ANALISA KEEFEKTIFAN MYSQL *CLUSTER* DAN *NON-CLUSTER* DALAM MEMPROSES DATA AKADEMIK

Wanda Ichsanul Isra[1)], Taufik Fuadi Abidin[2)], Razief Perucha Fauzie Afidh[3)]
Data Mining and Information Retrieval Research Group[1,2,3)]
Jurusan Informatika, FMIPA, Universitas Syiah Kuala
Jl. Syech Abdurrauf No. 10, Kopelma Darussalam, Banda Aceh, 23111
E-mail: taufik.abidin@unsyiah.ac.id[2)]

Abstrak

Pertumbuhan data akademik dalam jumlah besar pada institusi perguruan tinggi tidak dapat dihindari lagi saat ini. Dengan semakin besar data akademik, pengelolaannya secara efektif dalam Database Management System (DBMS) menjadi kebutuhan. Salah satu teknologi DBMS dengan sistem ketersediaan tinggi adalah MySQL cluster. Dalam makalah ini, keefektifan dari MySQL cluster dan non-cluster dalam mengakses dan memproses data akademik dikaji. Analisa dilakukan dengan mengirimkan sejumlah query dalam bentuk Structured Query Language (SQL) secara bersamaan atau Queries per Second (QPS). Parameter yang dikaji adalah waktu respon dan waktu eksekusi. Hasilnya menunjukkan bahwa MySQL cluster lebih unggul bila dibanding MySQL non-cluster untuk kedua waktu tersebut.

Kata kunci: *MySQL cluster, data akademik, queries per second, waktu respon, waktu eksekusi.*

Abstract

The growth of academic data in higher education institutions can no longer be avoided nowadays. Large volume of academic data must be managed effectively in a Database Management System (DBMS). An example of DBMS with high system availability is MySQL cluster. In this paper, we analyze the effectiveness of MySQL cluster and non-cluster in accessing and processing academic data. The analysis is done by sending a number of Structured Query Language (SQL) statements at once, a.k.a Queries per Second (QPS). The parameters observed are response time and execution time. The results show that MySQL cluster outperforms non-cluster MySQL in both response and execution time.

Keywords: *MySQL cluster, academic data, queries per second, response time, execution time.*

1. PENDAHULUAN

Selama dua dekade terakhir, jumlah data di seluruh dunia terus meningkat. Pada tahun 2002, data dalam format digital diperkirakan telah mencapai lebih dari 5 EB (*ExaBytes*) atau setara dengan 5x100 GB (*GigaBytes*). Tercatat bahwa peningkatan data pada interval tahun 1999-2002 secara berurutan adalah 2,1 EB, 3,2 EB, 3,4 EB, dan 5,4 EB. Jumlah data diperkirakan akan terus meningkat [1].

Perguruan tinggi mengelola data akademik dalam jumlah yang terus bertambah seiring dengan bertambahnya mahasiswa yang diterimanya setiap tahun. Perguruan tinggi seperti Universitas Gadjah Mada, Institut Teknologi Bandung, Universitas Indonesia, Institut Pertanian Bogor, Institut Teknologi Sepuluh Nopember, dan Universitas Syiah Kuala mengelola lebih dari 30 ribu mahasiswa. Jika setiap mahasiswa strata 1 (S1) aktif rata-rata mengambil lebih kurang 16 mata kuliah dalam setahun atau dua semester, maka jumlah *record* yang harus dikelola berjumlah hampir mencapai 500 ribu. Dalam 10 tahun, jumlah *record* akademik akan mencapai lebih dari 5 juta. Jumlah itu belum termasuk data-data lainnya yang juga dikelola dan merupakan bagian dari data akademik.

Data dalam jumlah besar perlu dikelola dengan baik menggunakan sistem pengelolaan *database* (DBMS) yang berbasis *cluster* dan tidak hanya dikelola menggunakan DBMS tunggal (*non-cluster*) [2]. Namun keefektifan DBMS berbasis *cluster* harus dikaji menggunakan beberapa komponen, seperti ukuran dataset, *query* yang dieksekusi, dan waktu rata-rata yang dibutuhkan oleh DBMS untuk merespon dan mengeksekusi sebuah *query*. Menggunakan *dataset* yang relatif besar dan jenis *query* yang rumit dapat menjadi tolak ukur keefektifan dari suatu DBMS [3].

