Γλώσσες Ερωτήσεων

Εισαγωγή

1. ΜΟΝΤΕΛΑ ΔΕΔΟΜΕΝΩΝ
   - Μοντέλα
   - → **Γλώσσες Ερωτήσεων**
     - **Μονοπάτια**
     - **Μια βασική γλώσσα για ημι-δομημένα δεδομένα**
     - **Σημασιολογία**
     - **XML-QL**
     - **Γλώσσες ερωτήσεων για το web**
2. ΑΡΧΙΤΕΚΤΟΝΙΚΕΣ
3. ΤΡΟΠΟΙ ΜΕΤΑΔΟΣΗΣ

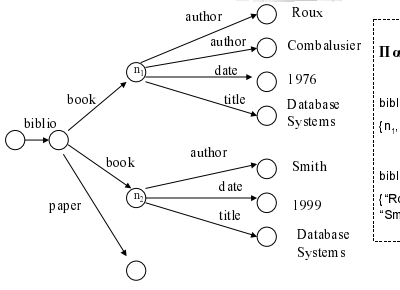
Εκφράσεις για μονοπάτια

Εκφράσεις μονοπατιών

Έκφραση μονοπατιού (path expression): μια ακολουθία από ετικέτες ακμών: $l_1. l_2, \ldots l_n$

- Μια έκφραση μονοπατιού μπορεί να θεωρηθεί μια απλή ερώτηση με αποτέλεσμα ένα σύνολο κόμβων

Εκφράσεις μονοπατιών



Παράδειγμα

biblio.book

{n_1, n_2}

biblio.book.author

{"Roux", "Combalusier", "Smith"}

Εκφράσεις μονοπατιών

Το αποτέλεσμα μιας έκφρασης μονοπατιών $l_1. l_2, \ldots l_n$ σε ένα γράφο είναι το σύνολο των κόμβων v_n τέτοιοι ώστε: υπάρχουν ακμές (r, l_1, v_1), $(v_1, l_2, v_2), \ldots (v_{n-1}, l_n, v_n)$ στο γράφο όπου r είναι η ρίζα

Θα θέλαμε επίσης να μπορούμε να προσδιορίζουμε μονοπάτια με βάση κάποια ιδιότητά τους

Χρήση κανονικών εκφράσεων

Εκφράσεις μονοπατιών

• Για μονοπάτια

- book | paper, π.χ., biblio. (book | paper).author
- _: wild card (ταιριάζει οποιοδήποτε χαρακτήρα), π.χ., biblio._.author
- *, π.χ., biblio._*._author

e ::= l | ∈ | e '.' e | '(' e ')' | e '_' e | e '*' | e '+' | e '?'

Εκφράσεις μονοπατιών

• Για τα ονόματα των labels

((s | S)ection|paragraph(s)?

Οι κανονικές εκφράσεις που αφορούν ετικέτες περικλείονται από εισαγωγικά

biblio._*.section.("[tT]itle" | paragraph."*heading.*")

όχι απαραίτητα μόνο χαρακτήρες

Εκφράσεις μονοπατιών

❑ Όταν ένας γράφος έχει ***κύκλους***

- μπορεί να έχουμε μονοπάτια οποιουδήποτε μήκους
- η έκφραση _* ταιριάζει άπειρο αριθμό μονοπατιών (βέβαια το αποτέλεσμα είναι πεπερασμένο)

❑ **Υπολογισμός**

- κατασκευή του πεπερασμένου (μη ντετερμινιστικού αυτόματου) που αντιστοιχεί στην κανονική έκφραση
- ταυτόχρονη διαπέραση του γράφου και του αυτόματου - Closure

Γλώσσες για Ημιδομημένα Δεδομένα

Μια βασική γλώσσα

Μια γλώσσα ερωτήσεων μόνο με εκφράσεις μονοπατιών δεν είναι αρκετή

- δεν μπορούν να κατασκευάσουν νέους κόμβους
- συνένωση
- έλεγχος τιμών

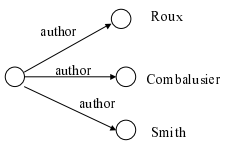
Μια βασική γλώσσα

select author: X
from biblio.book.author X

Αποτέλεσμα:
{author: "Roux", author: "Combalusier", author: "Smith"}

• *bind τη μεταβλητή X με τον κάθε κόμβο που καθορίζεται από την έκφραση μονοπατιού*

• *ένας νέος κόμβος και σύνδεση του με ακμές με ετικέτα author με τους κόμβους που προκύπτουν από τον υπολογισμό της έκφρασης μονοπατιού*

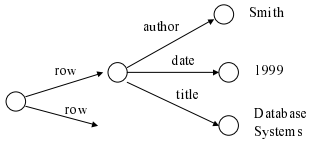


Μια βασική γλώσσα

```
select row: X
from biblio._ X
where "Smith" in X.author
```

Αποτέλεσμα:

```
{row: { author: "Smith",
        date: 1999
        title: "Database Systems",}, ...}
```



• *το αποτέλεσμα στη γενική περίπτωση είναι ένα σύνολο και το κατηγόρημα where ελέγχει τη συμμετοχή στο σύνολο*

Μια βασική γλώσσα

```
select author: Y
from biblio._ X,
     X.author Y,
     X.title Z
where matches (".*(D|d)atabase.*", Z)
```

