Perbandingan Kemampuan *Database* NoSQL dan SQL dalam Kasus ERP Retail

Faizal Anugrah Bhaswara, Riyanarto Sarno,dan Dwi Sunaryono
Departemen Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember (ITS)
e-mail: riyanarto@if.its.ac.id

***Abstrak*—Makalah ini akan membahas dua jenis database, yaitu jenis *Relational* (SQL) dan *Non-Relational* (NOSQL). Mengingat dewasa ini para *developer* berlomba untuk memberikan *database* yang tepat dengan performa yang terbaik untuk aplikasinya. Kemudian makalah ini memberikan analisa perbandingan *database* NoSQL dengan SQL dalam hal kinerja, fleksibilitas, dan skalabilitas. Setelah terbukti *database* mana yang tepat, maka akan diterapkan aplikasi ERP Retail dengan yang berorientasikan *multitenancy* dengan tujuan agar aplikasi memiliki performa bagus, fleksibel dalam penyimpanan data, juga mendukung penyimpanan data yang terus berkembang seiring berjalannya waktu. Dalam uji coba yang telah dilakukan, *database* NoSQL terbukti memiliki kecepatan penyimpanan data yang lebih unggul dalam hal CRUD daripada SQL. Juga memiliki struktur penyimpanan data yang fleksibel karena model data berupa BSON (Binary JSON). Dan memiliki kemampuan untuk menjadi *scalable* dengan metode *sharding*. Jadi dalam hal ini database NoSQL akan lebih baik untuk diterapkan pada ERP Retail.**

Kata Kunci—ERP (Enterprise Resource Planning), Retail, Multitenancy, NoSQL

I. PENDAHULUAN

PADA setiap aplikasi yang dibangun oleh developer tidak akan lepas dari peran *database*. *Database* atau basis data sendiri adalah kumpulan data yang disimpan secara sistematis di dalam komputer yang dapat diolah maupun direkayasa menggunakan perangkat lunak untuk menghasilkan informasi. Basis data merupakan aspek yang sangat penting dalam sistem informasi dimana basis data merupakan gudang penyimpanan data yang akan diolah lebih lanjut. Basis data menjadi penting karena dapat mengorganisasi data, menghidari duplikasi data, hubungan antar data yang tidak jelas dan juga update yang rumit. Dan beberapa sistem yang telah ada masih mengadopsi cara kerja *single-tenant database.* Cara kerja seperti ini cenderung akan membatasi proses bisnis yang akan berkembang kedepannya.

Perencanaan sumber daya perusahaan, atau sering disingkat ERP dari istilah bahasa Inggris-nya *Enterprise Resource Planning*, adalah sistem informasi yang diperuntukkan bagi perusahan manufaktur maupun jasa yang berperan mengintegrasikan dan mengotomasikan proses bisnis yang berhubungan dengan aspek operasi, produksi maupun distribusi di perusahaan bersangkutan.

Ritel/eceran atau disebut pula *retail* dalam bahasa inggris adalah salah satu cara pemasaran produk meliputi semua aktivitas yang melibatkan penjualan barang secara langsung ke konsumen akhir untuk penggunaan pribadi dan bukan bisnis. Jadi ERP Retail merupakan sebuah sistem informasi yang menangani kebuhan perusahaan dalam hal pemasaran produk termasuk penjualan barang hingga sampai ke konsumen.

Isi dari makalah ini adalah membandingkan antara *database relational* (SQL) dengan *database non-relational* (NoSQL) dalam hal *performance, flexibility,* dan *scalability.* Setelah terbukti *database* mana yang lebih unggul dari keduanya maka akan diterapkan untuk aplikasi ERP Retail yang berorientasikan *multitenancy*. Dengan begitu diharapkan ERP Retail yang dibangun akan memiliki performa yang optimal dalam hal *store* data. Juga menjadikan aplikasi yang bersifat fleksibel agar mampu mensuport penyimpanan data yang dinamis berdasarkan skema setiap *tenant*, dan perkembangan proses bisnis yang akan selalu berkembang kedepannya. Dan menjadi aplikasi yang *scalable* dalam artian mampu mengatasi jumlah data yang selalu berkembang dengan *distributed database.*

Arsitektur *database* pada aplikasi ini menerapkan *shared database* dengan *shared schema*. Dengan tujuan menghemat jumlah database untuk menangani banyak *tenant* dengan optimal. Juga database mampu menerima skema dan struktur setiap *tenant* yang berbeda-beda..

