Capítulo 2

Data Warehousing

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 uma única base de dados, conhecida 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 SGBD relacionais, orientados a objetos e objeto-relacionais até bases de conhecimento, sistemas legados (tais como sistemas hierárquicos e de rede), documentos HTML e SGML (*Standard Generalized Markup Language)*, dentre outros [Wid95].

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*; e
- as consultas, quando realizadas, são executadas diretamente no *data warehouse*, sem acessar os provedores de informação originais.

Desta forma, a informação integrada torna-se disponível para consulta ou análise imediata de usuários de SSD 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 de um determinado produto no ano de 1998?), análises comparativas (quais as vendas mensais dos produtos de uma determinada marca nos últimos três anos?) e análises de tendência múltiplas (quais as vendas mensais dos produtos de uma determinada marca nos últimos três anos, de acordo com as promoções realizadas no período do Dia dos Namorados 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 as análises podem ser realizadas rápida e eficientemente. Além disto, uma vez que as consultas e as 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 [HGMW+95]. 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 ao volume de produtos e serviços oferecidos, ao volume de dados manipulado por aplicações que utilizam esta tecnologia, e à adoção desta tecnologia pela indústria. Outro aspecto através do qual pode-se verificar tal crescimento refere-se ao número cada vez maior de usuários que utilizam estas aplicações, como discutido detalhadamente na seção 1.1.

O mercado de *data warehousing*, incluindo *hardware*, *software* de banco de dados e ferramentas de acesso está expandindo rapidamente. Em 1995, o tamanho deste mercado era de 2 bilhões de dólares [Rud96] e em 1997, de 3,5 bilhões de dólares [Wie97]. Em 1999, pesquisas previam um crescimento deste mercado de 5 bilhões de dólares no início desse ano para 15 bilhões de dólares por volta do ano 2000 [Met99a]. Uma tabela contendo estimativas de vendas, juntamente com a taxa esperada de crescimento anual para seis diferentes categorias de ferramentas pode ser encontrada em [Vas00]. A tabela apresentada refere-se aos anos de 1998 a 2002, enquanto que as seis categorias de ferramenta referem-se a: vendas de SGBD relacionais para *data warehouses*; *data marts*; ferramentas de extração, tradução e limpeza; ferramentas de qualidade dos dados; gerenciadores de metadados; e ferramentas OLAP, incluindo serviços de implementação.

Não apenas o mercado está crescendo, mas também o volume de dados manipulado pelas aplicações de *data warehousing*. Por volta de 1996, a maioria das aplicações utilizava *data warehouses* com volume na faixa de 50 *gigabytes* [Rud96]. Já em 1999 tinha-se a expectativa de que 30% dos *data warehouses* existentes excederiam 1 *terabyte* de dados [Met99a]. Por exemplo, o *data warehouse* Wal-Mart Stores, Inc. [Wal] continha 24 *terabytes* de dados em 1998 [GMLWZ98].

Por fim, diversas corporações, através de todos os tipos de negócio, estão utilizando a tecnologia de *data warehousing* para disponibilizar melhores serviços aos clientes e realizar estratégias de *marketing*. Exemplos de indústrias que exploram esta tecnologia e as suas respectivas aplicações, além de algumas empresas comerciais classificadas de acordo com estas indústrias são: bancária e de serviços financeiros, para a redução de fraudes e a análise de riscos de crédito – Bank of America [Ban], ARM Financial Group, Inc. [ARM]; de serviços e de consultoria, para a análise de utilização – ADP Brokerage Information Service Group [ADP]; de fabricação, para o suporte ao cliente e à remessa de encomendas – Xerox Corporation Corporate Strategic Services (CSS) [Xer], General Electric [Gen]; de telecomunicações, para a análise das chamadas e de tarifas – AT&T Corp. [ATT]; de varejo, para a exploração de vendas de produtos, a análise de promoções e o gerenciamento de estoque – Wal-Mart Stores, Inc.; e de transporte e de distribuição, para o gerenciamento de logística – American Airlines [Ame]. Outras corporações que utilizam esta tecnologia englobam indústrias seguradoras (Fremont Compensation Insurance Company [Fre]), de saúde (Hospital das Clínicas USP – INCOR [Hos]), farmacêuticas (Agouron Pharmaceuticals [Ago]), petroquímicas (Shell E&P Company [She]) e de energia (San Diego Gas & Electric [San]), além de organizações governamentais (United States Postal Service [Uni]) e universidades (Fordham University [For]).

2.1 *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. 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.

A obtenção de informações estratégicas, relativas ao contexto de tomada de decisão, é de suma importância para o sucesso de uma empresa. Tais informações permitem à empresa um planejamento rápido frente às mudanças nas condições do negócio, essencial na atual conjuntura de um mercado globalizado.

Inicialmente, a obtenção de informações estratégicas era efetuada diretamente no ambiente operacional, o qual é constituído por aplicações responsáveis pela operação da empresa. Dentro deste contexto, programas de extração foram criados com o objetivo de obter os dados relevantes à tomada de decisão. Programas de extração são caracterizados por selecionarem determinados dados de um arquivo ou banco de dados e por armazenarem estes dados em outro arquivo ou banco de dados separado, conhecido como extrato de dado. Tais extratos permitiam a manipulação dos dados sem o comprometimento da integridade destes com relação às aplicações já existentes, além de um melhor desempenho do que o desempenho proporcionado com o compartilhamento de recursos computacionais.

A proliferação de extratos de dados e o subseqüente uso desses extratos resultou na criação de uma verdadeira "teia de aranha", na perda de credibilidade dos dados gerados nas análises e relatórios produzidos para o nível estratégico e na diminuição da produtividade de relatórios contendo informações oriundas de diversos extratos de dados, uma vez que os dados existentes eram heterogêneos e inconsistentes. Como exemplo, departamentos podiam produzir, a partir de seus extratos de dados, resultados distintos para um mesmo relatório. Tal inconsistência era gerada a partir das características de cada extrator, os quais possuíam algoritmos de extração diferentes contendo fontes de dados e período de coleta de dados distintos. Ademais, a localização e o tratamento dos dados e o desenvolvimento de extratores customizados para tecnologias diferentes demandavam enorme esforço e tempo, diminuindo a produtividade do processo de análise. Muitas vezes, a ausência de dados históricos ou a diferença temporal presente nos diversos sistemas não integrados inviabilizavam a obtenção de informações estratégicas a partir dos dados armazenados.

Devido aos problemas acima destacados, a separação entre o ambiente operacional, constituído por aplicações que dão suporte ao dia a dia do negócio, e o ambiente informacional, constituído por aplicações que analisam o negócio, tornou-se uma forte tendência. Como conseqüência, o *data warehouse* emergiu como a base para o ambiente informacional de uma empresa. Para tanto, manipula os dados presentes nos diversos provedores de informação que compõem o ambiente operacional da empresa, extraindo-os, transformando-os e integrando-os para fins de exploração e análise. Desde que o *data warehouse* é povoado por dados gerados pelo ambiente operacional, o fluxo de informação em um ambiente de *data warehousing* é geralmente no sentido provedores de informação → *data warehouse*.

2.1.1 Diferenças entre os Ambientes Operacional e Informacional

O ambiente de dados para o suporte aos processos de gerência e tomada de decisão (ambiente informacional) é fundamentalmente diferente do ambiente convencional de processamento de transações (ambiente operacional). Tipicamente, o *data warehouse* é mantido separadamente dos bancos de dados operacionais. Segundo Inmon [Inm96], esta separação é motivada pela diferença presente nos dados, tecnologias, usuários e processamento de ambos ambientes, além da necessidade de segurança e de desempenho na execução das aplicações.

A Tabela 2.1 contrasta as principais diferenças existentes entre os ambientes operacional e informacional [BS96, Ken96, BS97, WB97]. Já a seção 2.1.2 descreve detalhadamente as características dos dados do *data warehouse*, enquanto que a seção 2.1.3 aprofunda a discussão sobre granularidade dos dados.

Aspecto	Ambiente Operacional	Ambiente Informacional
ambiente	voltado ao processamento de transações OLTP	voltado ao processamento de consultas OLAP[1]
tipos de operação mais freqüentes	atualização remoção inserção	leitura
volume de transações	relativamente alto	relativamente baixo
características das transações	pequenas e simples acessam poucos registros por vez	longas e complexas acessam muitos registros por vez e realizam várias varreduras e junções de tabelas
tipos de usuários	administradores do sistema projetistas do sistema usuários de entrada de dados	usuários de SSD por exemplo: executivos, analistas gerentes, administradores
número de usuários concorrentes	grande (geralmente milhares)	relativamente pequeno (geralmente centenas)
interações com os usuários	pré-determinadas estáticas	*ad-hoc* dinâmicas
volume de dados	*megabytes* a *gigabytes*	*gigabytes* a *terabytes*
projeto do banco de dados	normalizado para suporte às propriedades ACID (atomicidade, consistência, isolação e durabilidade)	multidimensional, refletindo as necessidades de análise dos usuários de SSD
granularidade dos dados	detalhado	agregado detalhado
principal questão de desempenho	produtividade da transação	produtividade da consulta
tempo de resposta	geralmente poucos segundos	de minutos a horas
exemplos de aplicação	transações bancárias contas a pagar empréstimos de livros	planejamento de *marketing* análise financeira

Tabela 2.1 Ambiente operacional *versus* ambiente informacional.

[1] o termo OLAP foi introduzido em 1993 por Codd *et al*. [CCS93] para definir a categoria de processamento analítico sobre um banco de dados histórico voltado para os processos de gerência e tomada de decisão.

2.1.2 Características dos Dados

Os dados presentes em um *data warehouse* são caracterizados por serem orientados a assunto, integrados, não-voláteis e históricos [Inm96, Mel97].

Em contraste ao ambiente operacional organizado a partir de aplicações funcionais (orientado a aplicações), os dados armazenados em um *data warehouse* são **orientados a assunto**, ou seja, 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.

A **integração** dos dados presentes no *data warehouse* é a base para uma análise precisa sobre toda a organização. Os dados de um *data warehouse* são obtidos, em sua grande maioria, a partir do ambiente operacional, o qual possui espalhado em diversos sistemas não integrados informações importantes relativas ao negócio da empresa. No processo de extração, somente um subconjunto dos dados operacionais necessários à análise estratégica é copiado, sendo requerido nesse processo a correção de possíveis inconsistências devido a diferenças semânticas, nos formatos dos dados e na utilização de sistemas operacionais distintos. Como exemplo, devem ser resolvidos problemas de homônimos e de sinônimos, e conflitos de chave e de domínio, além da forma de representação dos dados.

A característica de **não-volatilidade** está relacionada ao fato de que o conteúdo do *data warehouse* permanece estável por longos períodos de tempo. O ambiente de *data warehousing* é caracterizado pela carga volumosa de dados e pelo acesso a estes dados através de consultas efetuadas por usuários de SSD (ambiente do tipo carregamento de dados e acesso). Assim, apenas dois tipos de transações, ambas caracterizadas por serem de longa duração e complexas, são geralmente efetuadas: transação de manutenção (para a carga dos dados, visando a manutenção da consistência do conteúdo do *data warehouse* com relação ao dados dos provedores de informação) e transação de leitura (consultas dos usuários de SSD, do tipo somente para leitura (*read-only*)).

Em sistemas de *data warehousing* comerciais atuais, a transação de manutenção é tipicamente a única transação a atualizar o conteúdo dos dados. Essa transação é realizada durante a "janela noturna", a qual representa o período no qual o *data warehouse* permanece indisponível para transações de leitura. Ou seja, operações de modificação (remoção, inserção e atualização) não são efetuadas durante o período de utilização do *data warehouse*. Com isto, pode-se efetuar simplificações no gerenciamento dos dados, tais como mecanismos de controle de *deadlock* e recuperação de falhas. Em especial, as operações de atualização e remoção ocorrem somente em caso de carga incorreta, ao passo que a operação de inserção é caracterizada por apenas anexar dados aos já existentes (operação do tipo somente para inserção no final do arquivo (*append-only*)). No entanto, já existe pesquisa no sentido de se aumentar a disponibilidade de utilização do *data warehouse*, visando que transações de atualização e de leitura sejam manipuladas pelo sistema de forma *on-line*, simultaneamente [QW97].

Por fim, os dados do *data warehouse* são **históricos**, ou seja, relevantes a algum período de tempo, em contraste com o ambiente operacional, no qual os dados são válidos somente para o momento de acesso. 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 (5 a 10 anos) encontram-se disponíveis. Por outro lado, tal característica influencia diretamente no tamanho do

banco de dados. Em geral, o volume de dados armazenado é muito superior ao volume presente no ambiente operacional (Tabela 2.1), representando uma complexidade adicional à administração de um *data warehouse*.

2.1.3 Organização dos Dados

Em um *data warehouse*, os dados são organizados segundo diferentes níveis de agregação (Figura 2.1). O nível inferior possui dados primitivos coletados diretamente do ambiente operacional, constituindo a base para futuras investigações desconhecidas. No entanto, devido ao grande volume de dados periodicamente coletado, ocorre o processo de envelhecimento, no qual os dados presentes no nível inferior são transferidos, no mesmo nível de granularidade, para o nível antigo. Enquanto que no nível inferior os dados são armazenados em discos, no nível antigo os dados são armazenados em fitas. A lentidão no acesso aos dados do nível antigo torna necessária a sua monitoração, a qual pode ser efetuada através da taxação pelos recursos consumidos e pela limitação da quantidade de acessos aos dados de acordo com o perfil de cada usuário de SSD.

