

# Query Processing su Collezioni di Documenti XML

Dott. Wilma Penzo



wpenzo@deis.unibo.it  
DEIS - Facoltà di Ingegneria  
Università di Bologna

1

## XML (eXstensible Markup Language)

- Standard W3C [XML] [XQuery]
- Documenti di testo
  - Intestazione
  - Elementi individuati da tag
  - [Attributi, ID, IDref, xmlns,...]
  - I dati sono contenuti negli elementi
  - Gli elementi definiscono il contesto cui i dati si riferiscono

*cd.dtd*

```
<!ELEMENT cdlist (cd*)>  
<!ELEMENT cd (title, singer, tracklist)>  
<!ELEMENT title (#PCDATA)>  
<!ELEMENT singer (#PCDATA)>  
<!ELEMENT tracklist (track*)>  
<!ELEMENT track (title)>
```

CDlist.xml

```
<?xml version="1.0"?>  
<!DOCTYPE cdlist SYSTEM "cd.dtd">  
<cdlist>  
...  
<cd>  
  <title>One night only</title>  
  <singer>Elton John</singer>  
  <tracklist>  
    <track>  
      <title>  
        Can you feel the love tonight  
      </title>  
    </track>  
    ...  
  </tracklist>  
</cd>  
...  
</cdlist>
```

## Perché database XML

- Vantaggi
  - Semantica
  - Supporto all'eterogeneità dei dati
  - Formato semplice e flessibile
    - Trasmissione, interscambi fra applicazioni
    - Separazione dati da presentazione/logica applicativa
- Ambiti
  - Ovunque sia richiesta una gestione di dati non strutturati
    - Digital libraries, E-business, mobile database, Web content, sistemi CRM, sistemi supply chain

3

## In cosa un RDBMS è carente

- Indipendenza dallo schema
- Query capability per dati semi-strutturati
- Supporto built-in per contenuti di tipo non numerico
  - Stemmer
  - Dizionari
  - Ontologie di dominio
- Molti sistemi forniscono solo un interfacciamento a XML
- Nascono i primi database XML nativi (Tamino [XDBMS])
- I vendor più noti (Oracle [XMLDB], IBM [XMLExt], Microsoft [XSQL]) forniscono un supporto integrato e completo

Sono in atto molti investimenti

4

## Retrieval di documenti XML

- Query language standard: XQuery 1.0

- Selettori:

- path expression
- wildcard
- posizione di elementi in sequenza
- predicati aritmetici e di confronto (Booleano!)
- formule di predicati combinati con connettivi logici (and, or)

**Esempi:**

1) *CD di Elton John venduti in negozi di New York*

```
cdstore/cd[artist="Elton John" and ../address//!="New York"]
```

2) *titolo della seconda canzone del CD di Elton John intitolato "One night only"*

```
cd/song[2]/stitle[../../title="One night only"] oppure  
cd[title="One night only"]/song[2]/stitle
```

5

## Storage di documenti XML

- File XML

- su file system, in BLOB o CLOB in DB relazionali
- Adatto per file XML document-centric (narrativi)
- Il contenuto non può essere usato per il retrieval
- Creazione di indici a parte
- Retrieval del documento intero
- Query di aggregazione inefficienti

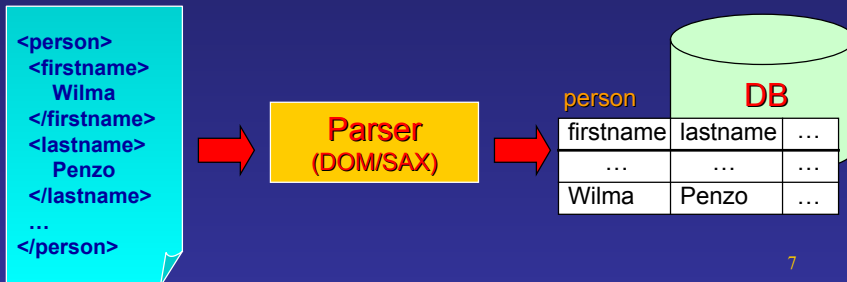
book

author	title	content
...	...	
Harold	XML....	

```
<chapter>1  
<title>  
  XML Concepts  
</title>  
<content>  
  XML is a W3C standard  
  for document markup  
  ...  
</content>  
</chapter>  
...
```

## Storage di documenti XML

- Tabelle relazionali
  - Estrazione dei dati
  - Adatto per file XML data-centric con struttura regolare
  - Il documento originale viene perso
  - Inefficienza nella ricostruzione del documento XML



7

## Database XML Nativi

- Unità di memorizzazione: documento XML
- Linguaggio di interrogazione: XQuery
- Update: Manipolazione DOM
- Esempio: TAMINO XML Server
  - Full-Text Retrieval
  - query execution engine implementa un'algebra di operatori su alberi (basata su ricorsione)
  - structure-index + value-index



