UNIVERSITY of HOUSTON

DEPARTMENT OF COMPUTER SCIENCE

COSC 4351 Fall 2023

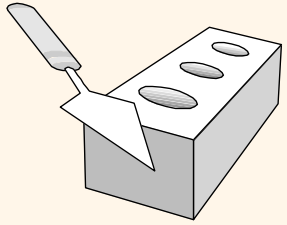
Software Engineering

M & W 4 to 5:30 PM

Prof. Victoria Hilford

PLEASE TURN your webcam ON

NO CHATTING during LECTURE



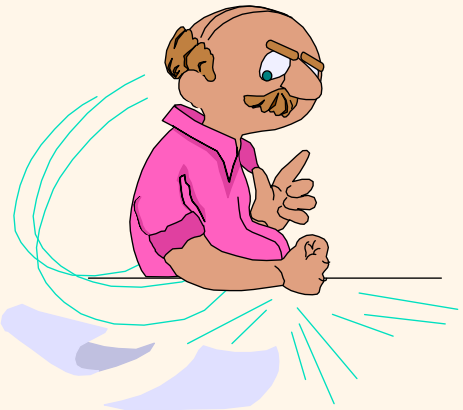
COSC 4351

4 to 5:30

PLEASE

LOG IN

CANVAS



Youyi [A-L]

Kevin [M-Z]

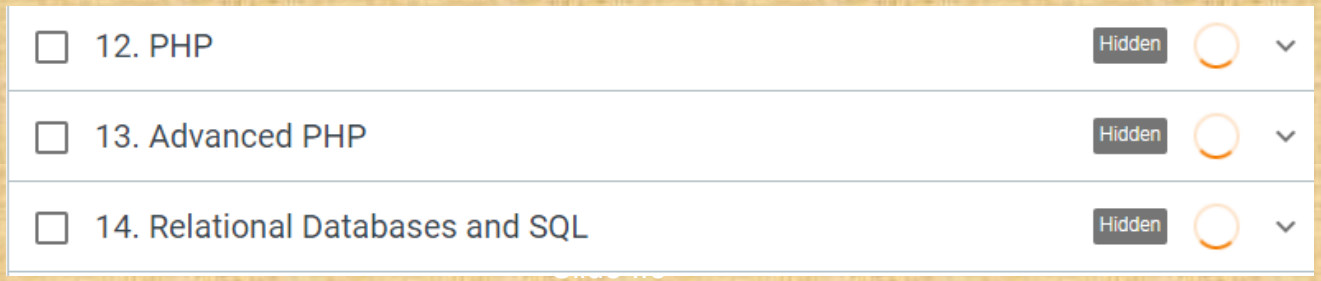
Please close all other windows.

11.08.2023 (W 4 to 5:30) (23)		WEB TECHNOLOGIES	Start ZyBook: Sections 12-14		
11.13.2023 (M 4 to 5:30) (24)		Lecture 8: Implementation			
11.15.2023 (W 4 to 5:30) (25)		Lecture 9: Testing Tutorial 6 TDD			
11.20.2023 (M 4 to 5:30) (26)		EXAM 4 REVIEW (CANVAS)	Download ZyBook: Sections 12-14		
11.27.2023 (M 4 to 5:30) Optional (27)					Q & A Set 4 topics.
11.29.2023 (M 4 to 5:30) (28) LAST CLASS					EXAM 4 (CANVAS)

Class 23

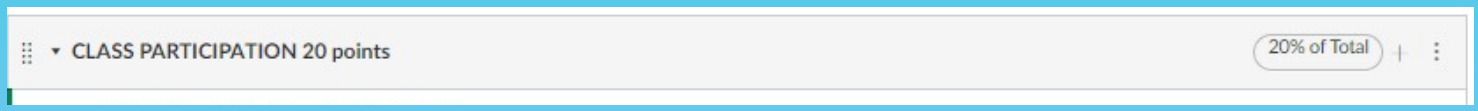
COSC 4351

Software engineering

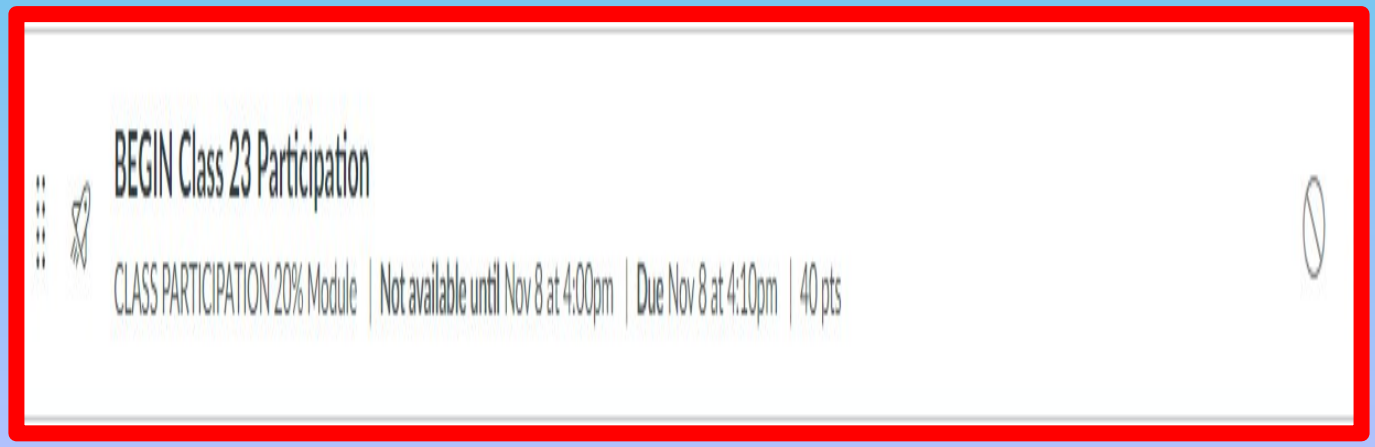


From 4:00 to 4:10 PM – 10 minutes.

11.08.2023 (W 4 to 5:30) (23)		WEB TECHNOLOGIES	Start ZyBook: Sections 12-14		

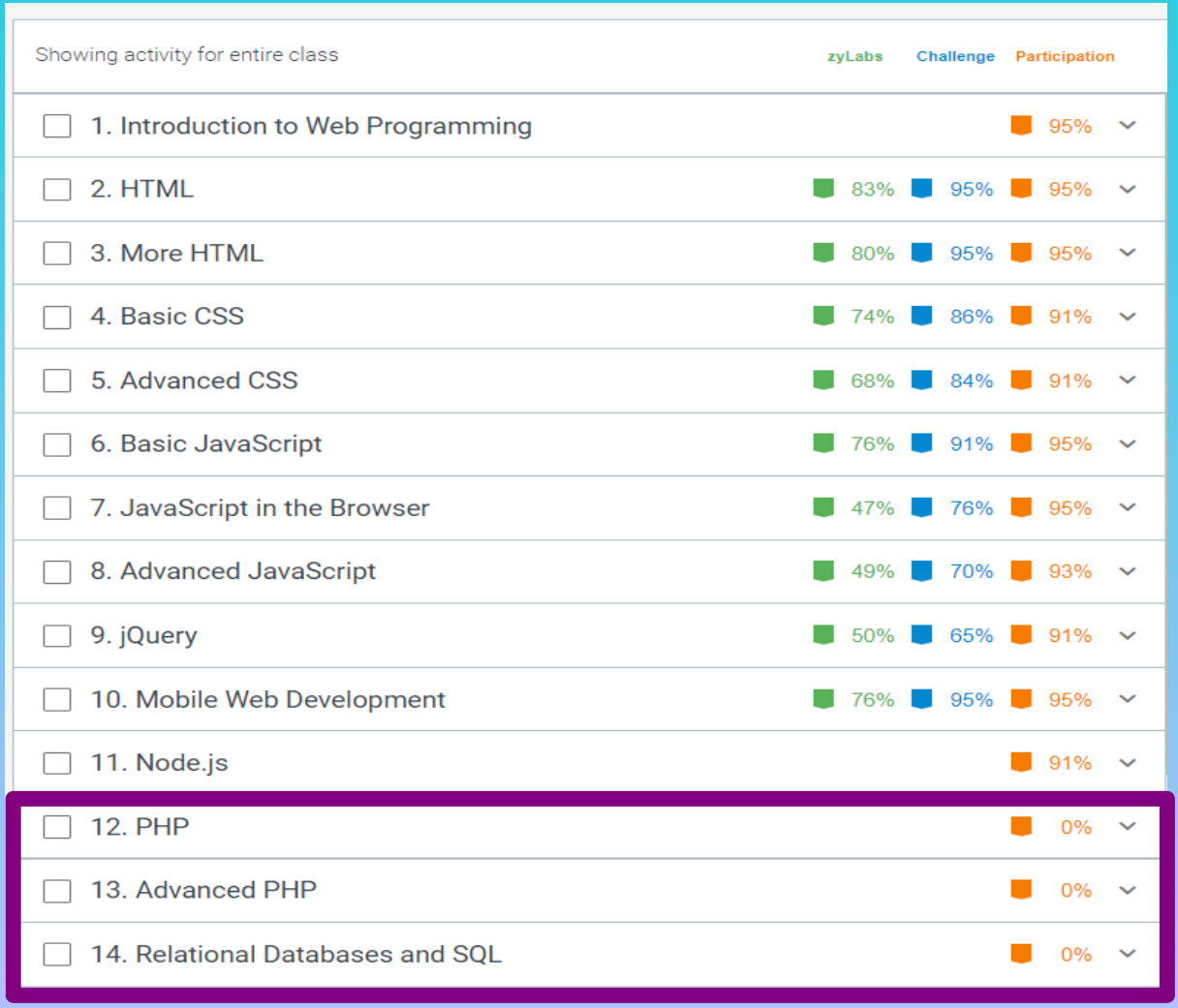


PASSWORD: SET 4



From 4:10 to 5:00 – 50 minutes.

