Imperial Journal of Interdisciplinary Research (IJIR)
Vol.2, Issue-1 , 2016
ISSN : 2454-1362 , www.onlinejournal.in

Natural Language Interface to Database-An Introduction

Joginder Singh
Department of Computer Engineering
Kalpana Chawla Government Polytechnic For Women, Ambala City, Haryana, India

Pratiba Verma
(Asst. Prof.) Dept. of Computer Engineering
Sri Sukhmani Institute of Engg. & Technology Derabassi, Punjab, Indiaia

Navneet Kaur
(Asst. Prof.) Dept. of Computer Engineering
Sri Sukhmani Institute of Engg. & Technology Derabassi, Punjab, India

Abstract

Information plays an important role in our lives .it is required numerous applications. One of the major sources of information is databases. Databases and database technology are growing rapidly and are having major impact on the growing use of computers. Computers are used in accessing of information from database. Almost all IT applications are storing, retrieving organizing, accessing analyzing, and information from databases. using the internet. Retrieving information from database requires knowledge of database languages like Structured Query Language (SQL) However, not everybody is able to write SQL queries as they may not be aware of the syntax of SQL structure of the database,. This has lead to the developing such a system where non-expert users compose their questions in their natural language and obtain the results in the form of database. So instead of working with the SQL one can query relational databases in their natural language. This new idea is provoked to develop new type of processing system called natural language interface to database Natural language interface to database enhances the users performance and is a flexible approach querying in database. This paper is an introduction to natural language interfaces to databases (NLIDBS). Covers a brief overview of the NLIDB its components its advantages disadvantages, approaches and techniques

.

Keywords: SQL (Structured Query Language), DBMS Database Management System), NLIDB (Natural Language Interface To Database).

1. Introduction

In the today's computing world computer based technologies are extensively used in managing information in almost all the sort off organizations e.g educational, private, government etc. Information management systems are used to manage data. Information can be accessed from anywhere and by anyone. Largest source of information in computer world is database. **Database**: A database is collection of related the data, organized in a systematic way, in such a way that computer can quickly select a particular data from huge data. Databases are comprehensive and important element in organizations In order to extract information from database one needs to formulate a query that the computer can understand and produce the desired data. Plenty of database tools are available in the market for retrieving information from database. To retrieve information from database, applications require knowledge of database language such as Structured Query Language (SQL). One has to learn language SQL to access the information stored in relational databases. Hence it becomes difficult for normal user to retrieve information without SQL. The most common way for people to obtain information is by asking questions in their natural language. To find a alternate to this many researches turned out to the use of Natural Language like Hindi, English, Punjabi, Tamil, French etc. for retrieving information. This approach is helpful for naïve users in accessing the databases.

2. Natural Language Interfaces to Databases

In recent times there is has been a rising demand for non expert users to query relational databases in natural languages instead of artificial languages This idea led to the development of the new processing system called Natural Language Interface to database system (NLIDB). It is interface between computer and human. In this human will type a question in natural language for the information required from data base. Thus it frees the user from knowledge of query language and database access is simplified. (Androutsopoulos et al., 1995). User need not learn a new database language, can frame a query in natural/native language, reducing the burden on user. Although the earliest research has started since the

late sixties [1]. Several NLIDB systems have also been made for commercial use; regardless, the use of NLIDB systems certainly is not wide-spread and it is not a standard option for interfacing to a database. There are many deficiencies in NLIDB yet it is helpful in many ways.

2.1 Advantages of NLIDB

- No requirement of Artificial Language:
- No need of Training
- Simple and easy to use
- Better for some question
- Easy to use for multiple database tables
- Learning not required
- Knowledge of the physical structure of data is not required:
- Fault tolerance

2.2. Disadvantages

- Linguistics coverage is not obvious
- Linguistics vs. conceptual failure
- Inappropriate Medium
- Deals with limited set of natural language
- Tedious configuration

3. HISTORYOF NLIDB

The late sixties and early seventies were an active period in database research. Since late 1960s there has been a large number of research work introducing the theories and implementations of NLIDBs. Here are some examples of the Natural Language Interface to Database systems

