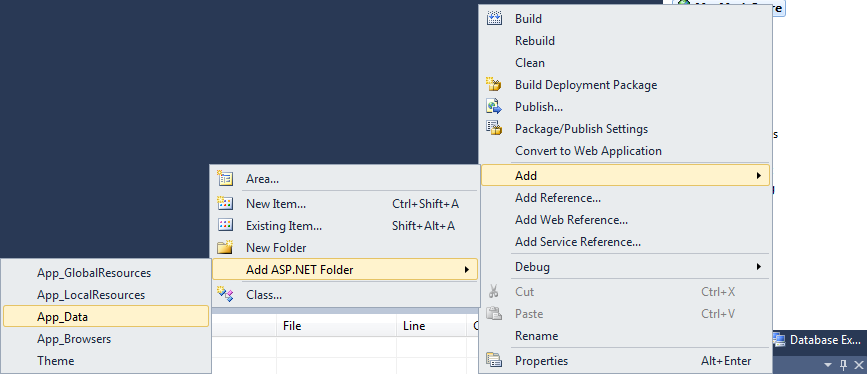
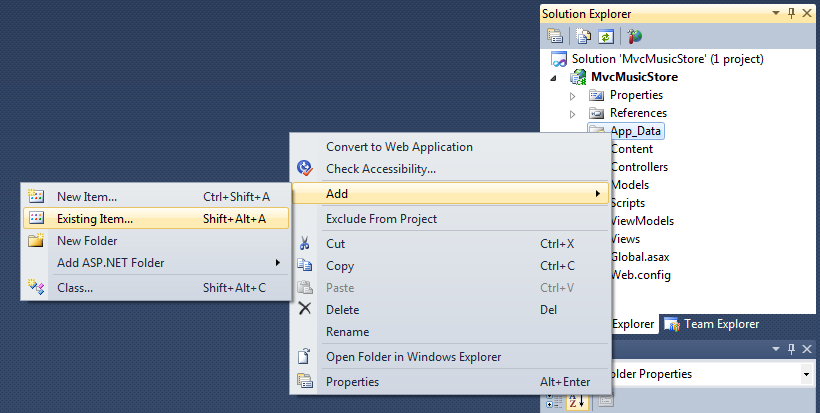
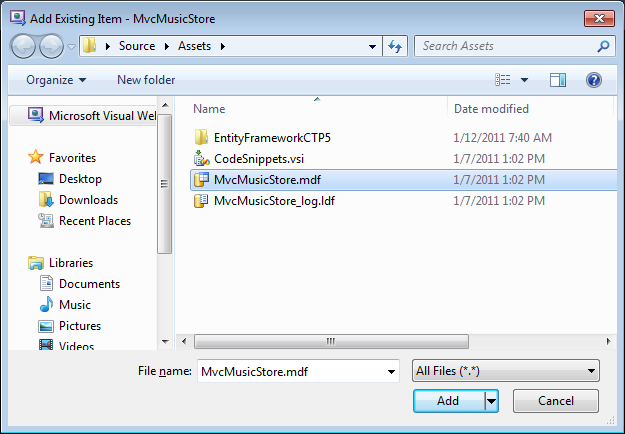
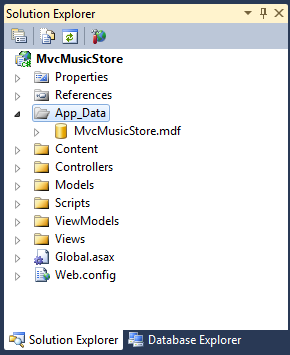
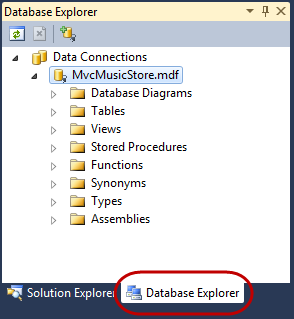
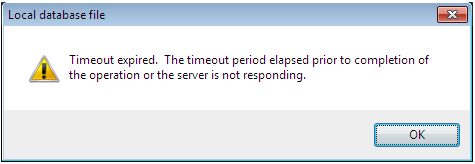
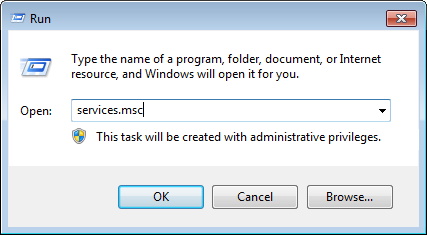
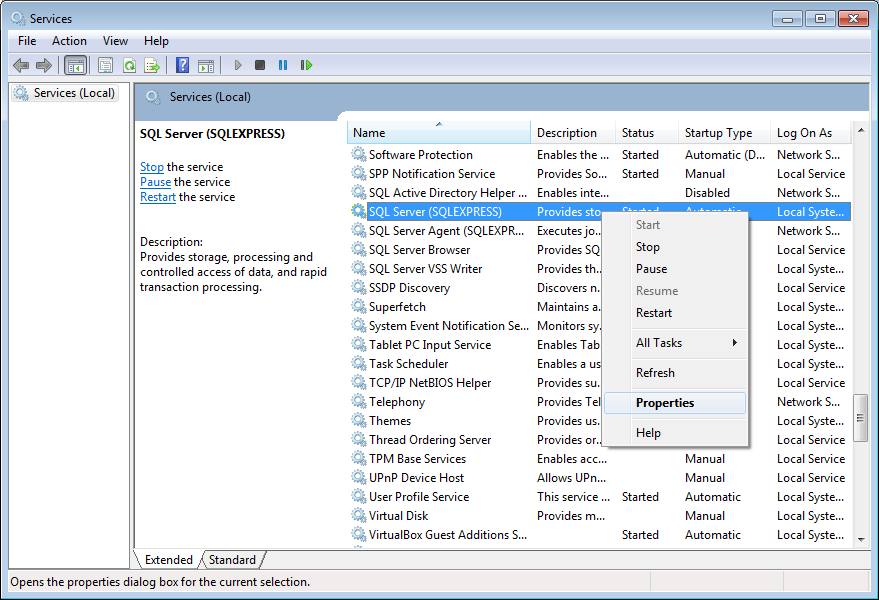
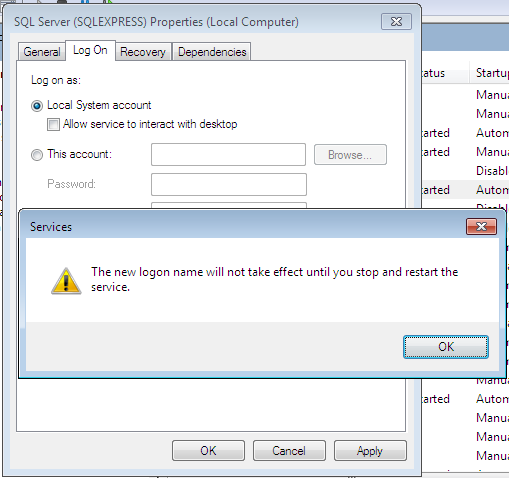
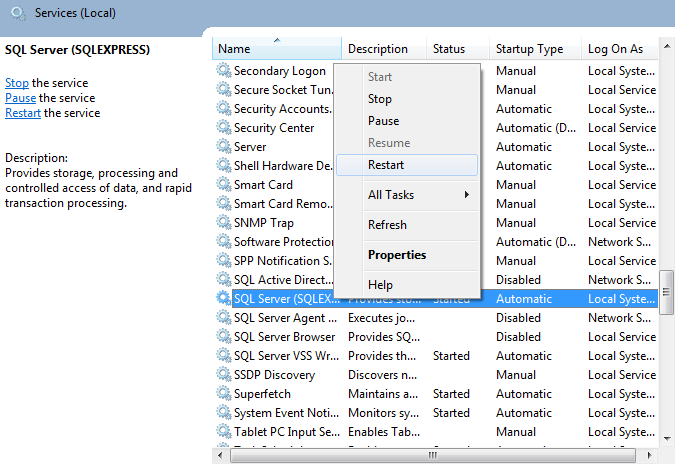
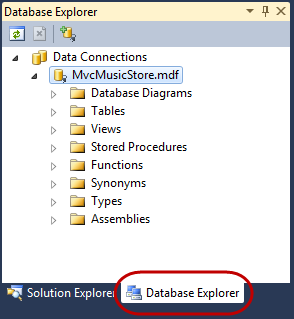
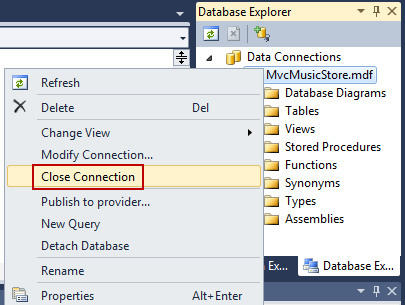
Bài tập 1: Thêm database

