Book Recommendation Using *Neo4j* Graph Database in BibTeX Book Metadata

I Nyoman Pande Wahyu Dharmawan, Riyanarto Sarno
Informatics Department, Faculty of Information Technology Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
nyoman.wahyu13@mhs.if.its.ac.id, riyanarto@if.its.ac.id

***Abstract*—In digital era, book has an important role in life. There are not only a lot of books for different purpose. But also, there are many book metadata which can use for another reason, such as book recommendation. By processing the book metadata, an information can be given to user that needs book recommendation. By combining BibTeX book metadata and Graph Database from Neo4j, data from metadata can be processed. Then, with cypher query by inputting author's parameter or book type's parameter, user can get book recommendation based on their input's criteria. The result is exactly the same with process the metadata manually in relational database. Neo4j, from this paper, takes 180 milliseconds to execute cypher query with author's criteria and takes 184 milliseconds to execute cypher query with book type's criteria**

Keywords—book recommendation; bibtex; graph database; Neo4j

I. INTRODUCTION

Book has become very important in human's life recently. There are a lot of books with different purpose and different author. For instance, there is a book for fun like a novel, book for education and also book for hobbies. In addition, the field of recommender system, from an International Retrieval (IR) perspective, has been recommended by internet based on filtering [1]. During last decade, web users have increased rapidly. In addition, a lot of book metadata in internet comes, such as BibTeX metadata, GoodsRead metadata, and Dublin Core metadata. Metadata, Website and resources recommender system actually can give each other beneficial situation [2].

With the condition in technology, some researches focus on changing metadata into relational database schema. Other researches make their main focus as transforming direct metadata into ontologies [3]. In this paper, researcher conducted book recommendation using graph database and book metadata that is stored in Neo4j. Neo4j itself is a graph database allowing storing data as nodes that connected by arcs. Because metadata can be easily represented as graph [4]. Neo4j provides a cypher query to extract data from graph database.

In the previous work, relational database has been proposed to create a movie recommender system using MySQL. The rating data from the Netflix database are used by the system where linking to the content as well as the information conducted by Internet Movie Database [5]. The system creates a HTML website and uses MySQL relational database. On this paper, researcher focuses on providing the user with variation of recommendation system [6]. Besides, there is also a research that creating Personalized Movie Database System (PMDS). It is a dynamic web application created for purpose to see information about movie [6]. PMDS uses PHP as web programming and MySQL database as relational database [6].

II. STUDY LITERATURE

In the introduction, the paper has already explained that BibTeX metadata will be processed using graph database to occur time execution in producing book recommendations.

A. NoSQL database and Relational database

Recently, there has been a popular investigation in data store that does not use SQL database which is so popular. One of the new data store is called NoSQL [7]. Yet, there are a lot of company still use relational database for their ERP (Enterprise Resource Planning). ERP itself has become the reference of process management's usage [8]. Before a company knows which ERP feature fits with his company's vision, the company has to determine the right model of their business process [9]. There is one method, for example, to determine the model, which is by using event log [9]. It also can be stored in the lexical database, which means there will be similarity in some event log. To avoid a duplication between event log, semantic string similarity checking is one of good method in determining semantic similarity [10]. Later, the next step comes up with decision mining [11]. Decision mining is a combination between process mining with machine learning that is used to bring forward an information about the effects of a case in business process [11].

On the other hand, fraud detection also takes an important role in the ERP system. Fraud recently has caused a disappearance of 1,4 billion US Dollar in 96 countries whereas there were 1,388 frauds happened [12]. There are several methods which can detect the anomaly that was done in ERP, one of them is called process mining techniques. Process mining is used to detect anomalies by analyzing the running processes [12]. However, nowadays customers are tended to confuse about their needs. Customers realize that it is the time that ERP system should be more flexible [13]. So, vendors

978-1-5090-5864-8/17/$31.00 ©2017 IEEE

have to make the ERP and the business process itself as flexible as needed [13]. Because of it, ERP comes up with NoSQL database to provide its flexibility [13].

