

CSY2028

Web Programming

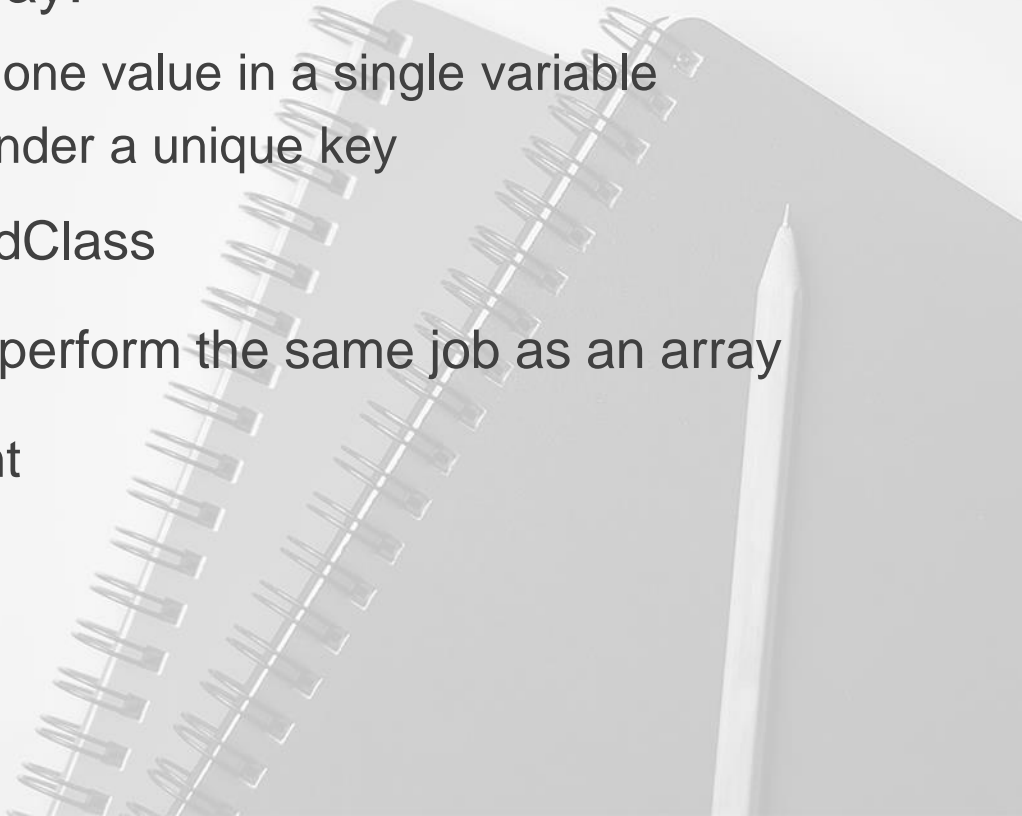


Introduction to MySQL

- Crash course in Object-Oriented programming
- Introduction to MySQL
 - Connecting to MySQL using XAMPP
 - Creating a table
- Connecting to MySQL with PHP



What is an Object?

- An object is similar to an array:
 - You can store more than one value in a single variable
 - Each variable is stored under a unique key
 - PHP includes a class called `stdClass`
 - `stdClass` can often be used to perform the same job as an array
 - However, the syntax is different
- 

Object Syntax

- To use stdClass you have to create an instance
- To create an instance you must use the new keyword

```
$myObj = new stdClass;
```

- This creates an instance
- You can store values inside the object using the operator - >

```
$myObj = new stdClass;  
$myObj->property1 = 'value 1';  
$myObj->property1 = 'value 2';
```

Objects vs Arrays

- Objects can usually be used to solve the same problems that arrays can

```
<?php
$myObj = new stdClass;

$myObj->property1 = 'value 1';

$myObj->property1 = 'value 2';

echo $myObj->property1; //Prints "value 1"
?>
```

```
<?php
$myArray = [];

$myArray['property1'] = 'value 1';

$myArray['property2'] = 'value 2';

echo $myArray['property1']; //Prints "value 1"
?>
```

- With one major difference: Objects cannot have numeric keys

Objects vs Arrays

- Objects and arrays differ in one major way, when you pass an object into a method it passes in the same instance. Arrays are passed as a copy

```
<?php
function processArray($array) {
    $array['property2'] = 'value changed';

    echo '<p>Array in function:</p>';
    var_dump($array);
}

$myArray = [];
$myArray['property1'] = 'value 1';
$myArray['property2'] = 'value 2';

echo '<p>Original Array:</p>';
var_dump($myArray);

processArray($myArray);

echo '<p>Array after processing:</p>';
var_dump($myArray);
?>
```

Original Array:

```
array (size=2)
  'property1' => string 'value 1' (length=7)
  'property2' => string 'value 2' (length=7)
```

Array in function:

```
array (size=2)
  'property1' => string 'value 1' (length=7)
  'property2' => string 'value changed' (length=13)
```

Array after processing:

```
array (size=2)
  'property1' => string 'value 1' (length=7)
  'property2' => string 'value 2' (length=7)
```

Objects vs Arrays

- When an object is passed, it is not copied. Changes to the array in a function will be reflected elsewhere in the program

```
<?php
function processObject($object) {
    $object->property2 = 'value changed';

    echo '<p>Object in function:</p>';
    var_dump($object);
}
```

```
$object = new stdClass;
$object->property1 = 'value 1';
$object->property2 = 'value 2';

echo '<p>Original Object:</p>';
var_dump($object);

processObject($object);

echo '<p>Object after processing:</p>';
var_dump($object);
```

Original Object:

```
object(stdClass)[1]
  public 'property1' => string 'value 1' (length=7)
  public 'property2' => string 'value 2' (length=7)
```

Object in function:

```
object(stdClass)[1]
  public 'property1' => string 'value 1' (length=7)
  public 'property2' => string 'value changed' (length=13)
```

Object after processing:

```
object(stdClass)[1]
  public 'property1' => string 'value 1' (length=7)
  public 'property2' => string 'value changed' (length=13)
```

