Modeling Data Exercise

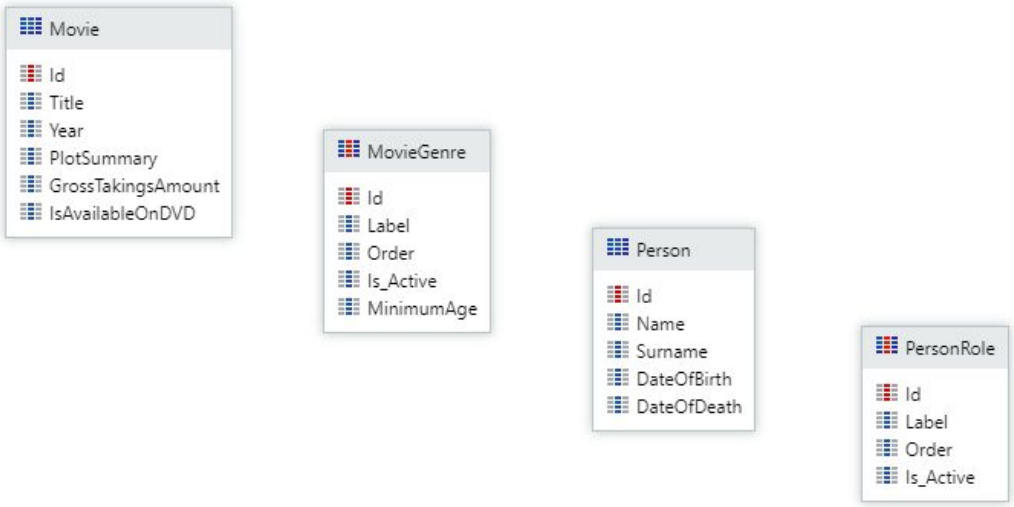


Table of Contents

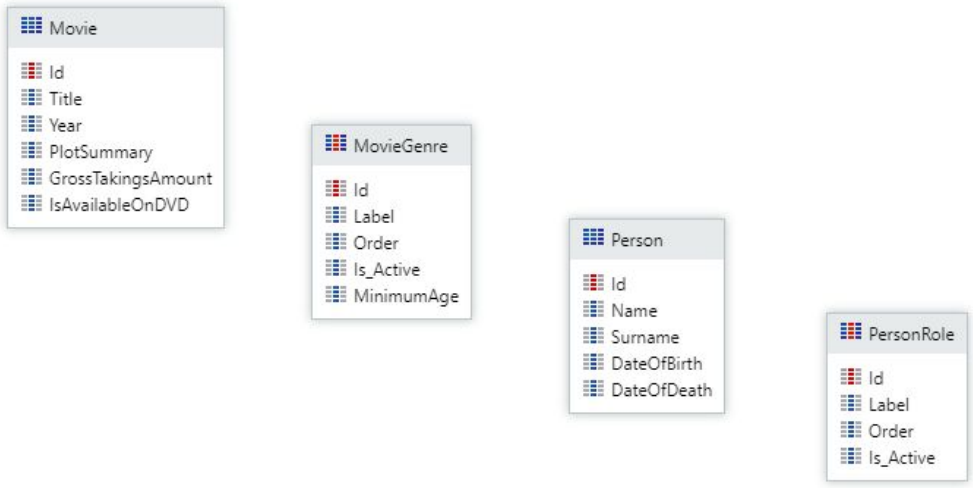
Introduction

In this exercise lab, we already have the OSMDb application created, so we will start to progressively build our app, by creating the data model.

The data model of this application will exclusively be created on the OSMDb_Core module, and will consist at this stage on two Entities, Movie and Person, and on two Static Entities, MovieGenre and PersonRole.

These Entities will represent the movies (Movie) in the database and their genres (MovieGenre), as well as the cast and crew (People) and their role in the movies (PersonRole).

