



Armazenamento de Documentos XML



Vanessa Braganholo

XML em diferentes contextos

XML para
comunicação
entre
aplicações

XML na
gerência de
dados e
conteúdo

XML

XML para
comunicação
entre
componentes

...

Como armazenar?

1. Sistema de arquivos
2. Banco de Dados Relacional/Objeto Relacional/etc. com suporte a XML (habilitado a XML ou híbrido)
3. Banco de Dados Nativo



1 – Sistema de arquivos

- ▶ Vantagens

- ▶ Flexível

- ▶ Desvantagens

- ▶ Segurança
 - ▶ Impossibilidade de otimização de consultas (ausência de índices, etc.)



2 – Banco de Dados com suporte a XML

- ▶ Alternativas:

- ▶ Fazer mapeamento “na mão”
 - ▶ Genérico ou
 - ▶ Específico para uma DTD ou XML Schema
- ▶ Utilizar um banco de dados Habilitado a XML
 - ▶ SGBDs com extensões para transferir dados entre documentos XML e suas estruturas.
- ▶ Utilizar um banco de dados híbrido
 - ▶ SGBD relacional/objeto-relacional que possui suporte a armazenamento XML nativo



3 – Banco de Dados Nativo

- ▶ SGBDs que armazenam XML em sua forma nativa, geralmente como texto indexado ou como uma variante do DOM mapeado para uma estrutura proprietária.



Fazendo o mapeamento “na mão”...

O princípio...

- ▶ Proposta publicada em 1993 (ABITEBOUL; CLUET; MILO; 1993) intitulada “**Querying and Updating the file**” sugeria usar a tecnologia relacional para consultar arquivos textuais
 - ▶ Para isso, seria necessário armazenar tais arquivos em BDs relacionais
 - ▶ Ideia: explorar a **estrutura intrínseca** de arquivos tais como arquivos SGML, código fonte, etc., para armazená-los no BD



XML

- ▶ XML possui tal estrutura intrínseca, e portanto poderia se beneficiar das idéias lançadas em 93
- ▶ Várias propostas específicas para armazenamento de XML surgiram ao longo dos anos:
 - ▶ (FLORESCU; KOSSMANN, 1999)
 - ▶ (DEUTSCH; FERNANDEZ; SUCIU, 1999)
 - ▶ (SHANMUGASUNDARAM et al., 1999)
 - ▶ (LEE; CHU, 2000)
 - ▶ (CHEN; DAVIDSON; ZHENG, 2002, 2003)



Consultas

- ▶ Mas, não basta só armazenar os docs XML. É necessário também poder consultá-los. Propostas que exploram este problema são:
 - ▶ (SHANMUGASUNDARAM et al., 1999)
 - ▶ (MANOLESCU; FLORESCU; KOSSMANN, 2001)
 - ▶ (SHANMUGASUNDARAM et al., 2001)
 - ▶ (TATARINOV et al., 2002)
 - ▶ (DEHAAN et al., 2003)



Tipos de proposta

▶ Armazenamento:

- ▶ Técnicas que exploram a estrutura do XML (elementos, atributos, relação pai-filho)
 - ▶ (FLORESCU; KOSSMANN, 1999)
 - ▶ (DEHAAN et al., 2003)
 - ▶ (TATARINOV et al., 2002)
- ▶ Técnicas que exploram o esquema do doc. XML (DTD ou XML Schema)
 - ▶ (SHANMUGASUNDARAM et al., 1999)
- ▶ Técnicas que exploram alguma relação semântica entre os dados (ex. dependências funcionais)
 - ▶ (CHEN; DAVIDSON; ZHENG, 2002, 2003)
 - ▶ (LEE; CHU, 2000)



Técnicas que exploram a estrutura
do XML

Documentos a serem armazenados

(FLORESCU; KOSSMANN, 1999)

```
<person id=1, age=55>
  <name>Peter</name>
  <address>4711 Fruitdale Ave.</address>
  <child>
    <person id=3, age=22>
      <name>John</name>
      <address>5361 Columbia Ave.</address>
      <hobby>swimming</hobby>
      <hobby>cycling</hobby>
    </person>
  </child>
  <child>
    <person id=4, age=7>
      <name>David</name>
      <address>4711 Fruitdale Ave.</address>
    </person>
  </child>
</person>
```

```
<person id=2, age=38, child=4>
  <name>Mary</name>
  <address>4711 Fruitdale Ave.</address>
  <hobby>painting</hobby>
</person>
```

FLORESCU; KOSSMANN, 1999

Proposta *Edge* (Aresta)

- ▶ Armazenar todos os documentos em uma única tabela chamada *Edge*

- ▶ Edge(source, ordinal, name, flag, target)

Nome do elemento
ou atributo

Id que indica o
documento XML ao
qual aquele
elemento pertence

Numero para
preservar a ordem
entre os elementos
de um mesmo
documento

FLORESCU; KOSSMANN, 1999

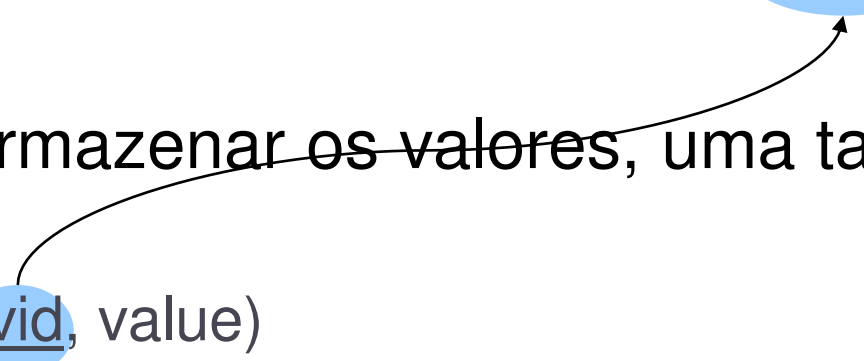
Proposta *Edge* (Aresta)

- ▶ Armazenar todos os documentos em uma única tabela chamada *Edge*
 - ▶ $\text{Edge}(\underline{\text{source}}, \underline{\text{ordinal}}, \text{name}, \text{flag}, \text{target})$
- ▶ Para armazenar os valores, uma tabela *V* para cada tipo:
 - ▶ $V_{\text{type}}(\underline{\text{vid}}, \text{value})$



FLORESCU; KOSSMANN, 1999

Proposta *Edge* (Aresta)

- ▶ Armazenar todos os documentos em uma única tabela chamada *Edge*
 - ▶ $\text{Edge}(\underline{\text{source}}, \underline{\text{ordinal}}, \text{name}, \text{flag}, \text{target})$
 - ▶ Para armazenar os valores, uma tabela *V* para cada tipo:
 - ▶ $V_{\text{type}}(\underline{\text{vid}}, \text{value})$
- 

FLORESCU; KOSSMANN, 1999

<i>Edge</i>				
<i>source</i>	<i>ordinal</i>	<i>name</i>	<i>flag</i>	<i>target</i>
1	1	age	int	<i>v1</i>
1	2	name	string	<i>v2</i>
1	3	address	string	<i>v3</i>
1	4	child	ref	3
1	5	child	ref	4
2	1	age	int	<i>v4</i>
...	

<i>V_{int}</i>		<i>V_{string}</i>	
<i>vid</i>	<i>value</i>	<i>vid</i>	<i>value</i>
v1	55	v2	Peter
v4	38	v3	4711 Fruitdale Ave.
v8	22	v5	Mary
v13	7	v6	4711 Fruitdale Ave.
		v7	painting
	
		v15	4711 Fruitdale Ave.

```

<person id=1, age=55>
  <name>Peter</name>
  <address>4711 Fruitdale Ave.</address>
  <child>
    <person id=3, age=22>
      <name>John</name>
      <address>5361 Columbia Ave.</address>
      <hobby>swimming</hobby>
      <hobby>cycling</hobby>
    </person>
  </child>
  <child>
    <person id=4, age=7>
      <name>David</name>
      <address>4711 Fruitdale Ave.</address>
    </person>
  </child>
</person>

<person id=2, age=38, child=4>
  <name>Mary</name>
  <address>4711 Fruitdale Ave.</address>
  <hobby>painting</hobby>
</person>

```

FLORESCU; KOSSMANN, 1999

Edge				
source	ordinal	name	flag	target
1	1	age	int	v1
1	2	name	string	v2
1	3	address	string	v3
1	4	child	ref	3
1	5	child	ref	4
2	1	age	int	v4
...

V_{int}		V_{string}	
vid	value	vid	value
v1	55	v2	Peter
v4	38	v3	4711 Fruitdale Ave.
v8	22	v5	Mary
v13	7	v6	4711 Fruitdale Ave.
		v7	painting
	
		v15	4711 Fruitdale Ave.

```

<person id=1, age=55>
  <name>Peter</name>
  <address>4711 Fruitdale Ave.</address>
  <child>
    <person id=3, age=22>
      <name>John</name>
      <address>5361 Columbia Ave.</address>
      <hobby>swimming</hobby>
      <hobby>cycling</hobby>
    </person>
  </child>
  <child>
    <person id=4, age=7>
      <name>David</name>
      <address>4711 Fruitdale Ave.</address>
    </person>
  </child>
</person>

<person id=2, age=38, child=4>
  <name>Mary</name>
  <address>4711 Fruitdale Ave.</address>
  <hobby>painting</hobby>
</person>
  
```

FLORESCU; KOSSMANN, 1999

Edge				
source	ordinal	name	flag	target
1	1	age	int	v1
1	2	name	string	v2
1	3	address	string	v3
1	4	child	ref	3
1	5	child	ref	4
2	1	age	int	v4
...

```

<person id=1, age=55>
  <name>Peter</name>
  <address>4711 Fruitdale Ave.</address>
  <child>
    <person id=3, age=22>
      <name>John</name>
      <address>5361 Columbia Ave.</address>
      <hobby>swimming</hobby>
      <hobby>cycling</hobby>
    </person>
  </child>
  <child>
    <person id=4, age=7>
      <name>David</name>
      <address>4711 Fruitdale Ave.</address>
    </person>
  </child>
</person>

<person id=2, age=38, child=4>
  <name>Mary</name>
  <address>4711 Fruitdale Ave.</address>
  <hobby>painting</hobby>
</person>
  
```

Elemento Complexo:
valor armazenado
na própria tabela
Edge

V _{int}		V _{string}	
vid	value	vid	value
v1	55	v2	Peter
v4	38	v3	4711 Fruitdale Ave.
v8	22	v5	Mary
v13	7	v6	4711 Fruitdale Ave.
		v7	painting
	
		v15	4711 Fruitdale Ave.

FLORESCU; KOSSMANN, 1999

- ▶ Propõem também várias variações, a mais usada é chamada de **inlining**...
- ▶ Armazenar tudo em uma única tabela
- ▶ Duas variações:
 - ▶ Uma coluna para cada tipo de valor
Edge(source, ordinal, name, v_{string} , v_{int} , ..., target)
 - ▶ Uma coluna única para todos os tipos de valores (todos os valores seriam convertidos para string)
Edge(source, ordinal, name, v, target)



FLORESCU; KOSSMANN, 1999

Processamento de consultas

► Reconstrução do documento:

Tabela Edge: junção com tabelas V, seleção pelo *source*, ordenar pelo *ordinal*

Tabela Inlinning: não há necessidade de junção, seleção pelo *source*, ordenar pelo *ordinal*



FLORESCU; KOSSMANN, 1999

Processamento de consultas

- ▶ Consultas com seleção (ex. selecionar todos os elementos `hobby="swimming"`)

Tabela Edge: junção com tabelas *V*, seleção pelo *source* e por *value*="swimming"

Tabela Inlinning: não há necessidade de junção, seleção pelo *source* e por *value*="swimming"



Dynamic Interval

(DEHAAN et al., 2003)

- Relação muito simples que guarda o elemento e a codificação do intervalo que o engloba

```
<site>
<people>
  <person id="person0">
    <name>Jaak Tempesti</name>
    <emailaddress>mailto:Tempesti@labs.com</emailaddress>
    <phone>+0 (873) 14873867</phone>
    <homepage>http://www.labs.com/~Tempesti</homepage>
  </person>
  <person id="person1">
    <name>Cong Rosca</name>
    <emailaddress>mailto:Rosca@washington.edu</emailaddress>
    <phone>+0 (64) 27711230</phone>
    <homepage>http://www.washington.edu/~Rosca</homepage>
  </person>
  ...
</people>
<closed_auctions>
  <closed_auction>
    <seller person="person0" />
    <buyer person="person1" />
    <itemref item="item1" />
    <price>42.12</price>
    <date>08/22/1999</date>
    <quantity>1</quantity>
    <type>Regular</type>
  </closed_auction>
  ...
</closed_auctions>
...
</site>
```

Dynamic Interval

(DEHAAN et al., 2003)

```
0<site>
  <people>
    <person id="person0">
      <name>Jaak Tempesti</name>
      <emailaddress>mailto:Tempesti@labs.com</emailaddress>
      <phone>+0 (873) 14873867</phone>
      <homepage>http://www.labs.com/~Tempesti</homepage>
    </person>
    <person id="person1">
      <name>Cong Rosca</name>
      <emailaddress>mailto:Rosca@washington.edu</emailaddress>
      <phone>+0 (64) 27711230</phone>
      <homepage>http://www.washington.edu/~Rosca</homepage>
    </person>
    ...
  </people>
  <closed_auctions>
    <closed_auction>
      <seller person="person0" />
      <buyer person="person1" />
      <itemref item="item1" />
      <price>42.12</price>
      <date>08/22/1999</date>
      <quantity>1</quantity>
      <type>Regular</type>
    </closed_auction>
    ...
  </closed_auctions>
  ...
85</site>
```

s	l	r
<site>	0	85
<people>	1	46
<person>	2	23
@id	3	6
person0	4	5
<name>	7	10
Jaak Tempesti	8	9
.	.	.
.	.	.
.	.	.

