LAPORAN TUGAS BESAR MANAGEMEN BASIS DATA

TUNNING

Disusun :

Aldi Indrawan (14117055)

INSTITUT TEKNOLOGI SUMATERA 2019/2020

DAFTAR ISI

BAB I Teori Dasar

A. Tunning : indexing

1. Pengertian

index adalah sebuah objek dalam sistem database yang dapat mempercepat proses pencarian (query) data. Saat database dibuat tanpa menggunakan index, maka kinerja server database dapat menurun secara drastis. Hal ini dikarenakan resource CPU banyak digunakan untuk pencarian data atau pengaksesan query SQL dengan metode table-scan. Index membuat pencarian data akan lebih cepat dan tidak banyak menghabiskan resource CPU.
Berikut ini adalah beberapa alasan kenapa index diperlukan:

1. Kolom sering digunakan dalam klausa WHERE atau dalam kondisi join
2. Kolom berisi nilai dengan jangkauan yang luas
3. Kolom berisi banyak nilai null
4. Tabel berukuran besar dan sebagian besar query menampilkan data kurang dari 2-4%

Perlu kita perhatikan bahwa terdapat beberapa kondisi dimana tidak diperlukan kehadiran index, yaitu ketika:

1. Table kecil
2. Kolom tidak sering digunakan sebagai kondisi dalam query
3. Kebanyakan query menampilkan data lebih dari 2-4% dari seluruh data
4. Table sering di-update

2. Beberapa jenis indexing

a. B-Tree index
berguna pada saat memilih row yang sesuai dengan kriteria tertentu. Index jenis ini bisa dibuat dengan perintah CREATE INDEX.
Contoh: CREATE INDEX nama_pegawai_idx ON pegawai(nama);
bebrapa sub-tipe B-Tree :

- Index-organized tables: Pada index-organized table, rows dimasukkan kedalam index yang di definisikan pada primary key table.
- Reverse key indexes: Index yang digunakan untuk data yang sangat beragam(unik) atau increment.
  Contoh: CREATE INDEX nomor_ktp_idx ON biodata (nomor_ktp) REVERSE;
- Descending indexes: Index yang memasukan data ke dalam kolom tertentu dalam urutan menurun.
- B-tree cluster indexes: Index ini digunakan untuk mengindeks table cluster key.

b. Bitmap indexing

Index yang biasa digunakan untuk kolom yang memiliki sedikit nilai yang unik, seperti jenis kelamin, agama, atau status perkawinan. Index ini biasa digunakan untuk point ke multiple rows.

Contoh: CREATE BITMAP INDEX jenis_kelamin_idx ON biodata (jenis_kelamin);

c. Function-based indexes

Selain melakukan index terhadap kolom, seperti kolom Nama misalnya, maka Anda juga dapat meng-index suatu kolom yang berbasis fungsi, misalkan fungsi UPPER. Function-based index akan memberikan kesempatan bagi Oracle optimizer beberapa pilihan ketika memilih execution path.

Contoh: CREATE INDEX total_gaji_idx ON penggajian (gaji_pokok + bonus);

B. Tuninng : configuration DBMS

Konfiguras MySQL Server bisa dilakukan dengan mengubah file *my.ini*. file tersebut dapat ditemukan di lokasi *C:\Program Files\MySQL\MySQL Server*. Untuk lokasi yang lebih akurat, kita dapat menjalankan MySQL dari Command Prompt dan memberikan perintah `mysqld --help --verbose | more`.

Salah satu pengaturan yang paling umum adalah mengubah nilai `innodb_buffer_pool_size`. Ini adalah besarnya wilayah di memori yang dialokasikan khusus untuk menampung cache yang berisi informasi tabel dan index. Semakin besar ukuran *buffer pool*, maka semakin sedikit operasi disk yang dibutuhkan.Dokumentasi MySQL Server merekomendasikan nilai 80% dari jumlah memori untuk database yang sibuk. Jangan lupa bahwa nilai i`nnodb_buffer_pool_size` yang terlalu besar malah bisa mengakibatkan perlambatan akibat *paging*.

