Ciferri, C.D.A., Souza, F.F. Data Warehousing: Estado da Arte e Direções Futuras. *In Memorias da XXVI Conferencia Latinoamericana de Informática*, pages CD-ROM, El Tecnológico de Monterrey, Campus Estado de México, México, September 2000. 12 pp.

Data Warehousing: Estado da Arte e Direções Futuras

Cristina Dutra de Aguiar Ciferri[*]
Departamento de Informática
Universidade Estadual de Maringá
Av. Colombo, 5790 – CEP 87020-900
Maringá, PR, Brasil
cdac@din.uem.br

Fernando da Fonseca de Souza
Centro de Informática
Universidade Federal de Pernambuco
Caixa Postal 7851– CEP 50732-970
Recife, PE, Brasil
fdfd@cin.ufpe.br

Resumo

O tema *data warehousing* engloba algoritmos e ferramentas que tanto possibilitam que dados selecionados de provedores de informação autônomos, heterogêneos e distribuídos sejam integrados em um único repositório de dados, conhecido como *data warehouse*, quanto garantem o acesso aos dados desse repositório em atendimento às consultas e análises multidimensionais realizadas pelos usuários finais. O *data warehousing* provê a base para o ambiente informacional de uma empresa, proporcionando eficiência e flexibilidade na obtenção de informações estratégicas e mantendo os dados sobre o negócio com alta qualidade e confiabilidade. Este artigo tem por objetivo apresentar o estado da arte no tema *data warehousing*, discutindo os principais aspectos relacionados a esse tema. O artigo também destaca problemas em aberto para futuras pesquisas.

Palavras-Chave: Banco de Dados**,** *Data Warehousing*, Sistemas OLAP.

1 Introdução

O tema *data warehousing* engloba arquiteturas, algoritmos e ferramentas que possibilitam que dados selecionados de provedores de informação autônomos, heterogêneos e distribuídos sejam integrados em um único repositório de dados, conhecido como *data warehouse*. Através da arquitetura de um ambiente de *data warehousing*, é possível identificar os componentes que participam do ambiente, o relacionamento que existe entre estes componentes e as funcionalidades de cada um. Tais funcionalidades são oferecidas por algoritmos com propósitos específicos e, muitas vezes, assistidas por ferramentas. Já os provedores de informação podem possuir uma variedade de formatos e modelos e podem incluir desde sistemas gerenciadores de banco de dados (SGBDs) relacionais, orientados a objetos e objeto-relacionais até bases de conhecimento, sistemas legados (tais como sistemas hierárquicos e de rede), documentos HTML e SGML, dentre outros [44].

O acesso à informação integrada dos diferentes provedores de informação é realizado, geralmente, em duas etapas. A informação de cada provedor é extraída previamente, devendo ser traduzida, filtrada, integrada à informação relevante de outros provedores e finalmente armazenada no *data warehouse*. As consultas, quando realizadas, são executadas diretamente no repositório, sem acessar os provedores de informação originais. Desta forma, a informação integrada torna-se disponível para consulta ou análise imediata de usuários finais e programas de aplicação. Por exemplo, uma aplicação de *data warehousing* de uma cadeia de supermercados poderia integrar dados relativos a vendas, produtos e promoções ao longo dos anos. Usuários desta aplicação poderiam realizar, utilizando estas informações integradas, análises de tendências simples (quais as vendas mensais do produto P no ano de 1998?), análises comparativas (quais as vendas mensais dos produtos da marca M nos últimos três anos?) e análises de tendência múltiplas (quais as vendas mensais dos produtos da marca M nos últimos três anos, de acordo com as promoções realizadas no período do Natal e do Dia das Mães?).

Como o *data warehouse* armazena informações integradas, cujas diferenças semânticas e de modelo já foram eliminadas, ambas as consultas e análises podem ser realizadas rápida e eficientemente. Ademais, devido ao fato de que as consultas e análises são executadas diretamente no *data warehouse* sem acessar os provedores de informação originais, o *data warehouse* encontra-se disponível mesmo quando os provedores não estiverem disponíveis [20]. Outra vantagem refere-se ao fato de que o processamento local nos provedores de informação originais não é afetado por causa da participação destes no ambiente de *data warehousing*.

Nos últimos anos, tem havido um crescimento explosivo da tecnologia de *data warehousing* com relação: (1) à quantidade de dados manipulada por aplicações que utilizam esta tecnologia; (2) ao número de usuários

[*] trabalho realizado com suporte financeiro do programa PICDT-CAPES/UEM.

suportados por estas aplicações; e (3) à adoção desta tecnologia pela indústria. Ademais, diversos produtos comerciais e trabalhos acadêmicos têm sido propostos, com as mais variadas funcionalidades. Dentro deste contexto, este artigo tem por objetivo apresentar o estado da arte no tema *data warehousing*, discutindo os principais aspectos relacionados a esse tema.

O artigo está estruturado da seguinte forma. A próxima seção define o conceito de *data warehouse*, com ênfase nas características e forma de organização de seus dados. A seção 3 descreve a arquitetura típica de um ambiente de *data warehousing*, enfocando as funcionalidades oferecidas por seus componentes. Aspectos relacionados à modelagem multidimensional e ao uso de visões materializadas são discutidos, respectivamente, nas seções 4 e 5. O artigo é finalizado na seção 6 com propostas de pesquisa na área.

2 *Data Warehouse*

O surgimento do *data warehouse*, considerado o coração do ambiente de *data warehousing*, foi motivado pela necessidade de separação entre os ambientes operacional e informacional das empresas. O ambiente informacional de uma empresa (voltado ao processamento de consultas OLAP – *on-line analytical processing*) é fundamentalmente diferente do ambiente convencional de processamento de transações (ambiente operacional) [5, 7, 27, 28, 29, 37, 45]. O volume de dados armazenado é muito maior (de *gigabytes* a *terabytes*), os usuários são diferentes (analistas, executivos, gerentes), o projeto do *data warehouse* reflete as necessidades de análise dos usuários finais (multidimensional) e a carga de trabalho é direcionada à consulta intensiva, sendo formada principalmente por consultas complexas, *ad hoc*, dinâmicas e que podem pesquisar grandes volumes de dados. A principal questão de desempenho desse ambiente é, portanto, a produtividade (*throughput*) das consultas OLAP.

De maneira geral, um *data warehouse* é um banco de dados voltado para o suporte aos processos de gerência e tomada de decisão, e tem como principais objetivos prover eficiência e flexibilidade na obtenção de informações estratégicas e manter os dados sobre o negócio com alta qualidade. Para tanto, armazena dados orientados a assunto, integrados, não-voláteis e históricos.

