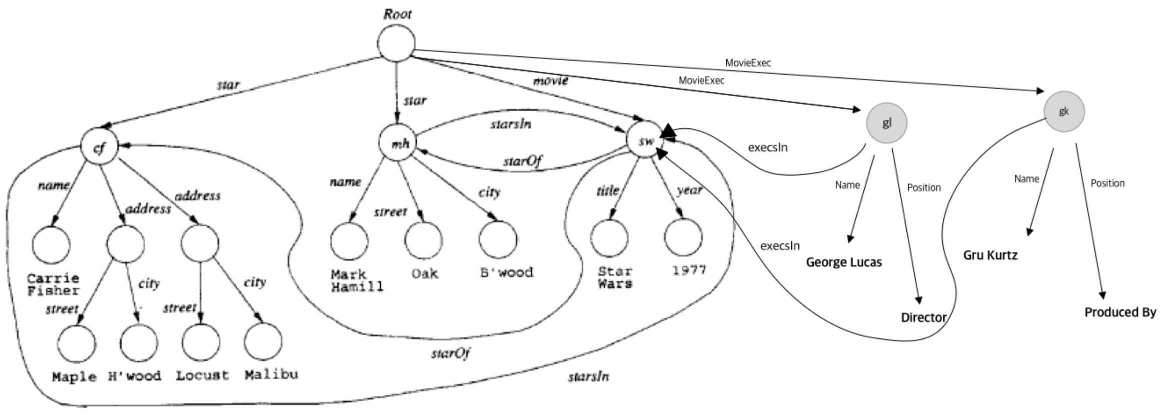
CSC343 Worksheet 9 Solution

June 28, 2020

1. **Exercise 11.1.1:**

a) 

<u>Notes:</u>

- Semistructured data
  - serves as a model suitable for **databases integration**, that is, for describing the data contained in two or more databases that contain similar data with different schemas
  - It serves as the underlying model for notations such as XML, to be taken up in Section 2, that are being used to share information on the web.
- Semistructured Data Representation
  - is a collection of nodes

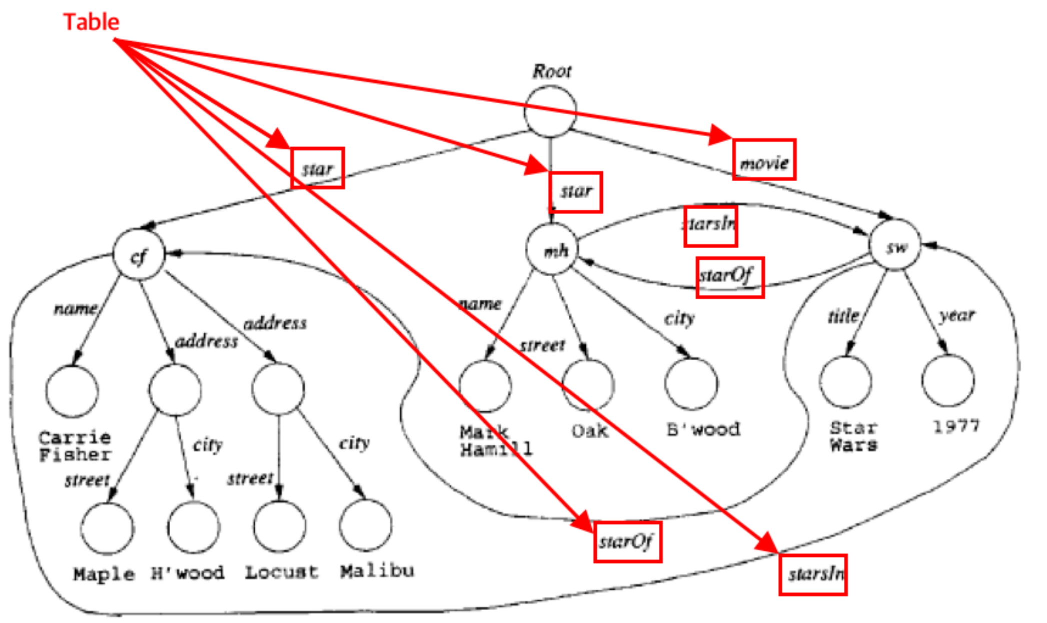
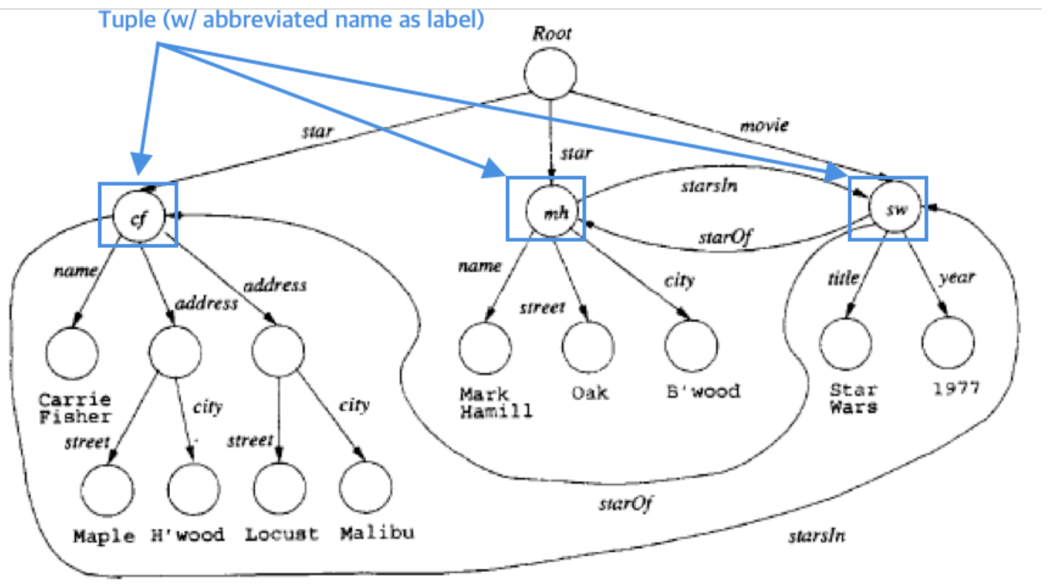
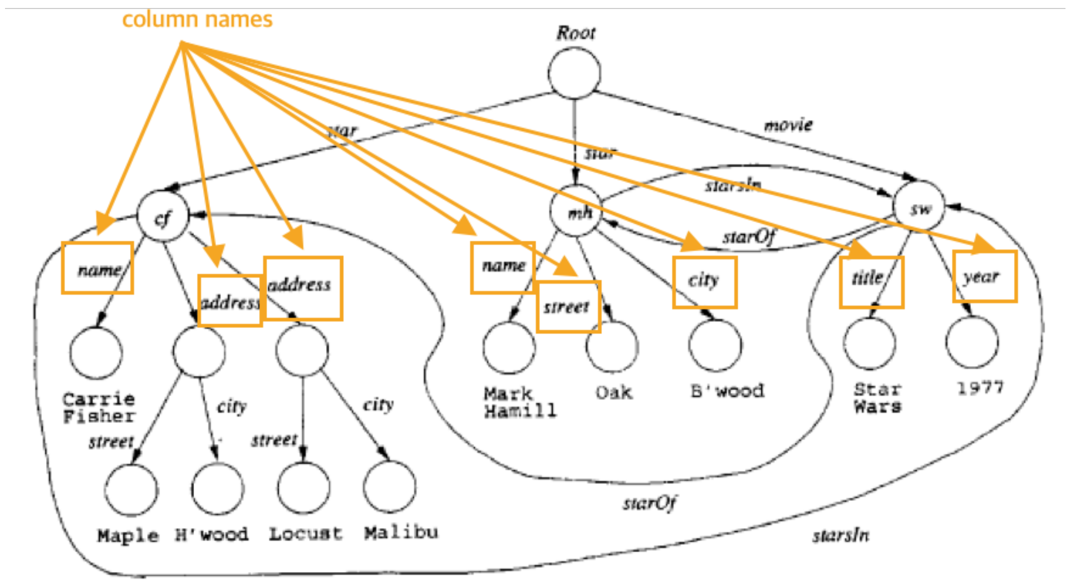
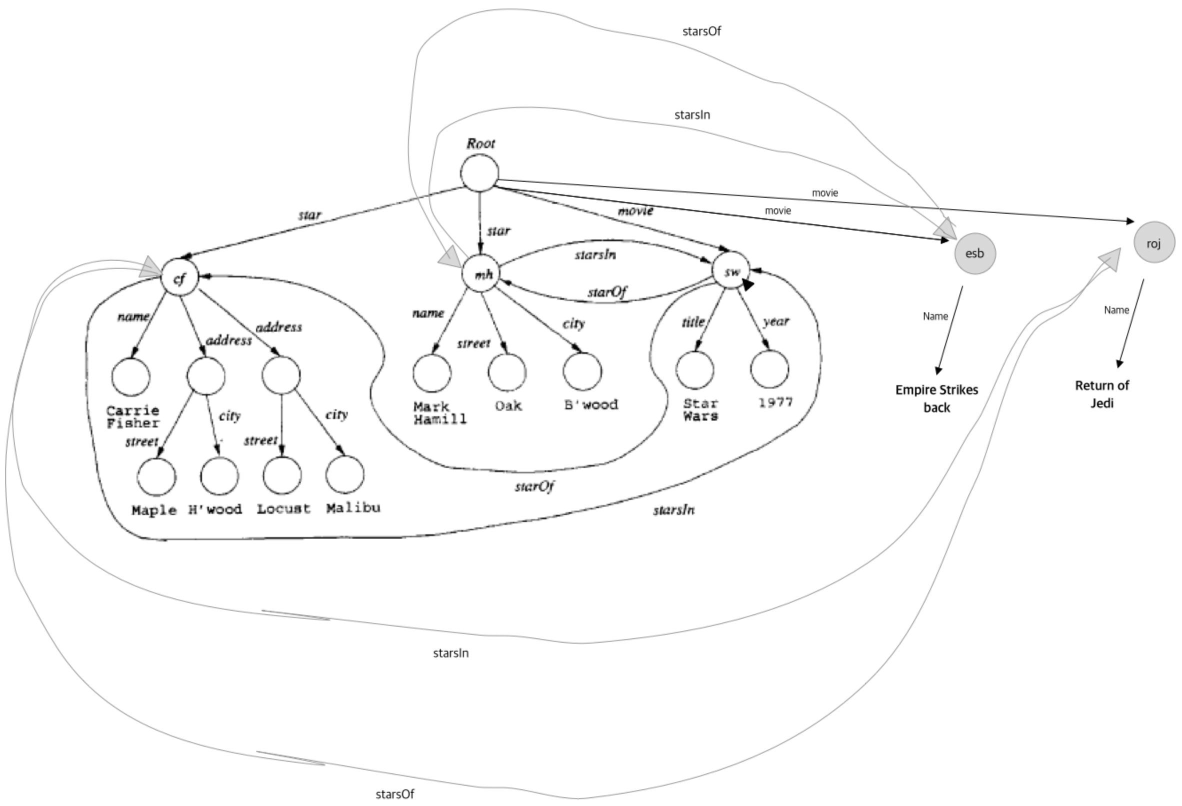


