PHP DATABASE Basics and Implementation

Dr. Prabu M, Ph.D.,(CSE)
Assistant Professor, Department of Computer Science,
CHRIST (Deemed to be University), Bangalore-560029.

PHP has support for over 20 databases, including the most popular commercial and open source varieties. Relational database systems such as MySQL, PostgreSQL, and Oracle are the backbone of most modern dynamic websites.

Using PHP to Access a Database

There are two ways to access databases from PHP. One is to use a database-specific extension; the other is to use the database-independent PDO (PHP Data Objects) library.

- If you use a database-specific extension, your code is intimately tied to the database you're using.
- PDO, on the other hand, hides the database-specific functions from you with an abstraction layer, so moving between database systems can be as simple as changing one line of your program or your php.ini file.

Relational Databases and SQL

- A Relational Database Management System (RDBMS) is a server that manages data for client.
- The data is structured into tables, where each table has a number of columns, each of which has a name and a type.
- PHP communicates with relational databases such as MySQL and Oracle using the Structured Query Language (SQL). You can use SQL to create, modify, and query relational databases.

- The syntax for SQL is divided into two parts. The first, Data Manipulation Language or DML, is used to retrieve and modify data in an existing database. DML is remarkably compact, consisting of only four actions or verbs: SELECT, INSERT, UPDATE, and DELETE.

- The set of SQL commands used to create and modify the database structures that hold the data is known as Data Definition Language, or DDL.

PHP Data Objects

- The PHP Data Objects (PDO) extension defines a lightweight, consistent interface for accessing databases in PHP.
- Each database driver that implements the PDO interface can expose database-specific features as regular extension functions.
- Note that you cannot perform any database functions using the PDO extension by itself;
- you must use a database-specific PDO driver to access a database server.

PDO has (among others) these unique features:

- PDO is a native C extension.
- PDO takes advantage of the latest PHP 5 internals.
- PDO uses buffered reading of data from the result set.
- PDO gives common DB features as a base.
- PDO is still able to access DB-specific functions.
- PDO can use transaction-based techniques.
- PDO can interact with LOBS (Large Objects) in the database.
- PDO can use prepared and executable SQL statements with bound parameters. PDO can implement scrollable cursors.
- PDO has access to SQLSTATE error codes and has very flexible error handling

Making a connection

The first thing that is required for PDO is that you make a connection to the database and hold that connection in a connection handle variable, as in the following code:

```
$db = new PDO ($dsn, $username, $password);
```

```
$db = new PDO("mysql:host=localhost;dbname=library", "petermac", "abc123");
```

PDO and prepared statements

PDO also allows for what are known as prepared statements. This is done with PDO calls in stages or steps. Consider the following code:

```php
$statement = $db->prepare( "SELECT * FROM books");
$statement->execute();

// gets rows one at a time
while ($row = $statement->fetch()) {
  print_r($row);
  // or do something more meaningful with each returned row
}

$statement = null;
```

```php
$statement = $db->prepare("INSERT INTO books (authorid, title, ISBN, pub_year)"
  . "VALUES (:authorid, :title, :ISBN, :pub_year)");

$statement->execute(array(
  'authorid' => 4,
  'title'    => "Foundation",
  'ISBN'     => "0-553-80371-9",
  'pub_year' => 1951)
);
```

Here, we prepare the SQL statement with four named placeholders: authorid, title,ISBN, and pub_year. These happen to be the same names as the columns in the database.

Transactions

- Some RDBMS support transactions, in which a series of database changes can be committed (all applied at once) or rolled back (discarded, with none of the changes applied to the database).

- For example, when a bank handles a money transfer, the withdrawal from one account and deposit into another must happen together—neither should happen without the other, and there should be no time between the two actions.

- PDO handles transactions elegantly with try...catch structures. Check the code in next slide:

```php
try {
  $db = new PDO("mysql:host=localhost;dbname=banking_sys", "petermac", "abc123");
  // connection successful
}
catch (Exception $error) {

  die("Connection failed: " . $error->getMessage());
}

try {
  $db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
  $db->beginTransaction();

  $db->exec("insert into accounts (account_id, amount) values (23, '5000')" );
  $db->exec("insert into accounts (account_id, amount) values (27, '-5000')" );

  $db->commit();
}
catch (Exception $error) {
  $db->rollback();
  echo "Transaction not completed: " . $error->getMessage();
}
```

MySQLi Object Interface

- The most popular database platform used with PHP is the MySQL database.
- If you look at the MySQL website (www.mysql.com/) you will discover that there are a few different versions of MySQL you can use.
- We will look at the freely distributable version known as the community server.
- PHP has a number of different interfaces to this database tool as well, so we will look at the object-oriented interface known as MySQLi.

```php
$db = new mysqli("localhost", "petermac", "1q2w3e9i8u7y", "library");

$sql = "INSERT INTO books (authorid, title, ISBN, pub_year, available)
  VALUES (4, 'I, Robot', '0-553-29438-5', 1950, 1)";

if ($db->query($sql)) {
    echo "Book data saved successfully.";
}
else {
    echo "INSERT attempt failed, please try again later, or call tech support" ;
}

$db->close();
```

