Heritage University 03-18-24

Mathematics and Computer Science Dept. due: 03-24-24

Database II

Homework 6

(MongoDB,PHP,MYSQL)

1. Consider that there are three product categories (small(1), medium(2), large(3)) with colors (orange, blue, green, etc) and with the brands (“br1”,”br2”,”br3”) prepare the following:

a. the product MongoDB database “product” (database)

b. the collections (“Tshirtsmall”,”Tshirtmedium”,”Tshirtlarge”)

**CODE FOR A & B:**

use product

db.createCollection(“Tshirtsmall”)

db.createCollection(“Tshirtmedium”)

db.createCollection(“Tshirtlarge”)

**OUTPUT:**

A screen shot of a computer code

Description automatically generated

c. insert the first data for each collection.

**CODE:**

db.Tshirtsmall.insertMany([{size: [1, 2, 3]},{colors:["orange", "blue", "green"]},{brands:["br1", "br2", "br3"]}])

db.Tshirtmedium.insertMany([{size:[1, 2, 3]},{colors:["orange", "blue", "green"]},{brands:["br1", "br2", "br3"]}])

db.Tshirtlarge.insertMany([{size: [1, 2, 3]},{colors:["orange", "blue", "green"]},{brands:["br1", "br2", "br3"]}])

**OUTPUT:**

A screenshot of a computer program

Description automatically generated

d. add more data for the small category (“Tshirtsmall”) (i.e. size:1,color:”black”,brand:”b2”,

size:1,color:”yellow”,brand:”b3”)

**CODE:**

db.Tshirtsmall.insertMany([{size:1, color:"black", brand:"b2"},{size:1, color:"yellow", brand:"b3"}, {size:2, color:"orange", brand:"b2"}, {size:3, color:"blue", brand:"b1"}])

**OUTPUT:**

A computer screen with text and numbers

Description automatically generated

e. select data of “Tshirtsmall” in JSON format

**CODE:**

db.Tshirtsmall.find()

**OUTPUT:**

A screen shot of a computer program

Description automatically generated

f. find the “Tshirtsmall” with brand:”b2”.

**CODE:**

db.Tshirtsmall.find({brand: "b2"})

**OUTPUT:**

A screen shot of a computer code

Description automatically generated

2. Consider the collection "collect2" please insert the data:

name:peter, job:engineer, company: ABC, age:20,adress: street(23 Bever st.),area:Paton,city: Portland.

name:mary, job:programmer,company:ABC, age:22,adress: street(25 Bever st.),area:Green,city: Portland.

name:paul, job:analyst,company:DEF, age:25,adress: street(12 Bever st.),area:Yellow,city: Seattle

Please prepare the following:

db.collect2.insertMany([{name:"peter", job:"engineer", company:"ABC", age:20, adress: "23 Bever St.", area: "Paton", city : "Portland"},{name:"mary", job:"programmmer", company: "ABC", age: 22, adress: "25 Bever St.", area: "Green", city : "Portland"},{name: "paul", job: "analyst", company: "DEF", age: 25, adress: "12 Bever St.", area: "Yellow", city:"Seattle"}])

a.change the Mary's job to XYZ. Please test the change.

**CODE:**

db.collect2.updateOne({name:"mary"}, {$set:{company:"XYZ"}})

**OUTPUT:**



b. find the number of documents

**CODE:**

db.collect2.countDocuments()

**OUTPUT:**



3. Consider the two lists of names:

names1="peter","ann","mary","paul".

names2="george","mary","joan","melissa".

Please find the documents of the " collect2" with the common names for each one of the lists

(names1,names2).

**CODE:**

db.collect2.aggregate([{$project: { commonNames : {$setIntersection : ["$names1", "$names2"]}}}, {$match : {commonElements: {$ne: []}}}])

**OUTPUT:**

A black screen with yellow green and white text

Description automatically generated

4. Consider the survey collection:

{ \_id: 1, results: [ { product: "abc", score: 10 }, { product: "xyz", score: 5 } ] }

{ \_id: 2, results: [ { product: "abc", score: 8 }, { product: "xyz", score: 7 } ] }

{ \_id: 3, results: [ { product: "abc", score: 7 }, { product: "xyz", score: 8 } ] }

a. find the documents that the results array contains at least one element with both product equal to "xyz" and score greater than or equal to 8.