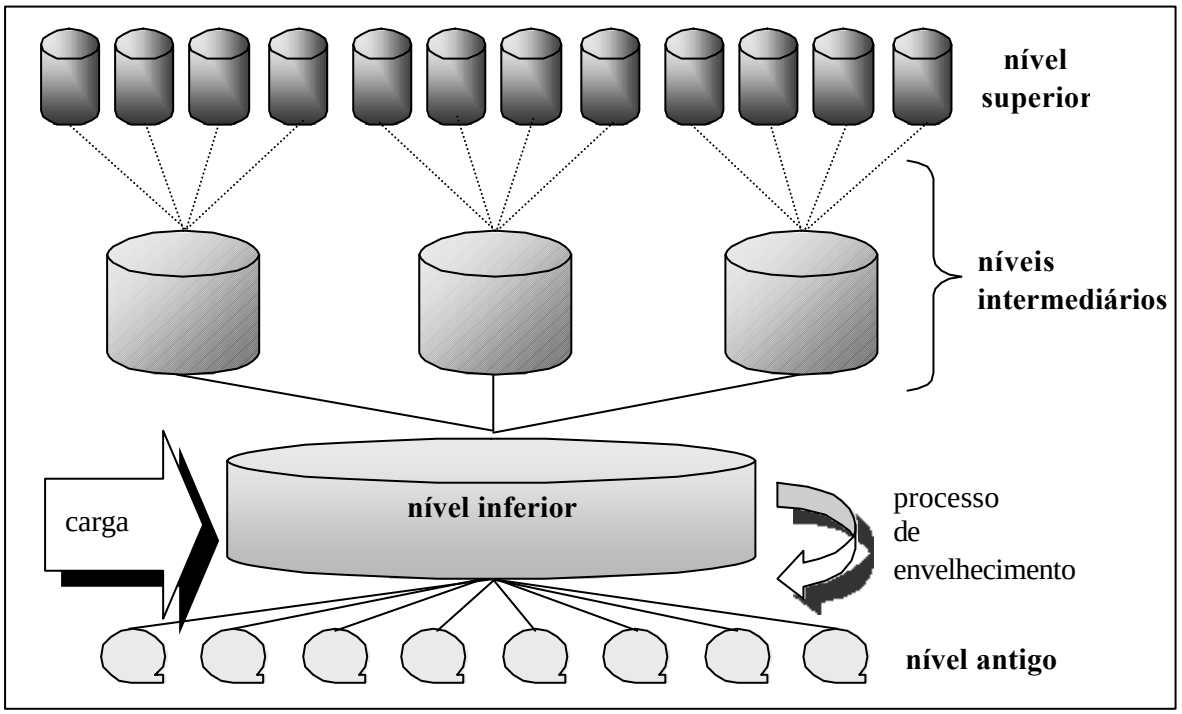

Figura 2.1 Estrutura de um *data warehouse*.

Por outro lado, o nível superior da hierarquia de agregação 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 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. Dentro deste contexto, os dados detalhados do *data warehouse* (nível inferior da hierarquia de agregação) podem ser vistos como um conjunto de visões materializadas definidas sobre as relações base armazenadas nos provedores de informação operacionais. Já cada um dos demais níveis de agregação (dados agregados) pode ser visto como um conjunto de visões materializadas definidas sobre os dados do nível imediatamente subjacente.

Dados resumidos são compactos, armazenados em disco, altamente indexados, facilitam o processo de reestruturação e permitem a visualização de panoramas específicos e globais da empresa, de acordo com o grau de agregação. Usuários de SSD devem ser estimulados a acessar os dados dos níveis superiores, os quais demandam menor esforço no uso de recursos computacionais.

Desta forma, usuários de SSD podem iniciar suas análises 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: (i) vendas anuais dos produtos da marca `M` em todas as filiais; (ii) vendas mensais no ano de `1998` dos produtos da marca `M` fora de promoção nas filiais `1` e `2`; e (iii) vendas diárias no mês de `outubro de 1998` do produto `P` da marca `M` fora de promoção na filial `1`.

A granularidade dos dados armazenados no nível inferior do *data warehouse* é uma questão de projeto muito importante, uma vez que determina a dimensionalidade do banco de dados. Granularidade refere-se ao nível de detalhe em que as unidades de dados são mantidas. Quanto maior o nível de detalhe dos dados, menor o nível de granularidade, e vice-versa.

A escolha da granularidade dos dados do nível inferior causa um impacto profundo no volume de dados armazenado no *data warehouse*, além de afetar os tipos de consulta que podem ser respondidas pelo sistema. Quando se tem um nível de granularidade muito pequeno, o tamanho do *data warehouse* é muito grande, porém praticamente qualquer consulta pode ser respondida. Por outro lado, quando o nível de granularidade é muito alto, ambos o tamanho do *data warehouse* e o número de questões que podem ser manipuladas são reduzidos. Outros impactos causados pela granularidade dos dados são relacionados às estruturas de indexação necessárias e à quantidade de recursos computacionais utilizados no processamento da consulta.

No exemplo da cadeia de supermercados, o qual armazena as vendas de produtos com relação a promoções e filiais ao longo dos anos, uma granularidade apropriada para o nível inferior é o movimento diário de produtos, ou seja, *produto por filial por promoção por dia* [Kim96]. Caso os dados armazenados representassem o movimento anual de produtos (*produto por filial por promoção por ano*), consultas importantes para a tomada de decisão neste ramo, tal como "Desde o início deste ano, as vendas do produto `P` cresceram ou diminuíram?" não poderiam ser respondidas. Por outro lado, o armazenamento de cada produto comprado em cada compra por cada consumidor implicaria em um *data warehouse* extremamente volumoso.

Com relação às informações a serem armazenadas nos demais níveis de agregação, já existem vários trabalhos realizados no sentido de se determinar o conjunto de agregações a serem armazenadas de forma que o custo total de armazenamento seja mínimo. Estes trabalhos são abordados na seção 2.5.1.

Como pode ser observado, um sistema de *data warehousing* não armazena somente dados resumidos, mas também dados primitivos coletados diretamente do ambiente operacional. Isto garante maior flexibilidade ao sistema, uma vez que este pode responder tanto a consultas e análises previamente antecipadas quanto a consultas e análises *ad-hoc*.

2.2 Arquitetura Típica de um Ambiente de *Data Warehousing*

A Figura 2.2 ilustra a arquitetura genérica utilizada para criar, manter e consultar um *data warehouse*.

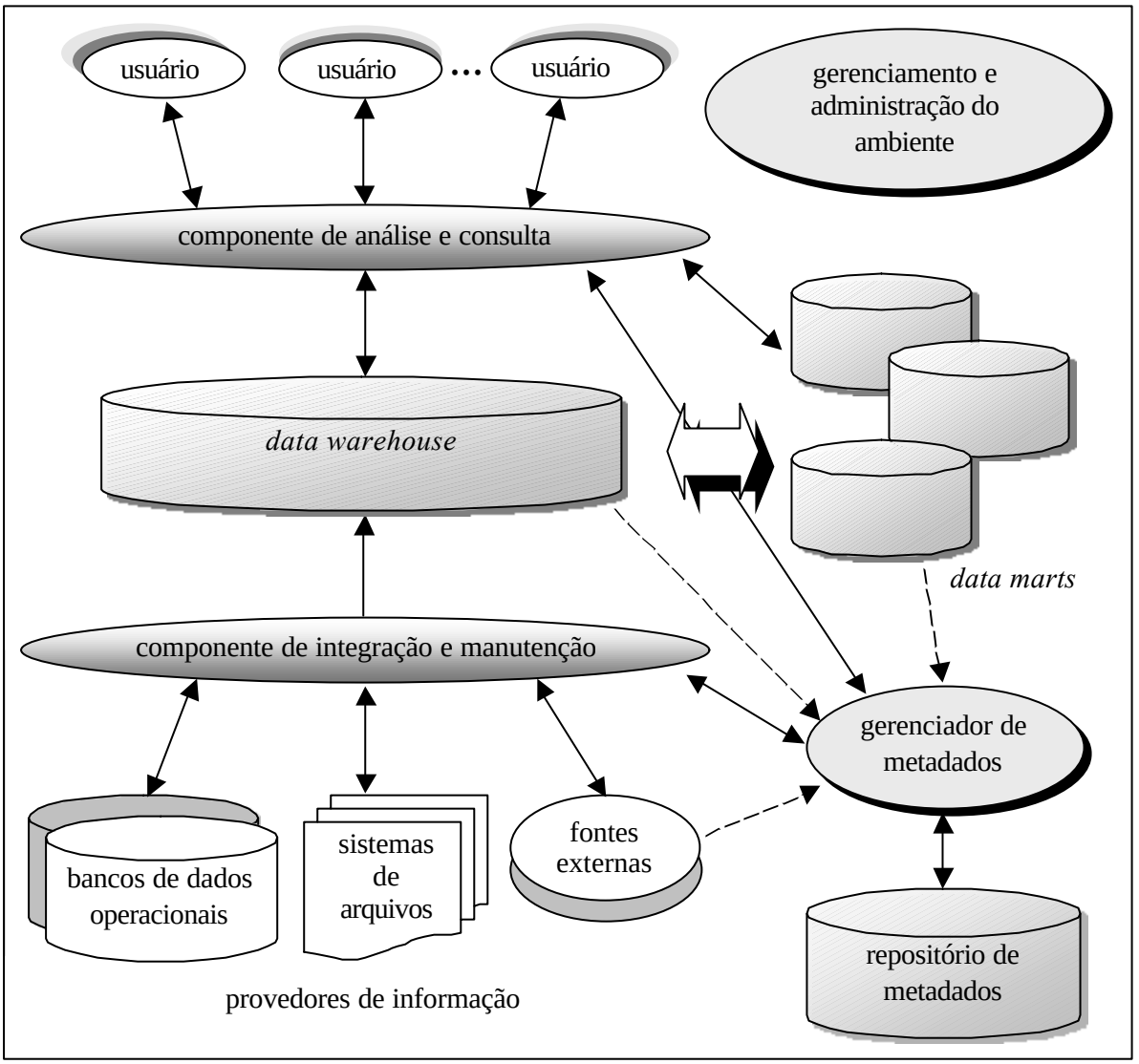

Figura 2.2 Arquitetura típica de um ambiente de *data warehousing*.

Os **provedores de informação** contêm dados operacionais armazenados segundo diferentes modelos e formatos. Dados relevantes destes provedores são extraídos, traduzidos, filtrados, integrados e armazenados no *data warehouse* pelo **componente de integração e manutenção**, o qual 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*.

Em adição ao *data warehouse* principal, podem existir diversos ***data marts*** departamentais, contendo réplicas de porções do *data warehouse*. Ambos os dados do *data warehouse* e dos *data marts* são acessados pelos usuários de SSD através do **componente de análise e consulta**, o qual provê visões multidimensionais destes dados. Este componente e o de integração e manutenção não são independentes. Por exemplo, a determinação de quais informações devem ser armazenadas no *data warehouse* depende das necessidades dos usuários de SSD.

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 2.2 representar apenas um repositório de metadados, White [Whi99] argumenta

que em um futuro próximo a arquitetura incluirá vários repositórios de metadados compartilhados.

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 o gerenciamento de segurança, testes de qualidade dos dados, o gerenciamento e a atualização dos metadados e cópias de reserva (*backup*), além de auditoria e relato da utilização do *data warehouse* para gerenciamento do tempo de resposta e do uso dos recursos.

A seção 2.2.1 contextualiza os componentes da arquitetura com relação às fases de projeto, de criação, de manutenção e de acesso do ambiente. As seções 2.2.2 e 2.2.3 descrevem detalhadamente as funcionalidades oferecidas, respectivamente, pelo componente de integração e manutenção e pelo componente de análise e consulta. A seção 2.2.4 aprofunda a discussão sobre *data marts*, ao passo que a seção 2.2.5 discute as principais conceitos relacionados a metadados.

2.2.1 Fases de Projeto, Criação, Manutenção e Acesso

A fase de projeto de um ambiente de *data warehousing* inicia-se com a definição da arquitetura a ser utilizada. Nesta etapa deve-se realizar o planejamento da capacidade do ambiente e selecionar *hardware* e *software* apropriados. Dentre as principais atividades envolvidas, pode-se citar:

- selecionar servidores de armazenamento, servidores OLAP e de banco de dados, além de ferramentas clientes;
- integrar os servidores e as ferramentas clientes;
- identificar os provedores dos dados relevantes e integrar estes provedores ao ambiente;
- identificar os dados a serem armazenados no *data warehouse*, tanto no nível inferior quanto nos demais níveis da hierarquia de agregação; e
- definir a organização física do *data warehouse*, incluindo a escolha de métodos de acesso.

Após a definição e a especificação do projeto do ambiente, o componente de integração e manutenção deve realizar o carregamento dos dados dos provedores de informação no *data warehouse*. Este carregamento ocorre tanto durante a fase de criação, na qual o *data warehouse* é povoado pela primeira vez, quanto durante a fase de manutenção, na qual alterações nos dados dos provedores são refletidas no *data warehouse*. Outras atividades da fase de manutenção são descritas na seção 2.2.2. Finalmente, o componente de análise e consulta oferece funcionalidades relacionadas ao acesso aos dados, ou seja, como os dados corretamente armazenados no ambiente de *data warehousing* podem ser utilizados por usuários de SSD.

2.2.2 Componente de Integração e Manutenção

O componente de integração e manutenção oferece as seguintes funcionalidades: carregamento dos dados dos provedores de informação no *data warehouse*, atualização periódica desta base de dados e expiração de seus dados. Essa seção descreve cada uma destas funcionalidades em maior nível de detalhe.

2.2.2.1 Carregamento dos Dados

O carregamento dos dados dos provedores de informação, o qual engloba os processos de extração, de tradução, de limpeza, de integração e de armazenamento, representa a atividade mais complexa, cara e demorada, sendo essencial ao bom funcionamento do *data warehousing*. Assim sendo, antes dessa atividade ser iniciada, deve ser desenvolvido um plano de carregamento dos dados para se determinar a melhor forma de migrar os dados dos provedores para o *data warehouse*. Este plano deve ponderar diversos fatores, tais como os recursos disponíveis, os volumes de dados dos provedores, o número de esquemas distintos dos provedores e o número e os tipos de métodos de acesso e plataformas diferentes, além do projeto do *data warehouse*.

Existem diversas ferramentas que auxiliam a atividade de carregamento dos dados, as quais podem ser específicas, enfocando apenas um dos processos, ou integradas, enfocando dois ou mais dos processos envolvidos. Exemplos dessas ferramentas, incluindo as suas respectivas características, podem ser encontrados em [Orl96, BS97, Wil97, Gre].

As seções 2.2.2.1.1 a 2.2.2.1.5 discutem respectivamente particularidades dos processos de extração, de tradução, de limpeza, de integração e de armazenamento dos dados, ao passo que a seção 2.2.2.1.6 salienta a importância do processo de recuperação de falhas.

2.2.2.1.1 Extração dos Dados

O processo de extração dos dados dos provedores de informação que participam do ambiente de *data warehousing* é usualmente implementado de três formas básicas: através de uma interface de comandos padronizados (interface comum), através de um protocolo comum de acesso a dados e através de um conversor de comandos de manipulação de dados e de formatos de dados (*gateway*) [IH94, MST97]. Cada uma destas três abordagens enfatiza um elemento distinto que é comum entre o cliente e o servidor. Considerando-se o processo de extração, uma aplicação que realiza a extração dos dados pode ser considerada um cliente, ao passo que cada provedor de informação é um servidor de dados.

