# Heuristic Query Optimization for Query Multiple Table and Multiple Clause on Mobile Finance Application

*by* I Ketut Gede Sudiartha

**Submission date:** 28-May-2023 11:28PM (UTC-0400)
**Submission ID:** 2104202929
**File name:** Indrayana_2018_J._Phys.__Conf._Ser._953_012049.pdf (462.38K)
**Word count:** 3165
**Character count:** 18682

# Heuristic query optimization for query multiple table and multiple clausa on mobile finance application

IOP Conf. Series: Journal of Physics: Conf. Series **953** (2018) 012049 doi:10.1088/1742-6596/953/1/012049

# Heuristic query optimization for query multiple table and multiple clausa on mobile finance application

**I N E Indrayana[1], N M Wirasyanti D P[2],I KG Sudiartha[3]**

[1,3]Electrical Engineering Department, PoliteknikNegeri Bali
Bukit Jimbaran, PO Box 1064 TubanBadung Bali
[2]Accounting Department, PoliteknikNegeri Bali
Bukit Jimbaran, PO Box 1064 TubanBadung Bali

email:eddyindrayana@pnb.ac.id

**Abstract.** Mobile application allow many users to access data from the application without being limited to space, space and time. Over time the data population of this application will increase. Data access time will cause problems if the data record has reached tens of thousands to millions of records.The objective of this research is to maintain the performance of data execution for large data records. One effort to maintain data access time performance is to apply query optimization method. The optimization used in this research is query heuristic optimization method. The built application is a mobile-based financial application using MySQL database with stored procedure therein. This application is used by more than one business entity in one database, thus enabling rapid data growth. In this stored procedure there is an optimized query using heuristic method. Query optimization is performed on a "Select" query that involves more than one table with multiple clausa. Evaluation is done by calculating the average access time using optimized and unoptimized queries. Access time calculation is also performed on the increase of population data in the database. The evaluation results shown the time of data execution with query heuristic optimization relatively faster than data execution time without using query optimization.

## 1.Introduction

The Internet enables applications to be massively accessible. An internet based application can be accessed by many users, anytime and anywhere. The number of users accessing an application will increase the data population on the server [1][2]. Access time will not be a problem if the number of record rows from the table in the database ranges from tens to hundreds of records, but will cause the problem of access time if the data record has reached tens of thousands to millions of records. In this condition, efforts are needed to keep the performance of the application [3][4]. Complex queries, either single query or multiple queries, there are usually subexpression in it, which performs join operations, sorting, selection, etc[5]. Subexpression need time for execution. In the fast growing digital world, efficient query and data access are the main factors and goals that must be achieved[6]. Database administrators (DBAs) must have high capabilities in query optimization techniques to maintain data access performance[7][8]. The query optimization problem is how to determine the best join operation execution time in a relational database. So the join operation can produce the fastest execution time[9][10]. Optimize query do comparison to the number of memory required by some

query plan .Query plan that produces limited time execution is are the major factors of a query optimizer[11][12] .Heuristic query optimization and sub query[13] approach to reduce communication time. Query join is separated into two parts, namely tree and cyclic. The result Conversion of cyclic queries into the query tree using different approaches such as adding nodes, deleting nodes, relationship decomposition and edge deletion can increase data execution time[14]. Performance SQL is also affected by the index creation in the table column used as the primary key. This primary key has a function to distinguish one record with another record[15].

Android application is one of the most massively accessible applications. So need to calculate data access time to maintain data access performance of the android application. The more data accessed and more users accessing the data at the same time, can slow down data access time.This android-based financial applications require efforts to maintain the performance of data access time. One such effort is the use of query optimization techniques using heuristic methods. This query optimization method transforms the query with a number of rules to speed up the performance of data execution in database. It tries to minimize the number of accesses by reducing the number of tuples and number of columns to be searched, finally to get optimal execution time[16] [17][18].Heuristics has always proved to be a useful tool. Sometimefor a small data, does not show significant performance, but once the data has grown will shows improved data access time[19].

## 2. Related Work

Sunita M. et al [14], conducted query optimization research. This research uses query heuristic optimization approach and semi-join approach to reduce communication time. Query join is separated into two parts, namely tree and cyclic. The result Conversion of cyclic queries into the query tree using different approaches such as adding nodes, node removal, relationship decomposition and edge removal can increase data execution time.

