Pós-Graduação em Ciência da Computação

“Uma Abordagem Baseada em Contexto para Reescrita de Consultas a Bancos de Dados Relacionais”

Por

Paulo Roberto Moreira Maciel

Tese de Doutorado

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

Recife, 2015

UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

PAULO ROBERTO MOREIRA MACIEL

"UMA ABORDAGEM BASEADA EM CONTEXTO PARA REESCRITA DE CONSULTAS A BANCOS DE DADOS RELACIONAIS"

ESTE TRABALHO FOI APRESENTADO AO PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO DO CENTRO DE INFORMÁTICA DA UNIVERSIDADE FEDERAL DE PERNAMBUCO, COMO REQUISITO PARCIAL PARA OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIA DA COMPUTAÇÃO.

ORIENTADORA: DR. ANA CAROLINA SALGADO

CO-ORIENTADORA: DRA. PATRÍCIA AZEVEDO TEDESCO

Recife, 2015

Catalogação na fonte
Bibliotecária Jane Souto Maior, CRB4-571

M152a Maciel, Paulo Roberto Moreira
Uma abordagem baseada em contexto para reescrita de consultas a bancos de dados relacionais / Paulo Roberto Moreira Maciel. – Recife: O Autor, 2015.
200 p.: il., fig., tab., quadros.

Orientador: Ana Carolina Salgado.
Tese (Doutorado) – Universidade Federal de Pernambuco. CIn, Ciência da computação, 2015.
Inclui referências e apêndices.

1. Banco de dados. 2. Computação sensível a contexto. I. Salgado, Ana Carolina (orientadora). II. Título.

005. 74 CDD (23. ed.) UFPE- MEI 2015-63

Tese de Doutorado apresentada por **Paulo Roberto Moreira Maciel** à Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, sob o título **"Uma Abordagem Baseada em Contexto para Reescrita de Consultas a Bancos de Dados Relacionais"** orientada pela **Profa. Ana Carolina Brandão Salgado** e aprovada pela Banca Examinadora formada pelos professores:

_____

Prof. Fernando da Fonseca de Souza
Centro de Informática / UFPE

_____

Prof. Robson do Nascimento Fidalgo
Centro de Informática / UFPE

_____

Prof. Altigran Soares da Silva
Instituto de Computação / UFAM

_____

Profa. Maria da Conceição Moraes Batista
Departamento de Estatística e Informática / UFRPE

_____

Prof. Javam de Castro Machado
Departamento de Computação / UFC

Visto e permitida a impressão.
Recife, 27 de fevereiro de 2015.

_____

Profa. Edna Natividade da Silva Barros
Coordenadora da Pós-Graduação em Ciência da Computação do
Centro de Informática da Universidade Federal de Pernambuco.

À minha família, bem maior da minha vida.

AGRADECIMENTOS

Agradeço a Deus, por minha saúde e por ter permitido que eu concluísse mais este trabalho.

Agradeço à minha esposa Luciana por todo amor, compreensão e apoio.

Agradeço à minha família, por me dar o suporte necessário para que pudesse chegar onde estou.

Agradeço também ao amigo e companheiro de pesquisa Antônio Mendonça, que me ajudou durante esta caminhada.

Agradeço às minhas amigas e orientadoras, Ana Carolina e Patrícia, por todo o auxílio, paciência e confiança, indispensáveis, durante este trajeto.

Agradeço aos professores Ivon Fittipaldi e Amaro Lins, coordenadores da Representação do MCTI no Nordeste, pelos incentivos pessoais e institucionais para a realização deste doutoramento.

De forma geral meu muitíssimo obrigado a todos que conviveram comigo nesses últimos anos pela paciência, orações e incentivo.

RESUMO

Com o desenvolvimento das tecnologias para Internet e sua utilização em larga escala, as consultas a bancos de dados não são mais realizadas como anos atrás, quando os usuários tinham um perfil conhecido e realizavam suas consultas em computadores de mesa, por meio de aplicações dedicadas. Atualmente, com a descentralização e o aumento massivo da distribuição de informação, há uma grande variedade de fatores presentes durante uma consulta, que podem influenciar na adequação e relevância das respostas fornecidas ao usuário, tais como: preferências pessoais, localização (e.g. casa, trabalho, hotel, avião), clima (e.g. chuvoso, ensolarado) ou dispositivo usado. Esses fatores usualmente não são levados em consideração, quando da realização de consultas a bancos de dados.

O conjunto de condições e influências relevantes que tornam uma situação única e compreensível é entendido como contexto. Elementos contextuais são definidos como qualquer dado, informação ou conhecimento que permite caracterizar uma entidade em um domínio. O conjunto dos elementos contextuais instanciados, necessários para apoiar uma tarefa em um dado momento, denota o contexto.

Embora adotem tecnologia madura e consagrada, os Sistemas Gerenciadores de Bancos de Dados relacionais não possuem os recursos necessários para responder consultas considerando o contexto. Este trabalho propõe uma abordagem denominada Texere, para que consultas a bancos de dados relacionais sejam dotadas de sensibilidade a contexto. A hipótese concebida para a solução do problema baseia-se na análise dos elementos contextuais associados a uma consulta (oriundos da aplicação, do dispositivo, do usuário e dos ambientes físico e computacional) e na utilização de diretivas de reescrita, capazes de alterar consultas convencionais, para que retornem respostas mais adequadas ao contexto sob as quais foram realizadas e mais relevantes para o usuário.

Para avaliação do trabalho, experimentos foram realizados com um protótipo. A análise dos resultados produzidos pelo julgamento de usuários evidencia a viabilidade da abordagem e ganhos na adequação ao contexto e relevância das respostas produzidas pelas consultas reescritas.

Palavras-chave: Sensibilidade a contexto. Preferências. Sistemas Gerenciadores de Bancos de Dados.

ABSTRACT

With the development of Internet technologies and their large scale usage, database queries have evolved from situations where users had a known profile and submitted their queries on desktop computers through dedicated applications to situations with decentralized and massive information distribution. However, there are a variety of factors which may influence the appropriateness and relevance of the answers provided to users' queries, such as: personal preferences, location (e.g. home, work, hotel, plane), climate (e.g. rainy, sunny), and device used. These factors are not usually taken into consideration when submitting queries to databases.

This set of relevant conditions and influences that make a situation unique and understandable is called context. Contextual elements are defined as any data, information or knowledge that allows us to characterize an entity in a domain. The set of instantiated contextual elements needed to support a task at a given time denotes the context.

Although database management systems (DBMS) are a mature, widely known technology and are considered to be the most efficient systems to manage and respond to structured data queries, they currently lack the resources needed to meet the demand for information that is adapted to the current context. This work proposes an approach named Texere, that endows conventional relational database queries with context sensitivity. The solution is based on the analysis of contextual elements associated with a query (arising from the application, device, user and physical and computational environments) and the use of directives for query rewriting able to alter queries devoid of context awareness to others whose answers are more appropriate to the context in which they were made and thus are more relevant to the user.

For the evaluation of this work, experiments were performed with a prototype. The results produced by the users' evaluation points to the feasibility of the approach and gains in adaptation to the context and relevance of the answers produced by the rewritten queries.

Keywords: Context Sensitivity. Preferences. Database Management Systems.

LISTA DE FIGURAS

LISTA DE QUADROS E TABELAS

LISTA DE SÍMBOLOS

:=	Atribuição
||	Concatenação
[ ]	Opcionalidade
E	Entidade
E.a	Atributo *a* pertencente à entidade *E*
Q	Uma consulta SQL
Q`	Uma consulta SQL reescrita
R	Relação
α	Uma lista de atributos
π	Projeção
σ	Seleção
ϕ	Condição
τ	Ordenação
γ	Agregação
$\bowtie$	Junção
<oc>	Operador condicional
	Operador lógico
<vl>	Valor instanciado

LISTA DE ABREVIATURAS E SIGLAS

AD	Administrador de Dados
BD	Banco de Dados
BDEC	Base de Dados de Elementos Contextuais
DRC	Diretiva de Reescrita de Consulta
EC	Elemento Contextual
ED	Especialista no Domínio
SGBD	Sistema Gerenciador de Banco de Dados
SSC	Sistema Sensível ao Contexto

SUMÁRIO

1. INTRODUÇÃO

Com o desenvolvimento das tecnologias para Internet e sua utilização em larga escala, as aplicações computacionais têm que lidar cada vez mais com uma quantidade crescente de informação, disponível em uma grande variedade de dispositivos (e.g. *tablet, smartphone, notebook, Google Glass*[1]). Por sua vez, esses dispositivos levam informação para um número cada vez maior de usuários, os quais possuem interesses e realidades diferentes.

Ao mesmo tempo, há uma grande variedade de fatores presentes durante a especificação de uma consulta por informação que podem influenciar na adequação e relevância das respostas fornecidas ao usuário, tais como: preferências pessoais do usuário, localização (e.g. casa, trabalho, hotel, avião), e clima (e.g. chuvoso, ensolarado). Esses fatores usualmente não são levados em consideração quando do processamento de consultas a bancos de dados. O enorme volume de dados disponíveis em serviços para Internet e possibilidades de distribuição em várias fontes de dados também aumenta a possibilidade de respostas em excesso e pouco relevantes para o usuário.

1.1. Motivação

O conjunto de condições e influências relevantes que tornam uma situação única e compreensível é entendido como contexto (BRÉZILLON, 1999). Em outras palavras, contexto é o conhecimento que está por trás da habilidade de discriminar o que é ou não importante em um dado momento, auxiliando indivíduos a melhorar a qualidade da interação e a compreender situações, ações ou eventos (VIEIRA et al., 2009).

Para ilustrar a questão de adequação das respostas a consultas, imagine um usuário que realiza uma mesma consulta sobre os pontos turísticos da cidade do Recife em duas ocasiões distintas: uma, estando fora do Brasil, utilizando um computador de mesa, e outra, durante uma visita à referida cidade, utilizando um *smartphone*. Para a primeira ocasião, uma lista dos locais turísticos que se baseiem

[1] Google Glass, www.google.com/glass. Acesso em: 15 jun. 2014.

no perfil de preferências do usuário é esperada como adequada à consulta (e.g. o usuário prefere pontos de cunho cultural). Já para a segunda consulta, a adequação da consulta e consequente relevância das respostas estão sujeitas a outras variáveis e situações distintas. Por exemplo, a resposta pode ser considerada mais relevante se trouxer apenas locais apropriados para visitação sob as condições climáticas correntes, e se também estiver ordenada pela proximidade do ponto turístico em relação à localização do usuário.

É nesse cenário que se encontra a motivação deste trabalho, a Abordagem Texere, que propõe alternativa para que consultas realizadas a bancos de dados possam considerar o contexto sob o qual foram realizadas e assim obter respostas mais relevantes para seus usuários. O nome Texere vem de parte da palavra latina para contexto, *contexere,* que significa *con*, junto e *texere,* tecer. A abordagem Texere, tal qual seu significado, busca “tecer dados” para dar significado a uma situação, da mesma forma que as pessoas tecem, entrelaçam suas experiências e informações com tudo que está ao seu redor, para interpretar ou dar significado para algo.

1.2. Definição do Problema

Desde os anos 2000, o estudo do Contexto tem sido impulsionado por áreas da computação que necessitam de sistemas computacionais sensíveis ao contexto, tais como computação ubíqua, computação móvel, computação pervasiva e inteligência para ambientes (BUNNINGEN et al., 2005; FENG et al., 2004). A evolução dos sistemas computacionais tradicionais para sistemas que recebem informações de sensores dispostos nos mais diversos ambientes segue de forma bastante rápida (PERERA et al., 2013). Em Ciência da Computação, um sistema é denominado Sistema Sensível ao Contexto (SSC) se utiliza contexto para fornecer informações ou serviços relevantes para o usuário (DEY; ABOWD, 2001; VIEIRA et al., 2011).

Embora adotem tecnologia madura e consagrada e sejam considerados a forma mais eficiente de gerenciar e responder consultas a dados estruturados, os Sistemas Gerenciadores de Bancos de Dados (SGBD) tradicionais (Relacional ou Objeto-relacional) utilizam modelos de dados do negócio específicos para as suas aplicações clientes com o objetivo de fornecer a melhor resposta para cada consulta possível pré-formatada e não possuem recursos necessários para atender à

demanda por informação considerando o contexto. Esse fato faz com que o gerenciamento do dado sensível ao contexto seja delegado às aplicações.

Assumir as tarefas de gerenciamento dos dados sensíveis ao contexto aumenta a complexidade dos SSC, exigindo soluções de projeto que especifiquem como o contexto deva ser modelado em um banco de dados convencional. O SSC deve, além de controlar o seu próprio comportamento em função do contexto, assumir parte das funções típicas do SGBD ao realizar o gerenciamento e a apresentação dos dados contextuais.

Essa conjuntura tem estimulado a comunidade de bancos de dados a criar alternativas para suprir a demanda por novos paradigmas para a modelagem e gerenciamento de dados contextuais em SGBD (BOLCHINI et al., 2009; ROUSSOS; SELLIS, 2008; THEODORAKIS et al., 2002). Ainda em estágios iniciais de pesquisa, percebemos que usar dados contextuais gerenciados no SGBD requer novos modelos de banco de dados que contemplem o contexto desde a fase de projeto do esquema de dados, mecanismos de consulta mais expressivos (e.g. que deem suporte a preferências do usuário) e métodos de acesso a dados especialmente concebidos para melhorar a eficiência das operações de leitura em função do contexto identificado.

Ao contrário de trabalhos correlatos na área de pesquisa, este trabalho busca uma solução aplicável às tecnologias atuais de SGBD, que possa ser integrada ao enorme parque de bancos de dados relacionais existentes.

1.3. Objetivo

Diante do panorama apresentado, o objetivo deste trabalho é apresentar a abordagem Texere, que tem como objetivo possibilitar que SGBD relacionais convencionais possam responder a consultas de forma sensível ao contexto.

Para atingir o objetivo geral, foram traçados os seguintes objetivos específicos:

- Investigar como o contexto pode ser usado para tornar respostas a consultas a bancos de dados mais relevantes para seus usuários;
- Especificar um modelo de contexto aplicável a bancos de dados;
- Especificar uma abordagem que forneça aos SGBD relacionais convencionais sensibilidade a contexto;

- Especificar uma estratégia para identificação do contexto associado a consultas a bancos de dados;
- Definir um conjunto de diretivas para reescrita de consultas de forma sensível ao contexto sob as quais foram realizadas; e
- Implementar protótipo e realizar experimentos para avaliação da abordagem proposta.

1.4. Abordagem Texere

A Abordagem Texere, apresentada neste trabalho, baseia-se no uso de uma camada mediadora entre a aplicação e o SGBD e visa aproveitar as tecnologias atuais de banco de dados, que são maduras e bastante difundidas no mercado. Essa forma separa o gerenciamento dos dados contextuais do processamento da inferência do contexto.

A lógica da interpretação do contexto e a semântica dos dados contextuais localizam-se na camada mediadora enquanto que os dados contextuais são armazenados em um banco de dados relacional. Isso permite vantagens em poupar tempo e esforço computacional para a camada da aplicação e concentrar de forma estruturada os dados contextuais coletados em um local específico; busca evitar que dispositivos leves (por exemplo, *smartphone*s) tenham que lidar com grande quantidade de dados; e propicia tratar os dados contextuais de maneira uniforme, independentemente do dispositivo e da aplicação utilizada.

Os seguintes aspectos foram considerados na concepção do trabalho: utilização de um modelo de representação do contexto; especificação da técnica a ser utilizada para inferência do contexto e especificação do método de consulta a SGBD relacional, considerando o contexto corrente e preferências do usuário.

A Abordagem Texere, configura-se composta pelos seguintes elementos:

a) *Mapeamento do contexto* - um processo para identificação de elementos contextuais e geração de um repositório de metadados, baseado em um modelo de contexto. Os elementos contextuais são dados que permitem a caracterização de uma entidade em um domínio e podem ser encontrados na própria fonte de dados da aplicação como também no ambiente externo.

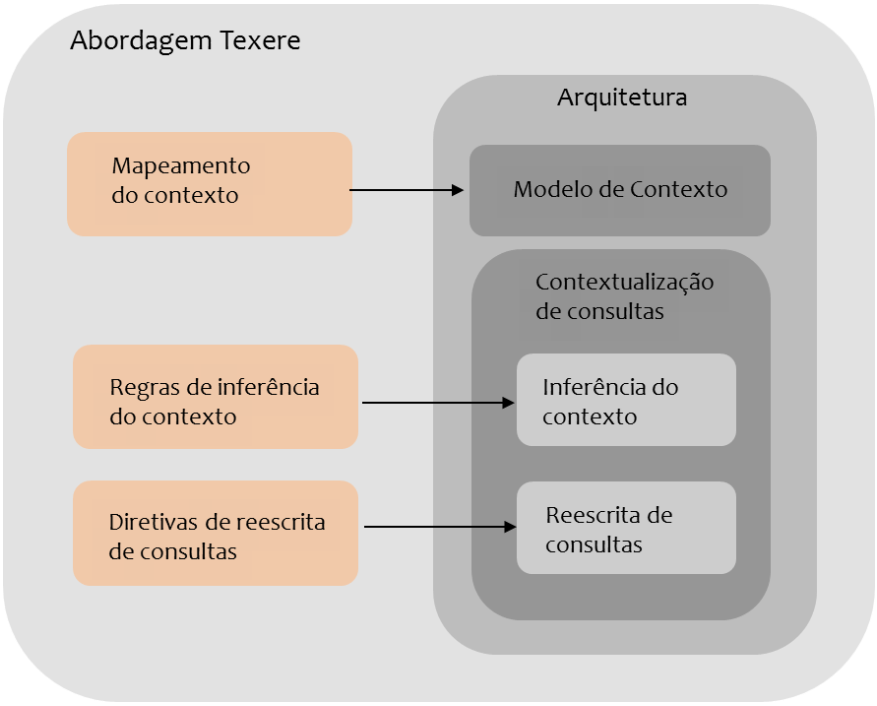
b) *Regras de inferência do contexto* - um método para definição de regras de inferência do contexto que informam como os dados devem ser selecionados em função do contexto inferido. Essas regras podem ser definidas tanto por usuários especialistas no domínio da aplicação como pelos usuários finais da aplicação. Elas permitem a realização de inferência do contexto de forma dinâmica e podem ser processadas por um motor de inferência.

c) *Diretivas de reescrita de consultas* - a especificação de um conjunto de diretivas de reescrita de consultas a bancos de dados relacionais. As diretivas informam como reescrever consultas, alterando-as para considerar o contexto sob a qual foram realizadas.

d) *A arquitetura* de *software* capaz de materializar a implementação da Abordagem Texere, baseada no modelo de contexto e em um módulo de contextualização de consultas.

A estrutura conceitual da Abordagem Texere está apresentada na Figura 1.1. O gerenciamento do dado contextual utiliza um modelo de representação do contexto, regras para inferência do contexto e reescrita de consultas baseadas em diretivas traduzidas para a linguagem SQL padrão ANSI (SQL ISO 1992, 2003, 2008).

Figura 1.1 – Estrutura conceitual da Abordagem Texere



Fonte: O Autor.

Em uma visão geral do funcionamento da abordagem, consultas convencionais emitidas para o banco de dados de uma aplicação são interceptadas e reescritas para considerarem contexto sob o qual foram emitidas. Por contexto de uma consulta foi considerado o contexto denotado pelo conjunto de elementos contextuais da aplicação, do usuário, do dispositivo e dos ambientes físico e computacional associados. Elementos contextuais pertencentes ao banco de dados da aplicação são ditos elementos contextuais internos; os encontrados em fonte externa ao banco de dados da aplicação são chamados externos.

Para possibilitar a consulta sensível ao contexto, os elementos contextuais relevantes para o domínio da aplicação (que foram mapeados previamente para uma base de metadados da arquitetura Texere) têm seus valores instanciados verificados e, baseado nos valores encontrados, regras para inferência do contexto são disparadas e diretivas de reescrita executadas.

As diretivas de reescrita alteram a consulta original para adicionar, suprimir, filtrar, ordenar ou classificar informações a serem recuperadas, adequando-as tanto ao contexto inferido da consulta quanto às preferências do usuário.

De uma forma geral, a maior contribuição deste trabalho é propor uma alternativa que propicie uma forma de representação do dado contextual no banco de dados e confira sensibilidade a contexto para consultas aos SGBD relacionais convencionais largamente adotados.

1.5. Hipóteses da Tese

As hipóteses para esta tese são:

H1. É possível utilizar técnicas de reescrita como forma de conferir sensibilidade a contexto em consultas a bancos de dados relacionais.

H2. Consultas que considerem o contexto sob o qual foram realizadas tendem a propiciar respostas mais relevantes para seus usuários.

H3. Consultas reescritas com utilização de preferências independentes e dependentes do contexto tendem a produzir resultados mais relevantes para o usuário, do que se forem utilizadas apenas preferências independentes do contexto.

Foram consideradas respostas relevantes, aquelas que estão de acordo com o contexto da consulta e atendem aos critérios de preferências do usuário. Para comprovar essas hipóteses foram realizados experimentos utilizando um protótipo da arquitetura Texere.

1.6. Estrutura do Documento

Este documento está organizado em cinco outros capítulos, dispostos como descrito a seguir:

O Capítulo 2 apresenta a fundamentação teórica sobre Contexto Computacional, Sistemas Sensíveis ao Contexto e aplicação de contexto para SGBD, que formam os principais conceitos utilizados neste trabalho. Também são apresentadas definições, características e estratégias de processamento do dado contextual e técnicas de reescrita de consultas. O Capítulo 3 apresenta um levantamento do estado da arte de propostas de sistemas de bancos de dados orientados ao contexto, correlatos ao tema deste trabalho.

A abordagem Texere e seus componentes encontram-se descritos no Capítulo 4. Nele, também é apresentada a Arquitetura Texere, as tecnologias utilizadas e exemplos de reescrita de consultas. No Capítulo 5 são descritos a implementação e funcionamento do protótipo da Arquitetura Texere em um sistema de teste e os experimentos realizados para validação das hipóteses do trabalho. Por fim, no Capítulo 6, são apresentadas as conclusões e contribuições específicas desta Tese, bem como sugestões de trabalhos futuros.

2. FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são abordados os conceitos de Contexto Computacional e Sistemas Sensíveis ao Contexto, os principais fundamentos teóricos utilizados neste trabalho. Além disso, é discutida a relação entre os Sistemas Gerenciadores de Bancos de Dados e o dado contextual, e descritos os conceitos que fundamentam as técnicas de reescrita de consultas.

2.1. Contexto Computacional

Contexto tem sido objeto de pesquisa, por muitos anos, em comunidades científicas, tais como Linguística (CLARK; CARLSON, 1981; LEECH; THOMAS, 1990) e Psicologia Cognitiva (PERKINS; SALOMON, 1989). Em Ciência da Computação, a noção de contexto foi adotada desde a década de 1980, sendo a Inteligência Artificial considerada a primeira grande área a estudar e procurar formalizar o conceito (AKMAN; SURAV, 1996; BRÉZILLON; POMEROL, 1999; McCARTHY, 1987, 1993; SCHILIT et al., 1994).

Para exemplificar, considere a situação na qual uma pessoa está decidindo o local onde deve almoçar com um colega. Uma série de fatores pode influenciar a decisão: o restaurante está aberto? Está chovendo? É um restaurante vegetariano? Servem massas? É um restaurante caro? É perto de onde estou? As respostas para essas perguntas são fatores que, com maior ou menor relevância, vão influenciar na decisão a ser tomada.

De maneira mais ampla, Contexto é entendido como um conjunto de variáveis que possam ser de interesse de um agente e influenciar suas ações em uma situação particular. Várias definições podem ser encontradas na literatura, mas a mais referenciada é aquela apresentada por Dey (2000):

> Contexto é qualquer informação que pode ser usada para caracterizar uma situação de uma entidade. Uma entidade é uma pessoa, um lugar, ou um objeto que é considerado relevante para a interação entre um usuário e uma aplicação, incluindo o próprio usuário e a própria aplicação (Dey 2000, tradução nossa).

Na visão de Brézillon (1999), o contexto sempre está associado ao foco de atenção do usuário, que pode ser um passo na execução de uma tarefa ou na tomada de uma decisão. O foco determina quais elementos devem ser instanciados e usados para compor o contexto. Por exemplo, para um assistente virtual de trânsito, ao executar a tarefa de selecionar melhor rota, os elementos relevantes a serem considerados são o horário, a condição física das ruas e o clima, dentre outros.

Também é importante destacar o caráter interativo do contexto, em que recursos reconfiguráveis, migratórios, distribuídos e mensurados em escalas diferentes interagem com um ambiente em constante mudança (COUTAZ et al., 2005).

Especificamente na Ciência da Computação, o estudo do contexto é importante na medida em que permite que sistemas e aplicações computacionais ofereçam serviços e informações que sejam mais relevantes aos seus usuários e adequados à suas necessidades ou à situação em que estes se encontrem. Outros benefícios são a contribuição para diminuição de ambiguidades e conflitos, bem como a melhoria da comunicação entre usuário e aplicação, o que ajuda a tornar as aplicações mais flexíveis, amigáveis e fáceis de utilizar (VIEIRA et al., 2009).

Destacam-se dentre as áreas da computação que buscam estudar a utilização, estruturar e formalizar o conceito de contexto: Inteligência Artificial, Computação Ubíqua, Inteligência Ambiental, Sistemas Colaborativos, Hipermídia Adaptativa e Interação Humano-Computador.

Os Elementos Contextuais

Vieira (2008) aprofundou o debate sobre os elementos usados para compor o contexto e procurou explicitar a diferença entre esses dois conceitos básicos: Contexto e Elemento Contextual (EC):

> Um **elemento contextual** é qualquer dado, informação ou conhecimento que permite caracterizar uma entidade em um domínio; e
>
> O **contexto** da interação entre um agente e uma aplicação, para executar alguma tarefa, é o conjunto de elementos contextuais instanciados que são necessários para apoiar a tarefa atual (VIEIRA 2008, tradução nossa).

Os elementos contextuais caracterizam situações, que podem ser analisadas segundo seis dimensões (MORSE et al., 2000; TRUONG et al., 2001), representadas por perguntas e denominadas 5W+1H, que correspondem às suas iniciais em inglês:

- *Who* (quem) - caracteriza informações de identificação da entidade envolvida na análise, tais como nome, e-mail ou impressão digital;
- *What* (o quê) - identifica a atividade que está sendo realizada. São exemplos: uma consulta a um banco de dados, uma viagem;
- *Where* (onde) - caracteriza a localização da entidade envolvida, por exemplo, país, cidade, latitude e longitude;
- *When* (quando) - indica informações temporais, a exemplo de datas, estações, durações e períodos envolvidos na realização de uma atividade;
- *Why* (por que) - demonstra as intenções e motivações da entidade para realizar a atividade envolvida. De difícil identificação, essas motivações geralmente são inferidas por meio da combinação de outros atributos e/ou tarefas. Por exemplo, a realização de repetidas buscas (em uma ferramenta de buscas na Internet) por um mesmo gênero de livro, pode indicar uma preferência ou gosto; e
- *How* (como) - caracteriza a forma ou o meio como os elementos contextuais são capturados. Exemplos disso são os dispositivos e sensores que identificam localização ou temperatura.

Korpipää et al. (2003) definiram critérios para guiar a seleção de elementos contextuais ao projetar uma aplicação sensível ao contexto:

- Habilidade para descrever propriedades úteis, do mundo real;
- Habilidade para inferência de contextos complexos; e
- Facilidade ou viabilidade de ser medido ou reconhecido, automaticamente, de forma precisa e não ambígua.

Estudos distintos realizaram categorizações aplicáveis aos elementos contextuais que ajudam na identificação do contexto. Destacam-se as classificações a seguir.

Em relação à granularidade, os elementos contextuais influenciam na classificação do contexto (WANG et al., 2004), em que **contextos básicos** são

caracterizados por elementos contextuais de granularidade baixa e de identificação direta (e.g. número de identidade, temperatura, localização). Já **contextos complexos** são derivados ou inferidos via composição de elementos contextuais de granularidade mais baixa. Por exemplo, um contexto básico de localização pode indicar onde está uma pessoa e a composição da localização, horário e identificação da pessoa podem caracterizar um contexto complexo inferido sobre um professor que está realizando uma palestra em um evento.

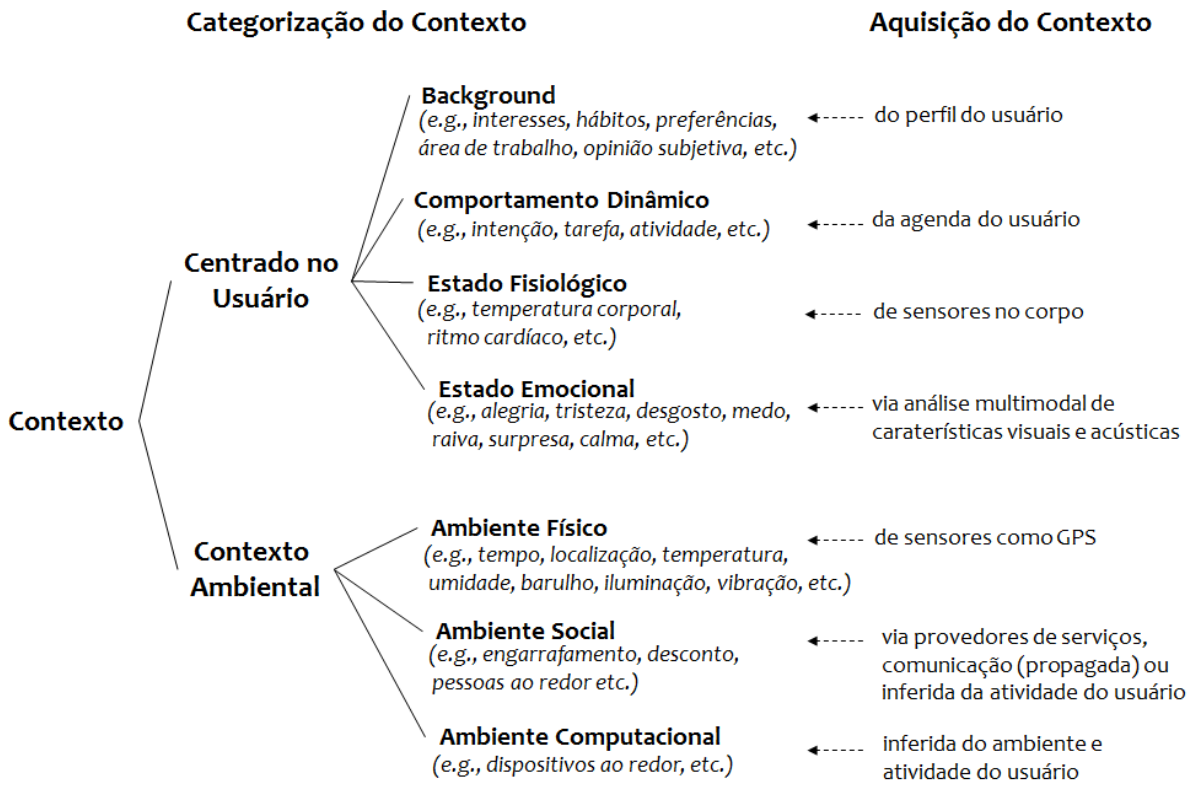
Em relação à frequência de atualização, pode-se classificar a dinamicidade (HONG; LANDAY, 2004) dos elementos contextuais em estáticos e dinâmicos e, por conseguinte, a classificação dos seus contextos correspondentes:

- **Contexto estático** - baseado em informações contextuais que não mudam ou mudam pouco frequentemente. Por exemplo, informações pessoais de usuários e identificações de objetos; e
- **Contexto dinâmico** - indicado por elementos contextuais que mudam e, portanto, precisam de monitoramento constante, tais como localização de objetos móveis, temperatura corporal e atividades realizadas por pessoas.

Feng et al. (2004) definem uma categorização de contexto baseada no seu foco de análise, dividida em dois tipos: **centrado no usuário** e **centrado no ambiente** (vide Figura 2.1). O contexto, aqui, é visto com um espaço *n*-dimensional, composto por *n* atributos contextuais. Cada dimensão do contexto é representada por um atributo contextual que descreve uma perspectiva do contexto. O domínio de um atributo contextual pode ser um valor numérico, uma sequência de caracteres, um conjunto de valores numérico/alfabético, vazio ou nulo, dependendo da semântica da aplicação. Também na Figura 2.1, são relacionadas as formas de aquisição de informação contextual para cada categoria de contexto.

Do ponto de vista da fonte de dados, Stefanidis et al. (2011) definem contexto como qualquer informação **externa ao banco de dados** (BD), que pode ser usada para caracterizar a situação de um usuário ou informação **interna ao BD** usada para caracterizar o dado propriamente dito. Para os autores, o contexto define a maneira como dados são representados e interpretados em cada banco de dados (PITOURA et al., 2004).

Figura 2.1 – Categorização e aquisição de Contexto



Fonte: Feng et al. (2004), tradução nossa.

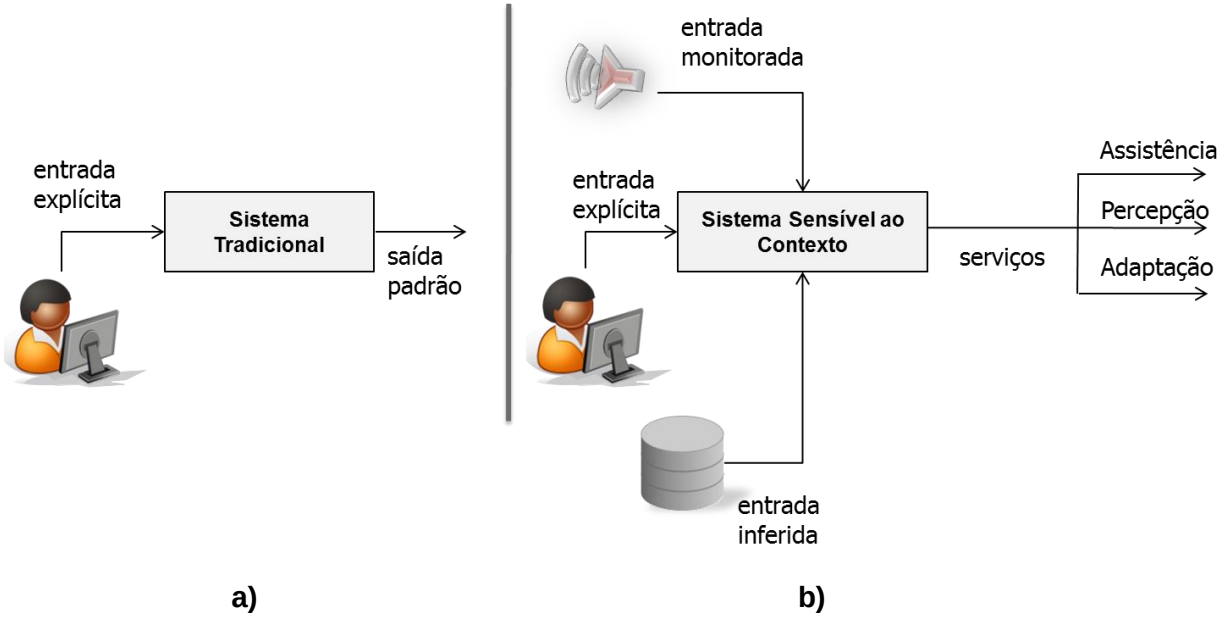
2.2. Sistemas Sensíveis ao Contexto

Aplicações tradicionais são sistemas computacionais que agem levando em consideração apenas as solicitações e informações fornecidas explicitamente pelos usuários (Figura 2.2 a). Já as aplicações sensíveis ao contexto consideram as informações explícitas fornecidas pelos usuários, informações armazenadas em bases de conhecimento contextual e informações adquiridas por meio de inferência ou percebidas a partir do monitoramento do ambiente (Figura 2.2 b) (VIEIRA et al., 2009).

A partir da compreensão do contexto, o sistema pode, em circunstâncias diversas, mudar sua sequência de ações, o estilo das interações e o tipo de informação fornecida aos usuários, de modo a adaptar-se às necessidades atuais destes. Sistemas que utilizam contexto para direcionar ações, recebem o nome de Sistemas Sensíveis ao Contexto (SSC) (CHALMERS, 2004; DEY, 2000; DEY; ABOWD, 2001; VIEIRA et al., 2011).

De uma forma geral, o uso do contexto permite aos sistemas computacionais prover serviços ou informações relevantes aos seus usuários (DEY; ABOWD, 2001), reduzir ambiguidades e, consequentemente, aumentar a satisfação do usuário.

Figura 2.2 – Visão geral comparativa entre aplicação tradicional e SSC



Fonte: Vieira et al. (2009)

O uso do contexto em SSC tem a finalidade de apoiar um agente na execução de alguma tarefa. Os elementos básicos manipulados pelo SSC são os elementos contextuais. O apoio fornecido pelo SSC pode estar associado tanto a possibilitar adaptações no comportamento do sistema de acordo com mudanças no contexto, quanto a ampliar a percepção do agente com informações relevantes, relacionadas à tarefa em execução (VIEIRA et al., 2009).

Como exemplos de SSC podem ser citados três casos, dentre os mais conhecidos:

a) WAZE[2] é uma aplicação para trânsito utilizada em *smartphone* que permite que seus usuários digitem um endereço de destino, e ao dirigirem seus veículos com a aplicação ligada, passam a contribuir passivamente com informações por onde trafegam (tempo de deslocamento e localização) ou ativamente, compartilhando alertas sobre trânsito, acidentes, perigos, polícia e outros eventos ao longo do percurso, ajudando outros usuários da mesma

[2] WAZE, disponível em:https://www.waze.com. Acesso em: 15 dez. 2014.

área com informações atualizadas sobre o que está acontecendo ao redor. A aplicação fornece informações contextualizadas em função do itinerário dos usuários, como: rotas e mapas com informações em tempo real, alertas, sistema de navegação por GPS e informações adicionais como postos de combustível próximos (vide Figura 2.3).

Figura 2.3 – Tela da aplicação WAZE

Fonte: www.waze.com. Acesso em: 15 dez. 2014.

b) O GMail[3] é um sistema de gerenciamento de e-mails que utiliza informações contextuais para adaptar seu conteúdo publicitário às preferências do usuário. No GMail há uma área específica para utilização de propagandas dentro da tela de visualização de e-mails. Essas propagandas são sensíveis ao contexto e possuem seu conteúdo relacionado ao assunto (sob a forma de palavras-chave) contido no e-mail que estiver sendo exibido, e também no histórico de e-mails. Essa contextualização da propaganda procura oferecer produtos ou serviços que, provavelmente, o usuário necessita ou gostaria de adquirir.
c) O Amazon.com[4] é um portal de vendas que usa informações contextuais para adaptar seu conteúdo às necessidades do usuário. Ao consultar por um determinado produto, além de informações específicas dele, o usuário

[3] GMAIL, The Google Mail. 2010. Disponível em: http://mail.google.com. Acesso em: 15 dez. 2014.
[4] AMAZON, *Online Shopping for Eletronics, Apparel, Computers, Books, DVDs & more*. Disponível em: http://www.amazon.com. Acesso em: 15 dez. 2014.

> recebe recomendações de outros produtos que podem lhe interessar. Para fornecer tais recomendações o sistema cruza informações navegacionais do cliente (e.g. produtos que comprou, consultou ou pesquisou, juntamente com o produto pesquisado no momento). A partir da informação obtida, o sistema a compara com as informações referentes a outros usuários para obter padrões do tipo “usuários que compraram certo produto também compraram outro” e assim poder fornecer sugestões de possíveis produtos de interesse.

Correntes de pesquisa têm devotado tempo e esforço consideráveis para desenvolver metamodelos e modelos para SSC (BALDAUF et al., 2007; BETTINI et al., 2009; BOLCHINI et al., 2007a), metodologias para desenvolvimento de SSC (HENRICKSEN; INDULSKA, 2006; VIEIRA, 2008) e modelos para representação, armazenamento e busca de dados contextuais (BUNNINGEN, 2008; SHEN et al., 2005; WANG et al., 2004).

Por se tratar de um conceito intuitivo, o contexto precisa ser formalizado para ser usado em sistemas computacionais. Um bom formalismo para modelagem da informação contextual reduz a complexidade das aplicações sensíveis ao contexto e aprimora suas habilidades para manutenção e evolução (BETTINI et al., 2009). A seção a seguir abordará a modelagem do contexto.

2.3. Modelagem do Contexto

Modelos de contexto, em geral, são usados para especificar os conceitos usados e para identificar e descrever situações em um domínio. Diferentes abordagens para modelagem do contexto têm sido utilizadas (STRANG; LINNHOF-POPIEN, 2004), sendo as mais relevantes: modelos baseados em pares de chave-valor (SCHILIT et al., 1994), modelos de linguagem de marcação (e.g. XML[5], HTML[6], SGML[7]), modelos gráficos (BRÉZILLON, 2003), modelos baseados em ontologias (USCHOLD; GRÜNINGER, 1996), modelos orientados a objetos (POWER, 2003) e modelos baseados em lógica (AKMAN; SURAV, 1997; BUVAČ; MASON, 1983; McCARTHY, 1993, SERAFINI; BOUQUET, 2004).

---

[5] XML disponível em: http://www.w3.org/XML/. Acesso em: 15 dez. 2014.
[6] HTML disponível em: http://www.w3.org/html/. Acesso em: 15 dez. 2014.
[7] SGML disponível em: http://www.w3.org/MarkUp/SGML/Overview.html. Acesso em: 15 dez. 2014.

Diferentes subproblemas de contexto e diferentes aplicações têm requisitos distintos e soluções genéricas ainda não estão disponíveis. Como consequência, o modelo de contexto deve ser selecionado dependendo do objetivo da aplicação. Abordagens híbridas que exploram técnicas diferentes em cenários de contextos específicos tendem a ser mais usadas como meio de superar as limitações de técnicas isoladas. Exemplos dessa combinação são os trabalhos de Henricksen et al. (2004) que combinam ontologias com um modelo gráfico baseado em *Object-Role Modeling* (ORM) e de Vieira (2008) que utiliza a representação contextual em um modelo de classes e o comportamento do sistema com grafos contextuais.

O Metamodelo de Contexto de Vieira

Metamodelos de contexto fornecem a infraestrutura conceitual para apoiar a construção de modelos de contexto específicos, abstraindo e formalizando os conceitos relacionados ao contexto. Um metamodelo pode guiar um projetista de SSC para elaborar seus próprios modelos de contexto.

Vieira (2008) propôs um Metamodelo da Estrutura de Contexto que serve de base para o modelo de contexto utilizado nesta tese. No metamodelo de Vieira (2008) há uma separação entre os conceitos de contexto e de elemento contextual. A definição de um elemento contextual (EC) pode ser realizada em tempo de projeto, sendo uma informação que pode ser conhecida, codificada e representada, enquanto o contexto é dinâmico e definido de acordo com uma tarefa ou interação particular, em tempo de execução (VIEIRA et al., 2011).

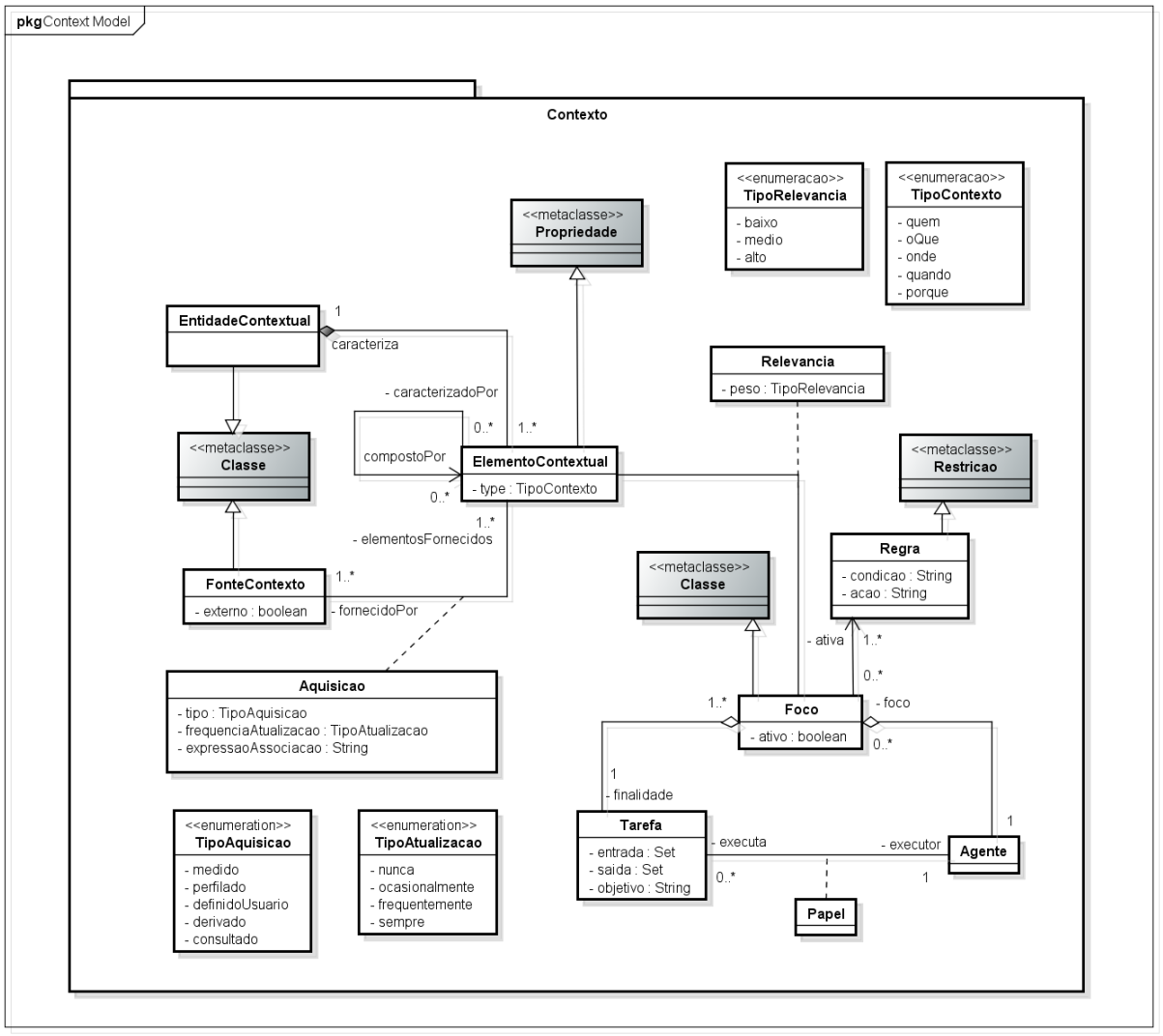
Os principais conceitos desse modelo estão descritos a seguir e representados na Figura 2.4.

Entidade Contextual

Entidades são objetos do mundo real relevantes para descrever um domínio e que podem ser identificados distintamente. Entidades são usadas para classificar indivíduos com características similares e contêm descrições desses indivíduos, representadas por atributos. São exemplos de entidades: `pessoa`, `funcionário`, `escola`, `automóvel`. Em se tratando de um modelo de contexto, Entidades Contextuais são entidades que devem ser consideradas para efeitos de

interpretação do contexto. Uma Entidade Contextual é caracterizada por pelo menos um Elemento Contextual.

Figura 2.4 – Metamodelo de Contexto



Fonte: Adaptado de Vieira et al. (2011).

Elemento Contextual

Um Elemento Contextual (EC) representa uma propriedade[8] usada para caracterizar uma Entidade Contextual. Um EC pode ser identificado a partir de um conjunto de atributos e relacionamentos associados a uma entidade. Por exemplo, os EC `nome`, `idade`, `sexo`, `naturalidade`, são atributos de uma entidade contextual `aluno`.

[8] Propriedade é definida como uma relação binária entre dois indivíduos, chamada relacionamento, ou entre um indivíduo e ele mesmo, chamada atributo.

Fonte de Contexto

Uma Fonte de Contexto é o lugar de origem de um elemento contextual. Um EC pode ser oriundo de fontes heterogêneas, internas ou externas ao SSC (e.g. interfaces de consulta, perfis de usuário, sensores físicos, bancos de dados).

Agente, Tarefa e Foco

Agente é quem executa uma tarefa e pode ser uma pessoa, um grupo de pessoas, um processo ou um agente de *software*. O Foco é definido como a composição de uma tarefa e o agente que a está executando. Por exemplo, em um sistema de reserva de mesas de um restaurante, um agente `Cliente` pode executar a tarefa `Solicitar Mesa`. A tupla <`Cliente`, `Solicitar Mesa`> constitui o foco. O Foco determina quais elementos contextuais devem ser instanciados e usados para compor o contexto.

Relevância

Pela definição adotada, o contexto é um conceito dinâmico que deve ser reconstruído a cada novo foco, sendo composto por todos os EC que são relevantes para apoiar a tarefa. O nível de relevância dessa associação é afetado pelo agente que está executando a tarefa. Por exemplo, em um sistema de venda de pacotes turísticos, se o agente for um `estudante`, a tarefa `reservar hotel` deve ter relevância maior para o elemento contextual `preço` com tarifas mais baratas.

Regra

Regras estão relacionadas com o aspecto dinâmico da manipulação do contexto, que é a identificação de EC relacionados a um *foco* (e suas relevâncias) e a determinação de como o sistema deve agir de acordo com as variações do contexto.

O processo de identificação de um EC e do comportamento do SSC considera a utilização de regras associadas. No Metamodelo de Contexto, uma regra é representada por um conjunto de uma ou mais *condições*, e um conjunto de uma ou mais *ações*. Cada valor instanciado do EC representa uma expressão de condição, que resulta em um valor verdadeiro, falso, ou nulo (desconhecido) quando avaliada com os dados disponíveis. Uma ação indica um procedimento que deve ser

executado quando as condições da regra são satisfeitas (por exemplo, para desencadear o comportamento de um sistema, para atribuir um novo valor para o EC, ou para atribuir um novo peso para a relevância de uma associação entre o EC e um Foco) (VIEIRA et al., 2011).

Exemplos de regras são as Regras de Produção (NEWELL, 1973; DAVIS; KING, 1975), que são sentenças lógicas que modelam o conhecimento de forma muito próxima ao processo cognitivo humano. Tais regras consistem em duas partes, uma pré-condição (SE) e uma ação (ENTÃO). Quando a pré-condição é satisfeita, então a regra é acionada e uma asserção é feita ou ação é executada (RUSSELL; NORVIG, 2014).

2.4. SGBD e o Dado Contextual

> Em termos práticos, apesar de muitas pesquisas terem foco no contexto, aquisição eficiente do contexto e arquiteturas para computação móvel, pervasiva e sensível ao contexto; o aspecto do gerenciamento do dado sensível ao contexto é delegado às aplicações (ROUSSOS; SELLIS, 2008, tradução nossa).

Apesar da grande maioria das aplicações sensíveis ao contexto tomarem para si a tarefa do gerenciamento do dado contextual e a afirmação citada ainda hoje ser válida, os benefícios de esse gerenciamento ser realizado no SGBD vêm sendo objeto de pesquisa. Nesta seção serão apresentadas as características do armazenamento, gerenciamento e consulta do dado contextual em SGBD.