BAB II Deskripsi percobaan

A. Data

Data
advisor = 100, student = 100, section = 200,takes = 200
advisor = 200, student = 200, section = 400,takes = 400
advisor = 500, student = 500, section = 1000,takes = 1000

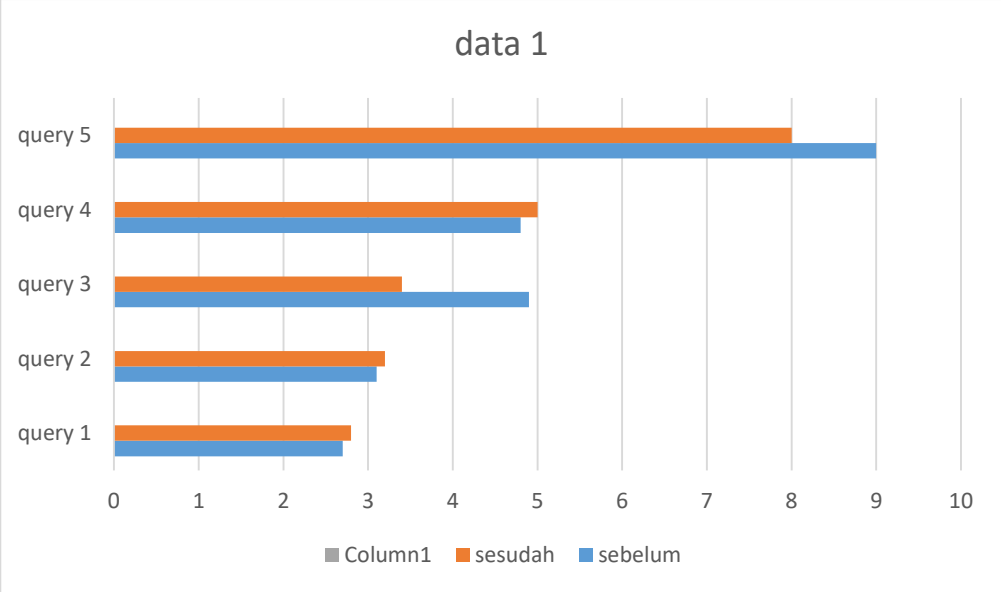
B. Tunning:indexing

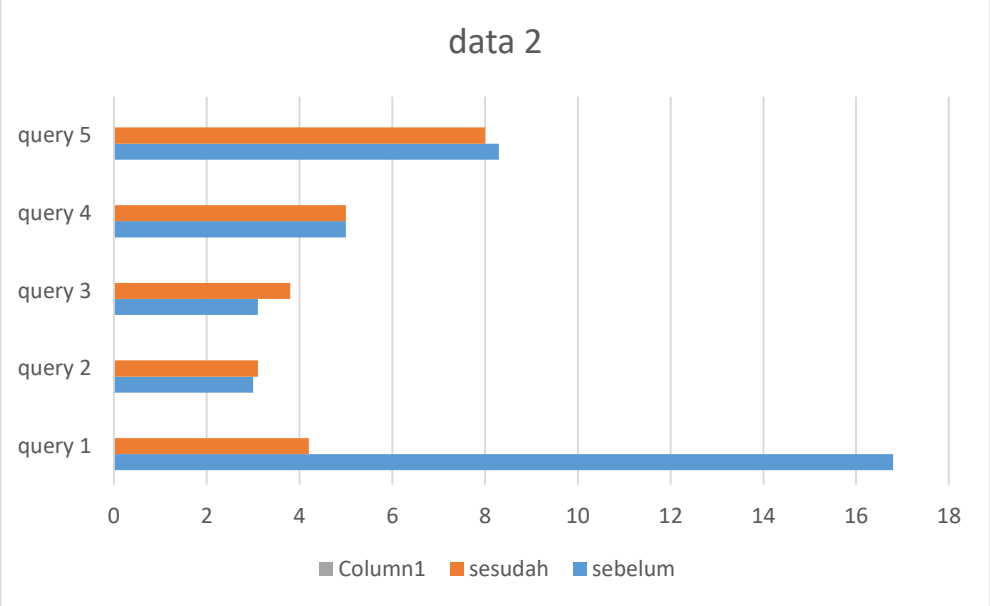
Pada indexing saya melakukan index pada table section dengan index colom cource_id dan mengindex pada table student dengan kolom ID dan didapat data sebagai berikut

