SPARQL 1.0

Spracované podľa prednášok

- http://www.inf.tu-dresden.de/content/institutes/ki/cl/study/summer14/fswt/slides/FSWT2014-L06-SPARQL-Syntax.pdf
- http://www.inf.tu-dresden.de/content/institutes/ki/cl/study/summer14/fswt/slides/FSWT2014-L07-SPARQL-Semantics.pdf
- http://www.inf.tu-dresden.de/content/institutes/ki/cl/study/summer14/fswt/slides/FSWT2014-L08-SPARQL-Algebra.pdf

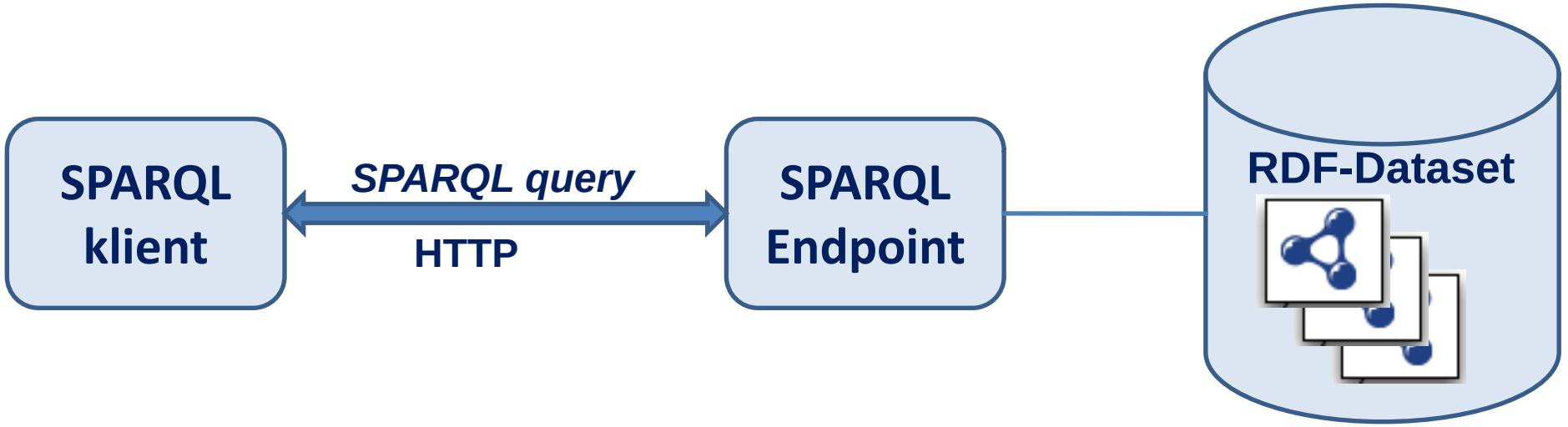
a **dokumentov W3C**

- http://www.w3.org/TR/sparql11-overview/
- http://www.w3.org/TR/rdf-sparql-query/ resp. http://www.w3.org/TR/sparql11-query/

Dalšie zdroje

- http://www.cambridgesemantics.com/semantic-university/learn-sparql

SPARQL



SPARQL je jazyk, ktorý slúži na **dotazovanie** znalostných báz reprezentovaných RDF-grafmi. *(Analogia jazyka SQL)*

SPARQL-endpoint je **servis**, ktorý umožňuje užívateľom/klientským aplikáciám dotazovať prostredníctvom jazyka SPARQL . *(Analógia databázového servera)*

- je **endpoint** – má **URI** na ktorej je prístupný pre klientov
- je to **web-servis** – komunikačný **SPARQL-protokol** je založený na HTTP.
- umožňuje klientom dotazovať informácie uložené v **RDF-Dataset**e pozostavajúcom z jedného alebo viacerých RDF-grafov. *(Analógia relačnej databázy)*

Dátové sady - Datasets

Každý SPARQL dotaz sa vykonáva (vyhľadáva riešenie) nad jednou konkrétnou sadou dát RDF-datasetom.

