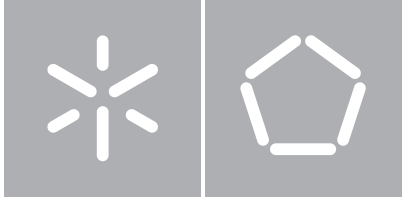
Universidade do Minho
Escola de Engenharia

João da Cunha Martins

Um Explorador Visual para Sistemas de Bases de Dados Relacionais

Agosto de 2014



Universidade do Minho
Escola de Engenharia
Departamento de Informática

João da Cunha Martins

Um Explorador Visual para Sistemas de Bases de Dados Relacionais

Dissertação de Mestrado
Mestrado em Engenharia Informática

Trabalho realizado sob orientação de

Professor Doutor Orlando Manuel Oliveira Belo

Agosto de 2014

Anexo 3

DECLARAÇÃO

Nome

_____

Endereço electrónico: _____ Telefone: _____ / _____

Número do Bilhete de Identidade: _____

Título dissertação □/tese □

_____

_____

_____

Orientador(es):

_____

_____ Ano de conclusão: _____

Designação do Mestrado ou do Ramo de Conhecimento do Doutoramento:

_____

Nos exemplares das teses de doutoramento ou de mestrado ou de outros trabalhos entregues para prestação de provas públicas nas universidades ou outros estabelecimentos de ensino, e dos quais é obrigatoriamente enviado um exemplar para depósito legal na Biblioteca Nacional e, pelo menos outro para a biblioteca da universidade respectiva, deve constar uma das seguintes declarações:

1. É AUTORIZADA A REPRODUÇÃO INTEGRAL DESTA TESE/TRABALHO APENAS PARA EFEITOS DE INVESTIGAÇÃO, MEDIANTE DECLARAÇÃO ESCRITA DO INTERESSADO, QUE A TAL SE COMPROMETE;

2. É AUTORIZADA A REPRODUÇÃO PARCIAL DESTA TESE/TRABALHO (indicar, caso tal seja necessário, nº máximo de páginas, ilustrações, gráficos, etc.), APENAS PARA EFEITOS DE INVESTIGAÇÃO, , MEDIANTE DECLARAÇÃO ESCRITA DO INTERESSADO, QUE A TAL SE COMPROMETE;

3. DE ACORDO COM A LEGISLAÇÃO EM VIGOR, NÃO É PERMITIDA A REPRODUÇÃO DE QUALQUER PARTE DESTA TESE/TRABALHO

Universidade do Minho, ___/___/_____

Assinatura: _____

Aos meus Pais e ao meu Irmão

Agradecimentos

Quero deixar aqui expressos os meus agradecimentos a todos aqueles que me acompanharam e apoiaram nesta jornada, de crescimento pessoal, intelectual e que contribuíram para que fosse possível, concluir este trabalho de dissertação com sucesso:

- Ao meu orientador Professor Doutor Orlando Manuel Oliveira Belo, pela disponibilidade, incentivo e criticas ao longo deste trabalho, que fizeram com que este trabalho fosse mais além do proposto, mas também pela confiança transmitida.
- Ao Francisco Dourado, pela disponibilidade em todos os momentos para me ajudar com a implementação da ferramenta.
- À Ana Oliveira, ao André Carvalho, ao Eduardo Fonseca, ao Francisco Dourado e ao Miguel Dias pelo apoio e pelos anos de verdadeira amizade.
- Ao meu Irmão Carlos Martins pelo apoio incondicional, pela força que sempre me deu, pela disponibilidade para me ouvir mesmo quando não era uma companhia agradável e pela confiança que me deu.
- À Diana Marques pelo apoio incondicional, por não me deixar desistir e principalmente pela paciência para me aturar.
- A todos os meus colegas e amigos que sempre me apoiaram, incentivaram e me permitiram momentos de descontração nas horas boas e nas mais difíceis.
- A todos os Professores que me acompanham na minha vida de estudante e que todos os dias me permitiram aprender algo novo.
- Por ultimo aos meus pais, Rosa Maria e João Martins que sem os sacrifícios deles nada disto seria possível, para eles o meu eterno obrigado.

Resumo

Um Explorador Visual para Sistemas de Bases de Dados Relacionais

Atualmente, vivemos um período no qual a vulgarização da tecnologia é uma realidade. A importância das tecnologias da informação e da comunicação no nosso quotidiano não pode ser mais escamoteada. É uma das nossas componentes de vida. As tecnologias da informação e da comunicação encontram-se tão presentes na vida das pessoas, que encontramos já projetos para a sua introdução nos primeiros ciclos de aprendizagem, que surgem a um ritmo bastante rápido, com aplicações próprias e com uma forte carga pedagógica associada. Apesar de tal circunstância, neste domínio a área dos sistemas de bases de dados tem sido relegada para segundo lugar, sendo vista como uma área de trabalho bastante complexa e direcionada para utilizadores muito específicos. Apesar disso, ao longo dos últimos anos, várias iniciativas neste domínio têm vindo a realizar esforços significativos na aproximação ao utilizador comum, refletindo-se no desenvolvimento de linguagens textuais de alto nível, que usam termos correntes, do dia-a-dia, até à disponibilização de linguagens visuais, bastante amigáveis e proporcionando interfaces de exploração muito avançados. Apesar de todas estas iniciativas serem direcionadas para utilizadores considerados inexperientes e com poucos conhecimentos no domínio dos sistemas de bases de dados, nunca foi possível, pelo menos até ao momento e de acordo com o nosso conhecimento, disponibilizar um ambiente de exploração simples e intuitivo que facilitasse qualquer processo de exploração de dados. Nesta dissertação abordamos essa problemática com o intuito de discutir as características mais elementares que um qualquer explorador de dados deve possuir, de forma a simplificar a sua utilização ao utilizadores mais comuns. Na realidade, pretende-se disponibilizar um ambiente de exploração para todos os utilizadores, dos "7 aos 77", que não possuam conhecimentos avançados de bases de dados, mas que pretendam contudo desenvolver atividades de exploração de dados com vista a aplicação dos seus resultados numa qualquer aplicação do mundo real. O ambiente projetado, e que foi desenvolvido, será discutido de forma a possibilitar uma ideia concreta dos seus fundamentos, arquitetura e serviços de exploração que disponibiliza na versão atual.

Palavras-chave: Base de Dados Relacionais, Interfaces para Exploração de Bases de Dados, Ambientes Visuais para Exploração de Dados, Ferramentas de Ensino e de Aprendizagem.

Abstract

A Relacional Database System Visual Explorer

Today, we live in a time where tecnology has become commonplace. The importance of the information and communication technology cannot be overlooked anymore. It has become an essential component to our lives. The information technologies and communications are so widespread and ingrained into people's lives, that plans are being made to introduce them in the earliest stages of the educational system, sprouting at a pretty fast with their own applications and a heavy pedagogical focus. In spite of all these tendencies, the database domain has been continuously dismissed for being associated with a higher complexity and a very specific set of users. Even so, there have been through the years a few initiatives on this field, who have tried to narrow the gap with the common user, either through the development of high-level text languages which use familiar day-to-day keywords, or the creation of user friendly (but with advanced exploration interfaces) visual languages. But although these initiatives are focused towards the inexperienced and lacking knowledge on the Database management system domain user, it has not been possible, so far and to our knowledge, to create a simple and intuitive exploration environment that could facilitate the data exploration process. On this dissertation we shall address this issue, discussing the basic features a data explorer must have, so as to simplify its use by common users. In truth, we aim to create an exploration environment for all users, age "7 to 77", with no advanced knowledge on databases, but who seek to explore data and a apply their findings in whatever project they need. This environment, designed and developed here, shall be discussed in such a way, as to allow for a clear picture of its fundations, arquitecture and exploration features it includes in its current version.

Keywords: Relation Database, Database Exploration Interface, Data Exploration Visual Environment, Learing and Teching Tools

Índice

Índice de Figuras

Índice de Tabelas

Lista de Siglas e Acrónimos

TI	: Tecnologias da Informação
IBM	: International Business Machines
SEQUEL	: Structured English Query Language ou Linguagem de Consulta Estruturada em Inglês
SQL	: Structrured Query Language ou Linguagem de Consulta Estruturada
DDL	: Data Definition Language ou Linguagem de Definição de Dados
DML	: Data Manipulation Language ou Linguagem de Manipulação de Dados
QBE	: Query By Example
CUPID	: Casual User Pictorial Interface Design
PBE	: Patern By Example
SGBD	: Sistema Gestão de Base de Dados
QL	: Query Language
PICASSO	: PICture Aided Sophisticated Sketch Of database queries
SQUARE	: Specifying Queries As Relational Expressions
QUEL	: Query Language
SQLOO	: Structrured Query Language Object Oriented
XML	: Extensible Markup Language
VQS	: Visual Query Systems
DOODLE	: Declarative Language for Object-Oriented Databases
QBI	: Query By Icons
CSS	: Cascading Style Sheets
PHP	: Hypertext Preprocessor
RAM	: Random Access Memory
HTML	: Hyper Text Markup Language ou linguagem de Marcação de Hipertexto

Capítulo 1

Introdução

1.1 Contextualização

Nas últimas décadas, a humanidade tem vivido numa era dita de "globalização", em que as pessoas passam o dia "agarradas" a aparelhos electrónicos que lhes permitem estarem ligadas com o mundo, 24 sobre 24 horas. Em grande parte, este novo modo de vida assenta na Internet ou em aplicações tais que têm feito com que as pessoas mudem a imagem que previamente tinham concebido sobre a Informática. Esta nova imagem tem gerado um grande movimento em volta das *Tecnologias da Informação e da Comunicação* (TIC), algo que tem sido descrito por alguns como revolucionário. Este movimento tem tentado introduzir, o mais cedo possível, a área das TIC na vida das crianças a partir de jogos como o Play-i (Play-i 2013), o Primo (Labs 2013) ou o Robot Turtles (Shapiro 2013). Além disso, tal "onda" tem tentado também entrar nos primeiros ciclos nas redes de ensino de todo o mundo, justificando-se tal atitude através da defesa de que tal estimula o desenvolvimento das crianças, bem como a sua capacidade de raciocínio e pensamento lógico.

Um dos principais braços deste movimento (senão o principal) denomina-se por CODE.ORG (Code.org 2012), que tenta ensinar os alunos a programar através

da utilização de ferramentas próprias. Todas as ferramentas usadas são simples e pedagógicas. Porém, apesar da sua importância no mundo real, nenhuma destas ferramentas trabalha com problemas ou analisa situações na área dos sistemas de bases de dados. Mesmo sendo relegada para segundo lugar por projetos como este, ao longo dos últimos tempos ela tem vindo a ser trabalhada por alguns investigadores e técnicos no sentido de se aproximar às necessidades das pessoas ditas comuns.

Os primeiros passos nesta área foram dados nos anos 60 do século passado. Porém, só nos anos 70 é que esta área começou a adquirir um "lugar ao sol". Nesta altura, foram dados na IBM passos decisivos para a evolução dos sistemas de dados. Vários funcionários e equipas de desenvolvimento da empresa desenvolveram e publicaram vários artigos sobre a criação e a evolução das tecnologias para a manipulação de dados e, em especial, sobre o modelo relacional. Assim, em 1970, Edgar Codd apresentou o artigo "*Relational Model of Data for Large Shared Data Banks*" (CODD 1970), no qual apresentou pela primeira vez o modelo relacional. Como sabemos, este modelo acolhe uma organização na qual os dados estão armazenados em entidades e em relacionamentos entre entidades, na forma de registos (tuplos) que se podem relacionar entre si. Nesse artigo discutiu também as características e objetivos do modelo relacional. De referir:

- A existência de um alto grau de independência dos dados, o que significa que qualquer alteração que seja realizada ao nível da organização dos dados, não se deve sentir ao nível das aplicações.
- A capacidade do modelo em lidar com a própria semântica dos dados, a consistência e os problemas de redundância.
- A disponibilização de uma base para a criação de novas linguagens de manipulação de dados, mais fáceis, eficientes e flexíveis.

Ainda na mesma empresa surgiram também mais dois projetos, que usualmente são descritos como linguagens para bases de dados. Como tal, estes permitem através de mecanismos específicos executar tarefas de gestão e de manutenção de uma base de dados. Apesar de surgirem na mesma

empresa, os projetos porém resultaram em paradigmas completamente diferentes. Um deles deu origem a uma linguagem para bases de dados textuais, enquanto que o outro resultou numa linguagem de exploração de dados assente num sistema com uma interface gráfica "amigável".

O primeiro projeto, denominado SEQUEL, surgiu no seio de uma equipa de desenvolvimento orientada por Donald Chamberllin (Chamberlin & Boyce 1974), que viria a desenvolver o seu projeto baseado nos trabalhos de Edgar Codd (CODD 1970), originando uma linguagem para as vertentes DDL, DML e DCL (Connolly e Begg 2004), ou seja, uma linguagem que suporta a manipulação de dados, a definição da estrutura da base de dados e a definição das regras para controlo de acessos. Com isto tudo a linguagem SQL obteve bastante sucesso, sendo hoje a linguagem padrão para sistemas de dados relacionais. Esta tem vindo a evoluir até aos dias de hoje. Cada fornecedor de sistemas de gestão de bases de dados (SGBD) implementado a sua própria distribuição, consequentemente surgiu um leque diversificado de produtos na área.

O segundo projeto, originalmente criado para facilitar a recuperação de dados, surgiu pelas mãos de Moshé Zloof (Zloof 1981). Denominado por QBE (*Query By Example*), este projeto pretendia, a partir de uma interface intuitiva, para a altura, converter todas as ações do utilizador em expressões específicas de uma linguagem de manipulação. Ao contrário da SQL, na QBE o utilizador não descreve qualquer tipo de procedimento, apenas descreve o resultado que pretende obter. O sucesso deste projeto passou pela sua inclusão em muitas aplicações de bases de dados, nas quais frequentemente acontecia que o utilizador que estava a utilizar o sistema não se apercebia que estava a utilizar a QBE.

Posteriormente, a QBE veio influenciar e inspirar também muito outros projetos para além da esfera universitária. Entre eles podemos encontrar o CUPID (Nancy H. McDonald 1975), o PICASSO (Hyoung-Joo Kim 1988) e o PBE (Benzaken et al. 2008). Porém, apesar da sua grande diversidade, todos estes projetos de uma forma geral apresentavam o mesmo conjunto de falhas:

- funcionavam apenas em processos que envolviam consultas simples;

- funcionavam apenas sobre esquemas de base de dados relativamente simples;
- não apresentavam mecanismos eficientes para suporte à realização de operações de junção.

Com o passar do tempo, todas as ferramentas evoluíram em diversos sentidos. Mas, o objetivo de atenuar a complexidade dos processos de exploração e aumentar a facilidade de exploração não foi ainda alcançado, tendo a sua prossecução ocorrido até ao momento de forma muito lenta.

Face ao exposto, esta dissertação tem como objetivo principal a criação de um sistema de interrogação para base de dados relacionais capaz de ser utilizado pelos mais variadíssimos utilizadores ("dos sete ao setenta e sete") sem exigir grandes conhecimento sobre sistemas de bases de dados, a não ser saber como manipular objetos gráficos que, na prática, representam objetos de dados de uma bases de dados. Porém, mesmo de uma forma simples, o sistema deverá ter a capacidade de poder acolher todos os comandos presentes em qualquer linguagem de interrogação de dados atual.

1.2 Objetivos do Projeto

Quando falamos de um *sistema de gestão de bases de dados* (SGBD), falamos de uma peça de *software* bastante sofisticada, que tem como objetivo a gestão e manutenção de bases de dados. Para além disso, um SGBD acolhe a definição de vários perfis de utilização, bem como dos próprios utilizadores, regulando as suas ações de acordo com as suas características e privilégios de exploração do sistema. Segundo Connolly e Begg (Connolly & Begg 2004), podemos organizar os utilizadores de um SGBD em várias categorias de utilizadores, a saber:

- Administrador de dados e administrador de base de dados, na grande maioria dos projetos estes dois intervenientes são a mesma pessoa. O administrador é o responsável pela gestão do SGBD, o que inclui a gestão e a realização de atividades como o planeamento do próprio sistema de base de dados, a definição de políticas e procedimentos de

acesso, o controlo da integridade, a garantia do desempenho satisfatório, entre outros.

