Graduate Program in Production Engineering and Systems

(PPGEPS)

Important Terms in an Ontology (Part 6/6)

Dr. Yongxin Liao

Course Outlines

- Important Terms in an Ontology (6/6)
  - SPARQL Query
    - What is SPARQL?
      - The RDF Triple Representation
    - The SPARQL Query Structure
      - Prefix Declaration
      - Result Forms
      - Query Patterns
      - Solution Modifiers
      - Property Path
  - Protégé Practices

PUCPR

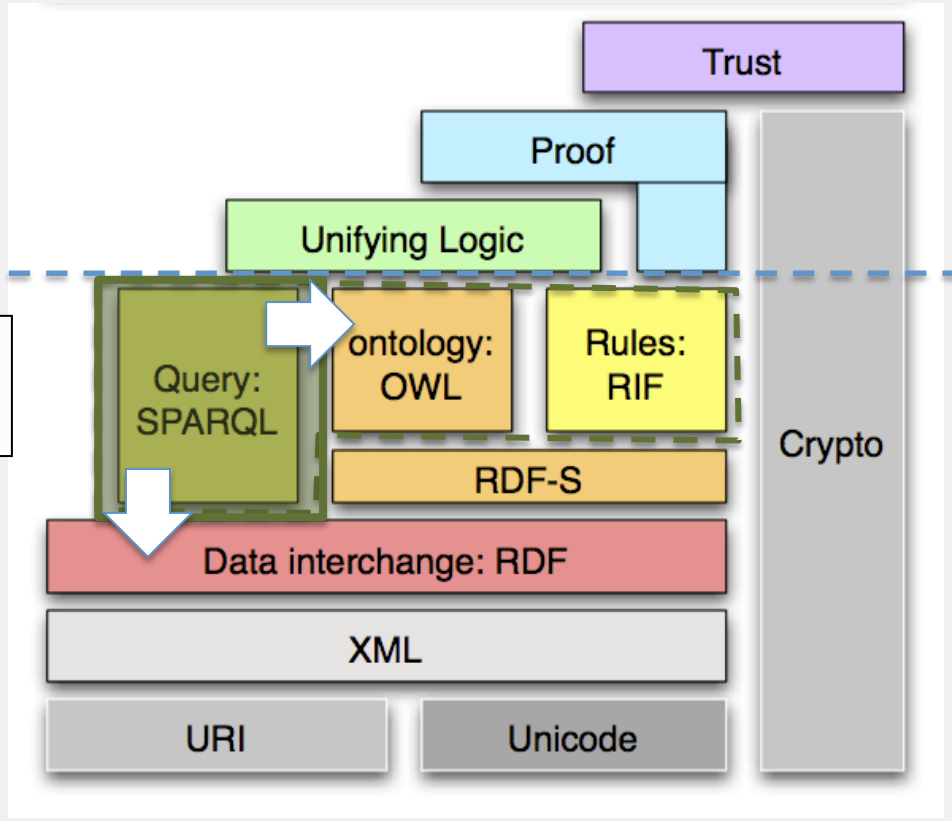
Important Terms in an Ontology

- Several Important Terms
  - Axioms
  - Concepts (Individuals and Classes)
  - Relationships (Class Assertions, Subclasses Disjoint/Equivalent Classes, Individual Equality/Inequality Properties, Property Assertions, Property Characteristics, and Property Descriptions)
  - Complex Class Expressions (Enumeration of Individuals, Propositional Connectives, Object Property Restrictions, Necessary and Sufficient Conditions, Data Property Restrictions)
  - Data Ranges (Data Types and Data Type Restrictions)
  - Reasoning Rules
  - Knowledge Base (T-box and A-box)
  - **SPARQL Query**

Part 1/6
Part 2/6
Part 3/6
Part 4/6
Part 5/6
Part 6/6

PUCPR

SPARQL Query Language



protégé

PUCPR

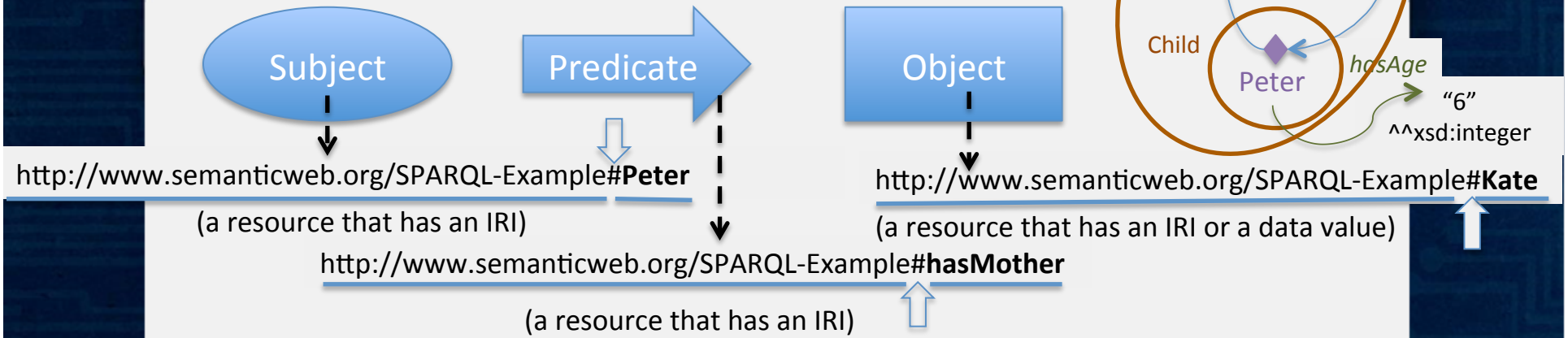
What is SPARQL?

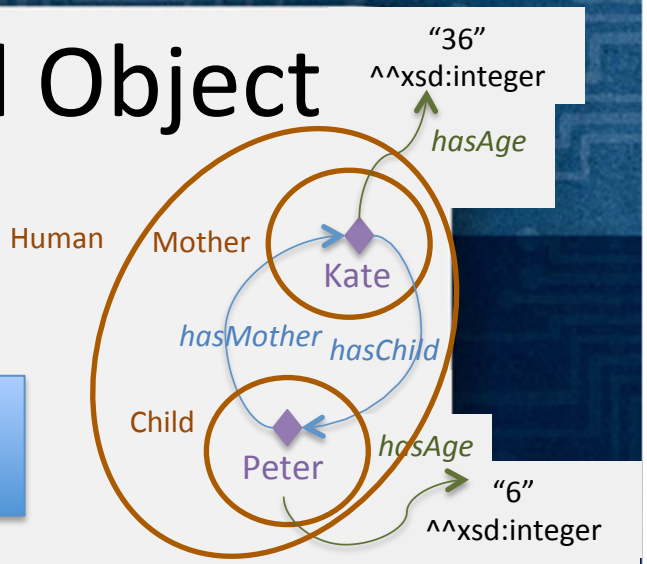
- A Query Language.
  - It is originally designed for RDF query.
  - It query what exists in ontologies
  - It allows us:
    - Query information from an ontology based on existing concepts and relationships.
    - Perform complex joins of different ontologies in a single, simple query.

PUCPR

Subject, Predicate, and Object

- The RDF Triple Representation





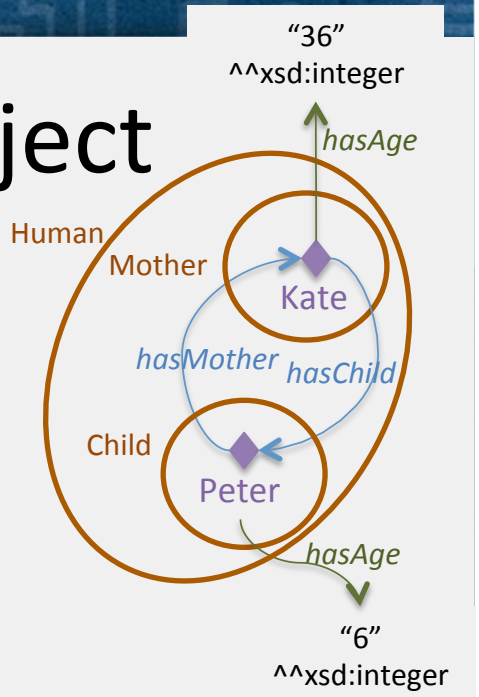
– They can be abbreviated as "prefix + : + name"

PREFIX O1: <http://www.semanticweb.org/SPARQL-Example#>.