Dynamic Interval

(DEHAAN et al., 2003)

- ▶ Tradução das consultas – operações matemáticas sobre os intervalos

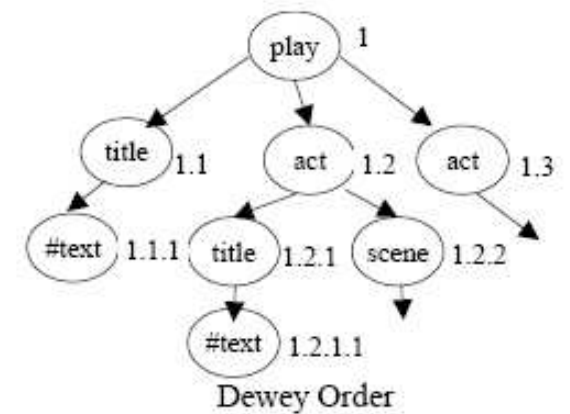
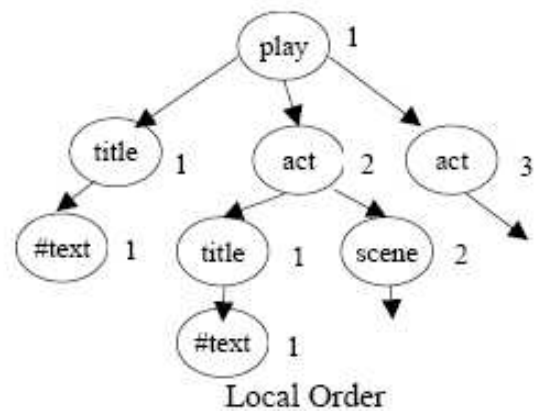
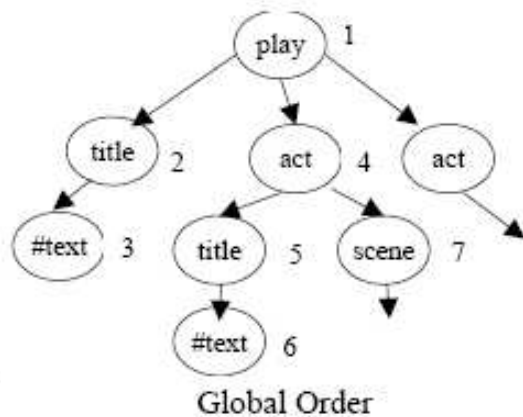
```
CREATE VIEW T_XNODE_item AS
  SELECT s, l+i*92 AS l, r+i*92 AS r
  FROM   I,
        ( SELECT '<item>' AS s, 0 AS l, 91 AS r
          FROM   UNIT
        UNION ALL
          SELECT s, l+1 AS l, r+1 AS r
          FROM
            ( SELECT s, l-i*90 AS l, r-i*90 AS r
              FROM   T_e
              WHERE  i*90<=l AND r<(i+1)*90
            )
        )
  )
```



Opções para armazenar ordem

(TATARINOV et al., 2002)

- ▶ Global Order
- ▶ Local Order (irmãos)
- ▶ Dewey Order

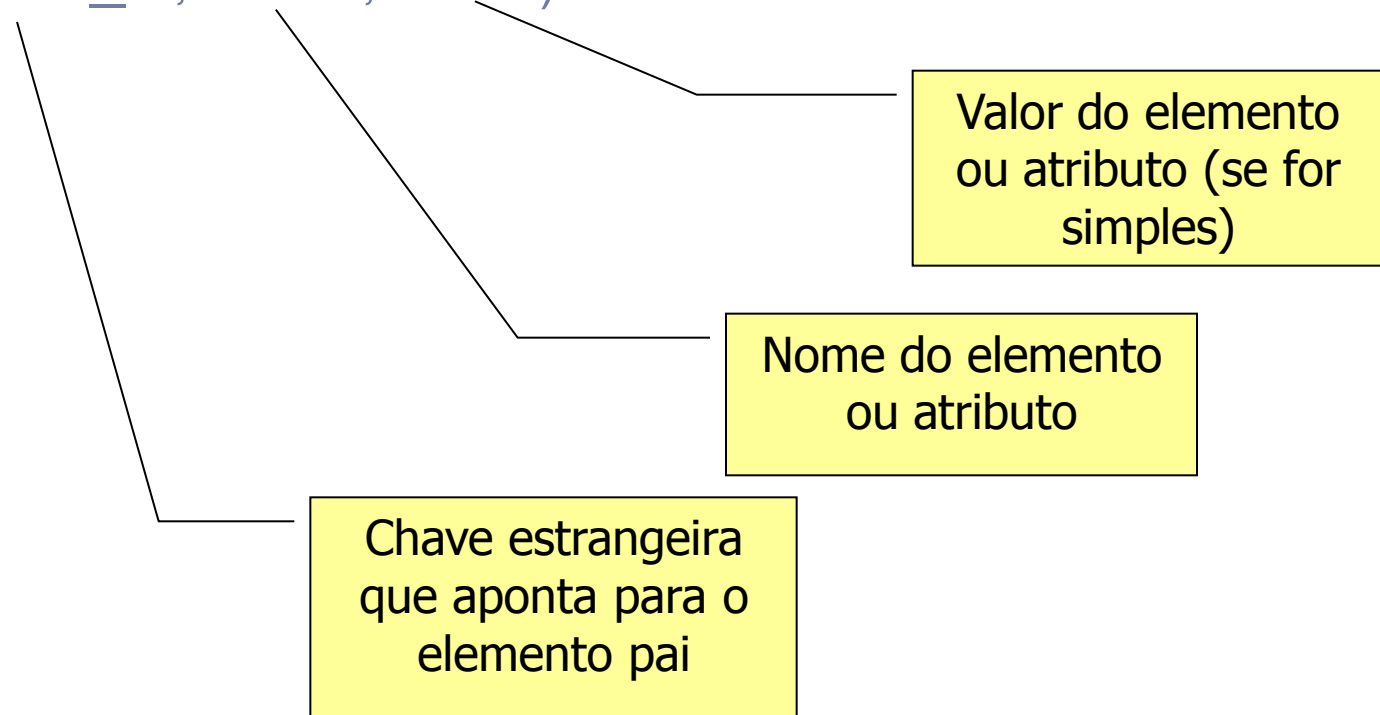


Proposta de Armazenamento

(TATARINOV et al., 2002)

- Variação do esquema Edge proposto por FLORESCU

Edge(id, parent_id, name, value)



Proposta de Armazenamento

(TATARINOV et al., 2002)

- ▶ Variação do esquema Edge proposto por FLORESCU

Edge(id, parent_id, name, value)

Ao invés do nome, o caminho do nodo pode ser armazenado (ex. /play/act ao invés de act)

Para poupar espaço, uma tabela Path pode ser usada

Path(path_id, path)



Mas ainda falta a ordem...

(TATARINOV et al., 2002)

- ▶ **Global Order:**

Edge(id, parent_id, end_desc_id, path_id, value)

Id – é o Global Order do nodo

end_desc_id – id do último descendente do nodo

- ▶ **Local Order:**

Edge(id, parent_id, sIndex, path_id, value)

Id – um ID único (que não precisa seguir a ordem do doc.)

sIndex – Local Order do nodo

- ▶ **Dewey Order:**

Edge(dewey, path_id, value)



Atualização

(TATARINOV et al., 2002)

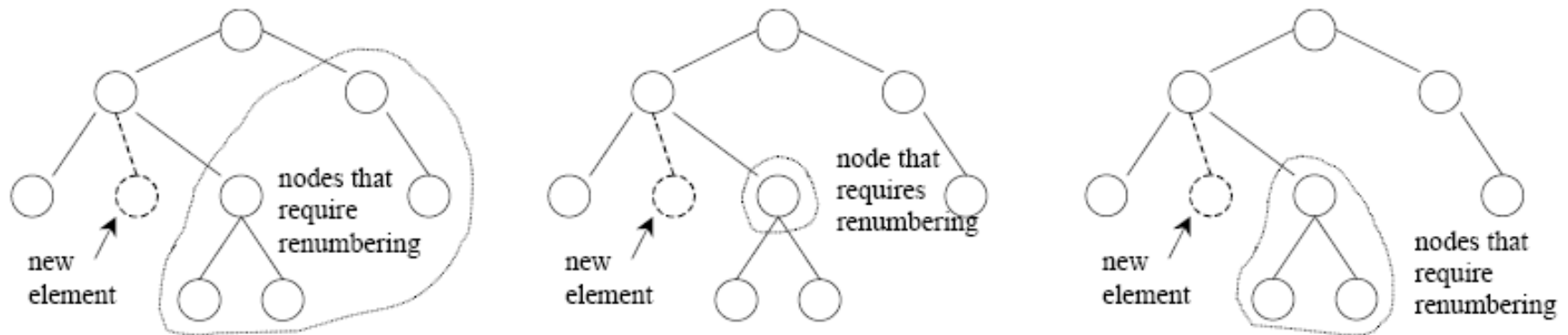


Figure 2. The worst case renumbering scenarios for Global, Local, and Dewey order encodings.

Consultas

(TATARINOV et al., 2002)

- ▶ Consultas suportadas: XPath



Técnicas que exploram o esquema
do doc. XML

SHANMUGASUNDARAM et al., 1999

```
<!ELEMENT book (booktitle, author)
<!ELEMENT article (title, author*, contactauthor)>
<!ELEMENT contactauthor EMPTY>
<!ATTLIST contactauthor authorID IDREF IMPLIED>
<!ELEMENT monograph (title, author, editor)>
<!ELEMENT editor (monograph*)>
<!ATTLIST editor name CDATA #REQUIRED>
<!ELEMENT author (name, address)>
<!ATTLIST author id ID #REQUIRED>
<!ELEMENT name (firstname?, lastname)>
<!ELEMENT firstname (#PCDATA)>
<!ELEMENT lastname (#PCDATA)>
<!ELEMENT address ANY>
```

```
<book>
  <booktitle> The Selfish Gene </booktitle>
  <author id = "dawkins">
    <name>
      <firstname> Richard </firstname>
      <lastname> Dawkins </lastname>
    </name>
    <address>
      <city> Timbuktu </city>
      <zip> 99999 </zip>
    </address>
  </author>
</book>
```



SHANMUGASUNDARAM et al., 1999

▶ Técnicas:

- ▶ Basic Inlining
- ▶ Shared Inlining
- ▶ Hybrid Inlining



Basic Inlining

SHANMUGASUNDARAM et al., 1999

► Basic Inlining

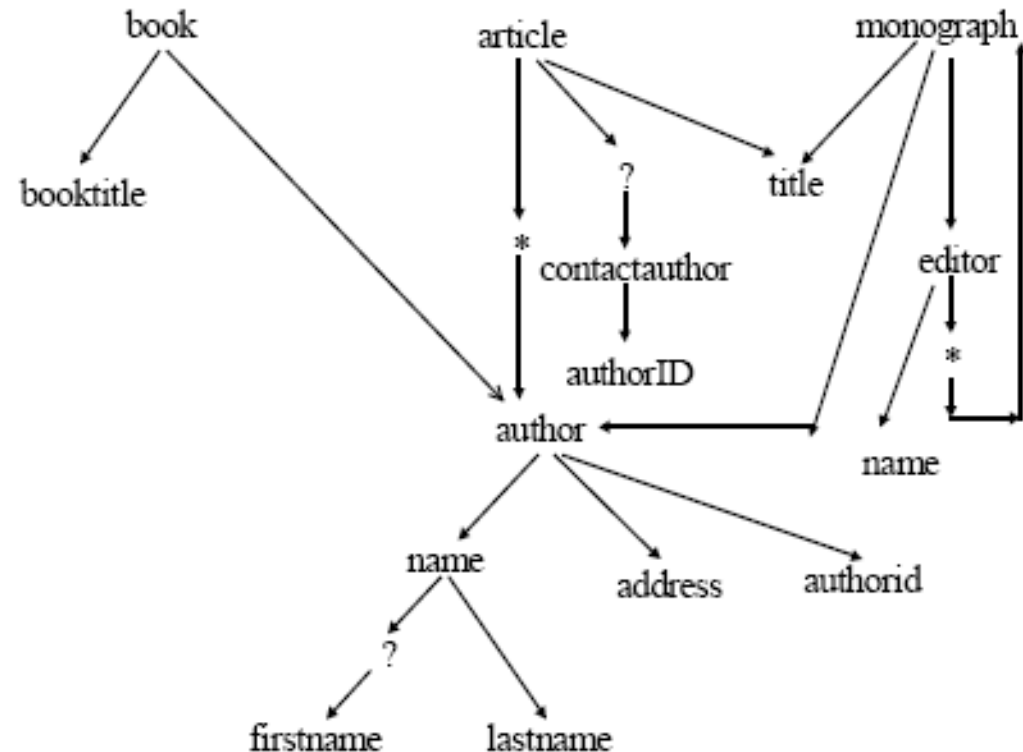
- Criar uma tabela para cada elemento da DTD, pq um documento XML pode usar como raiz qualquer um dos elementos de uma DTD (por isso declaramos a raiz em DOCTYPE)
- Para lidar com elementos que se repetem (*), um grafo é construído a partir da DTD



Basic Inlining

SHANMUGASUNDARAM et al., 1999

```
<!ELEMENT book (booktitle, author)
<!ELEMENT article (title, author*, contactauthor)>
<!ELEMENT contactauthor EMPTY>
<!ATTLIST contactauthor authorID IDREF IMPLIED>
<!ELEMENT monograph (title, author, editor)>
<!ELEMENT editor (monograph*)>
<!ATTLIST editor name CDATA #REQUIRED>
<!ELEMENT author (name, address)>
<!ATTLIST author id ID #REQUIRED>
<!ELEMENT name (firstname?, lastname)>
<!ELEMENT firstname (#PCDATA)>
<!ELEMENT lastname (#PCDATA)>
<!ELEMENT address ANY>
```



Basic Inlining

SHANMUGASUNDARAM et al., 1999

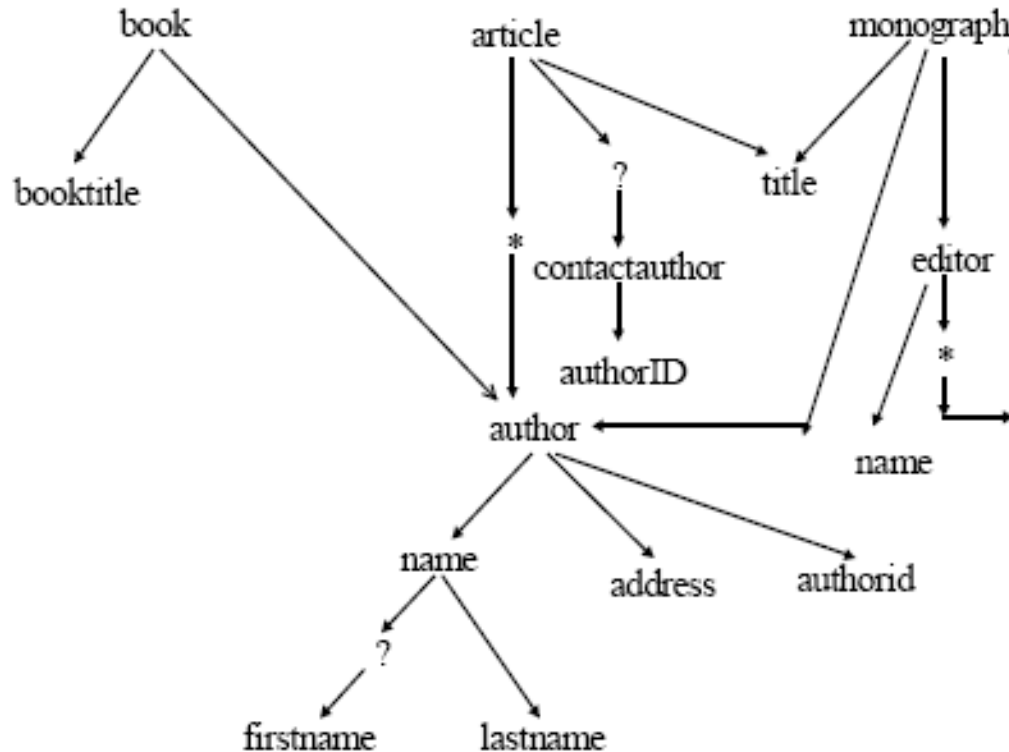
- ▶ O esquema para armazenar documentos que seguem uma DTD é a união dos conjuntos de relações criadas para cada elemento
- ▶ Para determinar o conjunto de relações necessário para armazenar um determinado elemento, construímos um grafo chamado “element graph”
 - ▶ Um elemento é escolhido para percorrer o grafo
 - ▶ Grafo vai sendo percorrido e cada elemento vai sendo marcado como visitado
 - ▶ Se o algoritmo tentar visitar um nodo já marcado, adicionar um “backpointer”
 - ▶ O resultado é uma árvore



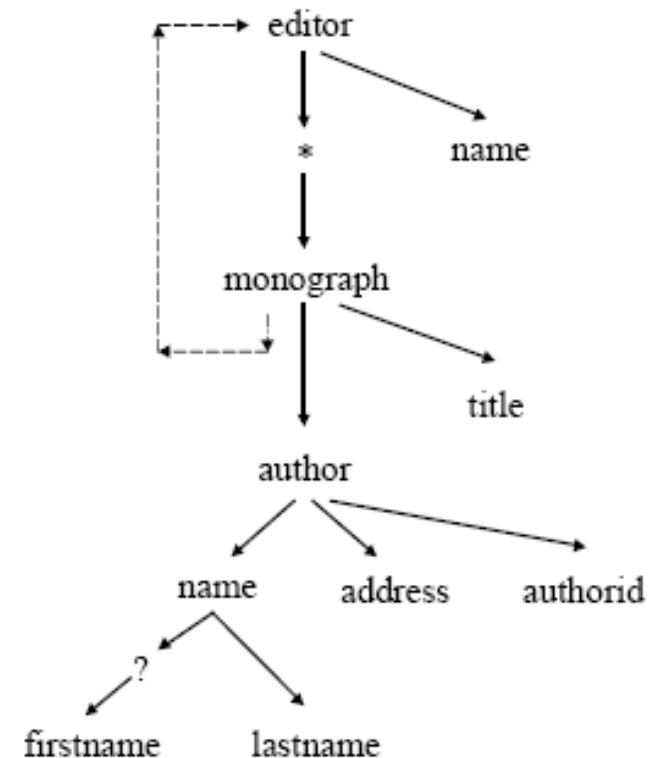
Basic Inlining

SHANMUGASUNDARAM et al., 1999

Grafo



Element Graph para elemento editor



Basic Inlining

SHANMUGASUNDARAM et al., 1999

- ▶ Dado um *element graph*, as relações são criadas como segue:
 - ▶ Uma relação é criada para o elemento raiz do *element graph*
 - ▶ Todos os descendentes são aninhados dentro desta relação, exceto:
 - ▶ Filhos de * são acomodados em relações separadas
 - ▶ Todo nodo que tem um backpointer é armazenado em uma relação separada (nova relação para lidar com recursão)
 - ▶ Ligações são feitas por chave estrangeira



