RDFS

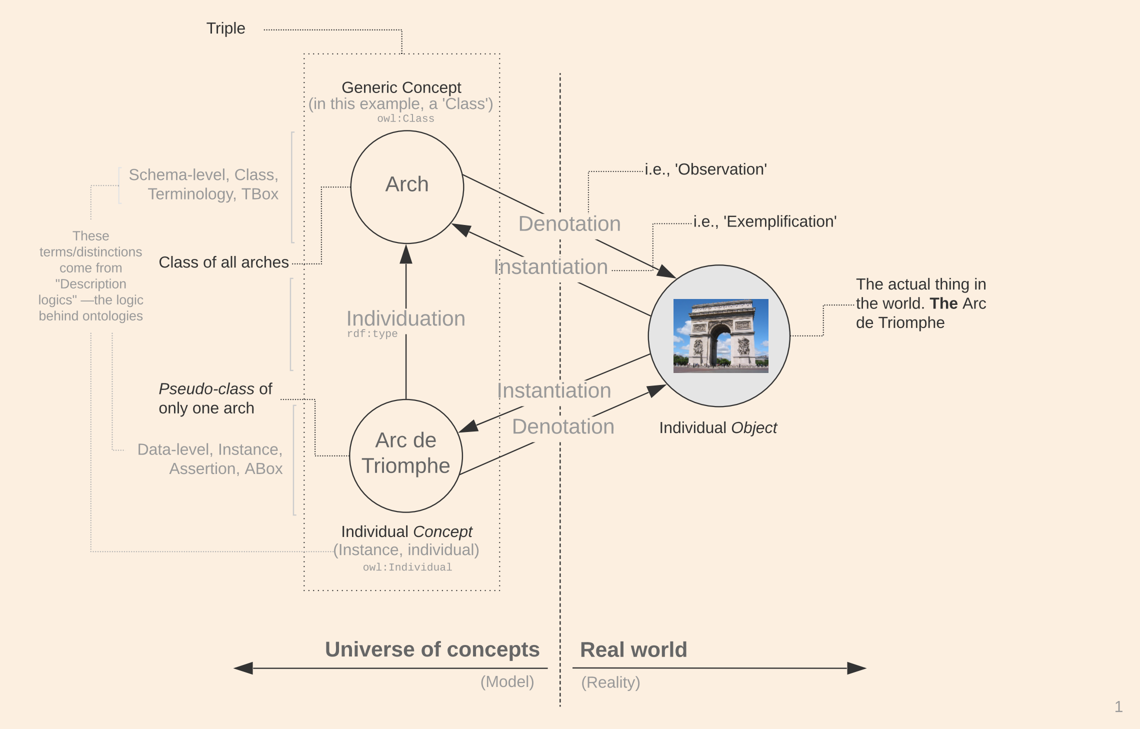
Schema, Semantics, Vocabulary, and Usage