A Triple	O1:Peter	O1:hasMother	O1:Kate.
	(Subject)	(Predicate)	(Object)

PUCPR

Subject, Predicate, and Object



- Class Declaration

O1:**Human**	rdf:**type**	owl:**Class**.
O1:**Mother**	rdf:**type**	owl:**Class**.
O1:**Child**	rdf:**type**	owl:**Class.**

- Subclass Axioms

O1:**Mother**	rdfs:**subClassOf**	O1:**Human**.
O1:**Child**	rdfs:**subClassOf**	O1:**Human**.

- Property Declaration

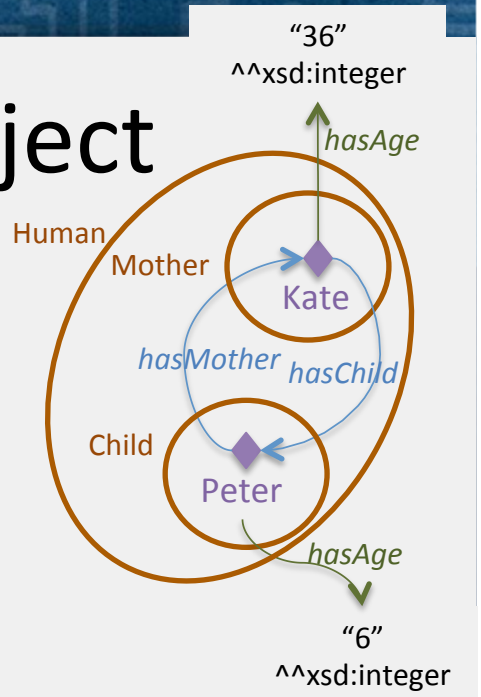
O1:**hasMother**	rdf:**type**	owl:**ObjectProperty**.
O1:**hasChild**	rdf:**type**	owl:**ObjectProperty**.
O1:**hasAge**	rdf:**type**	owl:**DatatypeProperty**.

- Individual Declaration

O1:**Peter**	rdf:**type**	owl:**NamedIndividual**.
O1:**Kate**	rdf:**type**	owl:**NamedIndividual**.

PREFIX O1: <http://www.semanticweb.org/SPARQL-Example#>.
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
PREFIX owl: <http://www.w3.org/2002/07/owl#>.

Subject, Predicate, and Object



- Class Assertion Axioms

  O1:**Peter** rdf:**type** O1:**Child**.

  O1:**Kate** rdf:**type** O1:**Mother**.

- Object Property Assertion Axioms

  O1:**Peter** O1:**hasMother** O1:**Kate**.

  O1:**Kate** O1:**hasChild** O1:**Peter**.

- Data Property Assertion Axioms

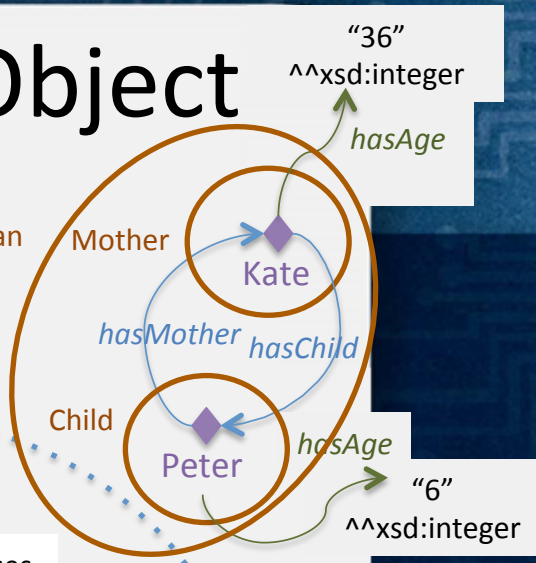
  O1:**Peter** O1:**hasAge** "6"^^xsd:integer.

  O1:**Kate** O1:**hasAge** "36"^^xsd:integer.

PREFIX O1: <http://www.semanticweb.org/SPARQL-Example#>.
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
PREFIX owl: <http://www.w3.org/2002/07/owl#>.

Subject, Predicate, and Object

- Each Triple Describes a Fact



Subject	Predicate	Object	
O1:Human	rdf:type	owl:Class.	rdf built-in properties
O1:Mother	rdf:type	owl:Class.	
O1:Child	rdf:type	owl:Class.	
O1:Mother	rdfs:subClassOf	O1:Human.	Classes
O1:Child	rdfs:subClassOf	O1:Human.	
O1:hasMother	rdf:type	owl:ObjectProperty.	rdfs built-in properties
O1:hasChild	rdf:type	owl:ObjectProperty.	
O1:hasAge	rdf:type	owl:DatatypeProperty.	owl built-in classes
O1:Peter	rdf:type	owl:NamedIndividual.	
O1:Peter	rdf:type	O1:Child.	Classes
O1:Kate	rdf:type	owl:NamedIndividual.	owl built-in classes
O1:Kate	rdf:type	O1:Mother.	Classes
O1:Peter	O1:hasMother	O1:Kate.	Individuals
O1:Kate	O1:hasChild	O1:Peter	
O1:Peter	O1:hasAge	"6"^^xsd:integer.	Data Values
O1:Kate	O1:hasAge	"36"^^xsd:integer.	

Classes: O1:Human, O1:Mother, O1:Child

Object Properties: O1:hasMother, O1:hasChild

Data Properties: O1:hasAge

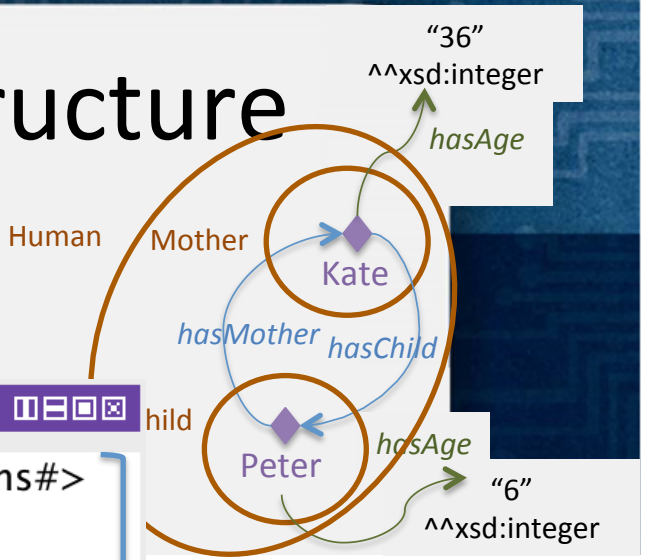
Individuals: O1:Peter, O1:Kate

```
PREFIX  O1:   <http://www.semanticweb.org/SPARQL-Example#>.
PREFIX  rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
PREFIX  rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
PREFIX  owl:  <http://www.w3.org/2002/07/owl#>.
```

PUCPR

The SPARQL Query Structure

- An Example in Protégé



SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX O1: <http://www.semanticweb.org/SPARQL-Example#>

