QUERY RESPONSE TIME COMPARISON NOSQLDB MONGODB WITH SQLDB ORACLE

Humasak T. A. Simanjuntak[1], Lowiska Simanjuntak[2], Goretti Situmorang[3] and Adesty Saragih[3]
[1, 2, 3, 4]Information System Department, Institut Teknologi Del
Jln. Sisingamangaraja, Sitoluama, Laguboti, Toba Samosir
e-mail: humasak@gmail.com[1], lowiskafelicia@gmail.com[2], gorettis010@gmail.com[3], adestysaragih17@gmail.com[4]

ABSTRAK

Penyimpanan data saat ini terdapat dua jenis yakni relational database dan non-relational database. Kedua jenis DBMS (Database Managemnet System) tersebut berbeda dalam berbagai aspek seperti performansi eksekusi query, scalability, reliability maupun struktur penyimpanan data. Kajian ini memiliki tujuan untuk mengetahui perbandingan performansi DBMS antara Oracle sebagai jenis relational database dan MongoDB sebagai jenis non-relational database dalam mengolah data terstruktur. Eksperimen dilakukan untuk mengetahui perbandingan performansi kedua DBMS tersebut untuk operasi insert, select, update dan delete dengan menggunakan query sederhana maupun kompleks pada database Northwind.

Untuk mencapai tujuan eksperimen, 18 query yang terdiri dari 2 insert query, 10 select query, 2 update query dan 2 delete query dieksekusi. Query dieksekusi melalui sebuah aplikasi .Net yang dibangun sebagai perantara antara user dengan basis data. Eksperimen dilakukan pada tabel dengan atau tanpa relasi pada Oracle dan embedded atau bukan embedded dokumen pada MongoDB. Response time untuk setiap eksekusi query dibandingkan dengan menggunakan metode statistik.

Eksperimen menunjukkan response time query untuk proses select, insert, dan update pada MongoDB lebih cepatdaripada Oracle. MongoDB lebih cepat 64.8 % untuk select query;MongoDB lebihcepat 72.8 % untuk insert query dan MongoDB lebih cepat 33.9 % untuk update query. Pada delete query, Oracle lebih cepat 96.8 % daripada MongoDB untuk table yang berelasi, tetapi MongoDB lebih cepat 83.8 % daripada Oracle untuk table yang tidak memiliki relasi.Untuk query kompleks dengan Map Reduce pada MongoDB lebih lambat 97.6% daripada kompleks query dengan aggregate function pada Oracle.

Kata Kunci*: Response time query, oracle, mongodb,performance comparison.*

ABSTRACT

Nowadays, there are two types of Data Management System, relational databases and non-relational databases. Both types of DBMS (Database management System) differ in various aspects such as performance, scalability, reliability of the query, and data storage structure. The purpose of this study is to determine the performance comparison between Oracle DBMS as a type of relational databases and MongoDB as a kind of non-relational databases to process structured data. The experiments have done to measure insert, select, update, and delete query response time on both databases. Northwind Database is used as a database case.

To achieve the objective of experiment, 18 queries which consist of 2 insert queries, 10 select queries, 2 update queries and 2 delete queries executed. These queries executed through .Net Application that built as an intermediaries between user and databases. The experiments carried out to the table with/without relation in Oracle and embedded/no embedded document in MongoDB. The Query response time for each query execution compared by using statistical methods.

The experiment result show query response time for select, insert, and update operation in MongoDB is faster than Oracle. For each operation, MongoDB faster 64.8 %, 72.8 % and 33.9 % for relation/non relation table. However, delete operation on Oracle faster 96.8 % than on MongoDB for relation table, but MongoDB 83.8 % faster than Oracle for non-relation table. Furthermore, complex query with MapReduce function in MongoDB is slower 97.6% than aggregate function in Oracle.

Keywords*: mongodb, oracle, performance comparison, response time*

I. INTRODUCTION

At this time there are two kinds of DBMS, relational database and non-relational database [1]. Relational database is a popular and common DBMS that used to store structured data (data with a format and size that has been specified) [2], such as Oracle, MySQL, and Microsoft SQL Server. DBMS Oracle is the most widely used by companies such as PT. Perusahaan Gas Negara (Persero) and IBM. Based on survey that publishes in DB-Engines site, Oracle is a most used relational database in industry [3]. Relational databases are used in a large scope of applications due to their rich set of features, query capabilities and transaction management. However, they are not able to store and process big data (that commonly has unstructured data) effectively and are not very efficient to make transactions and join operations [4]. Recently, emerge a new paradigm, NoSQLDB databases, to overcome some of these problems, which are more suitable for the usage in web environments. NoSQLDB or Non-Relational database is a DBMS that used to stored and process unstructured data (data that does not follow a specific data format, varied and can grow in real time) [2]. Some of examples non-relational database are MongoDB, Apache Hadoop, Cassandra, and CouchDB [3]. MongoDB is one of the non-relational database that is widely used by companies such as Cisco, EA and eBay. Based on survey, MongoDB is in first ranking for Non-Relational database that used in industry [3]. This survey support this research to use Oracle and MongoDB as an example for each type of database. Furthermore, Oracle and MongoDB has a complete feature for relational database and non-relational database.

