Graph Database on Medical Research Data for Integrated Life Science Research

Aly Lamuri
Jonathan Midlando Adithama Purba
Adrian Reynaldo Sudirman
Randy Sarayar

Abstract

Indonesia is having an increasing surge of published scientific articles during recent years. In medical science, published articles greatly vary from both pre-clinical and clinical studies where each study possesses different methodological approach and hypothetical premises. However, some articles do not include a rigorous documentation as to make it reproducible. Moreover, the lack of centralized database further impedes researcher from reanalyzing previous findings and integrating them with the new study. This paper delineates such an issue by constructing a graph database to centralize and integrate clinical research data. Database is constructed using `Neo4j` and `cypher` querying language populated with 5,000 medical records generated by `synthea` program. We addressed the viabilities of our proposed data curation method by simulating data of different size. Our database able to answer queries requiring complex relationship while minimizing the amount of database hits. As a concluding remark, graph database is quite performant to solve data integration and centralization issue faced by life science research institutes.

1 Introduction

Scientific publication in Indonesia underwent manifold increases within the past decades. Reported by Maula, Fuad and Utarini (2018), numbers of published article on dengue-related subject increased 13 times in 2017 as compared to 2007. Such an increase also followed by *h-index* improvement, resulting in Indonesia placed as the 5$^{\text{th}}$ most scientifically productive ASEAN country in investigating dengue-related topic (Maula, Fuad and Utarini, 2018). Another bibliometric analysis investigated by Sarwar and Hassan (2015) also enlisted Indonesia within 11 most scientifically productive Islamic countries. However, these articles often neither robustly elaborate the methodological procedure nor provide obtained data for reanalysis, two factors contributing to reproducibility and credibility issue in scientific publication (Pashler and Wagenmakers, 2012; Stark, 2018; Resnik and Shamoo, 2016). Besides enabling preprint access (Oakden-Rayner, Beam and Palmer, 2018) and thorough documentation on methodology, data availability is also a crucial component for reproducibility in science (Peng, 2015). Therefore, we proposed utilizing graph database to integrate research findings in life science-related fields.

1.1 Graph Database

Data management system should appropriately consider interoperability and scalability which enable data storing, indexing and retrieving. Databases aggregate integrated object in a structure defined by its metadata. The presence of metadata implies a self-defined property of the database, whereas in relational database management system (RDBMS) such definition included within its particular schema (Berg, Seymour and Goel, 2012). During the development of RDBMS, emerging is the need to quickly retrieve the data through syntactically and logically feasible manner, therefore inducing the conceptual design of `SQL`, a structured querying language. However, with data being stored in a multi-tabular layout, relational database (RDB) faced massive disadvantages in handling highly-connected data. Hence the development of schema-less database initiated by NoSQL (Berg, Seymour and Goel, 2012; Fabregat *et al.*, 2018), with graph database being one of its variants (Oussous *et al.*, 2015).

Graph database is more performant in storing data with intricate relationships, e.g protein interactions or chemical reaction pathways, as compared to its RDB counterparts (Fabregat *et al.*, 2018). `Neo4j` is a graph database platform developed in `java` and compliant towards ACID system (Atomicity, Consistency, Isolation, Durability) (Oussous *et al.*, 2015). As a native graph database, `Neo4j` shall store data as explicitly defined relationships in a schema-less management system. Therefore, `Neo4j` treats database querying as a graph traversing process. This redeeming feature of graph database in general enables higher performance and flexibility in storing the data. `Neo4j` employs `cypher` as a querying language to define patterns on traversing the relationship graph. Furthermore, ASCII-Art syntax of `cypher` enables a more intuitive querying process. Such uniquely written language and ACID-compliant platform could become a two-fold advantages to use `Neo4j` in delivering graph database management system.

