Criterion C

Contents

Database structure – explanation and justifications

This part will provide an overview of the database and justifications of different elements created.

Table	Primary key	Content of the table	Reason for creating table
students_info	ID_students	Information about students	
topic_info	ID_topic	Information about topics for studying	
unit_info	ID_unit	Information about units for studying	
sub_topic	ID_sub	Information about subtopics in units and topics	
sub_student	ID_sw	Remarks about students studying a subtopic	Due to "many to many" relationship between "students_info" and "subt_topic", a separate table is needed to avoid data redundancy
grades_info	ID_grade	Information about grades	
users	ID_users	Usernames and passwords	Ensure security of data access

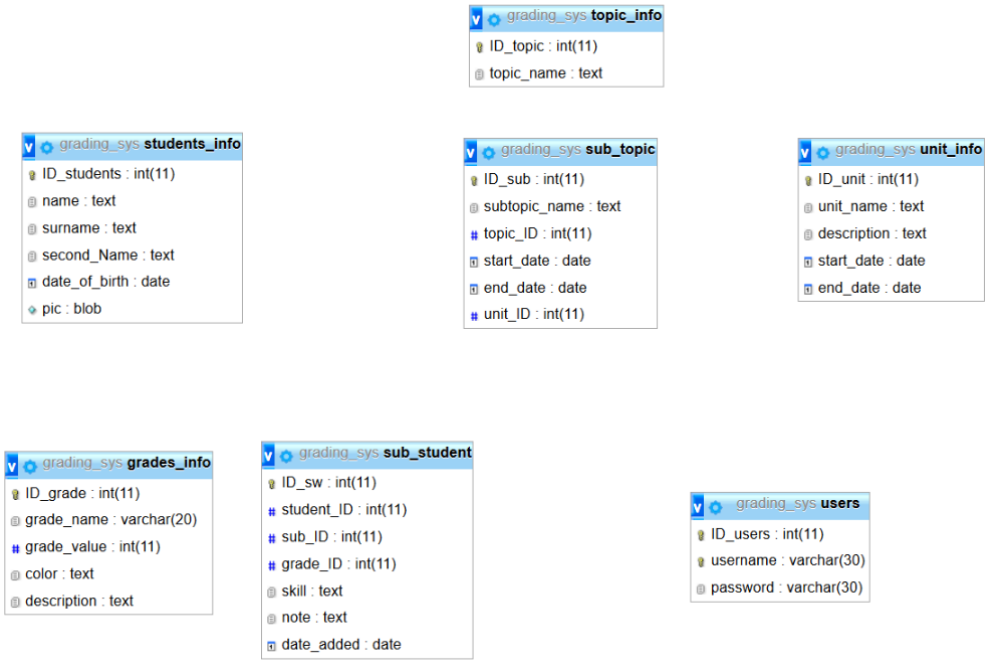


Figure 1 Database structure created using phpMyAdmin

The table and picture above represent the relational database created for this project. As you can see from *Figure 1*, various tables were created to meet needs of Ms. Magdalena.

It should be noted that there is no formal relation between any tables, as you can see from *Figure 1*. Relations between tables was achieved by having various PHP functions.

SQL commands used

Queries

Various SQL simple and complex queries were used to retrieve information from the database. The following examples will provide explanation of complex queries used.

Example #1 – retrieving information about subtopics

```
SELECT `sub_topic`.`ID_sub`, `sub_topic`.`subtopic_name`, `topic_info`.`topic_name`,
`unit_info`.`unit_name`, `sub_topic`.`start_date`, `sub_topic`.`end_date` FROM
`sub_topic`,`unit_info`, `topic_info` WHERE `sub_topic`.`topic_ID`=`topic_info`.`ID_topic`
AND `sub_topic`.`unit_ID`=`unit_info`.`ID_unit` ORDER BY `sub_topic`.`start_date`;
```

Figure 2 SQL code for retreiving data from sub_topic table

The example above selects to display information from fields: ID_sub, subtopic_name, topic_name, unit_name, start_date, end_date under condition that information from topic_ID is equal to the ID_topic, and that information from unit_ID is equal to the ID_unit. By running this SQL code, the output can be as follows:

ID_sub	subtopic_name	topic_name	unit_name	start_date ▲ 1	end_date
1	Shapes	Math	Unit of inquiry 1	2017-09-11	2017-09-18
3	Maps	English	Unit of inquiry 1	2017-12-04	2017-12-11

Figure 3

ID_sub	subtopic_name	topic_ID	start_date	end_date	unit_ID
1	Shapes	2	2017-09-11	2017-09-18	1
3	Maps	1	2017-12-04	2017-12-11	1

Figure 4 Output for example 1 if conditions are not set

From *Figure 3*, it should be noticed that instead of foreign keys, according names are displayed. This is achieved by setting conditions under which name must be displaced when a foreign key matches the primary key of the referenced table. Thus, in this case (see *Figure 4* for topic_ID and unit_ID values), names of the topics with IDs, 2 and 1 are displayed, and name of the units with ID 1 is displayed.

