ANALISA KINERJA *QUERY STORED PROCEDURE* PADA *DATABASE MANAGEMENT SYSTEM (DBMS) MYSQL*

Indra Warman[1)], Wildani[2)]
Fakultas Teknik, Institut Teknologi Padang
E-mail: indrawmn@gmail.com, Wildani1000@gmail.com

Abstrak

Abstrak : Penggunaan *query* pada *database* dengan jumlah *record* data transaksi yang besar tentunya membutuhkan akses sumber daya DBMS *(Database management System),* pada *database* MySQL, *query* yang digunakan untuk jumlah *record* data ribuan tentunya akan mempengaruhi kinerja *database*, eksekusi *SQL query* dengan jumlah *record* yang sangat banyak akan memerlukan waktu yang lama dan mempengaruhi waktu proses pengolahan data. Penggunaan perintah *SQL stored procedure* dapat mengoptimalkan penggunaan *query* terutama untuk melakukan perintah yang sering digunakan dalam manipulasi data, sehingga akan mempercepat proses, *stored procedure* merupakan kumpulan pernyataan *Query* yang disimpan pada *database*, untuk itu perlu dilakukan pengujian kinerja waktu eksekusi *query*. Penelitian ini bertujuan memberikan analisis kinerja *query stored procedure* pada *Relational Database Management System* (RDBMS) MySQL. Pada penelitian dilakukan pengujian kinerja *query stored procedure* dengan jumlah *record* data dari 50 sampai dengan 25000. *Query stored procedure* yang diuji menggunakan perintah *SQL* (*Create, Select, Update, Delete*) dengan kasus *database* transaksi belanja *online*. Hasil rekapitulasi pengujian kinerja *query stored procedure* pada MySQL menunjukkan pada pengujian *query stored procedure select* dan *delete* waktu eksekusinya berbanding lurus dengan jumlah *record* datanya. Sedangkan pengujian *query stored procedure create* dan *update* terjadi peningkatan waktu eksekusi yang cukup signifikan terutama pengujian dengan jumlah 25000 *record* data.

Kata Kunci: *SQL Query, Stored Procedure,* MySQL, Kinerja, RDBMS

***Abstract** : The Utilization of queries on databases with a large number of transaction data records will make requires access to DBMS (Database management System) resources, in MySQL databases, queries used for the number of data records in the thousands will certainly affect the performance of the database. SQL query execution with a very large number of records will take a long time and will affect the data processing time. The use of SQL stored procedure can optimize the use of queries, especially for SQL that are often used in data manipulation, so that it will speed up the process, stored procedures are a collection of query statements that are created and then stored in the database and can be reused. execution in query stored procedure. This study aims to provide a performance analysis of the query stored procedure in the MySQL Relational Database Management System (RDBMS). In this research, the performance of the query stored procedure was tested with the number of data records from 50 to 25000. The query stored procedure being tested is using the SQL command (Create, Select, Update, Delete) with the transaction database case of on online shopping. The recapitulation of performance testing for stored procedure queries in MySQL shows that in testing the stored procedure, select and delete execution time is directly proportional to the number of data records. Meanwhile, testing of query stored procedure create and update has a significant increase in execution time, especially in testing with a total of 25000 data records.*

***Keywords**: SQL Query, Stored Procedure, Mysql, Performance, RDBMS*

PENDAHULUAN

Penggunaan Teknologi *Database* terutama *Relational Database Management System* (RDBMS) yang berisi data-data yang terstruktur, dimana antara satu *table* dengan *table* yang lainnya dihubungkan dengan kunci relasi (Kadir, 2020). Berdasarkan *db-engine ranking* Mei tahun 2020 RDBMS MySQL merupakan urutan kedua RDBMS terpopuler dan

banyak digunakan *(https://db-engines.com, 2020*). *Database Mysql* mendukung untuk aspek keamanan dan penggunaan *stored procedure* (Solichin, 2010). Pada *database* MySQL, *query* SQL yang digunakan untuk jumlah *record* data dengan jumlah ribuan tentunya akan mempengaruhi kinerja *database* (Muliyantoro, 2013). eksekusi *SQL query* dengan jumlah *record* yang sangat banyak akan memerlukan waktu yang lama dan akan mempengaruhi waktu proses pengolahan data (Hastono, 2019a). Untuk mengoptimalkan *query* dengan *record* data yang sangat banyak dapat dilakukan dengan optimasi dengan menggunakan *stored procedure* (Hastono, 2019b).

