PROCESSAMENTO ANALÍTICO EM DADOS XML

XML DATA ANALITICAL PROCESSING

Paulo Caetano da Silva
paulo.caetano@bcb.gov.br
Salvador University – UNIFACS, Brazil

Mateus Silqueira Hickson Cruz
mateus@ceusystem.com.br
Ceu System, Brazil

Resumo: O uso de ferramentas de processamento analítico de dados (OLAP) para realização de análises estratégicas de uma organização possibilita que usuários responsáveis pela tomada de decisões possam identificar tendências e padrões, de forma a conduzir melhor o negócio da empresa em que atuam. Entretanto, o desenvolvimento de sistemas de processamento analítico em dados XML nos meios acadêmico e comercial não possui todas as funcionalidades das ferramentas OLAP para dados tradicionais e também não contempla documentos XML interdependentes. Portanto, a necessidade de desenvolver sistemas OLAP para auxiliar nas análises estratégicas dos dados de uma organização, representados no formato XML e interligados por um conjunto de referências, constitui a principal motivação para o desenvolvimento deste trabalho. Atualmente, pesquisas vêm sendo desenvolvidas no contexto acadêmico com o objetivo de realizar processamento analítico em dados representados em XML. No entanto, em razão destas tecnologias terem sido originalmente concebidas para propósitos distintos, esta não é uma tarefa trivial. Para ajudar no desenvolvimento desses sistemas OLAP, neste trabalho são discutidos os desafios que devem ser resolvidas para a realização de um processamento analítico eficiente sobre dados XML e avaliados alguns trabalhos acadêmicos que se propõe a realizar esta tarefa.

Palavras-chave: XML; OLAP; XOLAP; XML-OLAP; XLink.

Abstract: The use of tools for data analytical processing (OLAP) to perform the strategic analysis of an organization allows users responsible for making the decisions to identify trends and patterns in order to better conduct the business of the company where they work. However, the academic and commercial development of analytical processing systems for XML data does not have all the features of the existing OLAP tools for traditional data and interrelated XML documents. Therefore, the need for developing OLAP systems to assist in the strategic analysis of data from one organization, represented in XML format and interconnected by a set of references, is the main motivation for developing this work. Currently, research is being developed in the academic context in order to perform analytical processing of data represented as XML. However, because these technologies were originally designed for different purposes, this is not a trivial task. To assist in the development of OLAP systems, this paper discusses the challenges that must be resolved to conduct an efficient analytical processing over XML data and evaluated some academic work that intends to accomplish this task

Keywords: XML; OLAP; XOLAP; XML-OLAP; XLink.

1 INTRODUÇÃO

Utilizando-se linguagens de marcação derivadas de XML, é possível representar informações semanticamente semelhantes de diferentes formas. Isto leva a três tipos de heterogeneidade dos dados em XML: (i) semântica, na qual informações similares são representadas por diferentes nomes (e.g. empresa e companhia) ou quando informações dissimilares possuem nomes iguais (e.g. ativo na área contábil e na área de saúde); (ii) sintática de conteúdo, de modo que o conteúdo semanticamente igual é representado de diversas maneiras, como em idiomas diferentes ou em unidades de medidas diversas (metros, pés, moedas, ºC, ºF); e (iii) estrutural, na qual o dado é organizado em diversas estruturas (e.g. em diferentes tipos de hierarquias, atributos ou elementos) [62]. Esta flexibilidade de representação é importante, pois permite a criação de linguagens de marcação para domínios específicos, porém torna a utilização de conceitos OLAP sobre dados XML uma tarefa complexa. Por exemplo, as linguagens de consultas XML, como XPath (*XML Path Language*) [96] e XQuery (*XML Query Language*) [98], geralmente são orientadas a caminho, requerendo do usuário o conhecimento prévio sobre a estrutura dos documentos a serem processados.

Aplicações e tecnologias derivadas de XML usam documentos interligados para representação da semântica e da estrutura da informação, expressando relacionamentos entre conceitos, normalmente definidos em um esquema de dados semi-estruturados. Informações sobre os conceitos relatados no documento

XML (e.g. sobre hierarquia entre eles) podem não estar presentes no próprio documento que contém os dados, sendo mantidas em arquivos que possuem *linkbases* (agrupamento de *links*). Um dos problemas que ocorre no processamento de documentos que possuem ligações com outros documentos, está no fato das linguagens padrões de consultas (i.e. XPath, XQuery) disponíveis atualmente não proverem suporte para navegação entre as referências existentes nos documentos. Por exemplo, apesar de a linguagem XPath ser adotada como padrão para consulta em documentos XML, ela não provê suporte à navegação em *links*, dificultando a manipulação e o acesso a documentos que fazem uso deste recurso. Apesar da existência de linguagens para navegação em *links*, elas possuem restrições que dificultam a realização de processamento analítico em documentos XML interligados. Logo, é importante desenvolver soluções para o processamento analítico de dados em documentos XML que fazem uso de *links*, ou seja, documentos que se interligam com outros por meio de referências. Um exemplo de tecnologia derivada de XML que utiliza *links* para expressar informações é a linguagem XBRL (*eXtensible Business Reporting Language*) [30], um padrão tecnológico para representação, intercâmbio e publicação de informações financeiras adotado por diversas organizações. Desta forma, os problemas de heterogeneidade que ocorrem em dados no formato XML e as informações distribuídas entre documentos XML que contêm ligações com outros documentos tornam o processamento analítico sobre este tipo de dados diferente do que é realizado pelas tradicionais ferramentas OLAP disponíveis atualmente no mercado e na academia.

Diversos trabalhos vêm sendo desenvolvidos com o propósito de realizar consultas analíticas em dados XML [11, 12, 44, 65, 66, 67, 86, 87, 88, 89]. Alguns destes ambientes de consultas OLAP sobre dados XML estão sendo referidos por alguns autores como XOLAP (OLAP para XML) [87, 89] ou XML-OLAP [65]. Entretanto, nenhum desses trabalhos considera o uso de documentos XML interligados. Uma linguagem de consulta multidimensional para dados XML que considere informações adicionais estabelecidas em relacionamentos entre documentos possibilitará a recuperação de informações possivelmente importantes para o processo de tomada de decisão.

Este artigo está organizado da seguinte forma: a Seção 2 trata dos conceitos fundamentais sobre dados semi-estruturados e OLAP. Alguns desafios que devem ser superados para a realização de consultas OLAP em documentos XML são discutidos na Seção 3. Na Seção 4 são apresentados trabalhos relacionados a OLAP em XML. Por fim, as considerações finais são expostas na Seção 5.

2 CONCEITOS BÁSICOS

Esta seção descreve alguns conceitos básicos relacionados ao desenvolvimento deste trabalho. Com relação a dados semi-estruturados, na Seção 2.1.1 são discutidas tecnologias usadas para descrever e manipular dados representados por XML. Sistemas de processamento analítico são abordados na Seção 2.1.2.

2.1 Dados Semi-Estruturados

Dados semi-estruturados possuem uma representação estrutural heterogênea, ou seja, em alguns casos os dados possuem uma descrição uniforme; em outros, algum padrão estrutural pode ser identificado; ou então, praticamente não existem informações descritivas associadas. Além disso, os dados podem ser autodescritivos, não existindo descrição separada do tipo ou da estrutura dos dados [2, 28].

XML é considerada uma metalinguagem adequada para a descrição e representação de dados semi-estruturados [2]. Uma instância XML (documento que contém os dados) pode ser validada em um esquema de dados. Um esquema é a representação da organização dos dados, definindo tipos, relacionamentos e restrições [28, 37]. O esquema descreve a estrutura e restringe o conteúdo dos documentos de instâncias XML. Além disso, uma instância XML pode fazer uso de mais de um esquema de dados [97]. Como uma instância XML pode ser estruturada por mais de um esquema, existe um encadeamento de ligações entre a instância e os seus esquemas. Assim, é necessária a navegação pelos esquemas para a validação do documento de instância XML. Entre as tecnologias usadas para criar esquemas para documentos XML, destacam-se DTD (*Document Type Definition*) [74] e XML *Schema* (uma linguagem padrão do W3C (*Word Wide Web Consortium*) [97]), que possibilitam a definição de gramáticas para linguagens baseadas em XML.

A relação entre elementos XML e documentos XML ocorre por meio de *links*. Um *link* é um mecanismo utilizado para associar dois ou mais recursos. Os *links* usados em XML podem estar contidos no próprio documento onde estão os recursos ou em outro documento, denominado *linkbase*, que contém uma coleção de *links*. Os *links* associam recursos locais e

remotos. Um recurso local é um elemento XML que participa de uma ligação em virtude dele, ou do elemento que o contém (elemento pai), ser um elemento de ligação. Já um recurso remoto participa de uma ligação em razão de ser endereçado por uma URI (*Universal Resource Identifier*). Um recurso local é especificado por valor e um recurso remoto por referência [95]. Uma tecnologia usada para estabelecer *links* entre dados representados em XML é XLink (*XML Linking* Language) [95], que define dois tipos principais de *links*: os simples e os estendidos.

Um *link* simples associa exatamente dois recursos, um local e um remoto. Essa associação cria um arco de ligação entre eles, cuja origem é o recurso local e o destino é o recurso remoto. Os arcos são representados por elementos que indicam os recursos participantes da ligação.

Por sua vez, os *links* estendidos permitem associar um número arbitrário de recursos participantes na ligação. Um *link* estendido consiste basicamente de um elemento XML que contém outros elementos, nos quais atributos especificados por XLink são declarados, conferindo a estes sub-elementos determinadas funcionalidades. XLink provê quatro tipos de sub-elementos: (i) *locator*, usado para referenciar recursos remotos por meio de uma URI; (ii) *resource*, usado para encapsular informações no elemento de *link* estendido; (iii) *arc*, usado para estabelecer relações direcionais entre pares de elementos *locator* ou *resource*; e (iv) *title*, que provê informações descritivas a respeito do *link*, a fim de serem entendidas por pessoas.

A utilização em conjunto de tecnologias para definir a estrutura e relacionamentos entre instâncias XML forma uma rede de documentos XML. A Figura 1.1 ilustra como pode ocorrer tal rede através do uso das tecnologias XML *Schema* e XLink. Uma instância pode apontar para um esquema, que por sua vez pode apontar para outros esquemas. Conjuntos de esquemas podem referenciar *linkbases*, e estes podem referenciar outros *linkbases*. Além disso, a instância e os esquemas também podem possuir *links* internos. Assim, um encadeamento de documentos é formado, sendo necessária a navegação por eles para se encontrar informações adicionais. Portanto, percebe-se que é possível navegar de uma instância XML para esquemas ou para *linkbases*, ou entre esquemas, ou entre *linkbases*. Deste modo, *links* internos e externos aos documentos XML podem ser criados. Essa estrutura permite a definição de relacionamentos entre os elementos, servindo de informação complementar àquelas presentes nas instâncias XML.

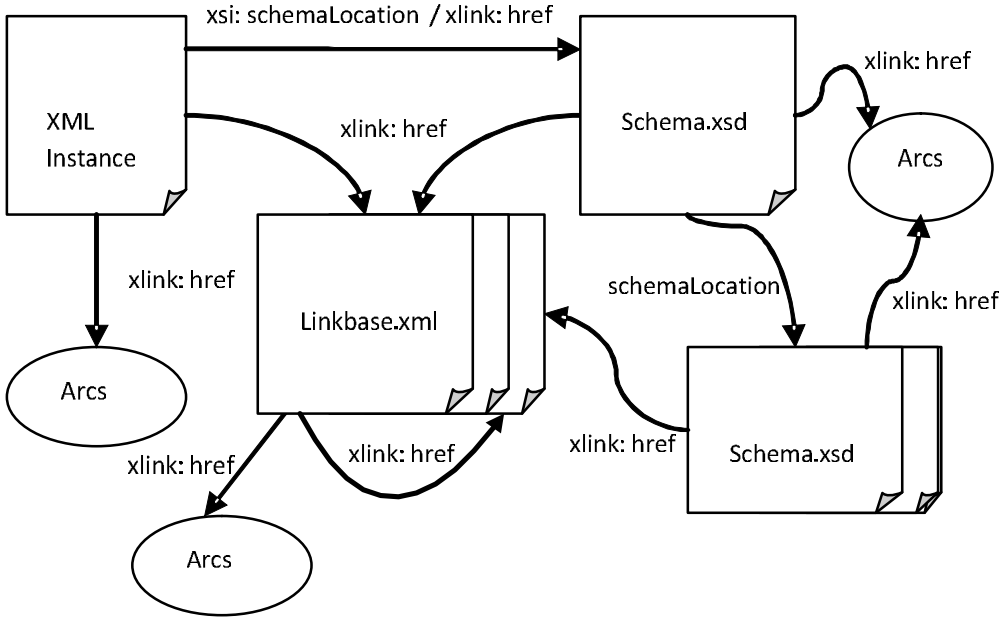

Figura 1.1 - Rede de documentos XML.

Uma forma de representação conceitual de um documento XML é uma árvore, na qual cada parte do documento representa um nó, sendo que cada nó pode conter um número ilimitado de outros nós. Além disso, há exatamente um único nó, chamado de raiz, que contém todos os outros nós. A partir desta representação, linguagens de consulta para dados semi-estruturados são usadas para capturar nós (ou grupos deles) de tal árvore. Essa categoria de linguagens faz uso de expressões de caminho, chamadas *location path*, que especificam como navegar em uma árvore XML de um nó a outro. Uma *location path* é composta por passos, que por sua vez contêm um *eixo*, uma *especificação de nó* e um *predicado*, sendo que este último é opcional e consiste de testes condicionais para filtrar o resultado da consulta. Para localizar um nó específico em um documento XML, podem ser combinados vários passos, sendo cada um deles avaliado em relação a um nó em particular, chamado de nó de contexto.