At the end of the Lab, we should have our data model looking like this



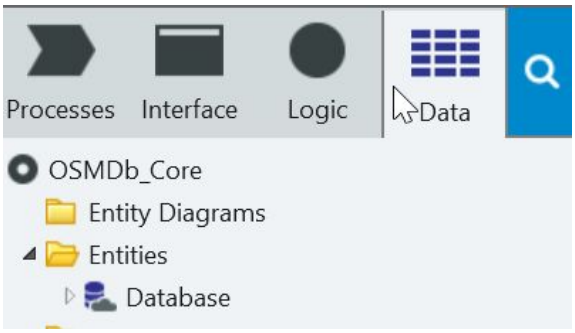
Create the Movie and Person Entities

In this section, we will create the initial Entities of the application: **Movie** and **Person**. An Entity in OutSystems requires a **Name**, an **Id** (identifier) attribute and at least one other attribute. Entities can be initialized with data from an Excel spreadsheet. This process is called Bootstrapping.

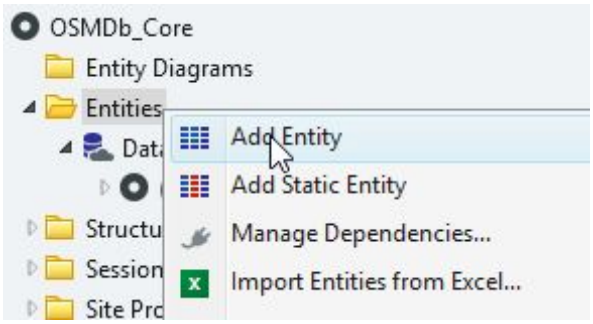
Since our application will have two modules, and the UI module will reference these Entities, they will be defined as **Public**.

1) Create the **Movie** Entity in the OSMDb_Core module, with the following attributes: *Title* (Text and mandatory), *Year* (Integer and mandatory), *PlotSummary* (Text, with max. length of 500 characters), *GrossTakingsAmount* (Currency) and *IsAvailableOnDVD* (Boolean). The Entity should be **Public** and **Exposed with write permissions**.

   a) Click the Data tab in the upper right corner of the workspace, to switch to the Data elements.

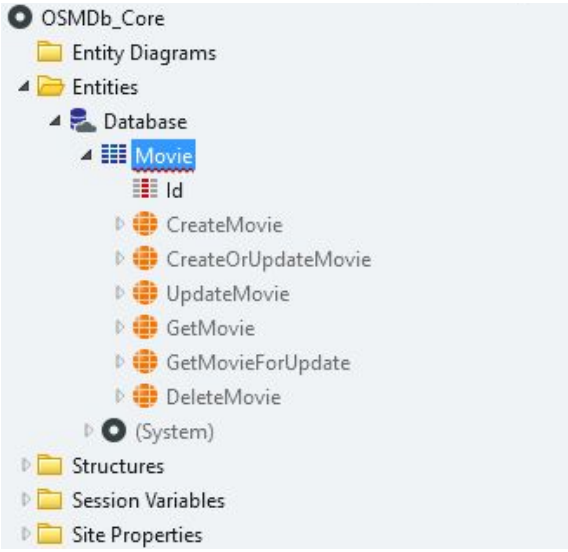


   b) In the elements area, right-click the **Entities** folder and select **Add Entity.**



   c) Type *Movie* for the **Name** of the Entity.

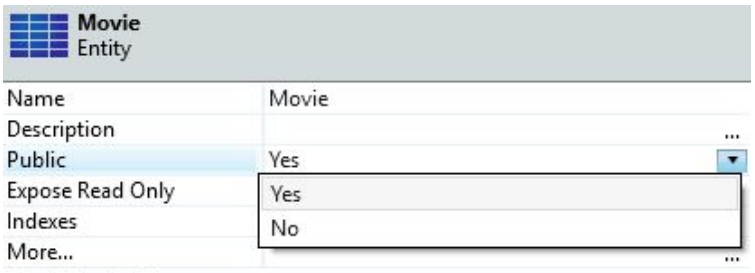
d) Expand the Movie Entity. There is an auto-number **Id** attribute and six Entity Actions to provide typical Create, Read, Update and Delete (CRUD) functionality.



