PAULO HENRIQUE DOS SANTOS

UMA ARQUITETURA PARA MONITORAMENTO DE BANCO DE DADOS E RECOMENDAÇÕES UTILIZANDO SISTEMA DE BANCO DE DADOS ATIVOS

Dissertação submetida ao Programa de Pós-Graduação em Computação Aplicada da Universidade Tecnológica Federal do Paraná como requisito parcial para a obtenção do título de Mestre em Computação Aplicada.

Curitiba PR

Agosto de 2014

PAULO HENRIQUE DOS SANTOS

UMA ARQUITETURA PARA MONITORAMENTO DE BANCO DE DADOS E RECOMENDAÇÕES UTILIZANDO SISTEMA DE BANCO DE DADOS ATIVOS

Dissertação submetida ao Programa de Pós-Graduação em Computação Aplicada da Universidade Tecnológica Federal do Paraná como requisito parcial para a obtenção do título de Mestre em Computação Aplicada.

Área de concentração: *Engenharia de Sistemas Computacionais.*

Orientador: Prof. Dr. Roberto C. Betini.
Co-orientadora: Profa. Dra. Nádia P. Kozievitch.

Curitiba PR

Agosto de 2014

Dados Internacionais de Catalogação na Publicação

S237a Santos, Paulo Henrique dos
Uma arquitetura para monitoramento de banco de dados e recomendações utilizando sistema de banco de dados ativos / Paulo Henrique dos Santos. – 2014.
67 f.: il.; 30 cm

Orientador: Roberto Cesar Betini.
Coorientadora: Nádia Puchalski Kozievitch.
Dissertação (Mestrado) - Universidade Tecnológica Federal do Paraná. Programa de Pós-Graduação em Computação Aplicada, Curitiba, 2014.
Bibliografia: f. 57-60.

1. Banco de dados - Gerência. 2. Banco de dados orientado a objetos. 3. Sistemas de recomendação (Filtragem da informação). 4. Algoritmos. 5. Protótipos - Testes. 6. Computação - Dissertações. I. Betini, Roberto Cesar, orient. II. Kozievitch, Nádia Puchalski, coorient. III. Universidade Tecnológica Federal do Paraná - Programa de Pós-graduação em Computação Aplicada. IV. Título.

CDD 22 – 621.39

Biblioteca Central da UTFPR, Campus Curitiba

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Curitiba
Diretoria de Graduação e Educação Profissional
Departamento Acadêmico de Informática – DAINF
Programa de Pós-Graduação em Computação Aplicada - PPGCA
Mestrado Profissional

ATA DA DEFESA DE DISSERTAÇÃO DE MESTRADO 19

DISSERTAÇÃO PARA OBTENÇÃO DO GRAU DE MESTRE EM **COMPUTAÇÃO APLICADA**

No dia 29 de agosto de 2014, às 09:00 horas, reuniu-se na Sala B-204 - bloco B - 2º andar do Câmpus Curitiba, a banca examinadora composta pelos professores doutores:

Prof. Roberto Cesar Betini, Dr. (Presidente)	UTFPR - CT
Prof. João Alberto Fabro, Dr.	UTFPR - CT
Prof. Antonio Morais da Silveira, Dr.	UFPA
Prof. Nadia P. Kozievitch, Dr.	UTFPR - CT

sob Presidência de **Roberto Cesar Betini** para examinar a dissertação do candidato **PAULO HENRIQUE DOS SANTOS**, intitulada**:** "Uma Arquitetura para Monitoramento de Banco de Dados e Recomendações Utilizando Sistema de Banco de Dados Ativos". Após a apresentação, o candidato foi arguido pelos examinadores e foi dada a palavra aos presentes para formularem perguntas ao candidato. Os examinadores reunidos deliberaram pela _____ da dissertação.

O candidato foi informado que a concessão do referido grau, na área de concentração Engenharia de Sistemas Computacionais, está condicionada à (i) satisfação dos requisitos solicitados pela Banca Examinadora e lavrados na documentação entregue ao Candidato; (ii) entrega da dissertação em conformidade com as normas exigidas pela UTFPR; e (iii) entrega da documentação necessária para elaboração do Diploma. A Banca Examinadora determina um **prazo de _____ dias** para o cumprimento dos requisitos (desconsiderar esse parágrafo caso a dissertação seja reprovada).

_____

_____

_____

Nada mais havendo a tratar, a sessão foi encerrada às _____, dela sendo lavrada a presente ata que segue assinada pela Banca Examinadora e pelo Candidato.

_____
Prof. **Roberto Cesar Betini, Dr.**
presidente - (UTFPR - CT)

_____
Prof. **João Alberto Fabro, Dr.**
(UTFPR - CT)

_____
Prof. **Antonio Morais da Silveira Dr.**
(UFPA)

_____
Prof. **Nadia P. Kozievitch**
(UTFPR - CT)

Candidato: _____

DECLARAÇÃO PARA A OBTENÇÃO DO GRAU DE MESTRE

A coordenação do Programa declara que foram cumpridos todos os requisitos exigidos pelo Programa de Pós-Graduação para a obtenção do grau de mestre.

Curitiba, ____de _____de 20____.

"A Ata de Defesa original está arquivada na Secretaria do PPGCA".

Este trabalho é dedicado às mulheres da minha vida, dona Adair, minha mãe, e à linda Vanessa, minha esposa.

AGRADECIMENTOS

Agradeço primeiramente a Deus por me acompanhar todos os dias da minha vida com sua infinita misericórdia.

Agradeço a minha esposa que por diversas vezes me incentivou, ajudou e apoiou nessa difícil caminhada. Minha mãe, que é minha heroína e um exemplo de dedicação e responsabilidade.

Agradeço ao Prof. Dr. Roberto Cesar Betini, meu orientador, pela dedicação, paciência e pelas muitas horas de orientação, a minha co-orientadora, Profa. Dra. Nádia Puchalski Kozievitch, por acreditar no meu trabalho e me ajudar a entender muitas coisas que eram novas e complexas pra mim.

Agradeço aos professores Dr. Adolfo Gustavo Serra Seca Neto, Dr. João Alberto Fabro, Dr. Gustavo Alberto Giménez Lugo, por me ajudarem por diversas vezes com suas ideias, correções e sugestões.

Por fim, mas não menos importante, agradeço a Ouro Verde Locação e Serviços SA pelo apoio, ao meu gestor Alexsandro Ramires pelo apoio e compreensão, e aos amigos Leonardo Sippel, Filipe Daros e Ricardo Doi pela ajuda nas diversas vezes que precisei.

RESUMO

O monitoramento de forma integrada pode se tornar complexo em ambientes com bancos de dados heterogêneos, devido às particularidades na sintaxe e em ferramentas disponíveis. Em particular, bancos de dados ativos permitem o desenvolvimento de mecanismos e a automação de processos que envolvam os dados ou objetos. Este trabalho propõe o desenvolvimento de uma arquitetura para identificar e monitorar eventos DDL *(Data Definition Language)*, utilizando a abordagem de banco de dados ativos e recomendação. As aplicações desta arquitetura variam desde o simples monitoramento de eventos DDL em um ou mais bancos de dados, até a recomendação de possíveis configurações ou mudanças que podem ser realizadas no banco de dados monitorado. Neste contexto, essa dissertação propõe: (i) uma arquitetura integrada de banco de dados ativos e recomendação; (ii) a adaptação de um algoritmo de recomendação; e (iii) a validação dos conceitos aplicados através de um protótipo.

Palavras chaves: Sistemas de Banco de Dados Ativos, Sistema Gerenciador de Banco de Dados, Regras Ativas, Sistemas de Recomendação, Monitoramento.

ABSTRACT

Integrated environments monitoring can become complex with heterogeneous databases, due to the particularities in the language syntax and available tools. In particular, active databases allow developing mechanisms and automation of processes involving data or objects. This work proposes the development of an architecture to identify and monitor DDL (Data Definition Language), exploring the active databases and recommendations approach. This architecture could be explored in several ways: simple monitoring of DDL events in one or more databases, or recommendation of future DDLs or settings within the monitored database. In this context, this work proposes: (i) an integrated architecture of active databases and recommendation; (ii) the adaptation of a recommendation algorithm; and (iii) the validation of concepts through a prototype.

Key words: *Active Database, Database Management System, Active Rules, Recommendation Systems, Monitoring.*

SUMÁRIO

Lista de Figuras

Lista de Tabelas

Lista de Abreviações

SBDA	Sistema de Banco de Dados Ativos
SGBD	Sistema Gerenciador de Banco de Dados
ECA	Evento Condição Ação
DDL	*Data Definition Language*
DML	*Data Manipulation Language*
IA	Inteligência Artificial
CRM	*Customer Relationship Management*
CCPN	*Conditional Colored Petri Net*
SR	Sistema de Recomendações
ERP	*Enterprise Resource Plannin*

1 INTRODUÇÃO

Os Sistemas Gerenciadores de Bancos de Dados (SGBD) são poderosos softwares que controlam de forma muito eficiente os diversos bancos de dados em seus domínios (ROB e CORONEL, 2011). Estes sistemas, atualmente, compõem uma parte essencial de toda a empresa, não só para armazenar tipos de informações que são comuns à maioria, como também a informação que é específica à categoria de cada empresa, e por esta razão são amplamente utilizados por diversos sistemas (KORTH et al., 2012).

Muitas pesquisas e trabalhos práticos relacionados à SGBDs têm sido desenvolvidos, em especial, o monitoramento das mudanças ocorridas nas informações armazenadas nos bancos de dados. O desenvolvimento de mecanismos para automação de processos e manipulação destas informações também é bastante explorado.

Entretanto, o monitoramento das alterações que ocorrem na estrutura (objetos) dos bancos de dados, é um tanto negligenciado devido à complexidade na integração e no tratamento da sintaxe dos diversos SGBDs disponíveis (JUN, 2010).

Segundo Oliveira (2013), tanto a área pública como a privada, passaram a perceber o valor das informações que têm a sua disposição, e a considerá-las como um bem importante para o aumento da produtividade, eficiência e competitividade (OLIVEIRA, 2013).

Com essa percepção, também se identifica a necessidade do monitoramento e registro de forma automática das mudanças ocorridas no banco de dados. Por exemplo, a identificação de eventos disparados de forma interna ou externamente para alteração de objetos pertencentes à estrutura do banco de dados. Estes eventos podem ser monitorados e registrados para diversos fins, como auditoria, restrições, acionamento de outros mecanismos ou eventos, etc.

Para desenvolver mecanismos que possibilitem o monitoramento de forma dinâmica de tais alterações, é possível utilizar um Sistema de Banco de Dados Ativos (SBDA), que é a combinação da tecnologia de SGBD com técnicas de programação baseada em regras e orientada a eventos (HAMID, 2012).

Este tipo de sistema suporta a definição de regras, monitoramento e execução de ações em resposta às regras definidas, ou seja, suportam a execução de ações automáticas diante de eventos ocorridos no banco de dados.

Para o monitoramento de eventos referentes às alterações de objetos do banco de dados, ou seja, eventos DDL[1] (*Data Definition Language)*, o nível de complexidade aumenta, isso porque os SGBDs possuem particularidades específicas em sua sintaxe no desenvolvimento de funcionalidades para identificação destes eventos.

Por exemplo, para o desenvolvimento de um procedimento armazenado ou um gatilho em um SGBD (A) e em outro SGBD (B), será necessário implementar estes objetos de acordo com a sintaxe de cada um, isso torna complexa a padronização e generalização dos códigos para trabalhar com diversos bancos de dados.

Entretanto, é possível fazer uso de semântica para diminuir a complexidade na manipulação das regras ativas aplicadas nos bancos de dados. Por exemplo, na criação de uma tabela no SGBD *Microsoft SQL Server* não é preciso definir um *tablespace*, já para a criação de uma tabela no SGBD *Oracle* é sugerida a definição de um *tablespace*[2]. Logo, é preciso trabalhar com semânticas diferentes entre estes SGBDs.

Outro fator importante e pouco explorado nos diversos trabalhos analisados é a integração de SBDA com Sistemas de Recomendação (SR). Essa integração pode auxiliar o SBDA de diversas formas, pois a partir dos registros das alterações ocorridas no banco de dados, ou seja, com base nas informações referentes aos eventos DDL aplicados no banco de dados monitorado, o SR pode sugerir possíveis inclusões de objetos no banco de dados, neste caso, a criação de tabelas por exemplo.

Na arquitetura proposta neste trabalho, é considerado o desenvolvimento e a integração de mecanismos que permitam o armazenamento das regras ativas, bem como a transformação e aplicação destas regras, criando gatilhos *(triggers)* que farão o monitoramento e o registro dos eventos DDL aplicados no banco de dados. Também fazemos uso de semântica para definir como as regras ativas podem ser executadas pelos gatilhos *(triggers)* de monitoramento dos eventos DDL.

Assim, as informações registradas por estes mecanismos (gatilhos) podem ser utilizadas para diversos fins, como na geração de relatórios gerenciais, na auditoria dos objetos alterados e na recomendação da criação de possíveis tabelas em determinado banco de dados.

---

[1] Instruções *DDL – (Data Definition Language)* definem as estruturas de dados, como por exemplo, as tabelas, que compõem um banco de dados. Existem cinco tipos básicos de instruções DDL: *CREATE, ALTER, DROP, RENAME e TRUNCATE*. PRICE, J. SQL Domine SQL e PL/DQL no banco de dados Oracle, c. 1, p. 31. São Paulo: ARTMED Editora S.A. 2009.

[2] *Tablespace* é uma unidade lógica de armazenamento onde são agrupados os arquivos de dados *(datafiles)* como tabelas, índices, etc. em um banco de dados Oracle. BRYLA, B; LONEY, K. Oracle Database 11g Manual do DBA, c. 1, p. 30. São Paulo: BOOKMAN Editora S.A. 2007.

Este trabalho envolve a pesquisa de sistemas de banco de dados ativos, sistemas gerenciadores de bancos de dados e sistemas de recomendação. Propondo através do desenvolvimento com base nesta pesquisa, mecanismos que permitem a integração e o funcionamento adequado dos processos e funcionalidades contidas na arquitetura proposta.

1.1 MOTIVAÇÃO

Identificar eventos DDL disparados contra um banco de dados é o primeiro passo para encontrar informações que possam contribuir na análise das alterações ocorridas. Os mecanismos responsáveis por identificar esse tipo de evento precisam ter seu nível de codificação generalizada, para que possam atender diferentes SGBDs.

Neste trabalho a utilização de SBDA se dá justamente para atender esta necessidade. Através do monitoramento por regras ativas é possível registrar as ocorrências de eventos DDL, e a partir destes registros, um SR pode recomendar possíveis mudanças ou configurações para o banco de dados monitorado dentro de um ambiente com diferentes SGBDs.

O foco deste trabalho está na integração de SBDA com SR, para que a partir dos registros de eventos DDL (monitorados por regras ativas) ocorridos nos bancos de dados monitorados, sejam feitas recomendações (através de um sistema de recomendação) de possíveis mudanças ou configurações nos objetos afetados por estes eventos.

O fato de grande parte dos trabalhos relacionados à SBDA não explorarem o monitoramento das alterações ocorridas por eventos DDL, e grande parte dos trabalhos relacionados à SRs explorarem somente sistemas web focando em produtos e clientes, já é uma grande motivação para o desenvolvimento deste trabalho.

Como os eventos DDL estão diretamente ligados à estrutura dos bancos de dados, logo, as técnicas de SRs podem ser aplicadas neste contexto, para auxiliar usuários desse sistema a ter um controle mais efetivo sobre esse tipo de evento.

1.2 JUSTIFICATIVA

O fato de poucos trabalhos relacionados a Sistemas de Recomendação, explorarem a integração com Sistemas de Banco de Dados Ativos, foi um ponto importante para a concepção deste trabalho.

A integração de SR com SDBA permite a idealização e desenvolvimento de mecanismos, que possibilitam o monitoramento de alterações realizadas em bancos de dados. A partir do histórico destas alterações, algoritmos de recomendação podem ser aplicados ou adaptados para gerarem diversas recomendações sobre as alterações ocorridas nos bancos de dados.

1.3 OBJETIVOS

1.3.1 Objetivo Geral

O objetivo deste trabalho é propor e validar uma arquitetura com a integração de mecanismos, que proporcionem o monitoramento das alterações ocorridas por eventos DDL no banco de dados monitorado. E que por meio de um SR, faça recomendações da criação de possíveis tabelas por meio da análise do histórico dos eventos DDL *(CREATE TABLE)* ocorridos neste banco de dados.

1.3.2 Objetivos Específicos

Para atingir o objetivo geral, são considerados os seguintes objetivos específicos:

- Desenvolver uma arquitetura integrada de banco de dados ativos e recomendação.
- Adaptar um algoritmo de recomendação, para recomendar a criação de possíveis tabelas em bancos de dados monitorados.
- Validar os conceitos apresentados neste trabalho por meio de um protótipo.

1.4 ESTRUTURA DA DISSERTAÇÃO

Além desta introdução, o capítulo 2 apresenta a revisão bibliográfica na qual são descritos os trabalhos relevantes para este trabalho. O capítulo 3 apresenta a metodologia do trabalho. O capítulo 4 apresenta a arquitetura proposta neste trabalho. O capítulo 5 apresenta a implementação da arquitetura e os experimentos realizados para validação da arquitetura proposta. E por fim, no capítulo 6, são apresentadas a conclusão e sugestões para trabalhos futuros.

2 REVISÃO BIBLIOGRÁFICA

2.1 SISTEMA DE BANCO DE DADOS ATIVOS

O SBDA é uma tecnologia que integra as funcionalidades convencionais do SGBD aos mecanismos que permitem a especificação, análise, execução e monitoramento de regras que asseguram um modelo comportamental, denominadas regras ativas (JOSKO, 2011).

O SBDA fornece um paradigma para pesquisas de desenvolvimento a partir de uma combinação das tecnologias de SGBDs e Inteligência Artificial (MORGENSTERN, 1983).

Esse tipo de sistema amplia a capacidade do SGBD, tornando possível a resposta automática a eventos disparados contra o banco de dados de forma interna ou externa. Esses eventos são identificados e de acordo com uma condição é executada uma ação em resposta (HAMID, 2012).

Várias aplicações como automação em fábricas e complexos sistemas financeiros (por exemplo, bolsa de valores), precisam ter monitoramento automatizado, de forma que permitam ao SGBD realizar a operação adequada automaticamente. Assim, os SBDAs têm ajudado de forma prática nos diversos cenários de automação de funcionalidades em bancos de dados dentro das empresas (HAMID, 2012).

Segundo Telnarová (2012), muitas das atividades que são implementadas como aplicações para apoiar as políticas de gestão de dados podem utilizar SBDA.

Existem vários tipos de regras que desempenham um papel importante em SGBD, mas ao contrário dos SGBDs convencionais, onde o serviço é apenas a execução da consulta eletrônica, SBDAs oferecem um número de serviços avançados, como por exemplo, o raciocínio dedutivo, responder consultas com base em regras dedutivas, e o processamento de entrada de eventos com base em regras ativas.

Ainda segundo Telnarová (2012), SGBDs convencionais não funcionam com regras dedutivas e sua função parcial é exercida por consultas, já em SBDA, as regras são conceitos utilizados para a obtenção de informações a partir do banco de dados de maneira mais eficaz.

Alguns exemplos de utilização para o SBDA são: controle da integridade de objetos ou dados, controle de acesso, segurança e atualizações de dados (CILIA, 1996). Esses sistemas monitoram eventos executados no banco de dados e, quando eles ocorrem, de acordo com uma condição é retornada uma resposta automaticamente caracterizada por uma mensagem, ou até mesmo, a execução de uma sequência de comandos. O comportamento desejado é

expresso em regras de produção, que são definidas e armazenadas no banco de dados (RABUZIN, 2011).

Devido ao fato dos SGBDs convencionais possuírem atuação passiva em seu funcionamento, ou seja, as operações como: consultas, atualizações e inserções serem realizadas apenas quando solicitadas por uma aplicação ou por um usuário, os SBDAs podem ser considerados uma extensão dos SGBDs (CAMPIN, 1997).

Segundo Montesi (1995), o comportamento reativo do SBDA é baseado em um sistema de regras que integram uma causa a um efeito previsto. Estas regras, denominadas regras ativas, permitem o monitoramento e reação a eventos específicos e pertinentes.

Na estrutura de um sistema de banco de dados, causa e efeito são manipulações de dados. Tais regras permitem expressar, de forma natural, importantes conceitos de banco de dados como cenários, restrições e métodos. Uma regra ativa é uma generalização deste simples esquema de linguagem (CARDOSO, 2004).

