Analyzing Consumer Spending Behavior Using Graph Databases

I.S. Hristoski[**], T.M. Spaseska[*], D.S. Odžaklieska[*] and T.P. Dimovski[**]

[*] “St. Kliment Ohridski” University – Bitola, Faculty of Economics, Prilep, North Macedonia

[**] “St. Kliment Ohridski” University – Bitola, Faculty of Information and Communication Technologies, Bitola, North Macedonia

{ilija.hristoski, tatjana.spaseska, dragica.odzaklieska, tome.dimovski}@uklo.edu.mk

Abstract - In order to ensure their financial stability, both families and individuals have to effectively manage their incomes and spending on a daily basis. Since these are based on various types of financial transactions, understanding spending behavior can significantly help people in making better decisions. However, this cannot be achieved by simply analyzing raw data. In this paper, we propose a methodology for analyzing financial transactions using graph databases, based on the analysis of the relationships found among the counterparts while carrying out financial transactions. Starting out from a corresponding E-R diagram depicting financial transactions, we first infer an equivalent graph database model. In order to demonstrate the effectiveness of such an approach, we then implement a test graph database in Neo4j, and by using Cypher query language (CQL), we make noteworthy insights about people’s behavior in terms of how they perceive their finances through the process of money spending.

I. Introduction

Financial management is one of the functional areas of general management, which refers to planning and controlling of financial resources of firms or individuals. It is an indispensable part of the economic and non-economic activities that lead to making relevant decisions regarding the efficient procurement and utilization of finances in a profitable manner. Financial management is traditionally separated into two basic functions: the acquisition of funds and the investment of those funds. The first function, also known as the financing decision, involves generating funds from internal sources or from sources external to the firm, at the lowest long-run cost possible. The second function, the investment decision, is concerned with the allocation of funds over time in such a way that shareholder wealth is maximized. In this paper, we put focus on the second function made by individuals, also known as consumer spending.

Effective financial management helps in achieving specific goals, both business-oriented and individual. On a firm level, its importance can be seen through the effects of financial planning, acquisition and proper use of funds, protecting funds towards achieving business goals, appropriate allocation of funds, in making sound financial decisions, in improving the profitability and increasing the value of the firm, in ensuring the economic growth and stability, and maximizing firm’s wealth through promoting savings [1]. On an individual level, proper financial management can contribute towards someone’s financial stability, improving his/her standard of living, providing many investment opportunities, giving someone peace of mind and keeping someone financially stress free [2].

An important constituent part of financial management is consumer spending. Consumer spending or ‘personal consumption expenditure’, which can be regarded as opposed to personal saving, is another term for voluntary private household consumption, or the exchange of money for goods and services. Contemporary measures of consumer spending include all private purchases of durable goods (consumer goods that do not have to be purchased frequently and tend to last for at least three years), nondurables, and services. According to Kenton (2018), “many economists, especially those in the tradition of John Maynard Keynes, believe consumer spending is the most important short-run determinant of economic performance and is a primary component of aggregate demand” [3]. Consumer spending is a vital economic variable since the consumption of final goods (i.e., not capital goods or investment assets) is the result of economic activity, during which individuals ultimately use these goods and services to satisfy their own needs and wants. However, if consumers spend too much of their income, future economic growth could be compromised because of insufficient savings and investment. That’s why modern governments and central banks often examine consumer spending patterns when considering current and future fiscal and monetary policies. Consumer spending is a significant investment indicator, too: the more money consumers spend at a given company, the better that company tends to perform. On the contrary, if consumers provide fewer revenues for a given business or within a given industry, companies must adjust by reducing costs, wages, or innovating and introducing newer and better products and services. Companies that do this most effectively, usually earn higher profits and, if publicly traded, tend to experience better stock market performance.

Capturing consumer spending patterns is also important at a micro-scale because by analyzing those patterns, individuals and families can carry out proper planning for their retirement days, college teaching fees, buying cars, houses, etc. Yet, just looking at the bank statements and shopping bills is not enough for consumers to understand their spending patterns. The starting point in capturing consumer spending patterns is a financial transaction, which denotes a single exchange of money between a buyer and a seller. Since a systematic approach of organizing financial transactions into a corresponding database is needed as a key premise to successful capturing and analysis of consumer spending patterns, our approach is based on the utilization of graph databases in achieving this goal.

The paper is organized as follows. In Chapter 2, we provide an E-R diagram that encompasses the key entities, their attributes, and corresponding relationships needed for depicting a generic financial transaction. In addition, we briefly introduce the main rules of transforming the E-R diagram into an equivalent graph database model. Based on the implemented test graph database, in Chapter 3 we provide Cypher query language (CQL) programming codes needed to implement some basic analyses *vis-à-vis* the consumer spending behavior. The last chapter concludes.

