Experiment of Query Optimization Techniques to get the Efficient Query

Luqman Hakim
Master of Informatics
AMIKOM University,
Yogyakarta, Indonesia

Suryanto Nugroho
Master of Informatics
AMIKOM University,
Yogyakarta, Indonesia

Sigit Hadi Waluyo
Master of Informatics
AMIKOM University,
Yogyakarta, Indonesia

ABSTRACT
Optimization of a database query becomes one of the critical phases in data processing handling. Database optimization is needed to make a structured query language more concise and efficient. This study aims to conduct experiments with structured query language modifications compared to structured query language examples commonly used in the manufacture of software. This study uses a database with a record large enough to try to implement its SQL. The results of the experiments of all modified query writing are more efficient than the queries used by the information system in the database regarding data processing time.

General Terms
Database, Structured Query Language

Keywords
Database Relation, Structured Query Language, Database Optimization, Database Management System.

1. INTRODUCTION
The development of technology in the world of information so fast and rapidly in various fields. This development of course in addition to requiring adequate human resources also required the existence of new solutions to create something that allows users to perform activities associated with it. One of the most indispensable solutions in the world of information technology is the provision of accurate data, as well as real-time so that that data changes can be processed quickly.

Companies with large data transactions in addition to requiring fast data access also require data analysis that contains information related to their activities. A manufacturer or trading company needs analysis for some information, such as analysis of sales trends and supply of goods. Information, also, to be fast and reliable also required an attractive interface and easy to understand that will facilitate the business processes that occur in it.

In finding a solution, one thing that should not be forgotten is the cost requirements used to apply the solution. A solution that benefits one side should not incur losses on the other. For example, there is a solution to connect to the database but must be accompanied by the addition of devices with high qualifiers and high prices. The decision to add devices does solve the problem on one side, but from the financial side will cause costs that may be detrimental to the company.

Query optimization is required to improve the speed in database processing. This query optimization is needed by a developer or a database administrator (DBA), as this is one of the necessary skills a developer or database administrator must have in managing the database [1]. Also, developers or database administrators can also create a query execution plan well if doing database optimization [2].

Information systems with complex problems and involving multiple tables will be a separate issue when linked to the relation between tables in the database will require a long query time. Real-time data processing also becomes the demands of an information system to improve data processing so that it can be used accurately and on time. According to Habimana [2], knowledge of database optimization and query writing is required to obtain maximum query results. The best way to improve performance is to try to write your query in some different ways. And compare their reading and execution plans. In this paper we will try various techniques that you can use to try to optimize your database query.

2. BASIC THEORY

2.1 DATABASE
The understanding database is an organized collection of data. Such data are usually held to be able to model the aspects of reality in a way that supports the process in need. According to Gordon C. Everest [3] A database is a collection or data collection that is mechanical, shared, formally defined and also centrally controlled within an organization.

2.2 SRUCTURED QUERY LANGUAGE
SQL (Structured Query Language) is a standard language or programming for RDBMS (Relational Database Management System). Although called language, it may be a bit awkward when we call the SQL programming language, more familiar if those sounds are programming C, Visual Basic, Java, Delphi, and so on [4]. The following languages are included in the imperative programming; the simplest is the language in the form of core instructions. Whereas, SQL is involved in declarative programming, which is more in sentence form or statement. SQL has at least two kinds of commands used to manage and organize databases:

1. Data Definition Language (DDL)
The commands used by database administrators (DBA) to define the schema into the DBMS. The scheme is a complete description of table structure, recording and data relationships in the database. DDL is also used to define subschema. Subschemes are views for database users that are subsets of the schema. When an item is not listed in a user's schema, the item is not available to the user. Subschema can be a security mechanism of the base system data, namely by setting the right accessing items in the database. DDL is also used to create, modify and delete databases [4].

2. Data Manipulation Language (DML)
Are the commands used to change, manipulate and retrieve data in the database? Actions like deleting, converting and retrieving data become part of DML. DML is mainly divided into two:

A. Procedural, which requires the user to determine what data is needed and how to get it.

B. Nonprocedural, which requires the user to specify what data is required, but no need to mention how to get it [4].

3. PAST RESEARCH

Wenjiao Ban et al. [1] conducts research on query optimization on distributed databases using genetic algorithms and max-min ant systems; both theories will be parallelly combined to obtain efficient queries. The results of the experiments performed by the proposed algorithm for multi-join query processing.