Resultado

SHANMUGASUNDARAM et al., 1999

book (bookID: integer, book.booktitle : string, book.author.name.firstname: string, book.author.name.lastname: string, book.author.address: string, author.authorid: string)

booktitle (booktitleID: integer, booktitle: string)

article (articleID: integer, article.contactauthor.authorid: string, article.title: string)

article.author (article.authorID: integer, article.author.parentID: integer, article.author.name.firstname: string, article.author.name.lastname: string, article.author.address: string, article.author.authorid: string)

contactauthor (contactauthorID: integer, contactauthor.authorid: string)

title (titleID: integer, title: string)

monograph (monographID: integer, monograph.parentID: integer, monograph.title: string, monograph.editor.name: string, monograph.author.name.firstname: string, monograph.author.name.lastname: string, monograph.author.address: string, monograph.author.authorid: string)

editor (editorID: integer, editor.parentID: integer, editor.name: string)

editor.monograph (editor.monographID: integer, editor.monograph.parentID: integer, editor.monograph.title: string, editor.monograph.author.name.firstname: string, editor.monograph.author.name.lastname: string, editor.monograph.author.address: string, editor.monograph.author.authorid: string)

author (authorID: integer, author.name.firstname: string, author.name.lastname: string, author.address: string, author.authorid: string)

name (nameID: integer, name.firstname: string, name.lastname: string)

firstname (firstnamedID: integer, firstname: string)

lastname (lastnamedID: integer, lastname: string)

address (addressID: integer, address: string)

Resultado

SHANMUGASUNDARAM et al.,

book (bookID: integer, book.booktitle : string, book.author.name.firstname: string, book.author.address: string, author.authorid: string)

booktitle (booktitleID: integer, booktitle: string)

article (articleID: integer, article.contactauthor.authorid: string, article.title: string)

article.author (article.authorID: integer, article.author.parentID: integer, article.author.name.firstname: string, article.author.name.lastname: string, article.author.address: string, article.author.authorid: string)

contactauthor (contactauthorID: integer, contactauthor.authorid: string, contactauthor.title: string)

title (titleID: integer, title: string)

monograph (monographID: integer, monograph.parentID: integer, monograph.title: string, monograph.editor.name: string, monograph.author.name.firstname: string, monograph.author.name.lastname: string, monograph.author.address: string, monograph.author.authorid: string)

editor (editorID: integer, editor.parentID: integer, editor.name: string)

editor.monograph (editor.monographID: integer, editor.monograph.parentID: integer, editor.monograph.title: string, editor.monograph.author.name.firstname: string, editor.monograph.author.name.lastname: string, editor.monograph.author.address: string, editor.monograph.author.authorid: string)

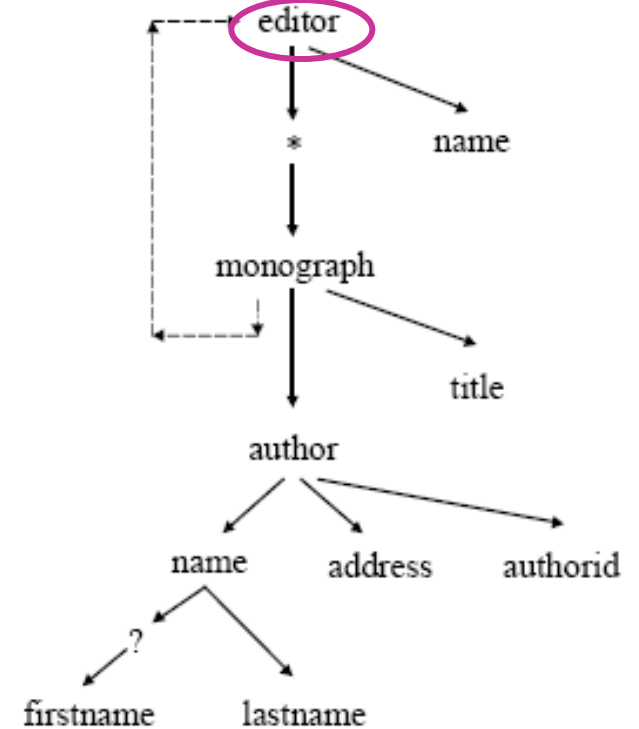
author (authorID: integer, author.name.firstname: string, author.name.lastname: string, author.address: string, author.authorid: string)

name (nameID: integer, name.firstname: string, name.lastname: string)

firstname (firstnamedID: integer, firstname: string)

lastname (lastnamedID: integer, lastname: string)

address (addressID: integer, address: string)



Resultado

SHANMUGASUNDARAM et al.,

book (bookID: integer, book.booktitle : string, book.author.name.firstname: string, book.author.address: string, author.authorid: string)

booktitle (booktitleID: integer, booktitle: string)

article (articleID: integer, article.contactauthor.authorid: string, article.t

article.author (article.authorID: integer, article.author.parentID: integer, article.author.name.firstname: string, article.author.name.lastname: string, article.author.address: string, article.author.authorid: string)

contactauthor (contactauthorID: integer, contactauthor.authorid: string, contactauthor.name.firstname: string, contactauthor.name.lastname: string, contactauthor.address: string, contactauthor.authorid: string)

title (titleID: integer, title: string)

monograph (monographID: integer, monograph.parentID: integer, monograph.title: string, monograph.editor.name: string, monograph.author.name.firstname: string, monograph.author.name.lastname: string, monograph.author.address: string, monograph.author.authorid: string)

editor (editorID: integer, editor.parentID: integer, editor.name: string)

editor.monograph (editor.monographID: integer, editor.monograph.parentID: integer, editor.monograph.title: string, editor.monograph.author.name.firstname: string, editor.monograph.author.name.lastname: string, editor.monograph.author.address: string, editor.monograph.author.authorid: string)

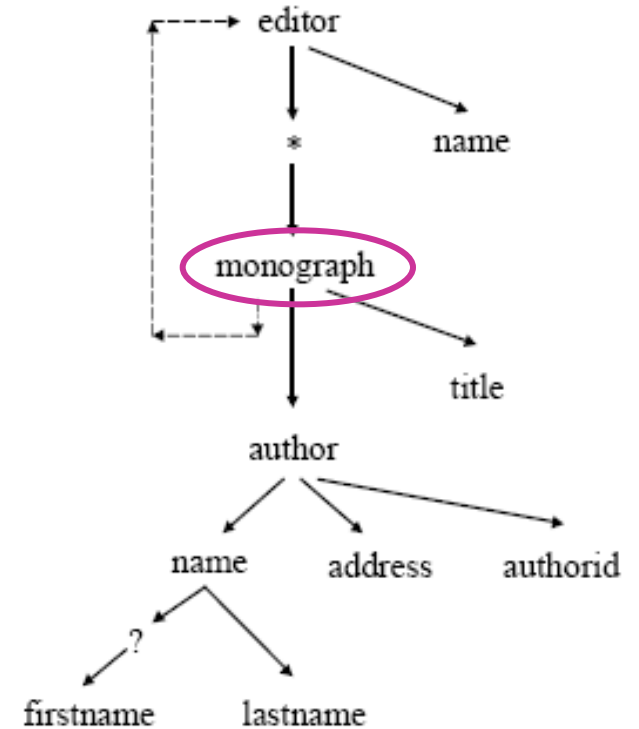
author (authorID: integer, author.name.firstname: string, author.name.lastname: string, author.address: string, author.authorid: string)

name (nameID: integer, name.firstname: string, name.lastname: string)

firstname (firstnamedID: integer, firstname: string)

lastname (lastnamedID: integer, lastname: string)

address (addressID: integer, address: string)



Resultado

SHANMUGASUNDARAM et al.,

book (bookID: integer, book.booktitle : string, book.author.name.firstname: string, book.author.address: string, author.authorid: string)

booktitle (booktitleID: integer, booktitle: string)

article (articleID: integer, article.contactauthor.authorid: string, article.t

article.author (article.authorID: integer, article.author.parentID: integer, article.author.name.firstname: string, article.author.name.lastname: string, article.author.address: string, article.author.authorid: string)

contactauthor (contactauthorID: integer, contactauthor.authorid: string, contactauthor.name.firstname: string, contactauthor.name.lastname: string, contactauthor.address: string, contactauthor.authorid: string)

title (titleID: integer, title: string)

monograph (monographID: integer, monograph.parentID: integer, monograph.title: string, monograph.editor.name: string, monograph.author.name.firstname: string, monograph.author.name.lastname: string, monograph.author.address: string, monograph.author.authorid: string)

editor (editorID: integer, editor.parentID: integer, editor.name: string)

editor.monograph (editor.monographID: integer, editor.monograph.parentID: integer, editor.monograph.title: string, editor.monograph.author.name.firstname: string, editor.monograph.author.name.lastname: string, editor.monograph.author.address: string, editor.monograph.author.authorid: string)

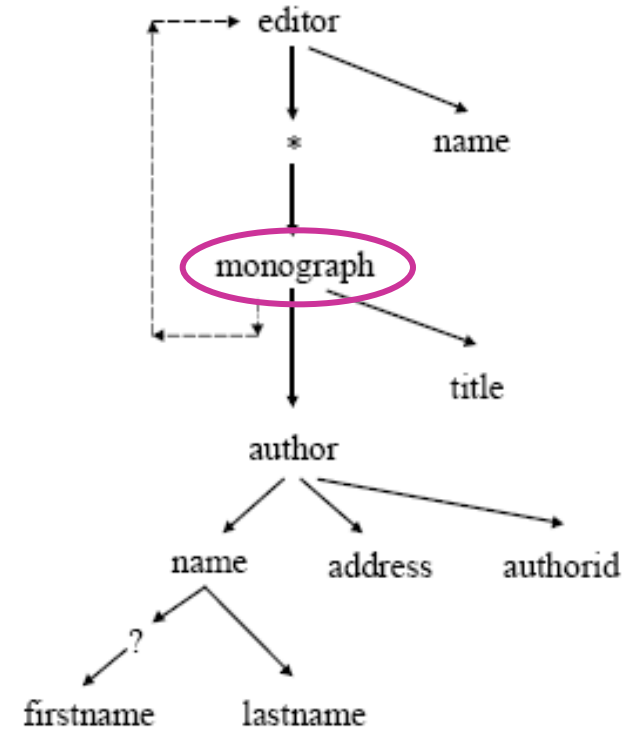
author (authorID: integer, author.name.firstname: string, author.name.lastname: string, author.address: string, author.authorid: string)

name (nameID: integer, name.firstname: string, name.lastname: string)

firstname (firstnamedID: integer, firstname: string)

lastname (lastnamedID: integer, lastname: string)

address (addressID: integer, address: string)



Basic Inlining

SHANMUGASUNDARAM et al., 1999

- ▶ Eficiente para consultas do tipo: me dê todos os autores dos livros
- ▶ Provavelmente muito ineficiente para outros tipos de consulta
 - ▶ Ex.: Liste todos os autores cujo primeiro nome é Jack
 - ▶ União de 5 consultas separadas
- ▶ Pensem nas atualizações! Redundância de informações! Péssima prática de modelagem!



Resultado

SHANMUGASUNDARAM et al., 1999

book (bookID: integer, book.booktitle : string, book.author.name.firstname: string, book.author.name.lastname: string, book.author.address: string, author.authorid: string)

booktitle (booktitleID: integer, booktitle: string)

article (articleID: integer, article.contactauthor.authorid: string, article.title: string)

article.author (article.authorID: integer, article.author.parentID: integer, article.author.name.firstname: string, article.author.name.lastname: string, article.author.address: string, article.author.authorid: string)

contactauthor (contactauthorID: integer, contactauthor.authorid: string)

title (titleID: integer, title: string)

monograph (monographID: integer, monograph.parentID: integer, monograph.title: string, monograph.editor.name: string, monograph.author.name.firstname: string, monograph.author.name.lastname: string, monograph.author.address: string, monograph.author.authorid: string)

editor (editorID: integer, editor.parentID: integer, editor.name: string)

editor.monograph (editor.monographID: integer, editor.monograph.parentID: integer, editor.monograph.title: string, editor.monograph.author.name.firstname: string, editor.monograph.author.name.lastname: string, editor.monograph.author.address: string, editor.monograph.author.authorid: string)

author (authorID: integer, author.name.firstname: string, author.name.lastname: string, author.address: string, author.authorid: string)

name (nameID: integer, name.firstname: string, name.lastname: string)

firstname (firstnamedID: integer, firstname: string)

lastname (lastnamedID: integer, lastname: string)

address (addressID: integer, address: string)

Shared Inlining

SHANMUGASUNDARAM et al., 1999

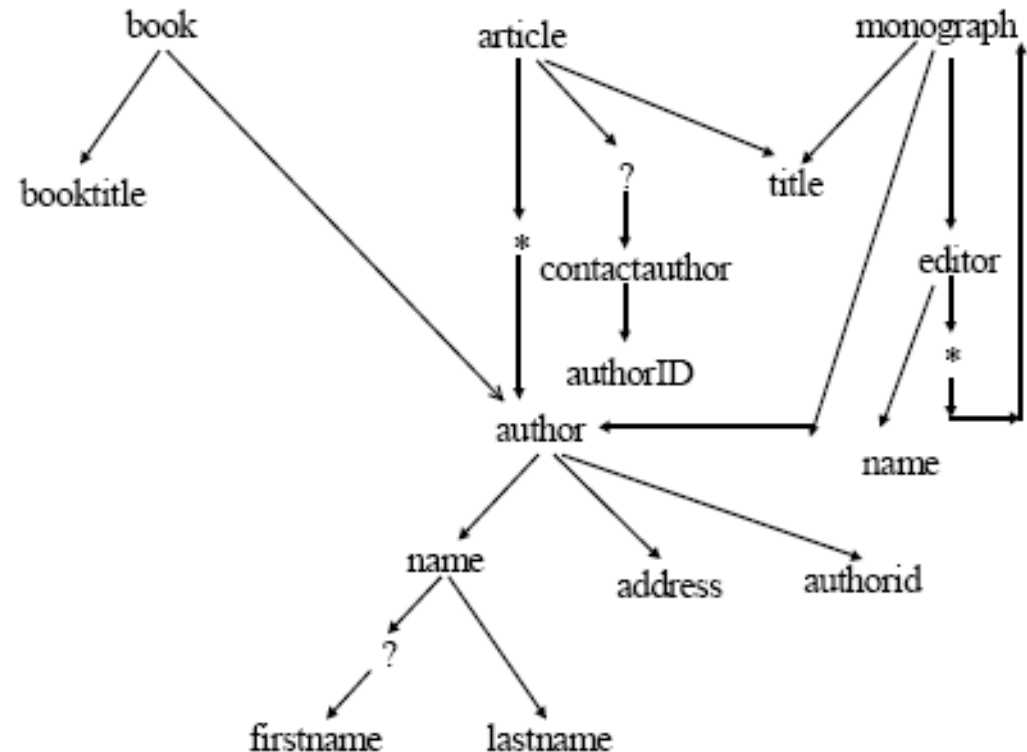
- ▶ Garante que cada elemento é representado em apenas uma tabela
- ▶ Para isso: identificar que nodos são representados várias vezes



Shared Inlining

SHANMUGASUNDARAM et al., 1999

- ▶ Usar o grafo:
 - ▶ Nós que têm mais de uma aresta entrando são transformados em relações próprias
 - ▶ Restantes são colocados dentro de tabelas já existentes