This both types DBMS differ in various aspects such as performance of the query execution, scalability, reliability and structure of data storage in storing structured data [5]. Currently, a number of studies have been conducted to compare these DBMS, but not many studies compare the query response time on these DBMS in processing structured data [1] and to support idea "is it possible relational database replaced with NoSQLDB Database?".The previous research performed a comparison i.e. comparing NoSQLDBMongoDB with SQLDB Microsoft SQL Server in processing structured data by using simple Data Manipulating Language (DML) query, without join operation, aggregate function, WHERE clause condition in query, and map reduce feature in MongoDB[1].

This research performed comparison of query response time performance on MongoDB as NoSQL Database with Oracle as a Relational Database. Both of them are most widely used database today. The other reasons are MongoDB is developer-oriented that pay close attention to ease of use rather than other NoSQL database, have rich document structure, and offer flexible querying capabilities [6]. For Oracle, It is used for almost all large application, more efficient, and responds very well with excellent performance in demanding environments [7].

This research conducted by using simple and complex DML queries against structured data. Complex query consists of selection query from more than one table (using join operation), using where clause, and/or using aggregate function. Furthermore, response time query comparison also performed between MapReduce feature in MongoDB and the aggregation function in Oracle.

II. RESEARCH METHOD

The type of this research is mainly to obtain the result of query response time performance between Oracle SQLDB and NoSQLDBMongoDB in processing of structured data. Our research was conducted by applying some approaches with following steps below:

1) Conduct literature study on relational and non-relational database, specifically on data storage and data model of Oracle and MongoDB. At this stage, the learning and understanding of those concepts was conducted by reading and understanding papers and text book available on those fields.
2) Analyze query, data storage, and data model of MongoDB and Oracle DBMS. The analysis resulted a Northwind data model for MongoDB database and all queries that executed in experiment. Analysis also conducted for developing program as an intermediate tool between user and database.
3) Develop an application which consists of method to execute insert, select, update, and delete query for supporting the experiment. The application display the query response time for each query execution.
4) The experiment performed by using application to measure response time query in MongoDB and Oracle. Northwind Database used as a database case.
5) Evaluate the result of experiment to produce a performance comparison between Oracle and MongoDB.
6) Conclude the evaluation result in order to summarize the result of query response time comparison.

III. ANALYSIS

In this section, we conducted an analysis of the Northwind data representation on relational database and non-relational database, analysis of the query to be executed on both MongoDB and Oracle DBMS, analysis tool for experiment.

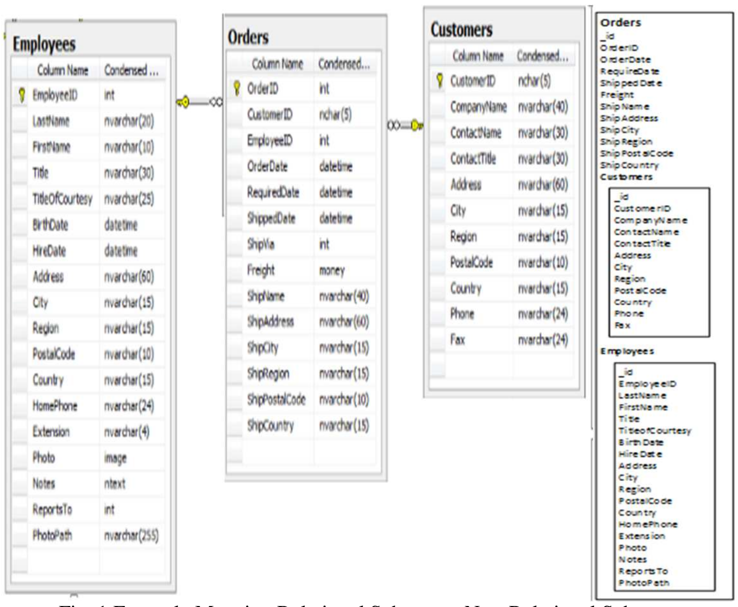
A. *Relational database Data Representation*

Northwind database is a sample database in Microsoft SQL Server. This research use Relational data representation for Northwind database that already built and published by Microsoft [8]. Northwind database consists of 13 tables that related to another tables. Each table has a primary key as the table identity and if it is related to other tables, this primary key will be a foreign key in the relation table.