Os dados de um *data warehouse* são **orientados a assunto** uma vez que são relativos aos temas de negócio de maior interesse da corporação, tais como clientes, produtos, promoções, contas e vendas. Esta abordagem é centrada em entidades de alto nível, com ênfase exclusivamente na modelagem de dados, contendo somente dados relevantes ao contexto de tomada de decisão e estruturados, em geral, de forma desnormalizada. Ademais, os dados de um *data warehouse* são **integrados** a partir do ambiente operacional, o qual possui espalhado em diversos provedores de informação autônomos, heterogêneos e distribuídos informações importantes relativas ao negócio da empresa. No processo de extração, somente um subconjunto dos dados operacionais necessários à análise estratégica são copiados, sendo requerido neste processo a correção de possíveis inconsistências devido a diferenças nos formatos dos dados, semânticas, e na utilização de sistemas operacionais distintos. Já a característica de **não-volatilidade** está relacionada ao fato de que o conteúdo do *data warehouse* permanece estável por longos períodos de tempo. Existem somente dois tipos de operação que ocorrem em um ambiente de *data warehousing*: o carregamento dos dados, o qual engloba os processos de extração, tradução, limpeza, integração, armazenamento dos dados no *data warehouse* e recuperação de falhas, e o acesso a esses dados por usuários finais e programas aplicativos através de ferramentas de análise e consulta. Em especial, operações de alteração praticamente nunca são realizadas como parte normal do processamento. Por fim, os dados do *data warehouse* são **históricos**, ou seja, relevantes a algum período de tempo. Assim, para cada mudança relevante no ambiente operacional é criada uma nova entrada no *data warehouse*, a qual contém um componente de tempo associado implícita ou explicitamente. Devido à característica temporal dos dados, torna-se possível efetuar análise de tendências, uma vez que dados relativos a um grande espectro de tempo encontram-se disponíveis.

Em especial, esses dados são organizados em diferentes níveis de agregação. O nível inferior da hierarquia de agregação possui dados primitivos coletados diretamente a partir do ambiente operacional, constituindo a base para futuras investigações desconhecidas. Por outro lado, o nível superior da hierarquia possui dados altamente resumidos. Entre os níveis inferior e superior podem existir vários níveis intermediários, representando graus de agregação crescente. Os dados armazenados em um nível superior *n* correspondem a alguma forma de agregação dos dados armazenados em um nível *n-1*, sendo que o nível intermediário mais

inferior utiliza os dados do nível inferior como base para suas agregações. Cada nível da hierarquia de agregação pode ser considerado como um conjunto de visões dos dados do nível imediatamente subjacente, sendo que o nível inferior representa um conjunto de visões dos dados operacionais.

Desta forma, um usuário típico de sistemas de tomada de decisão pode iniciar sua análise no nível superior para obter uma visão geral do negócio, podendo percorrer a hierarquia de agregação até o nível inferior à medida que dados específicos são necessários. Como exemplo, em uma aplicação de *data warehousing* para uma cadeia de supermercados, as seguintes consultas poderiam ser solicitadas: vendas anuais da marca M em todas as filiais; vendas mensais no ano de 1998 dos produtos da marca M nas filiais 1 e 2; e vendas diárias no mês de outubro de 1998 do produto P da marca M na filial 1.

3 Arquitetura Típica de um Ambiente de *Data Warehousing*

A figura 1 ilustra a arquitetura genérica utilizada para criar, manter e consultar um *data warehouse*.

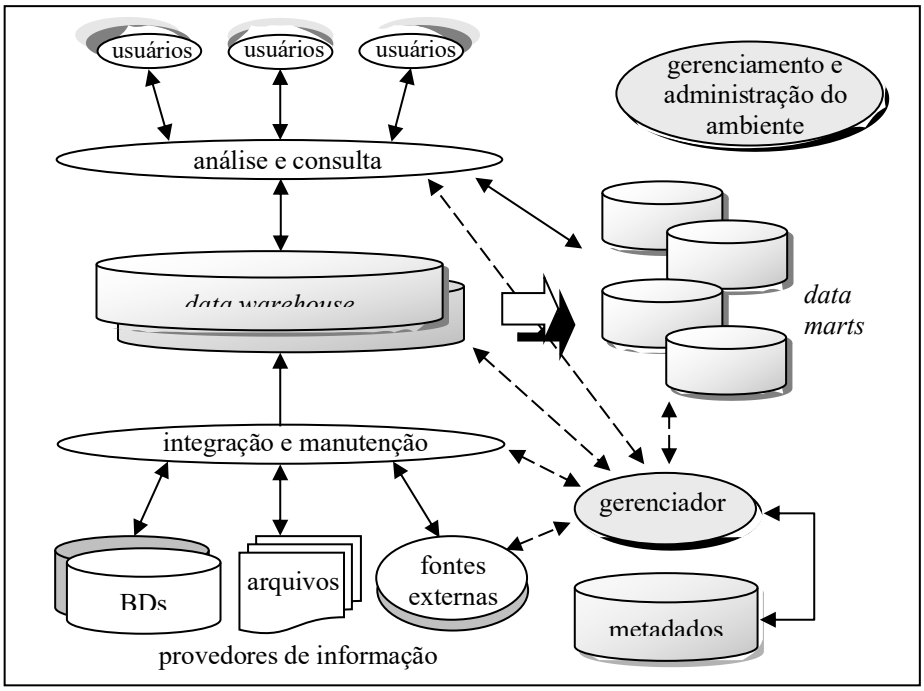

Figura 1 Arquitetura típica de um ambiente de *data warehousing*.

Componente de Integração e Manutenção

Dados relevantes dos provedores de informação que participam do ambiente de *data warehousing* são extraídos, traduzidos, filtrados, integrados e armazenados no *data warehouse* pelo componente de integração e manutenção. O **carregamento dos dados** é uma atividade complexa, cara e demorada, sendo essencial ao bom funcionamento do ambiente. Entretanto, já existem atualmente diversas ferramentas que auxiliam essa atividade, as quais podem ser específicas, enfocando apenas um dos processos, ou integradas, enfocando dois ou mais dos processos envolvidos. Uma lista abrangente de ferramentas com este propósito pode ser encontrada em [3, 18].

Os dados são extraídos dos provedores de informação usualmente utilizando-se interfaces de comandos padronizados, protocolos comum de acesso a dados e conversores de comandos de manipulação de dados e de formatos de dados [28], e são convertidos de seus formatos nativos para o formato e modelo utilizados pelo ambiente de *data warehousing*. Tais traduções devem incluir tanto transformações de esquema quanto transformações nos valores dos dados correspondentes. Ademais, durante o processo de tradução, informações temporais devem ser associadas aos dados sendo manipulados, mesmo que os provedores de informação não sejam históricos.

