QUESTION PATTERNS FOR NATURAL LANGUAGE TRANSLATION IN SQL QUERIES

Mariya Zhekova, George Totkov*

Computer Systems and Technologies Department
University Of Food Technology, Plovdiv
Bulgaria

* Corresponding Author, e-mail: m_jekova@uft-plovdiv.bg

Abstract: The ability to use a natural database query language has the potential to attract many users who have no experience in using database query manipulation languages such as SQL, which is the primary language for storing and retrieving queries in relational databases. The aim here is to solve the problem of transformation and machine interpretation of the natural language. In this article we offer an approach to solving the problem of translating from context free text into an executable SQL query. We propose an approach based on grammatical structures called question-patterns of user questions, where each question-pattern model contains a dependency graph. In addition, we offer a set of functions, procedures and checks to synthesize the correct SQL query based on the query patterns when the word order in the user query does not matter. The question-pattern is recognized by predefined syntactic structure of a sequence of SQL statements. The most important thing is to assess the dependence of the relationship defined in the question.

Key words: SQL query patterns, SQL constructions, SQL statements, Natural language translating to SQL.

1. INTRODUCTION

Conscious and rational human activity is associated with the generation, storage, processing, management and use of a significant amount of information. In each of these activities, the work of analyzing and extracting useful information for consumers is of paramount importance for the end result. Data sources are diverse, but the extraction of information from them does not depend on the organization and subject area of the primary data.

The fact is that a system can extract the necessary information from the database if the information is structured so that the system can read and process it. Understanding the meaning of words is a serious challenge for researchers working in the field of natural language processing.

The Structured Querying Language (SQL) is the standard querying language for relational databases [1]. Any interaction of a software application to the data it processes requires either a graphical user interface or experience in the language for structured database queries [2]. Often the menus have fixed options and sometimes do not cover the specific needs of users [3]. Instead, natural language interfaces allow the user to communicate directly with the database (DB), in the form of commands that translate a natural language question in a database query.

Software systems with natural language interface (NLI) for working with databases allow to ask questions that are formulated in the native language of the user. Building a system with natural language interface not only for direct communication to databases, but also helps to provide access to data and structures designed on the basis of semantic and syntactic classes and relationships [4].

In order for a software system to understand a user question in its natural language, it is necessary to introduce the term natural language processing (NLP). This type of processing in systems in which the interaction with users is carried out in natural language, carries out its activities through tools such as keywords and phrases, regular expressions, finite automats, rules and techniques for inferences, statistical methods and more.

NLP is based on computer models of linguistic phenomena, creates and uses standardized language resources (corpora) and uses methods for comparison and evaluation. The processing includes processes such as segmentation, lemmatization /normalization, conversion of a sequence of words into a semantic representation, useful for subsequent processing. This includes processing unstructured text and retrieving names, dates, organizations, events, and more [5]. For this purpose it is necessary to study the structure, content and meaning of word forms in the subject area.

The subject of the research are methods and algorithms for the formation of a SQL query to response from a system with a natural language interface to a database.

To solve the task we must answer the question – *"How many question-patterns are needed to generate a correct SQL query to database"*. In this article, we propose an approach to solving the problem of out-of-context translation in an executable SQL query. This allows the synthesis of a large number of SQL queries, but limits the variation between them only to that covered by their patterns. We consider the list of attributes, parameters and conditions mentioned in the question-pattern's description.

2. RELATED WORK

Some of the commonly used approaches in natural language processing systems and tools described in [6] are:

- analyzes of a custom query text based on an additional relational structure that links the query elements to table and column names [7];
- analyzes of a user string using a dependency analyzer and proposing a candidate for SQL queries using heuristic rules [8];
- obtaining an SQL template using a semantic analyzer and subsequent formatting based on the content of the query [9];

- translation of a user query into an SQL query by analyzing a number of language expressions, including user feedback, in order to correctly interpret the question [10];
- reducing the language units from the input text to predefined sets to define the expressions in the WHERE clause of the database query [11].

