Analysis of Cyber Activities of Potential Business Customers Using Neo4j Graph Databases

Suglo Tohari Luri

Abstract—**Data analysis is an important aspect of business performance. With the application of artificial intelligence within databases, selecting a suitable database engine for an application design is also very crucial for business data analysis. The application of business intelligence (BI) software into some relational databases such as Neo4j has proved highly effective in terms of customer data analysis. Yet what remains of great concern is the fact that not all business organizations have the neo4j business intelligence software to implement for customer data analysis. Further, those with the BI software lack personnel with the requisite expertise to use it effectively with the neo4j database. The purpose of this research is to demonstrate how Neo4j program code alone can be use to analyze e-commerce website customer visits. As neo4j database engine is optimized for handling data relations with capabilities of building high performance, scalable systems for connected nodes, it will ensure that business owners who advertised their products at websites using neo4j as a database are able to determine the number of visitors so as to know which products are regularly visited for the necessary decision making. It will also help in knowing the best customer segments in relation to specific goods so as to place more emphasis on their advertisement on the said websites.**

Index Terms—**Data, engine, customer, intelligence.**

I. Introduction

If a relational database knowledge is the experience acquired by a database system developer in his database programming career, the developer may ask several questions on why non-relational database systems still exist. This is because the solutions offered by a relational database in solving database programming problems are very clear and seems to have no alternative. Yet relational databases have several substantial disadvantages as compared to others [1] when applied to real life situations. Some of these limitations includes;

- **Limited Volume** − Relational database stores are not optimized to be able to handle very large volumes of data
- **Velocity** – During read/write operations, the performance of a relational database suffers seriously when very huge amount of data is required to be processed in a read/write operations.
- **Lack of relationships** – In relational databases. The data stores cannot describe multiple relationships apart from standard one-to-one, one-to-many, and many-to-many relationships.
- **Variety** − Relational databases have no flexibility when dealing with data types that cannot be described using database schema. They are also not effective in handling big binary and semi-structured data such as Java Script Object Notation (JSON) and XML.
- **Scalability** − Horizontal scaling is also not efficient in the use of relational databases or data stores.

To be able to overcome these problems, several relational databases were created but most of them lacked clear relationships and references and thus, making it difficult to query highly connected data. This therefore, calls for the use of a database that makes use of connected data in the form of a tree [2]. In this case as the data grows, the system is able to scale and undergo fault tolerance to accommodate the expansion of the network.

Neo4j is a NoSQL open-source, graph database that provides transactional [3] backend applications which are ACID (Atomicity, Consistency, Isolation and Durability)-compliant [4]. This implies that data is stored exactly the way it is entered, system transaction is always robust and the database applies pointers for the navigation and traversing of the graph.

Graph Databases take business and stakeholder needs and convert them into a structure that is use to sort the data. When this is done, the database performance becomes consistent all the time as the volume of data and data point connections increases. Neo4j is very flexible [5] as it is possible to keep adding nodes to the existing graphs without disturbing its current functionality. It also allows the labelling of data in order to create grouping and establish trends.

II. Related Literature

Many individuals have used Relational Databases as data storage platforms for their database systems. Relational Database Management Systems has been the major database engine for handling and processing many transactions, reporting and operation systems and online applications [6]. However, in order to fit relational database tables and columns with data, normalization of data structures to provide minimization of data duplications across data tables is required. In most cases, these normalizations often leads to different structures as compared to what is needed. Furthermore, the structure of a relational database management system (RDBMS) has limitations on the execution and analysis of complex data structures and their interconnected nodes as all necessary and valuable information associated with connections of entities cannot be obtained.

To solve this problems, graph databases, which is a form of an object relational mapping [7] (ORM) is required to

Manuscript received October 9, 2021; revised June 12, 2022.
The authors are with the Dept. of Computer Engineering, Thakur College of Engineering and Technology, University of Mumbai, India (e-mail: tsluri@outlook.com).

doi: 10.18178/ijmlc.2022.12.6.1121

provide data mapping to all models of objects. These databases can be implemented with the help Neo4j which is a relational database system developed using graph theory techniques.

III. Graph Theory

A graph consists of a pair of sets $Z = (E, F)$ where;

- E is made of vertices and
- F is made of elements that are subsets of E, called edges.