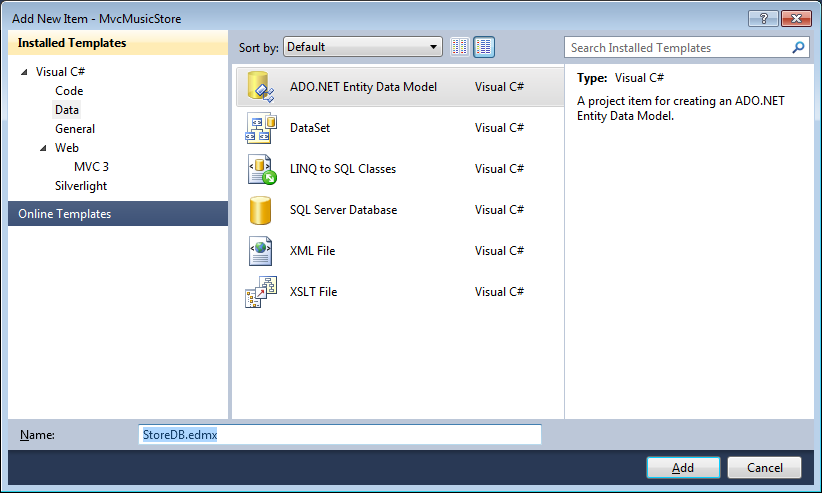
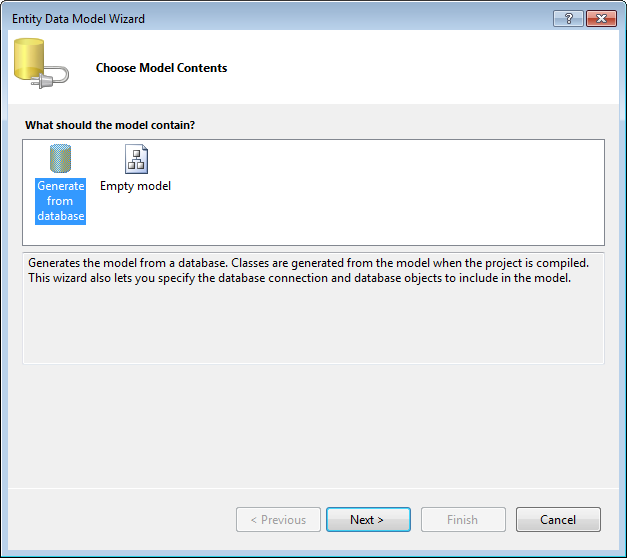
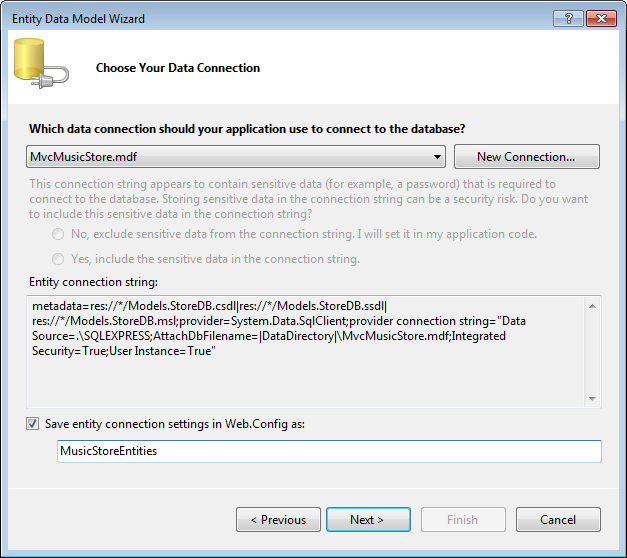
Trong bài tập này, bạn sẽ tìm hiểu làm thế nào để thêm một database với các bảng vào solution MusicStore và tương tác dữ liệu.

