

Perbandingan Kecepatan Baca Dan Tulis Data Pada Mysql Menggunakan Primary Key Auto Increment Dengan Universally Unique Identifier (UUID)

Juri Pebrianto
Universitas Pamulang
juripebrianto@gmail.com

Abstract

Technological advances at this time make data growth continue to increase and become the current trend in data processing called "big data" where a lot of data is gathered for later processing and making decisions. In its development, application developers use database software that can accommodate the data generated by its users. In more detail, in the database row, it is necessary to have a key that can distinguish between one row and another, which is usually called the primary key. Giving a value to the primary key is important because it will have an impact on the speed of execution of the data line, both written and read. Many software makers use auto-increments or universally unique identifiers. The addition of hardware specifications would be another option in order to keep running applications that have big data well, but hardware changes are also a problem if it is done. The primary key using auto-increment has a maximum data of 9,223,372,036,854,775,807 which if the data continues to grow it is certain that a malfunction will occur from the recording, UUIDs are present for this solution because they are random and have the same length and have no limit in their number which in insert/add data activity when compared to auto-increment is a better performance using the MPE Method which is able to determine the priority order of alternative decisions using several criteria..

Keywords: Big data, primary key, auto-increment, UUID, MPE.

Abstrak

Kemajuan teknologi pada saat ini membuat pertumbuhan data terus meningkat dan menjadi tren saat ini dalam pengolahan data yang disebut "big data" dimana data yang banyak berkumpul untuk nantinya diolah dan dijadikan sebuah keputusan-keputusan. Dalam perkembangannya, para pengembang aplikasi menggunakan perangkat lunak basis data yang dapat menampung data-data yang dihasilkan oleh para penggunanya. Lebih detail lagi, dalam baris basis data perlu adanya kunci yang dapat membedakan antara baris satu dan lainnya yang biasa disebut primary key. Pemberian nilai pada primary key menjadi hal penting karena akan berdampak pada kecepatan eksekusi baris data tersebut, baik tulis dan baca. Para pembuat perangkat lunak banyak yang menggunakan auto-increment atau pun Universally unique identifier. Penambahan spesifikasi perangkat keras akan menjadi pilihan lainnya agar dapat tetap menjalankan aplikasi yang memiliki data besar dengan baik, namun perubahan perangkat keras juga menjadi masalah jika itu dilakukan. Pada primary key menggunakan auto increment memiliki maksimal data sebanyak 9.223.372.036.854.775.807 yang jika data terus bertambah sudah pasti akan terjadi malfungsi dari pencatatannya, UUID hadir untuk solusi tersebut karena sifatnya yang acak dan memiliki panjang yang sama serta tidak memiliki batasan dalam jumlahnya yang pada aktivitas insert/tambah data jika dibandingkan dengan auto-increment lebih baik performanya menggunakan Metode MPE yang mampu untuk menentukan urutan prioritas alternatif keputusan dengan menggunakan beberapa kriteria.

Kata kunci: Big data, primary key, auto-increment, UUID, MPE.

I. PENDAHULUAN

Pengguna teknologi yang semakin hari kian bertambah, baik pengguna telepon genggam, komputer dan alat lain yang terhubung ke internet atau istilah dalam teknologi disebut Internet of Things. Internet digunakan disemua lini dan juga barang. Sekarang kita dapat menemukan lampu yang dapat dihidup-matikan melalui telepon genggam, mobil yang dapat berjalan sendiri dan barang lainnya yang semakin hari semakin pintar dengan bantuan internet.

Hal tersebut mendorong para penggiat perangkat lunak untuk terus berinovasi dalam karya aplikasi yang dibuat. Hal itu pula yang membuat para pengguna aplikasi

menjadi banyak yang menggunakan. Salah satu jenis aplikasi yang sering digunakan adalah social media. Setiap harinya ada jutaan orang mengunggah gambar, video dan tulisan kedalam social media. Bisa dibayangkan berapa besar penyimpanan yang harus digunakan oleh perusahaan penyedia social media untuk menyimpan data para penggunanya.

Peranan data sangat penting terutama memasuki era ledakan data atau "Big Data". Oleh karenanya, pihak yang mampu mengolah dan memanfaatkan data-data yang sangat besar, cepat, variatif, dan kompleks, dapat mengambil keuntungan yang besar (Emyana Ruth Eritha Sirait, 2016).

Data adalah hal penting pada era digital seperti sekarang ini. Data menjadikan setiap keputusan dalam

analisa dapat mendekati kesesuaian, namun untuk mendapatkan keputusan tersebut dibutuhkan banyak data. Masalah lain soal bagaimana data dapat disimpan dan diolah muncul karena banyaknya data yang ada. Perlu adanya suatu rancangan database yang dapat mengakomodir barisan data tersebut. Dalam perangkat lunak basis data semua baris yang ada harus memiliki identitas yang satu dan lainnya berbeda.

Para penggiat perangkat lunak saat ini diberi kemudahan terhadap apa-apa yang dapat membantu dalam pembuatan aplikasi baik dari Bahasa pemrograman maupun perangkat lunak basis data. Konsentrasi pada saat pembuatan aplikasi perangkat lunak adalah basis data. Bagaimana menggunakannya, fitur apa yang diberikan, seberapa cepat membaca data dan hal lainnya menjadi pertimbangan saat membuat atau merancang basis data.

Ada banyak hal yang perlu diperhatikan ketika para pembuat perangkat lunak ingin menggunakan basis data sebagai media penyimpanan datanya. Apa saja yang harus dimasuka sebagai entiti, berapa panjang data yang akan disimpan, tipe data apa yang akan disimpan dan yang terpenting adalah *primary key* sebagai kunci dari baris data yang disimpan kedalam basis data.

Primary key adalah satu atribut yang bukan hanya mengidentifikasi secara unik suatu kejadian spesifik, tetapi juga dapat mewakili setiap kejadian dari suatu entitas (J. I. Maanari, 2013). Pemberian nilai pada *primary key* sangat lah penting karena sangat berpengaruh kepada kecepatan tulis dan baca data, jika data masih terbilang kecil mungkin tidak terlalu berdampak aktivitas tulis dan baca, namun jika data sudah memiliki baris yang sangat banyak maka akan sangat terasa sekali waktu tunggu untuk sampai data dieksekusi.

Pada pemberian nilai untuk *primary key*, penggunaan *auto-increment* lebih sering karena sifatnya yang otomatis memberikan susunan angka setiap baris yang diinput menjadikan semua baris data berbeda atau unik. Namun juga banyak yang belum mengetahui pemberian nilai pada *primary key* menggunakan *Universally unique identifier* (UUID) dimana UUID memberikan nilai berbeda atau unik setiap baris yang diinput berupa digit acak, gabungan antara huruf dan angka yang jumlah digitnya selalu sama (tidak berurut).