- Arquiteto ("Designer") de base de dados, que é o responsável pela implementação do projeto do sistema de bases de dados e da sua configuração para o SGBD em questão; esta última atividade é um pouco delicada já que cada um SGBD tem as suas próprias características. Este tipo de utilizador também é o responsável por fazer a identificação dos dados (entidades e atributos), relações e restrições, que são frequentemente denominadas por regras do negócio.
- Programador, que é o responsável pelo desenvolvimento das aplicações cliente que interagem diretamente como SGBD e como sistema de dados implementado.
- Utilizador final, que é, de facto, o cliente final do sistema de bases de dados. Este tipo de utilizador constitui o grupo com o maior número de pessoas, podendo ser organizado em dois tipos de utilizador: inexperiente e sofisticado. O primeiro deste tipo de utilizador tem normalmente credenciais para aceder ao sistema de bases de dados através de programas específicos, que foram desenvolvidos de acordo com as suas competências e necessidades de utilização. O segundo tipo por norma tem um nível de conhecimento que lhe permite estar à vontade com a estrutura da própria base de dados e com as facilidades oferecidas pelo SGBD para a sua exploração.

Quando partimos para o desenvolvimento de uma aplicação, antes de mais, precisamos de fazer um bom planeamento do desenrolar das várias atividades necessárias para a sua implementação e conhecer muito bem o seu público-alvo, os seus futuros utilizadores. No nosso caso, quando pensámos no desenvolvimento de um sistema de exploração para "toda a gente" focámo-nos principalmente no primeiro destes grupos de utilizadores, acrescentando-lhe contudo um pouco mais de ambição ao desenharmos um sistema que permitisse a exploração de um sistema de bases de dados também por crianças – tal como já referido, pretendemos disponibilizar uma experiência de exploração de dados dos "7 aos 77" anos.

De forma complementar, pretendemos também incluir o sistema idealizado e implementado nesta dissertação no primeiro e segundo ciclo de estudos, a partir de projetos como o CODE.ORG (Code.org 2012), para que a curto prazo pudéssemos ter uma população de utilizadores mais numerosa e com novas perspetivas em termos de exploração de sistemas de bases de dados e sua aplicação prática a problemas reais. Nesse sentido, definimos os seguintes objetivos para esta dissertação:

- construir um sistema de interrogação para bases de dados relacionais, que seja intuitivo, de fácil utilização e que não exija qualquer tipo de conhecimento da linguagem SQL;
- disponibilizar um modelo e respetivos mecanismos de interrogação para sistemas de bases de dados relacionais que fosse capaz de cativar o utilizador a experimentar o próprio sistema, garantindo o acesso a uma base de dados e à sua consequente exploração, conhecendo-se apenas o assunto e o conteúdo da base de dados envolvida;
- providenciar um conjunto de manipuladores gráficos capazes de representar os diferentes objetos de uma base de dados, cuja conjugação permitirá fazer a combinação dos dados relativos a diferentes tipos de objetos, com a capacidade de inferir qual a operação pretendida pelo utilizador;
- construir um ambiente de exploração de dados simples e intuitivo, no qual os utilizadores fossem capazes de realizar as suas operações de exploração de dados sobre uma base de dados relacional, sem que se apercebam do modelo de dados em questão ou da forma como a base de dados está organizada;
- conceber um conjunto de objetos gráficos representativo dos vários tipos de objetos de dados de uma base de dados relacional, cujas propriedades – e.g. tamanho e cor - revelarão as características do tipo de objeto em exploração, como seja o número de registos ou o número de atributos;

- providenciar uma nova experiência de exploração de dados, agradável, simples e intuitiva, livre de instalações e disponível para qualquer tipo de plataforma.

1.3 Estrutura do Documento

Para além do presente capítulo, esta dissertação integra um conjunto de mais quatro outros capítulos, que estão organizados da seguinte forma:

- **Capítulo 2** – *Linguagens para Sistemas de Base de Dados*. Neste capítulo faz-se uma breve introdução às linguagens para sistemas de bases de dados, apresentando-se alguns dos paradigmas textuais e visuais que vários projetos nos foram propiciando ao longo dos últimos tempos.
- **Capítulo 3** – *A Especificação do Sistema*. Ao longo deste capítulo apresenta-se a especificação do sistema desenvolvido com recurso à Álgebra Relacional e à linguagem SQL. Na especificação apresentada estão incorporados todos os comandos do sistema bem como a representação de todos os objetos gráficos utilizados na representação dos objetos de uma base de dados relacional.
- **Capítulo 4** – *O Sistema de Interrogação*. Neste capítulo expomos o processo de desenvolvimento do sistema de interrogação idealizado, apresentando também o caso de estudo que selecionámos para a demonstração de todos os comandos que foram desenvolvidos. Para além disso, discutimos a estrutura física do sistema bem como as ferramentas utilizadas no seu desenvolvimento.
- **Capítulo 5** – *Conclusões e Trabalho Futuro*. Neste capítulo apresenta-se uma breve análise crítica do trabalho realizado. Além disso, são revistos também alguns aspetos relevantes sobre a utilização de linguagens visuais na exploração de sistemas de bases de dados e a forma como o sistema desenvolvido os acolheu. Terminamos este capítulo enunciando alguns dos aspetos que o desenvolvimento de uma nova versão do sistema poderá melhorar ou incorporar através de novos serviços.

Capitulo 2

Linguagens para Sistemas de Bases de Dados

2.1 Um Modelo de Dados

Em termos gerais, podemos dizer que as bases de dados surgiram na década de 60 do século passado nos Estados Unidos. Se até essa altura os dados eram armazenados em dispositivos designados por arquivos de texto, a partir de 1963, pelas mãos do Departamento de Defesa dos Estados Unidos, emergiu a primeira definição de base de dados: "Uma base de dados é uma coleção de arquivos inter-relacionados " (Olle 2006).

Nessa mesma década haveria de surgir também o primeiro programa que iria permitir trabalhar diretamente com uma base de dados. Na altura trabalhava-se com módulos de dados organizados hierarquicamente numa rede, o que implicava a manutenção de uma estrutura de dados em árvore. Tal operação era, pois, de difícil realização. Tal acentuava-se quando tínhamos que lidar com grandes quantidades de dados, o que, em muitos casos, impossibilitava o uso destes modelos. Assim, dada essa limitação, não se podia afirmar então que a aplicação de tais bases de dados a problemas reais era uma atividade considerada de sucesso. Além da dificuldade de acolhimento de grandes volumes de dados disso, as bases de dados hierárquicas apresentavam uma grande complexidade em termos de exploração de dados, mesmo em operações de manipulação de dados consideradas simples.

Na década de 70, um funcionário da IBM, Edgar Codd, criou o modelo relacional, que descreveu nesse mesmo ano num artigo intitulado "A Relacional Model of Data for Large Shares Data Banks" (CODD 1970). Codd sugeriu o modelo relacional motivado pela observação de como as bases de dados armazenavam os dados até à data (Chamberlin et al. 1981), em particular tendo em atenção aspetos como os conteúdos dos registos e a forma como estes se ligavam entre si. Porém, o modelo existente apresentava uma característica bastante pertinente: os dados não eram independentes. Ao deparar com esta situação, Codd desenvolveu e apresentou o modelo relacional (CODD 1970), no qual definiu que (Chamberlin et al. 1981):

- a informação devia ser representada pelos valores dos dados e não por "conexões" visíveis ao utilizador;
- o sistema deveria suportar linguagens de alto nível que possibilitassem aos utilizadores fazer consultas aos dados existentes sem terem a necessidade de especificar o algoritmo para fazer o seu processamento.

Codd defendia no seu artigo (CODD 1970), o que foi posto em prática, um modelo no qual os dados se encontravam armazenados em relações (ou tabelas), que eram constituídos por linhas (registos) e colunas (atributos) e que se podiam relacionar entre si. Quando este modelo foi apresentado, apresentava logo um conjunto de vantagens que fazia com que ele se distinguisse de qualquer outro:

- ao contrário dos modelos anteriores o modelo relacional tinha um suporte matemático, o que permitia, obviamente, uma expansão e organização dos problemas bastante superior;
- o utilizador ou o programador de aplicações não tinha que conhecer a implementação física da base de dados.

O trabalho de Edgar Codd fez despoletar dois projetos na área das bases de dados. Estes projetos também viriam a surgir na IBM. O primeiro seria denominado por SEQUEL e o segundo por QBE.

A partir dos anos 80 o *hardware* e o *software* começaram a evoluir de forma muito significativa. No caso do *hardware* verificou-se um enorme aumento da sua capacidade bem como uma redução do seu preço, enquanto que no *software* houve uma melhoria substancial em termos de acessibilidade. Todos estes fatores fizerem com que houvesse um aumento claro da presença de computadores nos mais diversos postos de trabalho. Tal aumento provocou também uma evolução sem precedentes ao nível dos SGBD. Estes sistemas apresentavam-se com um leque de características muito interessantes e atrativas, das quais se ressaltam as seguintes (Connolly & Begg 2004):

- o fornecimento de um interface simples aos seus utilizadores, de uma linguagem de alto nível para descrição, manipulação e controlo de dados e de garantir a independência dos dados;
- o suporte a diferentes tipos de exploração de base de dados, como sejam as operações programadas, as consultas *ad hoc*, ou a geração de relatórios;
- o suporte para a realização de mudanças rápidas sobre a base de dados, bem como a alteração de índices, de tabelas, ou de *views*;
- o suportar acessos simultâneos de vários utilizadores, sempre de forma a garantir a manutenção da integridade da base de dados;
- a capacidade de recuperar uma base de dados de um estado inconsistente para um estado consistente anterior à ocorrência que provocou esse estado irregular da base de dados;
- a possibilidade de construir de vistas sobre os dados armazenados;
- o suporte para a utilização de funções de linguagens de alto nível com um desempenho comparável ao das linguagens de baixo nível.

A evolução destes sistemas continuou a um ritmo bastante constante, verificando-se ainda hoje um investimento enorme de recursos financeiros e técnicos por várias empresas do domínio das TIC, procurando apresentar um produto que se evidencie num sector que absorve uma fatia muito significativa do mercado global das TIC .

2.2 Uma Linguagem para Consulta de Dados

Na década de 80 com o aparecimento da Internet e com o aumento do poder de compra da população mundial, as empresas começaram a ter algumas dificuldades no acolhimento e tratamento dos dados envolvidos nas suas atividades. A manutenção dos seus atuais sistemas de armazenamento e manipulação de dados começava a ser colocada em causa. Requeria-se, sistemas com mais capacidades, mais flexíveis e de mais fácil utilização. Os SGBD apresentaram-se na altura como uma alternativa muito viável. As suas características funcionais e orgânicas fizeram com rapidamente assumissem um papel fundamental, disponibilizando meios para uma gestão eficiente e acesso simultâneo a vários utilizadores aos seus repositórios de dados. Apesar disso, os SGBD só conseguiram cimentar tal posição, tornando-se fundamentais nas mais variadas organizações, por disponibilizarem meios concretos e de fácil utilização para descrever, manipular e controlar o acesso aos seus dados – as linguagens de bases de dados. Tal, foi a chave do seu sucesso.

Uma linguagem de consulta para base de dados pode ser definida como uma linguagem de computador de alto nível para ser utilizada em processos de manipulação de dados que estejam contidos numa base de dados ou noutro tipo qualquer arquivo. Geralmente é uma linguagem interativa, capaz de oferecer suporte a consultas que frequentemente podem ser classificadas como *ad-hoc* (Matthias Jarke 1985). É nesta vertente que surgem as linguagens de consulta para sistemas de dados relacionais. Porém, apesar de existirem muitas linguagens deste tipo, todas elas são diferentes na forma como atuam. Contudo, todas elas também têm uma característica em comum: todas são tão poderosas quanto a Álgebra Relacional.

Quando trabalhamos com linguagens de consulta, podemos dividi-las em dois grupos distintos de linguagens, nomeadamente:

1) Consulta textual.
2) Consulta visual.

Quando falamos de uma linguagem textual, estamos a referirmo-nos em particular a uma linguagem que requer ao utilizador alguma experiência no seu domínio de aplicação, bem como o conhecimento da estrutura da própria base de dados e os fundamentos da Álgebra Relacional. Tudo isso é necessário para que a curva de aprendizagem da linguagem não seja muito longa.

Quanto às linguagens de consulta visual, estas referem-se a ferramentas que recorrem a estímulos visuais para expressar pedidos de informação a um dado repositório de dados, recebendo em troca, se possível, algum *feedback* em termos de dados. Por norma, este tipo de linguagem é usada por utilizadores inexperientes. Basicamente são utilizadores que só conseguem utilizar a linguagem a partir de um interface simples que lhes permite de forma fácil realizar as consultas que pretendem sem que para isso precisem de conhecer os detalhes do esquema da base de dados ou a sintaxe da própria linguagem (Hyoung-Joo Kim 1988).

2.2.1 Linguagens de Consulta Textual

Quando surgiu o modelo relacional, a IBM decidiu promover a utilização de linguagens de alto nível. Nesse sentido nos seus laboratórios foi desenvolvido o System R (Chamberlin et al. 1981), o primeiro sistema para gestão de base de dados relacionais. Com base neste sistema, a partir 1973, surgiu a primeira linguagem de interrogação de bases de dados de alto nível, o SQUARE (Reisner et al. 1975), suportada obviamente pelos fundamentos e conceitos subjacentes à Álgebra Relacional. Mas, desde o seu início, o facto de ser uma linguagem baseada em Álgebra Relacional, fez com que o SQUARE fosse criticado, uma vez que ao adotar uma sintaxe pouco compreensível para o utilizador comum, a sua escrita estava impregnada de muitos símbolos matemáticos. Assim, para resolver o problema da falta de adaptação por parte dos utilizadores ao SQUARE, surgiu a SEQUEL (Reisner et al. 1975).

Quando a SEQUEL surgiu, esta vinha com o System R. Nessa altura o SEQUEL apareceu também com o QUEL, um projeto da Universidade da Califórnia, que estava inserido no INGRES. Com a SEQUEL foi dado um grande passo na eliminação da complexidade dos processos de acesso aos dados e sua

consequente exploração. A SEQUEL apresentava uma sintaxe bastante mais *user-friendly* e compreensível, uma vez que para trabalhar com os dados disponibilizava um conjunto de palavras de língua inglesa como comandos base de descrição, manipulação e controlo. O facto de ser constituída por palavras usadas no dia-a-dia, não necessitando de uma aprendizagem tão exigente como aquela que o SQUARE exigia, permitiu ao SEQUEL um grande sucesso entre os utilizadores que não tinham experiência na área das bases de dados, tais como os contabilistas, os engenheiros ou os arquitetos. Além da sua sintaxe, o SEQUEL também deve o seu grande sucesso ao facto de ser uma linguagem de consulta não procedimental. Na SEQUEL, um utilizador para executar um processo qualquer de consulta de dados apenas tem que usar um ”bloco” de instruções, que, na nossa opinião, é bastante claro: o bloco SELECT-FROM-WHERE. Este bloco permite ao SEQUEL funcionar de uma forma muito simples, requerendo ao utilizador que indique que colunas deseja obter (SELECT), quais as tabelas em que pretende as referidas colunas (FROM) e quais as condições de filtragem (se alguma) que devem ser aplicadas (WHERE) (Chamberlin & Boyce 1974).

Ainda nos anos 70, a IBM viu-se forçada a alterar o nome da SEQUEL para SQL, uma vez que esse nome já se encontrava registado pela empresa Hawker Siddeley. Devido ao facto da linguagem SQL ser um projeto aberto, a sua comunidade de utilizadores foi gradualmente aumentando. Além de ser utilizada por utilizadores ditos independentes, a linguagem foi ganhando também bastante espaço no mercado dos sistemas de bases de dados ao ponto de se tornar um padrão ao nível dos sistemas relacionais. Porém, só no final da década de 80 é que a SQL foi considerada como padrão no domínio da descrição, manipulação e controlo de bases de dados relacionais. Tal aconteceu em 1986 pela ANSI e em 1987 pela ISO. Na altura a versão do SQL denominava-se por SQL-89.

A seguir a esta primeira versão da SQL seguiram-se mais três versões, todas elas classificadas também como padrão. Essas três versões foram designadas, respetivamente, por SQL-92, SQL3 (ou SQL-99) e, por último, SQL-2003. Estas

três versões contribuíram para uma forte evolução da linguagem SQL, inovando e fortalecendo muitas das suas capacidades em particular na vertente de descrição de dados (DDL) e na vertente de manipulação de dados (DML). A versão SQL3 surgiu no mercado com bastantes alterações, relativamente à versão anterior da linguagem. Além de permitir a definição de *queries* recursivas e de gatilhos (*triggers*), esta versão integrava já inúmeros aspetos das linguagens orientadas ao objeto, o que veio permitir a definição de tipos de dados muito mais complexos. Esta versão provocou também algumas alterações nos padrões até então definidos, fazendo com que a SQL passasse a ser incluída em SGBD, já com suporte para a definição de *queries* orientadas ao objeto, como é o caso dos sistemas disponibilizados pela Oracle (Oracle 8) e pela IBM (DB2). Devido a estas características, esta versão ficou conhecida como SQLOO. Foi aprovada pela ANSI em 2001. A última versão da SQL, o SQL-2003, trouxe-nos como novidade a introdução de características da linguagem XML – e.g. sequências padronizadas ou as colunas de valor auto-incrementável.