Looping Over Objects

- Like arrays, you can use foreach to loop over the different properties:

```
<?php  
  
$object = new stdClass;  
$object->property1 = 'value 1';  
$object->property2 = 'value 2';  
$object->property3 = 'value 3';  
  
foreach ($object as $name => $value) {  
    echo '<p>' . $name . ' is set to ' . $value . '</p>';  
}  
  
?>
```

property1 is set to value 1

property2 is set to value 2

property3 is set to value 3

Objects

- Objects can also contain functions.
- The stdClass has no functions, only properties
- Those functions work on data stored inside the object
- This avoids passing lots of arguments into a function repeatedly, the information is “encapsulated” inside the object and does not need to be sent to the function each time
- PHP includes a class called DateTime which can be used to represent dates and times
- This is similar to the date function but has more functionality

DateTime

- When you create an instance of the DateTime class its date is set to today's date
- You can display the date using the format function
- To call a function, you use the arrow operator -> and the function name, followed by arguments:

```
<?php  
  
$date = new DateTime();  
  
echo $date->format('d/m/Y');  
  
?>
```

Output:
14/11/2016

DateTime

- Each DateTime instance can be used to represent a single date and time.
- The date can be defined when you create the instance by passing it in as an argument:
- When a new instance is created, it can be passed values as arguments. You can use the instance name like a function

```
<?php
$date = new DateTime('2012-02-01');
echo $date->format('d/m/Y');
?>
```

Output:
01/02/2012

Objects

- The current date is stored inside the object. When you print out the date, the date is “remembered” by the object and all you need to tell it is how to format the date
- You can set the date once, and use it multiple times

```
<?php  
  
$date = new DateTime('2012-02-01');  
  
echo $date->format('d/m/Y');  
  
echo $date->format('m-d-Y');  
  
?>
```

Output:
01/02/2012
2012-02-01

Objects

- As the date is stored inside the object. You can adjust the date using the modify function
- Because the date is already stored, you only have to pass it information about how the date is changing:

```
<?php
$date = new DateTime('2016-11-20');

echo $date->format('d/m/Y');

$date->modify('-2 Days');

echo $date->format('d/m/Y');

$date->modify('-2 Days');

echo $date->format('d/m/Y');

?>
```

Output:

```
03/05/2015
01/05/2015
29/04/2015
```

Objects

- Objects are used to represent a real-world concept (e.g. a date)
- This can be one or more related values (e.g. day/month/year to represent a date)
- As well as some behaviour for modifying or retrieving those values
 - e.g. subtracting or adding to the date
 - Or formatting the date using a specific format

Objects

- You can create multiple instances that hold different values
- For example you can use the DateTime class to simultaneously represent two different dates
- Each instance is entirely independent from each other. Making changes to one instance will only affect that instance

```
$date1 = new DateTime('2016-05-04');  
$date2 = new DateTime('2016-05-03');  
  
echo '<p>Date 1 is ' . $date1->format('d/m/Y') . '</p>';  
echo '<p>Date 2 is ' . $date2->format('d/m/Y') . '</p>';  
  
$date2->modify('2 weeks');  
  
echo '<p>Date 1 is ' . $date1->format('d/m/Y') . '</p>';  
echo '<p>Date 2 is ' . $date2->format('d/m/Y') . '</p>';
```

Output:

```
<p>Date 1 is 04/05/2016</p>  
<p>Date 2 is 03/05/2016</p>  
<p>Date 1 is 04/05/2016</p>  
<p>Date 2 is 17/05/2016</p>
```

MySQL Database

- A database server (in our case, MySQL) is a program that can store large amounts of information in an organized format that's easily accessible through programming languages like PHP
- To connect to a MySQL database you need a MySQL client
- A MySQL client lets you view and alter the contents of the database
- We will be using PHPMYADMIN of XAMPP to access database

MySQL Database

Start
MySQL
service in
XAMPP

XAMPP Control Panel v3.2.2 [Compiled: Nov 12th 2015]

XAMPP Control Panel v3.2.2

Service	Module	PID(s)	Port(s)	Actions
<input type="checkbox"/>	Apache	1916 9292	80, 443	Stop Admin Config Logs
<input type="checkbox"/>	MySQL	8736	3306	Stop Admin Config Logs
<input type="checkbox"/>	FileZilla			Start Admin Config Logs
<input type="checkbox"/>	Mercury			Start Admin Config Logs
<input type="checkbox"/>	Tomcat			Start Admin Config Logs

Config Netstat Shell Explorer Services Help Quit

9:30:09 PM [main] Initializing Control Panel
9:30:09 PM [main] Windows Version: Windows 8.1 Pro 64-bit
9:30:09 PM [main] XAMPP Version: 7.1.9
9:30:09 PM [main] Control Panel Version: 3.2.2 [Compiled: Nov 12th 2015]
9:30:09 PM [main] You are not running with administrator rights! This will work for most application stuff but whenever you do something with services there will be a security dialogue or things will break! So think about running this application with administrator rights!
9:30:09 PM [main] XAMPP Installation Directory: "d:\xampp7\
9:30:09 PM [main] Checking for prerequisites
9:30:09 PM [main] All prerequisites found
9:30:09 PM [main] Initializing Modules
9:30:09 PM [main] Starting Check-Timer
9:30:09 PM [main] Control Panel Ready

Go to <http://localhost/phpmyadmin>

The screenshot shows the phpMyAdmin interface in a web browser. The browser's address bar displays `localhost/phpmyadmin/`. The interface includes a sidebar on the left with a tree view of databases, including `admin`, `bookmaster`, `chapagain`, `csy2028ab`, `csy2028adb`, `csy2028c`, `hrm`, `information_schema`, `kws`, `laravelproject`, `laravelprojecta`, `laravelprojectb`, `laravelprojectc`, `laravelprojectd`, `laravelprojecte`, `malikaincorporate_backup`, `malikawebsite`, `messages_demo`, and `municipality`. The main content area is titled "Server: 127.0.0.1" and contains several tabs: `Databases`, `SQL`, `Status`, `User accounts`, `Export`, `Import`, and `More`. The `Databases` tab is active, showing three panels:
1. **General settings**: Displays "Server connection collation" set to `utf8mb4_unicode_ci`.
2. **Appearance settings**: Displays "Language" set to `English`, "Theme" set to `pmahomme`, and "Font size" set to `82%`. A "More settings" link is also present.
3. **Database server**: Lists server details:

- Server: 127.0.0.1 via TCP/IP
- Server type: MariaDB
- Server version: 10.1.26-MariaDB - mariadb.org binary distribution
- Protocol version: 10
- User: root@localhost
- Server charset: UTF-8 Unicode (utf8)