SELECT   ?X  ?Y
WHERE { ?X rdf:type O1:Human.
        ?X O1:hasAge  ?Y.}
ORDER BY DESC(?Y)
```

The Prefix declarations

The Result Form

The Query Pattern

The Solution Modifier

X	Y
Kate	"36"^^<http://www.w3.org/2001/XMLSchema#integer>
Peter	"6"^^<http://www.w3.org/2001/XMLSchema#integer>

Solution

PUCPR

The SPARQL Query Structure

- The Prefix declarations
  - Default Prefix Declarations
    - Generated by Protégé OWL Editor Automatically

      E.g. **PREFIX** rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

      PREFIX owl: <http://www.w3.org/2002/07/owl#>

      PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

      PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

  - Ontology Prefix Declarations
    - Created Manually
    - The ontology IRI can be found in "Active Ontology" Tab

E.g. **PREFIX** PO: <http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

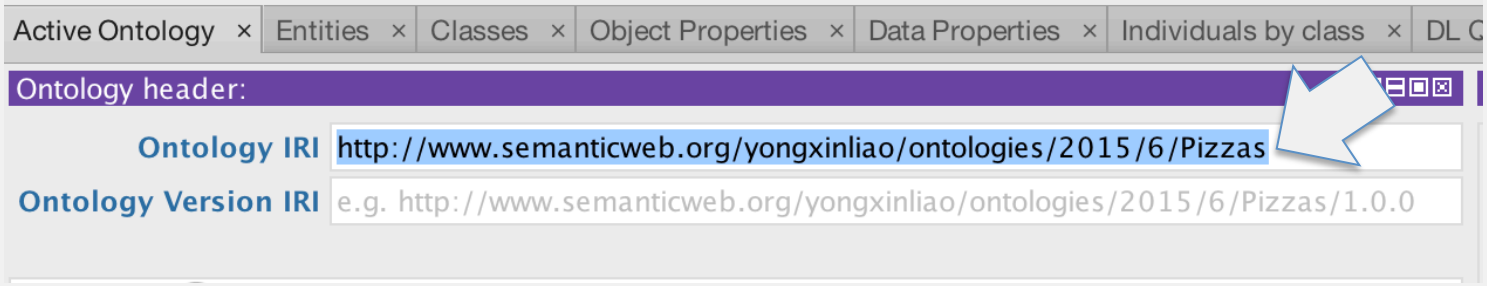
Abbreviation of Your Ontology

Your Ontology IRI

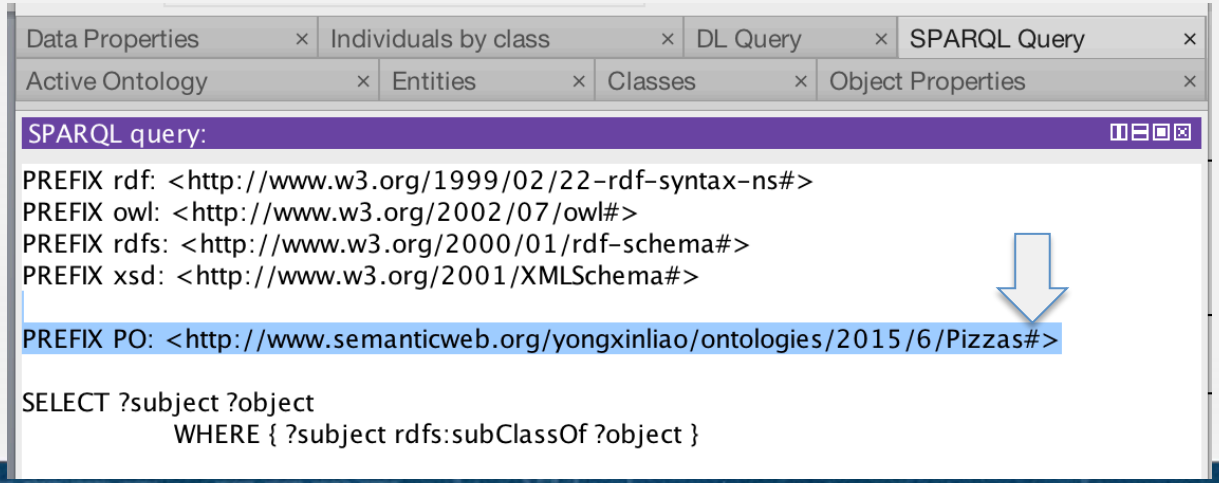
PUCPR

Protégé Practice

- Ontology Prefix Declaration
  - **Copy** Your Ontology IRI from "Active Ontology" Tab



  - Create Your Prefix in "SPARQL Query" Tab



PUCPR

The SPARQL Query Structure

- The Result Forms (1/2)
  - SELECT
    - Syntax: **SELECT** ?variable$_1$ … ?variable$_n$

Starts with a "?" symbol

      - Each variable is separated by **an empty space**.
      - E.g. SELECT ?X ?Y ?Z
    - The SELECT returns a table of variables and values that satisfy the query patterns
    - The query patterns contain all (or some) of those variables.
- The Query Patterns (1/5)
    - Syntax: **WHERE** { Triple (s)}
      - Any parts of a triple can be replaced with a variable
      - Each Triple is ended by the "." symbol
      - E.g. WHERE { ?X rdf:type PO:Pizza.}

PUCPR

Query #1 SELECT (single variable)

- Find all the pizzas
  - Query all the individuals of the class Pizza
  - Query all the subjects (?X) that are connected to the object (PO:Pizza) by the predicate (rdf:type)

Answer 1 for Query #1:

SELECT ?X
WHERE { ?X rdf:type PO:Pizza. }

PUCPR

Query #1 SELECT (single variable)

SPARQL query:

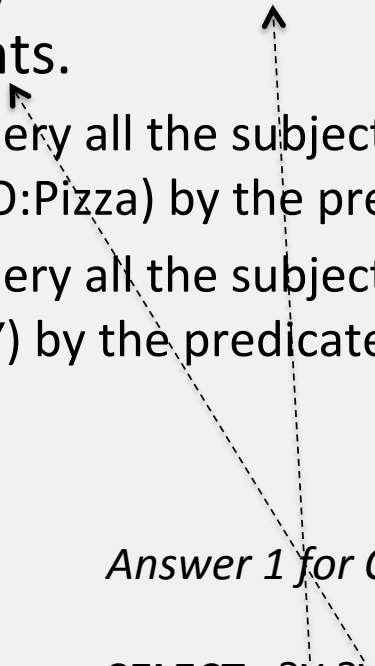
```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO:<http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

SELECT ?X
WHERE { ?X   rdf:type    PO:Pizza. }
```

X
PizzaID3
PizzaID6
ExamplePizzaX
PizzaID2
PizzaID5
PizzaID1
PizzaID4
PizzaID7

PUCPR

Query #2 SELECT (Multiple Variables)

- Find all the pizzas and their weights
  - Query all the individuals of the class Pizza and their weights.
    - Query all the subjects (?X) that are connected to the object (PO:Pizza) by the predicate (rdf:type).
    - Query all the subjects (?X) that are connected to the object (?Y) by the predicate (PO:hasWeightInGrams)



Answer 1 for Query #2 (1/2):

```
SELECT   ?X ?Y
WHERE {   ?X rdf:type PO:Pizza.
          ?X PO:hasWeightInGrams ?Y. }
```

PUCPR

Query #2 SELECT (Multiple Variables)

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO:<http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

SELECT   ?X ?Y
WHERE {   ?X rdf:type PO:Pizza.
              ?X PO:hasWeightInGrams ?Y. }
```

X	Y
PizzaID6	"370"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID2	"375"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID3	"565"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID5	"560"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID4	"175"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID1	"470"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID7	"380"^^<http://www.w3.org/2001/XMLSchema#decimal>

PUCPR

The SPARQL Query Structure

- The Solution Modifier (1/3)
  - ORDER BY
    - Syntax: **ORDER BY** ASC(?variable)/DESC(?variable)
      - It rearranges the query result based on the value of one or more variables.
      - ASC( ) => Query result is in ascending order
      - DESC( ) => Query result is in descending order
      - Multiple ASC( ) and DESC ( ) can appear in the expression, which are separated by ***empty spaces***
      - E.g. ORDER BY ASC(?X)

        ORDER BY ASC(?X) ASC(?Y) DESC(?Z)

PUCPR

Query #3 ORDER BY (1/2)

- Find all the pizzas, their weights
- Arrange the results by the weights in ascending order

Answer 1 for Query #3(1/2):

SELECT ?X ?Y
WHERE { ?X rdf:type PO:Pizza.
?X PO:hasWeightInGrams ?Y. }
ORDER BY **ASC(?Y)**

=

ORDER BY **(?Y)**

Query #3 ORDER BY (1/2)

SPARQL query:

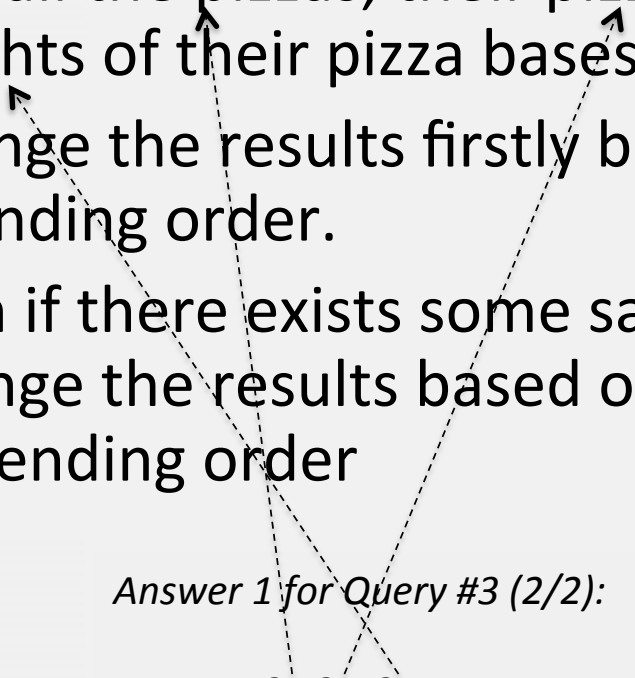
```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO: <http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizza

SELECT   ?X ?Y
WHERE {   ?X a PO:Pizza.
          ?X PO:hasWeightInGrams ?Y}
ORDER BY ASC(?Y)
```