In the study of Vishal H. et al [16], Heuristic query optimization is applied. In this research optimization is done in the use of query select. As a result, data access time can be minimized because the number of tuples and the number of columns is reduced

Kumar S. et al [19]has conducted research using heuristic query optimization. In their research, they used the heuristic function approach to evaluate query search efficiency in database operations. Their simulation results show an increase in query search compared to traditional query searches. Therefore, they assume heuristic-based query optimization is a better approach.

Sofia G. et all [2], conducts research on distributed databases with query optimization. Communication Costs and the amount of data transmitted are factors involved in distributed databases. Semi join Operation and Join Operation are 2 options proposed to minimize the factor. The number of data transfers occurs more than using a semi join operation So sub operations are run dynamically to increase communication costs.

Gupta Er. Shobit [20] conducting research on web-based databases. Gupta performed a comparison of data execution between inline query techniques compared with query optimization in stored procedure using 8 remote mechine. The result of using query optimization in stored procedure is relatively faster than using inline query technique.

## 3.Research Method

In this study, we use some basic theories of relational algebra and heuristic query optimization methods. The original query will go through a heuristic optimization process to get efficient queries.

### *3.1 Relational Algebra*

In Relational Algebra, we can use several rules to transform an equation to another equation equivalent to it. Some operasion in relational algebra such as projection ($\pi$), selection($\sigma$), join ($\bowtie$) , Cartesian product (X), set Union ($\cup$), set Intersection($\cap$) can transform can turn into other equations without changing their result[14]. There are several theorems and definitions that we can use to perform query optimization. It transform from the original query to the optimized query , generate the same set of attributes. The order of the resulting attribute set does not matter.

*3.2.Heuristic Query Optimization Algorithm*

The heuristic query optimization algorithm uses some rules to transform the initial query into an optimal query. The heuristic query optimization algorithm[14][16][18][19]can be seen in the following figure

***Input:*** Relational Algebra Query
***Output:*** Best Evaluation Plan
***Algorithm:***
Step 1: Design the initial canonical tree of the query
Step 2: Move the SELECT operation down the query tree
Step 3: Apply more restrictive SELECT operation first. If the two relations are residing atsame site, they will be handled first.
Step 4: Replace CARTESIAN PRODUCT and SELECT with JOIN operation
Step 5: Move PROJECT operations down the query tree

**Figure 1**. Heuristic Query Optimization Algorithm

**4. Result and Discussion**

In this mobile finance application there are several tables that are related to each other. Tables is stored in a MYSQL database. This table will be retreivedby query select. As for the tables is:

**Table 1**. Sample of record table SimpleBusiness.

| SBId | Name | Address | Phone | ContactPerson |
|---|---|---|---|---|
| A000000001 | Candra Collection | Jl. Mekar II Blok A 1 No 8 Pemogan Denpasar | 081236 01XXX | KadekMelanSupardiani |
| A000000002 | Tri Jaya Shop | DesaTenganan Kec. Manggiskarangasem | 0812391XXXX | I WayanSurita |
| A000000003 | SinarBambu Handicraft | Br. KutaUndisan, Ds KayubihiBangli | 08124694XXX | I Wayan Wardana |
| A000000004 | Bala-Bali Handicraft | Jl. Raya UlakanGg. Anggrek No. 10 Amlapura | | I Gd PancaSunarbawa |

**Table 2**. Sample of record table User.

| UserId | UserName | FullName | password | SBId |
|---|---|---|---|---|
| U00001 | melan | KadekMelanSupardiani | XXXXXXXX | A000000001 |
| U00002 | rusmini | Ni WayanRusmini | XXXXXXXX | A000000001 |
| U00003 | Surita | I WayanSurita | XXXXXXXX | A000000002 |
| U00004 | sundari | Sundari | XXXXXXXX | A000000002 |
| U00005 | w4rd4na | I Wayan Wardana | XXXXXXXX | A000000003 |
| U00006 | pancaku | I Gd PancaSunarbawa | XXXXXXXX | A000000004 |