Koncept **RDF-dataset** je definovaný štandardom
http://www.w3.org/TR/rdf11-datasets/

RDF-dataset obsahuje:

- jeden nepomenovaný **default** RDF-graf**.**
- a ľubovoľného počtu ďalších **pomenovaných** RDF-grafov.
  Pomenované grafy sú identifikované svojím IRI

Pozn. Štandard nešpecifikuje v či a v akom vzťahu sú jednotlivé grafy datasetu. Záleží na kontexte, endpointy môžu implementovať rôzne prístupy. Viac neskôr.

SPARQL Endpoint

V praxi sa stretáme s dvoma typmi endpointov

Generic endpoint umožnuje dotazovať *dataset vytvorený z akékýľvek RDF dokumentov prístupných na internete*. Datovú sadu špecifikuje klient v dotaze a endpoint ju načíta on-the-fly.

- OpenJena SPARQLer http://sparql.org/sparql.html
- OpenLink Virtuoso http://demo.openlinksw.com/sparql

Specific endpoint sprístupňuje jednu konkrétnu dátovú sadu – default /*vlastnú dátovú sadu endpointu*.

- http://dbpedia.org/sparql RDF-databáza wikipédie
- http://europeana.ontotext.com/sparql/queries
- http://vocab.getty.edu/queries#Finding_Subjects

Zoznam SPARQL-endpoints - pozri napr. : http://www.w3.org/wiki/SparqlEndpoints

Fuseki1 server

Download, inštalácia a štart servera podľa Getting Started With Fuseki1:
http://jena.apache.org/documentation/serving_data/index.html#getting-started-with-fuseki

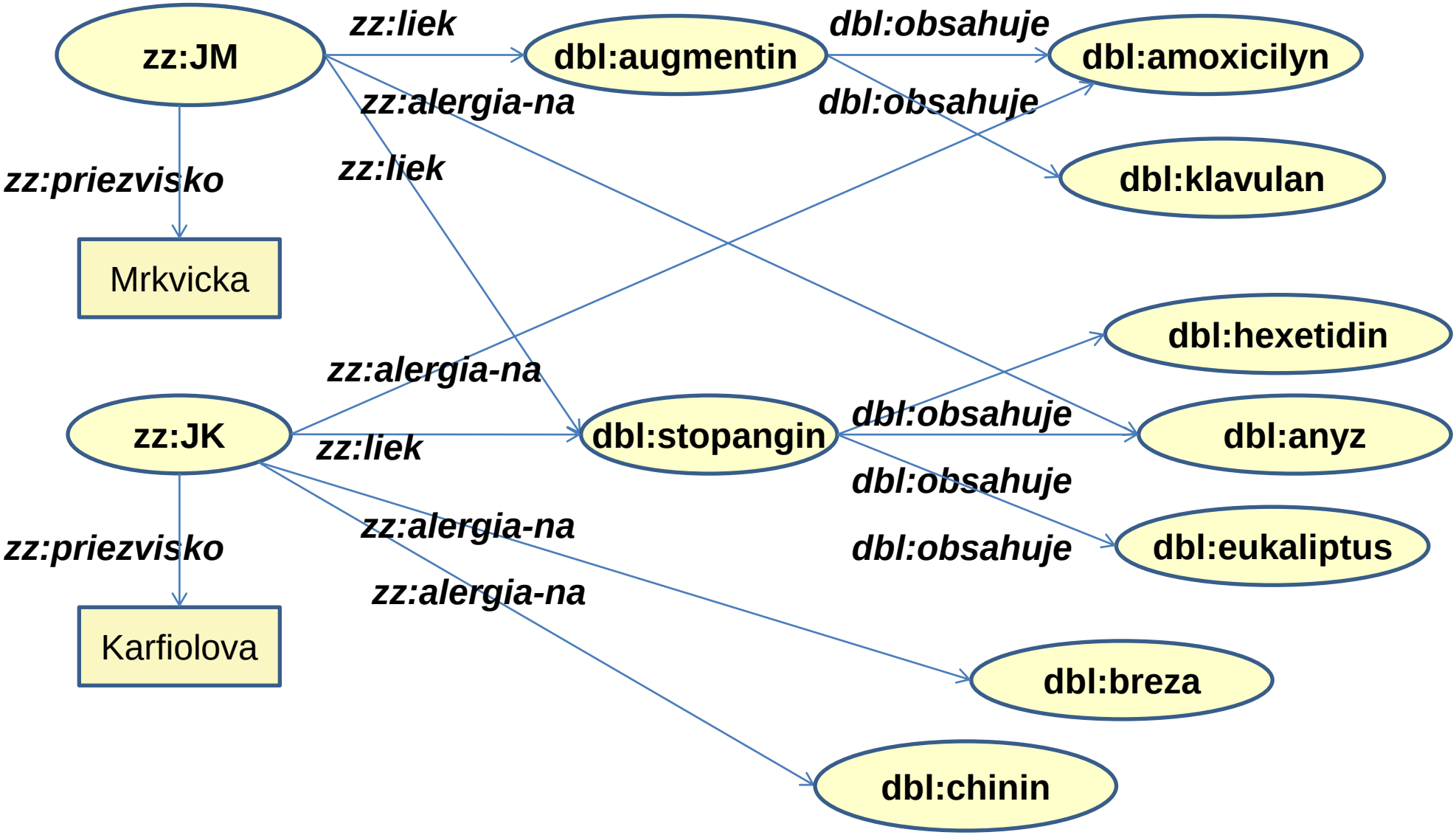
Naštartovanie servera a načítanie RDF-grafu zo súboru

```
cd \opt\Apache\Fuseki1-1.1.2
fuseki-server --update --file=C:\EDU\FEI-RZZ\work\P8data1.ttl /ds
```