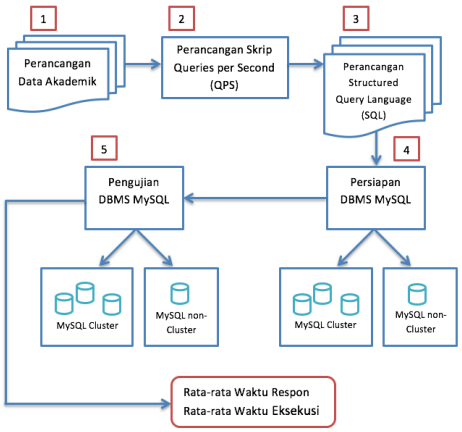
Cluster dalam basis data bermakna kumpulan dari dua atau lebih komputer yang disatukan menjadi *server* yang independen untuk menjalankan suatu fungsi secara terorganisir [4]. Kumpulan komputer tersebut terhubung satu sama lain dalam sebuah jaringan internet yang transparan dari pengguna. Artinya, kumpulan komputer tersebut berada dalam satu kesatuan sistem sehingga pengguna merasa menggunakan layanan DBMS tunggal. Dengan metode *cluster*, kegagalan DBMS dapat diminimalkan. *Cluster* dapat diterapkan menggunakan metode *shared-nothing* dan *shared-disk* [4]. Metode *shared-nothing* digunakan karena adanya keterbatasan dari metode *shared-disk*. Dengan metode ini, setiap *server* memiliki media penyimpanan (*disk*) masing-masing yang dikelola dengan baik. DBMS yang menggunakan metode ini adalah *MySQL cluster* [4]. Sementara, metode *shared-disk* memanfaatkan media penyimpanan secara bersama-sama. Penggunaan bersama media penyimpanan dikelola oleh *lock manager* untuk mengatur setiap *node* pada setiap transaksi. *Lock manager* memastikan bahwa *node* siap dan tidak membatalkan tugas yang diterima dari *node* lain [4].

Salah satu DBMS yang populer adalah *MySQL cluster* yang bersifat terbuka dan memiliki skalabilitas yang tinggi [5]. Secara umum, *MySQL cluster* disebut dengan *Network DataBase* (NDB) dan menggunakan media penyimpanan yang dikenal dengan istilah NDBCLUSTER [6]. *MySQL cluster* memiliki tiga jenis *node* yaitu *node* penyimpanan, *node* pengelolaan, dan *node MySQL server* [5]. *Node* penyimpanan merupakan *node* utama yang digunakan untuk menyimpan data. Secara otomatis, data direplikasi pada setiap *node* penyimpanan untuk mengantisipasi jika terjadi kegagalan pada *node* yang lain. *Node* pengelolaan mengatur konfigurasi sistem dan penggunaan beberapa *node* [5], sementara *node MySQL server* mengatur cara mengakses *node* penyimpanan. Jumlah *node* minimum yang disarankan adalah satu *node* pengelolaan, dua *node* penyimpanan, dan dua *node MySQL server* [5].

Kajian terkait *MySQL cluster* masih terus dilakukan hingga saat ini. Salah satunya yaitu kajian yang menerapkan metode *load balancing* untuk meningkatkan kinerja dari *MySQL cluster* [7]. Kajian tersebut membuktikan bahwa *MySQL cluster load balancing* memiliki kinerja lebih baik bila dibanding dengan *MySQL cluster default*. Secara khusus, kajian tersebut hanya melakukan analisa terhadap kinerja *MySQL cluster*, namun tidak halnya dengan *MySQL non-cluster*. Dalam makalah ini, keefektifan *MySQL cluster* dan *non-cluster* dalam memproses data akademik dianalisa. Parameter yang diamati adalah rata-rata waktu respon dan waktu eksekusi.

2. METODE PENELITIAN

Secara umum, ada 5 tahapan yang dilakukan dalam kajian ini. Urutan dari setiap tahapan diilustrasikan pada Gambar 1. Tahapan pertama adalah membuat data *dummy* sistem akademik. Hal ini dilakukan karena, demi keamanan dan menjaga privasi data mahasiswa, data akademik yang asli tidak digunakan. Data *dummy* merupakan data relasional yang dibangun mengacu pada skema data akademik secara umum. Jumlah data *dummy* yang dibangun dirangkum pada Tabel 1.



