KONVERSI DATA DARI DATABASE RELASIONAL MYSQL KE DATABASE NOSQL MONGODB (STUDI KASUS PADA SISTEM INFORMASI UMKM)

Erma Susanti[1)], Muhammad Sholeh[2)], Rr. Yuliana Rachmawati[3)]
[1),2),3)]FTI, Institut Sains & Teknologi AKPRIND Yogyakarta
Email: erma@akprind.ac.id, muhash@akprind.ac.id, yuliana@akprind.ac.id

Abstract

Increasing volume of data that stored in a relational database can cause a reduction in data access speed. The addition of data and database queries can affects the speed of the performance of the information system. The use of document-oriented, non-relational (NoSQL) databases such as MongoDB can help in grouping data logically. Documents and collections do not need to be determined well before making, because it is very flexible. Built-in features of MongoDB have high scalability and are compatible with a variety of software. Data conversion is a very scalable solution, because data will be stored in cloud-based storage, thus saving storage space and server maintenance costs. The latest MongoDB feature also supports multi document ACID transactions to support data integrity. Data conversion case study will use data from the UMKM information system (Usaha Mikro Kecil Menengah) that has been developed previously with a MySQL relational database. The UMKM information system database will support a new database based on NoSQL MongoDB, in order to improve access performance, increase scalability, save on server maintenance costs and help strengthen data integrity. MongoDB database performance test results using the YCSB Client with an increased number of threads affect the linear increase in throughput (ops / sec) and the amount of data.
Keywords: Database, NoSQL, Non-relational, MongoDB, UMKM

Pendahuluan

Latar Belakang

Database (basis data) merupakan koleksi informasi yang terorganisasi sehingga memudahkan untuk diakses, dikelola, dan di-*update*. Jenis *database* secara umum dibagi menjadi *relational database* dan *non-relational*. *Relational database* biasanya berisi data terstruktur, sedangkan *non-relational database* berisi data semi terstruktur (Damodaran et al., 2016). *Database* didefinisikan sebagai kumpulan data dan sistem yang menangani data, transaksi, masalah *database* adalah DBMS (*Database Management System*). *Relational Database* antara lain MySQL, PostgreSQL, SQL Server, SQLite, Oracle dan lain-lain. *Non-relational database* antara lain Neo4j, MongoDB, Cassandra, Redis, Firebase, RethinkDB, HBase/Hadoop, CouchDB, Hipertable dan lain-lain. Penggunaan *relational database* berbasis SQL pada umumnya memiliki skema *database* yang kaku, sulit dalam membuat *query* untuk tabel dan relasi yang kompleks, dan menyulitkan pada saat akan memperbesar skalanya. Sistem RDBMS menawarkan konsistensi data,

sedangkan NoSQL menawarkan skalabilitas yang lebih tinggi yaitu dapat berjalan lebih cepat dan mendukung beban yang lebih besar. NoSQL (*Not Only SQL*) merupakan solusi untuk mengatasi kekakuan pada *relational database*. NoSQL memiliki skema yang tidak terstruktur dan memiliki *query* yang tidak rumit sehingga direkomendasikan untuk dilakukan denormalisasi. NoSQL dirancang untuk model *database* terdistribusi untuk skala *cloud* (komputasi awan). Karakteristik NoSQL yaitu dapat menangani volume data yang besar, replikasi dan distribusi secara *scalable*, tanpa skema yang kaku, format data yang fleksibel dapat diubah kapan pun dan *open source* (Mearaj, 2018).

MongoDB merupakan jenis *database* berbasis dokumen NoSQL dengan format JSON. Data pada *relational database* berbentuk SQL disimpan dalam bentuk tabel-tabel, sedangkan data pada MongoDB disimpan menggunakan format JSON yang berbentuk dokumen. MongoDB adalah produk di antara *relational database* dan *database* NoSQL yang menggunakan teknologi terdistribusi. MongoDB hanya menyimpan data tanpa mekanisme yang eksplisit dan terstruktur untuk menghubungkan data dari *bucket* yang berbeda. MongoDB memiliki kelebihan pada ukuran data yang besar, fleksibilitas, dan transportasi. MongoDB juga merupakan *realtime database* (Khedkar and Thube, 2017). Beberapa jenis *realtime database* antara lain Firebase, MongoDB, GraphDB, OrientDB, RethinkDB, dan lain-lain. MongoDB adalah *database* yang dikembangkan oleh MongoDB Inc dan bersifat *open source*. Pada MongoDB data disimpan dalam bentuk format dokumen JSON. Informasi yang berelasi dapat disimpan bersama-sama dengan akses *query* secara cepat melalui bahasa *query* MongoDB. MongoDB menggunakan skema dinamik yang membantu untuk membuat *record* tanpa mendefinisikan struktur, seperti atribut atau tipe data. Hal ini juga dimungkinkan untuk mengubah struktur *record* secara sederhana dengan menambahkan atribut baru atau menghapus *field* yang telah ada. Model seperti ini membantu untuk merepresentasikan hierarki *relationship*, untuk menyimpan *array*, dan struktur kompleks lainnya secara mudah. Dokumen dalam *record* tidak mempunyai pengaturan identik dalam *field*-nya. MongoDB didesain dengan *high availability* dan *scalability* termasuk replikasi.