2.2 REGRAS ATIVAS

Há muito tempo as regras que definem as ações que são disparadas automaticamente por determinados eventos, têm sido consideradas melhorias importantes para os SGBDs (ELMASRI, 2011).

O conceito de *triggers* utilizado para especificar alguns tipos de regras ativas já fazia parte das primeiras versões da especificação SQL para os bancos de dados relacionais. Agora os *triggers* já fazem parte do padrão SQL-99 e de outros da atualidade (WATSON, 2010).

Vários SGBDs relacionais como: *Oracle*[3], *DB2*[4] *e Microsoft SQL Server*[5]disponibilizam diversas modalidades de *triggers*. Entretanto, várias pesquisas têm sido feitas para identificar um modelo geral para banco de dados ativo (WATSON, 2010).

Segundo Zaniolo (1997), as aplicações de regras ativas apoiam os SGBDs convencionais em diversos pontos, como integridade relacional, manutenção dos dados, replicação, monitoramento das alterações ocorridas, etc. Estas são aplicações referentes aos eventos internos. Ainda segundo Zaniolo (1997), com relação à aplicação referente aos eventos externos, comumente chamados de regras de negócio, são executadas como parte do serviço da computação que normalmente está contido no código da aplicação. Assim, as

[3]*http://www.oracle.com/br/products/database/overview/index.html*
[4]*http://www-03.ibm.com/software/products/en/db2advaenteservedit*
[5]*http://technet.microsoft.com/pt-br/sqlserver/*

regras ativas também são capazes de monitorar e reagir a cenários específicos relevantes para cada aplicação.

2.2.1 Modelos de Regras em Sistemas de Bancos de Dados Ativos

Um SBDA dispõe de um modelo de regras, constituído por um modelo de definição e por um modelo de execução de regras.

O modelo de definição estabelece o que deve ser especificado na regra (sintaxe da regra), ao passo que o modelo de execução define como se dará o processamento das regras.

O modelo de regras difere de um SGBD para outro, entretanto, os conceitos básicos dentro destes modelos estão presentes na linguagem SQL-99. Esses modelos (definição e execução) usados pela linguagem SQL são descritos a seguir.

2.2.2 Modelo de Definição de Regras em Sistemas de Bancos de Dados Ativos

O modelo de definição apóia essencialmente na descrição das funcionalidades de regras ativas admitidas pelo SGBD, e pode ser composto de três componentes principais: um evento, uma condição e uma ação. A Tabela 1 apresenta um exemplo da sintaxe da codificação deste modelo.

EVENTO	`CREATE TRIGGER [nome da regra]` `{BEFORE|AFTER|INSTEAD OF} {INSERT|DELETE|UPDATE [OF column(s)]} ON [tabela]`
CONDIÇÃO	`WHEN [expressão lógica];`

AÇÃO	`BEGIN`
	`  Bloco a ser executado caso condição for verdadeira;`
	`END;`

Tabela 1 – Tabela de componentes de regras ativas e sintaxe de codificação.

O modelo ECA define o princípio do funcionamento desta tecnologia, e tem sido muito utilizado para especificar regras de SBDA (ELMASRI, 2011). Os componentes deste modelo estão representados na Figura 1 e descritos em seguida:

Figura 1 – Esquema do Modelo de Regras ECA (JOSKO, 2011).

a) **Os eventos (E) que disparam a regra:** estes eventos normalmente são operações de atualização do banco de dados que são aplicadas explicitamente. Contudo, no modelo geral, eles também poderiam ser eventos temporais ou outros tipos de eventos externos.

b) **A condição (C) que determina se a ação da regra deve ser executada:** quando o evento que dispara a ação tiver ocorrido, uma opção condicional pode ser avaliada, e somente se for avaliada como verdadeira, a ação da regra é executada.

c) **A ação (A) a ser tomada:** a ação pode ser uma sequência de comandos SQL, uma transação do banco de dados, ou um programa externo executado automaticamente.

Estes eventos ainda podem ser de dois tipos: primitivos ou compostos. O evento simples ou primitivo acontece quando o evento tem uma única ocorrência, como por exemplo, uma inclusão de registro em uma tabela.

A combinação de eventos primitivos com operadores "*and*" ou "*or*" corresponde aos eventos compostos, que por sua vez podem conter outros eventos compostos (CHAKRAVARTHY, 1993). A Figura 2 ilustra esses eventos, que são descritos a seguir:

Eventos: são definidos como uma ocorrência atômica sobre o banco de dados e são produzidos por alguma fonte, por exemplo, um sistema externo ao SGBD, o próprio SGBD, ou por interação do próprio usuário com o banco de dados.

Eventos Primitivos: representam uma única ocorrência no banco de dados, por exemplo, um comando DDL *(ALTER TABLE)* para alteração de uma tabela.

Eventos Compostos: são derivados da combinação de eventos primitivos com operadores "*AND" ou" OR",* que por sua vez podem conter outros eventos compostos.

Eventos de BD: são operações de manipulação de dados como *insert*, *update* ou *delete*.

Eventos Explícitos: são os que fazem parte das regras de negócio que são implementadas na aplicação.

Eventos Temporais: podem ser definidos dentro de dois domínios, absoluto ou relativo. No domínio absoluto, o evento corresponde à definição de um momento específico, por exemplo, "07 de Maio de 2013 às 10 horas". No relativo, o evento temporal, define uma ocorrência em relação à outra, por exemplo, "30 minutos após a ocorrência do primeiro evento".

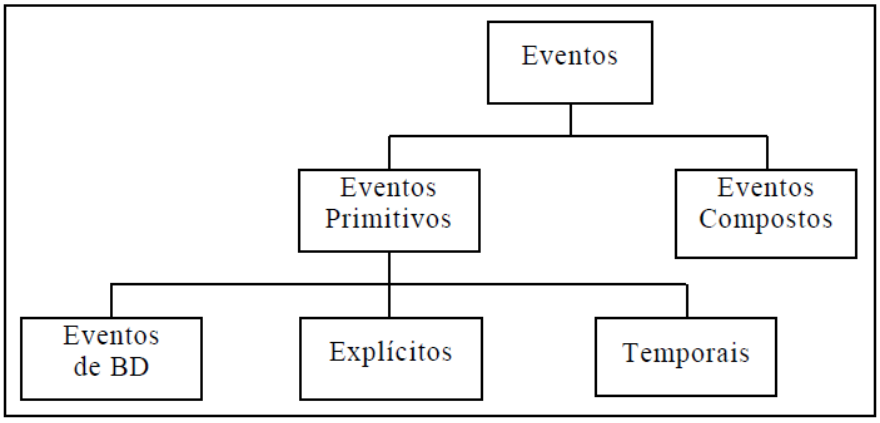

Figura 2 – Classificação de Eventos (CHAKRAVARTHY, 1993).

2.2.3 Modelo de Execução de Regras em Sistemas de Bancos de Dados Ativos

O modelo de execução determina como se dará o processamento de uma regra ou de um conjunto de regras, e também as propriedades da execução das regras, como granularidade de processamento, modo de junção, etc., (PATON e DIAZ, 1999).

A execução das regras é efetuada por meio de mecanismos que auxiliam no monitoramento do banco de dados e, com base nos eventos que ocorrem, os mecanismos verificam as condições definidas e executam as ações correspondentes (NORMAN, 1999).

Estas regras podem ser utilizadas de diversas formas, por exemplo, na necessidade de disparar uma ação, para registrar as alterações efetuadas por eventos DDL no banco de dados. Quando um evento qualquer de DDL *(drop, alter, create)* ocorrer, e alguma alteração for realizada no banco de dados, a condição de execução da regra será avaliada e, atendendo as condições definidas, a ação será executada registrando a ocorrência em um determinado local, que pode ser um arquivo de log ou um modelo de dados utilizado por uma aplicação para disponibilizar essas informações.

A Figura 3 ilustra as principais etapas realizadas no processamento de um conjunto de regras ativas. 1º) Na caixa “Evento Detectado” ocorre a detecção do evento; 2º) na caixa “Regras Disparadas” a regra é acionada de acordo com o evento detectado; 3º) na caixa “Regras Selecionadas” a condição é avaliada; e 4º) se a condição avaliada for verdadeira a regra é executada, a caixa “Regras Executadas” representa essa última etapa.

Nota-se também na Figura 3, que existe um laço para seleção de regras, isso ocorre quando existem mais regras a serem executadas dependendo do tipo de evento detectado.

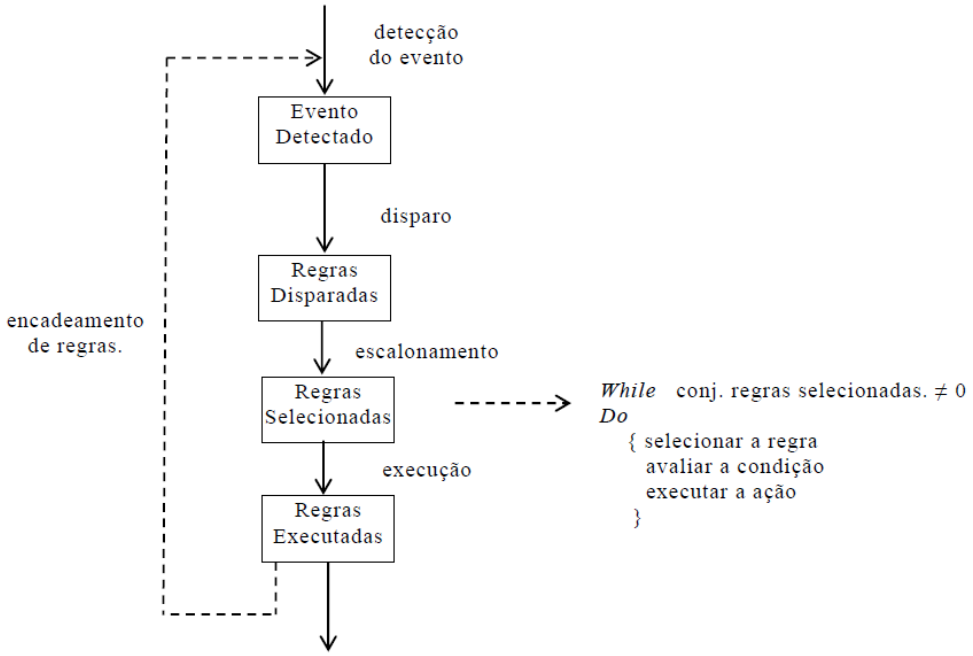

Figura 3 – Etapas do Tratamento de Eventos (PATON e DIAZ, 1999).

Em um SBDA podem existir diferentes abordagens para a especificação de uma arquitetura baseada em regras. A primeira abordagem é a construção de um SBDA a partir de sua concepção. A segunda é utilizar o SGBD passivo, para estender suas funcionalidades (PATON e DIAZ, 1999).

Os SGBDs convencionais são considerados passivos, porque executam operações de consulta e outras transações sobre o banco de dados somente a partir de solicitação explícita de um usuário ou de uma aplicação. Entretanto, os SBDAs estendem os SGBDs, incorporando a capacidade de executar automaticamente ações em resposta a eventos gerados de forma interna ou externa do banco de dados, o que é possível por meio da especificação de regras ativas (PATON e DIAZ, 1999).

2.3 SEMÂNTICA

Em SBDAs, parte da semântica que geralmente é codificada dentro das aplicações pode ser apresentada por meio das regras ativas, o que dá às aplicações uma nova dimensão de autonomia. Esse processamento reativo pode ser codificado uma só vez no banco de dados, por meio das regras ativas, e ficar automaticamente compartilhado entre todos os usuários e aplicações que acessam o banco de dados. A modificação desse processamento é feita por meio da mudança do conteúdo das regras, e não mais pela mudança das aplicações (ZANIOLO, 1997).

O processamento das regras é feito por “mecanismos de regras ativas”, e para tanto, podem ser aplicadas diferentes semânticas. Estes “mecanismos de regras ativas” que monitoram os eventos causados por transações do banco de dados, e executam as regras de acordo com determinadas políticas, permitem um comportamento reativo do banco de dados, que difere do comportamento passivo dos sistemas de banco de dados convencionais (ZANIOLO, 1997).

2.4 TRABALHOS CORRELATOS DE SISTEMA DE BANCO DE DADOS ATIVOS

Durante o levantamento bibliográfico foram analisados diversos trabalhos relacionados à SBDAs, com funcionalidades comuns e também distintas. A seguir são descritas as principais características destes trabalhos e ao final da seção, a comparação com o trabalho proposto.

Segundo Jun (2010), há duas linhas de pesquisa de Banco de Dados Ativos: (1) a tecnologia de banco de dados orientado a objetos, para alcançar um novo Sistema Gerenciador de Banco de Dados Ativos (essa linha de pesquisa concentra-se no estudo e desenvolvimento de modelos e métodos para banco de dados ativos) e (2) a tecnologia de banco de dados relacional, sistema já existente, onde o foco principal está no aperfeiçoamento dos mecanismos de funções ativas, disponíveis em SGBDs como *Oracle*[6]*, Microsoft SQL Server*[7]*, Sybase*[8] *e DB2*[9].

Além dessas duas linhas, o conceito de regras ativas está sendo introduzido em várias áreas relacionadas, tais como sistemas de banco de dados em tempo real, sistemas de banco de dados dinâmicos e sistemas de banco de dados dedutivo (JUN, 2010). Esta proposta apresenta um SBDA Web e sua aplicabilidade com sistemas CRM *(Customer Relationship Management)*. O autor propõe a implementação de “mecanismos ouvintes” que são responsáveis pelo monitoramento e execução de ações correspondentes às regras ativas relacionadas conforme o modelo ECA.

Diferente do modelo tradicional, esses mecanismos são implementados em um servidor de aplicação e não diretamente no banco de dados. Com isso, os mecanismos que monitoram

[6] *http://www.oracle.com/br/products/database/overview/index.html*
[7] *http://technet.microsoft.com/pt-br/sqlserver/*
[8] *http://www.sybase.com.br/products/databasemanagement/adaptiveserverenterprise*
[9] *http://www-03.ibm.com/software/products/en/db2advaenteservedit*

os eventos disparados sobre o banco de dados ficam independentes do SGBD, e a implementação destes mecanismos não fica limitada somente a um SGBD.

Os SBDAs por si só podem monitorar o estado interno do banco de dados, eventos disparados contra ele de forma interna ou externa, e agendar tarefas relacionadas sem atuação do usuário, com a função de satisfazer as exigências de tempo e consistência. Com base neste conceito, Huang (2009) propõe um algoritmo de detecção ativa de eventos em tempo de execução. Para tanto, ele utiliza o modelo ECA, e as funções das regras ativas são implementadas em uma aplicação a parte, juntamente com um algoritmo proposto. Este sistema funciona da seguinte forma, através de um mecanismo central é detectado o evento disparado por uma atividade de um usuário, e as informações referentes a este evento são coletadas e armazenadas para análise e auditoria (HUANG, 2009).

Apesar do trabalho apresentar mecanismos que permitam monitorar eventos em tempo de execução, fica limitada somente ao *SQLite*, que não é um SGBD completo, pois apesar de ser um mecanismo de armazenamento seguro, não possui controle de concorrência e não oferece integridade referencial[10].

Segundo Medina-Marín (2009), um evento que pode desencadear uma regra ECA pode ser de dois tipos: primitivo ou composto. Um evento primitivo é gerado pela execução de um *(insert, update, delete ou select)* no banco de dados, já um evento composto é formado por uma combinação de eventos primitivos e/ou compostos.

Os eventos compostos aumentam a complexidade de uma base de regras ativas (MEDINA-MARÍN, 2009). Para resolver este problema, Medina-Marín (2009) propõe o modelo CCPN *(Conditional Colored Petri Net)*, um mecanismo de análise automatizada de regras ativas em tempo de execução, que monitora os caminhos cíclicos de eventos primitivos e/ou compostos, e impede que esses eventos disparem a execução de regras em um laço infinito no banco de dados.

Apesar do modelo CCPN resolver o problema de execução redundante de regras ativas, ele fica centralizado no estudo da tecnologia, ou seja, o foco do estudo está na resolução de um problema do modelo ECA, relacionado à detecção de potenciais eventos de execução infinita, e não na aplicabilidade do modelo para resolução de problemas práticos encontrados nas empresas.

[10] *http://www.sqlite.org/limits.html*

Outra pesquisa relacionada ao modelo ECA é a de Jin (2009). E em seu trabalho ele propõe o algoritmo CIRS, extensão do seu algoritmo IRS desenvolvido em uma primeira pesquisa que permite o agendamento da execução de regras desencadeadas por eventos simples (JIN, 2007). O algoritmo CIRS permite que regras desencadeadas por eventos compostos, possam ser analisadas e agendadas para execução em um determinado momento com uma sequência específica, o que assegura a execução aderente à regras simultaneamente acionadas (JIN, 2009).

Assim como a pesquisa de Medina-Marín (2009), a de Jin (2009) se detém em criar uma extensão do modelo ECA, para agendamento da execução das regras ativas, sobre eventos compostos para um determinado tempo e sequência.

A pesquisa de Qiao (2007) também trabalha sobre modelo ECA e descreve um modelo de regras gráficas baseadas neste modelo tradicional, com um conjunto de novos eventos temporais para especificar restrições em tempo de execução. Segundo o autor, para suportar um nível maior de inteligência em tempo real, os SBDAs devem ter grande capacidade de resolver problemas complexos, especialmente os problemas com propriedades temporais. Em seu trabalho, é apresentado um conjunto com novos eventos temporais para interagir com regras compostas, ou seja, essas regras atuam somente na resolução de problemas que necessitam da implementação de regras com a definição de períodos para sua execução (QIAO, 2007). Assim como os outros trabalhos analisados, o modelo de regras gráficas proposto por Qiao (2007) tem seu foco na expansão do modelo de regras ECA, para atender eventos temporais na execução das regras ativas.

Ainda em relação a mecanismos de monitoramento de banco de dados, Oliveira (2013) descreve em seu artigo a importância da segurança em bancos de dados, e enfatiza os princípios de segurança da informação como: confiabilidade, integridade e disponibilidade.

Oliveira (2013) também apresenta medidas de boas práticas de segurança, como: segurança física, redução da área de ataque, criação de contas de serviços específicos, autenticação e criptografia. Entretanto, por mais que sejam apresentadas boas práticas de monitoramento e segurança, não é apresentada nenhuma implementação, sendo feitas apenas algumas referências ao mecanismo nativo de auditoria do SGBD *Microsoft SQL Server 2008*.

O trabalho de Simon (2008) apresenta uma aplicação que utiliza um Sistema Gerenciador de *Streams* de Dados (SGSD) para prover a auditoria em um banco de dados.

Essa aplicação permite ao usuário administrador do banco de dados, definir os parâmetros de auditoria e os dados a serem auditados através de consultas.

Apesar de a aplicação permitir a parametrização das consultas para a auditoria sobre o banco de dados, ela fica limitada às alterações efetuadas nos dados, e também fica dependente de um SGSD para verificar os registros dos logs.

O trabalho de Badia (2003) propõe uma extensão dos principais mecanismos *(triggers)* de SBDA, para monitorar eventos complexos disparados contra o banco de dados. Para tanto, são desenvolvidos algoritmos para monitoração da evolução das alterações ocorridas no banco de dados. Este trabalho fica limitado ao monitoramento e atuação somente sobre os registros das alterações ocorridas nos dados, e não nos objetos do banco de dados, ou seja, não contempla o monitoramento dos eventos DDL, que é o foco da arquitetura aqui proposta.

A pesquisa realizada por Lu (2008) propõe uma arquitetura de um sistema de monitoramento das alterações efetuadas no banco de dados. Esse sistema monitora as alterações ocorridas no banco de dados e as registra em um modelo de dados, armazenando o histórico destas alterações efetuadas.

Apesar de ser uma arquitetura mais completa do que a solução proposta por Badia (2003), esta solução também foca somente nas alterações ocorridas nos dados e não nos objetos do banco de dados.

A Tabela 2 apresenta uma análise comparativa entre as principais propostas dos trabalhos analisados nesta seção. Na primeira coluna, estão listadas as características de cada trabalho. Nas colunas subsequentes, na primeira linha, estão os números referentes a cada autor, que estão descritos em uma legenda abaixo da tabela com o respectivo ano de publicação. A última coluna representa a proposta. O identificador “+” indica que o trabalho proposto nesta dissertação abrange tal característica e “-“ que não abrange.