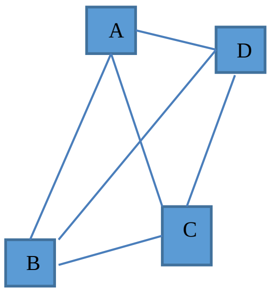

Fig. 1. Simple graph.

Base on this definition, it is significant to draw the conclusion that, in Fig. 1 above, vertices are the boxes and edges correspond to the lines of the graph. The combination of these boxes (vertices) and lines therefore represents the graph (E, F), where;

$E = \{A, B, C, D\}$ and

$F = \{(A, B), (A, C), (A, D), (B, C), (B, D), (C, D)\}$

Graphs can be in the form of single directed or multi-directed [8] graphs base on their applications but in this project, limitation will be made to only multi-directed graphs as several interconnected data points are applied.

IV. Graph Database Modelling

This is the process of expressing a domain as a graph which is interconnected with nodes and relationships [9]. They are very important components for building graphs since both nodes and relationships can have properties which provide more information between data points. Graph database modeling [10] can help find essential information between nodes. This capabilities makes them a good choice for storing and analyzing large set of data relationships in platforms. An example is shown in the Fig. 2 below.

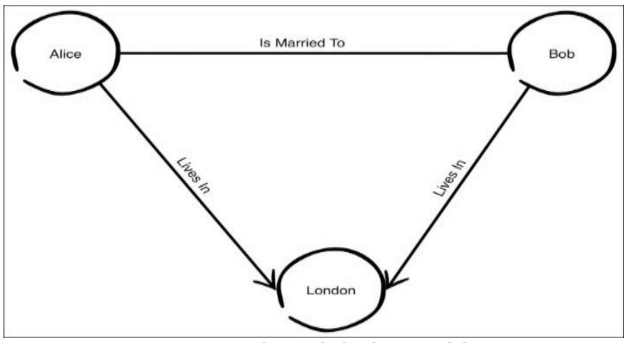

Fig. 2. Sample graph database model.

In Fig. 2, both Nodes and Relationships should have "names" just as concepts and their relationships that can be visualized. They are directed by the arrow heads. Relationships and nodes can also be associated with Properties as further illustrated by the diagram.

In graph databases, the important components are the names and the structures which formed the nodes and the relationships. Properties give supplemental solution to the structure by adding the desired content. They can also be called labels.

V. Neo4j As A Graph Database

Graph databases are based on graph theory [11]. As mentioned earlier, they are structures made of vertices and edges and all Edges may consist of numeric values known as weights.

The structure of a graph database enables database developers to design and model any scenario characterized by relationships. For instance, they can provide ground for the modelling of a social network [12] with nodes representing users and relations interconnected between them.

Neo4j has its own concept with respect to graph theory. The label property of graph model in Neo4j has the following important component:

- **Nodes**: They are the data elements which are interconnected through data relationships. A node can have one or more labels and properties and they are equivalent to vertices mentioned in graph theory.
- **Relationships:** They interconnect nodes that can have several relations. They can also have one or more properties. They are equivalent to edges in graph theory.
- **Labels:** Labels are used for grouping nodes. Each node is capable of being assigned to many labels which are applied to quickly find nodes or data points in a graph.
- **Properties**. They are nodes and relationships attributes. Since Neo4j graph database allows key-value pairs storage format, it implies that properties can take any value and they can be strings or numbers.

An example of a simple graph data model in Neo4j showing relationship between two people is shown below.

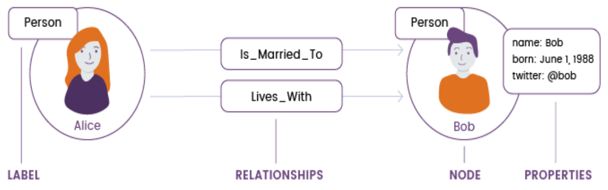

Fig. 3. Neo4j graph data model showing relationship between individuals.

As seen in Fig. 3 above, there are two nodes. These are Bob and Alice connected by relationships shown in the lines. Both nodes have the same labels called Person. In Neo4j, all node and relationship can possess properties.

