Unioeste - Universidade Estadual do Oeste do Paraná
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
Colegiado de Informática
Curso de Bacharelado em Informática

Extensões de Modelos de Dados para Aplicações Avançadas

Lucas da Silva Grando e Hudson João Magalhães

CASCAVEL
2009

Lucas da Silva Grando e Hudson João Magalhães

Extensões de Modelos de Dados para aplicações Avançadas

Trabalho apresentado como requisito parcial para obtenção de média na disciplina de Banco de Dados I, do Curso de Bacharel em Informática do Centro de Ciências Exatas e Tecnológicas da Universidade Estadual do Oeste do Paraná - Campus de Cascavel

Orientador: Prof. Carlos José Maria Olguín

CASCAVEL

2009

Sumário

Resumo

Atualmente com a crescente difusão do uso de computadores nos mais diversos segmentos da sociedade ouve-se cada vez mais o termo banco de dados. Banco de dados caracteriza-se por um conjunto de dados armazenados sendo que cada um deles possuem um significado explícito. Um banco de dados pode ser classificado sobre várias características e/ou conceitos. Este visa apresentar os conceitos de bancos de dados ativos, temporal, multimídia e dedutivo.

.

Palavras-chave: Banco de dados, ativos, temporal, multimídia, dedutivo.

Capítulo 1

Introdução

Introduziremos os conceitos de banco de dados em aplicações avançadas, e que começaram a apresentar uso difundido. As características que cobriremos são as regras ativas. Essas regras podem ser ativadas automaticamente pela ocorrência de algum evento, como atualização no banco de dados, ou certo horário alcançando, e podem disparar certas ações que foram especificadas na declaração da regra em dada condição. Muitos pacotes comerciais já fornecem alguma funcionalidade de bancos de dados ativos na forma de gatilhos (triggers), conceitos de temporalidade, usados em aplicações de banco de dados temporais que permitem que o sistema de banco de dados armazene um histórico de alterações, possibilitando aos usuários o exame dos estados atuais e passados do banco de dados, alguns modelos também permitem o armazenamento do futuro esperado da informação, como horários planejados, e rapidamente, alguns assuntos que envolvem bancos de dados multimídias que oferecem facilidades para o armazenamento e consulta de diferentes tipos de informações multimídias, incluindo imagens, clipes de vídeo, audioclipes e documentos. Também discutiremos banco de dados dedutivos é aquele que possui capacidade para definir regras que podem deduzir ou inferir informação adicional dos fatos que são armazenados em um banco de dados.

Capítulo 2

Banco de Dados Ativos

Os bancos de dados ativos são sistemas de banco de dados estendidos com um sistema de regras. Este sistema é capaz de reconhecer eventos, ativar as regras correspondentes e quando a condição é verdadeira executa as ações que correspondam. Este sistemas podem ser usados para aplicações financeiras, aplicações multimídia, controle da produção industrial, monitoramento (controle de tráfego aéreo, etc), entre outros. Também são usados para funções do próprio núcleo do banco de dados, como por exemplo, manutenção de consistência, manutenção de visões, controle de acesso, gerenciamento de versões, entre outras.

Um banco de dados é *passivo* quando não oferece suporte para o gerenciamento automático de condições definidas sobre o estado do banco de dados em resposta a estímulos externos. Os sistemas de gerenciamento de banco de dados (SGBDs) convencionais são passivos, só executando transações quando são explicitamente requisitadas pelo usuário ou aplicação. Um banco de dados é *ativo* quando *eventos* gerados interna ou externamente ao sistema provocam uma resposta do próprio banco de dados (BD), independente da solicitação do usuário. Neste caso, alguma *ação* é tomada automaticamente dependendo das *condições* que foram especificadas sobre o estado do banco de dados. Este paradigma é útil para implementar várias funções do próprio banco de dados ou mesmo para estendêlas. Alguns exemplos de aplicações são: controle de integridade, controle de acesso, políticas de segurança e atualização.

Há diferentes formas de transformar um banco de dados passivo em ativo, as mais tradicionais são: escrever no próprio programa de aplicação a condição que se deseja testar; e avaliar continuamente a condição, ou seja, polling. Uma desvantagem da primeira alternativa é que a avaliação da condição é responsabilidade do programador. No segundo caso, o problema pode ser a baixa utilização dos recursos se houver uma excessiva freqüência dos testes de condição. Estes problemas são resolvidos parcialmente com o uso de regras e gatilhos.

Bancos de dados *dedutivos* fornecem um mecanismo para derivar dados que não estão explicitamente armazenados no banco de dados (conhecidos como dados virtuais ou

dados derivados). São mais poderosos e expressivos que as visões, embora mais problemáticos para serem suportados. Não existe uma divisão óbvia entre os bancos de dados dedutivos e os ativos. A principal diferença está baseada no modelo de execução. No primeiro tipo, geralmente a preocupação é a derivação de informação, e as regras são executadas explicitamente pela aplicação. No segundo, as regras (ou gatilhos) são disparadas como efeito colateral das ações normais do banco de dados. As pesquisas em banco de dados ativos (BDA) podem ser divididas em três categorias: i) a definição das regras ou gatilhos, ii) seu correspondente modelo de execução, e iii) a sua otimização.

Na primeira categoria definem-se quais são os tipos de eventos, condições e ações. Uma questão muito importante é a expressividade da linguagem de especificação. Na segunda categoria, a maioria dos artigos especifica com detalhe os modelos de execução e alguns outros só esquematizam o sistema implementado. Nesta categoria discute-se quando devem ser avaliadas as condições, ou como devem ser resolvidos os conflitos (quando mais de uma regra é habilitada ao mesmo tempo). Por último, existe o problema da otimização da execução, tendo em consideração estratégias eficientes para avaliação da condição. Na seção seguinte são descritas algumas das aplicações dos sistemas de gerenciamento de BDA. A seguir são apresentadas as características dos BDA segundo as três categorias descritas anteriormente (definição de regras, modelo de execução, e otimização da execução). Finalmente, são descritas as características "ativas" dos principais protótipos desenvolvidos na área.

