Pedro Santos Rodrigues

Bachelor in Computer Science

Accelerating SQL with Complex Visual Querying

Dissertation plan submitted in partial fulfillment
of the requirements for the degree of

Master of Science in
Computer Science and Informatics Engineering

Adviser: Teresa Romão, Assistant Professor,
NOVA University of Lisbon

Co-advisers: Rui Nóbrega, Assistant Professor,
NOVA University of Lisbon

Tiago Simões, Principal Product Designer,
OutSystems

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

January, 2020

Abstract

The dissertation must contain two versions of the abstract, one in the same language as the main text, another in a different language. The package assumes that the two languages under consideration are always Portuguese and English.

The package will sort the abstracts in the appropriate order. This means that the first abstract will be in the same language as the main text, followed by the abstract in the other language, and then followed by the main text. For example, if the dissertation is written in Portuguese, first will come the summary in Portuguese and then in English, followed by the main text in Portuguese. If the dissertation is written in English, first will come the summary in English and then in Portuguese, followed by the main text in English.

The abstract shoul not exceed one page and should answer the following questions:

- What's the problem?
- Why is it interesting?
- What's the solution?
- What follows from the solution?

Keywords: Keywords (in English) ...

Resumo

Independentemente da língua em que está escrita a dissertação, é necessário um resumo na língua do texto principal e um resumo noutra língua. Assume-se que as duas línguas em questão serão sempre o Português e o Inglês.

O *template* colocará automaticamente em primeiro lugar o resumo na língua do texto principal e depois o resumo na outra língua. Por exemplo, se a dissertação está escrita em Português, primeiro aparecerá o resumo em Português, depois em Inglês, seguido do texto principal em Português. Se a dissertação está escrita em Inglês, primeiro aparecerá o resumo em Inglês, depois em Português, seguido do texto principal em Inglês.

O resumo não deve exceder uma página e deve responder às seguintes questões:

- Qual é o problema?
- Porque é que ele é interessante?
- Qual é a solução?
- O que resulta (implicações) da solução?

E agora vamos fazer um teste com uma quebra de linha no hífen a ver se a LaTeX duplica o hífen na linha seguinte...

zzzz zzz zzzz zzz zzzz zzz zzzz zzz zzzz zzz zzzz zzz zzzz zzz zzzz zzz zzzz comentar--lhe zzz zzzz zzz zzzz

Sim! Funciona! :)

Palavras-chave: Palavras-chave (em Português) ...

Contents

Chapter 1

Introduction

This chapter will introduce this thesis, starting with a brief contextualization about the actual technological framework, followed by a description of the motivation behind it. In addition, will be presented an overview of the problem, such as expected contributions and the structure of the document.

1.1 Context

Nowadays, Information and Computer Systems have been dominating data processing. These fields are genrators of positive impact the to personal and business areas presenting a prominent place on any information system. However, well before any electronic system, since the people started to count or write, they have needed to store pieces of information. [1] Thenceforth, by many years, people have used physical information, like paper, to store data, however, with the digital transformation, these resources are less and less used.

In the decades of the 1960s, Database Management Systems (DBMS) arose, and later at 1970s new management systems that use relational models, designated as Relational Database Management Systems (RDBMS). Moreover, first Data Query Languages (DQL) appeared, like SQL [4] which was considered by the ANSI [1] and ISO [2] as the standard query language [5]. These technological evolutions have improved the effectiveness and the efficiency of the querying process. However, to find more specific and complex information on databases a higher degree of DQL understanting was required. Thus, if from one side the technological evolution and the digital transformation have optimized

[1] American National Standards Institute
[2] International Organization for Standardization

the data querying process, only a subset of people could use these powerful querying technologies.

Visual Query Systems (VQSs), defined by Catarci, *et. al.* [2] as "systems for querying databases that use a visual representation to depict the domain of interest and express related requests", are used to mitigate some problems already referred. These systems use different visual representations and interaction strategies to make database queries, using a more intuitive visual approaches, instead of using textual languages, which are more difficult to learn mainly for people without programming base knowledge. In addition, even if it is not mandatory to be considered a VQS, some systems have also data visualization features which can be useful to view the query result, even as the possibility to manage the database schema in a visual way too.

However, usually, these visual languages are associated as more useful to naive users, while textual languages are associated to more expert users. Conversely, some studies have revealed that visual languages might be convenient to the expert users too. For example, the comparison made by Catarci and Santucci [3] concludes that diagrammatic languages can reduce the error rate of the queries, in comparison with those made in a textual language by expert users. These results imply that even expert users make mistakes in simple queries (e.g. they may not remember the name of the tables or the precise syntax of some language expressions). Thus, it is important to analyse how those languages could be used to optimize the querying process not only to the users with a low experience level but also for highly experienced users.