VI. Neo4j Graph Database in E-Commerce

By applying graph database technology to Neo4j, it is possible to build an e-commerce database system. This is a very essential feature for many online businesses, namely, online stores and related marketplaces.

The e-commerce management system will assist in the analysis of customer behavior in relations to the product posted in the web site and therefore help businessmen to know which products to offer for sale through the online portal.

With the help of this database, businesses are capable of offering relevant goods to customers through advertisements. This improves conversations with customers and contribute to overall customer satisfaction and convenience [13] as individuals continue to have access to relevant goods and products. To build this database, it is required to install and configure Neo4j software engine.

A. Interactive Model Diagram (UCD)

Use Case Diagrams are behavioral diagram [14] created during information systems design and development. They are designed to be visual representations base on the analysis of use cases. They show and highlights in graphical representations [15], overviews of system under design functionalities in terms of their Actors, Goals and Dependencies or Association between them. As indicated in Fig. 4, they have functions and their main purpose are to indicate system functionalities.

- Actors

Actors are often external to the subjects and they interact by exchanging signals and data that is necessary for the proper functioning of the entire system. Actors in this project are Humans. They also interact with the system by initiating Use Cases, providing them input, and/or receiving output from it.

Examples of actors for this project include;

- Potential Customers of the goods in the database advertised.
- Business owner

- Use Cases

They are lists of actions that defines the nature of interaction between Actors and the system to meet the e-commerce objectives. They can be external systems as well as humans.

- Associations

Any association between an Actor and a Use Case is significant to show that the Actor and the Use Case interact and communicate with each other. An Association therefore, is the relations between a Use Case and an Actor. This means that an Actor can make use of certain functionality of the entire system. If a business Use Case is made of many Actors, it does not often show in the diagram if the Actor is capable of conducting the Use Case alone or the Actors unanimously conducting the Use Case. Association only indicate that an Actor and Use Case interacts.

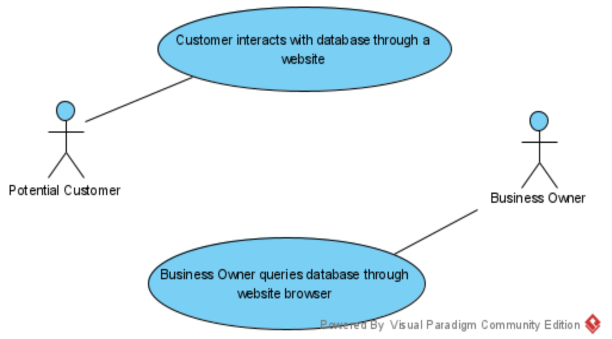
Customer interacts with database through a website
Potential Customer
Business Owner
Business Owner queries database through website browser
Powered By Visual Paradigm Community Edition

Fig. 4. Interactive model (Use case diagram).

B. The E-Commerce Data Modelling

The Neo4j database has proved useful for building an e-commerce system. Firstly, the business purpose and how it will operates with the system is determined. Because the system to be built will be used to monitor customer, marketers must monitor the behavior of the customers interested in their products to enable them launch efficient marketing campaigns [16] through their websites. This is necessary to create competitive advantages.

This system is expected to possess the following entities containing attributes in parentheses:

- Category (title)
- **Product** (title, description, price, availability)
- **Promotional Offer** (type, content)

Each of these entities in the graph database must have nodes with their respective labels. In the modelling phase, all entities interconnect through relationships. One-to-many connections [17], commonly used in relational databases is applied here. Some relationships in the model will also have the property:

- **Product** *is_in* **Category**

In Neo4j, it is not necessary to model relationships in a bidirectional fashion. Graph databases allow the direction of edges in both directions. It is also possible to modify, delete or add new relationships in a graph database without worrying about foreign keys or links as is done in NoSQL and other relational databases.

C. The Neo4j Database Software Life Cycle

In the Neo4j database design, the iterative life cycle approach to software development was adopted during the application development process. The iterative model which improved the database software quality at each incremental phase stage of the software life cycle [18] is shown in Fig. 5.