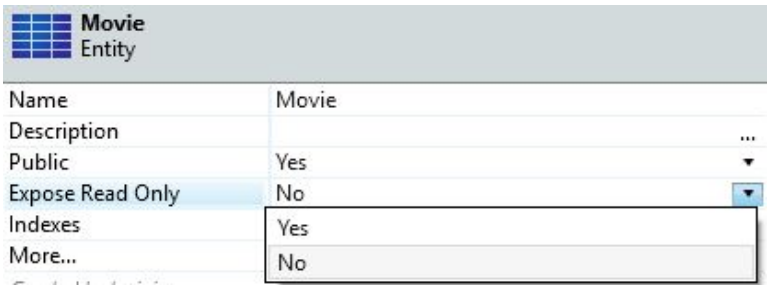
NOTE: Notice that the Entity is underlined in red and that the **TrueChange** tab has a red X, meaning that the module has an error. Entities cannot be made up of a single Auto-Numbered attribute.

---

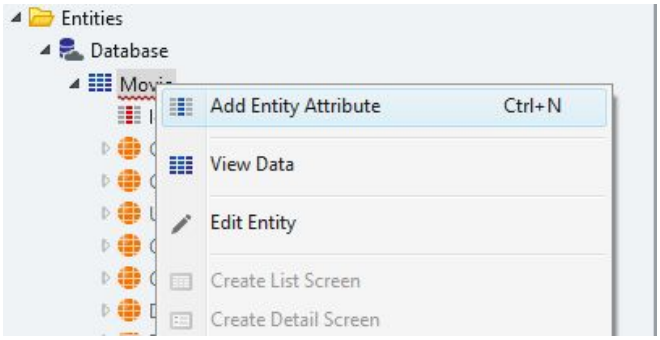
e) Since we will be using this Entity on our UI module, in the properties editor at the bottom right, change the **Public** property to *Yes.*



f) Still in the properties of the Entity, set the **Expose Readonly** to *No*, to make the Entity exposed with write permissions.

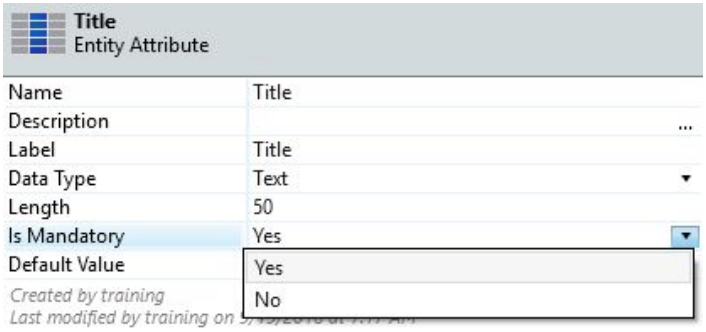


g) Right-click the Movie Entity and select **Add Entity Attribute.**



h) Enter *Title* for the **Name** of the attribute. Notice that the errors disappear.

i) Change the **Is Mandatory** property to *Yes.*

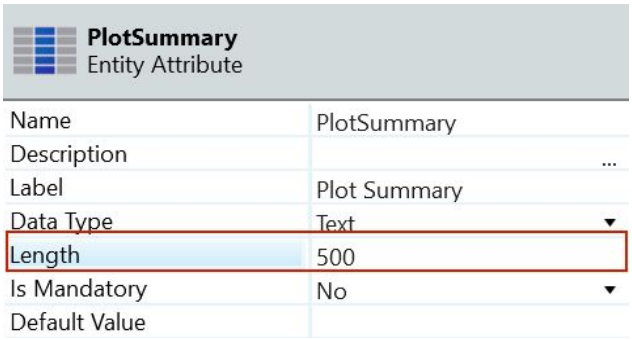


j) Right-click the Movie Entity and select **Add Entity Attribute.**

k) Enter *Year* for the Name of the attribute. Notice the default **Data Type** is *Integer*. Make this attribute **Mandatory**.

l) Right-click the Movie Entity and select **Add Entity Attribute**.

m) Enter *PlotSummary* for the **Name** of the attribute. Notice the default **Data Type** is *Text* with a **Length** of *50* characters. Change it to *500*.