X	Y
PizzaID4	"175"^^<http://www.w3.org/2001/XMLSchema#
PizzaID6	"370"^^<http://www.w3.org/2001/XMLSchema#
PizzaID2	"375"^^<http://www.w3.org/2001/XMLSchema#
PizzaID7	"380"^^<http://www.w3.org/2001/XMLSchema#
PizzaID1	"470"^^<http://www.w3.org/2001/XMLSchema#
PizzaID5	"560"^^<http://www.w3.org/2001/XMLSchema#
PizzaID3	"565"^^<http://www.w3.org/2001/XMLSchema#

PUCPR

Query #3 ORDER BY (2/2)

- Find all the pizzas, their pizza bases, and the weights of their pizza bases
- Arrange the results firstly based on the weights in ascending order.
- Then if there exists some same weight pizza bases, arrange the results based on the pizza names in descending order



Answer 1 for Query #3 (2/2):

```
SELECT   ?X ?Y  ?Z
WHERE {   ?X   rdf:type                  PO:Pizza.
          ?X   PO:hasBase                ?Y.
          ?Y   PO:hasWeightInGrams  ?Z.        }
ORDER BY ASC(?Z) DESC(?X)
```

PUCPR

Query #3 ORDER BY (2/2)

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO: <http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

SELECT   ?X ?Y  ?Z
WHERE {   ?X    rdf:type                PO:Pizza.
          ?X    PO:hasBase              ?Y.
          ?Y    PO:hasWeightInGrams  ?Z.}
ORDER BY ASC(?Z) DESC(?X)
```

X	Y	Z
PizzaID4	PizzaBaseNo4-1	"85"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID2	PizzaBaseNo2-1	"150"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID6	PizzaBaseNo6-1	"175"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID7	PizzaBaseNo7-1	"180"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID3	PizzaBaseNo3-1	"180"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID5	PizzaBaseNo5-1	"250"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID1	PizzaBaseNo1-1	"310"^^<http://www.w3.org/2001/XMLSchema#decimal>

PUCPR

The SPARQL Query Structure

- The Solution Modifie
  - LIMIT
    - Syntax: **LIMIT** n
      - It limits the numb
      - It returns the rov
      - It means that the discarded.
      - E.g. LIMIT 15

A p

1000 Church Street	CSI250	8/1/2008
A V I Charleston - MR	800052-259	10/1/2000
A V I East Liverpool - MT	800052-068	10/1/2000
A V I Martins Ferry - MR	800052-284	10/1/2000
A V I Martins Ferry - MT	800052-084	10/1/2000
A V I Office - MR	800052-137	5/1/2004
A V I Parkersburg - MR	800052-282	10/1/2000
A V I Parkersburg - MT	800052-082	10/1/2000
A V I South Charleston - MR	UNKNOWN	1/1/2004
A V I WV State University - MR	800052-134	4/1/2004
A V I Washington - MR	800052-177	8/1/2008
ABB Automation Analytical Division	3013	9/1/2001
AMFM Inc/Commercial Holding	5923	1/1/2007
Aaron M Magner	8J9000898	6/1/2007
Aaron M Morris	58788368	11/1/2006
Aaron P Aman	UNKNOWN	9/1/2009
Abbotts Wrecker Service	994340-001-002	12/1/2006
Abdalla Z Bandak	60320561	12/1/2008
Abdollah Hadavand	60114844	1/14/2008
Access and Mobility Enterprise	758052-001-002	8/1/2004
Adam Black	60221547	7/11/2008
Adam D Stephens	40121412	1/17/2008
Administrative Resource Management	UNKNOWN	6/1/2011
Adrian L Donatelli	40117657	11/8/2007
Advocate LLC - MT	UNKNOWN	6/1/2010
Aetna Building Maintance Inc	UNKNOWN	3/1/2010
Ahmad A Barghouthi	58912329	3/26/2007
Aiden T Cumpston	60232679	8/1/2008
Ajay T Patel	58774447	11/1/2006
Akron Porcelain & Plastics Co - MT	595144-001-002	9/1/2004
Alan M Clay	60292135	11/12/2008
Albert Motors Inc	UNKNOWN	4/1/2010
Alen Jackson	UNKNOWN	3/27/2010
Alexander Otellin, MD	145937	5/1/2008
Alexandria D Combs	3569383	8/1/2010
Alfab Inc	ALF04193	8/1/1999
Alfred Kooken	UNKNOWN	3/25/2010
Ali Co	03038E	9/1/2009

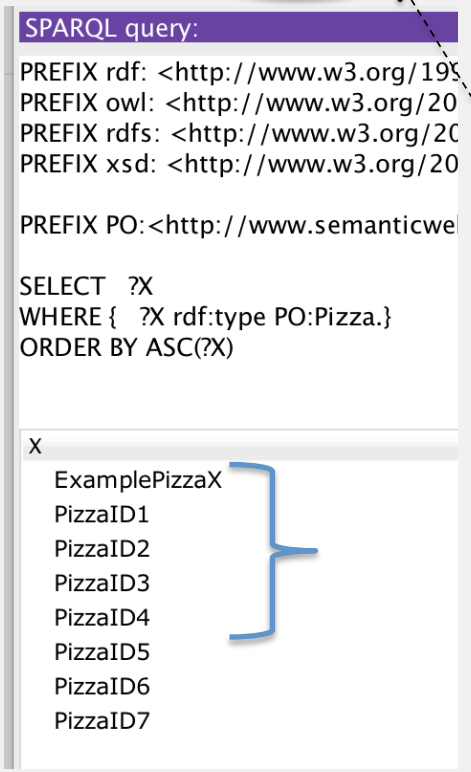
From line 1 to line 15 will be returned

From line 16 to the end (line 38) will be discarded

PUCPR

QUERY #4 LIMIT

- Find all the pizzas
- Arrange the result by the their names in ascending order
- Return the first 5 pizzas



Answer 1 for Query #4:

```
SELECT   ?X
WHERE {   ?X rdf:type PO:Pizza.}
ORDER BY ASC(?X)
LIMIT 5
```

PUCPR

QUERY #4 LIMIT

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO:<http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

SELECT   ?X
WHERE {   ?X rdf:type PO:Pizza.}
ORDER BY ASC(?X)
LIMIT 5
```

X
ExamplePizzaX
PizzaID1
PizzaID2
PizzaID3
PizzaID4

PUCPR

The SPARQL Query Structure

- The Solution Modifie
  - OFFSET A p
    - Syntax: **OFFSET** n
      - It limits the num
      - It returns the rov
      - It means that the
      - Usually it is used extract a subset
      - E.g. LIMIT 15
        OFFSET 15
      - Usually it is for p