MongoDB merupakan *database* yang efisien untuk digunakan pada aplikasi dengan *database* NoSQL yang berorientasi dokumen. MongoDB adalah *cross-platform*, *database open source* yang menggunakan model data berorientasi dokumen. MongoDB menerapkan konsep koleksi dan dokumen, dimana koleksi adalah *group* dari dokumen MongoDB atau ekuivalen dengan tabel RDBMS. Koleksi yang ada menggunakan *database single*. Sedangkan konsep dokumen yaitu sekumpulan dari pasangan *key-value*. Skema dinamik dimiliki oleh dokumen. Skema dinamik memiliki arti bahwa dokumen yang terdapat pada koleksi yang sama tidak perlu memiliki struktur atau sekumpulan *field*, dan pada umumnya *field* dalam ada koleksi dokumen memiliki tipe data yang berbeda. Ada beberapa kemungkinan untuk menggunakan MongoDB sebagai *database graph* yang efisien antara lain Neo4j dan MongoDB atau menggunakan GraphQL dan MongoDB (Parmar and Roy, 2018).

Permasalahan

Peningkatan volume data pada *relational database* seperti MySQL dapat menyebabkan berkurangnya kecepatan akses data. Adanya pertambahan data dan *query database* pada suatu sistem informasi mempengaruhi kecepatan performa dari sistem informasi. Perlu adanya solusi dengan melakukan proses konversi data dari *relational*

database ke *non-relational database* (NoSQL), sehingga dapat menghemat ruang penyimpanan dan meningkatkan kecepatan akses data.

Tujuan Penulisan

Tujuan penelitian ini adalah untuk melakukan konversi data dari *relational database* ke *database* NoSQL MongoDB. Studi kasus akan menggunakan data dari sistem informasi UMKM (Usaha Mikro Kecil Menengah) yang telah dikembangkan sebelumnya dengan menggunakan *relational database* MySQL. Pengujian performa *throughput* pada operasi *insert*, *read* dan *update* dilakukan menggunakan *tools* YCSB (*Yahoo Cloud Serving Benchmark*).

Landasan Teori

Sebelum dilakukan konversi *database* ke MongoDB, ada baiknya untuk mengetahui kinerja beberapa *database* NoSQL. Perbandingan kinerja antara MySQL dan MongoDB pernah diteliti oleh Damodaran, et al (2016) dengan studi kasus pada platform aplikasi supermarket; Andersson and Berggren (2017); Deari et al. (2018). Evaluasi performa atau kinerja yang dilakukan terkait dengan *execution time* (waktu eksekusi). Hasilnya saat jumlah *record* yang dientrikan atau dicari jumlahnya kecil, maka tidak ada perbedaan waktu yang berarti untuk setiap operasi yang dilakukan antara MySQL dan MongoDB. Sedangkan pada saat jumlah *record* yang dientrikan meningkat, maka MongoDB menunjukkan pengurangan waktu eksekusi yang signifikan bila dibandingkan dengan MySQL. Jadi, dalam hal waktu eksekusi MongoDB memiliki performa yang lebih baik saat terjadi peningkatan jumlah *record*. Hasil lainnya untuk operasi CRUD saat bekerja dengan data yang besar dan memiliki konkuren akses memiliki performa yang lebih baik dari MySQL.