4. **Web server**: Lists web server details:

- Apache/2.4.27 (Win32) OpenSSL/1.0.1j PHP/7.1.9
- Database client version: libmysql - mysqlnd 5.0.12-dev - 20150407 - \$Id: b396954eeb2d1d9ed7902b8bae237b6\$
- PHP extension: mysqli, curl, mbstring
- PHP version: 7.1.9

phpmyadmin

- Before you can store any data, you must create a schema
- A schema is also called a database. Each database can store tables
- To create a database, click on the databases tab and create a database

Admin

to search all X

Server: 127.0.0.1

Databases SQL Status User accounts Export Import

Databases

Create database ?

csy2028 Collation Create

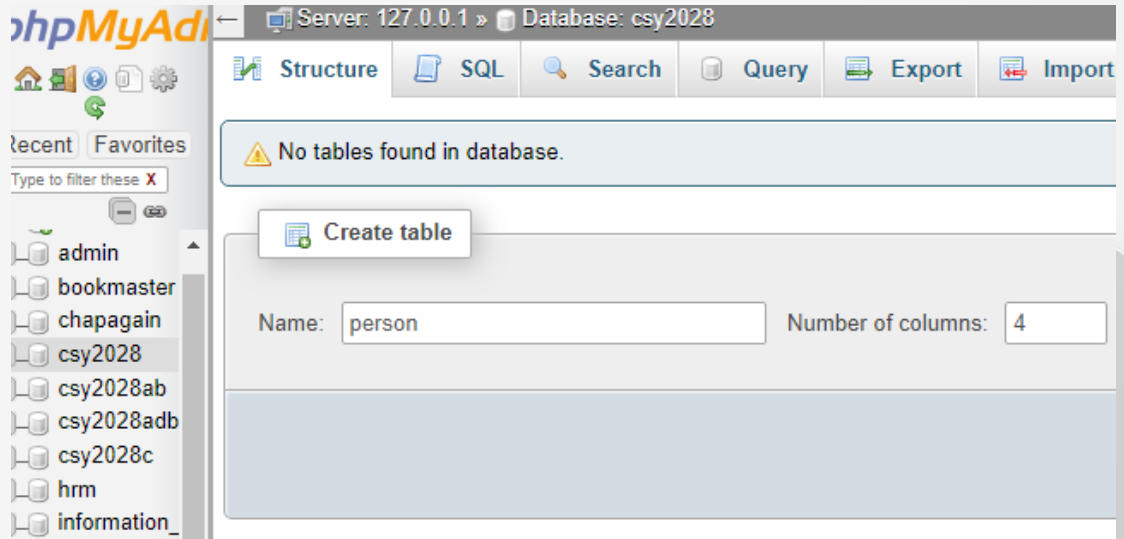
Filters

Containing the word:

Database	Collation	Action
<input type="checkbox"/> admin	latin1_swedish_ci	Check privileges
<input type="checkbox"/> bookmaster	latin1_swedish_ci	Check privileges
<input type="checkbox"/> chapagain	latin1_swedish_ci	Check privileges

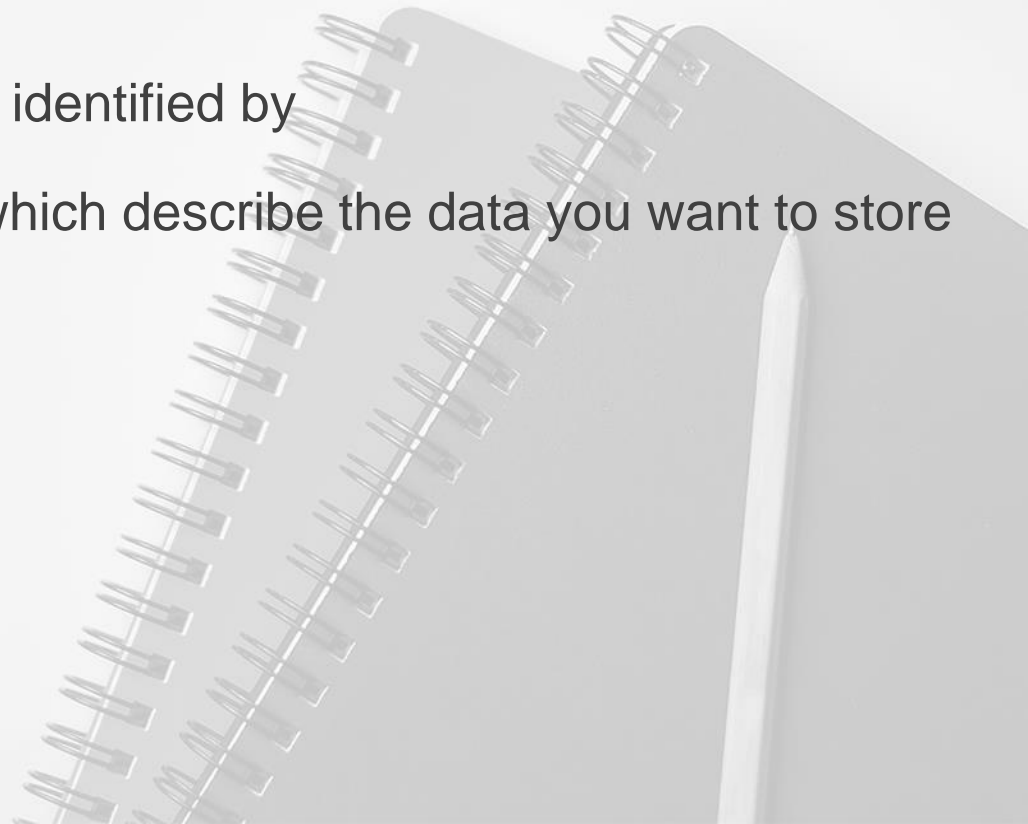
phpmyadmin

- Now you have a schema you can create a table
- Database tables can be used to store the information used by your program
- To create a table:
Click on your database and enter a table name, select number of fields and click on Go



phpmyadmin

- Each table consists of:
 - A unique name that it is identified by
 - One or more columns which describe the data you want to store

