Jurnal Teknovasi
Volume 04, Nomor 01, 2017, 42-50
ISSN : 2355-701X

METODE PENCARIAN DATA MENGGUNAKAN QUERY HASH JOIN DAN QUERY NESTED JOIN

Junus Sinuraya[1*]
[1]Jurusan Teknik Komputer dan Informatika
Politeknik Negeri Medan,Medan,Indonesia
[*]Email: Junus.Sinuraya2012@gmail.com

ABSTRAK

Pengaksesan data atau pencarian data dengan menggunakan *Query* atau *Join* pada aplikasi yang terhubung dengan sebuah database perlu memperhatikan ketepatgunaan implementasi dari data itu sendiri serta waktu prosesnya. Ada banyak cara yang dapat dilakukan oleh database manajemen sistem dalam memproses dan menghasilkan jawaban sebuah *query*. Semua cara pada akhirnya akan menghasilkan jawaban (*output*) yang sama tetapi pasti mempunyai harga yang berbeda-beda, seperti misalnya kecepatan waktu untuk merespon data Beberapa query yang sering digunakan untuk pemrosesan data yaitu *Query Hash Join* dan *Query Nested Join*, kedua query memiliki algoritma yang berbeda tapi menghasilkan *output* yang sama. Dengan menggunakan aplikasi yang dirancang menggunakan Microsoft Visual Studi 2010 dan Microsoft SQL Server 2008 berbasis Jaringan untuk melakukan pengujian kedua algoritma atau query dengan paramter *running time* atau kecepatan waktu merespon data. Pengujian dilakukan dengan jumlah tabel yang dihubungkan dan jumlah baris/record. Hasil dari penelitian adalah kecepatan waktu query untuk merespon data untuk jumlah data yang kecil *query hash join* lebih baik sedangkan jumlah data yang besar *query nested join* lebih baik.

Kata Kunci : ***Query, Hash Join, Nested Join.***

PENDAHULUAN

Database Manajemen Sistem (DBMS) merupakan perantara bagi pemakai dengan basis data. Untuk berinteraksi dengan DBMS (basis data) menggunakan bahasa basis data yang telah ditentukan oleh perusahaan DBMS. Bahasa basis data biasanya terdiri atas perintah-perintah yang di formulasikan sehingga perintah tersebut akan diproses olah DBMS. Perintah-perintah biasanya ditentukan oleh user. Perintah *user* berinterakasi dengan Database Manajemen Sistem diantaranya yaitu memasukkan data, mengubah data, Menghapus dan menampilkan data atau disebut dengan *Data Manipulation Language* (DML). *Structure Query Language(SQL)* adalah standar bahasa untuk berinteraksi dengan database dengan menggunakan operasi-operasi umum pada basis data.

Query adalah semacam kemampuan untuk menampilkan suatu data dari database dimana mengambil dari tabel-tabel yang didatabase, namun tabel tersebut tidak semua ditampilkan sesuai yang diinginkan atau data yang ingin ditampilkan.

Join table adalah penggabungan tabel-tabel menggunakan *Query* yang dilakukan melalui kolom/*key* tertentu yang memiliki nilai terkait untuk mendapatkan satu set data dengan informasi lengkap. Lengkap disini artinya kolom data didapatkan dari kolom-kolom hasil *join* antar tabel tersebut. Join diperlukan karena perancangan tabel pada sistem transaksional kebanyakan dinormalisasi, salah satu alasannya untuk mengurangi redundansi.