Αποτέλεσμα:

όλοι οι συγγραφείς με βιβλίο που στον τίτλο του υπάρχει "database" ή "Database"

{author: "Roux", author: "Combalusier", author: "Smith"}

• *κάθε γραμμή του from εισάγει μια καινούργια μεταβλητή*

Μια βασική γλώσσα

```
select E
from B
where C
```

έστω ότι στο B εμφανίζονται οι μεταβλητές X, Y, Z

1ο Βήμα

υπολογισμός του συνόλου των bindings των μεταβλητών που εμφανίζονται στο from

κάθε binding απεικονίζει τις μεταβλητές σε oids (κόμβους) του γράφου

$\{(x_1, y_1, z_1), \ldots (x_n, y_n, z_n)\}$

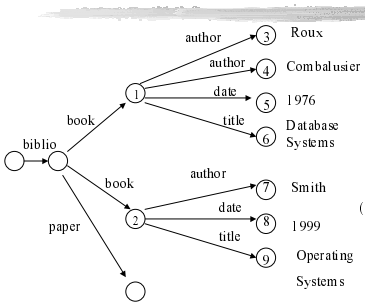
2ο Βήμα

επιλέγουμε όσες ικανοποιούν τη συνθήκη C

3ο Βήμα $\{E(x_1), \ldots, E(x_k)\}$

κατασκευάζουμε το αποτέλεσμα {E( ), E( )....} οι μεταβλητές όπως έχουν γίνει bind

Μια βασική γλώσσα




```
select author: Y
from biblio._ X,
     X.author Y,
     X.title Z
where matches
(".*(D|d)atabase.*", Z)
```

Βήμα 1

(1, 3, 6), (1, 4, 6), (2, 7, 9)

Βήμα 2

(1, 3, 6), (1, 4, 6)

Βήμα 3

{author:3, author:4}

Μια βασική γλώσσα

• *το αποτέλεσμα μπορεί να έχει παραπάνω από έναν κόμβους*

```
select row: {title: Y, author: Z}
from biblio.book X, X.title Y, X.author Z
```

Αποτέλεσμα:

```
{row: { title: "Database Systems", author: "Roux"}
 row: { title: "Database Systems", author: "Combalusier"}
 row: { title: "Database Systems", author: "Smith"}}
```


Μια βασική γλώσσα

• *nesting*

```
select row: (select author: Y
             from X.author Y)
from biblio.book X
```

Αποτέλεσμα:

```
{ row: {author: "Roux", author: "Combalusier"},
  row: {author: "Smith"} }
```

bind X - υπολογισμός του nested select

Μια βασική γλώσσα

• *nesting*

```
select row: (select author: Y, title: T
            from author Y,
                 X.title  T)
from biblio.book   X
where "Roux" in X.author
```

Αποτέλεσμα:

{ row: {author: "Roux", title: "Database Systems"},
row: {author: "Combalusier", title: "Database Systems"}}

Μια βασική γλώσσα

• *joins*

```
{r1: {row  {a:1, b:2},
      row  {a:1, b:3},
r2:  {row{b:2, c:4},
      row:{b:2, c:3}}}
```

```
select  a:A, c:C
from r1.row X
     r2.row Y
     X.a   A, X.b   B, Y.b   B', Y.c   C
where  B = B'
```


Μια βασική γλώσσα

• *joins - πολλαπλές τιμές;*

```
{r1: {row  {a:1, b:2},
      row  {a:1, b:3, b:2},
r2:  {row{b:2, c:4},
      row:{b:2, c:3}}}
```


Lorel

• *παράλειψη των labels*

```
select X
from biblio.book.author X
```

Ένα default label (answer)

{row: "Roux", row: "Combalusier", row: "Smith"}

Lorel

• *εκφράσεις μονοπατιών στο select*

```
select   X.author   ──→   select author: Y
                          from X.author Y
from biblio.book   X
```

Γενικά

```
X.p.l   ──→   select l:Y
              from X.p.l Y
```

Query re-write

Lorel

• *σύγκριση με σύνολα τιμών*

```
select   row: X
from biblio.paper X
where  X.author   =   "Smith"   ──→   where exists Y in X.author ( Y = "Smith")
```

σύνολο τιμών

Lorel

• *μεταβλητές για labels*

```
select L : X
from biblio._*.L  X
where matches(".*Shakespear.*", X)
```

```
{author: "William Shakespear",
 title: "Shakespeare's Tragedies", … }
```


Lorel

```
select new-person: (select L: Y
                    from X.L T
                    where not (L = salary)
drom db.person X
```


Lorel

• *labels ↔ data*

```
select publication: {type : L, title : T}
from biblio.L   X,   X.title  T
from X.Year > 1989
```

```
{publication: {type:"book",
               title: "Database Systems"},
 publication: {type: "paper",
               title: "Semistructured data"} …"
```


Lorel

• *group-by*

```
select  Y : (select X
             from biblio.paper  X
             where X.Year = Y)
from biblio.paper.year Y
```