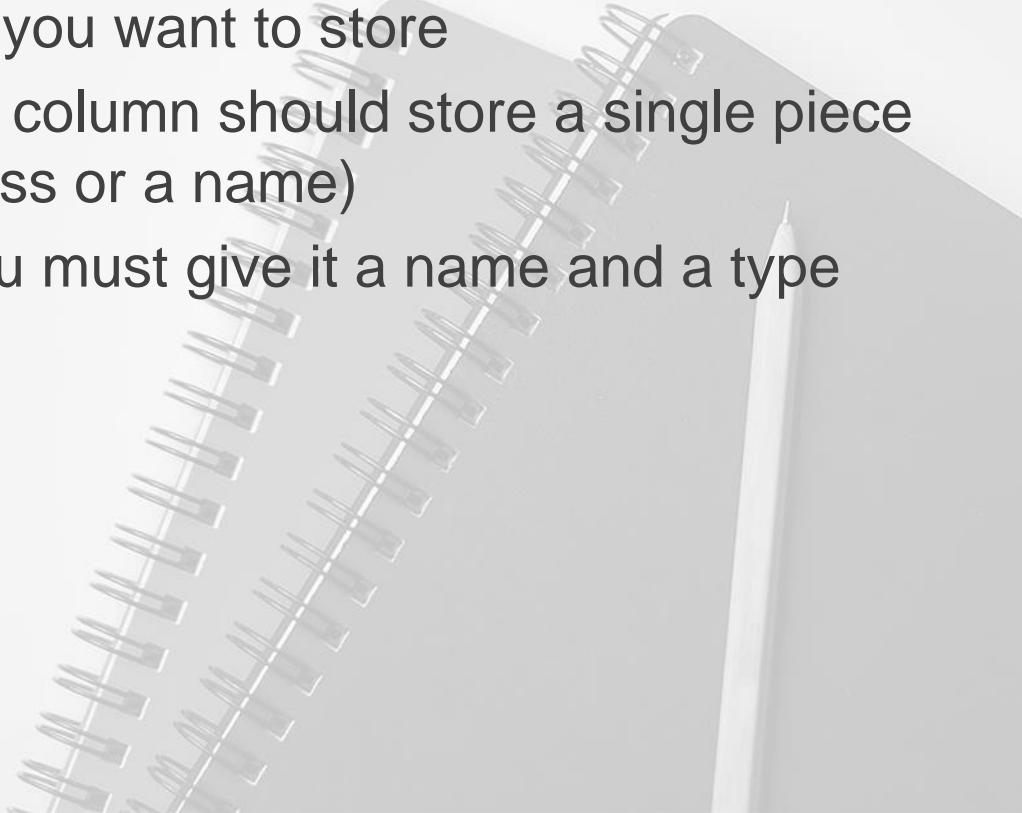


MySQL Tables

- For example to store information about a person you might want to store their:
 - First name
 - Surname
 - Birth date
 - E-mail address



MySQL Tables

- A table describes the structure of the data with columns for each of the pieces of information you want to store
 - When creating a table each column should store a single piece of data (e.g. an email address or a name)
 - When you add a column you must give it a name and a type
- 

MySQL Tables

phpMyAdmin

Server: 127.0.0.1 » Database: csy2028 » Table: person

Browse Structure SQL Search Insert Export Import Privileges More

Table name: Add column(s)

Name	Type ?	Length/Values ?	Default ?	Collation	Attributes
<input type="text"/>	INT ▼	<input type="text"/>	None ▼	<input type="text"/>	<input type="text"/>
Pick from Central Columns					
<input type="text"/>	INT ▼	<input type="text"/>	None ▼	<input type="text"/>	<input type="text"/>
Pick from Central Columns					
<input type="text"/>	INT ▼	<input type="text"/>	None ▼	<input type="text"/>	<input type="text"/>
Pick from Central Columns					
<input type="text"/>	INT ▼	<input type="text"/>	None ▼	<input type="text"/>	<input type="text"/>
Pick from Central Columns					

MySQL Types

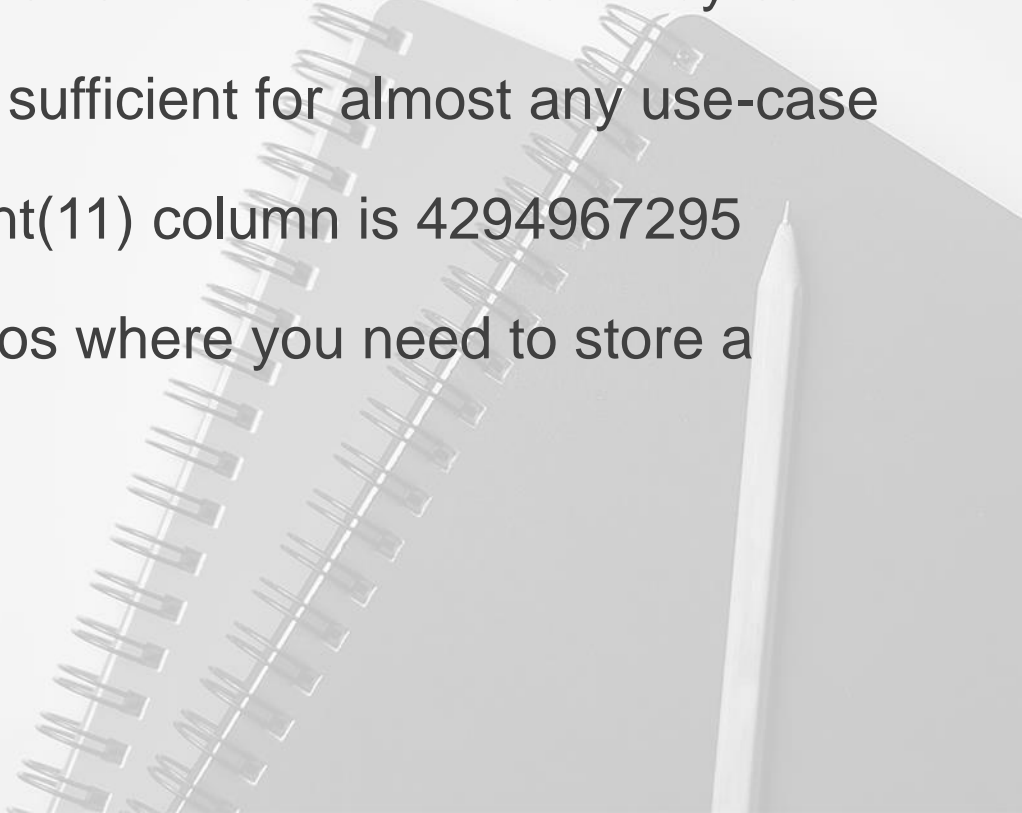
- There are many types in MySQL, however you usually only need to choose one of the following 3:
 - VARCHAR – for strings
 - INT – for numbers
 - DATETIME – for dates