Data manipulation

One of the needs of Ms. Magdalena was ability to add, edit and delete information. Below are examples of how this need was achieved on a level of SQL commands.

Adding

```
INSERT INTO `unit_info` (`ID_unit`, `unit_name`, `description`, `start_date`, `end_date`)
    VALUES (NULL,'$unit','$description','$start_date','$end_date');
```

Figure 5

The code from above adds into the table “unit_info” listed values. It should be noted that dollar sign indicates a PHP variable. Consequently, values of PHP variables are inserted into the table. Furthermore, it should be noted that Null (no value) is passed to the field “ID_unit.” This is due to the fact that the field is autoincremented, so if no values are passed, it will create unique values automatically.

Deleting

```
DELETE FROM `students_info` WHERE `students_info`.`ID_students` = '$ID'
```

Figure 6

The code from above deletes a record from the table “students_info” where the ID of the student matches the value of PHP variable “ID”

Editing

```
UPDATE `students_info` SET `name`='$name', `surname`='$surname',
`second_Name`='$second_name', `date_of_birth`='$birth', `pic`='$pic' WHERE
`ID_students`='$student_ID'
```

Figure 7 SQL

In the *Figure 7* a record with ID that matches value of PHP variable “student_ID” is updated “students_info” using the values of PHP variables.

User interactions using HTML/PHP

Before the SQL code can be run to perform actions, user needs to choose what kind of actions need to be performed. Consequently, user needs to interact with the code through web pages. This chapter will overview main user interactions in the project.

Forms

In order to create interaction between web page and PHP file with SQL code, HTML form tag was used. The form tag allows to create a form for user inputs; and inside of the tag, post method is available. The post method allows to pass inputted

data to a targeted file in a form of associative array(in the case of this project, the PHP files with SQL code)

```
<form action = "add_unit_mysql.php" method="post">
  <div class="row">
    <div class="col-sm-9">
      <h1>Please input the following information to proceed</h1>
    </div>
  </div>
  <div class="row row-space">
    <div class="col-sm-2">
      <label for="unit">Unit Name</label>
    </div>
    <div class="col-sm-2">
      <input type = "text" name = "unit">
    </div>
  </div>
  <div class="row row-space">
    <div class="col-sm-2">
      <label for="description">Description</label>
    </div>
    <div class="col-sm-2">
      <input type = "text" name = "description">
    </div>
  </div>
  <div class="row row-space">
    <div class="col-sm-2">
```

Form tag with post method is used

Targeted PHP file

Input box

Index to be used in the array

Figure 8 Example of form tag to add new unit

```
<?php
  include 'functions.php';
  $connect=connection();
  $unit = mysqli_real_escape_string($connect,$_POST["unit"]);
  $description = mysqli_real_escape_string($connect,$_POST["description"]);
  $start_date = mysqli_real_escape_string($connect,$_POST["start_date"]);
  $end_date = mysqli_real_escape_string($connect,$_POST["end_date"]);
  $sql = "INSERT INTO `unit_info` (`ID_unit`, `unit_name`, `description`, `sta
  if(mysqli_query($connect, $sql)){
    echo "Records inserted successfully.";
} else{
    echo "ERROR: Could not able to execute $sql. " . mysqli_error($link);
}
  con_close($connect); ?>
```

PHP variable to be used to add information

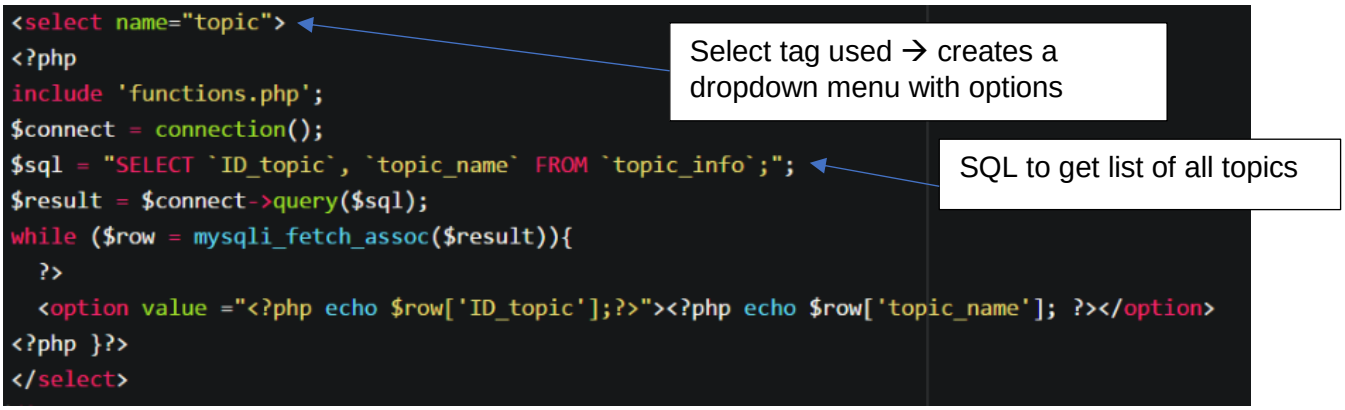
Accessing associative array passed by indexes