• *duplicate elimination*

Lorel

• *paths ↔ data*

```
select  @P
from db1  @P.X
where matches (.,  X)
```

• *άπειρο μήκος - άπειρα μονοπάτια*

Σημασιολογία

Λογική 1ης Τάξης

Μοντελοποίηση του γράφου ως μια πεπερασμένη δομή σχέσης

Τύπος **oid** για ονόματα μεταβλητών

Sort **dom** για ατομικές τιμές (έστω μόνο strings) και ετικέτες

Έστω ότι τα σύνολα dom και oid είναι μετρήσιμα άπειρα και ξένα

Λογική 1ης Τάξης

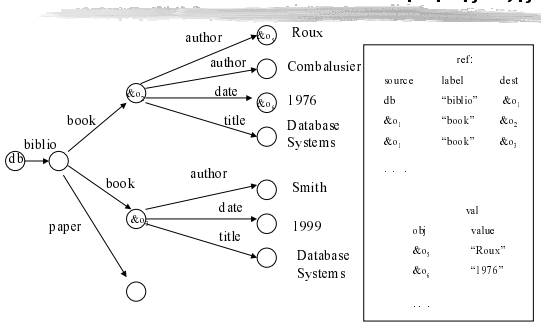
- Ένας γράφος είναι ένα στιγμιότυπο (instance) του σχήματος:
  - ref (source:oid, label:dom, destination: oid)
  - val (obj: oid, value:dom)

ref(o_1, l, o_2) : o_1 και o_2 είναι oids ενός σύνθετου αντικειμένου o και υπάρχει μια ακμή με ετικέτα l ανάμεσα τους

val (obj: oid, value:dom) : o είναι ατομικό με τιμή v

- σταθερές που δηλώνουν τα oids κάποιων κόμβων -- roots of persistence

Λογική 1ης Τάξης



Λογική 1ης Τάξης

Περιορισμοί

1. Ατομικό/σύνθετο (δεν υπάρχει oid που να εμφανίζεται και στην πρώτη στήλη του ref και στο val)

2. Reachability (για κάθε αντικείμενο o υπάρχει ένα κατευθυνόμενο μονοπάτι από κάποια root of persistence r

3. Κλειδί (το γνώρισμα oid είναι κλειδί της σχέσης val)

4. Κλειδί (το γνώρισμα value είναι κλειδί της σχέσης val) -- προαιρετικά

Λογική 1ης Τάξης

```
select author: X
from biblio.book Y,
     Y.author X
where "Database Systems" in Y.title
```

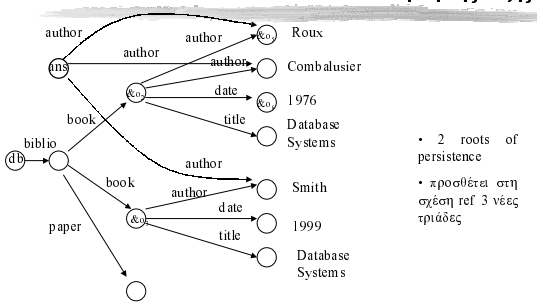
{X | ∃ Y, Z, V, W (ref(db, "biblio", W) ∧
ref(W, "book", Y) ∧
ref(Y, "author', X) ∧
ref(Y, "title", V) ∧
val(V, "Database Systems))}

Λογική 1ης Τάξης

rule

ref(ans, "author", X) ← ref(db, "biblio", W),
ref(W, "book", Y),
ref(Y, "author', X),
ref(Y, "title", V),
val(V, "Database Systems")

Λογική 1ης Τάξης



- 2 roots of persistence
- προσθέτει στη σχέση ref 3 νέες τριάδες

Λογική 1ης Τάξης

```
select title: X
from biblio.paper.section.(title | paragraph.heading) X

ref(ans, "title", X) ← section(Y),
                       ref(Y, "title", X)
ref(ans, "title", X) ← section(Y),
                       ref(Y, "paragraph", Z),
                       ref(Z, "heading", X)
      section(Y) ← ref(db, "biblio", W),
                   ref(W, "paper", V),
                   ref(V, "section", Y)
```


Λογική 1ης Τάξης

- **Kleene closure (*) -- αναδρομή**

```
select part: Y
from product X, X.(subpart)* Y

ref(ans, "part", Y) ← q(Y)
          q(Y) ← ref(db, "product", Y)
          q(Y) ← q(Z)
                 ref(Z, "subpart", Y)
```


Δημιουργία Αντικειμένων

```
select row: {title: T, year: Y}
from biblio.book X,
     X.title T,
     X.year Y

ref(ans, "row", f(X)) ← ref(db, "biblio", B),
                        ref(B, "book", X)
  ref(f(X), "title", T) ←ref(X, "title", T)
  ref(f(X), "year", Y) ←ref(X, "year", Y)
```

Object identifiers σε κάθε γραμμή

tag f: $f(o_1), f(o_2), \ldots,$

Skolem function

- **Fusion queries**

Δημιουργία Αντικειμένων

Fusion queries -- Joins

```
select row: {num: N,  title: Y}
where {number: N, title: Y} in db1.report

select row: {num: N,  postscript: P}
where {num: N, postscript P} in db2.myrep
```

Για κάθε report (με βάση τον αριθμό του num(ber)) θέλουμε όλες τις διαθέσιμες πληροφορίες

```
select report: {number: N,
                (select  title: T
                  where {number: N, title T} in db1.report),
                  (select  postscript: P
                  where {num: N, postscript: P} in db2.myrep),
 where N in db1.report.number union db2.myrep.num
```


