



**DAIANE ROBERTA CANDIDA**

**AVALIAÇÃO DE IMPLEMENTAÇÕES DA  
RECOMENDAÇÃO XQUERY FULL TEXT EM  
SGBD's XML**

**Lavras – MG**

**2014**

**DAIANE ROBERTA CANDIDA**

**AVALIAÇÃO DE IMPLEMENTAÇÕES DA RECOMENDAÇÃO  
XQUERY FULL TEXT EM SGBD's XML**

Monografia apresentada ao  
colegiado do Curso de Sistemas  
de Informação, para a obtenção  
do título de Bacharel em  
Sistemas de Informação.

**ORIENTADOR**

Dr. Leonardo A. Ribeiro.

**COORIENTADOR**

Dr. Denilson Alves Pereira

**Lavras – MG**

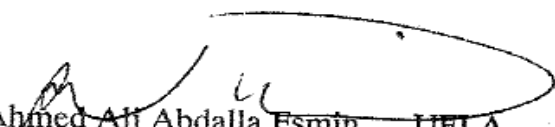
2014

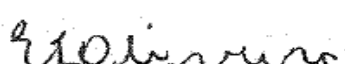
**DAIANE ROBERTA CANDIDA**

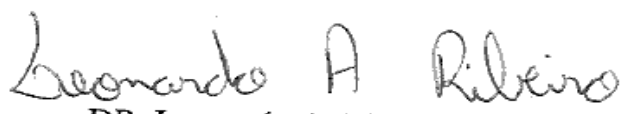
**AVALIAÇÃO DE IMPLEMENTAÇÕES DA RECOMENDAÇÃO  
XQUERY FULL TEXT EM SGBD's XML**

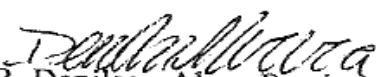
Monografia apresentada ao  
colegiado do Curso de Sistemas  
de Informação, para a obtenção  
do título de Bacharel em  
Sistemas de Informação.

APROVADA em 04 de Julho de 2014.

  
Dr. Ahmed Ali Abdalla Esmin UFLA

  
Me. Erasmo Evangelista de Oliveira UFLA

  
DR. Leonardo A. Ribeiro.  
(Orientador)

  
DR. Denilson Alves Pereira  
(Coorientador)

Lavras – MG

2014

## **VALIAÇÃO DE IMPLEMENTAÇÕES DA RECOMENDAÇÃO**

### **RESUMO**

Estamos vivenciando o crescimento da linguagem de marcação denominada XML (eXtensible Markup Language) juntamente com o avanço da Web. Uma característica do modelo de XML é a capacidade de representar dados semiestruturados. Dados desta natureza são caracterizados por conterem porções estruturadas, tais como tabelas do modelo relacional, e não estruturados, tais como texto livre. A representação de dados semiestruturados é extremamente importante porque um número cada vez maior de aplicações produzem e utilizam estes tipos de dados. Uma linguagem de consulta sobre dados semiestruturados deve prover meios para especificação tanto de condições estruturais rígidas quanto condições imprecisas ou vagas. No contexto de dados XML com consultas baseadas em palavras-chave atualmente existe um grande número de implementações do XQuery Full Text em SGBD's XML comerciais e de código livre. Entretanto, a literatura ainda carece de uma análise destes produtos. Com foco em sistemas de código livre e gratuitos, este trabalho apresenta uma análise de SGBD's com suporte ao XQuery Full Text no contexto de facilidade de utilização por pessoas sem treinamento especializado, tais como: estudantes, funcionários de empresas ou órgão público. Os SGBD's analisados foram o BaseX, MXQuery, IBM DB2 Express e o ORACLE XMLDB Express. Foram utilizados como critério de análise: instalação, criação de índice, funcionalidade, desempenho e suporte técnico.

**PALAVRAS-CHAVES:** XQuery Full Text, SGBD's e XML.

## LISTA DE FIGURAS

Figura 1- DTD do xml .....	7
Figura 2- documento XML .....	9
Figura 3- Árvore XML .....	10
Figura 4- sequência .....	11
Figura 5- Consulta de predicado .....	12
Figura 6- Consulta FLWOR .....	14
Figura 7- Consulta FTContainsExpr .....	15
Figura 8- Processo de indexação.....	19
Figura 9- Processamento de consulta.....	28
Figura 10 - Consultas de Funcionalidade.....	38

## LISTA DE ABREVIATURAS

DTD - *Document Type Definition*

ER - Entidade-Relacionamentos

HTML - *Hyper Text Markup Language*

SGBD's - Sistemas Gerenciadores de Banco de Dados.

SGBDR's - Sistemas Gerenciadores de Banco de Dados Relacionais

SQL - *Structured Query Language*

URL - Uniform Resource Locator

W3C - *World Wide Web Consortium*

XML - *eXtensible Markup Language*

XQFT – XQuery Full Text

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO.....</b>	<b>1</b>
<b>2</b>	<b>REFERENCIAL TEÓRICO .....</b>	<b>4</b>
<b>2.1</b>	<b>XML: Importância e diferença em relação ao modelo relacional .....</b>	<b>4</b>
<b>2.2</b>	<b>Composição de Dados XML.....</b>	<b>6</b>
<b>2.3</b>	<b>XQuery: Linguagem de Consulta para XML.....</b>	<b>10</b>
<b>2.4</b>	<b>Modelo de Dados do XQuery .....</b>	<b>11</b>
<b>2.5</b>	<b>XPath .....</b>	<b>12</b>
<b>2.6</b>	<b>Expressões XQuery .....</b>	<b>12</b>
<b>2.7</b>	<b>XQuery Full Text: Extensão do XQuery para Lidar com Texto.....</b>	<b>14</b>
<b>3</b>	<b>SGBD'S XML.....</b>	<b>15</b>
<b>3.1</b>	<b>Oracle XML DB.....</b>	<b>16</b>
<b>3.1.1</b>	<b>Oracle suporte ao XQuery Full Text.....</b>	<b>16</b>
<b>3.1.2</b>	<b>Carga e Armazenamento de Dados e Processamento de Consultas 18</b>	
<b>3.1.3</b>	<b>Limitações do Oracle XE.....</b>	<b>21</b>
<b>3.2</b>	<b>SGBD XML Basex .....</b>	<b>22</b>
<b>3.2.1</b>	<b>Basex suporte ao XQuery Full Text .....</b>	<b>22</b>
<b>3.2.2</b>	<b>Carga e Armazenamento de Dados e Processamento de Consultas 24</b>	

3.2.3	Basex limitações .....	25
3.3	MXQUERY .....	26
3.3.1	MXQUERY suporte ao XQuery Full Text .....	27
3.3.2	Carga e Armazenamento de Dados e Processamento de Consultas 28	
3.3.3	MXQUERY limitações .....	30
3.4	SGBD IBM DB2 .....	30
3.4.1	IBM DB2 Suporte ao XQuery Full Text .....	31
3.4.2	Carga e Armazenamento de Dados e Processamento de Consultas 32	
3.4.3	IBM DB2 limitações .....	33
4	METODOLOGIA .....	34
4.1	Classificação .....	34
4.2	Hardware .....	35
4.3	Instalação .....	36
4.4	Importação .....	36
4.5	Teste de Funcionalidade .....	37
4.6	Teste de Desempenho .....	40
4.7	Métricas utilizadas .....	42
5	RESULTADOS .....	44
5.1	Criação de Índices .....	45
5.2	Funcionalidade .....	46



<b>5.3</b>	<b>Desempenho.....</b>	<b>48</b>
<b>5.4</b>	<b>Suporte.....</b>	<b>48</b>
<b>5.5</b>	<b>Sumário dos Resultados .....</b>	<b>49</b>
<b>6</b>	<b>CONCLUSÃO.....</b>	<b>51</b>
<b>7</b>	<b>REFERÊNCIAS.....</b>	<b>52</b>
<b>1</b>	<b>APÊNDICES .....</b>	<b>59</b>
<b>2</b>	<b>ANEXOS.....</b>	<b>67</b>

## 1 INTRODUÇÃO

Diante da dificuldade de sistemas gerenciadores de banco de dados relacionais (SGBDR's) em armazenarem dados semiestruturados e não estruturados, foram desenvolvidas ferramentas que armazenam e processam informações não estruturadas, motivo pelo qual é interessante mensurar quão eficazes são tais ferramentas.

Em SGBDR's, antes da carga dos dados, é especificado um esquema descrevendo a estrutura desses dados, mas isto não impõe limitações significativas sobre sistemas tradicionais, que lidam com dados com uma estrutura rígida. Exemplos de domínios de aplicação populares nesse contexto são sistemas de folhas de pagamento, controle de estoque, folha de ponto, entre outros. Por outro lado, a necessidade de uma estrutura prévia para armazenar dados restringe-nos diante da vasta gama de informações que encontramos nos dias de hoje e se encontram representadas em formatos heterogêneos.

Atualmente as empresas utilizam aplicações modernas onde elas têm que lidar com dados onde há estrutura regular ou não. Por exemplo, em um sistema de segurança, juntamente com campos para entradas padronizadas como vítima, local e horário, pode haver um campo texto para entrar com a descrição de testemunhas. Outro exemplo são sistemas para suporte ao consumidor, onde a descrição estruturada de um produto pode estar associada com texto livre representando comentários ou reclamações de consumidores.

Neste contexto, XML é o padrão mais seguido para representação de dados e XQuery é a linguagem de consulta sobre dados XML, tal como SQL é a linguagem de consulta sobre dados relacionais. Para atender a demanda dessas aplicações, vários sistemas gerenciadores de banco de dados (SGBD's) comerciais ou de código aberto foram construídos ou adaptados para suportar nativamente XML e XQuery. (HAUSTEIN, 2007 e BEYER, 2005)

Apesar de permitir a especificação de consultas explorando certas características estruturais, o suporte à consultas destinadas à porções não-estruturadas de dados XML ainda é limitado em XQuery. Por exemplo, não é possível consultar porções de texto livre em um documento XML usando palavras-chave.

Recentemente, uma extensão de XQuery chamada XQuery Full Text foi criada para permitir a especificação de consultas baseadas tanto na estrutura de dados XML quanto em palavras-chave, de acordo com grupo internacional desenvolve padrões para Web, World Wide Web Consortium - W3C. Em particular, XQuery Full Text permite a exploração de técnicas de Recuperação de Informação em XML's, ponderação de termos e contagens de palavras em consultas sobre dados XML. (BAEZA-YATES e RIBEIRO-NETO 2011)

O suporte à XQuery Full Text tem sido divulgado por vários SGBD's tais como: BASEX , IBM DB2, ORACLE XMLDB e MXQUERY. Entretanto, pouco ainda é conhecido a respeito do desempenho e do grau de adequação dos SGBD's à recomendação XQuery Full Text da W3C. Este trabalho visou realizar uma análise em SGBD's de código livre e gratuitos no contexto de facilidade de utilização por pessoas sem treinamento especializado, tais como: estudantes, funcionários de empresas ou órgão público.

As empresas, no contexto da sociedade em geral, se beneficiarão da presente pesquisa porque com um SGBD que suporta XQuery Full Text a empresa poderá representar de forma mais completa os dados complexos gerados no seu dia-a-dia.

Na realidade, as pequenas empresas ou órgãos públicos e privados não dispõem de recursos financeiros para investir em ferramentas caras, e por outro lado, as grandes empresas se resguardam de investimentos em tecnologias que possam coloca-las em risco operacional/financeiro ou porventura não lhes

tragam benefício algum. É interessante ter conhecimento da ferramenta antes da aquisição.

As versões gratuitas das ferramentas proprietárias ajudam a conhecer o produto porque dá para testá-lo, antes de pagar pela licença, já que foco deste trabalho está na examinação das ferramentas gratuitas. Assim, as empresas que se encontram com dificuldades em escolher entre os SGBD's disponíveis no mercado terão dados que servirão de parâmetro para dar suporte na escolha da ferramenta.

O objetivo geral do presente trabalho foi realizar uma análise de implementações gratuitas da recomendação XQuery Full Text em SGBD's XML, onde os aspectos considerados na análise foram desempenho, funcionalidades, suporte técnico e compatibilidade com a recomendação XQuery Full Text.

- Os objetivos específicos se dividiram em:
- Estudar o suporte à funcionalidades do XQuery Full Text em SGBD's XML comerciais e de código aberto.
- Construir de um ambiente de testes para comparação entre SGBD's XML que caracterize cenários reais de aplicações baseadas em dados semiestruturados.
- Adaptar modelos de consultas XQuery Full Text sobre coleção de dados XML . (GRAY, 1993)
- Avaliar e comparar o desempenho e as funcionalidades dos SGBD's XML que suportem XQuery Full Text, de código livre e proprietário.

Este trabalho se encontra organizado da seguinte forma. Na Seção 2 REFERENCIAL TEÓRICO foram elencadas as bases bibliográficas para reforçar a nossa discussão, na Seção 3 SGBD's XML listamos as características dos SGBD's estudados, na Seção 4 está a METODOLOGIA de como o

trabalho foi desenvolvido, na seção 5 estão os RESULTADOS obtidos após o desenvolvimento e 6 CONCLUSÃO.

## **2 REFERENCIAL TEÓRICO**

Este capítulo apresenta os conceitos básicos dos temas relacionados com desenvolvimento deste trabalho.

### **2.1 XML: Importância e diferença em relação ao modelo relacional**

Em SGBDR's, os dados ficam armazenados em tabelas distribuídos em linhas, colunas e objetos. Esses sistemas trabalham com dados estruturados, e é necessário ter informações *a priori* a respeito dos mesmos, tais como:

- estrutura e propriedades das informações a serem modeladas;
- como se dará o relacionamento entre as entidades;
- qual vocabulário será utilizado pelas pessoas envolvidas na manipulação destes dados.

Diante da necessidade de armazenar dados que não estão estruturados da forma descrita acima, o modelo relacional passou a ser insuficiente diante das novas realidades das informações que classificaremos como dados semiestruturados e não estruturados.

Dados semiestruturados são dados que possuem uma certa estrutura, mas esta estrutura não é rígida e regular como no modelo relacional. São chamados semiestruturados porque os dados são brutos, sem tipificação ou organização aparente, e são desprovidos de estrutura.

A Web fornece exemplos de dados semiestruturados; nela os dados consistem de arquivos em um particular formato, como HTML, com alguma estruturação primitiva tais como tags e âncoras. Um outro exemplo de dados

semiestruturados temos um blog na Web, normalmente eles não têm uma estrutura explícita, porque ele é carregado ou abastecido pelo autor e seguem comentários de outros participantes sem que haja qualquer padrão de representação podendo conter texto, imagem, vídeos ou sons cujos arquivos têm extensão .doc, .txt, .pdf, entre outros.

Os tipos de documentos citados acima estão representados normalmente em linguagem natural, ou seja, a informação não está estruturada (ABITEBOUL, 1997).

Para lidar com dados desestruturados o modelo XML apresenta vantagens como: flexibilidade em representar dados complexos, heterogêneos e independentes. De acordo com Howard Katz (2003), as diferenças entre dados relacionais e dados XML são:

- Dados relacionais são planos, isto é, são organizados em linhas e colunas ao contrário dos XML que são aninhados e podem ser apresentar profundidade arbitrária (FLORESCU, 2005).
- Dados relacionais podem representar estruturas de dados aninhados usando hierarquia de tipos ou tabelas com chaves estrangeiras. Entretanto, a busca nessas estruturas não é fácil devido ao desconhecimento da profundidade do aninhamento, ao contrário do XML que permite consultar um objeto num documento, mesmo desconhecendo sua estrutura.
- Dados relacionais são regulares e homogêneos, cada linha da tabela tem os mesmos nomes e tipos (real, inteiros, caracteres entre outros), ao contrário dos XML que são irregulares e heterogêneos, a organização dos dados é descrita através de etiquetas (tags), e não por tipos de dados descritos em um esquema separado dos dados em si. Em outras palavras, a informação em XML é representada como uma hierarquia de

elementos que têm nomes, atributos opcionais e conteúdos opcionais podendo ser constituídos de texto, elementos aninhados ou pela mistura destes.