ZyBook – Web Programming



Showing activity for entire class	zyLabs	Challenge	Participation
1. Introduction to Web Programming			95%
2. HTML	83%	95%	95%
3. More HTML	80%	95%	95%
4. Basic CSS	74%	86%	91%
5. Advanced CSS	68%	84%	91%
6. Basic JavaScript	76%	91%	95%
7. JavaScript in the Browser	47%	76%	95%
8. Advanced JavaScript	49%	70%	93%
9. jQuery	50%	65%	91%
10. Mobile Web Development	76%	95%	95%
11. Node.js			91%
12. PHP			0%
13. Advanced PHP			0%
14. Relational Databases and SQL			0%

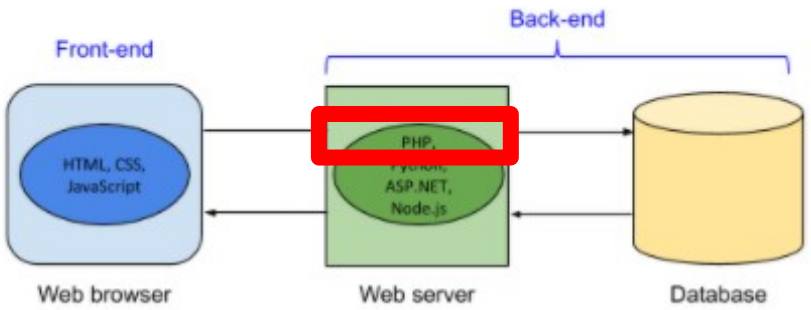
12.1 Full-stack development (PHP)

Present Note

Overview of front-end and back-end development

Most websites and web applications require the development of client-side technologies that interact with server-side technologies. ***Client-side*** (or ***front-end***) refers to those technologies that run in the web browser like HTML, CSS, and JavaScript. ***Server-side*** (or ***back-end***) refers to those technologies that run on the web server like PHP, Python, Node.js, etc. and databases. Ex: Amazon uses server-side technologies to store information on millions of products and a client-side search interface that interacts with the web server so customers can find and purchase products.

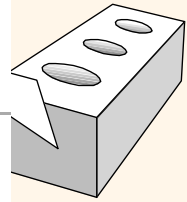
Figure 12.1.1: Front-end and back-end technologies.



Feedback?

A ***front-end developer*** is a developer that is proficient in client-side technologies. A ***back-end developer*** is a developer that is proficient in server-side technologies. Many developers strive to be proficient in both front-end and back-end technologies and how the two sides work together. A ***full-stack developer*** is a developer who has expertise in all aspects of a website or web application's development, including client technologies, server technologies, data modeling, and user interfaces. The "stack" in "full-stack" refers to the various layers that compose websites and web applications. Technology stacks have increased in complexity over the years, so even "full-stack" developers typically specialize in a few areas of the technology stack.

PHP



From Wikipedia, the free encyclopedia

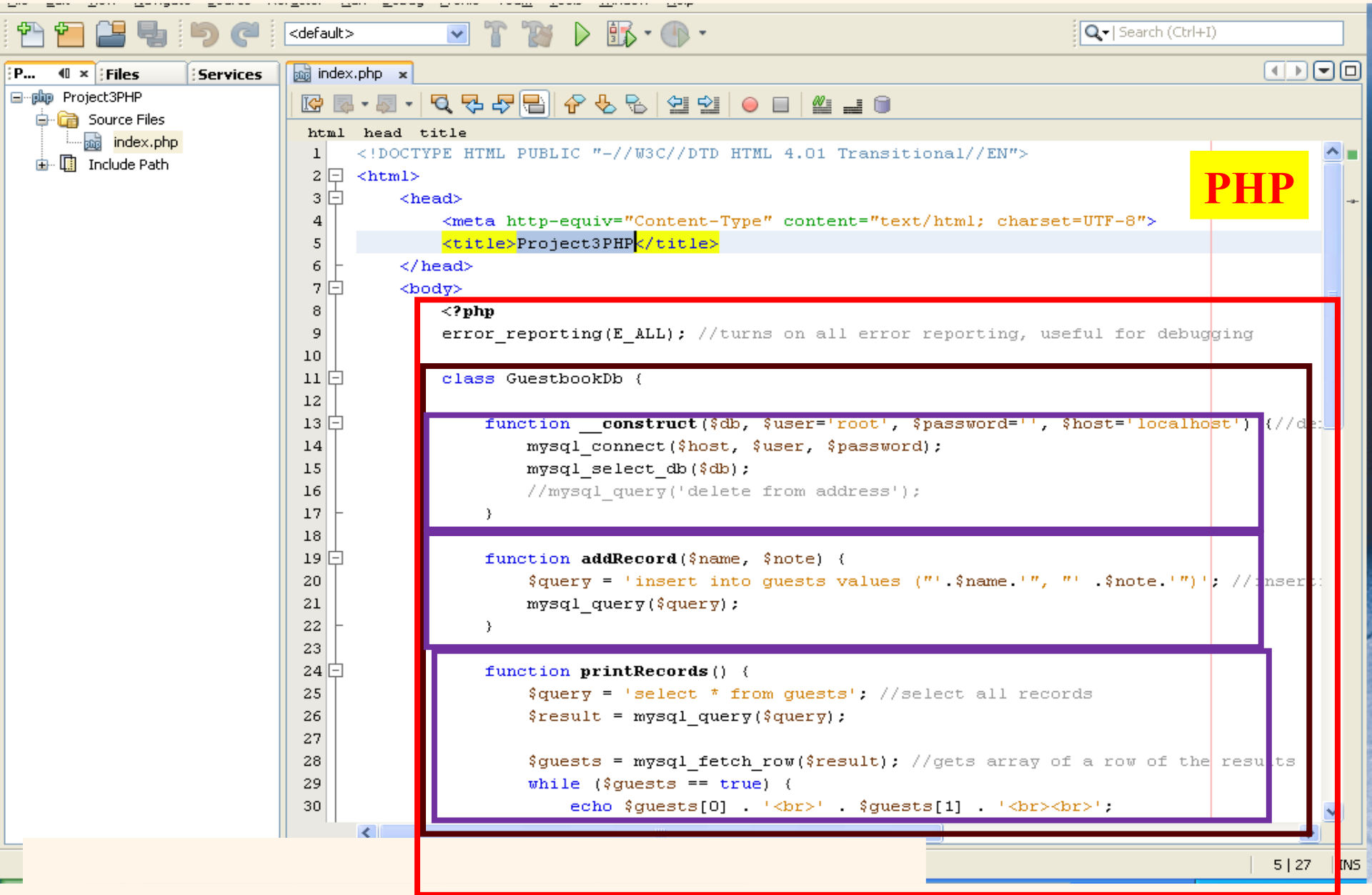
This article is about the scripting language. For other uses, see PHP (disambiguation).

PHP is a general-purpose server-side scripting language originally designed for Web development to produce dynamic Web pages. It is one of the first developed server-side scripting languages to be embedded into an HTML source document rather than calling an external file to process data. The code is interpreted by a Web server with a PHP processor module which generates the resulting Web page. It also has evolved to include a command-line interface capability and can be used in standalone graphical applications.[2] PHP can be deployed on most Web servers and also as a standalone shell on almost every operating system and platform free of charge.[3] A competitor to Microsoft's Active Server Pages (ASP) server-side script engine[4] and similar languages, PHP is installed on more than 20 million Web sites and 1 million Web servers.[5] Software that uses PHP includes MediaWiki, Joomla, Wordpress, Concrete5, MyBB, and Drupal.

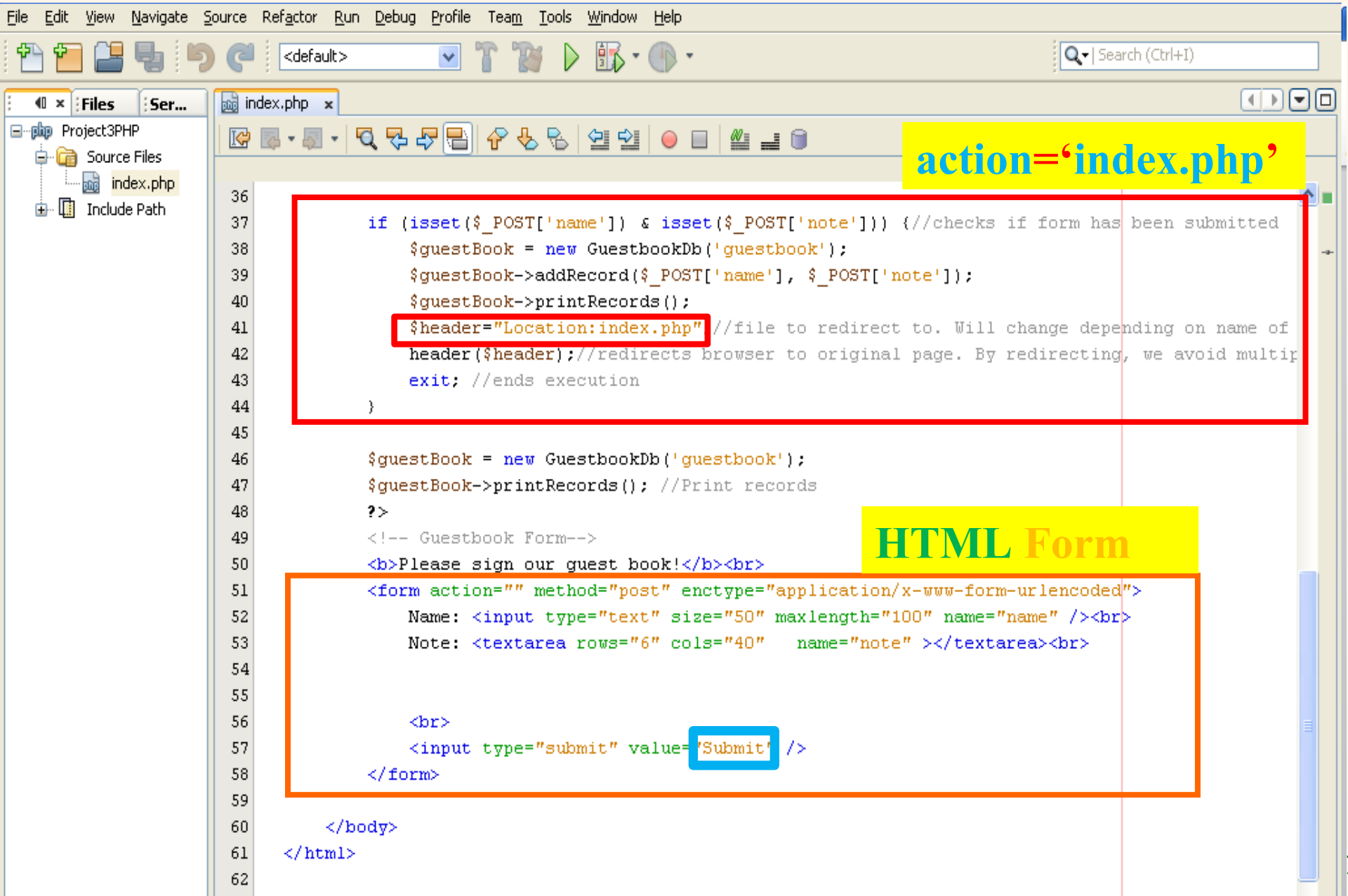
PHP was originally created by Rasmus Lerdorf in 1995. The main implementation of PHP is now produced by The PHP Group and serves as the formal reference to the PHP language.[6] PHP is free software released under the PHP License, which is incompatible with the GNU General Public License (GPL) due to restrictions on the usage of the term *PHP*.[7]