A primeira abordagem usa a interface (tanto do cliente quanto do servidor) como um elemento comum para o acesso aos dados gerenciados por diferentes produtos baseados na linguagem SQL. O princípio por trás desta abordagem consiste na especificação de programas de aplicação clientes de acordo com uma interface genérica (API – *Application Programming Interface*), a qual é independente do tipo do servidor. Em tempo de execução, a aplicação identifica qual a fonte de dados a ser acessada e um módulo específico (*driver*) converte os formatos de dados e os comandos padronizados para os formatos compreendidos pelo servidor alvo. Como resultado, o código da aplicação cliente não precisa ser alterado quando esta for redirecionada para outro tipo de servidor. Exemplos de padrões de interface comum incluem o ODBC (*Open Database Connectivity*) e o BDE (*Borland Database Engine*).

Por outro lado, a abordagem de protocolo comum utiliza um protocolo bem definido para conectar aplicações clientes aos vários tipos de servidores. Intuitivamente, cada interface cliente recebe requisições de aplicações clientes e as traduz em um protocolo comum apropriado. De forma similar, cada interface servidora identifica requisições geradas de acordo este protocolo, processa tais requisições e traduz os resultados obtidos de acordo com o protocolo comum. Através desta abordagem, a interoperabilidade entre os tipos de servidores que utilizam o mesmo protocolo para a comunicação é garantida mesmo que diferentes API tenham sido empregadas. Os padrões DRDA (*Distributed Relational Database Architecture*) e RDA (*Remote Data Access*) são exemplos de protocolos comuns.

Já a abordagem de *gateway* é representada pelo uso de tradutores de comandos de acesso a dados (*gateways* para bancos de dados) como elementos comuns entre clientes e servidores. Tais tradutores assumem a funcionalidade de mapeamento de dados e de comandos entre os vários clientes e servidores. Por não serem padronizados, *gateways* reduzem a portabilidade das aplicações desenvolvidas, uma vez que estas utilizam as API por eles disponibilizadas. Como exemplos de produtos de *gateway* pode-se citar o Database Gateway para DB2 e o Informix Enterprise Gateway.

As alternativas acima descritas podem ser utilizadas tanto separadamente quanto conjuntamente, de acordo com as necessidades da organização. A abordagem de *gateway* usualmente complementa ou estende as outras duas abordagens. Quando desenvolvida com alguma combinação de interface comum ou protocolo comum, adiciona flexibilidade à arquitetura do sistema.

2.2.2.1.2 Tradução dos Dados

O processo de tradução consiste na conversão dos dados do formato nativo dos provedores de informação para o formato e o modelo utilizados pelo ambiente de *data warehousing*. Este processo, assim como o de extração dos dados, é altamente dependente do provedor sendo considerado. Por exemplo, se o provedor de informação armazena seus dados de acordo com o modelo hierárquico e o *data warehouse* utiliza o modelo relacional, rotinas de tradução dos dados do modelo hierárquico para o relacional devem ser especificadas. Uma vez que os provedores de informação que participam do ambiente armazenam seus dados segundo diferentes formatos, diversas rotinas de tradução devem ser desenvolvidas. Tais traduções devem incluir tanto transformações de esquema (alteração da estrutura dos dados) quanto transformações dos dados correspondentes.

Um aspecto muito importante do carregamento dos dados refere-se à característica temporal dos dados do *data warehouse*. Embora a maioria dos provedores de informação não seja histórico, ou seja, armazene somente valores correntes dos dados, o *data warehousing* deve manter informações históricas, uma vez que os usuários de SSD usualmente estão interessados no histórico de como os dados dos provedores evoluíram ao longo do tempo. Desta forma, durante o processo de tradução dos dados, informações temporais devem ser associadas aos dados sendo manipulados. Segundo Yang e Widom [YW98], tais informações podem refletir tanto o momento de atualização dos dados no provedor de informação quanto o momento de armazenamento destes no *data warehouse*, ou ambos.

O problema de tradução de dados não é específico à área de *data warehousing*, tendo sido amplamente pesquisado na área de bancos de dados heterogêneos [EP90, SL90, BHP92, Hsi92]. Dentre os trabalhos existentes, pode-se citar: Sacramento e Laender [SL94] para a tradução do modelo orientado a objetos (modelo de dados do sistema O_2 [Deu+92]) para o modelo relacional, Keim *et al*. [KKM93] para a tradução do modelo de dados relacional para o modelo orientado a objetos, Premerlani e Blaha [PB94] para mapeamentos do modelo relacional para o modelo de dados da metodologia OMT [RBPE91] e Aguiar [Agu95] para mapeamentos entre os modelos ECR [EWH85] e OMT, e entre este último e o modelo de dados do sistema O_2. Por outro lado, os trabalhos de Cluet *et al*. [CDSS98], Siméon e Cluet [SC98] e Abiteboul *et al*. [ACM+99] descrevem o sistema YAT, o qual oferece ferramentas para a especificação e a implementação de conversões de dados entre provedores de dados heterogêneos. Este sistema é composto por um modelo de dados *middleware* para o qual os dados dos provedores são mapeados, uma linguagem declarativa baseada em regras para especificar o processo de tradução dentro deste modelo de dados, um mecanismo de customização e uma interface gráfica. As traduções de um

formato origem para um formato destino são obtidas através da: (i) importação dos dados origem no modelo de dados *middleware*, (ii) tradução para outra representação *middleware* que melhor represente a estrutura destino e (iii) exportação dos dados traduzidos para o sistema destino. Através da linguagem declarativa pode-se descrever as correspondências existentes entre os dados heterogêneos, as quais são utilizadas para derivar automaticamente as regras de tradução entre os formatos. Dentre os formatos disponibilizados pelo sistema, pode-se citar: SGML, HTML, relacional e o formato do sistema O_2.

2.2.2.1.3 Limpeza dos Dados

Desde que o ambiente de *data warehousing* é utilizado para suporte à decisão, é importante que os dados mantidos no ambiente sejam corretos e de qualidade. Em outras palavras, contribuições potenciais dependem da qualidade das decisões tomadas, as quais, por sua vez, dependem da qualidade dos dados utilizados no processo decisório. Em especial, dentre os diversos fatores que podem acarretar o insucesso de um ambiente de *data warehousing*, a falta de atenção com relação à qualidade dos dados representa um fator extremamente comprometedor. Dados duvidosos e/ou incorretos podem gerar informações errôneas que causam um efeito adverso nos lucros da corporação.

A qualidade dos dados está diretamente relacionada às regras de negócio a eles associadas. Uma vez que estas regras são definidas, o dado deve ser testado para verificar a sua consistência e a sua completude com relação a estas regras. Tais regras podem ser inferidas tanto através da identificação das regras de negócio da corporação e a posterior associação destas regras aos dados quanto através da análise dos próprios dados e da identificação de padrões que se aplicam à maioria dos dados analisados. Se, por exemplo, em 95% dos registros analisados a idade das pessoas que compram carros conversíveis com preços mínimos de R$100.000,00 é superior a 40 anos, um registro com valor de idade igual a 18 anos deve ser examinado cautelosamente.

Existem diversas situações nas quais os valores de dados armazenados podem estar incorretos e, conseqüentemente, nas quais a limpeza dos dados torna-se necessária: comprimentos de campos inválidos, dados incompletos ou em branco, duplicações, descrições inconsistentes, violação de restrições de integridade, associações de valores inconsistentes, abreviações de valores não padronizadas e utilização de caracteres ou de tipos de dados não desejados. A atribuição do valor *350* para um campo *idade* é um exemplo de associação de valor inconsistente, ao passo que a existência de uma chave de cliente em um arquivo que não corresponde às chaves do arquivo de clientes representa uma violação da restrição de integridade. Outros exemplos incluem campos com valores de *cor* e *telefone* incorretos, *endereços* parcialmente preenchidos (somente o campo *CEP* é preenchido, os demais campos – *nome da rua*, *número*, *complemento*, *cidade*, *estado*, *país* – são deixados em branco) e *datas* armazenadas erroneamente na forma *ddmmaa* com significado *mmddaa* (010399: 1° de março de 1999 ou 03 de janeiro de 1999).

Por ser uma atividade de extrema importância para o sucesso de um ambiente de *data warehousing*, o processo de limpeza dos dados não deve ser realizado como uma atividade separada, mas sim durante todos os demais processos de extração, de tradução, de integração e de armazenamento dos dados no *data warehouse*.

2.2.2.1.4 Integração dos Dados

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 (representadas de forma igual ou não) ou informações correlatas armazenadas em diferentes provedores que são inconsistentes entre si. Esta diversidade de representações está relacionada, em primeiro lugar, ao fato de que as aplicações representadas por estes provedores foram criadas independentemente, de forma não coordenada e sem considerar que um dia seriam integradas. Depois, a modelagem de cada aplicação varia de acordo com diversos fatores, tais como as necessidades, os objetivos e o universo de atuação de cada aplicação. Esses fatores determinam entidades próprias a cada domínio. Por fim, diferentes analistas podem possuir diferentes percepções do mundo real, originando, muitas vezes, modelos distintos para uma mesma aplicação.

Como resultado desta diversidade de representações, o processo de integração depende da identificação de similaridades e de diferenças existentes entre os dados dos provedores de informação que foram previamente traduzidos, além da identificação de conjuntos destes dados que, apesar de serem distintos entre si, são relacionados por alguma propriedade semântica [Agu95]. Estas similaridades e diferenças devem ser detectadas tanto em nível de esquema quanto em nível de instância.

Considerando-se o nível de esquema, os conflitos existentes entre os dados a serem integrados podem ser divididos em três grupos: conflitos de nome, conflitos semânticos e conflitos estruturais. O primeiro tipo de conflito, conflito de nome, refere-se aos nomes utilizados para representar os diferentes elementos existentes nos esquemas a serem integrados. Diferentes nomes podem ser aplicados ao mesmo elemento (problema dos sinônimos) ou o mesmo nome pode ser aplicado a diferentes elementos (problema dos homônimos). Um exemplo de sinônimo ocorre quando o nome *cliente* é utilizado para representar, em um esquema, todos os clientes atendidos por uma loja, enquanto que o nome *comprador* é utilizado em outro esquema para representar a mesma situação.

Após a identificação dos conflitos de nome, devem ser identificados os conflitos semânticos. Este tipo de conflito surge quando o mesmo elemento é modelado nos diferentes esquemas, porém representando conjuntos que se sobrepõem. Em outras palavras, o conjunto de instâncias do elemento de um esquema é mais abrangente do que o conjunto de instâncias do elemento do outro esquema. Por exemplo, em uma aplicação de um provedor, o elemento *produto* representa todos os produtos de todas as seções de um supermercado, ao passo que em uma outra aplicação de outro provedor, *produto* representa apenas os produtos da seção de cosméticos.

Finalmente, conflitos estruturais surgem sempre que diferentes construtores estruturais são utilizados para modelar o mesmo conceito representado em diferentes aplicações. Como exemplo, considerando-se o modelo entidade-relacionamento, o mesmo conjunto de objetos do mundo real pode ser representado como um tipo-entidade em um esquema e como um atributo de um tipo-entidade em outro esquema. A diversidade de conflitos estruturais que podem aparecer em um problema de integração depende da semântica do modelo de dados utilizado.

Em um ambiente de *data warehousing*, não somente a integração dos esquemas das diferentes aplicações é importante, mas também a integração das suas instâncias correspondentes. Como um exemplo simples da necessidade de integração de instâncias, os dados correspondentes ao campo *sexo* de diferentes provedores podem ser codificados de forma inconsistente. Em uma primeira aplicação, esse campo é instanciado com os valores "M" (masculino) ou "F" (feminino), enquanto que em outra aplicação os valores permitidos são "0" (masculino) ou "1" (feminino). Já em uma terceira aplicação, os valores assumidos podem ser "H" (homem) ou "M"(mulher). Independentemente do formato final do campo integrado *sexo* a ser armazenado no *data warehouse*, os diferentes valores devem ser corretamente decifrados antes de serem armazenados. Outro exemplo consiste em um mesmo campo *largura* armazenado

em quatro diferentes provedores, porém com valores que representam unidades de medidas distintas: centímetros, metros, polegadas e jardas.

Como resultado do processo de integração, tem-se uma visão integrada dos dados do *data warehouse*, sem duplicatas ou inconsistências, e de alta qualidade.

Assim como o problema de tradução dos dados não é específico ao tema *data warehousing*, técnicas de integração de esquemas e de dados propostas na área de bancos de dados heterogêneos podem ser adaptadas a esse tema. Por exemplo, o trabalho de Silva *et al.* [SSU+00] propõe uma arquitetura para ambientes de *data warehousing* que utiliza o SGBD heterogêneo fortemente acoplado HEROS (*Heterogeneous Object System*) [ULM98, UM99] para solucionar o problema de integração dos dados nesses ambientes. Para tanto, o esquema global do sistema HEROS passa a oferecer um metamodelo orientado a objetos baseado no esquema estrela (seção 2.4.2.1.1), o qual é primeiro instanciado e depois especializado de forma a incorporar a semântica de *data warehousing* à federação. A arquitetura proposta faz uso de um programa extrator que requisita ao HEROS a extração/integração dos dados dos provedores de informação e armazena os resultados obtidos no *data warehouse*, baseado no metamodelo correspondente. A idéia subjacente à proposta de arquiteturas de bancos de dados integrados para SGBD heterogêneos e *data warehouses* é introduzida em [SSSB98], sendo descrita mais detalhadamente em [AOSS00]. Esses dois trabalhos também discutem a possibilidade de utilização de um modelo orientado a objetos como modelo de dados canônico para a definição do esquema do *data warehouse*.

2.2.2.1.5 Armazenamento dos Dados no *Data Warehouse*

Após os processos de extração, de tradução, de limpeza e de integração, ocorre o armazenamento dos dados trabalhados no *data warehouse*. Durante esta etapa, vários processamentos adicionais ainda são realizados, tais como a verificação de restrições de integridade, a ordenação dos dados, a geração de agregações, a construção de índices e a condensação dos dados visando-se diminuir o volume de dados a ser armazenado.