3.1 LADDER

The Ladder (Language Access to Distributed Data with Error Recovery) [4] system was designed as a natural language interface to a database of information about US Navy ships. It takes queries in English language. The system was designed as a management aid to navy decision makers LADDER smears all the necessary information concerning the vocabulary and syntax of query, the name of specific attribute, how they are formulated and where the attributes are physically located to deliver the output. According to [5], the LADDER system uses semantic grammar to parse questions to query a distributed database. The system uses semantic grammars technique that interleaves syntactic and semantic processing. The queries from the INLAND are directed to the Intelligent Data Access (IDA), which is the second component of LADDER. IDA split a query against the entire VLDA into a sequence of queries against individual files According to [6], the INLAND component builds a fragment of a query to IDA for each lower level syntactic unit in the English language input query and these fragments are then combined to higher level syntactic units to be recognized. At the sentence level, the combined fragments are sent as a command to IDA. IDA would compose an answer that is relevant to the user's original query in addition to planning the correct sequence of file queries.
The third component of the LADDER system is for File Access Manager (FAM).The task of FAM is to find the location of the generic files and manage the access to them in the distributed database. The system LADDER was implemented in LISP. At the time of the creation of the LADDER system was able to process a database that is equivalent to a relational database with 14 tables and 100 attributes.

3.2 LUNAR SYSTEM

Lunar was the first system informally introduced in 1971 The system LUNAR [4] is a system which answers questions about samples of rocks brought back from the moon.. To accomplish its function the LUNAR system uses two databases.

1. One for the chemical analysis
2. Other for literature references.

The LUNAR system uses an 'Augmented Transition Network (ATN) parser and Woods' procedural semantics. According to [8], the LUNAR system performance was quite impressive; it managed to handle 78% of requests without any errors and this ratio rose to 90% when dictionary errors were corrected. But these figures may be misleading because the system was not subject to intensive use due to the limitation of its linguistic capabilities. A scientist who used it to extract information for everyday work would soon have found that he wanted to make requests beyond the linguistic ability of the system. ATN parsers are useful because they are very efficient, even for large grammars; however, ungrammatical sentences are not handled well and they are not very flexible.

3.3 RENDEZVOUS SYSTEM

This system appeared in late seventies. In this, users could access databases via relatively unrestricted natural language. In this Codd's system, special emphasis is placed on query paraphrasing and in engaging users in clarification dialogs when there is difficulty in parsing user input.

3.4 PLANES

This was developed in late seventies for (Programmed Language-Based Enquiry System) at the University of Illinois Coordinated Science Laboratory. PLANES include an English language front end with the ability to understand and explicitly answer user requests. It carries out clarifying dialogues with the user as well as answer vague or poorly defined questions. This work is being carried out using database based upon information of the U.S. Navy 3-M (Maintenance and Material Management), it is a database of aircraft maintenance and flight data, although the ideas can be directly applied to other non-hierarchic record-based databases [9]. PHILIQA This was developed in 1977 and was known as Philips Question Answering System [10], uses a syntactic parser which runs as a separate pass from the semantic understanding passes. This system is mainly involved with problems of semantics and has three separate layers of semantic understanding. The layers are called "English Formal Language", "World Model Language", and "Data Base Language" and appear to correspond roughly to the "external", "conceptual", and "internal" views of data.

3.5 CHAT-80 SYSTEM

The system CHAT-80 [11] is one of the most referenced NLP systems in the eighties. The system was implemented in Prolog. According to [12], the CHAT-80 was an impressive, efficient and sophisticated system. The database of CHAT-80 consists of facts (i. e. oceans, major seas, major rivers and major cities) about 150 of the countries world and a small set of English language vocabulary that are enough for querying the database. The CHAT-80 system processes an English language question in three stages as depicted.

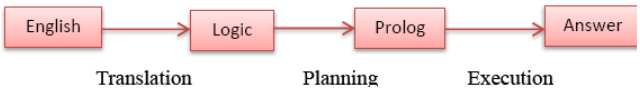


Figure : Working phases of CHAT-80

The system translates the English language question into three serial and complementary functions where:

1. Words are represented by logical constants.
2. Verbs, nouns, and adjectives with their associated prepositions are represented by predicates. The predicates can have one or more arguments.
3. Complex phrases or sentences are represented by conjunctions of predicates.

