Exercises

By: Mosh Hamedani

Building a Model using Database First Workflow

Your job is to build an application for a video rental store called Vidzy. For the purpose of this course, you don't need to worry about user interface, so you'll be doing all these exercises in a console application.

1st Iteration

You attempt to build the Vidzy app in an iterative way. In the first iteration, you want to implement the ability to add videos in the database.

You start by building your database first. So, create a new database called **Vidzy**. Download the database script in the supplementary materials of this lecture and run it on the **Vidzy** database to create the initial tables: **Videos, Genres** and **VideoGenres**. (Note that here we are assuming there is a many-to-many relationship between videos and genres.)

Once your database is ready, build an entity data model using database-first workflow. Bring all the tables and **spAddVideo** stored procedure. We don't prefix method names with **sp** in C#. So, change the name of the function import to **AddVideo.**

Use the **AddVideo** method to add a few videos to the database.

2nd Iteration

You realize that you over-engineered the solution and each Video needs one and only one genre.

So, add a new NULLABLE **tinyint** column to **Videos** table called **GenreId.** Edit the records in the table and assign them a genre.

Open the **Videos** table in the table designer again, remove the nullable attribute from the **GenreId** column and create a relationship between **Genres** and **Videos.**

Next, drop the **VideoGenres** table.

Finally, modify the **spAddVideo** stored procedure accordingly so you can still add new videos.

Bring the changes into your model. Note that there are two relationships between **Genre** and **Video:** One many-to-many relationship (from before) and a new one-to-many relationship because of the recent changes. So, as you see, sometimes refreshing your model doesn't quite work the way you expect. In this case, the old relationship was not deleted, so you need to manually delete it.

Delete the many-to-many relationship and validate your model. Save the changes and re-build the project. Ensure you can still add videos to the database in your console application.

3rd Iteration

Vidzy tells you that they need to classify their videos into three categories: Silver, Gold and Platinum. You decide to implement this by adding a new **tinyint** column in the **Videos** table, called **Classification**. Make sure this column is not nullable and use 1 as the default value to assign all existing videos to the silver category.

Modify the stored procedure accordingly.

Bring the changes into your model, and map the **Classification** property to a new enum called: **Classification**. This enum should have three members: Silver, Gold and Platinum.

Ensure you can still add new videos using your console application.

Solution

To see my solution, download the solution zip file. Temporarily rename your database to **Vidzy_Exercise**. Create a new database called Vidzy and run the SQL script included in the solution zip file on this database. This would be the database after all changes in these iterations. Now you can open the console application in Visual Studio and run the code.