2.4.1. Gerenciamento do Dado Contextual em SGBD

Apesar de requerer novos modelos de dados e mecanismos de consulta, o manuseio dos dados na mesma camada das fontes de dados (SGBD) tem as seguintes vantagens, segundo Roussos et al. (2005):

a) Projeto de dados abrangente - contexto e suas relações com os dados são levados em consideração na fase de projeto do esquema de dados. Embora isso introduza uma complexidade maior ao projeto, a incorporação das condições de contexto no esquema do BD também habilita a verificação de

consistência em operações de inserção e atualização de dados dependentes do contexto.

b) Espectro de consultas mais amplo - modelos de dados e linguagens de consulta que diretamente incorporam informações contextuais podem ser mais expressivas, conduzindo a novos tipos de consultas tais como “mostre as fotos de câmeras que são mais caras quando pagas em dinheiro que quando pagas com cartão de crédito” ou “sob quais contextos não há fotos”.

c) Acesso eficiente aos dados - tendo o gerenciamento do contexto no SGBD, pode-se ter métodos especialmente projetados para melhoria de desempenho do acesso ao dado contextual nele armazenado.

O gerenciamento do dado contextual dentro do SGBD permite a realização de consultas aos dados levando em consideração o contexto do usuário e da aplicação (e.g. preferências de usuário, dispositivo de consulta usado, horário, localidade e condições ambientais). O processamento da resposta à consulta não demandará processamento posterior na camada da aplicação para realizar adequações ao contexto.

Outras razões adicionais tornam útil e, em alguns casos, até mesmo necessário manter os dados contextuais armazenados em uma estrutura persistente e gerenciável, dentre as quais podem ser destacadas:

a) Poupar tempo de processamento e esforço computacional, na camada da aplicação;
b) Concentrar todos os dados coletados por sensores, em um mesmo local, de forma estruturada;
c) Lidar com uma quantidade de dados difícil de processar, para dispositivos leves (e.g. *smartphone*, *Google Glass*[9]);
d) Registrar o dado contextual ao longo do tempo (contexto histórico); e
e) Tratar os dados contextuais de maneira uniforme, independentemente do dispositivo ou da plataforma usados.

[9] Google Glass, www.google.com/glass. Acesso em: 15 jun. 2014.

2.4.2. **Contextos Internos e Externos ao BD**

Do ponto de vista do banco de dados, Stefanidis et al. (2011) definem contexto como:

> "Contexto é qualquer informação externa ao banco de dados que pode ser utilizada para caracterizar a situação de um usuário ou qualquer informação armazenada internamente, que pode ser usada para caracterizar os dados por si". (STEFANIDIS et al., 2011, tradução nossa).

Tipicamente o contexto externo refere-se à situação externa ao BD como o contexto computacional (e.g. rede de dados), contexto do usuário (e.g. localização, humor, acompanhante), contexto do ambiente físico (e.g. temperatura, nível de ruído) e contexto temporal. Já o contexto interno refere-se a atributos armazenados como dados no BD, como por exemplo, o atributo gênero, que pode caracterizar contextos relativos a filmes (e.g. comédia, drama).

2.4.3. **Estratégias para Processamento da Consulta ao Dado Contextual**

Sobre as estratégias para processamento de requisições de consulta sensíveis ao contexto, Feng et al. (2004) definem:

1. *Contexto como condição do ambiente da consulta* - Em ambientes altamente dinâmicos, inteligentes e responsivos, usuários são convidados a fazer consultas *ad-hoc* em qualquer lugar e a qualquer hora. Essas consultas normalmente envolvem o contexto corrente (e.g. tempo, local, tráfego) como referencial de consulta. Exemplo: "Quais os próximos voos que eu posso pegar?".
2. *Contexto como condição ou alvo de consulta baseada no passado* - Para humanos, é mais fácil lembrar o contexto de consultas passadas, do que o dado propriamente dito. Exemplo: "Qual o valor médio do barril de petróleo durante a última crise do petróleo?" ao invés de "Qual o valor médio do barril de petróleo durante os anos de 1973 a 1980?".
3. *Contexto como restrição de consulta* - Por meio do Contexto, é possível inferir conhecimento em *background*, que pode ser explorado como restrição em consultas para as seguintes situações:

a) *Entendendo qual a real intenção do usuário* - quando um usuário submete uma consulta, ele tem em mente um propósito. Idealmente, o acesso ao banco de dados deve ser direcionado à tarefa específica do usuário. Por exemplo, se um usuário solicita informações sobre restaurantes na cidade, com a intenção de convidar clientes para almoçar nos próximos minutos, apenas restaurantes próximos fazem sentido para esse usuário.

b) *Buscando melhorar a utilidade da requisição de dados para o usuário* - a utilidade de uma consulta, normalmente é relativa ao contexto. Por exemplo, para um sistema de informação de rotas, pode não ser útil informar uma rota deserta e pouco iluminada, caso a consulta seja realizada em período noturno.

c) *Aperfeiçoando o formato de apresentação do conteúdo da consulta* - a apresentação adequada do dado a ser consultado pode ser baseada também no contexto. Por exemplo, exibir imagens em alta definição em dispositivos de tela grande, e imagens menores e de baixa definição em *smartphones*. Outro exemplo é realizar agregações ou sumarizações de dados em dispositivos com tela pequena.

4. *Contexto como critério para adequação do resultado da consulta* - Devido às limitações de pequenos dispositivos, é conveniente que as respostas de consultas sejam exibidas ordenadas da maneira potencialmente mais útil.

5. *Contexto como guia para a formatação da entrega do resultado da consulta* - A modalidade de exibição deve ser adaptada ao contexto do usuário. Por exemplo, se o usuário está dirigindo, é conveniente que a resposta da consulta seja dada por meio de dispositivo de voz, considerando que o usuário não tenha limitação auditiva.

Ainda na visão de Feng et al. (2004), no âmbito do processamento da consulta sensível ao contexto, os atributos do contexto são tratados como atributos normais de relações. O grupo sugere que o processamento de consulta seja dividido em três fases: pré-processamento, execução da consulta e pós-processamento.

Durante o pré-processamento, uma consulta original é limitada pela adição de restrições que podem incluir atributos de contexto (elementos contextuais, na visão do presente trabalho) limitados a valores exatos.

Após a execução da consulta, na fase de pós-processamento, os resultados podem ser classificados sendo observadas regras predefinidas (e.g. critérios de *ranking* ou ordenações).

2.5. Reescrita de Consultas

Reescrita de consultas é uma denominação genérica para técnicas usadas nas áreas de Banco de Dados, Sistemas de Recuperação de Informação e Busca na *Web*. As técnicas de reescrita podem ser aplicadas para três objetivos distintos: a melhoria das respostas obtidas, a melhoria do desempenho da consulta ou a integração de dados.

Em relação ao objetivo de melhoria das respostas, o termo **reformulação de consultas** é utilizado como sinônimo de reescrita de consultas. Necib e Freytag (2004) defendem que a reescrita de consultas pode atuar como uma **otimização semântica da consulta**, na qual várias formas de conhecimento semântico, dentre as quais metamodelos, enciclopédias, regras semânticas e restrições de integridade, auxiliem a transformação das consultas de usuários em consultas semanticamente equivalentes. Uma consulta pode ser definida por um conjunto de seleções e projeções sobre objetos de um banco de dados, que satisfaçam um conjunto de condições. Essas condições são definidas por um conjunto de termos que são usados para especificar as informações a serem recuperadas e determinam a resposta à consulta, por meio da coincidência ou não com os valores existentes no banco de dados. No trabalho referido, são usadas ontologias para expressar o conhecimento semântico e assistir à reformulação de uma consulta inicial para uma nova consulta com termos capazes de produzir resultados mais significativos e que atendam à intenção do usuário.

Já Kraft et al. (2006), no âmbito da busca na *Web*, definem a reescrita de consultas como o aumento de cada consulta com termos apropriados ao contexto da busca. Jansen et al. (2009) caracterizam **reformulação** como o processo de alteração de uma dada consulta para melhorar sua busca ou desempenho em recuperação de dados.

A técnica de **substituição de consultas** tem por objetivo substituir a consulta original do usuário por outra consulta similar, fortemente relacionada à consulta original, que contenha termos intimamente relacionados aos termos originais

(JONES et al., 2006). São utilizadas operações de mudança ortográfica, substituição por sinônimo, generalização e especialização, para que novos termos substituam os termos usados na consulta original do usuário. O trabalho de Jones e equipe (2006) considera a utilização de consultas e frases predefinidas derivadas de consultas de sessões do usuário e *rankings* de similaridade.

Ainda com o objetivo de melhoria de respostas, expansão e relaxamento de consultas são outras técnicas correlatas bastante empregadas na reescrita de consultas. **Expansão de consultas** é definida como o processo de aumentar uma consulta por meio da inclusão de termos adicionais, associados aos termos da consulta original, que levem à recuperação de dados mais relevantes para o usuário (AKRIVAS et al., 2002; ANDREOU, 2005). Da mesma forma que nas técnicas de reformulação e substituição, ontologias são bastante utilizadas como fontes de termos a serem adicionados (YAGUINUMA et al., 2007).

Relaxamento de consultas refere-se à generalização de uma consulta para capturar informações "vizinhas" como forma de obter informação possivelmente relevante (GAASTERLAND, 1997). Para Zhou et al. (2007), o relaxamento de consulta refere-se à simplificação da consulta por meio da flexibilização de restrições (na expressão da consulta) em busca de um número maior de resultados, mantendo-se o controle da probabilidade de resultados indesejados e definindo critérios de preferência para seleção dos relaxamentos "melhores" ou "mais úteis". Formas de promover o relaxamento incluem: remoção de partes da consulta, generalização critérios (superclasses), supressão de proposições falsas e utilização de relações de tolerância que atendam critérios caracterizados como difusos ou *fuzzy*[10] (BOSC et al., 2006).

Usualmente trabalhos com o objetivo de melhoria do desempenho da consulta utilizam o termo **otimização de consultas** (CHAUDHURI, 1998; IOANNIDIS, 1996; LI et al., 2010). Esse objetivo foca em produzir outra consulta que obtenha mesma resposta, mas utilize menor tempo ou menos recursos computacionais durante a

[10] Critérios difusos dizem respeito à lógica difusa que, ao contrário da lógica booleana, admite valores lógicos intermediários entre o falso e o verdadeiro. Um valor lógico difuso é um valor entre 0 e 1, o que permite que estados indeterminados possam ser considerados e assim avaliar conceitos não-quantificáveis como a temperatura (e.g. quente, morno, médio), sentimentos (e.g. radiante, feliz, apático, triste...) ou a veracidade de um argumento (e.g. corretíssimo, correto, incoerente, falso, totalmente errado).

execução. É uma preocupação existente desde a estruturação dos SGBD na década de 1970.

Na área de Integração de Dados, o conceito de reformulação de consultas é amplamente utilizado como estratégia para integração de esquemas locais por meio de visões sobre esquemas mediadores (ARENS et al., 1996; HALEVY et al., 2006; SOUZA et al., 2008) como, por exemplo, as estratégias GAV (*Global as View*) / LAV (*Local as View*) (LENZERINI, 2002). Otimização de consultas e integração de dados não são foco de análise nesta tese.

No Capítulo 3 são apresentados exemplos de reescrita de consultas em trabalhos correlatos a esta tese.

2.6. Considerações

Este capítulo abordou os conceitos principais utilizados nesta tese. As definições de contexto, contexto computacional e a caracterização dos elementos contextuais são fundamentais para o entendimento deste trabalho. Também foram descritas características dos sistemas sensíveis ao contexto e do armazenamento, gerenciamento e consulta do dado contextual em SGBD.

Foram mencionados os aspectos envolvidos na modelagem do contexto e descrito o metamodelo de contexto de Vieira (2008), utilizado neste trabalho como base para representação contextual. Também, os conceitos de reescrita de consultas, especificamente os utilizados para melhoria das respostas, foram abordados por se tratar da técnica selecionada neste trabalho, para conferir sensibilidade a contexto, em consultas a bancos de dados.

O próximo capítulo apresenta uma revisão do estado da arte de trabalhos que oferecem propostas de como os SGBD podem utilizar o contexto, como consultar dados de forma contextualizada e de trabalhos que utilizam técnicas de reescrita de consultas.

3. SGBD E CONTEXTO

Neste capítulo é apresentado um levantamento do estado da arte em pesquisas sobre SGBD e contexto, com foco nas duas principais áreas relacionadas a esta tese. Na primeira seção do capítulo são descritos trabalhos centrados na modelagem do contexto. Nesses trabalhos a estrutura de representação do contexto guia a forma como os dados contextuais são armazenados, consultados e como o contexto é inferido. Já na segunda seção, são agrupados trabalhos que focam na consulta contextual, com destaque para a forma pela qual os procedimentos e técnicas desenvolvidos empregam estratégias de consulta (e.g. reescrita, algoritmos, operadores especiais, ontologias) que recuperem a informação levando em consideração o contexto corrente.

3.1. Trabalhos centrados na modelagem do contexto

Em virtude de se tratar de um tema que não apresenta correntes de pesquisa predominantes e nem padrões destacadamente aceitos, este levantamento foi realizado de forma extensiva, trazendo para análise conceitos e técnicas empregadas em outros modelos de banco de dados além do BD relacional, considerado nesta tese.

Nos trabalhos abordados a seguir, percebe-se a relevância dada à especificação formal de estruturas de dados que permitam suporte ao mapeamento, armazenamento e à estratégia de consulta ao dado contextual. As seguintes abordagens têm sido propostas: mecanismos de visões de dados, preferências baseadas em contexto, extensões ao BD relacional, extensões ao BD objeto-relacional e uso auxiliar de bases de conhecimento.

3.1.1 Mecanismos de Visões de Dados

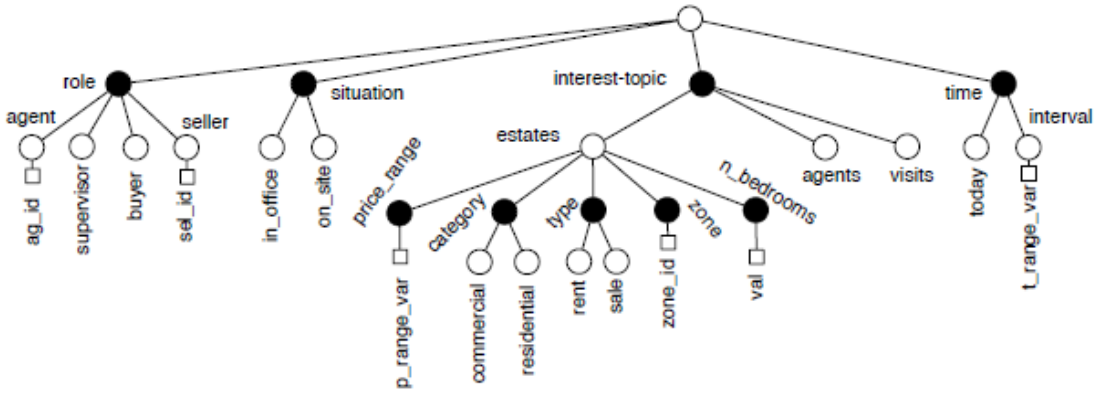
Nessa abordagem, proposta por Bolchini et al. (2007b, 2013), a informação contextual é modelada a partir de uma derivação do esquema global de dados de SGBD relacionais, por meio de uma sequência pré-determinada de operações

definidas pela metodologia CARVE *(Context-aware Relational View dEfinition*) (BOLCHINI et al., 2013).

A metodologia proposta especifica o contexto como conjuntos de variáveis, chamados *dimensões contextuais*, baseados no domínio da aplicação. Essas dimensões são modeladas em uma estrutura denominada *árvore de dimensões contextuais (ou CDT, do inglês Contextual Dimension Tree)*, usada para descrever e selecionar porções de informação consideradas necessárias para os usuários. Isso permite capturar o contexto desses usuários em um processo de adequação de dados denominado *data tailoring.*

Na Figura 3.1 temos um exemplo de árvore de dimensões contextuais, na qual os círculos pretos representam dimensões e subdimensões contextuais, os brancos representam conceitos das dimensões e os quadrados representam atributos.

Figura 3.1 – Exemplo de árvore de dimensões contextuais



Fonte: Bolchini et al. (2007b)

A CDT pode ser formalizada de várias maneiras, tais como a implementação em linguagem OWL[11] ou em representação XML[12] baseada em DTD[13] (do inglês, *Document Type Definition*).

Uma vez que a árvore de dimensões contextuais tenha sido projetada, representando as situações contextualizadas da aplicação, o próximo passo é derivar uma visão (de banco de dados) para cada possível contexto, adotando uma

[11] OWL disponível em: www.w3.org/TR/owl-features. Acesso em: 15 dez. 2014.
[12] XML disponível em: http://www.w3.org/XML/. Acesso em: 15 dez. 2014.
[13] DTD disponível em: www.w3schools.com/dtd/. Acesso em: 27 jan. 2015.

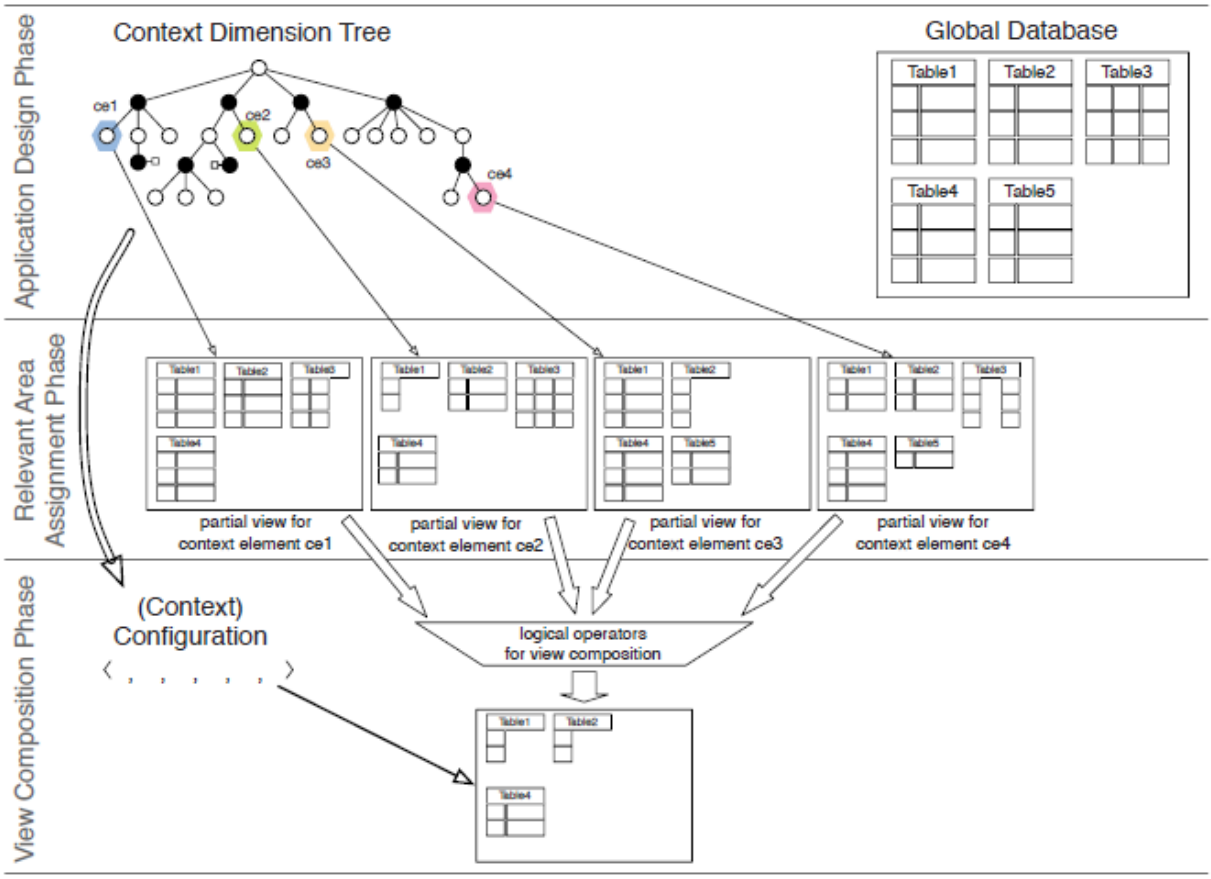
estratégia de mapeamento baseado em nós. O processo é composto por dois passos principais:

i. Assinalamento de áreas relevantes - a definição de visões parciais, expressas por um conjunto de expressões de álgebra relacional, associado com cada elemento contextual; e

ii. Composição de visões por operadores algébricos - combinação das visões parciais previamente definidas, para formar a definição de informações relevantes para obter a visão final.

Os autores definiram um novo grupo de operadores algébricos relacionais, e também estenderam a linguagem SQL (SQL ISO 1992), buscando dar suporte às operações de composição das visões parciais. Um panorama geral de todo o processo é mostrado na Figura 3.2. Uma vez que as visões finais de BD tenham sido produzidas, elas podem ser usadas como fonte de recuperação de informação contextual, apropriada para o usuário ou aplicação para as quais foram configuradas.

Figura 3.2 – A estratégia de mapeamento baseado em nós



Fonte: Bolchini et al. (2007b)

Na busca de reduzir o esforço de configuração de dados, a equipe de pesquisa tem investido no desenvolvimento da ferramenta *Relational Tailoring Assistant*, que tem o objetivo de auxiliar no processo de seleção de informações relevantes e composição de visões (BOLCHINI et al., 2013).

As principais críticas a essa abordagem são:

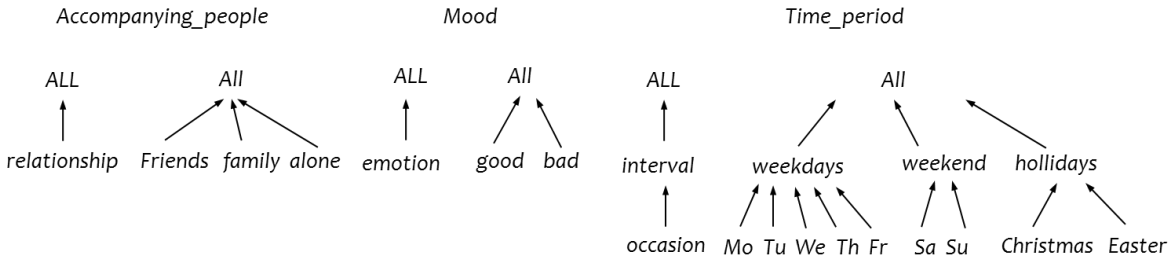
- O trabalho de composição de visões não poder ser inteiramente automatizado;
- Consultas sobre múltiplas visões (contextos) necessitam de junções complexas para serem manipuladas; e
- Ausência de suporte para gerenciamento do contexto dinâmico (em virtude das visões serem estruturas estáticas e predefinidas).

3.1.2 Modelo de Preferências Contextuais

Nesse trabalho (*Contextual Preference Model*) (PITOURA et al., 2011; STEFANIDIS et al., 2007), as preferências são vistas como dependentes do contexto, em função do fato que frequentemente usuários têm preferências diferentes, em diferentes circunstâncias. O contexto foi modelado como um conjunto de parâmetros contextuais que recebem valores de um domínio hierárquico com múltiplos níveis (por exemplo, o parâmetro *localização* pode receber valores dos domínios região, cidade ou país). Esses parâmetros representam informações que não fazem parte do banco de dados consultado. Formalmente, o contexto é modelado como um conjunto finito $\{C_1, C_2,..., C_n\}$ desses parâmetros (e.g. *{acompanhante, humor, período}*).

O modelo do contexto é representado em uma *hierarquia de contexto* que é a representação da hierarquia de um parâmetro de contexto em níveis e composta por uma árvore com nós em cada nível. A organização da *hierarquia de contexto* em níveis é denominada *hierarquia do esquema*, que quando instanciada recebe o nome de *hierarquia de conceito* (vide exemplo na Figura 3.3). Um *estado de contexto* é uma tupla $\omega = \{c_1, c_2,..., c_n\}$ que corresponde ao assinalamento de valor para cada um dos parâmetros de contexto, como ω = {*family, good, christmas*} no exemplo apresentado.

Figura 3.3 – Exemplo de hierarquias de esquema e de conceito

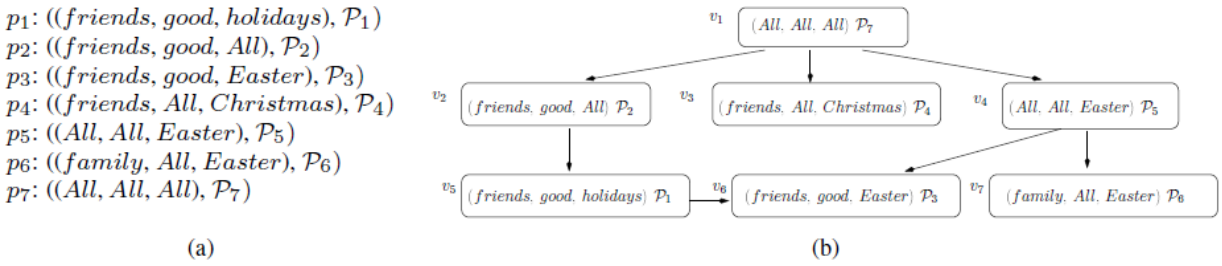


Fonte: Pitoura et al. (2011).

Para dar suporte à resolução eficiente do contexto, duas estruturas de dados foram desenvolvidas, o *grafo de preferências* (*preference graph*) e a *árvore de perfis* (*profile tree*) (STEFANIDIS et al., 2007), que permitem uma representação compacta de preferências dependentes de contexto.

O grafo de preferências corresponde a um conjunto de preferências, como apresentado na Figura 3.4. Isso torna possível um refinamento incremental de um estado de contexto. As preferências são especificadas atribuindo-se um número real entre 0 (sem interesse) e 1 (muito interesse) para determinados estados de contexto (que representam as tuplas de um banco de dados). Algoritmos para resolução de contexto usam ambas as estruturas auxiliares para consulta de um estado de contexto.

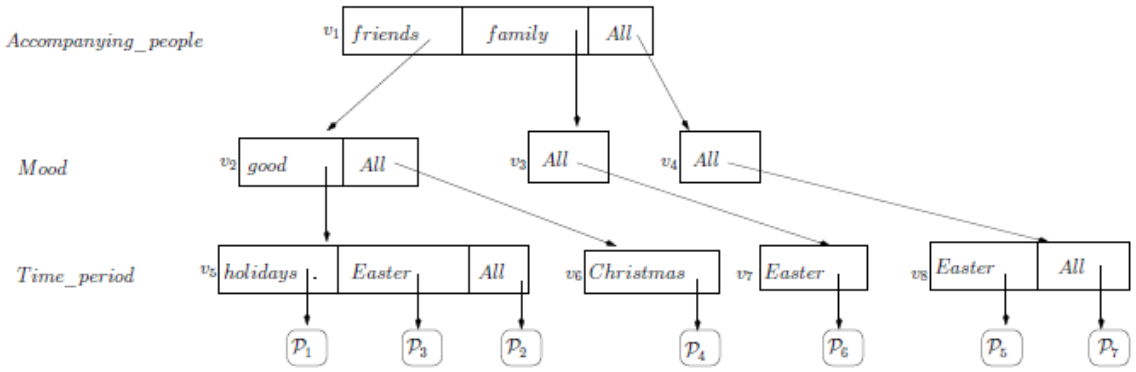
Figura 3.4 – Exemplos de preferências (a) e grafo de preferências (b)



Fonte: Pitoura et al. (2011).

Uma última estrutura de dados, a árvore de preferências, indexa preferências baseada nos seus contextos associados (Figura 3.5). Ela economiza mais espaço do que o grafo de preferências e tem melhor desempenho para consultas exatas.

Figura 3.5 – Árvore de preferências da Figura 3.4 a



Fonte: Pitoura et al. (2011).

Para uma preferência contextual ser selecionada, seu contexto deve ser o mesmo que o *estado de contexto* da consulta, ou mais genérico. Também são usados cálculos de distância entre estados para medir o grau de similaridade. Nesse trabalho as preferências são usadas para classificar o resultado após a consulta original ter sido executada embora a integração da preferência no processamento nativo do SGBD seja citada como um tópico promissor na área (STEFANIDIS et al., 2011).

Observa-se como pontos negativos desse trabalho a complexidade para modelagem do contexto e a dependência do mapeamento prévio de todas as opções de contexto do domínio e possíveis instâncias para os parâmetros contextuais[14]. O trabalho também carece de uma proposta para que a consulta ao banco de dados ocorra de forma sensível ao contexto.

3.1.3 Modelo Relacional Sensível ao Contexto

O estudo de Roussos e Sellis (2008) propõe um modelo de dados baseado na definição de relações multiesquema em que o "*Contexto é um parâmetro multidimensional que determina o escopo de uma instância ou de um esquema de uma relação*". Em outras palavras, a informação sensível ao contexto é representada sob a forma de relações, com diferentes instâncias ou mesmo diferentes esquemas de dados, em diferentes contextos, em um modelo relacional.

[14] Parâmetros contextuais, nesta tese, são conceituados como elementos contextuais.

O Contexto é formalmente especificado por meio da definição de um “Esquema de Contexto” (*Context Schema*), formado como um conjunto de atributos. Uma “Instância de Contexto” (*Context Instance*) é definida como o assinalamento de valores para os atributos de um determinado esquema de contexto.

Um “Especificador de Contexto” (*Context Specifier*) relacionado a um esquema de contexto é um conjunto de valores instanciados para um esquema de contexto. Um exemplo simples desta organização é mostrado no Quadro 3.1.

Quadro 3.1 – Exemplo de esquema, instância e especificador de Contexto

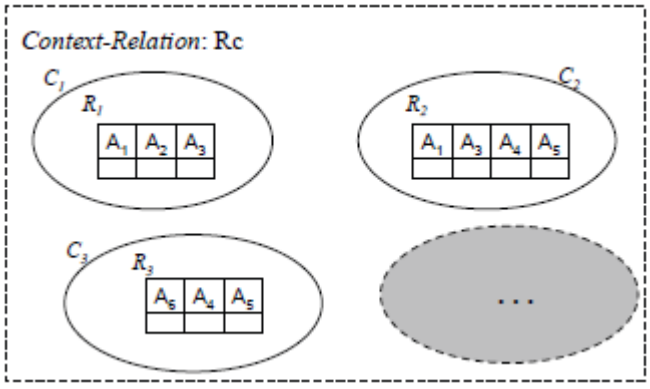
Context Schema C_s	Context Instance C_i	Context Specifier C_p
C_{s1} = <Language>	C_{i1} = <Italian>	C_{p1} = {<French>, <Greek>}
C_{s2} = <Location, Date>	C_{i2} = <*, 2005>	C_{p2} = {<*, 2003>, <Athens, 2004>}
C_{s3} = <Users, Devices>	C_{i3} = <david,PDA>	C_{p3} = { <george, *>, <jack,PC>, <mary,PC>}

Fonte: Roussos e Sellis (2008)

Como se pode perceber, o esquema de contexto é definido do mesmo modo que um esquema relacional; uma instância de contexto é como uma instância de uma relação nesse esquema; e um especificador de contexto corresponde a uma consulta sobre a relação com todas as possíveis instâncias de um esquema contextual.

Uma “Relação de Contexto” *Rc* relativa a um esquema de contexto *Cs* é definida como um conjunto de esquemas de relações {R1, R2,..., Rn}. Cada esquema de relação é associado com um especificador de contexto *Cp*, definido com respeito ao esquema de contexto *Cs*. Uma representação conceitual de uma relação de contexto é apresentada na Figura 3.6.

Figura 3.6 – Representação conceitual de uma relação de contexto Rc

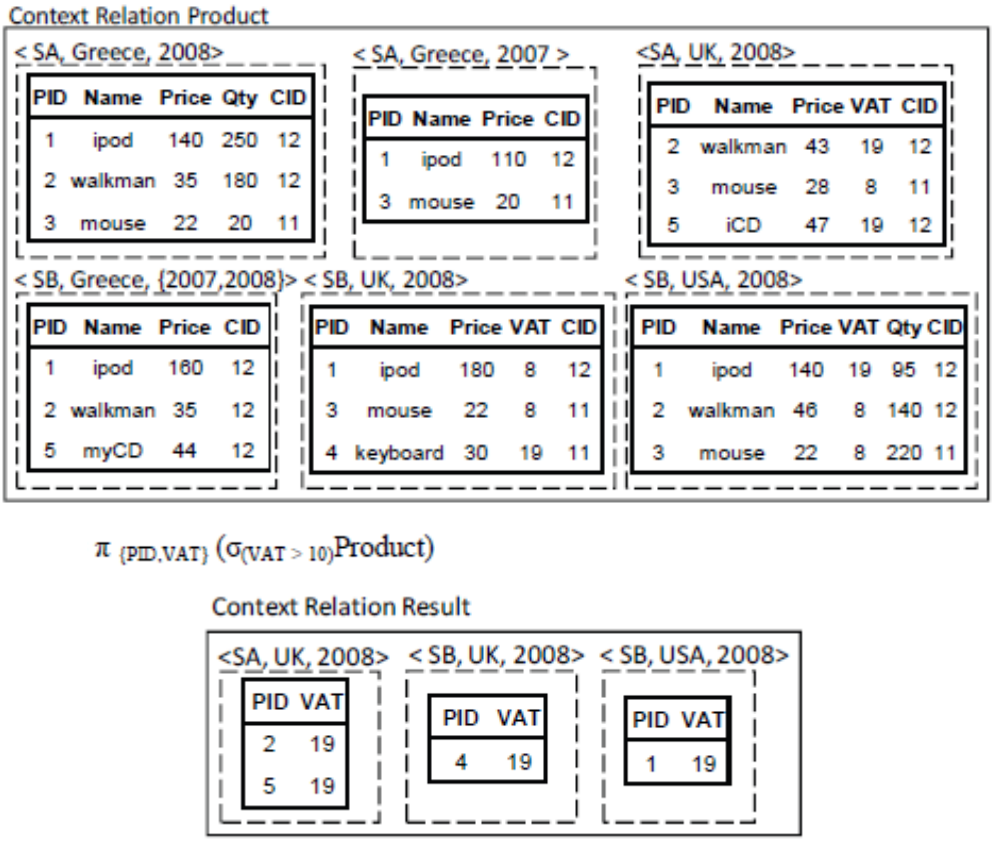


Fonte: Roussos e Sellis (2008)

Para dar suporte aos elementos descritos, um modelo de dados sensível ao contexto foi formalmente definido e também um conjunto de operações (e.g. *project, select, cartesian product, set*) sobre as relações contextuais, que estendem a álgebra relacional. Para os esquemas relacionais incluídos no esquema contextual também foi desenvolvido um conjunto adicional de operações (e.g. *schema_select, map, add context attribute, delete context attribute, de-contextualization, contextualization*) para referenciá-los e gerenciá-los.

Na Figura 3.7 é apresentado um exemplo de operação para a consulta "selecione todos os produtos (*Products*) que possuem imposto sobre venda (*VAT-value-added tax*) maior que 10%" ($\pi_{(PID,VAT)}$ $(\sigma_{(VAT>10)}$ $Product)$.

Figura 3.7 – Resultado de Operação de Consulta



Fonte: Roussos e Sellis (2008)

O principal ponto negativo do trabalho é a ausência da implementação dos operadores especificados em uma linguagem de consulta sensível ao contexto, extensão da linguagem SQL (SQL ISO 1992).

3.1.5 Extensão ao BD Objeto-relacional

O Chameleon (ELMONGUI et al., 2009), protótipo de SGBD sensível ao contexto construído por meio de extensão do SGBD PostgreSQL[15], apresenta-se como trabalho mais relevante para essa abordagem. A extensão feita baseia-se no usuário (emissor da consulta) e no dado a ser consultado. Ambos possuem seus contextos próprios: contexto do usuário e contexto de objeto. O contexto de um objeto pode ser um atributo ou um conjunto de atributos. Já o contexto do usuário é dividido em três dimensões, que irão se refletir na escolha do método de acesso das consultas afetadas por aquele contexto:

1. **Sinal do contexto**, que pode ser “negativo” |G| ou “positivo” |S|. O contexto *positivo* define o que é aceito no contexto. Por exemplo, numa consulta apenas para livros de capa dura, indica-se *positivo* para afirmar essa condição. Coso seja indesejável livro de ficção científica, deve-se aplicar *negativo* para evita-los.
2. **Relação contextual** indica a ordem de relevância de um dado contextual. Pode ser “relação de equivalência” |Q|, que indica que os dados que atendem aos requisitos serão retornados sem ordenação especial. Por exemplo, expressar que se deseja receber, de maneira equivalente, o resultado da consulta por livros de turismo e livros de viagens. A “relação de ordenação total” |T| indica que o resultado será ordenado de acordo com uma ordem de valores adequada para o dado. Por exemplo, para selecionar livros novos, a relação de ordenação total é apropriada para que esses venham primeiro. A terceira relação possível é a de “ordenação parcial” |P|, que não segue uma ordenação linear, como a ordenação total. Um exemplo de ordenação parcial seria para expressar preferência de um usuário por livros de ficção sobre livros de culinária e livros de turismo sobre livros de medicina, com nenhuma outra preferência específica entre outras combinações.
3. **Listagem de dados** refere-se a como os dados devem ser listados. Especificamente, se os dados não conformes com o contexto do usuário devem ser listados ou não. Há dois casos possíveis “não listados excluídos” |X| e “não listados incluídos” |N|. Como exemplo de “não listados incluídos”,

[15] PostgreSQL, disponível em: http://www.postgresql.org/. Acesso em: 15 dez. 2014.

um usuário gostaria de ver livros de capa dura, mas não se incomodaria se os outros livros viessem apresentados no final da listagem. A relação “não listados excluídos” não permitiria essa listagem adicional.

Construtores SQL objeto-relacional foram concebidos especialmente para habilitar a sensibilidade a contexto no SGBD Chameleon, são eles: `CREATE OBJECT CONTEXT`, `CREATE CONTEXT` (contexto de usuário), `SUBSTITUTE e SET ACTIVE CONTEXT`.

Um exemplo básico de construção de contexto e consultas a dados é apresentado a seguir, no qual o comando emitido pelo usuário é similar ao SQL tradicional. A consulta ‘*selecione todos os livros da livraria’* seria escrita da seguinte forma:

```
SELECT *
FROM books
WHERE books.stok;
```

Considerando que o contexto do usuário é preferir apenas livros de uma determinada categoria (por exemplo, ficção científica), o contexto deve ser definido pelo comando:

```
CREATE POSITIVE CONTEXT ctxt_category_SQX (
       category varchar(20),
       BINDING KEY (category)
                    REFERENCES books (category)
) AS EQUIVALENCE WITH UNLISTED EXCLUDED;

SET ACTIVE CONTEXT AS ctxt_category_SQX;
```

Quando o usuário emitisse a consulta, o comando real seria:

```
SELECT T.*
FROM books T
     INNER JOIN ctxt_category_SQX C1
     ON (T.category = C1.category
     AND C1.user_name = CURRENT_USER)
WHERE T.stock;
```

O foco inicial do Chameleon é a implementação das funcionalidades de contexto de privacidade e contexto espacial. Para ser possível a composição de contextos (contexto complexo) foi projetado um operador de consulta *Skyline* (BÖRZSÖNYI et al., 2001). Consultas *Skyline* são úteis quando múltiplos parâmetros são independentes e suas ordenações não podem ser agregadas em conjunto. Por exemplo: selecione todos os imóveis que tenham aluguel barato e sejam próximos da praia. Os operadores baseados em SQL tradicional são incapazes de estabelecer ordenações deste tipo.

A proposta principal do Chameleon é incorporar sensibilidade a contexto, dentro do SGBD. Permitir tal incorporação não só elimina a necessidade de adaptar sistemas especializados em direção a um determinado contexto, mas também permite a combinação de vários contextos para formar um contexto complexo.

Como pontos desfavoráveis do trabalho, apontamos:

- Aspecto pouco dinâmico, cujas opções de preferências para cada contexto admitido no domínio devem ser previstas e armazenadas em forma de código no BD.
- Ausência de um mecanismo ou estrutura que permita a inferência do contexto. Em outras palavras, o gatilho que dispara a ativação do contexto.

3.1.6 O Modelo CIM

O Modelo CIM (*Contextual Interest Model*) (BUNNINGEN, 2008; BUNNINGEN et al., 2006), baseado no interesse contextual do usuário, foi proposto como forma de adequar o interesse do usuário à apresentação do dado contextual. O objetivo foi prover um modelo para Ambientes Inteligentes (*Ambient Intelligence*), mas que pode ser aplicado a outros tipos de contexto.

O modelo armazena um conjunto ponderado (com pesos associados) de regras de preferência em conjunto com uma função de combinação de pesos e relevância. Cada regra indica o interesse de um usuário em um determinado contexto.

Os interesses e contextos são representados em Lógica de Descrição (LD) e armazenados em uma base de conhecimentos para dar suporte a mecanismos de raciocínio e manter as regras de maneira compreensível para os usuários. A função de combinação de escores determina a força (relevância) da preferência do usuário, no caso de uma situação ser satisfeita simultaneamente por mais de uma regra. As escolhas feitas em consultas passadas (histórico) formam a base para a atribuição dos pesos associados às preferências. O modelo CIM tem um protótipo implementado como extensão do SGBD PostgreSQL.

O contexto é utilizado sob duas perspectivas: centrado no usuário e ambiental. O contexto centrado no usuário refere-se às suas experiências (e.g. área de trabalho, amigos), estado físico-psicológico (e.g. temperatura corporal, ritmo cardíaco) e estado emocional (e.g. feliz, triste, amedrontado). Já contextos ambientais referem-

se ao ambiente físico (e.g. localização, tempo, umidade), ambiente social (e.g. congestionamento no trânsito, pessoas ao redor) e ambiente computacional (e.g. dispositivos ao redor).

No modelo desenvolvido, tanto preferências como contexto são tratados da mesma forma, usando LD. A base de conhecimento em LD é formada por um vocabulário com asserções sobre conceitos (e.g. `criança, feminino`) e regras (e.g. `temQuarto, temAtividade`). Conceitos e regras podem ser construídos usando os construtores: intersecção (⊓), união (⊔), complemento (¬), topo (⊤, todos os indivíduos), base (⊥, nenhum indivíduo), existe (∃) e para todo (∀). O operador -1 define o inverso de uma regra, por exemplo, o inverso de `temSala` é atribuído por $\texttt{temSala} \equiv \texttt{salaDe}^{-1}$. LD é usada para configurar uma ontologia para descrever as preferências contextualizadas das consultas, como mostrado na Figura 3.8.

Figura 3.8 – Uma ontologia simplificada em lógica de descrição

$Pessoa \sqsubseteq Coisa \sqcap \forall\ temSala.Sala$	$TipoAtividade \sqsubseteq Coisa$
$\sqcap \forall\ temTipoAtividade.TipoAtividade$	$AtividadeTempoLivre \sqsubseteq TipoAtividade$
$\sqcap \forall\ temAmigo.Pessoa$	$Lazer \sqsubseteq AtividadeTempoLivre$
$\sqcap \forall\ temInteresseTv.Genero$	$Esporte \sqsubseteq AtividadeTempoLivre$
$\sqcap \forall\ temAmigo.Pessoa$	$Local \sqsubseteq Coisa$
$Sala \sqsubseteq Local$	$Genero \sqsubseteq Coisa$
$ProgramaTV \sqsubseteq Coisa \sqcap temGenero.Genero$	$temInteresseTv \equiv interessadoEm^{-1}$

Fonte: Adaptado de Bunningen et al. (2006)

No trabalho proposto, as preferências são expressas como no exemplo a seguir, que descreve um contexto no qual o usuário Pedro está fazendo alguma atividade de lazer com pelo menos um amigo, na sua sala:

```
{PEDRO} ⊓ (∃ temTipoAtividade.AtividadeTempoLivre)
        ⊓ (∃ temAmigo (∃ temSala (∃ salaDe.{Pedro})))
```

O contexto é formalmente definido por uma tupla de forma (*Contexto, Preferência*), na qual *Contexto* e *Preferência* são expressões em LD. Por exemplo:

```
Context: {PEDRO} ⊓ (∃ temTipoAtividade.AtividadeTempoLivre)
Preference: ProgramaTv  ⊓ (∃ temGenero.{TALKSHOW})
```

Para transformar as preferências contextuais em um formato possível de ser consultado em um banco de dados no modelo relacional, cada conceito é mapeado para uma tabela, que usa o nome do *conceito* como o nome da tabela, tem apenas

um atributo e as tuplas correspondem a todos os indivíduos do *conceito*. Da mesma forma, cada *papel* é visto como uma tabela com mesmo nome do *papel* e dois atributos *origem* e *destino*. `Toptable` é uma tabela virtual que possui todos os indivíduos do domínio. A Figura 3.9 mostra o banco de dados do exemplo.

Figura 3.9 – BD do exemplo apresentado

Pessoa

ID
ERIC
PEDRO
MARCOS
....

Sala

ID
ROOM3061
ROOM4061
....

Programa

ID
OPRAH
24h
MALHACAO
....

Lazer

ID
TALKSHOW
DORMIR
ASSISTIR TV
....

temTipoAtividade

ORIGEM	DESTINO
ERIC	LER
PEDRO	ASSISTIR TV
MARCOS	DORMIR
....	

temInteresseTv

ORIGEM	DESTINO
ERIC	TALKSHOW
PEDRO	TALKSHOW
MARCOS	SCIFI

temAmigo

ORIGEM	DESTINO
ERIC	PEDRO
PEDRO	ERIC
PEDRO	MARCOS
....	

temSala

ORIGEM	DESTINO
ERIC	ROOM3061
PEDRO	ROOM3061
MARCOS	ROOM4061

temGenero

ORIGEM	DESTINO
OPRAH	TALKSHOW
24H	AVENTURA
VOYACER	SCIFI

Fonte: Adaptado de Bunningen et al (2006)

Para realizar o mapeamento de LD para consultas SQL, foi utilizada a estratégia de Borgida e Brachman (1993) para expressar conceitos em LD em consultas SQL, conforme Quadro 3.2.

Quadro 3.2 – Mapeamento de expressões em LD para expressões em SQL

DL	SQL
C	SELECT ID FROM C
a	VALUES ('a')
$\top$	(SELECT ID FROM TOPTABLE)
$\bot$	NULL
$\neg D$	(SELECT ID FROM TOPTABLE) EXCEPT (SELECT ID FROM D)
$D \sqcap E$	(SELECT ID FROM D) INTERSECT (SELECT ID FROM E)
$D \sqcup E$	(SELECT ID FROM D) UNION (SELECT ID FROM E)
$\exists R.D$	SELECT R.SOURCE FROM R WHERE R.DESTINATION IN (SELECT ID FROM D)
$\forall R.D$	(SELECT ID FROM TOPTABLE) EXCEPT (SELECT R.SOURCE FROM R WHERE DESTINATION IN ((SELECT ID FROM TOPTABLE) EXCEPT (SELECT ID FROM D)))

Fonte: Bunningen et al (2006)

Com esse mecanismo de mapeamento, foi possível construir consultas SQL para expressões em LD como no exemplo no qual um usuário realiza uma consulta sobre os programas de TV:

```
SELECT ID FROM Programa;
```

A preferência contextual contém um conceito `Programa` que aparece na consulta:

```
Context: {PEDRO} ⊔ (∃ temAmigo:( ∃ temSala( ∃ salaDe:{PEDRO})))
Preference: Programa ⊔ (∃ temGenero{TALKSHOW})
```

A expressão do contexto é mapeada diretamente na consulta:

```
SELECT * FROM ((VALUES ('PEDRO'))
INTERSECT
(SELECT temAmigo.ORIGEM FROM temAmigo
        WHERE temAmigo.DESTINO IN
(SELECT temSala.ORIGEM FROM temSala
        WHERE temSala.DESTINO IN
(SELECT salaDe.ORIGEM FROM salaDe
        WHERE salaDe.DESTINO IN (VALUES ('PEDRO')))
 ))) AS tabelacontexto;
```

Que pode se otimizada para:

```
SELECT *
FROM temAmigo, temSala, salaDe
WHERE temAmigo.ORIGEM  = 'PEDRO'          AND
      temAmigo.DESTINO =  temSala.ORIGEM AND
      temSala.DESTINO  =  salaDe.ORIGEM  AND
      salaDe.DESTINO   = 'PEDRO';
```

A consulta, portanto, é baseada no conhecimento que `salaDe` é o inverso de `temSala`, e lazer engloba `dormir`, `assistirTv`, entre outros.

Se uma consulta resultar em vazio, a consulta original será incrementada com a preferência, por meio da adição da cláusula WHERE:

```
SELECT ID FROM Program
 WHERE ID IN
(SELECT temGenero.ORIGEM  FROM temGenero WHERE
        temGenero.DESTINO IN  (VALUES ('TALKSHOW'));
```

Um protótipo deste trabalho foi implementado em uma camada sobre o SGBD DB2[16], utilizando um *plugin* para o Protegé[17], um *framework* editor de ontologias e bases de conhecimento. Apenas uma versão foi divulgada como tendo sido implementada, na qual consultas ao banco de dados são disparadas automaticamente por *triggers* do SGBD (*push queries*) e "oferecem" a resposta após um contexto e uma preferência serem atribuídos. No exemplo a seguir, observa-se a implementação de um *trigger* e o assinalamento dos parâmetros (as variáveis v) que o fazem ser disparado. A resposta gerada deve incluir o gênero '*talkshow*', conforme a base de dados apresentada na Figura 3.9:

```
Context:{v} ⊔ ∃ temSala:{ROOM3061}
Preference:∃ interessadoTv:v
```

[16] DB2, disponível em: http://www-01.ibm.com/software/data/db2/. Acesso em: 15 dez. 2014.
[17] Protegé, disponível em: http://protege.stanford.edu/. Acesso em: 15 dez. 2014.

```
CREATE TRIGGER consutaInteresseTv AFTER INSERT ON temSala
REFERENCING NEW AS n FOR EACH ROW
WHEN (n.DESTINO IN VALUES ('ROOM3061'))
       SELECT ORIGEM
       FROM temInteresseTv
      WHERE DESTINO = n.ORIGEM;
```

Um ponto desfavorável da abordagem é a dependência da modelagem dos dados da base de dados relacional (de onde os dados são efetivamente consultados) em relação à modelagem das preferências da base de conhecimentos em LD. Se por um lado isso torna possível o raciocínio do motor de inferência que analisa a base de conhecimento lógica, por outro lado, torna o seu mapeamento para utilização com BD relacional rígido e, portanto, de difícil aplicação de melhorias de desempenho.

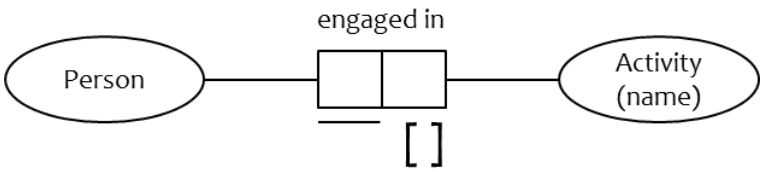
3.1.7 Mapeamento Relacional de Modelo ORM

Henricksen et al. (2003) propuseram a CML (*Context Modelling Language*), uma abordagem baseada em ORM (*Object Role Modelling*) (HALPIN, 2010), que busca mapear o contexto e foi criada originalmente para dar suporte ao processamento de consultas. O conceito básico em ORM é o “fato”. As tarefas de modelagem, portanto, envolvem identificar tipos de fatos apropriados e os papéis desempenhados pelas entidades que neles atuam.

A CML (HENRICKSEN et al., 2003; HENRICKSEN; INDULSKA, 2004; HENRICKSEN; INDULSKA, 2006) provê uma notação gráfica concebida para dar suporte à engenharia de *software* na análise e formalização de requisitos de um SSC. ORM tem a característica de poder ser mapeada para o modelo relacional de bancos de dados por meio de um processo chamado *Rmap* (HALPIN, 1995). Henricksen (2003) estendeu o *Rmap*, adaptando-o para mapear os modelos de contexto baseados em CML para BD relacional. Esse mapeamento possibilitou que a informação contextual (conjunto de fatos) fosse expressa em forma de tuplas de banco de dados.

