EAI Endorsed Transactions
on Scalable Information Systems

Research Article EAI.EU

A Naive approach: Translation of Natural Language to Structured Query Language

Jyotirmayee Rautaray[1,*] and Pranati Mishra[1]

[1]Odisha University of Technology and Research, Bhubaneswar, Odisha, India

Abstract

A database is a major source of information which plays an important role in our life. Information retrieval from the database requires formulating a querythatisunderstandablebythecomputerinordertoproducedesiredoutput. Generally, databases work with structured query language (SQL). But a naive user usually unfamiliar with the structured query language as well as structure of the table in the database. Hence, it becomes very difficult for the naïve-user to collect the desired information. This paper provides a solution to this problem and it enables users to retrieve information through natural language, such as English language. Being able to access information from the database by using natural language bridges the man-machine gap. Tokenization, lexical analysis, syntactic analysis, semantic analysis, and other complex stages are all involved in converting a natural language query into a SQL query. The purpose of this paper is to translate natural language queries into Structure Query Language queries, allowing non-technical people to get connected to databases and to gather the required information.

Keywords: Natural language query, Natural language processing, Natural language generation, SQL query, Syntactic analysis, Semantic analysis, Data dictionary, Natural language information database

Received on 07 August 2023, accepted on 30 October 2023, published on 09 November 2023

doi: 10.4108/eetsis.4344

*Corresponding Author. Email: jrautraycse@outr.ac.in

1. Introduction

The topic Natural language Processing (NLP) is a part of linguistics and computer science which uses machine translation from one natural language to another natural language, information retrieval, and language analysis. Accessing the database through a natural language interpreter would make it easier for naive users. It is a procedure in which software is used to automatically manipulate natural language in the form of speech and text. [22] NLP creates a smooth way through which computer and human being can interact with each other effectively [2][6]. Machines cannot easily learn natural language because the human mind is significantly more intelligent than a machine. NLP gives machines the capacity to read, comprehend, and generate accurate results. NLP has two components: NLU (natural language understanding) and NLG (natural language generation). We provide a text or audio signal as input, and the software converts the input into another form of text or audio signal so that NLU can grasp the language and create a valid output [8].

The storage, management, processing, and retrieval of data are made easier by data base management systems (DBMS). Data are expressed in the form of tuples in relational databases and a user needs to know both DBMS and structured query language (SQL) to access the database [7]. Therefore, to access the information from databases some basic training is needed about SQL and DBMS. Furthermore, the system can be inquired by a user utilizing natural language, such as English and that natural language is converted into structured query language which access the DBMS; finally, the translated system returns the result back to the end user with the natural language [9][4]. Particularly for inexperienced users who are unfamiliar with complicated query languages like SQL, natural language queries are a very convenient and straightforward approach to access databases. This particular machine translation system focuses on solving problems with syntactic and semantic analysis and dictionary compilation that arise while analyzing or creating natural language speech or text [7] [11]. The lexicon, a table used to generate a map the standard natural input terms to the formal objects in the database (relationships, names of

attributes, etc.), will be used during the translation process. The lexicon is used by both the parser and the semantic interpreter [6]. The formal answer is taken as an input by a natural language generator and inspected the parse tree to produce an acceptable natural language response [13]. The syntactic information and understanding of the existing database are used by natural language database systems to correctly relate the structure and content of the natural linguistic input to this existing database table. Syntactic knowledge often exists within the program's linguistic component, especially in syntax analyzer, although actual database information is included in the semantic data model [10]. Natural language inquiries turned into structured query language arguments. The database management system will execute the query to produce the necessary data once the assertion has been stated clearly [12]. All the data were again transferred to the natural language section, in which generation procedures produced a version of surface language of the solution [12]. This proposes the architecture to translate English Query into SQL. This architecture is demonstrated as four forms like:

1.1 Pattern Matching Systems

This model-based approach's key benefit is flexibility: there is no need for intricate parsing and decoding modules, and it is simple to create the systems [8] [13]. Additionally, when input is outside the range of phrases that need to be addressed by pat- terns, pattern matching systems frequently offer a logical solution.