Artigos / Características	1	2	3	4	5	6	7	8	Proposta
Regras ativas implementadas.	+	+	+	+	+	+	+	+	+
Armazenamento dos registros de alterações.	-	-	+	+	-	-	-	-	+
Independente de plataforma.	+	-	+	-	+	+	+	+	+
Baseado em eventos.	+	+	+	+	+	+	+	+	+
Monitoramento de alteração de dados.	+	+	+	+	-	+	+	+	-
Monitoramento de alteração da estrutura, eventos DDL (objetos do banco de dados).	-	-	-	-	-	-	-	-	+

1) BADIA (2003); 2) QIAO (2007); 3) LU (2008); 4) SIMON (2008); 5) HUANG (2009); 6) JIN (2009); 7) MEDINA-MARIN (2009); 8) JUN (2010).

Tabela 2 – Tabela comparativa de trabalhos relacionados à SBDA.

2.5 RECOMENDAÇÃO EM SISTEMAS DE INFORMAÇÃO

2.5.1 Definição de Recomendações

Atualmente vivemos num mundo inundado por informações, que aumentam em uma velocidade muito elevada em decorrência da evolução da Tecnologia da Informação. Todos somos contribuintes e consumidores de informação, e o acesso a essas informações de forma mais objetiva e facilitada, se torna um desafio (PIMENTEL e FUKS, 2012).

A recomendação pode nos ajudar nesse desafio, visando oferecer a partir de um grande volume de informações, aquilo que pode interessar a um grupo ou a somente uma pessoa. O Quadro 1 apresenta uma definição da palavra recomendação de acordo com o dicionário da língua portuguesa, Houaiss.

> **Recomendação** s. f. 1. Ato ou efeito de recomendar. 2. Derivação: por metonímia. Aquilo que adverte; conselho, advertência, aviso. 3. Qualidade de recomendável. Ex.: a grande clientela é sua melhor recomendação.

Quadro 1 – Uma Definição de Recomendação.

2.5.2 Sistemas de Recomendação

Muitos sistemas modernos, especialmente os que funcionam na internet, possuem mecanismos para "aprender as preferências de seus usuários". Outros se baseiam, em boa

parte, nas recomendações destes sistemas na hora de tomar decisões, particularmente na realização de compras. Estes sistemas são denominados Sistemas de Recomendação (SR) (CAZELLA, 2010).

Muitas vezes os SRs podem ser considerados como um tipo particular de personalização, que aprende sobre as necessidades de uma pessoa ou de uma comunidade e, em seguida, de forma proativa, identifica e recomenda a informação que atenda estas necessidades (SMEATON e CALLAN, 2005).

Esses sistemas atuam baseados na personalização da informação. Essa personalização está relacionada com o modo pelo qual a informação e os serviços podem ser ajustados às necessidades específicas de um usuário ou de uma comunidade. Então, pode-se dizer que SRs são utilizados para auxiliar os usuários a identificarem serviços ou produtos de seu interesse, que estejam dentro de uma grande quantidade de opções (SMEATON e CALLAN, 2005).

Os SRs definem uma classe de sistemas que podem ser desenvolvidos utilizando diversas técnicas e com diferentes propósitos, mas com a característica em comum de prover recomendações diversas a seus usuários (VIEIRA e NUNES, 2012). Utilizam filtros de informação, para apresentar itens ou objetos como páginas web, livros, medicamentos, lojas, artigos, ou qualquer coisa que provavelmente seja de interesse do usuário. O princípio desses sistemas se baseia em "o que é relevante pra mim, também pode ser relevante para alguém com interesse semelhante" (SCHAFER, 2001).

Os principais componentes de um SR são cliente e produto. Um produto é um recurso que pode ser de diferentes naturezas, por exemplo: um conteúdo, um arquivo, uma informação, uma pessoa, um objeto. A recomendação é uma função de mapeamento de interesses do cliente para obtenção de um ou mais produtos (PIMENTEL e FUKS, 2012).

Uma recomendação de acordo com *R* para a escolha de *"p"* deve ser feita de tal forma que:

$$R(c, p) = max\ F\ (c, pi) \quad (1)$$

Onde:

"c" representa um cliente que utiliza o SR;

"p" é um conjunto de produtos disponíveis para avaliação;

"pi" $\in$ ***"p"***; e

"F" é uma função que determina a relevância de ***"pi"*** em relação a ***"c"***.

Segundo Pimentel (2012), um processo de SR pode ser apresentado conforme a definição da função descrita acima, que avalia a utilidade de um produto para determinado cliente. Identificamos o usuário de um SR como cliente, para ressaltar a sua importância no processo. Geralmente, *"F"* leva em consideração a semelhança entre os perfis dos clientes (PIMENTEL e FUKS, 2012).

2.5.3 Técnicas de Sistemas de Recomendação

Existem diversas técnicas para auxiliar na geração de recomendações. A seguir são apresentadas algumas destas técnicas utilizadas por SRs.

2.5.4 Recomendação Baseada em Recuperação Direta da Informação

Nessa abordagem, o usuário especifica uma consulta e o SR recupera informações que satisfaçam a esta consulta. É uma técnica simples de ser implementada, porque é baseada em consultas efetuadas diretamente no banco de dados. Para esta abordagem funcionar de forma eficaz, é necessário que a fonte de dados esteja armazenada de forma estruturada e organizada em um banco de dados (CAZELLA, 2009).

Essa técnica pode funcionar da seguinte forma: o usuário informa em um formulário pré-definido no site, a categoria ou produto que deseja pesquisar, e o SR retorna as possíveis sugestões de acordo com as escolhas do usuário. A Figura 4 apresenta um formulário com este exemplo.

Figura 4 – Exemplo de interface utilizando busca direta da informação[11].

[11] *http://busca.livrariasaraiva.com.br/nav/secao/livros_informatica_bancodedados/area/livros_informatica/categoria/livros/0*

Na Figura 4, podemos perceber que a tela do site oferece a possibilidade do usuário fazer uma busca direta. Através da categoria que ele deseja, na parte destacada, nota-se que ele escolhe a categoria de livros. Dentro desta categoria, ele escolhe a área de informática e mais especificamente, a área de banco de dados. Com base nestas informações o SR retorna o conteúdo disponível referente à pesquisa do usuário em questão.

2.5.5 Recomendação Baseada em Filtragem Colaborativa

A técnica de filtragem colaborativa tenta prever o grau de interesse do cliente em determinados produtos, a partir da correlação entre as avaliações feitas por ele e as fornecidas por outros clientes. Basicamente essa técnica consiste em filtrar os produtos para um usuário, com base nas experiências de outros usuários com preferências similares.

Essa técnica busca descobrir as relações entre os usuários por meio de seus padrões de comportamento, "usuários com preferências semelhantes são colocados mais próximos". Então, é possível criar comunidades de usuários, o que permite fazer indicações mais eficientes, ou seja, permite ao SR indicar produtos que podem interessar aos usuários com preferências similares, mas que ainda não os acessaram.

Essa técnica parte do princípio de que os usuários tendem a interessar-se por itens semelhantes aos que demonstraram interesse no passado. Desta forma, é definida a similaridade entre os itens (ADOMAVICIUS et al., 2005).

Na técnica de Filtragem Colaborativa, três fases podem ser consideradas, e explicadas do ponto de vista das recomendações praticadas para um usuário (HERLOCKER, 2000):

1. calcular o peso de cada usuário do sistema em relação à similaridade com o usuário alvo (métrica de similaridade);

2. identificar um subconjunto de usuários com maiores similaridades (vizinhos) para considerar na previsão;

3. normalizar as avaliações e computar as previsões, ponderando as avaliações dos vizinhos com seus pesos.

Esta técnica também é chamada de *"k-nearest-neighbor"* ou *"user-based"* (HERLOCKER, 2000). O primeiro passo descrito, que trata da definição de similaridade, pode ser aplicado através de diversos coeficientes, sendo mais comumente aplicado o coeficiente de correlação de Pearson (CAZELLA, 2009). A seguir são apresentadas duas equações e a exemplificação desta técnica extraídos do trabalho de Cazella (2005).

$$w_{a,u} = \frac{\sum_{i=1}^{m} \left[\left(r_{a,i} - \bar{r}_a \right) * \left(r_{u,i} - \bar{r}_u \right) \right]}{\sqrt{\sum_{i=1}^{m} \left(r_{a,i} - \bar{r}_a \right)^2 * \sum_{i=1}^{m} \left(r_{u,i} - \bar{r}_u \right)^2}} \quad \textbf{(2)}$$

Na equação (2) de Cazella (2005), $w_{a,u}$ é a correlação do usuário ativo a com um determinado usuário u; $r_{a,i}$ é a avaliação do usuário ativo a para o item i; $\overline{r_a}$ é a média de todas as avaliações do usuário ativo a; $r_{u,i}$ é o conjunto de avaliações dos usuários semelhantes; e $\overline{r_u}$ é a média das avaliações dos usuários semelhantes (CAZELLA, 2005).

Observa-se a necessidade de mais de uma avaliação em comum para que a correlação seja viável, e os resultados variam entre 1 para similaridade completa, e -1 para nenhuma similaridade. Segundo Cazella (2005), o cálculo da previsão pode ser realizado através da equação (3).

$$p_{a,i} = \bar{r}_a + \frac{\sum_{u=1}^{n} \left(r_{u,i} - \bar{r}_u \right) * w_{a,u}}{\sum_{u=1}^{n} \left| w_{a,u} \right|} \quad \textbf{(3)}$$

O valor da previsão $p_{a,i}$ do item i para o usuário ativo a, é a média ponderada das avaliações dadas ao item i pelos n vizinhos u do usuário ativo a. A quantidade n dos vizinhos mais próximos, ou seja, os valores similares, é uma escolha de cada sistema que utiliza a técnica de filtragem colaborativa (CAZELLA, 2005).

Por padrão a entrada de um algoritmo de filtragem colaborativa pode ser uma matriz, onde as linhas representam os usuários e as colunas os produtos. Por exemplo, conforme a Tabela 3, o usuário U2 pontuou os itens de A até J com as seguintes notas [5; 1; 2; 2; 1; 7; 1; 6; 5], em uma escala de avaliação de [1-7], referindo-se a itens de menor interesse (1) a itens mais interessantes (7), conforme sua avaliação. Como se pode perceber, para o item H o usuário não fez uma avaliação (CAZELLA, 2005).

Usuários	Produtos									
	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**
U1	1	-	-	2	1	7	-	-	2	3
U2	5	1	2	2	1	7	1	-	6	5
U3	1	5	6	4	3	1	2	-	-	2
U4	6	5	4	1	2	-	-	5	-	3
U5	1	-	2	5	6	7	-	1	-	4
U6	-	2	-	3	-	1	-	2	-	-
U7	4	2	2	-	-	6	1	6	-	6
U8	-	1	-	3	7	6	-	3	-	1
U9	6	-	-	6	6	2	-	-	5	6

Tabela 3 – Recomendação baseada em filtragem colaborativa (CAZELLA, 2005).

De acordo com o exemplo de Cazella (2005), apresentado na Tabela 3, o usuário U2 não gostou do item B, avaliando-o com o valor 1. Por outro lado, este usuário gostou do item I, avaliando-o com valor 6. Pode-se observar também que o usuário U2 tende a concordar com U7 e discordar de U3 de acordo com o valor de pontuação apresentado.

Portanto, a filtragem colaborativa recomenda itens de acordo com a similaridade de avaliações fornecidas pelos usuários para os mesmos itens. Para medir essa similaridade, pode-se utilizar o coeficiente de Pearson (TORRES, 2004).

Tendo novamente como exemplo os usuários U2 e U7 da Tabela 3, as avaliações dos itens em comum seriam as pontuações [5; 1; 2; 7; 1; 5] e [4; 2; 2; 6; 1; 6] respectivamente, gerando uma média de 3,5 para ambos os usuários (CAZELLA, 2005).

Analisando as avaliações dos usuários U2 e U7 pode-se verificar que são semelhantes, conforme a concordância em suas avaliações. Tendo como usuário alvo U2 e aplicando o algoritmo do coeficiente de Pearson em todos os usuários, têm-se os seguintes resultados apresentados na Tabela 4 a seguir:

Usuário	Pearson (U2)
U1	0,63
U3	-0,66
U4	0,34
U5	0,09
U6	-0,78
U7	0,94
U8	0,11
U9	-0,66

Tabela 4 – Similaridade através do coeficiente de Pearson (CAZELLA, 2005).

Então, conforme os resultados da Tabela 4 verifica-se que o usuário U2 tem uma similaridade com o usuário U7, é relativamente similar a U4 e não é similar a U3, U6 e U9 em

preferências. Identificado quais são os usuários semelhantes ao usuário alvo, pode-se então gerar a previsão de novos itens para este usuário (CAZELLA, 2005).

Aplicando-se a Equação (3) para prever a nota que o usuário U2 atribuiria ao item H, e levando em consideração todos os itens que os usuários, com limiar superior a 0,7 (limite assumido), pontuaram em comum com o usuário alvo, tem-se o resultado de 5,22. Este resultado indica que caso o usuário U2 consumisse o item H este o avaliaria com uma nota igual a 5,22 (previsão) para este item tendo como base a avaliação dos vizinhos mais próximos (CAZELLA, 2005).

Herlocker descreve uma revisão dos objetivos, base de dados, e algoritmos referentes à filtragem colaborativa em (HERLOCKER et al., 2004).

2.5.6 Recomendação Baseada em Filtragem de Conteúdo

Essa técnica se baseia nas informações sobre o conteúdo dos itens. A filtragem por conteúdo utiliza algoritmos de aprendizagem de máquina, para "casar" o perfil do usuário com o conteúdo dos itens a serem recomendados, tendo em vista uma descrição das características dos conteúdos (CAZELLA, 2009). Por exemplo, livros podem ser categorizados em economia, saúde ou tecnologia. Assim, o livro é classificado a partir da maioria das características que se enquadram em uma determinada categoria.

Na filtragem baseada em conteúdo, os itens do sistema geralmente possuem atributos com diferentes pontuações, e estão associados a classes pré-definidas. Uma forma de se trabalhar com essa abordagem, é por meio de uma solicitação de análise de itens feita ao próprio usuário, onde alguns itens são avaliados e se estes são de interesse ou não. Concluída esta avaliação, o SR localiza itens que estão "casando" em conteúdo com o que foi classificado como interesse, e desconsidera os que estão "casando" em conteúdo com o que foi classificado de não interesse (CAZELLA, 2009).

Essa abordagem é composta por 4 fases:

1) classificação dos itens avaliados de acordo com as categorias pré-definidas;
2) para cada categoria é calculada a avaliação média de cada avaliador;
3) os itens avaliados são ordenados, para obter listas de preferências ordenadas por categorias;
4) e por fim, é calculada a previsão de avaliação segundo a equação (4). A predição da unidade do produto ***"i"*** para o usuário ***"u"*** é calculada como a avaliação média do

produto ***"i"*** ajustada pela avaliação média (normalizada) demonstrada pelo usuário ***"u"*** para os produtos classificados na classe "categoria".

$$P_{u,i} = \frac{\overline{r_i} * \overline{r}_{categoria,\,u}}{\overline{r}_{categoria}} \quad (4)$$

Onde:

$\overline{r}_{categoria}$ → é a avaliação média da categoria.

$\overline{r_i}$ → é a avaliação média do item ***i***.

$\overline{r}_{categoria,\,u}$ → é a avaliação média do usuário ***u*** da categoria da qual o item pertence.

A técnica de recomendação baseada em filtragem de conteúdo dispõe de algumas vantagens (PIMENTEL e FUKS, 2012):

- Não depende de dados de outros usuários.
- Permite recomendar itens novos ou não populares.
- Permite efetuar recomendações a usuários com preferências exclusivas.
- Permite fornecer explicações sobre itens recomendados, listando as características do conteúdo que conduziram à recomendação.

Entretanto, também apresenta algumas desvantagens:

- Recomendações estáticas, não consideram a opinião dos usuários.
- É necessário que o conteúdo seja codificado com características inteligíveis, porque muitas vezes o conteúdo é analisado automaticamente para se identificar as categorias.
- Possui baixa eficiência se o conteúdo não for muito informativo.

2.6 APLICAÇÕES DE SISTEMAS DE RECOMENDAÇÃO

Nesta seção são apresentadas algumas aplicações de Sistemas de Recomendação, onde se consideram exemplos acadêmicos e comerciais.

2.6.1 Projetos Acadêmicos

A seguir são apresentados alguns Sistemas de Recomendação desenvolvidos em projetos acadêmicos:

2.6.1.1 RINGO

A exploração de Sistemas de Recomendação começou a se estabelecer na década de 1990. Um dos primeiros SR foi o RINGO (SHARDANAND e MAES, 1995), sistema que recomendava músicas a partir do perfil do usuário, analisando e identificando semelhanças entre as preferências dos diferentes usuários, para fazer a recomendação das músicas. Este sistema comparava os perfis dos diversos usuários para identificar quais apresentavam preferências similares, ou seja, gostavam das mesmas músicas, álbuns, cantores e/ou dos quais não gostavam. Mais detalhes sobre este SR podem ser vistos em (SHARDANAND e MAES, 1995).

2.6.1.2 Grouplens

O Grouplens foi um dos primeiros sistemas a introduzir a filtragem colaborativa automática (KONSTAN et al., 1997). Este sistema tinha como base as avaliações (valores entre 1 e 5) feitas por usuários sobre os artigos que haviam lido. Então, utilizava essas avaliações para identificar os vizinhos mais próximos com avaliações semelhantes, e assim prever se os usuários poderiam gostar de outros artigos. Mais tarde esse sistema evoluiu para o MovieLens.

O Grouplens trouxe duas novas e importantes características: (1) uma arquitetura que possibilitava que as avaliações fossem compartilhadas por vários usuários e (2) implementava avaliações agregadas de diversos usuários, com base na similaridade de suas avaliações passadas (RESNICK et al., 1994).

A técnica de filtragem colaborativa aplicada neste sistema consiste em um método de classificação, que utiliza o paradigma do aprendizado baseado em instâncias. O paradigma tem como pressuposto que se duas instâncias são similares, então elas pertencem à mesma classe. Desta forma, quando uma nova instância é similar a uma instância conhecida, a classe desta é atribuída à nova instância (RESNICK et al., 1997).

2.6.1.3 MovieLens

Este sistema foi resultado da evolução do Grouplens e tem o objetivo de realizar recomendações de filmes, geradas a partir de correlações entre avaliações dos usuários.

O projeto está disponível ao público, através do site *http://movielens.umn.edu/login*. É permitido efetuar um cadastro para ter acesso ao MovieLens e fazer algumas avaliações, formar um perfil inicial e receber recomendações de filmes (HERLOCKER et al., 1999).

A Figura 5 mostra a tela de avaliações do MovieLens, três colunas são apresentadas, a primeira representa a avaliação (número de estrelas) feita para o filme em questão; a segunda é onde o usuário faz sua avaliação selecionando uma classificação entre 0,5 e 5,0 pontos; e a terceira apresenta o nome e categoria do filme.

Esse sistema utiliza a mesma técnica de filtragem colaborativa do Grouplens, alterando a instância de avaliação de artigos para filmes (HERLOCKER et al., 1999).

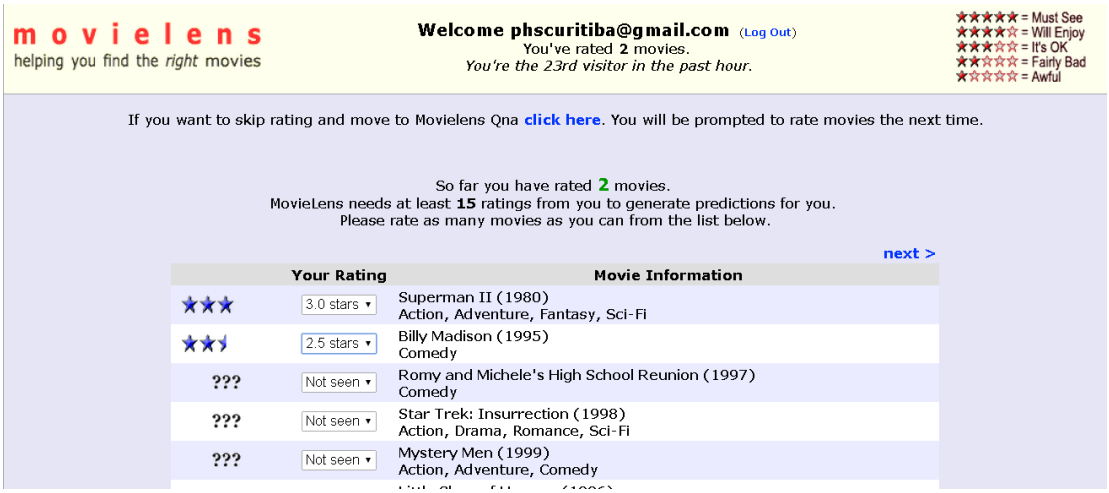

Figura 5 – Interface do Sistema de Recomendação MovieLens[12].