UUID memang memiliki kriteria panjang digit dan formatnya namun pada data umum yang sering kita jumpai, UUID juga sering digunakan namun dalam bentuk yang berbeda seperti ID karyawan, nomor KTP, nomor induk siswa, nomor induk mahasiswa, kode barang dan lainnya yang memiliki sifat berbeda dari setiap pemilik atau setiap surat yang dilabeli dengan nomor tersebut.

Auto-increment yang sangat sering digunakan memiliki kelemahan dengan strukturnya yang mudah ditebak, ini menyebabkan data yang di ekspos menggunakan digit *auto-increment* menjadi mudah disalahgunakan. Sebagai contoh, jika pada kolom alamat di browser misal terdapat alamat url seperti ini www.unpam.ac.id?id=1, dari url ini kita dapat melihat bagaimana bentuk atau struktur dari kunci basis data yang digunakan. Jika disalahgunakan, kelemahan id dengan data terurut ini menjadi celah dalam sebuah website.

Sedangkan UUID dengan formatnya yang acak dan sifat dari panjang digitnya yang statis maka, hal-hal yang

tidak diinginkan dari ekspos data menggunakan UUID bisa diminimalkan, contoh, www.unpam.ac.id?id=jd773hd7w-wydsdgsd-js, dari url ini dapat dilihat strukturnya tidak mudah ditebak, karena hal ini penggunaan UUID sebagai kunci menjadi lebih baik, namun apakah performa dari UUID lebih cepat dari *auto-increment*? hal ini lah yang perlu diteliti lebih lanjut.

Metode Perbandingan Eksponensial (MPE) adalah salah satu metode dari Sistem Pendukung Keputusan (SPK) yang digunakan untuk menentukan urutan prioritas alternatif keputusan dengan multi kriteria. MPE sangat cocok untuk penilaian skala ordinal (contoh sangat baik, baik, kurang, sangat kurang). Metode perbandingan eksponensial mempunyai keuntungan dalam mengurangi bias yang mungkin terjadi dalam analisis. Nilai skor yang menggambarkan urutan prioritas menjadi besar (fungsi eksponensial) ini mengakibatkan urutan prioritas alternatif keputusan lebih nyata (Marimin, 2011).

Metode MPE ini mampu untuk menentukan urutan prioritas alternatif keputusan dengan menggunakan beberapa kriteria. Metode ini mampu mengurangi bias yang mungkin terjadi dalam analisis. Untuk nilai skor yang dihasilkan, akan menggambarkan urutan prioritas yang menjadi besar, ini mengakibatkan urutan prioritas alternatif keputusan menjadi lebih nyata. Selain itu metode ini merupakan salah satu metode pengambilan keputusan yang mengkuantifikasikan pendapat seseorang atau lebih dalam skala tertentu. Pada prinsipnya ia merupakan metode skoring terhadap pilihan yang ada (Rangkuti, 2011).

Metode perbandingan digunakan pada saat pengujian yang berdampak dan menjadi parameter itu sendiri seperti penggunaan memori, prosesor dan harddisk yang pada sistem operasi dapat dilihat indikatornya pada task manager, bagaimana dampak terhadap performa perangkat keras yang disebutkan diatas. Hasil dari perbandingan tersebut menjadi sebuah acuan bagaimana menggunakan *auto-increment* dan UUID terhadap aktivitas baca dan tulis basis data untuk pengentasan *primary key*. Dalam perhitungan skor, formulasi untuk setiap alternatif pada metode MPE adalah:

$$Total\ Nilai\ (TN_i) = \sum_{j=1}^m (RK_{ij})^{TKK_j}$$

Tabel 1: Informasi keterangan dari rumus MPE

Total Nilai i	total nilai akhir dari alternatif ke –i
RK _{ij}	derajat kepentingan kriteria relatif ke-j pada pilihan keputusan i
TKK _j	derajat kepentingan kriteria relatif ke-j TKK _j > 0
N	jumlah pilihan keputusan
M	jumlah kriteria keputusan

Auto-increment didapat dari fungsi bawaan dari perangkat lunak basis data yang akan memberikan nomor urut pada setiap barisnya dan UUID didapat dari hasil

generate script PHP. Saat ini penulis berprofesi sebagai pembuat aplikasi perangkat lunak, penggunaan *auto-increment* atau *UUID* sudah pernah digunakan dan diimplementasikan kedalam aplikasi yang dibuat. Dari beberapa penjelasan diatas, sekilas penulis dapat membandingkan dari beberapa aspek, namun yang paling penting adalah kita dapat mengetahui bagaimana performa kecepatan baca dan tulis dengan menggunakan *auto-increment* dan *UUID* karena sampai saat ini penulis belum menemukan bagaimana performa kedua metode tersebut yang dapat memberikan pandangan terhadap penggunaan metode tersebut. Keputusan yang harus diambil dalam pembuatan aplikasi perangkat lunak yang paling penting adalah bagaimana manajemen basis data dan bagaimana memaksimalkannya oleh karena itu penulis ingin melakukan penelitian tentang Perbandingan Kecepatan Baca Dan Tulis Data Pada Basis Data Mysql Dengan *Primary Key* Menggunakan *auto-increment* dan *Universally Unique Identifier (UUID)*.

II. METODE PENELITIAN

Metode pengumpulan data merupakan bagian yang sangat penting dalam setiap kegiatan penelitian. Hal tersebut dilakukan untuk mendapatkan data yang akurat, terperinci, dan dapat dipercaya serta dapat dipertanggung jawabkan. Maka metode pengumpulan data harus tepat agar sesuai data yang diperlukan, di dalam penelitian ini maka diperlukan beberapa teknik pengumpulan data diantaranya yaitu:

1. Studi Pustaka. Studi pustaka diterapkan dengan melakukan penelaahan terhadap Jurnal, Buku dan Skripsi yang berupa informasi atau referensi perancangan sistem informasi media pembelajaran Teknik Komputer dan Jaringan berbasis android.
2. Observasi. Observasi dilakukan selama proses penelitian dengan melakukan pengamatan secara langsung terhadap objek dan aktivitas, dan dengan dilakukan observasi untuk memperoleh data dan informasi mengenai sistem yang akan dikembangkan secara detail dan akurat. Selain itu, melalui observasi juga dapat memperoleh gambaran langsung terhadap alur kerja sistem atau aktivitas sistem yang sedang berjalan secara jelas.

