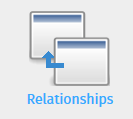
|  |
| --- |
| DEFINING RELATIONSHIPS |

Some drop-down lists can be approached via predefined lists as their contents are constant and never change. Some, however, are dependent on the contents of another database field, as is the case with our system’s Character Class and Race tables, which become the lookup tables for our Character’s Class and Race drop-down fields respectively. For this we will utilize another feature of the Cobalt code generator that focuses on building the relationships between database tables.

**DEFINING A NEW RELATIONSHIP**

Navigate to the “Relationships” option in the Cobalt Control Center.

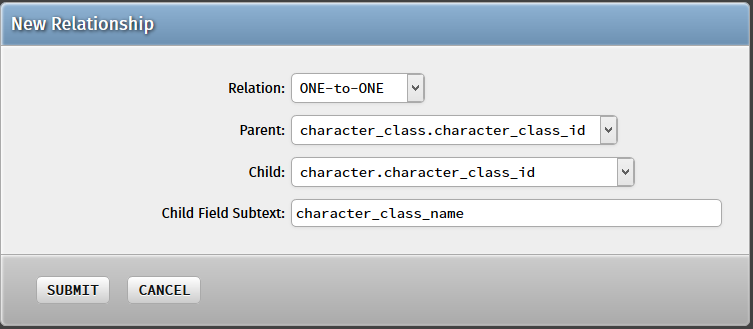


Hit “Define New Relationship to start creating one.

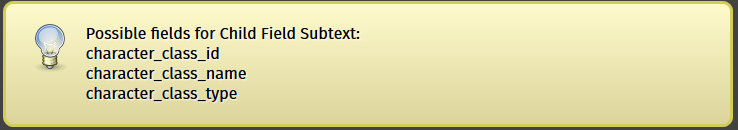


You will be taken to the “New Relationship” page, where you can define the relationship between two tables. You can define two types of relationships – ONE-to-ONE and ONE-to-MANY relationships.

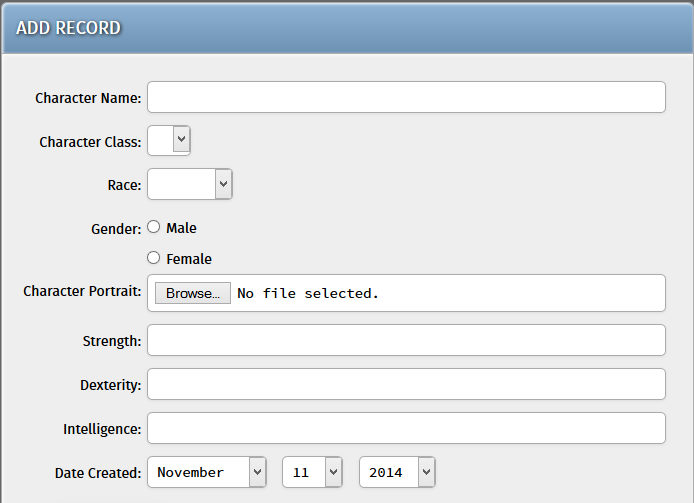
The character table has a ONE-to-ONE relationship with the character\_class and race tables. As a rule, the **parent** table is the table providing the information and the **child** table is the table asking for the information. Make sure their foreign key fields match according to our database design. As such, defining a relationship between the character and character\_class tables goes like so:



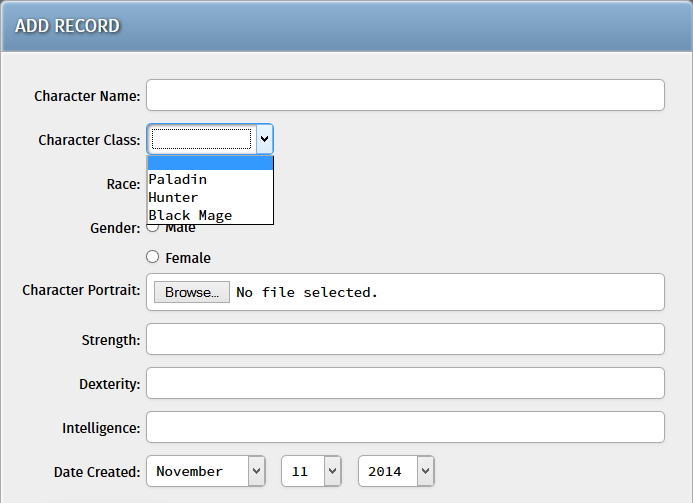
You will notice that as you select the “Parent” foreign key field, a help box will appear under the form that shows you possible fields to use for the “Child Field Subtext”. Entering any of the shown fields here will render that field’s value as the associated value with that foreign key relationship.