Assim como em qualquer ambiente que envolve vários provedores de informação heterogêneos, em um *data warehousing* existe uma alta probabilidade de se existir várias cópias da mesma informação em diferentes

provedores, ou informações correlatas armazenadas em diferentes provedores que são inconsistentes entre si. Como resultado, durante o processo de integração, devem ser solucionados conflitos de nome, semânticos e estruturais existentes entre os dados dos diferentes provedores que foram previamente traduzidos, obtendo-se uma visão integrada dos dados do *data warehouse*, sem duplicatas ou inconsistências. Somente após esta etapa é que ocorre o armazenamento propriamente dito dos dados "trabalhados" no *data warehouse*, durante o qual vários processamentos adicionais são realizados, tais como ordenações e agregações.

Durante os processos de extração, tradução, integração e armazenamento, especial atenção é dada à corretude e qualidade dos dados, uma vez que dados duvidosos e/ou incorretos podem gerar informações errôneas que causam um efeito adverso nos lucros da corporação.

Por ser uma atividade complexa e demorada, o carregamento dos dados está sujeito a falhas, as quais podem ocorrer durante qualquer um dos seus processos. Trabalhos voltados à recuperação de falhas em ambientes de *data warehousing* [13, 32] evitam que tanto leituras desnecessárias aos dados dos provedores quanto computações cujos resultados já foram armazenados no *data warehouse* sejam realizadas. Como conseqüência, o carregamento é reiniciado, mas somente dados que não foram corretamente calculados antes da falha ocorrer são considerados.

Além do carregamento dos dados, o componente de integração e manutenção também é responsável por **atualizar o *data warehouse*** periodicamente para refletir as alterações nos dados dos provedores de informação e realizar a **expiração dos dados** do *data warehouse*.

Após o carregamento inicial, os provedores de informação que participam do ambiente podem continuar a alterar os seus dados. Tais alterações devem ser detectadas, propagadas e incorporadas ao *data warehouse*, o qual pode ter seu conteúdo recomputado ou atualizado de forma incremental. Na recomputação, o conteúdo antigo do *data warehouse* é descartado e seus dados são novamente carregados a partir dos provedores de informação originais. Já na atualização incremental, apenas as alterações nos dados dos provedores de informação são propagadas ao ambiente, as quais, juntamente com os dados armazenados no *data warehouse*, são utilizadas para determinar o novo conteúdo do repositório (ver seção 5). Com relação a técnicas de detecção de alterações, trabalhos existentes identificam e extraem somente os dados relevantes ao ambiente, diminuindo assim a quantidade de dados a serem manipulados [8, 31]. Dentre as principais técnicas empregadas, pode-se citar: varredura de *timestamps*, utilização de um arquivo *delta*; utilização do arquivo de *log*, alteração do código fonte, comparação de versões sucessivas de *snapshots* e uso de gatilhos.

Por fim, o ciclo de vida do dado, incluindo sua completa eliminação ou arquivamento final, é uma parte ativa e importante da fase de manutenção do ambiente de *data warehousing*. De maneira geral, dados são expirados do *data warehouse* quando atingem o período de tempo no qual são válidos (dados são considerados "velhos"), quando não são mais relevantes ou necessários para o ambiente ou quando o espaço de armazenamento é insuficiente. No entanto, nem sempre os dados expirados são realmente eliminados do ambiente, podendo muitas vezes serem mantidos em níveis de agregação superiores ou transferidos para outros meios de armazenamento, tal como fita magnética [16, 45].

Componente de Análise e Consulta

O componente de análise e consulta garante o acesso às informações armazenadas no *data warehouse* aos usuários finais e programas aplicativos que participam do ambiente de *data warehousing*. Oferece, desta forma, funcionalidades relacionadas à **consulta e análise** dos dados armazenados, incluindo a habilidade de se determinar a **origem dos dados** sendo examinados.

Ferramentas dedicadas à análise e consulta dos dados oferecem facilidades de recuperação e navegação, permitindo que informações relevantes ao contexto de tomada de decisão sejam derivadas a partir de detecção de análise de tendências, monitoração de problemas e análises competitiva e comparativa. Dentre os principais tipos de ferramentas de acesso existentes, pode-se citar [5]: ferramentas de consulta gerenciáveis e geradores de relatório, ferramentas para sistemas de informações executivas, ferramentas OLAP e ferramentas de mineração de dados. Ferramentas OLAP, por exemplo, apresentam os dados aos usuários de acordo com o modelo multidimensional (ver seção 4), mesmo que estes não estejam organizados de acordo com esse modelo. Independentemente da simplicidade ou sofisticação apresentada por estes tipos de ferramenta, um fator primordial a ser considerado refere-se à visualização dos resultados obtidos. Técnicas de visualização dos dados

devem determinar a melhor forma de se exibir relacionamentos e padrões complexos em um monitor bidimensional, de modo que a solução seja claramente visível. Em especial, essas técnicas devem permitir que os usuários alterem tanto o tipo de informação em análise quanto o método de apresentação em uso.

Enquanto que existe atualmente uma variedade de ferramentas de análise e consulta comercialmente disponíveis (ver [18]), a determinação da origem dos dados sendo examinados somente recentemente tem recebido mais atenção dentro do contexto de *data warehousing* [11]. No entanto, a determinação de conjuntos de dados dos provedores de informação que produz pedaços específicos da informação no *data warehouse* é de grande interesse em aplicações OLAP e de mineração dos dados. Por exemplo, um analista pode desejar identificar as origens dos dados suspeitos ou incorretos gerados em uma visão de alto nível, com o intuito de verificar a integridade e confiabilidade dos provedores ou até mesmo corrigir os dados correspondentes destes provedores.

Demais Componentes

Em adição ao *data warehouse* principal, podem existir diversos ***data marts*** departamentais, contendo réplicas de porções do *data warehouse*. ***Data marts*** são implementações do *data warehouse* nas quais o escopo do dado é limitado, quando comparado ao *data warehouse* propriamente dito, sendo comumente projetados para atender às necessidades específicas de grupos de usuários dedicados. Por outro lado, em uma grande corporação, *data marts* também tendem a ser utilizados como uma política de construção evolucionária do *data warehouse*. Uma vez que o processo de construção de um *data warehouse* sobre toda a organização é longo e os custos envolvidos são altos, *data marts* são criados paulatinamente e, à medida que se consolidam, inicia-se a construção do *data warehouse* global.

Todas as informações estruturais e semânticas dos provedores de informação e do *data warehouse*, além de quaisquer outros dados importantes para o ambiente são armazenadas em um **repositório de metadados** e manipuladas por um módulo **gerenciador de metadados**. Apesar da figura 1 representar apenas um repositório de metadados, White [43] argumenta que em um futuro próximo a arquitetura incluirá vários repositórios de metadados compartilhados. Assim como para as atividades de carregamento dos dados e de análise e consulta, existe atualmente um grande número de ferramentas comerciais especificamente voltadas para o processo de criação e gerenciamento de metadados [18].

Finalmente, a arquitetura engloba ferramentas para o **gerenciamento e administração do ambiente**. Estas ferramentas são responsáveis pelo monitoramento do sistema, realizando tarefas importantes tais como gerenciamento de segurança, testes de qualidade dos dados e *backup*, além de auditoria e relato da utilização do *data warehouse* para gerenciamento do tempo de resposta e uso dos recursos.

