




# PHP

## PDO Clase 1



## ¿Cuál es el objetivo de hoy?

Ejecutar consultas en nuestra base de datos y obtener información.





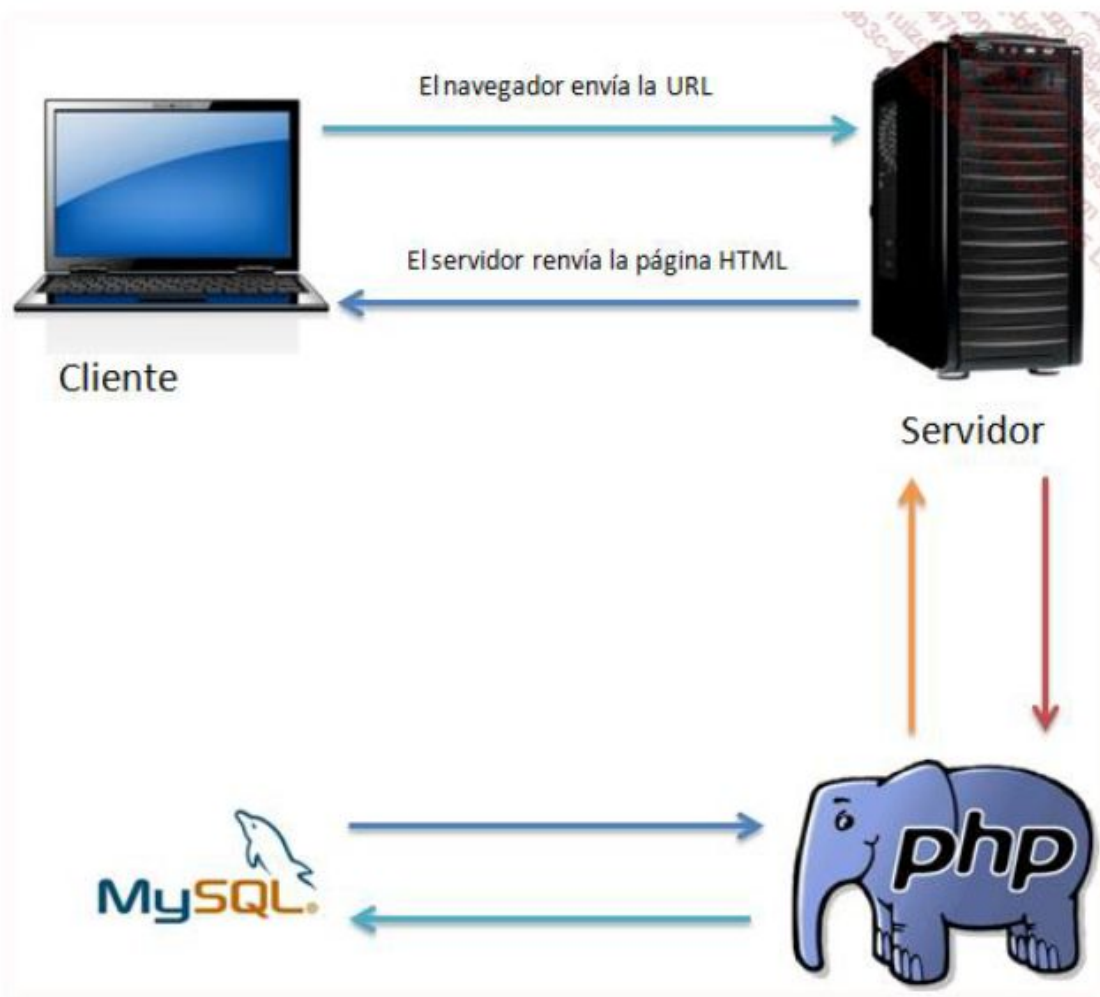
**Conectar nuestra app en PHP a la  
base de datos, ¿por qué?**





**¿Cómo lo hacemos?**







## Hace mucho tiempo...

```
<?php
```

```
    $link = mysql_connect('localhost', 'user', 'pass');
```

```
    mysql_connect('testdb', $link);
```

```
    mysql_set_charset('UTF-8', $link);
```

```
    $query = mysql_query('SELECT * FROM peliculas', $link);
```

```
    $results = mysql_fetch_rows($query);
```

```
...
```



## Todavía...

```
<?php
```

```
$mysqli = new mysqli('127.0.0.1', 'usuario_db', 'password_db', 'nombre_db');
```

```
if ( mysqli_connect_error() ) {
```

```
    printf("Connect failed: ", mysqli_connect_error() );
```

```
    exit();
```

```
}
```





# ¡Ahora!

```
<?php
$db = new PDO(
    'mysql:host = 127.0.0.1; dbname = testdb; port = 3306',
    'username',
    'password'
);
```



# PDO

PHP Data Object



Define una interfaz ligera para poder acceder a cualquier base de datos en PHP. PDO proporciona una capa de abstracción de acceso a datos.