B. *Non-Relational database Data Representation*

Data analysis was performed by changed schema of NorthWind database (relational database) to schema of non-relational database. This analysis includes of changing table become collection and column become key that has value. The steps of changing from relational database schema to non-relational database schema are:

1) Verify whether tables of relational database have relationship with other tables. If a table is related to another table (foreign key contained in the table), the table is changed into a collection with the name of table becomes the name of collection.
2) If a table has relationship with another table, the structure of the document will be embedded document. Tables that referred by this table become a sub-document (see Figure 1). If this table does not have relationship to the table, this table immediately become collection
3) Orders and OrderDetails tables in NorthWind database will be embedded document because it has relationship to the other tables which are Orders table related with Employees table, Customers and OrderDetails tables related to Orders and Products tables. Employees and Customers table become sub-documents in the Orders collection while in OrderDetails table, Orders and Products tables become sub-documents. Each Products, Suppliers, Customers, Categories, Employees, Region and Territories tables become collections. So, there are 9 collections which are Products, Suppliers, Customers, Categories, Employees, Region, Territories, Orders and OrderDetails.
4) The columns in each Products, Suppliers, Customers, Categories, Employees, Region, Territories, Orders, and OrderDetails tables, changed become keys. Add value for each key corresponding to the type of data in the modified table. Set the pair of key and value will formed a structure of documents (rows in the table). Northwind database schema on MongoDB database can be seen in Figure 2.



Employees

Column Name	Condensed ...
EmployeeID	int
LastName	nvarchar(20)
FirstName	nvarchar(10)
Title	nvarchar(30)
TitleOfCourtesy	nvarchar(25)
BirthDate	datetime
HireDate	datetime
Address	nvarchar(60)
City	nvarchar(15)
Region	nvarchar(15)
PostalCode	nvarchar(10)
Country	nvarchar(15)
HomePhone	nvarchar(24)
Extension	nvarchar(4)
Photo	image
Notes	ntext
ReportsTo	int
PhotoPath	nvarchar(255)

Orders

Column Name	Condensed...
OrderID	int
CustomerID	nchar(5)
EmployeeID	int
OrderDate	datetime
RequiredDate	datetime
ShippedDate	datetime
ShipVia	int
Freight	money
ShipName	nvarchar(40)
ShipAddress	nvarchar(60)
ShipCity	nvarchar(15)
ShipRegion	nvarchar(15)
ShipPostalCode	nvarchar(10)
ShipCountry	nvarchar(15)

Customers

Column Name	Condensed...
CustomerID	nchar(5)
CompanyName	nvarchar(40)
ContactName	nvarchar(30)
ContactTitle	nvarchar(30)
Address	nvarchar(60)
City	nvarchar(15)
Region	nvarchar(15)
PostalCode	nvarchar(10)
Country	nvarchar(15)
Phone	nvarchar(24)
Fax	nvarchar(24)

```
Orders
_id
OrderID
OrderDate
RequireDate
ShippedDate
Freight
ShipName
ShipAddress
ShipCity
ShipRegion
ShipPostalCode
ShipCountry
Customers
    _id
    CustomerID
    CompanyName
    ContactName
    ContactTitle
    Address
    City
    Region
    PostalCode
    Country
    Phone
    Fax
Employees
    _id
    EmployeeID
    LastName
    FirstName
    Title
    TitleofCourtesy
    BirthDate
    HireDate
    Address
    City
    Region
    PostalCode
    Country
    HomePhone
    Extension
    Photo
    Notes
    ReportsTo
    PhotoPath
```

