CENTRO UNIVERSITÁRIO FEEVALE

EDMILSON J. W. FELBER

PROPOSTA DE UMA FERRAMENTA OLAP EM UM *DATA MART* COMERCIAL: UMA APLICAÇÃO PRÁTICA NA INDÚSTRIA CALÇADISTA

Novo Hamburgo, junho de 2006.

EDMILSON J. W. FELBER

PROPOSTA DE UMA FERRAMENTA OLAP EM UM *DATA MART* COMERCIAL: UMA APLICAÇÃO PRÁTICA NA INDÚSTRIA CALÇADISTA

Centro Universitário Feevale
Instituto de Ciências Exatas e Tecnológicas
Curso de Ciência da Computação
Trabalho de Conclusão de Curso

Professor orientador: Juliano Varella de Carvalho

Novo Hamburgo, junho de 2006.

Agradecimentos

Agradeço a todos que me apoiaram, principalmente ao meu orientador, Juliano Varella de Carvalho, que acrescentou o seu conhecimento para o enriquecimento desse trabalho.

RESUMO

No contexto econômico atual, onde a competição entre as empresas está cada vez mais relacionada à sua capacidade de transformar informação em conhecimento e este último em decisões e ações de negócio, o *Data Warehouse* (DW) exerce um papel fundamental como ferramenta de apoio aos processos de tomada de decisão. Contudo, para que se possa tirar vantagem dos recursos de informação de forma satisfatória, é cada vez mais imperativo que as empresas otimizem e dêem mais ênfase à qualidade durante o processo de tomada de decisão.

Neste contexto, são empregados ferramentas e conceitos que organizam as informações de negócios das empresas. Dentre estes conceitos e ferramentas, além do DW, destacam-se também, *Data Mart* (DM)*, Business Intelligence* (BI), modelagem dimensional e ferramentas OLAP*,* que constituem os pilares estratégicos dos Sistemas de Apoio a Decisão (SAD).

O presente trabalho apresenta uma proposta de criação de um DM e uma ferramenta OLAP para a análise das informações do departamento comercial de uma indústria calçadista. A construção da solução será baseada em softwares de distribuição gratuita e/ou *Open Source*. Para tanto, o texto do trabalho aborda os conceitos sobre as tecnologias relacionadas.

Por fim, é construído um SAD utilizando as ferramentas *Mondrian e Jpivot* demonstrando como as mesmas podem auxiliar no processo de tomada de decisão em uma indústria calçadista.

Palavras-chave: Sistemas de Informação, Sistema de Apoio à Decisão, *Data Warehouse, Data Mart*, Modelagem Dimensional, OLAP, *Mondrian, Jpivot.*

ABSTRACT

In the current economic context, where companies competition is each time more related to its capacity to transform information into knowledge and after to transform this knowledge in decisions and action of business, Data Warehouse (DW) has a basic paper as tool of support to the processes of decision-making. However, to take off advantage of information's resources of satisfactory form, it's necessary that the companies optimize and give more emphasis in the quality to the process of decision-making.

In this context, tools and concepts are used to organize the business-oriented information of the companies. Amongst these concepts and tools, it can be cited, beyond DW, Data Mart (DM), Business Intelligence (BI), dimensional modeling and OLAP tools, that constitute the strategical pillars of the Decision Support System (DSS).

The present work shows a proposal of a DM creation and a OLAP tool for information's analysis of a shoe factory commercial department. The development of the solution will be based on free and/or Open Source software. So, the work's text approaches the concepts on the related technologies.

Finally, the SAD is developed, using the Mondrian and Jpivot tools showing as this technologies can help the decision taking process in a shoe factory.

Keywords: Information Systems, Decision Support System, Data Warehouse, Data Mart, Dimensional Modeling, OLAP, Mondrian, Jpivot.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
BI	*Business Intelligence*
BD	Banco de Dados
DM	*Data Mart*
DOLAP	*Desktop On-Line Analytical Processing*
DW	*Data Warehouse*
ER	Modelo Entidade-Relacionamento
HOLAP	*Hybrid On-Line Analytical Processing*
J2EE	*Java2 Platform Enterprise Edition*
J2SE	*Java2 Platform Standard Edition*
JSP	*Java Server Pages*
JVM	*Java Virtual Machine*
MDX	*Multidimensional Expressions*
MOLAP	*Multidimensional On-Line Analytical Processing*
OLAP	*On-Line Analytical Processing*
OLTP	*On-Line Transaction Processing*
RDBMS	*Relational Data Base Management System*
ROLAP	*Relational On-Line Analytical Processing*
SAD	Sistemas de Apoio à Decisão
SGBD	Sistema de Gerenciamento de Banco de Dados

SQL	*Structured Query Language*
TI	Tecnologia de Informação
UF	Unidade da Federação
WCF	*Web Component Framework*
WWW	*World Wide Web*
WOLAP	*Web On-Line Analytical Processing*
XML	*eXtensible Markup Language*
XMLA	XML *for Analysis*

SUMÁRIO

INTRODUÇÃO

Desde o início da era computacional, as organizações têm usado os dados de suas bases operacionais para atender as necessidades de informações. Nesta situação, além da dificuldade de encontrá-los, diversas vezes, dados inconsistentes são utilizados como base para tomada de decisões importantes.

Uma das grandes dificuldades das empresas é o gerenciamento dos dados oriundos de diversos sistemas do ambiente operacional. Grandes bases de dados são criadas pelo acúmulo de informações resultantes de operações transacionais, pouco aproveitadas pelos responsáveis pelas tomadas de decisões, devido a complexidade de extração das mesmas. Agrupar essas informações, interpretá-las e tirar conclusões, não é uma tarefa fácil. É preciso extrair de cada base de dados, as informações que realmente interessam e padronizá-las para que possam ser analisadas.

As aplicações para suporte de tomadas de decisão, baseadas em DW, podem tornar mais prática e fácil a exploração dos dados visando uma maior eficácia do negócio. Isto é o oposto de utilizar somente os dados provenientes das aplicações operacionais, os quais ajudam nas operações da empresa, em suas atividades diárias, mas que tornam complexa a obtenção de informações para uso gerencial.

Segundo Inmon (1997, p. 33), [...] "*Data warehouse* é um conjunto de dados baseado em assuntos, integrado, não-volátil, e variável em relação ao tempo, de apoio às decisões gerenciais" [...].

A criação de um DW requer tempo, capital e um considerável esforço gerencial. Por estes motivos, em diversas situações, as empresas iniciam o projeto de DW focando nas necessidades de áreas específicas dentro da organização. Estas estruturas menores de armazenamento de dados são chamadas *Data Marts* (DM). Um DM é um pequeno DW que fornece suporte a um número reduzido de pessoas, com uma visão mais especializada e

limitada dos dados.

Singh (2001, p. 14) define DM como sendo "um subconjunto do *data warehouse* empresa-inteira. Tipicamente, desempenha o papel de um *data warehouse* departamental, regional ou funcional" [...].

Atualmente, diversas empresas optam pelo DM por causa do seu custo mais baixo e do tempo menor de implementação da solução e, além disto, estes podem servir como um laboratório de teste àquelas empresas que desejam investigar e explorar os benefícios do DW.

> *A idéia, via DW, é armazenar os dados em vários graus de relacionamento e sumariação, de forma a facilitar e agilizar os processos de tomada de decisão por diferentes níveis gerenciais. [...] Esses dados, oriundos de sistemas de informação de produção, deverão estar "mastigados", integrados e disponíveis, permitindo diversas formas de consultas, através dos mecanismos amistosos das ferramentas de usuários* (BARBIERI, 2001, p. 51).

Dentre estas ferramentas, destaca-se a tecnologia denominada OLAP - *On-line Analytical Processing*, que se caracteriza por transformar dados em informações eficientes, dinâmicas e flexíveis. O Conselho OLAP[1], apud Inmon et al. (1999, p.175), define esta tecnologia como [...]"categoria da tecnologia de software que permite que analistas, gerentes e executivos obtenham, de maneira rápida, consistente e interativa, acesso a uma variedade de visualizações possíveis de informações que foi transformada de dados puros para refletir a dimensão real do empreendimento do ponto de vista do usuário".

O objetivo de uma ferramenta OLAP é transformar dados em informações capazes de dar suporte a decisões gerenciais de forma flexível e em tempo hábil. Assim, OLAP precisa oferecer informações existentes, oportunas, precisas e inteligíveis (THOMSEN, 2002, p. 8).

Conforme Bispo (1998), as ferramentas OLAP permitem aos usuários analisar os dados em várias dimensões, como região, produto, tempo e vendedor. Cada dimensão também pode conter hierarquias, por exemplo, a dimensão tempo pode conter as hierarquias ano, semestre, mês, semana ou dia. Pode-se navegar livremente de uma hierarquia para outra, até chegar-se na máxima granularidade dos dados.

---

[1] O Conselho OLAP (*OLAP Council)* é uma organização sem fins lucrativos, patrocinada por vários fornecedores de ferramentas OLAP, cujo objetivo é promover a educação sobre a tecnologia OLAP.

Uma das características das soluções proprietárias das ferramentas de análise baseadas em DW / DM é o alto custo da implementação destas soluções, o que, na maioria das vezes, torna proibitivo a sua utilização em pequenas, médias e em certas situações até em grandes empresas.

De acordo com Grimes (2005), este universo está começando a ser descoberto pelo movimento *Open Source*, fato esse que fornecerá ferramentas para que os desenvolvedores de aplicações ofereçam a seus clientes, soluções de gestão de informação com qualidade a um custo razoável. (Tradução nossa).

Considerando as tecnologias apresentadas, este trabalho propõe o desenvolvimento de um SAD composto por um DM e uma ferramenta OLAP para a análise das informações do departamento comercial de uma indústria calçadista, utilizando ferramentas de distribuição gratuita e/ou *Open Source*. Desta forma, espera-se que o mesmo agregue valor à organização, provendo uma solução viável em termos de qualidade e custo para o auxílio à tomada de decisões gerenciais. Nesse trabalho serão feitos o embasamento teórico e a escolha das ferramentas que serão utilizadas no desenvolvimento e implementação da solução.

O desenvolvimento desta solução iniciará com o estudo das tecnologias envolvidas, apresentando fundamentação teórica sobre as mesmas. Na sequência, será feito uma avaliação das ferramentas visando escolher aquelas que serão utilizadas para a implementação da solução. Após a avaliação, será feito um estudo de caso, com a finalidade de modelar o DM comercial que servirá de base para a construção da aplicação para a análise gerencial.

A solução será desenvolvida com a preocupação de ser autoexplicativa ao usuário, utilizando uma nomenclatura tanto de tabelas e colunas, quanto dos cubos, dimensões e medidas, que permita a este um fácil entendimento da estrutura do DM e das informações nele contidas. Com isso, este trabalho não irá implementar uma estrutura de metadados, pois entende-se que, através deste procedimento, o usuário conseguirá facilmente chegar a informação desejada.

Esse trabalho é composto de cinco capítulos. No primeiro e segundo capítulos, são abordados os conceitos que embasam o trabalho. No terceiro capítulo, são descritas as características das ferramentas OLAP, visando escolher aquelas mais adequadas ao desenvolvimento da aplicação. No quarto capítulo é feito o estudo de caso e a modelagem do DM e no quinto capítulo é apresentada a solução proposta.

1 SISTEMAS DE APOIO À DECISÃO (SAD)

Para competir em um mercado globalizado, as empresas necessitam cada vez mais deter conhecimento sobre os seus clientes, mercados, tecnologias e processos. Ter informações disponíveis é uma ferramenta poderosa para quem necessita tomar decisões e, tendo em vista esse princípio, as empresas começaram a extrair dados dos seus sistemas operacionais e armazená-los separadamente em um ambiente com a finalidade de prover informações para o auxílio à tomada de decisões.

Segundo Inmon (1997), os SADs tiveram seu início na década de 1960. Neste período, o processamento era feito em aplicações baseadas em relatórios e programas. Com os anos, na medida em que o volume de dados foi crescendo, a tarefa de análise dos mesmos se tornou bastante complexa e trabalhosa.

Com a crescente evolução da tecnologia e o armazenamento de dados em disco, surgiram os Sistemas Gerenciadores de Bancos de Dados (SGBD), com o objetivo de facilitar o armazenamento e o acesso aos dados. Porém, mesmo com as facilidades trazidas pelos SGBDs, o volume de dados continuava crescendo e, como não havia o devido planejamento no armazenamento das bases de dados, sua compreensão se tornava cada vez mais complexa. Atualmente estes SGBDs armazenam uma quantidade muito grande de dados, mas as ferramentas de análise destes dados não evoluíram na mesma velocidade que a tecnologia utilizada para armazená-los (MAZETTO, 2004).

Todos esses problemas enfrentados no passado, e que hoje ainda desafiam muitos profissionais de Tecnologia de Informação (TI), tem origem no fato de que, tradicionalmente, o foco da tecnologia da computação tem sido em automatizar tarefas rotineiras e melhorar a eficiência de processos existentes (MAZETTO, 2004).

Os sistemas de produção de uma empresa recuperam e atualizam um registro por vez, normalmente atendendo a muitos usuários de forma concorrente, exigindo também um

tempo de resposta muito baixo. Já os SADs, usualmente lidam com poucos usuários por vez e os requisitos em termos de tempo de resposta podem não ser críticos. No entanto, geralmente lidam com consultas complexas, não antecipadas ou previstas, envolvendo grande quantidade de registros referentes aos processos operacionais da empresa (CAMPOS; FILHO 2005).

Os SADs visam atender as necessidades dos executivos de uma organização, quanto à obtenção de informações para a tomada de decisão. O SAD é estruturado de forma a atender os diferentes níveis de conhecimento das pessoas que o acessam. Geralmente, os resultados das pesquisas são demonstrados através de gráficos e tabelas, sendo que as informações apresentadas têm alto índice de valor agregado.

O termo SAD, propriamente dito, tem sido utilizado cada vez menos, tanto em livros e revistas quanto em *Websites* na internet. No seu lugar tem sido cada vez mais freqüente o uso do termo *Business Intelligence* (BI). BI é um conceito empregado a ferramentas, tecnologias e metodologias, que tem como objetivo fornecer informações estratégicas, que apóiam na tomada de decisão.

Segundo Barbieri (2001, p. 5), BI "representa a habilidade de se estruturar, acessar e explorar informações, normalmente guardadas em DW/DM (*Data Warehouse Data Mart*), com o objetivo de desenvolver percepções, entendimentos, conhecimentos, os quais podem produzir um melhor processo de tomada de decisão".

BI é a utilização de uma série de ferramentas para coletar, analisar e extrair informações, que serão utilizadas no auxílio ao processo de gestão e tomada de decisão. A Figura 1.1 mostra os principais componentes deste ambiente.

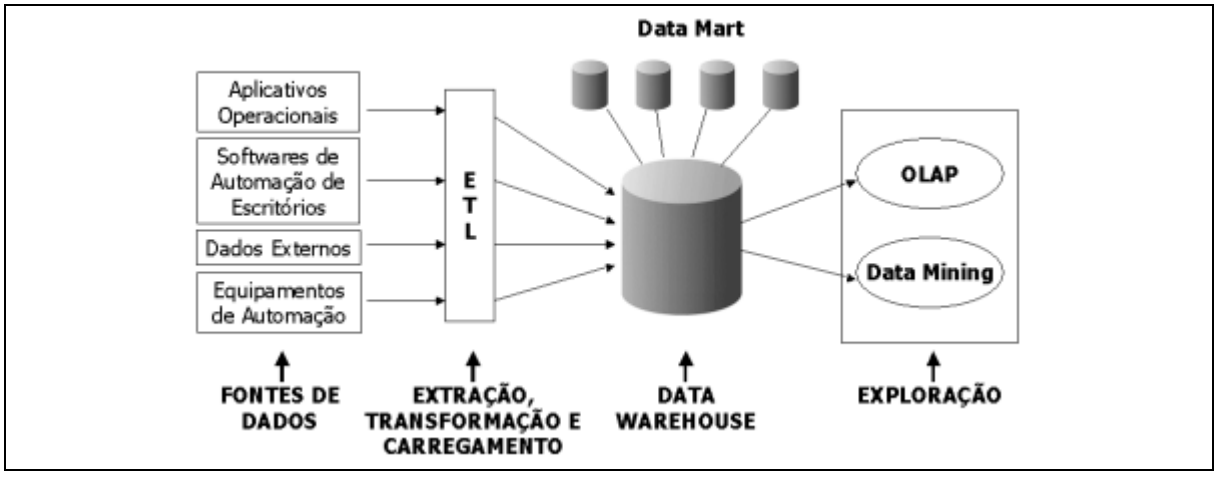


Figura 1.1 *Componentes de um ambiente BI.*
Fonte: Adaptação do autor segundo (BARBIERI, 2001).

O objetivo do BI é transformar não só dados em informações, mas principalmente em conhecimento, suportando o processo decisório, com o objetivo de gerar vantagens competitivas. Este termo foi criado pelo *Gartner Group*[1] nos anos 80 (WIKIPEDIA, 2005).

Após uma melhor compreensão do que representa um SAD e da sua evolução, a seguir serão abordados os conceitos que fazem parte deste universo.

1.1 Data Warehouse (DW)

Conforme Fortulan; Filho (2005), ao longo do tempo, os bancos de dados foram desenvolvidos para fins de processamentos de dados operacionais e analíticos, havendo maior ênfase no primeiro caso, ainda que ambos tivessem usuários com diferentes necessidades. Uma vez compreendida essa diferença, foram criados bancos de dados separados para fins analíticos, chamados de DW.

O DW oferece os fundamentos e os recursos necessários para uma estrutura de BI eficiente, fornecendo dados integrados e históricos que servem desde a alta direção, que necessita de informações mais resumidas, até as gerências de baixo nível, onde os dados detalhados ajudam a observar aspectos mais táticos da empresa.

1.1.1 Características de um DW

Segundo Inmon (1997, p. 33), [...] "*Data warehouse* é um conjunto de dados baseado em assuntos, integrado, não-volátil, e variável em relação ao tempo, de apoio às decisões gerenciais" [...]. Por estas características, de acordo com Singh (2001), entende-se:

Baseado em assuntos: Significa que o DW foca as principais entidades do negócio.

Integrado: Significa que os dados estão armazenados em formato consistente (ou seja, especificando convenções, restrições de domínio, atributos físicos e medições).

Não volátil: Significa que os dados não se alteram depois de incluídos no DW. Os novos dados são adicionados ao DW, integrando-se com àqueles previamente armazenados.

Variante no tempo: Significa que os dados estão associados a um ponto no tempo, como por exemplo, ano, semestre, mês, trimestre, período de pagamento, etc...

---

[1] Consultoria de pesquisas de mercado na área de tecnologia da informação. GARTNER. *Gartner Group*. Disponível em: <http://www.gartner.com>. Acesso em: 26 ago. 2005.

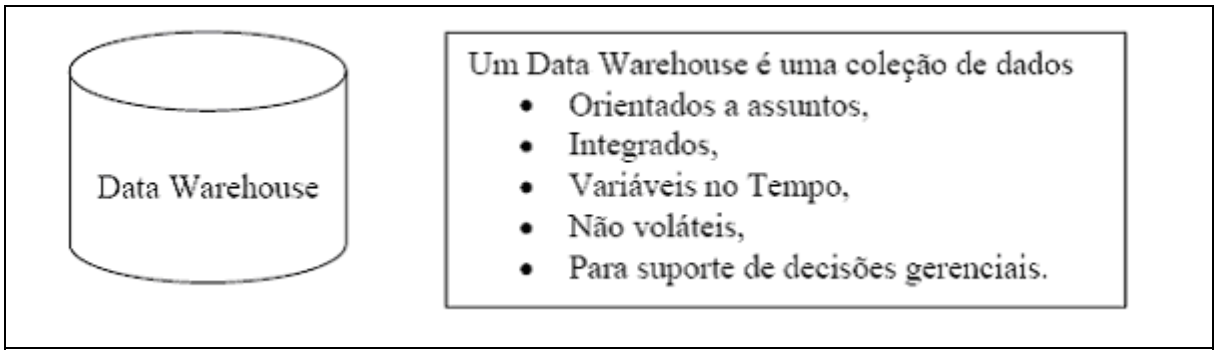


Figura 1.2 *Definição do DW*.
Fonte: (INMON, 1997).

1.1.2 Diferenças entre o ambiente operacional e o ambiente DW

O ambiente operacional é composto por sistemas que têm como finalidade automatizar e suportar as principais atividades do dia-a-dia de uma empresa, e que geram um alto volume de transações. Estes sistemas são chamados de sistemas OLTP (*On-Line Transaction Processing*). Por esta razão, os sistemas que compõem este ambiente devem ter alto desempenho e tempos de respostas baixos, visto que isto causa impacto diretamente na eficiência dos processos da empresa. Os dados operacionais são parte da infra-estrutura corporativa: são detalhados, atualizáveis e não-redundantes (COSTA; ANCIÃES, 2001).

Por outro lado, o ambiente de DW foi projetado para fornecer informações que auxiliem o processo de tomada de decisão. Em geral, os dados informacionais são sumariados e redundantes, suportando diferentes visões sobre eles, além de não serem atualizáveis. O ambiente de DW não difere do ambiente operacional somente quanto ao foco, mas também quanto ao escopo. Os dados transacionais são normalmente focados em áreas específicas, enquanto o ambiente de DW precisa englobar grandes volumes de dados de diferentes áreas da empresa, e transformá-los em dados consistentes para poderem ser utilizados para análise (COSTA; ANCIÃES, 2001). A Figura 1.3 ilustra essa diferença.

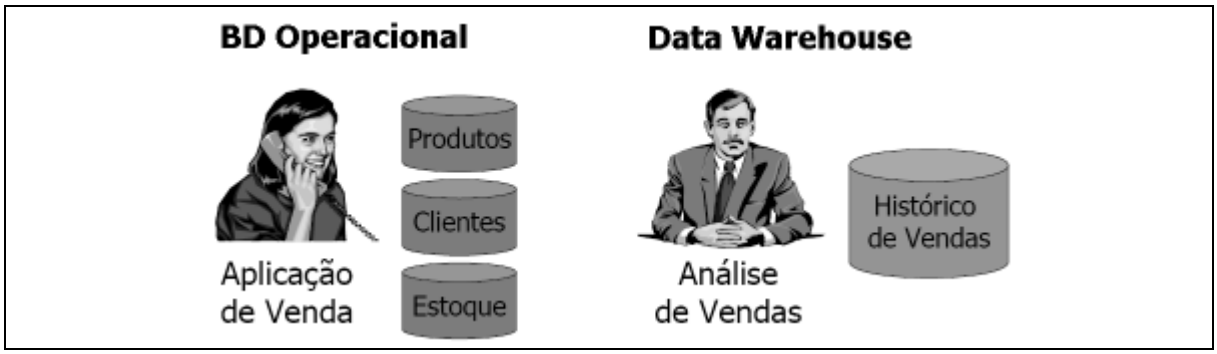


Figura 1.3 *Ambiente operacional X DW*.

O Quadro 1.1 lista as principais diferenças existentes entre o ambiente operacional e DW:

Quadro 1.1 *Diferenças entre o ambiente operacional e DW.*

	Operacional	**DW**
Usuários	Funcionários	Alta Administração
Utilização	Tarefas cotidianas	Decisões estratégicas
Padrão de uso	Previsível	Difícil de prever
Princípio de funcionamento	Baseado em transações	Baseado em análise de dados
Valores dos dados	Valores atuais e voláteis	Valores históricos e imutáveis
Detalhamento	Alto	Sumariado
Organização dos dados	Orientado a aplicações	Orientado a assunto
Conteúdo	Dados correntes, atômicos, isolados	Dados históricos, integrados, sumariados
Natureza dos dados	Dinâmicos, com atualizações contínuas	Estáticos, com atualizações programadas
Tipo de Acesso	Grande volume de inserções, atualizações e consultas	Grande volume de consultas complexas
Tempo de resposta	Em segundos	De segundos a minutos

Fonte: Adaptação pelo autor segundo (Inmon, 1999, p. 14; Singh, 2001, p. 37; Barbieri, 2001, p. 47).

1.1.3 Granularidade

> *A mais importante questão de projeto que o desenvolvedor do data warehouse precisa enfrentar, refere-se à definição da granularidade do data warehouse, ou seja, o nível de detalhe ou de resumo dos dados existentes no data warehouse. Quando a granularidade de um data warehouse é apropriadamente estabelecida, os demais aspectos de projeto e implementação fluem tranqüilamente; quando ela não é estabelecida, todos os outros aspectos se complicam* (INMON, 1997, p. 143).

À medida que o nível de granularidade aumenta, o número de consultas que podem ser atendidas diminui, sendo que em uma granularidade mínima as consultas mais detalhadas podem ser respondidas. Portanto, é necessário encontrar um ponto de equilíbrio. O nível adequado de granularidade deve ser definido de tal forma que atenda as necessidades do usuário, tendo como limitação os recursos disponíveis (HOKAMA et al, 2004).

A Figura 1.4 mostra um exemplo de granularidade diferentes em um mesmo assunto:

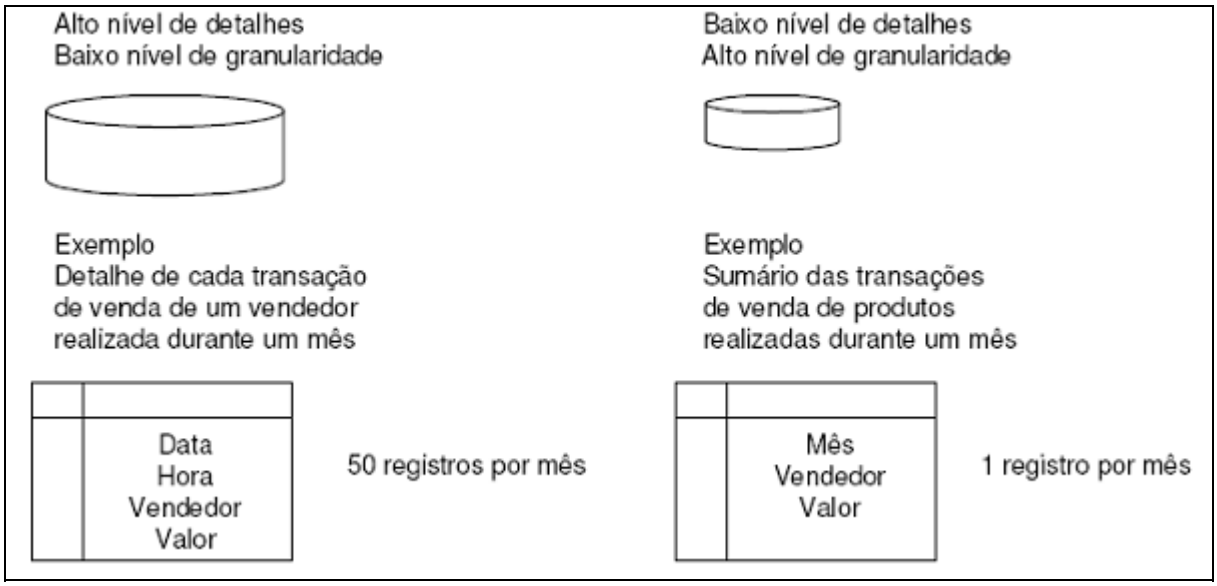


Figura 1.4 *Granularidade dos dados*.
Fonte: (MACHADO, 2000, p. 45).

A definição da granularidade de dados é a etapa mais importante do projeto de um DW, porque ela afeta profundamente o volume de dados que reside no DW e, ao mesmo tempo, afeta o tipo de consulta que pode ser atendida. Devem-se definir níveis adequados de granularidade, de acordo com as necessidades do usuário (MACHADO, 2000).

"A granularidade diz respeito ao nível de detalhe ou de resumo contido nas unidades de dados existentes no *data warehouse*. Quanto mais detalhe, mais baixo o nível de granularidade. Quanto menos detalhe, mais alto o nível de granularidade." (INMON, 1997, p. 45).

Existe a possibilidade de utilizar um nível duplo de granularidade (níveis duais de granularidade). Esta técnica se enquadra nos requisitos da maioria das empresas. São criadas duas camadas: uma camada para os dados levemente resumidos e outra para os dados históricos. Com a criação de dois níveis de granularidade, é possível atender a todos os tipos de consultas, com um melhor desempenho, visto que a maior parte do processamento analítico dirige-se aos dados levemente resumidos que são compactos e de fácil acesso e para as ocasiões em que um maior nível de detalhe deve ser analisado, existe o nível de dados históricos, o qual é complexo e de alto custo (INMON, 1997).

1.2 Data Mart (DM)

Um DM representa um subconjunto de dados do DW. Por ser menor, possibilita a análise multidimensional com uma visão mais especializada e limitada dos dados, visando aumentar a velocidade na consulta das informações, possuindo riscos e prazos menores para a sua implementação.

> *Data Mart é um subconjunto do data warehouse empresa-inteira. Tipicamente, desempenha o papel de um data warehouse departamental, regional ou funcional. Como parte do processo interativo do data warehouse, a empresa pode construir uma série de data marts ao longo do tempo e eventualmente vinculá-los através de um data warehouse lógico empresa-inteira* (SINGH, 2001, p. 14).

Em um DM, os dados podem ser obtidos diretamente dos sistemas de informação operacionais ou do DW da empresa, e as análises também são orientadas para áreas de interesse de uma unidade ou departamento. Os DMs são muito bem aceitos no campo empresarial, pois por suas características exigem menos investimento de infra-estrutura, produzem resultados mais rapidamente e são escaláveis até um DW.

O DM apresenta um custo inferior para o seu desenvolvimento em relação ao do *DW* e pode ser visto como um conjunto de soluções particionadas em termos das áreas de negócios das organizações, de forma que se pode ter DM relativos a áreas como vendas, marketing, finanças e estoque, conforme pode ser observado na Figura 1.5.

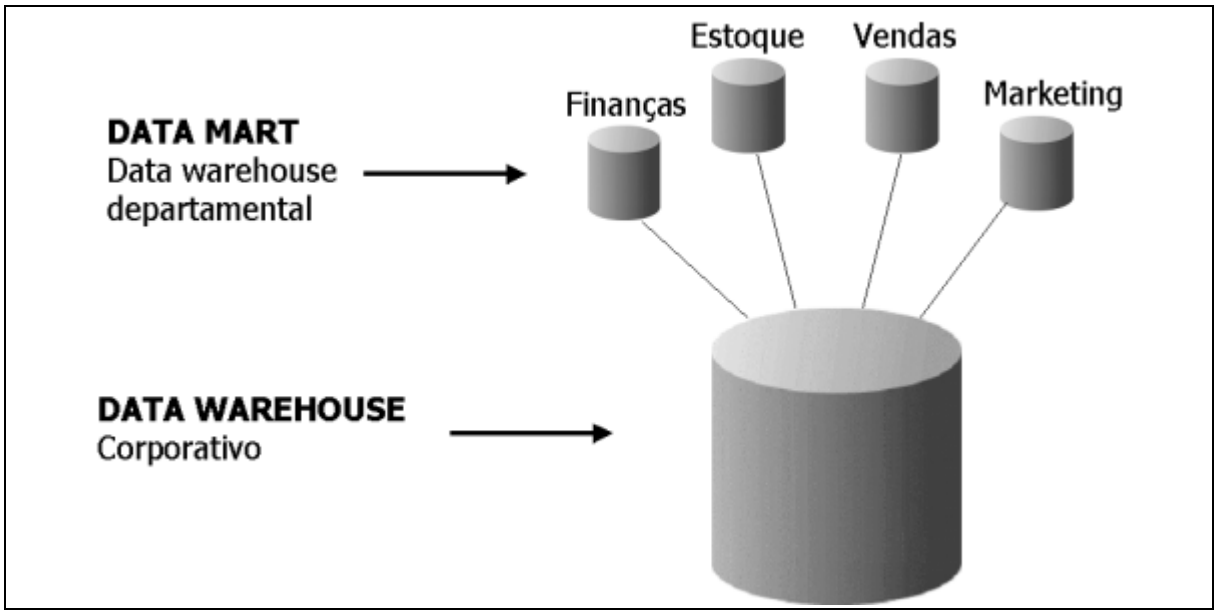


Figura 1.5 *DW X DM*.
Fonte: Adaptação do autor segundo (MACHADO,2000).