Retrieving Data for Display

```php
$db = new mysqli("localhost", "petermac", "1q2w3e9i8u7y", "library");

$sql = "SELECT a.name, b.title FROM books b, authors a
    WHERE a.authorid=b.authorid";
$result = $db->query($sql);

while ($row = $result->fetch_assoc()) {
  echo "{$row['name']} is the author of: {$row['title']}<br />";
}

$result->close();

$db->close();
```

The output would look like this:

```
J.R.R. Tolkien is the author of: The Two Towers
J.R.R. Tolkien is the author of: The Return of The King
J.R.R. Tolkien is the author of: The Hobbit
Alex Haley is the author of: Roots
Tom Clancy is the author of: Rainbow Six
Tom Clancy is the author of: Teeth of the Tiger
Tom Clancy is the author of: Executive Orders
...
```

MySQL

MySQL is the old database driver

Deprecated as of PHP 5.5.0

Procedural and manual escaping

MySQLi

MySQLi is the Improved driver

Included with PHP 5 and later

Object-oriented interface

Support for Prepared Statements

Support for Multiple Statements

Support for Transactions

SQLite is a relational database management system (RDBMS) contained in a C library. In contrast to many other database management systems, SQLite is not a client–server database engine. Rather, it is embedded into the end program.

Developer(s)	:	D. Richard Hipp
Initial release	:	17 August 2000; 19 years ago
Stable release	:	3.31.0 (January 22, 2020)
Written in	:	C
Operating system	:	Cross-platform
Type	:	RDBMS (embedded)
License	:	Public domain
Website	:	sqlite.org

SQLite library books table

```
$db = new SQLiteDatabase("c:/copy/library.sqlite");

$sql = "CREATE TABLE 'books' ('bookid' INTEGER PRIMARY KEY,
   'authorid' INTEGER,
   'title' TEXT,
   'ISBN' TEXT,
   'pub_year' INTEGER,
   'available' INTEGER)";

if ($db->queryExec($sql, $error) == FALSE) {
    echo "Create Failure - {$error}<br />";
}
else {
    echo "Table Books was created<br />";
}
```

```
$sql = <<<SQL
INSERT INTO books ('authorid', 'title', 'ISBN', 'pub_year', 'available')
VALUES (1, 'The Two Towers', '0-261-10236-2', 1954, 1);

INSERT INTO books ('authorid', 'title', 'ISBN', 'pub_year', 'available')
VALUES (1, 'The Return of The King', '0-261-10237-0', 1955, 1);

INSERT INTO books ('authorid', 'title', 'ISBN', 'pub_year', 'available')
VALUES (2, 'Roots', '0-440-17464-3', 1974, 1);

INSERT INTO books ('authorid', 'title', 'ISBN', 'pub_year', 'available')
VALUES (4, 'I, Robot', '0-553-29438-5', 1950, 1);

INSERT INTO books ('authorid', 'title', 'ISBN', 'pub_year', 'available')
VALUES (4, 'Foundation', '0-553-80371-9', 1951, 1);
SQL;

if (!$db->queryExec($sql, $error)) {
  echo "Insert Failure - {$error}<br />";
}
else {
  echo "INSERT to Books - OK<br />";
}
```

SQLite select books

```php
$db = new SQLiteDatabase("c:/copy/library.sqlite");

$sql = "SELECT a.name, b.title FROM books b, authors
                        a WHERE a.authorid=b.authorid";


$result = $db->query($sql);

while ($row = $result->fetch()) {
    echo "{$row['a.name']} is the author of: {$row['b.title']}<br/>";
}
```

- The database for modern applications MongoDB is a general purpose, document-based, distributed database built for modern application developers and for the cloud era. No database makes you more productive.

