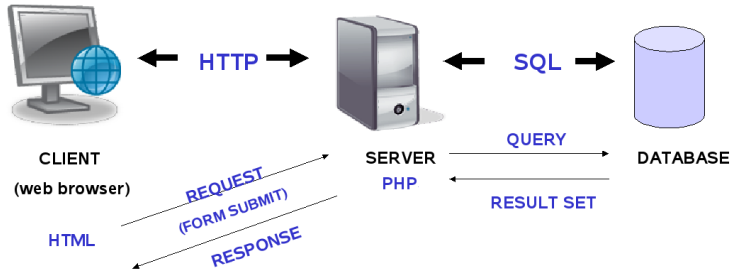
PHP DB use

PHP- DB connection and SQL

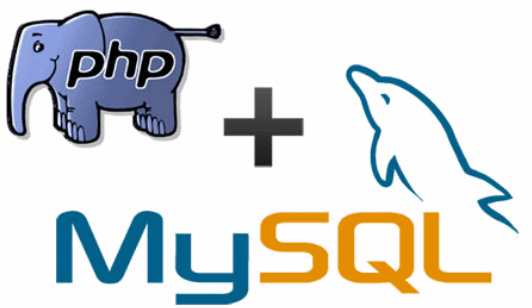
By prof. Igor Kanovsky

Ways to Work with DB

- **MySQLi extension** (the "i" stands for improved)
- **PDO** (PHP Data Objects)
- PDO will work on 12 different database systems, MySQLi is for MySQL only.
- MySQLi is simple effective and may be used in OOP style also.

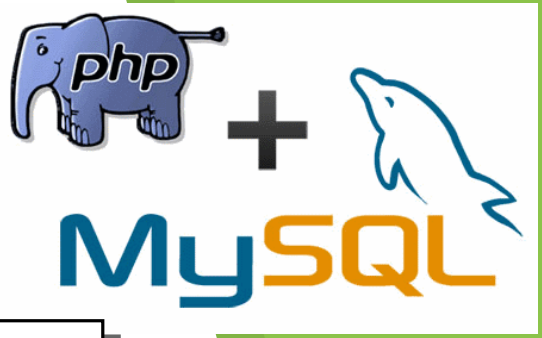


Connection



- **Connection is a resource (object) needed to perform any work with DB**
- To establish connection you have to use DB location, DB user name and password.
- Connection is a “heavy“ resource! But PHP managed a connections pool in optimal way, so you have not a duty to worry about this.

Connection (OO)

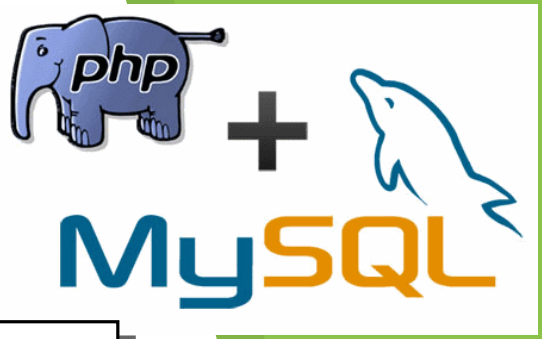



```
<?php

// Create connection
$conn =
new mysqli("localhost", "username", "password ", " DBname");

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully";
?>
```

Connection (procedural)



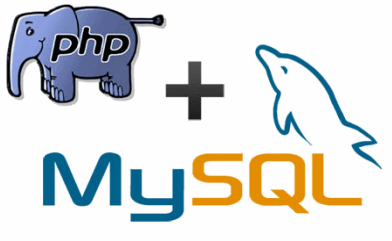

```
<?php

// Create connection
$conn =
mysqli_connect ("localhost", "username", "password‟,”DBname”);

// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
echo "Connected successfully";

?>
```

Execute Queries



Query

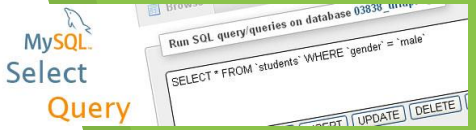
- Write an sql statement as a **string**.
- Run it with query function of the connection.

```
<?php

$sq="INSERT INTO `users`(`name`, `password`) VALUES ('test2','test2')";
$con->query($sq);

?>
```

Select Query



- Check the result.
- For ***select*** statement the result is an **object** (table).
- Each row of the table may be accessed by special **fetch** function

```
<?php
	$sq="SELECT `name`, `password` FROM `users`";
	$res=$con->query($sq);
	if($res->num_rows > 0)
		while($row = $res->fetch_object()){
			echo $row->name.'<br>';
		}
?>
```

Prepared Statements

php MySQL

Query String

- Problem with an sql statement as a **string**:

```
<?php
   $x="test";
   $sql="SELECT * FROM `users` WHERE name='".$x."'";
   $con->query($sql);
?>
```

- Hard to combine data from variables and rules of SQL statement in the case of **complex statement**.
- If the statement is used in loop, the DBMS can optimize the run by storing the command after statement parsing.
- Solution: **prepared statement**.

Prepared Statement

- In SQL string use **?** Instead variable data.
- Prepare the statement on DB with the connection.
- Bind variables to the question marks.
- Set the values and execute the statement.

```
<?php
$sql="INSERT INTO MyGuests (firstname, lastname, email) VALUES (?, ?, ?)"
$stmt = $conn->prepare($sql);
$stmt->bind_param("sss", $firstname, $lastname, $email);
$firstname = "John"; $lastname = "Travolta"; $email = "john@gamil.com";
$stmt->execute();
//the last 2 lines may be repeated with different data
?>
```

Binding

- Binding defined variables.
- Variables type has to be as in DB.
- Variables type is set by string with char for each var:

  i - integer

  d - double

  s - string