OPEN DATA EXAM

16th of June 2017. *The exam will take 2 hours. Answer each question in the provided space.* Answers out of such space will not be considered.

Name: ……

Question 1. [2p]

From a theoretical point of view, a CDM must provide high **expressiveness** and deal with **semantic relativeness**. We have seen several (graph) data models: the graph property data model, RDF and RDF(S)/OWL (i.e., data models providing formal semantics).

Use the table below to compare them with regard to the desirable properties of a CDM. **For each column** order the four models according to how good (4) or how bad (1) they perform in that criteria. The values must range between 1 and 4. Partial orders are possible. Below the table, for each criteria, briefly **justify your answer**:

	Instances & classes	**Rich relationships**	**Arbitrary constraints**	**Rich algebra**	**One basic structure**	**Multiple semantics**
Property GDM	4	4	4	3	1	1
RDF	1	1	1	2	4	3
RDF(S)	2	2	2	3	4	3
OWL	3	3	3	4	2	4

Instances and classes: How well the model supports defining instances and classes.
1) RDF is primarily a data model for representing info but doesn't inherently support rich class definitions without extensions like RDFS or OWL. 2) RDFS provides basic support for classes and instances but lacks the depth of OWL. 3) OWL supports detailed ontologies with extensive class hierarchies and instance definitions. 4) GDM supports rich definitions of instances and classes, often with complex structures

Rich relationships: How well the model supports defining complex relationships.

1) RDF supports basic relationships but lacks the expressiveness of RDFS and OWL.
2) RDFS supports hierarchical relationships and basic constraint.
3) OWL supports complex relationships, including transitive, symmetric, and inverse properties.
4) GDM supports complex relationships and nested structures

Arbitrary constraints: How well the model supports defining constraints.
1) RDF lacks support for arbitrary constraints without extensions.
2) RDFS supports some constraints, but not as extensively as OWL.
3) OWL supports detailed constraints, including cardinality and value restrictions.
4) GDM often supports arbitrary constraints due to its flexibility.

Rich algebra: How well the model supports operations and expressions for querying.
1) RDF supports basic operations for querying but lacks the rich algebra of OWL.
3) RDFS supports more operations than RDF, but less than OWL.
3) GDM support varies depending on the specific implementation, but generally supports rich operations.
4) OWL has rich support for logical operations and reasoning.

One basic structure: How well the model maintains simplicity for data representation.

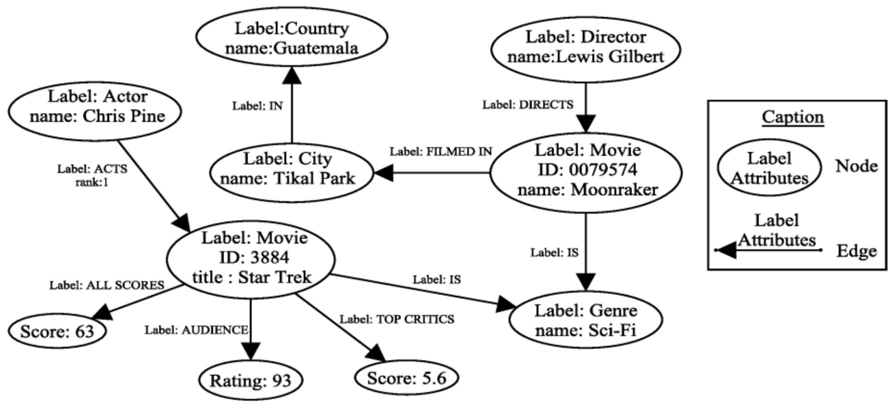
1) GDM often uses more complex structures which may not adhere to a single basic format.
2) OWL extends RDF with more complex structures and constructs.
4) RDF is designed with one basic triple structure (subject-predicate-object).
4) RDFS builds on RDF's basic structure with additional schema vocabulary.

Multiple semantics: How well the model supports different data interpretations.

1) GDM typically has a fixed semantic interpretation depending on the implementation.
2) RDF allows for some level of multiple semantics through extensions and additional vocabularies.
3) RDFS provides more semantics than RDF alone, but less than OWL.
4) OWL is designed to support multiple semantics, including different interpretations and reasoning.

Question 2. [2p]

Given the following property graph:



What **graph operation(s)** would you need to answer each of the following queries? **Justify your answer**:

- Has any movie by Lewis Gilbert been shot in Barcelona?

