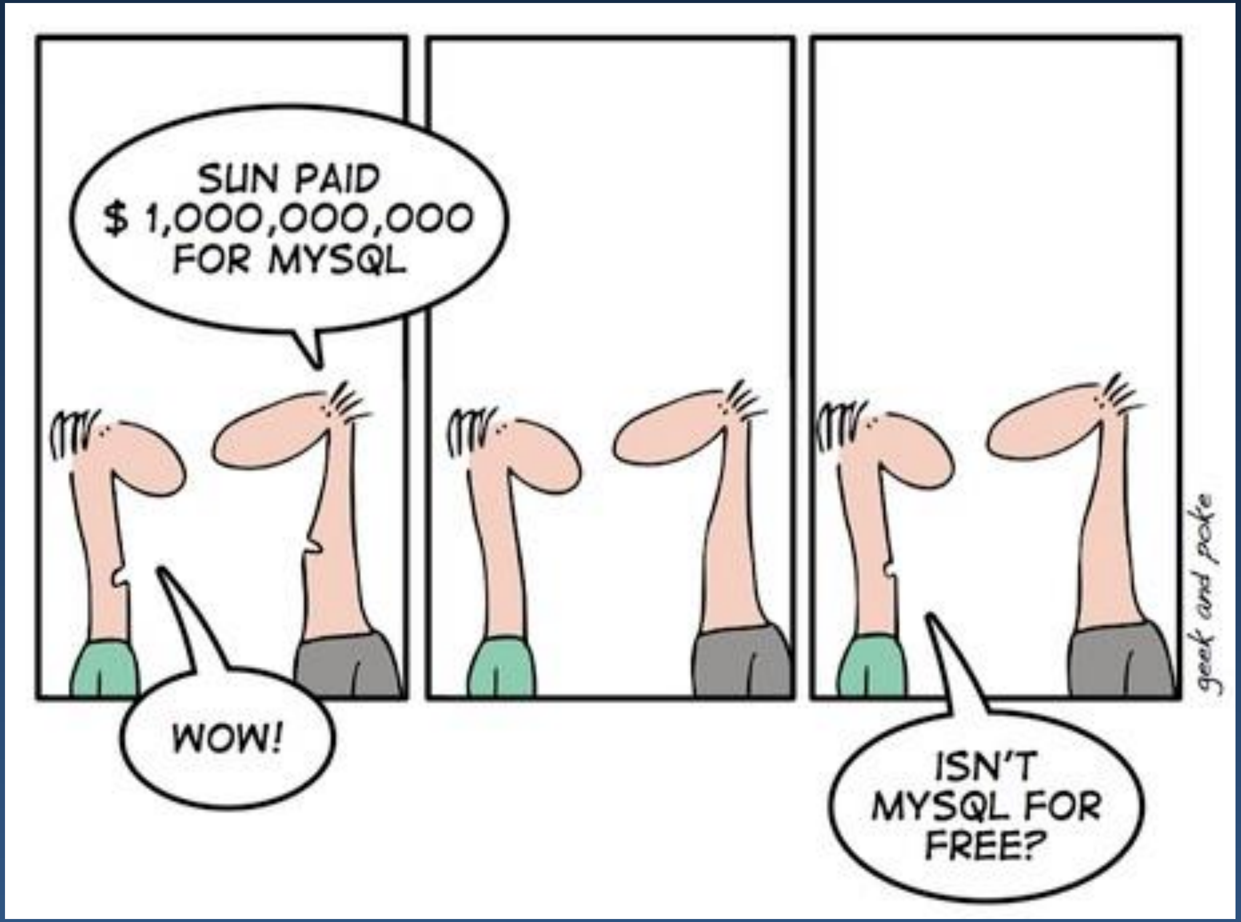