MySQL VARCHAR

- VARCHAR means “Variable number of characters”
- When creating a VARCHAR column you must specify a size
- The size is the number of characters that can be stored in the column
- This is a maximum number of characters!
- The default is 255. This is suitable for most strings:
 - Names, email addresses, etc
 - But may not be suitable for all strings e.g. comments or long links

MySQL INT


- MySQL integers also take a size. This is a number of bytes.
 - The default is 11 and this is sufficient for almost any use-case
 - The maximum value of an int(11) column is 4294967295
 - (There are very few scenarios where you need to store a number larger than that!)
- 

MySQL DATETIME

- Unlike most types in MySQL the DATETIME type does not take a size
- Dates are stored in the format
 - YYYY-MM-DD HH:MM:SS
 - e.g. 2016-11-03 12:32:54



person table

- To create the person table the following columns will be required:
 - firstname – VARCHAR(255)
 - surname – VARCHAR(255)
 - birthday – DATETIME
 - email – VARCHAR(255)
 - These can all be added using mhpmyadmin
- 

person table

- It's good practice to give each table a primary key this is something that is unique to each record
- For the person table this could be the email address as each person will have a unique email address
- To set a primary key in MySQL workbench check the “PK” box on the create table screen
- You should also check the “NN” box. This means that the fields cannot be empty when a record is added. Each person must have an email address

person table

Browse Structure SQL Search Insert Export

Table structure Relation view

	#	Name	Type	Collation	Attributes	Null	Default	Comm
<input type="checkbox"/>	1	firstname	varchar(255)	latin1_swedish_ci		No	None	
<input type="checkbox"/>	2	surname	varchar(255)	latin1_swedish_ci		No	None	
<input type="checkbox"/>	3	birthDate	date			No	None	
<input type="checkbox"/>	4	email	varchar(255)	latin1_swedish_ci		No	None	

☐ Check all With selected: Browse Change Drop

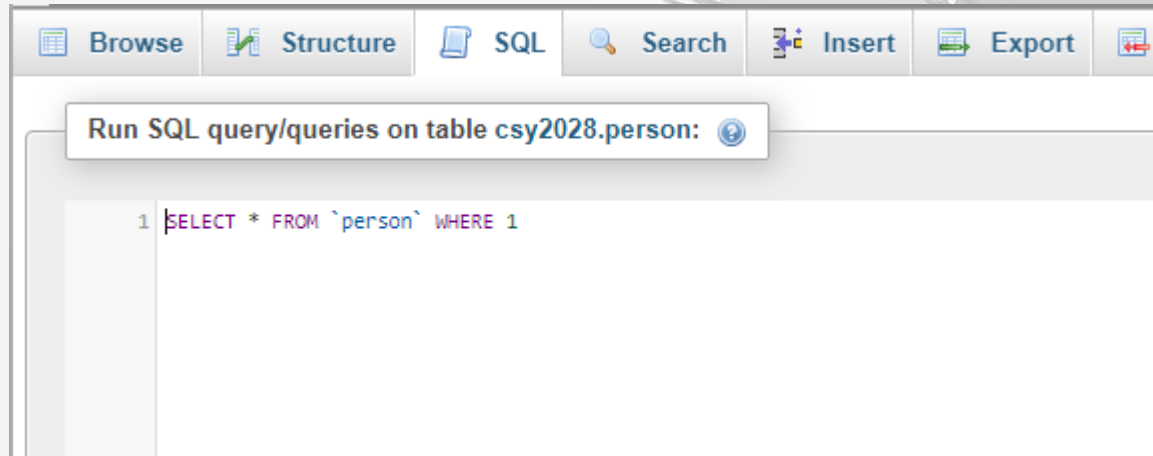
Add to central columns Remove from central columns

Print Propose table structure Track table Move columns

Add column(s)

Selecting Records

- Once you have data in the database you can use queries to search for the data
- This is done using a SELECT statement
- To select all the records in a table you can use the query
- `SELECT * FROM table_name`
- To run a query in phpMyAdmin, click on the SQL tab and write the query and execute



Selecting Records

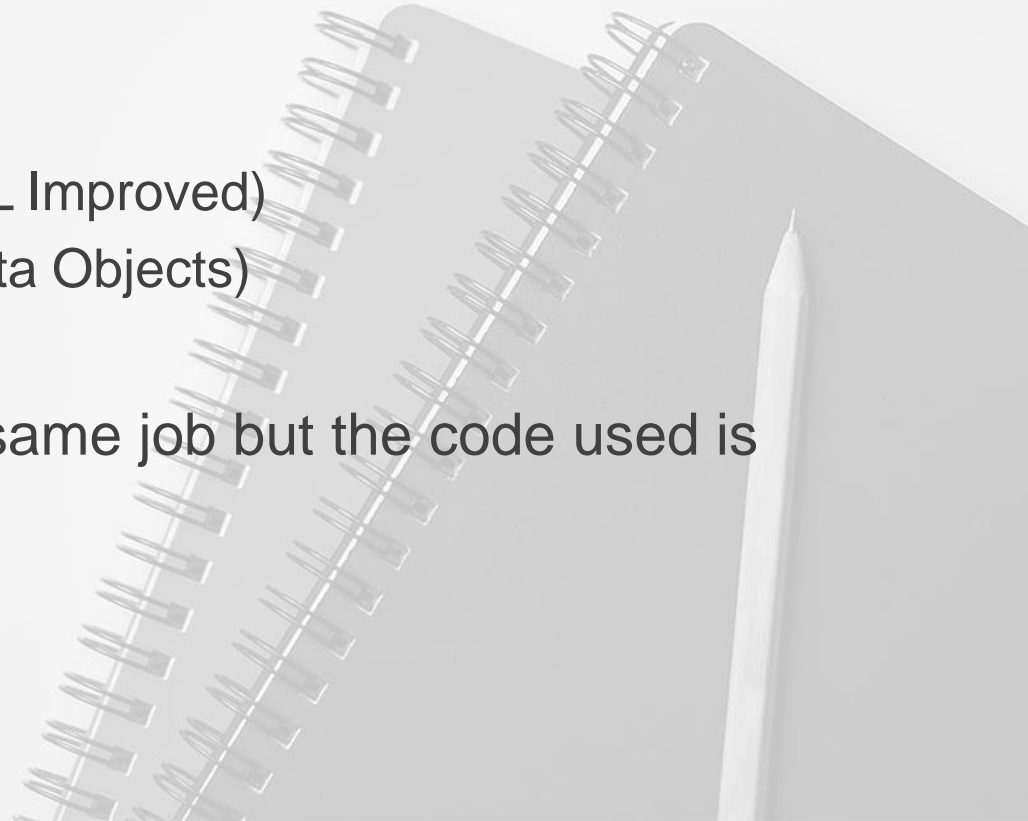
- You can add criteria to a SELECT query to only select specific records that match some search criteria
- `SELECT * FROM person WHERE surname = 'Smith';`
- Will only return records which have the surname field set to Smith