Pengetahuan tentang kinerja antara NoSQL *database* dalam *mode pseudo distributed* antara Cassandra, MongoDB dan Redis diteliti oleh Kumarasinghe et al (2015). Pengetahuan ini penting untuk pemilihan *database* untuk aplikasi tertentu. Hasil eksperimen menunjukkan hasil dalam mode *pseudo distributed* maka Redis memiliki performa yang lebih baik untuk aplikasi yang tidak membutuhkan *range query*. Developer lebih memilih Redis daripada Cassandra dan MongoDB untuk aplikasi yang mempunyai persyaratan. Analisis lebih lanjut masih perlu dilakukan untuk performa pada jumlah *record* yang besar dan operasi untuk *benchmark database*. Untuk *benchmark* dapat menggunakan YCSB (*Yahoo Cloud Serving Benchmark*).

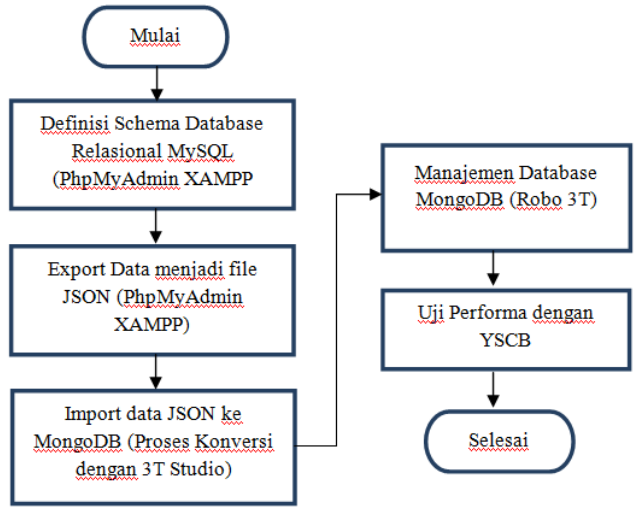
Stanescu et al. (2019) melakukan eksperimen tentang pengukuran performa untuk menghitung skalabilitas dan performa antara MongoDB dan SQL Server pada aplikasi medis. Prototipe lengkap aplikasi didesain menggunakan dua tipe *database* untuk *server handling* yaitu menggunakan *database* SQL Server dan *non relational database* MongoDB. Informasi yang digunakan untuk pengujian berdasarkan pada *medical record* yang disediakan oleh *Emergency Country Hospital*, Buzau. Hasilnya menunjukkan bahwa MongoDB memiliki performa yang lebih baik pada *database* skala besar daripada SQL Server.

Riset tentang BigData dengan data terstruktur dan tidak terstruktur dapat dilakukan dengan menggunakan MongoDB dan Hadoop (Saparkhojayev et al., 2018). Riset tentang BigData lainnya menggunakan MapReduce sebagai *framework* yang didesain untuk menyelesaikan permasalahan tentang data yang besar (Ajdari and Kasami,

2018). Penggunaan *Database as a Service* untuk data yang heterogen dapat menggunakan Neo4j dan MongoDB karena sangat membantu untuk menerima data berorientasi dokumen dan data grafis melalui aplikasi web. Penyimpanan data yang besar dalam *database* di dalam *data center* ditempatkan pada lingkungan *cloud* (Thillainayaki and Hemalatha, 2017). Penelitian tentang *database* NoSQL lainnya menggunakan Neo4j pernah dilakukan oleh (Zemzans, 2016). Neo4j merupakan *database graph* yang dibuat oleh Neo Technology. Neo4j ditulis menggunakan bahasa Java. Beberapa hal yang sangat menarik adalah pada penggunaan *endpoint* HTTP untuk menerima data transaksi. Ini sangat penting bagi developer yang familiar dengan HTTP. Neo4j menggunakan bahasa *query* yang disebut dengan Cypher. Neo4j hampir sama dengan SQL, hanya struktur *statement* sedikit berbeda untuk mengakomodasi *query data graph*. Neo4j juga dapat diakses menggunakan *interface* web dari browser (Zemzans, 2016).

Metode Penelitian

Metode penelitian yang dilakukan pada proses konversi data terdiri dari beberapa langkah. Konversi dimulai dari definisi model data relasional MySQL. Kemudian dari data relasional yang ada dilakukan tahapan proses konversi data, mulai dari meng-*export* data menjadi file JSON melalui antarmuka PhpMyAdmin pada aplikasi XAMPP. Selanjutnya *export file* JSON ke MongoDB, atau melalui *tools* 3T Studio lakukan *import* data JSON. Setelah berhasil di-*import*, lakukan manajemen data menggunakan aplikasi Robo 3T, selain itu juga lakukan *testing* (pengujian) dengan membuat *query* di MongoDB. Tahapan terakhir lakukan uji performa untuk melihat seberapa cepat waktu eksekusi *query database* MongoDB. Tahapan proses konversi data digambarkan pada Gambar 1.



