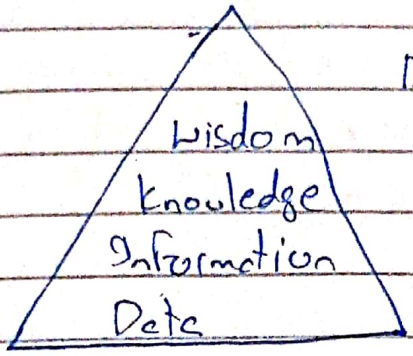
Knowledge graphs



DIKW pyramid

Not all structured data is information, as it might not have context to it.

semantics → meaning)

Ontology → formal representation of knowledge

dbpedia

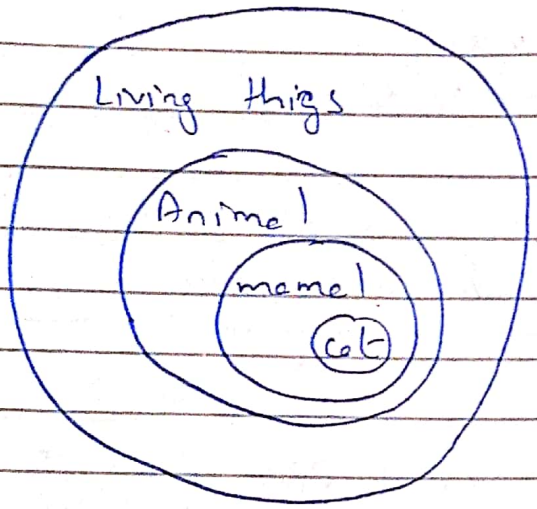
Scanned with CamScanner

p 30 am 2e

Intro to knowledge graph

Subsumption Hierarchy

→ Class x fully subsumes class y



Ontology is like a DB scheme. But it has rules indebted in it.
It can replace Data base.

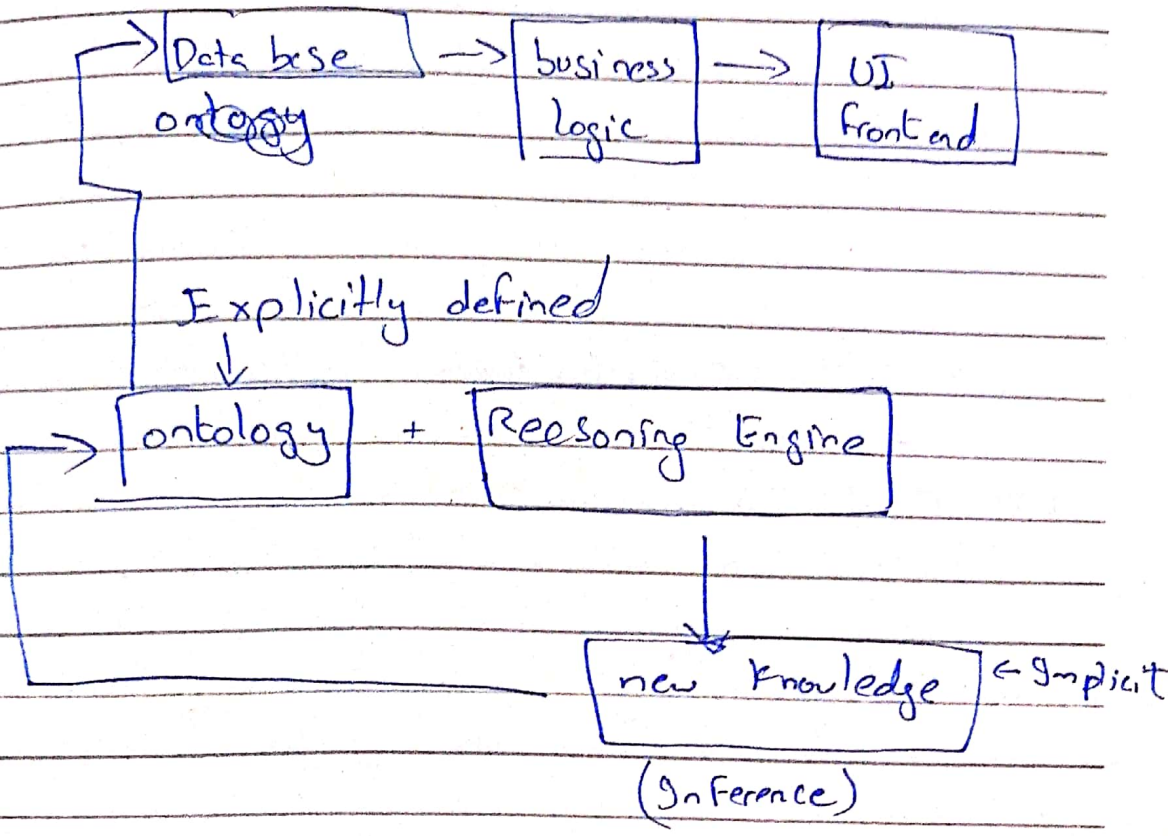
ontology + reasoning engine

↓

new knowledge (Inferences)

Scanned with CamScanner

DBpedia



This above creates a Finite loop of data enrichment, It is a conceptual diagram of a knowledge driven application.

H.W represent Moby in 3 different knowledge formal

Scanned with CamScanner

Correct interpretation depends on:-

- syntax
- Semantics
- Context → (Usually google checks location)
- Pragmatic → tone (sarcastic, angry, happy)
- Experience

Keys For speaking a Common language:

- Syntax
- Semantics
- Texonomy - Classification of concepts
- Thesauri - Relations b/w concepts
- Ontologies - rules & knowledge about which relations are allowed to make sense

Scanned with CamScanner

Intro to Knowledge graphs

for the machine

formal knowledge representation is a field of artificial intelligence (AI)

Semantic web -> web of data

Read article semantic web

Scanned with CamScanner

Information graph

universal resource link

- Web 1.0 → Web of documents (static)
- Web 2.0 → Web of Applications (Applications talk to each other API)
- Web 3.0 → Has a lot of things (for a generation of web technology)
  - ↳ Semantic web
  - ↳ Linked data
  - ↳ block-chain
  - ↳ decentralization

Application level interoperability

Linked data → the data on pages is understood & by understanding it connects it

3 - benefits of semantic web

- ↳ Gives direct ans instead of document to a question
- ↳ Semantics based "Inter-operability"
- ↳ As new knowledge is understood, so we now have new knowledge

Explicit → Reasoning → Implicit

- What is Linked data cloud ?
  - ↳ universal knowledge
  - ↳ universal format

- DBPedia is the nucleus of linked data cloud, it is the graph version of wikipedia. It is structured. It is in RDF (resource distribution framework)

* Linked data is the real life implementation of the idea, semantic web

* metadata → data about the data, data defining the data.

* The limitation of traditional web is that there is no explicit semantics in the traditional web.

Scanned with CamScanner

Intro to Knowledge Data

URI → universal resource identifier

↓

Not restricted to documents, it can point to a resource/knowledge inside a document. This is very helpful in linked data.

More expressive → more time → less computable by rules

* All url are uri, but all uri are not url.
* url are locatable
* But not all uri are locatable
* Some uri are expressed as url

Scanned with CamScanner

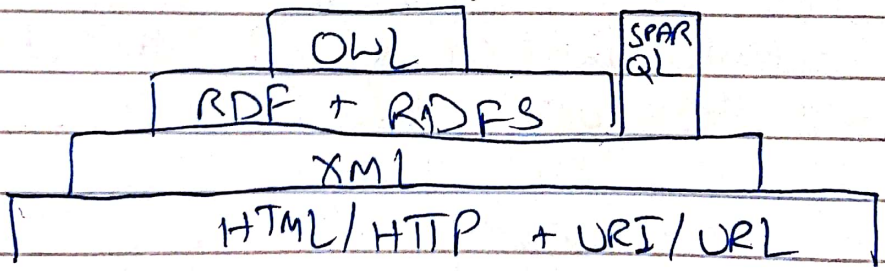
next class quiz

Knowledge Graph

① Why is xml not best suited for semantics based meta data representations?
or
What are the limitations of xml?

Ans: There are multiple ways to represent the meanings of the same thing, this causes confusion for the machine, It is too open ended & scheme can differ from one programmer to another.

② Semantic web technology stack



Scanned with CamScanner

③ Difference b/w URL & URI ?

URL ⊆ URI

universal resource identifier → URI
universal resource location → URL
↓
locatable means that if you put on browser we get useful info

Data heirarcy

1 star → Just data on web
2 star → 1 star + structured (eg csv etc)
3 star → 2 star + specific furmet
4 star → 3rd star + RDF
5 star → 4 star + connected to others.

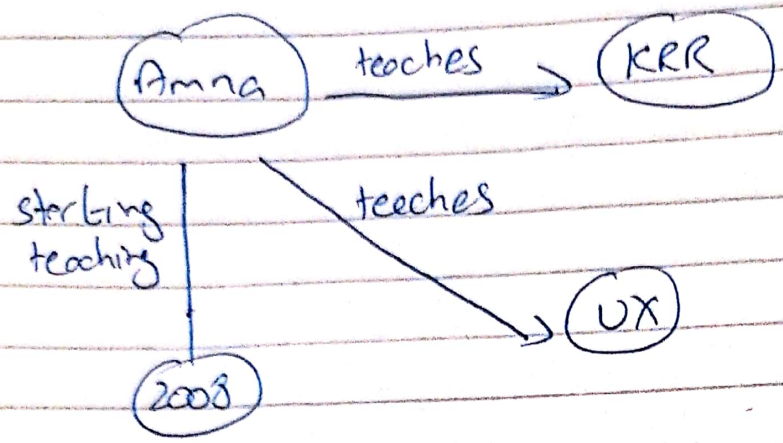
* All RDF data is in form of- triplets

S P O

S: always a resource (have URI)
P: always a resource (have URI)
O: not necessarily a resource can be a litera (URI not necessary)

Scanned with CamScanner

Convert to to RDF



a Base URI/URL is decided

<Base URL/Amna> <Base URL/teeches> <Base URL/KRR>

<Base URL/Amna> <Base URL/teaches> <Base URL/UX>

<Base URL/Amna> <Base URL/starting teaching> <"2008">

If a predicate pre-exist, we use it, otherwise use ours (vocabulary should be reuseable & reused)

Scanned with CamScanner

child-parent relation in predicate

* Not in pure RDF, but RDF-scheme does this thing

resource → oval

literal → rectangle

"symantic" ^^ xsd:string } This how a literal will be specified its data type

The literal will always be in ""

abc was born on 1st August 2001

base url: dbc — dbpedia:birthday

dbpedia: birthday

"25/08/2001"^^xsd:date

Knowledge graph

turtle notation

```
:The_Beatles rdf:Type :Band;
             :name :"Beatles";
             :member :John,
                     :Paul,
                     :Ringo
```

rdf:type => a

This becomes

PREFIX prefix_name : <URL>

Keyword — prefix name — The URL

So, prefix_name : some_thing

This means that URL is behind the some_thing

:something means that we have a prefix without any name

Scanned with CamScanner

c PREFIX that should be defined before

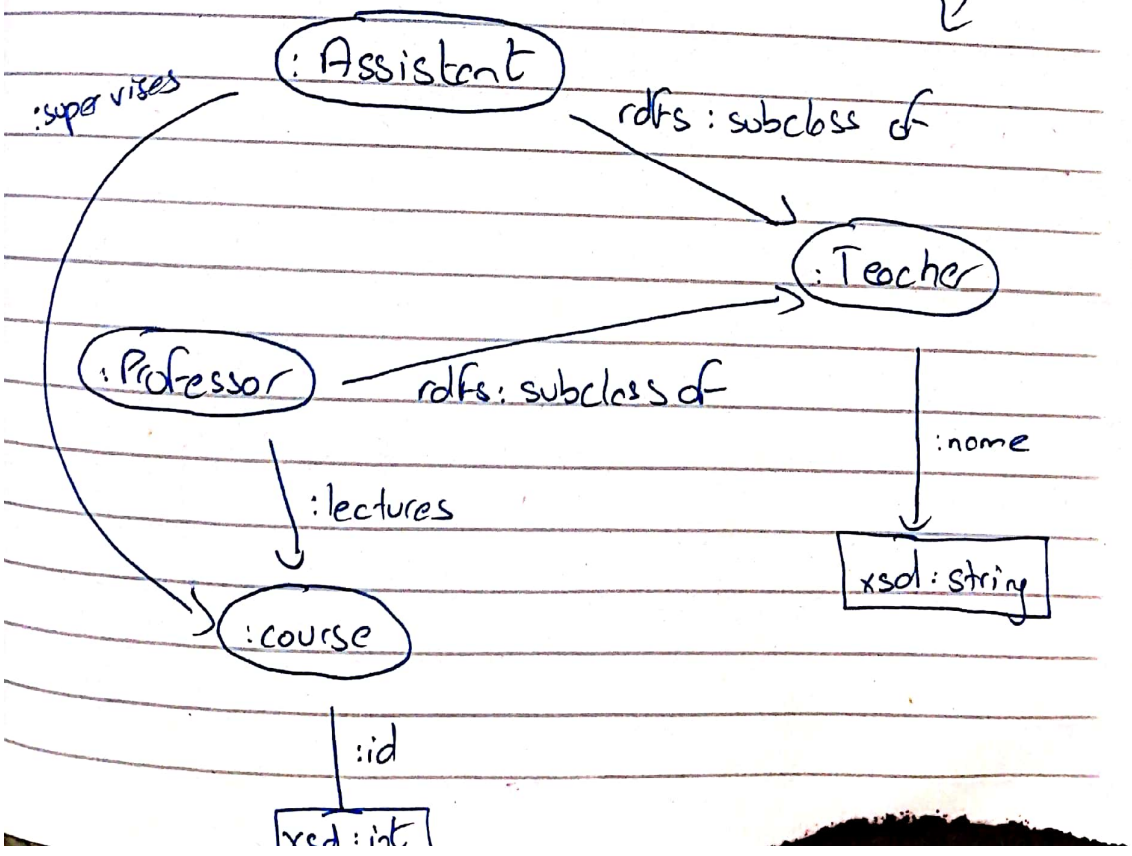
sp : Shakespeare — db: wrote — sp : Macbeth . (full stop imp)

subject — predicate — object

In n3 notation, you have to write in <> & write full uri.

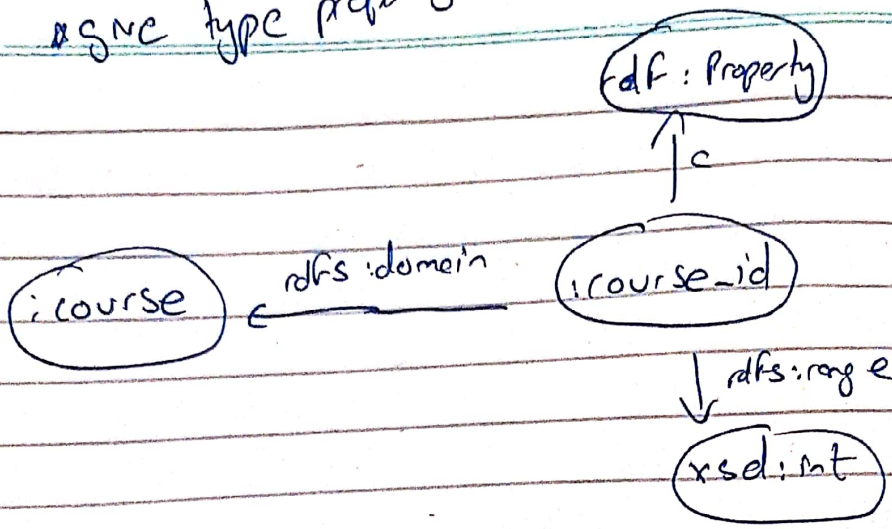
rdf & xsd are standard prefix for w3 vocabulary

When representing in structure graph we use single not plural, scheme of QS



Scanned with CamScanner

* Give type property & define domain & range



rdfs → allows design of schema

schema are called T-box
↑ termnology

when schema & data are connected we call it a-box
↳ assertion

Scanned with CamScanner

KNOWLEDGE GRAPH

The task of Text Classification

N-triples is the simplest.

Turtle is built on N-triples

@base is a keyword, it is for prefix of the main URI

So, <Pluto> means the base URI + pluto

rdf:ID = "something"

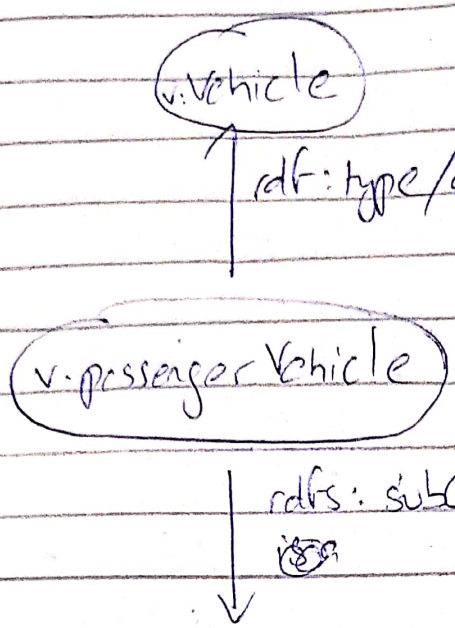
We are creating a new ID/resource "something", it will be appended on the base URI

rdf:resource: "something1"

We are referring to an already present resource "something1" & it will append on the base URI

Scanned with CamScanner

rdf:type => a



a <= rdf:type } b/w instances

rdfs:subClassOf } b/w classes

rdf -> more data related
rdfs -> more class/scheme related

* We use rdf & rdfs for reusability
They are standard vocabulary

Scanned with CamScanner

KNOWLEDGE GRAPH

rdf- vs rdfs

← cannot be used without rdf-

rdfs : → Can make texonomy with it (class, sub-class, property)
↳ dealing more with schema

rdf- : → cannot make texonomy with it
↳ Dealing more with instances
↳ we make statements with it (triples)

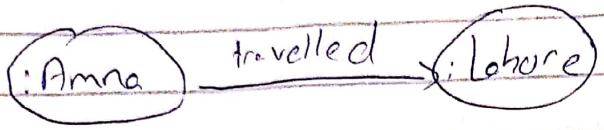
rdfs provides:

- Annotations
- Class / sub Classes
- Domain / Range
- SubProperty

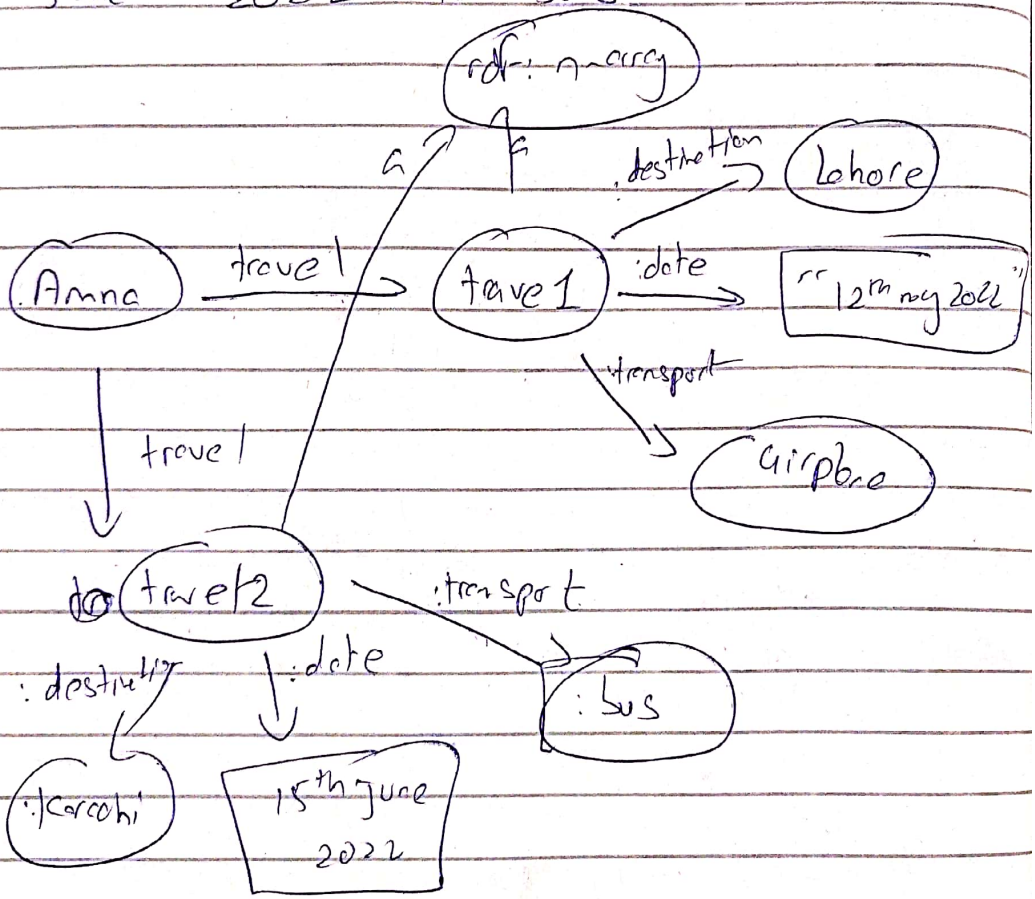
* We cannot say 2 classes are disjoint in rdfs & rdf-, This we can do in owl vocabulary

n triples ⊂ turtle ⊆ n3

Scanned with CamScanner



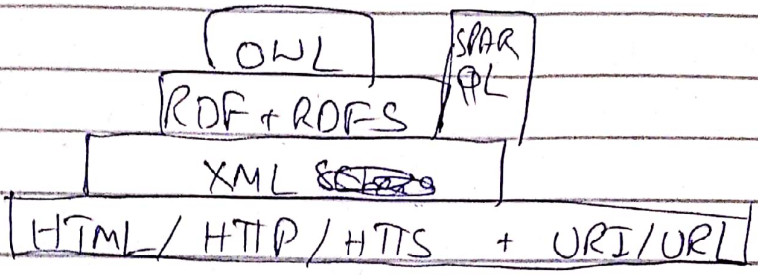
Amna Trvelled to Lahore on 12th may 2022 by airplore She travelled to karachi on 15th June 2022 vic bus.

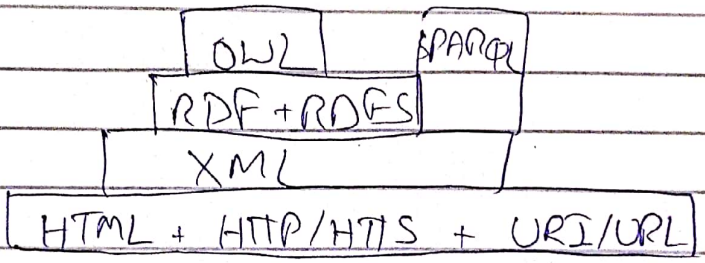


Scanned with CamScanner

KR Practice

Semantic Web technology stack





To process meaning of information
To relate & integrate heterogenous data
To deduce implicit information from existing information

Scanned with CamScanner

KBR Practice

(1) To process the meaning of information automatically

(2) To relate & integrate hetrogenous data

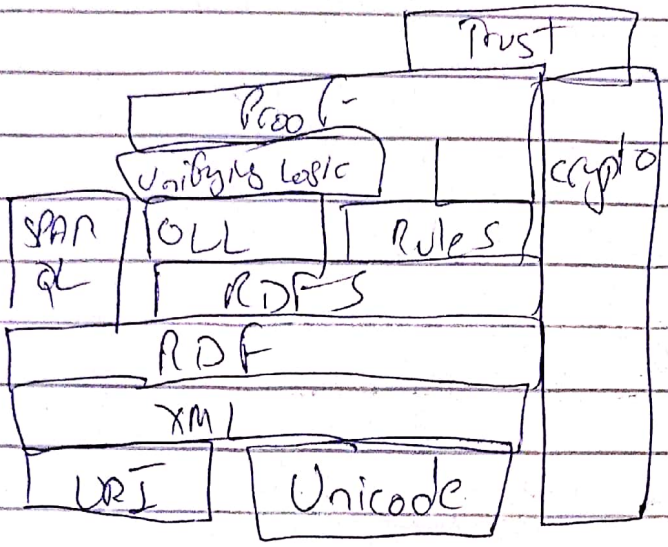
(3) To deduce & new info

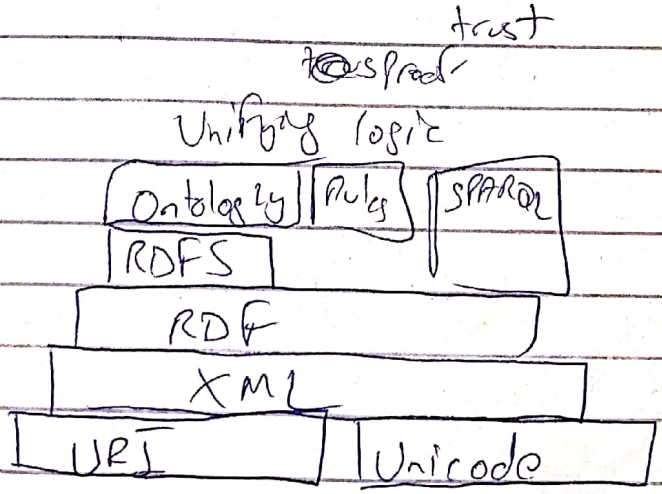
relating & integrating hetrogenous data

process meaning of information automatically

Scanned with CamScanner

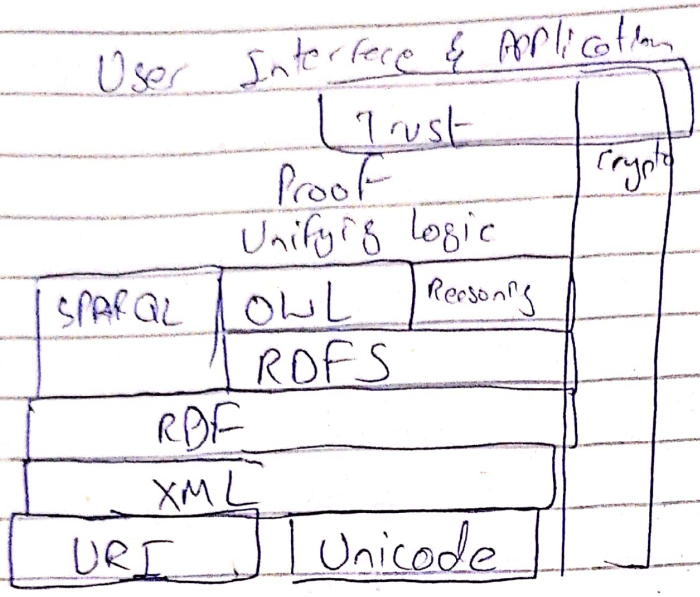
ICRR Practice

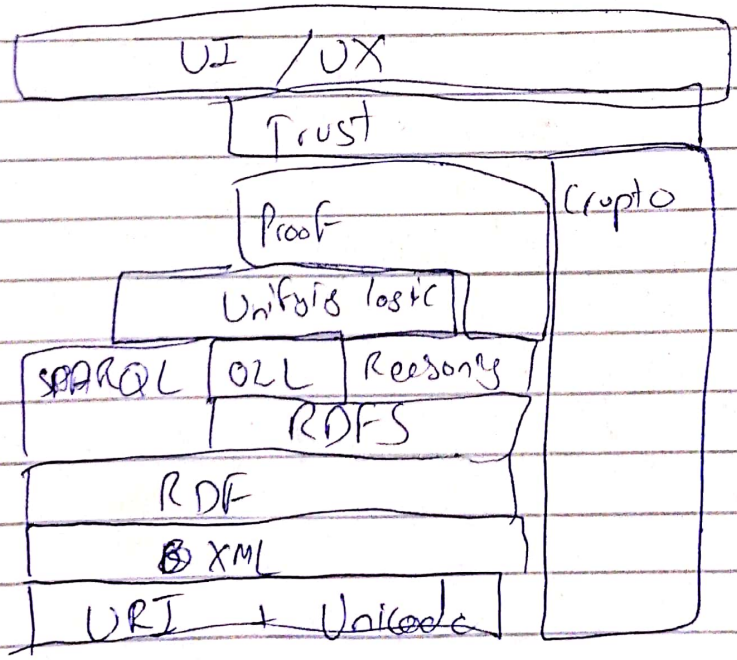




Scanned with CamScanner

KPR Practice





Scanned with CamScanner

KRR Practice

① meaning of info process automatically

② ~~listing~~ connecting & integrating heterogeneous data

③ Implicit knowledge from

②

* 3 advantages of Semantic Web
* Semantic Web technology stack
* Understanding language
* Speaking language
* URI vs URL
* Linked data cloud
* Semantic meta data
* Why XML not best suited
* Data heirarchy (Sister etc)
* RDF vs RDFS
* difference b/w web 1.0, web 2.0 & web 3.0

Scanned with CamScanner

ICFR Practice

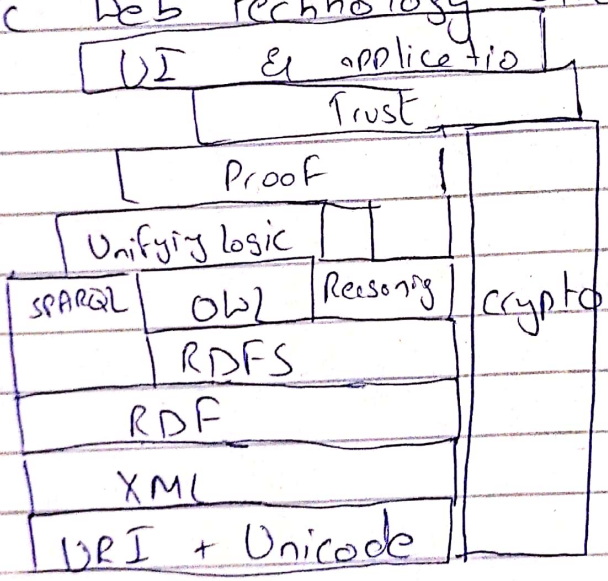
* 3 advantages of Semantic Web

① The meaning of information can processed automatically

② To relate & integrate heterogenous data

③ To derive implicit (new) knowledge from already existing explicit (old) knowledge

* Semantic Web technology Stack



Scanned with CamScanner

KRR Practice

* Understanding / Interpretation of Language

1. Syntax
2. Semantics
3. Context
4. Pragmatics
5. Experience

* Keys for speaking a common language

1. Syntax
2. Semantics
3. Taxonomy
4. Thesaurus
5. Ontology

Scanned with CamScanner

IRPR Practice

URI vs URL

① URI → universal resource identifyer
② URL → univeral resource locator

URL is subset of URI
URL ⊆ URI

All URL are URI, but not all URI are URL

Locatable
URL is locatable
↳ It means it returns something (document) when searched on a web engine

Linked Data Cloud

↳ Universal knowledge (open knowledge (public))
↳ Universal format

Linked data → The data is understood & connected to other similar things.

Scanned with CamScanner

KRR Practice

* Semantic Metadata

Metadata → Data about Data,
Data explaining data.

Semantic metadata → Data explaining the semantics / meaning of data

* Why XML is not best suited?

① There are multiple ways to represent the meaning of the same thing, this causes confusion for the machine.
② It is too open ended & scheme will differe from one programmer to another.

* Data heirarchy (Linked open data)

1-star → available on the web (what ever format) but with an open licence
2-star → 1 star + available as machine readable structured data (execel insted of scaning)
3-star → 2 star + non-proprietary format (csv instead of excel)
4 star → 3 star + use open standard from W3C (RDF + SPARQL) to identify things, so that people can point at your stuff

Scanned with CamScanner

KRR Practice

5 star → 4 star + link your data with other people's data to provide context.

RDF vs RDFS

↳ RDF is used to make statements about resources in form of subject, predicate, object triples

RDF is in general a method of conceptual data modelling

RDFS is used to describle the schema/structure that RDF can use.

Web 1.0, web 2.0, web 3.0

Web 1.0 → Linking b/w documents by hyper links

Web 2.0 → Interoperability b/w web applications

Web 3.0 → Semantic web, Linked data
Data/knowledge/information level interoperability

Scanned with CamScanner

KR Practice

T-Box vs A-Box

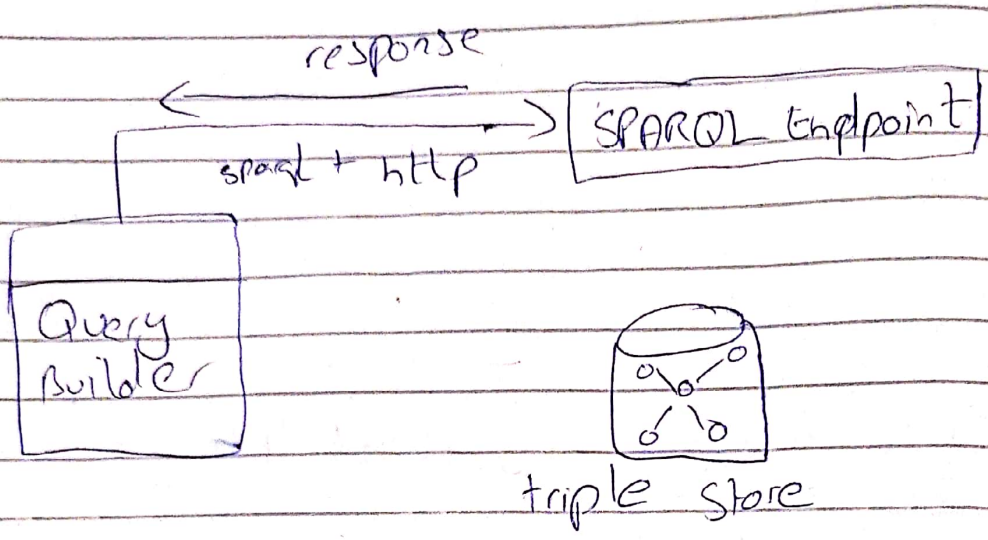
T-Box → Terminology Box

A-Box → Assertion Box

- The T-Box contains the axioms defining the classes & relations in an ontology
- The A-Box contains the assertion about the individuals in the domain

Scanned with CamScanner

KNOWLEDGE GRAPHS



Data visualization

One query can access different end-points (can gather data from different end-points)

Wikidata also has a strong query engine.

Scanned with CamScanner

FILTER REGEX(?title, "love", "i")

↑
Full text search

OPTIONAL → By using this it will not limit our results. If the optional thing is not available, it will show those as blank

FILTER → Filter date, more date related

HAVING → Filter the query, like more meta date related.

↓
Always used in combination with GROUP BY

Scanned with CamScanner

KRR

رَبَّنَا أَفْرِغْ عَلَيْنَا صَبْرًا

عَلَيْنَا → upon us
أَفْرِغْ → pour
رَبَّنَا → O our lord

WHERE { ?s p:knows+/p:known ?o }

This is like a loop
(any length more then 1)

WHERE { ?s p:knows/p:knows ?o }

Only set to
2 length.

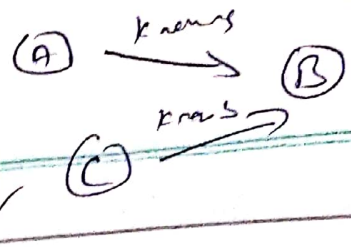
WHERE { ?s p:knows+/p:name ?o }

This + is with
knows, so, it will
go it it finds
knows then give
name

WHERE { ?s p:knows+/ ?o }

Any length equal or more then 1

Scanned with CamScanner

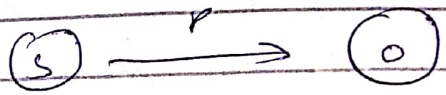


B, T... reverses. Let say it checks A from A -> B -> A, C & C -> B -> A, C

```
WHERE
{
    ?s  p:knows / ^p:knows  ?o.
}
FILTER (?s != ?o)
```

Which people do know the same people?

'^' is for reversal. Like it will give, suppose



This is a directional graph. We want to read it in opposite. We use ^

```
?any1  ^p  ?any2
```

This will give o in subject & s in predicate

Scanned with CamScanner

* Some times the same info is represented with different predicates in the same graph. The Alternative graph path query helps us in it.

rdfs:label | foaf:name

If it finds one or both, it will return them

It is like the union of these 2 predicates. This OR is more like UNION.

Scanned with CamScanner

KRR

SPARQL supports data virtualization?

TRUE, Same query can query multiple end points.

Scanned with CamScanner

Quiz next class

KNOWLEDGE GRAPH

tutorial point of view

T-box A-box

- Classes / indivisuals | HW

- properties < Obj property | EL, RL, QL } OWL suffix, find difference
  Data property

- Restrictions (Axioms)

pellet, pellet incremental

* If a property is functional, then it can have only 1 value.
* If you have more then 1, then some reasoners may give error, some may say that they are the same thing.

Heirarchy → class hierarchy

Transitivity → a > b > c, a > c

↳ we can declare a property as such

Scanned with CamScanner

~~INFORMATION~~

KNOWLEDGE GRAPH

Primitive class

- Only has necessary conditions
- If reasoner is started, it will not classify unknown class, objs as this primitive class, even if they fullfill the necessary conditions
  - → But all subclasses & entities of these subclasses + entities will fullfill these conditions

Defined class

- We can classify primitive class as defined class
- Same as point 3 of primitive class
- It has necessary & sufficient conditions
- If an unknown thing satisfies the ~~satisfies~~ sufficient conditions of defined class, the reasoner will classify that class as sub-class of this class or this class as that class subclass

---

Obj type property | Date type property

If subject is a class/resource it is obj type property

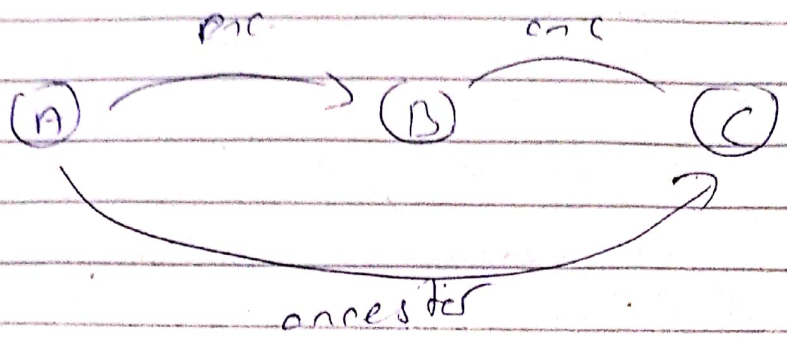
If subject is a string literal it is a datetype property

*In owl we must specify

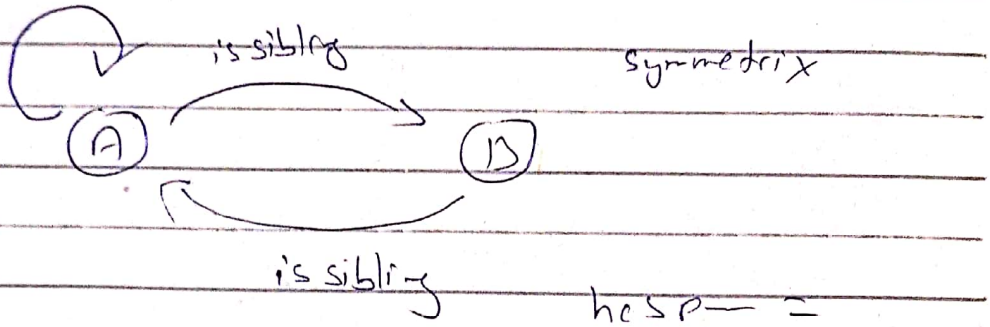
Scanned with CamScanner

SnomedCT snomed.org

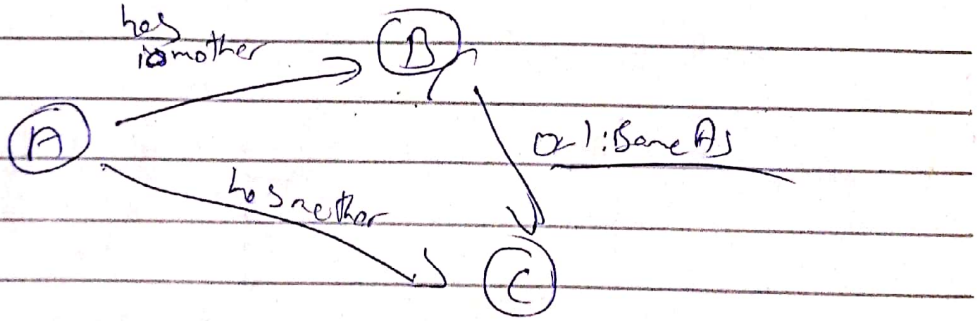
transitive



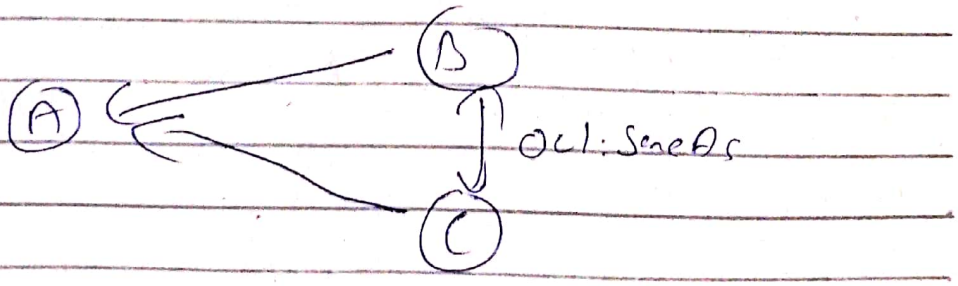
reflexive is human



function



inverse



Scanned with CamScanner

knowledge graph

Tool talks (5 obs)
Project

Tool talks → video presentation
↳ topic selection

project
↳M1
↳M2
↳M3
↳M4

Reflexive property ↓ A type of property

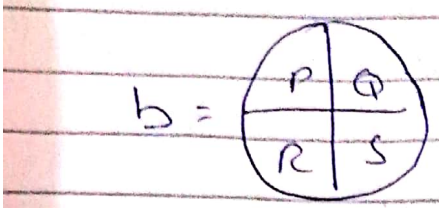
vs

reflexive property restriction ↓ A restriction on a property

Scanned with CamScanner

↑ class level

covering axiom: A class which has a covering axiom is fully covered by the union of its subclasses



$P \subseteq b$	$P \cap Q = \phi$	$b = P \cup Q \cup R \cup S$
$Q \subseteq b$	$Q \cap R = \phi$	
$R \subseteq b$	$R \cap S = \phi$	b is equivalent
$S \subseteq b$	$P \cap R = \phi$	to the union
	$P \cap S = \phi$	of its subclasses
	$Q \cap S = \phi$	

① Create subclasses
② Make em disjoint
③ make the union of the subclass equivalent to the super-class.

part of covering axiom, subclasses are mutually disjoint

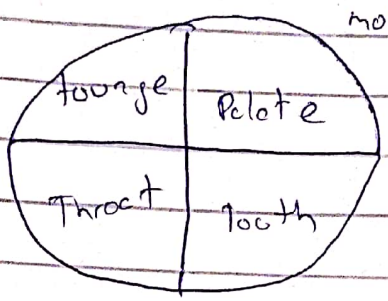
Closure axiom is property level

Quiz next tuesday

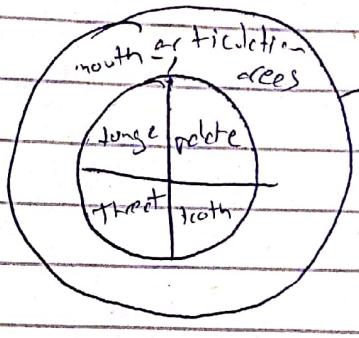
KRR (previous exam question)

past

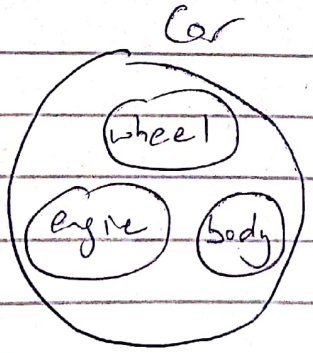
① A mouth consists of only 4 primary Articulation areas
tounge, palate, throat, teeth



← It means a tounge is a mouth, so, this is wrong, both are wrong



This is more prominent here



* The difference b/w knowledge base & data base is that we try to keep things logically & semantically correct.

Scanned with CamScanner

Ontology Design

Methodology

formal -> machine understandable
↓
meaning must be explicitly defined

* We have to document assumption so that if someone else is using our ontology should know our assumption as our assumption may differ from theirs'.

* If we have more than 1 way to model an ontology, we chose one of them & provide our rationale/ reason for selecting that one.

Scanned with CamScanner

protege → Graph design tool

→ Ontology development 101 (OD 101)

reuse

↳ Is there an ontology already out there that might cover some part or may be useful in our ontology

↳ Our ontology is reuseable

Methontology

↳ Domain competence questions (imp)

Diagramtic T-BOX (conceptualization)

OWL ready, Open link virtuoso

Scanned with CamScanner

Knowledge graph

Linked Data Engineering & Application

Linked Data

↳ Use URI as name of things

Select Domain
↳ Create Ontology (competency questions)
↳ Convert dataset to rdf in this Ontology.

Anything in triples is in RDF.

but in case of ontologies we must have a defined meaning.

Light weight vs heavy weight ontologies

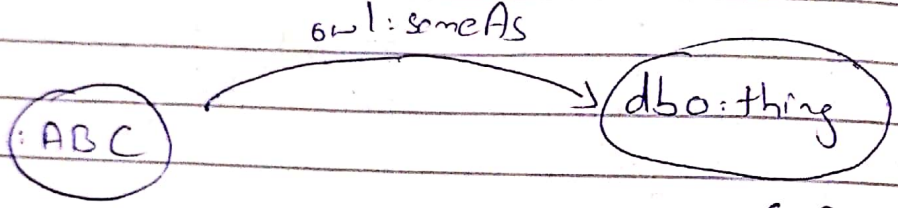
- Light weight ontologies have less expressivity, they donot have extensive logic
- Heavy weight ontologies have more expressivity, they have extensive logic (restrictions, properties etc)

Scanned with CamScanner

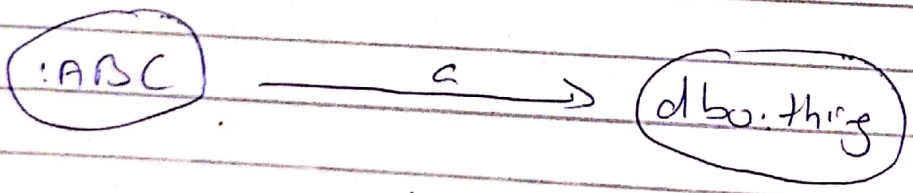
triple stores

~~In RDF file convert to~~

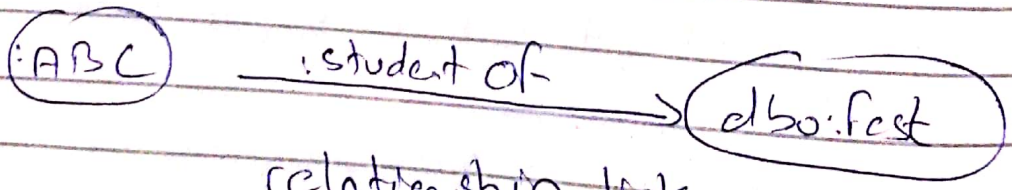
If we convert SQL database to RDF, most of the time data is kept in the database. This helps in virtualization, where we can now query multiple SQL database then we convert them to RDF with a single query.



Identity Link (A - box level)



Vocabulary Links (Abox to T-box level)



relationship link

Scanned with CamScanner