II. Obtaining a Graph Database Model from an E-R Diagram

Graph databases belong to the family of NoSQL databases; they address one of the great macroscopic business trends of today: "leveraging complex and dynamic relationships in highly connected data to generate insight and competitive advantage" [4]. Unlike the relational databases, which store data to efficiently store facts, graph databases store both facts and the relationships among the facts, making certain types of analyses more efficient and intuitive. Emerging as a major driver of innovation during recent years, graph databases have already exhibited many advantages over relational databases, like storing large volumes of data that might have little to no structure, sharing data across multiple servers in the cloud, speeding the development, boosting the horizontal scalability, demonstrating superior performances, and supporting iterative algorithms and other data mining and machine learning algorithms.

In order to establish a graph database framework for analyzing consumer spending behavior, we revert to a corresponding E-R diagram as a starting point (Figure 1). The E-R diagram consists of five entities: CUSTOMER, SELLER, PRODUCT, SUBCATEGORY, and CATEGORY. First three of them, along with the relationships that interconnect them, compose the basis of what is called a financial transaction, i.e. a specific customer, identified by his/her primary key *customerID*, buys a specific product (or service!), identified by its primary key *productID*, from a specific seller, identified by its primary key *sellerID*. The entity type CUSTOMER refers to each particular member of a given family. The entity type SELLER refers to any person or business entity (a vendor) that makes offers or contracts in order to make a sale to an actual or potential buyer (a customer), whilst the entity type PRODUCT refers to any merchandise (tangible good) or service (intangible good) that can be bought or sold, e.g. food, clothing, books, electricity supply, water supply, gas supply, cable TV access, Internet access, education/knowledge, etc. These three entity types are mutually connected by relationships of M:N cardinalities, all three having corresponding primary keys and a number of non-key attributes that entirely describe the interaction among them.

In Figure 1, each instance of the entity type PRODUCT belongs to a single subcategory (entity type SUBCATEGORY), which is further being categorized into a single category (entity type CATEGORY). Therefore, the relationships *BELONGS_TO* and *SUBCATEGORY_OF* are both of a cardinality M:1. For instance, the product 'Bread' (an instance of the entity type PRODUCT) belongs to a single (1) subcategory 'Bread and bakery', which is just a single (1) subcategory of the category 'Food'. The latter one includes many (M) other subcategories, e.g. 'Fruits', 'Vegetables', 'Cereal and cereal products', 'Dairy products and analogs', 'Meat and meat products', 'Fish and fish products', etc. [5]. Likewise, each subcategory may include many (M) products.

Before transforming the E-R diagram into a graph database model, it is convenient to point out the process of transformation, which can be carried out through a number of steps [6-10]:

- Each entity table is represented by a label on nodes;
- Each row in an entity table (i.e. instance) becomes a particular node;
- Columns on those tables (entity table's attributes) become node properties;
- Technical primary keys should be removed, whilst keeping business ones;
- Unique constraints should be added for business primary keys;
- Indexes should be added for frequent lookup attributes;
- Each foreign key that appears in entity tables (M) due to the existence of a relationship with a 1:M cardinality, should be replaced with a relationship to the other table (1) in the corresponding graph data model, and removed afterward from the original (originating) table/entity type;
- Data with default values should be removed, there is no need to store those;
- Data in tables that are de-normalized and duplicated might have to be pulled out into separate nodes to get a cleaner model;
- Indexed column names might indicate an array property;
- Simple JOIN tables that come out from the M:N relationships in the E-R diagram become simple relationships in the graph data model;
- Attributed JOIN tables that come out from the M:N relationships in the E-R diagram become relationships with properties in the graph data model.

The resulting graph database model, which is semantically equivalent to the E-R diagram depicted in Figure 1, is given in Figure 2.