**Table 3**. Sample of record table Journal.

| JournalId | JournalNo | Date | Description | Evidence | SBId | UserId |
|---|---|---|---|---|---|---|
| 1 | GJ00000001 | 2017-05-01 | Received Capital | RC-0001 | A000000001 | U00001 |
| 2 | GJ00000002 | 2017-05-01 | Buy Inventory Items (Credit) | BG-0110 | A000000001 | U00001 |
| 3 | GJ00000003 | 2017-05-05 | Debt Payment | RC-0002 | A000000001 | U00002 |
| 4 | GJ00000001 | 2017-05-02 | Received Capital | BJ-01010 | A000000002 | U00003 |
| 5 | GJ00000002 | 2017-05-02 | Buy Inventory(Cash) | CS98988 | A000000002 | U00003 |
| 6 | GJ00000001 | 2017-05-10 | Received Capital | B000001 | A000000002 | U00003 |

**Table 4**. Sample of record table JournalDetail.

| JournalDetailId | JournalId | AccountNo | Debet | Credit |
|---|---|---|---|---|
| 1 | 1 | 11001 | 10,000,000 | 0 |
| 2 | 1 | 30001 | 0 | 10,000,000 |
| 3 | 2 | 12001 | 1,000,000 | 0 |
| 4 | 2 | 21001 | 0 | 1,000,000 |
| 5 | 3 | 21001 | 1,000,000 | 0 |
| 6 | 3 | 11001 | 0 | 1,000,000 |
| 7 | 4 | 11001 | 15,000000 | 0 |
| 8 | 4 | 30001 | 0 | 15,000000 |
| 9 | 5 | 12001 | 5,000,000 | 0 |
| 10 | 5 | 11001 | 0 | 5,000,000 |
| 11 | 6 | 11001 | 6,000,000 | |
| 12 | 6 | 30001 | | 6,000,000 |

One of the queries used in this application isquery to display “Journal and Journaldetail transactions from one Small Bussiness in Mei 2017 and Small Business ID=A000000001”.
Here is the original query:

```
SELECT Journal.JournalNo, Journal.Date, Journal.Description,Journal.Evidence,
JournalDetail.Debet, JournalDetail.Kredit, SmallBusiness.Name, User. FullName
FROM Journal,JournalDetail,SmallBusiness,User
 Where Journal.SBId=SmallBusiness.SBId AND Journal.UserId=User.UserId AND
SmallBusiness.SBId=”A000000001” AND Journal.Date>=”2017-05-01” AND
Journal.Date<=”2017-05-31” AND Journal.JournalNo=JournalDetail.JournalNo
```

Heuristic optimization process:

Step 1. Inisialization canonical query tree

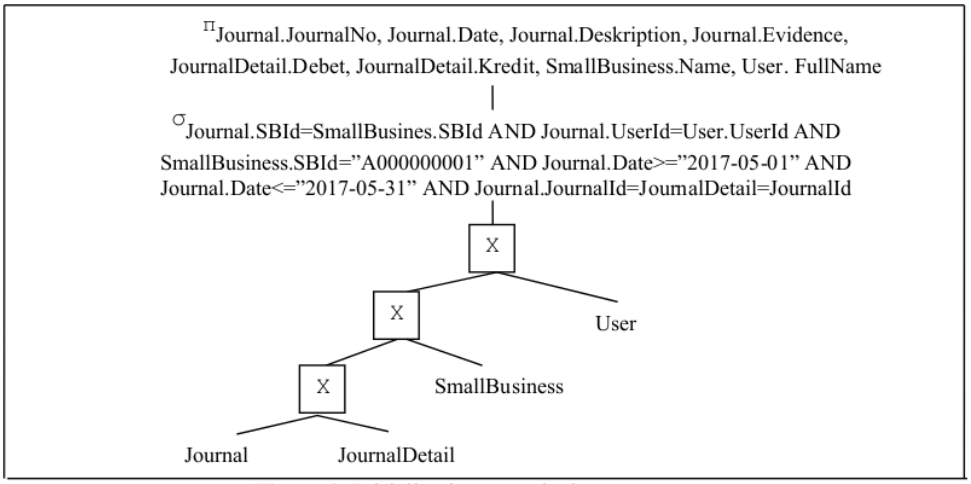


**Figure 2**. Inisialization canonical query tree

Step 2. Move the SELECT operation down the query tree

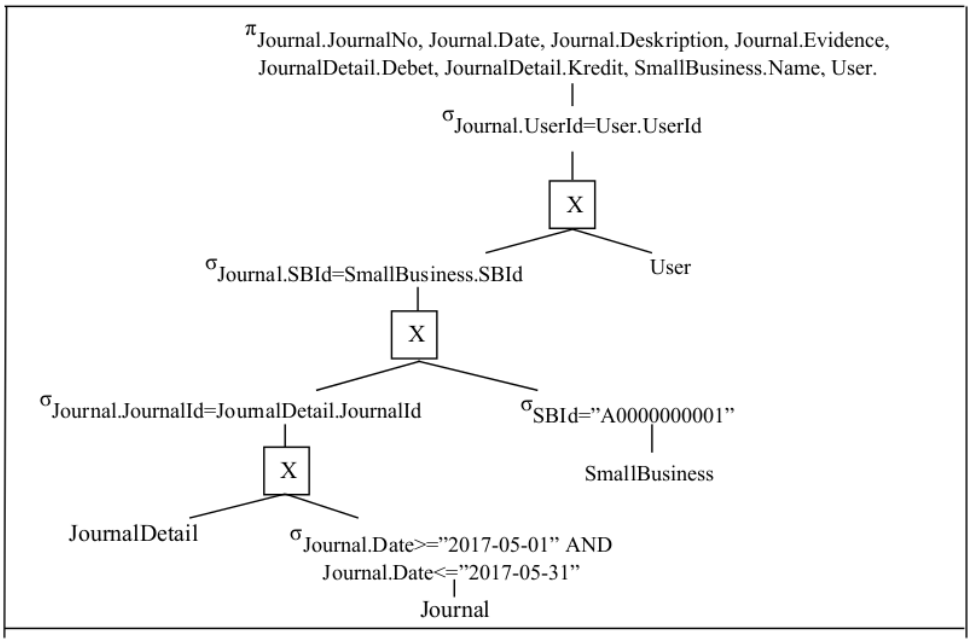


**Figure 3**. Move the SELECT operation down the query tree

Step 3.Apply more restrictive SELECT operation first

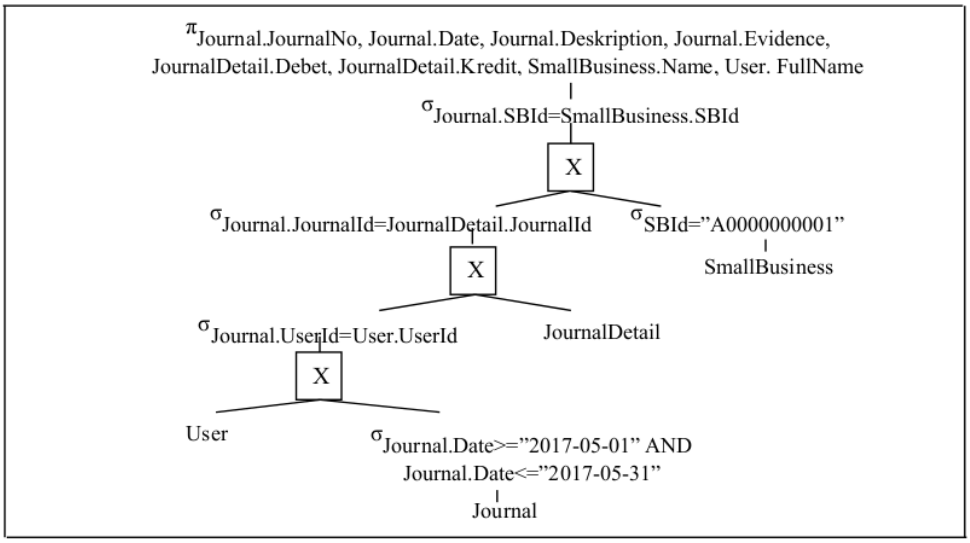


**Figure 4**. Apply more restrictive SELECT operation first

Step 4.Replace CARTESIAN PRODUCT and SELECT with JOIN operation

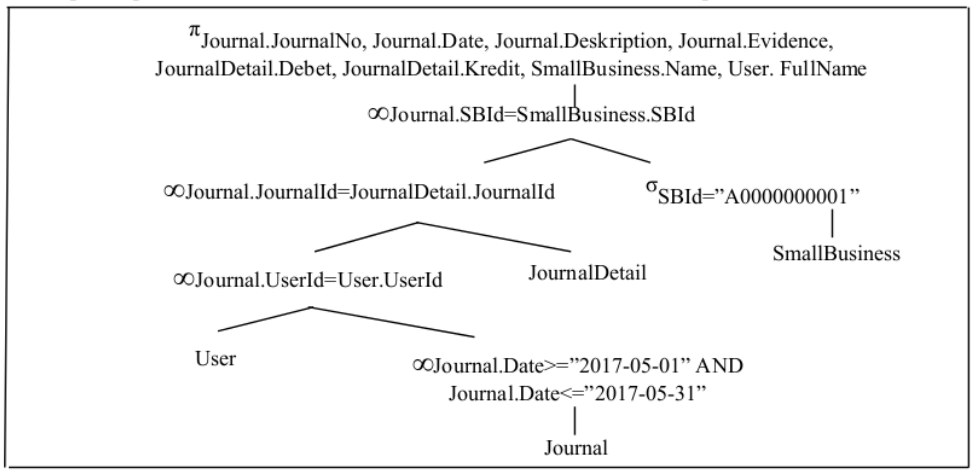


**Figure 5**. Replace CARTESIAN PRODUCT and SELECT with JOIN operation

Step5. Move PROJECT operations down the query tree


**Figure 6.** Move PROJECT operations down the query tree

Testing on the mysql database shows the execution time of data between queries with heuristic query optimization faster than the original query. Testing conducted by involving 4 tables namely Journal, JournalDetail, User and SmallBisnis.Journal table consists of 3000 tuples, JournalDetail consists of 8000 tuples, user consists of 400 tuples and SmallBusiness consists of 200 tuples. The result of eligible data execution, between original queries and queries optimized by heuristic method can be seen in figure 7.

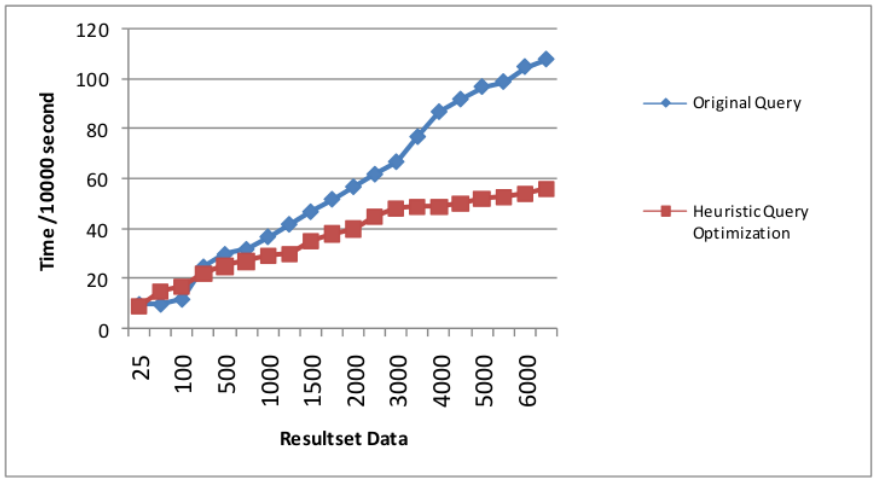