Jurnal Teknovasi
Volume 04, Nomor 01, 2017, 42-50
ISSN : 2355-701X

Pengaksesan data atau pencarian data dengan menggunakan *Query* atau *Join* pada database perlu memperhatikan ketepatgunaan implementasi dari data itu sendiri serta waktu prosesnya. Ada banyak cara yang dapat dilakukan oleh database manajemen sistem dalam memproses dan menghasilkan jawaban sebuah *query*. Semua cara pada akhirnya akan menghasilkan jawaban (*output*) yang sama tetapi pasti mempunyai harga yang berbeda-beda, seperti misalnya total waktu yang diperlukan untuk menjalankan sebuah *query*. Pencarian data atau pengolahan data tersebut dapat dilakukan dengan mengakses data yang terdapat dalam *database*. Pengaksesan *data* tersebut dilakukan dengan melakukan *query-query* pada basisdata- basisdata dengan *database* manajemen sistem. Kecepatan akses data dapat ditingkatkan dengan banyak cara, selain dari sisi perangkat kerasnya (*hardware*) dapat juga dilakukan dari sisi perangkat lunaknya (*software*) lebih khusus pada program aplikasinya.

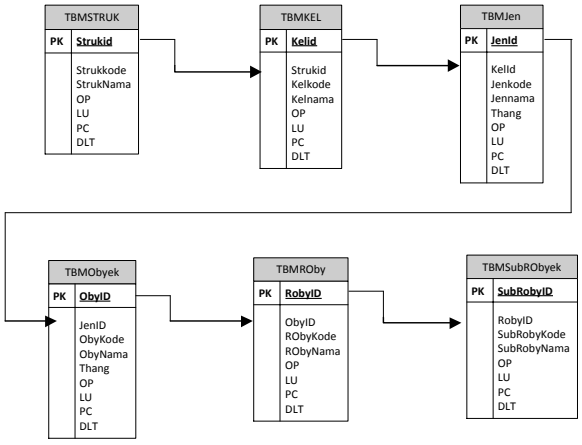
Algoritma query melakukan join ada beberapa algoritma yang sering digunakan yaitu Algoritma *Hash Join*, Algoritma *Nested Join*. Masing-masing algoritma memilik kegunaan yang sama dalam menggabungkan beberapa tabel atau relasi tabel pada database manajemen sistem.

Algoritma *Hash Join* adalah sebuah algoritma join untuk menggabungkan data yang berjumlah besar. Cara kerja *Hash Joins* adalah *Optimizer* membuat sebuah *Hash Table* berdasarkan predikat *Join*. Setiap tabel di *Inner* maupun *Outer* masing-masing dijadikan sebuah kode dengan *Hash Function* kemudian setiap kode *Hash* dari *Inner* akan dibandingkan dengan *Hash* Kode dari *Outer*. Apabila kode *hash* dari *Inner* dan *Outer* sama maka akan dilakukan proses pengecekan nilai dari kolom yang pada akhirnya akan dimasukkan ke dalam hasil jika nilai kolomnya sama.

Algoritma *Nested Join* adalah sebuah *Join* yang efektif jika *subset* yang digabungkan berjumlah sedikit dan jika kondisi dalam perintah *join* efisien untuk menggabungkan 2(dua) tabel tersebut.

METODE PENELITIAN

Sumber data untuk menjadi studi kasus dalam penelitian ini adalah master basis data Anggaran mempunyai 6 tabel yang saling berhubungan. Semua tabel tidak memiliki indeks atau *cluster* yang digunakan dalam pencarian data.



Gambar 1. Relasi tabel pengujian

Jurnal Teknovasi
Volume 04, Nomor 01, 2017, 42-50
ISSN : 2355-701X

Skenario Pengujian *Query*

Pengujian pencarian data menggunakan *query* dua algoritma yaitu *Query Hash Join dan Query Nested Join*. Pengujian *Query* dalam hal pencarian data berdasarkan banyak jumlah data untuk melakukan pencarian data dibagi atas beberapa tahap relasi dan group data. Masing-masing group mempunyai jumlah data yang berbeda . berikut group data sebagai uji coba dan berdasarkan jumlah data yang berbeda untuk penelitian berikut tahap relasi dan tabel group data:

Tabel 1 .Tahapan Relasi

No	Jumlah Relasi
1	1 Relasi
2	2 Relasi
4	3 Relasi
5	4 Relasi
6	5 Relasi

Tabel 2 . Tabel Group data Uji Penelitian

NO	Jumlah Data(Record)	Kelompok Data
1	10	Kecil
2	20	
3	40	
4	80	
5	160	
6	320	
7	640	Sedang
8	1280	
9	2560	
10	5120	
11	10240	
12	20480	
13	40960	Besar
14	81920	
15	163380	
16	655360	
17	1.310.720	

Jumlah record/baris data masing-masing tabel menggunakan data *dummy* dan penambahan data menggunakan perintah query untuk melakukan pengujian query algoritma. Berikut flowchart untuk menguji query pencarian data berdasarkan kriteria diatas.

Query Hash Join dan Nested Join

Query-query kedua algoritma yang akan diuji coba dalam penelitian ini dilakukan beberapa tahap relasi dan menghasilkan informasi yang sama. Untuk menghitung waktu yang diperlukan untuk menampilkan data yang dicari maka dibuat sebuah aplikasi menggunakan *Visual Basic.Net 2010* dan *Microsoft SQL Server 2008* sebagai Database Manajemen Sistem.

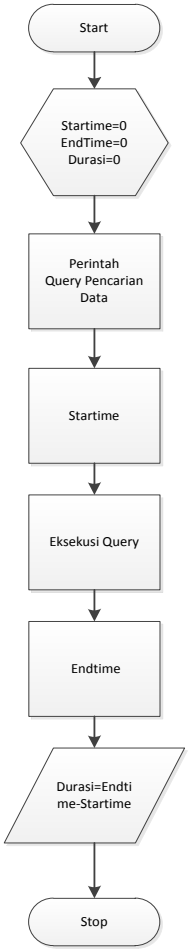
Paramter Pengujian Query

Parameter yang digunakan untuk menguji query-query diatas dalam pencarian data adalah kecepatan waktu mengeksekusi dan menampilkan informasi dalam pencarian data. Untuk menghitung waktu kecepatan yang diperlukan untuk mengakses data maka dibuat sebuah aplikasi. Satuan waktu untuk menghitung query-query diatas dalam satuan detik.

Alat Penelitian

Pada penelitian ini digunakan alat penelitian berupa perangkat keras dan perangkat lunak sebagai berikut:

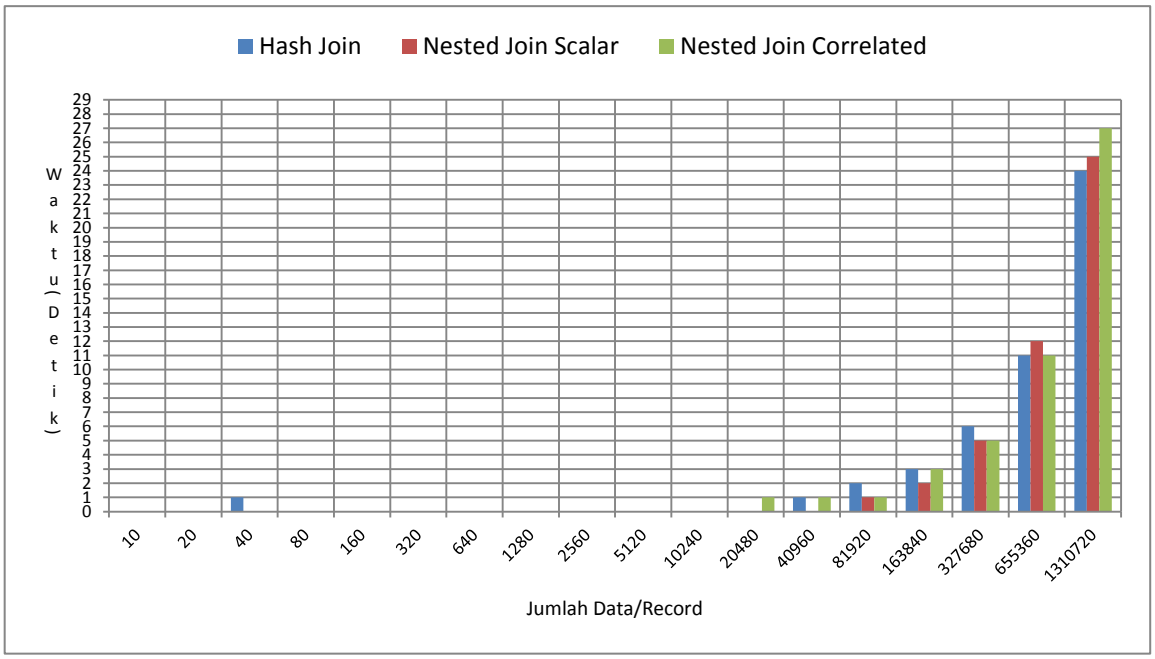
a. Perangkat keras
   1. *Processor* Intel Pentium Core Duo.
   2. RAM 2GB.
   3. *Harddisk* 500 GB.
   4. Monitor dengan resolusi 1024 x 768 *pixel* 32 *bit color.*
   5. *Mouse dan keyboard.*
b. Perangkat lunak
   1. Sistem Operasi Windows XP *service pack* 3.
   2. Visual Basic.Net 2010 dan Microsoft SQL Server 2008.