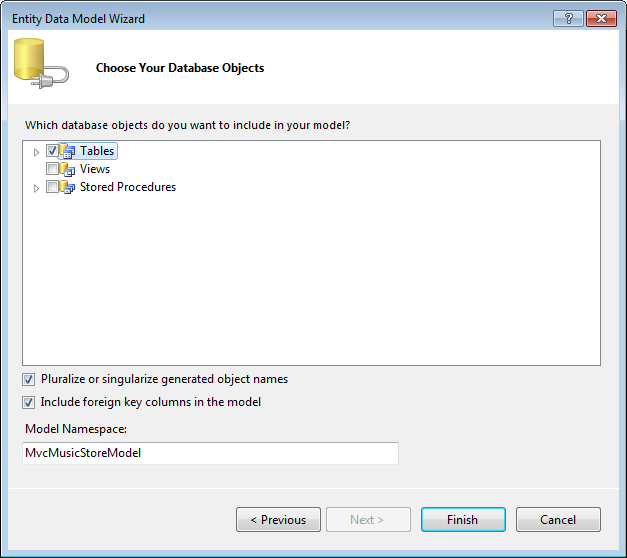
Task 1 – Thêm database

1. Trong task này, bạn sẽ thêm một database đã được tạo ra sẵn với các bảng chính của ứng dụng MusicStore vào solution.
   1. Khởi động Visual Studio
   2. Mở Solution **MvcMusicStore.sln**
   3. Thêm một thư mục **App\_Data** vào project để chứa các tập tin database SQL Server Express. **App\_Data** là một thư mục đặc biệt trong ASP.NET đã có điều khoản bảo mật để truy cập database.
      1. 
      2. Hình2
      3. Thêm thư mục App\_Data
   4. Thêm file database ứng dụng MusicStore vào solution: file **MvcMusicStore.mdf.**
      1. 
      2. Hình3
      3. Thêm file có sẵn
      4. 
      5. Hình4
      6. Tập tin database MvcMusicStore.mdf
      7. Database đã được thêm vào project. Ngay cả khi database nằm bên trong project, bạn có thể truy vấn và cập nhật nó giống như nó đã được lưu trữ trong một máy chủ database khác.
      8. 
      9. Hình5
      10. Database MvcMusicStore trong Solution Explorer
   5. Kiểm tra các kết nối database. Mở Database Explorer, nhắp đúp **MvcMusicStore.mdf**, Kết nối được thành lập.
      1. 
      2. Hình6
      3. Kết nối đến MvcMusicStore.mdf
      4. **Chú ý: nếu bạn gặp lỗi như hình, hãy làm theo các bước ở dưới**
      5. 
      6. 1. Mở Windows Services console. Chạy Run command bằng cách **Start** | **All Programs** | **Accessories**|**Run**, gõ**services.msc**và click **OK.**
      7. 
      8. **Hình7**
      9. *Chạy services.msc*
      10. 2. Click phải**SQL Server (SQLEXPRESS)** service and chọn**Properties**.
      11. 
      12. **Hình8**
      13. *SQL Server (SQLEXPRESS) service*
      14. 3. Mở tab**Log On**, chọn**Local System account**là tài khoản để đăng nhậpvà click **OK**. Sau đó click OK lần nữa.
      15. 
      16. **Hình9**
      17. *Thay đổi tài khoản đăng nhập*
      18. 4. Khởi động lại SQL Server (SQLEXPRESS) service.
      19. 
      20. **Hình10**
      21. *Khởi động lại SQL Server (SQLEXPRESS) service*
      22. 5. Khi service được khởi động, đóng **Services** console and kết nối database. Chọn**Database Explorer**, nhắp đúp**MvcMusicStore.mdf**. Kết nối được thiết lập.
      23. 
      24. **Hình11**
      25. *Kết nối MvcMusicStore.mdf*
      26. Để đóng kết nối, nhắp phải vào MvcMusicStore và lựa Close Connection.
      27. **
      28. *Hình 32*
      29. *Đong kết nối*