Sebastian Haas et al. [5] conducts research by integrating specialized hardware and enables the processing of energy-efficient queries and query optimization by applying selectivity estimation techniques. Our chip measurements show a 1000x energy boost in selected database operators compared to the current system.

S.Venkata Lakshmi and Valli Kumari Vatsavayi [6] Conducted research on query optimization on distributed databases using genetic algorithms. . Experimental Analysis The proposed methodology is performed on 100 different queries distributed in 20 different locations that have 8 relations in each query. This compares to DB2 distributed optimization and achieves increased reliability and high performance with respect to query optimization and query costs in a distributed database. The proposed technique provides efficient performance in different environments.

4. DISCUSSION

This research is simpler than the reference paper. This study uses query modification as a comparison in terms of time effectiveness. Optimization can be done in various ways, by understanding tuning performance in the database. Some techniques and methods may require different special treatment, depending on the database used. For example, performance improvements can be made from the administration of databases such as file configuration and updating of services or security packs, which of course each database has its uniqueness and technique. There is a set of methods and techniques commonly applied to RDBMS, perhaps not all of them can be implemented because they depend heavily on their respective application environments, but at least can be used as guides and references to form the best system according to the conditions at hand.

Optimization through SQL command also plays a role that is not less important. The core of SQL itself is the authority to perform retrieval, insertion, updating, and deletion of data, accompanied by administrative support and database management functions.

In this discussion will be tested by comparing the original query with a modified query. Some tips will be tested and compared data processing time to be done to see the efficiency of query modification.

Experiment #1
Use Column Name instead of * in SELECT statement

If you select only a few columns from the table, there is no need to use SELECT *. While it's easier to write the query, more time for the database to complete the query. By selecting only the columns you need, you reduce the size of the results table, reduce network traffic and also improve overall query performance [2].

Query example

```
SELECT * FROM KUNJUNGANPASIEN
```

Query modification

```
SELECT KPKD_PASIENN FROM
KUNJUNGANPASIEN
```

Fig 1: Query Example Select *

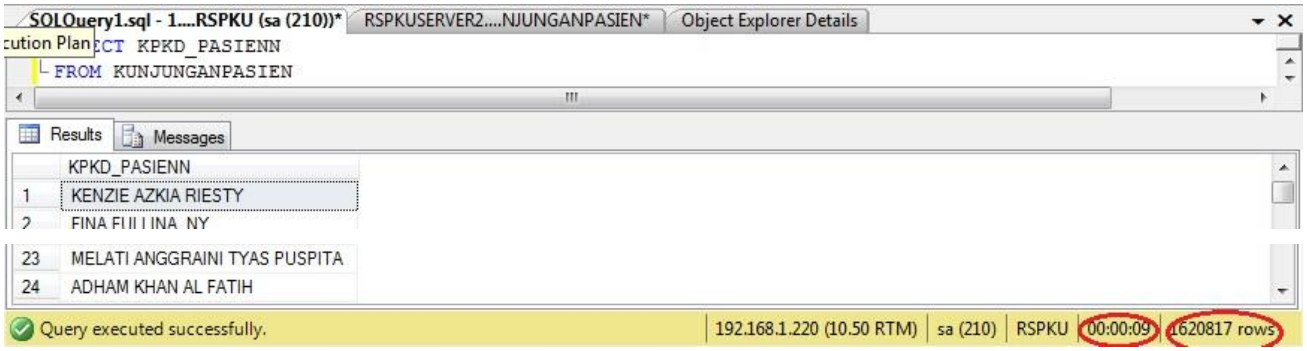

Fig 2: Query Modification Select Column name

In Figure 1 shows the query result in the example query, In the example SQL, it takes 40 seconds to execute 162,804 lines of query results whereas in Figure 2 for SQL modification it takes 9 seconds to run 162.817 lines so that query modification is more efficient than the query example.

Experiment #2

Avoid using HAVING in SELECT statements

The HAVING clause is used to filter rows after all rows are selected and used as filters. A clause like This is not very useful in a SELECT statement. It works by passing the final result table of a query that parses a row that does not meet the HAVING condition [2].

