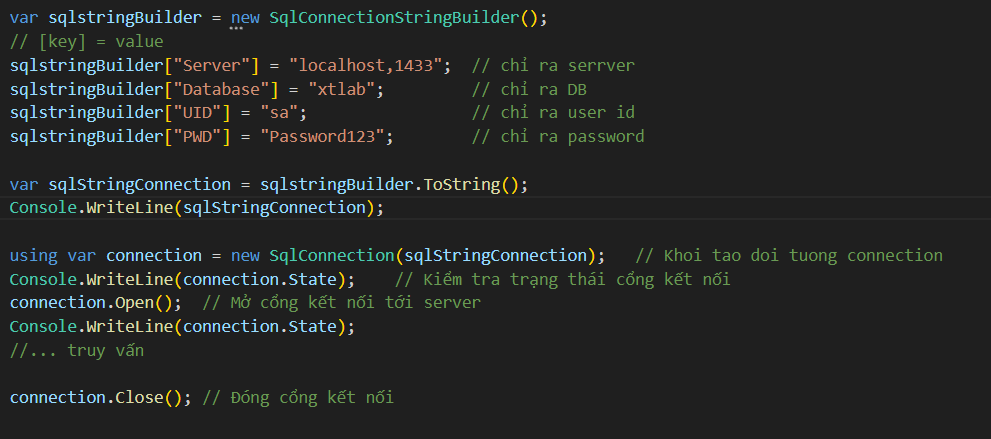
Sau khi có connection để mở kết nối đến server ta thực hiện

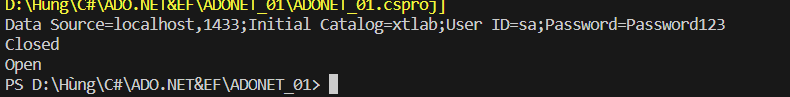




Khi cổng kết nối được mở ta có thể thực hiện các truy vấn sau khi thực hiện các truy vấn ta cần đóng lại chuỗi kết nối để giải phóng tài nguyên

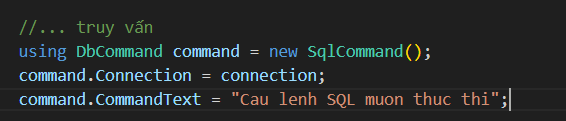
Trong .NET ta sẽ dùng đối tượng **SqlConnectionStringBuilder** để tạo ra chuỗi kết nối



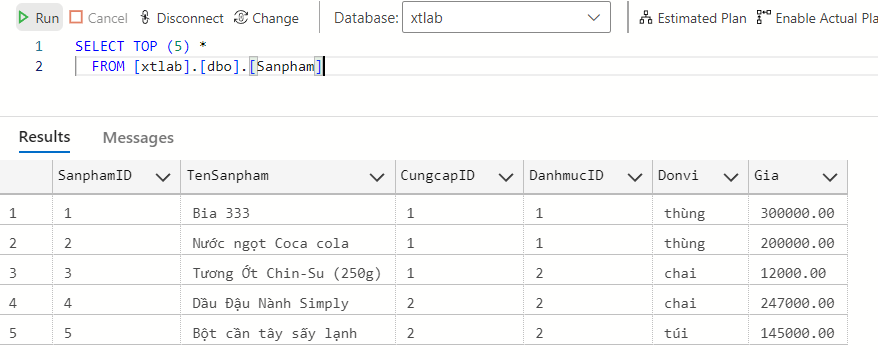


# DbCommand

* Đối tượng sử dụng để thực hiện câu truy vấn SQL đến CSDL



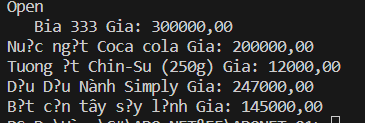
Giả sử trong SQL ta có câu lệnh để lấy ra 5 sản phẩm đầu tiên của bảng sản phẩm



Trong code ta thực hiện như sau:



Kết quả:



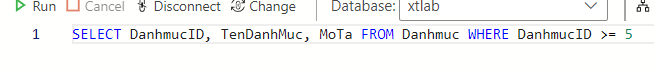
# SqlCommand

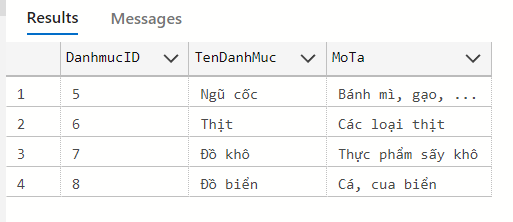
Như bài trước sau khi mở cổng kết nối 

Có 3 cách truy vấn như trong hình và mỗi cách có một công dụng khác nhau

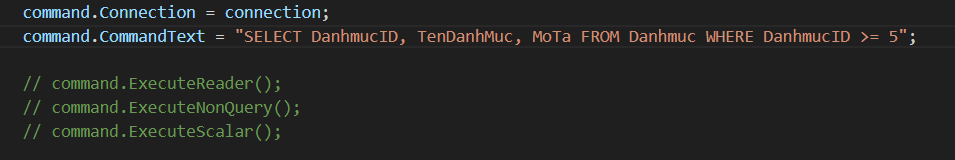
## Truyền tham số vào câu truy vấn SQL

Giả sử ta truy vấn như sau:





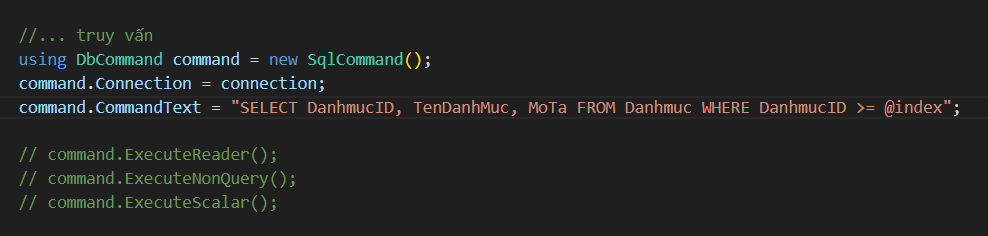
Như phần trước ta hoàn toàn có thể sử dụng câu truy vấn này trong code qua đối tượng SqlCommand



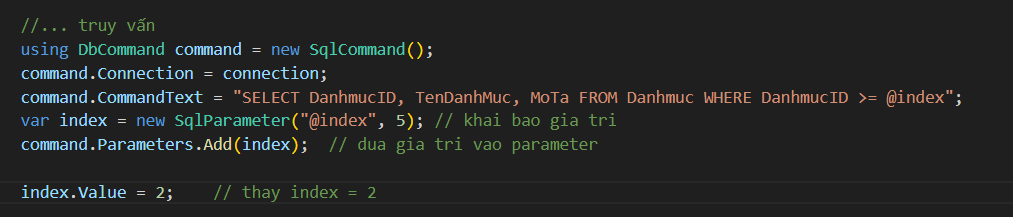
Sau đó khi chúng ta truy vấn với 1 trong 3 câu lệnh bên dưới câu truy vấn SQL sẽ được thực thi

Nhưng lúc này ta muốn DanhmucID có thể thay đổi mà không phải giá trị cứng

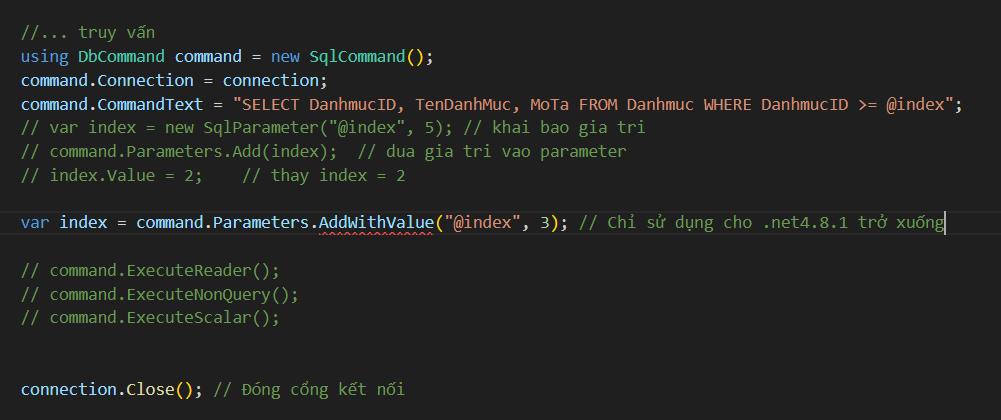
Để làm được điều này ta sẽ truyền vào trong câu lệnh SQL 1 tham số như sau:



Lúc này khi truy vấn SqlCommand sẽ tìm các tham số trong thuộc tính parameter

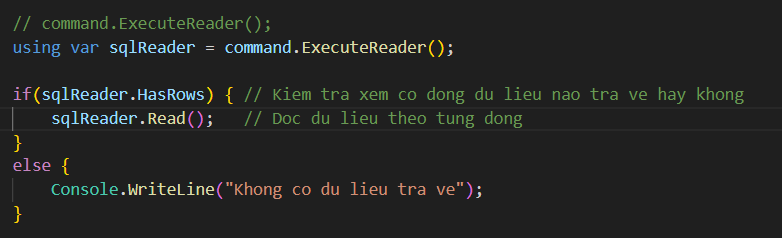


Ngoài cách này ta có thể làm như sau:



## ExecuteReader

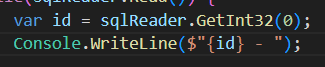
* Phương thức này trả về đối tượng SqlDataReader, và từ đối tượng này ta có thể đọc được các đối tượng truy vấn được

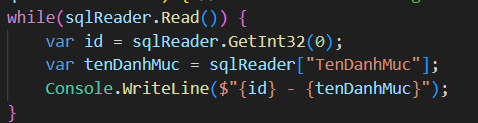


Phương thức Read() nếu gọi lần đầu tiên nó sẽ đọc dữ liệu dòng đầu tiên, tiếp đó con trỏ sẽ nhảy sang dòng tiếp theo, nếu thực hiện Read() lần thứ 2 thì nó sẽ đọc lần thứ 2,… cứ như vậy cho đến hết

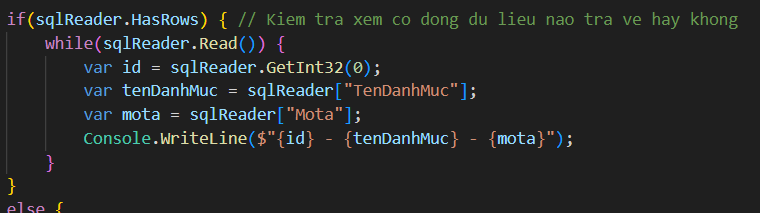
* Phương thức Read() nếu trả về True nghĩa là có dòng dữ liệu
* Phương thức Read() nếu trả về False nghĩa là đến cuối của dữ liệu không còn dữ liệu cần đọc

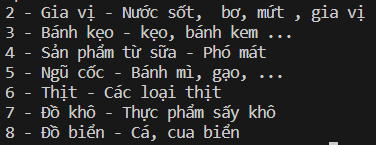
Vì vậy để đọc hết toàn bộ dữ liệu ta sẽ làm như sau:

Cách này đọc dữ liệu thông qua chỉ mục

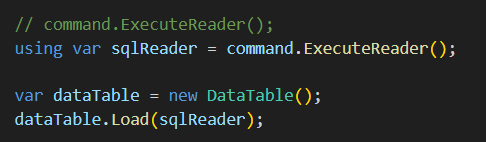


Cách này đọc dữ liệu thông qua Tên của trường dữ liệu



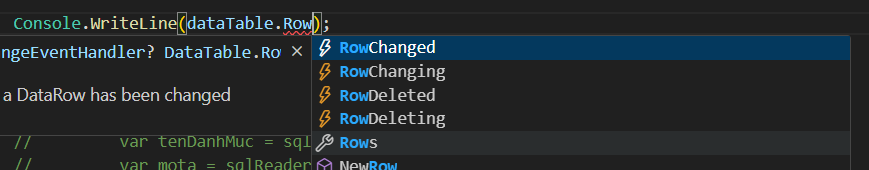


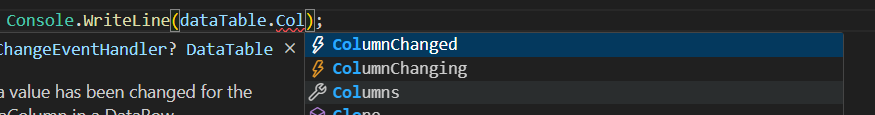
Sau khi có sqlReader thì toàn bộ dữ liệu truy vấn sẽ đổ ra 1 đối tượng là DataTable



Sau khi viết thế này thì toàn bộ dữ liệu truy vấn sẽ được lưu trong đối tượng dataTable

Trong đối tượng này gồm có nhiều hàng và cột





Mỗi hàng gồm các trường dữ liệu thì ta có thể duyệt qua cái hàng này để xử lý và sử dụng dữ liệu

Kết luận:

Chúng ta sử dụng ExecuteReader() khi câu truy vấn trả về một tapapj kết quả có nhiều dòng

## ExecuteScalar()