1.2.1 Abordagens de construção do DM

A construção do DM pode seguir duas abordagens distintas: *top-down* e *bottom-up.*

Na abordagem *top-down*, os DMs são criados a partir do DW, isto é, primeiramente constrói-se o DW global, para só depois se construir os DMs. Esta abordagem é defendida por *Bill Inmon* (INMON, 1997), considerado um dos precursores do DW. Já na abordagem *bottom-up*, que tem como seu principal defensor *Ralph Kimball* (KIMBALL; ROSS, 2002), os DMs devem ser construídos de forma a representarem as diversas áreas setoriais das organizações e, após a criação deles, parte-se para a construção do DW que deverá representar o conjunto de todas áreas das organizações.

1.2.2 Tipos de arquitetura do DM / DW

Segundo Machado (2000), a escolha da arquitetura é uma decisão gerencial, geralmente baseada nos fatores relativos à infra-estrutura disponível, ao tamanho da empresa, o escopo do projeto e aos recursos disponibilizados para investimento. A seguir são descritas estas arquiteturas.

a) Arquitetura Global: De acordo com Machado (2000, p. 32), esta arquitetura "é considerada como a que suporta toda ou a maior parte das necessidades de um *data warehouse* integrado com grande grau de acesso e utilização das informações para todos os departamentos de uma empresa". Machado (2000) afirma ainda que seu custo de implementação é muito alto, pois este tipo de ambiente consome muito tempo de desenvolvimento e administração. A arquitetura global pode ser fisicamente centralizada ou fisicamente distribuída. Na arquitetura global distribuída, o DW está distribuído em vários locais diferentes, enquanto na arquitetura global centralizada ele se encontra em um único local.

b) Arquitetura de DM Independente: Este tipo de arquitetura é controlado por um grupo determinado de usuários e atende as necessidades específicas e departamentais desse grupo, sem nenhum foco corporativo, ou seja, um DM de um departamento não tem ligação alguma com qualquer outro DM de qualquer outro departamento (MACHADO, 2000).

Essa arquitetura possibilita uma rápida implementação e não têm impacto nos recursos de TI, características que podem ser consideradas como vantagens. Os problemas apresentados por ela são o fato de não possuir integração corporativa e falta de uma visão

global. Além disso, este tipo de DM geralmente está acessível somente ao pessoal do departamento ao qual faz referência (MACHADO, 2000).

c) Arquitetura de DM Integrado: Machado (2000) explica que na arquitetura de DMs Integrados, esses são implementados separadamente por departamentos e integrados, a fim de prover uma visão corporativa. Apesar da maior complexidade de implementação, essa arquitetura apresenta vantagens em relação à arquitetura de DM Independente, como permitir aos usuários de um departamento o acesso e utilização dos dados de um DM de outro departamento e a maior quantidade de funções e informações fornecidas.

1.3 Metadados

Toda e qualquer informação no ambiente DW que não são os dados propriamente ditos, são chamados metadados. Estes são como uma enciclopédia para o DW. Eles estão presentes em uma variedade de formas e formatos para suportar as necessidades desiguais dos grupos de usuários técnicos, administrativos e de negócio do DW (KIMBALL; ROSS, 2002).

Em termos simples, metadado é definido como sendo dado sobre o dado. Ou seja, o metadado descreve ou qualifica outro dado, incorporando, a este, significado. Sem metadado, a informação se restringe a um conjunto de dados sem significado. Através de uma solução eficaz de metadados é possível avaliar o impacto das mudanças nos sistemas transacionais e, portanto, a tarefa de manutenção desses sistemas torna-se menos complexa. Da mesma forma, os metadados auxiliam o processo de construção e manutenção do DW (PEREIRA, 2000).

Singh (2001, p. 78) afirma: “O metadado representa para o *data warehouse* o mesmo que o catálogo de livros representa para a biblioteca. Ele serve para identificar o conteúdo e a localização dos dados no *data warehouse*”.

O motivo central de um SAD é apresentar informações de cunho prático aos analistas. Para tal, os metadados devem traduzir a terminologia técnica para os termos dos negócios. O suporte proporcionado deve ser enfocado sob o ponto de vista do usuário final e não apenas sob a ótica dos desenvolvedores (PEREIRA, 2000).

Em um ambiente operacional, os metadados são especialmente valiosos para os desenvolvedores de aplicação e os administradores do banco de dados. Os bancos de dados operacionais são usualmente utilizados via aplicações, que já contém as definições de dados embutidas. Seus usuários simplesmente interagem com as telas do sistema, sem precisar

conhecer como os dados são mantidos pelo banco de dados (CAMPOS; FILHO, 2005).

O ambiente de suporte à decisão, por sua vez, é bastante distinto. Nele, analistas de dados e executivos procuram por fatos não usuais e correlações que serão reconhecidas quando encontradas. Aplicações rotineiras e pré-definidas não fazem sentido neste ambiente. Os usuários de um DW precisam examinar seus dados e para tal, conhecer sua estrutura e significado (CAMPOS; FILHO, 2005).

1.4 *Data Staging Area* – extração, transformação e carga de dados

No ambiente de DW, os dados são inicialmente extraídos de sistemas operacionais e de fontes externas, posteriormente são integrados e transformados (limpos, eliminados, combinados, validados, consolidados, agregados e sumariados) antes de serem carregados no DW.

Esta é uma etapa crítica da construção de um DW, pois envolve toda a movimentação dos dados. A mesma se dá basicamente em três passos, conhecidos como ETC: Extração, Transformação (passo este que inclui a limpeza dos dados) e Carga dos dados.

De acordo com Kimball;Ross (2002, p. 10), "A *data staging area* abrange tudo entre os sistemas operacionais de origem e a área de apresentação dos dados".

A Figura 1.6 exemplifica o processo de extração, transformação e carga dos dados.

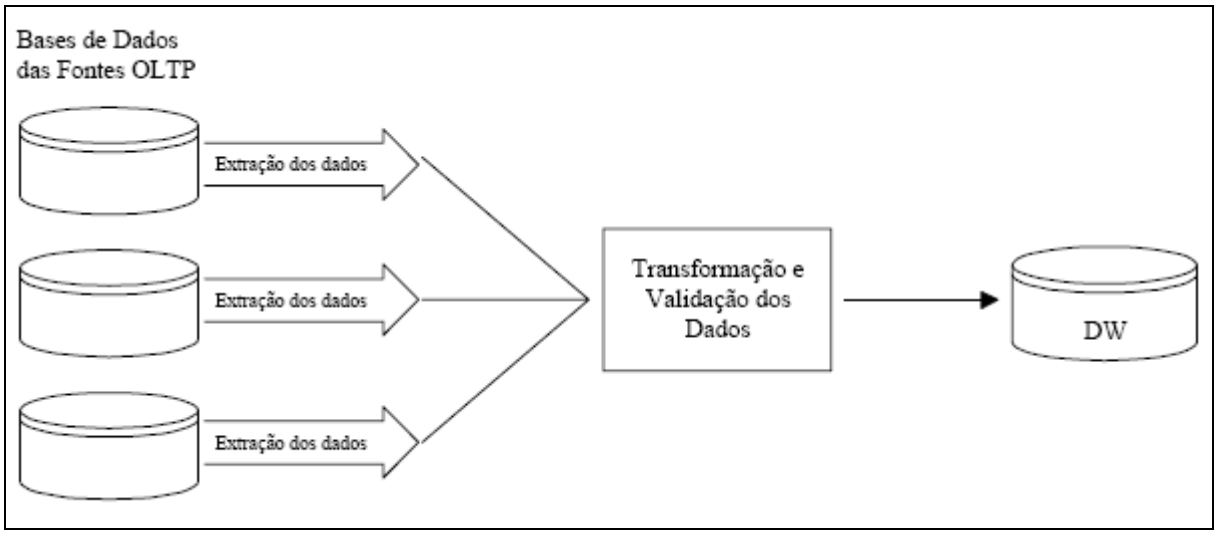


Figura 1.6 *Extração, transformação e carga de dados*.
Fonte: (PEREIRA, 2000, p. 6).

1.4.1 Extração

A extração é o primeiro passo na obtenção de dados para o ambiente do DW. Significa basicamente ler e entender as fontes de dados e copiar as partes necessárias para a área de transformação de dados, a fim de serem trabalhadas posteriormente (KIMBALL; ROSS, 2002).

Conforme Pereira (2000), a tarefa de extração de dados está relacionada com as seguintes etapas do ciclo de vida de DW: definição do escopo do projeto de DW, análise dos sistemas fontes e a especificação dos programas de extração.

No escopo do projeto são identificados os grupos de usuários que interagirão direta ou indiretamente com o sistema e definidos os requisitos de informação para os processos de negócios a serem suportados por uma abordagem DW.

A análise dos sistemas fontes tem por objetivo compreender os dados distribuídos pela organização e integrá-los de forma a refletir a perspectiva histórica de interesse às análises do ambiente de decisão.

Os programas de extração devem dar suporte a captura incremental dos dados que equivale a uma replicação baseada em dados modificados para posterior distribuição ao DW.

1.4.2 Transformação de dados

Uma vez que os dados tenham sido extraídos dos sistemas fontes, um conjunto de transformações deve ser processado sobre esses dados. A transformação dos dados pode ser simples ou complexa, dependendo da natureza dos sistemas fontes. Em algumas situações, múltiplos estágios de transformações são necessários (PEREIRA, 2000).

As rotinas de limpeza e integração atuam sobre os dados extraídos. A execução dessas rotinas de limpeza sobre os dados coletados permite assegurar sua consistência. Dados migrados para o DW, sem integração, não podem ser empregados no suporte a uma visão corporativa dos dados (INMON, 1997).

Segundo Hokama et al. (2004), após a extração dos dados dos sistemas fontes, são realizadas uma série de atividades sobre esses dados de modo a convertê-los em formato válido para o negócio e adequado para carga. A transformação dos dados poderá envolver um

ou vários processos, dependendo da necessidade e situação. Alguns dos processos mais comumente utilizados são:

Limpeza: constitui no conjunto de atividades realizadas, sobre os dados extraídos, de modo a corrigir o uso incorreto ou inconsistente de códigos e caracteres especiais, resolver problemas de conflito de domínios, tratar dados perdidos, corrigir os valores duplicados ou errados. Independente do problema a ser solucionado pelo processo de limpeza, a finalidade é deixar os elementos de dados dentro de formatos padrões (uniformizados), não duplicados, corretos, consistentes e espelhando a realidade.

Combinação: é realizada quando fontes de dados possuem exatamente os mesmos valores de chaves representando registros iguais ou complementares.

Desnormalização e normalização: o padrão no processo de transformação é reunir as hierarquias de dados, separadas em várias tabelas devido à normalização, dentro de uma única dimensão, de forma desnormalizada. Pode ocorrer, entretanto, que dados provenientes do processo de extração estejam completamente desnormalizados dentro de arquivos texto, nesse caso é possível que seja necessário normalizar partes dos registros.

Cálculos, derivação e alocação: são transformações a serem aplicadas às regras do negócio identificadas durante o processo de levantamento de requisitos. É conveniente que as ferramentas a serem empregadas possuam um conjunto de funções, tais como manipulação de textos, aritmética de data e hora, entre outras.

1.4.3 Carga de dados

Conforme Hokama et al. (2004), após os dados serem transformados, eles são carregados no DW. A parte de carga dos dados também possui uma enorme complexidade, sendo que os seguintes fatores devem ser levados em conta:

Integridade dos dados: no momento da carga, é necessário checar os campos que são chaves estrangeiras[2] com suas respectivas tabelas para certificar-se de que os dados existentes na tabela da chave estrangeira estão de acordo com a tabela da chave primária;

Tipo de carga a ser realizada - incremental ou total: a carga incremental

[2] Chave estrangeira é uma chave formada pela chave primária de outra tabela e a chave de um campo da tabela que recebe o relacionamento. É a chave que define um relacionamento entre as tabelas (WIKIPEDIA, 2005).

normalmente é feita para tabelas de fatos e a carga total é feita em tabelas de dimensão onde o analista terá que excluir os dados existentes e incluí-los novamente. Mas isso depende da necessidade do negócio em questão.

Otimização do processo de carga: todo banco de dados possui um conjunto de técnicas para otimizar o processo de carga, tais como evitar a geração de *log*[3] durante o processo, criar índices e agregar dados. Muitas dessas características podem ser invocadas dos bancos de dados ou registradas em *scripts* através da utilização de ferramentas sobre a área de organização de dados.

Suporte completo ao processo de carga: o serviço de carga também precisa suportar as exigências antes e depois da carga atual, como eliminar e recriar índices e particionamento físico de tabelas e índices.

[3] *Log* é a nomenclatura dada aos registros que um sistema mantém, de todos os eventos dignos de nota que acontecem enquanto ele está rodando.

2 A MODELAGEM DIMENSIONAL

Nos bancos de dados relacionais, a redundância dos dados é evitada, sendo aceita somente em determinados casos em que é realmente necessária. Esta redundância é eliminada através de processos de normalização.

A normalização das tabelas traz benefícios nos casos em que muitas transações são efetuadas, pois estas se tornam mais simples e rápidas. Já no caso do DW ocorre o contrário, as transações operam sobre um grande volume de dados e não são simples nem freqüentes, não sendo conveniente a normalização das tabelas, pois no ambiente de DW ocorrem poucas transações concorrentes e cada transação acessa um grande número de registros (PERNAS, 2003).

Conforme Pernas (2003), um outro ponto que distingue o banco de dados relacional do DW está relacionado à modelagem dos dados. Enquanto o banco de dado relacional geralmente utiliza a modelagem Entidade–Relacionamento (ER)[1], o DW utiliza-se de uma modelagem lógica chamada de modelagem dimensional. A modelagem dimensional é a técnica utilizada para se ter uma visão multidimensional dos dados.

De acordo com Barbieri (2001, p. 74), "a modelagem de dados é seguramente um dos fatores críticos de sucesso num projeto de *Data Warehouse*, e pode representar a fronteira entre o sucesso e o seu fracasso".

> *Como o modelo relacional trabalha com normalização, suas tabelas possuem menos registros e não têm redundâncias, apresentando assim uma melhor performance nas tarefas do dia a dia, como inclusões, alterações e exclusões de registros, mas ele só é adequado para consultas simples de poucos registros. Para análises mais complexas com um universo de registros maior, o modelo*

[1] Proposta por Peter P. Chen em 1976, A modelagem Entidade-Relacionamento (ER) envolve identificando, os elementos de importância na organização (entidades), as propriedades destes elementos (atributos) e como eles estão relacionados uns aos outros (relacionamentos). O modelo resultante da informação é independente de qualquer armazenamento de dados ou método de acesso (LOBO, 1998).

> *dimensional oferece uma melhor alternativa, economizando em junções com várias tabelas, e armazenando dados que facilitam a análise das informações* (HOKAMA et al, 2004, p. 32).

"O modelo dimensional é assimétrico, ou seja, possui uma grande tabela, que é a principal, está localizada no centro do diagrama e possui outras tabelas secundárias ao seu redor, que são menores e que se relacionam com a tabela principal. A tabela central é chamada de tabela de fatos e as demais são chamadas tabelas de dimensão" (BISPO, 1999, p. 24).

Um modelo dimensional é composto basicamente pela tabela de fatos e pelas tabelas de dimensões. A tabela fato traz o resultado da consulta, valores de medição. As restrições, objeções e questionamentos ficam nas tabelas dimensões, que trazem informações textuais sobre o valor medido na tabela fato. Perguntas como, por exemplo: Qual? Quem? Onde? Por quê? São respondidas com informações das tabelas dimensões (BARBALHO, 2003).

2.1 Tabela de fatos

Segundo Kimball; Ross (2002), a tabela de fatos é a principal tabela de um modelo dimensional, onde as medições numéricas de interesse da empresa estão armazenadas. A palavra *fato* é usada para representar uma medição de negócio, como quantidades, valores e indicadores. A tabela de fatos registra os fatos que serão analisados. É composta por uma chave primária (formada por uma combinação única de valores de chaves de dimensão) e pelas métricas de interesse para o negócio.

"Em uma tabela de fatos, uma linha corresponde a uma medição. Uma medição é uma linha em um tabela de fatos. Todas as medições em uma tabela de fatos devem estar alinhadas na mesma granularidade" (KIMBALL; ROSS, 2002, p. 21).

De acordo com Hokama et al. (2004), a tabela de fatos é sempre esparsa, ou seja, possui um número relativamente pequeno de todas as combinações possíveis de valores de chaves. Por exemplo, no caso de um banco de dados de uma área comercial, a presença de todas as combinações possíveis representaria que todos os clientes compram para todas as datas de entrega, todos os produtos da empresa, o que é praticamente impossível. Por isso podemos concluir que esse banco é extremamente esparso, pois uma porcentagem muito pequena de todas as combinações possíveis de clientes, produtos e datas de entrega aparecerá nele.

Um aspecto importante é que a tabela de fatos deve representar uma unidade do processo do negócio, não devendo misturar assuntos diferentes numa mesma tabela de fatos.

> *Os atributos mais comuns em uma tabela de fatos são valores numéricos. Estes valores são, em sua maioria, aditivos. As métricas aditivas são as que permitem operações como adição, subtração e média de valores por todas as dimensões, em quaisquer combinações de registros, como "total de itens vendidos" por combinação de data, produto e loja. Métricas aditivas são importantes porque normalmente as aplicações de DW não retornam uma linha da tabela de fatos, mas sim centenas, milhares e até milhões* (HOKAMA et al, 2004, p. 36).

A Figura 2.1 mostra um exemplo de uma tabela de fatos, chamada “VENDA”, composta pelas chaves das dimensões de data, cliente e produto e pelas medidas quantidade, valor, custo e lucro. (Grifo nosso).

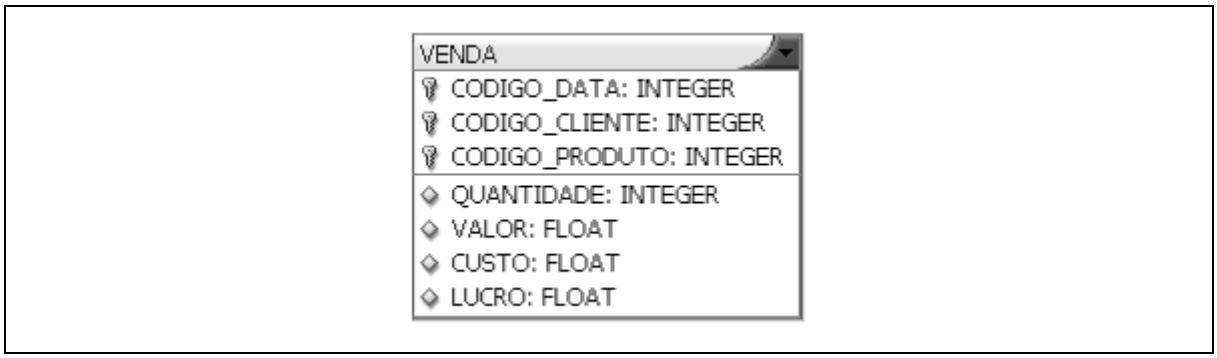


Figura 2.1 *Exemplo de tabela de fatos*.

2.2 Tabela de dimensão

Kimball; Ross (2002) afirmam que a tabela de dimensão contém as descrições textuais do negócio. Seus atributos são fonte das restrições das consultas, agrupamento dos resultados, e cabeçalhos para relatórios. As dimensões são os aspectos pelos quais se pretende observar as métricas relativas ao processo que está sendo modelado.

A qualidade do DW está diretamente ligada à qualidade dos atributos de dimensões, portanto devem ser dedicados tempo e atenção a sua descrição, ao seu preenchimento e a garantia da qualidade dos valores dos seus atributos.

Kimball; Ross (2002) dizem ainda que cada dimensão deve ser identificada com uma única chave primária. Essa chave é a base da integridade referencial no relacionamento com a tabela de fatos. Uma tabela de dimensão é composta também de atributos, os melhores

atributos são textuais e discretos, e devem consistir de palavras reais, evitando-se o uso de códigos e abreviações. Esses atributos descrevem as linhas na tabela de dimensão. A Figura 2.2 mostra um exemplo de tabela de dimensão "PRODUTO", identificada pela chave *codigo_produto* e composta pelos demais atributos que o caracterizam. (Grifo nosso).

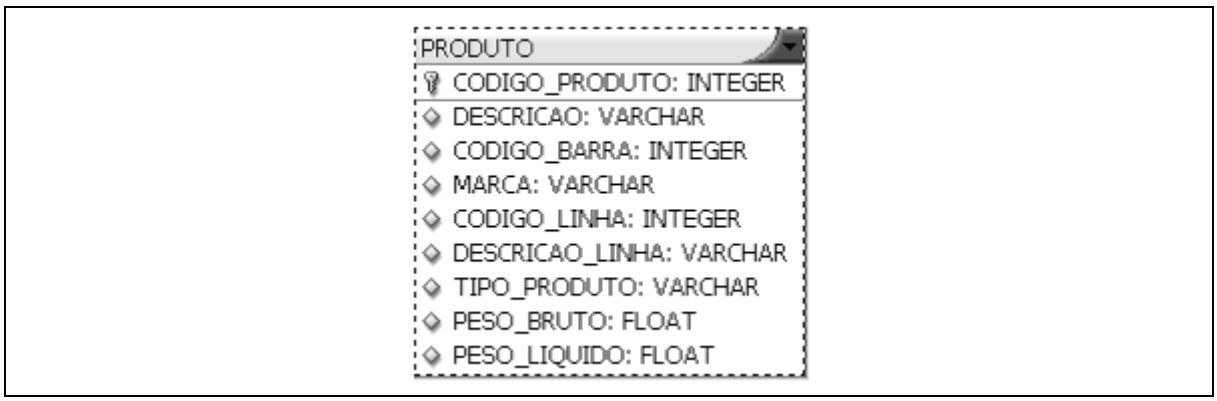


Figura 2.2 *Exemplo de tabela de dimensão.*

2.3 Agregados

Devido ao grande volume de dados envolvidos nas consultas para análise de dados, o tempo de resposta pode se tornar um fator crítico. Uma técnica utilizada para obter ganhos de performance é a criação de tabelas agregadas, sendo um dos principais recursos para ajuste de desempenho do DW. Basicamente, consiste em criar novas tabelas com os dados da tabela de fatos, mas alterando a granularidade da mesma, gerando assim tabelas menores, com dados sumariados (HOKAMA et al. 2004).

Os valores agregados podem representar uma solução e ao mesmo tempo um problema. Solução, porque com a criação de tabelas sumariadas facilitam o acesso aos dados e agilizam o processo de tomada de decisão. Problema, pois de certa forma contrariam os conceitos estabelecidos de não-redundância para os banco de dados. Além disso, ocupa mais espaço para o armazenamento dos dados em um estado pré-processado (BARBIERI, 2001).

A Figura 2.3 mostra um exemplo da utilização de tabelas de agregados com base na tabela de fatos. Foram criadas duas tabelas agregadas: a tabela "AGREGADA_1", onde as informações estão sumariadas para todos os clientes e a tabela "AGREGADA_2" onde as informações estão sumariadas por produto. (Grifo nosso).

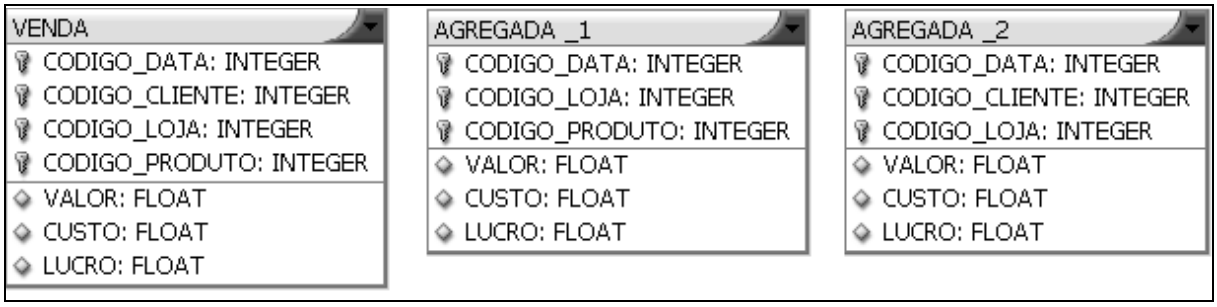


Figura 2.3 *Exemplo de tabela de agregados.*

2.4 Técnicas de modelagem

Existem técnicas específicas de modelagem dimensional. As mais conhecidas e mais utilizadas são a *Star schema* (esquema estrela) e a *Snowflake schema* (esquema flocos de neve), as quais são explicadas a seguir:

2.4.1 Star Schema

O esquema estrela é uma estrutura simples, com poucas tabelas e relacionamentos. Assemelha-se ao modelo de negócio, o que facilita a leitura e entendimento, não só pelos analistas, como por usuários finais não familiarizados com estruturas de banco de dados.

O nome estrela está associado à disposição das tabelas no modelo, que consiste de uma tabela central, a tabela de fatos, que se relaciona com diversas outras tabelas, as tabelas de dimensão. A Figura 2.4 demonstra este relacionamento.

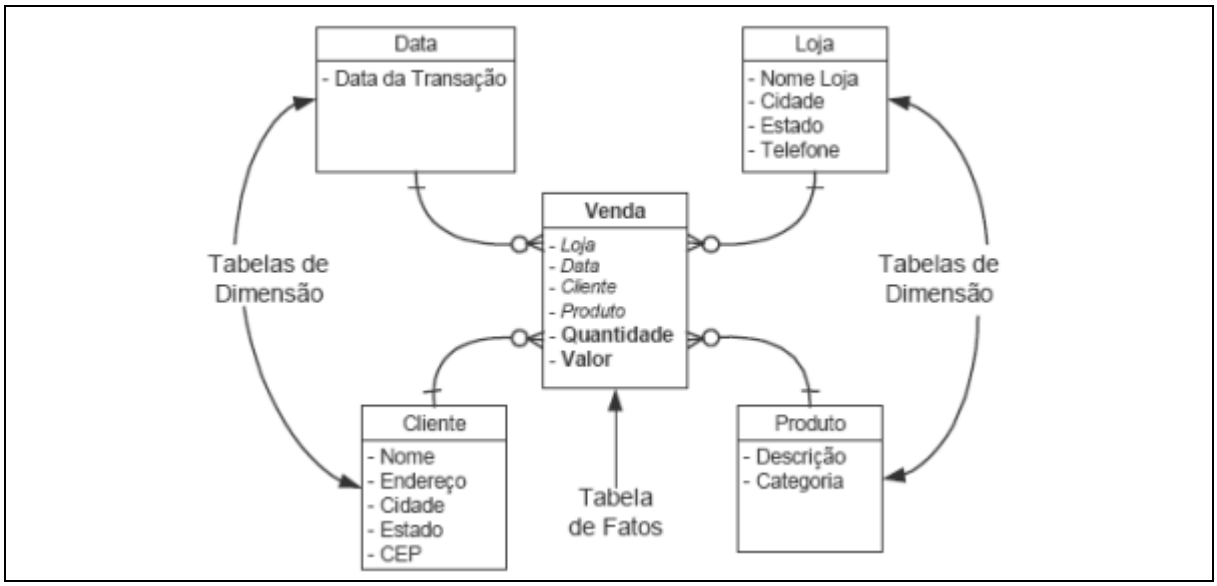


Figura 2.4 *Modelo Star Schema.*
Fonte: (BRAGA, 2004, p. 20)

De acordo com Machado (2000), neste modelo o relacionamento entre a tabela de fatos e as tabelas dimensão é a ligação entre duas entidades, representando relacionamento de um para muitos no sentido da dimensão para o fato.

2.4.2 Snowflake

O modelo *Snowflake* é o resultado da aplicação da terceira forma normal sobre as entidades dimensão. Este modelo é o resultado da decomposição de uma ou mais dimensões que possuem hierarquias entre seus membros (MACHADO, 2000).

Para ser criado é necessário remover os atributos de cardinalidade mínima de uma tabela de dimensão, para colocá-los em outra tabela de dimensão, conectados por uma chave (*snowflake key)* (KIMBALL, 2000).

A Figura 2.5 exemplifica essa técnica de modelagem, onde pode-se observar a normalização das tabelas de dimensão.

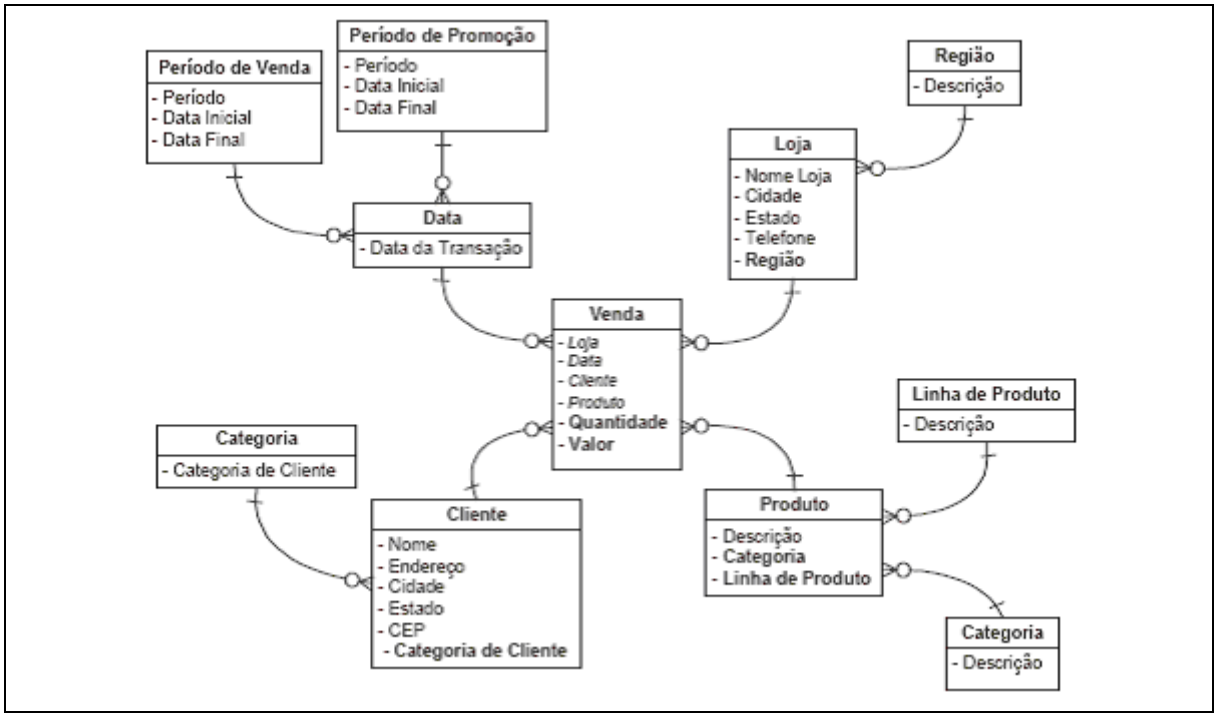


Figura 2.5 *Modelo Snowflake.*
Fonte: (BRAGA, 2004, p. 21).

2.5 Cubo de dados

> *Cubo de dados é uma estrutura multidimensional que expressa a forma na qual os tipos de informações se relacionam entre si. É formado pela tabela de fatos e pelas tabelas de dimensão que a*

> *circundam e representam possíveis formas de visualizar e consultar os dados. O cubo armazena todas as informações relacionadas a um determinado assunto, de maneira a permitir que sejam montadas várias combinações entre elas, resultando na extração de várias visões sobre o mesmo tema* (HOKAMA et al. 2004, p. 49).

Brito (2004), afirma que a idéia fundamental é que praticamente todo o tipo de negócio pode ser representado por uma espécie de cubo de dados, onde as células do cubo contêm valores mensuráveis e as bordas do cubo definem as dimensões naturais dos dados. Pode-se necessitar utilizar mais de três dimensões nos modelos, passando o cubo a ser chamado de *hiper-cubo.*

De forma resumida, o cubo é uma estrutura multidimensional que armazena os dados tornando-os mais fáceis de se analisar. Cada cubo contém uma tabela de fatos e várias dimensões. Por exemplo, um cubo contendo informações de vendas poderá ser composto pelas dimensões tempo, região, produto, cliente, cenário (orçado ou real) e medidas. Medidas típicas seriam, por exemplo, valor de venda, unidades vendidas, custos, margem de lucro, etc.