**Figure 7**. Performance between original query and heuristic query optimization

**5.Conclutions**

Heuristic query optimization makes the selected tuples in the table much less before join table operations. So at the time of join operation will produce a minimum tuples and requires a relatively short execution time. In this research, data execution speed uses heuristic query optimization, 23% faster than origimal query.

**6. Acknowledgments**

This research was support by the Directorate of Research and Community Service, Director General of Development and Research Enhancement, Ministry of Research, Technology and Higher Education, In accordance with the research contract: No. 023 / SP2H / LT / DRPM / IV / 2017.

**7. References**

[1] Gaarg Atul, Juneja Dimple, 2012, *A Comparison and Analysis of Various Extended Techniques Of Query Optimization*, International Journal of Advancement in Technology, ISSN 0976-4860,Vol 3 No 3.

[2] Sofia G, Rajinder S, Tirath S, 2014, *Executing Joins Dinamically in Distributed Database SystemQuery Optimizer*, International Journal of Computer Applications, (0975-8887), Volume 97. No 2.

[3] Khan Majid, Khan M.N.A, 2013, *Exploring Query Optimization Techniques In Relational Database*, International Journal in Databse Theory and Aplication, Vol 6 No 3.

[4] Bamnote G. R., Agrawal S.S, 2013, *Introduction to Query Processing and Optimization*, International Journal of Advanced Research in Computer Science and Software Engineering, Vol 3 Issue 7.

