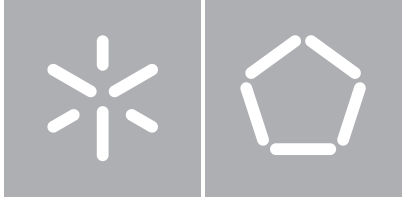
Universidade do Minho
Escola de Engenharia

Cláudio Campinho Novais

Uma Abordagem Alternativa para a Formalização de Espaços Multidimensionais em Sistemas OLAP

Outubro de 2013



Universidade do Minho
Escola de Engenharia
Departamento de Informática

Cláudio Campinho Novais

Uma Abordagem Alternativa para a Formalização de Espaços Multidimensionais em Sistemas OLAP

Dissertação de Mestrado
Mestrado em Engenharia Informática

Trabalho realizado sob orientação de

Professor Orlando Manuel de Oliveira Belo
Professor José Carlos Soares do Espírito Santo

Outubro de 2013

Aos meus familiares e namorada.

Agradecimentos

Embora uma dissertação seja, pela sua finalidade académica, um trabalho individual, há importantes apoios e incentivos que não podem e nem devem deixar de ser realçados.Por essa razão, é com muita satisfação que expresso aqui o mais profundo agradecimento a todos aqueles que tornaram a realização desta dissertação possível.

Agradeço em primeiro lugar aos meus pais por toda a paciência, pelo estímulo e grande amizade com que sempre me ouviram e sensatez com que sempre me ajudaram.Espero que esta etapa, que agora termino, possa, de alguma forma, retribuir e compensar o apoio incondicional e dedicação que, constantemente, me oferecem. Quero também agradecer à minha namorada que conseguiu provar que é dotada de muita paciência e que é capaz de incutir força de vontade mesmo nos piores momentos. Um obrigado também a todos os outros familiares que indiretamente me ajudaram.

Não posso deixar de parte a instituição que me acolheu no meu percurso académico, a Universidade do Minho, que fez solidificar a minha personalidade de uma forma à qual tenho orgulho. Momentos bons e momentos maus, momentos de pressão e momentos de alegria, um misto de sentimentos que juntamente com os ensinamentos dos vários professores me fizeram tornar no que sou hoje. Por esse motivo, agradeço a todos os professores, funcionários, colegas e amigos por todos os momentos que passei nesta academia.

Quero agradecer em particular o Departamento de Informática, a todos os professores, por me terem facilitado o percurso académico com as mais diversas ajudas. Não posso deixar de fazer um agradecimento particular ao Professor Doutor Orlando Belo pelos preciosos momentos de ensino e ao mesmo tempo bem estar, pela sua maneira única de ensinar. A sua retórica misturada com o profundo conhecimento no que leciona, tornou não só a minha formação facilitada como também acabou por me "contaminar" positivamente. Um agradecimento especial ao Professor Doutor José Carlos Espirito Santo pela dedicação e disponibilidade, mesmo em horários não usuais, na colaboração sempre que por mim foi solicitada. Acima de tudo, obrigado pela paciência e empenho que demonstrou ao longo das reuniões que se traduziram em autênticas aulas de matemática. Um sincero obrigado.

Resumo

Uma Abordagem Alternativa para a Formalização de Espaços Multidimensionais em Sistemas OLAP

A evolução dos sistemas OLAP tem sido extraordinária ao longo dos últimos anos. Desde os servidores de dados multidimensionais até às plataformas cliente para a exploração de dados é admirável o leque de soluções que emergiram, contribuindo para um grande aumento da sofisticação das plataformas de tomada de decisão. Porém, em contraste, não encontramos muitas propostas para a especificação e formalização dessas soluções, em particular no que diz respeito à forma como representamos e manipulamos dados multidimensionais. Como sabemos, uma especificação formal dessas estruturas contribuiria, em muito, para a correção, robustez e garantia da qualidade dessas mesmas soluções. Tendo isso em consideração, apresentamos nesta dissertação uma nova aproximação à formalização de espaços multidimensionais de dados utilizados pela generalidade dos sistemas OLAP, bem como a definição de uma das principais operações que se realiza sobre tais estruturas, o *roll-up*. Através de uma exposição baseada num exemplo muito prático, demonstramos a utilização do modelo de formalização definido, avaliando sempre que possível a sua aplicação e capacidade de representação.

Abstract

An Alternative Approach to Formalizing Multi-Dimensional Spaces in OLAP Systems

The evolution of OAL systems has been extraordinary over the past few years. Since the servers multidimensional data to the client platforms for data exploration is admirable range of solutions that have emerged, contributing to a large increase in the sophistication of decision making platforms. However, in contrast, we don't find many proposals for the specification and formalization of these solutions, in particular regarding to how we represent and manipulate multidimensional data. As we know, a formal specification of these structures would contribute greatly to the correctness, robustness, and guarantee of the quality of these solutions. With that in mind, this dissertation presents a new approach to the formalization of multidimensional data spaces used by most OLAP systems, as well as the definition of one of the most important operations that take place on these structures, Roll-up operation. Through an exposure based on a very practical example, we demonstrate the use of the defined formal model, by evaluating whenever it's possible their application and capacity of representation.

Índice

Índice de Figuras

Índice de Tabelas

Índice de Códigos

Capítulo 1

Introdução

1.1 Enquadramento

Os sistemas de suporte à decisão são hoje uma das tecnologias da informação mais emergentes devido às suas grandes potencialidades em termos de aplicação em ambientes industriais. Segundo um relatório da Gartner [Meulen e Rivera, 2013], as tecnologias de *Business Intelligence* (BI), às quais os sistemas de processamento analítico (OLAP) estão intrinsecamente ligados, são o quarto maior segmento de atividade e desenvolvimento na área do software. Estes sistemas catapultaram os processos de análise de dados e, consequentemente, de suporte à decisão para níveis de sofisticação nunca vistos. A análise de dados sempre foi uma atividade crítica em ambientes industriais. Dela dependem todos os processos de tomada de decisão que regulam, direta ou indiretamente, o dia a dia de uma qualquer empresa.

Os sistemas OLAP trouxeram novos meios e formas de organização e de exploração de dados, tornando os sistemas de suporte à decisão mais efetivos e flexíveis, com capacidade de se adaptarem rapidamente às mudanças que possam ocorrer em qualquer contexto de negócio. Através destes sistemas as empresas têm acesso a um novo leque de operações especialmente concebidas para o suporte de tarefas de análise de dados. Hoje, as consultas são dinâmicas e abordam simultaneamente mais do que uma dimensão de análise (multidimensional), acompanhando as várias perspetivas de abordar os negócios e permitindo diferentes níveis de sumarização de dados de acordo com esta ou aquela diretiva de negócio [Golfarelli e Rizzi, 2009].

A crescente adoção de sistemas OLAP por parte das empresas fez com que os processos de desenvolvimento destes sistemas se intensificassem e diferenciassem, não só para satisfazer

a procura de que estavam a ser alvos como também para se afastarem dos seus mais diretos concorrentes. Ao mesmo tempo que seguiam "rumos" de investigação e desenvolvimento, os investigadores e técnicos OLAP também foram usando e criando diferentes conceitos e metodologias, afastando-se, não raras vezes, daquilo que poderemos considerar como standards do domínio. Apesar disso, são vários os esforços que estão a ser feitos para tentar chegar a uma maior homogeneização desses conceitos e metodologias. Aliás, na literatura produzida na última década e meia é possível constatar os esforços científicos e técnicos que estão a ser realizados nesse sentido. Por exemplo, a estrutura temporal segundo múltiplas dimensões de análise adotada nos sistemas de *data warehousing* é hoje aceite como uma base sólida dos sistemas de suporte à decisão [Chaudhuri e Dayal, 1997]. Para além disso, numa área um pouco mais específica, a definição de espaços multidimensionais de dados através do conceito de hipercubo [Gray et al., 1997] criou uma visão mais unificada para as soluções de armazenamento de dados dos sistemas OLAP. Com a abordagem segundo as materializações dessas estruturas multidimensionais através de uma organização em reticulado (*lattice*) [Harinarayan et al., 1996], os hipercubos receberam uma nova estrutura, melhorada e organizada, com benefícios claros para os tão conhecidos modelos relacionais. Apesar dos hipercubos conterem na maioria dos casos mais do que três dimensões, estes são vulgarmente tratados simplesmente por cubos.

Todavia, a abordagem de materialização total de um cubo (por vezes utópica) declinou, uma vez que uma nova visão de navegação emergiu: para além da navegação sobre diferentes perspetivas ou dimensões de análise, passou a haver uma navegação segundo conjuntos de dados sobre um mesmo assunto, mas com diferentes níveis de sumarização.

1.2 Motivação e Objectivos

Apesar do caminho da formalização dos sistemas OLAP parecer estar traçado, há ainda uma carência efetiva de uma formalização com a capacidade de especificar todos os aspetos que uma área como esta envolve, nomeadamente especificar o EMD de forma abrangente para todos os casos práticos e naquilo que diz respeito a operações que podem ser realizadas sobre esse mesmo espaço. Atualmente, a dispersão de conceitos sobre as operações é muito grande. Para além de serem em grande número, muitos destes conceitos apresentam frequentemente definições contraditórias.

Assim, a motivação que levou a este estudo foi tão simples quanto criar um modelo que não só consiga abranger os vários aspetos que são relevantes na modelação de um sistema OLAP, mas também um modelo capaz de abranger as definições mais usais para afunilar o caminho que a literatura atual está a tender.

No sentido de atenuar um pouco a situação dispersa ao nível de conceitos, neste trabalho tivemos o objetivo de definir uma formalização alternativa para sistemas OLAP baseada

em reticulados, com enfoque particular nas estruturas multidimensionais de dados e suas operações de manipulação mais comuns. Essa formalização, apesar de alternativa, tem o objetivo de juntar os melhores conceitos teóricos e práticos dos sistemas OLAP para que seja um ponto de partida para uma unificação clara na sua modelação.

De uma forma mais específica, os objetivos deste trabalho consistiram em construir um modelo formal capaz de:

- explicar os vários aspetos de um EMD, nomeadamente, Tabelas de Factos, Dimensões, atributos de dimensão, métricas e hierarquias;
- considerar aspetos práticos como atributos descritivos, garantia de correção da estrutura do EMD, construção dessa mesma estrutura (reticulado);
- possibilitar a realização das operações mais comuns em OLAP.

1.3 Estrutura da Dissertação

De forma a expormos esta nossa abordagem, organizámos este documento segundo as várias vertentes de fundamentação e descrição do modelo de formalização desenvolvido. Assim, depois deste capítulo introdutório onde apresentamos genericamente a situação atual dos sistemas OLAP, explicamos de forma mais concreta o que são estes sistemas, qual a sua relevância e como normalmente são construídos para as empresas.

De seguida, no capítulo 3, apresentamos de forma muito concreta e formal estes sistemas OLAP através dos vários estudos apresentados na literatura da especialidade. São apresentadas não só as bases onde os sistemas atuais se centram, mas também alguns modelos formais que tentam explicar estes sistemas. No final do capítulo é apresentado o resultado de um estudo comparativo entre os vários modelos, onde se pode ver claramente que há dispersão na formalização destes sistemas.

Uma vez que o capítulo 3 é praticamente todo ele teórico, no capítulo 4 abordamos os mesmos modelos mas com base num exemplo prático. Assim é possível constatar de forma mais concreta as várias diferenças e similaridades que existem entre os vários modelos que têm vindo a ser apresentados na literatura. No final do capítulo existe um comparativo entre os vários modelos através do exemplo prático.

No capítulo 5 introduzimos e descrevemos o modelo proposto, abordando o tema desde a definição mais básica, como é o caso de um atributo, até à operação de *roll-up*. Acompanhando a definição, demonstramos a sua aplicação através do exemplo prático utilizado no capítulo 4.

Este documento termina com as conclusões que se destacaram no desenvolvimento deste modelo e enunciam-se algumas linhas de orientação para trabalho futuro, para a evolução e solidificação deste modelo.

Capítulo 2

O que é um sistema OLAP?

Um sistema OLAP é uma ferramenta muito útil no suporte a qualquer atividade relacionada com os processos de tomada de decisão. Os serviços que providenciam para o tratamento e análise de informação especializada faz deles elementos de trabalho bastante preciosos para qualquer agente de decisão empresarial.

Segundo estudo referido na Introdução [Meulen e Rivera, 2013], em 2013 espera-se que os lucros nesta área subam 7%, para cerca de 13.8 mil milhões de dólares. Isto depois de no ano anterior ter crescido 16%. Em 2016 espera-se que os lucros rondem os 17 mil milhões de dollares, valores muito elevados mas que, de certa forma, são naturais visto que são relativos a ferramentas que se forem bem utilizadas podem fazer as empresas lucrarem ainda mais.

Estas ferramentas são utilizadas para dar suporte às decisões tomadas pelas empresas pois têm uma estrutura multidimensional e um conjunto de operações otimizadas precisamente para análise segundo diferentes áreas de negócio. Para se entender melhor, nas secções seguintes são apresentadas informações úteis para a compreensão e relevância dos sistemas OLAP.

2.1 Relevância dos sistemas OLAP

Através dos sistemas OLAP é introduzido um novo leque de operações adaptadas especificamente a tarefas de análise de grandes volumes de dados. As consultas passam a ser dinâmicas e multidimensionais (realizadas segundo várias perspetivas de análise ou dimensões de negócio e segundo diferentes níveis de sumarização) sobre quantidades elevadas de dados [Golfarelli e Rizzi, 2009].

Devido ao elevado potencial de análise que estes sistemas acarretam, são amplamente explorados pela indústria, que procura resultados para os seus problemas concretos. Na sua maioria, os sistemas são feitos à medida ou de forma fechada (sistemas proprietários) o que os torna menos conhecidos ao nível da sua construção.

Contudo, a sua utilização em bases de dados gigantes é quase como normal, ou até obrigatória, devido às inúmeras possibilidades que advêm destes sistemas. Simples análises de mercado, conforme áreas de comercialização, tipos de produtos ou até ao longo do tempo, são possíveis de ser feitas pelos Agentes de Decisão sem terem de ser obrigados a analisar cada transação feita pela empresa e com tempos de espera dos resultados reduzidos (visto que costumam ser grandes volumes de dados).

Esta dinâmica e velocidade dos sistemas OLAP advém da sua estrutura altamente redundante, que contém os dados pré-processados. Porém isto acarreta algumas dificuldades, principalmente de espaço ocupado. No geral, existe três formas de ter estes sistemas [Chaudhuri et al., 2001]:

- *Multidimensional OLAP* (MOLAP): todos os dados (incluindo os materializados) são armazenados na memória, em *arrays*;
- *Relational OLAP* (ROLAP): os dados são armazenados em modelos relacionais (bases de dados);
- *Hybrid OLAP* (HOLAP): tal como o nome indica é um híbrido entre os dois primeiros. A sua estrutura depende da implementação, mas normalmente os dados com maior detalhe são guardados em bases de dados e os materializados (ou parte deles) estão em *arrays* (possivelmente em memória para aumentar a velocidade).

Uma vez que os tipos de sistemas apresentados anteriormente estão relacionados com a sua implementação, eles não têm interferência direta ao nível conceptual. Portanto, explicando de forma abstrata, estes sistemas são constituídos por duas partes:

- O **espaço** que pode ser considerado como um conjunto de bases de dados dispostas de forma organizada (Figura 2.4), cada uma contendo a mesma informação, mas explicada de maneiras diferentes.
- As **operações** que permitem navegar pelo espaço.

2.2 O que é um Espaço Multi-Dimensional (EMD)

Pela sua própria natureza, os sistemas OLAP são altamente redundantes. Essa característica permite-lhes conseguir satisfazer de forma muito ágil os pedidos dos seus utilizadores.

De facto, isso só é possível porque "todos" os resultados de "todas" as possíveis *queries* que podem ser lançadas sobre as suas estruturas multidimensionais de dados, incluindo os resultados que se possam obter nos diversos níveis de agregação de dados suportados pelas hierarquias definidas (apresentadas mais abaixo), que foram previamente processados e materializados na estrutura do cubo de dados envolvido. Desta forma, as operações de análise sobre os dados contidos nesse cubo não são mais do que "meras" consultas a uma ou mais células desse cubo. Podemos assumir que, no momento da operação de consulta não se realizam qualquer tipo de operação de cálculo.

Um EMD é, portanto, uma estrutura de dados composta por várias dimensões relacionadas entre si. Cada dimensão é um eixo de análise que caracteriza uma área informativa, por exemplo, os dias do ano, as características de um produto vendido, ou as características de uma localização, entre muitos outros aspetos. O relacionamento entre as dimensões caracteriza factos ocorridos. Por exemplo, se se cruzar as informações dos três exemplos anteriores, tem-se factos que expressam as vendas de um produto consoante o tempo e na localização especificada.

O cruzamento entre todas as dimensões é caracterizado pelo nível máximo de detalhe, que define factos verdadeiros que são originalmente disponibilizados pelo DW, numa estrutura de dados chamada tabela de factos. Todos os outros cruzamentos entre dimensões e/ou atributos de dimensões, na prática são materializações pré-processadas com base em tabelas mais detalhadas, nomeadamente a tabela de factos.

2.2.1 O Data Warehuose (DW)

Para se compreender melhor um EMD é necessário apresentar a sua base de sustentação, mostrar os seus conceitos mais elementares. Portanto, um sistema OLAP, normalmente, começa pela construção de um DW, que basicamente é uma estrutura de dados (pode ser uma base de dados) que tem uma organização com uma característica distinta: é uma coleção de dados integrados, orientados por assunto, não voláteis e variáveis em relação ao tempo. [Inmon, 2005]

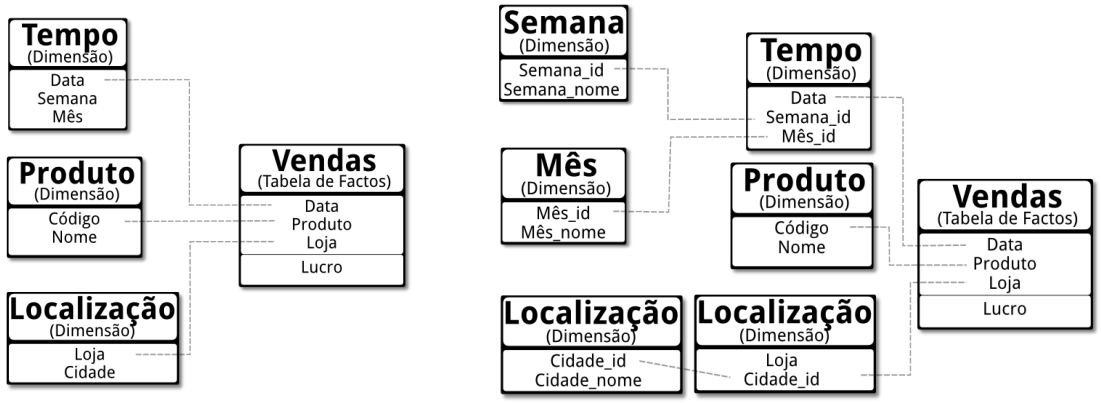
Uma vez que a sua estrutura é exclusivamente direcionada à análise (leitura dos dados), como suporte à decisão, ela é extremamente simples e resume-se a uma lista de factos ou acontecimentos que são o cruzamento entre todas as dimensões ou áreas a analisar (eixos de análise) do DM. Esse cruzamento de dados é feito por um dos elementos principais de um DW: a tabela de factos. Esta tabela tem dois tipos de informação:

- **chaves estrangeiras** (para as dimensões) que caracterizam o acontecimento;
- **métricas** que valorizam o que aconteceu.

Conforme a aplicação prática, as chaves estrangeiras poderão ser simples apontadores para dimensões que são tabelas, ou então poderão ser apenas colunas com informação útil, sem apontar para uma dimensão (tabela) – conceptualmente isto chama-se dimensão degenerada.

No geral, uma dimensão caracteriza um aspeto de um facto que está avaliado (medido) na tabela de factos. Uma dimensão pode ser por exemplo uma simples descrição sobre um assunto que fez o facto ser definido. No entanto, uma dimensão pode ser também uma descrição que inclua diferentes categorias nas quais essa informação é abrangida. Essas categorias, depois podem ser exploradas pelos Agentes de Decisão para simplificar as consultas.

Para se perceber rapidamente como uma tabela de factos funciona, considere-se que se quer analisar as vendas de uma empresa segundo 3 diferentes eixos de análise: a caracterização do produto, o local que ele foi vendido e em que altura foi vendido. Este exemplo prático é explorado no Capítulo 4. Então, cada um desses aspetos que se quer analisar são caracterizados nas dimensões e o cruzamento entre todas as dimensões define os factos. Esse cruzamento fica materializado na tabela de factos que mede cada facto pela métrica "lucro". Conceptualmente, essa estrutura pode ser vista na Figura 2.1a.



(a) Estrutura (em Estrela) das dimensoes e tabela de factos

(b) Estrutura (em Floco-de-Neve) das dimensões e tabela de factos

Figura 2.1: Um possível DM construído com dois esquemas diferentes

Rapidamente se consegue constatar que dentro da dimensão "Tempo", há atributos que englobam (categorizam) outros atributos da mesma dimensão. Por exemplo, o atributo "data" (que significa o dia de um ano) é abrangido pelo atributo "Mês". Nestes casos, os programadores poderão normalizar as dimensões para que a base de dados fique menos redundante. No entanto, uma normalização deste tipo pode tornar ineficiente o sistema OLAP e por isso existe opiniões divergentes quanto a este assunto.

Por existir estas divergências, há dois esquemas comuns para a construção de um DM: o esquema em estrela, sem normalização, representado pela Figura 2.1a e defendido por Kimball e Ross [2002]; e o esquema em floco-de-neve em que normaliza as dimensões, representado na Figura 2.1b, defendido por Inmon [2005].

Conforme cada tipo de implementação, os programadores poderão decidir normalizar ou não as tabelas das dimensões, no entanto elas não deixam de ser dimensões. Conceptualmente, uma dimensão é uma estrutura de dados que caracteriza uma área de negócio. Essa caracterização normalmente, e tal como referido, tem atributos que englobam outros para os categorizar – hierarquias de atributos. As hierarquias de uma dimensão definem a maneira como os seus atributos podem ser agregados, de forma a permitir uma vista mais concisa sobre uma determinada linha de exploração de um dado processo de análise de dados.

Existe várias maneiras de expressar as hierarquias (e o DM). A forma mais comum e simples de expressar uma hierarquia é através de grafos. Existe também um modelo interessante proposto por Golfarelli e Rizzi [2009] chamado "*Dimensional Fact Model*" que está representado na Figura 2.2. Este modelo caracteriza-se por ter as várias dimensões à volta da tabela de factos e expressas através de ramificações. Existe vários outros pormenores para uma modelação completa de um DM, nomeadamente tipos de atributos (descritivos ou não) e que métricas são afetadas pelas hierarquias.

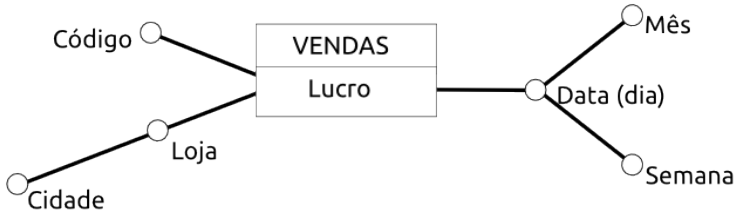


Figura 2.2: Representação de um possível modelo conceptual da estrutura da Figura 2.1a

Relativamente ao modelo representado na Figura 2.2, no centro, está a tabela de factos, chamada "Vendas" que contém apenas uma métrica, "lucro". Depois, existem três ramificações independentes, ou seja existe 3 dimensões ligadas à tabela de factos. No caso da dimensão do lado esquerdo com apenas um nodo, ela representa uma dimensão com apenas um atributo. Relativamente à ramificação logo abaixo, esta representa a dimensão "Localização" em que existe um hierarquia: o atributo "Cidade" que engloba o atributo "Loja". Relativamente à dimensão do lado direito, esta tem uma hierarquia um pouco mais complexa que dita que o atributo "Data" pode ser agregável pelos atributos "Mês" e "Semana", no entanto, estes dois como não têm qualquer ligação não são agregáveis entre eles.

O esquema da Figura 2.1a pode ser visto de uma forma menos conceptual na Figura 2.3, onde existe 4 tabelas definidas com um esquema em Estrela. Através desta figura, é possível

concluir facilmente que existe acontecimentos descritos na tabela de factos que podem ser explorados com informação útil, adicional, através das dimensões. Por exemplo, na primeira linha podemos ver que no dia 1 de Janeiro foi comprado um boné em Braga que rendeu (métrica) 3 unidades monetárias de lucro. Ou seja, não estamos a cingir apenas na tabela de factos mas estamos já a buscar informação das dimensões que dão nome ao código do produto e qual a localização dessas lojas.

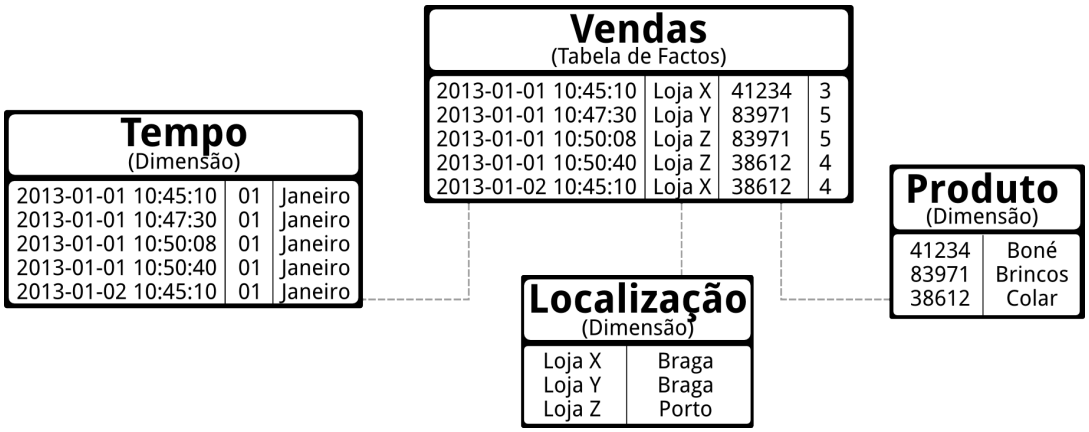


Figura 2.3: Exemplo prático de um DW

Uma vez que as Tabelas de Factos tendem a ter milhões de registos, a sua leitura é quase humanamente impossível. Daí que os Agentes de Decisão tendem a fazer análises de alto nível, análises com um menor nível de detalhe, ou seja, com os dados agregados para tornar a leitura mais sucinta. Essas agregações podem ser de vários níveis, por exemplo em vez de analisar ao dia, analisar os dados ao mês (agrupando os dados por um outro atributo da dimensão que seja mais abrangente); ou então, em vez de analisar os dados dos produtos pelo modelo, analisar pela categoria; ou simplesmente remover uma das dimensões, que neste caso poderia ser uma análise dos dados considerando apenas o "Tempo" e o "Produto", descartando a dimensão "Localização". Estas agregações podem ser de vários tipos, nomeadamente somatórios, médias, máximos, entre outras funções que possam ser úteis aos Agentes de Decisão.

2.2.2 A organização materializada do sistema OLAP

Todos estes pedidos dos agentes de decisão seriam possíveis numa base de dados com uma estrutura comum – *On-Line Transaction Processing* (OLTP)–, no entanto, os tempos de espera para cada consulta seriam consideráveis se os registos fossem muitos. E isto ocorreria por vários motivos: o sistema estava sobrecarregado a controlar os dados da empresa; a organização altamente normalizada prejudicaria a performance; processar este tipo de consultas requer tempo que os Agentes de Decisão não querem esperar.

E é precisamente por causa desta situação que os EMDs têm mais uma característica especial: eles têm a informação pouco normalizada e já pré-processada (materializada) para que as consultas dos dados possam ser realizadas com tempos de resposta muito curtos. Essa estrutura pré-processada dos dados normalmente é organizada num reticulado que permite, entre vários outros aspetos:

- produzir materializações parciais através de algoritmos de otimização, visto que materializações totais aumentam exponencialmente ao nível do tempo de processamento e do espaço ocupado consoante o número de dimensões [Morfonios et al., 2007];

- permite tornar a navegação dos dados entre diferentes níveis de agregação mais óbvia.