8

## DB2 XML Extender

- Incluso dalla versione 7.1
- documenti XML da dati DB2
- repository di DTD
- compatibile con il Text Extender (per la ricerca di testo)
  
- due modalità di memorizzazione:
  - 1) XML column (3 UDT):
    - ✓ XMLCLOB (Character Large Object)
    - ✓ XMLVARCHAR (tipicamente un URL)
    - ✓ XMLFile (riferimento a file su file system locale)
  - 2) XML collection
    - ✓ insieme di tabelle relazionali contenenti dati dei documenti XML
- Data Access Definition (DAD):
  - quali elementi/attributi indicizzare (1)
  - mapping da DTD a tabelle e colonne (2)
- UDF per insert, select e update di parti o interi documenti (1)
- Stored procedures per store, retrieve, update, search e delete (2)

## DB2 XML Extender

```
TABLE sales_tab
  invoice_number CHAR(6) NOT NULL
                  PRIMARY KEY
  sales_person   VARCHAR(20)
  order          XMLVARCHAR
```

**Esempio:** Trova i nomi dei commerciali che hanno ricevuto ordini da IBM con numero dell'invoice superiore a 100

```
Select sales_person
From sales_tab
Where extractVarChar(order, '/order/customer') LIKE "%IBM%"
And invoice_number > 100
```

La UDF `extractVarChar` esamina il documento XML contenuto in order alla ricerca della stringa IBM secondo il path /order/customer

**Alternativa:** Text-Extender per la colonna XML per usare il Text Indexing (UDF interna `db2bx.contains`)

## DB2 XML Extender

- Per XML collection:
  - Ricerca diretta sulle tabelle SQL
  - Stored procedures `dxxRetrieveXML()` e `dxxGenXML()` per interrogare e generare documenti XML
- DAD:
  - Specifica se effettuare il retrieval di porzioni o interi documenti
  - Inserimenti condizionati al valore di elementi e attributi XML

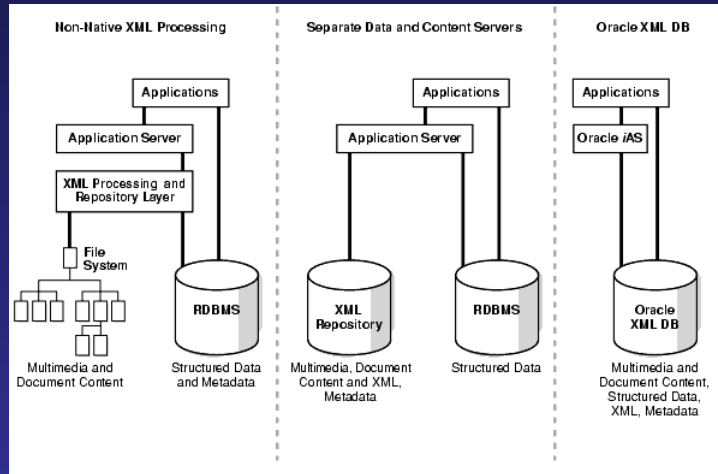
11

## Oracle XML DB

- XMLType nativo consente alternative di storage:
  - CLOB
  - Structured XML
- Schema-aware: mapping XML-to-relational database  
`dbms_xmlschema.registerSchema('http://127.0.0.1:8080/home/Scott/xsd/purchaseOrder.xsd','purchaseOrder.xsd')`
  - I documenti XML vengono effettivamente decomposti in tabelle relazionali secondo lo schema indicato
  - Dal punto di vista utente rimangono documenti XML
  - Metafora gerarchica (XML DB Foldering Module)
- Hierarchical Index trasparente all'utente
  - Speed up per navigazione di folder e path

12

# Oracle XML DB



13

# Oracle XML DB

- Supporta XML Schema (vincoli di ordine, cardinalità, integrità referenziale)
- DOM Fidelity (info su ordine e namespace)
- Storage
  - Eliminazione di tag e spazi
  - Ricostruzione dinamica tramite XML Schema
  - XML Object (XOB) vs. DOM
    - Versione DOM "light"
    - Lazily-Loaded Virtual DOM
- Query

```
Select value(x)
From purchaseorder x
Where existsNode(value(x),'/PurchaseOrder[User="Smith"]')=1
```

  - Implementa alcune funzioni del (futuro) standard SQL/XML (IBM, Microsoft, Oracle, Sybase sono alcuni vendors coinvolti)

14

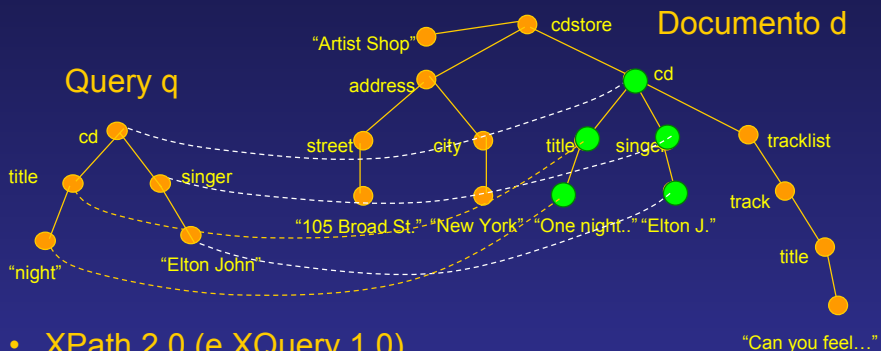
# La Ricerca su XML

- Strettamente legata a tematiche database:
  - Storage
  - Query Processing
  - Indexing
  - Data Integration
  - Publishing
  - Benchmarking
  - ...