Δημιουργία Αντικειμένων

Fusion queries -- Joins

```
ref(ans, "row", rep(N)) ← ref(db1, "report", X),
                          ref((X,  "number", Y),
                          val(Y, N)
ref(ans, "row", rep(N)) ← ref(db2, "myreport", X),
                          ref((X,  "num", Y),
                          val(Y, N)
   ref(rep(N), "title", T) ← val(Y, N),
                          ref(X, "number", Y),
                          ref((X,  "title", T)
ref(rep(N), "postscript",P) ←   val(Y, N),
                                ref(X, "num", Y),
                                ref(X,  "postscript", P)
```

Δομική Αναδρομή

{biblio: { book: {author: "Roux", author: "Combalusier", date: 1976},
book: {author: "Smith", date: 1999, title: "Database Systems"},
paper: {title: "Data Protection", author: "Casio"}}}

• *Σύνολα, π.χ.*, {author: "Roux", author: "Combalusier", date: 1976} -- σύνολο με 3 τρία στοιχεία

• *ένωση π.χ.*, t1 = {author: "Roux", author: "Combalusier", date: 1976} και
t2 = {author: "Smith", date: 1999, title: "Database Systems"},
t1 union t2 = {author: "Roux", author: "Combalusier", date: 1976, author: "Smith", date: 1999, title: "Database Systems"},}

Δομική Αναδρομή

Αναδρομικές συναρτήσεις f με τέσσερις περιπτώσεις

1. ατομικές τιμές
2. { }
3. {l: t}
4. t1 union t2

By default, f({}) = {} και f(t1 union t2) = f(t1) union f(t2)

• *ΠΑΡΑΔΕΙΓΜΑ : Όλοι οι ακέραιοι σε μια βάση δεδομένων*

f1(v) = if isInt(v) then {result: v} else {}
f1({}) = {}
f1({l: t}) = f1(t)
f1(t1 union t2) = f1(t1) union f1(t2)

Δομική Αναδρομή

• *ΠΑΡΑΔΕΙΓΜΑ : Μετατροπή όλων των συμβολοσειρών σε ακεραίους*

f2(v) = if isInt(v) then int2String(v) else v
f2({}) = {}
f1({l: t}) = {l: f2(t)}
f2(t1 union t2) = f2(t1) union f2(t2)

Δομική Αναδρομή

```
select answer: X
from *.paper.((section.subsection) | paragraph)    X
```

f4(v) = { }
f4({ l : t }) = if l = paper then f4(t) union g4(t)
else f4(t)

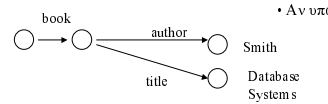
g4(v) = { }
g4({ l : t }) = if l = section then h4(t)
else if l = paragraph then {answer: t}
else { }

h4(v) = {}
h4({ l : t }) = if l = subsection then {answer: t} else { }

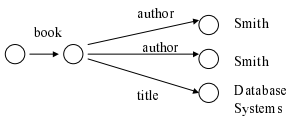
Bisimulation

• Απαλοιφή διπλοτύπων;

• Αν υπάρχουν oids;



t1 = {book: {author: "Smith", title: "Database Systems"}}



t2 = {book: {author: "Smith", author: "Smith", title: "Database Systems"}}

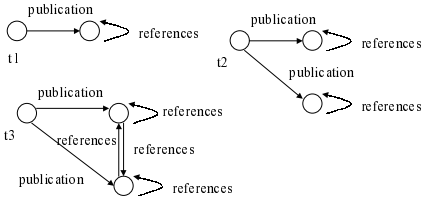
Bisimulation

• Από τη ρίζα προς την κορφή

{person: { name: "John",
phone: 8776,
phone: 8776},
person: { name: "Sue",
office: A471},
person: { name: "John"
name: "John",
phone: 8776}}

Bisimulation

- Όταν υπάρχουν κύκλοι;
- Αναδρομή;



Bisimulation

Bisimulation: δυαδική σχέση μεταξύ των κόμβων δύο γράφων t1 και t2 (έστω x, x' κόμβοι του t1 και y, y' κόμβοι του t2)

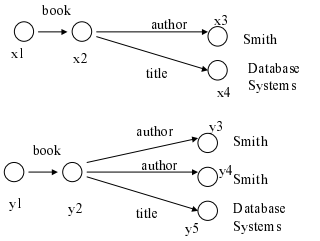
1. Αν x και y είναι ρίζες του t1 και t2 τότε x ~ y.

2. Αν x ~ y και ένας εκ των x και y είναι ρίζα, τότε και ο άλλος κόμβος είναι ρίζα

3. Αν x ~ y και (x, l, x') στο t1 τότε υπάρχει ακμή (y, l, y') στο t2 με την ίδια ετικέτα και x' ~ y'Αντίστοιχα, αν x ~ y και (y, l, y') στο t2 τότε υπάρχει ακμή (x, l, x') στο t1 με την ίδια ετικέτα και x'~ y'

4. Αν x ~ y και το x είναι φύλλο με τιμή v στο t1, τότε και το y είναι φύλλο με τιμή v στο t2. Αντίστοιχα, αν x ~ y και το y είναι φύλλο με τιμή v στο t2, τότε και το x είναι φύλλο με τιμή v στο t1.