Table
Root
star
star
movie
cf
mh
sw
starsIn
starOf
name
address
address
name
street
city
title
year
Carrie Fisher
Mark Hamill
Oak
B'wood
Star Wars
1977
city
city
street
street
Maple
H'wood
Locust
Malibu
starOf
starsIn

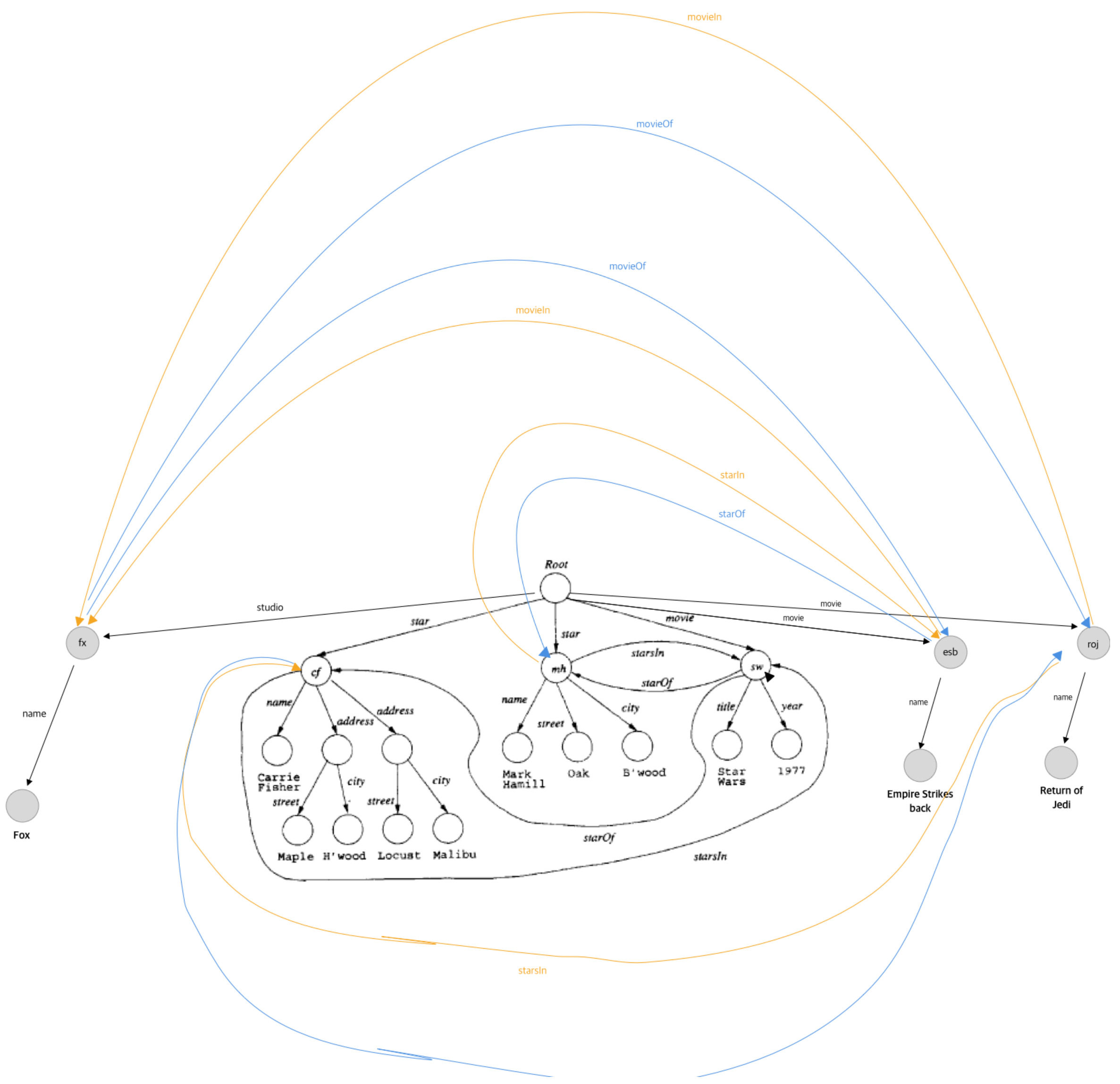

Tuple (w/ abbreviated name as label)
Root
star
star
movie
cf
mh
sw
starsIn
starOf
name
address
address
name
street
city
title
year
Carrie Fisher
Mark Hamill
Oak
B'wood
Star Wars
1977
city
city
street
street
Maple
H'wood
Locust
Malibu
starOf
starsIn


column names
Root
star
movie
starsIn
starOf
cf
mh
sw
name
address
address
name
street
city
title
year
Carrie Fisher
street
city
street
city
Maple
H'wood
Locust
Malibu
Mark Hamill
Oak
B'wood
Star Wars
1977
starOf
starsIn

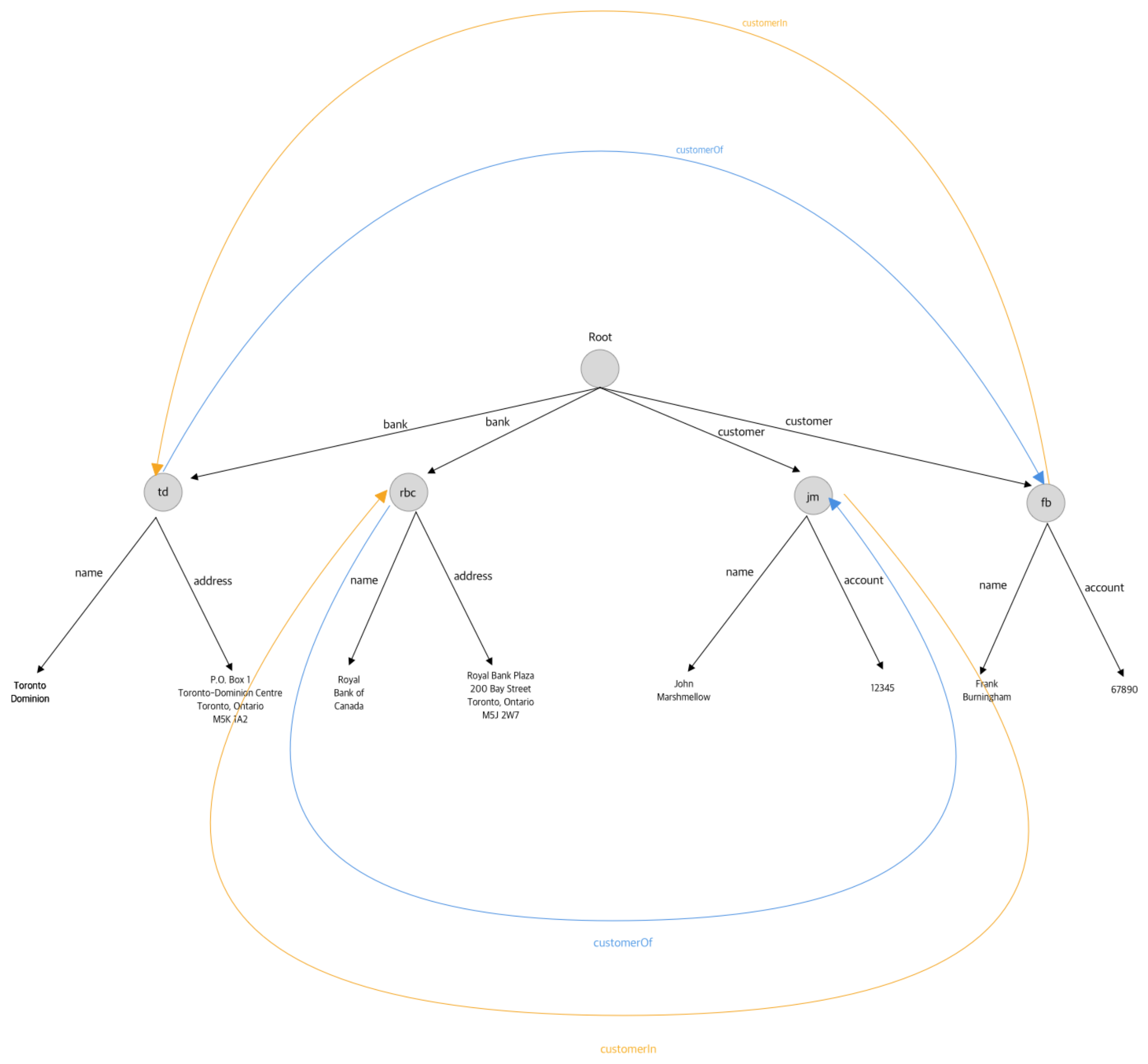
•


starsOf
starsIn
Root
movie
movie
movie
star
star
esb
roj
Name
Name
Empire Strikes back
Return of Jedi
cf
mh
sw
starsIn
starOf
name
address
address
name
street
city
title
year
Carrie Fisher
street
city
street
city
Maple
H'wood
Locust
Malibu
Mark Hamill
Oak
B'wood
Star Wars
1977
starOf
starsIn
starsIn
starsOf