De acordo com Fortulan; Filho (2005), os cubos são os principais objetos de uma ferramenta OLAP. Construídos com tecnologia que permite rápido acesso aos dados, normalmente eles são construídos a partir de subconjuntos de um DW e são organizados e sumariados dentro de estruturas multidimensionais definidas por dimensões e medidas.

A Figura 2.6 mostra um exemplo de um cubo de dados, contendo as dimensões tempo, produto e local, sendo quantidade vendida uma de suas medidas.

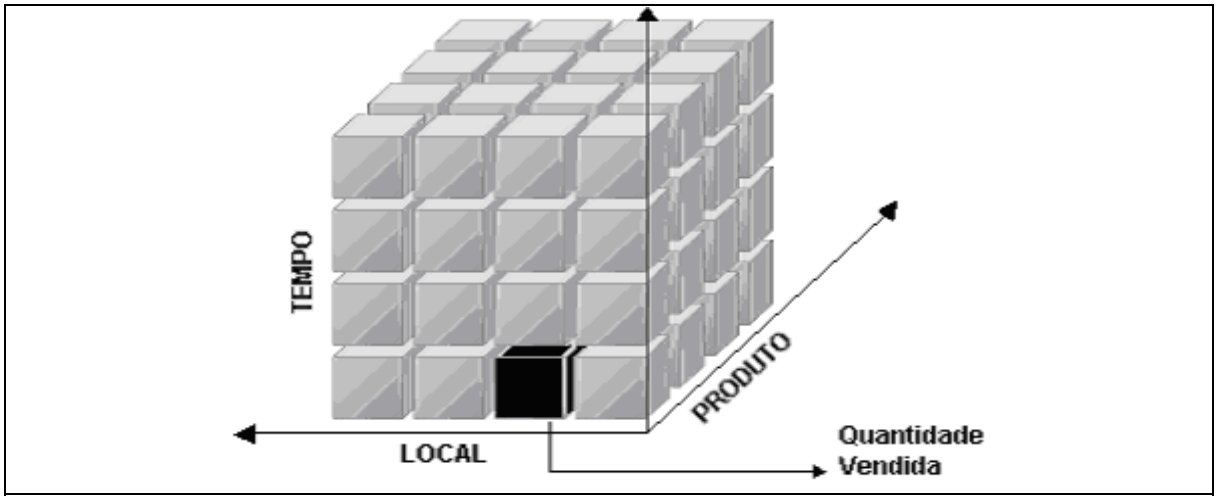


Figura 2.6 *Cubo de dados.*

2.6 OLAP

O termo OLAP possui vários significados, pois sua tecnologia está presente em várias camadas como: armazenamento, acesso, compiladores, linguagem e conceitos. Pode-se falar em conceitos OLAP, linguagens OLAP, camadas de produtos OLAP e produtos completos OLAP (THOMSEN, 2002).

> *O DW é utilizado para armazenar informações e o OLAP para recuperá-las, ambos são especializados para exercer suas funções de forma eficiente. As duas tecnologias são complementares de modo que um bom DW é planejado com produção de relatórios em mente. Desta forma, para explorar o DW completamente é necessário o OLAP que irá extrair e alavancar totalmente as informações nele contidas* (ANZANELLO, 2005, p. 5).

OLAP é uma ferramenta de BI utilizada para apoiar as empresas na análise de suas informações, em vários níveis estratégicos, visando obter novos conhecimentos que serão empregados na tomada de decisão.

A modelagem OLAP é descritiva. Nela não se utiliza código, nem siglas, e sim nomes representativos que permitem a identificação por qualquer pessoa, que, mesmo não sendo da área, entenda os valores apresentados em um relatório (THOMSEN, 2002).

A aplicação OLAP é bastante diversificada e seu uso encontra-se em diversas áreas de uma empresa. Por exemplo, em uma aplicação na área de vendas, o OLAP pode ser utilizado para fazer análises de vendas (por região, produto, vendedor, etc.), previsões, lucratividade de cliente/pedido, análise de canais de distribuição, dentre outros.

As operações utilizando ferramentas OLAP podem responder inúmeras questões, como por exemplo:

- Qual foi o valor das vendas realizadas por um determinado vendedor no mês de outubro?
- Quais foram os clientes atendidos por um determinado vendedor no primeiro semestre do ano anterior que resultou em lucros inferiores a 10%?
- Qual o produto que traz maior lucratividade para a empresa nos diversos meses do ano?

- Quais clientes ou grupos de clientes são mais lucrativos para a empresa?
- A Empresa necessita contratar mais vendedores? Quais regiões necessitam de tais vendedores?

2.6.1 Operações OLAP

De acordo com Machado (2000), Barbieri (2001), Singh (2001), as ferramentas OLAP permitem ao usuário, navegar entre diferentes granularidades de um cubo de dados. As mais freqüentes funcionalidades oferecidas pelas ferramentas OLAP são:

Drill-across: é a requisição de dados das tabelas de dimensão com o valor das consultas modificado, ocorre quando o usuário "pula" um nível intermediário dentro de uma mesma dimensão. Por exemplo, em uma dimensão tempo composta por ano, semestre, trimestre, mês e dia. O usuário estará executando um *drill-across* quando ele passar de ano direto para trimestre ou mês. (Grifo nosso).

Drill-down: aumento do nível de detalhe da informação, por parte do usuário, para solicitar uma visão mais detalhada em um conjunto de dados. Por exemplo, se os dados de vendas estiverem sumariadas e o usuário desejar informações mais detalhadas, por vendedor que realizou a venda, por exemplo, realiza-se então um *drill-down* para que os dados sejam específicos de cada vendedor.

Drill-up: é o contrário do *drill-down*, ocorre quando o usuário diminui o nível de detalhe da informação, solicitando uma visão menos detalhada de um conjunto de dados. Por exemplo, se o usuário está visualizando a venda de um determinado vendedor e passa a analisar as vendas sumariadas de todos os vendedores.

Drill-through: é quando se deseja uma informação em um nível de detalhe menor do que aquele armazenado na tabela de fato e permitido pela sua granularidade. Por exemplo, em um nível de granularidade de produto por dia e por loja deseja-se uma informação que está presente na nota fiscal. *O drill-through* é a operação que busca esta informação além do nível granularidade existente na estrutura dimensional. Esta busca poderia ser no próprio sistema transacional de origem, no caso de compatibilidade entre ambos.

Slice-dice: é a descrição padrão para a habilidade de acessar os dados do DW através de qualquer uma das dimensões de forma igual. Serve para modificar a posição de uma

informação, alterar linhas por colunas de maneira a facilitar a compreensão dos usuários e mudar de dimensão sempre que necessário. Esta operação equivale a “fatiar” o cubo de dados, ou seja, um determinado valor de uma dimensão é fixado. Por exemplo, podemos fixar o valor Sul para a dimensão Localidade, e analisar as outras dimensões. Assim, pode ser conhecida a eficiência de entrega desta região em função dos produtos (explorando os valores da dimensão Produtos) e da forma de distribuição utilizada (dimensão Distribuição). Esta operação permite conhecer em detalhes o relacionamento de um valor de uma dimensão com as outras dimensões. (Grifo nosso).

Pivoting (pivoteamento): é a mudança do arranjo das linhas e colunas em um relatório tabular, onde freqüentemente as linhas ou as colunas são derivadas de dimensões diferentes. É a inversão dos eixos das dimensões, para obter-se novas visões de consultas. Em algumas ferramentas, para que seja realizada esta operação, podem-se simplesmente utilizar as funções de "arrastar e soltar" com o *mouse.* (Grifo nosso).

Consultas *ad-hoc*: de acordo com Inmon, apud Cielio (2005), são consultas com acesso casual único e tratamento dos dados segundo parâmetros nunca antes utilizados, geralmente executadas de forma iterativa e heurística. Isso tudo nada mais é do que o próprio usuário gerar consultas de acordo com suas necessidades de cruzar as informações de uma forma não vista e com métodos que o levem a descoberta daquilo que procura.

2.6.2 Arquiteturas OLAP

De acordo com Cunha (2004), Cielio (2005), a arquitetura OLAP diz respeito ao método de armazenamento de dados utilizado para uma aplicação OLAP. Os métodos de armazenamento de dados são MOLAP, ROLAP, HOLAP, DOLAP e WOLAP. Cada um deles tem uma função específica e deve ser utilizada quando melhor atender às necessidades de análise pela ferramenta de OLAP. Segundo os autores citados, as características destas arquiteturas são:

MOLAP (*Multidimensional On Line Analytical Processing*): Processa-se da seguinte forma: com um servidor multidimensional o acesso aos dados ocorre diretamente no banco de dados, ou seja, o usuário trabalha, monta e manipula os dados do cubo diretamente no servidor. Isso traz grandes benefícios aos usuários no que diz respeito à performance, mas tem problemas com escalabilidade além de ter um custo alto para aquisição. A Figura 2.7 demonstra esta arquitetura.

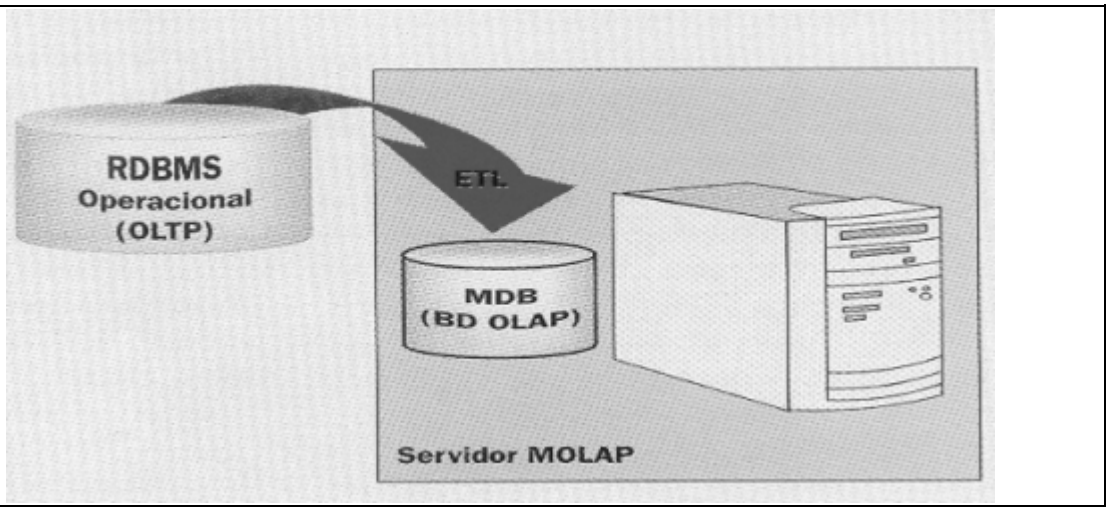


Figura 2.7 *Arquitetura MOLAP.*
Fonte: (CUNHA, 2004).

ROLAP (*Relational On Line Analytical Processing*): Possuem uma engenharia de acesso aos dados e análise OLAP com uma arquitetura um pouco diferente. Nesse caso a consulta é enviada ao servidor de banco de dados relacional e processada no mesmo, mantendo o cubo no servidor. O que se pode notar nesse caso é que o processamento OLAP se dará somente no servidor. A principal vantagem dessa arquitetura é que ela permite analisar enormes volumes de dados, em contra partida uma grande quantidade de usuários acessando simultaneamente poderá causar sérios problemas de performance no servidor. A Figura 2.8 exemplifica esta arquitetura.

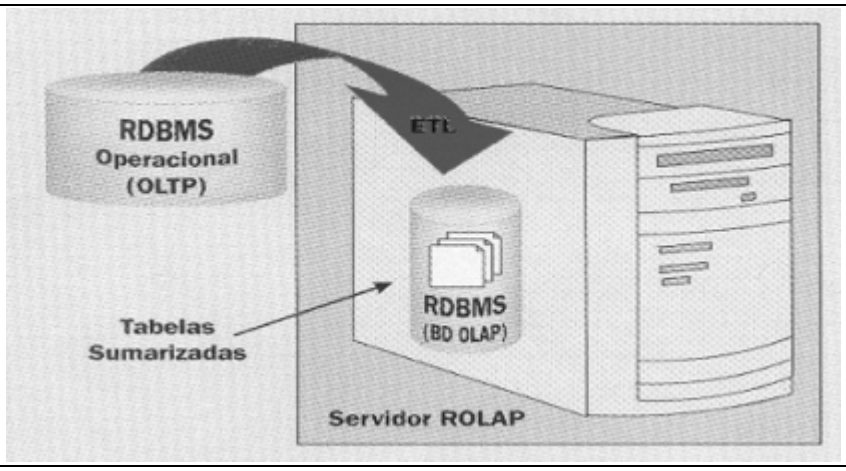


Figura 2.8 *Arquitetura ROLAP.*
Fonte: (CUNHA, 2004).

HOLAP (*Hybrid On Line Analytical Processing*): Essa forma de acessar os dados é uma mistura de tecnologias onde há uma combinação entre ROLAP e MOLAP. A vantagem é que com esta mistura de tecnologias pode-se extrair o que há de melhor de cada uma, ou seja,

a alta performance do MOLAP com a melhor escalabilidade do ROLAP. A Figura 2.9 mostra o exemplo desta arquitetura.

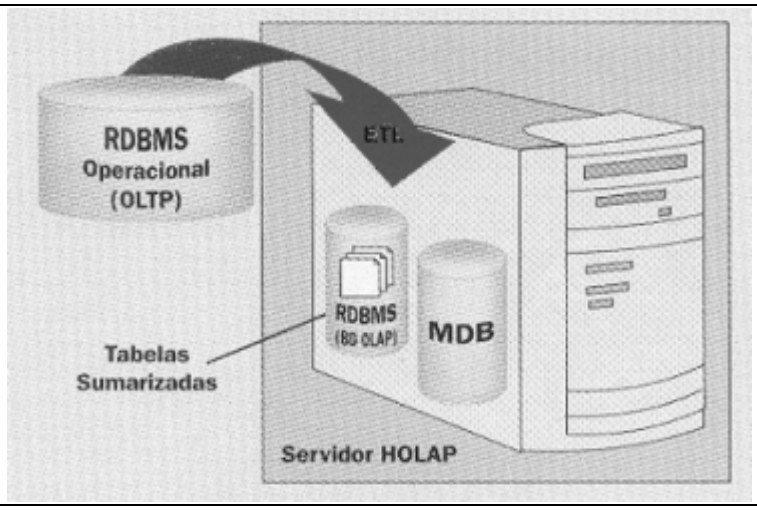


Figura 2.9 *Arquitetura HOLAP.*
Fonte: (CUNHA, 2004).

DOLAP *(Desktop On Line Analytical Processing*): São as ferramentas que, a partir de um cliente qualquer, emitem uma consulta para o servidor e recebem o cubo de informações de volta para ser analisado na estação cliente. O ganho com essa arquitetura é o pouco tráfego que se dá na rede, visto que todo o processamento OLAP acontece na máquina cliente, e a maior agilidade de análise, além do servidor de banco de dados não ficar sobrecarregado, sem incorrer em problemas de escalabilidade. A desvantagem é que o tamanho do cubo de dados não pode ser muito grande, caso contrário, a análise passa a ser demorada e/ou a máquina do cliente pode não suportar em função de sua configuração. Esta arquitetura é exemplificada na figura 2.10.

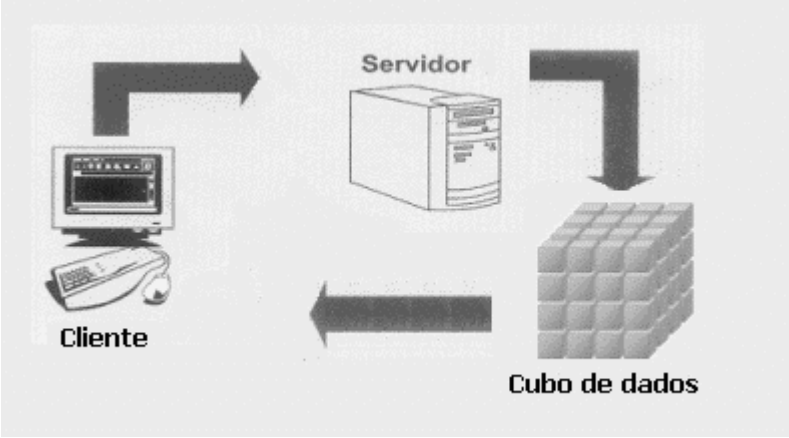


Figura 2.10 *Arquitetura DOLAP.*
Fonte: Adaptação pelo autor conforme (CUNHA, 2004).

WOLAP *(Web On Line Analytical Processing):* É a utilização de uma ferramenta OLAP a partir de um navegador *Web* (*browser*). A arquitetura das ferramentas WOLAP é uma variação da cliente/servidor. A diferença está na utilização de um *middleware*[2] do lado servidor que será o responsável pela comunicação entre o cliente e uma aplicação servidora, neste caso, o banco de dados. A Figura 2.11 exemplifica esta arquitetura.

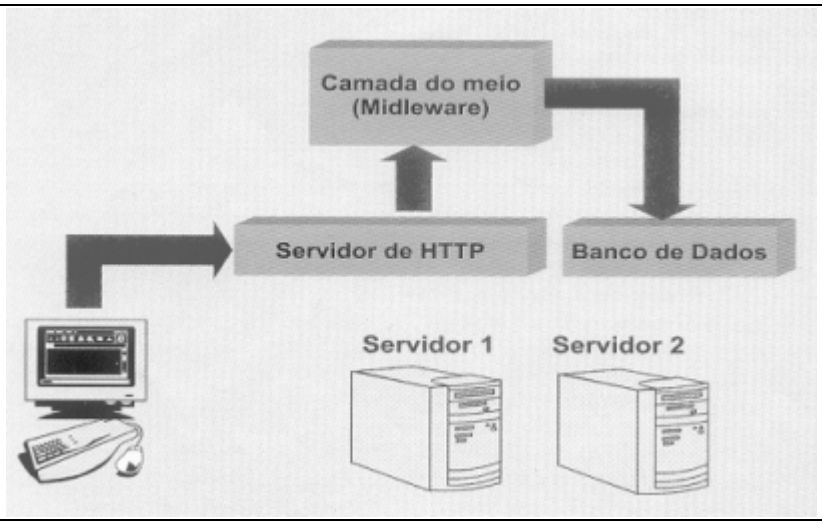


Figura 2.11 *Arquitetura WOLAP.*
Fonte: (CUNHA, 2004).

2.6.3 Tendências

JOLAP - É um esforço da *Java Community Process* (JCP)[3] de projetar uma API *Java*[4] para servidores e aplicações OLAP, aderentes ao ambiente *Java2 Plataform Enterprise Edition* (J2EE)[5]. Ela está sendo especificada para suportar a criação e manutenção de dados e metadados OLAP, independente de fornecedor. JOLAP é baseada em uma forte generalização, orientada a objeto e nos conceitos de OLAP. Este modelo suporta conceitos referentes a três áreas que são chave para as aplicações OLAP: Metadados, dados e pesquisas (ANZANELLO, 2005).

XMLA (XML[6] *for Analysis)* – O XMLA tem como proposta prover

[2] *Middleware* é uma camada de software localizada entre as aplicações e o sistema operacional. Seu objetivo é tanto facilitar o desenvolvimento de aplicações por parte dos programadores quanto esconder os detalhes das camadas inferiores e a heterogeneidade entre diferentes sistemas operacionais (WIKIPEDIA, 2005).
[3] O *Java Community Process* tem o objetivo de desenvolver e rever as especificações da tecnologia *Java*. JCP. *Java Community Process*. Disponível em: <http://www.jcp.org>. Acesso em: 06 ago. 2005.
[4] JAVA. *Java Technology.* Disponível em: < http://java.sun.com>. Acesso em: 06 ago. 2005.
[5] Plataforma *Java* para desenvolvimento e execução de aplicações servidoras, com capacidade de suporte ao desenvolvimento de aplicações robustas e escaláveis. JAVA. *Java Technology.* Disponível em: <http://java.sun.com>. Acesso em: 06 ago. 2005.
[6] XML. *Extensible markup language*. Disponível em: <http://www.w3.org/XML>. Acesso em: 02 out. 2005.

interoperabilidade em ambientes OLAP heterogêneos, utilizando a *Web* como infra-estrutura de comunicação (XMLA.ORG, 2005).

O XMLA é um padrão comercial aberto para interface de serviço *Web*, projetado especificamente para processamento OLAP e para funções de *data mining*[7] (XMLA.ORG, 2005). Ele tem como objetivo fornecer uma API (*Application Program Interface*) de acesso padrão para os provedores de OLAP do mercado. Ele define um modo padrão para conexão a fontes de dados OLAP ou servidores de banco de dados multidimensionais, análogo ao ODBC (*Open DataBase Connection*)[8] para os bancos de dados relacionais. Além disso, prevê consultas a dados e metadados, usando a sintaxe de uma linguagem de consulta padrão (AMARAL, 2003).

Essa linguagem é o mdXML, que é uma versão encapsulada da linguagem MDX (*Multidimensional Expressions*)[9]. A MDX é uma linguagem de expressão multidimensional definida na especificação OLE DB para OLAP[10] (MICROSOFT apud AMARAL, 2003, p. 64).

---

[7] *Data Mining* ou Mineração de Dados consiste em um processo analítico projetado para explorar grandes quantidades de dados (tipicamente relacionados a negócios, mercado ou pesquisas científicas), na busca de padrões consistentes e/ou relacionamentos sistemáticos entre variáveis e, então, validá-los aplicando os padrões detectados a novos subconjuntos de dados. O processo consiste basicamente em 3 etapas: exploração; construção de modelo ou definição do padrão; e validação/verificação (PUC-RIO, 2005).

[8] O ODBC é um método de acesso a banco de dados, com a finalidade de tornar possível acessar qualquer dado de qualquer aplicação independentemente do banco de dados usado.

[9] A linguagem MDX será descrita em detalhes no capítulo 3.

[10] O *Microsoft* OLE DB para OLAP é um conjunto de objetos e interfaces que estendem a habilidade do OLE DB para prover acesso a armazenagem de dados multidimensional. MICROSOFT. *Microsoft Corporation*. Disponível em: <http://www.microsfot.com>. Acesso em: 06 ago. 2005.

3 FERRAMENTAS OLAP

Neste capítulo será feito o estudo de alguns softwares dentre os quais serão selecionados aqueles com os quais pretende-se implementar a solução proposta por esse trabalho. Os softwares a serem avaliados são *Mondrian*[1], *Jpivot*[2], *OpenI*[3] e *Pentaho*[4]. O estudo será feito com base nas informações disponíveis em suas respectivas páginas na Internet, já que elas são ferramentas muito recentes e ainda carecem de documentação.

3.1 Mondrian

Mondrian é um servidor OLAP *Open Source* escrito em linguagem *Java*. Ele executa consultas escritas em linguagem MDX, lendo os dados de um RDBMS, e apresentando o resultado em um formato multidimensional através de uma API *Java*. A característica de realizar consultas OLAP em dados armazenados em um banco de dados relacional e retornar consultas em forma multidimensional classifica o *Mondrian* como um software ROLAP (BRITO, 2004).

3.1.1 Linguagem MDX

A Linguagem MDX é uma linguagem de consulta semelhante a linguagem SQL[5], porém exclusiva para consultas multidimensionais em bancos de dados. A linguagem MDX foi originalmente criada pela *Microsoft* para utilização com o produto *SQL Server OLAP Services* como parte da especificação OLE DB/OLAP API (MICROSOFT, 2005). Recentemente foi introduzida como parte da API do XMLA, o que motivou a adoção dela por

---

[1] MONDRIAN. *Mondrian OLAP Server*. Disponível em: <http://mondrian.sourceforge.net>. Acesso em: 06 ago. 2005.

[2] JPIVOT. *A JSP based OLAP*. Disponível em: < http://jpivot.sourceforge.net>. Acesso em: 06 ago. 2005.

[3] OPENI. *Open Source Web Application for OLAP Reporting*. Disponível em: <http://www.openi.org>. Acesso em: 06 ago. 2005.

[4] PENTAHO. *Open Source business intelligence.* Disponível em: <http://www.pentaho.org>. Acesso em: 06 ago. 2005.

[5] SQL é uma sintaxe, criada pela IBM, usada para a definição e manipulação de dados em um banco de dados relacional (WIKIPEDIA, 2005).

outros desenvolvedores de aplicações OLAP. Ela é a linguagem de consulta implementada pelo *Mondrian* e um exemplo de sentença MDX é exibido na Figura 3.1.

```
SELECT {[Measures].[Unit Sales], [Measures].[Store Sales]} ON COLUMNS,
  {[Time].[1997].[Q1].descendants} ON ROWS
FROM [Sales]
WHERE [Gender].[F]
```

Figura 3.1 *Instrução MDX.*
Fonte: Adaptação do autor segundo (MONDRIAN, 2005).

Este exemplo faz uma consulta ao cubo Sales ([Sales]), a cada uma das dimensões [Time], [Gender] e alguns de seus membros. O resultado é exibido na Figura 3.2:

[Time]	[Measures].[Unit Sales]	[Measures].[Store Sales]
[1997].[Q1]	10	12
[1997].[Q1].[Jan]	2	6
[1997].[Q1].[Feb]	3	3
[1997].[Q1].[Mar]	5	3

Figura 3.2 *Retorno de uma consulta MDX.*
Fonte: Adaptação do autor segundo (MONDRIAN, 2005).

3.1.2 Arquitetura do *Mondrian*

O servidor OLAP *Mondrian* é composto por quatro camadas: a camada de apresentação (*presentation layer*), a camada dimensional (*dimensional layer*), a camada estrela (*star layer*) e a camada de armazenamento (*storage layer*).

A camada de apresentação determina o que o usuário final vê em seu monitor, e como ele pode interagir para fazer novas consultas. Há muitas maneiras de apresentar conjuntos de dados multidimensionais, incluindo *pivot tables*[6], gráficos estáticos em diversos formatos (pizza, linha, barra, etc...) e ferramentas avançadas de visualização, tais como mapas interativos e gráficos dinâmicos. Estas ferramentas podem ser escritas em interface *Java Swing*[7] ou JSP (*Java Server Pages*)[8] , os gráficos, renderizados no formato JPEG ou GIF.

A segunda camada é a camada dimensional. A camada dimensional analisa

[6] *Pivot table é* um recurso que permite à uma tabela fazer um agrupamento dinâmico, onde o usuário simplesmente arrasta um atributo para linha ou coluna, e a tabela é recalculada e reagrupada para os novos valores.
[7] *Swing* é um *Framework* (conjunto de classes que constitui um *design* abstrato para soluções de uma família de problemas) para a criação de aplicações gráficas em *Java* (JAVA, 2005).
[8] JSP é uma tecnologia baseada em *Java* utilizada para o desenvolvimento de *sites* dinâmicos (JAVA, 2005).

gramaticalmente, valida e executa consultas MDX. Um transformador de consultas (*query transformer*) permite que a aplicação manipule as consultas existentes, ao invés de construir uma nova instrução MDX para cada solicitação.

A terceira camada é a camada estrela, que é responsável por apresentar valores relacionados a determinados níveis de informação. Estes valores podem ser resultados de uma consulta ao banco de dados relacional, ou ser o resultado de muitas consultas armazenadas em memória. É responsabilidade desta camada obter os valores, seja qual for a sua origem. (BRITO, 2004, p. 96).

A camada do armazenamento é um RDBMS. Ela é responsável por fornecer os dados utilizados nas consultas.

Todos estes componentes podem existir em uma mesma máquina, ou podem ser distribuídos entre várias máquinas, sendo que as camadas dimensional e estrela, que compreendem ao servidor *Mondrian*, devem estar na mesma máquina. A camada de armazenamento pode estar em uma outra máquina, acessada através da conexão remota com o uso de *drivers* JDBC[9]. A Figura 3.3 exibe a arquitetura do *Mondrian*.

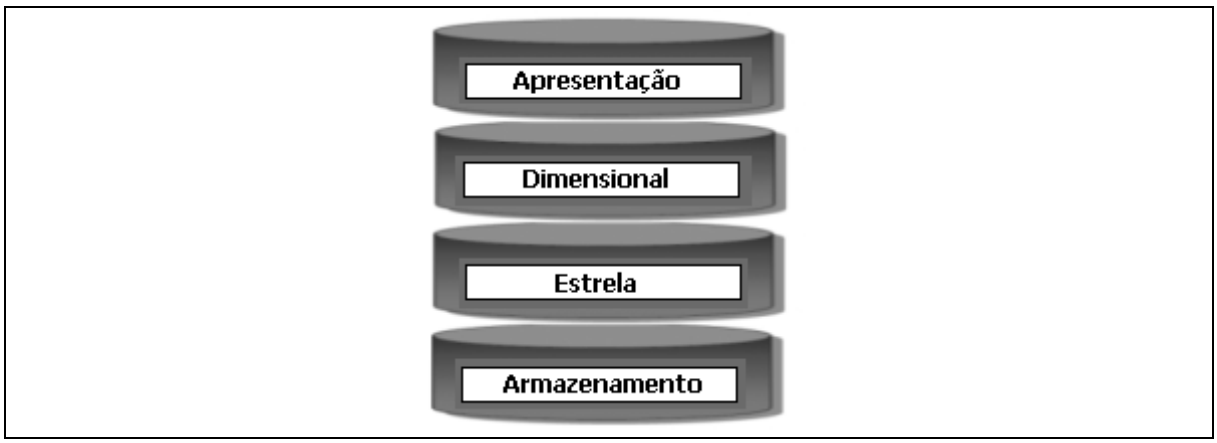


Figura 3.3 *Arquitetura do Mondrian.*
Fonte: Adaptação do autor segundo (MONDRIAN, 2005).

3.1.3 Estratégia de armazenamento e agregação

Na estrutura do *Mondrian*, basicamente três tipos de dados precisam ser armazenados: dados das tabelas fato, dimensões e agregados.

[9] JDBC é um conjunto de classes e interfaces (API) escritas em *Java* que fazem o envio de cláusulas SQL para qualquer banco de dados relacional compatível (JAVA, 2005).

Agregados pré-computados são necessários para grandes conjuntos de dados, caso contrário, certas consultas não poderiam ser executadas sem que fosse efetuada uma leitura completa do conteúdo da tabela de fatos. Na arquitetura ROLAP, os dados agregados são armazenados em tabelas separadas.

O componente final da estratégia de agregação é o *cache.* O *cache* mantém os agregados pré-computados em memória e então as consultas seguintes podem acessá-las sem a necessidade de consultar o disco.

A estratégia de agregação do *Mondrian* é manter os dados da tabela de fatos armazenados no RDBMS e ter os dados agregados no *cache* através da submissão de consultas *group by*[10]. A idéia geral é delegar ao servidor de banco de dados o que é específico do banco de dados.

3.1.4 Mondrian Schema

De acordo com Brito (2004), a lógica do *Mondrian* é implementada através de *Schemas*, que definem o modelo multidimensional lógico e o mapeamento deste modelo em um modelo físico e relacional.

O modelo lógico consiste de elementos definidos pelo *Schema*. Estes elementos são: cubos, dimensões, hierarquias, níveis e membros.

O modelo físico é a fonte de dados que é mapeada pelo modelo lógico através do *Schema*. Tipicamente, é um banco de dados onde as informações ficam armazenadas em tabelas relacionais.

Conforme Brito (2004), a representação destes *Schemas* é feita através de arquivos XML, utilizando os mesmos conceitos relacionados à análise dimensional: cubo (*cube*), representando, em alto nível, a lógica multidimensional do sistema, bem como os fatos (*measures*) e dimensões (*dimensions*). Atualmente, uma das maneiras de criar um *Schema* é editar manualmente o arquivo XML em um editor de texto. Existe também um *plugin* para o ambiente *Eclipse*[11] que facilita a edição do mesmo (JPIVOT, 2005).

