# Query Easy

[1] Mr. Sairaj Dattatray Pawar, [2] Mr. Vishal Subhash Dede, [3] Mr. Bhushan Shriniwas Salunkhe, [4] Mr. Sarfaraj Isahak Patel, [5] Mr.Shivtej Rajendrakumar Nalawade
[6] Guidence: Prof. Swapnali Patil

[1] *Student, Computer Science And Engineering, Shree Santkrupa Institute of Engineering and Technology Ghogaon, Maharashtra, India*
[2] *Student, Computer Science And Engineering, Shree Santkrupa Institute of Engineering and Technology Ghogaon, Maharashtra, India*
[3] *Student, Computer Science And Engineering, Shree Santkrupa Institute of Engineering and Technology Ghogaon, Maharashtra, India*
[4] *Student, Computer Science And Engineering, Shree Santkrupa Institute of Engineering and Technology Ghogaon, Maharashtra, India*
[5] *Student, Computer Science And Engineering, Shree Santkrupa Institute of Engineering and Technology Ghogaon, Maharashtra, India*
[6] *Professor, Computer Science And Engineering, Shree Santkrupa Institute of Engineering and Technology Ghogaon, Maharashtra, India*

## ABSTRACT

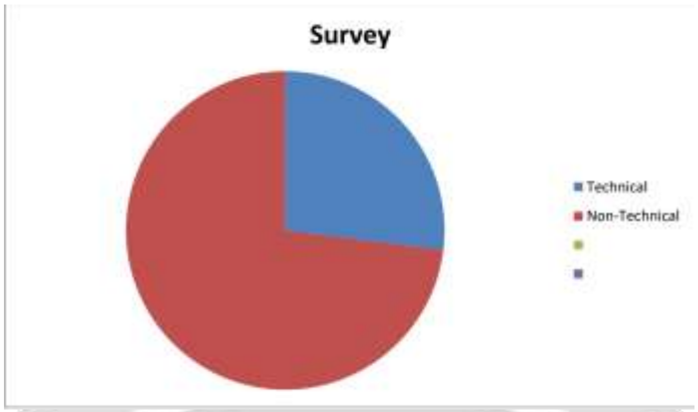
*The user can access data using English language rather than using complex SQL queries. Those who don't have knowledge to handle database can also access data. It will make human life comfortable. Chatbots are an extremely prominent way to interact with a software system. The need to build maintainable that scalable systems is more present than ever, while the understanding of needed technologies is generally lacking. This helps to interact easily with system. This is demonstrated by disconnected literature, the popularity of oversimplified building tools, and generally sub-par conversational agents.*

## 1. INTRODUCTION

Chatbots are revolutionizing the way people interact with technology. They bridge the gap between human understanding and respective computer interpretation. With a chatbot interface the user does not need any kind of technical background and leveraging the modern-day AI and NLP tools we can get maximum efficiency. We have implemented a Natural Language interface to access a Database Management System for users who do not have knowledge about SQL. We have reduced the communication gap between the user and the system. The user can access the database using a chatbot interface where he/she can retrieve data using simple English language rather than using complex SQL queries or instead of using the SQL queries user can access the data present in database by using simple understandable English sentence. This can reduce the efforts of remembering queries and saves data accessing time of the user.

## 2. LITERATURE SURVEY

There have been a large amount of research works introducing the theories and applications of NLIDBs. Asking question to databases in natural language is very appropriate and easy method of information access especially for informally users who do not comprehend complex database query language. In fact, database NLP may be one of the most significant successes in NLP since it began. NLTK contains the lot of inbuilt tools which helps to NLP for asking questions to databases in natural language instead of using database complex queries and get the results easily.



Most institutions have their data in the digital format. It becomes difficult for people with no technical background to access data and find insightful information from it. Companies have a compulsive need to invest in a team of technical professionals for these purposes. This graph shows the distribution of professional crowd with respect to their technical background.

## 3. OVERVIEW OF THE SYSTEM

Now days data is increasing rapidly and problems related to accesses and retrieve the data can arises rapidly. There are so many new database tools and technologies are rowing, therefore we can store large data, but the problem is that the technology or an interface which can process data and display the data as per the user request is not familiarized with many of the people. It means many people don't have proper knowledge of handling database.