Desde o seu aparecimento que a SQL tem trilhado novos caminhos, muitos deles à conta de trabalhos de investigação e de desenvolvimento realizados em universidades. Sempre que uma dada versão padrão da SQL não conseguia responder às necessidades dos utilizadores, iam surgindo desses processos módulos novos que foram dando à SQL a capacidade, por exemplo, de tratar dados multimédia, dados temporais, dados georreferenciados, operações de mineração de dados, ou adquirir algumas representações visuais. A juntar ao trabalho realizado pelas universidades, temos também o trabalho realizado nos diversos fabricantes de *software*. Estes, por sua vez, na tentativa de evoluírem os seus próprios produtos, têm vindo a desenvolver extensões específicas para satisfazer especificidades muito concretas que as aplicações dos seus produtos iam requerendo, sem que a SQL perdesse o seu estatuto de linguagem padrão para sistemas de bases de dados relacionais.

2.2.2 Linguagens de Consulta Visual

Nos últimos anos as linguagens de consulta visual têm tomado um grande espaço em ações de investigação realizadas no domínio das bases de dados. Como qualquer outra ferramenta nesta área, o objetivo deste tipo de linguagens é o de permitir o acesso e a exploração dos dados de uma dada base de dados através de meios e elementos gráficos. Se nos sistemas convencionais de acesso a dados a sua construção passa, primeiro, pela construção dos instrumentos necessários para "trabalhar" os dados, nos Visual Query Systems (VQS) o ponto de partida é o interface – uma espécie de mudança de paradigma. O interface deixa, pois, de ser algo puramente estético para passar a ser o ponto central deste tipo de linguagens.

A principal função de um VQS é traduzir os "gráficos" editados pelo utilizador em instruções SQL e fornecer um interface que possibilite ver os resultados de uma consulta depois de se executarem as instruções SQL (Cumming 2000). Um VQS pode ser visto como uma evolução das linguagens de consulta adotadas nos SGBD, uma vez que elas são projetadas para melhorar a eficácia da comunicação homem-máquina (CATARCI et al. 1997). Este tipo de interface vai representar um modelo externo, que permite ao utilizador a visualização e a manipulação de dados, e que, juntamente com o modelo interno constituem um VQS (C. Batini 1995). O modelo interno, encontra-se relacionado com o trabalho realizado em *background*, ou seja, tudo aquilo que se realiza acerca da forma como os dados são manipulados e armazenados. Isto faz com que o modelo interno seja dependente do SGBD escolhido e da sua linguagem de consulta. Este tipo de organização permite a um VQS possibilitar um desenvolvimento bastante independente, permitindo que um mesmo modelo externo possa ser usado com vários modelos internos.

Mas nem tudo são pontos positivos no VQS. O facto de não haver um padrão definido para os seus ícones, nem para a sua sintaxe, cria algumas situações um pouco ambíguas na sua interpretação. Depois algo como o VQS encontra-se sempre limitada pela tecnologia do momento do seu desenvolvimento, por exemplo pelo tamanho dos monitores. Apesar das suas limitações, os sistemas

deste tipo têm absorvido algum do trabalho de investigação desenvolvida na área das bases de dados, o que levou ao aparecimento de vários tipos de VQS. Assim, foi necessário fazer a sua classificação, de acordo com os elementos que constituem usualmente um VQS. Tendo isto em conta, podemos classificá-los como (CATARCI et al. 1997):

- **Linguagens baseadas em formulários**. Estas linguagens surgiram como resultado de uma primeira tentativa de exploração do espaço bidimensional, o que conduziu ao abandono do espaço mono-dimensional no qual as linguagens textuais estavam confinadas até então. Estas linguagens baseiam-se numa coleção de objetos, com a mesma estrutura, orientados para facilitar o trabalho aos utilizadores não especializados, aproveitando a sua capacidade de organização dos dados em tabelas. Outra das características destas linguagens são as suas representações estruturadas de uma abstração de formulários convencionais em papel. Estes formulários podem ser vistos como retângulos, cujos elementos podem resultar da combinação de células individuais ou de um grupo de células – uma célula é o elemento mais pequeno de um formulário. Uma célula permite três níveis de informação, de referir: a célula; subconjuntos de células; e a totalidade do conjunto das células. A primeira linguagem a surgir deste tipo foi um dos primeiros projetos sobre o modelo relacional proposto por Codd: o QBE. Este projeto, já referido anteriormente, disponibiliza uma interface gráfica assente num sistema tabular. Apesar de esta linguagem não ser muito do conhecimento do público em geral, ao longo dos tempos aos poucos tem vindo a ser reinventado. Esta evolução tem-lhe permitido ser inserido em vários outros projetos, por exemplo, na disponibilização de aplicações para a gestão de mensagens eletrónicas ou de dados georreferenciados.
- **Linguagens baseadas em diagramas**. As linguagens baseadas em diagramas surgiram com o intuito de representar graficamente os dados contidos em formulários. Apesar de acolherem a denominação "baseadas em diagrama", estas linguagens não estão restritas à utilização de tal representação gráfica. Por norma são usadas várias formas geométricas

para representar os diferentes elementos de uma base de dados, sendo usadas linhas, por exemplo, para representar os relacionamentos entre vários elementos de dados. Ao longo dos anos têm surgido vários projetos, com características muito variadas, relacionados com este tipo de linguagem. Refira-se a título de exemplo o SUPER (Dennebouy et al. 1995) e o PICASSO (Hyoung-Joo Kim 1988), entre outros. Mais recentemente, o desenvolvimento de ferramentas nesta área tem-se virado essencialmente para sistemas 3D, o que permite aumentar, obviamente, o poder de representação das linguagens. Projetos como o AMAZE (Boyle et al. 1996) ou o DOODLE (Cruz 1992) são já referências importantes nesta nova geração de linguagens 3D.

- **Linguagens baseadas em ícones**. Este tipo de linguagem assenta em processos computacionais baseados em imagens, que são objetos reais com a capacidade de representar ações processos ou mesmo conceitos. Segundo Fujii e Korfhage (1991) "Um ícone é um objeto visualmente segmentado que informa o utilizador sobre uma mensagem ou informação interior (conceito, função, estado, modo, etc.) atribuído pelo projetista". Apesar do conceito de ícone e de diagrama se poderem confundir, o primeiro tem como objetivo a representação de um conceito enquanto o segundo favorece a visualização da relação entre conceitos. Este tipo de sistemas são dirigidos a utilizadores que não estejam familiarizados com o conceito de modelos de dados. Apesar desta característica, e do ícone conter um significado metafórico, estes sistemas podem ser, por vezes, ambíguos. Esta ambiguidade provém da falta de um padrão nos ícones utilizados nas diferentes aplicações. Além disso, tal falta, bem como o aumento do número de ícones, pode provocar uma diminuição da capacidade de descrição, o que pode conduzir os utilizadores mais inexperientes à sua não interpretação. Apesar desta situação estes sistemas apresentam algumas vantagens em termos dos processos de manipulação de dados. Prova disso, é o facto de que as ações nestes sistemas resultam de simples combinação de ícones. Um projeto nesta área é o QBI (Massari et al. 1994), que a partir da

manipulação de ícones permite consultar os dados contidos numa dada base de dados. Este sistema como qualquer linguagem da base de dados do tipo visual é direcionada, obviamente, para utilizadores considerados inexperientes.

- **Linguagens híbridas**. A hibridização de linguagens é um processo que tem a vantagem de combinar qualquer uma das subcategorias dos VQS. Assim, uma linguagem classificada como híbrida apresenta uma combinação arbitrária de dois ou mais formalismos de outro tipo de linguagem. Esta característica permite a um utilizador dispor de várias alternativas para a representação de uma base de dados e para as ações de consulta realizadas por um único formalismo ou resultante da combinação de vários formalismos.

2.3 Nota Final

Apesar de toda a evolução que as linguagens como a SQL têm vindo a sofrer, um "fosso" significativo continua a existir entre as funcionalidades dos sistemas e os sistemas de interface das linguagens em termos de amigabilidade, separando-as do utilizador comum, dadas as grandes exigências que estas colocam na sua efetiva utilização. Portal fosso tem vindo a ser atenuado por outras linguagens, com características gráficas muito avançadas, que desde os anos 70 têm vindo a evoluir de forma muito significativa. De facto, tiveram o seu ponto de partida com a proposta da linguagem QBE. Desde os anos 70 que estas linguagens gráficas apresentam uma evolução surpreendente, estando presente na introdução de dimensões, o uso de cores, de diagramas ou de ícones. Apesar de todas estas propostas se encontrarem limitadas pela tecnologia, as linguagens gráficas apresentam uma enorme margem de progressão, tornando a sua aprendizagem cada vez mais simples e interessante, bem como o acesso e a exploração dos dados. Esta evolução tem permitido atrair cada vez mais um maior número de utilizadores inexperientes para a sua utilização. Assim, a estratégia para o desenvolvimento do sistema de exploração que apresentamos e discutimos nesta dissertação, passou por juntar conceitos vindos das linguagens baseadas em diagramas, nas quais se foi

buscar inspiração às linguagens baseadas em ícones, tanto em termos de elementos gráficos como em termos de formas de atuação.

Capitulo 3

A Especificação do Sistema

3.1 Especificações Formais

Uma especificação formal de uma peça de *software* (ou de *hardware)* é uma descrição matemática que pode ser utilizada na sua implementação, permitindo descrever o que tal peça deve fazer e não (necessariamente) como o deve fazer (Jcarnelian 2005). No desenvolvimento de ferramentas visuais para a manipulação de dados, existe sempre um grande esforço por parte de quem as desenvolve, no sentido de que as ações e processos associados com uma peça de *software* possam ser mantidos da forma mais simples possível. Para que tal simplicidade seja refletida nas ações de tais ferramentas, usualmente adotam-se estratégias muito diversificadas de interação que podem envolver desde o simples uso de formulários, até à utilização de ícones ou de diagramas. Mas, seja qual for a estratégia escolhida pela equipa de desenvolvimento da ferramenta, o interesse pela simplificação dos processos de interação resulta frequentemente da preocupação em permitir uma melhor usabilidade e perceção por parte do utilizador dos objetos existentes numa qualquer base de dados.

Apesar de todas estas ferramentas gráficas seguirem estratégias de desenvolvimento diferentes, estas baseiam-se simplesmente na ideia de disponibilizar uma interface intuitiva em que todas as ações são convertidas em

comandos para base de dados usando linguagens textuais, para que o utilizador possa explorar os seus dados de uma forma muito simples.

No caso da nossa aplicação utilizámos como base a linguagem SQL que, como todos sabemos, é amplamente utilizada no domínio dos SGBD. Assim, ao longo da apresentação e demonstração do sistema de exploração de dados que desenvolvemos, teremos a SQL presente na especificação formal das ações e dos processos do sistema de interrogação que desenvolvemos.

No capítulo anterior foi possível perceber a forte ligação, uma dependência mesmo direta, que existe entre a SQL e a Álgebra Relacional. Uma especificação formal necessita sempre de uma prova não ambígua, de preferência uma prova matemática. Assim, a especificação realizada ao longo capítulo das várias operações de exploração de dados contará também com uma prova em Álgebra Relacional para além da já referida e óbvia instrução em SQL.

Em ciências da computação, a Álgebra relacional é uma derivação da lógica de primeira ordem e da álgebra de conjuntos. Para as bases de dados relacionais a importância da Álgebra Relacional deve-se ao facto de esta permitir a sua fundamentação teórica, em particular da linguagem de alto nível que opera sobre si. Antes de se realizar a especificação formal dos comandos utilizados, faremos a especificação dos vários elementos visuais criados e integrados no interface do nosso sistema de exploração de dados.

3.2 Especificação dos Elementos Visuais

A definição e a especificação dos vários elementos visuais, de fácil manipulação e com a capacidade de expressar as características mais elementares de um objeto de dados segundo o modelo relacional, não foram tarefas fáceis. Segundo Edgar Codd (CODD 1970), as características gerais de uma tabela são, em termos gerais, as seguintes:

- uma linha representa um registo de dados (tuplo) de uma tabela;
- a ordem pela qual os atributos e as linhas de uma tabela é irrelevante;

- todas as linhas (os registos) são distintas e o número de linhas de uma tabela define a sua cardinalidade;
- o significado de cada coluna é parcialmente transmitida através da marcação com o nome do domínio correspondente;
- o número de atributos de uma tabela define o seu grau.

A partir deste conjunto base avaliámos uma série de possibilidades. Porém, tendo em consideração a representação gráfica do objeto de dados, bem como a simplicidade da sua manipulação em plataformas computacionais (com alguma influência das plataformas do tipo *tablet*) chegámos a uma solução, deveras simples: a utilização de elementos gráficos circulares (círculos). Para acolher esses elementos gráficos, pensámos numa tela – o painel de exploração de dados –, que acolheria tais elementos, que reagiriam a eventos e que refletiriam as propriedades do objeto de dados (e.g. cardinalidade ou grau) através da sua dimensão ou cor, por exemplo. De facto, tudo bastante simples. Vejamos então, nas próximas secções, como tais elementos foram definidos e especificados.

3.2.1 Porquê o Círculo?

Um círculo é uma forma simples da geometria euclidiana (figura 1). Este resulta do conjunto de todos os pontos num dado plano que estão posicionados a uma determinada distância de um ponto designado por centro. A distância entre qualquer um dos pontos e o centro é designada por raio.

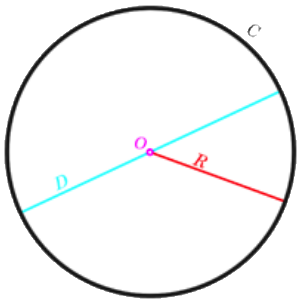


Figura 1 - Ilustração de um Circulo

Desde cedo que nos habituamos a associar a forma circular ao movimento, seja porque esta forma possa ser associada como a base de uma roda ou por esta

estar envolvida em muitas invenções relacionadas com engrenagens. O facto de relacionarmos o círculo com movimento foi uma das razões que nos levou a escolhe-lo como o elemento gráfico representativo de um objeto (tabela ou vista) de uma base de dados. Além da característica de transmissão de movimento, precisávamos também de uma figura geométrica passível de ser alterada, em particular em termos do seu tamanho e da sua cor, sem que para isso ela ficasse "desfigurada", perdesse as suas características base. Ao trabalharmos com o círculo evitámos algum trabalho na representação de um objeto de dados, nomeadamente o recalcular das suas dimensões para que a figura mantivesse sempre o seu aspeto visual. Com o círculo apenas o raio é afetado de forma a representar um maior ou menor número de atributos do objeto de dados em causa. O círculo tornou-se, assim, uma figura geométrica bastante "conveniente" para a representação dos objetos de dados e, mais tarde, na sua própria exploração.

3.2.2 Caracterização do Círculo como Elemento de Dados

As características dos elementos gráficos utilizados pelo sistema de exploração devem poder ser alteradas de forma a poderem refletir as propriedades dos objetos de dados que representam. Nesse sentido desenvolvemos os mecanismos necessários para que características como o tamanho, a cor ou o preenchimento da própria forma geométrica pudessem ser alteráveis. Estas mudanças nas características dos elementos gráficos tem como objetivo revelar de forma evidente – à primeira vista – algumas propriedades relevantes dos objetos de dados (tabelas ou vistas) em exploração, mesmo antes de conhecerem os dados se neles podem encontrar. A partir desta informação visual o utilizador poderá saber se o objeto de dados tem um grau alto ou baixo grau ou uma cardinalidade baixa ou alta, por exemplo. Como sabemos, o grau (ou aridade) de uma tabela é o número de atributos do esquema da relação (Elmasri e Navathe 2010), enquanto que a cardinalidade é o número de registos que uma tabela contém (Connolly e Begg 2004).

Assim, definimos que quanto mais escura fosse a cor do elemento visual, maior seria a cardinalidade do objeto de dados em representação. O cálculo da

intensidade da cor do elemento de dados é feito de uma forma muito simples, tendo sido definidos um conjunto de intervalos numéricos, cada um com uma dada cor associada. Dessa forma a intensidade da cor será tanto maior quanto maior for o número de registos de um dado intervalo. Quanto ao grau de uma tabela, definimos que ele influenciaria o raio do elemento visual, que como definimos, é um círculo. De forma semelhante à que realizámos para o cálculo da cor, o cálculo do raio é feito de forma semelhante. Aqui, também usámos intervalos de valores, mas agora representativos do número de atributos de um objeto de dados e, depois, a cada um destes intervalos associámos valores para a definição do raio do círculo em causa. Desta forma, um utilizador ao observar um elemento visual vai conhecer, de imediato, algumas das propriedades bases de um objeto de dados: quanto mais escura for a sua cor, mais registos tem, e quanto maior for o seu raio, maior será o seu número de atributos.

