MIRACLE
SOFTWARE SYSTEMS

Relations B/N Models in API Connect

Analogous concept of TABLE JOINS in API-Connect

GeetaKrishna.Adhikari
API Developer
Miracle Software Systems, Inc.

AIM:

In reality, models are often connected or related. When you build a real-world application with multiple models, you'll typically need to define *relations* between models.

You'll find this process analogous to **Joins** in Databases.

With connected models, LoopBack exposes as a set of APIs to interact with each of the model instances and query and filter the information based on the client's needs.

PROCEDURE:

- Create a loopback application.
- Create two Datasources and also 2 Models for each of them.
- To create Relation between 2 existing models ;
- Run the Application and note down all the available Operations. (for future correspondence)

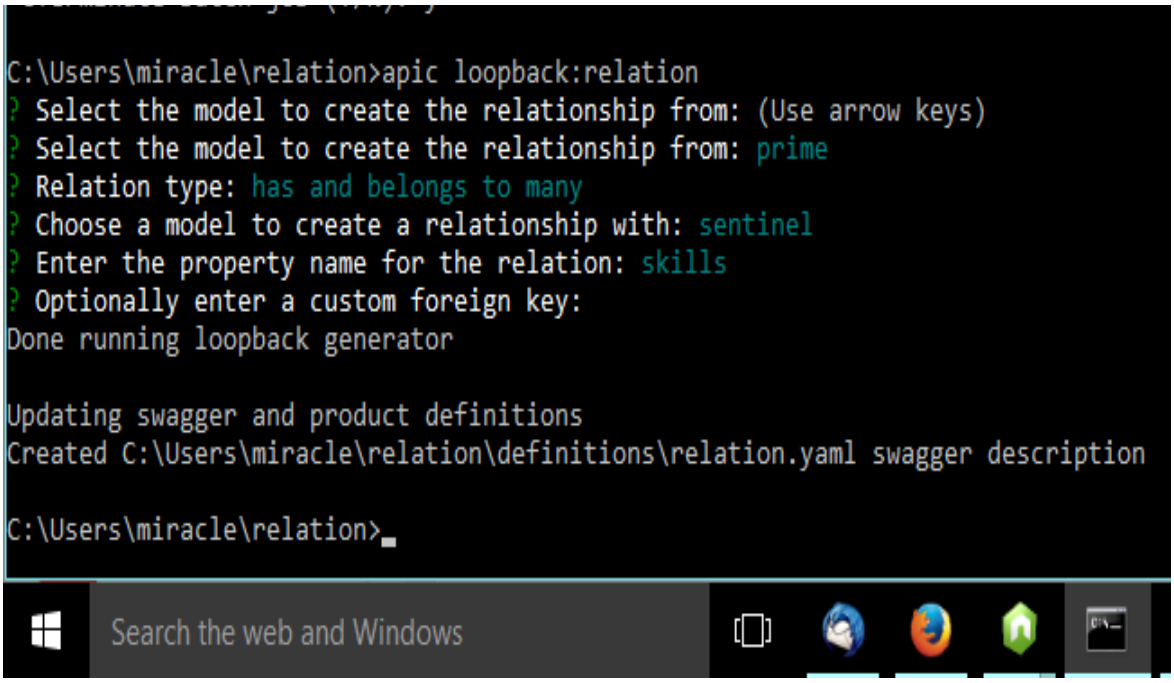
Run the Command: **apic loopback:relation**

Follow the prompts and select the Model from which you want to create relation.
You'll be prompted to select the type of relation you want the Model to have.