2.6.2 Sites Comerciais

A seguir são apresentados exemplos de Sistemas de Recomendação em sites comerciais:

2.6.2.1 Amazon.com

O site da Amazon (www.amazon.com) é um portal de comércio eletrônico com diversas categorias de produtos, e também possui algumas variações de Sistemas de Recomendação. Estas variações são apresentadas em (HERLOCKER et al., 2000), apontando as características de cada uma delas. A seguir são apresentadas algumas destas variações descritas.

Clientes que compraram: Como muitos sites de comércio eletrônico, o da Amazon está estruturado com uma página de informações para cada livro, dando detalhes do texto e

[12] *http://movielens.umn.edu*

informações de compra. São duas listas de recomendação separadas. A primeira recomenda livros frequentemente comprados por clientes que compraram o livro selecionado. Enquanto a segunda lista recomenda autores cujos livros são frequentemente comprados por clientes que compraram obras do autor do livro selecionado.

Suas recomendações: A Amazon incentiva a avaliação direta dos clientes, sobre livros que leram. O cliente pode classificar os livros lidos em uma escala de 1 a 5 pontos. Depois de avaliar uma amostra de livros, os clientes podem solicitar recomendações de livros que podem lhes interessar.

Amazon.com oferece: Os clientes podem escolher categorias ou gêneros específicos em uma lista, e periodicamente os editores da Amazon.com enviam por e-mail, suas últimas recomendações referentes a cada categoria.

Ideias de presente da livraria: O recurso de ideias de presente permite aos clientes receber recomendações dos editores. Os clientes podem escolher uma categoria de livros, que eles gostariam de receber algumas sugestões.

Comentários do cliente: este recurso permite aos clientes receber recomendações em formato texto, com base nas opiniões de outros clientes. Na página de cada livro estão as classificações em uma escala de 1 a 5 e comentários fornecidos por clientes que leram o livro.

2.6.2.2 eBay

O *site* de leilões *on-line* eBay.com, também possui algumas estratégias de recomendação, como por exemplo, as descritas a seguir.

Direito de resposta: este recurso permite aos compradores e aos vendedores avaliarem seus parceiros de negócio. A avaliação consiste em um índice de satisfação e representa a confiabilidade de cada negociante. A avaliação é usada para fazer recomendações para os compradores, que podem verificar o perfil dos negociantes. Quanto mais positiva a pontuação, mais confiável é o negociante, e quanto mais negativa a pontuação, menos confiável é o negociante.

Comprador pessoal: este recurso permite aos clientes indicar os itens que eles estão interessados em comprar. Periodicamente, o site realiza uma busca por estes itens e envia as recomendações por e-mail para os clientes.

2.7 TRABALHOS CORRELATOS DE SISTEMAS DE RECOMENDAÇÃO

Assim como no levantamento bibliográfico de SBDA, também foram analisados diversos trabalhos relacionados a Sistemas de Recomendação, e cada um destes trabalhos possui funcionalidades em comum e também distintas. A seguir são descritas as principais características destes trabalhos, e ao final da seção, a comparação com o trabalho proposto.

Nas empresas, as pessoas aprendem compartilhando conhecimento. Este tipo de tarefa, definida como colaboração, é importante para o aprendizado corporativo.

A colaboração pode ser auxiliada por ferramentas de tecnologia da informação como *chats*, *newsgroups*, fóruns e listas de discussão por e-mail. Entretanto, este tipo de auxilio permite somente a comunicação, não colaborando com as pessoas no processo de aprendizagem (LOH et al., 2002).

A arquitetura proposta por Loh (2002) apresenta um sistema de recomendação para apoiar na colaboração entre pessoas em uma empresa. Este sistema analisa mensagens textuais enviadas durante uma sessão de diálogo, identifica o contexto da discussão e sugere documentos, autoridades (as pessoas com competência em um assunto) e discussões passadas dentro do mesmo contexto.

O trabalho apresentado pelo autor tem como base um repositório de informações, onde o sistema analisa mensagens trocadas e efetua as recomendações de acordo com o contexto encontrado. Entretanto, sua aplicação é direcionada para utilização de um repositório de informações referentes a dados específicos, para auxiliar usuário de uma empresa e não no contexto de eventos DDL em bancos de dados.

Segundo Eirinaki (2013), SGBDs modernos fornecem recursos sofisticados para ajudar os usuários a organizar, armazenar, gerenciar e recuperar informações em um banco de dados. Entretanto, sua capacidade para gerir as consultas que os usuários executam sobre os dados, é um tanto limitada (EIRINAKI et al., 2013).

Nesse contexto, Eirinaki (2013) apresenta um SR para gestão de consultas colaborativas, que permite aos usuários executar tarefas simples, como navegar no log de todas as consultas apresentadas. Desta forma, os usuários são capazes de encontrar rapidamente, editar e reexecutar consultas efetuadas anteriormente no banco de dados. O sistema também analisa o log de consultas efetuadas anteriormente e recomenda possíveis consultas aos usuários, para ajudá-los a identificar consultas mais específicas que podem

atender melhor sua necessidade. O trabalho do autor envolve SGBD e SR, entretanto, não aborda eventos DDL que é o diferencial da arquitetura proposta neste trabalho.

Outros trabalhos relacionados à SR podem ser vistos, em (ADOMAVICIUS, 2005), (ANSARI et al., 2000), (SARWAT et al., 2013), (KANAGAL, 2012) e (AKBARNEJAD et al., 2010). Entretanto, nenhum destes trabalhos aborda a integração de SR com SDBA.

A arquitetura proposta por nosso trabalho utiliza de forma adaptada, a ideia aplicada por Eirinaki (2013). Entretanto, a aplicação do SR em nossa arquitetura, é em um contexto diferenciado, ou seja, o SR será desenvolvido para atuar com recomendações baseadas em eventos DDL, responsáveis pelas alterações efetuadas em objetos do banco de dados, como por exemplo, criação de tabelas. Neste contexto, de acordo com o banco de dados monitorado, e com base em um repositório de alterações, o SR identificará possíveis recomendações de criação de tabelas que podem ser aplicadas nos bancos de dados monitorados.

A Tabela 5 apresenta uma análise comparativa entre as diferentes propostas dos trabalhos analisados nesta seção. Na primeira coluna, estão listadas as características de cada trabalho. Nas colunas subsequentes, na primeira linha, estão os números referentes a cada autor, que estão descritos em uma legenda abaixo da tabela com o respectivo ano de publicação. A última coluna representa a proposta. O identificador "+" indica que o trabalho proposto nesta dissertação abrange tal característica e "-" que não abrange.

Artigos / **Características**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**Proposta**
Integração com SBDA	-	-	-	-	-	-	-	+
Recomendações sobre eventos DDL	-	-	-	-	-	-	-	+
Independente de plataforma	+	+	+	+	+	+	+	+
Monitoramento de alteração de objetos do banco de dados	-	-	-	-	-	-	-	+
Integração com SGBDs	-	-	-	-	+	+	+	+
Recomendação sobre eventos DML[13]	-	-	-	-	+	+	+	-

1) LOH (2002); 2) ADOMAVICIUS (2005); 3) ANSARI (2000); 4) AKBARNEJAD (2010); 5) KANAGAL (2012); 6) SARWAT (2013); 7) EIRINAKI (2013).

Tabela 5 – Tabela comparativa de trabalhos relacionados com SR.

[13] Instruções *DML – (Data Manipulation Language)* permitem consultas *(select)* ou manipulações *(insert, delete, update)* dos dados armazenados em bancos de dados. PRICE, J. SQL Domine SQL e PL/DQL no banco de dados Oracle, c. 1, p. 31. São Paulo: ARTMED Editora S.A. 2009.

3 METODOLOGIA

A metodologia seguida para o desenvolvimento da arquitetura proposta decorreu conforme descrito a seguir:

- Revisão bibliográfica, referente aos trabalhos relacionados a Sistemas Gerenciadores de Banco de Dados, Sistemas de Banco de Dados Ativos, Semântica e Sistemas de Recomendação.
- Levantamento das variáveis de informações que deveriam ser registradas ao ocorrer à alteração de objetos (eventos DDL) no banco de dados monitorado. Estas informações são relacionadas à estrutura do ambiente de banco de dados, por exemplo, data de alteração, nome de servidor, instância, bancos de dados, esquema, tipo de objeto, identificação do objeto, etc.
- Especificação dos repositórios necessários para a arquitetura proposta – Repositório de Alterações, de Parâmetros de Recomendação, de Regras, de Sintaxe e Semântica.
- Especificação da arquitetura proposta – definição dos principais módulos desenvolvidos e a interação destes módulos.
- Especificação do tipo de interface desenvolvida para utilização do sistema.
- Implementação do protótipo inicial e testes em ambiente disponibilizado por uma empresa de serviços de locação e transporte.
- Levantamento e verificação das limitações e possíveis problemas do sistema.
- Avaliação geral e conclusões.

4 ARQUITETURA PROPOSTA

4.1 VISÃO GERAL

A proposta deste trabalho é explorar a aplicabilidade e integração de SBDA com SR, no monitoramento automático de eventos DDL ocorridos em bancos de dados. Neste contexto, foi desenvolvida uma arquitetura com a integração dos mecanismos necessários para prover esta automatização, gerando os registros pertinentes às alterações efetuadas nos objetos dos bancos de dados monitorados, armazenando-os em um repositório padronizado para estas informações.

Os registros das alterações são disponibilizados através de uma interface web, e podem ser utilizados na auditoria; no monitoramento das alterações; e por um SR, para recomendar possíveis mudanças relacionadas aos eventos DDL aplicados nos bancos de dados monitorados.

O SR da arquitetura proposta distingue-se dos trabalhos descritos até aqui, por tratar as recomendações em um contexto de SBDA, o que não foi encontrado nos diversos trabalhos analisados. Além disso, esta arquitetura permite a geração de recomendações para SGBDs distintos, a partir da análise do log centralizado de alterações ocorridas nos bancos de dados monitorados.

A arquitetura proposta utiliza-se de aspectos de SBDA, em que regras ativas são aplicadas nos bancos de dados e são disparadas as ações correspondentes conforme a regra acionada. A Figura 6 apresenta o modelo geral da arquitetura e nas seções seguintes são descritas suas camadas e componentes. A Figura 7, na Camada de Bancos de Dados Monitorados, enfatiza que a arquitetura proposta permite o monitoramento de diversos SGBDs.

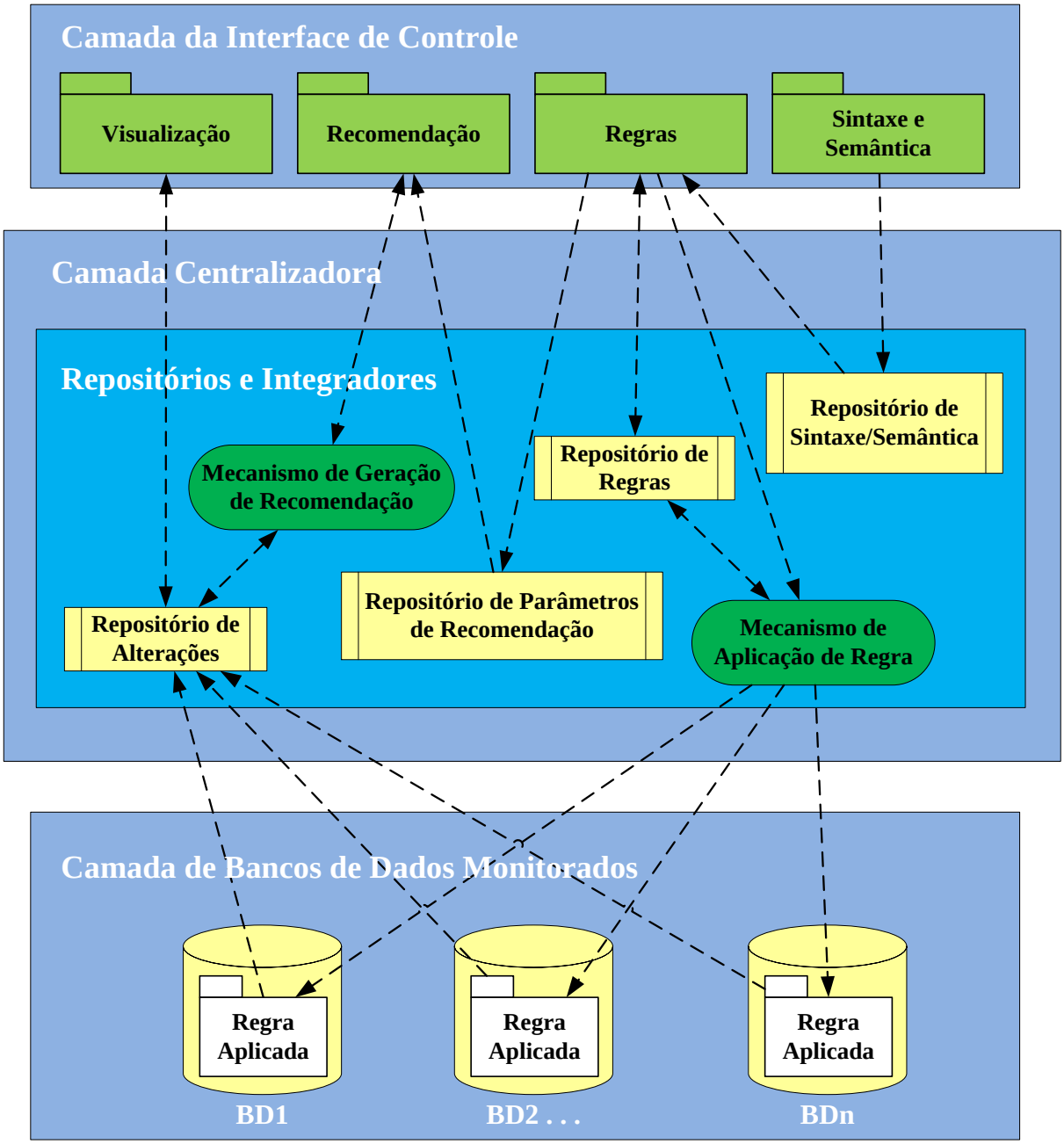

Figura 6 – Arquitetura Geral.

4.2 CAMADA DE BANCOS DE DADOS MONITORADOS

Os bancos de dados monitorados são os principais alvos de atuação na arquitetura proposta. Para o monitoramento dos eventos DDL disparados contra estes bancos de dados, são aplicadas regras ativas (representado na Figura 6 como: “Regra Aplicada”) transformadas em gatilhos *(triggers)*, principais mecanismos de um SBDA para identificação e registro de

como, quando e quem efetuou a alteração ou criação de um objeto na estrutura do banco de dados monitorado.

Estes registros formam um histórico de alterações que fica armazenado no “Repositório de Alterações”, descrito neste trabalho na seção 4.3.2. Este repositório é utilizado pelo Mecanismo de Geração de Recomendação da Camada Centralizadora e pelo Módulo de Recomendação da Camada da Interface de Controle, descritos neste trabalho nas seções 4.3.7 e 4.4.2 respectivamente, para recomendar possíveis criações de tabelas no banco de dados monitorado. A listagem "BD1, BD2... e BDn", representa os diferentes bancos de dados que podem ser monitorados. Na Figura 7 podem ser observados alguns exemplos de SGBDs que correspondem aos bancos de dados monitorados.

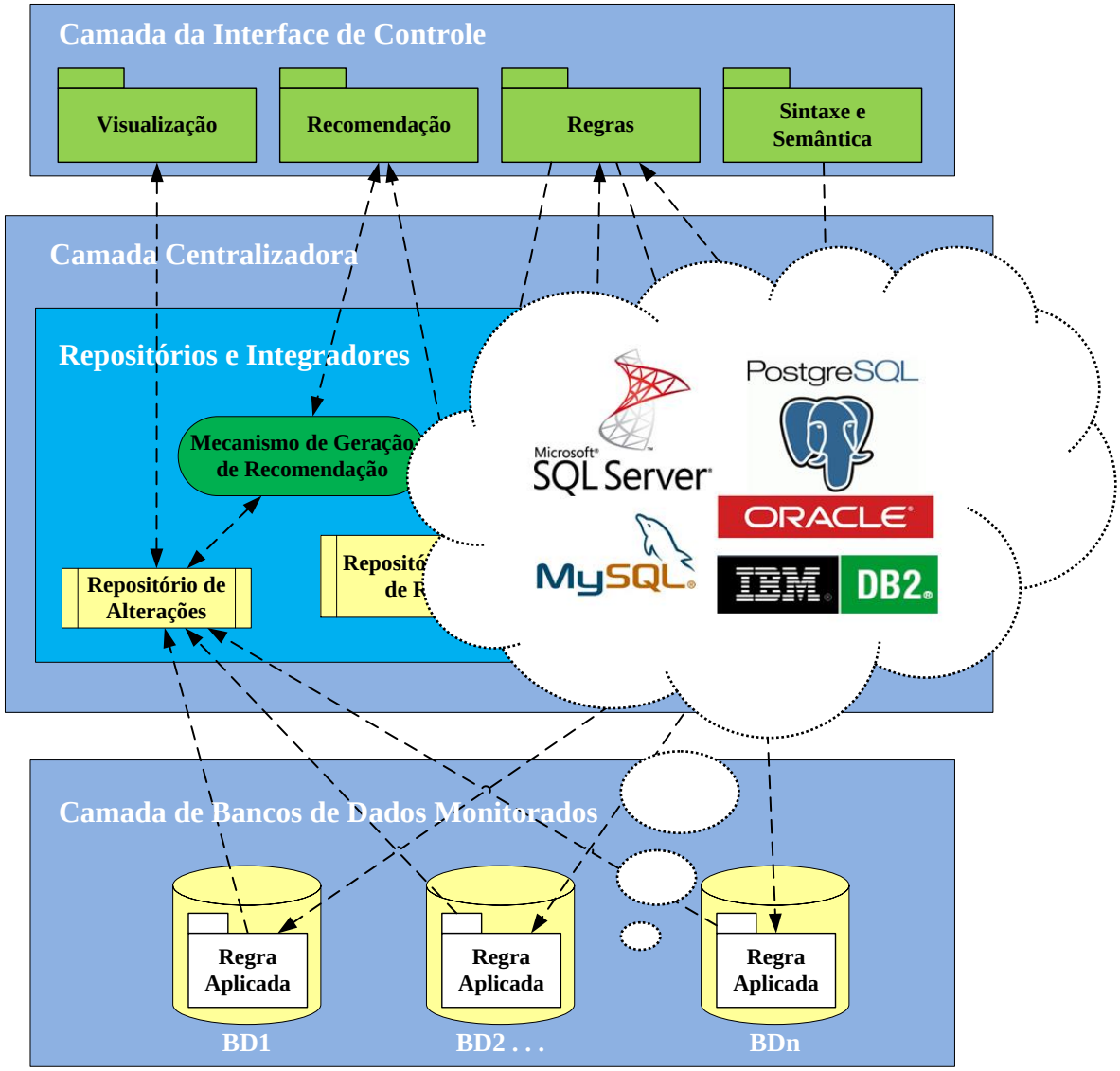

Figura 7 – Monitoramento de diversos tipos de SGBDs.

4.3 CAMADA CENTRALIZADORA

Esta é uma camada do SBDA que contempla os repositórios, e os integradores que são os mecanismos de geração de recomendação e aplicação de regras. Os componentes desta camada serão descritos nas subseções seguintes.

4.3.1 Repositórios

Disponíveis na Camada Centralizadora, os repositórios são responsáveis pelo armazenamento das informações referentes às alterações efetuadas por eventos DDL nos bancos de dados; informações dos parâmetros de consulta das recomendações a serem realizadas pelo SR; informações das regras; e informações da sintaxe e semântica de regras referente a cada banco de dados monitorado.

Nesta arquitetura são definidos repositórios distintos, para cada grupo de informação dos SGBDs monitorados (alterações, parâmetros de recomendação, regras, sintaxe e semântica). Isso permite o tratamento das informações de diferentes instâncias, uma vez que podem ser armazenadas as informações de cada tipo de SGBD, ou seja, é possível armazenar informações dos diferentes bancos de dados monitorados, e os mecanismos desenvolvidos na arquitetura proposta fazem uso das informações contidas nestes repositórios, para gerar as regras de acordo com a sintaxe e semântica de cada SGBD.

4.3.2 Repositório de Alterações

Este repositório é responsável pelo armazenamento das informações referentes às alterações realizadas por eventos DDL no banco de dados monitorado pelas regras ativas aplicadas. As informações armazenadas neste repositório são compostas, por exemplo, pelas variáveis: nome do servidor, nome da instância, nome do esquema, tipo do objeto, nome do objeto, data da alteração, script do objeto alterado, etc.

