Indexes can improve the performance of frequently asked queries. Views can be used to simplify complex queries or control data access. Instead of only inserting several tuples into tables by hand as you did in the last part, this time you will choose two tables from your schema from Project Part #3 to load a large number of tuples in order to make meaningful comparisons of running times.

[20] Generate a dataset of concerts, bands, songs, and setlists. You will use the PHP (Links to an external site.) faker library (<https://fakerphp.github.io/>.) to write the code necessary to create the fake data and input them into the database.

Requirements for Data Generation:

Generate 300 bands. Output success or failure result to the screen.

Note: you don't need to generate musicians for all the bands;

Generate 2,000 songs. Output success or failure result to the screen.

Note: it is OK that all songs are written by just a few musicians (the ones you originally created as part of project part 2);

Generate 1,000 concerts. Output success or failure result to the screen.

Generate 10,000 set\_list entries. Output success or failure result to the screen.

Notes: you will need to use a mix of the bands, concerts, and songs that you already generated, so plan your data structures wisely.

It's OK that songs are not composed by band members (after all, for this exercise, it's ok for bands to have no members!).

Output your fake data to a JSON file. You should use the json\_encode() (Links to an external site.) function to generate the JSON format, and fopen() (Links to an external site.), fwrite() (Links to an external site.), and fclose() (Links to an external site.) to write to disk. We will go through an example in class.

[30] Write a PHP program to load your data from the previous task into your database. Your application should connect to your database, delete all records from your selected tables, open your JSON files (fopen() (Links to an external site.), fread() (Links to an external site.), and fclose(), (Links to an external site.) json\_decode() (Links to an external site.)), and insert the data. Your method must not be susceptible to SQL injection from the text files. We will go through an example in class.

Before and after loading, show the result of query "select count(\*) from <TABLENAMEHERE>".

[20] Create an index for non-PK attributes in band, song, and concert tables. Run a query that leverages the indexes. Compare the running times of the query with and without using the index. The query you create should take at least twice as long without the index as the query with the index. (HINT: If you are having difficulty getting significant running time differences, try using additional joins, subqueries, and ORDER BY clauses.)

Explain why the presence of the index causes the running-time difference. Submit the queries and running times.

[30] Create a view that uses a combination of the tables from part 1 that includes at least one aggregate function and group by clause. Ensure that the view results are meaningful. Explain what the view does. Run a query using the view. Then, run an equivalent query without using the view. Try to update the contents of the view (using SQL UPDATE syntax).

Explain what happened and why.

DELIVERABLES: (Check that you have everything on this list in your submission)

The PHP files for Tasks 1 and 2.

NEW: your files should be stored in a folder on your local web space in the following structure:

$HOME/public\_html/database\_generator/

in other words, the files must be stored at:

/home/username/public\_html/database\_generator/

You may choose the file structure, as long as you follow these guidelines:

Task 1 can be fully executed by accessing the file datagen.php

Task 2 can be fully executed by accessing the file dataload.php

This means that, if you decide to use support files, such as config.php, and any other file (for example with functions), they should be included in your datagen.php and dataload.php code such that by accessing the files in order will generate the json file and populate the database from the json file respectively.

Make sure that the files are uploaded to the server within the deadline. If you change (or the files after the deadline, you will receive the late penalty (if it's within 24h of the deadline) or no credit at all.

For safety, you may still submit the PHP files on Canvas. However, those will be used for grading only in extenuating circumstances. Typically, the grading will be done on the server files, as described above.

A PDF format report including:

#1

The first and last ten lines of your JSON file (or top 1000 characters and bottom 1000 characters)

#2

A description of what your code does and how to run it.

Screenshots from the output of your code.

The results of "select count(\*) <TABLENAME>;" for all tables after inserting data – Important! Many of you lost unnecessary points in P3 due to not printing results

#3

The query to create your index

The timed query and result

The running time of your timed query before creating your index

The running time of your timed query after creating your index

A detailed explanation for how the index helps your specific query

#4

The query to create your view

The query and result of a query that uses the view

The query and result of the equivalent query which does not use the view

The query to try to update the view

An explanation to what the view does

An explanation for what happened when you tried to update the view

索引可以提高经常询问的查询的性能。视图可以用来简化复杂的查询或控制数据访问。这次你将从项目第三部分的模式中选择两个表来加载大量的图元，以便对运行时间进行有意义的比较，而不是像上一部分那样只用手插入几个图元到表中。