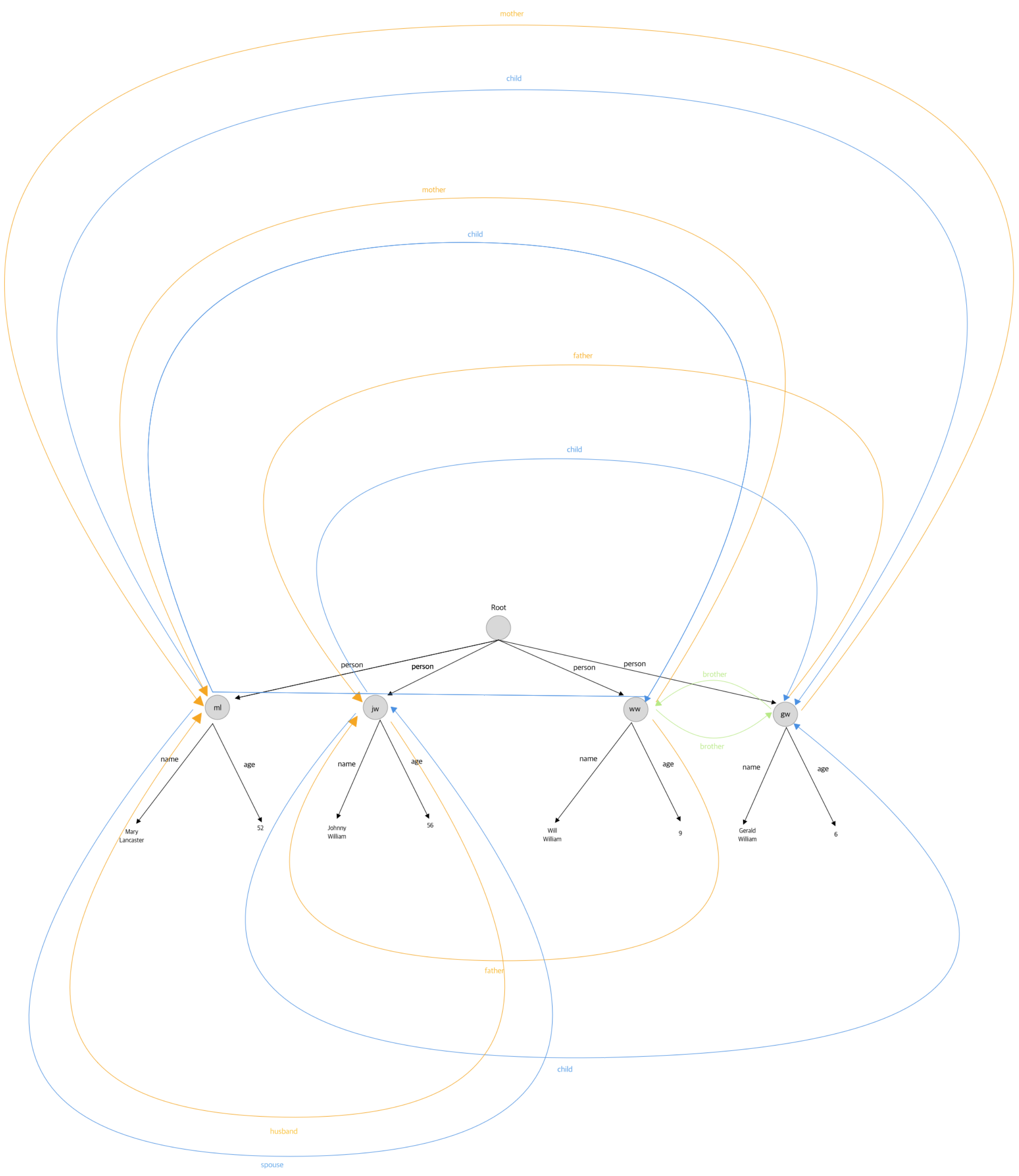
b)


movieIn
movieOf
movieOf
movieIn
starIn
starOf
Root
studio
star
movie
movie
movie
fx
cf
mh
sw
esb
roj
star
starsIn
starOf
name
name
name
address
address
street
city
city
street
city
title
year
Fox
Carrie Fisher
Maple
H'wood
Locust
Malibu
Mark Hamill
Oak
B'wood
Star Wars
1977
Empire Strikes back
Return of Jedi
starOf
starsIn
starsIn

c)


customerIn
customerOf
Root
bank
bank
customer
customer
td
rbc
jm
fb
name
address
name
address
name
account
name
account
Toronto Dominion
P.O. Box 1 Toronto-Dominion Centre Toronto, Ontario M5K 1A2
Royal Bank of Canada
Royal Bank Plaza 200 Bay Street Toronto, Ontario M5J 2W7
John Marshmellow
12345
Frank Burningham
67890
customerOf
customerIn

2.



3.

4. The difference is that UML must fit data into its schema, where as the semi structured data allows whatever schema information that is appropriate to be attached to data

<u>Notees:</u>

- Semi-structured Data
  - Is schemaless

  - Is motivated primarily by its flexibility
  - One could enter data at will, and attach to the data whatever schema information you felt was appropriate for that data.
  - Makes query processing harder
- Structured Data
  - Is rigid framework into which data is placed.
  - Data must fit into schema
  - Fixed schema allows data to be organized with data structures that support efficient answering of queries
  - e.g. UML, E/R, Relational, ODL

5. a)

```
<? xml version = "1.0" encoding="utf-8" standalone = "yes">
<StarMovieData>
    <Star starID="cf" starredIn="sw">
        <Name>Carrie Fisher</Name>
        <Address>
            <Street>123 Maple St.</Street>
            <City>Hollywood</City>
        </Address>
        <Address>
            <Street>5 Locust Ln.</Street>
            <City>Malibu</City>
        </Address>
    </Star>
    <Star starID="mh" starredIn="sw">
        <Name>Mark Hamill</Name>
        <Street>456 Oak Rd.</Street>
        <City>Brentwood</City>
    </Star>
    <Movie movieID="sw" starsOf="cf", "mh">
        <Title>Star Wars</Title>
        <Year>1977</Year>
    </Movie>
    <MovieExec movieExecID="gl" execsIn="sw">
        <Name>George Lucas</Name>
        <Position>Director</Position>
    </MovieExec>
    <MovieExec movieExecID="gk" execsIn="sw">
        <Name>Gru Kurtz</Name>
        <Position>Produced By</Position>
    </MovieExec>
</StarMovieData>

```

- XML
  - is called *Extensible Markup Language*
  - is an example of semistructured data
- XML with and without a Schema

- has two different types
  1. Well-formed XML
     * allows to invent your own tags
     * corresponds very-similarly to semi-structured data

     Example:

```xml
<? xml version = "1.0" encoding="utf-8" standalone = "yes
">
<StarMovieData>
    <Star>
        <Name>Carrie Fisher</Name>
        <Address>
            <Street>123 Maple St.</Street>
            <City>Hollywood</City>
        </Address>
        <Address>
            <Street>5 Locust Ln.</Street>
            <City>Malibu</City>
        </Address>
    </Star>
    <Star>
        <Name>Mark Hamill</Name>
        <Street>456 Oak Rd.</Street>
        <City>Brentwood</City>
    </Star>
    <Movie>
        <Title>Star Wars</Title>
        <Year>1977</Year>
    </Movie>
</StarMovieData>

```

  2. Valid XML
     * Involes ”Document Type Definition”
     * specifies allowable tags and gives a grammar for how they may be nested

- Attributes
  - is used to represent connections in a semistructured data graph

  Example:

```xml
<? xml version = "1.0" encoding="utf-8" standalone = "yes">
<StarMovieData>
    <Star starID="cf" starredIn="sw">
        <Name>Carrie Fisher</Name>
        <Address>
            <Street>123 Maple St.</Street>
            <City>Hollywood</City>
        </Address>
        <Address>
            <Street>5 Locust Ln.</Street>
            <City>Malibu</City>
        </Address>
    </Star>
    <Star starID="mh" starredIn="sw">
        <Name>Mark Hamill</Name>
        <Street>456 Oak Rd.</Street>
        <City>Brentwood</City>
    </Star>
    <Movie starID="sw" starOf="cf", "mh">
        <Title>Star Wars</Title>
        <Year>1977</Year>
    </Movie>
</StarMovieData>

```

- Namespaces
  - **Syntax:** xmlns:*name*:URI
  - Is similar to import numpy as np in python
  - Is used to distinguish tags coming from different sources, i.e. HTML