**CODE:**

db.survey.find({'results': {$elemMatch : {'product': "xyz", 'score': {$gte: 8}}}})

**OUTPUT:**

A screen shot of a computer

Description automatically generated

b. find the documents with product : “xyz”. Is there any other solution?

**CODE:**

db.survey.find({results: {$elemMatch : {product: "xyz"}}})

OR

db.survey.find({"results.product" : "xyz"})

**OUTPUT:**

A screen shot of a computer program

Description automatically generated

5. Consider the data:

db.employees.insert({"name":"Adma","dept":"Admin","languages":["german","french","english","hindi"] ,"age":30, "totalExp":10});

db.employees.insert({"name":"Anna","dept":"Admin","languages":["english","hindi"],"age":35,

"totalExp":11});

db.employees.insert({"name":"Bob","dept":"Facilities","languages":["english","hindi"],"age":36,

"totalExp":14});

db.employees.insert({"name":"Cathy","dept":"Facilities","languages":["hindi"],"age":31,

"totalExp":4});

db.employees.insert({"name":"Mike","dept":"HR","languages":["english", "hindi",

"spanish"],"age":26, "totalExp":3});

db.employees.insert({"name":"Jenny","dept":"HR","languages":["english", "hindi",

"spanish"],"age":25, "totalExp":3});

Q1: write the query that get all departments where average age of employees is greater than or equal to 35 and totalExp less than or equal 10.

**CODE:**

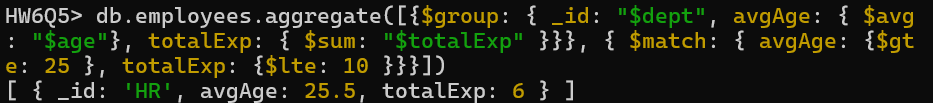
db.employees.aggregate([{$group: { \_id: "$dept", avgAge: { $avg: "$age"}, totalExp: { $sum: "$totalExp" }}}, { $match: { avgAge: {$gte: 35 }, totalExp: {$lte: 10 }}}])

**OUTPUT:**

A screen shot of a computer

Description automatically generated

If the query were to have the average age of employees is greater than or equal to 25



6a. Create a blog using database (myblog) and a table (entries). The user has to insert title, entry text and the date. At the end the user will post the entry.

The PHP program (add\_en.php) has also to validate the inputs providing analogous messages.

Example:

Entry Title: Math

Entry Text: Math is the science that works with numbers

XAMPP,PHP,MYSQL,practice5

b. Create also the program that can change the entry Text (Math in the number science).

Validate the change of the text in the blog

Please provide screenshot with the results

**CODE for A:**

<!--Q6A Add Blog Entries to Database-->

<!DOCTYPE HTML>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>add\_en</title>

</head>

<body>

<h1>Add Blog Entry</h1>

<form action="add\_en.php" method="post">

<p><label>Insert Title: <input type="text" name="title" space="20" maxlength="100"></label></p>

<p><label>Insert Entry: <textarea name="entry" rows="5" cols="50"></textarea></label></p>

<p><label>Date: <input type="date" name="date"></label></p>

<p><input type="submit" name="Submit" value="Add Entry"></p>

<?php

$errors = []; // Initialize an empty array to store errors

// Create the databse connection

$dbc = @mysqli\_connect("localhost", 'root', '') OR die('Could not connect MYSQL: '. mysqli\_connect\_error());

$query = "CREATE DATABASE IF NOT EXISTS myblog";

if (!mysqli\_query($dbc, $query))

echo "Error creating database: " . mysqli\_error($dbc);

$query = "USE myblog";

if (!mysqli\_query($dbc, $query))

echo "Error creating database: " . mysqli\_error($dbc);

$query = "CREATE TABLE IF NOT EXISTS entries(

id INT AUTO\_INCREMENT,

title VARCHAR(255) NOT NULL,

entry VARCHAR(255) NOT NULL,

date DATE NOT NULL,

PRIMARY KEY (id)

)";

if (!mysqli\_query($dbc, $query))

echo "Error creating database: " . mysqli\_error($dbc);

// If form is submitted

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

// Check if empty of each part of form

if (!isset($\_POST['title']) && empty($\_POST['title'])) {

$errors[] = "Title is required";

}

if (!isset($\_POST['entry']) && empty($\_POST['entry'])) {

$errors[] = "Entry is required";

}

if (!isset($\_POST['date']) && empty($\_POST['date'])) {

$errors[] = "Date is required";