A Figura 3.4 exibe um exemplo de arquivo XML definindo um *Schema.*

[10] Cláusula de grupo da linguagem de consulta SQL.
[11] ECLIPSE. *Eclipse Foundation*. Disponível em: <http://www.eclipse.org>. Acesso em: 06 ago. 2005.

```
<Schema>
  <Cube name="Vendas">
    <Table name="vendas_geral"/>
    <Dimension name="Genero" foreignKey="id_cliente">
      <Hierarchy hasAll="true" allMemberName="Todos Generos" primaryKey="id_cliente">
        <Table name="cliente"/>
        <Level name="Genero" column="genero" uniqueMembers="true"/>
      </Hierarchy>
    </Dimension>
    <Dimension name="Tempo" foreignKey="id_tempo">
      <Hierarchy hasAll="false" primaryKey="id_tempo">
        <Table name="dia"/>
        <Level name="Ano" column="ano" type="Numeric"
            uniqueMembers="true"/>
        <Level name="Trimestre" column="trimestre"
            uniqueMembers="false"/>
        <Level name="Mes" column="mes" type="Numeric"
            uniqueMembers="false"/>
      </Hierarchy>
    </Dimension>
    <Measure name="Quantidade Vendida" column="quantidade_vendida"
        aggregator="sum" formatString="#,###"/>
    <Measure name="Vendas da loja" column="vendas_loja"
        aggregator="sum" formatString="#,###.##"/>
    <CalculatedMember name="lucro" dimension="Medidas"
        formula="[Medidas].[Vendas da loja]-[Medidas].[Custos da loja]">
      <CalculatedMemberProperty name="FORMAT_STRING" value="$#,##0.00"/>
    </CalculatedMember>
  </Cube>
</Schema>
```

Figura 3.4 *Exemplo de arquivo XML definindo um Schema.*
Fonte: Adaptação do autor segundo (MONDRIAN, 2005).

Este *Schema* contem um único cubo, chamado “Vendas”. Este cubo possui duas dimensões, “Tempo” e “Genero”, e dois fatos, “Quantidade Vendida” e “Vendas da Loja”. A seguir será detalhado cada um dos componentes do *Schema.* (Grifo nosso):

3.1.4.1 Cubos

Um cubo é uma coleção de medidas e dimensões. A única coisa que as medidas e dimensões tem em comum é a tabela de fatos, “vendas_geral”, no nosso exemplo. A tabela de fatos é definida usando a *tag* <*Table*> conforme exemplo na Figura 3.5:

```
<Schema>
  <Cube name="Vendas">
    <Table name="vendas_geral"/>
    ...
  </Cube>
</Schema>
```

Figura 3.5 *Definição de um cubo.*
Fonte: Adaptação do autor segundo (MONDRIAN, 2005).

3.1.4.2 Medidas

Cada medida tem um nome, uma coluna na tabela de fatos e um agregador. O agregador é geralmente o operador "*sum*", mas "*count*", "*min*", "*max*", "*avg*" e "*distinct count*" também são permitidos. O atributo "*formatString*" é opcional e especifica como o valor deve ser exibido (Grifo nosso). A Figura 3.6 mostra um exemplo da definição de medidas.

```
<Measure name="Quantidade Vendida" column="quantidade_vendida"
    aggregator="sum" formatString="#,###"/>
<Measure name="Vendas da loja" column="vendas_loja"
    aggregator="sum" formatString="#,###.##"/>
```

Figura 3.6 *Definição de medidas.*
Fonte: Adaptação do autor segundo (MONDRIAN, 2005).

3.1.4.3 Dimensões, hierarquias, níveis e membros

Um nível é uma coleção de membros que possuem a mesma distância da raiz da hierarquia. Um membro é um ponto em uma dimensão, determinado por um conjunto particular de valores de atributo. Uma hierarquia é um conjunto de membros organizados em uma estrutura conveniente para análise, por exemplo, a hierarquia de tempo pode consistir do dia, mês, trimestre, ano, etc. Uma dimensão é uma coleção de hierarquias na qual cada uma delas esta associada a um atributo da tabela de fatos. A definição destas estruturas é mostrada na Figura 3.7.

```
<Dimension name="Tempo" foreignKey="id_tempo">
  <Hierarchy hasAll="false" primaryKey="id_tempo">
    <Table name="dia"/>
    <Level name="Ano" column="ano" type="Numeric"
        uniqueMembers="true"/>
    <Level name="Trimestre" column="trimestre"
        uniqueMembers="false"/>
    <Level name="Mes" column="mes" type="Numeric"
        uniqueMembers="false"/>
  </Hierarchy>
</Dimension>
```

Figura 3.7 *Definição de dimensões, hierarquias e níveis.*
Fonte: Adaptação do autor segundo (MONDRIAN, 2005).

Membros calculados: São medidas calculadas a partir de colunas das tabelas de fatos. Ou seja, são valores que podem ser obtidos baseados nos valores armazenados nas colunas das tabelas de fatos. Na Figura 3.8, é mostrado o exemplo do membro calculado "lucro" que é apurado a partir das quantidades vendidas da loja descontando-se os custos

operacionais da mesma. (Grifo nosso).

```
<CalculatedMember name="lucro" dimension="Measures"
    formula="[Medidas].[Vendas da loja]-[Medidas].[Custos da loja]">
  <CalculatedMemberProperty name="FORMAT_STRING" value="$#,##0.00"/>
</CalculatedMember>
```

Figura 3.8 *Definição de membros calculados.*
Fonte: Adaptação do autor segundo (MONDRIAN, 2005).

3.2 Jpivot

Conforme Brito (2004), O *Mondrian*, executa as consultas e as retorna. Este retorno, porém, não tem uma saída definida. A visualização das consultas retornadas depende de outro software, como por exemplo, o *Jpivot*. O *Jpivot* é uma *tag library*[12] que possibilita o acesso e disponibilização de tabelas de forma multidimensional, além de permitir operações básicas em consultas OLAP, como *slice and dice* e *drill down*. Com o uso deste software, é possível escrever aplicativos na forma de JSP, utilizando comandos específicos e gerando saídas através de tabelas e gráficos.

O *Jpivot* é estruturado para suportar diversos servidores OLAP, especialmente *Mondrian* e XMLA. O *Jpivot* não utiliza a API do *Mondrian* diretamente. Ele define sua própria interface OLAP. Para fazer o *Jpivot* funcionar com o *Mondrian*, estas interfaces precisam ser implementadas usando as APIs do *Mondrian.*

3.2.1 WCF

O *Jpivot* utiliza um conjunto de APIs, empacotadas sobre o nome de WCF - *Web Component Framework*, para dar suporte a construção de componentes de interface com o usuário (JPIVOT, 2005).

O WCF é um conjunto de classes *Java* que auxilia a criação, apresentação e validação de formulários HTML via XML e XSLT[13]. O WCF encapsula componentes reusáveis desenvolvidos para o *Jpivot*. O WCF foi desenvolvido para ser facilmente

[12] *Tag Library* é uma coleção de *Tags* personalizáveis que auxiliam no desenvolvimento de sistemas baseados em HTML. Alguns exemplos de tarefas que podem ser feitas por *Tag Libraries* incluem processamento de formulários, acesso à banco de dados, personalização de componentes e outros.
[13] XSLT é uma linguagem de marcação XML usada para transformar documentos XML (WIKIPEDIA, 2005).

integrado com outras aplicações *Web* como portais[14], JSF (*JavaServer Faces*)[15], *Struts*[16] entre outros.

3.2.2 Arquitetura

Brito (2004) explica que os scripts escritos em JSP devem conter as consultas na linguagem MDX, *tags* relativas à sintaxe JSP e *tags* pertencentes à JSP *tag library*. Estes scripts devem ser executados através de um navegador *Web*. O navegador submete o script a um *Web Server*[17] (*Apache Tomcat*[18]). O *Web Server* tratará a requisição e submeterá a consulta ao *Mondrian*, que por sua vez, submeterá ao banco de dados relacional. A consulta retornada pelo banco de dados será processada pelo *Mondrian* através das camadas dimensional e estrela, sendo os resultados remetidos à camada de apresentação, neste caso, representada pelo *Jpivot*. O *Jpivot* transforma em uma página *Web*, que é visualizada pelo mesmo navegador *Web* que fez a solicitação inicial.

A Figura 3.9 mostra um exemplo do retorno de uma consulta pelo *Jpivot*, onde se pode observar uma operação de *drill-down* na dimensão de produtos. Nesta figura, observa-se também a barra de ferramentas padrão do *Jpivot,* onde estão disponíveis todas as opções do *software,* tais como consultas *ad-hoc* em linguagem MDX, operações *drill-down, drill-across*, geração de gráficos, impressão e exportação em formato *xls,*[19] etc...

---

[14] Um portal pode ser definido como o ponto único de acesso, via *Web*, às informações, serviços e aplicações de uma empresa.

[15] JSF. *JavaServer Faces*. Disponível em: <https://javaserverfaces.dev.java.net.>. Acesso em: 20 set. 2005.

[16] STRUTS. *Apache Struts Project*. Disponível em: <http://struts.apache.org>. Acesso em: 20 set. 2005.

[17] Um *Web Server* (Servidor *Web*) é um aplicativo responsável por fornecer ao computador do cliente (usuários de *sites* e páginas eletrônicas), em tempo real, os dados solicitados (WIKIPEDIA, 2005).

[18] O software *Tomcat* desenvolvido pela Fundação *Apache*, permite a execução de aplicações para *Web*. Sua principal característica técnica é estar centrado na linguagem de programação *Java*, mais especificamente nas tecnologias de *Servlets* e de *Java Server Pages* (JSP). TOMCAT. *Apache Tomcat*. Disponível em: <http://tomcat.apache.org>. Acesso em: 06 ago. 2005.

[19] XLS é o formato da planilha de cálculo *Excel* da *Microsoft*. É um formato fechado, binário e proprietário (MICROSOFT, 2005).

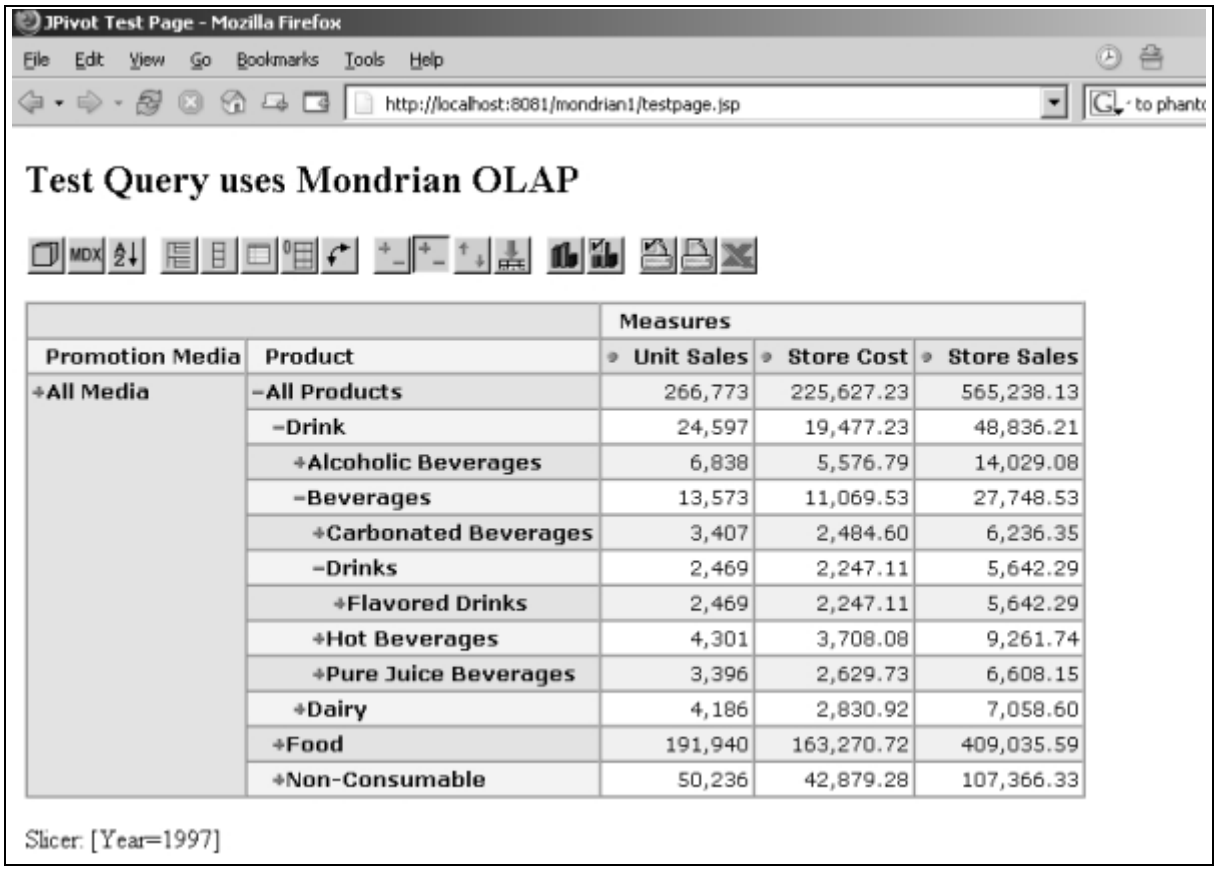


Figura 3.9 *Retorno de uma consulta pelo Jpivot.*
Fonte: (JPIVOT, 2005).

3.3 OpenI

O *OpenI* é uma aplicação BI *Open Source*, em fase inicial de desenvolvimento (atualmente na versão 1.1), que fornece uma solução para análise OLAP. O *OpenI* é disponibilizado como uma aplicação *Web* J2EE, e suporta servidores OLAP compatíveis com o padrão XMLA, incluindo o *Microsoft Analysis Services* e *Mondrian*. O principal objetivo desta ferramenta é permitir ao usuário final analisar os dados de forma fácil e intuitiva. Deve-se salientar também, que esta ferramenta possui ainda pouca documentação disponível.

O *OpenI* é uma aplicação *Web* J2EE com a implementação padrão executando no servidor *Web Apache Tomcat*. Ele publica relatórios analíticos baseados na *Web* a partir de três tipos de servidores de dados: Servidores OLAP, servidores de bancos de dados relacionais e servidores de *data mining.*

A arquitetura do *OpenI* é formada basicamente pelos seguintes componentes:

Componente de conexão (*Connectors*): A função dos conectores é "falar" o idioma

nativo das fontes de dados de análise. Para os bancos de dados relacionais, o *OpenI* utiliza o JDBC. Para os servidores OLAP, ele usa o XMLA como o protocolo padrão de comunicação. Para os serviços de *data mining*, o *OpenI* é integrado com a plataforma *Open Source* de *data mining The R project*[20]. Até a presente *release*, somente a conexão através do XMLA está operacional. (Grifo nosso).

Componente de Relatório (*Report Definitions*): O *OpenI* utiliza linguagens específicas de definição de relatório (RDL - *Report Definition Language*) para definir e explorar os relatórios criados na plataforma. Para os relatórios em base de dados relacionais, ele utiliza a RDL .jrmxl do *JasperReports*[21]. Para relatórios OLAP e *data mining*, ele implementa sua própria RDL baseada em XML.

Componente de interface com o usuário (*User Interface*): A Interface com o usuário disponibilizada pelo *OpenI* reúne outros projetos em uma única plataforma, tornando-a amigável aos usuários não-técnicos, como os analistas de negócio. São utilizados componentes dos projetos *Jpivot* e *JfreeChart*[22], unificando-os como um consistente *framework Web* de navegação.

Segurança: O *OpenI* utiliza a estrutura de segurança da plataforma J2EE, podendo integrá-la à camada de segurança da base de dados de origem, permitindo desta forma um completo controle de permissão de acesso a informação.

A Figura 3.10, mostra em detalhes a arquitetura do *OpenI*, onde pode-se observar os componentes de interface com o usuário, relatório, conexão e segurança ancorados no servidor de aplicações J2EE, além das fontes de dados que podem ser servidores OLAP, bancos de dados relacionais ou *data mining*.

---

[20] THE R PROJECT. *The R Project for Statistical Computing*. Disponível em: <http://www.rproject.org>. Acesso em: 03 out. 2005

[21] JASPERREPORTS. *Free Java reporting library*. Disponível em: <http://jasperreports.sourceforge.net>. Acesso em: 03 out. 2005.

[22] JFREECHART. *A free Java chart library*. Disponível em: <http://www.jfree.org/jfreechart>. Acesso em: 03 out. 2005.

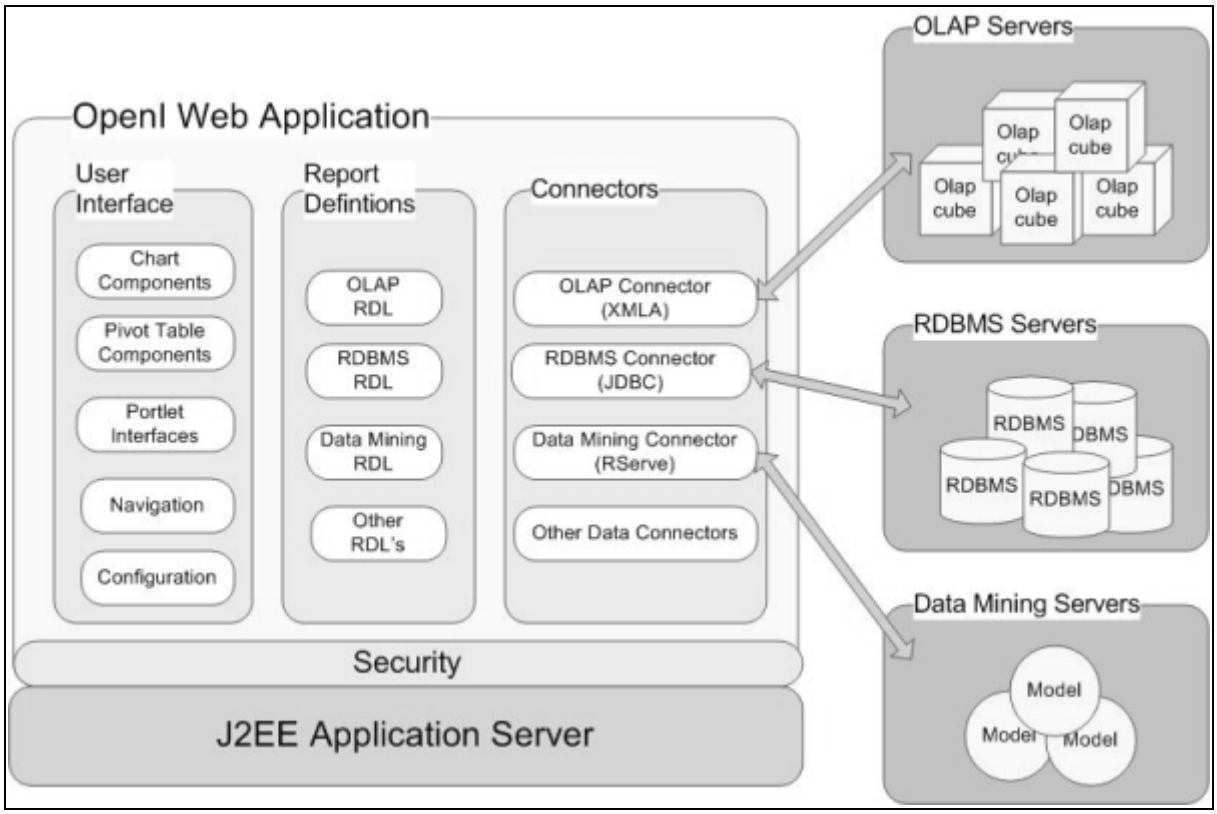


Figura 3.10 *Arquitetura do OpenI.*
Fonte: (OPENI, 2005).

3.4 *Pentaho*

O projeto *Pentaho* BI está em fase inicial de desenvolvimento e pretende ser uma solução *Open Source* completa para aplicações BI, com o objetivo de fornecer às organizações soluções de qualidade para o atendimento das necessidades de BI destas.

O projeto *Pentaho* BI planeja apresentar um conjunto completo de ferramentas de BI, sendo uma solução com suporte a relatórios, análises, *data mining* e *workflow*[23], à médias e grandes empresas, através de uma série de componentes que podem ser distribuídos juntos ou separados. O software será capaz de emitir relatórios financeiros, análises de vendas, análises de produtos e outras funções de negócios. Ele utiliza um método de desenvolvimento, distribuição e suporte, tornando possível o modelo de negócios *Open Source.*

O *Pentaho* BI abrange principalmente as seguintes áreas de aplicação:

Relatórios (*Reporting*): Fornece desde simples relatórios em uma página *Web* até

[23] *Workflow* é a automação do processo de negócio, na totalidade ou em partes, onde documentos, informações ou tarefas são passadas de um participante para o outro para execução de uma ação, de acordo com um conjunto de regras de procedimentos (WIKIPEDIA, 2005).

relatórios de alta qualidade tais como relatórios de indicações financeiras e relatórios ricos em conteúdos como tabelas, gráficos entre outros.

Análises (*Analysis*): Permite consultas *ad hoc*, exploração interativa com operações *slice-and-dice, drill-down* e *pivoting*. Inclui *front-end*[24] gráfico para exploração dos cubos OLAP.

Painéis (*Dashboards*): Reúne relatórios, análises e outras exposições em um único local para simplificar o acesso, podendo ser customizado por usuário, role ou assunto.

***Data mining*:** Descobre relacionamentos ocultos nos dados, que podem ser utilizados para otimizar os processos de negócio e prever resultados futuros. Permite que os resultados sejam exibidos em um formato de fácil entendimento ao usuário.

***Workflow*:** Liga diretamente as medidas de desempenho de negócio aos processos, promovendo um ciclo contínuo de melhorias.

A Figura 3.11 apresenta a arquitetura do *Pentaho* composta pelos módulos de apresentação, de infraestrutura, de integração de dados e a origem destes dados.

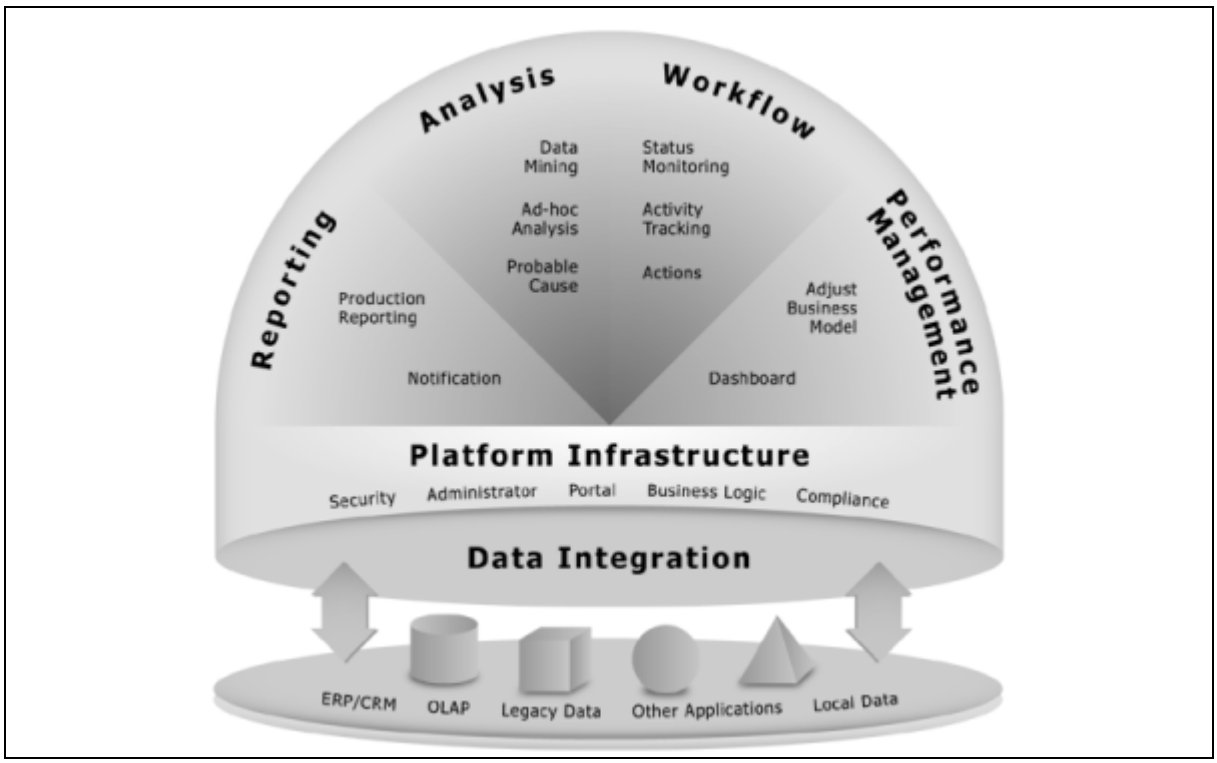


Figura 3.11 *Arquitetura do Pentaho* **.**
Fonte: (PENTAHO, 2005).

[24] *Front-end* é uma interface de usuário que facilita a interação com a aplicação.

O projeto *Pentaho* BI, planeja oferecer uma solução que poderá ser utilizada por desenvolvedores *Java* que utilizarão componentes do projeto para montar rapidamente soluções BI sob medida. Permite também a utilização para empresas desenvolvedoras de software que adicionem as funcionalidades de BI em seus produtos. O projeto pretende ainda permitir aos usuários finais soluções de BI com a qualidade dos softwares tradicionais, porém com um custo bem mais acessível.

Pela análise efetuada, as ferramentas escolhidas para o desenvolvimento da solução foram o *Mondrian* e o *Jpivot*, devido ao seu maior grau de maturidade e principalmente uma maior disponibilidade de documentação em relação às outras ferramentas analisadas. Outro ponto importante para a escolha, foi a comprovação por Brito (BRITO, 2004) sobre a qualidade e flexibilidade apresentadas por estas duas ferramentas.

4 ESTUDO DE CASO

Segundo SIMON (2006), "Estudo de caso é uma técnica de estudo, onde se faz uma pesquisa sobre um caso particular, para tirar conclusões sobre princípios gerais daquele caso específico."

Um estudo de caso é um questionamento empírico que investiga um fenômeno contemporâneo com seus contextos de vida real, quando as fronteiras entre fenômeno e contexto não são claramente evidentes, e nos quais fontes múltiplas de evidência são usadas (Yin, 2001).

Os DMs podem ser utilizados para os mais diversos assuntos, por se tratarem de uma nova técnica para o armazenamento e acesso a informação. Esta nova técnica, adotada na forma como os dados são dimensionados e armazenados, é que a torna mais complexa e, ao mesmo tempo, mais atraente do que os BDs tradicionais. Devido a sua estrutura planejada, o tratamento de grandes volumes de dados históricos não se torna uma tarefa tão desgastante e, muitas vezes, quase impossível de ser feita. Fato este que deve ser mostrado de forma a encorajar empresários de várias áreas a adotarem o uso do DM para tornar eficiente o uso dos dados que possuem, pois, devido a grande quantidade de dados que estes usuários precisam dominar e desprovidos de ferramentas adequadas, sentem-se incapazes de alcançar um diferencial competitivo no mercado.

Neste trabalho, busca-se o desenvolvimento de uma solução utilizando a arquitetura DM e este capítulo tem for finalidade descrever genericamente o funcionamento do departamento comercial de uma indústria calçadista baseado no estudo do software de gestão empresarial SFT, desenvolvido pela empresa Safetech Informática Ltda[1]. Baseada nas características levantadas, será feita a proposta da aplicação para atender a necessidade deste setor.

[1] Safetech Informática Ltda é uma *softwarehouse* que desenvolve sistemas de gestão empresarial com foco na indústria coureiro-calçadista. Localizada à Rua Carlos Biehl, 92 - Sala 9 - Centro - Sapiranga/RS 93800-000.

A estrutura do departamento comercial de uma indústria calçadista é formada por representantes, gerência comercial e pessoal de apoio. Os representantes são responsáveis por realizarem as visitas aos clientes, apresentando os produtos da empresa e anotando os pedidos para repassar à fábrica. A gerência comercial é responsável pelo controle dos representantes, definição de cota de venda e acompanhamento do desempenho das vendas dos representantes e dos produtos. O pessoal de apoio é responsável pelo trabalho burocrático do setor, como atendimento a clientes, cadastramento de produtos, condições de pagamento, interação com os demais setores da empresa como produção e modelagem, entre outros.

A gerência comercial divide o país em regiões de venda. Estas regiões podem ser compostas por uma única unidade da federação (UF), mais de uma unidade em casos de UFs com pouca população ou ainda uma região pode ser somente parte de uma UF em locais com maior população. Cada região é atendida por um representante.

Cada representante recebe uma cota de venda, geralmente semanal, que o mesmo precisa cumprir para manter a fábrica com sua capacidade plena de produção. Esta cota normalmente é definida no início de cada semestre, podendo sofrer alguma alteração durante o decorrer do mesmo. O representante pode possuir prepostos, que auxiliam o mesmo no atendimento aos clientes. Cada representante recebe um percentual de comissão pelas vendas efetuadas, a qual pode ser paga pela empresa no faturamento do pedido ou na liquidação financeira do título gerado por este faturamento.

O representante faz o pedido para o cliente. O pedido contém uma determinada data de entrega, possui também uma condição de pagamento, a transportadora que fará entrega e os itens do pedido, que são os produtos comprados pelo cliente. O pedido pode ser de dois tipos: produção ou pronta entrega. Nos pedidos de produção, a fábrica recebe o pedido, produz, fatura e entrega ao cliente. Já no pedido de pronta entrega, os produtos estão prontos e disponíveis para entrega. Neste caso, os produtos podem ser fabricados para este fim ou serem oriúndos de cancelamento de algum pedido de produção que já se encontrava em processo de fabricação, mas que por algum motivo o cliente não deseja mais receber a mercadoria. Nesta situação, a empresa então disponibiliza este produto para ser vendido como pronta entrega.

As empresas criam coleções de produtos. No caso de calçados femininos, por exemplo, geralmente são criadas duas coleções por ano, uma a cada início de semestre, sendo classificadas normalmente como primavera-verão e outono-inverno. O início de cada coleção

geralmente é marcado por uma feira de nível nacional, como, por exemplo, a Francal[2] e a Couromoda[3], realizadas na cidade de São Paulo, onde são apresentados aos clientes os produtos que fazem parte da nova coleção.

Cada coleção é composta por várias linhas de produtos. Cada linha é composta por vários modelos que possuem características em comum. Estes modelos podem ser produzidos em mais de um material e em uma variada combinação de cores.

Outro processo que precisa ser acompanhado pelo departamento comercial, é a devolução de venda. A devolução é solicitada pelo cliente quando este, por algum motivo, devolve a mercadoria para fábrica. A fábrica avaliará o produto e o motivo alegado pelo cliente para tomar a devida providência, que poderá ser a reposição do produto ou a indenização do cliente mediante compensação financeira.

A gerência comercial necessita fazer o acompanhamento constante do desempenho da área para identificar comportamentos e agir proativamente. Para tanto, precisa analisar as informações sob vários ângulos e diferentes cruzamentos.

Esta é uma das principais dificuldades enfrentadas pelos desenvolvedores de soluções para esta área, que utilizam ferramentas tradicionais para criar relatórios estáticos. Estas ferramentas somente apresentam a informação em um formato pré-determinado, não permitindo ao usuário uma maior interatividade. Além disto, é necessária a criação de um número muito grande de relatórios nos mais variados formatos para atender às necessidades dos usuários, e estes, em diversas situações, precisam emitir vários relatórios para finalmente chegar à informação desejada.

Com base neste cenário, é proposta a construção de um DM, composto pelas tabelas de fatos e dimensões caracterizadas a seguir:

4.1 Tabelas de fatos

O modelo é composto por três tabelas de fatos: VENDA, DEVOLUCAO e VENDA_DEVOLUCAO, as quais são detalhadas a seguir: (Grifo nosso).

---

[2] FRANCAL. ***Francal Feiras***. Disponível em: <http://www.francal.com.br>. Acesso em: 16 fev. 2006.
[3] COUROMODA. ***Couromoda.com***. Disponível em: <http://www.couromoda.com.br>. Acesso em: 16 fev. 2006.

4.1.1 Fato VENDA

A tabela de fato VENDA contém as informações sobre as vendas da empresa e é composta pelas chaves estrangeiras paras as dimensões descritas na próxima sessão e pela suas medidas. A seguir, segue uma explicação sobre os atributos que compõem esta tabela de fato. (Grifo nosso).