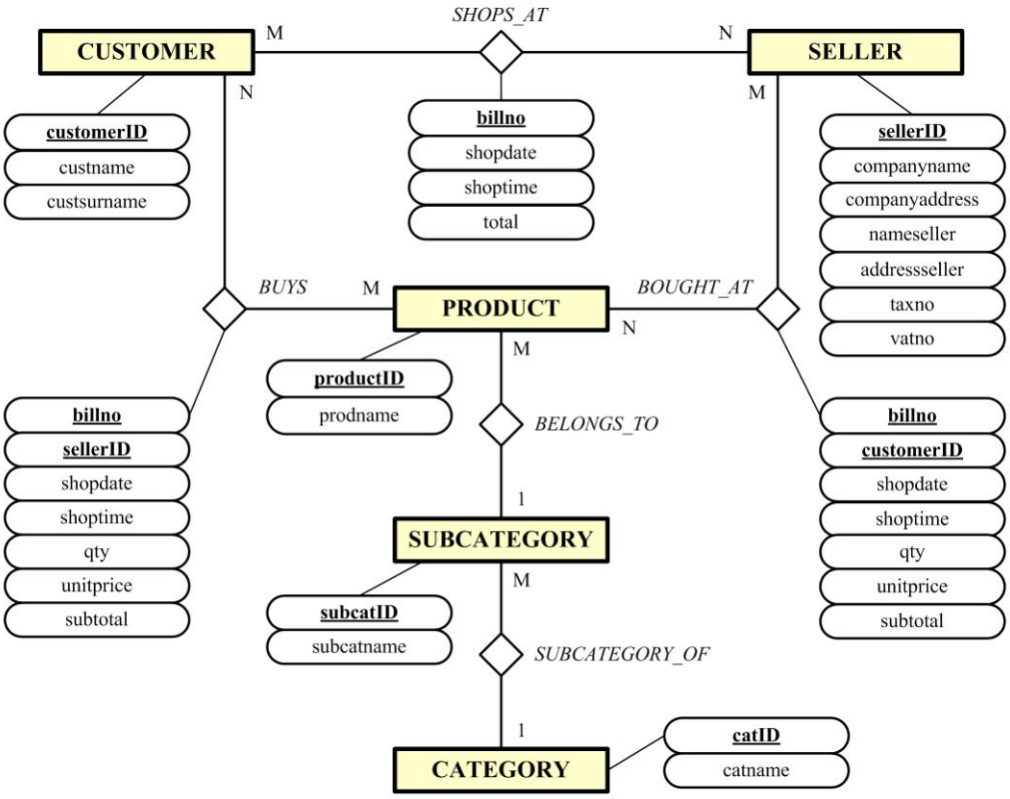


Figure 1. E-R diagram depicting the entities, their attributes, and relationships needed to model financial transactions and the consumer spending behavior

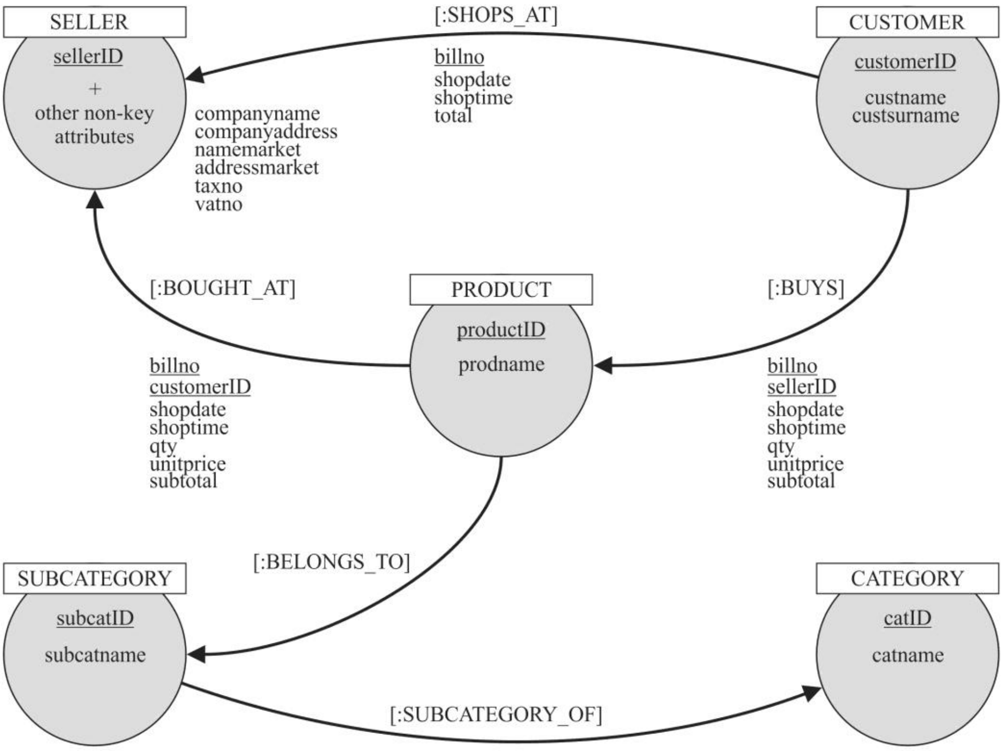


Figure 2. The equivalent graph database model

III. Some Basic Analyses Related to Consumer Spending Behavior

The graph database model, shown in Figure 2, is suitable for performing a number of analyses of the consumer spending behavior. In order to make evidence for some of them, we have set up a test graph database in Neo4j. It consists of two nodes labeled 'CUSTOMER' that correspond to two adult members of a hypothetical family. During a given period of time, they are spending their money in different ways, e.g. by shopping at local markets and supermarkets while buying food, household products, and clothing, or by conducting a preventive maintenance of their car, by paying bills for electric power supply, water supply, cable TV, Internet access, heating, etc. An excerpt from the test graph database, which corresponds to *customerID* = 1, is obtained by running the following CQL query and is being visualized in Figure 3.

```
MATCH (c:customer {customerID:1}),
(p:product), s:subcategory),
(t:category), (r:seller)
RETURN (c)-[:BUYS]->(p)-
[:BELONGS_TO]->(s)-
[:SUBCATEGORY_OF]->(t), p)-
[:BOUGHT_AT {customerID:1}]->(r),
(c)-[:SHOPS_AT {customerID:1}]->(r)
```

