

TUNING SISTEM MANAJEMEN BASIS DATA



ITERA

Tugas Praktikum Manajemen Basis Data

Disusun oleh:

M. Hafidh Dliyaul Haq /14117030

Dosen pengampu:

Arief Ichwani, S.Kom., M.Cs.

PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI PRODUKSI INDUSTRI DAN INFORMASI

INSTITUT TEKNOLOGI SUMATERA

2019

DAFTAR ISI

HALAMAN JUDUL	i
DAFTAR ISI.....	ii
DAFTAR GAMBAR.....	iii
BAB I STUDI LITERATUR.....	1
1.1 <i>Tuning: Indexing</i>	1
1.2 <i>Tuning: Setting Configuration DBMS</i>	1
BAB II DESKRIPSI PERCOBAAN	2
2.1 Sistematika Percobaan	2
2.2 Sebelum <i>Tuning</i>	7
2.3 <i>Tuning: Indexing</i>	7
2.4 <i>Tuning: Setting Configuration DBMS</i>	7
BAB III HASIL DAN PEMBAHASAN	9
3.1 Hasil	9
3.2 Pembahasan.....	10
DAFTAR PUSTAKA	iv
LAMPIRAN.....	v

DAFTAR GAMBAR

Gambar 1 Error dalam Eksekusi Mulai dari Data 4.....	2
Gambar 2 Membuat Basis Data	3
Gambar 3 Membuat Tabel	4
Gambar 4 Pembuatan Tabel Berhasil	4
Gambar 5 Eksekusi File Java.....	5
Gambar 6 Impor File berisi query untuk Memasukkan Nilai pada Tabel	5
Gambar 7 Impor File berhasil.....	6
Gambar 8 Impor file untuk Menghapus Nilai pada Tabel	6
Gambar 9 Pembuatan Index Menggunakan Terminal	7
Gambar 10 Grafik Data 1.....	9
Gambar 11 Grafik Data 2.....	10
Gambar 12 Grafik Data 3.....	10

BAB I

STUDI LITERATUR

1.1 *Tuning: Indexing*

Index merupakan sebuah objek dalam sistem basis data untuk mempercepat proses *query* data. Ketika basis data dibuat tanpa menggunakan *index*, performa basis data dapat menurun yang disebabkan oleh sumber daya CPU banyak digunakan untuk pencarian data dengan metode *table-scan*. *Index* membuat pencarian data lebih cepat dan tidak menghabiskan sumber daya CPU yang terlalu banyak.

Index merupakan objek struktur data yang tidak bergantung dengan struktur tabel. Setiap *index*, memiliki nilai kolom dan penunjuk(*row id*). Penunjuk tersebut akan langsung mengarahkan ke tabel tujuan sehingga dapat menghindari *full table-scan*. Berikut ini perlunya penggunaan *index*:

1. Kolom sering digunakan klausa *WHERE* atau kondisi *JOIN*
2. Kolom berisi nilai yang banyak
3. Kolom berisi banyak nilai *null*
4. Tabel berukuran besar dan sebagian *query* menampilkan data kurang dari 2 s.d. 4%

1.2 *Tuning: Setting Configuration DBMS*

Seting Configuration DBMS atau pengaturan konfigurasi DBMS merupakan salah satu cara untuk meningkatkan performa dari sistem basis data yang digunakan. *Tuning* dengan konfigurasi DBMS dilakukan dengan cara mengubah beberapa parameter pada konfigurasi DBMS yang digunakan, misalnya *MySQL*, *PostgreSQL*, *MariaDB*, dll.

Parameter yang terdapat pada konfigurasi DBMS memiliki nilai awal(*default*). Nilai awal tersebut masih belum berpengaruh terhadap performa jika basis data masih skala kecil dengan proses yang sedikit. Penurunan performa dapat terjadi ketika basis data sudah kompleks sehingga diperlukan pengaturan konfigurasi DBMS yang digunakan. Parameter yang akan diubah untuk meningkatkan performa dipilih sesuai dengan kebutuhan basis data kinerja laptop atau komputer yang digunakan.