2.1 Uso dos Sistemas Ativos

Os sistemas de bancos de dados ativos podem ser classificados em três grupos:

Suporte automático ao usuário

Notificação: Em algumas situações o banco de dados precisa da intervenção do usuário. As regras podem ser usadas para detectar automaticamente estas situações e informar ao usuário.

Execução automática de procedimentos: Ante a ocorrência de um evento, um procedimento pode ser executado mediante o uso do sistema de regras.

Provimento de valores default: Uma operação comum nas aplicações é a atribuição de valores *default*, que pode ser feita diretamente com regras.

Funcionalidade do modelo de dados

Manutenção da integridade: Uma *restrição de integridade*, num ambiente de BD, é um predicado sobre estados do BD que deve ser mantida, para assegurar a consistência dos dados. Num sistema ativo, as condições que violam a integridade são especificadas na parte da condição da regra e a ação corretora especificada na parte correspondente à ação. Este esquema traz duas vantagens importantes: i) o BD suporta especificação e manutenção de restrições, sem depender do código da aplicação, ii) o BD serve para todas as aplicações que têm a mesma visão do mundo.

Proteção: O acesso ao banco de dados pode ser controlado usando regras. Com este esquema não só pode ser aceito ou rejeitado o acesso aos dados, como também o sistema pode fornecer dados fictícios nos campos protegidos.

Gerenciamento dos recursos

As regras são usadas dentro do próprio núcleo do BD.

Otimização do armazenamento físico: Podem ser usadas para ações de *checkpoint*, *clustering*, caminhos de acesso, ou também para adaptar as estruturas de armazenamento segundo estatísticas das consultas.

Gerenciamento de visões: Visões materializadas complexas podem ser mantidas com regras.

2.2 Regras e Gatilhos

Bancos de dados ativos são via de regra implementados a partir de mecanismos de regras de produção. Regras são descrições de comportamento a serem adotadas por um sistema. As regras estão geralmente baseadas em três componentes: *evento*, *condição* e *ação* (E-C-A). O evento é um indicador da ocorrência de uma determinada situação. Uma condição é um predicado sobre o estado do banco de dados. Uma ação é um conjunto de operações a ser executado quando um determinado evento ocorre e a sua condição é avaliada como verdadeira. Um evento pode disparar uma ou mais regras. O primeiro na formalização de sistemas ativos foi Morgenstein. No trabalho descrito em (Morgestein, 1984) utilizam-se os sistemas ativos para manter restrições através das equações de restrições (Constraint Equation, CEs). As CEs permitem expressar restrições semânticas que requerem consistência entre muitas relações, de forma similar à manutenção de relações. As CEs constituem uma forma mais concisa que a escrita de procedimentos para

expressar e garantir restrições, por possuírem representação declarativa, e uma interpretação executável, sendo compiladas em rotinas para garantir automaticamente as restrições.Os gatilhos são associações de condições e ações, a execução da ação ocorre quando o banco de dados evolui para um estado que leva o gatilho à condição verdadeira.

2.3 Regras E-C-A

A forma atualmente aceita para considerar um BDA é a adoção de regras de produção E-C-A.

Evento. O evento é um indicador da ocorrência de uma determinada situação (quando avaliar). Existem basicamente três tipos de eventos: temporais (às 8:30, repetidas vezes toda sexta às 10:00), definidos pelo usuário (alta temperatura, user-login, etc.), e operações próprias dos BD (insert, delete, update, select).

Condição. Uma condição é um predicado sobre o estado do banco de dados (o que avaliar). Condições são comumente implementadas por consultas ou por procedimentos da aplicação.

Ação. Uma ação é um conjunto de operações a ser executado quando um determinado evento ocorre e a condição associada é avaliada como verdadeira (como responder). Um evento pode disparar uma ou mais regras. As ações típicas são: operações de modificação ou consulta, comando do BD (commit, rollback), ou procedimentos da aplicação (podendo ou não acessar o BD).

Existem dois aspectos importantes no projeto da linguagem de regras de produção: a sintaxe para a criação, modificação, e eliminação de regras; e a semântica do processamento das regras na execução. A sintaxe da maioria das linguagens é similar (baseadas na extensão da linguagem de consulta). No entanto, a semântica do processamento varia consideravelmente.

2.4 Componentes

São três as componentes básicas para a obtenção de um sistema ativo:

Monitoramento de Eventos: É o módulo encarregado de detectar eventos e ativar as regras que dependam desse evento.

Avaliação da Condição: Depois que o evento foi detectado o avaliador da condição é responsável pela avaliação eficiente das condições. Aquelas regras cujas condições sejam verdadeiras são passadas para o *Executor de ações*.

Execução de Ações: Este componente coordena o sincronismo entre detecção de eventos e execução de ações. As ações podem ser executadas de imediato (antes do fim da transação que disparou a regra), ou depois (numa transação independente). A ligação entre a execução de ações e regra é denominada *modo de acoplamento*. Dependendo dos *modos de acoplamento* a serem adotados pelo sistema de regras, o gerenciador de transações subjacente deve suporte transações aninhadas.

2.5 Linguagens para Especificar Eventos

Um BDA precisa detectar a ocorrência de qualquer evento definido para poder iniciar a ativação das regras. Para que as situações reais possam ser monitoradas, uma linguagem de definição de eventos deve ser empregada permitindo a modelagem de situações complexas. Alguns exemplos deste tipo de linguagens são Ode, Samos , Compose e Snoop entre outros. Álgebras de eventos são incorporadas às linguagens de especificação de eventos para permitir a composição de eventos. Os principais operadores encontrados em álgebras de composição como as de Ode, Samos, Snoop, e Compose são:

Seqüência. Indica que o evento composto acontece quando os eventos que constituem a seqüência tiverem ocorrido na ordem determinada.