Re-generate the project as always and update your web root. Navigate to the “Character” module and try adding a new record this time. You will find that the Character Class field has totally changed.



It’s a drop-down list now, but it’s empty! This just means that you need to add new Character Classes in the “Character Class” module for it to show data. Try adding a few Character Classes and navigate back to this page.



Here we added three new character classes – Paladin, Hunter, and Black Mage. Reloading the Add Character page updates the drop-down list with these three values.

**MORE ONE-TO-ONE RELATIONSHIPS**

Before proceeding, you might want to add ONE-to-ONE relationships for the other table relationships as well. If you ever get confused, look back to your MySQL Workbench visual schema – each relationship line that binds two tables together is one relationship entry for Cobalt.

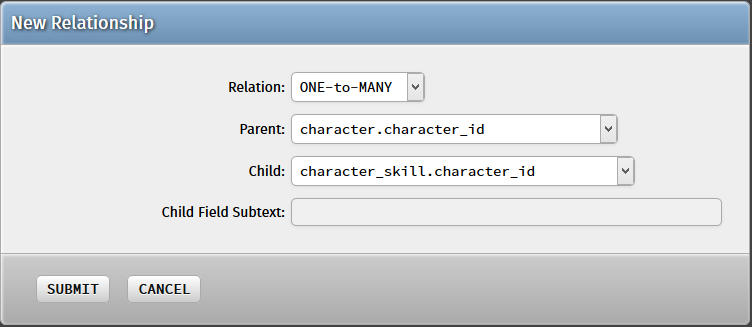
Define ONE-to-ONE relationships for the following table relationships (Parent => Child):

* race => character
* character => character\_skill
* skill => character\_skill

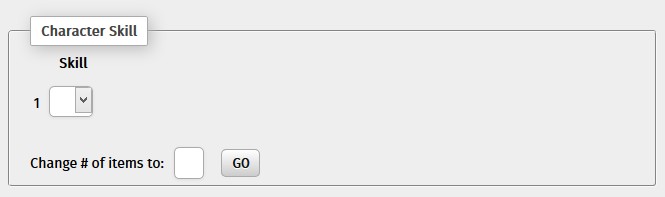
**ONE-TO-MANY RELATIONSHIPS**

ONE-to-ONE relationships in Cobalt are pretty straightforward, and ONE-to-MANY relationships are slightly simpler. In the case of our system, the character table has many character\_skill entries associated with it, so that is a ONE-to-MANY relationship for Cobalt.

Define a New Relationship again and this time set it as a “ONE-to-MANY” relationship. This time, the **parent** is the main table entry and the **child** is the “many” associated table entries with it. There is no Child Field Subtext required for this one and is disabled for this context.

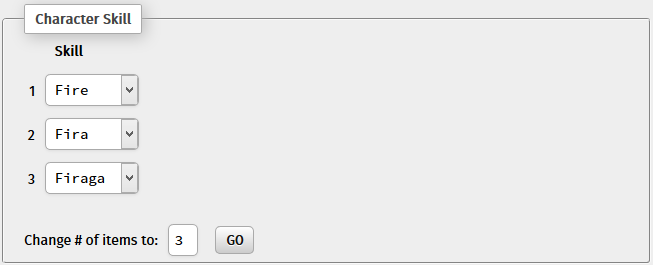


Hit “Submit” and regenerate the project once again. Navigate back to the Add Character page and you will find that a new section has been added to the bottom of the form.



This indicates that a successful ONE-to-MANY relationship has been established between the two tables. Entering a number in the “Change # of items to” field changes the number of skills you can enter for this Character (but of course you would need to add some Skills first!)

Putting in some skills on our “Skill” module, it should now look something like this:



With that last bit of information in hand, you can hit “Submit” and your character will be successfully created by the system! (Provided you have first entered the other relevant information above, however).