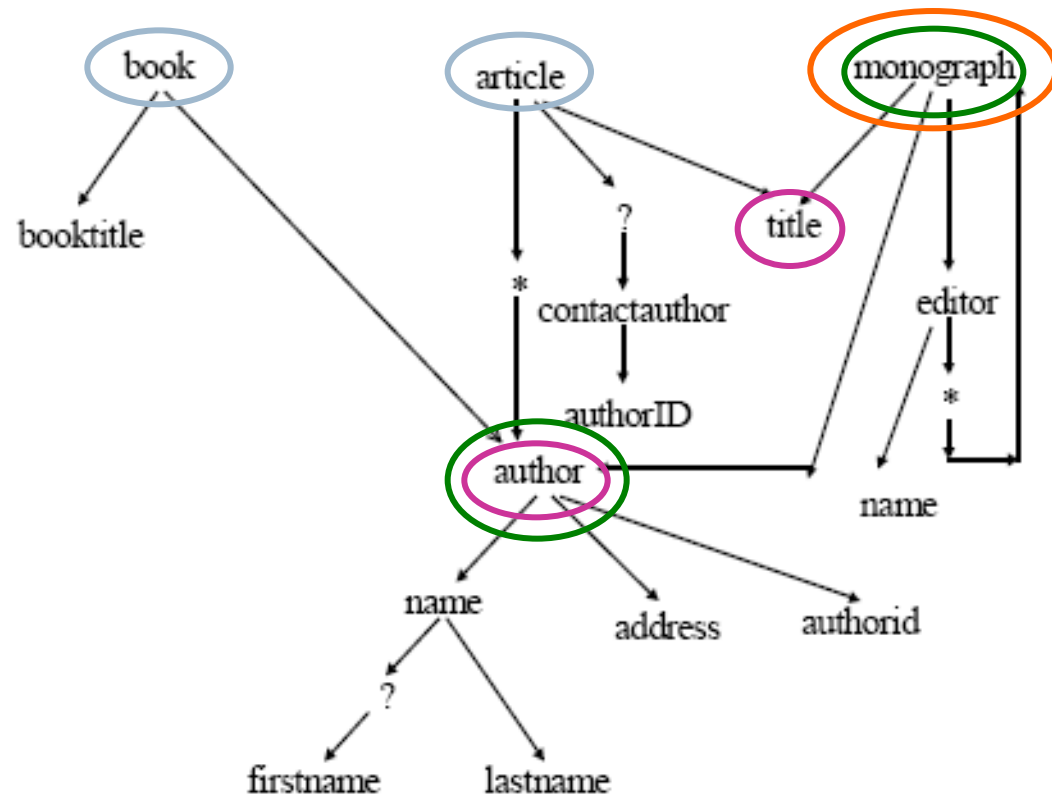


Shared Inlining

SHANMUGASUNDARAM et al., 1999

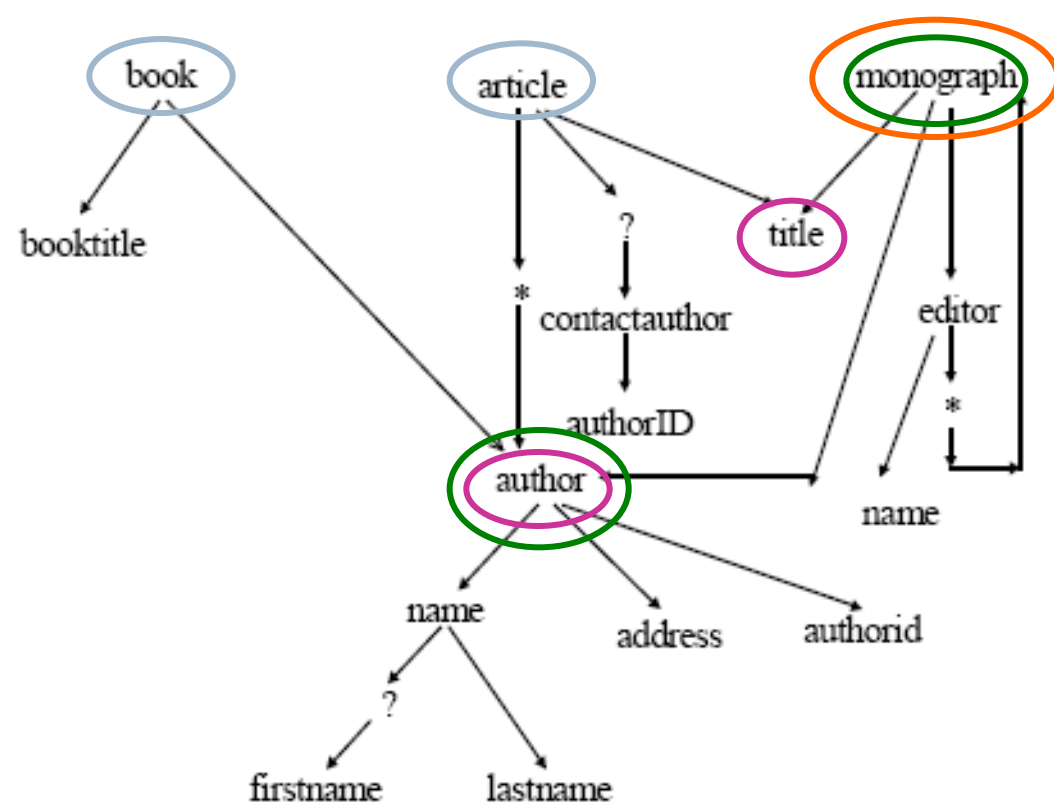
Usar o grafo:

- ▶ Relações próprias:
 - ▶ Nodos que têm **mais de uma aresta entrando**
 - ▶ Nodos com **zero arestas entrando** (pois eles não são acessíveis de nenhum outro nodo)
 - ▶ Elementos **depois de ***
 - ▶ Elementos **mutuamente recursivos** (elementos fortemente conectados – editor e monograph) – um deles é transformado em relação única
- ▶ Restantes são colocados dentro de tabelas já existentes



Shared Inlining

Resultado



book (bookID: integer, book.booktitle.isroot: boolean, book.booktitle : string)

article (articleID: integer, article.contactauthor.isroot: boolean, article.contactauthor.authorid: string)

monograph (monographID: integer, monograph.parentID: integer, monograph.parentCODE: integer, monograph.editor.isroot: boolean, monograph.editor.name: string)

title (titleID: integer, title.parentID: integer, title.parentCODE: integer, title: string)

author (authorID: integer, author.parentID: integer, author.parentCODE: integer, author.name.isroot: boolean, author.name.firstname.isroot: :boolean, author.name.firstname: string, author.name.lastname.isroot: boolean, author.name.lastname: string, author.address.isroot: boolean, author.address: string, author.authorid: string)

Shared Inlining

SHANMUGASUNDARAM et al., 1999

▶ Considerações

- ▶ Uniões não são mais necessárias
- ▶ No entanto, dependendo da consulta, junções são necessárias
- ▶ Abordagem que tenta balancear as vantagens das técnicas Basic e Shared: Hybrid

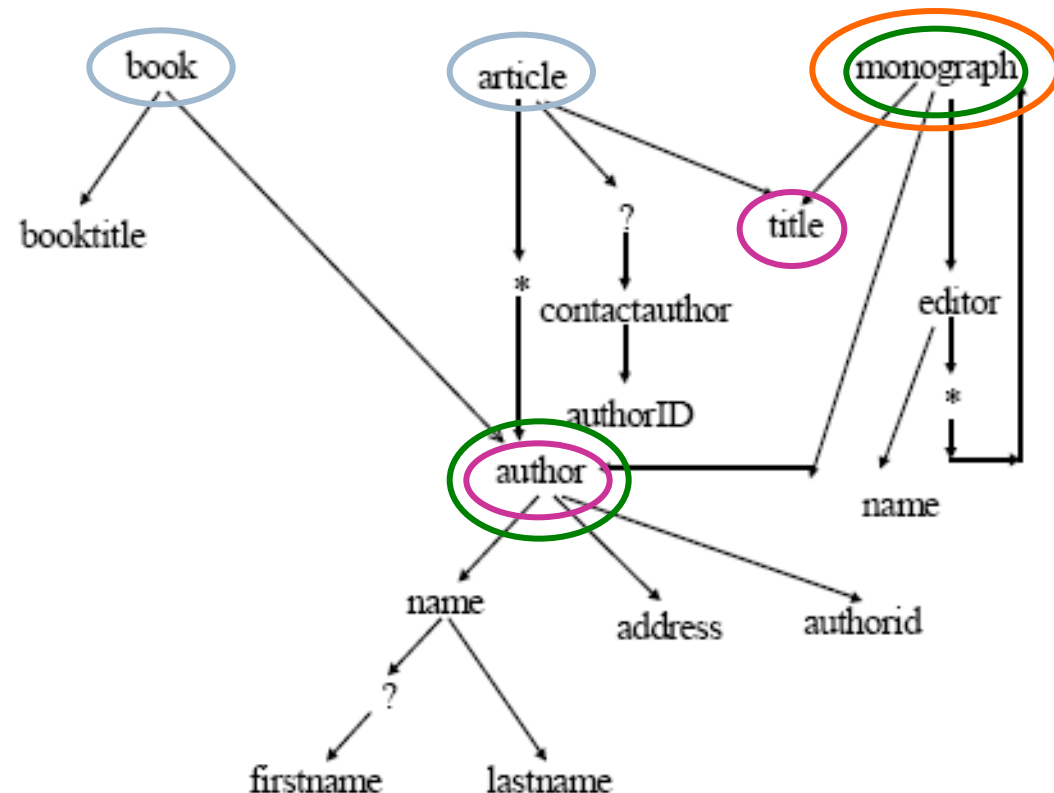


Hybrid Inlining

SHANMUGASUNDARAM et al., 1999

► Mesmo que Shared:

- Exceção: não cria relação separada para elementos que tem *in-degree* maior do que 1 e que não são recursivos e que não são filhos de *

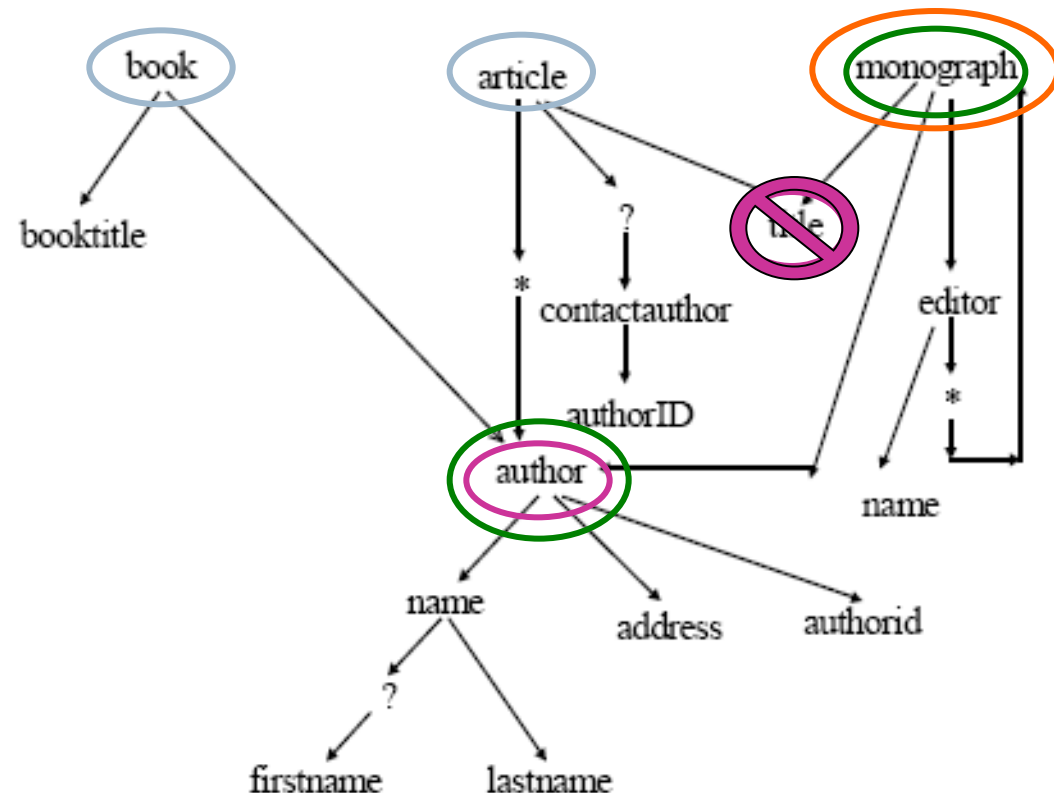


Hybrid Inlining

SHANMUGASUNDARAM et al., 1999

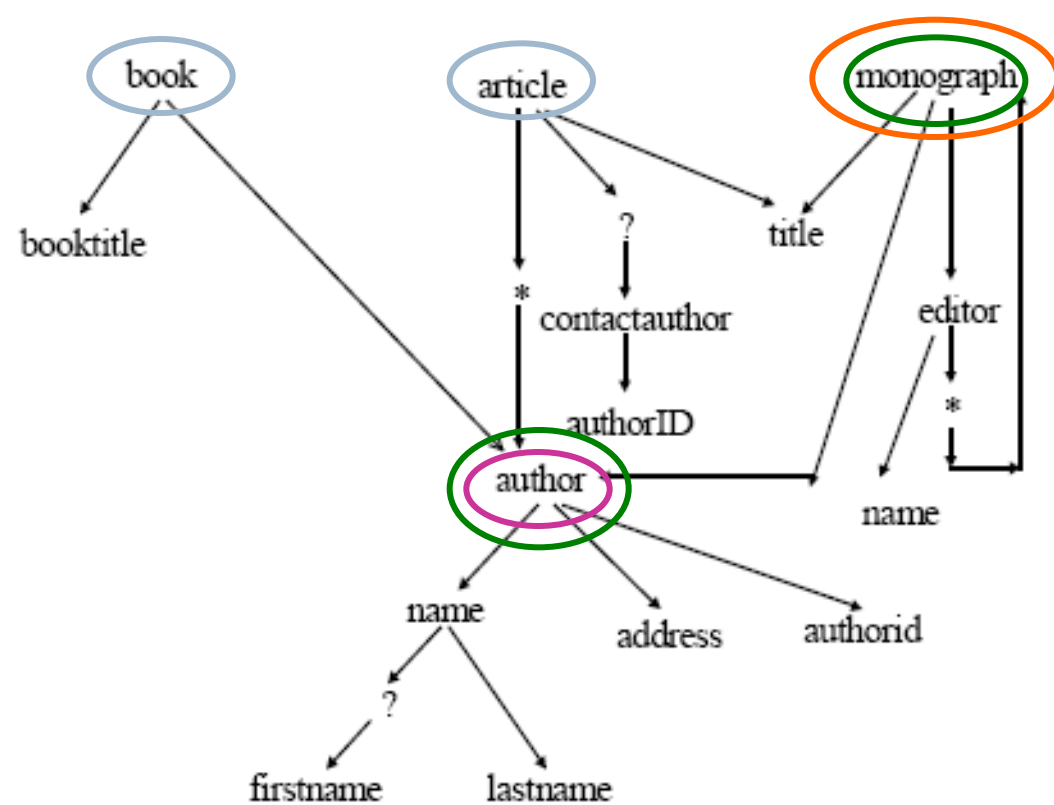
- ▶ Mesmo que Shared:

- ▶ Exceção: não cria relação separada para elementos que tem in-degree maior do que 1 e que não são recursivos e que não são filhos de *



Hybrid Inlining

Resultado



book (bookID: integer, book.booktitle.isroot: boolean, book.booktitle : string, author.name.firstname: string, author.name.lastname: string, author.address: string, author.authorid: string)

article (articleID: integer, article.contactauthor.isroot: boolean, article.contactauthor.authorid: string, article.title.isroot: boolean, article.title: string)

monograph (monographID: integer, monograph.parentID: integer, monograph.parentCODE: integer, monograph.title: string, monograph.editor.isroot: boolean, monograph.editor.name: string, author.name.firstname: string, author.name.lastname: string, author.address: string, author.authorid: string)

author (authorID: integer, author.parentID: integer, author.parentCODE: integer, author.name.isroot: boolean, author.name.firstname.isroot: boolean, author.name.firstname: string, author.name.lastname.isroot: boolean, author.name.lastname: string, author.address.isroot: boolean, author.address: string, author.authorid: string)

Referências

- ▶ CHEN, Y.; DAVIDSON, S. B.; ZHENG, Y. Constrain Preserving XML storage in Relations. In: INTERNATIONAL WORKSHOP ON THE WEB AND DATABASES, WEBDB, 2002, Madison, Wisconsin. Proceedings... [S.l.: s.n.], 2002. p.712.
 - ▶ CHEN, Y.; DAVIDSON, S. B.; ZHENG, Y. RRXS: redundancy reducing XML storage in relations. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, VLDB, 2003, Berlin, Germany. Proceedings... San Francisco:Morgan Kaufmann, 2003.
 - ▶ DEHAAN, D.; TOMAN, D.; CONSENS, M.; OZSU, M. T. A Comprehensive XQuery to SQL Translation using Dynamic Interval Encoding. In: INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, SIGMOD, 2003, San Diego, CA. Proceedings. . . [S.l.: s.n.], 2003.
 - ▶ DEUTSCH, A.; FERNANDEZ, M.; SUCIU, D. Storing semistructured data with STORED. In: INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, SIGMOD, 1999, Philadelphia, Pennsylvania. Proceedings... [S.l.: s.n.], 1999. p.431442.
-



Referências

- ▶ FLORESCU, D.; KOSSMANN, D. A performance evaluation of alternative mapping schemes for storing XML data in a relational database. France:INRIA, 1999. (Technical Report 3684).
- ▶ LEE, D.; CHU, W. W. Constraints-Preserving Transformation from XML Document Type Denition to Relational Schema. In: INTERNATIONAL CONFERENCE ON ENTITY RELATIONSHIP, ER, 2000, Salt Lake City, Utah, USA. Proceedings... [S.l.: s.n.], 2000. p.323338.
- ▶ MANOLESCU, I.; FLORESCU, D.; KOSSMANN, D. Pushing XML Queries inside Relational Databases. France: INRIA, 2001. (Technical Report 4112).
- ▶ SHANMUGASUNDARAM, J.; TUFTE, K.; ZHANG, C.; HE, G.; DEWITT, D. J.; NAUGHTON, J. F. Relational Databases for Querying XML Documents: limitations and opportunities. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, VLDB, 1999, Edinburgh, Scotland, UK. Proceedings... San Francisco: Morgan Kaufmann, 1999. p.302314.