Conjunção. Indica que o evento composto acontece se todos eventos que formam a conjunção ocorreram, independentes da ordem relativa entre eles.

Disjunção. Indica que o evento composto ocorre se pelo menos um dos eventos que formam a disjunção tiver acontecido.

Negação. Indica que o evento composto acontece se os eventos descritos na expressão de negação não tiverem acontecido num determinado intervalo de tempo.

2.6 Linguagens para Especificar Condições

Condições são expressas por fórmulas, às quais é atribuído um valor booleano quando executadas. Entre a classificação das possíveis teorias da lógica para linguagens de

especificação de condições estão as lógicas proposicionais, lógicas de primeira ordem, cláusulas de Horn, e a lógica temporal. As linguagens de consulta são geralmente usadas para a especificação de condições. Segundo seu resultado quando avaliada (vazio ou não) a condição será verdadeira dependendo da convenção adotada.

2.7 Linguagens para Especificar Ações

As linguagens para especificação de ações podem ser divididas em três grupos: linguagens de consulta, linguagens de consulta estendidas, e acesso direto ao banco de dados.

Linguagens de consulta: As ações são especificadas com a mesma linguagem de consulta dos sistemas de bancos de dados, como por exemplo, SQL. A vantagem é que o acesso ao banco de dados fica sob controle do próprio sistema, e a desvantagem é que limita a funcionalidade. Por exemplo, não podem ser executadas ações externas ao ambiente do banco de dados.

Linguagens de consulta estendidas: Para que as regras possam interagir com o ambiente externo, a linguagem de consulta precisa ser estendida. Assim a linguagem de especificação de ações pode basear-se em duas partes, a linguagem de consulta e a linguagem de comunicação. Esta última é baseada em chamadas a procedimentos convencionais ou em primitivas *send* e *receive*.

Acesso direto ao banco de dados: Uma linguagem algorítmica aumenta a expressividade das ações, quando não podem ser expressas com uma linguagem de consulta estendida. A desvantagem deste tipo de linguagem é a redução das possibilidades de otimização, que é de vital importância para obter uma performance aceitável.

2.8 Modelos de Execução

Os sistemas de banco de dados tradicionais são passivos. A inclusão de gatilhos (ou regras) os converte em sistemas ativos, com a característica de executar automaticamente tais gatilhos (ou regras). Esta inclusão afeta significativamente o modelo de execução. Segundo (Van der Voort, Kersten, 1993) os conceitos mais importantes envolvidos são:

Granularidade da ativação: É o nível onde a ativação do gatilho é detectada. Existem quatro opções para a escolha da granularidade: a sessão do próprio banco de dados (A1), a transação (A2), o comando (A3) e a operação (ou primitiva) do banco de dados (A4). Isto significa que a ativação dos gatilhos ou regras só pode ser feita respectivamente entre sessões, transações, comandos e operações.

Granularidade dos gatilhos (ou regras): Representa a atomicidade de execução do gatilho (ou regra), a saber: a própria regra é a unidade de execução (T1); evento, condição e ação são individualmente atómicos (T2); ou nível das operações (T3) que compõem o evento, a condição e a ação.

Escalonamento da Transação/Aplicação: Depende da unidade de execução estabelecida pela granularidade do gatilho e pela granularidade da aplicação. O uso de gatilhos ou regras para notificação, comunicação, e otimização é suportado por todos os modelos de execução. No entanto, uma granularidade de aplicação mais fina leva a maiores possibilidades no uso de regras. Por exemplo, se a granularidade da ativação é no nível de comando (A3), então podem-se definir regras para a interação com o usuário, o que seria impossível com a granularidade de ativação a nível de primitivas (A4). No caso da manutenção de integridade, é necessário que a integridade seja verificada (ou até corrigida) antes do commit da transação. Para isto é necessário ter o controle em nível de comando (A3), para que as regras possam ser executadas imediatamente antes do commit.

2.9 Escalonamento da Ativação das Regras

Múltiplos gatilhos ou regras podem estar habilitados ao mesmo tempo. Esta situação, conhecida como conjuntos *prontos para executar* pode levar o sistema a um comportamento inconsistente. Vários algoritmos são propostos para resolver este problema, a saber:

Execução em paralelo: Todos os gatilhos ou regras são executados em paralelo. Isto só pode ser feito se as regras não interferem entre si.

Ordem seqüencial de execução: Todos os gatilhos ou regras são executados seqüencialmente, a ordem podendo ser tanto aleatória, por prioridade, ou outras.

Execução simples de uma regra: Só uma regra é escolhida, com as mesmas opções que no caso anterior. A execução de uma regra pode também ativar outras regras, o que é

conhecido como ativação em cascata. Uma vantagem é o tratamento uniforme de todas as transações, mas a desvantagem é que possivelmente pode levar a situações de "livelock".

2.10 Otimização

Existem várias técnicas de otimização para execução de regras que variam segundo os diferentes objetivos dos sistemas ativos e das arquiteturas subjacentes. As características principais destas estratégias são classificadas em três grupos: otimização na execução das consultas, redução na execução de consultas, e otimização no armazenamento. As estratégias do primeiro grupo são as mesmas que as otimizações de ordenação de operações usadas nos sistemas de bancos de dados convencionais. No segundo grupo estão os algoritmos para a redução da quantidade de cálculo. Neste caso o objetivo é tentar excluir código redundante de subexpressões comuns e evitar execução de regras ou gatilhos que não são necessários. Há duas estratégias relevantes dentro deste grupo. A primeira é a otimização da avaliação da condição, como, por exemplo, avaliação incremental das condições.

