Object Type-based Data Description by Aggregation for Graph Databases

Adam Kleinedler and Adam Dudáš*
Matej Bel University
Banská Bystrica, Slovakia
*corresponding author: adam.dudas@umb.sk

***Abstract*—Aggregation of attribute values into a set of computed measures is one of the most widely used methods for a brief description of data in the initial phases of data analysis. This description by aggregation poses no problems in conventionally structured relational data, but issues arise as soon as aggregation functions run on non-relational semi-structured or unstructured data, such as the data stored in document or graph databases. Since in this work the latter of the two database types is considered, this study proposes design and implementation of two-phase aggregation based on object type and its structural sub-types in graph databases based on the Neo4j system. For the evaluational purposes, a synthetic graph database of specific structure is prepared, and the proposed aggregation model is examined and compared to the human-based aggregation of fields of the database objects.**

I. Introduction

Data aggregation in descriptive data analysis is a process, which involves summarization and consolidation of data with the objective of knowledge extraction. This process typically includes the computation of statistical values to reveal patterns and trends – the functionalities, which are simple in structured sets of data – eg. relational databases, where organization of data into tables with predefined schema enables efficient groupings and computation [1], [2]. However, when dealing with semi-structured data stored in non-relational databases, aggregation becomes more challenging due to inconsistent schemas, nested structures, and missing values [3].

One of the most common types of non-relational databases is the graph database, where the data is stored in the form of interconnected objects [4], [5]. Each of these objects has its own structure composed of $key : value$ pairs or fields describing properties of the modelled object. Naturally, since each of the objects can be structured differently, aggregation over such data is more than complex [6].

This motivates the main objective of the presented work – design and implementation of aggregation model for description of data stored in graph databases. This aggregation technique is based on the so-called two-phase description by aggregation, where in the first phase, object types and subtypes of the same structural form are identified in the graph database, and in the second phase, each field of these structural sub-types is aggregated via conventional aggregation metrics.

Hence, the novelty of the work can be described in the following points:

- Design of two-phase description by aggregation model for descriptive analysis of graph databases.
- Implementation of the proposed model in the larger original graph analysis tool for Neo4j graph databases.
- Examination and evaluation of the proposed model on the original synthetic graph database of a lifestyle store.

The rest of the work is structured into four main sections. In Section II, works related to the graph databases and aggregation of semi-structured or unstructured data is presented. Section III discusses general description of data by aggregation and presents the proposed model for graph databases. The implementation of the aggregation technique is then examined and evaluated via a case study on the synthetic database presented in Section IV. Conclusions for the research and potential future work areas are defined in Section V.

II. Related Work

Since this study focuses on aggregation methods in graph databases, in the following section, a brief overview of works related to graph database analytics and aggregation in the context of semi-structured data is presented. Focus is put on our previous work related to these topics and on other modern methods of descriptive data processes utilizing aggregation models.

Our previous work presented in [7] explores the necessity of effective visualization techniques in the context of graph databases. While traditional visualization tools often lack interactivity and efficiency, the proposed model introduces two graphical approaches – the standard, topological layout and the novel clustered layout of property graph of graph database. These models focus on scalability, object development, and effective representation. The advantages of this method of visualization are its' effectiveness, model variety, and object representation. The main result of the study is in the implementation of a novel graph database analysis system, which was extended in [8]. Since there is a growing trend of low and no-code programming, which minimizes dependence on code itself with the use of visual tools, such as drag-and-drop interfaces, and graph databases are well known for their structural complexity and intricate querying requirements, the extension of the graph database analysis system was focused on such no-code graph database techniques. The proposed toolset integrates forms, automatic query translation via large language models, and improved visualization of query results based on subgraph embeddings.

Authors of [9] highlight the increasing demand for efficient data processing and evaluation, particularly within temporal databases that track object state evolution. The presented study focuses on Oracle databases, which provide a powerful solution to the problem by incorporating indexing, partitioning, and – importantly – advanced analytical functions, allowing scalable data processing. The work specifically examines the performance of aggregational grouping in SQL functions – the process, where groups are determined not only by attributes but also by function results and transformations.