DATA_INCLUSAO – Data de inclusão do pedido;

DATA_ENTREGA – Data de entrega do pedido ao cliente;

CODIGO_CLIENTE – Código do cliente que fez a compra;

CODIGO_REPRESENTANTE – Código do representante que fez a venda ao cliente;

CODIGO_TRANSPORTADORA – Código da transportadora responsável pela entrega do pedido ao cliente;

CODIGO_PRODUTO – Código do produto comprado pelo cliente;

TIPO_PEDIDO – Tipo do pedido (produção ou pronta entrega);

CODIGO_CIDADE – Código da cidade do domicílio do cliente;

CODIGO_CONDICAO_PAGAMENTO – Código da condição de pagamento utilizada no pedido;

CODIGO_PREPOSTO – Código do preposto que fez o atendimento ao cliente;

VENDOR – Indica se o pedido é do tipo vendor[4] ou não;

QUANTIDADE – Medida que identifica a quantidade vendida;

VALOR_LIQUIDO – Medida que determina o valor líquido do produto vendido;

VALOR_BRUTO – Medida que determina o valor bruto do produto vendido;

[4] Vendor é uma alternativa de financiamento a vendas que permite substituir o financiamento direto, do fornecedor aos seus clientes, pelo financiamento bancário (UNIBANCO, 2006).

PERCENTUAL_DESCONTO – Medida que informa o percentual de desconto concedido no pedido;

PERCENTUAL_COMISSAO – Medida que informa o percentual de comissão paga ao representante que fez o pedido.

4.1.2 Fato DEVOLUCAO

A tabela de fato DEVOLUCAO contêm as informações sobre as devoluções de vendas feitas pelos clientes. Uma explicação sobre os atributos que compõem esta tabela de fato é feita a seguir: (Grifo nosso).

CODIGO_PRODUTO – Código do produto devolvido;

CODIGO_CLIENTE – Código do cliente que fez a devolução;

CODIGO_REPRESENTANTE – Código do representante que atende o cliente que fez a devolução;

CODIGO_CIDADE – Código da cidade do domicílio do cliente;

CODIGO_MOTIVO – Código do motivo da devolução alegado pelo cliente;

DATA_DEVOLUCAO – Data da devolução;

QUANTIDADE – Medida que identifica a quantidade devolvida;

VALOR – Medida que determina o valor da devolução.

4.1.3 Fato VENDA_DEVOLUCAO

A tabela de fato VENDA_DEVOLUCAO é o resultado do cruzamento das informações da tabela de fato VENDA com a tabela de fato DEVOLUCAO, com a finalidade de facilitar a análise das devoluções em relação às vendas. Fisicamente no DM, este fato não é uma tabela, e sim uma visão[5]. Esta visão é composta pelos seguintes atributos: (Grifo nosso).

DATA – Data da ocorrência da venda ou devolução;

[5] Uma visão pode ser definida como uma maneira alternativa de observação de dados de uma ou mais entidades (tabelas), que compõem uma base de dados. Pode ser considerada como uma tabela virtual ou uma consulta armazenada.

CODIGO_PRODUTO – Código do produto;

CODIGO_CLIENTE – Código do cliente;

CODIGO_REPRESENTANTE – Código do representante;

CODIGO_CIDADE – Código da cidade do cliente;

QUANTIDADE_DEVOLVIDA – Medida que identifica a quantidade devolvida;

QUANTIDADE_VENDIDA – Medida que identifica a quantidade vendida.

4.2 Tabelas de dimensão

A tabelas de fatos do modelo proposto, descritas anteriormente, relacionam-se com dez dimensões: CIDADE, CLIENTE, CONDICAO_PAGAMENTO, DATA, MOTIVO_DEVOLUCAO, PREPOSTO, PRODUTO, REPRESENTANTE, TIPO_PEDIDO e TRANSPORTADORA. Estas dimensões apresentam funções e particularidades específicas, que são explicadas a seguir: (Grifo nosso).

4.2.1 Dimensão CIDADE

Esta dimensão armazena dados sobre as cidades. Os atributos que compõem esta dimensão são descritos a seguir:

CODIGO_CIDADE – Código da cidade;

NOME – Nome da cidade;

UF – Unidade da federação onde a cidade está localizada;

REGIAO – Região geográfica da cidade;

CAPITAL_INTERIOR – Indica se a cidade é capital ou interior;

REGIAO_METROPOLITANA – Indica se a cidade faz parte de alguma região metropolitana.

4.2.2 Dimensão CLIENTE

Esta dimensão tem por finalidade armazenar dados relativos aos clientes. A seguir

são listados os atributos que caracterizam esta dimensão.

CODIGO_CLIENTE – Código do cliente;

NOME – Nome do cliente;

GRUPO_ECONOMICO – Grupo econômico (rede de lojas) ao qual um cliente pode pertencer;

CONCEITO – Conceito do cliente determinado pela empresa.

4.2.3 Dimensão CONDICAO_PAGAMENTO

Esta dimensão armazena os dados das condições de pagamento utilizadas nos pedidos, identificados pelos seguintes atributos:

CODIGO_CONDICAO_PAGAMENTO – Código da condição de pagamento;

DESCRICAO – Descrição da condição de pagamento;

PRAZO_MEDIO – Prazo médio da condição de pagamento. Por exemplo: 30, 60, 90 dias, etc...;

PERCENTUAL_DESCONTO – Percentual de desconto da condição de pagamento.

4.2.4 Dimensão DATA

Esta dimensão armazena a hierarquia de datas relacionadas às datas de inclusão, entrega e devolução utilizadas nas tabelas de fatos. Esta dimensão é composta pelos atributos:

DATA – Data armazenada no formato dd/mm/aaaa (dia/mês/ano);

DIA_SEMANA – Número do dia da semana;

NOME_DIA_SEMANA – Nome do dia da semana;

DIA_MES – Número do dia do mês;

SEMANA_MES – Número da semana no mês;

QUINZENA – Número da quinzena no mês;

MES – Número do mês no ano;

NOME_MES – Nome do mês;

TRIMESTRE – Número do trimestre no ano;

SEMESTRE – Número do semestre no ano;

ANO – Ano.

4.2.5 Dimensão MOTIVO_DEVOLUCAO

A dimensão MOTIVO_DEVOLUCAO contém as informações sobre os motivos utilizados para a devolução de venda, identificadas pelos atributos: (Grifo nosso).

CODIGO_MOTIVO – Código do motivo de devolução;

MOTIVO_DEVOLUCAO – Descrição do motivo da devolução.

4.2.6 Dimensão PREPOSTO

A dimensão PREPOSTO contém as informações sobre os prepostos dos representantes, identificados pelos seguintes atributos: (Grifo nosso).

CODIGO_REPRESENTANTE – Código do representante do preposto;

NOME_REPRESENTANTE – Nome do representante;

CODIGO_PREPOSTO – Código do preposto;

NOME – Nome do preposto.

4.2.7 Dimensão PRODUTO

A dimensão PRODUTO armazena as informações sobre a estrutura de produtos que são produzidos pela empresa e disponibilizados aos seus clientes. Os produtos fazem parte de uma estrutura cujo topo é a marca do produto. A seguir, segue uma explicação sobre os atributos que compõem esta tabela de dimensão. (Grifo nosso).

CODIGO_PRODUTO – Código do produto;

REFERENCIA – Código da referência;

CODIGO_LINHA – Código da linha da referência;

DESCRICAO_LINHA – Descrição da linha da referência;

CODIGO_MARCA – Código da marca da referência;

DESCRICAO_MARCA – Descrição da marca da referência;

TIPO_PRODUTO – Descrição do tipo de produto;

MATERIAL – Descrição do material principal no qual a referência é fabricada;

COR – Descrição da cor principal que caracteriza a referência;

FAMILIA – Descrição da família de produto a qual pertence a referência;

COLECAO – Descrição da coleção de produto a qual a referência faz parte.

4.2.8 Dimensão REPRESENTANTE

A dimensão REPRESENTANTE contém as informações sobre os representantes que fazem os pedidos para os clientes. Uma explicação sobre os atributos que compõem esta dimensão é feita a seguir. (Grifo nosso).

CODIGO_REPRESENTANTE – Código do representante;

NOME – Nome do representante.

4.2.9 Dimensão TIPO_PEDIDO

A dimensão TIPO_PEDIDO armazenada as informações sobre os tipos de pedidos que podem ser feitos pelo cliente. A seguir, segue uma explicação sobre os atributos que compõem esta tabela de dimensão. (Grifo nosso).

TIPO_PEDIDO – Código do tipo de pedido;

DESCRICAO_TIPO_PEDIDO – Descrição do tipo de pedido.

4.2.10 Dimensão TRANSPORTADORA

A dimensão TRANSPORTADORA contém as informações das transportadoras responsáveis pelo transporte e a entrega da mercadoria ao cliente, caracterizada pelos seguintes atributos: (Grifo nosso).

CODIGO_TRANSPORTADORA – Código da transportadora;

NOME – Nome da transportadora.

A figura 4.1 mostra o modelo construído:

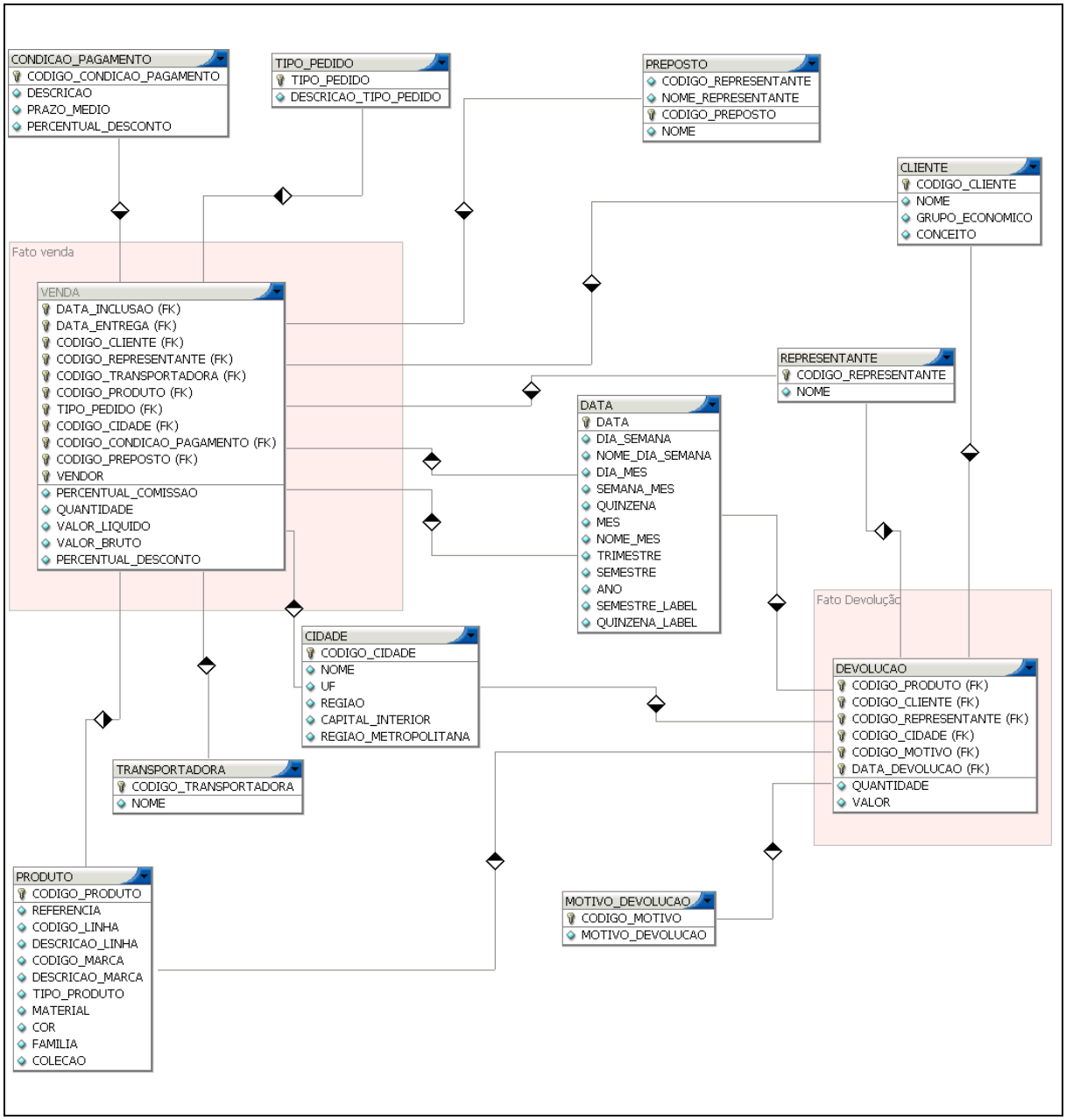


Figura 4.1 *Modelo proposto.*

Após a descrição do modelo do DM, no próximo capítulo será descrita em detalhes a ferramenta OLAP proposta para atender às necessidades de informação do departamento comercial da indústria calçadista.

5 SOLUÇÃO PROPOSTA

Este capítulo tem por finalidade apresentar a ferramenta proposta, utilizando como base a estrutura do DM descrito no capítulo anterior. A seguir serão listadas as características desta solução bem como os passos para instalação e configuração da aplicação e a apresentação de exemplos.

5.1 Características gerais da solução

A solução foi desenvolvida utilizando-se o ambiente *Eclipse*, que foi escolhido porque além de ser gratuito, é também muito versátil, devido a infinidade de *plugins* que são oferecidos para ele.

O DM foi construído seguindo-se a arquitetura de DM independente, ou seja, inicialmente será um DM sem qualquer integração com outros DMs ou DWs que a empresa possa utilizar.

Este DM foi modelado conforme a técnica *Star Schem*a, utilizando a arquitetura de armazenamento ROLAP, ou seja, os dados estão armazenados em um banco de dados relacional. A camada OLAP é desempenhada pelo software *Mondrian*, que faz o acesso a esta base de dados através do envio de consultas na linguagem MDX. Ao receber o resultado da consulta, o *Mondrian* monta o cubo de dados, que por sua vez, é acessado pelo *Jpivot*, camada responsável pela apresentação dos dados ao usuário.

Esta solução foi desenvolvida para a arquitetura *Web*, e é disponibilizada através do servidor *Apache Tomcat,* que é um servidor de aplicações *Java* para *Web*. Ele é *software* livre e de código aberto desenvolvido dentro do conceituado projeto *Apache* e oficialmente endossado pela *Sun*[1] como a implementação de referência para as tecnologias *Java Servlet*[2] e

[1] SUN. ***Sun Microsystems***. Disponível em: <http://www.sun.com>. Acesso em: 20 mar. 2006.
[2] Servlets são programas que rodam no lado do servidor e que estendem as funcionalidades do servidor *Web*, gerando páginas de forma dinâmica e interagindo com os clientes (JAVA, 2005).

JavaServer Pages (JSP)[3]. O *Tomcat* é robusto e eficiente o suficiente para ser utilizado em um ambiente de produção (WIKIPEDIA, 2005). A arquitetura da aplicação é observada na Figura 5.1, exibida a seguir.

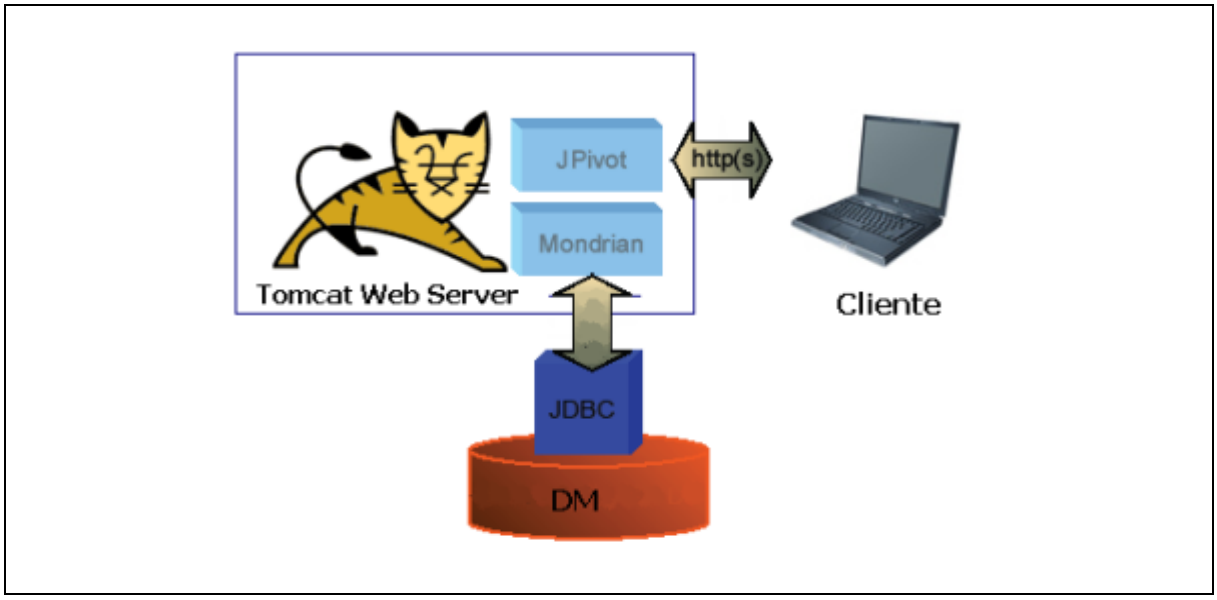


Figura 5.1 *Arquitetura da aplicação* .

O *Tomcat* é inteiramente escrito em *Java* e, portanto, necessita de uma *Java Virtual Machine* (JVM) para ser executado. Assim, é necessário ter a plataforma *Java* Padrão, *Java* 2 *Platform Standard Edition* (J2SE), previamente instalada no servidor onde o *Tomcat* será também instalado (TOMCAT, 2005).

Os dados do DM serão oriundos do módulo comercial do sistema de gestão SFT. O software desenvolvido foi denominado de *SFT Analysis*[4] e complementará a solução disponibilizada pela empresa Safetech Informática LTDA, visando no futuro, ser um módulo que esta possa oferecer a seus clientes.

O DM foi criado no mesmo servidor de banco de dados do sistema SFT. O banco de dados utilizado é o *Oracle 9i*[5]. Porém este não é um requisito obrigatório do sistema, pois como o banco de dados é acessado pelo *Mondrian* através de JDBC, pode ser utilizado qualquer banco de dados compatível com este padrão, incluindo os bancos de dados *freeware*

[3] JSP (*Java Server Pages*) é uma tecnologia para desenvolvimento de aplicações *Web* com a vantagem da portabilidade de plataforma podendo ser executado em diversos sistemas operacionais. Ela permite ao desenvolvedor de *sites* produzir aplicações que permitam o acesso a banco de dados, o acesso a arquivos-texto, a captação de informações a partir de formulários, a captação de informações sobre o visitante e sobre o servidor, o uso de variáveis e *loops* entre outras (WIKIPEDIA, 2005).
[4] Nome dado pelo autor.
[5] ORACLE. ***Oracle Corporation***. Disponível em: < http://www.oracle.com>. Acesso em: 06 mar. 2006.

mais conhecidos como o *Mysql*[6] e o *Postgresql*[7].

5.2 Extração, transformação e carga dos dados no DM

A etapa de extração, transformação e carga do DM será feita através da execução de *scripts* SQLs que farão a leitura dos dados na base de dados do sistema SFT e farão a inserção no DM. A carga dos dados no DM será sempre uma carga total, ou seja, os dados existentes serão eliminados, e após este procedimento, todo o DM será populado novamente. Optou-se por este procedimento porque o volume de dados não é muito extenso e durante as simulações desta carga o tempo dispensado nesta tarefa foi bastante baixo, inferior a uma hora de duração, o que permite que esta carga possa ser programada para rodar diariamente, fora de horário de utilização do sistema transacional da empresa, com absoluta tranqüilidade.

5.3 Instalação e configuração da aplicação

Como a aplicação foi desenvolvida baseada na tecnologia *Java*, a mesma herda uma das principais característica da linguagem: a multiplataforma. Desta forma, esta aplicação pode ser instalada nas plataformas para as quais existe uma JVM disponível.

A primeira etapa é a instalação do servidor *Web Apache Tomcat*. A versão utilizada para o desenvolvimento foi a 5.6.16. O *Tomcat* deve ser instalado no computador que deverá ser configurado como servidor *Web* com acesso via *Intranet* ou *Internet*.

Após a instalação do *Tomcat*, deve-se cadastrar os usuários que terão acesso a aplicação. Este cadastro é feito no arquivo \conf\tomcat-users.xml dentro do diretório raiz de instalação do *Tomcat*.

Os dados da aplicação estarão armazenados em um banco de dados relacional, que deve ser acessado através de um driver JDBC. Este por sua vez, deve ser instalado no *Tomcat* no diretório \common\lib. Na aplicação desenvolvida, foi utilizado o *driver ojdbc14.jar*, que é o *driver* disponibilizado pela *Oracle* para acesso ao banco de dados *Oracle 9i*.

A última etapa da instalação é disponibilização da aplicação no servidor *Tomcat*, no diretório \webapps\sft, com a estrutura de diretórios conforme verificado na figura 5.2, mostrada a seguir:

---

[6] MYSQL. ***Mysql AB***. Disponível em: <http://www.mysql.com>. Acesso em: 03 mar. 2006.
[7] POSTGRESQL. ***Postgresql***. Disponível em: <http://www.postgresql.org>. Acesso em: 03 mar. 2006

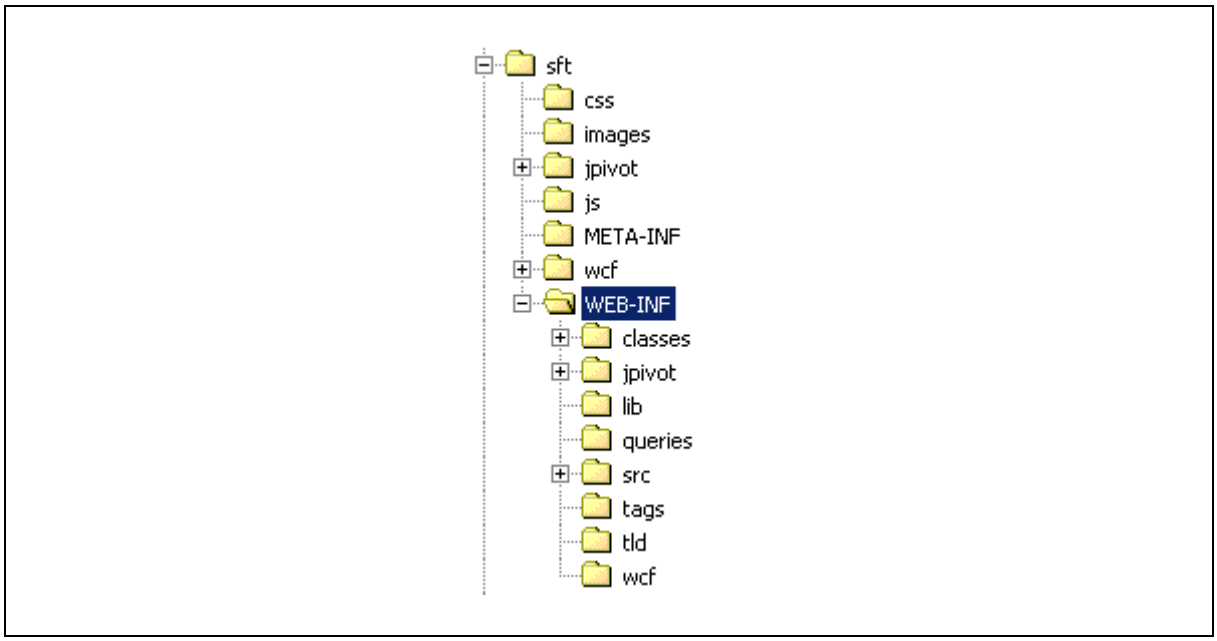


Figura 5.2 *Estrutura da aplicação disponibilizada no Tomcat* .

5.4 Executando a aplicação

Para executar a aplicação, é necessário que o servidor *Web* esteja iniciado. No exemplo ilustrado, o servidor foi instalado e configurado localmente, no endereço http://localhost:8080/sft.

A aplicação contém uma página de *login*, para permitir que somente usuários previamente autorizados possam utilizar o sistema.

O primeiro passo é executar um *Web browser* e navegar para o endereço no qual o servidor *Web* está inicializado. Neste instante a página de *login* é exibida conforme a figura 5.3, exibida a seguir:

Figura 5.3 *Tela de login do SFT Analysis* **.**

Após o *login*, será exibida a página principal da aplicação, onde é disponibilizado um menu com os cubos criados e as consultas já cadastradas pelo usuário, conforme figura 5.4.

Figura 5.4 *Tela principal do SFT Analysis.*

A aplicação disponibiliza os cubos para que o usuário possa montar as consultas conforme sua necessidade. Além de permitir a navegação nos cubos, é permitido ao usuário gravar as consultas efetuadas.

A gravação da consulta é feita em uma tabela no mesmo banco de dados onde estão as tabelas do modelo dimensional. Esta tabela é denominada CONSULTA e é composta pelas colunas: (Grifo nosso).

USUARIO: Usuário que criou a consulta;

DESCRICAO: Descrição dada pelo usuário a cada consulta criada;

CONSULTA_MDX: Consulta criada pelo usuário traduzida para a linguagem MDX;

CONTADOR: Número de utilizações da consulta;

DATA_ULTIMA_UTILIZACAO: Data da última utilização da consulta.

Além da gravação da consulta, a aplicação salva um acumulador do número de vezes que a consulta foi utilizada. Outro dado que também é mantido, é a data da última utilização desta consulta. Estas informações têm o objetivo de fornecer uma estatística ao usuário sobre quais são as consultas mais utilizadas e quais foram recentemente mais acessadas. Esta opção fica disponível ao usuário através do menu da aplicação, conforme observado na Figura 5.5.

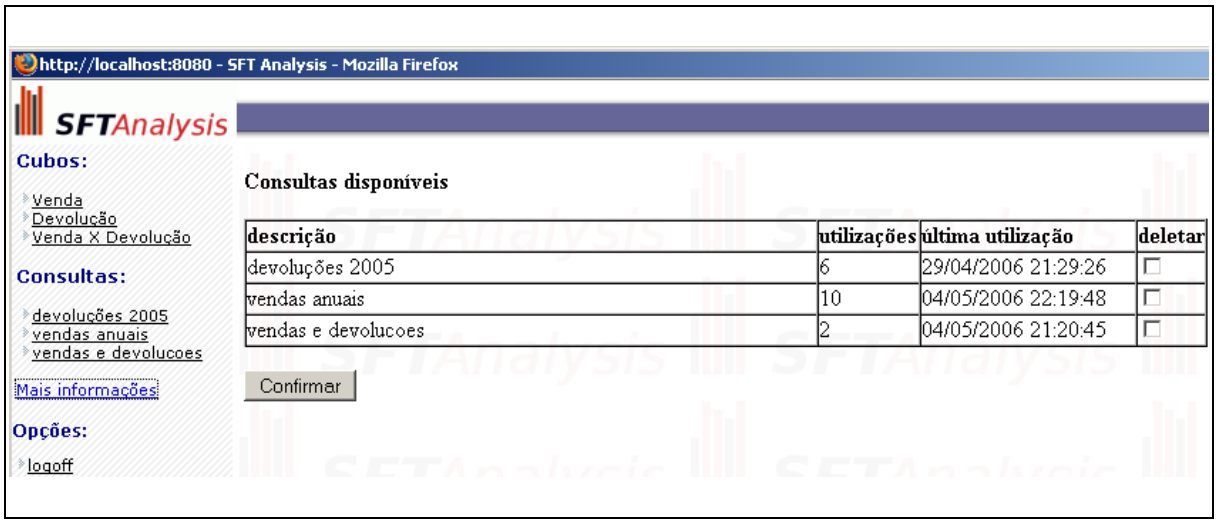


Figura 5.5 *Consultas disponíveis ao usuário.*

A manipulação dos dados nesta tabela é feita por uma classe *Java* criada especificamente para esta função. Esta classe é responsável pela conexão com o banco de dados, inserção e exclusão dos dados. Esta classe utiliza o mesmo *driver* JDBC usado pelo *Mondrian* para conexão com o banco de dados.

Após o usuário efetuar o *login* na aplicação, as consultas criadas por ele são exibidas, ficando disponíveis para execução a qualquer momento que o mesmo desejar. A figura 5.6 exibe o resultado de uma consulta no cubo VENDA, onde o usuário selecionou visualizar a dimensão DATA ENTREGA e a medida QUANTIDADE. (Grifo nosso).

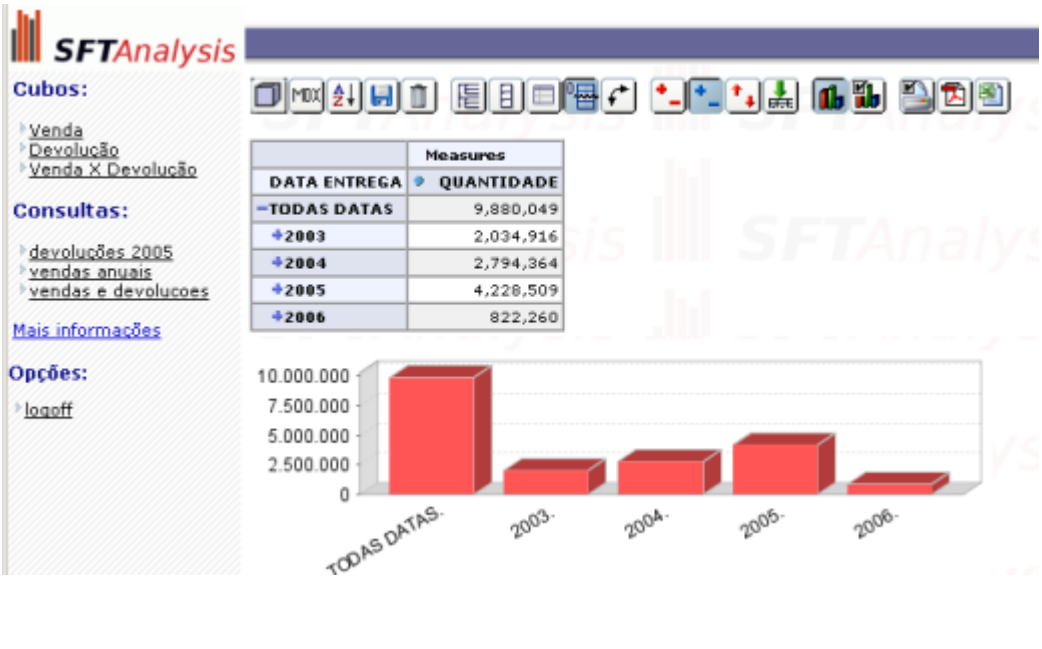


Figura 5.6 *Exemplo do retorno de uma consulta.*

A aplicação possui uma barra de ferramentas de navegação com a finalidade de tornar amigável e fácil ao usuário a sua operação. Algumas funções são disponibilizadas como padrão pelo *Jpivot*, outras foram criadas especificamente para a aplicação.

Cabe ressaltar que os exemplos mostrados nas figuras ilustradas a seguir, foram obtidos com dados gerados de forma aleatória, não retratando, de forma alguma, informações reais de qualquer empresa.

A seguir serão descritas as funcionalidades da barra de ferramentas de navegação, onde cada botão é indicado por um número e é descrita a sua funcionalidade. A figura 5.7 exibe a barra de ferramentas com os indicadores para posterior descrição (ROSA, 2005).

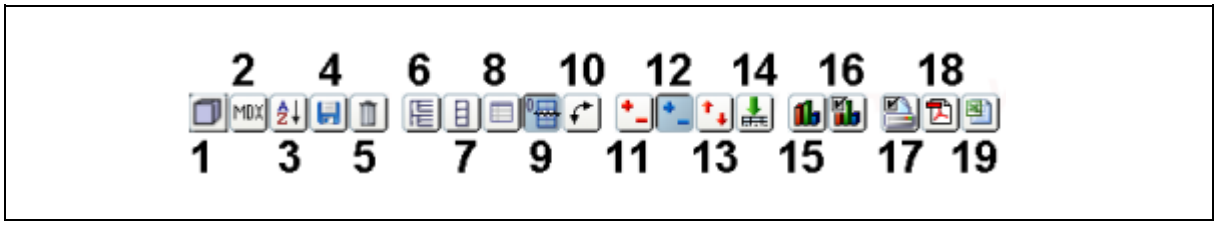


Figura 5.7 *Barra de ferramentas do SFT Analysis.*
Fonte: Adaptação do autor segundo (ROSA, 2005).