A segunda é a redução do tempo de computação das ações. Isto pode ser feito executando as regras o mais cedo possível, ou deixando-as para depois, quando talvez seja possível otimizar a execução através de abandono de regras supérfluas ou avaliação em conjunto. O último grupo consiste em estratégias para a otimização no armazenamento de dados e regras. A indexação dos dados neste caso é feita tendo em consideração os predicados das regras, reduzindo assim a quantidade de objetos considerados no momento da avaliação da condição.

2.11 Detecção de Eventos

É obvio que a implementação de um detector de eventos eficiente é crucial para ter um bom desempenho num BDA. Existem várias formas de detectar eventos, alguma das quais são descritas a seguir:

Centralizada. Quando um evento é gerado, todas as regras são verificadas. É o mais fácil de implementar, mas também o de pior desempenho.

Índices. Regras podem ser indexadas segundo os eventos que as ativaram. Assim, quando o evento ocorre, as regras que podem ser disparadas são encontradas rapidamente.

Subscrição. Associação de regras a objetos geradores de eventos. Quando um evento é gerado, este é enviado a todas as regras que subscreveram aquele evento. No Sentinel é usado este mecanismo, mas existe um detector de eventos local à cada regra.

Rede de Eventos. Para cada evento composto é construída uma rede. O sistema trabalha com uma combinação de redes que inclui todas as redes de todos os eventos compostos. Um evento pode participar em mais de uma composição, enquanto na combinação das redes só existe um estado para cada evento. A vantagem do uso destas regras é que os eventos compostos podem ser detectados passo a passo, dada a ocorrência de um evento primitivo, e não é preciso inspecionar um grande número de eventos primários armazenados no registro de eventos.

Capítulo 3

Banco de Dados Temporais

Em banco de dados temporais o tempo é considerado uma sucessão ordenada de pontos (chronon), com alguma granularidade, métrica, definida pela aplicação. Sendo a granularidade a menor unidade que pode ser representada por uma unidade em uma aplicação temporal, i.e., é a representação do chronon. Ao definir-se a granularidade de uma aplicação implica que todos os eventos que ocorrem dentro de uma mesma granularidade serão considerados eventos simultâneos, sendo que, pode acontecer dos mesmos não serem (Elmasri, Navathe, 2005). Está definição deve ser tomada com plena consciência das características da aplicação.

Para representação de dados temporais utiliza-se os seguintes tipos de dados: DATE, TIME, DATETIME, TIMESTAMP, INTERVAL e PERIOD.

3.1 Informação de evento contra informação de duração (ou estado)

Um banco de dados temporal armazena as informações sempre que ocorre um dado evento ou quando um evento é considerado verdadeiro. Um evento pontual, ou fato pontual, normalmente são associados aos bancos de dados por um único ponto de tempo representado por alguma granularidade. Esta informação é freqüentemente representada por uma série de dados temporal. Séries de dados temporais envolvem valores que são registrados de acordo com uma sucessão específica e predefinida de tempo (Elmasri, Navathe, 2005).

Os eventos de duração, por um outro lado, podem ser referenciados por um período de tempo específico em um banco de dados (Elmasri, Navathe, 2005), e.g., um funcionário ocupou a diretoria de uma empresa entre o período de 20 de março de 2008 à 31 de outubro de 2008. Um período de tempo é representado por um ponto inicial e um ponto final [START-TIME, END-TIME] (Elmasri, Navathe, 2005). Essa representação, assim como nos conjuntos matemáticos, podem ser representados por intervalos abertos ou fechados.

3.2 Tempo Válido e dimensões de tempo e transação

Dado um evento associado a um ponto ou a um período de tempo específico, pode-se apresentar significados diferentes. Devido a este fato é preciso interpretar os significados destas associações, a forma mais natural, é que o tempo associado é aquele em que o evento aconteceu ou o período em que o evento foi considerado verdadeiro (Elmasri, Navathe, 2005). Caso se empregue este tipo de interpretação diz-se que o tempo associado é um tempo válido. Bancos de dados temporais que se utilizam dessa interpretação são denominados banco de dados de tempo válido.

Uma outra interpretação utilizada é aquele onde o evento associado refere-se ao tempo de fato em que a informação foi armazenada no banco de dados. Tal interpretação de evento associado é denominada tempo de transação, e os banco de dados temporais que trabalham sobre esse paradigma são conhecidos como banco de dados de tempo de transação.

3.2 Outras Dimensões

Existem outros tipos de interpretação, porém as acima citadas são as mais comumente encontradas, e são chamadas de dimensões de tempo. Uma dada aplicação pode

necessitar de apenas uma dimensão, já em outra, faz-se necessário a presença de ambas dimensões, neste caso o banco de dados é dito banco de dados bitemporal. Pode-se ainda haver a necessidade de outras interpretações do tempo, e estas interpretações serão definidas especificamente para a aplicação, tais bancos serão chamados de banco de dados de tempo definido pelo usuário.

Exemplo: Será levado em conta como exemplo para fixação do conteúdo um banco de dados de tempo válido. O mesmo seguirá o seguinte modelo:

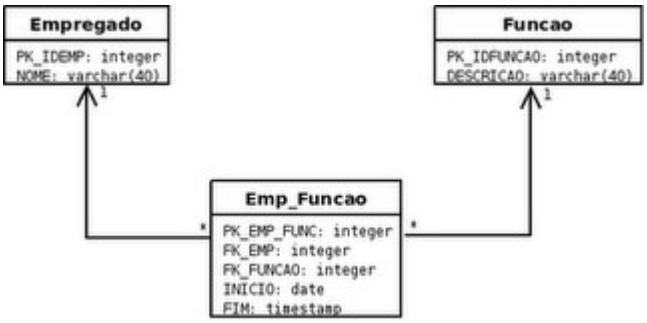

Figura 1. Modelo referente as tabelas do banco exemplificado

A tabela Empregada conta com dois campos: PK_IDEMP e NOME, e receberá o nome dos funcionários de uma empresa fictícia. A tabela Função, assim como a tabela Empregada, possui dois campos: PK_IDFUNCAO e DESCRICAO, e armazenará a descrição da função exercida por um dado empregado. A relação existente entre as duas gera uma terceira tabela denominada Emp_Funcao, que armazenará todas as funções que um empregado exercer na empresa.

