

# PDO 2/3

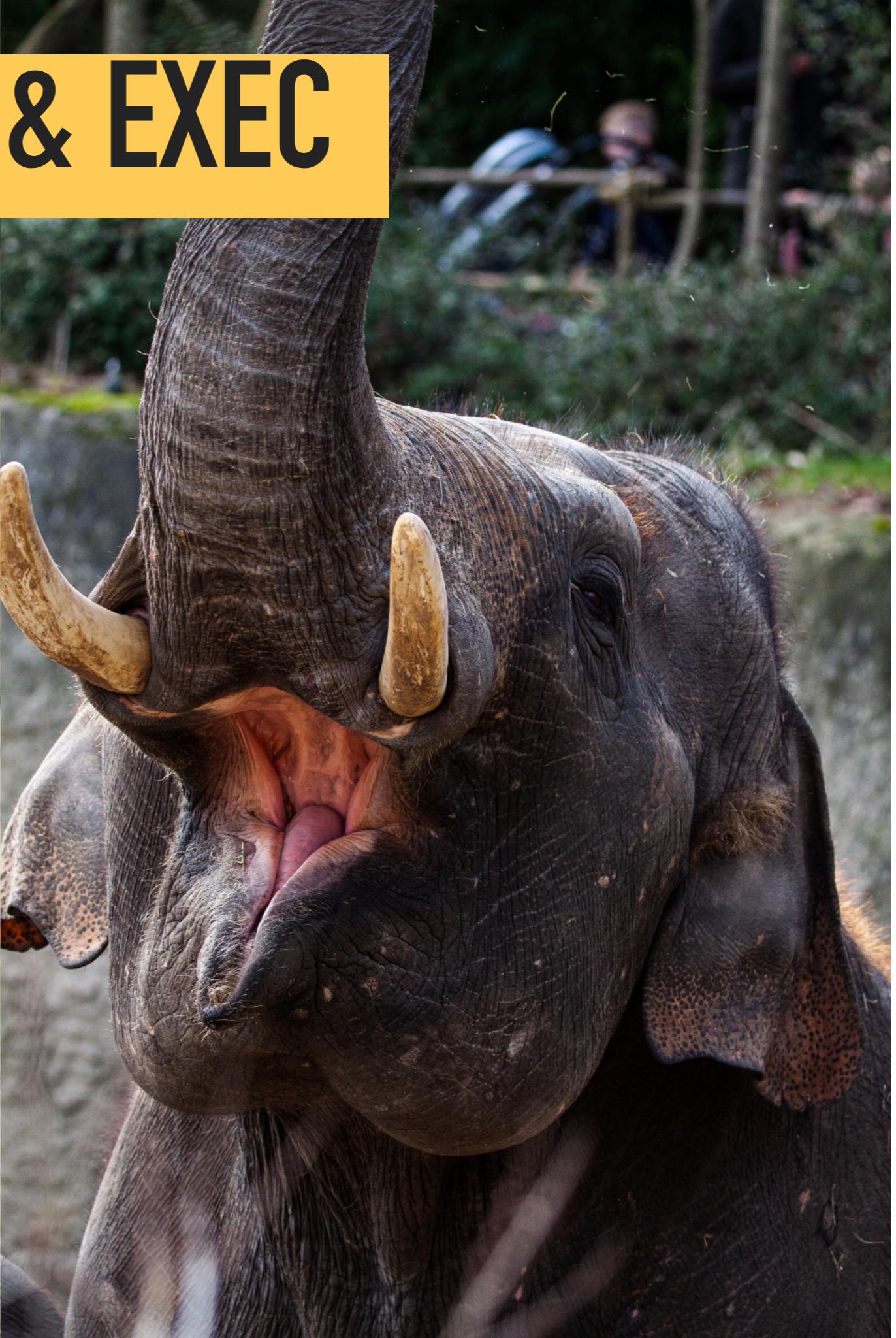
Auteur : Sébastien Inion

studi

Photo de Kritsada Seekham: <https://www.pexels.com/fr-fr/photo/nature-eau-bleu-animal-7836299/>