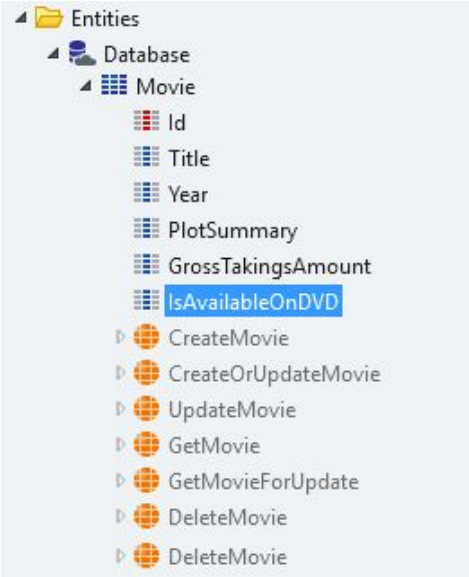
n) Add 2 more attributes: *GrossTakingsAmount* and *IsAvailableOnDVD*. Notice the default **Data Types** are correctly set to *Currency* and *Boolean*.

NOTE: OutSystems infers the data type of an attribute or variable from their name. Since *GrossTakinsAmount* has **Amount** in the name, OutSystems automatically infers it to be a *Currency* attribute. The same with *IsAvailableOnDVD* being *Boolean*, since the attribute's name starts with **Is**.

---

o) Select the GrossTakingsAmount attribute and change its Label property to **Gross Takings**. This will shorten the default label used in OutSystems, when generating UI, as we will see in a later Lab.

p) The Movie Entity should look like this



2) Create a new **Person** Entity to save all the people involved in the movies of our application. The Entity has the following attributes: *Name* (Text and mandatory), *Surname* (Text and mandatory), *DateOfBirth* (Date and mandatory) and *DateOfDeath* (Date). Set the Entity to **Public** and **exposed with write permissions**. Also, change the **plural label** of the Entity to *People*.

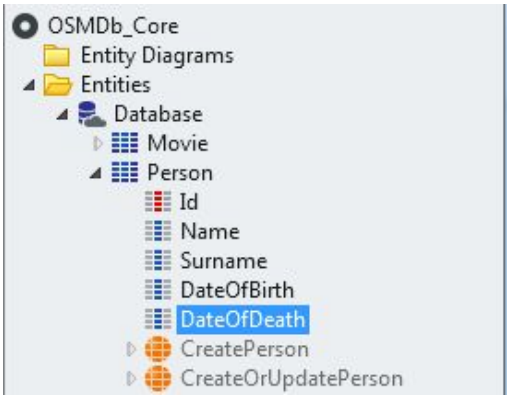
   a) On the Data tab, create a new **Entity** and set its name to *Person.*

   b) In the properties editor at the bottom right, change the **Public** property to *Yes* and the **Expose Read Only** to *No.*

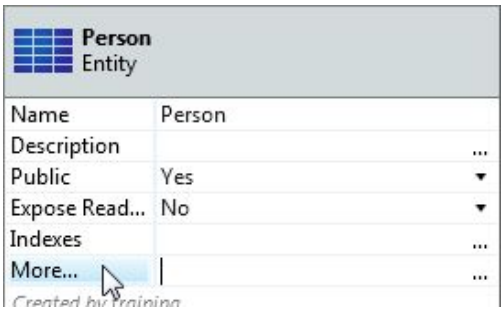
   c) Add the following attributes to the Entity: *Name*, *Surname*, *DateOfBirth,* and *DateOfDeath.* Did Service Studio guessed the Data Types correctly from the attribute name? **Always double-check!**

   d) Make Name, Surname, and DateOfBirth **mandatory**.