Note que existem dois campos na tabela Emp_Funcao que são tipos que representam datas em banco de dados. Os campos são INICIO e FIM, e receberão o START TIME e o END-TIME, respectivamente. Para este exemplo será levado em conta a granularidade de dia. Considere que cada tupla referente à relação Empregado-Funcao representa uma versão válida de um evento ocorrido, em um período[INICIO, FIM]. Geralmente, a versão atual de uma dada tupla, neste caso o campo FIM, possui um valor especial que neste exemplo chamar-se-á de ATUAL. Este valor especial é uma variável temporal que implicitamente representada o tempo corrente à medida que ele acontece (ELMASRI, NAVATHE, 2005). Note que, em uma relação não-temporal só serão armazenadas somente as relações correntes.

Observe os registros inseridos na figura 2.

	PK_IDEMP	NOME
1	1	JOÃO
2	2	MARIA
3	3	JOSÉ
4	4	JOANA

	PK_IDFUNCAO	DESCRICAO
1	1	VENDEDOR
2	2	GERENTE

Figura 2. Registros de Empregados e Funções.

A figura 3 mostra algumas versões de tuplas de tempo válido: duas versões para JOÃO, três para MARIA, uma para JOSÉ e uma para JOANA. Sempre que haver alguma alteração de um funcionário, e.g., promoção, demissão, etc., o seu campo FIM é mudado para o valor do término e, dependendo da situação, uma nova tupla é inserida com as novas informações. As tuplas 1 e 2 da figura 3 tem-se a movimentação das funções para o empregado JOÃO, a segunda tupla da relação indica que foi realizada uma promoção e à medida que a mesma se concretizou(1º de Julho de 2008) a primeira tupla teve seu encerramento um dia antes(30 de Junho de 2008 – GRANULARIDADE DE DIA). A segunda tupla passou a ser a corrente e a anterior passou a ser uma versão fechada ou histórica. O registros 3, 4 e 5 comporta-se da mesma forma que as tuplas 1 e 2. O registro 6 informa a saída de um funcionário da empresa o seu valor de saída é atualizado e não é inserido um novo registro na relação. O contrário ocorre na sétima e última tupla onde é a admissão de um empregado e seu tempo final está definido com o valor especial ATUAL.

	PK_EMP_FUNC	FK_EMP	FK_FUNCAO	INICIO	FIM
1	1	1	1	2008-01-31	2008-06-30
2	2	1	2	2008-07-01	ATUAL
3	3	2	2	2006-02-27	2008-06-30
4	4	2	1	2008-07-01	2008-12-31
5	5	2	2	2009-01-01	ATUAL
6	6	3	1	2008-03-29	2008-07-20
7	7	4	1	2009-03-01	ATUAL

Figura 3. Relação de Empregados por Função.

Detalhes: Note que em uma relação temporal o tempo de início, juntamente com um outro atributo trabalha como chave primária de uma relação, não repetindo em momento algum. Isto ocorre porque em ponto do tempo deveria haver no máximo uma versão de tempo válido. As atualizações podem assumir três tipos: proativa, retroativa e simultânea. As atualizações proativas ocorrem no banco antes que o evento ocorra no mundo real, ao contrário das atualizações retroativas que são persistidas em banco depois da ocorrência do evento. As atualizações simultâneas são as acontecem sobre um mesmo instante de tempo.

Capítulo 4

Banco de Dados Multimídias

Aplicações multimídia vêm se tornando cada vez mais importantes nos dias de hoje. Informações em formato multimídia são muito mais ricas do que em formato convencional (texto) e estão muito mais próximas da percepção humana. Essas aplicações começaram a surgir quando Sistemas de Gerenciamento de Bancos de Dados convencionais já constituíam uma tecnologia bastante madura. Apesar disso, o seu uso mostrou-se inadequado para lidar com dados multimídia, em função da grande quantidade de bytes necessários para este fim, com isso, tornou-se necessário o desenvolvimento de sistemas de bancos de dados especificamente voltados para essas aplicações.

Esses Sistemas de Bancos de Dados Multimídia – SGBD MM – diferem de SGBD's convencionais em vários aspectos. Um bom SGBD MM deve permitir consultas baseadas no conteúdo dos documentos e para isso é importante que ele seja capaz de fazer a interpretação dos dados, com a identificação de objetos conceituais nele contidos e seus relacionamentos. Enquanto nos SGBD's convencionais a apresentação da informação é uma tarefa trivial, nos SGBD MM ela é uma preocupação relevante. O gerenciamento de dados contínuos(áudio e vídeo) introduzem uma dimensão a mais no problema da apresentação: o tempo. Por tudo isso, o gerenciamento de informações multimídia torna-se bem mais complexo. Neste artigo, analisaremos a utilização do GIS – Geographical Information Systems – também conhecido como SIG (Sistemas de Informações Geográficas). Este tem papel relevante na extração de informações, sendo usado para "visualizar o problema, possibilitando observar, manipular e estudar os relacionamentos geográficos envolvidos, e também pode apresentar alternativas à solução do problema considerado" (Egenhofer, 1990). Ao contrário dos SGBD's convencionais, os SGBD MM se caracterizam pela incorporação de mídia contínua como vídeo, áudio e animação. Um dos grandes desafios é o armazenamento destes dados, em função da grande quantidade de bytes necessários para este fim.

4.1 Uso de SGBD's convencionais para modelagem de dados multimídia

Em geral, um sistema de banco de dados necessita suportar as seguintes propriedades: persistência, consistência, desacoplamento das aplicações, interface, acesso multi-usuário e possibilidade de recuperação em caso de falha. Para modelar dados multimídia utilizando SGBD's convencionais, as seguintes técnicas podem ser usadas:

Referências externas: O banco de dados simplesmente possui referências para arquivos que contém os objetos multimídia. Outros atributos podem conter informações como o título do documento, autor, etc. Nesse tipo de abordagem o SGBD não pode prover consistência, nem mesmo persistência dos dados, uma vez que os arquivos externos estão fora do seu controle.