1.2 Medical Informatics

Information in life science-related fields often possess an intelligible relationship of causative nature. Many of such information may present as a connection between one entity to another. Interractome, reactome and connectome are common examples we may find in currently emerging basic science research. In translational research paradigm, some interests highlighted the importance of genetic and proteomic interaction network. Meanwhile in clinical settings, we may also want to consider patient-doctor-institution as separate yet related entities. Therefore, the nature of data in medicine is actually a close resemblance of entity-relationship data. Indubitably, we shall consider applying graph database as an alternative to RDB to store life science-related research data.

2 Methodology

This study utilized a machine with Intel Core i7-7700HQ, 8GB of DDR4 RAM and 5400 RPM spinning hard disk. We employed `Neo4j` as a platform to create graph database with `cypher` as the querying language. Data used in this study are generated from `synthea` program, producing 5,000 – 50,000 medical records in json-based FHIR (Fast Healthcare Interoperability Resources) which directly converted into `*.csv` format. As shown in figure 1, we treated each entity as vertex and underlying relationship as an edge connecting two vertices. We first design constraints for unique input and indices for redundant vertices. To prevent random access memory (RAM) bottleneck, we enabled periodic commit for each 500 inputs, which especially beneficial when dealing with numerous entries. Afterwards, we load `*.csv` file generated by `synthea` as a query object and finally set the entity and relationship.

To measure performance of our proposed database, we created a log containing time consumption and number of created objects which include nodes, relationships, graph property and graph label. Said database model took data of various sizes as input: 5,000, 10,000, 20,000 and 50,000.

Considering exponential increment in our data, we applied power transformation according to Tukey's ladder of power to normalize the data. Anderson-Darling test then employed to challenge normality assumption. We computed correlation estimate between time and created objects based on p-value obtained from normality test. Kendall's tau estimated the correlation when any of imputed variable has $p < 0.05$ and Pearson's in otherwise cases. Data fitted into generalized linear model (GLM) with Gaussian link function.

Simulation process involved two different queries on all dataset. Database hits (db-hits) and time measured the efficacy in handling such queries. Results on simulation presented in bar plot to demonstrate the scalability of our database model. Queries written in `cypher` and presented as follow:

```
// List diagnoses in Massachusetts
match (p:Patient) -[:ATTENDED_AN]->
    (e:Encounter) <-[:PROVIDED_AN]-
    (o:Organization) -[:LOCATED_IN]-> (g:GeoLoc)
match (d:Diagnoses) <-[r:HAS_DIAGNOSES]- (e:Encounter)
return p.Name as Patient,
    d.Name as Diagnoses,
    o.Name as Institution,
    g.Name as City,
    r.Date as Date
;
```

This query returned a table representing list of diagnoses constrained within Ludlow City.

```
// List patients with hypertension
match (p:Patient) -[:ATTENDED_AN]-> (e:Encounter)
match (:Diagnoses {Name:'Hypertension'}) <-[:HAS_DIAGNOSES]-
    (e) <-[:PROVIDED_AN]- (o:Organization)
return p, e, o
;
```

We asked a specific information of patients with hypertension who attended a certain institution without any time constraint.