Pozn. Namiesto C:\EDU\FEI-RZZ\work\P8data1.ttl zadajte cestu k vašemu RDF-dokumentu v turtle formáte

Web client na URL: http://localhost:3030/

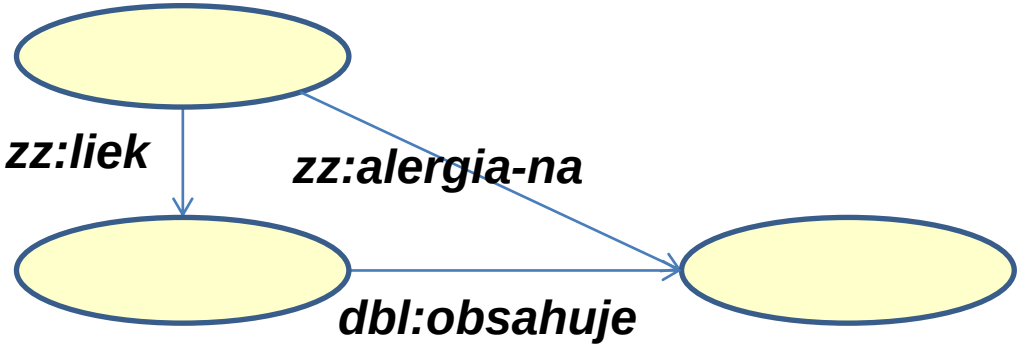
Príklad zo 4. prednášky



nema niektorý pacient predpisany liek, ktorý by mohol spôsobiť alergickú reakciu?