Gambar 1. Skema Metode Penelitian

Tahapan kedua adalah merancang skrip *Queries per Second* (QPS) untuk menguji kecepatan transaksi basis data per detik [4] yang meliputi pernyataan *SELECT*, *INSERT*, *UPDATE*, dan *DELETE* [8]. Skrip ini digunakan untuk menguji keefektifan *MySQL cluster* dan *non-cluster* dalam merespon *query* yang dikirim secara bersamaan melalui *thread* [9].

Tabel 1. Jumlah Record untuk Setiap Tabel

Nama Tabel	Jumlah Record
Mahasiswa	800.000
OrangTua	800.000
MataKuliah	11.000
Dosen	800.000
Jurusan	230
Nilai	800.000
Fakultas	12
Universitas	1

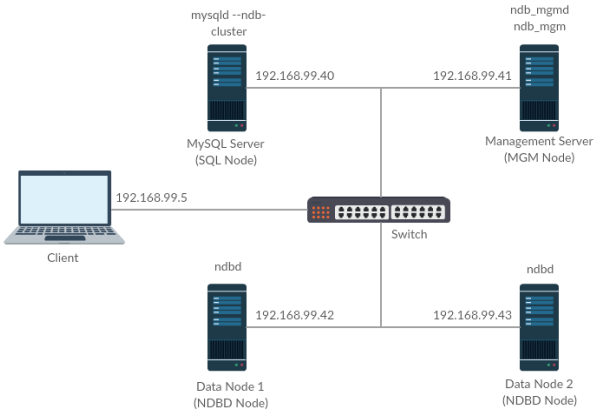
Tahapan ketiga adalah merancang *query* (SQL) untuk pengujian. Ada 3 jenis *query* yang digunakan untuk pengujian yaitu *query* ringan (*simple*), sedang (*moderate*), dan berat (*expensive*). *Query* ringan merupakan *query* sederhana yang hanya menggunakan perintah SELECT pada tabel tunggal dan dapat dieksekusi dengan sangat cepat [10]. *Query* sedang adalah *query* yang membutuhkan waktu eksekusi lebih lama bila dibanding dengan *query* sederhana. *Query* sedang melibatkan beberapa tabel dan menggunakan perintah JOIN [10]. *Query* berat adalah *query* yang rumit dan membutuhkan waktu yang lebih lama untuk dieksekusi. *Query* berat melibatkan dua atau lebih tabel dan menggunakan gabungan perintah JOIN, GROUP BY, *query* bersarang (*nested query*), dan beberapa fungsi agregasi standar lainnya [10]. Sebagai ilustrasi, contoh dari ketiga jenis *query* tersebut dirangkum pada Tabel 2.

Tabel 2. Contoh Query Pengujian

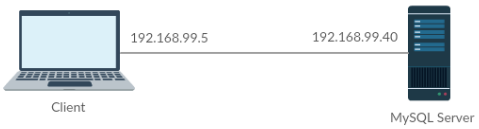
Query	Jenis
SELECT SQL_NO_CACHE * FROM Mahasiswa	Ringan
SELECT SQL_NO_CACHE m.NPM, m.Nama AS Nama_mhs, d.NIP, d.Nama AS Nama_doswal FROM Mahasiswa AS m INNER JOIN Dosen AS d ON m.Doswal_NIP = d.NIP	Sedang
SELECT SQL_NO_CACHE g.NPM, s.Nama AS Nama_mhs, s.Doswal_NIP, d.Nama AS Nama_doswal, SUM(Bobot) AS Total_bobot, SUM (Bobot/144) as IPK FROM Nilai AS g INNER JOIN Mahasiswa AS s ON g.NPM = s.NPM INNER JOIN Dosen AS d ON s.Doswal_NIP = d.NIP WHERE g.semester BETWEEN CONCAT('20', SUBSTR(g.NPM, 1, 2)) AND CONVERT(CONVERT(CONCAT('20', SUBSTR(g.NPM, 1, 2)), SIGNED INTEGER) + 2, CHAR(5)) GROUP BY NPM HAVING IPK < 2	Berat