Como pode ser observado na Figura 3.10, cada notação gráfica de ORM pode ser mapeada para classes que, por sua vez, podem ser mapeadas para entidades ou visões no modelo relacional. Tipos de contextos derivados são mapeados para visões.

Figura 3.10 – Exemplo de modelagem ORM




```
   EngagedIn(PersonName, StartTime, EndTime, ActivityName)
EngagedInNow(PersonName, Activity)
```

Fonte: Henricksen et al. (2003)

`EngagedInNow` é uma visão correspondente ao seguinte comando SQL:

```
CREATE VIEW EngagedInNow AS
     SELECT PersonName, ActivityName
       FROM EngagedIn
      WHERE CURRENT_TIMESTAMP >= StartTime and
           (CURRENT_TIMESTAMP <= EndTime or EndTime is null);
```

O caráter estático e predefinido das visões de dados torna pouco flexíveis as consultas a dados nos SSC desenvolvidos segundo esse modelo. Outro ponto negativo é que não há mecanismo para inferência do contexto, o que torna as aplicações restritas apenas às situações previstas e modeladas nas fases de projeto.

Resumo Comparativo dos Trabalhos Centrados na Modelagem do Contexto

O Quadro 3.3 apresenta o resumo das diferenças e similaridades entre trabalhos referenciados nesta seção. Com o objetivo de exibir uma visão geral dos trabalhos, os seguintes critérios de comparação são relacionados:

- Abordagem - identifica a linha de pesquisa de conduziu o desenvolvimento do trabalho;
- Modelagem - indica o tipo de modelo de representação do contexto utilizada;
- Recursos usados - resumo dos principais recursos (e.g. técnicas, processos, métodos) utilizados no trabalho; e
- Inferência do contexto - informa se há inferência do contexto no trabalho e indica a forma como é feita.

É possível perceber que ainda não há uma corrente de pesquisa preponderante quanto aos trabalhos que focam na modelagem do contexto para SGBD. As pesquisas que propuseram novos modelos de dados sensíveis ao contexto baseadas em extensões do modelo relacional (e.g. ROUSSOS; SELLIS, 2008) apresentam-se com fundamentação algébrica sólida, embora ainda tenham pela frente o enorme desafio da concepção e desenvolvimento de SGBD que implementem nativamente os seus fundamentos e atendam aos apelos de mercado por desempenho e usabilidade. Uma opção para extensão do modelo objeto relacional também foi proposta por Elmongui et al. (2009), embora focada para domínios específicos de localização e privacidade.

Os trabalhos de Stefanidis et al. (2007) e Bunningen (2008) são representantes de uma corrente que busca conciliar preferências e contexto em um único modelo. O primeiro por meio de um modelo baseado em estruturas hierárquicas e o segundo por meio de um modelo baseado em lógica de descrição, ambos com a intenção propiciar suporte à inferência do contexto.

Bolchini et al. (2007b) defendem um modelo baseado em visões de dados derivadas do esquema global de bancos de dados relacionais como forma de adequar consultas aos usuários, conforme seu contexto. Nos trabalhos de Henricksen et al. (2003) e (HENRICKSEN; INDULSKA, 2006), as visões de dados são resultado do mapeamento de modelos de contexto em ORM para SGBD relacional.

Quadro 3.3 – Comparativo entre trabalhos centradros na modelagem do contexto

Autor/Projeto/Ano	Abordagem	Modelagem	Recursos usados	Inferência do contexto
Bolchini et al. /CARVE/ 2013	Mecanismo de visões de dados	Gráfica (*ctx. tree*)	Visões de dados	Não
Henricksen et al. /CML/ 2003~2006	Mapeamento ORM para Relacional	Gráfica (ORM)	Mapeamento RMAP e geração de visões de dados	Não
Stefanidis et al./ *Contextual Preference Model* /2007~2011	Preferências	Gráfica (grafos e hierarquias)	Estruturas de dados auxiliares / análise algorítmica de preferências / classificação de resultados	Composição de parâmetros
Bunningen /CIM/2008	Preferências e contexto assistido por base de conhecimento	Regras em lógica de descrição	Algorítmos de classificação / aferição de relevância de preferências / motor de inferência	Regras em lógica de descrição
Roussos e Sellis,/2008	Extensão do modelo relacional	Relacional estendida	Construtores e operadores em álgebra relacional estendida	Não
Elmongui /Chameleon/ 2009	Extensão do modelo objeto-relacional	Orientada a objetos	Consulta tipo *skyline*	Composição de parâmetros

Fonte: O autor.

3.2. Trabalhos centrados na consulta contextual

Nesta seção será abordado o tema do contexto envolvido em consultas a bancos de dados. A primeira parte da seção é dedicada a trabalhos sobre *preferências,* um importante subconjunto do estudo do contexto em SGBD. A segunda parte analisará a literatura correlata sobre *reescrita contextual de consultas SQL*.

Como vimos, uma consulta pode gerar resultados diferentes, dependendo do contexto sob o qual ela foi executada. O processamento de uma consulta contextual pode ser visto como um processo realizado em dois passos: primeiro o contexto relevante para uma consulta é identificado e adquirido. Em seguida, esse contexto é integrado à consulta.

Segundo Pitoura et al. (2004) existem as seguintes formas básicas de integrar o contexto em consultas:

a) Contexto como parâmetro - por meio da integração explícita de parâmetros[18] contextuais na consulta. Parâmetros contextuais são tratados como atributos limitados ao valor corrente do contexto, quando a consulta é executada. No exemplo (fictício) a seguir, assume-se que o atributo tempo do contexto é limitado ao horário durante o qual a consulta é executada.

```
SELECT restaurante.id
  FROM restaurantes,contexto
 WHERE contexto.hora IN restaurante.horario_atendimento;
```

b) Contexto como predicado - outra forma de adquirir sensibilidade a contexto é expandir a consulta com predicados apropriados. Em particular, uma determinada consulta pode ser transformada em outra por meio da adição de restrições, quer seja pela adição de restrições aos atributos, quer seja pela associação de regras sobre atributos específicos ou relações, adicionadas à consulta. Por exemplo, a consulta a seguir retorna todos os restaurantes com espaço ao ar livre, quando o clima está bom, e nenhum em caso contrário.

```
SELECT restaurante.id
  FROM restaurantes;
```

[18] Neste trabalho, referenciados como elementos contextuais.

É transformada para:

```
SELECT restaurante.id
  FROM restaurantes,contexto
 WHERE (contexto.clima = 'ensolarado'
   AND restaurante.ar_livre = 'disponível')
    OR contexto.clima = 'chuvoso';
```

c) Contexto como associação para recordação - contexto pode ser usado para ser associado a eventos passados. Para que isso seja possível deve-se associar aos fatos ou objetos armazenados no banco de dados, informações sobre os elementos contextuais de quando esses fatos ou armazenamento de objetos ocorreram. Isso torna possível, por exemplo, a realização da consulta: “Retorne as fotografias da viagem ao Havaí”.

d) Contexto como preferência - em bancos de dados, o interesse nas preferências foi despertado observando as limitações da forma booleana de resposta, na qual critérios de consulta são inflexíveis (chamados *hard constraints,* que são as restrições declaradas na cláusula SQL WHERE) e uma resposta não vazia é retornada apenas se preencher todos os critérios da consulta. Desta forma, um usuário pode enfrentar um de dois problemas possíveis: 1) O problema da *resposta vazia*, na qual as condições são muito restritivas ou os dados não correspondem exatamente à consulta; ou 2) O problema de *respostas em demasia*, na qual também muitos resultados correspondem à consulta. É difícil lidar com esses problemas se o usuário não está familiarizado com a linguagem de consulta estruturada, em especial para a formulação de consultas precisas e ao acessar bancos de dados na web, cujo esquema e conteúdo são desconhecidos.

Incorporar critérios suaves (*soft constraints*) ou preferências em uma consulta pode ajudar a lidar com esses problemas. O problema da resposta vazia pode ser resolvido por meio do relaxamento de algumas das *hard constraints*, transformando-as em *soft constraints* ou as substituindo por outras restrições que possam retornar outros dados relacionados à consulta e que serão classificados de acordo com o quão bem eles correspondam à consulta original. Já os problemas das respostas em demasia podem ser resolvidos por meio do reforço da consulta com preferências adicionais para classificar e possivelmente focar os resultados da consulta.

De forma geral, há duas abordagens (BUNNINGEN et al., 2006; CHOMICKI, 2003) para expressar preferências em consultas:

- ***Qualitativa*** que busca especificar preferências entre tuplas de respostas a consultas como, por exemplo, “eu prefiro livros do gênero romance aos do gênero aventura se e somente se os primeiros tiverem preço menor que os do segundo tipo”. Essa abordagem pode ser embutida em linguagens de consulta relacionais por meio de extensões que utilizem operadores especiais ou algoritmos de classificação de tuplas (e.g. *Winnow* (CHOMIKI, 2003), Preference SQL BMO (KIEßLING et al., 2011) e *Skyline* (BÖRZSÖNYI et al., 2001)).
- ***Quantitativa*** que expressa preferências usando funções de classificação baseadas em pesos, que associam valores numéricos de relevância para cada tupla retornada pela consulta.

Quanto à forma de implementação em SGBD, segundo Levandoski (2010b) e Stefanidis et al. (2011), há três modos de se trabalhar com preferências:

- ***Tradução para SQL***, que traduz a sintaxe conceitual da preferência para comando SQL padrão e depois a submete ao SGBD. Essa opção não traz ganhos de expressividade à consulta, em relação ao que já é disponibilizado pelo SGBD.
- ***On-top****,* que usa a implementação de um programa externo ao SGBD ou se utiliza deste por meio de funções. O SGBD é tratado como uma caixa-preta e os operadores de preferências são desacoplados das operações internas do SGBD. É uma opção fácil de implementar e que tem a flexibilidade de trabalhar com SGBD diferentes.
- ***Built-in****,* que interfere no código do SGBD, implantando os operadores e estratégias de leitura direcionadas para o SGBD usado. Em termos de desempenho, esta estratégia potencialmente apresenta os melhores resultados, mas não possui a flexibilidade de usar SGBD diferentes.

3.2.1. **Trabalhos Baseados em Preferências**

Os trabalhos descritos nessa seção são exemplos representativos da corrente baseada no estudo de preferências contextuais.

3.2.1.1 CareDB

O CareDB (LEVANDOSKI et al., 2010a, 2011) é um sistema de banco de dados que provê serviços baseados em localização de acordo com as preferências do usuário e do contexto ao seu redor. O objetivo do CareDB é redefinir as respostas a consultas tradicionais.

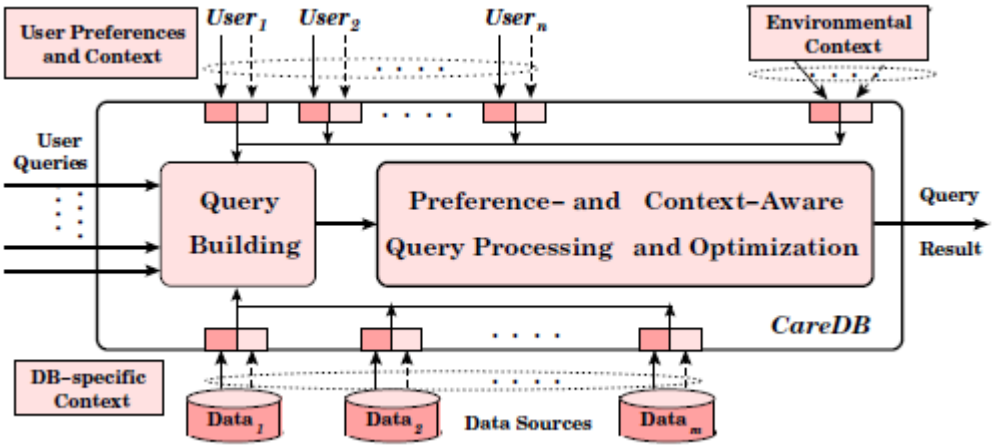
Como entrada, o CareDB utiliza preferências e dados contextuais. As preferências dizem respeito aos gostos dos usuários para atributos de dados em um domínio em particular (por exemplo, se um usuário busca um restaurante, seu perfil deve armazenar preferência de faixas de preço e restrições de dieta). As preferências de usuário são armazenadas em perfis que podem ser passados para o sistema ou aprendidas por intermédio do histórico de utilização.

O CareDB tem três tipos de entrada de contexto: o *contexto do usuário* é qualquer informação adicional sobre um usuário e pode ser estático, como profissão, ou dinâmico como localização. O c*ontexto do banco de dados* diz respeito aos dados de uma aplicação específica (por exemplo, uma base de dados de hotel) e também pode ser estático como preço e classificação ou dinâmico, por exemplo, o tempo de espera decorrido. *Contexto ambiental* é qualquer informação sobre o ambiente circundante, como condição de tráfego (dinâmico) ou clima (relativamente estático). Este último tipo de informação pode ser armazenado em instalações de terceiros e acessada via interface remota.

Na Figura 3.11 é mostrada a arquitetura do CareDB, que tem um protótipo implementado no SGBD de código aberto PostgreSQL[19]. O propósito do módulo de Construção de Consulta (*Query Building*) é personalizar as consultas para cada usuário, de modo que a melhor resposta seja retornada. Este módulo cria consultas de preferência (*preference queries*) enriquecendo a consulta enviada com as preferências armazenadas no perfil de preferências do usuário.

[19] PostgreSQL, disponível em: http://www.postgresql.org/. Acesso em: 15 dez. 2014.

Figura 3.11 – A arquitetura CareDB



Fonte: Levandoski et al. (2010a)

No módulo de Preferência e Processamento de Consultas Sensíveis ao Contexto (*Preference and Context-aware Query Processing module*), a *preference query* é usada como entrada e processada usando uma sintaxe estendida de SQL, com as cláusulas `preferring` e `using`, que especificam os atributos de preferência e qual método (de seleção ou ordenação, e.g. *skyline* (BÖRZSÖNYI et al., 2001) ou top-k (BRUNO et al., 2002; CHAUDHURI; GRAVANO, 1999) deve ser usado e os objetivos (e.g. min, de minimização, para o *skyline*) a serem aplicados sobre os atributos definidos na cláusula `preferring` e com valores específicos para parâmetros (e.g. k para top-k), conforme a sintaxe:

```
PREFERRING [preference attributes] USING [method]
      WITH [parameter] OBJECTIVES [objective]
```

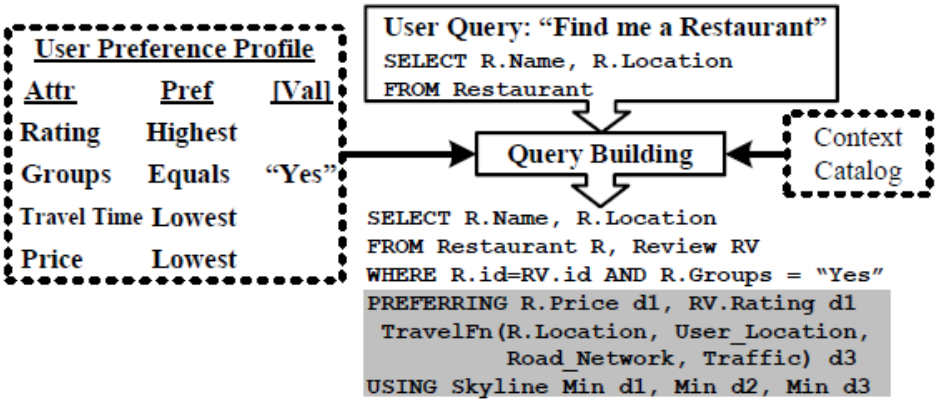
Por exemplo, a consulta a seguir aplica o operador *skyline* sobre `filmes`, na qual a preferência requer minimização da `duracao`, ao mesmo tempo da maximização do `ano` de lançamento:

```
SELECT * FROM filmes USING skyline
  WITH OBJECTIVES min duracao, max ano;
```

Então, a consulta é executada por um processador de consultas específico (e extensível) no SGBD que, por sua vez, incorpora algoritmos de vários tipos de processamento de consultas que permitam a exibição dos dados de forma a classificá-los e/ou filtrá-los de acordo com as preferências do usuário para a consulta em um SGBD relacional ou utilizando algoritmos de classificação (e.g. *skyline* (BÖRZSÖNYI et al., 2001), *top-k dominance* (YIU; MAMOULIS, 2007) e *k-dominance* (CHAN et al., 2006).

Outro exemplo de consulta pode ser visto na Figura 3.12, na qual, com base nas preferências do perfil do usuário, a consulta inicial é reformulada e acrescida do método *skyline* para classificação e filtragem das informações a serem retornadas.

Figura 3.12 – Exemplo de adequação de consulta e redefinição de resposta



Fonte: Levandoski et al. (2010b)

3.2.1.2 Preference SQL

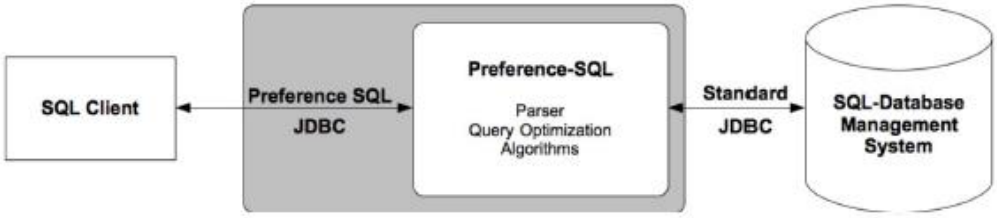
Preference SQL (KIEßLING et al., 2011, 2002) é uma extensão da linguagem SQL com preferências tratadas como ordens parciais estritas[20] (SPO, do inglês *Strict Partial Orders*). As preferências em SPO podem ser interpretadas de uma maneira intuitiva como "Eu gosto mais de A do que gosto de B". Formalmente, uma preferência P em um conjunto de atributos A é definida como P := (A, <p), em que <p $\subseteq$ dom(A) x dom(A) é uma ordem parcial estrita. A preferência 'x <p y' é interpretada como "Eu prefiro y a x".

A extensão da linguagem SQL foi realizada por meio da utilização de novas cláusulas específicas para tratamento de preferências: AROUND e BETWEEN (de aproximação), LOWEST e HIGHEST (minimização e maximização) e POS e NEG (preferido e não desejado). Dois outros operadores são responsáveis por composições de preferências sobre as tuplas retornadas: CASCADE (ordenação de relevância) e AND (atributos com relevância igual). A implementação do modelo foi

[20] Uma ordem parcial estrita trata de uma estrutura relacional na qual pelo menos um par não admite comparação.

realizada usando um *middleware* escrito na linguagem Java[21] conforme arquitetura apresentada na Figura 3.13.

Figura 3.13 – A arquitetura do sistema Preference SQL



Fonte: Kießling et al. (2011)

Foi desenvolvida também uma extensão da camada de uma JDBC[22] (*Java Database Connectivity*) padrão, criando o pacote Preference SQL/JDBC, que permite que aplicações clientes possam submeter consultas Preference SQL por intermédio de clientes SQL tradicionais. O *middleware* Preference SQL analisa a consulta e executa uma "otimização" baseada nas preferências do usuário. A consulta, posteriormente, é retraduzida para SQL padrão (SQL-92) e transferida via JDBC tradicional para o SGBD da aplicação cliente. De posse do resultado retornado, os operadores de preferências e os novos métodos algébricos usados no modelo são aplicados no *middleware* Preference SQL, que engloba os algoritmos Hexagon (PREISINGER; KIEßLING, 2007) e LESS (GODFREY et al., 2005) para consultas padrão Pareto e *Skyline*[23] (BÖRZSÖNYI et al., 2001).

Acompanhe um exemplo de uma consulta Preference SQL:

```
SELECT *
FROM carros_usados
PREFERRING km AROUND 20000
AND POS (categoria, {'passeio'});
```

A cláusula `PREFERRING` especifica duas preferências combinadas pela cláusula AND, uma indica quilometragem rodada (km) em torno de 20 mil e outra, a categoria, igual a passeio. Essa combinação será resolvida por uma composição do tipo PARETO que avaliará por meio de uma função de *ranking* quais as tuplas preferidas, dentre as tuplas da relação lida.

---

[21] Java, disponível em: http://www.java.com/. Acesso em: 15 dez. 2014.
[22] JDBC é conjunto de classes e interfaces escritas em linguagem Java que fazem o envio de instruções SQL para bancos de dados relacionais.
[23] Consultas *Skyline* resultam em ordenações nas quais não há dominância entre os argumentos da consulta. Por exemplo, "Ache os hotéis com preços baixos e próximos ao centro da cidade".

Uma composição Pareto $>P1,2 = >P1 \otimes >P2$, é definida por:

$(x,y) >P1,2 (x', y')$ *iff* $x \geq P1\ x' \wedge y \geq P2\ y' \wedge (x >P1 x' \vee y >P2 y')$

A composição Pareto de duas preferências P1 e P2 é formada pelas tuplas (x,y) preferidas em comparação a tuplas (x',y') se e somente se na tupla (x,y) x é preferido ou igual a x' para a preferência P1 e y é preferido ou igual a y' para a preferência P2 e x é preferido a x' na preferência P1 ou y é preferido a y' na preferência P2.

Suponha que a relação da Figura 3.14 seja consultada pelo comando SELECT apresentado, uma função de *ranking* seria avaliada para cada atributo da seguinte forma:

- atributo `categoria`: *level*(c) = 1 se valor(c) = 'passeio', senão *level*(c)= 2;
- atributo `km`: *distance*(km) = | valor(a) – valor ótimo |, no qual o valor ótimo de `km` é 20000 e o objetivo é minimizar a distância para o valor ótimo.

Figura 3.14 – Exemplo de aplicação de *ranking* em consulta Pareto

	carro	**categoria**	**Km**	***level(c)***	***distance*(km)**
t1	golf	passeio	20000	1	0
t2	puma	conversível	22500	2	2500
t3	siena	passeio	18000	1	2000
t4	towner	minivan	18000	2	2000
t5	kombi	minivan	26000	2	4000

Fonte: O autor.

O Preference SQL adota o critério BMO (*Best Matches Only*) no qual apenas as melhores tuplas são selecionadas, o que fará que as tuplas selecionadas sejam *t1* e *t3*. Observa-se que a execução do comando exibido é realizada apenas após ser feita uma tradução para comandos SQL-92 padrão que gera uma visão intermediária e só após isso são realizadas as operações de preferências na camada do *middleware* Preference SQL (KIEßLING; KÖSTLER, 2002).

3.2.1.3 CPref-SQL

CPref-SQL (AMO; PEREIRA, 2011; AMO; RIBEIRO, 2009; PEREIRA, 2011) é uma extensão da linguagem SQL, que incorpora restrições nas consultas por meio

da cláusula de preferências condicionais `ACORDING TO PREFERENCES`. As restrições são criadas por meio do comando `CREATE PREFERENCES`.

Por exemplo (PEREIRA, 2011), suponha um banco de dados de uma agência de viagens, no qual uma há uma relação com os seguintes atributos: `destino (D)`, `roteiro (R)`, `preço (P)` e `duração (Du)`. Considere as seguintes preferências de um usuário hipotético:

- “Prefiro viagens com preço inferior a R$3.000,00, independente de outros atributos”;
- “Se o preço for maior do que R$ 3.000,00, então eu prefiro ir à praia a ir para roteiros ecológicos, independente do destino e duração da viagem”.
- “Se o preço for maior do que R$3.000,00 e for uma opção de roteiro ecológico, então prefiro que seja uma duração de até cinco dias, para qualquer destino”.

Para os casos citados, a preferência myprefs é criada no banco de dados (relacional estendido), com o seguinte comando:

```
CREATE PREFERENCES myprefs
FROM viagens AS
P < 3000 > P ≥ 3000 [D, R, Du] AND
IF P > 3000 THEN R = ‘praia’ > R = ‘ecologico’ [D, Du] AND
IF R = ‘ecologico’ AND P > 3000 THEN Du ≤5 > Du > 5 [D];
```

Uma consulta que tem como objetivo saber os quatro pacotes que mais se adequam às preferências, desde que não sejam opções com preço acima de R$ 5.000,00 corresponde a:

```
SELECT * FROM viagens
WHERE P ≤ 5000 ACCORDING TO PREFERENCES (myprefs, 4);
```

Uma versão do CPref-SQL publicada em Amo e Pereira (2011) passou a também considerar o contexto (tanto quanto as preferências) no processo de consulta. Os contextos, neste caso, são associados a conjuntos de regras de preferências contextualizadas e consideradas para seleção de tuplas por meio de um novo operador de *ranking* apropriado para comparações de tuplas em contextos diferentes.

No exemplo a seguir, a criação da preferência `MyPrefs2` diz que os atributos de contexto relevantes são acompanhante (A) e temporada (T), com domínios dom (A) = {família, amigos, sozinho} e dom (T) = {carnaval, natal, verão}. Para uma viagem de carnaval acompanhado pelos amigos as preferências são as seguintes:

- Em geral, prefiro itinerários de praia a cruzeiros;
- Entre duas opções de viagens com o mesmo preço, eu prefiro praia do que itinerários urbanos;
- Entre duas opções de viagem com o mesmo roteiro, eu prefiro a que tenha custo inferior à R$ 2.300.

No entanto, se eu viajar acompanhado por minha família durante o Natal, as minhas preferências são as seguintes:

- Eu prefiro viajar em um cruzeiro com roteiro de praia;
- Para itinerários urbanos, prefiro opções que custam menos de US$ 1500.

O comando de criação de `MyPrefs2` é então configurado da seguinte forma:

```
CREATE PREFERENCES MyPrefs2
FROM viagens AS
Context (A = amigos, T = carnaval)
         R=praia ≻ I=cruzeiro [D,P,Du] AND
         R=praia ≻ I=urbano [D,Du] AND
         P < 2300) ≻ (P ≥ 2300) [D,Du],
Context (A = familia, T = natal)
         R=cruzeiro ≻ I=praia [D,P,Du] AND
         IF I=urbano THEN (P < 1500) ≻ (P ≥ 1500) [D,Du]
```

O comando de leitura correspondente é configurado da seguinte forma:

```
SELECT * FROM Travels
WHERE R <> ecological
ACCORDING TO PREFERENCES 4, MyPrefs2;
```

Para realizar efetivamente a consulta no SGBD, foram desenvolvidos os operadores Select-Best e SelectK-best que selecionam as tuplas preferidas (ou um número determinado k de tuplas preferidas) de uma dada relação, de acordo com a preferência especificada. Versões *built-in* e *on-top* foram desenvolvidas como extensões do SGBD PostgreSQL que armazenam as preferências e realizam consultas com os operadores desenvolvidos. A ferramenta CPrefSQL-Tool (DIAS, 2012) foi criada para permitir que usuários, que conheçam ou não a linguagem CPref-SQL, possam gerenciar e usar suas preferências para realizar consultas.

3.2.1.4 Modelo Generalizado de Preferências

Koutrika e Ioannidis (2004, 2005) propuseram o Modelo Generalizado de Preferências (*Generalized Preference Model*) formado por um modelo de preferência, algoritmos de personalização de consulta em SQL e funções de

classificação. A análise realizada foca a parte de personalização da consulta, em virtude de usar uma técnica de reescrita de consultas bastante expressiva para preferências, a *Simply Personalized Answers* (SPA). O objetivo da SPA é integrar as K principais preferências de um perfil de usuário em uma consulta inicial e construir uma nova consulta. A nova consulta é formulada como a união de um conjunto de subconsultas, cada uma mapeada de uma ou mais das *K* preferências selecionadas. Cada sub-consulta é construída a partir da extensão da consulta inicial.

Da mesma forma que os autores, este trabalho faz uso do exemplo dado para detalhar as operações envolvidas na técnica. Considere o exemplo no qual o usuário 'Al' submete uma consulta simples sobre filmes:

```
SELECT title FROM movie;
```

Suponha que as seguintes preferências foram selecionadas pelo sistema para inclusão na consulta (ver referências, para maiores detalhes de como é montado o perfil de um usuário e graus de interesse). O perfil do usuário informa que Al é extremamente interessado no diretor de um filme (P1), gosta muito do diretor W. Allen (P2), não gosta de filmes lançados antes de 1980 (P3) e fica feliz se o gênero do filme não for musical (P4).

	Preferência	Interesse	Tipo
(P1)	MOVIE.mid=DIRECTED.mid and	(1,0)	
	DIRECTED.did=DIRECTOR.did and	(0,9)	
(P2)	DIRECTOR.name='W. Allen'	(0,8)	presence
(P3)	MOVIE.year<1980	(0.8)	absence 1-1
(P4)	MOVIE.mid=GENRE.mid and	(1,0)	
	GENRE.genre='musical'	(0,7)	absence 1-n

A técnica é baseada na formação de sub-consultas que dependem do tipo de preferência. A preferência deve ser satisfeita por presença (*presence*) ou ausência (*absense*). Além disso, há uma distinção entre preferências de ausência 1-1 e 1-n (que informam o tipo de relacionamento entre as entidades envolvidas). De acordo com os graus de interesse informados no perfil do usuário, para cada subconsulta, são calculados os graus de interesse relativos aos atributos e valores instanciados e o tipo de preferência. Por exemplo, a subconsulta Q1 formatada a seguir, tem grau de interesse = 1 x 0,9 x 0,8 = 0,72, formados a partir das preferências P1 e P2.

Preferências de presença:

```
Q1:   select title, 0.72 degree
        from MOVIE M, DIRECTED D, DIRECTOR DI
       where M.mid=D.mid
         and D.did=DI.did and DI.name='W. Allen'
```

Preferências de ausência 1-1: são mapeadas para sub-consultas de forma semelhante às de presença. A única diferença representa a mudança do operador da condição.

```
Q2:   select title, 0.8 degree
             from MOVIE M
             where M.year >= 1980
```

Preferências de ausência 1-n:

```
Q3:   select title, 0.7 degree
        from MOVIE M
       where M.mid not in (select M.mid
        from MOVIES M, GENRE G
       where M.mid=G.mid and G.genre='musical'
```

Os resultados são obtidos realizando a união dos resultados parciais das sub-consultas, agrupados pelos atributos projetados na consulta inicial, e excluindo todos os grupos com menos de L linhas (no exemplo, L=2), ou seja, estejam satisfazendo pelo menos duas preferências. Os resultados são classificados com base na combinação de preferências satisfeitas.

```
select title,r(degree)
from (select title, 0.72 degree
        from MOVIE M, DIRECTED D, DIRECTOR DI
       where M.mid=D.mid and D.did=DI.did and DI.name='W. Allen'
Union All
      (select title, 0 degree
        from MOVIE M
       where M.year>=1980)
Union All
      (select title, 0.7 degree
        from MOVIE M
       where M.mid not in (select M.mid
        from MOVIES M, GENRE G
       where M.mid=G.mid and G.genre='musical'))
group by title
having count(*) >= 2
order by r(degree);
```

O trabalho abordado usa uma técnica de reescrita de consultas como forma de expressar as preferências do usuário. Na seção seguinte, serão detalhadas outras propostas que usam técnicas de reescrita como estratégia de consultas sensíveis ao contexto.

3.2.2. Trabalhos Baseados em Reescrita de Consultas SQL

Conforme visto na Seção 2.5, a reescrita de consultas é um termo genérico usado para definir processos que buscam melhoria de resultados, melhoria de desempenho ou integração de dados. Neste trabalho, a análise realizada focou na melhoria de resultados em consultas SQL, nas quais são empregadas técnicas correlatas de expansão, relaxamento e substituição.

Nesta seção são detalhados três trabalhos representativos no que concerne às técnicas utilizadas. A análise realizada independe da fonte de elementos que subsidiarão o raciocínio por trás da reescrita (objeto da substituição, supressão ou adição de termos), por exemplo, o uso de ontologias de domínio, mas direciona a atenção às técnicas empregadas na reescrita da consulta SQL.

3.2.2.1 Sistema FOQuE para Expansão Semântica de Consultas

O Sistema FOQuE (*Fuzzy Ontology-based Query Expansion*) (YAGUINUMA, 2007; YAGUINUMA et al., 2007) realiza a reescrita de consultas por meio de expansão com termos homônimos.

É realizada a associação de uma ontologia[24] ao esquema do banco de dados que se deseja consultar. Essa associação pode ser feita extraindo uma ontologia que se assemelha à estrutura do banco e contém definições que estendem a semântica dos dados ou fazendo *reuso* de ontologia que se assemelhe com as características do banco de dados utilizado. Também é necessário realizar o mapeamento da ontologia para o BD, ou seja, descrever as associações entre os elementos da ontologia e do banco de dados para possibilitar a reformulação de consultas contendo os conceitos expandidos.

No sistema FOQuE, esse mapeamento é composto por regras de formato *antecedente* ⇒ *consequente*, nas quais o *antecedente* contém elementos da ontologia difusa e o *consequente* elementos do banco de dados. Um exemplo de mapeamento é a regra *Produto(x), label(x, y)* ⇒ *Produto(x), nome(x, y)*, que

[24] Uma ontologia define um conjunto de primitivas de representação com as quais se pode modelar um domínio do conhecimento ou de discurso. Fornece, por exemplo, a possibilidade de sinônimos, especialização e generalização de termos e seus relacionamentos. Isso permite também que informações não armazenadas possam ser deduzidas.

descreve quais instâncias da classe `Produto` com um determinado rótulo (*label*) correspondem a dados da tabela Produto cujo atributo `nome` possui o mesmo valor que *label*. A partir dessas regras, é possível especificar quais tabelas e atributos correspondem a classes ou relacionamentos da ontologia e quais dados estão associados a conceitos homônimos.

Como exemplo, considere uma consulta SQL pelo produto ventilador, teríamos:

```
SELECT produto.preco
  FROM produto
 WHERE produto.nome = ´ventilador´;
```

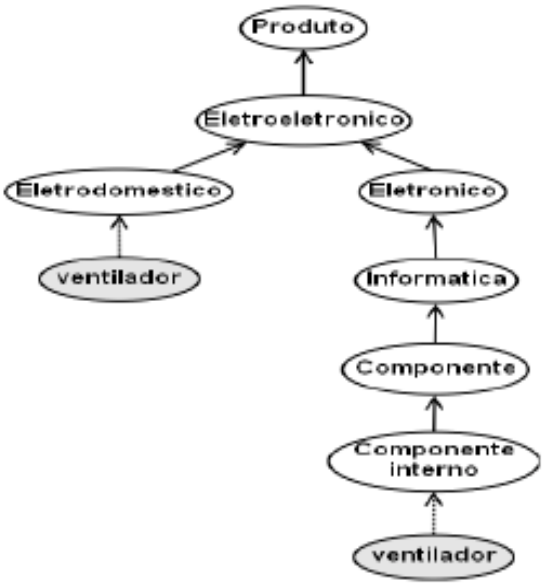
Depois da consulta ser submetida ao sistema, este analisará a consulta e o conjunto de mapeamentos para a ontologia no intuito de verificar quais expansões semânticas podem ser aplicadas ao termo `ventilador`. Pela Figura 3.15, verifica-se que há dois conceitos homônimos ao termo `ventilador`: eletrodoméstico e componente de informática. Para tratar essa ambiguidade, é utilizado algum atributo do banco que remova a ambiguidade dos dados como, no exemplo, o atributo tipo da tabela produto.

Por exemplo, para o tipo *eletrônico* existe um mapeamento entre o banco de dados e a ontologia referindo-se ao produto ventilador como *eletrodoméstico* (tipo = eletrodoméstico), de forma análoga para ventilador como componente (tipo = componente). Os contextos identificados são exibidos para o usuário, que decide qual deles é mais adequado para sua consulta.

Caso o usuário queira recuperar dados de ventiladores que sejam eletrodomésticos, a condição da consulta original deve ser alterada em função da regra de mapeamento como mostrado a seguir. A nova consulta a ser submetida ao banco de dados seria:

```
SELECT produto.preco
  FROM produto
 WHERE ((produto. nome = ´ventilador´)
   AND (produto.tipo = ´eletrodomestico´));
```

Figura 3.15 – Ontologia para o termo 'ventilador'



Fonte: Yaguinuma (2007).

O Sistema FOQuE realiza os seguintes tipos de expansão de consultas:

- *Homônimos* - identifica e apresenta ao usuário os diferentes contextos presentes em consultas com ambiguidades. O termo fornecido pelo usuário é usado para reescrever a consulta evitando resultados irrelevantes. Conforme visto no exemplo, utiliza o operador AND na cláusula WHERE para remover a ambiguidade.
- *Classes* - é realizada quando a consulta contém termos correspondentes a classes na ontologia. A consulta original é expandida para adicionar as instâncias da classe e das respectivas subclasses. Utiliza operador OR para cada instância adicionada e respeita o parâmetro de pertinência de classes estabelecido na ontologia.
- *Similaridade* - possibilita consultar conceitos semanticamente relevantes, considerando o grau de similaridade. O sistema modifica a consulta ao adicionar conceitos similares ao conceito especificado pelo usuário, desde que satisfaçam ao parâmetro de similaridade mínima. Os graus de similaridade são especificados na ontologia de forma manual por especialistas no domínio. Também utiliza operadores OR para expandir a consulta.
- *Proximidade todo-parte* - parte do princípio que conceitos constituídos por um conjunto aproximado de partes ou componentes em comum podem ser

semanticamente próximos. Assim o sistema analisa relacionamentos todo-parte para obter conceitos próximos aos requisitos do usuário e acrescenta condições OR para cada conceito cujo grau de proximidade seja maior ou igual ao parâmetro de proximidade mínima.

- *Transitividade* - analisa relacionamentos transitivos da ontologia que não são explícitos no banco de dados. Altera a consulta original pela adição de tabelas de relacionamento e alteração de condições de chaves estrangeiras e cláusulas UNION.

Além da característica da interação com o usuário para realização das expansões, as limitações reportadas deste trabalho são não considerar consultas com funções de agregação e consultas aninhadas.

3.2.2.2 Processamento Semântico de Consultas

Da mesma forma que no trabalho anterior, o trabalho de Vilar (2008) também se utiliza de ontologias de domínio e propõe um serviço de expansão de consultas cujo objetivo é obter resultados mais expressivos a partir da reformulação de consultas aplicadas a bases de dados que envolvam dados sobre biodiversidade. Um mecanismo de expansão de consultas pré-processa uma consulta de usuário, desambigua termos e agrega informações provenientes de ontologias, para aproximar o resultado da consulta à intenção do usuário.

As expansões que podem ser realizadas nas consultas consistem em:

- *Especializações* - utilizar subclasses de uma classe, preservando a semântica da consulta;
- *Generalizações* - adicionar superclasses de uma classe, como forma de obter resultados próximos ao original;
- *Instanciação* - incluir instâncias de uma classe, mantendo o significado da consulta; e
- *Equivalência* - utilizar classes relacionadas por axiomas de equivalências, sem alterar a semântica da consulta.

Como exemplo da aplicação com uso de subclasses (hipônimo), considere uma consulta que deseja obter dados sobre insetos da ordem *lepidoptera* que tenham antenas clavadas. Em SQL, a consulta pode ser representada em uma consulta:

```
SELECT * FROM colecao
 WHERE tipo = 'inseto'
   AND ordem = 'lepidoptera' AND antenas = 'clavada';
```

Considere a tabela do BD conforme ilustrado na Figura 3.16. Percebe-se que nem todos os atributos correspondem aos critérios da consulta, como `ordem`. Então é preciso encontrar termos alternativos que satisfaçam à consulta original.

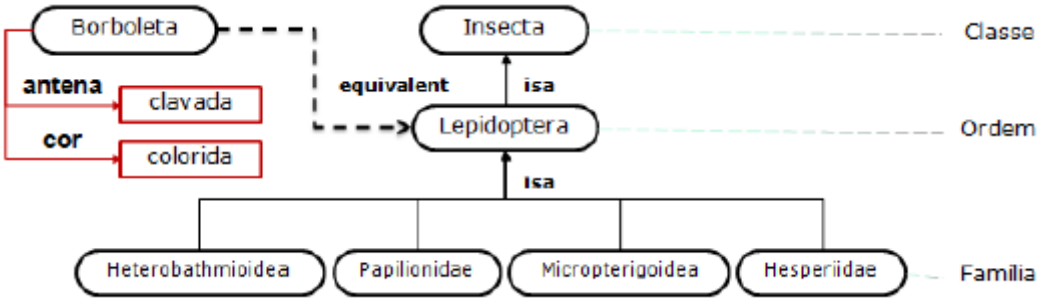
Figura 3.16 – Tabela do BD sem correspondência com a consulta

Id	Tipo	Família	Antena	Coloração
1	inseto	hesperiidae	clavada	colorida
2	inseto	lepidotrichidae	filiforme	monótona
3	inseto	cerambycidae	geníciulo-clavada	colorida

Fonte: Vilar (2008).

Nesse caso, pode-se obter uma consulta alternativa a partir de uma ontologia como a representada na Figura 3.17. Por meio da relação de herança entre os conceitos, é possível reconhecer que *heterobathmioidea*, *papilionidae*, *micropterigoidea* e *hesperiidae* são sub-conceitos de *lepidoptera*.

Figura 3.17 – Ontologia que descreve parte da classe *insecta*



Fonte: Vilar (2008).

A consulta então pode ser reformulada da seguinte forma:

```
SELECT * FROM colecao
 WHERE tipo='inseto' AND familia in ('hesperiidae',
      'papilionidae', 'micropterigoidea','heterobathmioidea')
   AND antenas = 'clavada';
```

O resultado é a utilização de um conjunto de famílias, pertencentes ao campo ordem especificado na consulta, do qual participam os mesmos indivíduos que estariam presentes na ordem lepidoptera. Tais indivíduos, portanto, satisfazem ao critério especificado na consulta.

O exemplo utilizou as informações referentes às subclasses para chegar a definições diferentes de um mesmo conceito, mas as ontologias têm outros conjuntos de recursos que podem ser empregados, tais como: Conceito (Idêntico, Equivalente, Subclasse, Superclasse, Instancia, Intersecção e União); Relacionamento/Propriedade (Idêntico, Equivalente, Transitivo, Simétrico e Parte/todo) e Axiomas.

3.2.2.3 Refinamento Interativo de Consultas

Mishra e Koudas (2009) propuseram uma forma de refinar uma consulta com o objetivo de manipular a quantidade de dados retornados, por meio da alteração de predicados numéricos ou categóricos e fazendo uso de um modelo de refinamento que utiliza as preferências do usuário. São utilizadas técnicas de expansão e contração de consultas.

Dada uma consulta Q e uma meta de uma quantidade k de respostas retornadas, deve-se gerar uma nova consulta Q' refinando os predicados de seleção de Q. Um *framework* denominado *Stretch 'n' Shrink* (SNS) foi desenvolvido para tratar consultas que retornem poucas respostas fruto de descrição muito restritiva e consultas muito amplas e que retornem respostas que não sejam relevantes para o usuário.

Os predicados numéricos admitem seleções com intervalo (e.g. <, ≤, >, ≥) ou igualdade (=). Já os predicados categóricos utilizam apenas o operador de igualdade. Os predicados categóricos são estruturados em uma classe de hierarquias que podem ser especificadas pelo usuário ou pelo projetista do esquema do banco de dados. Exemplos de hierarquias são: {país, estado, cidade, rua, nº} ou {arte, música, musica clássica} nos quais cada atributo pertence a certo nível na hierarquia. (e.g. país está no nível 1, e rua está no nível 4). Transformações de relaxamento aumentam o número de respostas retornadas; já as transformações de contração diminuem a quantidade de respostas.

Para a geração de uma nova consulta Q’ a partir de Q, o *framework* SNS identifica se a consulta original Q deve ser relaxada ou contraída e calcula os relaxamentos e as contrações necessárias:

- Para predicados numéricos, é calculado o número de refinamentos necessários e estimada a quantidade de respostas retornadas pela consulta

resultante. Por exemplo, suponha que uma consulta com predicados ano < 1960, idade < 25 e salário < 3000 retorna poucas respostas. O SNS calcula relaxamentos máximos 1980 para ano, 40 para idade e 7000 para salário. As etapas de refinamento de predicados indicariam relaxamento da idade para idade < 30, ano com intervalo entre 1960 e 1975 e salário entre 3000 e 6400, de acordo com as hierarquias projetadas.

- Para predicados categóricos, semelhante ao que ocorre com os predicados numéricos, deve-se calcular o relaxamento máximo. É utilizada uma estrutura de indexação e um esquema de navegação. É realizada uma indexação do esquema de hierarquias, que apoia o relaxamento ou contração de um predicado hierárquico. Em seguida, o esquema de hierarquias dos predicados categóricos é percorrido. Iniciando com os predicados da consulta original e trabalhando em ciclos, o usuário é solicitado a selecionar um predicado hierárquico que pode ampliar as respostas (no caso de relaxamento) ou reduzi-las (contrações). O processamento em ciclos se repete até que a consulta refinada exceda a quantidade máxima (para relaxamentos) ou se reduza para aquém da quantidade mínima de respostas (para contrações).

Ao final, as duas técnicas são usadas combinadas para a geração da nova consulta Q'. Considere os exemplos: i) Uma consulta Q tem os seguintes predicados ano > 1990 e custo < 5000, e poucas respostas são retornadas. Alterando-se a restrição ano para ano > 1980 podem-se ter retornadas mais respostas. ii) Uma consulta com os predicados país = 'USA' e estado = 'Califórnia' que retorna muitas respostas, pode ser modificada para incluir novos predicados, tais como cidade = 'Santa Cruz' e zip = '94112', de acordo com a hierarquia utilizada. Os novos atributos reduzirão a quantidade das respostas.

Resumos comparativos dos trabalhos centrados na consulta contextual

Na primeira parte desta seção foram apresentados alguns trabalhos representativos cuja abordagem é centrada em preferências. O objetivo principal desses trabalhos é prover uma alternativa para que as consultas realizadas possam expressar preferências do usuário.

O Quadro 3.4 destina-se a apresentar uma comparação entre os trabalhos descritos nesta seção e mais dois trabalhos da Seção 3.1, cujos focos dos modelos de dados são centrados em preferências (*Contextual Preference Model* e CIM). Os critérios de comparação utilizados são:

- Abordagem - identifica a abordagem utilizada para expressar preferências em consultas (vide Seção 3.2);
- Implementação - indica forma de implementação do modelo ou estratégia de preferência (vide Seção 3.2);
- Recursos usados - resumo dos principais recursos utilizados para a consulta baseada em preferências (e.g. hierarquias, base de conhecimento, extensão do SQL, reescrita) no trabalho; e
- Contexto: informa se o modelo ou estratégia de preferências considera o contexto.

Observa-se que o último trabalho abordado, de Koutrika e Ioannidis (2005), foge da classificação limitada às abordagens *qualitativa* ou *quantitativa*, uma vez que a abordagem quantitativa não captura as preferências negativas e preferências para a ausência de valores e a abordagem qualitativa não capta preferências expressas sobre as relações entre entidades (e.g. “eu estou muito interessado em atores de um filme”) e preferências implícitas (e.g. “eu gosto de filmes com duração em torno de 2h”). Esse trabalho foi assinalado como de abordagem quantitativa baseada em atributo, por atribuir índices de preferências a atributos do esquema de dados.

O Quadro 3.5 apresenta um resumo das diferenças e similaridades entre os trabalhos referenciados centrados na reescrita de consultas SQL, descritos na segunda parte desta seção. Da mesma forma que no quadro anterior, o trabalho *Contextual Preference Model* é considerado, em virtude de utilizar técnica de reescrita de consultas. Com o objetivo de exibir uma visão geral desses trabalhos, os seguintes critérios formam o quadro comparativo:

- Abordagem - identifica a linha de pesquisa de conduziu o desenvolvimento do trabalho. Neste caso, se foi motivada pela implementação da consulta baseada em preferências ou para melhoria de resultados;

- Técnicas de reescrita - indica as técnicas de reescrita utilizadas no trabalho;
- Recursos usados - resumo dos principais recursos (e.g. ontologia, estruturas hierárquicas, funções, algoritmos) utilizados no trabalho; e
- Contexto - informa se a reescrita de consultas considera o contexto.

Quadro 3.4 – Abordagem e implementação dos trabalhos centrados em preferências

Autor/Projeto /Ano	Abordagem	Implementação	Recursos usados	Contexto
Stefanidis et al./ *Contextual Preference Model* /2007~2011	Quantitativa	*On-top*	Estruturas de dados hierárquicas. / Classificação de respostas após leitura no BD.	Sim
Bunningen/ CIM/ 2008	Quantitativa	*Built-in*	Base de conhecimento em lógica descritiva mapeada para BD relacional	Contexto interno
Levandoski et.al./ CareDB/2010	Qualitativa	*Built-in*	Tradução de consultas e perfis de preferências	Sim
Kießling et al. / Preference SQL/ 2002~2011	Qualitativa / Quantitativa	Tradução para SQL e posterior classificação de resultados	Extensão do SQL, regras de preferência e reformulação de respostas	Não
De Amo et al. / CPrefSQL/ 2011	Qualitativa	*On-top* ou *Built-in*	Extensão do SQL com preferências armazenadas e operadores de consulta	Sim
Koutrika e Ioannidis / *Genealized Preference Model/ 2004*	Quantitativa baseada em atributo	Perfis de usuários e tradução para SQL	Reescrita de consultas mediante preferências em perfis de usuários	Não

Fonte: O autor.

Quadro 3.5 – Comparativo de trabalhos centrados na reescrita de consultas

Autor/Projeto /Ano	Abordagem	Técnicas de Reescrita	Recursos usados	Contexto
Stefanidis et al./ *Contextual Preference Model* /2007~2011	Preferências contextuais / Estruturas de dados hierárquicas	Relaxamento de consultas e classificação de resultados	Estruturas de dados hierárquicas para fundamentar relaxamentos	Sim
Koutrika e Ioannidis 2004	Preferências contextuais / Reescrita de SQL	Expansão de consultas e classificação de resultados	Perfil de preferências, Funções de classificação	Não
Yaguinuma 2007	Processamento semântico/ Reescrita de SQL	Expansão de consultas SQL	Ontologia de domínio	Não
Vilar 2008	Processamento semântico/ Reescrita de SQL	Expansão de consultas SQL	Ontologia de domínio	Não
Mishra e Koudas 2009	Refinamento semântico/ Reescrita de SQL	Expansão/contração de consultas SQL	Estruturas de dados hierárquicas algoritmos para expansão ou contração de consultas	Não

Fonte: O autor.

3.3. Considerações

Contexto é uma área relativamente nova no campo da Ciência da Computação, mas cuja exploração tem um enorme potencial, fomentado principalmente pelas tecnologias de propagação de informação em massa como a Internet e pelas tecnologias de disseminação de informação de maneira ubíqua. Esse fato representou um grande estímulo para o desenvolvimento de SSC, mas que não se refletiu nos SGBD com a mesma velocidade. Isso em parte é explicado pela maior facilidade de lidar com contexto na camada da aplicação e, também, por ser grande o esforço para que modificações em tecnologias maduras e largamente adotadas, como os SGBD tradicionais, sejam aceitas pelo mercado.

A integração de contexto em SGBD é uma tarefa complexa e que tem desafiado a comunidade científica das áreas de Banco de Dados e Inteligência Artificial. As pesquisas desenvolvidas buscam superar as dificuldades de como representar o contexto, como inferir o contexto a partir de elementos contextuais e como consultar dados de forma contextualizada.