A partir deste repositório, é possível extrair o histórico das alterações ocorridas nos bancos de dados monitorados, ou seja, outros módulos podem utilizar este repositório. Por exemplo, é possível implementar módulos de relatórios, para emitir relatórios gerenciais ou realizar a auditoria das alterações efetuadas no banco de dados monitorado.

As informações armazenadas neste repositório são referentes aos eventos DDL executados nos bancos de dados monitorados. Por exemplo, a criação, alteração ou remoção de uma tabela no banco de dados, seriam monitoradas e armazenadas neste repositório.

4.3.3 Repositório de Parâmetros de Recomendações

Este repositório armazena as informações referentes aos parâmetros utilizados pelo usuário para solicitar uma recomendação. Por exemplo, para um usuário receber recomendações de possíveis tabelas que devem ser criadas em determinados bancos de dados monitorados, ele deverá informar o servidor e instância, estas informações são consultadas neste repositório.

4.3.4 Repositório de Regras

Este repositório armazena as regras ativas cadastradas para serem aplicadas nos bancos de dados monitorados.

A Camada da Interface de Controle utiliza este repositório para gravar as regras que serão aplicadas nos bancos de dados. O Módulo de Regras da Camada de Interface e Controle, busca a regra neste repositório e a envia para o Mecanismo de Aplicação de Regra, para ser aplicada no banco de dados correspondente.

Quando um novo banco de dados é criado, uma regra de monitoramento de eventos DDL pode ser aplicada nesta base de dados (por exemplo, eventos relacionados à criação de tabelas e índices). As informações que os componentes descritos acima precisarão para tal procedimento estarão armazenadas neste repositório.

A Tabela 6 apresenta os exemplos das informações armazenadas que estas regras utilizarão. O campo (SGBD) detém a informação do SGBD no qual será aplicada a regra; o campo (Tipo Regra) o tipo da regra, por exemplo, se é uma regra de evento DDL; o campo (Nome Regra) identifica o nome que foi definido para a regra criada, e por fim; o campo (Descrição Regra) a descrição da sequência de execução do processo de evento, condição e ação da regra sobre o banco de dados onde está aplicada.

Regras				
Cod.	**SGBD**	**Tipo Regra**	**Nome Regra**	**Descrição Regra**
1	SQL Server	trgevtddl	monitcriatbl	E: Ao ser compilado um objeto no banco de dados. C: Se o evento for um comando DDL de criação de tabela. A: Armazenar o log do evento no repositório de alterações.
2	SQL Server	trgevtddl	monitcriaidx	E: Ao ser compilado um objeto no banco de dados. C: Se o evento for um comando DDL de criação de índice. A: Armazenar o log do evento no repositório de alterações.
. . .	. . .	. . .	. . .	. . .

Tabela 6 – Tabela das informações armazenadas no repositório de regras.

4.3.5 Repositório de Sintaxe/Semântica

Este repositório armazena as informações de sintaxe e semântica, para criação de regras para cada SGBD. O cadastro destas informações é realizado através do módulo de Sintaxe e Semântica, presente na Camada da Interface de Controle. Este repositório também é utilizado pelo módulo de Regras da Camada da Interface de Controle, para buscar a sintaxe e semântica para cada necessidade de criação e aplicação de regras através do módulo e mecanismo responsável.

Por exemplo, para a criação de uma regra em um banco de dados, em um determinado SGBD, antes é necessário consultar este repositório para identificar a sintaxe e semântica desta regra de acordo com o SGBD em questão.

A Tabela 7 apresenta exemplos de informações armazenadas neste repositório. O campo (SGBD) detém a informação de qual SGBD a sintaxe deste registro representa; o campo (Regra Utilizada) identifica o nome da regra; o campo (Tipo Objeto) identifica o tipo de objeto e por fim; o campo (Script) identifica a sintaxe e semântica na qual a regra vai ser compilada no banco de dados.

Sintaxe e Semântica				
Cod.	**SGBD**	**Regra Utilizada**	**Tipo Objeto**	**Script**

1	SQL Server	monitcriatbl	Trigger	`CREATE TRIGGER @NomeRegra = monitcriatbl`
				`ON @Atuacao = DATABASE`
				`FOR @Evento = CREATE_TABLE`
				`AS --Comandos da regra; ...`

2	SQL Server	monitcriaidx	Trigger	`CREATE TRIGGER @NomeRegra = monitcriaidx`
				`ON @Atuacao = DATABASE`
				`FOR @Evento = CREATE_INDEX`
				`AS --Comandos da regra; ...`

...	...	...	...	...

Tabela 7 – Tabela das informações armazenadas no repositório de Sintaxe/Semântica.

4.3.6 Mecanismo de Aplicação de Regra

Esse mecanismo é responsável pela aplicação das regras nos bancos de dados para serem monitorados. Para tanto, ele interage com o módulo de Regras da Camada da Interface de Controle, o qual é utilizado pelo usuário para acionar esse mecanismo, e também com o Repositório de Regras, onde se busca a regra a ser aplicada no banco de dados desejado.

Por exemplo, quando o usuário precisa aplicar uma regra de monitoramento, de evento DDL, de criação de tabelas em um banco de dados, esse mecanismo é acionado. Então, a regra selecionada é compilada no formato de um *trigger* de acordo com a sintaxe do banco de dados em questão.

4.3.7 Mecanismo de Geração de Recomendação

Esse mecanismo é responsável pela geração de recomendações referentes a um servidor e instância, escolhidos pelo usuário no momento da consulta de recomendação.

O mecanismo de geração de recomendação interage com o módulo de Recomendação da Camada da Interface de Controle, o qual é utilizado pelo usuário para acionar este mecanismo, e com o repositório de alterações, onde o mecanismo consulta os registros para identificar os eventos DDL executados no banco de dados monitorado, para efetuar as recomendações pertinentes.

Neste trabalho utilizamos o algoritmo de recomendação proposto por Eirinaki et al. (2013). Este algoritmo faz parte de um framework que gera recomendações de possíveis consultas a partir da sessão atual $\boldsymbol{S_0}$ de um usuário, combinando a consulta da sessão atual com um modelo preditivo gerado a partir das consultas feitas por outros usuários $\boldsymbol{S_i}$ em sessões passadas.

No framework proposto por Eirinaki et al. (2013), cada consulta feita no banco de dados é armazenada em um repositório, o qual é consultado pelo algoritmo para calcular a similaridade entre a consulta da sessão atual e as consultas já realizadas no banco de dados.

Para tanto, a equação (5) apresenta a primeira fase do processo de recomendação proposto por Eirinaki et al. (2013). Onde: $\boldsymbol{S_Q}$ representa as tuplas da consulta no banco de dados, e o valor de cada elemento de $\boldsymbol{S_Q}$ representa a importância da tupla $\boldsymbol{\tau}$ como testemunha para $\boldsymbol{Q}$. Para mais detalhes sobre esta equação verificar Eirinaki et al. (2013). Os resultados de nossa implementação podem ser verificados na seção 5.8, na Tabela 10.

$$S_Q[\tau] = \begin{cases} 1 & \text{if } \tau \text{ is a witness;} \\ 0 & \text{if } \tau \text{ is not a witness.} \end{cases} \quad \textbf{(5)}$$

A equação (6) representa a segunda fase do processo de recomendação proposto por Eirinaki et al. (2013), para resolver a matriz montada na primeira fase. Onde, dados os vetores $\boldsymbol{S_Q}$ para cada evento $\boldsymbol{Q}$ representado pelo usuário $\boldsymbol{i}$, é definido o sumário da sessão $\boldsymbol{Si}$ como descrito na equação (6). Os resultados de nossa implementação podem ser verificados na seção 5.8 na Tabela 11.

$$S_i = \sum_{Q \in Q_i} S_Q \quad \textbf{(6)}$$

A terceira e ultima fase do processo de recomendação proposto por Eirinaki et al. (2013) é representada na equação (7). Os resultados de nossa implementação podem ser verificados na seção 5.8 na Tabela 12.

$$S^{\text{pred}} = \alpha \cdot S_0 + (1 - \alpha) \cdot \sum_{i=1,\ldots,n} sim(S_i, S_0) \cdot S_i, \quad \textbf{(7)}$$

Onde $\boldsymbol{\alpha}$ é uma variável que representa um fator de combinação; $\boldsymbol{S_0}$ o usuário da seção atual; $\boldsymbol{S_i}$ cada usuário das seções anteriores em relação ao usuário atual e; ***sim*** $\boldsymbol{(S_i, S_0)}$ representa uma métrica de similaridade entre dois vetores, a saber, neste caso, uma função de similaridade de cosseno apresentada na equação (8).

A similaridade de cosseno é uma função baseada em vetores, que mede a similaridade entre duas cadeias de caracteres utilizando um modelo de vetor (TATA e PATEL, 2007). A equação (8) mostra como o cálculo de similaridade é executado. Um exemplo da execução desta equação pode ser verificado no Anexo C deste trabalho.

$$\frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}} \quad \textbf{(8)}$$

Como a função de similaridade de cosseno faz parte do algoritmo aplicado por Eirinaki et al. (2013), em nosso trabalho também foi implementada esta função para incorporá-la no cálculo de recomendação. Ela está representada na equação (9) como ***"sim (Si, S₀)"*** e a codificação completa encontra-se no Anexo B deste trabalho.

A Figura 8 apresenta o fluxo do processo do framework proposto por Eirinaki et al. (2013).

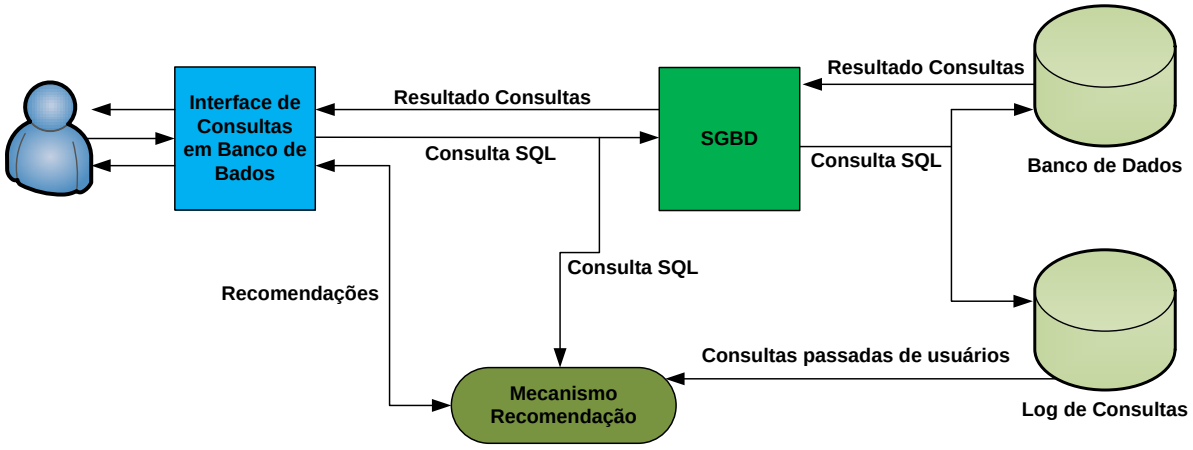

Figura 8 – Processo de Recomendação do Framework QueRIE (Eirinaki et al., 2013).

Note que o processo de recomendação apresentado na Figura 8 consiste de dois módulos principais: “Interface de Consultas de Banco de Dados”, por onde o usuário realiza a consulta no banco de dados alvo. O segundo módulo é o “Mecanismo de Recomendação”, que também recebe esta consulta e a confronta com consultas já realizadas e armazenadas no Log de Consultas, para gerar as recomendações.

O framework também utiliza dois repositórios: o “Banco de Dados”, que recebe as consultas e o de “Log de Consultas”, que armazena o log das consultas realizadas no Banco de Dados. Este sistema interage em tempo real com o banco de dados, utilizando as seções de usuários reais de consultas anteriores para gerar as recomendações. Mais detalhes sobre esta arquitetura podem ser verificados em Eirinaki et al. (2013).

Para o SR desenvolvido na arquitetura proposta neste trabalho, fizemos algumas adaptações no algoritmo, para incorporá-lo ao cenário de SBDA e eventos DDL de criação de tabelas *(CREATE TABLE).* Estas adaptações são detalhadas no capítulo 5 onde é descrito a implementação da arquitetura proposta e de seus mecanismos.

4.4 CAMADA DA INTERFACE DE CONTROLE

A Camada da Interface de Controle incorpora os módulos de interação com as funcionalidades desenvolvidas na arquitetura proposta neste trabalho.

Nesta arquitetura são definidos os módulos de interface descritos a seguir, para interagir com funcionalidades como: cadastro e aplicação de regras; consulta e geração de recomendações; visualização dos registros de alterações; e inclusão de sintaxe e semântica.

4.4.1 Módulo de Visualização

Este módulo é responsável por permitir a visualização das recomendações solicitadas, bem como os registros das alterações ocorridas no banco de dados monitorado.

Através deste módulo é possível analisar o histórico das alterações efetuadas por eventos DDL nos objetos do banco de dados, como por exemplo, em qual esquema, tipo de objeto, nome do objeto e quem efetuou determinada alteração, assim como também visualizar as recomendações realizadas pelo SR para o usuário solicitante.

Este módulo disponibiliza estas informações através de uma interface web, a qual pode ser acessada pelo usuário para visualização das informações desejadas.

4.4.2 Módulo de Recomendação

Este módulo é responsável pela interação com o Mecanismo de Geração de Recomendação e com o Repositório de Parâmetros para Recomendação. Assim, de acordo com os parâmetros selecionados em sua interface, o Repositório de Alterações é consultado para identificar as alterações realizadas no banco de dados monitorado e gerar as recomendações pertinentes.

Por exemplo, quando o usuário fizer uma consulta pela interface do módulo de recomendação, ele irá selecionar o servidor e instância para qual ele deseja consultar as recomendações. Desta forma, o SR fará as recomendações de possíveis criações de tabelas que podem ser realizadas.

Para consultar a recomendação, algumas verificações são necessárias. A Figura 9 apresenta o diagrama de caso de uso com estas verificações.

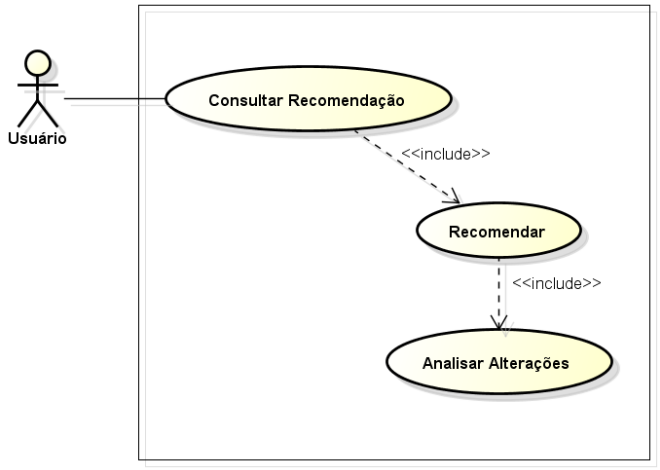

Figura 9 – Caso de uso de consulta de recomendação.

O ator Usuário é responsável pela solicitação da recomendação. Para que a consulta da recomendação possa ser realizada, alguns casos de uso precisam ser acionados, como podemos verificar na Figura 9. Para o caso de uso “Consultar Recomendação” ser completado, os casos de uso “Recomendar” e “Analisar Alterações” precisam ser acionados, ou seja, o processo de recomendação passa por estas etapas para ser realizado.

4.4.3 Módulo de Regras

Através de sua interface, este módulo interage com o Repositório de Regras, para armazenamento das regras criadas; com o Repositório de Sintaxe e Semântica, para encaminhar a regra de acordo com cada SGBD, para o Mecanismo de Aplicação de Regra aplicar a regra no banco de dados especificado.

Por exemplo, quando o usuário for criar uma nova regra, ele utilizará a interface deste módulo, para que ele possa cadastrar a nova regra e armazená-la no Repositório de Regras. Algumas destas regras são apresentadas na Tabela 6 já descritas na seção 4.3.4.

Para o cadastro de regras, algumas verificações são necessárias. A Figura 10 apresenta o diagrama de caso de uso com estas verificações.

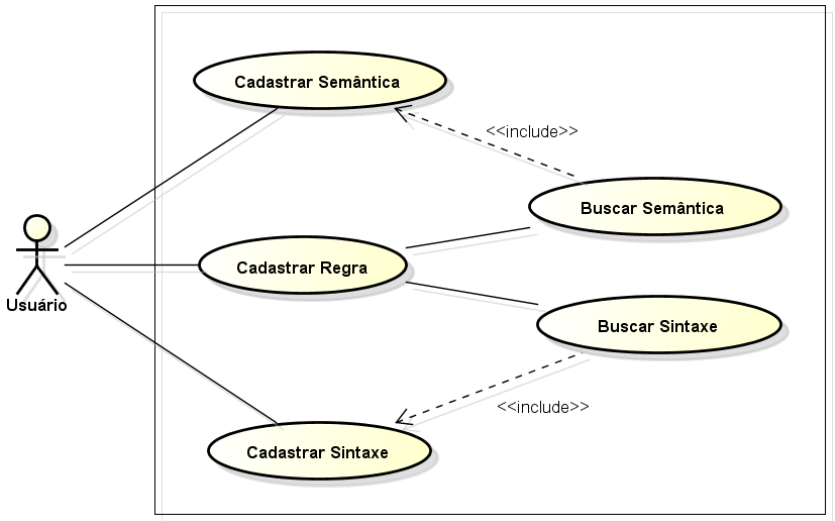

Figura 10 – Caso de uso de cadastro de regras.

O ator Usuário é responsável pelo cadastro de regras, semânticas e sintaxes. Para que o cadastro de uma regra possa ser realizado, alguns casos de uso precisam ser acionados e outros já precisam ter sido executados, como podemos verificar na Figura 10.

Para que o caso de uso “Cadastrar Regra” possa ser completado, ele aciona os casos de uso “Buscar Semântica” e “Buscar Sintaxe”.

Para que o caso de uso “Buscar Sintaxe” e “Buscar Semântica” possam ser executados, é preciso ser cadastrada uma sintaxe e uma semântica, com isso temos um relacionamento de dependência com os casos de uso “Cadastrar Sintaxe” e “Cadastrar Semântica”.

4.4.4 Módulo de Sintaxe e Semântica

Este módulo é utilizado pelo usuário, para incluir as informações de sintaxe e a semântica da regra a ser utilizada de acordo com o SGBD em questão. Através da sua interface, este módulo interage com o Repositório de Sintaxe/Semântica, onde estas informações são armazenadas.

Por exemplo, quando o usuário precisar cadastrar a sintaxe e a semântica correspondente a um SGBD, ele utilizará a interface deste módulo para cadastrar as informações relacionadas e armazená-las no repositório correspondente.

4.5 CONCLUSÕES

O nível de generalização desta arquitetura permite que ela seja explorada em ambientes com SGBDs distintos, por exemplo, em ambientes com SGBDs de fabricantes diferentes como: *Oracle*, *Microsoft SQL Server*, *etc*.

A arquitetura desenvolvida neste trabalho é modular e permite adaptações ou alterações para ampliar sua atuação. Além de gerar recomendações sobre eventos DDL de *CREATE TABLE*, também é possível adaptar ou ampliar seus mecanismos para gerar recomendações sobre outros tipos de eventos, como por exemplo, eventos DML.

Essa arquitetura também permite que o algoritmo de recomendação utilizado, possa ser alterado ou mesmo substituído por outro algoritmo de recomendação, permitindo flexibilidade em sua utilização.

5 IMPLEMENTAÇÃO

Neste capítulo o objetivo é descrever a implementação da arquitetura proposta, considerando: os repositórios de dados para armazenamento de informações contempladas na arquitetura; os mecanismos para controlar a criação e aplicação das regras ativas; os mecanismos de consultas e recomendações a partir do log das alterações por eventos DDL nos bancos de dados monitorados; e a interface web de controle destes mecanismos e repositórios.

5.1 DESCRIÇÃO DO AMBIENTE

Para centralização e armazenamento das informações geradas pelos eventos DDL nos bancos de dados monitorados, e também o controle da interface do sistema, foi configurado um servidor **(A)** de aplicação e banco de dados com a seguinte configuração de hardware: servidor *HP ProLiant BL460c G7*, 2 processadores *Six-Core Intel Xeon* de 2667 MHz, 16 GB de memória RAM, 600 GB de disco SAS de 15 rpm em RAID 5, e 4 interfaces de rede 1 Gbps cada uma.

Neste servidor estão instalados e configurados, o Sistema Operacional *Microsoft Windows Server 2008 R2*, o IIS 7.0 como tecnologia de servidor de aplicação, e o *Microsoft SQL Server 2008 R2* como Sistema Gerenciador de Banco de Dados.