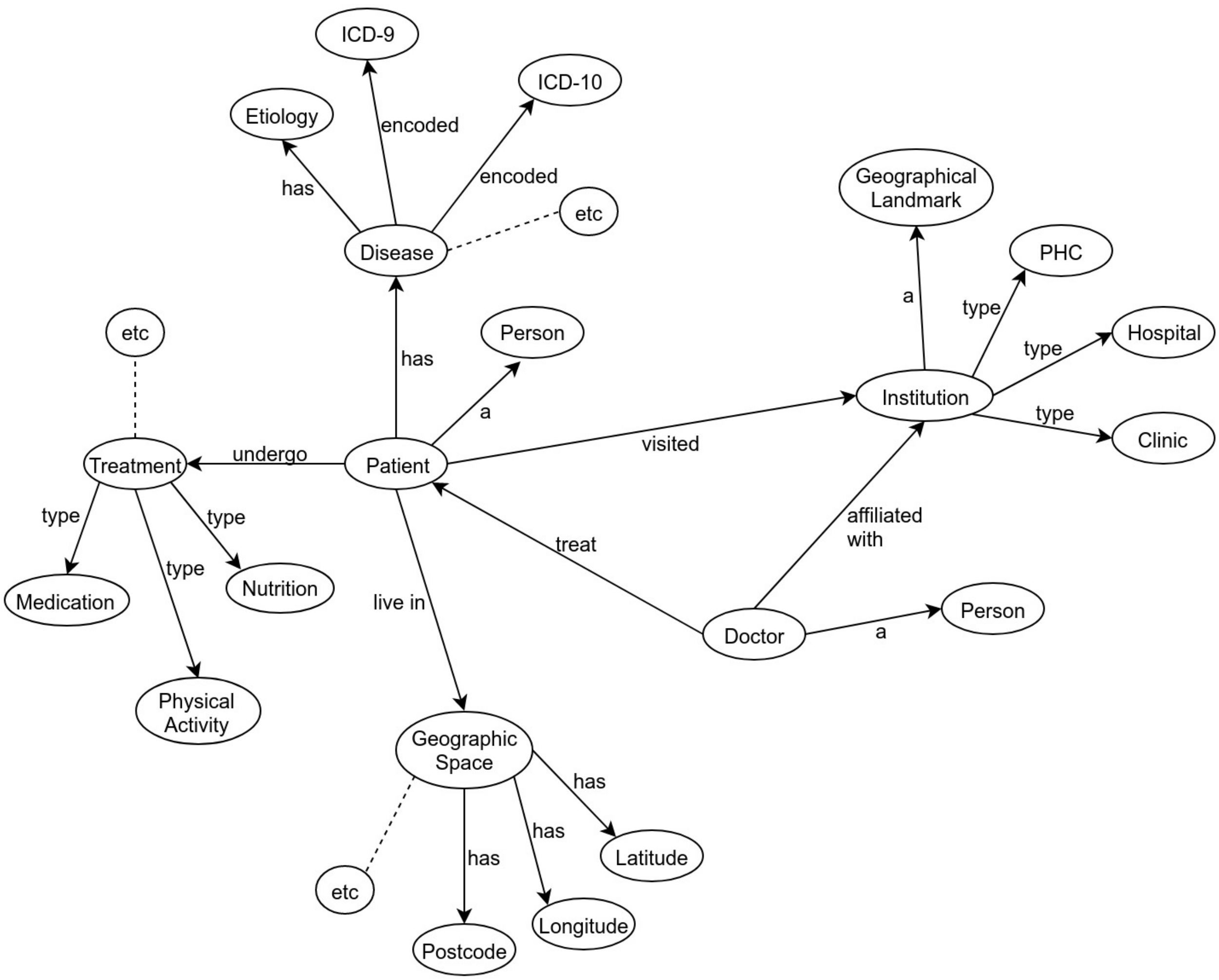

Figure 1: Schematic representation on graph database for medical records

3 Result

Constructed database is able to return answer to queries requiring with complex relationship. Our previous queries assumes data with complex relationship, where each returned a network of patient, institution and the encounter. Figure 2 and 3 depicted profile of database hits from both queries. Depicted in figure 4 is the representation of query scalability on data with different size. Fitted GLM presented as figure 5, where relationship and graph property have the most implication on data input runtime ($\rho = 0.79$, $\rho = 0.84$).