II. TINJAUAN PUSTAKA

Dalam tinjauan pustaka akan dibahas beberapa teori yang dipakai dalam penelitian ini, yaitu tentang NoSQL (MongoDB), ERP (*Enterprise Resource Planning*), performa, fleksibilitas, dan skalabilitas.

A. *ERP* (*Enterprise Resource Planning*)

ERP adalah sebuah sistem diperuntukan bagi perusahaan manufaktur maupun jasa yang berperan mengintegrasikan dan mengotomasikan proses bisnis yang berhubungan dengan aspek operasi, produksi, maupun distribusi di perusahaan bersangkutan [1].

Sistem ERP berfokus pada integrasi semua organisasi departemen, fungsi dan proses dalam satu komputer sistem informasi, mampu mendukung dalam semua bidang, dengan setiap kebutuhan individu dan spesifik mereka. Setiap departemen memiliki subsistem informasi komputer sendiri,

dioptimalkan sesuai dengan karakteristik yang lebih spesifik dari aktivitas pengembangan, namun sistem ERP menggabungkan semua karakteristik dalam program perangkat lunak terpadu yang berjalan pada database yang unik sehingga semua organisasi bisa berbagi informasi dengan lebih baik dan dengan demikian mampu mengkomunikasikan informasi dengan baik [2].

B. *MongoDB (Database Management System)*

Pada tahun 1998 pertama kalinya dikembangkan sebuah konsep penyimpanan basis data yaitu NoSQL oleh Carlo Strozzi, yang kemudian pada tahun 2009 Eric Evans memperkenalkan kembali teknologi NoSQL. Kehadiran NoSQL bukan berarti untuk menggantikan model RDBMS yang sudah ada. Awal kemunculannya dilatarbelakangi oleh beberapa masalah yang muncul dari RDBMS. NoSQL dan RDBMS memiliki kelebihan dan tempat masing-masing sehingga diharapkan dapat saling melengkapi teknologi penyimpanan basis data [3]. Pada MongoDB dokumen disimpan dalam bentuk BSON (Binary JSON) [4]. Dengan struktur yang mirip dengan JSON membuatnyanya cukup mudah untuk dibaca.

C. Performa

Dalam kinerja dan eksekusi berbagai jenis operasi, database NoSQL terbagi menjadi dua kategori. Yaitu *read* dan *write* yang dioptimasi [5]. Performa yang lebih cepat dikarenakan NoSQL database saat melakukan penyimpanan data tidak memerlukan pengecakan struktur data maupun *primary key* dan *foreign key.*

D. Fleksibilitas

MongoDB menyimpan data dalam format dokumen menggunakan JSON. Namun dalam MongoDB memiliki format yang lebih spesifik yaitu BSON (Binary JSON). Ini adalah skema dokumen dan pemetaan untuk jenis bahasa pemrograman asli [6].

Format JSON dalam 1 record ditandai dengan kurung kurawal, selanjutnya ditulis nama field dan diikuti dengan titik dua , baru dikuti dengan nilai dari kolom tersebut. Secara umum formatnya sebagai berikut:

{[nama Field] : [Nilai Field] , [nama Field2] : [Nilai Field]…}

Contoh : {NIM : ‘123456’, Nama : ‘Supriyanto’ , Isaktif : true}

Tipe data yang disupport oleh BSON/JSON antara lain:

- String
- Integer
- Boolean
- Arrays

Gambar 1. *Document* BSON [7]

E. *Skalabilitas*

Skalabilitas menyangkut kemampuan sistem untuk mengatasi beban kerja yang meningkat. Dalam konteks database, ini dapat didefinisikan sebagai perubahan kinerja saat *node* baru ditambahkan, atau perangkat keras ditingkatkan. Database NoSQL telah dikembangkan secara khusus untuk menargetkan skenario dimana skalabilitas sangat penting. Sistem ini bergantung pada skalabilitas horizontal dan "elastis", dengan menambahkan lebih banyak *node* ke sistem daripada mengupgrade perangkat keras. Istilah "elastis" mengacu pada elastisitas, yang merupakan karakterisasi dari cara cluster bereaksi terhadap penambahan atau penghilangan *node* [5].