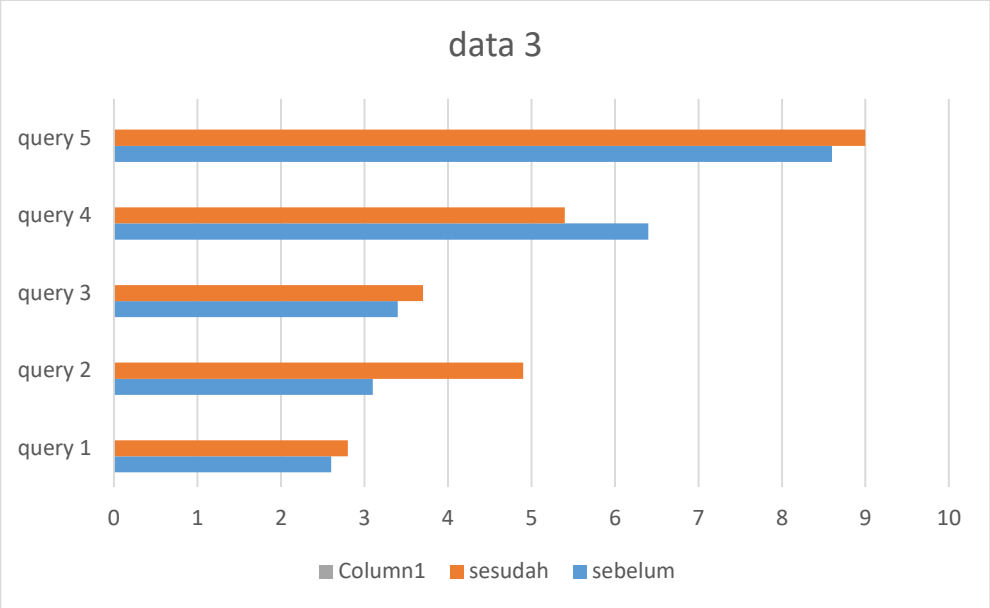
Waktu Sebelum Tunning (ms)				
query 1	query 2	query 3	query 4	query 5
0.0027	0.0031	0.0049	0.0048	0.09

0.0168	0.0030	0.0031	0.0050	0.0083
0.0026	0.0031	0.0034	0.0064	0.0086

Waktu Sesudah Tunning (ms) (indexing)				
query 1	query 2	query 3	query 4	query 5
0.0028	0.0032	0.0034	0.0050	0.008
0.0042	0.0031	0.0038	0.0050	0.0087
0.0028	0.0049	0.0037	0.0054	0.009







5. Pada ketiga data yang diuji beberapa query yang dijalankan menjadi lebih lambat bukan menjadi cepat karena ada kesalahkan saat dilakukan indexing,saya melakukan index pada data student dimana data tersebut memiliki record kecil dan Kolom tidak sering digunakan sebagai kondisi dalam query sehingga menyebabkan query menjadi lebih lambat

C. Tunning : configuration DBMS

Untuk configuration DBMS saya mengubah konfigurasi innodb_log_ menjadi 512MB yang tadinya 256BM.