Gambar 1. Tahapan Proses Konversi *Database* MySQL ke NoSQL MongoDB

Hasil dan Pembahasan

Tahapan proses konversi data sesuai dengan rancangan pada metodologi terdiri dari beberapa proses.

Definisi Model Data: *Schema Relational Database*

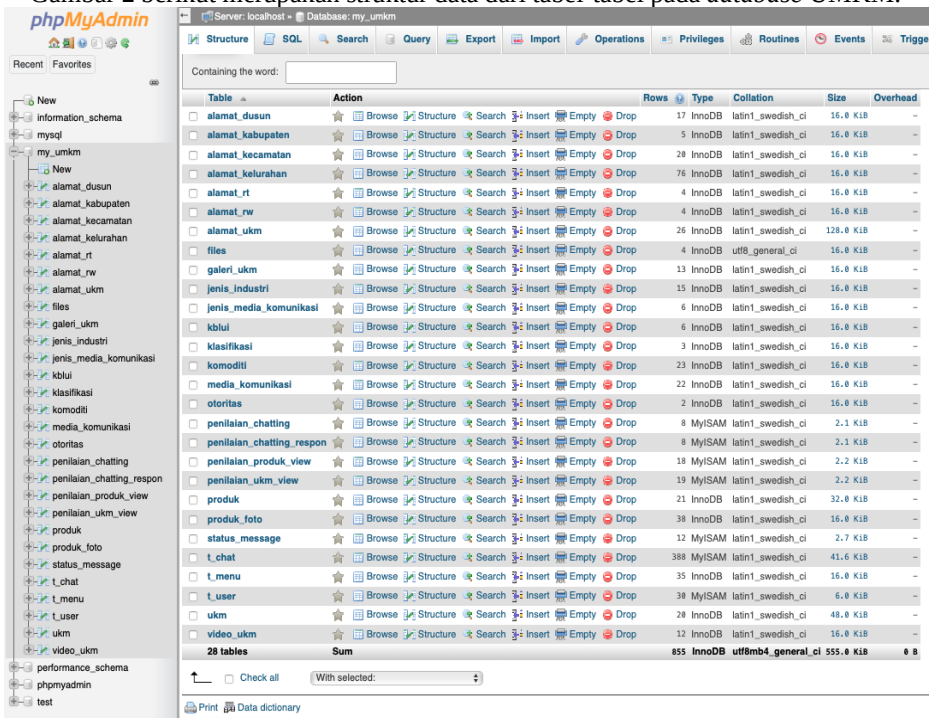
Relational database untuk sistem informasi UMKM dirancang memiliki relasi 6 (enam) buah tabel utama dan 22 (dua puluh dua) tabel tambahan yaitu tabel ukm, jenis industri, pedukuhan, kelurahan, kecamatan dan kabupaten. Berikut *schema database relasional* untuk setiap tabel utama pada database UMKM (Keterangan: * primary key, ** foreign key) yaitu:

1. Tabel jenis_industri: kode_industri (varchar[11])*, nama_industri (varchar[11]).
2. Tabel kabupaten: kode_kabupaten (varchar[10])*, nama_kabupaten (varchar[30]).
3. Tabel kecamatan: kode_kabupaten (varhar[10])**, kode_kecamatan (varchar[10])*, nama_kecamatan (varchar[30]).
4. Tabel kelurahan: kode_kecamatan (varchar[10])**, kode_kelurahan (varchar[10])*, nama_kelurahan (varchar[30]).
5. Tabel pedukuhan: kode_pedukuhan (varchar[10])*, nama_pedukuhan (varchar[30]), kode_kelurahan (varchar[10])**.
6. Tabel ukm: kode_umkm (varchar[10]), nama_umkm (varchar[30]), nama_pemilik (varchar[30]), deskripsi (text), alamat (varchar[50]), kode_pedukuhan (varchar[20])**, kode_industri (varchar[10])**.

