Abstract

The computer is an evolving machine. Today, the use of computer is pervasive to store, process and retrieve the information in all types of manufacturing and service sectors. The usage of computer is increased to manifold with the advent of Database Management System (DBMS) and with the dawn of Internet. Everyday most of the people are interacting with information stored in database, usually, through the specialized and predefined programs written for such interaction. The limitation of such programs is that it restricts the interaction between user and database to predefined set of queries. This is against the basic objective/principle of DBMS system which is developed to support ad-hoc queries. Only few people who have knowledge of database structure and formal database language (such as Structured Query Language (SQL) can retrieve the desired information from database.

A novice user having no knowledge of database structure and formal database query language cannot retrieve desired information if it is not supported by well thought application. Hence, it was a need of an hour to improve human computer interface that allows people to interact with the database in their natural language (such as English). The effective Natural Language Interface to Database (NLIDB) system has high potential to mimic computer systems as conversational system thereby making it easy to use by large mass of people enhancing its utility value.

At the end, several systems were designed and developed by researchers that support NLIDB. Some of the well-known NLIDB systems are: LIFFER, LUNAR, BANKS, ELF, NLBEAN, CINDI, SQL-TUTOR, SQ-HAL, Step, etc.

The challenge in designing and developing NLIDB systems is to derive the users’ intended meaning from limited number of query words
(sometimes, only few words) and to map it to structured database language for extracting the relevant output from database. Hence, the NLIDB systems need to perform, semantic analysis, word sense disambiguation and context resolution for correct interpretation of natural language query in addition to syntactic processing. The existing systems lack all these consideration of NLP. Additionally, most of the systems restrict user input to predefined templates and do not allow free-form query text.

The objective of the present research is, therefore, to provide more efficient and user friendly Natural-English Language Query Interface to Database (N-ELIDB) system that allows non-technical users, who are not aware of formal database language (such as SQL), to interact with database using their own English language. The present work proposes the general framework for efficient processing of natural language query to extract intended information from the database.

Some of the characteristics of the proposed system are as follows: (i) the user must enter the query in English Language (ii) the user can enter query using free-form text (grammar and spelling mistakes are allowed) (iii) while entering query, system suggests auto complete text (iv) the user can select the query from existing queries in knowledge base (v) the user can select the template, can modify it and can submit the modified query and (vi) the query entered by the user is not case sensitive.

The operational methodology of N-ELIDB system has major two components: (a) Linguistic Component and (b) Database Component. The Linguistic component discusses (i) Morphological Analysis like query pre-processing & context resolution, word-based n-gram processing, stop words removal, spelling check, domain mapping and knowledge base management (ii) Lexical Analysis like identifying token type and checking for attribute token (iii) Syntactic Analysis using Stanford Parser and Multi-Liaison algorithm, (iv) Semantic Analysis using WordNet for semantic representation of lexicon and proper noun
resolution. The Database component discusses (i) SQL Query Generator which processes intermediate query representation and generates SQL query for DDL, DML & SELECT statements, and (ii) SQL Query Execution using database adaptor.

The proposed algorithm (prototype system) can handle following types of queries: List Queries, Condition Queries, Group Queries, Order Queries, Time Oriented Queries, Range Queries, Negation Queries, Join Queries, Object Swapping Queries, Data Definition Language (DDL) Queries, and Data Manipulation Language (DML) Queries. It is tested with several real world queries from each query category. The accuracy of the system is measured in term of precision percentage with three classes that identifies query response as: Correct, Incorrect and Partially Correct. The results found are encouraging and the overall efficiency of system is observed to be more than 70%. Hence, the proposed framework performs well compared to similar systems such as CINDI, SQL-HAL, etc.