Um reticulado (muitas vezes chamado pela sua denominação em inglês *lattice*) tem uma estrutura conceptual característica que pode ser vista nas Figuras 4.4b e 4.5b. De uma forma mais prática, ela dita a organização das tabelas que resultam de pré-processamentos sobre a tabela de factos. Essas tabelas diferem da tabela de factos em níveis de agregação, que podem vir de duas situações: níveis de agregação internos das dimensões; cruzamentos com menos dimensões. Analisando a Figura 2.4, em que é representado um caso prático de um exemplo muito simples de 3 dimensões sem hierarquias, constata-se rapidamente que são muitas tabelas resultantes para algo tão básico. Acima da tabela de factos, encontram-se mais 7 tabelas que são resultado de todas as combinações possíveis entre as 3 dimensões (e combinações de hierarquias, se existissem). Cada uma dessas tabelas contém uma combinação diferente e portanto cada uma delas tem um conjunto de relacionamentos para dimensões diferente. Ou seja, na prática, a estrutura de um sistema OLAP é o conjunto de tabelas em que todas elas exceto o nodo superior (que tem os dados todos resumidos num só registo) têm pelo menos um relacionamento com dimensões.

Com esta estrutura, garante-se que a maioria dos pedidos que um Agente de Decisão pode fazer estão já calculados. Isto faz com que a performance de um sistema OLAP seja sempre algo muito rápido a agregar os dados. Isto ocorre porque esses dados não são agregados no momento, na verdade existe apenas passagens entre vistas que representam a combinação entre os eixos de análise que se está a analisar. Esta forma de analisar os dados através de vistas com diferentes perspetivas de análise, levaram ao conceito de Hipercubo (ou simplesmente cubo) [Gray et al., 1997] referido na Secção 3.1. Esta denominação ocorre porque os analistas conceptualmente passam a ver os dados em perspetiva, em que quando se está perante um sistema com 3 dimensões, ele pode ser mesmo representado por um cubo.

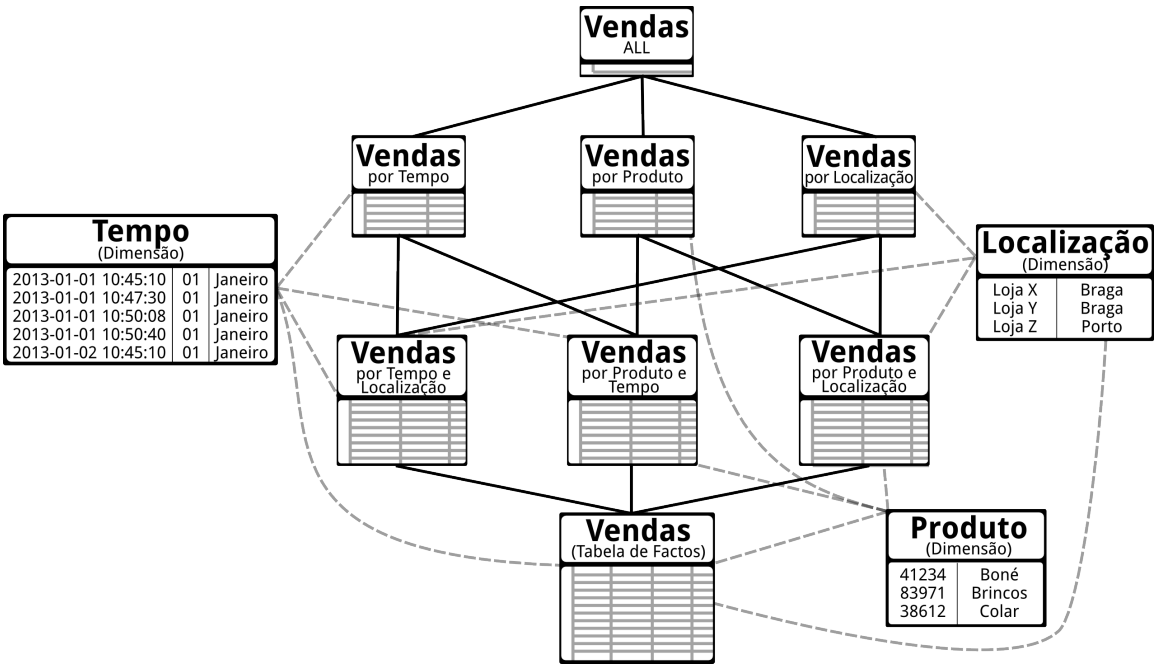


Figura 2.4: Na prática um reticulado é um conjunto de tabelas correlacionadas a apontar (com relacionamentos) para as dimensões, tal como a tabela de factos

2.3 Operações OLAP

Por cima do EMD é necessário uma camada que abstraia toda a sua estrutura multidimensional e repetitiva e apresente algo natural de navegar pelos dados. Desta forma, em vez de um simples operação de *Structured Query Language* (SQL) sobre cada uma das tabelas, os utilizadores do sistema passam a ter um leque de operações que abstraem todas estas tabelas com os mesmos dados mas com diferentes níveis de sumarização.

O maior problema ao nível conceptual dos sistemas OLAP está precisamente na definição de operações. Ao contrário dos sistemas OLTP que têm a Álgebra Relacional altamente estudada para produzir consultas SQL, nos sistemas OLAP existe uma "nuvem de indecisões". São inúmeros os artigos que expressam à sua maneira quais são as operações OLAP, muitas vezes operações com nomes diferentes, outras com discordâncias. Na verdade, a não ser algumas operações definidas de maneira contrária (conflituosa com outras) elas apresentam sempre um objetivo claro: permitir ao Agente de Decisão navegar (agregar e desagregar) pelo EMD e filtrar os dados.

Como foi referido anteriormente, na literatura desta área temos presenciado um culminar dos vários modelos e operações. De entre as muitas conjeturas, algumas têm maior des-

taque e começam a ser mais comuns noutros estudos. No caso das operações, as duas de navegação, *roll-up* e *Drill-Down*, têm claro destaque, visto que um sistema OLAP tem muitas materializações dos mesmos dados. Uma outra operação muito comum, apesar de ter diferentes definições conforme os autores, é o *Slice&Dice*, pois é a operação que permite filtrar os dados.

Mas que operações são realmente comuns na literatura especializada nesta área? Uma vez que existe muitas operações definidas pelos diversos autores de estudos do OLAP, algumas delas contraditórias, outras repetidas e outras que são resultado de operações mais simples, foi necessário procurar saber quais as operações mais comuns no OLAP. Segundo Romero e Abelló [2007], as operações mais comuns, baseadas numa análise de cerca de 20 artigos da área do OLAP são:

- Seleção
- Projeção
- *roll-up*
- União
- *Drill-Across*

No caso das operações Seleção e Projeção, estas constituem a referida operação *Slice&Dice*. Relativamente ao *roll-up* é uma operação importantíssima para permitir navegação para dados mais sumarizados, agrupados. Romero e Abelló [2007] não referem a operação *Drill-Down* como uma operação comum, pois ao nível de vistas temporárias isso seria impossível de ser realizado. No entanto, numa estrutura materializada e organizada num reticulado, essa operação é tão importante como o *roll-up*.

Relativamente às outras duas operações, a união e o *Drill-Across*, elas funcionam de maneira semelhante e têm um propósito importante no OLAP: unir cubos de dados. No caso da primeira, ela permite juntar informações de dois cubos distintos de acordo com a sua semântica, de forma similar à união no SQL. O *Drill-Across* é uma operação muito semelhante à União, no entanto exige que a operação seja feita sobre dois cubos que partilhem o mesmo EMD.

De seguida apresentam-se então as operações que iremos aplicar no modelo que estamos a desenvolver. De salientar que atualmente apenas temos definida a operação de *roll-up*. As operações escolhidas têm objetivo de "trabalharem" o cubo definido e, portanto, não são abordadas operações que juntam outros cubos.

2.3.1 Operação *Slice&Dice*

Esta operação permite escolher um cuboide (nodo do reticulado) e filtrá-lo, à imagem das consultas SQL. O nome *Slice&Dice* vem do facto de visualmente a operação ser um corte num cubo, escolhendo apenas a informação que se quer analisar.

Na prática, esta operação é uma Projeção e Seleção dos dados que um Agente de Decisão quer analisar. O *Slice&Dice* permite escolher os eixos de análise (cuboides) e permite filtrar apenas uma porção do cubo. Esta seleção dos eixos de análise é a escolha de um cuboide do reticulado (que pode ou não estar materializado). Uma vez que o Agente de Decisão define exatamente que dados quer analisar, esta operação não exige uma posição inicial, ao contrário das operações de navegação descritas na secção seguinte.

Esta operação está representada na Figura 2.5, em que é escolhido o cuboide com 2 dimensões, "Tempo" e "Produto", com dados apenas do mês Janeiro. O que está a acontecer é mediante uma tabela de factos (representada pelo cubo da esquerda), escolhe-se apenas os dados que interessa (que estão a cinzento). O resultado é um outro cubo (do lado direito) mas de menor tamanho e dimensionalidade.

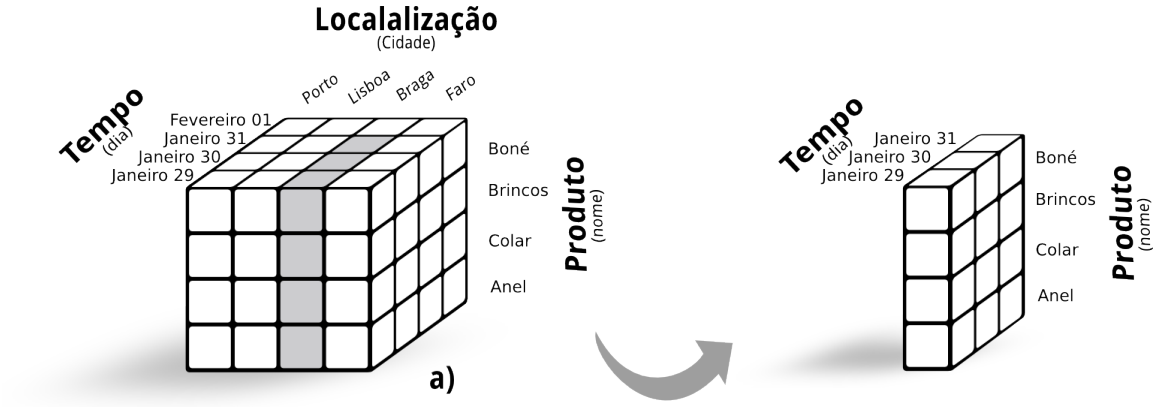


Figura 2.5: Operação de *Slice&Dice*, em que são escolhidas apenas duas dimensões e parte da dimensão "Tempo"

2.3.2 Operações de navegação

A construção de um reticulado não existe apenas para se saber quantas combinações possíveis se pode obter com as dimensões e suas hierarquias. Na verdade o conceito de reticulado encaixa muito bem num sistema OLAP pois para além de se saber que combinações são possíveis, ainda permite definir quais os cuboides que estão interligados entre eles, com dependências (numa hierarquia).

Ora, essas "ligações" permitem tornar os sistemas OLAP completamente dinâmicos e ágeis, pois o Agente de Decisão pode navegar entre os dados de forma eficaz (mais rápida). Por exemplo se tiver a ver um cubo com 3 dimensões e quiser remover uma delas, ele simplesmente navega para outro cuboide, evitando que o sistema faça um processamento que por

vezes é absurdo por ser agregações de milhões de registos. O mesmo acontece na operação inversa, em que não seria possível construir um inverso de uma agregação sem se ter algo detalhado. No entanto, como os dados estão já materializados, essa operação é possível e igualmente rápida.

As duas operações apresentadas a seguir exigem que haja um ponto de partida (cuboide) e um destino (outro cuboide).

roll-up

Genericamente, a operação *roll-up* permite diminuir a granularidade dos dados que se está a analisar, ou seja, permite agrupar os dados segundo algum eixo de análise associado. Se estivéssemos a referir a uma base de dados normal, estaríamos mediante uma operação simples de GROUP BY com uma função de agregação nas métricas (SUM, AVG, entre outras operações). Mediante a estrutura do reticulado, que na prática é o convencional de um sistema OLAP, compreende-se que esta operação de agregação na verdade é uma operação de navegação entre cuboides do reticulado, ou seja, troca de tabelas na consulta.

Essa navegação tem um sentido único, navegar para cuboides mais sumarizados. Vendo a Figura 2.4, esta operação permite então navegar apenas de cima para baixo, uma vez que nesse reticulado os cuboides superiores têm a informação mais agregada que os cuboides inferiores.

Conceptualmente esta operação apenas dita qual o cuboide a ser escolhido para analisar. No entanto na prática, isto não é exatamente assim. Existe muitos casos em que os reticulados estão construídos parcialmente pois é inviável a construção completa, devido à carga de processamento e capacidade de armazenamento. Nesses casos, uma operação de *roll-up* pode ser uma navegação ou então um GROUP BY dos dados que se está a analisar, mediante a existência ou não do cuboide pedido pelo Agente de Decisão.

Neste caso, atendendo a que estamos a estudar situações conceptuais, não se aborda essa implementação prática dos sistemas OLAP, considerando então que o reticulado está sempre 100% materializado. Assim, como exemplo, tendo em conta a Figura 2.4, se um Agente de Decisão tivesse a visualizar os dados da tabela de factos, ou seja do cuboide mais abaixo, e precisasse de remover a dimensão "Localização", ele faria um *roll-up* em que na prática passaria a visualizar os dados do cuboide que fica exatamente por cima, tal como se pode ver pela Figura 2.6. A execução do *roll-up* não é genérica, o Agente de Decisão tem de escolher que dimensões quer analisar, visto que há várias possibilidades de agregação dos dados em questão.

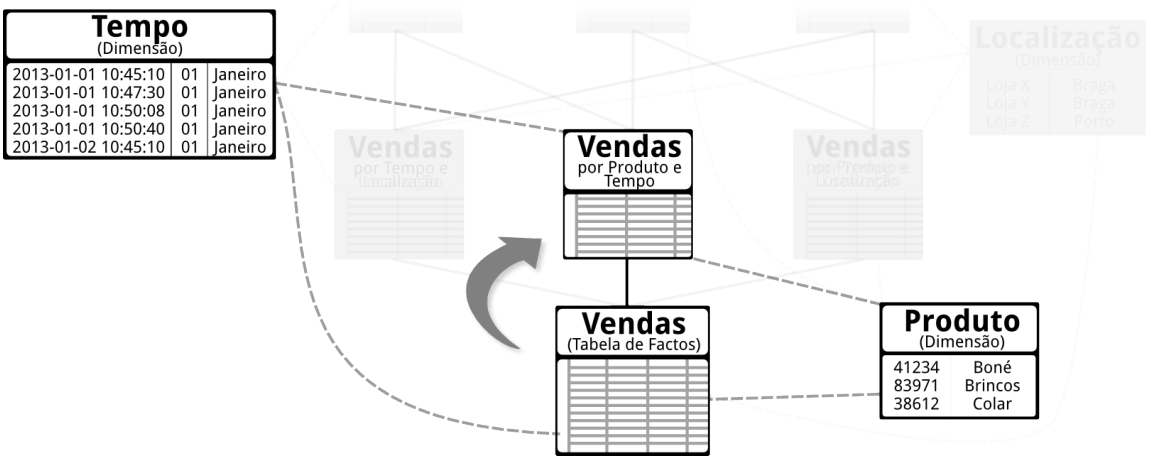


Figura 2.6: Operação de *roll-up*, em que é removida a dimensão "Localização"

Drill-Down

A operação de *Drill-Down*, conceptualmente, é a operação inversa do *roll-up*: navegar para cuboides com maior nível de detalhe. No entanto, é uma operação que tanto matematicamente como na prática é impossível de ser inversa.

Matematicamente, não é possível obter informação mais detalhada de algo que já foi calculado (agregado). Não é possível desagrupar algo sem conhecer as suas origens. Ao nível prático, utilizando um EMD em reticulado, a operação também não é exatamente uma simples navegação caso se esteja perante um reticulado parcial (onde voltamos à situação de não existir dados detalhados).

Na prática, a operação *Drill-Down* só é conveniente num sistema OLAP convencional, sobre um reticulado de dados materializados. Sem ela, o *Drill-Down* não passa de uma operação *Slice&Dice* e um GROUP BY sobre uma tabela de factos (pois não é possível desagrupar dados). Com o reticulado, a operação é quase inversa ao *roll-up* mas exige que haja cuboides materializados com dados mais detalhados para navegar para eles. Novamente, caso não existam, numa situação prática a operação tem de procurar um cuboide que esteja abaixo no reticulado (seja mais detalhado e tenha um caminho na hierarquia) e depois agrupar conforme a situação pedida.

Da mesma forma que explicado na secção anterior sobre o *roll-up*, neste trabalho centramo-nos apenas ao nível conceptual e portanto a operação *Drill-Down* é simplesmente a operação inversa ao *roll-up*, de movimentação. Ou seja, é uma operação em que exige que haja um ponto inicial em que o Agente de Decisão está a analisar e essa operação acrescenta detalhe aos dados, que na prática é uma movimentação para um cuboide com maior detalhe.

Visualmente, considerando novamente a Figura 2.4, e supondo que o Agente de Decisão está a analisar o cuboide que inclui apenas a dimensão "Tempo", se ele precisar de ver os dados mais detalhados, por exemplo acrescentando o eixo de análise referente aos dados da dimensão "Localização", ele faz um *Drill-Down* a acrescentar essa dimensão. O que o utilizador do sistema vê é um desmontar dos dados agregados, acrescentando mais informação detalhada. Apesar do processo parecer complexo, na prática, o sistema apenas muda a vista que está a apresentar. No caso desta situação, ela está representada na Figura 2.7 e basicamente está a passar da vista que incluía apenas a dimensão "Tempo" para a vista que incluía os dados da combinação entre a dimensão "Tempo" e a dimensão "Localização".

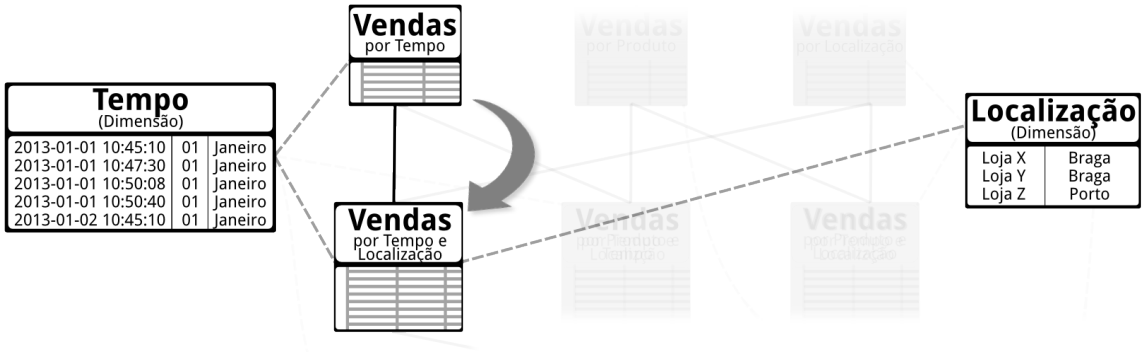


Figura 2.7: Operação de *Drill-Down*, em que se acrescenta a dimensão "Localização" à vista que se estava a analisar

2.3.3 Operação de *Pivoting*

A operação de *Pivoting* é uma operação de reorganização da vista apresentada ao Agente de Decisão. Geralmente, esta operação é associada ao resultado final apresentado ao Agente de Decisão, à folha de cálculo (tabela) que ele visualiza.

Há um conjunto considerável de estudos [Chaudhuri e Dayal, 1997; Gray et al., 1997; Thomas e Datta, 2001] que sugerem que esta operação é uma reestruturação e cálculo da mesma. Estes autores consideram *Pivoting* como uma transformação da folha de cálculo ao ponto de as métricas poderem passar a ser eixos de análise.

Um exemplo prático dessa situação está representado na página 18, em que inicialmente o Agente de Decisão está a visualizar uma vista de um cuboide de forma bruta (Tabela 2.1a), diretamente da base de dados, e depois passa a visualizar os dados de forma mais percetível (Tabela 2.1b) através da aplicação da operação de pivoting que transforma os valores de duas dimensões em colunas. Estes valores referidos são agrupados por ano e também por

Dia	Local	Produto	Lucro
1/Jan	Braga	Boné	3
1/Jan	Braga	Brincos	5
1/Jan	Porto	Boné	5
1/Jan	Porto	Brincos	4
2/Jan	Braga	Boné	1
2/Jan	Braga	Brincos	6
2/Jan	Porto	Brincos	3

(a) Vista do conteúdo original de um cuboide

	Localização/Produto			
	Braga		Porto	
Dia	Boné	Brincos	Boné	Brincos
1/Jan	3	5	5	4
2/Jan	1	6		3

(b) Resultado após uma operação de *Pivoting* aplicado à vista representada na Tabela 2.1a

Tabela 2.1: Aplicação da operação de *Pivoting* segundo alguns autores

localização.

Apesar de ser algo bastante útil para um Agente de Decisão, consideramos que isto é processamento de um outro nível, quase da camada de apresentação. Aliás, o resultado deste tipo de operação não é uma tabela relacional.

Como este trabalho centra-se num baixo nível do OLAP e de forma bastante conceptual, consideramos que a operação *Pivoting* realiza apenas uma rotação no cubo, ou seja, é uma reorganização da vista, sem reagrupamento dos dados. Esta visão da operação de *Pivoting* é apoiada por outros autores, nomeadamente Romero e Abelló [2007]. No caso de uma vista com duas dimensões (uma tabela clássica), *Pivoting* seria simplesmente trocar as colunas pelas linhas. No caso de algo complexo, podemos considerar que é uma rotação do Cubo, algo que é representado na Figura 2.8 em que um cubo com 3 dimensões é rodado na direção do leitor.

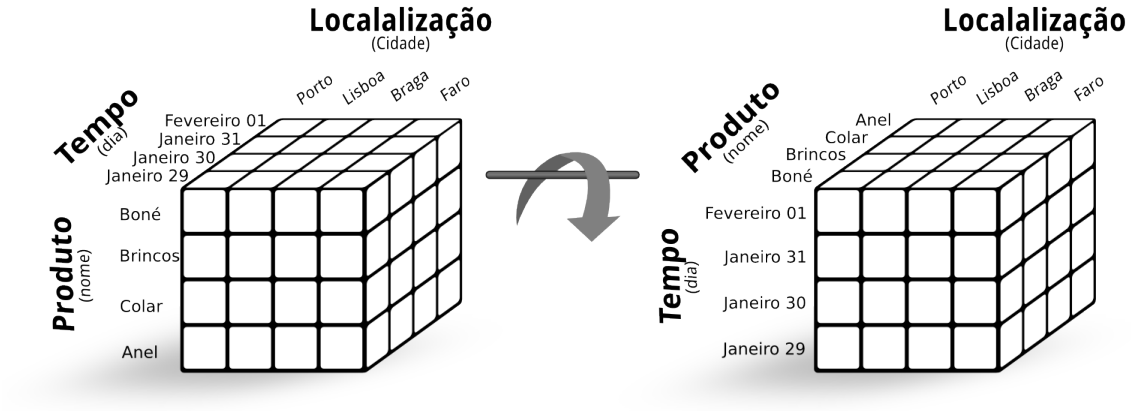


Figura 2.8: Operação de *Pivoting* é uma simples rotação, reorganização das vistas

Capítulo 3

Hipercubos e Espaços Multidimensionais

Os sistemas OLAP estão direcionados para um objetivo cada vez mais procurado: rentabilizar os processos das empresas analisando resultados. A procura existe pois a eficácia destes sistemas tem retornos consideravelmente altos nas várias aplicações em que são usados.

Uma vez que os sistemas OLAP cativaram rapidamente as grandes empresas, a sua procura e execução prática tem sido cada vez maior, o que produziu o desfasamento teórico destes sistemas.

Atualmente na literatura encontra-se vários modelos que tentam explicar todo o funcionamento do OLAP com base em tabelas de factos. A existência desses vários modelos é um indício de que ainda se tenta desenvolver uma prova generalizada que seja aceite por todos.

Apesar da independência desses modelos, na realidade eles têm vindo a convergir em algo bastante concreto. No entanto, esse "estado concreto" ainda tem muitas discrepâncias de conceitos entre as várias conjeturas apresentadas e atualmente ainda não existe um modelo consensual que explica na totalidade os EMDs.

Dos vários modelos existentes, alguns são definidos apenas com base em provas SQL (muito dependentes de implementações), outros com base em linguagem/metalinguagem de programação e alguns outros em formalização mais matemática.

3.1 Hipercubo: Operador de agregação

O primeiro modelo de representação do OLAP apresentado neste documento é o hipercubo, conjeturado por Gray et al. [1997], que é talvez o mais aceite na área. Esta aceitação refere-se acima de tudo pelo conceito de espaço e não pela formalização em si, uma vez que esta é feita com base na linguagem SQL.

Apesar de a formalização ser extensa, uma vez que explica as várias operações através de, essencialmente, operadores SQL de agregação, ela é, de uma forma geral, intuitiva. De uma forma simplificada, este modelo tem como base uma tabela de dados extraída de uma fonte, possivelmente uma tabela de factos. Com essa tabela, são geradas (pré-computadas) outras tabelas sumarizadas, que contêm os dados agregados sobre diferentes perspetivas: conjuntos diferentes de dimensões e possíveis hierarquias implícitas nessas dimensões.

3.1.1 O Espaço Multi-Dimensional (EMD)

O conjunto de todas as tabelas ou vistas resultantes das operações referidas anteriormente caracteriza o EMD, ou melhor: o Hipercubo. O Hipercubo, ou simplesmente Cubo, é então um operador com informação materializada suficiente para que seja possível "navegar" pelos diferentes níveis de agregação de forma ágil, sem esperas de cálculos de sumarização de grandes volumes de dados. Como esses diferentes níveis de agregação diferem apenas na granularidade da informação (nível de detalhe), então dentro de um cubo pode existir outro cubo, mas com menor detalhe de informação.

Tempo	Localização	Produto	Lucro por Tempo por Localização e por Produto	Lucro por Tempo e por Localização	Lucro por Tempo
1 de Janeiro	Braga	Boné	3		
		Brincos	5	8	
	Porto	Brincos	5		
		Colar	4	9	17
2 de Janeiro	Faro	Colar	4	4	4

Tabela 3.1: Lucro segundo diferentes combinações de dimensões

Os autores referem várias possibilidades de estruturas dos dados agregados e centram-se na representação dos cubos como tabelas estendidas com mais registos de dados agregados pré-processados, ou seja, tabelas que incluem os registos de todas as agregações possíveis. Inicialmente os autores sugerem que em vez de ser uma extensão horizontal, poderia ser

uma extensão vertical, com mais colunas, no entanto, como o cubo está fundamentado numa base relacional, não fazia sentido utilizar-se células das tabelas sem dados (Tabela 3.1) e portanto definiram um valor chamado '`ALL`' que representa uma agregação, tal como se pode ver pela Tabela 4.2.

Com esta disposição dos dados, a organização lógica fica imediatamente estruturada, mas não tão óbvia quanto o modelo do reticulado apresentado na Secção 3.2. Apesar dos autores não terem formalizado o cubo através de diferentes tabelas ou vistas, eles mostram essa possibilidade, mesmo que de forma virtual (Figura 4.3). Essas vistas são resultados de simples consultas explicadas na secção seguinte.

3.1.2 Operações

As operações apresentadas nesta formalização resultam praticamente nas possibilidades que os operadores `UNION` e `GROUP BY` permitem fazer. Os autores inicialmente apresentam a operação "*histogram*" que basicamente é uma função de categorização, para agregações adicionais.

Depois apresentam as duas operações mais importantes de navegação do OLAP: o *roll-up* e o *Drill-Down*. De uma forma simplificada, estes operadores são seleções filtradas das colunas do cubo, ou seja, através dos valores 'ALL', facilmente se seleciona apenas os dados que se quer analisar segundo um determinado nível de detalhe.

Os autores também apresentam a operação "*cross-tabulation*", que funciona apenas em espaços (vistas) com apenas duas dimensões. Esta operação serve para realizar sumarizações categorizadas dos dados, algo similar a uma operação de *Pivoting*.

