CONVERSION OF SPOKEN ENGLISH LANGUAGE QUERY INTO SQL

[1]Mrs. Shruthi J., [2]Ankita Anand, [3]Ankit Agrawal, [4]Bhaskar Todi
[1]Assistant Professor, [2]Student, [3]Student, [4]Student
[1]Department Of Computer Science & Engineering,
[1]B.M.S. Institute of Technology & Management, Bengaluru, India

Abstract: Databases are very common and used at a large scale within current applications and websites. Data retrieval from database requires knowledge of database language. But they are often used by people who do not have much competence in the field and not knowing their structure rigorously. Researchers have been trying to find a method that bridges the gap between information need and users' satisfaction. In this article, we propose a method aimed to allow user to access data using natural English language. The user submits the query in the form of speech signal. A language model converts the speech utterance into English sentence. The input sentence is then parsed and natural language Expressions are converted into SQL language. The generated SQL query is then used to retrieve information from database and the result is displayed in the form of text.

Index Terms **– Conversion of Spoken English into SQL Query, Natural Language Processing**

I. INTRODUCTION

Database access has been increasing rapidly these years, as every business element use database for data storing. Nowadays there are too many data which maintain in organizations, companies and universities databases, but only few pf the individuals who use these data are familiar with data query methods. The Internet has gradually become more popular and popularized, but the databases as for them still remain abstract for many people. Some positions do not require any computer training or in data administration but still need to work closely with databases, as in accounting or secretarial for example. A person, having no competence in the field management system, should at least understand how it works, interact with it and perform simple tasks on it (query, add, delete). This article aims at designing a translator allowing the interrogation of any database in English. We will describe how to extract information relating to a target database in order to know its structure and vocabulary, then to cross this information with the key words of the question asked, and thus generate as output the most probable equivalent SQL query. The request will be generated based on the presence, number and order of the keywords identified in the sentence entered by the user. Finally, we will present the evaluation of our approach, comparing the capabilities of our application to those of existing applications and evaluating its performance on a set of test queries.

II. LITERATURE SURVEY

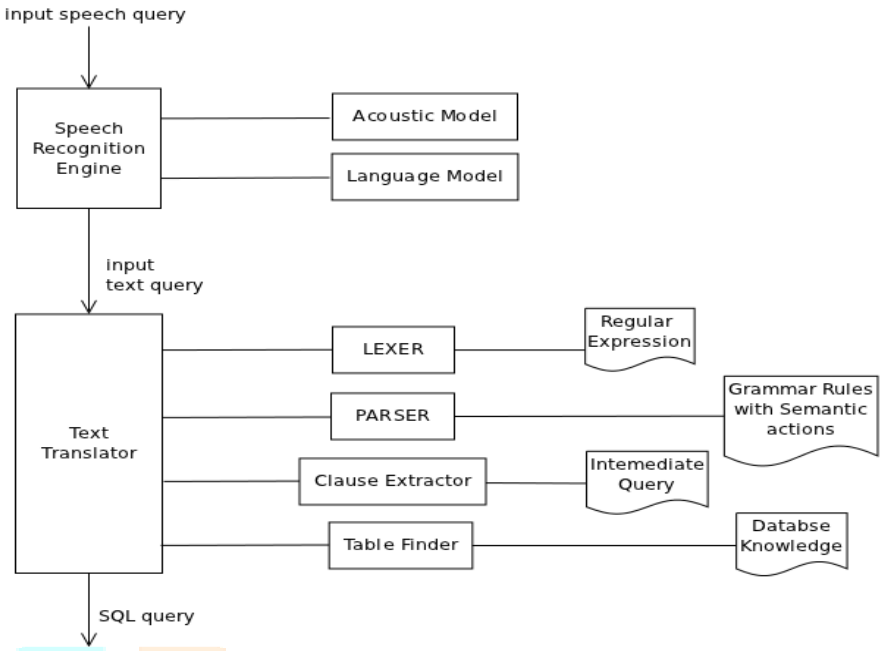
A huge amount of labor is required if we wish to obtain only the required information from the entire repository of any information system. Asking questions to databases in natural language is a very convenient and easy method of data access, especially for casual users who do not understand complicated database query languages such as SQL