Tahapan keempat adalah melakukan perancangan topologi jaringan untuk setiap jenis teknologi MySQL dan pengaturan perangkat yang digunakan untuk *MySQL cluster* dan *non-cluster*. Kedua bentuk MySQL tersebut dikonfigurasi menggunakan komputer dengan spesifikasi prosesor Intel® CoreTM i3 CPU 550 3.20GHz, memori 2048MB, Linux Ubuntu versi 14.04 yang digunakan sebagai sistem operasi, dan switch D-Link 1008D yang digunakan untuk mengatur konektifitas antar perangkat. Khusus untuk mewujudkan *MySQL cluster*, 4 komputer dengan spesifikasi yang sama digunakan dan dirangkai dalam topologi seperti yang diperlihatkan pada Gambar 2. Keempat komputer tersebut memiliki spesifikasi yang sama dengan *MySQL non-cluster* dan dirangkai menggunakan komputer berbeda dari *MySQL non-cluster*. *MySQL cluster* juga dirangkai menggunakan mesin penyimpanan dan konfigurasi bawaan yaitu NDBCLUSTER. Sementara untuk *MySQL non-cluster*, hanya digunakan sebuah *server* dengan topologi sederhana seperti yang diperlihatkan pada Gambar 3.

Tahapan kelima adalah pengujian. Pada tahapan ini, *query* dikirim secara simultan menggunakan skrip QPS yang telah dibuat pada tahap kedua. Pengujian ini dilakukan sebanyak 5 kali untuk setiap *query* dan menggunakan salah satu *index* umum dari mesin penyimpanan MySQL yaitu InnoDB. Jumlah *query* yang dikirim secara simultan masing-masing 2, 4, 6, 8, dan 10. Waktu yang diamati adalah waktu rata-rata DBMS merespon dan waktu rata-rata *query* dieksekusi. Contoh keluaran dari hasil pengujian terhadap *MySQL cluster* dan *non-cluster* diperlihatkan pada Gambar 4 dan 5.



Gambar 2. Topologi MySQL Cluster



Gambar 3. Topologi MySQL non-Cluster

```
~/Code/ta perl mysql-qps.pl --user=root --host=localhost --port=3306 --db=academic_db --clients=2 --query=="SELECT SQL_NO_CACHE * FROM students"

Your arguments input:
  User          root
  Host          localhost
  Port          3306
  Db            academic_db
  Clients       2
  Query         SELECT SQL_NO_CACHE * FROM students

  Db password:

Thread is supported in this version of perl

Starting execution of 2 clients
Time throwing the query: 11:22:12

Client 2 starting       => Response time: 11:22:13      => (0 min 1 sec)
Client 2 done           => Finished time: 11:22:14      => (0 min 1 sec)

Client 1 starting       => Response time: 11:22:13      => (0 min 1 sec)
Client 1 done           => Finished time: 11:22:14      => (0 min 1 sec)

All clients run to completion

Total Clients           => 2
Response time avg       => 0 min 1 sec
Finished time avg       => 0 min 1 sec
```

Gambar 4. Keluaran Pengujian 2 QPS query ringan MySQL cluster

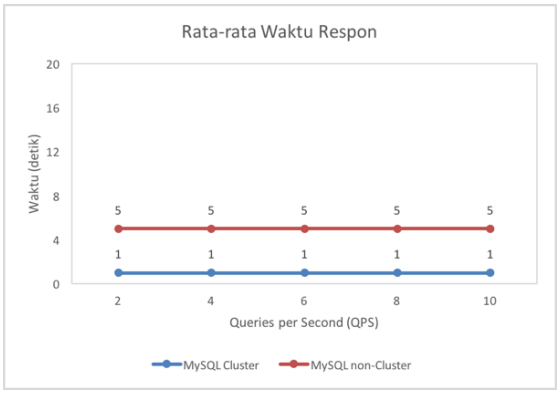
```
~/Code/ta perl mysql-qps.pl --user=root --host=localhost --port=3306 --db=academic_db --clients=2 --query=="SELECT SQL_NO_CACHE * FROM students"

Your arguments input:
  User          root
  Host          localhost
  Port          3306
  Db            academic_db
  Clients       2
  Query         SELECT SQL_NO_CACHE * FROM students

  Db password:

Thread is supported in this version of perl

Starting execution of 2 clients
Time throwing the query: 9:7:39

Client 2 starting       => Response time: 9:7:44        => (0 min 5 sec)
Client 2 done           => Finished time: 9:7:50        => (0 min 6 sec)

Client 1 starting       => Response time: 9:7:44        => (0 min 5 sec)
Client 1 done           => Finished time: 9:7:50        => (0 min 6 sec)

All clients run to completion

Total Clients           => 2
Response time avg       => 0 min 5 sec
Finished time avg       => 0 min 6 sec
```