3.1.3 Considerações finais

Uma vez que as bases de dados relacionais só têm um valor especial, o `NULL`, e não existe forma de acrescentar o `ALL` como valor especial, então os autores decidiram utilizar o valor `NULL` como representação do `ALL`. Esta decisão poderá acarretar problemas em situações em que os dados vindos das fontes trazem já valores `NULL`.

Na prática, os autores definem um cubo como uma materialização de uma tabela de factos "enriquecida" com dados agregados. Depois, virtualmente, cada vista resultante das operações é um ponto ou nível de agregação do EMD. De forma mais percetível, cada vista é então um cuboide do reticulado apresentado na secção a seguir, que resulta de um filtro à "tabela enriquecida" .

Esta decisão de incluir os dados desta forma pode gerar resultados bastante complexos com

problemas de escalonamento que não são a base de análise desta tese. No entanto, esses problemas podem ser atenuados com uma estruturação mais fracionada ao nível de tabelas, em que cada cuboide será então uma tabela.

3.2 Modelo do reticulado (Lattice)

Apesar do cubo de Gray et al. [1997] ter uma estrutura organizacional bastante concreta entre as várias vistas sumarizadas (agregadas), falta-lhe uma organização hierárquica global. Com base nesta premissa, foi conjeturada uma estrutura global, organizada hierarquicamente, do cubo por Harinarayan et al. [1996], chamada de reticulado ou *lattice*.

Esta estrutura organizada, com base em matemática de conjuntos, permite descobrir as dependências entre as várias tabelas sumarizadas e a própria tabela de factos. Tal como o cubo, este modelo de representação passou a ser aceite pela literatura uma vez que tem permitido realizar algoritmos de otimização de materialização parcial de tabelas sumarizadas em ROLAP. Para além disso, este modelo permite ter uma visão melhorada de certas operações OLAP, nomeadamente duas das mais importantes: o *roll-up* e o *Drill-Down* que basicamente são deslocações dentro do reticulado.

A base desta estrutura é o cubo de Gray et al. [1997] e, por isso mesmo, caracteriza-se por uma formalização também baseada em operações SQL. No entanto, as provas relativas às dependências entre tabelas são feitas com base em lógica de conjuntos ordenados, os reticulados.

3.2.1 O Espaço Multi-Dimensional (EMD)

Apesar de esta formalização basear-se no cubo, referido na secção anterior, existe um pormenor que foi assumido à partida pelos autores: o cubo está definido com diferentes tabelas de sumarização. Deste modo, em vez de o cubo ser uma tabela que engloba as várias sumarizações, tal como é demonstrado na Tabela 4.2, o cubo é constituído por várias tabelas, cada uma definindo um nível de agregação. Esta decisão, logo à partida, tem a vantagem de não ter tantas limitações ao nível de escalabilidade.

Portanto, um EMD, segundo Harinarayan et al. [1996], é um conjunto de tabelas em que cada uma apresenta um nível de agregação por cada dimensão (e por cada hierarquia de cada dimensão) e que estão todas relacionadas umas com as outras numa hierarquia principal. Esse conjunto organizado de tabelas chama-se reticulado (Figura 4.4b), devido exatamente ao nome dado na matemática por este tipo de conjuntos. A relação hierárquica entre as tabelas define níveis de agregação organizados para que as operações de agregação dos dados passem a ser meras navegações entre tabelas.

A hierarquia resulta de todas as combinações possíveis entre as várias dimensões (eixos de análise) de uma tabela de factos. Para além disso, caso as dimensões tenham alguma hierarquia definida internamente, então o reticulado aumenta de complexidade acrescentando mais essas combinações possíveis. Cada combinação de dimensões (e atributos com hierarquias) é um ponto do reticulado e é denominado de cuboide.

A construção de um reticulado, em OLAP, é muito simples e começa pela identificação das dimensões da tabela de factos. De seguida, para definir um novo nível hierárquico, faz-se as combinações possíveis, entre as várias dimensões (combinações de $n-1$, em que n é o número de dimensões). Daí resultam dependências entre a tabela de factos original e as combinações novas. Essas dependências são facilmente identificáveis e só existem entre diferentes níveis. As dependências só existem se as dimensões pertencerem a ambas as combinações. O procedimento ocorre assim sucessivamente até $n = 0$, em que neste caso a combinação é de zero dimensões, que representa o '`ALL`' referido no cubo de Gray et al. [1997], que é a agregação de todas as dimensões.

A notação definida para o reticulado é a mesma utilizada na matemática, ou seja há um conjunto de elementos (vistas ou agregações) L e uma relação de dependências: $\langle L, \preccurlyeq \rangle$. Assim, para os elementos a, b de um reticulado, b é antecessor de a se e só se $a \preccurlyeq b$.

Considerando esta notação e tendo em conta todas as consultas *ad-hoc* que um agente de decisão pode realizar, é possível que se tenha vistas agregadas por atributos das dimensões. Estes casos denominam-se hierarquias de atributos, uma vez que se agrega a informação não por uma dimensão inteira mas apenas por um dos atributos dela.

Para estes casos a notação é aplicada da mesma forma, por exemplo uma vista de uma dimensão a e uma vista de uma agregação por um atributo x, denominada aX, então aX é antecessor de a, ou seja: $a \preccurlyeq aX$. Uma vez que uma dimensão pode ter vários atributos que se agregam de forma dependente, então pode-se ter situações como este caso da dimensão tempo: ALL$\preccurlyeq$tempomês$\preccurlyeq$tempodia. Estas hierarquias podem levar a casos ainda mais complexos em que existe ramificações de dependências, por exemplo na dimensão tempo pode-se ter mais um atributo que define as semanas do ano (ALL$\preccurlyeq$temposemana$\preccurlyeq$tempodia) que não tem qualquer relação de dependência com o atributo que representa os meses do ano, surgindo então a notação: tempomês $\not\preccurlyeq$ temposemana. Estas hierarquias ramificadas dentro das dimensões estão representadas na Figura 4.5.

3.2.2 Operações

O ponto forte desta formalização é a possibilidade de criar uma estrutura organizada capaz de otimizar a performance de um sistema OLAP relativamente a duas operações muito importantes: o *roll-up* e o *Drill-Down*. Portanto, em termos de operações, só são abordadas estas duas.

Depois de ter um EMD organizado pelo reticulado com as várias tabelas de sumarização relacionadas entre elas, a possibilidade de navegação fica bastante simplificada, uma vez que uma operação de *roll-up*, ou seja, de agregação de dados (diminuição do nível de detalhe), passa a ser uma simples troca de tabela, em vez de uma operação complexa de agregação de dados.

Exemplificando, se o agente de decisão estiver a analisar a tabela de factos do DM da Figura 4.1, então ele está a analisar os dados do cuboide "TLP" (acrónimo dos nomes das 3 dimensões: Tempo, Localização, Produto). Caso ele precise analisar os dados de forma mais agregada, por exemplo omitindo os dados temporais, então ele faz uma operação de *roll-up* que na prática é uma navegação para o cuboide "LP" (acrónimo que representa uma vista com as dimensões "Localização" e "Produto").

No caso de o Agente de Decisão decidir querer visualizar os dados mais detalhados, acrescentando mais informação para análise, então ele estará a adicionar uma dimensão. Essa adição de uma dimensão é a operação inversa chamada *Drill-Down*. Sem uma materialização prévia, o custo desta operação seria bastante penoso, no sentido de que seria necessário expandir os dados para acrescentar mais informação que seria proveniente da tabela de factos.

No entanto, usando novamente o conceito de reticulado e sabendo que existe uma materialização desses dados pedidos, na prática o que ocorre é uma navegação para uma nova tabela em vez de processar a adição de informação de uma nova dimensão. Esta forma de executar as operações tem apenas uma desvantagem: a necessidade de haver já dados pré-computados. Aparte disso, a eficácia de realização deste tipo de operações é muito superior a uma implementação como a da Tabela 4.2.

3.2.3 Considerações Finais

O reticulado [Harinarayan et al., 1996] é uma peça fundamental na estruturação de um EMD otimizado. Aliás, o artigo deste modelo está no grupo dos 600 artigos mais referenciados na área das Ciências da Computação. Alguns dos modelos apresentados a seguir não utilizam esta estrutura diretamente, no entanto isso ocorre provavelmente porque esses modelos foram desenvolvidos na mesma altura, sem conhecimento prévio desta estrutura organizada.

De salientar que este não é um modelo equiparável ao apresentado nesta dissertação, uma vez que os objetivos são bem diferentes. Este modelo não tem um objetivo claro de definir formalmente um sistema OLAP, mas, sim, apresentar uma estrutura organizada que na prática, aquando da implementação, tem um potencial enorme. Por esse motivo, no modelo só são apresentadas duas operações comuns do OLAP, que permitem a navegação entre cuboides.

Para além disso, este modelo apesar de fundamentado numa estrutura algébrica bastante estudada, o reticulado, apenas demonstra a parte final do OLAP, a organização das várias materializações, não considerando as bases como tabelas de factos e outros elementos de um DM.

Como este modelo não contém a definição base da estrutura dos dados, não tem uma formalização mais detalhada do reticulado. É por esse motivo que os modelos apresentados nas secções a seguir são importantes para uma formalização do OLAP mais completa.

3.3 Modelo de Gyssens e Lakshmanan

O modelo de Gyssens e Lakshmanan [1997] foi a primeira formalização matemática do OLAP. Esta formalização contempla não só a definição do EMD ao nível da meta-informação, como também inclui conceitos de instância (algo equiparável aos cuboides do reticulado) e as operações mais importantes, segundo os autores, com base em Álgebra Relacional.

Neste modelo, um EMD é um DM, o que significa que se se quiser definir algo mais complexo é necessário ter várias aplicações deste modelo para juntar os vários DMs que possam existir dentro de um EMD.

Esta formalização começa por definir a estrutura espacial ao nível do conjunto das dimensões. Depois, com recurso ao conceito de instâncias interliga as várias dimensões (constrói relacionamentos com pesos) produzindo então as várias materializações de um EMD. Por conseguinte, este modelo perde o conceito de tabela de factos uma vez que a define como uma materialização comum, que neste caso é a mais detalhada.

3.3.1 O Espaço Multi-Dimensional (EMD)

Apesar de este modelo não funcionar da maneira mais usual na construção de DW, por causa da situação da tabela de factos, a sua estrutura conceptual faz bastante sentido. O modelo começa por definir apenas as dimensões e os vários atributos que as compõem, chamando a isso a *Table Schema*. Ou seja, define um esquema de um DM sem definir uma tabela de factos.

Assim, o modelo de Gyssens e Lakshmanan [1997] resume um DM, n-dimensional, como sendo uma *Table Schema* que é um tripleto $\langle D, R, par \rangle$ definido da seguinte maneira:

- $D = \{d_1, \ldots, d_n\}$ é um conjunto de nomes de dimensões;
- $R = \{A_1, \ldots, A_m\}$ é um conjunto de nomes de atributos;

- $par : D \rightarrow 2^{\{A_1,...,A_m\}}$ tal que:
  1. para todo o $i, j = 1, \ldots, n, i \neq j, par(d_i) \cap par(d_j) = \emptyset$;
  2. $\cup_{d \in D} par(d) \subseteq R$.

A definição da tabela de factos ocorre como consequência do cruzamento (relacionamento) entre todas as dimensões desse *Table Schema*. Os atributos desse relacionamento constituem as medidas da tabela de factos. Ou seja, não existe o conceito de tabela de factos, no entanto a forma como ela é expressada e tendo em conta a forma como o OLAP funciona, com as inúmeras tabelas materializadas, faz todo o sentido este modelo.

Relativamente aos dados agregados (cuboides na nomenclatura do reticulado), eles são resultado também de combinações entre dimensões mas com diferentes dimensionalidades. Todas estas combinações neste modelo são chamadas de Instâncias. Expressando de maneira mais formal, considerando o conjunto das medidas $M = R - \cup_{1 \leq i \leq n} par(d_i)$, então uma instância de uma *Table Schema* n-dimensional $\langle D, R, par \rangle$ é um conjunto finito de $n + 1$ relacionamentos da forma $rd_1(Tid, X_1), \ldots, rd_n(Tid, X_n), rm(rd_1.Tid, \ldots, rd_n.Tid, M)$, tal que:

1. a junção $\pi_{Tid}(rd_1) \times ... \times \pi_{Tid}(rd_n)$ é equivalente a $\pi_{rd_1.Tid,...,rd_n.Tid}(rm)$, por exemplo, para cada combinação dos valores Tid nos relacionamentos $rd_1, ..., rd_n$, existe pelo menos um tuplo correspondente em rm, e cada tuplo em rm corresponde a uma combinação de valores Tid nos relacionamentos $rd_1, ..., rd_n$;
2. para todos os $i = 1, ..., n$, Tid é chave do relacionamento rd_i;
3. para todos os $i, j = 1, ..., n, i \neq j, \pi_{Tid}(rd_i) \cap \pi_{Tid}(rd_j) = \emptyset$, por exemplo, os valores Tid em diferentes relacionamentos rd_i e rd_j são disjuntos.

Por fim, para definir o conjunto de instâncias de um EMD tem-se o seguinte:

- $S = \langle D, R, par \rangle$ uma *Table Schema*;
- r um relacionamento, $\overline{r}_s$ espaço completo em relação a S;
- $\mathcal{R}(R)$ a classe de todas as relações sobre R; e
- $\mathcal{T}(S)$ a classe de todas as instâncias;

3.3.2 Operações

As operações apresentadas neste estudo têm o objetivo de se aproximarem ao máximo à Álgebra Relacional. Para que isso seja possível, os autores deste modelo definem um

conjunto de pressupostos que permitem correlacionar a álgebra com as tabelas. Assim, considerando $S = \langle D, R, par \rangle$ uma *Table Schema*, então se τ for um instância do esquema S, $f(\tau)$ denota uma representação relacional e $rep(\tau)$ como uma relação do esquema R. Reciprocamente, se r for uma relação do esquema R, denota-se como uma representação tabular $g(r)$ e uma instância da tabela com esquema s como $tab_s(r)$.

Relativamente às operações, os autores começam por definir as mais elementares, nomeadamente os operadores unários, seleção (σ), projeção (π) e renomear ($\rho_{B \leftarrow A}$). Estes operadores são agrupados na definição **op** como sendo:

$$\mathbf{op}(\tau) = tab_s(\mathbf{op}(rep(\tau)) \tag{3.1}$$

Quanto à união, intersecção e diferença, considerando τ_1 e τ_2 instâncias de uma tabela, ambas com um esquema S e seja **op** uma união ($\cup$) ou uma intersecção ($\cap$) ou uma diferença ($\setminus$), então define-se da seguinte maneira:

$$\tau_1 \ \mathbf{op} \ \tau_2 = tab_s(rep(\tau_1) \ \mathbf{op} \ rep(\tau_2)) \tag{3.2}$$

Para se poder "produzir" tabelas, os autores definem duas funções que permitem remover e adicionar dimensões. De uma forma genérica podia-se dizer que este procedimento seria um *roll-up* e *Drill-Down*, respetivamente. Não o é, pois não são consideradas funções de agregação. Portanto estas funções apresentadas abaixo, são uma espécie de operadores de reestruturação e são essencialmente para produzir operações de *pivoting*.

Essas funções são o *fold* e o *unfold* são definidas da seguinte maneira (e demonstradas com um exemplo na página 4.4):

- **Fold:** Considere-se τ uma tabela com esquema $S = \langle D, R, par \rangle$ e d uma das dimensões de D. Define-se $\mathbf{fold}^d(\tau)$ como uma tabela com esquema $S' = \langle D \setminus \{d\}, R, par' \rangle$, em que para todo o d_i tem-se $D \setminus \{d\}, par'(d_i) = par(d_i)$ e instância $tab_{S'}(rep(\tau))$.
- **Unfold:** Conisderando novamente τ como uma tabela com esquema $S = \langle D, R, par \rangle$, d um novo nome de $\mathcal{N}$ que não aparece em τ e $X \subseteq M$ um conjunto de medidas (atributos). Define-se $\mathbf{unfold}^d_X(\tau)$ como uma tabela $S' = \langle D \cup \{d\}, R, par' \rangle$, em que para todo o d_i tem-se D, $par'(d_i) = par(d_i)$ e $par'(d) = X$, e com instância $tab_{S'}(rep(\tau))$.

Por fim, relativamente à agregação dos dados, os autores deste artigo definem uma função genérica, mas agregativa, em conjunto com uma função de classificação para construir uma relação (tabela).

Seguindo o exemplo genérico do cubo que foi apresentado no artigo, considere-se uma tabela τ com um esquema $S = \langle \{D_1, ..., D_{m-1}\}, \{A_1, ..., A_m\}, par \rangle$, com $i = 1, ..., m -$

$1, par(d_i) = \{A_i\}$, ou seja, existe uma medida tal como referido na página 26. Assim, a seguinte formalização algébrica define um cubo para a tabela τ da seguinte maneira:

1. Considere-se a instância All sobre S, em que a representação consiste num só m-tuplo $(All, ..., All, \perp)$. Não existe mais nenhuma instância na base de dados igual a esta.
2. Formação da tabela $All \cup \tau$.
3. Considerando $X = \{A_1, ..., A_{m-1}\}$ e r uma relação com o esquema R, então define-se uma função de classificação sobre X, para relações R:

$$f(r, a_1, ..., a_{m-1}) = \{(b_1, ..., b_{m-1}) | \\ \exists b_m : (b_1, ..., b_{m-1}) \in r \wedge \\ \forall i = 1, ..., m-1 : (b_i = a_i) \vee (a_i = All)\} \ .$$

   Através desta função, forma-se uma tabela classificada como $K(f, \tau)$. Neste ponto, para todo $i = 1, ..., m-1$, a dimensão d_i tem dois parâmetros, A_i e $f.A_i$. Através das funções **unfold** e **fold**, tira-se os parâmetros $A_1, ..., A_{m_1}$ para a medida.

4. Define-se a função de sumarização

$$g_{A_m \leftarrow A_m} : 2^{dom(A_1) \times ... \times dom(A_m)} \rightarrow dom(A_m)$$

   com $g(S) = \sum_{(b_1,...,b_m) \in S, b_m \neq \perp} b_m$. Através disto, forma-se uma tabela agregada com $A(K(f, \tau), g)$, com esquema S e uma representação relacional:

$$\{(a_1, ..., a_{m-1}, a_m) | \\ a_m = g(\{(b_1, ..., b_m) | (a_1, ..., a_{m-1}, b_1, ..., b_m) \in K(f, \tau)\})\} \ .$$

5. Renomear os atributos $f.A_i$ para A_i.

Tabela		Dimensão2					
		A2	**a21**	**a22**	...	**a2q**	All
Dimensão1	**A2**	*M*					
	a11		v11	v12	...	v1q	⊥
	a12		v21	v22	...	v2q	⊥
	⋮		⋮	⋮		⋮	⋮
	a1p		vp1	vp2	...	vpq	⊥
	All		⊥	⊥	...	⊥	⊥

Tabela		Dimensão2					
		A2	**a21**	**a22**	...	**a2q**	All
Dimensão1	**A2**	*M*					
	a11		v11	v12	...	v1q	$\sum_j$ v1j
	a12		v21	v22	...	v2q	$\sum_j$ v2j
	⋮		⋮	⋮		⋮	⋮
	a1p		vp1	vp2	...	vpq	$\sum_j$ vpj
	All		$\sum_i$ vi1	$\sum_i$ vi2	...	$\sum_i$ viq	$\sum_{ij}$ vij

Figura 3.1: Ilustração de um cubo de Gyssens e Lakshmanan [1997, p.113]

Este cubo, da Figura 3.1, não passa de uma tabela (instância) construída com base na álgebra previamente definida sobre uma *Table Schema*. No entanto, num contexto real um cubo é bastante mais que isso uma vez que contém várias instâncias que constituem um esquema organizado, normalmente em reticulado.

3.3.3 Considerações Finais

O modelo Gyssens e Lakshmanan [1997]apresenta um conceito diferente do que é comum numa situação real em OLAP, no entanto, algumas destas diferenças ao nível conceptual fazem bastante sentido. O facto de não haver uma definição explícita da estrutura da tabela de factos complica a definição de DWs, no entanto, permite visualizar o que de facto acontece num sistema OLAP, que tem dados factuais de diferentes granularidades conforme os diferentes cruzamentos (relacionamentos) entre dimensões.

Na verdade, cada uma das materializações poderia ser uma tabela de factos caso os programadores definissem que era aquela granularidade dos dados. Portanto, esta noção de não existir uma tabela de factos mas sim cruzamentos de dimensões que geram a informação dos acontecimentos, faz sentido. Claro que ao nível prático os dados provavelmente vêm com uma granularidade aproximada à comum tabela de factos e portanto é necessário ter cautela numa abordagem conceptual como esta.

Apesar de não considerar a estrutura organizada do reticulado, provavelmente porque os modelos foram feitos na mesma altura, a verdade é que se se juntarmos o conceito de reticulado com este conceito de cruzamento com pesos (medidas) entre dimensões chegamos a um conceito bastante interessante e plausível ao vermos na prática o que acontece num reticulado, tal como se pode ver pela Figura 2.4.

Porém, este modelo carece de vários problemas que não foram abrangidos. O facto de apenas considerar dimensões e instâncias não permite, pelo menos de forma imediata, definir certos aspetos da estrutura de um DM, nomeadamente tabelas ponte. Para além disso, não existe forma de caracterizar os atributos, segundo o tipo de agregação e sobre possíveis hierarquias entre atributos que produzem cubos com estruturas completamente diferentes.

3.4 Modelo de Golfarelli et al.

Este modelo é chamado de DFM e tem o objetivo de formalizar graficamente e conceptualmente um DW e tudo o que isso implica, nomeadamente as instâncias. Na formalização do modelo, não é abordada a materialização em reticulado, no entanto, são aceites instâncias materializadas, ou seja com cálculos agregados prontos a responder rapidamente ao Agente de Decisão. Na prática, estas instâncias são os cuboides do reticulado, no entanto, sem a organização de dependências da mesma.

Toda a formalização é detalhada a muitos dos aspetos de implementação de um DW, no entanto, a formalização é sempre conceptual, distinguindo-se do nível lógico, para que o modelo seja compatível com as várias formas de OLAP (MOLAP, ROLAP e HOLAP).

O objetivo principal desta formalização, tal como na maioria dos casos, é tentar finalmente construir um modelo genérico para esta área, tal como ocorre com a Álgebra Relacional. Depois, através desse modelo seria possível desenhar DW de forma conceptual e gráfica e por conseguinte transformar automaticamente o resultado em algo lógico.

O modelo dá uma importância enorme aos problemas práticos mais comuns num sistema OLAP: os tipos de agregações, associações entre agregações e limitações das mesmas. Isto ocorre porque na prática para um Agente de Decisão, um sistema OLAP é quase como uma base de dados com o objetivo de permitir agregar dados de forma eficaz sobre grandes volumes de dados.

3.4.1 O Espaço Multi-Dimensional (EMD)

O DFM representa um EMD, que é formalizado através de um esquema dimensional constituído, na prática, por três tipos de elementos: os factos, as dimensões e as hierarquias das dimensões. Tudo combinado, permite construir um DW e até possíveis instâncias.

Esta construção inicialmente é feita de forma conceptual, no sentido de que é formalizada toda a meta-informação do possível sistema OLAP. Depois, através dessa construção, é possível construir instâncias que são conjuntos de m-tuplos que representam os factos. Essas instâncias são todas resultado de operações.

O elemento mais importante deste modelo, que define o EMD, chama-se *Fact Scheme* (FS) e é um 6-tuplo definido pela Expressão 3.3. Cada elemento deste 6-tuplo tem um objetivo simples de definir concretamente os três aspetos referidos cima: os factos são resultado da combinação entre métricas e atributos; os atributos pertencem às dimensões que são definidas através de hierarquias de atributos; e existe uma relação entre métricas e atributos que definem as funções de agregação possíveis.

$$F = (M, A, N, R, O, S) \tag{3.3}$$

Estes elementos são definidos da seguinte maneira:

- M é o conjunto das métricas. Cada medida $m_i \in M$ é definida por uma expressão numérica ou booleana.
- A é o conjunto dos atributos das dimensões. Cada atributo $a_i \in A$ é caracterizado por valores discretos, pertencentes ao domínio: $Dom(a_i)$.
- N é o conjunto de atributos que também são das dimensões, mas que não pertencem às hierarquias. São apenas atributos descritivos, que acrescentam informação em vez

de a categorizar como ocorre nos atributos de A (por exemplo, os atributos "descrição" e "morada" da Figura 3.2).

- R é um conjunto de pares que conseguem definir as hierarquias das dimensões, chamadas *quasi-tree* que genericamente são grafos de atributos. Cada elemento de R tem a forma de (a_i, a_j) em que $a_i \in A \cup a_0$ e $a_j \in A \cup N(a_i \neq a_j)$, ou seja, é um grafo com (não acíclico) com raiz em a_o. Cada par de R define um relacionamento de N-para-1 entre o primeiro e o segundo elemento.

- O é o conjunto de relacionamentos de R que são opcionais. Este conjunto está portanto contido em R, ou seja $O \subset R$. Este conjunto de relacionamentos são necessários para determinados DWs que contêm atributos de definem apenas uma parte dos valores das dimensões. Por exemplo, um atributo que defina o valor calórico numa dimensão de produtos genéricos é apenas aplicável a produtos alimentares (por exemplo o nó "Calorias" na Figura 3.2). Ou seja, O tem a seguinte característica: $\exists(a_i, a_j) \in O$ em que a_j é nulo, que na prática quer dizer que há valores nulos de a_j para determinados atributos de a_i.

- S é o conjunto de expressões que definem as agregações das métricas. Cada elemento de S indica que conjunto de funções uma métrica pode ser agregável para um determinado atributo das hierarquias das dimensões. Deste modo, cada elemento de S é um 3-tuplo da forma (m_j, d_i, Ω), em que $m_j \in M$, $d_i \in Dim(f)$ e $\Omega \in \{'SUM', 'AVG', 'COUNT', 'MIN', 'MAX', ...\}$, ou seja, a métrica definida pode ser agregada segundo o atributo da dimensão especificada através da operação Ω.

De salientar apenas que quando é referida a variável d_i está-se a referir geralmente ao atributo raiz da dimensão, ou seja, na prática, está-se a referir à dimensão. Por isso, temos que $Dim(f) = \{a_i \in A | \exists(a_0, a_i) \in R\}$ e por conseguinte: $d_i \in Dim(f)$

Através desta especificação tem-se controlo para definir praticamente todos os pormenores de um DW. Não só as métricas e atributos são definidos, como a própria estrutura das dimensões, os atributos que as acompanham e a relação que esses atributos têm com as métricas quando são executadas operações de *roll-up*, ou seja, quando um determinado atributo é removido para um atributo mais sumarizado.

Uma vez especificado o DM através desta metodologia, todos os pormenores que não estiverem definidos não são regras aplicáveis. Quer isto dizer que se não houver por exemplo elementos de S para todas as métricas e/ou atributos das dimensões, então não existe possibilidade de realizar *Roll-ups*, ou melhor, ao realizar-se tal operação a métrica é removida pois não é agregável.

Esta formalização de conjuntos matemáticos tem um objetivo muito concreto: poder construir modelos gráficos conceptuais exatos, mas com bases sólidas de matemática para possibilitar depois a transição para o modelo lógico do sistema OLAP. Na especificação original,

são incluídas também informações que possibilitam a construção deste modelo ao nível gráfico. Um exemplo desse modelo, de forma gráfica está representado na Figura 3.2.

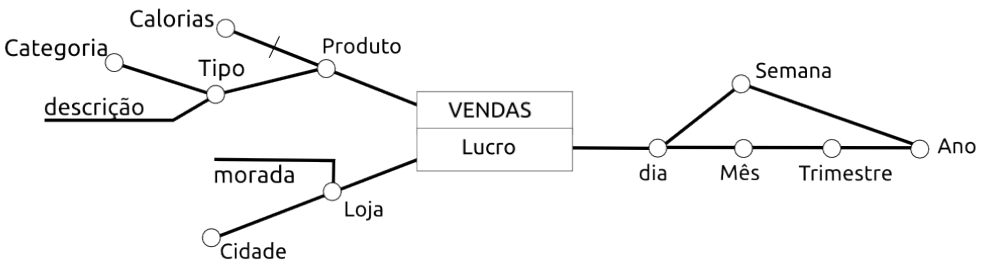


Figura 3.2: Exemplo de modelo conceptual DFM

Posto isto, depois de a estrutura conceptual estar construída, é possível construir instâncias. Para Golfarelli et al. [1998] uma instância é a estrutura de dados onde de facto os dados existem. Tal como ocorre nos modelos apresentados a seguir, uma instância é a representação dos factos num determinado nível de agregação conforme as dimensões usadas.