The work presented in [10] focuses on the role of partitioning of data in managing large datasets effectively. Partitioning helps organize tables into smaller, more manageable segments based on a partition key column, but it also brings challenges when records contain missing values. In the work, authors investigate the behavior of such entities in partitioned tables and propose a methodology to handle them efficiently. Based on the analysis of range, list, and hash partitioning, the study concludes that modifications in table and partition definitions can enable the insertion of records with missing partition keys, improving database flexibility and management.

When focusing on graph databases, in [11], authors present a concept for the aggregation and historicization of production entities into a graph database based on the original data processing and analysis pipeline process. In this process, the data type description is sent to the type library present on the aggregation server, which then dynamically aggregates the data using the aggregation agent and visualizes it through GraphQL methods.

On the other hand, the authors of [12] focus on the original toolkit for exploratory network data analysis (GRANEF) based on associations stored in a graph database. The study describes data transformation principles, the use of scalable graph databases, and data analysis techniques applicable in the proposed system and discusses the use of real-world data in the context of the proposed model.

III. Description by Aggregation in Graph Databases

The process of data analysis is conventionally separated into several phases, steps, or types of analyses. Some of these analysis types include exploratory data analysis based on the concepts of effective visualization of data and dimensionality reduction; predictive analysis of data, in which the logical predictions about data are created on the basis of identified trends; diagnostic analysis, where the cause of discovered relationships, trends and patterns is determined, and so on. For all of these data analysis types, the descriptive analysis forms basis of decision-making. In this analysis type the dataset is described with the use of various techniques from statistics, visualization, and the area of interest studied in the data [13].

One of the strongest initial descriptors of any dataset is its aggregation into a set of common values for all the dataset's attributes – the process called description by aggregation.

A. Conventional Aggregation Metrics

In the descriptive analysis of data based on descriptive statistics two sets of aggregation methods are used for the systematic identification of basic properties of data in an examined dataset. First set of these aggregation measures can be labelled as Central Tendency Measures (CTMs for short), which describe points and values of the dataset around which the data is naturally grouped, or values which describe crucial points of the studied dataset [14]. The second set of techniques is commonly referred to as Variability Measures (or VMs). Using these measures, analysts can compute the extent to which the data values in the dataset differ from each other or how distributed the data is in the considered space [15].

In the context of the work presented in this paper, we consider several basic centrality measures, which are most often used for the descriptive purposes in the initial phases of data analysis. For the identification of upper and lower bounds (or extremes) of the space described by the considered aggregated attribute of the dataset, its maximal ($\min$) and minimal ($\max$) values are computed.

Then, the measures describing the middle values of the studied dataset are computed. These metrics include mean value of an attribute A, which is computed as [14]:

$$\mu(A) = \frac{\sum_{i=1}^{n} A_i}{n} \tag{1}$$

for a dataset of n measurements, and median value of such an attribute defined as [14]:

$$median(A) = (\frac{n+1}{2})th \tag{2}$$

element of sorted array of the attribute values. Naturally, the difference between the two middle value metrics lies in the fact that the mean value is computed from the values of an attribute, while the median is one of the real values located in this attribute.

To somewhat describe the space and distribution of values between the mean or median and the extremes of the studied attribute A, this work also considers first ($Q1$) and third ($Q3$) quartile values, defined as [16]:

$$Q1(A) = \frac{\min(A) + \mu(A)}{2} \tag{3}$$

$$Q3(A) = \frac{\mu(A) + \max(A)}{2} \tag{4}$$

Generally speaking, variability measures used for the descriptive analysis of data consist of two strongly related metrics – standard deviation and variance. The standard deviation of attribute A ($\sigma(A)$) describes the distribution of values of attribute A when compared to its mean value, and therefore, it is measured as the mean distance between all of the attribute's values and its mean [15]:

$$\sigma(A) = \sqrt{\frac{\sum_{i=1}^{n}(A_i - \mu(A))^2}{n-1}} \tag{5}$$

Variance of attribute A ($V(A)$) is alternative to the standard deviation computed as:

$$V(A) = \sigma(A)^2 \tag{6}$$

For the purposes of this study, only the standard deviation of fields in graph database objects is considered.

