XML and Databases

Exam Preparation
Part 1

Sebastian Maneth
NICTA and UNSW

CSE@UNSW -- Semester 1, 2010

(1)**[4]** Consider this DTD:

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

This DTD is included in each of the following. Say for each whether or not it is well-formed XML (with respect to the DTD!).
If it is not well-formed, explain ***all*** violations that you can find.

```
a) <bib><book></book></bib>
b) <bib><journal isbn=”xyz”><author/><title/></journal></bib>
c) <bib><book isbn=”123”><author/><title/></book><journal><author/><title/>
<cites><book isbn=”123”><author/><title/><book/></cites></journal></bib>
d) <bib book=”isbn”></bib>
e) <bib>no entries</bib>
f) <bib><journal><author/><title/><!—- all empty>>->--></bib>
g) <bib></bib><bib></bib>
h) <bib><author></author><title></title></Bib>
```

(1)**[4]** Consider this DTD:

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

This DTD is included in each of the following. Say for each whether or not it is well-formed XML (with respect to the DTD!).
If it is not well-formed, explain ***all*** violations that you can find.

a) <bib><book></book></bib>

(1)**[4]** Consider this DTD:

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

This DTD is included in each of the following. Say for each whether or not it is well-formed XML (with respect to the DTD!).
If it is not well-formed, explain ***all*** violations that you can find.

a) `<bib><book></book></bib>` Not well formed!

- → book must have `author` and `title` children
- → book must have `isbn` attribute

(1)**[4]** Consider this DTD:

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

This DTD is included in each of the following. Say for each whether or not it is well-formed XML (with respect to the DTD!).
If it is not well-formed, explain ***all*** violations that you can find.

b) <bib><journal isbn=”xyz”><author/><title/></journal></bib>

(1)**[4]** Consider this DTD:

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

This DTD is included in each of the following. Say for each whether or not it is well-formed XML (with respect to the DTD!).
If it is not well-formed, explain ***all*** violations that you can find.

b) `<bib><journal isbn=”xyz”><author/><title/></journal></bib>`

Not well formed!

→ `journal` must not have an attribute

(1)**[4]** Consider this DTD:

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

This DTD is included in each of the following. Say for each whether or not it is well-formed XML (with respect to the DTD!).
If it is not well-formed, explain ***all*** violations that you can find.

```
c) <bib><book isbn=”123”><author/><title/></book><journal><author/><title/>
<cites><book isbn=”123”><author/><title/><book/></cites></journal></bib>
```

(1)**[4]** Consider this DTD:

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

This DTD is included in each of the following. Say for each whether or not it is well-formed XML (with respect to the DTD!).
If it is not well-formed, explain ***all*** violations that you can find.

```
c) <bib><book isbn=”123”><author/><title/></book><journal><author/><title/>
<cites><book isbn=”123”><author/><title/><book/></cites></journal></bib>
```

Not well formed!

→ wfc is violated (independent of the DTD)

→ ID attribution is violated: `isbn` may not have value “123” for different elements

(1)**[4]** Consider this DTD:

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

This DTD is included in each of the following. Say for each whether or not it is well-formed XML (with respect to the DTD!).
If it is not well-formed, explain ***all*** violations that you can find.

d) <bib book=”isbn”></bib>

(1)**[4]** Consider this DTD:

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

This DTD is included in each of the following. Say for each whether or not it is well-formed XML (with respect to the DTD!).
If it is not well-formed, explain ***all*** violations that you can find.

d) `<bib book=”isbn”></bib>`

Not well-formed.

→ `bib` must not have an attribute.

(1)**[4]** Consider this DTD:

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

This DTD is included in each of the following. Say for each whether or not it is well-formed XML (with respect to the DTD!).
If it is not well-formed, explain ***all*** violations that you can find.

e) <bib>no entries</bib>

(1)**[4]** Consider this DTD:

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

This DTD is included in each of the following. Say for each whether or not it is well-formed XML (with respect to the DTD!).
If it is not well-formed, explain ***all*** violations that you can find.

e) `<bib>no entries</bib>`

Not well-formed.

→ `bib` must not have text-content

(1)**[4]** Consider this DTD:

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

This DTD is included in each of the following. Say for each whether or not it is well-formed XML (with respect to the DTD!).
If it is not well-formed, explain ***all*** violations that you can find.

```
f) <bib><journal><author/><title/><!–- all empty>>->--></bib>
```

(1)**[4]** Consider this DTD:

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

This DTD is included in each of the following. Say for each whether or not it is well-formed XML (with respect to the DTD!).
If it is not well-formed, explain ***all*** violations that you can find.

```
f) <bib><journal><author/><title/><!–- all empty>>->--></bib>
```

Not well-formed!
→ wfc violated: no matching end-tag for `journal`

Question can a DTD say something about comments
or processing instructions?