MongoDB Atlas

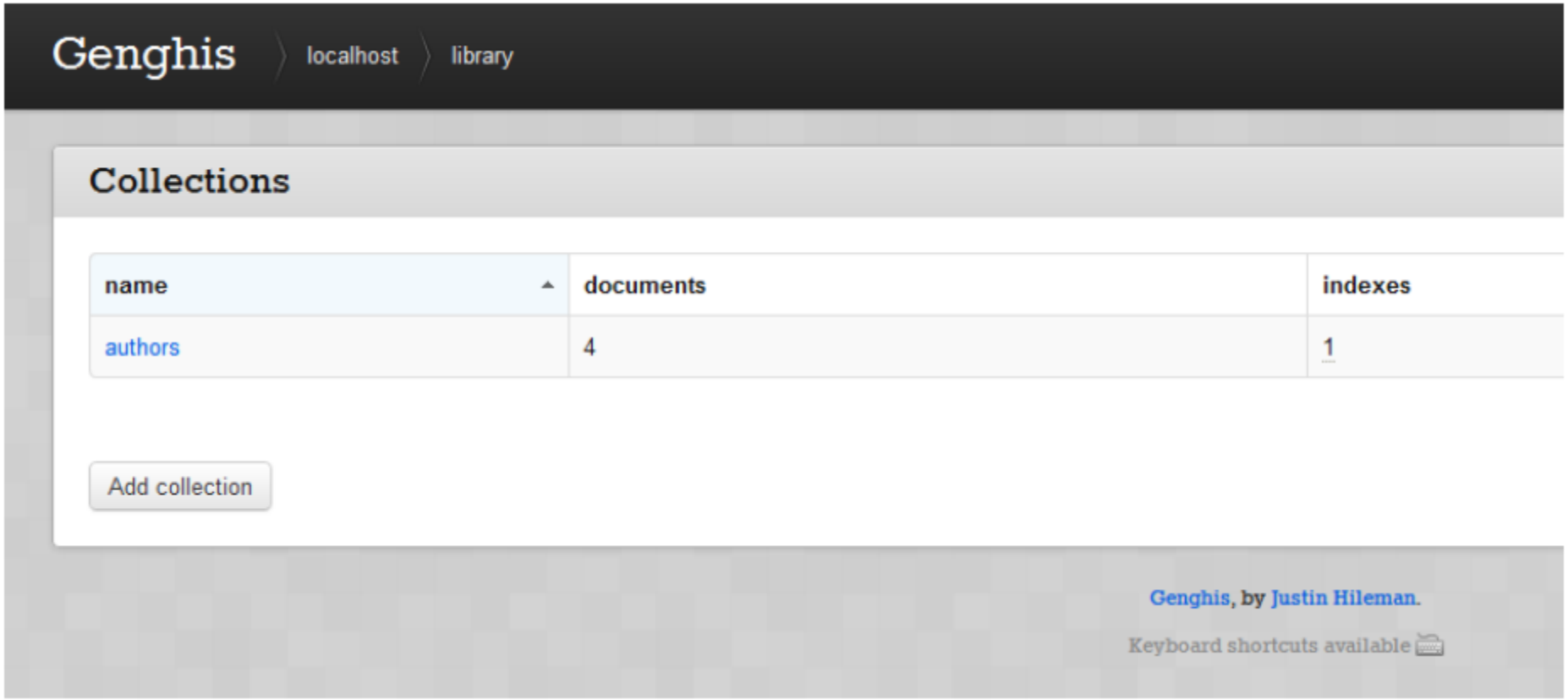
- Move faster with a cloud MongoDB service. Built for agile teams who'd rather spend time building apps than managing databases. Available on AWS, Azure, and GCP.

Developer(s)	:	MongoDB Inc.
Initial release	:	February 11, 2009;
Stable release	:	4.2.2 / 9 December 2019;
Repository	:	github.com/mongodb/mongo
Written in	:	C++, Go, JavaScript, Python
Operating system	:	Windows Vista and later, Linux, OS X 10.7 and later, Solaris, FreeBSD
Type	:	Document-oriented database
License	:	Source-available
Website	:	www.mongodb.com

Main features

- Ad hoc queries
- Indexing
- Replication
- Load balancing
- File storage
- Aggregation
- Server-side JavaScript execution
- Capped collections
- Transactions

A very useful web-based tool for browsing Mongo data and manipulating the collections and documents is known as **Genghis.**



Now let's get into some sample code. Take a look at the following code in Example to see the beginnings of a Mongo database taking shape.

MongoDB library

```
$mongo = new Mongo();
$db = $mongo->library;
$authors = $db->authors;

$author = array('authorid' => 1, 'name' => "J.R.R. Tolkien");
$authors->insert($author);

$author = array('authorid' => 2, 'name' => "Alex Haley");
$authors->insert($author);

$author = array('authorid' => 3, 'name' => "Tom Clancy");
$authors->save($author);

$author = array('authorid' => 4, 'name' => "Isaac Asimov");
$authors->save($author);
```

Retrieving Data

MongoDB data selection example

```php
$mongo = new Mongo();
$db = $mongo->library;
$authors = $db->authors;

$data = $authors->findone(array('authorid' => 4));

echo "Generated Primary Key: {$data['_id']}<br />";
echo "Author name: {$data['name']}";
```

REFERENCES :

- The Joy of PHP Programming: A Beginner's Guide – by Alan Forbes.
- PHP & MySQL Novice to Ninja – by Kevin Yank.
- Head First PHP & MySQL – by Lynn Beighley & Michael Morrison.
- Learning PHP, MySQL, JavaScript, and CSS: A Step-by-Step Guide to Creating Dynamic Websites – by Robin Nixon.
- PHP & MySQL Web Development – by Luke Welling & Laura Thompson.