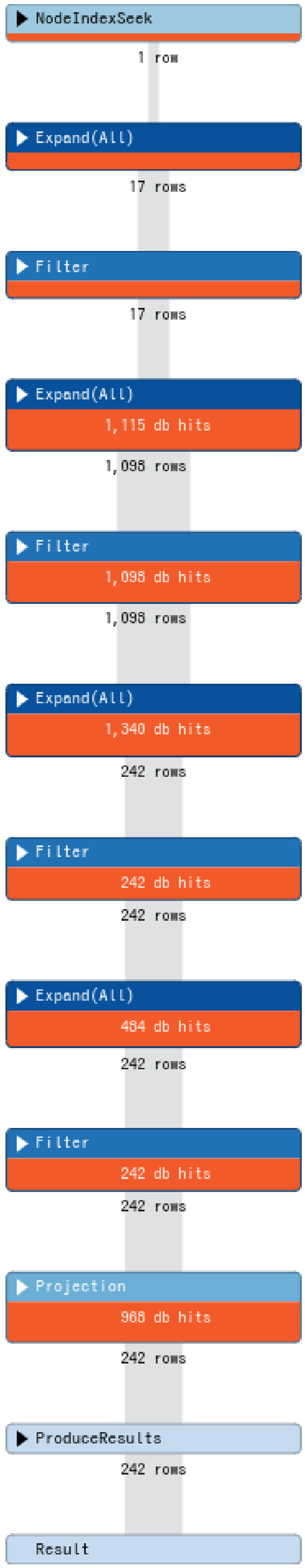

Figure 2: Database hits on the first query

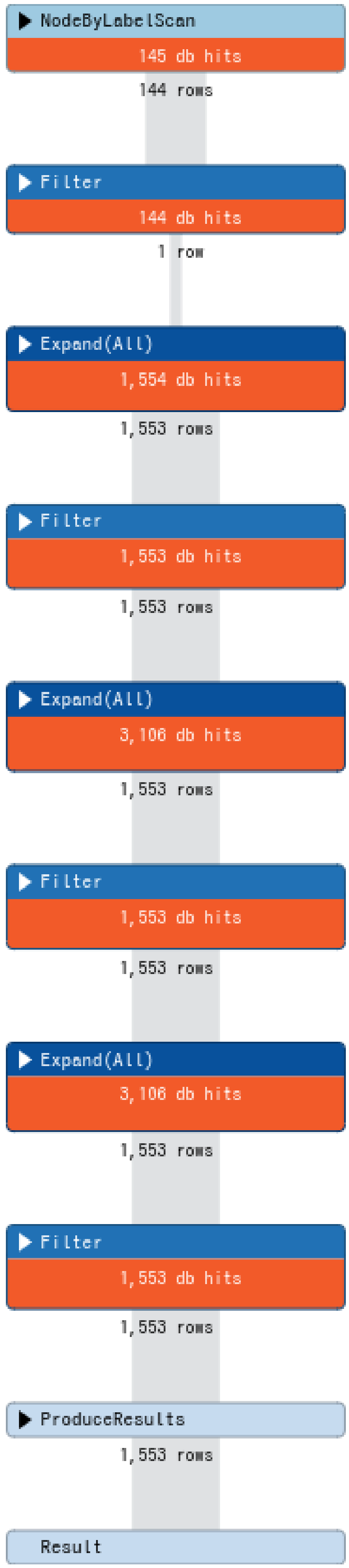

Figure 3: Database hits on the secoond query

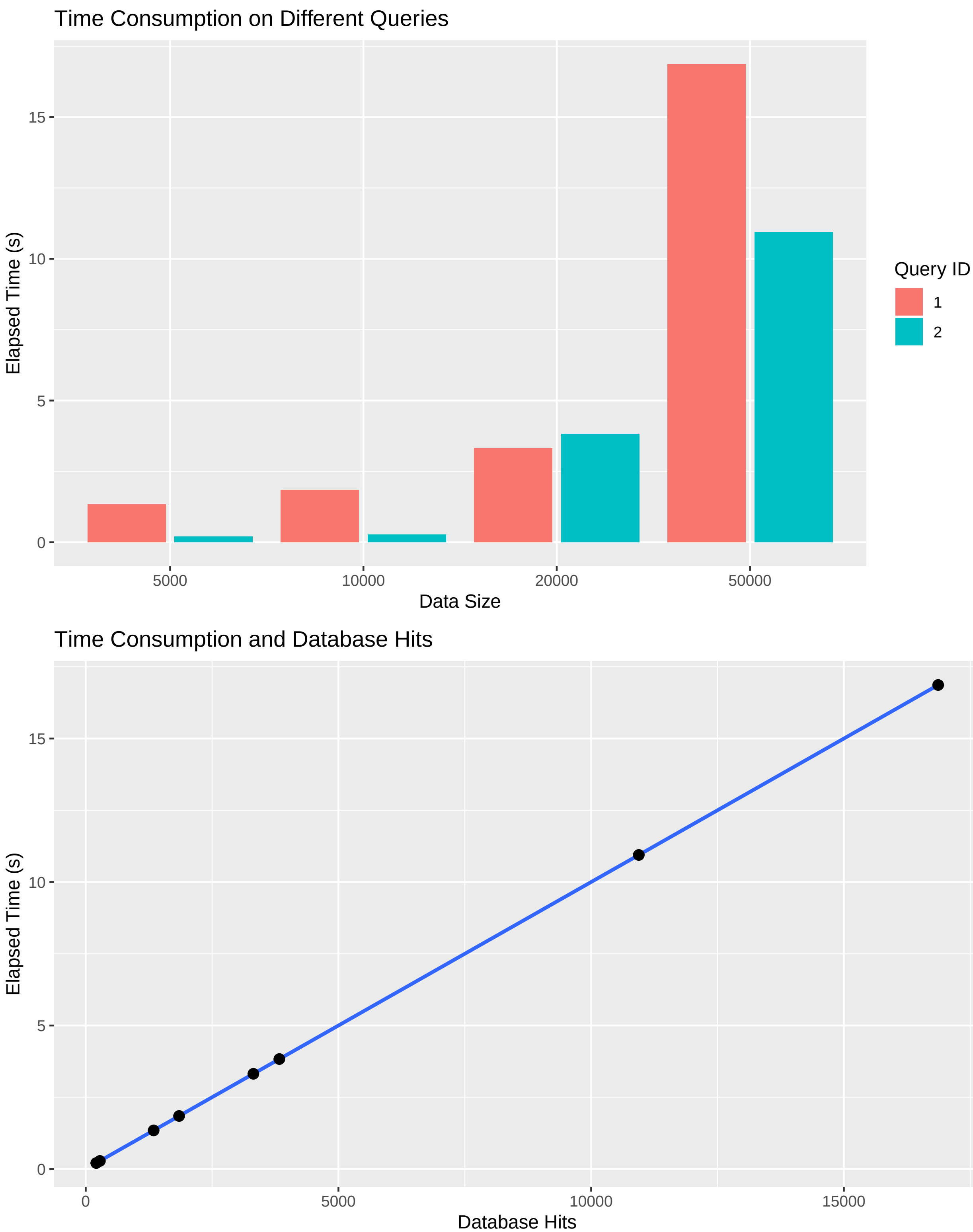

Figure 4: Queries on different dataset

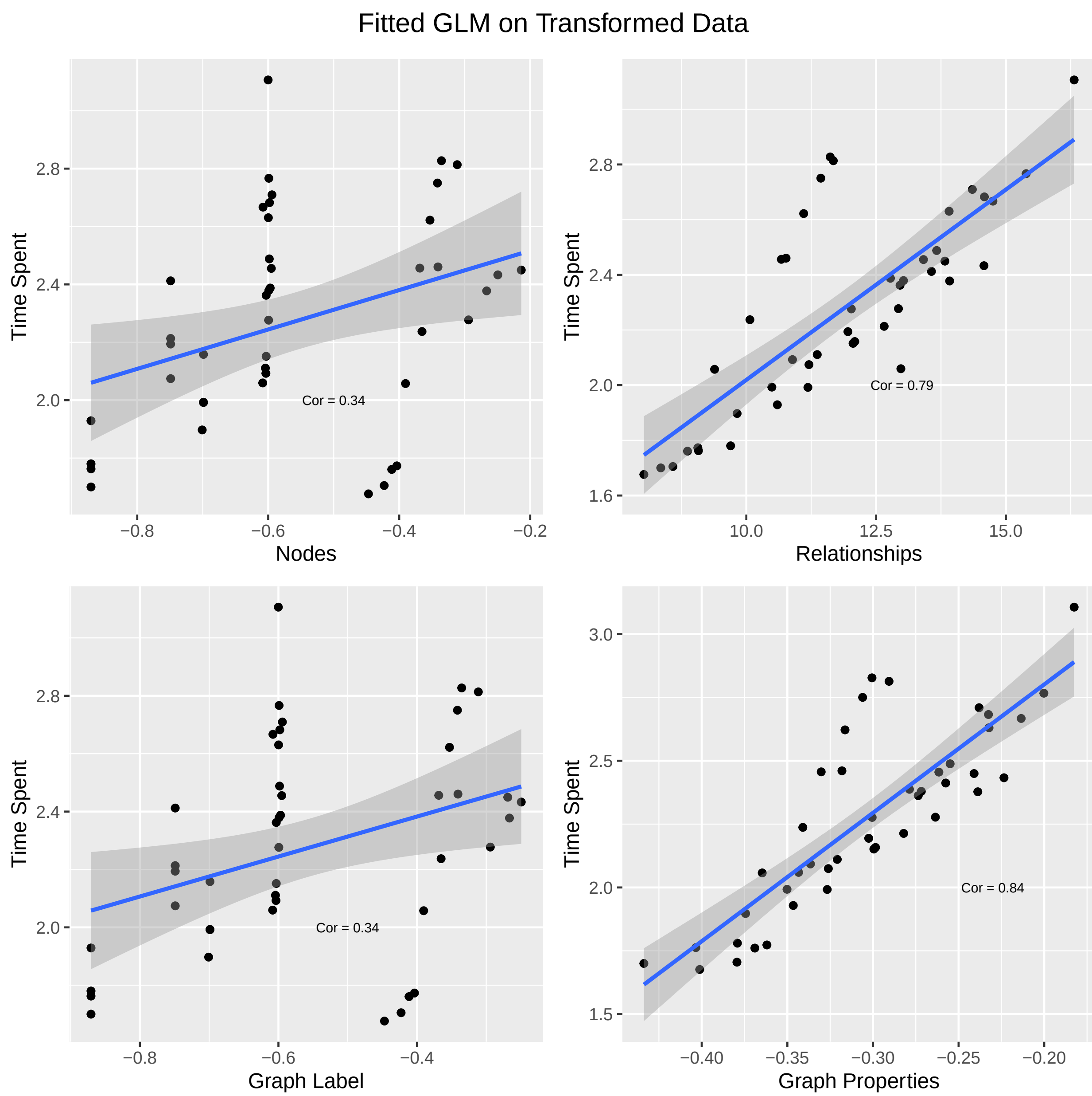

Figure 5: Data transformation and correlation

4 Discussion

Our study demonstrates graph database as a potential platform to store life science research data. Previous studies emphasizes on graph database credibility on storing interconnected data, where graph database pattern query on such data may outperform RDB (Medhi and Baruah, 2017; Fabregat *et al.*, 2018; Mathew and Kumar, 2014). However, on other cases requiring analytical query, RDB outperformed graph database, wherein their study Hölsch, Schmidt and Grossniklaus (2017) argued `Neo4j` became less performant due to less advanced disk and buffer management compared to RDB.

Our simulation demonstrated the viability on storing and querying large dataset. On exponentially increasing data size, time consumption on a particular query also increases exponentially, as demonstrated in figure 4.

However, it appears to us further optimization shall be of essence, considering query runtime increases from 20,000 to 50,000 dataset.

In preparing the database, the log captured objects causing immense burden during data input. Said objects include relationship and graph property, where previously mentioned graph database stores object explicitly instead of implying relationship. This feature actually aids graph database to answer queries for complex relationship. As such, longer time spent in creating object withing the database shall no be of issue. During data preparation, we observed longer time duration in bigger dataset. It seems `Neo4j` may perform better using smaller data, so we would suggest dividing data into smaller chunks to improve data input performance.

5 Conclusion

As a concluding remark, graph database is quite performant to integrate medical health record generated for 5,000 subjects using `synthea` program.