3.3 Especificação dos Comandos do Sistema

A quando do lançamento do modelo relacional, a Álgebra Relacional pouco interesse despertou fora do domínio daquilo que usualmente designamos por Matemática. Assim, Codd ao defender um modelo como o relacional, um modelo que poderia ser trabalhado através da utilização de linguagens de alto nível, e que tais linguagens seriam baseadas nessa álgebra, fez com que a Álgebra Relacional fosse vista com outros olhos. Porém, apesar disso, a primeira tentativa realizada através de uma linguagem de base de dados relacionais, o SQUARE (Reisner et al. 1975), foi amplamente criticada por usar na sua escrita bastantes conceitos matemáticos proveniente da Álgebra Relacional. Com isto, a IBM sentiu a necessidade de evoluir e criar a linguagem SEQUEL (Reisner et al. 1975), também ela fundamentada na Álgebra relacional. Porém, ao contrário da SQUARE, a SEQUEL foi muito bem recebida pelos profissionais da área. Este sucesso resultou do facto da linguagem SEQUEL "esconder" a sua origem matemática, recorrendo a termos da língua inglesa para realizar as operações mais usuais sobre uma base de dados.

Com o aparecimento das linguagens textuais emergiram vários projetos de interfaces e de linguagens visuais (VQS). Por norma, as VQS (CATARCI et al.

1997) recorrem a linguagens textuais de forma a suportar os processos de interação com os dados. Devido a esta característica, estas linguagens acabaram também por herdar as operações mais primitivas da Álgebra Relacional, nomeadamente: a seleção, a projeção, o produto cartesiano, a união e a diferença.

3.3.1 Especificação do Comando de Projeção

A operação de projeção é uma das operações fundamentais da Álgebra Relacional e, como tal, de extrema importância no âmbito do projeto de desenvolvimento do sistema de exploração de dados que idealizámos. Representada matematicamente pelo símbolo π (letra pi do alfabeto grego), a operação de projeção foi definida para fazer a seleção de colunas de uma tabela ou vista. Neste projeto, as operações de projeção permitirão fazer a caracterização inicial de um objeto de dados de uma base de dados. Para suporte à especificação desta operação, desenhámos uma pequena base de dados constituída por uma tabela - "Circulo" - constituída pelos seguintes atributos; "Id", "Nome", "Raio" e "Cor" (figura 2). Esta tabela acolherá a definição e caracterização dos elementos gráficos de suporte a um qualquer processo de interrogação.

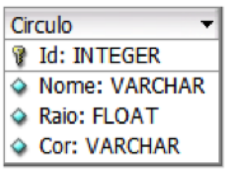


Figura 2 - Caracterização da entidade "Circulo"

Assim, a criação de um dado elemento gráfico representando um dado objeto de dados é realizada através da projeção de todos os atributos presentes na tabela ou vista em questão. Em SQL, esta operação pode ser representada através da seguinte *query*:

```
SELECT Id, Nome, Raio, Cor
    FROM Circulo;
```

que tem em Álgebra Relacional a seguinte representação:

$$\texttt{Resultado} \leftarrow \pi_{Id,Nome,Raio,Cor}(Circulo)$$

3.3.2 Especificação do Comando de Filtragem

Tal como em qualquer outro sistema ou linguagem de exploração de dados é necessário ter a possibilidade de por vezes restringir (filtrar) os resultados de uma dada operação sobre um elemento de dados. Não podemos disponibilizar apenas uma forma de consultar um objeto de dados, revelando todo o seu conteúdo, todos os seus registos. Como sabemos, é muito vulgar precisarmos de aplicar algum tipo de critério de filtragem para obtermos um conjunto mais fino de resultados. O comando de filtragem que desenvolvemos para o sistema de exploração de dados, na realidade resultou da combinação do operador de projeção, já especificado anteriormente, e do operador seleção. O comando de seleção, matematicamente, pode ser representado pelo símbolo σ. Tal como o comando de projeção, o comando de seleção é uma das operações primitivas da Álgebra Relacional. Com o objetivo de filtrar, o comando de seleção tem a capacidade de gerar subconjuntos dos dados contidos num dado objeto. Os elementos do subconjunto gerado pela operação, resultam dos elementos do conjunto inicial que respeitam a condição de filtragem indicada pelo utilizador.

De acordo com a sua definição original, a operação de seleção trabalha sobre os registos, as linhas, de uma dada tabela. Porém, no nosso sistema de exploração de dados, a operação de filtragem foi pensada e desenvolvida de forma a atuar tanto sobre registos e como sobre atributos. É, pois, uma operação que resultou da combinação das primitivas base das operações de seleção e de projeção da Álgebra Relacional. Para procedermos à sua especificação utilizámos, mais uma vez, a base de dados utilizada anteriormente, mantendo também o seu esquema geral (figura 2). Assim, o comando relativo à operação de filtragem permite, por exemplo, verificar quais os registos cujos valores do atributo "Id" é superior a '5', de resto uma

operação de filtragem bastante simples. Esta consulta pode ser realizada através da seguinte *query* em SQL:

```
SELECT Id, Nome, Raio, Cor
  FROM Circulo
  WHERE Id > 5;
```

tendo a seguinte representação em Álgebra Relacional:

$$\text{Resultado} \leftarrow \sigma_{Id>5}(\pi_{Id,Nome,Raio,Cor}(Bolha))$$

Em termos visuais, este comando tem algumas implicações uma vez que consegue alterar o número de atributos e o número de registos presentes no objeto de dados. Isto implica que os valores dos atributos "cor" e "raio" do elemento gráfico, que representam, respetivamente, a cardinalidade e o grau do objeto de dados em causa sejam calculados de acordo.

3.3.2 Especificação dos Comandos de Combinação de Dados

Uma das operações mais vulgares que se realiza num qualquer processo de manipulação de dados é a combinação da informação contida em diferentes tabelas (ou vistas). Muita da informação que esta ou aquela tabela contém está de alguma forma combinada com outra informação guardada numa outra tabela. É uma característica básica de uma base de dados relacional. Esta necessidade de combinar os dados de diferentes tabelas é uma necessidade real, bastante prática, que o próprio modelo relacional promove pela sua própria organização dos dados. Para este tipo de operação de combinação de dados a Álgebra Relacional tem duas primitivas que nos permitem realizar este tipo de ação, que são, nomeadamente, o produto e a união, duas típicas operações binárias.

Uma operação binária é uma função que, por exemplo, poderá ser de adição, subtração, divisão ou multiplicação, que contém duas variáveis de entrada (Weisstein n.d.) e apresentam algumas propriedades interessantes (Notare 2003):

- Comutatividade - seja $\oplus: A \times A \rightarrow A$ uma operação binária interna e fechada e x, y e z elementos quaisquer de A, então a operação $\oplus$ é associativa se:

  $$(\forall x)(\forall y)(\forall z)[x \oplus (y \oplus z) = (x \oplus y) \oplus z]$$

- Associatividade - seja $\oplus: A \times A \rightarrow A$ uma operação binária interna e fechada, e x e y dois elementos quaisquer de A, então a operação $\oplus$ é comutativa se:

  $$(\forall x)(\forall y)(x \oplus y = y \oplus x)$$

- Elemento neutro - seja $\oplus: A \times A \rightarrow A$ uma operação binária interna e fechada, e x um elemento qualquer de A, então a operação $\oplus$ tem elemento neutro se:

  $$(\exists e)(\forall x)(x \oplus e = e \oplus x = x)$$

- Elemento inverso - seja $\oplus: A \times A \rightarrow A$ uma operação binária interna e fechada, e x é um elemento qualquer de A, então a operação $\oplus$ tem elemento inverso se:

  $$(\forall x)(\exists x^{-1})(x \oplus x^{-1} = x^{-1} \oplus x = e)$$

Ambas as primitivas, produto e união, foram implementadas no sistema de exploração de dados que desenvolvemos, tendo dado origem, respetivamente, aos comandos produto e união. A forma como estes comandos atuam no sistema será apresentada e discutida no capítulo seguinte.

O Comando de Junção

A operação de junção – JOIN – é representada pelo símbolo ⋈ e é utilizada normalmente em operações de combinação de registos entre duas tabelas (Elmasri & Navathe 2010). O comando de junção que pretendemos integrar no nosso sistema de exploração é resultante da primitiva como mesmo nome da álgebra relacional. Como sabemos, é através desta primitiva que conseguimos fazer uma "fusão" entre tabelas de um dado sistema de dados, permitindo assim ao utilizador obter como resultado todas as combinações possíveis entre os registos das tabelas envolvidas com base num dado critério de junção.

Usualmente, os registos que resultam de uma operação de junção foram obtidos por combinação com base num critério de junção que envolve uma chave primária e uma chave estrangeira e que relaciona os registos nos quais as chaves referidas têm valores iguais (figura 3).

Empregado

Nome	IdEmp	DeptNome
Harry	3415	Finanças
Sally	2241	Vendas
George	3401	Finanças
Harriet	2202	Vendas

Departamento

DeptNome	Gerente
Finanças	George
Vendas	Harriet
Produção	Charles

Empregado ⋈ *Departamento*

Nome	IdEmp	DeptNome	Gerente
Harry	3415	Finanças	George
Sally	2241	Vendas	Harriet
George	3401	Finanças	George
Harriet	2202	Vendas	Harriet

Figura 3 - Ilustração das tabelas "Empregado", "Departamento" e a tabela resultante da sua junção

Ao observarmos a figura 3, facilmente percebemos o funcionamento de uma operação de junção. A operação de junção ilustrada envolve duas tabelas: "Empregado" e "Departamento". Se repararmos a tabela "Empregado" tem um atributo "DeptNome" que é uma chave estrangeira relativamente à tabela "Departamento", na qual está definido o atributo "DeptNome", que nesta tabela atua como chave primária. Com base nestes dois atributos temos a possibilidade de relacionar estas duas tabelas através de um critério de junção que envolve a igualdade de valores dos atributos referidos. A operação de junção sucede assente nesta igualdade. A operação de junção é também uma operação binária. Para demonstrarmos a forma como esta operação pode ser realizada utilizando já um dos nossos elementos gráficos, consideremos a utilização de uma nova tabela "Pessoa" de uma dada base de dados (figura 4). Através dessa figura, vemos que a nova tabela "Pessoa" contém dois atributos, nomeadamente "idPessoa" e "Nome". Quanto ao elemento de dados "Circulo" este mantém a mesma estrutura que foi revelada em exemplos anteriores.

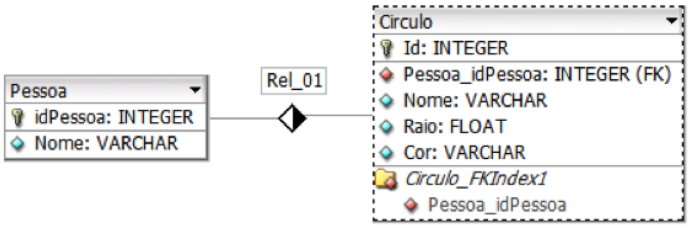


Figura 4 - Entidades Pessoa e Circulo

Na figura 4 vemos que está estabelecido um relacionamento entre os dois objetos de dados, um relacionamento 1:N. Assim, temos todas as condições para estabelecer uma operação de junção entre as duas tabelas, o que nos permitirá saber quais os círculos de cada um dos utilizadores. Em SQL, a *query* de junção pode ser representada da seguinte maneira:

```
SELECT Pessoa.idPessoa, Pessoa.Nome, Circulo.Id,
     Circulo.Cor, Circulo.Raio, Circulo.Nome
     FROM Pessoa INNER JOIN Circulo
          ON Pessoa.idPessoa =
     Circulo.Pessoa_idPessoa;
```

que em Álgebra Relacional pode ser representado pela seguinte expressão:

$$Resultado \leftarrow \pi_{Pessoa.idPessoa,Pessoa.Nome,Bolha.Id,Bolha.Nome,Bolha.Raio,Bolha.Raio} (\text{Circulo} \bowtie_{Circulo.Pessoa\_idPessoa=Pessoa.Id} Pessoa)$$

Como acontece em todos os comandos que forem realizados no sistema de exploração, o comando de junção provocará alterações nas representações dos elementos gráficos que representamos objetos de dados envolvidos na operação de junção. Assim, após a realização da operação desaparecerão os dois elementos gráficos (dois círculos) representantes dos dois objetos de dados "Circulo" e "Pessoa", sendo substituídos apenas um elemento gráfico representando o resultado da operação de junção, cuja caracterização será influenciada pelo número de registos e de atributos resultantes da junção. Essa representação será abordada durante o próximo capítulo.

O Comando de União

A operação de união entre duas relações R e S define uma relação que contem todos os registos de R, ou de S, ou de ambas as relações, na qual os registos duplicados são eliminados (Connolly & Begg 2004). Esta operação binária baseia-se na operação união da álgebra relacional e é representada pelo símbolo ∪. Para realizar esta operação criámos o comando união. A descrição deste comando é em muito semelhante à explicação apresentada para o comando junção. Mas, como sabemos, a operação de união atua de forma diferente da operação de junção. Vejamos então um pouco melhor como atua este comando. O comando de união junta numa mesma tabela os registos das duas tabelas envolvidas na operação. A sua forma de funcionamento é em todo semelhante à forma como atua a instrução UNION da linguagem SQL. Para que seja possível realizar a operação de união, os esquemas das duas tabelas envolvidas têm que ser compatíveis, ou seja, os seus esquemas deverão ter o mesmo número de atributos e estes atributos devem ser do mesmo tipo ou de um tipo compatível. Se estas condições se verificarem o comando será realizado e o resultado da operação integrará todos os registos das duas tabelas, mas com a eliminação dos registos que forem repetidos. Na figura 5 podemos ver uma ilustração do resultado de uma operação de união entre duas tabelas "R1" e "R2".

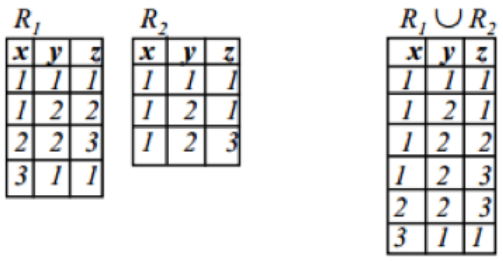


Figura 5 - Ilustração da união entre duas tabelas "R1" e "R2"

A operação de união ilustrada na figura 5 foi possível, uma vez que as duas tabelas envolvidas têm o mesmo grau e os tipos de dados dos atributos são compatíveis entre si. De referir que a tabela "R1" como a tabela "R2" possuem o registo x = 1, y = 1 e z = 1.Porém, na tabela resultante ($R_1 \cup R_2$) só se verifica a sua presença uma única vez, uma vez que a operação de união, por

omissão, faz a remoção dos registos duplicados na tabela resultante. Para realizar a sua especificação vamos usar uma pequena base de dados, muito simples, constituída apenas pelas tabelas "Circulo" e "Roda". A tabela "Circulo" tem dois atributos "IdCirculo" e "NomeCirculo", enquanto que a tabela "Roda" é constituída pelos atributos "IdRoda" e "NomeRoda".

Figura 6 - As tabelas "Roda" e "Circulo"

Esta operação pode ser especificada na prática através da seguinte query em SQL:

```
SELECT IdCirculo, NomeCirculo
     FROM Circulo
UNION
SELECT IdRoda, NomeRoda
     FROM Roda;
```

Por sua vez, esta query pode ser expressa em Álgebra Relacional através da seguinte expressão:

$$Resultado \leftarrow \pi_{IdRoda,NomeRoda}(Roda) \cup \pi_{IdCirculo,NomeCirculo}(Circulo)$$

Tal como aconteceu na especificação da operação de junção, a operação de união provoca também alterações nos elementos gráficos que envolve. Os dois círculos que representarão as duas tabelas envolvidas na operação de união, após a realização da operação de união darão origem a um só círculo, a só um elemento gráfico, cujas propriedades serão, obviamente, influenciadas pelo número de registos e de atributos resultantes da operação de união.