BAB II

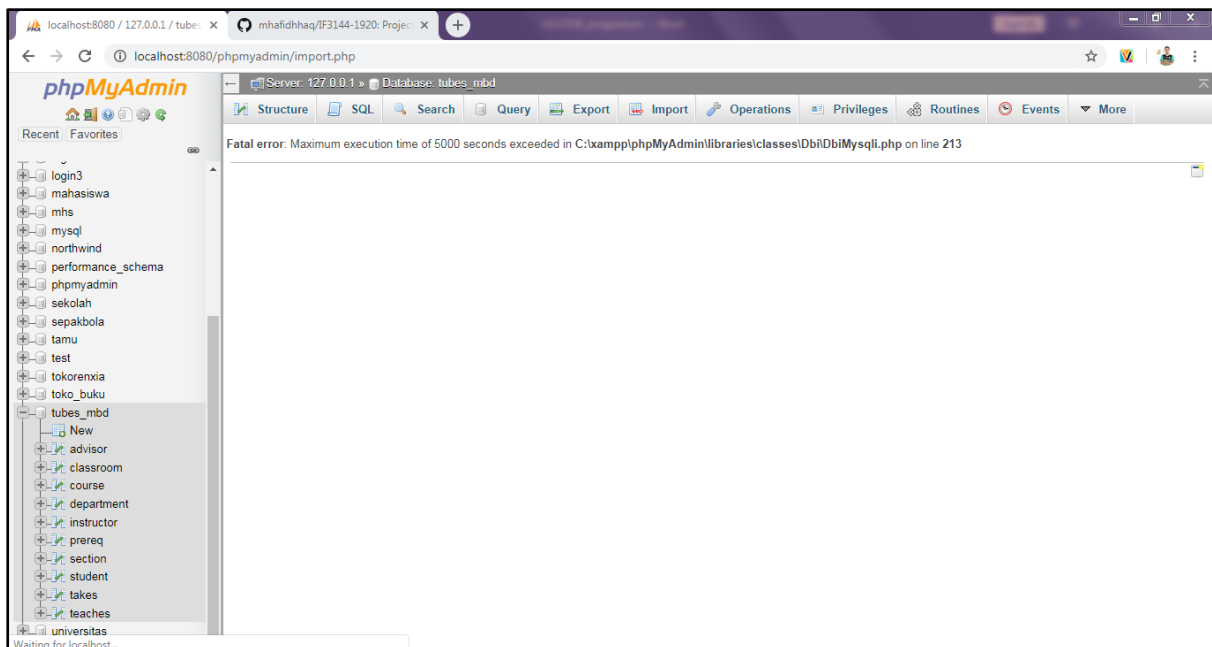
DESKRIPSI PERCOBAAN

2.1 Sistematika Percobaan

Percobaan yang dilakukan adalah menghitung waktu eksekusi 7 data dengan menggunakan 5 buah *query*. Data yang akan diuji sebagai berikut.

1. advisor = 100, student = 100, section = 200,takes = 200
2. advisor = 200, student = 200, section = 400,takes = 400
3. advisor = 500, student = 500, section = 1000,takes = 1000
4. advisor = 700, student = 700, section = 20000,takes = 20000
5. advisor = 1000, student = 1000, section = 100000,takes = 1000000
6. advisor = 1800, student = 1800, section = 180000,takes = 1800000
7. advisor = 10000, student = 10000, section = 30000000,takes = 30000000

Dari ketujuh data, hanya 3 data yang akan digunakan yaitu data 1, data 2, dan data 3. Hal ini disebabkan oleh performa *laptop* penulis yang tidak memadai sehingga membutuhkan waktu yang lama dalam mengeksekusi data 4 s.d. data 7 dan gagal dalam impor data ke basis data. Berikut ini gambar dari kendala yang dialami penulis dalam mengeksekusi mulai dari data 4.



Gambar 1 Error dalam Eksekusi Mulai dari Data 4

Waktu eksekusi pada DBMS telah diubah dari 30 menjadi 5000, tetapi masih belum memadai untuk mengeksekusi mulai dari data 4.

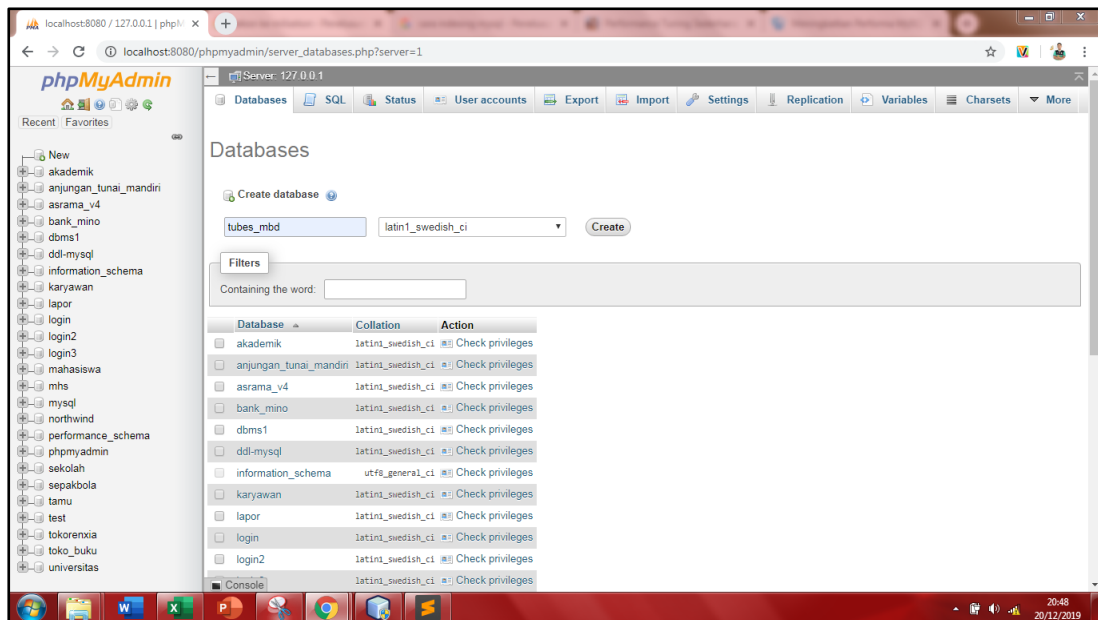
Berikut ini 5 *query* yang akan digunakan untuk menguji ketiga data.