B. Object Type-based Aggregation

The conventional problem in the process of description of dataset by aggregation is the semi-structurality or unstructurality of data − a well-know issue caused by missing values, which motivated the design and use of non-relational databases. Naturally, one such database types of interest are graph databases, in which each of the objects can be defined by its local structure − a set of key : $value$ pairs or fields. Since descriptive analysis (and description of data via aggregation) is one of the foundational methods of data analysis, the motivation for the graph database-specific analytical models arose.

In this work, we propose the two-phase description by aggregation process for data stored in graph databases (see Fig. 1), which can be summarized as follows:

1) Identification of object types and structural sub-types − After the reading of input data in the form of graph database, the proposed two-phase description by aggregation model identifies all of the object types present in the data. This step is conducted based on the basic labelling of objects in the database, yet this classification of objects into their types is not sufficient since the types do not have a stable scheme or structure. Therefore, the second step of the first phase focuses on the identification of object sub-types on the basis of the local object structure. In this step, all the objects with the same set of keys from key : $value$ pairs are grouped under the label of the same sub-type, eg. sub-type 1, sub-type 2 and so on. In this way, the first aggregation of graph database object data into object types and their structural sub-types is done.
2) Aggregation of individual fields − Since the first phase of the proposed process outputs objects classified into groupings of object sub-types, which are homogeneously structured, in the second phase of the method, the conventional description of individual object sub-types via descriptive analysis metrics can be conducted. For the purposes of the proposed project, each field in the sub-type is described by its minimum, first quartile, mean, median, third quartile, and maximum.

IV. Case Study of Object-based Aggregation in Graph Databases

Based on the design presented above, an implementation of object type-based aggregation for Neo4j graph databases [17] was created in TypeScript. The aggregation module implemented in this way is included in the tool focused on the analysis of graph databases, which was designed and implemented in our previous work in [7] and improved upon in [8].

For the purposes of examination of the proposed aggregation model, we created original, synthetic graph database describing a lifestyle store, which consists of three types of objects:

- 10 objects of Customer type − customers of the mentioned e-shop.
- 10 objects of Employee type − employees of the lifestyle shop.
- 10 objects of Person type − objects describing calorie data of people requesting dietary advice in the shop.

Since the data in this database was synthetically created to verify the proposed aggregational analysis concept, we have full control over the variability of the structures of its individual objects. Specifically, the objects are intentionally divided based on their variant structure as follows:

All Customer-type objects have a homogeneous structure consisting of the following key : $value$ pairs which can be aggregated:

- Customer structural sub-type 1:
  - $visit\_frequency$−attribute describing how often the customer visits the store in a month.
  - $purchase\_amount$−total price paid by the customer for their purchases.
  - $items\_bought$−a value that indicates how many products a given customer purchased.

The Employee-type objects are structurally divided into two structural sub-types:

- Employee structural sub-type 1:
  - $productivity\_score$−score of employee in the range of $0 - 100$.
  - $experience$−number of years of experience of the employee.
  - $projects\_completed$−number of completed projects of the employee.
  - $salary$−ammount paid montly.
- Employee structural sub-type 2:
  - $productivity\_score$
  - $experience$
  - $projects\_completed$

Finally, objects of the Person-type can be classified into the three separate structure types:

- Person structural sub-type 1:
  - $exercise\_hours$−the number of hours of exercise a day.
  - $sleep\_hours$−the number of hours of sleep a day.
  - $hydration$−volume of fluids drunk per day in liters.
  - $calories\_burned$−the amount of calories burnt throughout the day.
  - $heart\_rate$−average heart rate for a day.
  - $steps$−average number of steps done in a day.
- Person structural sub-type 2:
  - $exercise\_hours$
  - $hydration$

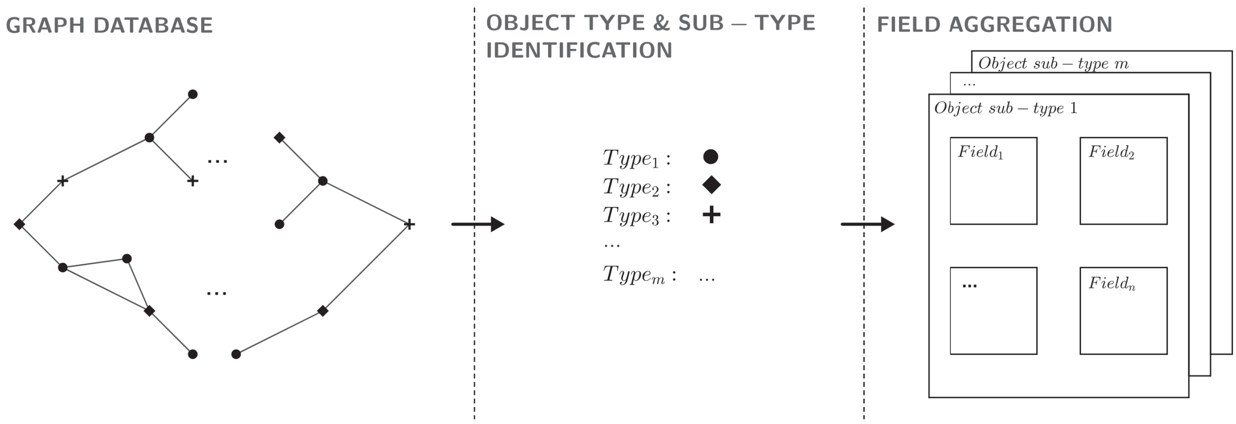

Fig. 1. Schema of the proposed two-phase description by aggregation for graph databases

  - *calories_burned*
  - *heart_rate*
  - *steps*
- Person structural sub-type 3:
  - *exercise_hours*
  - *sleep_hours*
  - *hydration*
  - *calories_burned*
  - *steps*

This database is used to evaluate the proposed aggregation model from the two points of view – identification of individual object types and their sub-types in the data and the description of the identified object types using aggregation methods. For the purposes of evaluational comparison, the tabulation of the database is presented in Table I.

A. Identified Types and Sub-types of Objects

The first functionality of the proposed aggregation module of the graph database analysis system focuses on the identification of object types and their structural sub-types in the examined database. Fig. 2 presents the grid of object buttons for the lifestyle store. As can be seen in the figure, each of the buttons corresponds to one of the object types present in the database – Customer, Employee, and Person.

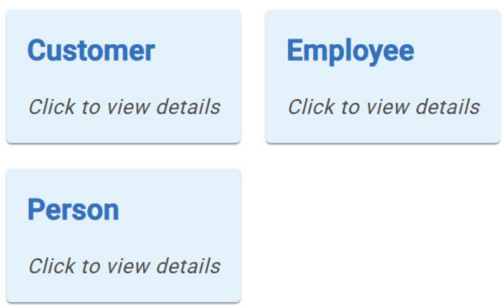

Fig. 2. The Basic Types of Objects Identified in the Used Graph Database

Additionally, the *Click to view details* option is visualized in the context of each button. After selecting the object of interest by clicking the button corresponding to the object type, each structural sub-type of object with its related sub-types is presented.

For the lifestyle store database, the module correctly identified individual structural sub-types of objects based on the number and content of *key* : *value* pairs. Therefore, the following object sub-types were identified:

- Customer objects – one structural sub-type.
- Employee objects – two structural sub-types.
- Person objects – three structural sub-types.