Gambar 2. Flowchart Skenario Pengujian Query

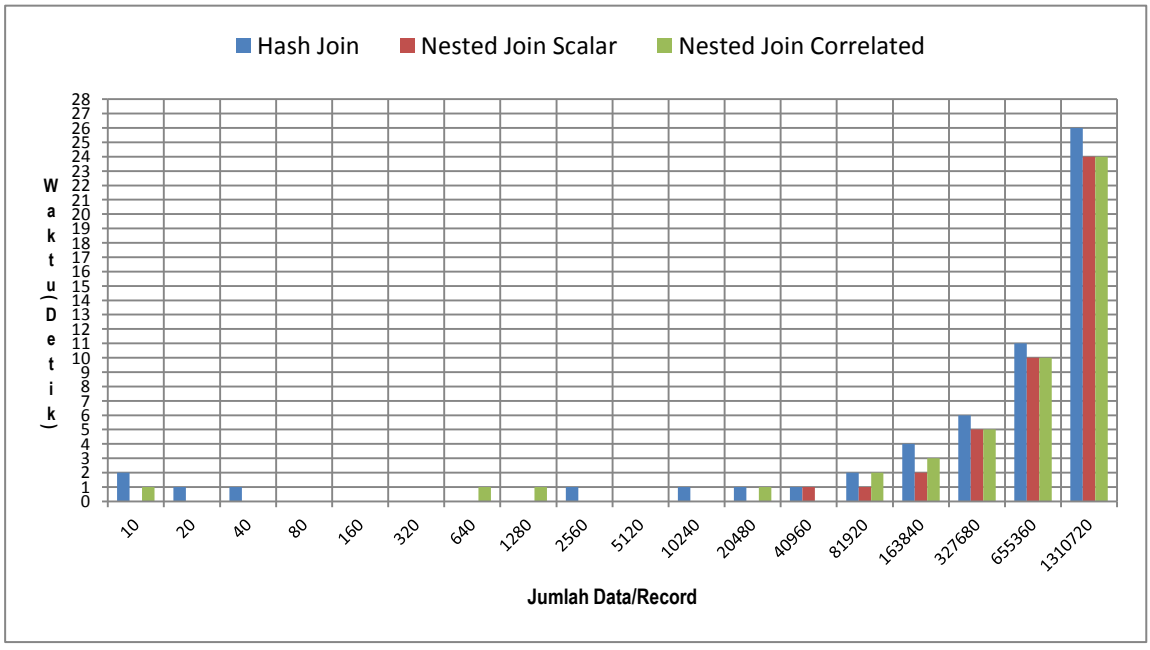
HASIL DAN PEMBAHASAN

Hasil Pengujian Query 1 Relasi



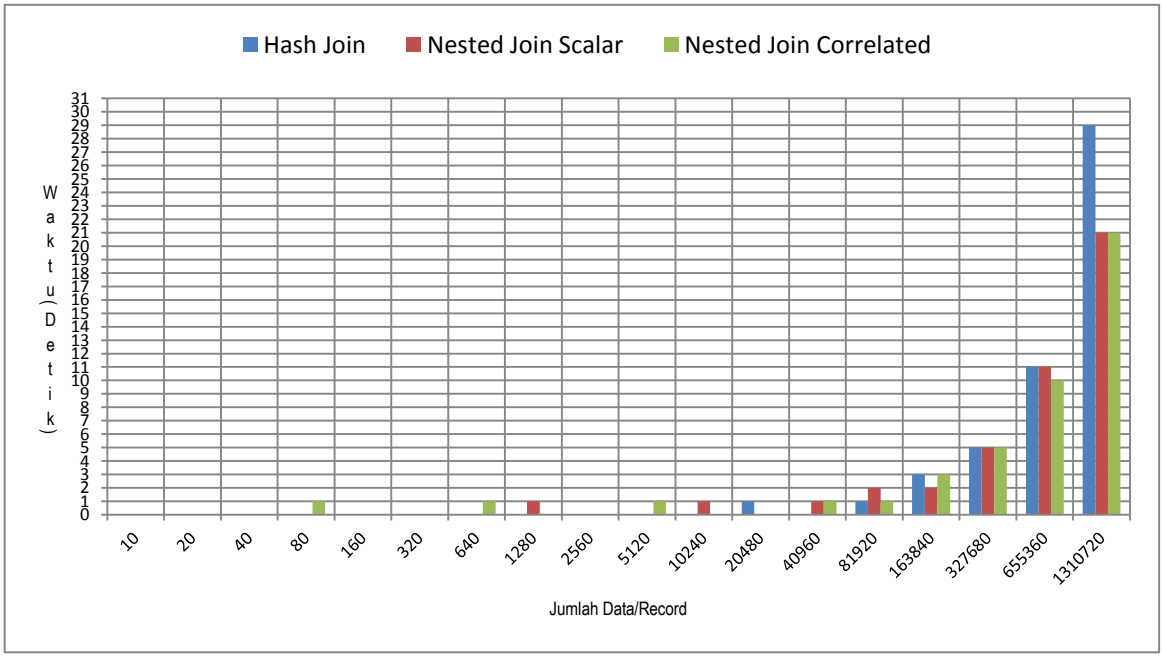
Gambar 3. Hasil Pengujian Query 1 Relasi