  Example:

  Retireving element *StarMovieData* from document **infolab.stanford.edu/movies**. Set md as the name of import

```
    <md:StarMovieData xmlns:md="http://infolab.stanford.edu/
  movies">

```

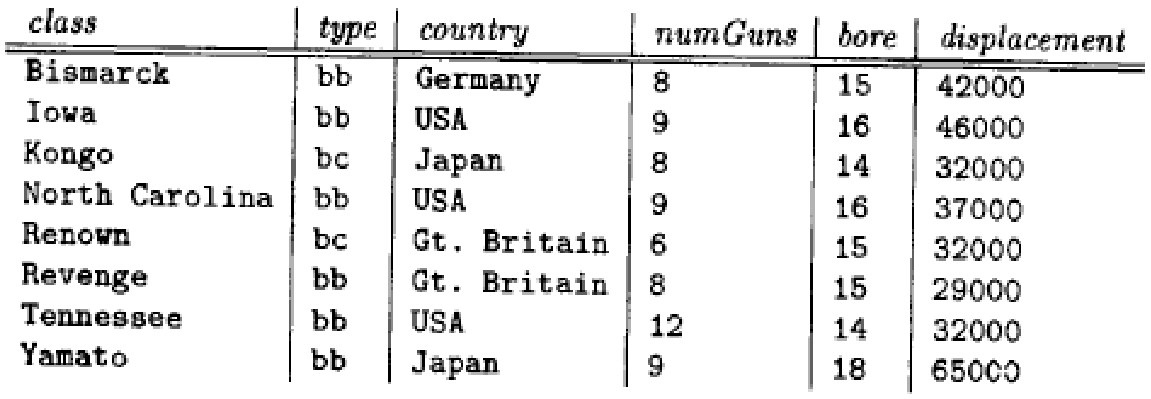
b)

```
    <? xml version = "1.0" encoding="utf-8" standalone = "yes">
    <StarMovieData>
        <Star starID="cf" starredIn="sw">
            <Name>Carrie Fisher</Name>
            <Address>
                <Street>123 Maple St.</Street>
                <City>Hollywood</City>
            </Address>
            <Address>
                <Street>5 Locust Ln.</Street>
                <City>Malibu</City>
            </Address>
        </Star>
        <Star starID="mh" starredIn="sw">
            <Name>Mark Hamill</Name>
            <Street>456 Oak Rd.</Street>
            <City>Brentwood</City>
        </Star>
        <Movie movieID="sw" starsOf="cf", "mh">
            <Title>Star Wars</Title>
            <Year>1977</Year>
        </Movie>
        <Movie movieID="esb" starOf="cf", "mh">
            <Title>Empire Strikes Back</Title>
            <Year>1980</Year>
        </Movie>
        <Movie movieID="roj" starOf="cf", "mh">
            <Title>Return of Jedi</Title>
            <Year>1983</Year>
        </Movie>
    </StarMovieData>

```

c)

```
    <? xml version = "1.0" encoding="utf-8" standalone = "yes">
    <StarMovieData>
        <Star starID="cf" starredIn="sw">
            <Name>Carrie Fisher</Name>
            <Address>
                <Street>123 Maple St.</Street>
                <City>Hollywood</City>
            </Address>
            <Address>
                <Street>5 Locust Ln.</Street>
                <City>Malibu</City>
            </Address>
        </Star>
        <Star starID="mh" starredIn="sw">
            <Name>Mark Hamill</Name>
            <Street>456 Oak Rd.</Street>
            <City>Brentwood</City>
        </Star>
        <Movie movieID="sw" starsOf="cf", "mh" movieIn="fx">
            <Title>Star Wars</Title>
            <Year>1977</Year>
        </Movie>
        <Movie movieID="esb" starOf="cf", "mh" movieIn="fx">
            <Title>Empire Strikes Back</Title>
            <Year>1980</Year>
        </Movie>
        <Movie movieID="roj" starOf="cf", "mh" movieIn="fx">
            <Title>Return of Jedi</Title>
            <Year>1983</Year>
        </Movie>
        <Studio studioID="fx" movieOf="esb", "roj", "sw">
            <Name>Fox</Name>
            <Address>Hollywood</Address>
        </Studio>
    </StarMovieData>

```

6. Consider the following relation Classes:



class	type	country	numGuns	bore	displacement
Bismarck	bb	Germany	8	15	42000
Iowa	bb	USA	9	16	46000
Kongo	bc	Japan	8	14	32000
North Carolina	bb	USA	9	16	37000
Renown	bc	Gt. Britain	6	15	32000
Revenge	bb	Gt. Britain	8	15	29000
Tennessee	bb	USA	12	14	32000
Yamato	bb	Japan	9	18	65000

(a) Sample data for relation Classes

```
<? xml version = "1.0" encoding="utf-8" standalone = "yes">
<shipsData>
    <Classes>
        <Class>Bismarck</Class>
        <Type>bb</Type>
        <Country>Germany</Country>
        <NumGuns>8</NumGuns>
        <Bore>15</Bore>
        <Displacement>42000</Displacement>
    </Classes>
    <Classes>
        <Class>Iowa</Class>
        <Type>bb</Type>
        <Country>USA</Country>
        <NumGuns>9</NumGuns>
        <Bore>16</Bore>
        <Displacement>46000</Displacement>
    </Classes>
    <Classes>
        <Class>Kongo</Class>
        <Type>bc</Type>
        <Country>Japan</Country>
        <NumGuns>8</NumGuns>
        <Bore>14</Bore>
        <Displacement>32000</Displacement>
    </Classes>
    <Classes>
        <Class>North Carolina</Class>
        <Type>bb</Type>
        <Country>USA</Country>
        <NumGuns>9</NumGuns>
        <Bore>16</Bore>
        <Displacement>37000</Displacement>
    </Classes>
    <Classes>
        <Class>Renown</Class>
```

```
            <Type>bc</Type>
            <Country>Gt. Britain</Country>
            <NumGuns>6</NumGuns>
            <Bore>15</Bore>
            <Displacement>32000</Displacement>
        </Classes>
        <Classes>
            <Class>Revenge</Class>
            <Type>bb</Type>
            <Country>Gt. Britain</Country>
            <NumGuns>8</NumGuns>
            <Bore>15</Bore>
            <Displacement>29000</Displacement>
        </Classes>
        <Classes>
            <Class>Tennessee</Class>
            <Type>bb</Type>
            <Country>USA</Country>
            <NumGuns>12</NumGuns>
            <Bore>14</Bore>
            <Displacement>32000</Displacement>
        </Classes>
        <Classes>
            <Class>Yamato</Class>
            <Type>bb</Type>
            <Country>Japan</Country>
            <NumGuns>9</NumGuns>
            <Bore>18</Bore>
            <Displacement>65000</Displacement>
        </Classes>
    </shipsData>

```

7. • DocRoot

```
    <DocRoot docID="" rootElementID="" />

```

• SubElement

```
    <Subelement parentID="" childID="" position="" />

```

• ElementAttribute

```
    <ElementAttribute elementID="" name="" value="" />

```

• ElementValue

```
    <ElementValue elementID="" name="" value="" />

```

Notes:

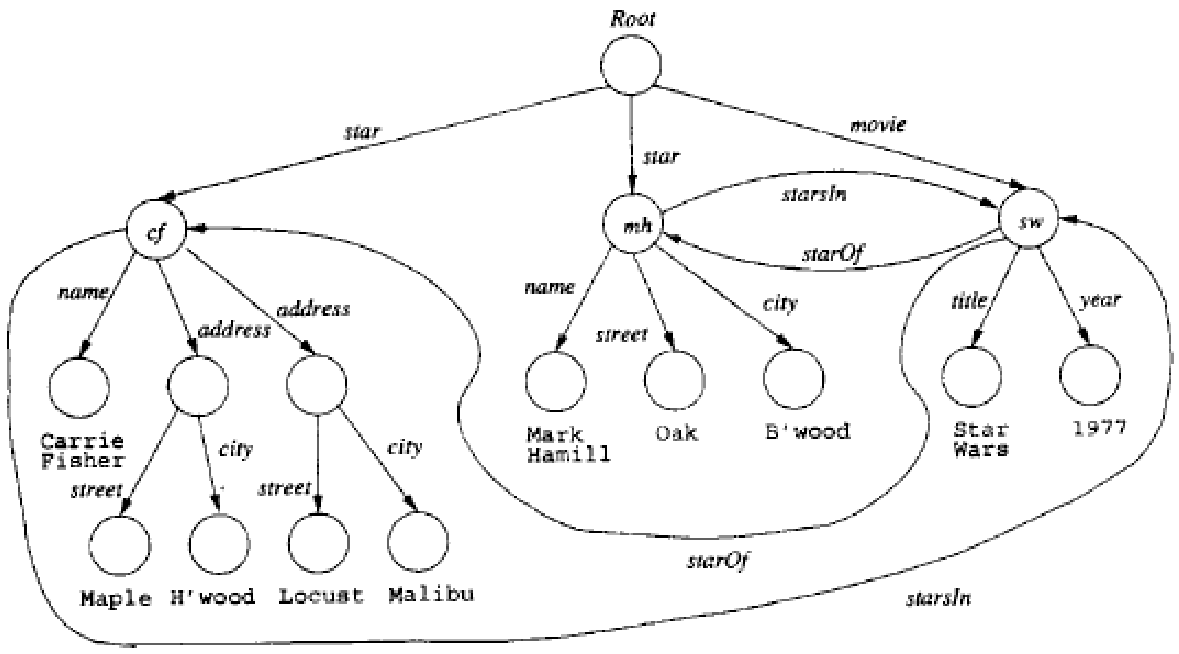
- Empty element
  - is an element that is complete by itself
  - rather than being composed of a start tag, data and an end tag, the empty element is a combined start and end tag.