B. Description of the Identified Object Types by Aggregation

After the identification of structural sub-types of a database, each of these sub-types has its own description by aggregation as proposed in Section III – this description consists of minimum, first quartile, median, mean, third quartile, maximum, and standard deviation measurements for each field of the object sub-type.

Fig. 3 presents the description by aggregation for the single identified structural sub-type of the Customer objects of the examined graph database. When compared to the values presented in Table I, we can see both – the correctness of identification of structural sub-types, and the correctness of aggregation values.

In the context of Employee objects, the proposed model identified two structural sub-types, the aggregation of which is presented in Figs. 4 and 5. As can be seen, the values of descriptive aggregators are consistent with the tabulation of data presented in Table I.

Lastly, Figs. 6 - 8 show the aggregation output of the proposed module for the three structural sub-types of Person objects. Comparing the values of aggregation functions to the tabular representation of the data shown in Table I, we can see the correct identification of three aggregated sub-types and the aggregated values themselves.

TABLE I
TABULATION OF DATA STORED IN LIFESTYLE STORE GRAPH DATABASE

Object type	Object structure
Customer	{visit_frequency: 12,purchase_amount: 150.75,items_bought: 3,name: ”Alice”,id: 1}
	{visit_frequency: 18,purchase_amount: 320.5,items_bought: 5,name: ”Bob”,id: 2}
	{visit_frequency: 22,purchase_amount: 215.0,items_bought: 4,name: ”Charlie”,id: 3}
	{visit_frequency: 5,purchase_amount: 80.3,items_bought: 2,name: ”David”,id: 4}
	{visit_frequency: 30,purchase_amount: 400.0,items_bought: 7,name: ”Eva”,id: 5}
	{visit_frequency: 10,purchase_amount: 95.0,items_bought: 2,name: ”Frank”,id: 6}
	{visit_frequency: 25,purchase_amount: 250.6,items_bought: 6,name: ”Grace”,id: 7}
	{visit_frequency: 20,purchase_amount: 310.2,items_bought: 8,name: ”Hank”,id: 8}
	{visit_frequency: 8,purchase_amount: 120.9,items_bought: 3,name: ”Ivy”,id: 9}
	{visit_frequency: 15,purchase_amount: 275.4,items_bought: 5,name: ”Jack”,id: 10}
Employee	{productivity_score: 85,name: ”Alice”,experience: 5,projects_completed: 20,salary: 75000}
	{productivity_score: 78,name: ”Bob”,salary: 67000,experience: 4,projects_completed: 18}
	{productivity_score: 92,name: ”Charlie”,experience: 6,projects_completed: 25,salary: 85000}
	{productivity_score: 65,name: ”David”,experience: 3,projects_completed: 12,salary: 55000}
	{productivity_score: 88,name: ”Eve”,salary: 80000,experience: 5,projects_completed: 22}
	{productivity_score: 75,name: ”Frank”,experience: 4,projects_completed: 17}
	{productivity_score: 80,name: ”Grace”,experience: 4,projects_completed: 19,salary: 72000}
	{productivity_score: 95,name: ”Hank”,experience: 7,projects_completed: 30}
	{productivity_score: 70,name: ”Ivy”,salary: 62000,experience: 3,projects_completed: 15}
	{productivity_score: 90,name: ”Jack”,salary: 88000,experience: 6,projects_completed: 28}
Person	{exercise_hours: 2,sleep_hours: 7,hydration: 2.5,calories_burned: 2400,name: ”Alice”,heart_rate: 72,steps: 12000}
	{exercise_hours: 1.5,sleep_hours: 6.5,hydration: 2.0,calories_burned: 1800,name: ”Bob”,heart_rate: 75,steps: 8500}
	{exercise_hours: 2.5,sleep_hours: 8,hydration: 3.0,calories_burned: 2800,name: ”Charlie”,heart_rate: 68,steps: 15000}
	{exercise_hours: 2,hydration: 2.2,calories_burned: 2300,name: ”David”,heart_rate: 70,steps: 11000}
	{exercise_hours: 1.7,hydration: 2.1,calories_burned: 1900,name: ”Eve”,heart_rate: 74,steps: 9000}
	{exercise_hours: 2.2,hydration: 2.8,calories_burned: 2500,name: ”Frank”,heart_rate: 69,steps: 13000}
	{exercise_hours: 1.8,sleep_hours: 7.5,hydration: 2.3,calories_burned: 2200,name: ”Grace”,steps: 10000}
	{exercise_hours: 1.2,sleep_hours: 6,hydration: 1.8,calories_burned: 1600,name: ”Hank”,steps: 7000}
	{exercise_hours: 2.4,sleep_hours: 7,hydration: 2.7,calories_burned: 2700,name: ”Ivy”,steps: 14000}
	{exercise_hours: 1.5,sleep_hours: 6.8,hydration: 2.1,calories_burned: 2000,name: ”Jack”,steps: 9500}