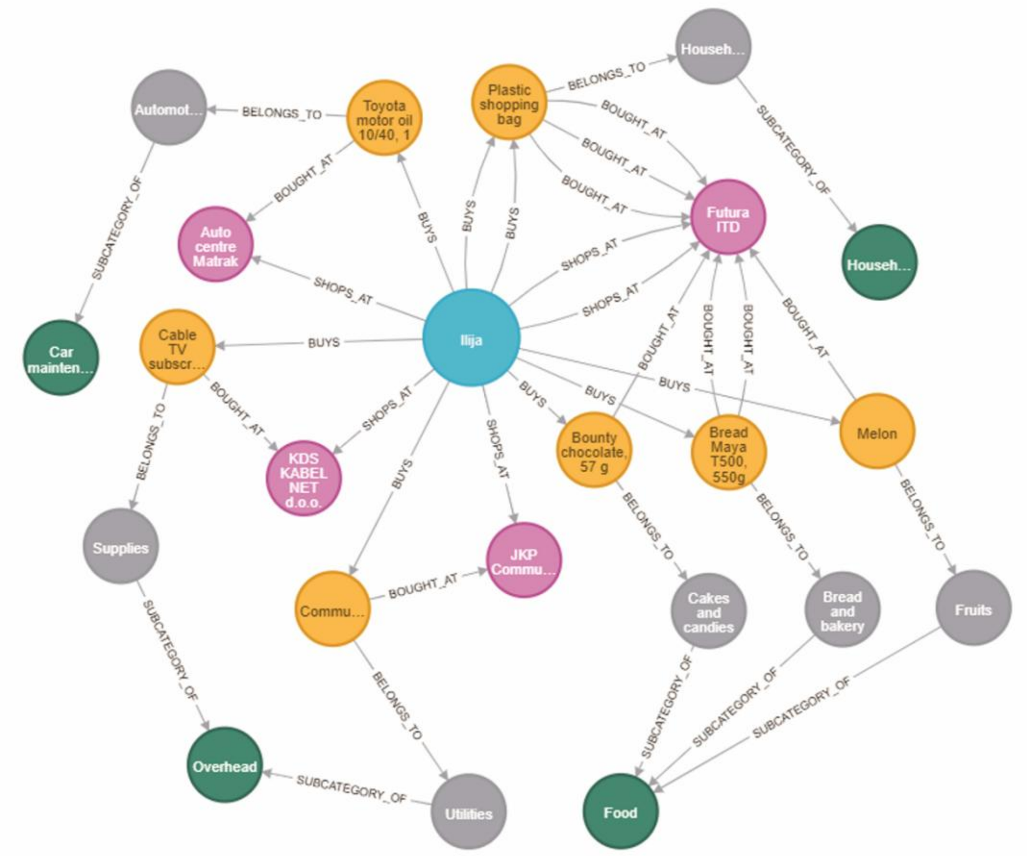


Figure 3. The outlook of the test Neo4j graph database for *CustomerID* = 1. Each particular financial transaction made by customers follows the structure and includes the constructs, of the graph database model shown in Figure 2

The so-called *sellers' vector of a particular customer* is, simply, a set of all sellers/merchants a particular customer is buying from during a given period of time. Given the test graph database, it can be obtained by running the following CQL query, which refers to *CustomerID* = 1 and the time period from August 01 to August 31, 2019:

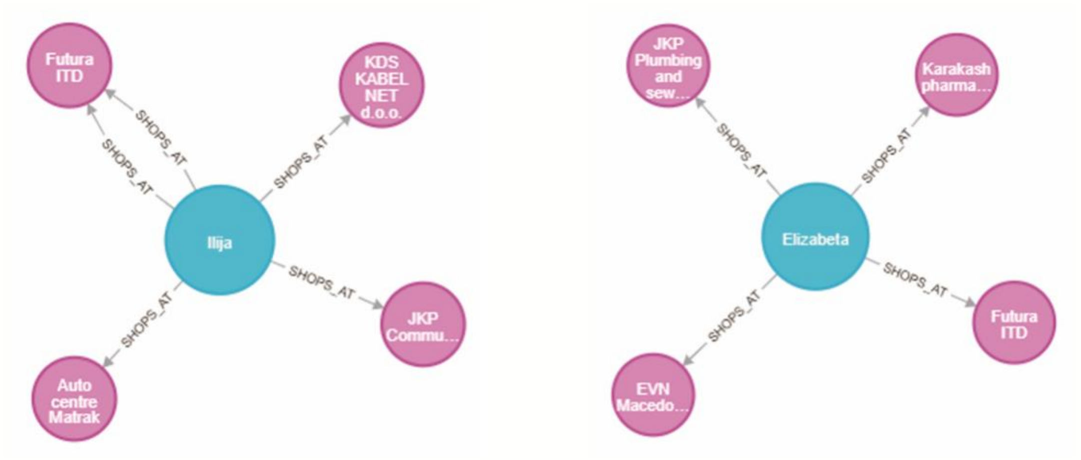
```
MATCH (n:customer {customerID:1})-
[s:SHOPS_AT]->(r:seller) WHERE
s.shopdate >= "01.08.2019" AND
s.shopdate <= "31.08.2019"
RETURN n, r
```