1.2 Syntax Based Systems

The user's request is searched in syntax systems (i.e., syntactically analyzed) and a phrase in certain query languages of a database is mapped directly to the parser. Based on lunar syntax, NLIDBS generally provides access to different database systems that offer carefully crafted database queries for mapping from the parser to the database query. A common instance of this method is the NLIDBS that maps rules which directly translates the parsing tree into some expression that is difficult to develop realistic database query language [11] [14].

Semantic Grammar Frameworks

Semantic grammar frameworks continue to parse the question and answer utilizing the source input and map from the parse tree input to a database query language. This situation is uncommon in that syntactic concepts might not always correspond to grammatical categories (i.e., non-leaf nodes in the parse tree) [4] [7]. As an engineering tool, semantic grammars have been added, that becomes easy to integrate semantic knowledge into the framework; because semantic grammar requires knowledge of a specific domain. When NLIDB is designed for the new knowledge domain, it is highly challenging to adapt systems built using this methodology to other domains of knowledge.

Languages for intermediate representation

Majority of the new NLIDBS turn each natural language query into an intermediate logical question in an internal representation language. Independent of the database layout, the abstract intermediate query communicates its importance of the user's problem in regard to high-level world concepts. The mid-term representation approach [8] allows for the division of the system into two sections. One section begins with a sentence until the logical query is produced. A creation of a database query follows logical query that begins the second section. First part's usage of logic queries enables language to associate the technique with logic by enclosing the reasoning portion in the logic sentence [1]. However, as the logical query language is distinct from a standard database. Various database query languages can have used it. Moreover, due to the language's implicit ambiguity natural language is a topic of theoretical interest. Many scientists have been using different language strategies different purposes. Different approaches to language processing can be essentially divided into two categories: empirical strategies or corpus-based strategies and symbolic strategies or rule-based strategies. In the natural language the words are considered as symbols which represents the real-world objects and ideas [6] [14]. Furthermore, the words are organized into phrases, that complies with well-designed grammar rules. The symbolic approach has therefore dominated work in natural language processing for several decades. Natural language processing seems to be an endeavor that seems strongly symbolic. Knowledge of language is specifically embedded in rules or other means of representation. In order to obtain knowledge, language is analyzed at different levels. Some rules are applied to this information obtained in order to get linguistic functionality; because human communication language skills involve rule-based reasoning, symbolic processing is well supported. For each stage of linguistic analysis, rules are created in symbolic processing. On the basis of these criteria, it makes an effort to find out the meaning of the basic language. Empirical techniques are enriched with statistical analysis of text corpora, which are raw data in the text form. A corpus is a group of texts, which can be typically stored in a computer database and easily readable by computer. The corpora are mainly used as a language source [8]. The empirical approach has been existing since the start of the NLP (e.g., early 1950s); but in the last 10 years, it has emerged as a significant alternative to rule-based NLP [9]. The corpus-based approach is another name for the empirical approach. There are numerous approaches that have been developed for corpus data analysis.

The demand for and use of MT are now expanding quickly. Machine translation (MT) converts text from the source input language to the target language as output. In order to its citizens for easily exchange and to share information and also knowledge, India's multilingual society—which has a population of approximately 1.37 billion people, 22 officially

categories languages, over 30 regional languages, and more than 2000 dialects—needs well-developed ICT (Information and Communication Technology) tools [16]. Hence it can help to reduce the language barrier and enable easier communication. Though a number of MT systems like Google Translate, Systran, Bablefish have already been developed, still it needs to be developed for other regional languages [19] [20]. During the translation of one source language to other target language, doing the target language syntactically and semantically (ambiguity free) correct is a more challenging and tedious task. Translation of a standard natural language into structured query language (SQL) is now made possible via the NLIDB (Natural Language Information Data Base) [18].

The remaining parts of the paper are organized as follows: Section 2 defines relevant research work, Section 3 proposes a suggested framework, and Section 4 suggests an algorithm. Section 4 defines the conclusion.

2. Related Work

Ahmad et al. [2] have suggested Urdu language NLIDBs. An algorithm is created that effectively converts an Urdu-language query into a SQL statement. PRECISE is the foundation of the system is formal semantics. Three essential tables make up the semantic knowledge base, where we store the semantic data for a database that the mapping algorithm will employ. With the resolution of attribute names and proper binding of attributes and values, Mapping Algorithm converts the request into an intermediate form.