Proses Konversi Data

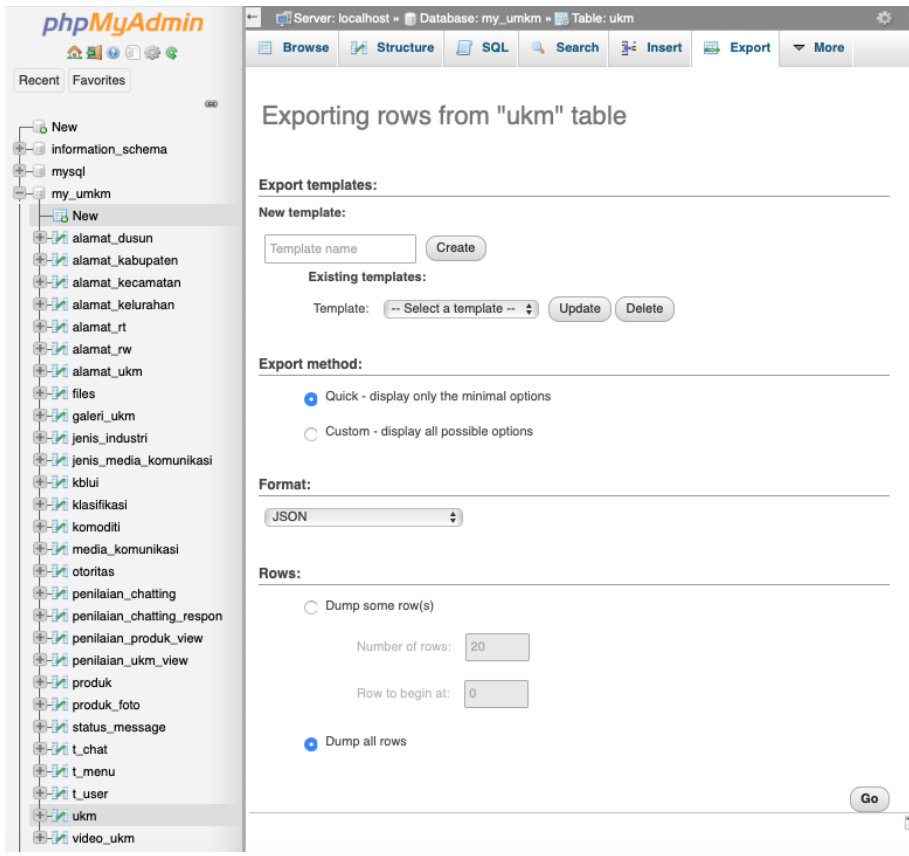
Proses konversi data terdiri dari beberapa tahapan yaitu antara lain:

1. Data akan di-*export* menjadi file JSON melalui PhpMyAdmin XAMPP. Data-data dari sistem informasi UMKM terdiri dari 6 tabel utama dan 22 tabel tambahan. Gambar 2 berikut merupakan struktur data dari tabel-tabel pada *database* UMKM.