In English, question sentence can be classified based on the grammatical form into 4 categories:

1. Yes-no question, in which principle can be answered with a simple "yes" or "no", e.g. "Is anyone home?"
2. Wh-question, which uses interrogative words (wh-words) such as what, who, when, etc., e.g. "Who are the employees?"
3. Choice question, which has two or more possible answers which are usually included in the question, e.g. "Are you supporting England, Italy, or Germany?"
4. Tag question, which is a declarative statement turned into a question by adding an interrogative fragment, e.g. "You are coming here, aren't you?"

Similarly, there are four types of user queries: 1) Yes/No Query, 2) Wh-Query, 3) Command Query and 4) Hybrid Query.
When a user speaks a sentence of the type:
How old are the students whose first name is Jean?
The application must execute a query like this:
SELECT age FROM student WHERE firstname = 'JEAN'
The passage from this first sentence to the second represents the whole problematic of the project.

The architecture of a model that coverts speech into text and then into SQL has following components:

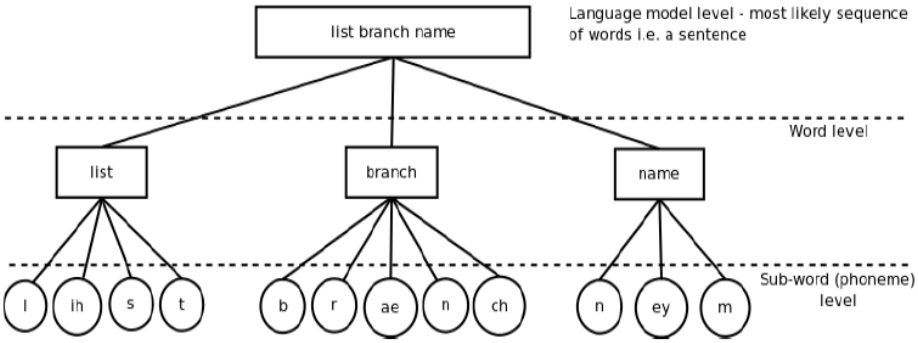


- Speech Recognizer:

Speech recognizer is used to convert a human spoken English sentence into a text string. Speech recognizer relies on acoustic and language models to convert a spoken sentence into text.

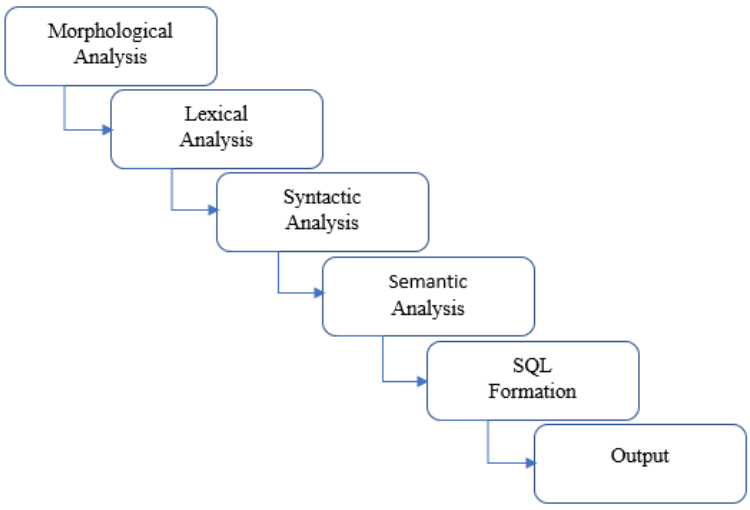
- Acoustic Model:

An acoustic model is used in automatic speech recognition to represent the relationship between an audio signal and the phonemes or other linguistic units that make up speech. The model is learned from a set of audio recordings and their corresponding transcripts. It is created by taking audio recordings of speech, and their text transcriptions, and using software to create statistical representations of the sounds that make up each word.