Fig. 1 Example Mapping Relational Schema to Non-Relational Schema

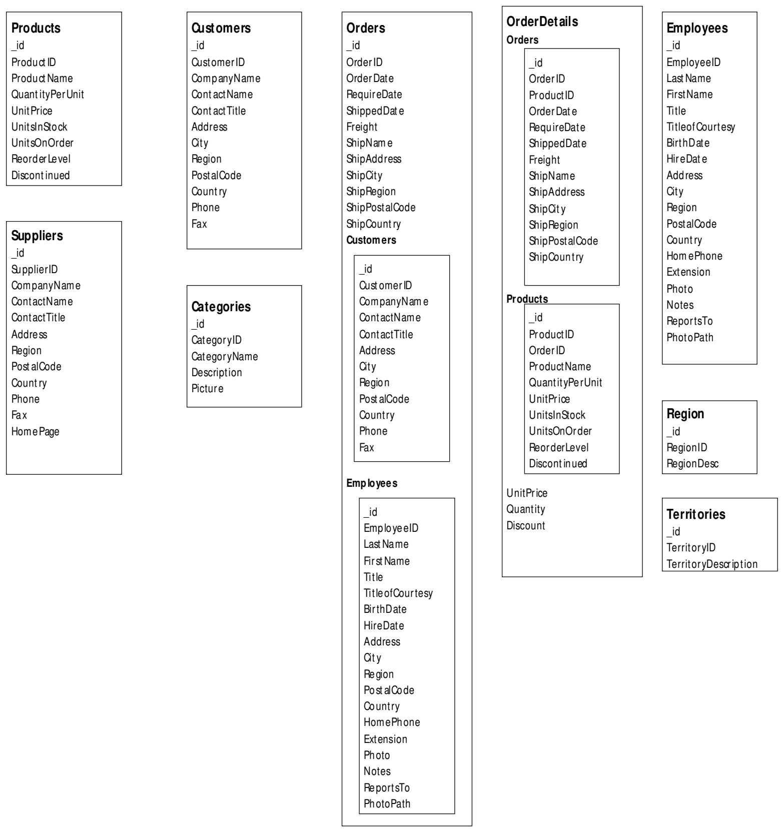


Fig. 2 Northwind Schema in MongoDB Database

C. Query Analysis

In previous research, queries executed to get thequery response time ofMongoDB and SQL server for processing simple query. To strengthen the results of response time query experiment, the query analysis was did to produce the best representation of the query. The queries that will be executed are simple and complex queries. The frequently accessed tables that involved in transactions such as Orders, Orders_Details, Customers, Products, Suppliers, and Employees table. The guideline to determine the criteria of query that will be executed are:

1) Simple and complex queries on table that does not has relationship to another table (Oracle) and one collection that does not has a subdocument (no embedded document) for MongoDB.
2) Simple and complex queries on tables that related to other tables in Oracle and collection that consist of subdocument (embedded document) in MongoDB.
3) Query with insert, select, update, and delete operation on tables or collections with certain conditions.
4) Query using aggregate function on the tables or collections.
5) Query with map-reduce function compared with query using aggregation function on MongoDB.

Based on analysis, there are 18 queries that will be executed in experiment. It consists of 2 insert queries, 10 select queries, 2 update queries, and 2 delete queries.

D. Application Development Analysis

Application developed using C#.Net. The application was built because there is no specific Integrated Development Environment (IDE) to perform response time experiment on DBMS, MongoDB 2.6 and Oracle 11g. In addition, this application ensure the experiment conducted in the same environment, so there is no different behavior in the experiment. The Application has two database connections, with Oracle and MongoDB. The Application becomes an intermediary between user and data in DBMS. The 18 queries already defined in the application, so user only need to entry the query criteria through application. After the query executed and running in DBMS, query response time displayed in application.

Overall, there are some comparisons resulted from the analysis that conducted in this research. The comparison can be seen in Table 1.

TABLE I
COMPARISON ORACLE VS MONGODB

	Oracle	**MongoDB**
Structure storage	Object oriented database. Data stored in table that consist of row and column	Document based. Data stored in collection that consist of document (key and value).
Type of Relation	One-to-one, One-to-many, Many-to-many. Using primary key and foreign key to built relationship.	One-to-one, One-to-many, Many-to-many. Using reference and embedded document depend on user choice. It can can affect application performance and database capacity
Many-to-Many Relation	Create a new table with primary key composed by every foreign key itself.	Not create new collection, but there is normalized document structure with object_id in a collection also appear in another collection that relate to first collection
Query	Using SQL query.	Using NOSQL query

IV. EXPERIMENTS AND RESULTS

A. *Response Time Query*

In the experiment, MongoDB 2.6 and Oracle 11g installed on a machine running an Intel Pentium CPU P6300 @2.27 GHz, RAM 3GB. Experiment was carried out in this scientific research has 5 scenarios according to the query operation that executed, such as insert, select, update, delete and map-reduce function. Each scenario was performed by executing a query with different rows are 1, 10, 100, 1000, 10000 and 100000 rows. Query executed 10 times and taking average to get the final result.

For each scenario, query response time result show in a graph. The graph depicted comparison of trendline for each operation. Type of trendline for statictic analysis is polynomial trendline because the result is uncluttered and fluctuation. We got formula used to calculate value of y (query response time) from value of x (number of rows) and R2 is used to determine significance level of the experimental results.

The comparison of experimental result can be seen in the following graph that divided in two categories: query response time on table or collection with relationship and query response time on table or collection without relationship.

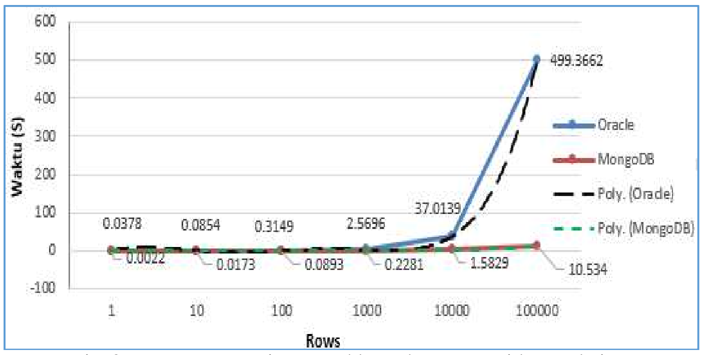


Fig. 3 Insert response time on table or document without relation

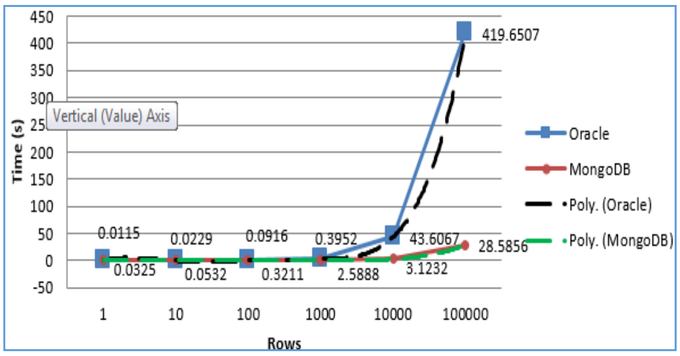


Fig. 4 Insert response time on table or document with relation

Figure 3 and 4 shows the comparison inserting time between the two database systems.Example Query that executed in this experiment are:

- Oracle

```
INSERT INTO Customers (CompanyName, ContactName, ContactTitle, Address, City, Region, PostalCode,
Country, Phone,
Fax)VALUES('value1','value2','value3','value4','value5','value6','value7','value8','value9','value10')
```

- MongoDB

```
db.Customer.insert(
{"CompanyName" : 'value1',"ContactName": 'value2',"ContactTittle": 'value3',"Address" : 'value4',
"City" : 'value5', "Region" : 'value6', "PostalCode" : 'value7', "Country" : 'value8' "Phone" :
'value9', "Fax" : 'value10'})
```

We can see MongoDB is constantly faster that Oracle, eventhough number of rows in database and number of rows that inserted is increasing. Especially in Oracle, response time query slows down significantly when the number of data records in the table and the document increases. Inserts operation on MongoDB is 72.8% faster than Oracle.

Insert operation on Oracle doing some stages (execution plan). Oracle must confirm the valid data type (domain integrity) with values that will be inserted, primary key must be unique, not null (intregrity entities), and relationships between tables where the foreign key must match the referenced primary key in another table. While the inserts in MongoDB is flexible, which means insert done directly without regard to the constraints because MongoDB uses dynamic scheme. It means there is no limitation to the schema or document structure in MongoDB before doing insert process. In addition, insert operation in MongoDB can be done directly without define the collection. That's why inserting data in MongoDB is much faster than inserting data in Oracle.