While PHP originally stood for *Personal Home Page*, it is now said to stand for *PHP: Hypertext Preprocessor*, a recursive acronym.[8]

PHP to mySQL Server DBMS : A Complete Example



PHP to mySQL DBMS Server : A Complete Example



Search content | Search | View content explorer | Configure zyBook

1. Set 1: 1 Introduction to Java — New Content
2. Set 1: 2 Basic Objects
3. Set 1: 3 Basic Methods + Classes
4. Set 1: 4 Data Types
5. Set 1: 5 Branches
6. Set 1: 6 Loops
7. PAs 1 Review (zyBook) — Hidden
8. EXAM 1 (BB) — Hidden
9. PAs 1 (zyBook) — Hidden
10. Set 2: 10 Arrays
11. Set 2: 11 Methods Continued
12. Set 2: 12 Classes Continued
13. Set 2: 13 Inheritance
14. PAs 2 Review (zyBook) — Hidden
15. EXAM 2 (BB) — Hidden
16. PAs 2 (zyBook) — Hidden
17. Set 3: 17 Abstract Class and Interfaces
18. Set 3: 18 Generics — New Content
19. Set 3: 19 Collections — New Content
20. Set 3: 20 GUI
21. Set 3: 21 JavaFX
22. PAs 3 Review (zyBook) — Hidden
23. EXAM 3 (BlackBoard) — Hidden
24. PAs 3 (zyBook) — Hidden
25. Set 4: 25 Input / Output
26. Set 4: 26 Exceptions — New Content
27. Set 4: 27 Recursion
28. Set 4: 28 Memory Management
29. Additional Material
30. PAs 4 Review (zyBook) — Hidden
31. EXAM 4 (BlackBoard) — Hidden
32. PAs 4 (zyBook) — Hidden

COSC 1320: Object Oriented Paradigm, JAVA and C++
Expires May 6th, 2022

Get started
Take a guided look into the interactive content and platform features.
Take the tour

Use a zyBook in your class
Start the adoption process to use this zyBook in your class.
Adopt this zyBook

Schedule a demo
Attend a live professional demonstration of key zyBook features.
Schedule now

JAVA & C++

Welcome | My class | Reporting | Assignments | Tools

Search content | Search | View content explorer | Configure zyBook

Showing activity for entire class — zyLabs | Challenge | Participation

1. SET 1	Empty			
2. Set 1: Basic Hardware, Software Concepts and ...		88%	93%	94%
3. Set 1: Variables and Expressions		70%	84%	86%
4. Set 1: Types - String, List, Dictionary		60%	74%	79%
5. SET 1 Practice (ZyBook)	Hidden	45%		
6. PAs 1 Review (ZyBook)	Hidden	70%		
7. PAs 1 (ZyBook)	Hidden	77%		
8. SET 2	Empty			
9. Set 2: Branching		67%	88%	91%
10. Set 2: Loops		53%	65%	80%
11. SET 2 Practice (ZyBook)	Hidden	45%		
12. PAs 2 Review (ZyBook)	Hidden	66%		
13. PAs 2 (ZyBook)	Hidden	64%		
14. SET 3	Empty			
15. Set 3: Functions		50%	73%	78%
16. Set 3: Strings		51%	73%	75%
17. SET 3 Practice (ZyBook)	Hidden	57%		
18. PAs 3 Review (ZyBook)	Hidden	60%		
19. PAs 3 (ZyBook)	Hidden	46%		
20. SET 4	Empty			
21. Set 4: Lists and Dictionaries		6%	5%	7%
22. Set 4: Plotting				5%
23. Set 4: Files		0%		3%
24. SET 4 Practice (ZyBook)	Hidden	0%		
25. PAs 4 Review (ZyBook)	Hidden	0%		
26. PAs 4 (ZyBook)	Hidden	0%		
27. FINAL EXAM PART II	Hidden	0%		

COSC 1336: Computer Science and Programming
Spring 2022

View activity and create a report

1. Select chapters and sections in the table of contents.
2. Then select class and time options below.

Entire class

From: Select date | Select time | CDT

Until: Apr 11th, 2022 | 11:59 pm | CDT

All activity up until Apr 11th, 2022 at 11:59 pm CDT will be downloaded.

Include data on time spent in chapters, sections, and on activities

Download report

You must select at least one section from the table of contents to download a report.

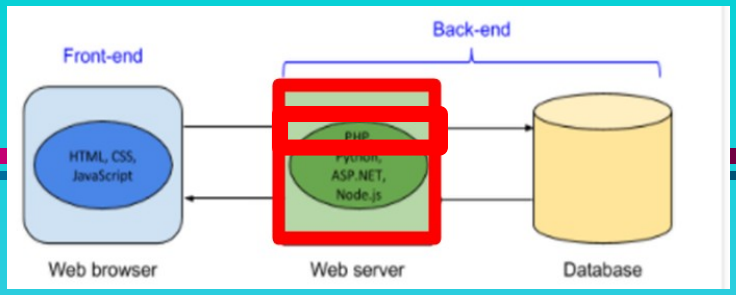
Class analytics (beta feature)

View time spent, completion, and timeline of activity for your entire class or individual students on chapters, sections, and activities. Watch a demo video.

View analytics

Python

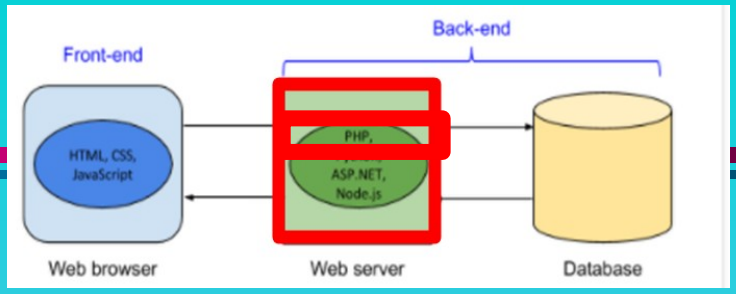
12. PHP



ZyBook – Web Programming

- 12. PHP — 0%
  - 12.1 Full-stack development (PHP) — 0%
  - 12.2 Getting started with PHP — 0%
  - 12.3 Arithmetic and comparisons (PHP) — 0%
  - 12.4 Conditionals (PHP) — 0%
  - 12.5 Loops (PHP) — 0%
  - 12.6 Arrays (PHP) — 0%
  - 12.7 Functions (PHP) — 0%
  - 12.8 Includes (PHP) — 0%
  - 12.9 Classes and objects (PHP) — 0%
  - 12.10 String, date/time, and math functions (PHP) — 0%
  - 12.11 Submitting forms (PHP) — 0%

PHP



5 Best Programming Languages for Web Development in 2021

4. PHP

I may get a lot of flaks for including PHP in this list of best programming languages for web development in 2021 but to be honest, PHP is one of the best programming languages when it comes to creating web applications.

It's a dynamic, server-side scripting language that makes it really easy to create fully functional web applications. If that's not enough, half of the internet is running on PHP's shoulder, remember WordPress, the most popular web application software is made on PHP.

PHP also has frameworks like *Laraval*, which is both powerful and allows you to swiftly create web applications using a model–view–controller architectural pattern.

Don't listen to people who say PHP is not useful, if you find it easy, go for it, and If you are want to learn PHP for web development in 2021

Your Answer?

1) PHP is used by Facebook, Wikipedia, and WordPress.

- (●) True
- ( ) False

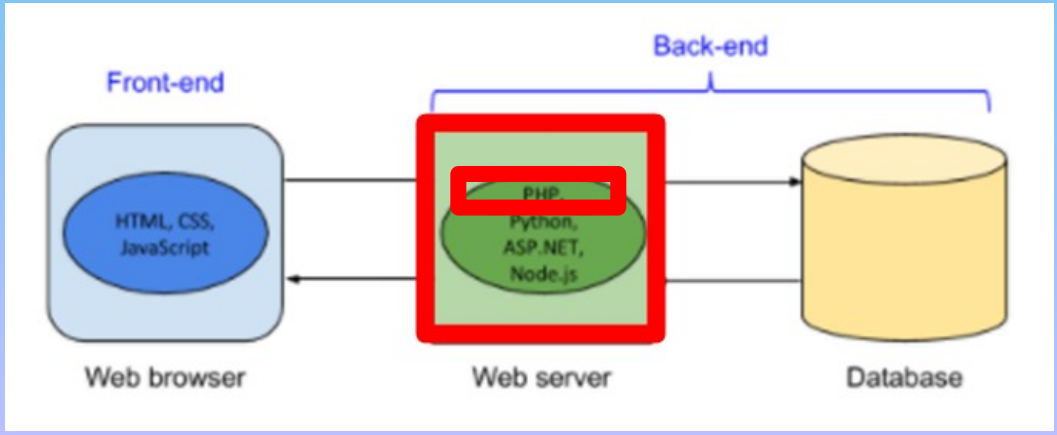
PHP is nearly as old as the Web, but PHP continues to remain a popular server-side technology, even for large websites.

Your Answer?

PHP code runs in a web browser.

- ○ True
- ◉ False

PHP code runs on a web server. JavaScript runs in a web browser.



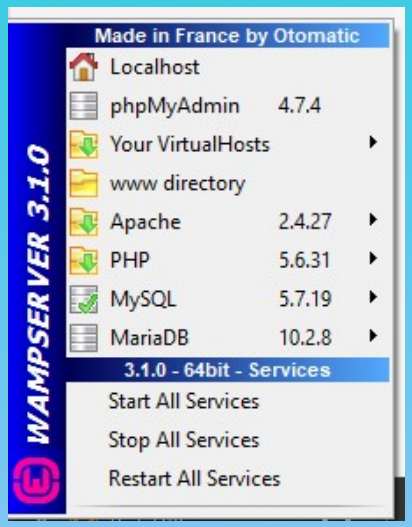
Your Answer?

Many PHP applications run on Linux with an Apache web server and MySQL as the database.