Para implementação da interface, foi utilizada a plataforma *Microsoft .Net (Visual Studio 2012)*. Estas tecnologias foram utilizadas porque o ambiente de TI cedido pela empresa que está apoiando este trabalho, utiliza em grande escala, tecnologias *Microsoft*.

Para este trabalho, os servidores mantenedores dos bancos de dados, que podem ser monitorados por regras criadas e aplicadas a partir da interface do sistema, possuem as seguintes configurações:

• Um servidor **(B)** *HP ProLiant BL460c G6,* com 2 processadores *Six-Core Intel Xeon* de 2667 MHz, 72 GB de memória RAM, 1,2 TB de discos SAS em RAID 10 disponíveis em uma *storage IBM DS3512*, 2 interfaces de rede de 1Gbps cada uma, e 1 interface Fibre Channel de 8 Gbps conectando este servidor à *storage*. O SGBD instalado e configurado neste servidor é o *Microsoft SQL Server 2008 R2*. Neste servidor estão armazenados 10 bancos de dados referentes aos sistemas de ERP, Financeiro e CRM.

• Um servidor **(C)** *HP ProLiant BL460c G7,* com 2 processadores *Six-Core Intel Xeon* de 2667 MHz, 96 GB memória RAM, 1,2 TB de discos SAS em RAID 10 em *storage IBM DS3512*, 2 interfaces de rede de 1 Gbps cada uma, e 1 interface Fibre Channel de 8 Gbps

conectando este servidor à *storage*. O SGBD instalado e configurado neste servidor é o *Microsoft SQL Server 2008 R2*. Neste servidor estão armazenados 8 bancos de dados referentes a diversos sistemas como RH (Recursos Humanos), Portal de Atendimento ao Cliente, Portal de Fornecedores e Sistema de Chamados Internos.

• Um servidor **(D)** *HP ProLiant BL460c G6*, com 1 processador *Six-Core Intel Xeon* de 2667 MHz, 8 GB de memória RAM, 600 GB de disco SAS de 15 rpm em RAID 5, e 2 interfaces de rede 1 Gbps cada uma. O SGBD instalado e configurado neste servidor é o *Oracle 11g*. Neste servidor está armazenado 1 banco de dados referente ao sistema legado de controle financeiro.

A Figura 11 apresenta o ambiente disponibilizado para realizarmos os experimentos necessários. No servidor (A), estão configurados os componentes da Camada de Interface de Controle e da Camada Centralizadora. Nos servidores (B, C e D), que representam a Camada de Bancos de Dados Monitorados, estão os bancos de dados monitorados.

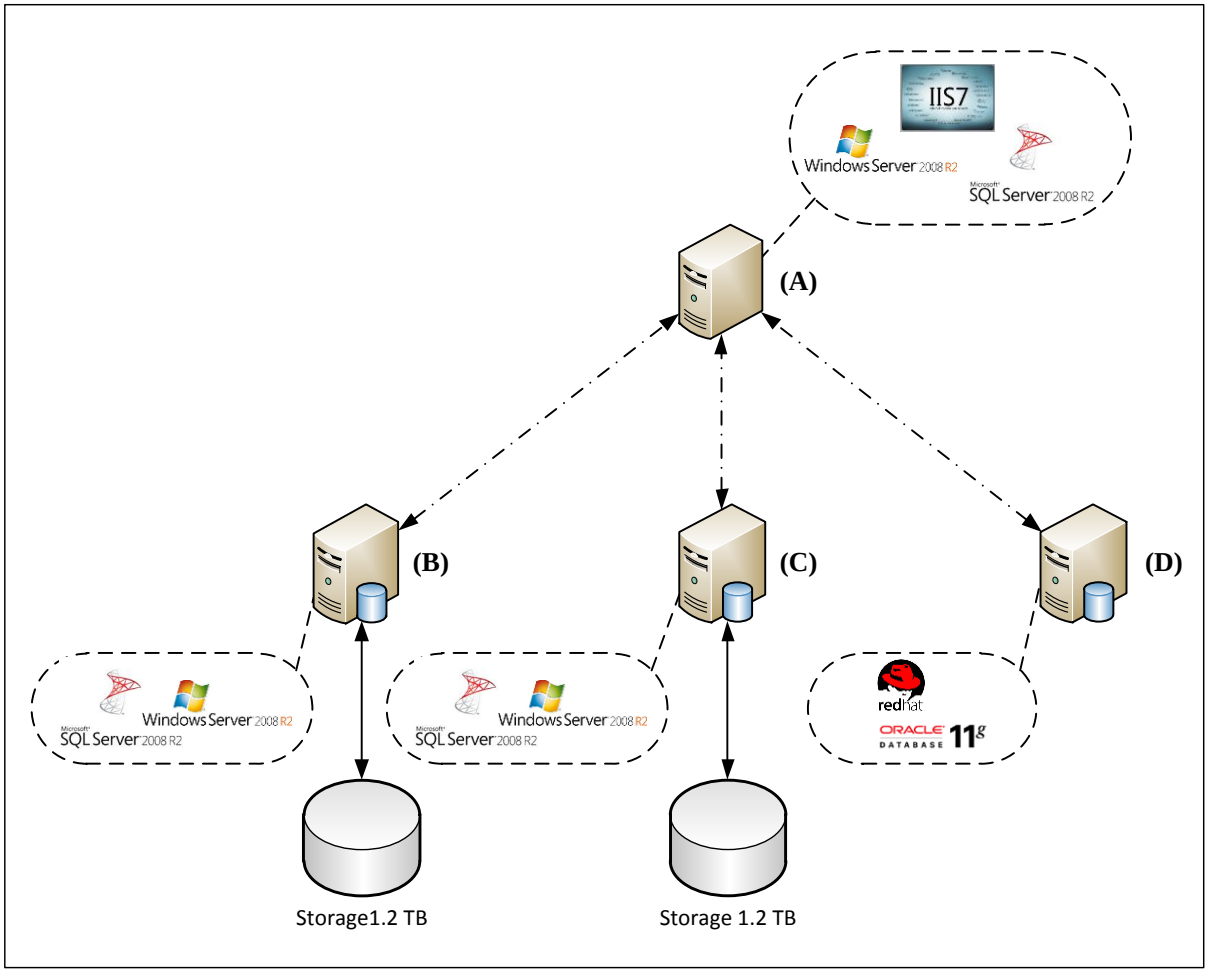

Figura 11 – Ambiente disponibilizado para experimentos pela empresa colaboradora.

5.2 LIMITAÇÕES DA IMPLEMENTAÇÃO

O presente trabalho não pretende abranger todos os SGBDs nesta implementação da arquitetura, mas a forma como foi idealizada permite que, com poucas adaptações, as regras ativas possam ser aplicadas em diferentes SGBDs que permitam o uso desse tipo de regra.

Isso fica mais claro observando-se a forma com que a regra é cadastrada no sistema, pois a arquitetura prevê o armazenamento da semântica e sintaxe do código do *trigger* de diferentes SGBDs. Então, quando o usuário precisar aplicar uma regra, ele pode recuperar a sintaxe de acordo com o SGBD desejado.

Neste trabalho, para efeito de testes, criamos regras sobre o SGBD *Microsoft SQL Server 2008 R2* por razões já mencionadas na seção 5.1. Entretanto, como a arquitetura proposta possui característica genérica e modular, ela pode atender ambientes com diversos tipos de SGBDs, como: *Oracle, Sybase, MySQL,* etc.

Também podemos listar outras limitações como: a) testes realizados somente sobre eventos DDL de *CREATE TABLE* (mas isso não significa que a arquitetura não possa ser ajustada para interagir com outros tipos de eventos DDL, como por exemplo, *ALTER PROCEDURE, ALTER INDEX*, etc.); b) testes somente sobre um algoritmo de recomendação e uma função de similaridade (mas também é possível utilizarmos outros algoritmos de recomendação e similaridade, pois como já mencionado, a arquitetura é genérica e pode ser adaptada ou modulada de acordo com outros algoritmos); c) um período curto de histórico de criações de tabelas (entretanto, isso não implica negativamente na análise dos resultados).

5.3 REPRESENTAÇÃO LÓGICA DO BANCO DE DADOS DA ARQUITETURA

A Figura 12 apresenta o modelo lógico do banco de dados utilizado pela arquitetura proposta.

As tabelas foram criadas de acordo com o desenho da arquitetura contemplando: os repositórios para armazenamento das informações referentes às regras e suas sintaxes e semânticas, que poderão ser criadas e manipuladas no sistema; usuários que interagirão com o sistema; bancos de dados monitorados; e por fim, a de log das alterações ocorridas nos bancos de dados monitorados. Estas tabelas estão incorporadas à Camada Centralizadora, sendo descritas a seguir:

As tabelas ***Regras*** e ***RegraScript*** fazem parte do Repositório de Regras, e nelas são armazenadas as informações das regras criadas no sistemas.

A tabela ***Sintaxe_Semantica*** faz parte do Repositório de Sintaxe/Semântica, e armazena as informações sobre a sintaxe e semântica das regras ativas criadas no sistema.

A tabela ***Usuario*** armazena as informações dos usuários que interagem com a interface do sistema.

A tabela ***BancoDados*** armazena as informações dos bancos de dados presentes no ambiente que o sistema pode monitorar.

E por fim, a tabela ***Alteracoes***, faz parte do Repositório de Alterações. Nela são armazenadas as informações dos eventos DDL aplicados nos bancos de dados monitorados.

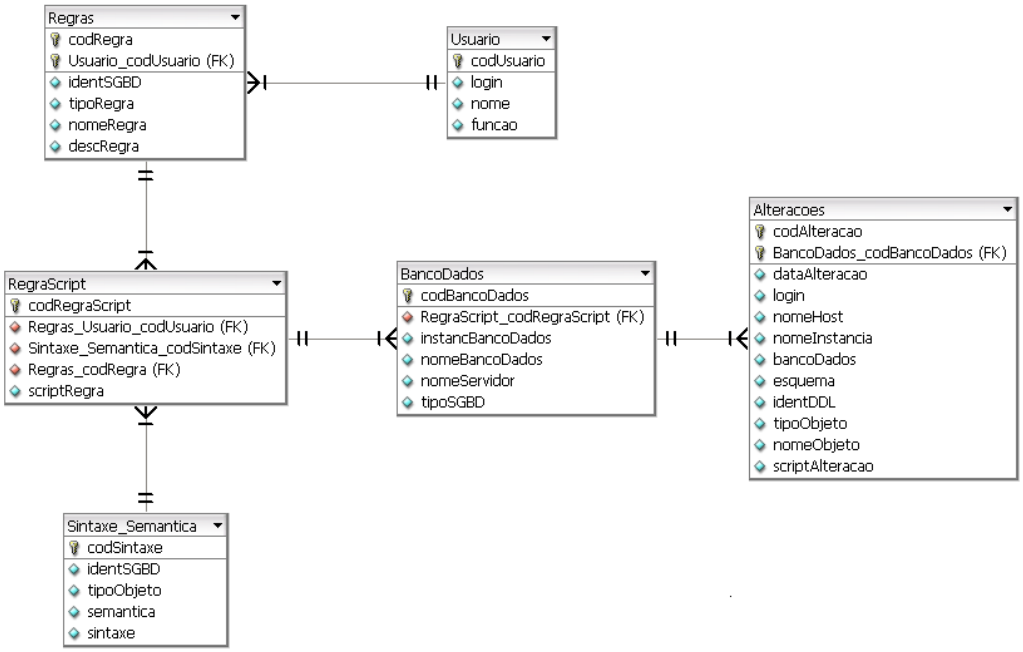

Figura 12 – Modelo Lógico do Banco de Dados da Arquitetura.

5.4 MECANISMO DE REGRAS

5.4.1 Criação de Regras

A sequência a seguir define como as regras são criadas a partir no Módulo de Regras. Considere *r_CriaTbl_Bd* uma regra definida por um usuário e armazenada no Repositório de Regras.

1) O usuário efetua o cadastro através do Módulo de Regras da interface web, definindo os parâmetros: SGBD, Tipo de Regra (Evento), Nome da Regra, Banco de Dados e Descrição da Regra.
2) O Módulo de Regras armazena *r_CriaTbl_Bd* no Repositório de Regras. Cada regra deve ter definido o tipo de SGBD correspondente.
3) Cada tipo de regra dá origem a um gatilho *(trigger)*, no qual serão especificadas as verificações das condições definidas pela regra, que por sua vez precisa estar de acordo com a sintaxe e semântica do banco de dados a ser monitorado. Estes gatilhos são configurados para serem disparados quando ocorrer um evento DDL no banco de dados onde são aplicados. A Figura 13, na subseção 5.6.1, apresenta a interface utilizada no processo descrito acima.

5.4.2 Aplicação de Regras

A sequência a seguir define como as regras são aplicadas nos bancos de dados. Considere *t_CriaTbl_Bd* um gatilho que será aplicado a partir de uma regra cadastrada.

1) Através da interface o usuário aciona o Módulo de Regras, e define o servidor e o banco de dados onde será aplicada a regra;
2) É selecionada a regra, *r_CriaTbl_Bd* é escolhida e verificada sua semântica e sintaxe;
3) É aplicada a regra *t_CriaTbl_Bd* no banco de dados definido. A Figura 14, na subseção 5.6.2, apresenta a interface utilizada no processo descrito acima.

5.5 ALGORITMO DE RECOMENDAÇÃO

5.5.1 Processo de Recomendação

Para a execução deste processo, o Mecanismo de Recomendação combina o evento DDL de *CREATE TABLE* do usuário atual, com as informações dos mesmos tipos de eventos processados por usuários de sessões anteriores, e armazenadas no Repositório de Alterações. Então, é gerada a recomendação da possível criação de tabelas, que é retornada para o usuário.

Consideramos um cenário onde os usuários criam tabelas de acordo com a necessidade de seus sistemas entre ambientes de desenvolvimento, teste integrado e homologação. O

objetivo é identificar quais tabelas podem ser sugeridas para criação, em determinados bancos de dados, de sistemas dentro destes ambientes.

5.5.2 Adaptação para SBDA e eventos DDL *(CREATE TABLE)*

Em nosso trabalho tratamos a recomendação de forma semelhante à proposta de Eirinaki et al. (2013) já descrita no capítulo 4. Entretanto, a ideia é gerar recomendações de criação de possíveis tabelas com base no histórico de tabelas criadas em diferentes bancos de dados dentro de um ambiente corporativo. Para tanto, fizemos algumas adaptações para realizar recomendações sobre eventos *CREATE TABLE* realizados em diversos bancos de dados.

O algoritmo de recomendação alterado para ser aplicado nesta arquitetura é apresentado na equação (9) e descrito a seguir:

$$S^{pred}(vusu, vinst, vsrv) = \alpha \cdot S_0 + (1 - \alpha) \cdot \sum_{i=1,\ldots,n} sim\ (S_i, S_0) \cdot S_i \qquad (9)$$

S^{pred} representa um procedimento armazenado *(stored procedure[14])*, no qual foi implementado o algoritmo de recomendação; ***vusu, vinst e vsrv*** identificam os parâmetros passados e representam respectivamente usuário, instância e servidor; **α**, no trabalho de Eirinaki et al. (2013), representa uma variável ajustável entre 0 e 1, que determina o valor de importância em relação ao evento (consulta) do usuário atual S_0 e os usuários anteriores. Para nosso cenário, esse valor foi definido como 0.5 para aplicação; S_0 representa o usuário atual; $\sum_{i=1,\ldots,n}$, é a somatória dos usuários sumarizados; e $sim\ (Si, S_0)$ é a aplicação da função de similaridade para os usuários sumarizados. A codificação completa do procedimento armazenado que implementa S^{pred} encontra-se no Anexo A deste trabalho.

Neste algoritmo fizemos duas modificações necessárias para adequarmos o processo de recomendação da criação de possíveis tabelas nos bancos de dados monitorados. Estas adequações são: *(i)* a inclusão de três parâmetros para serem considerados na execução do algoritmo, representados na equação (9) como ***vusu, vinst e vsrv*** (que representam usuário,

[14] Procedimento armazenado ou *stored procedure* no inglês, são módulos de programa armazenados pelo SGBD no servidor de banco de dados. Elmasri, R.; Navathe S. B. Sistemas de Banco de Dados. São Paulo. Pearson Education do Brasil, 2011, c. 13, p. 320.

instância e servidor respectivamente); *(ii)* a definição de **α** para 0.5, para definirmos o nível de importância do evento do usuário atual em relação aos usuários anteriores. Em nosso caso, este evento é o DDL *(CREATE TABLE)*. Um exemplo com os dados gerados a partir da execução deste algoritmo pode ser verificado na seção 5.8, onde está descrito o experimento relacionado à recomendação.

5.6 INTERFACE

Para implementação da interface, que interage com as funcionalidades desenvolvidas nesta arquitetura, utilizamos a plataforma *Microsoft .Net (Visual Studio 2012).*

5.6.1 Tela de Cadastro de Regras

A Figura 13 apresenta a interface utilizada pelo procedimento de criação e edição de regras. O campo SGBD identifica o SGBD para qual será criada a regra; o campo Evento especifica o tipo de comando DDL que será monitorado; o campo Nome identifica o nome da regra que será criada.

Na parte da semântica, os campos identificam como a regra que está sendo criada será executada. Desta forma, o campo Evento representa qual evento acionará a regra; o campo Condição representa qual será a condição de execução da regra; e o campo Ação representa a ação tomada pela regra.

Na parte do corpo do *trigger* é incluído o script de criação do *trigger*; o botão Buscar Sintaxe, busca a sintaxe relacionada ao SGBD para o qual está sendo criada a regra.

No fim da tela, o botão Buscar consulta uma regra para ser editada; e o botão Salvar grava a regra.

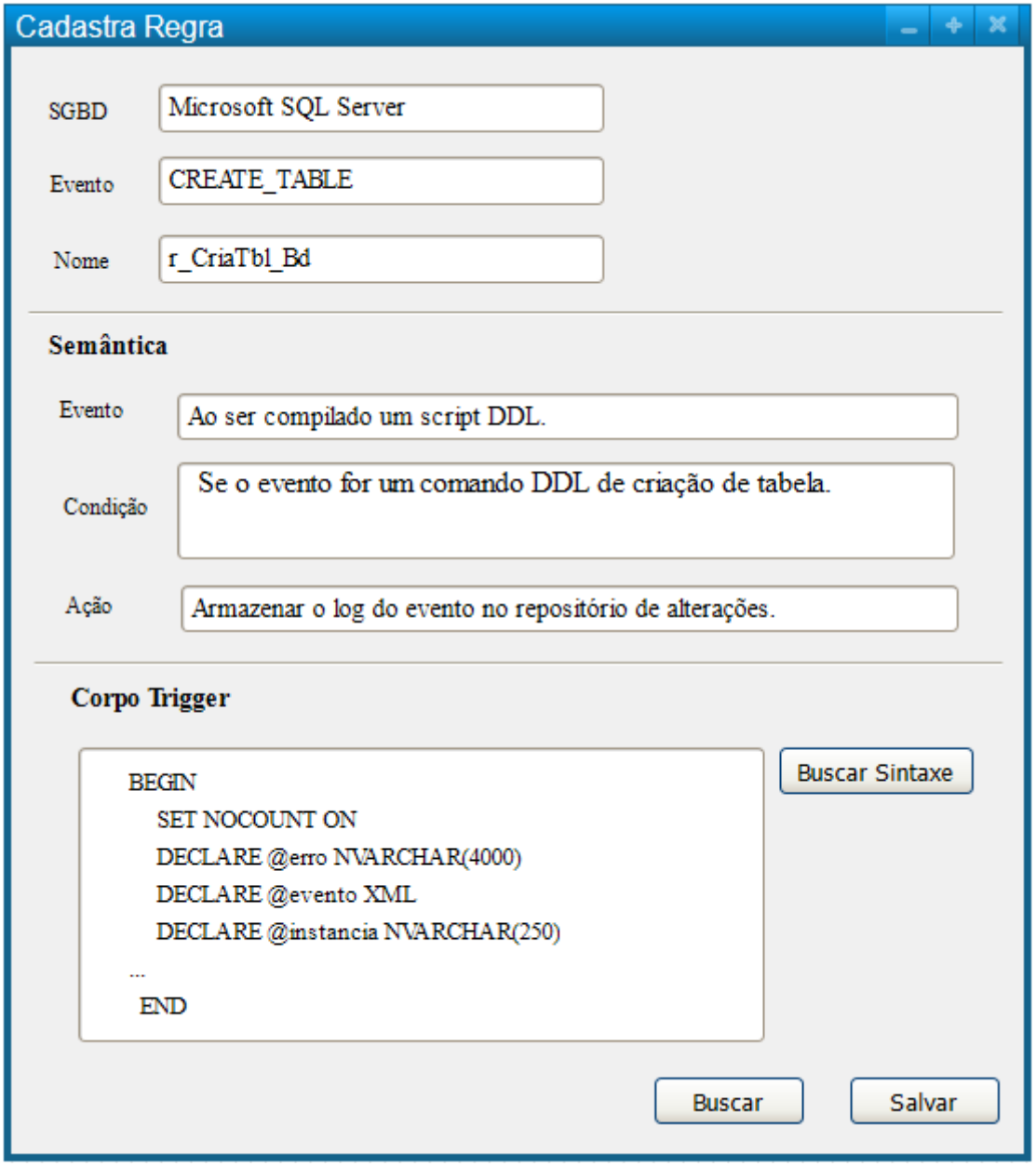

Figura 13 – Tela do sistema, representando a tarefa de cadastro de regra.