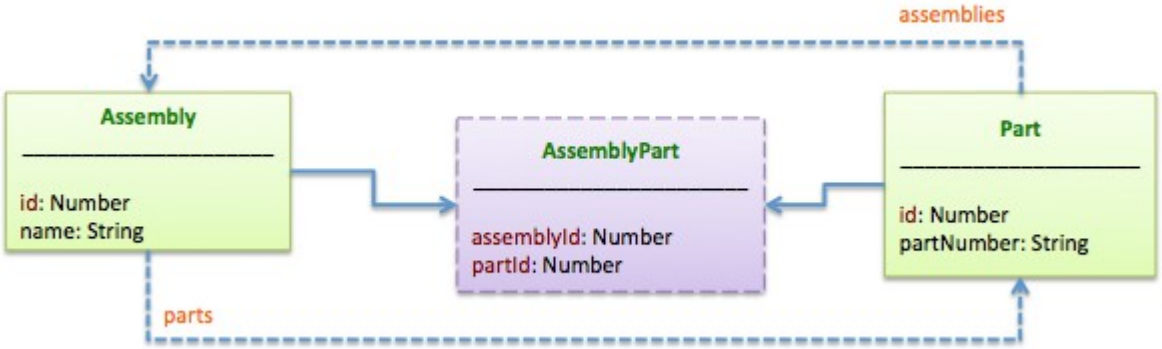
- Belongs to
- Has One
- Has Many
- Has and Belongs to many

Has Many and **Has and Belongs to many** works the same, but Has Many needs a Mediator Model to relate 2 models.

Whereas **Has and Belongs to many** also needs a Mediator Model but generates the Mediator model based on the 2 models that are being related.

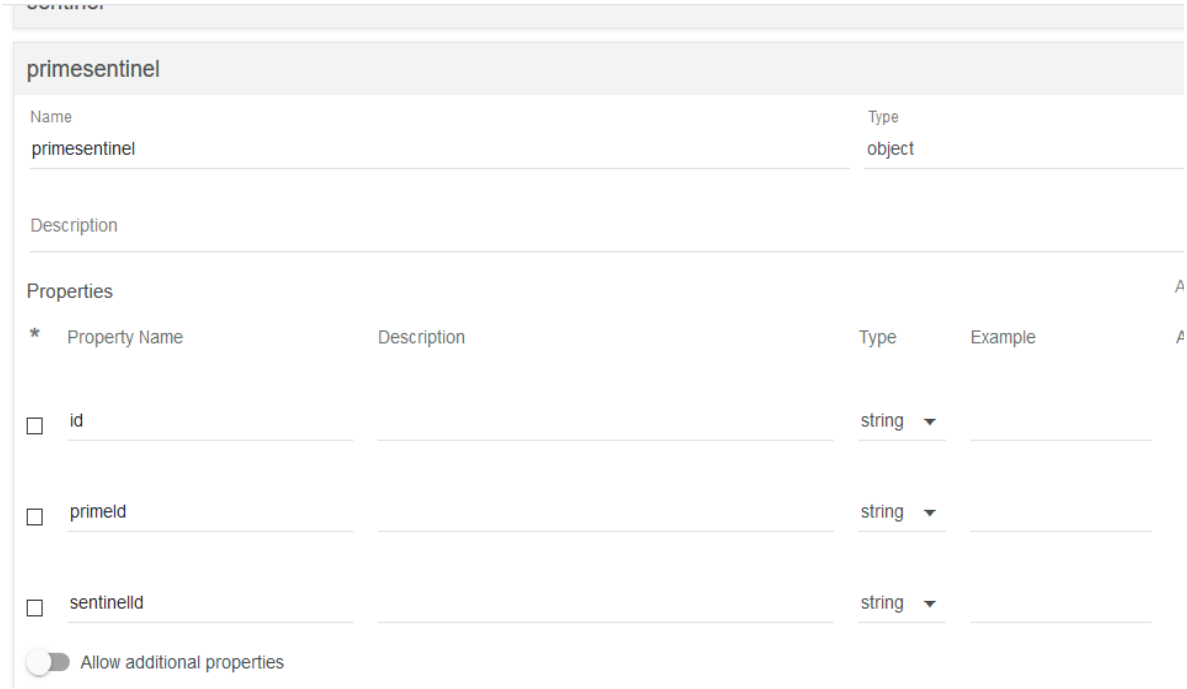


- Follow the next prompts and provide a property through which a relation is to configured.
- You can also provide a Foriegn Key .
- If you don't, the loopback will provide the Foriegn Key of it's own.
- The relation name appended with ‘Id’, for example, for relationname "customer" the default foreign key is "customerId".
- Now run the loopback application and Observe the Definitions.

