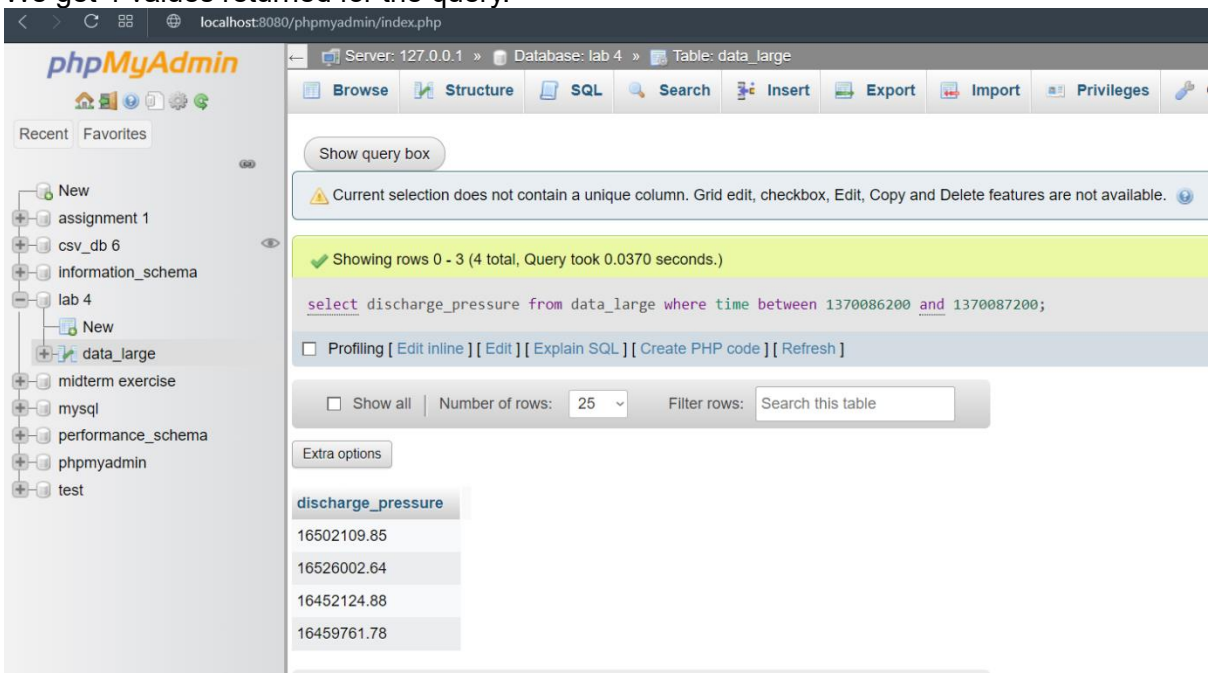
Objective: The objective of this lab is to show the usefulness of indices given a large dataset. Keep in mind that 'large' is subjective, using PHPMyAdmin, the largest dataset we can import to work with is about 41MB. That is large enough to make indices worthwhile but not as large as some of the datasets you might encounter in your next job!

Instructions:
In the lab folder you will see a CSV file named data-large. This data was obtained from Schlumberger, it is publicly available and completely ethical to use. I altered the data so that it is easier to import into PHPMyAdmin. You can find out more about the dataset here: Schlumberger Dataset.