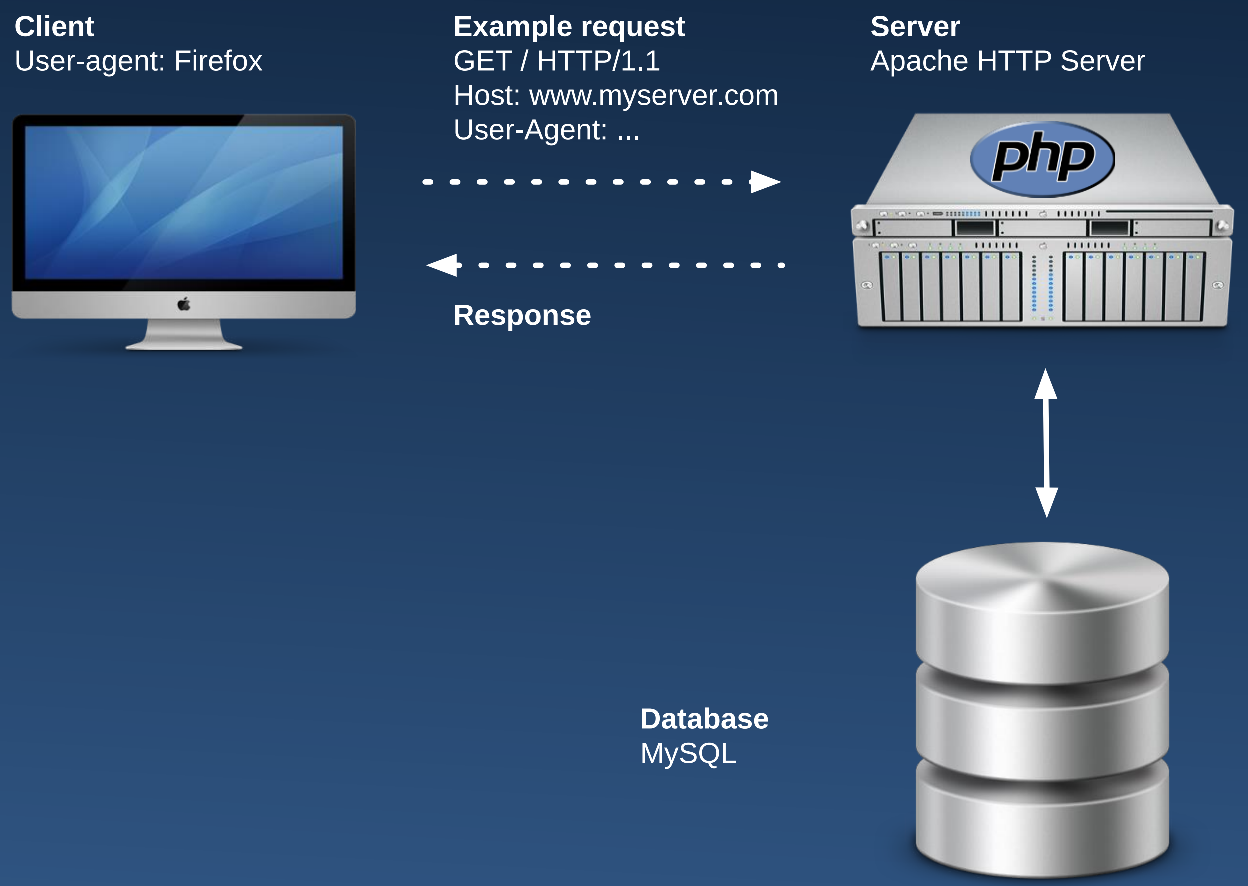