[https://github.com/SebInfo/PDO\\_Studi\\_Part2](https://github.com/SebInfo/PDO_Studi_Part2)

# QUERY & EXEC



# ETUDIONS LA MÉTHODE QUERY()

Change language: French

[Submit a Pull Request](#) [Report a Bug](#)

## PDO::query

(PHP 5 >= 5.1.0, PHP 7, PHP 8, PECL pdo >= 0.2.0)

PDO::query — Prépare et Exécute une requête SQL sans marque substitutive

### Description

```
public PDO::query(string $query, ?int $fetchMode = null): PDOStatement|false
```

```
public PDO::query(string $query, ?int $fetchMode = PDO::FETCH_COLUMN, int $colno): PDOStatement|false
```

```
public PDO::query(
    string $query,
    ?int $fetchMode = PDO::FETCH_CLASS,
    string $classname,
    array $constructorArgs
): PDOStatement|false
```

```
public PDO::query(string $query, ?int $fetchMode = PDO::FETCH_INTO, object $object): PDOStatement|false
```

PDO::query() prépare et exécute une requête SQL en un seul appel de fonction, retournant la requête en tant qu'objet [PDOStatement](#).

Pour une requête que vous devez exécuter plusieurs fois, vous réaliserez de meilleures performances si vous préparez l'objet [PDOStatement](#) en utilisant la fonction [PDO::prepare\(\)](#) et exécutez la requête via plusieurs appels à la fonction [PDOStatement::execute\(\)](#).

Si vous ne récupérez pas toutes les données du jeu de résultats avant d'exécuter le prochain appel à PDO::query(), votre appel peut échouer. Appeler [PDOStatement::closeCursor\(\)](#) pour libérer les ressources de la base de données associées à l'objet [PDOStatement](#) avant d'exécuter votre prochain appel à la fonction PDO::query().

#### Note:

Si **query** contient des marqueurs de substitution, la requête doit être préparé et exécuté séparément en utilisant les méthodes [PDO::prepare\(\)](#) et [PDOStatement::execute\(\)](#).

# ETUDIONS LA MÉTHODE QUERY()

---

## Liste de paramètres

---

### query

La requête SQL à préparer et à exécuter.

Si le SQL contient des marqueurs de substitution, [PDO::prepare\(\)](#) et [PDOStatement::execute\(\)](#) doivent être utilisé à la place. Alternativement, le SQL peut être préparé manuellement avant d'appeler [PDO::query\(\)](#), avec les données proprement formaté utilisant [PDO::quote\(\)](#) si le pilote le supporte.

### fetchMode

Le mode de récupération par défaut pour le [PDOStatement](#) retourné. Ceci doit être une des constantes [PDO::FETCH\\_\\*](#).

Si cet argument est passé à la fonction, le reste des arguments seront traité comme si [PDOStatement::setFetchMode\(\)](#) a été appelé sur l'objet de la requête résultante. Les arguments suivant dépendent en fonction du mode de récupération sélectionné.

## Valeurs de retour

---

Retourne un object [PDOStatement](#) ou **false** si une erreur survient.

# ETUDIONS LA MÉTHODE `FETCH()`

## PDOStatement::fetch

(PHP 5 >= 5.1.0, PHP 7, PHP 8, PECL pdo >= 0.1.0)

PDOStatement::fetch — Récupère la ligne suivante d'un jeu de résultats PDO

### Description

```
public PDOStatement::fetch(int $mode = PDO::FETCH_DEFAULT, int $cursorOrientation = PDO::FETCH_ORI_NEXT, int  
$cursorOffset = 0): mixed
```

Récupère une ligne depuis un jeu de résultats associé à l'objet PDOStatement. Le paramètre **mode** détermine la façon dont PDO retourne la ligne.