III. UJI COBA PERFORMA

Pengujian dibagi menjadi dua: uji performa dan uji fleksibilitas. Pengujian performa dilakukan dengan memasukkan dataset untuk pengujian uji coba *insert* dan pengelolaan data dalam database dengan query *join/aggregation*, *select*, *update*, dan *delete*. Untuk pengujian fleksibilitas dilakukan dengan memasukkan beberapa data berbeda yang membentuk skema ke dalam database.

A. *Dataset*

Dalam penelitian ini akan digunakan dua jenis dataset yaitu JSON untuk MongoDB dan SQL untuk MySQL.

Dataset akan dibagi secara bertahap. Untuk uji coba dilakukan dengan input dataset dengan jumlah yang sama dan dimasukkan secara berulang. Misalnya, inputan pertama menggunakan 1000 data. Lalu dilakukan berulang kali untuk input berikutnya. Dari data 1, 2, 3, 4, 5, ..8. Adapun *select*, *update*, dan uji coba uji coba mulai dari 1000 data yang meningkat secara bertahap dengan 2000, 3000, 4000, 5000, ... 8000.

B. *Performa*

ERP Retail merupakan aplikasi yang menapung data besar dan membutuhkan kemampuan yang cepat dan optimal dalam pengolahan data. Pada paper ini akan dibuktikan *database* mana yang lebih cepat performanya antara *database* jenis SQL dan jenis NoSQL. Dalam uji coba ini digunakan dataset dengan jumlah dan konten yang sama agar setara beban yang digunakan.

1) Uji coba Insert

Pada pengujian ini akan dibuktikan kemampuan *insert* dari database MySQL dan MongoDB. Adapun prosesnya akan dijabarkan lebih detail di bawah ini.

1. Membuat *query insert* untuk testing. *Query* akan dijabarkan pada **Gambar 2** dan **Gambar 3.**

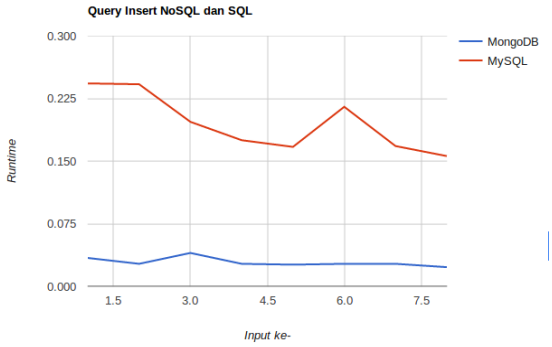
```
db.tes.insert(
{“id_account”:1,”username”:”jlawson2a”,”password
”:”gZnS9Su52RLH”})
```

Gambar 2. *Query* Insert MongoDB

```
INSERT INTO account VALUES (1,' jlawson2a',' gZnS9Su52RLH');
```

Gambar 3. *Query* Insert MySQL

2. Kemudian *query* dijalankan pada *collection* tujuan. Uji coba dilakukan 8 kali dimulai dari *insert* 1000 data untuk uji coba pertama dan lanjut kelipatan 1000 pada percobaan berikutnya dengan bertahap.
3. *Runtime* hasil uji coba dicatat setiap transaksi dari mulai *insert* 1000 data pertama, 2000 data kedua, 3000 data ketiga, dan seterusnya.
4. Berdasarkan data yang terkumpul dibuat graik perbandingan *runtime insert* dari MySQL dan MongoDB agar lebih mudah untuk dianalisa. Grafik ditampilkan pada **Gambar 4**.



Gambar 4. Grafik Perbandingan *Runtime Insert*

2) Uji coba Join/Agregasi

Pada pengujian ini akan dibuktikan kemampuan *join*/agregasi dari database MySQL dan MongoDB. Adapun prosesnya akan dijabarkan lebih detail di bawah ini.

1. Membuat *query join*/agregasi untuk testing. *Query* akan dijabarkan pada **Gambar 5** dan **Gambar 6.**

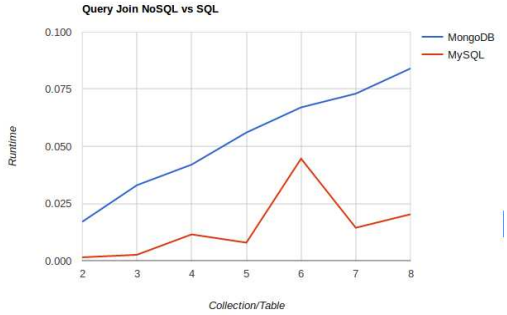
```
db.user.aggregate([
   {
      $lookup: {
         from: "dosen",
         localField: "id",
         foreignField: "id_dosen",
         as: "user1"
         }
      },
      {
         $undwind: "$user1"
      }
])
```