4 Modelagem Multidimensional

As análises efetuadas pelos usuários representam, de maneira geral, requisições multidimensionais aos dados do *data warehouse*, as quais têm por objetivo a visualização dos dados segundo diferentes perspectivas (dimensões) [4, 7]. Os **aspectos estáticos do modelo multidimensional** incluem um conjunto de medidas numéricas, que são os objetos de análise relevantes ao negócio, e um conjunto de dimensões, que determinam o contexto para a medida. Uma medida numérica pode ser definida como uma função de suas dimensões correspondentes, representando um valor no espaço multidimensional, ao passo que cada uma das dimensões pode ser descrita por um conjunto de atributos. Tais atributos podem se relacionar através de hierarquias de relacionamento, as quais especificam níveis de agregação e, consequentemente, granularidade dos itens de dados (ex.: os atributos *dia*, *mês*, *ano* da dimensão *tempo* representam uma hierarquia de relacionamento). Ademais, podem existir não somente várias hierarquias de relacionamento ao longo de uma dimensão, mas também alguns atributos que contêm apenas informações adicionais sobre outro atributo da dimensão e, portanto, não podem ser utilizados em agregações.

Já os **aspectos dinâmicos do modelo multidimensional** englobam um conjunto de operações básicas que atuam sobre os dados do *data warehouse*. Operações analíticas típicas incluem *drill-down* (analisa os dados em níveis de agregação progressivamente mais detalhados), *roll-up* (inverso de *drill-down*, ou seja, analisa os dados em níveis de agregação progressivamente menos detalhados), *slice and dice* (restringe os dados a um subconjunto destes dados), *pivot* (obtém diferentes perspectivas dos mesmos dados) e *drill-across* (compara medidas numéricas distintas que são relacionadas entre si através de pelo menos uma dimensão em comum). Estas operações são

utilizadas pelos usuários de tomada de decisão para análises de tendências, comparações, combinações de análise de tendências com comparações e pesquisas em níveis de detalhes sucessivamente mais baixos, dentre outros.

Existem duas principais abordagens utilizadas para a **representação lógica** do modelo de dados multidimensional: estruturas relacionais e estruturas de dados especializadas. Sistemas ROLAP (OLAP relacional) armazenam os dados do *data warehouse* em SGBDs relacionais, os quais são geralmente estendidos para suporte eficiente a consultas de tomada de decisão. Por outro lado, sistemas MOLAP (OLAP multidimensional) armazenam os dados multidimensionais em estruturas de dados especializadas e implementam operações analíticas diretamente sobre estas estruturas.

Sistemas ROLAP

Sistemas ROLAP são geralmente baseados em dois componentes: uma **máquina relacional,** SGBD relacional adaptado para o processamento analítico responsável pelo armazenamento dos dados, e uma **máquina ROLAP**, camada de *software* responsável em suprir as limitações da máquina relacional, apresentando os dados que residem no SGBD relacional de forma multidimensional para ferramentas de análise e consulta. Em particular, uma funcionalidade de extrema importância oferecida pela máquina ROLAP é o armazenamento de visões materializadas no *data warehouse* para aumento da capacidade de processamento de consultas. Esse tópico é discutido mais detalhadamente na seção 5.

Na máquina relacional, existem dois tipos de esquema utilizados para o armazenamento dos dados: esquema estrela e esquema floco de neve. O esquema estrela possui uma tabela de fatos dominante no centro do esquema e um conjunto de tabelas de dimensão nas extremidades. A tabela de fatos armazena as medidas numéricas relevantes ao negócio (fatos), além dos valores das dimensões descritivas para cada instância, os quais são responsáveis pela ligação dos fatos às diversas dimensões. Já as tabelas de dimensão armazenam os atributos alfanuméricos da dimensão e possuem uma chave primária para cada uma de suas instâncias. Em alguns casos, estruturas mais complexas, chamadas de constelações de fato, podem ser empregadas, nas quais várias tabelas de fato compartilham tabelas de dimensão comuns. Enquanto que o esquema estrela apresenta uma estrutura desnormalizada, visando obter um bom desempenho para consultas altamente complexas, o esquema floco de neve apresenta uma estrutura normalizada, na qual cada extremidade do esquema estrela passa a ser o centro de outras estrelas, suportando assim a representação explícita da hierarquia de relacionamento de atributos.

Além de oferecer uma perspectiva diferente na forma de organização dos dados, SGBDs relacionais têm que ser estendidos com diversas funcionalidades adicionais para garantir processamento eficiente de consultas OLAP. Uma primeira otimização consiste em **estender a linguagem SQL** para suportar operadores OLAP. Um operador de extrema importância proposto nesse sentido é o operador *cube* [17], o qual produz todas as agregações possíveis sobre as dimensões especificadas em uma visão multidimensional. Assim, se a visão engloba k dimensões, o operador *cube* produzirá 2^k subconjuntos destas dimensões (ou seja, o cubo de dados). Por exemplo, para a visão multidimensional vendas por *produto* por *filial* por *tempo*, o operador *cube* gerará os seguintes subconjuntos: vendas por *produto* por *filial*, vendas por *produto* por *tempo*, vendas por *filial* por *tempo*, vendas por *produto*, vendas por *filial*, vendas por *tempo* e *total* de vendas. Uma comparação de diversas técnicas propostas até então para cálculo eficiente desse operador pode ser encontrada em [38]. Outros trabalhos nesse contexto referem-se à inclusão de funções estatísticas e financeiras, dentre outras, em alguns sistemas de *data warehousing* comerciais (ex.: Red Brick [26]) e à proposta de operações algébricas voltadas à manipulação do cubo de dados [1].

Outra otimização da máquina relacional refere-se ao oferecimento de **estruturas de indexação** especializadas, com grande destaque para índices *bitmap* [34]. Além de apresentarem tamanho compacto, principalmente para atributos de baixa cardinalidade, índices *bitmap* permitem que operações tais como *and*, *or*, *xor* ou *not* sejam realizadas eficientemente. Atualmente, vários índices *bitmap* (por exemplo, *value-list index*, *projection index*, *bit-sliced index* e *groupset index*) têm sido projetados para diferentes tipos de consultas, incluindo consultas que utilizam funções de agregação básicas do SQL, consultas cujo predicado inclui faixas de valores (*range queries*) e consultas do tipo OLAP (que envolvem funções de agregação e agrupamento).

