INTERNATIONAL JOURNAL OF COMPUTER ENGINEERING & TECHNOLOGY (IJCET)

ISSN 0976 – 6367(Print)
ISSN 0976 – 6375(Online)
Volume 4, Issue 2, March – April (2013), pp. 379-385
© IAEME: www.iaeme.com/ijcet.asp
Journal Impact Factor (2013): 6.1302 (Calculated by GISI)
www.jifactor.com

IJCET

© I A E M E

"INTELLIGENT QUERY CONVERTER": A DOMAIN INDEPENDENT INTERFACE FOR CONVERSION OF NATURAL LANGUAGE QUERIES IN ENGLISH TO SQL

NeeluNihalani *
Associate Professor
Dept. of Computer Application,
UIT, RGPV,Bhopal, MP, India

Dr. Mahesh Motwani
Associate Professor
Dept. of CSE,
UIT, RGPV,Bhopal, MP, India

Dr. Sanjay Silakari
Professor & Head
Dept. of CSE,
UIT, RGPV, Bhopal, MP, India

ABSTRACT

Retrieval of information from databases using query language is always a professional and complex problem. This complexity reduces the usage of data existing in the databases. However , the databases will be more useful if a non-professional user can query the database using natural language. In this paper we present a domain Independent Natural language Interface-"Intelligent query converter" which converts Natural Language Query to SQL using Semantic matching technique. Semantic sets used for Semantic Matching is automatically generated by using WordNet.

Keywords: Natural Language processing; NLIDB; Sematic Matching; Domain independence; SQL.

1. INTRODUCTION

Natural language interfaces to databases (NLIDB) are systems that translate a natural language sentence into a database query like SQL [1]. Natural Language Interface to databases is a subset of Natural Language Processing that deals with NL inquiries against the databases. NLIDBs permits users to formulate queries in natural language, providing access to information without requiring knowledge of programming or database query languages. These systems takes a natural language (NL) query from a user as input and converts it to a SQL query based on the domain semantics and database schema. The SQL query thus retrieves appropriate data from the database and returns the output to the user.

Transformation of Natural Language query to SQL or some other database query language is the essential specialization of NLIDB. A truly natural language interface for database query is always an important technology because of its promises.

- First, such a technology could ease the user's burden of obtaining a complete set of specific information required by traditional querying methods - e.g., database objects and values needed in the clauses of SQL statements.
- Second, natural language query would be necessary for indirect or end users to use databases directly without going through some intermediary professionals.

The research of Natural Language interface to databases (NLIDB) has recently received attention from the research communities. The area of NLIDB research is still very experimental and these systems use only certain types of statements. When the systems are scaled up to cover larger domains, it becomes difficult transform these statements due to vast amount of information that needs to be incorporated.

Given below are some examples of Natural language queries that are often used by the users to query the database.

- Get orders for the supplier John Smith
- Get address where employeename is smith
- Get full details of all the suppliers
- Get supplier names for suppliers who supply Red part
- Get full details of products
- Get supplier details where part no is P2
- Select supplier name where partno = P2
- Get all sales manager

In our research we have developed an interface which converts these types of NL queries to database query language-SQL. We have classified these natural language queries into two types depending on the presence of Value Keywords and Conjunctive Clauses. We have constrained that conjunctive clauses are present in the middle of the query and not as the first and last element.

2. LITERATURE REVIEW

Most of NLIDB systems require the users to know the underlying schema to configure for different domains and databases [2]. These are called as domain dependent interfaces. Some of the most important NLIDBs that are domain dependent are SQ-Hal [12], MASQUE[1], Tiscover[3], Edite[4], EzQ[7]. These systems need to be configured for domain and database. Domain Independence has not been fully or adequately solved by existing NLIDBs.

Domain independent interfaces can be used with different domains and databases without intervention and avoids the tedious process of configuring the NLIDB for a given domain. EnglishQuery[10] and ELF[11] are thetwo commercially available domain independent systems. These systems automatically create a model collecting objects and extracting relationships from the database.

