



# Szemantikus adatbázisok

Nagypál Gábor  
nagypal@db.bme.hu



# Motiváció

# Ismétlés: Adat, Információ, Tudás

- Adat: a valóság értelmezhető, de nem értelmezett része
  - 42, “Jakab”
- Információ: értelmezett adat
  - 42 - cipőméret, “Jakab” - személynév
- Tudás: kontextusba helyezett információ(k)
  - Jakab cipőmérete 42-es

# Mérnöki feladatok

1. Hogyan tudjuk a tudás kinyerését minél inkább automatizálni?

- o NLP, data mining, text mining, ...

**1. Hogyan tudjuk a tudást reprezentálni és automatikusan felhasználni?**

# Tudásreprezentáció stratégiák

Szabványos címkék, mint pl. a Dublin Core, schema.org

- mindenki (gépek, emberek) tudja, mit jelentenek a címkék (title, author, ...)
- Sajnos ez így rugalmatlan, nehezen bővíthető

Használjunk **ontológiákat**, hogy a címkék jelentését leírjuk

- Az ontológiák definiálnak egy címkeszótárat
  - pl. ember, énekes
- Új címkék létrehozhatók a meglévők kombinációjával
  - Pl. kétgyermekes anya, platinalemezes énekes
- A kombinált címkék értelmezése formálisan specifikált
- **Új név: knowledge graph, “tudásgráfok”**

Figure 1. Hype Cycle for Emerging Technologies, 2018



© 2018 Gartner, Inc.

Source: Gartner (August 2018)

# Példa: Google Knowledge Graph



Leonardo da Vinci

Polihisztor

Leonardo di ser Piero da Vinci itáliai polihisztor. Leonardo festő, tudós, matematikus, hadmérnök, feltaláló, anatómus, szobrász, építész, zeneszerző, költő és író volt egy személyben. [Wikipédia](#)

**Születési dátum:** 1452. április 15., Anchiano, Olaszország

**Meghalt:** 1519. május 2., Clos Lucé, Amboise, Franciaország

**Teljes név:** Leonardo di ser Piero da Vinci

**Testvérek:** Bartolomeo da Vinci, Giovanni Ser Piero, [TOVÁBBIK](#)

**Szülők:** Caterina da Vinci, Piero da Vinci

Műalkotások

Még 15+



Mona Lisa  
1503.



Az utolsó vacsora  
1498.



Angyali üdvözlés  
1472.



Keresztelő Szent János  
1513.



Leonardo da Vinci

Polymath

Leonardo di ser Piero da Vinci, more commonly Leonardo da Vinci or simply Leonardo, was an Italian Renaissance polymath whose areas of interest included invention, painting, sculpting, architecture, ... [Wikipedia](#)

**Born:** April 15, 1452, Anchiano, Italy

**Died:** May 2, 1519, Clos Lucé, Amboise, France

**On view:** The Louvre, National Gallery of Art East Building, [MORE](#)

**Periods:** High Renaissance, Early renaissance, Renaissance, Italian Renaissance, Florentine painting

**Siblings:** Bartolomeo da Vinci, Giovanni Ser Piero, [MORE](#)

**Parents:** Caterina da Vinci, Piero Fruosino di Antonio da Vinci

# Példa: Wolfram Alpha

The screenshot shows the Wolfram Alpha interface. At the top, the search bar contains the word "golf". Below the search bar, a red oval highlights a disambiguation message: "Assuming 'golf' is a physical activity | Use as a financial entity or a word or a golf topic instead". Below this, the "Input interpretation:" section shows "playing golf (activity)". The "Metabolic properties:" section is expanded, showing a table of values. A "Show non-metric" link is visible to the right of the table. At the bottom, it says "(estimates based on CDC and ACSM recommendations)". The footer includes "Computed by: Wolfram Mathematica", "Source information »", and "Download as: PDF | Live Mathematica".

WolframAlpha™ computational... knowledge engine

golf

Assuming "golf" is a physical activity | Use as a financial entity or a word or a golf topic instead

Input interpretation:  
playing golf (activity)

Metabolic properties: [Show non-metric](#)

energy expenditure rate	20 kJ/(kg hr) (kilojoules per kilogram hour)
oxygen consumption rate	16 mL/(kg min) (milliliters per kilogram minute)
fat burning rate	0.61 g/(kg hr) (grams per kilogram hour)
metabolic equivalent	4.5 metabolic equivalents

(estimates based on CDC and ACSM recommendations)

Computed by: [Wolfram Mathematica](#) [Source information »](#) Download as: [PDF](#) | [Live Mathematica](#)





# Példa: DBpedia

## Virtuoso SPARQL Query Editor

Default Data Set Name (Graph IRI)

<http://dbpedia.org>

Query Text

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbp: <http://dbpedia.org/resource/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
```

```
SELECT ?name ?bandname where {
  ?person foaf:name ?name .
  ?band dbo:bandMember ?person .
  ?band dbo:genre dbp:Punk_rock .
  ?band foaf:name ?bandname .
}
```

limit 100

(Security restrictions of this server do not allow you to retrieve remote data)

Results Format:

Execution timeout:  millis

Options:

- ☒ Strict checking of void variable
- ☐ Log debug info at the end of o
- ☐ Generate SPARQL compilation

(The result can only be sent back to browser, not saved on the server)

Run Query

Reset

name	bandname
"Lora Logic"@en	"X-Ray Spex"@en
"Steve Diggle"@en	"Buzzcocks"@en
"Tré Cool"@en	"Green Day"@en
"Robert Grey"@en	"Wire"@en
"Robert Gotohed"@en	"Wire"@en
"Billy Zoom"@en	"X"@en
"Erik Sandin"@en	"NOFX"@en
"Colin Newman"@en	"Wire"@en
"Jim DeRogatis"@en	"Vortis"@en
"Pinch"@en	"The Damned"@en
"Baz Warne"@en	"The Stranglers"@en
"Jeff Dean"@en	"The Bomb"@en
"Pete Mittler"@en	"The Bomb"@en
"Jeff Pezzati"@en	"The Bomb"@en
"Captain Sensible"@en	"The Damned"@en
"Billie Joe Armstrong"@en	"Green Day"@en
"Fat Mike"@en	"NOFX"@en
"Baz Warne"@en	"The Stranglers"@en

# Példa: Wikidata

Wikidata Query Service

Examples Help More tools English

Query Helper

+ Filter instance of house cat

+ Show image

Limit

```

1 #Cats, with pictures
2 #added before 2016-10
3
4 #defaultView:ImageGrid
5 SELECT ?item ?itemLabel ?pic
6 WHERE
7 {
8   ?item wdt:P31 wd:Q146 .
9   ?item wdt:P18 ?pic
10  SERVICE wikibase:label { bd:serviceParam wikibase:language "[AUTO_LANGUAGE],en" }
11 }

```

35 results in 361 ms

Code Download Link



# Use case: természetes nyelvi interfész

- Chatbot, keresés, személyes ágens
- Pl. “I want to put a lady on the Gunners for a whitewash on Saturday afternoon.”
- Ami emberi nyelven ezt jelenti: “I would like to place a bet of £5 on Arsenal to win without conceding a goal in the forthcoming game on Saturday.”

Forrás: <https://www.dataversity.net/semantic-web-semantic-technology-trends-2019/#>

# QA: state of the art






melyek leonardo leghíresebb festményei?

All
Images
News
Videos
More
Settings
Tools

About 34,000 results (0.54 seconds)

**Minden idők 10 leghíresebb festménye | Látványosságok**  
<https://latvanyossagok.hu> › Világ ▼ [Translate this page](#)  
A Vénusz születése Sandro Botticelli **festménye, melyet** 1485-87 körül készített. ... Az Utolsó Vacsora egy 15. századi freskó Milánóban, **melyet Leonardo da ...**

**Images for melyek leonardo leghíresebb festményei?**

→ More images for melyek leonardo leghíresebb festményei?
Report images

## A világ 10 leghíresebb festménye - Bakonyi Art

<https://bakonyi.art/a-vilag-10-leghiresebb-festmenye/> ▼ [Translate this page](#)

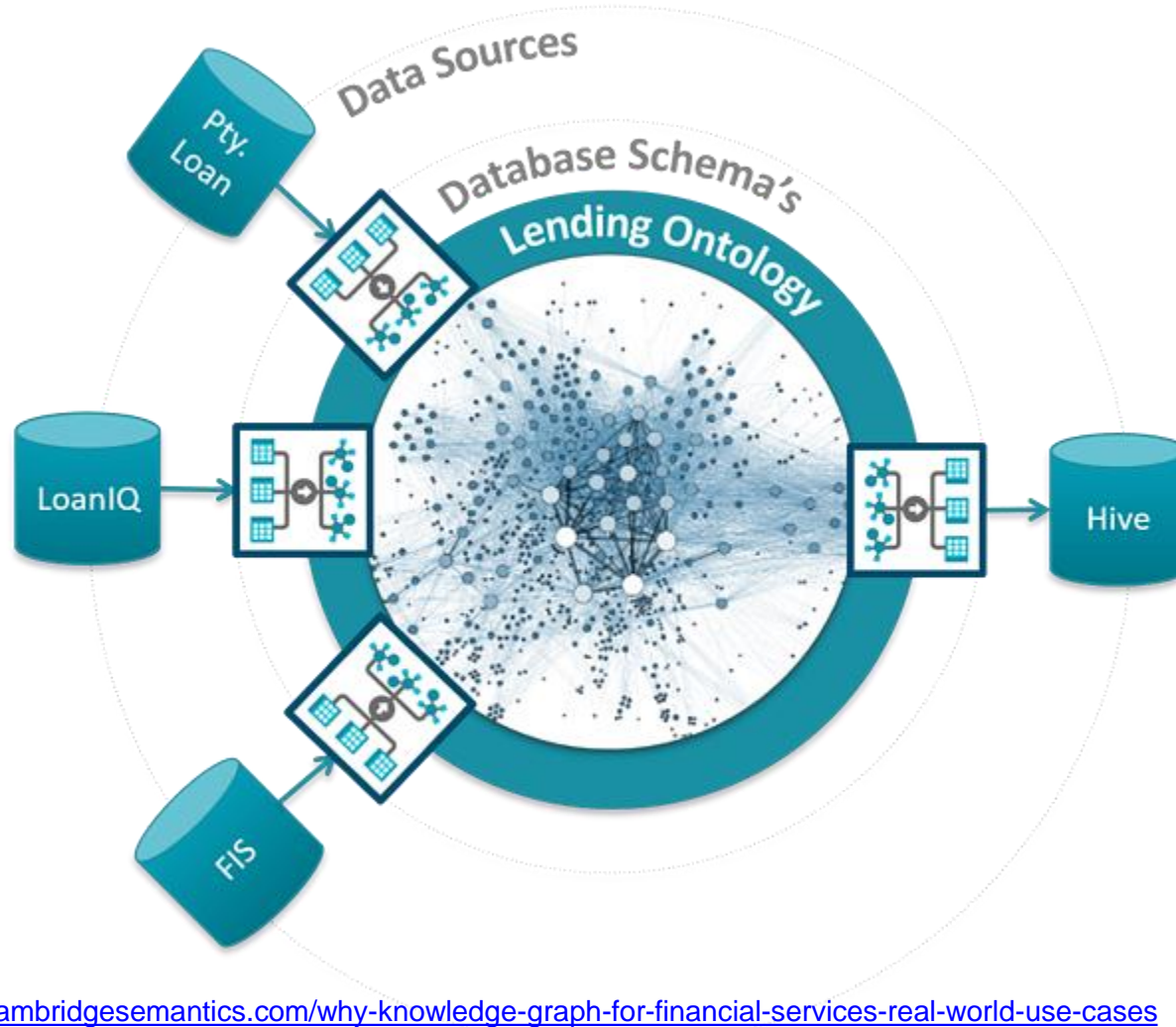
Sep 10, 2018 - Törp papa - A világ 10 **leg híresebb festménye**; Malatyinszki Pálné - Festészeti stílusok és irányzatok; Doknális Angéla - Festészeti stílusok és ...

# Use Case: Feature Engineering & Selection





# Use Case: Data Migration / Governance



Forrás: <https://blog.cambridgesemantics.com/why-knowledge-graph-for-financial-services-real-world-use-cases>



# Metaadatok: RDF

# Első lépés: metaadatok

“George Bush, the 41st President of the United States is the father of George W. Bush, the 43rd President of the United States.”