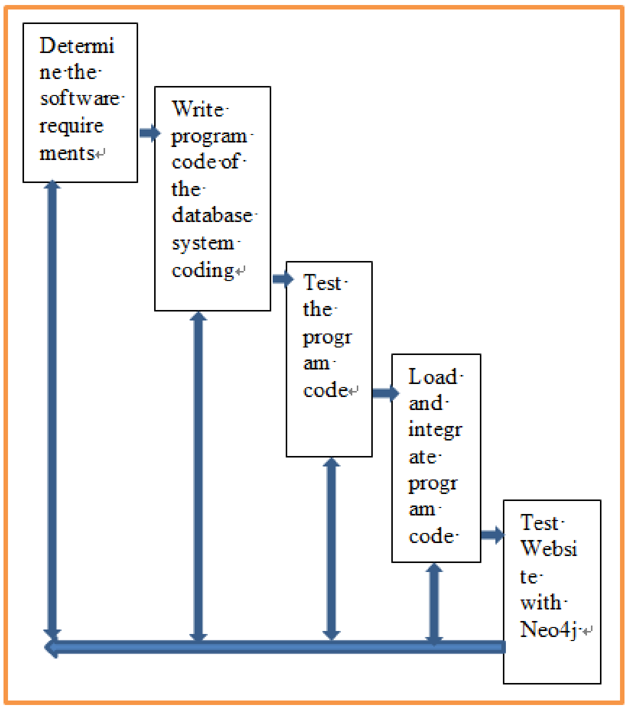
Determine the software requirements
Write program code of the database system coding
Test the program code
Load and integrate program code
Test Website with Neo4j

Fig. 5.The iterative approach to the database software design.

As can be seen in Fig. 5, the development of the database was grouped into 5 phases. After the determination of the software requirements, the database program was written and the code tested using neo4j community edition.

D. The Database Programming

Based on the model created in Fig. 5, the database was programmed by using Cypher query language. Cypher is a programming language that is declarative [19] in nature. It has characteristics of SQL with very distinctive semantics and allows the writing of very flexible and easy queries. Its syntax makes emphasis on the relationship among interconnected entities.

Cypher has recently become a project that is open sourced and is upgraded and maintained by several contributors forming a community.

```
CREATE (SSD: Category {title: 'SSD'}),
       (Labtops: Category {title: 'Labtops'}),
       (Cables: Category {title: 'Cables'})
// Solid State Drives
CREATE (SAS_3: Product {title: 'Serial Attached SCSI', price: 664.00, availability: true})
CREATE (SATA_3: Product {title: 'Serial ATA variant', price: 784.00, availability: true})
CREATE (PCIe_3_Times_4: Product {title: 'PCI Express', price: 220.0, availability: false})
CREATE (M_2: Product {title: 'SATA 3.0 Logical device interface', price: 920.20, availability: true})
CREATE (U_2: Product {title: 'storage device to support PCIe', price: 420.87, availability: true})
MERGE (SAS_3)-[:IS_IN]->(SSD)
MERGE (SATA_3)-[:IS_IN]->(SSD)
MERGE (PCIe_3_Times_4)-[:IS_IN]->(SSD)
MERGE (M_2)-[:IS_IN]->(SSD)
MERGE (U_2)-[:IS_IN]->(SSD)
// Laptops
CREATE (Lenovo_Ideapad: Product {title: 'Lenovo Ideapad Flex 5', price: 970.00, availability: false})
CREATE (HP_Omen: Product {title: 'HP Omen 15', price: 771.30, availability: true})
CREATE (Dell_Inspiron: Product {title: 'Dell Inspiron XPS 15', price: 1567.50, availability: true})
CREATE (Asus_Rog: Product {title: 'Asus Rog zephyrus G14', price: 1287.00, availability: true})
MERGE (Lenovo_Ideapad)-[:IS_IN]->(Labtops)
MERGE (HP_Omen)-[:IS_IN]->(Labtops)
MERGE (Dell_Inspiron)-[:IS_IN]->(Labtops)
MERGE (Asus_Rog)-[:IS_IN]->(Labtops)
// Cables
CREATE (Coaxial: Product {title: 'Coaxial Cable', price: 1120.00, availability: true})
CREATE (Fiber_Optic: Product {title: 'Fiber Optic Cable', price: 1612.35, availability: true})
CREATE (Twisted_Pair: Product {title: 'Twisted Pair Cable', price: 633.00, availability: true})
CREATE (Untwisted_Pair: Product {title: 'Untwisted Pair Cable', price: 456.00, availability: true})
MERGE (Coaxial)-[:BELONGS_TO]->(Cables)
MERGE (Fiber_Optic)-[:BELONGS_TO]->(Cables)
MERGE (Twisted_Pair)-[:BELONGS_TO]->(Cables)
MERGE (Untwisted_Pair)-[:BELONGS_TO]->(Cables)
MERGE (Cables)-[:INTERCONNECT]->(Labtops)
MERGE (SSD)-[:IS_USED_IN]->(Labtops);
```