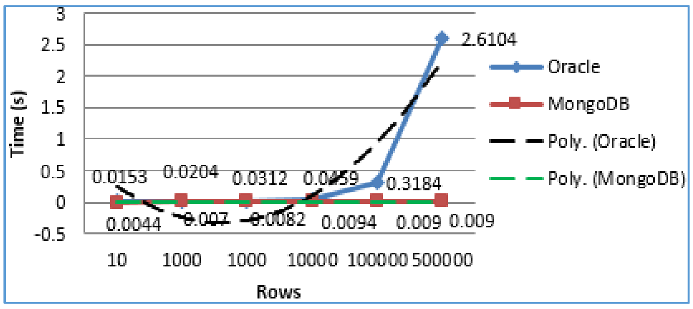


Fig. 5 Select 1 rows response time on table or document without relation

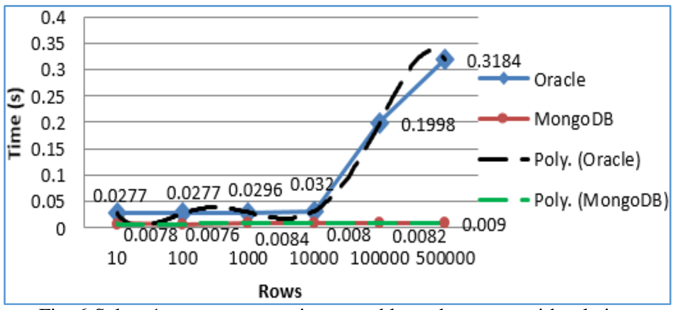


Fig. 6 Select 1 rows response time on table or document with relation

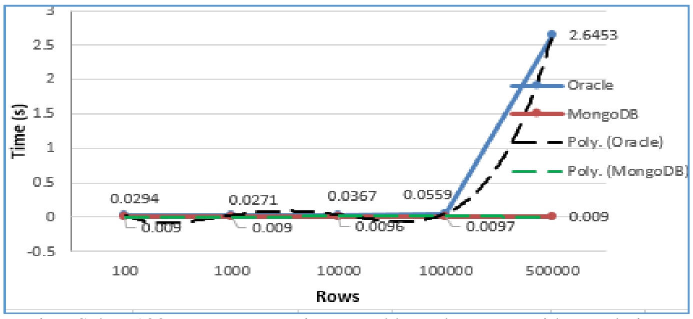


Fig. 7Select 100 rows response time on table or document without relation

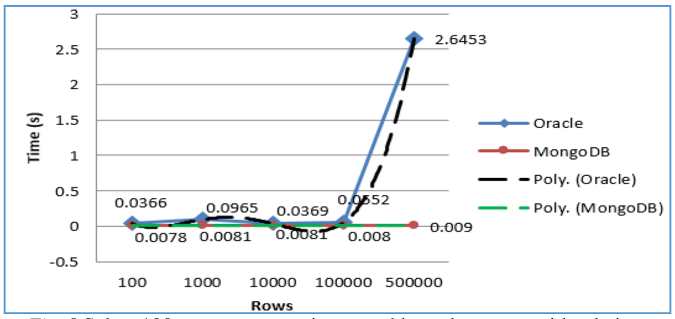


Fig. 8 Select 100 rows response time on table or document with relation

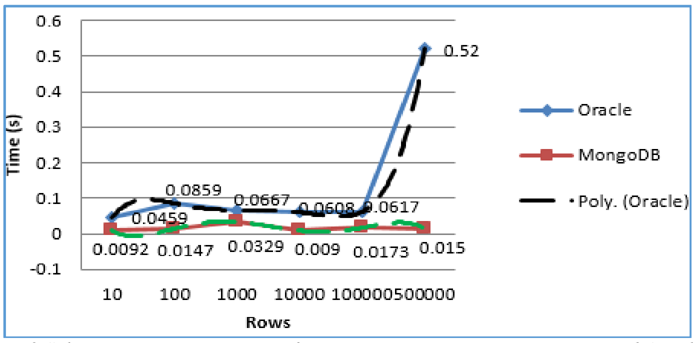


Fig. 9 Select rows using aggregate function response tine on MongoDB and Oracle

Figure 5, 6, 7, and 8 shows the comparison selection time between the two database systems and Figure 9 shows the comparison selection time using aggregate function. Query that used in this experiment is a complex query that used join operation (for Oracle) and some aggregation functions. On select operation using clause between and (select 1 or 100 rows on table/collection with/without relation) and using aggregate function, MongoDB is faster than Oracle. Experiment shows select operation on MongoDB is faster 64.8% than Oracle. Number of rows in Oracle database causes performance decrease significantly, while number of rows selected did not affect to the performance.

Example query that executed for figure 6 and 8 are:

- Oracle

```
Select cus.* from orders ord, customers cus
whereord.customerid = cus.customerid and
Ordered between value1 and value2
```

```
db.Orders.find(
{"Customers.CustmerID" :{'value1' , 'value2'},
 "Orders.CustomerID" : {'value1' , 'value2'}});
```

Example query that executed for figure 9 are:

- Oracle

```
Select distinct sup.companyname
from products pro, suppliers sup
wherepro.supplierid = sup.supplierid and pro.unitsinstock< (select avg(unitsinstock)
from products)
```

- MongoDB

```
db.Products.distinct({{"Suppliers.SupplierID" : "Products.ProductID"},{"Products.Sup-
plierID" : {$gt : $avg {"Products.UnitsInStock"}}},});
```

There are some steps (execution plan) in select operations on Oracle. Firstly, select statement will be executed. If select based on certain criteria (where clause), Oracle access a table that will be selected with full access table. Execution plan performed on the select statement is to filter based on the criteria. Select operations on tables that have relationship to other table executed by accessing table based on index rowid then scan the unique index (primary key). Filter also executed against foreign key (primary key from other tables) that has not null value and to the specific criteria. Select operation in MongoDB is more flexible due to using of dynamic schema and docu

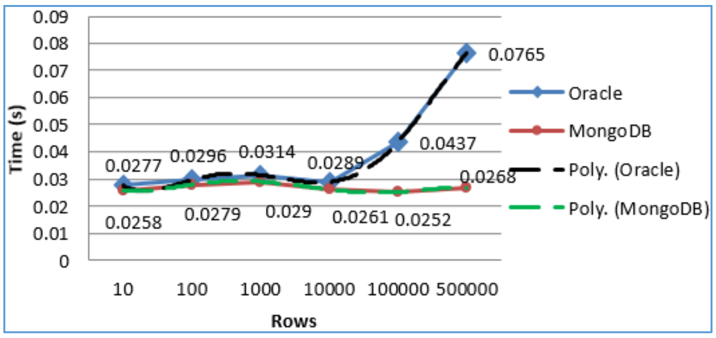


Fig. 10 Update row response time on MongoDB and Oracle

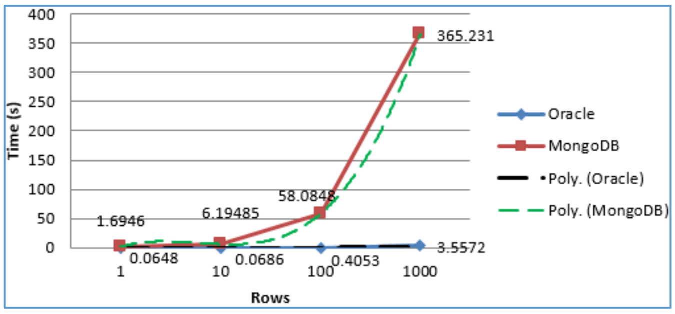


Fig. 11 Delete row response time on MongoDB and Oracle

ments using JSON format that consist of key and value. Select operation in MongoDB with conditions related documents (embedded document) is much faster because MongoDB retrieve data directly from documents in one collection (full collection scan) without having pay attention to the relationships with other documents. Data stored in document that has sub-document. In addition, MongoDB uses index for selection process that facilitate text searching. By using index, searching results that already obtained will be stored in a view (temporary storage) of the collection in an efficient structure.