Uma terceira funcionalidade que deve ser oferecida pela máquina relacional para suporte eficiente ao *data warehousing* consiste em otimizar o **processamento de consultas complexas** [7, 15, 21]. Em especial, a existência de diversas agregações correlacionadas pré-armazenadas incrementa o número de representações relacionais

possíveis para responder a uma consulta, à medida que diferentes níveis de agregação mais inferiores podem ser utilizados para responder a consultas de níveis de agregação mais superiores. Como consultas OLAP geralmente manipulam grandes volumes de dados, escolher um plano de execução que seja ruim pode implicar não somente em baixo desempenho em termos de tempo de resposta, mas também em uma grande quantidade de espaço adicional para executar a consulta.

Sistemas MOLAP

Sistemas MOLAP surgiram como uma segunda alternativa para o armazenamento do cubo de dados multidimensional. Do ponto de vista lógico, o cubo de dados pode ser visto como uma matriz multidimensional, na qual os índices, os quais representam as dimensões, variam sobre segmentos contínuos dos números naturais e as células, as quais representam as medidas numéricas, são formadas pela interseção de todas as dimensões. Apesar de sistemas MOLAP não possuírem uma tecnologia uniforme e comumente aceita para armazenar seus dados, muitos desses sistemas (ex.: Hyperion Essbase [25]) adotam uma representação de armazenamento em dois níveis, ou variações desta [12, 39].

Em uma representação de armazenamento de dois níveis, matrizes mais ou menos densas são indexadas por uma estrutura que contém combinações esparsas dos dados. Mais especificadamente, o nível inferior armazena as dimensões mais ou menos densas em forma de matrizes, enquanto que o nível superior armazena as combinações das dimensões esparsas como uma estrutura de índice que contém ponteiros para as matrizes do nível inferior, as quais possuem a informação desejada (medidas numéricas). Ademais, técnicas de compressão dos dados podem ser aplicadas às matrizes do nível inferior, uma vez que estas podem não ser totalmente densas. A determinação do conjunto de dimensões densas e esparsas é realizada através do uso de ferramentas de projeto ou pela entrada do usuário, ao passo que o endereçamento do nível inferior é geralmente realizado através da utilização de técnicas de indexação tradicionais, tais como árvore-B, árvore-B^+ e *hash*.

Uma característica peculiar de sistemas MOLAP é que muitas vezes eles são construídos no topo de sistemas relacionais. Como exemplo, o produto de *data warehousing* Pilot Decision Support da Pilot Software [35] suporta o armazenamento dos dados de menor nível de granularidade em um banco de dados relacional.

Impacto da Representação Lógica dos Dados Multidimensionais

A utilização de sistemas MOLAP apresenta como principais vantagens a facilidade de manipulação e visualização dos dados, eficientes mecanismos de armazenamento e recuperação, tratamento de esparcidade e presença das perspectivas embutidas diretamente na estrutura de armazenamento de dados. Entretanto, esses sistemas enfrentam dificuldades com relação ao processo de reestruturação de dimensões (por exemplo, adicionar dimensões em sistemas MOLAP implica que a forma da matriz que armazena os dados será alterada) e à capacidade de armazenamento (geralmente limitada pelo número de células suportadas).

Por outro lado, sistemas ROLAP simplificam o processo de reestruturação (adicionar dimensões em uma tabela relacional requer apenas a adição de novas colunas à tabela) e podem manipular *data warehouses* volumosos. Funcionalidades adicionais suportadas por sistemas ROLAP tiram proveito do fato que a tecnologia relacional encontra-se mais consolidada. Assim, o uso de SGBDs relacionais permite a incorporação de novas funcionalidades, tais como paralelismo e tratamento de versões, e garante maior integração com dados não-numéricos. No entanto, como destacado anteriormente, tais sistemas devem ser estendidos para suporte eficiente a consultas OLAP.

Uma vez que ambas abordagens apresentam pontos positivos e negativos, e que diversos trabalhos estão sendo realizados com o objetivo de se solucionar as limitações existentes, tais abordagens estão cada vez mais competitivas, sendo utilizadas através de todos os tipos de negócio.

5 Visões Materializadas

Uma visão materializada consiste em uma visão que possui dados, ou seja, os resultados obtidos em sua execução são armazenados como sua extensão. A materialização de visões correspondentes às diferentes agregações dos níveis da hierarquia de agregação (isto é, correspondentes às diferentes agregações do cubo de

dados multidimensional) consiste na técnica mais efetiva para melhorar o desempenho de consultas OLAP, tanto para sistemas ROLAP quanto para sistemas MOLAP.

O requisito processamento de consulta eficiente é conflitante tanto com o tamanho do *data warehouse* quanto com o tempo gasto para manter a consistência das diversas visões materializadas. Em um extremo, a materialização completa de todas as visões garante um excelente tempo de resposta às consultas submetidas ao sistema. No entanto, pré-calcular e armazenar cada visão não é uma solução adequada principalmente quando o espaço consumido torna-se excessivo, ou quando o tempo de manutenção é proibitivamente alto. O outro extremo, quando nenhuma visão com exceção das correspondentes ao nível inferior é materializada, implica em baixo desempenho de processamento de consultas, apesar de não requerer espaço de armazenamento adicional ou tempo gasto com a atualização dos demais níveis de agregação. Através da materialização no *data warehouse* de um conjunto selecionado apropriado de visões, os custos relacionados ao processamento de consultas e à manutenção de visões podem ser mantidos em níveis aceitáveis, sem desprezar as limitações de espaço de armazenamento.

Assim, trabalhos nessa área enfocam a determinação de um conjunto de agregações a serem materializadas de forma a minimizar os custos relacionados ou com o volume do *data warehouse* [19, 22, 40] ou com ambos o volume e a manutenção dos dados [2, 42], de forma estática (trabalhos anteriores) ou dinâmica [30]. Genericamente, tais trabalhos propõem algoritmos baseados em heurísticas que, de acordo com as restrições de espaço, o conjunto de consultas comumente realizadas pelos usuários e a freqüência dessas consultas, determinam quais visões devem ser materializadas quando o sistema não suporta a materialização de todas as visões.

Em especial, a existência de diversas agregações correlacionadas pré-armazenadas introduz a necessidade de reformulação transparente das consultas dos usuários em termos das agregações mais apropriadas para respondê-las. Em outras palavras, o usuário de tomada de decisão deve especificar suas consultas somente em termos do nível inferior do *data warehouse*, independentemente de quais visões correspondentes às agregações dos demais níveis são materializadas.

O principal objetivo dos trabalhos existentes no sentido de se reformular consultas utilizando visões materializadas [6, 14, 30, 41] consiste em, dado uma consulta Q e um conjunto de visões materializadas, encontrar uma reescrita de Q, chamada de Q', de tal forma que Q' seja equivalente a ou *maximally-contained* em Q, mas que seja definida em termos das visões materializadas. Duas consultas Q e Q' são equivalentes quando calculam o mesmo conjunto de resposta (incluindo valores duplicados, se for o caso) para qualquer estado do banco de dados. Já a consulta reescrita Q' é *maximally-contained* na consulta Q usando um conjunto de visões V se: (i) os nomes das relações que aparecem em Q' são somente os nomes das visões em V; (ii) Q' está contida na consulta Q; e (iii) não existe uma consulta P que também use somente as visões V tal que P está contido em Q e Q' seja contido mas não equivalente a P.

