

Workshop: Jointure

Objectifs

- Le but de ce workshop est de créer une jointure entre deux tables de la base de données.

Exemple Jointure



Partie 1 : Configuration PhpMyAdmin

1. Ajouter la table « Genre » à la base de données « atelierPHP » comme montré dans la figure ci- dessous :

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	idGenre	int(11)			No	None		AUTO_INCREMENT	Change Drop More
<input type="checkbox"/> 2	nom	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More

2. Ajouter la table « Album » à la base de données « atelierPHP » sachant que :

- Les attributs de cette table sont les propriétés de la classe Album.
- Ajouter un attribut « idAlbum » qui sera la clé primaire de la table.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	idAlbum	int(11)			No	None		AUTO_INCREMENT	Change Drop More
<input type="checkbox"/> 2	titre	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/> 3	prix	float			No	None			Change Drop More
<input type="checkbox"/> 4	image	varchar(150)	utf8mb4_general_ci		No	None			Change Drop More
<input type="checkbox"/> 5	genre	int(11)			No	None			Change Drop More

3. Au niveau de l'onglet « Structure », ouvrez la vue relationnelle :

4. Au niveau de ce formulaire (figure ci-dessous), on précise la liaison de jointure entre les deux tables.

5. L'attribut « genre » de la table « album » va référencer l'attribut « idGenre » de la table « genre ».

6. On clique sur « Save » et la nouvelle contrainte est ajoutée.

Indexes									
Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
	PRIMARY	BTREE	Yes	No	idAlbum	3	A	No	
	fk_album_genre	BTREE	No	No	genre	3	A	No	

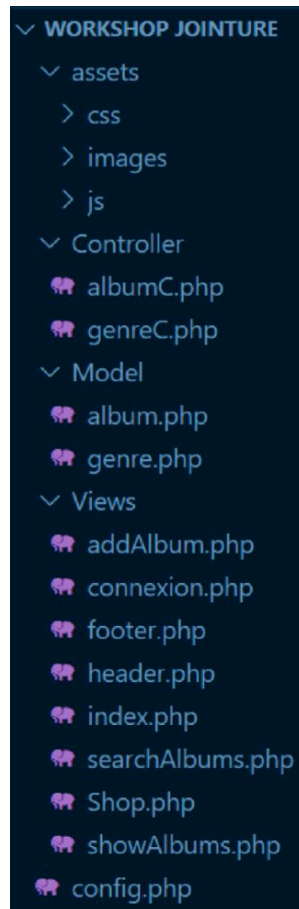
Create an index on columns

7. On peut vérifier la création au niveau de l'onglet « Structure »

Browse Structure SQL Search Insert Export Import Privileges Operations Triggers									
Table structure Relation view									
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	idAlbum	int(11)			No	None		AUTO_INCREMENT	
<input type="checkbox"/> 2	titre	varchar(50)	utf8mb4_general_ci		No	None			
<input type="checkbox"/> 3	prix	float			No	None			
<input type="checkbox"/> 4	image	varchar(150)	utf8mb4_general_ci		No	None			
<input type="checkbox"/> 5	genre	int(11)			No	None			

8. Insérer des lignes au niveau des tables « genre » et « album ».

Partie 2 : Arborescence de fichiers



```
WORKSHOP JOINTURE
├── assets
│   ├── css
│   ├── images
│   └── js
├── Controller
│   ├── albumC.php
│   └── genreC.php
├── Model
│   ├── album.php
│   └── genre.php
├── Views
│   ├── addAlbum.php
│   ├── connexion.php
│   ├── footer.php
│   ├── header.php
│   ├── index.php
│   ├── searchAlbums.php
│   ├── Shop.php
│   └── showAlbums.php
└── config.php
```

The image shows a file explorer interface with a dark background. The root directory is 'WORKSHOP JOINTURE'. It contains several subdirectories and files: 'assets' (with subdirectories 'css', 'images', and 'js'), 'Controller' (with 'albumC.php' and 'genreC.php'), 'Model' (with 'album.php' and 'genre.php'), 'Views' (with 'addAlbum.php', 'connexion.php', 'footer.php', 'header.php', 'index.php', 'searchAlbums.php', 'Shop.php', and 'showAlbums.php'), and a 'config.php' file at the root level. Each file is preceded by a small icon of a pink elephant.

Partie 3 : Affichage liste des albums par genre

9. Créer la méthode « `afficheAlbums($idGenre)` » dans le fichier « `genreC.php` » permettant d'afficher la liste des albums de la base de données.

```
class GenreC {
    public function afficheAlbums($idGenre) {
        try {
            $pdo = config::getConnexion();
            $query = $pdo->prepare("SELECT * FROM album WHERE genre = :id");
            $query->execute(['id' => $idGenre]);
            return $query->fetchAll();
        } catch (PDOException $e) {
            echo $e->getMessage();
        }
    }

    public function afficheGenres() {
        try {
            $pdo = config::getConnexion();
            $query = $pdo->prepare("SELECT * FROM genre");
            $query->execute();
            return $query->fetchAll();
        } catch (PDOException $e) {
            echo $e->getMessage();
        }
    }
}
```

10. Dans le fichier « `searchAlbums.php` », on va créer un formulaire qui va lister tous les genres comme le montre la figure suivante :

Recherche d'albums par genre

Sélectionnez un genre :

Albums correspondants au genre sélectionné :

a. Faire appel au « Controller » :

```
<?php
require_once "../controller/genreC.php";

$genreC = new GenreC();

// Traitement du formulaire
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    if (isset($_POST['genre']) && isset($_POST['search'])) {
        $idGenre = $_POST['genre'];
        $list = $genreC->afficheAlbums($idGenre);
    }
}
```

b. Formulaire pour sélectionner un genre

```
$genres = $genreC->afficheGenres();
?>
<!DOCTYPE html>
<head>
|   <title>Recherche d'albums</title>
</head>
<body>
|   <h1>Recherche d'albums par genre</h1>
|   <form action="" method="POST">
|       <label for="genre">Sélectionnez un genre :</label>
|       <select name="genre" id="genre">
|           <?php
|               foreach ($genres as $genre) {
|                   echo '<option value="' . $genre['idGenre'] . '">' . $genre['nom'] .
|               }
|           ?>
|       </select>
|       <input type="submit" value="Rechercher" name="search">
|   </form>
```

b. Afficher les albums correspondants au genre sélectionné :

```
<?php if (isset($list)) { ?>
|   <br>
|   <h2>Albums correspondants au genre sélectionné :</h2>
|   <ul>
|       <?php foreach ($list as $album) { ?>
|           <li><?= $album['titre'] ?> - <?= $album['prix'] ?> dt</li>
|       <?php } ?>
|   </ul>
|   <?php } ?>
</body>

</html>
```

c. Résultat :

Recherche d'albums par genre

Sélectionnez un genre :

Albums correspondants au genre sélectionné :

- album2 - 300 dt