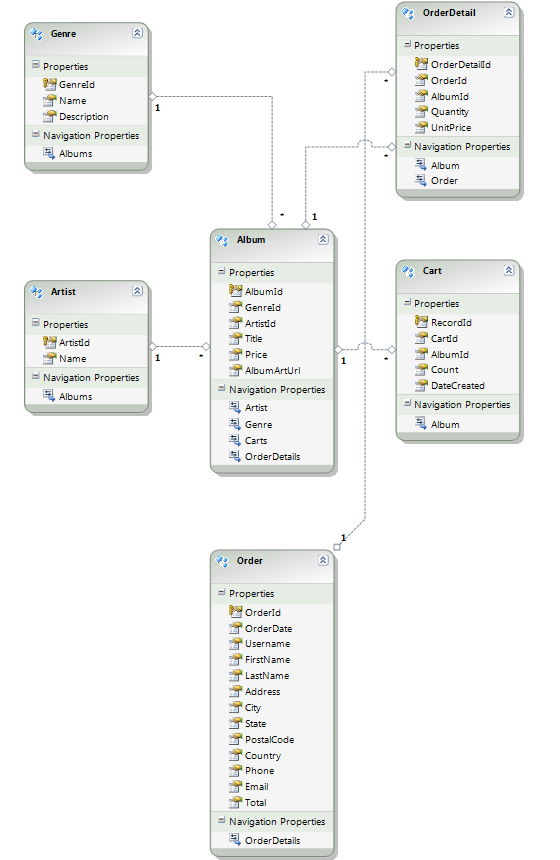
Task 2 – Tạo một Data Model

Trong task này, bạn sẽ tạo ra một mô hình dữ liệu (Data model) để tương tác với các database được thêm vào trong task trước.

1. Tạo ra một data model sẽ đại diện cho database được add vào. Trong Solution explorer, click phải lên thư mục Model, chọn Add/New Item, sau đó chọn template Data và chọn **ADO.NET Entity Data Model.** Đổi tên thành **StoreDB.edmx và click Add.**
   * 1. 
     2. Hình12
     3. Thêm một StoreDB ADO.NET Entity Data Model
2. Hộp thoại **Entity Data Model Wizard** xuất hiện, cái này sẽ giúp tạo ra các lớp mô hình (model layer), chọn **Generate from database** rồi chọn **Next.**
   * 1. 
     2. Hình13
     3. Chọn nội dung model
3. Sau khi tạo ra một mô hình từ database, bạn phải xác định database được sử dụng. Hộp thoại xuất hiện sẽ tìm ra database trong thư mục App\_Data của bạn, và sẽ điền thong tin chính xác kết nối đến database của bạn, các lớp được tạo ra sẽ có cùng tên với chuỗi thực thể kết nối của bạn, do đó hãy đổi tên thành **MusicStoreEntities** và click **Next.**
   * 1. 
     2. Hình14
     3. Chọn kết nối dữ liệu
4. Chọn các đối tượng cơ sở dữ liệu bạn sử dụng, đánh dấu vào **Table** và **Include foreign key columns in the model** ở bên dưới, đổi tên **Model Namespace** thành **MvcMusicStoreModel** và click Finish.

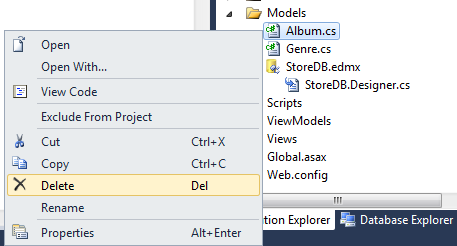
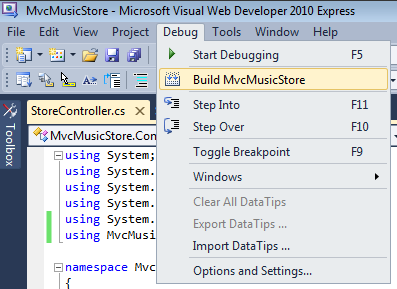
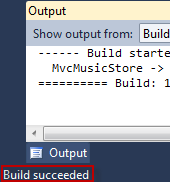
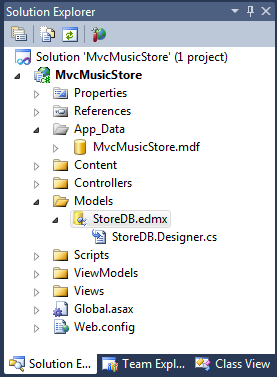


* + 1. Hình15
    2. Chọn đối tượng database

1. Một mô hình thực thể cho database sẽ xuất hiện. Các class riêng biệt sẽ được tạo ra tương ứng với mỗi bảng trong database của bạn. Điều này sẽ cho phép bạn truy vấn và làm việc với các đối tượng đại diện cho các hàng trong database.
   * 1. 
     2. Hình16
     3. Entity diagram

Task 3 – Build ứng dụng

Trong nhiệm vụ này, bạn sẽ kiểm tra rằng mặc dù bạn gỡ bỏ các lớp mô hình Album và Genre, project vẫn được build thành công, bằng cách sử dụng các lớp trong mô hình dữ liệu (Data model).