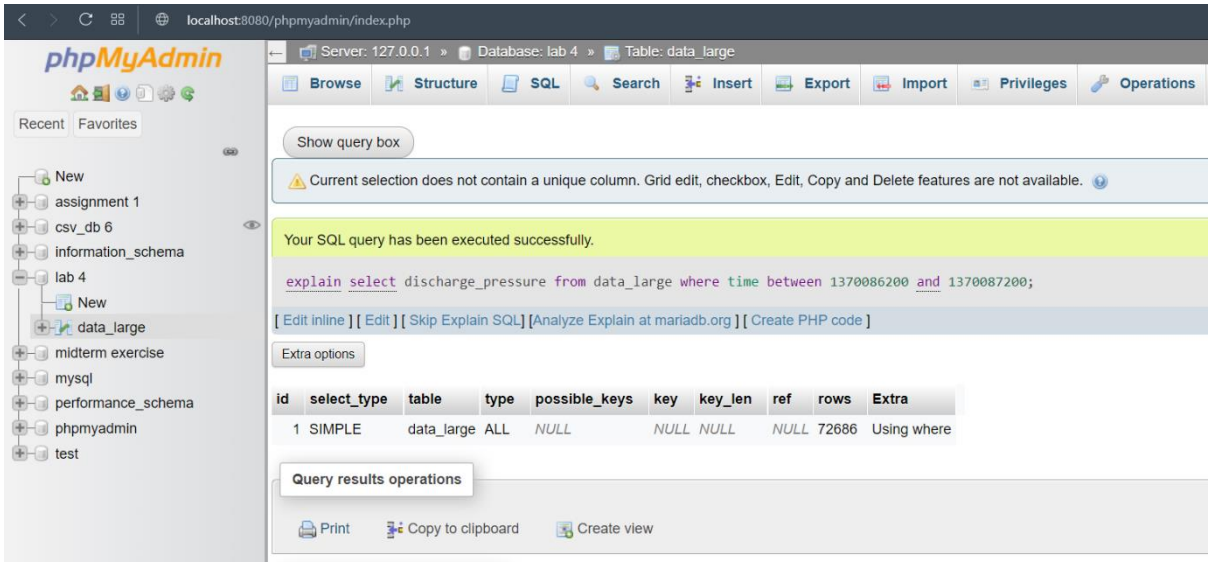
- Import data-large into a database in PHPMyAdmin. You can name the database whatever you like.
- If you run into some errors while importing such as mysql server has gone away or detected packet larger than max allowable packet, you will need to edit the my.ini file in your xampp/mysql/bin directory. It is a simple fix, you just need to set some fields higher than the default. A quick Google search should suffice.
- Create and run a query that returns the discharge_pressure between time = 1370086200 and time = 1370087200. Screenshot your query + your result.
- Use the EXPLAIN operation on the query you ran in the step 3. Screenshot your query + your result.
- Create an index on the time column. Screenshot your query + your result.
- Finally run the query from step 4 again, did anything change? Is it running more efficiently? Why or why not? Screenshot your query + your result, and include a short explanation.
- The EXPLAIN operation gives an output table. Explain what each column indicates (one sentence per column).

Aim: To show the usefulness of indices given a large dataset.

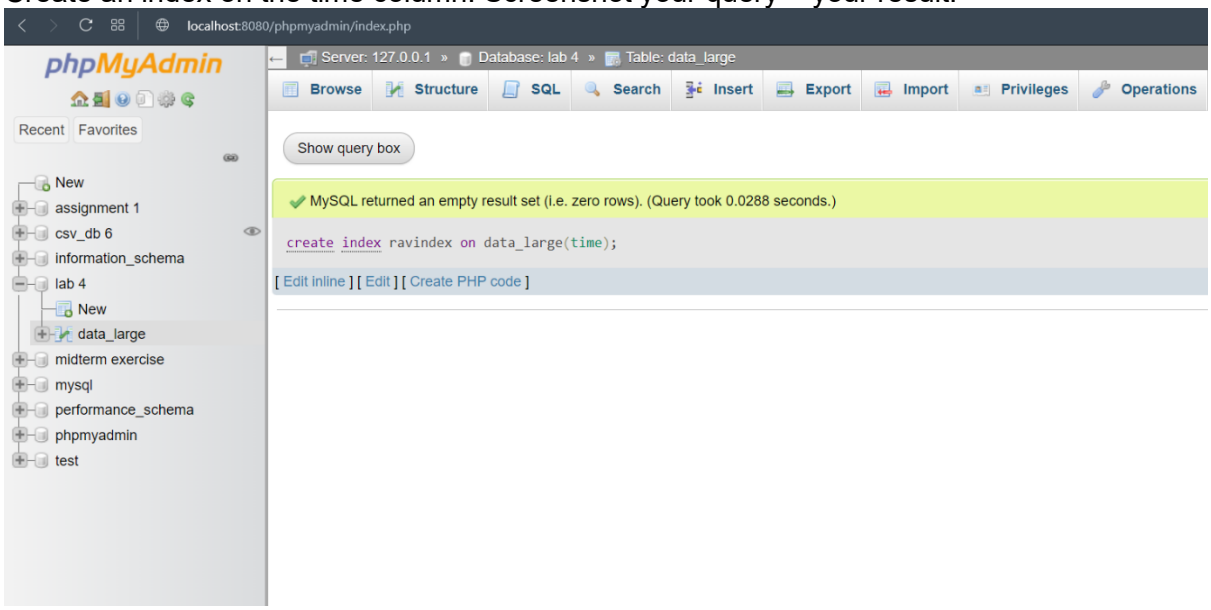
Successfully imported Schlumberger Dataset (data_large.csv)
Named the database accordingly.
Create and run a query that returns the discharge_pressure between time = 1370086200 and time = 1370087200. Screenshot your query + your result.
We get 4 values returned for the query.