Capitulo 4

O Sistema de Exploração de Dados

4.1 Caracterização de um Caso de Estudo

Antes de passarmos à descrição da implementação do sistema que projetámos, vamos apresentar de forma resumida o caso de estudo que vamos utilizar como base para a demonstração dos comandos que implementámos. Em termos gerais, o caso de estudo pode ser enunciado da seguinte maneira:

> *O Carlos, docente no Externato Infante D. Henrique, é licenciado em Matemática e um entusiasta da área das tecnologias da informação e da comunicação (TIC). Para transmitir esse seu gosto aos seus alunos, e por achar que isso os ajudaria no seu desenvolvimento no domínio, criou o núcleo de tecnologias da informação, um órgão com o objetivo de propiciar um espaço de trabalho, de discussão e de partilha de conhecimento no domínio das TIC. Para ter acesso a ferramentas e material didático, o Carlos entrou em contacto com o projeto CODE.ORG para avaliar a possibilidade deste propiciar tal apoio ao núcleo TIC. A resposta do projeto foi positiva e entusiasmante, resultando no fornecimento imediato de todo o material que o Carlos definiu como necessário. Entre as aplicações disponibilizadas encontrava-se um sistema gráfico para a interrogação de bases de dados. Com o intuito de introduzir o tema das bases de dados nas suas áreas de lecionação e trabalho, o Carlos decidiu utilizar essa ferramenta na conceção e desenvolvimento*

de uma base de dados, bastante simples, para acolher as compras de material feitas na papelaria da escola. No Externato Infante D. Henrique os alunos têm à sua disposição um pequeno quiosque, no qual podem carregar os seus cartões de estudante com algumas quantias financeiras. Este cartão de estudante, além de servir para identificar o aluno, serve também para efetuar compras dentro do recinto escolar, sendo a única forma de pagamento aceite. Nas suas infraestruturas, o Externato Infante D. Henrique conta com uma pequena papelaria, que tem a capacidade de vender todo o tipo de material que os seus alunos precisam para o seu dia-a-dia escolar. Nessa papelaria, sempre que é realizada uma aquisição é emitido o respetivo talão de compra. Neste talão figura o funcionário que registou a venda, os produtos adquiridos e respetivas quantidades, e o total da compra efetuada.

Uma análise breve ao caso apresentado permite-nos fundamentar e desenhar uma base de dados relacional (figura 7) com capacidade para acolher, em linhas gerais, a informação associada com o enunciado realizado. Será esta a base de dados que utilizaremos como suporte a todos os processos de demonstração que de seguida iremos apresentar.

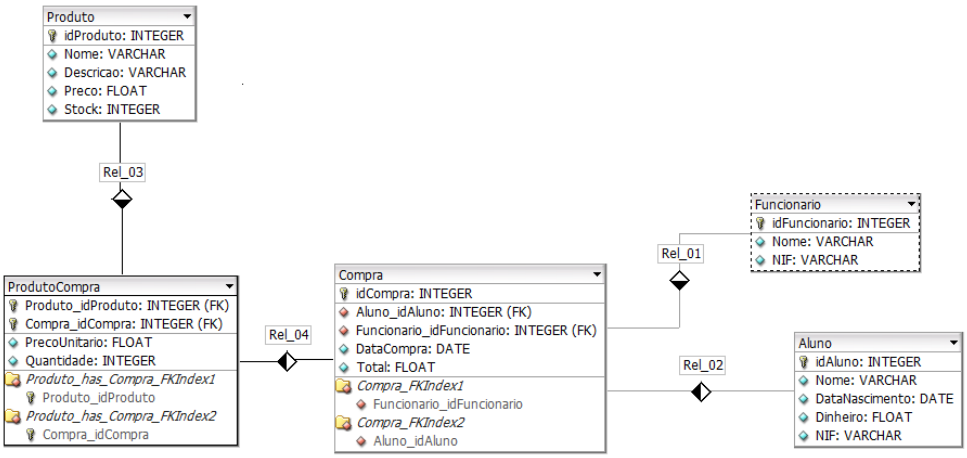


Figura 7 - Esquema da base de dados do caso de estudo escolhido

4.2 Tecnologia e Ferramentas Utilizadas

Desde o primeiro esboço que efetuámos sobre sistema que pretendíamos desenvolver, definimos que seria orientado para ser utilizado e explorado exclusivamente em ambiente Web, através de um *browser* convencional.

Basicamente, pretendíamos que o explorador de dados fosse independente de qualquer sistema operativo e livre de uma instalação complicada. Assim, utilizámos um servidor Windows Server 2012 como plataforma base de trabalho e, para o desenvolvimento propriamente dito do explorador, as linguagens HTML, CSS, PHP e JavaScript. A seleção destas últimas deveu-se, simplesmente, ao facto de termos já alguma experiência no desenvolvimento de aplicações com estas linguagens e por se encontrarem, também, muito bem documentadas em vários recursos Web. Com as CSS e a HTML desenvolvemos o *layout* da ferramenta, enquanto que para programar as respostas às ações do utilizador utilizámos o JavaScript, que para além disto nos permitiu também desenvolver os elementos gráficos que fomos implementando. Quanto à linguagem PHP, esta foi utilizada para desenvolver toda a parte algorítmica do sistema de exploração, bem como todos os processos de comunicação que se realizam entre si e o sistema de bases de dados.

Seguindo a mesma opção de desenvolvimento – todas as linguagens escolhidas são de tecnologia Web -, selecionámos como *software* para a base de desenvolvimento o Adobe Dreamweaver, uma vez que este possibilita uma maneira muito fácil e expedita a interação entre as várias linguagens utilizadas no desenvolvimento do sistema de exploração.

O armazenamento de dados do sistema ficou a cargo do sistema de gestão de base de dados MySQL, que está incluído na ferramenta de desenvolvimento Web phpMyAdmin e utiliza a linguagem SQL como linguagem base para a manipulação de sistemas de dados relacionais. Além disso é um SGBD amplamente usado pela comunidade TIC (IT 2012), o que faz com que este sistema esteja muito bem documentado, característica esta fundamental no processo de exploração e utilização dos metadados do próprio sistema.

4.3 Utilização do Sistema de Exploração

Ao longo dos últimos tempos o desenvolvimento de linguagens e ferramentas para sistemas de base de dados tem aumentado de forma acentuada. Porém, as ferramentas que desse processo emergiram não demonstraram a acessibilidade requerida pelos utilizadores que se enquadram no perfil de

exploração que designámos dos "7 aos 77". Todavia, os vários esforços que hoje se desenvolvem nesse âmbito apontam para soluções que se aproximam já de um patamar de exploração bastante interessante e fácil de utilizar. Com esse intuito foram sendo desenhadas e implementadas linguagens visuais, que introduziram inúmeros aspetos inovadores, desde a utilização de ícones até ao uso de formulários e de diagramas. Nada de novo. Já Catarci et al (1997) apontaram nessa direção. Apesar disso, a interação entre o utilizador e as ferramentas, por norma, ainda depende muito de periféricos como os ratos e os teclados. Limitar as ferramentas de exploração de dados a este tipo de periféricos nos dias de hoje não tem sentido. A evolução da tecnologia disponível ao público em geral não tem conhecido barreiras. Os computadores fixos estão a "ficar um pouco para trás", estando a dar lugar, a um ritmo muito acelerado, a aparelhos de bolso, de grande mobilidade. Esta tendência abre um grande leque de oportunidades (e de aplicações) para todos os sistemas de exploração, o mesmo acontecendo para as ferramentas no domínio dos sistemas de bases de dados.

Muito dos aspetos de utilização e amigabilidade referidos foram tomados em consideração no planeamento, desenho e implementação do nosso sistema de exploração de dados. Queríamos distinguir esta ferramenta de todas as outras ferramentas disponíveis no mercado orientadas especificamente para a exploração de bases de dados relacionais. Assim, o sistema de exploração deveria ser desenhado para aparelhos atuais, obviamente com *touch screen,* em que qualquer tipo de operação seria despoletada exclusivamente por ações realizadas por um utilizador sobre o ecrã ou de forma automática pelo próprio sistema. Tendo isto em consideração concebemos a ferramenta orientada para os novos dispositivos computacionais, com particular preferência para as plataformas *tablet*. Além disso, pretendíamos que o sistema de exploração assentasse numa plataforma de utilização universal (ou quase), o que implicou que o seu desenvolvimento fosse realizado especificamente orientado para que o sistema fosse executado através de um *browser* convencional. Apesar dos *browsers* correrem em sistemas móveis, não faz com que seja obrigatório o reconhecimento por parte deles dos movimentos característicos sobre um ecrã

(e.g. *swipe, tap hold*, e outros) por parte dos sistemas operativos direcionados para estes aparelhos. Para resolver esta situação utilizámos uma biblioteca em JavaScript (Major 2011). Nas seções seguintes veremos como tudo isto foi realizado e apresentaremos o sistema de exploração desenvolvido.

4.3.1 A Seleção da Base de Dados de Trabalho

Ao entrar no endereço do sistema de exploração, o utilizador irá deparar-se com um simples formulário que gere o acesso (*login*) a qualquer instalação do MySQL. Neste formulário o utilizador deverá introduzir o endereço do servidor a que quer aceder, o seu nome de utilizador bem como a sua chave de acesso (*password*), e indicar o número da porta de acesso ao servidor, caso este não esteja a usar a porta definida por omissão (a porta 3306). Todos os dados solicitados são completamente independentes do tipo da aplicação a queremos aceder, dependendo apenas do tipo instalação do MySQL realizada.

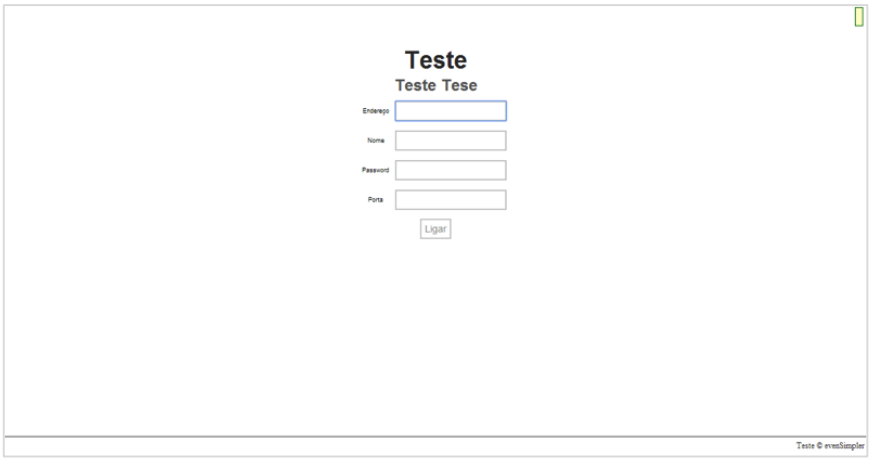


Figura 8 - Acesso ao sistema de exploração

Se os dados apresentados estiverem corretos, o acesso ao sistema de exploração será concedido ao utilizador. Após o *login* ter sido efetuado por parte do sistema de exploração no MySQL, teremos acesso ao seu catálogo de bases de dados. Para que o sistema de exploração possa mostrar no seu ambiente as bases de dados que estão disponíveis para exploração ele pede a execução da seguinte *query*:

```
SHOW DATABASES;
```

Baseado no resultado da execução da *query* anterior, que nos revela as bases de dados disponíveis, o sistema de exploração constrói um menu (figura 9) contendo as várias opções de consulta. É através desse menu que o utilizador escolherá a base de dados com que pretende trabalhar.

Figura 9 - Lista das bases de dados disponíveis para exploração

Vemos, assim, como é fácil selecionar a base de dados pretendida para exploração. Para o nosso caso, em particular, vamos escolher a base de dados "teste_reprografia" através da respetiva representação no botão do menu atrás referido. Após a seleção da base de dados de trabalho o sistema de exploração colocar-nos-á no ambiente de exploração de dados, o principal ambiente do sistema (figura 10). O ambiente de exploração é constituído por um único painel branco, com três botões de comandos. O painel é o local onde tudo acontece. É nele que surgem os diversos elementos gráficos com os quais o utilizador irá trabalhar nos seus processos de exploração da base de dados selecionada.

Figura 10 - O ambiente de trabalho do sistema – o painel de exploração

Além do painel de exploração, o ambiente principal do sistema incorpora três botões de ações, colocados à esquerda do ecrã, cada um com um objetivo bem definido. O botão '1' permite ao utilizador regressar ao ecrã anterior, que, como vimos, permite fazer a seleção da base de dados com que queremos trabalhar. Para ajudar na realização de alguns comandos mais complexos e também para permitir visualizar a informação contida nas tabelas foi criado um segundo botão de comandos (o botão 2). Este botão abre uma janela que além de permitir a visualização da informação contida nas tabelas, possibilita visualizar o resultado de diferentes operações como a junção, união entre outros. O terceiro botão revela ao utilizador a lista de todos os objetos da base de dados (tabelas e vistas) que ele pode utilizar nos seus processos de consulta. Essa lista é criada através das seguintes *querys* em SQL:

```
SHOW FULL TABLES IN teste_reprografia
     WHERE TABLE_TYPE LIKE 'BASE TABLE';
```

e

```
SHOW FULL TABLES IN teste_reprografia
     WHERE TABLE_TYPE LIKE 'VIEW';
```

que nos disponibilizam as tabelas base e as vistas, respetivamente, disponíveis na base de dados selecionada.

4.3.2 A Representação de um Objeto de Dados

Todos os objetos definidos e utilizados no sistema são representados por elementos gráficos, que na atual versão do sistema se limitam a representações circulares. Qualquer processo de exploração de dados desenvolve-se em torno de tais elementos gráficos, que vão sendo colocados no painel de exploração do sistema à medida que vão sendo requeridos à base de dados selecionada pelo utilizador. Para que se possa requerer uma tabela (ou vista) e coloca-la no painel para exploração posterior utilizamos um comando explicitamente desenvolvido para esse efeito, que será também o responsável pela representação das respetivas representações visuais. É um comando bastante simples – está assinalado com o número 3 na figura 10 -, que trás para o sistema uma lista (figura 11) com os objetos de dados disponíveis na base de dados.

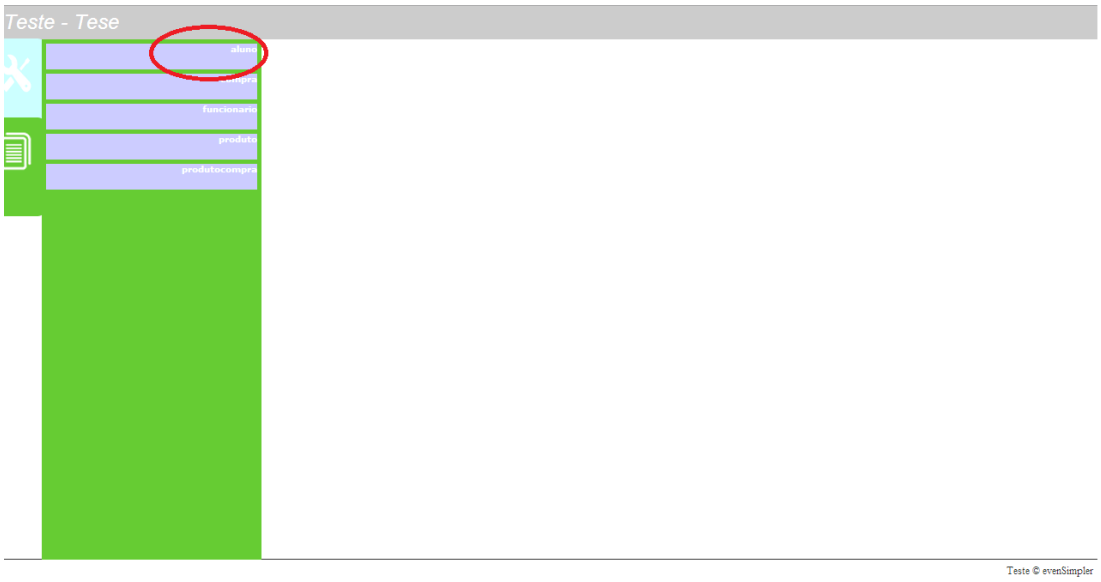


Figura 11 Lista de entidades e vistas existentes na base de dados

Assim, para iniciar o seu trabalho, o utilizador apenas terá de selecionar na lista apresentada o objeto de dados com que pretende trabalhar. Para efeitos de demonstração, neste caso, selecionámos a tabela "aluno". Com essa seleção aparecerá no painel de exploração um círculo, etiquetado com a referência "aluno", que representará o objeto de dados "aluno" (figura 12).