From the program codes, the database contains all necessary entities and relationships. As can be seen above, Cypher is so declarative that it is possible to guess exactly what every piece of code does.

The photos of the various goods (Laptops, Solid State Drives (SSD) and Cables) to be advertised in the database were stored in Amazon web services Simple Storage Service (AWS S3) in the form of binary information. .

E. Enabling S3 Bucket Website Hosting

The Amazon S3 console was opened after sign-in and in the buckets list, the S3 bucket for the static website (https://www.ettraining.com) hosting was selected. The bucket properties was then chosen and under the Static website hosting, the bucket was made to host the website by selecting **Enable** among the **Edit** icon.

Finally, within the index documents, the file name the index name https://www.ettrading.com was entered and **save changes** was clicked. This enabled the S3 bucket to host the website https://www.ettrading.com.

F. Setting Permissions for Website Access

To ensure that the S3 bucket is accessible by the public through the internet, Amazon S3 console was opened and the static website configured bucket selected. Then, ***Permissions*** in the ***Block bucket public access setting*** was **edited** by clearing the ***Block all public access*** icon. All changes were then saved.

In this experiment, an HTML page offered a browser-based application for viewing photos in an S3 bucket album. The photos are accessible through the web browser and the diagram showing various components is below in Fig. 6;

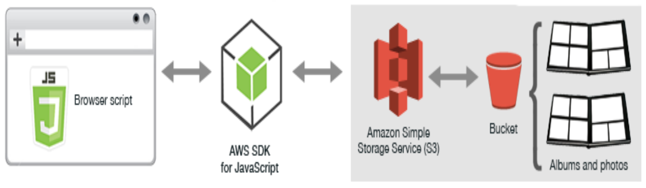

Fig. 6. Web-browser and S3 bucket interaction.

G. Uploading the Index Document

Since website hosting is enabled for the S3 bucket, a configured index document was also uploaded to the S3 bucket. This is the web-page that the S3 bucket returns when a request is made to the website root. For example,

If a website index.html is configured as https://www.ettrading.com, it will be embedded and referenced with amazon S3 bucket as;

http://ettrading-bucket.s3-website.Region.amazonaws.com/
or
http://ettrading-bucket.s3-website.Region.amazonaws.com.

The following URL therefore, requests the photo called parts.jpg stored at the root level in an S3 bucket.
http://ettrading-bucket.s3-website.us-west 2.amazonaws.com/parts.jpg

The Amazon S3 bucket that hosted the photos was referenced to the Neo4j graph database through cypher query language code. This relationship made the database information assessment possible during the website browsing.