NLIDB Precise [9] gives 100% correct results for the questions that are 'semantically tractable'. This class is small subset of questions. The goal is to answer wide range of questions. Literature survey reveals that the percentage of correctly answered queries for domain dependent interfaces (i.e. approx. 69.4–96.2%) is higher than domain independent interfaces. This is mainly because they are limited to one domain. Researchers have suggested various methods for improving the success percentages domain independent interfaces

3. ARCHITECTURE OF INTELLIGENT QUERY CONVERTER (IQC)

This section describes the working of Intelligent Query Processor (IQC) which converts NL query into SQL. IQC is expected to be easy and can be easily configured for the given relational database. It is composed two modules: pre-processor and run-time processor.

3.1. The pre-processor

Pre-processor module automatically generates the domain dictionary by reading the schema of the database.The vital information that briefly describes the tables and fields in the database is organized into a metadata set (M). It uses WordNetto create semantic sets (S) for each table and attribute name. It also creates index files (I) for the values stored in each of the attributes of tables stored in database.These index files are used to match the tokens of user query. If a token of user query matches with any of the entries in the index file, corresponding table and field names are associated to these token and we have termed these tokens as value tokens.

For each database schema the pre-processor runs only once. It stores the primary keys and referential keys of the database. It generates production rules for joining of tables which can be edited by the system administrator.

3.2. Run time processor

Run-time Processor employs expression mapping, stop words removal and semantic matching techniques to convert the amorphous query into a structured SQL query. The shaped query is executed and the results are presented to the user. The proposed approach employs predefined training structures: expression mapping set, display-part keyword set and Conjunction set. The Expression mapping set contains the list of commonly used conditional clauses and their associated mathematical symbols. It acts as a look up table to locate the SQL defined mathematical operators. Display-part keyword set is a set of keywords like get, get me, show, list, show me etc.

4. CLASSIFICATION OF NATURAL LANGUAGE QUERIES

The goal of IQC is to answer a wider class of questions. We have identified the type of the user query depending on the presence of conjunctive clauses like where, whose, for, for which etc. and designed separate algorithms for each type so as to increase the performance of the system. We have constrained that *conjunctive clauses* does not occur as the first or last element in user query.

Three type of natural language queries are

4.1. Queries that contain Conjunctive Clauses

In these type of queries Conjunctive Clauses are present in the natural language query, but not as the first and the last element. For example: 'Get department of employee whose name is Smith'. In this query whose is Conjunctive Clauseand 'smith' is **Value Keyword.**

4.2. Queries that do not contain Conjunctive Clauses but contain Value Keywords

These type of queries do not contain Conjunctive Clauses but contains Value Keywords. For example 'Get Red part suppliers'. In this query no Conjunctive Clause is present but contains Value Keyword 'Red'.

4.3. Queries that do not contain Conjunctive Clauses but does not contain Value Keywords.

These type of queries do not contain Conjunctive Clauses nor Value Keywords. For example 'Get details of suppliers'. In this query no Conjunctive ClausenorValue Keyword is present.

5. TRANSFORMATION OF NL QUERY TO SQL SELECT STATEMENT

To transform NL query to SQL, firstly we identify the keywords in the NL query and thenthe type of the NL query is identified.

5.1. Queries that contain Conjunctive Clauses

Each user query of this type is likely to contain a display or subjective part which specifies the intended result, and the Criteria part which describes the condition or constraint. The subjective part is mapped with the SELECT clause and Criteria part is mapped with the WHERE clause of the SELECT statement. We assume that the phrase that defines the select clause always precedes the phrase that defines the where clause.In English, the words that separate these phrases are: whose, that, with, for which, for, such that, where, having etc. the Conjunction part which determines the SQL definition Clause. Steps for transformation of NL query to SQL are as follows:

- Identification of display part and criteria part.
- Expression mapping for criteria part.
- Stop word removal in both display part and criteria part.
- Identification of fields and tables in display part and criteria part is done using semantic mapping and distance measure techniques.