Gambar 5. *Query* Agregasi MongoDB

```
SELECT * FROM user INNER JOIN dosen ON user.id=dosen.id_dosen;
```

Gambar 6. *Query Join* MySQL

2. Kemudian *query* dijalankan pada *collection* tujuan. Uji coba dilakukan pada 8 tabel secara bertahap dimulai dengan 2 tabel, 3 tabel, 4 tabel, dan seterusnya.
3. *Runtime* hasil uji coba dicatat setiap transaksi dari mulai *join*/agregasi 2 tabel, 3 tabel, 4 tabel, dan seterusnya.
4. Berdasarkan data yang terkumpul dibuat graik perbandingan *runtime join*/agregasi dari MySQL dan MongoDB agar lebih mudah untuk dianalisa. Grafik ditampilkan pada **Gambar 6**.



Gambar 7. Grafik Perbandingan *Runtime Join*/Agregasi

3) Uji coba Select

Pada pengujian ini akan dibuktikan kemampuan *select* dari database MySQL dan MongoDB. Adapun prosesnya akan dijabarkan lebih detail di bawah ini.

1. Membuat *query select* untuk testing. *Query* akan dijabarkan pada **Gambar 8** dan **Gambar 9**

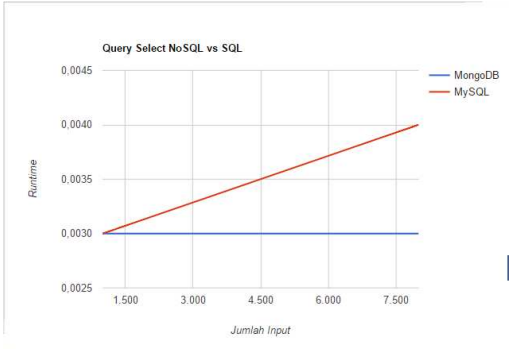
```
db.getCollection('dosen').find()
```

Gambar 8. *Query Select* MongoDB

```
SELECT * FROM 'dosen' WHERE 1;
```

Gambar 9. *Query Select* MySQL

2. Kemudian *query* dijalankan pada *collection* tujuan. Uji coba dilakukan 8 kali dimulai dari *select* 1000 data untuk uji coba pertama dan lanjut kelipatan 1000 pada percobaan berikutnya dengan bertahap.
3. *Runtime* hasil uji coba dicatat setiap transaksi dari mulai *select* 1000 data pertama, 2000 data kedua, 3000 data ketiga, dan seterusnya.
4. Berdasarkan data yang terkumpul dibuat graik perbandingan *runtime select* dari MySQL dan MongoDB agar lebih mudah untuk dianalisa. Grafik ditampilkan pada **Gambar 10.**



Gambar 10. Grafik Perbandingan Runtime Select

4) Uji coba Update

Pada pengujian ini akan dibuktikan kemampuan *update* dari database MySQL dan MongoDB. Adapun prosesnya akan dijabarkan lebih detail di bawah ini.

1. Membuat *query update* untuk testing. *Query* akan dijabarkan pada **Gambar 11** dan **Gambar 12.**

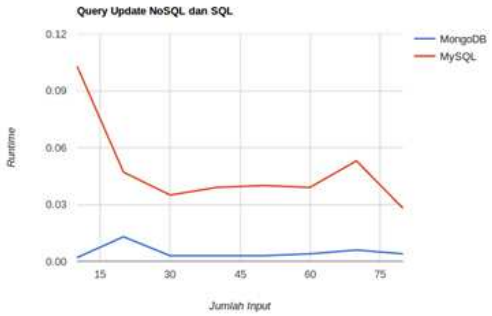
```
db.dosesn.updateMany(
   { “id_dosen”: {$lt:11}},
   {
     $set: { “title”: “test”}
   }
)
```

Gambar 11. *Query Update* MongoDB

```
UPDATE dosen SET title = “Test” WHERE
id_dosen < 11;
```