The seller's vector of *CustomerID* = 1 is visually depicted by Figure 4a and the one that refers to *CustomerID* = 2 by Figure 4b.

Instead visually, quantitative information can be obtained by using aggregate functions COUNT(*) and SUM(.), which count the number of shopping occasions with the sellers and sum the total amount paid, respectively (Figure 5):

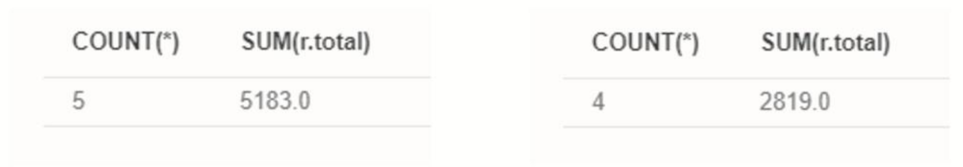
```
MATCH (n:customer {customerID:1})-
[r:SHOPS_AT]->(m:seller) WHERE
r.shopdate >= "01.08.2019" AND
r.shopdate <= "31.08.2019" RETURN
COUNT(*), SUM(r.total)
```

Other aggregate functions, like MIN(.), MAX(.), and AVG(.), which are used for computing the minimum, maximum, and average values, respectively, can be also used, as needed, in order to obtain more profound statistical insights.



(a) *CustomerID* = 1 (b) *CustomerID* = 2

Figure 4. Sellers' vectors for two members belonging to the same family, *CustomerID* = 1 and *CustomerID* = 2, during a given period of time, represented visually



COUNT(*)	SUM(r.total)
5	5183.0

(a) *CustomerID* = 1

COUNT(*)	SUM(r.total)
4	2819.0

(b) *CustomerID* = 2

Figure 5. Sellers' vector for two members belonging to the same family, *CustomerID* = 1 and *CustomerID* = 2, during a given period of time, represented through aggregated numbers

The so-called *consumer's spending pattern model* includes information about (particular/all) consumers, (particular/all) products they bought, as well as (particular/all) subcategories and categories those products belong to. For instance, the consumer's spending pattern model for the whole household for the category 'Food', represented in Figure 6, can be obtained by running the following CQL code:

```
MATCH (c:customer), (p:product),
(s:subcategory), (t:category),
(c)-[b:BUYS]->(p) WHERE t.catname =
"Food" AND  b.shoptime>="01.08.2019"
AND b.shoptime<="31.08.2019" RETURN
(c)-[:BUYS]->(p)-[:BELONGS_TO]->
(s)-[:SUBCATEGORY_OF]->(t)
```

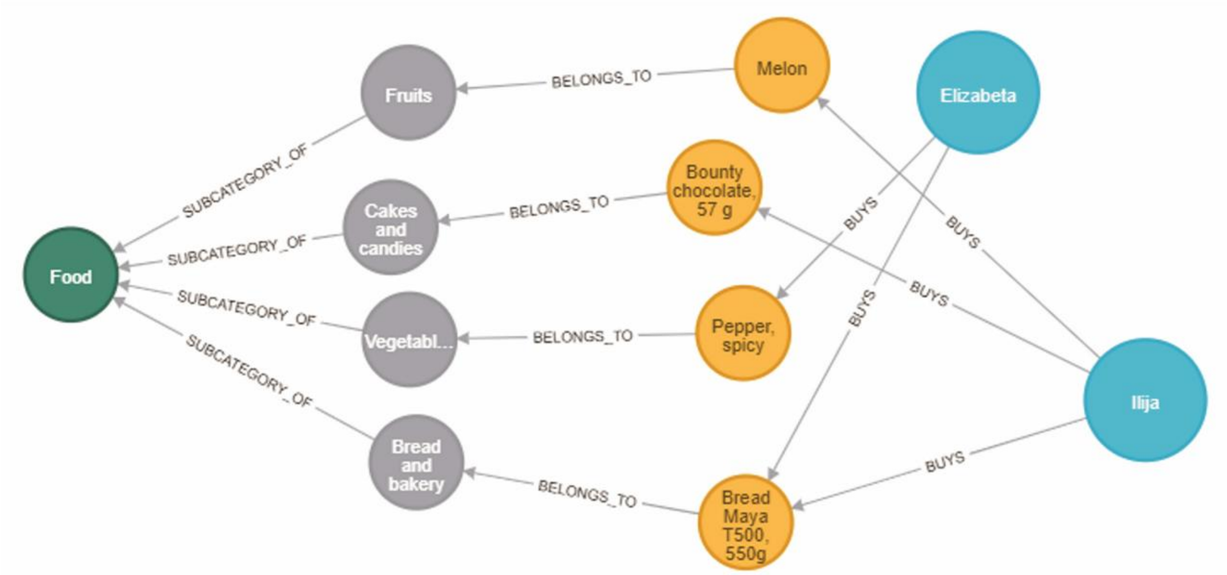
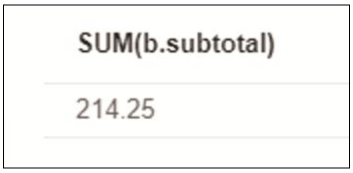