[20] 生成一个音乐会、乐队、歌曲和节目单的数据集。你将使用PHP(链接到外部网站。)faker库(链接到外部网站。)来编写必要的代码来创建假数据并将它们输入数据库。

对数据生成的要求。

生成300个波段。将成功或失败的结果输出到屏幕上。

注意：你不需要为所有的乐队生成音乐人。

生成2,000首歌曲。输出成功或失败的结果到屏幕上。

注意：所有的歌曲都是由几个音乐家（你最初作为项目第二部分创建的那些音乐家）写的，这也没关系。

产生1,000场音乐会。将成功或失败的结果输出到屏幕上。

生成10,000个set\_list条目。将成功或失败的结果输出到屏幕上。

注意：你将需要混合使用你已经生成的乐队、音乐会和歌曲，所以要明智地计划你的数据结构。

歌曲不是由乐队成员创作的也没关系（毕竟，在这个练习中，乐队没有成员也没关系！）。

把你的假数据输出到一个JSON文件。你应该使用json\_encode() (Links to an external site.) 函数来生成JSON格式，使用fopen() (Links to an external site.) 、fwrite() (Links to an external site.) 和fclose() (Links to an external site.) 来写入磁盘。我们将在课堂上通过一个例子。

[30] 编写一个PHP程序，将前一个任务中的数据加载到你的数据库中。你的程序应该连接到你的数据库，从你选择的表中删除所有记录，打开你的JSON文件（fopen() (链接到外部网站。), fread() (链接到外部网站。), 和fclose(), (链接到外部网站。) json\_decode() (链接到外部网站。)），并插入数据。你的方法必须不容易受到来自文本文件的SQL注入的影响。我们将在课堂上通过一个例子。

在加载之前和之后，显示查询 "select count(\*) from <TABLENAMEHERE>"的结果。

[20] 为乐队、歌曲和音乐会表中的非PK属性创建一个索引。运行一个利用索引的查询。比较使用和不使用索引的查询的运行时间。你创建的查询在没有索引的情况下所花的时间应该至少是有索引的查询的两倍。(提示：如果你很难获得明显的运行时间差异，可以尝试使用额外的连接、子查询和ORDER BY子句）。

解释为什么索引的存在导致了运行时间的差异。提交查询和运行时间。

[30] 创建一个视图，使用第一部分中的表的组合，包括至少一个聚合函数和group by条款。确保视图的结果是有意义的。解释该视图的作用。使用该视图运行一个查询。然后，在不使用视图的情况下运行一个等效的查询。尝试更新视图的内容（使用SQL UPDATE语法）。

解释发生了什么以及为什么。

要交的文件：（请检查你的提交物中是否有此清单上的所有内容）

任务1和任务2的PHP文件。

新的：你的文件应该存储在你本地网络空间的一个文件夹中，结构如下。

$HOME/public\_html/database\_generator/。

换句话说，这些文件必须存放在。

/home/username/public\_html/database\_generator/。

你可以选择文件结构，只要你遵循这些准则。

任务1可以通过访问文件datagen.php来完全执行。

任务2可以通过访问文件dataload.php来完全执行。

这意味着，如果你决定使用支持文件，如config.php，和任何其他文件（例如带有函数），它们应该包含在你的datagen.php和dataload.php代码中，这样通过访问文件的顺序将分别生成json文件和从json文件中填充数据库。

确保文件在期限内上传到服务器。如果你在截止日期后改变（或文件），你将收到逾期罚款（如果是在截止日期的24小时内）或完全没有信用。

为了安全起见，你仍然可以在Canvas上提交PHP文件。但是，只有在情有可原的情况下，这些文件才会被用来打分。通常情况下，评分将在服务器文件上进行，如上所述。

一份PDF格式的报告包括。

#1

你的JSON文件的第一和最后十行（或前1000个字符和后1000个字符）。

#2

对你的代码的描述以及如何运行它。

你的代码输出的截图。

插入数据后所有表的 "select count(\*) <TABLENAME>; "的结果--重要! 你们中的许多人由于没有打印结果而在P3中失去了不必要的分数

#3

创建索引的查询

计时查询和结果

创建索引前的定时查询的运行时间

创建索引后的定时查询的运行时间

关于索引如何帮助你的特定查询的详细解释

#4

创建视图的查询

使用该视图的查询和结果

不使用视图的同等查询的查询和结果

尝试更新视图的查询

解释该视图的作用

解释当你试图更新视图时发生了什么？

通过www.DeepL.com/Translator（免费版）翻译