Bisimulation



x1 ~ y1 x3 ~ y3 x3 ~ y4

x4 ~ y5 x2 ~ y2

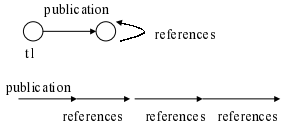
Bisimulation

Αλγόριθμος

- Ξεκίνα με το σύνολο {(x, y) | x in t1, y in t2}
- βγάλε από το σύνολο όσα ζεύγη δεν ικανοποιούν τον ορισμό
- bisimilar αν στο τέλος (t1, t2) ανήκουν στο σύνολο)

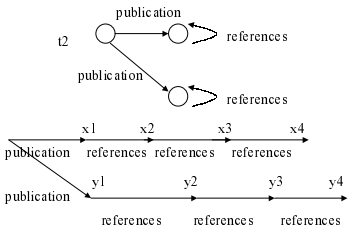
Bisimulation

unfolding



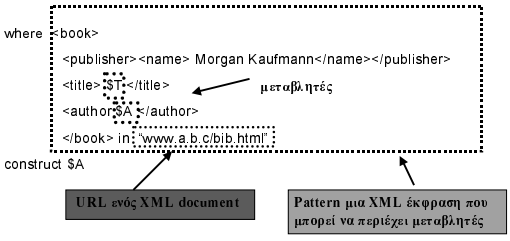
Bisimulation

unfolding



Γλώσσες για XML

- path expressions and patterns (in XML syntax) to extract data from the input
- variables to which this data is bound
- templates - how to construct the output



περιγράφει μόνο το τι πρέπει να υπάρχει υποχρεωτικά


```
where  <book>
          <publisher><name> Morgan Kaufmann</></>
          <title> $T </>
          <author $A </>
          </book> in "www.a.b.c/bib.html"
construct <result>
             <author> $A </>
             <title> $T </>
           </>
```



```
<book year="1991>
          <publisher><name> Morgan Kaufmann</name></publisher>
          <title> An Introduction to Parallel Algorithms and Architectures</title>
          <author> <lastname> Leighton </lastname> </author>
</book>
<book year="1995>
          <publisher><name> Morgan Kaufmann</name></publisher>
          <title>Active Database Systems</title>
          <author> <lastname> Ceri</lastname></lastname> </author>
          <author> <lastname> Widom</lastname> </author>
</book>
```



```
<result>
          <author> <lastname> Leighton </lastname> </author>
          <title> An Introduction to Parallel Algorithms and Architectures</title>
</result>
<result>
           <author> <lastname> Ceri</lastname></lastname> </author>
          <title>Active Database Systems</title>
</result>
<result>
          <author> <lastname> Widom</lastname> </author>
          <title>Active Database Systems</title>
</result>
```

XML-QL

• *προαιρετικά στοιχεία*

πως μπορούμε να διατυπώσουμε την ερώτηση: τους τίτλους όλων των βιβλίων και αν είναι διαθέσιμη και την τιμή τους

```
where  <book>
          <title> $T </>
          <price> $A </>
          </book> in "www.a.b.c/bib.html"
construct <result>
                <booktitle> $T </>
                <bookprice> $A </>
          </>
```


XML-QL

• *προαιρετικά στοιχεία & χρήση φωλιασμένων ερωτήσεων*

πως μπορούμε να διατυπώσουμε την ερώτηση: τους τίτλους όλων των βιβλίων και αν είναι διαθέσιμη και την τιμή τους

```
where  <book> $B$ </book> in  "www.a.b.c/bib.html"
          <title> $T </>  in $B
construct <result>
                <booktitle> $T </>
                where <price> $P </> in $B
                construct <bookprice> $P </>
          </>
```

Μεταβλητή (που αναφέρεται σε ένα κομμάτι URL ενός XML document)

Nested query (υπολογίζεται για κάθε binding της μεταβλητής B)

XML-QL

• ***element_as*** *: bind a variable to an element*

```
where  <book> <publisher><name> Morgan Kaufmann  </> </>
       </>
          element_as $B in "abc.xml"
construct $B
```

μεταφράζεται σε:

```
where  <book> $T </> in "abc.xml"
          <publisher><name> Morgan Kaufmann  </> </> in $T
construct  <book> $T </book>
```


XML-QL

• ***content_as*** *: bind a variable to an element's content*

```
where  <book> <publisher><name> Morgan Kaufmann  </> </>
       </>
          content_as $C in "abc.xml"
construct <result> C </result>
```

μεταφράζεται σε:

```
where  <book> $C </> in "abc.xml"
          <publisher><name> Morgan Kaufmann  </> </> in $C
construct  <result> $C </result>
```


XML-QL

• *γνωρίσματα*

```
where  <book language="French">
          <title></> element_as $T
       </> in "abc.xml"
construct  $T
```

```
where  <book language = $L> </> in "abc.xml"
construct  <result> $L </result>
```


XML-QL

• *μεταβλητές tags*

```
where  <$P>  <title> $T </title>
             <year> 1995 </>
             <$E> Smith </>
       </> in "www.a.b.c/bib.xml",
       $E in {author, editor}
construct <$P>  <title> $T </title>
             <$E> Smith </>
       </>
```