Este capítulo apresentou um estudo específico do estado da arte em termos de modelagem do dado contextual em SGBD e da consulta ao dado contextual. As pesquisas mais representativas e suas principais características foram agrupadas segundo a abordagem utilizada. Em virtude da heterogeneidade dos trabalhos e diversidade de objetivos, optou-se por elencar os objetivos, recursos usados, técnicas e estratégias e, assim, ter um panorama funcional da pesquisa no tema abordado.

4. REESCRITA DE CONSULTAS BASEADA EM CONTEXTO

Neste capítulo é apresentada a Abordagem Texere, concebida para capacitar bancos de dados relacionais para que respondam à consultas de forma sensível ao contexto sob o qual foram realizadas.

Para facilitar a apresentação do trabalho, este capítulo está subdividido em cinco seções que detalham suas diferentes perspectivas. A Seção 4.1 apresenta uma visão geral da abordagem e de sua arquitetura. A Seção 4.2 detalha a Arquitetura Texere e seus componentes. Na Seção 4.3 são apresentados o modelo de contexto utilizado, o mapeamento de elementos contextuais e a inferência do contexto. A Seção 4.4 detalha o processo de reescrita de consultas e o modelo de preferências. Concluindo o capítulo, a Seção 4.5 apresenta uma análise comparativa deste trabalho com outros.

4.1. Visão Geral

Imagine uma aplicação para bibliotecas que oferece sugestões de livros aos seus usuários. Para uma garota de oito anos, sugere apenas livros adequados à sua faixa etária e caso o período escolar seja de férias, exibe livros de lazer antes dos livros educativos. Para um senhor com deficiência visual, são informados, via dispositivo de voz sintetizada, apenas os áudio-livros e os livros em braile disponíveis.

A aplicação do exemplo obteve a informação sobre o contexto dos seus usuários, da aplicação que estava sendo utilizada e dos ambientes físico e computacional. As informações contextuais obtidas foram tratadas e adequadas ao usuário, antes das respostas das sugestões serem apresentadas.

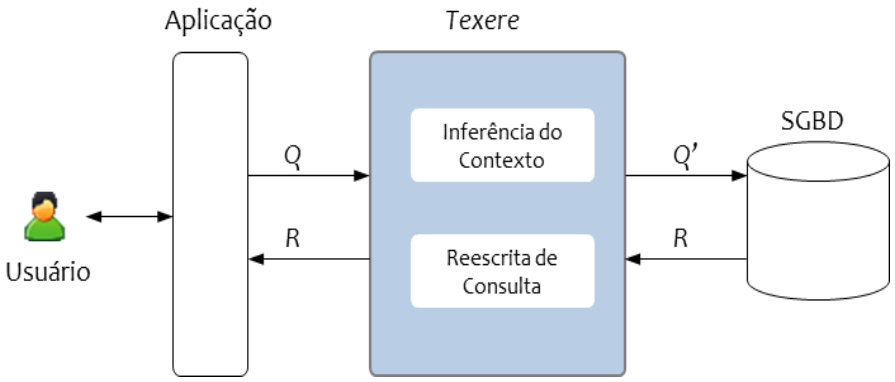
A Abordagem Texere utiliza a informação contextual para que a adequação da consulta seja feita em uma etapa anterior à devolução da resposta à aplicação. Uma premissa adotada é o emprego da abordagem junto a um SGBD relacional convencional.

A arquitetura que implementa a Abordagem Texere se localiza em uma camada entre a aplicação cliente e o SGBD alvo e utiliza elementos contextuais do usuário,

dos ambientes físico e computacional, do dispositivo de consulta e da aplicação para fazer a identificação do contexto sob o qual uma consulta foi realizada e reescrevê-la para que melhor se adeque a este contexto e obtenha respostas mais relevantes para o usuário.

Uma consulta original, ainda não reescrita é denominada consulta *Q*. Conforme o fluxo apresentado na Figura 4.1, as consultas ao SGBD são capturadas por um componente responsável pela inferência do contexto. Cada informação contextual irá disparar diretivas de reescrita do código de *Q*, com o intuito de restringi-la, adaptá-la ou estendê-la gerando uma consulta reescrita *Q'*. A consulta *Q'*, então, é submetida ao SGBD alvo e o seu resultado (R) devolvido à camada da aplicação.

Figura 4.1 – Fluxo da execução de uma consulta na Abordagem Texere



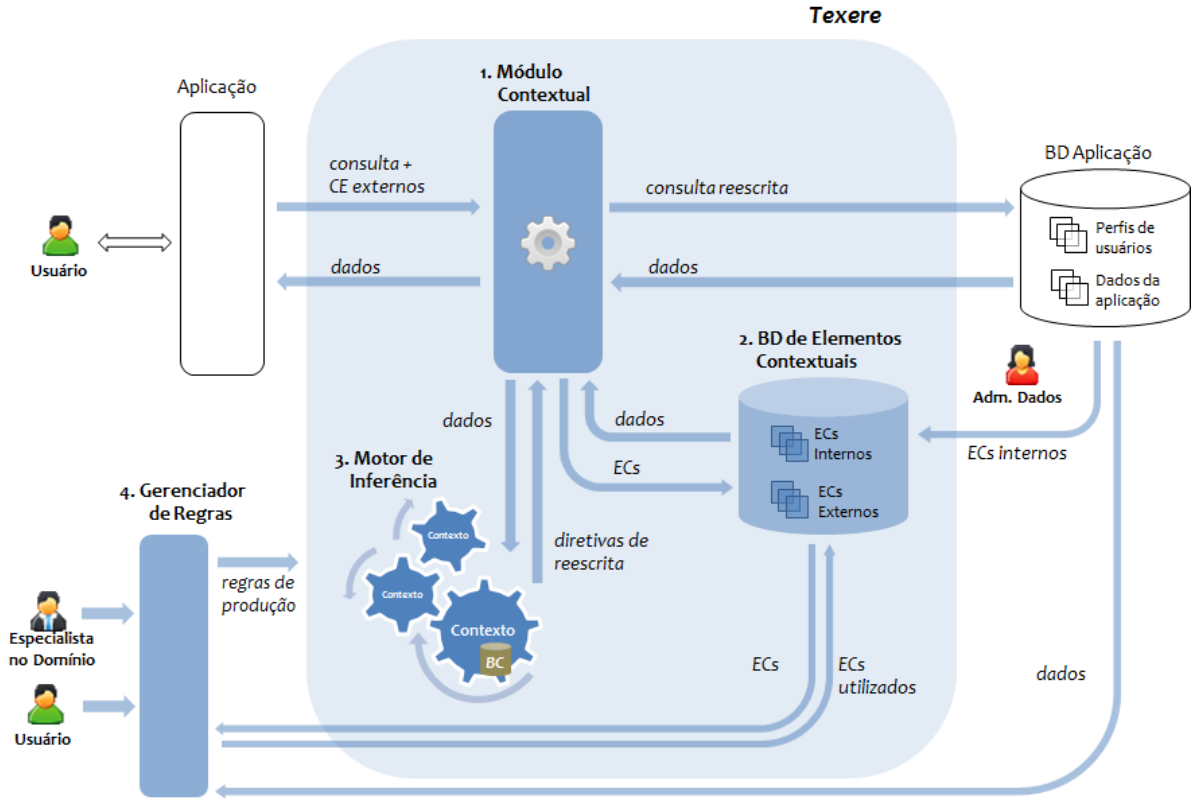
Fonte: O autor.

Uma visão geral da Arquitetura Texere está apresentada na Figura 4.2. Os principais componentes da arquitetura, destacados na figura, são: o **Módulo Contextual** (1), o **Banco de Dados de Elementos Contextuais** (2) e o **Motor de Inferência** (3). O último componente, o **Gerenciador de Regras** (4), é uma interface auxiliar para configuração das regras de produção que guiam o processo de inferência do contexto.

Dado que a Abordagem Texere foi definida para funcionamento com aplicações tradicionais e suas respectivas bases de dados, cada vez que um usuário submete uma consulta via aplicação, o Módulo Contextual (1) captura essa consulta, valida, analisa e a reescreve. No Módulo Contextual estão contidos todos os processos e validações que identificam e lidam com o dado contextual. Esta distribuição em

camadas garante uma separação de funções que mantém flexível a atividade de gerenciamento do dado contextual.

Figura 4.2 – Visão Geral da Arquitetura Texere



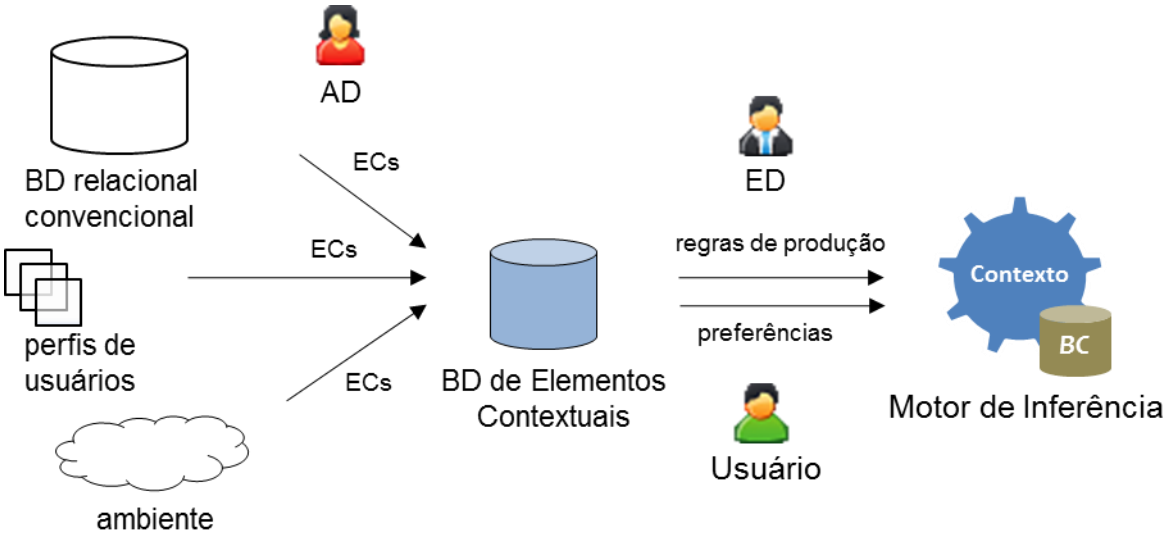
Fonte: O autor.

Conforme apresentado na Seção 2.1.1, os elementos contextuais (EC) compõem um contexto. Esses elementos podem ser encontrados na base de dados da aplicação ou no ambiente (físico ou computacional) no qual a aplicação está inserida. Para que a tarefa de inferência do contexto seja possível, deve haver um mapeamento prévio dos EC e das situações que influenciam na identificação dos diversos contextos admitidos no domínio dessa aplicação. Para esse fim, três atores fundamentais são envolvidos nesse processo de configuração: o Administrador de Dados (AD), o Especialista no Domínio (ED) e o usuário final.

O AD realiza as tarefas de identificação de elementos contextuais e catalogação no Banco de Dados de Elementos Contextuais (Figura 4.3). Usando os elementos contextuais catalogados, o ED cria regras de produção que avaliam os valores instanciados desses elementos e auxiliam na inferência do contexto relacionado a

uma consulta. As regras também atribuem diretivas que irão guiar o processo de reescrita. Da mesma forma, o usuário final também pode definir regras que reflitam suas preferências pessoais. Esses processos estão descritos nas próximas seções.

Figura 4.3 – Usuários e o processo de mapeamento do contexto



Fonte: O autor.

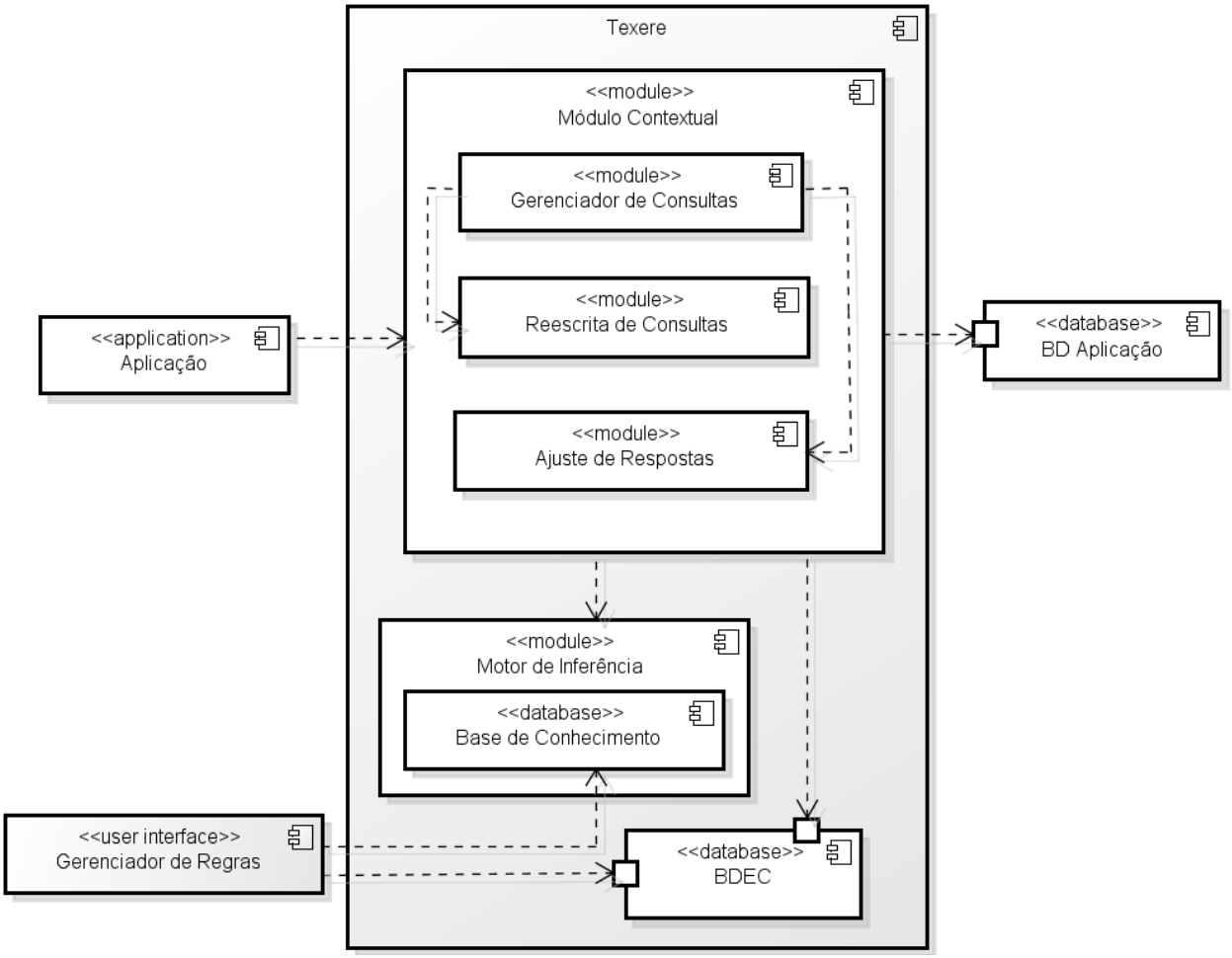
4.2. Arquitetura Texere

Conforme representado na Figura 4.4, em notação UML[25], a Arquitetura Texere é composta pelos seguintes módulos principais: o **Módulo Contextual**, o **Banco de Dados de Elementos Contextuais** (BDEC) e o **Motor de Inferência**. Estes três componentes interagem com aplicações tradicionais (Aplicação) e suas respectivas bases de dados (BD Aplicação). O componente **Gerenciador de Regras** é uma ferramenta adicional para configuração de regras para o Motor de Inferência.

As subseções seguintes apresentarão o funcionamento de cada um desses componentes e como atuam os usuários envolvidos nos processos de configuração e utilização da arquitetura.

[25] UML, Unified Modeling Language. Disponível em: www.uml.org. Acesso em: 15 dez. 2014.

Figura 4.4 – Diagrama de componentes da Arquitetura Texere



Fonte: O autor.

4.2.1. **Módulo Contextual**

O Módulo Contextual é o componente principal da Arquitetura Texere. Ele é o responsável por receber as consultas da aplicação e executar todas as tarefas de reescrita, de acordo com o contexto identificado e, posteriormente, submeter a consulta reescrita ao banco de dados da aplicação, reestruturar a resposta caso necessário, e retornar os resultados para a aplicação, conforme apresentado na Figura 4.4.

Este componente foi concebido em subcomponentes distintos:

- O componente **Gerenciador de Consultas** é responsável por obter e instanciar os elementos contextuais associados à consulta emitida por uma aplicação cliente. Cada consulta recebe um número identificador.

- O componente de **Reescrita de Consultas** é responsável por todo o processamento necessário para reescrever a consulta original (Q). O processo segue diretivas para reescrita da consulta que são informadas após o Motor de Inferência processar todas as regras de produção e identificar o que é necessário ser modificado na consulta original para melhor moldá-la ao contexto sob a qual foi realizada. Essas modificações adicionam, suprimem, restringem ou combinam cláusulas e argumentos ao código original da consulta.
- O componente de **Ajuste de Respostas** executa mudanças adicionais sobre o conjunto de resultados de uma consulta (original ou reescrita) que não tenham suporte no SGBD alvo, tais como classificações baseadas em *soft constraints*[26]. Tipicamente essas operações são realizadas por meio de algoritmos que ordenam, classificam e filtram o conjunto de resultados da consulta emitida, de acordo com preferências do usuário ou restrições da aplicação.

4.2.2. **Banco de Dados de Elementos Contextuais**

O Banco de Dados de Elementos Contextuais (BDEC) é um banco de dados específico para armazenar metadados sobre os elementos contextuais identificados como tendo uma interpretação válida e relevante para o domínio da aplicação. Por exemplo, se o contexto de dispositivos móveis é importante, uma entidade contextual *dispositivo* (referente ao dispositivo pelo qual a consulta foi submetida) deve ser registrado junto com seus elementos contextuais (e.g. nome, origem, tipo e características) e relacionamentos, que formam os seus metadados.

Os metadados referentes aos elementos contextuais serão utilizados pelo Módulo Contextual para encontrar a localização e ligações entre as entidades contextuais utilizadas nos processos de reescrita de consulta. Em outras palavras, tais metadados tornam possível descobrir onde e como uma entidade contextual está armazenada, e como se relaciona com outras entidades contextuais. Como visto, os elementos contextuais também serão usados pelo especialista no domínio e pelo usuário final para criar regras de produção no Motor de Inferência.

---

[26] Restrições que não são declaradas na cláusula SQL WHERE e representam preferências do usuário ou restrições do tipo *top-k.*

O banco de elementos contextuais deve ser implementado de forma que reflita o Modelo de Contexto Texere, detalhado na Seção 4.3.

4.2.3. **Motor de Inferência**

Motores de inferência são programas que oferecem respostas mediante perguntas pré-formatadas, utilizando uma base de conhecimento. Na Arquitetura Texere, a base de conhecimento utilizada armazena o conhecimento especializado do domínio no formato de regras de produção (NEWELL 1973).

Todos os elementos contextuais mapeados no BD de Elementos Contextuais podem ser usados para criar regras. Usando uma interface apropriada (o módulo Gerenciador de Regras), o especialista no domínio pode criar ou modificar as regras em tempo real, sem a necessidade de recarregar ou alterar as configurações de qualquer outro módulo. Da mesma forma, o usuário final da aplicação também pode definir regras para suas preferências pessoais.

O Motor de Inferência é chamado pelo Módulo Contextual que lhe repassa todos os elementos contextuais relevantes instanciados. Usando os valores recebidos, o Motor de Inferência dispara a execução das regras correspondentes. A saída do Motor de Inferência é um conjunto de diretivas de reescrita de consulta (definidas na Seção 4.4.1) que é retornado para o Módulo Contextual.

4.2.4. **Gerenciador de Regras**

A fim de tornar o processo de criação de regras mais fácil e menos suscetível a erros, uma interface apropriada está prevista. O ED pode utilizar o módulo Gerenciador de Regras para selecionar elementos contextuais catalogados e criar regras de produção.

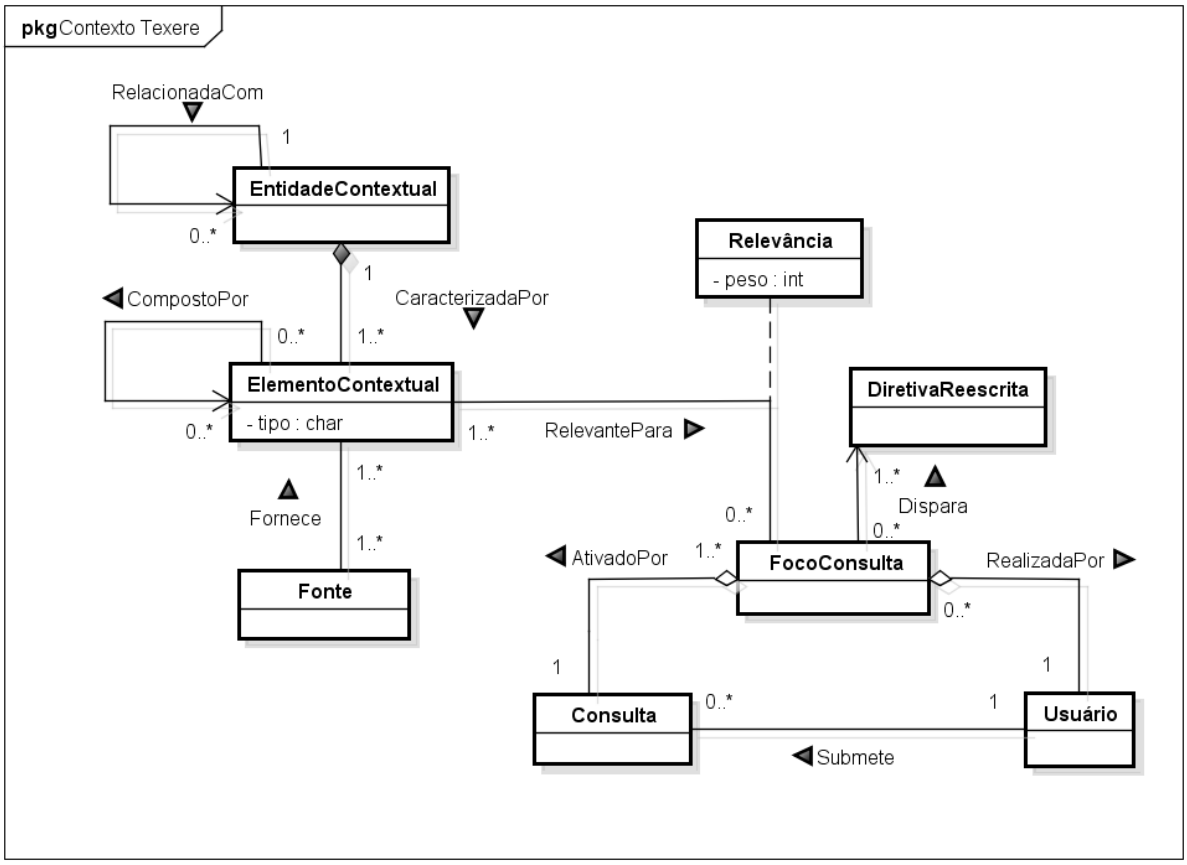
Esta interface também envia mensagens de alerta ao ED e ao AD, caso uma estrutura ou elemento contextual do BD da aplicação tenha sido modificada e causado inconsistência em alguma regra existente, já que regras identificadas como inconsistentes devem ser desprezadas para fins de reescrita da consulta.

4.3. Modelo de Contexto Adotado e Sua Extensão

Para que um sistema tenha a capacidade de lidar com a informação de forma contextualizada é necessário um modelo de representação e um modelo comportamental do contexto, ou seja, uma representação formal e uma especificação do comportamento (reativo ou proativo) do sistema em função do contexto.

O modelo adotado para representação do contexto na Abordagem Texere segue o metamodelo de contexto do *framework* CEManTIKA (VIEIRA, 2008), descrito no Capítulo 2. Instanciando-se o metamodelo de Vieira para a Abordagem Texere, tem-se um modelo de contexto conforme o diagrama de classes, em notação UML[27], apresentado na Figura 4.5.

Figura 4.5 – Modelo de contexto instanciado e estendido



Fonte: O autor.

[27] UML, Unified Modeling Language. Disponível em: www.uml.org. Acesso em: 15 dez. 2014.

O modelo de contexto instanciado para este trabalho estabelece que o *agente* é o *usuário* que submete a consulta ao banco de dados, a *tarefa* é a *consulta* e o *foco* (*FocoConsulta*) é a submissão da consulta pelo usuário. Cada *elemento contextual* (ElementoContextual) relevante para um determinado *foco* tem sua própria *relevância* (Relevância). Cada foco de consulta pode disparar a execução de suas próprias *regras* de reescrita (DiretivaReescrita).

Ressalta-se que foi realizada uma extensão no modelo de Vieira (2008). A classe EntidadeContextual recebeu o auto relacionamento RelacionadaCom em virtude do processo concebido para reescrita de consultas ser baseado no modelo relacional e necessitar das informações sobre os relacionamentos entre entidades internas ao BD para efetuar operações que envolvam junções de dados.

4.3.1. Elementos Contextuais Internos e Externos

Visto que a abordagem Texere tem como pré-requisito a identificação do elemento contextual, é necessário realizar uma etapa de mapeamento desses elementos das diversas fontes de contexto para o banco de dados de Elementos Contextuais.

O próprio BD relacional convencional da aplicação, não sensível ao contexto, é uma das *fontes* de elementos contextuais. O ambiente de processamento (sistema operacional, rede de computadores, Internet) e perfis de usuários são as outras fontes de elementos contextuais usadas.

Baseado nas definições de contexto interno e contexto externo ao banco de dados (STEFANIDS et al., 2011) (vide Seção 2.4.2) e na definição de elemento contextual (VIEIRA, 2008) (vide Seção 2.3), definimos:

Definição 1 – Elementos contextuais internos são aqueles cuja fonte é o banco de dados da aplicação. Já elementos contextuais que possuírem fonte diferente são ditos **elementos contextuais externos**.

O processo de mapeamento de elementos contextuais internos é realizado pelo administrador de dados, quando as entidades do BD relacional alvo relevantes para identificação do contexto são mapeadas para a base de elementos contextuais como entidades contextuais (do tipo *entidade contextual interna*) e seus atributos como

elementos contextuais. Os relacionamentos existentes entre essas entidades devem ser mapeados como relacionamentos entre entidades contextuais internas e terem discriminados quais atributos formam suas chaves estrangeiras. Os relacionamentos *nxm* devem ser mapeados com tipo *entidade de relacionamento.*

As entidades contextuais de fontes externas ao BD devem ser mapeadas como do tipo *entidade contextual externa* e da mesma forma terem seus elementos contextuais cadastrados.

4.3.2. **Atores Envolvidos na Modelagem do Contexto**

A modelagem do contexto, a carga do banco de elementos contextuais e a especificação de regras que ditam como o sistema deve reescrever consultas são realizadas em tempo de projeto, por três atores: o administrador de dados, o especialista no domínio e o usuário final.

O Administrador de Dados

O Administrador de Dados (AD) é o responsável por identificar todos os elementos de dados, internos ou externos ao banco de dados da aplicação, que sejam considerados elementos contextuais, ou seja, possam ter relevância contextual para o domínio da aplicação e, potencialmente, sejam úteis para criação de regras. Esta identificação de entidades e elementos contextuais, suas localizações e relacionamentos são, então, descritos na forma de mapeamentos de dados e armazenados no BD de Elementos Contextuais.

Por exemplo, suponha que, em uma aplicação médica, o AD queira mapear “febril” como um contexto. O AD sabe que, nos dados do domínio existe o atributo temperatura corporal. Assim, ele cria no BD de Elementos Contextuais, um mapeamento para este novo elemento contextual. Neste caso, é cadastrado um mapeamento que informa que no BD da aplicação existe a propriedade temperatura corporal como um atributo denominado `temperatura_corporal` de uma entidade `paciente`. Os mapeamentos gerados serão utilizados nos processos de configuração de regras e nas tarefas de reescrita de consultas.

Outra atividade importante realizada pelo AD é manter consistente o mapeamento dos elementos contextuais, atualizando-os sempre que houver modificações nas fontes de contexto, que possam gerar inconsistências e erros nas regras de produção (e.g. novo formato de dado de um elemento contextual obtido por um sensor, alteração em unidade de medida).

O Especialista no Domínio

O Especialista no Domínio (ED) é outro usuário de função importante na Abordagem Texere. O seu trabalho é configurar as regras de produção no Motor de Inferência. O ED seleciona dentre todos os elementos contextuais já mapeados, os que irão compor regras de produção. No exemplo anterior, uma regra relacionada ao elemento contextual “temperatura corporal" pode ser “SE a temperatura corporal for maior do que 37 graus Celsius, ENTÃO atribua verdadeiro ao contexto febril". Uma vez que o contexto for identificado, uma consulta que pede possíveis diagnósticos poderia ser reescrita para considerar que doenças que têm febre como sintoma principal sejam listadas no início.

Tipicamente as regras configuradas pelo ED referem-se a regras de negócio. Por exemplo, consultas a bases de dados bancários devem restringir o acesso apenas às contas do usuário consultante.

Para configurar as regras, o ED utiliza o componente Gerenciador de Regras. Também é responsabilidade do ED, o gerenciamento da base de conhecimento do Motor de Inferência, no qual são armazenadas as regras de produção.

O acompanhamento da evolução do esquema de dados do BD alvo e as manutenções necessárias nas regras de produção afetadas por essas mudanças devem ser feitos de forma cooperativa e sincronizada pelo AD e pelo ED, tendo no componente Gerenciador de Regras da Arquitetura Texere a ferramenta concebida para auxílio nessas tarefas.

O Usuário Final

O terceiro ator importante para a abordagem é o usuário final. Da mesma forma que o ED, o usuário final também tem acesso ao Gerenciador de Regras para configurar regras de produção. A diferença entre as regras produzidas por esses dois tipos de usuários está na abrangência das ações produzidas por elas. Os usuários finais podem apenas especificar regras sobre opções de ordenação, classificação, restrições e preferências, aplicáveis aos dados sobre os quais tenha privilégio de consulta.

4.3.3. Inferência do Contexto

Para realizar a adequação de uma consulta Q ao contexto, por meio da sua reescrita, é necessário, além da captura dos elementos contextuais externos, realizar a análise dos elementos contextuais internos e do comando de consulta SQL emitido.

Em uma consulta ao BD, dispõe-se da identificação do agente (o usuário) e da tarefa (realizar a consulta Q). A consulta inicial Q e o foco da consulta são definidos da seguinte forma:

Definição 2 – Define-se uma **consulta inicial Q** como uma consulta em SQL de formato:

Q: SELECT α
FROM R
[WHERE ϕ];

Onde,

α = {E1.a1, ... En.an} é um conjunto de atributos qualificados por suas respetivas entidades, projetados na consulta Q;

R = $\{E_1, \ldots E_n\}$ é um conjunto de entidades selecionadas na consulta Q; e

ϕ é uma sentença condicional.

A consulta Q é representada na notação de álgebra relacional (CODD, 1970) como:

Q: $\pi_{\alpha}[\sigma_{\phi}](R)$;

Definição 3 – O **Foco de uma consulta Q** é a composição entre o usuário que a submete e a tarefa de realizar esta consulta.

Foco de Q: <usuário, realizar consulta Q>.

Já o contexto de uma consulta Q é denotado pelo conjunto de elementos contextuais do usuário, do ambiente físico e lógico, da aplicação, do dispositivo e de demais entidades contextuais que se relacionem com Q e possam interferir na relevância, para o usuário, das respostas apresentadas.

Definição 4 - Seja Q uma consulta inicial e EC um elemento contextual. Definimos Ctx(Q) como o **contexto de uma consulta Q**, formado pelo conjunto de EC relacionados à Q e relevantes para o foco de Q.

$Ctx(Q) = \{EC_i \mid EC_i \in (ECE_i \cup ECD_i \cup ECA_i \cup ECU_i)\}$, i=1..n;

Onde, ECE = conjunto de EC relevantes dos ambientes físico e lógico;
ECD = conjunto de EC relevantes do dispositivo;
ECA = conjunto de EC relevantes da aplicação; e
ECU = conjunto de EC relevantes do usuário.

Uma regra de produção pode resultar em uma asserção que irá disparar outra regra ou resultar em uma determinação de reescrita de consulta, conforme formato "SE *condição* ENTÃO *asserção ou ação"*. No qual, condição é uma condição lógica sobre o valor instanciado de um elemento contextual, asserção é a atribuição de valor a um elemento contextual ou a uma variável, e ação é a determinação da reescrita de consulta.

Para ilustrar um processo de inferência e reescrita, considere o exemplo no qual o usuário Pedro submete uma consulta Q sobre nomes e endereços de restaurantes em Recife. Esta consulta terá foco definido pela tupla <Pedro, realizar consulta Q>, no qual:

```
Q = SELECT r.nome, r.endereco
      FROM restaurante r
     WHERE r.cidade = 'Recife';
```

Considere que a avaliação do elemento contextual *dieta*, coletado do perfil do usuário Pedro, indica um contexto de restrição alimentar. Em uma sentença como a da seguinte regra: *"SE o objeto da consulta contém restaurante E usuário.dieta = 'vegetariana' ENTÃO reescreva Q para restringir as respostas para restaurante.tipo = 'vegetariano'"*, desencadearia a reescrita da consulta Q para a forma contextualizada Q':

```
Q' = SELECT r.nome, r.endereco
       FROM restaurante r
      WHERE r.cidade = 'Recife'
        AND r.tipo = 'vegetariano';
```

Observa-se que a análise do foco da consulta do exemplo, durante o processo de inferência do contexto, levou em consideração um elemento contextual relevante, a restrição alimentar do usuário, que em conjunto com os demais elementos contextuais, denotou um contexto específico e determinou a ação de reescrita.

O processo de inferência e reescrita, exemplificado informalmente, é especificado, detalhado e formalizado nas seções seguintes.

4.4. Reescrita de Consultas

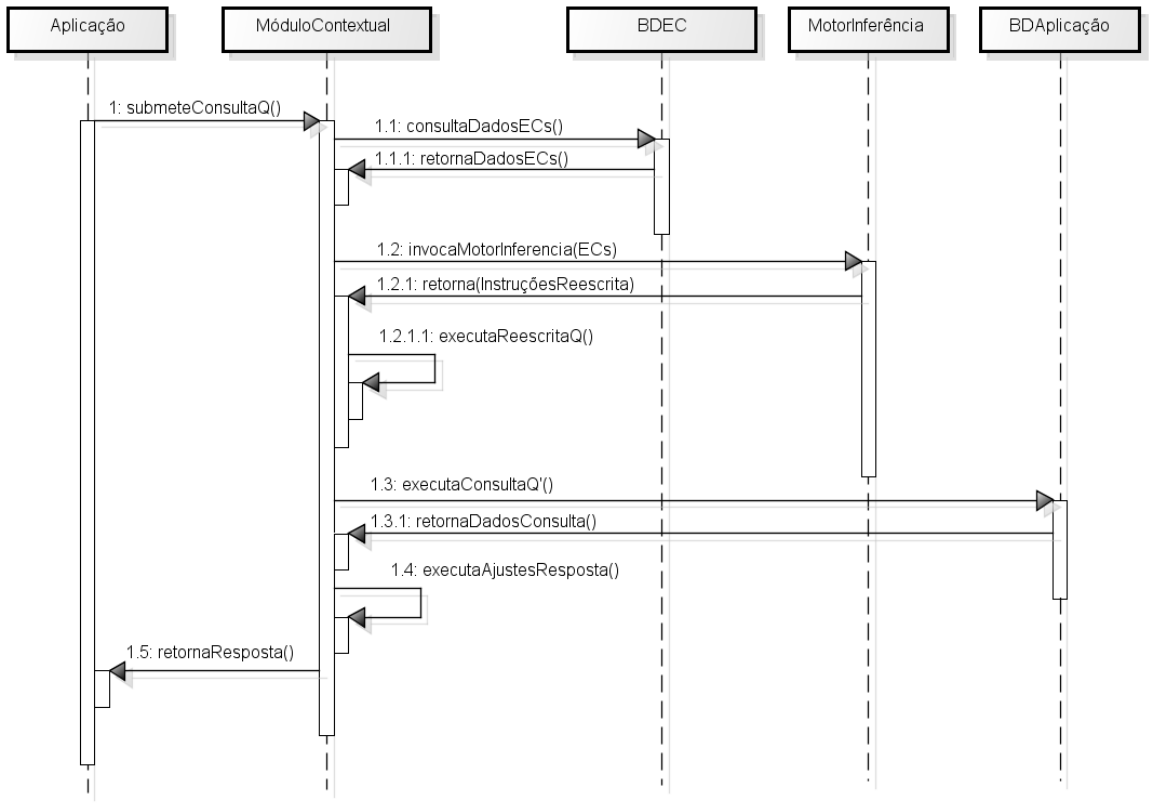
Para ter uma visão global do processo de reescrita de consultas, siga os fluxos do diagrama de sequência UML[28] Figura 4.6. Com a intenção de reescrever a consulta original Q submetida a partir da aplicação cliente, o Módulo Contextual atua como um mediador entre a aplicação e o banco de dados da aplicação, recebendo a consulta original Q (1). Depois disso, o Módulo Contextual acessa as informações sobre os elementos contextuais (1.1) e cria uma identificação da consulta.

Em seguida, o motor de inferência é chamado (1.2). Todos os elementos contextuais (internos e externos) são passados para o motor de inferência como parâmetros de entrada. Cada EC aciona regras correspondentes, com base nas características do domínio. O resultado da execução das regras irá identificar e/ou inferir o contexto e determinar ações para a reescrita da consulta. Um conjunto de instruções para reescrita é, então, repassado ao Módulo Contextual (1.2.1).

[28] UML, Unified Modeling Language. Disponível em: www.uml.org. Acesso em: 15 dez. 2014.

Nos últimos passos, o Módulo Contextual executa toda a transformação da consulta (1.2.1.1), submete a consulta reescrita Q' ao banco de dados da aplicação (1.3), realiza ajustes de formato na resposta, caso necessário (1.4), e retorna o resultado da consulta para a aplicação (1.5).

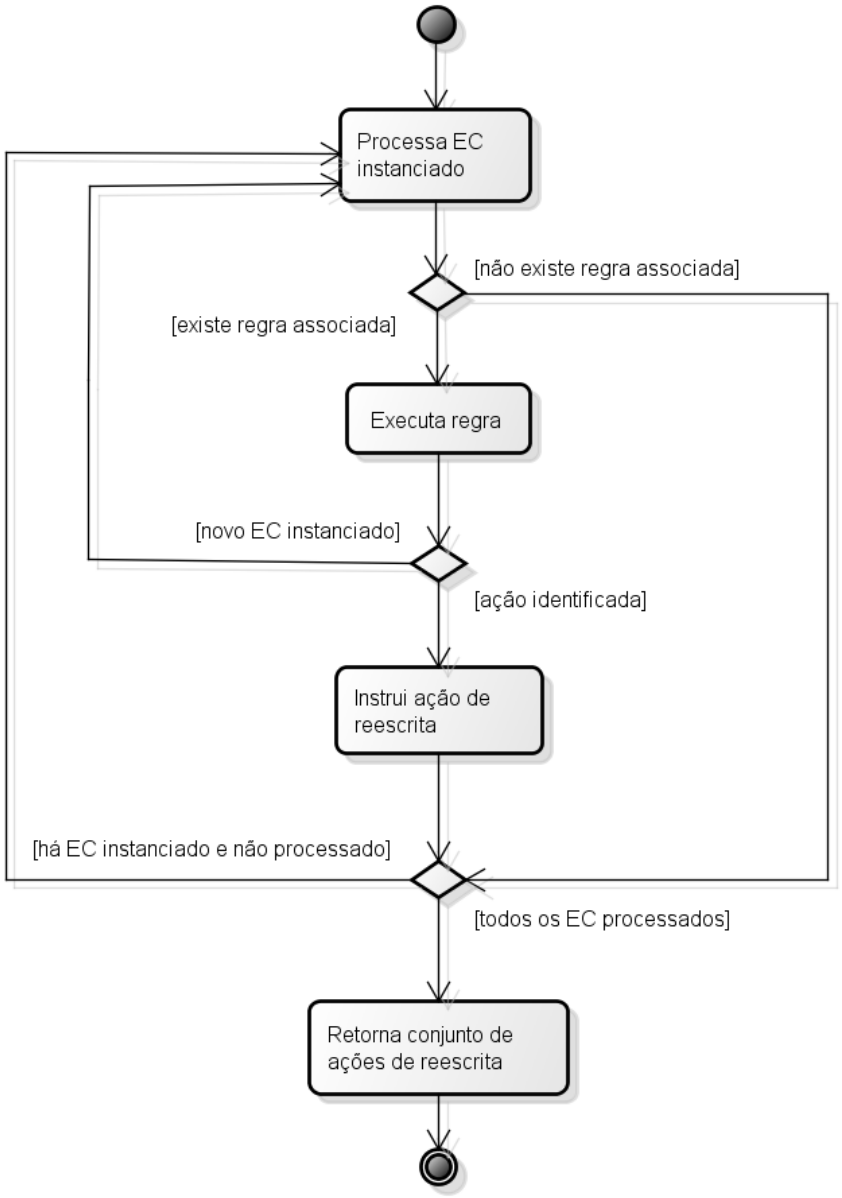
Figura 4.6 – Diagrama da reescrita e execução de consulta



Fonte: O autor.

A Figura 4.7 mostra o diagrama de atividades UML para o processamento das regras de produção pelo Motor de Inferência.

Figura 4.7– Diagrama de atividades para processamento de regras



Fonte: O autor.

A execução propriamente dita da reescrita obedece ao algoritmo Reescrita_Consulta, representado em pseudocódigo na Figura 4.8.

As instruções para reescrita de consultas, propostas nesta abordagem, são denominadas Diretivas de Reescrita de Consulta (DRC). As composições das DRC nas regras de produção encontram-se detalhadas nas próximas seções.

Figura 4.8 – Algoritmo em alto nível para reescrita de consulta

Reescrita_Consulta

```
Entrada: uma consulta Q,
         uma  lista de elementos contextuais instanciados (LEC);
Saída: a consulta reescrita Q';

01:   recebe Q;
02:   recebe lista de elementos contextuais instanciados LEC;
03:   SE válido (Q)
04:      chama motorInferencia (LEC);
05:      recebe conjunto de instruções de reescrita;
06:      Q' := Q;
07:      PARA cada instrução de reescrita recebida
08:           reescreve Q';
09:      FIM PARA
10:      retorne (Q');
11:    SENÃO
12:      retorna inválido (Q);
13:   FIM SE;
```

Fonte: O autor.

4.4.1. **Diretivas de Reescrita de Consultas**

Diretivas de reescrita de consultas são as instruções resultantes do processamento de regras de produção baseadas na análise de elementos contextuais externos e internos ao banco de dados e das entidades e atributos de uma consulta. Elas indicam quais alterações devem ser aplicadas ao código da consulta original para que esta seja transformada em uma consulta mais adequada ao contexto inferido e, neste sentido, atuam como ferramentas para ajuste ou sintonia fina do foco da consulta.

AS DRC são instruções para reescrita de uma consulta em SQL, de forma a melhor adaptá-la ao seu contexto e para as quais definimos:

Definição 5 - Seja Q uma consulta inicial em SQL e Δ uma **Diretiva de Reescrita de Consulta (DRC)**, define-se que a aplicação de Δ sobre uma consulta Q gerará uma consulta reescrita Q'.

$$\Delta\ (Q) \longrightarrow Q'$$

Onde, $\Delta \in \{\text{PROJECT, FILTER, ORDER, SKYLINE, GROUP, PREFER}\}$;

O processamento das DRC é realizado por algoritmos que consideram a reescrita de consultas no padrão SQL ANSI (SQL ISO 1992, 2003, 2008). O conjunto de DRC encontra-se relacionado no Quadro 4.1.

Quadro 4.1 – Conjunto de DRC

DRC	Aplicação	Descrição
PROJECT	Atributo	Determinar se um atributo deve ou não ser exibido.
FILTER	Instância	Determinar a aplicação de filtros de seleção para instâncias de atributos.
ORDER	Atributo/ Instância	Determinar a ordem de exibição de atributos ou instâncias de atributos.
SKYLINE	Atributo	Determinar preferências entre dois ou mais atributos de uma consulta em função de critérios de maximização ou minimização.
GROUP	Atributo	Determinar aplicação de função de agregação para atributos.
PREFER	Instância	Classificar o resultado de uma consulta de acordo com as preferências do usuário.

Fonte: O autor.

As formas gerais das DRC e seus algoritmos utilizam a convenção de símbolos apresentada no Quadro 4.2.

Quadro 4.2 – Símbolos utilizados nas formas gerais das DRC

Símbolo	Significado
[ ]	opcionalidade
E.a	entidade.atributo
not	operador de negação
||	operador de concatenação
α	uma lista de atributos
γ	função de agregação ∈ {MAX, MIN, COUNT, AVG, SUM}
λ	operação de limitação
ϕ	sentença condicional
π	operação relacional de projeção
σ	operação relacional de seleção
$\bowtie$	operação relacional de junção
τ	operação de ordenação
ASC	ordenação ascendente
DESC	ordenação descendente
R	relação
<oc>	Operador de comparação ∈ { LIKE | = | < > | < | > | <= | >= | IN | EXISTS}.
	operador lógico ∈ { AND | OR}
<vl>	valor instanciado

Fonte: O Autor.

Para a representação das consultas Q e Q` nas definições formais das DRC foi utilizada a notação das operações sobre conjuntos (projeção (π), seleção (σ), junção ($\bowtie$) e produto cartesiano (x)) da álgebra relacional (CODD, 1970). As

operações ordenação (τ) e agregação (γ) utilizam notação em álgebra relacional estendida (CODD, 1979; GRUMBACH; TOVA, 1996; ROTH et al., 1988) e a operação de limitação (λ), a notação de Li (2005) para consultas *top k*.

A seguir são definidas cada uma das cinco primeiras DRC e apresentadas as suas sintaxes. Na Seção 4.4.7 são apresentados exemplos de aplicação de cada DRC. Da mesma forma, o Apêndice B traz uma lista adicional de exemplos com situações mais complexas de reescrita. A DRC PREFER será apresentada de forma isolada na Subseção 4.4.8, na qual será abordado o modelo de preferências adotado.

Os algoritmos das diretivas apresentadas encontram-se listados, em formato de pseudocódigo, no Apêndice A.

4.4.2. **Diretiva Project**

A diretiva PROJECT tem como objetivo determinar se um ou mais atributos devem ou não ser exibidos no resultado de uma consulta reescrita, conforme forma geral e definição a seguir:

PROJECT L

Onde, $\mathrm{L} = \{[\mathrm{NOT}]\mathrm{E}_1.\mathrm{a}_1, \ldots[\mathrm{NOT}]\mathrm{E}_n.\mathrm{a}_n\}$, é uma lista de atributos qualificados pelas suas entidades.

Definição 6 - DRC PROJECT: Considere Q uma consulta SQL inicial e Q' a consulta reescrita resultante da aplicação da diretiva PROJECT sobre Q.

Dado: Q: $\pi_{\alpha}\left[\sigma_{\phi}\right](\mathrm{R})$;

Diz-se que: PROJECT L (Q) $\longrightarrow$ Q' | Q': $\pi_{\alpha'}\left[\sigma_{\phi}\right](\mathrm{R}')$ (1)

Onde, $\alpha' = \begin{cases} \alpha - \{\mathrm{E}i.\mathrm{a}j\} & \text{se } \mathrm{E}i.\mathrm{a}j \textit{ precedido por } \mathrm{NOT} \\ \alpha \cup \{\mathrm{E}i.\mathrm{a}j\} & \textit{em caso contrário.} \end{cases}$ (2)

R' é a relação resultante da junção natural ($\bowtie$) entre as entidades E_i cujos atributos perteçam a α'. (3)

Ou seja, dada uma consulta Q em SQL, a DRC PROJECT gera uma consulta reescrita Q' (1), na qual os atributos de uma lista L são inseridos ou removidos (2) da

cláusula SQL SELECT de Q. Para a inserção, caso a entidade possuidora desse atributo não faça parte da consulta Q, é realizada verificação na base de elementos contextuais se há relacionamento direto[29] ou indireto[30] entre entidades especificadas na diretiva e entidades da consulta Q. Caso haja relacionamento, deve ocorrer uma junção natural (⋈) entre essas entidades (3). A junção incluirá a(s) entidade(s) relacionada(s) na cláusula FROM de Q'. O formato da consulta em SQL, para Q' é:

Q': SELECT α'
FROM R'
[WHERE ϕ];

Observação: Para efeito de simplificação, assume-se a junção natural (natural join) para a formação de R'. A mesma situação é considerada para as outras definições de DRC apresentada a seguir.

Exemplo: Suponha a consuta Q e a diretiva PROJET a seguir, em um contexto de utilização de dispositivo com restrição de espaço para informações:

```
Q:    SELECT l.titulo, l.idioma, l.resumo
      FROM livro l;

SE &DISPOSITIVO.tipo = 'SMARTPHONE'  ENTÃO
       PROJECT (NOT livro.resumo, autor.nome);
FIM-SE;
```

A consulta Q', resultante da aplicação da diretiva, será:

```
Q':   SELECT l.titulo, l.idioma, a.nome
        FROM livro l, autor a
       WHERE a.autor_id = l.autor_id;
```

4.4.3. **Diretiva Filter**

FILTER é uma diretiva que tem como objetivo realizar filtros nas tuplas retornadas pela consulta reescrita, por meio da inserção de condições de filtragem na cláusula SQL WHERE da consulta original. A DRC FILTER possui a seguinte sintaxe:

FILTER C

[29] Relacionamento 1x1 ou 1xn.
[30] Relacionamento nxm.

Onde,

C é uma expressão condicional $C = (C_1 _1 C_2 \ldots _{n-1} C_n)\ i=1..n$;

Onde,

$$Ci = \begin{cases} [\gamma]\text{E.a } <oc> <vl> & \text{uma condição sobre um valor; ou} \\ [[\text{NOT}]\ \text{EXISTS}]\ \text{E.a } <oc>\ (SCc) & \text{uma condição sobre uma subconsulta correlacionada.} \end{cases}$$

 representa um operador lógico;

<oc> representa um operador de comparação;

<vl> representa um valor a ser comparado;

γ representa uma função de agregação, opcional, sobre E.a; e

SCc representa uma subconsulta SC para uma condição c.

Os operadores de comparação utilizados nas sentenças de condição da DRC FILTER e os valores admitidos para cada caso estão apresentados no Quadro 4.3.

Quadro 4.3 – Operadores admitidos na função de filtragem da DRC Filter

<oc>	<vl>	Descrição
[NOT] LIKE	's'	s é uma sequência de caracteres.
	C	C é uma expressão condicional que será avaliada pelo resultado de uma subconsulta.
= < >	x	x é uma constante.
	n	n é um número inteiro.
	null	Valor nulo.
	C	C é uma expressão condicional que será avaliada pelo resultado de uma subconsulta.
	E.a	Entidade.atributo.
	γ (E.a)	função de agregação sobre E.a.
< > <= >=	n	n é um número inteiro.
	E.a	Entidade.atributo.
	γ (E.a)	função de agregação sobre E.a.
[NOT] IN	('x_1',... 'x_n')	Comparação de igualdade com pelo menos um dos valores x e na qual os valores x são constantes alfanuméricas.
	(n_1,..., n_n)	Comparação de igualdade com pelo menos um dos valores n e na qual os valores n são constantes numéricas.
	C	Avaliação se um valor ou atributo está contido na relação resultante de uma subconsulta. C é a expressão condicional que será avaliada pela subconsulta.
[NOT] EXISTS	C	Avaliação se a subconsulta resultante da expressão condicional C for não vazia.