This paper however uses graph database as NoSQL derivative. Virtually every graph database has to store and query the graph like relational database in general. There has been also an increase of interest in graph that representing social network and web site link structure [7]. Yet, there has been an increase of data complexity which is no longer compatible with relational database. The usage of graph database is not limited on solving web site link structure. Transportation, protein – interaction and business networks can be modelled as graphs with no limitation of the data size [14]. One of an application that is quite popular in representing graph database is Neo4j. It has SparQL plugin which can connect with RDF data. It supports variety of programming languages such as Java, JavaScript, Python, .Net, PHP, Scala, and others [15].

B. Book Metadata using BibTeX

Metadata is a set of data. It tells a lot of description in web information, such as identity card, library card, and book recommendation [16]. BibTeX is a metadata that store book data. It has several datatypes. Such as hasKey, hasGenre, hasTitle, hasDescript. Each datatype contains metadata which can be used to give new information. BibTex is readable and also easily creates relationship between book metadata with user and determines that user want that book or not. BibTeX book metadata nowadays takes an important role as a tool to collect data becoming useful [16].

C. Graph Database Recommended System

Until now, there has been a lot of works on rebuilding new approaches for recommender system during the last decade. However, it is still shortfall in recommending's performance. To improve that, some researches in representing and saving of large data [17] are being conducted. In order to give a good recommendation, big data should be processed quickly and accurately.

Connected data has been managed in graph database with effective and flexible [18]. The graph actually consists of nodes that has property and relationships between nodes. Querying graph database gives lower cost if it is compare to querying relational database. Because graph database does not need joins cost to answer simply question such as 'who is John's friends of the friends?' [18].

D. Neo4j

Neo4j is a database application that using graph database. This Neo4j is working by storing the data into nodes that connected by relationship. Because firstly, graphs are one example that provide a good spot in representing metadata. Secondly, graph database provides a natural abstraction in storing metadata compare with relational database. Third, with its flexibility, graph database allows users to update new relation and new data [5].

1) Cypher query in Graph Database

Neo4j provides a query like in MySQL or oracle for relational database. It is called cypher query. SQL and

```
match(Novel:BookType{name:"Novel"})
match(SummerinSeoul:Book{hasTitle:"Summer in Seoul"})
create(SummerinSeoul)-[hasbooktype:HasBookType]->(Novel)
```

SPARQL are being referenced by cypher query in graph database [4], and Neo4j also provides effective and expressive query [19].

Fig. 1. Example of Cypher query.

Fig. 1, represents an example of Cypher query. The purpose query is to create a relationship between book type nodes and book nodes. The relationship itself is *hasBookType.*

III. Book Recommendation Using Graph Database

In order to create book recommendation from BibTeX book metadata, the metadata should be represented in graph database in Neo4j. After the metadata, has been represented in graph database, users can get book recommendation by common author or by common book type using cypher query.

A. Dataset

For book dataset, a BibTeX metadata is represented in graph database. There are ten book data that have property *hasTitle, hasYear, hasGenre, hasBookType,* and *hasAuthorName*. When representing into graph database, the metadata become three nodes. Book node that has property of *hasTitle, hasYear, hasGenre,* and *hasDescript*. Author node that has property of *hasName*, and Book type node that has property of *hasBookType*. Each node has their own relationship.

1) Book Node

BibTeX metadata for book are represented in book node. To create book node in Neo4j, the paper use cypher query by creating node that has book as a node name. Then cypher also creates node's property which has *hasTitle, hasYear, hasGenre,* and *hasDescript* as parameters.

```
create (limacm:Book {hasTitle:"5cm", hasYear:"2007",
hasGenre:"Inspirasi", hasDescript:"Novel ini menceritakan
tentang perjalanan 5 sahabat yakni Arial, Riani, Zafran,
Ian dan Genta. Novel ini mengajarkan tentang
harapan, impian, tekad, cinta dan persahabatan"})
```

Fig. 2. Book Node Cypher query.