Penggunaan *stored procedure* memiliki beberapa manfaat diantaranya : lebih sedikit hasil duplikasi, eksekusi lebih cepat, lalu lintas jaringan minimum dan keamanan yang lebih baik. *Stored procedure* ini dapat digunakan untuk membagi beban *resource* yang terpakai pada saat aplikasi yang sedang berjalan. *Stored procedure* juga efektif digunakan untuk eksekusi *query* yang dilakukan secara rutin dan diproses pada *database server* sehingga dapat memangkas waktu eksekusi dan akan mempercepat proses pada kerja dari pengolahan data, Selain meringkas penggunaan sintak *query* SQL juga dapat meningkatkan performasi dari *database*. Sehingga lebih mudah menentukan penerapan *database* terhadap pengguna dalam menggunakan *stored procedure*.

Berdasarkan hal tersebut maka diperlukan suatu analisa untuk melihat kinerja *query stored procedure* pada DBMS MySQL. Tujuan dari penelitian ini adalah untuk memberikan analisis kinerja *query stored procedure* pada *Relational Database Management System* (RDBMS) MySQL menggunakan eksekusi dari perintah SQL (*Create/Insert, Select, Update, Delete*).

METODE PENELITIAN

Penelitian ini merupakan suatu metode analisa, yaitu analisa kinerja *query stored procedure* pada *Database Management System* (DBMS) MySQL. Penelitian ini dilakukan agar dapat melihat kinerja *query stored procedure* terhadap *Database Management System* (DBMS) MySQL. *Database* MySQL yang digunakan pada penelitian ini adalah MySQL 5.6.21 *Communit*y, rancangan pengujian kinerja *query stored procedure* pada *DBMS* MySQL ini, dilakukan perintah *SQL* (*Create/Insert, Select, Update, Delete*) dengan jumlah *record* data mulai dari 50, 100, 500, 1000, 5000, 10000, sampai 25000 pada DBMS MySQL sehingga waktu eksekusi perintah *query* tersebut dapat dianalisa, pengukuran kinerja pada variabel (*Stored Procedure*) menggunakan *query* melalui MySQL *Command line*, hal ini dilakukan untuk validates waktu pengukuran, dibandingkan dengan menggunakan *tools* melalui *browser* yang nantinya akan dipengaruhi oleh *cache* pada *browser* tersebut.

Tahapan yang dilakukan dalam penelitian adalah instalasi dan konfigurasi *database* MySQL pada OS *Windows* 10 64bit, merancang struktur *table* pada *database* yang akan digunakan sebagai pengujian *query*, eksekusi *query stored procedure* (*Create/Insert, Select, Update, Delete*) dilakukan secara bertahap dengan jumlah *record* data yang berbeda.

HASIL DAN PEMBAHASAN

Pengujian Kinerja *Query* Pada DBMS MySQL

Pada rancangan pengujian kinerja *query stored procedure* pada DBMS MySQL nantinya dilakukan perintah SQL (*Create, Select, Update, Delete*) dengan jumlah *record* data mulai dari 50, 100, 500, 1000, 5000, 10000, sampai 25000, selanjutnya akan terlihat selisih waktu eksekusi pengujian *query stored*

procedure, sehingga kecepatan pada masing-masing *query* yang diujikan dapat dianalisa.

Query create/insert ini menggunakan perintah SQL *insert* untuk melakukan penambah data pada *table* orderan. Berikut ini adalah contoh penggunaan *query* dengan *stored procedure* dengan perintah *create/insert pada table orderan.*

```
delimiter //
create procedure insert_orderan25000(
te int,
id_pel varchar(8),
id_pro varchar(5),
ha int,
jum int,
dis float,
t_ba int,
tgl_pe date,
id_k int)
begin
insert into
orderan(trx,id_pelanggan,id_produk,harga,ju
mlah,diskon,total_bayar,tgl_pesan,id_ket)
values(te,id_pel,id_pro,ha,jum,dis,t_ba,tgl_p
e,id_k);
end;
//
call insert_orderan25000 (25001,'p0009',
'd0017', 125000, 1, 0, 125000, '2021-01-10',
1);
```