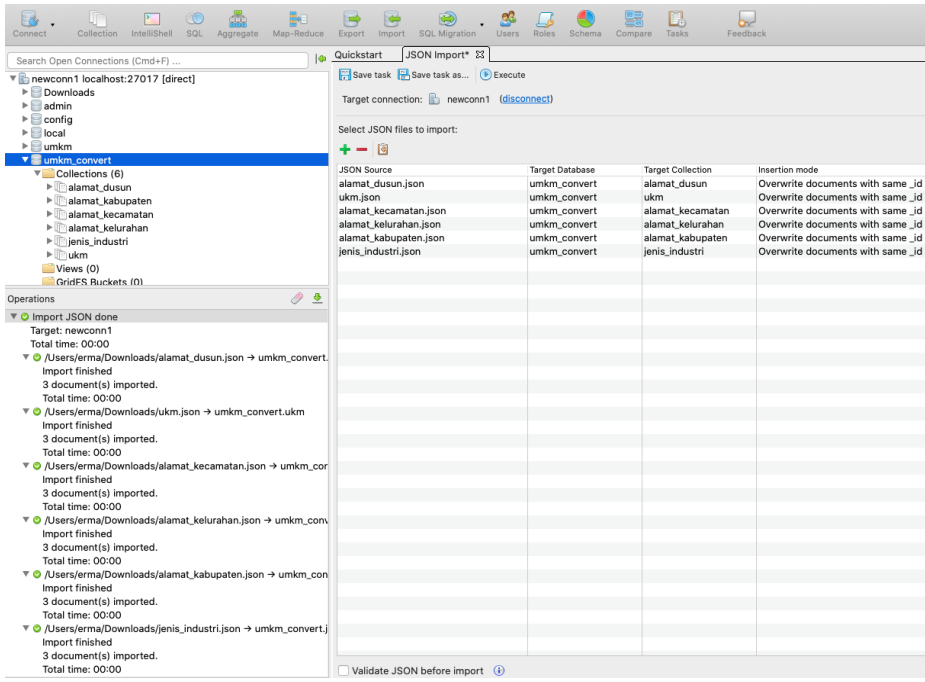
Gambar 2. Database Relational MySQL

2. Proses konversi awal dilakukan menggunakan *tools* XAMPP yaitu PhpMyAdmin. *Export data* dari MySQL menjadi file JSON, agar nanti dapat dikonversi ke MongoDB. Proses *export data* masih memanfaatkan *tools* PhpMyAdmin dengan memilih menu *Export* seperti pada Gambar 3. Atur format data ke format JSON, kemudian lakukan proses *dump* untuk semua *record data*.



Gambar 3. *Export* Data Menjadi Format File JSON

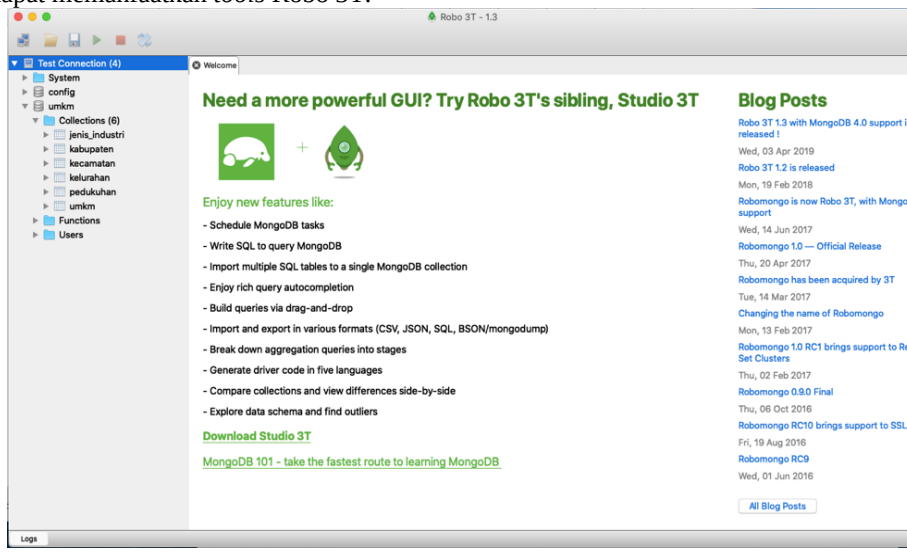
3. Langkah-langkah untuk *eksport data* JSON, hidupkan file exe pada MongoDB untuk NoSQL ManagerDb Professional. Sekali *exe console terminal* terbuka maka akan menuju ke NoSQL manajer MongoDB dan mulai *import data* JSON menggunakan *tools* XAMPP. Tahap untuk ekstraksi yaitu dengan memilih tipe JSON file dan memasukkan nama path. Tampilan seperti pada Gambar 4. Selanjutnya pilih opsi data JSON sehingga *file encoding methodology* akan dipilih pada fase ini. Berikutnya menentukan opsi *import* dengan *highlight* opsi koleksi baru dan tentukan nama koleksi yang diinginkan untuk digunakan, sehingga file JSON di-*import* dalam koleksi baru. Setelah itu jalankan tombol eksekusi. Terakhir periksa operasi status *import* apakah berhasil atau tidak.



Gambar 4. Konversi Data JSON ke MongoDB

Manajemen *database* MongoDB berbasis GUI dengan Robo 3T

Setelah proses konversi berhasil, maka untuk mengelola *database* MongoDB dapat memanfaatkan *tools* Robo 3T.