Data is often stored in different ways depending on the storage mechanisms in structural, tabular, unstructured, graph (text) or hybrid format [12]. Depending on this, the information retrieval strategy requires different languages for knowledge base queries. This diversification complicates the task of translating a natural language text into a structured request for extracting information from a database.

The standard approach according to [13] for solving the problem of NLP is to treat both the natural language description and SQL query as sequences and train a sequence-to-sequence model or its variants which can be used as the parser.

A novel approach called **SQLNet** to fundamentally solve this problem by avoiding the sequence-to-sequence structure when the order does not matter is proposed in [13]. In particular, they employ a sketch-based approach where the sketch contains a dependency graph so that one prediction can be done by taking into consideration only the previous predictions that it depends on. The neural network SQLNet, is used to predict the content for each slot in the sketch. The authors propose a sequence-to-set model as well as the column attention mechanism to synthesize the query based on the sketch. Their approach will necessarily require the SQL queries to be serialized. They said that the most challenging part is to generate the WHERE clause. In their approach, SQLNet employs the sketch to provide the dependency relationship of different slots so that the prediction for each slot is only based on the predictions of other slots that it depends on. To implement this idea, the design of SQLNet introduces two novel constructions: sequence-to-set and column attention. The first is designed to predict an unordered set of constraints instead of an ordered sequence, and the second is designed to capture the dependency relationship defined in the sketch when predicting.

The neural network **Seq2SQL** synthesis approach described in [14] also employs a sketch-based approach, although their sketch is more coarse-grained and they employ a sequence-to-sequence structure to fill in the most challenging slot in the sketch. In [14] the authors offer a neural-network model that translates a natural language question into a SQL query that returns the correct answer from a relational database. Their Seq2SQL model leverages the structure of SQL queries to significantly reduce the output space of generated queries.

3. APPROACH OF NATURAL LANGUAGE PROCESSING

The solution to the problem of understanding natural language can be realized in two main stages: linguistic processing and programming. In the first stage, the text request is received through the natural language user interface of the application. During the language processing the request in natural language is analyzed and linguistically processed – the individual tokens are identified and normalized, the synonyms are searched in the synsets of the concepts. From the tags / labels obtained during the processing, which are directly related to the database, the dependencies are found, the arguments in them are initialized and their corresponding values are found as names of

tables, columns and keys. At this stage, the goal is to obtain a list (set) of parameters that corresponds to the string of the user string in question. In the second stage – coding, using appropriate methods and functions, the question template is translated into the corresponding SQL query, and then executed by the system.

According to [15], the natural language processing process involves input by the user in the form of a natural language, which is obtained from a natural language interface, and the result of the processing also in a language form understood by the application software using this process. The software that performs this process needs a language to be able to translate the incoming text into a database retrieval request.

The translator from a custom text query to an SQL query solves the problem of translating and comparing the contents of a user query with an executable query to a relational database.

The research work on the model was a challenge due to the requirement for it to identify the exact query template that would interpret the input data from the text-query and logically accurately extract an answer from the database. The task is to build regular expressions (question templates / subgraph patterns), which largely correspond to the grammar of the SQL language. To generate more complex queries, it is necessary to define templates with more complex syntax.

The following two examples will illustrate the specific subtask:

1. Show / Find / Extract / Which are all disciplines and their codes?

```
SELECT d.Name, d.Code
FROM Discipline
```

2. How many of the disciplines for the specialty KST are optional?

```
SELECT COUNT(DISTINCT d.Name)
FROM Discipline as d
INNER JOIN Specialty as s ON s.ID = d.SpecialtyID
WHERE d.IsOptional='true' AND s.Name='Computer science'
```

Both questions seek information about university disciplines. But the form of the logical query, which corresponds to the content of the question, is radically different.