Figure 6. Consumer's spending pattern model for the whole household, for the category 'Food', during a given period of time, represented visually

The *total amount spent on a specific category* (e.g. ‘Food’) during August 2019, shown in Figure 7, can be calculated by running the following CQL code:

```
MATCH (c:customer)-[b:BUYS]
->(p:product)-[:BELONGS_TO]
->(s:subcategory)-[:SUBCATEGORY_OF]
->(t:category) WHERE t.catname =
"Food" AND b.shopdate >=
"01.08.2019" AND b.shopdate <=
"31.08.2019" RETURN SUM(b.subtotal)
```



SUM(b.subtotal)
214.25

Figure 7. Consumer’s spending pattern model for the whole household, for the category ‘Food’, during a given period of time, represented through an aggregated number

The consumer’s spending pattern model can be limited to a particular subcategory, as well. For instance, the consumer’s spending pattern model for the whole household during August 2019, for the subcategory ‘Bread and bakery’, which is visually depicted in Figure 8, can be obtained by running the following CQL code:

```
MATCH (c:customer), (p:product),
(s:subcategory {subcatname:"Bread
and bakery"}), (c)-[b:BUYS]->(p)-
[:BELONGS_TO]->(s) WHERE
b.shoptime>="01.08.2019" AND
b.shoptime<="31.08.2019"
RETURN c, p, s
```

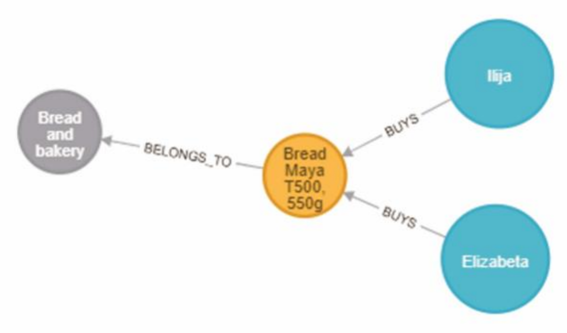


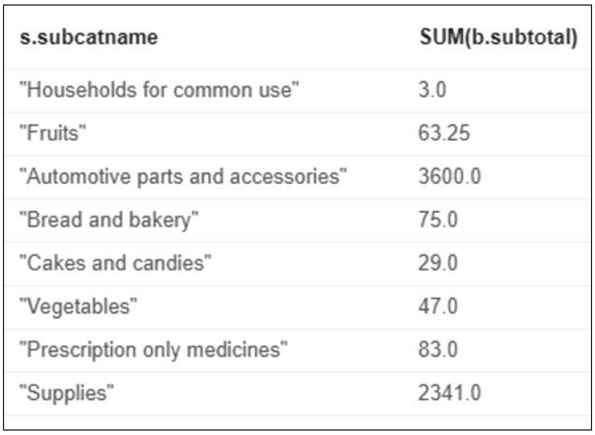
Figure 8. Consumer’s spending pattern model for the whole household, for the subcategory ‘Bread and bakery’, during a given period of time, represented visually

The following CQL code calculates the *total amounts spent by the family members on different subcategories* during August 2019 (Figure 9):

```
MATCH (c:customer), (p:product),
(s:subcategory), (c)-[b:BUYS]->(p)-
[:BELONGS_TO]->(s) WHERE
b.shoptime>="01.08.2019" AND
b.shoptime<="31.08.2019" RETURN
s.subcatname, SUM(b.subtotal)
```