[5] Dokeroglu T., Bayir M. A., Cosar A. 2015 *Robust Heuristic Algorithms For Exploiting The Common Tasks Of Relational Cloud Database Queries* Applied Soft Computing Journal, Elsivier Vol 30 pages 72-82.

[6] Peter P.B., Werner M., Ralph V., Jurgen M., Erich S. 2010 A Heuristic Query Optimization Approach for Heterogeneous Environments IEEE/ACM 10th International Conference on Cluster, Cloud and Grid Computing (CCGrid)( Melbourne:IEEE).

[7] Jean H, 2015, *Query Optimization Techniques – Tip For Writing Efficient And Faster SQL Queries*, International Journal Of Scientific & Technology Research, Vol 4 Issue 10..

[8] Navita Kumari, 2012, *Sql Server Query Optimization Techniques – Tips For Writing Efficient and Faster Queries*, International Journal Of Scientific and Research Publications, Volume 2, Issue 6

[9] Frederico A.C.A Goncalves, Frederico G. Guimaraes, Marcone J.F. Souza 2014 *Query join ordering optimization with evolutionary multi-agent systems* Expert Systems with Applications Journal, Elsivier Vol 41 Issue 15 Pages 6934-6944.

[10] Gopal C.D, Jayant R.H. 2007 *Robust Heuristics for Scalable Optimization of Complex SQL Queries* IEEE 23rd International Conference on Data Engineering (ICDE) (Istambul:IEEE).

[11] Dokeroglu T., Tosun U., Cosar A. 2012 *Particle Swarm Intelligence as a New Heuristic for the Optimization of Distributed Database Queries* IEEE 6th International Conference on Application of Information and Communication Technologies (AICT)(Giorgia:IEEE)

[12] Bruno N., Galindo-Legaria C., Joshi M. 2010 *Polynomial heuristics for query optimization* IEEE 26th International Conference on Data Engineering (ICDE) (Long Beach, CA,USA: IEEE).

[13] Saurabh G, Gopal S. T.,Umashankar P. 2015 *A Survey On Query Processing and Optimization in Relational Database Management System,* International Journal of Latest Trends in Engineering Technology (IJTLET), Vol 5 Issue 1.

[14] Sunita M. Mahajan, Vaishali P. Jadhav 2012 *General Frame Work For Optimization Of Distributed Queries*, International Journal of Databse Management System (IJDMS), Vol 4 No 3.

[15] Limper M. 2014 An *SQL-based approach to physics analysis* , Journal Of Physics : Conference Series 513 (2014) 022022, IOP Publishing.

[16] Vishal Hatmode, Sonali Rangdale 2014 *Heuristic Based Query Optimization*, International Journal Of Advanced Research In Computer and Communication Engineering, Vol 3.

[17] Seema,Parminder Kaur 2013 *Query Optimization Algorithm Based On Relational Algebra Equivalence Transformation*, International Journal Of Engineering and Management Science, Vol 4 (3), 326-331.

[18] Vineet M, Kaushik A, Amandeep S.B. 2014 *Issues in Query Processing and Optimization,* International Journal of Modern Trends in Engineering and Research, International Journal of Modern Trends in Engineering and Research (IJMTER), Volume 01, Issue 05.

[19] Kumar S., Khandelwal G, Varshney A., Arora M. 2011 *Cost-Based Query Optimization with Heuristics*, International Journal of Scientific & Engineering Research Volume 2, Issue 9.

[20] Gupta Er. Shobit 2015 *Database Management – Inline Queries vc Stored Procedures – an Extended Analysis*, International Journal Of Emerging Technologies in Computational and Applied Science (IJETCAS), Page 234-238.

# Heuristic Query Optimization for Query Multiple Table and Multiple Clause on Mobile Finance Application

ORIGINALITY REPORT

| 4% | 3% | 1% | 0% |
|---|---|---|---|
| SIMILARITY INDEX | INTERNET SOURCES | PUBLICATIONS | STUDENT PAPERS |

MATCH ALL SOURCES (ONLY SELECTED SOURCE PRINTED)

1%

★ ejournal2.litbang.kemkes.go.id

Internet Source

| | | | |
|---|---|---|---|
| Exclude quotes | On | Exclude matches | Off |
| Exclude bibliography | On | | |