Nonetheless, the widely users' background and the diversity of the data domain made that a lot of these systems need to be modelled to a specific domain. So, it is very difficult to find an global integrated solution that covers necessities of all users on all domains being this personal or professional.

1.2 Motivation

Low-Code Development is a recent development paradigm which seeks to reduce the time and effort spent in tasks that will not have a significant impact on the final product outcome. Just as the rise of high-level languages, APIs, and third-party infrastructures have provided to developers to be more productive and spend more effort on most value sections of the software they produce. Also, the goal of the Low-Code is to extend this reasoning, using visual IDEs, connectors between components and lifecycle managers, to put away some concerns as infrastructure and re-implementation of patterns and free up people to think better on things that could be more relevant to their objectives [8].

The goal of this project is to analyse and improve the OutSystems Platform data querying component that can be used to create relational database queries through drag and drop interactions and simple configurations beyond visual interactions and multiple drag and drop features. This platform aim is to provide a complete application development environment where users with different development backgrounds can build, deploy and

manage their applications above good practices and state-of-the-art technologies even without having to worry about those details.

However, conversely of No-Code approaches, the development through low-code systems give the possibility to use low-level code, written in textual languages like Java, .NET or SQL, in order to increase the extensibility and the power of low-code solutions [9]. Therefore, it is provided to users an alternative to perform their requests that are not supported by the low-code visual approaches. However, if the visual languages of low-code platforms are more robust, responding more thoroughly to users' requirements, there is a diminished demand to resort on these textual programming languages which are high error-prone and have a worse learning curve, requiring also, on multiple situations, previous coding experience.

Furthermore, as mentioned by Amaral *et. al.*, web and mobile applications produced on OutSystems' technology, have proven not only an increase on quality [6], but also allowed developers to increase 10.9x on productivity, when compared with IT Industry standards that do not use these rapid software development solutions [7]. These results reinforces the importance of this matter, the improvement of visual languages used on this platform.

1.3 Problem Description

The OutSystems platform provides a visual query language that allows users to retrieve data from databases by simple processes. Besides, with this language, it is possible to perform some naive operations that are usually supported by textual Data Querying Languages (DQL), namely join, filter, sort and group operations.

Currently, the OutSystems' solutions have been applied on digital transformation processes in multiple industries that lead with high quantities of data, in order to accelerate the application development processes, unlocking its value and growth.

To this extent, the already implemented querying tool can not deal with a set of scenarios, because it might be not accurate when the domain has a lot of tables involved, or does not support essential advanced constructors. Are part of these constructors not supported clauses like IN, NOT IN, EXISTS, NOT EXISTS, DISTINCT and the possibility to use subqueries.

Under the above mentioned circumstances, the goal of this project is to design and evaluate a new and more powerful Language and User Experience that allows developers to do all these complex data queries in a very easy way without using SQL.

1.4 Research Questions

- What query features are supported by the existing OutSystems visual query language?

- Why do OutSystems developers use SQL to perform database queries?
- Why are the queries that can be built visually are written though SQL?
- What are the main causes that users pointed out to use SQL?
- What users are more unsatisfied with the current provided approach to retrieve data?
- Can we enable OutSystems developers to easily do all kinds of database queries without ever using SQL?

1.5 Main Expected Contributions

This work aims to provide a set of contributions not only as a scientific research, but also with a perspective of creating the most value added to OutSystems. Thus, are presented below a summary of the main expected contributions for this project:

- A synthesization of the design concepts, including relevant interaction and conceptual models, usability definitions, guidelines and principles, and a description of the processes to evaluate the human-computer interaction in the context of a software analysis;
- Provide a state-of-the art about what are the most significant Visual Query Systems, presenting also a comparison of visual representation techniques and interaction strategies used, as well as other important features proper for this study;
- A description of the analysis made to identify what are the most impactful problems regarding the existing visual query language, which includes user interviews and data analysis;
- Design and implementation of a new graphical user interface prototype that tries to improve the existing solution to make queries visually. This prototype expects to fix some predominant problems selected as the most relevant to solve;
- An usability evaluation of the prototype developed through the use of user tests, confronting these results with the first obtained.

1.6 Structure

The remaining chapters of this thesis are organized as follows:

- Chapter 2 - Related Work: presents a short description of the OutSystems Platform, as well as a description of the existing techniques that already exist on the context of the main topic of this thesis - data visualization and visual querying. Besides,

other commercial applications will be enumerated which can have relevant content for this study;

- Chapter 3 - Proposed Solution: describes the proposed solution, starting with a requirement analysis, followed by a more detailed explanation about the problem, and finally with a definition of the development scope to understand what problem will be tackled on detail;
- Chapter 4 - Work Plan: includes a planning of the inherent total work. Thus, will be presented an overview of the tasks that were done on this dissertation plan, together with the preview of the work which will be the focus of the second phase of the thesis, the elaboration.