1. Xóa class **Album** và **Genre**. Trong Solution explorer, mở thư mục Models, click phải lên class **Album** và chọn **Delete**. Làm tương tự cũng xóa class **Genre**.
   * 1. 
     2. Hình17
     3. Xóa class
2. Build project bằng cách chọn menu **Debug** và chọn **Build MvcMusicStore**.
   * 1. 
     2. Hình18
     3. Build project
3. Project sẽ được thông báo là Build thành công, tại sao no vẫn làm việc?? Là bởi vì các bảng database có trường nằm trong properties của class **Album** và **Genre** bạn đã xóa nhưng các lớp mô hình dữ liệu (Data model) đã thay thế nó.
   * 1. 
     2. Hình19
     3. Build thành công
4. Khi nhắp vào **StoreDB.edmx** ta sẽ thấy một sơ đồ thực thể các lớp, chúng được hiển thị dựa trên các lớp C#, mở **StoreDB.edmx** trong Solution explorer bạn sẽ thấy một file có tên **StoreDB.Designer.cs.**
   * 1. 
     2. Hình20
     3. Tập tin StoreDB.Designer.cs

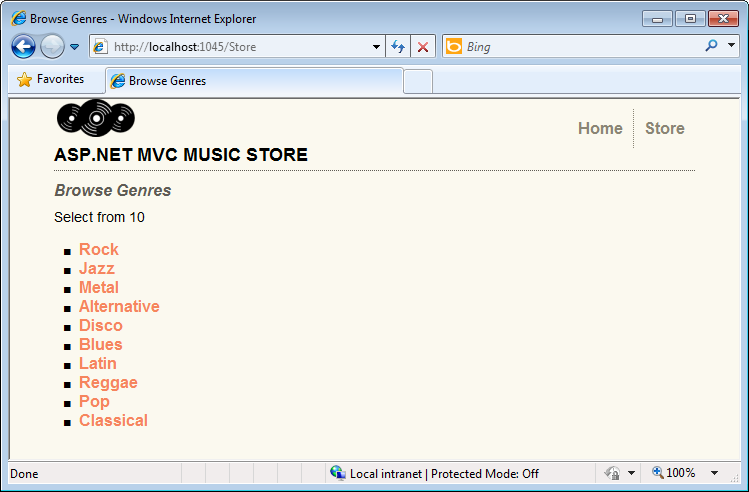
Task 4 – Truy vấn database

Trong task này, bạn sẽ cập nhật các class trong StoreController thay vì sử dụng các dữ liệu mã hóa cứng, nó sẽ truy vấn database để lấy ra các thông tin cần thiết.

1. Mở Controllers/StoreControllers.cs và thêm dòng này vào class chứa một thể hiện của class **MusicStoreEntities**, đó là storedDB:
   * 1. C#
     2. public class StoreController : Controller
     3. {
     4. **MusicStoreEntities storeDB = new MusicStoreEntities();**
2. Class **MusicStoreEntities** chứa những thuộc tính mỗi bảng của database. Cập nhật phương thức Index action của **StoreController** để lấy ra tất cả các tên **Genre** của database. Bây giờ bạn có thể viết câu truy vấn LINQ như sau để lấy ra thuộc tính **Name** của bảng Genre trong database:
   * 1. C#
     2. //
     3. // GET: /Store/
     4. public ActionResult Index()
     5. {
     6. **// Retrieve the list of genres**
     7. **var genres = from genre in storeDB.Genres**
     8. **select genre.Name;**
     9. // Create your view model
     10. }
3. Chuyển bộ thể loại (**genres**) vào một list. để làm điều này thay thế đoạn code sau đây:
   * 1. C#
     2. public ActionResult Index()
     3. {
     4. // Retrieve the list of genres
     5. var genres = from genre in storeDB.Genres
     6. select genre.Name;
     7. // Create your view model
     8. var viewModel = new StoreIndexViewModel
     9. {
     10. **Genres = genres.ToList(),**
     11. **NumberOfGenres = genres.Count()**
     12. };
     13. return View(viewModel);
     14. }

Task 5 – Chạy ứng dụng

Trong task này, bạn sẽ kiểm tra rằng trong Store Index sẽ hiển thị các thể loại (Genre) trong database. Không cần thay đổi gì trong View template vì **StoreController** trả về **StoreIndexViewModel** như trước mặc dù lúc này dữ liệu sẽ được lấy ra từ database.

1. Nhấn F5 để chạy ứng dụng
2. Project sẽ bắt đầu ở trang Home page. Thay đổi ULR thêm /Store để hiển thị danh sách các thể loại (**Genre**).
   * 1. ****
     2. Hình21
     3. Truy cập Genres database

Bài tập 2: Thêm database sử dụng các tiếp cận Code First

Trong bài tập này, bạn sẽ biết cách sử dụng cách tiếp cận **Code First** để thêm database với các bảng ứng dụng MusicStore để sử dụng dữ liệu của nó.

**Code First** được xây dựng dựa trên các thành phần được giới thiệu trong Entity Framewoork 4, **Code First** cho phép lập trình viên định nghĩa model với code thay vì phải sử dụng một tập tin **EDMX**.

Sau khi thêm và tạo ra các mô hình, bạn sẽ phải điều chỉnh thích hợp trong StoreController để cung cấp các View template với dữ liệu được lấy từ database thay vì mã hóa nó.

**Task 1 – Thêm một Database:** Tương tự như task 1 của bài tập 1

\*Chú ý: bài tập này là bài tập độc lập với bài tập 1.