(:Director {name: Lewis Gilbert})-[:DIRECTS]->(m:Movie)-[:FILMED IN]->(:City {name: Barcelona})

Reachability

- In how many movies has Chris Pine participated?

Adjacency (:Actor {name: Chris Pine})-[:ACTED]->(m:Movie)

- List all movies with two or more directors

Adjacency (d:Director)-[:DIRECTS]->(m:Movie)

- What is the movie with the best audience rating ever shot in Guatemala?

(:Country {name: Guatemala})<-[:IN]-(:City)<-[:FILMED IN]-(m:Movie)-[:AUDIENCE]->(:Rating)

Pattern matching

Question 3. [5p]

a) Model in RDF(S) the following statements:

"An insurance company has three kinds of insurances: car, housing and life. All of them must have the mandatory following information: holder, insured, starting date, ending date (if cancelled). Holder and insured must be mandatorily persons. Besides that, a car insurance must provide information about the car model (including brand, cc and date of purchase); housing insurances information about the house (including address, city and total insurable value). Additionally, for life insurances, the holder and insured person (only one) must be the same. "

Draw a RDF(S) **TBOX modelling as many statements as you can**. Identify the statements modelled in your graph by underlining them in the text above. Clearly identify in the graph the RDFS constructs. Also clearly distinguish concepts (in rectangles) from literals (in circles). Define your own namespace prefix for the URIs you need to create **[1,5p]**

b) Now, **assert the following instances** to your RDF graph (draw them below the RDF graph sketched above and clearly **separate them from the TBOX by a dashed line**): *John has filed a housing insurance for his house at 123, Avenue Roosevelt, New York City with a total insurable value of 340.000$* **[0,5p]**

c) For those elements in the statement that you could not express in RDF/RDF(S) express them in DL. Assume the URIs you created to assert the DL axioms. **[1p]**

```
_:a rdfs:subClassOf owl:Restriction
_:a owl:onProperty :holder
_:a owl:hasValue [
     _:a owl:Individual ;
     owl:sameAs [
 a owl:Individual ;
 owl:onProperty :insured
     ]
]
exam:LifeInsurance rdfs:subClassOf _:a
```

Is there any statement that you could yet not represent?

d) Assume a RDFS entailment regime for the RDF graph you just created. Represent the inferred knowledge in the form of triples **and** for each of them justify from what RDFS semantic construct it is inferred. **[1,5p]**

2 loại: Core-type inference và Domain-specific inference

:John :holder :LiIn1 => :John rdf:type :Person; :LiIn1 rdf:type :LifeInsurance
:John :name 'John' => :John rdf:type :Person
:LiIn1 :address '123 Avenue Roosevelt' => :LiIn1 rdf:type :LifeInsurance
:LiIn1 :city 'New York City' => :LiIn1 rdf:type :LifeInsurance
:LiIn1 :totalValue '$340,000' => :LiIn1 rdf:type :LifeInsurance

e) Is there any difference between these two assertions? Justify your answer **[0,5p]**

Option 1	Option 2
:takes rdfs:Domain :Person	:Person :takes :Insurance
:takes rdfs:range :Insurance	

Schema-Level vs. Instance-Level:
- Option 1 is a schema-level assertion (TBox), defining the domain and range of the :takes property.
- Option 2 is an instance-level assertion (ABox), stating a specific relationship between two individuals.

Purpose:
- Option 1 is used to define the types of subjects and objects that can be connected by the :takes property.
- Option 2 is used to declare that a particular :Person takes a particular :Insurance.

Implications:
- Option 1 imposes constraints on the use of the :takes property across the entire dataset.
- Option 2 is a single factual statement about specific instances in the dataset

Question 4. [1p]

Considering the *Open-World Assumption*, does the following ABOX entails the certain answer `Query(Brussels)`? **Justify your answer [1p]**

```
Entrusts(Brussels,Milano)
Entrusts(Brussels,Paris)
Entrusts(Milano,Paris)
Entrusts(Paris,Barcelona)
inDebt(Milano)
¬inDebt(Barcelona)
```

```
Query ≡ ∃Entrusts.(inDebt ⊓ ∃Entrusts.¬inDebt)

ABox ⊨ Query(Brussels)?
```

……

……

……

……

……