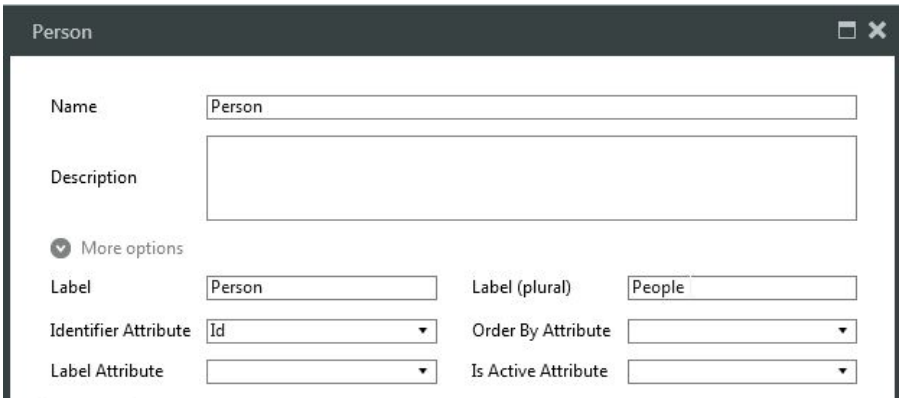
e) The Person Entity should look like this



f) Select the Person Entity and in the properties editor, double-click the **More...** property to launch the **Entity Editor**.



g) In the Entity Editor dialog, expand the **More Options** section and change the **Label (plural)** property to *People* and click the **Close** button.

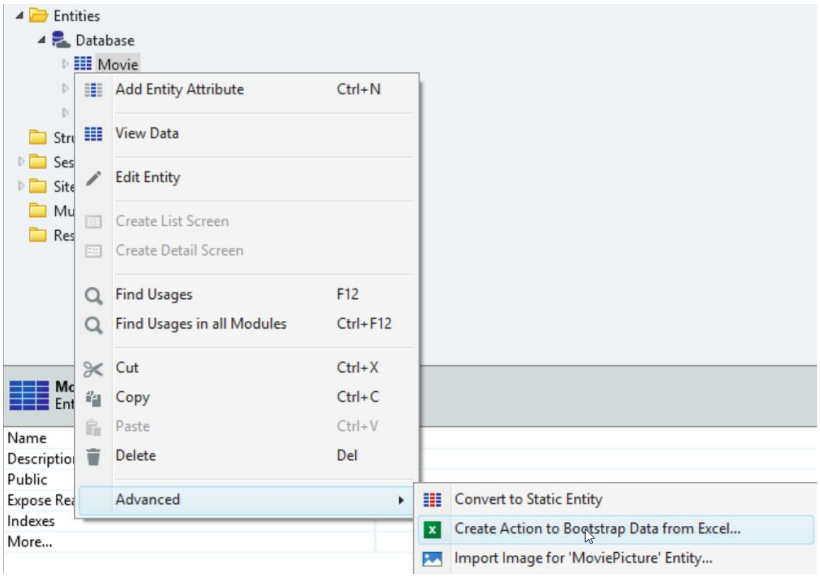


NOTE: The Entity Editor allows us to configure more advanced settings relating to your Entity, including the Labels which are used for the UI defaults while building the Screens.

Here, we are simply setting the correct plural for the Entity name **Person**, which unlike most English words, it is not simply a matter of adding an 's' at the end. This will be used whenever Service Studio needs to suggest the default name of any module element, or operation, that involves multiple instances of Person. It will also save manual adjustments later, if we want to keep grammar correctness.

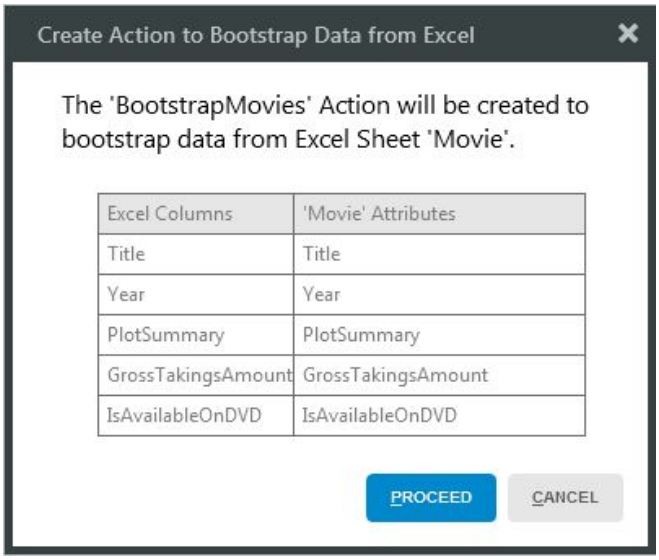
3) **Bootstrap** some seed data for the two newly created Entities, from two Excel files. To do that, we will use the **Movies.xlsx** and **People.xlsx** respectively, available in the Resources folder.