- Language Component:

Language component uses classical technique i.e. grammar-based technique for processing the English Text query which contains following components



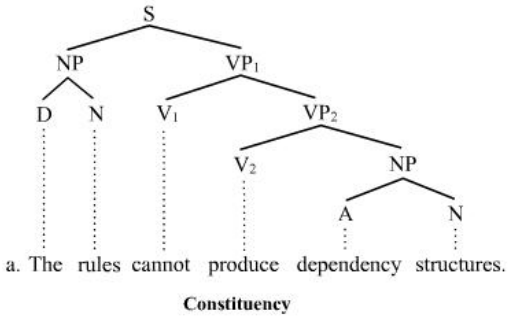
Components of Language Model

a. Morphological Analysis: Morphological analysis represents the properties of word forms with their structural component and its smallest unit of meaning.

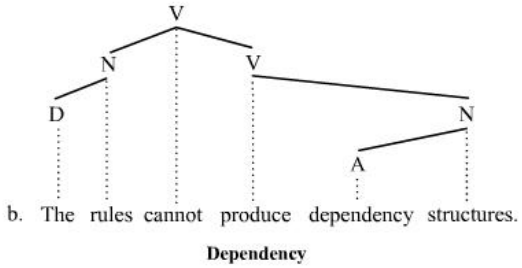
b. Lexical Analysis: Lexical Analysis is defined for relating words to create the vocabulary of language as distinguished from its grammar and construction. Lexical Analyzer also generates the tokens from the sequence of input words.

c. Syntactic Analysis: Syntactic Analysis or Parsing is the process of analyzing a string of symbols in natural languages according to the rules of formal grammar. Syntax parser converts the natural language query into syntax tree, according to lexicon and syntax grammar.

Parsing is done based on the type of grammar: Dependency grammar or constituency grammar



Constituency



Dependency

- A constituency parse tree breaks a text into sub-phrases. Non-terminals in the tree are types of phrases, the terminals are the words in the sentence, and the edges are unlabeled.
- A dependency parse connects words according to their relationships. Each vertex in the tree represents a word, child nodes are words that are dependent on the parent, and edges are labeled by the relationship.

d. Semantic Analysis: Semantic analysis focuses on the study of meaning of the words present in the natural language query and what do they actually stand for. This level deals with checking the different conditions like WHERE clause, relational operators, aggregate functions, natural JOIN and build the SQL query accordingly. The semantic database helps us to present an equal query for different sentences.

e. SQL Formation: After forming all join conditions these conditions are appended in WHERE clause. Thus, the final SQL query is generated which is used to retrieve the required information from the database.

III. RELATED WORK

- One of the first problems with querying a database by a user with no knowledge in this area is that he knows neither the structure nor the vocabulary employed within the base he seeks to question. The most trivial solutions consist either in limiting the vocabulary he can use, or in using a strict grammar, which with rules, will limit the sentences that he can build.
- Some works force the user to enter a question formatted according to a dictionary of words specific well known by the application, therefore limited to a particular base.
- LUNAR (Woods, 1973) involved a system that answered questions about rock samples brought back from the moon. Two databases were used, the chemical analyses and the literature references. The program used an Augmented Transition Network (ATN) parser and Woods' Procedural Semantics. The system was informally demonstrated at the Second Annual Lunar Science Conference in 1971.
- LIFER/LADDER (Hendrix, 1978) was one of the first good database NLP systems. It was designed as a natural language interface to a database of information about US Navy ships. This system, as described in a paper by Hendrix, used a semantic grammar to parse questions and query a distributed database. The LIFER/LADDER system could only support simple one-table queries or multiple table queries with easy join conditions
- QUESTION-ANSWERING SYSTEM (Nguyen Tuan Dang, 2009) proposed a method to build a specific Question-Answering system which is integrated with a search system for e-Books in the library. Users can use simple English questions for searching the library with information about the needed e-Books, such as title, author, language, category, publisher… In this research project, the main focus is on fundamental problems in the natural language query processing: approaches of syntax analysis and syntax representation, semantic representation, transformation rules for syntax structure of semantic structure.