Oleh karena itu, untuk mendukung proses pengujian dalam penelitian ini, dibutuhkan beberapa requirement sebagai berikut:

1. Prosesor Core i5
2. RAM 8gb
3. Storage SSD 256 gigabyte
4. Sistem Operasi Windows 10
5. PHP Versi 7
6. MySql Version 5.7
7. Sublime Text Editor
8. *Command Prompt / Terminal*

Penelitian ini akan melakukan penulisan data terhadap perangkat lunak basis data MySql secara berkala sampai jumlah yang dirasa dapat menjadi sampel atau perwakilan dari bahan uji penelitian. Sebagai contoh, data akan dimasukkan 10.000 baris lalu akan dilihat dampak terhadap

perangkat keras memori, storage dan prosesor dan waktu yang dibutuhkan dalam memasukan data.

Data kembali di input dengan penambahan jumlah data yang disimpan akan semakin besar yaitu 50.000 data atau baris. Begitupun pada aktivitas baca data, skrip akan dibuat untuk membaca data sama jumlahnya seperti pada saat memasukan data, contoh, jika data yang dimasukan adalah 10.000 maka pada skrip baca data juga akan diberikan 10.000 data.

Percobaan juga dilakukan beberapa kali guna memastikan hasilnya dapat dipertanggung jawabkan untuk kelak bisa menjadi sumber informasi bagi masyarakat yang ingin merancang basis data ataupun membuat aplikasi atau perangkat lunak. Hasil dari penelitian ini juga diharapkan dapat membantu para analisator basis data untuk bisa memilih perangkat keras yang digunakan jika ingin membangun sebuah server yang akan digunakan sebagai pusat data dari aplikasi yang dibuat.

Insert Data 10.000 Baris Primary key Auto-increment

Pemilihan database merupakan bagian yang sangat penting, karena database tersebut nantinya harus mampu mengelola data dalam jumlah yang sangat besar sehingga performa database tetap terjaga (Hendra & Andriyani, 2020). MySQL adalah sebuah database management system (manajemen basis data) menggunakan perintah dasar SQL (Structured Query Language).

Database management system (DBMS) MySQL multi pengguna dan multi alur ini sudah dipakai lebih dari jutaan pengguna di seluruh dunia. Pada tahun 2000, MySQL dirilis dengan lisensi ganda yang mengizinkan public untuk menggunakannya secara gratis dibawah lisensi GNU GPL (General Public License) yang menyebabkan popularitasnya melambung. MySQL mampu menangani puluhan ribu tabel dan miliaran baris data dengan cepat dan lancar (Silalahi & Wahyudi, 2018).

Aplikasi perangkat lunak yang akan digunakan dalam penelitian ini adalah MySQL, karena penulis sudah familiar dan sering menggunakannya. Database yang selama ini banyak dipakai adalah MySQL (Junaidi, 2016).



Gambar 1: Koneksi MySql menggunakan PHP

Pada gambar diatas (Gambar 1), penulis menuliskan skrip PHP untuk melakukan koneksi ke perangkat lunak basis data MySql.



	id	waktu
1	1	2022-01-26 04:23:17
2	2	2022-01-26 04:23:17
3	3	2022-01-26 04:23:17
4	4	2022-01-26 04:23:17
5	5	2022-01-26 04:23:17

Gambar 2: Bentuk data dari *auto-increment*

Pada gambar diatas (Gambar 2) menunjukan bentuk dari pemberian *primary key* (dengan entiti "id") menggunakan *auto-increment* yang dimana pemberian primary dilakukan oleh MySQL itu sendiri dan bentuknya yang berurut.

```
<?php
include 'database.php';
echo date('Y-m-d H:i:s')."\n";
for ($i = 1; $i <= 10000; $i++) {
    $sql = "INSERT INTO auto(waktu) VALUES('".date('Y-m-d H:i:s')."')";
    mysql_query($koneksi, $sql);
}
echo date('Y-m-d H:i:s')."\n";
?>
```

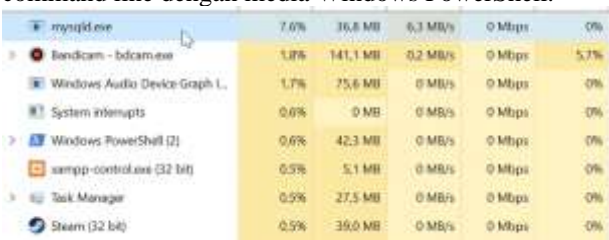
Gambar 3: Skrip PHP melakukan insert data ke MySQL sebanyak 10.000 data dengan *primary key auto-increment*

Pada gambar diatas (Gambar 3) adalah skrip pemrograman PHP untuk melakukan penambahan (insert) menggunakan *primary key auto-increment* dengan data sebanyak 10.000 data menggunakan fungsi for loop dan disimpan dengan nama auto.php.

```
PS C:\xampp\htdocs\penelitian> php auto.php
2022-01-26 04:23:17
2022-01-26 04:23:36
PS C:\xampp\htdocs\penelitian>
```

Gambar 4: Hasil eksekusi dari *query insert* 10.000 data menggunakan *primary key auto-increment*

Pada gambar diatas (Gambar 4), script auto.php yang sudah dibuat pada gambar 3 dijalankan menggunakan command line dengan media Windows PowerShell.



Process	CPU	Private	Working Set	IO	Network
mysql.exe	7.6%	36.8 MB	6.3 MB/s	0 MBps	0%
Bandicam - Bandicam.exe	1.0%	141.1 MB	0.2 MB/s	0 MBps	5.7%
Windows Audio Device Graph L...	1.7%	75.6 MB	0 MB/s	0 MBps	0%
System Interrupts	0.6%	0 MB	0 MB/s	0 MBps	0%
Windows PowerShell (2)	0.6%	42.3 MB	0 MB/s	0 MBps	0%
xampp-control.exe (32 bit)	0.5%	5.1 MB	0 MB/s	0 MBps	0%
Task Manager	0.5%	27.5 MB	0 MB/s	0 MBps	0%
Steam (32 bit)	0.5%	39.0 MB	0 MB/s	0 MBps	0%

Gambar 5: Monitoring penggunaan *resource* menggunakan task manager dari eksekusi dari *query insert* 10.000 data menggunakan *primary key auto-increment*

Pengujian dilakukan pada basis data MySQL menggunakan bahasa pemrograman PHP dengan menambahkan data sebanyak 10.000 baris menggunakan *primary key auto-increment* mendapatkan hasil sebagai berikut:

Tabel 2: Hasil *query insert* / tambah data MySQL menggunakan *primary key auto-increment* sebanyak 10.000 baris data

Waktu	Prosesor	Memori	Ukuran / Storage
19 Detik	7.6 %	36.8 MB	528 Kb

Insert Data 50.000 Baris *Primary key Auto-increment*

```
<?php
include 'database.php';
echo date('Y-m-d H:i:s')."\n";
for ($i = 1; $i <= 50000; $i++) {
    $sql = "INSERT INTO auto(waktu) VALUES('".date('Y-m-d H:i:s')."')";
    mysql_query($koneksi, $sql);
}
echo date('Y-m-d H:i:s')."\n";
?>
```