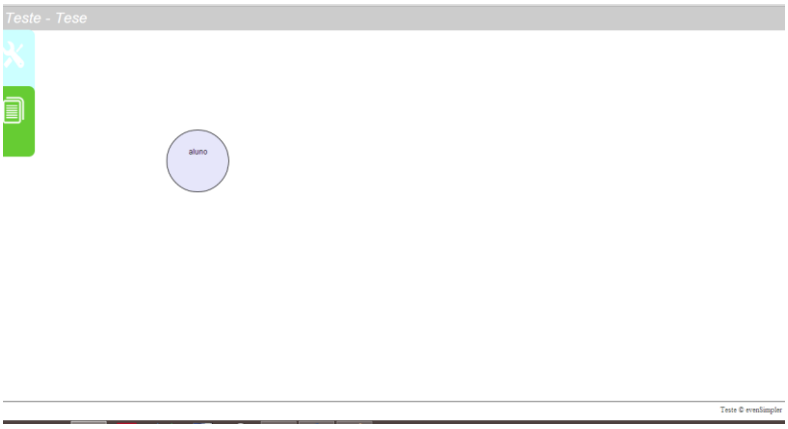


Figura 12 - Representação visual da tabela "Aluno" através do seu elemento gráfico respetivo no painel de exploração do sistema

Porém, para se conseguir a representação visual da tabela selecionada é necessário coletar informação, tal como já foi referido no capítulo anterior. Para angariar essa informação o sistema de exploração recorre à seguinte *query*:

```
SHOW COLUMNS
     FROM aluno;
```

Depois, esta *query* será executada pelo MySQL, que devolverá o seu resultado na forma como está apresentado na figura 13.

Field	Type	Null	Key	Default	Extra
idAluno	int(10) unsigned	NO	PRI	*NULL*	auto_increment
Nome	varchar(250)	YES		*NULL*	
DataNascimento	date	YES		*NULL*	
Dinheiro	float	YES		0	
NIF	varchar(11)	YES		*NULL*	

Figura 13 - Informação contida na tabela "aluno" no MySQL

Depois, a informação resultante da execução da *query* é tratada e enviada em formato XML para o sistema de exploração. Este formato permitirá que os scripts escritos em JavaScript possam reconhecer e trabalhar os dados. Os dados recebidos são usados para criar a representação visual de acordo com os parâmetros definidos (grau e cardinalidade). Quanto à informação relativa à tabela acedida ("aluno") esta será armazenada num *array*. Este *array* permitirá ao sistema de exploração saber sempre a posição no ecrã na qual se encontra o

elemento gráfico que representa a consulta realizada, bem como os atributos e respetivos tipos de dados.

4.3.3 Seleção de Colunas e Registos de um Objeto de Dados

A partir do momento em que o utilizador tem a permissão para gerir um dado elemento gráfico, um dado objeto de dados, ele tem a possibilidade de realizar uma série de pequenas tarefas bastante relevantes em qualquer processo de exploração de dados. Além de poder deslocar o objeto de dados no painel de exploração, por exemplo, para arrumar o seu espaço de trabalho, ele pode também consultar a informação no objeto de dados, bem como aplicar algum tipo de critério de filtragem para ajustar objeto às suas necessidades de exploração, o que pode implicar também operações de combinação de dados com outros objetos presentes no painel de exploração. Vejamos como é que o caso da consulta da informação de um dado objeto de dados pode ser realizada. Se um dado utilizador pretende realizar tal operação, ele apenas tem que realizar uma operação de *doubletap* sobre elemento gráfico correspondente ao objeto de dados que deseja consultar. Como consequência desta operação aparecerá sobre o painel de exploração uma tabela com a informação solicitada (figura 14), devolvendo o comando, também, a *query* que executou para apresentar os resultados requeridos. Para este caso em particular, a *query* devolvida foi a seguinte:

```
SELECT idAluno, Nome, DataNascimento, Dinheiro, NIF
    FROM aluno;
```

Figura 14 - Informação contida no objeto de dados "aluno"

Como já referimos, existem duas formas de obter a informação de uma dada tabela que foi associada a um elemento gráfico específico. Essas duas formas são: efetuar um *doubletap* ou *taphold* sobre o elemento gráfico correspondente. Qualquer uma dessas operações faz aparecer um pequeno menu sobre o elemento gráfico (figura 15).

A partir deste menu, o utilizador poderá escolher a opção 'Selecionar', que lhe permitirá aceder a um formulário (figura 16) no qual figuram todos os atributos da tabela representada pelo elemento gráfico em questão. Nessa altura, o utilizador selecionará os atributos pretendidos (seleção de colunas).

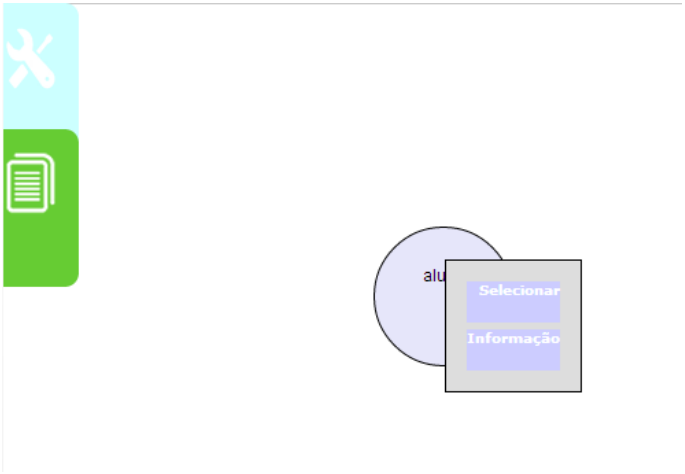


Figura 15 - Menu resultante de uma ação taphold sobre um elemento gráfico específico

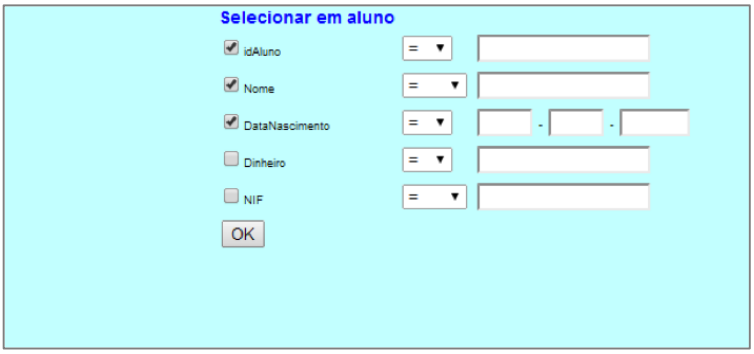


Figura 16 - Seleção de atributos de uma tabela com base num elemento gráfico

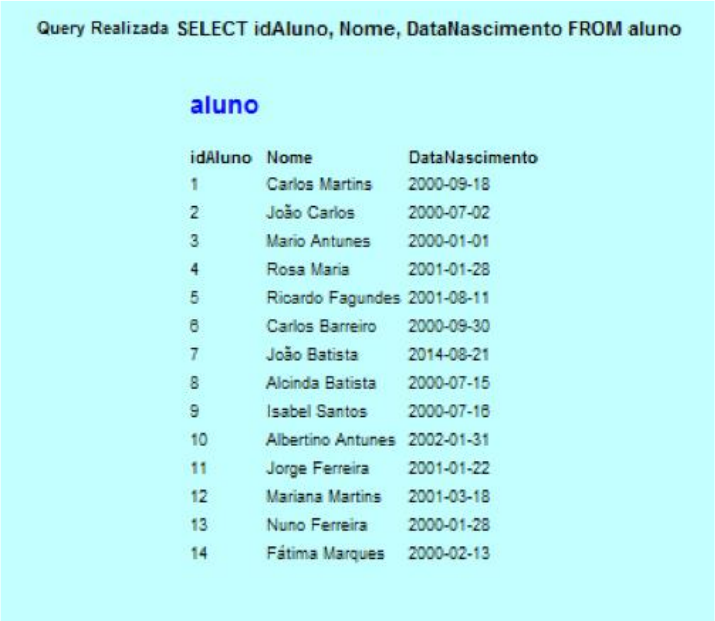
Para este caso em particular, e com base no elemento gráfico que representa a tabela "aluno", selecionámos os atributos "idAluno", "Nome" e "DataNascimento". Para visualizarmos o conteúdo da tabela de acordo com os atributos selecionados apenas temos que confirmar esta última operação carregando no botão 'OK' presente no formulário – aqui podemos também observar, na parte superior do formulário, a *query* que foi realizada para a obtenção da informação obtida.

Com base na figura 17, podemos ver as alterações que ocorreram ao nível do grau da relação, passando o elemento gráfico agora a representar a mesma tabela mas agora apenas com três atributos em vez dos anteriores cinco. Anteriormente o sistema de exploração tinha despoletado a *query*:

```
SELECT idAluno, Nome, DataNascimento, Dinheiro, NIF
     FROM aluno;
```

enquanto que agora, após a seleção de atributos ter sido efetuada pelo utilizador, o sistema utilizou a *query*:

```
SELECT idAluno, Nome, DataNascimento
     FROM aluno;
```



Query Realizada SELECT idAluno, Nome, DataNascimento FROM aluno

aluno

idAluno	Nome	DataNascimento
1	Carlos Martins	2000-09-18
2	João Carlos	2000-07-02
3	Mario Antunes	2000-01-01
4	Rosa Maria	2001-01-28
5	Ricardo Fagundes	2001-08-11
6	Carlos Barreiro	2000-09-30
7	João Batista	2014-08-21
8	Alcinda Batista	2000-07-15
9	Isabel Santos	2000-07-16
10	Albertino Antunes	2002-01-31
11	Jorge Ferreira	2001-01-22
12	Mariana Martins	2001-03-18
13	Nuno Ferreira	2000-01-28
14	Fátima Marques	2000-02-13

Figura 17 - Informação contida na tabela "aluno" resultante de um processo de seleção

Utilizando o formulário anterior (figura 16), mas agora com a sua nova configuração de atributos (figura 18), podemos também aplicar um processo de seleção dos registos que queremos obter. Assim, selecionando todos os atributos, devido à última operação de seleção de atributos neste caso apenas estarão presentes os atributos "idAluno", "Nome" e "DataNascimento", vamos aplicar um filtro de seleção sobre os valores de um desses atributos, por exemplo sobre o atributo "idAluno". O filtro que escolhemos permitirá apresentar apenas os registos da tabela cujo valor do atributo selecionado seja superior a cinco (figura 18).

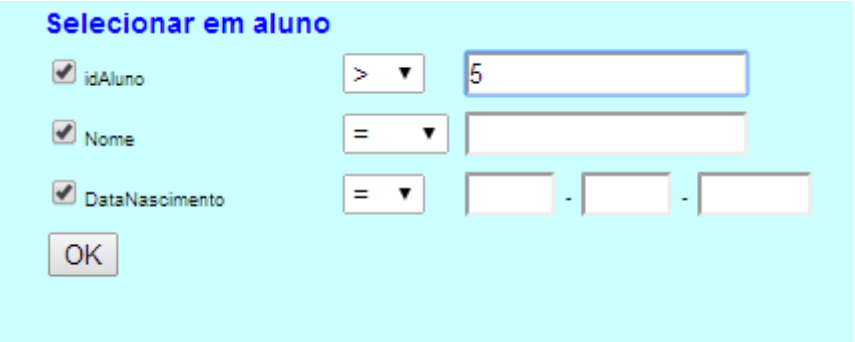


Figura 18 - Ilustração de uma operação de seleção de registos com base num dado atributo

Quando o utilizador confirmar esta operação de seleção, uma operação que afetará com certeza a cardinalidade da tabela em visualização. Após a operação de seleção ser confirmada, aparecerá, à semelhança do caso anterior, a informação que satisfez as condições de filtragem definidas, juntamente com a *query* que foi despoletada pelo sistema (figura 19).

Figura 19 - Informação contida no objeto de dados "aluno" após realização da operação de seleção

A *query* resultante desta última operação de seleção:

```
SELECT idAluno, Nome, DataNascimento
     FROM aluno
     WHERE idAluno > 5;
```

apresenta uma particularidade específica, que ainda não tinha sido verificada nos casos que apresentámos até ao momento. Nada de extraordinário. Mas aqui podemos ver, mais uma vez, a forma como o sistema vai configurando a instrução SELECT de acordo com as características que vamos definindo (ou redefinindo) sobre os elementos gráficos que representam os objetos de dados.

Além disso esta operação provocou alterações ao nível da representação gráfica do objeto de dados "aluno", tanto ao nível do seu tamanho com da sua cor, refletindo assim os novos "valores" das suas propriedades grau e cardinalidade.

Figura 20 - O novo aspeto do elemento gráfico "aluno"

4.3.4 Junção entre dois Objetos de Dados

No explorador de dados o comando de junção, tal como o seu próprio nome indica, permite realizar uma operação de junção entre dois objetos de dados. A sua execução assenta na instrução INNER JOIN da linguagem SQL. Como tal a sua execução segue as normas estabelecidas para essa instrução SQL. Para representarmos visualmente a execução deste comando, desta operação, apoiámo-nos no significado da palavra junção encontrada num dicionário (Editora n.d.): "*ponto onde duas ou mais coisas se reúnem ou se ligam; confluência, convergência*". A partir desta descrição definimos que o comando de junção seria solicitado (executado) sempre que dois objetos de dados, dois círculos que representassem duas tabelas de dados, estivessem em contacto, sem que, para isso, o ponto central dos círculos dos objetos não estivesse dentro do raio de qualquer um dos outros (figura 21).

Assim, o utilizador para realizar uma operação de junção entre dois objetos de dados, representados como sabemos por dois círculos no ambiente de exploração, apenas terá aproximar os dois elementos gráficos um do outro, com uma pequena sobreposição entre eles, e depois realizar um *doubletap* sobre o objeto que quer que represente a relação final, isto é que assuma os resultados da operação em causa. Na realização desta operação de junção, o comando solicitado pede ao MySQL para realizar a seguinte instrução em SQL:

```
Select concat(table_name, '.', column_name)
   as 'foreign key', concat(referenced_table_name,'.',
   referenced_column_name) as 'references'
   FROM information_schema.key_column_usage
        WHERE referenced_table_name = 'aluno' and
             table_name = 'compra';
```

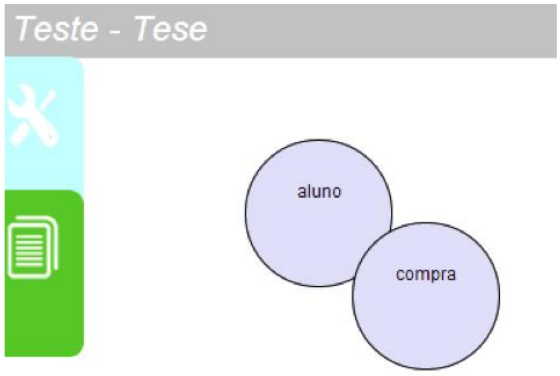


Figura 21 - Representação visual de uma operação de junção envolvendo dois objetos de dados, as tabelas "aluno" e "compra"

Na figura 22 podemos observar o reconhecimento desta operação de junção no ambiente do MySQL, que apresenta a "tradução" do comando solicitado graficamente já para a linguagem SQL.

Figura 22 - Query resultante da tradução da operação de junção para SQL

Após a verificação da possibilidade de realização da operação de junção solicitada, ou seja identificar um possível relacionamento válido entre as tabelas envolvidas na operação, surgirá no painel de exploração uma *query* que permitirá fazer a criação de uma *view* para refletir o resultado da junção (figura 23). A criação de um *view* para estes casos foi opção nossa.

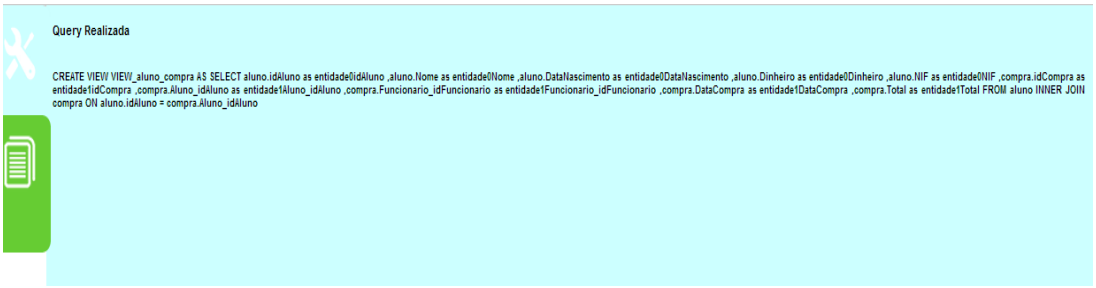


Figura 23 - Apresentação da query ao Utilizador

Basicamente, com isso pretendemos criar um meio para que depois da operação de junção fosse realizada o seu resultado pudesse ser utilizado em operações de exploração ou de combinação posteriores. Neste caso, em particular, a *view* que foi criada foi a seguinte:

```
CREATE VIEW VIEW_aluno_compra
AS
     SELECT aluno.idAluno as entidade0idAluno,
          aluno.Nome as entidade0Nome,
          aluno.DataNascimento as
          entidade0DataNascimento,
          aluno.Dinheiro as entidade0Dinheiro,
          aluno.NIF as entidade0NIF,
          compra.idCompra as entidade1idCompra,
          compra.Aluno_idAluno as
          entidade1Aluno_idAluno,
          compra.Funcionario_idFuncionario as
          entidade1Funcionario_idFuncionario,
          compra.DataCompra as entidade1DataCompra,
          compra.Total as entidade1Total
     FROM aluno INNER JOIN compra
          ON aluno.idAluno = compra.Aluno_idAluno.
```

As *views* têm uma representação gráfica específica no sistema de exploração, de forma a puderem ser distinguidas dos outros elementos gráficos que representam tabelas base. Tal como as tabelas, estas são representadas por um círculo, mas com uma representação o tracejado (figura 24).