   a) Right-click the Movie Entity, select **Advanced**, and then the option **Create Action to Bootstrap Data from Excel...**



   b) Browse to the Resources folder and select the **Movies.xlsx** Excel file.

c) In the "Create Action to Bootstrap Data from Excel" window, ensure that the column headers names from the Movies.xlsx excel file (**Excel Columns**) match the names of the Movie Entity attributes (**'Movie Attributes'**). If so, click on the Proceed button. Otherwise, click on the Cancel button and fix the names, or data types, of the mismatched Movie Entity attributes. Then, redo the third step again. Your Entity attributes **must match in name and type** what is in the Excel file.

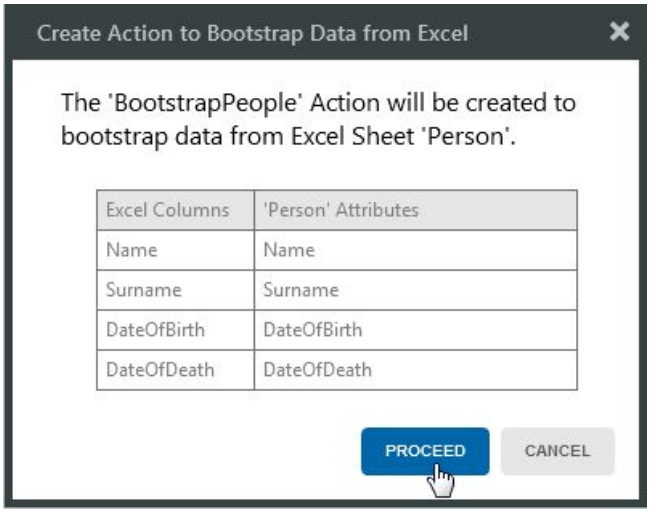


NOTE: Stars will appear in the interface when the previous step is completed. The stars indicate the areas where elements are being created.

The logic for fetching the data from the Excel file and add it to the database is created in the Action **BootstrapMovies,** under the Logic tab. It checks if any Movies currently exist. If not, it imports the Movies from the Excel spreadsheet and creates a Movie in the database, for each row in the spreadsheet. The Excel file will be saved inside the module, in the Resources folder under the Data tab. This Action runs when the module is published.

---

d) Follow the same process for the **Person** Entity, choosing the **People.xlsx** Excel file. Don't forget to verify that the values in **Excel Columns** match the values in **'Person' Attributes** before you click Proceed.

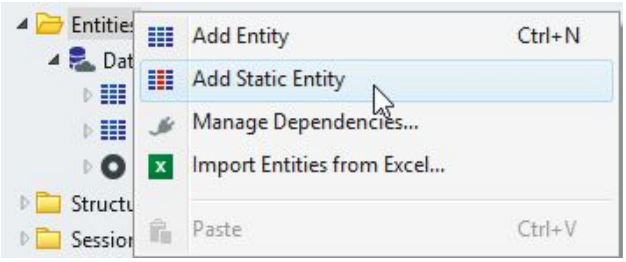


Create the MovieGenre and PersonRole Static Entities

After creating the Entities, we will create two Static Entities: **MovieGenre** and **PersonRole**. A Static Entity is initially created with a few attributes: **Label**, **Order** and **Is_Active**, but other attributes can be added.