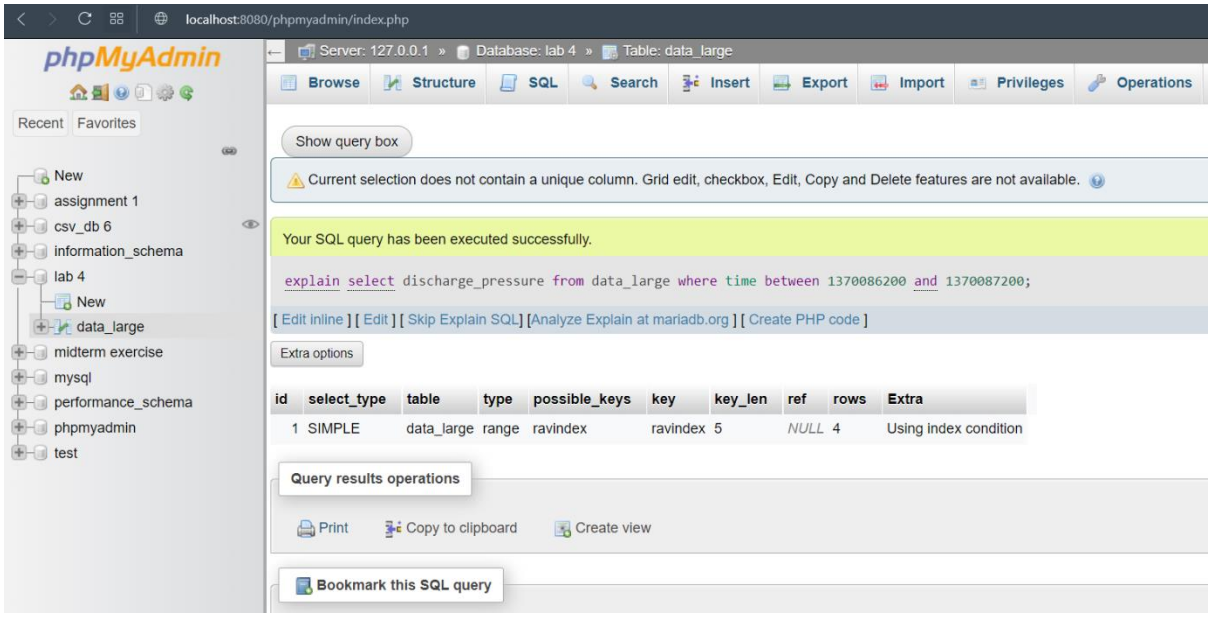
Use the EXPLAIN operation on the query you ran in the step 3. Screenshot your query + your result.



Create an index on the time column. Screenshot your query + your result.



Finally run the query from step 4 again, did anything change? Is it running more efficiently? Why or why not? Screenshot your query + your result, and include a short explanation.



After adding the index on the column, the “explain” operation is done which is shown in the below figure. Here it is observed that the time required for retrieving the data is faster as compared to without an index.

Indexes are used to retrieve data more quickly because they establish a secondary lookup table that is utilised to locate the necessary rows in accordance with the where clause of the query.

There are no keys in the explanation output table, thus all of the rows in the "data large" database are searched in order to produce the results. This execution was quicker after applying an index because just the necessary rows were searched.

The EXPLAIN operation gives an output table. Explain what each column indicates (one sentence per column).

The “Explain” operation is used in SQL to provide the overall information of the executed query.

a. Id: - The unique identification number assigned to every query passed to the operation.

b. Select_type:- The query's type is indicated by select type, which in our case displays "Basic" because it is a simple select statement. If we add a union with another query, a new row with select type set to "union" will be displayed.

c. Table: - The referred table for the query is shown under this column.

d. Type: - The type of the join used for the query.

e. Possible_keys: - This column shows the most appropriate key which could be used to make the query most optimized.

f. Key: - the key that the query actually utilised. In our scenario, "ravindex" is displayed as the key that will be utilised after the index has been applied.

g. Key_len: - The size of the key column is shown under this column.

h. Ref: - This column shows the columns which were used for the comparison with the index.

i. Rows: - This column shows the total number of rows which was scanned for getting the required output.

j. Filtered: - The percentage of the rows which were filtered by the condition used on the table.

k. Extra: - This column displays any additional comments for the query that was run.