Query Contoh

```
SELECT
K.KPKD_PASIEN,count(K.KPKD_PASIEN)
FROM KUNJUNGANPASIEN K
GROUP BY K.KPKD_PASIEN
HAVING K.KPKD_PASIEN != '0168751' AND
K.KPKD_PASIEN != '0276988';
```

Query Modifikasi

```
SELECT
K.KPKD_PASIEN,count(K.KPKD_PASIEN)
FROM KUNJUNGANPASIEN K
GROUP BY K.KPKD_PASIEN
WHERE K.KPKD_PASIEN != '0168751' AND
K.KPKD_PASIEN != '0276988';
```

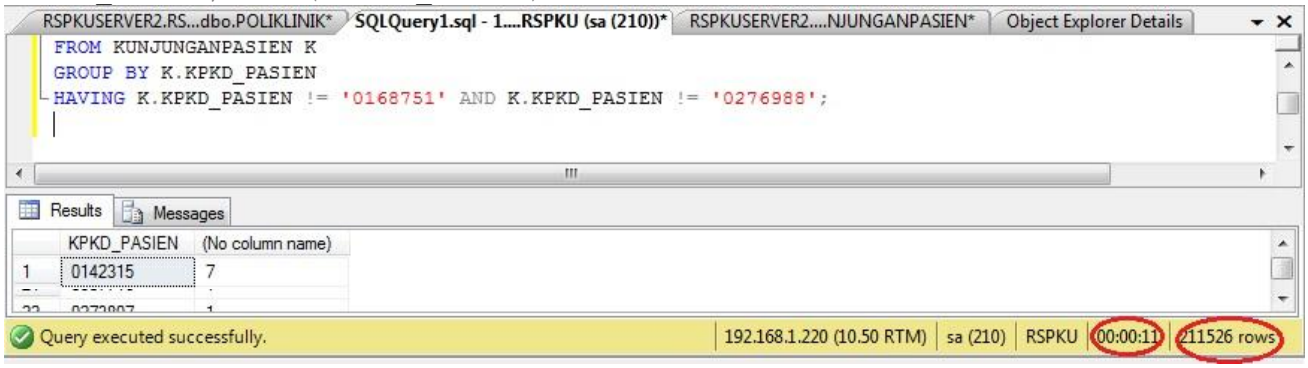

Fig 3: Query Example Using Having

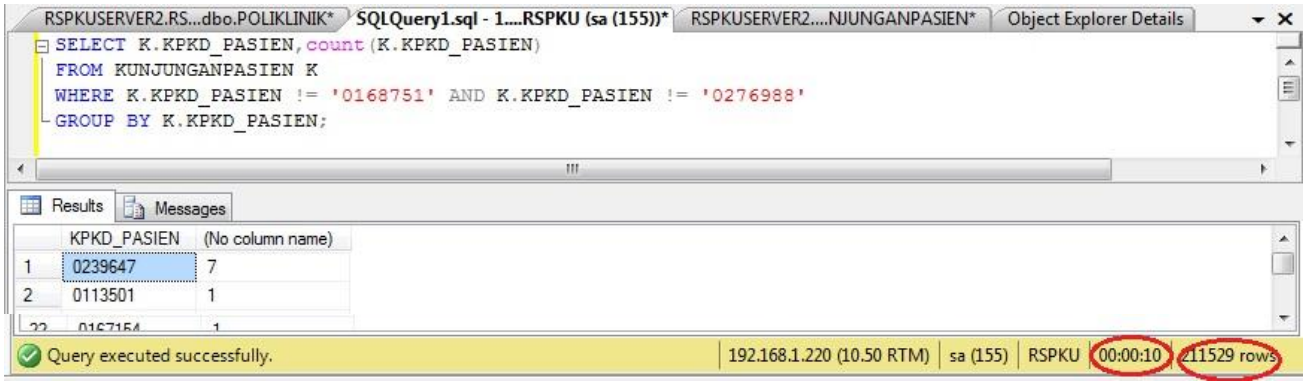

Fig 4: Query Modification does not use having

In Figure 3 shows the query result in the sample query, the sample query takes 11 seconds to execute 211,526 lines of query results while in Figure 4 for the modification query takes 10 seconds to run 211,529 lines so that the query modification is more efficient than the query example.

Experiment #3

Consider using the IN predicate when performing an indexed column query.

The IN predicate can be exploited for index retrieval, and also, optimization can sort the list of INs to match the order of the index sequences, leading to more efficient recovery. Note that IN only contains constants, or a constant value for one execution of the query block [2].