2.2.2.1.6 Recuperação de Falhas

Por ser uma atividade complexa e demorada, o carregamento dos dados está sujeito a falhas. Tais falhas podem ocorrer durante qualquer um dos seus processos de extração, de tradução, de limpeza, de integração e de armazenamento, tanto na fase de criação quanto na fase de manutenção do *data warehouse* [DDJ+98]. A idéia fundamental da recuperação de falhas durante o carregamento de dados consiste em evitar que tanto leituras desnecessárias aos dados dos provedores de informação quanto computações cujos resultados já foram armazenados no *data warehouse* sejam realizadas. Como conseqüência, o carregamento é reiniciado, mas somente os dados que não foram corretamente calculados antes da falha ocorrer são considerados.

Dentro deste contexto, o trabalho de Labio *et al*. [LWGMG00] propõe um algoritmo de recuperação de falhas que explora algumas propriedades semânticas genéricas do fluxo de dados que ocorre durante a atividade de carregamento dos dados. Este fluxo de dados é representado por um grafo direcionado acíclico, cujos nós correspondem aos processos de extração e de armazenamento, além de quaisquer outras computações realizadas nos dados, tais como agregações e cujas seqüências de parâmetros de entrada/saída correspondem aos dados que são passados de um nó para o outro. Frente a falhas, o algoritmo identifica quais dados em cada seqüência de entrada já foram processados, remove-os da entrada, de acordo com as

propriedades do fluxo de dados (por exemplo, se os dados são processados em ordem alfanumérica crescente) e com o conteúdo do *data warehouse* e procede o carregamento sem estes dados. Esse trabalho também contrasta diversas técnicas de recuperação tradicionais que poderiam ser adaptadas à atividade de carregamento dos dados em termos da sobrecarga apresentada durante o processamento normal da operação e da capacidade de manipular fluxos de dados genéricos.

2.2.2.2 Atualização dos Dados do *Data Warehouse*

Após o carregamento inicial, os provedores de informação que participam do ambiente de *data warehousing* podem continuar a alterar os seus dados. Visando a manutenção da consistência dos dados do ambiente com relação aos dados dos provedores, tais alterações devem ser propagadas e incorporadas ao *data warehouse* (tanto aos dados do conjunto de visões que representa o nível inferior quanto aos dados das demais visões derivadas armazenadas – demais níveis da hierarquia de agregação dos dados).

A periodicidade da incorporação das alterações aos dados do *data warehouse* depende das necessidades dos usuários de SSD e do nível de consistência desejado. Sistemas comerciais usualmente atualizam seus dados com periodicidade diária ou semanal. No entanto, caso consultas OLAP requeiram dados correntes, cada simples alteração realizada nos provedores de informação deve ser propagada continuamente. A freqüência de incorporação das alterações é determinada pelo administrador do *data warehouse*, podendo ser diferente para provedores distintos.

Existem duas técnicas para se atualizar o *data warehouse*: recomputação e atualização incremental. Na técnica de recomputação, o conteúdo do *data warehouse* é descartado e os seus dados são novamente carregados a partir dos provedores de informação operacionais. Já na técnica de 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 dessa base de dados. Segundo Gupta e Mumick [GM95], técnicas de atualização incremental baseiam-se na heurística de que somente uma parte da visão altera em resposta a alterações nas relações base. No entanto, se uma relação base for excluída e a visão do *data warehouse* evoluir para uma relação vazia, pode ser mais barato recalcular a visão novamente do que atualizá-la incrementalmente.

O processo de atualização incremental inicia-se com a detecção e a propagação de alterações nos dados dos provedores de informação que participam do ambiente. Rotinas de detecção de alterações devem identificar e extrair somente os dados relevantes ao ambiente, diminuindo assim o volume de dados a ser manipulado.

A detecção de alterações nos dados operacionais é realizada de acordo com as facilidades oferecidas pelos provedores de informação. Assim, rotinas de detecção de alterações devem considerar como o provedor indica que seus dados foram alterados e/ou que novos dados foram adicionados. Dentre as principais técnicas utilizadas, pode-se citar: (i) varredura de marcadores de tempo (*timestamps*); (ii) utilização de um arquivo *delta*; (iii) utilização do arquivo de *log*; (iv) alteração do código fonte; (v) comparação de versões sucessivas de instantâneos (*snapshots*); e (vi) uso de gatilhos [Inm96, LGM96].

A primeira técnica consiste na varredura dos marcadores de tempo associados aos dados. Nessa técnica, dados com data anterior à ultima atualização do *data warehouse* são desconsiderados. Já as segunda e terceira técnicas limitam os dados a serem extraídos através da varredura de um arquivo *delta* e do arquivo de *log*, respectivamente. Ambos arquivo *delta* e

arquivo de *log* possuem basicamente as mesmas informações, são criados pelas respectivas aplicações/sistemas e refletem somente as alterações que ocorreram nos dados. A quarta técnica, por sua vez, consiste na alteração do código fonte das aplicações para incorporar o envio de modificações ao ambiente de *data warehousing*. Na técnica de comparação de instantâneos, dois instantâneos sucessivos (um representando os dados operacionais no momento da extração de dados anterior e outro representando os dados no momento da extração atual) são comparados visando-se a identificação de diferenças específicas entre os seus dados. Em especial, o trabalho de Labio *et al*. [LGM96] apresenta algoritmos para a comparação de arquivos contendo conjuntos de registros que possuam chave e permitam as operações de inserção, de remoção e de atualização, ao passo que o trabalho de Chawathe *et al*. [CRGMW96] estuda o problema de detectar e de representar alterações significativas entre duas versões de dados estruturados hierarquicamente. Esse último trabalho também oferece suporte às operações para mover e copiar, em adição às de inserção, de remoção e de atualização. Por fim, SGBD relacionais que incorporam características de bancos de dados ativos podem utilizar gatilhos tanto para a detecção quanto para a notificação automática de alterações nos seus dados.

As técnicas acima discutidas possuem limitações e desvantagens, o que faz com que o processo de detecção de alterações nos dados dos provedores de informação represente um grande desafio para o ambiente de *data warehousing*. Por exemplo, somente poucos dados operacionais possuem marcadores de tempo a ele associados, somente poucas aplicações utilizam arquivos *delta* e somente poucos provedores oferecem recursos de bancos de dados ativos. Com relação ao uso do arquivo de *log*, muitas barreiras devem ser enfrentadas. Geralmente esse arquivo somente pode ser acessado utilizando-se o privilégio de administrador de banco de dados. Ademais, seu formato interno é construído para propósitos específicos do sistema, podendo ser difícil de ser entendido e conter informações adicionais desnecessárias. Por fim, o *log* é considerado um componente essencial no processo de recuperação de falhas, sendo protegido pelo sistema. Já o problema da técnica de comparação de versões sucessivas de instantâneos está relacionado ao fato de que, à medida que o volume de dados do provedor de informação cresce, comparações cada vez maiores precisam ser realizadas.

Uma vez detectada e propagadas as alterações relevantes, estas devem ser incrementalmente carregadas no *data warehouse*, de acordo com as especificações do ambiente. A seção 2.5.2 aborda vários trabalhos e otimizações neste sentido, além de definir o conceito de visão materializada e detalhar o relacionamento deste conceito com os dados armazenados no *data warehouse*.

2.2.2.3 Expiração dos Dados do *Data Warehouse*

Como discutido na seção 2.1.1, o volume de dados armazenado em um *data warehouse* é geralmente muito grande, variando de *gigabytes* a *terabytes* de tamanho. Da mesma forma que as rotinas de extração dos dados dos provedores de informação enfocam a limitação dos dados a serem extraídos destes provedores, o processo de expiração está relacionado com a remoção de dados do *data warehouse*, visando uma diminuição do volume de dados armazenado nessa base de dados. Como conseqüência, 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 limite de tempo no qual tornam-se inválidos (dados são considerados "velhos"), quando não são mais relevantes ou necessários para o ambiente, ou quando o espaço de armazenamento é insuficiente. Como pode ser observado através destes fatores, a expiração dos dados representa um processo

completamente diferente do de atualização dos dados do *data warehouse*, no qual ocorre a propagação de remoções dos dados dos provedores de informação.

Nem sempre os dados expirados são realmente eliminados do ambiente. Muitas vezes esses dados são mantidos em níveis de agregação superiores ou transferidos para outros meios de armazenamento, tal como fita. Quatro diferentes políticas de expiração são apresentadas no trabalho de Wu e Buchmann [WB97]: eliminação completa, eliminação seletiva, arquivamento e eliminação seletiva com arquivamento.

A primeira política, como o próprio nome diz, está relacionada com a eliminação completa dos dados expirados, tanto do nível inferior quanto dos demais níveis de agregação. Os metadados relacionados a estes dados, por outro lado, permanecem armazenados no ambiente de *data warehousing*, permitindo a identificação dos dados operacionais correspondentes caso tais dados precisem ser referenciados novamente. Esta política não é adequada em situações nas quais os dados estão sendo expirados por falta de espaço de armazenamento ou nas quais consultas dos usuários de SSD esperam que níveis com alto grau de agregação englobem informações antigas.

Na política de eliminação seletiva, os dados expirados são eliminados como uma função da capacidade de armazenamento, da freqüência de acesso e das necessidades de desempenho. Por exemplo, apenas os dados expirados do nível inferior da hierarquia de agregação são mantidos, enquanto que os dados relativos aos demais níveis são completamente eliminados, uma vez que os níveis intermediários e superiores são determinados a partir de agregações do nível inferior. No entanto, de acordo com as consultas dos usuários de SSD e o volume de dados armazenado em cada um dos níveis, pode ser mais interessante armazenar somente os dados relativos aos níveis mais superiores, enquanto que os de níveis mais inferiores são eliminados do ambiente. Neste caso, um modelo matemático deve ser utilizado para se determinar quais dados devem ser eliminados ou mantidos no sistema.

A terceira técnica consiste no arquivamento do dado expirado em algum outro dispositivo de armazenamento, antes desse ser completamente eliminado do ambiente. Basicamente, apenas os dados mais detalhados relativos ao nível inferior são armazenados, desde que as agregações podem ser derivadas destes. Esta política pode ser utilizada quando os provedores de informação não são capazes de oferecer dados históricos e quando os dados históricos serão acessados apenas esporadicamente. Um dado deve ser arquivado somente se possuir alguma probabilidade de acesso, mesmo que esta seja pequena. Quando sua probabilidade de acesso aproximar-se a zero, então este deve ser realmente removido.

Por fim, a técnica de eliminação seletiva com arquivamento representa uma combinação das duas técnicas anteriores, e é indicada para casos nos quais os provedores de informação não são históricos e a freqüência de acesso aos dados expirados é relativamente alta. Neste caso, os dados detalhados são arquivados antes de serem eliminados e alguns dados agregados correspondentes a estes dados arquivados continuam presentes no *data warehouse*.

Uma variação destas políticas é apresentada no trabalho de Garcia-Molina *et al*. [GMLY98], que propõe um método para a eliminação de dados expirados do nível inferior, sem que os demais níveis de agregação sejam afetados. Este método é baseado na premissa de que alguns dados podem ser removidos sem problemas, ao passo que outros devem ser mantidos no sistema porque podem ser indispensáveis ao processo de atualização do *data warehouse*. No entanto, os dados expirados mantidos no sistema tornam-se inacessíveis a consultas dos usuários de SSD. Como conseqüência, as agregações permanecem inalteradas pela expiração dos dados do nível inferior e, em adição, após a expiração, ainda existe informação suficiente para que estas agregações permaneçam consistentes com relação a alterações nos provedores.

2.2.3 Componente de Análise e Consulta

O componente de análise e consulta garante o acesso às informações armazenadas no *data warehouse* aos usuários de SSD e aos programas aplicativos que participam do ambiente de *data warehousing*. Oferece, desta forma, funcionalidades relacionadas à consulta e à análise dos dados armazenados, incluindo a habilidade de se determinar a origem dos dados sendo examinados. Essa seção descreve cada uma destas funcionalidades em maior nível de detalhe.

2.2.3.1 Consultando e Analisando o *Data Warehouse* através de Ferramentas

O principal propósito de um ambiente de *data warehousing* consiste em disponibilizar informação integrada aos usuários de SSD para a tomada de decisão estratégica. Tais usuários interagem com o ambiente através de ferramentas dedicadas à análise e consulta dos dados, as quais devem oferecer facilidades de navegação e de visualização. Em especial, estas ferramentas devem permitir que informações relevantes ao contexto de tomada de decisão sejam derivadas a partir da detecção de análise de tendências, da monitoração de problemas e de análises competitiva e comparativa.

Dentre os principais tipos de ferramentas de acesso existentes, pode-se citar: ferramentas de consulta gerenciáveis e geradores de relatório, ferramentas para sistemas de informações executivas, ferramentas OLAP e ferramentas de mineração de dados [CR97].

Ferramentas de consulta gerenciáveis e geradores de relatório são os tipos mais simples de ferramentas e, em geral, não são voltadas especificamente ao ambiente de *data warehousing*. Geradores de relatório, como o próprio nome diz, têm como principal objetivo produzir relatórios periódicos. Já ferramentas de consulta gerenciáveis oferecem aos usuários visões de negócio específicas ao domínio dos dados armazenados e permitem que estes usuários realizem consultas independentemente da estrutura e/ou da linguagem de consulta oferecida pelo banco de dados. Por exemplo, uma ferramenta deste tipo poderia permitir a criação de comandos SQL através da utilização do *mouse*. A saída destas ferramentas é geralmente na forma de um relatório.

Ferramentas para sistemas de informações executivas permitem que aplicações de suporte à decisão gráficas e customizadas sejam desenvolvidas, oferecendo aos usuários de SSD uma visão de alto nível do negócio. Esse tipo de ferramenta é caracterizado por utilizar uma visualização gráfica simplificada, representando exceções a atividades normais de negócio ou a regras através de diferentes cores. Tais ferramentas geralmente apresentam capacidades analíticas limitadas, uma vez que são projetadas visando-se o oferecimento de facilidade de uso tanto para a consulta quanto para a análise das informações.

Por outro lado, ferramentas OLAP são caracterizadas por permitir que usuários de SSD sofisticados analisem os dados usando visões multidimensionais complexas e elaboradas, e por oferecer navegação facilitada através dessas visões. Assim, os usuários de SSD podem analisar os dados sob diferentes perspectivas e/ou determinar tendências através da navegação entre diferentes níveis de hierarquias de agregação. Tais ferramentas apresentam os dados de acordo com o modelo multidimensional (seção 2.4.1), independentemente da forma na qual eles estão realmente armazenados. Aplicações típicas de negócio para essas ferramentas incluem o desempenho de vendas de um determinado produto e a sua lucratividade, a efetividade de um programa de vendas ou a campanha de *marketing* e o planejamento de vendas.