- A busca numa base relacional resulta em registros com dados do mesmo tipo e organização regular, ao passo que em busca por XML o resultado consistirá em expressões com dados de vários tipos heterogêneos e organização possivelmente irregular.
- Em base de dados relacional toda linha e toda coluna deve conter um valor, caso contrário o valor NULL é utilizado para representar ausência (ou desconhecimento) da informação relacionada, ao contrário de XML que os dados desconhecidos ou inaplicáveis simplesmente não aparecem.
- Em uma base de dados relacional não são consideradas a ordem das linhas da tabela, uma vez que se pode estabelecer qualquer ordenação pelos valores dos dados sem que seja prejudicada sua significância, ao contrário do XML que a ordem dos dados tem um significado.

A seguir, serão apresentados mais detalhes sobre o modelo de dados XML e seu emprego na representação de dados semiestruturados.

## 2.2 Composição de Dados XML

Os dados XML podem ser formados por meio de dois modelos, são eles os *Data-centric* e os *Document-centric*. Dados *Data-centric* têm a estrutura relativamente estática e previsível, aplicações tiram proveito desta estrutura, os dados tem o formato de um *Schema* XML, enfim, este tipo é um modelo de dados que decompõe o documento XML em uma entidade e relacionamento - ER e a indexação é uma árvore B. *Document-centric* cujos dados estão

relativamente estruturados sua indexação é feita pelo índice XML. As aplicações não tiram proveito de sua estrutura tratando os dados como se eles fossem desprovidos de estrutura (ORACLE, 05/2013).

No modelo de dados XML cada documento é representado como uma árvore de nós, onde cada nó tem uma identidade que o distingue dos outros nós. Os tipos de nós que podem ocorrer são: documento, elemento, atributo, texto, espaço\_nome (namespace), instrução de processamento e comentário. (HOWARD KATZ, 2003)

A partir de um DTD – Document Type Definition, podemos apresentar a estrutura de um documento XML. Por exemplo, no caso de um documento contendo uma bibliografia, é possível especificar que o mesmo é constituído por uma sequência de livros, cada livro tem um título, ano de publicação, um autor ou editor, e um editor tem uma afiliação. A Figura 1 é a representação de um DTD.

Figura 1- DTD do XML

Exemplo de DTD:

```
<!ELEMENT books (book* )>
<!ELEMENT books (number )>
<!ELEMENT book (title shortTitle, (author+ | editor+ ), content (p, note)>
<!ATTLIST book number CDATA #REQUIRED >
<!ELEMENT author>
<!ELEMENT editor>
<!ELEMENT title (#PCDATA )>
<!ELEMENT shortTitle (#PCDATA )>
<!ELEMENT content (#PCDATA )>t
```

De acordo, com a ordem de documento do modelo de dados XML, cada elemento aparece uma única vez de forma que ordenando os mesmos se remove



as duplicatas. Ele é uma ordenação que está definida entre todos os nós no modelo de consulta de dados de um documento, correspondendo à ordem de representação de vários nós, como eles seriam encontrados se o documento estivesse sendo serializado no formato XML. Cada nó elemento é seguido por seus nós espaço\_nome, nós atributo e nós filhos na ordem em que eles aparecem no documento. O conceito de *document order* é muito importante para a definição operadores das linguagens XPath e nas regras de semântica do XQuery, como será discutido mais adiante. (HOWARD KATZ, 2003)

O primeiro nó em qualquer documento é o nó *document* que contém o documento todo representando o próprio documento. Os nós elemento, comentários e instruções de processamento ocorrem no documento logo após a expansão das entidades. Os nós elementos ocorrem antes de seus filhos eles contêm nós elementos, nós texto, nós comentários e nós instruções de processamento.

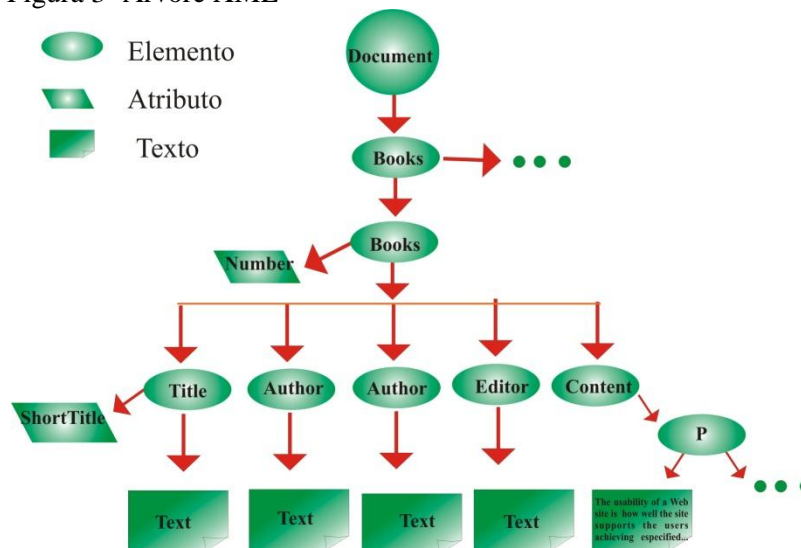
Os atributos são encontrados no documento depois do elemento e antes do filho do elemento, a ordem de atributos é estável, mas dependente da implementação. Veja o exemplo de arquivo XML na Figura 2 a ordem dos nós do documento respeita a estrutura apresentada no DTD, no Anexo II contém mais um exemplar de arquivo XML

Figura 2- documento XML

```
<Books>
<book number="1">
<title shortTitle="Improving Web Site Usability">Improving the Usability of a Web Site
Through Expert Reviews and Usability Testing</title>
<author>Millicent Marigold</author>
<author>Montana Marigold</author>
<editor>Véra Tudor-Medina</editor>
<content>
<p>The usability of a Web site is how well the site supports the users in achieving
specified goals. A Web site should facilitate learning, and enable efficient and
effective task completion, while propagating few errors.
</p>
<note>This book has been approved by the Web Site
Users Association.
</note>
</content>
</book>
...
</Books>
```

### 2.3 XQuery: Linguagem de Consulta para XML

Figura 3- Árvore XML



A Figura 3 ilustra a estruturação hierárquica de dados XML. As consultas XML seriam mais eficientes em uma nova linguagem de consulta ao invés de serem estendidas à linguagem relacional em virtude das várias diferenças entre os modelos de dados relacionais e XML mencionadas anteriormente.

Contudo a consulta aos dados XML precisa de uma linguagem de consulta adequada às suas características, ou seja, uma linguagem de consulta cuja sintaxe respeite a estrutura de um arquivo XML. Para atender este requisito, o W3C projetou o XQuery que é a linguagem de consulta sobre dados XML tal como SQL é a linguagem de consulta sobre dados relacionais. XQuery opera sobre o modelo de dados de um documento XML.

## 2.4 Modelo de Dados do XQuery

A entrada e a saída de XQuery são definidas em termos de um modelo de dados baseado na noção de uma sequência ordenada em conjunto de zero ou mais itens. Um item pode ser um nó ou um valor atômico. Um valor atômico é uma instância de um dos tipos de dados internos definido pelo XML *Schema*, tais como *strings*, inteiros, decimais e datas. Um nó está em conformidade com um dos sete tipos de elementos: nó, atributo, texto, documento, comentário, instrução de processamento, e nome-espço. Um nó pode ter outros nós como filhos, formando assim uma ou mais hierarquias de nós. Sequências podem ser heterogêneas, isto é, elas podem conter misturas de vários tipos de nodos e valores atômicos.(W3C, 14/12/2010)

Figura 4- sequência

```

let $sequence := ("1",
    "Improving Web Site Usability",
    "Improving the Usability of a Web Site Through
Expert Reviews and Usability Testing",
    "Millicent Marigold",
    "Montana Marigold",
    "New York",
    "Ersatz Publications",
    2001,
    2002
)

```

## 2.5 XPath

XPath é uma linguagem que opera sobre a estrutura de um documento XML pois ela é utilizada para localizar partes de um documento XML, ela fornece recursos básicos para manipulação de strings, números e booleanos. Desta forma XPath navega pela estrutura hierárquica de um arquivo recebendo o nome de um caminho como URLs (W3C, 1999).

Em XQuery as expressões XPath são usadas para localizar os nós de dados XML identificando sua localização na hierarquia da árvore por meio de expressões de caminho para selecionar nós ou conjunto de nós em um documento XML.

O nó é selecionado seguindo um caminho ou passos. No anexo II tem uma lista com algumas seleções em XPath. Para refinar ainda mais a seleção os predicados são utilizados e a expressão é escrita entre colchetes e retorna um valor booleano (AMER-YAHIA et al, 2006). Predicados são usados para encontrar um nó específico ou um nó que contém um valor específico, (W3CSCHOOLS, ca. 2000)

Figura 5- Consulta de predicado  
Fonte: AMER-YAHIA et al, 2006

```
/books/book[author = "Montana Marigold"]/title
```

## 2.6 Expressões XQuery

XQuery é uma linguagem funcional, isto é, que consiste em expressões que podem ser compostas por operadores que utilizam termos ou sintaxe com aplicação de função que são totalmente combináveis, e portanto, em qualquer local onde uma expressão é esperada, qualquer tipo de expressão pode ser

utilizada, além disto, cada expressão retorna um valor único e não tem efeitos colaterais.

Em expressões mais complexas em que são aceitos tanto o uso de funções predefinidas quanto funções definidas pelo usuário, uma chamada para uma função é uma forma básica de expressão podendo conter nelas literais, referência à variáveis, expressões de contexto de item, construtores e chamadas para funções.

As expressões ficam fechadas entre parênteses elas e são constituídas basicamente de expressões que consistem de símbolos terminais e separadores de símbolos tais como aspas simples ou duplas para os valores literais e o sinal de dólar para as variáveis (W3C, 14/12/2010).

Para combinar e reestruturar informações de documentos XML são utilizadas expressões FLWOR. (JONATHAN ROBIE, 2004)

A expressão FLWOR é um dos mais importantes tipos de expressões do XQUERY, ela é a expressão que controla a iteração e ordenação. A expressão de iteração, busca, ligação e ordenação FLWOR, cujo nome vem das palavras-chave for, let, where, order by, return é a chave para integrar a parte da funcionalidade.

As expressões FLWOR são semelhantes às declarações select, from, where da linguagem SQL, mas elas não são definidas em forma de tabelas, linhas e colunas, pois elas ligam variáveis a valores nas cláusulas let e for e usa as variáveis que foram ligadas para gerar novos resultados e esses resultados são chamados de tupla. (Howard Katz et al, 2004)

Figura 6- Consulta FLWOR

```
for $book in doc("full-text.xml") /books/book
let $title := $book/metadata/title
where $title contains text "improving usability"
  ordered distance at most 2 words at start
return $title
```

## 2.7 XQuery Full Text: Extensão do XQuery para Lidar com Texto

Nos dias atuais, as informações são disponibilizadas das mais variadas formas possíveis, uma delas é a prática de manter informações por meio de uma ferramenta Web, que permite acrescentar informações de forma colaborativa a fim de difundir o conhecimento dentro das empresas ou instituições públicas, tal ferramenta é denominada wiki.

Dentro das wikis, os artigos estão no formato de texto desta forma o leitor ao fazer uma pesquisa precisa de uma ferramenta que localize de forma rápida um título ou uma palavra-chave sem que ele tenha que ler todos os artigos. Este é um exemplo de aplicação que encoraja o uso da linguagem de consulta XQuery Full Text, pois ela pode ser usada para consultar perfeitamente tanto a estrutura como conteúdo texto de documentos XML.

XQuery Full Text foi desenvolvido como uma extensão à linguagem de consulta XQuery definida na Seção 2.1,3. XQuery Full Text permite aos usuários combinar consultas Full Text com consultas XQuery / XPath regulares.

A linguagem XQuery Full Text é uma linguagem funcional, pois ela utiliza funções como recurso de consulta. A expressão FTContainsExpr representa uma função que permite aos usuários pesquisar sobre nós XML combinando textos primitivos. A função FTContainsExpexpression permite

utilizar expressões FLWOR XQuery junto da função FTContainsExpr. O Anexo II apresenta um exemplo de consulta com expressões.

Figura 7- Consulta FTContainsExpr

Fonte: AMER–YAHIA et al, 2006

```
//book[(metadataftcontains"usabilitytests")and
(content/part/chapter/title ftcontains
"web-site usability")] /title
```

É possível tornar uma consulta mais específica ou mais detalhada por meio de uma busca aos outros nós, aprofundando na árvore por meio da mistura de XQuery com FTContainsExpr como na consulta da Figura 7 que retornará os títulos dos livros que contêm o termo "*usability tests*" e os títulos dos capítulos que tiverem a frase "*web-site usability*". FTSelection Expressions são usados para especificar a condição Full Text em uma expressão FTContainsExpr (AMER–YAHIA et al, 2006).

A tabela presente no Apêndice I apresenta as formas de seleção Full Text de acordo com a W3C (2011).

### 3 SGBD'S XML

Muitas aplicações de dados e conteúdos Web são armazenadas em um banco de dados relacional ou em arquivos de sistema, mas devido ao crescimento e troca do volume de dados estes métodos de armazenamento se tornaram ineficientes para acomodar conteúdo XML (ORACLE, 05/2008)



### 3.1 Oracle XML DB

*Oracle XML DB* é o nome dado para um conjunto de tecnologias relacionadas a alta performance de armazenamento e retorno de XML. Ele fornece suporte XML nativo pelo englobamento dos modelos SQL e XML de forma interoperável. Ele oferece suporte para a W3C XML e esquema de modelo de dados XML bem como, métodos de acesso padrão para navegar e consultar XML por meio de características próprias, tais como: um armazenamento independente, conteúdo independente e linguagem de programação independente e infraestrutura para armazenar e gerenciar dados XML. O repositório BD XML DB facilita esta manipulação pelo gerenciamento de hierarquia dos documentos XML.

A respeito de manipulação de dados, o Oracle XML DB suporta os principais padrões para atualização e transformação de dados XML, os padrões incluem a recomendação W3C XPath e o padrão ISO-ANSI SQL/XML. FTP, HTTP(S) e WebDav que podem ser usados para mover conteúdos XML para dentro ou para fora do Banco de Dados Oracle. Estas APIs fornecem acesso e manipulação de conteúdo XML contendo Java, C e PL/SQL (ORACLE, 05/2008).

#### 3.1.1 Oracle suporte ao XQuery Full Text

O Oracle Banco de dados XML foi o principal fornecedor a dar suporte a recomendação XQuery Full Text. A distribuição mais nova do XMLDB é a 12C e ela suporta os seguintes padrões W3C XQuery: Recomendação XQuery

1.0, Recomendação XQuery Facilidade de Atualização e a Recomendação XQuery e XPath Full Text 1.0 ( ORACLE, 26/06/2013).

A Tabela 1 apresenta as principais as compatibilidades do banco de dados Oracle XML com a W3C (ORACLE, 08/2008 e ORACLE, 05/2013).

Tabela 1- Suporte Oracle

W3C	Suporte Oracle	W3C	Suporte Oracle
Match Options:		Full-Text Operators Semantics:	
Language Option	não	FTOr	sim
Wildcard Option	sim	FTAnd	sim
Thesaurus Option	sim	FTUnaryNot	sim
Stemming Option	sim	FTMildNot	sim
Case Option	-	FTOrder	sim
Diacritics Option	-	FTScope	não
Stop Word Option	não	FTContent	não
Extension Option	-	FTWindow	sim
		FTDistance	sim
Logical Full-Text Operators:		FTTimes	-
Or-Selection	sim	FTWords	Sim
And-Selection	sim	FTWordsValue	sim
Mild-Not Selection	sim	FTAnyallOption	sim
Not-Selection	não	FTIgnoreOption	Não
		FTWeight	não
		FTScoreVar	não
		FLWOR	Sim

XQuery Full Text pode ser utilizado em consultas de dados do tipo XML independentemente do modelo de armazenamento utilizado. O SGBD Oracle suporta expressões Full Text *Contains*. Ela realiza uma pesquisa de contexto especificando os itens para a busca e uma seleção Full Text que filtra os itens selecionando e seus casamentos.