Figura 24 - Representação gráfica da view gerada a partir da operação de junção realizada

A partir das *views* que foram geradas pelos utilizadores podemos realizar qualquer operação de manipulação de dados, tal como se tratasse de um objeto de dados base, representando uma qualquer tabela da base de dados, algo semelhante ao que acontece na linguagem SQL relativamente à manipulação de *views*. Porém, tivemos que criar um comando específico para que pudéssemos também fazer a sua remoção do sistema de exploração, algo parecido com o tradicional comando da SQL DROP VIEW – este comando será analisado numa próxima seção.

Retomemos o nosso caso da operação de junção entre os dois objetos de dados envolvidos: "aluno" e "compra". Neste caso em particular a operação de junção foi de fácil realização uma vez que existia uma chave estrangeira entre as tabelas envolvidas. Contudo, caso isso não acontecesse, o utilizador teria que realizar *doubletap* para indicar qual seria o critério de junção a seguir. Com o *doubletap* aparece no ecrã um pequeno formulário, que tem como objetivo requerer ao utilizador que indique quais serão os atributos que servirão de base à realização da operação de junção (figura 25).

Figura 25 - Menu para a escolha dos elementos envolvidos numa junção

O formulário apresentado é criado de uma forma dinâmica, sendo construído com base nos metadados associados com o objeto de dados em questão. Para construir este formulário o sistema de exploração tem como base de trabalho a seguinte *query*:

```
SELECT column_name, column_type
    FROM information_schema.columns
    WHERE table_name =  '$sug' AND
              column_type like '$type%'
```

Na *query* apresentada, $sug será a variável que acolherá o nome da tabela em que queremos procurar e a variável $type um determinado tipo de dados.

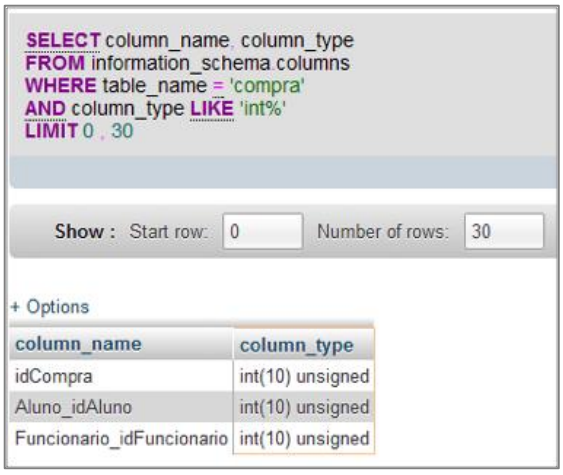


Figura 26 - Resultado da query de configuração, com as variáveis $sug e $type definidas, respetivamente, com os valores "compra" e "int"

Assim, neste caso, será realizada uma operação de junção com base nos atributos "idAluno" e "Aluno_idAluno" das tabelas "aluno" e "compra", respetivamente. Ao se confirmar esta configuração, o sistema de exploração realizará a operação de junção entre as tabelas referidas, criando uma *view* como resultado. Todas as *views* que foram criadas por um dado utilizador serão automaticamente removidas do sistema após o utilizador ter terminado a sua sessão de exploração, mantendo-se a base de dados na sua configuração e estado iniciais, como de resto seria de esperar.

4.3.5 A União entre Dois Objetos de Dados

Ao contrário do comando de junção, o comando de união não combina os registos das entidades que estão envolvidas na operação. Este faz algo um pouco diferente. Ao realizar a união entre duas tabelas o comando de união coloca "todos" os registos dos objetos de dados envolvidos num único objeto de dados. O comando em SQL é representado pela palavra UNION, e as entidades envolvidas necessitam de ter o mesmo grau, e os tipos dos atributos tem que ser os mesmos, tal como já referimos anteriormente. Um pouco à semelhança do que ocorreu como comando de junção, tivemos algumas dificuldades em encontrar a sua melhor representação gráfica. Todavia, seguindo a estratégia que adotámos para o comando de junção, decidimos que o sistema

reconheceria uma operação de união, quando os dois elementos gráficos envolvidos se encontrassem em contacto, mas em que o centro de um dos círculos tem que se encontrar dentro do raio do outro círculo (figura 27).

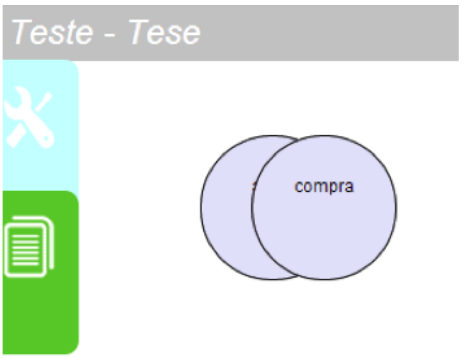


Figura 27 - Representação gráfica de uma operação de união

A forma como se realiza a união é, pois, muito semelhante à da junção. O posicionamento de um elemento gráfico (um círculo) relativamente ao outro é que muda. Depois disso, atuamos da mesma maneira. Fazendo um *doubletap* sobre um dos objetos de dados envolvidos aparecerá no ambiente de exploração um menu (figura 28). Através desse menu o utilizador poderá escolher quais os atributos que quer utilizar em cada uma das tabelas envolvidas, para que estes sejam incluídos no resultado final da operação de união.

Figura 28 - Menu para a seleção de atributos numa operação de união

No caso apresentado na figura 28, o utilizador estará a realizar uma união entre as tabelas "aluno" e "compra", estando a envolver os atributos "idAluno", "Nome" da tabela "aluno" e o atributo "idCompra" da tabela "compra". Tal

como acontece no comando de junção, tomámos a decisão de criar uma *view,* pelos mesmos motivos que expressámos para o caso da operação de união. A *view* gerada para este caso foi criada a parir da seguinte instrução SQL:

```
CREATE VIEW VIEW_aluno_compra
AS
SELECT idAluno ,Nome FROM aluno
UNION
SELECT idCompra ,NULL FROM compra;
```

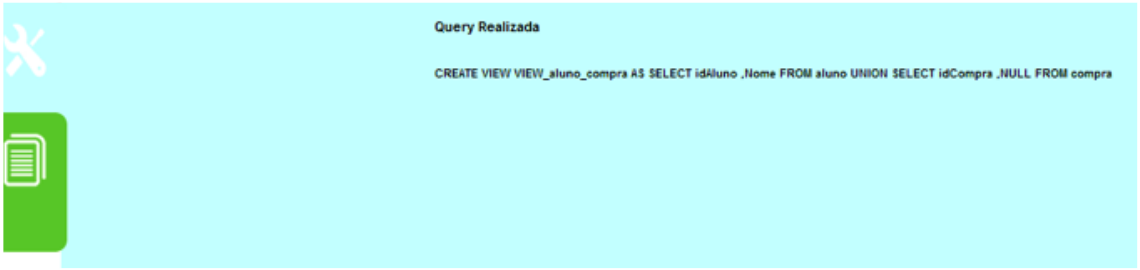


Figura 29 - Query executada com a execução do comando de uma operação de união

Depois da criação da vista aparece no painel de exploração do sistema o elemento gráfico correspondente. Tal como no comando de junção, a representação da *view* é um círculo a tracejado (figura 30).

Figura 30 - Query executada com a execução do comando de uma operação de união

4.3.6 Remoção de Vistas de Dados

O comando para a remoção de uma vista - DROP - foi desenhado e implementado com vista à eliminação das vistas criadas através de operações de combinação de dados, como a junção ou a união. Para realizar uma operação de remoção de uma vista o utilizador apenas terá que fazer um *swipedown* usando a vista que pretende remover do sistema de exploração. Ao realizar uma operação de remoção de uma vista aparecerá no ecrã a *query* que

o sistema fez gerar para realizar a operação de remoção requerida. Assim, aparecerá uma *query* como a revelada na figura 31.

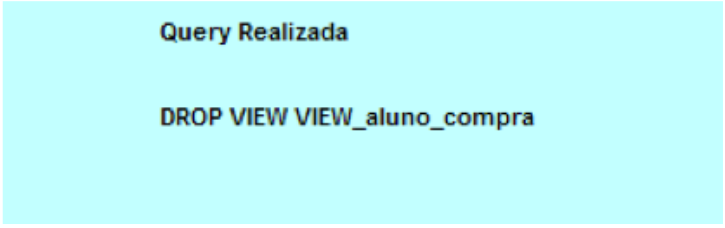


Figura 31 - Query apresentada ao utilizador para a realização da remoção de uma vista

4.3.7 Geração de Gráficos sobre Dados

De forma a completar um pouco mais as atuais funcionalidades do sistema de exploração de dados, encetámos um pequeno processo de desenvolvimento adicional: prover o sistema com mecanismos para suporte à análise de dados através de gráficos. Na realidade, pretendíamos somente demonstrar uma nova vertente de exploração, tão apetecível para quadros de gestão, que acrescentasse ainda mais valor ao interface que concebemos e implementámos. Assim, além da simplicidade de exploração de objetos de dados através de elementos gráficos, o sistema de exploração revela agora a possibilidade de visualizar os resultados de uma qualquer operação de exploração de dados através de gráficos (obviamente, com alguns condições básica de preparação para a geração de um gráfico, por exemplo, quando o objeto de dados produzido possua atributos do tipo data e atributos do tipo numérico – mais tarde, numa próxima evolução serão adicionadas novas características.

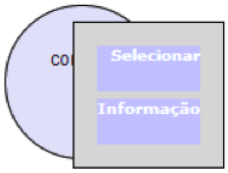


Figura 32 - Menu de apoio à geração de um gráfico

Para proceder à realização de um gráfico sobre um dado objeto de dados, um utilizador apenas necessita de escolher esse objeto, neste caso o objeto "compra", e realizar um *taphold* sobre a mesma. Como consequência aparecerá

um menu (figura 32) no painel de exploração do sistema no qual o utilizador deverá escolher a opção 'Informação'. Com a seleção dessa opção aparecerá no ecrã o menu apresentado na figura 33.

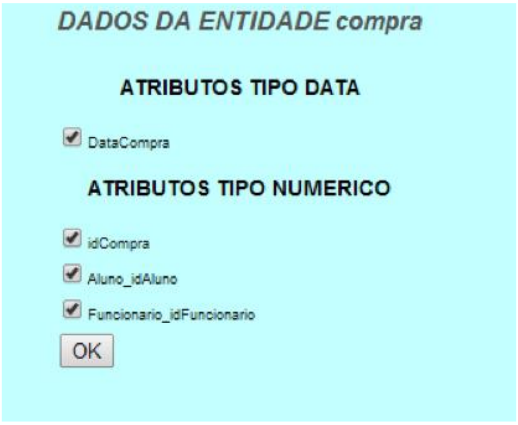


Figura 33 - Menu para a seleção de atributos a incluir num gráfico sobre os dados obtidos

Na atual versão do sistema de exploração, para que seja possível gerar um gráfico é necessário escolher um atributo do tipo data, neste caso selecionámos o atributo "DataCompra" (figura 33), e pelo menos um atributo numérico, para o nosso caso de demonstração escolhemos três atributos, nomeadamente: "idCompra", "Aluno_idAluno", "Funcionario_idFuncionario". Quando terminarmos a operação de seleção dos atributos a integrar na estrutura do gráfico que pretendemos, bastará confirma a operação através do botão "OK". O gráfico solicitado e configurado aparecerá de seguida (figura 34). Para fazermos a implementação do módulo de geração de gráficos recorremos a uma biblioteca de programas disponível para a linguagem PHP denominada por Libchart (Trémeaux 2011).

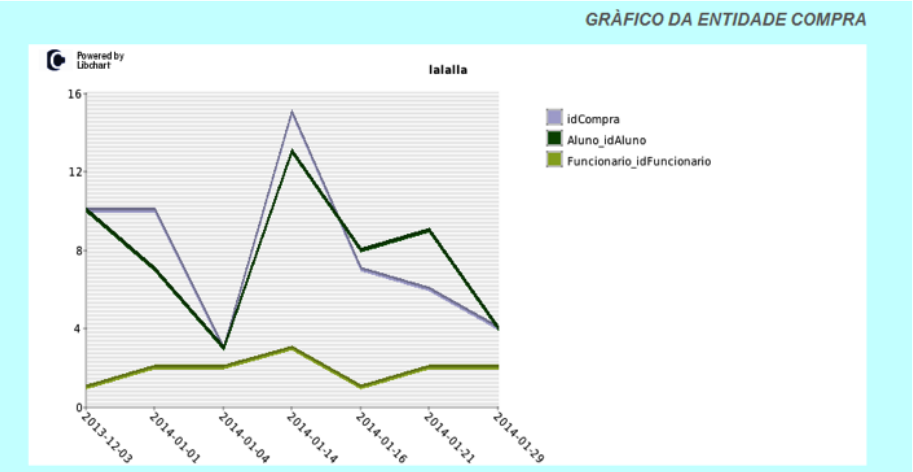


Figura 34 - Gráfico gerado a partir dos resultados de uma operação de consulta sobre um dado objeto de dados

4.4 Uma Pequena Súmula Final

Neste capítulo preocupámo-nos em apresentar as funcionalidades mais relevantes do sistema de exploração de dados desenvolvido ao longo desta dissertação. Esta foi a maneira que achámos mais conveniente (e direta) para demonstrar a utilidade do sistema de exploração desenvolvido, bem como apresentar cada uma das suas funcionalidades e serviços, acompanhando sempre que possível a sua demonstração com exemplos de aplicação bastante concretos. Desde simples consultas sobre objetos de dados específicos, até à realização de operações mais complexas, envolvendo comandos de junção ou de união, fomos revelando como é que de uma forma simples (e gráfica) este explorador cumpre o objetivo para o qual foi planeado: disponibilizar uma nova experiência de exploração sobre bases de dados a utilizadores dos "7 aos 77" anos. Terminamos este capítulo deixando na tabela 1 uma breve relação das operações que podemos realizar com a atual versão do explorador de dados, bem como as suas correspondentes representações gráficas. Pensamos que esta tabela será um instrumento útil para qualquer futuro utilizador do nosso sistema.

Nr	Rep. Gráfica	Descrição
1		Representação de um qualquer objeto de dados – relação, vista ou resultado. O exemplo apresenta um objeto de dados que é uma relação com a designação de "aluno".
2		Representação de uma operação de junção entre dois objetos de dados: "aluno" e "compra".
3		Representação de uma operação de união entre dois objetos de dados.
4		Representação de uma vista de dados. O exemplo apresentado refere-se a uma vista com a designação "VIEW_aluno_compra".

Tabela 1 - Descrição das operações sobre objetos de dados e suas representações gráficas

Capítulo 5

Conclusões e Trabalho Futuro

5.1 Comentários e Conclusões Finais

Quando Edgar Codd publicou o modelo relacional (CODD 1970) colocou as bases de dados num patamar mais próximo do utilizador comum. A organização do modelo que apresentou, assente na definição de relações e de relacionamentos entre os dados através de um simples elemento, a tabela, passou a influenciar fortemente as linguagens para definição, acesso e exploração de bases de dados. Para que fosse possível trabalhar com o modelo de Codd, surgiram logo na década de 70, pelas mãos da própria IBM, várias propostas para linguagens de alto nível, quer em termos textuais quer em termos visuais. Ao apresentar um modelo de organização de dados fácil e acessível e permitir o uso de linguagens de alto nível, o modelo proposto tornou-se a uma referência marcante (mais tarde um padrão) no domínio dos sistemas de bases de dados.

As linguagens de alto nível vieram, assim, derrubar a grande dificuldade que existia no acesso aos dados por parte dos utilizadores. Apesar de terem sido bem-sucedidas, estas não ficaram num estado de estagnação, pelo contrário, apresentaram sempre uma capacidade extraordinária de se reinventarem ao

longo do tempo. Os seus promotores e construtores não descansaram, estiveram (e continuam a estar) constantemente à procura de novas soluções que possam passar as linguagens para um novo patamar de utilização, mais fácil e ao mesmo tempo mais poderoso.