- (●) True
- ( ) False

LAMP (Linux, Apache, MySQL, PHP) is a popular bundling of technologies used to deploy many web applications.

WAMP



Your Answer?

Executing PHP code.

What file extension does a PHP script have?

- ○ .html
- ◉ .php
- ○ .png

The web server knows that the file contains PHP code because of the .php extension.

Your Answer?

Executing PHP code.

What does the code output?

```
$test = "adios";
echo "$test test";
```

- ○ $test test
- ○ $adios test
- ◉ adios test

The `$test` variable is output as "adios".

Your Answer?

Executing PHP code.

What output is produced by the PHP script?

```
<h1>This is a test</h1>
echo "This is only a test";
```

- (●) <h1>This is a test</h1> echo "This is only a test";
- ( ) <h1>This is a test</h1> This is only a test
- ( ) This is only a test

The `<?php` and `?>` tags are missing, so the PHP code is treated as HTML and output as-is.

Your Answer?

Executing PHP code.

Must PHP statements be terminated by a semicolon?

- (●) Yes
- ( ) No

Like many programming languages, PHP statements end with a semicolon.

Your Answer?

Which variable is illegally named?

- `$a1_`
- `$null`
- `$2b_`

Variables may not start with a digit after $.

Your Answer?

Which constant is illegally named?

- (●) $A1_
- ( ) _A1
- ( ) TEST_23

Constants may not begin with a $.

Your Answer?

Which code segment produces a notice error message?

- ```
  $count = 2;
  echo $COUNT;
  ```
- ```
  $count = 2;
  ECHO $count;
  ```
- Both code segments produce notice errors.

PHP variables are case-sensitive, so `$COUNT` is an uninitialized variable.

Your Answer?

Which statement is equivalent to `const EMAIL = "sue@gmail.com";`?

- `define("EMAIL") = "sue@gmail.com";`
- `define("EMAIL", "sue@gmail.com");`
- `EMAIL = "sue@gmail.com";`

One difference between **`const`** and **`define()`** is that **`define()`** can create a constant name from a variable, and **`const`** cannot.

Your Answer?

The `var_dump(variable)` function outputs the data type of a variable. What is the output of the code below?

```
$x = 1.23;
var_dump($x);
```

- ○ `int(1.23)`
- ◉ `float(1.23)`
- ○ `number(1.23)`

1.23 has a decimal place, so 1.23 is a floating point number.

Your Answer?

Which statement produces a parse error?

- `$owner = "Pam's";`
- `$owner = 'Pam's';`
- `$owner = 'Pam"s';`

A single quote in a string delimited with single quotes must be escaped with a backslash: `$s = 'Pam\'s';`

Your Answer?

```
(3 + 4) % 5 = ?
```

2

Mod % computes the remainder of the left number divided by the right number, so (3 + 4) % 5 = 7 % 5 = 2

Your Answer?

Arithmetic and comparisons.

```
$points = 10;
$points--;
$points + 2 = ?
```

11

$points-- is the same as: $points = $points - 1 = 10 - 1 = 9. Then, 9 + 2 = 11.

Your Answer?

Compound **assignment** operators.

What compound assignment operator makes `$points` become 2.5?

```
$points = 5;
$points ___ 2;
```

/=

$points /= 2 is the same as: $points = $points / 2 = 5 / 2 = 2.5.

Your Answer?

Compound **assignment** operators.

What value is $points?

```
$points = 2;
$points *= 3 + 1;
```

8

$Points = $Points * (3 + 1) = 2 * (3 + 1) = 2 * 4 = 8

Your Answer?

Compound **assignment** operators.

What value is $message?

```
$message = "Top";
$message .= "secret";
```

Topsecret

"Top" . "secret" = "Topsecret".

Your Answer?

Indicate if the expression evaluates to `true` or `false`.

1) `"bat" < "ball"`

- ○ true
- ◉ false

The "t" in "bat" has an ASCII value greater than the first "l" in "ball".

Your Answer?

Comparison operators.

Indicate if the expression evaluates to `true` or `false`.

2) `"123" == 123`

- (●) true
- ( ) false

When a string appears in an arithmetic expression, the string is first converted into a number before the expression is computed. Ex: 1 + "2" = 1+2 = 3.

"123" is a string, and 123 is an integer. Both operands must be the same data type to be identical.

Your Answer?

Comparison operators.

Indicate if the expression evaluates to `true` or `false`.

3) `"123" === 123`

- ○ true
- ◉ false

"123" is identical to "123".

Your Answer?

Comparison operators.

Indicate if the expression evaluates to `true` or `false`.

4) `"123" !== "123"`

- ○ true
- ◉ false

"123" is a string, and 123 is an integer. Both operands must be the same data type to be identical.

Your Answer?

Indicate if the expression evaluates to `true` or `false`.

1) `2 > 10 || 1 > 2`

- ○ true
- ◉ false

`false || false = false`

Your Answer?

Evaluating if-else and elseif statements.

What is $x at the end of each code segment?

```
$x = 0;
if ($x > 10) {
    $x = 1;
}
```

0

Since 0 is not > 10, the if statement is false, and the if block does not execute.

Your Answer?

What is $x at the end of each code segment?

```
if ("history" < "math") {
    $x = 1;
}
else {
    $x = 0;
}
```

1

The ASCII value for "h" is < "m", so "history" < "math" is true.

Your Answer?

The switch statement.

```
switch ($item) {
   case "apple":
   case "orange":
      $fruits++;
      break;
   case "milk":
      $drinks++;
   case "cheese":
      $dairy++;
      break;
   case "beef":
   case "chicken":
      $meat++;
      break;
   default:
      $other++;
}
```

If `$item` is "beef", what variables are incremented?

- ○ `$other`
- ◉ `$meat` only
- ○ `$meat` and `$other`

After incrementing `$meat`, the `break` statement stops executing code in the switch statement.

Your Answer?

The switch statement.

```
switch ($item) {
    case "apple":
    case "orange":
        $fruits++;
        break;
    case "milk":
        $drinks++;
    case "cheese":
        $dairy++;
        break;
    case "beef":
    case "chicken":
        $meat++;
        break;
    default:
        $other++;
}
```

If `$item` is "milk", what variables are incremented?

- ○ `$other`
- ○ `$drinks` only
- ◉ `$drinks` and `$dairy`

The statements under the "milk" and "cheese" cases are executed since the "milk" case does not end with a `break` statement.

Your Answer?

The while loop.

What are the first and last numbers output by the code segment?

```
$c = 100;
while ($c > 0) {
    echo $c;
    $c -= 10;
}
```

- ○ 100 and 0
- ○ 90 and 0
- ◉ 100 and 10

The loop terminates when $c is 0. At the start of the last iteration, $c is 10. $c is then decremented to 0. The loop condition is checked: 0 < 0 is false, and the loop terminates.

Your Answer?

The while loop.

What condition makes the loop output even numbers 2 through 20?

```
$c = 2;
while (_____) {
    echo $c;
    $c += 2;
}
```

- ○ $c >= 20
- ◉ $c <= 20
- ○ $c < 20

The last time this loop outputs a value is when $c is 20.

Your Answer?

What is the last number output by the loop?

```
$c = 0;
do {
    $c++;
    echo $c;
} while ($c < 5);
```

5

The numbers 1, 2, 3, 4, 5 are output. After 5 is output, the loop's condition is false, and the loop terminates.

Your Answer?

The do-while loop.

What conditional operator makes the `do-while` loop output 10, 20, 30, ..., 100?

```
$c = 10;
do {
    echo $c;
    $c += 10;
} while ($c ___ 100);
```

<=

The loop executes when $c is 100 and terminates after $c is incremented to 110.

Your Answer?

Which loop always executes the loop body at least once?

- ○ `while`
- ◉ `do-while`
- ○ `for`

The do-while loop body is executed once before the condition is first evaluated.

Your Answer?

The for loop.

What numbers are output by the code segment?

```
for ($x = 1; $x <= 3; $x++) {        // outer loop
    for ($y = 2; $y <= 4; $y++) {    // inner loop
        echo $y;
    }
}
```

- ○ 2, 3, 4
- ◉ 2, 3, 4, 2, 3, 4, 2, 3, 4
- ○ 2, 3, 4, 5, 6, 7, 8, 9, 10

The inner for loop outputs 2, 3, 4 three times.

Your Answer?

Output "Jose" from the $names array.

```
$names = ["Beth", "Lance", "Jose"];
echo _____;
```

$names[2]

$names is an indexed array with the third name at index 2.

Your Answer?

Add the name "Stacy" to the end of the `$names` array.

```
$names = ["Beth", "Lance", "Jose"];
$names[___] = "Stacy";
```

3

Stacy is stored at index 3 of the array. The PHP function `array_push()` also adds elements to the back of an array.

Your Answer?

Output Jose's grade from the $grades array.

```
$grades = ["Beth" => "B", "Lance" => "C", "Jose" => "A"];
echo _____;
```

$grades["Jose"]

$grades is an associative array that uses the person's name for the key and letter grade for the value.

Your Answer?

Add Tim's grade to the $grades array.

```
$grades = ["Beth" => "B", "Lance" => "C", "Jose" => "A"];
$grades[_____] = "D";
```

"Tim"

The associative array has 4 key/value pairs after Tim is added.

Your Answer?

Calling **functions**.

Choose the correct call to the function below.

```
function sayGoodbye($name) {
    echo "Goodbye, $name";
}
```

- ○ sayGoodbye();
- ◉ SAYGOODBYE("Sam");
- ○ say_goodbye("Sam");

Function names are case-insensitive, but good practice is to call a function using the same case as the function's declaration.

Your Answer?

What does the code below output?

```
sayGoodbye("Sam");

function sayGoodbye($name) {
    echo "Goodbye, $name";
}
```

- (●) Goodbye, Sam
- ( ) Goodbye, $name
- ( ) Goodbye,

`sayGoodbye()` outputs "Goodbye, " followed by the value of `$name`, which contains "Sam".

Your Answer?

The code segment outputs " How are you? ".

```
function askQuestion() {
   echo "<p>$question</p>";
}

$question = "How are you?";
askQuestion();
```

- ○ True
- ◉ False

The statement `global $question;` must exist in `askQuestion` for `askQuestion()` to access `$question`.

Your Answer?

What is output?

```
function addNumber($num) {
    global $sum;
    $sum += $num;
}

$sum = 0;
addNumber(2);
addNumber(5);
echo $sum;
```

- ○ 0
- ◉ 7

`addNumber()` adds the argument to the global variable `$sum`, so 0 + 2 = 2, and 2 + 5 = 7.

Your Answer?