***Stored Procedure create/insert* data**

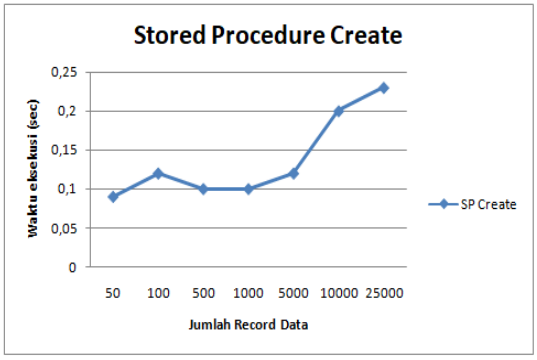
Hasil pengujian perintah SQL *stored procedure insert* data mulai dari 50 sampai dengan 25.000 *record sample* data yang dilakukan, selanjutnya dilakukan direkapitulasi pada tabel.1, untuk menampilkan hasil waktu pengujian proses query SQL *stored procedure insert.*

Tabel 1. Hasil *Store Procedure create/insert*

Keterangan (Jumlah Record Data)	50	100	500	1000	5000	10000	25000
Waktu Eksekusi (Sec)	0,09	0,12	0,10	0,10	0,12	0,20	0,23

***Stored Procedure create/insert* data (grafik)**

Untuk menampilkan hasil pengujian pada tabel.1, berikut adalah informasi dalam bentuk grafik yang terkait SQL *store procedured insert* data yang dilakukan pada *table* mulai dari 50 sampai dengan 25.000 *record*, pada gambar 1. menampilkan rekapitulasi pengujian data serta waktu proses query SQL *stored procedure insert*, seperti berikut :



Gambar 1. Grafik Hasil *Query Stored Procedure Create/insert data*

1. *Query Stored Procedure Select*

Query stored procedure dengan perintah *select* digunakan untuk melihat menyeleksi, atau memilih data pada tabel-tabel yang ada dalam *database*. *Query select* ini bisa menampilkan semua kolom, sebagian kolom, serta berdasarkan kondisi yang diinginkan, berikut adalah contoh penggunaan *query* dengan *stored procedure* dengan perintah *select pada tabe;* dalam *database*:

```
delimiter //
create procedure tampil()
begin
Select
pelanggan.nm_pelanggan,produk.nm_pro
duk,orderan.jumlah,orderan.total_bayar
from
pelanggan,produk,orderan
where
pelanggan.id_pelanggan=orderan.id_pela
nggan and
produk.id_produk=orderan.id_produk and
id_ket=2;
end;
//
 Call tampil ();
```

Stored Procedure Select

Hasil pengujian perintah SQL *store procedure select* yang dilakukan mulai

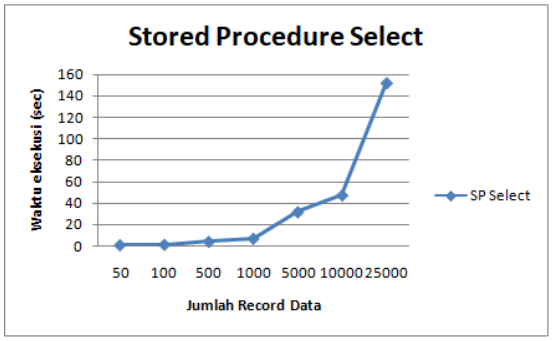
dari 50 sampai dengan 25.000 record sample data, selanjutnya dilakukan rekapitulasi untuk menampilkan waktu pengujian proses query SQL *stored procedure select*, seperti pada tabel 2 berikut :

Tabel 2. Hasil Pengujian *Select*

Keterangan (Jumlah Record Data)	50	100	500	1000	5000	10000	25000
Waktu Eksekusi (Sec)	0,53	0,93	3,70	6,64	31,50	47,20	152,28

***Stored Procedure select* (grafik)**

Untuk menampilkan hasil pengujian pada tabel.2, berikut merupakan informasi dalam bentuk grafik yang terkait SQL *stored procedure select record* data yang dilakukan pada *table* mulai dari 50 sampai dengan 25.000 *record*, pada gambar 2. menampilkan rekapitulasi hasil pengujian data serta waktu proses *query* SQL *stored procedure select*.