Gambar 12. *Query Update* MySQL

2. Kemudian *query* dijalankan pada *collection* tujuan. Uji coba dilakukan 8 kali dimulai dari *update* 10 data untuk uji coba pertama dan lanjut kelipatan 10 pada percobaan berikutnya dengan bertahap.
3. *Runtime* hasil uji coba dicatat setiap transaksi dari mulai *update* 10 data pertama, 20 data kedua, 30 data ketiga, dan seterusnya.
4. Berdasarkan data yang terkumpul dibuat graik perbandingan *runtime update* dari MySQL dan MongoDB agar lebih mudah untuk dianalisa. Grafik ditampilkan pada **Gambar 13.**



Gambar 13. Grafik Perbandingan *Runtime Update*

5) Uji coba Delete

Pada pengujian ini akan dibuktikan kemampuan *delete* dari database MySQL dan MongoDB. Adapun prosesnya akan dijabarkan lebih detail di bawah ini.

1. Membuat *query delete* untuk testing. *Query* akan dijabarkan pada **Gambar 14** dan **Gambar 15.**

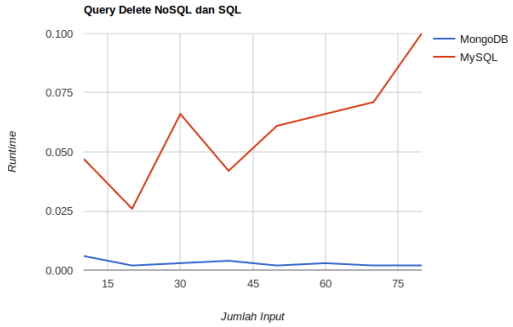
```
db.dosen.remove(
   {“id_dosen”: { $lt : 11} }
)
```

Gambar 14. *Query Delete* MongoDB

```
DELETE FROM dosen1 WHERE id_dosen < 11;
```

Gambar 15. *Query Delete* MySQL

2. Kemudian *query* dijalankan pada *collection* tujuan. Uji coba dilakukan 8 kali dimulai dari *delete* 10 data untuk uji coba pertama dan lanjut kelipatan 10 pada percobaan berikutnya dengan bertahap.
3. *Runtime* hasil uji coba dicatat setiap transaksi dari mulai *delete* 10 data pertama, 20 data kedua, 30 data ketiga, dan seterusnya.
4. Berdasarkan data yang terkumpul dibuat graik perbandingan *runtime delete* dari MySQL dan MongoDB agar lebih mudah untuk dianalisa. Grafik ditampilkan pada **Gambar 16.**

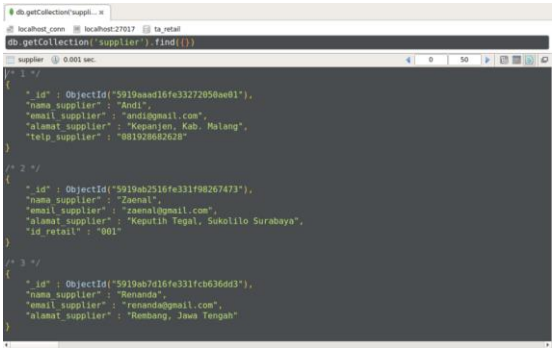


Gambar 16. Grafik Perbandingan *Runtime Delete*

3. Fleksibilitas

Aplikasi ERP Retail dirancang sebagai aplikasi yang memiliki skema dinamis dari tingkat antarmuka, layanan web, hingga database. Oleh karena itu akan dibuktikan pada tingkat database apakah database tipe NoSQL ini mampu menangani kebutuhan ini, yang databasenya cukup rumit dan bahkan tidak bisa diatasi. Proses uji coba akan dijelaskan di bawah ini.

1. Buat satu *collection* sebagai lokasi penyimpanan data yang berbeda dari setiap tenant.
2. Rancang aplikasi ERP Retail menjadi aplikasi yang dinamis. Dalam artian setiap tenant bebas merubah modul hingga *form* untuk membentuk skema yang menjadi kebutuhan masing-masing tenant.
3. Rancang Dari ketiga *form* data diatas hasil *store* data ditempatkan pada satu collection. Hasil data penyimpanan pada collection ditampilkan pada **Gambar 17.**



Gambar 17. *Colletion* Data Hasil Skema Tenant

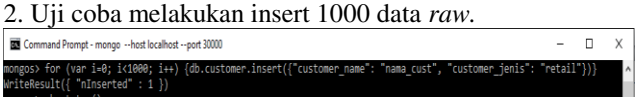
4. Skalabilitas

ERP Retail dirancang dengan *database* yang mampu menangani pertumbuhan data dengan cara mendistribusikannya ke *database* lain. Adapun proses uji coba pendistribusian data akan dijabarkan di bawah ini.