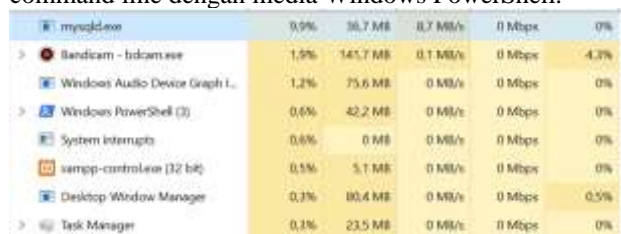
Gambar 6: Skrip PHP melakukan insert data ke MySQL sebanyak 50.000 data dengan *primary key auto-increment*

Pada gambar diatas (Gambar 6) adalah skrip pemrograman PHP dari gambar 3 dan diubah banyaknya perulangan / loop untuk melakukan penambahan (insert) menggunakan *primary key auto-increment* dengan data sebanyak 50.000.

```
Windows PowerShell
PS C:\xampp\htdocs\penelitian> php auto.php
2022-01-26 04:35:38
2022-01-26 04:37:14
```

Gambar 7: Hasil eksekusi dari *query insert* 50.000 data menggunakan *primary key auto-increment*

Pada gambar diatas (Gambar 7), script auto.php yang sudah dibuat pada gambar 6 dijalankan menggunakan command line dengan media Windows PowerShell.



Process	CPU	Private	Working Set	IO	Network
mysql.exe	9.9%	36.7 MB	8.7 MB/s	0 MBps	0%
Bandicam - Bandicam.exe	1.5%	145.7 MB	0.1 MB/s	0 MBps	4.3%
Windows Audio Device Graph L...	1.2%	75.6 MB	0 MB/s	0 MBps	0%
Windows PowerShell (2)	0.6%	42.2 MB	0 MB/s	0 MBps	0%
System Interrupts	0.6%	0 MB	0 MB/s	0 MBps	0%
xampp-control.exe (32 bit)	0.5%	5.1 MB	0 MB/s	0 MBps	0%
Desktop Window Manager	0.3%	80.4 MB	0 MB/s	0 MBps	0.5%
Task Manager	0.3%	23.5 MB	0 MB/s	0 MBps	0%

Gambar 8: Monitoring penggunaan *resource* menggunakan task manager dari eksekusi dari *query insert* 50.000 data menggunakan *primary key auto-increment*

Pengujian dilakukan pada basis data MySQL menggunakan bahasa pemrograman PHP dengan menambahkan data sebanyak 50.000 baris menggunakan *primary key auto-increment* mendapatkan hasil sebagai berikut:

Tabel 3: Hasil *query insert* / tambah data MySQL menggunakan *primary key auto-increment* sebanyak 50.000 baris data

Waktu	Prosesor	Memori	Ukuran / Storage
148 Detik	9.9 %	36.8 MB	2576 Kb

Insert Data 10.000 Baris Primary key UUID

UUID diperoleh dengan melakukan digit acak dan umumnya dilakukan oleh bahasa pemrograman yang digunakan. Dibawah ini adalah script *function* untuk melakukan *generate* UUID menggunakan bahasa pemrograman PHP.



Gambar 9: Kode PHP untuk melakukan generate UUID.

Pada skrip PHP di digambar 9 diatas menggunakan fungsi *mt_rand()* untuk mendapatkan angka acak dengan rentang angka yang dihasilkan akan dimulai dari angka 1 sampai dengan 2147483647 serta alphabet dari a sampai z.

		▼ id		waktu	
				00019640-4852-4671-9d3e-e1fcd9bc828	2022-01-26 05:18:04
				00037503-7a97-4448-9726-005ee5329dd	2022-01-26 05:28:17
				00085748-7827-4c9a-ba0a-5e428e19002f	2022-01-26 05:18:18
				00093734-8e58-84db-816d-eeeb9f1c1a12	2022-01-26 05:28:12
				00094842-4a39-48b6-0f63-30c3ba5ab210	2022-01-26 05:27:17
				00095d48-00c7-4000-88b0-0e0b1f9d9ad	2022-01-26 05:18:00
				000a96e3-d0d5-4406-81a8-8196364caee4	2022-01-26 05:26:28

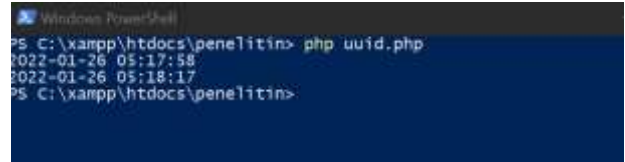
Gambar 10: Bentuk data dari UUID.

Pada gambar diatas (Gambar 10) menunjukan bentuk dari pemberian *primary key* (dengan entiti "id") menggunakan UUID yang dimana pemberian primary dilakukan oleh bahasa pemrograman PHP pada MySQL dari hasil generate script digambar 9.



Gambar 11: Skrip PHP melakukan insert data ke MySQL sebanyak 10.000 data dengan *primary key* UUID

Pada gambar diatas (Gambar 11) adalah skrip pemrograman PHP untuk melakukan penambahan (insert) menggunakan *primary key* UUID dengan data sebanyak 10.000 data menggunakan fungsi *for loop* dan disimpan dengan nama *uuid.php*.



Gambar 12: Hasil eksekusi dari query insert 10.000 data menggunakan *primary key* UUID

Pada gambar diatas (Gambar 12), script *uuid.php* yang sudah dibuat pada gambar 11 dijalankan menggunakan command line dengan media Windows PowerShell.

mysqld.exe	9.5%	11.4 MB	6.1 MB/s	0 Mbps	0%
Bandicam - bdcam.exe	1.8%	140.8 MB	0.1 MB/s	0 Mbps	7.0%
Windows PowerShell (3)	1.1%	25.0 MB	0 MB/s	0 Mbps	0%
Windows Audio Device Graph L...	0.9%	70.2 MB	0 MB/s	0 Mbps	0%
System Interrupts	0.7%	0 MB	0 MB/s	0 Mbps	0%
xampp-control.exe (32 bit)	0.5%	1.5 MB	0 MB/s	0 Mbps	0%
Desktop Window Manager	0.4%	56.2 MB	0 MB/s	0 Mbps	0.7%
Steam (32 bit)	0.3%	9.5 MB	0 MB/s	0 Mbps	0%

Gambar 13: Monitoring penggunaan *resource* menggunakan *task manager* dari eksekusi dari query insert 10.000 data menggunakan *primary key* UUID