15

## Interrogazioni su documenti XML



- XPath 2.0 (e XQuery 1.0)
  - Path expression, wildcard, posizione in sequenza,...

*CD di Elton John venduti in negozi dello stato di New York*

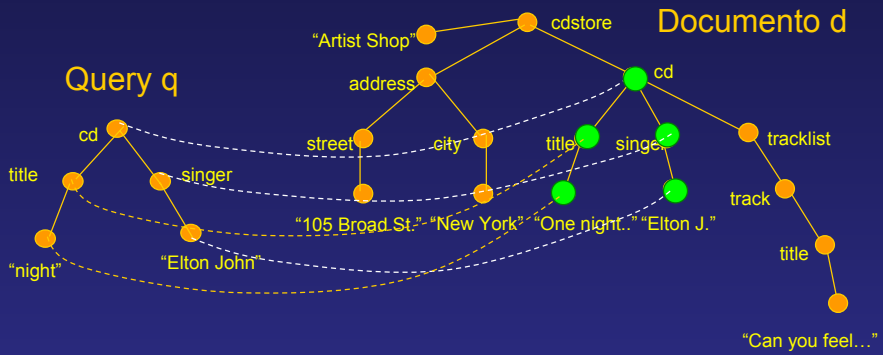
```
/cdstore/cd[singer="Elton John" and ../address/*="New York"]
```

*CD di Elton John contenenti nel titolo il termine "night"*

 `//cd[singer="Elton John" and title="night"]`

16

# Tree Matching



Q set di nodi query, D set di nodi dati

$f: Q \rightarrow D, \forall q_i, q_j \in Q:$

- 1)  $f(q_i) = f(q_j) \Leftrightarrow q_i = q_j$
- 2)  $label(q_i) = label(f(q_i))$
- 3)  $parent(q_i, q_j) \Leftrightarrow parent(f(q_i), f(q_j))$

# Tree Embedding



$f: Q \rightarrow D, \forall q_i, q_j \in Q:$

- 1)  $f(q_i) = f(q_j) \Leftrightarrow q_i = q_j$
- 2)  $label(q_i) = label(f(q_i))$
- 3)  $parent(q_i, q_j) \Leftrightarrow ancestor(f(q_i), f(q_j))$

## Limiti di XQuery

- ... e di molti altri linguaggi (XML-GL, XML-QL, Lorel,...)
- presuppone conoscenza sull'organizzazione dei dati (DTD)
- cattura solo soluzioni con match esatto sul contenuto e sul nome degli elementi/attributi

### Rilassamento di vincoli:

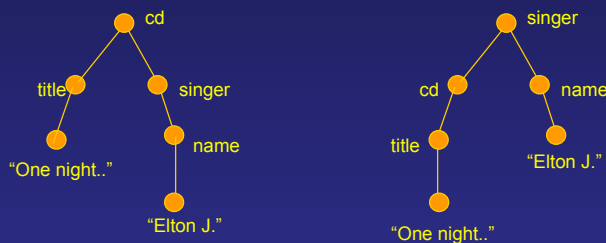
- introduzione di predicati di similarità
- approssimazioni sulla struttura
- ranking dei risultati per rilevanza



19

## Perché interrogazioni approssimate

- Assenza di conoscenza dello schema dei dati
- Rappresentazione non univoca di relazioni M:N



- Eterogeneità di relazioni strutturali per gli elementi di uno stesso documento/collezione
- [ACS02] [FG00] [SN00] [TW00]

20

## Tipi di approssimazioni

### Query:

Trova i CD contenenti brani con "love" nel titolo

### Goal: ricerca del pattern

```
<cd>
  <song>
    <title> love </title>
  </song>
</cd>
```

### Doc1.xml

```
<cd>
  <title>One night only</title>
  <singer>Elton John</singer>
  <tracklist>
    <track>
      <title>
        Can you feel the love tonight
      </title>
    </track>
    ...
  </tracklist>
</cd>
```

21

## Tipi di approssimazioni

### Query:

Trova i CD contenenti brani con "love" nel titolo

### Goal: ricerca del pattern

```
<cd>
  <song>
    <title> love </title>
  </song>
</cd>
```

Doc1.xml è rilevante, anche se approssimato:

- track è simile a "song" (semantic relaxation)
- tracklist può essere trascurato (structural relaxation)