XPath é uma linguagem de consulta padrão do W3C para endereçar partes de um documento XML, ou valores computacionais (e.g. *strings*, números ou valores booleanos) que fazem parte do conteúdo de um documento XML. Sendo uma linguagem de consulta para dados semi-estruturados, ela explora o contexto hierárquico e seqüencial de elementos de um documento XML. XPath é utilizada por outras tecnologias (e.g. XQuery, XPointer [38] e XSLT [99]) como o mecanismo básico para endereçamento de nós XML. Entretanto, XPath não fornece recursos para a navegação por um conjunto de documentos interligados. Outra linguagem de consulta padrão do W3C é XQuery, que é baseada em XPath e foi desenvolvida para consultas a documentos XML, permitindo o acesso a partes específicas destes, tais como nós ou conjuntos de nós. Ela permite acessar as informações armazenadas em arquivos XML de maneira similar às consultas SQL [23, 27]. Existem outras linguagens de consultas a dados semi-estruturados [19, 25, 76, 7], mas, além de não serem reconhecidas como padrões pelo W3C, nenhuma delas oferece recursos para navegação por meio de *links*, restringindo a possibilidade de obtenção de informação adicional por meio de conexões entre documentos e elementos XML.

2.2 Processamento Analítico On-Line

OLAP é um acrônimo para *On-Line Analytical Processing*, ou seja, processamento analítico em tempo real, e diz respeito a aplicações de banco de dados que permitem aos usuários (analistas, gerentes e executivos) visualizar, navegar, manipular e analisar bancos de dados multidimensionais. Dado multidimensional se refere ao fato quantificado por um conjunto de medidas, obtidas a partir da aplicação de uma função de agregação (e.g. de contagem, soma, média) aos dados operacionais de uma organização. Tais medidas são caracterizadas por um conjunto de variáveis, chamadas de dimensões. A medida é uma propriedade numérica do fato e a dimensão é uma das coordenadas de análise do fato. As dimensões fornecem diferentes visões sobre o fato. Dimensões típicas para um fato "quantidade vendida" podem ser "produto", "cliente" e "data" [35, 73, 81, 90].

OLAP é a tecnologia que permite ao usuário extrair e visualizar informações de um banco de dados de forma seletiva e sob diferentes pontos de vista [81]. Desta maneira, sistemas OLAP são uma categoria de software específica para realizar processamento analítico (i.e. consultas de suporte à decisão) sobre os dados de um *data warehouse* (DW) [35, 42, 45], de forma que este processamento deve: (i) ocorrer com alto desempenho, consistência e interatividade e (ii) auxiliar a tomada de decisão em uma organização por meio da interpretação desses dados com uma variedade de visões multidimensionais (i.e. dimensões). Essas visões multidimensionais são geradas a partir da navegação sobre a estrutura dimensional do *data warehouse* e podem ser compreendidas como eixos e pontos de um espaço multidimensional, no qual cada eixo pode ser visto como uma dimensão ou perspectiva (e.g. tempo, área geográfica, tipo de transação financeira), e os pontos como um valor medido e correspondente à interseção desses eixos [34].

A representação da realidade, construída utilizando este modelo dimensional, é formada a partir de um conjunto de conceitos. Os conceitos básicos modelados são fatos, medidas, dimensões e hierarquias [73, 81, 90]. A partir desses conceitos, um software OLAP extrai os dados da fonte, que normalmente está modelada como um cubo multidimensional [14]. Um cubo é uma representação virtual de um conjunto de dados organizados em uma estrutura multidimensional

definida por dimensões e medidas. Diversas operações sobre o cubo podem ser efetuadas, como *Drill Down* e *Drill Up/Roll Up*, as quais permitem o aumento e a diminuição do nível de detalhe de uma medida [73, 81, 90].

3 CONSULTAS OLAP EM XML: DESAFIOS

A utilização de XML para integração de fontes heterogêneas de dados tem tornado esta tecnologia um padrão de fato para o intercâmbio de dados. Desta forma, documentos XML podem ser uma rica fonte de informações para a tomada de decisão organizacional. Semelhantemente, o uso de sistemas de *data warehouse* [42, 45] e ferramentas OLAP (*On-Line Analytical Processing*) [20, 21, 81] possibilitam aos usuários tomarem decisões. Por isso, o desenvolvimento de sistemas OLAP para auxiliar nas análises de dados representados por XML compõe um desafio a ser enfrentado.

A análise sobre dados XML difere da análise OLAP tradicional tanto no modelo de dados, quanto nos padrões de consulta [14]. Algumas dessas diferenças são discutidas a seguir.

Diferenças no modelo de dados:

- *Conteúdo rico semanticamente* - Um documento XML é um texto cujo conteúdo é delimitado por estruturas delimitadoras (*tags*) autodescritivas e hierárquicas. Um documento XML deve seguir as regras da metalinguagem XML para ser considerado um documento bem formado e deve respeitar gramáticas especificadas por meio de DTD ou XML *Schema* para ser um documento válido. Desta maneira, XML pode ser usada para desenvolver diferentes vocabulários em domínios específicos, que podem codificar o conteúdo via a semântica das *tags* e codificar relações inerentes entre os conceitos por meio das hierarquias das *tags* ou de *linkbases*. Para fins analíticos, os nós de uma árvore XML podem ser vistos como membros de dimensões. Os membros de uma dimensão estão relacionados uns aos outros por meio da estrutura hierárquica de XML, ou por relacionamentos descritos por meio de *links*. No OLAP tradicional, as medidas são classificadas ou agrupadas pelos níveis e atributos da dimensão, cuja organização estrutural é fixa. Já em XML, qualquer elemento pode estar associado a um conjunto de atributos que fornece informações adicionais sobre este elemento. Tais informações podem também ser utilizadas para efeitos de análise. Logo, em razão da flexibilidade de representação das informações por meio de elementos, subelementos ou atributos, o processamento analítico sobre os dados pode ser executado de diversos modos;
- *Dados semi-estruturados* - Documentos XML bem formados estão em conformidade com uma representação em forma de árvore, cujos nós são ordenados e podem ter hierarquias com diferentes membros. Diferentemente do OLAP tradicional, que possui uma estrutura hierárquica definida e única para um mesmo cubo, o contexto de um dado XML é definido pela hierarquia em que ele está inserido. Sendo assim, um fato analisado pode aparecer em mais de um contexto (ou hierarquias), e uma operação analítica sobre uma medida pode não ser aplicável para a mesma medida em outro contexto. Finalmente, uma vez que nós XML estão ordenados em um documento, as medidas podem ser semanticamente relacionadas por este ordenamento; e
- *Endereçamento para navegação* - A navegação na árvore que representa o documento XML normalmente é realizada utilizando linguagens de consulta que usam expressões de caminho. Isto ocorre por meio de relações entre nós pai-filho e entre nós irmãos. Assim, qualquer nó de uma árvore XML pode ser endereçado sob diversas formas, inclusive por meio de *links*.

Diferenças nos padrões de consulta:

- *Consultas dependentes da ordem* - A linguagem XML ordena os nós do documento. O nó raiz é considerado pelas linguagens de consulta para dados semi-estruturados como a posição base em consultas sobre a representação em árvore do documento XML para, por exemplo, identificar o primeiro filho de um nó. Similarmente, consultas baseadas na posição, poderão ser utilizadas para analisar conjuntos ordenados de dados cuja ordenação exerce alguma semântica;
- *Agrupamento baseado no caminho* - Operações OLAP tradicionais (e.g. *Roll Up*) agrupam tuplas de uma relação baseada em valores de seus atributos. Em XML, pode-se também agrupar conceitos com base no caminho dos atributos ou por meio de padrões de árvores que expressam

relacionamentos entre os conceitos. O caminho dos atributos ou padrões generalizados de caminho podem ser especificados por meio de expressões de caminho;

- *Consultas baseadas na estrutura* - Devido à natureza semi-estruturada de documentos XML, análises com base em nome de *tags* e especificações de caminho são afetadas por mudanças na estrutura, erros ortográficos ou renomeações. Por exemplo, pode ser feito uso do sobrenome, em vez do último nome numa expressão de consulta. Do mesmo modo, a ordem de uma *tag* na especificação de um caminho pode ser invertida, como em *//pedido/cliente/endereço* e *//cliente/pedido/endereço*. Nesses casos, similaridades semântica [75] e sequencial [39] são importantes para o casamento de padrões, que é uma forma de expressão usada para determinar se um objeto específico corresponde a um dado critério. Por exemplo, ocorre casamento em *capitulo//paragrafo* quando o elemento corrente é *paragrafo*, o qual está dentro de um elemento *capitulo*;
- *Consultas Slice e Dice* - Em um sistema OLAP tradicional, fatiamento (*slicing*) [81] envolve reduzir dimensões de um cubo de dados e, em seguida, projetar este cubo usando a dimensão reduzida. Equivalentemente, uma árvore XML pode ter dimensões eliminadas pela seleção de subárvores particulares. Do mesmo modo, a operação *dice* [81] identifica e elimina subárvores baseada em valores derivados de propriedades estruturais (por exemplo, profundidade de um nó XML) ou valores de nó; e
- *Análise estrutural* - Nos sistemas OLAP tradicionais, análises do tipo *what-next* têm sido amplamente utilizadas para prever tendências futuras. Análises desta natureza envolvem a modificação de algumas medidas e o estudo dos seus efeitos sobre o conjunto dos dados utilizando diferentes funções de agregação. Na análise sobre XML, pode-se avaliar o impacto das relações alterando as estruturas dos dados XML.

4 PROCESSAMENTO ANALÍTICO EM DADOS XML

A base de dados mais comumente utilizada para a realização de consultas OLAP é a de *data warehouse*. Seus benefícios já foram amplamente discutidos e divulgados na literatura científica [21, 42, 45, 46, 47]. Do mesmo modo, modelagem multidimensional de dados XML possui diversos trabalhos disponíveis na literatura [8, 15, 16, 22, 24, 36, 43, 61, 63, 66, 67, 72, 82], mostrando a importância que este assunto tem em Tecnologia da Informação.

Encontrar um modelo adequado para um *data warehouse* baseado em dados XML tende a torna-se uma tarefa difícil, uma vez que várias soluções são propostas. Abordagens discutidas por Ravat et al. [32] vão desde o uso de dados XML incorporados em *data warehouses* tradicionais (e.g., ROLAP, MOLAP e HOLAP: arquiteturas relacional, multidimensional e híbrida) [20, 21, 81] até soluções que usam XML integralmente para o processo de *warehousing*. A Figura 4.1 ilustra tais abordagens de *data warehouse* para documentos XML.

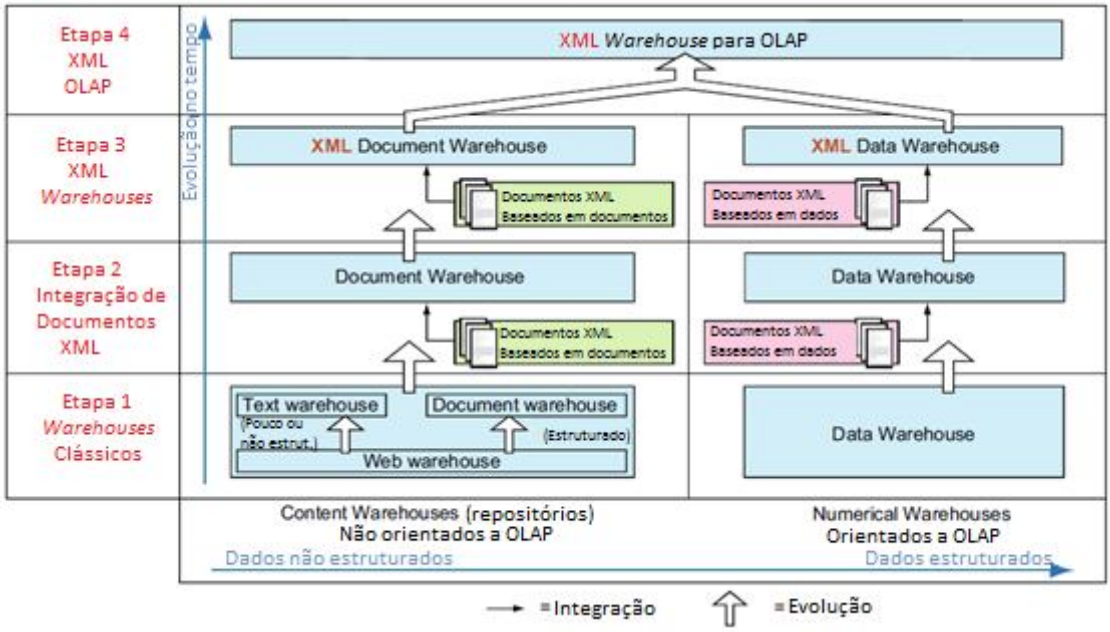

Figura 4.1 – O uso da tecnologia XML com as abordagens de warehousing. [adaptado de 32]

O trabalho de Ravat et al. [32] apresenta uma comparação entre diferentes *frameworks* de análise. São analisadas pesquisas que se referem (i) à integração de dados em *data warehouses* e *document warehouses*; (ii) a soluções para XML *data warehouse* e XML *document warehouse* e (iii) propostas OLAP para XML. Além disso, são apresentadas diretrizes para se selecionar um ambiente de tomada de decisões ao se projetar aplicações que combinam as tecnologias XML e *data warehouse*.