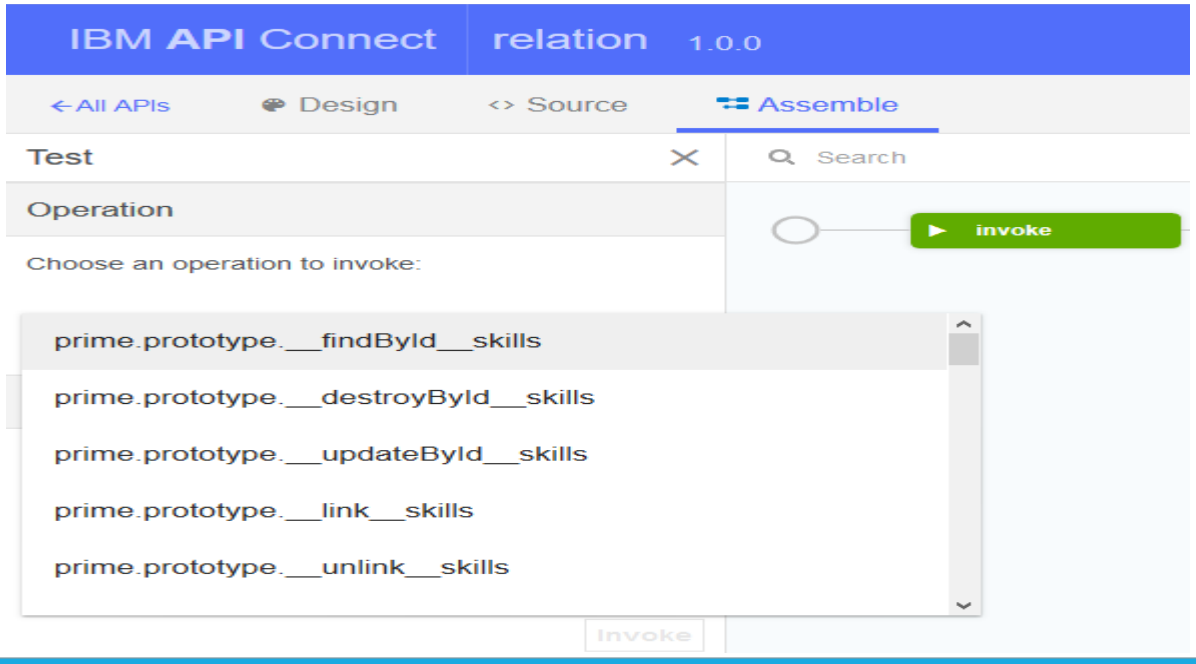


- You'll find a new definition which is the appended names of related models.
- You can also see this model is combination of their Model Id's.

- You must understand that id of a particular model acts as a primary key for that model.



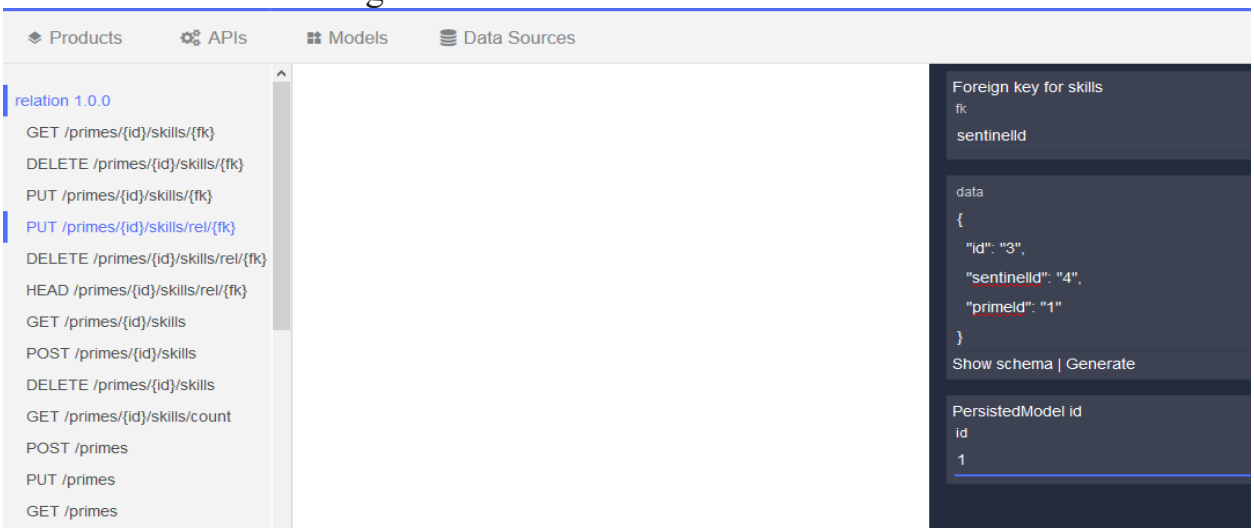
Also you'll find the change in Operation capabilities such as below:



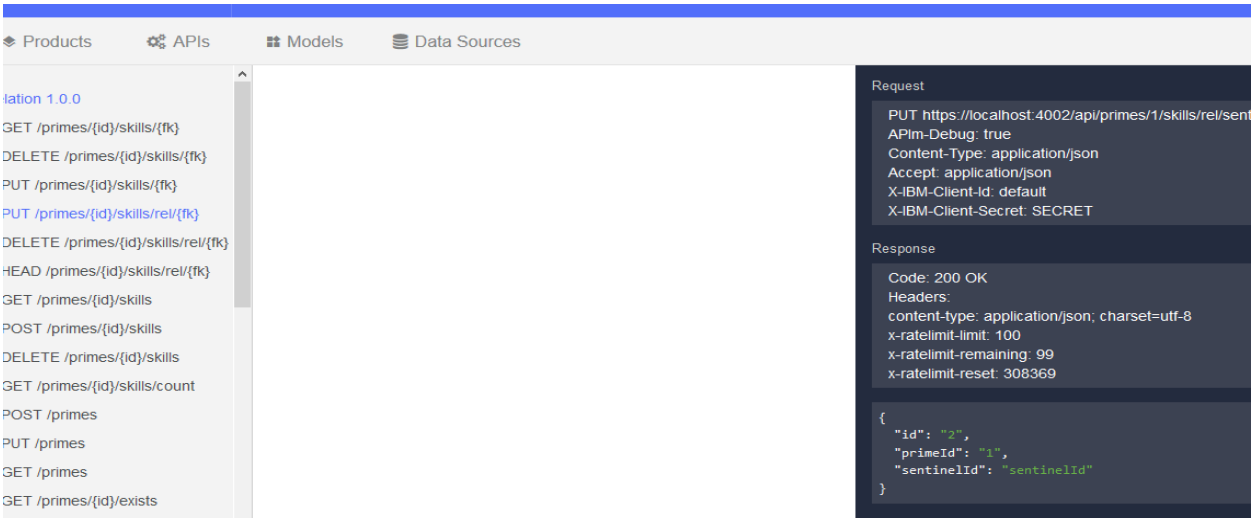
- These operations are those which relate both Models.
- You can perform an operation using data of **prime** model and it affects the model **sentinel.**

Working On Operations:

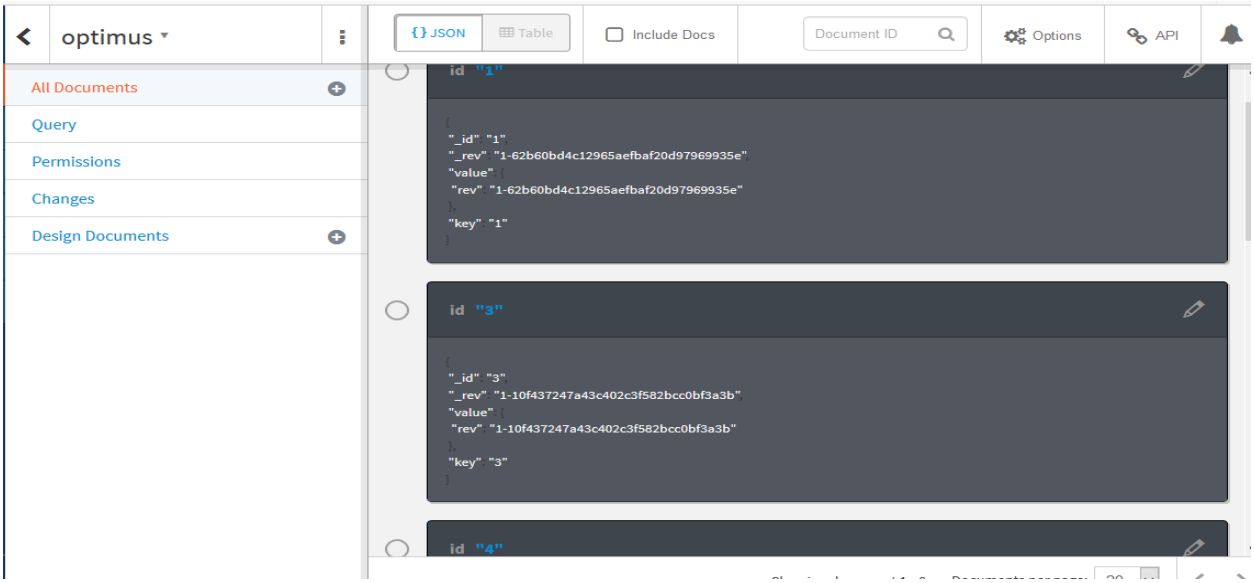
- First open the Database and see the available tables and data in it.
- For example my database Optimus has only 2 models(tables) with id's 1 & 4.
- Now take the operation **Link** and give the foriegnkey (here **sentinelId**), corresponding data and **ModelId(of Model1 i.e., prime)**
- Or follow the image below