Referências

- ▶ SHANMUGASUNDARAM, J.; SHEKITA, E.; KIERNAN, J.; KRISHNAMURTHY, R.; VIGLAS, E.; NAUGHTON, J.; TATARINOV, I. A general technique for querying XML documents using a relational database system. Sigmod Record, [S.l.], v.30, n.3, p.2026, Sept. 2001.
- ▶ TATARINOV, I.; VIGLAS, E.; BEYER, K.; SHANMUGASUNDARAM, J.; SHEKITA, E. Storing and Querying Ordered XML Using a Relational Database System. In: INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, SIGMOD, 2002, Madison, Wisconsin. Proceedings... [S.l.: s.n.], 2002.





SGBDs XML Nativos



Um SGBD nativo:

- ▶ Define um modelo lógico para um documento XML e armazena e recupera documentos de acordo com este modelo.
 - ▶ No mínimo, o modelo deve incluir elementos, atributos, PCDATA e a ordem dos elementos
 - ▶ Exemplos de modelos: modelo de dados do XPath data model, modelo DOM, etc.

Um SGBD nativo:

- ▶ Possui um documento como unidade fundamental de armazenamento lógico
 - ▶ Paralelo com tuplas dos bancos relacionais
- ▶ Não é exigido que exista nenhum modelo de armazenamento físico em particular
 - ▶ Exemplo: pode ser construído sobre um banco relacional, hierárquico, OO ou usar um formato proprietário qualquer (ex. arquivos compactados indexados)

Pontos principais

- ▶ Um SGBD nativo é um banco de dados especializado para armazenar dados XML
 - ▶ Armazena todos os componentes do modelo XML (elementos, atributos, etc.)
- ▶ *Documents go in and documents come out*
- ▶ Um SBGD nativo pode não ser um banco de dados isolado (pode ter um outro SGBD por “baixo dos panos”)

Alguns SGBDs nativos...

Berkley DB XML	Developer: Sleepycat Software URL: http://www.sleepycat.com/products/bdbxml.html Database type: Key-value
eXist	Developer: Wolfgang Meier URL: http://exist.sourceforge.net Database type: Proprietary
Tamino	Developer: Software AG (Germany), Consist (Brasil) URL: http://www.softwareag.com/tamino/ Database type: Proprietary. Relational data through ODBC
Timber	Developer: University of Michigan URL: http://www.eecs.umich.edu/db/timber Database type: Shore, Berkeley DB
XIndice	Developer: Apache Software Foundation URL: http://xml.apache.org/xindice Database type: Proprietary



Fonte:

<http://www.rpbourret.com/xml/XMLDatabaseProds.htm#native>
(tela capturada em Maio de 2012)

Product	Developer	License	DB Type
4Suite, 4Suite Server	FourThought	Open Source	Object-oriented
BaseX	University of Konstanz	Open Source	Proprietary
Berkeley DB XML	Oracle	Open Source	Key-value
DBDOM	K. Ari Krupnikov	Open Source	Relational
dbXML	dbXML Group	Open Source	Proprietary
Dieselpoint	Dieselpoint, Inc.	Commercial	None (indexes only)
DOMSafeXML	Ellipsis	Commercial	File system(?)
EMC Documentum xDB	X-Hive Corporation	Commercial	Proprietary. Relational through JDBC
eXist	Wolfgang Meier	Open Source	Proprietary
eXtc	M/Gateway Developments Ltd.	Free	Post-relational
Extraway	3D Informatica	Commercial	Files plus indexes
Infonyte DB	Infonyte	Commercial	Proprietary
Ipedo XML Database	Ipedo	Commercial	Proprietary
Lore	Stanford University	Research	Semi-structured
MarkLogic Server	Mark Logic Corp.	Commercial	Proprietary
M/DB:X	M/Gateway Developments Ltd.	Free	Hierarchical
MonetDB/XQuery	CWI Database Group	Open Source	Proprietary
myXMLDB	Mladen Adamovic	Open Source	MySQL
Natix	University of Mannheim	Free / non-commercial	Proprietary
ozone	ozone-db.org	Open Source	Object-oriented
Qizx	XMLMind	Commercial	Proprietary
Sedna XML DBMS	ISP RAS MODIS	Free	Proprietary
Sekaiju / Yggdrasill	Media Fusion	Commercial	Proprietary
SQL/XML-IMDB	QuiLogic	Commercial	Proprietary (native XML and relational)
Sonic XML Server	Sonic Software	Commercial	Object-oriented (ObjectStore). Relational and other data through Data Junction
Tamino	Software AG	Commercial	Proprietary. Relational through ODBC.
TeraText DBS	TeraText Solutions	Commercial	Proprietary
TEXTML Server	IXIASOFT, Inc.	Commercial	Proprietary
TigerLogic XDMS	Raining Data	Commercial	Pick
Timber	University of Michigan	Open Source (non-commercial only)	Shore, Berkeley DB
TOTAL XML	Cincom	Commercial	Object-relational?
Virtuoso	OpenLink Software	Commercial	Proprietary. Relational through ODBC
XediX TeraSolution	AM2 Systems	Commercial	Proprietary
Xindice	Apache Software Foundation	Open Source	Proprietary
xml.gax.com	GAX Technologies	Commercial	Proprietary
Xpiori XMS	Xpiori	Commercial	Proprietary
XQuantum XML Database Server	Cognetic Systems	Commercial	Proprietary
XStreamDB Native XML Database	Bluestream Database Software Corp.	Commercial	Proprietary
Xyleme Zone Server	Xyleme SA	Commercial	Proprietary

Características de SGBDs nativos

- ▶ Armazena documentos
- ▶ Gerenciam “coleções” de documentos
- ▶ Suportam consultas XPath e XQuery
- ▶ Atualizações: linguagens próprias ou XQuery (XQuery Update Facility - <http://www.w3.org/TR/xquery-update-10/>)



Podem oferecer o que os outros oferecem e mais?

- ▶ Suportar transações
- ▶ Acesso integrado a legados
- ▶ Suporte à distribuição
- ▶ Escalabilidade para grandes volumes
- ▶ Beneficiar-se da auto-descrição do XML
- ▶ Consultas complexas (mistas)
- ▶ Ter melhor desempenho que SGBDs padrão
- ▶ Usar XSL stylesheet para controlar o retorno do resultado
- ▶ Integração com outras aplicações de E-Business
- ▶ ...



Situação dos SGBDs Nativos

- ▶ Grande número de SGBDs nativos em oferta no mercado e na academia
- ▶ Armazenamento baseado em texto ou baseado em modelo proprietário
 - ▶ Ver discussão em <http://www.rpbouret.com/xml/XMLAndDatabases.htm#nativearchitecture>
- ▶ Lidam com grande volumes de documentos, alto *throughput*
- ▶ Grande eficiência das consultas
 - ▶ Consultas Full text



Alguns pontos que merecem consideração

- ▶ Nenhuma arquitetura padrão comum
- ▶ Nenhum padrão para benchmark de desempenho
 - ▶ Xmach – 1 (XML Data Management benchmark, September 2000, Timo Böhme, Erhard Rahm, University of Leipzig, Germany)
 - ▶ XBench (A Family of Benchmarks for XML DBMSs, Benjamin Bin Yao, M. Tamer Ozsü, and John Keenleyside)
 - ▶ XMark (XMark: A Benchmark for XML Data Management, Albrecht Schmidt, Florian Waas, Martin L. Kersten, Michael J. Carey, Ioana Manolescu, Ralph Busse. VLDB 2002: 974-985)





Sedna

Sedna

- ▶ SGBD XML nativo, Open Source
 - ▶ <http://www.modis.ispras.ru/sedna/>
- ▶ Implementado em C
- ▶ Forma de armazenamento
 - ▶ Lista encadeada de blocos que armazenam nós XML
 - ▶ Índices são implementados usando Árvores B+
- ▶ Consultas: XQuery e Path
- ▶ Atualizações: XQuery



Arquivos para Instalação

- ▶ SGBD:

- ▶ <http://modis.ispras.ru/sedna/download.html>
- ▶ Funciona via linha de comando

- ▶ Interface de Administração:

- ▶ Existem duas: uma oficial, outra da UFC
- ▶ Vamos usar a UFC, que funciona melhor
- ▶ <http://sednaadmin.great.ufc.br/>



Instalação

▶ SGBD

- ▶ Basta descompactar o arquivo
- ▶ Editar a variável de ambiente PATH, adicionando o diretório bin do Sedna