1. Hasil penampilan mongos (mongo server) dari *sharding*.

Gambar 18. Penampang Mongos dari *Sharding*

2. Uji coba melakukan insert 1000 data *raw*.



Gambar 19. Uji Coba Insert 1000 data *raw*

IV. HASIL UJI COBA

Pada bagian ini akan diberikan hasil evaluasi dari pengujian-pengujian yang telah dilakukan. Evaluasi yang diberikan meliputi evaluasi pengujian performa, fleksibilitas dan skalabilitas terhadap aplikasi.

1) Evaluasi Pengujian Performa

Rangkuman mengenai hasil pengujian *performance* dapat dilihat pada **Tabel 1.**

Tabel 1.
Hasil *Performance* MongoDB dan MySQL

No.	Uji Coba	MongoDB	MySQL	Selisih
1	Insert	0.028	0.195	0.167
2	Join/Agregation	0.053	0.053	0.039
3	Select	0.003	0.012	0.009
4	Update	0.004	0.048	0.044
5	Delete	0.003	0.059	0.056

2) Evaluasi Pengujian Fleksibilitas

Hasil pengujian kemampuan *flexibility* dapat dilihat pada **Tabel 2.**

Tabel 2.
Hasil Fleksibilitas MongoDB dan MySQL

No.	*Action*	MongoDB	MySQL
1	*Insert* dengan data yang dinamis	✓	×

1.1. Evaluasi Pengujian Skalabilitas

Hasil pengujian kemampuan *scalability* dapat dilihat pada **Tabel 3.**

Tabel 3.
Hasil Uji Coba Skalabilitas MongoDB

No.	*Action*	Hasil
1	Distribusi 1000 data raw ke shard1 dan shard2	Berhasil

V. KESIMPULAN

Database NoSQL terbukti unggul dalam proses transaksi CRUD daripada SQL dengan jumlah selisih *runtime insert* 0,167, *select* 0,009, *update* 0,44, *delete* 0,056. Namun lemah untuk menghadapi transaksi *join*/agregasi dengan jumlah selisih *runtime* 0,039. *Database* NoSQL mampu memenuhi kebutuhan aplikasi ERP Retail yang bisa mensupport perbedaan skema dari setiap tenant dan mengikuti perkembangan proses bisnis kedepannya. *Database* NoSQL bersifat *scalable* dibuktikan dengan mendistribusikan 1000 data *raw* ke dua shard (shard1 dan shard2).

DAFTAR PUSTAKA

[1] "Wiley: Integrated Business Processes with ERP Systems - Simha R. Magal, Jeffrey Word." [Online]. Available: http://www.wiley.com/WileyCDA/WileyTitle/productCd-EHEP001815.html. [Accessed: 29-May-2017].

[2] "Andrejs Tambovcevs and Tatjana Tambovceva 'ERP system implementation: benefits and economic effectiveness' Faculty of Computer Science and Information Technology Riga Technical University Riga, Latvia 2013." .

[3] "Mufid Itsnaini Zain and M. Rudyanto Arief 'PERBANDINGAN STRUKTUR PENYIMPANAN DAN PERFORMANSI NOSQL MONGODB DENGAN DBMS SQL SERVER' Jurusan Teknik Informatika STMIK AMIKOM YOGYAKARTA." .

[4] "Ameya Nayak, Anil Poriya, and Dikshay Poojary 'Type of NOSQL Databases and its Comparison with Relational Databases' Dept. of Computer Engineering Thakur College of Engineering and Technology University of Mumbai, March 2013." .

[5] J. R. Lourenço, B. Cabral, P. Carreiro, M. Vieira, and J. Bernardino, "Choosing the right NoSQL database for the job: a quality attribute evaluation," *J. Big Data*, vol. 2, no. 1, p. 18, Aug. 2015.

[6] "Divya Chauhan and K. L. Bansal 'Using the Advantages of NOSQL: A Case Study on MongoDB' Himachal Pradesh University Summerhill,Shimla,India." .

[7] "Rupali Arora, Rinkle Rani Aggarwal 'Modeling and Querying Data in MongoDB' July-2013." .