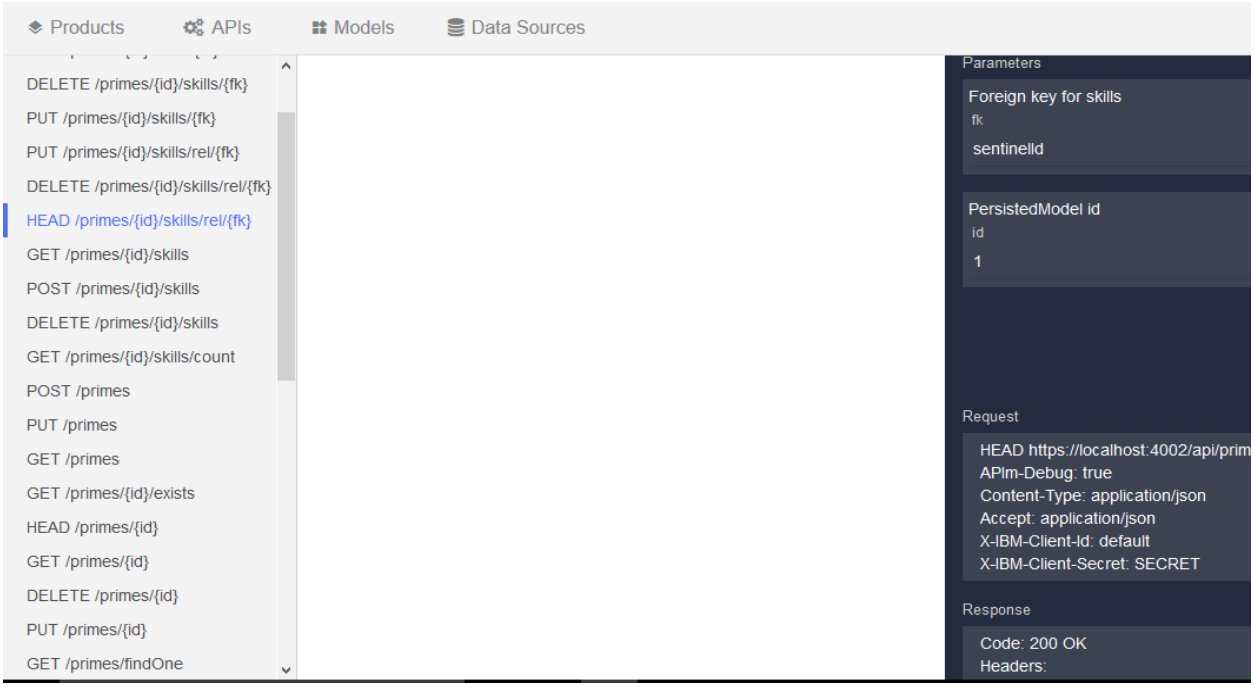
OUTPUT



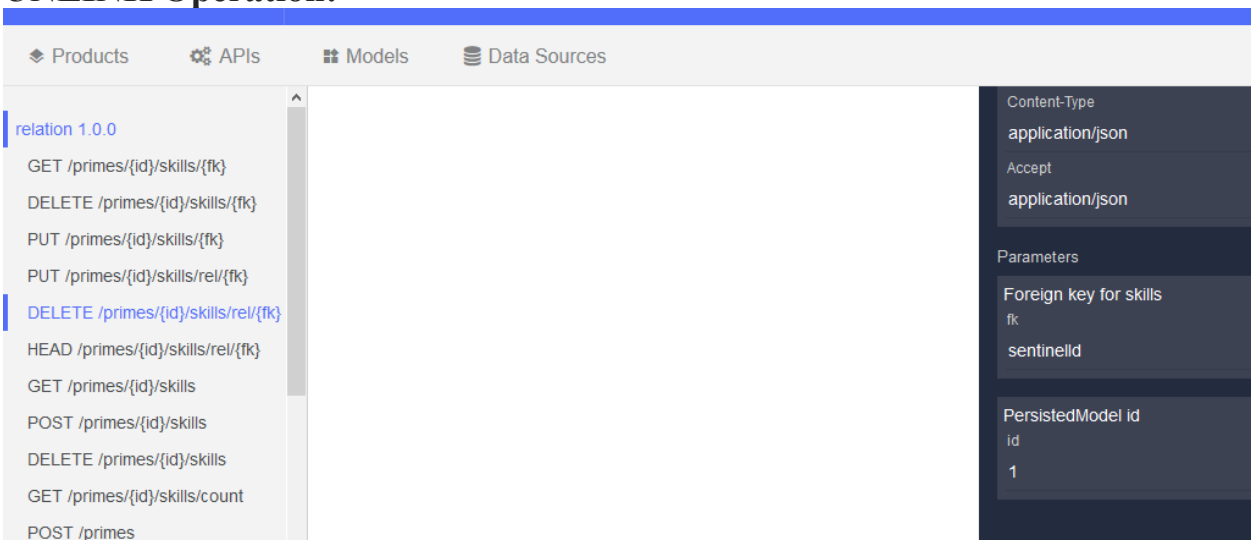
- To check if there exists a relation you can check your database for the additional table.



