For project part 4, you're expected to load the database with thousands of records. For that, you need to be able to do the following subtasks:

Generate fake data to populate your database with the PHP Faker library and encode it in a json file;

Decode the fake json file and load the database through a PHP script.

Here are resources and examples to help achieve both tasks.

Connecting to MySQL from PHP

You may use these files to connect to your database. Make sure you enter your correct data in dbconfig.php. You are also encouraged to use the PHP manual (<https://www.php.net/manual-lookup.php?pattern=doc&lang=en&scope=404quickref>. ) as much as you need, especially on PDO (<https://www.php.net/manual/en/book.pdo.php>. ) classes

dbconfig.php

<?php

/\*\*

\* Configuration for database connection

\*

\*/

$host = "localhost";

$username = "root";

$password = "root";

$dbname = "test";

$dsn = "mysql:host=$host;dbname=$dbname";

$options = array(

PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION

);

?>

connection.php

<?php

try {

require "dbconfig.php"; //database access details

$connection = new PDO($dsn, $username, $password, $options); //create database connection and get handler

} catch(PDOException $error) {

//if connection failed, print error and exit;

echo "Database connection error: " . $error->getMessage() . "<BR>";

die;

}

?>

simple\_query.php

<?php

try{

//create mysql connection

require("connection.php");

//query statement string

$sql = "SELECT \* FROM musician WHERE dob < :date;";

$date = "1990-01-01";

//sanitize statement

$statement = $connection->prepare($sql);

//binding parameters

$statement->bindParam(':date', $date, PDO::PARAM\_STR);

//execute statement

$statement->execute();

//get all results into an array

//only gets associative keys

$result = $statement->fetchAll(PDO::FETCH\_ASSOC);

/\* iterates through table results

and assigns the column name to $column

and the column data to $value \*/

foreach ($result as $row) {

foreach ($row as $colkey => $colval) {

echo $colkey . ": " . $colval . "<BR>";

}

echo "<BR>";

}

} catch(PDOException $error) {

//if connection failed, print error and exit;

echo "Error: " . $error->getMessage() . "<BR>";

die;

}

?>

PHP Faker

PHP Faker is installed for all users, so you do not need to install it in your public\_html. Here is a file with an example (and needed require) to use PHP Faker. Again, you are encouraged to check out its documentation (<https://fakerphp.github.io/>. ) at will.

simple\_faker.php

<pre>

<?php

// require Faker code

require\_once 'composer/vendor/autoload.php';

// use the factory to create a Faker\Generator instance

$faker = Faker\Factory::create();

// generate a single email and print it

echo $faker->email();

// print a new line

echo "\n";

// generate a lot of data by calling methods

for ($i = 0; $i < 10000; $i++){

$names[$i] = $faker->name();

}

//recursively print $names vector

print\_r($names);

?>

</pre>

JSON

Json manipulation is very simple in PHP. Use the documentation here (<https://www.php.net/manual/en/function.json-decode>. ) and here (<https://www.php.net/manual/en/function.json-encode>. ) to get started.

Final Comments

It is OK to base your code in examples from PHP.net or other sources. However, it is not OK to use code that does exactly what you need. Whenever you use example code, indicate the source in comments within your code, otherwise you will get a zero.

There is sufficient time to complete this assignment, and there will not be any extensions. So, start early and raise any questions you may have with sufficient time.