5.6.2 Tela de Aplicação de Regras

A Figura 14 apresenta a interface utilizada pelo procedimento de aplicação de regras. O campo Servidor identifica o servidor no qual será aplicada a regra; o campo Banco de Dados especifica em qual banco de dados será aplicada a regra; o campo SGBD identifica qual SGBD está associada a regra; o campo Regra identifica a regra que será aplicada; e o campo Script da Regra apresenta o script do *trigger* que será aplicado. No fim da tela, o botão Aplicar, aplica a regra no banco de dados especificado.

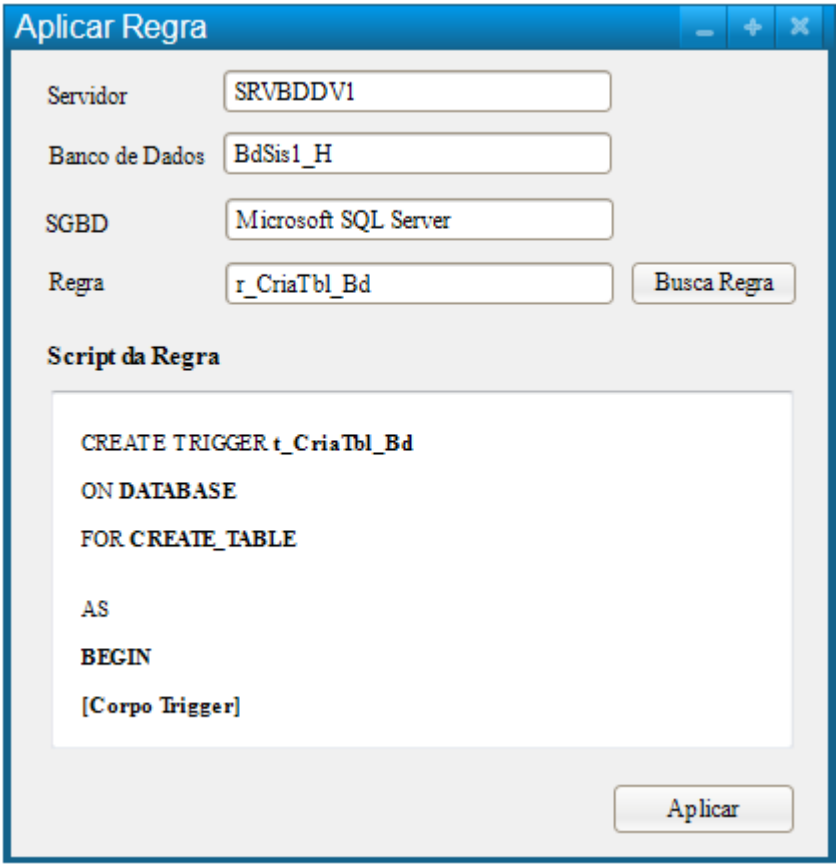

Figura 14 – Tela do sistema, representando a tarefa de aplicação de regra.

5.7 EXPERIMENTO RELACIONADO COM REGRAS

No primeiro experimento definimos uma regra DDL *(CREATE TABLE)* para o monitoramento de eventos de criação de tabelas.

Uma das razões de escolher o comando de criação de tabelas, é porque se baseia no ambiente de produção da empresa onde desenvolvemos e testamos a arquitetura proposta. Nela estavam sendo criadas algumas tabelas diretamente nos bancos de dados, sem passar pelos DBAs responsáveis por estas tarefas. O monitoramento atual também não identificava qual aplicação ou usuário estava executando estes eventos.

Neste contexto, foi criada a regra *(r_CriaTbl_Bd)* e aplicada no servidor de banco de dados do sistema de ERP *(Enterprise Resource Planning)* da empresa, onde as tabelas estavam sendo criadas. Para execução desta regra, foi definido o seguinte procedimento, adequado à semântica e sintaxe do SGBD: **(Evento)** Ao ser compilado um script DDL; **(Condição)** Se o evento for um comando DDL de criação de tabela *(CREATE TABLE)* e o nome do host for diferente do host local; **(Ação)** Armazenar o log do evento no repositório de alterações.

Com a aplicação desta regra, iniciou-se a captura dos eventos DDL *(CREATE TABLE)* disparados tanto por usuários como por aplicações externas, conforme log mostrado na Tabela 8. Os dados referentes aos campos login e nomeHost foram substituídos por motivo de segurança, mas correspondem ao monitoramento efetuado pela regra aplicada no servidor de um ambiente de produção.

dataAlteracao	login	nomeHost	identDDL	...
2014-02-20 17:15:00.823	Domínio\dba1	Desktop1	CREATE_TABLE	...
2014-02-13 04:00:01.677	Domínio\serviço	ServidorExt1	CREATE_ TABLE	...
2014-02-13 04:00:01.570	Domínio\serviço	ServidorExt1	CREATE_ TABLE	...
2014-02-20 01:51:26.557	Domínio\ dba2	Desktop2	CREATE_ TABLE	...
...	...	...	...	...

Tabela 8 – Log de eventos DDL *(CREATE TABLE)* efetuados no ambiente monitorado.

Como era preciso identificar de onde estavam partindo os eventos DDL e quem estava disparando-os, o *trigger* ficou monitorando e registrando estes eventos. Após uma semana de monitoramento foi identificado que, os eventos que estavam criando as tabelas sem autorização, era disparado por uma aplicação instalada em outro servidor (ServidorExt1) conforme destacado na Tabela 8.

O servidor no qual estava instalada a aplicação que gerava estes eventos faz parte do ambiente de homologação. Como o usuário de serviço tinha acesso à base do ambiente de produção, alguns desenvolvedores utilizaram esta "falha de segurança" para criar algumas tabelas no banco de dados de produção, para replicarem alguns dados de produção e realizarem diversas consultas. Neste caso, ao ser identificado a origem, estes acessos foram retirados.

5.8 EXPERIMENTO RELACIONADO COM RECOMENDAÇÃO

No segundo experimento referente às recomendações efetuadas pelo SR desenvolvido nesta arquitetura, executamos o algoritmo de recomendação sobre um log de alterações efetuadas no ambiente onde o sistema está instalado.

Como a regra de monitoramento de eventos DDL já estava sendo executada desde o mês de fevereiro de 2014, com aproximadamente 1103 registros, delimitamos um período de três meses, para executar o procedimento armazenado desenvolvido. Sobre este log aplicamos o algoritmo de recomendação para um usuário, instância e servidor, simulando um pedido de recomendação.

Inicialmente, separamos um período de março/2014 a maio/2014 retornando 93 ocorrências de eventos *CREATE TABLE*. A Tabela 9 apresenta os registros de criações de tabelas similares deste log. Os dados referentes à Usuário, Servidor, Instância, Banco de Dados e Nome do Objeto foram substituídos por motivos de privacidade da empresa.

Sessão	Data da Alteração	Usuário	Servidor	Instância	Banco de Dados	Nome do Objeto
19	22/03/2014	DOMINIO\usuario2	SRVBDDV1	SRVBDDV1	BdSis1	Tbl_D
19	22/03/2014	DOMINIO\usuario2	SRVBDDV1	SRVBDDV1	BdSis1	Tbl_C
21	23/03/2014	DOMINIO\usuario3	SRVBDDV1	SRVBDDV1	BdSis1_T	Tbl_C
23	23/03/2014	DOMINIO\usuario1	SRVBDDV1	SRVBDDV1	BdSis1	Tbl_B
17	24/03/2014	DOMINIO\usuario1	SRVBDDV1	SRVBDDV1	BdSis1_H	Tbl_C
22	23/03/2014	DOMINIO\usuario3	SRVBDDV1	SRVBDDV1	BdSis1_H	Tbl_B
22	23/03/2014	DOMINIO\usuario3	SRVBDDV1	SRVBDDV1	BdSis1_H	Tbl_A
20	22/03/2014	DOMINIO\usuario2	SRVBDDV1	SRVBDDV1	BdSis1_T	Tbl_D
20	22/03/2014	DOMINIO\usuario2	SRVBDDV1	SRVBDDV1	BdSis1_T	Tbl_B

Tabela 9 – Registros que tiveram equivalência no evento *CREATE TABLE*.

Note que estes registros foram extraídos do log de alterações, e representam os eventos de criação de tabelas ocorridos no servidor SRVBDDV1. Estes registros foram identificados a partir da execução do algoritmo de recomendação implementado como um procedimento armazenado de banco de dados. O código completo pode ser verificado no Anexo A deste trabalho. Executamos o algoritmo de recomendação passando os parâmetros descritos a seguir, que foram substituídos por motivo de segurança, para serem apresentados nesta dissertação.

- Usuário: “DOMINIO\usuario2”.
- Instância: “INSTSQL”.
- Servidor: “SRVBDDV1”.

Na primeira parte da execução, o algoritmo de recomendação mapeia as tabelas criadas, e sumariza os eventos de criação de tabelas do usuário atual em relação aos eventos de criação de tabelas dos usuários anteriores. Este processo resultou nas informações apresentadas na Tabela 10.

				Tabelas				
Sessão	**Usuário**	**Servidor**	**Instância**	**Tbl_A**	**Tbl_B**	**Tbl_C**	**Tbl_D**	**...**
17	DOMINIO\usuario1	SRVBDDV1	INSTSQL	0	0	1	0	...
19	DOMINIO\usuario2	SRVBDDV1	INSTSQL	0	0	1	1	...
20	DOMINIO\usuario2	SRVBDDV1	INSTSQL	0	1	0	1	...
21	DOMINIO\usuario3	SRVBDDV1	INSTSQL	0	0	1	0	...
22	DOMINIO\usuario3	SRVBDDV1	INSTSQL	1	1	0	0	...
23	DOMINIO\usuario1	SRVBDDV1	INSTSQL	0	1	0	0	...

Tabela 10 – Tabela com dados reais sumarizados na primeira etapa para o cálculo de recomendação.

Nesta tabela podemos perceber a primeira sumarização, identificando quais tabelas foram criadas pelos usuários, ou seja, o usuario1 criou as tabelas *(Tbl_B e Tbl_C)*; o usuario2 as tabelas *(Tbl_B, Tbl_C e Tbl_D)*; e o usuario3 as tabelas *(Tbl_A, Tbl_B e Tbl_C)*. Estas tabelas foram identificadas no ambiente que foi passado como parâmetro para obter a recomendação. A seguir, na Tabela 11, é apresentada a segunda parte da execução do algoritmo de recomendação. Este processo sumarizou os eventos gerando as informações apresentadas a seguir.

			Tabelas				
Usuário	**Servidor**	**Instância**	**Tbl_A**	**Tbl_B**	**Tbl_C**	**Tbl_D**	**...**
DOMINIO\usuario3	SRVBDDV1	INSTSQL	1	1	1	0	...
DOMINIO\usuario2	SRVBDDV1	INSTSQL	0	1	1	2	...
DOMINIO\usuario1	SRVBDDV1	INSTSQL	0	1	1	0	...

Tabela 11 – Tabela com dados reais sumarizados na segunda etapa para o cálculo de recomendação.

Nesta tabela são apresentados os dados referentes à segunda sumarização, que realiza a somatória dos pesos aplicados para as tabelas criadas e que possuem semelhanças, após a execução do algoritmo de recomendação com os parâmetros mencionados anteriormente, a classificação das tabelas criadas resultou nos dados apresentados na Tabela 12.

Tabela	**Peso**
Tbl_C	1.03
Tbl_B	1.03
Tbl_D	1.00
Tbl_A	0.23

Tabela 12 – Recomendação de tabelas que podem ser criadas no servidor passado como parâmetro.

No ambiente disponibilizado pela empresa cooperadora com este trabalho, existem servidores que possuem bancos de dados e seus respectivos sistemas para ambientes de teste

integrado e homologação. Com o resultado apresentado na Tabela 12, é possível identificar que as tabelas *Tbl_C* e *Tbl_B*, dos bancos de dados *BdSis1*, *BdSis1_T e BdSis1_H,* são tabelas que tiveram maior classificação na execução do algoritmo de recomendação, ou seja, estas tabelas foram criadas mais vezes no servidor analisado, e podem ser sugeridas como tabelas candidatas a serem criadas em um banco de dados do mesmo sistema em outro servidor.

Por exemplo, o banco de dados *BdSis1_H* é um banco de homologação do *Sis1*, o *BdSis1_T* é de teste integrado e o *BdSis1* é de desenvolvimento. Ao instalarmos o *Sis1* em um ambiente de pré-produção, com um novo banco de dados *BdSis1_PP*, e com suas tabelas padrões, podemos criar também as tabelas recomendadas. Pois para os bancos de dados do sistema *Sis1*, no ambiente monitorado, estas tabelas também foram criadas.

5.9 CONTRIBUIÇÕES

A arquitetura proposta neste trabalho é a primeira contribuição destacada, pois através do desenho idealizado e descrito na seção 4.1, na Figura 6, foi possível dar sequência no desenvolvimento dessa ideia e de todos os mecanismos descritos neste trabalho.

Ficou também evidente a possibilidade de integrar mecanismos de SBDA para atuação sobre eventos DDL em diferentes SGBDs, assim como a integração de SBDAs com SRs, demonstrando que as técnicas de recomendações utilizadas em grande escala nos sites comerciais, também podem ser aplicadas para ajudar em determinadas tarefas sobre os SGBDs e SBDAs.

A adaptação descrita na seção 5.5.2, nas páginas 46 e 47, do algoritmo proposto por Eirinaki et al. (2013), é mais uma contribuição deste trabalho. O algoritmo de Eirinaki et al. (2013), foi idealizado para trabalhar com recomendações a partir de um log de registros referentes às consultas efetuadas em um banco de dados. Em nosso cenário, foram necessários alguns ajustes, para que o algoritmo pudesse trabalhar sobre um log de registros referentes a eventos DDL (alterações efetuadas na estrutura de bancos de dados). O que é diferente do escopo apresentado por Eirinaki et al. (2013).

Outra contribuição deste trabalho foi a implementação do protótipo, para possibilitar a execução dos testes e análise dos resultados a partir das funcionalidades da arquitetura proposta.

5.10 CONSIDERAÇÕES FINAIS

Neste capítulo foram apresentados os experimentos realizados, envolvendo a criação de uma regra e o algoritmo de recomendação na arquitetura proposta. No primeiro experimento é descrita a parte da arquitetura que envolve o monitoramento de eventos DDL integrado a SBDA, e no segundo experimento é descrita a parte que envolve SR integrado à SBDA.

No primeiro experimento criamos uma regra ativa com a finalidade descrita na seção 5.7, e a aplicamos no servidor que precisava ter um banco de dados monitorado, para identificar o sistema ou usuário que estava aplicando eventos DDL. Após registrar algumas informações de eventos DDL ocorridos neste banco de dados, conseguimos identificar o usuário e o local de origem da execução destes eventos, conforme pode ser verificado na Tabela 8.

Para o segundo experimento, nos foi disponibilizado um ambiente com servidores e diversos bancos de dados, como: de sistema financeiro, compras, recursos humanos, etc. Tais bancos de dados pertecem aos ambientes de desenvolvimento, teste integrado e homologação de uma empresa, que os disponibilizou para a validação dos experimentos. Estes bancos de dados possuem um padrão de nomenclatura *(PrefixoNomeSistema_LetraAmbiente)*. Por exemplo, um banco de dados do sistema *XPO*, do ambiente de homologação, receberia o nome *BdXpo_H*.

A mesma arquitetura pode ser aplicada em empresas de TI que administram diferentes ambientes de bancos de dados de diversos clientes. Pode ser adaptado o tipo de monitoramento de eventos DDL assim como, as recomendações que podem ser feitas sobre os bancos de dados monitorados.

6 CONCLUSÃO E TRABALHOS FUTUROS

Este capítulo finaliza o presente trabalho descrevendo a conclusão e sua relevância, bem como apresenta sugestões de possíveis trabalhos futuros.

6.1 CONCLUSÃO

Diversos trabalhos alusivos à integração de monitoramento de eventos em bancos de dados, sistemas de recomendação e sistema de bancos de dados ativos, foram apresentados nesta dissertação. Estes trabalhos apresentam propostas de arquiteturas e modelos conceituais aplicados em diversos cenários.

Entretanto, mesmo que desenvolvidos diversos trabalhos solucionando problemas complexos deste contexto, ainda fica uma lacuna abordada neste trabalho, que é o monitoramento de eventos DDL integrado a SBDA, e também a integração de SRs com SBDA, para recomendação da possível criação de tabelas em bancos de dados.

Nesta dissertação foi apresentada uma arquitetura combinando aspectos teóricos e práticos de: 1) Sistemas Gerenciadores de Bancos de Dados; 2) Sistemas de Bancos de Dados Ativos; e 3) Sistemas de Recomendação. Ao combinar conceitos destes sistemas, desenvolvemos: 1) uma arquitetura que integra banco de dados ativos e sistema de recomendação; 2) adaptamos um algoritmo de recomendação para recomendar a criação de possíveis tabelas no ambiente monitorado; e 3) validamos estes conceitos a partir de um protótipo em um ambiente corporativo.

Com base nos experimentos realizados e aqui apresentados, foi possível verificar que a arquitetura proposta permite monitorar eventos DDL a partir da criação e aplicação de regras ativas nos bancos de dados. Também foi possível verificar a integração de SR com SBDA.

A característica mais expressiva da arquitetura proposta é a possibilidade de incluir SGBDs diferentes para serem monitorados pelo sistema, uma vez que é possível incluir a sintaxe específica do SGBD no sistema, e a partir desta, o próprio usuário pode criar a regra para o fim desejado.

Nos experimentos, para monitoramento dos bancos de dados, foram considerados eventos DDL de *CREATE TABLE* na criação das regras. Para o SR realizar as recomendações, também foram considerados estes tipos de eventos. Entretanto, a arquitetura é flexível e permite a adaptação ou ampliação de seus mecanismos, para criar diferentes regras de

monitoramento e gerar recomendações sobre outros tipos de eventos, como por exemplo, eventos DML.

A arquitetura proposta neste trabalho não abrange o monitoramento de todos os eventos DDL, e o SR também não faz recomendação da criação de todos os tipos de objetos do banco de dados, mas demonstra que, a partir dos mecanismos aqui desenvolvidos, é perfeitamente possível estender aos demais eventos DDL e também recomendar outras criações ou até mesmo alterações de objetos de bancos de dados.

Em resumo, as principais contribuições deste trabalho são: 1) a idealização e desenvolvimento da arquitetura proposta; 2) a integração de mecanismos de SBDA, para monitoramento de eventos DDL de diferentes SGBDs; 3) a integração de SBDA com SR; 4) a adaptação de um algoritmo de recomendação, para recomendar a criação de possíveis tabelas em bancos de dados; e 5) a implementação de um protótipo, para possibilitar a execução dos testes e análise dos resultados a partir das funcionalidades da arquitetura proposta.

6.2 TRABALHOS FUTUROS

Como trabalhos futuros, podemos sugerir a recomendação sobre outros tipos de eventos DDL ou até mesmo detalhar o próprio evento *CREATE*, podendo estender o algoritmo de recomendação para outros eventos DDL.

Um exemplo a ser explorado é a recomendação baseada em fragmentos (AKBARNEJAD et al., 2010). Esta possibilidade incluiria a identificação de particularidades dentro do script do evento. Por exemplo, dentro de um comando DDL de *ALTER TABLE*, analisar também os fragmentos deste evento como: qual coluna foi alterada ou adicionada na tabela alterada.

Muitas empresas de TI possuem diversos clientes, para os quais prestam serviços de administração de seus ambientes de bancos de dados, e nestes ambientes podem estar presentes diferentes SGBDs. Esta arquitetura pode ser adaptada para o monitoramento e identificação dos objetos criados nestes bancos de dados, e de acordo com a análise dos fragmentos do evento, recomendar alterações nestes objetos.