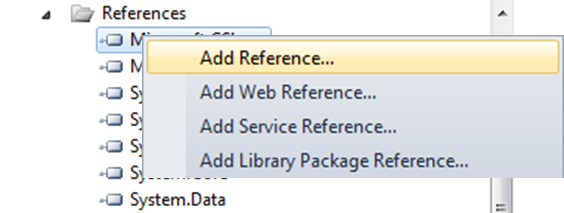
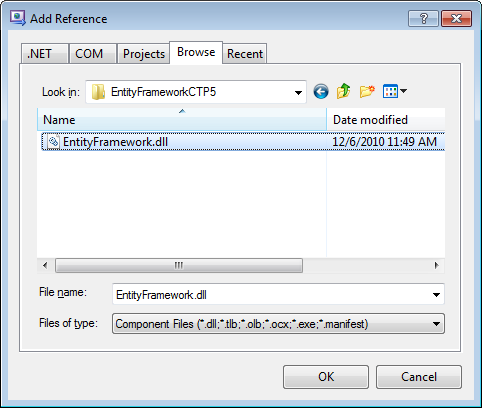
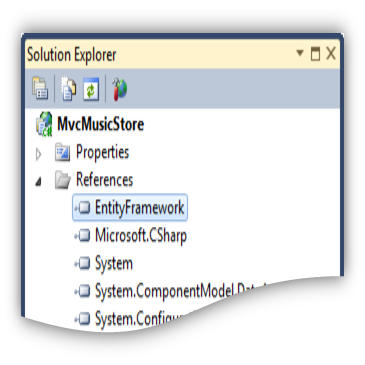
Task 2 – Chèn Code Firstvào Solution và kết nối đến Database

Bây giờ khi đã thêm database thành công vào project, chúng ta sẽ viết một chuỗi kết nối Web.config.

1. Viết chuỗi kết nối, Web.config trong trang gốc của project và chèn đoạn code sau:
   * 1. 
     2. Hình33
     3. Vị trí tập tin Web.config
     4. XML
     5. …
     6. **<connectionStrings>**

**<add name="MusicStoreEntities" connectionString="data source=.\SQLEXPRESS;initial catalog=MvcMusicStore;Integrated Security=SSPI;AttachDBFilename=|DataDirectory|\MvcMusicStore.mdf;User instance=true" providerName="System.Data.SqlClient" />**

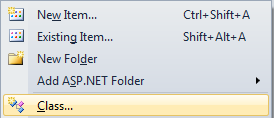
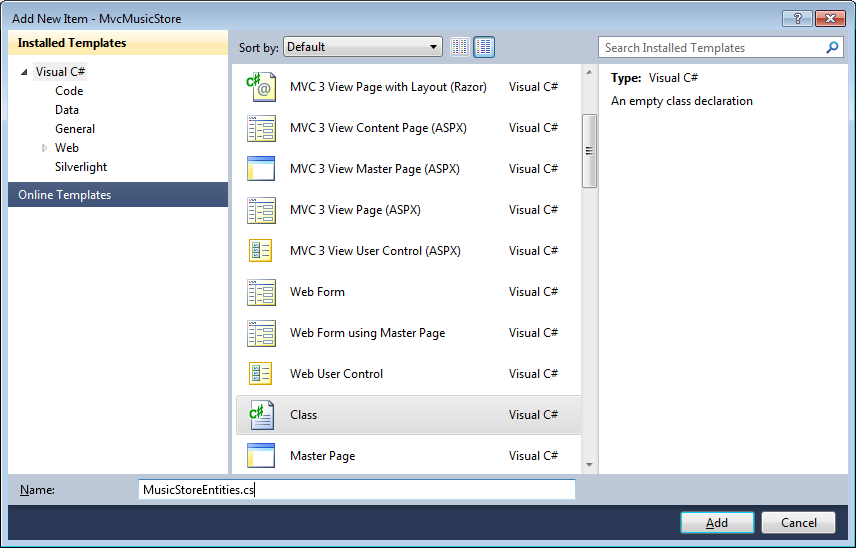
* + 1. **</connectionStrings>**
    2. </configuration>

1. Thêm mới một reference vào Code First assembly **EntityFramework.dll**. Để làm được click phải **Reference** chọn **Add Reference.**
   * 1. 
     2. Hình34
     3. Thêm một reference
2. Trong hộp thoại **Add Reference**, click **Browse** và chọn đến file **EntityFramework.dll**, sau đó click **OK**.
   * 1. 
     2. Hình35
     3. Thêm EntityFrameworkassembly
     4. 
     5. Hình36
     6. EntityFramework assemblyđã được thêm vào project

**Task 3 –Làm việc với Model**

Bây giờ khi đã thêm thư viện Code First và kết nối database, chúng ta sẽ liên kết mô hình (model) với các bảng của database. Trong task này, chúng ta sẽ tạo class để liên kết đến database với Code First, nhớ rằng chúng ta đã có lớp model POCO cần được sửa đổi.

Một trong những thuận lợi lớn của Code First đó là các class của bạn trở thành model, điều này có nghĩa rằng bất kì thay đổi nào cho model chỉ cần được thực hiện ở một nơi duy nhất – POCO class của bạn.