Fonte: O autor.

Definição 7 - DRC Filter: Considere Q uma consulta SQL sobre uma relação R:

Dado: $Q: \quad \pi_{\alpha}[\sigma_{\phi}](R);$

Diz-se que: $\text{FILTER } C\,(Q) \longrightarrow Q' \mid Q'\colon \pi_{\alpha}\,\sigma_{\phi'}\,(R'),$

$$\text{Onde,} \quad \phi' = \phi \text{ AND } C_i _i C_{i+1} _{i+1} \ldots C_n; \text{ e}$$

$$R' = \begin{cases} R, & \textit{caso a entidade } Ei \text{ da condição } Ci \in R \\ R \bowtie Ei, & \textit{em caso contrário.} \end{cases}$$

O formato em SQL da consulta Q' neste caso será:

```
Q':   SELECT α
        FROM R'
      WHERE Ci <ol> Ci+1 <ol> Cn;
```

Os casos de utilização de funções de agregação γ ou condição C em subconsulta implicarão na aplicação da condição de filtragem sobre o resultado de uma subconsulta (SC). Os formatos resultantes para Q' em SQL, nesses casos, são os seguintes:

```
Q':   SELECT α
        FROM R'
      WHERE γ(E.a) <oc> SC;
```

```
Q':   SELECT α
        FROM R'
      WHERE [NOT] EXISTS (SCc);
```

Exemplo: Suponha a consulta Q a seguir que informa produtos alimentícios lidos de uma entidade `produto`.

```
Q:      SELECT p.nome, p.preço
          FROM produto p
         WHERE p.tipo = 'alimento';
```

Considere uma regra de produção criada para clientes alérgicos a glúten que utiliza a diretiva FILTER informando a restrição sobre a entidade `ingrediente` relacionada a `produto`.

```
SE &USER.alérgico_glúten = TRUE ENTÃO
    FILTER (NOT EXISTS ingrediente.nome = 'glúten')
FIM-SE;
```

A aplicação da DRC realiza a reescrita com filtro utilizando a cláusula NOT EXISTS especificada e a junção (não declarada explicitamente) entre `produto`, `ingrediente` e a entidade de relacionamento `ingrediente_produto`, resultando na seguinte consulta:

```
Q':     SELECT p.nome, p.preço
          FROM produto p
         WHERE p.tipo = 'alimento'
           AND NOT EXISTS (SELECT *
                             FROM ingrediente i,
                                  ingrediente_produto ip
                            WHERE i.produto_id = p.produto_id
                              AND ip.produto_id = p.produto_id
                              AND i.nome = 'glúten');
```

A diretiva FILTER considera que caso a entidade possuidora do atributo E.a determinado na condição de filtragem C não faça parte da consulta original Q, deve ocorrer uma junção natural (⋈) entre essa entidade e as entidades da consulta Q, sendo necessário um procedimento de verificação na base de elementos contextuais se há relacionamento direto ou relacionamento indireto entre essas entidades, o que determina a junção entre elas.

Caso uma entidade possua mais de uma chave estrangeira para outra, a junção implícita não é possível, devendo-se explicitar na DRC PROJECT qual chave estrangeira deve ser utilizada. Esta situação está caracterizada no Exemplo IV do Apêndice B.

4.4.4. **Diretiva Order**

Dada uma consulta SQL original Q, a DRC ORDER é definida como uma instrução que determina a reescrita de Q especificando a ordem de exibição de atributos no resultado de uma consulta reescrita Q' e, opcionalmente, limitando o número de tuplas retornadas.

A diretiva ORDER utiliza as cláusulas SQL ORDER BY, ORDER BY CASE ou OVER (ORDER BY) e possui a seguinte sintaxe:

ORDER T [LIMIT RANK / DENSE_RANK k]

Onde, T é uma lista de argumentos de ordenação que pode conter expressões de dois tipos:

- <*atributo*> é uma lista de atributos para ordenação qualificada com formato ($E_1.a_1$ [DESC], ... $E_n.a_n$ [DESC]), na qual DESC significa ordenação decrescente, e cuja omissão implica em ordenação crescente.

  Exemplo: `ORDER (funcionário.setor DESC, funcionário.salário).`

- <*valor_atributo*> é uma expressão de formato (E.a CASE (<vl>$_1$ / <cond>$_1$, <arg>$_1$ [DESC]),... (<vl>$_n$ / <cond>$_n$, <arg>$_n$ [DESC]), (ELSE, <arg>) [DESC]).
  Onde, <vl> é um valor instanciado para o atributo e <cond> é uma expressão condicional para este valor. <arg> informa o argumento de ordenação utilizado caso o valor <vl> seja lido, ou em casos contrários (ELSE). Opcionalmente pode-se usar o próprio atributo como <arg>, o que significa utilizar o próprio valor instanciado.

A opção LIMIT determina uma limitação do número k de tuplas retornadas pela resposta em função de um critério de classificação de duas formas: mantendo a posição relativa após "empates" (RANK) ou sem manter a posição relativa após empates (DENSE_RANK).

Exemplo: Considere uma DRC hipotética que informa que a tupla cujo cargo seja 'presidente' deve ser listada primeiro, e em seguida as tuplas com outros cargos, ordenadas pelo próprio nome do cargo de forma crescente:

```
ORDER (cargo.nome CASE ('presidente', 'a'),(ELSE, cargo.nome);
```

Definição 8 - DRC Order: Considere Q uma consulta SQL inicial sobre uma relação R:

Dado: Q: $\pi_{\alpha}[\sigma_{\phi}](R)$;

Diz-se que:

i) ORDER T (Q) $\longrightarrow$ Q' | Q': $\tau_T \pi_{\alpha}[\sigma_{\phi}](R')$

Onde, $R' = \begin{cases} R, & caso \text{ a entidade Ei de T } \in R \\ R \bowtie Ei, & em\ caso\ contrário. \end{cases}$

τ_T representa uma expressão de ordenação T.

ii) ORDER T (Q) LIMIT φ k $\longrightarrow$ Q' | Q': $\lambda_k \tau_{T\varphi} \pi_{\alpha} \sigma_{\phi'}(R')$

Onde, $R' = \begin{cases} R, & caso \text{ a entidade Ei de T } \in R \\ R \bowtie Ei, & em\ caso\ contrário. \end{cases}$

λ_K representa a limitação em top k tuplas;

$\tau_{T\varphi}$ representa a expressão de ordenação T, sob critério de classificação φ, com $\varphi \in \{RANK, DENSE\_RANK\}$ ASC/DESC; e

$\phi' = \phi$ AND (φ <= k).

A sintaxe i) atende aos casos de ordenações simples, dos tipos <*atributo*> ou <*valor_atributo*>. Já a sintaxe ii) diz respeito a ordenações que solicitem cláusula de limitação LIMIT (ver exemplos VI e IX do Apêndice B). Em ambos os casos, se uma entidade referida na diretiva não faz parte da consulta Q, deve ocorrer uma junção (⋈) entre essas entidades.

A operação de ordenação definida pela DRC ORDER pode ser realizada com as cláusulas SQL ORDER BY ou ORDER BY CASE, para diretivas que não solicitem opção LIMIT ou por meio do uso da cláusula OVER (ORDER BY) (SQL ISO, 2008) em conjunto com as funções RANK e DENSE_RANK. LIMIT determina uma limitação do número k máximo de tuplas retornado pela resposta, em função de um critério de classificação, mantendo a posição relativa após empates (RANK) ou sem manter a posição relativa após empates (DENSE_RANK). Por exemplo, para a sequência numérica S= {40, 50, 60, 70, 70, 70, 90, 110, 110}, as opções de classificação são as seguintes: Rank = {1, 2, 3, 4, 4, 4, 7, 8, 8}; Dense_rank = {1, 2, 3, 4, 4, 4, 5, 6, 6}.

São formatos resultantes para Q', após processamento desta DRC:

a)
```
SELECT α
  FROM R'
 [WHERE C]
ORDER BY T;
```

b)
```
SELECT α
  FROM R'
 [WHERE C]
ORDER BY
  CASE WHEN <vl>1 / <cond>1
       THEN <arg>1
  ...
  CASE WHEN <vl>n / <cond>n
       THEN <arg>n
  ELSE <arg>
  END;
```

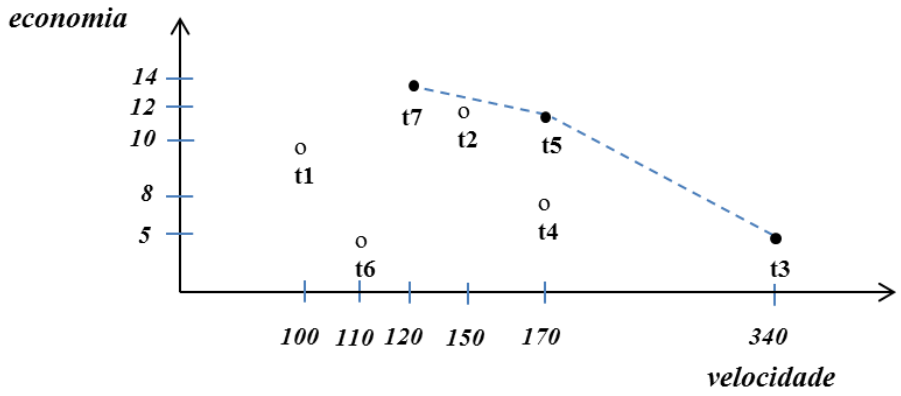
c)
```
SELECT *
 FROM (SELECT α, φ OVER (ORDER BY T) as <alias_rank>
         FROM R'
        [WHERE ϕ])
WHERE <alias_rank> <= k;
```

4.4.5. **Diretiva Skyline**

A diretiva Skyline tem por objetivo determinar preferência em consultas em função de critérios de maximização ou minimização de atributos. Esta diretiva utiliza a forma de consulta conhecida como *Skyline* (BÖRZSÖNYI et al., 2001) e tem aplicação para expressão de critérios de seleção conflitantes. As consultas do tipo *skyline* realizam filtragem das tuplas ditas “interessantes” que são aquelas que não são dominadas por nenhuma outra. Uma tupla *t1* é considerada dominante (ou interessante) sobre uma tupla *t2* se *t1* é igual ou melhor do que *t2* em todos os critérios de seleção adotados e *t1* é melhor que *t2* em pelo menos um critério. O nome *skyline* deve-se ao perfil característico formado pelos pontos selecionados por este tipo de distribuição, quando dispostos em forma de gráfico.

Por exemplo: Tenho interesse pela relação de carros mais velozes e de maior economia. A distribuição da consulta segue o gráfico bidimensional da Figura 4.9, na qual os pontos t3, t5 e t7 representam as tuplas da resposta skyline para MAX(velocidade) e MAX(economia).

Figura 4.9– Exemplo de distribuição *Skyline*



Fonte: O autor.

Como dito, uma consulta Skyline seleciona todas as tuplas interessantes, ou seja, tuplas que não são dominadas por nenhuma outra. Dadas as tuplas $p = (p_1,\ldots,p_k, p_{k+1},\ldots p_l)$ e $q = (q_1,\ldots,q_k, q_{k+1},\ldots q_l)$, para uma consulta Skyline: SKYLINE de d1 MIN, . . . , dk MIN, dk+1 MAX, . . . , dl MAX. Define-se que p domina q se satisfaz a essas duas condições (BÖRZSÖNYI et al., 2001)

$p_i \leq q_i \quad \text{para todo} \quad i = 1, \ldots, k$

$p_i \geq q_i \quad \text{para todo} \quad i = (k+1), \ldots, l$

Definição 9 - DRC Skyline: Dada uma consulta Q, a diretiva SKYLINE instrui a reescrita de Q para gerar respostas em função da classificação de atributos maximizados ou minimizados.

Dado: Q: $\pi_{\alpha}[\sigma_{\phi}](R)$;

Diz-se que: SKYLINE E (Q) $\longrightarrow$ Q' |

$$Q': \tau_T \pi_{\alpha}[\sigma_{\phi}](R[\alpha 1]) - \pi_{\alpha}[\sigma_{\phi}](\pi_{\alpha} R[\alpha 1] \bowtie_{\alpha 2\ DOM(D)\ \alpha 1} \pi_{\alpha} R[\alpha 2])$$

Onde: E determina um critério de dominância, com $E = \{(MAX/MIN(E.a_i), [<pos>_i])...MAX/MIN(E.a_{i+1}), [<pos>_{i+1}]) \ldots MAX/MIN\ (E.an), [<pos>_n])\}$; $DOM\ (D)$ representa a dominância sob o critério D;

$[\alpha_n]$ é a uma renomeação de α. Isso possibilita a comparação de dois atributos com mesmo nome (como nesta junção com a mesma relação). Neste caso, assuma que $\alpha=\{E.a_1, E.a_2, \ldots, E.a_n\}$ será renomeado para $\alpha 1=\{E1.a_1, E1.a_2, \ldots, E1.a_n\}$ e $\alpha 2=\{E2.a_1, E2.a_2, \ldots, E2.a_n\}$;

MAX/MIN determinam função de maximização ou minimização para classificação do atributo (E.a) informado;

$\bowtie_{\alpha 2\ DOM(D)\ \alpha 1}$ significa a junção de $R[\alpha_1]$ com $R[\alpha_2]$ sob critério $\alpha 2\ DOM(D)\ \alpha 1$; e

τ_T é uma expressão de ordenação com

$$T= (E.a_{i<pos>1}, E.a_{i<pos>2, \ldots,} E.a_{i<pos>n}) |$$
$$(E.a_{i<pos>1}, E.a_{i<pos>2, \ldots,} E.a_{i<pos>n}) \in E.$$

A diretiva SKYLINE promove a reescrita da consulta Q por meio da realização da junção ($\bowtie$) da relação presente em Q com ela própria. Isso possibilita a comparação de uma tupla com todas as outras e subtração das tuplas dominadas (operador -), ou seja, as tuplas cujos atributos que não atendam aos critérios de dominância D (maior ou igual da relação, se usada função MAX ou menor ou igual da relação, se usada função MIN). Opcionalmente, é possível atribuir uma ordenação (τ_T) para a listagem e posições $<pos>$ na sequência de ordenação da DRC para cada atributo $E.a_i$.

Com o objetivo de obter melhor desempenho, a especificação algorítmica da DRC SKYLINE não utiliza junção e diferença entre relações, mas utiliza subconsulta

com a cláusula NOT EXISTS, o que resulta na seguinte forma geral em SQL, para Q':

```
SELECT r1.α
  FROM R as r1
WHERE NOT EXISTS (
                SELECT r2.α
                  FROM R as r2
                WHERE r2.Ea1 <oc> r1.Ea2
                    . . .
                  AND r2.Ea1 <oc> r1.Ean
                   OR r2.Ea2 <oc> r1.Ea1
                  AND r2.Ea2 <oc> r1.Ea3
                    . . .)
ORDER BY E.ai, E.aj . . . ;
```

Exemplo: Considere a consulta Q e a regra R a seguir. R informa, baseada em duas preferências do usuário, que se deve usar uma consulta que maximize velocidade e minimize o consumo/litro dos veículos apresentados e estes sejam listados ordenados por velocidade em primeiro e depois consumo.

```
Q:    SELECT *
        FROM automóvel;

R:    SE usuario.interesse1 = 'velocidade' AND
         usuario.interesse2 = 'economia'
            ENTÃO
               SKYLINE {automovel (MAX (automóvel.velocidade,1),
                                   MIN (automóvel.consumo,2))};
```

Aplicando-se a DRC SKYLINE da regra R, a consulta Q' resulta na seguinte forma :

```
Q':    SELECT *
         FROM automovel AE
       WHERE NOT EXISTS (
                 SELECT *
                   FROM automovel AI
                  WHERE AI.velocidade >= AE.velocidade
                    AND AI.consumo > AE.consumo
                     OR AI.velocidade > AE.velocidade
                    AND AI.consumo >= AE.consumo)
       ORDER BY velocidade, consumo;
```

4.4.6. **Diretiva Group**

Esta diretiva tem como objetivo gerar resultados sumarizados por uma função de agregação no resultado de uma consulta reescrita. A sintaxe da DRC GROUP é:

GROUP A [, LIMIT RANK/DENSE_RANK, [<alias>]]

Onde: A = ({E.a_1,.. E.a_n}, γ(E.a_{n+1}) [<oc> <vl>]) é uma dupla composta por um conjunto de atributos a serem agregados e uma função γ de agregação aplicada sobre o atributo E.a_{n+1}, com $\gamma \in$ {SUM, COUNT, AVG, MAX, MIN};

γ(E.a_{n+1}) <oc> <vl> denota uma restrição sobre função de agregação;

LIMIT determina uma limitação do número k de tuplas retornado pela resposta em função de um critério de classificação de duas formas: mantendo a posição relativa após "empates" (RANK) ou sem manter a posição relativa após empates (DENSE_RANK); e

<alias> determina um apelido opcional para a exibição da resposta da função de agregação.

Definição 10 - DRC Group: Considere Q uma consulta SQL inicial sobre uma relação R:

Dado: Q: $\pi_{\alpha}[\sigma_{\phi}](R)$;

Diz-se que: i) GROUP A (Q) $\longrightarrow$ Q' | Q': $\gamma_A \pi_{\alpha'}[\sigma_{\phi'}](R')$

Onde $\alpha' = \{E.a_1,.. E.a_n\} \mid \{E.a_1,.. E.a_n\} \in A;$ (1)

$$R' = \begin{cases} R, & caso \text{ a entidade Ei de A } \in R \\ R \bowtie Ei, & em\ caso\ contrário. \end{cases} \quad (2)$$

$\phi' = \phi$ AND γ(E.a_{n+1}) <oc> <vl>. (3)

ii) GROUP A (Q) LIMIT φ <oc> k $\longrightarrow$ Q' | Q': $\lambda_k \gamma_A \pi_{\alpha'}[\sigma_{\phi'}](R')$

Onde $\alpha' = \{E.a_1,.. E.a_n\} \mid \{E.a_1,.. E.a_n\} \in A;$ (4)

$$R' = \begin{cases} R, & caso\ Ei \in R \\ R \bowtie Ei, & em\ caso\ contrário; e \end{cases} \quad (5)$$

$\phi' = \phi$ AND γ(E.a_{n+1}) <oc> <vl>. (6)

Onde,

λ_K representa a limitação em top k tuplas;

γ_A representa a agregação sobre um conjunto de atributos {E.a_1,.. E.a_n} e função de agregação $\gamma \in \{SUM, COUNT, AVG, MAX, MIN\}$ sobre E.a_{n+1};

φ representa o critério de classificação $\varphi \in \{RANK, DENSERANK\}$ ASC/DESC; e

$\sigma_{\phi'}$ representa a expressão de condição para linhas e valores agregados.

A sintaxe i) define os resultados consolidados em função de uma função de agregação γ. A projeção α' de Q' deverá conter os atributos a serem agregados e a função de agregação γ especificados na diretiva (1). Caso a entidade de agrupamento referida na diretiva não faça parte da consulta Q, deve ocorrer uma junção entre essas entidades (2). Havendo filtragem sobre o(s) valor(es) agregado(s) (3), esta deverá estar presente em Q' por meio da cláusula SQL HAVING. O formato resultante para Q', após o processamento da DRC GROUP para esta sintaxe utiliza o comando SQL GROUP BY:

```
SELECT α', γ(E.an+1) [AS <alias>]
  FROM R'
[WHERE ϕ]
GROUP BY α'
[HAVING γA <oc> <vl>];
```

Exemplo: Suponha uma consulta Q a um banco de dados de sensores de tráfego, que realize a leitura de movimentos detectados. Dependendo da rotina que realize esta consulta ela deverá ser adaptada a uma forma de agregação específica (contexto da aplicação). Considere que a regra R foi configurada para contabilização mensal de movimentos, por sensor e objetos com mais de 100 movimentos. Para tanto foi usada a DRC GROUP a seguir, em conjunto com uma DRC FILTER.

```
Q:    SELECT s.sensor_id, o.objeto_id, m.timestamp
        FROM sensor s, movimento m, objeto o
       WHERE m.sensor_id = s.sensor_id
         AND m.objeto_id = o.objeto_id;

R:    SE aplication.name = '100 mais mês' ENTÃO
        FILTER((movimento.mes = &time.mes)
                AND (movimento.ano = &time.ano));
         GROUP (sensor.sensor_id, objeto.objeto_id,
                 COUNT(movimento.timestamp) > 100, 'Qtd. movimentos');
```

```
Q`:   SELECT s.sensor_id, o.objeto_id,
             count(m.timestamp) as 'Qtd. movimentos'
        FROM sensor s, movimento m, objeto o
       WHERE m.sensor_id = s.sensor_id
         AND m.objeto_id = o.objeto_id;
         AND m.mes = &time.mes
         AND m.ano = &time.ano
    GROUP BY s.sensor_id, o.objeto_id, count(m.timestamp)
      HAVING COUNT(*) > 100;
```

Já a sintaxe ii), além dos elementos necessários para agregação, especifica a limitação para as k primeiras tuplas retornadas. A exemplo da DRC ORDER, a cláusula LIMIT (λ_k) impõe a utilização da cláusula OVER, mas neste caso em combinação com a cláusula PARTITION BY (SQL ISO, 2008) e as funções de classificação RANK e DENSE_RANK (vide exemplo XII do Apêndice B). O formato em SQL resultante para Q', após processamento da sintaxe ii) é:

```
SELECT *
  FROM (SELECT α', φ (γ(E.an+1)) OVER (PARTITION BY α') as <alias>
          FROM R'
        [WHERE ϕ])
 WHERE <alias> <= k;
```

O algoritmo Group e algoritmos complementares Groupby e Groupover encontram-se listados em formato de pseudocódigo no Apêndice A.

4.4.7. **Conflitos Entre Regras**

A configuração de regras de reescrita é uma tarefa que exige atenção no que diz respeito à possibilidade de conflitos entre DRC ou conflitos de consulta. Os conflitos entre DRC envolvem chamadas às DRC Order, Group, Prefer e Skyline, que devem ser mutuamente exclusivas, uma vez que essas DRC atuam na forma como a consulta realiza a ordenação dos dados. Já a DRC Project pode ser chamada em conjunto com uma das anteriores.

Conflitos de consulta dizem respeito ao sentido semântico da consulta, por exemplo, uma consulta não pode ser reescrita para selecionar um objeto cujo atributo cor seja igual a 'amarelo' e igual a 'verde' ao mesmo tempo, ou um atributo salário ser menor que dez mil e maior que onze mil. Casos como esses devem ser evitados para uma mesma situação de contexto que provoque a chamada a regras distintas.

De forma geral é de responsabilidade do ED a verificação de regras de inferência que disparem situações conflitantes.

4.4.8. **Exemplos de Regras e Aplicação de DRC**

Para exemplificar a utilização das DRC apresentadas, considere o esquema de dados relacional da Figura 4.10, utilizado por uma aplicação de vendas de livros desenvolvida para ser utilizada segundo a Abordagem Texere. As chaves primárias estão sublinhadas e para efeito de simplificação, os atributos com o mesmo nome de chaves primárias são considerados chaves estrangeiras.

Figura 4.10 – Esquema relacional de exemplo

LIVRO	GENERO	EDITORA	CLIENTE	LIVRARIA
id_livro	id_genero	id_editora	id_cliente	id_livraria
nome	nome	nome	idade	endereço
nome_autor			id_pais	
id_genero				
id_formato	RESUMO_VENDA	PAIS	ESTOQUE	FORMATO
idioma	id_livro	id	id_livraria	id_formato
resenha	qtd_venda	nome	id_livro	nome
imagem_capa			qtd	
preco				

Fonte: O autor.

Considere que na base de conhecimento do Motor de Inferência, as seguintes regras de produção encontram-se cadastradas:

```
R1:                                          /* faixa_etaria criança
    SE &usuário.idade <= 12 ENTÃO
       &infantil := true;

R2:                                          /* criança em férias
    SE &infantil ENTÃO
      FILTER (genero.nome IN('contos_de_fada','novela_infantil','escolar')),
      SE &data.mês IN ('janeiro', 'fevereiro', 'julho') ENTÃO
          ORDER (genero.nome CASE (('escolar',2), (ELSE, 1)));

R3:                                          /* limitação fisica visual
    SE &usuário.limite_fisico = 'visual' ENTÃO
      SE Q.livro ENTÃO
           FILTER (livro.formato IN ('áudio', 'braile'));
R4:                                          /* dispositivo smartphone
    SE Q.livro ENTÃO
      SE dispositivo = 'smartphone' ENTÃO
           PROJECT (NOT livro.resumo, livro.imagem_capa);

R5:                                          /* aplicação rotina de ofertas
      SE &aplicação.nome = 'melhores ofertas' ENTÃO
          SKYLINE (MIN(livro.preco,1), MAX(resumo_venda.qtd_venda,2));
          PROJECT (livro.preco);
```

As variáveis iniciadas com ‘Q.’ indicam a participação de entidade (ou atributo qualificado com entidade) contida na consulta e são avaliadas como um dado booleano. Variáveis iniciadas com ‘&’ indicam elementos contextuais externos ou contextos intermediários (como `infantil`, no exemplo).

Baseado nas regras mencionadas, considere as situações a seguir como exemplos da análise dos elementos contextuais envolvidos em uma consulta e das regras de produção utilizadas. A consulta Q_1 realizada é a seguinte:

```
Q1 = SELECT nome, resumo FROM livro;
```

Situação a) Consulta realizada por uma garota de 8 anos sobre livros disponíveis para venda, utilizando um *tablet*. Os elementos contextuais externos capturados foram `{&usuário.idade = 8, &dispositivo.tipo = tablet, &aplicação.nome = ‘sugestão de livros’, &data = 01/07/2014}`. A informação complementar extraída da análise da consulta foi `Q.livro.nome = true` e `Q.livro.resumo = true`, que indica o conteúdo da consulta inicial.

Uma vez que o motor de inferência processe as regras de produção, as regras R1 e R2 são disparadas, e a consulta Q_1’ será reescrita para a forma:

```
Q1’ = SELECT nome, nome_autor, resenha
        FROM livro, genero
       WHERE livro.id_genero = genero.id_genero
         AND genero.nome
             IN (‘contos_de_fada’, ‘novela_infantil’, ‘escolar’)
    ORDER BY
        CASE WHEN genero.nome = ‘escolar’ THEN 2
                                          ELSE 1
                                          END ASC;
```

Situação b) A mesma consulta Q, é realizada por um senhor deficiente visual (utilizando aplicação por voz). Neste caso, os elementos contextuais externos capturados foram `{&usuário.idade = 67, &usuário.limite_fisico = ´visual´, &aplicação.nome = ‘sugestão de livros’, &data=01/07/2014}`. Apenas as regras R3 e R5 foram disparadas, o que gerou a consulta reescrita:

```
Q1’ = SELECT nome, resumo
        FROM livro,
       WHERE livro.id_formato = formato.id
         AND formato.nome IN (‘áudio’,‘braile’);
```

Situação c) Um usuário não registrado, solicita a lista de melhores ofertas, utilizando um *smartphone* `{&aplicação.nome = ‘melhores ofertas’,`

&data=15/01/2015}. Considere a consulta original Q_2 a seguir. A regra R4 é disparada e a consulta é reescrita, então, para a forma:

```
Q2 =     SELECT l.nome, l.imagem_capa, l.preco, rv.qtd_venda
           FROM livro l, resumo_venda rv
          WHERE l.id_livro = rv.id_livro


Q2' =    SELECT l1.nome, l1.imagem_capa, l1.preco, rv1.qtd_venda
           FROM livro l1, resumo_venda rv1
          WHERE l1.id_livro = rv11.id_livro
            AND NOT EXISTS
                (SELECT *
                   FROM livro l2, resumo_venda rv2
                  WHERE l2.id_livro = b2.id_livro
                    AND l2.preco <= l1.preco
                    AND rv2.qtd_venda < rv1.qtd_venda
                     OR l2.preco<l1.preco
                    AND rv2.qtd_venda <= rv1.qtd_venda)
      ORDER BY l1.preco, rv1.qtd_venda;
```

4.4.9. **Modelo de Preferências e Diretiva Prefer**

Tendo em vista que a abordagem Texere tanto trata aspectos de adequação ao contexto quanto de personalização, faz-se necessário adotar um modelo que também possa refletir as preferências do usuário. Preferências pessoais são dependentes do contexto do usuário e a adaptação de consulta a essas preferências pode ser considerada como uma personalização (KOUTRIKA et al., 2013) da consulta. As preferências gerarão classificações mais refinadas do que as descritas na DRC ORDER e são atendidas pela DRC PREFER, cujo objetivo é realizar classificação sobre instâncias de atributos diversos, mas avaliados de forma combinada.

Conforme apresentado no Capítulo 3, há uma vasta literatura sobre preferências e contextualização de preferências (AMO; PEREIRA, 2011; KIEßLING et al., 2011; KOUTRIKA et al., 2013; LEVANDOSKI et al., 2010b). Neste trabalho, optou-se por usar uma estratégia baseada em parte do modelo de Koutrika e Ioannidis (2004, 2005), em virtude de se tratar de um modelo de preferências aplicado à linguagem SQL. No trabalho referenciado, as consultas ao banco de dados são reescritas de forma a se adequarem às preferências armazenadas nos perfis dos usuários.

O trabalho de Koutrika e Ioannidis (2004, 2005) define preferências sobre atributos de entidades do BD da aplicação sem considerar o contexto. Esta tese

difere do trabalho tomado como referência por incorporar no perfil dos usuários preferências também condicionadas à avaliação de elementos contextuais externos.

Perfis de Usuários

O esquema de perfis de usuários utilizados neste trabalho é formado de acordo com apresentado na Figura 4.11. Os perfis podem ser de dois tipos: a) de usuário (*user*) ou b) de grupo (*group*).

O perfil de usuário é dividido nos seguintes blocos:

a) Um primeiro bloco, marcado com o rótulo <*Profile*>, que contém informações sobre o tipo do perfil e a identificação do usuário;
b) Bloco com informação sobre participação do usuário em perfis de grupo, rotulado como <*Groups*>;
c) Bloco com uma sequência de preferências não dependentes de contexto, que utiliza o rótulo <*Preferences*>; e
d) Bloco rotulado como <*Contextual Preferences*>, que apresenta uma sequência de preferências dependentes do contexto.

As preferências do usuário são especificadas em relação a atributos de entidades do banco de dados e têm índices de preferência (IP) especificados com valores entre -1,0 e +1,0.

Figura 4.11 – Esquema de perfil de usuário

a)

```
<Profile>
type = user;
user_id: 102;

<Groups>
name = surdo;

<Preferences>
P1. Ator.nome = 'Chaplin' (0,7);
P2. Filme.dt_ano < 1960 (0,8);
P3. Filme.genero = 'Comédia' (0,7);
P4. Filme.genero = 'Terror' (-0,9);

<Contextual Preferences>
Acompanhante.tipo = 'esposa'
Cp1.  Filme.genero = 'romance' (0,8);
CP2.  Sessão.horario > 19h  (0,5);
Acompanhante.tipo = 'criança'
Cp3.  Filme.genero = 'animação' (0,8);
Cp4.  Estudio.nome = 'Disney' (0,9);
```

b)

```
<Profile>
type: group;
name: surdo;

<Preferences>
Gp1. Filme.legenda = 'sim' (1,0);
```

Fonte: O autor.

A definição desses índices deve ser feita pelo especialista no domínio e refletir critérios de preferência que indiquem -1,0 para rejeição extrema e +1,0 para preferência extrema. O índice 0 (zero) indica indiferença ao valor determinado para o atributo. Define-se o IP de uma preferência p, como:

$$\mathrm{IP}_{(p)} \in [-1 , +1]$$

Uma opção para especificar os IP, é por meio de respostas do usuário em formulários, tal qual usado no perfil apresentado na Figura 4.11, baseado nas afirmações do usuário de identificação 102: “Gosto do ator Charles Chaplin”, “Prefiro os filmes anteriores a 1960” e “Detesto filmes de terror”. Os predicados (e.g. gosto, gosto muito, prefiro, detesto, não me importo, sou fã) usados em relação a entidades e atributos são, então, mapeados para os IP utilizados na aplicação. No exemplo

apresentado a seguinte gradação é considerada: gosto muito (0,9); prefiro (0,8); gosto (0,7); aceito (0,5); não gosto (-0,7) e detesto (–0,9).

O estudo e formalização do mapeamento de predicados de preferência para IP não faz parte do escopo do presente trabalho. Estes valores devem ser definidos pelo especialista no domínio.

As preferências dependentes do contexto (*Contextual Preferences*) obedecem à formatação semelhante, mas se encontram agrupadas em função de uma restrição de elemento contextual externo. No perfil do exemplo (Figura 4.11-a) o usuário informou as seguintes preferências dependentes do contexto: “Estando acompanhado da minha esposa, prefiro assistir a romances à noite, já com as crianças prefiro o gênero infantil e nós gostamos muito dos estúdios Disney”. Nesse exemplo, `Acompanhante.tipo` é o elemento contextual externo. Havendo satisfação do elemento contextual externo, as respectivas preferências contextuais serão aplicadas na reescrita da consulta.

No modelo de preferências deste trabalho também foi introduzida a associação de usuários com perfis coletivos. Essa associação é feita pelo rótulo <*groups*> para facilitar o cadastro de preferências típicas de grupos de indivíduos. No exemplo apresentado, o usuário está associado ao grupo *surdo*, que referencia um perfil de grupo (Figura 4.11-b) que especifica uma preferência extrema para filmes legendados.

A utilização de perfis de grupo, evita retrabalho na composição de perfis. Na hipótese de haver conflito de regras sobre mesma entidade e atributo, prevalecerá a preferência descrita diretamente no perfil do usuário em detrimento da regra descrita no perfil coletivo, o mesmo acontece favorecendo preferências contextualizadas em detrimento de preferências não contextualizadas.

Da mesma forma que nas regras de produção, a base de elementos contextuais é usada como fonte de dados, para formação de preferências e preferências contextualizadas.

Diretiva PREFER

Esta diretiva tem por função determinar a aplicação de preferências segundo o perfil do usuário ou perfil de grupo de usuários. A diretiva PREFER possui a seguinte sintaxe:

> PREFER U
>
> Onde, U é a identificação do usuário (&USER.id) emissor da consulta Q.

As informações de perfil são usadas nas regras de produção por meio da DRC PREFER, com a qual é possível instruir a reescrita do comando Q com as preferências do usuário emissor, como no exemplo:

```
SE Q.filme
   ENTÃO PREFER (&usuario.id);
```

Apenas as preferências que incidem sobre atributos de entidades utilizadas na consulta ou em atributos de entidades relacionadas a estas serão consideradas para efeito de reescrita e cada preferência deverá ter seu índice de preferência (IP) usado como base para o cálculo do grau de importância desta preferência na consulta (GIC). A fórmula de cálculo do GIC é definida pela fórmula:

$$\mathrm{GIC}_{(p)} = |\mathrm{IP}_{(p)} * \mathrm{IPX}|$$

Onde, IPX é um índice de proximidade entre a entidade possuidora do atributo especificado na preferência p e as entidades presentes na consulta Q. o IPX deve ser definido pelo especialista no domínio para duas situações: um valor arbitrado caso a preferência p refira-se à entidade contida em Q ou relacionada diretamente com entidade contida em Q (relacionamentos tipo 1x1 ou 1xn) e outro valor para casos nos quais a preferência p refira-se à entidade relacionada indiretamente com entidade contida em Q (relacionamentos tipo nxm).

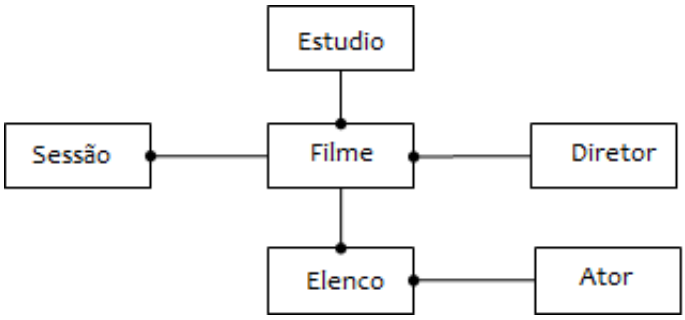
O objetivo do IPX é possibilitar que o ED atribua pesos diferentes para atributos de entidade presente na consulta, diretamente relacionadas e indiretamente

relacionadas com entidade base[31] da consulta. Por exemplo, considere o caso no qual um IP 0,9 (adequada) sobre o atributo `censura` para a entidade `filme` deve ter grau de importância numa consulta sobre filmes maior que o mesmo peso 0,9 (gosto muito) sobre o atributo da entidade relacionada `ator`. O ED pode arbitrar que o IPX para entidade diretamente ligada à entidade base tenha IPX igual a 1,0 e em caso contrário 0,9. Nesse caso o cálculo dos GIC da censura adequada será 0,9 * 1.0 = 0,9, tendo um peso maior na consulta que e o GIC de um ator do qual se goste muito (GIC = 0,9 * 0,9 = 0,81).

Como exemplos de cálculos de GIC, considere o perfil da Figura 4.11, o esquema relacional da Figura 4.12, IPX=1,0 para mesma entidade ou relacionada diretamente, IPX=0,9 para entidade relacionada indiretamente e a consulta a seguir:

Q: `SELECT nome FROM filme;`

Figura 4.12 – Esquema de dados de exemplo - filmes



Fonte: O autor.

Os GIC referentes às preferências, em relação à consulta Q, são:

$GIC_{(p1)}$ = |0,7 * 0,9| = 0,63; $GIC_{(p2)}$ = |0,8 * 1,0| = 0,8; $GIC_{(p3)}$ = |0,7 * 1,0| = 0,7;

$GIC_{(p4)}$ = |-0,9 * 1,0| = 0,9; $GIC_{(Cp1)}$ = |0,8 * 1,0| = 0,8; $GIC_{(Cp2)}$ = |0,5 * 1,0| = 0,5;

$GIC_{(Cp3)}$ = |0,8 * 1,0| = 0,8; $GIC_{(Cp4)}$ = |0,9 * 1,0| = 0,9. $GIC_{(Gp1)}$ = |1,0 * 1,0| = 1,0.

A estratégia de reescrita para a DRC PREFER obedece a uma adaptação da abordagem *Simply Personalized Answers* (SPA) (KOUTRIKA; IOANNIDIS, 2005), que estabelece a reescrita da consulta original para cada uma das preferências, obedecendo a suas restrições individualizadas, e depois realizando união entre elas.

[31] Entidade base da consulta é aquela que possui a chave primária no relacionamento com outra entidade. No modelo entidade-relacionamento é conhecida como entidade forte.

O resultado dessa união é finalmente agrupado por um atributo identificador e depois classificado segundo o grau de interesse das preferências na consulta.

Para exemplificar a utilização dessa estratégia, a consulta reescrita Q' do exemplo referente à Figura 4.12, em um caso no qual não foi inferido o contexto, ficaria da seguinte maneira:

```
SELECT nome, AVG(GIC)
  FROM (
      SELECT nome, 0.63 as GIC
        FROM filme f1, elenco e1, ator a1
       WHERE EXISTS
            (SELECT *
               FROM elenco e1, ator a1
              WHERE e1.id_ator = a1.id_ator
                AND e1.id_filme = f1.id_filme
                AND a1.ator_nome = 'Chaplin')
  UNION ALL
      SELECT nome, 0.8 as GIC
        FROM filme f2
       WHERE f2.dt_ano < 1960
  UNION ALL
      SELECT nome, 0.7 as GIC
        FROM filme f3
       WHERE EXISTS
            (SELECT *
               FROM genero g3
              WHERE g3.id_genero = f3.id_genero
                AND g3.nome = 'comédia')
  UNION ALL
      SELECT nome, 0.9 as GIC
        FROM filme f4
       WHERE EXISTS
            (SELECT *
               FROM genero g4
              WHERE g4.id_genero = f4.id_genero
                AND g4.nome <> 'terror')
  UNION ALL
      SELECT nome, 1.0 as GIC
        FROM filme f5
       WHERE f5.legendado = 'sim'
GROUP BY nome
HAVING COUNT(*)>= 2
ORDER BY AVG(GIC) DESC, COUNT(*) DESC;
```

Observe que a preferência com valor negativo -0,9 que indica rejeição tem seu GIC calculado pelo módulo (0,9) e implementação com operador diferente (< >). A restrição `HAVING COUNT(*)>= 2` faz com que apenas tuplas que satisfaçam pelo menos duas das preferências (codificadas em subconsultas distintas) sejam consideradas como preferências aceitas. Esse parâmetro de número mínimo de preferências aceitas também deve ser informado pelo especialista no domínio da

aplicação ou pelo gestor da aplicação e aplicado ao algoritmo de reescrita da DRC PREFER.

Como exemplo de uma consulta que atenda ao contexto, suponha que o usuário do exemplo anterior, agora esteja planejando assistir filmes com os filhos pequenos. A consulta é reescrita obedecendo às preferências adequadas ao elemento contextual externo `Acompanhante.tipo = 'criança'` (preferências C_{p3} e C_{P4} da Figura 4.11-b), com Q' resultante:

```
SELECT *, AVG(GIC)
  FROM (
      SELECT nome, 0.8 as GIC
        FROM filme f1
       WHERE f1.genero = 'animação'
  UNION ALL
      SELECT nome, 0.9 as GIC
        FROM filme f2
       WHERE EXISTS
            (SELECT *
               FROM studio e2
              WHERE e2.id_estudio = f2.id_estudio
                AND e2.nome = 'Disney')
  UNION ALL
      SELECT nome, 0.9 as GIC
        FROM filme f3
       WHERE f3.genero <> 'terror'
  UNION ALL
      SELECT nome, 1.0 as GIC
        FROM filme f4
       WHERE f4.legendado = 'sim'
GROUP BY nome
HAVING COUNT(*)>= 2
ORDER BY AVG(GIC) DESC, COUNT(*) DESC;
```

Observe que o conflito entre preferências pelo gênero do filme (em atributo e operador de igualdade) provocou que a preferência dependente do contexto Cp_3 prevalecesse em detrimento da preferência não dependentes do contexto P_3.

Na estratégia de implementação da DRC PREFER foi adotada a classificação do resultado em ordem decrescente da média dos GIC (`AVG(GIC)`) individuais relativas às preferências e tendo como segundo argumento de classificação a quantidade de preferências atendidas pela tupla (`COUNT(*)`). Outra opção feita, em favorecimento do desempenho, é a especificação de subconsultas com a cláusula SQL EXISTS, que permite uma velocidade de processamento mais rápida, em especial para tabelas com volumes de dados maiores, uma vez que encontrada uma tupla que satisfaça a condição de busca, a subconsulta é interrompida sem ter que buscar todas as tuplas que a satisfazem (a exemplo do que faz a operação de junção).

A DRC PREFER tem a seguinte definição:

Definição 11 - DRC Prefer: Considere Q uma consulta SQL inicial sobre uma relação R:

Dado: Q: $\pi_{\alpha}[\sigma_{\phi}](R)$;

$UP = \{Up_1, Up_2, \ldots, Up_n\}$, o conjunto de preferências p_i do usuário U;

Up_i é uma preferência no formato $Up_i = E.a_i$ <oc>$_i$ <vl>$_i$, no qual <oc>$_i$ é um operador condicional <oc> ∈ { =, >, < , <= , >=, < >} e vl_i é um valor instanciado;

$UpQ = \{Up_1Q, Up_2Q, \ldots, Up_nQ\}$, o conjunto de preferências p_i do usuário U aplicáveis à consulta Q (preferências sobre atributos da consulta Q ou relacionados);

$GIC_{UQ} = \{GIC_{1UQ}, GIC_{2UQ}, \ldots, GIC_{nUQ}\}$, o conjunto de graus de interesse das preferências do usuário U para a consulta Q;e

kp, o número mínimo de preferências que Q' deve atender.

Diz-se que: PREFER U (Q) ⟶ Q' | Q': $\tau_T \gamma_{GIC} \pi_{\alpha'} \sigma_{\phi'} (R_1 \cup R_2 \ldots \cup R_n)$ (1)

Onde, $\alpha' = (\alpha, AVG(GIC))$ (2)

$\phi' = COUNT(*) >= kp$ (3)

$T = AVG(GIC)\ DESC, COUNT(*)\ DESC$ (4)

γ_{GIC} é uma agregação sobre α e função de agregação $\gamma = AVG\ (GIC)$; e

R_i é uma relação derivada de R, aplicando-se a restrição Up_iQ e grau de interesse na consulta GIC_{UQi}, com formato:

$R_i: \pi_{\alpha'}[\sigma_{\phi'}](R)$ (5)

Onde $\alpha' = (\alpha, GIC_{UQi})$; e (6)

$$\phi' = \begin{cases} \phi\ AND\ UpiQ, & caso\ E.ai \in R \\ EXISTS\ SC(UpiQ), & em\ caso\ contrário. \end{cases} \quad (7)$$

EXISTS SC(Up_iQ) denota a filtragem pela subconsulta com cláusula SQL EXISTS para a condição informada em Up_iQ.

A definição da diretiva PREFER diz que a consulta original Q deve ser reescrita para outra em formato Q' que deve ser o resultado de uniões (U) de

consultas parciais R_i (1). Cada consulta parcial R_i (5) diz respeito à reescrita de Q em função de uma preferência do usuário aplicável a Q ($U_{p_i}Q$) e deve ter sua projeção acrescida do seu GIC (GIC_{UQi}) correspondente (6). Caso o atributo referido nesta preferência seja de entidade que esteja em Q, deve-se acrescentar a condição da preferência da consulta parcial (cláusula WHERE), em caso contrário deve-se utilizar subconsulta com a cláusula SQL EXISTS (7).

A consulta principal, resultante das uniões das consultas parciais R_i, deve realizar agregação (γ) dos atributos projetados em Q em função da média dos GIC de consultas parciais (2). Deve obedecer à filtragem de tuplas que atendam um número mínimo kp de consultas parciais (3) (cláusula SQL HAVING) e ordenar (cláusula SQL ORDER BY) o resultado de acordo com a média dos GIC e número de preferências atendidas (número de vezes que uma tupla foi resultado de consulta parcial) de forma decrescente (4).

O pseudocódigo do algoritmo de aplicação da DRC PREFER encontra-se listado no Apêndice A.

4.5. Análise comparativa

No Capítulo 3 foi apresentado um levantamento do estado da arte em pesquisas sobre SGBD e contexto, com foco em áreas relacionadas com este trabalho: modelagem de contexto para SGBD, consultas sensíveis ao contexto e reescrita de consultas SQL. O

Quadro 4.4 apresenta um resumo das principais características de cada trabalho pesquisado, como objetivo de apontar diferenças e similaridades com a Abordagem Texere. Os critérios de comparação apresentados são:

- Abordagem - identifica a abordagem utilizada, correlata ao uso de contexto ou aspecto desenvolvido neste trabalho;
- Modelagem do contexto - indica o tipo de modelo de representação do contexto utilizado;
- Recursos usados - resumo dos principais recursos utilizados no trabalho, correlatos ao uso de contexto ou aspecto desenvolvido neste trabalho;
- Inferência do contexto - informa se há inferência do contexto no trabalho e indica a forma como é feita; e

- SGBDR - aponta se a solução proposta utiliza SGBD relacional de forma convencional.

Em relação às propostas de modelos de dados que dão suporte a SGBD que incorporem a noção de contexto, pode-se dizer que a presente proposta é mais flexível que as abordagens que utilizam mecanismos de visões para moldar os dados mediante contextos identificados (BOLCHINI et al., 2007b), uma vez que todo o processo de composição de visões deve ser realizado em tempo de projeto e essas ficam associadas a contextos pré-determinados.

Já em relação aos trabalhos de Henricksen et al. (2003) e Henricksen e Indulska (2006), o vínculo à modelagem do negócio em ORM, e posterior mapeamento em visões relacionais, restringe a solução para aplicações concebidas nesse modelo. O trabalho de Roussos e Sellis (2008) propõe a extensão do modelo relacional com operadores algébricos especiais para manipulação de dados contextuais, o que distancia essa abordagem deste estudo. Da mesma forma, os trabalhos que focam na modelagem do contexto auxiliada por estruturas de dados hierárquicas e reclassificação de respostas (STEFANIDIS et al., 2007) utilizam técnicas de inferência e sensibização de consultas ao contexto que não preservam a consulta em SQL, ao contrário do proposto nesta tese.

Em Levandoski et al. (2011), a técnica utilizada pode ser considerada híbrida, uma vez que a consulta é reescrita para considerar preferências e contexto. A consulta é submetida ao SGBD e depois tem sua resposta reclassificada de acordo com um *ranking* mais apropriado de forma *built-in*. Nesse trabalho, assim como no estudo de Kießling et al. (2011), é utilizado um módulo intermediário entre a aplicação e o SGBD, tal qual a arquitetura aqui apresentada.

Em Amo e Pereira (2011), preferências são incorporadas ao BD e utilizadas como argumento de consultas, embora não haja inferência do contexto e as preferências devam ser armazenadas previamente e atreladas a relações do BD pré-definidas. As consultas aos dados utilizam operadores que realizam classificação de tuplas mais interessantes mediante as preferências determinadas em uma cláusula de preferências associada à consulta.

Com a abordagem de Bunningen (2008), este trabalho tem a similaridade de ter uma base de conhecimento com suporte em motor de inferência, para realizar a identificação do contexto e preferências de usuários. Ambos os trabalhos realizam

consultas em SQL, embora o trabalho referenciado necessite que o banco de dados relacional a ser consultado tenha seu modelo de dados atrelado ao modelo lógico definido na base de conhecimendo, o que torna inflexível e pouco prática a sua utilização para consultas a bancos de dados relacionais.

Dos trabalhos focados em preferências, o mais correlacionado a este e do qual foi adotada estratégia de reescrita para preferências é o de Koutrika e Ioannidis (2005). A principal diferença do referido trabalho para outros dirigidos a preferências é que estas são relativas a entidades, atributos e relacionamentos entre entidades (assinalada no quadro como quantitativa baseada em atributo), ao contrário das abordagens *quantitativa* e *qualitativa* de consultas com preferências, que buscam a comparação entre tuplas de uma relação.

Os trabalhos de Yaguinuma (2007), Vilar (2008) e Mishra e Koudas (2009) da mesma forma que este, propõem reescrita de SQL, embora as técnicas utilizadas limitem-se à incorporação de termos semelhantes aos usados na consulta original e expansões ou relaxamento de consultas por meio da flexibilização de cláusulas de restrição. Em outro sentido, este trabalho apresenta a reescrita de consultas para fins de adequação ao contexto e personalização e contempla técnicas de reescrita mais complexas, que utilizam subconsultas, ordenações, agregações e classificacões.

Por fim, não foram encontradas na literatura consultada, propostas de especificação de diretivas de reescritas de consultas em SQL nem tampouco utilização de regras de produção para inferência de contexto de consultas a bancos de dados.