The code segment outputs 6.

```
function multiplyNumbers($x, $y) {
    $answer = $x * $y;
    return $answer;
}

$z = multiplyNumbers(2, 3);
echo $answer;
```

- ○ True
- ◉ False

The local variable `$answer` is not accessible outside of `multiplyNumbers()`, so an error is produced.

Your Answer?

Refer to the Book class below.

```
class Book {

}
```

The code below instantiates a single Book object.

```
$classicBook = new Book;
```

- (●) True
- ( ) False

The new operator instantiates an object with the given class.

```
class Person {

  // Property
  public $name;

  // Method
  function sayHello() {
     echo "Hi! I'm $this->name!\n";
  }
}

$bob = new Person;
$bob->name = "Bob";
$bob->sayHello();
```

Your Answer?

Refer to the Book class below.

```
class Book {

}
```

Which line of code adds a "title" property to the Book class?

- ○ public title;
- ◉ public $title;

```
class Person {

  // Property
  public $name;

  // Method
  function sayHello() {
     echo "Hi! I'm $this->name!\n";
  }
}

$bob = new Person;
$bob->name = "Bob";
$bob->sayHello();
```

Although the property does not have a **$** at the beginning, the variable representing the property does.

Your Answer?

Refer to the Book class below.

```
class Book {

}
```

Which line of code sets the "title" property?

- (●) $classicBook->title = 'Pride and Prejudice';
- ( ) $classicBook->$title = 'Pride and Prejudice';

```
class Person {

  // Property
  public $name;

  // Method
  function sayHello() {
     echo "Hi! I'm $this->name!\n";
  }
}

$bob = new Person;
$bob->name = "Bob";
$bob->sayHello();
```

The property does not have a $ at the beginning.

Your Answer?

Refer to the Book class below.

```
class Book {

}
```

What is missing from the `read()` method?

```
function read() {
   echo "I'm reading <i>_____->title</i>.";
}
```

- (●) $this
- ( ) this

`$this` is a variable, so `$` is required at the beginning.

```
class Person {

  // Property
  public $name;

  // Method
  function sayHello() {
     echo "Hi! I'm $this->name!\n";
  }
}

$bob = new Person;
$bob->name = "Bob";
$bob->sayHello();
```

Your Answer?

Object references.

```php
class Person {

  // Property
  public $name;

  // Method
  function sayHello() {
     echo "Hi! I'm $this->name!\n";
  }
}

$bob = new Person;
$bob->name = "Bob";
$bob->sayHello();
```

$person1 and **$person2** have the same name.

```php
$person1 = new Person;
$person1->name = "Dolly";
$person2 = $person1;
$person2->name = "Angie";
```

- (●) True
- ( ) False

$person1 and **$person2** refer to the same object, so setting **$person2->name** to "Angie" also changes **$person1->name** to "Angie".

Your Answer?

Object references.

```
class Person {

  // Property
  public $name;

  // Method
  function sayHello() {
     echo "Hi! I'm $this->name!\n";
  }
}

$bob = new Person;
$bob->name = "Bob";
$bob->sayHello();
```

The code below outputs "Angie".

```
$person1 = new Person;
$person1->name = "Dolly";
$person2 = new Person;
$person2->name = "Angie";
echo $person1->name;
```

- ○ True
- ◉ False

$person1 and **$person2** refer to different objects, so **$person1->name** remains "Dolly".

Your Answer?

Constructors.

Refer to the Book class below.

```
class Book {
   public $title;
   public $author;
   function read() {
      echo "I'm reading <i>$this->title</i> by $this->author.\n";
   }
}
```

What code calls the constructor below?

```
function __construct($title, $author) {
   $this->title = $title;
   $this->author = $author;
}
```

- ○ $book = new Book;
- ◉ $book = new Book('1984', 'Orwell');

The constructor initializes the title and author.

Figure 12.8.1: A constructor for Person that initializes the name.

```
class Person {
  public $name;

  // Constructor
  function __construct($name = "John Doe") {
     $this->name = $name;
  }

  function sayHello() {
     echo "Hi! I'm $this->name!\n";
  }
}

// Call the constructor with no argument
$john = new Person;
$john>sayHello();

// Call the constructor with an argument
$sue = new Person("Sue");
$sue->sayHello();
```

```
Hi! I'm John Doe!
Hi! I'm Sue!
```

Your Answer?

Destructors.

Refer to the Book class below.

```
class Book {
   public $title;
   public $author;
   function read() {
      echo "I'm reading <i>$this->title</i> by $this->author.\n";
   }
}
```

An object's destructor is called when no other references to the object exist.

- (●) True
- ( ) False

The destructor is called automatically by the PHP engine.

Your Answer?

Figure 12.8.2: A destructor for Person that says goodbye.

```
class Person {
  public $name;

  // Constructor
  function __construct($name = "John Doe") {
     $this->name = $name;
  }

  // Destructor
  function __destruct() {
     echo "Say goodbye to $this->name!\n";
  }

  function sayHello() {
     echo "Hi! I'm $this->name!\n";
  }
}

$john = new Person("John");
$sue = new Person("Sue");
$john->sayHello();
$sue->sayHello();

// Destructors called when script terminates
```

```
Hi! I'm John!
Hi! I'm Sue!
Say goodbye to Sue!
Say goodbye to John!
```

Destructors.

Refer to the Book class below.

```
class Book {
   public $title;
   public $author;
   function read() {
      echo "I'm reading <i>$this->title</i> by $this->author.\n";
   }
}
```

What goes in the blank to declare the Book's destructor?

```
function _____() {
  echo "Closing the book.\n";
}
```

- ○ destructor
- ◉ __destruct

The destructor is always called **__destruct**.

Figure 12.8.2: A destructor for Person that says goodbye.

```
class Person {
  public $name;

  // Constructor
  function __construct($name = "John Doe") {
     $this->name = $name;
  }

  // Destructor
  function __destruct() {
     echo "Say goodbye to $this->name!\n";
  }

  function sayHello() {
     echo "Hi! I'm $this->name!\n";
  }
}

$john = new Person("John");
$sue = new Person("Sue");
$john->sayHello();
$sue->sayHello();

// Destructors called when script terminates
```

```
Hi! I'm John!
Hi! I'm Sue!
Say goodbye to Sue!
Say goodbye to John!
```

Your Answer?

Which statement creates a class called **Novel** that inherits from **Book**?

- ○ `class Novel inherits Book`
- ◉ `class Novel extends Book`
- ○ `class Novel->Book`

Novel is the child class, and **Book** is the parent class.

Your Answer?

If **Novel** inherits from **Book**, and both classes define a method called **read()**, which class' method is called below?

```
$novel = new Novel("The Great Gatsby", "Fitzgerald");
$novel->read();
```

- ○ Book's
- ◉ Novel's
- ○ Neither

The Novel's method overrides Book's method.

Your Answer?

Can a parent class call a child class' method?

- ○ Yes
- ◉ No

The parent can only access the parent's methods.

Your Answer?

Data Validation.

Client-side data validation makes server-side data validation unnecessary.

- ○ True
- ◉ False

Server-side data validation is always necessary because client-side data validation can be circumvented.

Your Answer?

Writing to files.

Complete the code that creates a file called results.txt or empties the contents of results.txt.

```
$results = fopen(_____);
```

"results.txt", "w"

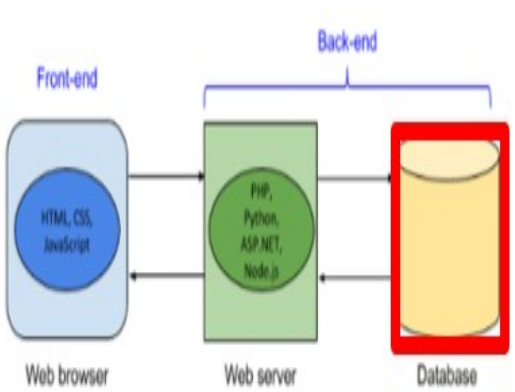
The filename is "results.txt", and write mode is "w".

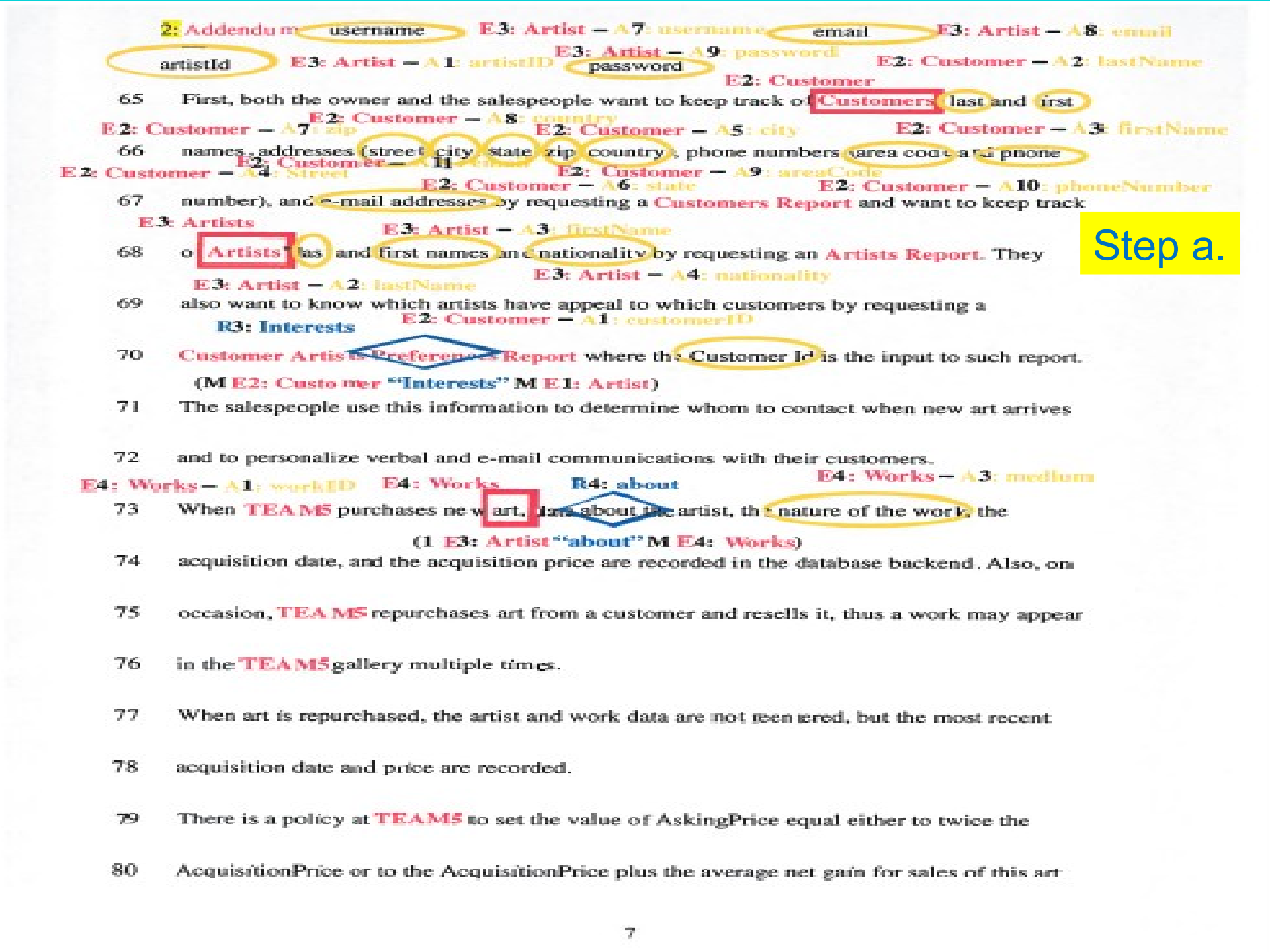
Your Answer?