Portanto, ao utilizar-se todas as dimensões (com o atributo raiz) tem-se a tabela de factos, pois é a instância com maior detalhe possível desse DM. Todas as outras instâncias que tenham outros conjuntos de dimensões (ou outros atributos de dimensões) são apenas instâncias mais sumarizadas da tabela de factos, algo similar aos cuboides do modelo de Harinarayan et al. [1996].

Tal como num DW comum, a tabela de factos é também um ponto-chave deste modelo. Nesta especificação, a tabela de factos é o conjunto de "*primary fact instance*", ou seja, é como um conjunto base instâncias unitárias (factos) pela qual todas as outras vão nascer. De facto é mesmo isso que acontece na prática, visto as instâncias são tabelas com valores calculados da tabela de factos.

Então sendo cada facto um conjunto de atributos com um determinado conjunto de métricas, que valorizam os factos, o mesmo acontece neste modelo. Uma "*primary fact instance*", é representada por pf tem a seguinte definição: $pf(\alpha_1, \ldots, \alpha_2)$ em que $(\alpha_1, \ldots, \alpha_2) \in Dom(d_1) \times \cdots \times Dom(d_n)$.

Visto que os sistemas OLAP são construídos para análises analíticas centradas acima de tudo em dados agregados, obviamente que é necessário tornar toda esta complexidade em algo mais agrevável. É necessário portanto as instâncias de dados mais agregados. Para tal é necesário, apenas para controlo, considerar uma variável P que define o padrão de agregação, ou seja, o estado atual de uma determinada instância. Esta variável é o conjunto de atributos que definem as métricas para uma determinada instância, por exemplo, no

caso da Figura 3.2, se se quiser definir a tabela de factos, tem-se que P é o conjunto $\{Loja, Dia, Produto\}$.

No caso de um *roll-up* para que suba toda a hierarquia até ela desaparecer, ou seja, no caso de uma remoção de uma das dimensões, então não existe nenhum atributo dessa dimensão em P. Portanto, para que uma determinada instância efetivamente existe, é necessário garantir que P é aplicável segundo as regras definidas anteriormente, que resumidamente chega-se à Expressão 3.4.

$$\forall d_k \in P \ \ \exists (m_j, d_k, \Omega) \in S \tag{3.4}$$

Então, considerando instância que seja uma agregação da tabela de factos (ou pf) que tenha um padrão de agregação $P = \{a_1, \ldots, a_v\}$, um d_h* denotando uma dimensão que tem uma hierarquia que inclui $a_h \in P$, então ela é definida da seguinte maneira:

$$\{pf(\alpha_1, \ldots, \alpha_n) | \forall k \in \{1, \ldots, n\} \alpha_k \in Dom(dk) \wedge \forall h \in \{1, \ldots, v\} \alpha_h * .a_h = \beta_h\} \tag{3.5}$$

Sendo portanto a combinação de valores $(\beta_1, \ldots, \beta_v) \in Dom(a_1) \times, \ldots, \times Dom(a_v)$. Estes valores são a agregação proveniente da tabela de factos, calculados para os atributos definidos.

Estes valores podem ou não ditar métricas. Caso nenhuma métrica satisfaça todos os 3-tuplos de S para todos estes atributos de P, então esta instância não tem métricas associadas. Portanto uma instância para ter métricas tem de garantir que cada métrica tem uma operação Ω para cada atributo. Os autores deste modelo ainda estudam a ordem das operações, principalmente quando existe operações diferentes numa agregação.

3.4.2 Operações

Esta especificação centra-se acima de tudo na construção do EMD e da forma como ele permite criar instâncias que são resultado de operações de consulta. Quer isto dizer que a não ser operações de consulta (que podem ser associadas ao *Slice&Dice*) não existe mais nenhuma especificação de operações, nomeadamente o funcionamento das operações clássicas de *roll-up* e *Drill-Down*. Apesar disso, é possível tomar algumas conclusões para cada uma dessas operações.

Genericamente, e de uma forma mais simplista uma operação de consulta é uma expressão que cria uma instância da seguinte maneira:

<expressão de instância> ::= <nome do facto>(<padrão de agregação>;<operação de seleção>)
<padrão de agregação> ::= lista separada por vírgulas de <elementos de padrão>

<elementos de padrão> ::= <nome da dimensão>/<nome da dimensão>.<nome do atributo>
<operação de seleção> ::= lista separa por vírgulas de <predicados>

Ou seja, uma consulta é construída sob a forma:

$$f(d_1, \ldots, d_p, a_{p+1}, \ldots, a_v; e_1(b_{i_1}), \ldots, e_h(b_{i_h})) \tag{3.6}$$

Em que os primeiros p elementos pertencem ao padrão de agregação envolvendo dimensões e os restantes $v - p$ são atributos de dimensão. Cada predicado $e_j (j = 1, \ldots, h, h \geq 0)$ recebe um atributo b_{i_j} pertencente a uma hierarquia com raiz na dimensão d_{i_j}.

Daqui conclui-se rapidamente que a tabela de factos é definida quando $p = v$ em que os atributos são precisamente as raizes das dimensões.

Tendo em conta esta definição completamente genérica de consultas, é possível tirar algumas conclusões sobre as operações mais importantes do OLAP. Relativamente ao *Slice&Dice*, este é precisamente uma operação de consulta explicada anteriormente. O Agente de Decisão, relativamente à projeção escolhe as dimensões (não precisa de escolher todas) e escolhe os atributos das hierarquias para definir o nível de agregação. Quanto à seleção, utiliza os predicados para limitar os dados. Na Secção 4.5 são demonstrados exemplos deste tipo de consultas.

Relativamente ao *roll-up*, esta operação resume-se a retirar uma das dimensões de uma expressão de uma instância, ou então a substituir um atributo da hierarquia por algum atributo superior. Relativamente à operação inversa, o *Drill-Down*, a situação é também o inverso. Se precisar de adicionar uma dimensão, acrescenta a dimensão e o atributo que quer acrescentar maior detalhe, ou então simplesmente desce o atributo pela hierarquia da dimensão.

Quanto à operação de *Pivoting*, esta também é possível bastando para isso trocar a ordem dos atributos da expressão da instância a ser calculada.

3.5 Modelo de Thomas e Datta

Uma vez que a área de OLAP estava (e está) com um potencial enorme ao nível empresarial, quando Gray et al. apresetaram o operador "Cubo", a comunidade efetivamente achou-o como um conceito importante para os EMDs, no entanto, ele tinha a desvantagem de estar modelado como uma extensão ao SQL comum.

Exatamente pela falta de formalização do Cubo, Thomas e Datta [2001] apresentam uma redefinição do conceito de Cubo, mas através de uma formalização algébrica simples, capaz de realizar as operações mais comuns do OLAP. Este modelo teve como base não só o

conceito de cubo, mas também o modelo de Gyssens e Lakshmanan [1997], referido na secção anterior, e no trabalho de Agarwal et al. [1996] que apresenta algorítmos otimizados para a agregação de dimensões.

Ainda assim, os autores referem que estes trabalhos têm problemas por "apresentarem restrições irreais", nomeadamente limitações ao nível do número de atributos por dimensão ou número total de medidas representáveis no cubo. Este modelo tenta contraria esses pormenores ao ser mais flexível.

3.5.1 O Espaço Multi-Dimensional (EMD)

Neste modelo o cubo é a construção fundamental do EMD, e na prática é isso que acontece de facto, e serve como *input* para os operadores referidos na secção a seguir. De uma forma similar à definição da Secção 3.3, os autores definem o cubo como sendo um 4-tuplo que difere essencialmente no facto de ter as métricas como um conjunto separado de informação. Isto ocorre porque para Thomas e Datta [2001] uma instância é definida de forma semelhante e para produzir a operação *Pivoting* a utilização de uma exclusão (ver definição de M da página 26) tornaria o processo mais complexo.

Assim, um cubo é definido como um 4-tuplo, $\langle D, M, A, f \rangle$, em que os quatro componentes são definidos da seguinte maneira:

- $D = \{d_1, d_2 \ldots, d_n\}$ é um conjunto n dimensões, em que d_i é o nome da dimensão extraída do domínio $dom_{dim(i)}$.
- Um conjunto k de medidas $M = \{m_1, m_2, \ldots, m_k\}$ em que m_i é o nome da medida, extraído do domínio $dom_{measure(i)}$.
- Os conjuntos de nomes das dimensões e dos nomes das medidas são disjuntos: $D \cap M = \emptyset$.
- Um conjunto t de atributos $A = \{a_1, a_2, \ldots, a_t\}$ em que a_i é o nome do atributo, extraído do domínio $dom_{attr(i)}$.
- Um mapeamento de um para muitos $f : D \rightarrow A$, ou seja, existe um conjunto de atributos para cada dimensão. O mapeamento é tal que os conjuntos de atributos são disjuntos: $\forall i, j, i \neq j, f(d_i) \cap f(d_j) = \emptyset$.

De salientar que um cubo definido desta forma não é mais que uma estrutura abstrata. Para ter uma estrutura materializada do cubo, é necessário atribuir valores às várias dimensões. Para uma melhor compreensão, convém verificar o exemplo da página 4.6.

Então para se ter uma materialização de um cubo, também designado como uma instância do cubo, define-se um 6-tuplo da seguinte maneira:

$$\langle D, M, A, f, V, g \rangle \tag{3.7}$$

Os elementos D, M, A e f são herdados do seu "Cubo" original, enquanto que V representa o conjunto de valores que foram usados para materializar o cubo. Note-se que cada elemento $v_i \in V$ é um k-tuplo $\langle \mu_1, \mu_2 \ldots, \mu_k \rangle$, em que cada μ_i é uma instanciação da medida m_i.Finalmente, g representa um mapeamento $g : dom_{dim(1)} \times dom_{dim(2)} \times \cdots \times dom_{dim(n)} \rightarrow V$. Assim, g indica que valores estão associados a cada "célula" da instância do cubo, ou seja, duas instâncias do cubo que correspondam ao mesmo cubo, apenas diferem no 2-tuplo $\langle V, g \rangle$

3.5.2 Operações

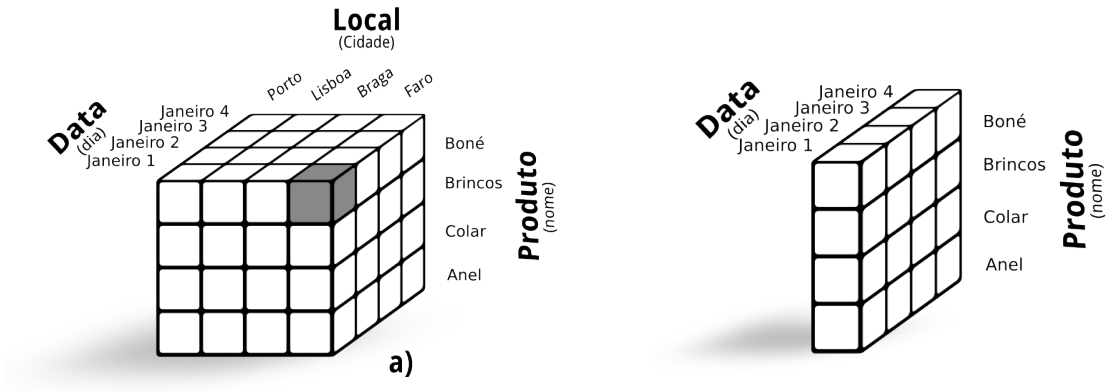
Os autores deste modelo começam por definir o que é um DW comum (uma tabela de factos com relacionamentos para as dimensões) e depois as várias operações existentes nos EMDs. As operações que foram consideradas foram o *slice, dice, drill-down, roll-up* e *pivot*.

Inicialmente identificam o *slice* e só depois o *dice* como operações distintas, mas acabam por referir que a operação *Slice&Dice* têm um efeito de redução da dimensionalidade do cubo. Relativamente ao *slice* fazem uma conotação à Seleção (σ) da álgebra relacional. Assim, uma execução desta operação significa escolher as dimensões de uma vista. Quanto à operação *dice* esta está relacionada com o operador Projeção (π) da álgebra relacional, ou seja, permite escolher efetivamente as "posições" ou valores da dimensão, por exemplo escolher a cidade do "Faro", com a data "1 de Janeiro" e o produto "Boné" do cubo da Figura 3.3a dentro das respetivas dimensões.

Duas das operações mais importantes dos EMDs são o *roll-up* e o *Drill-Down*. Estas operações são também abordadas pelos autores e o efeito destas operações, tal como referido na secção 3.2, não é mais que uma movimentação no reticulado, em termos de agregação de dados. Ou seja, esta operação para ser realizada precisa de ter um espaço inicial e o resultado dessa operação é um outro espaço.

Para explicar sob a forma de exemplo, partindo do princípio que o agente de decisão está a ler os dados detalhados do cubo da Figura 3.3a, se ele agora quiser ver os dados sem informações do local das vendas, ou seja se quiser ver os dados agregados, ele faz um *roll-up* agrupando os dados que estavam distribuídos pelas várias cidades, resultando num espaço representado na Figura 3.3b. A operação inversa, inclusão de uma dimensão (ou hierarquia) denomina-se *Drill-Down*.

Por fim, os autores definem a operação *Pivoting* como sendo a agregação sobre um ou mais



(a) Zona cinza é um resultado de uma operação *Slice&Dice* num cubo de dados

(b) Resultado de um *roll-up* relativamente ao cubo da Figura 3.3a

Figura 3.3: Operações OLAP

dimensões, produzindo uma nova instância do cubo, com um atributo por cada dimensão e um atributo adicional por cada medida. Basicamente para os autores, a operação *Pivoting* tem um objetivo claro de transformar medidas em atributos. Isto ocorre porque os autores consideram que os agentes de decisão fazem determinadas consultas *ad-hoc* que usam as métricas como eixo de análise (atributo).

Explicando de uma forma mais detalhada e começando por uma das operações mais elementares, o operador Seleção (σ), equivalente ao *dice* anunciado atrás, considere-se um *predicado atómico*, denotado por p, uma expressão lógica (podendo incluir negações) envolvendo apenas uma dimensão, um *predicado composto*, denotado por P e uma expressão envolvendo um conjunto de predicados atómicos $\{p_1, p_2, \ldots, p_l\}, l \geqslant 1$, da forma:

$$P = p_1 \langle op \rangle p_2 \langle op \rangle \ldots \langle op \rangle p_l \tag{3.8}$$

Então $\langle op \rangle$ representa um operador lógico (por exemplo, $\wedge, \vee$). Assim, a notação matemática para este operador é precisamente a mesma da Álgebra Relacional:

$$\sigma_{(Produto=Bone)} Vendas \tag{3.9}$$

De salientar que o resultado pode não conter valores e nesse o resultado é um cubo vazio: $\langle D, M, A, f, \emptyset, g \rangle$. Um outro pormenor a ter em conta é que este operador trabalha apenas com dimensões, mas tal como referido os autores queriam que este trabalho permitisse que se realização restrições ao nível das métricas. Esse objetivo não é invalidado por esta restrição uma vez que através da operação *Pivoting* pode-se transformar uma métrica numa dimensão e posteriormente realizar a operação de restrição.

Quanto à operação Projeção (π), esta não é anunciada neste artigo, no entanto, através da formalização, uma possível projeção será apenas uma alteração ao nível da função g da

Expressão 3.7, em que o seu domínio passa a ser diferente.

Tal como no trabalho de Gyssens e Lakshmanan [1997], não há uma procura de vistas (ou instâncias) materializadas do reticulado. Por este motivo, apesar das instâncias serem utilizadas nesta formulação, na verdade elas não são mais que vistas temporárias resultantes de operações (analogia ao SQL). Prova disso é a operação de agregação que é talvez das operações mais importantes no OLAP uma vez que ela é a génese de duas das operações mais importantes: o *roll-up* e o *Drill-Down*.

A operação de agregação (α) é baseada nas funções de agregação relacional (SUM, AVG, MAX) e permite que estas funções sejam aplicadas a cubos com uma ou mais dimensões, especificando os atributos a agrupar. Por exemplo, se se quiser agrupar a dimensão tempo do cubo das Vendas (Figura 2.1) por mês, então o operador terá a seguinte forma, em que recebe uma função de agregação sobre uma métrica e recebe os atributos das dimensões a definir:

$$\alpha_{[SUM(lucro),\{nome,mes\}]}(Vendas) \tag{3.10}$$

Os autores ainda apresentam e demonstram mais algumas operações binárias para relacionar dois cubos, nomeadamente: o **Produto Cartesiano** ($\times$), por exemplo para relacionar o cubo das vendas (que inclui os lucros) e um cubo com informação de descontos dos produtos; a **Junção** ($\bowtie$), que é um caso de Produto Cartesiano mas entre dois cubos com dimensões em comum; a **União** ($\cup$) permite unir dois cubos e é particularmente importante para situações em que há cubos com estrutura igual, mas por exemplo para regiões diferentes do país.

Por fim, os autores ainda apresentam duas das operações mais importantes deste trabalho que basicamente transformam métricas em dimensões e vice versa, respetivamente **pull** e **push**. A operação **pull** (ϕ), basicamente converte um conjunto de métricas para dimensões, através de uma função de mapeamento. Na prática o que acontece é que alguns elementos do conjunto M desaparecem e passam a coexistir (através de uma reunião de conjuntos) no conjunto A, da Expressão 3.7.

Quanto ao operador **push** (ψ), este faz exatamente o inverso. Esta operação pode ser particularmente importante para voltar a pôr uma medida que outrora foi filtrada através do operador σ que exigia que fosse uma dimensão. De salientar que estas operações não têm qualquer tipo de restrição ao nível do tipo de dados. Isto não é referido no documento, mas uma vez que as funções têm domínios e contradomínios específicos para situação, então isso significa que não problemas de compatibilidades, nomeadamente realizar um SUM a um valor de uma dimensão.

3.6 Modelo de Cabibbo e Torlone

O modelo de Cabibbo e Torlone [1998] não é tão detalhado matematicamente quanto os dois anteriores, no entanto, tem uma estrutura muito interessante ao nível do EMD em termos de hierarquias. Neste trabalho, a abordagem aos EMDs é feita como se fosse algo paralelo ao modelo do hipercubo de Gray et al. [1997], não havendo sequer qualquer referência ao modelo de Harinarayan et al. [1996].

De uma forma geral, a abordagem funciona da seguinte maneira: existe um conjunto de dimensões que podem ou não ser uma única tabela, ou seja, uma dimensão pode estar normalizada numa "rede" (grafo) de tabelas que formam as hierarquias de um possível reticulado.

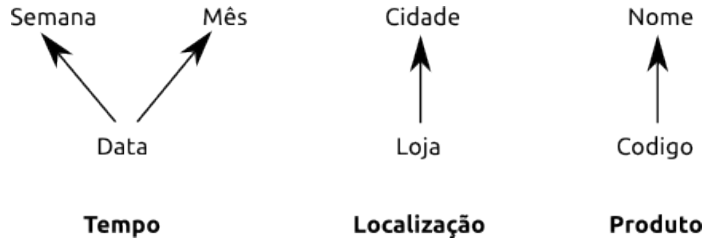
Existe uma "espécie" de tabela de factos, chamada de *f-table*, que basicamente é resultado de uma expressão algébrica que junta as várias dimensões e associa métricas. Esta *f-table* é utilizada como *output* para as várias materializações possíveis ao longo das operações *roll-up* e portanto perde o conceito de tabela de factos. Ou seja, ela também pode ser resultado de operações que juntam atributos de dimensões para que seja possível ter hierarquias dentro das dimensões.

Tal como referido, a formulação deste modelo, é um pouco diferente das abordagens de Thomas e Datta [2001] e de Gyssens e Lakshmanan [1997], primeiro porque não existe uma formulação matemática clara de todo o EMD e em segundo lugar porque parte desta abordagem é feita sob a forma visual, através de grafos que explicam de forma bastante interessante as hierarquias das dimensões. No entanto, deve-se ressalvar que a maneira como são criadas as *f-table* é bastante similar ao modelo de Gyssens e Lakshmanan [1997], em que são o resultado do relacionamento (com pesos) entre dimensões (e atributos de dimensões).

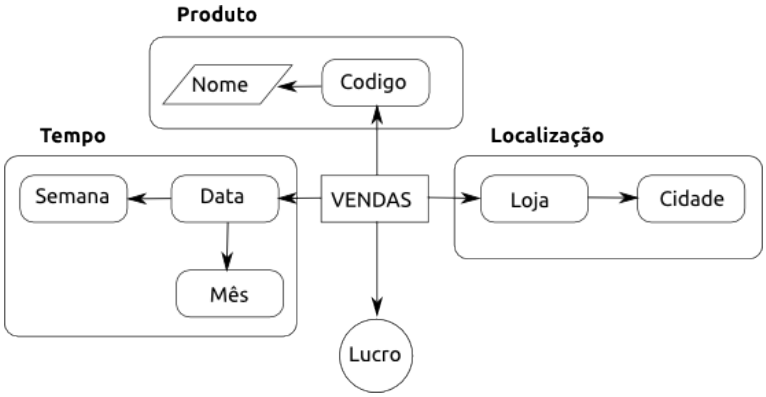
3.6.1 O Espaço Multi-Dimensional (EMD)

O EMD deste modelo é nomeado de *"MultiDimensional Data Model"*, ou simplesmente $\mathcal{MD}$. Ele é baseado em dois construtores: a dimensão e a *f-table*. As dimensões são categorias sintáticas que permitem especificar de diferentes perspetivas a maneira como se mostra a informação, de acordo com as diferentes perspetivas de negócio.

Cada dimensão está organizada numa hierarquia de níveis, correspondendo a diferentes níveis de granularidade da informação (Figura 3.4a). Quando um nível de uma hierarquia da dimensão l_1 precede um nível l_2, a nomenclatura é exatamente a mesma utilizada no modelo de Harinarayan et al. [1996]: $l_1 \preccurlyeq l_2$.



(a) Hierarquia interna das dimensões



(b) Representação gráfica da *f-table*

Figura 3.4: DM da Figura 2.1 através do modelo de Cabibbo e Torlone [1998]

Relativamente às *f-tables*, estas são funções parciais de coordenadas simbólicas (definidas através de diferentes combinações de níveis) para medidas. De uma forma geral podia-se dizer que são todas vistas temporárias consoante o nível de materialização que se quer ter num sistema OLAP. Uma linha da *f-table* é portanto uma coordenada na qual ela define os vários pontos de cada dimensão e valoriza com métricas.

Apesar de o documento centrar-se bastante na demonstração via grafos, o resumo da especificação do EMD segundo Cabibbo e Torlone [1998] é o seguinte:

- Existe um conjunto de dimensões D constituídos por vários nodos. Cada nodo está associado a nível e portanto podemos dizer que: $D = \{l_1, l_2 \ldots, l_n\}$.
- $F = [A_1 : l_1, \ldots, A_n : l_n] \rightarrow [M_1 : l'_1, \ldots, M_m : l'_n]$ é uma *f-table* sobre D.
- Para cada atributo A_i existe um l_i e para cada l_i pode existir um l'_i tal que $l_i \preccurlyeq l'_i$, definido da mesma maneira que um conjunto parcial ordenado, de forma similiar à definição da Secção 3.2.
- Para o caso dos atributos sem hierarquia, aqueles apenas acrescentam informação, descrições, eles são construídos através de uma função (que resulta em nodos com um

aspeto de paralelogramo).

- Cada medida M_i também tem um nível l_i.

3.6.2 Operações

Neste modelo são apresentadas várias operações, nomeadamente as mais conhecidas da álgebra relacional. De salientar que não existe um definição concreta da operação *Drill-Down* e que a operação *Pivoting* também não é definida apesar de ser possível tê-la através das operações normais da álgebra relacional.

Começando pela Seleção (σ), se E é uma expressão-f com o esquema $[A_1, \ldots, A_n] \to [M_1, \ldots, M_m]$ e ϑ uma condição (definida no prágrafo a seguir), então $\sigma_\vartheta(E)$ é uma expressão sobre o mesmo esquema E.

A expressão sobre um esquema *f-table* é uma expressão boleana (lógica) da forma $t \Theta t'$, em que t e t' são atributos, medidas ou constantes e Θ é um predicado de comparação, nomeadamente $=, <, \neq$, etc.

O resultado tem o mesmo esquema de E, mas apenas com as entradas que satisfazem a condição ϑ. No geral, o resultado satisfaz as mesmas dependências funcionais de E. No entanto, se a condição ϑ for da forma $A_i = A_j$, em que A_i e A_j são atributos, então o resultado satisfaz também as dependências funcionais: $A_i \to A_j$ e $A_j \to A_i$.

Relativamente à Projeção (π), se E é uma expressão-f com o esquema $[A_1, \ldots, A_n] \to [M_1, \ldots, M_m]$, $h \leq n$ e $k \leq m$, e E satisfaz as dependências funcionais $A_1, \ldots, A_h \to A_i$, para cada $i > h$, então $\pi_{[A_1,\ldots,A_h]\to[M_1,\ldots,M_k]}(E)$ é uma expressão-f com o esquema $[A_1, \ldots, A_h] \to [M_1, \ldots, M_k]$.

Ou seja, a projeção é feita sobre atributos e não sobre métricas e o número de entradas da tabela *input* é o mesmo número da tabela *output*.

Quanto à Agregação (ψ), considere-se E como uma expressão-f com o esquema $[A_1, \ldots, A_n] \to [M_1, \ldots, M_m]$, $k \leq n$, $N_1, \ldots, N_l$ nomes das medidas e $g_1, \ldots, g_l$ uma função de agregação de $\mathcal{F}$. Então $\psi_{A_1,\ldots,A_k}^{N_1=g_1(M_{i_1}),\ldots,N_l=g_l(M_{i_l})}(E)$ é uma expressão-f com o esquema $[A_1, \ldots, A_n] \to [N_1, \ldots, N_l]$

Por fim, relativamente ao *roll-up* (ϱ), considere-se novamente E como uma expressão-f com o esquema $[A_1, \ldots, A_n] \to [M_1, \ldots, M_m]$ e $A_i : l$ um atributo de E. Se A' é um nome de um atributo e l' é um nível tal que $l \preccurlyeq l'$, então $\varrho_{A_i:l_i}^{A':l'}(E)$ é uma expressão com o esquema $[A_1, \ldots, A_{n-1}, A'] \to [M_1, \ldots, M_m]$.

Os autores ainda referem outras outras expressões, nomeadamente o **Produto Cartesiano**,

a **Junção Natural**, **Renomear**, entre alguns outros.

Quanto à não existência da operação *Drill-Down*, isto ocorre uma vez que este modelo trabalha sobre vistas temporárias e portanto não consegue acrescentar informação, apenas consegue filtrar as vistas já existentes. De uma forma conceptual, o *Drill-Down* seria uma operação que transformaria um atributo A com um nível l num outro atributo A' com nível l' tal que: $l' \preccurlyeq l$. O problema aqui estaria na obtenção dos valores das métricas, uma vez que seria impossível arranjar operações de "desagregação de cálculos".

Por exemplo, não seria possível calcular a métrica "lucro" da Figura 4.10a vindo de um esquema da Figura 4.10b.

3.6.3 Considerações finais

Este modelo centra-se acima de tudo numa formulação de uma linguagem gráfica para fazer uma associação direta ao que o Agente de Decisão realmente quer, navegação dos dados. O modelo é fundamentado também através de uma matemática simples de conjuntos parciais ordenados e sobre uma álgebra que tenta ser o mais similar possível à Álgebra Relacional.

O facto de ser uma formulação mais prática permite ter-se uma noção clara do que é um espaço OLAP na prática e as suas operações. Na prática, um espaço OLAP é um conjunto de tabelas que associam acontecimentos consoante as coordenadas (atributos das dimensões). Daqui tem-se de destacar o facto de perder o conceito de tabela de factos visto que pode-se rapidamente concluir que este modelo define então um espaço OLAP como sendo um conjunto de tabelas que definem muitos factos consoante o relacionamento entre tabelas.