SQL code to be run

Figure 9 Example of PHP code to add new unit

Selecting option

One of the key user interaction happens when user has to select an option. Example of this interaction happens when adding new subtopic. In this example user needs to choose to which topic the subtopic is related to. Since list of topics can change anytime, I decided to have an interactive code that will display an option for each topic stored in the database. In order to achieve this, while loop and SQL query search were needed.




```
<select name="topic">
<?php
include 'functions.php';
$connect = connection();
$sql = "SELECT `ID_topic`, `topic_name` FROM `topic_info`;";
$result = $connect->query($sql);
while ($row = mysqli_fetch_assoc($result)){
  ?>
  <option value ="<?php echo $row['ID_topic'];?>"><?php echo $row['topic_name']; ?></option>
<?php }?>
</select>
```

Figure 10

The *Figure 10* shows an example of how while loop and SQL query were used. In the example, select tag is first declared. Then, PHP code is used to get all topic from the database. The data retrieved is the stored in a resource, which is later converted into an associative array. After array is stored, a while loop for each element inside of the array begins, i.e. while loop stops only when each element was read. For each iteration of the while loop, an option with topic name and topic ID as value is created.

While loop and SQL for viewing

Another key interaction happens when user wants to view information from the database. Since information can change, while loop and SQL search were used again.

```
</tr>
<?php while ($row=mysqli_fetch_assoc($result)){?>
    <tr>
      <th><?php echo $row['name'].' '.$row['surname'];?></th>
      <th><?php echo $row['subtopic_name']; ?></th>
      <th><?php echo $row['skill']; ?></th>
      <th><div style="background<?php echo $row['color']; ?>"></div>
      <th><?php echo $row['topic_name'];?></th>
      <th><?php echo $row['unit_name'];?></th>
      <th><?php echo $row['date_added'];?></th>
      <th><div class="container" style="width:75px">
```

This is also a nested loop, since it is located in a bigger loop used to create display information and create new rows

Figure 11

The *Figure 11* shows an example of while loop used to display information from “sub_student” table. First, before the loop starts, information from the database is retrieved and stored in associative array. Then, an iteration happens for every element in the associative array. For every iteration, new row is created and data is displayed in the row.

Deleting information

Another important interaction happens when user wants to delete information. This interaction is important since some information can be related to multiple tables. Thus, in order to avoid data redundancy, when deleting information, related information should be deleted too.

```php
<?php }elseif ($action=="delete") {
  $sql_delete="DELETE FROM `unit_info` WHERE `ID_unit`='$unit_ID'";
  if (mysqli_query($connect,$sql_delete)){
    $sql  = "SELECT `ID_sub` FROM `sub_topic` WHERE `unit_ID`='$unit_ID'";
    $result = $connect->query($sql);
    while ($row=mysqli_fetch_assoc($result)){
      $sql = "DELETE FROM `sub_student` WHERE `sub_ID`='row['ID_sub']'";
      $connect->query($sql);
    }
    $sql = "DELETE FROM `sub_topic` WHERE `unit_ID`='$unit_ID'";
    if (mysqli_query($connect,$sql)){
      header("Location: view_units.php");
    }
    else {
      echo "Error";
    }
  }
}?>
```

Figure 12

Figure 12, is an example how related information must be deleted. In this example, SQL code is first run to delete a record from “unit_info”. Since the table is related to “sub_topic” which is in turn related to “sub_student”, related data must be deleted from both “sub_topic” and “sub_student” To achieve this, first SQL code is run to find subtopics contain ID of the unit to be deleted. The data retrieved is stored in an associative array. Then loop is started for every element in the array. For every iteration, a record from “sub_student” with “sub_ID” equal to the ID stored in the array is deleted. Once, loop finishes, all subtopics containing “unit_ID”, which equals to the ID of the unit to be chosen, must be deleted.

Web design – CSS and JS

This part will provide an overview of key CSS and JS features used.

Grid

One of the need was optimization of web pages for devices with small screen sizes. To meet this need, I used a CSS library “bootstrap” The library allows to divide the web page into 12 column layouts, while also specifying how many columns must be used for big screens and small screens.

```
<div class="row">
  <?php while ($row = mysqli_fetch_assoc($result)){?>
    <div class="col-lg-4 col-sm-12">
      <div class="row"> <!-- beginning of upper part -->
        <div class="col-sm-9">
          <img src="<?php echo $row['pic'];?>">
        </div>
        <div class="col-sm-3">
```

Figure 13

Figure 13 is an example of grid used. This code is used to create student cards on home page. When small screen is used, the card takes up the whole row, but when big screen is used, there can be 3 cards per row.

Word Count: 1253