Pengujian dilakukan pada basis data MySQL menggunakan bahasa pemrograman PHP dengan menambahkan data sebanyak 10.000 baris menggunakan *primary key* UUID mendapatkan hasil sebagai berikut:

Tabel 4: Hasil query insert / tambah data MySQL menggunakan *primary key* UUID sebanyak 10.000 baris data

Waktu	Prosesor	Memori	Ukuran / Storage
101 Detik	9.5 %	13.4 MB	1552 Kb

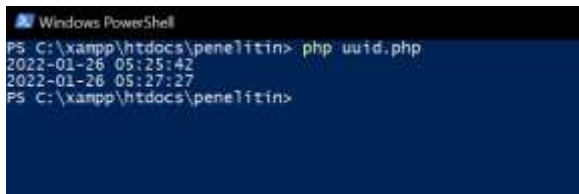
Insert Data 50.000 Baris Primary key UUID



Gambar 14: Skrip PHP melakukan insert data ke MySQL sebanyak 50.000 data dengan *primary key* UUID

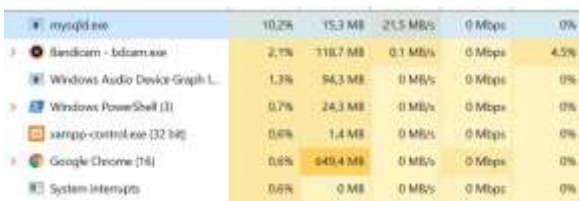
Pada gambar diatas (Gambar 14) adalah skrip pemrograman PHP dari gambar 11 dan diubah banyaknya perulangan / loop untuk melakukan penambahan (insert)

menggunakan *primary key* UUID dengan data sebanyak 50.000.



Gambar 15: Hasil eksekusi dari *query insert* 50.000 data menggunakan *primary key* UUID

Pada gambar diatas (Gambar 15), script *auto.php* yang sudah dibuat pada gambar 11 dijalankan menggunakan command line dengan media Windows PowerShell.



Gambar 16: Monitoring penggunaan *resource* menggunakan *task manager* dari eksekusi dari *query insert* 50.000 data menggunakan *primary key* UUID

Pengujian dilakukan pada basis data MySQL menggunakan bahasa pemrograman PHP dengan menambahkan data sebanyak 50.000 baris menggunakan *primary key* UUID mendapatkan hasil sebagai berikut:

Tabel 5: Hasil *query insert* / tambah data MySQL menggunakan *primary key* UUID sebanyak 50.000 baris data

Waktu	Prosesor	Memori	Ukuran / Storage
135 Detik	10.2 %	15.3 MB	6672 Kb

Pembacaan Data

Setelah dilakukan proses input atau insert data maka selanjutnya adalah bagaimana data tersebut akan dibaca, peneliti menemukan adanya hal yang tidak signifikan untuk dilakukan perbandingan perhitungan dalam kecepatan baca data baik data yang menggunakan *primary key auto-increment* dan UUID, kecepatan baca data tidak sampai 1 detik, gambar berikut adalah hasil dari pembacaan data-datanya:



Gambar 17: Pembacaan data dengan 10.000 baris data

Pada gambar diatas (Gambar 17) adalah command line dari MySQL untuk membaca 10.000 data yang sudah dimasukan / insert baik data *primary key auto-increment* dan juga UUID.



Gambar 18: Pembacaan data dengan 50.000 baris data

Pada gambar diatas (Gambar 18) adalah command line dari MySQL untuk membaca 50.000 data yang sudah dimasukan / insert baik data *primary key auto-increment* dan juga UUID.

Metode Pembanding Eksponensial (MPE)

Metode penelitian yang penulis gunakan dalam penelitian ini adalah Metode Perbandingan Eksponensial (MPE) adalah salah satu metode dari Sistem Pendukung Keputusan (SPK) yang digunakan untuk menentukan urutan prioritas alternatif keputusan dengan multi kriteria. MPE sangat cocok untuk penilaian skala ordinal (contoh sangat baik, baik, kurang, sangat kurang). Metode perbandingan eksponensial mempunyai keuntungan dalam mengurangi bias yang mungkin terjadi dalam analisis. Nilai skor yang menggambarkan urutan prioritas menjadi besar (fungsi eksponensial) ini mengakibatkan urutan prioritas alternatif keputusan lebih nyata.

Metode ini digunakan untuk membandingkan keberadaan satu variabel atau lebih pada dua atau lebih sampel yang berbeda atau waktu yang berbeda. Oleh karena itu penggunaan MPE dalam penelitian ini adalah dengan membandingkan penggunaan *resource* memori, prosesor dan storage pada komputer yang digunakan untuk pengujian.

Metode ini menekankan pada fenomena-fenomena objektif yang dikaji secara kuantitatif. Tujuan penggunaan metode ini dalam penelitian penelitian adalah untuk membuat gambaran atau lukisan secara sistematis, faktual dan akurat mengenai fakta, sifat serta hubungan antara berbagai fenomena. Peneliti ingin mengetahui adakah pengaruh terhadap kecepatan tulis dan baca data serta terhadap penggunaan perangkat keras memori, prosesor dan *storage*.

Penggunaan basis data membutuhkan perangkat keras sebagai media pengaturannya. Bagaimana data disimpan dan dibaca itu semua membutuhkan kinerja dari perangkat keras. Berapa besar dampak yang dihasilkan jika ada aktifitas baca dan tulis data dalam basis data terhadap perangkat keras juga menjadi konsentrasi penelitian ini. Karena kemampuan atau kecepatan aktifitas basis data sangat berpengaruh kepada besar kecilnya spesifikasi perangkat keras komputer yang digunakan.

Metode MPE ini diharapkan dapat mengidentifikasi data-data yang dihasilkan dari aktivitas penelitian, dalam kasus ini, penulis mendapatkan data dari waktu baris data ditulis dan dibaca pada perangkat lunak basis data MySQL, baik berupa waktu, jumlah data dan penggunaan *resource* seperti ram dan prosesor pada komputer. Langkah-langkah yang akan dilakukan dalam penelitian ini antara lain sebagai berikut:

1. Memasang perangkat lunak basis data MySQL pada komputer dengan sistem operasi Windows 10
2. Memasang perangkat lunak Bahasa pemrograman PHP Versi 7
3. Menulis skrip kode pemrograman PHP untuk menulis data
4. Menulis skrip kode pemrograman PHP untuk membaca data
5. Menjalankan skrip pemrograman dengan command line pada Command Prompt dan
6. Mencatat data penggunaan memori (ram), prosesor dan storage (hdd/ssd) untuk nantinya dihitung sebagai perbandingan

Penelitian ini juga diharapkan dapat berguna bagi para pelajar dan mahasiswa untuk menambah wawasan tentang bagaimana membuat atau mendesain basis data agar dapat maksimal dan digunakan dalam waktu panjang.