De facto, esta noção é a mais aproximada da realidade, no entanto, em Data Warehousing, só existe uma tabela de factos e todas as materializações são tabelas redundantes calculadas a partir da tabela de factos e portanto não definem exatamente um facto, visto que tem uma granularidade diferente.

Apesar de este modelo incluir os conjuntos parciais ordenados, que definem um reticulado, não existe qualquer associação com o artigo de Harinarayan et al. [1996]. No entanto, em termos de navegação a similaridade é elevada. Apenas existe a diferença de que quando existe navegação (*roll-up*) existe uma perda de informação que inevitavelmente afeta todo o modelo, uma vez que não permite voltar atrás, executar o *Drill-Down*.

Em todo o caso, em termos de dados materializados, vendo de uma forma similar ao modelo de Harinarayan et al. [1996], podia-se facilmente definir na prática o *Drill-Down*.

3.7 Modelo de Pardillo et al.

O modelo de Pardillo et al. [2008] é bastante mais atual que os anteriores e portanto é normal que já tenha conseguido resolver algumas das problemáticas. Aliás, o título do *paper* sugere precisamente isso.

Apesar de a formulação ser diferente do normal, talvez para algo mais objetivo, ela tem o problema de não ser apresentada sob uma forma mais lógica. A formulação é apresentada através de meta-código que explica as operações. O modelo tem o objetivo de juntar o UML a consultas SQL sobre um sistema OLAP.

Os autores deste modelo começam por referir que o problema atual das formulações existentes é simples: exageram na semântica, complicando depois a produção de algo realmente ótimo para trabalhar. Por outro lado, a matemática poderá também limitar certos aspetos nomeadamente a definição de meta-conteúdos de categorização do funcionamento do cubo.

Por este motivo, os autores tentaram abordar o problema mais ao nível dos dados e não dos meta-dados dos EMDs tal como ocorre nas outras formulações apresentadas nas secções anteriores.

3.7.1 O Espaço Multi-Dimensional (EMD)

O modelo centra-se numa definição de cubo global, em que as instâncias seguem uma estrutura similar. Este aspeto é importante para permitir realizar operações de *Drill-Down*.

Um cubo é definido como um conjunto de tuplos que basicamente são os factos, os dados. Cada tuplo representa uma coordenada (em relação às dimensões) e contém também uma medida associada. De uma forma geral, o cubo define-se da seguinte maneira:

$$set(Tuple(V : T, CO : Tuple(D_1 : L_1, \ldots, D_n : L_n))) \tag{3.11}$$

Portanto, o cubo é um cojunto de tuplos. Cada tuplo é constituído por um valor (métrica) e por um outro tuplo que identifica a coordenada. Quanto ao valor V ele pertence ao domínio T. Por outro lado, CO é o sistema de coordenadas que aponta para as dimensões do cubo.

Cada coordenada tem então definida a dimensão e também o nível hierárquico para se poder distinguir diferentes materializações.

3.7.2 Operações

As operações realizadas neste artigo são todas elas demonstradas através de meta-código, algo que não tem interesse para esta tese. Aliás, nem existe meta-código genérico para as várias operações, apenas existe demonstrações de como manuesear o cubo.

Por exemplo, a operação seguinte, remove uma dimensão do cubo que se está a analisar:

```
def Cube::removeDimension(x: Axis, a: Additivity): Cube =
  let cos = self->collect(c_i |
     c_i.co.excluding(x))->asSet() in
  self->allMeasures()->collect(m |
    cos->collect(co_i | Tuple {
      co = co_i,
      v = self->select(c_i | c_i.v.type = m)
      ->select(co_i.includes(c_i.co))
      ->collect(v)->a
    }) )->flatten()
```

Comando 3.1: Operação de remoção de dimensões

Analizando o código, apercebemo-nos facilmente que o conjunto das dimensões é afetado, ficando sem um dos eixos de análise, um dos elementos da coordenada. Para além disso, a métrica, é também alterada, neste caso com uma operação de adição.

Os autores destinguem esta operação da operação *roll-up*. Conceptualmente as duas operações seriam a mesma coisa, no entanto, como isto são situações práticas eles resolveram criar esta distinção. No caso de um *roll-up* a operação foi definida desta maneira:

```
def Cube::rollUp(x: Axis, r: Rolling, a: Additivity):
    Cube =
  let cos = self->collect(c_i | c_i.co.rollUp(x, r))
    ->flatten()->asSet() in
  self->allMeasures()->collect(m |
    cos->collect(co_i | Tuple {
      co = co_i,
      v = self->select(c_i |
        c_i.co.rollUp(x, r).equals(co_i)
      )->collect(v)->a
    }) )->flatten()
```

Comando 3.2: Operação *roll-up*

O artigo inclui a maioria das operações que normalmente são apresentadas na literatura atual, nomeadamente o *Slice&Dice*, que é uma operação de seleção de dimensões e filtragem de valores, o *Drill-Down* que permite aumentar o nível de detalhe com base numa fonte de dados mais detalhada (naturalmente não é possível calcular valores previamente agrupados

ou sumarizados), *Drill-Across* em que existe uma junção de cubos, entre algumas outras operações.

Através deste conjunto de operações os autores estão construir uma framework capaz de lidar com um sistema OLAP ficando ela entre o cliente que faz as consultas e o SQL da base de dados que suporta o sistema. Este funcionamento via UML [Demuth e Hussmann, 1999] visa substituir as consultas MDX.

3.7.3 Considerações

As operações deste modelo estão construídas através de meta-código e portanto não são o objetivo desta tese. No entanto, este EMD tem uma construção intessante ao nível da visão espacial, através de coordenadas. O facto de um cubo ser definido através de conjuntos de dados e não através construções da meta-informação da estrura OLAP, dá uma visão diferente.

A forma como constroi os cubos, através de meta-código, não é importante para este trabalho, no entanto ao nível conceptual mostra uma forma importante de um resultado posterior às formalizações apresentadas nas secções anteriores.

A utilização de um modelo que transforme as formalizações das secções anteriores em algo ao nível de conjuntos de dados, tal como é congeturado neste artigo, é um passo importante para uma melhor formalização do OLAP.

3.8 Comparativo entre os modelos

Os modelos apresentados neste documento estão com uma ordenação temporal em que praticamente todos eles se correlacionam no sentido em que tentam melhorar o que estava feito previamente. Portanto, o nível de completude tendeu a melhorar conforme o tempo.

A Tabela 3.2 ilustra de forma resumida os aspetos mais importantes dos modelos apresentados, neste capítulo.

Modelo	Tipo de Formalização	Operações	Definição de Cubo	Instância ou Cuboide
Gray et al. [1997]	Baseada em SQL	*Histogram*, sub-total, *roll-up*, *Drill-Down* e *cross-tabulation*	Tabela de Factos em SQL que cresce horizontalmente conforme o número de cuboides	Vista produzida em SQL
Harinarayan et al. [1996]	Baseada em SQL e matemática de conjuntos	*roll-up* e *Drill-Down*	Conjunto de tabelas organizado	Uma tabela que inclui um nível de sumarização
Gyssens e Lakshmanan [1997]	Matemática com objetivo de aproximar o OLAP à Álgebra Relacional	Maioria das operações da Álgebra Relacional e ainda *push* e *pull* para converter medidas em dimensções	Definição da meta-informação através do 3-tuplo: $\langle D, R, par \rangle$	Uma instância é um conjunto dos relaciomanentos entre as dimensões com pesos (métricas). Genericamente: $rm(rd_1.A, ..., rd_n.A, M)$
Golfarelli et al. [1998]	Conceptual ao nível matemático e gráfico de DWs	As operações são de consultas sobre instâncias	Um cubo é definido pelo 6-tuplo $F = (M, A, N, R, O, S)$	Uma instância é um conjunto de factos, cada um definido como um m-tuplo: $(\alpha_1, \ldots, \alpha_i) \in Dom(d_1)\times, \ldots, \times Dom(d_i)$
Thomas e Datta [2001]	Matemática, através de conjuntos	As operações tendem todas a serem similares às da Álgebra Relacional	Definição da meta-informação através do 4-tuplo $\langle D, M, A, f \rangle$	Uma instância acrescenta os dados ao 4-tuplo do cubo, ficando: $\langle D, M, A, f, V, g \rangle$
Cabibbo e Torlone [1998]	Definição matemática com objetivo de aproximar o OLAP a grafos	São navegações sobre o Grafo	Conjunto de *f-tables* definidas assim: $F = [A_1 : l_1, \ldots, A_n : l_n] \rightarrow [M_1 : l'_1, \ldots, M_m : l'_n]$	Não tem esse conceito. No entanto é uma *f-table* também mas de menor dimensionalidade
Pardillo et al. [2008]	UML com sistema de coordenadas	Operações construídas em meta-código	É o conjunto de todos os dados, da seguinte maneira: $set(Tuple(V : T, CO : Tuple(D_1 : L_1, \ldots, D_n : L_n)))$	Não define instância. Mas uma operação de navegação altera a estrutura da coordenada CO

Tabela 3.2: Resumo das características de cada Modelo

Capítulo 4

Aplicação prática dos Modelos

4.1 Situação Prática

Nesta secção mostra-se a situação à qual os vários modelos apresentados deverão responder. O exemplo apresentado nesta secção é um problema clássico na demonstração do funcionamento de um DW e de um sistema OLAP.

A situação prática é a seguinte: existe uma empresa que tem várias lojas por Portugal e armazena informação de cada venda, nomeadamente do local, qual produto e a data em que foi comercializado. Pretende-se então criar um sistema de DW que funcione como base a um sistema OLAP que permita os Agentes de Decisão analisarem os dados das vendas.

Os Agentes de Decisão deverão poder analisar os dados ao nível da localização, ou seja, investigar como estão as vendas de cada loja, ou então agrupando as lojas por cidade para resumir os dados de forma a tornar análise mais simples. Para além disso, os Agentes de Decisão poderão analisar os dados conforme dias do ano, semanalmente ou até mensalmente. Por fim, eles poderão também analisar os dados pelo produto. Todas as análises visam analisar o lucro que a empresa tem.

A conclusão imediata, ao nível da construção do DW, é uma estrutura em estrela com 3 dimensões: Tempo, Produto e Localização. No meio das dimensões contempla a Tabela de Factos que inclui apenas uma métrica que define o lucro de cada facto. Essa estrutura está ilustrada na Figura 4.1.

A tabela da Figura 4.1 apresenta um excerto dos dados que esse DW pode conter, ou seja, apresenta 4 colunas, em que 3 delas contêm informação das dimensões e portanto

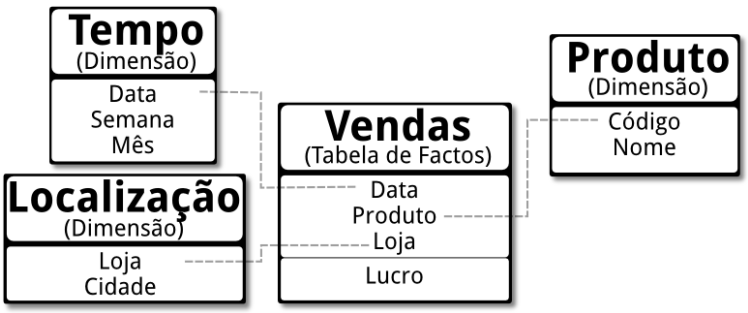


Figura 4.1: Estrutura do DM "Vendas"

Tempo	Produto	Localização	Lucro
1 de Janeiro	Boné	X	3
1 de Janeiro	Brincos	Y	5
1 de Janeiro	Brincos	Z	5
1 de Janeiro	Colar	Z	4
2 de Janeiro	Colar	X	4

Tabela 4.1: Excerto da tabela de factos "Vendas"

são os eixos de análise. A 4ª coluna apresenta a métrica que valoriza os factos, neste caso contempla o valor do lucro de cada produto vendido.

Data	Semana	Mês
1 de Janeiro	1	1
2 de Janeiro	1	1
...	...	...
30 de Janeiro	52	12
31 de Janeiro	52	12

(a) Dimensão *Tempo*

Código	Nome
41234	Boné
83971	Brincos
38612	Colar
...	...

(b) Dimensão *Produto*

Loja	Cidade
LojaX	Braga
LojaY	Braga
LojaZ	Porto
...	...

(c) Dimensão *Localização*

Figura 4.2: Excertos das dimensões do caso de estudo

4.2 Hipercubo de [Gray et al., 1997]

Este modelo, tal como foi referida na parte teórica que o apresentou, é bastante prático e portanto resume-se à aplicação de SQL. No caso do exemplo prático, considerando-se então que se quer construir um sistema OLAP que permita analisar os lucros segundo esses 3 eixos de análise, então a nomenclatura usada no modelo de Gray et al. [1997] para construir o sistema, ou seja, o cubo, é a apresentada no Comando 4.1.

Exceto a operação "`GROUP BY CUBE`", todo o comando é SQL comum de uma base de dados

OLTP. Relativamente a essa operação, ela é uma operação que junta duas operações mais simples do SQL: o GROUP BY que agrega dados repetidos e a UNION que junta dados na mesma tabela.

```
SELECT Tempo, localizacao, produto, SUM(lucro)
FROM Vendas;
GROUP BY CUBE  Tempo, localizacao, produto
```

Comando 4.1: Lucro por Tempo, por Localização e por Produto

Assim, o resultado da execução do Comando 4.1 é uma tabela que junta as várias agregações possíveis para cada eixo de análise (coluna referente a cada dimensão). Assim, o cubo final não é mais que uma tabela que para além dos factos provenientes da Tabela de Factos, inclui também outras entradas que têm as agregações pré-calculadas. Para cada uma dessas agregações, quando um eixo é removido, é célula que o identifica fica com o valor `ALL`. O resultado pode ser visto na Tabela 4.2.

De uma forma mais simplista, mas não relacional, esta mesma tabela pode ser vista conceptualmente Tabela 3.1 apresentada na Secção 3.1. Uma vez que a base conceptual deste modelo é analisar os dados segundo diferentes perspetivas, como se um espaço multidimensional matemático se tratasse, então os autores deste modelo chamaram ao modelo de Hipercubo. Este nome vem do facto de poder haver múltiplos eixos de análise. No caso de um exemplo simplista como é este apresentado neste capítulo, só existe 3 eixos de análise e pode-se até chamar de Cubo. Aliás, normalmente em vez de se chamar hipercubo, para simplificar o nome, chama-se simplesmente de Cubo, mesmo com muitos eixos de análise.

Neste caso em concreto, conceptualmente, o cubo deste exemplo está representado na Figura 4.3, que mostra também algumas das materializações possíveis. Essas diferentes materiliazações são acedidas através das operações OLAP que basicamente são consultas simples à Tabela 4.2 recorrendo aos valores `ALL` referidos anteriormente.

Relativamente ao *roll-up*, no caso do modelo de Gray et al. [1997], esta operação gera sempre um cubo, mas é utilizado apenas sobre hierarquias de dimensões, não abrangendo nem unificando com a operação de remoção de dimensões, tal como o modelo de Harinarayan et al. [1996] sugere. Na prática neste modelo o *roll-up* apenas permite acrescentar os "ALL" referidos anteriormente consoante a métrica e a respetiva hierarquia da mesma.

Supondo por exemplo que se quer fazer um *roll-up* segundo a métrica "Cidade" da dimensão "Localização", então o comando a ser executado seria o seguinte:

```
SELECT  data, mes, localizacao, produto, SUM(lucro)
FROM    Vendas
ROLLUP  mes(data) AS mes,
CUBE  data, localizacao, produto
```

Comando 4.2: Operação *roll-up* que remove o eixo Produto

Tempo	Localização	Produto	Lucro
1 de Janeiro	Loja X	Boné	3
1 de Janeiro	Loja X	ALL	3
1 de Janeiro	Loja Y	Brincos	5
1 de Janeiro	Loja Y	ALL	5
1 de Janeiro	Loja Z	Brincos	5
1 de Janeiro	Loja Z	Colar	4
1 de Janeiro	Loja Z	ALL	9
2 de Janeiro	Loja X	Colar	4
2 de Janeiro	Loja X	ALL	4
1 de Janeiro	ALL	Boné	3
1 de Janeiro	ALL	Brincos	10
1 de Janeiro	ALL	Colar	4
2 de Janeiro	ALL	Colar	4
1 de Janeiro	ALL	ALL	17
2 de Janeiro	ALL	ALL	4
ALL	Loja X	Boné	3
ALL	Loja X	Colar	4
ALL	Loja X	ALL	7
ALL	Loja Y	Brincos	5
ALL	Loja Y	ALL	5
ALL	Loja Z	Brincos	5
ALL	Loja Z	Colar	4
ALL	Loja Z	ALL	9
ALL	ALL	Boné	3
ALL	ALL	Brincos	10
ALL	ALL	Colar	8
ALL	ALL	ALL	21

Tabela 4.2: Lucro segundo diferentes combinações de dimensões

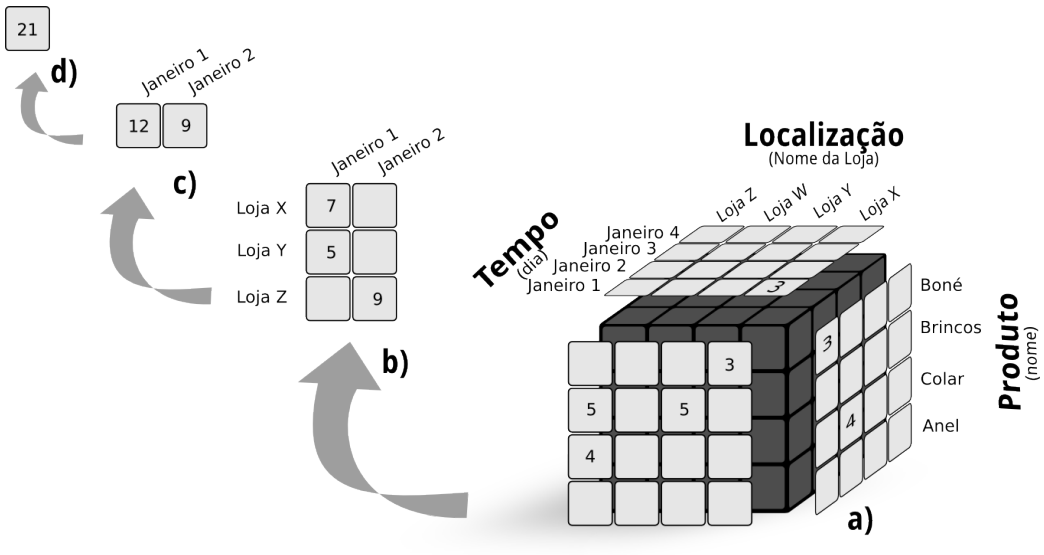


Figura 4.3: O Cubo é constituído por várias vistas de agregação. a), b) e c) representam respetivamente as 3 últimas colunas da Tabela 3.1 e d) a soma de todos os lucros

4.3 A reticulado de Harinarayan et al. [1996]

A modelo conjeturado por [Harinarayan et al., 1996] acrescenta algum dinamismo no hipercubo de [Gray et al., 1997] e acima de tudo alguma organização com uma base matemática sólida: os conjuntos parcialmente ordenados, também chamados de reticulados ou *lattice*.

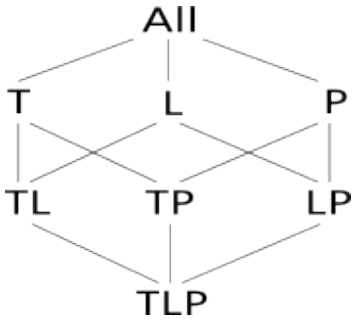
O facto deste modelo basear-se no cubo e mudar a noção de armazenamento das materializações em diferentes tabelas, torna-o imediatamente mais escalável na prática. Apesar de necessitar depois de um controlo mais rigoroso para interagir entre as várias tabelas das materializações, este modelo já não carece dos problemas que a utilização dos valores `ALL` trariam, nomeadamente ocupar imenso espaço e exigir muito processamento para filtrar os dados nas consultas.

De uma forma geral, este modelo veio trazer uma organização capaz não só economizar espaço mas também de permitir criar algoritmos matemáticos de materialização parcial que são inevitavelmente necessários em bases de dados gigantes em que o pre-processamento seria quase impossível. Este facto, de se saber se é impossível ou não, pode ser facilmente previsto pois a complexidade de um reticulado aumenta exponencialmente visto que ela é resultado de todas as combinações possíveis, neste caso entre os diferentes atributos e dimensões.

Assim, seguindo o exemplo deste capítulo, cada conjunto de materializações apresentados na Tabela 4.2, na verdade são tabelas distintas. Deste modo, por exemplo, a última linha que tem as 3 dimensões como `ALL` gerariam uma tabela com um único registo. Um outro exemplo seriam as 3 últimas seguintes que seriam uma outra tabela contendo apenas informação relativa a cada Produto, descartando os eixos de análise Localização e Data.

TL $\preccurlyeq$ TLP
LP $\preccurlyeq$ TLP
TP $\preccurlyeq$ TLP
T $\preccurlyeq$ TL
ALL $\preccurlyeq$ L $\preccurlyeq$ LP $\preccurlyeq$ TLP
ALL $\preccurlyeq$ L $\preccurlyeq$ TL $\preccurlyeq$ TLP
Mês $\preccurlyeq$ dia

(a) Exemplos de dependências entre dimensões



(b) Representação do reticulado

Figura 4.4: Possível reticulado do EMD da Figura 2.1

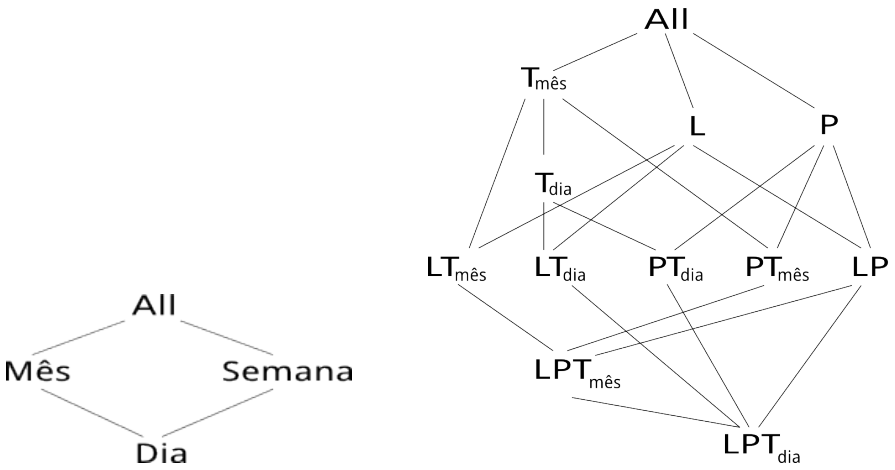
A organização de todas as tabelas materializadas é feita através do conceito de reticulado. De uma forma resumida, essa organização é o conjunto das combinações possíveis entre as várias dimensões. O resultado pode ser visto na Figura 4.4b. Na prática o reticulado em OLAP (ao contrário das *Lattices* em matemática que tem a construção inversa) começa por ser construído na Tabela de Factos. Assim o primeiro nodo é a Tabela de Factos e é representado na Figura 4.4b como `TLP` (iniciais das 3 dimensões).

O nível seguinte na construção do reticulado, terá dados mais agregados. Neste caso, cada cuboide será uma tabela à imagem da Tabela de Factos, mas sem uma das dimensões. Portanto este nível tem o mesmo número de cuboides que as combinações possíveis com menos um elemento entre as dimensões do nível anterior. Ou seja, para n dimensões o nível seguinte são as combinações para $n-1$ dimensões e portanto: $\binom{n}{n-1} = \binom{3}{2} = 3$.

Tendo agora os 3 cuboides resultantes, $\{TL, LP, TP\}$, volta-se a fazer as combinações para $n-1$ dimensões, resultando agora novamente três possíveis: T, L, P, visto que: $\binom{3}{1} = 3$. Por fim, faz-se o mesmo para zero dimensões, que dá um único cuboide.

A complexidade aumenta com o número de dimensões, como é óbvio. Mas aumenta igualmente por cada hierarquia de cada dimensão. Uma hierarquia está presente numa dimensão quando um atributo abrange um outro atributo. No caso deste exemplo, o atributo `mes` agrupa vários valores do atributo `data` (dia do ano), ou seja, há vários dias que pertencem a

um mês. Essa hierarquia, ou dependência, está representada na última linha da Tabela 4.4a.



(a) Hirarquia de atributos da dimensão tempo

(b) Reticulado do EMD com apenas uma hierarquia

Figura 4.5: Hierarquias dos reticulados

Se se voltarmos a construir o reticulado, mas agora com esta pequena diferença, a complexidade aumenta consideravelmente, tal como se pode constatar pela Figura 4.5b.

No caso de dependências como as da Figura 4.5a, uma vez que restringem a hierarquia, elas não aumentam tanto o número de combinações, mas aumentam a complexidade de construção do reticulado. Para definir estas dependências em ramo que constituem a dimensão "Tempo", tem-se o seguinte:

$$ALL \preccurlyeq Mes \tag{4.1}$$
$$ALL \preccurlyeq Semana \tag{4.2}$$
$$Mes \preccurlyeq Dia \tag{4.3}$$
$$Semana \preccurlyeq Dia \tag{4.4}$$
$$Mes \not\preccurlyeq Semana \tag{4.5}$$

A construção do reticulado tem então que acrescentar uma nova hierarquia em relação à da Figura 4.5b. No entanto, é necessário garantir que não existe dependências entre o Mês e Semana, ficando com uma estrutura representada na Figura 4.6.

Voltando ao reticulado mais simples (Figura 4.4b), esta estrutura conceptual na prática é um conjunto de tabelas que têm uma organização muito bem definida que é depois utilizada pelas operações OLAP de navegação, nomeadamente do *roll-up* e *Drill-Down*. Esse conjunto de tabelas está representado na Figura 2.4. Em vez dessa representação por tabelas, visto que cada uma dessas tabelas na prática são cubos com diferentes dimensionalidades, pode-

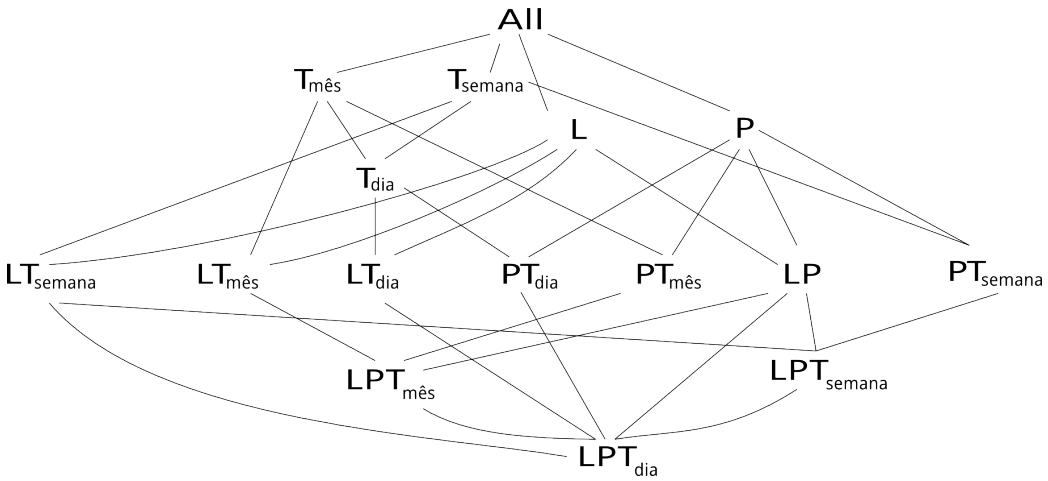


Figura 4.6: Reticulado com a hierarquia em ramo apresentada na Figura 4.5a

se também representar um cubo tal como é mostrado na Figura A.1, que neste caso é um cubo do reticulado apresentado na Figura 4.5b.

4.4 Modelo de Gyssens e Lakshmanan [1997]

Com base na formalização apresentada na Secção 3.3 é possível definir toda a meta-informação de um DM. Essa formalização base, define um Cubo que depois pode ser instanciado (construção dos cuboides, ou materializações).