### 3.1.2 Carga e Armazenamento de Dados e Processamento de Consultas

Os índices são um recurso essencial para otimizar consultas aos documentos XML. O tipo dados apropriado para armazenar XML no ORACLE XML DB é o XMLTYPE de forma que as tabelas e visões XMLTYPE podem ser indexadas usando XMLIndex, árvore B, função baseada e/ou índices de texto. Para realizar busca sobre dados cujo esquema é conhecido deve previamente ser criado um índice de texto. A seguir, será descrito o fluxo que representa processo de indexação de texto no Oracle XML DB.

O objeto denominado Armazenamento faz a leitura de como os documentos são armazenados no sistema de acordo com seu armazenamento de dados de preferência, em seguida, o fluxo passa através do objeto Filtro,

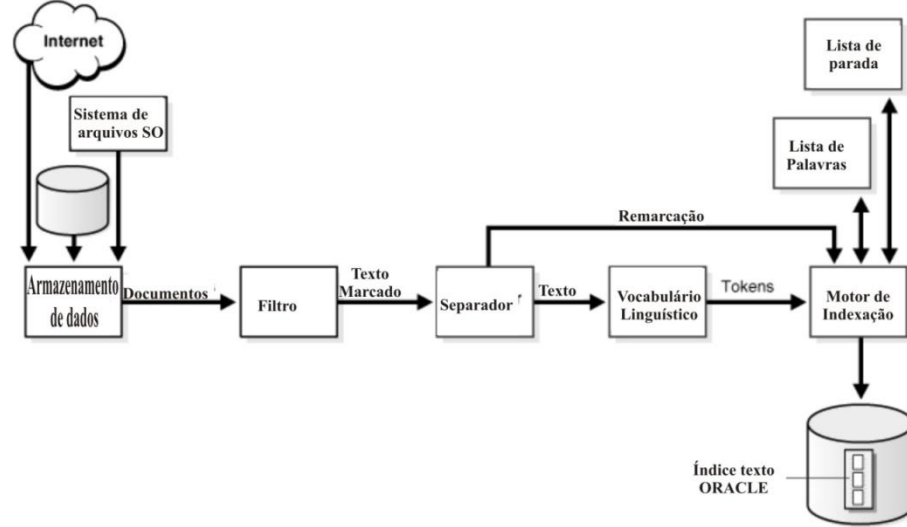
O Filtro pode ser especificado conforme a preferência, NULL\_FILTER, AUTO\_FILTER ou CHARSET\_FILTER, não deve aplicar filtros em documentos plain text, HTML ou XML. Depois de ser filtrado, o texto marcado passa através do objeto Separador.

O Separador separa o fluxo em textos e em seções de informações, tais como, onde o fluxo de texto começa e termina. A informação da seção é passada diretamente para o motor de indexação que será usado depois do texto ter passado pelo objeto Vocabulário Linguístico.

O vocabulário linguístico tem a função de especificar a linguagem do texto a ser indexado.

O motor de indexação cria um índice invertido que mapeia os tokens para os documentos que os contêm. Nesta fase o Oracle Text usa a Lista de parada para especificar exclusão de palavras de parada ou temas de parada do índice. Oracle Text também usa o parâmetro definidos na preferência denominada lista de Palavras que diz ao sistema como criar o prefixo do índice ou o índice substring caso disponível (ORACLE, 04/2013).

Figura 8- Processo de indexação



O propósito dos índices de texto é fornecer um acesso rápido para uma passagem particular do texto dentro dos nós texto do XML, ou seja, ele indexa cadeias de caracteres Full Text, deste modo o padrão de índices de base de dados que geralmente utilizados, tais como, árvore B ou bitmap não sejam adequados para acessar partes particulares de um documento XML.

Existem dois tipos de índices apropriados para trabalhar com documentos XML no ORACLE XML DB são eles o XMLINDEX e o CONTEXT Índex.

O índice Full Text é o *CONTEXT*, ele é criado sobre uma coluna do tipo XMLTYPE e este habilita as funções SQL *contains()* e facilita a função XQuery XPATH *ora:contains()* para realizar busca sobre XML. Ele é um índice invertido onde cada palavra contém uma lista de documentos que contêm a mesma palavra. Por exemplo, após uma simples operação inicial de indexamento a palavra DOG deve ter uma entrada desta forma: DOG DOC1 DOC2 DOC5. Este processo também é conhecido como indexação por *token*. Veja um exemplo de criação de índice Full Text no Apêndice III (ORACLE, 02/2014).

O índice XML é o XMLINDEX, ele fornece em geral um índice específico que indexa a estrutura interna do dado XML. Ele é um tipo de índice lógico projetado especificamente para dados XML. Ele pode ser usado juntamente com qualquer outro índice; pode ser utilizado com XML baseados em *schema* ou não baseado em *schema*; pode ser usado com os armazenamentos desestruturados, híbridos, e BINARY XML; e também as expressões XPath devem ser conhecidas a priori para serem utilizadas nas consultas.

O XMLINDEX desestruturado se aplica em todas as expressões XPATH possíveis para o XML, isto é possível por meio de uma tabela de caminhos composto por nós do documento e um conjunto de índices secundários na tabela de caminhos. Assim o uso de um componente desestruturado sozinho pode levar a problemas de ineficiência envolvendo vários exames e *self-joins* das partes das tabelas para consultas que projetam partes estruturadas. Veja o exemplo de criação de índice XML no Apêndice III (ORACLE, 02/2014.).

Para um melhor tratamento dos dados durante a realização das consultas aos documentos XML, a escolha da forma de armazenamento apropriada é indispensável a fim de se obter um retorno satisfatório de dados. O Banco de Dados Oracle possui suporte para armazenar arquivos XML nos modelos de dados Binary XML, Estruturados e Desestruturados. OS dados Binary XML estão armazenados internamente utilizando LOB's, *Large Objects*, porém atualmente utiliza o armazenamento SecureFiles LOB. O limite de tamanho máximo para os LOB's são de 8 para 128 TERABYTES, dependendo do seu tamanho de bloco do banco de dados (ORACLE, 06/2005).

O tipo de dado utilizado para Binary XML é o XMLTYPE. Com a introdução do XMLTYPE como tipo de dados foram fornecidas técnicas que facilitam a persistência do conteúdo XML dentro do banco de dados. Entre estas técnicas incluem a habilidade de armazenar documentos XML como uma coluna

ou tabela ou em um DB repositório XML, O XMLTYPE permite executar validações, operações e otimizações em conteúdos XML.

A natureza do dado XML e a forma como será utilizado definirá o formato do XMLTYPE pois dados altamente estruturados são denominados caso de uso *data-centric*, centrados nos dados e os dados altamente desestruturados são denominados *document-centric*, centrados no documento.

### 3.1.3 Limitações do Oracle XE

O Oracle XML Database Express é a versão gratuita do SGBD XML Oracle e por isto apresenta algumas limitações quando comparado com a versão paga da ferramenta. As principais limitações apresentadas neta versão:

Cada nó de texto ou valor de atributo processado está limitado ao tamanho de 64 K bytes (ORACLE, 05/2008 B)

Limite de tamanho do identificador XML que suporta apenas identificadores XML que estejam no máximo entre 32767 bytes ou 4000 bytes, dependendo do tamanho do valor de inicialização do MAX\_STRING\_SIZE.

Limite de tamanho do repositório de arquivo no qual o tamanho máximo de um arquivo dentro do repositório é por volta de 4 gigabytes.

Limite de recurso de arquivo de configuração do Repository-Wide, pois não pode criar mais de 125 recursos de arquivos de configuração para Repository-Wide.

Exclusão de pastas recursivas: Não pode deletar mais de 50 níveis de pastas aninhadas usando opções para deleção recursiva.

Não tem compressão de coluna híbrida isto porque a compressão coluna híbrida que está disponível somente para Oracle Exadata Storage Server Software não pode ser usada com uma coluna XMLType que está armazenada como objeto relacional ou com uma tabela XMLTYPE. O XMLTYPE suporta

apenas compressões básicas e compressão por OLDP. Pode ser um dos fatores para a para o caso de a baixa performance.

Não tem encriptação em nível de coluna para XMLTYPE, ele possui apenas encriptação nível tablespace para todos os modelos de armazenamento XMLTYPE.

Não tem acesso XMLType sobre links de banco de dados como não suporta acesso remoto à tabelas ou colunas XMLType (ORACLE, 05/2013 A).

Oracle Database XE pode ser instalado qualquer máquina com qualquer número de CPU's, armazena no máximo 11GB de dados de usuário, utiliza no máximo 1 GB de memória e utiliza apenas uma CPU na máquina hospedeira (ORACLE, 06/2014).

Esta versão é gratuita e está disponível nas plataformas Linux e Windows.(ORACLE, 09/2011).

## **3.2 SGBD XML Basex**

Basex é um banco de dados XML de alta performance, escalável e é um processador de XQuery 3.0 com suporte às extensões Full Text. Ele está focado em armazenamento, consulta e visualização de XML grandes, documentos JSON e coleções.

Ele possui uma interface visual permite aos usuários explorar dados interativamente e avaliar consultas em tempo real.

### **3.2.1 Basex suporte ao XQuery Full Text**

Basex fornece uma implementação da W3C das linguagens XPath 3.0 e XQuery 3.0 que estão intimamente ligadas ao armazenamento de banco de dados permitindo acessar fontes de dados locais e remotas.

O Basex foi o primeiro processador de consultas a oferece suporte completo para a recomendação W3C XQuery Full Text 1.0. O Basex trata as expressões Full Text por meio da comparação entre as strings de busca e entrada divididas em *tokens*. Muitas normalizações são entradas no processo de divisão em *tokens*, tais como, maiúsculas ou minúscula e acentos são removidos e uma linguagem opcional, linguagem de *stemming* será aplicada. Além disto os caracteres especiais, tais como espaços em branco e pontuações serão ignorados.

O Basex possui biblioteca com suporte às linguagens inglesa e alemã. Ele disponibiliza um índice Full Text para poder manipular as mais variadas combinações de opções de casamento definidas na recomendação Full Text, por padrão a maior parte das opções se encontram desabilitadas.

A Tabela 2s apresenta as funcionalidades W3C suportadas pelo Basex (BASEX, 2014 ):

Tabela 2- Suporte Basex

W3C	Suporte Basex	W3C	Suporte Basex
Match Options:		Full-Text Operators Semantics:	
Language Option	Sim	FTOr	Sim
Wildcard Option	sim	FTAnd	sim
Thesaurus Option	sim	FTUnaryNot	Sim
Stemming Option	Sim	FTMildNot	Sim
Case Option	Sim	FTOrder	Sim
Diacritics Option	Sim	FTScope	-
Stop Word Option	Sim	FTContent	-
Extension Option	não	FTWindow	Sim
		FTDistance	Sim
Logical Full-Text Operators:		FTTimes	-
Or-Selection	sim	FTWords	Sim
And-Selection	sim	FTWordsValue	Sim
Mild-Not Selection	-	FTAnyallOption	Sim
Not-Selection	sim	FTIgnoreOption	-
		FTWeight	-



		FTScoreVar	Sim
		FLWOR	Sim

### 3.2.2 Carga e Armazenamento de Dados e Processamento de Consultas

O BaseX oferece estratégias de execução diferentes para consultas XQFT, pois a escolha dependerá da entrada dos dados e da existência de um índice Full Text. O compilador de consultas tenta otimizar e acelerar a consultas por meio da aplicação de uma estrutura de índice Full Text sempre que for útil e possível. Um exemplo de criação de índice pode ser visto no Apêndice V

Existem três estratégias de avaliação disponíveis: a padrão, exame sequencial do banco de dados; uma avaliação baseada em índice Full Text e a híbrida que representa a combinação das duas estratégias anteriores (BASEX, 2014).

A estratégia exame sequencial requer que o documento seja escaneado sequencialmente, a expressão *localpath*, caminho local, começa a partir do nó raiz e atravessa todos os nós filhos. Cada elemento é passado no próximo passo para o filho, os elementos resultantes são filtrados pela expressão *FTcontains*.

A estratégia baseada em índice com inversão de caminho o otimizador de consultas irá reescrever e inverter a localização dos predicados e caminhos sempre que o acesso ao índice for possível. Esta abordagem é caracterizada como sendo de baixo para cima ao contrário da estratégia de exame sequencial. Primeiro o índice Full Text é acessado e depois faz o caminho de volta das folhas até a raiz passando pelos nós. A estratégia híbrida é a junção da avaliação sequencial com o uso de índice.

O BaseX possui funções úteis tais como as funções que permitem acessar um índice diretamente. Os resultados de Full Text podem ser marcados com

elementos adicionais ou partes relevantes podem ser extraídas dos mesmos. (GRUN. C, GATH. S, HOLUPIREK. A, SCHOLL. M H, ca. 2000).

Quanto ao armazenamento, um banco de dados BASEX pode conter documentos XML e também arquivos binários. Eles são manipulados de forma uniforme: um caminho de banco de dados único serve como chave e os conteúdos podem ser retornados via comandos do banco de dados, XQuery ou as várias API's existentes.

Os dados tipo binários são armazenados em seu formato original dentro de um subdiretório denominado *raw*. A boa performance do Basex se deve ao fato dos arquivos de sistema geralmente executarem muito bem quando eles vêm para a atualização e o retorno dos arquivos binários. O foco do Basex está no armazenamento eficiente da estrutura dos dados hierárquicos e formatos de arquivos tais como XML ou JSON (BASEX, 2012).

### **3.2.3 Basex limitações**

As limitações apresentadas pelo Basex são os limites de performance e chaves. Muitos arquivos de sistemas não são capazes de manipular milhares ou milhões de recursos binários em um diretório único de forma eficiente, isto ocorre quando existe um grande número de documentos XML que precisam ser importados para o banco de dados Basex ou exportado dele. A solução para evitar o gargalo é distribuir os binários relevantes em subdiretórios adicionais. O Basex não suporta o uso de chaves no sistema de arquivo corrente e a solução para a questão das chaves é adicionar um documento XML dentro do banco de dados que contenha o mapeamento de todos pares palavra\_chave/caminho. (BASEX, 2012).

### 3.3 MXQUERY

MXQUERY é uma ferramenta de busca XQuery de código aberto para trabalhar com dados XML . MXQuery possui alguns diferenciais por suportar grande variedade de plataformas por ser implementado em Java. Ela fornece uma ampla cobertura os padrões mais atuais da W3C como XQuery 3.0, *scripting*. Ela possui um compilador cruzado da plataforma Java, isto permite ao compilador produzir um código executável para uma plataforma diferente da de criação, ideal para uso pelo navegador de internet. Ela também possui processamento de *streaming* de dados (MXQuery, ca. 2000 ).

MXQuery foi desenvolvido para ser uma máquina XQuery para rodar em dispositivos móveis. Para isso, é existe uma integração com modelo de aplicação Android usando a classe import do java para isto (Sync/Async/Service) e também a habilidade de para chamar funcionalidade de plataforma Android usando a classe Java Import. Para manter o uso de memória baixo a máquina MXQuery inicialmente suportou apenas um subconjunto de XQuery, não suportava tipos definidos pelo usuário e nem a tipificação completa do sistema XQuery restringindo-se apenas a um subconjunto de tipos atômicos como `xs:untypedAtomic`, `xs:string`. Isto reduz o tamanho de armazenamento e memória da máquina consideravelmente (MXQUERY, 2008).

