TP2 : BDD et l'ORM Doctrine

L'objectif principal de ce premier TP est de créer sa BDD et d'interagir avec en y insérant des données.

1. Création de la BDD et connexion

- Lancez votre terminal et saisissez :

« php bin/console doctrine:database:create ».

Qu'observez-vous et pourquoi ?

- Vous avez accès à de nombreuses fonctionnalités pour communiquer avec votre BDD à partir de cette commande. Quelle est celle pour updater votre BDD après mis à jour vos entités ?

- Pour corriger les erreurs rencontrées lors du lancement de la commande plus haut, il est nécessaire de configurer la connexion BDD - Symfony. Pour cela, ouvrez le fichier « .env » à la racine de votre application.

```
###> doctrine/doctrine-bundle ###
# Format described at https://www.doctrine-project.org/projects/doctrine-dbal/en/latest/reference/configuration.html#connecting-using-a-url
# For an SQLite database, use: "sqlite:///%kernel.project_dir%/var/data.db"
# For a PostgreSQL database, use: "postgresql://db_user:db_password@127.0.0.1:5432/db_name?serverVersion=11&charset=utf8"
# IMPORTANT: You MUST configure your server version, either here or in config/packages/doctrine.yaml
DATABASE_URL=mysql://db_user:db_password@127.0.0.1:3306/db_name?serverVersion=5.7
###< doctrine/doctrine-bundle ###
```

La ligne à modifier est la 32 :

DATABASE_URL=mysql://db_user:db_password@127.0.0.1:3306/db_name?serverVersion=5.7

- Nous allons mentionner le nom d'utilisateur de notre BDD (db_user), son mot de passe (db_password), et le nom de la base de données (db_name) dans le fichier.

Par exemple, pour mon environnement local, j'aurais :

DATABASE_URL=mysql://root:root@127.0.0.1:3307/concert?serverVersion-5.7

(Attention, avoir « root » en user et « root » en mdp n'est clairement pas sécurisé…
+ Attention mon port local d'exposition de la BDD est 3307… Le vôtre peut être différent).

- Vérifiez que le fichier .env est bien dans votre .gitignore !
- Relancer la première commande de création de votre BDD.

=> Si tout s'est bien passé, vous devriez avoir cela dans votre terminal :

```
aka@Air-de-Aka concertProject % php bin/console doctrine:database:create
Created database `concert` for connection named default
```

2. Création de votre première entité

Pour cela, nous allons utiliser le MakerBundle, qui permet de générer du code automatiquement. Nous allons créer la première entité « band » telle que nous l'avons défini précédemment via cette commande:

php bin/console make:entity [EntityName]

- Je vous laisse découvrir et répondre à l'ensemble des questions qui vous sont posées, en accord avec l'architecture que nous avons déterminée précédemment.
- Rendez vous dans votre IDE, et observez les changements.
  - Quels sont les fichiers crées ?
  - Que contiennent-ils ?

- Rendez vous ensuite dans PhpMyAdmin : qu'observez-vous ? Pourquoi ? Que faire pour y remédier ?

- Fin du suspense, il faut mettre à jour votre BDD ! Dans votre console, saisissez :

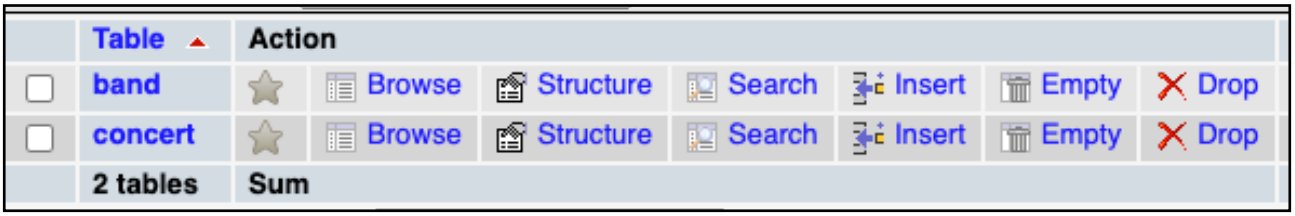
php bin/console doctrine:schema:update --dump-sql

Que s'affiche-t-il dans la console ?