SUN PAID $ 1,000,000,000 FOR MYSQL
WOW!
ISN'T MYSQL FOR FREE?
geek and poke

Web Technology

Lecture 18: PHP3: Connecting to Databases using PDO

Three-tiered Web Site: LAMP



Client
User-agent: Firefox

Example request
GET / HTTP/1.1
Host: www.myserver.com
User-Agent: ...

Server
Apache HTTP Server

Response

Database
MySQL

Built-in Functions

- Built-in functions are pre-made pieces of code that are executed by a call to the function.
- PHP has a **LARGE** set of functions
  - File System
  - Mail
  - Audio, video and image Manipulation
  - Date and Time
  - Compression
  - Credit card processing
  - Cryptography
  - Database

http://www.php.net/manual/en/

MySQL

- MySQL is a Relational Database Management System (RDBMS).
- Data is stored in database objects called tables.
- A table is a collections of related data entries and it consists of columns and rows.
- Databases are useful when storing information categorically. A company may have a database with the following tables: "Employees", "Products", "Customers" and "Orders".

Database Tables

- A database most often contains one or more tables. Each table is identified by a name (e.g. "Customers" or "Orders"). Tables contain records (rows) with data.
- Below is an example of a table called ”Customers":

LastName	FirstName	Address	City
Red	John	10 High Street	London
White	Paul	5 Union Street	Aberdeen
Green	Adam	2 King Street	Edinburgh

- The table above contains three records (one for each customer) and four columns (LastName, FirstName, Address, and City).

Why use a database?

- Data could also be stored as files

- Databases are optimised for storing large amounts of data
- Databases can support concurrent operations
- Databases can be more secure
- Databases allow relationships to be established across tables
  - Outwith the scope of this course

Why use a database (2)

- Databases are optimized so that their data can be **queried** efficiently
- Ideal for data storage on a website as you may hold a large amount of data that is accessed by hundreds (or thousands, or tens of thousands) of users at a time
- Queries can be written in many forms, but one of the most common ways is to send SQL (Structured Query Language) requests to the database

Queries

- Queries are how data is retrieved from the database
- With MySQL, we can query a database for specific information and have a recordset returned.
- Look at the following query:

```
 SELECT LastName FROM Customers
```

- The query above selects all the data in the "LastName" column from the "Customers" table, and will return a recordset like this:

LastName
Red
White
Green

Other SQL Statements

- SELECT
  - extracts data from a database
- UPDATE
  - updates data in a database
- DELETE
  - deletes data from a database
- INSERT INTO
  - inserts new data into a database

Connecting to MySQL in PHP

- Old way: the PHP mysql_ library
  - Also known as mysql_connect
  - Deprecated, insecure and is therefore **evil**
- MySQLi (MySQL Improved)
  - Upgrade to the old mysql library
- PDO (PHP Data Objects)
  - Object oriented
  - Support for other database vendors (SQLite, Sequel Server, Oracle)

Objects in PHP

- Similar to other programming languages, objects keep related data together
- Define the object and the fields it has:

```
class Dog {
    $name;
    $breed;
    $shoes_eaten;
}
```

- Then make a new instance of this object:

```
$puppy = new Dog();
$puppy->name = 'Rover';
$puppy->breed = 'mongrel';
$puppy->shoes_eaten = 6;

echo $puppy->name; //will print Rover
```

Create a Connection to a MySQL DB

- Before you can access data in a database, you must create a connection to the database, which is a new PDO object:

```
$db = new PDO('mysql:host=hostname;dbname=dbname;charset=utf8mb4',
   $username, $password);

$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

Example Connection

```
<?php
$username = 'anythingotherthanroot';
$password = 'cheeseandbiscuitsarereallytastyandsoarelongcredentials';
$host = 'mysql.abdn.ac.uk';
$dbname = 'lovelycheeses';



$db = new PDO("mysql:host=$host;dbname=$dbname;charset=utf8mb4", $username,
   $password);
$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

// some code
?>
```

Catching connection errors

- What if your database connection fails?
- Surround your database connection with try…catch block to handle them elegantly

```
try {
    $db = new PDO('mysql:host=host;dbname=db;charset=utf8mb4', $username, $password);
    $db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
} catch (PDOException $ex} {
    //die($ex); use this in development only
    redirect('brokensite.php'); //send user to an error page
}
```

- AVOID printing the details of the exception on production servers!

Select Data from a Database Table

- The SELECT statement is used to select data from a database.