### 3.3.1 MXQUERY suporte ao XQuery Full Text

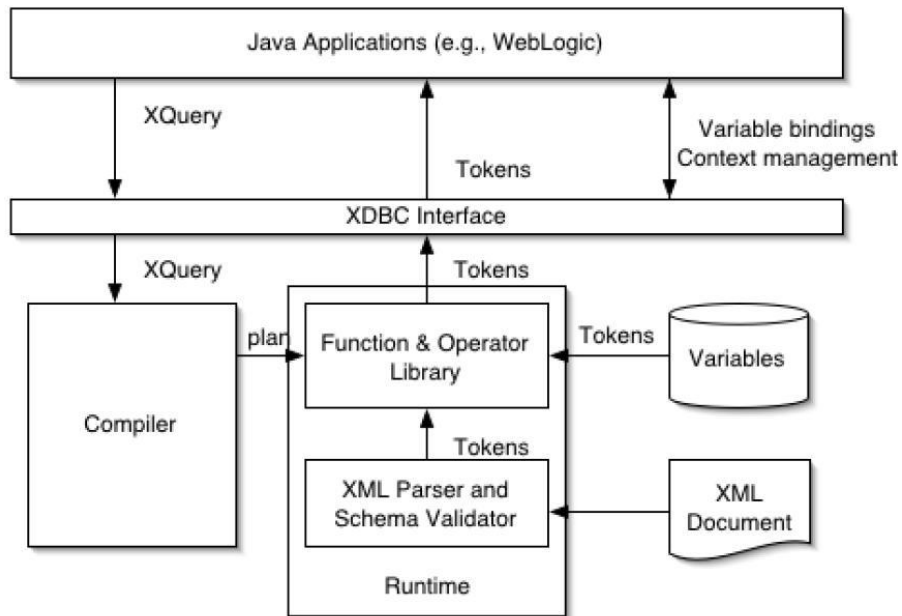
MXQuery está parcialmente em conformidade com XPath/XQuery FullText 1.0, conforme Tabela 3. (MXQUERY, 2009).

Tabela 3- suporte MXQUERY

W3C	Suporte MXQUERY	W3C	Suporte MXQUERY
Match Options:		Full-Text Operators Semantics:	
Language Option	Não	FTOr	Sim
Wildcard Option	Sim	FTAnd	Sim
Thesaurus Option	Não	FTUnaryNot	Sim
Stemming Option	Sim	FTMildNot	Sim
Case Option	Sim	FTOrder	Sim
Diacritics Option	Sim	FTScope	-
Stop Word Option	Sim	FTContent	-
Extension Option	-	FTWindow	Sim
		FTDistance	Não
Logical Full-Text Operators:		FTTimes	Não
Or-Selection	Sim	FTWords	Sim
And-Selection	Sim	FTWordsValue	Sim
Mild-Not Selection	Não	FTAnyallOption	Sim
Not-Selection	Não	FTIgnoreOption	Não
		FTWeight	Não
		FTScoreVar	Não
		FLWOR	Sim

### 3.3.2 Carga e Armazenamento de Dados e Processamento de Consultas

Figura 9- Processamento de consulta



Sobre máquina XQuery streaming a Figura 9 pode dar uma visão geral de seu funcionamento. Uma aplicação java submete consultas XQuery e os resultados da consulta consumida através de uma interface referida como XDBC.

O compilador de consulta faz o *parse* (análise da palavra) e a otimização da consulta. O compilador gera um plano de consulta, que é uma árvore de operadores que consomem dados de um outro em cascata de moda. O plano de consultas é interpretado pelo sistema de tempo de execução que consiste da implementação de todas as funções e operadores da biblioteca XQuery e do núcleo XQuery.

Além disso o sistema de tempo de execução contém um analisador XML e um validador de esquema que são requeridos quando um dado XML externo deve ser processado como parte da consulta. As mensagens de entrada XML são analisadas e tem seu esquema validado uma única vez e então é armazenado em um formato especial para que então ele possa ser utilizado em muitas consultas XQuery sem que se tenha que realizar todos esses passos para cada consulta invocada.

As mensagens são limitadas como variáveis livres para consultas e variáveis de ligação são também para executadas por meio da interface XDBC. Todos os dados XML são interpretados como uma corrente de tokens. Esta corrente de token minimiza as exigências de memória da máquina.

Em adição, a corrente de token permite a avaliação postergada das consultas (referenciada como *lazy* pelo autores). Em tempo de execução cada operador runtime recebe como entrada um tokens por vez e os dados que não são requeridos são descartados. A corrente de tokens corresponde (casa) com o modelo de dados ( FLORESCU et al., Ca 2000).

O MXQuery é um modelo de iterador pull-based, um ponto forte deste tipo de protocolo de streaming é a simplicidade que faz dele uma opção para muitos sistemas de *streaming* do mundo real. A característica principal é que o protocolo inclui (*buffer map exchanges*) mapa de trocas armazenamento periódico e falta de programação centralizada. Esta característica contribui muito para a simplicidade do protocolo pull-based mesh (engrenagem baseada em puxar), mas ao mesmo tempo conduz para uma brecha entre os limites fundamentais e a performance atual ( CHEN FENG, BAOCHUN E LI, BO LI, Ca. 2000).

A representação de dados usados internamente é uma sequência de tokens chamado de corrente de token (token stream) que é um modelo de corrente de tokens tipificados em XML e acessado via uma API pull-based para

produzir e consumir tokens vagorosamente. Além do mais os tokens para começo de documento, elementos, ou atributos são aumentados para carregar os ids dos nós a fim de comparar nós por ambos igualdade e ordenação de documentos (DIAO, Ca 2000).

Esta abordagem apresenta algumas vantagens tais como: baixo requerimento de memória principal, evita a materialização de modularidade resultados intermediários, suporta avaliação postergada, Lazy que é muito importante para streamings infinitas que já foram testadas pela máquina (TSOULO, 2008).

Quanto ao armazenamento de dados o MXQUERY a fonte de dados pode ser um esquema XM ou um arquivo (VAKALI, A. ,PALLIS, G, 2007).

### **3.3.3 MXQUERY limitações**

O MXQUERY não dá suporte completo ao XQuery, as expressões de ordenação, agrupamento e extensões variáveis de itens e de contexto externo não se aplicam no momento. Com relação ao Full Text ele também não fornece suporte de forma completa porque as funções *scoring* e *Weights* não estão implementadas. Não atendem satisfatoriamente: Seleção de armazenamento automático Fine-grained, Mistura atualizável de armazenamento de streaming e Full Text dentro do mesmo módulo de consulta, Melhorias da performance, CPU e memória (MXQUERY, Ca 2000).

### **3.4 SGBD IBM DB2**

O DB2 Express-C é a versão gratuita de um dos sistemas de gerenciamento de banco de dados da IBM. O DB2 Express-C 9.7 está disponível

para Linux, Unix, Windows e até mesmo Mac OS X. Para lidar com XML utiliza uma tecnologia denominada pureXML. (DB2 Express-C, Ca 2000).

### **3.4.1 IBM DB2 Suporte ao XQuery Full Text**

A funcionalidade do IBM DB2 para realizar consultas XQuery Full Text é conhecida como DB2 Text Search, a qual habilita um usuário do banco de dados IBMDB2 para Linux, UNIX e Windows criarem aplicações com capacidade de busca Full Text por meio da adição de cláusulas Full Text nas declarações SQL e XQuery. DB2 Text Search possibilita fazer busca em dados estruturados e desestruturados armazenados no banco de dados, tais como busca Full Text em textos, HTML e documentos XML. SQL completamente integrado, SQL/XML, suporte XQuery, incluindo sintaxe XPath e subconjuntos para busca em documentos XML (IBM, Ca 2000).

A fim de alcançar boa performance e escalabilidade DB2 Text Search faz uso de streaming para evitar o alto consumo de recursos durante as consultas.



Tabela 4- suporte DB2

W3C	Suporte DB2		
Match Options:		Full-Text Operators Semantics:	
Language Option	Sim	FTOr	Sim
Wildcard Option	Sim	FTAnd	sim
Thesaurus Option	Sim	FTUnaryNot	Sim
Stemming Option	Sim	FTMildNot	Não
Case Option	-	FTOrder	Sim
Diacritics Option	Sim	FTScope	-
Stop Word Option	Sim	FTContent	-
Extension Option	-	FTWindow	Sim
		FTDistance	Sim
Logical Full-Text Operators:		FTTimes	-
Or-Selection	Sim	FTWords	Sim
And-Selection	Sim	FTWordsValue	-
Mild-Not Selection	Não	FTAnyallOption	-
Not-Selection	Sim	FTIgnoreOption	-
		FTWeight	-
		FTScoreVar	Sim
		FLWOR	Sim

As funções disponíveis para busca Full Text são as consultas com palavras ou frases usando *CONTAINS()* ou *xmlcolumn-contains()* (IBM, Ca 2000).

### 3.4.2 Carga e Armazenamento de Dados e Processamento de Consultas

No DB2 há disponível o suporte à indexação de dados armazenados em colunas XML, os índices XML indexam parte de uma coluna onde indica que partes de uma coluna será indexada especificando um padrão limitado por uma expressão XPath (IBM, 2013).

O índice de texto pode ser criado sobre uma variedade de tipos de dados de VARCHAR pequenos até BLOB's grandes que contenham objetos texto. Em um documento XML o índice pode ser criado para elementos, atributos ou valores, nós textos (DB2 Express-C, Ca 2000).

Um índice text search consiste de termos significantes que são extraídos dos textos de documentos. A chave primária da linha da tabela é usada no index para identificar a fonte dos termos (IBM, Ca 2000).

O DB2 armazena os dados XML nativamente, ou seja, ele utiliza o modelo de dados hierárquicos do XML para armazenar e processar XML internamente. Não existe mapeamento para o modelo relacional, os documentos XML não são convertidos em strings (CLOB ou VARCHAR)..

Dados relacionais e dados hierárquicos podem ambos ser armazenados no DB2 banco de dados híbrido. Os dados relacionais são acessados utilizando SQL ou XQuery, os dados XML podem ser acessados usando SQL/XML ou XQUERY (IBM, Ca 2000).

### **3.4.3 IBM DB2 limitações**

O DB2-Express-C oferece partilhamento do mesmo código fonte das edições comerciais, limites de 2 Cores (1 CPU) , 2 GB de RAM, Sem limites de tamanho da base de dados, Sem limites de conexões e Sem limites de usuários ou quaisquer outros limites (DB2 Express-C, Ca 2000).

O servidor DB2 Text Search não é instalado por padrão juntamente com o servidor de banco de dados DB2, pois ele requer a execução de uma configuração stand alone para que o Text serch seja habilitado.

## 4 METODOLOGIA

Nesta seção serão explicados os critérios utilizados para a criação do ambiente de testes nos quais os dados, os SGBD's, os procedimentos de execução das tarefas e as informações obtidas após este processo serviram de subsídio para realizar a avaliação de facilidade de uso, funcionalidade e desempenho dos SGBD's.

Dentro da perspectiva da facilidade de uso dos SGBD's buscaremos avaliar as dificuldades sob o ponto de vista de uma equipe técnica de uma empresa sem o auxílio direto do suporte técnico especializado *in loco*. Neste contexto, este suporte através de canais virtuais como fóruns e listas de e-mail também será um critério de avaliação.

No que diz respeito à funcionalidade buscaremos avaliar o suporte ao Full Text entre os SGBD's com o intuito de descobrir o quão adequados eles estão com a recomendação de XQuery Full Text de acordo com a W3C.

A avaliação de desempenho consiste no tempo de resposta obtidos dos SGBD's no contexto de uma carga maior de dados, a fim de explorar o potencial máximo das consultas sobre um volume de dados alto.

### 4.1 Classificação

Nesta seção apresentaremos a classificação da pesquisa.

A presente pesquisa se classifica quanto à natureza como básica, quanto aos objetivos de forma descritiva, quanto às abordagens como pesquisa qualitativa e quantitativa. Para a colimação dos objetivos deste trabalho, os seguintes passos foram realizados:

- para a revisão bibliográfica buscamos informações atualizadas do momento em que os testes foram realizados.

- para estudar o suporte às funcionalidades do XQuery Full Text em SGBD's XML comerciais e de código aberto foram consultadas as documentações dos SGBDXs escolhidos;
- para construir de um ambiente de testes para comparação entre SGBD's XML que caracterizassem cenários reais de aplicações baseadas em dados semiestruturados trabalharemos na configuração do ambiente, na coleta de dados reais e na definição do conjunto de consultas a serem usadas nos testes;
- para avaliar e comparar o desempenho e as funcionalidades dos dois SGBD's XML que suportam XQuery Full Text utilizamos uma tabela com valores ponderados de avaliação e comparação.

As bases de dados que foram utilizadas a serem utilizadas como métrica para avaliação do XQuery Full Text são:

O caso de uso da W3C, XQuery and XPath Full Text 1.0, forneceu modelos de consultas em documento XML e o INEX, Iniciativa para avaliação de recuperação de informação XML, forneceu a coleção de dados da wikipédia.

. Teremos por objeto de avaliação dos SGBD's XML: BASEX e MXQUERY, de código aberto (BASEX, 2012. MXQUERY, 2009 ). e o SGBD's proprietários Oracle XML Database e IBM DB2(ORACLE, 2013. IBM,2013).

A seguir, apresentaremos mais detalhes a respeito das avaliações de funcionalidade e desempenho.

## **4.2 Hardware**

Nesta seção apresentaremos os equipamentos técnicos que serviram para o desenvolvimento deste trabalho.

Para construir de um ambiente de testes que tornasse possível realizar a comparação entre SGBD's XML que caracterizassem cenários reais de aplicações baseadas em dados semiestruturados foi utilizado um servidor plataforma Ubuntu Linux. A configuração da máquina era de 8GB DE RAM e 1 TB de HD.

### **4.3 Instalação**

BaseX: foi instalado com sucesso sem nenhuma complexidade apresentada durante o processo de instalação.

IBM DB2: foi instalado parcialmente porque a ferramenta DB2 Text Search a qual habilita busca full text não estava inclusa na instalação padrão, por isto foi necessário executar a instalação da mesma de forma stand alone, o tutorial disponível é muito complexo e nem um pouco didático.

MXQUERY: foi instalado com sucesso e o processo de instalação não apresentou nenhuma complexidade.

ORACLE XML DB: foi instalado com sucesso sem apresentar problemas.

### **4.4 Importação**

BaseX: para realizar a importação de dados para o BaseX foi realizada a referência do diretório no sistema de arquivos no SGBD no momento da criação do banco a validação dos dados pode ser habilitada nos parâmetros do SGBD.

IBM DB2: A importação é por meio de comandos e o arquivo XML foi importado para o banco sem apresentar restrições ou erros. O SGBD exigiu

tratamentos dos dados e por isso foi necessário alterar alguns caracteres como aspas simples por dupla e vice-versa para depois executar a importação.

**MXQUERY:** A importação dos dados também foi simples, este processo é composto pela referência do local do diretório do arquivo XML na máquina que é passado como parâmetro dentro da consulta, o que causou um pouco de transtorno foi no momento da importação da coleção de arquivos XML, pois devido ao MXQUERY não fornecer na sua documentação os comandos para criar collections iria inviabilizar realizar a consulta sobre os 26.416 arquivos. A fim de contornar esta deficiência do MXQUERY e poder prosseguir com o trabalho foi criado um arquivo XML único composto por todos os arquivos da base de dados.

#### **4.5 Teste de Funcionalidade**

Nesta seção será apresentada a descrição dos dados e das consultas utilizadas durante os teste de funcionalidades.

A base de dados foi composta por uma amostra de dados de uma coleção de apenas três livros que foram coletados do caso de uso XQuery e XPath Full Text 1.0 que estão disponíveis em W3C (2011). O caso de uso XQuery e XPath Full Text 1.0 é composto por um modelo de dados XML, uma lista de consultas mais o retorno delas. O documento XML pode ser visualizado no Anexo I e as consultas escolhidas para serem utilizadas nos teste se encontram mais adiante nesta mesma seção.