No que concerne à realização de consultas OLAP em dados XML, existem duas abordagens na literatura. A primeira realiza as análises com base em ferramentas OLAP convencionais, migrando os dados de documentos XML para ambientes relacionais [66, 67, 68, 69, 70, 100, 101]. Essa abordagem, que se enquadra na classificação de integração de documentos XML dada por Ravat et al. (Figura 4.1 - Etapa 2), requer o mapeamento dos dados semi-estruturados para o modelo relacional, tendo-se um custo de processamento adicional. Ademais, o processamento analítico dos dados contendo *links* não é considerado nos trabalhos analisados por Ravat et al. [32]. A outra abordagem trata apenas dados contidos em documentos XML, sendo necessário o desenvolvimento de técnicas para o processamento analítico de dados XML [11, 12, 14, 63,65, 88, 89]. Para isso define-se estruturas comuns para *data warehouses*. Nesse ambiente, os documentos XML baseados em dados, são organizados para serem utilizados por sistemas OLAP. A Figura 4.2 ilustra este contexto. Ravat et al. [32] consideram como pontos a serem explorados neste ambiente: a falta de estruturação para múltiplos documentos XML e a ausência de um formato comum para dados e metadados em todos os trabalhos por eles avaliados.

As pesquisas realizadas para este artigo enquadram-se na categoria de documentos XML baseados em dados. Por isso, esta seção está subdividida em três: na Seção 4.1, são avaliadas soluções acadêmicas e comerciais para navegação em *links*, uma vez que é necessária a obtenção de informações em documentos XML interligados para a realização de análise dos dados; na Seção 4.2, discute-se os trabalhos sobre a organização de documentos XML para permitir a análise multidimensional; na Seção 4.3, são contempladas as propostas para realização de consultas OLAP em dados XML; e as conclusões são apresentadas na Seção 4.4.

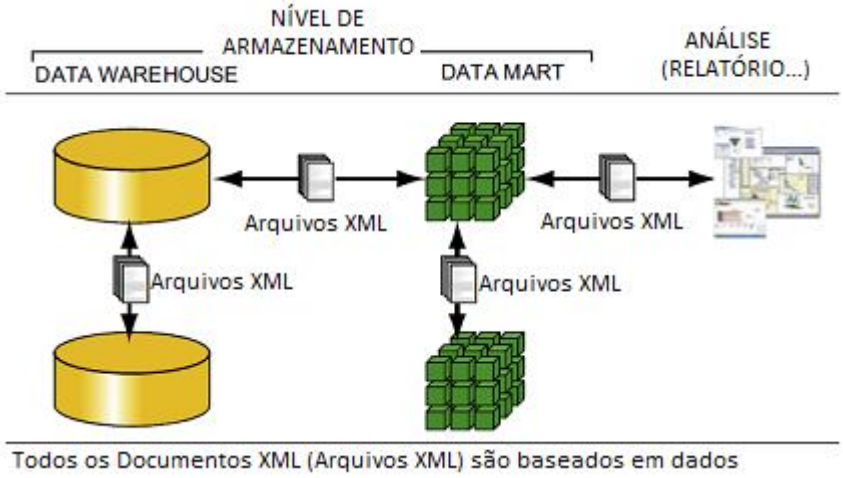

Figura 4.2 – Ambiente de apoio a suporte à decisão com documentos XML baseados em dados [adaptado de 32].

4.1 Navegação em Dados XML que Usam Links

Para suprir a necessidade de navegação em *links* mantidos em documentos XML existem algumas propostas comerciais e na literatura acadêmica. Lizorkim e Lisovsky [55] definem algumas categorias para implementações de XLink, entre elas a categoria de API (*Application Programming Interface*), a qual provê ao usuário recursos para navegação e gerenciamento de *links*. Os trabalhos discutidos nesta seção se enquadram em tal categoria.

May e Malheiro [57, 58, 59] propõem a criação de um modelo de documentos XML e de *linkbases*. Esta abordagem, *LoPiX*, é voltada para o tratamento de casos em que a navegação por meio de *links* envolve documentos que não estão presentes em um mesmo repositório. É proposta uma extensão para o *namespace xlink*, *dbxlink*, que possibilita a especificação de comportamentos dos *links* no momento da consulta. Assim, são sugeridas novas formas de avaliação e armazenamento dos *links* e dos documentos apontados por estes *links*. Por meio da inclusão do *namespace dbxlink*, é possível escolher como os elementos do *linkbase* são adicionados ao documento XML original. Na consulta, a distribuição dos arquivos é totalmente transparente ao usuário. No entanto, para isso, deve-se alterar o documento XML, incluindo ao menos, a configuração *dbxlink:transparent*. Isto é pouco prático em aplicações reais e não é aplicável em documentos cuja semântica dos relacionamentos representados por *links* é importante, pois com a semântica dos relacionamentos nos *linkbases*, o que se espera é recuperar os significados sem alterar os documentos. A partir desta abordagem, *dbxlink*, May et al. [56] propõem a realização de consultas em documentos XML interligados. Para isso, o sistema de banco de dados XML nativo eXist [31] é estendido. No entanto, nesse trabalho os *links* não são vistos como ligações explícitas, nas quais os usuários devem estar cientes dos *links* e determinar como atravessá-los em suas consultas. Esse mecanismo implícito de navegação pode provocar a recuperação de informações não desejadas pelo usuário.

Bry e Eckert [17] propõem uma extensão de XLink para o processamento de *linkbases* na Web. Eles introduzem o conceito de interface para a estrutura de *links* e um mecanismo de gerenciamento de *linkbases* baseado em três modos de ligações entre *linkbases* e o documento XML (ou um conjunto de documentos): "*transient*", "*temporary*" e "*permanent*". Isto ocorre pela definição de três valores para o atributo *xlink:arcrole* [95] que informam o modo como um *linkbase* é carregado em um *browser* ou aplicação de usuário. No modo *transient*, o *linkbase* é carregado no documento corrente e quando o usuário atravessa um *link*, deixando o documento, o *linkbase* e toda informação contida no *link* é

descarregada. O modo *permanent* tem efeito oposto ao *transient*, o *linkbase* é carregado no documento corrente e em todos os futuros documentos navegados. Uma vez carregado o *linkbase*, este permanece em memória até a sessão do *browser* expirar. No terceiro modo, *temporary*, o *linkbase* é ligado a um grupo restrito de documentos. Os *links* permanecem disponíveis para este grupo de documentos até que o usuário navegue para um documento que não esteja no grupo. Além das definições dos *arcroles*, uma extensão de XLink foi feita com a definição de elementos de ligação para a definição de interfaces (*xlink:type=”interfacedef”*) e para o referenciamento das interfaces (*xlink:type=”interfaceref”*). Uma interface fornece um nome simbólico para uma *homepage*, tornando os links menos vulneráveis a mudanças na localização física dos documentos.

O trabalho de Lizorkim [52] sugere a extensão de XPath, a partir da adição de três eixos: *traverse*, *arc* e *traverse-arc*. O primeiro, *traverse*, trata dos nós que são destinos para todos os arcos que se iniciam no nó de contexto. O segundo, *arc*, refere-se a uma lista de nós, filtrada a partir de um determinado requisito, que são retornados se o nó contexto é o início de, pelo menos, um arco. Por fim, o eixo *traverse-arc* equivale ao uso dos dois eixos anteriores em conjunto. A implementação desta solução é realizada com base em expressões *S*, um tipo nativo na linguagem funcional *Scheme*. Isto requer a conversão de arquivos XML para o formato *SXML*[53], uma representação da XML *Information Set* [94] em forma de *S-expression*.

A proposta *SXPath* [51, 53, 54, 55] refere-se à implementação da linguagem XPath para manipular *links* na linguagem funcional *Scheme*. Tal abordagem é justificada pela eliminação do conflito de *impedance mismatch* [28], causado pelas diferentes formas de representações de um documento XML em contextos interno a uma aplicação, por meio de modelos de documento como o DOM; e externo, por meio de texto, que é a forma geralmente utilizada [51]. Esse conflito ocorre quando se usa diferentes modelos de dados e paradigmas de programação, como por exemplo, Java, C e SQL que usam tipos de dados diferentes [28]. Sendo assim, na abordagem *SXPath*, há inicialmente a conversão do documento XML para *SXML*. Esta etapa pode ser realizada por intermédio do *framework SSAX* [78]. Em termos gerais, documentos XML e *SXML* são bastante similares, possuindo poucas diferenças sintáticas. Porém, como a nova representação é uma *S-expression*, há maior compatibilidade com as funções desenvolvidas em *Scheme*, as quais são utilizadas para a realização das consultas. Outra característica dessa abordagem é que funções podem ser definidas para uso posterior.

XLink Processor (XLip) é uma implementação da Fujitsu [93] para extrair informações do tipo XLink e XPointer em documentos XML. Baseada em Java e DOM, ela possibilita o uso de todos os tipos de *links* expressos por XLink. No entanto, é uma solução proprietária. Outra solução comercial para lidar com XLink, baseada em Java e DOM, é a *Batavia XBRL Java Library* (BXJL) [9], todavia é específica para manipular apenas documentos XBRL.

Java XBRL API Implementation [92] é um projeto de código aberto que fornece um processador XLink para documentos XBRL. Esse processador é uma solução que utiliza SAX [77], em lugar de uma árvore DOM. Outra abordagem baseada em SAX é a *XLink Filter* [49], que cria uma coleção de *links* para atender requisições de aplicações para identificar quais elementos contêm *links*. Alega-se que desta forma os usuários ganham em conveniência, uma vez que não é necessário construir processadores XLink em aplicações que utilizam SAX [49]. A utilização de SAX tem a vantagem de ser mais eficiente quanto ao uso de memória, mas não é uma solução padronizada pelo W3C.

XLinkit [64] é uma ferramenta para gerar *links* de acordo com regras e verificar a consistência dos documentos. Como entrada, a ferramenta recebe um grupo de documentos XML e um grupo de potenciais regras que ligam o conteúdo destes documentos. As regras expressam restrições de consistências entre os documentos. *XLinkit* retorna um *linkbase* com um grupo de *links* do tipo XLink, os quais permitem a navegação entre os elementos dos documentos.

Ahmedi [5, 6] baseou-se no protocolo LDAP (Lightweight Directory Access Protocol) para consultar, por meio de expressões XPath, documentos XML interligados com *links* do tipo XLink. Como LDAP possui um modelo de dados diferente de XML, os documentos XML interligados precisam ser mapeados para este modelo. Essa proposta tem a vantagem de possibilitar o acesso aos dados e realizar consultas

em arquivos XML em diferentes locais, como servidores internos de uma organização ou na Internet. Solução semelhante para a realização de consultas em documentos XML interligados a partir de expressões XPath é feita por Ahmedi e Arifaj [3, 4], mas nesta proposta não há necessidade de mapeamento para um modelo de dados diferente do XML. No entanto, as consultas XPath, nesses trabalhos [3, 4, 5, 6], são baseadas no atributo *xlink:href*, não explorando todo o poder de expressão semântica de XLink, i.e., o processamento de consultas não considera os atributos semânticos de XLink (*title*, *role* e *arcrole*), os de comportamento (*show* e *actuate*), nem os valores para o atributo *xlink:type* (*resource*, *locator*, *arc* e *title*).

XSPath [18] é uma linguagem derivada de XPath, porém com foco em consultas sobre documentos XML *Schema* [99]. Em XSPath, a noção de consultas realizadas em passos, presente em expressões XPath, é estendida para permitir a navegação por meio das ligações entre os elementos e seus tipos de dados, contidos no esquema de dados, especificado em documentos XML *Schema*, inclusive dentro da estrutura hierárquica dos tipos complexos de dados [85]. Contudo, nem sempre há interesse em explorar essa estrutura complexa. Por conseguinte, XSPath oferece duas abordagens distintas de navegação: expressões de baixo nível, que analisam a estrutura interna de tipos complexos, e expressões de alto nível, cujos passos independem de como elementos são combinados para formar tipos complexos.

4.2 Cubo de Dados XML

Nas Seções 4.2.1 e 4.2.2, são descritos os trabalhos sobre a organização de documentos XML para permitir o processamento de consultas OLAP. Na Seção 4.2.1 é discutido um modelo de dados, X-Cube, baseado em XML *Schema*, que define estruturas para dimensões, fatos e cubos. Na Seção 4.2.2, é detalhada XBRL *Dimensions*, uma solução específica para a representação de relatórios financeiros, que se fundamenta em XML *Schema* e XLink para estruturar dados multidimensionais.

4.2.1 XCube

Uma arquitetura para intercâmbio de dados multidimensionais na *Web* foi proposta por Hümmer [41]. Ela se baseia em uma arquitetura cliente-servidor, na qual o servidor envia ao cliente o esquema dos dados e os dados do *data warehouse*. Essa é uma solução não proprietária que se propõe a ser uma padronização de dados XML multidimensionais. Para isso, foi definido um conjunto de modelos de

documentos XML para armazenar e trocar dados, chamado de XCube. XCube é composto por três esquemas XML para a definição do modelo de dados para o cubo: (i) *XCubeSchema*, que armazena o esquema multidimensional dos dados; (ii) *XCubeDimension*, que explicita a estrutura hierárquica das dimensões; e (iii) *XCubeFact*, que contém os fatos do *data warehouse*. Essa divisão de esquemas possibilita o reuso de alguns desses documentos.

XCubeSchema é o esquema central para descrever a estrutura multidimensional do cubo de dados. Um exemplo desse tipo de documento pode ser observado no Quadro 4.1. O elemento *cubeSchema* do documento contém a coleção de fatos e dimensões, enquanto que o elemento *classSchema* descreve os níveis de classificação das dimensões envolvidas. Ao lado do subelemento *attribute*, que determina o nível atual, *rollUp* aponta para o nível superior, definindo assim a hierarquia das dimensões do cubo. A ligação entre a dimensão e a hierarquia acontece pela referência que ocorre entre *cubeSchema/dimension* e *classSchema/classLevel*.