1. `SELECT * FROM student;`
2. `SELECT * FROM student WHERE tot_cred > 30;`
3. `SELECT `name`, dept_name FROM student WHERE tot_cred > 30;`
4. `SELECT * FROM takes JOIN student ON takes.ID = student.ID JOIN section ON takes.course_id = section.course_id;`
5. `SELECT student.`name`, student.dept_name, takes.sec_id AS pengambilan, takes.semester, section.room_number, section.building, course.course_id, course.dept_name FROM takes JOIN student ON takes.ID = student.ID JOIN section ON takes.course_id = section.course_id JOIN course ON section.course_id = course.course_id;`

Proses pengujian dalam menghitung waktu eksekusi dilakukan melalui 3 tahap yaitu, pengujian sebelum *tuning*, pengujian setelah *tuning* menggunakan *indexing*, dan pengujian setelah *tuning* dengan pengaturan konfigurasi DBMS. DBMS yang digunakan pada percobaan adalah *MySQL*.

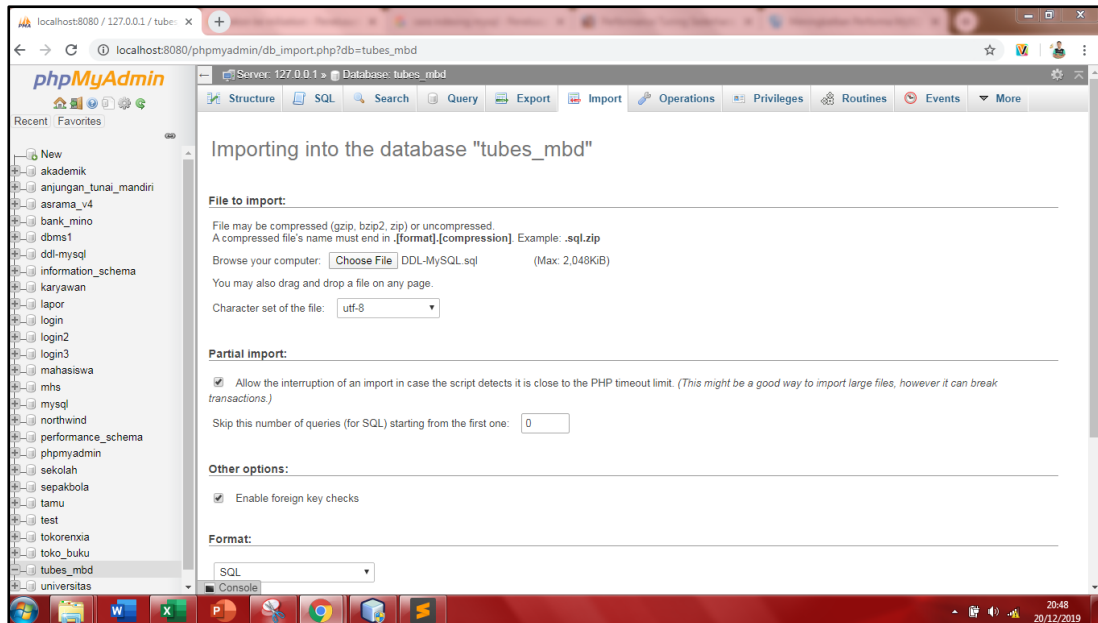
Langkah-langkah percobaan secara umum yang akan digunakan pada setiap tahap pengujian adalah sebagai berikut.