After following the above steps, fields, tables and condition are tracked down, which are then mapped to SELECT statement as shown in Figure 1

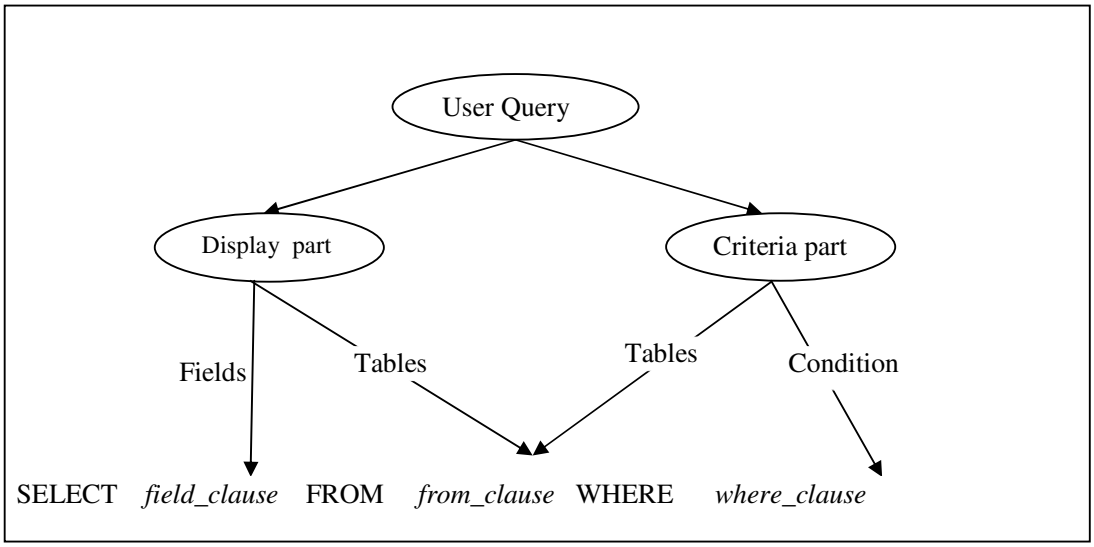

Figure 1: Mapping of Natural Language query to SQL SELECT statement

If more than one table is present in the from clause, joining condition for the tables is appended to the where clause using production table.

5.2. Queries that do not contain Conjunctive Clauses but contain Value Keywords

Totransform these types of queries to SQL, following steps are performed.

- Stop word removal
- Identification of Display Keyword: Display Keywords are the keywords like Show, get, display etc.
- Identification of tables and fields
- Identification of Keyword Token
- Expression Mapping
- If the next keyword of Display keyword is a field, then that field is appended to field clause of SELECT statement.
- Tables are appended to from clause.
- Condition is formed using value keyword and expression mapping and appended to where clause

SELECT statement is generated using field clause, from clause and where clause

5.3. Queries that do not contain Conjunctive Clauses and Value Keywords

To transform these types of queries following steps are performed.

- Stop word removal
- Identification of Display Keyword: Display Keywords are the keywords like Show, get, display etc.
- Identification of tables and fields.
- If the next keyword of Display keyword is a field, then that field is appended to field clause of SELECT statement.
- Tables are appended to from clause.

SELECT statement is generated using *field_clause*, *from_clause* and *where_clause*

6. EXPERIMENTATION

One of the most important domain independent interface is Microsoft English Query (EQ), which we have selected for comparison. For comparison, we have considered translation success i.e. the semantic equivalence between the natural language query and the SQL statement.We have tested IQC on the three database: Northwind, Geobase and Supplier-part database. Same questions were given as inputs to each of EQ and IQC systems. The outputs were compared to the outputs of correct SQL query and percentage of correctly answered queries for IQC and EQ are given in Table 3.

Table 3. Performance of IQC and EQ

System	Questions	Correct translation	Percentage
Northwind Database			
IQC	31	20	64.5
EQ	31	9	29.0
Supplier-Parts Database			
IQC	32	24	75
EQ	32	12	37.5
Geobase Database			
IQC	108	95	87.9
EQ	108	56	51.8

7. CONCLUSIONS

IQC system accepts an English language requests that is interpreted and translated into SQL command using semantic grammar technique. It automatically creates domain dictionary for the database and uses Wordnet for creation of semantic sets. This approach favors domain independence, since the NLIDB does not need to be manually configured with a set of keywords for carrying out specific actions. The design of the system had carried out with three major concepts in mind that is vital for any NLP system. The three major concepts are:

- The user query is in natural English language which is analyzed semantically and field token, table token and value tokens are identified.
- The construction of SQL statement that represent the users' query.
- The retrieval of the result that is required by the users' query.