Quadro 4.1 - Documento XCubeSchema [41].

```
<multidimensionalSchema version="0.4"
    xmlns="http://www.xcube-open.org/V0_4/XCubeSchema.xcsd">
    <cubeSchema id="sale">
            <fact id="sales"/>
            <fact id="revenue"/>
            <dimension id="geography" granularity="branch"/>
            <dimension id="product" granularity="article"/>
    </cubeSchema>
    <classSchema>
            <!—geography-->
            <classLevel id="branch">
                    <attribute id="manager"/>
                    <rollUp toLevel="city"/>
            </classLevel>
            <classLevel id="city">
                     <rollUp toLevel="region"/>
           </classLevel>
            <!--product-->
                    <classLevel id="article">
                       <attribute id="articleName"/>
                       <attribute id="brand"/>
             <rollUp toLevel="productGroup"/>
            </classLevel>
            <classLevel id="productGroup">
                    <rollUp toLevel="productFamily"/>
            </classLevel>
    </classSchema>

</mutidimensionalSchema>
```

XCubeDimension possui o propósito de formalizar estruturas dimensionais. Em essência, um documento *XCubeDimension* contém os nós pertencentes aos níveis de classificação definidos em *XCubeSchema*, instanciando as hierarquias das dimensões do cubo. A estrutura básica do documento *XCubeDimension* pode ser visualizada no Quadro 4.2, no qual o elemento *units* define unidades e o elemento *classification* define um grupo de nós para cada nível da dimensão com seus possíveis valores.

É importante notar que, apesar da similaridade existente entre os relacionamentos *Roll Up*, nos documentos dos Quadros 4.1 e 4.2 eles representam associações distintas, pois enquanto nos documentos *XCubeSchema* os elementos r*ollUp* expressam os relacionamentos entre níveis de classificação (e.g. *branch – city*, no Quadro 4.1), em *XCubeDimensions* eles expressam as associações entre nós, isto é, instâncias dos níveis ou membros (e.g. *Northern Germany – Germany*, no Quadro 4.2).

Após descrever as dimensões e seus relacionamentos, as células dos cubos de dados podem ser definidas utilizando-se um esquema *XCubeFact*. Com base nesse esquema, uma coleção de células é especificada para cada cubo. Cada célula é constituída de duas partes: a dimensão, representando as coordenadas dimensionais; e o valor do fato. Um exemplo ilustrando o uso de *XCubeFact* é mostrado no Quadro 4.3. Nesse exemplo, nota-se que uma célula pode conter vários fatos, ou seja, as medidas *sales* e *revenues* se relacionam com as mesmas dimensões, e isso é explicitado no elemento *cell* do documento. Além da descrição básica de cubos de dados, XCube contém outros componentes, como *XCubeText* que representa descrições textuais e comentários, em diversos idiomas, para os elementos dos documentos *XCubeSchema* e *XCubeDimension*. Portanto, esta representação de cubos com

esquemas definidos pode facilitar o uso de XCube em diferentes *data warehouses*, além de fornecer a possibilidade de suporte a diversos idiomas e domínios diferentes de aplicações, por meio de *XCubeText*.

Quadro 4.2 - Documento XCubeDimension [41].

```
<dimensionData version="0.4"
        xmlns= "http://xcube-open.org/V0_4/XCubeDimension_base.xcsd>
        <units>
                <entry unitType="currency" unit="BUR"/>
        </units>
        <classification>
                <!-- dimension geography-->
                <level id="country">
                        <node id="Germany"/>
                        <node id="Switzerland"/>
                        <node id="Fance"/>
                        <!-- ... -->
                </level>
                 <level id="region">
                        <node id="Northen Germany">
                                <rollUp toNode="Germany" level="country"/>
                        </node>
                        <node id="Western Germany">
                                <rollUp toNode="Germany" level="country"/>
                        </node>
                        <node id="Eastern Germany">
                                <rollUp toNode="Germany" level="country"/>
                        </node>
                        <node id="Southern Germany">
                                <rollUp toNode="Germany" level="country"/>
                        </node>
                        <!-- ... -->
                </level>
                <!-- ... -->
        </classification>
</dimensionData>
```

Quadro 4.3 - Documento XCubeFact [41].

```
<cubeFacts version="0.4"
        xmlns="http://www.xcube-open.org/V0_4/XCubeFact_base.xcsd"/>
         <cube id="sale">
                <cell>
                        <dimension id="geography" node="branch48"/>
                        <dimension id="product" node="MA-450"/>
                        <dimension id="time" node="2003-07-24"/>
                        <fact id="sales" value="3"/>
                        <fact id="revenue" value="960"/>
                </cell>
                <cell>
                        <dimension id="geography" node="branch75"/>
```

```
                    <dimension id="product" node="MA-450"/>
                <dimension id="time" node= "2003-07-24"/>
                <fact id="sales" value="3"/>
                    <fact id="revenue" value="960"/>
            </cell>
    </cube>
</cubeFacts>
```

XCube é uma proposta de padronização de cubos de dados para documentos XML, na qual são apresentados esquemas para representação de dimensões, fatos e cubos para a construção de *data warehouses* baseados em XML. Esta abordagem tem as vantagens inerentes a um ambiente padronizado, facilitando a reutilização de documentos (e.g. documentos de dimensões) para domínios diferentes. XCube resolve parte dos problemas de heterogeneidade em XML, tais como: (i) estrutura do documento, que é definida em esquemas baseados em XML *Schema*; e (ii) diferença sintática de conteúdo, que é resolvida parcialmente, quando uma mesma informação está representada em unidades diferentes, as quais são definidas pelo atributo *units*. Porém, não é discutido o tratamento dado a elementos com conteúdos similares, mas com nomes diferentes ou escritos em idiomas diferentes. A heterogeneidade semântica não é abordada em *XCube*. Outras questões não consideradas são a especificação de um mesmo membro em mais de uma dimensão, além de questões relativas ao ordenamento dos elementos e à apresentação destes ao usuário em idiomas diferentes.

4.2.2 XBRL 2.1 e XBRL Dimensions

XBRL 2.1 (*eXentensible Business Reporting Language*) é uma especificação baseada em XML *Schema* e XLink. Essa linguagem é um padrão aberto e gratuito, desenvolvido por um consórcio de aproximadamente 650 organizações e agências governamentais de aproximadamente 30 países [26, 91]. Sua concepção tem como base a criação, o intercâmbio e a análise de relatórios de informações financeiras. Como tal, permite que investidores, pesquisadores e profissionais do mercado financeiro analisem e extraiam informações por meio de suas aplicações, simplificando uma das fases principais da análise financeira: a obtenção e conversão de formatos de dados.

XBRL especifica a sintaxe a ser utilizada para reportar o valor de um fato financeiro, baseada em um conjunto de conceitos definidos dentro de um contexto particular. XBRL divide a informação do relatório financeiro em dois componentes distintos: instância e taxonomia. A instância contém os fatos reportados, enquanto a taxonomia, formada por um ou mais esquemas de dados e *linkbases*, fornece os conceitos comunicados pelos fatos. A combinação de uma instância XBRL, do esquema de sua taxonomia e do conjunto de *linkbases* associados constitui um relatório financeiro XBRL. A Figura 4.3 ilustra o conjunto de documentos usados na estrutura de XBRL para descrever os fatos financeiros.

Figura 4.3 - Documentos na Estrutura XBRL.

Uma taxonomia é definida por um documento *XML Schema* e um conjunto de *links* estendidos, referenciados no documento *XML Schema*. A terminologia de XBRL descreve conceitos para os quais os fatos são reportados. Conceitos são criados pela especificação de elementos no documento *XML Schema*, definindo um esquema de dados. No esquema de dados de uma taxonomia, um conceito recebe um nome e o tipo de dado que ele representa. Os links estendidos expressam o significado dos conceitos, por meio da representação das relações existentes entre eles e associando conceitos com sua documentação. Relações entre fragmentos de informações em XML ocorrem em XBRL como relações entre a instância XBRL e sua taxonomia, ou como relações internas às instâncias (e.g. entre fatos e notas de rodapé). A semântica destes conceitos é expressa por meio de *linkbases*, ou seja, os *linkbases*: expressam relações hierárquicas entre elementos, criam rótulos em diversos idiomas para os elementos, definem ordem de apresentação dos elementos em uma instância e estabelecem relações de cálculos e referências. XBRL expressa esses relacionamentos usando as sintaxes de *links* simples e *links* estendidos, definidas na especificação XLink.

A instância XBRL, por sua vez, contém os valores a serem reportados para cada conceito definido em uma ou mais taxonomias. Um documento de instância XBRL pode ser baseado em mais de uma taxonomia. Por sua vez, taxonomias podem ser interconectadas, estendendo ou modificando umas às outras de várias maneiras. De um modo geral, é necessário considerar múltiplos esquemas de taxonomias e *linkbases* relacionados para interpretar uma instância XBRL.

Complementando a especificação XBRL 2.1, *XBRL Dimensions* [40] foi desenvolvida para representar múltiplas dimensões relativas a determinado fato financeiro. A especificação XBRL 2.1 padronizou apenas a representação de duas dimensões: a dimensão tempo e a dimensão entidade. Porém, para muitos propósitos de análise, múltiplas dimensões são requeridas em relatórios financeiros. *XBRL Dimensions* define quatro possíveis funções de taxonomias para informações multidimensionais: (i) taxonomia *primary*, que é uma taxonomia sem elementos dimensionais. É a taxonomia puramente baseada na especificação XBRL 2.1, que define os elementos, chamados de itens primários [40], representando os fatos financeiros; (ii) taxonomia *domain-member*, cujos itens formam um conjunto de membros pertencentes a um determinado domínio. Por exemplo, uma taxonomia do domínio de territórios geográficos, ou do domínio de linhas de produtos financeiros. A taxonomia *domain-member* modela o conteúdo das dimensões; (iii) taxonomia *dimensions*, que determina como um fato pode ser observado, definindo os elementos dimensionais e as relações entre eles e os domínios; e (iv) taxonomia *template*, que importa todas as taxonomias *primary* e *domain-member* e adiciona estruturas dimensionais a serem usadas pelo documento de instância.

Particularmente, uma taxonomia *template* define hipercubos, que, por sua vez, especificam um produto cartesiano de dimensões. Cada dimensão é definida sobre seus domínios, que são compostos por membros. O propósito da taxonomia *template* é definir a estrutura dos hipercubos e associá-los aos elementos primários. Um documento de instância pode usar qualquer número de taxonomias dimensionais para representar as possíveis combinações entre membros de domínios e fatos.

A Figura 4.4 mostra como estas definições estão inseridas na construção de uma instância XBRL *Dimensions*. A especificação XBRL 2.1 foi estendida com dois arquivos XML *Schema* que definem as restrições necessárias para a construção de taxonomias *Dimensions*. A especificação dimensional define a sintaxe e a semântica necessárias à criação de taxonomias dimensionais, as quais, por sua vez, definem as dimensões a serem usadas nos documentos de instância. Essa especificação define: (i) a sintaxe dos elementos que representam dimensões e seus membros e (ii) os arcos padrões que definem os relacionamentos existentes entre esses elementos para a criação da estrutura hierárquica, e entre elementos dimensionais e os que representam os fatos, permitindo, assim, a formação do cubo de dados. Essas taxonomias são sintaticamente idênticas àquelas definidas com base na especificação XBRL 2.1, com restrições que devem ser seguidas quando forem usadas como taxonomias dimensionais. Destarte, é fornecida a semântica necessária para uma análise multidimensional, a qual divide o documento de instância XBRL em uma parte não dimensional, que reporta os fatos, e em outra parte dimensional, na qual estão presentes as estruturas para a análise dos fatos.

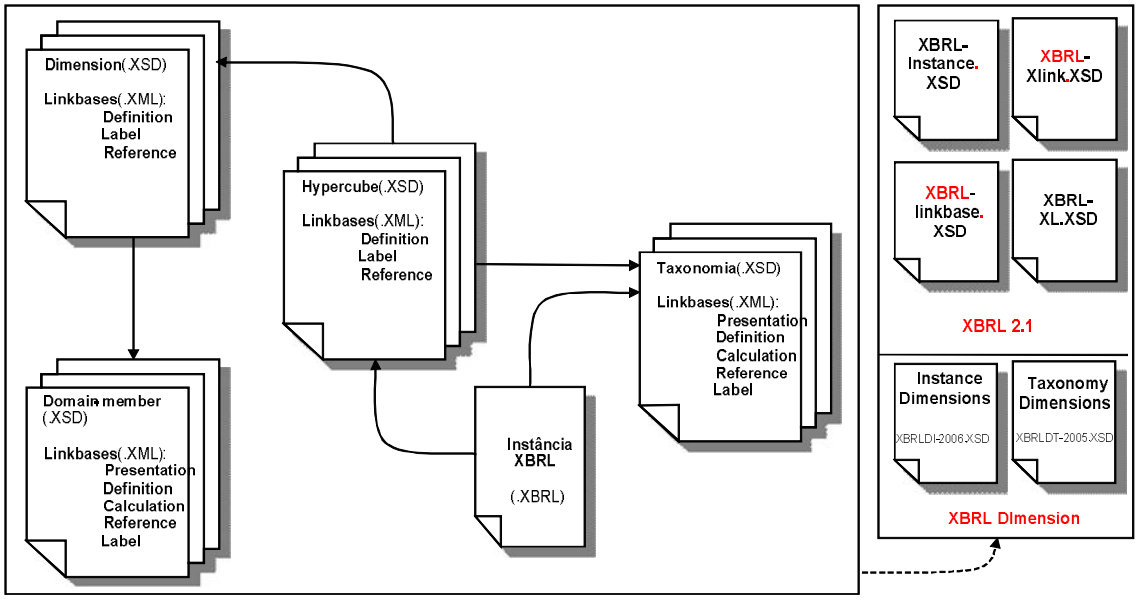

Figura 4.4 - Instância e Taxonomia XBRL *Dimensions*.