These functions are being, parsing, interpretation and scoping. The parsing module function determines the grammatical structure of a sentence and the interpretation and scoping consist of various translation rules, expressed directly as Prolog clauses. The basic method followed by Chat-80 is to append some extra control information to the logical form of a query in order to make it an efficient piece of Prolog program that can be executed directly to produce the answer. According to [11], the generated control information comes into two forms:

1. Orders the predications for a query that will determine the order in which Prolog will attempt to satisfy them.
2. Separates the overall program into a number of independent sub problems to limit the amount of backtracking performed by Prolog.

3.6 TEAM SYSTEM

It was developed in 1987. A large part of the research of that time was devoted to portability issues. TEAM was designed to be easily configurable by database administrators with no knowledge of NLIDBs [13, 14].

3.7 ASK SYSTEM

This system developed in 1983, allowed end-users to teach the system new words and concepts at any point during the interaction. ASK was actually a complete information management system, providing its own built-in database and the ability to interact with multiple external databases, electronic mail programs and other computer applications. All the applications connected to ASK were accessible to the end user through natural language requests. The user stated his/her requests in English and Ask transparently generated suitable requests to the appropriate underlying systems.

3.8 JANUS SYSTEM

It had similar abilities to interface to multiple underlying systems (databases, expert systems, graphics devices, etc). All the underlying systems could participate in the evaluation of a natural language request, without the user ever becoming aware of the heterogeneity of the overall system. JANUS is also one of the few systems to support temporal questions [15].

3.9 EUFID SYSTEM

The EUFID system consists of three major modules, not counting the DBMS. First is analyzer module, second is mapper module and third is translator module. [16]

3.10 DATALOG SYSTEM

It is an English database query system based on Cascaded ATN grammar. By providing separate representation

schemes for linguistic knowledge, general world knowledge, and application domain knowledge, DATALOG achieves a high degree of portability and extendibility [17]. Systems that also appeared in mid-eighties were LDC [18], TQA [19] TELI [20] and many others.

3. WENHUA

Wenhua (1992) designed an NLP system for Computer Integrated Manufacturing (CIM) databases, which are large, diverse and represent completely different concepts, from accounting to computer-aided-design and schedule planning. Four different databases were used, and one NLP system was designed to access all these databases, as necessary, to answer the user's questions. This system uses a Definite Clause Grammar (DCG) and a semantic interpreter to process the English question into a database query. This system is much larger than the earlier attempts at database NLP systems and advanced queries are possible across the different databases and different data representations.

4 Natural Language Interface for Unstructured Data:

These systems do not restrict themselves to interact with data in database tables only. Data from various sources can be used and accumulated. Following are the some of the systems discussed.

4.1 ELIZA- by Joseph Weisenbaum (1966)

This program is a natural language interface to a psychiatrist. It used pattern-matching rules that were triggered based on key words found in user's dialog. ELIZA used literal text form within users dialog to reformulate questions. There was no 'understanding' of what was being said, ELIZA just gave back questions that seemed most relevant according to the last user input. Weisenbaum reported that some subjects were convinced that ELIZA was a real person. He notes "The human speaker will contribute much to clothe ELIZA's responses investments of plausibility."

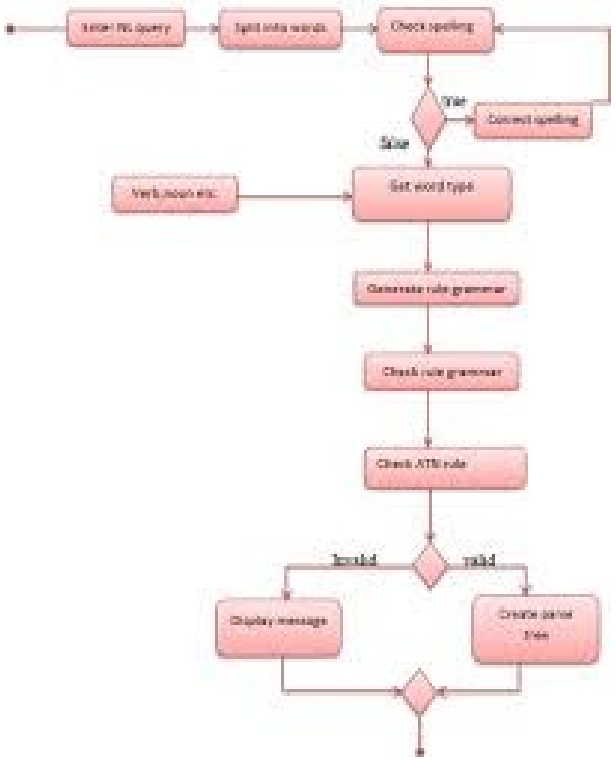
4.2 SHRDLU – by Terry Winnograd (1973)