### Liste de paramètres

#### mode

Contrôle comment la prochaine ligne sera retournée à l'appelant. Cette valeur doit être une des constantes PDO::FETCH\_\*, et par défaut, vaut la valeur de PDO::ATTR\_DEFAULT\_FETCH\_MODE (qui vaut par défaut la valeur de la constante PDO::FETCH\_BOTH).

- PDO::FETCH\_ASSOC: retourne un tableau indexé par le nom de la colonne comme retourné dans le jeu de résultats
- PDO::FETCH\_BOTH (défaut): retourne un tableau indexé par les noms de colonnes et aussi par les numéros de colonnes, commençant à l'index 0, comme retournés dans le jeu de résultats
- PDO::FETCH\_BOUND: retourne **true** et assigne les valeurs des colonnes de votre jeu de résultats dans les variables PHP à laquelle elles sont liées avec la méthode [PDOStatement::bindColumn\(\)](#)
- PDO::FETCH\_CLASS: retourne une nouvelle instance de la classe demandée, liant les colonnes du jeu de résultats aux noms des propriétés de la classe et en appelant le constructeur par la suite, sauf si PDO::FETCH\_PROPS\_LATE est également donné. Si **fetch\_style** inclut PDO::FETCH\_CLASS (c'est-à-dire PDO::FETCH\_CLASS | PDO::FETCH\_CLASSTYPE), alors le nom de la classe est déterminé à partir d'une valeur de la première colonne.
- PDO::FETCH\_INT: met à jour une instance existante de la classe demandée, liant les colonnes du jeu de résultats aux noms des propriétés de la classe

## PDOStatement

```
+bindColumn()  
+bindParam()  
+bindValue()  
+errorCode()  
+errorInfo()  
+execute()  
+fetch()  
+fetchAll()  
+fetchColumn()  
+getAttribute()  
+rowCount()  
+setAttribute()  
+setFetchMode()
```

# PDO::FETCH\_BOTH

```
$req="SELECT nom, prenom FROM acteur";
$result = $bdd->query($req);

// On affiche les résultats si pas d'erreur
if ($result === false) die("erreur requête !");

while($unActeur = $result->fetch(PDO::FETCH_BOTH)) {
    echo "<pre>";
    print_r($unActeur);
    echo "</pre>";
}
```

```
Array
(
    [nom] => Johnny
    [0] => Johnny
    [prenom] => Deep
    [1] => Deep
)

Array
(
    [nom] => Al
    [0] => Al
    [prenom] => Pacino
    [1] => Pacino
)

Array
(
    [nom] => Suraj
    [0] => Suraj
    [prenom] => Sharma
    [1] => Sharma
)

Array
(
    [nom] => Brad
    [0] => Brad
    [prenom] => Pitt
    [1] => Pitt
)

Array
(
    [nom] => Edward
    [0] => Edward
    [prenom] => Norton
    [1] => Norton
)

Array
(
    [nom] => D'artagnan
    [0] => D'artagnan
    [prenom] => Edouard
    [1] => Edouard
)
```

# PDO::FETCH\_NUM

```
$req="SELECT nom, prenom FROM acteur";
$result = $bdd->query($req);

// On affiche les résultats si pas d'erreur
if ($result === false) die("erreur requête !");

while($unActeur = $result->fetch(PDO::FETCH_NUM)) {
    echo "<pre>";
    print_r($unActeur);
    echo "</pre>";
}
```

## Lister les acteurs

```
Array
(
    [0] => Johnny
    [1] => Deep
)
```

```
Array
(
    [0] => Al
    [1] => Pacino
)
```

```
Array
(
    [0] => Suraj
    [1] => Sharma
)
```

```
Array
(
    [0] => Brad
    [1] => Pitt
)
```

```
Array
(
    [0] => Edward
    [1] => Norton
)
```

```
Array
(
    [0] => D'artagnan
    [1] => Edouard
)
```