Em primeiro lugar define-se uma *Table Schema* genérica, que basicamente define um cubo, através do 3-tuplo seguinte, chamado "Vendas":

$$Vendas = \langle D, R, par \rangle \tag{4.6}$$

De seguida, define-se cada elemento que compõe essa *Table Schema*. Começando pelo conjunto D, que contém os nomes das dimensões, seguindo então o exemplo deste capítulo, tem-se o seguinte:

$$D = \{ Tempo, Produto, Localizacao \} \tag{4.7}$$

Seguidamente, define-se o conjunto R que inclui todos os atributos presentes no DM. Isto significa que neste conjunto devem estar não só os atributos das dimensões, mas também as métricas da Tabela de Factos. Relativametne aos atributos da Tabela de Factos que são chaves estrangeiras, estes não são considerados visto que são atributos das dimensões.

Assim tem-se:

$$\begin{aligned} R = \{ & data, semana, mes, \\ & codigo, nome, \\ & loja, cidade, \\ & lucro\} \end{aligned} \tag{4.8}$$

Por fim, é necessário correlacionar as dimensões com os atributos. Para tal, define-se a função *par* da seguinte maneira:

$$par(Tempo) = \{data, semana, mes\} \tag{4.9}$$
$$par(Produto) = \{codigo, Nome\} \tag{4.10}$$
$$par(Localizacao) = \{loja, cidade\} \tag{4.11}$$

De forma mais extensa, pode-se expressar a *Table Schema* como sendo então:

$$\begin{aligned} Vendas = \langle \{ & Tempo, Produto, Localizacao\}, \\ & \{data, semana, mes, codigo, nome, loja, cidade, lucro\}, \\ & par\rangle \end{aligned} \tag{4.12}$$

Em particular, neste caso, facilmente se repara que existe um atributo que não é abrangido pela função *par*. Isto ocorre porque esse atributo não pertence a nenhuma dimensão mas, sim, à Tabela de Factos. É portanto uma métrica, que pertence ao conjunto M. Esta métrica é apenas utilizada na definição de instâncias uma vez que ela pode ou não existir, consoante o nível de agregação e as propriedades de agregação que cada métrica tem (algo que o modelo de Gyssens e Lakshmanan [1997] não abrange).

Agora com a *Table Schema* da Expressão 4.12 é possível construir as instâncias. Por exemplo, a seguinte instância satisfaz todas as condições referidas atrás, e basicamente é um conjunto de 4 relacionamentos constroi a Tabela de Factos:

$$\textbf{rTempo}(data, semana, mes) \tag{4.13}$$
$$\textbf{rProduto}(codigo, nome) \tag{4.14}$$
$$\textbf{rLocalizacao}(loja, cidade) \tag{4.15}$$

$$\textbf{rm}((r)T(empo).data, (r)P(roduto).codigo, (r)L(ocalizacao).loja, lucro) \tag{4.16}$$

Com base neste trabalho de Gyssens e Lakshmanan [1997] é possível então definir um EMD com uma relativa exatidão. Aliás, é facilmente percetível que *rm* tem chaves estrangeiras para as dimensões. No entanto há aspetos que não são abordados (Secção 3.3.3).

Vendas			Tempo						
			Mês	Jan			Fev		
			Semana	1	2	...	1	2	...
Localização & Produto	Loja	Nome	*(Lucro)*						
	Loja X	Boné		(3)	(4)	...	(23)	(18)	...
		T-shirt		(14)	(13)	...	(14)	(12)	...
		⋮		⋮	⋮		⋮	⋮	
	Loja Y	Brincos		(5)	(27)	...	(28)	(26)	...
		Sapatos		(14)	(11)	...	(12)	(13)	...
		⋮		⋮	⋮	...	⋮	⋮	
	⋮	⋮		⋮	⋮		⋮	⋮	

Figura 4.7: Parte da TF do exemplo apresentado na página 48

Relativamente às operações de estruturação "Fold" e "UnFold", elas funcionam de forma algo similar ao objetivo do pivoting. Servem para reestruturar uma tabela. Para se compreender bem a nomenclatura e forma como elas funcionam, tome-se como exemplo a tabela da Figura 4.7 que representa a Tabela de Factos do exemplo deste capítulo mas com mais dados. Então, a Figura 4.8 resulta da expressão seguinte:

$$\mathbf{fold}^{tempo}(\mathbf{unfold}^{Prdouto}_{codigo}(\mathbf{unfold}^{Localizacao}_{loja}(\mathbf{fold}^{Localização}(Vendas)))) \tag{4.17}$$

Vendas		Localização			
		Loja	Loja X	Loja Y	...
Produto	Nome	*Mês, Semana, Lucro*			
	Boné		(Jan,1,3)	(Jan,1,26)	...
	T-shirt		(Jan,2,20)	(Jan,2,27)	...
	⋮		⋮	⋮	
	Camisa		(Fev,1,23)	(Fev,1,28)	...
	Camisa		(Fev,2,18)	(Fev,1,26)	...
	⋮		⋮	⋮	...
	Sapatos		⊥	(Jan,1,14)	...
	Sapatos		⊥	(Jan,2,11)	...
	⋮		⋮	⋮	

Figura 4.8: Tabela resultante da Expressão 4.17

Quanto às operações de seleção e de "navegação" pelo cubo (*roll-up* e *Drill-Down*), na prática, elas são operações que constroem tabelas. Por exemplo, a tabela da Figura 4.7

resulta da seguinte da Expressão 4.18:

$$\begin{aligned}(TabelaFactos[Produto(Nome : \_\mathbf{N}), Localização(Loja : \_\mathbf{C}) \\ \rightarrow (Mes : \_\mathbf{M}, Dia : \_\mathbf{D}, Lucro : \_\mathbf{L})]| \\ Vendas[LocalizacaoProduto(Nome : \_\mathbf{N}, Loja : \_\mathbf{C}), Tempo(Mes : \_\mathbf{M}, Dia : \_\mathbf{D}) \\ \rightarrow (Lucro : \_\mathbf{L})]\end{aligned} \tag{4.18}$$

Uma tabela com informação agregada seria por exemplo a da Figura 4.9, que basicamente apresenta a média acumulada diária dos lucros do mês de Janeiro, resultante da Expressão 4.19.

$$\begin{aligned}avg_{avglucro \leftarrow lucro}(acumulativo[Produto(Nome : \_\mathbf{N}), Localização(Cidade : \_\mathbf{C}), Interval(Upto : \_\mathbf{D}) \\ \rightarrow (Dia : \_\mathbf{D'}, Lucro : \_\mathbf{L})]| \\ Vendas[LocalizacaoProduto(Nome : \_\mathbf{N}, Cidade : \_\mathbf{C}), Tempo(Mes : janeiro, Dia : \_\mathbf{D'}) \\ \rightarrow (Lucro : \_\mathbf{L})]\wedge \\ Vendas[Tempo(Mes : janeiro, Dia : \_\mathbf{D}] \wedge \_\mathbf{D'} \leq \_\mathbf{D})\end{aligned} \tag{4.19}$$

Em que *avg* (Expressões 4.20 e 4.21) é uma função de sumarização que calcula a média.

$$avg_{avglucro \leftarrow lucro} : 2^{dom(Dia) \times dom(Lucro)} \rightarrow dom(avglucro) \tag{4.20}$$

$$avg(L) = (1/|L|) \sum_{(d',l) \in L} L \tag{4.21}$$

Vendas			Tempo	
			Mês	Jan
Localização & Produto	Loja	Nome	*(avgLucro)*	
	Loja X	Boné		(3)
		Brincos		(14.5)
		⋮		⋮
	Loja Y	Brincos		(5)
		Sapatos		(12.5)
		⋮		⋮
	⋮	⋮		⋮

Figura 4.9: Tabela resultante da Expressão 4.19

Apenas para finalizar, um entrada numa tabela não é mais que um m-tuplo. Por exemplo,

o primeiro elemento da tabela da Figura 4.9 define-se da seguinte maneira:

$$(LojaX, Bone, Jan, 3) \ .$$

4.5 Modelo de Golfarelli et al.

A especificação deste modelo é feita em duas partes, a parte conceptual e depois os factos expressados nos tuplos. Relaticamente ao modelo conceptual do DW, começa-se por definir genericamente o DM da seguinte maneira:

$$F = (M, A, N, R, O, S) \tag{4.22}$$

Tendo em conta que nesta situação prática apenas existe uma métrica, então:

$$M = \{lucro\} \tag{4.23}$$

Todos os atributos das dimensões exceto o atributo "nome" da dimensão "Produto" estão em hierarquia. Portanto o conjunto de atributos A é definido da seguinte maneira:

$$A = \{Data, Semana, Mes, Loja, Cidade, Codigo\} \tag{4.24}$$

Relativamente ao atributo referido anteriormente, o atributo "nome", como ele é apenas um atributo descritivo e não entra na hierarquia, então:

$$N = \{Nome\} \tag{4.25}$$

Relativamente à estrutura das hierarquias, estas são definidas pelos tuplos do seguinte conjunto R:

$$R = \{(Data, Semana), (Data, Mes), (Loja, Cidade)\} \tag{4.26}$$

Por fim, é necessário definir associar a função de agregação aplicável à métrica "lucro". Como esta métrica é o lucro obtido pela empresa, ela pode estar associada a qualquer atributo do DM. Por isso, considerando apenas que se quer analisar a métrica em termos de somas de lucro, tem-se:

$$\begin{aligned} S = \{&(lucro, Data, SUM), (lucro, Semana, SUM), (lucro, Mes, SUM), \\ &(lucro, Loja, SUM), (lucro, Cidade, SUM), (lucro, Codigo, SUM), \} \end{aligned} \tag{4.27}$$

Tempo	local.cidade	Produto	Lucro
1 de Janeiro	Braga	Boné	3
1 de Janeiro	Braga	Brincos	5
1 de Janeiro	Porto	Brincos	5
1 de Janeiro	Porto	Colar	4
2 de Janeiro	Faro	Colar	4

Tabela 4.3: Resultado da operação de *roll-up*, em que se passa a visualizar os dados da Tabela 4.1 por Cidades. Os valores das cidades estão descritos na Tabela 3.1

Com base nestas estrutura, já possível instanciar os dados. No caso da Tabela de Factos, esta é definida pelo conjunto dos tuplos $(d_t, d_l, d_p) \in Dom(tempo) \times Dom(Localizacao) \times Dom(Produto)$. Para o caso concreto especificado neste capítulo, com base na Tabela 4.1, tem-se então os seguintes tuplos como domínio do DM:

$$(\text{"1 } de\ Janeiro\text{"}, \text{"}Bone\text{"}, \text{"}Loja\ X\text{"})$$
$$(\text{"1 } de\ Janeiro\text{"}, \text{"}Brincos\text{"}, \text{"}Loja\ Y\text{"})$$
$$(\text{"1 } de\ Janeiro\text{"}, \text{"}Brincos\text{"}, \text{"}Loja\ Z\text{"})$$
$$(\text{"1 } de\ Janeiro\text{"}, \text{"}Colar\text{"}, \text{"}Loja\ Z\text{"})$$
$$(\text{"2 } de\ Janeiro\text{"}, \text{"}Colar\text{"}, \text{"}Loja\ X\text{"})$$

Analisando pela variável P que define o padrão de agregação, para este caso desta instância acima, tem-se que $P = \{Data, Loja, Codigo\}$ e analisando S, conclui-se que estes 3 atributos permitem que a métrica seja calculada. Portanto, a instância da Tabela de Factos inclui métricas e por isso mesmo se se fizer uma operação de consulta aos dados através da Expressão 4.28, obtém-se exatamente a Tabela 4.1.

$$VENDAS(data, loja, codigo).lucro \tag{4.28}$$

No caso de se querer realizar um *roll-up*, por exemplo agrupando o atributo "Lojas" para visualizar os dados por cidade, então faz-se a consulta seguinte para, novamente obter o lucro (de salientar que neste modelo não existe nomes de dimensões, elas recebem o nome do primeiro atributo):

$$VENDAS(data, loja.cidade, codigo).lucro \tag{4.29}$$

O resultado desta operação está apresentado na Tabela 4.3 o que se pode constatar não existe nenhuma agregação, pois não há dados que possam ser agregáveis.

Tempo	local.cidade	Lucro
1 de Janeiro	Braga	8
1 de Janeiro	Porto	9
2 de Janeiro	Faro	4

Tabela 4.4: Resultado da operação de *roll-up* da Expressão 4.30

Tempo	local.cidade	Lucro
1 de Janeiro	Braga	8
1 de Janeiro	Porto	9

Tabela 4.5: Resultado da consulta da Expressão 4.31

No entanto, se se fizer uma nova operação de *roll-up* sobre o resultado anterior, mas agora removendo uma das dimensões, por exemplo a dimensão "Produto", então a consulta expressa-se da seguinte maneira, novamente para obter a métrica "Lucro":

$$VENDAS(data, loja.cidade).lucro \tag{4.30}$$

O resultado desta operação está apresentado da Tabela 4.4. De salientar que através de S sabe-se como agrupar as métricas, que neste caso está definida a operação "SUM", ou sumatório.

Relativamente a uma operação de *Slice&Dice*, que é uma consulta normal, mas com projeto e seleção dos dados, por exemplo, se se quiser saber quanto lucro teve esta empresa no dia 1 de Janeiro por cidade, então aplica-se a operação seguinte:

$$VENDAS(data, loja.cidade; data =' 1\ de\ Janeiro').lucro \tag{4.31}$$

Resultado está representado na Tabela 4.5 e apresenta apenas dois valores, um para cada cidade do dia 1 de Janeiro.

4.6 Modelo de Thomas e Datta [2001]

O modelo de [Thomas e Datta, 2001] segue uma nomenclatura muito semelhante ao do modelo apresentado na secção anterior. Centra-se acima de tudo na formalização da meta-informação do cubo e depois, sobre ele são criadas as instâncias de forma ainda não materializada, ou seja, novamente com meta-informação.

Começando pela formalização do cubo, ou melhor, do DM o primeiro passo, genérico é a

definição do 4-tuplo seguinte:

$$Vendas = \langle D, M, A, f \rangle \tag{4.32}$$

Este é um cubo genérico que de seugida definido consoante as restrições e caracterizações do problema apresentado neste capítulo. Começando pelo nome das dimensões, define-se então D:

$$D = \{Tempo, Produto, Localizacao\} \tag{4.33}$$

Para além das dimensões, e ao contrário do que acontecia no modelo anterior, é necessário definir as métricas que serão usadas neste DM. O conjunto das métricas M é definido da seguinte forma:

$$M = \{Lucro\} \tag{4.34}$$

Para além das métricas é também necessário definir os atributos que compõem as dimensões. Assim, o conjunto A, definido a seguir, contém todos os atributos encontrados em todas as dimensões:

$$\begin{aligned} A = \{&data, semana, mes, \\ &codigo, nome, \\ &loja, cidade\} \end{aligned} \tag{4.35}$$

Por fim, a função de mapeamento f associa as dimensões às várias métricas:

$$f(Tempo) = \{data, semana, mes\} \tag{4.36}$$
$$f(Produto) = \{codigo, nome\} \tag{4.37}$$
$$f(Localizacao) = \{loja, cidade\} \tag{4.38}$$

Resumindo, este cubo, chamado "Vendas", é expressado apenas pelas seguinte expressão:

$$\begin{aligned} Vendas = \langle &\{Tempo, Produto, Localizacao\}, Lucro\}, \\ &\{data, semana, mes, \\ &codigo, nome, \\ &loja, cidade\}, f \rangle \end{aligned}$$

Quanto às instâncias, estas funcionam como um acrescento ao cubo original. Assim, uma instância genericamente definda da seguinte maneira:

$$I_{Vendas} = \langle D, M, A, f, V, g \rangle \tag{4.39}$$

Os elementos D, M, A e f mantêm-se iguais e a diferença está em em V e g. Relativamente

ao conjunto V, este contém os dados materiliazados em m-tuplos. Assim, no caso em concreto da Tabela de Factos representada na Figura 4.1 tem-se o seguinte:

$$V = \{\langle 3\rangle, \langle 5\rangle, \langle 5\rangle, \langle 4\rangle, \langle 4\rangle\} \tag{4.40}$$

Quanto à função de mapeamento g, esta relaciona a "coordenada" de valores associados a um valor em V. Portanto, para completar a instanciação da Tabela de Factos referida, define-se g da seguinte maneira:

$$g(1deJaneiro, Bone, LojaX) = \langle 3\rangle \tag{4.41}$$
$$g(1deJaneiro, Brincos, LojaY) = \langle 5\rangle \tag{4.42}$$
$$g(1deJaneiro, Brincos, LojaZ) = \langle 5\rangle \tag{4.43}$$
$$g(1deJaneiro, Colar, LojaZ) = \langle 4\rangle \tag{4.44}$$
$$g(2deJaneiro, Colar, LojaX) = \langle 4\rangle \tag{4.45}$$

Relativamente às operações, todas elas tendem a ser similares à Álgebra Relacional. Para demonstrar como elas funcionam, segue-se alguns exemplos de perguntas que se poderia fazer ao DM deste capítulo.

Se se quiser saber a de vendas semanais só do mês de janeiro, então a expressão seguinte permite obter a informação pedida. Note-se que é praticamente igual à Álgebra Relacional:

$$\sigma_{(tempo.mes \geq 1 \wedge tempo.mes \leq 3)}(\alpha_{[AVG(lucro),\{tempo.mes\}]}(\mathbf{Vendas})) = C_{resultado} \tag{4.46}$$

Por outro lado, se se quiser procurar por produtos em que o total de lucros da primeira semana de Janeiro são superiores a um quarto dos lucros totais do mês de Fevereiro. Inicialmente cria-se o cubo com os lucros do mês de Fevereiro:

$$\alpha_{[SUM(lucro),\{Produto.nome\}]}\sigma_{tempo.mes=Fevereiro}(\mathbf{Vendas}) = C_F \tag{4.47}$$

Depois cria-se um outro cubo, mas agora relativo apenas ao verão do mesmo ano (considere-se t, como o nome da dimensão "tempo"):

$$\alpha_{[SUM(lucro),\{Produto.nome\}]} \sigma_{(t.semana=1 \wedge t.mes=janeiro)}(\mathbf{Vendas}) = C_J \tag{4.48}$$

Como é necessário agora filtrar os lucros, é preciso converter as métricas em dimensões, através da operação **pull**. Esta operação cria uma dimensão chamada LUCROS e um atributo da dimensão chamado Lucro (k é a função de mapeamento que transforma o nome

da métrica no nome do atributo):

$$\phi_{[\{lucro\},\{LUCROS\},k(lucro)=LUCROS]}(C_F) = C_{fevereiro} \tag{4.49}$$

$$\phi_{[\{lucro\},\{LUCROS\},k(lucro)=LUCROS]}(C_J) = C_{janeiro} \tag{4.50}$$

Por fim os dois cubos são juntados com a restrição previamente exigida:

$$\sigma_{c_{janeiro}.lucro>1/3c_{fevereiro}.lucro}(C_{fevereiro} \bowtie C_{janeiro}) \tag{4.51}$$

4.7 Modelo de Cabibbo e Torlone [1998]

Apesar de este modelo estar construído também sobre matemática de conjuntos, ele funciona de maneira alternativa em relação aos anteriores. O modelo não distingue uma Tabela de Factos de uma instância e o cubo é um estado, ou instância, que se está a analisar.

Começando pela definição das dimensões, tem-se então o conjunto $\mathcal{D}$ (não definido no modelo mas necessário para expressar as dimensões) que inclui as várias dimensões:

$$\mathcal{D} = \{D_{Tempo}, D_{Localizacao}, D_{Produto}\} \tag{4.52}$$

Cada dimensão é constituída por vários atributos, que neste modelo são nodos de um grafo. Portanto, cada dimensão é definda da seguinte maneira:

$$D_{Tempo} = \{data, semana, mes\} \tag{4.53}$$

$$D_{Localizacao} = \{loja, cidade\} \tag{4.54}$$

$$D_{Produto} = \{codigo, nome\} \tag{4.55}$$

Cada nodo tem associado um nível hirárquico, que permite então depois controlar as diferentes hierarquias de cada dimensão e que também permite depois contruir o grafo. Assim, tem-se as seguintes regras:

$$semana \preccurlyeq data$$
$$mes \preccurlyeq data$$
$$cidade \preccurlyeq loja$$

Por fim, a definição da *f-table* é a seguinte:

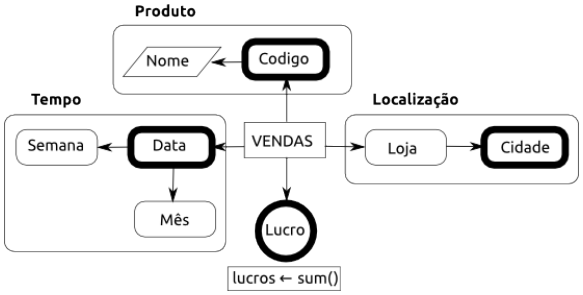
$$VENDAS[Localização : loja, Produto : codigo, Tempo : data] \rightarrow [Lucro : numeric] \tag{4.56}$$

Partindo dos dados da Figura 4.1, um exemplo de uma coordenada da *f-table*"Vendas" é $[Data : 1\ de\ Janeiro, Produto : Boné, Local : Braga]$, que aponta para o valor 3 da métrica *Lucro*. A representação gráfica desta *f-table* e das dimensões é mostrada na Figura 3.4b.

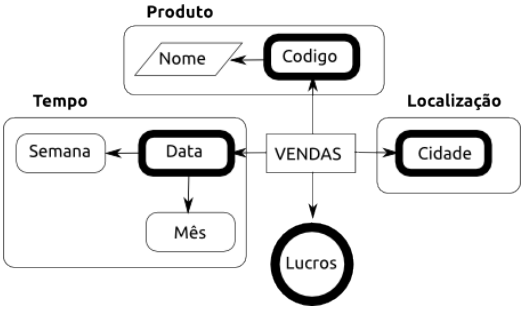
Relativamente às operações de navegação pelo EMD, visualmente elas resumem-se a seguir o grafo. Por exemplo um *roll-up* na dimensão tempo, não é mais que a *f-table* da seguinte expressão:

$$VENDAS[Localização : Loja, Produto : id, Tempo : Mes] \rightarrow [Lucro : numeric] \tag{4.57}$$

Graficamente esta operação da Expressão 4.57 pode ser vista na Figura 4.10. O interessante deste modelo gráfico é que consegue-se saber imediatamente quais as hierarquias que é possível voltar a fazer *roll-up* para agregar mais os dados.



(a) Dimensões escolhidas para realizar o *roll-up*



(b) Representação gráfica da *f-table* resultante da operação

Figura 4.10: Operação *roll-up* em relação à *f-table* da Figura 3.4b

Por outro lado, também é fácil constatar que a operação de *Drill-Down* fica imeditamente impossível de ser realizada. Na prática ela é sempre possível caso os dados se mantenham, no entanto, graficamente, existe exclusões de atributos aquando das operações *roll-up* que depois não permitem voltar atrás.

Quanto à operação que permite chegar à *f-table* da Expressão 4.57, ela resume-se à seguinte expressão:

$$\psi^{lucros=sum(lucro)}_{Data,Cidade,id}(\varrho^{Cidade:Cidade}_{Localizacao:Loja}(VENDAS)) \tag{4.58}$$

4.8 Modelo de Pardillo et al. [2008]

O modelo de [Pardillo et al., 2008] é bastante prático, muito direcionado à programação em UML, mas tem uma maneira de definir um DM bastante interessante, através de um sistema de coordenadas, e igualmente através de conjuntos.

No caso prático deste capítulo, o cubo seria definido genericamente pela expressão seguinte:

$$\begin{aligned} &set(Tuple(Lucro : Integer, \\ &\qquad CO : Tuple(Tempo : data, Produto : id, Localizacao : loja))) \end{aligned} \tag{4.59}$$

Ou seja, existe um conjunto de tuplos com aquela estrutura que formam o cubo. De uma forma mais prática, seguindo apenas a Tabela de Factos representada na Figura 4.1, teríamos então essa tabela definida como o seguinte conjunto::

$$\begin{aligned} &\{(3, (1dejaneiro, Bone, LojaX)), \\ &\ (5, (1dejaneiro, Brincos, LojaY)), \\ &\ (5, (1dejaneiro, Brincos, LojaZ)), \\ &\ (4, (1dejaneiro, Colar, LojaZ)), \\ &\ (4, (2dejaneiro, Colar, LojaX))\} \end{aligned} \tag{4.60}$$

Relativamente às operações, elas são definidas em metacódigo e na prática o *output* delas são outros conjuntos de tuplos de valores e coordenadas. No caso dos valores, estes são calculados normalmente através de funções de agregação. Relativamente à estrutura das coordenadas, ela é afetada consoante o tipo de operações, por exemplo no caso de um *roll-up*, o sistema de coordenadas poderá ter um domínio diferente (no caso de um *roll-up* dentro de uma hierarquia de uma dimensão) ou então poderá ter uma cardinalidade diferente, com menos elementos.

Capítulo 5

Um modelo de OLAP baseado em reticulados

No domínio dos sistemas OLAP, facilmente se encontram artigos que foram desenvolvidos com base no modelo apresentado em Harinarayan et al. [1996], trazendo cada um deles diferentes algoritmos de materialização de estruturas, em particular materializações parciais Morfonios et al. [2007]. Aquele é, portanto, um modelo importante em sistemas OLAP. No entanto, como veremos há aspetos básicos desse modelo carecem de uma definição mais formal. Por outro lado, os artigos que abordam a questão da especificação formal de cubos e operações de consulta subjacentes acabam frequentemente por não adotar o modelo de reticulados ou, então, simplesmente, descuram alguns pormenores importantes dos sistemas OLAP, nomeadamente as restrições das funções de agregação ou mesmo as próprias materializações da estrutura do cubo.

Neste capítulo propomos um modelo capaz de unir esses pontos-chave: formalizamos os aspetos básicos do modelo de Harinarayan et al. [1996], mas com um tratamento de atributos descritivos, métricas e funções de agregação. Ao longo do capítulo o modelo será ilustrado através do exemplo do Capítulo 4.

5.1 Meta dados e EMD

Começamos por definir os conceitos que servem para tipificar as estruturas de dados (tabelas) a introduzir posteriormente. Especificamente, o conceito de meta dados servirá de "tipo" para o *Data Mart* e o Espaço Multi-Dimensional (EMD) pode ser visto como um

conjunto de endereços onde se localizarão as diferentes materializações do cubo de dados.

Definição 1 (Dimensão).

1. Um *atributo* é um conjunto.

2. Uma *dimensão* é um reticulado finito, $D = (\mathcal{H}, \leq)$, em que $\mathcal{H}$ é um conjunto de atributos. ditos *atributos de hierarquia* da dimensão D e $\leq$ define uma hierarquia entre os mesmos; e em que o supremo de $\mathcal{H}$, $max\mathcal{H}$, é o atributo singular $\{*\}$, que em OLAP normalmente é denominado "ALL".

3. Se D é uma dimensão, $hrq(D)$ representa o conjunto $\mathcal{H}$ de D.

4. O conjunto de todas as dimensões representa-se por $\mathcal{D}$. O número de dimensões é n. Assumimos que $\mathcal{D}$ tem uma ordem total; portanto $\mathcal{D} = \{D_1, \ldots, D_n\}$.

■

Os atributos de hierarquia não incluem os atributos descritivos, que serão apresentados na definição seguinte.

Utilizando o exemplo prático da Secção 4.1 e da Figura 4.1 iremos ilustrar a formalização à medida que apresentamos os conceitos. Este exemplo tem $n = 3$ dimensões:

$$\begin{aligned} D_1 =& T(empo) \\ D_2 =& L(ocalizacao) \\ D_3 =& P(roduto) \end{aligned}$$

Tem-se:

$$D = (\mathcal{H}_D, \leq_D), \qquad D \in \{T, L, P\}$$

pelo que, para $D \in \{T, L, P\}$, $hrq(D) = \mathcal{H}_D$.