SQL Query that executed on Figure 10 is

```
Update products set unitprice=value1, unitsinstock=value2, reorderlevel=value3 where suppli-
erid= 90.
```

Update operations on MongoDB with conditions related documents (embedded document) is faster than Oracle. MongoDB is faster 33.9% than Oracle.

SQL Query that executed on Figure 11 is

```
Delete Employees from orders where OrderId = value1.
```

Delete operations on MongoDB with conditions related documents (embedded document) is slower than Oracle. Oracle performance is faster 96.8% than Oracle. In this experiment, MongoDB delete more data than Oracle because data in Employee is a subdocument of Orders. All data that related to the certain OrderId that deleted will remove from Order document.

Last comparison is complex query with aggregate function in Oracle and Map Reduce feature in MongoDB. Aggregations operations process data records and return computed results. Aggregation operations group values from multiple documents together, and can perform a variety of operations on the grouped data to return a single result. MongoDB provides three ways to perform aggregation: the aggregation pipeline, the MaprRduce function, and single purpose aggregation methods and commands. This experiment focus on aggregation using MapReduce-function.

MapReduce is a programming model and an associated implementation for processing and generating large datasets that is amenable to a broad variety of real-world tasks [9].MapReduce has emerged as a popular way to harness the power of large clusters of computers. It allows programmers to think in a data-centric fashion: they focus on applying transformations to sets of data records, and allow the details of distributed execution, network

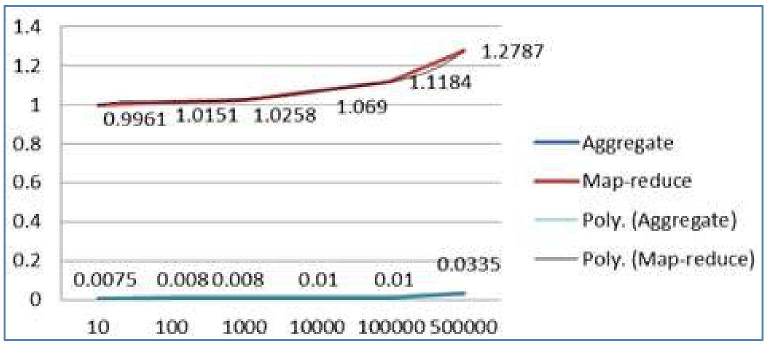


Fig. 12 Aggregation and MapReduce Function on table or document without relation

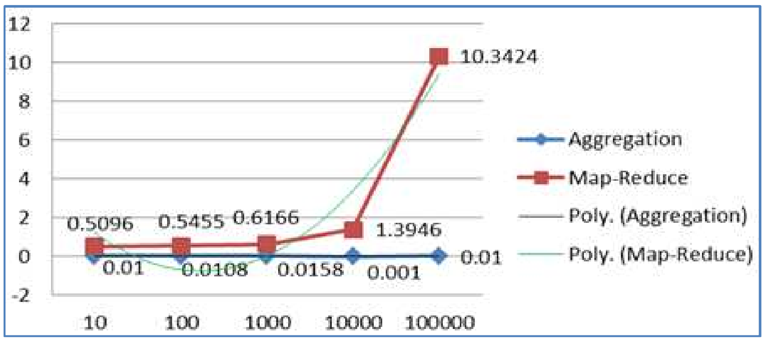


Fig. 13 Aggregation and MapReduce Function on table or document without relation

communication and fault tolerance to be handled by the MapReduceframeworkData Growth [10]. The experiment result of complex query with MapReduce function can be seen in Figure 12 and 13 below.

Both Figures, shows query response time with Mapreduce function on MongoDB database is much slower than aggregation in Oracle Database. The MongoDB execution plan shows that two stages of MapReduce (map function and reduce function) need longer time to execute. Furthermore, MapReduce executed in one limited server, so it gave a major influence for query response time. MapReduce is suitable for distributed servers (multiple servers) andusing sharding feature in multiple machine. The experiment shows, MapReduce performance in MongoDBis 97.6% slower than aggregation in Oracle.