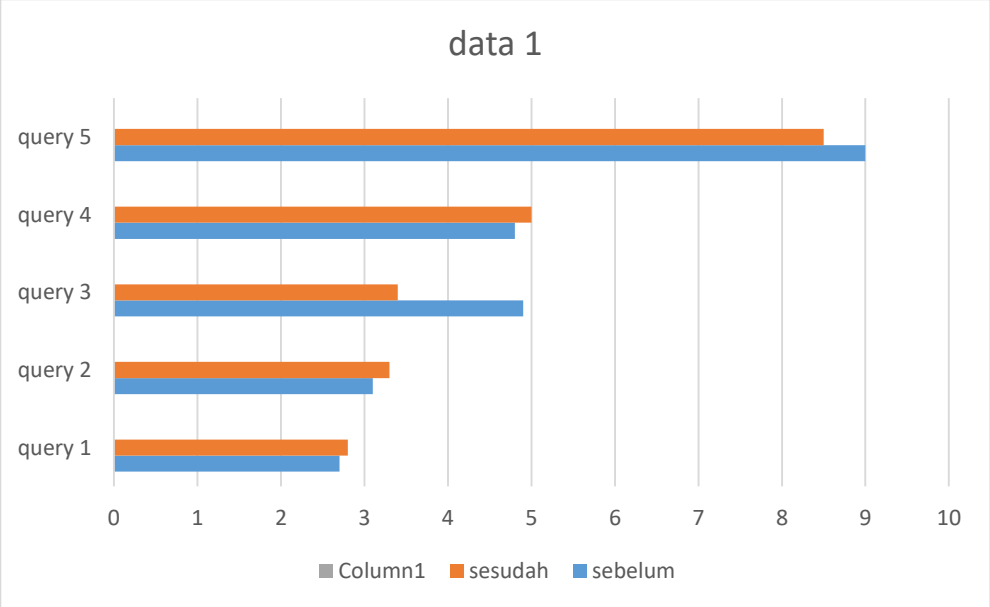
```
# Size of each log file in a log group. You should set the combined size
# of log files to about 25%-100% of your buffer pool size to avoid
# unneeded buffer pool flush activity on log file overwrite. However,
# note that a larger logfile size will increase the time needed for the
# recovery process.
innodb_log_file_size = 512M

# Total number of files in the log group. A value of 2-3 is usually good
# enough.
innodb_log_files_in_group = 3

# Location of the InnoDB log files. Default is the MariaDB datadir. You
# may wish to point it to a dedicated hard drive or a RAID1 volume for
# improved performance
#innodb_log_group_home_dir

# Maximum allowed percentage of dirty pages in the InnoDB buffer pool.
# If it is reached, InnoDB will start flushing them out aggressively to
# not run out of clean pages at all. This is a soft limit, not
# guaranteed to be held.
innodb_max_dirty_pages_pct = 90
```

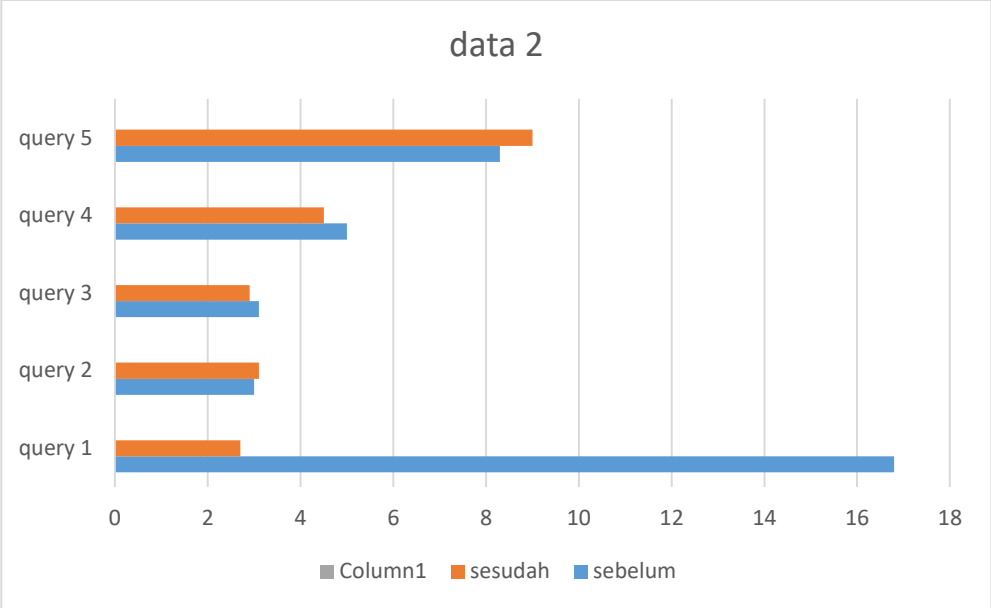
Setelah disave restart ulang server apache pada xampp. Dan lakukan query yang diinginkakn.didapat data sebagai berikut