(1)**[4]** Consider this DTD:

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

This DTD is included in each of the following. Say for each whether or not it is well-formed XML (with respect to the DTD!).
If it is not well-formed, explain ***all*** violations that you can find.

g) <bib></bib><bib></bib>

(1)**[4]** Consider this DTD:

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

This DTD is included in each of the following. Say for each whether or not it is well-formed XML (with respect to the DTD!).
If it is not well-formed, explain ***all*** violations that you can find.

g) `<bib></bib><bib></bib>`

Not well-formed!

Question is it a wfc-violation, or a violation to the XML grammar??

(1)**[4]** Consider this DTD:

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

This DTD is included in each of the following. Say for each whether or not it is well-formed XML (with respect to the DTD!).
If it is not well-formed, explain ***all*** violations that you can find.

h) <bib><author></author><title></title></Bib>

(1)**[4]** Consider this DTD:

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

This DTD is included in each of the following. Say for each whether or not it is well-formed XML (with respect to the DTD!).
If it is not well-formed, explain ***all*** violations that you can find.

h) `<bib><author></author><title></title></Bib>`

Not well-formed!

→ wfc violated: `/Bib` does not mach `bib`

→ `bib` may only have book or journal children (not author or title)

(1)**[4]** Consider this DTD:

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

(2)**[4]** Consider again the DTD from number (1).
If a journal or book subtree appears below a cites-node, then we say that this journal or book is being cited.

(a) Write pseudo code that uses DOM and prints each journal and book that is being cited, together with the number of times it is cited.

(b) Is it possible, with the DTD of (1), that a book cites itself? Explain. Do you see a better way of citing, using attributes? How?

Write pseudo code that uses DOM and prints each journal and book that is being cited, together with the number of times it is cited.

Idea: recursive traversal, if cite-node, then check for children if in HashMap, if not, add it with value 1.

```
void Traverse(Node n, HashMap result){
  Nodelist children = n->childrenList();
  if (n-> type==“Element” && n->name==“cites”){
    for each Node c in children call addResult(c, result)
  }
  if (!children→isEmpty()){
    for each Node c in children call Traverse(c, result)
  }
}
void addResult(node c, hashmap result){
  if(result.contains(c)){
    int temp=result.get(c)+1;
    result→put(c,temp);
  }  else result→put(c,1)
}

void main{
  result=new HashMap<String,int>;
  Traverse(root, result)
  Print(result)
}
```

Note
sorry for the confusion during the lecture, this code *is* correct:
For all children of cites nodes,
we count (add into the hash-map)!

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

(2)**[4]** Consider again the DTD from number (1).
If a journal or book subtree appears below a cites-node, then we say that this journal or book is being cited.

(b) Is it possible, with the DTD of (1), that a book cites itself?
Explain. Do you see a better way of citing, using attributes? How?

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

(2)**[4]** Consider again the DTD from number (1).
If a journal or book subtree appears below a cites-node, then we say that this journal or book is being cited.

(b) Is it possible, with the DTD of (1), that a book cites itself?
Explain. Do you see a better way of citing, using attributes? How?

No, a book cannot cite itself! (has no cite-child)
But needs to repeat precisely, the author title info.
Better would be to give

→ each book and journal a unique ID (in form of an attribute)

→ use an attribute “cites” and have

```
<book isbn=“9876-345”><author>…</author>
      <title>…</title>
      <cites citeID=“1234-789”/><cites citeID=“9876-345”/>
      … <cites citeID=“9999-666”/>
</book>
```

```
<!DOCTYPE bib [
  <!ELEMENT bib (book | journal)*>
  <!ELEMENT book (author, title)>
  <!ELEMENT journal (author, title, cites?)>
  <!ELEMENT cites (book | journal)*>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ATTLIST book isbn ID #REQUIRED>
]>
```

(2)**[4]** Consider again the DTD from number (1).
If a journal or book subtree appears below a cites-node, then we say that this journal or book is being cited.

(b) Is it possible, with the DTD of (1), that a book cites itself?
Explain. Do you see a better way of citing, using attributes? How?