XML-QL

• *κανονικές εκφράσεις μονοπατιών*

<part*>
<$*> $ wild card
<*.brand> . Concatenation
|

• *διαφορά από κανονικές εκφράσεις σε DTDs*

XML-QL

• *διάταξη*

• *δυο εκδοχές της γλώσσας*

Stylesheets

Γραμματικές ειδικού σκοπού για την μετατροπή XML documents σε HTML documents ώστε να είναι δυνατή η αναπαράστασή τους

Δύο προτάσεις

- Cascading Style Sheets
- Extensible Stylesheet Language XSL

Η βασική ιδέα είναι να αντιστοιχήσουμε με κάθε τύπο στοιχείου μια αναπαράσταση

XSL

Extensible **S**tylesheet **L**anguage XSL: μια γλώσσα για μετασχηματισμό και μορφοποίηση XML documents

XSLTransformation Language (XSLT)

XSLTransformation Language (XSLT) Stylesheet: μια σειρά από κανόνες μετασχηματισμού (**transformation rules**) που παίρνουν ως είσοδο ένα πηγαίο XML document (source tree) και παράγουν ένα νέο XML document (result tree)

XSL

• Ένα XSL πρόγραμμα είναι ένα σύνολο από κανόνες

• Κάθε κανόνας αποτελείται από ένα πρότυπο (**pattern**) (– where) και ένα **template** (–construct)

• Τα patterns ταιριάζουν με τους κόμβους του πηγαίου δέντρου, ενεργοποιούνται οι αντίστοιχες templates

• Η επεξεργασία αρχίζει από τη ρίζα και αναζητείται ένα pattern που ταιριάζει τη ρίζα - όταν βρεθεί εκτελείται το αντίστοιχο template , το οποίο συνήθως αφορά την παραγωγή κάποιου XML αποτελέσματος και την αναδρομική εφαρμογή του templates στα παιδιά του κόμβου

XSL

```
<xsl:stylesheet xmlns:xsl="http://w3.org/XSL?transform/1.0"
                xmlns="http://w3.org/TR/xhtml1"
                indent-result="yes">
<! -- Rule 1 ---> <xsl:template match = "/">
                  <html><head><title>Our New Catalog</title><head>
                        <body>
                            <xsl:apply-templates/>
                        </body>
                  </html>
                </xsl:template>
```

πρότυπο (pattern)

XSL

XSL και δομική αναδρομή

XML-QL

<!ELEMENT bib (book | article)*>
<!ELEMENT book (author+, title, publisher)>
<!ATTLIST book year PCDATA>
<!ELEMENT article (author+, title, year?, (shortversion | longversion))>
<!ATTLIST article type PCDATA>
<!ELEMENT publisher (name, address)>
<!ELEMENT author (firstname?, lastname)>

Γλώσσες Ερωτήσεων για XML

Απαιτήσεις

1. Ακριβής Σημασιολογία (precise semantics)

ισοδυναμία (για βελτιστοποίηση ερωτήσεων) -- περιεκτικότητα (containment: π.χ., για caching)

2. Rewritability -- Optimizability

σχεσιακά, αντικειμενοστραφή μοντέλα κ.λ.π. σε XML: μια XML ερώτηση εύκολα μετατρέψιμη στη γλώσσα ερωτήσεων του αρχικού μοντέλου (αντί μετατροπής των αρχικών δεδομένων σε XML)

αρχικά δεδομένα σε XML: δυνατότητα για βελτιστοποίηση ερωτήσεων

Γλώσσες Ερωτήσεων για XML

3. XML Output

Το αποτέλεσμα μιας XML ερώτησης πρέπει να είναι XML δεδομένα

Γλώσσες Ερωτήσεων για XML

4. Τελεστές Ερωτήσεων

- *selection* (επιλογή ενός document με βάση το περιεχόμενο, τη δομή ή τα γνωρίσματα)
- *extraction* (εξαγωγή συγκεκριμένων στοιχείων ενός document)
- *reduction* (απομάκρυνση συγκεκριμένων υποστοιχείων ενός στοιχείου)
- *restructuring* (κατασκευή νέου συνόλου στοιχείων για τα υπο-εξέταση δεδομένα)
- *combination* (συγχώνευση ενός ή περισσότερων στοιχείων σε ένα)

σε *μία* XML ερώτηση

5. Διατήρηση της Διάταξης και των Συσχετίσεων (association)

- διατήρηση της διάταξης των στοιχείων σε κάθε XML document
- διατήρηση της ομαδοποίησης των υποστοιχείων σε κάθε στοιχείο

XML-QL

```
where <bib> <book year = $y>
        <publisher><name> Morgan Kaufmann</name></publisher>
        <title> $T </title>
        <author> $A </author>
      </book></bib> in "www.a.b.c/bib.html", $y > 1991
construct $A
```

extraction μέσω μεταβλητών

select μέσω patterns και συνθηκών

XML-QL

```
where <bib> <book year = $y>
          <publisher><name> Morgan Kaufmann</></>
          <title> $T </>
          <author> $A </>
      </book></bib> in "www.a.b.c/bib.html", $y > 1991
construct <result> <author> $A </>
                   <title> $T </>
          </>
```

restructuring: μέσω του template στο construct

reduction: καθορίζοντας ποια στοιχεία θα επιστραφούν στο construct

- **επίσης, διατήρηση των συσχετίσεων ζεύγη (author, title) όπως στο XML document**