Para preparar para ambiente de teste de funcionalidades foram utilizados Casos de Uso XML Full Text para que que tivéssemos cenários de usos de consultas com possíveis soluções em XQuery e XPath,. dentre estas consultas foram selecionadas dez consultas a fim de compor um conjunto de consultas Full Text. Os casos de uso forneceram além do um arquivo XML das consultas,

exemplos de entradas de dados, tais como, funções, características dos textos suportados em XQuery e XPath, ilustrando consultas simples e complexas mais suas respectivas saídas de dados.

A escolha das consultas que foram utilizadas na construção do ambiente testes foram escolhidas mediante o seguinte critério: intenção de aplicar consultas coerentes com realidade de uma organização que gostaria de extrair dados de documentos XML, desta forma classificamos as consultas dentro das categorias simples, medianas e complexas conforme exibido na Figura 15 .

As consultas 1,2 e 3 foram consideradas simples porque representam buscas por uma palavra ou uma simples frase.

As consultas 4,5, e 6 foram consideradas medianas porque são compostas por uma consulta mais avançada por conter declarações de expressões FLWOR e FTSelections.

AS consultas 7,8,9 e 10 foram consideradas complexas porque além de conterem declarações presentes nas consultas simples e medianas elas possuem variáveis, wildcards, e consulta em atributo.

Figura 10 - Consultas de Funcionalidade

Consultas Simples	
Q1	2.2.1 Q1 - Encontrar todos os títulos de livro que contém a palavra "usability". Esta consulta encontra uma palavra em um elemento.
doc("full-text.xml") /books/book/metadata/title[. contains text "usability"]	
Q2	2.2.2 Q2 - Encontrar todos os assuntos dos livros que contenham a frase "usability testing". Esta consulta encontra uma frase em um elemento.
doc("text.xml") /books/book/metadata/subjects/subject[. contains text "usability testing"]	
Q3	3.2.6 Q6 - Encontrar alguma palavra "mouse" em todos os livros. Esta consulta busca em todo o documento.
doc("full-text.xml")[. contains text "mouse"]/books/book	
Consultas medianas	

Q4	2.2.6 Q6 - Encontrar todos os títulos de livro que começam com "improving" seguida de 2 palavras pela "usability". Esta consulta encontra um elemento que começa com uma palavra específica.
<pre>for \$book in doc("full-text.xml") /books/book   let \$title := \$book/metadata/title   where \$title contains text "improving" ftext "usability"     ordered distance at most 2 words at start return \$title</pre>	
Q5	10.2.5 Q5 Busca And Not. Encontrar todos os livros com a palavra "usability" e não tenham a palavra "plan" no metadata.
<pre>for \$book in doc("ull-text.xml") /books/book   let \$sup := \$book/metadata   where \$sup contains text "usability" ftext fnot "plan" return \$book</pre>	
Q6	16.2.1 Q1 - Full-Text Query construção de um novo elemento. Para os livros que contêm "usability" no título será criada uma lista rasa de todos pares título-autor. Cada par estará agrupado no elemento novo construído.
<pre>for \$book in doc("full-text.xml") /books/book   let \$var := \$book/metadata/title   where \$var contains text "usability" return &lt;result&gt;{\$book/metadata/title, \$book/metadata/author} &lt;/result&gt;</pre>	
<b>Consultas complexas</b>	
Q7	4.2.1 Q1 - Encontrar todos os livros com as palavras "improve" "web" "usability" no short title. Esta consulta busca por palavras múltiplas em um atributo permitindo palavras variantes e permitindo a palavra em qualquer ordem com um número máximo especificado de palavras no meio.
<pre>for \$book in doc("full-text.xml") /books/book   where \$book/metadata/title/@shortTitle contains text "improve"     using stemming ftext "web" ftext "usability" distance at most 2 words return \$book/metadata/title</pre>	
Q8	6.2.1 Q1 - Encontrar todos os títulos de livro que contêm a palavra "test" no texto. Esta consulta encontra uma palavra e suas variantes (derivadas)
<pre>for \$book in doc("full-text.xml") /books/book   let \$scont := \$book/content   where \$scont contains text "test" using stemming</pre>	



return \$book	
Q9	10.2.4 Q4 - Encontrar em todos os livros a coleção que não contenha "usability testing". Esta consulta busca por livros que não contenham uma frase em um elemento e em seus descendentes.
for \$book in doc("full-text.xml") /books/book where \$book contains text fnot "us.* testing" using wildcards return \$book	
Q10	12.2.2 Q2 - Encontrar todos os livros com as palavras "efficient task completion". Esta consulta busca por palavras múltiplas em um atributo permitindo palavras variantes e permitindo a palavra em qualquer ordem com um número máximo especificado de palavras no meio ordenados.
for \$book in doc("full-text.xml") /books/book let \$cont := \$book/content where \$cont contains text "efficient" fband "task" fband "completion" ordered distance at most 10 words return \$book	

#### 4.6 Teste de Desempenho

O benchmark é executado em vários sistemas diferentes ranqueando os resultados dos custos e performance de cada sistema submetido, baseando se pela carga de trabalho medida usualmente em trabalho/segundo ou transação/segundo. (GRAY, 1993).

No presente trabalho adotamos o modelo de Gray (1993), aplicando o benchmark nos SGBD's Basex, MXQUERY, IBM DB2 e ORACLE XML DB. O conjunto de dados consiste de dados compostos por um conjunto de arquivos XML, para isto o tamanho representado pela carga de dados foi uma das preocupações. Os dados para compor o conjunto de dados a serem importados foram obtidos da base de dados de arquivos XML criada pelo INEX. O INEX tem por objetivo fornecer grandes coleções de documentos com informações

relevantes que possibilitem realizar a avaliação sobre ferramentas de retorno de informação.

Dentre as coleções disponíveis, a coleção denominada Wikipedia 2009 consiste de 14 GB e um total de até 32311 tags distribuídas entre os arquivos. Contudo para cada 1GB de dados seriam necessários 4 GB para processá-los. Esta foi a base de dados selecionada para compor dos experimentos, juntamente com a coleção de dados foi adquirido o arquivo 2009 Topics 2009001-2009115 (v1.0) e o qrels.txt .

O arquivo Topics contém informações relevantes sobre a coleção de dados, tais como os assuntos relacionados aos artigos que se encontram no conjunto de dados e uma estrutura que mostra onde aquele assunto pode ser encontrado dentro da hierarquia dos nós dos XML apresentada como XPath. Para certificar de que a palavra-chave ou o a frase citada em TOPICS está realmente no diretório dos arquivos, para isto pode ser utilizado o documento qrels.txt . Cada tópico contido no arquivo TOPICS tem um número de id que está também no arquivo qrels.txt seguido de alguns números, nesses números, os três últimos algarismos do id representa pastas onde o documentos que contém as palavras-chaves poderão ser encontrados. As demais colunas representam o número do documento XML e a posição da palavra. (INEX, Ca. 2000).

O teste de desempenho foi realizado sobre uma parte da coleção Wikipédia 2009 composta por 26.416 arquivos que representam 512 MB em dados armazenados. Estes arquivos foram modificados para atender os requisitos de importação exigidos pelos SGBD's tais como, alterações nos caracteres HTML para o hexadecimal dos mesmos e a primeira linha de comentário foi excluída.

A criação das consultas para o teste de desempenho tiveram como base as consultas do teste de funcionalidade de forma que foram adaptadas para o

conjunto de dados da Wikipédia 2009, O critério realizado foi o seguinte: as dez consultas do teste de funcionalidade foram usadas de modelo de forma que as funções XQuery Full Text foram preservadas, o nome da coleção foi alterado para o nome da coleção de dados dos respectivos SGBD's, os caminho dos nós foram coletados dos TOPICS, e as palavras-chave foram escolhidas consultando palavras aleatórias de dentro dos arquivos. Veja o apêndice III e IV.

#### **4.7 Métricas utilizadas**

Nesta seção serão apresentadas e discutidas as métricas utilizadas durante os testes realizados nos SGBD's.

Os itens instalação, importação e criação de índices o critério de avaliação é baseado desta perspectiva de como ocorreu o andamento das atividades, Assim foi colocado da seguinte forma:

Para a atividade de instalação do SGBD que foi concluída com sucesso atribui-se o valor igual a 2, se a instalação teve problemas e terminou incompleta atribui-se o valor igual a 1 mas se finalmente a instalação foi concluiu atribui-se o valor igual a 0.

Para a importação de dados para o SGBD que a própria ferramenta de consultas realiza a importação com sucesso de forma simples atribui-se o valor 2, se a importação requer a instalação e uso de outras ferramentas e envolve o estudo de mais técnicas para realizar a tarefa de forma que realiza a importação com sucesso mas com complexidade no processo atribuiu o valor 1 e finalmente se não foi possível concluir a importação atribui-se o valor igual a 0.

Para a atividade de criação dos índices o SGBD's que permite realizar a tarefa de forma simples ou não precisa criar índice manualmente atribui-se o valor 2, se os comandos para criação de índice são simples atribuiu-se o valor igual a 1 e finalmente se para criar o índice apresentou muitas variedades de

sintaxes com detalhamentos complexos para cada tipo de aplicabilidade atribuiu-se o valor 0.

Para a funcionalidade definimos que no SGBD em que as consultas não precisaram ser adaptadas e executaram com sucesso atribuiu-se o valor 2, se fosse necessário fazer muitas adaptações nas consultas que não executavam porque expressão Full Text não é suportada ou porque a sintaxe é diferente atribuiu-se o valor 1 e se a consulta não pode ser executada atribuiu-se o valor igual a 0.

Para analisar o desempenho dos SGBD's foi olhado o seguintes requisitos: se as consultas executaram com sucesso, se elas retornaram com erro e o tempo de execução. No caso onde as consultas retornaram com sucesso com menor tempo comparado entre eles, atribuiu-se o valor 2, nas consultas que retornaram com um tempo mais alto comparando uns com os outros, atribuiu-se o valor 1 e finalmente para as consultas que não retornaram nada ou erro de estouro de memória atribuiu-se o valor 0.

Ainda sobre a análise de desempenho foi seguido o seguinte critério para a tomada dos tempos de retorno das consultas: Cada uma das dez consultas foram executadas cinco vezes, mas para fazer o cálculo da média dos tempos delas os tempos obtidos das consultas das extremidades foram removidos, de modo que a primeira e última a consulta não entraram no cálculo da média do tempo. A unidade de tempo resultante foi convertida em milissegundos.

Para analisar o atendimento ao usuário fornecidos pelos SGBD's foi visto como o suporte é oferecido, se é oferecido suporte e ele é ativo e eficiente atribuiu-se o valor igual a 2, se oferece suporte mas o mesmo não é eficiente atribuiu-se o valor 1 e finalmente se não oferece suporte atribuiu-se o valor 0.

## 5 RESULTADOS

A Tabela 5 apresenta o de significado qualitativo dos índices e os valores dados para cada um dos requisitos de avaliação para cada um dos SGBD's avaliados.

Tabela 5- Índices com Valores Qualitativos

Índices	Enquadramento		
	0	1	2
Atividade			
Instalação	Não instalou	Instalou sem a funcionalidade full text	Instalou
Importação	Não importou	Importação complexa	Importação simples
Criação de índice	complexa	simples	Automatica/desnecessária
Funcionalidade	Não atendeu	parcialmente	totalmente
Desempenho	Estouro de memória/sem retorno/Não executou	Alto tempo de resposta	Resposta em curto espaço de tempo
Suporte	Não possui	Possui mas não é eficiente	Suporte ativo e eficiente

ORACLE XML DB: A importação dos dados foi complexa e problemática devido a necessidade de utilizar ferramentas de carregamento de dados, SQL LOADER, para importar os dados.

O *script* para realizar a importação requer comandos complexos e durante a execução da importação ocorrem muitas validações de arquivo XML que faz com que o tempo de carga seja muito alto, no final desse processo o arquivo de log retorna apenas até tipos 50 erros, e passar desta quantidade não é possível saber o que causou falha na importação. Difícil é se tiver que abrir cada um dos os 26.416 arquivos para analisar e tratá-los manualmente esta seria uma tarefa inviável.

Para prosseguir com as atividades foi buscado no arquivo de log os arquivos classificados como *badfiles* estes arquivos foram removidos pois foram considerados como dados ilegais para o carregamento. Nos casos em que estavam sendo acusados problemas com caracteres inválidos exigiu tratamento dos dados e então em cada um dos arquivos nesse caso foi tratado utilizando um algoritmo na linguagem de script Shell foi criado e executado um alterar os caracteres HTML pelo respectivo caractere em hexadecimal.

Para cada restrição encontrada durante o processo de importação dos dados era necessário recomençar o processo de importação do início e fazer inúmeras checagens no arquivo de log para então depois finalizar a importação.

Uma informação importante para reduzir o tempo de carregamento de dados é que é necessário desabilitar os índices da tabela antes de iniciar a importação e habilitá-los depois que os dados já estiverem consistentes no banco. Devido a tantas validações de restrições o processo de importação se tornou moroso.

## 5.1 Criação de Índices

Basex: A criação e ativação do índice Full Text foi simples pois só há uma sintaxe para criação de índice

IBM DB2: A criação de índice foi realizada com sucesso.

MXQUERY: Não foi necessário criar índices.

ORACLE XML DB: A criação de índices requer um estudo entre os vários tipos de índices disponíveis, para então escolher o índice apropriado para os dados desestruturados com base na finalidade e forma de consulta para obter eficiência com eles.

## 5.2 Funcionalidade

BaseX: Na realização do teste de funcionalidades não foram necessárias adaptações relacionadas a sintaxe consultas. Todas as funções rodaram sem apresentar restrições ou erros retornando dados conforme o esperado.

IBM DB2: na sintaxe das consultas que foram analisadas foram identificadas algumas peculiaridades da implementação do XQuery Full Text na Ferramenta as quais houve necessidade de adaptar as consultas da W3C para execução. Infelizmente o SGBD IBM EXPRESS – C não passou pelas avaliações de funcionalidade porque depois da instalação não foi possível iniciar o serviço do DB2 Text Search.

MXQUERY: O teste de funcionalidades foi realizado com sucesso após ajustar a consulta às exigências de sintaxe e exclusão do parâmetro de ordenação que não está disponível no MXQUERY. Foram encontradas algumas peculiaridades na sintaxe então foi necessário adaptar as consultas para realizar os testes. Não é preciso criar índices no XMQUERY manualmente.

ORACLE XML DB: O Oracle também apresentou peculiaridades com relação à sintaxe de consulta XQuery comparada com a W3C, por isto foi necessário despende de tempo para transformar as consultas do caso de uso em consultas apropriadas para executar no Oracle. O teste foi realizado com sucesso no retorno das consultas.

Tabela 1 Comparação das funcionalidades

<b>W3C</b>	<b>Oracle</b>	<b>BaseX</b>	<b>MXQUERY</b>	<b>IBMDB2</b>
<b>Opções de casamento</b>				
Language Option	<b>Não</b>	Sim	<b>Não</b>	Sim
Wildcard Option	Sim	Sim	Sim	Sim
Stemming Option	Sim	Sim	Sim	Sim
Stop Word Option	<b>Não</b>	Sim	Sim	Sim
<b>Operadores lógicos</b>				
Or-Selection	Sim	Sim	Sim	Sim
And-Selection	Sim	Sim	Sim	Sim
Not-Selection	<b>Não</b>	Sim	Não	Sim
<b>Operadores semânticos</b>				
FTOr	Sim	Sim	Sim	Sim
FTAnd	Sim	Sim	Sim	Sim
FTUnaryNot	Sim	Sim	Sim	Sim
FTMildNot	Sim	Sim	Sim	<b>Não</b>
FTOrder	Sim	Sim	Sim	Sim
FTDistance	Sim	Sim	<b>Não</b>	Sim
FTScoreVar	<b>Não</b>	Sim	<b>Não</b>	Sim
FLWOR	Sim	Sim	Sim	Sim



### 5.3 Desempenho

BaseX: teste de performance retornou valores com sucesso numa escala de milissegundos em média, mas foi necessário configurar o tamanho da pilha de memória do java para executar as consultas,

IBM DB2: Não foi possível avaliar o desempenho.