V. CONCLUSION

In the scope of this study, a module aimed at the technique of description by aggregation for data stored in graph databases was designed, implemented, and examined in the context of the controlled case study on a synthetic lifestyle store database. The proposed module focuses on the two-step aggregation process, in which identification of types and subtypes of objects in a graph database is done as a first phase of aggregation and then conventional aggregation of values of object fields for each structural sub-type of object is conducted as a second phase of the aggregation.

As described in Section IV of this work, results of the proposed aggregation process were examined from the point of view of functionality and comparison with tabularisation of semi-structured data stored in the used graph database. Reached results are deemed satisfactory from both points of view – the aggregation technique allows simple, in database, description by aggregation for fields of objects in graph database, and it reaches precision needed for proper further decision-making. Estimated computational complexity of the proposed two-phase aggregation approach is $O(n^2)$ for dense graphs caused mainly by element-wise node traversal algorithms.

When working on the design and implementation of the proposed description by aggregation module for graph databases, several future work areas naturally arose. Specifically, these areas can be broadly summarized into:

- Since the semi-structurality of the graph database data poses a problem for conventional machine learning algorithms, a piece-wise decision-making model similar to the presented two-phase aggregation would be highly utilizable. Example of such a model would be object type-based decision tree, which woul classify the objects of one type to various classes based on some of its fields and values, or regressors focused on quantitative value estimation for the fields of objects of specific type.
- As mentioned in the previous sections of the work, the proposed aggregation model is implemented in the Neo4j system only. This offers a possibility for generalization of the model for other graph database systems and platforms, such as JanusGraph or TigerGraph.
- The proposed aggregation approach focuses on object types and their structural sub-types. Exploration of alternative aggregation strategies based on other criteria (eg. relationship-driven aggregations) could enhance the descriptive power of the method.

DATA AND CODE AVAILABILITY

Graph database used for the case study and code for the Object Type-based Aggregation in graph databases is available at:
github.com/Eldam804/GraphDatabaseViewerImproved

Fig. 3. Aggregation of Customer Object Fields

ACKNOWLEDGEMENT

This work was supported by the VEGA 1/0192/24 project – Developing and applying advanced techniques for efficient processing of large-scale data in the intelligent transport systems environment.