XML-QL

Πιο περίπλοκο restructuring μέσω nested queries

```
where <bib> <book> <title> $T </>
            <publisher><name> Morgan Kaufmann</></>
      </> content_as $P </> in "www.a.b.c/bib.html
construct <result> <title> $T </>
                   where <author> $A </> in $P
                   construct <author> $A </>
          </>
```


XML-QL

Πιο περίπλοκο restructuring μέσω nested queries - συνέχεια

πως μπορούμε να διατυπώσουμε την ερώτηση: όλους τους συγγραφείς και για κάθε συγγραφέα τα βιβλία που έγραψε

```
where  <bib> <book> <author> $A </> </> in "www.a.b.c/bib.html"
construct <result>
                <author> $A </>
                where <bib> <book> <author> A </A>
                            <title> $T </>
                      </> in "www.a.b.b/bib.html",
                construct <title> $T </>
          </>
```


XML-QL

combination -- συνένωση

```
where  <bib><book><author> $A  </></></>
          content_as $B1 in "abc.xml",
       <bib><book><author> $A  </></></>
          content_as $B2 in "abc.xml",
       B1 != B2
construct <result> $A </result>
```


XML-QL

combination -- συνένωση

```
<!ELEMENT reviews (entry)*>
<!ELEMENT entry  (title, review)>
<!ELEMENT review (#PCDATA)>

where  <bib> <book> <title> $T </> <publisher> $P </> </>
          </>  in "www.a.b.c/bib.xml"
       <reviews> <entry> <title>  $T </> <review> $R </> </>
          </>  in "www.a.b.c/reviews.xml"
construct <book> <title> $T </> <publisher> $P </> </> <review> $R </> </>
```


XML-QL

combination με συναρτήσεις Skolem

```
{ where  <bib> <book> <title> $T </> <publisher> $P </> </> </>  in "www.a.b.c/bib.xml"
  construct <book ID=f($T)> <title> $T </> <publisher> $P </> </> </>
}
{ where <reviews> <entry> <title>  $T </> <review> $R </> </></>  in "www.a.b.c/reviews.xml"
  construct <book ID=f($T)> <title> $T </>   <review> $R </> </>
}
```

- ανεξάρτητη εκτέλεση των δυο ερωτήσεων - αν αγνοήσουμε τη συνάρτηση Skolem ID = f($T) παράγονται δυο ξένα σύνολα
- το γνώρισμα ID είναι τύπου ID και αναθέτει ένα μοναδικό κλειδί σε κάθε στοιχείο
- η συνάρτηση Skolem ελέγχει την ανάθεση των IDs --- f : ένα νέο ID για κάθε binding του $T
- στη δεύτερη ερώτηση όταν δίνει ένα παλιό ID άνει append τα υποστοιχεία title και review στο υπάρχον στοιχείο

Γλώσσες Ερωτήσεων για XML

6. Compositional Semantics

• οι εκφράσεις της XML γλώσσας ερωτήσεων πρέπει να υποστηρίζουν **referential transparency**: η σημασία (meaning) μιας έκφρασης πρέπει να είναι η ίδια όπου και αν εμφανίζεται μια έκφραση

• επίσης οι εκφράσεις με ίσο (equal) τύπο αποτελέσματος θα πρέπει να επιτρέπεται να εμφανίζονται στο ίδιο context -- ιδιαίτερα, όπου επιτρέπεται η εμφάνιση ενός XML term θα πρέπει να επιτρέπεται και η εμφάνιση μιας έκφρασης που επιστρέφει έναν XML term

7. Η Ύπαρξη Σχήματος δεν Είναι Απαραίτητη

Δεν είναι απαραίτητη η από την αρχή γνώση κάποιου σχήματος (DTD)

8. Χρησιμοποίηση Υπάρχοντος Σχήματος

Όταν υπάρχει DTD, θα πρέπει να πρέπει να είναι δυνατόν να ελεχθεί αν η ερώτηση είναι διατυπωμένη σωστά αναφορικά με το DTD καθώς και ο υπολογισμός του DTD του αποτελέσματος

XML-QL

Η ύπαρξη ή η γνώση του σχήματος δεν είναι απαραίτητη

Με χρήση:

- tag variables
- κανονικές εκφράσεις μονοπατιών

```
<!ELEMENT part (name, brand, part*>
<!ELEMENT name (#PCDATA)>
<!ELEMENT brand (#PCDATA)>

where  <(part)*> <name>  $R </> <brand> Ford </> </>  in "www.a.b.c/bib.xml"
construct  <result> $R </>
```


Γλώσσες Ερωτήσεων για XML

9. Mutually Embedding with XML

• μια XML ερώτηση θα πρέπει να μπορεί να περιέχει οποιαδήποτε XML δεδομένα (σταθερές, μερικός υπολογισμός μιας ερώτησης)

• ένα XML document θα πρέπει να μπορεί να περιέχει οποιαδήποτε XML ερώτηση (ένα XML document θα μπορεί να περιέχει και αποθηκευμένα και ιδεατά δεδομένα)

10. Υποστήριξη Νέων Τύπων Δεδομένων

Θα πρέπει να υπάρχει ένας μηχανισμός που θα επιτρέπει επέκταση της γλώσσας για την υποστήριξη συνθηκών και πράξεων για νέους τύπους δεδομένων (π.χ., πολυμέσα)