# PDO::FETCH\_ASSOC

```
$req="SELECT nom, prenom FROM acteur";
$result = $bdd->query($req);

// On affiche les résultats si pas d'erreur
if ($result === false) die("erreur requête !");

while($unActeur = $result->fetch(PDO::FETCH_ASSOC)) {
    echo "<pre>";
    print_r($unActeur);
    echo "</pre>";
}
```

```
Array
(
    [nom] => Johnny
    [prenom] => Deep
)

Array
(
    [nom] => Al
    [prenom] => Pacino
)

Array
(
    [nom] => Suraj
    [prenom] => Sharma
)

Array
(
    [nom] => Brad
    [prenom] => Pitt
)

Array
(
    [nom] => Edward
    [prenom] => Norton
)

Array
(
    [nom] => D'artagnan
    [prenom] => Edouard
)
```

# PDO::FETCH\_OBJ

```
$req="SELECT nom, prenom FROM acteur";
$result = $bdd->query($req);

// On affiche les résultats si pas d'erreur
if ($result === false) die("erreur requête !");

while($unActeur = $result->fetch(PDO::FETCH_OBJ)) {
    echo "<pre>";
    print_r($unActeur);
    echo "</pre>";
    echo $unActeur->nom;
}
```

```
stdClass Object
(
    [nom] => Johnny
    [prenom] => Deep
)

stdClass Object
(
    [nom] => Al
    [prenom] => Pacino
)

stdClass Object
(
    [nom] => Suraj
    [prenom] => Sharma
)

stdClass Object
(
    [nom] => Brad
    [prenom] => Pitt
)

stdClass Object
(
    [nom] => Edward
    [prenom] => Norton
)

stdClass Object
(
    [nom] => D'artagnan
    [prenom] => Edouard
)
```

# ETUDIONS LA MÉTHODE `FETCHALL()`

## `PDOStatement::fetchAll`

(PHP 5 >= 5.1.0, PHP 7, PHP 8, PECL pdo >= 0.1.0)

`PDOStatement::fetchAll` — Retourne un tableau contenant toutes les lignes du jeu d'enregistrements

### Description

```
public PDOStatement::fetchAll(int $mode = PDO::FETCH_DEFAULT): array
```

```
public PDOStatement::fetchAll(int $mode = PDO::FETCH_COLUMN, int $column): array
```

```
public PDOStatement::fetchAll(int $mode = PDO::FETCH_CLASS, string $class, ?array $constructorArgs): array
```

```
public PDOStatement::fetchAll(int $mode = PDO::FETCH_FUNC, callable $callback): array
```

### Liste de paramètres

#### `mode`

Contrôle le contenu du tableau retourné comme documenté dans la fonction [PDOStatement::fetch\(\)](#). Valeur par défaut : `PDO::ATTR_DEFAULT_FETCH_MODE` (qui prend sa valeur par défaut de `PDO::FETCH_BOTH`).

Pour retourner un tableau contenant toutes les valeurs d'une seule colonne depuis le jeu de résultats, spécifiez `PDO::FETCH_COLUMN`. Vous pouvez spécifier quelle colonne vous voulez avec le paramètre `column`.

Pour récupérer uniquement les valeurs uniques d'une seule colonne depuis le jeu de résultats, utilisez `PDO::FETCH_COLUMN` avec `PDO::FETCH_UNIQUE`.

Pour retourner un tableau associatif groupé par les valeurs d'une colonne spécifique, utilisez `PDO::FETCH_COLUMN` avec `PDO::FETCH_GROUP`.