}

// If there are no errors, proceed with insertion

if (empty($errors)){

$stmt = mysqli\_prepare($dbc, "INSERT INTO entries (title, entry, date) VALUES (?, ?, ?)");

// Bind parameters

mysqli\_stmt\_bind\_param($stmt, "sss", $title, $entry, $date);

// Set parameters and execute

$title = mysqli\_real\_escape\_string($dbc, trim($\_POST['title']));

$entry = mysqli\_real\_escape\_string($dbc, trim($\_POST['entry']));

$date = date('Y-m-d', strtotime($\_POST['date']));

if (mysqli\_stmt\_execute($stmt)) {

echo "<br>SUCCESS<br><br>";

echo '<p>Entry Title: ' . $title . '<br>

Entry Text: ' . $entry . '<br></p>';

echo "<p><a href=\"alter\_en.php?id=" . mysqli\_insert\_id($dbc) . "\">Alter the Entry Text</a></p>";

} else {

echo "Error: " . mysqli\_error($dbc);

}

// Close statement

mysqli\_stmt\_close($stmt);

}

}

// Display errors from NULL values

if (!empty($errors)) {

echo "<h3>Error List:</h3>";

echo "<ul>";

foreach ($errors as $error) {

echo "<li>$error</li>";

}

echo "</ul>";

}

// Close connection

mysqli\_close($dbc);

?>

</body>

</html>

**OUTPUT FOR A DOWN BELOW**

**OUTPUT FOR A:**

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

**CODE FOR B:**

<!--Q6B Add Blog Entries to Database-->

<!-- Accessed From add\_en.php -->

<!DOCTYPE HTML>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Update Entry</title>

</head>

<body>

<?php

//insert header

echo '<h1>Update a entry</h1>';

// Check for a valid user ID, through GET or POST:

if ((isset($\_GET['id'])) && (is\_numeric($\_GET['id']))) {

$id = $\_GET['id']; //accessed through href link

} else if ((isset($\_POST['id'])) && (is\_numeric($\_POST['id']))) {

$id = $\_POST['id']; //if id is gained from form submission

} else {

echo '<p class="error">This page has been accessed in error.</p>';

exit();

}

$dbc = @mysqli\_connect('localhost', 'root', '', 'myblog');

// Check if the form has been sumbitted:

if ($\_SERVER['REQUEST\_METHOD'] == 'POST') {

$error = ""; //intialize error string

// Check for new entry has input:

if (empty($\_POST['new\_entry'])) {

$error = 'Please insert a new entry';

} else {

$new\_entry = mysqli\_real\_escape\_string($dbc, trim($\_POST['new\_entry']));

}

if (empty($errors)) { // if there are no errors

// Make the query:

$query = "UPDATE entries SET entry='$new\_entry'

WHERE id=$id";

$result = @mysqli\_query($dbc, $query);

if (mysqli\_affected\_rows($dbc) == 1) {

//print a message:

echo '<p>The entry has been updated.</p>';

} else {

echo '<p class="error">The entry could not be due updated due to a system error.

We apologize for any inconvenience.</p>'; // Public message.

echo '<p>' . mysqli\_error($dbc) . '<br>Query: ' . $query . '</p>';

// Debugging message

}

} else { //Report the errors.

echo '<p class="error">The following error occurred:<br>'

. $error . '</p><p>Please try again.</p>';

}

}

//Retrieve the id's entry

$query = "SELECT entry FROM entries WHERE id=$id";

$result = @mysqli\_query($dbc, $query);

if (mysqli\_num\_rows($result) == 1) {

//Get the users's information:

$row = mysqli\_fetch\_assoc($result);

// Create the form:

echo '<form action="alter\_en.php" method="post">

<p>Old entry: &nbsp&nbsp'. $row['entry'] . '</p>

<p>New entry: <textarea name="new\_entry" rows="5" cols="50"></textarea></p>

<p><input type="submit" name="submit" value="Submit"></p>

<input type="hidden" name="id" value="' . $id .'"></form>';

} else {

echo '<p class="error">This page has been accessed in error.</p>';

}

//gives you link to return to add\_en.php

echo '<p><a href="add\_en.php">Return to add\_en.php</a></p>';

mysqli\_close($dbc);

?>

</body>

</html>

**OUTPUT FOR B DOWN BELOW**

**OUTPUT FOR B:**