This is one of the first programs that could carry out tasks and provide responses in natural language well. It was bound within an artificial blocks world of coloured bricks and pyramids. SHRDLU was able to perform tasks like moving objects around within the limited world, when directed to do so in English. The program used a procedural representation for semantics. This means that each English predicate or term was associated with a procedure that conveyed the meaning (or semantics) of the term. The problem with procedural semantics is that they do not scale up into large domains.

4.3 GINLIDB

The system was developed in 2009. It is called a Generic Interactive Natural Language Interface to Database. It is designed by the use of UML and developed using visual basic former component control the natural language query correctness as far as the grammatical structure and the successful transformation to SQL statement, opens a connection to database in use, execute the generated SQL statement and return the query result to the user. The user submitted query in English language is analyzed for the syntactic as well as semantic merits if the query is correct and can be answered efficiently with respect to system knowledge base. Then the construction of valid SQL statement that represent the user's query is done and to retrieve the result of query to the user. The results of GINLIDB are very successful and satisfactory.

NET 2005. The system consists of two major components .Linguistic handling component and SQL constructive component [24].



4.4 Recent Developments in NLIDB

This section provides a brief overview of three specific NLIDB systems developed recently in different universities.

(i) NALIX SYSTEM

NALIX (Natural Language Interface for an XML Database) is an NLIDB system developed at the University of Michigan, Ann Arbor by Yunyao Li, Huahai Yang, and H. V. Jagadish (2006). The database used for this system is extensible markup language (XML) database with Schema Free X Query as the database query language. Schema-Free X Query is a query language designed mainly for retrieving information in XML. The idea is to use keyword search for databases. However, pure keyword search certainly cannot be applied. Therefore, some richer query mechanisms are added [23]. Given a collection of keywords, each keyword has several candidate XML elements to relate. All of these candidates are added to MQF (Meaningful Query Focus), which will automatically find all the relations between these elements. The main advantage of Schema-Free X Query is that it is not necessary to map a query into the exact database schema, since it will automatically find all the relations given certain keywords. NALIX can be classified as syntax based system, since the transformation processes are done in three steps: generating a parse tree, validating the parse tree, and translating the parse tree to an X Query expression. However, as implied in the paper [22][23], NALIX is different from the general syntax based approaches; in the way the system was built: NALIX implements a reversed-engineering technique by building the system from a query language toward the sentences.

(ii) PRECISE

Is a system developed at the University of Washington by Ana-Maria Popescu, Alex Armanasu, Oren Etzioni, David Ko, and Alexander Yates (2004). The target database is in the form of a relational database using SQL as the query language. It introduces the idea of semantically tractable sentences which are sentences that can be translated to a unique semantic interpretation by analyzing some lexicons and semantic constraints [24]. PRECISE was evaluated on two database domains. The first one is the ATIS domain, which consists of spoken questions about air travel, their written forms, and their correct translations in SQL query language. In ATIS domain, 95.8% of the questions were semantically tractable. Using these questions gives PRECISE 94% precision. The second domain is the GEOQUERY domain. This domain contains information about U.S. Geography. 77.5% of the questions in GEOQUERY are semantically tractable. Using these questions gives PRECISE 100% accuracy. The strength of PRECISE is based on the ability to match keywords in a sentence to the corresponding database structures. This process is done in two stages, first by narrowing the possibilities using Max flow algorithm and second by analyzing the syntactic structure of a sentence. Therefore PRECISE is able to perform impressively in semantically tractable questions. As other NLIDB systems, PRECISE has its own weaknesses. While it is able to achieve high accuracy in semantically tractable questions, the system compensates for the gain in accuracy at the cost of recall. Another problem is as PRECISE adopts a heuristic based approach, the system suffers from the problem of handling nested structures.