Hasil Pengujian Query 2 Relasi



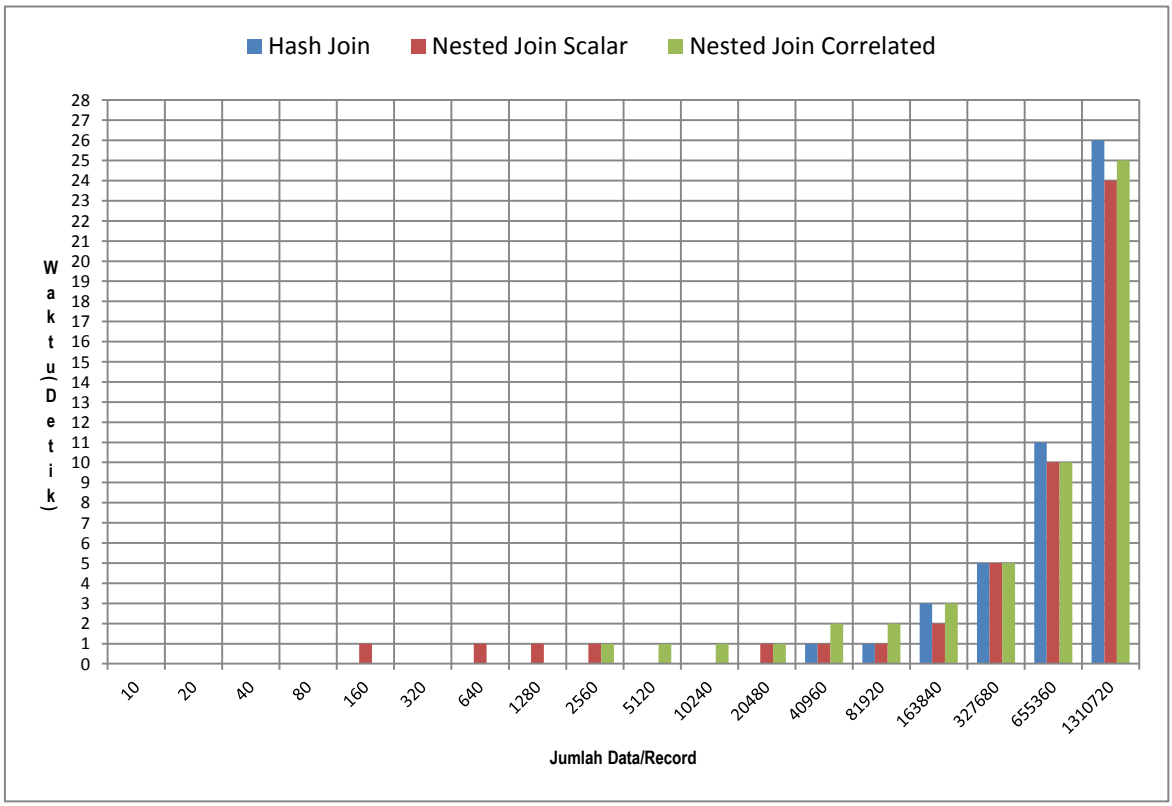
Gambar 4. Hasil Pengujian Query 2 Relasi