Alexander et al. [16] introduced the NLIDB, allowing the user to access the server through a simple internet interface in a language that is more like English. To make a bridge of the gap between computer and human communication, a natural language Web Interface for Database (NLWIDB) is being developed. This paper also explains how we may provide rich tools for non-experts to construct sophisticated queries while avoiding a structured query language which can be handled fairly by specialists only.

For mapping questions to SQL queries, Dollak et al [4] have suggested data sets. First of all, questions created by humans and compared with automatically generated questions, properties of queries required for real applications. Furthermore, they show that the current division of information into learning and tests evaluates the robustness of the problems, but only partly check show well new queries generalize systems. This paper also discusses lot fitting based on models, which competes with the previous work on several data-sets. This paper offers splitting based on the SQL query to test robustness in the current SQL queries.

Kolhe et al. [19] explains that the main objective of this research article is to explore the optimize way to translate natural language query to SQL and prepare the data suitable for the semantic extraction. It also provides a user-friendly interface between the end-user and the data base for easy access to social web data from various web sources like Facebook, twitter etc. Suggested framework created by this research study can run DDL and DML queries and accept user input in natural language.

Ghosh et al. [13] developed a process that can translates query of natural language to SQL using speech to android text API. The applications of natural language processing include data extraction and organizing of information, machine translation. ‘Android speech’ package and a class within it called ‘android speech’. Recognizer Intent’ are the key component of the Speech to Text Android API. Here by the purpose of the name ‘android speech. Recognizer Intent’ is enabled, which displays a dialog box that enables the input to be understood in speech. Using an SQL database does not require a database set up procedure or management in android. Currently, this system is able to handle simple queries with some complex queries.

Authors Agrawal et al [15] proposed a system “Intelligent Querying System” (DBIQS) representing a fully automated, quick and secure way of querying a database DBIQS encompasses one of the key developments in the form of Automatic Semantic Map Generator, which significantly reduces user work in the manual development of the semantic map. Additionally, it offers an Intelligent Engine that automatically resolves ambiguity problems in the discourse and pragmatic analysis of a multi-relational query. It excels at turning intractable requests into partially or fully tractable ones and it is particularly effective at producing SQL queries from tractable natural language questions. Users can easily show information in the form of tables, bar charts and pie charts, as well as develop custom views.

Mandhan et al. [12] defined the structural design approach used by the Data Dictionary given in a database to translate English Query into SQL. This paper describes an interface in a natural language database, which lines complex queries to the relation database using a probabilistic context free grammar (PCFG). First some popular NLDBI structures are summarized. Take a server into account, claim DB. Several tables are inserted with properly normalized attributes into this DB database. The system removes this aspect and allows the end user in their own language to use the tables.

Kaur et al [18] proposed a module interface that transforms the user query into a corresponding SQL command in natural language. We find an ORACLE database in this frame work and use a default table that is properly normalized. The program is built in the language of JAVA. The system's Data Dictionary must be regularly updated with relevant system-specific words.

Mathur et al [11] proposed the architecture to use Semantic Grammar with added Voice Recognition feature to convert input sentence(s) into SQL query. SQL's "Natural Language Query Processing" framework can enable the automated generation of SQL queries for both users and software engineers. The novel user's job will be improved by providing an easy interface that will become more familiar and user-friendly.

Manasamithra et al [10] introduced a hybrid system to use natural language to search and retrieve data from the database. It is a mixture of approaches of keyword and semantic analysis. M-way B-tree is used to store keywords that serve as a base of knowledge. The system's performance

will be improved by using more reliable data structure to store keywords. All keyword combinations are stored with a root node and all the leaves in alphabetical order in different m-way B-trees to make searching easy. Separate B-tree is built for table names, attributes, constants, and so on.

Singh et al [20] created a smart interface using asemantic matching technique that converts natural language query to SQL using a collection of production rules and data dictionary. To convert Natural Language Query (NLQ) to SQL Query, a set of steps such as lower-case conversion, speech tagging, tokenization and extraction of SQL elements are used. The proposed system is designed to reduce a human-computer communication gap. The NLTSQLC system is designed in this research to address challenges in the processing of NLQ. The objective is to assess correct NLQ SQL translations.