### Doc1.xml

```
<cd>
  <title>One night only</title>
  <singer>Elton John</singer>
  <tracklist>
    <track>
      <title>
        Can you feel the love tonight
      </title>
    </track>
    ...
  </tracklist>
</cd>
```



22

# XXL [Theobald & Weikum 2000]

- Introduce predicati di similarità semantica

**Esempio:** Trova i brani dei CD di Elton John in cui suonano sassofonisti

```
Select T
From http://my.cdcollection.edu/CDstores.xml
Where #.cd As C
And C.#.(~artist) As A
And A = "Elton John"
And C.#.(~track)? As T
And T. ~musician As M
And M.# ~ "saxophone"
```

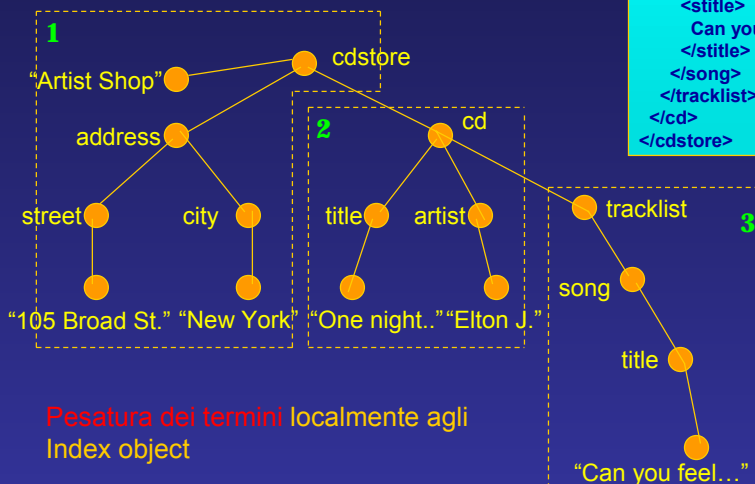
- Ranking

- I predicati Booleani hanno valore 0 o 1 a seconda del valore di verità
- score di similarità inteso come probabilità di rilevanza
- combinazione degli score assumendo indipendenza fra le condizioni:
  - **AND algebrico per path expression e congiunzioni** ( $\pi_1 * \pi_2$ )
  - **OR algebrico per disgiunzioni** ( $\pi_1 + \pi_2 - \pi_1 * \pi_2$ )
- le wildcard assumono valore 1

23

# XIRQL [Fuhr & Großjohann 2001]

Albero partizionato in **index object**

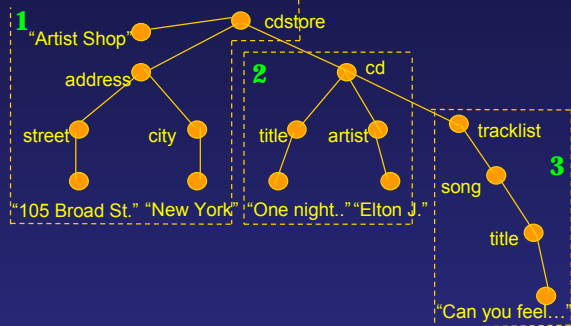


**Pesatura dei termini** localmente agli  
Index object

```
<cdstore>Artist Shop
<address street="105 Broad St."
  city="New York"></>
<cd>
<title>One night only</title>
<artist>Elton John</artist>
<tracklist>
  <song>
  <title>
  Can you feel ...
  </title>
  </song>
</tracklist>
</cd>
</cdstore>
```

24

# XIRQL [Fuhr & Großjohann 2001]



Esempio di query:

```
//cd[./artist = "Elton John" $and$ ./song/title $cw$ "love"]
```

Ranking: combinazione Booleana di eventi probabilistici indipendenti

$[2, "Elton John"] \wedge [3, "love"]$  score calcolato come:  $W_{1,2} * W_{2,3}$

con  $W_{i,j}$  peso del termine della condizione  $i$  nell'index object  $j$

se "love" fosse presente anche in 1:  $[2, "Elton John"] \wedge ([1, "love"] \vee [3, "love"])$