1000 Church Street	CSI250	8/1/2008
A V I Charleston - MR	800052-259	10/1/2000
A V I East Liverpool - MT	800052-068	10/1/2000
A V I Martins Ferry - MR	800052-284	10/1/2000
A V I Martins Ferry - MT	800052-084	10/1/2000
A V I Office - MR	800052-137	5/1/2004
A V I Parkersburg - MR	800052-282	10/1/2000
A V I Parkersburg - MT	800052-082	10/1/2000
A V I South Charleston - MR	UNKNOWN	1/1/2004
A V I WV State University - MR	800052-134	4/1/2004
A V I Washington - MR	800052-177	8/1/2008
ABB Automation Analytical Division	3013	9/1/2001
AMFM Inc/Commercial Holding	5923	1/1/2007
Aaron M Magner	8J9000898	6/1/2007
Aaron M Morris	58788368	11/1/2006
Aaron P Aman	UNKNOWN	9/1/2009
Abbotts Wrecker Service	994340-001-002	12/1/2006
Abdalla Z Bandak	60320561	12/1/2008
Abdollah Hadavand	60114844	1/14/2008
Access and Mobility Enterprise	758052-001-002	8/1/2004
Adam Black	60221547	7/11/2008
Adam D Stephens	40121412	1/17/2008
Administrative Resource Management	UNKNOWN	6/1/2011
Adrian L Donatelli	40117657	11/8/2007
Advocate LLC - MT	UNKNOWN	6/1/2010
Aetna Building Maintance Inc	UNKNOWN	3/1/2010
Ahmad A Barghouthi	58912329	3/26/2007
Aiden T Cumpston	60232679	8/1/2008
Ajay T Patel	58774447	11/1/2006
Akron Porcelain & Plastics Co - MT	595144-001-002	9/1/2004
Alan M Clay	60292135	11/12/2008
Albert Motors Inc	UNKNOWN	4/1/2010
Alen Jackson	UNKNOWN	3/27/2010
Alexander Otellin, MD	145937	5/1/2008
Alexandria D Combs	3569383	8/1/2010
Alfab Inc	ALF04193	8/1/1999
Alfred Kooken	UNKNOWN	3/25/2010
Ali Co	03038E	9/1/2009

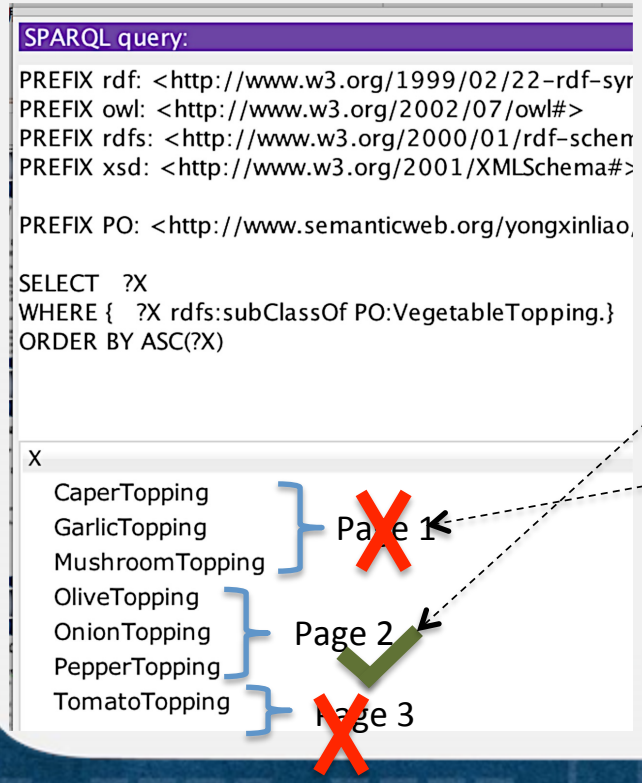
From line 1 to line 15 will be discarded

From line 16 to line 30 will be returned

From line 31 to the end line 38 will be discarded

QUERY #5 OFFSET

- Find all kinds of the vegetable toppings
- Arrange the results by the their names in ascending order
- 3 kinds per page, return the second page



Answer 1 for Query #5:

```
SELECT   ?X
WHERE {   ?X rdfs:subClassOf PO:VegetableTopping.}
ORDER BY ASC(?X)
LIMIT 3
OFFSET 3
```

PUCPR

QUERY #5 OFFSET

SPARQL query:

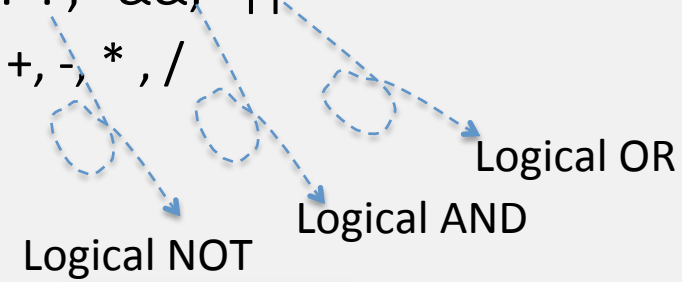
```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO: <http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

SELECT   ?X
WHERE {   ?X rdfs:subClassOf PO:VegetableTopping.}
ORDER BY ASC(?X)
LIMIT 3
OFFSET 3
```

X
OliveTopping
OnionTopping
PepperTopping

PUCPR

The SPARQL Query Structure

- The Query Patterns (2/5)
  - FILTER
    - Syntax: WHERE { Triple(s).
      FILTER (an expression).}
      - It filters out unwanted query results.
      - Built-in filter functions
        - » Comparison: =, >, <, !=, >=, <=
        - » Logical: ! , &&, ||
        - » Math: +, -, * , /



QUERY #6 FILTER (1/3)

- Find all the pizzas that are less than 400 grams
- Arrange the results by their weights in ascending order

Answer 1 for Query #6:

```
SELECT   ?X ?Y
WHERE {   ?X rdf:type  PO:Pizza.
          ?X PO:hasWeightInGrams ?Y.
          FILTER (?Y<400)}
ORDER BY (?Y)
```

PUCPR

QUERY #6 FILTER (1/3)

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO: <http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

SELECT   ?X ?Y
WHERE {   ?X rdf:type  PO:Pizza.
          ?X PO:hasWeightInGrams ?Y.
          FILTER (?Y<400)}
ORDER BY (?Y)
```

X	Y
PizzaID4	"175"^^<http://www.w3.org/2001/XMLSchema#integer>
PizzaID6	"370"^^<http://www.w3.org/2001/XMLSchema#integer>
PizzaID2	"375"^^<http://www.w3.org/2001/XMLSchema#integer>
PizzaID7	"380"^^<http://www.w3.org/2001/XMLSchema#integer>

PUCPR

QUERY #6 FILTER (2/3)

- Find all the pizzas that are greater than 300 grams and less than 400 grams
- Arrange the results by their weights in ascending order

Answer 1 for Query #6 (2/3):

```
SELECT   ?X ?Y
WHERE {   ?X rdf:type  PO:Pizza.
          ?X PO:hasWeightInGrams ?Y.
          FILTER ( (?Y>300)  &&  (?Y<400) )}
ORDER BY (?Y)
```

PUCPR

QUERY #6 FILTER (2/3)

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO: <http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

SELECT   ?X ?Y
WHERE {   ?X rdf:type  PO:Pizza.
          ?X PO:hasWeightInGrams ?Y.
          FILTER ((?Y>300)&&(?Y<400))}
ORDER BY (?Y)
```

X	Y
PizzaID6	"370"^^<http://www.w3.org/2001/XMLSchema#integer>
PizzaID2	"375"^^<http://www.w3.org/2001/XMLSchema#integer>
PizzaID7	"380"^^<http://www.w3.org/2001/XMLSchema#integer>

PUCPR

QUERY #6 FILTER (3/3)

- Find all the pizzas that the weights of their bases are more than half of their own weights
- Arrange the results by their weights in ascending order

Answer 1 for Query #6 (3/3):

```
SELECT   ?X ?Y ?Z ?I
WHERE {   ?X rdf:type  PO:Pizza.
          ?X PO:hasWeightInGrams ?Y.
          ?X PO:hasBase ?Z.
          ?Z PO:hasWeightInGrams ?I.
          FILTER (?I/?Y>0.5)}
ORDER BY (?Y)
```

PUCPR

QUERY #6 FILTER (3/3)

SPARQL query:

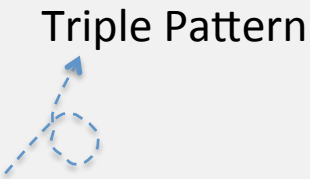
```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO: <http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

SELECT   ?X ?Y ?Z ?I
WHERE {   ?X rdf:type  PO:Pizza.
          ?X PO:hasWeightInGrams ?Y.
          ?X PO:hasBase ?Z.
          ?Z PO:hasWeightInGrams ?I.
          FILTER (?I/?Y>0.5)}
ORDER BY (?Y)
```

X	Y	Z	I
PizzaID1	"470"^^<http://ww	PizzaBaseNo1-1	"310"^^<http://ww

PUCPR

The SPARQL Query Structure

- The Query Patterns (3/5)
  - OPTIONAL
    - Syntax: WHERE { Triple(s).