1- Detalhamento do cubo, onde é possível selecionar exatamente as dimensões e medidas desejadas, para a melhoria da visualização. Nesta opção é possível também selecionar a ordem das colunas exibidas bem como aplicar filtros para selecionar somente a faixa de dados desejada. A Figura 5.8 exemplifica a utilização desta opção.

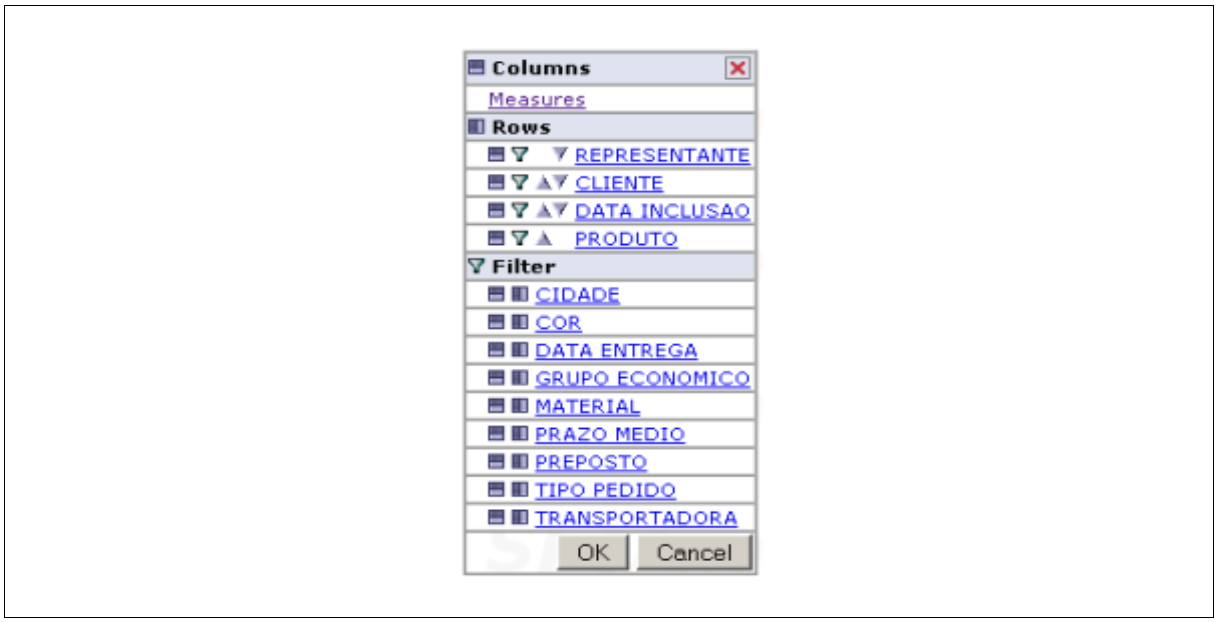


Figura 5.8 *Detalhamento do cubo de dados.*

2- Linguagem MDX, para usuários mais avançados e que desejam analisar ou modificar a linguagem de acesso ao cubo. Esta opção exibe a consulta na linguagem MDX, relativa a consulta efetuada, permitindo a edição do código para uma nova execução, conforme pode ser observado na Figura 5.9.

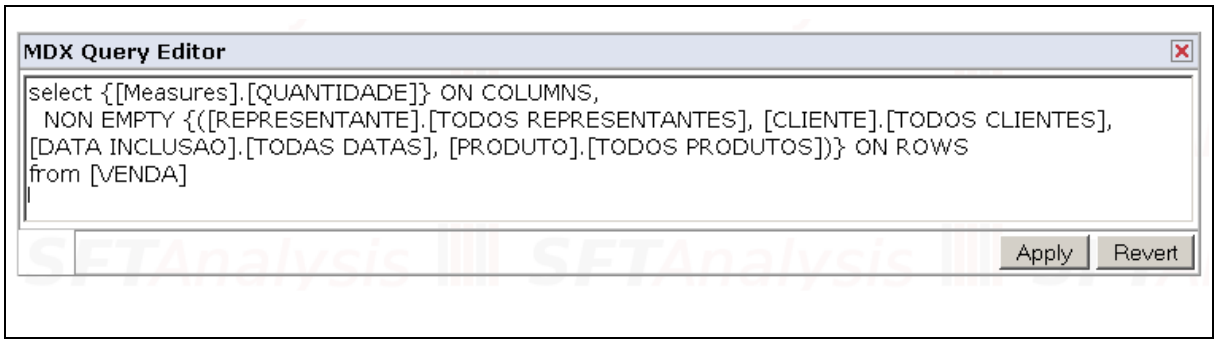


Figura 5.9 *Editor da consulta em linguagem MDX.*

3- Configuração do modo de ordenação das consultas. Permite a ordenação de consultas e também a definição do número dos maiores ou menores valores a serem analisados. A Figura 5.10 mostra esta opção.

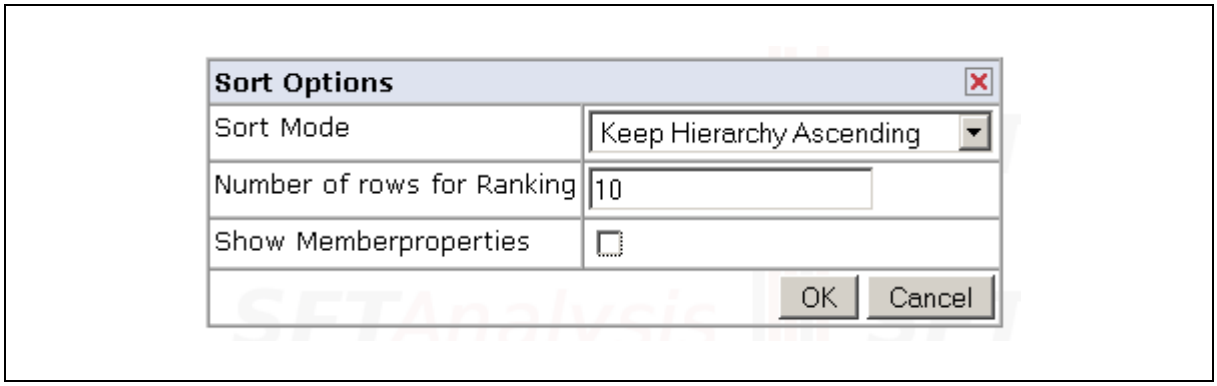


Figura 5.10 *Modo de ordenação das consultas.*

4- Gravar as consultas. Esta é uma opção criada especificamente para a aplicação (não existe no *Jpivot*) possibilitando ao usuário gravar as consultas para posterior execução. Esta implementação é feita através da captura da consulta na linguagem MDX e a posterior gravação na base de dados, para disponibilizar ao usuário. A Figura 5.11 exemplifica o momento da gravação de uma consulta onde o usuário define uma descrição para salvar a consulta na base de dados.

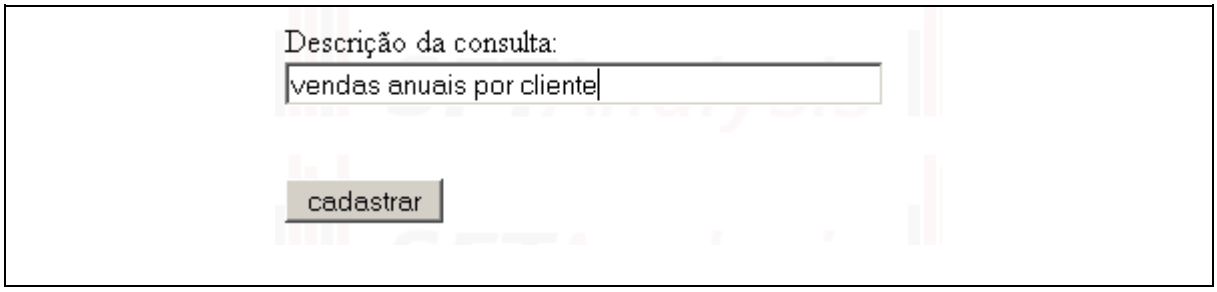


Figura 5.11 *Cadastro de consultas pelo usuário.*

5- Excluir as consultas. Esta opção, também criada especificamente para aplicação, permite ao usuário excluir as consultas por ele criadas no ítem anterior. Conforme se pode observar na Figura 5.12, ao clicar neste botão, todas as consultas cadastradas pelo usuário são listadas, permitindo que ele selecione quais deseja excluir.

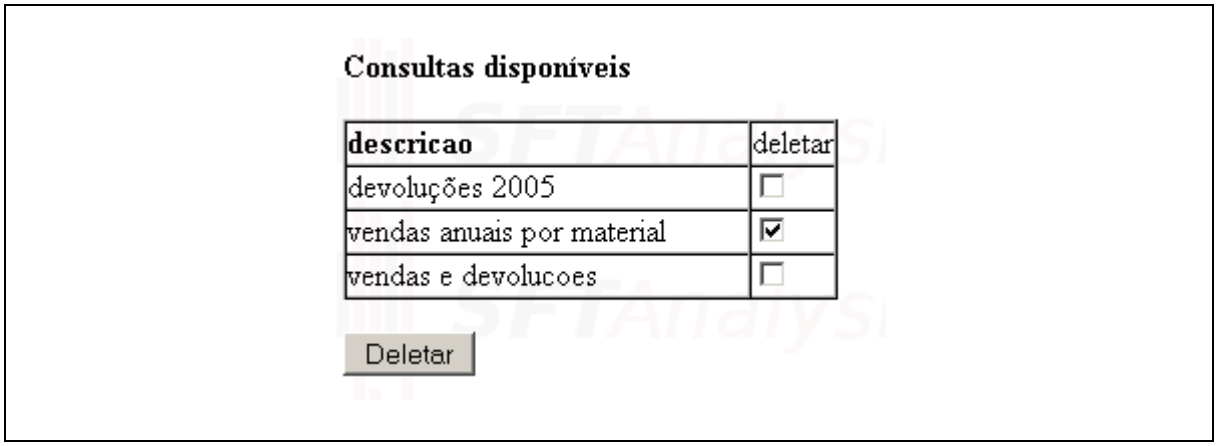


Figura 5.12 *Exclusão das consultas.*

6- Mostra ou não os membros superiores, considerados pais dentro da estrutura hierárquica. Exibe uma linha com os dados dos membros superiores, conforme podemos observar na Figura 5.13, onde, na coluna PRODUTO, é exibido o membro superior TODOS PRODUTOS. (Grifo nosso).

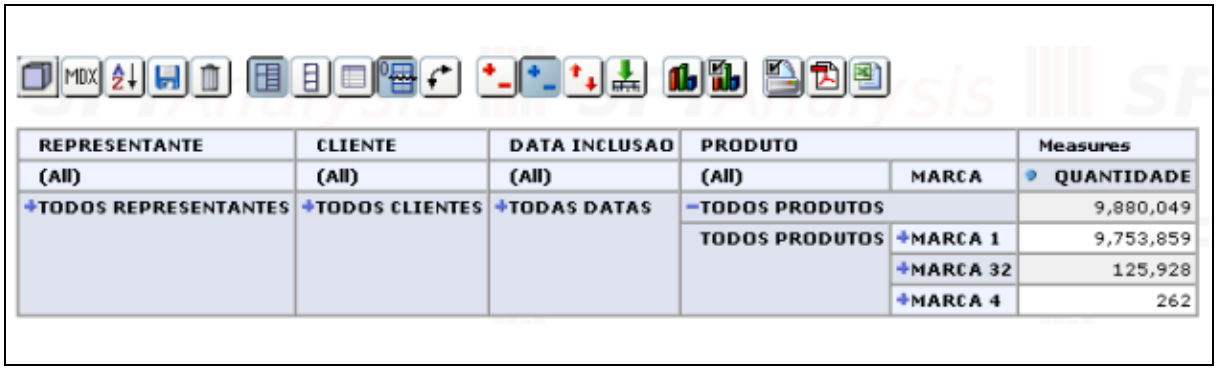
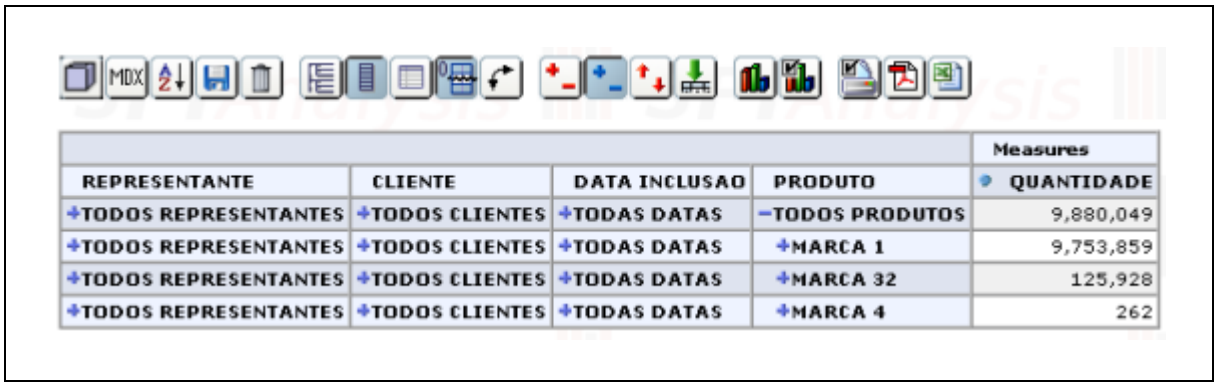


Figura 5.13 *Exibição dos membros pais.*

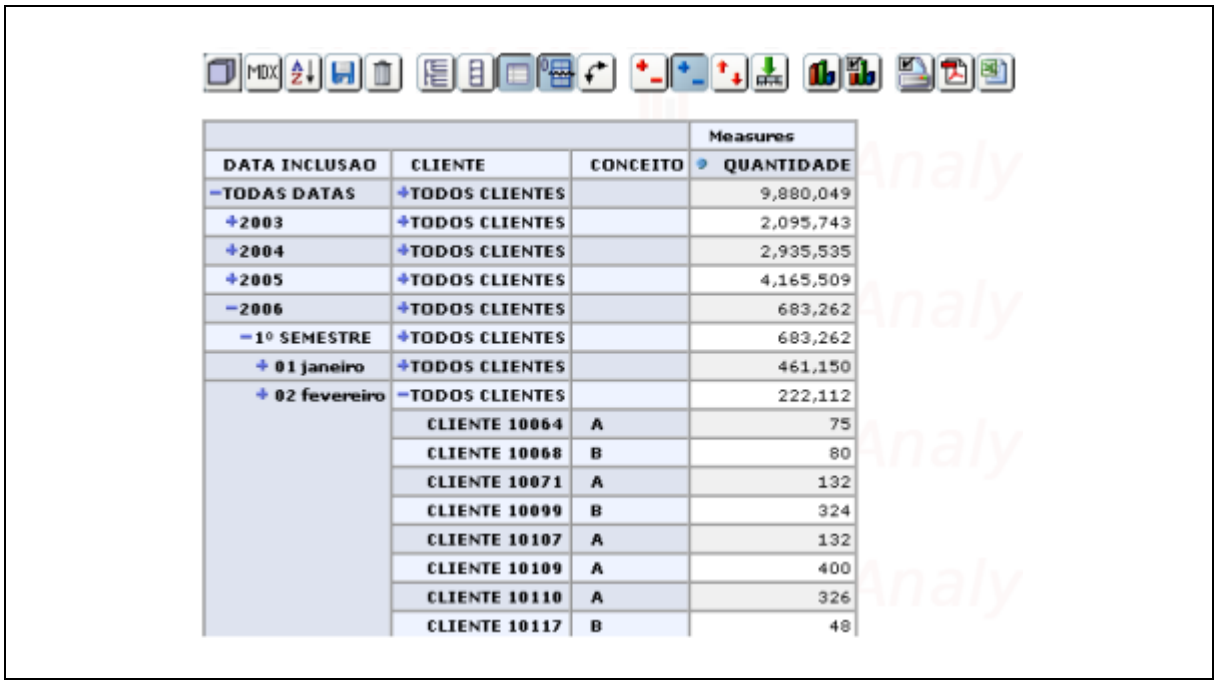
7- Suprime ou não *labels* repetidos. Esta opção permite, por padrão, que os *labels* repetidos sejam suprimidos, para facilitar visualização dos dados. A Figura 5.14 exibe o resultado de uma consulta onde o usuário optou não suprimir os *labels* repetidos. Percebe-se nesta figura uma certa poluição visual, dificultando a leitura da informação.



				Measures
REPRESENTANTE	CLIENTE	DATA INCLUSAO	PRODUTO	QUANTIDADE
+TODOS REPRESENTANTES	+TODOS CLIENTES	+TODAS DATAS	-TODOS PRODUTOS	9,880,049
+TODOS REPRESENTANTES	+TODOS CLIENTES	+TODAS DATAS	+MARCA 1	9,753,859
+TODOS REPRESENTANTES	+TODOS CLIENTES	+TODAS DATAS	+MARCA 32	125,928
+TODOS REPRESENTANTES	+TODOS CLIENTES	+TODAS DATAS	+MARCA 4	262

Figura 5.14 *Exibição de labels repetidos.*

8- Mostra as propriedades das dimensões do cubo, caso existam. Na Figura 5.15 podemos observar a coluna CONCEITO que é uma propriedade da dimensão CLIENTE. (Grifo nosso).

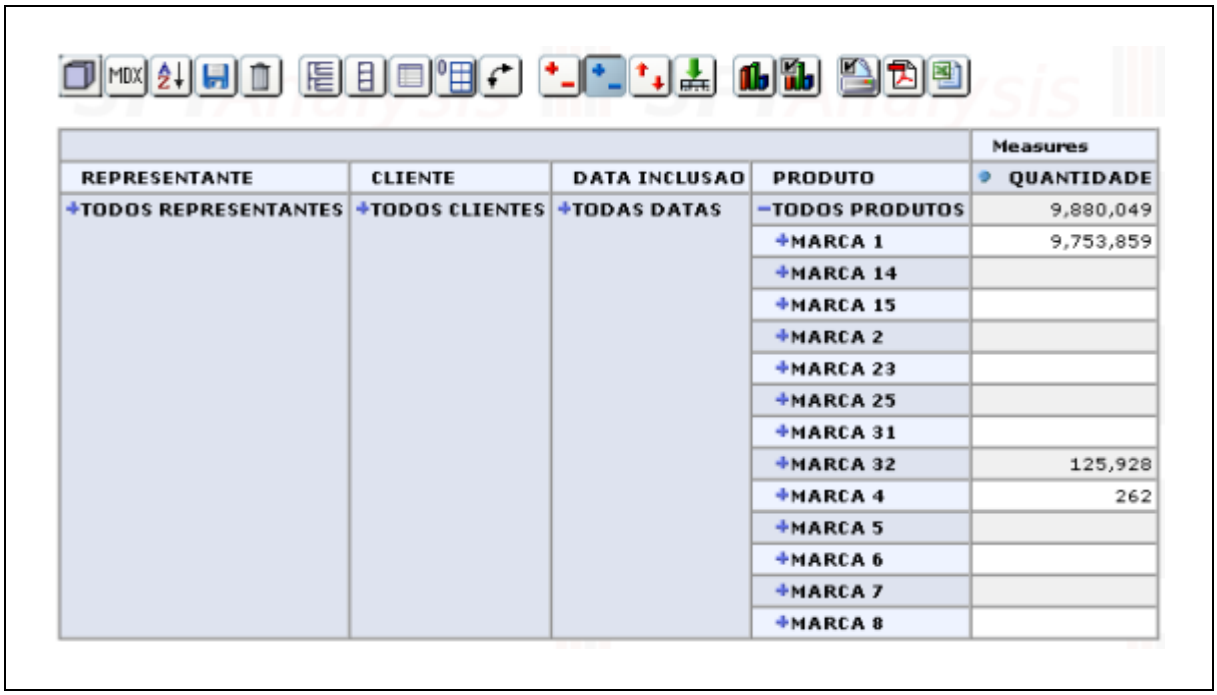


			Measures
DATA INCLUSAO	CLIENTE	CONCEITO	QUANTIDADE
-TODAS DATAS	+TODOS CLIENTES		9,880,049
+2003	+TODOS CLIENTES		2,095,743
+2004	+TODOS CLIENTES		2,935,535
+2005	+TODOS CLIENTES		4,165,509
-2006	+TODOS CLIENTES		683,262
-1º SEMESTRE	+TODOS CLIENTES		683,262
+ 01 janeiro	+TODOS CLIENTES		461,150
+ 02 fevereiro	-TODOS CLIENTES		222,112
	CLIENTE 10064	A	75
	CLIENTE 10068	B	80
	CLIENTE 10071	A	132
	CLIENTE 10099	B	324
	CLIENTE 10107	A	132
	CLIENTE 10109	A	400
	CLIENTE 10110	A	326
	CLIENTE 10117	B	48

Figura 5.15 *Exibição das propriedades das dimensões.*

9- Suprime ou não colunas ou linhas vazias. Com esta opção, permite que somente as linhas ou colunas com informações válidas sejam exibidas. A Figura 5.16 mostra o resultado

de uma consulta onde as linhas vazias não foram suprimidas.



				Measures
REPRESENTANTE	CLIENTE	DATA INCLUSAO	PRODUTO	QUANTIDADE
+TODOS REPRESENTANTES	+TODOS CLIENTES	+TODAS DATAS	−TODOS PRODUTOS	9,880,049
			+MARCA 1	9,753,859
			+MARCA 14	
			+MARCA 15	
			+MARCA 2	
			+MARCA 23	
			+MARCA 25	
			+MARCA 31	
			+MARCA 32	125,928
			+MARCA 4	262
			+MARCA 5	
			+MARCA 6	
			+MARCA 7	
			+MARCA 8	

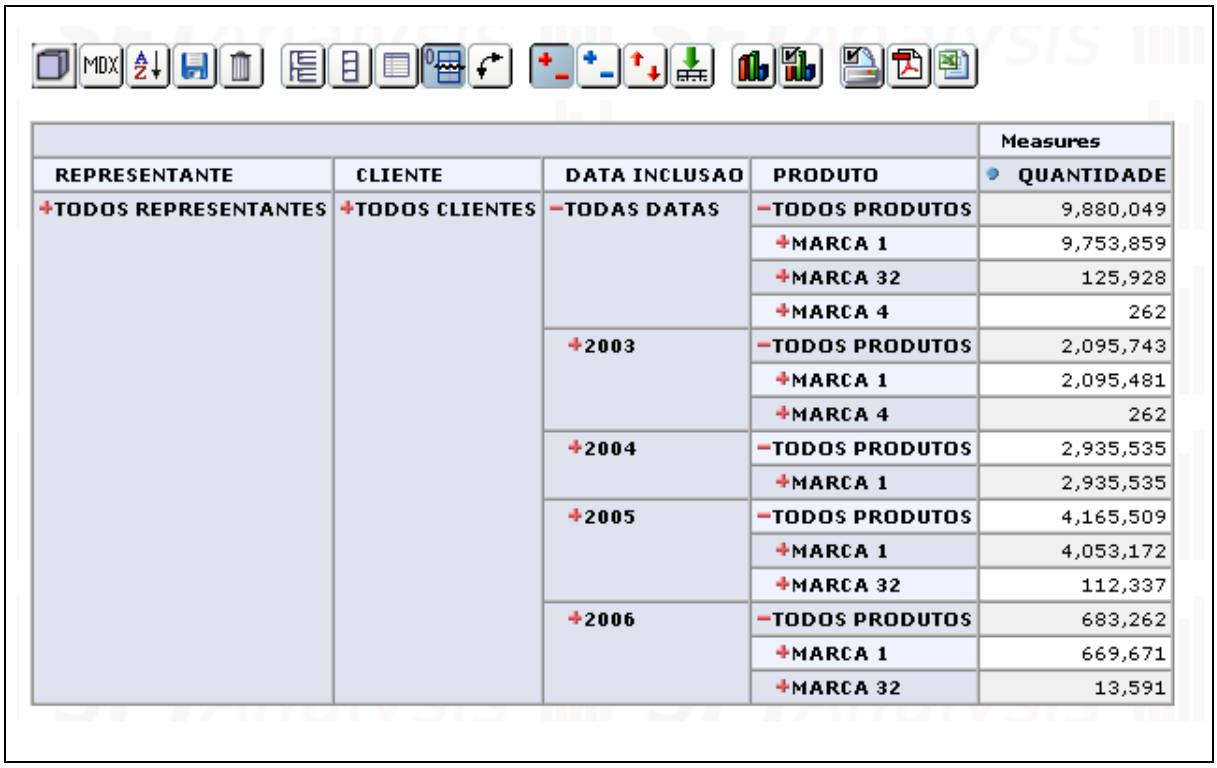
Figura 5.16 *Exibição de linhas com valores nulos.*

10- Inversão de eixos, comumente chamado de pivoteamento. Esta opção permite que as linhas sejam transformadas em colunas e vice-versa. A Figura 5.17 exemplifica esta opção onde se pode observar que a medida quantidade, normalmente exibida em forma de linha, está disposta na forma de coluna.

	REPRESENTANTE			
	+TODOS REPRESENTANTES			
	CLIENTE			
	+TODOS CLIENTES			
	DATA INCLUSAO			
	+TODAS DATAS			
	PRODUTO			
Measures	−TODOS PRODUTOS	+MARCA 1	+MARCA 32	+MARCA 4
QUANTIDADE	9,880,049	9,753,859	125,928	262

Figura 5.17 *Pivoteamento da análise.*

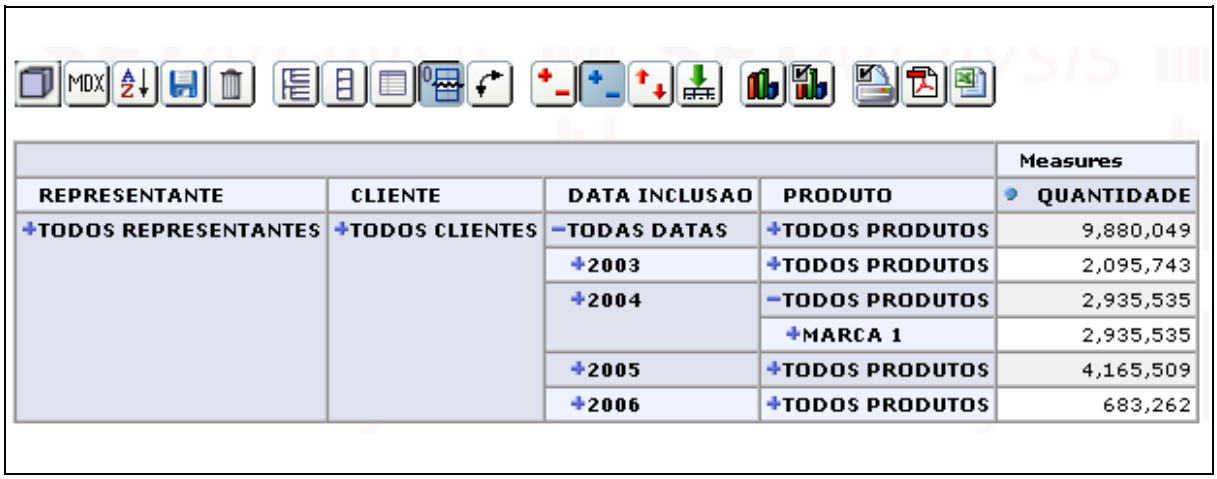
11- *Drill Member*: Abre todas as ocorrências daquele item na árvore. A Figura 5.18 exibe o resultado de uma consulta, onde se observa o *drill member* na coluna PRODUTO. (Grifo nosso).



				Measures
REPRESENTANTE	CLIENTE	DATA INCLUSAO	PRODUTO	QUANTIDADE
+TODOS REPRESENTANTES	+TODOS CLIENTES	−TODAS DATAS	−TODOS PRODUTOS	9,880,049
			+MARCA 1	9,753,859
			+MARCA 32	125,928
			+MARCA 4	262
		+2003	−TODOS PRODUTOS	2,095,743
			+MARCA 1	2,095,481
			+MARCA 4	262
		+2004	−TODOS PRODUTOS	2,935,535
			+MARCA 1	2,935,535
		+2005	−TODOS PRODUTOS	4,165,509
			+MARCA 1	4,053,172
			+MARCA 32	112,337
		+2006	−TODOS PRODUTOS	683,262
			+MARCA 1	669,671
			+MARCA 32	13,591

Figura 5.18 *Drill Member.*

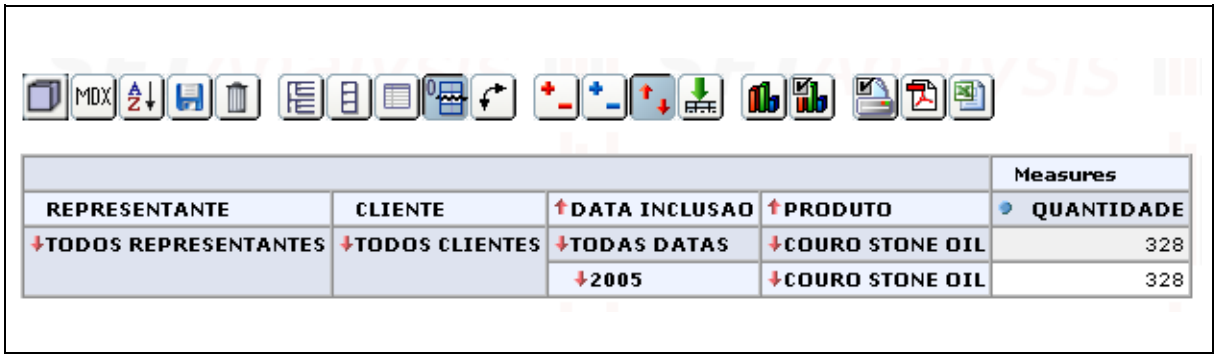
12- *Drill Position*: Abre apenas a ocorrência selecionada daquele item na árvore. A Figura 5.19 mostra o resultado de uma consulta onde se pode notar o *drill position* na coluna PRODUTO, notando-se a diferença com o *drill member* exibido na Figura 5.18. (Grifo nosso).