The result of the number of experiments in the form of trials in a user friendly environment had been very successful and satisfactory.
We would like conclude this work by giving the advantages of the system in the following points:

- IQC is capable to answer common queries for any database.
- IQC is Portable i.e. able to be used against any database with the minimum amount of work.
- User friendly interface.

REFERENCES

[1] Androutsopoulos, I.,Ritchie,G., Thanisch, P.:MASQUE/SQL –An Efficient and Portable Natural Language Query Interface for Relational Databases. In: 6th Int. Conf. on Industrial & Engineering Applications of Artificial Intelligence and Expert Systems (1993)

[2] Copestake, A., Sparck-Jones, K.: Natural language interfaces to databases. The Knowledge EngineeringReview,225–249(1990)

[3] Dittenbach, M.,Merkl,W., Berger,H.: A natural language query interface for tourism information. In: Int. Conference onInformation and Communication Technologies in Tourism, pp.152–162.Springer,Heidelberg(2003)

[4] Filipe,P.,Mamede,N.:Databasesand Natural Language Interfaces.VJornadasdeBases de Datos.Valladolid(2000)

[5] Gonzalez, B.J.J., Pazos,R.R.A., Cruz,C.I.C., Fraire, H.H.J.: Issues in Translating from Natural Language to SQL in a Domain-Independent Natural Language Interface to Databases. In: Gelbukh, A., Reyes-Garcia, C.A. (eds.) MICAI 2006. LNCS (LNAI), vol.4293,pp. 922–931.Springer,Heidelberg(2006)

[6] Li,Y.,Yang,H.,Jagadish,H.:NaLIX: an Interactive Natural Language Interface for Querying XML. In: ACM SIGMOD, Baltimore (2005)

[7] Pazos, R.R.A., Gelbukh, A.F., González, B.J.J., Alarcón, R.E., Mendoza, M.A., Domínguez, S.A.P.: Spanish NaturalLanguage Interface for a Relational Database Querying System.In: Sojka,P., Kopeček,I., Pala, K.(eds.)TSD2002.LNCS(LNAI), vol.2448,pp. 123–130.Springer,Heidelberg(2002)

[8] Pazos, R., P´erez O. J., Gonz´alez, B. J., Gelbukh, A. F., Sidorov, G. and Rodr´ıguez, M. M. (2005) "A Domain Independent Natural Language Interface to Databases Capable of Processing Complex Queries". Lecture Notes in Artificial Intelligence, Vol. 3789, pp. 833-842.

[9] Popescu, A.M., Etzioni, O.,Kautz, H.:Towards a theory of natural language interfaces to databases. In: 8th International Conference on Intelligent user interfaces. Miami (2003)

[10] English Query,http://msdn2.microsoft.com/en-us/sqlserver/

[11] EnglishLanguage FrontEnd(VB-ELF),http://www.elfsoft.com

[12] SQ-HAL,http://www.csse.monash.edu.au/hons/projects/2000/Supun.Ruwanpura/

[13] Mousmi Chaurasia and Dr. Sushil Kumar, "Natural Language Processing Based Information Retrieval for the Purpose of Author Identification", International Journal of Information Technology and Management Information Systems (IJITMIS), Volume 1, Issue 1, 2010, pp. 45 - 54, ISSN Print: 0976 – 6405, ISSN Online: 0976 – 6413.

[14] Shambhavi.B.R and Dr.Ramakanth Kumar.P, "Current State of the Art Pos Tagging for Indian Languages – A Study", International journal of Computer Engineering & Technology (IJCET), Volume 1, Issue 1, 2010, pp. 250 - 260, ISSN Print: 0976 – 6367, ISSN Online: 0976 – 6375.

[15] Sanjeev Kumar Jha, Pankaj Kumar and Dr. A.K.D.Dwivedi, "An Experimental Analysis of MySQL Database Server Reliability", International journal of Computer Engineering & Technology (IJCET), Volume 3, Issue 2, 2012, pp. 354 - 371, ISSN Print: 0976 – 6367, ISSN Online: 0976 – 6375.