Hasil Pengujian Query 3 Relasi



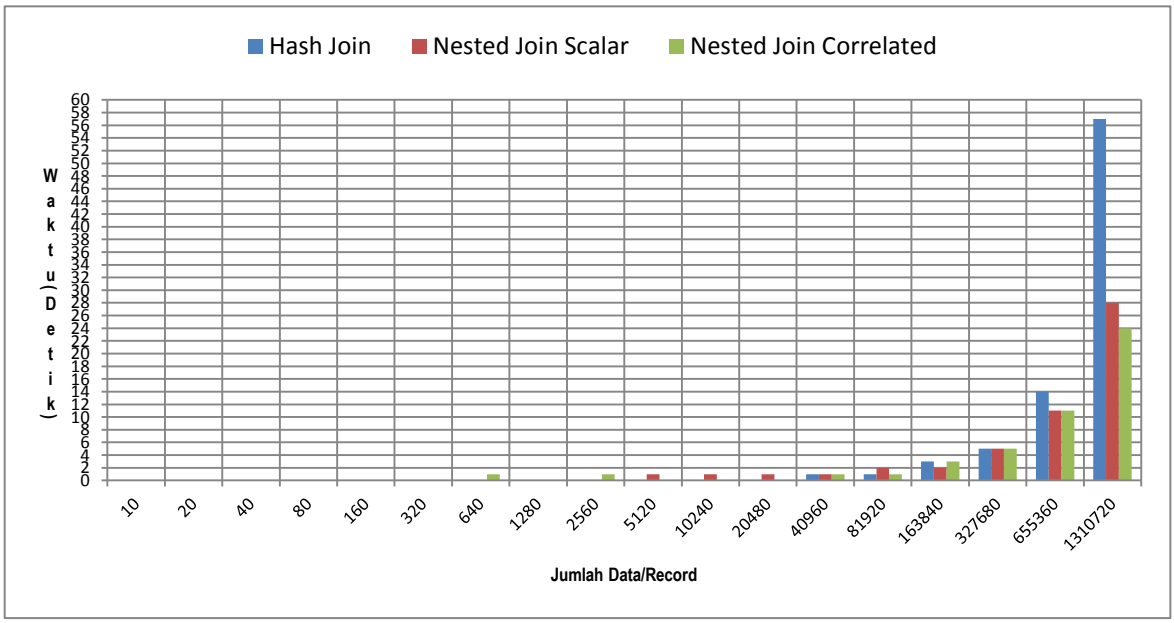
Gambar 5. Hasil Pengujian Query 3 Relasi