De maneira geral, o componente de análise e consulta possui duas funcionalidades básicas: facilitar o acesso aos dados do *data warehouse* e permitir que informações, padrões e tendências de negócio "escondidas" nestes dados sejam descobertas. Em oposição aos tipos de ferramentas acima descritos, os quais enfocam a primeira funcionalidade, ferramentas de mineração de dados permitem que os usuários de SSD explorem e infiram informação útil a partir dos dados, identificando relacionamentos desconhecidos. Mineração dos dados [AMS+96, Fre98] é uma área que se encontra em plena fase de crescimento, podendo ser considerada uma interseção de diversas outras áreas, como banco de dados, inteligência artificial e estatística. Segundo [Fay98], mineração de dados refere-se à aplicação de algoritmos específicos para a extração de padrões dos dados, a qual consiste em uma das etapas do processo de descobrimento de informação (conhecido como KDD – *Knowledge Discovery in Databases*). A qualidade do conhecimento descoberto por um algoritmo é altamente dependente da aplicação e tem um aspecto subjetivo inerente.

Independentemente da ferramenta de acesso aos dados utilizada, 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 o problema inteiro e/ou a solução sejam claramente visíveis. Por exemplo, padrões podem ser muito mais facilmente detectados se forem expressos graficamente, melhor do que através de simples tabelas. Em especial, técnicas de visualização devem oferecer interação com os usuários de SSD, os quais devem ser capazes de alterar tanto o tipo de informação sendo analisada quanto o método de apresentação sendo utilizado (como histogramas, mapas hierárquicos e gráficos de dispersão).

Uma lista de ferramentas voltadas à análise e consulta dos dados do *data warehouse* pode ser encontrada em [Gre]. Por outro lado, Pendse e Creeth [PC] apresentam informações sobre aproximadamente 30 produtos OLAP, além de relatórios atualizados contendo novidades, dados sobre o mercado e estudos de caso relacionados a esse tema.

2.2.3.2 Identificando a Origem dos Dados Analisados

Uma questão de grande interesse em aplicações OLAP e de mineração dos dados é a habilidade de se determinar o conjunto de dados origem que produz pedaços específicos da informação. Por exemplo, um usuário de SSD 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 a confiabilidade dos provedores de informação ou até mesmo de corrigir os dados correspondentes desses provedores.

O problema de identificar o conjunto exato de itens de dado das relações base que produzem um determinado item de dado de uma visão materializada e o processo pelo qual ele foi produzido é conhecido como problema de linhagem dos dados. Segundo Cui e Widom [CW00], alguns sistemas de *data warehousing* comerciais disponibilizam a linhagem de dados em nível de esquema, ou oferecem facilidades específicas para a manipulação das visões multidimensionais (tal como *drill-down*). Em contrapartida, o trabalho de Cui e Widom, descrito em maior nível de detalhamento em [CWW97], propõe algoritmos para a determinação da linhagem de dados em nível de instância.

Os algoritmos consideram que tanto as visões materializadas quanto os itens de dado base estão armazenados de acordo com o modelo relacional [EN00]. Em adição, as visões materializadas podem ser complexas, definidas em termos dos operadores da álgebra relacional de seleção, de projeção, de junção, de diferença entre conjuntos e de união de conjuntos, além de

um operador adicional de agregação. Essas visões podem tanto armazenar valores de instâncias duplicados quanto possuir uma semântica de conjunto (sem duplicações). Baseado na definição da visão e nos dados base, além de algumas informações auxiliares adicionais, os algoritmos determinam os dados que derivaram o dado da visão materializada.

A definição da visão é expressa na forma de uma árvore de consulta, na qual os nós folhas representam as relações base e os nós internos representam os operadores. Inicialmente, os algoritmos transformam a árvore de consulta da visão em uma forma canônica composta de seqüências de segmentos. Tais segmentos podem ser de dois tipos: segmentos AUSPJ (descritos em termos dos operadores de agregação, de união, de projeção, de seleção e de junção) e segmentos D (descritos em termos do operador diferença). Se a visão é definida como um único segmento, a linhagem dos dados definidos por este segmento é determinada através da execução de consultas relacionais baseadas na forma canônica sobre as relações dos provedores, chamadas de consultas de descoberta, as quais são parametrizadas pelos dados sendo analisados. Neste caso, resultados intermediários não são necessários. Em contrapartida, caso a visão seja especificada por vários segmentos, os algoritmos dividem estes segmentos, definem uma visão intermediária para cada segmento e percorrem recursivamente a hierarquia de visões intermediárias de acordo com a abordagem *top-down*. Em cada nível da hierarquia de visões intermediárias, consultas de descoberta para um segmento único são utilizadas para determinar a linhagem dos dados correntes sendo analisados em relação às visões dos níveis subjacentes até atingir as relações base. Pode-se observar que, neste caso, visões intermediárias são necessárias.

Em geral, as visões intermediárias podem ser tanto armazenadas quanto calculadas a partir das tabelas base quando necessário. Dentro do contexto de *data warehousing*, os algoritmos acima descritos optam pela materialização dessas visões auxiliares, uma vez que os mecanismos de atualização incremental geralmente materializam visões intermediárias, as quais são as mesmas necessárias ao problema de linhagem dos dados. Sob este aspecto, o problema de linhagem de dados está diretamente relacionado ao processo de atualização dos dados do *data warehouse*, uma vez que a consistência das visões intermediárias com relação aos dados dos provedores deve ser garantida. Visando-se um aumento de desempenho, os algoritmos propostos também armazenam cópias das tabelas dos provedores.

2.2.4 *Data Marts*

Um *data mart* consiste na implementação de um *data warehouse* no qual o escopo do dado é limitado, quando comparado ao *data warehouse* propriamente dito. Entretanto, os dados armazenados em *data marts* compartilham as mesmas características que os dados do *data warehouse*, ou seja, são orientados a assunto, integrados, não-voláteis e históricos, além de serem organizados segundo diferentes níveis de agregação.

Um *data mart* pode ser tanto definido como um subconjunto dos dados do *data warehouse* quanto considerado uma política no projeto de construção de um *data warehouse* corporativo.

Data marts podem ser considerados subconjuntos dos dados do *data warehouse* uma vez que possuem cópias replicadas de porções de dados dessa base de dados e são projetados para atender às necessidades específicas de grupos de usuários de SSD dedicados. Por exemplo, um *data mart* departamental pode conter dados departamentais altamente agregados, os quais são utilizados pelos usuários de SSD do departamento em questão para a tomada de decisão local. Tais *data marts* são chamados de *data marts* dependentes, pois independentemente da tecnologia utilizada em suas implementações e do número de *data marts* existentes, grupos de usuários distintos acessam visões de informação derivadas da mesma visão integrada dos dados. Segundo

Moeller [Moe01], a existência de *data warehouses* dependentes em uma corporação representa uma conseqüência natural do volume crescente de informações armazenadas no *data warehouse*.

A criação de *data marts* como subconjuntos do *data warehouse* está relacionada a questões de desempenho, de simplicidade de entendimento e de descentralização de acesso. Como o número de usuários de SSD concorrentes e o volume de dados armazenado em um *data mart* são usualmente menores do que o número de usuários de SSD concorrentes e o volume de dados armazenados no *data warehouse*, o tempo de resposta deste componente tende a ser menor. Em adição, o escopo limitado do *data mart* simplifica o entendimento de seu projeto e, conseqüentemente, a sua manutenção. Por fim, uma vez que grupos de usuários são capazes de manipular de forma autônoma as suas porções de dados de interesse, ocorre uma descentralização de acesso aos dados do *data warehouse*.

Por outro lado, em uma grande corporação, *data marts* 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 complexo e os custos envolvidos são altos, *data marts* são construídos paulatinamente e, à medida que estes se consolidam, inicia-se a construção do *data warehouse* global. De maneira geral, tais *data marts* representam soluções fragmentadas de porções de negócio da empresa, e são chamados de *data marts* independentes. Diferentemente dos *data marts* dependentes, os quais usam o *data warehouse* como fonte de dados, *data marts* independentes obtêm seus dados diretamente a partir dos provedores de informação.

A utilização de *data marts* independentes tende, inicialmente, a reduzir problemas financeiros, uma vez que a construção desses *data marts* exige recursos monetários inferiores do que os despendidos com a construção de um *data warehouse* corporativo e que usuários de SSD são capazes de reconhecer o valor e a potencialidade da solução de *data warehousing* em um período menor de tempo. Entretanto, em longo prazo, a criação de *data marts* independentes pode conduzir a problemas de integração e de escalabilidade, caso um modelo de negócio completo não seja desenvolvido. Cada *data mart* independente pode assumir formas diferentes de consolidar seus dados. Como conseqüência, os dados através dos diversos *data marts* podem ser inconsistentes. Problemas de escalabilidade, por sua vez, ocorrem em situações nas quais o *data mart* inicial de tamanho pequeno e projeto simplificado tende a crescer, tanto no volume de dados armazenado quanto no número de usuários concorrentes que o utilizam.

2.2.5 Metadados

Dados de nível mais baixo podem ser descritos por dados de nível mais alto graças ao conceito de metadados. Metadados consistem em uma abstração dos dados, e permitem que dados armazenados nos mais diferentes formatos tenham significado. Por exemplo, metadados associados a uma seqüência de 0's e 1's devem indicar se tais caracteres representam palavras, ou números, ou dados estatísticos sobre vendas da empresa, ou ainda informações sobre a distribuição populacional do país [APT96].

O armazenamento de metadados no ambiente de *data warehousing* possui um nível de importância elevado, uma vez que os usuários de SSD necessitam conhecer a estrutura e o significado dos dados no processo de busca por fatos não usuais sobre o negócio. Em outras palavras, metadados constituem-se no principal recurso para a administração dos dados nesse ambiente, sendo utilizados durante a criação, a manutenção, o acesso e o gerenciamento do *data warehouse*. Metadados possuem característica temporal, desde que refletem tanto dados

históricos mantidos no *data warehouse* quanto alterações estruturais do *data warehouse* ao longo do tempo.

Em geral, uma grande variedade de metadados precisa ser armazenada, visando a utilização efetiva do ambiente de *data warehousing*. Wu e Buchmann [WB97] dividem os metadados em três categorias:

- **metadados administrativos**: contêm informações relacionadas à construção e à utilização do *data warehousing*, tais como os esquemas dos provedores de informação e do *data warehouse*, além dos mapeamentos existentes entre os diversos esquemas; regras de extração, de tradução, de limpeza e de atualização dos dados, em adição às regras de mapeamento utilizadas para a solução de problemas de heterogeneidade existentes entre os dados dos diversos provedores de informação que participam do ambiente; especificações sobre grupos de usuários e privilégios a eles associados, incluindo políticas de controle de acesso, autorização e perfis; ferramentas de integração e manutenção, e regras associadas aos processos envolvidos; ferramentas de análise e consulta; consultas, agregações e relatórios pré-definidos;
- **metadados específicos da aplicação:** incluem um conjunto de terminologias específicas ao domínio da aplicação, além de restrições da aplicação e outras políticas; e
- **metadados de auditoria:** mantêm informações relacionadas à linhagem dos dados, à geração de relatórios de erros, às ferramentas de auditoria empregadas e às estatísticas de utilização do ambiente de *data warehousing*, incluindo dados sobre a freqüência das consultas, os custos para se processar uma determinada consulta, o tipo de acesso aos dados e o desempenho do sistema.

Já Campos e Rocha Filho [CR97] argumentam que existem, de um modo geral, três camadas de metadados em um ambiente de *data warehousing*. A primeira camada, denominada de **metadados operacionais**, define a estrutura dos dados mantidos pelos provedores de informação operacionais. **Metadados centrais ao *data warehouse*** representam a segunda camada, e englobam mapeamentos de como os dados dos provedores de informação são transformados, definições de agregados e relações existentes entre os diferentes níveis de agregação, dentre outros. A última camada, **metadados do nível do usuário**, mantém metadados que mapeiam os metadados do *data warehouse* para conceitos que sejam familiares aos usuários de SSD.

Por outro lado, Campos [Cam99] e Vaduva e Dittrich [VD01] também distinguem metadados de acordo com o seu uso em **metadados de negócio**, requeridos principalmente por usuários de SSD e **metadados técnicos**, produzidos e utilizados pelos administradores do banco de dados ou pelos componentes de *software* que participam do ambiente de *data warehousing*. A categoria de **metadados de negócio** contém documentação específica dos usuários de SSD, dicionários, conceitos e terminologias do negócio, detalhes sobre consultas predefinidas e relatórios, dentre outras informações. Em contrapartida, **metadados técnicos** incluem: definições de esquema e especificações de configuração; informações sobre o armazenamento físico; direitos de acesso; e especificações executáveis tais como regras de transformação e de integração.

Ainda outra distinção refere-se a **metadados descritivos** e **metadados transformacionais** [VD01]. Enquanto que **metadados descritivos** incluem informações relacionadas à estrutura dos provedores de informação, do *data warehouse* e dos *data marts* que compõem o ambiente de *data warehousing*, **metadados transformacionais** incluem informações associadas ao processamento de dados, como exemplo as regras empregadas durante os processos de extração, de tradução, de limpeza, de integração e de agregação dos dados.

Independentemente da taxonomia utilizada, Kimball [Kim98a] explora de forma exaustiva as informações que devem ser armazenadas sobre essa enorme variedade de metadados, considerando desde o processo de leitura dos dados dos provedores de informação até o armazenamento desses dados no *data warehouse*. Por exemplo, informações sobre as agregações incluem as suas definições, os *logs* de modificação dessas agregações e as suas estatísticas de utilização, além de quais agregações são potenciais.

Em um ambiente de *data warehousing*, os metadados são armazenados em um ou mais repositórios de metadados e manipulados por módulos gerenciadores de metadados. Mais detalhadamente, um repositório consiste em um banco de dados voltado ao armazenamento e à recuperação dos metadados. Repositórios representam os componentes integradores da arquitetura do ambiente, e são (logicamente) independentes do *data warehouse*, ainda que a mesma plataforma de SGBD seja utilizada. Já os módulos gerenciadores devem coordenar a utilização dos repositórios de metadados, incluindo o acesso, a forma de armazenamento dos dados e a sua manutenção. Outras funcionalidades importantes a serem oferecidas por módulos gerenciadores de metadados incluem, por exemplo: (i) prover mecanismos de controle de versões que suportem a característica temporal dos metadados; (ii) prover mecanismos de notificação que propaguem todas as alterações nos metadados a ferramentas e usuários interessados; (iii) possibilitar o estabelecimento de relacionamentos entre as informações armazenadas visando-se análise de impacto; e (iv) facilitar o fluxo de metadados entre diferentes repositórios.