14. Relational DataBases and SQL

ZyBook – Web Programming

- [ ] 14. Relational Databases and SQL 0%
  - [ ] 14.1 Relational databases 0%
  - [ ] 14.2 Structured Query Language (SQL) 0%
  - [ ] 14.3 Creating, altering, and deleting tables 0%
  - [ ] 14.4 Inserting rows 0%
  - [ ] 14.5 Selecting rows 0%
  - [ ] 14.6 SQL functions 0%
  - [ ] 14.7 Joining tables 0%
  - [ ] 14.8 Updating and deleting rows 0%





65 First, both the owner and the salespeople want to keep track of Customers last and first

66 names, addresses (street, city, state, zip, country), phone numbers (area code and phone

67 number), and e-mail addresses by requesting a Customers Report and want to keep track

68 of Artists' last and first names and nationality by requesting an Artists Report. They

69 also want to know which artists have appeal to which customers by requesting a

70 Customer Artists Preferences Report where the CustomerId is the input to such report.

(M E2: Customer "Interests" M E1: Artist)

71 The salespeople use this information to determine whom to contact when new art arrives

72 and to personalize verbal and e-mail communications with their customers.

73 When TEAM5 purchases new art, data about the artist, the nature of the work, the

(1 E3: Artist "about" M E4: Works)

74 acquisition date, and the acquisition price are recorded in the database backend. Also, on

75 occasion, TEAM5 repurchases art from a customer and resells it, thus a work may appear

76 in the TEAM5 gallery multiple times.

77 When art is repurchased, the artist and work data are not reentered, but the most recent

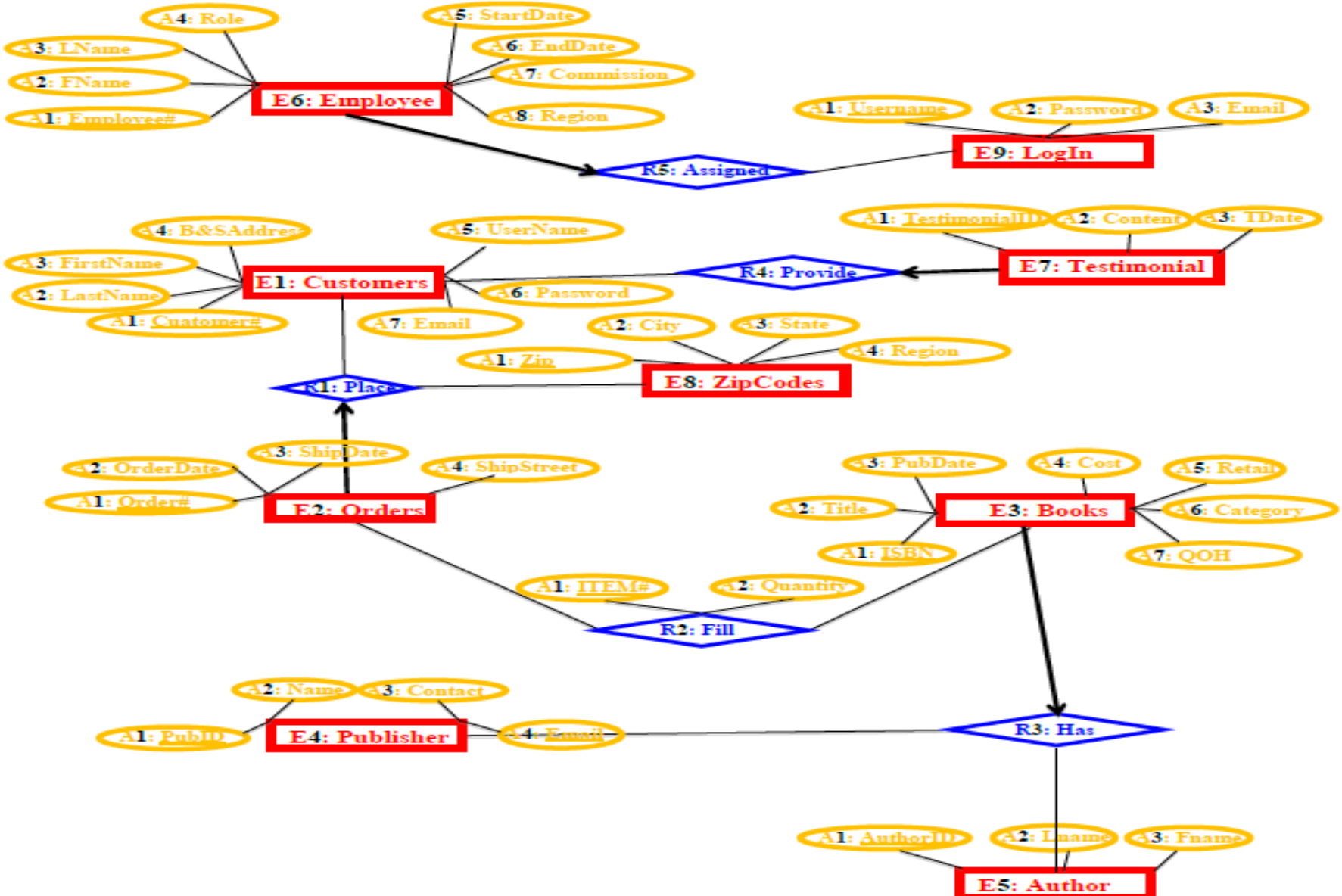
78 acquisition date and price are recorded.

79 There is a policy at TEAM5 to set the value of AskingPrice equal either to twice the

80 AcquisitionPrice or to the AcquisitionPrice plus the average net gain for sales of this art

ERD Model (WHAT data)

Step b.



a. Using the ERD model created, create a list of each Relation (Table)

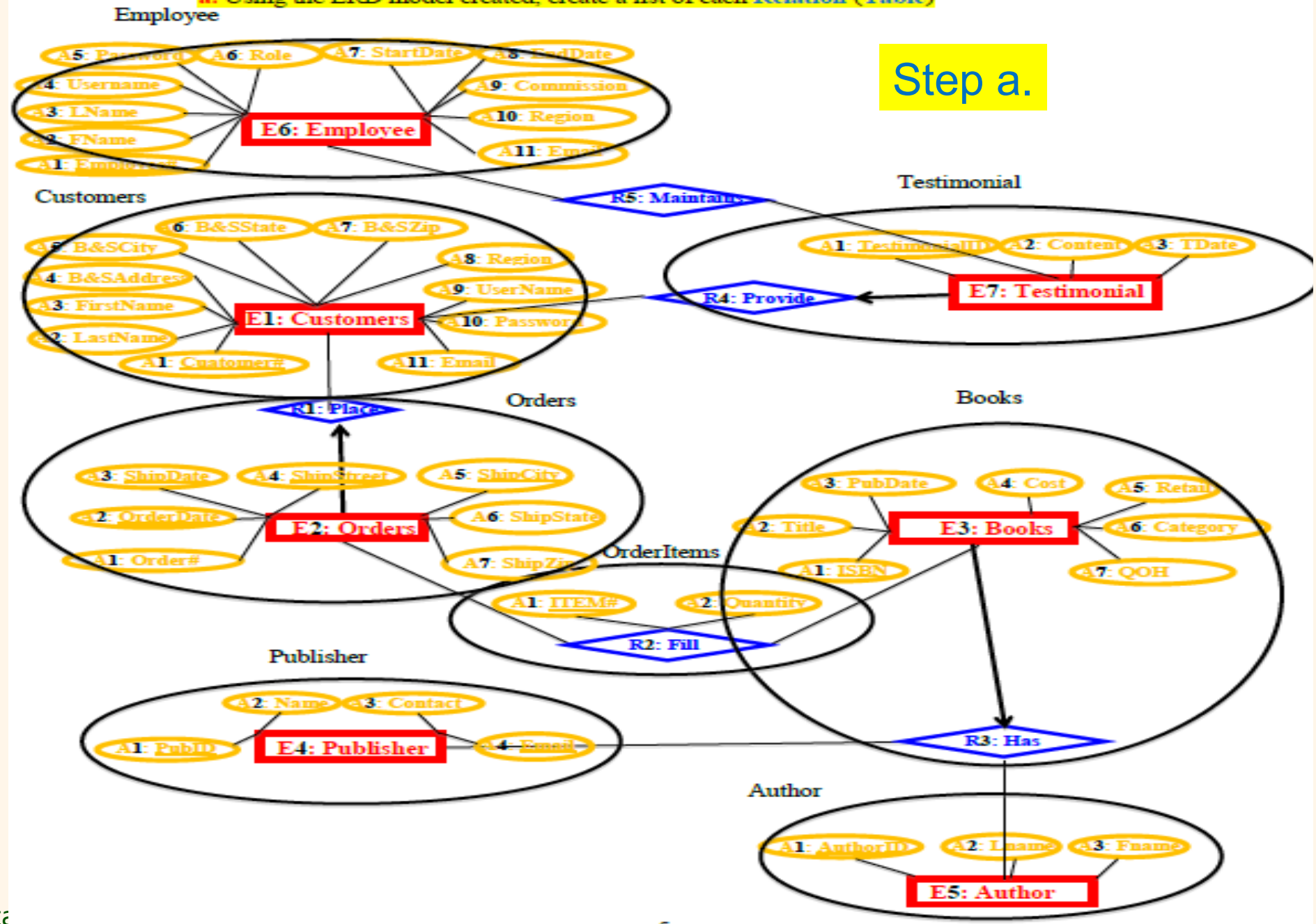


TABLE format

Step b.