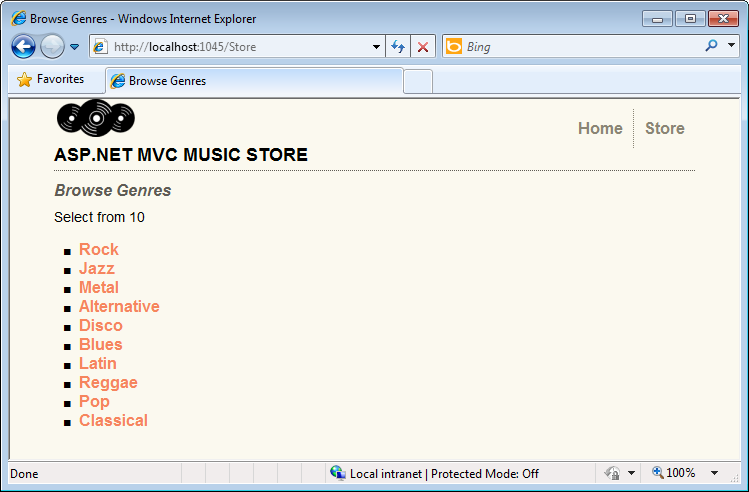
1. Mở class POCO model Genre từ thư mục của project /Models và chèn vào ID, mô tả thuộc tính và một bộ sưu tập album.
   * 1. C#
     2. using System;
     3. using System.Collections.Generic;
     4. using System.Linq;
     5. using System.Web;
     6. namespace MvcMusicStore.Models
     7. {
     8. public class Genre
     9. {
     10. **public int GenreId { get; set; }**
     11. public string Name { get; set; }
     12. **public string Description { get; set; }**
     13. **public virtual ICollection<Album> Albums { get; set; }**
     14. }
     15. }
2. Mở class POCO model **Album** từ thư mục của project /**Models** và chèn vào thuộc tính AlbumID và GenreID.
   * 1. C#
     2. using System;
     3. using System.Collections.Generic;
     4. using System.Linq;
     5. using System.Web;
     6. namespace MvcMusicStore.Models
     7. {
     8. public class Album
     9. {
     10. **public int AlbumId { get; set; }**
     11. public string Title { get; set; }
     12. **public int GenreId { get; set; }**
     13. public virtual Genre Genre { get; set; }
     14. }
     15. }
3. Click phải lên thư mục project /**Models** chọn **Add** và chọn **Class** để thêm class mới, đặt tên là **MusicStoreEntities.cs** và chọn **Add**.
   * 1. 
     2. Hình37
     3. Thêm một class
     4. 
     5. Hình38
     6. Thêm một class (2)
4. Mở class bạn vừa mới tạo, MusicStoreEntities.cs, và chèn System.Data.Entity
   * 1. C#
     2. using System;
     3. using System.Collections.Generic;
     4. using System.Linq;
     5. using System.Web;
     6. **using System.Data.Entity;**
     7. **using System.Data.Entity.Infrastructure;**
     8. …
5. Thêm đoạn code này vào:
   * 1. C#
     2. using System;
     3. using System.Collections.Generic;
     4. using System.Linq;
     5. using System.Web;
     6. using System.Data.Entity;
     7. using System.Data.Entity.Database;
     8. namespace MvcMusicStore.Models
     9. {
     10. **public class MusicStoreEntities : DbContext**
     11. **{**
     12. **public DbSet<Genre> Genres { get; set; }**
     13. **public DbSet<Album> Albums { get; set; }**
     14. **protected override void OnModelCreating(System.Data.Entity.DbModelBuilder modelBuilder)**
     15. **{**
     16. **modelBuilder.Conventions.Remove<IncludeMetadataConvention>();**
     17. **modelBuilder.Entity<Genre>().ToTable("Genre");**
     18. **modelBuilder.Entity<Album>().ToTable("Album");**
     19. **base.OnModelCreating(modelBuilder);**
     20. **}**
     21. **}**
     22. }
     23. **Chú ý:** Với Entity Framework **DbContext** và **DBSet** bạn có thể truy vấn class POCO Genre.
     24. Bằng cách override OnModelCreating, chúng ta đặc tả mã để Genre ánh xạ tới bảng cơ sở dữ liệu.

Task 4– Truy vấn Database: Tượng tự như task 4 của bài tập 1

Trong task này, bạn sẽ cập nhật các class trong StoreController thay vì sử dụng các dữ liệu mã hóa cứng, nó sẽ truy vấn database để lấy ra các thông tin cần thiết.

1. Mở Controllers/StoreControllers.cs và thêm dòng này vào class chứa một thể hiện của class **MusicStoreEntities**, đó là storedDB:
   * 1. C#
     2. public class StoreController : Controller
     3. {
     4. **MusicStoreEntities storeDB = new MusicStoreEntities();**
2. Class **MusicStoreEntities** chứa những thuộc tính mỗi bảng của database. Cập nhật phương thức Index action của **StoreController** để lấy ra tất cả các tên **Genre** của database. Bây giờ bạn có thể viết câu truy vấn LINQ như sau để lấy ra thuộc tính **Name** của bảng Genre trong database:
   * 1. C#
     2. //
     3. // GET: /Store/
     4. public ActionResult Index()
     5. {
     6. **// Retrieve the list of genres**
     7. **var genres = from genre in storeDB.Genres select genre.Name;**
     8. // Create your view model
     9. }
3. Chuyển bộ thể loại (**genres**) vào một list. để làm điều này thay thế đoạn code sau đây:
   * 1. C#
     2. public ActionResult Index()
     3. {
     4. // Retrieve the list of genres
     5. var genres = from genre in storeDB.Genres select genre.Name;
     6. // Create your view model
     7. var viewModel = new StoreIndexViewModel
     8. {
     9. **Genres = genres.ToList(),**
     10. **NumberOfGenres = genres.Count()**
     11. };
     12. return View(viewModel);
     13. }

Task 5 –Chạy ứng dụng

1. Nhấn F5 để chạy ứng dụng
2. Project sẽ bắt đầu ở trang Home page. Thay đổi ULR thêm /Store để hiển thị danh sách các thể loại (**Genre**).
   * 1. ****
     2. Hình39
     3. Truy cập Genres từ cơ sử dữ liệu