Sontakkeet al [21] have implemented a rule-based system that will fulfil the user's need and will recognize Hindi as a request and will only give out in Hindi. They developed a system that can manage Hindi query to retrieve single or multiple columns from tables stored in databases and also worked on independent query's semantic behavior. Once the request has been fired, translation time and execution time are also determined. There is also the technique of randomizing automatic record generation so that we can easily produce maximum number of records in very less time.

Kovács et al [8] proposed an interface module that would translate the request provided by the user into a corresponding SQL command in natural language. A push down automaton is used to test the syntax after clustering the input sentence. A se- mantic matcher module produces the corresponding SQL code. Some findings have also been reported on the design of an NLP engine to turn natural language sentences into SQL commands. The approach's innovation concerns the combination of the following characteristics: Hungarian language processing, multi-level command generation levels, and similarity-based sentence processing. For a predefined application domain, the created system provides a versatile and efficient command generation.

3. Proposed Framework

The first aspect is that the user needs to sign in. The user might query, such as ‘get...’, ‘search...’, ‘I need...’, ‘'put…’. You may submit a query in a text format. The system also offers the ability to change and delete database tables. It will also be possible for the user to insert data into the database. The user question is slowly analyzed.

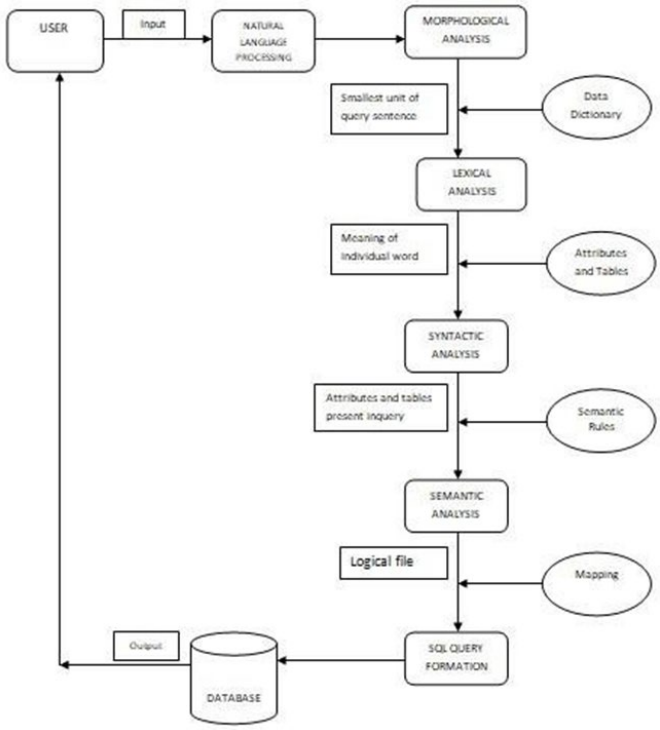

Fig.1. Proposed System Architecture for conversion from Natural Language to SQL

All the levels shown in Fig.1 are explained below:

3.1 Morphology

At this point, the expression is divided in the smallest sense unit. At this level, the input query sentence is divided into all words it contains in a natural language and stored in a list. For example, if the user gives a query such as “Find the cost of a Dell laptop”, then in this phase, each word present in the sentence i.e., find, the, cost, of, a, Dell, laptop will be stored in a list like [‘find’, ‘the’, ‘cost’, ‘of’, ‘a’, ‘Dell’, ‘laptop’].

3.2 Lexical

Humans and NLP programs understand the context of individual words at this point. Each word of the tokenized expression is mapped to the same term in the data dictionary.

For example, the list created in the morphological phase will be mapped as:

-find: select
-cost: price
-laptop: laptop

3.3 Syntatic

At this step, we can define the attributes of the word produced in the lexical stage in the provided input query. Each of them is verified with the dictionary attributes that include all the tables and their attributes. After that tables which contain those attributes are searched. For example, the output generated in the previous phase, we derive attributes as price and it belongs to the table laptop.

3.4 Semantic

Semantics focuses on the analysis of the meaning of the words found in the natural language query and the relationship between objects such as words signs phrases etc. Linguistic semantics, deals with the study of meaning that interprets human expression by words. This stage discusses the Terms such as clause, relational operators, aggregate functions, natural join and builds the SQL query accordingly.