Gambar 2. Grafik Hasil *Query Stored Procedure Select*

2. *Query Stored Procedure Update*

Query stored procedure dengan perintah *update* berguna untuk mengubah data pada *database*, berikut adalah contoh penggunaan *query* dengan *stored procedure* dengan perintah *update* dalam *database*.

```
delimiter //
create procedure updatedata50()
begin
update orderan50 set id_ket=3
where
id_ket=2;
end;
//
call updatedata50();
```

Stored Procedure update

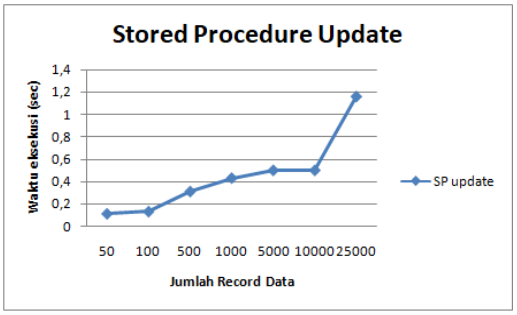
Hasil pengujian perintah SQL *stored procedure update record* data mulai dari 50 sampai dengan 25.000 *record sample* data yang dilakukan, selanjutnya dilakukan direkapitulasi pada tabel 3, untuk menampilkan hasil waktu pengujian proses query SQL *stored procedure create* yang dibutuhkan, seperti berikut :

Tabel 3. Hasil Pengujian *Update*

Keterangan (Jumlah Record Data)	50	100	500	1000	5000	10000	25000
Waktu Eksekusi (Sec)	0,11	0,13	0,31	0,43	0,50	0,50	1,16

***Stored Procedure update* (grafik)**

Untuk menampilkan hasil pengujian pada tabel.3, berikut merupakan informasi dalam bentuk grafik terkait SQL *stored procedure update record* data yang dilakukan pada *table* mulai dari 50 sampai dengan 25.000 *record* data, pada gambar 3 menampilkan rekapitulasi hasil pengujian data serta waktu proses yang dibutuhkan *query* SQL *stored procedure update.*



Gambar 3. Grafik Hasil *Query Stored Procedure Update*

Query stored procedure dengan perintah *delete* berguna untuk menghapus data pada *database*. Berikut ini adalah contoh penggunaan *query* dengan *stored procedure* dengan perintah *delete* dalam *database*:

```
delimiter //
create procedure deletedata50()
begin
delete from orderan50
```

```
where id_ket=3;
end;
//
call deletedata50();
```

Stored Procedure delete

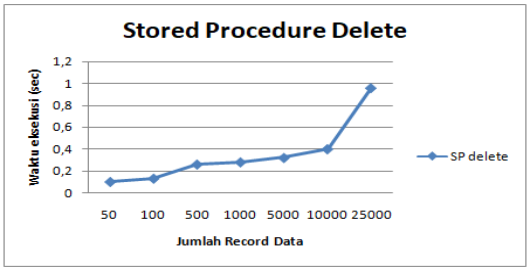
Pada pengujian perintah SQL *stored procedure delete record* data mulai dari 50 sampai dengan 25.000 *record sample* data yang dilakukan, selanjutnya hasil query dilakukan direkapitulasi pada tabel 4, untuk menampilkan hasil waktu pengujian proses *query* SQL *stored procedure delete*, seperti berikut :

Tabel 4. Hasil Pengujian *Delete*

Keterangan (Jumlah Record Data)	50	100	500	1000	5000	10000	25000
Waktu Eksekusi (Sec)	0,10	0,13	0,26	0,28	0,32	0,40	0,96

***Stored Procedure delete* (grafik)**

Untuk menampilkan hasil pengujian pada tabel.4, berikut merupakan informasi dalam bentuk grafik yang terkait SQL *store procedure delete record* data pada *table* mulai dari 50 sampai dengan 25.000 *record* data, pada gambar 4. menampilkan rekapitulasi hasil pengujian data serta waktu proses *query* SQL *stored procedure delete.*



Gambar 4. Grafik Hasil *Query Stored Procedure Delete*

Rekapitulasi SQL Stored Procedure

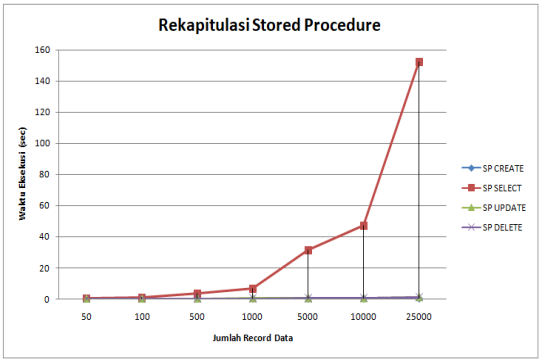
Pada rekapitulasi *query* berikut merupakan hasil pengujian *store procedure (create/insert, select, update, delete)* pada table 1 sampai dengan tabel 4, seperti terlihat pada tabel.5 berikut ini :