```
<div vocab="http://schema.org/" typeof="Person">
  <span property="name">George Bush</span>, the
  <span property="disambiguatingDescription">41st President of the
    United States</span>
  is the father of
  <div property="children" typeof="Person">
    <span property="name">George W. Bush</span>, the
    <span property="disambiguatingDescription">43rd President of the
      United States</span>.
  </div>
</div>
```

- Többféle formátum létezik
  - Microdata, RDFa, JSON-LD
- A legtöbb ontológia-nyelv az RDF-en alapul



# Resource Description Framework

- Az **RDF** általános és absztrakt **modell** amely bármilyen típusú metaadat leírására alkalmas, bármilyen olyan dologról, avagy erőforrásról (resource), amely egyedi, webes azonosítóval (URI vagy IRI) rendelkezik.
- Az **RDF** W3C ajánlás (2004 óta).

# RDF alapok

Az RDF-ben az információ állítások együttese.

Egy állítás egy dolog (resource) egy tulajdonságáról állít valamit.

- amiről állítunk valamit: resource, subject
- a tulajdonság: predicate, property
- a tulajdonság értéke: object

Van XML reprezentáció

Notation3 (N3) / Turtle:

`<#jakab> <#szeret> <#julcsi> .`

Gráf:



# RDF állítások

- Az állítás elemeit egy URI (IRI) azonosítja. Az érték (object) lehet egy szöveges vagy numerikus érték (literal) is, nem csak URI.

`<#jakab> <#eletkor> „34” .`

- A tulajdonság / property a másik két elem kapcsolatát fejezi ki.

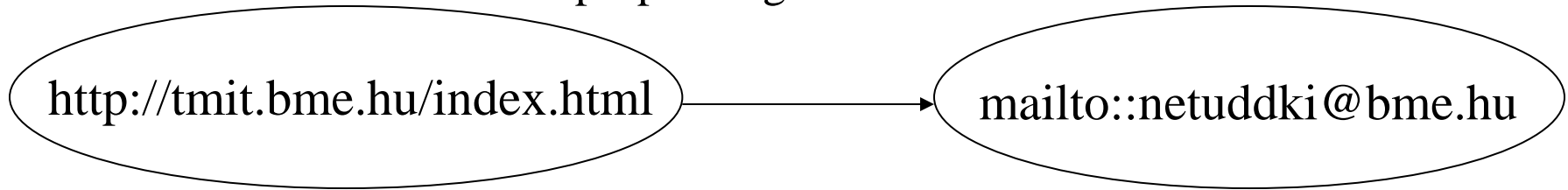
`<#jakab> <#gyereke> <#pista> .`

`<#jakab> <#tanul> <#BME> .`



# RDF gráf példa

`http://purl.org/dc/terms/creator`



**Erőforrás**

**Tulajdonság**

**Érték**

SUBJECT

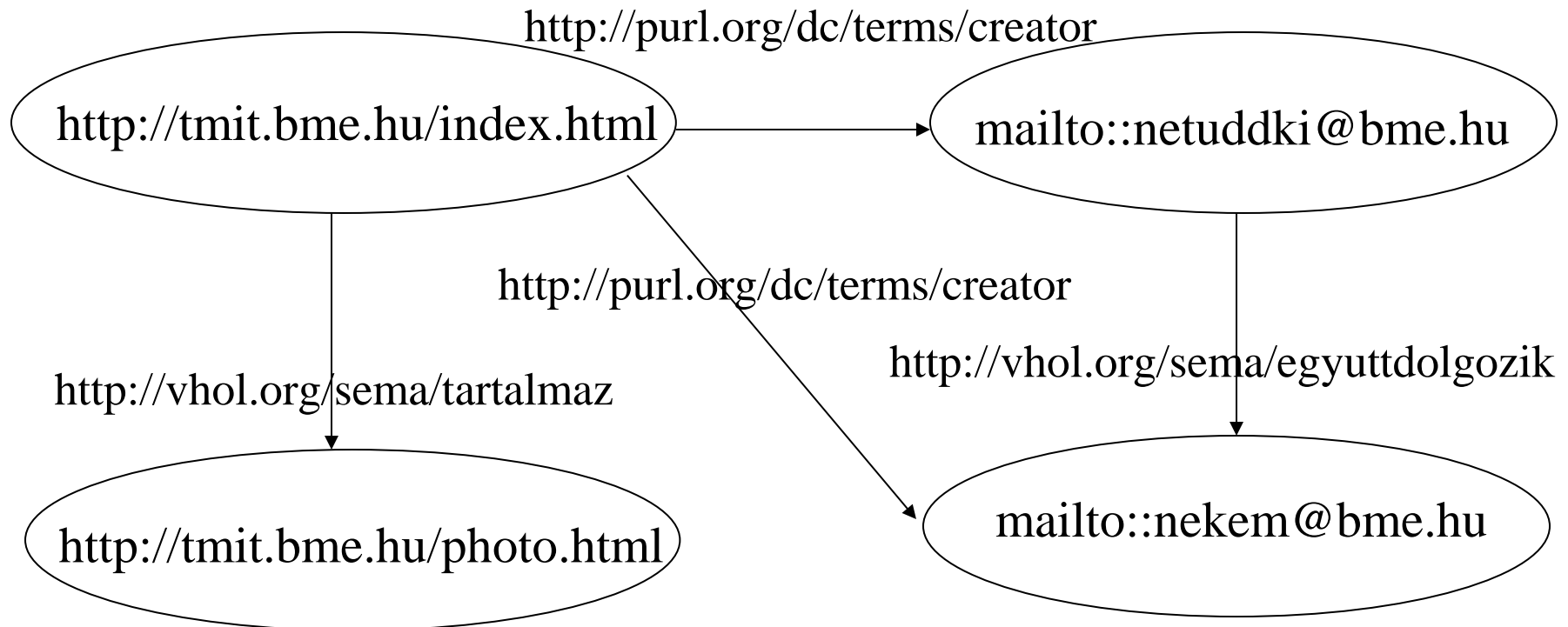
PREDICATE

OBJECT

„Az index.html létrehozója netuddki.”

`Creator(index.html, netuddki@bme.hu)`

# Elem-hármas és gráf



**„Az *index.html* létrehozója *netuddki* és *nekem* (akik együtt dolgoznak). Az *index.html* tartalmazza *photo.html*-t.”**

# Szemantika

```
<#jakab> <#eletkor> "34";  
<#szemszine> "kék" .
```

- A „jakab”, „eletkor” és „szemszine” karakterek URI-k, a gép számára semmi jelentést nem hordoznak ezen kívül !
- Megoldás: közös szótárak, “ontológiák” definiálása

# Példa: Dublin Core (DC)

- Szabványos tulajdonságok
  - <http://purl.org/dc/terms/title>
  - <http://purl.org/dc/terms/language>
  - <http://purl.org/dc/terms/creator>
- DC-t használó alkalmazások ismerik a tulajdonságok jelentését

# Példa: schema.org

schema.org

[Home](#)
[Schemas](#)
[Documentation](#)

## Person

[Thing](#) > [Person](#)

A person (alive, dead, undead, or fictional).

[\[more...\]](#)

Property	Expected Type	Description
<b>Properties from <a href="#">Person</a></b>		
<a href="#">additionalName</a>	<a href="#">Text</a>	An additional name for a Person, can be used for a middle name.
<a href="#">address</a>	<a href="#">PostalAddress</a> or <a href="#">Text</a>	Physical address of the item.
<a href="#">affiliation</a>	<a href="#">Organization</a>	An organization that this person is affiliated with. For example, a school/university, a club, or a team.
<a href="#">alumniOf</a>	<a href="#">EducationalOrganization</a> or <a href="#">Organization</a>	An organization that the person is an alumni of. Inverse property: <a href="#">alumni</a> .
<a href="#">award</a>	<a href="#">Text</a>	An award won by or for this item. Supersedes <a href="#">awards</a> .
<a href="#">birthDate</a>	<a href="#">Date</a>	Date of birth.
<a href="#">birthPlace</a>	<a href="#">Place</a>	The place where the person was born.



# Névterek

Szabványos szótár elemeinek URI-ja azonosan kezdődik: azonos névtérbe tartoznak