No, a book cannot cite itself! (has no cite-child)
But needs to repeat precisely, the author title info.
Better would be to give

- → each book and journal a unique ID (in form of an attribute)
- → use an attribute “cites” and have

```
<book isbn=“9876-345”><author>…</author>
      <title>…</title>
      <cites citeID=“1234-789”/><cites citeID=“9876-345”/>
      … <cites citeID=“9999-666”/>
</book>
```

Question

Show correct DTD-rules with:

- → isbn of type ID (done)
- → citeID of type IDREF

(5)**[8]** Again, under the DTD of (1), but without the isbn-attribute.
Write XPath queries that select
a) all journal nodes which do not have a cites-child
b) all title nodes that appear below journal nodes
c) the right-most leaf
d) the deepest node of the tree (right-most one, if not unique)
e) all authors which have written a journal and a book (i.e., author-nodes that appear under journal and book nodes, with the same text-string below them)
f) all books which are cited (based on their title)
g) same as f, but each book printed only once, in pre-order
h) all author names which appear inside the title of a book

(5)**[8]** Again, under the DTD of (1), but without the isbn-attribute.

Write XPath queries that select

a) all journal nodes which do not have a cites-child

b) all title nodes that appear below journal nodes

c) the right-most leaf

d) the deepest node of the tree (right-most one, if not unique)

e) all authors which have written a journal and a book (i.e., author-nodes that appear under journal and book nodes, with the same text-string below them)

f) all books which are cited (based on their title)

g) same as f, but each book printed only once, in pre-order

h) all author names which appear inside the title of a book

```
(a)  //journal[not(./cites)]
```

(5)**[8]** Again, under the DTD of (1), but without the isbn-attribute.
Write XPath queries that select

a) all journal nodes which do not have a cites-child

b) all title nodes that appear below journal nodes

c) the right-most leaf

d) the deepest node of the tree (right-most one, if not unique)

e) all authors which have written a journal and a book (i.e., author-nodes that appear under journal and book nodes, with the same text-string below them)

f) all books which are cited (based on their title)

g) same as f, but each book printed only once, in pre-order

h) all author names which appear inside the title of a book

```
(a)  //journal[not(./cites)]

(b)  //title[ancestor::journal]
```

(5)**[8]** Again, under the DTD of (1), but without the isbn-attribute.
Write XPath queries that select
a) all journal nodes which do not have a cites-child
b) all title nodes that appear below journal nodes
c) the right-most leaf
d) the deepest node of the tree (right-most one, if not unique)
e) all authors which have written a journal and a book (i.e., author-nodes that appear under journal and book nodes, with the same text-string below them)
f) all books which are cited (based on their title)
g) same as f, but each book printed only once, in pre-order
h) all author names which appear inside the title of a book

(a) //journal[not(./cites)]

(b) //title[ancestor::journal]

(c) //*[not(child::*) and not(following::*)]
or /descendant::*[position()=last()]

(5)**[8]** Again, under the DTD of (1), but without the isbn-attribute.
Write XPath queries that select
a) all journal nodes which do not have a cites-child
b) all title nodes that appear below journal nodes
c) the right-most leaf
d) the deepest node of the tree (right-most one, if not unique)
e) all authors which have written a journal and a book (i.e., author-nodes that appear under journal and book nodes, with the same text-string below them)
f) all books which are cited (based on their title)
g) same as f, but each book printed only once, in pre-order
h) all author names which appear inside the title of a book

(a) //journal[not(./cites)]

(b) //title[ancestor::journal]

(c) //*[not(child::*) and not(following::*)]
or /descendant::*[position()=last()]

(d)???

(e) //book//author[.=//journal/author]

(5)**[8]** Again, under the DTD of (1), but without the isbn-attribute.
Write XPath queries that select
a) all journal nodes which do not have a cites-child
b) all title nodes that appear below journal nodes
c) the right-most leaf
d) the deepest node of the tree (right-most one, if not unique)
e) all authors which have written a journal and a book (i.e., author-nodes that appear under journal and book nodes, with the same text-string below them)
f) all books which are cited (based on their title)
g) same as f, but each book printed only once, in pre-order
h) all author names which appear inside the title of a book

(a) //journal[not(./cites)]

(b) //title[ancestor::journal]

(c) //*[not(child::*) and not(following::*)]
or /descendant::*[position()=last()]

(d)???

(e) //book//author[.=//journal/author]

Question
How to report only **distinct** such authors?