IV. UNDERSTANDING THE SYSTEM

The main characteristics of this System are that it provides an interface for users to query the database through speech signal. The system can be configured automatically for different databases. Users don't need to know the underlying database schema to query. System automatically finds the table names required to satisfy the query. The System uses rule-based translation technique which comprises lexical analyzer, Syntax analyzer and Syntax directed translation.

- First step is to record the speech utterance of the user and convert it into text. An algorithm used by Google Speech API has been used for this purpose.
  In this model the default value used for continuous is ‘false’, meaning that when the user stops talking, speech recognition will end. This mode is great for simple text like short input fields. The only results returned by the recognizer are final and will not change. The spoken language for the speech recognizer "lang" is set via the selection drop-down list, for example “en-US” for English-United States. If this is not set, it defaults to the lang of the HTML document root element and hierarchy. After setting

the language the speech recognizer is activated. Once it begins capturing audio, it calls the on-start event handler, and then for each new set of results, it calls the on-result event handler. This handler concatenates all the results received so far into two strings. The resulting strings may include "\n", such as when the user speaks “new paragraph”. First String is a local variable and is completely rebuilt each time this event is called because it’s possible that all interim results have changed since the last on-result event but final text never changes.

- Second step is to recover only the meaningful words in the sentence entered by the user. Indeed, it is important to be able to subsequently perform a correspondence between certain concepts entered by the user and elements of the database. To do this, the Tree Tagger tool has been used to filter the empty words according to their class grammatical (prepositions, pronouns, determiners, etc.) and to perform rooting of the remaining words. Consider the sentence:

  How old are the students whose first name is Ankit?

  The filter must return the elements: "age, pupil, first name, Ankit".

- The Third step of the process is to recover the structure of the database on which the query has to be performed in order to know the entities (columns, tables, primary and secondary keys, etc.) to allow in a second time a correspondence with the words extracted from the user request.

  Two methods have been implemented to achieve this result. The first method is to collect the information by querying the database using SQL queries like "SHOW TABLES, SHOW COLUMNS, DESCRIBE, etc. ".

  The second method is to analyze the backup or database creation file.

  If the user does not enter a correctly spelled sentence or whose vocabulary used is not strictly the same as that of the database, no correspondence between his sentence and elements of the base will be found and no relevant result will be returned. So, it is important to maximize the number of words that will result in a relevant match between a keyword of the input request and an element of the database. To do this, a dictionary is loaded. For each word, table of concepts containing all the words of the language by which it can be substituted can be accessed. The purpose of this translator is to make the query of a database accessible to a person who does not know either structure and keywords and therefore being able to use a synonym for a word used in the base instead of the word itself. It is therefore more judicious to represent a word by a concept, a table of all the words by which it can be substituted rather than only by itself.

- At the next step each key word of the request entered by the user is extracted. The idea now is to look for a correspondence between the key words of the application or their synonyms and the entities of the database in order to perform a segmentation of demand according to the correspondences found and thus to know at best the structure of the request to generate. Each keyword found is tagged according to whether it is a column or a table the database being queried, or something else still unknown at the moment. The presence of a SELECT and FROM segment is mandatory in the sentence, the first designates which will be the type of selection and on which element(s) exactly, the second specifies where to look, in which table(s), the element of the selection. The JOIN and WHERE segments are optional. The JOIN segment is used during explicit joins and the WHERE to specify, if any, the constraints on selection. In function of the number and position of the keywords in the sentence, the division is not the same and will not give place to the same output query structure. It can be noted, in particular, that if a request contains no word similar to a table, it will necessarily be invalid and will generate an error.
- From each of the segments obtained during the cutting the keywords are analyzed. These words can be presence factors of a query called counting, algebraic calculations, a negation, etc., or else a value on which a constraint must be made. In this way, if a word referring to the count as for example, "how much" is found in the first segment of the sentence, that corresponding to the SELECT, the system identifies the request to be generated as being a count request, that is to say a SELECT COUNT.