Gambar 5. Keluaran Pengujian 2 QPS query ringan MySQL non-cluster

3. HASIL DAN PEMBAHASAN

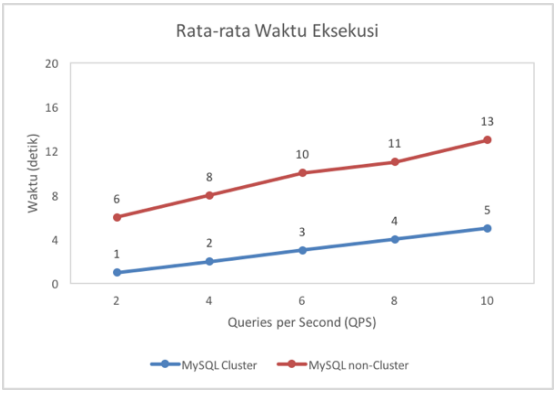
Pengujian dilakukan dengan mengamati waktu rata-rata DBMS merespon dan waktu rata-rata *query* dieksekusi menggunakan skrip QPS secara simultan. Pengujian parameter pertama tidak dilakukan pada ketiga jenis *query* dikarenakan waktu respon tidak terkait langsung dengan jenis *query*. Sementara, jenis *query* digunakan untuk menguji waktu respon *MySQL cluster* dan *non-cluster*.

Pengujian waktu respon diperlihatkan pada Gambar 6. Hasil memperlihatkan bahwa *MySQL cluster* dan *non-cluster* berhasil merespon dengan waktu yang relatif konstan, walaupun terlihat nyata bahwa *MySQL cluster* merespon lebih cepat bila dibanding dengan *MySQL non-cluster*. Perbedaan waktu respon antara kedua jenis DBMS tersebut adalah lebih kurang 4 detik.