Armazenamento de dados multimídia não interpretados: O SGBD armazena os dados multimídia em atributos dos tipos campo longo ou BLOB (binary large object) para suportar multimídia. No entanto esses tipos reduzem a visão de dados multimídia simplesmente a largos dados não interpretados, o que não é adequado para a rica semântica dos dados multimídia. Em particular, operações dependentes de tempo não podem ser modeladas, ficando o SGBD atuando apenas como um armazenador de dados. Nesse caso o SGBD provê apenas a persistência dos dados, "bufferização" do acesso aos dados, suporte a multi-usuário, recuperação e controle de autorização. Os dados armazenados ainda não são interpretados e as funções disponíveis para operá-los são genéricas. Se não forem providos mecanismos de abstração, não é possível o desacoplamento das aplicações multimídia, em relação à semântica dos dados. Cada aplicação tem que implementar a semântica, por si mesma, inclusive determinar as técnicas de apresentação aplicáveis no objeto. Para processamento eficiente de dados contínuos, conceitos particulares de armazenamento e "bufferização" são necessários. Uma implementação baseada em campos longos de propósito geral não suporta tais mecanismos, em geral, em SGBD's relacionais, o armazenamento e recuperação de documentos multimídia apresentam dificuldades adicionais, devido às estruturas hierárquicas complexas e estruturas seqüenciais, que tipicamente são difíceis de mapear para estruturas relacionais. Isso conduziria para muitas tabelas e expressões de recuperação complexas.

Uso de funções externas: Para contornar essas limitações, alguns SGBD's relacionais implementaram algumas extensões para permitir a representação de tipos de dados multimídia e extensões da linguagem de consulta, como por exemplo, o STARBURST e o POSTGRESS/Ilustra. Extensibilidade permite funcionalidades externas a serem incorporadas ao sistema. Alguns SGBD's permitem o uso de funções externas para processar dados armazenados no banco de dados. Isso é devido às limitações da linguagem de manipulação de dados como a SQL. Freqüentemente, é útil reusar algoritmos externos, programas e ferramentas no contexto de apresentações multimídia.

Orientação a Objetos: Essa abordagem oferece mecanismos para extensão de tipos existentes e a definição de novos tipos (classes) junto com suas operações e oferece o suporte mais flexível, além de permitir a modelagem de relacionamentos complexos entre as entidades. Isso resulta num melhor suporte para modelar objetos multimídia estruturados complexos, a definição de tipos de mídia abstratos e operações sobre unidades de dados de mídia. Abstração de dados é muito importante para processar os documentos eficientemente, evitando duplicatas. Abstrações mais complexas podem ser usadas para indexar dados e prover um acesso rápido. As abstrações podem ser fornecidas pelo usuário ou pelo sistema, baseado no conteúdo dos dados multimídia. É interessante armazená-las, uma vez que sua computação pode ser bem cara. Deve ser possível prover várias camadas de abstração. SGBD's orientados a objeto podem ser usados em aplicações que lidam com gráficos, textos e figuras. Documentos multimídia estáticos com estruturas complexas podem ser modelados sem restrições. Para mídia dependente de tempo e documentos dinâmicos, os problemas de acesso orientado a"stream", acesso em tempo real e técnicas de armazenamento apropriadas continuam. Capacidades de modelagem temporal também faltam. Nenhuma das abordagens convencionais é capaz de suportar "constraints" de sincronização ou QOS. Isso conduz a uma deficiência da interface do banco de dados, na medida em que ele não pode suportar as operações providas pelo modelo apropriadamente. A situação torna-se mais séria em caso de acesso multi-usuário. Embora em muitas situações a semântica de dados multimídia possa ser representada em tipos de dados abstratos, isso pode resultar em baixa eficiência, uma vez que as operações não são diretamente implementadas pelos recursos do SGBD (linguagem de implementação, sistema operacional, etc.), mas em DML (database manegement language), que é interpretada [1].

4.2 Tipos de Dados

Para armazenamento e recuperação de dados multimídia, foram desenvolvidos no banco de dados Oracle os tipos de dados LOB (Large Objects). Segundo a Oracle (2000), os quatro tipos de dados LOBs são BLOB (Binary Large Objects), CLOB (Character Large Object), NCLOB (National Character Large Object) e BFILE (Binary File Object). No banco de dados SQL Server foram desenvolvidos os tipos de dados Image, Text e Ntext para armazenamento de tipos de dados multimídia maiores, que aceitam arquivos de até 2 GB de tamanho. Existem ainda outros tipos de dados, como o binary e o varbinary que armazenam pequenos objetos multimídia, como ícones e bitmaps. No Oracle, uma coluna BLOB pode armazenar objetos binários tais como gráficos, vídeos ou arquivos de áudio. Uma coluna CLOB armazena objetos

caracteres simples, de largura fixa, como documentos de texto. Um tipo de coluna NCLOB é como o tipo de coluna CLOB, mas para conjunto de caracteres de múltiplos bytes. Estes três tipos são chamados de LOBs internos, porque ficam armazenados internamente a base de dados. (Oracle 2000).