Em suma, tais trabalhos diferem entre si principalmente em termos: (1) da linguagem utilizada para definir ambas as consultas dos usuários e as visões materializadas, incluindo o conjunto de operadores suportados; (2) do fato que valores de instâncias duplicados podem estar presentes ou não e que visões e/ou consultas são recursivas ou não; e (3) da abordagem utilizada para resolver o problema, englobando tanto a premissa na qual o trabalho é baseado quanto a forma na qual o problema é atacado (equivalência ou *query containment*).

Visões materializadas tornam-se inconsistentes à medida que as relações base correspondentes aos diferentes provedores de informação que participam do ambiente de *data warehousing* são alteradas. A manutenção da consistência dos dados do *data warehouse* através da atualização incremental das visões materializadas representa um grande desafio ao ambiente, uma vez que, durante a manutenção da visão, pode ser necessário que as alterações realizadas nos provedores sejam integradas com dados provenientes de outros provedores, antes de serem armazenadas definitivamente no *data warehouse*. Durante este processamento, outras atualizações podem ter ocorrido, gerando inconsistências nos dados integrados.

Pesquisas existentes nesse sentido visam a manutenção incremental das visões materializadas em duas situações: quando os dados dos provedores de informação são acessados durante o processo de manutenção [47, 48, 49] e quando a visão pode ser mantida sem a necessidade de acesso aos provedores de informação originais, ou seja, quando a visão é *self-maintainable* [23, 24, 36, 46]. De maneira geral, na primeira situação, cada

provedor de informação primeiro executa a operação de atualização em suas relações base e em seguida envia a operação ao ambiente de *data warehousing*. O ambiente, assim que recebe a notificação da operação, produz uma consulta (consulta de manutenção) a ser enviada ao provedor, com a finalidade de calcular o novo resultado da visão materializada. Possíveis inconsistências são solucionadas compensando-se o resultado da consulta de manutenção com operações que porventura tenham sido notificadas ao ambiente durante o processamento dessa consulta, mas antes que a operação em questão tenha sido refletida no *data warehouse*. Por outro lado, na segunda situação, as alterações a serem refletidas na visão materializada são calculadas usando somente a visão materializada propriamente dita e as alterações realizadas nos provedores de informação, ou ainda usando um conjunto de visões materializadas auxiliares armazenadas no *data warehouse* de tal forma que ambas a visão materializada e as visões auxiliares sejam *self-maintainable*. Por fim, uma vez que expressões padrão para manter visões materializadas incrementalmente podem ser utilizadas em ambientes de *data warehousing* quando as visões são *self-maintainable*, testes das várias alternativas existentes para se comparar quantidades de trabalho durante a atualização das visões utilizando tais expressões tornam-se necessários [33].

6 Direções Futuras

Este artigo discutiu os principais aspectos relacionados ao tema *data warehousing*, desde sua caracterização até funcionalidades oferecidas por seus componentes. Esta visão geral sobre o tema foi abordada em termos da arquitetura do ambiente de *data warehousing*, da descrição detalhada do principal componente desse ambiente, o *data warehouse*, e da discussão de diversos tópicos em evidência sobre o assunto, tais como modelagem multidimensional e visões materializadas.

Apesar de muita pesquisa na área de *data warehousing* estar sendo realizada, ainda existem muitos problemas em aberto. Ademais, os resultados práticos obtidos até então são limitados. É evidente que já existem diversos produtos comerciais disponíveis atualmente no mercado, como exemplificado ao longo do artigo, mas grande parte das soluções utilizadas por estes sistemas são inflexíveis e assumem diversas simplificações. Assim, muitos resultados, tanto teóricos quanto práticos, ainda precisam ser obtidos.

Uma questão fundamental que merece destaque está relacionada à distribuição dos dados manipulados pelo ambiente de *data warehousing*. Do ponto de vista dos dados do *data warehouse* propriamente dito, poucos trabalhos são voltados ao oferecimento de uma metodologia que permita a distribuição (replicação e/ou fragmentação) de seus dados nos diversos *data marts* que compõem o ambiente, de acordo com os requisitos impostos pelas aplicações, tais como multidimensionalidade dos dados e necessidades dos usuários [9,10]. Com relação ao repositório de metadados, a distribuição monitorada de seus dados em diversos repositórios relacionados também torna-se imprescindível, uma vez que, assim como o *data warehouse*, o repositório de metadados armazena uma quantidade enorme de informação e, em adição, tais informações auxiliam o ambiente durante todo o seu funcionamento.

A existência de diversas ferramentas que auxiliam processos específicos de um sistema de *data warehousing* também introduz desafios. Por um lado, a utilização de diferentes ferramentas voltadas ao suporte destes processos introduz a necessidade de interoperabilidade entre estas ferramentas. Esta interoperabilidade é de suma importância para a flexibilidade das empresas na introdução de novos serviços de aplicação, na redução de custos e na supressão de necessidades. Por outro lado, existem várias ferramentas distintas que apresentam a mesma funcionalidade. Assim, torna-se interessante oferecer um mecanismo automatizado que direcione a escolha da melhor ferramenta, levando-se em consideração fatores tais como os requisitos da aplicação, a relação custo x desempenho da ferramenta e sua facilidade de utilização, além de sua interoperabilidade com outras ferramentas.

Um tópico que sempre merecerá grande evidência consiste no aprimoramento das funcionalidades oferecidas pela máquina relacional – extensão da linguagem SQL, estruturas de indexação, processamento de consultas complexas – visando diminuir cada vez mais as funcionalidades da máquina ROLAP. Por exemplo, a necessidade de índices eficientes para o aumento do desempenho de execução de consultas é contínua. Apesar dos índices *bitmap* serem eficazes, Sarawagi [39] mostra que alguns índices inicialmente projetados para bancos de dados espaciais, tal como *R-trees*, apresentam melhores resultados que índices *bitmap* em certas situações.

Assim, sempre existirão novas oportunidades nesta área. Outras extensões, referentes ao processamento de consultas complexas, são o suporte a consultas aproximadas, assim como em bancos de dados estatísticos, e a otimização simultânea de várias expressões logicamente equivalentes correspondentes à reformulação de uma consulta do usuário, visando diminuir os custos envolvidos neste processo. Já para sistemas MOLAP, o trabalho de Dinter *et all* [12] detalha diversos pontos falhos ou ineficientes relacionados a estes sistemas, tais como distribuição, paralelismo, conceitos transacionais e suporte a versões.