  Example:

```
<phone>
    <entry>
        <name>Chip</name>
        <extension number="3"/>
    </entry>
</phone>

```

  - Phone is not an empty element
  - extension is an empty element

8. The following will be used as an example.



```
<? xml version = "1.0" encoding="utf-8" standalone = "yes">
<DocRoot docID="root" rootElementID="">
    <SubElement parentID="root" childID="cf" position="1">
        <ElementAttribute elementID="cf-starsIn" name="starsIn"
value="sw"/>
```

```
            <ElementValue elementID="cf-name" name="name" value="Carrie Fisher"/>
            <SubElement parentID="cf" childID="cf-add_1" position="1">
                <City elementID="cf-add_1-city" value="Hollywood">
                <Street elementID="cf-add_1-street" value="Maple">
            </SubElement>
            <SubElement parentID="cf" childID="cf-add_2" position="2">
                <ElementValue elementID="cf-add_2-city" value="Malibu">
                <ElementValue elementID="cf-add_1-street" value="Locust">
            </SubElement>
        </SubElement>
        <SubElement parentID="root" childID="mh" position="2">
            <ElementAttribute elementID="mh-starsIn" name="starsIn" value="sw"/>
            <ElementValue elementID="mh-name" value="Mark Hamill"/>
            <SubElement parentID="mh" childID="mh-add_2" position="1">
                <ElementValue elementID="mh-add_2-city" value="Bollywood">
                <ElementValue elementID="mh-add_2-street" value="Oak">
            </SubElement>
        </SubElement>
        <SubElement parentID="root" childID="sw" position="3">
            <ElementAttribute elementID="mh-starsOf_1" name="starsOf" value="cf"/>
            <ElementAttribute elementID="mh-starsOf_2" name="starsOf" value="mh"/>
            <ElementValue elementID="sw-title" value="Star Wars" />
            <ElementValue elementID="sw-year" value="1977" />
        </SubElement>
    </DocRoot>

```

9. a)

```
<? xml version = "1.0" encoding="utf-8" standalone = "yes">
<StarMovieData>
    <Star starID="cf" starredIn="sw">
        <Name>Carrie Fisher</Name>
        <Address>
            <Street>123 Maple St.<Street>
            <City>Hollywood</City>
        </Address>
        <Address>
            <Street>5 Locust Ln.</Street>
            <City>Malibu</City>
        </Address>
    </Star>
    <Star starID="mh" starredIn="sw">
        <Name>Mark Hamill</Name>
        <Address>
            <Street>456 Oak Rd.</Street>
            <City>Brentwood</City>
        </Address>
    </Star>
    <Movie movieID="sw" starsOf="cf mh">
```

```
        <Title>Star Wars</Title>
        <Year>1977</Year>
    </Movie>
    <Movie movieID="esb" starsOf="cf mh">
        <Title>The Empire Strikes Back</Title>
        <Year>1980</Year>
    </Movie>
    <Movie movieID="roj" starsOf="cf mh">
        <Title>Return of the Jedi</Title>
        <Year>1983</Year>
    </Movie>
</StarMovieData>

```

```
<? xml version = "1.0" encoding="utf-8" standalone = "yes">
<StarMovieData>
    <Star starID="cf" starredIn="sw">
        <Name>Carrie Fisher</Name>
        <Address>
            <Street>123 Maple St.<Street>
            <City>Hollywood</City>
        </Address>
        <Address>
            <Street>5 Locust Ln.</Street>
            <City>Malibu</City>
        </Address>
    </Star>
    <Star starID="hf" starredIn="sw fw">
        <Name>Harrison Ford</Name>
    </Star>

    <Star starID="mh" starredIn="sw">
        <Name>Mark Hamill</Name>
        <Address>
            <Street>456 Oak Rd.</Street>
            <City>Brentwood</City>
        </Address>
    </Star>
    <Movie movieID="sw" starsOf="cf mh hf">
        <Title>Star Wars</Title>
        <Year>1977</Year>
    </Movie>
    <Movie movieID="esb" starsOf="cf mh hf">
        <Title>The Empire Strikes Back</Title>
        <Year>1980</Year>
    </Movie>
    <Movie movieID="roj" starsOf="cf mh hf">
        <Title>Return of the Jedi</Title>
        <Year>1983</Year>
    </Movie>
    <Movie movieID="fw" starsOf="hf">
        <Title>Firewall</Title>
        <Year>2006</Year>
    </Movie>
```

```
</StarMovieData>

```

b)
```
<? xml version = "1.0" encoding="utf-8" standalone = "yes">
<StarMovieData>
    <Star starID="cf" starredIn="sw">
        <Name>Carrie Fisher</Name>
        <Address>
            <Street>123 Maple St.<Street>
            <City>Hollywood</City>
        </Address>
        <Address>
            <Street>5 Locust Ln.</Street>
            <City>Malibu</City>
        </Address>
    </Star>
    <Star starID="hf" starredIn="sw fw">
        <Name>Harrison Ford</Name>
    </Star>

    <Star starID="mh" starredIn="sw">
        <Name>Mark Hamill</Name>
        <Address>
            <Street>456 Oak Rd.</Street>
            <City>Brentwood</City>
        </Address>
    </Star>
    <Movie movieID="sw" starsOf="cf mh hf">
        <Title>Star Wars</Title>
        <Year>1977</Year>
    </Movie>
    <Movie movieID="esb" starsOf="cf mh hf">
        <Title>The Empire Strikes Back</Title>
        <Year>1980</Year>
    </Movie>
    <Movie movieID="roj" starsOf="cf mh hf">
        <Title>Return of the Jedi</Title>
        <Year>1983</Year>
    </Movie>
    <Movie movieID="fw" starsOf="hf">
        <Title>Firewall</Title>
        <Year>2006</Year>
    </Movie>
    <Movie movieID="hhs" starsOf="cf">
        <Title>Hannah and Her Sisters</Title>
        <Year>1985</Year>
    </Movie>
</StarMovieData>

```

d)
```
<? xml version = "1.0" encoding="utf-8" standalone = "yes">
<StarMovieData>
```

```
        <Star starID="cf" starredIn="sw">
            <Name>Carrie Fisher</Name>
            <Address>
                <Street>123 Maple St.<Street>
                <City>Hollywood</City>
            </Address>
            <Address>
                <Street>5 Locust Ln.</Street>
                <City>Malibu</City>
            </Address>
        </Star>
        <Star starID="hf" starredIn="sw fw">
            <Name>Harrison Ford</Name>
        </Star>
        <Star starID="mh" starredIn="sw">
            <Name>Mark Hamill</Name>
            <Address>
                <Street>456 Oak Rd.</Street>
                <City>Brentwood</City>
            </Address>
        </Star>
        <Star starID="md" starredIn="bi">
            <Name>Matt Damon</Name>
        </Star>
        <Movie movieID="sw" starsOf="cf mh hf">
            <Title>Star Wars</Title>
            <Year>1977</Year>
        </Movie>
        <Movie movieID="esb" starsOf="cf mh hf">
            <Title>The Empire Strikes Back</Title>
            <Year>1980</Year>
        </Movie>
        <Movie movieID="roj" starsOf="cf mh hf">
            <Title>Return of the Jedi</Title>
            <Year>1983</Year>
        </Movie>
        <Movie movieID="fw" starsOf="hf">
            <Title>Firewall</Title>
            <Year>2006</Year>
        </Movie>
        <Movie movieID="hhs" starsOf="cf">
            <Title>Hannah and Her Sisters</Title>
            <Year>1985</Year>
        </Movie>
        <Movie movieID="bi" starsOf="md">
            <Title>The Bourne Identity</Title>
            <Year>2002</Year>
        </Movie>
    </StarMovieData>

```

10.
```
    <!DOCTYPE BankData [
        <!ELEMENT BankData (Accounts*, Customers*)>
        <!ELEMENT Accounts(acctNo, type, balance)>
```

```
            <!ATTLIST Accounts
                acctNo ID #REQUIRED
                acctIn IDREFS #IMPLIED
            >
        <!ELEMENT Customers(firsrName, lastName, idNo, account)>
            <!ATTLIST Accounts
                idNo ID #REQUIRED
                account IDREFS #IMPLIED
            >
        <!ELEMENT acctNo (#PCDATA)>
        <!ELEMENT type (#PCDATA)>
        <!ELEMENT balance (#PCDATA)>
        <!ELEMENT firstName (#PCDATA)>
        <!ELEMENT lastName (#PCDATA)>
        <!ELEMENT idNo (#PCDATA)>
        <!ELEMENT account (#PCDATA)>
    ]>

```

Correct Solution:

```
    <!DOCTYPE BankData [
        <!ELEMENT BankData (Accounts*, Customers*)>
        <!ELEMENT Accounts(acctNo, type, balance)>
            <!ATTLIST Accounts
                acctNo ID #REQUIRED
                acctIn IDREFS #IMPLIED
            >
        <!ELEMENT Customers(firsrName, lastName, idNo, account)>
            <!ATTLIST Accounts
                idNo ID #REQUIRED
                account IDREFS #IMPLIED
            >
        <!ELEMENT acctNo (#PCDATA)>
        <!ELEMENT type (#PCDATA)>
        <!ELEMENT balance (#PCDATA)>
        <!ELEMENT firstName (#PCDATA)>
        <!ELEMENT lastName (#PCDATA)>
        <!ELEMENT idNo (#PCDATA)>
        <!ELEMENT account (#PCDATA)>
    ]>