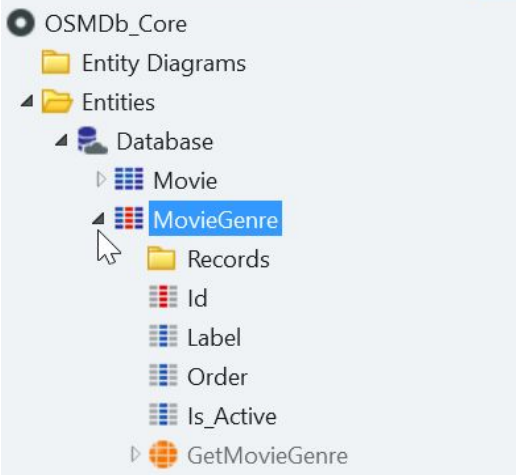
Unlike regular Entities, the Static Entity data is defined and initialized at design time. Each of the possible values of a Static Entity is called a Record.

1) Create the **MovieGenre** Static Entity to save the different genres of a movie. Add a new attribute for the minimum age for that genre of movies: *MinimumAge* (Integer). Add four records to the Entity: *Comedy*, *Action*, *Drama* and *Horror*. Set the appropriate minimum age for each record (6, 12, 16 and 18) and make the Entity **public**.
    a) Switch to the Data tab in Service Studio.
    b) In the elements area, right-click the **Entities** folder and select **Add Static Entity**.



    c) Enter *MovieGenre* for the **Name** of the Static Entity.

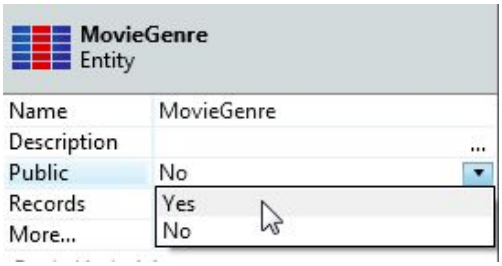
d) Expand the **MovieGenre** Entity by clicking on the expand symbol.



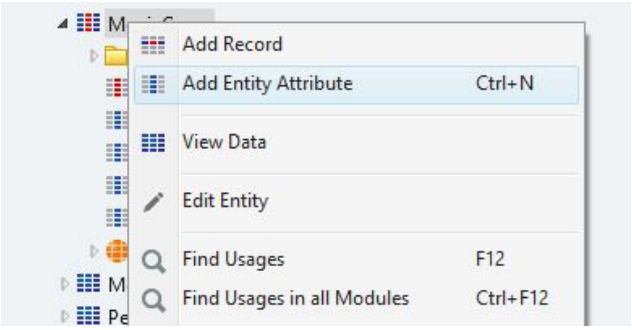
NOTE: Notice that the Static Entity only has one Entity Action: **Get<StaticEntity>**. Since it is not possible to dynamically create or update the Static Entity records in runtime, the Create and Delete Actions are not available.

---

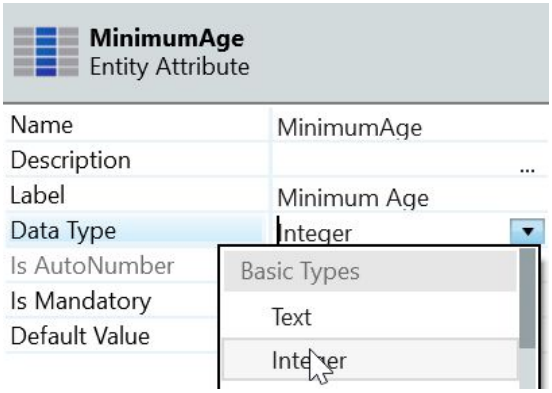
e) Since we will be using this Static Entity on our UI module, change its **Public** property to *Yes*.



f) Right-click the MovieGenre Entity and select **Add Entity Attribute**.