Embora a definição dos *linkbases* em XBRL não tenha sido realizada com o propósito de tratar a heterogeneidade em dados XML, nem a literatura específica de XBRL tenha abordado este assunto [10, 13, 26, 29, 30, 33, 40, 71, 79, 80], suas características solucionam tal questão. Por exemplo, a heterogeneidade semântica em XBRL é tratada pelo uso dos *linkbases definition* e *label*. O uso do atributo *unit* no contexto da instância e também do *linkbase label* resolvem a heterogeneidade sintática de conteúdo. Questões relativas à heterogeneidade estrutural são solucionadas com o esquema de dados, baseado em XML *Schema*, e o *linkbase definition*. O ordenamento dos elementos para a apresentação ao usuário é feito pelo *linkbase presentation*. Entretanto, como a obrigatoriedade do uso desses *linkbases* não é determinada pela especificação XBRL, o tratamento desses problemas podem não ser considerados pelo usuário desenvolvedor da taxonomia, o que indica que a

solução para a heterogeneidade dos dados pode ser resolvida apenas parcialmente. Outra situação presente em XBRL é que sua aplicação é restrita ao domínio financeiro e dentro deste domínio ainda existem restrições de uso, já que as relações matemáticas que podem ser representadas neste contexto referem-se apenas àquelas de soma, não podendo ser representadas outras relações, tal como a divisão, que é muito utilizada para indicar índices financeiros.

4.3 OLAP em XML

Nas seções seguintes, serão discutidos trabalhos relativos a consultas OLAP sobre dados representados em XML. Nas seções 4.3.1, 4.3.2, 4.3.3 e 4.3.4 são descritas propostas de linguagens que apresentam operadores para consultas OLAP em XML, enquanto na Seção 4.3.5 é vista uma extensão para incorporar em XQuery alguns recursos de consultas OLAP.

4.3.1 XQ-Cube

Um *framework* para análise multidimensional sobre documentos XML, chamado XML-OLAP, foi proposto por Park et al. [65]. XML-OLAP baseia-se em um *data warehouse* composto por dados no formato XML, chamado de XML *Warehouse*, no qual tanto os fatos como as dimensões são representados como documentos XML. A partir disso, são construídos cubos contendo não apenas dados numéricos como valores de medidas, como geralmente ocorrem em um *data warehouse* convencional, mas também dados textuais. Esse trabalho apresenta uma linguagem de consulta a dados multidimensionais, denominada XML-MDX, inspirada na linguagem MDX [60]. Sentenças XML-MDX usam expressões XQuery para indicar dados de medidas, dimensões e operações de seleção. O modelo do XML *Warehouse* possui um repositório de documentos XML para os fatos, no qual cada fato é descrito em um documento XML. Esse modelo baseou-se no trabalho apresentado por Nassis et al. [63, 83, 84], cujos fatos têm uma estrutura de árvore contendo dados estruturados e não estruturados. Dimensões também são descritas em documentos XML, então diversos repositórios são utilizados, um para cada dimensão. A Figura 4.5 ilustra esse modelo de dados. Estruturas de índices são usadas para relacionar os fatos com as dimensões. Esses índices são construídos junto com o repositório das dimensões.

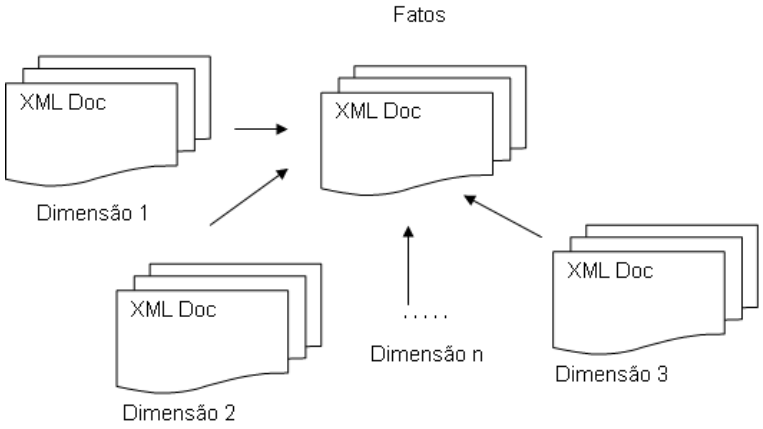

Figura 4.5 - Modelo Multidimensional de um XML Warehouse [65].

Como o XML *Warehouse* possui documentos XML para representar os fatos, o cubo construído deve ter células cujos valores são uma agregação de documentos XML. Esses cubos, denominados XQ-Cube, são construídos por XML-MDX com valores descritos por uma expressão XQuery. XML-MDX possui duas sentenças: CREATE XQ-CUBE, para criar um novo XQ-Cube, e SELECT, para realizar consultas sobre o XQ-Cube.

A modelagem dimensional para XML discutida nesta seção tem a vantagem de permitir a realização de consultas analíticas sobre documentos XML. Como XML possui uma estrutura hierárquica, a representação das hierarquias das dimensões é facilitada. Porém, por XML ser

flexível na representação hierárquica, pode ocorrer divergência na estrutura interna dos documentos. A utilização de expressões XQuery para construir o cubo é vantajosa, pois, além de ser um padrão W3C, XQuery manipula dados tanto numéricos como textuais, possibilitando a construção de cubos de ambos os tipos. A linguagem XML-MDX, além de possuir as características de MDX, adiciona recursos para a realização de consultas analíticas sobre documentos XML. No entanto, observa-se a ausência da definição do esquema de dados dos documentos XML que definem o *data warehouse*, o que implica na falta de tratamento das questões de heterogeneidade em XML.

4.3.2 Xaggregation

Wang et al. [86, 87, 88] apresentam conceitos de XOLAP (OLAP para XML) e a operação *Xaggregation*, que, por meio de expressões XPath e de operadores de agregação para documentos XML, constrói dimensões e medidas baseadas em dados XML.

Esse trabalho considera que medidas e dimensões são representadas em um documento XML por meio de seus elementos. O elemento no qual a agregação é baseada, tal qual uma tupla no ambiente relacional, é definido como objeto de agregação, cujo nó que o identifica é chamado de nó raiz. O resultado de *Xaggregation* consiste na agregação dos fatos sobre um objeto de agregação, que é agrupado pelas dimensões e satisfaz uma expressão XPath dada na consulta. Assim, *Xaggregation* é uma agregação baseada na medida *m*, agrupada pelas dimensões d_1, d_2,..., d_n, sendo *m*, d_1, d_2, ..., d_n, descritos por expressões XPath.

Em *Xaggregation*, as medidas e dimensões não possuem uma descrição simples como em um ambiente relacional. Se um objeto de agregação tem mais do que uma medida, a agregação pode ser considerada pelos valores das medidas. Para um mesmo nó raiz, pode haver mais de um nível de dimensão com o mesmo elemento (*tag*), mas com diferentes valores ou expressão XPath variada. Isto ocorre porque um nó em uma árvore XML não é representado apenas por seu valor, mas também pelo seu caminho, ou seja, é possível ter em XML, para um elemento raiz, mais de um nível de dimensão com o mesmo elemento. Então, uma agregação pode acontecer pelo valor das dimensões, pelo seu caminho ou pela combinação de valores e caminhos. Assim, a semântica de uma expressão de agregação pode ter diferentes significados.

A possibilidade de uma mesma dimensão ocorrer diversas vezes e com diferentes estruturas hierárquicas é que irá diferenciar uma operação de agregação de outra. Para que esta diferenciação seja explicitada na consulta OLAP, foram definidas restrições a serem adicionadas em uma expressão XPath, que avaliam os diversos valores, caminhos e composições hierárquicas das dimensões. Desta maneira, a operação de agregação em documentos XML é enriquecida com expressões XPath que levam em conta as diferenças hierárquicas existentes, buscando solucionar o problema da heterogeneidade estrutural. Essa operação pode ser embutida em uma consulta XQuery, que traz a vantagem de ser uma linguagem de consulta padrão para XML. Esse trabalho procura resolver a questão da heterogeneidade estrutural em XML, entretanto, por manipular documentos com estruturas diversas, não é explicitado um modelo de dados multidimensional, persistindo a heterogeneidade semântica.

4.3.3 IX-Cube

IX-Cube, *Iceberg XML cube* [48], consiste em uma abordagem de cubo de dados para XML desenvolvida para permitir a execução de consultas OLAP sobre dados XML, considerando que tais dados não possuem esquemas disponíveis (i.e. dados do mesmo tipo podem estar armazenados de formas distintas). Jian et al. [44] apresentam extensões de operações OLAP para dados XML, além de um algoritmo para as consultas sobre IX-Cube.

Dada uma árvore XML, um IX-Cube é especificado por uma 3-tupla (E_{xcube}, M_{xcube}, D_{xcube}), na qual E_{xcube} é uma entidade, M_{xcube} é uma medida e D_{xcube}, um grupo de dimensões. E_{xcube} é obtida por uma expressão XPath que fornece um caminho até um nó de interesse (nó de contexto). Uma dimensão é definida por uma 2-tupla (*DName, Paths*), sendo *DName*, o nome da dimensão e *Paths* um grupo de caminhos iniciados pelo nó de contexto. A medida é definida por uma 3-tupla (*AggFunction*, *Paths*, *min_supp*), sendo *AggFunction* uma função de agregação, *Paths* um grupo de caminhos a ser considerado para a agregação, cujo ponto de partida é o nó de contexto, e *min_supp* a restrição da agregação.

Esta abordagem para realização de consultas OLAP sobre documentos XML apresenta a vantagem de usar uma sintaxe semelhante a SQL e um método simples para construir o cubo de dados XML. Por se basear em XPath, são fornecidas facilidades ao usuário para a construção do cubo de dados XML e elaboração das consultas analíticas. Todavia, é necessário o conhecimento prévio das possíveis estruturas hierárquicas presentes no documento, restringindo a consulta a um universo de documentos com estruturas iguais. Também não se considera um esquema multidimensional para os dados XML, o que acarreta o problema de heterogeneidade estrutural para uma coleção de documentos. Além disso, nessa abordagem, a consulta é realizada em um único documento XML, e não sobre uma coleção de documentos, o que é esperado por aplicações de banco de dados.

4.3.4 OLAP XML

Bordawekar e Lang [14] apresentam uma solução OLAP fundamentada em dois tipos de consultas: no valor e na estrutura do documento XML. O primeiro tipo especifica o conteúdo de texto em um documento XML, que pode ser o valor de atributos ou de nós do tipo texto. O segundo especifica padrões na estrutura de uma árvore XML. Para exemplificar esses dois tipos de análise, considere a estrutura de árvore de um documento XML, ilustrado pela Figura 4.6, representando uma estrutura organizacional composta de divisões, departamentos e grupos, os quais possuem empregados com os atributos salário e data de nascimento.

Uma análise baseada em valor seria "*//employee/dob*", que representa o grupo de todas as datas de nascimento de todos os empregados. Já uma análise baseada na estrutura poderia ser "*//division/{$x | $x/$y}/group*", que encontraria os padrões */division/departament/group* e */division/departament/departament/group*. O primeiro corresponde à variável $x para os nós *departament*, e o segundo coincide com as variáveis $x e $y, ambas para os nós *departament*. Então, baseados nesses conceitos, são especificados os operadores OLAP *Group-By*, *Roll-Up* e *Cube* para lidar com documentos XML.

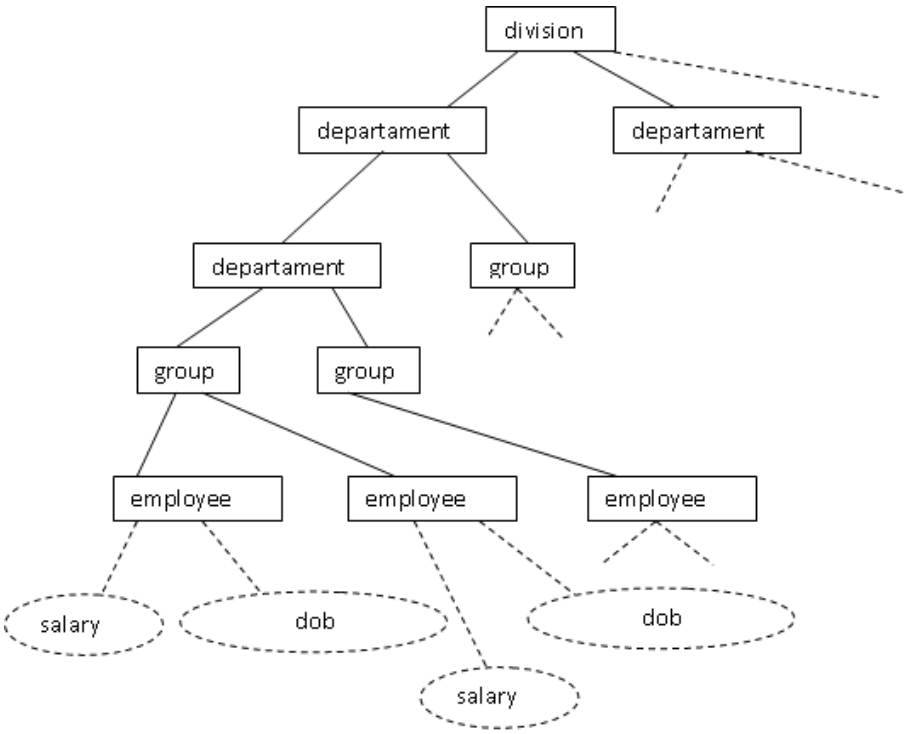

Figura 4.6 - Representação em árvore de uma estrutura organizacional [14].