Possibile pesatura delle condizioni: `//cd[0.6*cond1 $and$ 0.4*cond2]`

25

# Approximate Tree Embedding



$e: Q \mapsto D, \forall q_i, q_j \in \text{dom}(e):$

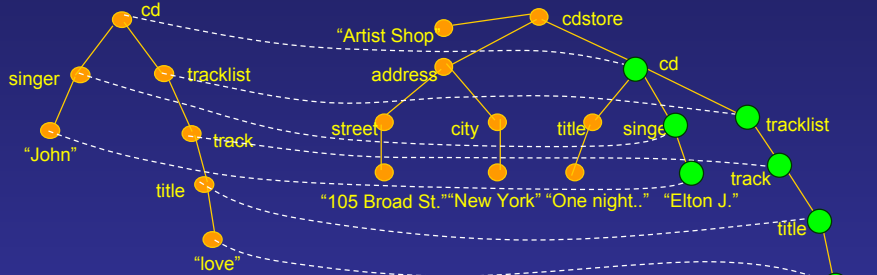
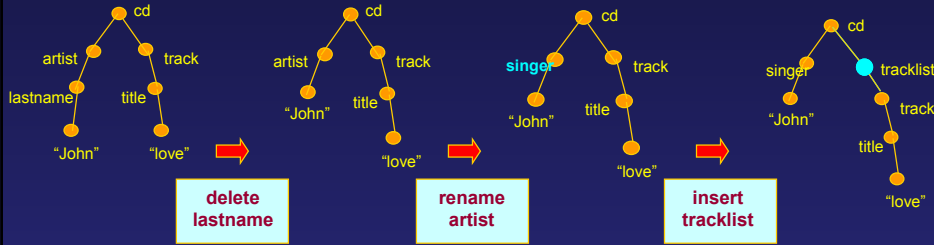
1)  $e(q_i) = e(q_j) \Leftrightarrow q_i = q_j$

2)  $\text{sim}(\text{label}(q_i), \text{label}(e(q_i))) > 0$

3)  $\text{parent}(q_i, q_j) \Leftrightarrow \text{ancestor}(e(q_i), e(q_j))$

26

# Ranking: ApproxQL [Sch01]



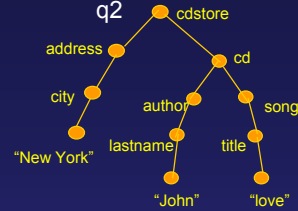
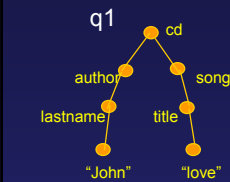
Se ogni trasformazione costa 1  
Costo totale = 3

Ranking di similarità sul costo  
minor costo → maggior similarità

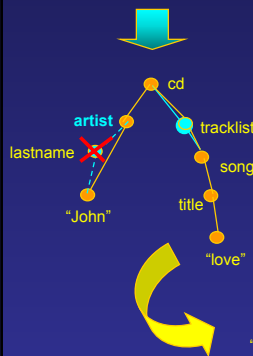
"Can you feel..."  
27

# Ranking

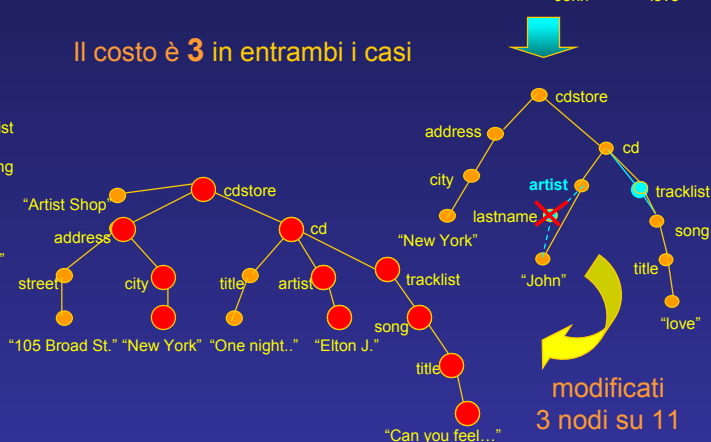
Score assoluto  
e.g. ApproxQL:  
numero di trasformazioni



Il costo è 3 in entrambi i casi



modificati  
3 nodi su 7



modificati  
3 nodi su 11

# SATES [Ciaccia & Penzo 2002]



## Approximate Tree Embedding

Q set di nodi query, D set di nodi dati,  $e: Q \rightarrow D, \forall q_i, q_j \in \text{dom}(e)$ :

- 1)  $\text{sim}(\text{label}(q_i), \text{label}(e(q_i))) > 0$
- 2)  $\text{parent}(q_i, q_j) \Rightarrow (\text{ancestor}(e(q_i), e(q_j)) \vee \text{sibling}(e(q_i), e(q_j)))$
- 3)  $\text{sibling}(q_i, q_j) \Rightarrow (\text{sibling}(e(q_i), e(q_j)) \vee \text{ancestor}(e(q_i), e(q_j)))$

Penalizzazione strutturale per il ranking

29

# SATES [Ciaccia & Penzo 2002]



Ranking:

misura di **correttezza** e **completezza** semantica e strutturale  
 misura di **coesione** dei risultati