Quadro 4.4 – Comparativo de características entre trabalhos correlatos

Autor/Projeto /Ano	Abordagem	Modelagem do Contexto	Recursos usados	Inferência do contexto	SGBDR
Bolchini et al. /CARVE/ 2013	Mecanismo de visões	Gráfica (*ctx. tree*)	Visões de dados	Não	Sim (mapeamento)
Stefanidis et al./ *Contextual Preference Model* /2007~2011	Preferências quantitativas	Gráfica (grafos e hierarquias)	Estruturas de dados auxiliares / análise algorítmica de preferências / Relaxamento de consultas/ Classificação de resultados / Implementação *on-top*	Composição de parâmetros	Não
Roussos e Sellis,/2008	Extensão do modelo relacional	Relacional estendida	Construtores e operadores em álgebra relacional estendida	Não	Extensão
Elmongui /Chameleon/ 2009	Extensão do modelo objeto-relacional	Orientada a objetos	Consulta tipo *skyline*	Composição de parâmetros	Não
Bunningen /CIM/2008	Preferências quantitativas e contexto assistido por base de conhecimento	Regras em lógica de descrição	Algorítmos de classificação / Aferição de relevância de preferências / Motor de inferência / Base de conhecimento em lógica descritiva mapeada para BD relacional / Implementação *built-in*	Regras em lógica de descrição	Extensão
Henricksen et al. /CML/ 2003~2006	Mapeamento ORM para Relacional	Gráfica (ORM)	Mapeamento RMAP e geração de visões de dados	Não	Não
Levandoski et.al./ CareDB/2010	Preferências qualitativas	Não	Tradução de consultas e perfis de preferências / Implementação *built-in*	Composição de parâmetros e perfis de usuários	Não

continua...

continuação do Quadro 4.4

Autor/Projeto /Ano	Abordagem	Modelagem do Contexto	Recursos usados	Inferência do contexto	SGBDR
Kießling et al. / Preference SQL/ 2002~2011	Preferências qualitativas e quantitativas	Não	Tradução para SQL e posterior classificação de resultados / Extensão do SQL, regras de preferência e reformulação de respostas / Middleware	Não	Não
De Amo et al. / CPrefSQL/ 2011	Preferências qualitativas	Não	Implementação *On-top* ou *Built-in*	Extensão do SQL com preferências armazenadas e operadores de consulta	Não
Koutrika e Ioannidis / *Genealized Preference Model/ 2004*	Preferências quantitativas sobre atributo / Reescrita de SQL	Não	Reescrita de consultas mediante preferências em perfis de usuários / Expansão de consultas e classificação de resultados	Não	Sim
Yaguinuma 2007	Processamento semântico/ Reescrita de SQL	Não	Ontologia de domínio / Expansão de consultas SQL	Não	Sim
Vilar 2008	Processamento semântico/ Reescrita de SQL	Não	Ontologia de domínio / Expansão de consultas SQL	Não	Sim
Mishra e Koudas 2009	Refinamento semântico/ Reescrita de SQL	Não	Estruturas de dados hierárquicas algoritmos para expansão ou contração de consultas	Não	Sim
Texere	Reescrita de SQL/ Preferências quantitativas baseada em atributo e contexto	Orientada a objetos	Reescrita de consultas mediante contexto / Motor de inferência / Base de conhecimento em regras de produção / Middleware	Regras de produção	Sim

Fonte: O autor.

4.6. Considerações

Conforme apresentado no Capítulo 3, SGBD que lidem com dados contextualizados são vistos como ferramentas importantes para preencher a lacuna existente entre os SGBD tradicionais e as aplicações computacionais que demandam, cada vez mais, informações adaptadas ao contexto. Neste trabalho, busca-se alternativa que leve à utilização do dado contextual e sua semântica, e ao mesmo tempo viabilize o uso dos SGBD relacionais, amplamente adotados no mercado.

A Abordagem Texere teve sua concepção baseada em uma arquitetura cujos componentes capturam informação contextual tanto da aplicação que faz uso do SGBD alvo, como do usuário e do ambiente externo. Um componente responsável pela inferência contextual usa regras de produção para inferir o contexto que está em torno de uma consulta original *Q*. As informações sobre o contexto irão guiar alterações no código dessa consulta submetida, com o intuito de restringí-la, adaptá-la ou estendê-la para uma forma mais adequada e precisa (uma consulta reescrita *Q'*).

Esta pesquisa focou duas áreas principais: o modelo de contexto, na qual a informação contextual é incorporada ao banco de dados com a finalidade da caracterização do dado convencional e as consultas que são concebidas para lidar com a informação de forma sensível ao contexto. Procurou-se realizar um levantamento das técnicas de consulta a dados que, de alguma forma, representassem adaptação ao contexto. Isso levou a buscar uma solução que pudesse abranger as duas áreas. Em virtude do objetivo inicial de dotar os BD relacionais de mecanismo para responder a consultas de forma sensível ao contexto, este trabalho foi desenvolvido com foco na reescrita de consultas SQL.

5. IMPLEMENTAÇÃO E EXPERIMENTOS

Neste capítulo são descritos os experimentos realizados com a abordagem Texere. São apresentados o propósito, as hipóteses, os objetivos desses experimentos e o protótipo elaborado. Também são apresentadas as métricas empregadas e analisados os resultados obtidos.

5.1. Plano de experimentação

Nesta subseção é definido o que é esperado do experimento, objetivos e hipóteses.

5.1.1. Relevância e o Propósito do Experimento

Os experimentos concebidos têm como propósito geral avaliar se consultas reescritas de forma a considerar os contextos sob os quais foram emitidas apresentam respostas mais relevantes para o usuário.

Relevância é definida como a particularidade do que é relevante, onde relevante é definido como a qualidade de algo "*intimamente ligado ou apropriado para o que está sendo feito ou considerado.*" ou "*adequado ao tempo, período, ou circunstâncias correntes; de interesse contemporâneo*" (OXFORD DICTIONARY 2015, tradução nossa).

A relevância é considerada um conceito intuitivo, de entendimento universal e noção chave para a área de Recuperação da Informação (SARACEVIC, 1975; 2007a). Shamber et al. (1990) concluíram que a natureza da relevância apresenta as seguintes visões, para sistemas de informação:

- Relevância é um conceito cognitivo multidimensional cujo significado é amplamente dependente da percepção dos usuários a respeito da informação e de suas situações de necessidade de informação;
- Relevância é um conceito dinâmico que depende do julgamento dos usuários a respeito da qualidade do relacionamento entre informação e necessidade de informação em determinado período; e

- Relevância é um conceito complexo, mas sistemático e mensurável se aproximado conceitualmente e operacionalmente da perspectiva do usuário.

Nos conceitos citados, percebe-se a importância da avaliação dos usuários para a mensuração da relevância. Neste trabalho, conceituamos que uma resposta é relevante se ela está apropriada ao contexto envolvido na consulta (vide Seção 4.3.3), levando em consideração os elementos contextuais da aplicação, do usuário, do dispositivo e dos ambientes físico e computacional.

5.1.2. **Hipóteses**

O processo de experimentação tem como objetivo realizar observações, medições e análises, a fim de provar ou refutar hipóteses formuladas. Para cada hipótese, resultados quantitativos serão apresentados para fundamentar se uma hipótese é aceita.

Neste trabalho, definimos as seguintes hipóteses:

H1. É possível utilizar técnicas de reescrita como forma de conferir sensibilidade a contexto em consultas a bancos de dados relacionais.

H2. Consultas que considerem o contexto sob o qual foram realizadas tendem a propiciar respostas mais relevantes para seus usuários.

H3. Consultas reescritas com utilização de preferências independentes e dependentes do contexto tendem a produzir resultados mais relevantes para o usuário, do que se forem utilizadas apenas preferências independentes do contexto.

5.1.3. **Objeto de Medição**

O objeto de investigação e medição que propiciará os resultados necessários para avaliação das hipóteses é formado pelo conjunto de avaliações de usuários para respostas a consultas a bancos de dados e consultas reescritas, nas seguintes formas:

- **Consulta Q**: consulta em sua forma original conforme emitida para o BD da aplicação;

- **Consulta Q'**: é a consulta Q, reescrita mediante a inferência do contexto utilizando apenas elementos contextuais (EC) internos; e
- **Consulta Q''**: é a consulta Q, reescrita mediante a inferência do contexto utilizando EC internos e externos (vide Seção 4.3.1).

5.2. Implementação do Protótipo

Um protótipo da Arquitetura Texere foi desenvolvido para avaliar as hipóteses deste trabalho. Para realização de testes e avaliações, também foi elaborada uma aplicação denominada Texere *Movie Database* (TMDb) capaz de fornecer sugestões de filmes para usuários que se cadastrem e preencham seus perfis de preferências sobre o tema.

5.2.1. Plataforma de Desenvolvimento

Como plataforma de desenvolvimento, foi utilizada a Linguagem Java[32] e a interface de desenvolvimento Java Eclipse[33]. Todos os bancos de dados envolvidos (BD da Aplicação e BD de Elementos Contextuais) foram criados usando o SGBD PostgreSQL[34] versão 9.1. O componente de conexão com o BD foi construído com bibliotecas Java JDBC CAPI[35]. Todas as conexões usadas para os diversos bancos de dados da arquitetura são criadas e gerenciadas por este componente. A codificação de reescrita de consultas foi desenvolvida para a linguagem SQL padrão ANSI/ISO (SQL ISO 1992, 2003, 2008).

No protótipo, foi utilizado o motor de inferência JBoss Drools[36], de forma adaptada, para processar as regras contextuais criadas pelo Especialista no Domínio. O JBoss Drools torna possível programar regras declarativas de negócio e mantê-las atualizadas de forma dinâmica, visto que separa as regras do código fonte da aplicação. O JBoss Drools também oferece alto desempenho e uma plataforma escalável.

---

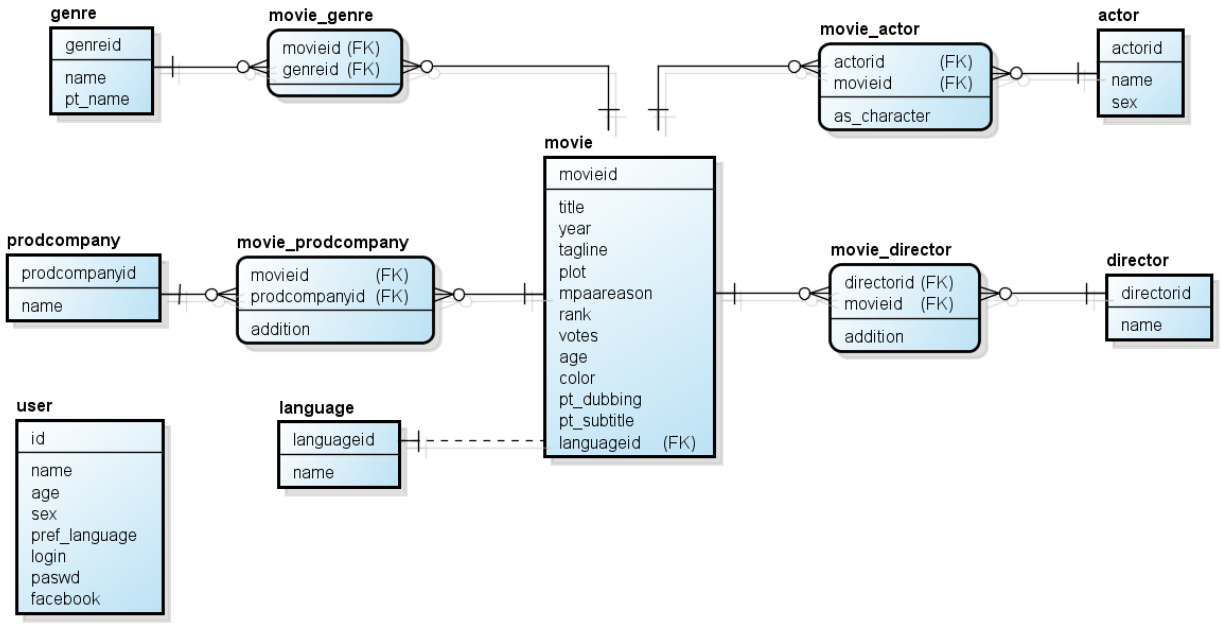
[32] Java, disponível em: http://www.java.com/. Acesso em: 15 dez. 2014.
[33] Eclipse, disponível em: http://www.eclipse.org/downloads/moreinfo/java.php. Acesso em: 15 dez. 2014.
[34] PostgreSQL, disponível em: http://www.postgresql.org/. Acesso em: 15 dez. 2014.
[35] JDBC, disponível em: http://www.oracle.com/technetwork/java/javase/tech/index-jsp-136101.html. Acesso em: 15 dez. 2014.
[36] JBoss Drools, disponível em: http://www.jboss.org/drools/. Acesso em: 15 dez. 2014.

5.2.2. **Amostra de Dados**

Como base de dados para consultas, foi confeccionado um banco de dados baseado no IMDb (*Internet Movie Database*[37]), um sítio WEB que mantém o maior acervo de informações sobre filmes e séries de televisão disponível na Internet.

A amostra de dados utilizada, um subconjunto do IMDB, conta com cerca de 27 mil filmes (*movie*), classificados em 22 gêneros (*genre*), contando com 470 mil atores (*actor*), 11 mil diretores (*director*) e 25 mil estúdios (*prodcompany*). Os dados extraídos em formato textual foram formatados e armazenados em um banco de dados relacional, cujo modelo está apresentado na Figura 5.1, em notação IE (*Information Engineering*) (MARTIN, 1989).

Figura 5.1– Modelo relacional do banco de dados de experimento



Fonte: O autor.

5.2.3. **Aplicação TMDb**

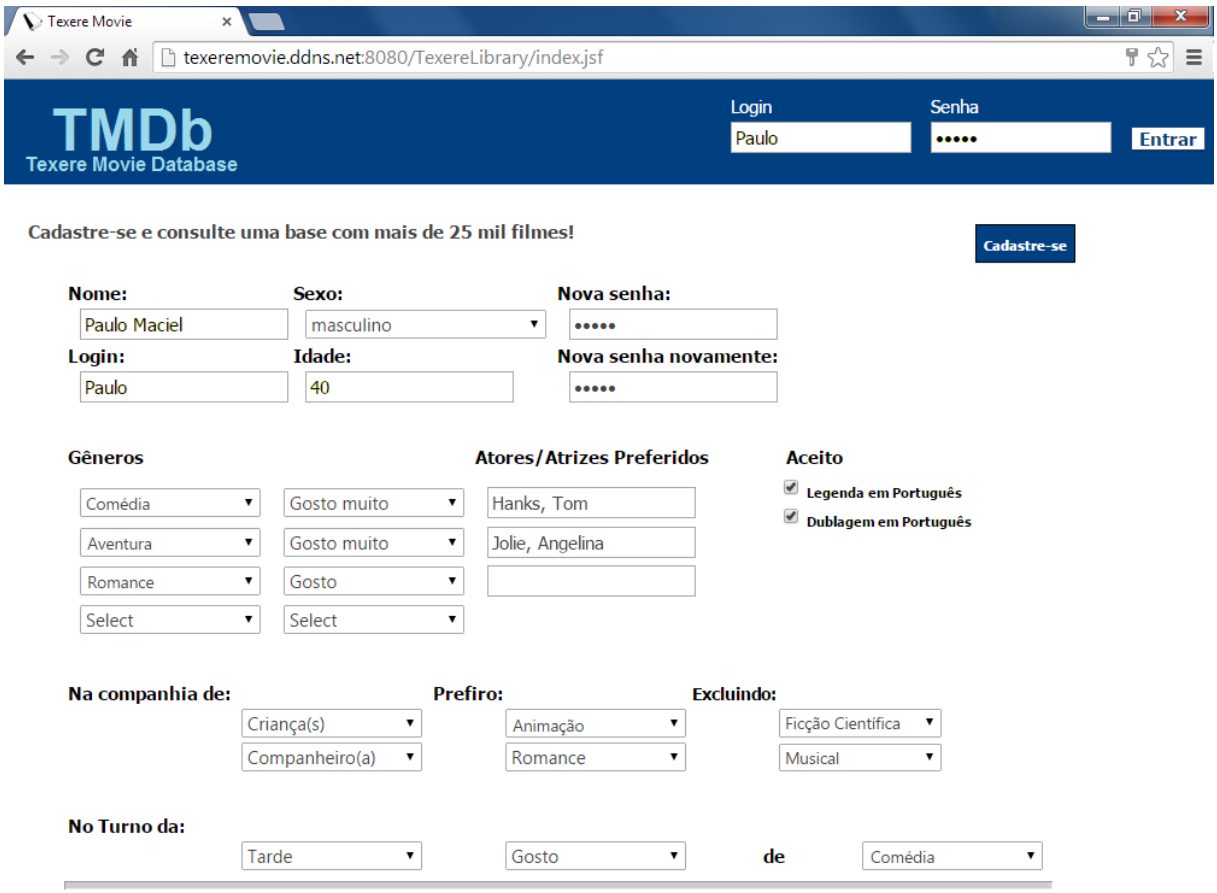
O protótipo desenvolvido trata-se de uma aplicação para ambiente WEB que permite que usuários se cadastrem e realizem consultas à base de dados da aplicação TMDb. Os usuários também são convidados a informar suas preferências

[37] IMDb, *Internet Movie Database*, disponível em: http://www.imdb.com/. Acesso em: 15 dez. 2014.

por atores ou atrizes e gêneros de filmes. Em relação aos gêneros, também é solicitada informação de opções preferidas, em função da companhia (e.g. filhos, esposa, amigos) e horário (e.g. manhã, tarde, noite) que esteja planejando uma sessão de apresentação do filme.

A Figura 5.2 mostra a tela de cadastramento de usuários. Nesta tela de exemplo, o usuário Paulo informa que gosta muito dos gêneros Comédia e Aventura e gosta de Romance. Prefere os atores Tom Hanks e Angelina Jolie. Na companhia de crianças prefere assistir filmes de animação que não sejam do gênero Ficção Científica e na companhia da companheira prefere Romance que não seja Musical. Para o turno da tarde, o usuário tem predileção por comédias.

Figura 5.2 – Tela de cadastramento de usuários da aplicação TMDB



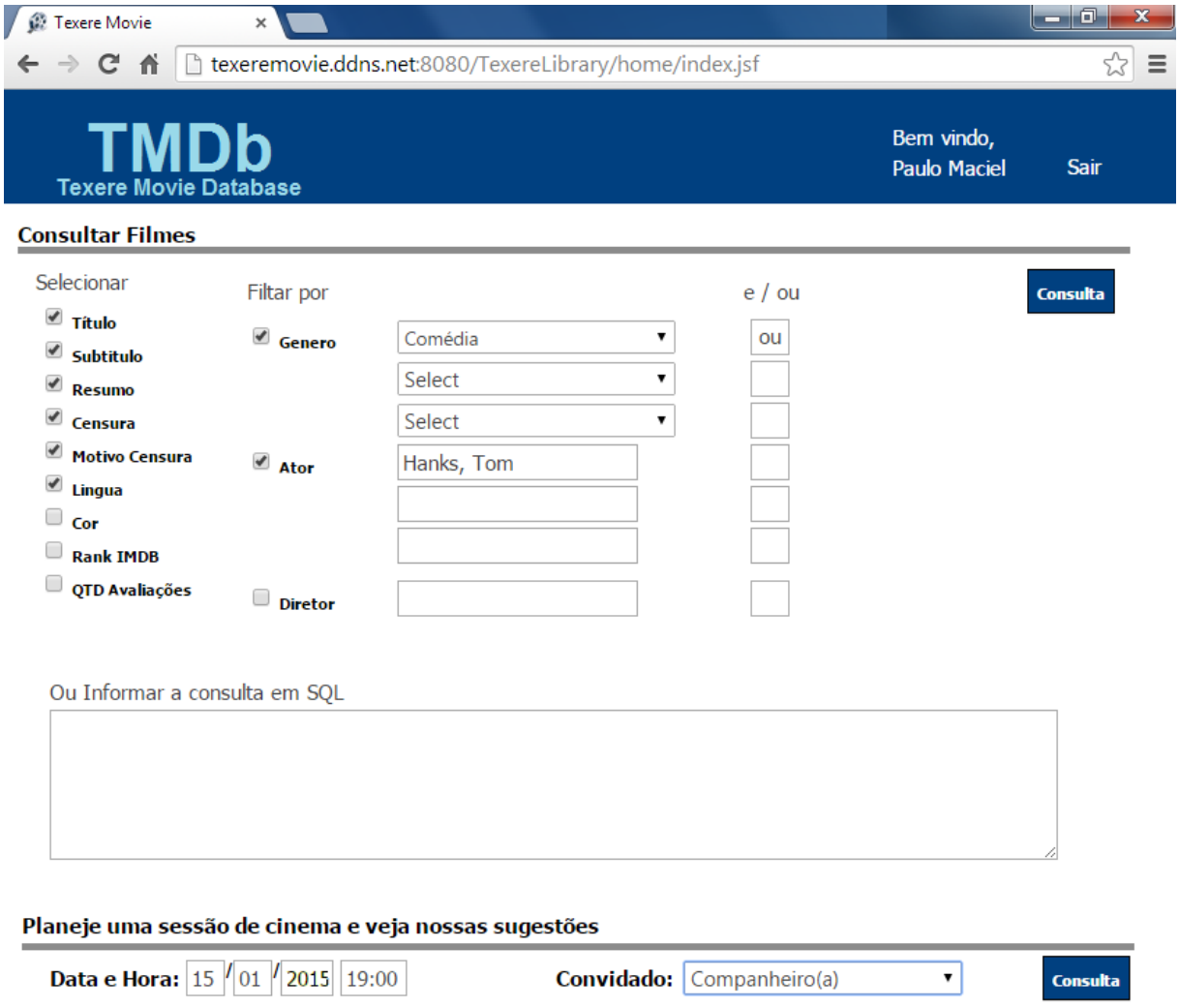
Fonte: O autor.

No cenário de testes, o objetivo da aplicação é que os usuários realizem uma consulta sobre filmes e avaliem as respostas correspondentes à consulta original Q e consultas reescritas Q' (consulta Q reescrita considerando elementos contextuais

internos) e Q'' (consulta Q reescrita considerando elementos contextuais internos e externos).

Para este fim, é disponibilizada uma interface que permite a formatação de consultas em SQL, conforme apresentado na Figura 5.3. O usuário seleciona as informações sobre filmes que deseja que apareçam na sua consulta, e filtros sobre gêneros, atores/atrizes e diretores. Adicionalmente o usuário é convidado a planejar a sessão de exibição para a qual está solicitando sugestões de filmes. Este planejamento é composto pelas informações sobre acompanhantes e dia e horário da sessão, preenchidos na parte inferior da tela de consultas (Figura 5.3). A caracterização da sessão de apresentação do filme tem por objetivo simular a captura de elementos contextuais externos.

Figura 5.3 – Tela de consulta da aplicação TMDB



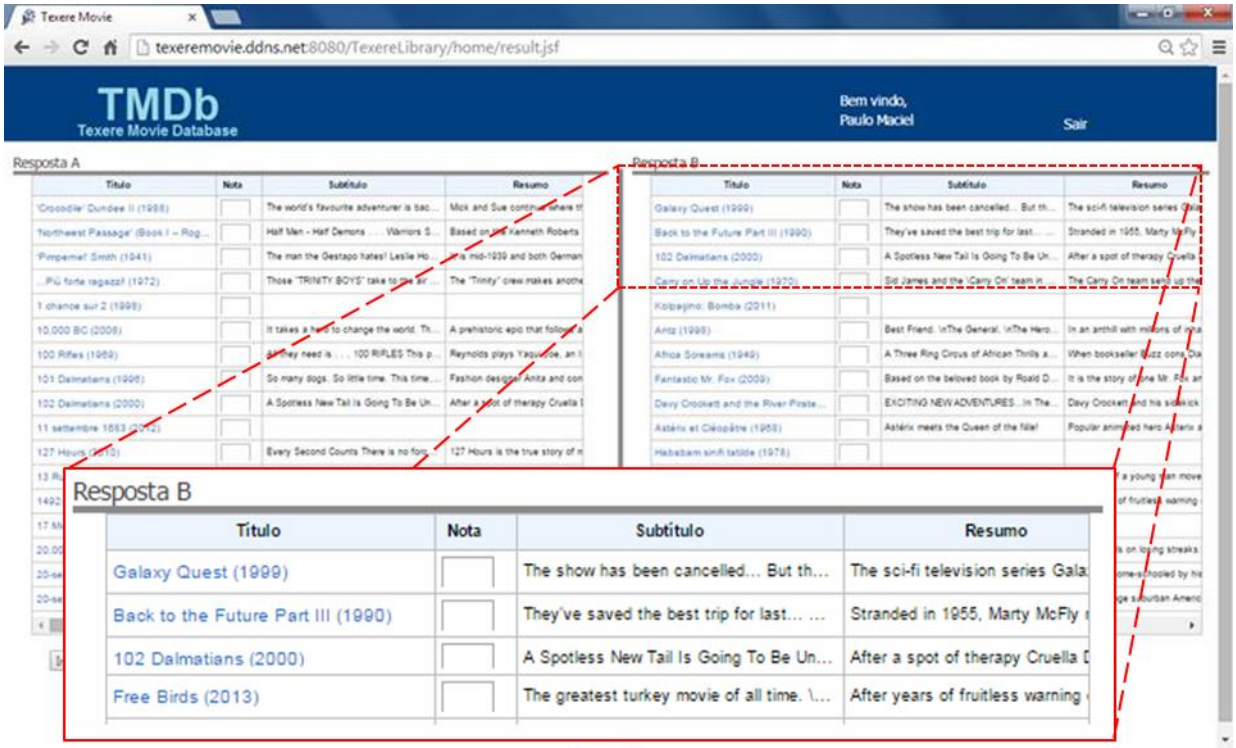
Fonte: O autor.

Após essas informações serem preenchidas, a consulta ao banco de dados da aplicação TMDb pode ser realizada. No experimento, foram utilizados painéis lado-a-lado (*side-by-side panels*) (BAEZA-YATES; RIBEIRO NETO, 2010) para a exibição das respostas das consultas. Isso possibilita avaliar, de forma comparativa, os resultados entre consultas, evitar diferenças de opinião para mesmos itens em painéis diferentes e controlar influências sobre a opinião do usuário produzidas pela ordenação dos resultados.

Por motivos de usabilidade, as três respostas (Q, Q' e Q'') não são apresentadas simultaneamente. Os dois botões de consulta existentes na tela (Figura 5.3) exibem os resultados das consultas em dois pares de respostas lado a lado. O botão superior exibe as respostas Q e Q' e o botão inferior, as respostas Q e Q''.

As respostas são apresentadas para os usuários conforme tela da **Figura 5.4**.

Figura 5.4 –Tela de consulta TMDb e destaque da avaliação de filmes



Fonte: O autor.

Cada painel exibido é formado por vinte (20) linhas de respostas e em cada linha há um campo no qual os usuários são convidados a expressarem sua opinião acerca da relevância daquela sugestão de filme, de acordo com suas preferências

e/ou adequação ao contexto da sessão de exibição planejada. O acionamento do segundo botão de consulta repete as respostas da consulta Q no painel à esquerda, o que exige que o usuário complete apenas a avaliação das respostas de Q''.

Os usuários não foram informados sobre detalhes técnicos de reescrita e nem sobre as características das sugestões exibidas em cada painel. Foram dadas apenas orientações para manter a mesma nota, caso um filme conste em mais de um painel.

Ao final da sessão de avaliações, cada usuário teve registrada sua opinião sobre a adequação de três conjuntos ordenados de vinte (20) filmes cada, correspondentes às respostas das consultas Q, Q' e Q''.

5.2.4. **Público-alvo e Critérios de Avaliação**

Inicialmente um teste piloto foi realizado com quatro usuários com o objetivo de identificar problemas de execução da aplicação, possibilidades de interpretação equivocada do usuário em relação ao objeto de avaliação, problemas de usabilidade e inconsistências de dados. Após a correção dos pontos de erro e realização de ajustes para melhorias, a bateria de testes foi aplicada a vinte (20) usuários não participantes do teste piloto.

Para execução dos testes foram selecionados usuários de perfil variado, com idade entre 15 e 49 anos, estudantes ou profissionais liberais, familiarizados com o ambiente de aplicações para Internet.

Os usuários foram convidados a informar uma nota sobre a relevância de cada um dos vinte (20) primeiros filmes exibidos, conforme a seguinte gradação arbitrária:

3 - muito relevante, gostei muito;
2 - relevante, gostei;
1 - pouco relevante, indiferente;
0 - irrelevante, não gostei.

5.3. Experimento

Nesta seção são descritas as métricas utilizadas para avaliação dos resultados obtidos com a realização do experimento e as regras de produção e DRC utilizadas no protótipo. A título de exemplificação do processo de reescrita, é detalhado o caso de um usuário real do experimento.

5.3.1. Métricas Empregadas

Desde os estudos de Kent et. al (1955), a relevância tornou-se e manteve-se o critério base para a medição da eficácia para recuperação de informação e *precisão* e *revocação* são as medidas padrão para medir a relevância. Essas medidas indicam a probabilidade de concordância entre o que um sistema recuperou ou elaborou como relevante (relevância do sistema) e o que o usuário avaliou como relevante (relevância do usuário) (SARACEVIC 2007a).

Precisão é a razão das tuplas recuperadas que são relevantes para uma dada consulta em relação ao total de tuplas retornadas na consulta, enquanto que revocação é a razão entre o número de tuplas recuperadas que são relevantes para uma consulta e o total de tuplas no banco de dados que são relevantes para essa consulta.

Em virtude da impossibilidade dos usuários avaliarem todo o conteúdo do banco de dados, a medida revocação e suas variações não foram utilizadas no experimento. As medidas utilizadas nestes experimentos são: Precisão em n (P@n), *Mean Average Precision* (MAP) e *Discounted Cumulative Gain* (DCG) (BAEZA-YATES; RIBEIRO NETO, 2010), descritas a seguir.

Precisão em n (P@n)

A medida P@n mede a relevância dos n primeiros documentos de uma lista ordenada:

$$P@n = \frac{r}{n}$$

Onde: n é o número de tuplas retornadas; e

r é o número de tuplas consideradas relevantes e retornadas até a posição n da lista ordenada.

Por exemplo, se as 10 primeiras tuplas retornadas por uma consulta são {relevante, irrelevante, irrelevante, relevante, relevante, relevante, irrelevante, irrelevante, relevante, relevante} então os valores de P@1 até P@10 são $\{1, 1/2, 1/3, 2/4, 3/5, 4/6, 4/7, 4/8, 5/9, 6/10\}$, respectivamente.

A medida P@n foi utilizada nos experimentos como forma de obter uma visão geral das avaliações dos usuários sobre a relevância das consultas realizadas.

Mean Average Precision (MAP)

Para um conjunto de consultas, deve-se calcular a média dos P@n. MAP é uma medida que tenta sumarizar todos os valores P@n e baseia-se na *Average Precision* (AP), uma média dos valores de P@n para todas as tuplas relevantes, definida como:

$$AP = \frac{\Sigma_{n=1}^{N} P@n \; x \; rel(n)}{r_q}$$

Onde: r_q é o número total de tuplas consideradas relevantes para a consulta;

N é o número de tuplas recuperadas; e

$rel(n)$ é uma função binária sobre a relevância da n-ésima tupla:

$$rel(n) = \begin{cases} 1, & \text{se a n} - \text{ésima tupla é relevante;} \\ 0, & \text{em caso contrário.} \end{cases}$$

No exemplo anterior:

$$\begin{aligned} \text{AP} = \; & ((1\text{x}1) + (0{,}5\text{x}0) + (0{,}33\text{x}0) + (0{,}5\text{x}1) + (0{,}6\text{x}1) + (0{,}66\text{x}1) + \\ & (0{,}57\text{x}0) + (0{,}5\text{x}0) + (0{,}55\text{x}1) + (0{,}6\text{x}1)) \; / \; 6 \\ = \; & (1 + 0 + 0 + 0{,}5 + 0{,}6 + 0{,}66 + 0 + 0 + 0{,}55 + 0{,}6) \; / \; 6 \\ = \; & 3{,}91 \; / \; 6 \\ = \; & 0{,}65 \end{aligned}$$

Assim sendo, o valor de MAP é a média dos valores AP para todas as consultas realizadas.

$$MAP = \frac{\Sigma_{t=1}^{T} AP(t)}{T}$$

Onde: T é o número consultas realizadas.

A medida MAP foi selecionada para este experimento como forma de sumarização do perfil da avaliação sobre as respostas de conjuntos de consultas de mesmo perfil (i.e. consultas Q, Q' e Q'').

Ganho Cumulativo Descontado (DCG)

As medidas P@n e MAP apenas conseguem tratar situações nas quais o nível de relevância é binário (relevante ou não relevante) e diferenças entre métodos para recuperação de informação ótimos e métodos bons pode não ser percebida.

A medida do Ganho Cumulativo Descontado (DCG, do inglês *Discounted Cumulative Gain*) foi desenvolvida (JÄRVELIN; KEKÄLÄINEN, 2000) para casos nos quais seja necessário manipular múltiplos níveis de relevância. Nestes casos, os julgamentos de relevância sobre os resultados consultados são fornecidos por meio de escalas de valores, por exemplo: 3 (muito relevante), 2 (relevante), 1 (pouco relevante) e 0 (não relevante).

A medida DCG é utilizada em especial, em casos nos quais é desejável que o sistema retorne primeiro documentos com maior relevância, dentre os documentos julgados relevantes. O objetivo da medida DCG é fazer com que esta diferença na ordenação se torne visível, na avaliação dos métodos de consulta.

Para calcular a medida DCG deve-se, primeiro, encontrar o ganho acumulado (CG) na posição *n*, que é calculado somando-se a relevância das tuplas encontradas a partir da posição 1 até a posição *n* na lista ordenada. Formalmente, o CG na posição *i* é definido recursivamente da seguinte maneira:

$$CG[n] = \begin{cases} G_{[n]}, & se\ n = 1; \\ G_{[n-1]} + G_{[n]}, & se\ n > 1. \end{cases}$$

Onde: $\mathrm{G[n]}$ é a relevância do documento encontrado na posição n da lista ordenada.

Por exemplo, considere a seguinte lista de relevância dos documentos retornados: $\mathrm{G'} = \{3,2,3,0,0,1,2,2,3,0\}$. Para a lista G' é retornada a seguinte lista de ganho acumulado: $\mathrm{CG'} = \{3,5,8,8,8,9,11,13,16,16\}$. O ganho acumulado de qualquer posição na lista de ordenação pode ser lido diretamente, por exemplo, $\mathrm{CG'}[7] = 11$.

A medida DCG fundamenta-se em dois conceitos:

- Documentos extremamente relevantes são mais importantes (valiosos) que documentos com relevância marginal; e
- Quanto mais baixa a posição do documento na lista ordenada (*ranking*), menor o valor deste documento para o usuário.

A medida DCG é implementada utilizando a equação:

$$DCG[i] = \begin{cases} G_{[i]}, & se\ i = 1; \\ DCG_{[i-1]} + \dfrac{G_{[i]}}{Log\ [i]}, & se\ i > 1. \end{cases}$$

Ou seja, quanto maior a posição da tupla na lista, maior será o desconto aplicado. Para G' do exemplo, é retornada a seguinte lista:

$$\text{DCG'} = \{3,\ 3+(2/1),\ 5+(3/1{,}58),\ \ldots$$

$$\text{DCG'} = \{3{,}5,\ 6.89,\ 6.89,\ 6.89,\ 7.28,\ 7.99,\ 8.66,\ 9.61,\ 9.61\}.$$

Em virtude da importância da apresentação dos resultados em consultas a bancos de dados, a medida DCG foi utilizada neste experimento como forma de melhor avaliar respostas potencialmente reescritas por diretivas de reescrita que provocam operações de ordenação e classificação e, portanto, influenciam nas posições relativas nas quais tuplas são apresentadas.

5.3.2. **Consultas, Perfis de Usuários e DRC**

As consultas originais emitidas pelos usuários são de formatação livre, no que diz respeito aos atributos projetados e tabelas relacionadas, embora sigam o padrão a seguir (vide Figura 5.3):

```
SELECT <atributos selecionados na tela de consulta>
  FROM movie [, genre, actor, director, language]
[WHERE <junções>
   AND <restrições para genre, actor, director
        selecionadas na tela de consultas>];
```

As consultas reescritas têm formatação dependente das regras de produção acionadas e, consequentemente, das DRC utilizadas no processo de reescrita. As seguintes regras de produção foram incluídas no motor de inferência para a aplicação TMDb:

```
R1. IF Q.movie THEN

R2.      IF &acompanhante.tipo = 'criança' OR
            user.age < 9 THEN
            FILTER ((genre.name NOT IN ('crime', 'guerra', 'filme-noir',
                                        'horror') OR
                     (movie.age >= 9) OR
                     (movie.pt_dubbing is false
                      AND language.name <> 'portugues')));
             ORDER (movie.year desc);
         END IF;

R3.      IF &acompanhante.tipo = 'adolescente' OR
            user.age <= 14 THEN
            FILTER (genre.name NOT IN ('crime', 'guerra'),
                     movie.age > 14));
         END IF;

        PROJECT (movie.mpaareason);
        PREFER (&user.id);

    END IF;
```

A regra R1 Informa que se deve usar as preferências do usuário, caso haja consulta para a entidade `movie` e acrescentar a projeção do atributo motivo da censura `(movie.mpaareason)`.

Caso o usuário tenha idade menor que nove anos ou esteja acompanhado de criança (regra R2), deve-se filtrar consultas à entidade `movie` restringindo filmes do gênero crime, guerra, film-noir e horror e também filmes cuja censura (`movie.age`) seja maior ou igual a nove anos. Também devem ser restritos filmes que não sejam falados ou dublados em português. Adicionalmente devem ser ordenados pelo critério de filmes mais recentes primeiro.

Já a regra R3 informa que se usuário tiver idade menor ou igual a catorze anos ou esteja acompanhado de adolescente, deve-se filtrar consultas à entidade `movie` restringindo filmes do gênero crime e guerra, e também filmes cuja censura (`movie.age`) seja maior que catorze anos.

Para o experimento, os critérios de atribuição de preferências da interface de cadastramento geram índices de preferências (IP) atribuídos arbitrariamente, conforme o Quadro 5.1:

Quadro 5.1 – Índices de preferência admitidos no experimento

EC	Entidade.atributo	Critério	IP
gênero do filme ator diretor	Genre.name, Actor.name, Director.name	gosto muito	0,9
		gosto	0,7
		indiferente	0,1
		não gosto	- 0,5
		detesto	-0,9
tipo do acompanhante	Genre.name	prefiro	1,0
		excluindo	-1,0
turno do horário	Genre.name	prefiro	1,0

Fonte: O autor.

Após coletadas as informações cadastrais, são montados os perfis de preferências dos usuários. O limite kp (vide DRC PREFER, no Capítulo 4), que informa o número mínimo de preferências que uma tupla de resposta deve atender para aplicação da DRC Prefer foi atribuído para maior ou igual a dois (2).

Neste experimento foram utilizadas as DRC PROJECT, FILTER, ORDER e PREFER. As DRC GROUP e SKYLINE não foram utilizadas no experimento em virtude de não terem uso prático em uma aplicação de sugestão de filmes. Considerou-se que este fato não prejudica o experimento e não influencia na confirmação ou refutação das hipóteses do trabalho.

5.3.3. **Exemplo**

Para exemplificar o processo de reescrita foram coletados os *logs* do experimento realizado pelo usuário 13, que tem as seguintes características: sexo feminino, idade 15 anos, gosta muito de comédias e filmes de aventura e gosta de animações. Acompanhada dos amigos prefere assistir comédias que não sejam faroeste. Baseado nessas informações e nos índices de preferências arbitrados para a aplicação TMDb (Quadro 5.1), o esquema do perfil de preferências do usuário 13 foi configurado conforme a Figura 5.5-a. Este usuário possui associado diretamente um perfil de usuário (user_id =13) e indiretamente um perfil de grupo (adolescente).

Figura 5.5 – Perfis de preferências do usuário 13

a)

```
<Profile>
type = user;
user_id = 13;

<Groups>
name = adolescente;

<Preferences>
P1. Genre.name = 'Comédia' (0,9);
P2. Genre.name = 'Aventura' (0,9);
P3. Genre.name = 'Animação' (0,7);

<Contextual Preferences>
Acompanhante.tipo = 'amigo(s)'
Cp1. Genre.name = 'Comédia' (1,0);
Cp2. Genre.name = 'Faroeste' (-1,0);
```

b)

```
<Profile>
type: group;
name: adolescente;

<Preferences>
Gp1. Movie.year >= '1999' (0,9);
```

Fonte: O autor.

O perfil de grupo *adolescente* (Figura 5.5-b) foi configurado pelo ED e determina que filmes posteriores a 1999 tenham índice de preferência 0,9, o que atribui para adolescentes uma preferência hipotética por filmes mais recentes.

Por meio da tela de consultas, este usuário selecionou os seguintes atributos para projeção: título (movie.title), subtítulo (movie.tagline), resumo (movie.plot), censura (movie.age) e língua (movie.language). Também escolheu restringir os gêneros (genre.name) para 'Comédia' ou 'Aventura'. No planejamento da sessão de exibição de filme, foi escolhido 'amigo(s)' como companhia.

De acordo com as opções feitas, Q foi configurada pela aplicação TMDb da seguinte forma:

```
Q:  SELECT distinct movie.movieid, movie.title, movie.tagline,
           movie.plot, movie.age, language.name as language
      FROM movie, language, genre, movie_genre
     WHERE language.languageid = movie.languageid
       AND genre.genreid = movie_genre.genreid
       AND movie.movieid = movie_genre.movieid
       AND (genre.name = 'Comédia' OR genre.name = 'Aventura');
```

A consulta Q teve tempo de leitura de 360 ms e 11.530 tuplas retornadas.

Para a reescrita de Q', a regra R1 foi acionada (vide Seção 5.3.2), o que provocou a aplicação das DRC Project que acrescentou a projeção do atributo motivo de censura (movie.mpaareason) e DRC Prefer que reorganizou a seleção em função de seleções parciais associadas às preferências do usuário (vide P1, P2 e P3 da Figura 5.5-a e Gp1 da Figura 5.5-b) unidas por cláusulas UNION ALL, agrupadas e ordenadas pelos GIC de cada tupla retornada. O formato resultante de Q' foi:

```
Q': SELECT Q.movieid, Q.title, Q.tagline, Q.plot, Q.age,
           Q.mpaareason, Q.language, AVG(GIC),count(*)
      FROM (
         SELECT distinct 0.9 * 0.9 as GIC, movie.movieid, movie.title, movie.tagline,
                movie.plot, movie.age, movie.mpaareason, language.name as language
           FROM movie, language, genre, movie_genre
          WHERE language.languageid = movie.languageid
            AND genre.genreid = movie_genre.genreid
            AND movie.movieid = movie_genre.movieid
            AND (genre.name = 'Comédia' OR genre.name = 'Aventura')
            AND movie.age < 16 AND EXISTS (SELECT smovie.movieid
                                     FROM movie smovie, genre sgenre,
                                               movie_genre smovie_genre
                                    WHERE smovie.movieid = movie.movieid
                                      AND sgenre.genreid = smovie_genre.genreid
                                      AND smovie.movieid = smovie_genre.movieid
                                      AND sgenre.name = 'Comédia')
    UNION ALL
         SELECT distinct 0.9 * 0.9 as GIC, movie.movieid, movie.title, movie.tagline,
                movie.plot, movie.age, movie.mpaareason, language.name as language
           FROM movie, language, genre, movie_genre
          WHERE language.languageid = movie.languageid
            AND genre.genreid = movie_genre.genreid
            AND movie.movieid = movie_genre.movieid
            AND (genre.name = 'Comédia' OR genre.name = 'Aventura')
            AND movie.age < 16 AND EXISTS (SELECT smovie.movieid
                                        FROM movie smovie, genre sgenre,
                                             movie_genre smovie_genre
                                       WHERE smovie.movieid = movie.movieid
                                         AND sgenre.genreid = smovie_genre.genreid
                                         AND smovie.movieid = smovie_genre.movieid
                                         AND sgenre.name = 'Aventura')
    UNION ALL
         SELECT distinct 0.7 * 0.9 as GIC, movie.movieid, movie.title, movie.tagline,
                movie.plot, movie.age, movie.mpaareason, language.name as language
           FROM movie, language, genre, movie_genre
          WHERE language.languageid = movie.languageid
            AND genre.genreid = movie_genre.genreid
            AND movie.movieid = movie_genre.movieid
            AND (genre.name = 'Comédia' OR genre.name = 'Aventura')
            AND movie.age < 16 AND EXISTS (SELECT smovie.movieid
                                        FROM movie smovie, genre sgenre,
                                             movie_genre smovie_genre
                                       WHERE smovie.movieid = movie.movieid
                                         AND sgenre.genreid = smovie_genre.genreid
                                         AND smovie.movieid = smovie_genre.movieid
                                         AND sgenre.name = 'Animação')
    UNION ALL
         SELECT distinct 1.0 * 1.0 as GIC, movie.movieid, movie.title, movie.tagline,
                movie.plot, movie.age, movie.mpaareason, language.name as language
           FROM movie, language, genre, movie_genre
          WHERE language.languageid = movie.languageid
            AND genre.genreid = movie_genre.genreid
            AND movie.movieid = movie_genre.movieid
            AND (genre.name = 'Comédia' OR genre.name = 'Aventura')
            AND movie.age < 16
            AND movie.year >= '1999'
    ) as Q
    GROUP BY Q.movieid, Q.title, Q.tagline, Q.plot, Q.age, Q.mpaareason, Q.language
    HAVING Count(*) >= 2
    ORDER BY AVG(GIC) DESC, COUNT(*) DESC
```

A consulta Q’ retornou 6.221 tuplas com os seguintes tempos de processamento:

- Motor de Inferência: 162 ms;
- Reescrita: 22 ms; e
- Leitura: 667 ms.

A reescrita de Q’’, além das preferências utilizadas na reescrita de Q’, considerou o elemento contextual externo *acompanhante* igual a amigo(s), informado na tela de consulta, o que provocou a aplicação da DRC Preferer com suas preferências (vide Cp1 e Cp2, da Figura 5.5-a). Observa-se que não há conflito entre preferências, visto que o modelo de dados admite que um filme tenha mais de um gênero (ao contrário do exemplo da Seção 4.4.8). O formato resultante da reescrita para Q’’ foi:

```
Q": SELECT Q.movieid, Q.title, Q.tagline, Q.plot, Q.age,
          Q.mpaareason, Q.language, AVG(GIC),count(*)
    FROM (
        SELECT distinct 0.9 * 0.9 as GIC, movie.movieid, movie.title, movie.tagline,
               movie.plot, movie.age, movie.mpaareason, language.name as language
          FROM movie, language, genre, movie_genre
         WHERE language.languageid = movie.languageid
           AND genre.genreid = movie_genre.genreid
           AND movie.movieid = movie_genre.movieid
           AND (genre.name = 'Comédia' OR genre.name = 'Aventura')
           AND movie.age < 16 AND EXISTS (SELECT smovie.movieid
                                            FROM movie smovie, genre sgenre,
                                                 movie_genre smovie_genre
                                           WHERE smovie.movieid = movie.movieid
                                             AND sgenre.genreid = smovie_genre.genreid
                                             AND smovie.movieid = smovie_genre.movieid
                                             AND sgenre.name = 'Comédia')
    UNION ALL
        SELECT distinct 0.9 * 0.9 as GIC, movie.movieid, movie.title, movie.tagline,
               movie.plot, movie.age, movie.mpaareason, language.name as language
          FROM movie, language, genre, movie_genre
         WHERE language.languageid = movie.languageid
           AND genre.genreid = movie_genre.genreid
           AND movie.movieid = movie_genre.movieid
           AND (genre.name = 'Comédia' OR genre.name = 'Aventura')
           AND movie.age < 16 AND EXISTS (SELECT smovie.movieid
                                            FROM movie smovie, genre sgenre,
                                                 movie_genre smovie_genre
                                           WHERE smovie.movieid = movie.movieid
                                             AND sgenre.genreid = smovie_genre.genreid
                                             AND smovie.movieid = smovie_genre.movieid
                                             AND sgenre.name = 'Aventura')
    UNION ALL
        SELECT distinct 0.7 * 0.9 as GIC, movie.movieid, movie.title, movie.tagline,
               movie.plot, movie.age, movie.mpaareason, language.name as language
          FROM  movie, language, genre, movie_genre
         WHERE language.languageid = movie.languageid
           AND genre.genreid = movie_genre.genreid
           AND movie.movieid = movie_genre.movieid
           AND (genre.name = 'Comédia' OR genre.name = 'Aventura')
           AND movie.age < 16 AND EXISTS (SELECT smovie.movieid
                                            FROM movie smovie, genre sgenre,
                                                 movie_genre smovie_genre
                                           WHERE smovie.movieid = movie.movieid
                                             AND sgenre.genreid = smovie_genre.genreid
                                             AND smovie.movieid = smovie_genre.movieid
```

```
                                        AND sgenre.name = 'Animação')
UNION ALL
     SELECT 1.0 * 1.0 as GIC, movie.movieid, movie.title, movie.tagline,
            movie.plot, movie.age, movie.mpaareason, language.name as language
       FROM movie, language, genre, movie_genre
      WHERE language.languageid = movie.languageid
        AND genre.genreid = movie_genre.genreid
        AND movie.movieid = movie_genre.movieid
        AND (genre.name = 'Comédia' OR genre.name = 'Aventura')
        AND movie.age < 16
        AND movie.year >= '1999'
UNION ALL
     SELECT 1.0 * 0.9 as GIC, movie.movieid, movie.title, movie.tagline,
            movie.plot, movie.age, movie.mpaareason, language.name as language
       FROM movie, language, genre, movie_genre
      WHERE language.languageid = movie.languageid
        AND genre.genreid = movie_genre.genreid
        AND movie.movieid = movie_genre.movieid
        AND (genre.name = 'Comédia' OR genre.name = 'Aventura')
        AND movie.age < 16 AND EXISTS (SELECT smovie.movieid
                                         FROM movie smovie, genre sgenre,
                                              movie_genre smovie_genre
                                        WHERE smovie.movieid = movie.movieid
                                          AND sgenre.genreid = smovie_genre.genreid
                                          AND smovie.movieid = smovie_genre.movieid
                                          AND sgenre.name = 'Comédia')
UNION ALL
     SELECT 1.0 * 0.9 as GIC,   movie.movieid, movie.title, movie.tagline,
            movie.plot, movie.age, movie.mpaareason, language.name as language
       FROM movie, language, genre, movie_genre
      WHERE language.languageid = movie.languageid
        AND genre.genreid = movie_genre.genreid
        AND movie.movieid = movie_genre.movieid
        AND (genre.name = 'Comédia' OR genre.name = 'Aventura')
        AND movie.age < 16 AND NOT EXISTS (SELECT smovie.movieid
                                             FROM movie smovie, genre sgenre,
                                                  movie_genre smovie_genre
                                            WHERE smovie.movieid = movie.movieid
                                              AND sgenre.genreid = smovie_genre.genreid
                                              AND smovie.movieid = smovie_genre.movieid
                                              AND sgenre.name = 'Faroeste')
)
as Q
GROUP BY Q.movieid, Q.title, Q.tagline, Q.plot, Q.age, Q.mpaareason, Q.language
HAVING Count(*) >= 2
ORDER BY AVG(GIC) DESC, COUNT(*) DESC
```

A consulta Q'' teve 9.954 tuplas retornadas e os seguintes tempos de processamento:

- Motor de Inferência: 206 ms;
- Reescrita: 7 ms; e
- Leitura: 1.408 ms

5.4. Resultados

Os resultados dos experimentos foram extraídos diante da perspectiva de comparação entre as avaliações de consultas originais versus avaliações de consultas reescritas, o que permitiu a análise das medidas P@20, MAP e DCG (ver Seção 5.2), conforme descrito nas subseções seguintes.

5.4.1. **Cálculo da Precisão**

A primeira avaliação feita procurou mostrar uma visão geral das respostas dos usuários, e para tanto foi utilizado o Histograma de Precisão. O objetivo é comparar o julgamento dos usuários para as respostas apresentadas. Esse histograma é utilizado para observar o comportamento de um algoritmo para consultas individuais.