1. Buat basis data baru. Nama basis data yang penulis buat yaitu *tubes_mbd*.

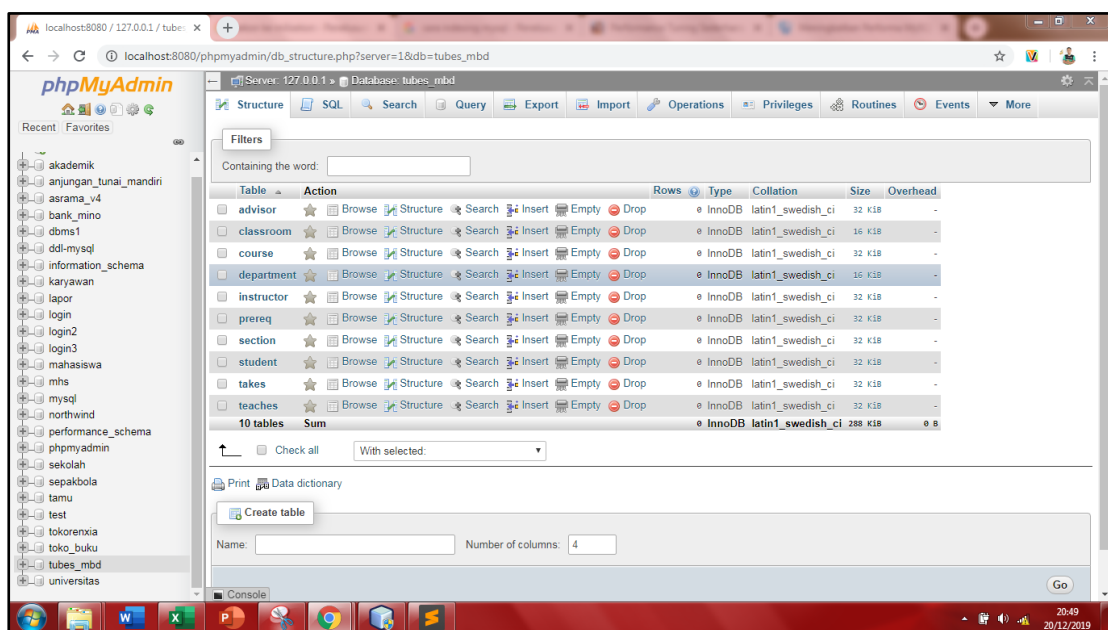


Gambar 2 Membuat Basis Data

2. Membuat tabel dengan menggunakan file DDL-MySQL.sql. Impor file tersebut ke basis data.

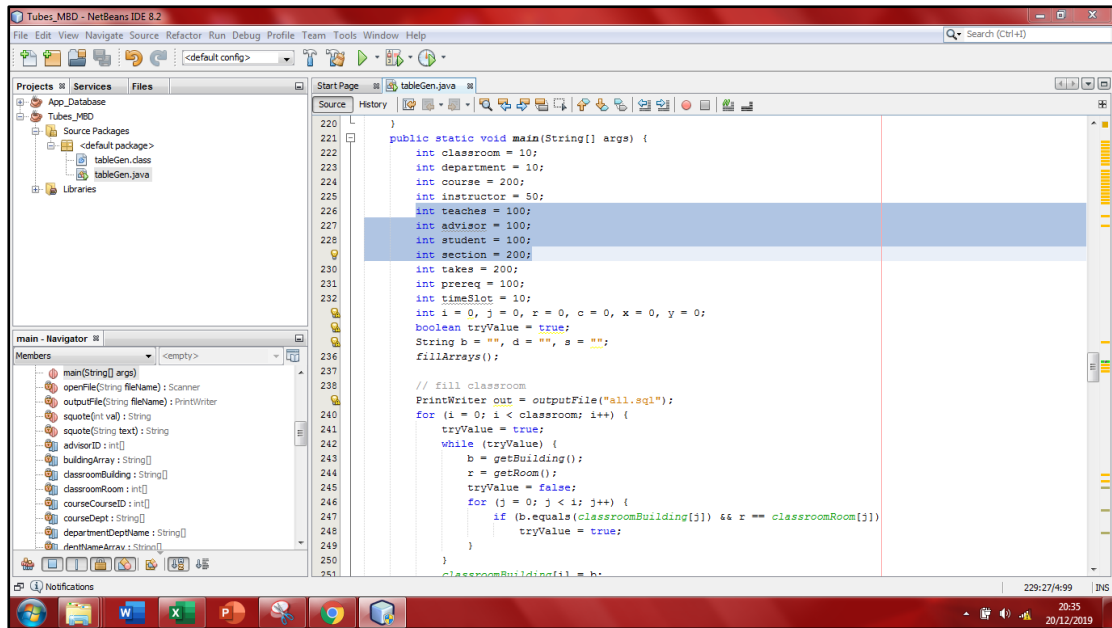


Gambar 3 Membuat Tabel



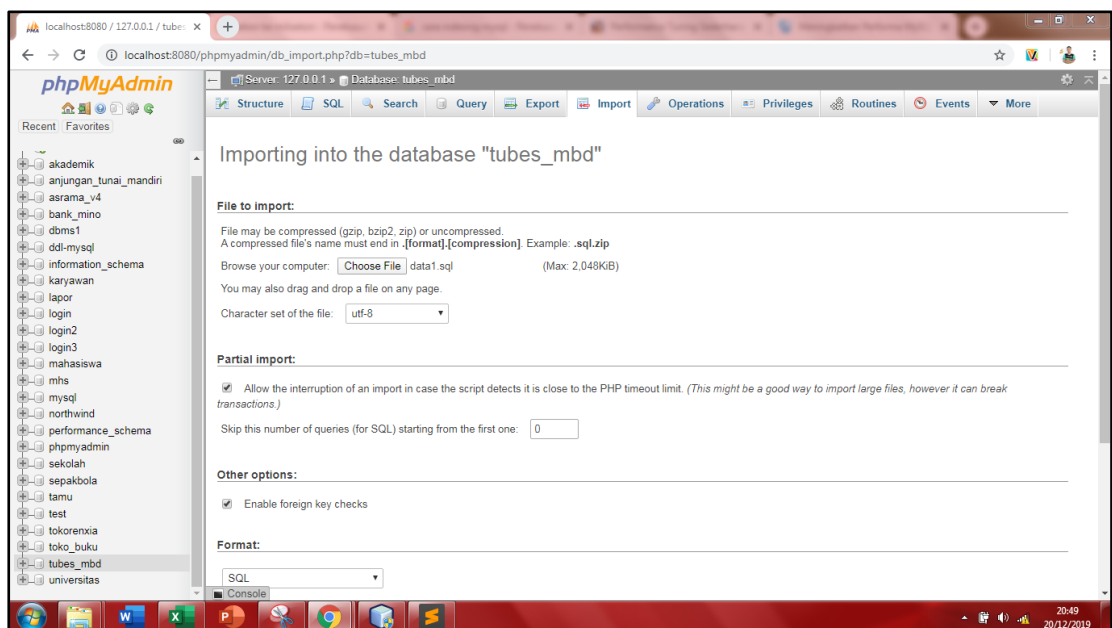
Gambar 4 Pembuatan Tabel Berhasil

3. Buka file *java* yang akan digunakan untuk mendapatkan *query*. *Query* yang diperoleh berisi nilai-nilai yang akan dimasukkan ke dalam tabel. Lakukan perubahan nilai pada bagian *main* dari file *java* sesuai dengan data-data yang akan diuji. Ketika file *java* dijalankan, hasil yang didapat adalah file dengan nama *all.sql*.