Exercise 1

1. Create the `person` table from the previous slides and insert at least 5 records. Give at least two people the same surname
2. Run a `SELECT` query to find people with a specific surname. Search for at least 3 different surnames in the table
3. Optional – Create a second table called films with the fields:
 - Title
 - Year of release (int)
 - Director
 - Genre
4. Add at least 10 films to the table, with at least 3 by the same director and 5 with the same genre
5. Use SELECT queries to filter the records to search for films of a specific Genre or Director

Connecting to MySQL From PHP

- There are three different libraries that can be used in PHP to connect to MySQL
 - mysql
 - mysqli (short for MySQL Improved)
 - PDO (short for PHP Data Objects)
- They all essentially do the same job but the code used is different for each one!



MySQL

- The MySQL extension is the oldest way of connecting to MySQL from PHP and is now very out of date
- You may come across code examples using this online (e.g. w3Schools) , however they will not work with recent PHP versions
- If you see code using the function `mysql_connect()` to connect to the database look for a newer example (the date of the article will be a giveaway, it will probably be from about 2001!)
- `mysql_connect()`, `mysql_query()` and similar functions are no longer supported by PHP. If you use them they will not work unless you're using an old version of PHP. If you use them in the assignment you will get a very low grade because I will not be able to run your work!

MySQLI

- MySQLI, short for MySQL Improved is an updated method of connecting from PHP to MySQL
- It offers several features and security enhancements that are not available in the basic MySQL extension
- However, it's not used frequently any more as it's been superseded by PDO
- It's more difficult to use than PDO and is less flexible so there's little reason to choose it.

PDO

- PDO stands for PHP Data Objects and isn't specific to MySQL
- It allows you to connect to any kind of database e.g.
 - MySQL
 - SQLite
 - Microsoft SQL Server
 - Oracle
- The commands for PDO are not specific to the database, unlike the mysqli extension
 - If you learn how to use PDO to connect to MySQL you can use almost identical code to connect to other databases
- Using PDO generally means writing less code as well!
- Most PHP projects and developers choose PDO for these reasons.
- See also:
 - <https://www.sitepoint.com/re-introducing-pdo-the-right-way-to-access-databases-in-php/>
 - <https://code.tutsplus.com/tutorials/pdo-vs-mysqli-which-should-you-use--net-24059>
 - <https://www.conetix.com.au/blog/why-you-should-use-pdo-instead-mysqli>

PDO

- Connecting to MySQL from PDO needs the same information as connecting from MySQL workbench:
 - A server address
 - A username
 - A password
- However it also needs a schema name



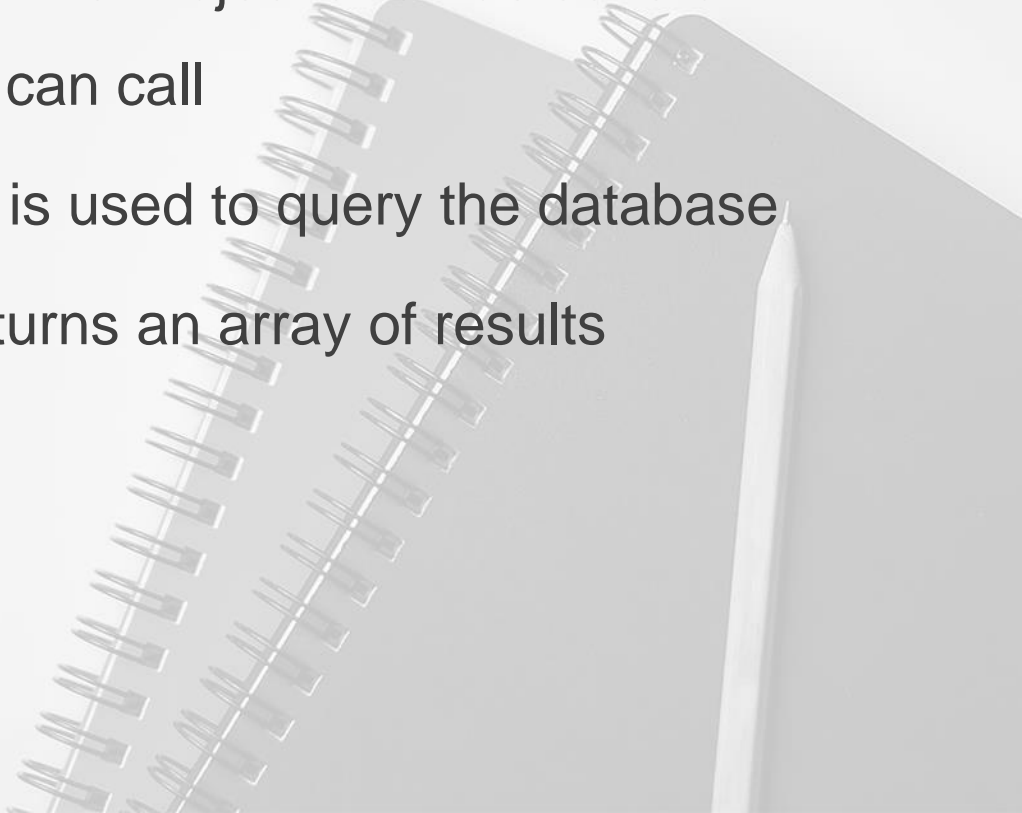
PDO

- To connect to the database you can use the following code:

```
$server = 'v.je';  
$username = 'student';  
$password = 'student';  
  
//The name of the schema we created earlier in MySQL workbench  
//If this schema does not exist you will get an error!  
$schema = 'csy2028';  
  
$pdo = new PDO('mysql:dbname=' . $schema . ';host=' . $server, $username, $password,  
    [ PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION]);
```