## Ventajas

- ◇ Múltiples Manejadores de Bases de Datos.
- ◇ Soporta últimas actualizaciones SQL:
  - **Reutilizar sentencias.**
  - **Transacciones.**
- ◇ Interfaz más amigable.
- ◇ **Parámetros** para hacer queries optimizados.
- ◇ Más funciones y **Control de errores.**
- ◇ Viene con PHP.



## Abrir una conexión

```
<?php
```

```
$dsn = 'mysql:host=localhost; dbname=movies_db; port=3306';  
$db_user = 'root';  
$db_pass = '123456';  
$opt = [PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION];  
$db = new PDO($dsn, $db_user, $db_pass, $opt);
```

// dsn: Data Source Name



**Si hay errores**  
¿qué hacemos?





# Capturarlos usando:

`if(){ } else { }`

`try{ } catch( ){ }` [Gestion de Errores de PHP](#)



# Capturar excepciones

```
<?php
try {
    $db = new PDO($host, $db_user, $db_pass, $opt);
}
catch( PDOException $Exception ) {
    echo $Exception->getMessage();
}
```

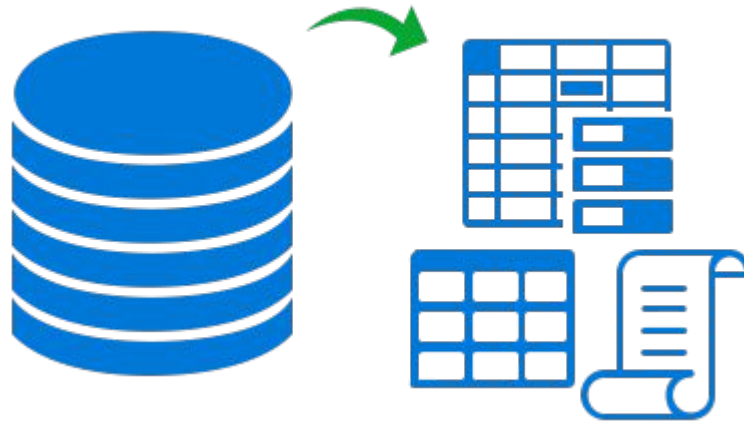
**¡A practicar!**

Conectarse a MySQL



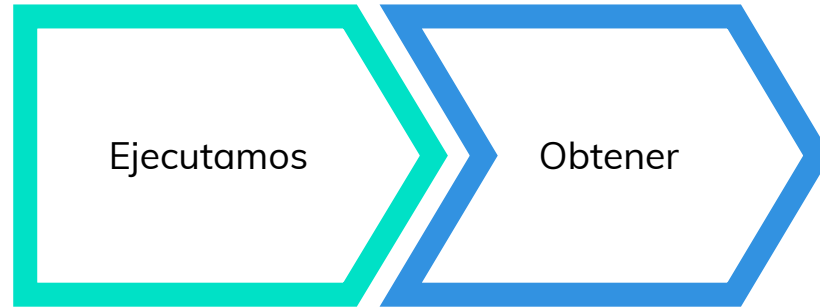
```
<?php  
    new PDO( ... );  
?>
```

**Ahora a obtener datos**






## Query sencillo (pasos)





## El método/función: Query

Ejecuta una sentencia SQL, devolviendo un conjunto de resultados como un “objeto” PDOStatement.



# Ejecutar

```
<?php
    try {
        $db = new PDO($host, $db_user, $db_pass, $opt);
    }
    catch( PDOException $Exception ) {
        echo $Exception->getMessage();
    }

    $query = $db->query('SELECT * FROM movies');
```

## ¿Qué obtenemos?

# Ejecutar

```
<?php
    try {
        $db = new PDO($host, $db_user, $db_pass, $opt);
    }
    catch( PDOException $Exception ) {
        echo $Exception->getMessage();
    }

    $query = $db->query('SELECT * FROM movies');
```

Trás haber ejecutado **new PDO** utilizaremos dicha variable de retorno para ejecutar métodos a base de datos.