Príklad - hypotéza

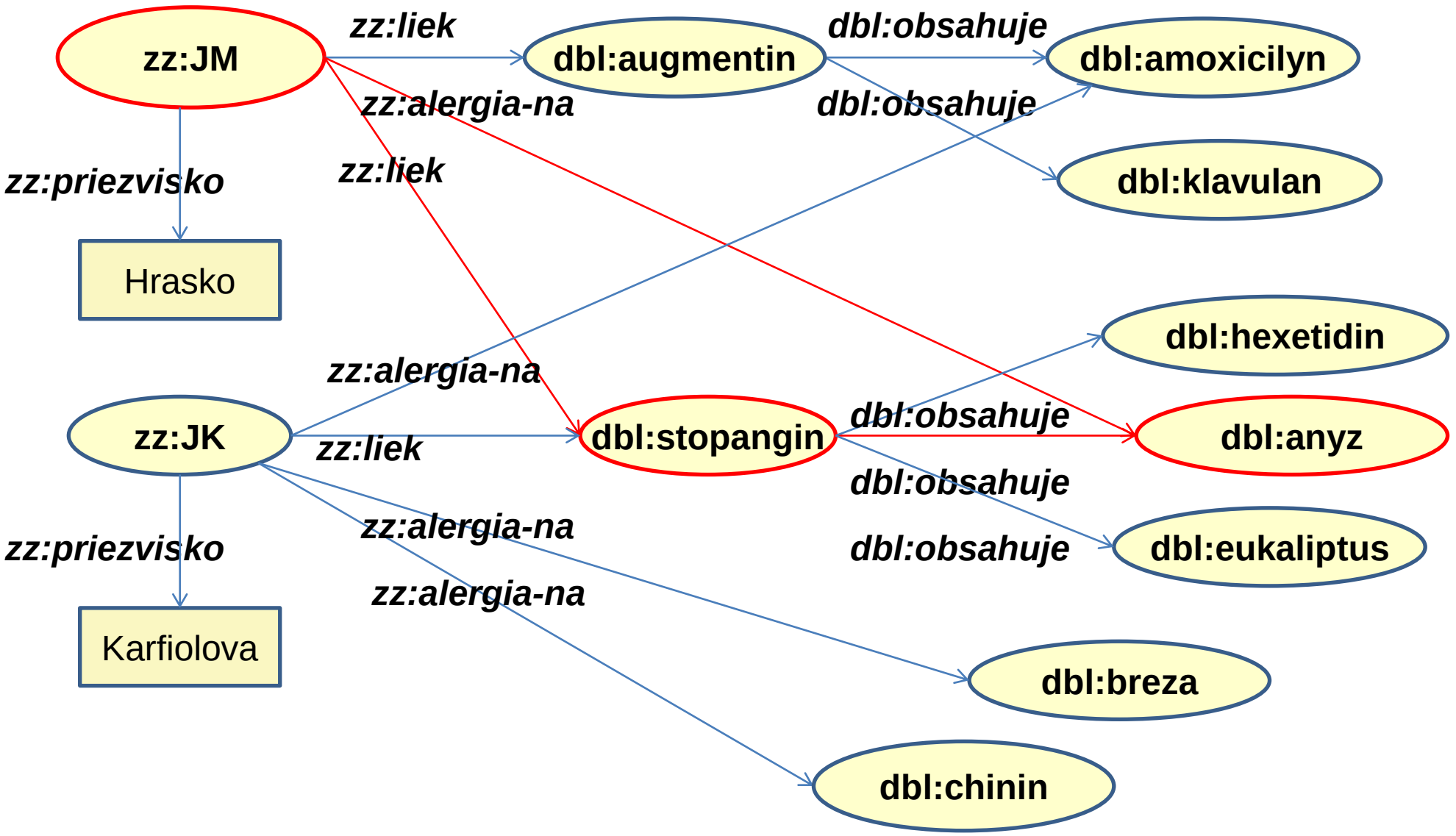
Graf hypotéza H:



Na 4. prednáške sme sa pýtali, len či zo znalostí reprezentovaných grafom G z predchádzajucej strany nevyplýva, že existuje niekto, kto by mohol mať alergickú reakciu na liek ktorý má predpísaný. Teda či hypotéza reprezentovaná grafom H je dôsledkom faktov popísaných grafom G. T.j. Či existuje inšancia H, ktorá je podgrafom/dôsledkom G

Teraz nás však bude zaujímať viac: kto konkrétne bude mať spomínaný problém. T.j. chceme nájsť konkrétne všetky inštancie H ktoré dôsledkom G

Match


zz:JM
zz:liek
dbl:augmentin
dbl:obsahuje
dbl:amoxicilyn
zz:alergia-na
dbl:obsahuje
zz:liek
dbl:klavulan
zz:priezvisko
Hrasko
dbl:hexetidin
zz:alergia-na
zz:JK
dbl:stopangin
dbl:obsahuje
dbl:anyz
zz:liek
dbl:obsahuje
zz:alergia-na
dbl:obsahuje
dbl:eukaliptus
zz:priezvisko
zz:alergia-na
Karfiolova
dbl:breza
dbl:chinin

Štruktúra jednoduchého dotazu

```
PREFIX ex: <http://eg.org/>
SELECT  ?p ?m ?z
WHERE {
   ?p zz:liek ?m .
   ?m  dbl:obsahuje ?z .
   ?p zz:alergia-na ?z .
}
```

• podmienka vo **WHERE** klauzuly sa nazýva **query pattern** a reprezentuje vzor dotazu - šablónu vyhľadávaných dát.

• **jednoduchý vzor** – basic graph pattern **BGP** pozostáva z trojíc

  – trojice môžu obsahovať aj **premenné**

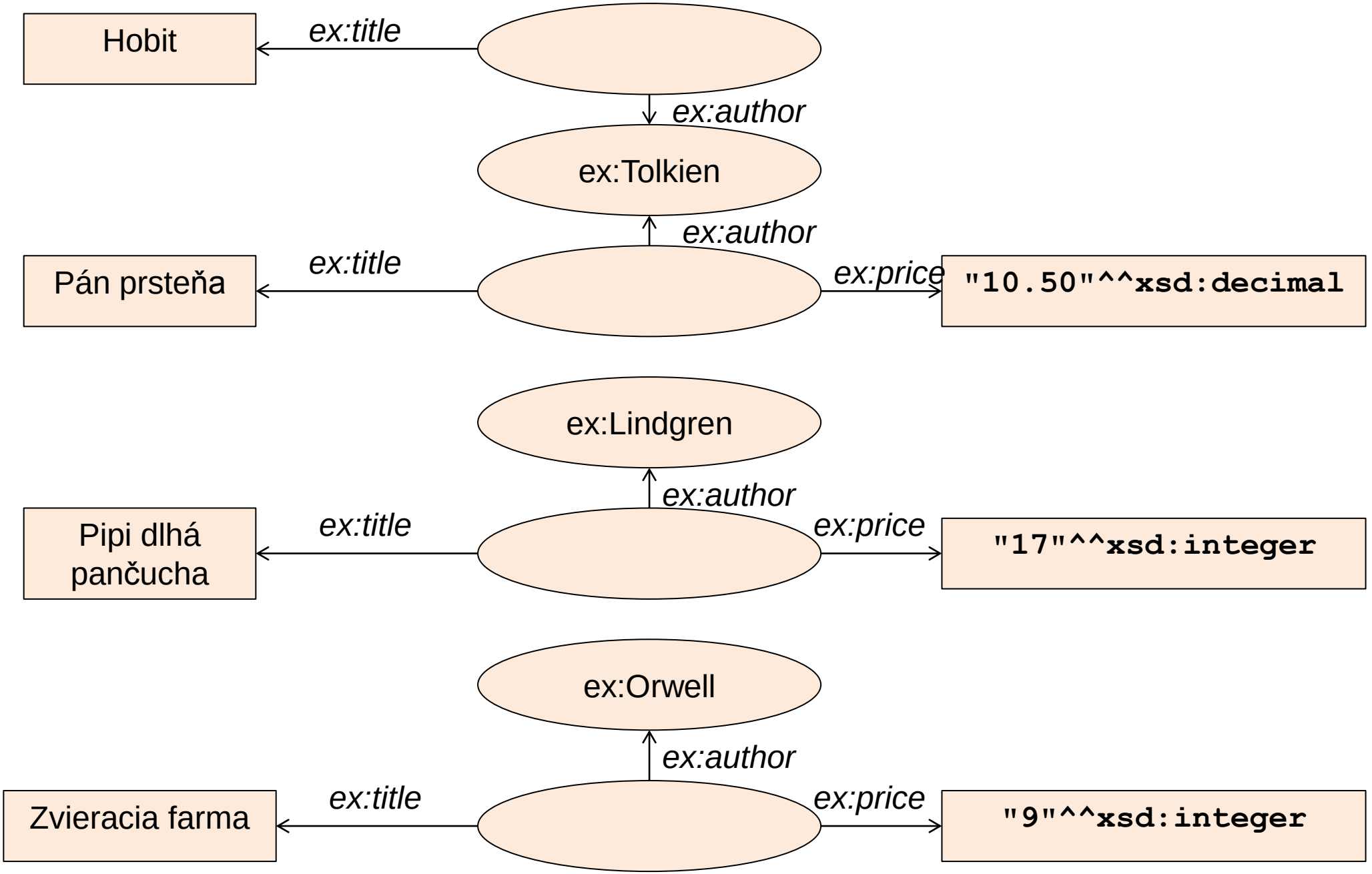
  – na zápis BGP sa používa **turtle** syntax