Tabel 5. Rekapitulasi *Query Stored Procedure*

Jumlah *record* Data	Waktu Eksekusi *Stored Procedure*			
	Create	*Select*	*Update*	*Delete*
50	0,09	0,53	0,11	0,10
100	0,12	0,93	0,13	0,13
500	0,10	3,70	0,31	0,26
1000	0,10	6,64	0,43	0,28
5000	0,12	31,50	0,50	0,32
10000	0,20	47,20	0,50	0,40
25000	0,23	152,28	1,16	0,96

Rekapitulasi *SQL Stored Procedure*

Rekapitulasi hasil pengujian pada masing-masing *store procedure (create/insert, select, update, delete)* pada pada tabel 1 sampai dengan 4, ditampilkan dalam bentuk grafik seperti gambar 5. Hasil pengujian *stored procedure* dengan perintah *select* pada *record* data 500 sampai 25000 waktu eksekusinya sangat meningkat, waktu eksekusi meningkat secara signifikan pada pada *record* data 5000, 10000, dan 25000 dengan waktu eksekusi 31,50 *sec*, 47,20 *sec*, dan 152,28 *sec*. Pengujian *stored procedure* menggunakan perintah *create/insert data* waktu eksekusinya bervariasi dimana perbandingam waktu eksekusinya kecil, waktu eksekusi terbesar yang digunakan 0,23 *sec* pada *record* data 25000. Selanjutnya pengujian *stored procedure* menggunakan perintah *update* waktu eksekusinya bervariasi. Pada *record* data 5000 dan 10000 memerlukan waktu eksekusi yang sama 0,50 *sec*, untuk waktu eksekusi terlama pada pengujian *record* data 25000 yaitu 1,16 *sec*, sedangkan pengujian *stored procedure* menggunakan perintah *delete* waktu eksekusinya mengalami peningkatan setiap melakukan pengujian dengan penambahan *record* data, waktu eksekusi naik secara signifikan pada *record* data 5000, 10000, dan 25000 dengan waktu eksekusi 0,32 *sec*, 0,40 *sec*, dan 0,96 *sec*.



Gambar 5. Grafik Rekapitulasi *Stored Procedure*

KESIMPULAN

Berdasarkan hasil rekapitulasi pengujian kinerja *query stored procedure* pada MySQL menunjukkan bahwa pada pengujian *query stored procedure select* dan *delete* waktu eksekusi berbanding lurus dengan jumlah *record* datanya, sedangkan pengujian *query stored procedure create* dan *update* terjadi peningkatan waktu eksekusi yang cukup signifikan terutama pada pengujian dengan jumlah 25000 *record* data.

DAFTAR PUSTAKA

Dudu, A. *et al.* 'Pengukuran Kinerja Stored Procedure Pada Database Relasional ',Jurnal Siliwangi Sains Teknologi, 5(2), 2019, pp. 51–55.

Hastono, T. 'Optimasi Query Sistem Informasi Menggunakan Stored Procedure', *Jurnal Dinamika Informatika* , 8(2), 2019a, pp. 79–89.

Hastono, T. 'Query pada Sistem Informasi Keuangan SD Muhhamadiyah Sidoarum Berbasis Client-Server', 2019b, pp. 35–39.

https://db-engines.com *The DB-Engines Ranking ranks database management systems according to their popularity. The ranking is updated monthly.*, *https://db-engines.com*. Available at: https://db-engines.com/en/ranking, 2020.

Kadir, A. *Dasar Perancangan dan Implementasi Database Relasional (Edisi Revisi)*. Penerbit Andi, 2020

Muliyantoro, H. S. (2013) 'Penerapan Metode Load-Balancing Clusters Pada Database Server Guna Peningkatan Kinerja Pengaksesan Data', *Jurnal Techno Nusa Mandiri*, 10(1), pp. 97–108.

Solichin, A. *MySQL5:* Dari Pemula Hingga Mahir. Achmad Solichin. 2010