Translation of Spanish Statistics Expressions to SQL *

Ismael Esquivel Gámez, Ma. de los Ángeles Marrujo, Omar Pérez García

Universidad Veracruzana, México
iesquivel@uv.mx, angeles_marrujo@hotmail.com,
omarpg85@hotmail.com

Abstract. Grouping and aggregate data to obtain summarized values from a database is one of the most representative ways of generating information through a computer system. End users when requesting reports of this nature incorporate statistical expressions as *total, subtotal, suma, promedio, máximo, mínimo, etc.* (total, subtotal, sum, average, maximum, minimum, less, more, etc.), like in: ¿Cuál es el total de salarios pagados a los diseñadores? (What is the total wages paid to designers?). This paper presents a system for recognizing the statistics expressions in Spanish, found in natural language queries made to a database and converting them to SQL. It is a module of a natural language interface to database (NLIDB), still developing. The system handles three types of typical Spanish queries involving statistical expressions.

Keywords: NLIDB, SQL, Statistical expressions

1 Introduction

The natural language interfaces to databases (NLIDBs) are systems that allow end users to access a database introducing their queries in natural language [1]. Translate those queries to a database language like SQL, run the command and show the results. Some of these requests contain statistical expressions equivalent to processing the data stored in one or more tables, through the aggregated and grouping.

With SQL [2], data can be grouped and aggregated so that users can interact with them at a higher level of granularity that as they are stored in the database. The aggregated is carried out [3], by internal functions of SQL that allow you to calculate a single value from the rows in a result set or from the values returned by a arithmetic expression.

Although the commercial database engines support a wide of internal functions, in this paper we handle the functions of the ANSI SQL-92 standard, to extend the usefulness of the prototype presented.

* Work supported by the Mexican Government (SEP/PROMEP) under grant 103.5/09/4482

In a natural language query may appear multiple statistical expressions, creating some degree of complexity in its conversion. In this report, we describe a module of a Spanish NLIDB, named SNL2SQL, under construction. Converts the query issued by an end user, containing statistical expressions, to the corresponding SQL SELECT command.

2 Related work

According to the classification of queries in [4], consisting of six types, this paper focuses on solving the query type that requires special functions (number 5). For this reason, works in Spanish and English that translate this type to SQL, were reviewed.

In [5] handles a module, which translates from Spanish to an SQL query through four steps. In each, the query is subjected to several transformations by: The generation of an equivalent template semantic, the adaptation of the template data to those found in the database, the generation of the complete and final template and finally, the formulation of the corresponding SQL statement. In their work, the use of internal and external functions, it was proposed. For the first ones: *Unidades, diferencia_fecha, superlativo, cuenta* y *promedio* (*Units, date difference, superlative, average* and *count*), of which only the last three were translated directly into SQL aggregate functions and the other required a pre-conditioning.

To improve the process of setting up their NLIDB [6], they propose ontology to achieve greater acceptance and therefore, accessibility. The ontology defines categories to organize the concepts that have the context of the database. Among the highest hierarchy, were defined: Database elements, words, sets of synonyms (synsets) and functions. The latter were classified in: SQL Functions, User defined and call-link. For the former, groups of words or synsets, were equivalent to aggregate functions, semantically speaking.

In their methodology [7], they used natural language generation (NLG) as an alternative to remove the interpretation stage of queries. Those are constructed by the user through successive interactions with a natural language text, called feedback text. Changes to it are reflected in the semantic content of the query and are made in parts of the text enclosed by [], named anchors. In these, values can be changed, add aggregate functions, change or remove filters and order conditions; to generate the corresponding SELECT statement.

In their case [8] designed a graphical user interface and a grammar-driven phrases, like natural language grammar, called PDG. To enter a query and displaying the results, it follows a process of constructing sentences, targeted for users with no experience in data access and visualization. PDG manages aggregate functions with operators such as “number of”, “sum of”, and “avg of”; and whose argument is specified by [data], which can be used with "For Each data" phrase. The lowest and larger operators are used to obtain minimum and maximum. It also handles grouping facilities to generate queries shorter and clearer than SQL.

3 Proposed approach

As part of the implementation and after analyzing the corpus of queries made to the test database by students and faculty from IT area, three types of questions associated with statistical expressions were found. Table 1 shows the types that this proposal addresses and the SELECT commands associated.

Table 1. Types of queries and associated commands

Type	Query	SELECT command
1	¿Cuál es el total de salarios? (What is the total wages?)	Select sum(Empl.salary) From Empl
2	¿Cuántos empleados se tienen por cada departamento? (How many employees have every department?)	Select Empl.Workdept, count(*) From Empl Group by 1 Order by 2 Desc
3	¿Quién gana más salario de los empleados? (Who earns more salary for employees?)	Select Empl.Empno, Sum(Empl.salary) From Empl Group by 1 Order by 2 Desc

Although the third type can be solved with a subquery, we have found advantages with the approach proposed, among some, the orderly deployment of multiple records, allowing a more comfortable view of the information collected.

Likewise, were detected most common statistical expressions found in queries in Spanish, which are partially showed in Table 2. After a comprehensive analysis of questions and expressions, it was found that some of them created uncertainty, about the type of question that was being dealt (1 or 3), being regularly the following: *más, mayor, menos, menor* (*more*, *higher*, *less*, *smallest*), named special terms.

3.1 Assumptions and limitations

For the database, we used the assumptions that were proposed in [9], from C1 to C15 and for the queries made by the end user (A1 and A2), adding the management of mandatory queries ("Displays the maximum salary").

Managing expressions as an argument to a function is limited to those explicitly handle the attributes and processing, for example *"El total del salario menos la commission"* (*The total salary minus the commission*)

This first prototype does not handle the filtering of groups with the HAVING clause of the SELECT or relationship between tables (JOIN).

3.2 Previous configuration

Before using the module, the administrator needs to do certain tasks, among which are:

3.2.1 Registration of domain values

For attributes that handle a set of valid values that can take (domains), mainly for those with short values, for example SEX (‘F’, ‘M’), is necessary to capture the synonyms for these values: ‘MUJERES’, ’FÉMINAS’, ’FEMENINO’, ’HOMBRES’, ’VARONES’, ’MASCULINO’, etc. ('WOMEN', 'FEMINIST', 'FEMALE', 'MALE', 'MALE', 'MALE', etc.), in a file called Content Dictionary. It serves to facilitate processing, primarily the WHERE clause and thus, it encourages the use of terms, characteristic of the enterprise environment, where the prototype is implemented.

Table 2. Statistical Expression and aggregate functions

Expressions	Data type	SQL Function
Total, suma, sumatoria, subtotal, … (Total, sum, sum, subtotal…)	Numeric	SUM([DISTINCT] X)
Promedio, media, … (Average, average,…)	Numeric	AVG([DISTINCT] X)
Menos, menor, mínimo, más (reciente), …, (Less, smallest, minimum, Most (recent)…,)	All	MIN([DISTINCT] X)
Más, mayor, máximo, menos (viejo), …, (More, higher, maximum, Less (older), …,)	All	MAX([DISTINCT] X)
Cuántos, cantidad, número de, cuenta de,…, (How many, quantity, number of…, count of…,)	Values	COUNT([DISTINCT] X)
Cuántos, cantidad, número de, cuenta de,…, (How many, quantity, number of…, count of…,)	Tuples	COUNT(*)

3.2.2 Definition of grouping attributes

It is required to select the attributes that are grouped in a dictionary. Also a series of queries of the database can be run for purposes of automating the operation. For each attribute, is issued a command like this:

Select <table.attribute>, count(*) from <table> group by 1 having count(*) > 1

3.2.3 Election of the attribute to use by default

It is necessary to define for each table, a unique attribute that fully describe each tuple and that this is enough descriptive, like in [10], when the query does not express the grouping attribute, for example *¿Quién gana más salario?* (*Who earns more salary?*). For the present case, were chosen: Employee Number (Empl.Empno), department name (Dept.Deptname), project name (Project.Projname).

The last two definitions are stored in the domain dictionary, which is described below.

The prototype uses two dictionaries, the main also called "domain" and the secondary, called "content." The first was built from a synonym dictionary and the database metadata, based on the paper of Pazos *et. al* [8]. From the metadata is extracted by each table: name, description, and details of their columns (name, data type, size, permissibility of null values and its description. From the processing of the latter, are obtained nouns and partnership to the columns and tables, that in its description containing them or their synonyms. Also is obtained the equations representing the links between the tables to process queries with join. The dictionary content is updated similarly for columns that handle sets of valid values.

For this work, the synonym dictionary has been built manually using the online dictionary provided in [11], but the idea is to generate it automatically, in the future.

4 Main algorithm

The translation process is carried out following the next steps:

For all three types of questions:

1. Statistical expressions are identified in the question in any of the following cases:
    a. There are one or more and do not correspond to the special terms, in which case proceed to Step 2. For example the queries "*¿Cuál es el monto total de salarios para diseñadores y operadores?"* (*What is the total amount of salaries for designers and operators?* or *"¿Cuál*

es el salario máximo y salario mínimo de los gerentes?" (*What is the maximum wage and minimum wage for managers?*).

b. There are two or more and one of them is special term like in *"¿Qué departamento tiene el promedio de salarios más alto?"* (*What department has the highest average salary?*), in which case, the expression is ignored momentarily and move on to step 2

c. There is one and belongs to such terms, for which we continue to step 8. For example, *"¿Qué mujeres ganan más salario?"* (*What women earn more salary?*).

2. Found expressions define aggregate functions to use according the Table 1, eliminating such expressions of the query.
3. Determine the argument of the function, from the processing of words associated with the statistical expression, arranged in one of the following cases:
   a. Next, before or after, like in *"Salario mínimo"* (*minimum wage*).
   b. Connected with the preposition "*de"* (of) or "*de"* (of) and an article. Examples: *"Total de bonos"* (*Total of bonus*), *"promedio de la commission"* (*Average of the commission*).
   c. As a noun attached to "Cuántos" (How many), for example: *"¿Cuántos empleados…?"* (*How many employees…?*).
   d. As an arithmetic expression, denoted by the use of preposition (*"of"*) and conjunction (*"and"*), as resolved in [9]. An expression like *"El total de salarios y comisión"* (*Total of wages and commission*).
   e. Connected to words that involve the processing of non-repeated values, for example *"Únicos", "distintos", "diferentes"* (*Unique, different, etc.*), in which case the term DISTINCT is added to the argument as a prefix.

   Then it searches the domain dictionary to determine the pair table-attribute that will serve as an argument. The question is marked as "invalid" when it is not found or when the attribute handle a data type incompatible with the function. When there is more than one pair, the query is marked as "ambiguous". In the case of the COUNT function, the argument is the symbol (*), when asked the number of rows, therefore only needs to extracted from the question the name of the table to process, like in *"¿Cuántos trabajadores son varones?"* (*How many workers are male?*).
4. Nouns for the argument of the function and eventually, the terms of distinction, are removed from the question.

Question type 2:

5. All expressions involving grouping are identified, for example *"Por", "por cada", "en cada", "en", etc.* (*By, for every, every, in, etc*.), in the order they appear in the question.
6. Nouns associated with the previous expressions, are sought in the domain dictionary entries, only for those attributes previously marked as grouped, to determine the column(s) to use in the grouping clause. If not found, the question is marked as "invalid" and if there is more than one, as "ambiguous" Then both, nouns and expressions associated, are removed from the question.
7. Add the "order by" clause, indicating the numbers of columns that functions occupy in the SELECT clause with the DESCENDING option, to show higher figures, first.

Question type 3:

8. Given that there has not been defined the aggregate function, nouns associated with the special terms: *más, mayor, menos, menor* (*more, higher, less, smallest*) are processed as in step 3, to get the argument of the function, which will be one of the following:
    a. If the data type is numeric, SUM(X)
    b. If it is date or date and time, MIN(X)
9. It is determined the column(s) used (s) in the grouping clause, searching for the remaining nouns in the dictionary domain. If not found, for example: *¿Quién gana más salario de los empleados?* (*Who earns more salary for employees*?), the default defined column, described previously, is used. Then the *"order by"* clause is added, indicating the number of the column that the function occupies in the select clause, adding the Ascending or Descending option, according to the special term yet found in the text of the question. So Ascending option is associated with terms such as *menos, menor* (*less, smallest*), and DESCENDING with *más, mayor* (*more, highest*). However, there are nouns like *old* and *antique*, in which case, result in reverse.

Finally, for the three types:

10. Expressions of filtering results are processed, to build the WHERE clause, then removing those expressions.

5 Experimental Results

The test database used is shown in Figure 1. The corpus of natural language queries was formed after the presentation of the Database and explanation of the project to

teachers and students of the IT area. They were asked to write queries of the three types, and 27, 17 and 21, respectively were obtained, after eliminating similar.

From the corpus generated, which consisted of 65 requests, the results obtained are shown in Table 3. The conversion process took questions from a file and marked the non-translated queries with an “I” for invalid, “A” for ambiguous, and “X” for incorrect.

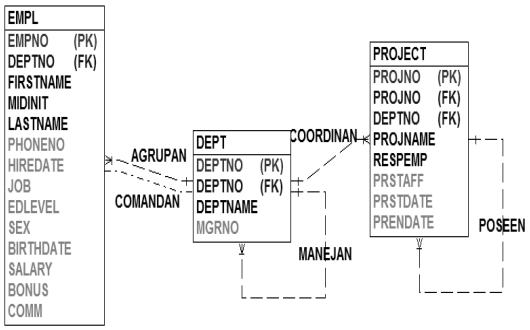

Fig. 1. Logical diagram of the test database.

It was found that all the "invalid" queries, contained terms that could serve as function argument, but not explicitly indicated attributes and processing. For example: *“¿Cuál es la edad promedio de los gerentes?”* (*What is the average age of managers?*), *“¿Cuál ha sido la duración promedio de los proyectos?”* (*What was the average duration of projects?*). The "Incorrect" were those that contained no statistical expression, like in *“¿Cuál es el bono anual que reciben los diseñadores?”* (*What is the annual bonus received by designers?*)

In general, the results were quite promising for being a first approximation, and the support for implicit arithmetic expressions, undoubtedly will improve the effectiveness. It is important to note that verification of the translation was done manually, because no connection has yet implemented with a commercial database engine.

Table 3. Results obtained

Query type	1	2	3	Total
Queries Formulated	27	17	21	65
“Invalid”	3	1	2	6
“Incorrect”	2	3	4	9
“Ambiguous”	0	0	0	0
Translated	22	13	15	50
Effectiveness	81.48%	76.47%	71.43%	76.46%

The programming language used for development was Rexx (REstructured eXtended eXecutor), a programming language developed at IBM by Michael Cowlishaw, of which there are numerous open source implementations available [12]. Version 3.4 of the Regina Rexx interpreter for Windows was used and was chosen because it has a wide set of functions, especially for the treatment of texts and dates. Another crucial advantage to use Rexx is that it is a multiplatform language. It can, with slight modifications, run on any computer under any operating system and work.

5 Conclusions

The module presents a very specific approach to treat statistical expressions that are translated into SQL functions, grouping and order clauses. It is contemplated to test it with real-applications databases containing a larger number of attributes of grouping and aggregated.

The structure of queries from the corpus formed has been analyzed and found three types that predominate in reporting requirements. Such questions are representative of the reports requested by the executives and middle management, and therefore, automatically translating is a key factor in development of the SNL2SQL NLIDB. However, improvements are needed. One of the areas to improve is carry out the translation of more complex queries containing expressions of group filtering and the relationship between two or more tables. Another of the improvements is the automation of both, the generation of the Dictionary of Synonyms and the verification of the conversion.

There were several queries that were related to variables not considered in the database, but with a more refined processing, can be obtained. It is possible to enrich the tasks of the previous configuration and allow the entry of these variables, the associated database attributes, system variables and arithmetic operations. For that reason, the development of an enhanced version, to improve effectiveness, is a priority line of work.

References

1. Androutsopoulos, I., Ritchie, G., Thanisch, P.: Natural language interfaces to databases - An introduction. Journal of Language Engineering, 1(1):29-81, 1995.
2. Beaulieu, A.: Learning SQL, Second Edition, O'Reilly Media, Sebastopol (2009).
3. Viescas, J. L., Hernandez M. J.: SQL Queries for Mere Mortals: A Hands-On Guide to Data Manipulation in SQL, Second Edition. Pearson Education, Massachusetts (2008).
4. González J., Pazos, R. A., Gelbukh, A., Sidorov, G., Fraire, H., Cruz, C.: Prepositions and conjunctions in a natural language interfaces to databases. In: Thulasiraman, P., et al. (eds.) ISPA 2007 Workshops. LNCS Vol. 4723, pp. 173-182, Springer, Heidelberg (2007)
5. Colás, J.: Estrategias de incorporación de conocimiento sintáctico y semántico en sistemas de comprensión del habla continua en español. Estudios de lingüística española (ELIEs). Publicación periódica de monografías sobre lingüística española (2001), http://elies.rediris.es/elies12/index.html
6. Zarate, J. M., Pazos, R. A., Gelbukh, A., Pérez J.: Improving the customization of natural language interface to databases using an ontology. In: Gervasi, O, Gavrilova, M. (eds.) ICCSA 2007. LNCS Vol. 4705, Part. I, PP. 424-435, 2007.
7. Hallett, C., Hardcastle, D.: Towards a bootstrapping NLIDB system. In: Kapetanios, E., Sugumaran, M. Spiliopoulou (eds.) NLDB 2008. LNCS, Vol. 5039, pp. 199-204, Springer, Heidelberg (2008)
8. Lee, S. Y., Neumann, U.: A Phrase-Driven Grammar System for Interactive Data Visualization. In: Börner, K., Gröhn, M., Park, J., Roberts J. C. (eds.) Visualization and Data Analysis 2008. SPIE Int. Soc. Opt. Eng. Vol. 6829 (2008).
9. Pazos R., Pérez J., González J. J., Gelbukh A., Sidorov G. y Rodríguez M. 2005. "A Domain Independent Natural Language Interface to Databases Capable of Processing Complex Queries". MICAI 2005: Advances in Artificial Intelligence. MICAI 2005: 833-842
10. Stratica, N., Kosseim, L., Desai, B. C.: Using semantic templates for a natural language interface to the CINDI virtual library, Data & Knowledge Engineering, v.55 n.1, pp. 4-19, October 2005 doi: 10.1016/j.datak.2004.12.002
11. Rodríguez S. y Carretero J. 2008. COES: Herramientas para Procesamiento de Lenguaje Natural en Español. http://www.datsi.fi.upm.es/~coes/interactivo/sinonimos.cgi
12. Mertz David. 2004. Rexx for everyone: Scripting with Free Software Rexx implementations. http://www.ibm.com/developerworks/library/l-rexx.html