```
SELECT column_name(s)
FROM table_name
```

- To learn more about SQL:

  http://www.w3schools.com/sql/default.asp

Reminder: Associative Arrays

- PHP allows arrays to be indexed by strings rather than numbers
- Useful for interacting with databases

```
$stmt = $db->query('SELECT * FROM customers');
$results = $stmt->fetchAll(PDO::FETCH_ASSOC);
```

- $results is an array of associative arrays!
  - Each internal associative array corresponds to a single table row

The $results array

Our $results array contains all this data

$results[0] is an internal associative array containing the data from the first row of the table

$results[0]['LastName'] is "Red"

LastName	FirstName	Address	City
Red	John	10 High Street	London
White	Paul	5 Union Street	Aberdeen
Green	Adam	2 King Street	Edinburgh

Getting data from the $results array

- What if we want to get at all of the data in the $results array, row by row
- We could hard code it, e.g. $results[0]['firstName']
- But what if we don't know how many rows there are, we just want the data from each row until there are no more rows

- Easy solution: We loop!

Looping

```
foreach ($results as $row) {
   echo $row['FirstName'];
   echo $row['LastName'];
   echo $row['Address'];
   echo $row['City'];
}
```

We know the keys of each internal array ($row) as they match the column name in the database.

Query Example

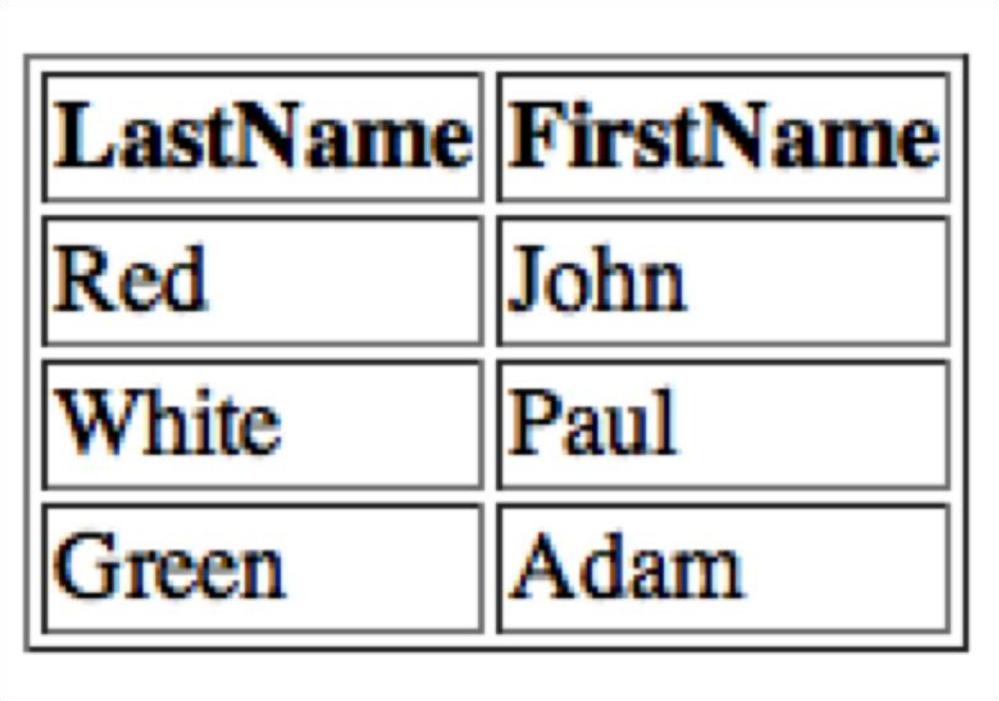
```php
<?php $username = 'charlie'; $password = 'ilikecatsandknitting';
try {
    $db = new PDO('mysql:host=host;dbname=dbname;charset=utf8mb4',    $username,
    $password);
    $db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $stmt = $db->query('SELECT * FROM customers');
    $results = $stmt->fetchAll(PDO::FETCH_ASSOC);
} catch (PDOException $ex) {
    die($ex); //again, fine for development, not for production!
}
foreach($results as $row) {
  echo $row['FirstName'] . ' ' . $row['LastName'];
  echo '<br />';
}
?>
```

Display Results in a HTML Table