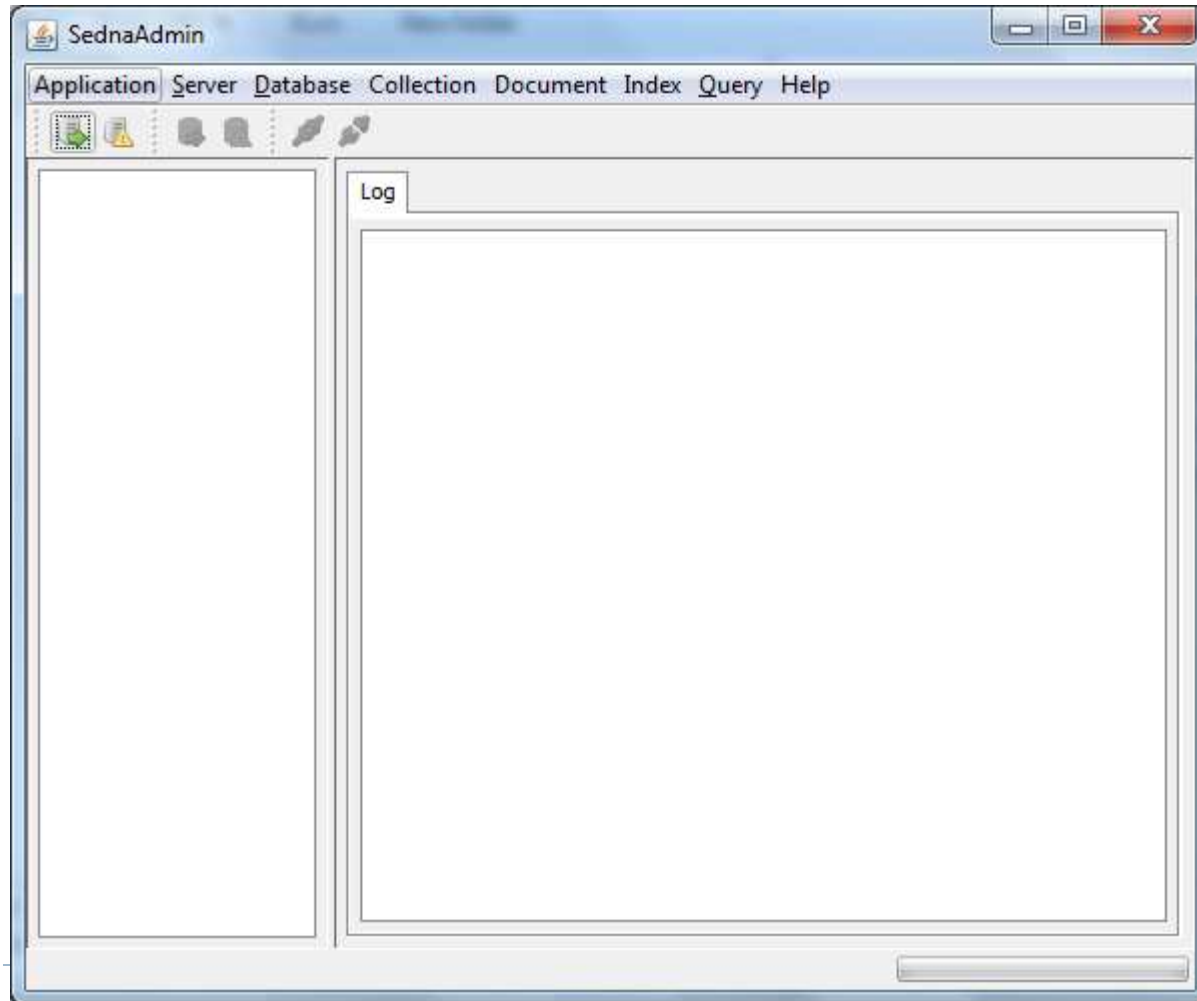
▶ Interface de Administração

- ▶ Basta descompactar o arquivo
- ▶ Informar qual o diretório raiz do Sedna



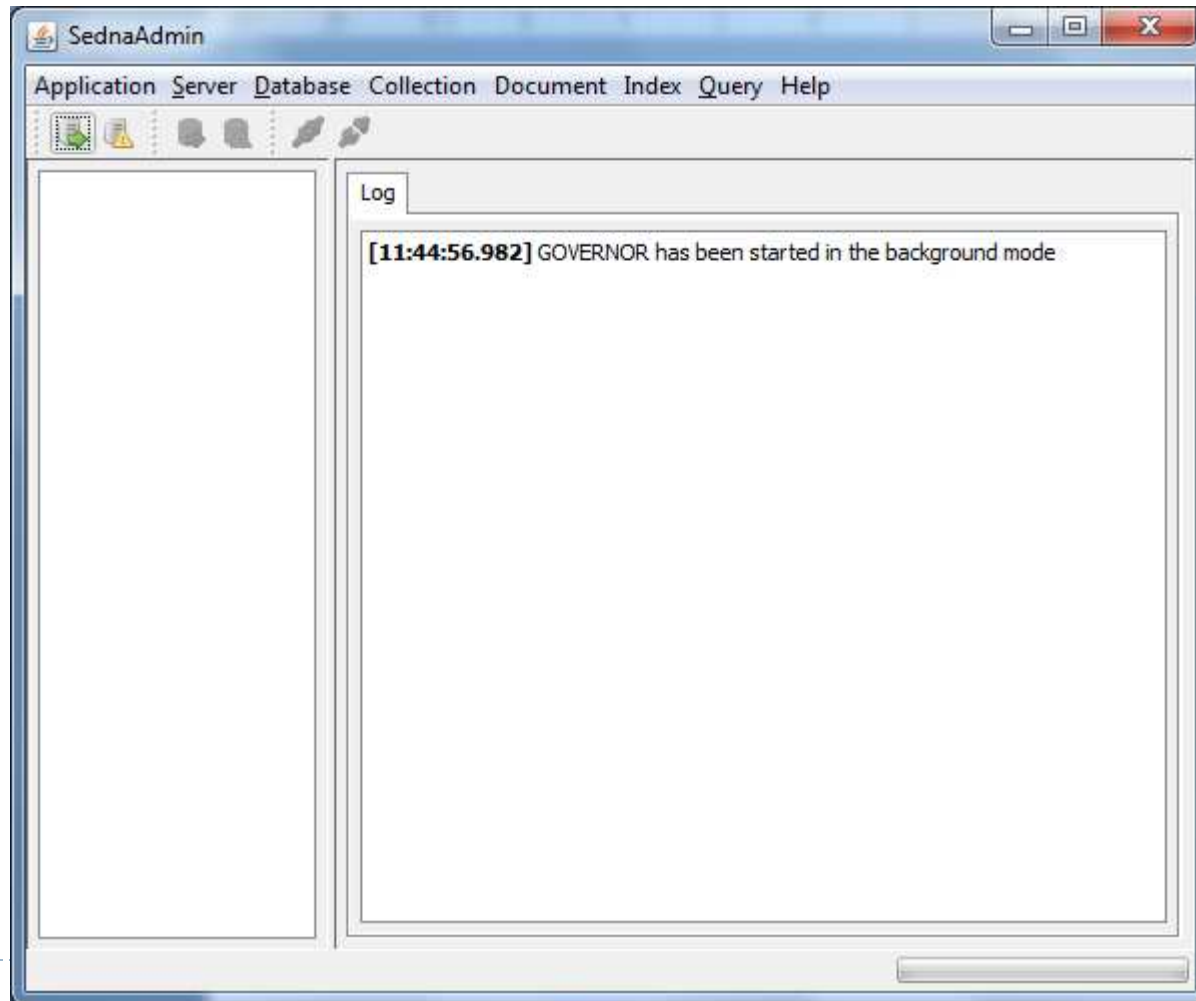
Para executar

- ▶ Iniciar a interface de Administração



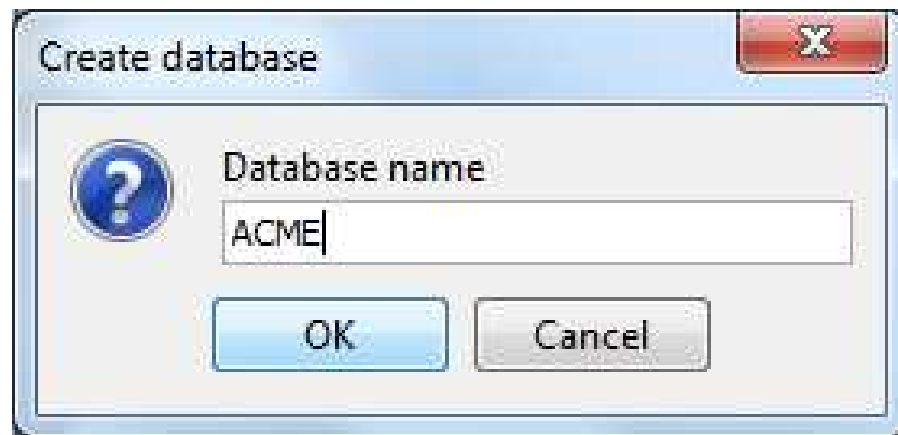
Iniciar o Servidor Sedna

► Menu Server/Start

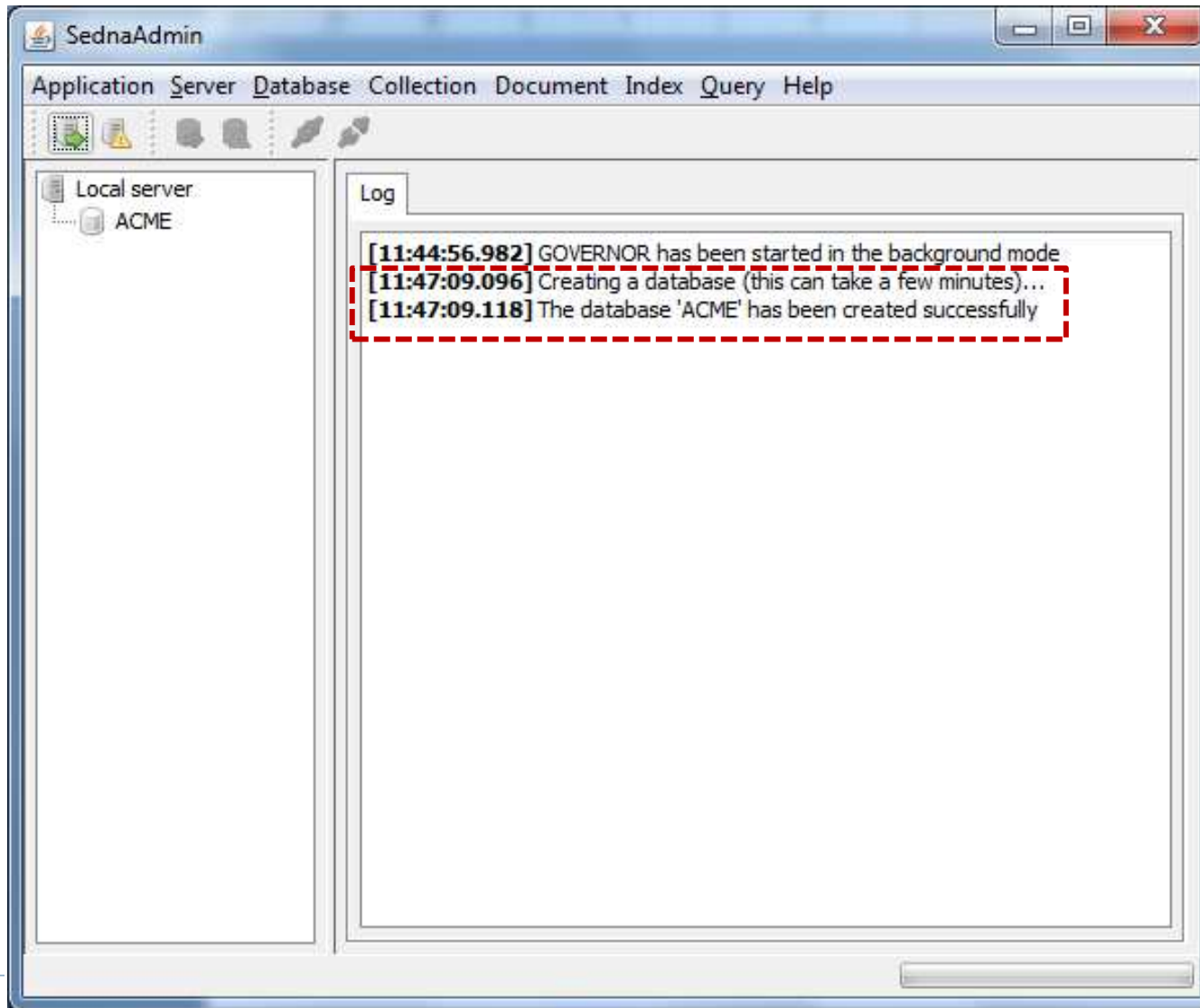


Criar um database

- ▶ Menu Database/Create
- ▶ Informar o nome do database: ACME

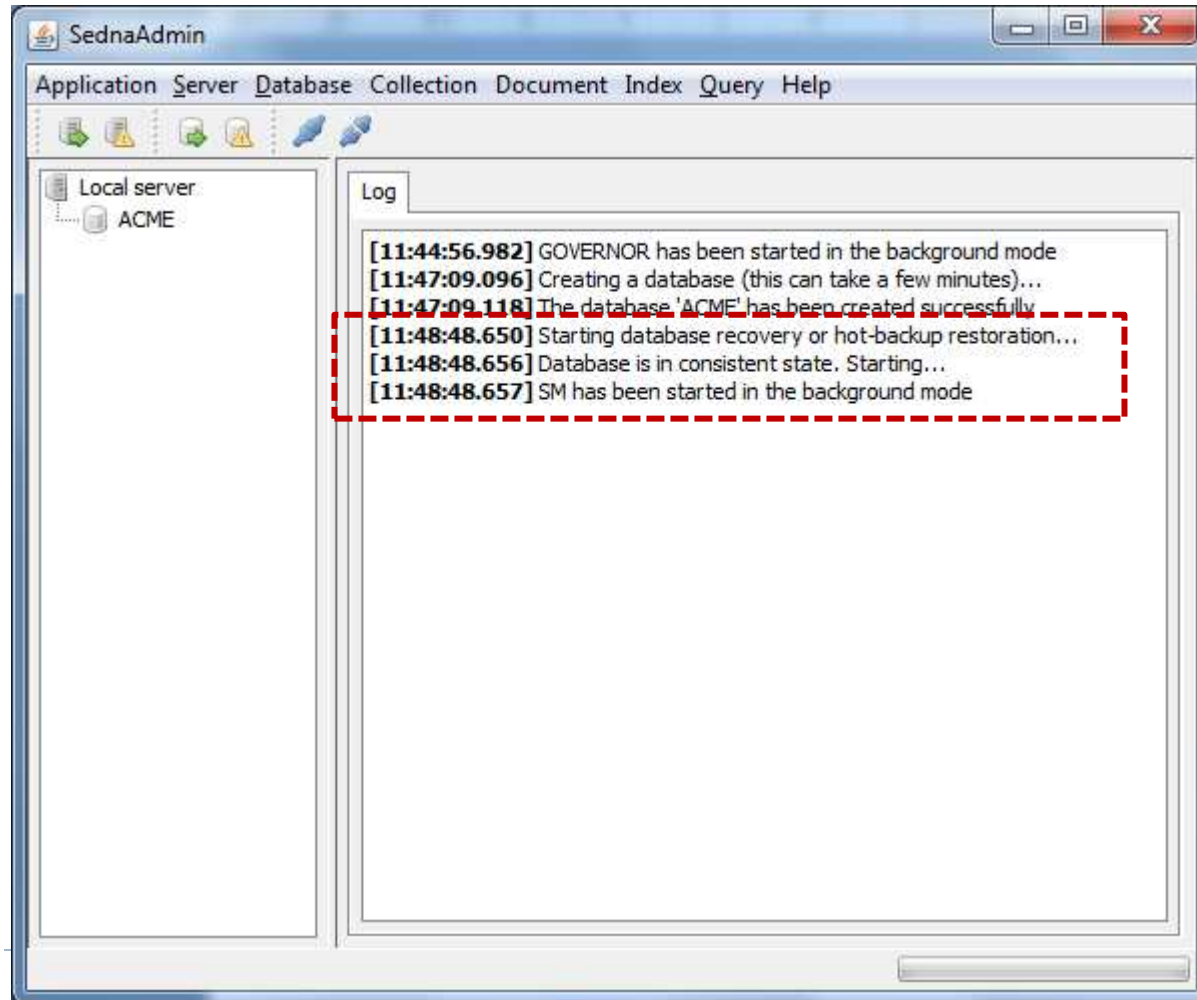


Criar um database



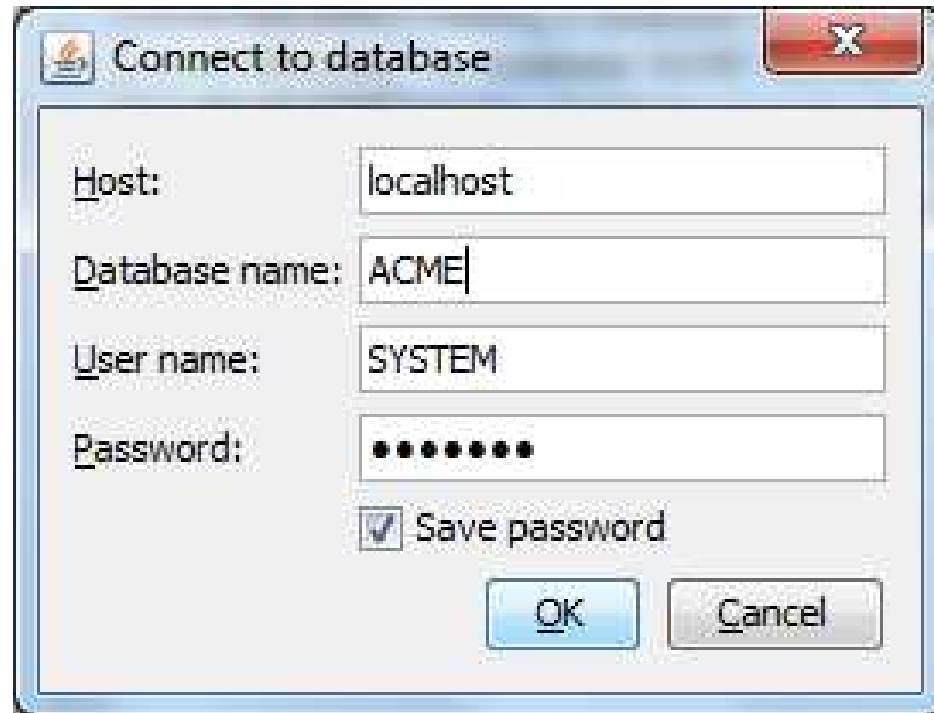
Iniciar o database

► Menu Database/Start



Conectar ao Database

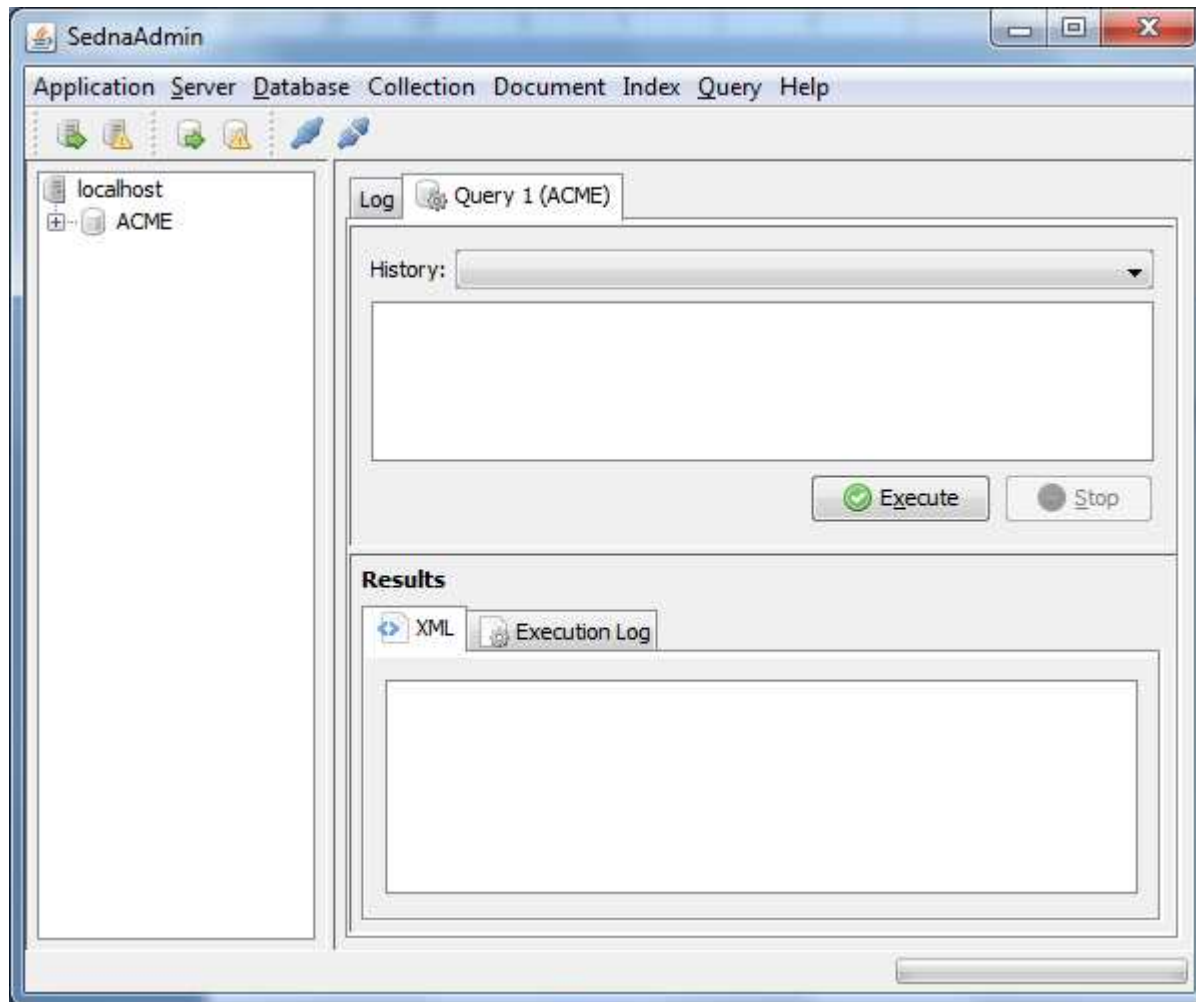
► Menu Database/Connect



A screenshot of a Windows-style dialog box titled "Connect to database". The dialog box has a standard title bar with a minimize button, a maximize button, and a close button (X). Inside the dialog, there are four text input fields arranged vertically. The first field is labeled "Host:" and contains the text "localhost". The second field is labeled "Database name:" and contains the text "ACME". The third field is labeled "User name:" and contains the text "SYSTEM". The fourth field is labeled "Password:" and contains ten black dots, indicating a masked password. Below the password field, there is a checkbox labeled "Save password" which is checked. At the bottom right of the dialog, there are two buttons: "OK" and "Cancel".