(iii) WASP

WASP (Word Alignment-based Semantic Parsing) is a system developed at the University of Texas, Austin by Yuk Wah Wong [25]. While the system is designed to address the broader goal of constructing "a complete, formal, symbolic, meaningful representation of a natural language sentence", it can also be applied to the NLIDB domain. A predicate logic (Prolog) was used as the formal query language. WASP learns to build a semantic parser given a corpus a set of natural language sentences annotated with their correct formal query languages [26]. It requires no prior-knowledge of the syntax, because the whole learning process is done using statistical machine translation techniques. WASP was evaluated on the GEOQUERY domain, the same domain as PRECISE. GEOQUERY corpus consists of 880 questions in the training set and 250 questions in the test set, which are merged together into one larger data set. Each data set was divided to 10 equal-sized subsets, and standard 10-fold cross validation was used to estimate the system performance. WASP achieved 86.14% precision and 75.00% recall in the GEOQUERY domain. The system was also evaluated on a variety of other natural languages: English, Spanish, Japanese and Turkish. There were no significant differences observed between English and Spanish, but the Japanese corpus has the lowest precision and the Turkish corpus has the lowest recall. The strength of WASP comes from the ability to build a semantic parser from annotated corpora. This approach is beneficial because it uses statistical machine translation with minimal supervision. Therefore, the system does not have to manually develop a grammar in different domains. Moreover, while most of NLIDB systems use English as their natural language, WASP has been tested on several languages. In spite of the strength, WASP also has two weaknesses. The first is: the system is based solely on the analysis of a sentence and its possible query translation, and the database part is therefore left untouched. There is a lot of information that can be extracted from a database, such as the lexical notation, the structure, and the relations within. Not using

this knowledge prevents WASP to achieve better performances. The second problem is that the system requires a large amount of annotated corpora before it can be used, and building such corpora requires a large amount of work.

5 Component of NLIDB

NLIDB has divided into two sub-components by computing scientists [19]. One is Linguistic and other is Database component. These are briefly discussed below.

5.1 Linguistic component*:*

Linguistic component is responsible for translating natural language input into a formal query and generating a natural language response based on the results from the database search.

5.2 Database Component

It performs traditional Database Management functions. Natural language database systems make use of syntactic knowledge and knowledge about the actual database in order to properly relate natural language input to the structure and contents of that database. A lexicon is a table that is used to map the words of the natural input onto the formal objects of the database. A lexicon consists of a number of tables that store natural language words and their corresponding mapping to formal objects that will be used to create a formal query. These tables can have entries of relations name, attribute names, verbs, adverbs etc. Both parser and semantic interpreter make use of the lexicon. A natural language generator takes the formal response as its input, and inspects the parse tree in order to generate adequate natural language response. . Natural language database systems make use of syntactic knowledge and knowledge about the actual database in order to properly relate natural language input to the structure and contents of that database.Syntactic knowledge usually resides in the linguistic component of the system, in particular in the syntax analyzer whereas knowledge about the actual database resides to some extent in the semantic data model used. Questions entered in natural language translated into a statement in a formal query language. Once the statement unambiguously formed, the query is processed by the database management system in order to produce the required data. These data then passed back to the natural language component where generation routines produce a surface language version of the response.

6 Architecture of NLIDB Systems

Natural language is the area of interest from computational perspective due to the hidden uncertainty that language possesses.The field of NLIDB has gained a remarkable expansion in the last few decades.Many researchers have adopted different techniques to deal with language in different systems, in order to enhance the efficiency of the NLIDB interface.Next few sub-sections describe miscellaneous strategies that are used to process language for a range of purposes.

6.1 Pattern Matching Systems

The early efforts in the NL interfaces area started back in sixties [2]. Prototype systems had appeared in the late sixties and early seventies. Many of these systems relied on pattern matching to directly mapping the user input to the database [2]. Formal List Processor (FLIP) is an early language for pattern matching based on LISP structure [27] works on the bases that if the input matches one of the patterns then the system is able to build a query for the database. In the pattern matching based systems, the database details were inter-mixed into the code, limited to specific databases and to the number and complexity of the patterns. As the usage of databases has spread during the 1970's, the concept of user interface presented new challenges to the designers. One approach was the use of natural language processing, where the user interactively is allowed to interrogate the stored data.
The main advantage of the pattern-matching approach is its simplicity. In such systems no elaborate parsing and interpretation modules are needed, and the systems are easy to implement [26]. Also, pattern matching systems often manage to come up with some reasonable answer, even if the input is out of the range of sentences the patterns were designed to handle. One of the best natural language processing system that role in this style is ELIZA. ELIZA functions by processing users, by these responses to the scripts. It typically says differently and rephrased the statements of the users as questions and replies the answers of those questions to the patient. ELIZA was programmed by Mr. Joseph Weizenbaum in nearly from 1964 to 1966.