The query (SQL) created at the end is after checking all the conditions is select price from laptop.

In the proposed system architecture, the user provides a natural language query to the system which is fed to the morphological analysis phase directly. Morphological analysisbreaksthesentenceintosmallerwordsorsmallestunitof meaningandstores them in a list. That list again sent to the next phase which is lexical analysis. In lexical analysis the words are mapped onto the data dictionary to find the meaning of the individual words. Semantic analysis determines the relationship among the attributes and tables present in the given query. Semantic analysis generates a logical file which can be mapped on to the database. The SQL query when performs operations on the database, the user can get his/her required information. This system can be added with speech recognition as well. By using Google API as an interface in python programming language the system will be able to process Speech as an input as well. While using speech recognition it must be noted that the system will convert the speech in to text in the first step and then the process will continue as usual.

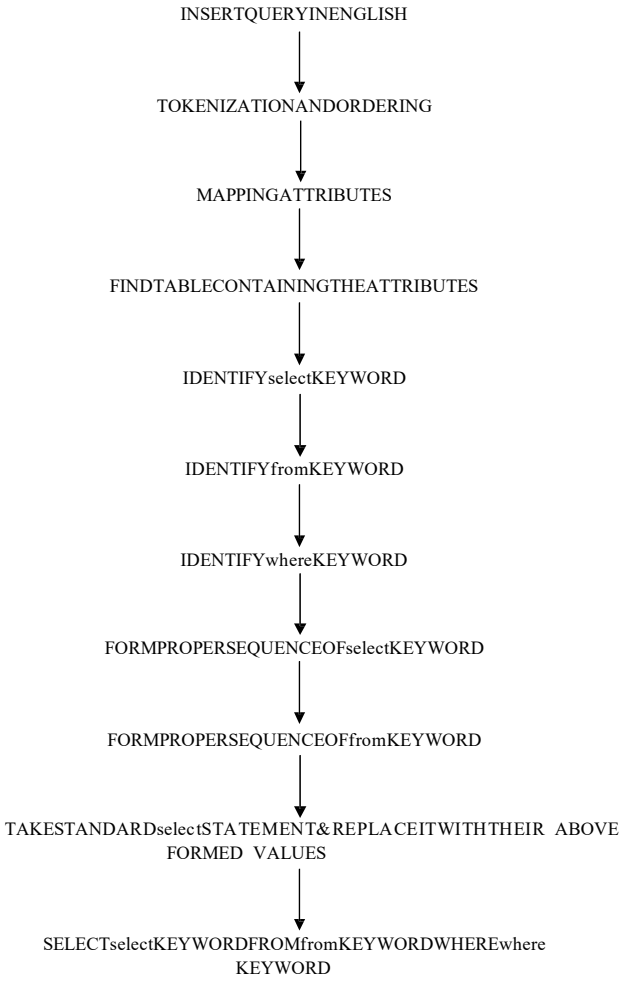

Fig.2. Workflow diagram for transformation from natural language to SQL

The workflow diagram described in Fig. 2 explains the construction of SQL query from the User's provided natural language.

Algorithm

Step1-Store the natural language query, given by the user, as a string.
For example: Find the students whose marks in maths is greater than 80, marks in science are > 70 and aggregate is > 60 in descending order.

Step2- By taking into account the gaps in the text, the sentence is divided into individual words (tokens) and kept in another list. All the tokens are used for the above query.

Step3- All tokens are matched with the separated ignore list, only important words are kept and words that appear in the ignore list are simply removed.

Ignore list: a, an, is, are, for, to, of, in, mark, than etc.

Step4-A data dictionary is used to map the tokens with their synonyms and word type (attributename / tablename, keyword, attributevalue) is added to it.

SELECT <action>,*<keyword>, Where<clause>, <value>, AND, <attribute_name>, <value>, AND, <attribute_name>, 60<value>, ORDER BY<clause>

Step5-Than syntax for the query statement is decided with respect to the word type.

- Attribute_name word types are associated to the SELECT keyword.
- Table _name word types are associated to the FROM keyword.
- SELECT <action>* FROM <table_name>WHERE<clause> <attribute_name> <value>AND <attribute_name><value>AND <attribute_name> <value>ORDER BY<clause>

Step 6-Finding the condition and value in the input query and map them to the data dictionary.