Com relação ao tópico de visões materializadas, muito trabalho ainda precisa ser realizado no sentido de se determinar dinamicamente qual o melhor conjunto de visões a serem materializadas de forma a oferecer um processamento de consultas eficiente sem confrontar com os requisitos de espaço de armazenamento e tempo de manutenção. Um possível caminho a ser seguido consiste na utilização de blocos como unidades de materialização [40]. Ademais, tipos mais complexos de consultas dos usuários devem ser considerados. Com respeito à atualização dos dados do *data warehouse*, existe a necessidade de um estudo sistemático que permita determinar qual a periodicidade de manutenção adequada, em termos do nível de consistência desejado, do tempo gasto para a atualização, da quantidade de agregações suportadas e das necessidades das aplicações.

Outro desafio associado ao tema *data warehousing* consiste na necessidade de exploração e análise dos dados armazenados no *data warehouse* via Web, sem afetar a segurança, a privacidade e a autonomia dos provedores de informação individuais e oferecendo aos usuários acesso uniforme e consistente às informações desejadas. Um ambiente de *data warehousing* para Web deve oferecer: (i) uma interface que facilite que usuários típicos de tomada de decisão realizem consultas de maneira amigável; (ii) suporte a um grande número de usuários; (iii) mecanismos de comunicação eficientes e seguros que permitam a troca de informações entre os provedores localizados na Web e o *data warehouse*, e entre o *data warehouse* e os clientes; (iv) tolerância a falhas; (v) disponibilidade dos dados 24 horas por dia, 7 dias por semana; e (vi) reconfiguração dinâmica. Enquanto que esse desafio utiliza a Web como uma tecnologia subjacente, Harinarayan [21] discute a utilização da tecnologia OLAP como uma tecnologia subjacente para muitos *sites* atualmente disponíveis na Web, através de uma mudança no paradigma de domínio de analistas para o domínio dos consumidores. Esse trabalho argumenta que *sites* de busca, tais como Alta Vista e Lycos, são basicamente "*data warehouses*", mas apresentam uma infra-estrutura para suporte a decisão que é extremamente rudimentar e limitada. Através da estruturação destes "*data warehouses*" aplicando-se os avanços obtidos até então nesta área, consultas complexas tais como as comumente realizadas em sistemas de suporte a decisão podem ser eficientemente respondidas.

Por fim, alguns temas abordados ao longo deste artigo que somente recentemente têm recebido mais atenção da área acadêmica dentro do contexto de *data warehousing* são os problemas de determinação da origem dos dados, de recuperação de falhas e de expiração dos dados do *data warehouse*. O problema de determinação da origem dos dados pode ser adaptado ao processo de limpeza dos dados durante a fase de carregamento, facilitando a identificação e a correção de dados suspeitos ou incorretos. Já a recuperação de falhas em ambientes de *data warehousing* deve ser realizada em duas situações distintas: tanto durante as fases de carregamento dos dados e de atualização dos dados do *data warehouse*, quanto durante o processamento de consultas dos usuários de tomada de decisão. Em ambos os casos, as atividades realizadas são de longa duração, manipulam uma quantidade razoável de dados e utilizam muitos recursos do sistema. Consequentemente, soluções eficazes baseadas em baixa sobrecarga do sistema são essenciais para integrar a recuperação de falhas às demais atividades sendo realizadas. Atualmente, a determinação de quais dados devem ser expirados é realizada pelo administrador do ambiente de *data warehousing*, de acordo com a regulamentação da organização e das necessidades das aplicações. Assim, a especificação de regras que auxiliem esta tomada de decisão representa um tópico de interesse para este problema. De maneira geral, trabalhos apresentando novas metodologias para resolver os problemas de linhagem dos dados, de recuperação de falhas e de expiração dos dados do *data warehouse*, com algoritmos mais eficientes, são indispensáveis para um avanço nestes temas.

Referências Bibliográficas

[1] Agrawal, R., Gupta, A., Sarawagi, S. Modeling Multidimensional Databases. In *Proc. 13th ICDE*, pp. 232-243, Birmingham, UK, April 1997.

[2] Baralis, E., Paraboschi, S., Teniente, E. Materialized View Selection in a Multidimensional Database. In *Proc. 23rd VLDB Conference*, pp. 25-29, Athens, Greece, August 1997.

[3] Berson, A., Smith, S. *Data Warehousing, Data Mining, and OLAP*. McGraw-Hill, USA, 1997. 630 pp.

[4] Cabbibo, L., Torlone, R. Design and Development of a Logical OLAP System. Technical Report InterData T4-R09, Dipartimento di Informatica e Automazione, Universitá degli Studi Roma Tree, Roma Italy, February 1999. 18pp. Available at URL http://www.dia.uniroma3.it/~cabibbo/pub.

[5] Campos, M.L., Rocha Filho, A.V. Data Warehouse. In *Anais da XVI JAI – XVII Congresso da SBC*, pp. 221-261, Brasília, DF, Brasil, August 1997.

[6] Chang, J-y., Lee, S.-g. Query Reformulation Using Materialized Views in Data Warehouse Environment. In *Proc. 1st DOLAP*, pp. 54-59, Washington, DC, USA, November 1998.

[7] Chaudhuri, S., Dayal, U. An Overview of Data Warehousing and OLAP Technology. *SIGMOD Record*, 26(1):65-74, March 1997.

[8] Chawathe, S.S., Rajaraman, A., Garcia-Molina, H., Widom, J. Change Detection in Hierarchically Structured Information. In *Proc. 1996 SIGMOD Conference*, v. 25 of *SIGMOD Record*, pp. 493-504. June 1996.

[9] Ciferri, C.D.A., Ciferri, R.R., Souza, F.F. Uma Arquitetura de Replicação/Fragmentação para Acesso Distribuído a *Data Warehouse* via Web. In *Memorias XXV CLEI*, v. 1, pp.139-150, Paraguay, September 1999.

[10] Ciferri, C.D.A., Souza, F.F. Distributing the Data Warehouse. In *ACM SIGMOD Digital Symposium Collection – DiSC*, September 2000. Available at URL http://www.acm.org.

[11] Cui, Y., Widom, J. Lineage Tracing in a Data Warehousing System. Demonstration Proposal, Stanford University, 1999. 4pp. Available at URL http://www-db.stanford.edu/pub/papers/demo.ps.

[12] Dinter, B., Sapia, C., Höfling, G., Blaschka, M. The OLAP Market: State of the Art and Research Issues. In *Proc. 1st DOLAP*, pp. 22-27, Washington, DC, USA, November 1998.

[13] Do, L., Drew, P., Jin, W., Jumani, V., Rossum, D.V. Issues in Developing Very Large Data Warehouses. In *Proc. 24th VLDB Conference*, pp. 633-636, New York, USA, August 1998.

[14] Duschka, O.M., Genesereth, M., R. Answering Recursive Queries Using Views. In *Proc. 16th PODS*, pp. 109-116, Tucson, Arizona, May 1997.