  Here only the inner joins are considered. Two types of inner joins exist, implicit and explicit. When there is a selection on constraint on a column that is not part of the FROM table, i.e. when the table to which the targeted column belongs is not mentioned in the sentence entry is an implicit join. In this case, make a join between the table of the target column and the table FROM which is specified in the sentence. In the case of an explicit join, the table on which to perform the join is specified directly in the sentence.
- It is then the role of the rules of grammar to reduce again the space of the possible queries until proposing the most plausible. An integer is assigned to each key element of the request, the 1 for the elements of table type, 2 for columns, 3 for values, 4 for a count word, etc. The entry request gives hence place in a series of numbers that one concatenates, thus forming a single integer. The 0 corresponds to the fact that the element previous in the string can be present 1 to N times within the rule. So, the result is an array of indexed rules by integers. To find the rule equivalent to a request entered, just search for the integer corresponding to the demand structure, which also happens to be the key to value, representing the structure of the query to be produced at the exit, in the table. The query is generated based on the presence, number and order of keywords identified in the sentence entered by the user.
- Once a first candidate SQL query obtained, it is run on the database. If the request is invalid is that it is badly constructed the system then tries to build it differently, but executing it, if it still returns an error, the system returns an error message.

V. Results And Discussion

If the user does not enter a correctly spelled sentence or whose vocabulary used is not strictly the same as that of the database, no correspondence between his sentence and elements of the base will be found and no relevant result will be returned. Also, if a request contains no word similar to a table, it will necessarily be invalid and will generate an error. The following example shows how the system fails to generate query if the sentence spoken is not proper.

The presence of a SELECT and FROM segment is mandatory in the sentence, the first designates which will be the type of selection and on which element(s) exactly, the second specifies where to look, in which table(s), the element of the selection. The

JOIN and WHERE segments are optional. The JOIN segment is used during explicit joins. The system is capable of generating SQL query where two or more tables need to be joined. The following image shows one of such conditions

Query with a Join

VI. CONCLUSION

- Speech recognition models in association with Google Web Speech API is used.
- User doesn't need to know the table names. Tables are fond with the help of attributes specified in the query.
- System has been checked for single tables and multiple tables and it gives correct result if the input query is syntactically consistent with the Syntactic Rules.
- System be configured automatically for different databases.

VII. FUTURE WORK

- Area in which system can be improved is the analysis and acceptance of more general query by improving the grammar.
- The complex queries that can be considered are queries that make use of HAVING clauses.
- We will also note the impossibility of this one to manage the mute constraints and to generate nested queries. In future works, we plan to deal with mute constraints, keeping the verbs as key words and by adding some rules in grammar.
- In addition, it is intended to detect the language of the request entered by the user to use a dictionary related to this language and adjust the rules according to the language so to make the robust system with variable languages.
- The accuracy for the speech recognition can be improved in order to process the natural language better.

VIII. ACKNOWLEDGMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crowned all efforts with success. We are grateful to our faculties and friends for the guidance, inspiration and constructive suggestion that helped us in the preparation of this project. We would like to thank B.M.S. Institute of Technology & Management for allowing us to do this project successfully.

REFERENCES

[1] F.Siasar djahantighi, M.Norouzifard, S.H.Davarpanah2, M.H.Shenassa, "Using Natural Language Processing in Order to Create SQL Queries", Proceedings of the International Conference on Computer and Communication Engineering 2008, Published by IEEE.

[2] Sachin Kumar, Ashish Kumar, Dr. Pinaki Mitra, Girish Sundaram, "System and Methods for Converting Speech to SQL" International Conference on "Emerging Research in Computing, Information, Communication and Applications" ERCICA 2013 pp: 291-298, Published by Elsevier Ltd.

[3] Prabhdeep Kaur, Shruthi J "Conversion of Natural Language Query into SQL" International Journal of Engineering Sciences & Emerging Technologies, Jan. 2016, Volume 8, Issue 4, pp: 208-212 published by IJESET.

[4] Prasun Kanti Ghosh, Saparja Dey, Subhabrata Sengupta, "Automatic SQL Query Formation from Natural Language Query", International Journal of Computer Applications (0975 – 8887), International Conference on Microelectronics, Circuits and Systems (MICRO-2014).