Reference

Berg, K.L., Seymour, T. and Goel, R. (2012) History of databases. *International Journal of Management & Information Systems (IJMIS)* [online]. 17 (1), pp. 29. Available from: https://doi.org/10.19030/ijmis.v17i1.7587doi:10.19030/ijmis.v17i1.7587.

Fabregat, A., Korninger, F., Viteri, G., Sidiropoulos, K., Marin-Garcia, P., Ping, P., Wu, G., Stein, L., D'Eustachio, P. and Hermjakob, H. (2018) Reactome graph database: Efficient access to complex pathway data Timothée Poisot (ed.). *PLOS Computational Biology* [online]. 14 (1), pp. e1005968. Available from: https://doi.org/10.1371/journal.pcbi.1005968doi:10.1371/journal.pcbi.1005968.

Hölsch, J., Schmidt, T. and Grossniklaus, M. (2017) On the performance of analytical and pattern matching graph queries in neo4j and a relational database. In: Yannis Ioannidis (ed.). *Proceedings of the workshops of the edbt/icdt 2017 joint conference* CEUR workshop proceedings [online]. 2017 Aachen: CEUR-WS.org. Available from: http://ceur-ws.org/Vol-1810/GraphQ_paper_01.pdf.

Mathew, A.B. and Kumar, S. (2014) An efficient index based query handling model for neo4j. *IJCST*. 3 (2), pp. 12–18.

Maula, A.W., Fuad, A. and Utarini, A. (2018) Ten-years trend of dengue research in indonesia and south-east asian countries: A bibliometric analysis. *Global Health Action* [online]. 11 (1), pp. 1504398. Available from: https://doi.org/10.1080/16549716.2018.1504398doi:10.1080/16549716.2018.1504398.

Medhi, S. and Baruah, H. (2017) Relational database and graph database: A comparative analysis. *Journal of Process Management. New Technologies* [online]. 5 (2), pp. 1–9. Available from: https://doi.org/10.5937/joupromanS-13553doi:10.5937/joupromanS-13553.

Oakden-Rayner, L., Beam, A.L. and Palmer, L.J. (2018) Medical journals should embrace preprints to address the reproducibility crisis. *International Journal of Epidemiology* [online]. 47 (5), pp. 1363–1365. Available from: https://doi.org/10.1093/ije/dyy105doi:10.1093/ije/dyy105.

Oussous, A., Benjelloun, F.-Z., Lahcen, A.A. and Belfkih, S. (2015) Comparison and classification of nosql databases for big data. In: *Proceedings of international conference on big data, cloud and applications*. 2015

Pashler, H. and Wagenmakers, E. (2012) Editors' introduction to the special section on replicability in psychological science. *Perspectives on Psychological Science* [online]. 7 (6), pp. 528–530. Available from: https://doi.org/10.1177/1745691612465253doi:10.1177/1745691612465253.

Peng, R. (2015) The reproducibility crisis in science: A statistical counterattack. *Significance* [online]. 12 (3), pp. 30–32. Available from: https://doi.org/10.1111/j.1740-9713.2015.00827.xdoi:10.1111/j.1740-9713.2015.00827.x.

Resnik, D.B. and Shamoo, A.E. (2016) Reproducibility and research integrity. *Accountability in Research* [online]. 24 (2), pp. 116–123. Available from: https://doi.org/10.1080/08989621.2016.1257387doi:10.1080/08989621.2016.1257387.

Sarwar, R. and Hassan, S.-U. (2015) A bibliometric assessment of scientific productivity and international collaboration of the islamic world in science and technology (s&T) areas. *Scientometrics* [online]. 105 (2), pp. 1059–1077. Available from: https://doi.org/10.1007/s11192-015-1718-zdoi:10.1007/s11192-015-1718-z.

Stark, P.B. (2018) Before reproducibility must come preproducibility. *Nature* [online]. 557 (7707), pp. 613–613. Available from: https://doi.org/10.1038/d41586-018-05256-0doi:10.1038/d41586-018-05256-0.