In Fig. 2, book node is created with has book node as limacm. It has properties hasTitle as 5cm, heasYear as 2007, hasGenre as inspirasi and hasDescript as Novel ini menceritakan tentang perjalanan 5 sahabat yakni Arial, Riani, Zafran, Ian dan Genta. Novel ini mengajarkan tentang harapan impian, tekad, cinta, dan persahabatan. However, the paper has created 11 book nodes that has different property values from BibTeX book metadata. After creating all the book nodes, Neo4j will has 11 nodes that have book as the type of nodes.

The Neo4j book nodes representation will be explained in Fig. 3.

Fig. 3. Book Node Representation in Neo4j.

2) Book Type Node

This node is represented to create a parameter for querying book recommendation. These are containing two nodes, hobby and novel as book type node. Each node contains hasName property.

```
create(Novel:BookType{name:"Novel"})
create(Hobby:BookType{name:"Hobby"})
```

Fig. 4. Book Type Node Cypher Query.

From Fig. 4, which is representing the cypher query, two nodes containing book types are created, novel book type and hobby book type. Each node contains name property that represent node's name. After the cypher query in Fig. 4, being executed, Neo4j instantly create two nodes that is represented in Fig. 5.

Fig. 5. Book Type Representation in Neo4j

3) Author Node

This node represents author as a node. Author should become node because in the BibTeX book metadata, one author can make more than one book. So, to show that condition in graph representation in Neo4j, author should be separated from book and become different node. Author's node itself has one property, which is name property.

```
create(adikusrianto:Author{name:"Adi Kusrianto"})
create(agnesdanovar:Author{name:"Agnes Danovar"})
```

Fig. 6. Author Node Cypher Query.

The cypher query in Fig. 6, represents two author nodes. First node has *Adi Kusrianto* as a name. Second node has *Agnes Danovar* as a name. These two authors, in BibTeX book metadata, write all of 11 books that are already store in Neo4j as book nodes. The graph representation in Neo4j will be explained in Fig. 7.

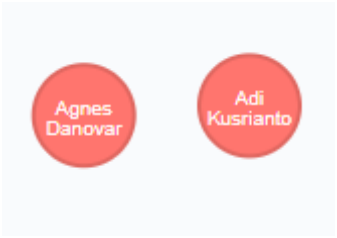

Fig. 7. Author Representation in Neo4J.

4) Relationship

In this section after all BibTeX metadata book has been represented in Neo4j as nodes, in order to create complete graph database, each node should have relationship. From BibTeX metadata book, there are two kinds of relationships, has Book Type and has Author. Neo4j provides cipher query to create a relationship between nodes.

In Fig. 8, book node which has title 5 cm connected to book type node called Novel. It means that book node has relationship with book type node, and their relationship is called novel.

```
match(Novel:BookType{name:"Novel"})
match(limacm:Book{hasTitle:"5 cm"})
create(limacm)-[hasbooktype:HasBookType]->(Novel)
```

Fig. 8. Has Book Type Relationship Cyper Query.

The advantage using that database, graph database actually does not have actual term like relational database. It is far more flexible. Graph database can adjust their own relationship by declare their parameter first. After that, by using cypher, relationship should be created. But, if the graph database has a lot of nodes, each node should manually create, using query, one by one connect the relationship to nodes. It is still unknown how to implement loop in cypher query in Neo4j. So, to connect all of the book nodes and boo type nodes, query in Fig. 8, should be repeated eleven times. After creating book type relationship, the next step is connecting author nodes with book nodes with *hasAuthor* relationship. In Fig. 9, node named *Adi Kusrianto* is the author of book title Summer in Seoul. So Neo4j will add relationship between that two nodes.

```
match(adikusrianto:Author{name:"Adi Kusrianto"})
match(SummerinSeoul:Book{hasTitle:"Summer in Seoul"})
create(SummerinSeoul)-[hasbooktype:HasAuthor]->(adikusrianto)
```

Fig. 9. Has Author Relationship Cypher Query.