Γλώσσες Ερωτήσεων για XML

11. Κατάλληλη για Μεταδεδομένα

μια XML γλώσσα θα πρέπει να είναι χρήσιμη ως κομμάτι προσδιορισμού μεταδεδομένων

12. Υπολογισμός στη Μεριά του Εξυπηρέτη

13. Χειρισμός μέσω Προγραμμάτων

δυνατότητα διατύπωσης ερωτήσεων μέσω user-interfaces και εργαλείων

14. XML Αναπαράσταση

Μια XML ερώτηση θα πρέπει να μπορεί να αναπαρασταθεί σε XML -- ως XML δεδομένα - ώστε να μη χρειάζονται ειδικοί μηχανισμοί για την αποθήκευση και τη μεταφορά XML ερωτήσεων

Γλώσσες Ερωτήσεων για XML

	XML-QL	XSL	LOREL
Precise semantics	Y	Y	Y
Re-writability	Y	N	N
XML output	Y	Y	N
Server procedures	Y	Y	Y
All query operations	Y	N	Y
Compq semantics	Y	Y	N
No schema req.	Y	Y	Y
Exploiting schema	N	N	N
Preserve order	Y	Y	Y
Programming manipulation	Y	Y	Y
XML represenation	N	Y	N
XML embedded	Y	Y	Y
New data types	N	N	N
Metadata	Y	Y	Y

Γλώσσες για το web

Γλώσσες για το Web 1ης Γενιάς

WebSQL

- Μοντελοποίηση του *web ως μια σχεσιακή βάση δεδομένων* με δύο (ιδεατές) σχέσεις:
  - documents και
  - anchors
- Η σχέση *document* έχει μια πλειάδα για κάθε document στο web
- Η σχέση *anchor* έχει μια πλειάδα για κάθε anchor σε κάθε document στο web

Γλώσσες για το Web 1ης Γενιάς

WebSQL

Navigation από ένα γνωστό URL

Εκφράσεις μονοπατιών

- d1 → d2 (d1 δείχνει στο d2 και τα δύο είναι αποθηκευμένα στον ίδιο server)
- d1 ⇒ d2 (d1 δείχνει στο d2 τα οποία είναι αποθηκευμένα σε διαφορετικούς servers)

Γλώσσες για το Web 1ης Γενιάς

WebSQL

Παράδειγμα: τις τριάδες (d1, d2, label) όπου d1 είναι ένα document αποθηκευμένο τοπικά, d2 κάπου αλλού και το d1 δείχνει στο d2 με ένα link με ετικέτα label

```
select d.url, e.url, a.label
from Document d such that "www.mysite.start" →* d,
     Document e such that d ⇒ e,
     Anchor a such that a.base = d.url
where a.href = e.url
```

Επίσης με βάση το περιεχόμενο

```
d mentions "database"
```


Γλώσσες για το Web 2ης Γενιάς

- Χειρίζονται τις web σελίδες ως ατομικά αντικείμενα με δύο ιδιότητες:
  - περιέχουν συγκεκριμένο κείμενο
  - δείχνουν σε συγκεκριμένα αντικείμενα
- Γλώσσες 2ης Γενιάς
  - επιτρέπουν προσπέλαση στη δομή των web αντικειμένων που χειρίζονται
  - επιτρέπουν την δημιουργία νέων σύνθετων δομών ως αποτέλεσμα των ερωτήσεων

Γλώσσες για το Web 2ης Γενιάς

WebOQL

Υπερ-δέντρο (hypertree)

Διατεταγμένο δέντρο με ετικέτες στις ακμές και δυο ειδών ακμές

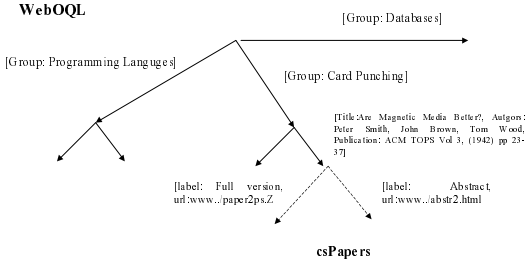
- εσωτερικές (που περιγράφουν τη δομή των αντικειμένων) και
- εξωτερικές (που περιγράφουν συνδέσεις - hyperlinks - ανάμεσα στα αντικείμενα)

οι ετικέτες στις ακμές είναι εγγραφές

Σύνολα από σχετιζόμενα υπερ-δέντρα - web

Γλώσσες για το Web 2ης Γενιάς

WebOQL

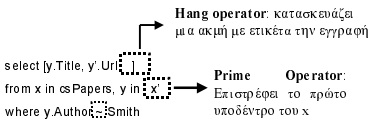


csPapers

Γλώσσες για το Web 2ης Γενιάς

WebOQL

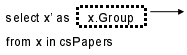
Full version and url όλων των papers του Smith



Γλώσσες για το Web 2ης Γενιάς

WebOQL

• *Δημιουργία web*



Μια ερώτηση είναι μια συνάρτηση που απεικονίζει ένα web σε ένα άλλο

Γλώσσες για το Web 2ης Γενιάς

WebOQL

• *navigational patterns*

```
select [x.Url, x. Text]
from x in browse ("root.html")
     via (^*[Text ~ "Next"]>)*
```