OPTIONAL { Triple(s)}.}

      - Usually it is inside some other query patterns
      - Even if the optional triple pattern(s) is (are) failed, it still returns the results that match the no optional one(s).
      - E.g.

```
WHERE {?X  PO:hasProductionDate ?Y.
        OPTIONAL {?X PO:hasWeightInGrams?Z.}
       }
```

PUCPR

QUERY #7 OPTIONAL

- Find all the items that have production dates and optionally have weights in grams.
- Arrange the results by item names in ascending order

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO: <http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>
SELECT *
WHERE {
        ?X  PO:hasProductionDate ?Z.
        ?X  PO:hasWeightInGrams ?Y.

 }
ORDER BY (?X)
```

X	Z	Y
PizzaID1	"2015-11-06T14:00:00"^^<http://www.\	"470"^^<http://www.w3.org/2001/XMLSchema#integer>
PizzaID2	"2015-11-06T14:05:00"^^<http://www.\	"375"^^<http://www.w3.org/2001/XMLSchema#integer>
PizzaID3	"2015-11-06T13:00:00"^^<http://www.\	"565"^^<http://www.w3.org/2001/XMLSchema#integer>
PizzaID4	"2015-11-06T13:30:00"^^<http://www.\	"175"^^<http://www.w3.org/2001/XMLSchema#integer>
PizzaID5	"2015-10-02T08:00:00"^^<http://www.\	"560"^^<http://www.w3.org/2001/XMLSchema#integer>
PizzaID6	"2015-10-02T09:00:00"^^<http://www.\	"370"^^<http://www.w3.org/2001/XMLSchema#integer>
PizzaID7	"2015-11-06T15:00:00"^^<http://www.\	"380"^^<http://www.w3.org/2001/XMLSchema#integer>

A query without OPTIONAL

```
SELECT *
WHERE {
        ?X  PO:hasProductionDate ?Z.
        ?X  PO:hasWeightInGrams ?Y.}
ORDER BY (?X)
```

PUCPR

QUERY #7 OPTIONAL

- Find all the items that have production dates and optionally have weights in grams.
- Arrange the results by item names in ascending order

Answer 1 for Query #7:

```
SELECT *
WHERE {
        ?X  PO:hasProductionDate ?Z.
        OPTIONAL { ?X  PO:hasWeightInGrams ?Y.}
 }
ORDER BY (?X)
```

PUCPR

QUERY #7 OPTIONAL

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO: <http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

SELECT *
WHERE {
        ?X  PO:hasProductionDate ?Z.
        OPTIONAL { ?X  PO:hasWeightInGrams ?Y.}
 }
ORDER BY (?X)
```

X	Z	Y
ExamplePizzaX	"2015-07-05T09:00:00"^^<http://www	
PizzaID1	"2015-11-06T14:00:00"^^<http://www	"470"^^<http://www.w3.org/2001/XMLSchema#integer>
PizzaID2	"2015-11-06T14:05:00"^^<http://www	"375"^^<http://www.w3.org/2001/XMLSchema#integer>
PizzaID3	"2015-11-06T13:00:00"^^<http://www	"565"^^<http://www.w3.org/2001/XMLSchema#integer>
PizzaID4	"2015-11-06T13:30:00"^^<http://www	"175"^^<http://www.w3.org/2001/XMLSchema#integer>
PizzaID5	"2015-10-02T08:00:00"^^<http://www	"560"^^<http://www.w3.org/2001/XMLSchema#integer>
PizzaID6	"2015-10-02T09:00:00"^^<http://www	"370"^^<http://www.w3.org/2001/XMLSchema#integer>
PizzaID7	"2015-11-06T15:00:00"^^<http://www	"380"^^<http://www.w3.org/2001/XMLSchema#integer>

PUCPR

The SPARQL Query Structure

- The Query Patterns (4/5)
  - UNION
    - Syntax:

```
WHERE {   {Triple(s).}
          UNION
          {Triple(s).}
          ...
      }
```

Triple Pattern 1

Triple Pattern 2

      - Usually it is inside some other query patterns
      - It forms a disjunction of two or more triple patterns.
      - The variables that satisfy either one those triple patterns will be returned as results.

QUERY #8 UNION

- Find all kinds of Pizza Toppings that are either Cheese Toppings or Vegetable Toppings
- One variable

Answer 1 for Query #8:

```
SELECT   ?X
WHERE {
          {?X rdfs:subClassOf  PO:CheeseTopping.}
          UNION
          {?X rdfs:subClassOf PO:VegetableTopping.}
        }
```

PUCPR

QUERY #8 UNION

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO:<http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

SELECT   ?X
WHERE {
            {?X rdfs:subClassOf  PO:CheeseTopping.}
            UNION
            {?X rdfs:subClassOf PO:VegetableTopping.}
          }
```

X
ParmesanTopping
MozzarellaTopping
OliveTopping
CaperTopping
GarlicTopping
TomatoTopping
PepperTopping
MushroomTopping
OnionTopping

PUCPR

The SPARQL Query Structure

- The Query Patterns (5/5)
  - MINUS
    - Syntax: WHERE { Triple(s).
      MINUS{ Triple(s)}.
      }
    - Usually it is inside some other query patterns
    - It remove the solutions that match its description.

PUCPR

QUERY #9 MINUS

- Find all the pizzas that have not Mozzarella Toppings.

Answer 1 for Query #9:

```
SELECT  ?X
WHERE {
        ?X   rdf:type          PO:Pizza.
         MINUS{
                ?X PO:hasTopping ?Y.
                ?Y rdf:type PO:MozzarellaTopping.
               }

       }
```

PUCPR

QUERY #9 MINUS

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO:<http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>


SELECT  ?X
WHERE { ?X   rdf:type            PO:Pizza.
             MINUS{?X PO:hasTopping ?Y.
                       ?Y rdf:type PO:MozzarellaTopping.
                      }
```

X
PizzaID6
ExamplePizzaX
PizzaID7

PUCPR

The SPARQL Query Structure

- The Result Forms (2/2)
  - ASK
    - ASK { Triple(s) }
      - Different from SELECT
      - It returns “true” of “false” as an answer
      - Each Triple is ended by the “.” symbol
    - If any query pattern has any matches in the ontology it returns “true”, otherwise “false”.

PUCPR

QUERY #10 ASK

- Is PizzaID1 heavier than the PizzaID5?

Answer 1 for Query #10:

```
ASK
WHERE {
          PO:PizzaID1 PO:hasWeightInGrams ?X.
          PO:PizzaID5 PO:hasWeightInGrams ?Y.
          FILTER(?X>?Y)
        }
```

Answer 2 for Query #10:

```
ASK
WHERE {
          PO:PizzaID1 PO:isHeavierThan PO:PizzaID5
        }
```

PUCPR

QUERY #10 ASK

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO: <http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

ASK
WHERE { PO:PizzaID1 PO:hasWeightInGrams ?X.
        PO:PizzaID5 PO:hasWeightInGrams ?Y.
        FILTER(?X>?Y)
      }
```

Result

False

PUCPR

The SPARQL Query Structure

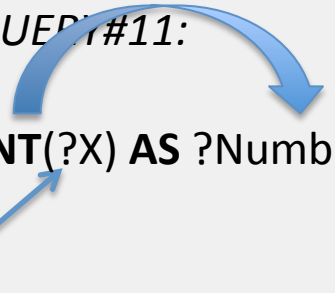
- Aggregates (1/2)
  - Apply expressions over groups of solutions.
  - Use when the querier wishes to see the result that is computed over a group of solutions, rather than a single solution.
  - SUM, MAX, COUNT, AVG, …
    - (SUM(?X) AS ?Y)
    - (MAX(?X) AS ?Y)
    - (COUNT(?X) AS ?Y)
    - …
    - E.g. The average weight of all pizzas
      The sum weight of all toppings on top of a pizza
      The number of pizza toppings on top of a pizza
      …

QUERY #11 COUNT

- How many Pizzas?

Answer 1 for QUERY#11:

```
SELECT   (COUNT(?X) AS ?NumberOfPizza)

WHERE {
          ?X   rdf:type    PO:Pizza.
         }
```



QUERY #11 COUNT

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO:<http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

SELECT  (COUNT(?X) AS ?NumberOfPizza)
WHERE { ?X   rdf:type    PO:Pizza.
        }
```