Similarly to the sellers’ vector, the *consumers’ vector of a particular seller/merchant* shows visually the set of all shopping occasions including the members of a given family who shop at that particular seller/merchant (e.g. ‘Futura ITD d.o.o’) during a given period of time (e.g. August 2019), depicted in Figure 10, which can be visualized by running the following CQL query:

```
MATCH (c:customer), (s:seller),
(c)-[r:SHOPS_AT]->(s) WHERE
s.companyname="Futura ITD d.o.o."
AND r.shopdate>="01.08.2019" AND
r.shopdate<="31.08.2019"
RETURN c, s
```



s.subcatname	SUM(b.subtotal)
"Households for common use"	3.0
"Fruits"	63.25
"Automotive parts and accessories"	3600.0
"Bread and bakery"	75.0
"Cakes and candies"	29.0
"Vegetables"	47.0
"Prescription only medicines"	83.0
"Supplies"	2341.0

Figure 9. Consumer’s spending pattern model for the whole household, by different (all) subcategories of products, during a given period of time, represented through aggregated numbers

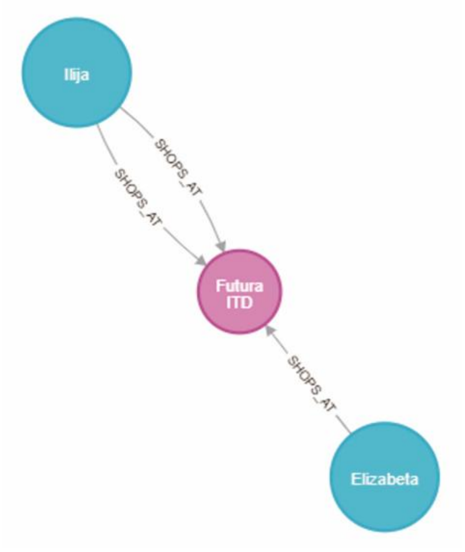
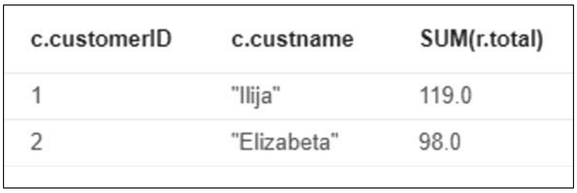


Figure 10. Consumer’s vector of a particular seller (‘Futura ITD d.o.o.’) for the whole household, during a given period of time, represented visually

The following CQL code calculates the total amounts spent by the family members with a particular seller during a specified period of time (Figure 11):

```
MATCH (c:customer), (s:seller),
(c)-[r:SHOPS_AT]->(s) WHERE
s.companyname="Futura ITD d.o.o."
AND r.shopdate>="01.08.2019" AND
r.shopdate<="31.08.2019"
RETURN c.customerID, c.custname,
SUM(r.total) ORDER BY c.customerID
```



c.customerID	c.custname	SUM(r.total)
1	"Ilija"	119.0
2	"Elizabeta"	98.0

Figure 11. Consumer's vector of a particular seller ('Futura ITD d.o.o.') for the whole household, during a given period of time, represented through aggregated numbers

Finally, the following CQL code provides an insight into the *set of products that are bought by the family members at a particular seller/merchant* (e.g. 'Futura ITD d.o.o.'), visually presented in Figure 12, during a specified period of time (e.g. August 2019):

```
MATCH (p:product), (s:seller),
(p)-[r:BOUGHT_AT]->(s) WHERE
s.companyname="Futura ITD d.o.o."
AND r.shopdate>="01.08.2019" AND
r.shopdate<="31.08.2019"
RETURN p, s
```

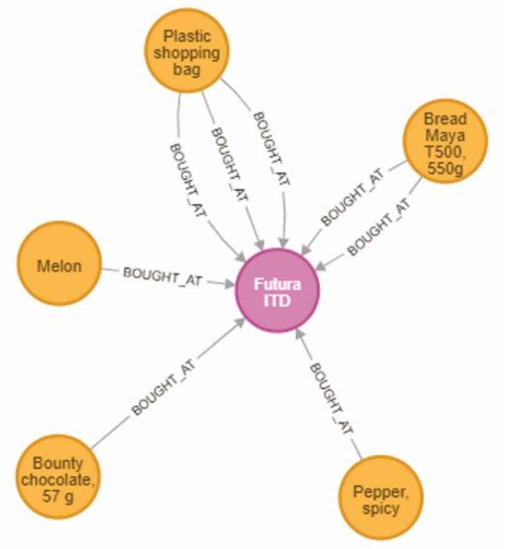


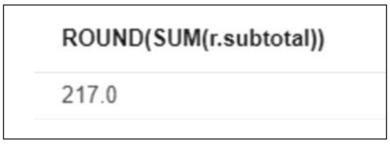
Figure 12. Set of products bought by the family members at a particular seller/merchant ('Futura ITD d.o.o.'), during a given period of time, represented visually

Based on the previous CQL query, the *total amount spent on products bought at a particular seller* during a specified period of time can be calculated by the following CQL code:

```
MATCH (p:product), (s:seller),
(p)-[r:BOUGHT_AT]->(s) WHERE
s.companyname="Futura ITD d.o.o."
AND r.shopdate>="01.08.2019" AND
r.shopdate<="31.08.2019"
RETURN ROUND(SUM(r.subtotal))
```

The execution of the above CQL code generates an output as shown in Figure 13. It should be notified that it is completely equivalent to the one presented in Figure 11.