Hasil Pengujian Query 4 Relasi



Gambar 6. Hasil Pengujian Query 4 Relasi

Hasil Pengujian Query 5 Relasi



Gambar 7. Hasil Pengujian Query 5 Relasi

KESIMPULAN DAN SARAN

Query adalah perintah SQL yang mempunyai peran penting dalam database manajemen system (DBMS), namun penggunaan query haruslah tepat. Faktor yang mempengaruhi terhadap kecepatan akses data tidak hanya pada perintah *Query* saja akan tetapi terhadap hal-hal lain yang berpengaruh, seperti aplikasi, jaringan komputer, struktur fisik penyimpanan database, *hardware*, desain logik database, sistem operasi, *cluster* atau index *database*.

Pencarian data menggunakan *Query* atau perintah SQL banyak algoritma atau *query* bisa dilakukan dengan output yang sama. Beberapa algoritma *query* yang banyak digunakan yaitu *Query Hash Join* dan *Query Nested Join*.

Penggunaan *Query* bukan saja digunakan pada database manajemen sistem saja akan tetapi disebuah aplikasi dalam pemrosesan data atau pengolahan data menggunakan query akan lebih mudah dan cepat, akan setiap algoritma *query* memiliki *running time* atau kecepatan waktu melakukan respon data apalagi aplikasi berbasis *client server*.

Setelah penulis melakukan penelitian dengan pengujian *query Hash join* dan *Query Nested* dengan aplikasi yang dirancang menggunakan *Microsoft Visual Studi 2010* dan *Microsoft SQL Server 2008* sebagai database. Pengujian running time atau kecepatan waktu merespon data yang dilakukan untuk pemrosesan data khususnya pencarian data berdasarkan jumlah tabel yang direlasikan, jumlah baris/record maka penulis membuat kesimpulan sebagai berikut:

1. Kelompok data kecil jumlah data antara 10-320 baris *Query Hash join* lebih baik dibandingkan *Nested Join*.
2. Kelompok data besar jumlah data antara 40960-1310720 baris *query Nested join scalar* lebih baik dibandingkan *query Hash Join*.

Saran

Dari hasil penelitian penulis lakukan, terdapat beberapa saran yang disampaikan yaitu:

1. Pengujian untuk data lebih besar lagi dan jumlah relasi diperbanyak.
2. Pengembangan pengujian menggunakan index dan cluster.
3. Penelitian selanjutnya perlu dilakukan pengujian berbasis website.
4. Pengujian dengan algoritma yang berbeda.

Jurnal Teknovasi
Volume 04, Nomor 01, 2017, 42-50
ISSN : 2355-701X

DAFTAR PUSTAKA

Chanowich, E. & Sendelbach, E. (2001). Query Optimization in CSE 498G – Advanced Database.