Kriteria dan Pembobotan

Untuk melakukan pembobotan variabel dilakukan dengan metode perbandingan pasangan, bobot variabel ditentukan dengan cara normalisasi vektor eigen, yang diasosiasikan dengan nilai eigen maksimum pada suatu matriks rasio sebelum membandingkan harus ditentukan skala nilai pengaruh atau penting antara variabel, seperti yang terlihat pada tabel berikut:

Tabel 6: Daftar keterangan nilai

Nilai	Keterangan
5	Sangat Baik
4	Baik
3	Cukup Baik
2	Kurang Baik
1	Buruk

Tabel 7: Daftar alternatif

Kode	Alternatif
A1	Penggunaan Prosesor
A2	Penggunaan Memori / Ram

A3	Penggunaan Storage / Disk / Ukuran
A4	Waktu Eksekusi

Tabel 8: Daftar kriteria tulis data

Kode	Kriteria
T1	Tambah data 10.000 dengan <i>auto-increment</i>
T2	Tambah data 50.000 dengan <i>auto-increment</i>
T3	Tambah data 10.000 dengan UUID
T4	Tambah data 50.000 dengan UUID

Tabel 9: Daftar kriteria baca data

Kode	Kriteria
B1	Baca data 10.000
B2	Baca data 50.000

III. HASIL PENELITIAN

Penggunaan metode MPE dilakukan agar dapat memberikan nilai perbandingan antara penggunaan *primary key* basis data MySQL dengan *auto-increment* dan UUID. Berdasarkan data yang sudah disajikan maka penelitian ini selanjutnya akan dilakukan perhitungan sesuai rumus dari MPE, berikut ini adalah perhitungannya:

Tabel 10: Pemberian bobot pada alternatif

N O	Kriteria	Bo bot	Alternatif					
			B 1	B 2	T 1	T 2	T 3	T 4
1	Waktu	5	5	5	5	1	3	2
2	Prosesor	4	5	5	5	2	3	1
3	Memory / Ram	4	5	5	1	1	5	4
4	Storage / Ukuran	3	5	5	5	3	4	1

Tahap perhitungan:

1. B1 (Baca Data 10.000 baris)
$$= 5^5 + 5^4 + 5^4 + 5^3$$

$$= 3.125 + 625 + 625 + 125$$

$$= \mathbf{4.500}$$
2. B2 (Baca Data 50.000 baris)

$$= 5^5 + 5^4 + 5^4 + 5^3$$

$$= 3.125 + 625 + 625 + 125$$

$$= \mathbf{4.500}$$

3. T1 (Tambah data 10.000 dengan *auto-increment*)

$$= 5^5 + 5^4 + 1^4 + 5^3$$

$$= 3.125 + 625 + 4 + 125$$

$$= \mathbf{3.879}$$

4. T2 (Tambah data 50.000 dengan *auto-increment*)

$$= 1^5 + 2^4 + 1^4 + 3^3$$

$$= 5 + 16 + 4 + 27$$

$$= \mathbf{52}$$

5. T3 (Tambah data 10.000 dengan UUID)

$$= 3^5 + 3^4 + 5^4 + 4^3$$

$$= 243 + 81 + 625 + 64$$

$$= \mathbf{1.013}$$

6. T4 (Tambah data 50.000 dengan UUID)

$$= 2^5 + 1^4 + 4^4 + 1^3$$

$$= 32 + 4 + 256 + 3$$

$$= \mathbf{295}$$

Setelah mendapatkan perhitungan menggunakan Metode Perbandingan Eksponensial (MPE), selanjutnya data-data alternatif diberikan *ranking* untuk dapat melihat perbandingannya maka didapat hasil sebagai berikut:

Tabel 11: Pemberian ranking untuk semua alternatif

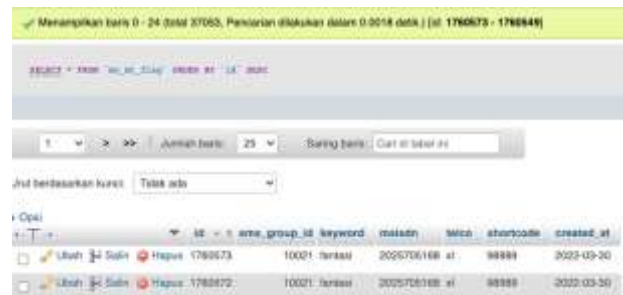
No	Alternatif	Nilai	Ranking
1	B1 (Baca Data 10.000 baris)	4.500	2
2	B2 (Baca Data 50.000 baris)	4.500	1
3	T1 (Tambah data 10.000 dengan <i>auto-increment</i>)	3.879	3
4	T2 (Tambah data 50.000 dengan <i>auto-increment</i>)	52	6
5	T3 (Tambah data 10.000 dengan UUID)	1.013	4
6	T4 (Tambah data 50.000 dengan UUID)	295	5

Pembacaan data, baik 10.000 dan 50.000 dengan spesifikasi komputer yang digunakan tidak menunjukkan

adanya perbedaan yang signifikan, pembacaan data terbilang cepat dengan penggunaan *resource* yang sedikit. Hasil yang signifikan ditunjukkan pada *query insert* data, sebagaimana ditunjukkan pada tabel diatas.

Tambah data menggunakan *primary key auto-increment* dengan banyaknya data 10.000 lebih cepat dari *primary key* menggunakan UUID sedangkan tambah data menggunakan UUID dengan banyaknya data 50.000 lebih baik dari pada *primary key auto-increment*.

Penulis juga menemukan dalam penggunaan *primary key* dengan *auto-increment* akan sulit dalam reduksi data jika ada kasus dimana data dalam tabel dihapus namun tidak mengurangi ukuran dalam database. Berikut ini penulis menemukan hal tersebut pada sebuah tabel



Gambar 12: Perbedaan *primary key auto-increment* dengan jumlah data

Dari Gambar 12 diatas dapat dilihat pada data menunjukkan sebanyak 37.053 baris, namun id *increment* nya menunjukkan 1.760.549, hal ini dikarenakan dalam penggunaan *auto-increment*, penambahan didasari dari angka sebelumnya ditambah 1.

Sebagai contoh jika dalam tabel ada 10 baris data dengan id *increment* 1-10 lalu data dihapus sebanyak 5 dari id 1-5 maka jika ditambah data kembali ke tabel tersebut maka sistem *auto-increment* akan memberikan id data baru yaitu 10 + 1 menjadi 11. Jika kita menggunakan basis data untuk menampung data besar maka perlu diperhatikan jika menggunakan *auto-increment* karena panjang digit nya terbatas sebagaimana ditunjukkan pada tabel dibawah ini

Tabel 12: Nilai maksimal pada tipe data MySQL

Tipe Data	Maksimal
INT Signed	2,147,483,647
INT Unsigned	4,294,967,295
BIGINT Signed	9,223,372,036,854,775,806
BIGINT Unsigned	18,446,744,073,709,551,615

UUID berguna untuk memberi entitas nama khusus mereka sendiri, misalnya, dalam database. Ada beberapa cara untuk menghasilkannya, termasuk metode berdasarkan waktu, alamat MAC pada perangkat

komputer, hash / md5, dan nomor acak atau gabungan angka dan huruf yang diacak.