6.2 Syntax-Based Systems

In syntax-based systems the users question is parsed (i.e. analyzed syntactically) and the resulting parse tree is directly mapped to an expression in some database query language. Syntax-based systems use a grammar that describes the possible syntactic structures of the user's questions. Syntax based NLIDBs usually interface to application-specific database systems that provide database query languages carefully designed to facilitate the mapping from the parse tree to the database query. It is usually difficult to devise mapping rules that will

transform directly the parse tree into some expression in a real life database query language (e.g. SQL).The main advantage of using syntax based approaches is that they provide detailed information about the structure of a sentence. A parse tree contains a lot of information about the sentence structure; starting from a single word and its part of speech, how words can be grouped together to form a phrase, how phrases can be grouped together to form more complex phrases, until a complete sentence is built. Having this information, we can map the semantic meanings to certain production rules (or nodes in a parse tree).Unfortunately not all nodes should be mapped; some nodes have to be left just as they are without adding any semantic meanings. And it is not always clear which nodes should be mapped and which should no. Moreover the same node in different parse trees is not necessarily going to be translated in all the trees. The second problem is a sentence can have multiple correct parse trees, and if all are translated, they may lead to different query results. The last problem is that it is difficult for a syntax based approach to directly map a parse tree into some general database query language, such as SQL (Structured Query Language). In semantic grammar systems, the question-answering is still done by parsing the input and mapping the parse tree to a database query. The difference, in this case, is that the grammar's categories do not necessarily correspond to syntactic concepts. Semantic information about the knowledge domain is hard-wired into the semantic grammar that's why systems based on this approach are very difficult to port to other knowledge domains a new semantic grammar has to be written whenever the NLIDB is configured for a new knowledge domain. Semantic grammar categories are usually chosen to enforce semantic constraints [28]. Much of the systems developed till now like LUNAR, LADDER, use this approach of semantic grammar.

6.3 Semantic Grammar System

A semantic grammar system is very similar to the syntax based system, meaning that the query result is obtained by mapping the parse tree of a sentence to a database query. The basic idea of a semantic grammar system is to simplify the parse tree as much as possible, by removing unnecessary nodes or combining some nodes together. Based on this idea, the semantic grammar system can better reflect the semantic representation without having complex parse tree structures. Therefore, a production rule in a semantic grammar system does not necessarily correspond to the general syntactic concepts. Instead of smaller structures, the semantic grammar approach also provides a special way for assigning a name to a certain node in the tree, thus resulting in less ambiguity compared to the syntax based approach. Both of the ambiguities that can occur in mapping a node to its semantic label and the number of different parse trees which are possible for a particular sentence [29]. The main drawback of semantic grammar approach is that it requires some prior- knowledge of the elements in the domain, therefore making it difficult to port to other domains. In addition, a parse tree in a semantic grammar system has specific structures and unique node labels, which could hardly be useful for other applications. Regardless, there are on-going attempts to automatically build the grammar rules by obtaining the prior-knowledge based on user interaction or by automatically extracting it from a corpus [30].

6.4 Symbolic Approach (Rule Based Approach)

Natural Language Processing is a robustly symbolic activity. Words are symbols that set for objects and concepts in real worlds, and they follow the well meticulous grammar rules. Knowledge about language is clearly programmed in rules. Language is analysed at various levels to obtain information. On this obtained information several rules are applied to achieve linguistic functionality. As Human Language include rule-based reasoning, and it is supported well by representational processing. The rules are formed for every level of linguistic analysis.