```
<?php
$username = 'charlie'; $password = 'ilikecatsandknitting';
try {
    $db = new PDO('mysql:host=mysql.abdn.ac.uk;dbname=custdb;charset=utf8mb4', $username, $password);
    $db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $stmt = $db->query('SELECT * FROM customers');
    $results = $stmt->fetchAll(PDO::FETCH_ASSOC);
} catch (PDOException $ex) { die($ex); }
?>
<table>
  <tr>
  <th>FirstName</th> <th>LastName</th>
  </tr>

<?php foreach ($results as $row): ?>
 <tr>
 <td><?=$row['FirstName']?></td>
 <td><?=$row['LastName']?></td>
 </tr>
<?php endforeach; ?>

</table>
```



LastName	FirstName
Red	John
White	Paul
Green	Adam

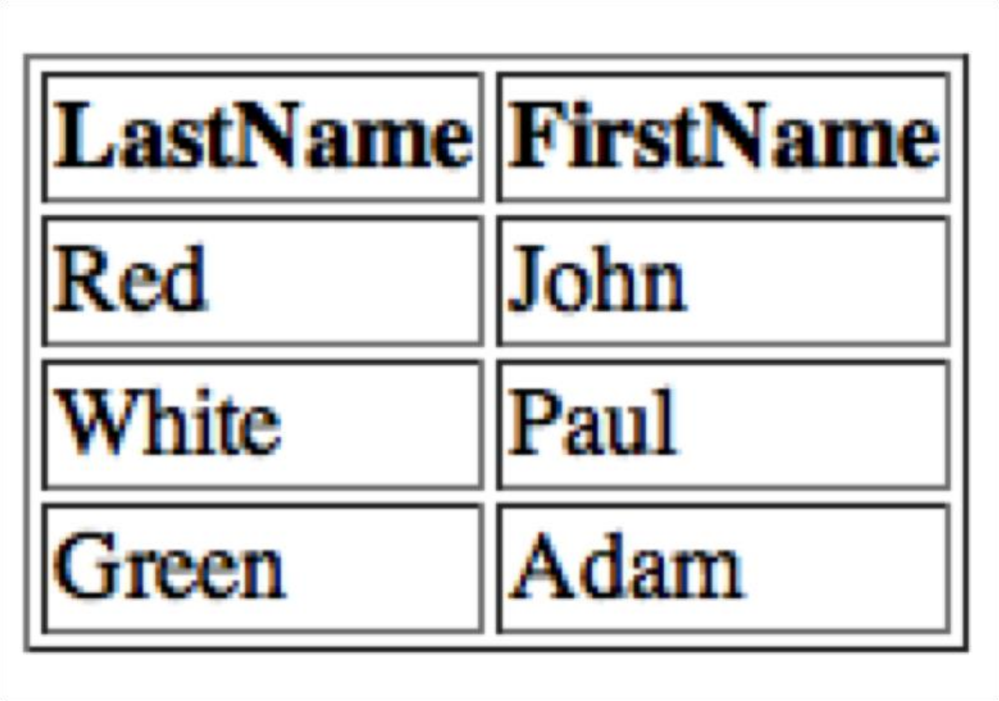
Getting the column names

- In the previous example, we hard-coded the column names on our outputted HTML table.
- What if we wanted to determine the **names** of the columns automatically from the database name?
- We can use array_keys() to do this

Display Results in a HTML Table

```
<?php
$username = 'charlie'; $password = 'ilikecatsandknitting';
try {
    $db = new PDO('mysql:host=host;dbname=dbname;charset=utf8mb4', $username, $password);
    $db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $stmt = $db->query('SELECT * FROM customers');
    $results = $stmt->fetchAll(PDO::FETCH_ASSOC);
} catch (PDOException $ex) { die($ex); }
?>
<table>
  <tr>
 <?php foreach (array_keys($results[0]) as $header): ?>
    <th><?=$headers?></th>
 <?php endforeach; ?>
  </tr>