All of the previously presented CQL examples can be easily modified in order to answer additional queries *vis-à-vis* the consumer's spending behavior.



ROUND(SUM(r.subtotal))
217.0

Figure 13. Total amount spent on products bought by the family members at a particular seller during a specified period of time represented through an aggregated number

IV. Conclusion

Graph databases address the shortcomings of relational databases for modeling and querying complex relationships because they treat relationships as separate objects to achieve superb performances and increase the readability of the model. They can help generate valuable insights from the existing data, especially in cases where relationships between data points matter more than the individual points themselves. This is a consequence of the fact that with graph databases, relationship information is treated as a 'first-class citizen'.

When it comes to modeling consumer spending behavior, the Neo4j graph database, along with the Cypher query language, provides an indispensable tool for conducting a myriad of analyses on specific graph subsets that reveal valuable insights, both visual and quantitative. If applied on a higher level, e.g. country level, such analyses can give an answer to highly important questions that can speak a lot where the national economy may be heading to, such as:

- What is the share of different expenditure categories (e.g. Food, Transportation, Personal care, Health care, Apparel, Entertainment, Housing-related expenditures, etc.) in the overall consumer expenditure?
- Are consumers spending less or more of their incomes on traditional retail items such as Apparel and Groceries?
- Just how have consumer spending patterns changed over time?
- What are the residents of a particular country doing with their growing incomes?

Answering these can significantly help policy-makers in making better decisions.

Apropos the proposed graph data model, it has the following features:

- It has a relatively simple, yet semantically rich structure, comprised of five categories of nodes and five categories of relationships;
- Despite its simple structure, the proposed graph data model allows one to perform myriad of CQL queries to obtain a profound understanding of the consumers' spending patterns;
- The proposed graph database model can be easily enriched by including additional constructs (i.e. nodes and/or relationships) to meet other, more sophisticated analyses' requirements, as needed.

REFERENCES

[1] H. Kileo, “Importance of Financial Management”, LinkedIn.com, 2016. [Online]. Available: https://www.linkedin.com/pulse/importance-financial-management-hamza-kileo/. [Accessed 26- Jun- 2019].

[2] S. Shah, “Chapter 2: Importance of Financial Management”, WikiFinancepedia.com, 2019. [Online]. Available: https://wikifinancepedia.com/finance/financial-management/top10-importance-of-financial-management. [Accessed: 26- Jun- 2019].

[3] W. Kenton, “What is Consumer Spending?”, Investopedia, 2018. [Online]. Available: https://www.investopedia.com/terms/c/consumer-spending.asp. [Accessed: 26- Jun- 2019].

[4] I. Robinson, J. Webber, and E. Eifrem, Graph Databases: New Opportunities for Connected Data. 2nd ed., Sebastopol, CA, USA: O’Reilly, 2015.

[5] FAO & WHO, “GSFA Online Food Categories”, Fao.org, 2019. [Online]. Available: http://www.fao.org/gsfaonline/foods/index.html?collapse=53. [Accessed: 07- Aug- 2019].

[6] M. Hunger, R. Boyd, and W. Lyon, “Ebook: The Definitive Guide to Graph Databases for the RDBMS Developer”. Neo4j Graph Database Platform, 2016. [Online]. Available: https://neo4j.com/whitepapers/rdbms-developers-graph-databases-ebook/. [Accessed: 10- Aug- 2019].

[7] M. Hunger, “From Relational to Graph: A Developer’s Guide”. DZone.com, 2016. [Online]. Available: https://dzone.com/storage/assets/2054302-dzone-refcardz-231-neo4j.pdf. [Accessed: 11- Aug- 2019].

[8] S. Yang, “How to Map Relational Data to a Graph DB in Four Steps”. Tibco.com, 2018. [Online]. Available: https://www.tibco.com/sites/tibco/files/resources/sb-graph-database-final.pdf. [Accessed: 11- Aug- 2019]

[9] Neo4j, “Model: Relational to Graph”, Neo4j Graph Database Platform, 2019. [Online]. Available: https://neo4j.com/developer/relational-to-graph-modeling/. [Accessed 12- Aug- 2019]

[10] M. Hunger, R. Boyd, and W. Lyon, “RDBMS & Graphs: Relational vs. Graph Data Modeling”, Neo4j Graph Database Platform, 2016. [Online]. Available: https://neo4j.com/blog/rdbms-vs-graph-data-modeling/. [Accessed: 13- Aug- 2019].