Problem 1 [70 points]: Write the following queries in XQuery, with the output delimited by the tags ⟨answer⟩… ⟨/answer⟩:

1. Output the years, titles, and conferences of all the papers (inproceedings in db-inprocs.xml) authored by Hector Garcia-Molina, nested inside a publication tag.

   ⟨answer⟩
   { for $p in document("db-inproc.xml")/dblp/inproceedings
     where $p/author = "Hector Garcia-Molina"
     return ⟨publication⟩ {$p/title, $p/year, $p/booktitle} ⟨/publication⟩ }
   ⟨/answer⟩

2. For each author of the paper entitled “Efficiently Publishing Relational Data as XML Documents.”, list all conferences where they have published, grouped by author.

   ⟨answer⟩
   { for $p in document("db-inproc.xml")/dblp/inproceedings, $a in $p/author
     where $p/title = "Efficiently Publishing Relational Data as XML Documents."
     return ⟨author name="{$p/author}⟩ { for $p2 in document("db-inproc.xml")/dblp/inproceedings,
       $a2 in $p2/author
       let $b := $p2/booktitle
       where $a = $a2
       return $b
     } ⟨/author⟩
   } ⟨/answer⟩

If we want to eliminate repetitions of conferences:

   ⟨answer⟩
   { for $p in document("db-inproc.xml")/dblp/inproceedings, $a in $p/author
     where $p/title = "Efficiently Publishing Relational Data as XML Documents."
     return ⟨author name="{$p/author}⟩ { fn:distinct-values(

   ⟨/answer⟩
for $p2 in document("db-inproc.xml")/dblp/inproceedings,
   $a2 in $p2/author
let $b := $p2/booktitle
where $a = $a2
return $b)
} </author>
</answer>

3. List the titles of all papers cited by “Efficiently Publishing Relational Data as XML Documents.”.

<answer> {
for $p in document("db-inproc.xml")/dblp/inproceedings
where $p/title = "Efficiently Publishing Relational Data as XML Documents."
return <title> {
for $q in document("db-inproc.xml")/dblp/inproceedings
where $p/cite = $q/@key
return $q/title/text()
} </title>
} </answer>

4. Count the number of papers in db-inproc.xml.

<answer> {
let $c := fn:count(
   for $p in document("db-inproc.xml")/dblp/inproceedings
   return $p)
return $c
} </answer>

5. Find the count of papers written by the author of the most papers.

<answer> {
let $m := fn:max(
   for $a in
   fn:distinct-values(document("db-inproc.xml")/dblp/inproceedings/author/text())
   let $c :=
   fn:count(document("db-inproc.xml")/dblp/inproceedings/[author/text() = $a])
return $c
} return $m
} </answer>

6. List all conference proceedings titles in descending alphabetical order.

2
<answer> 
    for $t$ in document("db-proc.xml")/dblp/proceedings/title 
    order by $t$/text() descending 
    return $t$ 
} </answer>

7. List the second author along with the paper title for each paper in SIGMOD 1999.

<answer> 
    for $p$ in document("db-inproc.xml")/dblp/inproceedings, 
    $a$ in $p$/author[fn:position() = 2]/text() 
    where $p$/@key = "conf/sigmod/99" 
    return <paper> {$p/title} <second-author> {$a} </second-author> </paper> 
} </answer>
Problem 2 [30 points]:

There are many possible schemas for this one, and virtually anything reasonable was acceptable. The “distributes” and “authors” relationship sets might be encoded in XML simply via containment of attributes: e.g., a distributor might be represented a parent element with a set of CDs inside it. In an RDBMS, we must explicitly encode every relationship set as a separate relation (unless it is one-to-many or one-to-one). An example of part of the schema would be:

```xml
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="book">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="isbn" type="xsd:string"/>
        <xsd:element name="title" type="xsd:string"/>
        <xsd:element name="publisher" type="xsd:string"/>
        <xsd:element name="editor" type="xsd:string" minOccurs="0"/>
      </xsd:sequence>
      <xsd:key name="bookKey">
        <xsd:selector xpath="=./book"/>
        <xsd:field xpath="./isbn"/>
      </xsd:key>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="author">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="firstName" type="xsd:string" minOccurs="0"/>
        <xsd:element name="lastName" type="xsd:string"/>
        <xsd:element name="wrote" type="xsd:string"/>
      </xsd:sequence>
      <xsd:keyref name="bookRef" type="bookKey">
        <xsd:selector xpath="=./author"/>
        <xsd:field xpath="./wrote"/>
      </xsd:keyref>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```