Jika UUID dibuat secara acak, maka pasti ada kemungkinan bahwa suatu urutan dibangkitkan lebih dari sekali, terutama karena kita menghasilkan lebih banyak dan lebih banyak lagi. Mari kita hitung probabilitas, p , bahwa tidak ada duplikat di antara sekumpulan r UUID.

Pada umumnya, setiap karakter UUID dapat berupa angka 0 sampai 9, atau huruf a sampai f. 32 heksadesimal $\times \log_2(16)$ bit/heksadesimal = 128 bit dalam UUID. Dalam versi 4, varian 1 jenis UUID, 6 bit diperbaiki dan 122 bit sisanya dihasilkan secara acak, dengan total 2^{122} kemungkinan UUID. Kami akan merujuk ke nilai ini sebagai n . Segera, kita tahu bahwa jika jumlah UUID yang dihasilkan melebihi ruang kemungkinan UUID, yaitu jika $r > n$, maka harus ada beberapa duplikat.

Banyaknya cara agar tidak ada duplikat adalah $n * (n - 1) * (n - 2) * \dots * (n - (r - 1))$. UUID pertama dapat berupa salah satu dari n kemungkinan, yang kedua dapat berupa salah satu dari n kecuali yang pertama ($n - 1$), yang ketiga dapat berupa apa saja kecuali dua yang pertama ($n - 2$), dan seterusnya. Dan jumlah total cara untuk menghasilkan r UUID adalah n^r karena masing-masing r UUID memiliki n kemungkinan yang berbeda. Bagilah dua hitungan untuk mendapatkan probabilitas bahwa tidak ada UUID duplikat, berikut ini cara menghitung probabilitas duplikasi UUID:

$$\frac{n!}{n^r(n-r)!}$$

Dengan angka sebesar ini, menghitung probabilitas secara langsung tidak layak secara komputasi. Faktorial besar dapat didekati dengan rumus Stirling, untuk mendapatkan perkiraan probabilitas menghasilkan r UUID unik

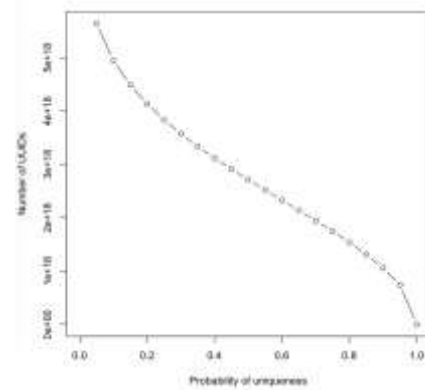
$$\frac{n}{(n-r)} n^{-r+5} e^{-r}$$

Jika r adalah tetap dan $n \rightarrow \infty$, bagian pertama dari persamaan di atas dapat dinyatakan sebagai berikut:

$$\sqrt{\frac{n}{n-r}} \left(1 + \frac{r}{n-r}\right) n^{-r} \rightarrow 1. e^{-r}$$

Kalikan dengan bagian kedua, dan hasilnya adalah 1. Jadi jika $n \gg r$, hampir pasti tidak akan ada duplikat. Tetapi jika r mendekati n , UUID mungkin berulang.

Persisnya seberapa besar r seharusnya? Kita dapat mendekati masalah ini dari sudut yang berbeda, di mana daripada memulai dengan n dan r , dan mencoba menghitung p , kita mulai dengan n dan p , dan sampai pada r . Di bawah ini adalah plot yang menunjukkan perkiraan jumlah UUID untuk probabilitas keunikan tertentu. Nilainya dihitung dengan rumus aproksimasi di mana $r = .5 + \sqrt{.25 - 2 * \ln(p) * n}$.



Gambar 13: Probabilitas kemunculan data yang sama pada generate UUID

Sampel $3,26 * 10^{16}$ UUID memiliki peluang 99,99% untuk tidak memiliki duplikat. Menghasilkan banyak UUID, dengan kecepatan satu per detik, akan memakan waktu satu miliar tahun. Jadi sementara UUID tidak benar-benar unik, mereka cukup unik untuk tujuan praktis, dengan mempertimbangkan batasan alami dari rentang hidup manusia dan pemisahan sistem.

Selain ketetapan pemberian nilai dan panjang UUID, pengguna juga dapat membuat UUID dengan susunannya sendiri, sebagai contoh kita dapat menggunakan tanggal dan jam (*datetime*) dan mencampur dengan angka acak lainnya sehingga duplikasi dapat dihindari.

IV. PEMBAHASAN

UUID didasari dengan nilai acak yang dibuat oleh pengguna itu sendiri. Pengidentifikasi Unik *Universal*, atau UUID, memiliki panjang angka 128 bit, terdiri dari 16 oktet dan direpresentasikan sebagai 32 karakter dasar-16, yang biasanya digunakan untuk mengidentifikasi informasi *id* pada seluruh sistem komputer. Dalam pemberian nilai *primary key* menggunakan UUID, berdasarkan hasil pengujian diatas tidak dianjurkan untuk penyimpanan data yang kecil atau sedikit karena panjang digit pada nilai UUID akan terlihat signifikan, namun pada kasus penggunaan basis data dengan *big data*.

Jika pada *auto-increment*, nomor akan ditambah secara eksponen dengan cara menambahkan 1 dari *increment* terakhir, maka pada *auto-increment* tidak akan ditemukan kesamaan angka namun nilainya terbatas sebagaimana diterangkan pada tabel 12 diatas. Berbeda dengan UUID yang panjang karakternya terbilang tetap atau statis dan tidak dibatasi dengan panjang karakter atau maksimal digit nya, namun ada hal yang perlu dihindari dari penggunaan UUID yaitu kesamaan nilai yang akan muncul dalam generate nilai UUID.

Kecepatan penulisan data terhadap basis data MySQL menggunakan *primary key auto-increment* lebih cepat dari UUID jika memasukan sedikit data, jika data yang dimasukan banyak, UUID lebih cepat.

Kecepatan membaca data baik data yang menggunakan *primary key auto-increment* ataupun UUID

terbilang cepat, sebagai acuan penulis sudah memberikan informasi perangkat yang digunakan.