6.5 Empirical Approach (Corpus Based Approach)

This approach is based on statistical analysis and data driven analysis of raw data. And which is in the form of a collection of machine readable text. The approach has been approximately since NLP began in the early 1950s. The empirical NLP replaced rule-based NLP in the last decade. Corpora are used as a basis of information about language and many techniques have discovered to enable the analysis of corpus data. Syntactic analysis can be achieved on the basis of statistical probabilities estimated from a training corpus. Lexical ambiguities can be resolved by considering the possibility of one or another interpretation on the basis of context. This approach shows a positive result. Several different symbolic and statistical methods have been in use,but
they are used to generate a larger information mining system.

Intermediate Representation Languages

Most current NLIDBs first convert the natural language question into an intermediate logical query, expressed in some interior meaning representation. The intermediate logical query expresses the meaning of the users question in terms of high level world notions, which are not dependent on the database structure. The logical query is

then translated to an expression in the databases query language, and calculated against the database. Because of the difficulties of directly translating a sentence into a general language, and calculated against the database.

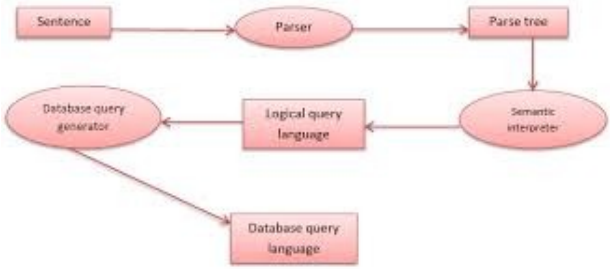


Figure : Intermediate Representation Language Architecture

database query languages using a syntax based approach, the intermediate representation systems were proposed. The idea is to translate a sentence into a logical query language first, and then further convert this logical query language into a general database query language. In this process there can be more than one intermediate meaning representation language

Conclusion

Natural Language Interface is the prominent research area in the field of artificial intelligence From the last few decades many systems were developed .on Natural Language Interface to Databases System which provide considerable improvement over another in terms of usability and accuracy. Now-a-days such interfaces are increasingly important on websites, particularly as access information from cell phones and PDA's and other small screen devices where GUI is less appealing. Different architectures are followed by different systems which distinguish their approach from one another. Also, with the revolutionary advancement in hardware technology and their processing speed, NLIDB system becomes more efficient, accurate and user-friendly

But the use of NLIDB system is not wide-spread and not a primary choice for interfacing to atabase. eason behind this problem is large no. of deficiencies in NLIDB system in order to understand a NL (natural language). In the future, work could be done to improve the linguistic coverage by the NLIDB system.

RREFERENCES

[1] I. Androutsopoulos, G.D. Ritchie, and P. Thanisch, Natural Language Interfaces to Databases - An Introduction, Journal of Natural Language Engineering 1 Part 1 (1995), 29–81.

[2] Mrs.Neelu Nihalani, Dr. Sanjay Silakari, Dr. Mahesh Motwani. "Natural language Interface for Database: A Brief review*", IJCSI International Journal of Computer Science Issues*, Vol. 8, Issue 2, March 2011 ISSN (Online): 1694-0814.

[3] A. Kaur, and P. Bhatiya, "Punjabi Language Interface to Database", M.tech thesis, Department of CSED, Thapar University, 2010.

[4] A. Shingala, P. Virparia ," Enriching Document Features for Effective Information Retrieval using Natural Language Query Interface", International Journal of IT, Engineering and Applied Science Research, ISSN: 2319- 4413, 2012.

[5] G. Hendrix, E. Sacrdoti, D. Sagalowicz, and J. Slocum, "Developing a natural language interface to complex data", ACM Transactions on Database Systems, Volume 3, No. 2, USA, 1978, pp. 105 – 147.

[6].Hendrix, G. (1977). The LIFER manual A guide to building practical natural language interfaces. SRI Artificial Intelligence Center, Menlo Park, Calif. Tech. Note 138.

[7].Woods, W., Kaplan, R. and Webber, B. (1972). The Lunar Sciences Natural Language Information System. Bolt Beranek and Newman Inc., Cambridge, Massachusetts Final Report. B. B. N. Report No 2378.