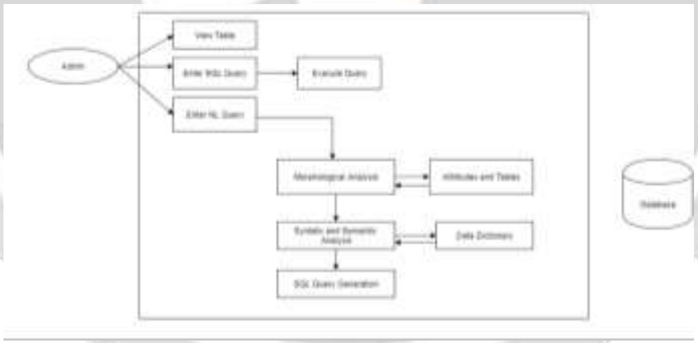


**Fig 1 : System Architecture**

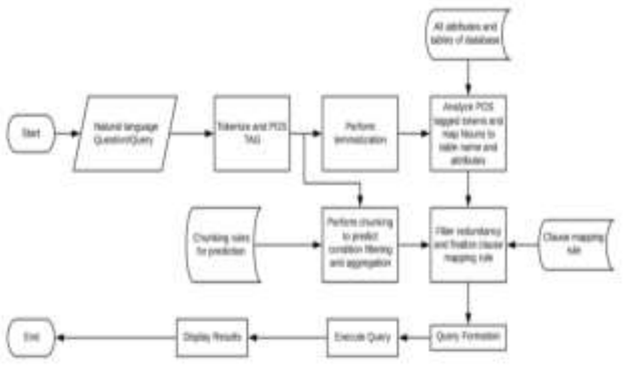

All attributes and tables of database
Start
Natural language Question/Query
Tokenize and POS TAG
Perform lemmatization
Analyze POS tagged tokens and map Nouns to table name and attributes
Chunking rules for prediction
Perform chunking to predict condition filtering and aggregation
Filter redundancy and finalize clause mapping rule
Clause mapping rule
End
Display Results
Execute Query
Query Formation

**Fig 2 : Flow Chart**

The proposed approach aims to use knowledge of SQL to create a corpus which will help to identify SQL command words ie SELECT, INSERT, DELETE, UPDATE and map the tokens with appropriate POS. Word similarities will be calculated with the input tokens to the database schema (table names, column names, data) to insert table names, column names, and data comparisons into the query.

- User Interface: The user interacts with the system via Graphical User Interface and types his/her Natural Language Query for the further output.
- Lowercase Conversion: The Natural Language Query is then translated into lowercase and passed to the tokenization.
- Tokenization: The query after lowercase conversion is then transformed into stream of tokens and a token id is providing to each word of NLQ.
- Escape word removal: The extra/stop words are removed which are not needed in the analysis of query.
- Part of Speech Tagger: The tokens are then classified into nouns, pronouns, verb and string/integer variables.
- Relations-Attributes Clauses Identifier: Now the system classifies the tokens into relations, attributes and clauses on the basis of tagged elements and also separates the Integer and String values to form clauses.
- Ambiguity Removal: It removes all the ambiguous characteristics that exists in multiple relation with the same attribute name and maps it with the correct relation.
- Query Formation: After the relations, attributes and clauses are extracted, the final query is built.
- Query Execution and Data Fetching: The query is then executed and data is got from the database.
- Results: The final query result is displayed to the user on the Graphical User Interface.

## 3.1 IMPLEMENTATION

- System Design: System Design We propose a system which looks to overcome the shortcomings of existing system that gets a natural language sentence as an input, which is then passed through various phases of NLP to form the final SQL query.

- Tokenize and Tag: The input natural language query gets split into different tokens with the help of the tokenizer word_tokenizer, from 'NLTK' package. The tokenized array of words is tagged according to the part-of-speech tagger using the Stanford POS tagger [14]. All processes following this step use these tagged tokens for processing.

- Analyze tagged tokens: Based on the tagged tokens of earlier step, the noun map and verb list is prepared through one iteration over the tokens. The tokens corresponding to aggregate functions are also mapped with their respective nouns using a pre-created corpus of words. The decision whether the natural language statement represents a data retrieval query (SELECT) or a DML query (INSERT, UPDATE, DELETE) is taken at this stage with the help of certain 'data arrays' for denoting type of query. For example, when words like 'insert' and its certain synonyms appear in the input, the type of query is 'INSERT' and so on. In any type of query, the tentative tags 'S' (SELECT), 'W' (WHERE), 'O' (ORDER BY) are mapped to the nouns indicating the clauses to which they belong. For this, we have designed 'data dictionaries' for different clauses. These data dictionaries consist of the token clause term pair, for e.g. aggregate clause data dictionary is ”number”: ”COUNT”, ”count”: ”COUNT”, ”total”: ”SUM”, ”sum”: ”SUM”, ”average”: ”AVG”, ”mean”: ”AVG”. Thus, if any of these tokens is encountered, it is likely to have aggregate clause and accordingly the nouns are tagged with the clause tag.

- Map to table names and attributes: Using the noun map and verb list, the table set is prepared, which will hold the tables that are needed in the query to be formed. This is based on the fact that the table names are either nouns or verbs. The noun map is used to find the attributes which are needed in the final query. The attributes, the table associated with the attribute and the clause tag are stored in an attribute-table map which is used in the final stage of query formation.

## 4. CONCLUSIONS

This project has given us a great opportunity to come up with an solution for writing tedious queries. This project though helps resolving basic queries but with time it can be made powerful to handle complex queries, normalization and also can be extended for nosql. We were able to learn and implement NLTK, python3. We have got accuracy around 30-50% in basic queries.

## 5. ACKNOWLEDGEMENT

We would like to express our deep gratitude towards Prof. Swapnali Patil mam for their constant encouragement and guidance. Their support and cooperation throughout the project work has been of immerse help to us. We express our sincere thanks to all the teaching and non teaching staff and all those who have directly or indirectly helped in making project a success.

## 6. REFERENCES

[1]. Natural Language To SQL Conversion System, International Journal of Computer Science Engineering and Information Technology Research (IJCSEITR)
[2]. Gauri Rao(IJCSE) International Journal on Computer Science and Engineering.
[3]. A Survey of Natural Language Query Builder Interface to Database, International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 5 Issue 4, 2015