After all of the relationship is defined, the BibTeX book metadata is completely represented in graph database. In Fig. 10, complete graph database is shown based on BibTeX book metadata.

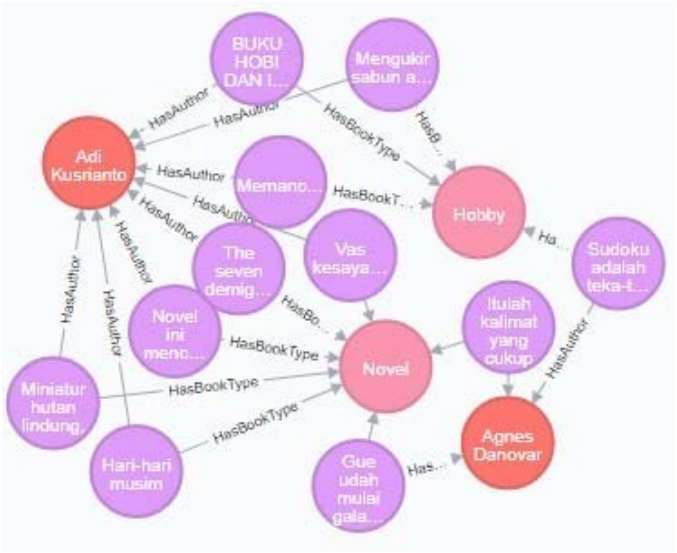

Fig. 10. Complete BibTeX Book Metadata Graph.

From Fig. 10, there are four book nodes that are connected to hobby book type nodes and there are seven book nodes that are connected to novel book type nodes. Also, there are eight book nodes that are connected to author nodes named *Adi Kusrianto*, and there are three nodes book that are connected to *Agnes Danovar*. There are 15 nodes and there are 20 relationships represented BibTeX metadata book.

B. Book Recommendation Cypher Query

After all of BibTeX book metadata represented in Neo4j graph database, by using cypher query, the researcher figures a book recommendation. The book recommendation parameter is by matching the author of the books or matching the book type of the books.

1) Author Matching Cypher Query

One of the parameter to get book recommendation for users is matching the nodes that have same author's name with the input. In the Fig. 11, below, the cypher query is to get all the book node data with conditions.

```
select by author :
Match (R:Author{name:'Adi Kusrianto'})
match (p:Book)-[:HasAuthor]-(R)
Return p
```

Fig. 11. Author Matching Cypher Query.

In Fig. 11, the query is about giving book recommendation by giving all of the book information with parameter author. User inputs name of the author. In that case, the author is "Adi Kusrianto". When the query is executed, Neo4j will match "Adi Kusrianto" with node that has type author. Then, Neo4j will find all the relationship that the node has. In row 3, Neo4j will filter the relationship and only select the relationship which is *HasAuthor*. Later, Neo4j will represent the book nodes that are fulfill the criteria.

Fig. 12. Author Matching Result.

In Fig. 12, the result was shown in Neo4j graph database after the cypher query was finish executed. Fig. 12, shows that it has eight nodes shown that match the query. So, there are eight book that is recommended to user when the input of the author is "*Adi Kusrianto*".

2) Book Type Matching Cypher Query

Another parameter to get book recommendation for users is matching the nodes that have book type name with the input. In the Fig. 12, the cypher query is to get all the book node data with that conditions.

```
select by booktype :
Match (T:BookType{name:'Novel'})
match (p:Book)-[:HasBookType]-(T)
Return p
```

Fig. 13. Book Type Matching Cypher Query.