Existe atualmente um grande número de ferramentas comerciais especificamente voltadas ao gerenciamento de metadados, além de diversas outras funcionalidades relacionadas à utilização desse repositório incorporadas em ferramentas que auxiliam outros processos do carregamento dos dados. Em [APT96], diversas ferramentas existentes são classificadas com relação à sua principal funcionalidade, ao passo que em [Gre] uma lista de ferramentas *stand-alone* voltadas ao gerenciamento de metadados pode ser encontrada.

O gerenciamento de metadados representa uma área de pesquisa muito pouco desenvolvida, podendo até mesmo ser considerada imatura [Cam99]. Por um lado, existe um grande espectro de metadados, os quais devem ser capturados, armazenados e gerenciados consistentemente visando-se tanto minimizar esforços no desenvolvimento, na manutenção e na administração do *data warehousing* quanto melhorar a extração efetiva da informação a partir do dado. Por outro lado, existe uma grande quantidade de produtos comerciais que alegam gerenciar metadados, mas que possuem características diferentes entre si e, acima de tudo, mantêm seus próprios metadados em formatos proprietários. Assim, um desafio de extrema importância relacionado ao gerenciamento de metadados refere-se ao compartilhamento e à troca de metadados de forma que estes possam ser uniformemente acessíveis pelos usuários de SSD e pelas ferramentas do ambiente de *data warehousing*.

Na prática, o principal problema de integração é a falta de um padrão, o qual produtos comerciais de diferentes fabricantes possam seguir. Uma iniciativa que tem sido adotada atualmente neste sentido diz respeito à definição, à especificação e à implementação de formatos de padrão de troca de metadados e de seus mecanismos de suporte por grupos de vendedores. Dois grupos têm merecido grande destaque: Metadata Coalition (MDC) e Object Management Group (OMG). MDC, atualmente liderado pela Microsoft, é o criador do padrão *Metadata Interchange Specification* (MDIS). Como resultado dessa liderança, MDIS tem sido integrado com o padrão *Open Information Model* (OIM). Mais detalhadamente, OIM enfoca o compartilhamento e o reuso de metadados através do oferecimento de um modelo formal de descrição de tipos de metadados. OIM utiliza como base outros padrões de mercado: UML (*Unified Modeling Language*) como a linguagem formal de especificação, XML (*eXtensible Markup Language*) como o formato para a troca de informações entre repositórios baseados em

OIM, e SQL como a linguagem de extração. Por outro lado, OMG possui como membros especialistas tais como IBM, Oracle e Unisys, e trabalha na especificação do padrão para intercâmbio *Common Warehouse Metadata* (CWM). Assim como OIM, CWM também é fundamentado em outros padrões de mercado, a saber: UML, MOF (*Meta Object Facility*), e XMI (*XML Metadata Interchange*), um padrão baseado em XML para a troca de metadados originado pelo OMG. Uma análise comparativa entre os padrões OIM e CWM pode ser encontrada em [VVS00]. Já em [MO98], [SM00] e [Cam00] podem ser encontradas descrições dos padrões MDIS, OIM e MOF, respectivamente.

Como pode ser observado, a necessidade da criação de um padrão para a representação e a troca de metadados em ambientes de *data warehousing* e a existência de diferentes grupos de especialistas propondo seus próprios padrões representam metas antagônicas. Neste sentido, MDC e OMG juntaram-se recentemente através de um relacionamento corporativo para definir um consenso em padrões [Met99b]. A proposta consiste na unificação dos padrões OIM e CWM. No entanto, segundo Vetterli *et al*. [VVS00], apesar das similaridades existentes entre esses dois padrões, existem diferenças significativas que fazem com que a unificação destes seja uma tarefa difícil.

2.3 *Data Warehousing* Virtual

A principal diferença existente entre um ambiente de *data warehousing* convencional (seção 2.2) e um ambiente de *data warehousing* virtual é a presença física ou não do *data warehouse*. Em um ambiente de *data warehousing* convencional, dados de interesse dos provedores de informação são extraídos, traduzidos, filtrados quando necessário, integrados e finalmente armazenados fisicamente no *data warehouse*. Em contrapartida, um ambiente de *data warehousing* virtual apenas oferece aos usuários de SSD uma visão lógica ou virtual dos dados armazenados nos provedores de informação [CR97, Ken00]. Mais especificamente, nesse último ambiente não existe um *data warehouse* físico que armazena os dados obtidos do ambiente operacional.

Assim, usuários de SSD do ambiente de *data warehousing* virtual podem acessar os dados desejados diretamente a partir dos provedores de informação. Isto é geralmente realizado através do uso de ferramentas de análise e consulta, as quais pesquisam os dados operacionais e exibem os resultados aos usuários como se estes resultados tivessem sido obtidos a partir de um *data warehouse* consolidado. Entretanto, as consultas submetidas ao *data warehousing* virtual são constantemente executadas em dados não trabalhados e não estruturados, os quais não foram eficientemente projetados para consulta ou análise. Em especial, essas ferramentas escondem de seus usuários a complexidade do ambiente operacional e, de acordo com as suas funcionalidades, podem até oferecer visões multidimensionais dos dados.

A Figura 2.3 ilustra a arquitetura básica de um ambiente de *data warehousing* virtual. Nesse ambiente, alguns componentes importantes de um ambiente de *data warehousing* convencional não estão presentes:

- não existem bases de dados específicas (ou seja, *data warehouse* e/ou *data marts*) que integram os dados extraídos, traduzidos e filtrados oriundos do ambiente operacional;
- não existe um componente de integração e manutenção que centraliza os processos executados durante a atividade de carregamento dos dados; e
- não existe um repositório de metadados específico que armazena os metadados referentes ao ambiente de *data warehousing*.

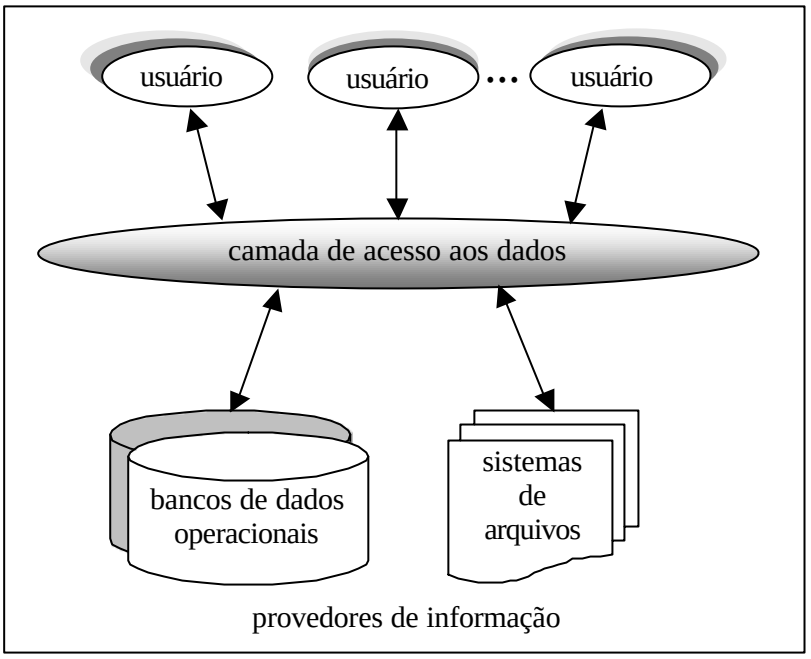

Figura 2.3 Arquitetura básica de um ambiente de *data warehousing* virtual.

Desde que ambientes de *data warehousing* virtuais utilizam os provedores de informação como servidores de banco de dados, pouco investimento monetário precisa ser despendido em *software* e *hardware* adicionais. Em particular, o principal investimento monetário nestes ambientes refere-se às ferramentas de análise e consulta que garantem o acesso aos dados operacionais. Ambientes de *data warehousing* virtuais também são caracterizados por sua simplicidade. Em geral, esses ambientes não oferecem rotinas complexas voltadas à extração, à tradução, à limpeza e à integração dos dados. Além disto, uma vez que o *data warehouse* não está presente na arquitetura desses ambientes, não existe a necessidade de se realizar a atualização periódica dos dados dessa base de dados. Conseqüentemente, o tempo gasto na implementação de um ambiente de *data warehousing* virtual é menor, quando comparado com o tempo de implementação de um ambiente de *data warehousing* convencional [Mim99].

Entretanto, ambientes de *data warehousing* virtuais apresentam diversas desvantagens, as quais contribuem para que o uso desses ambientes seja indicado somente em poucas situações. A seção 2.3.1 destaca as desvantagens de ambientes de *data warehousing* virtuais, com base nas funcionalidades oferecidas por ambientes de *data warehousing* convencionais. Já a seção 2.3.2 identifica situações nas quais pode ser interessante utilizar um *data warehousing* virtual.

2.3.1 *Data Warehousing* Virtual *versus Data Warehousing* Convencional

Ambientes de *data warehousing* virtuais são muito menos robustos do que ambientes de *data warehousing* convencionais. Essa seção compara essas duas abordagens, destacando as desvantagens e as limitações apresentadas por *data warehousing* virtuais. Tal comparação é realizada em termos: (i) das funcionalidades analíticas oferecidas; (ii) da qualidade dos dados manipulados; (iii) da centralização do processo de integração dos dados operacionais; (iv) do desempenho no suporte a consultas OLAP; e (v) da separação existente entre os ambientes operacional e informacional da corporação.

Em primeiro lugar, as funcionalidades analíticas geralmente oferecidas por ambientes de *data warehousing* virtuais são muito mais simples e limitadas do que as funcionalidades analíticas oferecidas por ambientes de *data warehousing* convencionais. Uma vez que os dados manipulados em um ambiente de *data warehousing* virtual são obtidos diretamente a partir do ambiente operacional, estes dados são altamente voláteis, além de não serem orientados a assunto. Ademais, um *data warehousing* virtual raramente manipula dados históricos [Tha99]. Como resultado, análises comparativas complexas e análises de tendência são fortemente prejudicadas, quando não impossíveis de serem realizadas.

Outra desvantagem apresentada por ambientes de *data warehousing* virtuais, quando comparados a ambientes de *data warehousing* convencionais, refere-se à qualidade dos dados. Por um lado, o *data warehouse* mantém os dados sobre o negócio com alta qualidade, principalmente devido ao processo criterioso de limpeza aplicado aos dados durante a atividade de carregamento. Por outro lado, a qualidade dos dados de *data warehousing* virtuais é altamente dependente da qualidade dos dados dos provedores de informação. No entanto, provedores de informação podem armazenar dados duvidosos e/ou incorretos. Em especial, funcionalidades relacionadas à limpeza dos dados são geralmente ausentes em ambientes de *data warehousing* virtuais [Gri01]. Isto significa que a exatidão dos resultados produzidos por esses ambientes em resposta às consultas dos usuários de SSD pode ser questionada.

Não somente a qualidade dos dados é crítica em ambientes de *data warehousing* virtuais, mas também as funcionalidades relacionadas ao processo de integração dos dados operacionais. Como discutido na seção 2.2.2, em *data warehousing* convencionais, estas funcionalidades são oferecidas pelo componente de integração e manutenção. Desde que este componente não é presente na arquitetura de um ambiente de *data warehousing* virtual, ferramentas de análise e consulta desse ambiente enfrentam o desafio de manipular os conflitos de nome, semânticos e estruturais existentes entre os dados dos provedores de informação. É evidente que tais conflitos podem ser resolvidos individualmente por cada uma das ferramentas envolvidas, porém isto gera redundância de esforços e possíveis incompatibilidades.

Em ambientes de *data warehousing* convencionais, os dados do *data warehouse* são organizados visando-se um processamento eficiente de consultas OLAP. Ou seja, um *data warehouse* armazena tanto dados detalhados quanto dados agregados. Em contrapartida, ambientes de *data warehousing* virtuais manipulam apenas dados detalhados, os quais são acessados diretamente a partir dos provedores de informação. Como destacado no início da seção 2.3, os dados operacionais não são estruturados visando-se um bom desempenho no processamento de consultas OLAP. Conseqüentemente, em ambientes de *data warehousing* virtuais, os custos de entrada/saída e de processamento são significativamente maiores, conduzindo a tempos de reposta menos satisfatórios aos usuários de SSD.

Além de comprometerem o desempenho no suporte a consultas OLAP, *data warehousing* virtuais também causam um impacto negativo no desempenho do ambiente operacional. Tal situação é decorrente do fato que em *data warehousing* virtuais não existe uma separação entre os ambientes operacional e informacional da corporação, como ocorre em *data warehousing* convencionais. Mais detalhadamente, consultas OLAP competem com transações OLTP pelos mesmos recursos em *data warehousing* virtuais. Assim, o ambiente operacional também passa a oferecer tempos de reposta menos satisfatórios a seus usuários. Em particular, quanto maior a quantidade de dados que uma corporação manipula, maior a necessidade de se realizar uma separação entre os ambientes operacional e informacional da corporação.

2.3.2 Aplicabilidade do *Data Warehousing* Virtual

Um ambiente de *data warehousing* virtual não oferece as mesmas funcionalidades que um ambiente de *data warehousing* convencional. Isto é decorrente do fato que na arquitetura de um *data warehousing* virtual não estão presentes componentes importantes tais como o *data warehouse* propriamente dito, o repositório de metadados e o componente de integração e manutenção. Assim, *data warehousing* virtuais devem ser utilizados apenas em situações particulares.

A utilização de um ambiente de *data warehousing* virtual pode ser interessante em situações nas quais existe uma demanda pouco freqüente pelos dados. Em especial, esta demanda deve ser composta principalmente por consultas simples, que não afetem de forma demasiadamente negativa o desempenho do ambiente operacional. É importante destacar, entretanto, que a demanda de consultas submetidas a um ambiente de *data warehousing* tende a crescer à medida que os usuários de SSD conscientizam-se da importância desse ambiente. Além disto, usuários de SSD geralmente estão interessados em análises complexas e elaboradas dos dados.