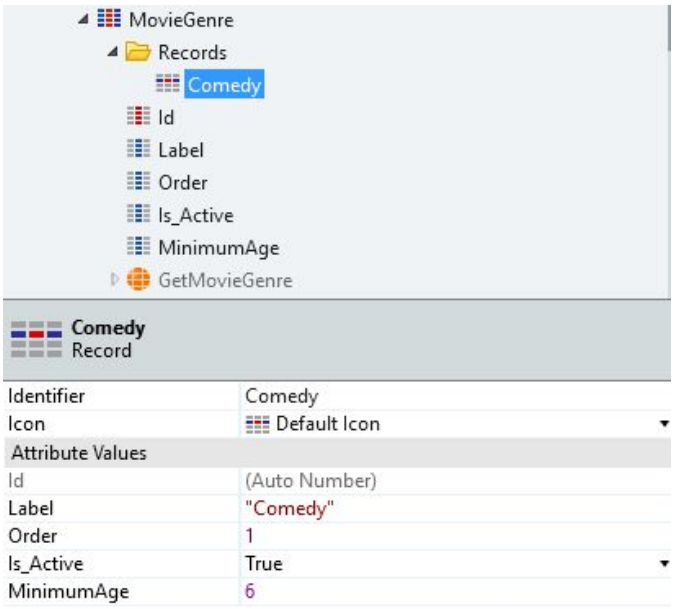


g) Set its **Name** to *MinimumAge* and change the **Data Type** to *Integer*.



h) Right-click the MovieGenre Static Entity and select **Add Record**.

i) Enter *Comedy* for the identifier of the Record. In the properties editor, set the value 6 in the **MinimumAge** attribute this Record.



j) Add 3 more Records: *Action*, *Drama* and *Horror*. Set these Records' **MinimumAge** to *12*, *16* and *18*, respectively.

k) The Static Entity should look like this

2) Create the **PersonRole** Static Entity, to represent the different roles a person can have in a movie. These roles can be: *Director*, *Producer*, *Actor* and *Crew*. Make the Entity **public**.

a) Create a new Static Entity and set its **Name** to *PersonRole*.

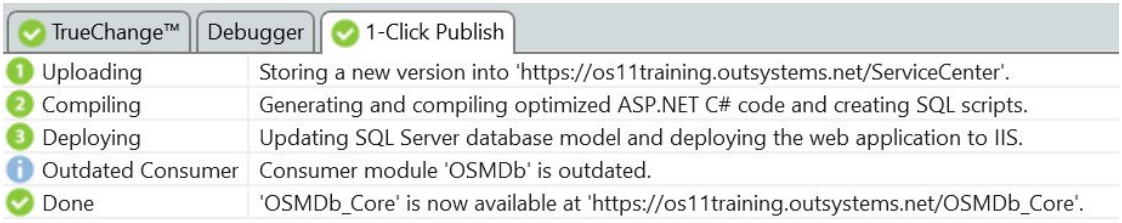
b) In the properties at the bottom right, change the **Public** property to *Yes*.

c) Add four Records to the Static Entity: *Director*, *Producer*, *Actor* and *Crew*.

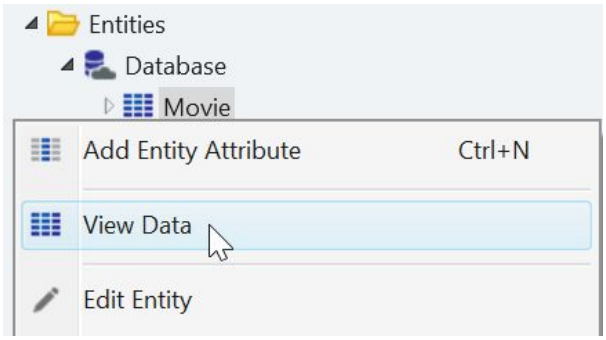
d) The Static Entity should look like this

e) Click the **1-Click Publish** button to publish the module.

f) Make sure the module was published successfully, with a message indicating that the OSMDb module is outdated. This will be addressed in the next Lab.



g) After a few seconds, right-click on the **Movie** or **Person** Entity, select *View Data*, and make sure that the data is previewed properly.



End of Lab

In this exercise, we created an initial data model for the movie database web application, in its Core module. The main application concepts are **Movie** and **Person** (member of a movie's cast and crew). These concepts were represented as Entities, which will correspond to a table in the database.

In preparation for more advanced modeling of these Entities and their relationships, we also created two Static Entities: **MovieGenre** and **PersonRole**. The Static Entities can only be changed at design time.

The application module was then published to the server, thus creating the appropriate database tables for these Entities. We will start visualizing these Entities, and their data, in the next lab.