In Fig. 13, the query is about giving book recommendation by giving all of the book information with parameter book type. In Fig. 13, user wants to find a recommendation book that the book type is novel. When the query is executed, Neo4j will find the node that has the same criteria as input. After get the novel's book type node, Neo4j will find the relationship that the selected node has. Neo4j will filter the relationship and only choose *HasBookType* relationship. Later, after all criteria are matched, Neo4j will represent the book nodes.

Fig. 14. Book Type Matching Result.

In Fig. 14, the result was shown in Neo4j graph database after the cypher query was finish executed. Fig. 13, shows that it has seven nodes shown that matched the query. So, there are seven book that is recommended to user when the input of the book type is "Novel".

IV. EXPERIMENT RESULT

The paper experiment methodology designed to compare the time execution between Neo4j database and MySQL database [6] in executing author query parameter and data type query parameter that is already made before. Each query was executed five times to get the average of the execution time both in Neo4j database and MySQL database. By calculating the execution time, a database can be judged whether it is fit to store a massive data and give back an information as fast as it supposed to be. In this paper, 10 BibTeX book metadata are represented in graph database using Neo4j and execute the cypher query for book recommendation. The data itself also stored in MySQL database as many as Neo4j data. All of the execution times were recorded in milliseconds (ms). The execution time result show that Neo4j execute quite fast. Table I shows the time that Neo4j needs to execute the query.

TABLE I. EXECUTION TIME OF QUERY

Query	Execution time				
	1	*2*	*3*	*4*	*5*
Book recommendation using author's query parameter	180 ms	90 ms	129 ms	134 ms	116 ms
Book recommendation using data type's query parameter	184 ms	96 ms	116 ms	94 ms	130 ms

Main criteria for good database are respond time and the result itself [5]. Compare to relational database that the researcher creates for guidelines, Neo4j show same result with relational database for book recommendation by using author's parameter or book type's parameter. It means that Neo4j also gave exact result for book recommendation.

Another comparison is effectiveness of the nodes with table relations. There are two different structure shown from relational database and graph database.

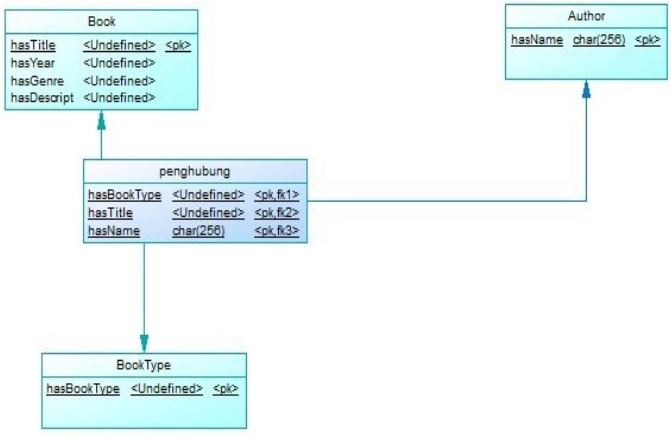

Fig. 15. Physical Data Model of Metadata.

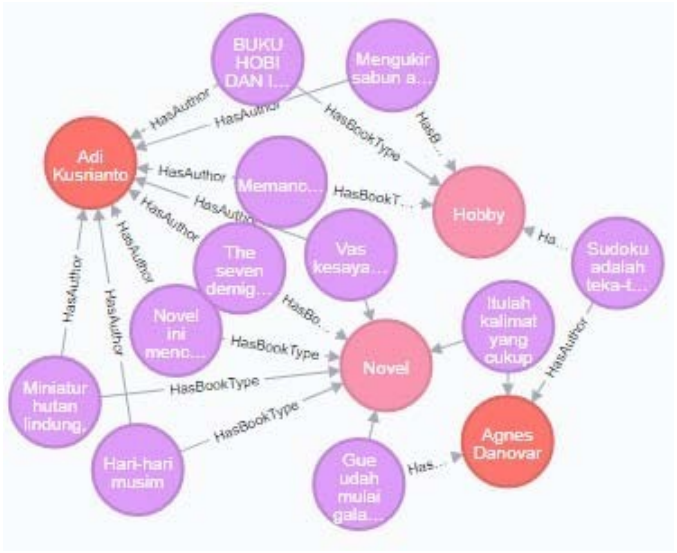

Fig. 16. Graph Representation in Neo4j.