```
@prefix dcterms: <http://purl.org/dc/terms/> .  
<http://my.domain/index.html> dcterms:title  
    „Névterek”.
```

prefix nélkül

```
<http://my.domain/index.html>  
<http://purl.org/dc/terms/title> „Névterek”.
```

# Reifikáció

- „Józsi mondta, hogy Jakabnak Pista a gyereke”

my:jakab my:gyereke my:pista . (saját ontológia a my névtérben)

\_:allitas123 rdf:type rdf:Statement .

\_:allitas123 rdf:subject my:jakab .

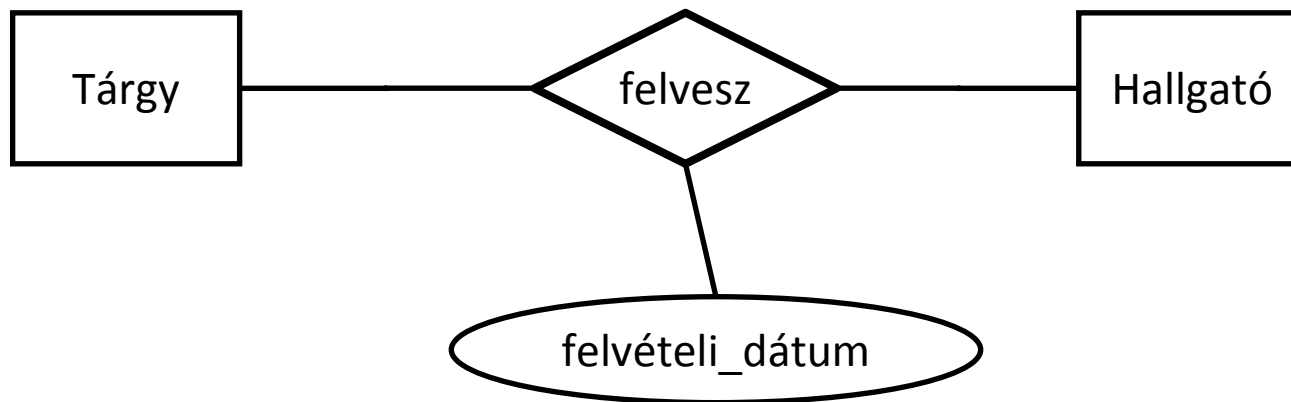
\_:allitas123 rdf:predicate my:gyereke .

\_:allitas123 rdf:object my:pista .

my:jozsi my:mondta \_:allitas123 .

\_ névtér: üres csomópontoknak lokális id definiálásához

# RDF: nem bináris relációk?



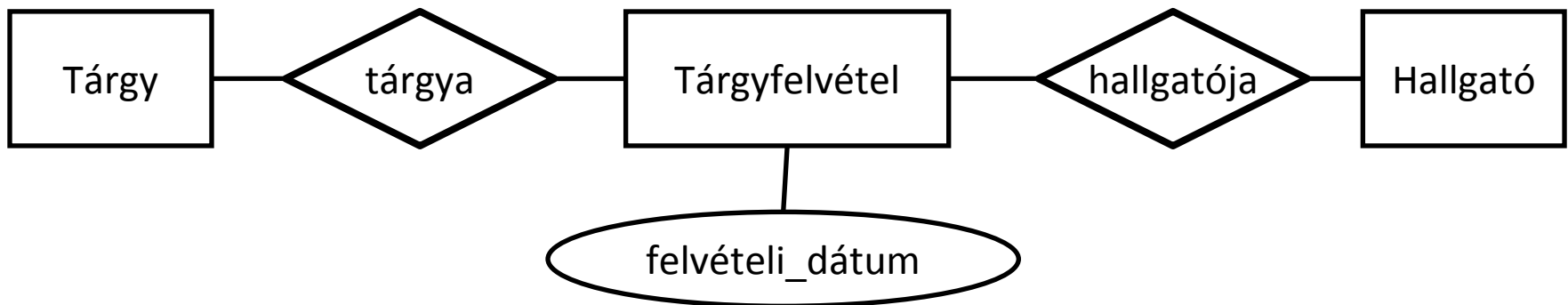
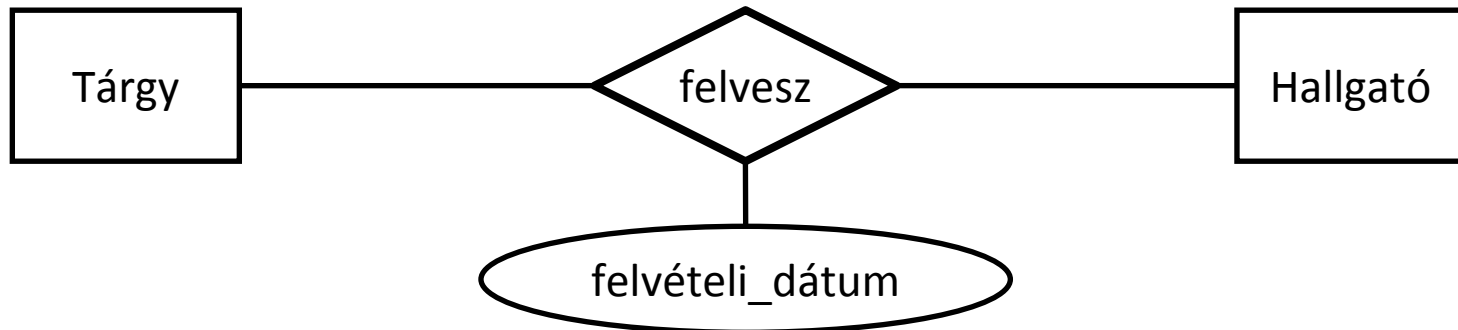
# Reifikáció (másként)

```
my:jakab rdf:type my:Hallgato .  
my:adatb2 rdf:type my:Targy .  
my:jakab :felvesz my:adatb2 .
```

hol marad a dátum? Helyette:

```
[ ] a :Targyfelvetel ; („a” : rdf:type rövidítése, [] az üres csomópont)  
  my:targya :adatb2 ;  
  my:hallgatoja :jakab ;  
  my:datuma „2018-01-10” .
```

# Azaz: nem bináris relációk

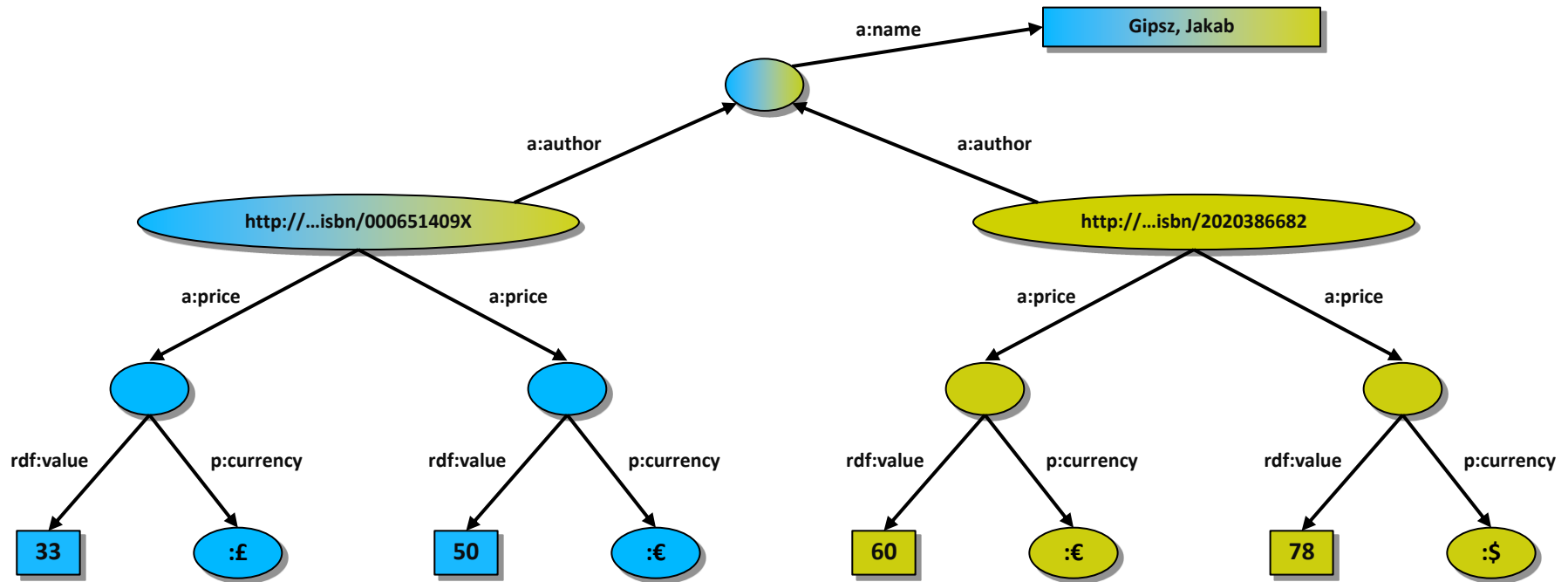


# SPARQL

- RDF gráfok lekérdező nyelve
- W3C szabvány (2008)
- Lekérdezés-típusok
  - SELECT: adatok lekérése
  - CONSTRUCT: új RDF gráf készítése
    - hasonló az XSLT-hez, XQuery-hez
  - ASK: igen/nem kérdések (igaz-e az állítás?)
  - DESCRIBE: egy RDF gráf/csomópont adatai
    - a SPARQL szervertől függ, mit ad vissza

# SPARQL SELECT példa

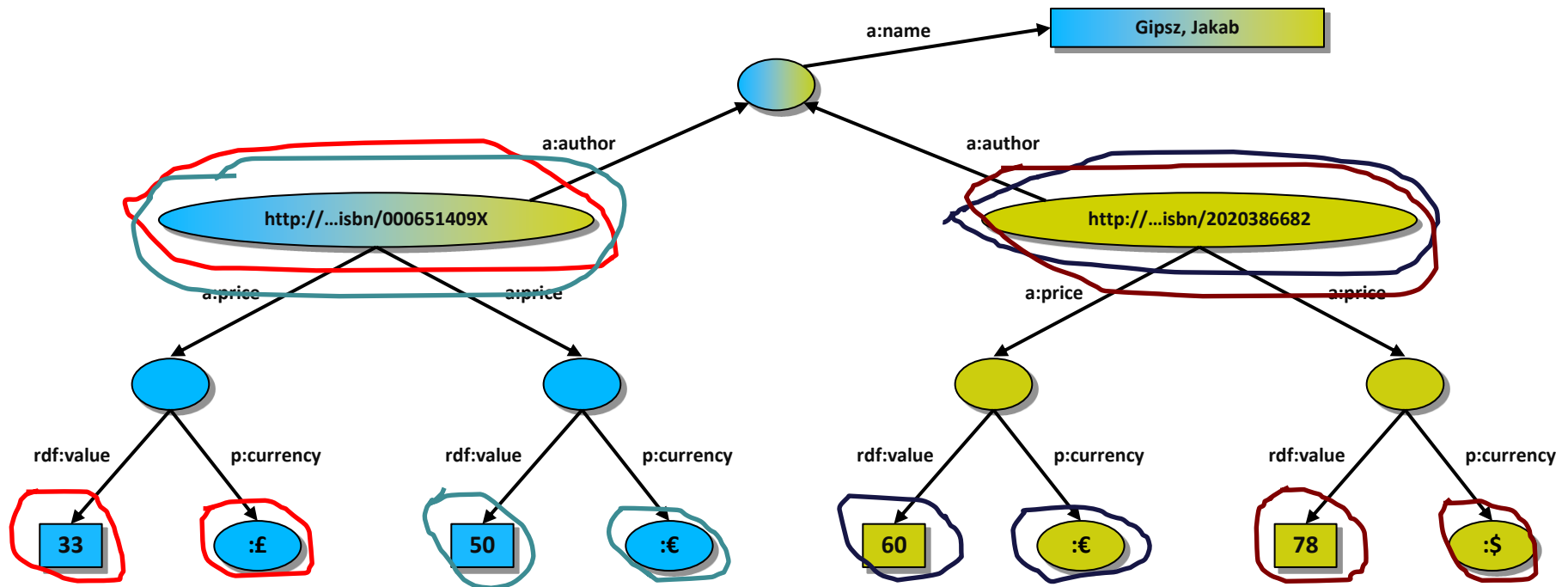
```
SELECT ?isbn ?price ?currency # note: not ?x!  
WHERE {?isbn a:price ?x. ?x rdf:value ?price. ?x p:currency ?currency.}
```



# SPARQL SELECT példa

```
SELECT ?isbn ?price ?currency # note: not ?x!  
WHERE {?isbn a:price ?x. ?x rdf:value ?price. ?x p:currency ?currency.}
```

Eredmény: [<...409X>,33,:£], [<...409X>,50,:€],  
[<...6682>,60,:€], [<...6682>,78,:\$]

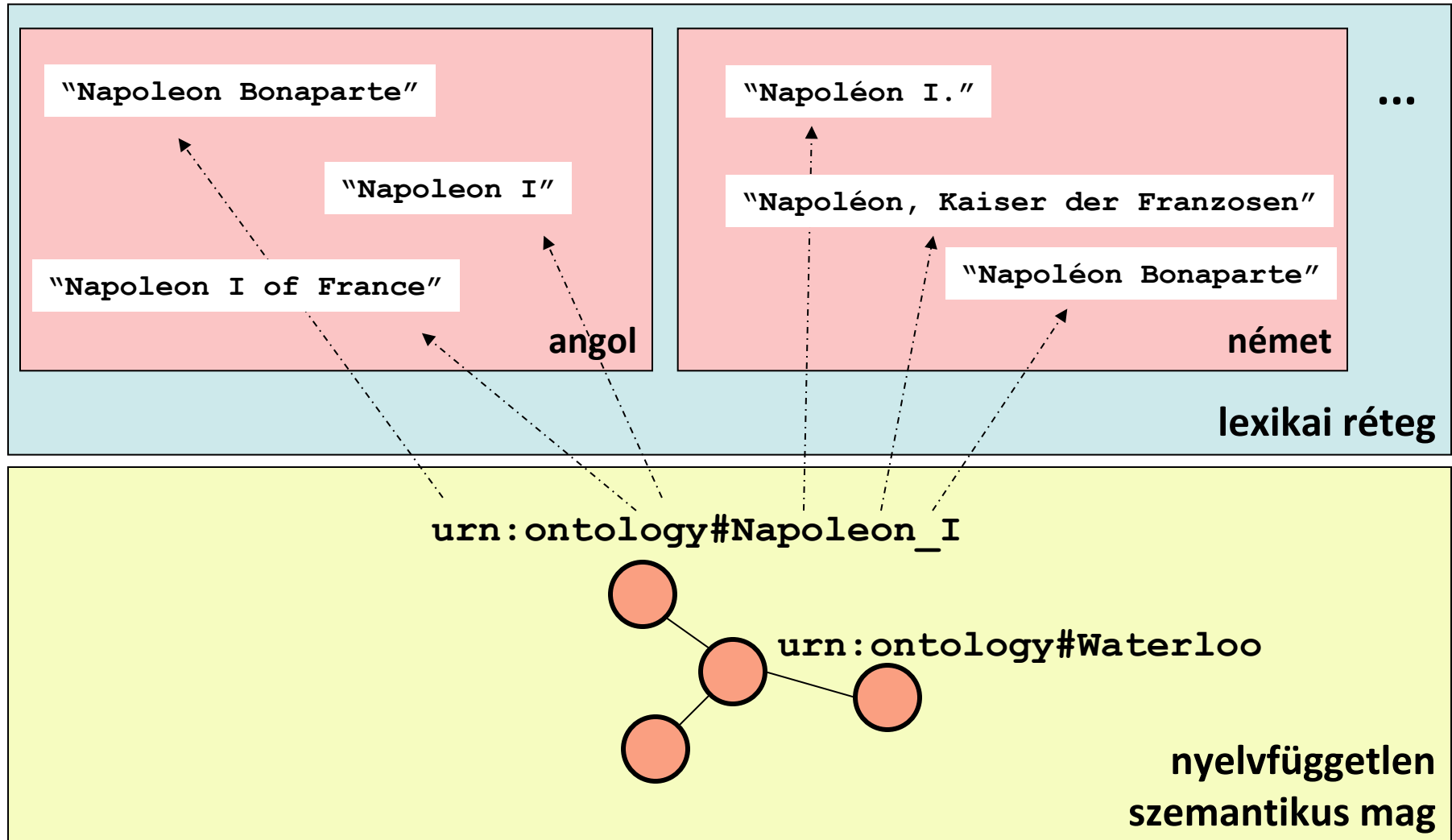




# rdfs:label, rdfs:comment

- Best practice: „beszélő” URI-k
  - legtöbbször angol nyelven
  - pl. #creator, #Truck
- De az URI-k akarmik is lehetnek!
  - #12345, #asdbmdfbmfnfs
  - pár RDF-t használó szótár ilyen, pl. Wikidata
- rdfs:label – emberek számára olvasható név
  - xml:lang
  - többnyelvű ontológiák
- rdfs:comment – emberek számára olvasható leírás
  - ez is lehet többnyelvű

# Szemantikus mag és lexikai réteg





# Ontológiák: RDFS, OWL

# Ontológia a filozófiában

- „a létről szóló tan”
- *ontosz* (lenni, létezni) + *logosz* (tudomány)
  - Arisztotelész, Aquinói Szent Tamás (istenérvek)
  - XX. Század: Husserl, Hartmann, Heidegger,
- “Érzékfeletti irracionális intuíció útján felfogott lét legáltalánosabb fogalmainak rendszere” (Filozófiai kislexikon)
- **Az ontológia a legfontosabb fogalmak és viszonyaik leírása.**

# Ontológia az informatikában

- Gruber: „egy adott felhasználói csoport által egy adott témakörben közösen használt világkép formális leírása”
  - shared, explicit, formal specification of a conceptualization
- Tehát már nem akarja senki a teljes tudást leírni, részterületeket kell megcélózni.
- egyezményes terminológiát állít fel egy közös érdeklődésű közösség tagjai között.
  - A tagok lehetnek emberek vagy gépi ügynökök.

# Ontológia definíció

Shared, explicit, formal  
specification of a conceptualization.

# Ontológia definíció

Shared, explicit, formal  
specification of a  
**conceptualization.**

a fejünkben levő  
mentális modell

# Ontológia definíció

Shared, explicit, formal  
**specification** of a  
conceptualization.



leírva



# Ontológ

számítógép számára  
feldolgozható formában

Shared, explicit, **formal**  
specification of a conceptualization.



# Ontológia definíció

minden ami számít, benne  
van, ami nincs benne, az  
nem létezik

Shared, **explicit**, formal  
specification of a conceptualization.

# Ontológia definíció

egy közösség minden tagja érti

**Shared**, explicit, formal  
specification of a conceptualization.

# Ontológia alapok

- **Fogalom, osztály (Concept, class)** – dolgok halmaza, osztálya
  - állat, számítógép, ...
  - taxonómia: subclass, pl. számítógép -> gép
- **Példány (Instance, individual)** – példányok, a halmazok elemei
  - gorilla, Dell Latitude D620, ...
- **Reláció (property, role, relation)** – példányok közti kapcsolatok
  - szeret, ismer, életkor
  - taxonómia! – szeret -> ismer
- **Attribútum (attribute)** – példányok és primitív értékek közti kapcsolatok
  - életkora: 25

# Mi is a szemantika a gépeknek?

- Leggyakoribb logikai alapú formalizmusok esetén: halmazelméleti alapú modell-elmélet
- A világ releváns elemeit (domain of discourse) egy halmaz tartalmazza ( $\Delta$ )
- A világ objektumait  $\Delta$  elemeinek tekintjük
  - az osztályok  $\Delta$  részhalmazai
  - A (bináris) relációk  $\Delta \times \Delta$  részhalmazai

# Mi a szemantika (2)

- Egy  $\mathcal{I}$  interpretáció (mapping) köti össze az ontológia elemeit  $\Delta$  -val
  - $\text{Gorilla}^{\mathcal{I}} \in \Delta$ ,  $\text{szeret}^{\mathcal{I}} \in \Delta \times \Delta$
- A formális nyelv elemeit a modellen értelmezzük
  - pl. C alfogalma D-nek  $\rightarrow C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
  - pl. szeret tulajdonsága ismer-nek  $\rightarrow \text{szeret}^{\mathcal{I}} \subseteq \text{ismer}^{\mathcal{I}}$
  - pl. Amy példánya Gorillának  $\rightarrow \text{Amy}^{\mathcal{I}} \in \text{Gorilla}^{\mathcal{I}}$

# Mikor „igaz” egy állítás

- Open World Assumption (OWA)
  - nem ismerjük a világ összes tényét
  - pl. mert elosztott a rendszer (Világháló)
  - csak az igaz, ami az összes lehetséges modell (összes lehetséges  $\mathcal{I}$ ) esetén (halmazelméletileg) igaz
  - csak az hamis, amihez egyáltalán nincs olyan interpretáció, hogy (halmazelméletileg) igaz
  - a többi állítás, amelyhez van olyan interpretáció is, hogy igaz, meg olyan is, hogy hamis, az „nem tudom”.

# Closed World Assumption (CWA)

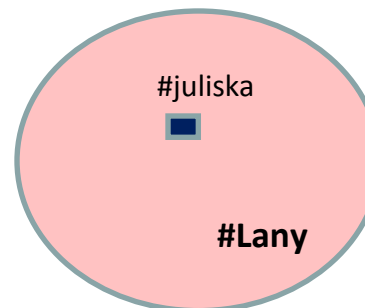
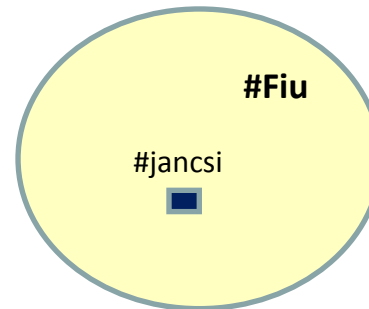
- Adatbázisok esetén gyakori
- Egyszerűbb
- Amiről nem tudjuk, hogy igaz, az hamis
- Lényegében az adatbázisunk az egyetlen modell
- Nincs „nem tudom” állítás



# OWA modell példa

#jancsi a #Fiu

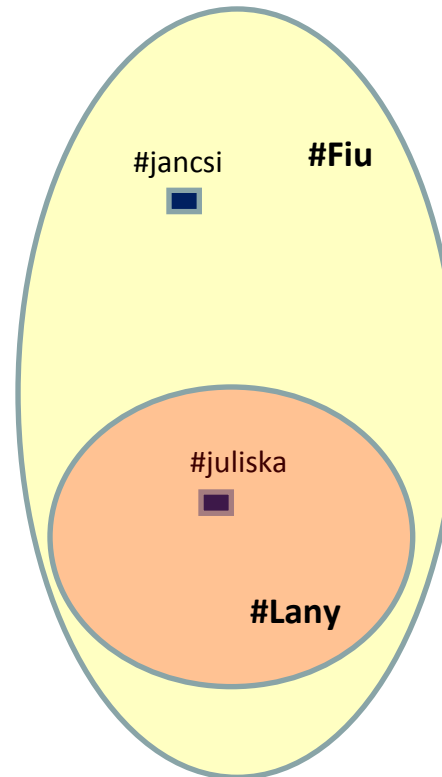
#juliska a #Lany



# OWA modell példa

#jancsi a #Fiu

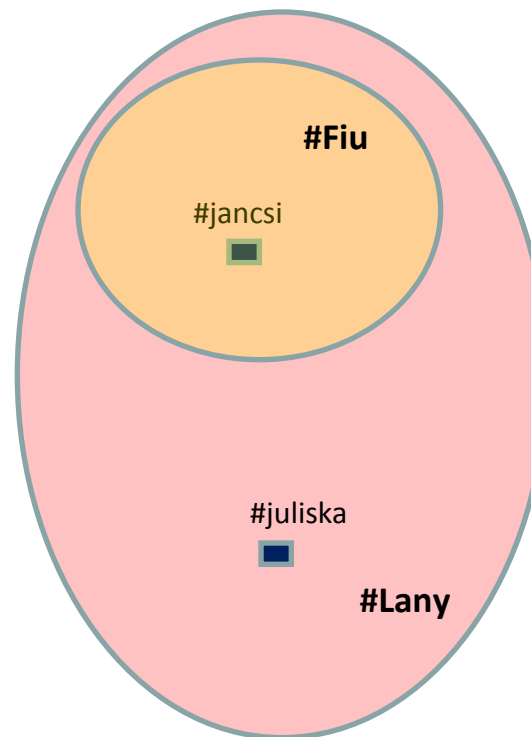
#juliska a #Lany



# OWA modell példa

#jancsi a #Fiu

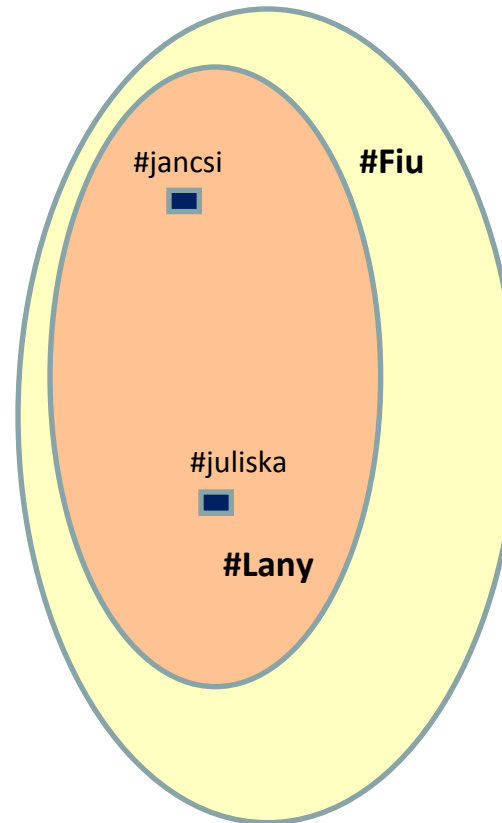
#juliska a #Lany



# OWA modell példa

#jancsi a #Fiu

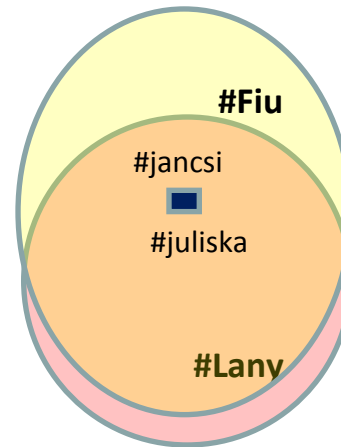
#juliska a #Lany



# OWA modell példa

#jancsi a #Fiu

#juliska a #Lany

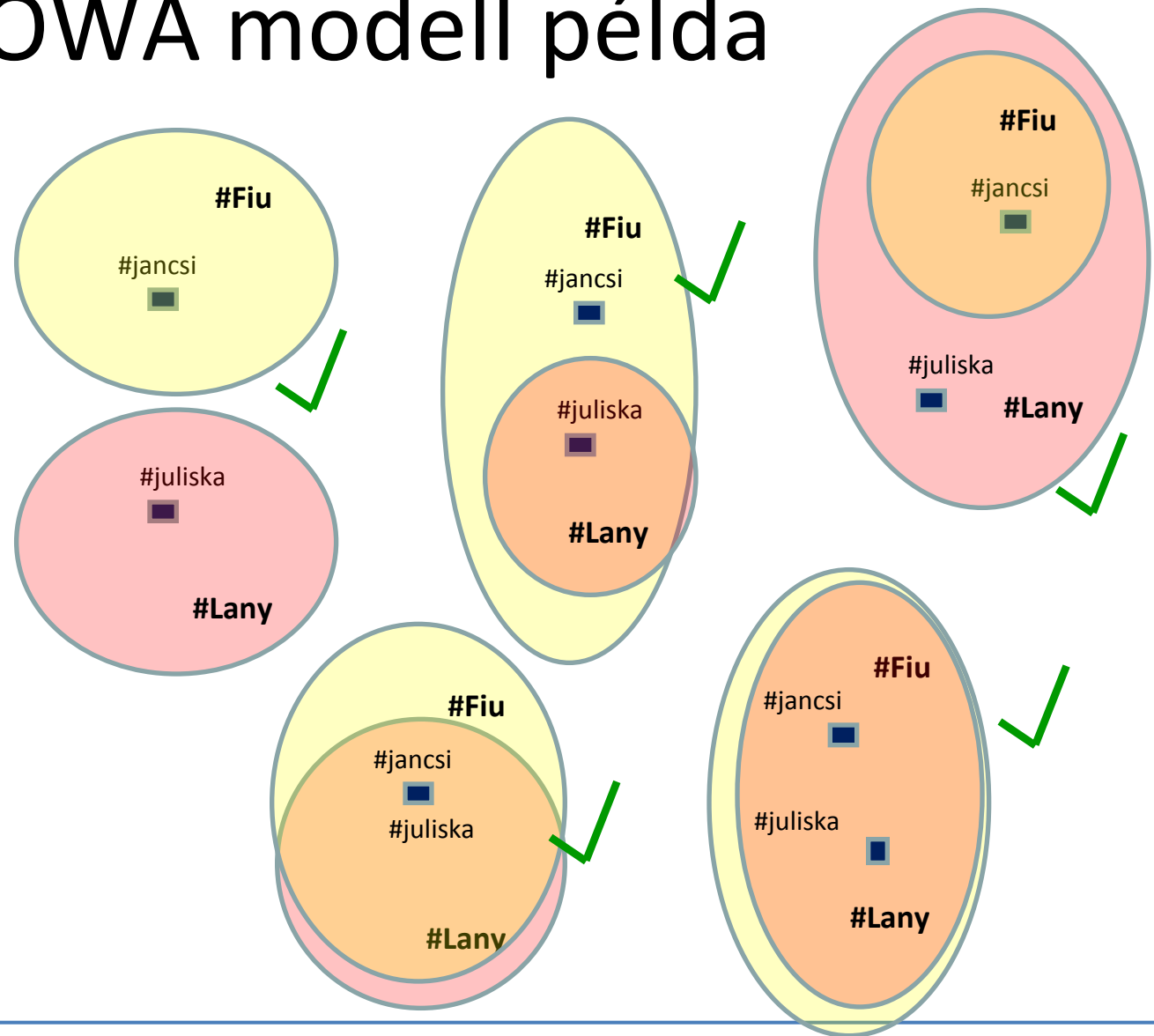


# OWA modell példa

#jancsi a #Fiu

#juliska a #Lany

#jancsi a #Fiu ?



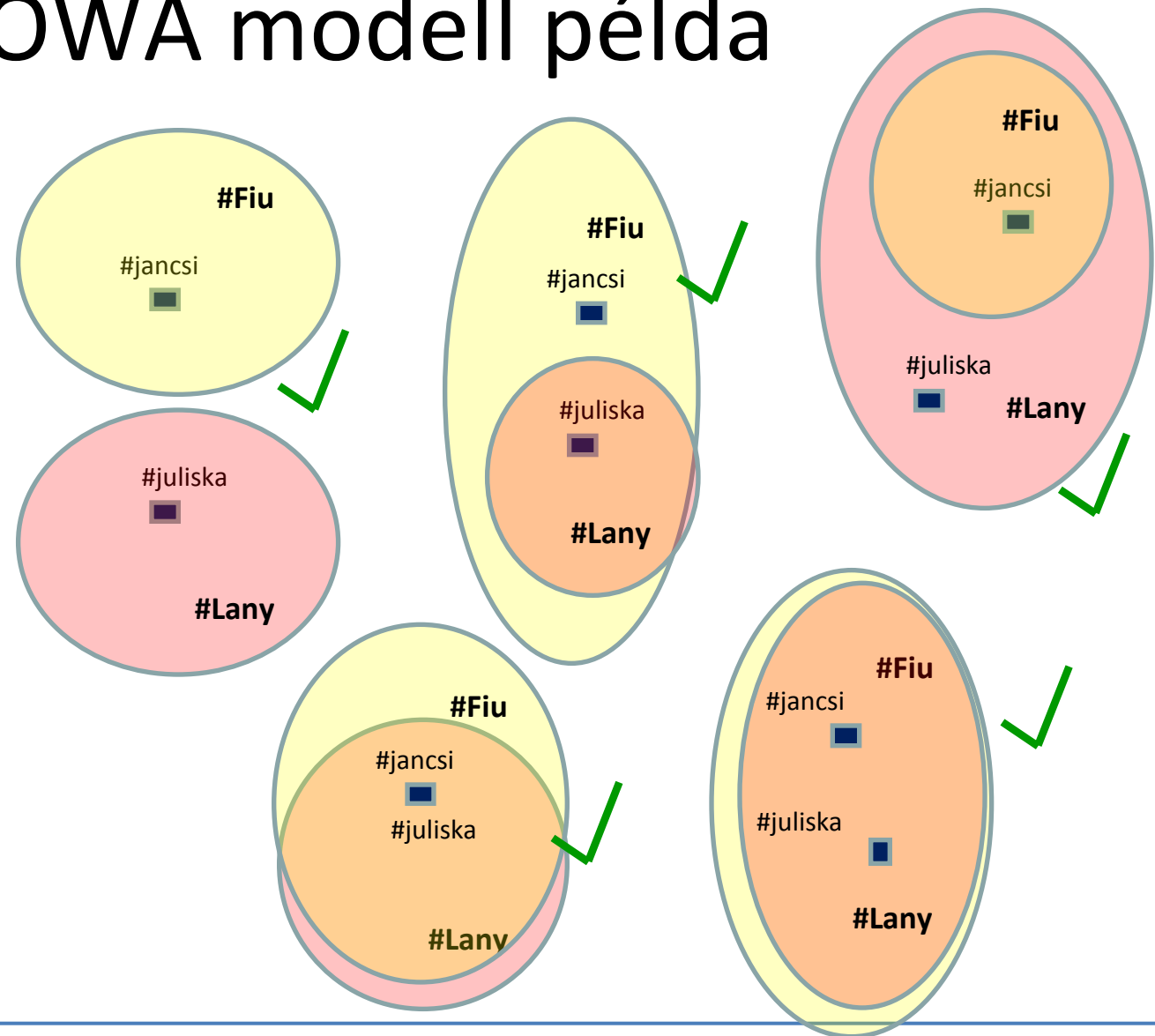
# OWA modell példa

#jancsi a #Fiu

#juliska a #Lany

#jancsi a #Fiu ?

IGEN

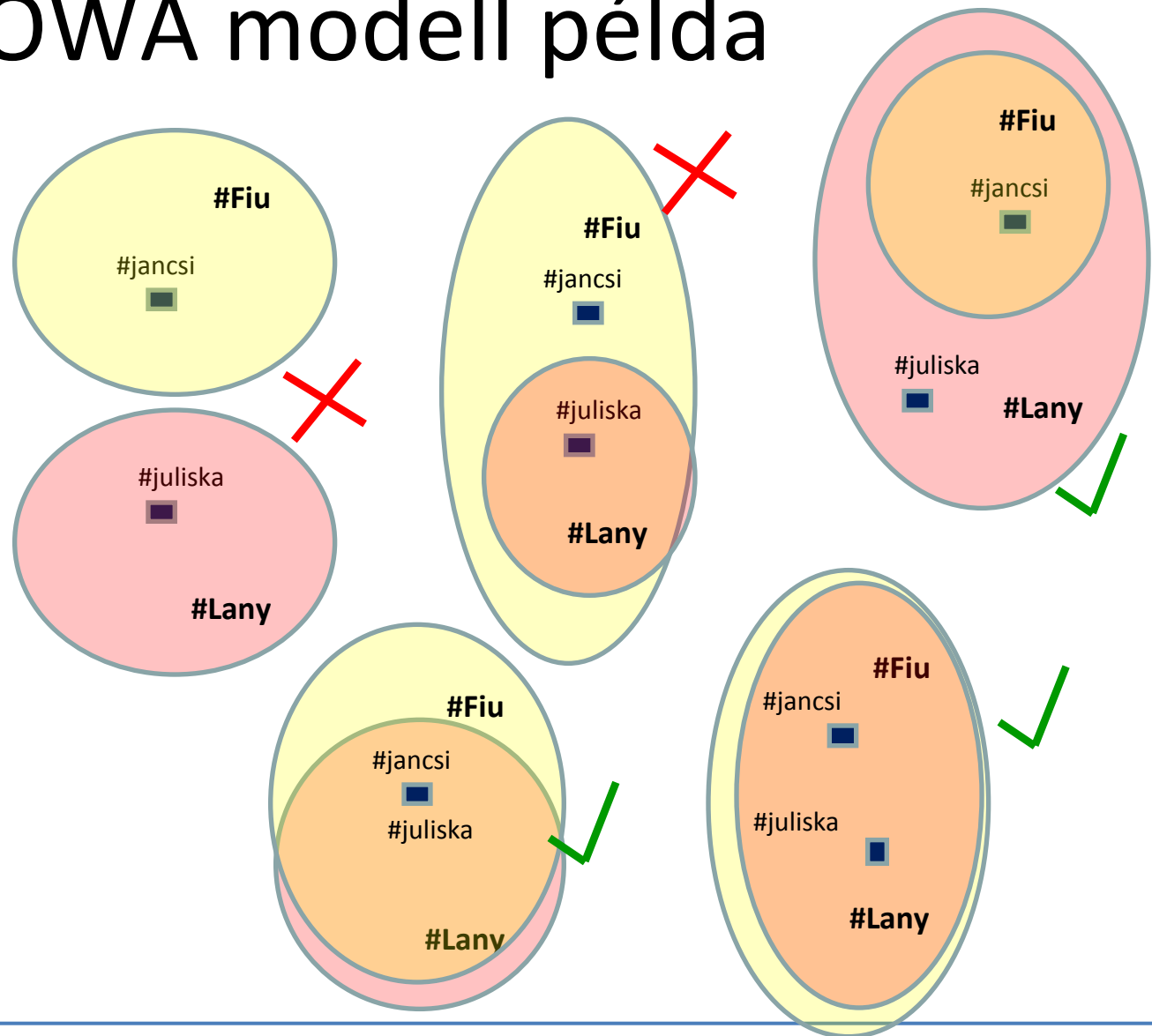


# OWA modell példa

#jancsi a #Fiu

#juliska a #Lany

#jancsi a #Lany ?





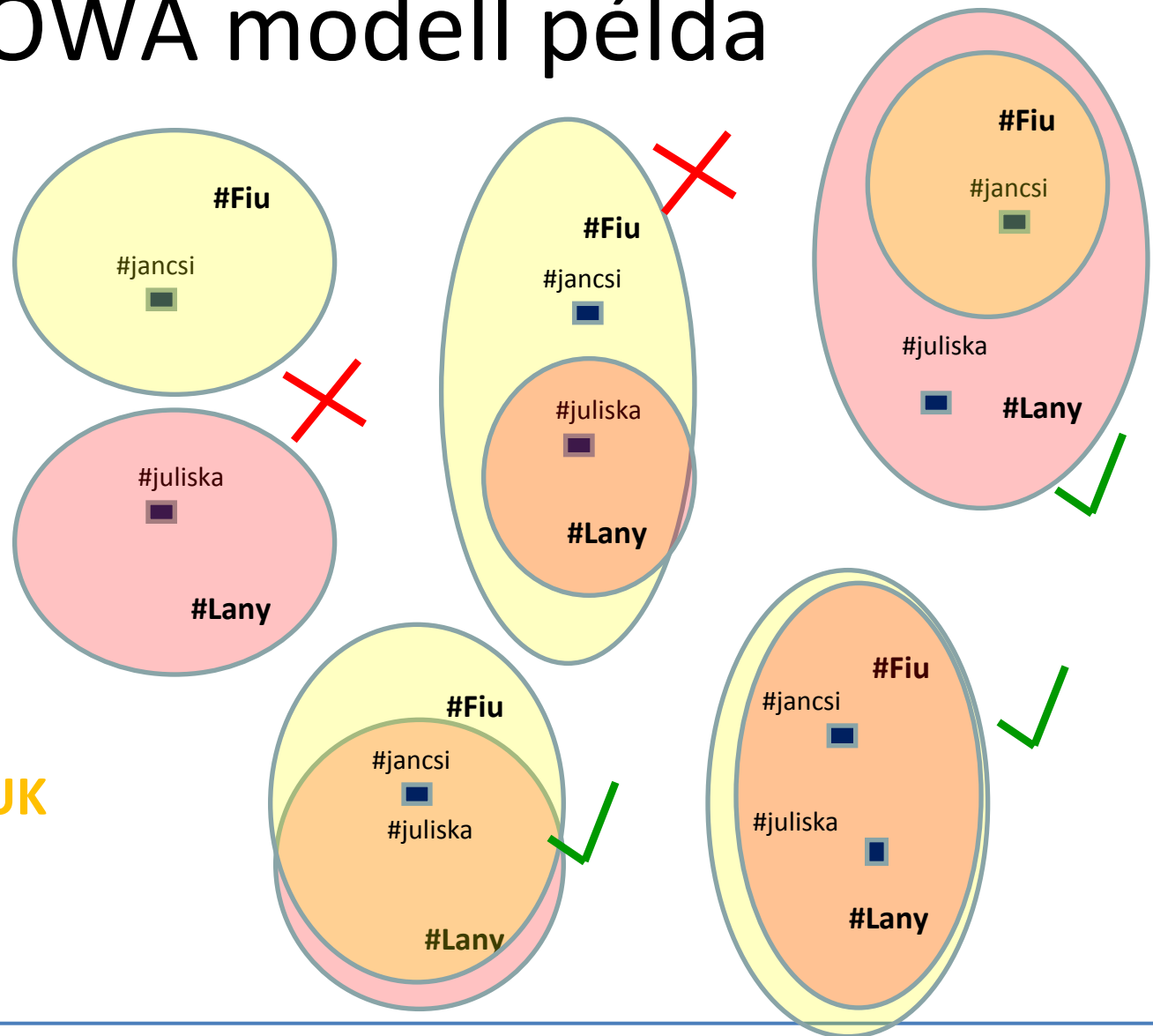
# OWA modell példa

#jancsi a #Fiu

#juliska a #Lany

#jancsi a #Lany ?

NEM TUDJUK



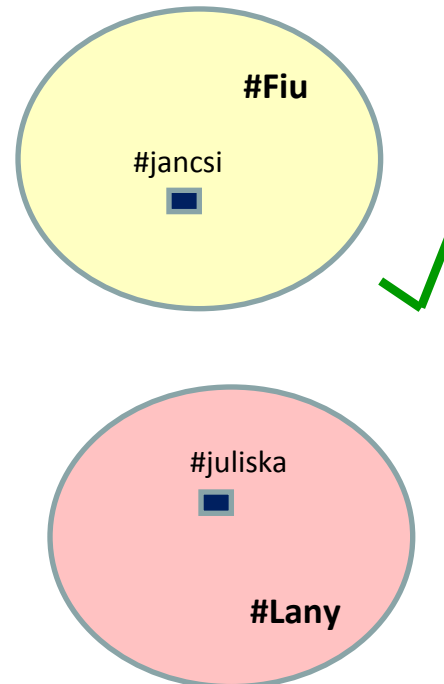
# CWA modell példa

#jancsi a #Fiu

#juliska a #Lany

**#jancsi a #Fiu ?**

**IGEN**



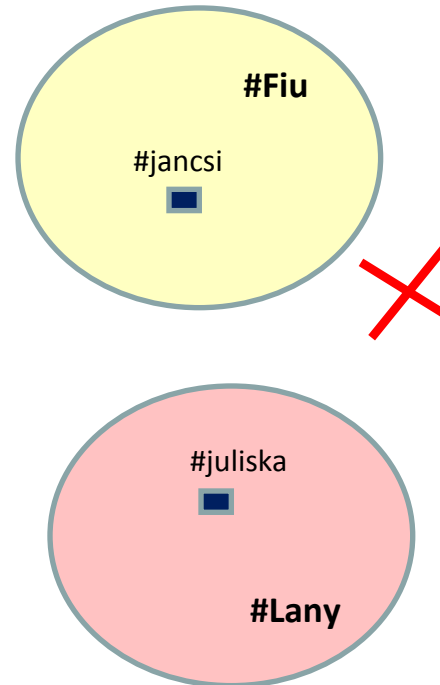
# CWA modell példa

#jancsi a #Fiu

#juliska a #Lany

#jancsi a #Lany ?

**NEM**



# Monoton vs. nem-monoton logika

- Monoton: ha egy állítás igazságtartalma nem változik
  - OWA
  - max. „nem tudom”-ról lesz igaz v. hamis
- Nem-monoton: állítás igazságtartalma változhat
  - CWA
  - pl.  $\text{nem lány}(x) \rightarrow \text{fiú}(x)$ ,  $\text{ember}(\text{Juliska})$  esetén  $\text{fiú}(\text{Juliska})$  igaz, amíg nem mondjuk, hogy  $\text{lány}(\text{Juliska})$

# RDF Séma

Az RDF szabványhoz szorosan kapcsolódik az **RDF Schema (RDFS)** nyelv, amellyel egyszerű „szótárakat”, **ontológiákat** (csomópontok és tulajdonságok előre definiált halmazát) definiálhatunk.

W3C ajánlás (2004 óta - mint az RDF)

# RDF Schema elemei

- `rdfs:Class` – osztályok, fogalmak
- `rdf:Property` – tulajdonságok
- `rdfs:subClassOf` – osztályhierarchia
- `rdfs:subPropertyOf` – tulajdonsághierarchia (!)
- `rdf:type` – példány
- `rdfs:domain` – tulajdonság értelmezési tartománya
- `rdfs:range` – tulajdonság értékkészlete
  - typed value (!)

# RDFS (folytatás)

- Egy objektum több osztályban lehet (!), melyek között nem kell hierarchikus viszonynak lenni.
  - vö. OO programozási nyelvek
- Egy osztálynak ill. tulajdonságnak több őse is lehet.
  - „többszörös öröklődés”
- Konvenció (nem szabvány, nem kötelező):
  - Osztály azonosító nagy kezdőbetűvel
  - Tulajdonság kis kezdőbetűvel

# RDFS Példa

```
:Ember a rdfs:Class (a = rdf:type)
:Hallgato rdfs:subClassOf :Ember
:Jakab a :Hallgato
:ismer a rdf:Property
:szeret a rdf:Property
:szeret rdfs:subPropertyOf :ismer
:ismer rdfs:domain :Ember (szeret domain-je is!)
:ismer rdfs:range :Ember (szeret range-e is!)
:Jakab :szeret :Julcsi (:Jakab :ismer :Julcsi is!)
```



# Metamodellezés

- Halmazelméleti szemantikát alkalmazó logikákban általában el kell dönteni, hogy valami halmaz (fogalom), hatványhalmaz (reláció) vagy halmazelem (példány)
- Néha azonban hasznos ezeket keverni
  - #gorilla a #faj, #Amy a #gorilla
- Akkor is hasznos, ha az ontológia elemeiről akarunk állításokat tenni
  - pl. #gorilla dct:creator #Gipsz\_Jakab
  - pl. #gorilla rdfs:label „Ape“

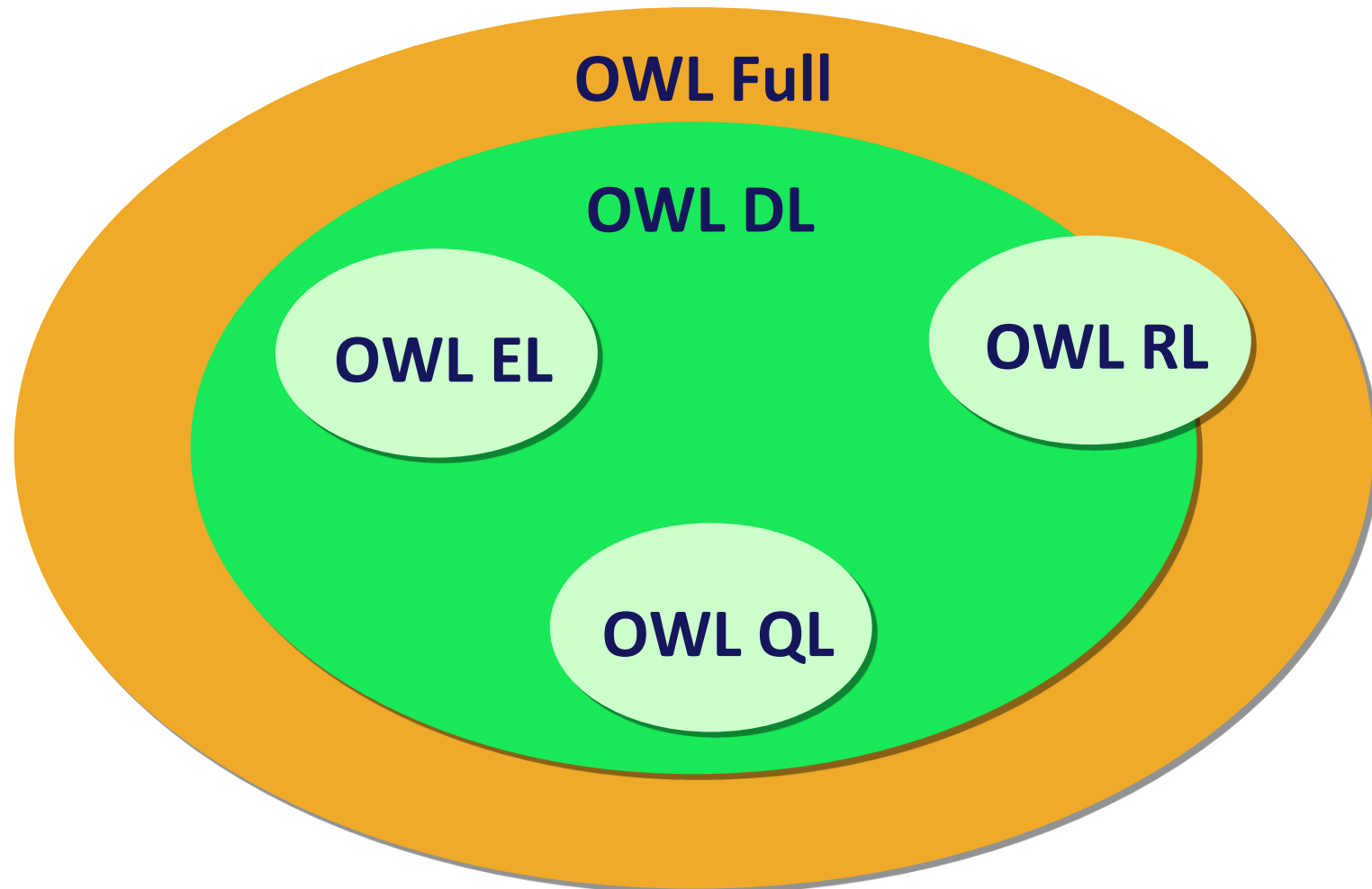
# Web Ontology Language (OWL)

- W3C szabvány ontológia formalizmus
  - OWL1: 2004
  - OWL2: 2009
- Miért nem elég az RDFS?
  - Nem elég kifejező, túl gyenge.
    - lokalizált range: gyereke mutasson emberre ember esetén, elefántra elefánt esetén
    - kardinalitás: **minden** embernek van anyja, két gyerekes apukának **pontosan két** gyereke van
    - tulajdonságok túl egyszerűek, nincs tranzitív, inverz, ..., pl. #része
  - Mégis túl nehéz
    - metamodellezés miatt

# Kompromisszum: kifejezőerő és hatékonyság

- Általában is elmondható: minél kifejezőbb egy formalizmus, annál kevésbé hatékony benne a következtetés
- Sok formalizmus nem is **eldönthető**
  - nem garantált, hogy kérdéseink véges időben megválaszolhatók
  - pl. elsőrendű logika „félig eldönthető”: egy állítás hamisságát nem tudjuk mindig véges idő alatt bizonyítani
  - pl. OWL nem eldönthető (!)

# OWL fajták



# OWL Full

- Minden tekintetben az RDFS kiterjesztése
  - owl:Class = rdfs:Class
  - owl:Thing = rdf:Resource
- Metamodellezés
- Nem eldönthető
- Adatcserére kiváló
- Ha következtetni (reasoning) akarunk, akkor egyszerűsíteni kell (OWL profil)

# Unique Name Assumption (UNA)

- Ha egy ontológia/adatbázis két eleme különböző névvel (URI-val, id-val) rendelkezik, akkor két különböző dolog
- OWL NEM követi az UNA-t!
- Azaz: Jancsi lehet Juliska is, amíg nem mondjuk, hogy különbözőek

# OWL példa: sameAs

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .  
@prefix owl: <http://www.w3.org/2002/07/owl#> .  
mailto:gj@gmail.com foaf:age 23 .  
mailto:jakab.g@gmail.com foaf:name „Gipsz Jakab” .  
→ még nem tudjuk összekapcsolni a két állítást!
```

```
mailto:gj@gmail.com owl:sameAs mailto:jakab.g@gmail.com .  
→ most már tudjuk Gipsz Jakab nevét és életkorát is
```

# OWL példa: differentFrom

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .  
@prefix owl: <http://www.w3.org/2002/07/owl#> .  
mailto:gj@gmail.com foaf:age 23 .  
mailto:jakab.g@gmail.com foaf:name „Gipsz Jakab” .  
→ még nem tudjuk, hogy a két mailto URL ugyanazt a  
személyt jelöli-e
```

```
mailto:gj@gmail.com owl:differentFrom  
mailto:jakab.g@gmail.com .  
→ most már tudjuk, hogy nem
```





# Szabályok CWA következtetés

# Szabályok – miért?

- OWL(-DL) inkább az ún. taxonómikus következtetésben erős
  - Minden „kétgyerekes apa” egyben „sokgyerekes apa” is?
  - Lehet-e példánya a „kétéltű járműnek”?
- Nem lehet felírni olyan következtetéseket, amikhez változók kellenek
  - pl. olyan nők, akiknek van kétgyermekes, elvált, „Jancsi” nevű barátjuk
  - pl. tranzitivitást sem lehetne, ha nem lenne beépítve
- Szabályokat nehéz (ill. néha, ha változók is kellenek, akkor nem lehet) felírni
  - ha A, B, C és D igaz, abból következik E

# RIF Core avagy Datalog

- RIF – Rule Interchange Format, W3C szabvány
- Horn logika funktorok nélkül
  - azaz  $p(a,b,c)$  OK,  $p(f(a),g(b),c)$  nem OK
- Terminológia: term (konstans vagy változó), predikátum, atom (=predikátum+termek), literál (esetleg negált atom), szabály
- Egy szabály fejében atom áll, testében literálok logikai és (AND) kapcsolata. A test és a fej között logikai implikáció áll fenn.

# Datalog példák

`fiu(x) :- not lany(x).`

`ember(x) :- fiu(x).`

`benne(x,z) :- benne(x,y), benne(y,z).`

`lakosa(l,o) :- ember(l), helyseg(h), lakik(l,h), benne(h,o).`

## Szabályok

`fiu('jancsi').`

`helyseg('Budapest').`

`lakik('jancsi','Budapest').`

`benne('Budapest','Magyarország').`

`benne('Magyarország','Europa').`

# Datalog példák

`fiu(x) :- not lany(x).`

`ember(x) :- fiu(x).`

`benne(x,z) :- benne(x,y), benne(y,z).`

`lakosa(l,o) :- ember(l), helyseg(h), lakik(l,h), benne(h,o).`

`fiu('jancsi').`

`helyseg('Budapest').`

`lakik('jancsi','Budapest').`

`benne('Budapest','Magyarország').`

`benne('Magyarország','Europa').`

Tényállítások

# Datalog példák

`fiu(x) :- not lany(x).`

`ember(x) :- fiu(x).`

`benne(x,z) :- benne(x,y), benne(y,z).`

`lakosa(l,o) :- ember(l), helyseg(h), lakik(l,h), benne(h,o).`

`fiu('jancsi').`

`helyseg('Budapest').`

`lakik('jancsi','Budapest').`

`benne('Budapest','Magyarország').`

`benne('Magyarország','Europa').`

**`lakosa('jancsi','Europa') ?`**

# Datalog példák

$\text{fiu}(x) \text{ :- not lany}(x).$

$\text{ember}(x) \text{ :- fiu}(x).$

$\text{benne}(x,z) \text{ :- benne}(x,y), \text{ benne}(y,z).$

$\text{lakosa}(l,o) \text{ :- ember}(l), \text{ helyseg}(h), \text{ lakik}(l,h), \text{ benne}(h,o).$

$\text{fiu}(\text{'jancsi'}).$

$\text{helyseg}(\text{'Budapest'}).$

$\text{lakik}(\text{'jancsi'}, \text{'Budapest'}).$

$\text{benne}(\text{'Budapest'}, \text{'Magyarország'}).$

$\text{benne}(\text{'Magyarország'}, \text{'Europa'}).$

**$\text{lakosa}(\text{'jancsi'}, \text{'Europa'}) ?$  IGEN**

# Triplestore-ok

- Specializált gráf adatbázisok RDF hármassok (triple), azaz állítások tárolására
  - akár több milliárd, sőt billió állítás (!)
- SPARQL interfész
- következtetés: RDFS, néha OWL egyes elemei
- Példák: Oracle (!), AllegroGraph, GraphDB
- Újabban: Neo4J, AWS Neptune (!)



# Összefoglalás