Host:	localhost
Database name:	ACME
User name:	SYSTEM
Password:	••••••••••
<input checked="" type="checkbox"/> Save password	
<input type="button" value="OK"/> <input type="button" value="Cancel"/>	

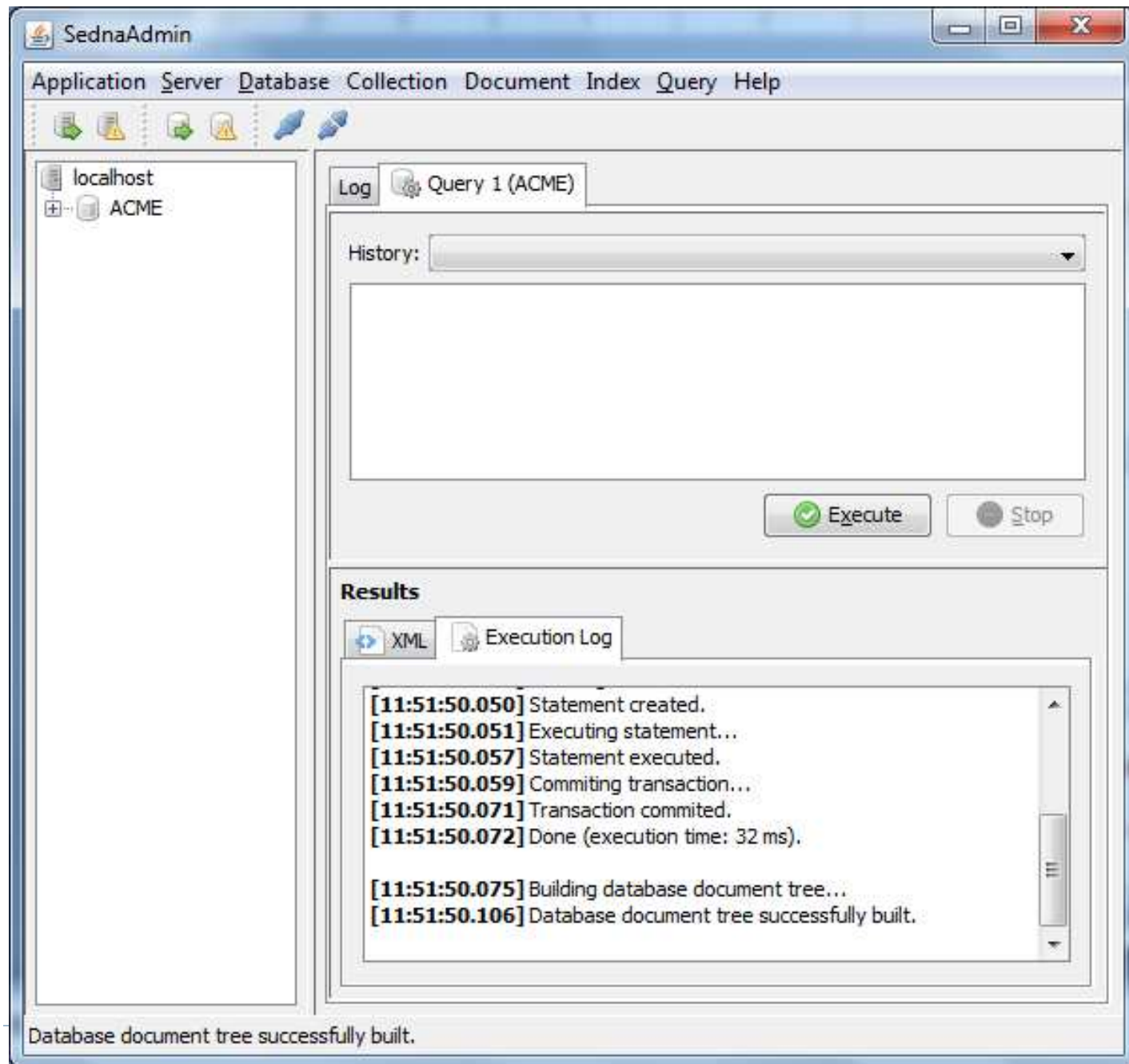
Conectar ao Database



Criar uma coleção

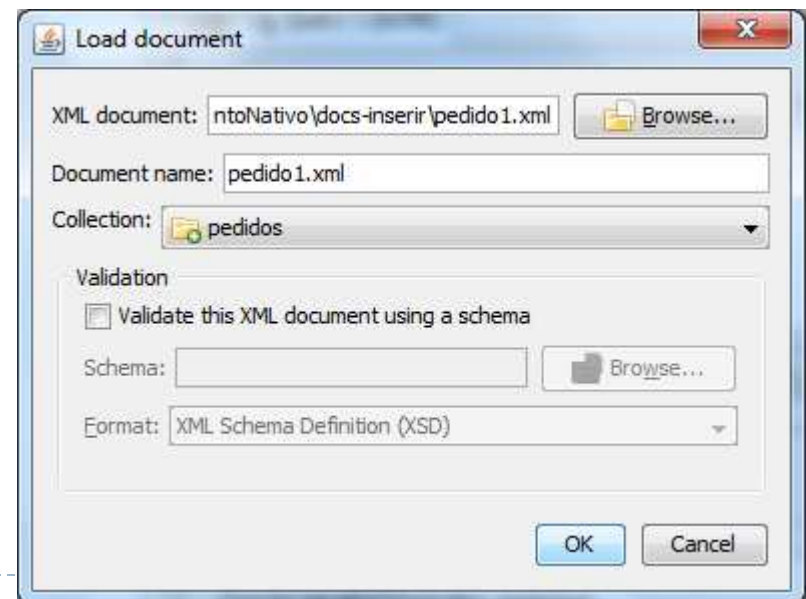
- ▶ Menu Collection/Create collection
- ▶ Digitar o nome da coleção
 - ▶ Vamos criar uma coleção chamada “pedidos” para armazenar os pedidos que são recebidos via Web pela empresa ACME

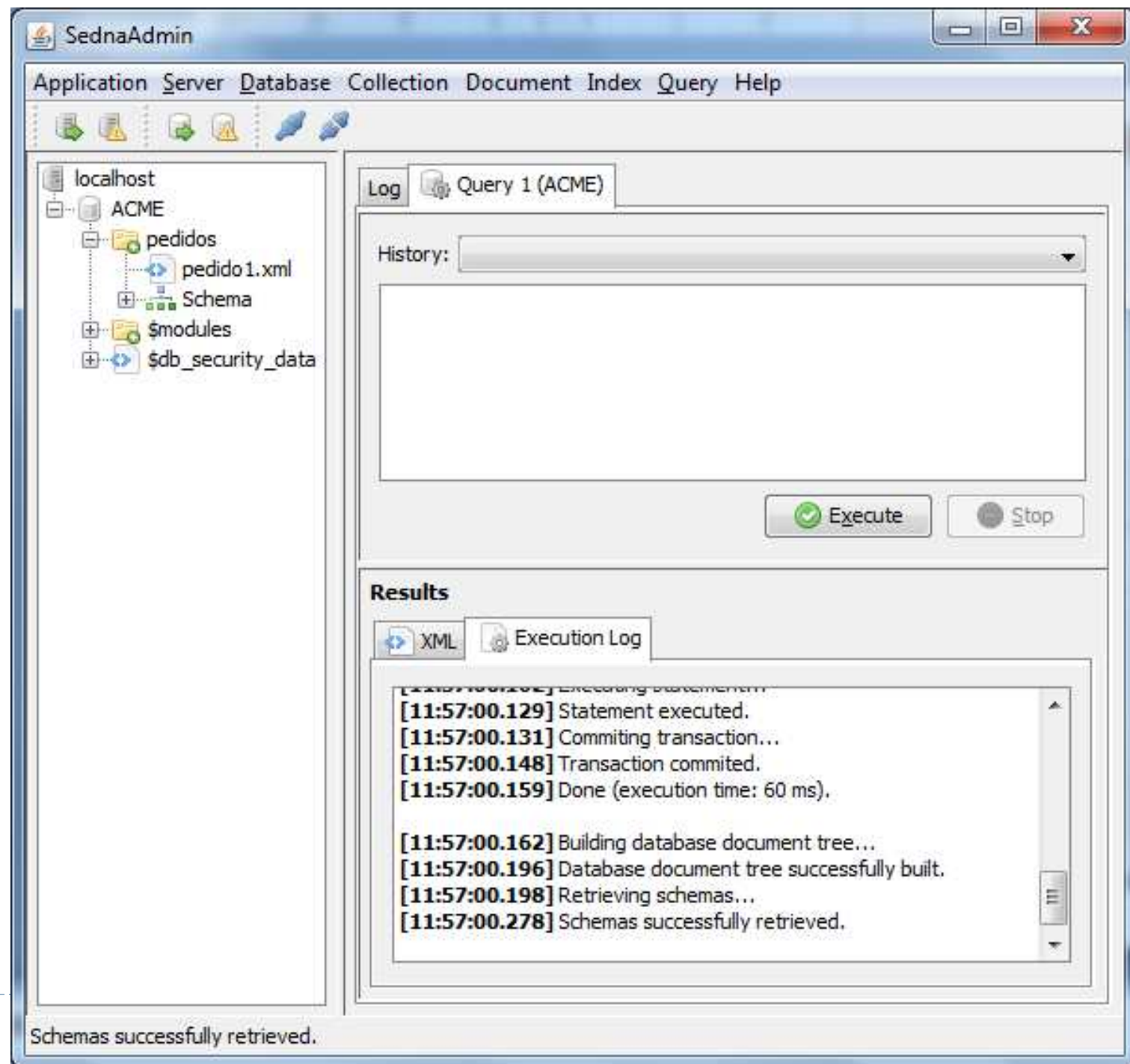




Agora vamos inserir documentos

- ▶ Baixar arquivo zip da página da disciplina que contém os documentos
- ▶ Menu Document/Load Document
- ▶ Selecione o documento pedido1.xml
- ▶ Verifique o nome que o documento terá (chave primária)
- ▶ Selecione o nome da coleção onde o documento deve ser inserido (pedidos)





Outra forma de inserir

- ▶ O Sedna possui uma funcionalidade que permite carregar vários documentos de uma única vez para uma coleção
- ▶ A funcionalidade exige que os documentos sejam especificados através de um comando **LOAD**
- ▶ O comando **LOAD** tem 3 parâmetros
 1. Nome do arquivo a ser carregado
 2. Chave que vai ser usada para identificar o documento
 3. Nome da coleção (opcional)
- 4. Se houver o uso de mais de um comando LOAD, separá-los com **&**



Exemplo: arquivo load-data.xquery

```
LOAD "pedido2.xml" "pedido2.xml" "pedidos" &  
LOAD "pedido3.xml" "pedido3.xml" "pedidos" &  
LOAD "pedido4.xml" "pedido4.xml" "pedidos"
```



Carregamento de arquivos

- ▶ O carregamento deve ser feito via linha de comando
- ▶ A interface de administração ainda não permite isso
- ▶ Digitar o comando
 - ▶ `se_term -file load-data.xquery ACME`



Carregamento de arquivos

```
C:\Windows\system32\cmd.exe

15/11/2010 11:27 <DIR>
15/11/2010 11:27 <DIR>
15/11/2010 11:58      135 load-data.xquery
15/11/2010 11:18    1.559 pedido.xsd
24/05/2006 12:42    481 pedido1.xml
15/11/2010 11:26    484 pedido2.xml
30/03/2004 14:07    613 pedido3.xml
30/03/2004 14:07    505 pedido4.dtd
15/11/2010 11:26    615 pedido4.xml
15/11/2010 11:15    596 pedidoComDTD.xml
15/11/2010 11:20    643 pedidoComERROSSchema.xml
15/11/2010 11:18    670 pedidoComSchema.xml
24/05/2006 13:50    1.082 xupdate.xml
      11 File(s)              7.383 bytes
      2 Dir(s)  77.907.456.000 bytes free

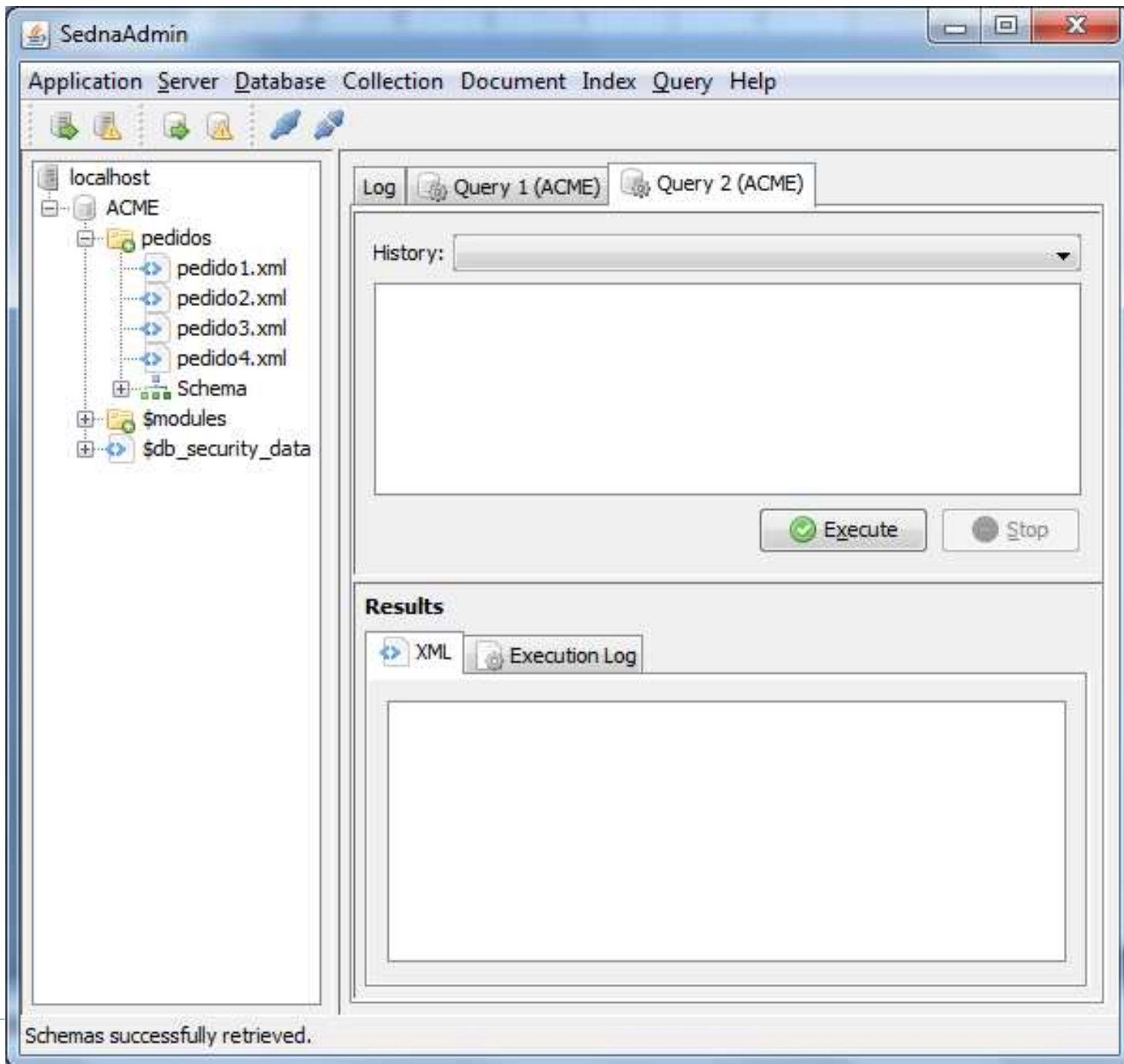
C:\Users\vanessa\Documents\Disciplinas\2010-2-XML\Aulas\06-ArmazenamentoNativo\d
ocs-inserir>se_term -file load-data.xquery ACME
Bulk load succeeded
Bulk load succeeded
Bulk load succeeded

C:\Users\vanessa\Documents\Disciplinas\2010-2-XML\Aulas\06-ArmazenamentoNativo\d
ocs-inserir>
```