Fig. 15, shows the relation between every table in representing BibTeX book metadata. There are three main table in Fig. 15, book table, book type table, and Author table. Each table has their own attribute with their primary key. Each table actually cannot connect directly. It is because every table is categorized strong entity and has characteristic many – to – many relations. In relational database, table which has many – to – many relations should connect by using middleman table. That's why *penghubung's* table was created. His main job is to connect relation between book, book type, and author table. In penghubung's table, there are attribute that come from primary key attribute in each of strong entity. *Penghubung's* table has *hasName* attribute from author's primary key table, has *hasBookType* attribute from book type's primary key table, and has *hasTitle* attribute from book' primary key table. After all the table connected, the database can be inserted with data from the metadata. Because of its rules, after insert the data from book's table, author's table, book type's table, data in *penghubung's* table must be declare first before the relational database can be queried to perform book recommendation. It is far different from the rules in graph database in Fig. 16. To perform simple query, graph database gives easier way to provide the database. Table in relational database, are represented as node in graph database in Neo4j. Each of node does not need assign a primary key attribute. Each of node are not distinguished as strong entity. So, every node can connect directly to each other by using relationship. It is one of the graph database's characteristic with its flexibility.

Last, researcher compares execution time of the query from the relational database and graph database using Neo4j. Query itself were executed 5 times to get the exact number by get the average number of each database. The result of the average of the execution time both in relational database and Neo4j database are shown in Table II.

TABLE II. COMPARISON OF EXECUTIONTIME

Query	Metadata	Database	Average of execution time
Author	Bibtex	Graph Database	129.8 ms
	Bibtex	Relational Database	15.4 ms
Book type	Bibtex	Graph Database	124 ms
	Bibtex	Relational Database	12.6 ms

On this paper, author query and book type query were also being executed in relational database. In this case, the paper uses MySQL database as relational database for comparing with Neo4j graph database. The query also being executed five times in MySQL database. From Table II, the average of the execution time of BibTeX graph database is quite long. Compare to relational database, it has huge different millisecond in execution time. When author query was being executed, relational database only took 15.4 MS. On the other hand, Neo4j took longer time which got 129.8 on the average of time execution. It also happened in book type query execution time. Neo4j took longer time which got 9.8 times than relational database. It means that relational database is faster than graph database in executing same parameter query and same amount of data.

V. CONCLUSION

The paper has introduced about graph database usage in book recommendation. The paper considers a graph database oriented to represent BibTeX book metadata to create a book recommendation.

From the comparison between graph database and relational database, both relational database and graph database can represent the data very well. These two kinds of database can represent book recommendation database which has book and author table. But in storing book recommendation into database, graph database is more flexible than relational database. Nodes from all graph database can connect directly from one node to another. It makes more efficient in preparing data before querying it. Given that in book recommendation, the database should be as flexible as possible because user rarely know about what book they needed. So, it will be a lot of possibility in giving advice to user. But when the data is being querying, graph database takes more time when executing the same query as relational database. Graph database takes 100times millisecond extra. As the result, graph database fit with the data that need high flexibility but takes more times to execute the query.

REFERENCES

[1] D.S. Eugenio, "Movie Recommendation with DBpedia," Politecnico di Bari – Via Orabona, 4,70125 Bari, Italy.

[2] Z. Fattane, K. Mohsen, and P. Samar, "Using Graph Database for File Recommendation in PAD Social Network," 2014 7th International Symposiun on Telecommunciations(IST'2014).

[3] Y. Shihan, and W. Jinzhao, "Mapping Relational Databases into Ontologies through a Graph – based Formal Model," 2010 Sixth International Conference on Semantics, Knowledge and Grids, doi: 10.1109/SKG.2010.33.