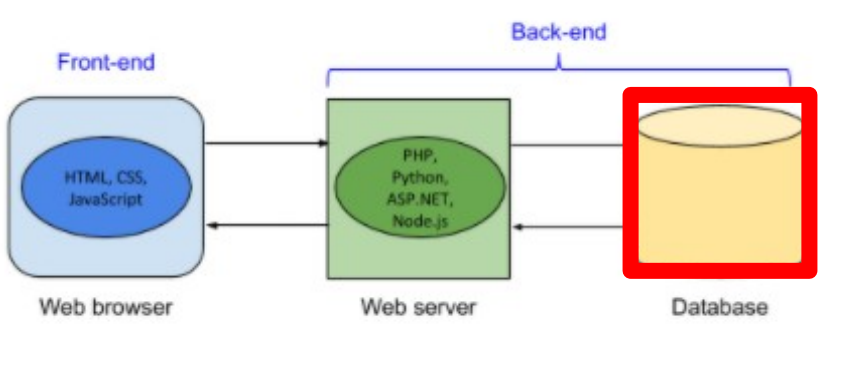
```
TABLE Customer (
                CustomerID          CHAR(20),
                Cname               CHAR(20),
                Address             CHAR(20),
                State               CHAR(20),
                City                CHAR(20),
                PostalCode          CHAR(15),
                Email               CHAR(20),
                UserName            CHAR(20),
                Password            CHAR(20),
                PRIMARY KEY         (CustomerID))

TABLE ProductLine (LineID           CHAR(30),
                   Lname            CHAR(30),
                   PRIMARY KEY(LineID))

TABLE Order (
                OrderID             CHAR (20),
                orderDate           DATE,
                  CustomerID                CHAR(20),
                PRIMARY KEY     (OrderID),
                FOREIGN KEY     (CustomerID) REFRENCES Customer)

TABLE Product (
                ProductID           CHAR(20),
                Pname               CHAR(20),
                Finish              CHAR(20),
                StandartPrice.      FLOAT,
                Photo               IMAGE,
                LineID              CHAR(20),
                PRIMARY KEY         (ProductID),
                FOREIGN KEY (LineID) REFRENCES ProductLine)

TABLE OrderLine (
                Quantity            INTEGER,
                SalePrice.          FLOAT,
                ProductID                   CHAR(20),
                OrderID             CHAR(20),
                PRIMARY KEY                 (ProductID, OrderID),
                FOREIGN KEY (ProductID) REFERENCES Product,
                FOREIGN KEY (ProductLineID) REFRENCES Order)
```

Relational databases.

A/an _____ executes SQL statements.

- (●) RDBMS
- ( ) table

An RDBMS manages all data for a relational database.

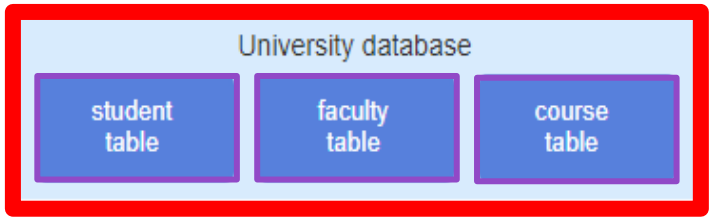


Your Answer?

Relational databases.

PARTICIPATION ACTIVITY | 14.1.1: Students table in a University database.

Start ☐ 2x speed



student table

stu_id	last_name	first_name	gpa
123	Miller	Susan	3.1
456	Lopez	Juan	2.8
789	Rida	Adib	3.7
555	Nguyen	Tye	3.0

← column names

} rows

Captions ^

1. A relational database is a collection of tables. A university database has a table for students, faculty, and courses.
2. The student table has 4 columns for storing each student's ID, last name, first name, and GPA.
3. Each row stores values for one student. The table has 4 rows that represent 4 students.

Relational databases.

*Database designers often use an **entity relationship diagram** (**ERD**) to visualize the entities and relationships that need to be mapped into a relational database. Below is an example of an ERD for the M:N relationship between student and class.*

PARTICIPATION ACTIVITY | 14.1.6: Creating a linking table.

Start | 2x speed

student M:N class

student

stu_id	last_name	first_name	gpa
123	Miller	Susan	3.1
456	Lopez	Juan	2.8
789	Rida	Adib	3.7
555	Nguyen	Tye	3.0

class

class_id	course_id	section	time	room
1001	50	1	MWF 8am	AMR 200
1002	50	2	TTh 9am	AMR 210
1003	51	1	TTh 10am	GBL 101
1004	52	1	MWF 1pm	SCI 200

student 1:M enroll

enroll

stu_id	class_id	grade
123	1001	A
123	1003	B
123	1004	B
789	1002	A
789	1004	A

```
TABLE OrderLine (
    Quantity        INTEGER,
    SalePrice.      FLOAT,
    ProductID       CHAR(20),
    OrderID         CHAR(20),
    PRIMARY KEY     (ProductID, OrderID),
    FOREIGN KEY (ProductID) REFERENCES Product
    FOREIGN KEY (ProductLineID) REFERENCES Order
```

Captions

1. A student and class have an M:N relationship. One table is created for student and one for class.
2. A linking table called enroll breaks the M:N relationship into two 1:M relationships.
3. The primary keys from student and class are foreign keys in enroll. The stu_id and class_Id columns together form the enroll table's primary key.
4. A grade column stores a student's grade in the enrolled class.
5. Susan Miller enrolls in 3 classes. Susan gets an A in 1001 and a B in 1003 and 1004.
6. Adib Rida enrolls in 2 classes and gets As in both classes.

14.2 Structured Query Language (SQL)

Present Note

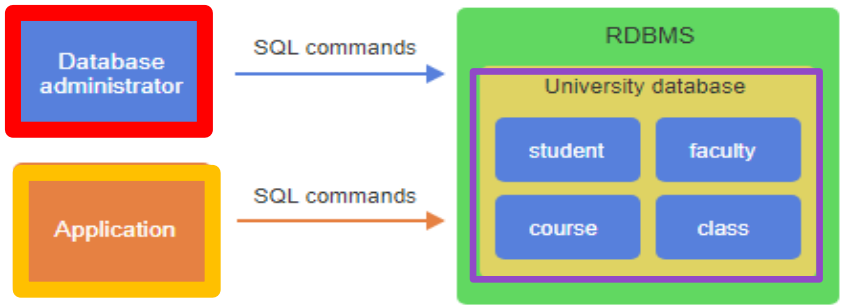
RDBMS and SQL

A ***relational database management system*** (***RDBMS***) is software that implements a relational database. Many commercial and open source RDBMSs exist including IBM DB2, MySQL, Microsoft Access, SQL Server, Oracle, PostrgreSQL, and SQLite. Database administrators and applications use SQL to interact with an RDBMS.

Structured Query Language (***SQL***) is a language used by an RDBMS to create, query, modify, and delete databases, tables, and table data. SQL is often pronounced "see-qual". Although SQL was standardized by the American National Standards Institute (ANSI) in the 1980s, each RDBMS may customize SQL in some situations. This material uses MySQL syntax. MySQL is a popular open-source RDMBS.

PARTICIPATION ACTIVITY | 14.2.1: SQL interaction with a RDBMS.

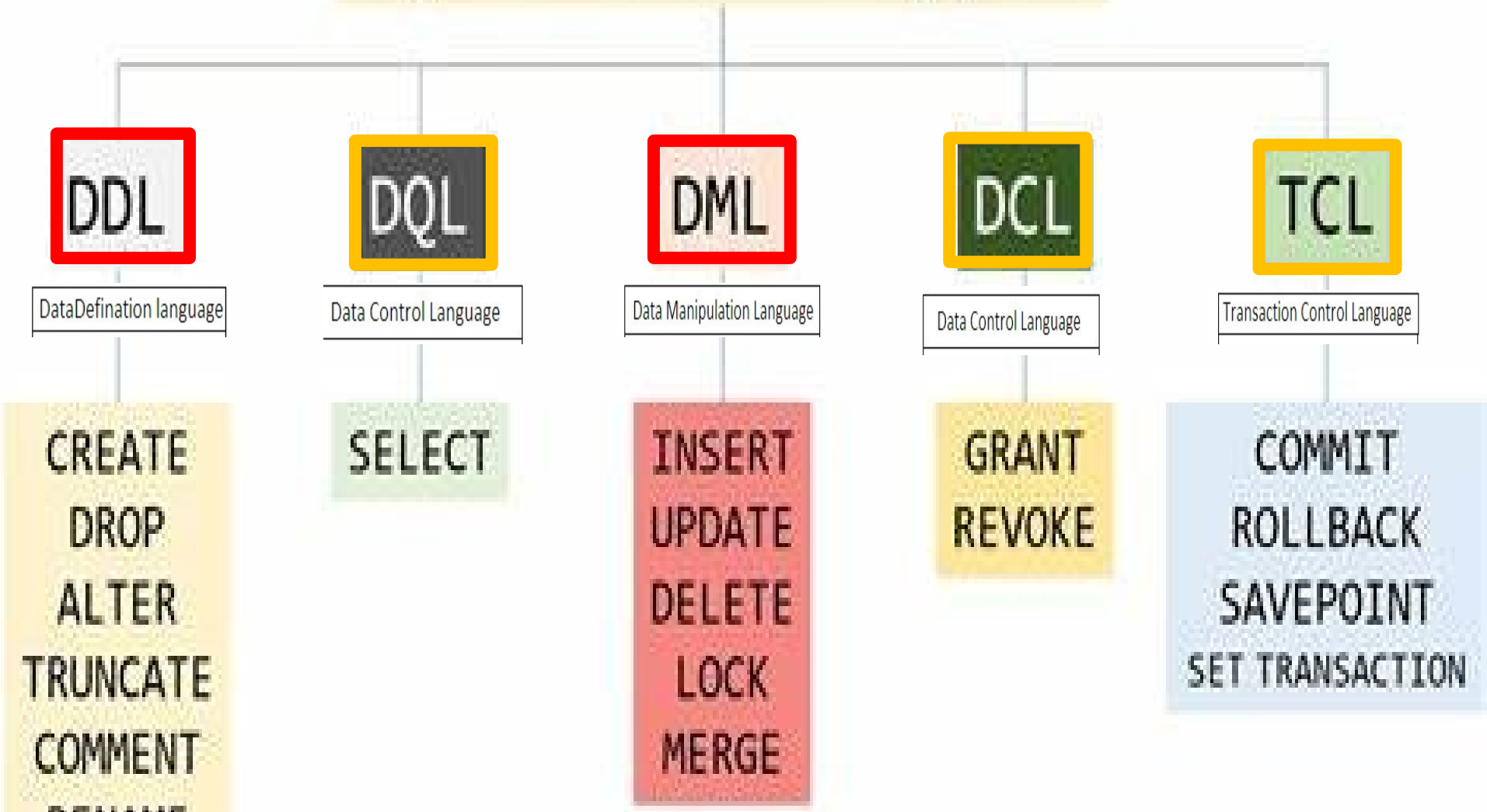
Start 2x speed



Captions

1. A database administrator uses SQL to create a database and the database's tables.
2. Applications use SQL to insert, retrieve, update, and delete data from the tables.