Si vous êtes en accord avec ça, jouez :

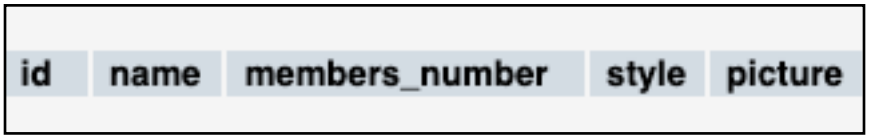
php bin/console doctrine:schema:update --force

- Retournez maintenant dans votre PhpMyAdmin….



Et la magie a opéré. Observez également les noms des champs : ils ont été automatiquement mis en forme selon la norme SQL.

- Parce que je suis plutôt pointilleuse sur la nomenclature, je veux que tous les champs soient en anglais. Veuillez modifier le champ « image » en « picture », de même pour l'attribut ; effectuez les autres corrections si nécessaires.



3. Création d'autres entités : les relations

- Créer les autres entités selon l'architecture définie ensemble.
- Attention, ces autres entités ont des relations entre elles :

https://www.doctrine-project.org/projects/doctrine-orm/en/latest/reference/association-mapping.html#association-mapping

Je vous laisse consulter la documentation, elle vaut mieux qu'un long discours.
Quel est le cas qui nous intéresse dans le cadre de notre application ?

- OrphanRemoval = Suppression des entités qui n'ont pas de relations, si celles-ci sont obligatoires. ex : on supprime un groupe de la BDD. Est-ce que les musiciens liés sont supprimés de la BDD également ou non ?
Si oui : il faut activer l'option « orphanRemoval », qui signifie littéralement « suppression des entités orphelines ». C'est très important pour garder une BDD propre, mais attention à l'utilisation, on peut se retrouver à supprimer beaucoup de données en cascade en un seul clic…

Pour vous aider:

```
Add another property? Enter the property name (or press <return> to stop adding fields):
> members

Field type (enter ? to see all types) [string]:
> relation

What class should this entity be related to?:
> member

What type of relationship is this?
------------ ---------------------------------------------------------------
 Type         Description
------------ ---------------------------------------------------------------
 ManyToOne    Each Band relates to (has) one member.
              Each member can relate to (can have) many Band objects

 OneToMany    Each Band can relate to (can have) many member objects.
              Each member relates to (has) one Band

 ManyToMany   Each Band can relate to (can have) many member objects.
              Each member can also relate to (can also have) many Band objects

 OneToOne     Each Band relates to (has) exactly one member.
              Each member also relates to (has) exactly one Band.
------------ ---------------------------------------------------------------

Relation type? [ManyToOne, OneToMany, ManyToMany, OneToOne]:
> OneToMany

A new property will also be added to the member class so that you can access and set the related Band object from it.

New field name inside member [band]:
> band

Is the member.band property allowed to be null (nullable)? (yes/no) [yes]:
> no

Do you want to activate orphanRemoval on your relationship?
A member is "orphaned" when it is removed from its related Band.
e.g. $band->removeMember($member)

NOTE: If a member may *change* from one Band to another, answer "no".

Do you want to automatically delete orphaned App\Entity\member objects (orphanRemoval)? (yes/no) [no]:
>
```

Je vous laisse réfléchir aux autres relations !

Attention : on crée d'abord les entités sans relations, où les unités plus petites, avant de les lier aux unités plus importantes (celles qui portent l'information).

Conseil : crée vos entités une par une, et mettez à jour après chaque création votre BDD. Ainsi s'il y a une erreur, vous saurez plus facilement quelle est l'entité concernée.

- Une fois tout crée, ouvrez les fichiers d'entités et observez les nouvelles méthodes. A quoi sert la méthode « remove » ? Dans quel cas est-elle créée ?

- Allez dans PhpMyAdmin, regardez les tables : qu'est-ce que la table « show_band »?

4. Pour les plus motivés….

- Créer une table Picture(name, alternativeName, url) qui est liée à chaque table qui porte un champ « picture » et faites les modifications conséquentes.