D. A. Schneider, D. J. DeWitt, *A Performance Evaluation of Four Parallel Join Algorithms in a Shared-Nothing Multiprocessor Environment*, Proc. ACM SIGMOD Conference 1989, Portland, Oregon, pp. 110-121

D. J. DeWitt, R. H. Gerber, *Multiprocessor Hash-Based Join Algorithms*, Proc. 1985 VLDB, pp. 151-164

H. Zeller, *Parallelisierung von Anfragen auf komplexen Objekten durch Hash Joins* , Proc. GI/SI Fachtagung: Datenbanksysteme in Büro, Technik und Wissenschaft, IFB 204, Springer-Verlag (in German).

Hansjörg Zeller, Jim Gray, Hash Join Algorithms in a Multiuser Environment**,** TANDEM 19333 Vallco Parkway, LOC 3-05 Cupertino, CA, 95014

Immanuel Chan,2008, *Oracle Database Performance Tuning Guide, 10g Release 2 (10.2),* Redwood City, CA.

K. Bratbergsengen, *Hashing Methods and Relational Algebra Operations*, Proc. 10th VLDB, Aug. 1984, pp. 323-332

Korth, H.F., dan Silberschatz, A., 1991, Database System Concepts, McGraw Hill, Singapura.

Kusrini, 2006, *Optimasi Query Untuk Pencarian Data dengan Subset Query*, Bandung

L. D. Shapiro, *Join Processing in Large Database Systems with Large Main Memory*, ACM TODS 11,3 (Sept. 86), pp. 239-264

M. Kitsuregawa, H. Tanaka, T. Moto-oka, *Application of Hash to Data Base Machine and Its Architecture*, New Generation Computing 1, 1983, pp. 63-74

M. Nakayama, M. Kitsuregawa, M. Takagi, *Hash-Partitioned Join Method Using Dynamic Destaging Strategy*, Proc. 14th VLDB (1988), pp. 468-478

M. Pong, *NonStop SQL Optimizer: Query Optimization and User Influence*, Tandem Systems Review 4,2 (1988), pp. 22-38, Tandem Computers, Part. No. 13693)

M. Stonebraker et.al., *Parallelism in XPRS*, UC Berkeley, Electronics Research Laboratory, Report M89/16, February, 1989

Metta Santiputri, Mira Chandra Kirani, Anni, 2010, *Perbandingan Cross-Product Dan Subset Query Pada Multiple Relasi Dengan Metode Cost-Based*, Seminar Nasional Informatika UPN Veteran Yogyakarta.

Nielsen, Paul., Mike White, and Uttam Parui. *Microsoft SQL Server 2008 Bible*, Wiley Publishing, Inc., 2009

Patrick Valduriez, Georges Gardarin, *Join and Semijoin Algorithms for a Multiprocessor Database Machine*, ACM TODS 9,1 (1984), pp. 134-161.

R. H. Gerber, *Dataflow Query Processing Using Multiprocessor Hash-Partitioned Algorithms*, Dissertation, University of Wisconsin-Madison, Computer Sciences Technical Report #672, Oct. 86

Sagi Arsyad, 2008, *Pengenalan .NET dan C#, Microsoft Innovation Center*, Universitas Indonesia, Jakarta.

Sandra Cheevers, 2006,*Oracle Database Product Familiy, An Oracle White Paper*, Redwood Shores, CA USA, Agustus 2006, pp 3-4.

Setiawan, M.A., 2004, *Optimasi SQL Query untuk Informasi Retrievel pada Aplikasi Berbasis Web*, Proceedings Seminar Nasional Aplikasi Teknologi Informasi UII, Yogyakarta

Tandem Database Group, *NonStop SQL: A Distributed, High-Performance, High-Availability Implementation of SQL*, Proc. 2nd Int. Workshop on High Performance Transaction Systems, Asilomar, CA (Lecture Notes in Computer Science 359, Springer-Verlag, Ed.: D. Gawlick, A. Reuter, M. Haynie)

Tom Best dan M.J. Billings,2005,*Oracle Database 10g: Administration Workshop I, Electronic Presentation*, Redwood Shores, California USA, November 2005, pp. 41-42.