REFERENCES

[1] M. Kvet. Effective Restriction of the Data Manipulation Operations in Oracle Database. In the Proceedings of 36th Conference of Open Innovations Association FRUCT, 2024. DOI: 10.23919/FRUCT64283.2024.10749869
[2] M. Kvet. Methodology of Bivalent Data Management in Oracle Database. In the Proceedings of International Conference on New Trends in Signal Processing NTSP, 2024. DOI: 10.23919/NTSP61680.2024.10726314
[3] L. de Espona Pernas et al. Automatic Indexing for MongoDB. Communications in Computer and Information Science, 2023. DOI: 10.1007/978-3-031-42941-5_46
[4] I. Robinson et al. Graph Databases. O'Reilley Media, 2015. ISBN: 9781491930892
[5] M. Besta et al. Demystifying graph databases: Analysis and taxonomy of data organization. System Designs, and Graph Queries. ACM Computing Surveys, 2019. DOI: 10.1145/3604932
[6] S. Timón-Reina et al. An overview of graph databases and their applications in the biomedical domain. Database, 2021. DOI: $10.1093/database/baab026$
[7] A. Dudáš and A. Kleinedler. Effective Visualization of Data Structures in Graph Databases. Journal of Image and Graphics, 2024. DOI: $10.18178/joig.12.3.283-291$
[8] A. Kleinedler and A. Dudáš. Towards Low and No Code Programming in Graph Databases with Forms, Natural Language Querying, and Visualization. Lecture Notes in Networks and Systems, 2025. $(in\ print)$
[9] M. Kvet. Performance Analysis of the Data Aggregation in the Oracle Database. Lecture Notes in Networks and Systems, 2024. DOI: 0.1007/978-3-031-60328-0_16
[10] M. H. Durneková and M. Kvet. Referring Null Values in Partitioned Tables. In the Proceedings of 35th Conference of Open Innovations Association FRUCT, 2024. DOI: 10.23919/FRUCT61870.2024.10516341
[11] S. Schmied et al. An approach for aggregation and historicization of production entities in the graph. In the Proceedings of 25th IEEE International Conference on Emerging Technologies and Factory Automation, 2020. DOI: 10.1109/ETFA46521.2020.9211907
[12] M. Čermák and D. Šramková. GRANEF: Utilization of a Graph Database for Network Forensics. In the Proceedings of the18th International Conference on Security and Cryptography, 2021. DOI: 10.5220/0010581807850790
[13] D.P. Salgado et al. WheelSimAnalyser: A MATLAB tool for multimodal data analysis of WheelSimPhysio-2023 dataset. Software Impacts, 2025. DOI: 10.1016/j.simpa.2024.100731
[14] S.S. Skiena. The Data Science Design Manual. Springer, 2017. DOI: 10.1007/978-3-319-55444-0
[15] A.F. Siegel. Variability: Dealing with Diversity. Practical Business Statistics, 2017. DOI: 10.1016/B978-0-12-804250-2.00005-5
[16] J. Sheard. Quantitative data analysis. Research Methods, 2018. DOI: 10.1016/B978-0-08-102220-7.00018-2
[17] Neo4j. [Online]. Available at: $neo4j.com$