**sem**(q1) > **sem**(q2) [cd\_vendor ~ cdstore]

**sem**(q1) < **sem**(q2) [6/7 matching nodes vs. 8/9 matching nodes]

**str**(q1) = **str**(q2) [1 penalità: song/author vs. cd/artist]

**str**(q1) < **str**(q2) [9/15 (=0.6) vs. 16/23 (=0.69...) matching arcs]

**coes**(q1) > **coes**(q2) [6/7 (=0.85...) vs. 8/11 (=0.72...) relevant data nodes]

## La Ricerca su XML

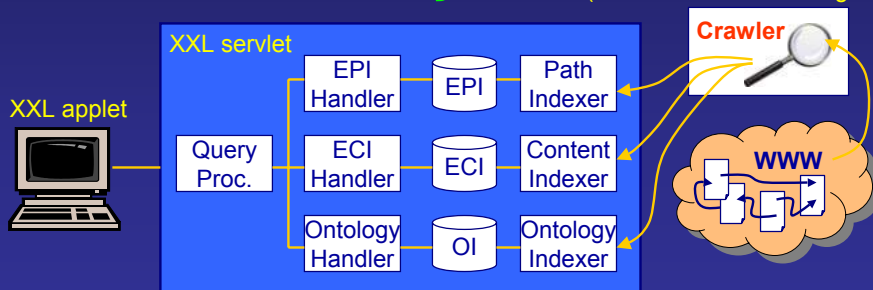
- Strettamente legata a tematiche database:
  - Storage
  - Query Processing
  - Indexing
  - Data Integration
  - Publishing
  - Benchmarking
  - ...



31

## Index-based XXL SE [Theobald & Weikum 2002]

- Ispirato a Web search engine
    - Indici costruiti e aggiornati da crawler
  - 3 tipi di indice
    - Element Path Index (EPI)
    - Element Content Index (ECI)
    - Ontology Index (OI)
- Inverted List  
(elemento, <[URL1,oid1,parent1,children1], ...>)
- Tabelle Oracle  
InterMedia (Oracle text retrieval engine)

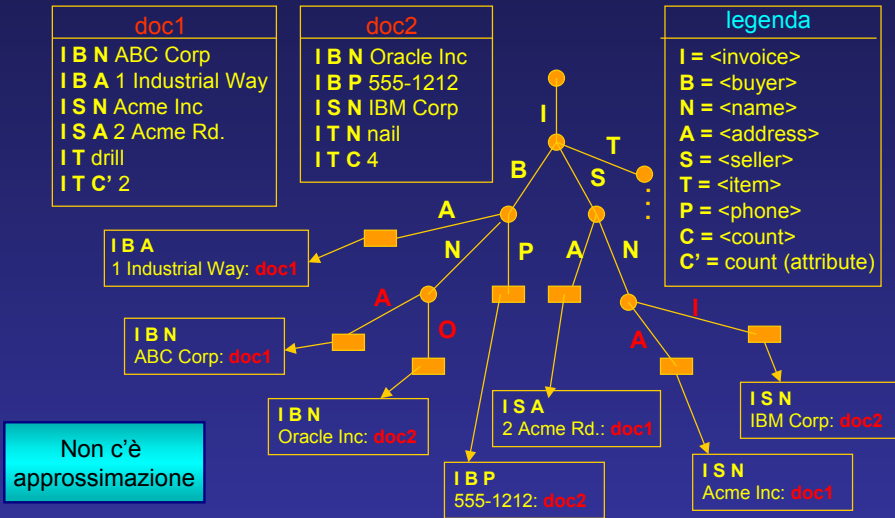


L'approssimazione strutturale richiede wildcard esplicite nella query  
Navigazione di tutti i collegamenti!!!

32

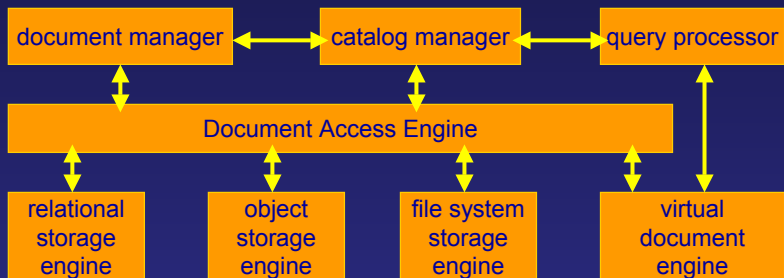
# Index Fabric [Cooper et al. 2001]

- Codifica dei path in stringhe
- Indice ottimizzato per la ricerca di stringhe (Patricia Trie)



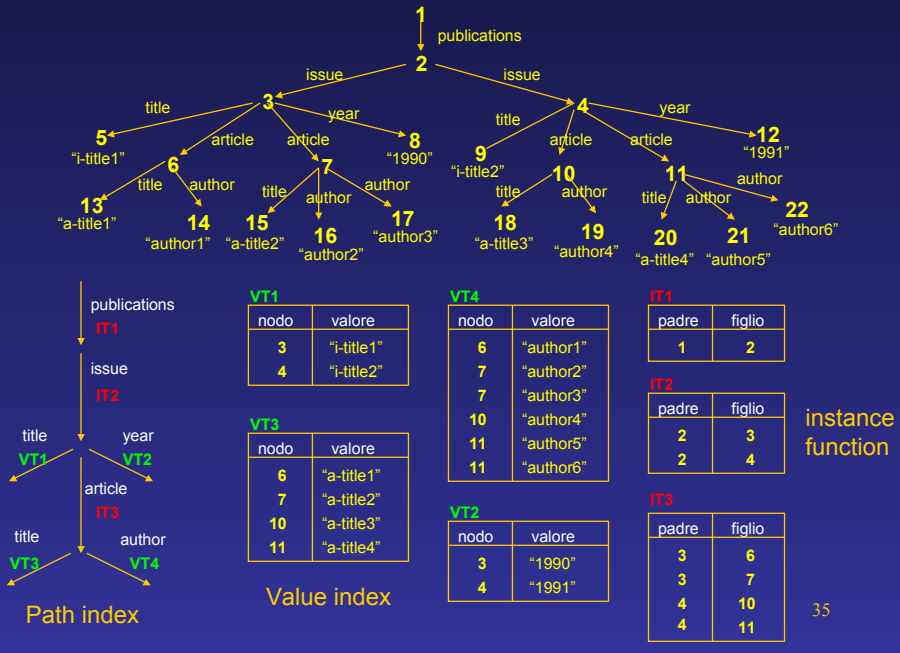
# ToX [Barbosa et al. 2001]