SQL Command Types



The word "CREATE" must be capitalized in a CREATE statement.

- ○ True
- ◉ False

SQL reserved words are not case-sensitive, so the capitalized letters in the example CREATE statement may be lowercase. A common practice is to capitalize SQL reserved words.

Your Answer?

```
TABLE OrderLine (
        Quantity        INTEGER,
        SalePrice.      FLOAT,
        ProductID       CHAR(20),
        OrderID         CHAR(20),
        PRIMARY KEY             (ProductID, OrderID),
        FOREIGN KEY (ProductID) REFERENCES Product,
        FOREIGN KEY (ProductLineID) REFRENCES Order)
```

14.3 Creating, altering, and deleting tables

CREATE TABLE statement

The ***CREATE TABLE*** statement creates a new table by specifying the table name, column names, column data types, and constraints. A ***constraint*** is a rule that applies to table data. The ***PRIMARY KEY*** constraint is used in the CREATE TABLE statement to specify the table's primary key, the column(s) that uniquely identify each row. The primary key can be indicated in two different ways, as illustrated in the figure below.

Figure 14.3.1: CREATE TABLE syntax.

Primary key listed last

```
CREATE TABLE table_name (
  column1 datatype,
  column2 datatype,
  ...
  PRIMARY KEY(column_name)
);
```

Primary key next to column

```
CREATE TABLE table_name (
  column1 datatype PRIMARY KEY,
  column2 datatype,
  ...
);
```

A table commonly has at least one column that uniquely identifies each row.

- (●) True
- ( ) False

The primary key is the column that must contain unique values.

Your Answer?

Every _____ in a database has a data type.

a. entity

b. row

c. table

d. column

Your Answer?

Relational databases.

student table

stu_id	last_name	first_name	gpa
123	Miller	Susan	3.1
456	Lopez	Juan	2.8
789	Rida	Adib	3.7
555	Nguyen	Tye	3.0

The students table may contain two students with the same stuId.

- ○ True
- ◉ False

The stuId column is the primary key, so each row in the students table must have a unique stuId value.

Your Answer?

14.4 Inserting rows

Present Note

INSERT statement

An ***INSERT*** statement inserts one or more rows into a table. The column names can be listed in parentheses after the table name. The column names can be omitted if all the values are inserted in the same order as the table's columns.

Figure 14.4.1: INSERT syntax.

Column names	No column names
`INSERT INTO table_name (column1, column2, column3, ...)` `VALUES (value1, value2, value3, ...);`	`INSERT INTO table_name` `VALUES (value1, value2, value3, ...);`

14.5 Selecting rows

Present Note

SELECT statement

The ***SELECT*** statement retrieves data from a table. The data returned from the SELECT statement is stored in a ***result table***. The column names listed after "SELECT" are included in the result table. If "*" is given as a column name, then all columns are included in the result table.

Figure 14.5.1: SELECT syntax.

Specific columns	All columns
`SELECT column1, column2, column3, ...` `FROM table_name;`	`SELECT *` `FROM table_name;`

Feedback?

PARTICIPATION ACTIVITY 14.5.1: Selecting from the student table.

Start 2x speed

student

stu_id	last_name	first_name	gpa
123	Miller	Susan	3.1
456	Lopez	Juan	2.8
987	Rida	Adib	3.7

```
SELECT *
FROM student;
```

result

stu_id	last_name	first_name	gpa
123	Miller	Susan	3.1
456	Lopez	Juan	2.8
987	Rida	Adib	3.7

```
SELECT first_name, gpa
FROM student;
```

result

first_name	gpa
Susan	3.1
Juan	2.8
Adib	3.7

CRUD

CRUD operations

The acronym ***CRUD*** names the four common operations that are performed on data: Create, Retrieve, Update, and Delete. Ex: A Create operation creates new data. The most common SQL statements can be categorized as CRUD operations. Ex: The `CREATE TABLE` statement is a Create operation because the command creates a table.

Before CRUD operations can be performed on tables and table data, a database is created with the ***CREATE DATABASE*** statement. Ex: `CREATE DATABASE testdb;` creates a database called "testdb". The ***DROP DATABASE*** statement deletes a database. Ex: `DROP DATABASE testdb;` deletes the testDB.

PARTICIPATION ACTIVITY | 14.2.5: SQL statements for tables.

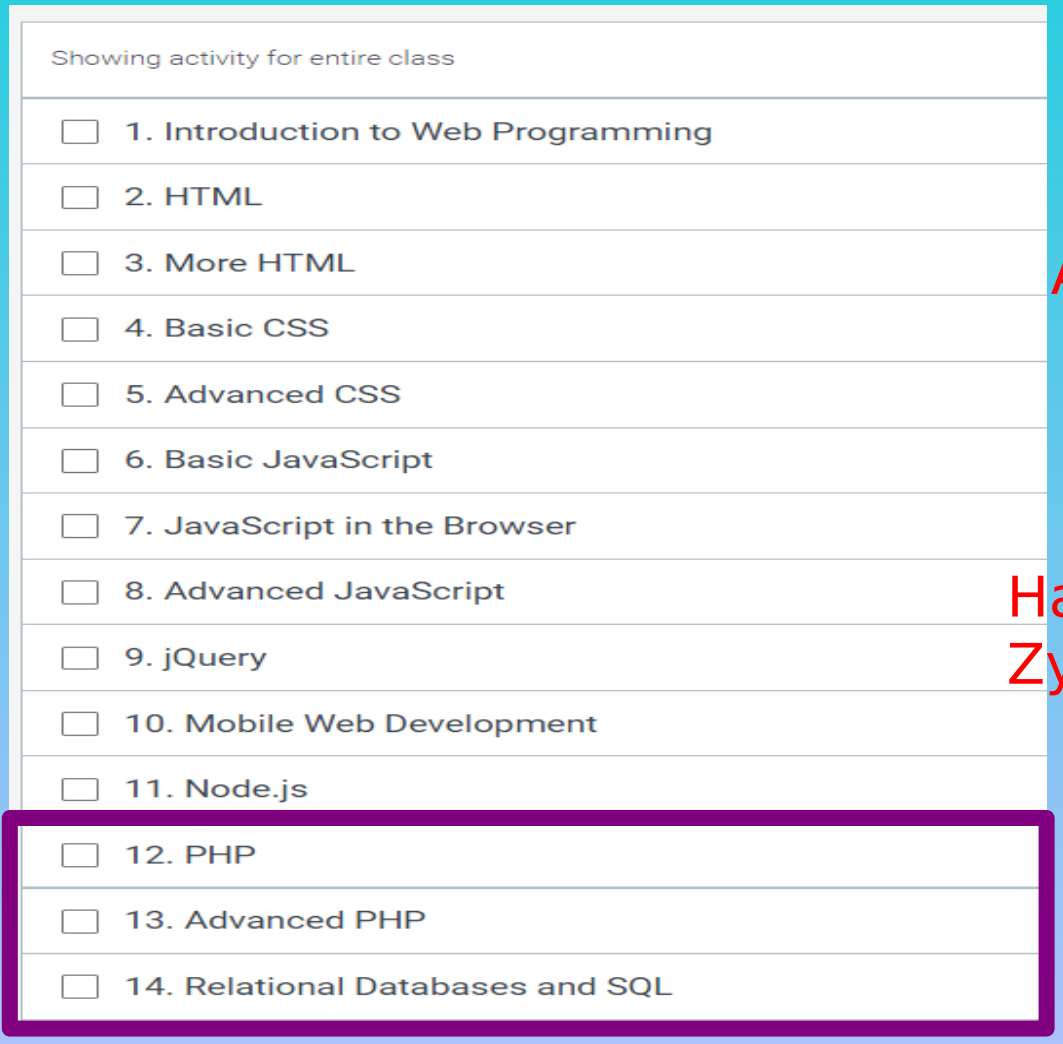
Match the SQL statement to the CRUD operation that works on tables.

DROP TABLE **CREATE TABLE** **ALTER TABLE** **SHOW TABLES**

	Create
	Retrieve
	Update
	Delete

From 4:10 to 5:00 – 50 minutes.

ZyBook – Web Programming



And so on...

Have fun!

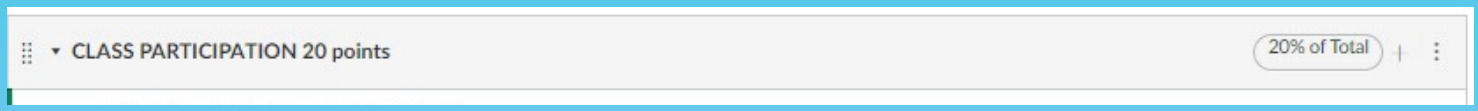
ZyBooks takes you step by step

At 5:00 PM

11.13.2023 (M 4 to 5:30) (24)		Lecture 8: Implementation			
11.15.2023 (W 4 to 5:30) (25)		Lecture 9: Testing Tutorial 6 TDD			
11.20.2023 (M 4 to 5:30) (26)		EXAM 4 REVIEW (CANVAS)	Download ZyBook: Sections 12-14		
11.27.2023 (M 4 to 5:30) Optional (27)					Q & A Set 4 topics.
11.29.2023 (M 4 to 5:30) (28) LAST CLASS					EXAM 4 (CANVAS)

From 5:05 to 5:15 PM – 10 minutes.

11.08.2023 (W 4 to 5:30) (23)		WEB TECHNOLOGIES	Start ZyBook: Sections 12-14		



PASSWORD: IN TEAMS

At 5:15 PM.

End Class 23

VH, Download Attendance Report
Rename it:
11.08.2023 Attendance Report FINAL

VH, upload Class 23 to CANVAS.