Um ambiente de *data warehousing* virtual também pode ser utilizado como uma estratégia preliminar até que um *data warehousing* convencional seja desenvolvido [Alu96]. Esta situação, em particular, representa a situação mais indicada para o uso deste ambiente. Uma vez que o tempo de implementação de um ambiente de *data warehousing* virtual é reduzido, e o investimento monetário requerido tende a ser pequeno, esse ambiente pode ser utilizado pela corporação com o objetivo de se identificar *se* e *como* os usuários de SSD utilizarão os dados para a tomada de decisão estratégica. Pode-se determinar, por exemplo, qual o público alvo do ambiente de *data warehousing*, qual o escopo dos dados a serem armazenados no *data warehouse* e quais as necessidades de consultas desse público alvo.

Logo após a fase de monitoramento do ambiente de *data warehousing* virtual, um ambiente de *data warehousing* convencional deve ser construído. *Data warehousing* virtuais devem, portanto, ser considerados como soluções temporárias de curta duração. Mais especificamente, *data warehousing* virtuais devem ser substituídos por *data warehousing* convencionais tão rápido quanto possível.

2.4 Modelagem Multidimensional

Em um ambiente de *data warehousing*, as análises efetuadas pelos usuários de SSD representam, de maneira geral, requisições multidimensionais aos dados do *data warehouse*, as quais têm por objetivo a visualização dos dados segundo diferentes perspectivas (dimensões). Sejam as seguintes entidades para o exemplo da cadeia de supermercados: filial, produto, tempo e vendas. No processo de melhoria do desempenho do negócio é necessário examinar os dados sobre as vendas segundo diversas perspectivas. Assim, pode-se examinar o volume de vendas por filial, o volume de vendas por produto, o volume de vendas por tempo, o volume de vendas por filial por produto, etc. Tais análises permitem a identificação de problemas e de tendências, garantindo aos executivos da empresa a possibilidade de formular estratégias efetivas.

O termo dado multidimensional normalmente se refere a dados representando objetos ou eventos que podem ser descritos, e portanto, classificados, por dois ou mais de seus atributos [Wel95, CR97]. Dados que mantêm uma correspondência única entre si, como é o caso de cliente e CPF, não são adequados para serem armazenados de acordo com a representação multidimensional [Pil98]. Na verdade, a determinação de quais dados serão multidimensionais ou não em um ambiente de *data warehousing* depende das consultas a serem realizadas pelos

usuários desse ambiente. Se tais consultas envolvem a recuperação de vários valores e a agregação desses valores, então os dados referenciados por estas consultas devem ser armazenados multidimensionalmente.

Resumindo, o paradigma multidimensional é adequado para modelar a estrutura natural de problemas de suporte à decisão, uma vez que permite a criação de modelos conceituais de negócios. Isto facilita a investigação, o resumo e a organização de dados voltados à análise destes tipos de problemas. Em particular, a visão do negócio do ponto de vista dos usuários de SSD enfoca conjuntos de dados multidimensionais e agregações estatísticas sobre as dimensões destes conjuntos de dados.

A seguir são discutidos os principais aspectos relacionados à modelagem multidimensional. Em particular, são enfocados: os aspectos estáticos e dinâmicos do modelo de dados multidimensional (seção 2.4.1), o armazenamento dos dados do *data warehouse* em sistemas relacionais e multidimensionais proprietários (seção 2.4.2), e alguns trabalhos existentes na literatura voltados à modelagem conceitual dos dados multidimensionais (seção 2.4.3).

2.4.1 Modelo de Dados Multidimensional

A utilização do modelo de dados multidimensional para a modelagem de aplicações de negócio influencia diretamente as ferramentas de análise e consulta que acessam os dados e o projeto do *data warehouse*. Essa seção discute tanto os aspectos estáticos (seção 2.4.1.1) quanto os aspectos dinâmicos (seção 2.4.1.2) do modelo de dados multidimensional, além de introduzir o conceito do cubo de dados (seção 2.4.1.3). Os aspectos estáticos enfocam a modelagem dos dados, enquanto que os aspectos dinâmicos englobam um conjunto de operações básicas que atuam sobre estes dados.

2.4.1.1 Aspectos Estáticos

Os aspectos estáticos do processamento analítico incluem um conjunto de **medidas numéricas**, que são os objetos de análise relevantes ao negócio, e um conjunto de **dimensões**, as quais determinam o contexto para a medida. Uma medida numérica pode ser definida como uma função de suas dimensões correspondentes, representando, desta forma, um valor no espaço multidimensional. Como exemplo, as medidas numéricas unidades-vendidas e número-clientes no *data warehouse* da cadeia de supermercados podem ser determinadas pelas dimensões produto, promoção, filial e tempo.

Medidas numéricas podem ser classificadas em aditivas, semi-aditivas e não aditivas. Uma medida numérica é **aditiva** quando pode ser somada através de todas as suas dimensões. A medida numérica unidades-vendidas é aditiva, uma vez que através de cada combinação de suas dimensões ela pode ser aritmeticamente somada. Em outras palavras, a agregação de unidades-vendidas diárias para unidades-vendidas mensais é realizada somando-se os valores de unidades-vendidas para cada um dos dias que formam o mês. O mesmo é válido para a agregação de unidades-vendidas por produto para unidades-vendidas para todos os produtos, de unidades-vendidas por filial para unidades-vendidas por todas as filiais, e assim por diante.

Medidas numéricas **semi-aditivas**, por outro lado, podem ser somadas somente através de algumas de suas dimensões, enquanto que através de outras o processo aditivo não tem significado algum. Número-clientes não é aditiva através da dimensão produto. Por exemplo, para dois produtos P_1 e P_2 vendidos na mesma filial sob a mesma promoção no mesmo dia, a soma de número-clientes do produto P_1 com número-clientes do produto P_2 não é válida, uma vez

que o mesmo cliente pode estar sendo contabilizado duas vezes. Neste caso, qualquer análise sobre os dados deve ser realizada considerando-se produtos individuais: a agregação de número-clientes do produto P_1 por dia para número-clientes do produto P_1 por mês é correta.

Já medidas numéricas **não aditivas** simplesmente não podem ser somadas. Preço é uma medida não aditiva, uma vez que a agregação de preço diário em preço mensal não é realizada somando-se todos os preços correspondentes aos dias que formam o mês. A agregação de preço por produto por filial em preço por produto para todas as filiais também não implica na soma de todos os preços por produto existentes para cada filial. Como conseqüência, esta medida numérica deve ser calculada utilizando-se média aritmética ou algum outro cálculo mais complexo.

Quanto às dimensões, cada uma delas pode ser descrita por um conjunto de **atributos**. A dimensão produto pode possuir três atributos: marca-produto, categoria e indústria. Assim, o produto "bolacha salgada" da marca "Bolachas & Cia" pertence à categoria "produtos alimentícios" e é produzido pela indústria "Bolachas & Bolachas". Atributos para as demais dimensões são: (i) dimensão promoção: tipo, redução-preço; (ii) dimensão filial: endereço, cidade, estado, região e (iii) dimensão tempo (dia): mês, trimestre, quadrimestre, semestre e ano. Durante toda esta seção, a nomenclatura dia ao invés de tempo será utilizada, para indicar a granularidade diária desta dimensão

Os atributos de uma dimensão podem se relacionar através de **hierarquias de relacionamento**, as quais especificam níveis de agregação e, conseqüentemente, granularidade dos itens de dados. No exemplo anterior, a filial é relacionada aos seus atributos cidade, estado e região através de um relacionamento hierárquico. Assim, filial $\rightarrow$ cidade $\rightarrow$ estado $\rightarrow$ região é uma hierarquia de relacionamento de nível quatro na dimensão filial. Similarmente, dia $\rightarrow$ mês $\rightarrow$ trimestre $\rightarrow$ semestre $\rightarrow$ ano é considerada uma hierarquia de relacionamento de nível cinco para o propósito de agregação, uma vez que dia (1 ... 30/31) = mês; mês (janeiro ... março) = 1° trimestre; mês (abril ... junho) = 2° trimestre; mês (julho ... setembro) = 3° trimestre; mês (outubro ... dezembro) = 4° trimestre; trimestre (1°, 2°) = 1° semestre; trimestre (3°, 4°) = 2° semestre; semestre (1°, 2°) = ano. Em especial, podem existir várias hierarquias de relacionamento ao longo de uma dimensão. Outra hierarquia para a dimensão dia é dia $\rightarrow$ mês $\rightarrow$ quadrimestre $\rightarrow$ ano.

Por fim, alguns atributos das dimensões contêm apenas informações adicionais sobre outro atributo da dimensão e, portanto, não podem ser utilizados em agregações. É o caso do atributo endereço (endereço da filial) da dimensão filial: ele tem sentido se relacionado apenas com filial.

2.4.1.2 Aspectos Dinâmicos

Uma característica essencial do modelo de dados para OLAP é a ênfase em agregações de medidas por uma ou mais dimensões, representando as requisições multidimensionais dos usuários de SSD e estabelecendo a organização dos dados do *data warehouse* segundo diferentes níveis de agregação (seção 2.1.3). Operações analíticas típicas incluem *drill-down*, *roll-up*, *slice and dice*, *pivot* e *drill-across*.

Drill-down consiste no processo de analisar os dados em níveis de agregação progressivamente mais detalhados, ou de menor granularidade. Como exemplo, um usuário pode iniciar sua análise em um alto nível de agregação (unidades-vendidas por marca-produto por ano) e sucessivamente detalhar sua análise através de atributos mais específicos das hierarquias de relacionamento das dimensões (unidades-vendidas por marca-produto por semestre, a seguir, unidades-vendidas por marca-produto por trimestre, a seguir, unidades-vendidas por produto por

mês, a seguir, unidades-vendidas por produto por dia). Esta operação é extremamente útil, uma vez que os usuários de SSD geralmente iniciam as suas análises em níveis altos de agregação e procuram detalhes mais específicos à medida que a identificação da origem de problemas encontrados é necessária.

Por outro lado, a operação *roll-up* representa o processo inverso da operação *drill-down*, possibilitando que a análise dos dados seja realizada em níveis de agregação progressivamente menos detalhados, ou de maior granularidade. Assim, usuários de SSD poderiam requisitar as seguintes visões: unidades-vendidas por produto por dia, a seguir, unidades-vendidas por produto por mês, a seguir, unidades-vendidas por marca-produto por trimestre, a seguir, unidades-vendidas por marca-produto por semestre, e finalmente, unidades-vendidas por marca-produto por ano.

Já a operação *slice and dice* permite que os usuários de SSD restrinjam os dados sendo analisados a um subconjunto destes dados. *Slice* refere-se ao "corte" através de uma ou mais dimensões para um valor fixo, enquanto que *dice* está relacionado à seleção de algumas faixas de valores para as dimensões remanescentes. Assim, a partir da visão multidimensional unidades-vendidas por produto por filial por dia, usuários podem aplicar a operação *slice and dice* para visualizar o subconjunto dos dados contendo as unidades-vendidas para o produto P_3 (*slice*) nas filiais F_1 e F_2 nos dias de D_4 a D_8 (*dice*).

Diferentes perspectivas dos mesmos dados podem ser obtidas pela operação *pivot*, a qual reorienta a visão multidimensional dos dados. Esta operação altera a ordem das dimensões, modificando, conseqüentemente, a orientação sob a qual os dados são visualizados. Através da utilização desta operação, usuários de SSD podem gerar qualquer combinação das dimensões de sua visão multidimensional, investigando desta forma diferentes relacionamentos que podem existir entre os dados. Aplicando-se a operação *pivot* à visão multidimensional unidades-vendidas por produto por filial por dia descrita acima, o usuário pode visualizar os seus dados de interesse através de diferentes combinações equivalentes: unidades-vendidas por produto por dia por filial, unidades-vendidas por dia por produto por filial, e assim por diante. Em relatórios bidimensionais, por exemplo, a operação *pivot* permite que as dimensões representadas em cabeçalhos de colunas e linhas sejam trocadas entre si em combinações arbitrárias, ou ainda permite mover uma das dimensões da linha em uma dimensão da coluna (Figura 2.4).

A operação *drill-across* compara medidas numéricas distintas que são relacionadas entre si através de pelo menos uma dimensão em comum. Dimensões em comum devem ser exatamente as mesmas, ou seja, devem possuir o mesmo significado e devem ter as mesmas restrições definidas para os seus atributos. Este princípio de projeto é obviamente satisfeito quando as dimensões em comum são, na verdade, compartilhadas entre as medidas numéricas sendo consideradas. Por exemplo, as três dimensões em comum entre a medida numérica unidades-vendidas determinada pelas dimensões produto, filial, dia e promoção e entre a medida numérica quantidade-enviada determinada pelas dimensões dia, produto, filial e fabricante podem ser exatamente as mesmas. Conseqüentemente, essas duas medidas numéricas podem ser combinadas e/ou comparadas entre si através da operação *drill-across*. Mesmo que as dimensões comuns difiram entre si pelo nível de granularidade, a realização desta operação ainda é totalmente plausível. Desde que os atributos da dimensão de maior granularidade sejam corretamente construídos a partir de agregações da dimensão de menor granularidade, a operação *drill-across* pode ser realizada baseada somente em atributos que existam em ambas as versões das dimensões.