[8].Woods, W. (1973). An experimental parsing system for transition network grammars. In Natural language Processing, R. Rustin, Ed.,Algorithmic Press, New York. 2, USA, Pages 105 – 147

[9] D.L. Waltz., "An English Language Question Answering System for a Large Relational Database", Communications of the ACM, 21(7): (July 1978), pp 526– 539

[10] R.J.H., Scha., "Philips Question Answering System PHILIQA1", In SIGART Newsletter, no.61. ACM, New York, (February 1977)

[11].Warren, D., Pereira, F. (1982). An efficient and easily adaptable system for interpreting natural language queries in Computational Linguistics. Volume 8 pages 3 – 4.

[12]. Amble, T. (2000). Bus TUC – A Natural Language Bus Route Oracle. 6th Applied Natural Language Processing Conference, Seattle, Washington, USA

[13] B.J. Grosz, "TEAM: A Transportable Natural-Language Interface System", In Proceedings of the 1st Conference on Applied Natural Language Processing, Santa Monica, California, (1983), pp 39–45

[14]B.J. Grosz, D.E. Appelt, P.A. Martin, and F.C.N. Pereira, "TEAM: An Experiment in the Design of Transportable Natural-Language Interfaces", Artificial Intelligence, 32: (1987), pp 173–243

[15].P. Resnik, "Access to Multiple Underlying Systems in JANUS", BBN report 7142, Bolt Beranek and Newman Inc., Cambridge, Massachusetts, (September 1989)

[16].M. Templeton and J. Burger, "Problems in Natural Language Interface to DBMS with Examples from EUFID", In Proceedings of the 1st Conference on Applied Natural Language Processing, Santa Monica, California, (1983), pp 3–16
[17].C.D. Hafner, "Interaction of Knowledge Sources in a Portable Natural Language Interface", In Proceedings ofthe 22nd Annual Meeting of ACL, Stanford, California, (
1984) pp 57–60
[18].B.W. Ballard, J.C. Lusth, and N.L. Tinkham, "LDC-1: A Transportable, Knowledge based Natural Language Processor for Office Environments", ACM Transactions on Office Information Systems, 2(1): (January 1984), pp 1–25
[19].F. Damerau, "Operating statistics for the transformational question answering system", American Journal of Computational Linguistics, 7: (1981), pp 30–42
[20].B. Ballard and D. Stumberger, "Semantic Acquisition in TELI", In Proceedings of the 24th Annual Meeting of ACL, New York, (1986), pp 20–29
[21] A. Faraj EI-Mouadib, S. ZubiZakaria, A. Ahmed Almagrous and S. Irdess EI-Feghi "Generic Interactive Interface to Databases", International Journal of Computers issue 3, vol. 3, 2009
[22].Yunyao Li, Huahai Yang, and H.V. Jagadish, Nalix:an Interactive Natural Language Interface for Querying XML, SIGMOD (2005).
[23].Yunyao Li, Huahai Yang, and H.V. Jagadish, Constructing a Generic Natural Languaoge Interface fr an XML Database, EDBT (2006).
[24].Ana-Maria Popescu, Alex Armanasu, Oren Etzioni, David Ko, and Alexander Yates, Modern Natural Language Interfaces to Databases: Composing Statistical Parsing with Semantic Tractability, COLING (2004).
[25].Yuk Wah Wong, Learning for Semantic Parsing Using Statistical Machine Translation Techniques, Technical Report UT-AI-05-323, University of Texas at Austin, Artificial Intelligence Lab, October 2005.
[26] M. R. Joshi, R. A. Akerkar "Algorithms to Improve Performance of Natural Language Interface",
International Journal of Computer Science and Applications, vol. 5, No. 2, pp. 52-68, 2010.
[27] B. Sujatha, S. Viswanadha Raju and Humera Shaziya "A Survey of Natural Language Interface to Database Management System" International Journal of Science and Advance Technology", vol.2,no.6, June 2012.
[28] Himani Jain and Parteek Bhatia "Hindi Language Interface to Databases", Journal of Global
Research in Computer Science, vol.2, no.2, pp. 107-112, April 2011.
[29] Xu Yiqiu, Wang Liwei, Yan Shi "The Study of Natural Language Interface to Relational Database" in2nd Conference on Environmental Science and Information Application Technology, IEEE, pp.596-599, 2010.
[30] Ana-Maria Popescu, Alex Armanasu, Oren Etzioni, David Ko and Alexander Yates, "Modern
Natural Language Interfaces to Databases: Composing Statistical Parsing with Semantic
Tractability", COLING, 2004