Exam

SDXML
Models and languages for handling semi-structured data and XML

YYYY-MM-DD

- The exam consists of three parts:
  Part 1 Theory and modeling (questions 1, 2, 3)
  Part 2 Query languages for SSD and XML (questions 4, 5)
  Part 3 XML and relational databases (question 6)

- The exam is graded based on the grading scale F/Fx/E/D/C/B/A.

- In order to pass the exam, the student must perform at a certain level on each part and at a certain level in total, thus fulfilling all three course goals (which correspond to the three exam parts). The exact levels for each grade are:

	Whole exam	Per part
Minimums for A	92%	84%
Minimums for B	84%	73%
Minimums for C	76%	62%
Minimums for D	68%	51%
Minimums for E	60%	40%

  In the unlikely event that a student achieves a very uneven result (very good result, 85% or more, in one or two parts, without reaching the minimums for E), the student will be offered the option of skipping parts during the retake. This will be denoted by the grade Fx and will be explained in the feedback to the exam. Any student that chooses to use this option will be able to receive at most the grade E. Students that have been offered this option can ignore it. When this option is used, the skipped parts are excluded from the grade calculation and the minimums for E apply to the remaining parts.

- Specify the **course code**, the **date**, the **exercise number** and your **anonymous examinee code** on each submitted page!

- Fill out the scanning page with the **number of submitted pages** and specify **which questions that have been answered**!

- No aids allowed.

Reference

SQL

Syntax for SQL SELECT:

```
SELECT [DISTINCT] <column list>
FROM <table list>
[WHERE <conditions>]
[GROUP BY <column list>
  [HAVING <conditions>]]
[ORDER BY <column list>]
```

Common SQL/XML functions:
XMLELEMENT, XMLATTRIBUTES, XMLFOREST, XMLAGG, XMLCONCAT, XMLEXISTS, XMLQUERY, XMLTABLE, XMLCOMMENT, XMLPI, XMLCAST

XML Schema

Common elements:
schema, element, attribute, complexType, simpleType, group, all, sequence, choice, restriction, extension, simpleContent, complexContent

XQuery

Syntax for XQuery FLWOR:

```
for <loop variables>
let <variable assignments>
where <conditions>
(group by <grouping expressions>) only in XQuery 3
order by <sorting expressions>
return <result>
```

Common XQuery functions:
distinct-values(), count(), sum(), min(), max(), avg(), empty(), exists(), not(), data()

XSLT

Common elements:
transform, output, variable, template, for-each, for-each-group, apply-templates, call-template, if, choose, when, otherwise, element, attribute, comment, processing-instruction, value-of, text, copy, copy-of, sort, param, with-param

SQL Server SQL

SQL clause: FOR XML PATH | AUTO | RAW
Relevant keywords: ELEMENTS, ROOT, TYPE
XML methods: query, value, nodes, exist, modify
XQuery functions: sql:column, sql:variable

Question 1 (6 points)

Question 1 consists of 6 terms to be explained/defined in short. The answers only need to illustrate understanding. No need for any long essays.

Explain the following terms:

1. Valid XML
2. Infoset
3. processing-instruction
4. JSON Schema
5. shredding
6. SVG

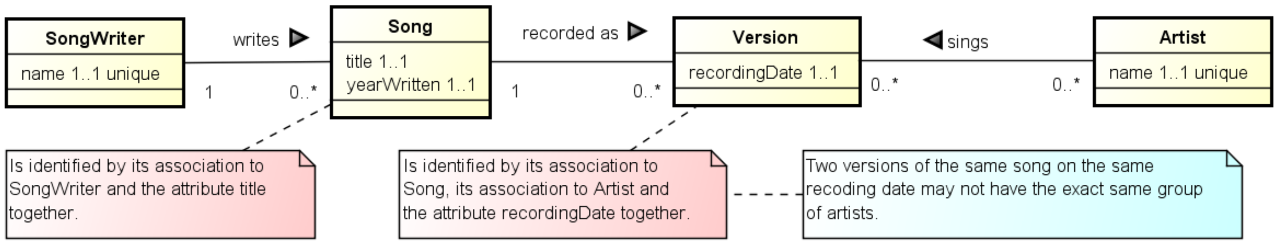
Question 2 (2 points)

Create a JSON object that conforms to following JSON Schema!

```
{
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "$id": "http://dsv.su.se/SDXML/jsonschema/song",
  "type": "object",
  "title": "A song",
  "description": "An object representation of a song",
  "properties": {
    "name": {"type": "string"},
    "lyrics": {"type": "string"},
    "year": {"type": "integer"},
    "writers": {
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "name": {"type": "string"},
          "yearofbirth": {"type": "integer"}
        },
        "additionalProperties": false,
        "required": ["name", "yearofbirth"]
      },
      "minItems": 1
    },
    "knownPerformers": {"type": "array", "items": {"type": "string"}}
  },
  "additionalProperties": false,
  "required": ["name", "year", "writers"]
}
```

Question 3 (3 points)

Construct an XML document (with sample data) and a corresponding XML Schema for representing the same information as the provided conceptual model! No need to explicitly define what is unique. The solution must consider integrity rules and avoid unnecessary redundancy.


SongWriter
name 1..1 unique
writes
1
0..*
Song
title 1..1
yearWritten 1..1
recorded as
1
0..*
Version
recordingDate 1..1
sings
0..*
0..*
Artist
name 1..1 unique
Is identified by its association to SongWriter and the attribute title together.
Is identified by its association to Song, its association to Artist and the attribute recordingDate together.
Two versions of the same song on the same recoding date may not have the exact same group of artists.

Question 4 (2 + 2 + 2 = 6 points)

Consider the XML document after question 5 that only contains sample data in order to illustrate the structure.

a) Write XQuery for the following:
Retrieve information about every member of parliament according to the following structure:

```
<Result>
    <MP Name="" Party="" NumberOfCommittees=""/>
    <MP Name="" Party="" NumberOfCommittees=""/>
    <MP Name="" Party="" NumberOfCommittees=""/>
</Result>
```

b) Write XQuery for the following:
Retrieve information about each committee according to the following structure!

```
<Result>
    <Committee Name="">
        <Party Name="" NumberOfMembers=""/>
        <Party Name="" NumberOfMembers=""/>
        <Party Name="" NumberOfMembers=""/>
    </Committee>
    <Committee Name="">
        <Party Name="" NumberOfMembers=""/>
        <Party Name="" NumberOfMembers=""/>
    </Committee>
</Result>
```

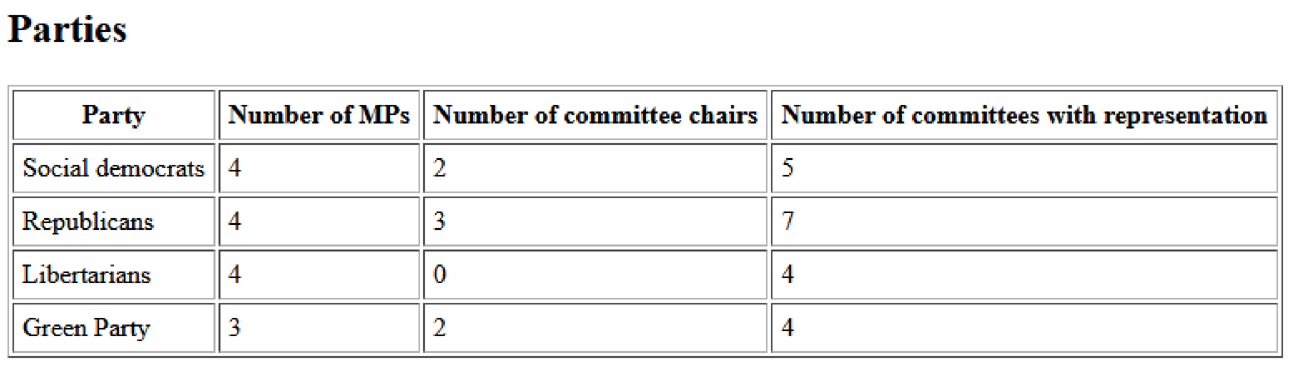
Note: NumberOfMembers shall be the number of MPs of the party in the committee independent of their role.

c) What is the result of the following XQuery expression with the given sample data?

```
element U {distinct-values(//*[@*/name() = "name"]/child::*/child::*/@*/name())}
```

Question 5 (3 points)

Consider the XML document on the next page that only contains sample data to illustrate the structure!
Write an XSLT that presents the data as html in the following fashion:



Parties

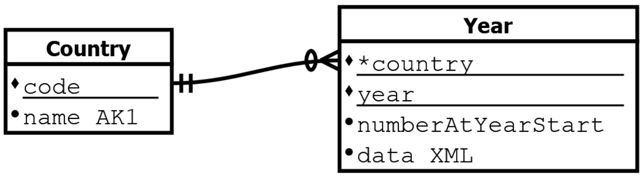
Party	Number of MPs	Number of committee chairs	Number of committees with representation
Social democrats	4	2	5
Republicans	4	3	7
Libertarians	4	0	4
Green Party	3	2	4

XML document for question 4 and question 5

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<ParliamentDB>
    <Party name="Social democrats">
        <MP name="Johan Löfstrand" birthyear="1976">
            <Assignment role="chair" committee="Environment committee"/>
            <Assignment role="member" committee="Finance committee"/>
        </MP>
        <MP name="Hillevi Larsson" birthyear="1974">
            <Assignment role="alternate" committee="Defense committee"/>
            <Assignment role="member" committee="Culture committee"/>
        </MP>
        <MP name="Monica Green" birthyear="1959">
            <Assignment role="chair" committee="Infrastructure committee"/>
            <Assignment role="alternate" committee="Culture committee"/>
        </MP>
        <MP name="Arhe Hamednaca" birthyear="1953"/>
    </Party>
    <Party name="Republicans">
        <MP name="Ewa Thalén Finné" birthyear="1959">
            <Assignment role="member" committee="Environment committee"/>
            <Assignment role="member" committee="Finance committee"/>
        </MP>
        <MP name="Jessika Roswall" birthyear="1972">
            <Assignment role="member" committee="Defense committee"/>
            <Assignment role="chair" committee="Justice committee"/>
        </MP>
        <MP name="Cecilia Widegren" birthyear="1973">
            <Assignment role="chair" committee="Taxation committee"/>
            <Assignment role="member" committee="Infrastructure committee"/>
            <Assignment role="member" committee="Culture committee"/>
        </MP>
        <MP name="Maria Stockhaus" birthyear="1963">
            <Assignment role="alternate" committee="Justice committee"/>
            <Assignment role="chair" committee="Defense committee"/>
        </MP>
    </Party>
    <Party name="Libertarians">
        <MP name="Roger Hedlund" birthyear="1979">
            <Assignment role="member" committee="Environment committee"/>
            <Assignment role="alternate" committee="Education committee"/>
        </MP>
        <MP name="Angelika Bengtsson" birthyear="1990">
            <Assignment role="alternate" committee="Defense committee"/>
            <Assignment role="alternate" committee="Culture committee"/>
        </MP>
        <MP name="Linus Bylund" birthyear="1978"/>
        <MP name="Dennis Dioukarev" birthyear="1993" />
    </Party>
    <Party name="Green Party">
        <MP name="Jonas Eriksson" birthyear="1967"/>
        <MP name="Emma Hult" birthyear="1988">
            <Assignment role="member" committee="Defense committee"/>
            <Assignment role="member" committee="Justice committee"/>
            <Assignment role="chair" committee="Culture committee"/>
        </MP>
        <MP name="Annika Hirvonen" birthyear="1989">
            <Assignment role="chair" committee="Finance committee"/>
            <Assignment role="member" committee="Culture committee"/>
        </MP>
    </Party>
</ParliamentDB>
```

Question 6 (2 + 2 + 2 + 2 = 8 points)

Consider the following relational database model!



Comments:
The column data is of the data type XML and accepts data that conform to the rules in the following DTD with root element PopulationChanges. All columns are defined as NOT NULL.

```
<!ELEMENT PopulationChanges (NrOfImmigrants, NrOfEmigrants, NrOfBirths, NrOfDeaths+)>
<!ELEMENT NrOfImmigrants (#PCDATA)>
<!ELEMENT NrOfEmigrants (#PCDATA)>
<!ELEMENT NrOfBirths (#PCDATA)>
<!ELEMENT NrOfDeaths (#PCDATA)>
<!ATTLIST NrOfDeaths Cause CDATA #REQUIRED>
```

There are never two NrOfDeaths with the same cause. There is always one NrOfDeaths with cause "natural". All other values (the text nodes) are integers.

Write **standard SQL** for the following:

a) Retrieve information about the total immigration to Sweden, Denmark, Finland, Norway and Iceland! The result shall have one row per year and the following columns: Year, NumberOfImmigrants.

b) Retrieve the countries where no murders (cause of death: murder) occurred in 2017! The result shall have the following structure:
```
<Result>
    <Country>
        <Code>the country code</Code><Name>the name of the country</Name>
    </Country>
    <Country>
        <Code>the country code</Code><Name>the name of the country</Name>
    </Country>
</Result>
```

c) Retrieve information about countries with less immigrants than emigrants in 2019! The result shall have the following structure:
```
<Result>
    <Country code="" name="" nrOfImmigrantsIn2019=""/>
    <Country code="" name="" nrOfImmigrantsIn2019=""/>
    <Country code="" name="" nrOfImmigrantsIn2019=""/>
</Result>
```

Write **SQL Server SQL** for the following:

d) Retrieve the countries where no murders (cause of death: murder) occurred in 2017! The result shall have the following structure:
```
<Result>
    <Country code="">the name of the country</Country>
    <Country code="">the name of the country</Country>
    <Country code="">the name of the country</Country>
</Result>
```