V. KESIMPULAN

Hasil pengujian yang dilakukan mendapati bahwa pembacaan data MySql menggunakan *primary key auto-increment* atau UUID tidak terlalu signifikan dengan banyaknya data yang diujikan, namun pada penambahan data ke MySql, penggunaan *auto-increment* dengan jumlah data yang di insert terbilang sedikit lebih cepat / baik dari UUID namun pada jumlah yang diujikan lebih banyak maka penggunaan UUID menjadi lebih cepat / baik. Penelitian ini juga mendapati beberapa hal yang menjadi keuntungan dan juga hal yang merugikan atau kurang baik dalam penggunaan *primary key* menggunakan *auto-increment* atau UUID dapat dilihat pada tabel dibawah ini:

Tabel 13: Keuntungan dan kerugian pada *primary key auto-increment*

Keuntungan	Kerugian
Kinerja pada data sedikit atau kecil sangat cepat	Jika digunakan untuk data besar atau big data, <i>auto-increment</i> memiliki batasan pada nilai yang dapat disimpan
Tidak perlu melakukan generate angka ada bahasa pemrograman karena sudah disediakan oleh basis data MySql	Jika melakukan penghapusan data maka nomor <i>increment</i> tetap tidak berkurang hal ini membuat penghapusan data pada tabel MySql tidak berpengaruh pada ukuran tersimpan.

Tabel 14: Keuntungan dan kerugian pada *primary key* UUID

Keuntungan	Kerugian
Tidak ada limitasi panjang data	Generate nilai UUID masih mengandalkan bahasa pemrograman lalu digunakan dalam basis data
Baik untuk data yang banyak atau big data	Pelu cermat dalam membuat atau generate UUID agar terhindar dari duplikasi atau nilai yang sama di generate lebih dari 1 kali
Jika melakukan penghapusan data maka data akan berkurang hal ini membuat penghapusan data pada tabel MySql dapat berpengaruh pada ukuran tersimpan	

VI. REFERENSI

- [1] Andria & Mei Lenawati. 2015. Perancangan Basis Data Sistem Pembayaran Sport Center Berbasis Mysql. Data Manajemen Dan Teknologi Informasi. Amikom Sultan Agung. Diakses dari <https://media.neliti.com/media/publications/91352-ID-perancangan-basis-data-sistem-pembayaran.pdf>
- [2] Agung Pujiyanto, Awin Mulyati & Rachmawati Novaria. 2018. Pemanfaatan Big Data Dan Perlindungan Privasi Konsumen Di Era Ekonomi Digital. Majalah Ilmiah BIJAK Universitas 17 Agustus 1945. Diakses dari <https://ojs.stiami.ac.id/index.php/bijak/article/download/201/134>
- [3] Alvin Dwi Hardiansyah & Catur Nugraheni Puspita Dewi. 2020. Perancangan Basis Data Sistem Informasi Perwira Tugas Belajar (sipatubel) Pada Kementerian Pertahanan. Seminar Nasional Mahasiswa Ilmu Komputer dan Aplikasinya (SENAMIKA). Diakses dari <https://conference.upnvj.ac.id/index.php/senamika/article/download/529/436>
- [4] Budi Maryanto. 2017. Big Data Dan Pemanfaatannya Dalam Berbagai Sektor. Jurnal STMIK LIKMI. Diakses dari https://jurnal.likmi.ac.id/Jurnal/7_2017/0717_02_BudiMaryanto.pdf
- [5] Emyana Ruth Eritha Sirait. 2016. Implementasi Teknologi Big Data Di Lembaga Pemerintahan Indonesia. Jurnal Penelitian Pos & Informatika. Diakses dari <https://jurnal-ppi.kominfo.go.id/index.php/jppi/article/view/060201>
- [6] Hendra, & Andriyani, W. (2020). STUDI KOMPARASI MENYIMPAN DAN MENAMPILKAN DATA HISTORI ANTARA DATABASE TERSTRUKTUR MARIADB DAN DATABASE TIDAK TERSTRUKTUR INFLUXDB. Jurnal Teknologi Technoscintia, 12(2), 168–174. Diakses dari <https://ejournal.akprind.ac.id/index.php/technoscintia/article/view/2663>
- [7] Herman Yuliansyah. 2014. Perancangan Replikasi Basis Data Mysql Dengan Mekanisme Pengamanan Menggunakan Ssl Encryption. JURNAL INFORMATIKA Vol. 8, No. 1. Diakses dari <https://media.neliti.com/media/publications/102982-ID-perancangan-replikasi-basis-data-mysql-d.pdf>
- [8] J. I. Maanari, R. Sengkey, F. Wowor & Y. D. Y. Rindengan. 2013. Perancangan Basis Data Perusahaan Distribusi dengan Menggunakan Oracle. E-Journal Universitas Sam Ratulangi. Diakses dari <https://ejournal.unsrat.ac.id/index.php/elekdankom/article/download/1719/131>

- [9] Marimin & Maghfiroh. 2011. Aplikasi Teknik Pengambilan Keputusan Dalam Manajemen Rantai Pasok. Bogor: IPB Press. Diakses dari <https://repository.ipb.ac.id/handle/123456789/42649>
- [10] Muh.Nasir & Muhammad Akbar. 2020. Perancangan Website Smp Negeri 2 Lamasi. Open Jurnal Universitas Cokroaminoto Palopo. Diakses dari <https://journal.uncp.ac.id/index.php/computare/article/view/162/154>
- [11] Raghu Ramakrishnan & Johannes Gehrke. 2002. Database Management Systems, 3rd Edition. McGraw-Hill. Diakses dari http://131.193.209.39:8003/view_syllabi/static/view_syllabi/syllabus/IDS%20521%20Chang%20Sp17.pdf
- [12] Haris Rangkuti. 2011. Teknik Pengambilan Keputusan Multi Kriteria Menggunakan Metode Bayes, MPE, CPI, dan AHP. ComTech Vol.2 No. 1 Juni 2011: 229-238. Diakses dari <https://media.neliti.com/media/publications/166428-ID-teknik-pengambilan-keputusan-multi-krite.pdf>
- [13] Sapri & Ferry Hari Utami. 2015. Pembuatan Website Sekolah Menengah Pertama (smp) Negeri 12 Seluma. Jurnal Universitas Dehasen Bengkulu. Diakses dari <https://jurnal.unived.ac.id/index.php/jmi/article/download/40/39/>
- [14] Silalahi, M., & Wahyudi, D. (2018). Perbandingan performansi database mongodb dan mysql dalam aplikasi file multimedia berbasis web. Cbis (Computer Based Information System) Journal, 01, 63–78. Diakses dari <http://113.212.163.133/index.php/cbis/article/view/574/414>
- [15] Sprague, R. H. Building Effective Decision Support Systems. Grolier, New Jersey, 1982