Para responder à pergunta "*Qual a renda média por data de nascimento?*", a consulta do Quadro 4.4 seria especificada. Avaliando a questão "*Qual o salário médio por departamento?*", seria necessária uma análise baseada na estrutura e a definição da consulta seria como ilustrado no Quadro 4.5.

Quadro 4.4 - Consulta para análise baseada em valor [14].

```
for $e in //employee
GROUP BY(//employee/dob)
let $s := avg( $e/salary )
return
<ageGroup>
   <dob_range> $e/dob </dob_range>
   <avgSalary> $s </avgSalary>
</ageGroup>
```

Quadro 4.5 - Consulta para análise baseada na estrutura [14].

```
var d,d1,d2=department
for $e in //employee
GROUP BY (//division/{$d | $d1/$d2}/employee )
let $s := avg( $e/salary ),
    $l := ( $d1 == NULL ? 1 : 2 )
Return
<levelGroup>
    <level> $l </level>
    <avgSalary> $s </avgSalary>
</levelGroup>
```

De forma semelhante, consultas baseadas em outros operadores OLAP (e.g. *roll up*, *group by*) podem ser realizadas. Nesse trabalho, também é apresentada uma forma reduzida para os usuários realizarem suas consultas. Com o acréscimo da palavra reservada *Topological*, o usuário não precisa ter um conhecimento exato do esquema do documento XML. O Quadro 4.6 exibe um exemplo de uma consulta que faz uso dessa palavra chave e o Quadro 4.7 mostra a mesma consulta sem o uso de *Topological*.

Quadro 4.6 - Consulta com a palavra reservada *topological* [14].

```
GROUP BY TOPOLOGICAL CUBE(//division/$d1/$d2/employee, //division/$d1/$d2/budget)
```

Quadro 4.7 - Consulta sem a palavra reservada *topological* [14].

```
GROUP BY CUBE(//division/employee,
//division/$d1/employee, //division/$d1/$d2/employee,
//division/budget, //division/$d1/budget,
//division/$d1/$d2/budget)
```

Esse trabalho possui a vantagem de realizar o processamento da consulta baseado em valor ou na estrutura do documento, fornecendo maior flexibilidade ao usuário. A forma reduzida de consulta, pelo uso da palavra-chave *Topological*, elimina a necessidade de conhecimento prévio das estruturas presentes no documento, permitindo a realização de consultas mais flexíveis. Outra característica dessa abordagem diz respeito a ela ser uma extensão de XQuery, fornecendo os mesmos benefícios e restrições já discutidos nas seções anteriores. Apesar de tratar a heterogeneidade estrutural, ela é restrita a apenas um documento XML. Em razão de não se ter um esquema de dados definido, os outros tipos de heterogeneidade, semântica e sintática, não são abordados.

4.3.5 X^3 Cube

O operador *X^3*, apresentado por Wiwatwattana et al. [89], para realizar consultas OLAP sobre dados XML, foi definido para considerar possíveis variações de estruturas nas árvores de documentos XML. São discutidos algoritmos que manipulam problemas de sumarização [50], tal como a ausência de um elemento opcional em uma das ramificações da árvore do documento. Esta situação pode, por exemplo, acarretar na exclusão de um elemento usado como parâmetro de agrupamento em uma agregação, produzindo um resultado incorreto. Outro problema da mesma categoria considerado na especificação do *X^3* é a contagem repetida de sub-elementos que ocorrem em grupos distintos. Por exemplo, analisando a Figura 4.7, a primeira publicação é membro dos grupos (John, p1, 2003) e (Jane, p1, 2003). Então, o grupo (p1, 2003) contém somente a primeira publicação e sua contagem resulta no valor um. No entanto, uma operação de *roll-up* no nível dos grupos mencionados, resultará em uma contagem individual, que somados produzirão o valor dois, o qual é errado.

Devido à heterogeneidade dos dados XML, que dificulta em algumas situações a definição de um único padrão de árvore que coincida com todos os itens que o usuário deseja agrupar, ou para situações nas quais não se tenha uma informação precisa do esquema de dados XML, é proposto um relaxamento no padrão da estrutura da árvore de agrupamento:

• *Parent-Child to Ancestor-Descendant Edge Generalization* (PC-AD) - é um relaxamento estrutural da relação entre dois elementos, do tipo pai-filho para ancestral-descendente. Por exemplo, na Figura 4.7, o padrão *publication/author* não está presente em todas as publicações, no entanto, com o relaxamento de padrão, *publication//author*, coincidirá com todas as publicações;

- *Sub-tree Promotion* (SP) - move uma subárvore de um nó *n* para ser filha de um nó ascendente de *n*. Por exemplo, *publication*[./*author*/*name*] pode ser flexibilizada para [./*author*][.//*name*]; e
- *Leaf Node Deletion* (LND) - permite que padrões de árvores sejam considerados, mesmo na ausência de um elemento folha. Por exemplo, uma consulta que inclua *publisher* na busca, não pode ser encontrada no terceiro elemento *publication* da Figura 4.7. Porém, se o nó *publisher* é opcional, então a terceira árvore fará parte do padrão de busca.

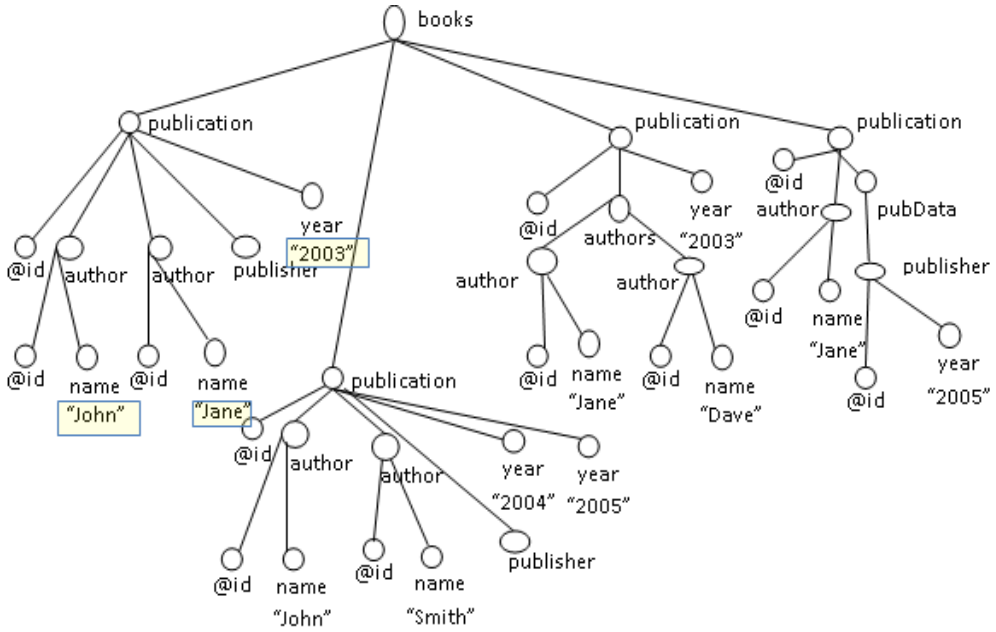

Figura 4.7 - Documento XML para publicações [89].

A cláusula *X^3* é incluída em expressões XQuery para identificar explicitamente os relaxamentos permitidos em cada variável e garantir que a agregação seja computada da forma desejada pelo usuário. O Quadro 4.8 ilustra o uso deste operador com expressões XQuery.

Quadro 4.8 - X^3 em expressões XQuery [89].

```
for     $b in doc("book.xml")//publication,
$n in $b/author/name,
$p in $b//publisher/@id,
$y in $b/year
X^3     $b/@id by $n (LND, SP, PC-AD), $p (LND, PC-AD), $y (LND)
return COUNT($b)
```

Esse trabalho apresenta a vantagem de efetuar consultas sobre documentos XML considerando a flexibilidade das estruturas de árvores dos documentos. Com o uso de expressões XQuery e a inserção dos relaxamentos nas estruturas hierárquicas da árvore XML, a consulta se torna mais flexível e precisa, já que se evita os problemas provenientes da heterogeneidade da estrutura de dados XML. No entanto, não é levado em conta que determinados domínios podem utilizar documentos XML interligados para expressar as relações semânticas e hierárquicas dos elementos presentes nas instâncias XML. Além disso, não é considerada uma estrutura multidimensional para representar o cubo de dados nem uma linguagem de consulta com este fim (e.g. MDX). Assim sendo, as heterogeneidades semântica e sintática dos dados XML não são sanadas.

4.3.6 Estendendo XQuery com Consultas OLAP

É importante para a manipulação de documentos XML que o padrão XQuery permita a realização de consultas analíticas. Apesar de ser possível expressá-las implicitamente com a atual sintaxe de XQuery, a falta de construtores explícitos de agregação torna a construção destas consultas difícil e não intuitiva. Com o objetivo de suprir estas deficiências, é proposta, em Beyer *et al.* [11, 12], uma extensão das expressões FLWOR de XQuery [98] com sintaxe de agrupamento e de numeração dos resultados.

Os autores desse trabalho acrescentam uma cláusula opcional *group by*, seguida da cláusula *where*, na sintaxe das expressões FLWOR. Estando a cláusula *group by* presente na expressão da consulta, então podem ocorrer, opcionalmente, as cláusulas *let* e *where*. A cláusula *nest*, introduzida na sintaxe de XQuery por esse trabalho, permite que todas as informações sobre a seqüência de tuplas que contribuem para o agrupamento, sejam usadas em outras computações após o *group by*. O Quadro 4.9 mostra um documento XML utilizado para uma consulta de agregação ilustrada no Quadro 4.10. A mesma consulta realizada com as extensões propostas para XQuery pode ser vista no Quadro 4.11. Essa consulta, além de simplificar a expressão, permite que uma seqüência vazia seja considerada como um valor distinto para os propósitos de agrupamento.

Quadro 4.9 - Documento XML [12].

```
<book>
   <title>Transaction Processing</title>
   <author>Jim Gray</author>
   <author>Andreas Reuter</author>
   <publisher>Morgan Kaufmann</publisher>
   <year>1993</year>
   <price>59.00</price>
   <discount>6.00</discount>
</book>
```

Quadro 4.10 - Expressão XQuery para uma consulta de agregação [12].

```
for $p in distinct-values(//book/publisher),
    $y in distinct-values(//book/year)
let $netprices :=  //book[publisher = $p and year = $y]/(price – discount)
where fn:exists($netprices)
order by $p, $y
return
  <group>
    <publisher>{$p}</publisher>
    <year>{$y}</year>
    <avg-net-price>{avg($netprices)}</avg-net-price>
  </group>
```

Quadro 4.11 - Expressão XQuery estendida com a cláusula *group by* [12].

```
for $b in //book
group by $b/publisher into $p, $b/year into $y
nest $b/price - $b/discount into $netprices
return
  <group>
    {$p, $y}
    <avg-net-price>{avg($netprices)}</avg-net-price>
  </group>
```

Outras extensões foram definidas, como a cláusula *using*, que especifica uma função de comparação, usada em expressões de comparação de agrupamentos, e a possibilidade de usar a cláusula *order by* após a cláusula *next*, de maneira que seja possível ordenar o resultado dentro do agrupamento.

A extensão proposta, além de simplificar a escrita e semântica de consultas analíticas com XQuery, abrange não só os casos de agrupamentos básicos, mas também questões de ordenação. Porém este trabalho não apresenta uma solução para resolver os problemas de heterogeneidade de dados XML, sendo apenas uma extensão de XQuery para realização de consultas analíticas.

5 CONSIDERAÇÕES FINAIS

De acordo com o estudo das propostas para navegar em documentos XML interligados por *links*, algumas soluções requerem uma transformação do documento XML (como as propostas baseadas em *SXML* e *LoPiX*), outras são limitadas a soluções específicas (como XBRL e *Xlinkit*). *Java XBRL API Implementation* e *XLink Filter* são fundamentadas pelo uso de SAX, o que restringe sua aplicação, já que SAX não é um padrão W3C. Ademais, com exceção de SXPath, nenhuma das outras soluções são especificadas como linguagem para navegação em *links* do tipo XLink. Porém, SXPath também não é baseada em padrões W3C.

O Quadro 5.1 apresenta uma comparação entre as soluções para a navegação em *links* discutidas anteriormente. Esta análise é feita considerando se a solução requer modificação nos documentos XML originais, se é de código aberto e baseada em padrões W3C (DOM, XPath, XML *Schema* e XLink.) e se é uma linguagem de navegação para XLink. Estas características são consideradas importantes para a definição de uma linguagem de navegação em *links*.

Quadro 5.1 - Comparação entre trabalhos para navegação em *links*.

Trabalhos	Modifica o documento XML	Baseada em padrões W3C	Solução de código aberto	Linguagem de navegação para XLink
dblink [57, 58, 59]	Sim	Não	Não Informado	Não
SXML / SXPath [51, 52, 53, 54, 55]	Sim	Não	Sim	Sim
Xlink Filter [49]	Não	Não	Não Informado	Não
XLiP [93]	Não	Sim	Não	Não
Batavia XBRL [9]	Não	Sim	Não	Não
Java XBRL API [92]	Não	Não	Sim	Não
Xlinkit [64]	Sim	Sim	Não Informado	Não
Gerenciamento de *Linkbases* [17]	Sim	Não	Sim	Não
XSPath [18]	Não	Sim	Sim	Não

Observa-se que um modelo multidimensional de dados baseado em XML é bastante flexível, podendo ser especificado de diversas maneiras. No entanto, a utilização de XML *Schema* e XLink permite a definição de um modelo de dados adaptável a um determinado domínio, além de possibilitar a inclusão de novas definições de fatos, dimensões e de seus relacionamentos. As abordagens X-Cube e XBRL *Dimensions* definem, com base em XML *Schema* e XLink, o modelo para o cubo de dados, definindo dimensões, fatos e cubos, assim como o modelo de dados para o documento XML, o qual possibilita a estruturação dos dados na instância XML. No entanto, estes dois trabalhos não abordam as questões relacionadas a consultas OLAP em dados XML.

