# Relations entre entités Symfony

[sommaire]

1. ## Objectifs

- À la fin de ce cours, vous serez capables de modéliser efficacement les relations entre les entités dans une application Symfony en utilisant les annotations Doctrine, en comprenant et en appliquant les concepts de relations OneToOne, OneToMany, ManyToOne et ManyToMany, et en mettant en œuvre ces relations de manière correcte et efficace dans vos propres projets Symfony."

## Présentation

- Pour les bases de données qui fonctionnent sur la base de relations, vous devez définir la structure de votre base de données avant d'y insérer des données.
- MySQL le plus connu au monde est une base de données relationnelle. Les autres SGBDR célèbres sont MSSQL, Oracle, Microsoft Access et SQLite. Dans ces bases de données, une collection de données similaires est appelée une table.
- La clé primaire d'une table est la clé étrangère d'une autre table sont reliées pour faire la relation entre ces deux tanles.
- Il existe trois types de base de relations **un-à-un**, **un-à-plusieurs** et **plusieurs-à-plusieurs**.
- Dans PHP personnalisé ou dans des Frameworks qui fonctionnent sur la base du modèle Active Record, vous devez gérer les relations à l'aide de jointures tandis que dans Laravel, vous avez également cette option, mais vous pouvez gérer toutes ces relations de manière très simple en utilisant des modèles éloquents .

## Propriétaire et inverse

- Il existe également une notion très importante dans ces relations : **propriétaire** et **inverse**.
- Dans une relation entre deux entités, il y a toujours une entité dite **propriétaire**, et une dite **inverse**. L'entité propriétaire est celle qui contient la **référence** à l'autre **entité**.
- Prenons un exemple, supposons qu'il existe une relation entre deux entités : **review (avis)** et **product (produit)**. La table **review** peut comporter une colonne **product_id** qui fait référence à la table **product**.
- Ainsi, la table **review** est le propriétaire de cette relation, car c'est elle qui contient la colonne de liaison **product_id**, indiquant quelle critique est associée à quel produit.

apcpedagogie.com

- En d'autres termes, la table **review** détient la référence vers la table **product**, ce qui en fait l'entité propriétaire de la relation.
- Supposons qu'il existe une relation entre deux autres entités : **Student (étudiant)** et **Course (cours)**. La table **Enrollment (inscription)** peut comporter une colonne **student_id** qui fait référence à la table **Student**, indiquant à quel étudiant un cours est associé.
- De plus, la table **Enrollment** peut également contenir une colonne **course_id** qui fait référence à la table **Course**, indiquant à quel cours un étudiant est inscrit.
- Ainsi, la table Enrollment est le propriétaire de cette relation, car elle détient les références vers les entités Student et Course via les colonnes student_id et course_id, respectivement.
- En d'autres termes, la table **Enrollment** détient les références vers les tables **Student** et **Course**, ce qui en fait l'entité propriétaire de la relation.

## Générer la relation

- Tout d'abord, exécutez la commande suivante dans votre terminal pour démarrer le processus de création d'une nouvelle entité :`php bin/console make:entity`
- Lorsque vous y êtes invité, saisissez le nom de l'entité que vous souhaitez créer. Pour cet exemple, nous utiliserons le nom "**Enrollment**" pour représenter les inscriptions des étudiants à des cours.
- Ensuite, le générateur vous demande d'ajouter des champs à votre entité. À ce stade, nous n'avons pas encore défini la relation entre "**Student**" et "**Enrollment**", donc ajoutez les champs nécessaires à votre entité "**Enrollment**" selon vos besoins spécifiques.
- Après avoir ajouté les champs, le générateur vous demandera si vous souhaitez ajouter une relation à une autre entité. Répondez "yes" pour indiquer que vous souhaitez ajouter une relation.
- À la question suivante, vous devez spécifier le type de relation. Dans notre cas, puisque chaque inscription est liée à un étudiant, la relation à utiliser est "**ManyToOne**".
- Ensuite, le générateur vous demande avec quelle classe cette entité devrait être liée. Vous devez spécifier la classe "**Student**" pour établir la relation avec la table des étudiants.
- Après avoir spécifié la classe "**Student**", le générateur vous demandera le nom de la propriété de la relation dans votre entité "**Enrollment**". Vous pouvez choisir un nom significatif, par exemple "**student**".
- Enfin, le générateur vous demandera si la propriété "**student**" dans votre entité "**Enrollment**" est autorisée à être nulle. Puisque chaque inscription doit être liée à un étudiant, vous devez répondre "**no**".

---

apcpedagogie.com

**Cours Symfony**

Source : https://apcpedagogie.com/relations-entre-entites-symfony · téléchargé le 03/11/2026 · © APC Pédagogie · apcpedagogie.com

apcpedagogie.com