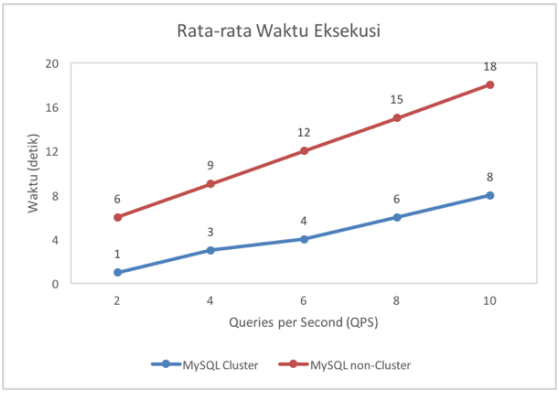


Gambar 6. Rata-rata Waktu Respon

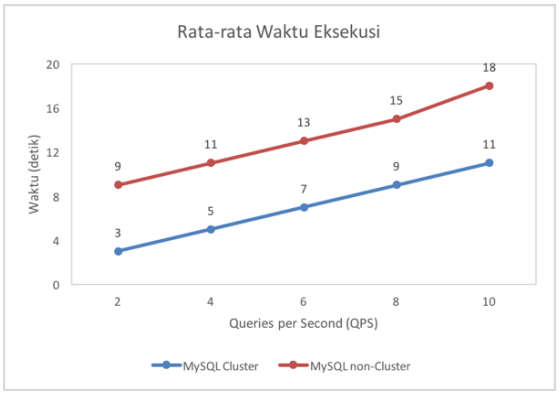
Pengujian waktu eksekusi dilakukan untuk setiap jenis *query*. Gambar 7 (a) memperlihatkan waktu eksekusi untuk *query* ringan. Terlihat bahwa *MySQL cluster* lebih cepat bila dibanding dengan *MySQL non-cluster*. Waktu eksekusi adalah linear dengan tren semakin banyak *query* yang dikirimkan secara bersamaan semakin banyak pula waktu yang dibutuhkan oleh MySQL untuk mengeksekusinya walaupun jumlah waktunya relatif sama untuk setiap *query*. Gambar 7 (b) memperlihatkan rata-rata waktu eksekusi untuk *query* sedang. Tren rata-rata waktu eksekusi pada *query* sedang ini juga mirip dengan waktu eksekusi *query* ringan. Terlihat bahwa *MySQL cluster* lebih cepat dari *MySQL non-cluster* dan waktu eksekusi adalah linier. Gambar 7 (c) memperlihatkan tren waktu eksekusi menggunakan *query* berat, yaitu *query* yang mengandung perintah join, group by, pernyataan bersarang, dan fungsi-fungsi aggrigasi lainnya. Walaupun waktu eksekusi lebih lama bila dibanding mengeksekusi *query* ringan dan sedang, namun *MySQL cluster* tetap lebih unggul bila dibanding dengan *MySQL non-cluster*.



(a) Query Ringan



(b) Query Sedang



(c) Query Berat

Gambar 7. Rata-rata Waktu Eksekusi untuk Setiap Jenis Query

4. KESIMPULAN

Dari beberapa hasil pengujian dapat disimpulkan bahwa *MySQL cluster* merespon lebih cepat bila dibanding dengan *MySQL non-cluster*. Rata-rata waktu respon relatif konstan sementara rata-rata waktu eksekusi adalah linear terhadap jumlah *query* yang dikirim per detik secara simultan. Hasil pengujian menunjukkan bahwa *MySQL cluster* lebih efektif bila dibanding dengan teknologi *MySQL non-cluster* untuk *query* ringan, sedang, maupun berat.

Pada kajian ini, jumlah QPS yang diuji masih tergolong rendah, yaitu berkisar dari 2 sampai 10. Dalam analisa yang dilakukan, jumlah QPS tidak dapat dibesarkan lagi karena memori mesin yang digunakan sangat terbatas. Disarankan, pada kajian selanjutnya jumlah QPS diperbesar dan jumlah *record* diperbesar mencapai puluhan juta.

5. DAFTAR RUJUKAN

[1] Lyman, P., dan Varian, H., *How Much Information 2003* [Online] (Diupdate 30 Okt 2003) http://www2.sims.berkeley.edu/research/projects/how-much-info-2003/printable_report.pdf. [diakses tanggal 10 Oktober 2015].

[2] Fisher, D., DeLine, R., Czerwinski, dan M. Drucker S. Interactions with big data analytics. *Interactions*, 19 (3), pp.50–9, 2012.

[3] Fuad, A., Erwin, A., dan Ipung, H. P. Processing Performance on Apache Pig, Apache Hive and MySQL Cluster. In: ICTS (Information, Communication Technology and System), *Int. Conference on Information, Communication Technology and System*. Surabaya, Indonesia, 24 September 2014.

[4] Davies, A., and Fisk, H., *MySQL Clustering*. 1st ed. New York: MySQL Press, 2006.

[5] Davies, A., *High Availibility MySQL Cookbook*. 1st edition, Birmingham: Packt Publishing, 2010.

[6] Ronstrom, M., and Thalmann, L., *MySQL Cluster Architecture Overview: High Availability Features of MySQL Cluster*, http://wiki.jokeru.ro/wp-content/uploads/2013/ 10/mysql-cluster-technical-whitepaper.pdf. [Diakses 16 November 2015].

[7] Gani, T. A., Arafat, A., Melinda, 2015, Analisis Kinerja MySQL Cluster Menggunakan Metode Load Balancing, *Jurnal Rekayasa Elektrika*, 11 (4), pp.129-134.

[8] Coronel, C., Morris, S., Rob, P., *Database Systems: Design, Implementation, and Management*. 10th ed. Stamford: Cengage Learning, 2012.

[9] Sugalski, D., 2015. *Tutorial on threads in Perl*, http://perldoc.perl.org/perlthrtut.pdf. [Diakses 10 Oktober 2015].

[10] Nugroho, A., *Perancangan dan Implementasi Sistem Basis Data*. 1st ed. Yogyakarta: Andi, 2011.