Os atributos de cada dimensão são:

$$\begin{aligned} \mathcal{H}_T =& \{Data, M\hat{e}s, Semana, ALL\} \\ \mathcal{H}_L =& \{Loja, Cidade, ALL\} \\ \mathcal{H}_P =& \{C\acute{o}digo, ALL\} \end{aligned}$$

Sendo cada atributo um conjunto matemático, temos o seguinte (com base na Figura 4.2):

$$\begin{aligned} Data &= \{\text{"}1de Janeiro\text{"}, \text{"}2de Janeiro\text{"}, \ldots, \text{"}31de Dezembro\text{"}\} \\ Mês &= \{\text{"}Janeiro\text{"}, \ldots, \text{"}Dezembro\text{"}\} \\ Semana &= \{x \in \mathbb{N} | x \leq 52\} \\ Loja &= \{\text{"}LojaX\text{"}, \text{"}LojaY\text{"}, \text{"}LojaZ\text{"}\} \\ Cidade &= \{\text{"}Braga\text{"}, \text{"}Porto\text{"}\} \\ Codigo &= \{\text{"41234"}, \text{"83971"}, \text{"38612"}\} \\ Nome &= \{\text{"}Bone\text{"}, \text{"}Brincos\text{"}, \text{"}Colar\text{"}\} \end{aligned}$$

Para cada dimensão D a ordem dos atributos $\leq_D$ é dada pela Figura 5.1.

Figura 5.1: Definição das 3 dimensões do exemplo prático da Secção 4.1

De uma forma menos gráfica, as ordens são definidas da seguinte maneira:

$$\begin{aligned} \leq_T = \{&(Data, Data), (Mês, Mês), (Semana, Semana), (ALL, ALL), \\ &(Data, Mês), (Data, Semana), (Data, ALL), (Mês, ALL), (Semana, ALL)\} \\ \leq_L = \{&(Loja, Loja), (Cidade, Cidade), (ALL, ALL), \\ &(Loja, Cidade), (Loja, ALL), (Cidade, ALL)\} \\ \leq_P = \{&(Código, Código), (ALL, ALL), (Código, ALL)\} \end{aligned}$$

Definição 2 (Atributos).

1. O conjunto de todos os atributos de hierarquia é $HRQ_{\mathcal{D}} = \bigcup_{D \in \mathcal{D}} hrq(D)$.
2. O conjunto dos atributos descritivos denota-se por $DSCR$.
3. Uma *associação de atributos de* $\mathcal{D}$ *a atributos de DSCR* é uma função $par_{\mathcal{D}} : DSCR \rightarrow HRQ_{\mathcal{D}}$.

4. $ATR(D)$ é o conjunto de atributos da dimensão D e é dado pela função:

$$ATR : \mathcal{D} \to (HRQ_{\mathcal{D}} \cup DSCR)$$
$$D \mapsto (hrq(D) \backslash \{ALL\}) \cup par_{\mathcal{D}}^{-1}[hrq(D)] \ .$$

5. Para cada dimensão D, assumimos uma *ordem total de apresentação* dos elementos de $ATR(D)$; portanto, $ATR(D) = \{A_1, \ldots A_k\}$. O *campo de* D é o produto cartesiano

$$cmp(D) = A_1 \times \cdots \times A_k \ .$$

■

Voltando ao exemplo, o conjunto de todos os atributos de hierarquia é:

$$HRQ_{\mathcal{D}} = \{Data, Mês, Semana, Loja, Cidade, Código, ALL\} \ .$$

Para além destes, existem os atributos descritivos que apenas acrescentam informação a um determinado atributo de hierarquia que lhe está associado pela função $par_{\mathcal{D}}$.

No caso do exemplo prático, apenas temos um atributo descritivo:

$$DSCR = \{Nome\} \ .$$

Este atributo descritivo é associado ao atributo da hierarquia *Código*, através de:

$$par_{\mathcal{D}}(Nome) = Código \ .$$

Ora $Código \in hrq(P)$. Então $Nome$ é um dos elementos do conjunto dos atributos da dimensão P:

$$\begin{aligned} ATR(P) &= (hrq(P) \backslash \{ALL\}) \cup par_{\mathcal{D}}^{-1}[hrq(P)] \\ &= \{Código\} \cup \{Nome\} \\ &= \{Código, Nome\} \end{aligned}$$

Para a ordem de apresentação dos elementos de $ATR(P)$ convencionamos que

$$Código < Nome \ .$$

Os conjuntos de atributos das restantes dimensões são constituídos apenas pelos respetivos atributos de hierarquia e as respetivas ordens de apresentação são:

$$\begin{aligned} Data < Mês < Semana \\ Loja < Cidade \end{aligned}$$

As ordens de apresentação determinam os campos das dimensões:

$$cmp(T) = Data \times Mês \times Semana$$
$$cmp(L) = Loja \times Cidade$$
$$cmp(P) = Código \times Nome$$

Num DM para além dos atributos das dimensões, que são incluídos como chaves estrangeiras na Tabela de Factos (e posteriores materializações dos cuboides no cubo), existem também os atributos que valorizam os factos, as métricas.

Definição 3 (Valores e métricas).

1. VAL é um conjunto, em que cada $V \in VAL$ é um conjunto de valores.
2. Uma *métrica* é uma função parcial $M : H_1 \times \cdots \times H_n \hookrightarrow V$ em que:
   i) o i-ésimo fator do domínio de M é um atributo de hierarquia da i-ésima dimensão, ou seja: para todo $1 \leq i \leq n$, $H_i \in hrq(D_i)$.
   ii) $V \in VAL$;
3. Uma métrica diz-se *primitiva* se, para cada $1 \leq i \leq n$, $H_i = min(hrq(D_i))$.

■

Através desta forma de definir as métricas, consegue-se ser abrangente o suficiente para as várias materializações do reticulado. Por exemplo uma métrica primitiva tem o maior nível de detalhe e é métrica da Tabela de Factos.

Na prática uma métrica representa o valor dos factos com máximo de detalhe que o DM consegue definir. Consoante os níveis de agregação/sumarização, essa métrica vai sofrendo alterações de valor à medida que o nível de detalhe vai mudando, perdendo o conceito original de métrica.

Nesse sentido, uma Tabela de Factos, que define os factos e portanto tem o maior detalhe, tem a métrica na sua forma mais primitiva. Essa métrica é definida pelas chaves estrangeiras que são os atributos das dimensões que definem o menor detalhe, ou seja os atributos que são mínimos das dimensões.

No caso concreto do exemplo prático que temos vindo a explorar, temos apenas uma métrica, "Lucro", que é a seguinte função parcial:

$$lucro : Data \times Loja \times Código \hookrightarrow VME$$

VME é o conjunto de Valores Monetários em Euros com duas casas decimais, que matematicamente se expressa por:

$$VME = \{z \times 0.01 \in \mathbb{Z}\} = \{x \in \mathbb{R} | 100x \in \mathbb{Z}\}$$

Como neste caso específico temos apenas uma métrica, VAL é o conjunto singular:

$$VAL = \{VME\}$$

É importante definir quais as funções de agregação que as métricas devem apresentar conforme cada perspetiva de análise. É comum aplicar-se a função de agregação 'soma'; no entanto, pode ser necessário aplicar funções como 'máximo', 'mínimo', ou 'média', ou até outras.

Definição 4 (Funções de agregação).

1. O conjunto das *funções primitiva de agregação* é

$$FAGR = \{SUM, MIN, MAX, COUNT\} \ .$$

2. Uma *associação de funções de agregação para VAL e D* é uma função

$$AGR : VAL \times HRQ_{\mathcal{D}} \to \wp(FAGR)$$
$$(V, H) \mapsto F \subseteq FAGR$$

Tal que:

a) Para cada V, H, o conjunto $AGR(V, H)$ está totalmente ordenado;

b) Para todo $D \in \mathcal{D}$, $H, H' \in hrq(D)$, se $f \in AGR(V, H')$ e $H \leq H'$ então $f \in AGR(V, H)$;

c) Para cada V, H, se $f \in AGR(V, H)$ então (V, f) é um monóide abeliano[1].

■

A condição c) garante que a operação binária f pode ser generalizada a uma operação de aridade arbitrária Φ; e em vez de, digamos, $f(x_1, x_2, x_3)$ escrevemos $\overset{3}{\underset{i=1}{\Phi}} x_i$. Esta notação generaliza a notação de somatório. Por exemplo se $V = \mathbb{R}$ e $f = SUM$ então f é a operação binária $x_1 + x_2$ de adição de números reais, que é depois generalizada para a operação $\sum_i x_i$.

[1] Ou seja, f é comutativa, associativa e tem de ter um elemento neutro em V.

Através desta definição garantimos que uma métrica que produza valores em V pode ser agregada pelo atributo H usando a função de agregação f apenas se $f \in AGR(V, H)$. Portanto, a função AGR deve ser especificada para evitar agregações impossíveis ou desajustadas da realidade (por exemplo fazer um somatório de uma métrica que significa uma média).

Ao nível de agregações, só faz sentido realizar uma determinada agregação a um determinado nível caso essa agregação seja possível nos níveis anteriores, quando existam. A condição b) garante exatamente esta propriedade.

Voltando ao exemplo prático, a métrica "lucro" poderá ser agregada de várias maneiras, nomeadamente através de somatórios ou até de mínimos e máximos. Com vista a definir AGR, introduzimos F como o conjunto de funções de agregação

$$F = \{SUM, MIN, MAX\} \ ,$$

com a ordenação $SUM < MIN < MAX$.

Apenas para ilustrar casos excecionais, vamos supor que não queremos que a métrica seja agregada de nenhuma maneira relativamente aos atributos da dimensão *Produto*, e que, no caso do atributo que define a *Cidade*, apenas queremos agregar pela função SUM. Assim a função AGR define-se da seguinte forma:

$$\begin{aligned} AGR(VME, Data) &= F \\ AGR(VME, Mês) &= F \\ AGR(VME, Semana) &= F \\ AGR(VME, Loja) &= F \\ AGR(VME, Código) &= \{\} \\ AGR(VME, Cidade) &= \{SUM\} \\ AGR(VME, ALL) &= \{F\} \end{aligned}$$

Definição 5 (Meta dados de um *Data Mart*).

1. Os *meta-dados* do *Data Mart* são um quinteto ordenado:

   $$\mathbf{MD} = < \mathcal{D}, DSCR, par_{\mathcal{D}}, VAL, AGR >$$

   em que as cinco componentes de **MD** são definidas de acordo com as definições anteriores.

■

Relativamente ao exemplo que temos vindo a apresentar, este quinteto ordenado é constituído pelos vários conjuntos e funções anteriormente definidos.

Definição 6 (Espaço Multi-Dimensional (EMD)).

1. O EMD é o reticulado do produto das dimensões de $\mathcal{D}$:

$$EMD = D_1 \times \cdots \times D_n$$

2. Os elementos de EMD designam-se por *endereços*. Portanto, um endereço é um tuplo $e = (H_1, \ldots, H_n)$, em que para cada $1 \leq i \leq n$, $H_i \in hrq(D_i)$.

■

Portanto, um endereço é um tuplo $e = (H_1, \ldots, H_n)$, em que, para cada $1 \leq i \leq n$, $H_i \in hrq(D_i)$. Dados $e = (H_1, \ldots, H_n)$ e $e' = (H'_1, \ldots, H'_n)$, a ordem em EMD é definida por: $e \leq e'$ se e só se, para todo $1 \leq i \leq n$, $H_i \leq H'_i$.

O exemplo abordado para a explicação prática deste modelo tem um conjunto de dimensões $\mathcal{D}$, em que cada dimensão está apresentada na Figura 5.1. Nesse sentido, o espaço multi-dimensional representado por EMD tem o aspeto da Figura 5.2, que é um simples produto entre os reticulados que representam as dimensões. No Apêndice B pode-se visualizar com maior detalhe um exemplo do desenvolvimento deste produto.

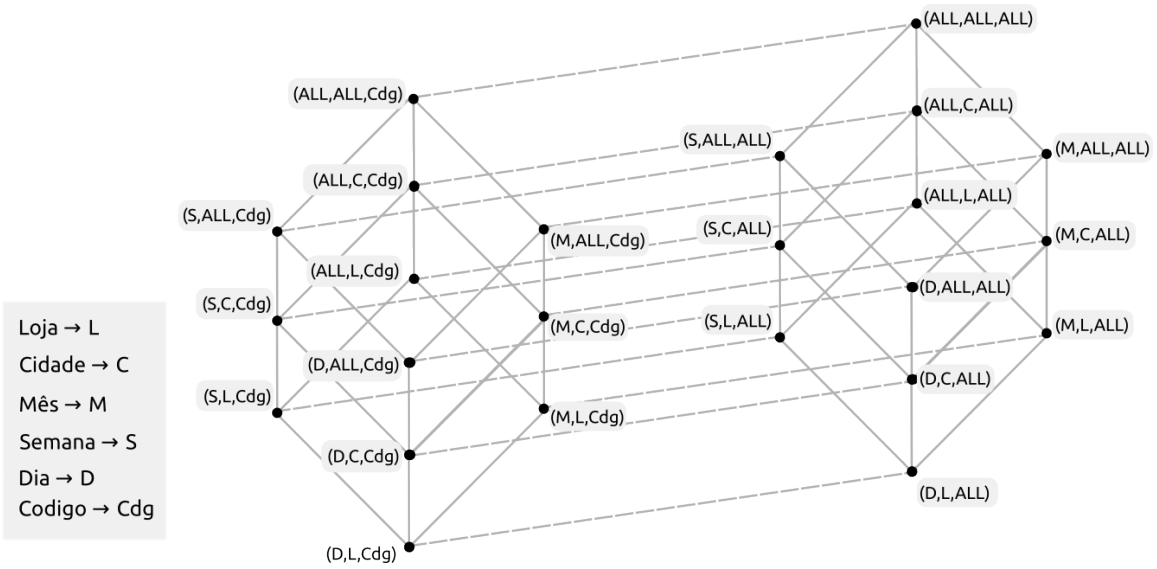


Figura 5.2: Produto entre as dimensões do exemplo prático da página 47

5.2 *Data Mart* e as tabelas de dados

Nesta secção definem-se as estruturas de dados que compõem o sistema OLAP.

Definição 7 (*Data Mart*).

1. Um *data mart* de "tipo" MD é um par $DM = (TF, T)$ tal que:

   i) $TF \subseteq H_1 \times \cdots \times H_n \times V_1 \times \cdots \times V_m$ diz-se a *Tabela de Factos* e é tal que:

      a) Para cada $1 \leq i \leq n$, $H_i = min(hrq(D_n))$.

      b) Para cada $1 \leq j \leq m$, $V_j \in VAL$.

      c) Para cada $1 \leq j \leq m$, se T_j é a projeção de TF às colunas $H_1, \ldots, H_n, V_j$, então T_j é uma função do tipo $H_1 \times \cdots \times H_n \hookrightarrow V_j$, ou seja, T_j é uma métrica primitiva, que vamos denotar por M_j.

   ii) T é a função que associa a cada dimensão uma tabela. Formalmente[2]:

   $$T : (D : \mathcal{D}) \to \wp(cmp(D))$$

   $$D \mapsto T(D) \subseteq cmp(D)$$

   iii) Para cada $D \in \mathcal{D}$, $H, H' \in hrq(D)$, se $H \leq H'$, então a projeção da tabela $T(D)$ às colunas H e H' define uma função do tipo $H \hookrightarrow H'$, que designamos por $h_{H,H'}$. Convenciona-se que $h_{H,ALL}$ é a única função total do tipo $H \to ALL$.[3]

   iv) Para cada $D \in \mathcal{D}$, $H \in hrq(D)$, se $A \in par_{\mathcal{D}}^{-1}(hrq(D))$ então a projeção da tabela $T(D)$ às colunas H e A define uma função do tipo $H \hookrightarrow A$.

   v) Para cada $1 \leq i \leq n$, se $x \in H_i$ ocorre na projeção de TF à coluna H_i, então x também ocorre na projeção de $T(D_i)$ à coluna H_i.

■

A condição iii) diz que a tabela de cada dimensão respeita a hierarquia de atributos da mesma. Note-se que, na mesma condição, se $H = H'$, então a projeção da tabela $T(D)$ às colunas H e H' é subconjunto da função identidade em H.

A condição v) garante que uma chave estrangeira só pode existir caso a chave primária da dimensão já exista.

[2]Note-se que o tipo de T é um tipo dependente.

[3]Note-se que $h_{H,ALL}(x) = *$.

Voltando ao exemplo, a tabela da dimensão *Tempo* é:

$$\begin{aligned} T(D_1) = \{&(\text{"}1de Janeiro\text{"}, \text{"}Janeiro\text{"}, \text{"}1\text{"}),\\ &(\text{"}2de Janeiro\text{"}, \text{"}Janeiro\text{"}, \text{"}1\text{"}),\\ &\ldots\\ &(\text{"}31de Dezembro\text{"}, \text{"}Dezembro\text{"}, \text{"}52\text{"})\} \end{aligned}$$

Recorde-se que $T(D_1) \subseteq cmp(D_1) = Data \times M\hat{e}s \times Semana$.

Pela ordenação definida pelo reticulado da Figura 5.1, temos que:

$$Data \leq M\hat{e}s \qquad\qquad Data \leq Semana$$

De acordo com a condição iii) da definição c existem duas funções

$$h_{Data,M\hat{e}s} \qquad\qquad h_{Data,Semana}$$

que garantem que para cada *Data* temos apenas um *Mês* e uma *Semana* associados. Estas duas funções são:

$$\begin{aligned} &h_{Data,M\hat{e}s} : Data \hookrightarrow M\hat{e}s\\ &h_{Data,M\hat{e}s}(\text{"}1de Janeiro\text{"}) = \text{"}Janeiro\text{"}\\ &h_{Data,M\hat{e}s}(\text{"}2de Janeiro\text{"}) = \text{"}Janeiro\text{"}\\ &\qquad\ldots\\ &h_{Data,M\hat{e}s}(\text{"}31de Dezembro\text{"}) = \text{"}Dezembro\text{"} \end{aligned} \qquad \begin{aligned} &h_{Data,Semana} : Data \hookrightarrow Semana\\ &h_{Data,Semana}(\text{"}1de Janeiro\text{"}) = 1\\ &h_{Data,Semana}(\text{"}2de Janeiro\text{"}) = 1\\ &\ldots\\ &h_{Data,Semana}(\text{"}31de Dezembro\text{"}) = 52 \end{aligned}$$

A tabela da dimensão *Produto* é:

$$\begin{aligned} T(D_3) = \{&(\text{"}41234\text{"}, \text{"}Bon\acute{e}\text{"}),\\ &(\text{"}83971\text{"}, \text{"}Brincos\text{"}),\\ &(\text{"}38612\text{"}, \text{"}Colar\text{"})\} \end{aligned}$$

Esta tabela tem a particularidade de ter um valor descritivo *Nome*. A função seguinte mapeia cada valor do atributo *Código* a um valor do atributo *Nome*:

$$\begin{aligned} &nome : Codigo \hookrightarrow Nome\\ &nome(\text{"}1\text{"}) = \text{"}Bon\acute{e}\text{"}\\ &nome(\text{"}2\text{"}) = \text{"}Brincos\text{"}\\ &nome(\text{"}3\text{"}) = \text{"}Colar\text{"} \end{aligned}$$

A dimensão *Localização* segue os mesmos padrões da dimensão *Tempo*, ou seja, existe uma função $h_{Loja,Cidade}$ que garante que para cada *Loja* existe um *Cidade* associada:

$$\begin{aligned} &h_{Loja,Cidade} : Data \hookrightarrow M\hat{e}s \\ &h_{Loja,Cidade}(\text{"}LojaX\text{"}) = \text{"}Braga\text{"} \\ &h_{Loja,Cidade}(\text{"}LojaY\text{"}) = \text{"}Braga\text{"} \\ &h_{Loja,Cidade}(\text{"}LojaZ\text{"}) = \text{"}Porto\text{"} \end{aligned}$$

A tabela da dimensão *Localização* é então:

$$\begin{aligned} T(D_3) = \{&(\text{"}LojaX\text{"}, \text{"}Braga\text{"}), \\ &(\text{"}LojaY\text{"}, \text{"}Braga\text{"}), \\ &(\text{"}LojaZ\text{"}, \text{"}Porto\text{"})\} \end{aligned}$$

Por fim, resta definir a TF:

$$\begin{aligned} TF = \{&(\text{"}1de Janeiro\text{"}, \text{"}LojaX\text{"}, \text{"}Bon\acute{e}\text{"}, 3), \\ &(\text{"}1de Janeiro\text{"}, \text{"}LojaY\text{"}, \text{"}Brincos\text{"}, 5), \\ &(\text{"}1de Janeiro\text{"}, \text{"}LojaZ\text{"}, \text{"}Brincos\text{"}, 5), \\ &(\text{"}1de Janeiro\text{"}, \text{"}LojaX\text{"}, \text{"}Colar\text{"}, 4), \\ &(\text{"}2de Janeiro\text{"}, \text{"}LojaX\text{"}, \text{"}Colar\text{"}, 5)\} \end{aligned}$$

Note-se $TF \subseteq Data \times Loja \times C\acute{o}digo \times V_1$, onde $V_1 = VME$. Então, pela condição 2 i) c), a projeção de TF às colunas *Data*, *Loja*, *Código*, V_1 é uma métrica primitiva M_1, precisamente a métrica *lucro* referida anteriormente:

$$lucro = m_1 = T_1 = TF : Data \times Loja \times Cod \hookrightarrow VME \ .$$

Neste caso, em que o valor de m referido no ponto 2 da definição **??** é 1, a métrica M_1 é a única que se obtém por projeção e coincide com a própria TF. Quando $m > 1$, esta coincidência não se verifica. A Tabela de Factos apresenta a informação com máximo nível de detalhe, pois *Data*, *Loja* e *Código* são os mínimos das hierarquias das dimensões respetivas.

Finalmente, para ilustrar o ponto v) da definição **??**, os valores "1 de Janeiro" e "2 de Janeiro" da coluna *Data* de TF ocorrem na coluna *Data* de $T(D_1)$.

Proposição 1. *Para todo $d \in \mathcal{D}$, $H = min(hrq(D))$ é chave primária de $T(D)$.*

Demonstração. Por redução ao absurdo. Suponhamos que H não é chave primária de $T(D) \subseteq cmp(D) = A_1 \times \cdots \times A_k$. Sem perda de generalidade suponhamos que o $H = A_1$. Então existem registos diferentes $(a_1 \times \cdots \times a_k), (a'_1 \times \cdots \times a'_k) \in T(D)$ com $a_1 = a'_1$. Como os registos são diferentes então existe um $1 < i \leq k$ tal que $a_i \neq a'_i$.

Primeiro caso: $A_i \in hrq(D)$. Define-se $j = i$.

Segundo caso: $A_i \notin hrq(D)$. Então A_i é um atributo descritivo. Seja j tal que $A_j = par(A_i)$ e seja f a função do tipo $A_j \hookrightarrow A_i$ que resulta de projetar $T(D)$ às colunas de A_j e A_i. Então $f(a_j) = a_i$ e $f(a'_j) = a'_i$. Como $a_i \neq a'_i$, então $a_j \neq a'_j$.

Em ambos os casos concluímos que existe $1 < j \leq k$ tal que $a_j \neq a'_j$ e $A_j \in hrq(D)$. Então os registos diferentes mencionados acima mostram que a projeção de $T(D)$ às colunas A_1 e A_j não define uma função. Mas isto contradiz a definição de *data mart* porque $A_1 \leq A_j$. □

Proposição 2. *Para todo $D = (\mathcal{H}, \leq) \in \mathcal{D}$ e atributos $H, H', H'' \in \mathcal{H}$:*

a) se $H \leq H' \leq H''$ então $h_{H,H''} = h_{H',H''} \circ h_{H,H'}$;

b) se $H \leq H'$ então $h_{H,H'}$ não depende do caminho entre H e H'' no diagrama de Hasse de D.

Demonstração. a) Suponhamos que $H \leq H' \leq H''$. Sejam i, j, k as posições de H, H', H'' na ordem de apresentação de $cmp(D)$. É necessário garantir que $h_{H,H''}(x) = x''$ se e só se existe $x' \in H'$ tal que $h_{H,H'}(x) = x'$ e $h_{H',H''}(x') = x''$.

⇒ Suponhamos $h_{H,H''}(x) = x''$. Então existem $y_1, \ldots, y_n$ tais que $(y_1, \ldots, y_n) \in T(D)$ e $y_i = x$ e $y_k = x''$. Tome-se $x' = y_j$. Então:

- $h_{H,H'}(x) = x'$, pois o par (y_i, y_j) pertence à projeção de $T(D)$ às colunas H e H', na medida em que $(y_1, \ldots, y_n) \in T(D)$.
- $h_{H',H''}(x') = x''$: demonstração análoga.

⇐ Suponhamos agora que existe um $x' \in H'$ tal que $h_{H,H'}(x) = x'$ e $h_{H',H''}(x') = x''$. Então existem $y_1, \ldots, y_n$ tais que $(y_1, \ldots, y_n) \in T(D)$ e $y_i = x$ e $y_j = x'$ e existem $z_1, \ldots, z_n$ tais que $(z_1, \ldots, z_n) \in T(D)$ e $z_j = x'$ e $z_k = x''$. Segue que $y_k = h_{H',H''}(y_j) = h_{H',H''}(z_j) = z_k$. Logo $h_{H,H''}(y_i) = z_k$, ou seja $h_{H,H''}(x) = x''$.

b) É consequência imediata da alínea a).

□

Definição 8 (Tabela de dados).

Seja $e = (H_1, \ldots, H_n)$ um endereço de EMD. Então $TD \subseteq H_1 \times \cdots \times H_n \times V_1 \times \cdots \times V_m$ é uma *tabela de dados localizada em e*, se:

a) Para cada $1 \leq i \leq n$, $H_i \in hrq(D_i)$.

b) Para cada $1 \leq j \leq m$, $V_j \in VAL$.

c) Para cada $1 \leq j \leq m$, se T_j é a projeção de TD às colunas $H_1, \ldots, H_n, V_j$, então T_j é uma função do tipo $H_1 \times \cdots \times H_n \hookrightarrow V_j$, ou seja, T_j é uma métrica.

■

TF é uma tabela de dados localizada no endereço mínimo $(min(hrq(D_1)), \ldots, min(hrq(D_n)))$.

5.3 Operações

Depois de termos definido as estruturas que contêm os dados, passamos ao estado das operações que operam nessas estruturas. Por limitações de tempo, tratamos apenas da operação de navegação *roll-up*.

Definição 9 (*Roll-up*).

Seja $TD \subseteq H_1 \times \cdots \times H_i \times \cdots \times H_n \times V_1 \times \cdots \times V_m$ uma tabela de dados localizada no endereço $e = (H_1, \ldots, H_i, \ldots, H_n)$, e suponhamos que $H_i \leq H_i'$ na dimensão D_i. Seja h a função $h_{H_i, H_i'}$. Para todo $1 \leq j \leq m$, seja k_j o número de funções de agregação em $AGR(V_j, H_i)$. Então *o roll-up de TD na dimensão D_i de H_i para H_i' é a tabela*

$$TD[H_i \leq H_i'] \subseteq H_1 \times \cdots \times H_i' \times \cdots \times H_n \times V_1^{k_1} \times \cdots \times V_m^{k_m}$$

definida por:

$$(x_1, \ldots, x_i', \ldots, x_n, w_1, \ldots, w_m) \in TD[H_i \leq H_i']$$

se e só se

1. O excerto de TD denotado por $TD[x_1, \ldots, x_i', \ldots, x_n]$ e definido por

$$\{(x_1, \ldots, x_i, \ldots, x_n, v_1, \ldots, v_m) \in TD \quad | \quad h(x_i) = x_i'\}$$

   é não vazio.

2. Para todo $1 \leq j \leq m$, se $f_1, \ldots, f_{k_j}$ for a ordem de $AGR(V_j, H_i)$, então

$$w_j = (v_{j1}, \ldots, v_{jk_j})$$

onde, para todo $1 \leq l \leq k_j$,

$$v_{jl} = \underset{\vec{x} \in \tau}{\Phi_l} M_j(\vec{x})$$

em que Φ_l é a generalização de f_l a uma operação de aridade arbitrária e τ é a projeção de $TF[x_1, \ldots, x'_i, \ldots, x_n]$ às primeiras n colunas.