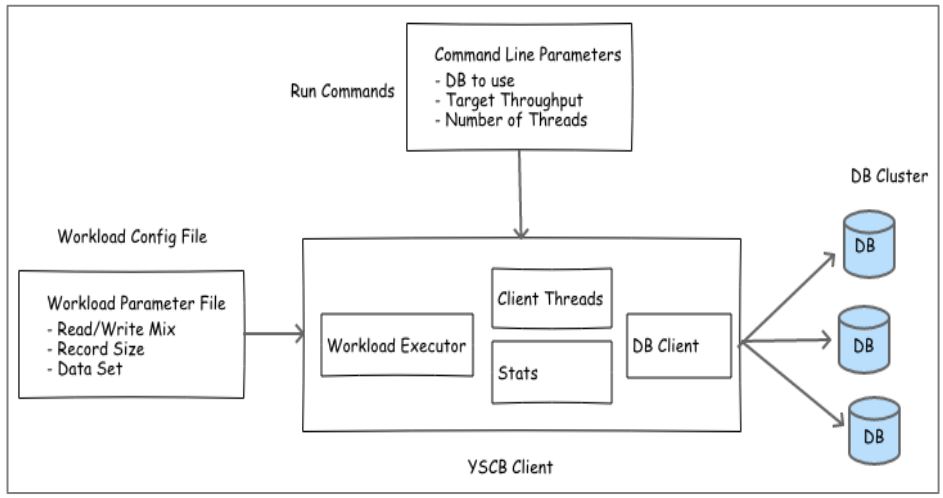


Gambar 5. Kelola *Database* NoSQL dengan Robo 3T

Tools Robo 3T merupakan *tools open source* berlisensi GPL-3.0 untuk melakukan manajemen *database* berbasis GUI dan tersedia untuk semua sistem operasi. Gambar 5 menunjukkan Pengelolaan *database* dengan Robo 3T.

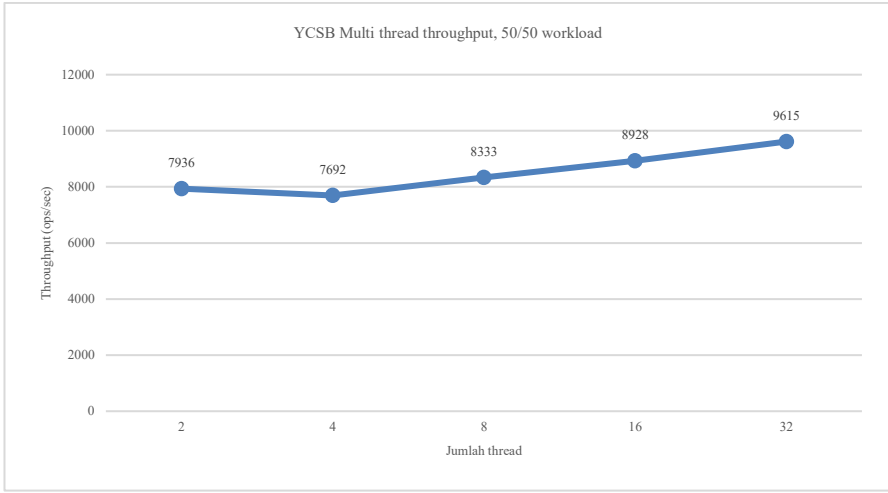
Uji performa dengan YCSB

YCSB merupakan program java yang dikembangkan oleh Yahoo! untuk membandingkan performa berbagai NoSQL *database*. YCSB adalah spesifikasi *open-source* Java yang populer dan merupakan rangkaian program yang dikembangkan oleh Yahoo! untuk membandingkan kinerja relatif dari berbagai *database* NoSQL. YCSB di sini digunakan untuk menguji kinerja MongoDB yang berjalan sebagai layanan *container*. *Workload* (beban kerja) yang digunakan YCSB A dan B. *Workloads* digunakan untuk membandingkan berbagai studi NoSQL *Database*. *Workload* A (*update workload* yang berat) yaitu 50/50% gabungan dari *reads/writes*. Sedangkan pada *Workload* B (*read workload*) merupakan gabungan 95/5% *reads/writes*. Cara kerja YCSB *Client* dapat dilihat pada Gambar 6.



Gambar 6. YCSB Client

Sedangkan untuk uji coba performa *throughput* MongoDB menggunakan YCSB *Client* dilakukan menggunakan perintah *workloads* dan dilakukan dengan jumlah *thread* yang diuji dari 2, 4, 8, 16 dan 32. Hasil uji coba menunjukkan bahwa peningkatan jumlah *thread* mempengaruhi peningkatan *throughput* dalam ops/sec. Hasil eksperimen *throughput* kinerja konversi MongoDB ditampilkan pada Gambar 7. Pengujian *throughput* dengan menggunakan *thread* 2 dapat memproses data sebanyak 7936 menghasilkan 8000 ops/sec, thread 4 memproses jumlah data sebanyak 7692 menghasilkan *throughput* <8000 ops/sec, dan thread 32 memproses jumlah data 9615 menghasilkan *throughput* >8000 ops/sec pada operasi *insert*, *read* dan *update*.