Em *XQ-Cube*, discutido na Seção 4.3.1, é apresentado um modelo de *data warehouse* baseado em documentos XML e uma extensão da linguagem MDX, que permite a criação do cubo sobre o qual diversos tipos de consulta podem ser processados. O uso de XQuery na extensão proposta de MDX flexibiliza a realização de consultas, já que possibilita a manipulação tanto de dados numéricos quanto textuais. Porém, as questões relacionadas à heterogeneidade dos dados XML não são consideradas.

Na Seção 4.3.2, que contempla *Xaggregation*, observa-se que elementos de mesmo nome podem representar dimensões com caminhos diferentes. Por isto, foram definidas restrições para as expressões XPath, as quais avaliam os diversos valores, caminhos e composições hierárquicas das dimensões. Desta maneira, a operação de agregação em documentos XML é utilizada com expressões XPath que levam em conta as diferenças hierárquicas existentes. Porém, heterogeneidades semânticas não são abordadas.

Os trabalhos descritos nas Seções 4.3.3, 4.3.4 e 4.3.5 (IX-Cube, OLAP XML, X^3 Cube) requerem o conhecimento antecipado das possíveis estruturas hierárquicas. A necessidade de conhecimento prévio do esquema do documento e a realização de consulta em apenas um documento XML são desvantagens dessas abordagens, pois, para a análise das informações, uma grande quantidade de dados é fundamental, os quais normalmente estão distribuídos em diversos documentos. Este problema é agravado quando não se tem uma definição do modelo de dados usado para organizar os documentos.

A utilização de processamento baseado em valor e na estrutura do documento, discutida na Seção 4.3.4 (OLAP XML), diversifica as consultas do usuário. Flexibilidade também ocorre quando é utilizada a forma de consulta com sintaxe reduzida, pela combinação das possíveis estruturas que podem ocorrer no documento, sem seu prévio conhecimento. Contudo, a heterogeneidade semântica não é considerada.

A proposta de extensão de XQuery, descrita na Seção 4.3.6, é uma alternativa para padronização de consultas OLAP com base na especificação do W3C. No entanto, é restrito a consultas baseadas apenas no valor.

Embora XCube não considere o uso de documentos XML interconectados, ele representa cubo, dimensões e fatos. XBRL *Dimensions*, único dos trabalhos avaliados a considerar o uso de *links* em documentos XML, tem seu modelo de dados projetado para um domínio especifico de aplicação, o financeiro.

No Quadro 5.2 é apresentada uma comparação entre os trabalhos analisados no que se refere às questões do modelo de dados e ao tratamento dado à heterogeneidade existente em dados XML. Apesar de a heterogeneidade estrutural ter sido considerada nos trabalhos discutidos nas Seções 4.3.1, 4.3.4 e 4.3.5, (XQ-Cube, OLAP XML e X^3 Cube), ela é restrita a apenas um documento XML. Os trabalhos discutidos nas seções 4.2.1, 4.2.2 e 4.3.2 (XCube, XBRL *Dimensions* e Xaggregation) buscam solucionar esta questão para permitir a realização de consultas analíticas em mais de um documento, mas não abordam todas as questões de heterogeneidade presentes em dados XML. Os trabalhos discutidos nas Seções 4.3.1 e 4.3.6 (XQ-Cube e extensão de XQuery) não resolvem a heterogeneidade estrutural. Observa-se que as abordagens discutidas nas Seções 4.3.3, 4.3.4, 4.3.5 e 4.3.6 (IX-Cube, OLAP XML, X^3 Cube e extensão de XQuery) não propõem ou fazem uso de um modelo de dados para o cubo e para o documento XML. Apesar de XQ-Cube apresentar uma solução para a criação do cubo de dados, para o documento XML, não é definido um modelo a ser adotado.

Quadro 5.2 - Tratamento da heterogeneidade de dados XML nas propostas OLAP- XML.

Propostas Correlatas	**Heterogeneidade Semântica**	**Heterogeneidade Sintática de Conteúdo**	**Heterogeneidade Estrutural**	**Modelo de dados para o cubo de dados**	**Modelo de dados para o documento XML**	**Domínio de Aplicação**
X – Cube [41]	Não	Parcialmente	Sim	Sim	Sim	Qualquer
XBRL *Dimensions* [40]	Parcialmente	Parcialmente	Parcialmente	Sim	Sim	Financeiro
XQ – Cube [65]	Não	Não	Não	Sim	Não	Qualquer
Xaggregation [88]	Não	Não	Sim	Não	Não	Qualquer
IX – Cube [48]	Não	Não	Sim	Não	Não	Qualquer
OLAP-XML [14]	Não	Não	Sim	Não	Não	Qualquer
X^3 Cube [91]	Não	Não	Sim	Não	Não	Qualquer
Extensão de XQuery [11,12]	Não	Não	Não	Não	Não	Qualquer

O Quadro 5.3 apresenta uma comparação entre os trabalhos, no que diz respeito às consultas OLAP sobre documentos XML. Esta comparação é feita com base nas características consideradas relevantes para o desenvolvimento de uma solução OLAP para XML, abordadas na Seção 4. X-Cube e XBRL *Dimensions* não abordam as questões relacionadas às consultas OLAP, por isso, eles foram excluídos deste quadro comparativo. Os demais trabalhos possuem a característica comum de não considerarem o uso de *links* entre documentos XML, já que as tecnologias utilizadas, XPath e XQuery, para elaboração das soluções, não navegam entre documentos XML.

Quadro 5.3 - Características das consultas nas propostas OLAP – XML.

Proposta Correlata	Consulta baseadas no valor	Consulta baseadas na estrutura	Consulta sobre mais de um documento XML	Consulta sobre o cubo de dados	Consulta baseadas em MDX	Consulta em *Links* (XLink)
XQ – Cube [65]	Sim	Não	Sim	Sim	Sim	Não
Xaggregation[88]	Sim	Sim	Sim	Não	Não	Não
IX – Cube[48]	Sim	Sim	Não	Sim	Não	Não
OLAP-XML[14]	Sim	Sim	Não	Não	Não	Não
X^3 Cube[91]	Sim	Sim	Não	Não	Não	Não
Extensão de XQuery[11, 12]	Sim	Não	Sim	Não	Não	Não

A partir da análise e das conclusões apresentadas, infere-se a necessidade de uma solução que possua a capacidade de realização de consultas em documentos XML, considerando o relacionamento entre múltiplos documentos e a semântica que pode ser extraída de tais relações. Além disso, busca-se também uma abrangência que permita a aplicação da proposta em diversos domínios.

REFERÊNCIAS

1. ABITEBOUL, S. et al. **The Lorel Query Language for Semistructured Data**.

2. ABITEBOUL, S.; BUNEMAN, P.; SUCIU, D. **Gerenciando dados na WEB**. Rio de Janeiro: Ed. Campus, 2000.

3. Ahmedi, L. e Arifaj, M. **Processing XPath/XQuery to be Aware of XLink Hyperlinks.** 2nd European Computing Conference (ECC'08), Malta, September 11-13, 2008, pg. 217-221.

4. Ahmedi, L. e Arifaj, M. **Querying XML documents with XPath/XQuery in presence of XLink hyperlinks.** WSEAS Transactions on Computers, Issue 10, Volume 7, October 2008, pg 1752-1751.

5. Ahmedi, L. **Global Access to Interlinked XML Data Using LDAP and Ontologies**, PhD dissertation from Albert-Ludwigs, Freiburg, 2001.

6. Ahmedi, L. **Making XPath Reach for the Web-Wide Links.** 2005 ACM Symposium on Applied Computing, March 13-17, 2005, Santa Fe, New Mexico, USA, pg 1714-1721.

7. ALASHQUR, A. M.; SU, S. Y. W.; LAM, H. **OQL: a query language for manipulating object-oriented databases**. Proceedings of the 15th international conference on Very large data bases. 1989. p. 433 - 442.

8. BARIL, X.; BELLAHSÈNE, Z.: **Designing and Managing an XML Warehouse**. In: XML Data Management: Native XML and XML-Enabled Database Systems. ed. Addison Wesley Professional. 2004, pp. 455–474

9. **BATAVIA XBRL Java Library (BXJL)**. Disponível em: http://www.batavia-xbrl.com/. Acesso em : janeiro de 2008.

10. BERGERON, B. **Essentials of XBRL - Financial Reporting in the 21st Century**, New York: Jonh Wiley & Sons,Inc., 2003.

11. BEYER, K. et al. **XQuery for Analytics: Challenges and Requirements**. Workshop on XQuery Implementation Experience and Perspectives XIMEP. 2004, p. 3-8.

12. BEYER, K. et al. **Extending XQuery for Analytics**. International Conference on Management of Data, Baltimore, Maryland .p. 503 – 514, Year of Publication: 2005.

13. BOIXO, I.; FLORES, F. **New Technical and Normative Challenges for XBRL: Multidimensionality in the COREP Taxonomy**. The International Journal of Digital Accounting Research. v. 5, n. 9, p. 79-104, may 2005. ISSN: 1577-8517

14. BORDAWEKAR, R. R.; LANG, C. A. **Analytical Processing of XML Documents: Opportunities and Challenges**. SIGMOD Record, v. 34, n. 2, June 2005.

15. BOUSSAID, O.; MESSAOUD, R. B.; CHOQUET, R.; ANTHOARD, S. **X-Warehousing: An XML-Based Approach for Warehousing Complex Data**. East-European Conference on Advances in Databasesand Information Systems (ADBIS), LNCS 4152, p. 39-54. Springer-Verlag Berlin Heidelberg 2006.

16. BOUSSAID, O.; TANASESCU, A.; BENTAYEB, F.; DARMONT, J. **Integration and dimensional modeling approaches for complex data warehousing**. Springer Science+Bussiness Media B.V. 2006.

17. BRY, F. e ECKERT, M. **Processing Link Structures and Linkbases in the Web's Open World Linking**. Proceedings of sixteenth ACM Conference on Hypertext and Hypermedia, Salzburg, Austria, 2005.

18. CAVALIERI, Federico; GUERRINI, Giovanna; MESITI, Marco. **Navigation Path Expressions on XML Schema**. Turin: 19th International Conference on Database and Expert Systems Applications – DEXA, 2008.

19. CHAMBERLIN, D.; ROBIE, J.; FLORESCU, D. **Quilt: an XML Query Language for Heterogeneous Data Sources**. In: Lecture Notes in Computer Science. Springer-Verlag, dec. 2000. Also available at: http://www.almaden.ibm.com/cs/people/chamberlin/quilt_lncs.pdf. Acesso em: janeiro de 2008.

20. CHAUDHURI, S. ; DAYAL, U**. Data warehouse and OLAP for decision support**, in Tutorials of the Twenty-Second international Conference On Very Large Data Base, Bombay.1996, p. 295-30.

21. _____. **An overview of data warehousing and olap technology**. SIGMOD Record 26(1), 1997, p. 65–74.

22. DARMONT, J.; BOUSSAÏD , O.; RALAIVAO, J.C; AOUICHE, K. SMAiDoC: **Un Système Multi-Agents pour l'Integration des Données Complexes**. Reveu des Nouvelles Technologies de L'Information, n. 1, 2003, p. 13-24.n.

23. **DATABASE LANGUAGES – SQL, ISO/IEC 9075:2003**. Disponível em: http://www.iso.org/iso/catalogue_detail.htm?csnumber=34132. Acesso em: janeiro de 2008.

24. Decision Support Database Group (BDD) Research project 2006. **XML Complex Data Warehousing Plataform**. Disponível em: http://bdd.univ-lyon2.fr.. Acesso em: janeiro de 2008.

25. DEUTSCH, A.; FERNANDEZ, M., FLORESCU, D.; LEVY, A.; SUCIU, D. **XML-QL: a query language for XML**. 1999. In Proceedings of the International World Wide Web Conference.

26. ECCLES, R.G.; WATSON, Liv; WILLIS, M. **Here Comes XBRL**, Harvard Bussines Review, Frebuary 2007. Disponível em: http://harvardbusinessonline.hbsp.harvard.edu/hbrsa/en/issue/0702/article/R0702A.jhtml;jsessionid=FYU4PU4SIN41UAKRGWCB5VQBKE0YOISW?type=F#section10. Acesso em: março de 2007.

27. EISENBERG, A.; KULKARNI, K.; MELTON, J.; MICHELS, Jan-Eike and ZEMKE, F. **SQL: 2003 Has Been Published**. SIGMOD Record. v. 33, v. 1, March 2004.

28. ELMASRI, R.; NAVATHE, S. B. **Sistemas de Banco de Dados**. Pearson Education do Brasil, 2005. ISBN 85-88639-17-3.

29. ENGEL, E.; HAYES, R. M.; WANG, X. **The Sarbanes–Oxley Act and firms' going-private decisions**. In: Journal of Accounting and Economics. 2007, p. 116-145.

30. ENGEL, P. et al. **Extensible Business Reporting Language (XBRL) 2.1** (XBRL recommendation). 2003. Disponível em: http://www.xbrl.org/Specification/XBRL-RECOMMENDATION-2003-12-31+Corrected-Errata-2005-11-07.htm>. Acesso em: março de 2007.

