HW3CSDS234

1

$r(A,B,C)$ with 40K tuples and $s(C,D)$ with 60K tuples. Memory contains one block for each relation to compute join operations.

Estimate (1) the largest output size and (2) the worst case I/O cost in terms of the total number of data blocks to be transferred for the following queries and settings:

$r \bowtie_{r.C=s.C} s$, Nested Loop Join, the memory contains the entire smaller relation (the one that takes a smaller number of data blocks to hold) of r and s;

> 1. 40K$\times$60K=2.4×10^9
> 2. $B(r)+B(s)$ because smaller relation fits entirely in memory

$r \bowtie_{r.A=S.C} s$, Block-Nested Loop Join, memory is small and contains one block from each table of r and s.

> 1. 40K$\times$60K=2.4×10^9
> 2. $B(r)\times B(s)+B(r)$

Hash join $r \bowtie_{r.A=S.C} s$, with a hash function mod h for r and s.

> 1. 40K$\times$60K=2.4×10^9
> 2. Need to read, write, and then join (another read). So the I/O cost is $3\times(B(r)+B(s))$

$r \bowtie_{r.C=s.C} s$, Index Nested Loop Join (using a B+ tree primary index with degree d and height h1 defined on S.C, and S.C is a key attribute of relation S);

> 1. 40K, since S.C is a key attribute each tuple in r matches at most one tuple in s
> 2. $B(r)+N(r)\times(h_1+1)\rightarrow B(r)+40k\times(h_1+1)$

$r \bowtie_{r.C=s.C} s$, Index Nested Loop Join (using a secondary B+ tree index with degree d and height h2 defined on r.C)

> 1. 40K$\times$60K=2.4×10^9

> 2. $B(s) + N(s) \times (h_2 + N(r)) \rightarrow B(s) + 60k \times (h_2 + 40k)$

2

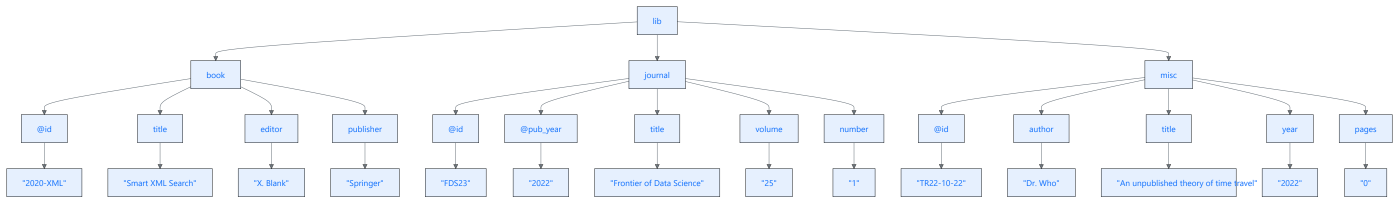
Consider the following XML document describing library entries

```
<lib>
    <book id=2020-XML>
        <title> Smart XML Search</title>
        <editor> X. Blank </editor>
        <publisher>Springer</publisher>
    </book>
    <journal id= FDS23 pub_year = “2022”>
        <title>Frontier of Data Science</title>
        <volume>25<number>1</volume></number>
    </journal>
    <misc id= “tr22-10” id= “TR22-10-22”>
        <author>Dr. Who</author>
        <title>An unpublished theory of time travel </title>
        <year>2022</year> <pages>0<page>
    </misc>
</lib>
```

Check if the XML document is well structured, if not, correct the document.

```
<lib>
    <book id="2020-XML">
        <title> Smart XML Search</title>
        <editor> X. Blank </editor>
        <publisher>Springer</publisher>
    </book>
    <journal id="FDS23" pub_year = "2022">
        <title>Frontier of Data Science</title>
        <volume>25</volume>
        <number>1</number>
    </journal>
    <misc id= "TR22-10-22">
        <author>Dr. Who</author>
        <title>An unpublished theory of time travel </title>
        <year>2022</year>
        <pages>0</pages>
    </misc>
</lib>
```

Draw a tree representation of the well formed XML document.



Write a DTD so the well-formed version of the example in (1) is also a valid XML document given the DTD, along with the following additional constraints:

1. Lib entries must have book, journal, and misc. elements occurred in an order, i.e., journal occurred after book, and misc. after journal;
2. Lib must have at least one book and at least one journal, but misc is optional;
3. title is mandatory in the book element
4. there can be at most one title, editor, publisher, volume, and year for a journal;
5. a library entry (as either a book or a journal element) may have more than one authors

```
<!DOCTYPE lib [
  <!ELEMENT lib (book+, journal+, misc*)>
  <!ELEMENT book (title, editor*, publisher*, author*)>
  <!ATTLIST book id ID #REQUIRED>
  <!ELEMENT journal (title, editor?, publisher?, volume?, number?, year?, author*)>
  <!ATTLIST journal id ID #REQUIRED pub_year CDATA #REQUIRED>
  <!ELEMENT misc (author+, title, year?, pages?)>
  <!ATTLIST misc id ID #REQUIRED>
  <!ELEMENT title (#PCDATA)>
  <!ELEMENT editor (#PCDATA)>
  <!ELEMENT publisher (#PCDATA)>
  <!ELEMENT volume (#PCDATA)>
  <!ELEMENT number (#PCDATA)>
  <!ELEMENT year (#PCDATA)>
  <!ELEMENT pages (page+)>
  <!ELEMENT page (#PCDATA)>
  <!ELEMENT author (#PCDATA)>
]>
```

4 A web “attacker” may modify the DTD on a server to make a case that there exists no XML that is valid for the revised DTD (a case of DTD that is known to be “not satisfiable”), blocking the processing of XML documents. Give an example of such a DTD by revising the one in (3), and briefly explain why it is not satisfiable.

```
<!DOCTYPE lib [
  <!ELEMENT lib (lib)>
]>
```

3

Consider the XML code in Question 2 as a fragment of a large XML document and its DTD in Question 2. Write XPath to retrieve the following information.

List the titles of all the books edited by “X.Blank” with publisher “Springer”;

> /lib/book[editor=‘X. Blank’ and publisher=‘Springer’]/title/text()

Find the first two journal elements as children of the Lib element;

> /lib/journal[position() <= 2]

Find all the journals with editor “Y.Wu” that is published later than 2020;

> /lib/journal[@pub_year > 2020 and editor='Y.Wu']/title/text()

Select all the editors and publishers of the books written by “M.Stonebreaker.“

> /lib/book[author=‘M.Stonebreaker’]/(editor | publisher)

List the titles of misc. entries written by “Dr. Who,” which were published after 2020.

> /lib/misc[author=‘Dr. Who’ and year > 2020]/title/text()