(5)**[8]** Again, under the DTD of (1), but without the isbn-attribute.
Write XPath queries that select
a) all journal nodes which do not have a cites-child
b) all title nodes that appear below journal nodes
c) the right-most leaf
d) the deepest node of the tree (right-most one, if not unique)
e) all authors which have written a journal and a book (i.e., author-nodes that appear under journal and book nodes, with the same text-string below them)
f) all books which are cited (based on their title)
g) same as f, but each book printed only once, in pre-order
h) all author names which appear inside the title of a book

```
(a)  //journal[not(./cites)]

(b)  //title[ancestor::journal]

(c)  //*[not(child::*) and not(following::*)]
  or /descendant::*[position()=last()]

(d)???

(e)  //book//author[.=//journal/author]
```

Question
How to report only **distinct** such authors?

```
//book/author[.=//jour/author and not(.=preceding::book/author)]
```

(selects the *first* occurrences)

(5)**[8]** Again, under the DTD of (1), but without the isbn-attribute.
Write XPath queries that select
a) all journal nodes which do not have a cites-child
b) all title nodes that appear below journal nodes
c) the right-most leaf
d) the deepest node of the tree (right-most one, if not unique)
e) all authors which have written a journal and a book (i.e., author-nodes that appear under journal and book nodes, with the same text-string below them)
f) all books which are cited (based on their title)
g) same as f, but each book printed only once, in pre-order
h) all author names which appear inside the title of a book

(a) `//journal[not(./cites)]`

(b) `//title[ancestor::journal]`

(c) `//*[not(child::*) and not(following::*)]`
or `/descendant::*[position()=last()]`

(d)???

(e) `//book//author[.=//journal/author]`

(f) `//cites/book`
or `/bib/book[title=//cites/book/title]`

(5)**[8]** Again, under the DTD of (1), but without the isbn-attribute.
Write XPath queries that select
a) all journal nodes which do not have a cites-child
b) all title nodes that appear below journal nodes
c) the right-most leaf
d) the deepest node of the tree (right-most one, if not unique)
e) all authors which have written a journal and a book (i.e., author-nodes that appear under journal and book nodes, with the same text-string below them)
f) all books which are cited (based on their title)
g) same as f, but each book printed only once, in pre-order
h) all author names which appear inside the title of a book

```
(a)  //journal[not(./cites)]

(b)  //title[ancestor::journal]

(c)  //*[not(child::*) and not(following::*)]
```
or `/descendant::*[position()=last()]`

```
(d)???

(e)  //book//author[.=//journal/author]

(f)  //cites/book
 or /bib/book[title=//cites/book/title]
```

Question
How to change this query, if we instead use the ID/IDREF attributes of before?

(5)**[8]** Again, under the DTD of (1), but without the isbn-attribute.
Write XPath queries that select
a) all journal nodes which do not have a cites-child
b) all title nodes that appear below journal nodes
c) the right-most leaf
d) the deepest node of the tree (right-most one, if not unique)
e) all authors which have written a journal and a book (i.e., author-nodes that appear under journal and book nodes, with the same text-string below them)
f) all books which are cited (based on their title)
g) same as f, but each book printed only once, in pre-order
h) all author names which appear inside the title of a book

```
(a)  //journal[not(./cites)]

(b)  //title[ancestor::journal]

(c)  //*[not(child::*) and not(following::*)]
```
or `/descendant::*[position()=last()]`

```
(d)???

(e)  //book//author[.=//journal/author]

(f)  //cites/book
 or /bib/book[title=//cites/book/title]

(g)//cite/book[not(.=preceding::book[parent::cites])]
Or  /bib/book[title=//cites/book/title and not(.=preceding::book[parent::bib])]
```

(5)**[8]** Again, under the DTD of (1), but without the isbn-attribute.
Write XPath queries that select
a) all journal nodes which do not have a cites-child
b) all title nodes that appear below journal nodes
c) the right-most leaf
d) the deepest node of the tree (right-most one, if not unique)
e) all authors which have written a journal and a book (i.e., author-nodes that appear under journal and book nodes, with the same text-string below them)
f) all books which are cited (based on their title)
g) same as f, but each book printed only once, in pre-order
h) all author names which appear inside the title of a book

```
(a)  //journal[not(./cites)]

(b)  //title[ancestor::journal]

(c)  //*[not(child::*) and not(following::*)]
  or /descendant::*[position()=last()]

(d)???

(e)  //book//author[.=//journal/author]

(f)  //cites/book
 or /bib/book[title=//cites/book/title]

(g)//cite/book[not(.=preceding::book[parent::cites])]
Or  /bib/book[title=//cites/book/title and not(.=preceding::book[parent::bib])]
```

```
(h) //author/text()[contains(//book/title,.)]
```