MXQUERY: Desempenho - Para realizar o teste de performance foi necessário configurar a pilha de memória do java para executar a consulta, contudo o MXQUERY não se saiu bem porque ele não conseguiu completar o trabalho resultando em estouro de memória de 8 GB e *overhead*.

ORACLE XML DB - Mesmo com a utilização do índice Full Text e a configuração de parâmetros a fim de obter a melhor performance, parâmetros tais como, limpeza da memória após a execução de cada consulta, configuração máxima para os limites de saída de dados por página e ativação do otimizador de consultas, foram utilizados, contudo as consultas retornaram com estouro de memória depois de horas de processamento das 10 consultas. Apenas uma consulta retornou com sucesso e pode ser percebido que os dados puderam ser exibidos com sucesso porque o retorno na consulta se referia a um nó cujo conteúdo era menor que 64 bytes, por exemplo, o retorno de um título.

### 5.4 Suporte

BaseX: O suporte está disponível e possui uma equipe ativa que dá assistência por meio de uma lista de email.

IBM DB2: Não oferecem suporte na ferramenta gratuita, o suporte é restrito aos clientes da ferramenta paga restando portanto como alternativa a utilização do fórum, contudo não foi possível obter ajuda satisfatória por meio dos participantes deste fórum.

MXQUERY: não possui suporte porque o projeto foi finalizado em 2009.

ORACLE XML DB Express: O suporte para a versão gratuita do Oracle é feito por meio de um fórum ativo de colaboradores, uma dúvida no fórum chegou a demorar semanas para ser respondida e teve casos em que a resposta não foi útil.

## 5.5 Sumário dos Resultados

Nesta seção será apresentado o sumário dos resultados na unidade de tempo apresentada como milissegundos. A Tabela 6 mostra o tempo de execução das consultas realizadas nas ferramentas que conseguiram chegar até o teste de desempenho onde na coluna Query indica a consulta e nas colunas Oracle índice FT +XML e Oracle índice FT mostram quantas vezes o Oracle XMLDB foi mais lento que o Basex.

Tabela 6- Tempo de execução em ms

Query	Oracle índice FT + XML	Oracle índice FT
1	42x	41x
2	45x	44x
3	65x	1x
4	1x	1x
5	0,2x	0,24x
6	1x	1x
7	65x	1x
8	100x	1x
9	1x	1x
10	1x	1x
	<b>317,2x</b>	<b>85,24x</b>

O resultado acima já era esperado devido ao fato de o Oracle XMLDB ter uma arquitetura híbrida, que suporta tanto relacional quanto XML, e por um outro lado o BaseX ser um SGBD especializado para XML.

A Tabela 7 apresenta a avaliação o resultado da avaliação das ferramentas com valores quantitativos atribuídos para cada atividade do trabalho: instalação, importação, criação de índice, funcionalidade, desempenho e suporte

Tabela 7- Análise baseada na tabela de índices

Etapas avançadas	<b>BASEX</b>	<b>IBM DB2</b>	<b>MXQUERY</b>	<b>ORACLE XML DB</b>
Instalação	2	0	2	2
Importação	2	0	2	1
Criação de índice	1	0	2	0
Funcionalidade	2	0	2	2
Desempenho	2	0	0	1
Suporte	2	0	0	1
<b>Totais</b>	<b>12</b>	<b>1</b>	<b>8</b>	<b>7</b>
<b>Média</b>	<b>2,0</b>	<b>0,16</b>	<b>1,33</b>	<b>1,16</b>

Tabela 7 distribui os dados avaliados da seguinte forma: as colunas são preenchidas com a atividade e os valores atribuídos para cada SGBD as linha Totais representa a soma dos valores atribuídos em cada atividade realizada, a linha média é o resultado da soma dos valores dividida pelas atividades.

## 6 CONCLUSÃO

No contexto de uma empresa que precisa de uma ferramenta para consultar dados XML semiestruturados, mas não dispõem de recursos financeiros para pagar por uma ferramenta proprietária, as ferramentas de código livre e as versões Express das ferramentas pagas representam uma possibilidade de obter uma ferramenta para suas tarefas rotineiras. Foi observado no presente trabalho que para empresas que terão uma base de dados consideravelmente volumosa ou não, a ferramenta que melhor atenderia às necessidades é o SGBD BASEX, pois ele foi a ferramenta que permitiu com que fosse atingido o final dos casos de teste proposto neste trabalho. Com ele foi proporcionado o benefício de ter à disposição a lista de discussão de suporte que têm uma atuação bastante efetiva. Contudo os outros SGBD's analisados apresentaram muitos fatores dificultantes durante a realização deste trabalho, isto faz com que no contexto de um estudante de graduação ou um funcionário de uma empresa sem treinamento especializado não consiga explorar completamente o SGBD's, por falta de ajuda técnica especializada à disposição, e também é um fator desmotivante porque é necessário investimento em dinheiro para isto. Além de tudo a documentação encontrada não é didática e difícil extrair todo conhecimento necessário dentro da documentação dos desenvolvedores.

## 7 REFERÊNCIAS

ABITEBOUL, Serge. **Querying Semi-Structured Data**. Proceedings ICDT 1997.

AMER-YAHIA et al. XQuery Full-Text Extensions Explained. vol 45, no. 2, **IBM SYSTEMS JOURNAL**. 2006.

BAEZA-YATES, Ricardo A. ; RIBEIRO-NETO, Berthier A. . **Modern Information Retrieval - the concepts and technology behind search, Second edition Pearson Education Ltd.**, Harlow, England 2011

BASEX. **BaseX. The XML Database**. 2012. Disponível em: < <http://basex.org/> >. Acesso em: 22 abr. 2012.

BASEX. **Full-Text**. . 2014. Disponível em: <<http://docs.basex.org/wiki/Full-Text> >. Acesso em: 22 abr. 2014.

BASEX. **Index Processing**. . 2014. Disponível em: < [http://docs.basex.org/wiki/Full-Text#Index\\_Processing](http://docs.basex.org/wiki/Full-Text#Index_Processing) >. Acesso em: 22 abr. 2014.

BASEX. **Binary Data**. 2012. Disponível em: < [http://docs.basex.org/wiki/Binary\\_Data](http://docs.basex.org/wiki/Binary_Data) >. Acesso em: 22 abr. 2012.

BEYER, Kevin S. et al. **System RX: One Part Relational, One Part XML**. SIGMOD Conference 2005: 347-358

CHEN FENG, BAOCHUN E LI, BO LI. **Capítulo Conclusão**. Understanding the Performance Gap between Pull-based Mesh Streaming Protocols and Fundamental Limits. Ca 2000. Disponível em: < <http://iqua.ece.toronto.edu/~bli/papers/cfeng-infocom09.pdf> >. Acesso em 15 jun. 2014.

CLUSTERPOINT. **Full Text Search Speed Benchmarks**. Disponível em: < <http://www.clusterpoint.com/performance/full-text-search-speed-benchmarks.html> >. Acesso em: 22 abr. 2012.

DIAO et al. **3 The BEA Streaming XQuery Processor**. Implementing memoization in a streaming XQuery processor. Ca 2000. Disponível em < <http://people.cs.umass.edu/~yanlei/publications/xsym-2004.pdf> >

DB2 Express-C, Ca 2000. Disponível em: < <http://db2express.com/pt/> >. Acesso em: 15 jun. 2015.

EXIST. **eXist-db Open Source Native XML Database**. Ca. 2000. Disponível em: < <http://exist-db.org/exist/index.xml;jsessionid=kuz1ysjus8he1p7vjtvuuxtjf> >. Acesso em: 22 abr. 2012.

FLORESCU, Daniela. Managing Semi-Structured Data. Vol. 3 No. 8, Outubro 2005, **ACMQUEUE**. p.18 – 24.

FLORESCU et al. **BEA Streaming XQuery Processor**. VLDB Journal.,2004. Disponível em: < <http://www.vldb.org/conf/2003/papers/S30P01.pdf> >. Acesso em: 15 jun. 2014.

HOWARD KATZ. **Influences On the Design of XQuery**. In: Howard Katz et al. XQuery from de Experts: A Guide to the W3C XML Query Language. 2003. p.81 – 103.

FLORESCU et al. **4 XQuery Engine Overview**. The BEA/XQRL Streaming XQuery Processor . CA 2000. Disponível em: <<http://www.vldb.org/conf/2003/papers/S30P01.pdf>>. Acesso em 15 jun. 2014.

GRAY, Jim. **The Benchmark Handbook for Database and Transaction Systems**, 2nd. Morgan Kaufmann 1993, ISBN 1-55860-292-5

GRUN C. ,et al. **XQuery Full Text Implementation in BaseX**. Ca. 2000. Disponível em: <<http://www.inf.uni-konstanz.de/gk/pubsys/publishedFiles/GrGaHo09.pdf>>. Acesso em: 01 fev. 2014.

HAUSTEIN, Michael Peter; HÄRDER, Theo. **An efficient infrastructure for native transactional XML processing**. Data Knowl. Eng. 61(3): 500-523, 2007

IBM. **Text Search Guide**. Ca 2000. Disponível em: <[ftp://ftp.software.ibm.com/ps/products/db2/info/vr101/pdf/en\\_US/DB2TextSearch-db2tse1010.pdf](ftp://ftp.software.ibm.com/ps/products/db2/info/vr101/pdf/en_US/DB2TextSearch-db2tse1010.pdf)>. Acesso em 10 set. 2013.

IBM. **PureXML Guide**. 2013. Disponível em:< [http://froebe.net/blog/wp-content/uploads/2013/09/DB2-10.1-LUW-pureXML-Guide-IBM-Inc\\_.pdf](http://froebe.net/blog/wp-content/uploads/2013/09/DB2-10.1-LUW-pureXML-Guide-IBM-Inc_.pdf)>. Pag 2. Acesso em: 01 jan. 2014.

INEX. **About INEX**. Ca. 2000. Disponível em: < <https://inex.mmci.uni-saarland.de/about.html> >. Acesso em 22 abr. 2014

MELLO, R. S. et al. **Dados Semi-Estruturados**. In: SIMPÓSIO BRASILEIRO DE BANCO DE DADOS, SBBD, 15. 2000, João Pessoa. **Anais...** João Pessoa: [s.n.], 2000

MXQUERY. **MXQuery on Android – Overview**. Ca. 2000. Disponível em: < [http://mXQuery.org/?page\\_id=250](http://mXQuery.org/?page_id=250)>. Acessado em 15 Jun. 2014.

\_\_\_\_. **About Ca. 2000.** Disponível em: < [http://mXQuery.org/?page\\_id=2](http://mXQuery.org/?page_id=2) >. Acessado em 15 Jun. 2014.

\_\_\_\_. **Planned features, fixes and enhancements.** Ca. 2000. Disponível em: <https://svn.code.sf.net/p/mXQuery/code/trunk/MXQuery/Documentation/Roadmap>. Acesso em 10 jun. 2014.

\_\_\_\_. **MXQuery 0.6.0 released.** 06 mai. 2009. Disponível em: < (<http://mXQuery.org/?p=198>)> . Acesso em 15 jun. 2014.

ORACLE. Oracle® Database Application Developer's Guide - Large Objects 10g Release 2 (10.2) B14249-01. **Terabyte-Size LOB Support.** 06/2005 Disponível em: <[http://docs.oracle.com/cd/B19306\\_01/appdev.102/b14249.pdf](http://docs.oracle.com/cd/B19306_01/appdev.102/b14249.pdf)>. Acessado em: 01 mai. 2014

\_\_\_\_. Oracle® XML DB Developer's Guide 10g Release 2 (10.2) B14259-02. E **Oracle XML DB Restrictions.** 08/2005. Disponível em: < [http://docs.oracle.com/cd/B19306\\_01/appdev.102/b14259/appjspec.htm](http://docs.oracle.com/cd/B19306_01/appdev.102/b14259/appjspec.htm) >. Acesso em 22 abr. 2012.

\_\_\_\_. **Oracle® Database New Features Guide 11g Release 2 (11.2) E41360-03.** Disponível em: < [http://docs.oracle.com/cd/E11882\\_01/server.112/e10881/chapter1.htm#NEWFT293](http://docs.oracle.com/cd/E11882_01/server.112/e10881/chapter1.htm#NEWFT293) >. Acesso em 22 abr. 2012.

\_\_\_\_. **Oracle® Database Licensing Information 11g Release 2 (11.2) E47877-01.** 06/2014 Disponível em: < [http://docs.oracle.com/cd/E11882\\_01/license.112/e47877.pdf](http://docs.oracle.com/cd/E11882_01/license.112/e47877.pdf) > Acesso em 22 abr. 2012.

\_\_\_\_. Oracle® XML DB Developer's Guide 11g Release 2 (11.2) E23094-04. **XMLIndex Unstructured Component.** Disponível em: <



[http://docs.oracle.com/cd/E11882\\_01/appdev.112/e23094/xdb\\_indexing.htm#ADXDB4344](http://docs.oracle.com/cd/E11882_01/appdev.112/e23094/xdb_indexing.htm#ADXDB4344) >. Acesso em 22 abr. 2012.

\_\_\_\_.Oracle® XML DB Developer's Guide 11g Release 2 (11.2) E23094-04.  
**Oracle Text Indexes**. Disponível em: <  
[http://docs.oracle.com/cd/E11882\\_01/appdev.112/e23094/xdb\\_indexing.htm#ADXDB5817](http://docs.oracle.com/cd/E11882_01/appdev.112/e23094/xdb_indexing.htm#ADXDB5817)>. Acesso em 22 abr. 2012.

\_\_\_\_.Oracle® Text Application Developer's Guide 11g Release 1 (11.1) B28303-03. **Optimizing the Index** . 08/2008 Disponível em: <  
[http://docs.oracle.com/cd/B28359\\_01/text.111/b28303.pdf](http://docs.oracle.com/cd/B28359_01/text.111/b28303.pdf) >. Acessado em 19 Jan. 2014.

\_\_\_\_.Oracle® Text Application Developer's Guide 12c Release 1 (12.1) E17748-07. **The Oracle Text Indexing Process** .04/2013. Disponível em:<  
[http://docs.oracle.com/cd/E16655\\_01/text.121/e17748.pdf](http://docs.oracle.com/cd/E16655_01/text.121/e17748.pdf) >. Acesso em Acesso em: 22 abr. 2012

\_\_\_\_[B]. Oracle® XML DB Developer's Guide -11g Release 1 (11.1) -B28369-04. **Features of Oracle XML DB**. 05/2008. Disponível em: <  
[http://docs.oracle.com/cd/B28359\\_01/appdev.111/b28369.pdf](http://docs.oracle.com/cd/B28359_01/appdev.111/b28369.pdf) >. Acesso em 20 Jan. 2014.

\_\_\_\_.Oracle XML DB: **Best Practices to Get Optimal Performance out of XML Queries**. 01/2013. Disponível em: <  
<http://www.oracle.com/technetwork/database-features/xmldb/xmlqueryoptimize11gr2-168036.pdf> >. Acessado em 19 Jan. 2014.

\_\_\_\_.**Oracle Database 11g Express Edition Quick Tour**. 09/2011. Disponível em: <<http://www.oracle.com/technetwork/articles/sql/11g-xe-quicktour-498681.html> >. Acesso em: 01 mar. 2014.

\_\_\_\_.Oracle White Paper— **Oracle XML DB : Choosing the Best XMLType Storage Option for Your Use Case**. 2013. Disponível em: < <http://www.google.com.br/url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&cad=rja&ved=0CEYQFjAC&url=http%3A%2F%2Fwww.oracle.com%2Ftechnetwork%2Fdatabase%2Ffeatures%2Fxmldb%2Fxmldb-store-12cr1-2043451.pdf&ei=Y-F7Uq7uL4iQiAeuqoDAAG&usg=AFQjCNFnjJtvxINCuOjsMRKwOA7Q2ybitQ&bvm=bv.56146854,d.aGc> >. Acesso em: 18 jan. 2014.