Nesta avaliação foi utilizada a medida P@n com n=20 (P@20), computada para as várias consultas e comparadas as avaliações das repostas das consultas Q (original) com as respostas das consultas Q' (Q reescrita com elementos contextuais internos) e com as respostas das consultas Q'' (Q reescrita com elementos contextuais internos e externos). Considere:

P_Q@20: Precisão da consulta Q no vigésimo item;
$P_{Q'}$@20: Precisão da consulta Q' no vigésimo item;
$P_{Q''}$@20: Precisão da consulta Q'' no vigésimo item;

$P_{Q'/Q}$@20= $P_{Q'}$@20 – P_Q@20
$P_{Q''/Q}$@20 = $P_{Q''}$@20 – P_Q@20

Dado que a finalidade desta avaliação foi traçar um perfil global das consultas, para efeito de simplificação, consideramos avaliações 3 (muito relevante) e 2 (relevante) como relevantes e avaliações 1 (pouco relevante) e 0 (irrelevante) como irrelevantes.

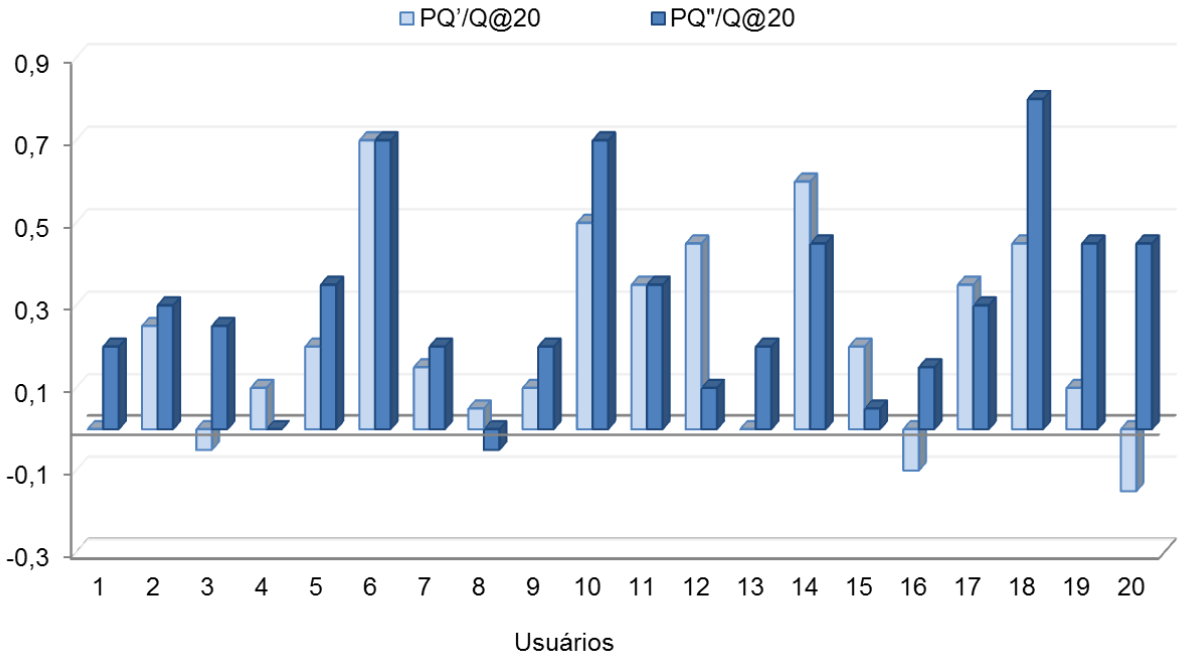
Análise dos Resultados Obtidos

Os resultados obtidos podem ser verificados na Figura 5.6. Observa-se que:

- Nas consultas Q' dos usuários 3, 16 e 20 houve uma quantidade menor de itens considerados relevantes em detrimento das consultas Q correspondentes (colunas com marcação negativa);
- Apenas o usuário 8 considerou uma quantidade menor de resultados relevantes para Q'' em comparação a Q;
- Dos 20 usuários, 12 consideraram haver mais respostas relevantes em Q'' do que em Q';
- 6 usuários consideraram haver mais respostas relevantes em Q' do que em Q'';

- Para 2 usuários, a quantidades de resultados relevantes de Q’ e Q’’ foi igual; e
- A totalidade dos usuários considerou haver mais resultados relevantes em pelo menos uma das consultas reescritas.

Figura 5.6 – Histograma de P@20 para consultas de 20 usuários

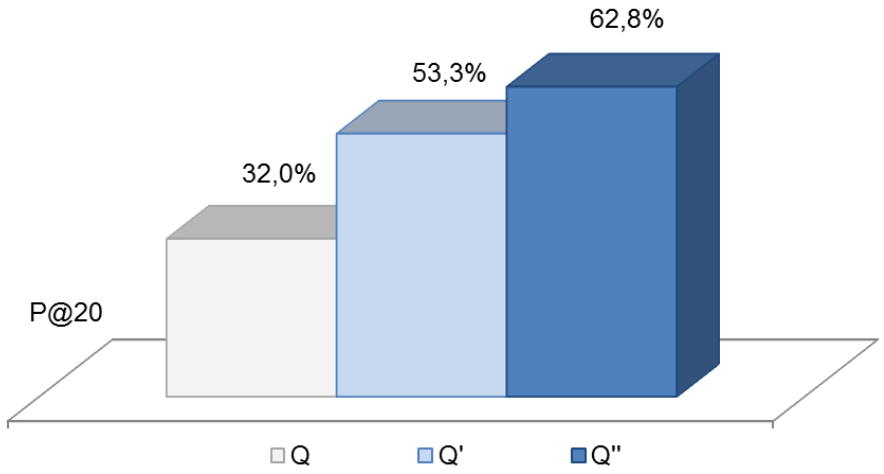


Fonte: O autor.

O valores médios das P@20, ao longo dos conjuntos de consultas dos 20 usuários, está representada na Figura 5.7, com resultados:

- Média de P_Q@20 = $\frac{\Sigma_{i=1}^{20} P@20(Qi)}{20}$ = 6,40/20 = 0,320 = 32,0%;
- Média de $P_{Q'}$@20 = $\frac{\Sigma_{i=1}^{20} P@20(Q'i)}{20}$ = 10,65/20 = 0,533 = 53,3%;
- Média de $P_{Q''}$@20 = $\frac{\Sigma_{i=1}^{20} P@20(Q''i)}{20}$ = 12,55/20 = 0,628 = 62,8%;

Figura 5.7 – Valores médios de P@20



Fonte: O autor.

Verificação das Hipóteses

De acordo com os resultados apresentados nos gráficos das Figuras 5.6 e 5.7, foi observado que para 100% dos usuários pelo menos um dos dois casos de consultas reescritas (Q’ ou Q’’) apresentou melhorias de relevância das respostas. Desse grupo, 65% consideraram que ambas as consultas reescritas apresentaram melhorias, em comparação à consulta original.

Ainda, segundo análise dos valores médios da relevância dos resultados, apontados pelos usuários, as consultas reescritas apresentaram substancial melhoria com 21,3% a mais de relevância para as consultas Q` (53,3%) em relação às consultas originais (32,0%) e consultas Q’’ (62,8%) apresentando quase o dobro da relevância apontada para as consultas Q.

Esses resultados levam a confirmar as hipóteses H1 e H2 (vide Seção 5.1.2) desta tese.

5.4.2. Cálculo da Precisão Média

A segunda avaliação calculou a precisão média utilizando a medida *Mean Average Precision* (MAP) para os valores P@20 coletados para as consultas Q, Q’ e Q’’.

Análise dos Resultados Obtidos

Os valores aferidos estão apresentados na Tabela 5.1.

Tabela 5.1 – Resultados de MAP para 20 usuários

Usuário	MAP_Q	$MAP_{Q'}$	$MAP_{Q''}$
1	0,00	0,00	**0,34**
2	0,46	0,64	**0,65**
3	0,26	0,51	**0,72**
4	**0,70**	0,69	0,43
5	0,51	0,72	**0,96**
6	0,08	0,64	**0,84**
7	0,46	0,41	**0,73**
8	0,57	**0,71**	0,65
9	0,58	0,66	**0,72**
10	0,35	0,78	**1,00**
11	0,69	**1,00**	**1,00**
12	0,40	**0,92**	0,52
13	0,31	0,44	**0,47**
14	0,08	**0,88**	0,53
15	0,68	**0,88**	0,87
16	0,39	0,36	**0,50**
17	0,25	0,55	**0,63**
18	0,42	0,71	**0,92**
19	0,33	0,15	**0,58**
20	0,63	0,22	**0,88**
Média	0,41	0,59	**0,70**

Fonte: O autor.

Nos resultados obtidos, destacam-se os seguintes pontos:

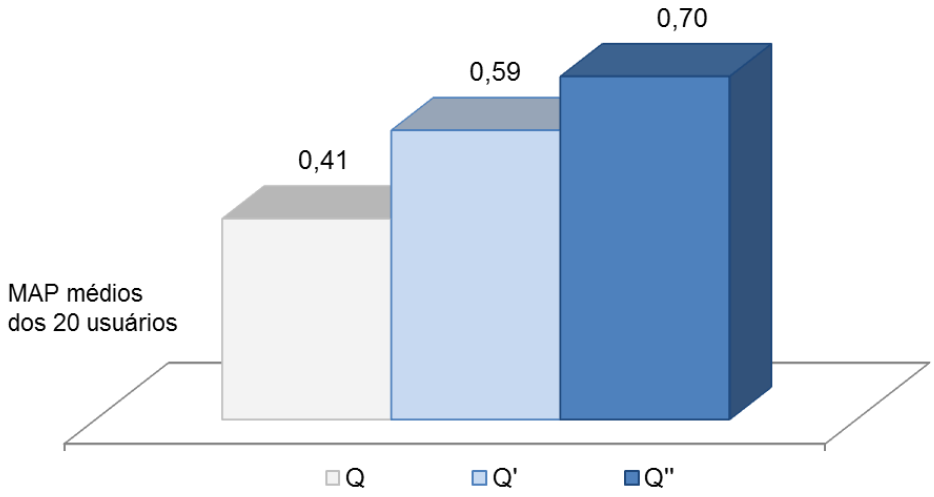
- O usuário 3, que em termos absolutos tinha a consulta Q com mais tuplas relevantes que Q' (medida P@20), na análise MAP foi constatada que as tuplas de Q' tinham relevância maior que as de Q. Fato semelhante ocorreu com o usuário 8 em relação às consultas Q e Q'';
- Para o usuário 17, a análise do MAP indicou maior satisfação com a consulta Q'' em relação à consulta Q, ao contrário do que apontava a medida P@20.

- Observamos que os maiores valores MAP, macados em negrito, concentram-se na coluna correspondente a Q'' com quinze (15) ocorrências, em seguida a coluna correspondente a Q', com cinco (5) ocorrências, enquanto que a coluna corresmpondente à consulta Q apresenta apenas uma (1) ocorrência.

Verificação das Hipóteses

Na Figura 5.8, é possível observar os valores médios da medida MAP dos vinte (20) usuários do experimento. A análise da relevância apontada pelos usuários, levando-se em consideração a ordenação do resultado, destaca a tendência de que consultas reescritas utilizando EC internos e externos (Q'') apresentem resultados mais relevantes para os usuários em comparação a consultas reescritas que utilizem apenas EC internos. Esses resultados confirmam a hipótese H3 desta tese.

Figura 5.8 – Valores médios MAP para consultas Q, Q' e Q'' de 20 usuários



Fonte: O autor.

5.4.3. **Cálculo do Ganho Cumulativo Descontado**

A terceira e última métrica utilizada nesta tese foi o Ganho Cumulativo Descontado (DCG). Esta medida, ao contrário das anteriores, considera os índices de relevância apontados de forma não binária, o que permite aplicar a análise da relevância considerando cenários nos quais a opinião dos usuários varie entre valores em diferentes gradações (e.g. ótimos, bons, indiferentes, ruins, péssimos).

Esta análise é importante, visto que a adequação ao contexto é passível de ser aplicada a domínios distintos, com características que necessitem adequação de avaliação em um nível de granularidade maior.

Análise dos Resultados Obtidos

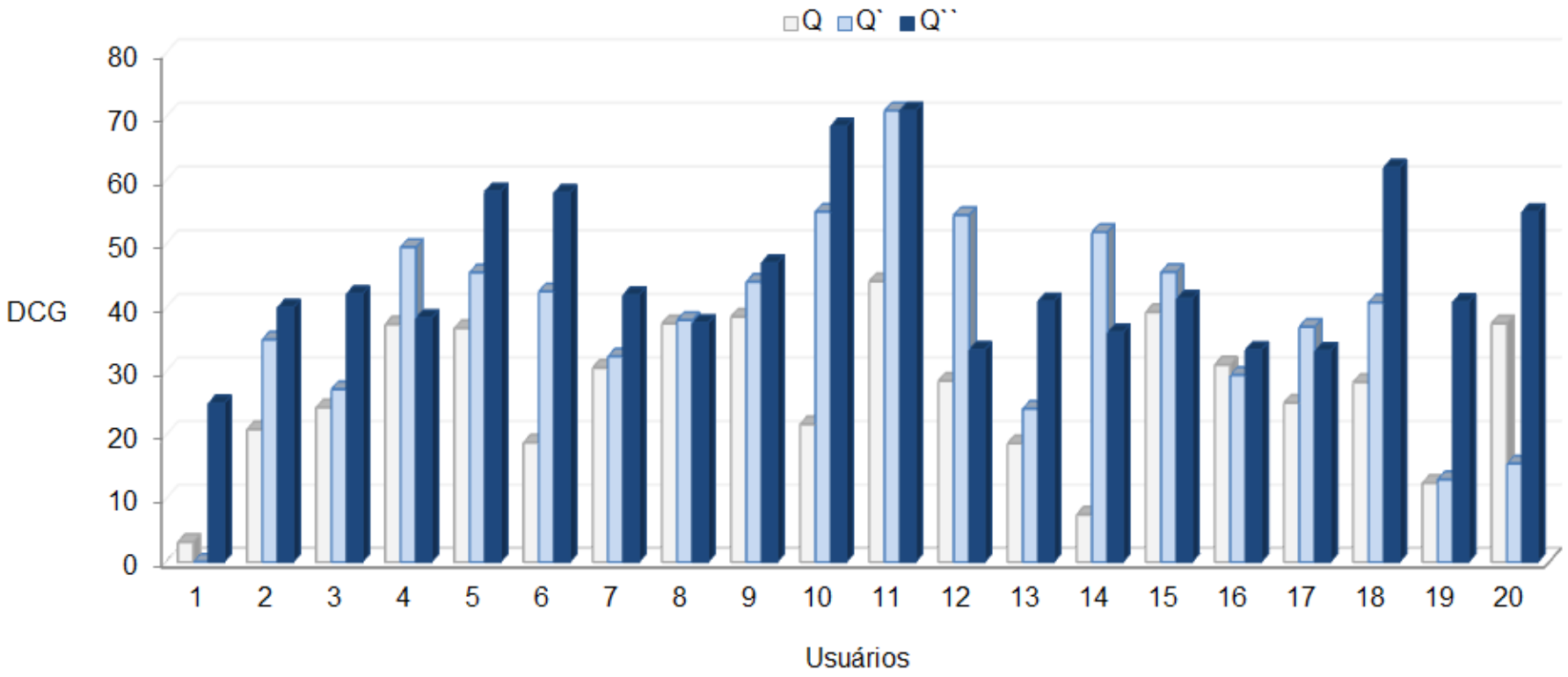
A análise dos resultados, tendo como base a medida DCG, apresentou a distribuição exibida na Figura 5.9. Neste gráfico, pode-se observar:

- Para o usuário 1, as medidas P@20 para Q e Q'' eram semelhantes, mas a medida DCG revelou a maior relevância em favor da consulta Q;
- Para 6 usuários (30%), a consulta Q foi considerada mais relevante;
- Dos 20 usuários participantes do experimento, 13 (65%) avaliaram a consulta Q'' como mais relevante;
- 1 usuário considerou Q' e Q'' igualmente relevantes;
- Para 18 usuários (90%), as duas consultas reescritas foram consideradas mais relevantes que a consulta original; e
- 100% dos usuários consideraram que pelo menos uma das consultas reescritas foi mais relevante que a consulta original.

Verificação das Hipóteses

A análise das medidas DGC confirma as hipóteses H1 e H2 desta tese e indicam que, para domínios cujos parâmetros de satisfação dos usuários em relação à relevância de respostas que utilizem escalas com granularidade maior, as consultas Q'' acentuaram a tendência de melhoria em relação às consultas Q'. Assim, conclui-se que esse experimento ratifica a hipótese H3, apresentada pela métrica MAP.

Figura 5.9 – Avaliação do DCG



Fonte: O autor.

5.4.4. **Considerações Sobre os Resultados**

De forma a complementar à análise dos dados obtidos com os experimentos, faz-se necessária uma discussão sobre os resultados que, apesar de não implicarem na refutação das hipóteses formuladas, claramente estão fora da tendência observada nos resultados do grupo de usuários. Para estes casos, foram considerados como fora da tendência os resultados nos quais a avaliação DCG para a consulta Q não foi igualada ou superada pela avaliação de ambas as consultas reescritas Q' e Q'' correspondentes. Esses resultados podem ser observados nas avaliações dos usuários de números 1, 16 e 20 da Figura 5.9.

Para esses casos as seguites análises foram realizadas:

Usuário 1:

Gêneros preferidos: comédia, romance;

Atores preferidos: não informou;

Acompanhantes: Acompanhado da criança prefere filmes de animação que não sejam ficção científica;

Consulta realizada: gêneros animação ou comédia ou fantasia;

Seção de exibição: companhia de criança, às 16h.

Observa-se que para o usuário 1, a consulta Q foi realizada diretamente para os gêneros animação, comédia e fantasia, que incluiu o gênero animação, coincidente com o gênero preferido para exibição na companhia de criança. A consulta reescrita Q' não levou em consideração o elemento contextual externo "companhia", mas apenas elementos contextuais internos de preferência pelos gêneros comédia e romance. Esses dois gêneros foram adicionados à consulta original, o que levou Q' a ter resposta com filmes com variedade de gêneros maior que a consulta Q, portanto menos focada nas opções desejadas.

A consulta Q" foi reescrita enfatizando a preferência pelo gênero associado ao EC externo companhia (gênero preferido animação), o que resultou em resposta mais focada e bem avaliada.

Usuário 16:

Gêneros preferidos: animação, crime, comédia;

Atores preferidos: Vin Diesel, Denzel Washington, Russell Crowe;

Turnos: à tarde prefere animações;

Acompanhantes: Acompanhado de crianças prefere animações que não sejam ficção científica e acompanhado da companheira prefere comédias que não sejam musicais.

Consulta realizada: gêneros animação ou fantasia ou comédia

Seção de exibição: companhia de crianças, às 16h.

O caso do usuário 16, é semelhante ao do usuário 1, com o agaravante que a reescrita de Q' levou em consideração a preferência do gênero crime, inadequada para exibição em companhia de crianças. Da mesma forma que no caso do usuário 1, a consulta reescrita Q", por considerar o elemento externo companhia, obteve melhor avaliação que a consulta Q, mesmo sendo esta última selecionada apenas com gêneros preferidos pelo usuário para assistir com cianças (animação, fantasia ou comédia).

Usuário 20:

Gêneros preferidos: crime, suspense, romance;

Atores preferidos: Ethan Hawke, Denzel Washington, Julia Roberts.

Acompanhantes: Acompanhado da companheira prefere filmes de ação que não sejam horror.

Consulta realizada: gêneros ação ou crime ou romance e atores Nicolas Cage ou Brad Pitt ou Denzel Washington;

Seção de exibição: companhia da companheira, às 21h.

Para o caso do usuário 20, entende-se que a influência advinda do EC interno para atores/atrizes preferidos, em especial Ethan Hawke e Julia Roberts fez com que as respostas de Q' contemplassem respostas com uma quantidade maior de filmes de gênero diferente de "ação", informado como preferido na companhia da companheira. Isso provocou a avaliação pior de Q' em relação à Q.

Conclui-se que, para domínios que exijam caracterização do contexto externo, a utilização apenas de elementos contextuais internos pode levar a uma inferência

incompleta do contexto e utilização de regras que levem a efeitos indesejados de respostas menos focadas ou inadequadas. É de responsabilidade do especialista no negócio mapear essas situações e configurar regras que não permitam que esse tipo de situação aconteça.

5.4.5. **Desempenho**

A análise detalhada do desempenho do protótipo não fez parte do escopo deste trabalho. Para o protótipo implementado, ações para melhoria de desempenho, tais como análises de planos de leitura, análises de índices e particionamentos de dados não foram realizadas.

Os experimentos foram realizados em um computador portátil com processador Intel[38] Core i5-2520M de 2,50 GHz, 4 GB de memória RAM e sistema operacional Microsoft Windows[39] 7 de 64 Bits.

A título de ilustração dos resultados aferidos na utilização do protótipo, e levando em consideração que os usuários realizaram consultas diferentes, temos os seguintes tempos médios de processamento, para 20 consultas:

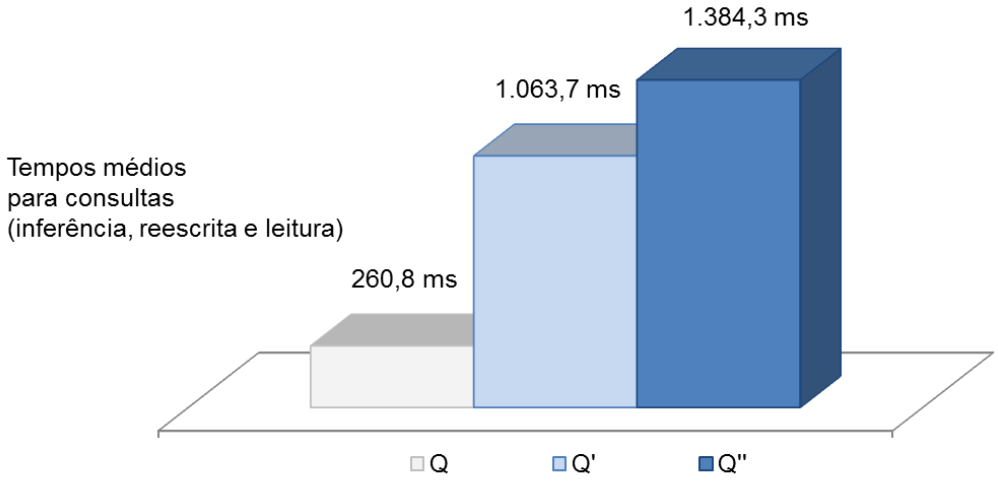
Consultas Q:	
Tempo médio de leitura de dados:	260,8 ms.
Consultas Q':	
Tempo médio de processamento do motor de inferência:	247,6 ms.
Tempo médio de processamento de reescrita':	14,1 ms.
Tempo médio de leitura de dados:	802,9 ms.
Consultas Q'':	
Tempo médio de processamento do motor de inferência:	248,8 ms.
Tempo médio de processamento de reescrita:	23,9 ms.
Tempo médio de leitura de dados:	1.111,6 ms.

O gráfico exibido na Figura 5.10 apresenta o comparativo entre os tempos médios decorridos para todo processo de consulta (inferência do contexto, reescrita

[38] Intel, disponível em: http://www.intel.com.br. Acesso em 02/02/2015.
[39] Microsoft Windows, disponível em: http://windows.microsoft.com. Acesso em 02/02/2015.

e leitura de dados), para as consultas realizadas dos tipos Q, Q’ e Q”, dos 20 usuários.

Figura 5.10 – Tempos médios de consulta



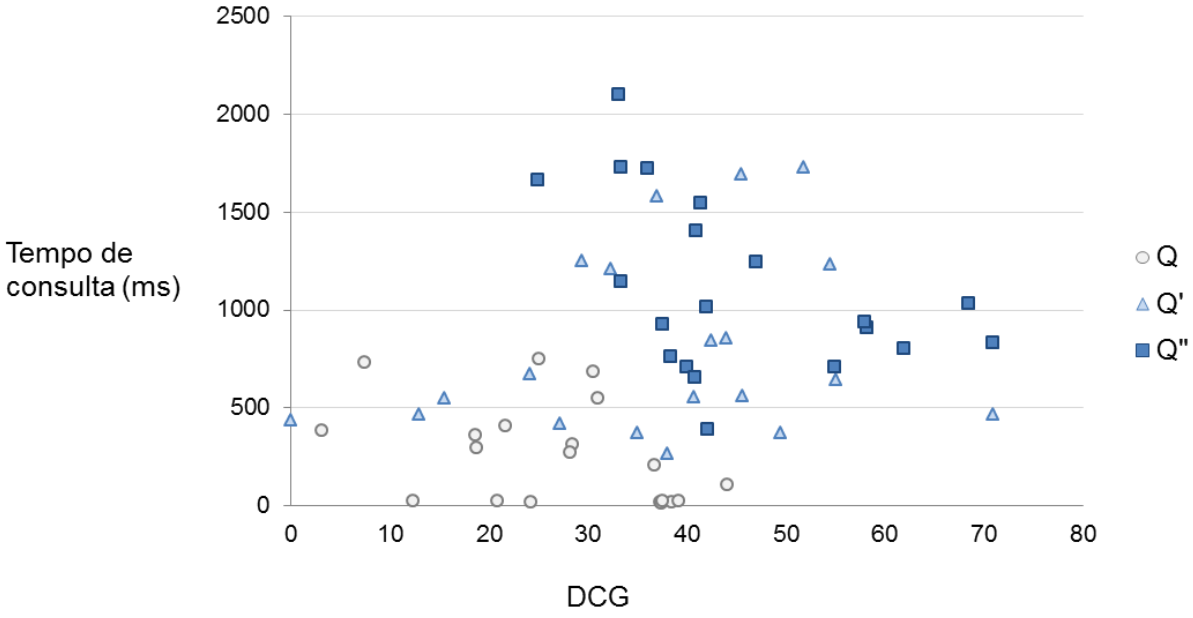
Fonte: O autor.

Questões relacionadas ao desempenho da abordagem são dependentes de cada SGBD usado e do conjunto de regras de inferência do contexto definidas, que disparam regras de reescritas específicas para cada sistema. O bom funcionamento da abordagem depende também da configuração pertinente das regras pelo especialista no domínio. O volume de dados armazenados nos bancos de dados das aplicações, e características de *hardware* dos computadores utilizados são outros fatores críticos para o desempenho de consultas.

Esta diversidade de situações faz com que seja recomendável que a decisão pela adoção da Abordagem Texere seja fruto de avaliações em ensaio em escala reduzida para cada sistema e domínio e de avaliação do custo da implementação versus benefícios da sensibilização de consultas ao contexto. A Figura 5.11 apresenta um gráfico de distribuição relativo aos valores aferidos dos tempos de consulta dos vinte usuários do experimento e seus respectivos índices DCG. A análise de valores como os apresentados no Figura 5.11, em conjunto com o que se espera da aplicação e seus requisitos de desempenho darão subsídios para que os projetistas de sistemas sensíveis ao contexto possam avaliar a opção pela implementação da Abordagem Texere. No caso apresentado, é possível traçar um

perfil do custo para processamento das consultas reescritas em relação aos índices de relevância atingidos.

Figura 5.11 – Distribuição do tempo de consulta x DCG



Fonte: O autor.

Observa-se que não fez parte do escopo deste trabalho a realização de estudos sobre a escalabilidade da abordagem proposta. Tais estudos propiciariam análises de fatores tais como número de regras de produção cadastradas na base de conhecimento do motor de inferência, número de elementos contextuais avaliados em regras de produção, volume de dados e número de tabelas da base de dados alvo e como esses fatores influenciam o desempenho dos processos de reecrita e consulta.

5.5. Considerações

Este capítulo mostrou aspectos relacionados à implementação de um protótipo e experimentação da abordagem Texere. Foram descritos os experimentos realizados para avaliação das hipóteses propostas nesta tese. Os aspectos relacionados à adequação ao contexto e relevância das repostas retornadas por consultas originais e reescritas foram comparados com base na opinião de usuários reais.

A análise dos resultados obtidos indica que a Abordagem Texere proposta permite a obtenção dos ganhos em relevância de respostas a consultas a bancos de dados, levantados nas hipóteses deste trabalho. O próximo capítulo apresenta as conclusões desta tese, aponta suas limitações e indica trabalhos futuros.

6. CONCLUSÃO E TRABALHOS FUTUROS

O desenvolvimento de tecnologias para Internet mudou o perfil e necessidades dos usuários que realizam consultas a bancos de dados. As aplicações que disponibilizam consultas a dados contam com enorme volume de informação armazenada de forma distribuída. O expressivo aumento do acesso à informação provocou a ampliação da diversidade de usuários, com necessidades e preferências diversas. Esses usuários utilizam dispositivos de consulta móveis e/ou distribuídos pelo ambiente, fazendo com que o modelo tradicional de consultas pré-formatadas, utilizado nas aplicações dedicadas não mais se apresente apropriado. A diversidade de usuários, ambientes e situações pode ocasionar que respostas a consultas a bancos de dados retornem excesso, inadequação ou pouca relevância dos dados retornados.

Contexto é entendido como o conjunto de elementos que caracterizam uma entidade em um domínio e que são considerados relevantes em uma determinada situação. SGBD que lidem com dados contextualizados são vistos como ferramentas importantes para preencher a lacuna existente entre os SGBD tradicionais e as aplicações computacionais que demandam, cada vez mais, informações sensíveis a contexto. Apesar de largamente adotados e possuírem tecnologia madura e eficiente para gerenciar e responder consultas a dados estruturados, os SGBD relacionais não possuem recursos necessários para responder consultas considerando o contexto.

Neste trabalho, buscou-se alternativa que possibilite que os SGBD relacionais possam responder a consultas de forma sensível ao contexto. A abordagem Texere, apresentada nesta tese, especifica: um processo de mapeamento de elementos contextuais para um repositório de metadados; um método para definição de regras de inferência do contexto que informam como os dados devem ser selecionados em função do contexto inferido; a especificação de um conjunto de diretivas de reescrita de consultas a bancos de dados relacionais; e a arquitetura de *software* capaz de materializar a implementação da Abordagem Texere, baseada no modelo de contexto e em um módulo de contextualização de consultas. A reescrita de consultas

tem como objetivo a obtenção de respostas mais adequadas ao contexto sob as quais foram emitidas e, portanto, mais relevantes para o usuário.

Este capítulo está organizado da seguinte forma: a Seção 6.1 apresenta as contribuições da tese e a Seção 6.2 relaciona trabalhos futuros complementares ao apresentado.

6.1. Contribuições

A contribuição geral desta tese é a proposição de uma abordagem para a resolução do problema de como responder consultas a bancos de dados relacionais de forma sensível ao contexto. Dentre as contribuições específicas, as mais importantes, pelo caráter de originalidade e aplicabilidade, são a especificação e a implementação de um conjunto de diretivas de reescrita de consultas SQL.

As contribuições específicas deste trabalho são as seguintes:

- *Especificação de abordagem e arquitetura de software para que consultas a SGBD relacionais sejam reescritas para refletir o contexto sob o qual foram submetidas;*

  Foi especificada e modelada uma arquitetura de *software* configurada em camada entre o banco de dados relacional e suas aplicações clientes. Foram especificados os componentes desta arquitetura suas funcionalidades, métodos, comunicações e interação com atores envolvidos.

- *Especificação de um modelo de contexto para bancos de dados;*

  O metamodelo de contexto de VIEIRA (2008) foi instanciado e estendido para bancos de dados;

  Foi proposto o relacionamento entre entidades contextuais, inexistente no metamodelo referenciado; e

  Foram apresentadas as definições de elementos contextuais externos e internos ao banco de dados, contexto de uma consulta a bancos de dados e foco de uma consulta a banco de dados.

- *Especificação de um método para definição de regras de identificação do contexto, que informam como os dados devem ser selecionados em função do contexto inferido;*
  Propusemos a aplicação de método baseado em regras de produção para identificação do contexto associado a uma consulta a bancos de dados e consequente instrução de reescrita dessa consulta.

- *Especificação e implementação de um conjunto de diretivas de reescrita de consultas SQL;*
  Foi proposto e especificado um conjunto de diretivas para reescrita de consultas em SQL.

- *Especificação e implementação de modelo de preferências, extensão do modelo de Koutrika e Ioannidis (2005) para consideração do contexto; e*

- *Desenvolvimento e avaliação experimental da abordagem.*
  Implementação de protótipo da arquitetura Texere;
  Implementação de aplicação para experimento;
  Realização de experimento com usuários reais para análise de resultados e comprovação de hipóteses; e
  Investigação comparativa da influência de elementos contextuais internos e externos para adequação de consultas a bancos de dados.

6.2. Limitações e Trabalhos Futuros

Os aspectos a seguir configuram-se como limitações identificadas neste trabalho:

- A dependência do Especialista no Domínio para as tarefas de identificação de elementos contextuais pertinentes e configuração das regras de produção. Para que as operações de reescrita de consultas sejam bem sucedidas, é necessário que as regras de produção tenham sido bem elaboradas e isentas de conflitos, sob risco das consultas reescritas não retornarem resultados relevantes para os usuários ou mesmo resultados errados;

- A presente versão da abordagem Texere foi desenvolvida apenas consultas a SGBD relacional e linguagem SQL, padrão ANSI (SQL ISO 1992, 2003, 2008);

- O trabalho realizado não contemplou estratégias para melhoria de desempenho das consultas reescritas nem análises da correção ou eficiência das consultas originais; e

- O trabalho proposto necessita de avaliação de escalabilidade com análises de influência da quantidade de regras de produção, quantidade de tabelas do banco de dados alvo, quantidade de entidades envolvidas em consultas com a estratégia de preferência adotada e volume de dados do banco de dados alvo.

As seguintes pesquisas são consideradas trabalhos relacionados a esta tese e que complementariam as pesquisas realizadas:

- *DRC aplicáveis a outros modelos de BD e outras linguagens de consulta;*

  Especificação de conjunto de diretivas para reescrita de consultas aplicadas a outras linguagens de consulta, em especial para linguagem SPARQL[40].

- *Evolução das DRC para incorporarem estratégias de melhoria de desempenho;*

  Evolução das versões iniciais dos algoritmos das DRC para preverem e adaptarem-se a situações que necessitem desempenho melhor.

- *Desenvolvimento de novas DRC;*

- Proposição de estratégia para automatizar a geração de regras de identificação de contexto e acionamento de DRC;

[40] SPARQL, disponível em: http://www.w3.org/TR/rdf-sparql-query/. Acesso em: 15 dez. 2014.

- *Detalhamento e implementação do módulo de ajuste de resposta;*

  Detalhamento e desenvolvimento do módulo de ajuste de respostas cuja especificação neste trabalho foi apenas funcional.

- *Detalhamento e implementação do módulo de gerenciamento de regras; e*

- *Análise de fatores de desempenho para a abordagem.*

  Análise dos fatores que influenciam em consultas reescritas (e.g. volume de dados, número de junções, número de regras disparadas, número de atributos afetados pela reescrita).

REFERÊNCIAS

AKMAN V.; SURAV M. **Steps toward formalizing context**. AI Magazin, v. 17, n. 3. p 55-72,1996.

AKMAN V.; SURAV M. **The use of situation theory in context modeling**. Computational Intelligence, v. 13, n.3, p 427-438, 1997.

AKRIVAS, G. et al. **Context-sensitive semantic query expansion**. Artificial Intelligence Systems, ICAIS. p. 109-114. IEEE International Conference on. IEEE, 2002.

AMO, S.; RIBEIRO, M. R. **CPrefSQL: A Query Language Supporting Conditional Preferences**. Em of the 2009 ACM SYMPOSIUM ON APPLIED COMPUTING, Proceedings…. p. 1573-1577. ACM, 2009.

AMO, S.; PEREIRA F.S.F. **A Context-Aware Preference Query Language: Theory and Implementation**. Relatório Técnico, Universidade Federal de Uberlândia, Escola de Computação, 2011.

ANDREOU, A. **Ontologies and Query expansion**. Master of Science, School of Informatics, University of Edinburgh, 2005.

ARENS, Y.; KNOBLOCK, C. A.; SHEN, W. **Query reformulation for dynamic information integration**. p. 11-42, Springer US, 1996.

BAEZA-YATES, R.; RIBEIRO NETO, B. **Modern Information Retrieval: The Concepts and Technology behind Search**, 2nd Edition. ACM Press Books, 2010.

BALDAUF M.; DUSDAR S.; ROSEMBERG F.: **A Survey on Context-Aware Systems**. Int. Journal of Ad. Hoc and Ubiquitous Computing, v. 2, n. 4, p. 263-277, 2007.

BETTINI, C. et al. **A survey of Context Modelling and Reasoning Techniques**. Pervasive and Mobile Computing, v.6, n.2, p. 161–180, 2009.

BOLCHINI, C. et al.: **A data-oriented survey of context models**. SIGMOD Record, p. 19-26, 2007a.

BOLCHINI, C; QUINTARELLI, E; ROSSATO R. **Relational Data Tailoring Through View Composition**. ER2007, LNCS 4801. Proceedings… p 149-164, 2007b.

BOLCHINI, C. et al. **And what Can Context Do for Data?** Communications of the ACM, v.52, n.11, p. 136–140. 2009.

BOLCHINI, C.; QUINTARELLI, E TANCA, L. **CARVE: Context-aware automatic view definition over relational databases**. Information Systems 38. p. 45-67. 2013.

BOSC, P.; HADJALI, A.; PIVERT O. **Relaxation paradigm in a flexible querying context**. FQAS, volume 4027 ,Lecture Notes in Computer Science, p. 39–50, 2006.

BORGIDA, A.; BRACHMAN, R. **Loading data into description reasoners**. Em: SIGMOD, Proceedings… p. 217–226, 1993.

BÖRZSÖNYI S.; KOSSMAN D.; STOKER K. **The Skyline Operator**. Em: Data Engineering, 2001. 17th International Conference on. Proceedings...IEEE, p. 421-430, 2001.

BRÉZILLON, P. **Context in problem solving: A survey**. The Knowledge Engineering Review, v. 14, n. 1, p. 1-34, 1999.

BRÉZILLON, P. **Representation of Procedures and Practices in Contextual Graphs**. The Knowledge Engineering Review, v. 18, n. 2, p. 147-174, 2003.

BRÉZILLON, P.; POMEROL, J.-C.: **Contextual Knowledge Sharing and Cooperation in Intelligent Assistant Systems**. Le Travail Humain, PUF, Paris, v. 62, n. 3, p. 223-246, 1999.

BRUNO N.; CHAUDHURI S.; GRAVANO L. **Top-k Selection Queries Over Relational Databases: Mapping Strategies and Performance Evaluation**. TODS, v. 27, n. 2, p. 153-187, 2002.

BUNNINGEN A.H. VAN; FENG L; APERS P. **Context for Ubiquitous Data Management**. Em: 2005 International Workshop on Ubiquitous Data Management UDM'05, IEEE, Proceedings…, p. 17-24, 2005.

BUNNINGEN A.H. VAN; FENG L.; APERS P. **A Context-Aware Preference Model for Database Querying in Ambient Intelligent Environment**. Em: Database and Expert Systems Applications DEXA, LNCS 4080, Proceedings…, p. 33-43, 2006.

BUNNINGEN A. **Context-aware Querying, Better Answers with Less Effort**. Tese de Doutorado, Centre for Telematics and Information Technology (CTIT) Enschede, The Nethelands, 2008.

BUVAČ S.; MASON I.A. **Propositional Logic of Context**. Eleventh National Conference on Artificial intelligence. Proceedings…, Menlo Park, California: AAAI Press, p. 412-419, 1983.

CHALMERS, M. **A historical view of context**. Computer Supported Cooperative Work 13, 3, p.223–247, 2004.

CHAN, C.-Y. et al. **Finding k-Dominant Skylines in High Dimensional Space.** ACM International Conference on Management of Data, SIGMOD, Proceedings… p. 503-514, 2006.

CHAUDHURI, S. **An overview of query optimization in relational systems**. Em: of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems, Proceedings... p. 34-43, ACM, 1998.

CHAUDHURI, S.; GRAVANO. L. **Evaluating Top-K Selection Queries**. Em: of the International Conference on Very Large Data Bases, VLDB, Proceedings… p. 397-410, 1999.

CODD, E. F. **A relational model of data for large shared data banks** CODD, Communications of the ACM, v. 13, n. 6, p. 377-387, 1970.

CODD, EDGAR F.: **Extending the database relational model to capture more meaning.** Em ACM *Transactions on Database Systems* (TODS) v. 4. N. 4. p. 397-434, 1979.

CHOMIKI J. **Preferences formulas in relational queries**. ACM Trans Database System, v.28, n.4, p. 47-466, 2003.

CLARK, H. H.; CARLSON, T. B. **Context for Comprehension**. Attention and Performance IX, eds. J. Long and A. Baddeley, p. 313–330. Hillsdale, N.J.: Lawrence Erlbaum, 1981.

COUTAZ J. et al. **Context is Key**. Communications of the ACM, v. 48, n. 3, p. 49-53, 2005.

DAVIS, R.; KING, J. **An overview of production systems**. Stanford Univ CA Dept of Computer Science, 1975.

DEY, A. K.: **Providing Architectural Support for Building Context-Aware Applications**, Ph.D. Thesis, Georgia Institute of Technology, 2000.

DEY, A. K.; ABOWD, G. D. **A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications**. Human-Computer Interaction, v. 16, n. 2-4, p. 97-166, 2001.

DIAS, V. **CPrefSQL-Tool: Uma Ferramenta Web para Consultas com Suporte a Contextos e Preferências do Usuário**. SBBD 2012. São Paulo, 2012.

ELMASARI R.; NAVATHE E. **Data Warehousing and Data Mining**. Fundamentals of Database Systems, Ed. Adison-Wesley, p. 841-872, 2000.

ELMONGUI H.; AREF W.; MOKBEL M. **Chameleon: Context-Awareness inside DBMSs**. Em ICDE'09, p. 1335-1338, 2009.

FENG L.; APERS P.; JONKER W. **Towards Context-Aware Data Management for Ambient Intelligence**.Springer Berlin Heidelberg, p. 422-431, DEXA, 2004.

GAASTERLAND, T. **Cooperative answering through controlled query relaxation**. IEEE expert, v. 12, n. 5, p. 48-59, 1997.

GRUMBACH, S.; TOVA M. **Towards tractable algebras for bags.** Journal of Computer and System Sciences 52, p. 570-588, 1996.

GODFREY, P.; SHIPLEY, R.; GRYZ, J. **Maximal Vector Computation in Large Data Sets**. Em: of the 31st International Conference on Very Large Data Bases (VLDB), Proceedings... p. 229–240, Trondheim, Noruega, 2005.

HALEVY, A.; RAJARAMAN, A.; ORDILLE, J. **Data integration: the teenage years**. Em: of the 32nd international conference on Very large data bases, Proceedings… VLDB Endowment. p. 9-16, 2006.

HALPIN T.A. **Conceptual Schema and Relational Database Design**, 2nd ed. Prentice Hall Australia, Sydney, 1995.

HALPIN T.A. **Information Modeling and Relational Databases: Em "Conceptual Analysis to Logical Design**, Morgan Kaufman, San Francisco, 2010.

HENRICKSEN K., **A framework for context-aware pervasive computing applications**, Ph.D. thesis, School of Information Technology and Electrical Engineering, The University of Queensland, 2003.

HENRICKSEN, K.; INDULSKA, J.; RAKOTONIRAINY, A. **Generating Context Management Infrastructure from High-Level Context Models**. 4th International Conference on Mobile Data Management (MDM) – Industrial Track, Melbourn, 2003.

HENRICKSEN, K.; LIVINGSTONE, S.; INDULSKA, J. **Towards a Hybrid Approach to Context Modelling, Reasoning and Interoperation**, Em: of the 1st International Workshop on Advanced Context Modelling, Reasoning and Management, Proceedings… p. 54-61, Nottingham, UK, 2004.

HENRICKSEN K., INDULSKA J. **Modelling and using imperfect context information**. Em: of 1st Workshop on Context Modeling and Reasoning (CoMoRea), PerCom Workshop, Proceedings…, IEEE Computer Society, p. 33-37, 2004.

HENRICKSEN, K.; INDULSKA J. **Developing Context-Aware Pervasive Computing Applications: Models and Approach**. Pervasive and Mobile Computing 2, p. 37-64, 2006.

HONG, J. I.; LANDAY, J. A. **An Architecture for Privacy-Sensitive Ubiquitous Computing**, Em: of the 2nd International Conference on Mobile Systems, Applications, and Services, Proceedings… p. 177-189. ACM, 2004.

INMON, W. H.: **Building the Data Warehouse**. Ed. Wiley, 1992.

IOANNIDIS, Y. E. **Query optimization**. ACM Computing Surveys (CSUR), v. 28, n. 1, p. 121-123, 1996.

JANSEN, B. J.; BOOTH, D. L.; SPINK, A. **Patterns of query reformulation during Web searching**. Journal of the american society for information science and technology, v. 60, n. 7, p. 1358-1371, 2009.

JÄRVELIN, K.; KEKÄLÄINEN, J. **IR evaluation methods for retrieving highly relevant documents**. Em: of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, Proceedings…. p. 41-48. ACM, 2000.

JONES R. et al. **Generating query substitutions**. L. Carr, D. D. Roure, A. Iyengar, C. A. Goble, and M. Dahlin, editors, WWW, p. 387–396. ACM, 2006.

KENT, A. et al. **Machine literature searching VIII. Operational criteria for designing information retrieval systems.** American documentation, v. 6, n. 2, p. 93-101, 1955.

KIEßLING, W.; KÖSTLER, G. **Preference SQL - design, implementation, experiences**. VLDB. p. 990–1001, 2002.

KIEßLING, W. **Foundations of preferences in database systems.** Em: of the 28th international conference on Very Large Data Bases. VLDB Endowment, Proceedings…, 2002.

KIEßLING W.; ENDRES M.; WENZEL F. **The Preference SQL System – An Overview**. Bulletin of the IEEE Computer Society Technical Committee on Data Engineeering, IEEE Data Eng. Bull., v. 34, n. 2, p. 11-18, 2011.

KORPIPÄÄ, P. et al. **Managing Context Information in Mobile Devices**, IEEE Pervasive Computing, v. 2, n. 3, p. 42-51, 2003.

KOUTRIKA, G.; IOANNIDIS, Y. **Personalized queries under a generalized preference model.** Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on. IEEE, p. 841-852, 2005.

KOUTRIKA, G.; IOANNIDIS, Y. **Personalization of Queries in Database systems**. Em of 20th Intl. Conf. ICDE, Proceedings… p. 597-608, 2004.

KOUTRIKA, G.; PITOURA, E.; STEFANIDIS, K. **Preference-Based Query Personalization.** Advanced Query Processing. Springer Berlin Heidelberg, p. 57-81, 2013.

KRAFT, R. et al. **Searching with context**. Em: of the 15th international conference on World Wide Web, Proceedings… p. 477-486, ACM, 2006.

LEECH, G; THOMAS J.: **Language, Meaning and Context: Pragmatics**, An encyclopaedia of language, p. 173-243, 1990.

LENZERINI, M. **Data integration: A theoretical perspective**. Em: of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, Proceedings. p. 233-246, ACM, 2002.

LEVANDOSKI J. J.; MOKBEL M. F.; KHALEFA M. E. **CareDB: A Context and Preference-Aware Location-Based Database System**. Em: of the VLDB Endowment, Proceedings… v. 3 n. 1-2, p. 1529-1532, 2010a.

LEVANDOSKI, J. J.; MOKBEL M. F.; KHALEFA. M. E. **FlexPref: A framework for extensible preference evaluation in database systems**. Em Data Engineering (ICDE), IEEE 26th International Conference on. IEEE, p. 828-839, 2010b.

LEVANDOSKI J.; KHALEFA M.E.; MOKBEL M. F. **An Overview of the CareDB Context and Preference-Aware Database Sysem**. Em Bulletin of the IEEE Computer Society Technical Committee on Data Engineeering, v. 34, n. 2, p. 41-46, 2011.

LI, C. et al. **RankSQL: query algebra and optimization for relational top-k queries**. Em: of the 2005 ACM SIGMOD international conference on Management of data, Proceedings… p. 131-142. ACM, 2005.

LI, D.; HAN, L.; DING, Y. **SQL query optimization methods of relational database system**. Computer Engineering and Applications (ICCEA), Second International Conference. IEEE. p. 557-560, 2010.

MARTIN, JAMES.: ***Information Engineering,*** Book I: Introduction". 1st edition. Ed. Prentice Hall, 1989.

McCARTHY J. **Generality in Artificial Intelligence**. Em: Communications of the ACM v.30, n.12, p. 1030-1035, 1987.

McCARTHY J. **Notes on Formalizing Context**. Em: of the Thirteenth International Joint Conference on Artificial Intelligence, Proceedings… p. 555-560, 1993.

MISHRA, C.; KOUDAS, N. **Interactive query refinement**. Em: of the 12th International Conference on Extending Database Technology: Advances in Database Technology, Proceedings… ACM, p. 862-873, 2009.

MORSE, D. R.; ARMSTRONG, S.; DEY, A. K. **The what, who, where, when, why and how of context-awareness**. CHI'00 extended abstracts on Human factors in computing systems. p. 371-371, ACM, 2000.

NECIB, C. B; FREYTAG J. C. **Using Ontologies for Database Query Reformulation.** ADBIS (Local Proceedings), v. 4, p. 1-1, 2004.

NEWELL A.: **A Production Systems: Models of Control Structures**. Visual Information Processing, Willian G. Chase (ed.), p. 463-526, Academic Press, 1973.

OXFORD DICTIONARY 2015: Em www.oxforddictionaries.com. Acesso em 11 jan. 2015.

PEREIRA, F. **CPref-SQL: uma linguagem de consulta com suporte a preferências condicionais-teoria e implementação.** Dissertação de Mestrado, Univ. Federal de Uberlândia-MG, 2011.

PERERA C. et al. **Context aware computing for the internet of things: A Survey**. Em Communications Surveys Tutorials, IEEE, v. 16, n. 1, p. 414-454, 2014.

PERKINS, D. N.; SALOMON, G.: **Are cognitive skills context-bound?** Educational researcher, v. 18, n. 1, p. 16-25, 1989.

PITOURA, E.; STEFANIDIS, K; ZAVLAVSKY, A. **Context in Databases**. University of Ioannina Technical Report, Grécia, 2004.

PITOURA, E.; STEFANIDIS, K.; VASSILIADIS, P. **Contextual Database Preferences**. IEEE Computer Society Bulletin of Technical Committee on Data Engineering, v. 34, n. 2, p. 19-26, 2011.

PREISINGER T.; KIEßLING. W. **The Hexagon Algorithm for Evaluating Pareto Preference Queries**. Em: Communications Surveys & Tutorials, IEEE, v. 16, n. 1, p. 414-454, 2014.

POWER, R.: **Topic Maps for Context Management**, Em: Workshop on Adaptive Systems for Ubiquitous Computing at the 1st International Symposium on Information and Communication Technologies, Proceedings…. p. 199-204, 2003.

ROTH, M. A.; HERRY F. K.; SILBERSCHATZ, A. **Extended algebra and calculus for nested relational databases.** ACM Transactions on Database Systems (TODS) 13.4 p. 389-417, 1988.

ROUSSOS I.; STAVRAKAS Y.; PAVLAKI V. **Towards a Context-Aware Relational Model**. Em: of the Contextual Representation and Reasoning Workshop of the 5th International and Interdisciplinary Conference on Modeling and Using Context, Proceedings… (CONTEXT'05), Paris, France, 2005.