[15] Fang, M., Shivakumar, N., Garcia-Molina, H., Motwani, R., Ullman, J.D. Computing Iceberg Queries Efficiently. In *Proc. 24th VLDB Conference*, pp. 299-310, New York, USA, 1998.

[16] Garcia-Molina, H., Labio, W.J., Yang, J. Expiring Data in a Warehouse. In *Proc. 24th VLDB Conference*, pp. 500-511, New York, USA, August 1998.

[17] Gray, J., Bosworth, A., Layman, A., Pirahesh, H. Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals. In *Proc. 12th ICDE*, pp. 152-159, USA, February 1996.

[18] Greenfield, L. The Data Warehousing Information Center. Available at URL http://www.dwinfocenter.org.

[19] Gupta, H., Harinarayan, V., Rajaraman, A., Ullman, J.D. Index Selection for OLAP. In *Proc. 13th ICDE*, pp. 208-219, Birmingham, UK, April 1997.

[20] Hammer, J., Garcia-Molina, H., Widom, J., Labio, W., Zhuge, Y. The Stanford Data Warehousing Project. *IEEE Data Engineering Bulletin*, 18(2):41-48, June 1995.

[21] Harinarayan, V. Issues in Interactive Aggregation. *IEEE Data Engineering Bulletin*, 20(1):12-18, 1997.

[22] Harinarayan, V., Rajaraman, A., Ullman, J.D. Implementing Data Cubes Efficiently. In Jagadish, H.V., Mumick, I.S., editors, *Proc. 1996 SIGMOD Conference*, v. 25 of *SIGMOD Record*, pp. 205-216. June 1996.

[23] Hurtado, C.A., Mendelzon, A.O., Vaisman, A.A. Maintaining Data Cubes under Dimension Updates. In *Proc. 15th ICDE*, pp. 346-355, Sydney, Australia, March 1999.

[24] Huyn, N. Multiple-View Self-Maintenance in Data Warehousing Environments. In *Proc. 23rd VLDB Conference*, pp. 16-25, Athens, Greece, August 1997.

[25] Hyperion. Essbase. Internet. Available at URL http://www.hyperion.com.

[26] Informix Red Brick. Decision-Makers, Business Data and RISQL. White paper. Available at URL http://www.informix.com/informix/solutions/dw/redbrick/wpapers.risql.html.

[27] Inmon, W.H. *Building the Data Warehouse*. John Wiley & Sons, Inc, USA, 2nd edition, 1996. 401 pp.

[28] Inmon, W.H., Hackathorn, R.D. *Using the Data Warehouse*. John Wiley & Sons, Inc., USA, 1994. 285 pp.

[29] Kimball, R. *The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses*. John Wiley & Sons, Inc., USA, 1996. 388 pp.

[30] Kotidis, Y., Roussopoulos, N. DynaMat: A Dynamic View Management System for Data Warehouses. In *Proc. SIGMOD Conference*, pp. 371-382, Philadephia, EUA, June 1999.

[31] Labio, W.J., Garcia-Molina, H. Efficient Snapshot Differential Algorithms for Data Warehousing. In *Proc. 22nd VLDB Conference*, pp. 63-74, Mumbai (Bombai), India, September 1996.

[32] Labio, W.J., Wiener, J.L., Garcia-Molina, H., Gorelik, V. Efficient Resumption of Interrupted Warehouse Loads. Technical Report, Stanford University, 1998. 31 pp. Available at URL http://www-db.stanford.edu/warehousing/publications.html.

[33] Labio, W.J., Yerneni, R., Garcia-Molina, H. Shrinking the Warehouse Update Window. In *Proc. SIGMOD Conference*, Philadelphia, Pennsylvania, USA, June 1999.

[34] O'Neil, P., Quass, D. Improved Query Performance with Variant Indexes. In Peckham, J., editor, *Proc. 1997 SIGMOD Conference*, v. 26 of *SIGMOD Record*, pp. 38-49. ACM Press, June 1997.

[35] Pilot Software. An Introduction to OLAP – Multidimensional Terminology and Technology. White paper, 1998. 24pp. Available at URL http://www.pilotsw.com/olap/olap.htm.

[36] Quass, D., Gupta, A., Mumick, I.S., Widom, J. Making Views Self-Maintainable for Data Warehousing. In *Proc. PDIS*, pp. 158-169, Miami Beach, Florida, December 1996.

[37] Ralph Kimball Associates. A Bibliography about Data Warehouse: Books and Articles. Available at URL http://www.rkimball.com/.

[38] Ross, K.A., Srivastava, D. Fast Computation of Sparse Datacubes. In *Proc. 23rd VLDB Conference*, pp. 116-125, Athens, Greece, August 1997.

[39] Sarawagi, S. Indexing OLAP Data. *IEEE Data Engineering Bulletin*, 20(1):36-43, March 1997.

[40] Shukla, A., Deshpande, P.M., Naughton, J.F. Materializes View Selection for Multidimensional Datasets. In *Proc. 24th VLDBConference*, pp. 488-499, New York, USA, August 1998.

[41] Srivastava D., Dar S., Jagadish H.V., Levy A.Y. Answering Queries with Aggregation Using Views. In *Proc. 22nd VLDB Conference*, pp. 318-329, Mumbai (Bombai), India, September 1996.

[42] Theodoratos, D., Sellis, T. Data Warehouse Configuration. In *Proc. 23rd VLDB Conference*, pp. 126-135, Athens, Greece, August 1997.

[43] White, C. Managing Distributed Data Warehouse Meta Data. *Data Management Review Magazine*, February 1999. Available at URL http://www.dmreview.com/master_sponsor.cfm?NavID=55&EdID=159.

[44] Widom, J. Research Problems in Data Warehousing. In *Proc. 4th CIKM Conference*, pp. 25-30, USA, 1995.

[45] Wu, M.-C., Buchmann, A.P. Research Issues in Data Warehousing. In *Proc. BTW Conference*, pp. 61-82, Ulm, Germany, March 1997.

[46] Yang, J., Widom, J. Maintaining Temporal Views Over Non-Historical Information Sources for Data Warehousing. In *Advances in Database Technology –Proc. 6th EDBT*, v. 1377 of *LNCS*, pp. 389-403. 1998.

[47] Zhuge, Y., Garcia-Molina, H., Hammer, J., Widom, J. View Maintenance in a Warehousing Environment. In *Proc. 1995 SIGMOD Conference*, v. 24 of *SIGMOD Record*, pp. 328-339. ACM Press, June 1995.

[48] Zhuge, Y., Garcia-Molina, H., Wiener, J.L. Consistency Algorithms for Multi-Source Warehouse View Maintenance. *DAPD Journal*, 6(1):7-40, January 1998.

[49] Zhuge, Y., Wiener, J.L., Garcia-Molina, H. Multiple View Consistency for Data Warehousing. In *Proc. 13th ICDE*, pp. 289-300, Birmingham, UK, April 1997.