\_\_\_\_.Oracle® Database SQL Language Reference, 12c Release 1 (12.1) E17209-15. 12/2013. Disponível em: < [http://docs.oracle.com/cd/E16655\\_01/server.121/e17209.pdf](http://docs.oracle.com/cd/E16655_01/server.121/e17209.pdf) >

\_\_\_\_.Oracle® XML DB Developer's Guide 12c Release 1 (12.1)E17603-09. **Supported XQuery Full Text FTSelection Operators**. 05/2013. Disponível em: < [http://docs.oracle.com/cd/E16655\\_01/appdev.121/e17603.pdf](http://docs.oracle.com/cd/E16655_01/appdev.121/e17603.pdf) >. Acesso em 22 abr. 2012

[A]\_\_\_\_.Oracle® XML DB Developer's Guide 12c Release 1 (12.1)E17603-09. **Oracle XML DB Restrictions**. 05/2013. Disponível em: < [http://docs.oracle.com/cd/E16655\\_01/appdev.121/e17603.pdf](http://docs.oracle.com/cd/E16655_01/appdev.121/e17603.pdf) >. Acesso em 22 abr. 2012

\_\_\_\_. **Oracle XML DB in Oracle Database 12c**. 26/06/2013. Disponível em: < <http://www.oracle.com/technetwork/database-features/xmldb/overview/xmldb-twp-12cr1-1964803.pdf> >. Acesso em 10 mar. 2014.

SEDNA. **Native Xml Database System**. Ca. 2003. Disponível em: < <http://www.sedna.org/> >. Acesso em: 22 abr. 2012.

SILVA, E. L, MENEZES, E. M.; **Metodologia da pesquisa e elaboração de dissertação**. 3. ed. rev. atual. – Florianópolis: Laboratório de Ensino a Distância da UFSC, 2001. 121p.

TSOULO, K. . **XML Schema Support in MXQuery. 2008. Tese de Mestrado. Disponível em:** <<http://e-collection.library.ethz.ch/eserv/eth:30589/eth-30589-01.pdf>>. Acesso em: 15 Jun. 2014.

VAKALI, A. ,PALLIS, G. **Web Data Management Practices: Emerging Techniques and Technologies.** 2007. Disponível em: <[http://books.google.com.br/books?id=z9SyhvVcfzkC&pg=PA209&lpg=PA209&dq=mXQuery+stored&source=bl&ots=fKeanh0EtJ&sig=68n11U\\_FrcHyd5HsT8iBxy1Jwgc&hl=pt-BR&sa=X&ei=YSrU9b3OOissQSR44D4Cw&ved=0CGQQ6AEwCQ#v=onepage&q=mXQuery%20stored&f=false](http://books.google.com.br/books?id=z9SyhvVcfzkC&pg=PA209&lpg=PA209&dq=mXQuery+stored&source=bl&ots=fKeanh0EtJ&sig=68n11U_FrcHyd5HsT8iBxy1Jwgc&hl=pt-BR&sa=X&ei=YSrU9b3OOissQSR44D4Cw&ved=0CGQQ6AEwCQ#v=onepage&q=mXQuery%20stored&f=false)>. Acesso em: 25 jun. 2014.

W3C. **XQuery and XPath Full Text 1.0.** 2011. Disponível em: <<http://www.w3.org/TR/xpath-full-text-10/>>. Acesso em 22 abr. 2012.

\_\_\_\_. **XQuery and XPath Full Text 1.0 Use Cases.** 2001. Disponível em: <[http://www.w3.org/TR/xpath-full-text-10-use-cases/#FT\\_UC\\_Prelim](http://www.w3.org/TR/xpath-full-text-10-use-cases/#FT_UC_Prelim)>. Acesso em: 22 abr. 2012.

\_\_\_\_. **XQuery and XPath Full Text 1.0 Use Cases.** 2001. Disponível em: <[http://www.w3.org/TR/xpath-full-text-10-use-cases/#FT\\_UC\\_SampleData](http://www.w3.org/TR/xpath-full-text-10-use-cases/#FT_UC_SampleData)>. Acesso em: 04 jun. 2013.

\_\_\_\_. **Use Case "XQUERY-XPATh-COMPOSABILITY": Queries Illustrating Composability of Full-Text with Other XQuery and XPath Functionalities,** 2011. Disponível em: <<http://www.w3.org/TR/xpath-full-text-10-use-cases/#XQuery-XPath-Composability>>. Acesso em 19 set. 2012.

\_\_\_\_. **XML Information Set,** 2001. Disponível em: <<http://www.w3.org/TR/2001/WD-xml-infoSet-20010316/>>. Acesso em: 10 out. 2012.

\_\_\_\_. **XML Path Language (XPath) Version 1.0**: W3C Recommendation 16 November 1999. Disponível em: < <http://www.w3.org/TR/xpath/>>. Acesso em: 22 abr. 2012.

\_\_\_\_. **2.5 Sequences**. 14/12/2010. Disponível em: < <http://www.w3.org/TR/xpath-datamodel/#sequences>>. Acesso em: 22 abr. 2012.

\_\_\_\_. **3.1 Primary Expressions** . 14/12/2010. Disponível em: < <http://www.w3.org/TR/XQuery/#doc-XQuery-Expr>>. Acesso em: 22 abr. 2012.

W3SCHOOLS . **XPath Syntax**. Ca. 2000. Disponível em: < [http://www.w3schools.com/xpath/xpath\\_syntax.asp](http://www.w3schools.com/xpath/xpath_syntax.asp)>. Acesso em: 22 abr. 2012.

W3SCHOOLS . **XML DOM Node Types**. Ca. 2000. Disponível em: < [http://www.w3schools.com/dom/dom\\_nodetype.asp](http://www.w3schools.com/dom/dom_nodetype.asp)>. Acesso em: 7 jan. 2014.

XQBENCH. **XQBench, XQuery Benchmark Environment**. Ca. 2000. Disponível em: < <http://xqbench.org/>>. Acesso em: 22 abr. 2012.

\_\_\_\_ **XQbench documentation**. 2007. Disponível em : < <https://trac.systems.inf.ethz.ch/trac/systems/xqbench/wiki> > . Acesso em: 19 set. 2012.

## 1 APÊNDICES

### Apêndice I

Seleções Full-Text	Utilização
Match Options	Usada para modificar o comportamento da seleção
Language Option	Especificar a linguagem a ser usada na consulta
Wildcard Option	Especifica se haverá reconhecimento de wildcards

	(caracteres em partes da palavra)
Thesaurus Option	Especifica se irá usar Thesaurus
Stemming Option	Especifica se irá usar Stemmin (radical de palavras)
Case Option	Especifica se serão considerados caracteres maiúsculos ou minúsculos
Diacritics Option	Especifica como serão tratados os sinais gráficos nas palavras
Stop Word Option	Especifica palavra para interromper a consulta
Extension Option	Identificação de um QName and a StringLiteral
Logical Full-Text Operators	Operadores lógicos para Full Text
Or-Selection	Operador OU
And-Selection	Operador E
Mild-Not Selection	Operador NOT IN
Not-Selection	Operador de negação NOT
Full-Text Operators Semantics	
FTOr	Resultado é o casamento de cada uma dos casamentos em <b>Match</b> do FTSelection
FTAnd	Resultado de dois casamentos de aninhados de FTSelection
FTUnaryNot	Negação do resultado aninhado de <b>FTSelection</b>
FTMildNot	Resultado de duas <b>FTSelection</b>
FTOrder	Resultado aninhado de um <b>FTSelection</b>
FTScope	Resultado de um <b>FTSelection</b> dependendo do escopo

FTContent	Resultado de busca por contexto ou conteúdo
FTWindow	Resultado de <b>FTSelection reunindo tipo e tamanho</b>

## Apêndice II

-----  
 -- Tabela com coluna xml:

-----  
 CREATE TABLE ORACLE.TB\_INEX2009 (DOC VARCHAR(30) NOT  
 NULL, XML XMLTYPE, PRIMARY KEY (DOC))

-----  
 --- Criando diretório para os dados  
 -- CREATE DIRECTORY xmldirInex2 AS  
 '/home/arwen/ORACLE\_oracle/loading\_inex2009/wikixml';

-----  
 --desabilitando indice

ALTER TABLE ORACLE.TB\_INEX2009 MODIFY PRIMARY KEY  
 DISABLE CASCADE

-- Habilitando indice

ALTER TABLE ORACLE.TB\_INEX2009 ENABLE PRIMARY KEY  
 EXCEPTIONS INTO except\_table

-----  
 -- criando text index

--CREATE INDEX Itxt\_TB\_INEX2009 ON ORACLE.TB\_INEX2009 (XML)  
 INDEXTYPE IS CTXSYS.CONTEXT

-----  
 CREATE INDEX Itxt\_TB\_INEX2009  
 ON ORACLE.TB\_INEX2009(XML)

```
INDEXTYPE IS ctxsys.context
PARAMETERS ('datastore ctxsys.default_datastore
            filter ctxsys.null_filter
            section group ctxsys.auto_section_group'
            )
```

---

```
-----
-- índice com base nas consultas
CREATE INDEX ORACLE.Ixml_TB_INEX2009 ON
ORACLE.TB_INEX2009 (XML) INDEXTYPE IS XDB.XMLINDEX
PARAMETERS ('PATHS (INCLUDE(
//article//language
//article
//article//p
//article//sec
//article//person
//ss1/p
//article//title
//article//bdy
//article//film_festival
//article//movie
//article//actor
//article//actress
//article/header/title/header/id
//article//link
//article//link/*
))'
/
```

### Apêndice III

Script das consultas executadas no teste de performance:

Query:

```
collection("bd_inex2009")//article//language[ . contains text "java" ]
```

Query:

```
collection("bd_inex2009")//article[.//text() contains text "Nobel prize" ]
```

Query:

```
for $book in collection("bd_inex2009")//article let $intro :=
$book//(p|sec|person) where $intro [text() contains text "star" using wildcards]
return $book
```

Query:

```
for $book in collection("bd_inex2009")//ss1 let $cont := $book/p where $cont
[text() contains text "fast" using stemming] return $book
```

Query:

```
for $book in collection("bd_inex2009")//ss1 let $cont := $book/p where $cont
[text() contains text "fast" using stemming] return $book
```

Query:

```
for $book in collection("bd_inex2009")//article let $title := $book//title where
$title [text() contains text "The" fband "Love" ordered distance at most 2 words
at start] return $title
```

Query:



```
for $article in collection("bd_inex2009")//article let $cont := $article/bdy where
$cont contains text "The" ftand "power" ftand "Love" ordered distance at most
10 words return $article
```

Query:

```
for $book in collection("bd_inex2009")//article let $cont := $book where $cont [
.//(film_festival|movie|actor|actress) contains text ("israeli" ftand "director"
using stemming) ffor ("israeli" using stemming ftand "actor" using stemming)]
return $book
```

Query:

```
for $article in collection("bd_inex2009")//article let $var := $article/header/title
where $var contains text "Love" return <result>
{ $article/header/title,$article/header/id } </result>
```

Query:

```
for $book in collection("bd_inex2009")//article let $sup := $book/bdy where $sup
[ text() contains text "power" ftand fnot "love"] return $book
```

Query:

```
for $book in collection("bd_inex2009")//article where $book//link[@*][ text()
contains text "Nation" using stemming ftand "Day" distance at most 2 words]
return $book/bdy/b
```

#### Apêndice IV

Variação de sintaxe entre as ferramentas dada uma mesma consulta:

**BASEX:**

```
XQUERY collection("bd_inex2009")//article//language[ . contains text "java"
all]
```

**ORACLE:**

```
XQUERY fn:collection("oradb:/ORACLE/TB_INEX2009")//article//language
[ora:contains(text(), "java") > 0]
/
```

**MXQUERY:**

```
fn:doc('todosxml.xml')//article//language[ . ftcontains 'java']
```

## Apêndice V

A ativação e a criação de índice Full Text são realizados pelas seguintes linhas de comando: (<http://docs.basex.org/wiki/Full-Text#Introduction>)

```
SET FTINDEX true; CREATE DB input.xml  
CREATE INDEX fulltext
```

## 2 ANEXOS

### Anexo I

```
<books>
<book number="1">
  <metadata>
    <title shortTitle="Improving Web Site Usability">Improving
the Usability of a Web Site Through Expert Reviews and
Usability Testing</title>
    <author>Millicent Marigold</author>
    <author>Montana Marigold</author>
    <publicationInfo>
      <place>New York</place>
      <publisher>Ersatz Publications</publisher>
      <dateIssued>2001</dateIssued>
      <dateRevised>2002</dateRevised>
    </publicationInfo>
    <price>25.99</price>
    <subjects xml:lang="en">
      <subject>Usability testing</subject>
      <subject>Web site development</subject>
      <subject>Heuristic evaluation</subject>
      <subject>Cognitive walk-through</subject>
      <subject>Web site usability</subject>
    </subjects>
```

```

<subjects xml:lang="fr">
  <subject>Tests d'ergonomie</subject>
  <subject>Développement de site web</subject>
  <subject>Évaluation heuristique</subject>
  <subject>Parcours cognitif</subject>
  <subject>Ergonomie de site web</subject>
</subjects>
<subjects xml:lang="zh">
  <subject>????</subject>
  <subject>????</subject>
  <subject>????</subject>
  <subject>????</subject>
  <subject>????</subject>
</subjects>
</metadata>
<content>
  <introduction>
    <author>Elina Rose</author>
    <p>The usability of a Web site is how well the
    site supports the user in achieving specified
    goals. A Web site should facilitate learning,
    and enable efficient and effective task
    completion, while propagating few errors.
    Satisfaction with the site is also important.
    The user must not only be well-served, but must
    feel well-served.</p>
    <p>Expert reviews and usability testing are
    methods of identifying problems in layout,

```

```

terminology, and navigation before they frustrate
users and drive them away from your site.</p>
<p>The most successful projects employ multiple
methods in multiple iterations. As Millicent
Marigold remarked during a recent conference,
"Don't stop. Iterate, iterate, then iterate
again."</p>
<p>This book has been approved by the Web Site
Users Association.</p>
</introduction>
<part number="1">
<title>Expert Reviews</title>
<introduction>
<p>Expert reviewers identify problems
and recommend changes to web sites based
on research in human computer interaction
and their experience in the field.</p>
<p>Two expert review methods are discussed
here. They are heuristic evaluation and
cognitive walk-through.</p>
<p>Expert review methods should be
initiated early in the development process,
as soon as paper <b>p</b>rototypes
(hand-drawn pictures of Web pages) or
<b>w</b>ireframes (electronic mockups) are
available. They should be conducted using
the hardware and software similar to that
employed by users.</p>