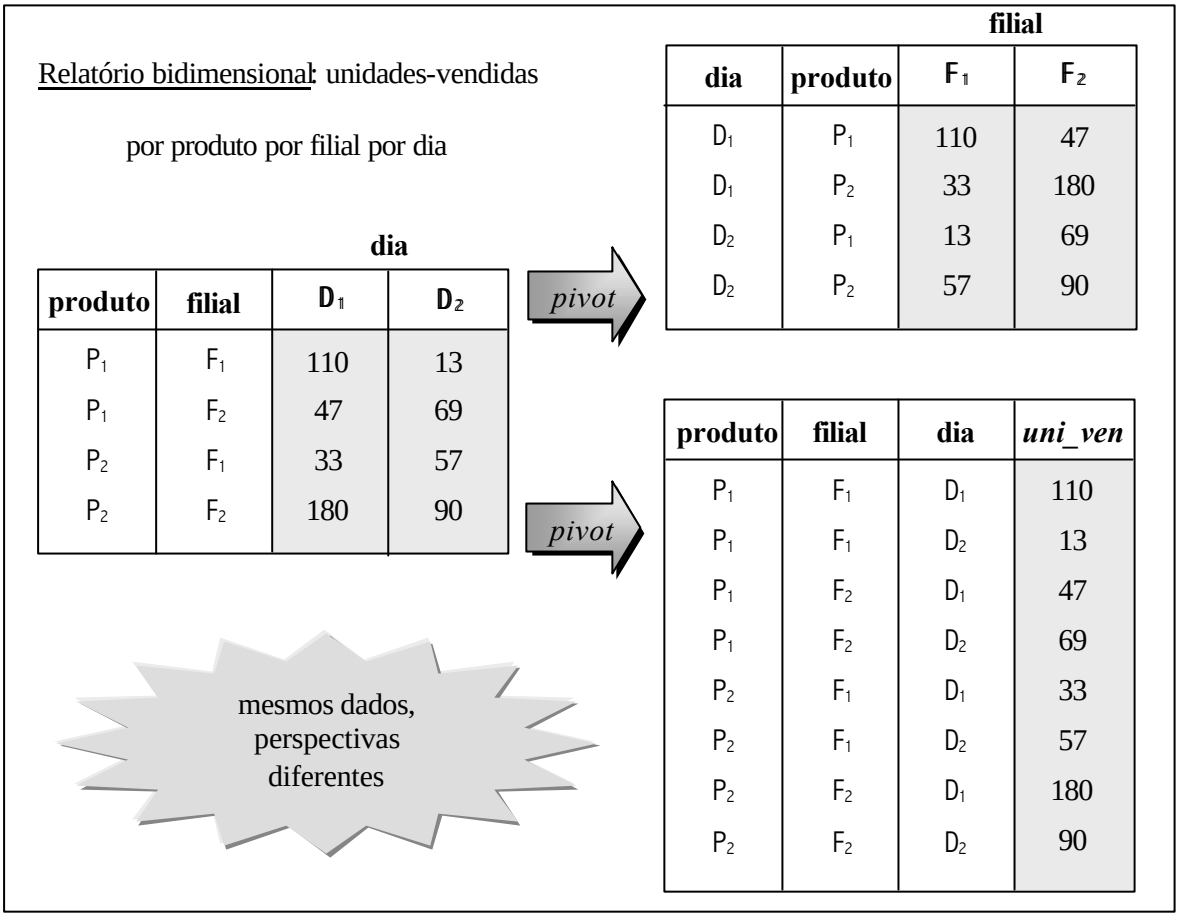

Figura 2.4 Exemplos da operação *pivot* em um relatório bidimensional.

2.4.1.3 Cubo de Dados Multidimensional

Consultas dos usuários de SSD ao ambiente de *data warehousing* são geralmente longas e complexas. Assim sendo, como descrito na seção 2.2.3.1, é importante que tais usuários sejam capazes de analisar os dados do *data warehouse* através de visões multidimensionais elaboradas, podendo navegar através destas visões de forma facilitada. Dentro deste contexto, o cubo de dados consiste em uma representação gráfica de grande utilidade para apresentar as visões dos usuários no espaço multidimensional [Sho97].

A Figura 2.5 mostra um cubo tridimensional, no qual os eixos x, y e z denotam, respectivamente, as dimensões produto, filial e dia. Os valores em cada célula do cubo representam medidas numéricas de interesse. Desta forma, o valor na coordenada (P_2,F_5,D_1) representa o valor agregado da medida numérica quando a dimensão produto tem valor P_2, a dimensão filial tem valor F_5 e a dimensão dia tem valor D_1. Do ponto de vista do usuário de SSD, é muito mais intuitivo visualizar a sua visão multidimensional (no caso, unidades-vendidas por produto por filial por dia) através deste cubo de três dimensões. Cubos que possuem mais do que três dimensões são chamados de hipercubos.

Na verdade, a semântica subjacente ao cubo de dados multidimensional permite não somente a visualização de valores de coordenadas específicas, mas também a identificação das várias agregações que podem ser originadas ao longo de todas as dimensões sendo consideradas. Por exemplo, a Figura 2.6, adaptada de Gray *et al*. [GBLP96], mostra o conceito de agregação (função de agregação = *soma*) para cubos de dados de até três dimensões, em termos de pontos, linhas, planos e cubos. Nessa figura, um cubo 0-dimensional é considerado um ponto, ao passo

que um cubo unidimensional é representado por uma linha com um ponto. Já um cubo bidimensional é especificado por uma tabela cruzada, ou seja, um plano, duas linhas e um ponto. Por fim, um cubo tridimensional pode ser visualizado como um cubo com três tabelas cruzadas que se intersectam. Um diagrama que tem sido comumente utilizado para representar logicamente o cubo de dados multidimensional é o grafo de derivação (seções 2.5.1.1 e 5.1.2).

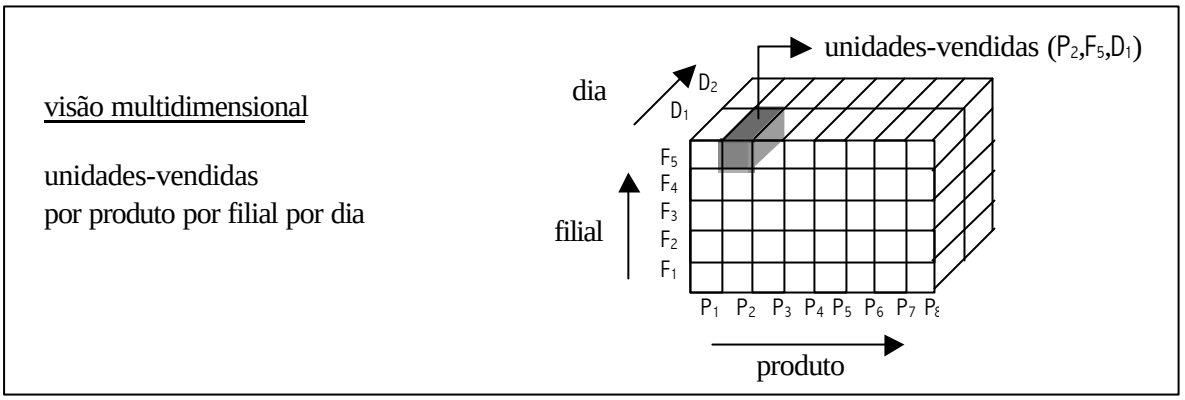

Figura 2.5 Cubo de dados tridimensional.

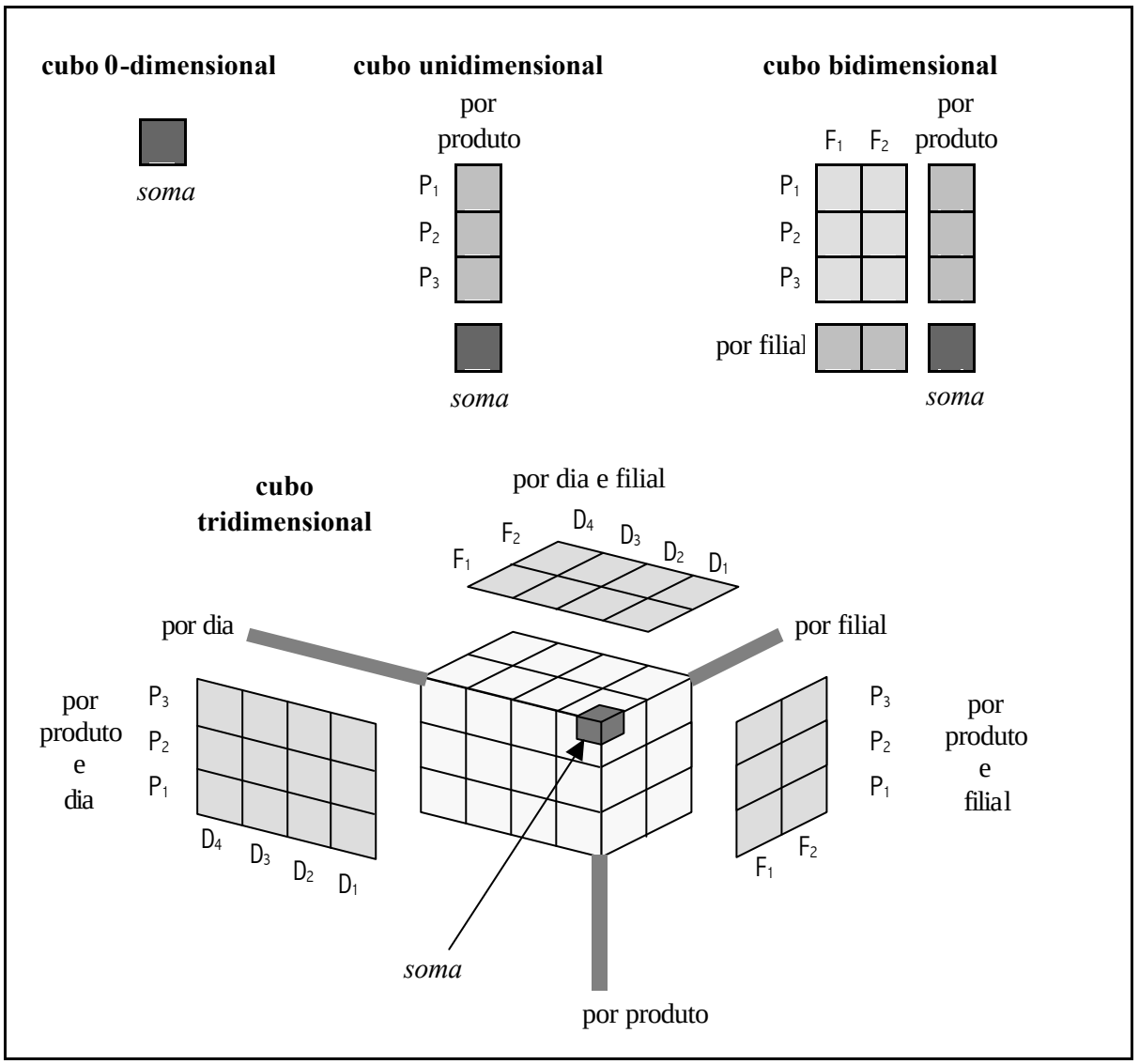

Figura 2.6 Representação do conceito de agregação para cubos de dados.

O cubo de dados também é uma interface natural para representar certas operações analíticas, tais como *pivot* e *slice and dice* (Figura 2.7).

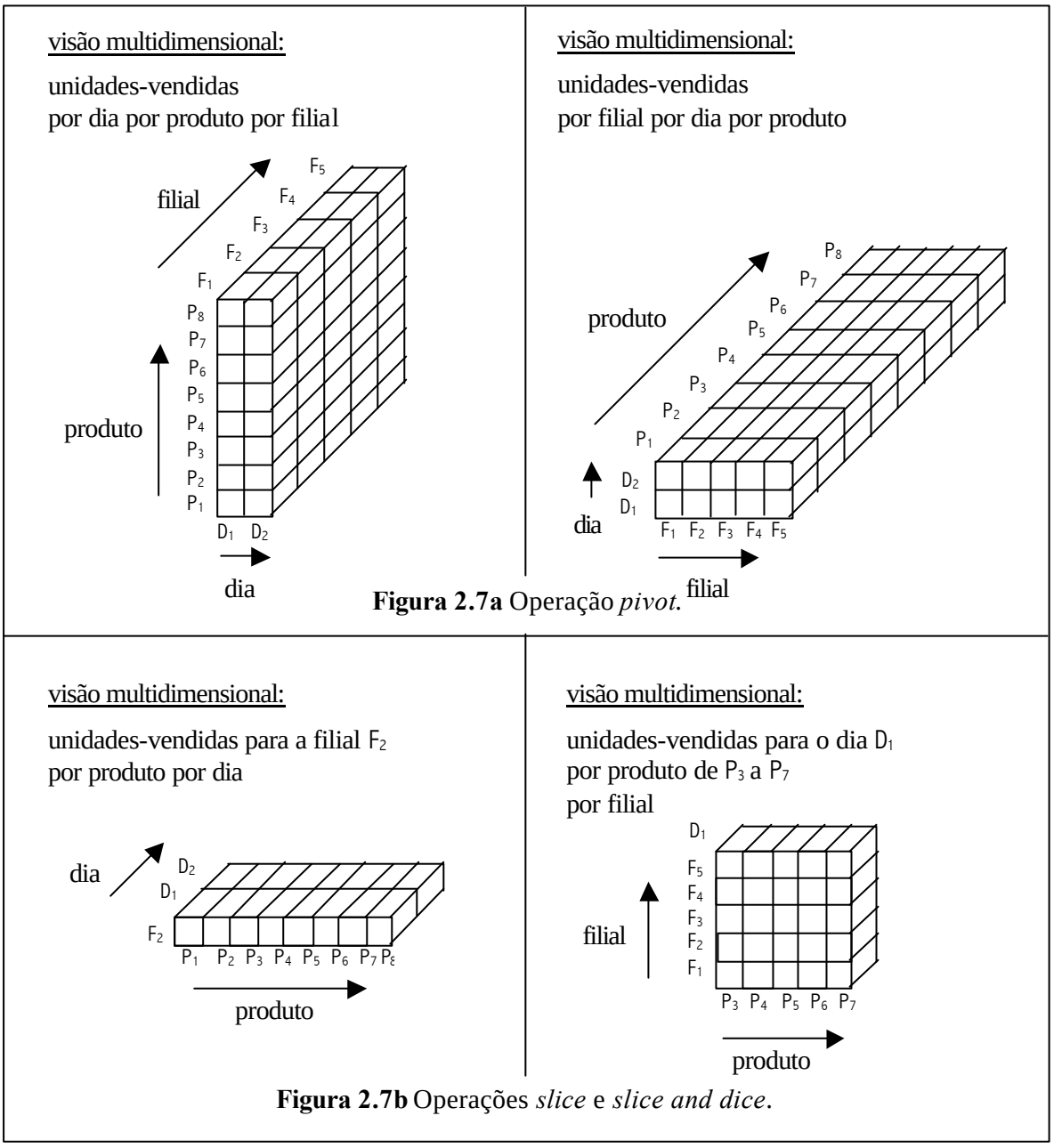

Figura 2.7a Operação *pivot*.

Figura 2.7b Operações *slice* e *slice and dice*.

Figura 2.7 Operações analíticas sobre o cubo de dados da Figura 2.5.

Por permitir que ambas as visões multidimensionais dos usuários de SSD e as operações analíticas sejam facilmente visualizadas através de sua estrutura, a perspectiva multidimensional do cubo de dados é referenciada como modelagem multidimensional. No entanto, o cubo de dados somente pode ser utilizado para expressar os conceitos desta modelagem. Em bancos de dados reais, esta estrutura deve ser mapeada em relações (em sistemas de banco de dados relacionais: seção 2.4.2.1) ou matrizes (em bancos de dados multidimensionais: seção 2.4.2.2).