REFERÊNCIAS

ADOMAVICIUS, G.; TUZHILIN, A. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. IEEE Transactions on Knowledge and Data Engineering, New York, v. 17, n. 6, p.734-749, 2005.

AKBARNEJAD, J.; EIRINAKI, M.; KOSHY, S.; POLYZOTIS, N. SQL QueRIE Recommendations: A Query Fragment-Based Approach. PersDB'10 4th International Workshop on Personalized Access, Profile Management, and Context Awareness in Databases, Singapore, 2010.

ANSARI, A.; ESSEGAIER, S.; KOHLI, R. Internet Recommendation Systems. Journal of Marketing Research, v.37, n.3, p. 363-375, 2000.

BADIA, A. Active Database Systems for Monitoring and Surveillance. Proceedings of the 1st NSF/NIJ Conference on Intelligence and Security Informatics, Berlin, p. 296-307, 2003.

CAMPIN, J.; PATON, N.; AND WILLIAMS, M. Specifying Active Database Systems in an Object-Oriented Framework. International Journal of Software Engineering and Knowledge Engineering, p. 101–123, 1997.

CARDOSO, V. M. Uma Ferramenta para Teste Estrutural de Regras Ativas. 2004. 171 p. Dissertação (Mestrado em Engenharia Elétrica) – Universidade Estadual de Campinas, São Paulo.

CAZELLA, S. C.; REATEGUI, E.; MACHADO, M.; BARBOSA, J. Recomendação de Objetos de Aprendizagem Empregando Filtragem Colaborativa e Competências. In: Simpósio Brasileiro de Informática na Educação (SBIE), 2009.

CAZELLA, S. C.; REATEGUI, E. B. Sistemas de Recomendação. Em XXV Congresso da Sociedade Brasileira de Computação, São Leopoldo – RS, 2005.

CAZELLA, S. C.; NUNES, M. A. S. N.; REATEGUI, E. B. A Ciência da Opinião: Estado da arte em Sistemas de Recomendação. XXX Congresso da SBC, Belo Horizonte – MG, 2010.

CHAKRAVARTHY, S.; ANWAR, E.; MAUGIS, L.: Design and Implementation of Active Capability for an Object-Oriented Database. Technical Report UF-CIS TR-93-1, CIS Department, University of Florida, 1993.

CILIA, M. A. Bancos de Dados Ativos. Biblioteca Digital da UNICAMP, Campinas, 1996.

EIRINAKI, M.; ABRAHAM S.; POLYZOTIS, N.; SHAIKH, N. QueRIE: Collaborative Database Exploration. IEEE Transactions on Knowledge and Data Engineering, p.1-14, 2013.

ELMASRI, R.; NAVATHE S. B. Sistemas de Banco de Dados. São Paulo. Pearson Education do Brasil, c. 26, p. 625-635, 2011.

HAMID; REZA, F.; AHMAD, H. N. Rule Scheduling Methods in Active Database Systems: A Brief Survey. The 6th International Conference on Application of Information and Communication Technologies, Georgia, Tbilisi, 2012.

HERLOCKER, J. L. Explaining Collaborative Filtering Recommendations. CSCW Computer Supported Cooperative Work, v. 1, p. 241-250, 2000.

HERLOCKER, J. L.; KONSTAN, J. A.; BORCHERS, A.; RIEDL, J. An Algorithmic Framework for Performing Collaborative Filtering. In Proceedings of the 22nd International Conference on Research and Development in Information Retrieval. ACM, New York, p. 230–237, 1999.

HERLOCKER, J.; KONSTAN, J.; TERVEEN, L.; RIEDL, J. Evaluating Collaborative Filtering Recommender Systems. In ACM Transactions on Information Systems, v. 22, p. 5-53, 2004.

HUANG, K. The Research for Embedded Active Database Based on ECA Rule and Implementation in SQLite Database. Database Technology and Applications, First International Workshop, Wuhan, p. 476-479, 2009.

JIN, Y.; URBAN S. D.; DIETRICH S. W. A Concurrent Rule Scheduling Algorithm for Active Rules. Data and Knowledge Engineering, Elsevier Science, v.60, p. 530-546, 2007.

JIN, Y. Management of Composite Events for Active Database Rule Scheduling. Information Reuse & Integration, 2009. IRI '09. IEEE International Conference, p. 300-304, 2009.

JOSKO, J. M. B. Bancos de Dados Ativos: Conceitos, Aplicações e Limitações. Rio de Janeiro, SQL Magazine, ed. 94, p. 18-24, 2011.

JUN, C. Active Web Database Technology and Its Applications in CRM. Environmental Science and Information Application and Technology (ESIAT), International Conference, Wuhan, v. 3, p. 676-678, 2010.

KANAGAL, B.; AHMED, A.; PANDEY, S.; JOSIFOVSKI, V.; YUAN, J.; and PUEYO L. G. Supercharging Recommender Systems using Taxonomies for Learning User Purchase Behavior. PersDB’12 6th International Workshop on Personalized Access, Profile Management, and Context Awareness in Databases, p. 956–967, Istanbul, 2012.

KORTH, H. F.; SUDARSHAN, S; SILBERSCHATZ, A. Sistema de Banco de Dados. Rio de Janeiro, Elsevier Editora Ltda, c. 1, p. 3-7, 2012.

KONSTAN, J. A.; MILLER, B. N.; MALTZ, D.; HERLOCKER, J. L.; GORDON, L. R.; RIEDL, J. GroupLens: Applying Collaborative Filtering to Usenet News. p. 77-87, 1997.

LOH, S.; LICHTNOW, D.; GARIN, R. S. Arquitetura de Um Sistema de Recomendação para Apoio à Colaboração. VIII Congreso Argentino de Ciencias de la Computación. 2002.

LU, W. Audit Guard: A System for Database Auditing Under Retention Restrictions. Proceedings of the VLDB Endowment, v. 1, p. 1484-1487, 2008.

MEDINA-MARIN, J.; PEREZ-LECHUGA, G.; LI, X. ECA Rule Analysis in a Distributed Active Database. Computer Technology and Development, ICCTD. International Conference, Kota Kinabalu, v. 2, p. 113-116, 2009.

MONTESI, D.; TORLONE, R.A Transaction Transformation Approach to Active Rule Processing. In Proc. of the Eleventh International Conference on Data Engineering, Taipei, Taiwan, 1995.

MORGENSTERN, M. Active Databases as a Paradigm for Enhanced Computing Environments. VLDB'83 Proceedings of the 9th International Conference on Very Large Data Bases, p. 34-42, 1983.

NORMAN, W. P. Active Rules in Database Systems. ACM Computing Surveys, v.31, p. 63-103, New York, USA, 1999.

OLIVEIRA, V. S., Spinola, R. O. Obtendo bancos de dados seguros. Rio de Janeiro, SQL Magazine, ed. 108, p. 63-66, 2013.

PATON, N. W.; DIAZ, O. Active Database Systems. Journal ACM Computing Surveys, v. 31, p.63-103, New York, 1999.

PIMENTEL, M.; FUKS, H. Sistemas Colaborativos. Rio de Janeiro: Elsevier Editora Ltda, 2012.

QIAO, Y.; ZHONG K.; WANG, H.; LI, X. Developing Event-Condition-Action Rules in Real-Time Active Database. ACM Symposium on Applied Computing, p. 511-516, New York, USA, 2007.

RABUZIN, K. Simulating Proactive Behaviour in Active Databases. 5th International Symposium, p. 25-29, Floriana, 2011.

RESNICK, P.; IACOVOU, N.; SUCHAK, M.; BERGSTROM, P.; RIEDL, J. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In Conference on Computer Supported Collaborative Work, Eds. ACM Press, New York, p 175-186, 1994.

RESNICK, P., VARIAN, H. R. Recommender Systems. Magazine Communications of the ACM, v. 40, p. 56-58, 1997.

ROB, P.; CORONEL, C. Sistemas de Banco de Dados – Projeto, Implementação e Administração. São Paulo: Cengage Learning, c. 1, p. 6-9, 2011.

SARWAT, M.; AVERY, J.; MOKBEL, M. F. RecDB in Action: Recommendation Made Easy in Relational Databases. Very Large Data Base Endowment Inc., v. 6, p. 1242-1245, 2013.

SCHAFER, J. B.; KONSTAN, J. A.; RIEDL, J. E-Commerce recommendation Applications. Data Mining and Knowledge Discovery, Vol. 5, Nº 1-2. p. 115-153, 2001.

SIMON, F.; SANTOS A. L.; HARA C. S; Um Sistema de Auditoria baseado na Análise de Registros de Log. IV Escola Regional de Banco de Dados – ERBD 2008, Universidade Federal de Santa Catarina – UFSC, 2008.

SHARDANAND, U. AND MAES, P. Social Information Filtering: Algorithms for Automating Word of Mouth, In Human Factors in Computing Systems. Denver, Colorado, USA, 1995.

SMEATON, A. F.; CALLAN, J. Personalisation and Recommender Systems in Digital Libraries. International Journal on Digital Libraries, v. 5, p. 299-308, 2005.

TATA, S.; PATEL, J. M. Estimating the Selectivity of Tf-Idf Based Cosine Similarity Predicates. 26th ACM SIGMOD/PODS - International Conference on Management of Data, New York, v. 36, p. 7-12, 2007.

TELNAROVÁ, Z. Adding Rules into Database Systems. IEEE Explore Digital Library, Federated Conference, Wroclaw, p. 155-159, 2012.

TORRES, R. Personalização na Internet. São Paulo: Editora Novatec, 2004.

VIEIRA, F. J. R., NUNES, M. A. S. DICA: Sistema de Recomendação de Objetos de Aprendizagem Baseado em Conteúdo. São Cristovão, Scientia Plena, v. 8, n. 5, 2012.

WATSON, J.; RAMKLASS, R. Oracle Database 11g Fundamentals SQL. Rio de Janeiro: Alta Books, 2010.

ZANIOLO, C.; CERI, S.; FALOUTSOS, C.; SNODGRASS, R.; SUBRAHMANIAN, V. S.; ZICARI, R. Advanced Database Systems. Morgan Kaufmann Publishers, 1997.

Anexo A – Código do procedimento armazenado do algoritmo de recomendação

```
CREATE PROCEDURE [dbo].[sp_spred]
      @usuario varchar(100),
      @instancia varchar(100),
      @servidor varchar(100)
AS
BEGIN

      SET NOCOUNT ON;

--Variaveis temporarias para sumarizacao
declare @objeto_tmp varchar(100),
            @sessao_tmp int,
            @S_tmp varchar(100),
            @servidor_tmp varchar(100),
            @instancia_tmp varchar(100),
            @usuario_tmp varchar(100),
            @totalObjetos int

-- Passo 1 - Resumo da sessao do usuario atual
      select distinct sessao
      into #sessoesUsuario
      from RepAlteracoesNova2
      where login = @usuario
                  and nomeHost = @servidor
                  and nomeInstancia = @instancia

-- Passo 2 - Localizar objetos criados por essas sessoes
      select nomeObjeto
      into #objetosUsuario
      from RepAlteracoesNova2
      where login = @usuario
                  and nomeHost = @servidor
                  and nomeInstancia = @instancia
                  and sessao in (select sessao from #sessoesUsuario)

-- Passo 3 - Localizar outras sessoes que criaram os mesmos objetos
      select      sessao,
                  login,
                  nomeHost,
                  nomeInstancia
      into #sessoesSimilares
      from RepAlteracoesNova2
      where nomeObjeto in
      (
            select nomeObjeto from #objetosUsuario
      )
      and sessao not in (select sessao from #sessoesUsuario)
      and login <> @usuario
      and nomeHost = @servidor
      and nomeInstancia = @instancia

--select * from #sessoesSimilares

create table #sumarios (
```

```
	usuario varchar(100),
	servidor varchar(100),
	instancia varchar(100),
	nomeObjeto varchar(300),
	sessao int,
	SQ bit
)

create table #sumarios2
(
	sessao varchar(100),
	usuario varchar(100),
	servidor varchar(100),
	instancia varchar(100),
	S varchar(100)
)

-- Passo 4 - Localizar os outros objetos que as sessoes do select 2 tambem
criaram
	DECLARE CURSOROBJETOS SCROLL CURSOR FOR
		select distinct nomeObjeto
		from RepAlteracoesNova2
		where sessao in (
			select sessao from #sessoesSimilares
		)
		and login <> @usuario
		and nomeHost = @servidor
		and nomeInstancia = @instancia

	union
		select nomeObjeto
		from RepAlteracoesNova2
		where login = @usuario
		and nomeHost = @servidor
		and nomeInstancia = @instancia
		and sessao in (select sessao from #sessoesUsuario)
		order by nomeObjeto


	insert into #sessoesSimilares
	select sessao, @usuario, @servidor, @instancia from #sessoesUsuario

	declare CURSORSESSOES cursor for
	select		sessao,
			login,
			nomeHost,
			nomeInstancia
	from #sessoesSimilares

	--select * from #sessoesSimilares

	open CURSOROBJETOS
	open CURSORSESSOES

--Itera pelas sessoes
	FETCH NEXT FROM CURSORSESSOES INTO @sessao_tmp, @usuario_tmp,
@servidor_tmp, @instancia_tmp

	WHILE @@FETCH_STATUS = 0
```

```
	BEGIN

		set @S_tmp = ''

		--select @sessao_tmp
		FETCH FIRST FROM CURSOROBJETOS INTO @objeto_tmp
		WHILE @@FETCH_STATUS = 0
		BEGIN

			insert into #sumarios
			select	@usuario_tmp,
				@servidor_tmp,
				@instancia_tmp,

				@objeto_tmp,
				@sessao_tmp,
				COUNT(*)
			from RepAlteracoesNova2
			where sessao = @sessao_tmp
			and nomeObjeto = @objeto_tmp
			and nomeHost = @servidor
			and nomeInstancia = @instancia

			set @S_tmp = @S_tmp + (
				select
					CONVERT(varchar, COUNT(*))-- + ','
				from RepAlteracoesNova2
				where sessao = @sessao_tmp
					and nomeObjeto = @objeto_tmp
					and nomeHost = @servidor
					and nomeInstancia = @instancia
			)

			FETCH NEXT FROM CURSOROBJETOS INTO @objeto_tmp
		END

		insert into #sumarios2
		select @sessao_tmp, @usuario_tmp, @servidor_tmp,
@instancia_tmp, @S_tmp

		FETCH NEXT FROM CURSORSESSOES INTO @sessao_tmp, @usuario_tmp,
@servidor_tmp, @instancia_tmp
	END

		CLOSE CURSORSESSOES;
		DEALLOCATE CURSORSESSOES;

		CLOSE CUROROBJETOS;
		DEALLOCATE CUROROBJETOS;

set @S_tmp = ''

/**Exibicao dos resultados*/
		select * from #sumarios  order by sessao
		select * from #sumarios2 order by 1

		--Verificar o total de objetos
		select @totalObjetos = count(distinct nomeObjeto) from
#sumarios
```

```
		select
			usuario,
			convert(varchar, sum(convert(DECIMAL(20,0),S))) Cont
		into #sumarizacao
		from #sumarios2
		group by usuario


		select		usuario,
				REPLICATE('0', @totalObjetos - len(Cont)) + Cont S

		from #sumarizacao


/**Calcular a soma da sumarizacao*/
		create table #soma
		(
			i int,
			x decimal(10,2)
		)

		declare @sumarioUsuario varchar(100),
				@soma_tmp decimal(10,2),
				@cos_tmp decimal(10,2),
				@i int


		select @sumarioUsuario = Cont
		from #sumarizacao
		where usuario = @usuario

		declare cursorSumarios cursor for
		select usuario, REPLICATE('0', @totalObjetos - len(Cont)) +
Cont
		from #sumarizacao where usuario <> @usuario

		SET @sumarioUsuario = REPLICATE('0', @totalObjetos -
len(@sumarioUsuario)) + @sumarioUsuario --DIFF

		open cursorSumarios

		FETCH NEXT FROM cursorSumarios INTO @usuario_tmp, @S_tmp
		WHILE @@FETCH_STATUS = 0
		BEGIN
			set @i = 0

			--Aplicando funcao de similariade de cosseno
			select @cos_tmp = dbo.fc_sim_cossV1(@sumarioUsuario,
@S_tmp)
			--select @cos_tmp as cosseno

			while @i < @totalObjetos
			begin

				if(select count(*) from #soma where i = @i) = 0
				begin
					insert into #soma
					select @i, 0
```

```
                        end

                        set @soma_tmp = (select x from #soma where i = @i)
                        set @soma_tmp = @soma_tmp + ( CONVERT(decimal,
SUBSTRING(@S_tmp, @i+1, 1)) * @cos_tmp)

                        update #soma set x = @soma_tmp where i = @i
                        set @i = @i + 1
                  end

                  --select * from #soma
                  FETCH NEXT FROM cursorSumarios INTO @usuario_tmp, @S_tmp
            END
            CLOSE cursorSumarios;
            DEALLOCATE cursorSumarios;


/**Calcular Spred*/
      create table #spred
      (
            i int,
            classificacao decimal(10,2)
      )

      declare @spred decimal(10,2)

      --select * from #soma
      update #soma set x = x/2

      --select * from #soma

      select @sumarioUsuario = REPLICATE('0', @totalObjetos -
len(@sumarioUsuario)) + @sumarioUsuario

      set @i = 0
      while @i < @totalObjetos
            begin

                  set @soma_tmp = CONVERT(decimal,
SUBSTRING(@sumarioUsuario, @i+1, 1)) / 2

                  insert into #spred
                  select @i, @soma_tmp + (select x from #soma where i = @i)

                  set @i = @i + 1
            end

            select * from #sumarios

            select classificacao from #spred order by classificacao desc

      END
GO
```

Anexo B – Código da função de similaridade de cosseno

```
CREATEfunction [dbo].[fc_sim_cossV1]
      (@vet1 nvarchar(200),@vet2 nvarchar(200))RETURNSfloat
AS
begin
      --Variaveis temporarias para colocar o vetor em uma tabela
      declare
                  @vetS nvarchar(200),
                  @vetQ1 nvarchar(200),
                  @vetQ2 nvarchar(200),
                  @tam_vet int,
                  @tam_vet2 int,
                  @i int,
                  @v1 int,
                  @v2 int,
                  @rs int,
                  @soma int,
                  @qd int,
                  @somaqd1 int,
                  @somaqd2 int,
                  @raiz1 float,
                  @raiz2 float,
                  @rsraiz float,
                  @resultado float
            begin
            set @i=1
            set @soma=0
            set @somaqd1=0
            set @somaqd2=0
            set @vetS=''
            set @vetQ1=''
            set @vetQ2=''
                  set @tam_vet =len(@vet1)
                  while @i <= @tam_vet
                  begin
                        set @v1 =substring(@vet1,@i,1)
                        set @v2 =substring(@vet2,@i,1)
                        set @rs = @v1 * @v2
                        set @soma = @soma + @rs
                        set @vetS = @vetS +convert(varchar,@rs)+';'
                        set @qd =SQUARE(@v1);
                        set @somaqd1 = @somaqd1 + @qd;
                        set @vetQ1 = @vetQ1 +convert(varchar,@qd)+';'
                        set @qd =SQUARE(@v2);
                        set @somaqd2 = @somaqd2 + @qd;
                        set @vetQ2 = @vetQ2 +convert(varchar,@qd)+';'
                        set @i = @i + 1
                  end
            end
            set @raiz1 =SQRT(@somaqd1);
            set @raiz2 =SQRT(@somaqd2);
            set @rsraiz = @raiz1 * @raiz2;
            if (@rsraiz = 0)
            set @rsraiz = 1
            set @resultado = @soma / @rsraiz
            returnround(@resultado,4)
end
```

Anexo C – Função de similaridade de cosseno

$$cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}} \quad \textbf{(12)}$$

Sendo que:

A é um vetor com uma cadeia de caracteres.

B é um vetor com outra cadeia de caracteres.

||A|| representa a raiz quadrada da soma dos valores do vetor A.

||B|| representa a raiz quadrada da soma dos valores do vetor B.

A*i* representa o vetor A e *i* representa suas posições.

B*i* representa o vetor B e *i* representa suas posições.

Note um exemplo em relação a equação (12):

A = [1, 1, 1, 0]

B = [0, 1, 1, 2]

A . B = 1 * 0 + 1 * 1 + 1 * 1 + 0 * 2 = 2

$||A|| = \sqrt{1^2 + 1^2 + 1^2 + 0^2} = 1{,}73$

$||B|| = \sqrt{0^2 + 1^2 + 1^2 + 2^2} = 2{,}44$

$$cos(\theta) = \frac{2}{1{,}73 \text{ x } 2{,}44}$$

$cos(\theta) = 0{,}47$