Chapter 2

Related Work

On this thesis, it is pretended to apply Human-computer interaction (HCI), Data Visualization and Visual Querying concepts, techniques and technologies to improve a Visual Querying Feature of the OutSystems Low-code Development Platform. Thus, in this chapter, will be presented the results of a study that analysed what is the Low-code development platform background and its actual situation, as well as what are the techniques and technologies which already exist, including some comparison between them. Finally, will be enumerated what products, technologies and tools exist on other commercial applications which can be related with the topics of this thesis.

It is necessary to add more sections to describe some key concepts about user experience testing and analysis.

2.1 OutSystems Background

The entirety of this thesis has the aim of improving the OutSystems Platform, so it is very important to understand what is that product and what can be developed with it. This section provides an overview of this Low-code Development Platform, describing its value proposal, and its technological goals and approaches. In addition, it will be used some images to illustrate some relevant aspects of the platform.

The next sections depend on the research done.

2.2 Data Visualization

2.3 Visual Queries

2.4 Data User Exprerience and Expressiveness

2.5 Technologies and Commercial Applications

Such as the techniques research, it is also very important to search what are the technologies related with the subjects of this thesis that already exist, as this knowledge can be very important to the concept of a solution proposal.

Furthermore, in any research, the academic content should not be the only taken into account, because sometimes the knowledge does not evolve only in the research centres but also in the companies. Since this thesis is made to improve a company product, the latter assertion has additional strength, so also, will be introduced commercial applications which can be useful to all the entirety of this process.

Chapter 3

Proposed Solution

This chapter presents the solution proposed to the problems presented, that includes a description of the process realized to understand why the people use SQL to made queries instead of Aggregates, such as Personal Interviews and a Quantitative Analysis of the queries which customers ran on the cloud. Furthermore, this chapter presents what is the scope of the project, so it will be explained what problems will be tacked in detail.

3.1 Requirements Analysis

As referred above, this section presents the results of the analysis made to understand why developers use SQL to make queries instead of Aggregates through its Visual Querying Features.

3.2 Proposed Implementation

Following, will be detailed the pretended project, indicating all the problems identified and all the approached that will be adopted.

3.3 Scope Definition

In this section, will be presented what were the decisions made of which problems will be addressed in this thesis, once the initial problem presentation had a wide scope and it was concluded that it's not possible to resolve all the Aggregates expressiveness and experience problems in this project.

Chapter 4

Work Plan

This chapter includes a planning of the total work from beginning to end. Thus, will be presented in a chronological way all the work realized in the preparation phase. Furthermore, after all the analysis realized on this dissertation plan, will be presented an expected plan to the remaining time until the final of this thesis.

Bibliography

[1] K. L. Berg, T. Seymour, and R. Goel. "History of databases." In: *International Journal of Management & Information Systems (IJMIS)* 17.1 (2013), pp. 29–36.

[2] T. Catarci, M. F. Costabile, S. Levialdi, and C. Batini. "Visual query systems for databases: A survey." In: *Journal of Visual Languages & Computing* 8.2 (1997), pp. 215–260.

[3] T. Catarci and G. Santucci. "Diagrammatic vs textual query languages: a comparative experiment." In: *Working Conference on Visual Database Systems*. Springer. 1995, pp. 69–83.

[4] D. D. Chamberlin and R. F. Boyce. "SEQUEL: A Structured English Query Language." In: *Proceedings of the 1974 ACM SIGFIDET (Now SIGMOD) Workshop on Data Description, Access and Control*. SIGFIDET '74. Ann Arbor, Michigan: Association for Computing Machinery, 1974, 249–264. ISBN: 9781450374156. DOI: 10.1145/800296.811515. URL: https://doi.org/10.1145/800296.811515.

[5] J. Gehrke and R. Ramakrishnan. *Database management systems*. McGraw-Hill, 2003.

[6] H. Henriques, H. Lourenço, V. Amaral, and M. Goulão. "Improving the Developer Experience with a Low-Code Process Modelling Language." In: *Proceedings of the 21th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*. MODELS '18. Copenhagen, Denmark: Association for Computing Machinery, 2018, 200–210. ISBN: 9781450349499. DOI: 10.1145/3239372.3239387. URL: https://doi.org/10.1145/3239372.3239387.

[7] OutSystems. *OutByNumbers - Benchmark Overview Repor*. Tech. rep. 2013.

[8] M. Revell. *What Is Low-Code?* 2020. URL: https://www.outsystems.com/blog/what-is-low-code.html (visited on 01/24/2020).

[9] C. Souther. *Low-Code vs. No-Code: What's the Real Difference*. 2019. URL: https://www.outsystems.com/blog/posts/low-code-vs-no-code/ (visited on 01/25/2020).