[4] K. Petri, and I. Mitsuru, "Improving Semantic Queries by Utilizing UNL Ontology and a Graph Database," 2012 IEEE Sixth International Conference on Semantic Computing.

[5] R. JaIdert, and V. Tessa, "A Simple Hybrid Movie Recommender System."

[6] G. Jayaprakash, 2015, "Personalized Movie Datatabase System."

[7] V. Chad, M. Michael, Z. Zhendong, N. Xiaofei, C. Yixin, and W. Dawn, "A Comparison of a Graph Database and a Relational Database A Data Provenance Perspective."

[8] S. Riyanarto, G. Hari, W.P. Endang, and S. Dwi, "Clustering of ERP Business Process Fragments," 2013 International Conference on Computer, Control, Informatics and Its Applications (IC3INA), 2013, pp. 319-324, doi: 10.1109/IC3INA.2013.6819194.

[9] S. Riyanarto, A.W. Widyasari, Kartini, and H. Fitrianing, "Determining Model Using Non – Linear Heuristic Miner and Control – Flow Pattern," *TELKOMNIKA* (Telecommunication Computing Electronics and Control), 2016, pp. 349-359, http://dx.doi.org/10.12928/telkomnika.v14i1.3257.

[10] S. Riyanarto , W.P. Endang, and M. Abdul, "B- BabelNet : Business – Spesific Lexical Database for Improving Semantic Analysis of Business Process Models," Telkomnika Vol 15 No 1 : March 2017, 2017.

[11] S. Riyanarto, L.I.S. Putu, G. Hari, S. Dwi, and M. Imam, "Decision Mining for Multi Choice Workflow Patterns," 2013 International Conference on Computer, Control, Informatics and its Applications (IC3INA), 2013, pp. 337-342, doi: 10.1109/IC3INA.2013.6819197.

[12] S. Riyanarto, Frenandes, S. Dwi, and M. Abdul, "Business Process Anomaly Detection using Ontology – based Process Modelling and Multi – Level Class Association Rule Learning," The 2015 International Conference on Computer, Control, Informatics and its Applications (IC3INA), 2015, doi: https://doi.org/10.1109/ic3ina.2015.7377738.

[13] S. Riyanarto, W.P. Endang, S. Dwi, and Sarwosri, "Business Process Composition based on Meta Models," International Seminar on Intelligent Technology and its Applications (ISITIA), 2015, doi: https://doi.org/10.1109/isitia.2015.7219998.

[14] S. Jouili, and V. Valentin, 2013, "An Empirical Comparison of Graph Database", SocialCom/PASSAT/BigData/Econ/Com/BioMedCom.

[15] A.P.E. German, and S.P. Oswaldo, "A Comparison of NoSQL Graph Databases," Computing Colombian Conference (9CCC), 2014, doi: 10.1109/ColumbianCC.2014.6955355.

[16] G. Hao, J. Chen, and T. Guo, "Web – Oriented Book Domain Metadata Description Research," 2010 International Conference on Intelligent Computing and Cognitive Informatics, 2010, doi: 10.1109/ICICCI.2010.13.

[17] G. Ankit, J. Rohan, and S. Shiwei, "Movie Recommendations Using Social Networks." 2008.

[18] H. David, and J. Jan, "Using Neo4j for Mining Protein Graphs : A Case Study," 2015 26th International Workshop on Database and Expert System Applications, 2015, doi: 10.1109/DEXA.2015.59.

[19] E. Mahmoud, R. Mohamed, A. Mostafa, and -B.M.S Abdel, "Large – Scale Ontology Storage and Query Using Graph Database – Oriented Approach: The Case of Freebase," 2015 IEEE Seventh International Conference on Intelligent Computing and Information System (ICICIS'15), 2015, doi: 10.1109/IntelCIS.2015.7397191.