- Or you can use the Operation **EXISTS (or follow as in image below)**

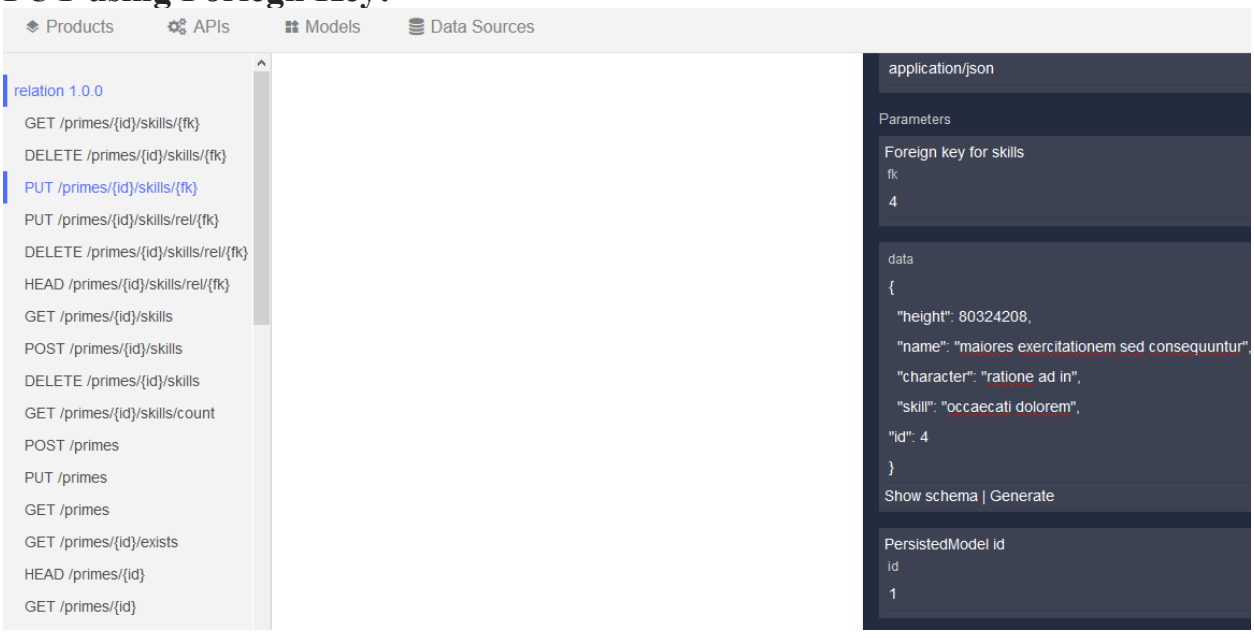


UNLINK Operation:



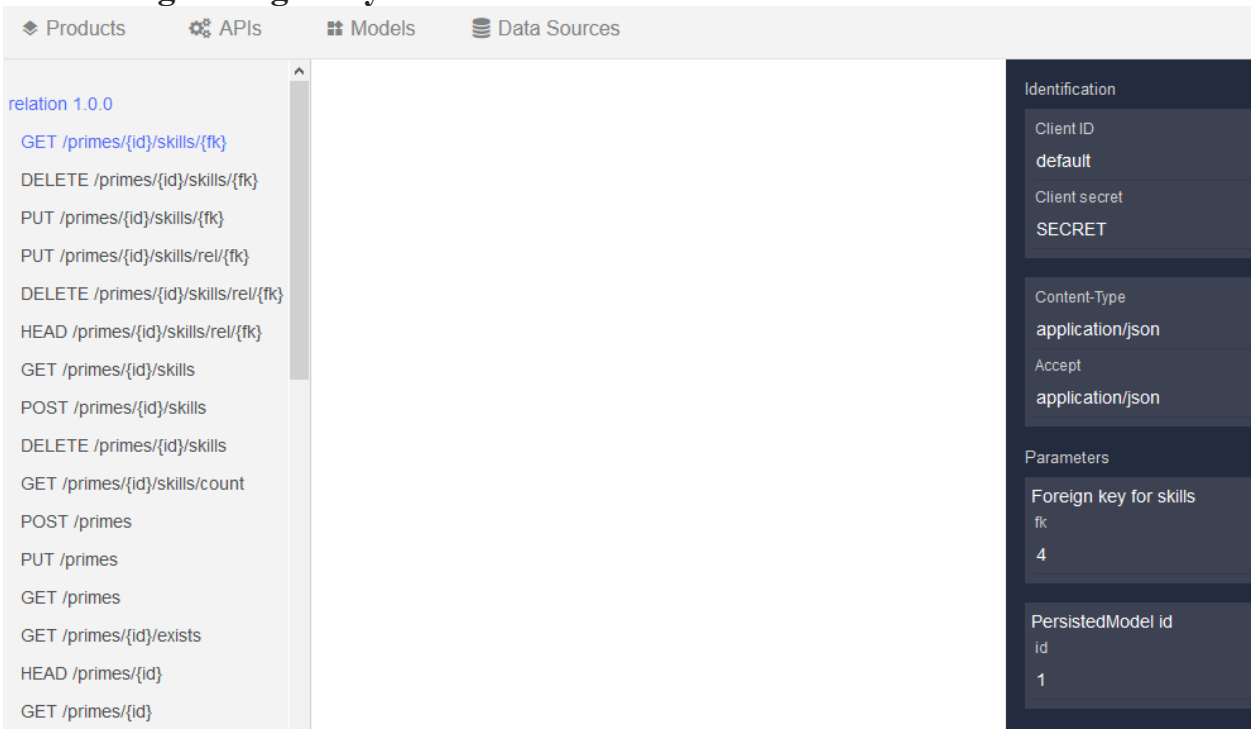
- If you don't want the relation to exist you can **DELETE** or **UNLINK.**
- You can check your Database and find the the related table with missing here **id:3.**
- **Now** you can understand that Link operation creates a new model(table) with properties of both Parent Models.
- It's like a joined table after Join a operation.
- Whereas UNLINK is Inverse of Link operation.
- Explore the other HTTP methods.