This component model aims to formalize control sequences (subgraphs), which we will call prototypes (patterns) of user questions. The graph naturally captures the connections between concepts (objects), as they are connected to each other through a chain of relationships. The object types and the relationships between them serve to recognize the architecture of the SQL query, which describes a subset of the complex grammatical construction of the question. A graph compiled on the natural language text and the result of a proximity algorithm based on the obtained graph identifies the relationships between different objects in the given text-question and applies the concept of the template hidden in the question.

The algorithm for detecting a question pattern in text is based on global information, recursively calculated from the entire graph, instead of local information specific to a particular vertex in the subgraph. The main idea realized here is the proximity estimate obtained when one peak connects with another. The sequence of well-ordered unambiguous, computable operations here is based on a technique for extracting a grammatical structure from a source of knowledge – the Tree of Concepts.

The algorithm for modeling the relationship between the question in natural language and relational database includes identifying the set of parameters that are important for understanding the user question and for creating a question pattern.

Model $\mathbf{M_{qp}}$ of the question-pattern is an oriented graph $H_q = (V_q, E_q)$, which is a subgraph of concept's tree G_d, $G_d = (V_d, E_d)$, represents an ordered pair of sets, where V_d is a set of vertices (nodes) corresponding to the concepts of the domain, and E_d is the set of ribs defining the parent-child relationships between the concepts, $E_d \subseteq V_d \times V_d$:

- each vertex of $\mathbf{H_q}$ is also a vertex of G_d, $V_q \subseteq V_d$;
- each arc of $\mathbf{E_q}$ is an arc of G_d and each arc of $\mathbf{H_q}$ connects the same vertices as it connects in G_d;
- $V_q = V'_d \cup B$, the set of nodes V'_d ($V'_d \subseteq V_d, B \cap V'_d = \emptyset$) represents the concepts of domain area, B is a set of types of relations between concepts (for example AND, or, NOT);
- Param = (ParamType, ParamName) a ordered pair of "parameter type" and "parameter name";
- inputParams is n-number of input parameters Param;
- FindQuestionPattern (inputParams) is a piece of code, defined as a separate method, which is called when a user request text is received, and accepts a list of input parameters (normalized tokens).

The concept's tree graph about domain area, is made up of linguistic units (words and phrases) connected to each other by multiple connections between their vertices. It is logical and useful to include in the model the strength of the connection between two vertices, as the weight w_{ij} added to the corresponding arc (relation) that connects the two vertices. Taking into account the weights of the arcs, it will be possible to determine the closeness between the objects extracted from the query text and this will help to calculate the result related to determining the architecture of the structure of the question.

The concepts and values derived from the user text-query are used as mainstay fragments in generating the query-template, from which the actual SQL query to the database is later formed. It is a kind of compiler from natural language to standard SQL, i.e. when it finds an input concept or phrase, it replaces it with an appropriate SQL statement.

A set of functions related to the existence of conditions, aggregation functions or subqueries is performed on the subgraph thus obtained. When the input parameters are implicitly defined as objects from one table in the database, the approach for generating an SQL structure (template-question) is clear. But in case the question is composed and there are connected groups or subqueries in it, in order to fill in the desired joins in the pattern, functions for nested queries or join functions are used to connect tables.

The method for identifying the construction of the question template does not require in-depth language knowledge, nor specific to the software in question or linguistically annotated corpora, which makes it stable and transferable to other areas or languages.

The proposed methodology for forming a question-patterns is illustrated with sample tables. Each table shows the grammatical structure of the question-pattern,

indicates the necessary and sufficient conditions and validation functions and procedures for its implementation. Depending on the analysis of the input language parameters, the corresponding SQL query can be modeled differently.

The templates of the possible SQL-constructs of queries to a given database are not many. The question-patterns architectures proposed in this subsection are designed in a sufficiently general way to be able to cover all possible SQL database query constructs. Therefore, their use does not make it difficult to generalize the results in the proposed approach. The implementation of the method and the software implementation of the algorithm are subject of a separate publication.

4. MODEL OF A QUESTION-PATTERNS

4.1. Question-pattern 1 (QP1)

This pattern applies when the user search is only on a single table in the database.

Grammatical structure of user question – [Select | Show | Find | Extract] [all] X

The condition for application of this pattern is the found concept in input text to be the only one representative of the *TableN* class and no specific attributes are listed (see grammar constructions in Table 1). As a result, all records from the specific table are returned.

After the analysis of the user question and specifying the properties of the concepts involved in the application, if all the necessary conditions are met, therefore template 1 can undoubtedly be applied because there are no other attributes of class *TableN* or other objects of class *ColumnN*. There is only one specific table and the result is a list of all available records in it.

List of conditions:

- X is a table name
- X is a representative of a class *TableN*
- X is only one representative of a class *TableN*
- There are no other attributes of class *TableN* or other objects of class *ColumnN*

TableN is a node (concept, vertex) in the conceptual graph of this question-pattern. The corresponding SQL construct for this particular pattern is: SELECT * FROM X

Table 1. Table describing the grammatical structure of a type QP 1

Grammar structure of the user question	***SQL-query pattern 1***	***Example***
Which are | Find | Select | Show X?	*SELECT * FROM X*	*Select all teachers?* *List all curricula.*
Select | Show | Find | Extract [all] X?		
Find | Show | Select | Extract [a list of all] X?		

4.2. Question-pattern 2 (QP2)

This pattern applies when in addition to a representative of the *TableN* class, there are also representatives of the *ColumnN* class, which are the fields in the table in question. For example: “Show the names and faculty numbers of all students.”

In Table 2 there are summary grammar statement for this question's type.

Grammatical structure of the user question - [Show | Find | Which are | Select | Extract] A1, A2,…[and | ,] An [of | of all] X

In this pattern X is the only representative of the class *TableN*, and A1,,,,An – are the required attributes, representatives of the class *ColumnN*. The validations and checks that are mandatory in this case are:

- is there another representative of the *TableN* class in the user string
- do all concepts meet the condition to be columns of the only one table.

As a result, only the columns corresponding to the searched attributes from the specified table are returned.

X – a single instance of the *TableN* class

A1,…,An – *ColumnN* class representatives for which *isColumnOf (X) = true*

In the specific example the pattern can be applied, because after the performed analysis the presence of one representative of class *TableN* and necessarily one or more representatives of class *ColumnN* has been established.

List of conditions:

- X is a table name
- X is a representative of a class *TableN*
- X is only one representative of a class *TableN*
- A1, A2,… are the names of columns of X, representatives of class *ColumnN*

The SQL query construction corresponding to the template is: SELECT A1, A2,…, An FROM X

Table 2. Table describing the grammatical structure of a type QP 2

Grammar structure of the user question	***SQL-query pattern 2***	***Example***
Extract | Show | Find | Select A1 [A2, A3,…] [of] X?	*SELECT A1, A2, A3, ….. FROM X*	*Show PhD students' names, numbers and emails?*
Extract | Show | Find | Select A1 [A2, A3,…] [of all] X?		

4.3. Question-pattern 3 (QP3)

This pattern applies when the following language constructs are identified in the question: one table, *TableN* representative and specific (text / numeric) attribute values, representatives of class *ColumnN*. The condition is also fulfilled: *IsColumnOf (X) = true.*

Grammatical structure of the user question – [Show | Select | Find | Extract [all] X, [where | for which when] A1 = 'Value_1' [and | ,] A2 = 'Value_2' [and | ,]… An = 'Value_N'

In the template, as is shown in Table 3, X is the only representative of the class *TableN*, the terms A1,…,An are representatives of *ColumnN*, and 'value_1', 'value_2',… .. are values of attributes from the same table. For the regular expression (template) in question, the verification methods include:

- whether there is another representative of the *TableN* class in the user string;
- whether the found values meet the condition to be columns in the searched table.

As a result, all records that meet the specified conditions are returned.

X – only one instance of the *TableN* class

A1, A2,… – representative for which it is implemented: *isColumnOf (X) = true*

Therefore, the template can be applied if the analysis reveals the presence of only one representative of the *TableN* class and one or more representatives of the *ColumnN* class, attributes of the same table that meet a specific condition.

List of conditions:

- X is a table name
- X is a representative of a class *TableN*
- X is only one representative of a class *TableN*
- A1, A2,… are the names of columns of X, representatives of class *ColumnN*
- value_1, value_2,… are specific values from the columns in the table X

The corresponding SQL-query construction is:

SELECT * FROM X WHERE A1 = 'Value_1' AND / OR A2 = 'Value_2' AND / OR……

Table 3. Table describing the grammatical structure of a type QP 3

Grammar structure of the user question	***SQL-query pattern 3***	***Example***
Extract | Find | Select | Show [all of] X [where | when | who] [so] [is true] A1 = 'value_1' [and | or A2 = 'value_2'] ...	*SELECT * FROM X WHERE A1='value_1' and / or A2='value_2' ...*	*Extract all students who are 3rd year and study the discipline 'Computer Engineering'.*

4.4. Question-pattern 4 (QP4)

In this grammar structure of the user string the entities have been recognized as representatives of the two classes *TableN* and *ColumnN*, but unlike QP3, in this one have been pointed specific attributes, not all columns in the table. There is also a condition, the implementation of which will lead to the correct result, i.e. an attribute or attributes whose specific numeric or text values are conditions specified in the user request are specified.

Grammatical structure of the user question – [Show | Find | Select | Extract] [all] A1, A2,…, An [from | for | in | of] X, [where | for which when] A1 = 'value_1' [and / or A2 = 'value_2'] [and / or…]

In the pattern Q4, as is shown in the grammar examples in Table 4, X is the only one representative of the class *TableN*, the terms A1, A2, ... etc. are the searched attributes, representatives of class *ColumnN*, and 'value_1', 'value_2',…, etc. are specific values of attributes from the same table. For the pattern under consideration, the verification methods include:

- whether there is another representative of the *TableN* class in the user string;
- whether all found concepts meet the condition to be columns of the searched table.

As a result, only the columns corresponding to the required attributes from the specified table and corresponding to the specified condition are returned.

List of conditions:

- X is a table name

- X is a representative of a class *TableN*
- X is only one representative of a class *TableN*
- A1, A2,… are the names of columns of X, representatives of class *ColumnN*
- C1, C2, C3… are the names of columns of X, representatives of class *ColumnN*
- value_1, value_2,… are specific values from the columns in the table X

Therefore, the pattern can be applied when the analysis detects the presence of only one representative of the *TableN* class and one or more representatives of the *ColumnN* class, in addition to a condition requirement for one of the representatives of the *ColumnN* class.

The corresponding SQL-query is: SELECT A1, A2,…., An FROM X WHERE Ak = ”value”

Table 4. Table describing the grammatical structure of a type QP 4

Grammar structure of the user question	***SQL-query pattern 4***	***Example***
Show | Find | Select | Extract [all] C1 [and |,] [C2 and |,] [A1 and |,]..... [from | for | in | of] X [so] [is true] A1 = 'value1' [and | or |,] [A2 = 'value2']……	*SELECT C1, C2, C3.... FROM X WHERE A1='value_1' [and | or] [A2='value_2'] ……..*	*Extract all emails and phone numbers of students who lives in Sofia.*

4.5. Question-pattern 5 (QP5)

An important option within the select statement is the join operation, which allows the query to gather data from multiple tables [1].

In this pattern, the required language subsets were identified as representatives of the two classes *TableN* and *ColumnN*. Both specific output attributes and specific values set as a condition in the user request are observed. Unlike QP4, here the search is in two different related tables.

Grammatical structure of the user question - [Show | Select | Find | Extract] [all] X.A1, Y.A2 [from | for | in | of] X, [so] [is true] A1 = 'value_1' [and | or] A2 = 'value_2'… and X.PK = Y.FK

In this pattern X and Y are tables, representatives of class *TableN*, A1, A2,…, etc. are the searched attributes, representatives of class *ColumnN*, and 'value_1', 'value_2', etc. are specific numeric or text values of attributes in both tables.

This pattern is characterized by the presence of more than one table. The connection between them is made through the *relatesColToTable* property, which indicates that they are directly connected. For the pattern under consideration, the verification methods include:

- the presence of at least two representatives of the *TableN* class – two interconnected tables in the user string (two adjacent vertices in the subgraph in the database schema);
- all found concepts positively meet the condition to be columns in the respective tables.

As a result, only the columns corresponding to the searched attributes from the specified tables that meet the specified conditions are returned.

A1, A2, A3 – are representatives of the class *ColumnN*, so is fulfilled: *isColumnOf (X) = true*

List of conditions:

- X is a table name
- Y is a table name
- X is not the only representative of a class *TableN*
- Y is not the only representative of a class *TableN*
- C1, C2, C3 – are the names of columns of X, representatives of the class *ColumnN*, so is fulfilled: *isColumnOf (X) = true*
- D1, D2, D3… are the names of columns of X, representatives of class *ColumnN*, so is fulfilled: *isColumnOf (X) = true*
- A1, A2, A3…..are the names of column of X or Y, representatives of class *ColumnN*
- value_1, value_2,… are specific values from the columns in the tables

Therefore, the pattern can be applied when the analysis reveals the presence of at least two representatives of the *TableN* class and one or more representatives of the *ColumnN* class, in addition to a condition requirement for any of the representatives of the *ColumnN* class, as shown in Table 5. As an additional condition to the ones specified in the user string, the relation connecting the two tables must be added by means of a primary key from the first table and a secondary key from the second table.

The SQL-query corresponding to this pattern is: SELECT X.A1, Y.A2,…., X.Ak, Y.An FROM X WHERE X.A1 = 'Value_1' AND / OR Y.A2 = 'Value_2' AND X.PK = Y.FK

FK – Foreign key

PK – Primary key

Table 5. Table describing the grammatical structure of a type QP 5

Grammar structure of the user question	***SQL-query pattern 5***	***Example***
Show | Find | Select | Extract C1 [and |, C2] [of | of all] X [and] D1 [and |, D2]… [of | of all] Y [so] [is true] A1 = 'value_1' [and | or |,] [A2 = 'value_2']	*SELECT X.C1, X.C2,….,Y.D1, Y.D2,….. FROM X WHERE A1='value_1' and A2='value_2' and X.FK = Y.PK*	*Which teachers teach optional disciplines?*

5. CONCLUSION

Combining natural language processing and the proposed method for creating a SQL query to a knowledge base effectively improves the communication process between a person and a computer and is a means by which ordinary users can obtain useful information in their native language. The proposed question-patterns to the database of the system do not limit the user to the choice of words and syntax.

The proposed approach is designed to address the problem of generating SQL queries from freestyle text. We offer grammatical structures and show that they help to find SQL grammar query more easily. The theoretical approach includes creating a

parallel in the functional descriptions of SQL-query and the natural language. Such a parallel allows as an applied aspect to indicate the more effective and faster formalization of linguistic knowledge in the construction of a system with natural language interface to the database.

The contribution of the study is twofold. First, a study of existing approaches to analysis and processing in natural language and second – an algorithm for identifying important elements of a user question to generate a structural query to the database. The work in the research is focused on the methodology for extracting information from different data warehouses in response to user questions.

The proposed methods for processing a query in free text, contribute to the development and usefulness of systems with natural language interface to the database compared to traditional dialog systems, which provide direct answers by accepting parameters from a graphical user interface.

REFERENCES

[1] Wheeler, Jared Thomas. *Extracting a Relational Database Schema from a Document Database*. UNF Graduate Theses and Dissertations. April 2017, 133 p.; Available at: https://digitalcommons.unf.edu/etd/730.

[2] Mony M., Rao J., Potey M., An Overview of NLIDB Approaches and Implementation for Airline Reservation System, *International Journal of Computer Applications*, Vol. 107, No 5, December 2014, pp. 36-41.

[3] Zhekova M., Totkov G., Model of process and model of natural language processing system, *IOP Conference Series: Materials Science and Engineering*, Vol. 878, No. 1, Code 161893, ISSN: 1757-8981, 2020, DOI: 10.1088/1757-899X/878/1/012028.

[4] Zhekova M., Totkov G., Frame Model for Presentation of Semantic Rolls and Processes for the Establishment of Natural language Interface, *Proceedings of the National Conference on Education and Research in the Information Society*, Plovdiv, 2019, pp. 042-052.

[5] Zhekova M., Totkov G., Conceptual frame model for the presentation of the concepts and rules in natural language interface for database, *IOP Conference Series: Materials Science and Engineering*, Vol. 618, No. 1, Code 155190, ISSN: 1757-8981, 2020, DOI: 10.1088/1757-899X/618/1/012035.

[6] Zhekova M., Totkov G., Methodology of SQL queries generator to database set by natural language text, *Scientific Works of the Union of Scientists in Bulgaria, Series C. Techniques and Technologies*, Vol. XVIII, 2019, pp. 98-101.

[7] Popescu A., Etzioni O., Kautz H., Towards a Theory of Natural Language Interfaces to Databases, *Proceedings of the 8th International Conference on Intelligent User Interfaces*, Miami, USA, 2003, pp. 149-157.

[8] Giordani A., Moschitti A., Translating questions to SQL queries with generative parsers discriminatively reranked, in *COLING*, 2012, pp. 401 – 410.

[9] Yaghmazadeh N., Wang Y., Dillig I., Dillig T., Sqlizer: Query synthesis from natural language, *International Conference on Object-Oriented Programming, Systems, Languages and Applications*, ACM 2017, pp. 63:1 – 63:26.

[10] Iyer S., Konstas I., Cheung A., Krishnamurthy J., Zettlemoyer L., Learning a neural semantic parser from user feedback, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, Vancouver, Canada 2017.

[11] Xu B., Cai R., Yang X., Zhang Zh., Li Z., An encoder-decoder framework translating natural language to database queries. arXiv:1711.06061v2 [cs.CL], 9 Jun 2018.

[12] Dar, H., Lali, M., Din, M.U., Malik, K., Bukhari, S. Frameworks for Querying Databases Using Natural Language: A Literature Review, *International Journal of Data Warehousing and Mining (IJDWM)*, IGI Global, vol. 17, No. 2, April 2019, pp. 21-38, ArXiv, abs/1909.01822.

[13] Xiaojun Xu, Chang Liu, Dawn Song. SQLNet: Generating Structured Queries from Natural Language Without Reinforcement Learning, preprint arXiv:1711.04436, 2017

[14] Zhong V., Xiong C., Socher R., Seq2sql: Generating structured queries from natural language using reinforcement learning. *CoRR journal*, 2017, arXiv:1709.00103.

[15] Sethi G., Singla S., Role of natural language processing in education. *International Journal of Scientific Research and Education*, Vol 4, No 3, 2016.

Information about the authors:

Mariya Zhekova – PhD student who currently works in the Department of Computer Systems and Technologies in University Of Food Technology - Plovdiv as an assistant. Research programming languages, Databases and Artificial Intelligence related to natural language comprehension. Developing a model in which syntactic information and operations that were previously thought to be related to lexical entries are replaced by a syntactic-semantic functional structure.

George Totkov – Professor at Plovdiv University, who works with e-learning standards, Big data in economics, Digital maturity models. Teaching activity - conducting lectures in the field of informatics and information technologies in different universities. Research in the field of informatics and its applications (computer linguistics, e-learning and distance learning environments). Management and participation in international, national and university projects. Guide for graduates (more than 130) and PhD students (successfully defended 10). Management of teams for providing methodological and technological support of distance and e-learning in higher education.

Manuscript received on 06 April 2021