<?php foreach ($results as $row): ?>
 <tr>
 <td><?=$row['FirstName']?></td>
 <td><?=$row['LastName']?></td>
 </tr>
<?php endforeach; ?>
</table>
```



LastName	FirstName
Red	John
White	Paul
Green	Adam

Retrieving objects

- You can also retrieve data as an array of **objects** rather than associative arrays
- Column names will be the **member variable** of the objects

- $result->FirstName;
- $result->LastName;

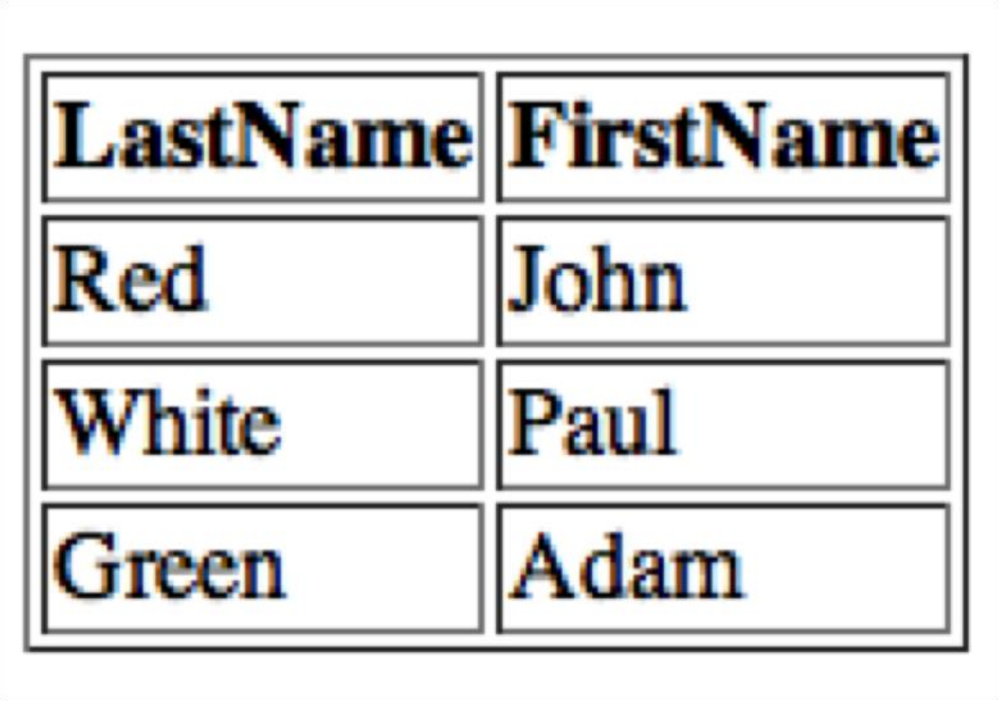
- Etc.

Display Results in a HTML Table

```
<?php
$username = 'charlie'; $password = 'ilikecatsandknitting';
try {
    $db = new PDO('mysql:host=mysql.abdn.ac.uk;dbname=custdb;charset=utf8mb4', $username, $password);
    $db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $stmt = $db->query('SELECT * FROM customers');
    $results = $stmt->fetchAll(PDO::FETCH_OBJ);
} catch (PDOException $ex) { die($ex); }
?>
<table>
   <tr>
   <th>FirstName</th> <th>LastName</th>
   </tr>

<?php foreach ($results as $row): ?>
 <tr>
 <td><?=$row->FirstName?><td>
 <td><?=$row->LastName?></td>
 </tr>
<?php endforeach; ?>

</table>
```



LastName	FirstName
Red	John
White	Paul
Green	Adam

Getting the number of rows returned

- Sometimes getting the total number of rows returned from a query is useful

- Safest approach is to use the PHP's count function on the $results array

```
<p>Total number of employees: <?=count($results)?></p>
```

Working with the University MySQL server

- The university's MySQL server is mysql.abdn.ac.uk

- This server is not available outside the university network

- If you are working on home computers, transfer your scripts to your public_html folder on your H:\ drive to see them work.

Transferring your files to H:\

- The university provides SFTP access on ftpweb.abdn.ac.uk
- Note: FTP will **not** work!

- On Windows, use WinSCP
- On Mac, use FileZilla or Fetch (paid)
- Linux: Use the command line or FileZilla

- Note: as of 2016 the university no longer provides an SSH server for students