```

```
</introduction>
<chapter>
  <title>Heuristic Evaluation</title>
  <p>Expert reviewers critique an interface to
  determine conformance with recognized
  usability principles. <footnote>One of the
  best known lists of heuristics is <citation
  url="http://www.useit.com/papers/heuristic
  /heuristic_list.html">Ten Usability
  Heuristics by Jacob Nielsen</citation>. Another
  is <citation url="http://usability.gov
  /guidelines/index.html"> Research-Based Web
  Design and Usability Guidelines</citation>
  </footnote></p>
</chapter>
<chapter>
  <title>Cognitive Walk-Through</title>
  <p>Expert reviewers evaluate Web site
  understandability and ease of learning while
  performing specified tasks. They walk through
  the site answering questions such as "Would a
  user know by looking at the screen how to
  complete the first step of the task?" and "If
  the user completed the first step, would the
  user know what to do next?," with the goal of
  identifying any obstacles to completing the
  task and assessing whether the user would
  cognitively be aware that he was successful in
```

```
        completing a step in the process.</p>
    </chapter>
</part>
<part number="2">
    <chapter>
        <title>Usability Testing</title>
        <p>Once the problems identified by expert
        reviews have been corrected, it is time to
        conduct some tests of the site with your unique
        audience or audiences by conducting usability
        testing.</p>
        <p>Users are asked to complete tasks which
        measure the success of the information
        architecture and navigational elements of the
        site.</p>
        <p>Then changes are made to improve service to
        users.</p>
    </chapter>
</part>
</content>
</book>

<book number="2">
    <metadata>
        <title shortTitle="Usability Basics">Usability
        Basics: How to Plan for and Conduct Usability Tests
        on Web Site Thereby Improving the Usability of Your
        Web Site</title>
```



```

<publicationInfo>
  <place>New York</place>
  <publisher>Ersatz Publications</publisher>
  <publisher>Electronic BookWorks</publisher>
  <dateIssued>2000</dateIssued>
  <dateRevised>2001</dateRevised>
</publicationInfo>
<price>174.00</price>
<subjects xml:lang="en">
  <subject>Usability testing</subject>
  <subject>Web site development</subject>
  <subject>Guides and finding aids</subject>
</subjects>
<subjects xml:lang="fr">
  <subject>Tests d'ergonomie</subject>
  <subject>Développement de site web</subject>
  <subject>Guides et outils de recherche</subject>
</subjects>
<subjects xml:lang="zh">
  <subject>????</subject>
  <subject>????</subject>
  <subject>??????</subject>
</subjects>
</metadata>
<content>
  <introduction>
    <p>This is a basic handbook for planning and
    conducting usability tests on Web sites. Usability
  
```

```
testing should be used in conjunction with other
expert review methods.</p>
<p>This book has not been approved by the Web Site
Users Association.</p>
</introduction>
<part number="1">
  <chapter>
    <title>Planning then Conducting Usability
    Tests</title>
    <p>Take the following steps to plan usability
    testing. <step number="1">Clarify and
    articulate the goal of the usability testing.
    </step> <step number="2">Identify tasks which
    are critical for users to be able to complete
    successfully.</step> <step number="3">Compile
    a script of questions or instructions which
    will prompt the user to attempt those
    tasks.</step> <step number="4">Identify your
    users and begin recruiting them.</step> <step
    number="5">Conduct a pretest on a few users.
    </step> <step number="6">Edit the script based
    on insights gleaned from the pretest.</step>
    <step number="7">Resume testing.</step></p>
  </chapter>
</part>
<part number="2">
  <chapter>
    <title>Conducting Usability Tests</title>
```

<p>Users can be tested at any computer workstation <footnote>They may be more comfortable at their own workstation than in a lab.</footnote> or in a lab.</p>  
<p>Give the user the script, then assure them that you are testing the Web site, not them. Users are asked to verbalize their thoughts as they complete the tasks. The event is recorded or someone takes notes. It is often preferable to have two testers, <footnote>Usability testing can be done at great expense or on a shoe string, using <testingProcedure>in-house expertise</testingProcedure> or <testingProcedure>contracting with human computer interaction professionals </testingProcedure>.</footnote> one to ask the questions, another to take notes. Testers should offer no guidance or comments to the user. Mouse movements, typing, expressions, and the user's words should be recorded.</p>

</chapter>

<chapter>

<title>Evaluating and Implementing Results</title>

<p>Compile the results and review collectively. Make changes to the site to alleviate the problems found in Web site components which were propagating the largest number of or the most devastating errors. Begin new iterations of testing and changes, until

```

        users are successful in the accomplishing the
        tasks.</p>
    </chapter>
</part>
</content>
</book>

<book number="3">
    <metadata>
        <title shortTitle="Usabilityguy Manuscript
        Guide">John Wesley Usabilityguy: A Register of His
        Papers</title>
        <author>Millicent Marigold</author>
        <author>Morty Marigold</author>
        <publicationInfo>
            <place>Washington, D.C.</place>
            <publisher>Ersatz Manuscript Library</publisher>
            <dateIssued>1998</dateIssued>
            <dateRevised>2002</dateRevised>
        </publicationInfo>
        <price>21.49</price>
        <subjects xml:lang="en">
            <subject>Computers</subject>
            <subject>Software evaluation</subject>
            <subject>Usability testing</subject>
            <subject>Manuscript collections</subject>
        </subjects>
        <subjects xml:lang="fr">

```

```

<subject>Ordinateurs</subject>
<subject>Évaluation de logiciels</subject>
<subject>Tests d'ergonomie</subject>
<subject>Collections de manuscrits</subject>
</subjects>
<subjects xml:lang="zh">
  <subject>??</subject>
  <subject>??</subject>
  <subject>??</subject>
  <subject>??</subject>
</subjects>
</metadata>
<content>
  <introduction>
    <p>The papers of John Wesley Usabilityguy span the
    years 1946-2001, with the bulk of the items
    concentrated in the period from 1985 to 2001. The
    papers feature his career as a developer of software
    applications and usability specialist. The collection
    consists of correspondence, memoranda, journals,
    speeches, article drafts, book drafts, notes, charts,
    graphs, family papers, clippings, printed matter,
    photographs, résumés and other materials.</p>
  </introduction>
  <part number="1"><container type="box">1-12</container>
    <title>Subject File, <date normalize="1930/1974">
    1930-1974</date></title>
    <introduction>

```

<p>Correspondence, telegrams, memoranda, journals, logs, testimony, approved travel orders, invitations, charts, graphs, forms, biographical data, photographs, book drafts, clippings and other printed matter, résumés and miscellaneous material. Organized by name of person or organization, topic, or type of material.</p>

</introduction>

<component><container type="box">1</container>

<componentTitle>Computers</componentTitle>

<subComponent>

<componentTitle>Software,

<componentDate normalize="1946/1947">1946-1947

</componentDate>

</componentTitle>

</subComponent>

<subComponent>

<componentTitle>Human Computer Interaction

research, <componentDate normalize="1945/1952">1945-1952</componentDate>

</componentTitle>

<subsubComponent>

<componentTitle>Flow diagram,

<componentDate normalize="1950">1950

</componentDate>

</componentTitle>

</subsubComponent>

<subsubComponent>

```

    <componentTitle>General,
    <componentDate normalize="1947/1951">1947-1951
    </componentDate>
  </componentTitle>
</subsubComponent>
<subsubComponent><container type="box">2</container>
  <componentTitle>Eye Movement research,
  <componentDate normalize="1949/1950">1949-1950
  </componentDate>
  </componentTitle>
</subsubComponent>
<subsubComponent>
  <componentTitle>User profiling,
  <componentDate normalize="1950/1959">1950s
  </componentDate>
  </componentTitle>
</subsubComponent>
</subComponent>
</component>
<component>
  <componentTitle>Web User Appreciation Award,
  <componentDate normalize="1956">1956</componentDate>
  </componentTitle>
</component>
</part>
<part number="2"><container type="box">3-5</container>
  <title>Writings File,
  <date normalize="1985/1999">1985-1999</date>

```

```

</title>
<introduction>
  <p>Correspondence, articles, book drafts, notes,
  contracts, clippings, and printed matter. Arranged
  alphabetically by type (articles, books, reports,
  and miscellaneous) and therein alphabetically by
  type of material, subject, or title.</p>
</introduction>
<component>
  <componentTitle>Writings by Usabilityguy
  </componentTitle>
  <subComponent>
    <componentTitle><componentDate normalize="1996">
    1996</componentDate>
    </componentTitle>
    <subsubComponent>
      <componentTitle>"How Many Users Are Enough
      for User Testing?"</componentTitle>
    </subsubComponent>
    <subsubComponent>
      <componentTitle>"How to Evaluate Results from
      User Tests."</componentTitle>
    </subsubComponent>
    <subsubComponent>
      <container type="box">5</container>
      <componentTitle>"When Are You Done Testing?"
      </componentTitle>
    </subsubComponent>
  </subComponent>
</component>

```



```

    <subsubComponent>
      <componentTitle>"Do-It-Yourself User Testing"
    </componentTitle>
    </subsubComponent>
  </subComponent>
</component>
<component>
  <componentTitle>Charitable Contributions
</componentTitle>
  <subComponent>
    <componentTitle>Diseases: AIDS, Hepatitis,
    Tuberculosis <componentDate normalize=
    "1990/1999">1990-1999</componentDate>
    </componentTitle>
  </subComponent>
  <subComponent>
    <componentTitle>Environmental Conservation:
    Rivers <componentDate normalize="1995">1995
    </componentDate>
    </componentTitle>
  </subComponent>
</component>
</part>
</content>
</book>
</books>

```

## Anexo II

O exemplo abaixo facilita a compreensão da estrutura de um XML. A sequência está marcada com sublinhas, o segundo nó é o comentário, seguido do elemento *book* e o atributo *year*. O elemento título, o nó texto contém TCP/IP Illustrated e o elemento autor contém o último elemento contendo Stevens e o primeiro elemento contendo W.

```

<!-- document order -->
<book year="1994">
<title>TCP/IP Illustrated</title>
<author>
<last>Stevens</last>
<first>W.</first>
</author>
</book>

```

XPath usa expressões de caminho para selecionar nós ou conjunto de nós em um documento XML. O nó é selecionado seguindo um caminho ou passos. A tabela a seguir lista algumas seleções em XPath.

<b>Expressão</b>	<b>Descrição</b>
<i>nodename</i>	Seleciona todos os nós com o nome " <i>nodename</i> "
/	Seleciona o nó raiz
//	Seleciona os nós no documento corrente que correspondam a seleção não importando onde eles estão.

.	Seleciona o nó corrente
.	Seleciona o pai do nó atual
@	Seleciona atributos

Tabela 1: Expressões

Fonte: W3CSCHOOLS [ca. 2000].

Expressões Path são utilizadas para selecionar partes dentro da hierarquia do documento, tais expressões são formadas por passos separados por barras (“/”), o eixo *child* é o padrão quando está especificado no eixo com “:.” como no exemplo. As consultas abaixo buscam pelos autores com o nome igual a "Montana Marigold".

```
/child::books/child::book[child::author = "Montana Marigold"]
  /child::title
```

De forma abreviada a expressão acima poderia ser escrita da seguinte maneira:

```
/books/book[author = "Montana Marigold"]/title
```

A tabela abaixo lista algumas expressões Path e os seus resultados.

Expressão Path	Resultado
bookstore	Seleciona todos os nós com "bookstore"
/bookstore	Seleciona o elemento raiz bookstore <b>Nota:</b> Se o caminho começa com uma barra (/) ele sempre representa o caminho absoluto para um elemento!
bookstore/book	Seleciona todos os elementos Book que são filhos de bookstore
//book	Seleciona todos os elementos book não importando onde eles se encontram no documento

bookstore//book	Seleciona todos os elementos Book que são descendentes do element bookstore, não importando onde eles estão sob o elemento bookstore
//@lang	Selecione todos os atributos que são nomeados lang

**Tabela 2: Expressões**

Fonte: W3CSCHOOLS [ca. 2000]

Predicados são usados para encontrar um nó específico ou um nó que contém um valor específico, (W3CSCHOOLS, ca. 2000)

<b>Expressões Path</b>	<b>Resultado</b>
/bookstore/book[1]	Seleciona o primeiro element Book que é o filho do element bookstore.
/bookstore/book[last()]	Seleciona o ultimo elemento Book que é filho do elemento bookstore.
/bookstore/book[last()-1]	Seleciona o ultimo mais um elemento book que é filho do elemento bookstore.
/bookstore/book[position()<3]	Selects the first two book elements that are children of the bookstore element
//title[@lang]	Seleciona todos os elementos title que tem um atributo nomeado lang
//title[@lang='eng']	Seleciona todos os elementos title que tem uma atributo nomeado lang com o valor 'eng'.
/bookstore/book[price>35.00]	Seleciona todos os elementos book do elemento bookstore que tem um elemento price com um valor maior que 35.00.
/bookstore/book[price>35.00]/title	Seleciona todos os elementos title dos elementos book do elemento bookstore que tem um elemento price com um valor maior que 35.00.

**Tabela 3: Expressões**

Fonte: W3CSCHOOLS [ca. 2000]

No exemplo abaixo a consulta retorna os títulos dos livros que contém os termos improve, web e usability. (AMER-YAHIA et al, 2006)

```
./@shortTitle ftcontains "improve" && "web" && "usability")
```

É possível tornar uma consulta mais específica ou mais detalhada por meio de uma busca aos outros nós, aprofundando na árvore por meio da mistura de XQuery com FTContainsExpr como na consulta do exemplo a seguir que retornará os títulos dos livros cujos que contêm o termo usabilitytests e os títulos dos capítulos que tiverem a frase web-site usability.

```
//book[(metadata:ftcontains"usabilitytests")and  
  (content/part/chapter/title ftcontains  
  "web-site usability")] /title
```

FTSelection Expressions são usados para especificar a condição Full Text em uma expressão FTContainsExpr.

A função ft:search retorna todos os nós do índice Full Text do banco de dados \$db que contém o termo especificado \$terms. Sintaxe utilizada ft:search(\$db as xs:string, \$terms as item()\*) as text()\* . Exmpl: Busca os nós que contém o termo QUERY.

```
ft:search("DB", "QUERY")
```

A função ft:contains verifica se um item especificado \$input contém o \$term especificado. Ela faz o mesmo que expressão Full Text

contains text, mas as opções podem ser especificadas mais dinamicamente. Sintaxe ft:contains(\$input as item()\*, \$terms as item()\*) as xs:boolean. Exemplo: Verifica se Jack ou John ocorre na string de entrada John Doe.

```
ft:contains("John Doe", ("jack", "john"), { "mode": "any" })
```

A função ft:mark coloca um elemento marcador ao redor dos \$nodes resultantes de uma requisição de índice Full Text. Exemplo: Verifica se um nó texto do banco de dados contém o valor "Hello World" e retorna <XML><mark>hello</mark> world</XML>.

```
A função ft:mark(db:open('DB')//*[text() contains text 'hello'])
```

A função ft:extract faz a extração e o retorno de partes relevantes dos resultados Full Text. Ele coloca um elemento marcador envolvendo os \$nodes resultantes de uma requisição de um índice Full Text e corta seções irrelevantes do resultado. Sintaxe ft:extract(\$nodes as node()\*) as node()\*. Exemplo: Verifica se um nó texto do banco de dados contém o valor "Hello World" e retorna <XML>...<b>hello</b>...<XML>.

```
ft:extract(db:open('DB')//*[text() contains text 'hello'], 'b', 1)
```

A função ft:count retorna o número de ocorrências de uma busca por termos especificados na expressão Full Text. Sintaxe: ft:count(\$nodes as node()\*) .

A função `ft:score` retorna uma pontuação de valores (0.0 - 1.0) que foi anexada para itens especificados. Se o valor de retorno foi zero nenhuma pontuação foi anexada. Sintaxe: `ft:score($item as item(*) as xs:double*`. Exemplo: Verifica se contém o texto 'a' dentro da String dada "a" e retorna o valor `XS:DOUBLE`.

```
ft:score('a' contains text 'a')
```

A função `ft:tokens` retorna todos tokens Full Text armazenados no índice do banco de dados `$db`, juntamente com o número de suas ocorrências. Sintaxe `ft:tokens($db as xs:string) as element(value)*`. Exemplo: Encontra o número de ocorrências para uma simples entrada de índice específica, o predicado posicional acelera o retorno.

```
let $term := ft:tokenize($term)
return data((ft:tokens('db', $term)[. = $term])[1]/@count)
```

A função `ft:tokenize` cria tokens para *string* dada `$input` utilizando o padrão corrente de opções Full Text. Sintaxe `ft:tokenize($input as xs:string) as xs:string*`. Exemplo: Cria os tokens para cada uma das palavras `No` e `Doubt`.

```
ft:tokenize("No Doubt")
```

Bibliotecas disponíveis em adição ao padrão XQuery Functions: