DOMAIN SPECIFIC LANGUAGE CHART REPRESENTATION OF DATA

Ina BOTNARU, Dumitru CREȚU, Ecaterina EJOVA, Marius SARMANIUC

Technical University of Moldova

Abstract: *This paperwork presents a case study of the Domain-Specific Languages and the performance of the top Database Management System. The Structured Query Language is the main programming language designed to manage data stored in database systems. A Domain-Specific Language will be designed for chart representation of data. For developing the language, it was analyzed and created the context free grammar as the first step in building its architecture. In this article, also is presented an example of algorithm and, based on it, the syntactic and semantic tables are referred.*

Keywords: *Domain Specific Language, database, charts, Structured Query Language.*

Introduction

Digital world is growing very fast and requires efficient, thoughtful decisions from companies, countries in order to maintain or improve positions on modern markets and international economics. Those decisions can be made using well-made statistics and its analysis, that allows to find sources of many complex processes and activities. Consequently, sources can be improved, if they give positive impact, or removed, if they have unwelcome influence [1].

Statistics requires availability of considerable size of data, that can be saved and processed by complex Structed Query Language (SQL) databases. The most efficient way to analyze this data is to represent it via different charts and diagrams. However, SQL doesn't have solution of the problem. Wherefore, any analysis requires usage of additional software and copying of recorded data to programs that have such functionality.

Database are the nerve center of our economy. Every piece of personal information is stored there. Many applications and servers use databases, which no doubt belong alongside operating systems as key middleware and infrastructure that engineers must think about.

MySQL is to relational databases what Linux is to operating systems, but Microsoft SQL Server (MSSQL) works greater with Microsoft-based systems. For this reason, was chosen MSSQL as main platform for this software. It is essential to know which platforms are submitted by users, in view of the fact that it is obligatory to consider separate platforms can work superior only with specific type of SQL.

1. Principles of work

It is vital to specify principles of work for software in order to make some general notes for interaction of user with this software. There are the five main types of charts that will be used to present data: multi-axis charts, line charts, pie charts, bar graphs and bar histograms.

Using of the above described diagrams and charts can be functional in deep analysis of data. The problem is the user will have a desire to make changes for standard view of charts. For such purposes user needs definite tools to display data graphically:

- Xaxis - parameter that sets name of Xaxis for chart;
- Yaxis - parameter that sets name of Yaxis for chart;
- Zaxis - parameter that sets name of Zaxis for chart (available only for multi-axis chart);
- color - change color of figures in charts and diagrams;
- grid - sets background grid for all chart;
- size - changes size of chart.
- update - update the chart choosing either add a column with data or delete one;
- delete - delete the whole chart;
- alias - create the alias which will help the user to draw a different chart based on the same data.

The user is given the opportunity to manipulate charts. Below is the syntax in MSSQL:

DRAW <nameOfGraph>
FOR (SELECT <nameOfColumn> FROM <nameOfTable>)
WHERE <condition>

The WHERE clause is used to filter records, to extract only those records that fulfill a specified condition:

Xaxis = <nameOfAxisX> AND Yaxis = <nameOfAxisY> AND color = 'typeOfColor' AND size = <someSize> AND grid [ON | OFF]

The syntax is similar to SQL script style. As a result, it will be easy for the user to understand the syntax and keywords that can be used. Creating a convenient syntax is necessary, as it will be used by ordinary users without programming knowledge or programmers who work only with SQL. By limiting the features and conditions that can be specified in WHERE clause, the development of the language becomes easier.

2. Grammar via Semantics

It should be emphasized that the number of non –terminal and terminal symbols is a basic description measure of Chomsky grammars and of their extensions.

$$S = \{S\}$$

$$V_N = \{< typeOfGraph >, < forStatement >, < selectStatement >, < conditionalStatement >, < nameOfColumn >, < fromStatement >, < nameOfTable >, < condition >, < nameOfAxes >, < nameOfAxe >, < color >, < size > < grid >\}$$

$$V_T = \{DRAW, FOR, SELECT, FROM, WHERE, AND, X, =, Y, on, off, 1,2,3,4,5,6,7,8,9,0, a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, x, w, y, z, red, blue, green, \varepsilon, line, pie$$

The main part in creating the grammar is Production, which represents a set of rules which connects starting points, terminal and non-terminal symbols. These rules will limit the language, accepting just a specified type of input.

$$P = \{S \rightarrow DRAW < typeOfGraph > < forStatement > |$$

$$< forStatement > \rightarrow FOR\ (< selectStatement >) < conditionalStatement >$$

$$< selectStatement > \rightarrow SELECT < nameOfColumns > < fromStatement >$$

$$< fromStatement > \rightarrow FROM < nameOfTable >$$

$$< conditionalStatement > \rightarrow WHERE < condition >$$

$$< condition > \rightarrow < nameOfAxes > | < nameOfAxes > < optional >$$

$$< optional > \rightarrow < color > | < size > | < grid > | < optional > | \varepsilon$$

$$< nameOfAxes > \rightarrow X = < nameOfAxe > AND\ Y = < nameOfAxe >$$

<nameOfAxe> → a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|x|w|y|z|<nameOfAxe>|ε

<nameOfColumn> → a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|x|w|y|z| <nameOfColumn>|ε

<nameOfTable> → a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|x|w|y|z|<nameOfTable>|ε

<typeOfGraph>→multiaxes|line|pie|bar

<color>→red|blue|green

<size>→1|2|3|4|5|6|7|8|9|0|<size>| ε

<grid>→on|off

For analyzing language from semantic and syntactic point of view, below is an example of source code. It represents a general template for the language, starting with comments (that can be written in any part of the program), then specifying what type of char to draw and for what data. Due to the SQL features of manipulating data, every combination of columns that satisfy the conditions can be use by the program. Also, the parameters like name of axes, size, colors, using or not grid can be specified at the end. The source code will be analyzed in Table 1 Type of Tokens, from syntactical/semantical point of view, using tokens.

```
/* Analyzing students marks */
DRAW bar_graph
FOR (SELECT student_id, average_mark FROM studenti)
WHERE X = "UTM Students" AND Y = "Semester Mark" AND color = "red" .
```

Table 1. Type of Tokens.

Code	Tokens	Lexeme
1	keyword	DRAW, FOR, SELECT, FROM, WHERE, AND, X, Y, color
2	constants	UTM Students, Semester Mark, red
3	comments	/* Analyzing students marks */
4	identifier	bar_graph, student_id, average_mark, studenti
5	symbol	,) (= “ ”

The Table 2. Syntactic Analysis is required in which was analyzed each element of the source code below based on the token code. The process is repeated until the last element is analyzed.

Table 2. Syntactic Analysis.

3	1	4	1	5	1	4	5	4	1	4	5	1	1	5	5	2	5	1	1	5	5	2	5	1	1	5	5	2	5

A token represents a category of terminal symbols. As the Table 2. Syntactic Analysis shows, there are different constants, which should be presented in a specific table each (fig. 1).

Cod	Keyword
1	DRAW
2	FOR
3	SELECT
4	FROM
5	WHERE
6	AND
7	X
8	Y
9	color

Cod	Constants
1	UTM Students
2	Semester Mark
3	red

Cod	Symbol
1	,
2	(
3	)
4	=
5	“
6	”

Cod	Identifier
1	bar_graph
2	student_id
3	average_mark
4	studenti

Cod	Comment
1	/* Analyzing students marks */

Fig. 1. Tables for each token

Semantic Table: To create a semantic table, user need to look at the code for each element from the table above. For example, keyword SELECT is located in the 6th box of Syntactic table, but in “Keyword” table it has code 3. Accordingly, in the table below, code 3 is inserted in 6th place (yellow color).

Table 3. Semantic Analysis.

1	1	1	2	2	3	2	1	3	4	4	3	5	7	4	5	1	6	6	8	4	5	2	6	6	9	4	5	3	6

Conclusions

The Domain Specific Language will allow to create the most effective forms of various types of charts and diagrams. These charts can be used to analyze data that contained in one or another form in the SQL database. The great advantage is that these charts and diagrams can be edited by the user himself, so he can select the configurations interest. In order to develop it, starting from creating the basic grammar, a semantic and syntactic analyses was made to identify the most important parts of the language.

References:

1. Aho A.V., Ullman J. D. *The Theory of Parsing Translation and Compiling*, vol.1, Prentice-Hall, Inc. Upper Saddle River, NJ, USA, 1972.
2. Berry G., Sethi R. 'FROM REGULAR EXPRESSIONS TO DETERMINISTIC AUTOMATA', *Theoretical Computer Science* , 48(117-126), 1986, pp. 117 - 126.
3. Bogdan Walek, Cyril Klimes. 'A methodology for Data Migration between Different Database Management Systems ', *International Journal of Computer, Electrical, Automation, Control and Information Engineering*, 6(5), 2012, pp. 536-54.