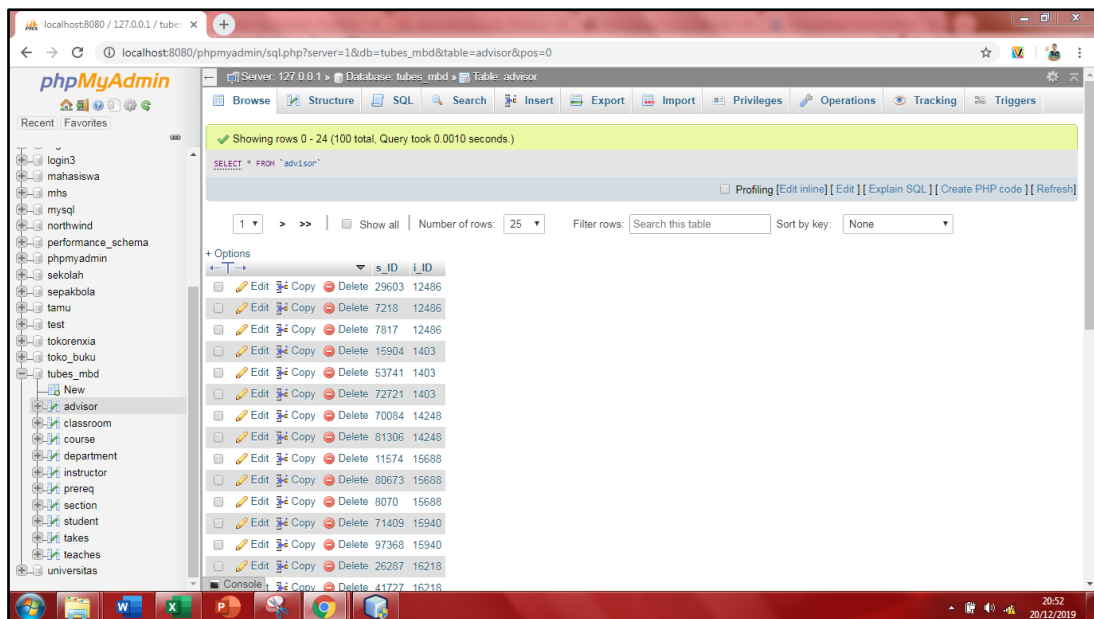


Gambar 5 Eksekusi File Java

4. Setelah file all.sql didapatkan, impor file tersebut ke basis data untuk mengisi nilai pada setiap tabel

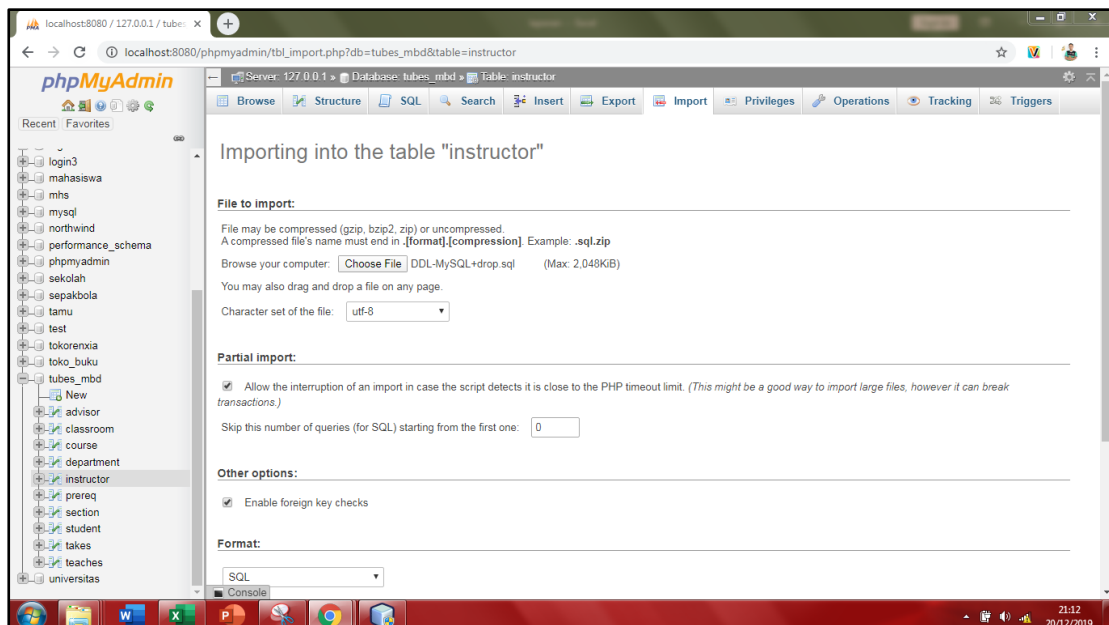


Gambar 6 Impor File berisi query untuk Memasukkan Nilai pada Tabel



Gambar 7 Impor File berhasil

5. Untuk menguji setiap data, lakukan pemrosesan kelima *query* yang digunakan dengan menggunakan terminal. Khusus untuk tahap *tuning*, terdapat langkah-langkah yang akan dijelaskan pada subbab berikutnya.
6. Setelah pemrosesan setiap data selesai, lakukan *reset* pada basis data agar nilai pada tabel kembali kosong. Penghapusan nilai pada tabel dilakukan dengan cara impor file DDL-MySQL+drop.sql.