PUT using Foriegn Key:

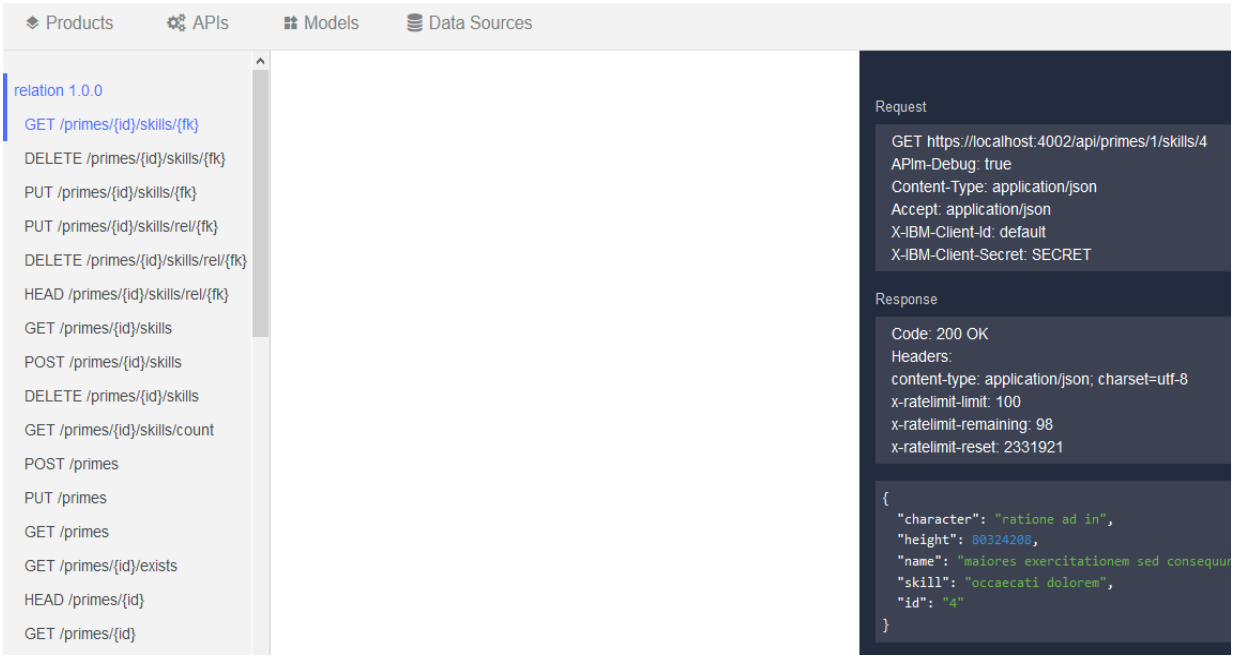


OUTPUT:

GET using Foriegn Key:



OUTPUT:



The Other GET operation: