There is more to life than SQL!

Veronika Heimsbakk

There is more to life than SQL!

Veronika Heimsbakk

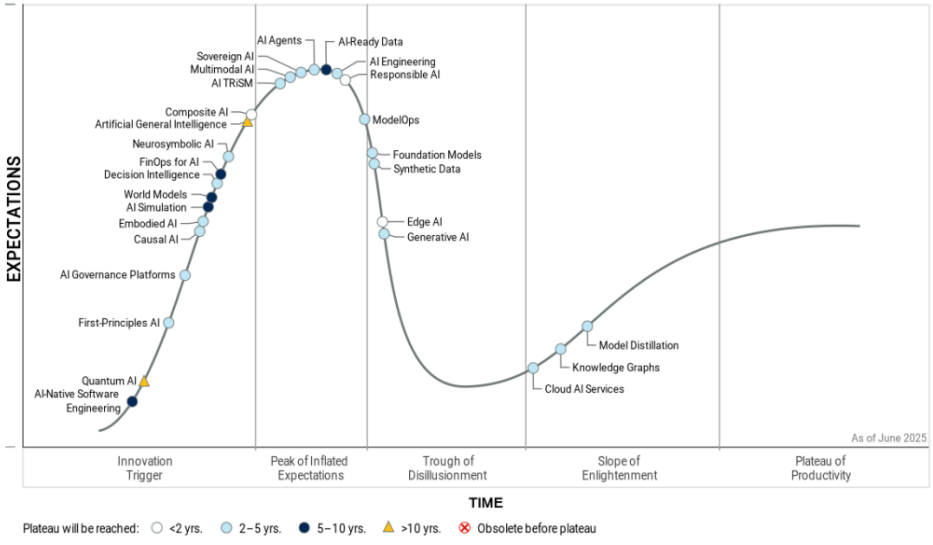
Knowledge Graph Specialist | Data Treehouse

DATA TREEHOUSE

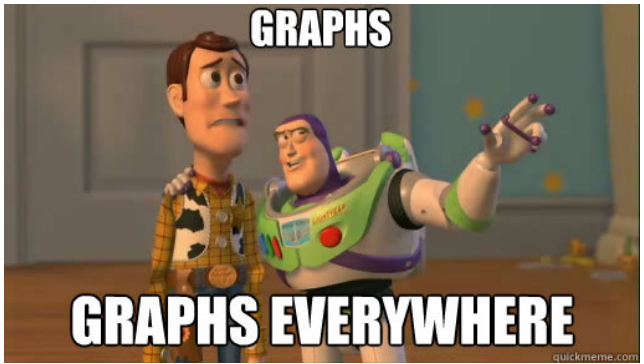
veronika@data-treehouse.com

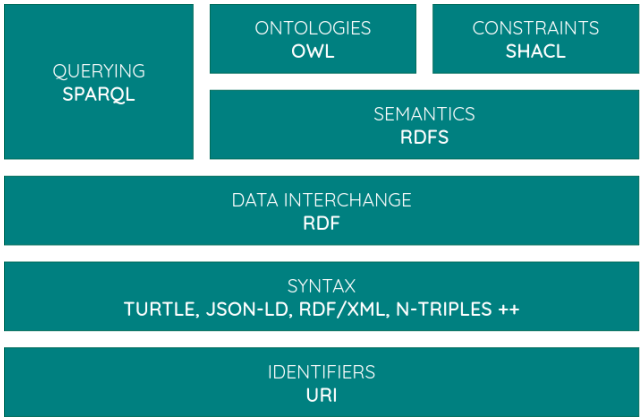
veleda
vheimsbakk
veronahe.no

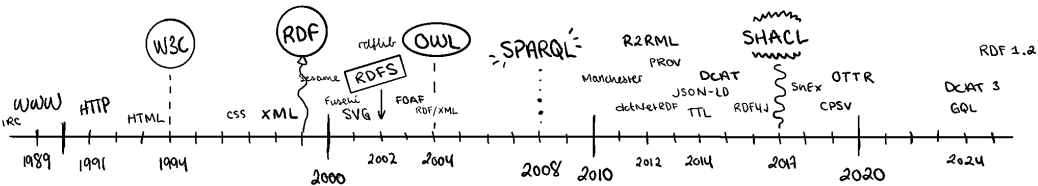
Hype Cycle for Artificial Intelligence, 2025



Gartner


GRAPHS
GRAPHS EVERYWHERE
quickmeme.com


QUERYING
SPARQL
ONTOLOGIES
OWL
CONSTRAINTS
SHACL
SEMANTICS
RDFS
DATA INTERCHANGE
RDF
SYNTAX
TURTLE, JSON-LD, RDF/XML, N-TRIPLES ++
IDENTIFIERS
URI


W3C
RDF
OWL
SPARQL
R2RML
SHACL
RDF 1.2
WWW
IRC
HTTP
HTML
CSS
XML
Sesame
RDFS
Fusion
SVG
FOAF
RDF/XML
redfoot
Manchester
PROV
DCAT
JSON-LD
TTL
RDF4J
SHEX
OTTR
CPSV
DCAT 3
GQL
1989
1991
1994
2000
2002
2004
2008
2010
2012
2014
2017
2020
2024

Think of data as a **directed graph**, and that all ***things*** has a relation to other things.

Terminology

Think of data as a directed graph, and that all things has a relation to other things.

- An open standard for representing data as graphs.

- Data described as **triples**.
- A triple is also called a **fact** or a **statement**.
- The elements of a triple are also called **resources**.

```
subject predicate object
```

- Use **Uniform Resource Identifiers** (URI) as global, unique identifiers.

URI

- **Only** a **name**. Does not need to link to anything, URI not URL.

```
scheme:[//[user:password@]host[:port]][/]path[?query][#fragment]
```

Example
`http://data.eksempel.no/Kraftverk`

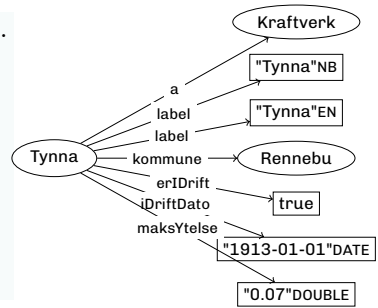
URI	
Namespace	**Resource name**
`http://data.eksempel.no/`	`Kraftverk`

```
http://data.eksempel.no/Tynna
  http://www.w3.org/1999/02/22-rdf-syntax-ns#type http://data.eksempel.no/Kraftverk .
```

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix : <http://data.eksempel.no/> .

:Tynna rdf:type :Kraftverk .
```

Literals and URIs

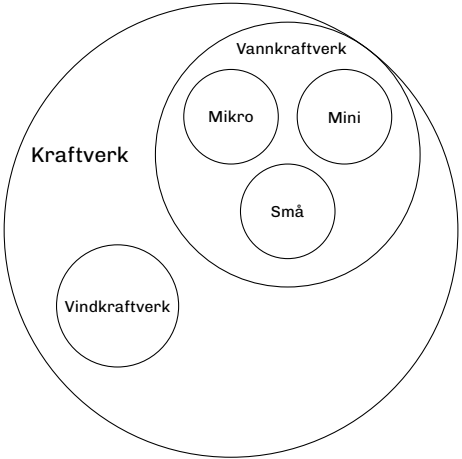
```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix : <http://data.eksempel.no/> .

:Tynna a :Kraftverk ;
  rdfs:label "Tynna"@nb, "Tynna"@en ;
  :kommune :Rennebu ;
  :erIDrift true ;
  :iDriftDato "1913-01-01"^^xsd:date ;
  :maksYtelse "0.07"^^xsd:double .
```



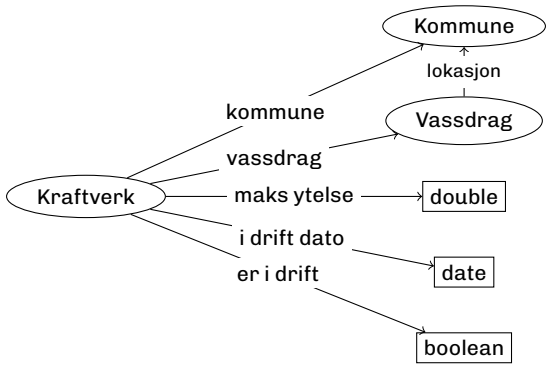
Property semantics

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix : <http://data.eksempel.no/> .

:kommune a rdf:Property ;
  rdfs:label "kommune"@nb, "municipality"@en ;
  rdfs:domain :Kraftverk ;
  rdfs:range :Kommune .
```

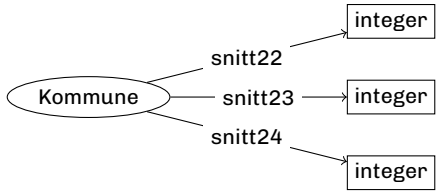
```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix : <http://data.eksempel.no/> .

:Kraftverk a rdfs:Class ;
  skos:prefLabel "Kraftverk"@nb, "Power station"@en ;
  skos:altLabel "Power plant"@en .

:Vannkraftverk rdfs:subClassOf :Kraftverk ;
  skos:prefLabel "Vannkraftverk"@nb, "Vasskraftverk"@nn, "Hydroelectric power station"@en .

:Mikrovannkraftverk rdfs:subClassOf :Vannkraftverk .

:Vindkraftverk rdfs:subClassOf :Kraftverk .
```


Kraftverk
Vannkraftverk
Mikro
Mini
Små
Vindkraftverk

DATA



NVE Vannkraftverk `https://www.nve.no/energi/energisystem/vannkraft/vannkraftdatabase/`

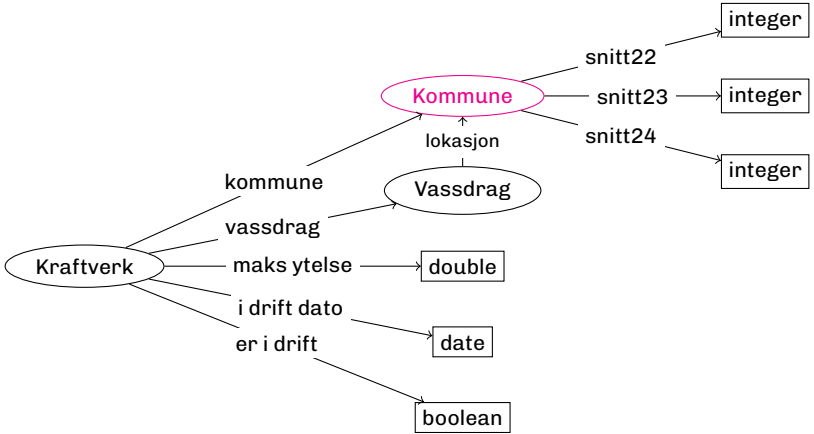


SSB Lønn `https://www.ssb.no/statbank/table/12852/`

ONTOLOGIES

Ontology

VocabularyA collection of words, concepts, terms (and/or RDF resources). TaxonomyA classification of the vocabulary. Ontology A set of concepts and categories that shows their properties and relations between them. Including accessible semantics and logic.
The software I use: Protégé — `https://protege.stanford.edu/`


integer
snitt22
Kommune
snitt23
integer
lokasjon
snitt24
integer
kommune
Vassdrag
vassdrag
Kraftverk
maks ytelse
double
i drift dato
date
er i drift
boolean

MAPPING

Resonable Ontology Templates (OTTR)

- Mapping language for RDF
- Open source
- Developed by academia in Norway, used in industry

`https://ottr.xyz/`

`https://github.com/veleda/ottr-masterclass`

SQL		SPARQL
SELECT * FROM A INNER JOIN B ON A.KEY = B.KEY	$A \cap B$ 	SELECT * WHERE A B
SELECT * FROM A LEFT JOIN B ON A.KEY = B.KEY	A 	SELECT * WHERE A OPTIONAL {B}
SELECT * FROM A LEFT JOIN B ON A.KEY = B.KEY WHERE B.KEY IS NULL	$A \setminus B$ 	SELECT * WHERE A FILTER NOT EXSISTS {B}
SELECT * FROM A OUTER JOIN B ON A.KEY = B.KEY	$A \cup B$ 	SELECT * WHERE {A} UNION {B}