Fig. 4. Aggregation of Employee Object Sub-type 1 Fields

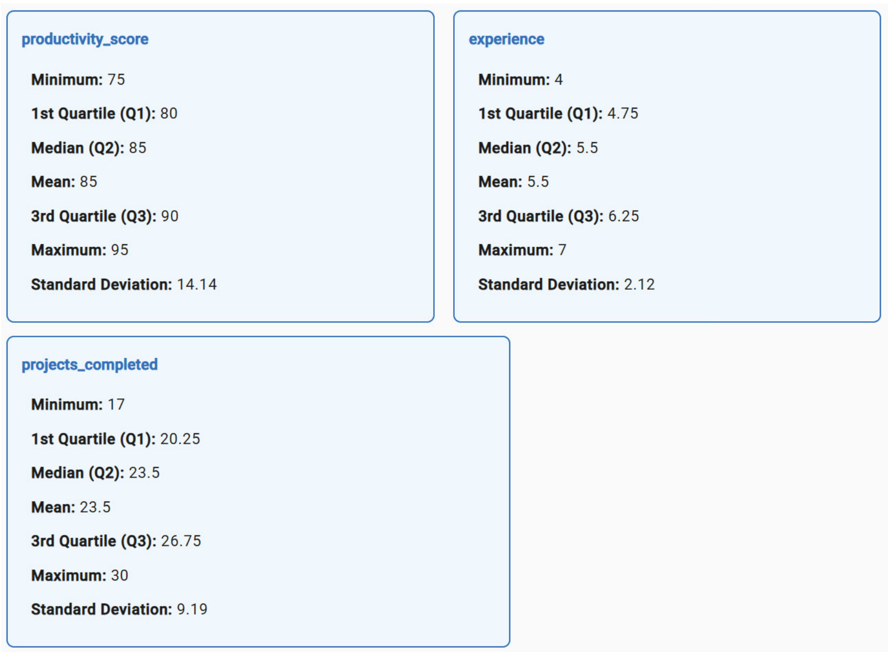

Fig. 5. Aggregation of Employee Object Sub-type 2 Fields

Fig. 6. Aggregation of Person Object Sub-type 1 Fields

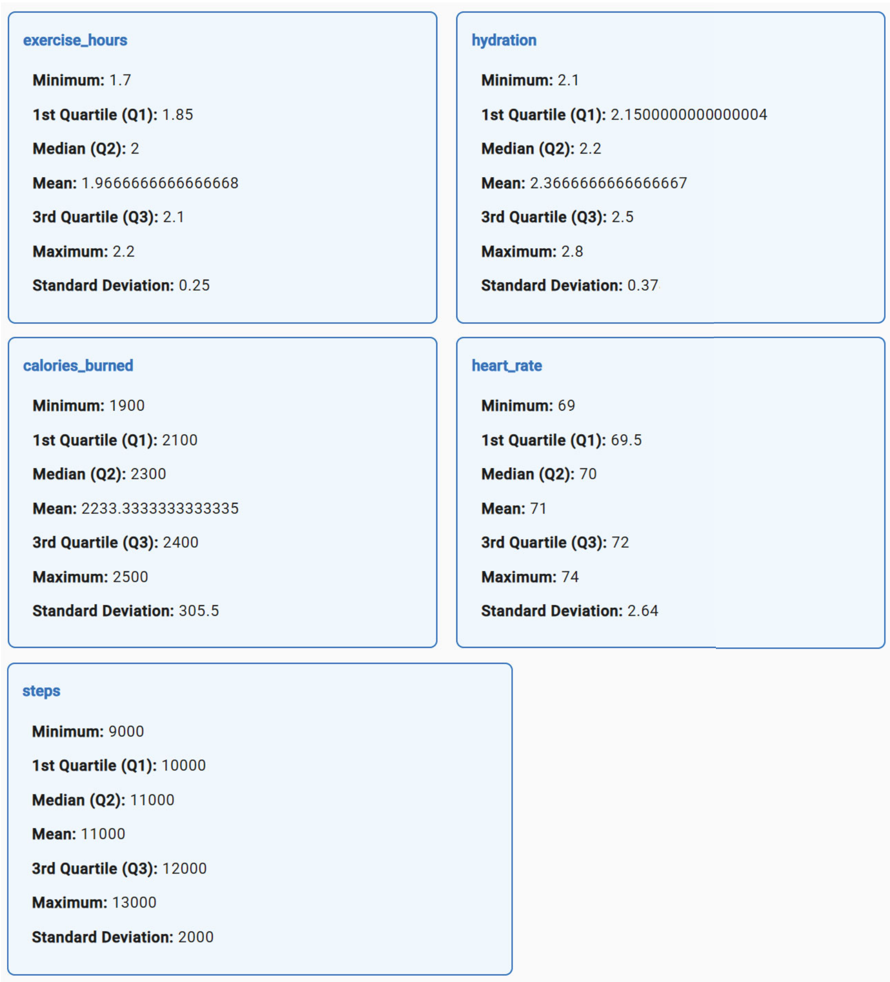

Fig. 7. Aggregation of Person Object Sub-type 2 Fields

Fig. 8. Aggregation of Person Object Sub-type 3 Fields