- The last part (PDO::ATTR_ERRMODE => PDO_ERRMODE_EXEPTION) is important! Without it, the database will not show any errors and it will be very difficult to see if you've made any mistakes

PDO

- PDO is an object like `DateTime` object and has several functions available that you can call
 - One of them is query which is used to query the database
 - Query takes a string and returns an array of results
- 

PDO

- To run the query `SELECT * FROM person` that lists all the people in the database use the query function on the `$pdo` object
- This will store all the records in the `$results` variable

```
$server = 'v.je';
$username = 'student';
$password = 'student';

//The name of the schema we created earlier in MySQL workbench
//If this schema does not exist you will get an error!
$schema = 'csy2028';

$pdo = new PDO('mysql:dbname=' . $schema . ';host=' . $server, $username, $password,
    [ PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION]);

$results = $pdo->query('SELECT * FROM person');
```

PDO

- You can loop through each result using a foreach loop:

```
$results = $pdo->query('SELECT * FROM person');  
foreach ($results as $row) {  
}
```

- The \$row variable is an array which represents the record
- The keys are the field names and the values are the field values

PDO

- To print out each person's first name you can read the firstname key from the \$row array:
- The keys in the array are created from the column names in the database table

```
$results = $pdo->query('SELECT * FROM person');  
  
foreach ($results as $row) {  
    echo '<p>' . $row['firstname'] . '</p>';  
}
```

Output:

Dave

John

Jo

Sue

PDO

- Each field from the table is represented in the \$row array:
 - firstname
 - surname
 - email
 - birthday



PDO

- Each field from the table is represented in the \$row array:
 - firstname, surname, email, birthday

```
$results = $pdo->query('SELECT * FROM person');  
foreach ($results as $row) {  
    echo '<p>' . $row['firstname'] . ' ' . $row['surname'] . ' was born on ' . $row['birthday'] . '</p>';  
}
```

Output:

```
Dave Smith was born on 1993-12-23 00:00:00  
John Smith was born on 1991-02-25 00:00:00  
Jo Richards was born on 1989-03-03 00:00:00  
Sue Evans was born on 1984-09-04 00:00:00
```

PDO

- PDO queries support anything supported by the database e.g. WHERE clauses:

```
$results = $pdo->query('SELECT * FROM person WHERE surname = "Smith"');  
  
foreach ($results as $row) {  
    echo '<p>' . $row['firstname'] . ' ' . $row['surname'] . ' was born on ' . $row['birthday'] . '</p>';  
}
```

Output:
Dave Smith was born on 1993-12-23 00:00:00

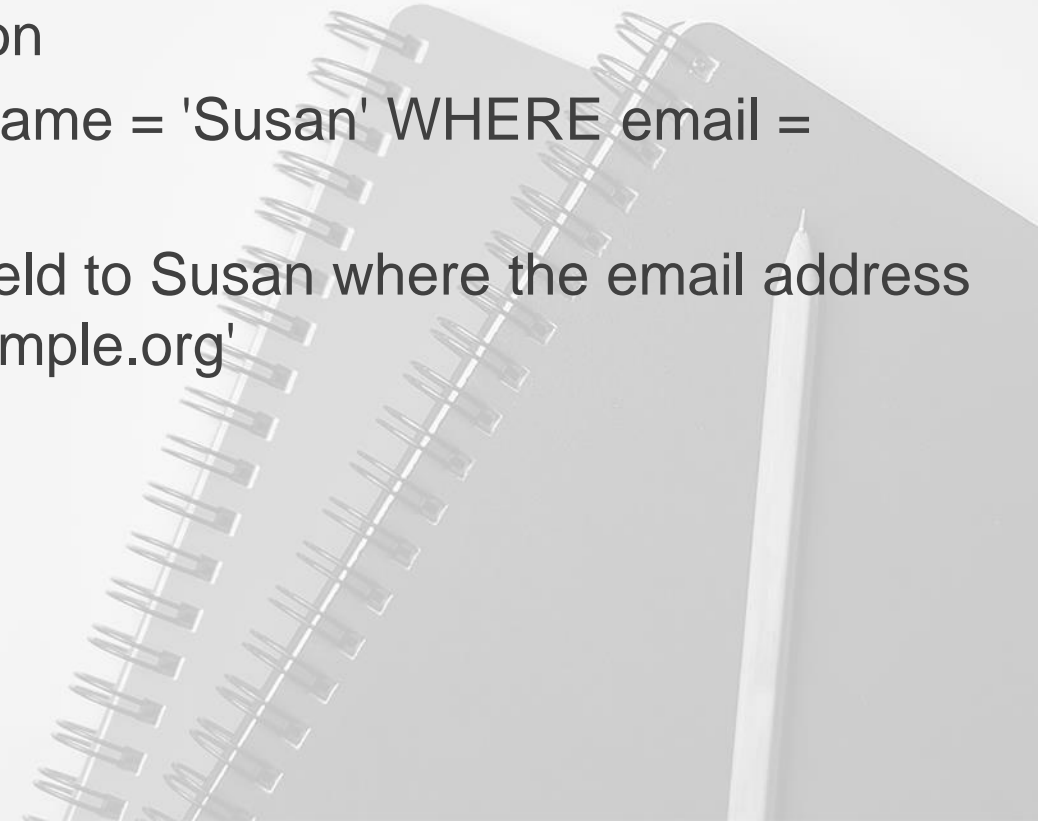
John Smith was born on 1991-02-25 00:00:00