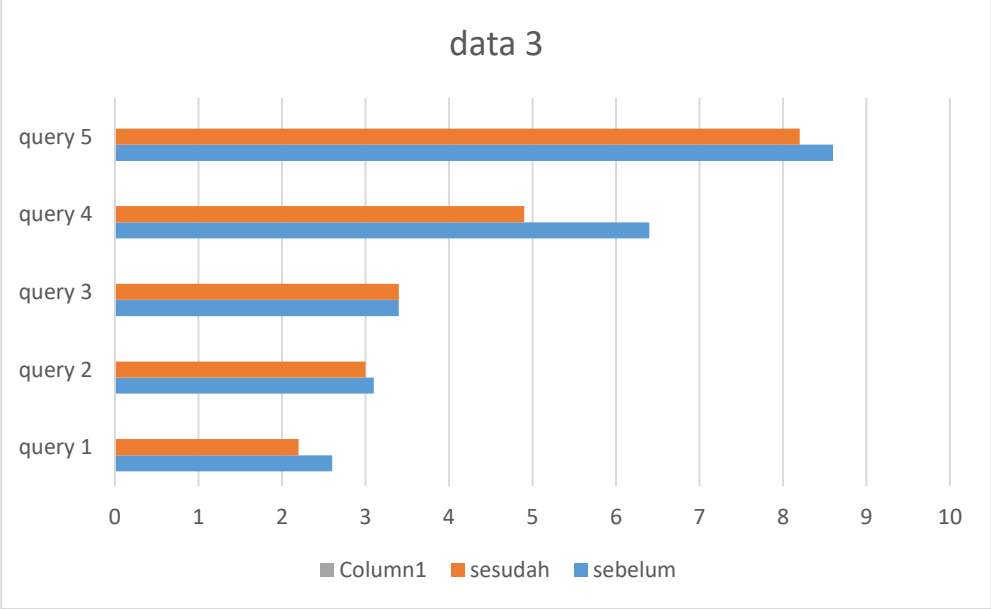
Waktu Sebelum Tunning (ms)				
query 1	query 2	query 3	query 4	query 5

0.0027	0.0031	0.0049	0.0048	0.09
0.0168	0.0030	0.0031	0.0050	0.0083
0.0026	0.0031	0.0034	0.0064	0.0086

Waktu Sesudah Tunning (ms) (configurasi DBMS)				
query 1	query 2	query 3	query 4	query 5
0.0028	0.0033	0.0034	0.0050	0.0085
0.0027	0.0031	0.0029	0.0045	0.0079
0.0022	0.0030	0.0034	0.0049	0.0082







Pada 3 data yang diuji dengan data yang ada setelah dilakukan configuration DBMS terdapat penigkatan waktu eksekusi pada beberapa query menjadi lebih cepat,karena kita menambah innodb_log_file_size menjadi lebih besar sehingga mengurangi aktifitas flush.

Catatan :

1. SELECT * FROM student
2. SELECT * FROM student WHERE tot_cred > 30;
3. SELECT `name`, department 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
```

Daftar pustaka

1. https://www.i-3.co.id/2016/10/07/index-pada-database/
2. https://thesolidsnake.wordpress.com/2014/12/28/belajar-menyetel-database-mysql-server/