Na interface de Administração

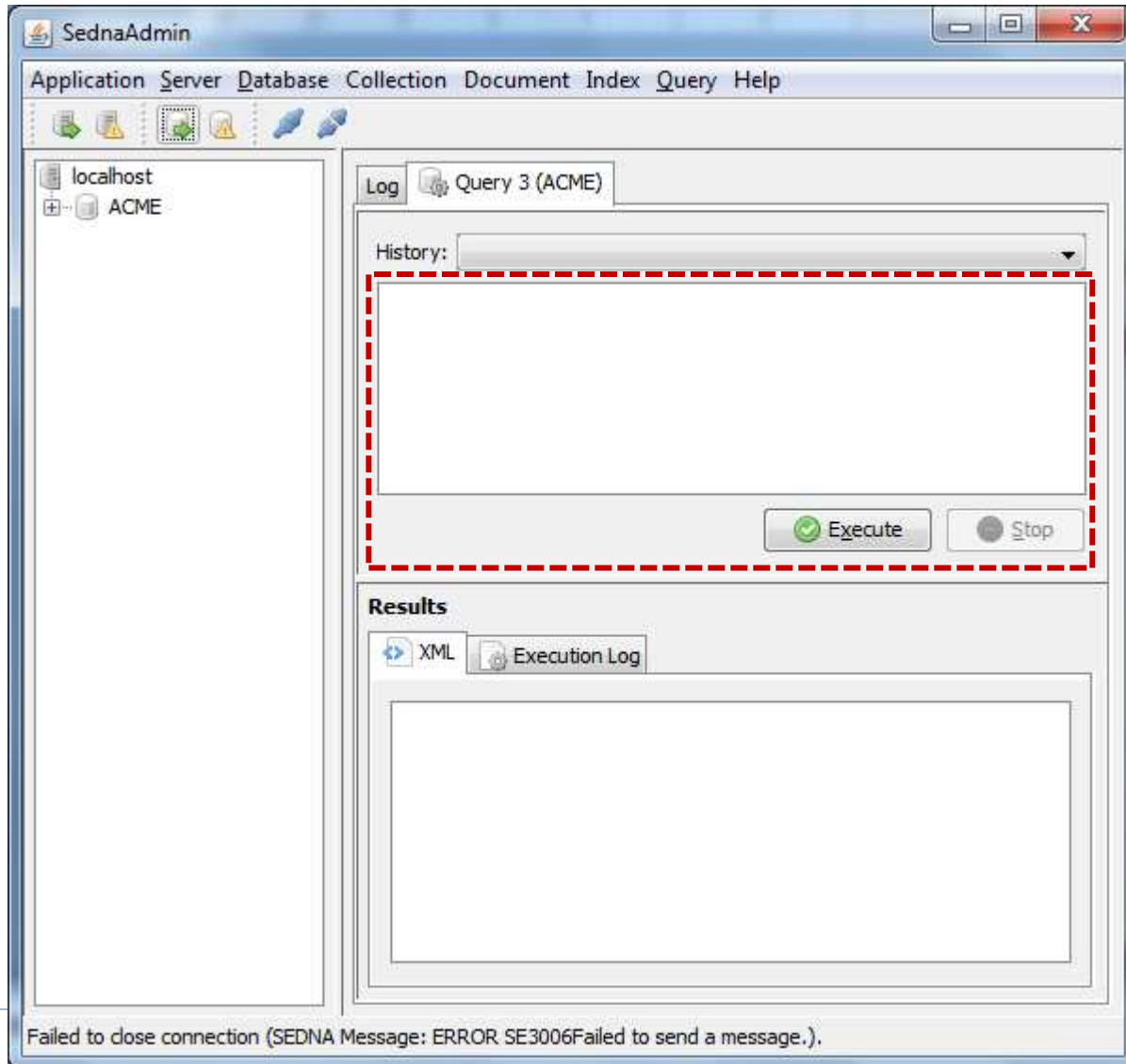
- ▶ Existe um problema de refresh
- ▶ É necessário conectar novamente ao database para ver os documentos que acabamos de inserir
- ▶ Menu Database/Connect



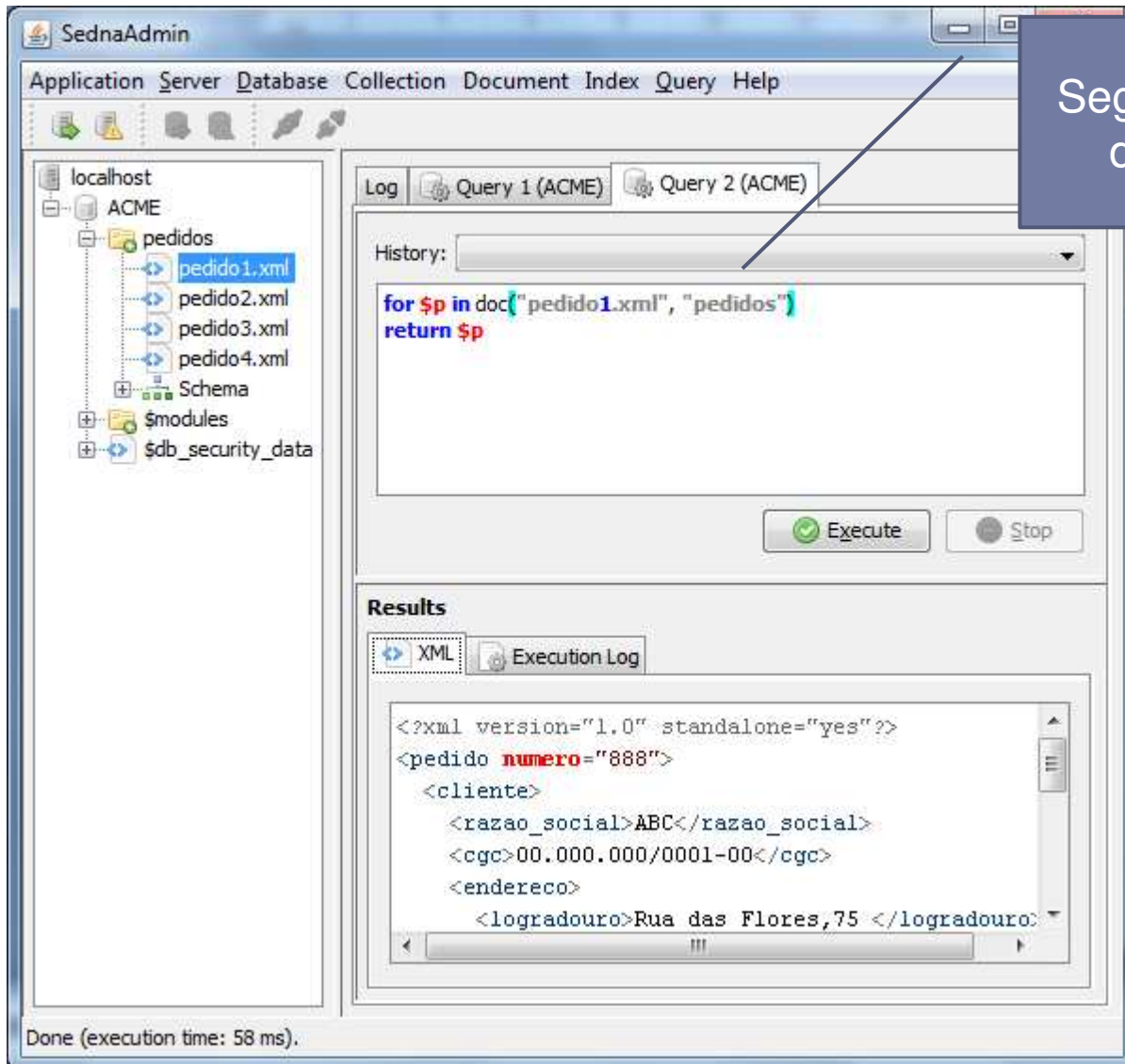


Fazendo consultas XQuery...

Consultas



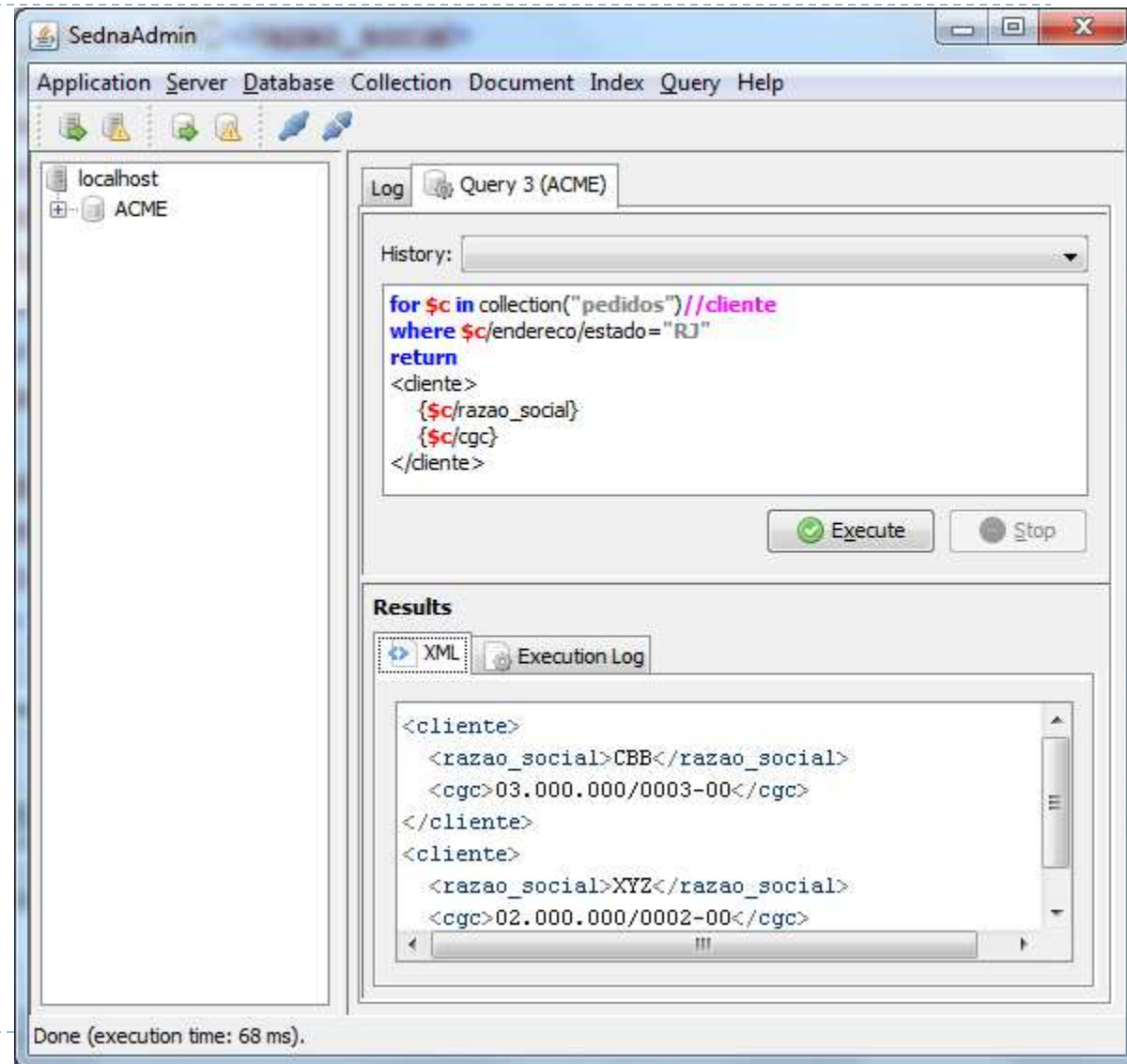
Para ver o conteúdo de um document



Segundo parâmetro da função
doc é o nome da coleção

Para consultar uma coleção uso de **collection**

- ▶ Consultar os pedidos que foram feitos por clientes do estado RJ
- ▶ Retornar razao_social e cgc



Exercícios

1. Faça uma consulta que retorna o elemento `itens_pedido` de todos os pedidos em que a razão social do cliente é “ABC”.
2. Faça uma consulta que retorna todos os produtos de todos os pedidos.
3. Crie mais 2 documentos XML e os insira na coleção pedidos. Repita as consultas anteriores.
4. Faça uma consulta que retorne todos os itens cujo preço é igual à quantidade comprada.
5. Faça uma consulta que retorna todos os itens que possuem “caneta” como parte do nome do produto.



Usar o Sedna em aplicações...

- ▶ Developer Guide: [ProgGuide.pdf](#) dentro do diretório docs do Sedna
- ▶ Exercício: fazer uma aplicação Java que:
 - ▶ Se conecta no Sedna
 - ▶ Cria uma coleção
 - ▶ Insere documentos na coleção
- ▶ **IMPORTANTE:** a senha do usuário SYSTEM é **MANAGER**



Ao terminar

- ▶ Database/Stop
- ▶ Server/Stop



Banco de Datos Habilitado a XML/
Banco de Datos Híbrido

Motivação

- ▶ Empresas que já investiram enormes quantias em licenças de SGBDs relacionais/objeto relacionais
 - ▶ Não estão dispostas a adotar outros tipos de SGBDs para armazenar seus documentos XML
 - ▶ Necessidade de manter DBAs treinados nestes novos SGBDs implica em aumento de custos
- ▶ Empresas fabricantes dos principais SGBDs perceberam este filão de mercado e investiram para permitir armazenamento de docs. XML em seus SGBDs



Primeira geração:

SGBDs Habilitados a XML

- ▶ Novo tipo de coluna (XML), que era capaz de armazenar um DOC XML; e/ou
- ▶ Uso de arquivos de mapeamento para “espalhar” o conteúdo dos docs XML em diversas tabelas do SGBD
- ▶ Suporte a consultas limitado



Nova geração: SGBDs híbridos

- ▶ Suporte a armazenamento de docs. XML em sua forma nativa, ao mesmo tempo em que mantém suporte a armazenamento de dados relacionais/objeto-relacionais



Tarefa em dupla:

- ▶ Cada dupla pesquisa sobre um dos BDs habilitados a XML/híbridos (Oracle, DB2, SQL Server – ou outro que vcs encontrarem)
- ▶ Ver o suporte que o banco escolhido dá para XML
 - ▶ Como armazenar um DOC XML no banco?
 - ▶ Como recuperar o documento armazenado?
 - ▶ É possível gerar um doc. XML a partir de dados relacionais pré-existentes? Como?
- ▶ Apresentar no final da aula