SELECT <action> * FROM WHERE <clause> <attribute_name> <value> AND <attribute_name> <value>AND <attribute_name> <value> ORDER BY <clause>

Step 7-The final query is generated and executed on the Database to get the final output.

SELECT * FROM student WHERE maths > 80 AND science > 70 AND aggregate > 60 ORDER BY name DESC.

4. Results

- At this point, the expression is divided in the smallest sense unit. At this level, the input query sentence is divided into all words it contains in a natural language and stored in a list. For example, if the user gives a query such as "Find the cost of a Dell laptop", then in this phase, each word present in the sentence i.e., find, the, cost, of, a, Dell, laptop will be stored in a list like ['find', 'the', 'cost', 'of', 'a', 'Dell', 'laptop'].

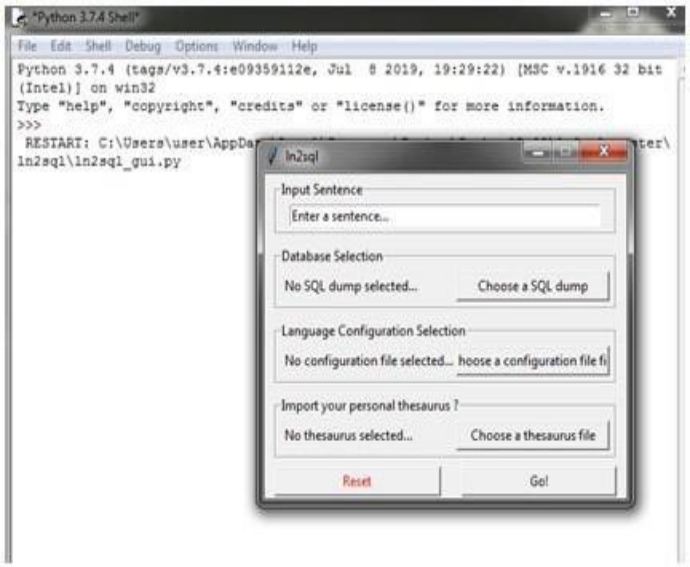

Fig.3.GUI

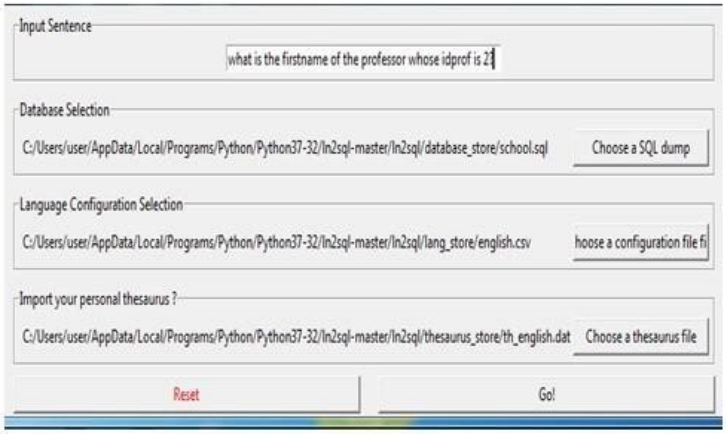

Fig.4. Query given "what is the first name of the professor whose id proof is 2?"

```
SELECT professor.firstname
FROM professor;
```

Fig.5. Result of the query given in Fig4

5. Conclusion

Natural Language Processing is an extremely powerful tool that can alter the user interface's full function. Now, this system can handle easy inquiries together with some difficult questions. Research and implementation were relatively new to the NLP. Our approach to the problem of generating a robust NLI for the data base is both realistic and theoretical. Such interfaces are now more and more relevant. With- out a single SQL insert statements on the top of processed data, we store intermediate data and introduce new information. The architecture is best suited for naturally writ- ten extraction. This is a much quicker approach.

References

[1] Abhilasha Kate, Satish Kamble, Aishwarya Bodkhe, and Mrunal Joshi. “Conversion of natural language query to sql query”. In 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA), pages 488-491. IEEE, 2018.

[2] Ahmad, Rashid, Mohammad Abid Khan, and Rahman Ali. "Efficient transformation of a natural language query to SQL for Urdu." In Proceedings of the Conference on Language & Technology, p. p53. 2009.

[3] AJ Shaikh and VL Kolhe. “Framework for web content mining using semantic search and natural language queries”. In 2013 IEEE International Conference on Computational Intel- ligence and Computing Research, pages 1-5. IEEE, 2013.

[4] Catherine Finegan-Dollak, Jonathan K Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. “Improving text-to-sql evaluation methodol- ogy.” arXiv preprint arXiv:1806.09029, 2018.

[5] Chalermpol Tapsai. “Information processing and retrieval from csv file by natural lan- guage.” In 2018 IEEE 3rd International Conference on Communication and Information Systems (ICCIS), pages 212-216. IEEE, 2018.

[6] Ion Androutsopoulos, Graeme D Ritchie, and Peter Thanisch.” Natural language interfaces to databases-an introduction.” Natural language engineering, 1(1):29-81, 1995.

[7] Josephine E Petralba. “An extracted database content from wordnet for natural language processing and word games.” In 2014 International Conference on Asian Language Processing (IALP), pages 199-202. IEEE, 2014.

[8] Kovács, László. "SQL generation for natural language interface." Journal of Computer Science and Control Systems 2, no. 18 (2009): 19-22.

[9] Kumar, Sachin, Ashish Kumar, Pinaki Mitra, and Girish Sundaram. "System and methods for converting speech to SQL." arXiv preprint arXiv:1308.3106 (2013).

[10] Manasamithra, P., and H. C. Vijayalakshmi. "NLP for Information Retrieval using B Trees." International Journal of Computer Applications 975: 8887.

[11] Mathur, Amrita, Atmiya Fadia, Bhagyashri Lokhande, Sagar Malaviya, and Seaman Mu- nis. "Database access for non-technical users using nlp and voice recognition."4

[12] Nikita Mandhan, Aishwarya Ahuja, Komal Jain, and HD Gadade.” Q&a on database using nlp.”

[13] Prasun Kanti Ghosh, Saparja Dey, and Subhabrata Sengupta. “Automatic sql query for- mation from natural language query”. International Journal of Computer Applications, 975:8887, 2014.

[14] Rashid Ahmad, Mohammad Abid Khan, and Rahman Ali. “Efficient transformation of a natural language query to sql for urdu”. In Proceedings of the Conference on Language & Technology, page p53, 2009.

[15] Rohit Agrawal, Amogh Chakkarwar, Prateek Choudhary, Usha A Jogalekar, and Deepa H Kulkarni. “Dbiqs an intelligent system for querying and mining databases using nlp”. In 2014 International Conference on Information Systems and Computer Networks (ISCON), pages 39-44. IEEE, 2014.

[16] Rukshan Alexander, Prashanthi Rukshan, and Sinnathamby Mahesan. “Natural language web interface for database (nlwidb)”. arXiv preprint arXiv:1308.3830, 2013.

[17] Ruslan Posevkin and Igor Bessmertny. “Translation of natural language queries to struc- tured data sources.” In 2015 9th International Conference on Application of Information and Communication Technologies (AICT), pages 57-59. IEEE, 2015.

[18] Saravjeet Kaur and Rashmeet Singh Bali. “Sql generation and execution from natural lan- guage processing”. International Journal of Computing & Business Research ISSN (Online), pages 2229-6166, 2012.

[19] Satwik Kolhe and Nitish Muley. “Natural language processing based sql query generation for retrieval of data from temporal/non-temporal database.”

[20] Singh, Garima, and Arun Solanki. "An algorithm to transform natural language into SQL queries for relational databases." Selforganizology 3, no. 3 (2016): 100-116.

[21] Sontakke, Abhijeet R., and Amit Pimpalkar. "A Review Paper on Hindi Language Graph- ical User Interface to Relational Database using NLP." International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) 3, no. 10 (2014).

[22] Rautaray, Jyotirmayee, Asutosh Hota, and Sai Sankar Gochhayat. "A shallow parser-based Hindi to Odia machine translation system." In Computational Intelligence in Data Mining: Proceedings of the International Conference on CIDM 2017, pp. 51-62. Springer Singapore, 2019.