Existem ainda os chamados LOBs externos, que podem ficar armazenados em um CDROM, PhotoCD, ou diretamente em arquivos de um sistema operacional. Ao definir uma coluna de dados BFILE em tabela do banco de dados Oracle, é armazenado no banco de dados apenas um ponteiro a estes dados externos. Ao planejar a forma de armazenar os objetos, deve-se verificar o tamanho deles, quantas tabelas os acessarão e qual a freqüência de atualização. Primeiro deve-se decidir se os dados serão armazenados na mesma tabela ou em uma tabela separada, e então decidir se as necessidades serão melhor satisfeitas por um LOB interno ou externo. (Oracle 2000). Segundo Sherer, Gaynor, Valentinsen e Cursetjee (2000:410) um LOB interno tem como os mesmos benefícios proporcionados por um banco de dados a qualquer outro tipo de dado, "... tais como segurança, facilidade de gerenciamento, backup e recuperação e controle de transação". Por exemplo, se uma linha for removida de uma tabela, o documento Word correspondente é removido com ela, se a coluna for do tipo BLOB; mas o documento não será removido se a coluna for definida como um tipo BFILE.

4.3 Aplicações

Entre as diversas aplicações que se beneficiam dos Bancos de Dados Multimídia, destacamos a ***TV Digital***, que "...vem abrir oportunidades para aplicações de tecnologias oriundas de diversas áreas de pesquisa, notadamente as relacionadas ao armazenamento e recuperação de dados multimídia, em função da disponibilização da interatividade", os ***Laudos Periciais*** ,onde ".. Foi desenvolvido um sistema de informação baseado em um banco de dados multimídia para Consulta a Laudos Periciais. Estes laudos contêm informações textuais, imagens,áudio e vídeo. Foi utilizado um sistema de gerenciamento de banco de dados (SGBD) baseado em modelo objeto-relacional (Oracle8i) da Oracle. Este tipo de SGBD é uma evolução das implementações de banco de dados baseados nos modelos relacional e orientado para objetos, e mostra ser adequado para o armazenamento e gerenciamento de dados multimídia." e o ***GIS(SIG)*** – "um sistema que processa dados gráficos e não gráficos (alfanuméricos) com ênfase a análises espaciais e modelagens de superfícies.".

Capítulo 5

Banco de Dados Dedutivos

Pegue um banco de dados com informações contidas nas relações básicas, explicitamente inseridas e adicione a capacidade de definir regras (dedutivas) que permitem derivar novos dados à partir das relações básicas, podendo deduzir ou inferir informação adicional a partir de fatos que estão armazenados. O resultado é um BDD – Banco de Dados Dedutivos. Nele utiliza-se uma linguagem declarativa para especificar regras que descrevem o que se deseja alcançar e como alcançar. Um mecanismo de inferência dentro do sistema pode deduzir novos fatos a partir do banco de dados interpretando essas regras, o que é muito utilizado em Inteligência Artificial.

5.1 Relações Básicas e derivadas

Apesar do grande número de pesquisas na área, os sistemas de bancos de dados dedutivos não são muito utilizados no desenvolvimento de aplicações do mundo real. A tecnologia encarada como promissora é freqüentemente considerada mais teórica do que prática e existem duas razões que explicam isto. Em primeiro lugar, a linguagem de consulta utilizada é baseada no Datalog (evolução do Prolog), criada especificamente para uso com bancos de dados e não possui características que a tornem uma linguagem adequada para utilização em aplicações práticas. Operações aritméticas, de comparação e funções agregadas são características desejáveis em uma linguagem de consulta, ausentes no Datalog puro e em muitas das linguagens utilizadas em implementações de BDDs. A forma com que os atributos de uma relação são referenciados - pela sua posição e não pelo seu nome - também é uma característica incômoda. Além disso, elas geralmente só atendem a consultas e não são capazes de expressar atualizações, característica essencial em aplicações reais. Outro fator que dificulta seu uso prático está relacionado ao uso de um Sistema Gerenciador na implementação de muitos bancos de dados dedutivos. Este tipo de implementação não permite que se mantenha a base de dados já instalada, tendo que convertê-la para o SGBD sobre o qual o mesmo foi criado.

5.2 Atualizações

As atualizações no BDD necessitam de tratamento adicional àquele feito por um banco de dados tradicional. Por isso ele pode ser dividido em duas partes:

- Banco de Dados Extensional (BDE), formado pelo conjunto de fatos básicos, contidos nas relações base, que foram explicitamente inseridos;
- Banco de Dados Intensional (BDI), formado pelas informações contidas nas relações derivadas, deduzidas pela aplicação das regras dedutivas sobre o BDE.

O estado do banco em um BDD não é formado apenas pelo conteúdo das relações básicas, mas por este e por todos os dados implícitos que podem ser derivados do BDE através das regras dedutivas. As características dos sistemas dedutivos, como a capacidade de responder a consultas recursivas, a linguagem de consulta declarativa e a dedução de novas informações, permitem que eles realizem tarefas não suportadas por bancos convencionais. Por isso, aplicações como bases de dados científicas, controle de tráfego aéreo, análises exploratórias de dados são freqüentemente citadas como aplicações que encontrariam grandes vantagens se resolvidas com um BDD.

5.3 Esclarecendo o Datalog

As tentativas iniciais de utilização de linguagens de programação baseadas em lógica buscaram adaptar o Prolog para uso com bancos de dados. O Prolog, porém, apresentou uma série de deficiências, entre elas a influência no resultado final (é desejável uma linguagem de consulta totalmente declarativa, onde a ordem das regras não fosse importante) e a estratégia de avaliação (é mais adequada a recuperação de conjuntos de tuplas por vez, enquanto o Prolog recupera uma tupla por vez). Em vista disso, foi definida uma nova linguagem baseada em lógica para uso específico em bancos de dados: o Datalog.

O Datalog não possui predicados pré-definidos, negação, disjunção e símbolos funcionais. Nele, a ordem das regras não tem importância. No Datalog, uma regra é avaliada derivando o conjunto de todas as constantes possíveis que fazem a cabeça da regra verdadeira e a interpretação do conjunto de regras fica baseada na semântica de ponto fixo. Nesta

semântica, as regras são aplicadas repetidamente até um ponto fixo ser atingido, isto é, até que o estado do banco não possa ser modificado pela aplicação de uma regra. As diversas restrições do Datalog, como a ausência de funções, negação, disjunção, operações aritméticas e operações de comparação, simplificam as formulações teóricas e garantem que toda derivação será sempre finita. No entanto, um sistema de banco de dados construído nas bases desta teoria não atende os requisitos impostos pelas aplicações do mundo real. Assim, os bancos dedutivos, em geral, utilizam extensões do Datalog puro como linguagem de consulta.