H. Referencing the AWS S3 Bucket

```
// Cables

MATCH (a: Coaxial)
SET a.image_url =
"http://ettrading-bucket.s3-website.us-west-2.amazonaws.co
m/Coax.jpg"
RETURN a

MATCH (b: Fiber_Optic)
SET b.image_url =
"http://ettrading-bucket.s3-website.us-west-2.amazonaws.co
m/Fiber.jpg"
RETURN b

MATCH (c: Twisted_Pair)
SET c.image_url =
"http://ettrading-bucket.s3-website.us-west-2.amazonaws.co
m/Twisted.jpg"
RETURN c

MATCH (d: Untwisted)
SET d.image_url =
"http://ettrading-bucket.s3-website.us-west-2.amazonaws.co
m/Untwisted.jpg"
RETURN d

// Laptops

MATCH (e: Lenovo_Ideapad)
SET e.image_url = "http://ettrading
-bucket.s3-website.us-west-2.amazonaws.com/Lenovo.jpg"
RETURN e

MATCH (f: HP_Omen)
SET f.image_url =
"http://ettrading-bucket.s3-website.us-west-2.amazonaws.co
m/Hp_Omen.jpg"
RETURN f

MATCH (g: Dell_Inspiron)
SET f.image_url =
"http://ettrading-bucket.s3-website.us-west-2.amazonaws.co
m/Dell_Inspiron.jpg"
RETURN g

MATCH (h: Asus_Rog)
SET h.image_url =
"http://ettrading-bucket.s3-website.us-west-2.amazonaws.co
m/ Asus_Rog.jpg"
RETURN h

// Solid State Drives

MATCH (i: SAS_3)
SET i.image_url =
"http://ettrading-bucket.s3-website.us-west-2.amazonaws.co
m/SAS_3.jpg"
RETURN i

MATCH (j: SATA_3)
SET j.image_url =
"http://ettrading-bucket.s3-website.us-west-2.amazonaws.co
m/SATA_3.jpg"
RETURN j

MATCH (k: PCIe_3_Times_4)
SET k.image_url =
“http://ettrading-bucket.s3-website.us-west-2.amazonaws.co
m/ _PCIe_3_Times 4.jpg”_
RETURN k

MATCH (l: M_2)
SET l.image_url =
"http://ettraining-bucket.s3-website.us-west-2.amazonaws.c
om/ M_2.jpg"
RETURN l

MATCH (m: U_2)
SET m.image_url =
"http://ettraining-bucket.s3-website.us-west-2.amazonaws.c
om/ U_2.jpg"
RETURN m
```

VII. Testing and Evaluation

For this sample system, Neo4j was first of all downloaded and installed in a PC. It was then ran and the web browser launched, the interactive console which is always the Neo4j browser is installed by default during the installation of the Neo4j server. In order to get access to the Neo4j browser, a visit was made to the local website http://localhost:7474/browser/. This opened up the programming platform after launching the Neo4j start icon.

To visualize the graph, the above code was executed using the **MATCH (n)** and **RETURN n** query which returned all data points and nodes in the graphs. The output of the execution showed the graph in Fig. 7 below.

Code: **MATCH (n) RETURN n;**

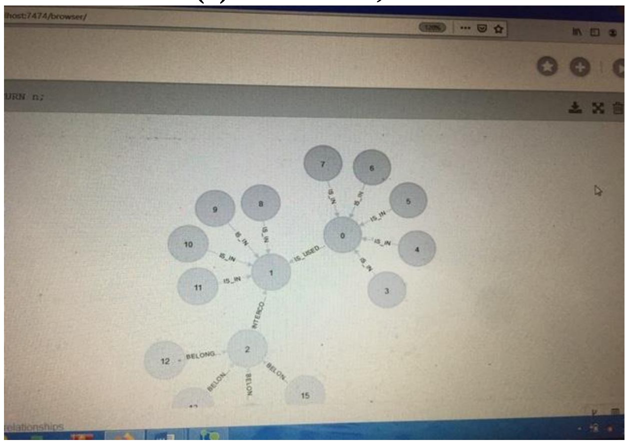

Fig. 7. Neo4j graphs.

These graphs are scalable [20] and can be applied with larger datasets. As can be seen, Neo4j can also help data modelling to create very valuable insights into the said data. The code was executed and allowed to ran in the neo4j database engine

In Neo4j database programming, all edges that go between views and pages are known as object edges.

After test executing the above code from the local host web browser using neo4j community edition, the program was connected to sales website of Electronics marketing. This was a fake website created to test the performance of the Neo4j graph database in relations to customer activities in their website. To check for the number of browsers who visited the site, all view nodes that led to the home page were located. It is assumed that anybody who visited the website that the program is connected, is interested in buying one or more of the products advertised there. By determining the number of the website visits through cypher querying at routine intervals, customer data from number of visits were recorded for analysis. Through this analysis, business owners may place more emphasis on the frequently visited product for decision-making.