Query Example

```
SELECT K.* FROM KUNJUNGANPASIEN K
WHERE K.KPKD_POLY = 'PK005' OR
K.KPKD_POLY = 'PK008';
```

Query Modification

```
SELECT K.* FROM KUNJUNGANPASIEN K
WHERE K.KPKD_POLY IN
('PK005','PK008');
```

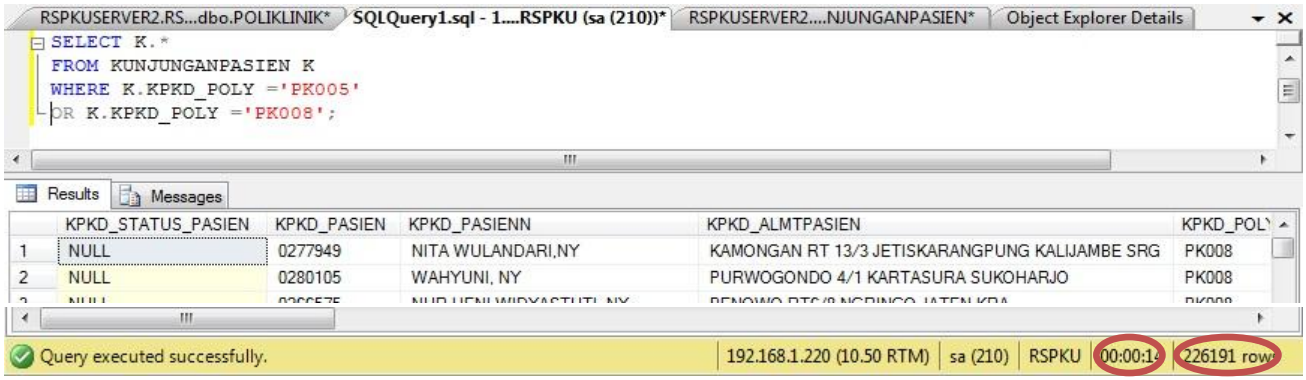

Fig 6: Query Example Without IN

Fig 6: Query Modification With IN

In Figure 5 shows the query result in the query example, it takes 14 seconds to execute 226,191 rows of query result while in figure 6 for query modification it takes 5 seconds to run 226.191 rows so that query modification is more efficient than query example.

Experiment #4

Eliminate DISTINCT Unnecessary Conditions

Considering the following example case, DISTINCT keyword in the original query is not necessary because table_name contains the primary key, which is part of the result set [2].

Query Example

```
SELECT DISTINCT * FROM
KUNJUNGANPASIEN K
JOIN POLIKLINIK P
ON K.KPKD_POLY= P.FMPKLINIK_ID
WHERE P.FMPPENUNJANG2 = 0;
```

Query Modification

```
SELECT * FROM KUNJUNGANPASIEN K
JOIN POLIKLINIK P
ON K.KPKD_POLY= P.FMPKLINIK_ID
WHERE P.FMPPENUNJANG2 = 0;
```