				Measures
REPRESENTANTE	CLIENTE	DATA INCLUSAO	PRODUTO	QUANTIDADE
+TODOS REPRESENTANTES	+TODOS CLIENTES	−TODAS DATAS	+TODOS PRODUTOS	9,880,049
		+2003	+TODOS PRODUTOS	2,095,743
		+2004	−TODOS PRODUTOS	2,935,535
			+MARCA 1	2,935,535
		+2005	+TODOS PRODUTOS	4,165,509
		+2006	+TODOS PRODUTOS	683,262

Figura 5.19 *Drill Position.*

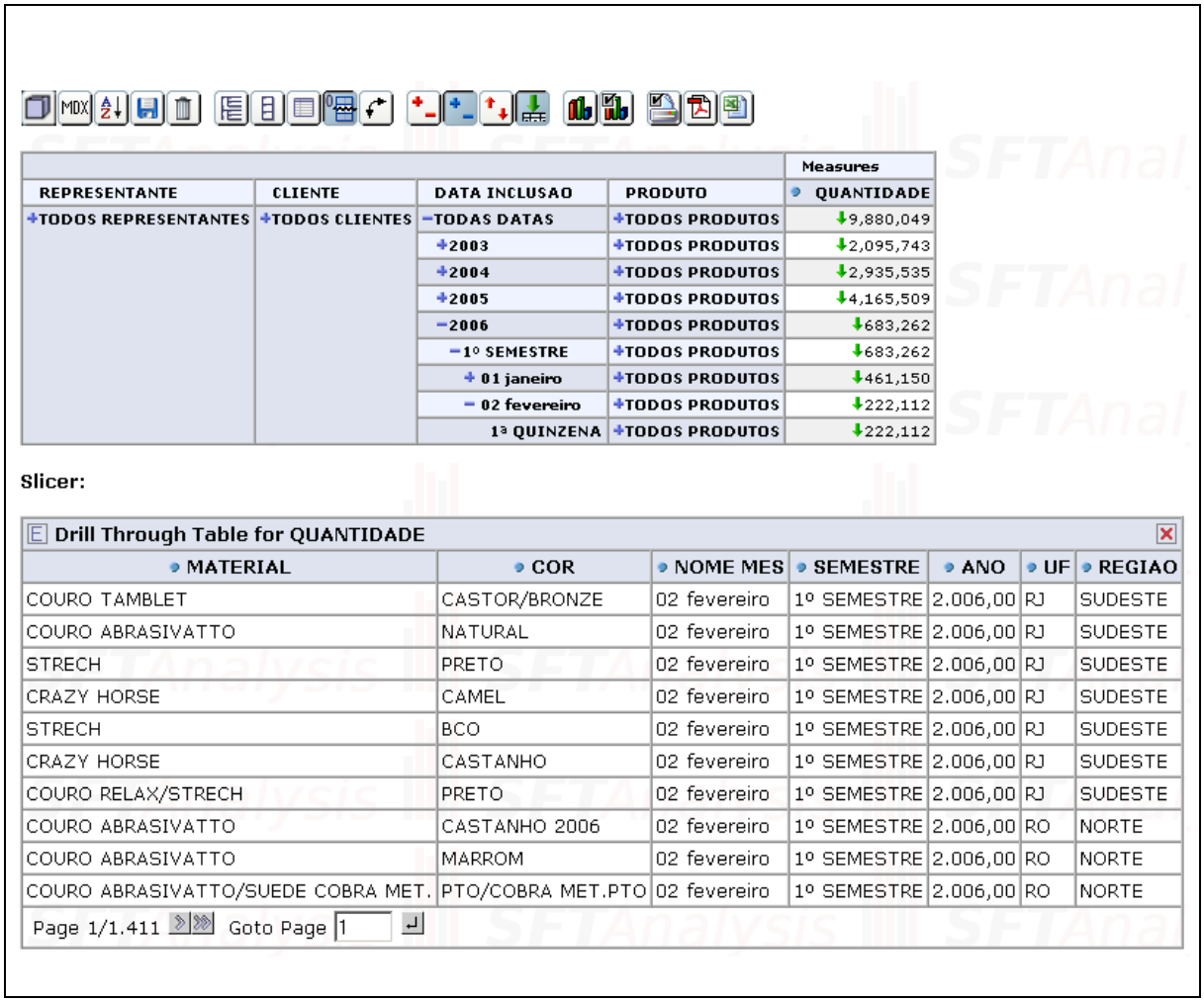
13- *Drill Replace:* Substitui a raiz da dimensão atual pelo item selecionado. A Figura 5.20 exemplifica o resultado de uma consulta utilizando-se esta opção.



				Measures
REPRESENTANTE	CLIENTE	DATA INCLUSAO	PRODUTO	QUANTIDADE
TODOS REPRESENTANTES	TODOS CLIENTES	TODAS DATAS	COURO STONE OIL	328
		2005	COURO STONE OIL	328

Figura 5.20 *Drill Replace.*

14- *Drill Through*: explica de onde foi calculado aquele dado. Nesta opção são exibidos os dados básicos que geraram a informação acumulada. Esta opção exibe os registros da tabela de fato na granularidade armazenada, conforme exibido na Figura 5.21.



				Measures
REPRESENTANTE	CLIENTE	DATA INCLUSAO	PRODUTO	QUANTIDADE
TODOS REPRESENTANTES	TODOS CLIENTES	TODAS DATAS	TODOS PRODUTOS	9,880,049
		2003	TODOS PRODUTOS	2,095,743
		2004	TODOS PRODUTOS	2,935,535
		2005	TODOS PRODUTOS	4,165,509
		2006	TODOS PRODUTOS	683,262
		1º SEMESTRE	TODOS PRODUTOS	683,262
		01 janeiro	TODOS PRODUTOS	461,150
		02 fevereiro	TODOS PRODUTOS	222,112
		1ª QUINZENA	TODOS PRODUTOS	222,112

Slicer:

Drill Through Table for QUANTIDADE

MATERIAL	COR	NOME MES	SEMESTRE	ANO	UF	REGIAO
COURO TAMBLET	CASTOR/BRONZE	02 fevereiro	1º SEMESTRE	2.006,00	RJ	SUDESTE
COURO ABRASIVATTO	NATURAL	02 fevereiro	1º SEMESTRE	2.006,00	RJ	SUDESTE
STRECH	PRETO	02 fevereiro	1º SEMESTRE	2.006,00	RJ	SUDESTE
CRAZY HORSE	CAMEL	02 fevereiro	1º SEMESTRE	2.006,00	RJ	SUDESTE
STRECH	BCO	02 fevereiro	1º SEMESTRE	2.006,00	RJ	SUDESTE
CRAZY HORSE	CASTANHO	02 fevereiro	1º SEMESTRE	2.006,00	RJ	SUDESTE
COURO RELAX/STRECH	PRETO	02 fevereiro	1º SEMESTRE	2.006,00	RJ	SUDESTE
COURO ABRASIVATTO	CASTANHO 2006	02 fevereiro	1º SEMESTRE	2.006,00	RO	NORTE
COURO ABRASIVATTO	MARROM	02 fevereiro	1º SEMESTRE	2.006,00	RO	NORTE
COURO ABRASIVATTO/SUEDE COBRA MET.	PTO/COBRA MET.PTO	02 fevereiro	1º SEMESTRE	2.006,00	RO	NORTE

Page 1/1.411 Goto Page 1

Figura 5.21 *Drill Through.*

15- Mostrar gráfico: exibe ou não o gráfico dos dados que estão sendo exibidos para o usuário;

16- Configuração do gráfico: tipo (linhas, barra vertical, barra horizontal, 3D), altura e largura, título e demais opções conforme exibido na Figura 5.22.

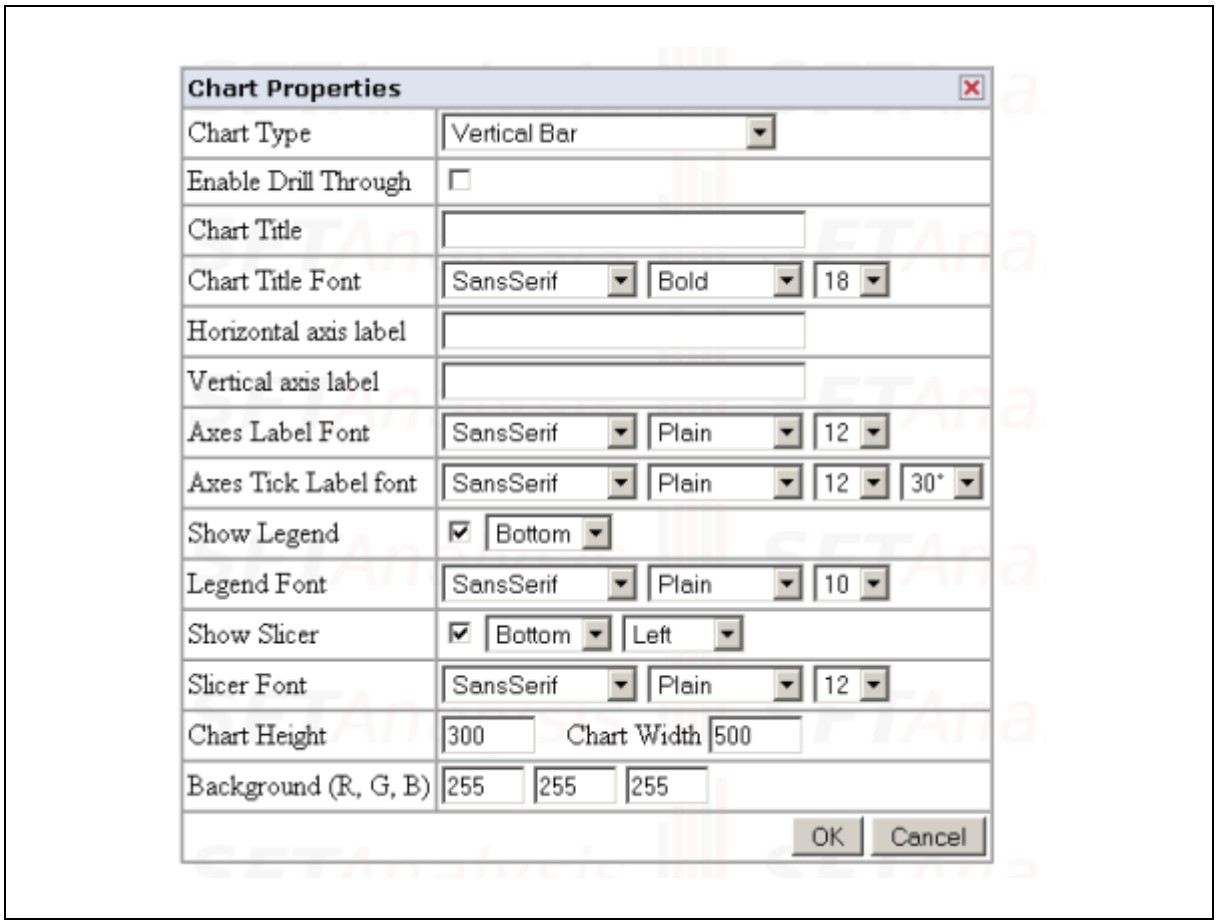


Figura 5.22 *Configuração do gráfico.*

17- Configurar impressão. Permite configurar os parâmetros de impressão, tais como orientação e tamanho do papel, tamanho do gráfico entre outros.

18- Exportar consulta para o formato PDF[8]. Possibilita que o resultado da consulta seja exportado para o formato PDF.

19- Exportar consulta para planilha em formato XLS. Permite que o resultado da consulta seja exportado para planilhas no formato XLS. Nesta opção, os dados são exportados conforme estão dispostos na tela, não permitindo que operações OLAP sejam executadas diretamente na planilha.

[8] O formato PDF (*Portable Document Format*) é uma especificação utilizada para distribuição de documentos eletrônicos (ADOBE, 2006).

5.5 Exemplos de consultas

A seguir, será exibida uma consulta que exemplifica o resultado que pode ser obtido com a utilização do *SFT Analysis*.

O exemplo foi construído supondo um cenário onde o usuário necessite a informação da quantidade vendida por região e por ano relativo a data de entrega, para os pedidos do tipo produção. Para responder a esta solicitação, o primeiro passo é o usuário selecionar o cubo VENDA, restringir o tipo de pedido e após, selecionar as dimensões DATA ENTREGA e CIDADE, conforme exibido na Figura 5.23. (Grifo nosso).

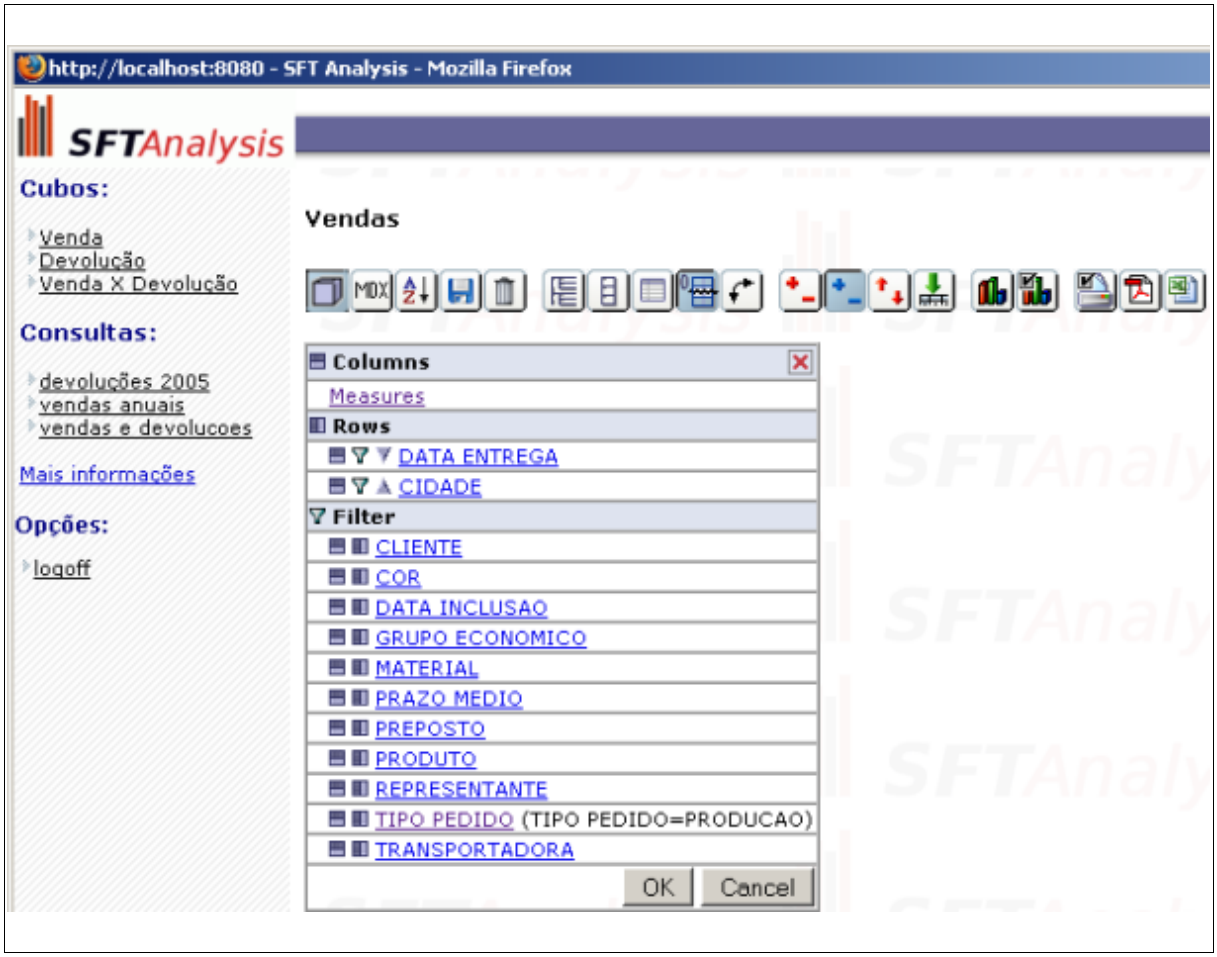


Figura 5.23 *Seleção das colunas da consulta.*

Após o usuário selecionar as colunas desejadas, é mostrado o resultado da consulta onde são exibidos os valores de venda por região para cada ano. Este é o resultado inicial da consulta, sendo que a partir dela o usuário poderá fazer as operações de *drill* na informação desejada. O resultado da consulta é exibido na Figura 5.24.

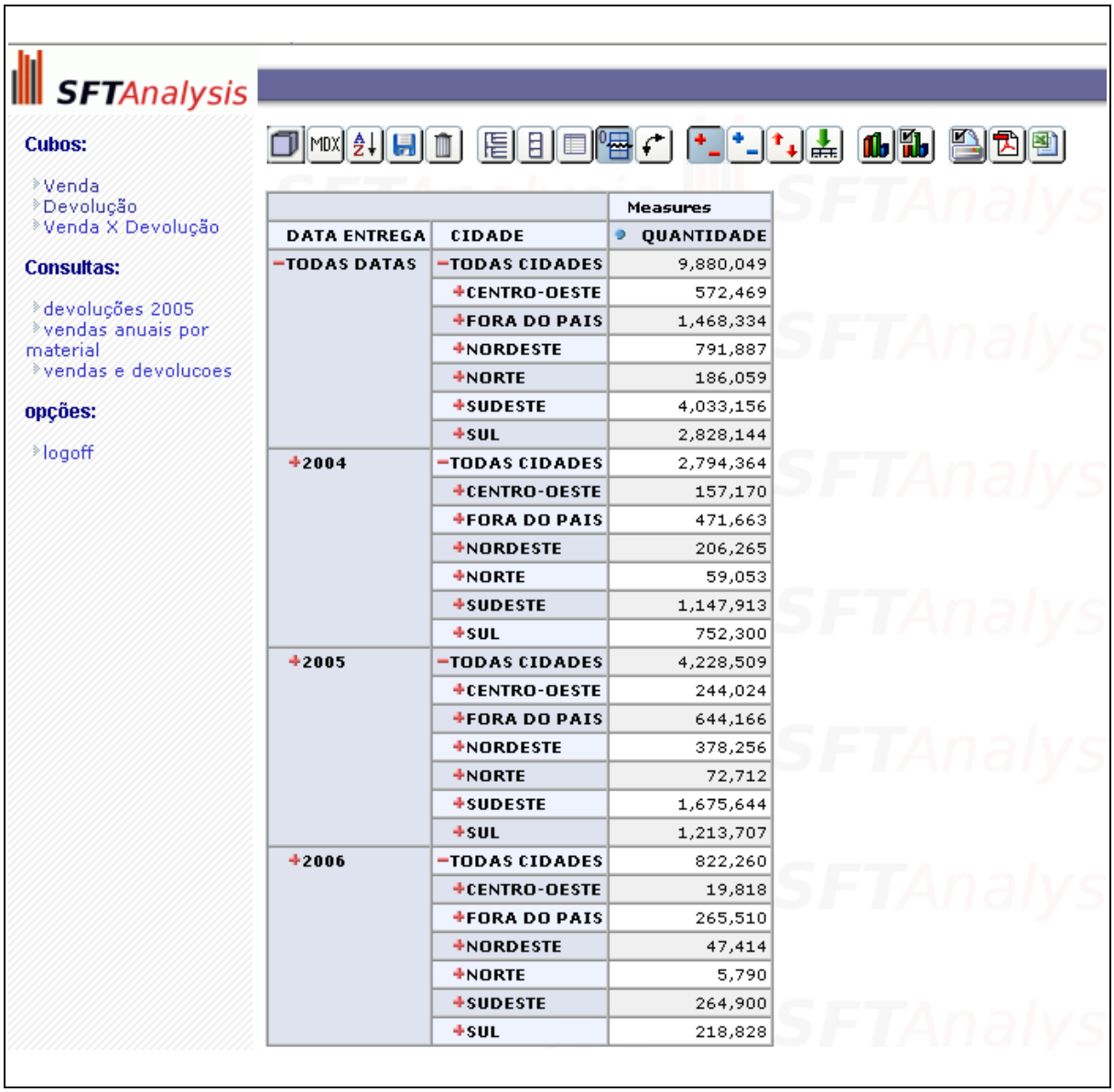


Figura 5.24 *Resultado da consulta.*

Conforme o usuário explora os dados através da navegação no cubo, o *Mondrian* automaticamente traduz a consulta gerada para linguagem MDX. A Figura 5.25 representa a consulta exibida na Figura 5.24 em formato MDX.

```
SELECT {[MEASURES].[QUANTIDADE]} ON COLUMNS,
       NON EMPTY HIERARCHIZE(UNION({({[DATA ENTREGA].[TODAS DATAS],
                                      [CIDADE].[TODAS CIDADES])},
                                  CROSSJOIN([DATA ENTREGA].[TODAS DATAS].CHILDREN,
      {[CIDADE].[TODAS CIDADES]}))) ON ROWS
FROM [VENDA]
WHERE [TIPO PEDIDO].[TODOS TIPOS].[PRODUCAO]
```

Figura 5.25 *Consulta traduzida para a linguagem MDX.*

Esta figura mostra a estrutura de uma consulta MDX, a qual é composta pelas seguintes cláusulas:

Cláusula SELECT: Esta cláusula define os eixos para a estrutura da consulta MDX, identificando os membros das dimensões incluídos em cada eixo.

Cláusula FROM: Define o cubo que está sendo consultado, ela determina qual a origem dos dados multidimensionais será utilizada. Esta cláusula somente pode listar um único cubo. As consultas estão restritas a uma única fonte de dados ou cubo.

Cláusula WHERE (opcional): Esta cláusula é responsável pela restrição do dados retornados pela consulta. A cláusula WHERE determina qual dimensão ou membro é usado como *slicer*.

Além destas cláusulas, neste exemplo são utilizadas também as seguintes funções:

NON EMPTY: Retorna somente os registros com dados válidos, descartando os registros vazios;

HIERARCHIZE: Organiza os membros em uma ordem hierárquica;

UNION: Retorna o resultado da união de dois conjuntos de dados;

CROSSJOIN: Função que retorna o produto do cruzamento de um conjunto especificado;

CHILDREN: Retorna os membros filhos de um determinado membro pai.

5.6 Sugestão para uma estrutura de metadados

Embora este trabalho não implemente uma fonte de metadados para o DM proposto,

será feita uma sugestão para a construção de uma estrutura para armazenamento e utilização dos mesmos.

É sugerida uma estrutura de metadados que seja permitido a sua manipulação pelos usuários finais. Estes metadados seriam descritivos e teriam por finalidade ajudá-los, auxiliando no funcionamento das análises e consultas e na geração de relatórios.

Esta estrutura poderia utilizar-se de tabelas no banco de dados onde está o modelo dimensional ou ainda de arquivos XML, permitindo a manipulação pelo usuário através da própria aplicação. O usuário poderia manipular as informações sobre as tabelas de fatos e dimensões e seus atributos, e estas informações poderiam ser exibidas ao usuário através de *tooltips*[9] ou um *link* específico na aplicação.

Outra informação que os metadados poderiam conter seria o histórico de atualizações do DM, para permitir ao usuário identificar a sua evolução.

5.7 Dificuldades encontradas durante o desenvolvimento

Durante o desenvolvimento da solução foram encontradas algumas dificuldades que precisaram ser superadas para que este fosse concluído a contento. A principal destas dificuldades foi devido à falta de uma boa documentação, sendo que a maioria das informações adquirida durante este estudo, foi originada de fóruns de discussão de desenvolvedores e dos próprios *sites* das ferramentas *Mondrian* e *Jpivot.*

Além disso, foram encontrados alguns pequenos *bugs* nas ferramentas citadas acima. Estes *bugs* não chegaram a comprometer o desenvolvimento, porém foi necessário um tratamento específico para cada problema encontrado. Para alguns destes *bugs*, foram liberadas correções pelos desenvolvedores das ferramentas e outros tiveram que ser contornados via programação, pois não havia solução definitiva para o problema, seguindo instruções destes mesmos desenvolvedores.

Em algumas situações, ocorreu o erro "*java.lang.OutOfMemoryError: Java heap space*", que é gerado quando uma aplicação *Java* necessita alocar mais memória que a disponível para ela. A ocorrência deste erro indica que o servidor onde a aplicação for

[9] *Tooltip* é um elemento comum de interface gráfica com o usuário. Ele é usado em conjunto com um cursor, geralmente um ponteiro de um mouse. O usuário posiciona o cursor sobre um item, sem clicá-lo, e uma pequena caixa aparece com um nome ou descrição deste item (WIKIPEDIA, 2005).

disponibilizada deve possuir uma quantidade suficiente de memória para evitar este problema. (Grifo nosso).

5.8 Melhorias e trabalhos futuros

Baseado no estudo apresentado abre-se a possibilidade de inúmeras melhorias e trabalhos futuros na área. A seguir são apresentadas algumas sugestões:

- Aprimorar a proposta apresentada, adicionando novas funcionalidades à solução, como por exemplo novos cubos de dados, priorizando as informações sobre o retorno financeiro gerado por pedidos e clientes;
- Criar novos DMs para outros setores da indústria calçadista, como por exemplo, para os setores financeiro e programação da produção;
- Criar um portal *Web* para acessar os novos DMs criados;
- Utilizar soluções para simplificar o desenvolvimento de aplicações *Web* J2EE, como, por exemplo, os *frameworks*[10] JSF ou *Struts,* para permitir uma melhor organização do código;
- Criar a documentação completa para a aplicação desenvolvida;
- Criar uma estrutura de metadados para o DM e permitir ao usuário a sua manipulação.

Enfim, existem várias possibilidades para novas pesquisas objetivando o aprimoramento da solução com a adição de novos recursos e funcionalidades, tornando-a mais robusta e versátil para que a mesma satisfaça plenamente as necessidades dos usuários aos quais se destina.

[10] *Framework* é uma estrutura de suporte definida em que um projeto do software pode ser organizado e desenvolvido. Um *framework* pode incluir programas de apoio, bibliotecas de código, linguagens de *script* e outros *softwares* para ajudar a desenvolver e juntar diferentes componentes do projeto (WIKIPEDIA, 2005).

CONCLUSÃO

O DW e demais tecnologias relacionadas mostram-se ferramentas muito interessantes para empresas que possuem grandes volumes de dados gerados e acumulados durante sua existência e necessitam utilizar estes dados de uma forma que eles possam auxiliar os administradores destas empresas a tomarem decisões estratégicas rápidas e com segurança.

Inicialmente, esse trabalho apresentou uma fundamentação teórica sobre as tecnologias relacionadas com a solução proposta, visando criar o embasamento necessário para a implementação desta solução. Foi feita também, uma descrição das características das ferramentas de distribuição gratuita e/ou *Open Source* visando escolher àquelas que atendiam as necessidades do projeto.

As ferramentas escolhidas foram utilizadas para o desenvolvimento da aplicação proposta, que teve por objetivo construir um SAD composto por um DM e uma ferramenta OLAP que trabalhasse com informações para a área comercial de uma indústria calçadista. Os dados para o DM foram originados do sistema de gestão empresarial SFT, desenvolvido pela empresa Safetech Informática LTDA.

Com o desenvolvimento da aplicação, pôde-se comprovar a viabilidade da utilização de ferramentas gratuitas e/ou O*pen Source* para criar SADs, pois foi possível criar um DM e uma ferramenta OLAP utilizando-se exclusivamente estas ferramentas, sendo que estas apresentaram, além de uma boa facilidade de utilização, uma diversidade de recursos que permitem criar excelentes opções de análises de informações ao usuário. Um dos principais pontos a salientar no uso destas ferramentas é a possibilidade de adição de novos recursos, ampliando a gama de funcionalidades oferecidas pelas mesmas, tornando a sua utilização mais atraente para os desenvolvedores de soluções de BI.

Observou-se ainda, que um dos pontos a serem melhorados nestas ferramentas é a

criação de uma documentação mais abrangente para permitir que os desenvolvedores possam extrair o máximo de recursos das mesmas. Como estas ferramentas estão em constante evolução, novas versões poderão agregar ainda mais recursos tornando-as cada vez mais interessantes para este tipo de solução.

Abre-se também a oportunidade de trabalhos futuros na área, como a melhoria da ferramenta OLAP proposta com a adição de novos recursos e a criação de novos DMs abrangendo outros setores da indústria calçadista.

REFERÊNCIAS BIBLIOGRÁFICAS

ADOBE. ***Adobe Acrobat***. Disponível em: <http://www.adobe.com/products/acrobat/>. Acesso em: 06 mar. 2006.

AMARAL, Glenda Carla Moura. **AQUAWARE: Um ambiente de suporte à qualidade de dados em *Data Warehouse***. Rio de Janeiro, 2003. Dissertação (Mestrado em Informática) - Universidade Federal do Rio de Janeiro - UFRJ. Disponível em <http://genesis.nce.ufrj.br/dataware/DataWarehouse/Teses/Glenda/Tese_Glenda.pdf>. Acesso em 02 out. 2005.

ANZANELLO, Cynthia Aurora. **OLAP conceitos e utilização.** Instituto de Informática, UFRGS. Disponível em: <http://www.inf.ufrgs.br/~clesio/cmp151/cmp15120021/ artigo_cynthia.pdf>. Acesso em 02 out. 2005.

BARBALHO, Patrícia. Descubra o *Data Warehouse*: produtividade e rapidez. **Revista SQL magazine**. Edição 3, 2003.

BARBIERI, Carlos. ***Business Intelligence*: modelagem e tecnologia**. Rio de Janeiro: Axcel Books, 2001.

BISPO, Carlos Alberto Ferreira. **Uma análise da nova geração de sistemas de apoio à decisão**. São Carlos, 1998. 160 p. Dissertação (Mestrado) - Escola de Engenharia de São Carlos, Universidade de São Paulo. Disponível em: <http://www.teses.usp.br /teses/disponiveis/18/18140/tde-04042004-152849/publico/dissertacao_carlos.pdf>. Acesso em: 06 ago. 2005.

BRAGA, André Luiz. **Um arcabouço para a implementação de *Data Warehouses* estendidos com tipos abstratos de dados**. Rio de Janeiro, 2004, 280 p. (COPPE/UFRJ, D. Sc., Engenharia de Sistemas e Computação, 1998) Tese - Universidade Federal do Rio de Janeiro. Disponível em: <http://ge.cos.ufrj.br/twiki/pub/Main/TesesDeDoutoradoOrientadas/ DataWarehouseparaTADs-10102004-Versaofinalregistro.pdf>. Acesso em: 20 set. 2005.

BRITO, Maiquel de. **Proposta de um *Data Warehouse* de informações acadêmicas.** Novo Hamburgo: 2004. 111 p. Projeto de Diplomação (Bacharelado em Ciência da Computação) – Instituto de Ciências Exatas e Tecnológicas, Centro Universitário Feevale, Novo Hamburgo.

CAMPOS, Maria Luiza; FILHO, Arnaldo V. Rocha. ***Data Warehouse***. Disponível em <http://genesis.nce.ufrj.br/dataware/tutorial/home.html>. Acesso em: 20 de set. 2005.

CIELO, Ivã. **Arquiteturas OLAP**. Disponível em <http://www.datawarehouse.inf.br/artigos/

olap.asp>. Acesso em: 20 de set. 2005.

COSTA, André Fernandes; ANCIÃES, Felipe Curvello. **Aspectos de criação e carga de um ambiente de *Data Warehouse*.** Rio de Janeiro: 2001. 111 p. Projeto de Diplomação (Bacharelado em Informática) – Departamento de Ciência da Computação, Universidade Federal do Rio de Janeiro. Disponível em: <http://dataware.nce.ufrj.br:8080/dataware/publicacoes/dataware/fisico/projetosFinais/datawarehousing/COSTA-ANCIAES-2001.pdf>. Acesso em: 20 de set. 2005.

COUROMODA. ***Couromoda.com***. Disponível em: <http://www.couromoda.com.br>. Acesso em: 16 fev. 2006.

CUNHA, José Antonio da. **Arquiteturas de ferramentas OLAP**. 2004. Disponível em: <http://www.engcomp.ufrn.br/~max/cefet/banco2/Arquitetura_OLAP.ppt>. Acesso em: 20 set. 2005.

ECLIPSE. ***Eclipse Foundation***. Disponível em: <http://www.eclipse.org>. Acesso em: 06 ago. 2005.

FRANCAL. ***Francal Feiras***. Disponível em: <http://www.francal.com.br>. Acesso em: 16 fev. 2006.

FORTULAN, Marcos Roberto; FILHO, Eduardo Vila Gonçalves. **Uma proposta de aplicação de *Business Intelligence* no chão-de-fábrica.** São Carlos, SP: 2005. Disponível em: <http://www.scielo.br/pdf/gp/v12n1/a06v12n1.pdf>. Acesso em: 03 out. 2005.

GARTNER. ***Gartner Group***. Disponível em: <http://www.gartner.com>. Acesso em: 26 ago. 2005.

GRIMES, Seth. *Open-Source Releases Invade the Reporting Market*. ***Intelligent Enterprise***. Estados Unidos, 2005. Disponível em: <http://www.iemagazine.com/showArticle.jhtml?articleID=163100786>. Acesso em: 06 ago. 2005.

HOKAMA, Daniele Del Bianco et al. **A modelagem de dados no ambiente *Data Warehouse*.** São Paulo: 2004. 121 p. Projeto de Diplomação (Bacharelado em Sistemas de Informação) – Faculdade de Computação e Informática, Universidade Presbiteriana Mackenzie. Disponível em: <http://meusite.mackenzie.com.br/rogerio/tgi/2004ModelagemDW.pdf>. Acesso em: 20 set. 2005.

INMON, W. H. **Como construir o *Data Warehouse*.** 2. ed. Rio de Janeiro: Campus, 1997.

INMON, W. H. et al. **Gerenciando *Data Warehouse*.** São Paulo: Makron Books, 1999.

JASPERREPORTS. ***Free Java reporting library***. Disponível em: <http://jasperreports.sourceforge.net>. Acesso em: 03 out. 2005.