Example query that executed for figure 12 (table without relation) are:

- Oracle

```
select  extension, sum (reportsto) as total
from employees where notes='idle' group by extension
```

- MongoDB

```
db.Employee.mapReduce
(function(){
emit(this.Notes, this.ReportsTo);
},
function(key,values) {return.Array.sum(values)}
{query:{Notes: idle},
out: "result"})
```

Example query that executed for figure 13 (table with relation) are:

- Oracle

```
Select Customers.CompanyName, count(Customers.CustomerID) from Customers, Orders
where Customers.CustomerID = Orders.CustomerID
group by CompanyName having count(Customers.CustomerID)>1 order by CompanyName
```

- MongoDB

```
db.Order.mapReduce
(function(){for (varidx    = 0; idx<this.Customer.length; idx++) {var key = this.Cus-
tomer[idx].Customerid;
var value = {count: 1, qty:this.Customer[idx]. CompanyName};
emit(key, value);}},
function(keySKU,countObjVals) {reducedVal = { count: 0, qty: 0 };
for (varidx = 0; idx<countObjVals.length; idx++) {
reducedVal.count += countObjVals[idx].count;
reducedVal.qty += countObjVals[idx].qty;}
returnreducedVal;}
```

Data growth on Oracle and MongoDB occurred after insert operation with amount of data that inserted is increasing from 10, 100, 1000, 10000 and 100000 rows were performed by 10 times. The growth of data in each DBMS can be seen in Table 2 AND Table 3.

TABLE 1
DATA GROWTH ON TABLE/DOCUMENT WITHOUT RELATION

Volume (Rows)	Oracle (MB)	MongoDB (MB)
First Volume	0	0
10	0	0.0026
110	0	0.0316
1110	0	0.32
11110	5	3.25
111110	44.625	35.3
511110	145.797	163.47

TABLE 2
DATA GROWTH ON TABLE/DOCUMENT WITH RELATION

Volume(Rows)	Oracle (MB)	MongoDB (MB)
First Volume	0	0
10	0	0.0096
110	0	0.11
1110	0	1.07
11110	3.25	10.68
111110	24	106.81
511110	127.05	491.33

Table 2 and Table 3 shows that MongoDB need more storage to stored data than Oracle. If there are some or many documents with relation (many embedded document in DBMS) then the data storage need more capacity. In this experiment, MongoDB needs data storage three times larger than without embedded document.

V. CONCLUSION AND SUGGESTIOS

From this research, we concluded that MongoDB has better runtime performance for insert, select, update queries. Oracle faster than MongoDB on Delete queries. Some conclusions based on experiment of this research are:

1) The experiment show query response time for select, insert, and update operation in MongoDB is faster than Oracle. For each operation it faster 64.8 %, 72.8 % and 33.9 % for relation/non relation table. However, delete operation on Oracle faster 96.8 % than on MongoDB
2) MapReduce function is not improve query performance if MongoDB running in single machine or not implementing Sharding feature. The developer is better to use common aggregate function.
3) Based on the results of select query execution, query response time in Oracle is affected by number of rows data stored in the database. More number rows in the database, query response time query swill be slower.
4) Based on the result of experiment, data model design on MongoDB (reference or embedded document) affect the performance of database. In complex query that use join, the relational database should query more than one table, but MongoDB only need to query one document if using embedded data model. User must choose the right data model for better performance. The data model also affect the capacity of data storage.

5) This research can be improved by executing more complex queries and implementing MongoDB in multiple machine. The complex query executed in this research only involve two tables using join operation. In addition, it is possible to change data model (references or embedded) for knowing the best data model for system.

REFERENCES

[1] *Parker Zachary, Poe Scott, Vrbsky Susan. (2013). "Comparing NoSQLMongoDB to an SQL DB", ACMSE'13.* [Online]. Available: http://delivery.acm.org/

[2] *Hurwitz Judith, Nugent Alan, Fern Halper Dr., Kaufman Marcia, "Big data For Dummies",John Wiley & Sons, Inc, Canada, 2013*

[3] solid IT. (2014). *DB-Engines Ranking. [Online]. Available:* http://db-engines.com/en/ranking.

[4] *VeronikaAbramova, Jorge Bernardino. (2013). "NoSQL Databases: MongoDBvs Cassandra", Conference C3S2E.* [Online]. Available: http://delivery.acm.org/

[5] *HadjigeorgiouChristoforos, "RDBMS vsNoSQL: Performance and Scaling Comparison", 2013, The University of Edinburg*

[6] *(2014). RethinkDB compared to MongoDB. [Online]. Available: http://rethinkdb.com/docs/rethinkdb-vs-mongodb/*

[7] *Chandra Vennapoosa. (2006). Advantages of Using Oracle. [Online]. Available: http://www.learn.geekinterview.com/database/oracle/advantages-of-using-oracle.html*

[8] *Antony Binoj. Microsoft. (2011). Northwind Database. [Online]. Available: https://www.codeplex.com/site/users/view/binoj7*.

[9] Dean Jeffrey, GhemawatSanjay. (2008). MapReduce: Simplified Data Processingon Large Clusters. *COMMUNICATIONS OF THE ACM. [Online]. Available: http://www.allendowney.com/ss08/handouts/dean08mapreduce.pdf*

[10] *Condie Tyson, Conway Neil, etc. "MapReduce Online", UC Berkeley, Yahoo! Research. [Online]. Available: http://db.cs.berkeley.edu/papers/nsdi10-h*

[This page intentionally left blank]