- The Toronto XML Engine



- Storage e indexing eterogenei sulla base della *structuredness* dei documenti
- DB2 (future work: altri RDBMS via ODBC)
- Ozone: storage engine per oggetti Java

# ToXin (ToX indexing)



35

# ApproXQL Index [Schlieder 2002]



Albero "decorato":

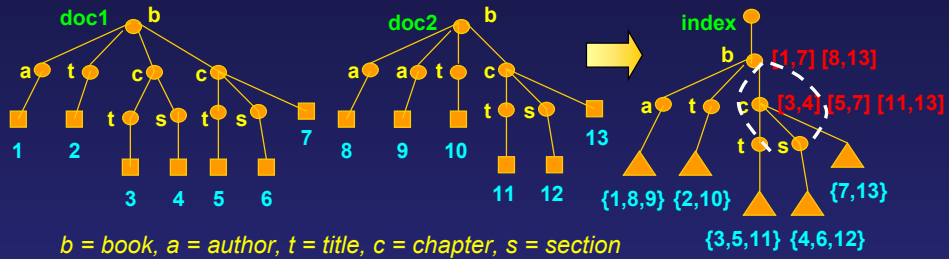
- Numerazione: (preorder # : max id delle foglie sottese)
- costi di inserimento nodo e inserimento path (omessi in figura...)

Operazioni su liste (con calcolo del costo di approssimazione)

- Navigazione bottom-up dell'albero query
- accesso all'indice I<sub>text</sub> per reperire i nodi rilevanti per le foglie query
- accesso all'indice I<sub>struct</sub> per reperire i nodi intermedi
- check di ancestorship sfruttando la numerazione

36

# Collection Index [Ciaccia & Penzo 2003]



L'indice individua lo schema della struttura dei dati

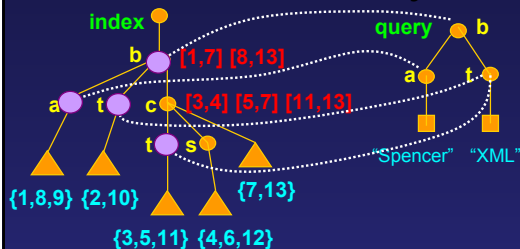
- Ogni path individua una classe di equivalenza di path dati
- Ogni nodo interno mantiene:
  - i riferimenti ai discendenti
  - una lista di **range**
- Ogni nodo foglia referencia un **value-index**

Creazione:

- Numerazione delle foglie dati
- Inserimento dei path dei documenti senza ripetizioni

37

# Query Processing



Data Stream path	Relevant data
ds1: /book/author	{1}
ds2: /book/title	{10}
ds3: /book/chapter/title	{3,5,11}

Risultato 1: {1,3,5}

```
<book>
  <author>
    Spencer
  </author>
  <chapter>
    <title>
      XML basics
    </title>
  </chapter>
  <chapter>
    <title>
      XML tools
    </title>
  </chapter>
</book>
```

Risultato 2: {10,11}

```
<book>
  <title>
    XML
  </title>
  <chapter>
    <title>
      XML for DigLib
    </title>
  </chapter>
</book>
```

- 1) Esplorazione top-down alla ricerca dei match dei nodi query
  - ✓ Navigazione diretta sui discendenti
  - ✓ Contestualizzazione a ogni match
- 2) Costruzione risultati bottom-up
  - ✓ Set di data stream per ogni path query
  - ✓ Test di inclusione fra range

38

## Collection Index

- Supporto efficiente per query approssimate complesse
  - Rappresentazione “intensionale”
    - Template esteso della struttura dei dati
    - Non è vincolato alla presenza di un DTD
    - Lo schema viene determinato dai dati stessi
  - La ricerca dell’embedding avviene solo sul template dei dati
  - I risultati sono ranked secondo la misura SATES
  - Flessibilità nelle strategie di ricerca
  - Consente l’indicizzazione di collezioni eterogenee