JAVA. ***Java Technology***. Disponível em: < http://java.sun.com>. Acesso em: 06 ago. 2005.

JCP. ***Java Community Process***. Disponível em: <http://www.jcp.org>. Acesso em: 06 ago. 2005.

JFREECHART. ***A free Java chart library***. Disponível em: <http://www.jfree.org/jfreechart>. Acesso em: 03 oct. 2005.

JSF. ***JavaServer Faces***. Disponível em: <https://javaserverfaces.dev.java.net.>. Acesso em: 20 set. 2005.

JPIVOT. ***A JSP based OLAP***. Disponível em: <http://jpivot.sourceforge.net>. Acesso em: 06 ago. 2005.

KIMBALL, Ralph; MERZ, Richard. ***Data Webhouse*: Construindo o *Data Warehouse* para a *Web***. Tradução: Edson Furmankiewicz, Joana Figueiredo. 1. ed. Rio de Janeiro: Campus, 2000.

KIMBALL, Ralph; ROSS, Margy. ***The Data Warehouse Toolkit*: guia completo para modelagem dimensional**. 2. ed. Rio de Janeiro: Campus, 2002.

LOBO, Adilton. **Modelagem de um estudo de caso utilizando a ferramenta entidade/relacionamento, baseada no modelo de Peter Chen**. Joinville, SC: 1998. Disponível em: <http://pages.udesc.br/~r4al/artentre.htm>. Acesso em: 14 nov. 2005.

MACHADO, Felipe N. R. **Projeto de *Data Warehouse*: uma visão multidimensional**. São Paulo: Érica, 2000.

MAZETTO, Carlos Alberto et al. **SAD BI.** Americana: 2004. 82 p. Projeto de Diplomação (Bacharelado em Análise de Sistemas) – Centro Universitário Salesiano de São Paulo, Campus Dom Bosco – Americana, SP. Disponível em: <http://geocities.yahoo.com.br/projetosadg7/Monografia.pdf>. Acesso em: 20 set. 2005.

MICROSOFT. ***Microsoft Corporatio*n**. Disponível em: <http://www.microsoft.com>. Acesso em: 06 ago. 2005.

MONDRIAN. ***Mondrian* OLAP *Server***. Disponível em: <http://mondrian.sourceforge.net>. Acesso em: 06 ago. 2005.

MYSQL. ***Mysql AB***. Disponível em: <http://www.mysql.com>. Acesso em: 03 mar. 2006.

OPENI. ***Open Source Web Application for OLAP Reporting***. Disponível em: <http://www.openi.org>. Acesso em: 06 ago. 2005.

ORACLE. ***Oracle Corporation***. Disponível em: < http://www.oracle.com>. Acesso em: 06 mar. 2006.

PENTAHO. ***Open Source Business Intelligen*ce**. Disponível em: <http://www.pentaho.org>. Acesso em: 06 ago. 2005.

PEREIRA, Denise Maciel. **Uso do padrão OIM de metadados no suporte às transformações de dados em ambiente de *Data Warehouse***. Rio de Janeiro, 2000. Dissertação (Mestrado) – Universidade Federal do Rio de Janeiro. Disponível em: < http://dataware.nce.ufrj.br:8080/dataware/publicacoes/dataware/fisico/teses/metadados/PEREIRA-2000.pdf>. Acesso em: 01 out. 2005.

PERNAS, Ana Marilza da Rosa. **Modelagem de um *Data Webhouse* voltado a produção e comercialização de sementes.** Pelotas: 2003. 70 p. Projeto de Diplomação (Bacharelado em Ciência da Computação) – Instituto de Física e Matemática, Universidade Federal de Pelotas, Pelotas. Disponível em: <http://www.ufpel.edu.br/prg/sisbi/bibct/acervo/info/2003/mono_ana_pernas.pdf>. Acesso em: 20 set. 2005.

POSTGRESQL. ***Postgresql***. Disponível em: <http://www.postgresql.org>. Acesso em: 03 mar. 2006.

PRODANOV, Cleber C. **Manual de metodologia científica**. 3. ed. Novo Hamburgo: Feevale, 2003.

PUC-RIO. ***Data Mining* - conceitos, técnicas, ferramentas e aplicações**. Rio de Janeiro, 2005. Disponível em: <http://www.cce.puc-rio.br/informatica/dataminingcentro.htm>. Acesso em: 03 out. 2005.

ROSA, Katia Liane da. **Análise de ferramentas OLAP com acesso à banco de dados.** Novo Hamburgo: 2005. 87 p. Projeto de Diplomação (Bacharelado em Ciência da Computação) – Instituto de Ciências Exatas e Tecnológicas, Centro Universitário Feevale, Novo Hamburgo.

SIMON, Inre. **Um Estudo de Caso: A Produção e Disseminação da Literatura Acadêmica.** Disponível em: <http://www.ime.usp.br/~is/ddt/mac339-01/aulas/www.linux.ime.usp.br/hvila/mac339/tema8.html. Acesso em: 01 abr. 2005.

SING, Harry. ***Data Warehouse***. São Paulo: Makron Books, 2001.

SPRAGUE, R. H.; WATSON, H. J. **Sistemas de apoio à decisão: colocando a teoria em prática**. Rio de Janeiro: Campus, 1991.

STRUTS. ***Apache Struts Project***. Disponível em: <http://struts.apache.org>. Acesso em: 20 set. 2005.

SUN. ***Sun Microsystems***. Disponível em: <http://www.sun.com>. Acesso em: 20 mar. 2006.

THE R PROJECT. ***The R Project for Statistical Computing***. Disponível em: <http://www.r-project.org>. Acesso em: 03 out. 2005.

THOMSEN, Erik. **OLAP: construindo sistemas de informações multidimensionais**. 2. ed. Rio de Janeiro: Campus, 2002.

TOMCAT. ***Apache Tomcat***. Disponível em: <http://tomcat.apache.org>. Acesso em: 06 ago. 2005.

UNIBANCO. **Unibanco**. Disponível em: <http://www.unibanco.com.br>. Acesso em: 06 abr. 2006.

WIKIPEDIA. ***The free encyclopedia***. Disponível em <http://pt.wikipedia.org> Acesso em: 03 out. de 2005.

XML. ***Extensible markup language***. Disponível em: <http://www.w3.org/XML>. Acesso em: 02 out. 2005.

XMLA.ORG. ***XML for Analysis Specification***. Disponível em: <http://www.xmla.org/docs_pub.asp>. Acesso em: 02 out. 2005.

YIN, R. **Estudo de caso: Planejamento e Métodos**. 2ª ed. Porto Alegre: Bookman, 2001.

ANEXO 1 – S*cript* SQL para fazer a carga do D*ata* M*art*

```
declare
  vdata_inicial   date := to_date('01-jan-2003','dd-mon-yyyy');
BEGIN
  -- faz o truncate (eliminação dos dados) das tabelas
  truncate table condicao_pagamento;
  truncate table motivo_devolucao;
  truncate table produto;
  truncate table representante;
  truncate table transportadora;
  truncate table cidade;
  truncate table cliente;
  truncate table tipo_pedido;
  truncate table preposto;
  truncate table data;
  truncate table data_temp;
  truncate table venda;
  truncate table devolucao;
  --
  -- faz a carga da dimensão data
  for vreg in 0..(360 *5)
  loop
    insert into data_temp values(vdata_inicial + vreg);
  end loop;
  --
  insert into data
    (data,
     dia_semana,
     nome_dia_semana,
     dia_mes,
     semana_mes,
     quinzena,
     mes,
     nome_mes,
     trimestre,
     semestre,
     ano,
     semestre_label,
     quinzena_label)
  select  distinct data data,
          to_number(to_char(data , 'd', 'nls_date_language=portuguese')) dia_semana,
          to_char(data , 'day', 'nls_date_language=portuguese') nome_dia_semana,
          to_number(to_char(data , 'dd', 'nls_date_language=portuguese')) dia_mes,
          to_number(to_char(data , 'w', 'nls_date_language=portuguese')) semana_mes,
          to_number(decode(to_char(data , 'w','nls_date_language=portuguese'),1,1,2,1,2)) qui,
          to_number(to_char(data , 'mm', 'nls_date_language=portuguese')) mes,
          to_char(data , 'month', 'nls_date_language=portuguese') nome_mes,
          to_number(to_char(data , 'q', 'nls_date_language=portuguese')) trimestre,
          to_number(decode(to_char(data , 'q','nls_date_language=portuguese'),1,1,2,1,2)) sem,
          to_number(to_char(data , 'yyyy', 'nls_date_language=portuguese')) ano,
          null,
          null
  from   data_temp;
  --
  update data
  set quinzena_label = '2'||chr(170)||' quinzena'
  where dia_mes > 15 ;
  --
  update data
  set quinzena_label = '1'||chr(170)||' quinzena'
```

```
where dia_mes <= 15;
--
update data
set semestre_label = '1'||chr(186)||' semestre'
where dia_mes <=6;
--
update data
set semestre_label = '2'||chr(186)||' semestre'
where dia_mes >6;
--
-- carga da dimensão cidade
insert into cidade
  (codigo_cidade,
   nome,
   uf,
   regiao,
   capital_interior,
   regiao_metropolitana)
select c.codigo_cidade,
       replace(c.nome_cidade,'''',' '),
       c.uf,
       r.descricao,
       c.capital_interior,
       c.regiao_metropolitana
from   cidade c,
       estado uf,
       regiao r
where  uf.uf = c.uf
and    r.codigo_regiao = uf.codigo_regiao;
--
-- carga da dimensão cliente
insert into cliente
  (codigo_cliente,
   nome,
   grupo_economico,
   conceito)
select c.codigo_pessoa,
       p.nome,
       ge.descricao,
       pck_cliente.fnc_conceito_financeiro(p.codigo_pessoa,
                                           vdata_inicial,
                                           trunc(sysdate)) conceito
from   pessoa   p,
       generico.cliente  c,
       grupo_economico ge
where  c.codigo_pessoa = p.codigo_pessoa
and    ge.codigo_grupo_economico(+) = c.codigo_grupo_economico;
--
-- carga da dimensao condicao_pagamento
insert into condicao_pagamento
  (codigo_condicao_pagamento,
   descricao,
   prazo_medio,
   percentual_desconto)
select codigo_condicao_pagamento,
       descricao,
       prazo_medio,
       percentual_desconto_nf
from   pcp.condicao_pagamento p;
--
-- carga da dimensao motivo_devolucao
insert into motivo_devolucao
  (codigo_motivo,
   motivo_devolucao)
select p.codigo_motivo,
       p.descricao
from   motivo_devolucao p;
--
-- carga da dimensao preposto
insert into preposto
  (codigo_representante,
   nome_representante,
   codigo_preposto,
   nome)
select p.codigo_pessoa_representante,
       p.codigo_marca,
       p.codigo_preposto,
```

```
       p.nome
from   preposto p,
       representante marca rm,
       marca m,
       pessoa pe
where  p.codigo_pessoa_representante = rm.codigo_pessoa
and    p.codigo_marca = rm.codigo_marca
and    m.codigo_marca = rm.codigo_marca
and    pe.codigo_pessoa = rm.codigo_pessoa;
--
-- carga da dimensao produto
insert into produto
  (codigo_produto,
   referencia,
   codigo_linha,
   descricao_linha,
   codigo_marca,
   descricao_marca,
   tipo_produto,
   material,
   cor,
   familia,
   colecao)
select vr.codigo_item,
      vr.referencia,
      vw.codigo_linha,
      vw.descricao_linha,
      vw.codigo_marca,
      vw.descricao_marca,
      vw.descricao_tipo_produto,
      pck_coordenacao_componente.fnc_descr_material_etiqueta(vr.referencia,
                                                             vr.sequencia_referencia),
      c.descricao ,
      f.descricao ,
      col.descricao
from   versao_referencia vr,
      referencia r,
      vw_referencia vw,
      cor c,
      familia f,
      colecao col
where  vw.referencia = vr.referencia
and    c.codigo_cor = vr.codigo_cor
and    r.referencia = vr.referencia
and    f.codigo_familia = r.codigo_familia
and    col.codigo_colecao = r.codigo_colecao;
--
-- carga da dimensao representante
insert into representante
  (codigo_representante,
   nome)
select p.codigo_pessoa,
      p.nome
from   representante_marca rm,
      pessoa p,
      marca m
where  p.codigo_pessoa = rm.codigo_pessoa
annd   m.codigo_marca = rm.codigo_marca;
--
-- carga da dimensao transportadora
insert into transportadora
  (codigo_transportadora,
   nome)
select p.codigo_pessoa,
       p.nome
from   pessoa p,
      generico.transportadora t
where  t.codigo_pessoa = p.codigo_pessoa;
--
-- carga da dimensao tipo_pedido
insert into tipo_pedido
  (tipo_pedido,
   descricao_tipo_pedido)
select tipo_pedido,
       descricao
from   tipo_pedido;
--
```

```
-- carga do fato devolucao
insert into devolucao
  (codigo_produto,
   codigo_cliente,
   codigo_representante,
   codigo_cidade,
   codigo_motivo,
   data_devolucao,
   quantidade,
   valor)
select indv.codigo_item,
    dv.codigo_emitente,
    dv.codigo_pessoa_representante,
    ci.codigo_cidade,
    mtdv.codigo_motivo,
    nfse.data_movimento,
    sum(infe.quantidade),
    round(sum((infe.quantidade * infe.preco_unitario) +
              nvl(infe.valor_acessorio,0) +
              nvl(infe.valor_frete,0) + nvl(infe.valor_seguro,0) +
              nvl(infe.valor_ipi,0) + nvl(infe.valor_desconto,0) -
              nvl(indv.valor_desconto_lvr_comercio,0) -
              nvl(indv.valor_desconto_vendor,0) -
              nvl(indv.valor_desconto_suframa,0) ),2)
from     nota_fiscal_entrada nfse
,        item_nf_entrada infe
,        item_nf_devolucao_venda  indv
,        motivo_devolucao mtdv
,        devolucao_venda dv
,        pessoa p1
,        cidade ci
where    nfse.data_movimento      >= vdata_inicial
and      infe.numero_nota_fiscal  = nfse.numero_nota_fiscal
and      infe.serie               = nfse.serie
and      infe.codigo_emitente     = nfse.codigo_emitente
and      indv.numero_nota_fiscal  = infe.numero_nota_fiscal
and      indv.serie               = infe.serie
and      indv.codigo_emitente     = infe.codigo_emitente
and      indv.codigo_item         = infe.codigo_item
and      indv.diferenciador_item  = infe.diferenciador_item
and      mtdv.codigo_motivo       = indv.codigo_motivo_devolucao
and      dv.numero_nota_fiscal    = infe.numero_nota_fiscal
and      dv.serie                 = infe.serie
and      dv.codigo_emitente       = infe.codigo_emitente
and      p1.codigo_pessoa         = nfse.codigo_emitente
and      ci.codigo_cidade         = p1.codigo_cidade
group by indv.codigo_item
,     dv.codigo_pessoa_representante
,     dv.codigo_emitente
,     ci.codigo_cidade
,     mtdv.codigo_motivo
,     nfse.data_movimento;
--
--  carga do fato venda
insert into venda
  (data_inclusao,
   data_entrega,
   codigo_cliente,
   codigo_representante,
   codigo_transportadora,
   codigo_produto,
   tipo_pedido,
   codigo_cidade,
   codigo_condicao_pagamento,
   codigo_preposto,
   vendor,
   quantidade,
   valor_liquido,
   valor_bruto,
   percentual_desconto,
   percentual_comissao)
select /*+ ordered
           use_nl(pr,tp,gp,r,l,ma,vr,m,c,p,p1)
           index (pr pedido_data_inclusao_i) */
       pr.data_inclusao,
       pr.data_entrega_cliente,
       pr.codigo_pessoa_cliente,
```

```
          pr.codigo_pessoa_representante,
          pr.codigo_pessoa_transportadora,
          vr.codigo_item,
          pr.tipo_pedido,
          ci.codigo_cidade,
          pr.codigo_condicao_pagamento,
          pr.codigo_preposto,
          pr.vendor,
          pck_pedido.fnc_qtd_grade_pedido(gp.numero_pedido,
                                          gp.sequencia_grade) quantidade,
          pck_pedido.fnc_valor_grade_pedido_cotacao(gp.numero_pedido,
                                                    gp.sequencia_grade) valor_liquido,
          pck_pedido.fnc_valor_grade_pedido_bruto(gp.numero_pedido,
                                                  gp.sequencia_grade) valor_bruto,
          pr.percentual_desconto,
          pr.percentual_comissao
  from   pedido            pr,
          tipo_pedido       tp,
          grade_pedido      gp,
          versao_referencia vr,
          pessoa            p,
          cidade            ci
  where  pr.data_inclusao             >= vdata_inicial
  and    gp.numero_pedido              = pr.numero_pedido
  and    tp.tipo_pedido                = pr.tipo_pedido
  and    tp.participa_relatorio_vendas = 'S'
  and    pr.estado_pedido             != 'C'
  and    vr.referencia                 = gp.referencia
  and    vr.sequencia_referencia       = gp.sequencia_referencia
  and    p.codigo_pessoa               = pr.codigo_pessoa_cliente
  and    ci.codigo_cidade              = p.codigo_cidade;
  --
END;
```

ANEXO 2 - XML de definição do *schema*

```
<?xml version="1.0" encoding="UTF-8"?>
<Schema name="VENDAS">
  <Dimension name="CLIENTE">
    <Hierarchy hasAll="true" allMemberName="TODOS CLIENTES" primaryKey="CODIGO_CLIENTE"
primaryKeyTable="CLIENTE">
      <Table name="CLIENTE"/>
      <Level column="NOME" levelType="Regular" name="NOME CLIENTE" type="String"
uniqueMembers="true">
        <Property name="CONCEITO" column="CONCEITO"/>
      </Level>
    </Hierarchy>
  </Dimension>

   <Dimension name="GRUPO ECONOMICO">
    <Hierarchy hasAll="true" allMemberName="TODOS GRUPOS" primaryKey="CODIGO_CLIENTE"
primaryKeyTable="CLIENTE">
      <Table name="CLIENTE"/>
      <Level column="GRUPO_ECONOMICO" levelType="Regular" name="GRUPO ECONOMICO" type="String"
uniqueMembers="true"/>
      <Level column="NOME" levelType="Regular" name="NOME CLIENTE" type="String"
uniqueMembers="true"/>
    </Hierarchy>
  </Dimension>


  <Dimension  name="REPRESENTANTE">
    <Hierarchy hasAll="true" allMemberName="TODOS REPRESENTANTES"
primaryKey="CODIGO_REPRESENTANTE" primaryKeyTable="REPRESENTANTE">
      <Table name="REPRESENTANTE"/>
      <Level column="NOME" levelType="Regular" name="NOME REPRESENTANTE"
uniqueMembers="true">
      </Level>
    </Hierarchy>
  </Dimension>

  <Dimension  name="PREPOSTO">
    <Hierarchy hasAll="true" allMemberName="TODOS PREPOSTOS" primaryKey="CODIGO_PREPOSTO"
primaryKeyTable="PREPOSTO">
      <Table name="PREPOSTO"/>
      <Level column="NOME_REPRESENTANTE" levelType="Regular" name="NOME REPRESENTANTE"
uniqueMembers="true"/>
      <Level column="NOME" levelType="Regular" name="NOME PREPOSTO"  uniqueMembers="true"/>
    </Hierarchy>
  </Dimension>


  <Dimension name="TRANSPORTADORA">
    <Hierarchy hasAll="true" allMemberName="TODAS TRANSPORTADORAS"
primaryKey="CODIGO_TRANSPORTADORA" primaryKeyTable="TRANSPORTADORA">
      <Table name="TRANSPORTADORA"/>
      <Level column="NOME" levelType="Regular" name="NOME TRANSPORTADORA"
uniqueMembers="true"/>
    </Hierarchy>
  </Dimension>

  <Dimension  name="TIPO PEDIDO">
    <Hierarchy hasAll="true" allMemberName="TODOS TIPOS" primaryKey="TIPO_PEDIDO"
primaryKeyTable="TIPO_PEDIDO">
      <Table name="TIPO_PEDIDO"/>
```

```
      <Level column="DESCRICAO_TIPO_PEDIDO" levelType="Regular" name="TIPO PEDIDO"
uniqueMembers="true"/>
    </Hierarchy>
  </Dimension>

  <Dimension  name="MOTIVO DEVOLUCAO">
    <Hierarchy hasAll="true" allMemberName="TODOS MOTIVOS" primaryKey="CODIGO_MOTIVO"
primaryKeyTable="MOTIVO_DEVOLUCAO">
      <Table name="MOTIVO_DEVOLUCAO"/>
      <Level column="MOTIVO_DEVOLUCAO" levelType="Regular" name="MOTIVO"
uniqueMembers="true"/>
    </Hierarchy>
  </Dimension>

  <Dimension name="DATA ENTREGA" type="TimeDimension">
    <Hierarchy hasAll="true" allMemberName="TODAS DATAS"  primaryKey="DATA"
primaryKeyTable="VW_DATA">
      <Table name="VW_DATA"/>
      <Level column="ANO" levelType="TimeYears" name="ANO" type="Numeric"/>
      <Level column="SEMESTRE_LABEL" levelType="TimeDays" name="SEMESTRE" type="String"/>
      <Level column="NOME_MES" levelType="TimeMonths" name="NOME MES" type="String"/>
      <Level column="QUINZENA_LABEL" levelType="TimeWeeks"  name="QUINZENA" type="String"/>
    </Hierarchy>
  </Dimension>

  <Dimension name="DATA INCLUSAO" type="TimeDimension">
    <Hierarchy hasAll="true" allMemberName="TODAS DATAS"  primaryKey="DATA"
primaryKeyTable="DATA">
      <Table name="DATA"/>
      <Level column="ANO" levelType="TimeYears" name="ANO" type="Numeric"/>
      <Level column="SEMESTRE_LABEL" levelType="TimeDays" name="SEMESTRE" type="String"/>
      <Level column="NOME_MES" levelType="TimeMonths" name="NOME MES" type="String"/>
      <Level column="QUINZENA_LABEL" levelType="TimeWeeks"  name="QUINZENA" type="String"/>
    </Hierarchy>
  </Dimension>

  <Dimension name="DATA DEVOLUCAO">
    <Hierarchy hasAll="true" allMemberName="TODAS DATAS"  primaryKey="DATA"
primaryKeyTable="DATA">
      <Table name="DATA"/>
      <Level column="ANO" name="ANO" type="Numeric"/>
      <Level column="SEMESTRE_LABEL" name="SEMESTRE" type="String"/>
      <Level column="NOME_MES" name="NOME MES" type="String"/>
      <Level column="QUINZENA_LABEL" name="QUINZENA" type="String"/>
    </Hierarchy>
  </Dimension>

  <Dimension name="DATA">
    <Hierarchy hasAll="true" allMemberName="TODAS DATAS"  primaryKey="DATA"
primaryKeyTable="DATA">
      <Table name="DATA"/>
      <Level column="ANO" name="ANO" type="Numeric"/>
      <Level column="SEMESTRE_LABEL" name="SEMESTRE" type="String"/>
      <Level column="NOME_MES" name="NOME MES" type="String"/>
      <Level column="QUINZENA_LABEL" name="QUINZENA" type="String"/>
    </Hierarchy>
  </Dimension>

  <Dimension name="PRODUTO">
    <Hierarchy allMemberName="TODOS PRODUTOS" hasAll="true" primaryKey="CODIGO_PRODUTO"
primaryKeyTable="PRODUTO">
      <Table name="PRODUTO"/>
      <Level column="DESCRICAO_MARCA" name="MARCA" table="PRODUTO" uniqueMembers="false"/>
      <Level column="DESCRICAO_LINHA" name="LINHA" table="PRODUTO" uniqueMembers="false"/>
      <Level column="REFERENCIA" name="REFERENCIA" table="PRODUTO" type="Numeric"/>
      <Level column="MATERIAL" name="MATERIAL" table="PRODUTO" uniqueMembers="false"/>
      <Level column="COR" name="COR" table="PRODUTO" uniqueMembers="false"/>
    </Hierarchy>
  </Dimension>

  <Dimension name="MATERIAL">
    <Hierarchy allMemberName="TODOS MATERIAIS" hasAll="true" primaryKey="CODIGO_PRODUTO"
primaryKeyTable="PRODUTO">
      <Table name="PRODUTO"/>
      <Level column="MATERIAL" name="MATERIAL" table="PRODUTO" uniqueMembers="false"/>
    </Hierarchy>
  </Dimension>
```

```
  <Dimension name="COR">
    <Hierarchy allMemberName="TODAS CORES" hasAll="true" primaryKey="CODIGO_PRODUTO"
primaryKeyTable="PRODUTO">
      <Table name="PRODUTO"/>
      <Level column="COR" name="COR" table="PRODUTO" uniqueMembers="false"/>
    </Hierarchy>
  </Dimension>

  <Dimension name="PRAZO MEDIO">
    <Hierarchy allMemberName="TODOS PRAZOS" hasAll="true"
primaryKey="CODIGO_CONDICAO_PAGAMENTO" primaryKeyTable="CONDICAO_PAGAMENTO">
      <Table name="CONDICAO_PAGAMENTO"/>
      <Level column="PRAZO_MEDIO" name="PRAZO MEDIO" type="Numeric" uniqueMembers="false"/>
      <Level column="DESCRICAO" name="CONDICAO PAGAMENTO" uniqueMembers="true"/>
    </Hierarchy>
  </Dimension>

  <Dimension  name="CIDADE">
    <Hierarchy allMemberName="TODAS CIDADES" hasAll="true" primaryKey="CODIGO_CIDADE"
primaryKeyTable="CIDADE">
      <Table name="CIDADE"/>
      <Level column="REGIAO" levelType="Regular" name="REGIAO"/>
      <Level column="UF" levelType="Regular" name="UF"/>
      <Level column="NOME" levelType="Regular" name="NOME CIDADE"/>
    </Hierarchy>
  </Dimension>

  <Cube name="VENDA">
    <Table name="VENDA"/>
    <DimensionUsage foreignKey="CODIGO_CLIENTE" name="CLIENTE" source="CLIENTE"/>
    <DimensionUsage foreignKey="CODIGO_CLIENTE" name="GRUPO ECONOMICO" source="GRUPO
ECONOMICO"/>
    <DimensionUsage foreignKey="CODIGO_PRODUTO" name="MATERIAL" source="MATERIAL"/>
    <DimensionUsage foreignKey="CODIGO_PRODUTO" name="COR" source="COR"/>
    <DimensionUsage foreignKey="CODIGO_REPRESENTANTE" name="REPRESENTANTE"
source="REPRESENTANTE"/>
    <DimensionUsage foreignKey="CODIGO_TRANSPORTADORA" name="TRANSPORTADORA"
source="TRANSPORTADORA"/>
    <DimensionUsage foreignKey="DATA_INCLUSAO" name="DATA INCLUSAO" source="DATA INCLUSAO"/>
    <DimensionUsage foreignKey="DATA_ENTREGA" name="DATA ENTREGA" source="DATA ENTREGA"/>
    <DimensionUsage foreignKey="CODIGO_PRODUTO" name="PRODUTO" source="PRODUTO"/>
    <DimensionUsage foreignKey="CODIGO_CONDICAO_PAGAMENTO" name="PRAZO MEDIO" source="PRAZO
MEDIO"/>
    <DimensionUsage foreignKey="CODIGO_CIDADE" name="CIDADE" source="CIDADE"/>
    <DimensionUsage foreignKey="TIPO_PEDIDO" name="TIPO PEDIDO" source="TIPO PEDIDO"/>
    <DimensionUsage foreignKey="CODIGO_PREPOSTO" name="PREPOSTO" source="PREPOSTO"/>
    <Measure name="QUANTIDADE" column="QUANTIDADE" aggregator="sum"
      formatString="Standard"/>
    <Measure name="VALOR LIQUIDO" column="VALOR_LIQUIDO" aggregator="sum"
      formatString="$#,##0.00"/>
    <Measure name="VALOR BRUTO" column="VALOR_BRUTO" aggregator="sum"
      formatString="$#,##0.00"/>

    <CalculatedMember
      name="PERCENTUAL PRODUTO"
      dimension="Measures"
      visible="true">
      <Formula>([PRODUTO].CURRENTMEMBER) / ([QUANTIDADE], [PRODUTO].[TODOS
PRODUTOS])</Formula>
      <CalculatedMemberProperty name="FORMAT_STRING" value="%##0.00"/>
    </CalculatedMember>
    <CalculatedMember
      name="PERCENTUAL DATA ENTREGA"
      dimension="Measures"
      visible="true">
      <Formula>([DATA ENTREGA].CURRENTMEMBER) / ([QUANTIDADE], [DATA ENTREGA].[TODAS
DATAS])</Formula>
      <CalculatedMemberProperty name="FORMAT_STRING" value="%##0.00"/>
    </CalculatedMember>

  </Cube>
  <Cube name="DEVOLUCAO">
    <Table name="DEVOLUCAO"/>
    <DimensionUsage foreignKey="CODIGO_CLIENTE" name="CLIENTE" source="CLIENTE"/>
    <DimensionUsage foreignKey="CODIGO_CLIENTE" name="GRUPO ECONOMICO" source="GRUPO
ECONOMICO"/>
```

```
    <DimensionUsage foreignKey="CODIGO_REPRESENTANTE" name="REPRESENTANTE"
source="REPRESENTANTE"/>
    <DimensionUsage foreignKey="DATA_DEVOLUCAO" name="DATA DEVOLUCAO" source="DATA
DEVOLUCAO"/>
    <DimensionUsage foreignKey="CODIGO_PRODUTO" name="PRODUTO" source="PRODUTO"/>
    <DimensionUsage foreignKey="CODIGO_PRODUTO" name="MATERIAL" source="MATERIAL"/>
    <DimensionUsage foreignKey="CODIGO_PRODUTO" name="COR" source="COR"/>
    <DimensionUsage foreignKey="CODIGO_CIDADE" name="CIDADE" source="CIDADE"/>
    <DimensionUsage foreignKey="CODIGO_MOTIVO" name="MOTIVO DEVOLUCAO" source="MOTIVO
DEVOLUCAO"/>
    <Measure name="QUANTIDADE" column="QUANTIDADE" aggregator="sum"
      formatString="Standard"/>
    <Measure name="QUANTIDADE DEVOLVIDA" column="QUANTIDADE" aggregator="sum"
      formatString="Standard"/>
    <Measure name="VALOR" column="VALOR" aggregator="sum"
      formatString="$#,##0.00"/>
  </Cube>

  <Cube name="VENDADEVOLUCAO">
    <Table name="VW_VENDA_DEVOLUCAO"/>
    <DimensionUsage foreignKey="CODIGO_CLIENTE" name="CLIENTE" source="CLIENTE"/>
    <DimensionUsage foreignKey="CODIGO_CLIENTE" name="GRUPO ECONOMICO" source="GRUPO
ECONOMICO"/>
    <DimensionUsage foreignKey="CODIGO_REPRESENTANTE" name="REPRESENTANTE"
source="REPRESENTANTE"/>
    <DimensionUsage foreignKey="DATA" name="DATA" source="DATA"/>
    <DimensionUsage foreignKey="CODIGO_PRODUTO" name="PRODUTO" source="PRODUTO"/>
    <DimensionUsage foreignKey="CODIGO_PRODUTO" name="MATERIAL" source="MATERIAL"/>
    <DimensionUsage foreignKey="CODIGO_PRODUTO" name="COR" source="COR"/>
    <DimensionUsage foreignKey="CODIGO_CIDADE" name="CIDADE" source="CIDADE"/>
    <Measure name="QUANTIDADE VENDIDA" column="QUANTIDADE_VENDIDA" aggregator="sum"
      formatString="Standard"/>
    <Measure name="QUANTIDADE DEVOLVIDA" column="QUANTIDADE_DEVOLVIDA" aggregator="sum"
      formatString="Standard"/>
    <CalculatedMember
      name="PERCENTUAL DEVOLUCAO"
      dimension="Measures"
      visible="true">
      <Formula>[Measures].[QUANTIDADE DEVOLVIDA] / [Measures].[QUANTIDADE VENDIDA]</Formula>
      <CalculatedMemberProperty name="FORMAT_STRING" value="%##0.00"/>
    </CalculatedMember>

  </Cube>
</Schema>
```