PDO

- Like any string, you can add variables to a query:
- Hint: This is useful when you want to use search criteria from `$_GET` or `$_POST`!

```
$surname = 'Smith';  
  
$results = $pdo->query('SELECT * FROM person WHERE surname = "' . $surname . '"');  
  
foreach ($results as $row) {  
    echo '<p>' . $row['firstname'] . ' ' . $row['surname'] . ' was born on ' . $row['birthday'] . '</p>';  
}
```


PDO

- SQL also supports queries for adding information to a table and updating existing information
 - `UPDATE person SET firstname = 'Susan' WHERE email = 'sueevans@example.org';`
 - Will update the firstname field to Susan where the email address is equal to 'sueevans@example.org'
- 

PDO

- You can run this from PHP via PDO and like any query, you can use variables in place of any part

```
$pdo = new PDO('mysql:dbname=' . $schema . ';host=' . $server, $username, $password);  
$email = 'sueevans@example.org';  
$newName = 'Susan';  
$pdo->query('UPDATE person SET firstname = "' . $newName . '" WHERE email = "' . $email . '"');
```

- An update query will not produce a result set, if you want to see the updated records you will need to do a separate SELECT query

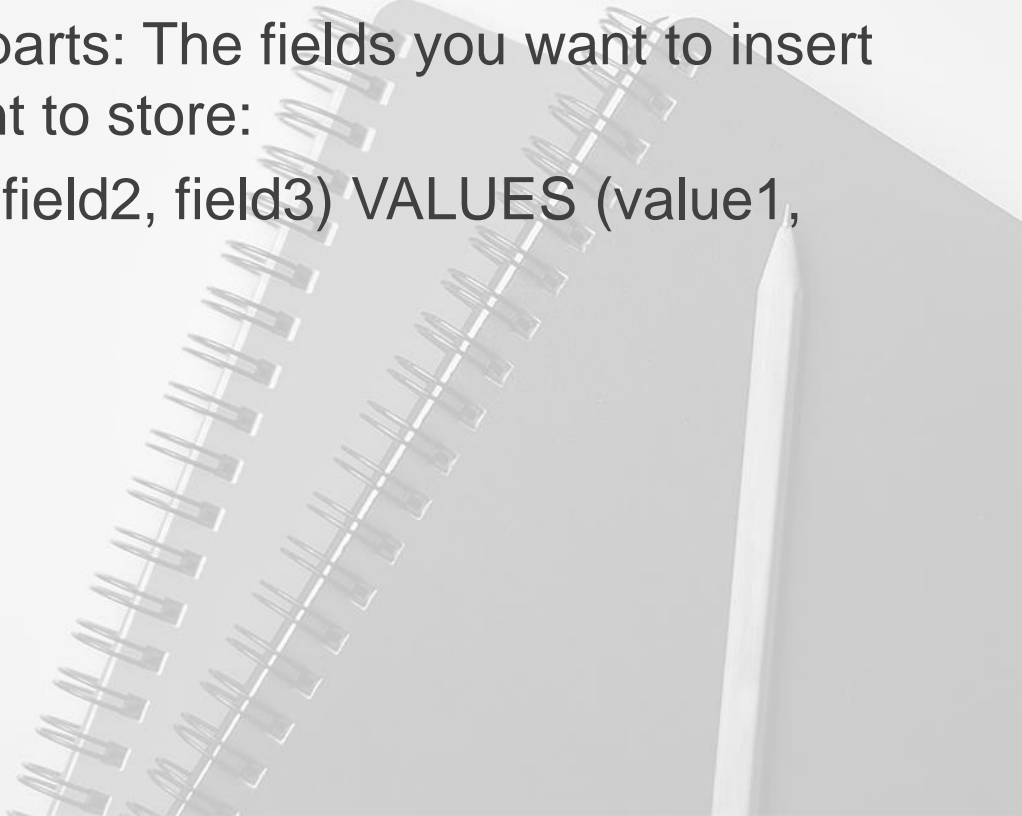
Update Queries

- You can update multiple fields at once by separating them with a comma:

```
$pdo = new PDO('mysql:dbname=' . $schema . ';host=' . $server, $username, $password);  
  
$email = 'sueevans@example.org';  
$newName = 'Susan';  
$newSurname = 'Smith';  
  
$pdo->query('UPDATE person SET firstname = "' . $newName . '",  
                                     surname = "' . $newSurname . '"  
          WHERE email = "' . $email . '"');
```

- **WARNING:** If you don't provide a WHERE clause ALL OF THE RECORDS IN THE TABLE WILL BE UPDATED

Adding Records

- To add records to a table you can use an INSERT query
 - An INSERT query has two parts: The fields you want to insert into and the values you want to store:
 - `INSERT INTO table (field1, field2, field3) VALUES (value1, value2, value3)`
- 

Adding Records

- INSERT queries can be run in the same way as UPDATE queries
- Like UPDATE queries they do not return a set of results

```
$pdo->query('INSERT INTO person (firstname, surname, birthday, email)
VALUES ("Peter", "Hill", "1987-02-01", "peter@example.org")
');
```

```
$name = 'Peter';
$surname = 'Hill';
$birthday = '1987-02-01';
$email = 'peter@example.org';
```

```
$pdo->query('INSERT INTO person (firstname, surname, birthday, email)
VALUES ("' . $name . '", "' . $surname . '", "' . $birthday . '", "' . $email . '")
');
```

Exercise 2

1. Get PHP to list the details of all the people in the database in a element
 - The list should be in the format:
 - [Firstname] [Surname] was born on [birthday] and their email address is [email]
2. Add a “Search” form with a user input and allow searching of the user by their name
 - Hint: Add the search form to the list from part (1)!
3. Add a drop down for “field” that allows selection of which database field to search e.g. “Surname”, “First name”, “Email”
4. Add a registration form that allows the user to add a record to the database table using the website. You will need to use an INSERT query
 - Hint: Use 3 select boxes to handle the date

Exercise 2

5. Add a form that allows editing of an existing user
 - Hint: You can set `value=""` on text boxes to populate them when the page loads, see Topic 6.
 - Hint: You will need to specify which user is edited. You can use `edit.php?person=john@example.org`
6. Amend the list from (4) so that each user is linked and clicking on the user brings up the edit form and allows changing their details
7. Does the edit still work if the email address is changed? If not fix it so it does!
 - Hint: You can use `<input type="hidden" value="somevalue">` to create a hidden input that sends a value to the server but can't be edited by the user