Gambar 7. Hasil Pengujian *Throughput* Pada Operasi *Read/Write* pada MongoDB

Simpulan

Proses konversi data ke format MongoDB terdiri dari pasangan *field* (kolom) dan *value* (nilai). Dokumen dalam MongoDB mirip dengan JSON. JSON mentransfer data antara server dan aplikasi web, sehingga menjadi format yang *human readable* (mudah dibaca oleh manusia). Berdasarkan hasil penelitian dapat disimpulkan bahwa MongoDB memberikan performa yang tinggi pada data persisten, mendukung model *embedded* data dalam mengurangi aktivitas I/O pada interaksi *database* dengan kecepatan tinggi yang mencakup kunci dari dokumen *embedded* dan *array*. Data yang dihasilkan repositori akan disimpan dalam format JSON dokumen. Dokumen JSON tersebut dapat dengan mudah di-*parsing* secara langsung oleh Javascript.

Tahapan konversi data dari *database* MySQL ke NoSQL MongoDB berhasil dilakukan dengan meng-*import* data dari MySQL ke dalam format data JSON. Selanjutnya data JSON dikonversi ke MongoDB dengan memanfaatkan Studio 3T. Setelah data berhasil terkonversi selanjutnya dilakukan manajemen data dengan Robo 3T. Pengujian *throughput* MongoDB menggunakan YCSB *Client* dilakukan menggunakan perintah *workloads* dan dilakukan dengan jumlah *thread* yang diuji dari 2, 4, 8, 16 dan 32. Hasil uji coba menunjukkan bahwa peningkatan jumlah *thread* mempengaruhi peningkatan *throughput* dalam *ops/sec*. Hasil pengujian performa *database* MongoDB menggunakan YCSB Client dengan jumlah thread yang meningkat mempengaruhi peningkatan secara linier *throughput* (*ops/sec*) dan jumlah data.

Daftar Pustaka

Ajdari, J. and Kasami, B., 2018. MapReduce Performance in MongoDB Sharded Collections. *International Journal of Advanced Computer Science and Applications*, 9(6), pp. 115-120.

Andersson, E. and Berggren, Z., 2017. A Comparison Between MongoDB and MySQL Document Store Considering Performance. Umea University.

Damodaran, B.D., Salim, S. and Vargese, S.M., 2016. Performance evaluation of MySQL and MongoDB databases. *International Journal on Cybernetics & Informatics (IJCI)*, 5(2), pp. 387-394.

Deari, R., Zenuni, X., Ajdari, J., Ismaili, F. and Raufi, B., 2018. Analysis and Comparison of Document-Based Databases With SQL Relational Databases: MongoDB vs MySQL. *Proceedings of the International Conference on Information Technologies (InfoTech 2018)*, 20-21 September 2018, pp. 1-10.

Khedkar, S., Thube, S., Estate, W.I. and Naka, C., 2017. Real time databases for applications. *International Research Journal of Engineering and Technology (IRJET)*, 4(06), pp. 2078-2082.

Kumarasinghe, C., Liyanage, K., Madushanka, W. and Mendis, R., 2016. Performance Comparison of NoSQL Databases in Pseudo Distributed Mode: Cassandra. MongoDB & Redis. University of Moratuwa.

Mearaj, I., Maheshwari, P. and Kaur, M.J., 2018, November. Data Conversion from Traditional Relational Database to MongoDB using XAMPP and NoSQL. *The Fifth HCT Information Technology Trends (ITT)*. Amity University Dubai. 28-29 Nov 2018. pp. 94-98.

Parmar, R.R. and Roy, S., 2018. MongoDB as an Efficient Graph Database: An Application of Document Oriented NOSQL Database. Data Intensive Computing Applications for Big Data. *Data Intensive Computing Applications for Big Data*. Vol 29. pp. 331.

Saparkhojayev, N., Mukasheva, A. and Saparkhojayev, P., 2018, October. The development of information system of formation and use of information resources for evaluation of parameters and evaluation of recommendations based on big data technology tools: work with Mongo DB. *International Conference on Cyber Security and Computer Science (ICONCS'18)*. pp. 18-20.

Stanescu, L., Brezovan, M. and Spahiu, C.S., 2019. MongoDB vs. SQL Server In Medical Applications. *International Journal of Computer Science & Applications*, vol 16(1). pp 38-54.

Thillainayaki, M. and Hemalatha, M., 2017. A Protein Database as DAAS in The Cloud Environment. *Technology*, 8(2), pp.100-111.

Zemzans, O., 2016. Exploring NoSQL Databases: Comparison of Databases. University of Applied Sciences MAMK.