After 12 visits to the website, execution of the query, returned a table that informed the number of views of the home page.

```
MATCH (Visitors:User)-[r:VIEWS]->(home:Page {page: "https://www.ettrading.com/"})
RETURN home.page AS url, count(r) AS visits;
```

TABLE I: NUMBER OF VISITORS INCLUDING "BOUNCERS"

url	visits
www.ettrading.com	12

Table I displayed the total number of visits including people who arrived accidentally at the website. It is the aim of the system developer based on the research objectives to get rid of the number of the website accidental visitors. As they visited the homepage and left the site shortly, it implies that they do not want to operate any business there. They are known as "Bouncers". The code used is as follows;

```
MATCH (Visitors:User)-[r:VIEWS]->(home:Page {page: "https//:www.ettrading.com"})
WHERE NOT (home)-[:NEXT]->()
RETURN home.page AS url, count(r) AS visits;
```

TABLE II: NUMBER OF VISITORS THAT ARE "BOUNCERS"

url	visits
www.ettrading.com	5

Table II displayed the number of the website visitors that are "bouncers". This means that of the twelve visits, five (5) persons did not have any business with the website whilst seven (7) persons were true visitors and interested in the products advertised.

To conduct further finding base on the objectives of the research, the database program was made to function with the www.ettrading.com website for one and half month. Visits were then made to the website to view the solid state storage drives, the laptop computers and their cables on daily bases by a selected group of people (Total Number = 131). The Neo4j database was queried at one week intervals for a period of six (6) weeks. The results were then tabulated and a bar chart constructed to view the pattern of visits.

VIII. RESULTS AND FINDINGS

These codes below indicate the number of visits and bouncers per week for a total of six (6) weeks. (Table III)

TABLE III: WEEKLY WEBSITE VISITS

	Total No. of visitors (Cumulative value)	Total Bouncers (Cumulative Value)	Net No. of visitors per week	Net bouncers per week	Net No. of true website visits per week
Week 1	24	5	24	5	19
Week 2	54	7	30	2	28
Week 3	73	10	19	3	16
Week 4	98	19	25	9	16
Week 5	115	21	17	2	15
Week 6	131	26	16	5	11

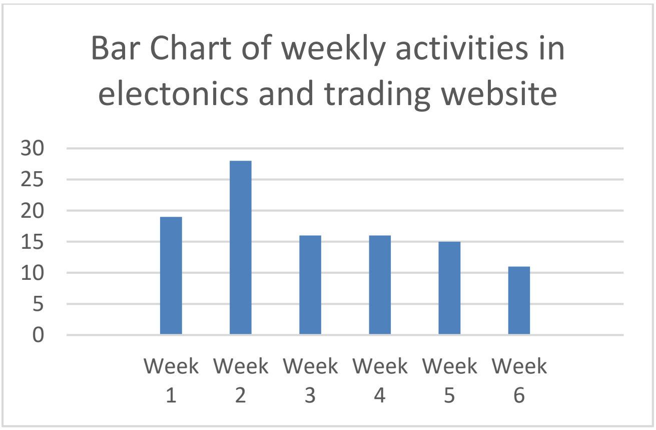

Fig. 8. Bar chart

The chart in Fig. 8 indicates the net number of true website visits per week. Visitors who went to the web pages accidentally are called "bouncers". They are therefore not considered even though they were indicated by the program.

IX. CONCLUSIONS AND FUTURE RESEARCH

This experiment with Neo4j is capable of answering questions about user activities as they browse through the website in which the database is implemented. It tried to explore people behavior in a way that did not put limitations on analysis of user activities with respect to certain products advertised in the internet. For example, the owner of the online shopping can apply the visualized data to find out why there was low patronage of the website from week 3 to week 6 and based on that, he may institute measures to conduct micro-marketing by initiating email messages to all visitors to improve customer relations and product sales. The results obtained from this research work has proved very promising. It showcased how Neo4j database programming can be used to perform customer analysis for decision making.

Since this is a demonstration which showed the program strength and abilities. It is also highly recommended that further research be conducted on how to automate the query code within the website to assist people speed up the query process.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

ACKNOWLEDGMENT

I would like to express my profound gratitude to all the staff members of Grid Telecommunications of Ghana Grid Company Ltd and Dr. Irvin Lamarca of AMA University, Philippines. For their immersed contribution in various ways to the success of this paper. Their support and mentoring helped me to understand new things in neo4j programming and I am really thankful to them.