```

11.
```
    <!DOCTYPE TeamData [
        <!ELEMENT TeamData (Players*, Teams*, Fans*)>
        <!ELEMENT Players(FirstName, LastName, Team)>
            <!ATTLIST Players
                playerId ID #REQUIRED
                playerIn IDREFS #IMPLIED
            >
        <!ELEMENT FirstName (#PCDATA)>
        <!ELEMENT LastName (#PCDATA)>
        <!ELEMENT Team (#PCDATA)>
        <!ELEMENT Teams(Name)>
            <!ATTLIST Teams
                teamId ID #REQUIRED
                playerOf IDREFS #IMPLIED
                fanOf IDREFS #IMPLIED
            >
        <!ELEMENT Name (#PCDATA)>
        <!ELEMENT Fans(FirstName, LastName)>
            <!ATTLIST Fans
                faId ID #REQUIRED
                fanIn IDREFS #IMPLIED
            >
    ]>

```

Notes:

- There is no need to write same tags twice. (e.g. FirstName, LastName)

12.

```
    <!DOCTYPE GenealogyData [
        <!ELEMENT GenealogyData (Person)>
        <!ELEMENT Person(FirstName, LastName, Age)>
            <!ATTLIST Person
                personId ID #REQUIRED
                spouse IDREFS #IMPLIED
                husband IDREFS #IMPLIED
                child IDREFS #IMPLIED
                brother IDREFS #IMPLIED
            >
        <!ELEMENT FirstName (#PCDATA)>
        <!ELEMENT LastName (#PCDATA)>
    ]>

```

Notes:

- PCDATA
  - Is text that will be parsed by a parser.
  - Tags inside the text will be treated as markup and entities will be expanded.
- CDATA
  - Is text that will not be parsed by a parser.
  - Tags inside the text will not be treated as markup and entities will not be expanded
  - e.g. attributes

```
    <Movie title="Star Wars" year="1977" genre="sciFi">

    <!ELEMENT Movie EMPTY>
        <!ATTLIST Movie
            title CDATA #REQUIRED
            year  CDATA #REQUIRED
            genre (comedy | drama | sciFi | teen) #IMPLIED
        >


```

13.
```
    <!DOCTYPE ShipBattleData [

    ]>

```

14. • An example of a document that conforms tothe XML Schema

```
    <? xml version = "1.0" encoding = "utf-8" ?>
    <Movies>
        <Movie>
            <Title>Star Wars</Title>
            <Year>1977</Year>
        </Movie>
        <Movie>
            <Title>Return of the Jedi</Title>
            <Year>1983</Year>
        </Movie>
    </Movies>

```

• An example of one that has all the elements mentioned, but does not conform to definition

```
    <? xml version = "1.0" encoding = "utf-8" ?>
    <Movies>
        <Movie>
            <Title>Star Wars</Title>
            <Year>1977</Year>
        </Movie>
        <Movie>
            <Title>Return of the Jedi</Title>
            <Year>This is not correct</Year>
        </Movie>
    </Movies>

```

Notes:

- XML Schema
  - is an alternative way to provide a schema for XML documents

  - is more powerful than DTD
    * allows to declare types such as integer or float
    * gives ability to declare keys and foreign keys
- The Form of an XML Schema
  - Each XML-Schema document has the form

    <? xml version = ”1.0” encoding = ”utf-8” ?>
    <xs:schema xmlns:xs=”http://www.w3.org/2001/XMLSchema”>
    ...
    </xs:schema>
- Elements
  - **Syntax:**

    <xs:element name = element name type = element type>
    constraints and / or structure information
    </xs:element>

    Example:

```
   <xs:element name = "title" type = "xs:string" />
   <xs:element name = "Year" type = "xs:integerg" />


```

- Complex Types
  - is an XML element that contains other elements and/or attributes. [1]
  - **Syntax:**

    <xs:complexType name = *type name*>
    <xs:sequence>
    list of element definitions
    </xs:sequence>
    </xs:complexType>

  Example:

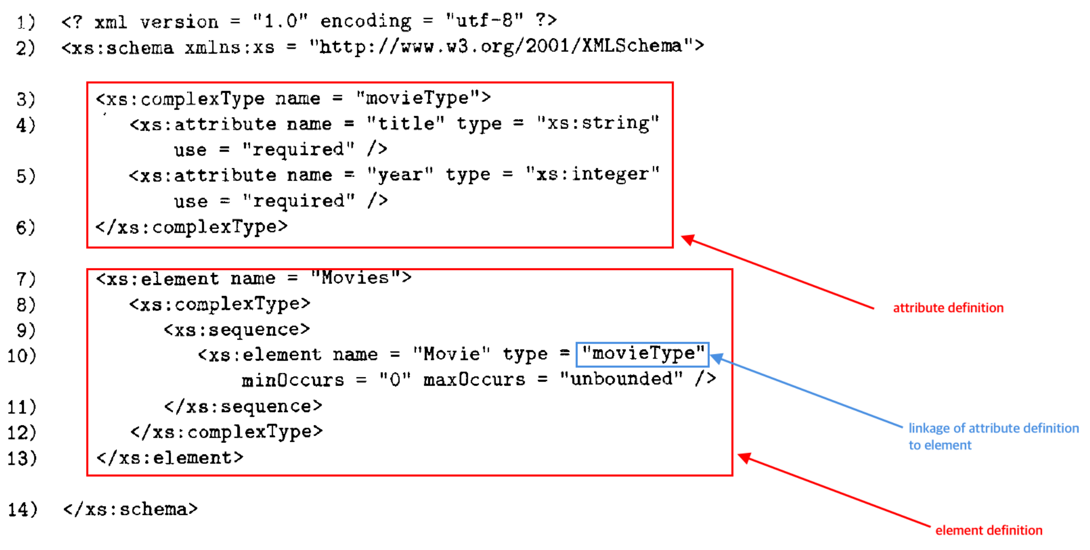
```
   <? xml version = "1.0" encoding = "utf-8" ?>
   <xs:schema xmlns:xs = "http://www.w3.org/2001/XMLSchema">

       <xs:complexType name = "movieType">
           <xs:sequence>
               <xs:element name = "Title" type = "xs:string" />
               <xs:element name = "year" type = "xs:integer" />
           </xs:sequence>
       </xs:complexType>
```

```

        <xs:element name = "Movies">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="Movie" type="movieType"
    minOccurs="0" maxOccurs = "unbounded" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:schema>

```

- Complex Types Attributes
  - **Syntax:**
    <xs:attribute name = *attribute name* type = *type name other information about the attribute* />

    - *other information* includes
      1. default value
      2. required
      3. type

  <u>Example:</u>



```
<? xml version = "1.0" encoding = "utf-8" ?>
<xs:schema xmlns:xs = "http://www.w3.org/2001/XMLSchema">

<xs:complexType name = "movieType">
   <xs:attribute name = "title" type = "xs:string"
       use = "required" />
   <xs:attribute name = "year" type = "xs:integer"
       use = "required" />
</xs:complexType>

<xs:element name = "Movies">
   <xs:complexType>
      <xs:sequence>
         <xs:element name = "Movie" type = "movieType"
             minOccurs = "0" maxOccurs = "unbounded" />
      </xs:sequence>
   </xs:complexType>
</xs:element>

</xs:schema>
```

attribute definition
linkage of attribute definition to element
element definition

Above allows us to have the tag <Movies title="value" year="1">

- Restricted Simple Types
  - Can be used as an attribute or element
    1. Restricting numerical values by <u>setting lower bound and upperbound</u>
       - **Syntax:**

         <xs:simpleType name = *type name*>

<xs:restriction base = base type >
upper and/or lower bounds
</xs:restriction >
</xs:simpleType>

∗ Use **minInclusive** to state the lower bound
∗ Use **maxInclusive** to state the upper bound

Example:

```
    <xs:simpleType name = "movieYearType">
        <xs:restriction base = "xs:integer">
            <xs:minInclusive value = "1915" />
        </xs:restriction>
    </xs:simpleType>

```

2. Restricting values to an enumerated type

   ∗ **Syntax:**

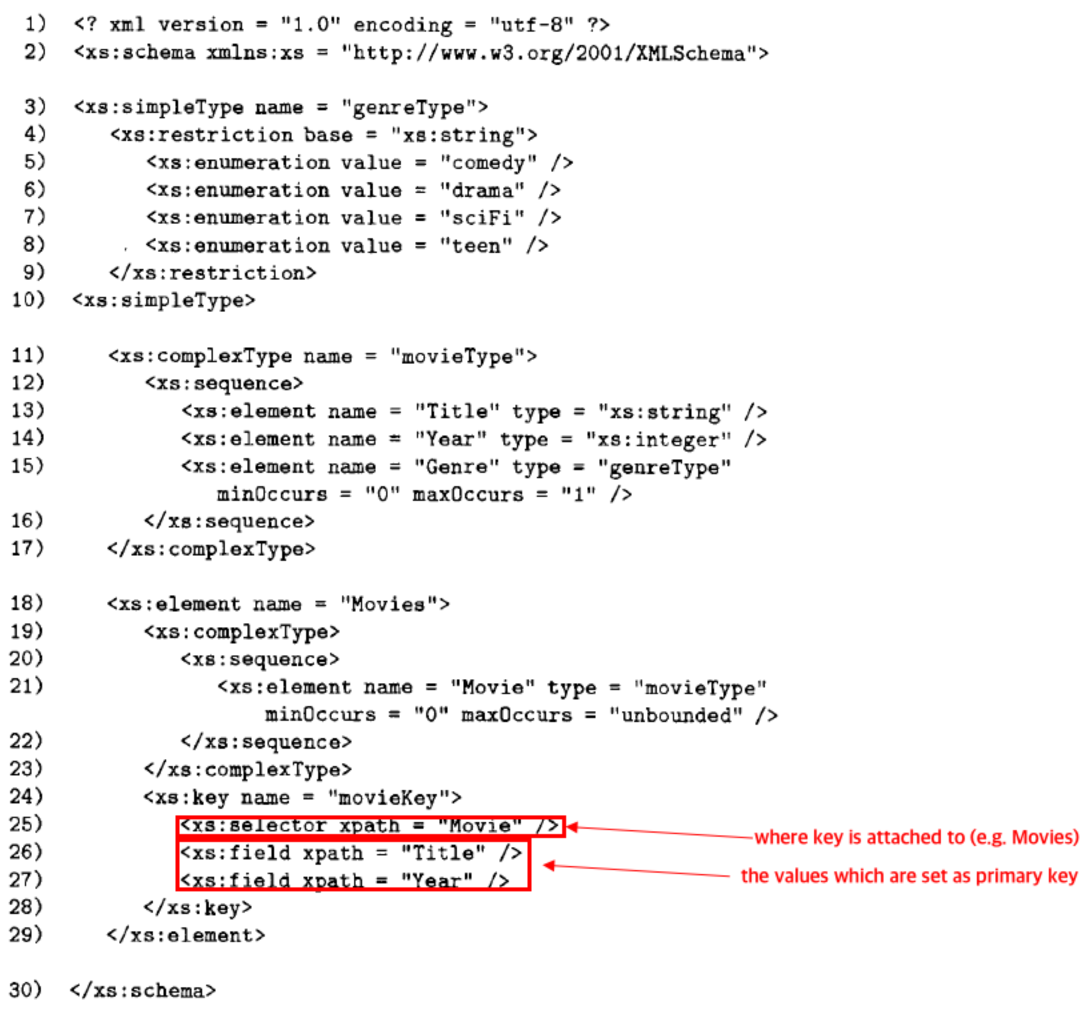
<xs:simpleType name = "movieYearType" >
<xs:restriction base = "xs:string">
<xs:enumeration value = "comedy"/ >
<xs:enumeration value = "drama"/ >
<xs:enumeration value = "sciFi"/ >
<xs:enumeration value = "teen"/ >
</xs:restriction>
</xs:simpleType>

Example:

```
    <xs:simpleType name = "genreType">
        <xs:restriction base = "xs:string">
            <xs:enumeration value = "comedy"/>
            <xs:enumeration value = "drama"/>
            <xs:enumeration value = "sciFi"/>
            <xs:enumeration value = "teen"/>
        </xs:restriction>
    </xs:simpleType>

```

- Keys in XML Schema
  - **Syntax:**
    <xs:key name = *key name*>
    <xs:selector xpath= *path description* >
    <xs:field xpath = *path description* >
    </xs:key>

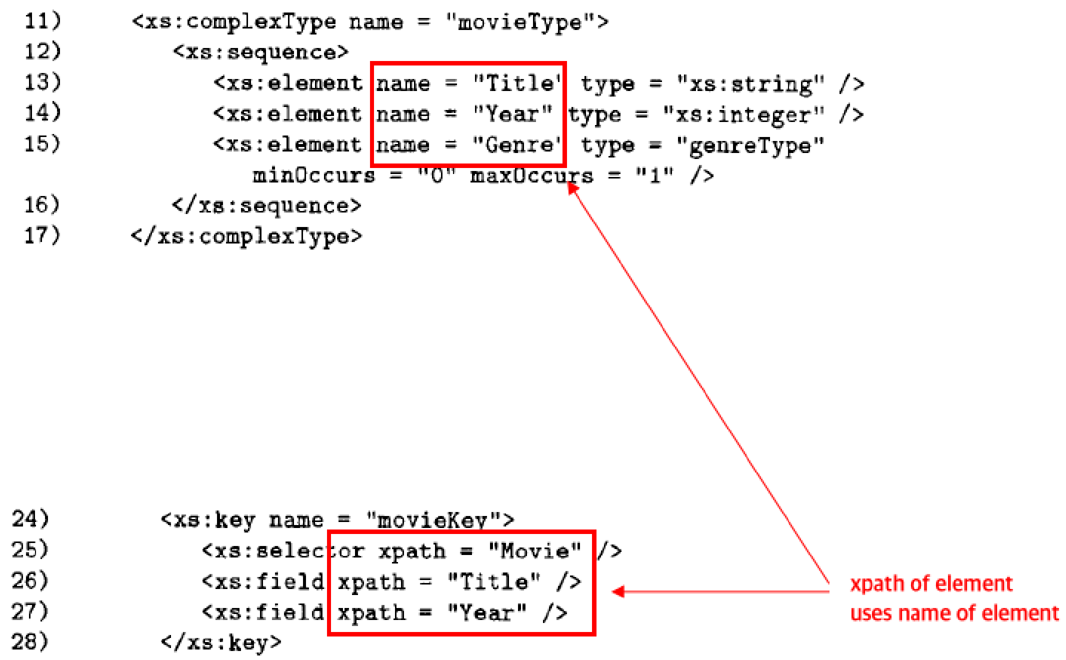
    Example:



```
<? xml version = "1.0" encoding = "utf-8" ?>
<xs:schema xmlns:xs = "http://www.w3.org/2001/XMLSchema">

<xs:simpleType name = "genreType">
   <xs:restriction base = "xs:string">
      <xs:enumeration value = "comedy" />
      <xs:enumeration value = "drama" />
      <xs:enumeration value = "sciFi" />
      <xs:enumeration value = "teen" />
   </xs:restriction>
<xs:simpleType>

   <xs:complexType name = "movieType">
      <xs:sequence>
         <xs:element name = "Title" type = "xs:string" />
         <xs:element name = "Year" type = "xs:integer" />
         <xs:element name = "Genre" type = "genreType"
            minOccurs = "0" maxOccurs = "1" />
      </xs:sequence>
   </xs:complexType>

   <xs:element name = "Movies">
      <xs:complexType>
         <xs:sequence>
            <xs:element name = "Movie" type = "movieType"
                minOccurs = "0" maxOccurs = "unbounded" />
         </xs:sequence>
      </xs:complexType>
      <xs:key name = "movieKey">
         <xs:selector xpath = "Movie" />
         <xs:field xpath = "Title" />
         <xs:field xpath = "Year" />
      </xs:key>
   </xs:element>

</xs:schema>
```

where key is attached to (e.g. Movies)
the values which are set as primary key

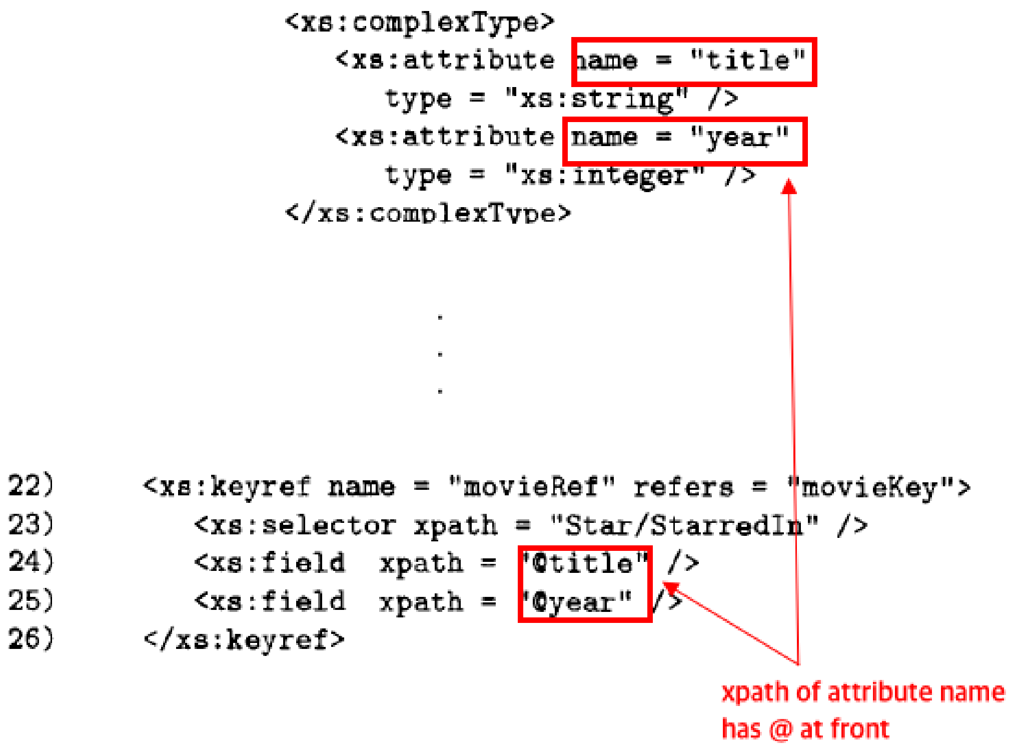
– XPath of element → *element name*



```
<xs:complexType name = "movieType">
   <xs:sequence>
      <xs:element name = "Title" type = "xs:string" />
      <xs:element name = "Year" type = "xs:integer" />
      <xs:element name = "Genre" type = "genreType"
         minOccurs = "0" maxOccurs = "1" />
   </xs:sequence>
</xs:complexType>
```

```
   <xs:key name = "movieKey">
      <xs:selector xpath = "Movie" />
      <xs:field xpath = "Title" />
      <xs:field xpath = "Year" />
   </xs:key>
```

– XPath of attribute → @*attribute name*



```
<xs:complexType>
   <xs:attribute name = "title"
      type = "xs:string" />
   <xs:attribute name = "year"
      type = "xs:integer" />
</xs:complexType>

.
.
.

<xs:keyref name = "movieRef" refers = "movieKey">
   <xs:selector xpath = "Star/StarredIn" />
   <xs:field  xpath = "@title" />
   <xs:field  xpath = "@year" />
</xs:keyref>
```

- Foreign Keys in XML Schema
  - **Syntax:**

    <xs:keyref name = *foreign-key name* refer = *key name* >
       <xs:selector xpath = *path description* >
       <xs:field xpath = *path description* >

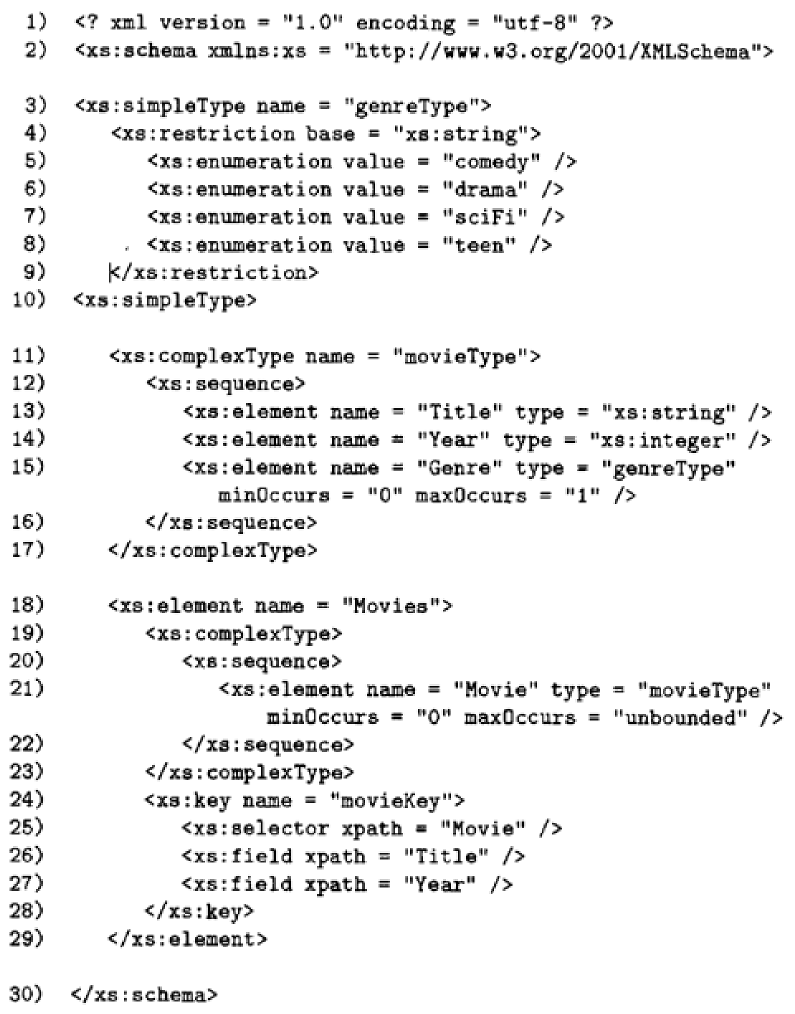
</xs:keyref >

References:

1. w3School : Complex Type, link

15.
```
<? xml version = "1.0" encoding = "utf-8" ?>
<xs:schema xmlns:xs = "http://www.w3.org/2001/XMLSchema">

    <xs:complexType name = "movieType">
        <xs:sequence>
            <xs:element name = "Title" type = "xs:string" />
            <xs:element name = "year" type = "xs:integer" />
        </xs:sequence>
    </xs:complexType>

    <xs:element name = "Movies">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="NotMovie" type="movieType"
minOccurs="0" maxOccurs = "unbounded" /> // <- Corrected
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

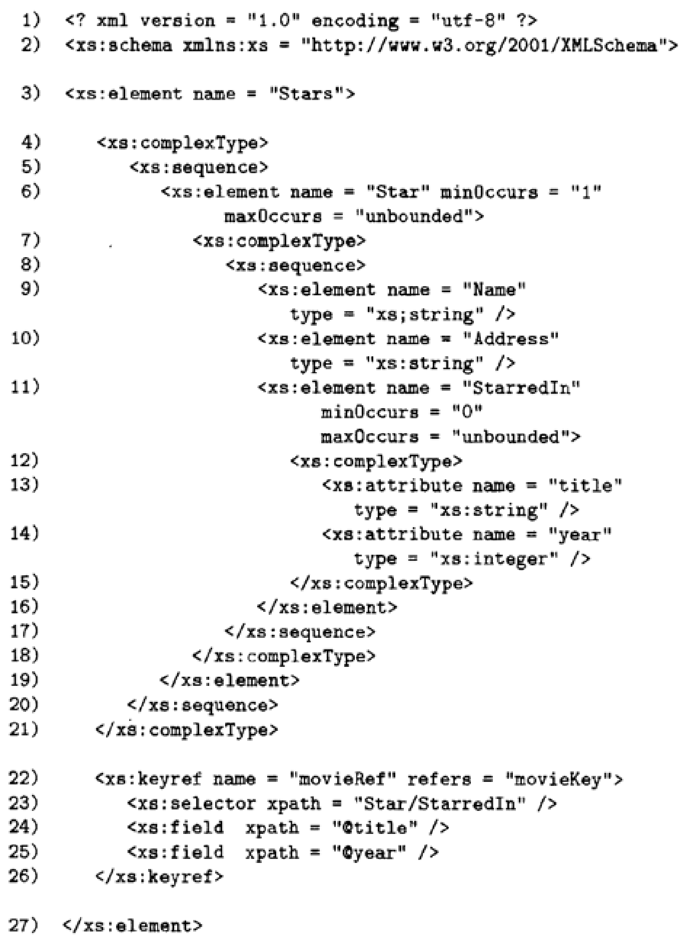
16. a) Converting schema in figure 19 to DTD



```
<? xml version = "1.0" encoding = "utf-8" ?>
<xs:schema xmlns:xs = "http://www.w3.org/2001/XMLSchema">

<xs:simpleType name = "genreType">
   <xs:restriction base = "xs:string">
      <xs:enumeration value = "comedy" />
      <xs:enumeration value = "drama" />
      <xs:enumeration value = "sciFi" />
      <xs:enumeration value = "teen" />
   </xs:restriction>
<xs:simpleType>

   <xs:complexType name = "movieType">
      <xs:sequence>
         <xs:element name = "Title" type = "xs:string" />
         <xs:element name = "Year" type = "xs:integer" />
         <xs:element name = "Genre" type = "genreType"
            minOccurs = "0" maxOccurs = "1" />
      </xs:sequence>
   </xs:complexType>

   <xs:element name = "Movies">
      <xs:complexType>
         <xs:sequence>
            <xs:element name = "Movie" type = "movieType"
                minOccurs = "0" maxOccurs = "unbounded" />
         </xs:sequence>
      </xs:complexType>
      <xs:key name = "movieKey">
         <xs:selector xpath = "Movie" />
         <xs:field xpath = "Title" />
         <xs:field xpath = "Year" />
      </xs:key>
   </xs:element>

<\xs:schema>
```

```
    <!DOCTYPE Movies [
        <!ELEMENT Movies(Movie*)>

        <!ELEMENT Movie (Title, Year, Genre)>
            <!ATTLIST Movie
                title CDATA #REQUIRED
                year CDATA #REQUIRED
                genre (comedy | drama | sciFi | teen) #IMPLIED
            >
    ]>

```

b) Converting schema in figure 20 to DTD



```
<? xml version = "1.0" encoding = "utf-8" ?>
<xs:schema xmlns:xs = "http://www.w3.org/2001/XMLSchema">

<xs:element name = "Stars">

   <xs:complexType>
      <xs:sequence>
         <xs:element name = "Star" minOccurs = "1"
                 maxOccurs = "unbounded">
            <xs:complexType>
               <xs:sequence>
                  <xs:element name = "Name"
                     type = "xs;string" />
                  <xs:element name = "Address"
                     type = "xs:string" />
                  <xs:element name = "StarredIn"
                        minOccurs = "0"
                        maxOccurs = "unbounded">
                     <xs:complexType>
                        <xs:attribute name = "title"
                           type = "xs:string" />
                        <xs:attribute name = "year"
                           type = "xs:integer" />
                     </xs:complexType>
                  </xs:element>
               </xs:sequence>
            </xs:complexType>
         </xs:element>
      </xs:sequence>
   </xs:complexType>

   <xs:keyref name = "movieRef" refers = "movieKey">
      <xs:selector xpath = "Star/StarredIn" />
      <xs:field  xpath = "@title" />
      <xs:field  xpath = "@year" />
   </xs:keyref>

</xs:element>
```

```
    <!DOCTYPE Stars [
        <!ELEMENT Stars(Star+)>

        <!ELEMENT Star (Name, Address, StarredIn*)>
        <!ELEMENT Name (#PCDATA)>
        <!ELEMENT Address (#PCDATA)>
        <!ELEMENT StarredIn EMPTY>
            <!ATTLIST StarredIn
                title ID #REQUIRED
                year ID #REQUIRED

            >
    ]>
```