31. **eXist - Open Source XML Native Database**, 2009. Disponível em : http://exist.sourceforge.net/. Acesso em: setembro de 2009.

32. F. Ravat, O. Teste, R. Tournier, G. Zurfluh. **Finding an application-appropriate model for XML data warehouses**. Information Systems, n. 35, 2010, pp. 662–687.

33. FELDEN, C. **Multidimensional XBRL**. (2007) New Dimensions of Business Reporting and XBRL. Publisher DUV Part 4 pp 191-209 ISBN 978-3-8350-0835-9.

34. FIDALGO. R. N. **JDCI: Uma API Java para disponibilização e integração de serviços OLAP**. 2000, Dissertação (Mestrado em Ciência da Computação) , Cin/UFPE, 2000.

35. GATZIU, S.; VAVOURAS, A. **Data Warehousing: Concepts and Mechanisms**. Informatik – Informatique 1/1999, p. 8-11.

36. GOLFARELLI, M.; RIZZI, S.; VRDOLJAK, B. **Data Warehouse Design from XML Sources**. In: Proceedings of the 4th ACM International Workshop on Data Warehousing and OLAP (DOLAP 2001), Atlanta, Georgia, USA, ACM Press, 2001 p. 40–47.

37. GRAVES, M. **Projeto de Banco de Dados com XML**. Pearson Education do Brasil. ISBN 85-346-1471-7., 2003.

38. GROSSO, P. et al. **XPointer Framework** W3C Recommendation, 25 March 2003. Disponível em: http://www.w3.org/TR/2003/REC-xptr-framework-20030325/. Acesso em: janeiro de 2008.

39. GUSFIELD, D. **Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology**. Cambridge University Press, January 1997.

40. HERNÁNDEZ-ROS, I.;WALLIS, H. **XBRL Dimensions 1.0**. 2006. Disponível em : http://www.xbrl.org/Specification/XDT-REC-2006-09-18.htm. Acesso em: setembro de 2006.

41. HÜMMER, W.; BAUER, A.; HARDE, G. **XCube – XML for Data Warehouses**. Proc. The 6th ACM Intl Workshop on Data Warehousing and OLAP (DOLAP03). New Orleans, Louisiana, USA, p. 33–40, 2003.

42. INMON, W. **Building the DataWarehouse**. 4th edn. John Wiley and Sons, 2005.

43. JENSEN, M.R.; MOLLER, T.H.; PEDERSEN, T.B. **Specifying OLAP Cubes On XML Data**. In: Tecnichal Report o2-5003. Departament of Computer Science, Alborg University. June 2001.

44. JIAN, F. M. et al. **IX-Cubes: Iceberg Cubes for Data Warehousing and OLAP on XML Data**. In: CIKM'07, nov., p. 6-8, 2007, Lisboa, Portugal.

45. KIMBALL, R. **A Dimensional Manifesto, DBMS**. ago. 1997.

46. KIMBALL, R.; CASERTA, J. **The Data Warehouse ETL Toolkit**. John Wiley, Sept, 2004.

47. KIMBALL, R.; ROSS, M. **The DataWarehouse Toolkit**. John Wiley and Sons, 2002.

48. L. FINDLATER, H.J. HAMILTON, **Iceberg-cube algorithms: an empirical evaluation on synthetic and real data**, Journal of Intelligent Data Analysis 7 (2) (2003) 77–97 IOS Press.

49. LAURENT, S. S. **XLinkFilter: An Open Source Java XLink SAX Parser Filter**. 1998. Avaliable at http://www.simonstl.com/projects/xlinkfilter/index.htm. Acesso em: janeiro de 2008.

50. LENZ, H. J.; SHOSHANI, A. **Summarizability in OLAP and Statistical Databases**. In: SSDBM, 2001.

51. LIZORKIM, D.A. ; LISOVSKY, K. Yu. **XSLT and XLink and their implementation with functional techniques**. Russian Digital Library, v. 5, issue 2, 2003.

52. LIZORKIM, D.A. **The Query Language to XML Documents Connected by XLink Links**. In: Programming and Computer Software. v. 31, n. 3, 2005, p 133-148.

53. _____. **SXML: an XML document as an S-expression**, Russian Digital Library, v. 6, issue 2, 2003.

54. _____. **XML Path Language (XPath) and its functional implementation SXPath**. Russian Digital Library, v. 6, issue 4, 2003.

55. _____. **Implementation of the XML Linking Language XLink by Functional Methods**. In: Programming and Computer Software. v. 31, n. 1, 2005, p. 34-46.

56. May, W., Behrends, E. e Fritzen, O. **Integrating and Querying Distributed XML Data via XLink**, Information Systems, to appear, 2008.

57. MAY, W.; MALHEIRO, D. **LoPiX: A System for XML Data Integration and Manipulation**. Proceedings of the 27th VLDB Conference, Roma, Italy, 2001.

58. _____. **Querying Linked XML Document Networks in the Web**. Proceedings of the 11th International World Wide Web Conference (WWW 2002).

59. _____. **A Logical, Transparent Model for Querying Linked XMLDocuments**, In: GI-Fachtagung Datenbanksysteme fur Business, Technologie und Web (BTW 2003), Febuary.

60. **MDX**. 2007. Disponível em: technet.microsoft.com/en-us/library/ms145506.aspx. Acesso em: janeiro de 2008.

61. MESSAOUD, R. B.; BOUSAÏD, O.; LOUDCHER, S. **A data Mining-Based OLAP Aggregation of Complex Data: Application on XML Documents**. International Journal of Data Warehousing and Mining. 2006.

62. NÄPPILÄ, T.; JÄRVELIN, K.; NIEMI, T. **A tool for data cube construction from structurally heterogeneous XML documents.** Journal of the American Society for Information Science and Technology (JASIST), v. 59, Issue 3, Date: 1 February 2008, p. 435-449.

63. NASSIS, V.; RAJUGAN, R.; DILLON, T. S.; RAHAYU, W. **Conceptual Design of XML Document Warehouses**. In: Proc. Data Warehousing and KnowledgeDiscovery, 6th International Conference, DaWaK 2004, p. 1–14, Zaragoza, Spain, 2004.

64. NENTWICH, C.; CAPRA, L.; EMMERICH, W.; FINKELSTEIN, A. **Xlinkit: a consistency checking and smart link generation service**. ACM Transactions on Internet Technology, 2 (2), 2002 p. 151-185.

65. PARK, B. K.; HAN, H.; SONG, I.Y. **XML-OLAP: A Multidimensional Analysis Framework for XML Warehouses**. 7th International Conference in Data Warehousing and Knowledge Discovery (DaWaK 2005), LNCS 3589, p. 32-42, 2005. Springer-Verlag Berlin Heidelberg, 2005.

66. PEDERSEN, D.; RIIS, K.; PEDERSEN, T.B. **XML – Extended OLAP Querying**. In: Tecnichal Report o2-5001, Departament of Computer Science, Alborg University. May 2002.

67. _____. **A Powerful and SQL-Compatible Data Model and Query Language for OLAP**. In: Proceedings of the Thirteenth Australasian Database Conference, p. 121.130, 2002.

68. _____. **Achieving Adaptivity for OLAPXML Federations**. In: DOLAP'03, November 7, 2003, New Orleans, Louisiana, USA.

69. _____. **Integration of XML Data in the Targit OLAP System**. Data Engineering, 2004. Proceedings. 20th International Conference on Volume , Issue , 30 March-2 April 2004.

70. _____. **Synchronizing XPath Views**. Database Engineering and Applications Symposium, 2004. IDEAS apos;04. Proceedings. International Volume , Issue , 7-9 July 2004, p. 149 - 160.

71. PIECHOCKI, M.; FELDEN, C.; GRÄNING, A. (2007) **Multidimensional XBRL Reporting**. Technische Universität Bergakademie Freiberg, Lessingstraße 45, 09599 Freiberg, Germany.

72. POKORNY, J. **Modelling Stars Using XML**. In: Proceedings of the 4th ACM International Workshop on Data Warehousing and OLAP (DOLAP 2001), Atlanta, Georgia, USA, ACM Press, p. 24-31, 2001.

73. RAFANELLI, M. **Multidimensional Databases: Problems and Solutions**. Idea Group Publishing, 200. ISBN 1-591140-053-8.

74. REFSNES, I. E. **Introduction to DTD**. Disponível em: http://www.xmlfiles.com/dtd/dtd_intro.asp. Acesso em: janeiro de 2010.

75. RESNIK, P. **Using information content to evaluate semantic similarity in a taxonomy**. In: Proceedings of IJCAI, p. 448-453, 1995.

76. ROBIE, J.; LAPP, J.; SCHACH, D. **XML Query Language (XQL)**, 1998. Disponível em: http://www.w3.org/TandS/QL/QL98/pp/xql.html. Acesso em: janeiro de 2008.

77. **SAX version 2.0.2 Simple API for XML**. Disponível em: http://www.saxproject.org/. Acesso em: janeiro de 2008.

78. **S-exp-based XML parsing/query/conversion**. Disponível em: http://ssax.sourceforge.net/. Acesso em: Janeiro de 2008.

79. SILVA, P. C; AQUINO, I. J. S.; SIQUEIRA, E.; FIDALGO, R.; TIMES, V.C. **Uma visão multidimensional para informações financeiras na web: XBRL Dimensions**. Proc. 4th CONTECSI International Conference on Information Systems and Technology Management. 30 May-01 June, 2007 USP/São Paulo/SP.

80. SWANSON, Z.; DURLER, G.; REMINGTON, W. **How Do Firms Address Multiple Taxonomy Issues? New Dimensions of Business Reporting and XBRL**. Publisher DUV Part 4 pp 191-209 ISBN 978-3-8350-0835-9, 2007.

81. THOMSEN, E. **OLAP construindo sistemas de informações multidimensionais**. Ed. Campus, 2002.

82. TRUJILLO, J.; LUJ` an-MORA, S.; SONG, I. **Applying UML and XML for Designing and Interchanging Information for Data Warehouses and OLAP Applications**. Journal of Database Management 15(1), 2004, p. 41–72.

83. V. NASSIS, R. RAJAGOPALAPILLAI, T.S. DILLON, J.W. RAHAYU, **Conceptual and systematic design approach for XML document warehouses**, International Journal of Data Warehousing & Mining (ijDWM) 1 (3) (2005) 63–87 Idea Group Publishing.

84. V. NASSIS, T.S. DILLON, R. RAJAGOPALAPILLAI, J.W. RAHAYU, **An XML document warehouse model**, in: 11th International Conference on Database Systems for Advanced Applications (DASFAA), LNCS 3882, Springer, 2006, pp. 513–529.

85. W3 Schools. **XML Schema ComplexType Element**. Available at: http://www.w3schools.com/Schema/el_complextype.asp.

86. WANG, H.; LI, J.; HE, Z.; GAO, H. **Xaggregation: Flexible Aggregation of XML Data**. In Advances in Web_Age Information Management, 4th International Conference, WAIM, 2003, p. 104-115, 2003.

87. _____. OLAP for XML Data. Proceedings of the 2005 The Fifth International Conference on Computer and Information. Shanghai, China. ISBN: 0-7695-2432-X.

88. _____. **Flexible and Effective Aggregation operator for XML Data**. In Information Technology Journal 6 (5), Asian Network for Scientific Information, p. 697-703, 2007.

89. WIWATWATTANA, N.; JAGADISH, H.; LAKSHMANAN, L.; SRIVASTAVA, D. **X^3: A cube operator for xml olap**. In: Proc. 2007 Int. Conf. Data Engineering (ICDE'07), 2007.

90. WREMBEL, R.; KOMCILIA, C. **DataWarehouses and OLAP Concepts**, Architeture and Solutions. IRM Press, 2007. ISBN 1-59904-364-5.

91. **XBRL Consortium**. Disponível em: http://www.xbrl.org. Acesso em: abril de 2006.

92. **XBRLAPI Java XBRL API implementation**, version 3. 2007. Disponível em: http://www.xbrlapi.org/. Acesso em: janeiro de 2008.

93. **XLiP (XLink Processor), User's Guide**. 2005. Available at: .http://software.fujitsu.com/en/interstage-xwand/activity/xbrltools/xlip/index.html. Acesso em: janeiro de 2008.

94. **XML Information Set (Second Edition) W3C Recommendation** 4 February 2004. Disponível em: http://www.w3.org/TR/2004/REC-xml-infoset-20040204. Acesso em: janeiro de 2008.

95. **XML Linking Language (XLink)**, version 1.0 (w3c recommendation), 2001. Disponível em: http://www.w3.org/TR/xlink. Acesso em: abril de 2006.

96. **XML Path language (XPath) version 2.0 (w3c recommendation)**, 2007. Disponível em: http://www.w3c.org/tr/xpath20/. Acesso em: abril de 2006.

97. **XML Schema Part 1: Structures. version 1.0 (w3c recommendation)**. 2004. Disponível em: http://www.w3.org/TR/xmlschema-1. Acesso em: abril de 2006.

98. **XQuery 1.0: An xml query language, version 1.0 (w3c recommendation)**. 2007. Disponível em: http://www.w3.org/TR/xquery. Acesso em: janeiro de 2008.

99. **XSL Transformations (XSLT), version 1.0 (w3c recommendation)**, 1999.

100. XYLEME, L. **A dynamic warehouse for xml data of the web**. In: IEEE Data Engineering Bulletin. Database Engineering & Applications, International Symposium. Grenoble, France: 2001

101. YIN, X.; PEDERSEN, T. B. **Algebra-Based Optimization of XML-Extended OLAP Queries**. International

102. Conference on Management of Data COMAD 2006, Delhi, India, December 14–16, 2006. Computer ociety of Indi