5.4 Tapando os Buracos

Para suprir as deficiências do Datalog puro, citadas acima, foi criada a linguagem Dedalo (dedução, dados e lógica), utilizada pelo BDD de mesmo nome. Adicionalmente, criaram notações que permitem referenciar os atributos pelo seu nome e não por sua posição e especificar a ordem desejada das tuplas na relação derivada resultante. A linguagem Dedalo permite também o raciocínio aproximado, através de adaptações da lógica fuzzy para bancos de dados dedutivos.

A lógica e os conjuntos fuzzy foram introduzidos na década de 60 como uma alternativa à lógica bi-valorada, onde os fatos são totalmente verdadeiros ou totalmente falsos. Na lógica fuzzy, os fatos podem ser de 0 a 100% verdadeiros. Já em um conjunto fuzzy, a pertinência de um elemento x em um conjunto C é indicada por um valor entre 0 e 1, onde 0 indica certeza absoluta de que “x ? C” é falso e 1 indica certeza absoluta de que “x ? C” é verdadeiro. Cada relação base do banco de dados deve possuir um atributo com nome padronizado cf (confidence factor), cujo valor varia de 0 a 1 e indica o fator de certeza da tupla. O Dedalo permite que se associe a cada regra um fator de certeza que determina a confiança na verdade da regra.

5.5 Atualizar de Olho na Segurança

As operações de atualização (inserção, exclusão e modificação) podem ser solicitadas na linguagem Dedalo utilizando um dos operadores de atualização providos por ela. Estes operadores são: ins, para inserções, del para exclusões e upd para modificações.

A forma de utilização de cada um dos operadores de atualização é muito semelhante. Basicamente, cada um deles pode ser aplicado diretamente sobre uma relação ou em uma regra de atualização. A aplicação do operador diretamente sobre uma relação básica ou derivada, indica a solicitação da operação de atualização incondicionalmente. Caso o operador seja aplicado sobre uma regra, cuja cabeça é uma relação básica ou derivada em que se deseja efetuar a atualização, a operação será restrita às tuplas que satisfazem as condições expressas no corpo da regra.

Um conceito de grande importância em BDDs é a noção de regras seguras ou permitidas. A segurança de uma regra é garantida por uma verificação puramente sintática que decide se uma regra é tratável ou não. A regra é dita tratável se na sua avaliação todas as variáveis estão instanciadas com constantes. Este conceito é essencial para garantir a computabilidade da negação, as operações aritméticas, de comparação e as funções agregadas. Para uma regra ser considerada segura, todas as variáveis que ocorrem no literal negativo ou na operação de comparação devem ocorrer também no literal positivo. Isto garante que no momento da avaliação da regra, as variáveis do literal negativo ou da operação de comparação estejam instanciadas.

Conclusão

Neste trabalho foi apresentado o conceito para algumas das facilidades comumente necessárias aos bancos de dados para aplicações avançadas: banco de dados ativos, banco de dados temporais, banco de dados multimídia e bancos de dados dedutivos, tais como suas regras, funcionalidades, componentes, aplicações. De maneira geral esclarecendo de forma mais simples essas “facilidades” necessárias aos banco de dados

Apêndice

Questões Relacionadas a Extensões de Modelos de Dados para Aplicações Avançadas

1 – O que é a granularidade em Banco de Dados Temporais?

Em banco de dados temporais o tempo é considerado uma sucessão ordenada de pontos (chronon), com alguma granularidade, métrica, definida pela aplicação. Sendo a granularidade a menor unidade que pode ser representada por uma unidade em uma aplicação temporal.

2 – Defina brevemente Banco de Dados Dedutivo.

Pegue um banco de dados com informações contidas nas relações básicas, explicitamente inseridas e adicione a capacidade de definir regras (dedutivas) que permitem derivar novos dados à partir das relações básicas, podendo deduzir ou inferir informação adicional a partir de fatos que estão armazenados. O resultado é um BDD – Banco de Dados Dedutivos.

3 – A forma atualmente aceita para considerar um BDA é a adoção de regras de produção E-C-A. O que são as regras de produção E-C-A?

Evento. O evento é um indicador da ocorrência de uma determinada situação (quando avaliar). Existem basicamente três tipos de eventos: temporais (às 8:30, repetidas vezes toda sexta às 10:00), definidos pelo usuário (alta temperatura, user-login, etc.), e operações próprias dos BD (insert, delete, update, select).
Condição. Uma condição é um predicado sobre o estado do banco de dados (o que avaliar). Condições são comumente implementadas por consultas ou por procedimentos da aplicação.

Ação. Uma ação é um conjunto de operações a ser executado quando um determinado evento ocorre e a condição associada é avaliada como verdadeira (como responder). Um evento pode disparar uma ou mais regras. As ações típicas são: operações de modificação ou consulta, comando do BD (commit, rollback), ou procedimentos da aplicação (podendo ou não acessar o BD).

Referências Bibliográficas

http://informacaocomdiversao.blogspot.com/2009/04/banco-de-dados-temporal.html

http://www.timaster.com.br/revista/artigos/main_artigo.asp?codigo=982

http://www.dsc.ufcg.edu.br/~baptista/cursos/BDMM/Capitulo1.ppt

http://www.inf.ufrgs.br/~clesio/cmp151/cmp15120021/artigo_montgomery.pdf

NAVATHE, S.; ELMASRI, R. **Sistemas de Banco de Dados Fundamentos e Aplicações**. 2000. Terceia Edição, Rio de Janeiro.