• klauzula **SELECT** popisuje výstup - t.j. premenné, ktoré majú zahrnúť do výsledku

• klauzulu **PREFIX** možno použiť na deklaráciu **IRI-prefix**ov.

Pozn. Komplexné dotazy môžu mať zložitejšiu štruktúru a obsahovať aj ďalšie klauzuly

Príklad – RDFgraf



Príklad – RDFgraf

```
@prefix ex: <http://eg.org/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
_:a   ex:author   ex:Tolkien ;
      ex:title    "Hobit" .
_:b   ex:author   ex:Tolkien ;
      ex:title    "Pán prsteňa" ;
      ex:price    "10.50"^^xsd:decimal .
_:c   ex:author   ex:Lindgren ;
      ex:title    "Pipi dlhá pančucha" ;
      ex:price    "17"^^xsd:integer .
_:d   ex:author   ex:Orwell ;
      ex:title    "Zvieracia farma" ;
      ex:price    "9"^^xsd:integer .
```

Príklady jednoduchých dotazov

Vyhladať všetkých autorov

```
PREFIX ex: <http://eg.org/>
SELECT ?o
WHERE {?s ex:author ?o . }
```

Vyhladať všetky trojice

```
SELECT ?s ?p ?o
WHERE {?s ?p ?o . }
```

Vyhladať všetky názvy a ceny Tolkienových kníh

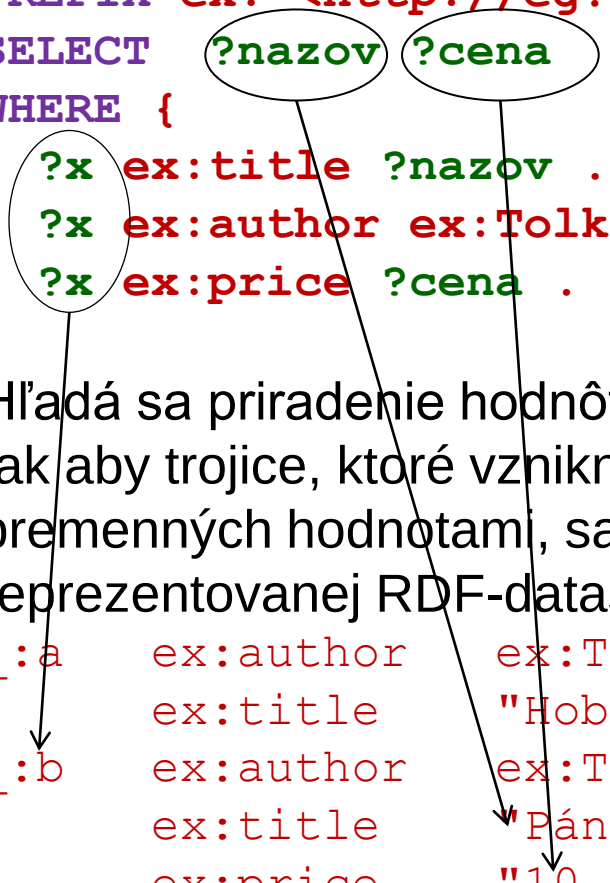
```
PREFIX ex: <http://eg.org/>
SELECT  ?nazov ?cena
WHERE {
  ?x ex:title ?nazov .
  ?x ex:author ex:Tolkien .
  ?x ex:price ?cena .
}
```

Výsledok

```
-------------------------
| nazov          | cena  |
=========================
| "Pán prsteňa" | 10.50 |
-------------------------
```

Intuitívna interpretácia jednoduchého dotazu

```
PREFIX ex: <http://eg.org/>
SELECT  ?nazov ?cena
WHERE {
  ?x ex:title ?nazov .
  ?x ex:author ex:Tolkien .
  ?x ex:price ?cena .
}
```



Hľadá sa priradenie hodnôt (URI, prázdnych uzlov alebo literálu) premenným, tak aby trojice, ktoré vzniknú z WHERE klauzule (BGP) nahradením premenných hodnotami, sa nachádzali v dotazovanej znalostnej báze reprezentovanej RDF-datasetom.

```
_:a   ex:author   ex:Tolkien ;
      ex:title    "Hobit" .
_:b   ex:author   ex:Tolkien ;
      ex:title    "Pán prsteňa" ;
      ex:price    "10.50"^^xsd:decimal .
_:c   ex:author   ex:Lindgren ;
      ex:title    "Pipi dlhá pančucha" ;
      ex:price    "17"^^xsd:integer .
_:d   ex:author   ex:Orwell ;
      ex:title    "Zvieracia farma" ;
      ex:price    "9"^^xsd:integer .
```

Zátvorky - grupovanie trojíc v BGP

```
PREFIX ex: <http://eg.org/>
SELECT  ?nazov ?cena
WHERE {
  ?x ex:title ?nazov .
  ?x ex:author ex:Tolkien .
  ?x ex:price ?cena .
}
```

Riešením dotazu je každý podgraf spĺňajúci súčasne všetky trojice v BGP. Konjunkcia výrokov je asociatívna a komutatívna. Nezáleží teda na ozátvorkovaní ani poradí trojíc v BGP

```
T and A and P = (T and A) and P = P and (T and A)
```

Rovnakú odpoveď dostaneme aj pre:

```
WHERE {
  ?x ex:price ?cena .
  { ?x ex:title ?nazov .
    ?x ex:author ex:Tolkien . }
}
```

Závtvorky majú zmysel len v kombinácii s ďalšími operáciami

UNION - alternatívne BGP

```
PREFIX ex: <http://eg.org/>
SELECT  ?nazov
WHERE {
{ ?x ex:title ?nazov .
  ?x ex:author ex:Tolkien . }
UNION
{ ?x ex:title ?nazov .
  ?x ex:author ex:Lindgren . }
}
```

(T and AT) or (T and AL) = T and (AT or AL)

Preto je to isté ako

```
WHERE {
?x ex:title ?nazov .
{ ?x ex:author ex:Tolkien . }
UNION
{ ?x ex:author ex:Lindgren . }
}
```

!Pozor na nezvyklé zátvorkovanie:

- narozdiel od matematiky má **UNION vyššiu prioritu**
- ale jednotlivé alternatívy musia byť uzavreté **{}**

OPTIONAL - nepovinné BGP

```
PREFIX ex: <http://eg.org/>
SELECT  ?nazov ?cena
WHERE {
  ?x ex:title ?nazov .
  ?x ex:author ex:Tolkien .
  ?x ex:price ?cena .
}
```

nevypísal Hobita, pretože nemá danú cenu.