ROUSSOS I; SELLIS, T. **A Model for Context Aware Relational Databases**. Knowledge and Database Systems Laboratory, School of Electrical and Computer Engineering, National Technical University of Athens Technical Report, Greece, 2008.

RUSSELL, S.; NORVIG, P. **Artificial Intelligence: A Modern Approach**, 3rd Edition, Pearson Education, 2014.

SARACEVIC, T. **Relevance: A review of and a framework for the thinking on the notion in information science**. Journal of the American Society for Information Science, v. 26, n. 6, p. 321-343, 1975.

SARACEVIC, T. **Relevance: A review of the literature and a framework for thinking on the notion in information science. Part II: Nature and Manifestations of Relevance.** Journal of the American Society for Information Science and Technology, v. 58, n. 13, p. 1915-1933, 2007a.

SARACEVIC, T. **Relevance: A review of the literature and a framework for thinking on the notion in information science. Part III: Behavior and effects of relevance.** Journal of the American Society for Information Science and Technology, v. 58, n. 13, p. 2126-2144, 2007b.

SCHILIT, B.; ADAMS, N.; WANT, R. **Context-Aware Computing Applications**, Em: Proc. Workshop on Mobile Computing Systems and Applications,; p. 85-90. Santa Cruz, CA; 1994.

SERAFINI L.; BOUQUET P. **Comparing formal Theories of Context in AI**. Artificial Intelligence, 155(1-2): p. 41-67, 2004.

SCHAMBER, L. et al. **A re-examination of relevance: toward a dynamic, situational definition.** Information processing & management, v. 26, n. 6, p. 755-776, 1990.

SHEN, X.; TAN, B.; ZHAI, C. **Context-sensitive information retrieval using implicit feedback**. Em: of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, Proceedings… p. 43-50. ACM, 2005.

SOUZA, D., et al. **Towards a context ontology to enhance data integration processes**. VLDB 08. ACM, p. 49-56, 2008.

SQL ISO 1992: ISO/IEC 9075: 1992 - Information technology - Database languages - SQL. Revisão informal. Disponível em: http://www.contrib.andrew.cmu.edu/~shadow/sql/ sql1992.txt. Acesso em: 25 jul. 2014.

SQL ISO 2003: Em http://www.iso.org/iso/iso_catalogue/catalogue_ics/ catalogue_detail_ics.htm?csnumber=34133. Acesso em: 25 jul. 2014.

SQL ISO 2008: Em http://www.iso.org/iso/iso_catalogue/ catalogue_detail_ics.htm?csnumber=45499. Acesso em: 25 jul. 2014.

STEFANIDIS K.; PITOURA E; VASSILIADIS P. **Adding Context to Preferences**. Em 23rd IEEE International Conference on Data Engineering, p. 846-855, 2007.

STEFANIDIS K.; KOUTRIKA G., PITOURA E. **A survey on representation, composition and application of preferences in database systems**. ACM Trans. Database System, v. 36, n.4, p. 19, 2011.

STRANG T.; LINNHOF-POPIEN, C.: **A Context Modeling Survey**, Em: of the Workshop on Advanced Context Modelling, Reasoning and Management, Proceedings…, 6th International Conference on Ubiquitous Computing, Nottinghan/England, 2004.

THEODORAKIS, M. et al.: **A theory of contexts in information bases.** Information Systems, v.27, n.3, p. 151-191, 2002.

TRUONG, K. N.; ABOWD, G. D.; BROTHERTON, J. A. **Who, What, When, Where, How: Design Issues of Capture & Access Applications**, Em: of the International Conference on Ubiquitous Computing, Proceedings… p. 209-224, 2001.

USCHOLD, M.; GRUNINGER. M. **Ontologies: Principles, methods and applications**. Knowledge engineering review, v. 11, n. 02, p. 93-136, 1996.

VIEIRA V.: **CEManTIKA: A Domain-Independent Framework for Designing Context-Sensitive Systems**. Tese de Doutorado, Centro de Informática, Universidade Federal de Pernambuco – Brasil, 2008.

VIEIRA V.; TEDESCO P.; SALGADO A.C. **Modelos de Processos para o Desenvolvimento de Sistemas Sensíveis ao Contexto**. Jornadas de Atualização em Informática (JAI'09), p. 381-431; 2009.

VIEIRA, V.; TEDESCO, P.; SALGADO, A. C.: **Designing Context-Sensitive Systems: An integrated Approach**. Expert Systems with Applications, v. 38, n. 2, p. 1119–1138, 2011.

VILAR, B. **Processamento de Consultas Baseado em Ontologias para Sistemas de Biodiversidade**. UNICAMP, Dissertação de mestrado, 2008.

WANG, X. H. et al. **Ontology Based Context Modeling and Reasoning using OWL**. Em: of the 2nd IEEE Conference on Pervasive Computing and Communications (PerCom2004), Proceedings… p. 18–22. Orlando, FL, USA; 2004.

YAGUINUMA, C. A.: **Sistema FOQuE para Expansão Semântica de Consultas Baseada em Ontologias Difusas.** Dissertação de mestrado em ciência da computação, Centro de Ciências Exatas e de Tecnologia, Universidade Federal de São Carlos, São Carlos; 2007.

YAGUINUMA C. A.; BIAJIZ, M.; SANTOS, M. T. P. **Sistema foque para expansão semântica de consultas baseada em ontologias difusas.** Em XXII Simpósio Brasileiro de Banco de Dados, p. 208–222, João Pessoa, PB; 2007.

YIU M. L.; MAMOULIS N. **Efficient Processing of Top-k Dominating Queries on Multi-Dimensional Data**. Em: of the International Conference on Very Large Data Bases, Proceedings… p. 483-494, VLDB, 2007.

ZHOU, Xuan et al. **Query relaxation using malleable schemas**. Em: of the 2007 ACM SIGMOD international conference on Management of data, Proceedings... p. 545-556. ACM, 2007.

APÊNDICE A – Algoritmos das DRC

Neste apêndice, são apresentados os pseudocódigos dos algoritmos complementares das diretivas para reescrita de consultas apresentadas no Capítulo 4.

A.1 DRC Project

O algoritmo de aplicação da DRC PROJECT encontra-se listado na Figura A.1.

Figura A.1 – O Algoritmo Project

Project

```
Entrada: uma consulta Q,
         uma  DRC do tipo Project;
Saída: uma consulta reescrita Q';

01: recebe Q;
02: recebe DRC;
03: Q':=Q;

04: PARA [NOT]E_i.a_i (1..n) de L
05: SE E_i.a_i possui NOT                                   /* remoção de projeção
06:       SE E.a é atributo único na cláusula de projeção de Q`
07:           substitui E.a por 'consulta não permitida';
08:       SENÃO
09:           remove E_i.a_i da cláusula de projeção de Q` (Select);
10:       FIM SE;
11: SENÃO                                                   /* inclusão de projeção
12:     SE E_i já está presente em Q'
13:         inclui E_i.a_i na cláusula de projeção de Q` (Select);
14:     SENÃO
15:         verifica, em BDEC, os relacionamentos entre E_i e entidades de Q';
16:         SE há relacionamento direto
17:             realiza junção entre E_i e entidades de Q' relacionadas a E_i;
18:             /* insere E_i na cláusula From e condição de junção na cláusula SQL Where
19:             inclui E_i.a_i na cláusula de projeção de Q` (cláusula SQL Select););
20:         SENÃO
21:             SE há relacionamento indireto (tabela de relacionamento)
22:                 realiza junções entre E_i e entidades de Q' que possuam relacionamento indireto com E_i;
23:                 /* insere entidades na cláusula From e condições de junção na cláusula SQL Where
24:                  acrescenta E_i.a_i na projeção de Q` (cláusula SQL Select)
25:             SENÃO
26:                 ignorar;
27:             FIM-SE
28:         FIM SE;
29:     FIM SE
30: FIM SE;
31: FIM PARA;
32: retorne (Q');
```

Fonte: O autor.

A.2 DRC Filter

O algoritmo *Filter* encontra-se listado em pseudocódigo na Figura A.2 e seus algoritmos complementares *Filter_Sentence* e *Filter_Exists* podem ser consultados nas Figuras A.3 e A.4;

Figura A.2 – O Algoritmo Filter

Filter

```
Entrada: uma consulta Q,
         uma  DRC do tipo Filter;
Saída: uma consulta reescrita Q';

01: recebe Q;
02: recebe DRC;
03: Q' := Q;
04:
05: Realiza análise da DRC;                  /* --- Separa sentenças Ci e operadores lógicos da DRC
06: PARA as sentenças Ci (1..n)  contidas na DRC
07:     Filter_Sentence (Q', Ci, nulo, nulo);
08: FIM PARA;
09:
10: Retorne (Q');
```

Fonte: O autor.

Figura A.3 – O Algoritmo Filter_Sentence

Filter_Sentence

```
Entrada: uma consulta Q';
         uma sentença C;
         um  operador (OP_juncao) para junção explícita (opcional);
         uma Entidade.atributo (EA_juncao) para junção explícita (opcional);
Saída: uma consulta reescrita Q';

01:   recebe Q', C ,  OP_juncao, EA_juncao;
02:
03:   Realiza análise da sentença C recebida          */ --- Decompõe C  recebida em 1 ou mais C i
04:
05:   PARA todas as sentenças C i (1..n) contidas em C
06:
07:     Decompor Ci   em: operando tipo <E.a> (Entidade.atributo a ser restrito),
08:                       <oc> (operador condicional),
09:                       <vl> (critério de restrição)
10:
11:   /* --- Realiza substituição de parâmetros atribuídos à entidades externas em tempo de execução (&E.a)
12:   /* --- (ex. substituir &Device.name por 'google glass').
13:
14:     SE há operando tipo &E.a em Ci
15:        realizar substituição de &Entidade.atibuto por valor instanciado;
16:     FIM SE;
17:
18:   /* --- Processa EXISTS
19:
```

```
20:     SE Ci tem EXISTS
21:           Q'' := Filter_Exists (Ci);
22:           Q' := Q' || <where/and> || 'EXISTS (' || Q''  || )';    /* --- Concatena subconsulta do exists em Q',
23:     SENÃO
24:
25:   /* --- Processa junção
26:
27:           SE a entidade <E> de Ci não está contida em Q'                    /* --- Demanda junção
28:               verifica os relacionamentos da entidade de Ci na BDEC;
29:               SE há relacionamento direto entre <E> e entidade de Q'        /* --- Junção direta
30:                   SE EA_juncao < > nulo                                     /* --- explícita
31:                       realiza junção em Q' entre <E> e entidades de Q' usando atributo EA_juncao
32:                       (insere entidades no from e atributos no where);
33:                   SENÃO                                                     /* --- implícita
34:                       realiza junção em Q' entre <E> e a entidades de Q'
35:                       (insere entidades no from e atributos no where);
36:                   FIM SE;
37:               SENÃO                                                         /* --- Junção indireta (n x n)
38                                                                              /* --- implícita
39:                   SE há relacionamento indireto (tabela de relacionamento) entre E e entidade de Q'
40:                       realiza junção em Q' entre <E> e entidades de Q' que possuam relacionamento
41:                       indireto (insere entidade na cláusula from e condições de junção na cláusula where);
42:                   SENÃO
43:                       Erro! DRC não aplicável ();                           /* --- Erro produto cartesiano
44:                       Retorne (Erro!);
45:                   FIM SE;
46:               FIM SE;
47:           FIM SE;
48:
49:   /* ---- Realiza atribuição da restrição ( E.a <oc> <vl> )
50:
51:       SE <vl> utiliza função de agregação γ
52:           Q_agreg =  '(Select' || γ (E.a) || ' from ' || E || ` where ` || E.a || <oc> || <vl> || `)`;
53:           Q' := Q' || <where/and> ||  E.a ||  <oc> || Q_agreg;
54:       SENÃO
55:           SE <vl> é do tipo sentença C (formato E.a <oc> (C))               /* --- <vl> é tipo C
56:               OP_juncao := <oc>;                                            /* --- gera recursão
57:               EA_juncao := <E.a>;
58:               Q' := Filter_Sentence (Q', <vl>, OP_juncao, EA_juncao);
59:               OP_juncao := nulo;
60:               EA_juncao := nulo;
61:           SENÃO
62:               SE <vl> é do tipo E.a                                         /* --- <vl> é atributo E.a
63:                   aplica restrição em Q', Q' := Q' || <where/and>  || E.a ||  <oc> || <vl>
64:               SENÃO
65:                   SE <vl> contém 'NULL'                                     /* --- <vl> é 'null'
66:                       SE <oc> ´=´
67:                           Q' := Q' || <where/and>  || E.a ||  ` IS NULL`  /* aplica restrição em Q',
68:                       SENÃO
69:                           Q' := Q' || <where/and>  || E.a ||  ` IS NOT NULL`  /* aplica restrição em Q',
70:                       FIM SE;
71:                   SENÃO                                                     /* --- <vl> é literal ou numérico
72:                       aplica restrição em Q', Q' := Q' || <where/and>  || E.a ||  <oc> || <vl>
73:                   FIM SE;
74:           FIM SE;
75:       FIM SE;
76:
77:     FIM SE;
78:   FIM PARA;
79:
80:   Retorne (Q');
```

Fonte: O autor.

Figura A.4 – O Algoritmo Filter_Exists

Filter_Exists

```
Entrada: uma sentença C;
Saída: uma consulta reescrita Q'';

01:  recebe C
02:
03:  Realiza análise da sentença C recebida;                    */ --- Decompõe C recebida em 1 ou mais Ci
04:
05:  PARA todas as sentenças Ci (1..n) contidas em C
06:
07:    Decompor C em: operando <E.a> (Entidade.atributo a ser restrito),
08:                                <oc> (operador condicional),
09:                                <vl> (critério de restrição)
10:
11:    /* --- Realiza substituição de parâmetros atribuídos à entidades externas em tempo de execução (&E.a)
12:    /* --- (ex. substituir &Device.name por 'google glass').
13:
14:      SE há operando tipo &E.a em Ci
15:          realizar macro substituição de &Entidade.atibuto por valor instanciado;
16:      FIM SE;
17:
18:  /* --- Monta Sql base da subconsulta EXISTS, obrigatoriamente determinado pela primeira sentença (C1)
19:
20:   SE i=1
21:       SE Ci tem EXISTS
22:           Erro! Sintaxe inválida 'EXISTS (EXISTS'
23:           Retorne (Erro!);
24:       SENÃO
25:           Q'':= 'SELECT * FROM' || <Entidade de  C1> || ' WHERE ' || <Entidade.atibuto de C1> ;
26:           SE <vl> contém 'NULL'
27:               SE <oc> ´=´
28:                   Q'':= Q'' || 'IS NULL';
29:               SENÃO
30:                   Q'':= Q'' || 'IS NOT NULL';
31:               FIM SE;
32:           SENÃO
33:               SE <vl> utiliza função de agregação γ
34:                   Q_agreg =  '(Select' || γ (E.a) || ' from ' || E || ` where ` || E.a || <oc> || <vl> || `)`;
35:                   Q'':= Q'' || <where/and> ||  E.a ||  <oc> || Q_agreg;
36:               SENÃO
37:                   Q'':= Q'' ||  <oc> || <vl> ;
38:               FIM SE;
39:           FIM SE;
40:       FIM SE;
41:   SENÃO
42:
43:   /* --- Processa EXISTS (em sentença diferente de C1)
44:
45:       SE Ci tem EXISTS
46:           Q_aux := Filter_Exists (Ci);
47:           Q':= Q' || <where/and> || 'EXISTS (' || Q''  || )';    /* --- Concatena subconsulta do exists em Q',
48:       SENÃO
49:
50:   /* --- Processa junção
51:
52:           SE a entidade <E> de Ci não está contida em Q''                    /* --- Demanda junção
53:               verifica os relacionamentos da entidade de Ci na base de metadados;
54:               SE há relacionamento direto entre <E> e entidade de Q''              /* --- Junção direta
55:                   SE EA_juncao < > nulo                                              /* --- explícita
56:                       formatar junção em Q'' entre <E> e entidades de Q'' usando atributo EA_juncao
57:                      (insere entidades no from e atributos no where);
58:                   SENÃO                                                              /* --- implícita
59:                       formatar junção em Q'' entre <E> e a entidades de Q''
```

```
60:                         (insere entidades no from e atributos no where);
61:                     FIM SE;
62:                 SENÃO                                                   /* --- Junção indireta  (n x n)
63                                                                              /* --- implícita
64:                     SE há relacionamento indireto (tabela de relacionamento) entre E e entidade de Q''
65:                         realiza junção em Q'' entre <E> e entidades de Q'' que possuam relacionamento
66:                         indireto (insere entidade na cláusula from e condições de junção na cláusula where);
67:                     SENÃO
68:                         DRC não aplicável ();                           /* --- Erro produto cartesiano
69:                        Retorne (Erro!);
70:                     FIM SE;
71:                 FIM SE;
72:             FIM SE;
73:
74:     /* ---- Realiza atribuição da restrição ( E.a <oc> <vl> )
75:
76:         SE <vl> utiliza função de agregação γ
77:             Q_agreg =  '(Select' || γ (E.a) || ' from ' || E || ` where ` || E.a || <oc> || <vl> || `)`;
78:             Q'' := Q'' || <where/and> ||  E.a ||  <oc> || Q_agreg;
79:         SENÃO
80:             SE <vl> é do tipo sentença C (formato E.a <oc> (C))         /* ------ <vl> é tipo C
81:                 OP_juncao := <oc>;
82:                 EA_juncao := <E.a>;
83:                 Q'' := Filter_Sentence (Q'', <vl>, OP_juncao, EA_juncao);
84:                 OP_juncao := nulo;
85:                 EA_juncao := nulo;
86:             SENÃO
87:                 SE <vl> é do tipo E.a                                   /* ---- <vl> é atributo E.a
88:                         aplica restrição em Q'', Q'' := Q'' || <where/and>  || E.a ||  <oc> || <vl>
89:                 SENÃO
90:                     SE <vl> contém 'NULL'                               /* --- <vl> = 'null'
91:                         SE <oc> ´=´
92:                             aplica restrição em Q'', Q'' := Q'' || <where/and>  || E.a ||  ` IS NULL`
93:                         SENÃO
94:                             aplica restrição em Q'', Q'' := Q'' || <where/and>  || E.a ||  ` IS NOT NULL`
95:                         FIM SE;
96:                     SENÃO                                               /* --- <vl> é literal ou numérico
97:                         aplica restrição em Q'', Q'' := Q'' || <where/and>  || E.a ||  <oc> || <vl>
98:                     FIM SE;
99:                 FIM SE;
100:            FIM SE;
101:
102:        FIM SE;
103:    FIM PARA;
104:
105:    Retorne (Q');
```

Fonte: O autor.

Observação:

- A sintaxe Filter (EXISTS (EXISTS E.a <oc> <vl>) não faz sentido e será criticada. Já Filter (EXISTS (E.a <oc> <vl> AND EXISTS (E.a <oc> <vl>)) é válida.

A.2 DRC Order

Os algoritmos Order, Orderby e Orderover, relativos à DRC Order, encontram-se listados nas Figuras A.5, A.6 e A.7.

Figura A.5 – O Algoritmo Order

Order

```
Entrada: uma consulta Q,
            uma  DRC do tipo Order;
Saída: uma consulta reescrita Q';

01: recebe Q;
02: recebe DRC;
03: Q':=Q;

04: Realiza análise da DRC;                    /* --- Separa sentenças τi

05: SE DRC possui LIMIT                        /* --- ordenação com cláusula over
06:       Orderover (Q');
07: SENÃO
08:       Orderby (Q');                        /* --- ordenação com cláusula order by
09: FIM SE
10:
11: Retorne (Q');
```

Fonte: O autor.

Figura A.6 – O Algoritmo Orderby

Orderby

```
Entrada: uma consulta Q',
            uma  DRC do tipo Order;
Saída: uma consulta reescrita Q'';

01: recebe Q';
02: recebe DRC;
03: Q'':=Q';

04: PARA todas as sentenças τi (1..n) de ordenação contidas na DRC
05:
06:    SE a entidade <E> de τi não está contida em Q''                  /* --- Demanda junção
07:       verifica os relacionamentos da entidade de τi na base de metadados;
08:       SE há relacionamento direto e único entre <E> e entidade de Q''     /* --- Junção direta única
09:             realiza junção em Q'' entre <E> e entidades de Q''
10:             (insere entidades no from e atributos no where);
11:       SENÃO
12:          DRC não aplicável ();                                  /* --- Erro – ordenação para junção não permitida
13:          Retorne (Erro!);
14:       FIM SE;
15:    FIM SE;
16:
17:   Realiza análise da sentença τi ;          */ --- Decompõe τi em E.a,  <ASC/DESC>
18:                                             */ --- ou E.a <oc> <vl> , <posição> <ASC/DESC>
19:   SE i = 1
20:       Q'':= Q'' || ' ORDER BY '
21:   ELSE
22:       Q'':= Q'' || ' , '
```

```
23:   FIM SE;
24
25:   SE sentença tipo  <atributo>                    /* --- ordenação baseada em atributo
26:
27:      Q'':= Q'' || E.a || <criterio>                    /* --- criterio = {ASC/ DESC}
28:
29:   SENÃO                                              /* --- ordenação baseada em valor de atributo
30:                                                      /* --- tipo <valor_atributo>
31:                                                      /* --- aplicação de order by com cláusula CASE
32:
33:   Q'':= Q'' || ' CASE WHEN ' || E.a || <oc> || <vl>  || ' THEN ' ||  <arg> ||
34:
35:      SE i = n e existe cláusula ELSE em τi
36:         Q'':= Q'' || ' ELSE' || < pos> || ' END ' || <critério>;
37:      FIM SE;
38:
39:   FIM SE;
40: FIM PARA;
41:
42: Retorne (Q'');
```

Fonte: O autor.

Figura A.7 – O Algoritmo Orderover

Orderover

```
Entrada: uma consulta Q',
            uma  DRC do tipo Order;
Saída: uma consulta reescrita Q'';

01: recebe Q';
02: recebe DRC;
03: Q'':=Q';

04: PARA todas as sentenças τi (1..n) de ordenação contidas na DRC
05:
06:   Realiza análise da sentença τi  recebida;          */ --- Decompõe τi
07:
08:   SE i = 1
09:      Q'':= 'SELECT ' || < atributos E.a de Q>  ||  ' FROM ( SELECT ' || < atributos E.a de Q>  || ', ' ||
10:            <critério de rank> || '( ) OVER ( ORDER BY ' ||
11:   ELSE
12:      Q'':= Q'' || ' , '
13:   FIM SE;
14:
15:   SE sentença tipo  <entity_order>                  /* --- ordenação baseada em atributo
16:
17:      Q'':= Q'' || E.a || <criterio>                    /* --- criterio = {ASC/ DESC}
18:
19:   SENÃO                                              /* --- ordenação baseada em valor de atributo
20:                                                      /* --- tipo <constraint_entity_order>
21:                                                      /* --- aplicação de order by com cláusula case
22:
23:   Q'':= Q'' || ' CASE WHEN ' || E.a || <oc> || <vl>  || ' THEN ' ||  <arg> ||
24:                ' ELSE' || <arg> || ' END ' || <critério>
25:
26:   FIM SE;
27: FIM PARA;
28:
29: Q'':= Q'' || ') AS  crit_rank FROM '
30:
31: SE a entidade <E> de τi não está contida em Q''                /* --- Demanda junção
32:   verifica os relacionamentos da entidade de τi na base de metadados;
33:   SE há relacionamento direto e único entre <E> e entidade de Q''      /* --- Junção direta única
34:      realiza junção em Q'' entre <E> e entidades de Q''
```

```
35:       (insere entidades no from e atributos no where);
36:    SENÃO
37:         DRC não aplicável ();                          /* --- Erro – ordenação para junção não permitida
38:         Retorne (Erro!);
39:    FIM SE;
40: FIM SE;
41:
42: Q’’:= Q’’ || ‘) as  foo WHERE ’ ||  ‘ crit_rank  <= ’ || <vl>
43:
44: Retorne (Q’’);
```

Fonte: O autor.

A.3 DRC Skyline

Esta diretiva é implementada por meio do algoritmo Skyline listado em formato de pseudocódigo na Figura A.8.

Figura A.8 – O Algoritmo Skyline

Skyline

Entrada: uma consulta Q,
uma DRC do tipo Skyline;
Saída: uma consulta reescrita Q’’;

```
01: recebe Q’;
02: recebe DRC;
03: Q’:=Q;
04:  Realiza análise da DRC                                                    */ --- Decompõe DRC
05:                                                                            */ --- monta consulta externa
06:  Q’:= Q’ || ‘ AE WHERE NOT EXISTS (SELECT * FROM  ’ || <E> || ‘ AI WHERE ’
07:                                                                                */  --- AE alias externo
08:                                                                                */  --- AI alias interno
09:
10:  PARA todas as sentenças MAX/MIN (E.a)i  (i:1..n) da lista da DRC          */ --- monta consulta interna
11:
12:    SE i < > 1
13:        Q’:= Q’ || ‘ OR ’;
14:    FIM SE;
15:
16:    SE <criterio>i = MAX
17:       Q’:= Q’ || ‘ AI.’ || ai || ‘  >=  ’ ||  ‘ AE.’ || ai
18:    SENÃO
19:       Q’:= Q’ || ‘ AI.’ || ai || ‘  <=  ’ ||  ‘ AE.’ || ai
20:    FIM SE;
21:
22:    PARA (E.a)i  (i:1..n)
23:      SE i < > j
24:          SE <criterio>i = MAX
25:              Q’:= Q’ || ‘ AND AI.’ || ai || ‘ > ’ ||  ‘AE.’ || ai
26:          SENÃO
27:              Q’:= Q’ || ‘ AND AI.’ || ai || ‘ < ’ ||  ‘AE.’ || ai
28:         FIM SE;
29:    FIM PARA;
30:
31:    AUX_ORDER = AUX_ORDER || <E.a>i ||
32:
33:    SE i < > n
34:       AUX_ORDER = AUX_ORDER ||  ‘ , ’
```

```
35:   FIM SE;
36:
37:  FIM PARA;
38:
39:  Q':= Q' || ' ORDER BY ' || AUX_ORDER
40:
41: Retorne (Q'');
```

Fonte: O autor.

A.4 DRC Group

Os algoritmos Group, Grouby e Groupover, relativos à DRC Group, encontram-se listados em pseudocódigp nas Figuras A.9, A.10 e A.11.

Figura A.9 – O Algoritmo Group

Group

```
Entrada: uma consulta Q,
            uma  DRC do tipo Group;
Saída: uma consulta reescrita Q';

01: recebe Q;
02: recebe DRC;
03: Q':=Q;

04: Realiza análise da DRC;
05: SE DRC possui LIMIT                          /* --- agregação com cláusula over
06:      Groupover (Q',DRC);
07: SENÃO
08:      Groupby (Q',DRC);                       /* --- agregação com cláusula group by
09: FIM SE
10:
11: Retorne (Q');
```

Fonte: O autor.

Figura A.10– O Algoritmo Groupby

Groupby

```
Entrada: uma consulta Q',
            uma  DRC do tipo Group;
Saída: uma consulta reescrita Q'';

01: recebe Q';
02: recebe DRC;
03: Q'':=Q';

04: PARA todas as E.a(1..n) da lista de agregação
05:
06:    SE a entidade <E> de E.ai não está contida em Q''          /* --- Demanda junção
07:       verifica os relacionamentos da entidade na base de metadados;
08:       SE há relacionamento direto e único entre <E> e entidade de Q''   /* --- Junção direta única
09:            realiza junção em Q'' entre <E> e entidades de Q''
10:            (insere entidades na cláusula FROM e atributos na cláusula WHERE);
11:       SENÃO
12:          DRC não aplicável ();                  /* --- Erro – agregação para junção não permitida
```

```
13:          Retorne (Erro!);
14:        FIM SE;
15:     FIM SE;
16:
17: FIM PARA;
18:
19: Inserir ‘ , ’ || <função de agregação> (E.a) na cláusula SELECT;
20:
21: SE <alias> consta na DRC
22:    Inserir ‘ as ’ || <alias> na cláusula SELECT;
23:  FIM SE;
24:
25:  Q’ := Q’ || ‘ GROUP BY ’ || <lista de atributos a serem agregados>;
26:
27: SE <oc> <vl> consta na DRC
28:    Q’ := Q’ || ‘ HAVING  ’ || <função de agregação> || <oc> || <vl>;
29: FIM SE;
30:
31: Retorne (Q’’);
```

Fonte: O autor.

Figura A.11 – O Algoritmo Groupover

Groupover

```
Entrada: uma consulta Q’,
             uma  DRC do tipo Group;
Saída: uma consulta reescrita Q’’;

01: recebe Q’;
02: recebe DRC;
03: Q’’:=Q’;

04: PARA todas as E.a(1..n) da lista de agregação
05:
06:    SE i = 1
07:        Q’’:= ‘SELECT ‘ || < atributos E.a de Q>  ||  ‘ FROM ( SELECT ‘ || < atributos E.a de Q>  || ‘, ’ ||
08:                  <critério de rank> || ‘(E.a) OVER ( PARTITION BY ’ ||
09:    ELSE
10:        Q’’:= Q’’ || ‘ , ’
11:    FIM SE;
12:
13:    Q’’:= Q’’ || E.ai
14:
15:    FIM SE;
16: FIM PARA;
17:
18: Q’’:= Q’’ || ‘) AS  crit_rank FROM ’
19:
20: SE a entidade <E> de τi não está contida em Q’’                          /* --- Demanda junção
21:    verifica os relacionamentos da entidade de τi na base de metadados;
22:    SE há relacionamento direto e único entre <E> e entidade de Q’’       /* --- Junção direta única
23:         realiza junção em Q’’ entre <E> e entidades de Q’’
24:        (insere entidades no from e atributos no where);
25:    SENÃO
26:         DRC não aplicável ();                                  /* --- Erro – ordenação para junção não permitida
27:         Retorne (Erro!);
28:    FIM SE;
29:
30: Q’’:= Q’’ || ‘) as  foo WHERE ’ ||  ‘ crit_rank  >= ’ || <vl>
31:
32: Retorne (Q’’);
```

Fonte: O autor.

A.4 DRC Prefer

O pseudocódigo do algoritmo de aplicação da DRC PREFER encontra-se listado na Figura A.12.

Figura A.12 – O algoritmo Prefer

Prefer

```
Entrada: uma consulta Q,
         uma  DRC do tipo Prefer;
         uma identificação de usuário;
Saída: uma consulta reescrita Q'';

01: recebe Q';
02: recebe DRC;
03: recebe user.id;
04: Q':=Q;

05: Realiza análise da DRC                                   */ --- Decompõe DRC
06:
07: Consultar preferências do usuário (&user.id);            */ --- Consulta preferências do usuário
08: Consultar número mínimo de preferências a serem satisfeitas pela reescrita (kp)
09:                                                          */ --- definido pelo especialista no domínio
10: Consulta preferências no perfil do usuário;
11: Analisa conflitos entre preferências e seleciona conjunto P_i de preferências aplicáveis;
12:
13: /* --- monta consulta externa
14: Q':= 'SELECT ' || <lista de atributos selecionados em Q> || ', AVG (GIC),  COUNT(*) FROM ( '
15:
16: /* --- monta consultas internas
17:
18: PARA todas as preferências P_i (i:1..n)
19:
20:    Q_AUX := Q;
21:
22:   SE <oc> de P_i = IN ou NOT IN                          */ --- Restrição de continência, resolver por EXISTS
23:
24:       SE <oc> de P_i = IN
25:           Q_AUX = Q_AUX ||  ' WHERE EXISTS  ( SELECT * FROM '
26:       SENÃO
27:           Q_AUX = Q_AUX ||  ' WHERE NOT EXISTS  ( SELECT * FROM '
28:       FIM SE;
29:                                                          */ --- Verifica demanda de junção subconsulta do  Exists
30:      SE a entidade <E> de P_i não está contida em Q
31:          verifica os relacionamentos da entidade de P_i na base de metadados;
32:          SE há relacionamento direto entre <E>de P_i e entidade de Q_AUX          /* --- Junção direta
33:              realiza junção em Q_AUX entre <E> e a entidades de Q_AUX
34:             (insere entidade no from e atributos no where);
35:              insere projeção adicional na cláusula select de Q_AUX:  ' , '  || < rank de P_i >  || ' as GIC';
36:          SENÃO                                                                    /* --- Junção indireta (n x m)
37:              SE há relacionamento indireto (tabela de relacionamento) entre E e entidade de Q''
38:                  realiza junção em Q_AUX entre <E> e entidades de Q_AUX que possuam relacionamento indireto
39:                  (insere entidades na cláusula from e condições de junção na cláusula where);
40:              insere projeção adicional na cláusula select de Q_AUX:  ' , '  || < rank de P_i * 0,9 >  || ' as GIC'
41:              SENÃO
42:                  DRC não aplicável ();                 /* --- Erro – restrição para junção inexistente
43:                  Retorne (Erro!);
44:          FIM SE;
45:
46:   SENÃO                                                  */ --- Restrição de atributo, resolver por Junção
47:      SE a entidade <E> de P_i não está contida em Q_AUX
48:          verifica os relacionamentos da entidade de P_i na base de metadados;
```

```
49:          SE há relacionamento direto entre <E>de Pi e entidade de Q_AUX                /* --- Junção direta
50:               realiza junção em Q_AUX entre <E> e a entidades de Q_AUX
51:               (insere entidade no from e atributos no where);
52:               insere projeção adicional na cláusula select de Q_AUX:  ‘ , ’  || < rank de Pi >  || ‘ as GIC’
53:          SENÃO                                                                         /* --- Junção indireta (n x m)
54:               SE há relacionamento indireto (tabela de relacionamento) entre E e entidade de Q_AUX
55:                    realiza junção em Q_AUX entre <E> e entidades de Q_AUX que possuam relacionamento indireto
56:                    (insere entidades na cláusula from e condições de junção na cláusula where);
57:                    insere projeção adicional na cláusula select de Q_AUX:  ‘ , ’  || < rank de Pi * 0,9 >  || ‘ as GIC’
58:               FIM SE;
59:          FIM SE;
60:    FIM SE;
61:                                                        /* --- monta restrição da preferência Pi
62:    SE <oc> de Pi = IN ou NOT IN
63:          Q_AUX = Q_AUX || ‘ AND ’|| <E.a> de Pi || ‘ = ’ || <valor de Pi>
64:     SENÃO
65:          Q_AUX = Q_AUX || ‘ AND ’|| <E.a> de Pi || <operador de Pi> || <valor de Pi>
66:    FIM SE;
67:
68:     SE i < > n                                          /* --- monta união de consultas internas
69:          Q’:= Q’ || Q_AUX  || ‘ UNION  ALL’;
70:     SENÃO
71:          Q’:= Q’ || Q_AUX  || ‘) GROUP BY ’ || <lista de atributos selecionados em Q>
72:                                                  || ‘ HAVING COUNT(*)  > ’ ||  kp  ||
73:                                                  || ‘ ORDER BY  AVG (GIC) DESC, COUNT(*)  DESC ’
74:     FIM SE;
75:
76: FIM PARA;
77:
78: Retorne (Q’);
```

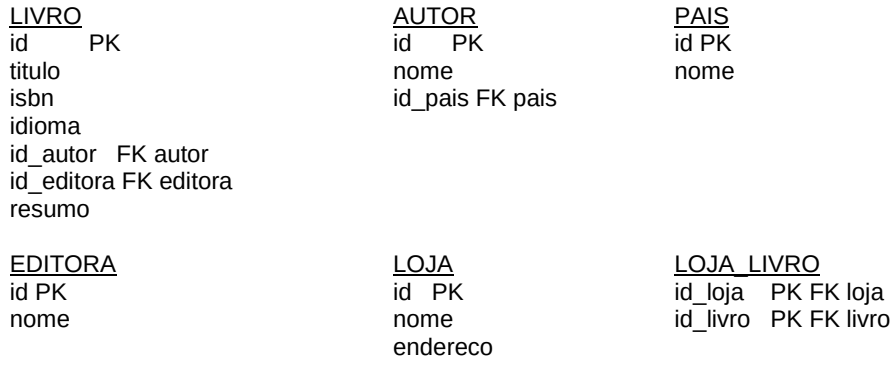
Fonte: O autor.

APÊNDICE B – Exemplos Adicionais de Aplicação de DRC

Exemplos da Diretiva Project

Como exemplos de utilização para adaptação ao dispositivo e à aplicação, considere o esquema relacional da Figura B.1 e o comando de consulta Q, para os exemplos a seguir e suas respectivas regras:

Figura B.1 – Esquema relacional de exemplo - livraria



Fonte: O autor.

I. Considere que a consulta Q foi realizada por meio de um smartphone que se utilizou de uma aplicação de sugestão de livros, o que ocasionou a execução das regras R1 e R2. A regra R1 condiciona ao uso de dispositivo do tipo smartphone a não exibição do atributo resumo da entidade livro. Já a regra R2 acrescenta informação sobre nome da editora e do autor para a aplicação “sugere_livro”.

```
Q:      SELECT titulo, idioma, resumo
          FROM livro;

R1:     SE dispositivo.tipo = 'smartphone´
          ENTÃO PROJECT (NOT livro.resumo);

R2:     SE aplicacao.funcao = 'sugere_livro'
          ENTÃO PROJECT (editora.nome, autor.nome)
```

A consulta inicial Q será reescrita da seguinte forma, após aplicação da regra R1:

```
Q':     SELECT titulo, idioma
          FROM livro;
```

Após a aplicação da regra R2 apresentará a forma:

```
Q':  SELECT  l.titulo, l.idioma, e.nome, a.nome
       FROM  livro l, editora e, autor a
      WHERE  l.id_editora = e.id
        AND  l.id_autor = a.id;
```

II. Considere agora que a mesma consulta do exemplo anterior foi realizada por uma função da aplicação que informa onde comprar livros. A regra R3 acrescenta informação sobre o nome da editora e do autor do livro, para a aplicação "onde comprar".

```
Q:       SELECT titulo, idioma
           FROM livro;

R3:      SE aplicacao.funcao = 'onde comprar'
            ENTÃO PROJECT (loja.nome, loja.endereco);
```

A consulta inicial Q será reescrita da seguinte forma após a aplicação da regra R3:

```
Q':      SELECT l.titulo, l.idioma, l.resumo, lj.nome, lj.endereco
           FROM livro l, loja_livro ll, loja lj
          WHERE ll.id_loja = lj.id
            AND ll.id_livro = l.id;
```

Exemplos da Diretiva Filter

III. Exemplo de utilização como preferências condicionantes de consulta: Para os exemplos a seguir e suas respectivas regras, considere o esquema relacional da Figura B.2 e comando de consulta Q a seguir:

Figura B.2 – Esquema relacional de exemplo - obras de arte

OBRA		ARTISTA		PAIS		ESTILO		ARTISTA_ESTILO	
id	PK	id	PK	id	PK	id	PK	id_artista	PK FK artista
titulo		nome		nome		nome		id_estilo	PK FK estilo
ano		id_pais	FK pais						
id_artista	FK artista	tipo							
descrição									

Fonte: O autor.

```
Q:       SELECT o.titulo, o.ano, a.nome
           FROM obra o, artista a
          WHERE o.id_artista = a.id;
```

Considere também que a consulta foi realizada por um usuário que deseja apenas informações sobre pintores e escultores holandeses. As regras seguintes foram configuradas baseadas nas preferências desse usuário.

Parâmetros Instanciados como foco de interesse da consulta:

- Q.obra = true;

Parâmetro externo Instanciado:

- user.id = 517;

Regra utilizada:

```
R4:       SE user.id = 517  ENTÃO
          SE Q.obra  ENTÃO
          FILTER ((artista.tipo = 'pintor'
                OR   artista.tipo = 'escultor')
               AND  (artista.id_pais = (pais.nome = 'holanda'))));
```

A regra é condicionada ao usuário 517 e qualquer atributo da entidade *obra* como foco de interesse da consulta.

Após aplicação da regra R4, a consulta será reescrita pelo algoritmo FILTER apresentado, da seguinte forma:

```
Q':      SELECT o.titulo, o.ano, a.nome
           FROM obra o, artista a, pais p
          WHERE o.id_artista = a.id
            AND (a.tipo = 'pintor' OR a.tipo = 'escultor')
            AND a.id_pais = p.id
            AND p.nome = 'holanda';
```

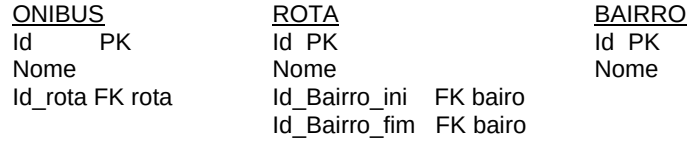
Considere adicionar a restrição do usuário apenas para artistas impressionistas. A regra R5 a seguir é possível ser atendida via junção com tabela de relacionamento (artista_estilo) e o resultado do processamento do algoritmo FILTER resulta na consulta Q' a seguir:

```
R5: SE user.id = 517
       SE Q.obra
       ENTÃO FILTER (artista.tipo = 'pintor' AND
                     (artista.id_pais = (pais.nome = 'holanda')));
             FILTER (estilo.nome = 'impressionismo');

Q':      SELECT o.titulo, o.ano, a.nome
           FROM obra o, artista a, pais p, estilo e, artista_estilo ae
          WHERE o.id_artista = a.id
            AND a.tipo = 'pintor'
            AND a.id_pais = p.id
            AND p.nome = 'holanda'
            AND ae.id_estilo = e.id
            AND ae.id_artista = a.id
            AND e.nome = 'impressionismo';
```

IV. Exemplo de utilização de restrição com junção explícita. Considere um BD que tenha as entidades ROTA e BAIRRO, conforme esquema da Figura B.3, a DRC da regra R6 será considerada inválida, já que há duas referências da entidade ROTA para a entidade BAIRRO. A regra deve ser codificada conforme o formato da regra R7, que explicita o atributo de junção a ser usado e não ocasionará erro na rotina de junção do algoritmo *Filter_Sentence* (vide Apêndice A).

Figura B.3 – Esquema relacional de exemplo - rotas de ônibus



Fonte: O autor.

```
R6:     SE Q.rota ENTÃO
              FILTER (bairro.nome = 'Arruda');
        FIM SE;


R7:     SE Q.rota ENTÃO
              FILTER (rota.id_bairro_fim =(bairro.nome = 'Arruda'));
        FIM SE;
```

Exemplos da Diretiva Order

V. Considere que consultas sobre a entidade automóvel têm regra associada que define a ordenação ascendente para o atributo fabricante e descendente para o atributo velocidade máxima (velmax):

```
Q: SELECT * FROM automovel;

R8: SE Q.automovel ENTÃO
       ORDER (automovel.fabricante ASC, automovel.velmax DESC);

Q':  SELECT *
       FROM automovel
       ORDER BY fabricante ASC, velmax DESC;
```

VI. Para a mesma consulta Q do exemplo anterior, temos agora a regra R9, que especifica uma opção de limitação para os cinco automóveis mais velozes:

```
R9: SE Q.automovel ENTÃO
      ORDER (automovel.velmax DESC LIMIT DENSE_RANK 5);
```

```
Q':  SELECT *
       FROM (SELECT *, DENSE_RANK() OVER (ORDER BY velmax DESC)
                                                        as denserank
             FROM automovel) as Q
      WHERE denserank  <=5;
```

VII. Considere a regra R10 que dita que para consultas ao atributo salário, da entidade funcionário, devem ser exibidas as informações do funcionário cujo nome seja igual ao do usuário corrente antes de outros e como segundo critério de ordenação os salários de forma decrescente.

```
Q:      SELECT * FROM funcionario;

R10:    SE Q.funcionario.salario ENTÃO
         ORDER
         ((funcionario.nome CASE (&USER.NOME, 1), (ELSE, 2)),
          (funcionário.salario DESC)));

Q':     SELECT * FROM funcionario
        ORDER BY CASE when nome = &user.nome then 1 else 2
                                                  end,
                          salario DESC;
```

VIII. Neste exemplo, a regra R11 informa que para consultas do atributo salário, da entidade funcionário, deve-se exibir primeiro as informações dos funcionários cujo nome do cargo seja igual a 'presidente' e depois os outros cargos, de forma ordenada por nome. Como segundo critério de ordenação os salários maiores que 250 devem ser listados antes que outros, sendo que estes devem ser ordenados pelo próprio valor de forma ascendente.

```
Q:      SELECT * FROM funcionário;

R11:    SE Q.funcionario.salario ENTÃO
         ORDER ((cargo.nome CASE ('presidente', 'a'),
                     (ELSE, cargo.nome ASC)),
                (funcionário.salario >250, 1),
                     (ELSE, funcionário.salario)ASC));
```

Observação: Nos casos de utilização da cláusula CASE, o critério utilizado para indicação do argumento de ordenação <arg> na DRC deve considerar tipos de dados semelhantes. Neste exemplo `cargo.nome` tem tipo alfabético, logo foi utilizado 'a' para indicar a primeira posição de ordenação, ao contrário do numeral 1, utilizado para indicar a posição inicial para tipo numérico.

```
Q':   SELECT *
      FROM funcionario f, cargo c
      WHERE c.id = f.id_cargo
```

```
ORDER BY
CASE WHEN c.nome = 'presidente' THEN ‘a’
                                ELSE c.nome
                                END ASC,
CASE WHEN f.salario >= 250 THEN 1
                           ELSE f.salario END;
```

IX. A Regra R12 é um exemplo da aplicação simultânea das opções <atributo>, <valor_atributo> e LIMIT. Essa regra solicita a ordenação dos automóveis do tipo ‘Popular’ antes dos outros tipos, que devem estar ordenados pelo tipo e como segundo critério de ordenação a velocidade máxima de forma ascendente (*default*). A quantidade de tuplas retornadas deve ser limitada a 5, de acordo com a função DENSE_RANK.

```
Q:      SELECT * FROM automovel;

R12:    SE Q.automovel
        ENTÃO
         ORDER (((automovel.tipo =‘Popular’,‘a’),
                 (ELSE, automovel.tipo ASC)) ,
                 (automovel.velmax DESC) LIMIT DENSE_RANK 5);

Q’:  SELECT * FROM
     (SELECT *, RANK() OVER (ORDER BY CASE WHEN tipo = 'Popular'
                                        THEN 'a' ELSE tipo END,
                                   consumo DESC) as denserank
        FROM automovel) as Q
       WHERE denserank <=5;
```

Exemplos da Diretiva Group

X. Suponha que uma rotina de contabilização mensal necessite de um agrupamento que liste os livros e respectivas editoras que venderam mais que 100 exemplares no mês corrente.

```
Q: SELECT l.nome, l.nome_editora FROM livro;

R14:    SE aplication.name = ‘Contabilidade Mensal’  AND
            aplication.function = ‘Destaques de venda’ ENTÃO
                 FILTER((vendas.mes = &time.mes)
                         AND (vendas.ano = &time.ano));
                   GROUP (vendas.ano, vendas.mes,
                          livro.nome, editora.nome, COUNT(*) > 100,
                                                    Qtd. Vendida);

Q`:     SELECT v.ano, v.mes, l.nome, e.nome, count(*) as ‘Qtd. vendida’
         FROM livro l, vendas v, editora e
        WHERE l.livro_id = v.livro_id
          AND e.id = l.editora_id
          AND v.mes = &time.mes
```

```
        AND v.ano = &time.ano
    GROUP BY v.ano, v.mes, l.nome, e.nome
      HAVING COUNT(*) > 100;
```

Observe que o agrupamento sobre atributo não presente em Q força a junção de entidades.

XI. Suponha uma rotina de contabilização mensal que necessite de um agrupamento que liste os livros com as cinco maiores vendagens. Para tanto foi utilizada a função DENSE_RANK, que permite empates de colocação.

```
Q:    SELECT l.name, v.valor
        FROM venda v, livro l
       WHERE v.livro_id = l.id;

R15: SE aplication.name = 'Contabilidade Mensal'  AND
           aplication.function = 'Top 5 Livros' ENTÃO
              FILTER(vendas.mes = &time.mes AND vendas.ano = &time.ano);
               GROUP(livro.nome, SUM(vendas.valor) LIMIT RANK 5);

Q`:  SELECT * FROM
     (SELECT l.nome,
       DENSE_RANK (SUM(v.valor)) OVER (PARTITION BY l.nome) AS denserank
        FROM livro l, vendas v
       WHERE v.livro_id = l.id
         AND v.mes = &time.mes
        AND v.ano = &time.ano)
     WHERE denserank <= 5;
```

Exemplo da Diretiva Prefer

XII. Suponha um usuário cujas preferências indicam que ele não gosta de pratos que contenham carne e aceita pratos que contenham massa. Considere o esquema relacional de um cardápio hipotético mostrado na Figura B.4, o perfil exibido a seguir, a regra de produção R16 e a consulta Q, de exemplo:

Figura B.4 – Esquema relacional de exemplo - cardápio

PRATO		PRATO_INGREDIENTE		INGREDIENTE	
id	PK	Prato_id	PK FK	Ingrediente_id	PK
nome		Ingrediente_id	PK FK	nome	

Fonte: O autor.

Perfil

```
_____________________________________________
<Profile>
Type=user;
User_id=0102;

<Preferences>
P1: ingrediente.nome = 'massa' (0,3));
P2: ingrediente.nome = 'carne' (-1,0));
_____________________________________________

R16:    SE Q.prato ENTÃO
           PREFER (&USER.id);

Q:      SELECT id_prato, nome_prato FROM prato;
```

Considere os dados de amostra, listados na Figura B.5 e a consulta reescrita Q':

Figura B.5 – Dados de amostra - cardápio

PRATO	PRATO_INGREDIENTE	INGREDIENTE
1 lasanha verde	1; 1;	1; queijo
2 lasanha a bolonhesa	1; 2;	2; massa
3 risoto de frango	2; 2;	3; carne
4 espaguete ao alho e óleo	2; 3;	4; alho
5 espaguete a bolonhesa	3; 7;	5; azeite
6 pizza aos 4 queijos	3; 8;	7; frango
	4; 4;	8; arroz
	5; 3;	6; tomate

Fonte: O autor.

```
Q':     SELECT id_prato, nome, AVG(GIC)
          FROM (
              SELECT p.id_prato, p.nome, 0.27 as GIC
                FROM prato p
                WHERE EXISTS ( SELECT *
                   FROM ingrediente i1, prato p1,
                        prato_ingrediente pi1
                  WHERE pi1.id_prato = p.id_prato
                    AND i1.id_ingrediente = pi1.id_ingrediente
                    AND i1.nome = 'massa' )
           UNION
               SELECT p.id_prato, p.nome, 0.9 as GIC
                 FROM prato p
                 WHERE NOT EXISTS( SELECT *
                      FROM ingrediente i1, prato p1,
                           prato_ingrediente pi1
                     WHERE pi1.id_prato = p.id_prato
                       AND i1.id_ingrediente = pi1.id_ingrediente
                       AND i1.nome = 'carne' )
          ) as q
            GROUP BY id_prato, nome
            HAVING count(*) >=1
            ORDER BY AVG(GIC) DESC, COUNT(*) DESC
```

Neste exemplo, consideramos a quantidade mínima de preferências a serem atendidas por uma tupla (expressa por *kp* e definida pelo ED) como maior ou igual a 1. A execução da consulta resultará nas seguintes tuplas:

Id_prato	Nome	AVG(GIC)
3	risoto de frango	0.90
4	espaguete ao alho e óleo	0.58
1	lasanha verde	0.58
6	pizza aos 4 queijos	0.58
2	lasanha a bolonhesa	0.27
5	espaguete a bolonhesa	0.27