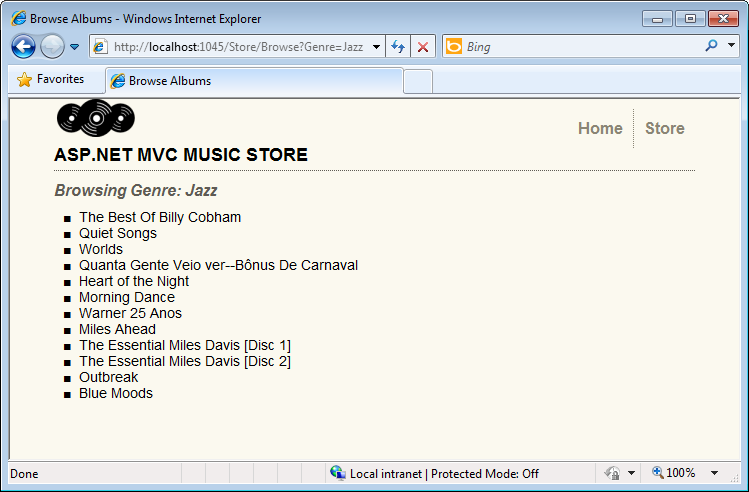
Bài tập 3: Truy vấn database với các tham số

1. Trong bài tập này, bạn sẽ tìm hiểu cách truy vấn database sử dụng tham số và cách sử dụng Query Result Shaping, một tính năng làm giảm số lượng truy cập đến database mà vẫn lấy được dữ liệu hiệu quả hơn.

Task 1 – Sửa StoreController để lấy Albums từ database:

* 1. Trong Task này, bạn sẽ đổi class StoreController để truy cập database và lấy albums từ một thể loại (genre) cụ thể.
  2. Mở **Microsoft Visual Studio 2010**.
  3. Trong menu **File**, chọn **Open Project**. Chọn **MvcMusicStore.sln** (đã làm từ bài tập 1 hoặc bài tập 2) và click **Open**.
  4. Mở class **StoreController** để đổi phương thức **Browse** action. Để làm được, trong **Solution Explorer**, mở thư mục **Controllers** và nhấp đúp **StoreController.cs**.
  5. Đổi phương thức **Browse** action để lấy albums cho thể loại (genre). Để làm được, thay thế đoạn code sau:
     1. C#
     2. //
     3. // GET: /Store/Browse?genre=Disco
     4. public ActionResult Browse(string genre)
     5. {
     6. **// Retrieve Genre and its Associated Albums from database**
     7. **var genreModel = storeDB.Genres.Include("Albums")**
     8. **.Single(g => g.Name == genre);**
     9. **var viewModel = new StoreBrowseViewModel()**
     10. **{**
     11. **Genre = genreModel,**
     12. **Albums = genreModel.Albums.ToList()**
     13. **};**
     14. return View(viewModel);
     15. }

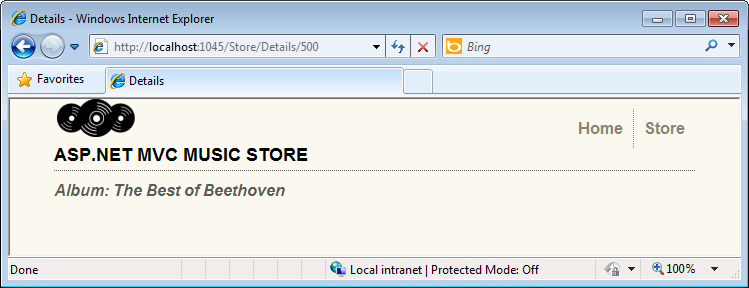
Task 2 – Chạy ứng dụng:

* 1. Trong task này, bạn sẽ chạy thử ứng dụng trên trình duyệt web và lấy được thông tin albums cho những thể loại (genre) cụ thể trong database.
  2. Nhấp **F5** để chạy ứng dụng.
  3. Project sẽ chạy ở trang Home. Thay đổi URL thành **/Store/Browse?genre=Jazz** để thấy kết quả được lấy từ database.
     1. 
     2. Hình40
     3. Browsing /Store/Browse?genre=Jazz

Task 3 – Truy cập Albums từ ID

* 1. Trong task này, bạn sẽ lặp lại bước trước, trong trường hợp này, lấy được albums bằng ID.
  2. Tắt trình duyệt, quay lại Visual Studio. Mở class **StoreController** để thay đổi phương thức **Details** action. Để làm được, trong **Solution Explorer**, mở thư mục **Controllers** và nhấp đúp **StoreController.cs**.
  3. Đổi phương thức **Details** action để lấy chi tiết albums dựa vào **Id** của nó. Để làm vậy, thay thế đoạn code sau:
     1. C#
     2. //
     3. // GET: /Store/Details/5
     4. public ActionResult Details(int id)
     5. {
     6. **var album = storeDB.Albums.Single(a => a.AlbumId == id);**
     7. return View(album);
     8. }

Task 4 – Chạy ứng dụng:

* 1. Trong task này, bạn sẽ thử chạy ứng dụng trên trình duyệt và lấy chi tiết albums dựa vào ID của nó.
  2. Nhấn **F5** để chạy ứng dụng.
  3. Project sẽ chạy ở trang Home. Thay đổi URL thành **/Store/Details/500** để thấy thong tin được lấy lên từ database.
     1. 
     2. Hình41
     3. Browsing /Store/Details/500