# ETUDIONS LA MÉTHODE FETCHALL()

```
$req="SELECT nom, prenom FROM acteur";
$result = $bdd->query($req);

// On affiche les résultats si pas d'erreur
if ($result === false) die("erreur requête !");
$acteurs = $result->fetchAll();
echo "<pre>";
print_r($acteurs);
echo "</pre>";

foreach($acteurs as $unActeur)
{
    echo $unActeur['nom']." ";
    echo $unActeur['prenom'];
    echo "<br />";
}
```

```
Array
(
    [0] => Array
        (
            [nom] => Johnny
            [0] => Johnny
            [prenom] => Deep
            [1] => Deep
        )

    [1] => Array
        (
            [nom] => Al
            [0] => Al
            [prenom] => Pacino
            [1] => Pacino
        )

    [2] => Array
        (
            [nom] => Suraj
            [0] => Suraj
            [prenom] => Sharma
            [1] => Sharma
        )

    [3] => Array
        (
            [nom] => Brad
            [0] => Brad
            [prenom] => Pitt
            [1] => Pitt
        )

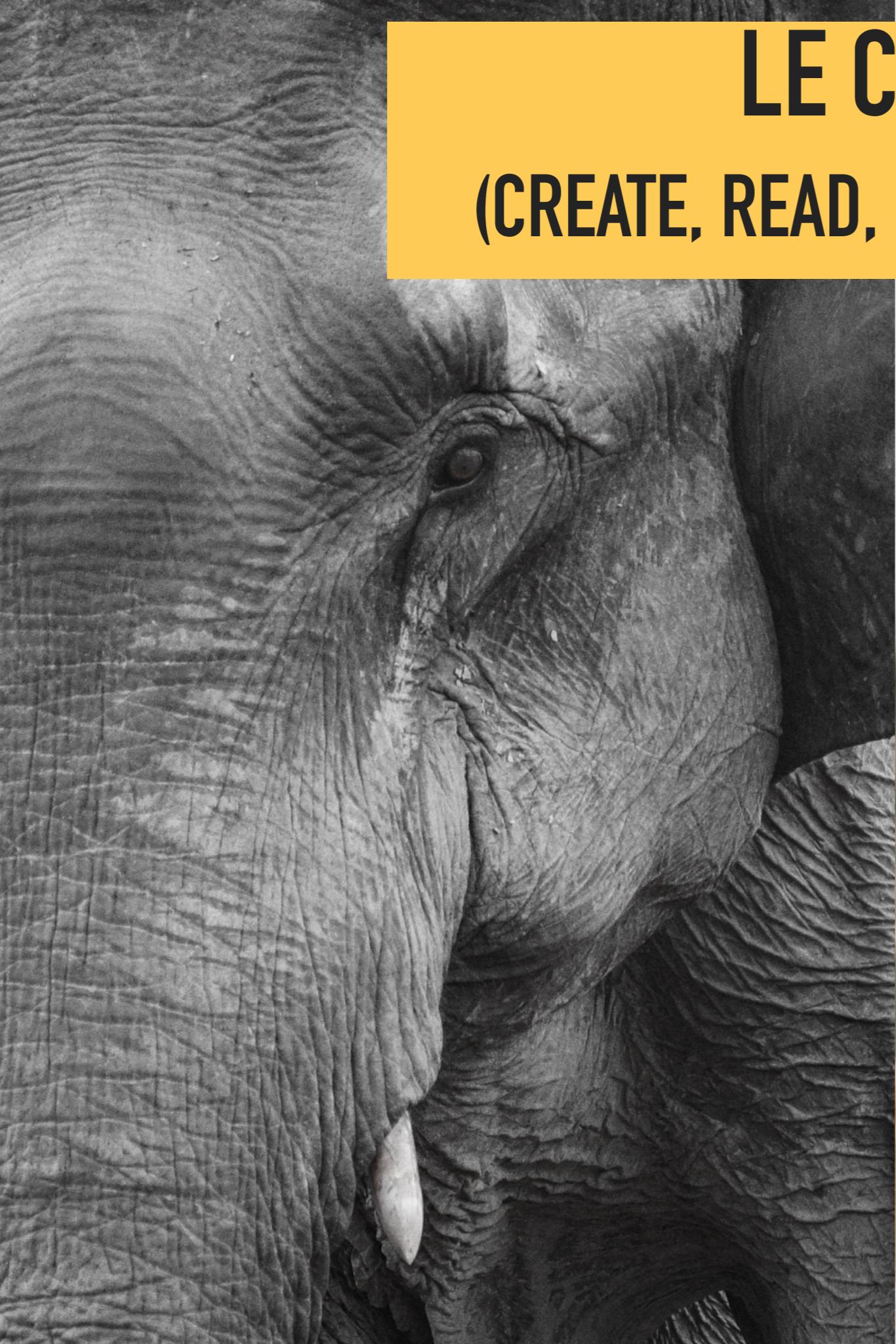
    [4] => Array
        (
            [nom] => Edward
            [0] => Edward
            [prenom] => Norton
            [1] => Norton
        )

    [5] => Array
        (
            [nom] => D'artagnan
            [0] => D'artagnan
            [prenom] => Edouard
            [1] => Edouard
        )
)
```