Gambar 8 Impor file untuk Menghapus Nilai pada Tabel

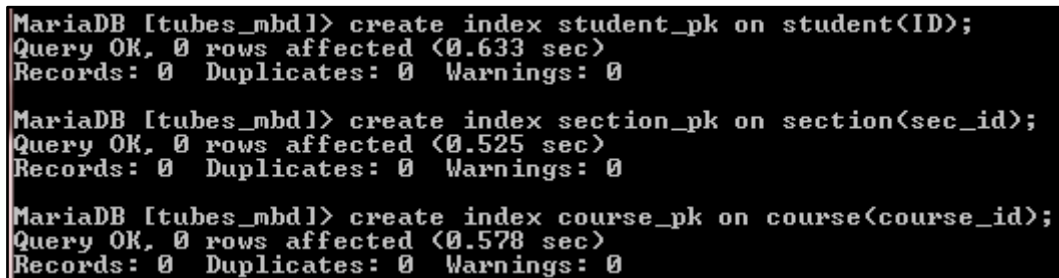
2.2 Sebelum Tuning

Langkah-langkah yang dilakukan untuk menguji data sebelum *tuning* tidak ada tambahan proses, sama seperti langkah-langkah umum yang dijelaskan di awal.

2.3 Tuning: Indexing

Langkah-langkah tambahan yang dilakukan dalam indexing adalah membuat *index* pada *field* yang dipilih. *Field* dipilih berdasarkan frekuensi kemunculan paling besar atau *field* yang sering digunakan pada *query* untuk menguji data. Pada percobaan, *field* yang akan dibuat *index* yaitu:

1. *Field ID* pada tabel *student*.
2. *Field sec_id* pada tabel *section*.
3. *Field course_id* pada tabel *course*.



```
MariaDB [tubes_mbd] > create index student_pk on student(ID);
Query OK, 0 rows affected (0.633 sec)
Records: 0 Duplicates: 0 Warnings: 0

MariaDB [tubes_mbd] > create index section_pk on section(sec_id);
Query OK, 0 rows affected (0.525 sec)
Records: 0 Duplicates: 0 Warnings: 0

MariaDB [tubes_mbd] > create index course_pk on course(course_id);
Query OK, 0 rows affected (0.578 sec)
Records: 0 Duplicates: 0 Warnings: 0
```

Gambar 9 Pembuatan Index Menggunakan Terminal

2.4 Tuning: Setting Configuration DBMS

Langkah-langkah tambahan yang dilakukan dalam konfigurasi DBMS adalah melakukan pengaturan dengan mengubah beberapa parameter pada file *my.ini* yang berada pada direktori *xampp/mysql/bin*. Parameter yang dipilih untuk diubah yaitu:

1. *innodb_buffer_pool_size*
2. *innodb_log_file_size*
3. *max_connections*
4. *innodb_file_per_table*
5. *innodb_flush_log_at_trx_commit*
6. *innodb_flush_method*
7. *innodb_log_buffer_size*
8. *query_cache_size*
9. *log_bin*
10. *skip_name_resolve*

Nilai awal dari sepuluh parameter yaitu:

1. `innodb_buffer_pool_size = 16M`
2. `innodb_log_file_size = 5M`
3. `max_connections` (belum ada)
4. `innodb_file_per_table` (belum ada)
5. `innodb_flush_log_at_trx_commit = 1`
6. `innodb_flush_method` (belum ada)
7. `innodb_log_buffer_size = 8M`
8. `query_cache_size` (belum ada)
9. `log_bin` (belum ada)
10. `skip_name_resolve` (belum ada)

Kemudian, sepuluh parameter yang dipilih diganti nilainya dan ditambahkan parameter yang belum tersedia.

1. `innodb_buffer_pool_size = 5G`
2. `innodb_log_file_size = 2G`
3. `max_connections = 100`
4. `innodb_file_per_table = 1`
5. `innodb_flush_log_at_trx_commit = 0`
6. `innodb_flush_method = O_DIRECT`
7. `innodb_log_buffer_size = 16M`
8. `query_cache_size = 48M`
9. `log-bin=mysql-bin`
`server-id=1`
10. `skip_name_resolve`