Riešenie informácia o cene je nepovinná - OPTIONAL

```
WHERE {
  ?x ex:title ?nazov .
  ?x ex:author ex:Tolkien .
  OPTIONAL {?x ex:price ?cena .}
}
```

Intuitívna interpretácia

(T and A) or (T and A and P)

!Pozor na zátvorkovanie:

- nepovinné BGP musia byť uzavreté v {}

Kombinácie OPTIONAL a UNION

```
PREFIX ex: <http://eg.org/>
SELECT ?nazov ?cena
WHERE {
  ?x ex:title ?nazov .
  {?x ex:author ex:Tolkien} UNION {?x ex:author ex:Lindgren}
  OPTIONAL {?x ex:price ?cena .}
}
```

je to iste ako

```
WHERE {
  ?x ex:title ?nazov .
  { {?x ex:author ex:Tolkien} UNION {?x ex:author ex:Lindgren} }
  OPTIONAL {?x ex:price ?cena .}
}
```

sa líši sa od

```
WHERE {
  ?x ex:title ?nazov .
  {?x ex:author ex:Tolkien .} UNION
  { {?x ex:author ex:Lindgren .} OPTIONAL {?x ex:price ?cena .} }
}
```

OPTIONAL a UNION majú rovnakú prioritu a su left-asociative

Filtre

Filter predstavuje ďalšiu podmienku, ktorú musia spĺňať hodnoty priradené premennným

```
PREFIX ex: <http://eg.org/>
SELECT ?nazov ?cena
WHERE {
  ?x ex:title ?nazov .
  ?x ex:price ?cena .
  FILTER (?cena < 15.0)
}
```

Podmienka je boolovský výraz, ktorý môže byť vytvorený pomocou

- identifikátorov premenných, konštánt,...
- operátorov porovnávania
- aritmetických operácií
- funkcií – využívať sa môže celý rad funkcií známych z XML-štandardov (XQuery, XPath...) ako j funkcie definované RDF štandardom

Poznámka k dátovým typom

Datatypes

```
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix ex:  <http://example.org/> .
ex:ex1  ex:p  "test" .
ex:ex2  ex:p  "test"^^xsd:string .
ex:ex3  ex:p  "test"@en .
ex:ex4  ex:p  "42"^^xsd:integer .
```

Which matches does the following BGP have?

```
{   ?subject  <http://example.org/p>  "test" . }
```

⇝ `ex:ex1` is the only result

⇝ Exact match for the datatypes is required

But: Abbreviations for numerical values allowed

```
{   ?subject  <http://example.org/p>  42 . }
```

⇝ The datatype is determined from the syntactic form

`xsd:integer` (42), `xsd:decimal` (42.2), `xsd:double` (1.0e6)

Triedenie množiny výsledkov

Sorting Results

Sorting is achieved with the keyword `ORDER BY`

```
SELECT ?book, ?price
WHERE { ?book  <http://example.org/Price>  ?price . }
ORDER BY ?price
```

- Sorting as with comparison operators in filters
- Alphabetical sorting of URIs as strings
- Order between elements of different types:
  unbound variables < blank nodes < URIs < RDF literals
- Not all possibilities defined by the specification

Further possible options:

- `ORDER BY DESC(?price)`: descending
- `ORDER BY ASC(?price)`: ascending (default)
- `ORDER BY DESC(?price), ?titel`: hierarchical ordering criteria

Obmedzenie veľkosti množiny výsledkov

LIMIT, OFFSET and DISTINCT

Limit the set of results:

- `LIMIT`: Maximal number of results
- `OFFSET`: Position of the first returned result
- `SELECT DISTINCT`: Removal of duplicate results

```
SELECT DISTINCT ?book, ?price
WHERE { ?book  <http://ex.org/price>  ?price . }
ORDER BY ?price  LIMIT 5  OFFSET 25
```

⇝ `LIMIT` and `OFFSET` only meaningful with `ORDER BY`!

Príklad specific endpoint – dbpedia

Pomocou sparql web-klienta Dbpedie

http://dbpedia.org/sparql

zistíme, **ktoré autá sa vyrábajú na Slovensku.**

```
PREFIX dbpedia: <http://dbpedia.org/resource/>
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>
SELECT ?znacka
WHERE
{?znacka dbpedia-owl:assembly dbpedia:Slovakia}
LIMIT 20
```

Domáca úloha:

Napíšte query , ktorá zistí,

- v ktorých krajinách montujú Audi Q7
- hlavné mestá európskych krajín
- hudobných skladateľov, ktorý sa narodili v Bratilave