References

[1] P. a. D. P. A. Nayak, "Type of NOSQL databases and its comparison with relational databases," *International Journal of Applied Information Systems (IJAIS)*, vol. 5, no. 4, p. 19, 2013.
[2] A. S. a. C. W. M. K. Reiter, "Distributed construction of a fault-tolerant network from a tree," in *Proc. the 2005 24th IEEE Symposium on Reliable Distributed Systems*, 2005, p. 2.
[3] G. D. H. R. E. a. E. G. S. A. Dubey, "Weaver: A high-performance, transactional graph database based on refinable timestamps," *Proceedings of the VLDB Endowment*, vol. 9, no. 11, p. 852, 2016.
[4] J. Pokorn ý, "Graph Databases: Their power and limitations," " in *Proc. 14th Computer Information Systems and Industrial Management (CISIM)*, 2015, pp. 58 - 69.
[5] D. Fernandes and J. Bernardino, "Graph databases comparison: AllegroGraph, ArangoDB, InfiniteGraph, Neo4J, and OrientDB," " in *Proc. 7th International Conference on Data Science, Technology and Applications*, 2018, p. 379.
[6] M. S. a. J. H. Joseph and M. Hellerstein, "Architecture of a database system," *Foundations and Trends in Dtabases*, vol. 1, no. 2, p. 143, 2007.
[7] J. Pokorny, "Integration of relational and graph databases functionally," *Foundation of Computing and Decision Sciences*, p. 428, 2019.
[8] T. Shafie, "A multi-graph approach to social network services," *Journal of Social Structure*, vol. 16, pp. 1 - 3.
[9] R. Angles, *A Comparison of Current Graph Database Models*, p. 1, 2012.
[10] S. B. a. B. Kalita, "Designing graph database models from existing relational databases," *International Journal of Computer Applications*, vol. 14, no. 1, pp. 25 - 30, 2013.
[11] A. N. a. R. S. A. Ismail, "Application of graph databases and graph theory concepts for advanced analysing of BIM models based on IFC standard," " in *Proc. 24th International Workshop on Intelligent Computing in Engineering*, Nottingham, 2017.
[12] A. C. a. M. S. A. Faisal, "Role of graph databases in social networks," *ResearchGate*, p. 1, 2015.
[13] N. Mkhize, "Determinants of online customer satisfaction in relation to online shopping: A South African perspective," " in *Proc. International Business Conference*, 2020.
[14] Sparx Systems, Using UML – Behavioral Modeling Diagrams, Sparx Systems, 2007, p. 6.
[15] R. Klimek and P.Szwed, "Formal analysis of use case diagrams," *Computer Science*, vol. 11, p. 116, 2010.
[16] M. F.Sabbagh, "Marketing and campaign management via social networks and the effects of electronic advertising," *International Journal of Economics & Management Sciences*, vol. 7, no. 4, p. 1, 2018.
[17] J. C. a. R. H. Bryce Merkl Sasaki, *Graph Databases For Beginners*, Neo4j, 2018.
[18] B.-Y. Tsai *et al.*, "Iterative design and testing within the software development life cycle," *Software Quality Journal*, vol. 6, no. 4, p. 4, 1997.
[19] A. T. N. Francis *et al.*, "Cypher: An Evolving Query Language for Property Graphs," *Association for Computing Machinery (ACM)*, p. 2, 2018.
[20] J. Pokorn ý, "Graph databases: their power and limitations," *International Federation for Information Processing*, p. 62, 2015.

Suglo Tohari Luri was born in Bolgatanga, in the upper east region of Ghana in 1974. He obtained his B. Sc. in electrical/electronic engineering at Regional Maritime University, Accra, Ghana, in 2010 and an M Sc. in ICT Ghana institute of management and public administration majoring in IT and telecommunications networking.

He has been working as an IT and telecommunications supervisor for Ghana Grid Company from 2006 up to date where he supervises and manages a team of 6 technicians in the field of IT projects with very high degree of success.

Mr. Suglo Tohari Luri is currently a member of Project Management Institute (PMI), USA and Chartered institute of IT, UK