39

## XML Benchmark

- Framework per analisi di:
  - funzionalità di interrogazione
  - performancedei sistemi di gestione di dati XML
- Esempi: Xmark, XMach-1, X007
- Index Collection su XMach:
  - Documenti con struttura eterogenea, profondità fino a 12 livelli, ripetizione di label su singolo path, ripetizione di path
  - Data generator (documenti)
    - ✓ SW Java
    - ✓ Genera documenti conformi a un dato DTD
    - ✓ Testo generato sinteticamente da una lista di 10000 termini inglesi, 2000 nomi e 8000 cognomi per gli autori
  - Test da 1000 a 150000 documenti (20 Mb - 2.3 Gb)

40

## Bibliografia

- [ACS02] S. Amer-Yahia, S. Cho, D. Srivastava. Tree Pattern Relaxation. Proc. 8th Int. Conf. On Extending Database Technology (EDBT), 2002.
- [BBM+01] D. Barbosa, A. Barta, A. Mendelzon, G. Mihaila, F. Rizzolo, P. Rodriguez-Gianolli. ToX – The Toronto XML Engine. Proc. Int. Workshop on Information Integration on the Web, 2001.
- [BR01] T. Böhme, E. Rahm. XMach-1: A Benchmark for XML Data Management. Proc. BTW 2001.
- [CP02] P. Ciaccia, W. Penzo. Adding Flexibility to Structure Similarity Queries on XML Data. 5th Int. Conf. On Flexible Query Answering Systems, 2002.
- [CP02a] P. Ciaccia, W. Penzo. Relevance Ranking Tuning for Similarity Queries on XML Data. 1st VLDB Work. EEXTT 2002.
- [CP03] P. Ciaccia, W. Penzo. The Collection Index to Support Complex Approximate Queries on XML Documents, 1st Int. VLDB XML and Database Symposium (XSym03) 2003.
- [CSF+01] B. Cooper et al. A Fast Index for Semistructured Data. VLDB 2001.
- [FG01] N. Fuhr, K. Großjohann. XIRQL: A Query Language for Information Retrieval in XML Documents. Proc. of 24th Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, 2001
- [GW97] R. Goldman, J. Widom. DataGuides: Enabling Query Formulation and Optimization in Semistructured Databases. VLDB 1997.

41

## Bibliografia

- [Kil92] P. Kilpeläinen. Tree Matching Problems with Application to Structured Text Databases. PhD. Thesis, Dept. of Comp. Sci., Univ. of Helsinki, 1992.
- [LM01] Q. Li, B. Moon. Indexing and Querying XML Data for Regular Path Expressions. VLDB 2001.
- [MS99] T. Milo, D. Suciu. Index Structures for Path Expressions. Proc. 7th Int. Conf. on Database Theory (ICDT), 1999.
- [Pen02] W. Penzo. Integration of Semantic and Structure Similarity for XML Data Ranking. SEBD 2002.
- [RM01] F. Rizzolo, A. Mendelzon. Indexing XML Data with ToXin. Proc. 4th Int. Work. on the Web and Databases (WebDB) 2001.
- [Sch02] T. Schlieder. Schema-Driven Evaluation of Approximate Tree-Pattern Queries. EDBT 2002.
- [SN00] T. Schlieder, F. Naumann. Approximate Tree Embedding for Querying XML Data. Proc. ACM SIGIR Work. on XML and Inform. Retrieval, 2000
- [TW00] A. Theobald, G. Weikum. Adding Relevance to XML. 3rd Int. Workshop on the Web and Databases (WebDB 2000)
- [TW02] A. Theobald, G. Weikum. The Index-Based XXL Search Engine for Querying XML Data with Relevance Ranking. EDBT 2002.

42

# Bibliografia

- [XDBMS] Tamino XML Server. <http://www.softwareag.com/tamino/>
- [XML] Extensible Markup Language (XML) 1.0. W3C Recommendation, (2nd ed.), 2000, <http://www.w3.org/TR/REC-xml>
- [XMLExt] IBM DB2 XML Extender.  
<http://www-3.ibm.com/software/data/db2/extenders/xmlxt/xmlxtbroch.pdf>
- [XMLDB] Oracle XML DB. <http://otn.oracle.com/tech/xml/xmlldb/content.html>
- [XSQL] MicroSoft SQLServer, XML Web Services.  
<http://www.microsoft.com/sql/techinfo/xml>
- [XQuery] XQuery 1.0: A Query Language for XML. W3C Working draft, apr. 2002, <http://www.w3.org/TR/xquery>
- The Apache XML Project. <http://xml.apache.org>
- Ozone: The Open Source Java ODBMS. <http://www.ozone-db.org>
- Xmark – The XML Benchmark Project. <http://www.xml-benchmark.org>
- XMach-1: A Benchmark for XML Data Management.  
<http://www.dbs.uni-leipzig.de/en/projekte/XML/XMLBenchmarking.html>
- The X007 Benchmark. <http://www.comp.nus.edu.sg/~ebh/X007.html>