■

Continuando o exemplo, considere-se $TD = TF$ e a operação de *roll-up* de TF, na dimensão *Tempo*, a partir do atributo *Data* para o atributo *Semana*. Recorde-se que $h_{Data,Mês}$ foi definido atrás e que $AGR(VME, Semana) = \{SUM, MIN, MAX\}$. Então temos que:

$$TD[Data \leq Semana] \subseteq Semana \times Loja \times Código \times VME \times VME \times VME$$

e

$$\begin{aligned} TD[Data \leq Semana] = \{&(\text{"1"}, \text{"}LojaX\text{"}, \text{"}Boné\text{"}, (3,3,3)),\\ &(\text{"1"}, \text{"}LojaY\text{"}, \text{"}Brincos\text{"}, (5,5,5)),\\ &(\text{"1"}, \text{"}LojaZ\text{"}, \text{"}Brincos\text{"}, (5,5,5)),\\ &(\text{"1"}, \text{"}LojaX\text{"}, \text{"}Colar\text{"}, (9,4,5))\} \end{aligned}$$

Demonstremos, por exemplo, que:

$$(1, LojaX, Boné, (3,3,3)) \in TD[Data \leq Semana] \ .$$

Pela definição de *roll-up*, temos de verificar duas condições:

1. $TD[1, LojaX, Boné] \neq \emptyset$. Uma vez que $i = 1$ (dimensão *Tempo*) e $m = 1$, por existir apenas uma métrica, então

$$\begin{aligned} TD[1, LojaX, Boné] &= \{(x, LojaX, Boné, v) \in TD | h_{data,semana}(x) = 1\}\\ &= \{(1deJaneiro, LojaX, Boné, 3)\} \ . \end{aligned}$$

2. Como $m = 1$, tem-se necessariamente que $j = 1$. Para $j = 1$ tem-se

$$k_j = \#AGR(VME, Data) = \#\{SUM, MIN, MAX\} = 3$$

e

$$w_j = (3, 3, 3) \ .$$

De acordo com a ordenação do $AGR(VME, Data)$ definida na página 73, tem-se que:

$$f_1 = SUM \qquad f_2 = MIN \qquad f_3 = MAX \ .$$

O que queremos demonstrar (para $j = 1$) é que

(a)
$$\sum_{\vec{x} \in \tau} M_1(\vec{x}) = 3$$

(b)
$$\underset{\vec{x} \in \tau}{MIN}\, M_1(\vec{x}) = 3$$

(c)
$$\underset{\vec{x} \in \tau}{MAX}\, M_1(\vec{x}) = 3$$

onde τ é a projeção de $TD[Data \leq Semana]$ às 3 primeiras colunas (3 é o número de dimensões). Logo $\tau = \{(1deJaneiro, LojaX, Boné)\}$, pelo que:

(a)
$$\sum_{\vec{x} \in \tau} M_1(\vec{x}) = M_1(1deJaneiro, LojaX, Boné) = 3$$

(b)
$$\underset{\vec{x} \in \tau}{MIN}\, M_1(\vec{x}) = M_1(1deJaneiro, LojaX, Boné) = 3$$

(c)
$$\underset{\vec{x} \in \tau}{MAX}\, M_1(\vec{x}) = M_1(1deJaneiro, LojaX, Boné) = 3$$

Outro exemplo: demonstremos que

$$(1, LojaX, Colar, (9, 4, 5)) \in TD[Data \leq Semana] \ .$$

Novamente pela definição de *roll-up*, temos de verificar duas condições:

1. $TD[1, LojaX, Colar] \neq \emptyset$. Ora

$$\begin{aligned} & TD[1, LojaX, Colar] \\ =& \{(x, LojaX, Colar, v) \in TD | h_{data,semana}(x) = 1\} \\ =& \{(1deJaneiro, LojaX, Colar, 4), (2deJaneiro, LojaX, Colar, 5)\} \ . \end{aligned}$$

2. Novamente $j = 1$ e agora $w_j = (9, 4, 5)$. Queremos demonstrar que:

   (a)
   $$\sum_{\vec{x} \in \tau} M_1(\vec{x}) = 9$$

   (b)
   $$\underset{\vec{x} \in \tau}{MIN}\, M_1(\vec{x}) = 4$$

   (c)
   $$\underset{\vec{x} \in \tau}{MAX}\, M_1(\vec{x}) = 5$$

   onde agora $\tau = \{(1deJaneiro, LojaX, Colar), (2deJaneiro, LojaX, Colar\}$, pelo que:

   (a)
   $$\begin{aligned} &\sum_{\vec{x} \in \tau} M_1(\vec{x}) \\ =&M_1(1deJaneiro, LojaX, Colar) + M_1(2deJaneiro, LojaX, Colar) \\ =&4 + 5 \\ =&9 \end{aligned}$$

   (b)
   $$\begin{aligned} &\underset{\vec{x} \in \tau}{MIN}\, M_1(\vec{x}) \\ =&min\{M_1(1deJaneiro, LojaX, Colar), M_1(2deJaneiro, LojaX, Colar)\} \\ =&min\{4, 5\} \\ =&4 \end{aligned}$$

   (c)
   $$\begin{aligned} &\underset{\vec{x} \in \tau}{MAX}\, M_1(\vec{x}) \\ =&max\{M_1(1deJaneiro, LojaX, Colar), M_1(2deJaneiro, LojaX, Colar)\} \\ =&max\{4, 5\} \\ =&5 \end{aligned}$$

Proposição 3. *Seponhamos que $H_i \leq H_i'$ na dimensão D_i. Se TD está localizada em $e = (H_1, \ldots, H_i, \ldots, H_n)$, então $TD[H_i \leq H_i']$ está localizada em $e' = (H_1, \ldots, H_i', \ldots, H_n)$.*

Demonstração. Pela Definição 8, ao conhecermos TD a localização sabemos que

$$TD \subseteq H_1 \times \cdots \times H_i \times \cdots \times H_n \times V_1 \times \cdots \times V_m$$

em que:

a) $\forall 1 \leq j \leq n$, $H_j \in hrq(D_j)$.

b) $\forall 1 \leq j \leq m$, $V_j \in VAL$.

c) $\forall 1 \leq j \leq m$, a projeção de TD às colunas $H_1, \ldots, H_n, V_j$ é uma métrica.

Pela definição de *roll-up*, temos que:

$$TD[H_i \leq H_i'] \subseteq H_1 \times \cdots \times H_i' \times \cdots \times H_n \times V_1^{k_1} \times \cdots \times V_m^{k_m} \ .$$

Queremos demonstrar que $TD[H_i \leq H_i']$ está localizada em e'. Pela Definição 8, temos que provar que

a') $\forall 1 \leq j \leq n$, $G_j \in hrq(D_j)$, em que $G_j = \begin{cases} H_j & j \neq i \\ H_i' & j = i \end{cases}$

b') $1 \leq j \leq m, \forall 1 \leq l \leq k_j, W_{jl} \in VAL$, em que $W_{jl} = V_j$.

c') $\forall 1 \leq j \leq m, \forall 1 \leq l \leq k_j$, a projeção de $TD[H_i \leq H_i']$ às colunas $H_1, \ldots, H_i', \ldots, H_n, W_{jl}$ é uma métrica, isto é, uma função de tipo $H_1 \times \ldots H_i' \times \cdots \times H_n \hookrightarrow V_j$.

A demonstração destas três afirmações é a seguinte:

a') Para $j \neq i$, $G_j = H_j$, e $H_j \in D_j$ por a). Relativamente a $j = i$, temos de $G_j = G_i = H_i'$ que pertence também a D_i, por hipótese.

b') $W_{jl} = V_j$ e $V_j \in VAL$ por b).

c') Suponhamos que $(x_1, \ldots, x_i', \ldots, x_n, v_{jl})$ pertence à projeção de $TD[H_i \leq H_i']$ às colunas $H_1, \ldots, H_i', \ldots, H_n, W_{jl}$. Queremos mostrar que v_{jl} é determinado por $(x_1, \ldots, x_i', \ldots, x_n)$.

Por definição de *roll-up*, temos que $v_{jl} = \underset{x \in \tau}{\Phi_l} M_j(\vec{x})$, onde

- Φ_l é a generalização de f_l a uma operação de aridade arbitrária, em que f_l é a l-ésima função de agregação em $AGR(V_j, H_i)$.
- τ é a projeção às primeiras n colunas de $TD[x_1, \ldots, x_i', \ldots, x_n]$, em que esta última é a tabela $\{(x_1, \ldots, x_i, \ldots, x_n, v_1, \ldots, v_m) \in TD \,|\, h(x_i) = x_i'\}$.
- M_j é a projeção de TD às colunas $H_1, \ldots, H_n, V_j$.

Como $h = h_{H_i, H_i'}$, h está determinado por $T(D_i)$. Portanto, τ é determinado por TD, $x_1, \ldots, x_i, \ldots, x_n$ e $T(D_i)$. j e l determinam f_l. TD e j determinam M_j. Logo, TD, $x_1, \ldots, x_i, \ldots, x_n$, $T(D_i)$, j e l determinam v_{jl}.

□

No exemplo anterior, $TD[Data \leq Semana]$ está localizada em $e = (Semana, Loja, Codigo)$.

A proposição anterior mostra que é possível iterar a operação de *roll-up*, por exemplo:

$$TD \rightsquigarrow TD[H_i \leq H_i'] \rightsquigarrow TD[H_i \leq H_i'][H_j \leq H_j']$$

Em particular, podemos definir Cubo de Dados.

Definição 10 (Cubo de dados).

1. A *materialização de TF no endereço $e = (H_1, \ldots, H_n)$* é a tabela:
$$TF[min(D_1) \leq H_1] \ldots [min(D_n) \leq H_n]$$
2. O *Cubo de Dados* é a função CD que associa a cada endereço $e = (H_1, \ldots, H_n)$ a materialização de TF no endereço e.

■

5.4 Questões em aberto

Um propriedade elementar desejável para a definição de Cubo de Dados é a seguinte: a materialização no endereço mínimo é igual à tabela de factos ou seja,

$$TF[min(D_1) \leq min(D_1)] \ldots [min(D_n) \leq min(D_n)] = TF \ .$$

Por falta de tempo não pudemos verificar esta propriedade.

A iteração da operação de *roll-up* levanta várias questões, nomeadamente:

1. Se $i \neq j$, $TD[H_i \leq H_i'][H_j \leq H_j'] = TD[H_j \leq H_j'][H_i \leq H_i']$?
2. Se $H_i \leq H_i' \leq H_i''$ na dimensão D_i:
$$TD[H_i \leq H_i'][H_i' \leq H_i''] = TD[H_i \leq H_i'']?$$

Uma resposta afirmativa à pergunta 1 terá como consequência que a definição de materialização não depende da ordem pela qual se fazem as n operações de *roll-up*. Uma resposta afirmativa à segunda questão terá como consequência que a definição de materialização no endereço e não depende do caminho entre o endereço mínimo e o endereço e no EMD.

Para além do *roll-up*, há outras operações comuns em sistemas OLAP, como por exemplo a operação *Drill-Down*, que gostaríamos de ver refletidas neste modelo.

Capítulo 6

Conclusões e Trabalho Futuro

Os sistemas OLAP continuam a apresentar características muito específicas, demasiadamente direcionadas a problemas muito concretos de empresas que desenvolvem as suas atividades de exploração e análise de dados de forma muito própria. Talvez esta situação ainda ocorra pela pouca flexibilidade dos modelos existentes que suportam os sistemas OLAP ou simplesmente por uma questão tão simples quanto a falta de conhecimento. Porém, estas situações estão a ser gradualmente eliminadas, não só pelo esforço dos produtores de software analítico, que se direcionam cada vez mais para o desenvolvimento de produtos mais abrangentes, como também pela necessidade cada vez maior de quadros especializados com capacidade para utilizarem estes sistemas.

O modelo proposto neste documento é bastante conceptual, no entanto, a sua formalização é, na nossa opinião, um verdadeiro ponto de partida para a construção de uma qualquer interface que de uma forma abrangente, mas efetiva, defina concretamente os vários problemas encontrados nestes sistemas. A proposta de formalização apresentada tem capacidade para abranger inúmeras situações práticas reais, que outros modelos não o conseguem fazer. Por exemplo, o facto de se poder acrescentar informação relativa aos tipos de agregações, e em que situações é possível agregar cada métrica, são para nós vantagens importantes no contexto dos sistemas de processamento analítico.

A definição dos reticulados relativos aos espaços multidimensionais de dados, sob a forma de conjuntos expressos formalmente, é algo que na prática, aquando do projeto de um sistema OLAP, não é muito viável. No entanto, a sua representação gráfica tem o mesmo valor em termos matemáticos e poderá constituir algo bastante vantajoso para a validação final do sistema, uma vez que as restrições básicas dos reticulados exigem que, por exemplo, uma dimensão da estrutura seja corretamente definida em termos das hierarquias que suportam.

Como consequência direta, tais restrições permitem que a definição das estruturas multi-dimensionais de dados – os hipercubos – seja matematicamente correta, garantindo uma formalização sem falhas.

Apesar de um conjunto parcialmente ordenado genérico poder ser uma solução para a situação dos atributos descritivos, a falta de restrições que se verifica ao nível da ordenação desses conjuntos genéricos faz perder qualquer garantia de ordenação, o que pode não ser uma solução ótima se não se conseguir arranjar alternativas viáveis. O facto de apenas termos definido a operação *roll-up*, faz com que tenhamos as próximas etapas bem deliniado ao nível do modelo formal. Para além disso, a forte definição matemática deste modelo é uma excelente base para uma possível interface de construção de modelos conceptuais capazes de serem facilmente traduzidos em modelos lógicos prontos a serem utilizados no mundo real. Este é portanto um dos maiores objetivos para o trabalho futuro.

Apêndice A

Representação de uma *lattice*

De uma forma mais alternativa à Figura 2.4, uma vez que cada cuboide da *lattice* pode ser visto como um cubo, então uma possível representação é a da Figura A.1

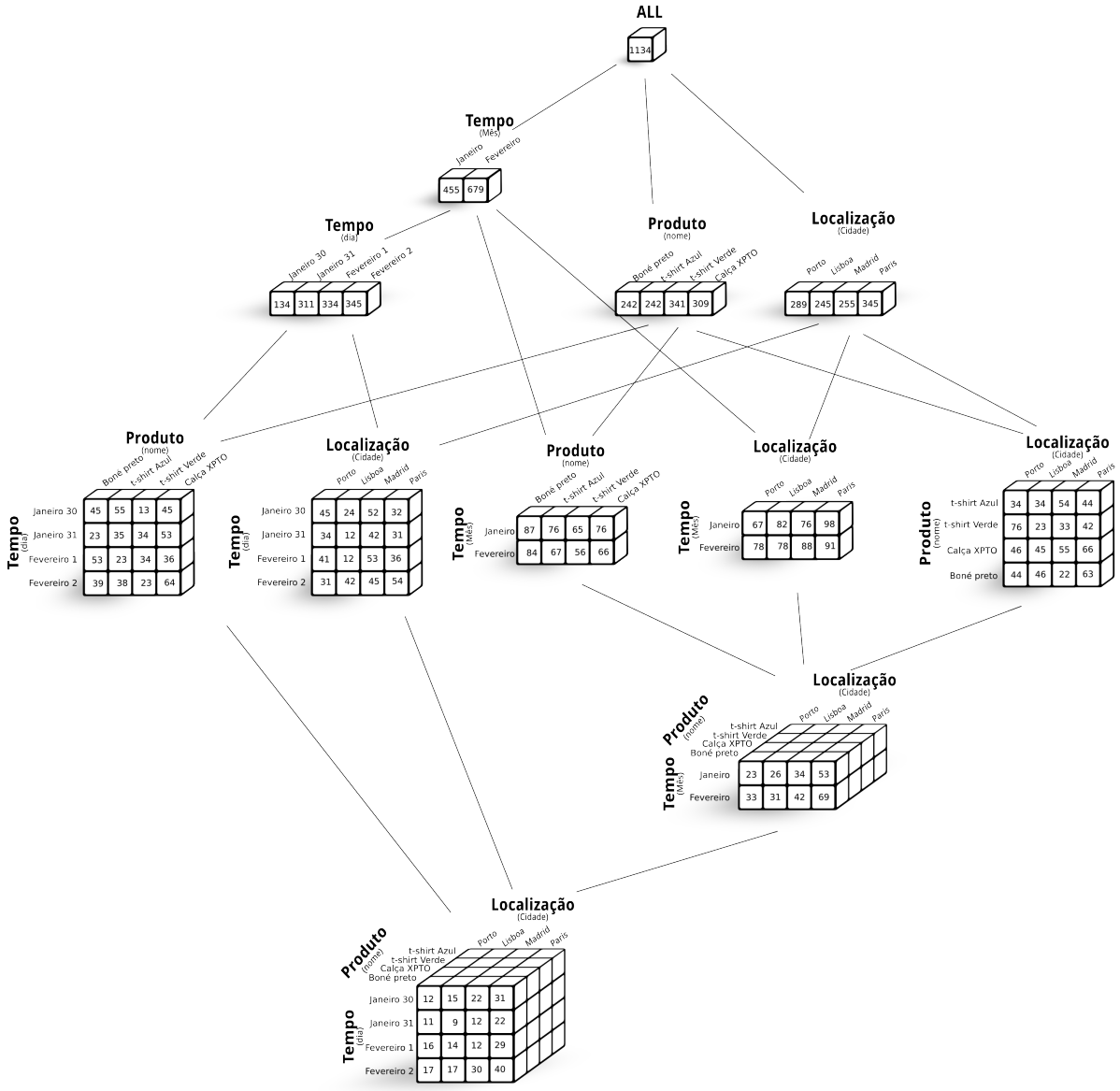


Figura A.1: Uma *lattice* é um conjunto de cuboides correlacionadas numa hierarquia

Apêndice B

Produto entre 3 dimensões

De seguida é ilustrado um exemplo de como se pode fazer o produto entre dimensões (reticulados). Este exemplo utiliza as 3 dimensões apresentadas na Figura 5.1, da página 69, em que uma delas é uma simples dimensão com um atributo de hierarquia, outra tem uma simples hierarquia de dois atributos e, por fim, uma hierarquia ramificada.

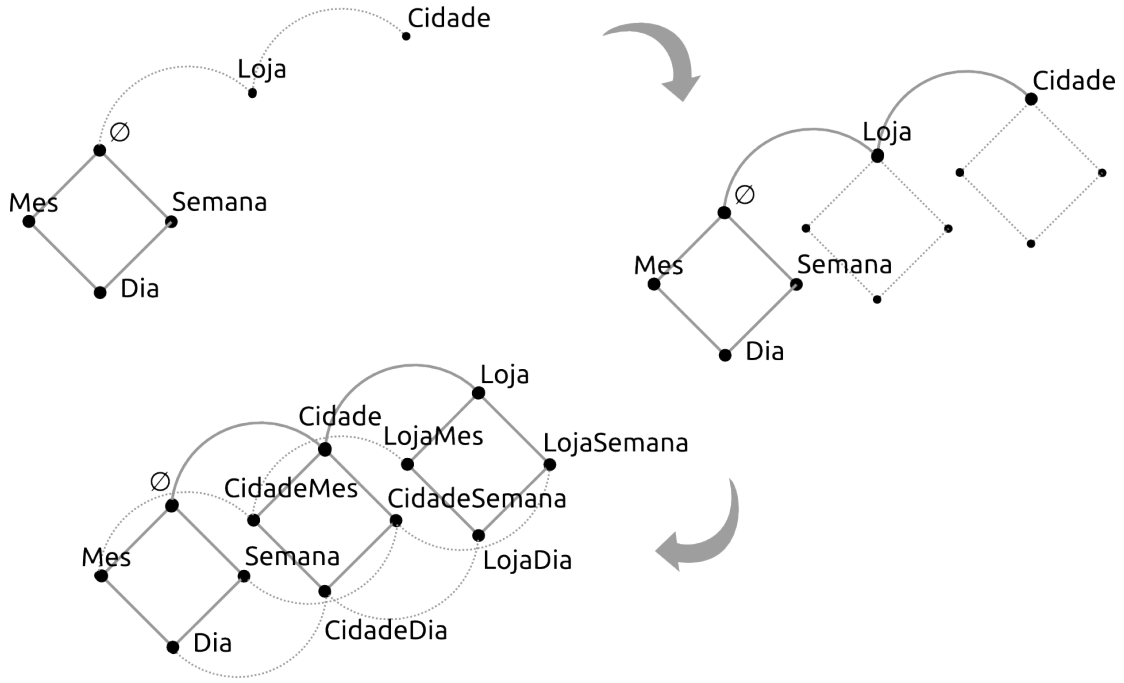


Figura B.1: Inicialmente o produto é feito entre duas dimensões

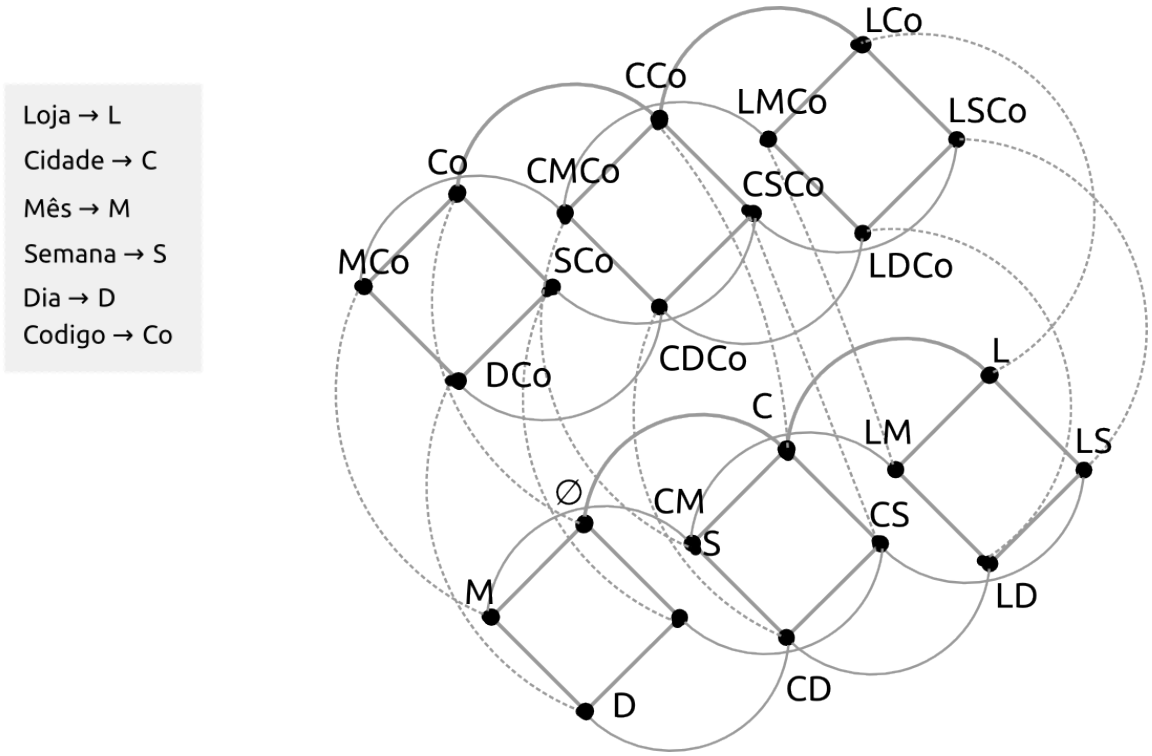


Figura B.2: Resultado final do produto entre 3 dimensões, que pode ter uma aparência alternativa semelhante à da Figura 5.2

Lista de Acrónimos

BI *Business Intelligence*

CPO Conjunto Parcialmente Ordenado

DFM *Dimensional Fact Model*

DM Data Mart

DW Data Warehuose

EMD Espaço Multi-Dimensional

FS *Fact Scheme*

HOLAP *Hybrid OLAP*

MOLAP *Multidimensional OLAP*

OLAP *On-Line Analytical Processing*

OLTP *On-Line Transaction Processing*

SQL *Structured Query Language*

SSD Sistema de Suporte à Decisão

ROLAP *Relational OLAP*

TS *Table Schema*

Bibliografia

Agarwal, Sameet, Agrawal, Rakesh, Deshpande, Prasad M., Gupta, Ashish, Naughton, Jeffrey F., Ramakrishnan, Raghu, e Sarawagi, Sunita. On the computation of multidimensional aggregates. In *In Proceedings Of The International Conference On Very Large Databases*, pages 506–521, 1996.

Cabibbo, Luca e Torlone, Riccardo. From a procedural to a visual query language for OLAP. In *Proceedings of the 10th International Conference on Scientific and Statistical Database Management*, SSDBM '98, pages 74–83, Washington, DC, USA, 1998. IEEE Computer Society. ISBN 0-8186-8575-1. doi: http://dx.doi.org/10.1109/SSDM.1998.688113.

Chaudhuri, Surajit e Dayal, Umesh. An overview of data warehousing and OLAP technology. *ACM Sigmod record*, 26(1), 1997.

Chaudhuri, Surajit, Dayal, Umeshwar, e Ganti, Venkatesh. Database technology for decision support systems. *Computer*, 34(12):48–55, 2001.

Davey, Brian A. e Priestley, Hilary A. *Introduction to Lattices and Order (2. ed.)*. Cambridge University Press, 2002. ISBN 978-0-521-78451-1.

Demuth, Birgit e Hussmann, Heinrich. Using uml/ocl constraints for relational database design. In *Proceedings of the 2nd international conference on The unified modeling language: beyond the standard*, UML'99, pages 598–613, Berlin, Heidelberg, 1999. Springer-Verlag. ISBN 3-540-66712-1.

Golfarelli, Matteo e Rizzi, Stefano. *Data warehouse design: Modern principles and methodologies*. McGraw-Hill, 2009.

Golfarelli, Matteo, Maio, Dario, e Rizzi, Stefano. The dimensional fact model: A conceptual model for data warehouses. *International Journal of Cooperative Information Systems*,

7:215–247, 1998.

Gray, Jim, Chaudhuri, Surajit, Bosworth, Adam, Layman, Andrew, Reichart, Don, Venkatrao, Murali, Pellow, Frank, e Pirahesh, Hamid. Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals. *Data Min. Knowl. Discov.*, 1 (1):29–53, 1997. ISSN 1384-5810. doi: 10.1023/A:1009726021843.

Gyssens, Marc e Lakshmanan, Laks V. S. A foundation for multi-dimensional databases. In *Proceedings of the 23rd International Conference on Very Large Data Bases*, VLDB '97, pages 106–115, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc. ISBN 1-55860-470-7.

Harinarayan, Venky, Rajaraman, Anand, e Ullman, Jeffrey D. Implementing data cubes efficiently. *SIGMOD Rec.*, 25:205–216, June 1996. ISSN 0163-5808. doi: http://doi.acm.org/10.1145/235968.233333.

Inmon, William H. *Building the data warehouse*. John Wiley & Sons, 2005. ISBN 9780764599446.

Kimball, Ralph e Ross, Margy. *The data warehouse toolkit: the complete guide to dimensional modeling*. Wiley, 2002. ISBN 9780471200246.

Lax, Peter D. *Linear algebra and its applications*. Number vol. 10 in Pure and applied mathematics. Wiley-Interscience, 2007. ISBN 9780471751564.

Meulen, Rob e Rivera, Janessa, 2013. Gartner Says Worldwide Business Intelligence Software Revenue to Grow 7 Percent in 2013. [online] Disponível em: <http://www.gartner.com/newsroom/id/2340216> [Acedido a 10-Abril-2013].

Morfonios, Konstantinos, Konakas, Stratis, Ioannidis, Yannis, e Kotsis, Nikolaos. Rolap implementations of the data cube. *ACM Comput. Surv.*, 39, November 2007. ISSN 0360-0300. doi: http://doi.acm.org/10.1145/1287620.1287623.

Pardillo, Jesús, Mazón, Jose-Norberto, e Trujillo, Juan. Bridging the semantic gap in OLAP models: platform-independent queries. In *Proceedings of the ACM 11th international workshop on Data warehousing and OLAP*, DOLAP '08, pages 89–96, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-250-4. doi: http://doi.acm.org/10.1145/1458432.1458448.

Romero, Oscar e Abelló, Alberto. On the need of a reference algebra for OLAP. In *DaWaK*, pages 99–110, 2007.

Thomas, Helen e Datta, Anindya. A conceptual model and algebra for on-line analytical processing in decision support databases. *Info. Sys. Research*, 12:83–102, March 2001. ISSN 1526-5536. doi: http://dx.doi.org/10.1287/isre.12.1.83.9715.