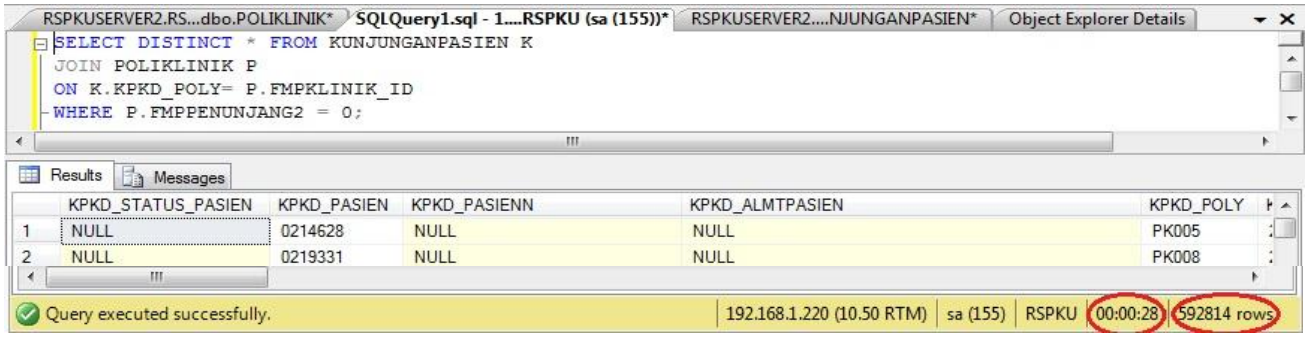

Fig 7: Query Example Using Distinct

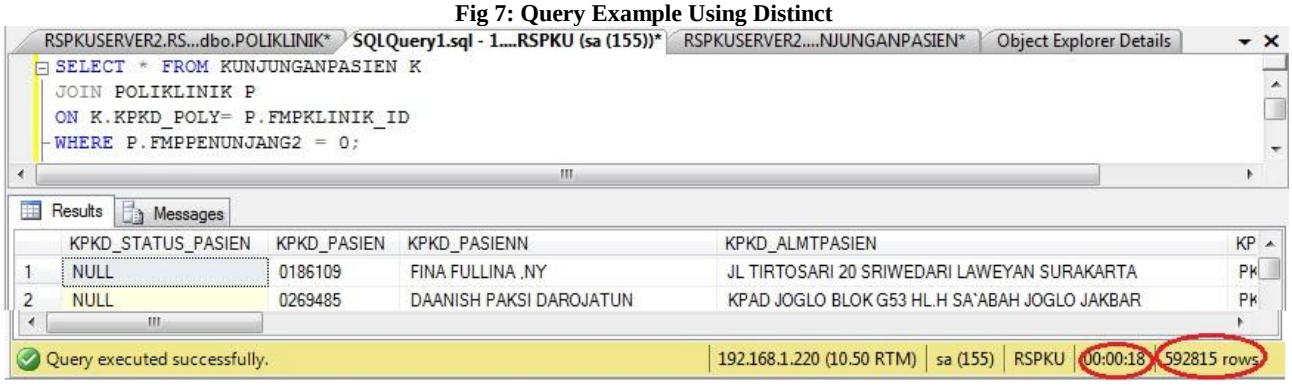

Fig 7: Query Modification Without Distinct

Figure 7 shows the query result in the query example takes 28 seconds to execute 592,814 lines of query results while in figure 8 for the query modification it takes 18 seconds to run 592.815 lines so that the query modification is more efficient than the query example.

The results of this study shows that the modification query can reduce the query execution time, the results can be seen in Table 1

Experiment	Query Example Time	Query Modification Time	Percentage reduction of time
Experiment #1	40 seconds	9 seconds	77.5 %
Experiment #2	11 seconds	10 seconds	9%
Experiment #3	14 seconds	5 seconds	64.2 %
Experiment #4	28 seconds	18 seconds	35.7%

Table 1: Precentage Reduction of Time Comparison

From table 1 we can see that the modification query takes less time in querying than the query example. We can see in the first experiment can reduce data processing time by 77.5 %, in the second experiment can reduce data processing time by 9%, in the third experiment can reduce data processing time by 64.2%, and in the fourth experiment can reduce data processing time by 35.7%. We can see the time required for query modification is more efficient than the query example.

5. CONCLUSION

Query optimization is a common task performed by database administrators and application designers to improve the overall performance of the database system. The purpose of this paper is to provide SQL scenarios to serve as a quick and easy reference guide for the development and maintenance of database queries. Even if you have an excellent infrastructure, its performance can be significantly degraded by inefficient demand.

Query optimization has a great impact on DBMS compliance and continues to evolve with more sophisticated new optimization strategies. So, we should try following the general tip as mentioned above to get a better query performance. Optimization can be achieved with some effort if we make it a standard practice to follow the rules.

The experimental results show a time reduction through a significant modification query when applied.

6. ACKNOWLEDGMENTS

We express gratitude to God Almighty, Parents, Family, PJJ Aptikom Batch 5 and friends who helped and support so that this work can be completed.

7. REFERENCES

[1] Wenjiao, B, Jiming, L, Jichao, T & LI, S. Query Optimization of Distributed Database Based on Parallel Genetic Algorithm and Max-Min Ant System. 2015 8th International Symposium on Computational Intelligence and Design, 2015.

[2] Habimana, J. 2015. Query Optimization Techniques - Tips For Writing Efficient And Faster SQL Queries. International Journal Of Scientific & Technology Research, 4, 22-26.

[3] Gordon C. Everest. 2005 *Fundamentals of Database System* Benyamin Tokyo.

[4] ARIPIN 2010. Meningkatkan Efektifitas Pengelolaan Database Dengan Optimasi SQL. *Techno.Com*, 9.

[5] Haas, S., Arnold, O., Scholzey, S., H¨Oppnery, S., Ellguthy, G., Andreas Dixiusy, Ungeth¨Umz, A., Mierz, E., N¨Othen, B., Mat´U¨s, E., Schiefery, S., Cederstroemy, L., Pilzx, F., Mayry, C., Sch¨Uffnyy, R. E., Lehnerz, W. & Fettweis , G. P. 2016. A Database Accelerator For Energy-Efficient Query Processing And Optimization.

[6] Lakshmi, S. V. & Vatsavayi, V. K. Query Optimization Using Clustering And Genetic Algorithm For Distributed Databases. 2016 International Conference On Computer Communication And Informatics (Iccci -2016), 2016 Coimbatore, India.