# Ejecutar

```
<?php
    try {
        $query = $db->query('SELECT * FROM movies');
    }
    catch( PDOException $Exception ) {
        echo $Exception->getMessage();
    }
}
```

Podemos también capturar **excepciones** cuando el nombre de la tabla o campos no existan, por eso usamos **PDO::ERRMODE\_EXCEPTION** en las opciones.

# Obtener

```
<?php
    try {
        $query = $db->query('SELECT * FROM movies');
    }
    catch( PDOException $Exception ) {
        echo $Exception->getMessage();
    }

    $results = $query->fetchAll(PDO::FETCH_ASSOC);
```



# Obtener

## Métodos

- `fetch` (<http://php.net/manual/en/pdostatement.fetch.php>)
- `fetchAll` (<http://php.net/manual/en/pdostatement.fetchall.php>)
- `rowCount` (<http://php.net/manual/en/pdostatement.rowcount.php>)

## Constantes

(<http://php.net/manual/en/pdo.constants.php>)

- `PDO::FETCH_ASSOC`

# ¡A practicar!

Prácticas con queries

A dark blue computer monitor with a white screen. The screen displays three lines of PHP code. The first line is the opening PHP tag, the second line is a database query, and the third line is the closing PHP tag.

```
<?php
    $db->query("select * from movies");
?>
```





# Statements

Plantillas compiladas para SQL.

- La consulta sólo necesita ser analizada (o preparada) **una** vez, pero puede ser ejecutada muchas veces con los mismos o diferentes parámetros.

[Documentacion](#)



# Statements (pasos)



# Definir

```
<?php
    try {
        $db = new PDO($host, $db_user, $db_pass, $opt);
    }
    catch( PDOException $Exception ) {
        echo $Exception->getMessage();
    }

    $query = $db->prepare('SELECT * FROM movies');
```

# Ejecutar

```
<?php
    try {
        $db = new PDO($host, $db_user, $db_pass, $opt);
    }
    catch( PDOException $Exception ) {
        echo $Exception->getMessage();
    }

    $query = $db->prepare('SELECT * FROM movies');

    $query->execute();
```

# Obtener

```
<?php
    try {
        $db = new PDO($host, $db_user, $db_pass, $opt);
    }
    catch( PDOException $Exception ) {
        echo $Exception->getMessage();
    }

    $query = $db->prepare('SELECT * FROM movies');

    $query->execute();

    // obtenemos los datos
    $results =
    $query->fetchAll(PDO::FETCH_ASSOC);
```

# INSERT

<?php

// Con la \$db ya instanciada

\$sql = "

INSERT INTO movies (title, release\_date)  
VALUES ('Big Fish', 2013)

";  
;

**\$query = \$db->prepare(\$sql);**

\$query->execute(); // primera insercción

\$query->execute(); // ¿segunda insercción?

\$query->execute(); // ¿tercera insercción?

// aplica lo mismo para update y delete

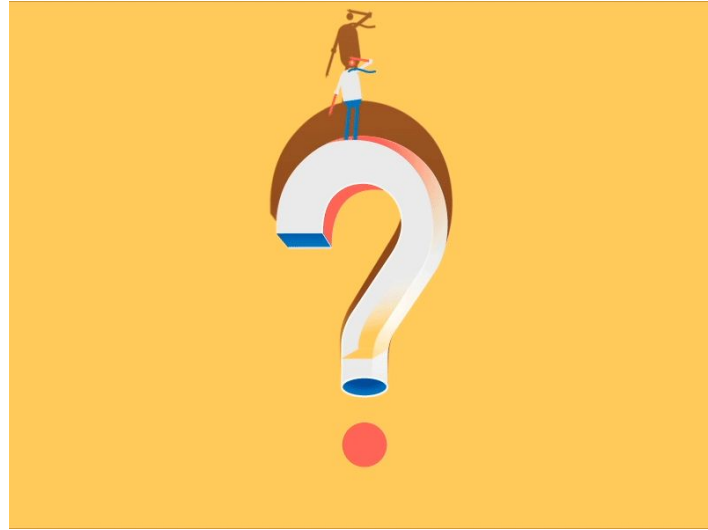
# ¡A practicar!

Statements



```
<?php  
    echo "¡Hora de practicar!";  
?>
```

**¡Gracias!**



**¿Preguntas?**