Como vimos, o caminho destas linguagens teve início com o aparecimento da SQUARE (Reisner et al. 1975), uma linguagem baseada na álgebra relacional que consolidou a aplicação prática do modelo proposto por Codd. Porém, a forma como esta linguagem se revelou, baseada em símbolos matemáticos, fez com que não fosse muito bem aceite pela comunidade de utilizadores da altura. Tal situação, levou a IBM a desenvolver e a apresentar a linguagem SEQUEL (Reisner et al. 1975), que mais tarde assumiu a denominação pela qual hoje é sobejamente conhecida, a SQL, que pouco mais tarde alcançou o estatuto de linguagem padrão para sistemas de bases de dados relacionais.

Ao mesmo tempo que as linguagens textuais evoluíam, as linguagens visuais começaram também a ganhar algum espaço entre os utilizadores mais inexperientes. Na base deste tipo de linguagens encontrava-se a linguagem QBE (Zloof 1981), também ela um projeto da IBM. Esta linguagem veio permitir realizar consultas através de exemplos, nos quais o utilizador se limita a indicar que tipos de resultados deseja obter, realizando o sistema o resto do trabalho de exploração. Tal como as linguagens textuais, as linguagens visuais evoluíram muito, originando variadíssimas instâncias para este tipo de linguagem. Apesar de já apresentarem, sensivelmente, quatro décadas de evolução, o conceito por trás de todas elas mantém-se o mesmo. Estas linguagens permitem converter as ações do utilizador nas mais variadas consultas com o suporte de uma linguagem textual. Uma das falhas apontadas a este tipo de linguagem, apesar de todo o tempo investido na sua evolução, prende-se com a falta de interatividade entre o utilizador e os sistemas de dados. Além disso, quando os comandos envolvidos nos processos de exploração de dados são complexos, envolvendo mais do que uma simples seleção, estas linguagens tendem, por norma, a perder a sua simplicidade, dificultando a vida aos seus utilizadores mais inexperientes.

Esta área das TIC não se tem mostrado muito profícua em termos de projetos que envolvam utilizadores cuja faixa etária seja baixa. Para este tipo de utilizadores costuma-se optar pelo desenvolvimento de aplicações específicas, para que assim possa ser introduzido conhecimento de bases de dados, usualmente de difícil aprendizagem, de uma forma simples e pedagógica. Esta aprendizagem tende a provocar uma evolução do pensamento lógico por parte das crianças que se encontram envolvidas nestes projetos. Assim, baseando-nos nas várias iniciativas que um pouco por todo o lado têm vindo a ser promovidas e realizadas sobre a introdução das TIC na educação das crianças em todo mundo (e na falta de uma aplicação específica na área dos sistemas de bases de dados que permita a aprendizagem de uma forma simples), desenvolvemos o sistema de exploração de dados que temos vindo a apresentar. Um "primeiro passo", foi o que achamos que demos até agora.

5.2 A Abordagem Seguida

Nesta altura, gostaríamos de fazer um ponto de situação com base nos vários objetivos traçados inicialmente para este trabalho de dissertação, enunciados no início deste documento, e que aqui os revemos de uma forma mais sintética:

- Projeto e construção de um sistema de exploração para uma base de dados relacional, que fosse intuitivo, de fácil utilização e que não exija conhecimento em SQL, livre de instalações e disponível para qualquer tipo de plataforma computacional disponível atualmente.
- Todos os processos de manipulação de dados a desenvolver no sistema deverá ser suportada por manipuladores gráficos com capacidade de representação dos diferentes objetos de uma base de dados.
- O modelo de dados relacional deverá ser completamente transparente ao utilizador, sendo apenas revelado ao utilizador os objetos de dados (tabelas e vistas) disponíveis na base de dados que ele escolheu para trabalhar através de elementos gráficos, cujos tamanho e cor revelarão as características do objeto de dados em questão.

Todas estas linhas de trabalho foram acompanhadas por uma outra, indispensável para a boa implementação do sistema de exploração: a escolha

de um SGBD de suporte. Seguindo a mesma linha de raciocínio anteriormente exposta, depois de estudarmos um grande número de opções a nossa escolha recaiu sobre o MySQL, um SGBD de código aberto, muito bem documentado – o que facilitou imenso o acesso aos metadados do sistema - e com um número de instalações muito significativa a nível mundial (IT 2012).

Para além disto, decidimos construir o sistema de exploração com base em tecnologia Web, cujas ferramentas já foram apresentadas e justificadas anteriormente. Tal circunstância levantou o problema da perda de alguns dos movimentos característicos que os sistemas operativos móveis permitem executar sobre um ecrã. Contudo depois de alguma pesquisa e estudo esta preocupação foi sanada, tendo-se recorrido a uma biblioteca desenvolvida em JavaScript (Major 2011). Todos os movimentos conseguidos a partir da biblioteca referida disponibilizaram ao sistema de exploração de dados um nível de interatividade muito interessante entre o utilizador e o sistema de gestão de bases de dados, bem como retiraram a complexidade de alguns dos comandos disponíveis como os que suportam as operações de união e de junção.

Mas não foram só os comandos das operações de união e de junção a beneficiarem da interatividade conseguida graças à utilização de JavaScript. Na sua grande maioria, todos os comandos do sistema de exploração são acionados através de um conjunto de simples toques no ecrã de qualquer dispositivo móvel, o que os torna simples de utilizar e, como tal, ao alcance de qualquer utilizador. Todavia, apesar de se ter conseguido melhorar a interatividade geral do sistema de exploração, bem como retirar parte da complexidade da sua utilização, face a outras ferramentas da área, outros aspetos mais precisariam de ser melhorados, uma vez que não foram desenvolvidos da forma como ambicionávamos. Veja-se, por exemplo, os mecanismos de filtragem desenvolvidos no sistema, em particular aqueles que atuam sobre os registos de um dado elemento de dados, que são ainda um pouco complexos, requerendo a curto prazo uma revisão cuidada. Outra característica interessante da versão atual do sistema é a possibilidade do utilizador poder desenvolver um gráfico a partir dos dados de um elemento de

dados disponível no painel de exploração. Algo que foi pensado e implementado para demonstrar, simplesmente, a capacidade de análise gráfica que um sistema destes pode ter. A implementação desta característica revelou-se de grande importância, uma vez que permitiu revelar que através de uma utilização simples, sem grande complexidade e direcionada para utilizadores pouco experiência, podemos ir um pouco mais para além do colhimento de simples processos de consulta, podendo ser expandido para outras áreas funcionais no domínio dos sistemas de bases de dados.

Na atual versão, o sistema de exploração desenvolvido não é mais do que uma "prova de conceito", um pequeno sistema piloto, que utiliza várias das tecnologias que têm vindo a ganhar grande espaço no mercado das TIC e que hoje se encontram já ao dispor de qualquer pessoa. Todavia, o sistema de exploração de dados desenvolvido, na nossa opinião, revela um grande potencial de desenvolvimento, para que numa próxima versão possa já ser, de facto, uma ferramenta para utilização prática real em processos de exploração de dados, com especial orientação para uma comunidade de jovens utilizadores das novas plataformas computacionais.

5.3 Trabalho Futuro

O sistema de exploração de dados desenvolvido não deve ser visto como um produto final, mas sim como um ponto de partida para algo um pouco mais ambicioso e ponderado, tal como foi pensado aquando da definição dos trabalhos desta dissertação. O sistema de exploração foi desenvolvido para utilizadores de faixas etárias muito baixas (a partir dos 7 anos) e como tal deverá ser testado, especificamente, com esse tipo de utilizadores a curto prazo. Basicamente, pretende-se ensaiar o sistema com uma comunidade de utilizadores específica - uma turma de alunos do ensino básico -, com uma carteira de problemas bem específica, pensada de acordo com as necessidades de tais utilizadores, sendo o seu feedback recolhido e estudado para avaliação prática real do sistema agora desenvolvido. De facto, precisa-se que esta prova de conceito, digamos assim, seja realmente demonstrada e a sua viabilidade comprovada. Complementarmente, dever-se-ia desenvolver um programa de

cooperação com elementos profissionais ligados à área de educação, como por exemplo o Instituto de Educação da Universidade do Minho, para ajustarmos o modelo de funcionamento às práticas pedagógicas correntes. Os elementos ligados à área da educação poderiam avaliar se o "modelo pedagógico" do sistema se encontra adequado ou não. Além disso, estes poderiam ser fundamentais no desenvolvimento de um conjunto de estímulos a serem desenvolvidos em conjunto, para melhorar a capacidade de atração e utilidade desta ferramenta.

Fora da área pedagógica, também ainda há muito trabalho a desenvolver, especialmente em termos de comandos de exploração de dados. Por exemplo, é necessário tratar da implementação de novos comandos que façam a ordenação dos registos de um dado elemento de dados ou dos resultados obtidos através de uma operação de combinação de dados. Com este tipo de comando seria possível ordenar de forma crescente e decrescente os dados pretendidos, utilizando, por exemplo, ações como *swipe rigth* ou *swipe left*, ficando o movimento à direita associado com a ordenação crescente e o movimento à esquerda com a ordenação decrescente. Mais tarde, também poderemos acrescentar novos comandos de junção, como o *rigth join* e o *left join*. Estes últimos comandos provocariam, naturalmente, a reformulação do comando de junção já implementado. Além disso, o algoritmo usado na construção das *queries* do comando *union* precisa de ser trabalhado no sentido de melhorar o seu desempenho. Este algoritmo é o menos eficiente de todos os que foram desenvolvidos no âmbito do sistema de exploração. Também, os mecanismos de geração de gráficos poderiam ser melhorados usando uma biblioteca em JavaScript mais poderosa como, por exemplo, a *flotcharts*. Esta opção permitirá a construção de gráficos mais elegantes e mais interativos. Por último, pretende-se que este sistema de exploração de dados possa interagir com outros SGBD, quer estes sejam ou não relacionais. Esperamos que a curto prazo possamos dar, então, o "segundo passo" no desenvolvimento do nosso sistema de exploração de dados.

Bibliografia

Benzaken, V. et al., 2008. Pattern by example. In *Proceedings of the 10th international ACM SIGPLAN symposium on Principles and practice of declarative programming - PPDP '08*. New York, New York, USA: ACM Press, p. 131. Available at: http://dl.acm.org/citation.cfm?id=1389449.1389466 [Accessed January 22, 2014].

Boyle, J., Leishman, S. & Gray, P.M.., 1996. From WIMPS to 3D: The Development of AMAZE. *Journal of Visual Languages & Computing*, 7(3), pp.291–319. Available at: http://linkinghub.elsevier.com/retrieve/pii/S1045926X96900166 [Accessed January 22, 2014].

C. Batini, T.C., 1995. Visual Query Systems: A Taxonomy.

CATARCI, T. et al., 1997. Visual Query Systems for Databases: A Survey. *Journal of Visual Languages & Computing*, 8(2), pp.215–260. Available at: http://linkinghub.elsevier.com/retrieve/pii/S1045926X97900379 [Accessed January 22, 2014].

Chamberlin, D.D. et al., 1981. A history and evaluation of System R. *Communications of the ACM*, 24(10), pp.632–646. Available at: http://portal.acm.org/citation.cfm?doid=358769.358784 [Accessed January 22, 2014].

Chamberlin, D.D. & Boyce, R.F., 1974. SEQUEL: A STRUCIURED ENGLISH QUERY LANGUAGE. *Proc. ACM SIGFIDET Conference*. Available at: http://researcher.ibm.com/researcher/files/us-dchamber/sequel-1974.pdf [Accessed December 18, 2013].

CODD, E.F., 1970. A Relational Model of Data for Large Shared Data Banks. *IBM Research Laboratory, San Jose, California*. Available at: http://www.seas.upenn.edu/~zives/03f/cis550/codd.pdf.

Code.org, 2012. Anybody can learn | Code.org. Available at: http://code.org/ [Accessed December 18, 2013].

Connolly, T.M. & Begg, C.E., 2004. *Database Systems: A Practical Approach to Design, Implementation and Management (4th Edition)*, Addison Wesley. Available at: http://www.amazon.com/Database-Systems-Practical-Implementation-Management/dp/0321210255 [Accessed January 22, 2014].

Cruz, I.F., 1992. DOODLE: A Visual Language for Object-Oriented Databases. In pp. 71 – 80.

Cumming, G., 2000. *Advanced Research in Computers and Communications in Education: New Human Abilities for the Networked Society - Proceedings of ICCE '99, 7th ... in Artificial Intelligence and Applications)*, IOS Press,US. Available at: http://www.amazon.co.uk/Advanced-Research-Computers-Communications-Education/dp/1586030272 [Accessed January 22, 2014].

Dennebouy, Y. et al., 1995. SUPER: Visual Interfaces for Object + Relationship Data Models. *Journal of Visual Languages & Computing*, 6(1), pp.73–99. Available at: http://linkinghub.elsevier.com/retrieve/pii/S1045926X85710051 [Accessed January 22, 2014].

Editora, P., Pesquisa global - Infopédia. Available at: http://www.infopedia.pt/pesquisa.jsp?qsFiltro=0&qsExpr=junção.

Elmasri, R. & Navathe, S., 2010. *Fundamentals of Database Systems*, Addison Wesley. Available at: http://www.amazon.co.uk/Fundamentals-Database-Systems-Ramez-Elmasri/dp/0136086209 [Accessed January 22, 2014].

Fujii, H. & Korfhage, R.R., 1991. Features and a model for icon morphological transformation. In *Proceedings 1991 IEEE Workshop on Visual Languages*. IEEE Comput. Soc. Press, pp. 240–245. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=238826 [Accessed January 22, 2014].

Hyoung-Joo Kim, H.F.K., 1988. PICASSO: a Graphical Query Language. *Softw., Pract. Exper.*, 18, pp.169 – 203.

IT, S., 2012. DB-Engines Ranking - popularity ranking of database management systems. Available at: http://db-engines.com/en/ranking.

Jcarnelian, 2005. A Case for Formal Specification || kuro5hin.org. Available at: http://www.kuro5hin.org/story/2005/7/29/04553/9714 [Accessed January 23, 2014].

Labs, S., 2013. Primo • A playful tangible programming interface. Available at: http://primo.io/.

Major, B., 2011. jQuery Mobile "Events" | Code Monkey. Available at: http://ben-major.co.uk/2011/11/jquery-mobile-events/.

Massari, A., Pavani, S. & Saladini, L., 1994. QBI. In *Proceedings of the workshop on Advanced visual interfaces - AVI '94*. New York, New York, USA: ACM Press, pp. 240–242. Available at: http://dl.acm.org/citation.cfm?id=192309.192360 [Accessed January 22, 2014].

Matthias Jarke, Y.V., 1985. A Framework for Choosing a Database Query Language. *ACM Comput. Surv.*, 17, pp.313 – 340.

Nancy H. McDonald, M.S., 1975. CUPID - The Friendly Query Language. In pp. 127 – 131.

Notare, M.R., 2003. 10.2 Propriedade das Operações Binárias for Matemática Discreta-02. *Universidade de Caxias do Sul*. Available at: http://pt.scribd.com/doc/50563805/56/Propriedade-das-Operacoes-Binarias.

Olle, T.W., 2006. *History of Computing and Education 2 (HCE2)* J. Impagliazzo, ed., Springer US. Available at: http://www.springerlink.com/index/10.1007/978-0-387-34741-7 [Accessed January 22, 2014].

Play-i, I., 2013. Delightful robots for children to program | Play-i. Available at: https://www.play-i.com/.

Reisner, P., Boyce, R.F. & Chamberlin, D.D., 1975. Human factors evaluation of two data base query languages. In *Proceedings of the May 19-22, 1975, national computer conference and exposition on - AFIPS '75*. New York, New York, USA: ACM Press, p. 447. Available at: http://dl.acm.org/citation.cfm?id=1499949.1500036 [Accessed January 22, 2014].

Shapiro, D., 2013. Robot Turtles | The Board Game for Little Programmers. Available at: http://www.robotturtles.com/.

Trémeaux, J.-M., 2011. Libchart - Simple PHP chart drawing library. Available at: http://naku.dohcrew.com/libchart/pages/introduction/.

Weisstein, E.W., Binary Operation -- from Wolfram MathWorld. Available at: http://mathworld.wolfram.com/BinaryOperation.html [Accessed January 23, 2014].

Zloof, M.M., 1981. QBE/OBE: A Language for Office and Business Automation. *IEEE Computer*, 14, pp.13 – 22.