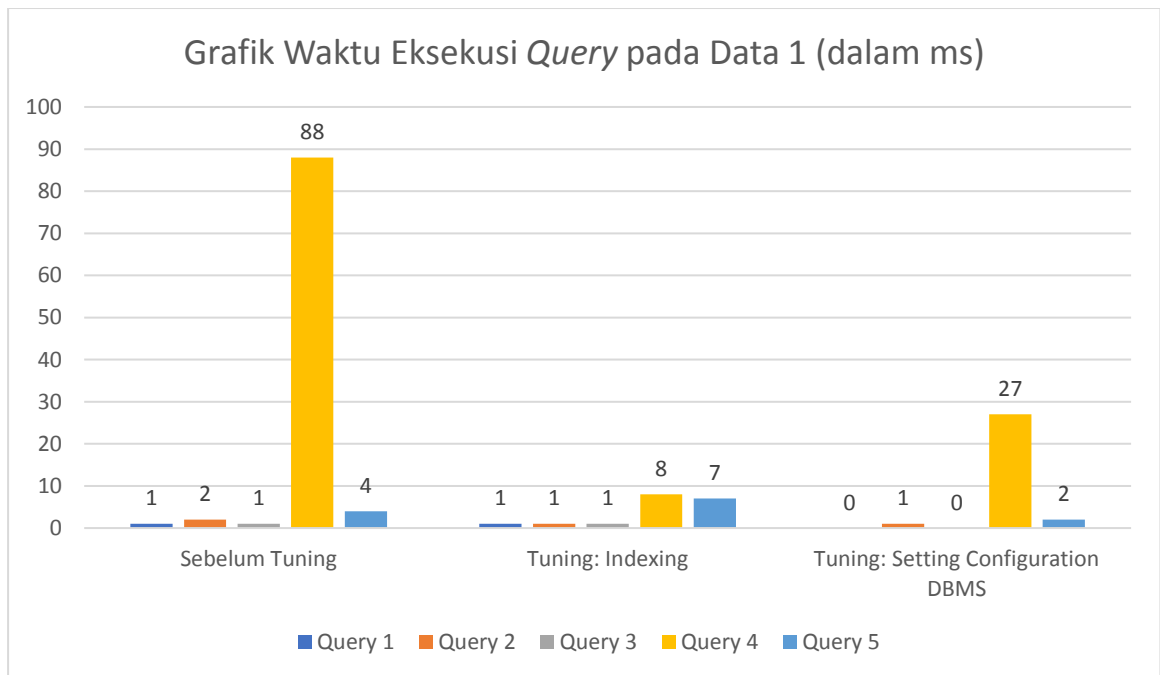
BAB III

HASIL DAN PEMBAHASAN

3.1 Hasil

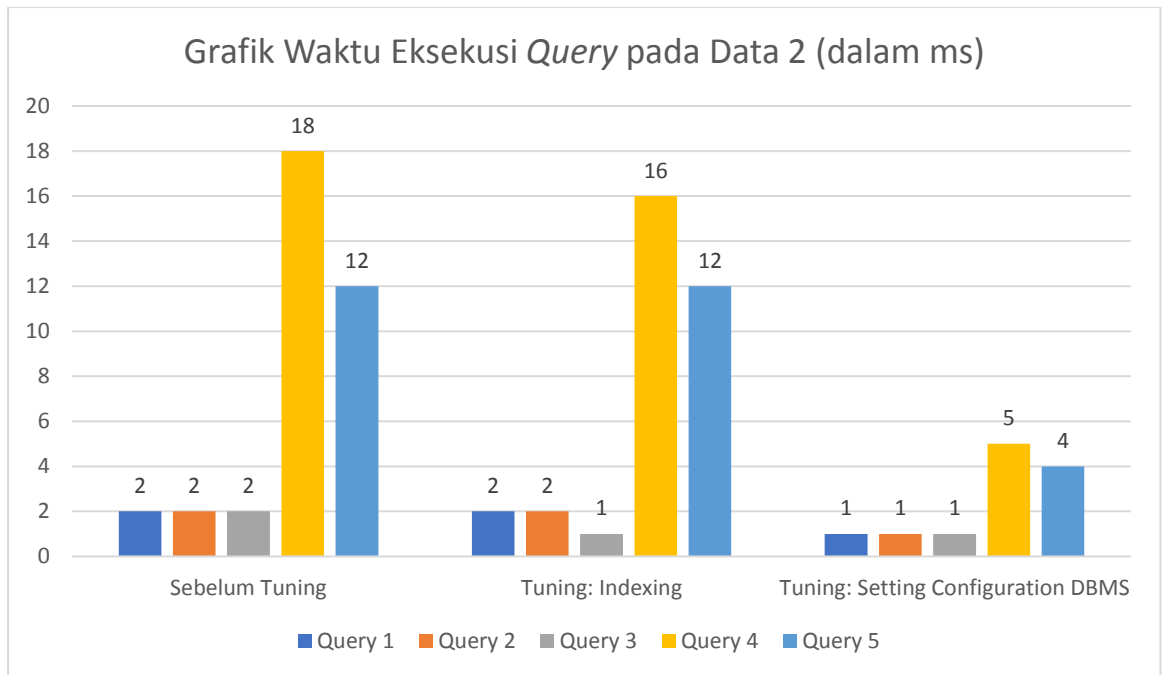
Hasil waktu eksekusi kelima *query* dari percobaan pengujian terhadap data 1, data 2, dan data 3 yang dilakukan sebelum *tuning*, setelah *tuning* menggunakan *indexing*, dan setelah *tuning* menggunakan konfigurasi DBMS yaitu:

1. Data 1 : *advisor* = 100, *student* = 100, *section* = 200, *takes* = 200



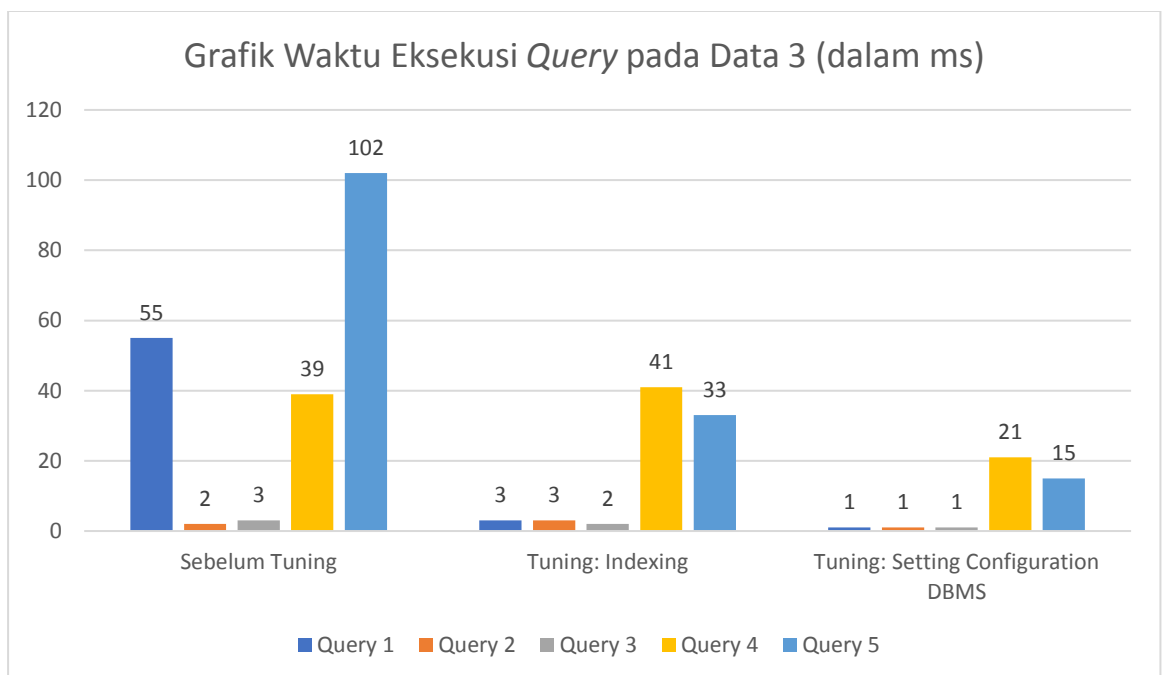
Gambar 10 Grafik Data 1

2. Data 2 : *advisor* = 200, *student* = 200, *section* = 400, *takes* = 400



Gambar 11 Grafik Data 2

3. Data 3 : *advisor* = 500, *student* = 500, *section* = 1000, *takes* = 1000



Gambar 12 Grafik Data 3

3.2 Pembahasan

Dari hasil percobaan yang dilakukan, diperoleh perubahan waktu eksekusi yang signifikan sebelum DBMS dilakukan *tuning* dan setelah DBMS dilakukan *tuning*. Rata-rata waktu eksekusi setelah *tuning* lebih cepat dibandingkan waktu eksekusi sebelum *tuning*. Dari

sebelum tuning sampai dengan *tuning* menggunakan konfigurasi DBMS, waktu eksekusi kelima *query* semakin cepat. Waktu eksekusi atau performa eksekusi dari hasil percobaan dipengaruhi oleh:

1. *Hardware* dari *laptop* yang digunakan hanya mampu mengeksekusi sampai data 3.
2. Parameter pengaturan basis data yang telah ditingkatkan.
3. Skema basis data dari setiap data yang ada semakin tinggi dan membutuhkan waktu yg semakin cepat.
4. *Indexing* pada beberapa *field* yang sering digunakan pada pemrosesan *query*.
5. *Query* yang diujikan. Semakin banyak klausa WHERE dan kondisi JOIN membutuhkan waktu yang lebih lama dan perlu dilakukan *tuning* untuk mempercepat waktu eksekusi.

Dari hasil percobaan, dapat disimpulkan bahwa beberapa *tuning* yang digunakan yaitu *indexing* dan *setting configuration DBMS* dapat mempercepat waktu dari proses eksekusi kelima *query* yang digunakan untuk menguji ketiga data.

DAFTAR PUSTAKA

- [1] F. Evangelou. [Online]. Available:
<https://gist.github.com/fevangelou/fb72f36bbe333e059b66>. [Diakses 19 Desember 2019].
- [2] I. I. Indonesia, “Inovasi Informatika Indonesia,” 7 Oktober 2016. [Online]. Available: <https://www.i-3.co.id/2016/10/07/index-pada-database/>. [Diakses 17 Desember 2019].
- [3] “MySQL,” [Online]. Available:
<https://dev.mysql.com/doc/refman/5.7/en/replication-howto-masterbaseconfig.html>. [Diakses 19 Desember 2019].
- [4] H. F. K. S. S. Abraham Silberschatz, Database System Concepts Sixth Edition, New York: MCGraw-Hill, 2011.

LAMPIRAN

1. Tautan dokumentasi gambar hasil waktu eksekusi kelima *query* dengan menggunakan terminal(CMD).

<https://drive.google.com/open?id=122F3my2MTW1jpEPQIwsOxs2z17FtpUth>