# REQUETE DE SELECTION

---

- ▶ Après l'execution des requêtes les données ne sont pas affichées.
- ▶ Elles sont en mémoires
- ▶ Soit sous forme d'un tableau associatif
- ▶ Soit un tableau numériquement indexé
- ▶ Soit un objet
- ▶ fetchAll() retourne toutes les données et libère le SGBD
- ▶ fetch() lit séquentiellement le résultat



# **LE CRUD**

## **(CREATE, READ, UPDATE, DELETE)**



# EXEC() ET QUERY()

---

- ▶ EXEC() est utilisé pour des requêtes ne retournant pas de résultat.
- ▶ QUERY() retourne directement un résultat et peut être utilisé pour des requêtes sans paramètre. Query prépare et exécute.

Requête SQL	Méthode PDO à utiliser
INSERT	exec()
UPDATE	exec()
DELETE	exec()
SELECT	query()
EXPLAIN	query()
SHOW	query()

# CREATION

---

- ▶ PDO::exec() retourne le nombre de lignes qui ont été modifiées ou effacées par la requête SQL exécutée. Si aucune ligne n'est affectée, la fonction PDO::exec() retournera 0 Cela ne fonctionne que pour les ajouts, modifications et suppressions.

```
<?php
require_once 'login.php';
$req="INSERT INTO acteur (idActeur,nom,prenom) VALUES ('7','Delon','Alain')";
try {
    $count = $bdd->exec($req);
}
catch(PDOException $e)
{
    echo $e->getMessage();
}
var_dump ($count);
?>
```

# LECTURE

---

- ▶ Déjà fait dans la première partie de ce cours et là on doit utiliser query() ou on verra plus tard plutôt prépare pour des raisons de sécurité.

```
$req="SELECT nom, prenom FROM acteur";
$result = $bdd->query($req);

// On affiche les résultats si pas d'erreur
if ($result === false) die("erreur requête !");

while($unActeur = $result->fetch(PDO::FETCH_OBJ)) {
    echo "<pre>";
    print_r($unActeur);
    echo "</pre>";
    echo $unActeur->nom;
}
```

# MODIFICATION

---

```
<h1>Modification d'un acteur</h1>
<?php
require_once 'login.php';
$req="UPDATE acteur set nom = UPPER(nom) WHERE prenom = 'Alain';";
try {
    $count = $bdd->exec($req);
}
catch(PDOException $e)
{
    echo $e->getMessage();
}
var_dump ($count);
?>
```

# SUPPRESSION

---

```
<?php
require_once 'login.php';
$req="DELETE FROM acteur WHERE prenom = 'ALAIN';";
try {
    $count = $bdd->exec($req);
}
catch(PDOException $e)
{
    echo $e->getMessage();
}
var_dump ($count);
?>
```



MERCI POUR  
VOTRE ATTENTION