Sheet 28

SCHEMA

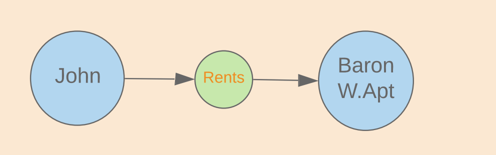
Knowledge Representation

Class, Type, Instance, Individual, Denotation, Concept, Object



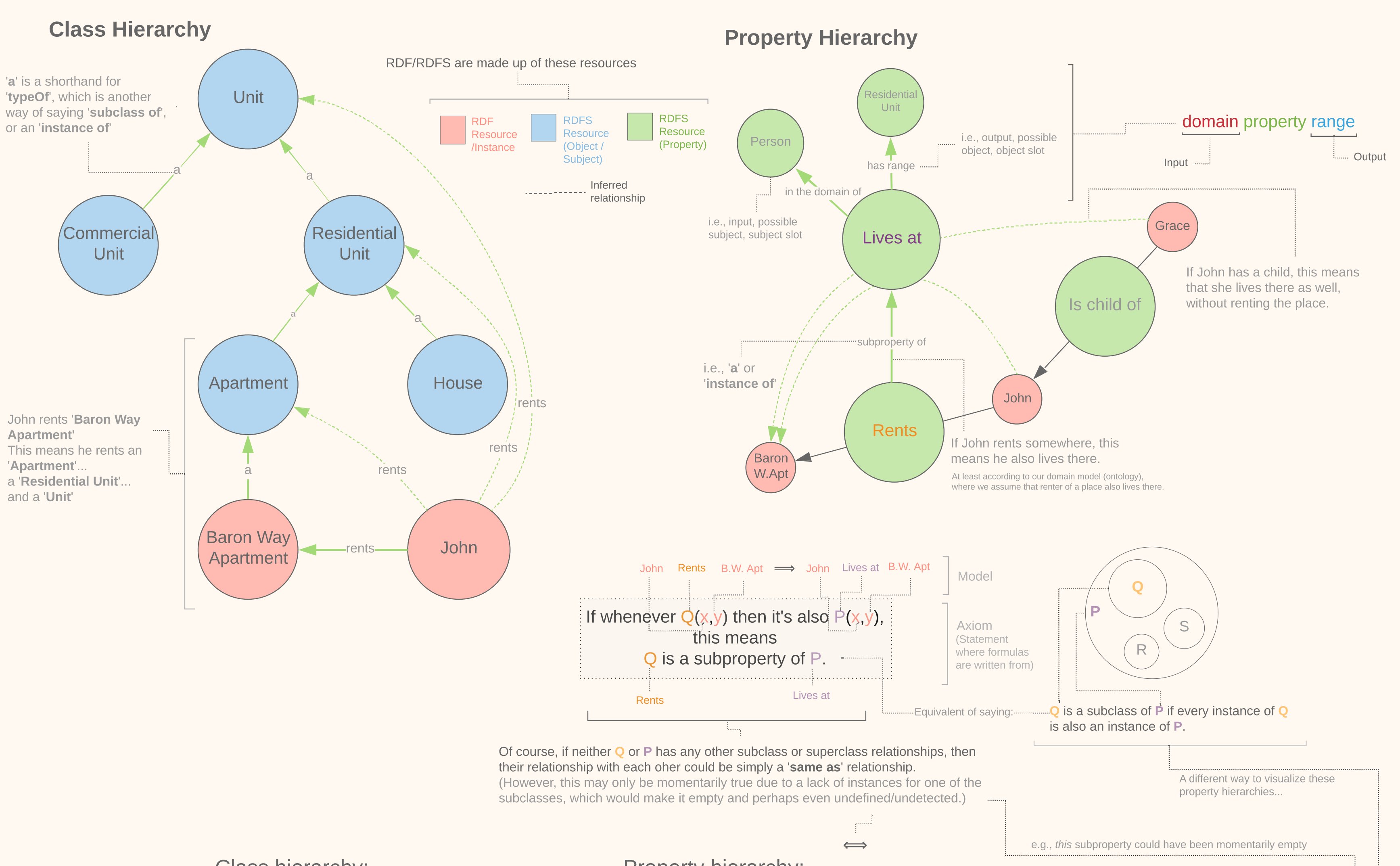
RDF

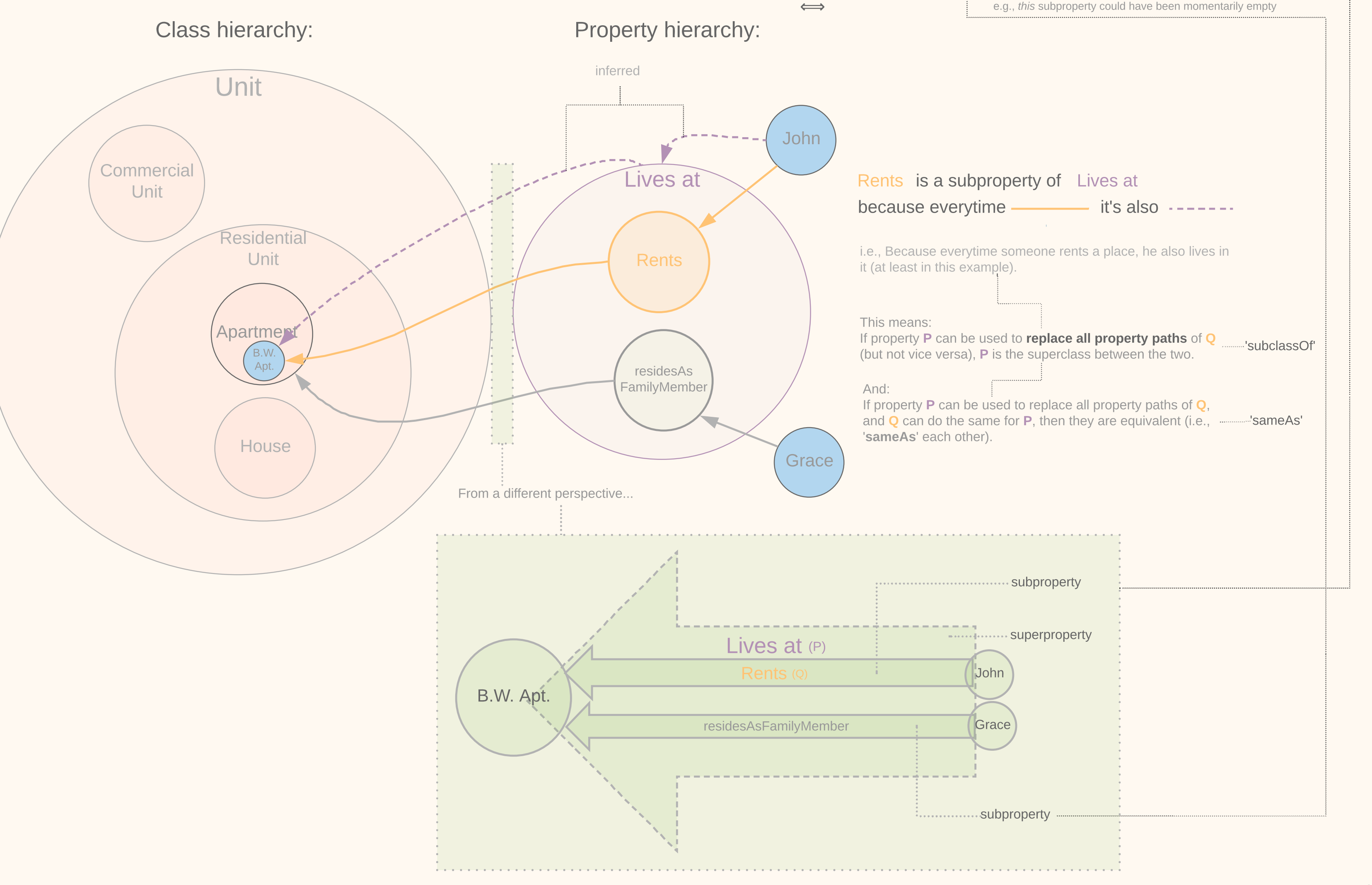
Resource description framework



RDFS (RDF Schema)

RDFS is a basic ontology for formally defining relationships in RDF. It is more limited, but also more efficient (e.g., when calculating entailments) than full-fledged ontology languages such as OWL.



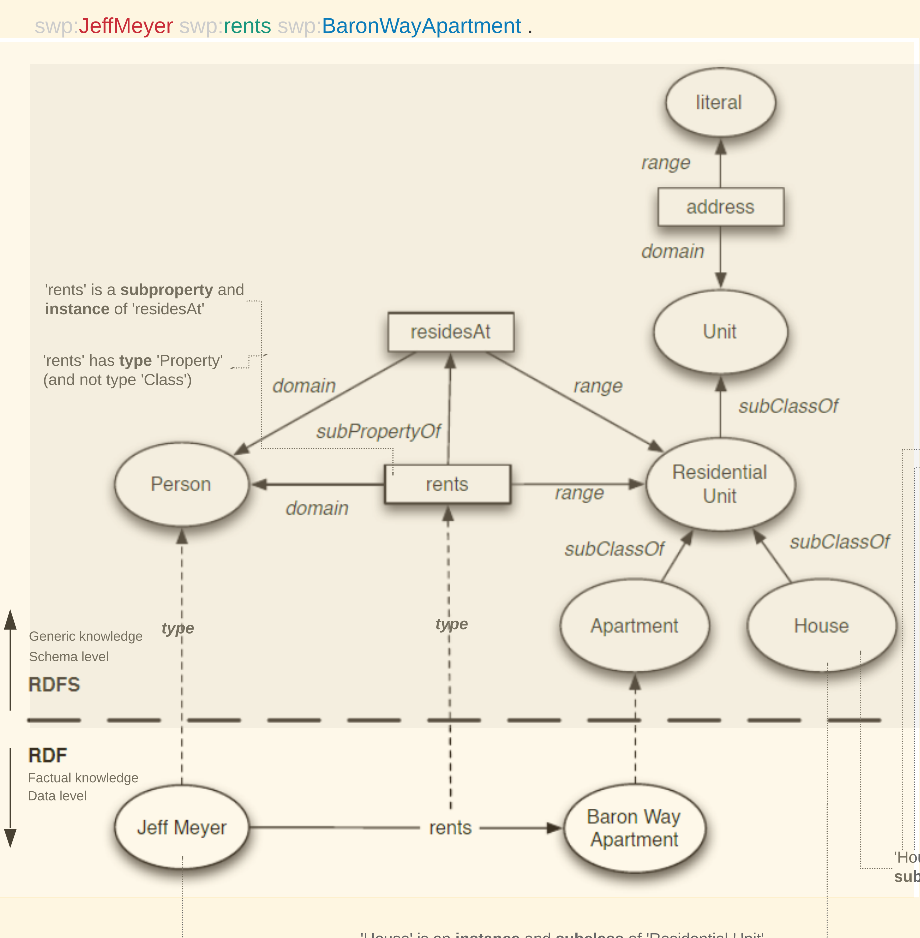


RDFS Example

An example RDF statement

@prefix swp:<http://www.semanticwebprimer.org/ontology/apartments.ttl#>.
@prefix rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#>.

swp:JeffMeyer swp:rents swp:BaronWayApartment .



An example RDFS

@prefix swp:<http://www.semanticwebprimer.org/ontology/apartments.ttl#>.
@prefix rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#>.

swp:Person	rdf:type	rdfs:Class .
swp:Person	rdfs:comment	"The class of people" .
swp:Unit	rdf:type	rdfs:Class .
swp:Unit	rdfs:comment	"A self contained section of accommodations in a larger building or group of buildings." .
swp:ResidentialUnit	rdf:type	rdfs:Class .
swp:ResidentialUnit	rdfs:subClassOf	swp:Unit .
swp:ResidentialUnit	rdfs:comment	"The class of all units or places where people live." .
swp:Apartment	rdf:type	rdfs:Class .
swp:Apartment	rdfs:subClassOf	swp:ResidentialUnit .
swp:Apartment	rdfs:comment	"The class of apartments" .
swp:House	rdf:type	rdfs:Class .
swp:House	rdfs:subClassOf	swp:ResidentialUnit .
swp:House	rdfs:comment	"The class of houses" .
swp:residesAt	rdf:type	rdfs:Property .
swp:residesAt	rdfs:comment	"Relates persons to their residence" .
swp:residesAt	rdfs:domain	swp:Person .
swp:residesAt	rdfs:range	swp:ResidentialUnit .
swp:rents	rdf:type	rdfs:Property .
swp:rents	rdfs:comment	"... It inherits its domain (swp:Person) and range (swp:ResidentialUnit) from its superproperty (swp:residesAt)." .
swp:rents	rdfs:subPropertyOf	swp:residesAt .
swp:address	rdf:type	rdfs:Property .
swp:address	rdfs:comment	"Is a property of units and takes literals as its value" .
swp:address	rdfs:domain	swp:Unit .
swp:address	rdfs:range	rdf:Literal .

'House' has **type** 'Class' and is a **subclassOf** 'Residential Unit'

'House' is an **instance** and **subclass** of 'Residential Unit'

'Jeff Meyer' or 'Baron Way Aparment' is an **instance** (but **not a subclass**) of 'Person'

'**type**' property defines the relationship between **class** and **instance**.

'**subclass**' property defines the relationship between **class** and **superclass** (and not between class and instance).

Example in Action: Class-based (weak) Inference with RDFS

Turtle File:

```
@prefix fb: <https://graph.facebook.com/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

fb:594486635        fb:likes       fb:289482145973 , fb:109327925759703.

fb:109327925759703  fb:name        "Waking Life"@en ;
                    fb:category    fb:movie .

fb:32985985640      fb:name        "Zeitgeist"@en ;
                    fb:category    fb:community .

fb:movie            rdfs:label     "Movie"@en .
fb:community        rdfs:label     "Community"@en .

fb:community        rdfs:subClassOf     fb:movie .
fb:category         rdfs:subPropertyOf  rdf:type .
```

SPARQL Query:

```
PREFIX fb: <https://graph.facebook.com/>

SELECT ?title WHERE {
    ?movie a        fb:movie ;
           fb:name  ?title .
}
```

Query Result:

?title
"Waking Life"@en
"Zeitgeist"@en

This triple allowed Zeitgeist to be inferred as a movie (as it specified that each community is also a movie). Without this line, only 'Waking Life' would have returned.

4

VOCABULARY

RDF, RDFS and Other Vocabularies

Resource Description Framework
Resource Description Framework Schema

prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>

RDF/RDFS Classes

Core Classes	Container Classes
rdfs:Resource	*rdf*:Bag
rdfs:Class	*rdf*:Seq
rdfs:Literal	*rdf*:Alt
rdfs:Property	rdfs:Container
rdfs:Statement	
rdfs:Datatype	

RDF/RDFS Properties

Defining Relationships	Restricting Properties	Utility	Reification
rdf:type	rdfs:domain	rdfs:seeAlso	*rdf*:subject
rdfs:subClassOf	rdfs:range	rdfs:isDefinedBy	*rdf*:predicate
rdfs:subPropertyOf		rdfs:comment	*rdf*:object
		rdfs:label	

```
geo:Amsterdam     rdf:type   rdf:Resource .
geo:containedIn   a          rdf:Property .
rdf:type          a          rdf:Property .
```

FOAF

Friend of a Friend

prefix foaf: <http://xmlns.com/foaf/0.1/>

Important Elements
foaf:Person
foaf:name
foaf:mbox
foaf:knows

```
vu:Rinke_Hoekstra  rdf:type   foaf:Person ;
                   foaf:knows vu:Albert_Merono ;
                   foaf:name  "Rinke Hoekstra" .
```

DC/DCT

Dublin Core / Dublin Core Terms

prefix dct: <http://purl.org/dc/terms>

Important Elements
dct:creator
dct:created
dct:description
dct:title

```
vu:SWPrimer  dct:creator  vu:Rinke_Hoekstra ;
             dct:created  "2012"^^xsd:date.
             dct:title    "A Semantic Web ..." .
```

SKOS

Simple Knowledge Organization System

prefix skos: <http://www.w3.org/2004/02/skos/core#>

Important Elements
skos:exactMatch

Good for using things that are *almost* the same. This is an informal way of establishing sameness.

```
fb:594486635        fb:likes         fb:105638652803531 .

fb:105638652803531  skos:exactMatch  imdb:tt0120669 .
```

For common prefixes and their namespaces:
http://prefix.cc

Social Side of Semantics

Berners-Lee: Using shared vocabularies is the key to establishing consistent semantics.

URI's may not always be unique. They are not a global identification service, but simply strings.

Rule of Thumb: "Only create vocabularies/URIs in a namespace within your own domain." (Important for trust).

Example in Action: Linking Vocabularies

Turtle File:

```
@prefix fb: <https://graph.facebook.com/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

fb:594486635        fb:likes       fb:289482145973 , fb:109327925759703.

fb:109327925759703  fb:name        "Waking Life"@en ;
                    fb:category    fb:movie .

fb:32985985640      fb:name        "Zeitgeist"@en ;
                    fb:category    fb:community .

fb:movie            rdfs:label     "Movie"@en .
fb:community        rdfs:label     "Community"@en .

fb:community        rdfs:subClassOf     fb:movie .
fb:category         rdfs:subPropertyOf  rdf:type .
```

This is Facebook's way of definig a 'type' relationship. It is used instead of 'rdf:type'.

This is triple aligns Facebook and RDF schema by bringing Facebook's 'category' property in line with 'type' property of of RDF.

Without this triple, results of the example SPARQL would have been empty, as there would not be any 'rdf:type' properties in the Turtle file. However, now that 'fb:category' is related to 'rdf:type', asking for 'rdf:type' in SPARQL also returns all categories.

SPARQL Query:

```
PREFIX fb: <https://graph.facebook.com/>

SELECT ?title WHERE {
    ?movie a        fb:movie ;
           fb:name  ?title .
}
```

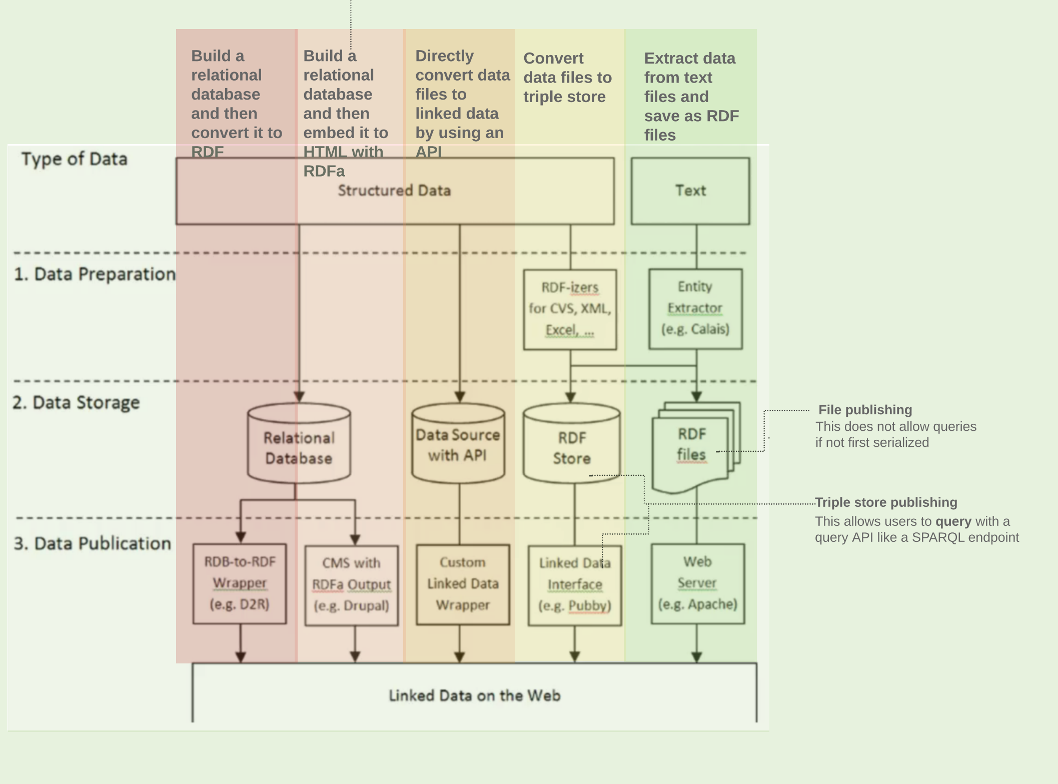
Query Result:

?title
"Waking Life"@en
"Zeitgeist"@en

5

USE CASES

Accessing and Publishing RDF & Linked Data



File publishing
This does not allow queries if not first serialized

Triple store publishing
This allows users to **query** with a query API like a SPARQL endpoint

6

Illustration: John C. Lokman
Produced in collaboration with Klaas Andries de Graaf

Image Credits: Lion by Yu luck from the Noun Project, Animal by Alina Oleynik from the Noun Project, Animal footprints by Gagana from the Noun Project