

## What is database collation and character set?

A **character set**<sup>1</sup> is a set of symbols and encodings. A **collation** is a set of rules for comparing characters in a character set. Let's make the distinction clear with an example of an imaginary character set.

Suppose that we have an alphabet with four letters: 'A', 'B', 'a', 'b'. We give each letter a number: 'A' = 0, 'B' = 1, 'a' = 2, 'c' = 3. The letter 'A' is a symbol, the number 0 is the encoding for 'A', and the combination of all four letters and their encodings is a character set.

Now, suppose that we want to compare two string values, 'A' and 'B'. The simplest way to do this is to look at the encodings: 0 for 'A' and 1 for 'B'. Because 0 is less than 1, we say 'A' is less than 'B'. Now, what we've just done is apply a collation to our character set. The collation is a set of rules (only one rule in this case): "compare the encodings." We call this simplest of all possible collations a binary collation.

But what if we want to say that the lowercase and uppercase letters are equivalent? Then we would have at least two rules: (1) treat the lowercase letters 'a' and 'b' as equivalent to 'A' and 'B'; (2) then compare the encodings. We call this a case-insensitive collation. It's a little more complex than a binary collation.

In real life, most character sets have many characters: not just 'A' and 'B' but whole alphabets, sometimes multiple alphabets or eastern writing systems with thousands of characters, along with many special symbols and punctuation marks. Also in real life, most collations have many rules: not just case insensitivity but also accent insensitivity (an "accent" is a mark attached to a character as in German 'ö') and multiple-character mappings (such as the rule that 'ö' = 'OE' in one of the two German collations).

	mysql	mysqli	PDO
Introduced	v2.0	v5.0	v5.1
Deprecated	<b>v5.5</b>	-	-
Included with PHP	Yes	Yes	Yes
Pre-configured for MySQL	Yes	Yes	
Other databases supported			Yes
Procedural interface	Yes	Yes	
Object-oriented Interface		Yes	Yes
Prepared statements		Yes	Yes

---

<sup>1</sup><http://stackoverflow.com/questions/341273/what-does-character-set-and-collation-mean-exactly>

PDO (PHP Data Objects) works with any database (PostgreSQL, Oracle, DB2, MySQL)  
To configure PDO, you add a couple of lines to your php.ini file.  
Object-oriented using classes and methods.

Procedural	Object-Oriented
mysqli_connect mysqli_connect_errno mysqli_connect_error mysqli_real_escape_string mysqli_query mysqli_fetch_assoc mysqli_close	\$mysqli = new mysqli \$mysqli->connect_errno \$mysqli->connect_error \$mysqli->real_escape_string \$mysqli->query \$mysqli->fetch_assoc \$mysqli->close

Notice that they're almost identical. The difference with OOP is you are creating a new object and that all of the functions called are inside the object. They are not functions at the root of PHP.

## Choosing an API (Application Programming Interface)

PHP offers three different APIs to connect to MySQL. Below we show the APIs provided by the mysql, mysqli, and PDO extensions. Each code snippet creates a connection to a MySQL server running on "example.com" using the username "user" and the password "password". And a query is run to greet the user.

### Example #1 Comparing the three MySQL APIs

```
<?php
// mysqli
$mysqli = new mysqli("example.com", "user", "password", "database");
$result = $mysqli->query("SELECT 'Hello, dear MySQL user!' AS _message FROM DUAL");
$row = $result->fetch_assoc();
echo htmlentities($row['_message']);

// PDO
$pdo = new PDO('mysql:host=example.com;dbname=database', 'user', 'password');
$statement = $pdo->query("SELECT 'Hello, dear MySQL user!' AS _message FROM DUAL");
$row = $statement->fetch(PDO::FETCH_ASSOC);
echo htmlentities($row['_message']);

// mysql
$c = mysql_connect("example.com", "user", "password");
mysql_select_db("database");
```

```
$result = mysql_query("SELECT 'Hello, dear MySQL user!' AS _message FROM DUAL");  
$row = mysql_fetch_assoc($result);  
echo htmlentities($row['_message']);
```

## Connect to MySQL Using PDO

MySQL is the choice of many web developers and will be used as the database of choice for much of this tutorial. Here's how to connect to a MySQL database.

```
<?php
```

```
/** mysql hostname */
```

```
$hostname = 'localhost';
```

```
/** mysql username */
```

```
$username = 'username';
```

```
/** mysql password */
```

```
$password = 'password';
```

```
try {
```

```
    $dbh = new PDO("mysql:host=$hostname;dbname=mysql", $username, $password);
```

```
    /** echo a message saying we have connected */
```

```
    echo 'Connected to database';
```

```
    }
```

```
catch(PDOException $e)
```

```
{
```

```
    echo $e->getMessage();
```

```
}
```

```
?>
```