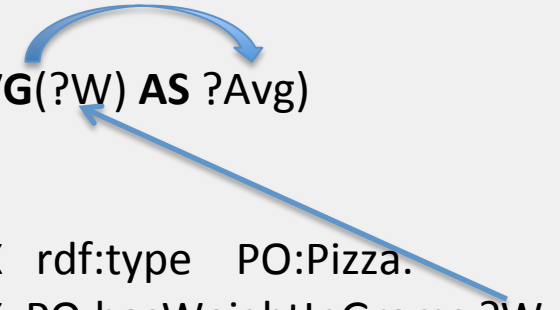
NumberOfPizza
"8"^^<http://www.w3.org/2001/XMLSchema#integer>

PUCPR

QUERY #12 AVG

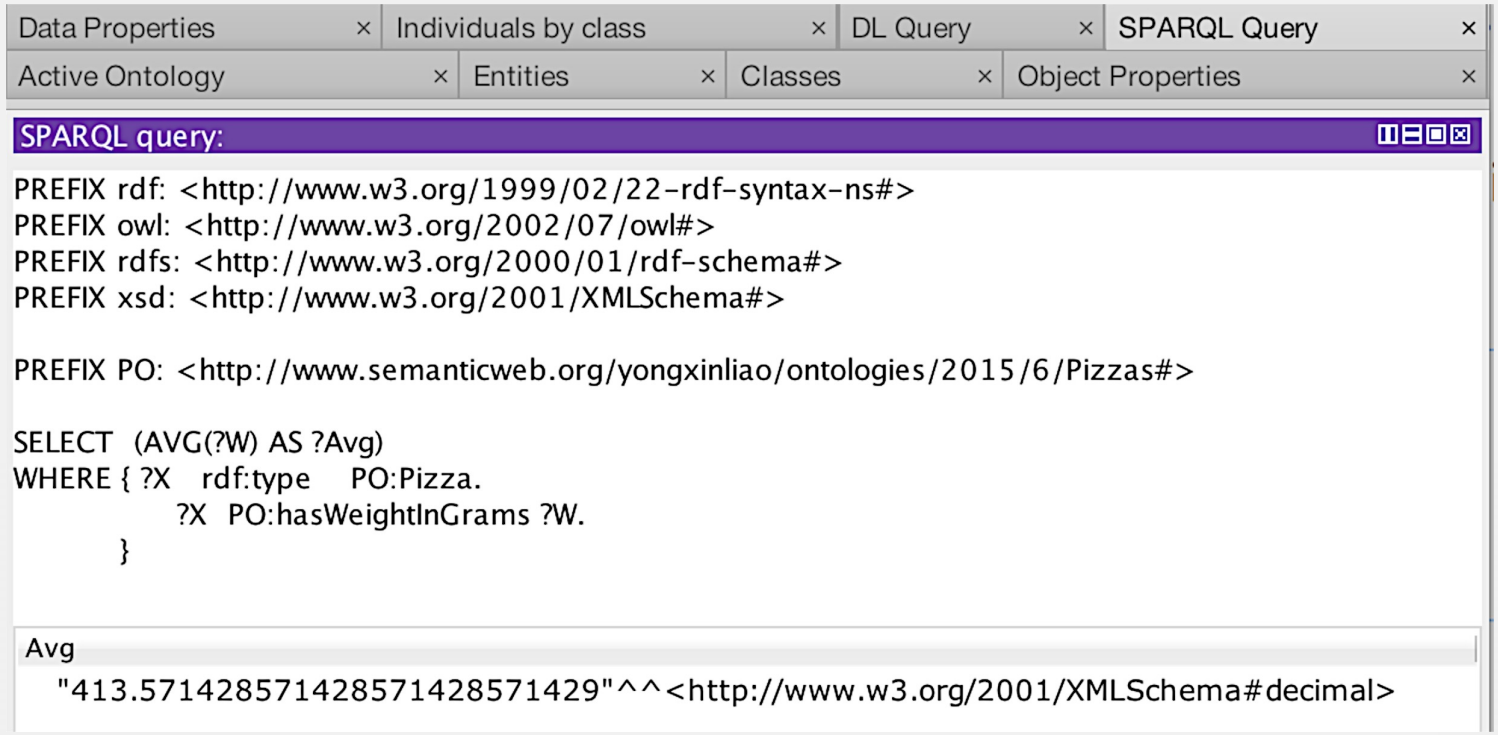
- What is the average weight of all those Pizzas?

Answer 1 for QUERY#12:



```
SELECT  (AVG(?W) AS ?Avg)

WHERE {
          ?X   rdf:type    PO:Pizza.
          ?X  PO:hasWeightInGrams ?W.
       }
```

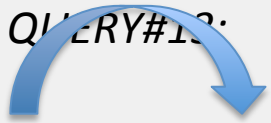
QUERY #12 AVG


Data Properties
Individuals by class
DL Query
SPARQL Query
Active Ontology
Entities
Classes
Object Properties
SPARQL query:
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX PO: <http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>
SELECT (AVG(?W) AS ?Avg)
WHERE { ?X rdf:type PO:Pizza.
?X PO:hasWeightInGrams ?W.
}
Avg
"413.57142857142857142857429"^^<http://www.w3.org/2001/XMLSchema#decimal>

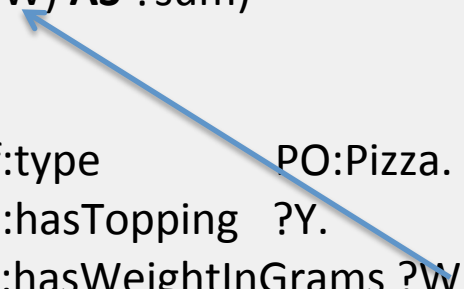
QUERY #13 SUM

- What is the total weight of all the Pizza Toppings?

Answer 1 for QUERY#13:



```
SELECT  (SUM(?W) AS ?sum)

WHERE {
        ?X  rdf:type            PO:Pizza.
        ?X  PO:hasTopping   ?Y.
        ?Y  PO:hasWeightInGrams ?W.
}
```



QUERY #13 SUM

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO: <http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

SELECT  (SUM(?W) AS ?sum)
WHERE { ?X   rdf:type              PO:Pizza.
        ?X   PO:hasTopping   ?Y.
        ?Y   PO:hasWeightInGrams ?W.
}
```

sum
"1565"^^<http://www.w3.org/2001/XMLSchema#decimal>

PUCPR

The SPARQL Query Structure

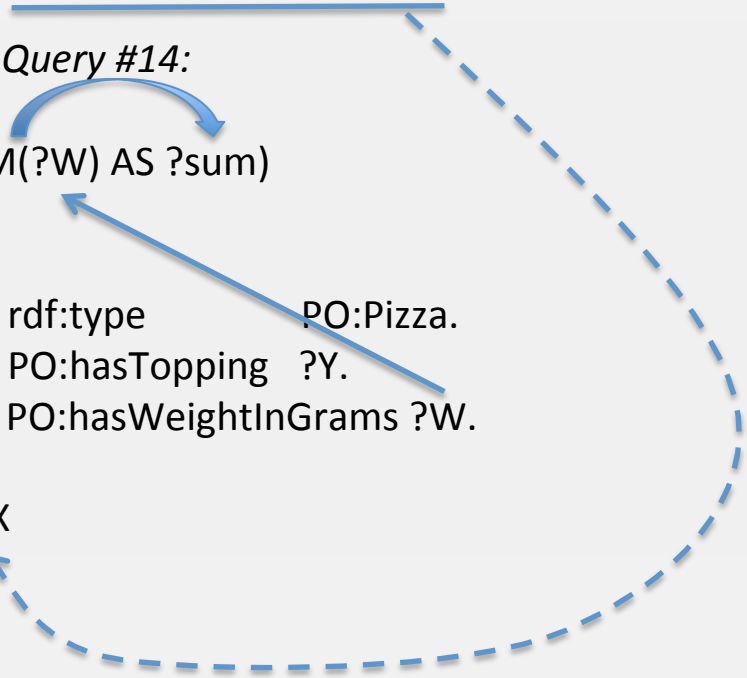
- Aggregates (2/2)
  - Apply expressions over groups of solutions.
  - Use when the querier wishes to see a result which is computed over a group of solutions, rather than a single solution.
  - GROUP BY
    - Syntax: GROUP BY ?variable
    - It is used together with other aggregates (SUM, MAX, COUNT, AVG, …)
    - In order to calculate aggregate values for a solution, the solution is first divided into one or more groups, and the aggregate value is calculated for each group.

Query #14 GROUP BY

- Calculate the total weight of all the pizza toppings that each pizza has.

Answer 1 for Query #14:

```
SELECT   (SUM(?W) AS ?sum)

WHERE {
          ?X   rdf:type              PO:Pizza.
          ?X   PO:hasTopping   ?Y.
          ?Y   PO:hasWeightInGrams ?W.
}
GROUP BY ?X
```



Query #14 GROUP BY

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO: <http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

SELECT  (SUM(?W) AS ?sum)
WHERE { ?X   rdf:type              PO:Pizza.
            ?X   PO:hasTopping   ?Y.
            ?Y   PO:hasWeightInGrams ?W.
}
GROUP BY ?X
```

sum
"200"^^<http://www.w3.org/2001/XMLSchema#decimal>
"225"^^<http://www.w3.org/2001/XMLSchema#decimal>
"195"^^<http://www.w3.org/2001/XMLSchema#decimal>
"90"^^<http://www.w3.org/2001/XMLSchema#decimal>
"160"^^<http://www.w3.org/2001/XMLSchema#decimal>
"310"^^<http://www.w3.org/2001/XMLSchema#decimal>
"385"^^<http://www.w3.org/2001/XMLSchema#decimal>

PUCPR

Query #14 GROUP BY

- Calculate the total weight of all the pizza toppings that each pizza has.

Answer 2 for Query #14:

```
SELECT  ?X (SUM(?W) AS ?sum)
WHERE { ?X   rdf:type              PO:Pizza.
        ?X   PO:hasTopping   ?Y.
        ?Y   PO:hasWeightInGrams ?W.
}
GROUP BY ?X
```

PUCPR

Query #14 GROUP BY

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO: <http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

SELECT  ?X (SUM(?W) AS ?sum)
WHERE { ?X   rdf:type              PO:Pizza.
            ?X   PO:hasTopping   ?Y.
            ?Y   PO:hasWeightInGrams ?W.
}
GROUP BY ?X
```

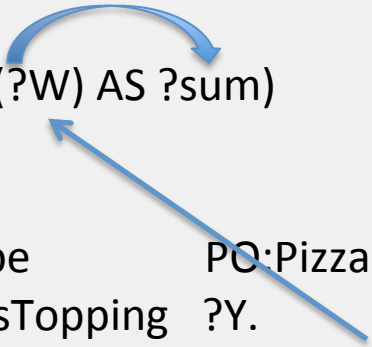
X	sum
PizzaID7	"200"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID2	"225"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID6	"195"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID4	"90"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID1	"160"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID5	"310"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID3	"385"^^<http://www.w3.org/2001/XMLSchema#decimal>

PUCPR

Query #15 Property Path

- Calculate the total weights for each kind of the Pizza toppings.

Answer 1 for Exercise #14:



```
SELECT  ?Z  (SUM(?W) AS ?sum)

WHERE {
        ?X   rdf:type              PO:Pizza.
        ?X   PO:hasTopping   ?Y.
        ?Y   PO:hasWeightInGrams ?W.
        ?Y   rdf:type            ?Z.
        ?Z   rdfs:subClassOf  PO:PizzaTopping.
}
GROUP BY ?Z
```

PUCPR

Query #15 Property Path

- Calculate the total weights for each kind of the Pizza toppings.

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO:<http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>
SELECT  ?Z (SUM(?W) AS ?sum)
WHERE { ?X   rdf:type              PO:Pizza.
        ?X   PO:hasTopping   ?Y.
        ?Y   rdf:type             ?Z.
        ?Z   rdfs:subClassOf  PO:PizzaTopping.
        ?Y   PO:hasWeightInGrams ?W.
}
GROUP BY ?Z
```

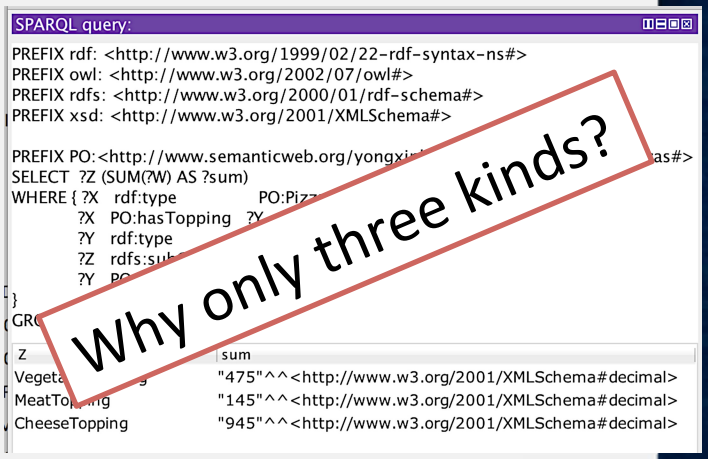
Z	sum
VegetableTopping	"475"^^<http://www.w3.org/2001/XMLSchema#decimal>
MeatTopping	"145"^^<http://www.w3.org/2001/XMLSchema#decimal>
CheeseTopping	"945"^^<http://www.w3.org/2001/XMLSchema#decimal>

PUCPR

Query #15 Property Path

- Calculate the total weights for each kind of the Pizza toppings.

```
SELECT  ?Z (SUM(?W) AS ?sum)
WHERE { ?X   rdf:type              PO:Pizza.
        ?X   PO:hasTopping   ?Y.
        ?Y   rdf:type              ?Z.
        ?Z   rdfs:subClassOf  PO:PizzaTopping.
        ?Y   PO:hasWeightInGrams ?W.
}
GROUP BY ?Z
```



- Property Path
  - It enables the query for any-length path.
    - E.g. rdfs:subClassOf*

PUCPR

Query #15 Property Path

- Property Path

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO:<http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

SELECT  ?Z (SUM(?W) AS ?sum)
WHERE { ?X   rdf:type              PO:Pizza.
        ?X   PO:hasTopping   ?Y.
        ?Y   rdf:type              ?Z
        ?Z   rdfs:subClassOf*  PO:PizzaTopping.
        ?Y   PO:hasWeightInGrams ?W.
}
GROUP BY ?Z
```

Z	sum
VegetableTopping	"475"^^<http://www.w3.org/2001/XMLSchema#decimal>
TomatoTopping	"380"^^<http://www.w3.org/2001/XMLSchema#decimal>
TomatoSauceTopping	"380"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaTopping	"1565"^^<http://www.w3.org/2001/XMLSchema#decimal>
GarlicTopping	"45"^^<http://www.w3.org/2001/XMLSchema#decimal>
PepperoniTopping	"145"^^<http://www.w3.org/2001/XMLSchema#decimal>
MeatTopping	"145"^^<http://www.w3.org/2001/XMLSchema#decimal>
CheeseTopping	"945"^^<http://www.w3.org/2001/XMLSchema#decimal>
ParmesanTopping	"145"^^<http://www.w3.org/2001/XMLSchema#decimal>
OliveTopping	"45"^^<http://www.w3.org/2001/XMLSchema#decimal>
MozzarellaTopping	"800"^^<http://www.w3.org/2001/XMLSchema#decimal>
PepperTopping	"5"^^<http://www.w3.org/2001/XMLSchema#decimal>
JalapenoPepperTopping	"5"^^<http://www.w3.org/2001/XMLSchema#decimal>

PUCPR

Property Path

- Property Path Syntax
  - Sequence Path

    $Property_1 / \ldots / Property_n$

    - A path from $Property_1$ to $Property_n$.
  - Alternative Path

    $Property_1 | \ldots | Property_n$

    - A path of $Property_1$ or $Property_n$ (all possibilities will be tried)
  - Other Property Paths

    Zero or more $property_1$ path: $Property_1*$

    One or more $property_1$ path: $Property_1+$

    Zero or one $property_1$ path: $Property_1?$

Query#15 Property Path

- Use two variables and the Sequence Path to find all the pizzas that have their bases heavier than 150 grams
- Use only three (or two) triple pattern in WHERE.

```
SELECT  ?X  ?Y ?Z
WHERE { ?X   rdf:type          PO:Pizza.
          ?X   PO:hasBase     ?Z.
          ?Z   PO:hasWeightInGrams  ?Y.
          FILTER(?Y>150)
}
```

Answer 1 for Query #15:

```
SELECT  ?X  ?Y
WHERE { ?X   rdf:type      PO:Pizza.
          ?X   PO:hasBase/PO:hasWeightInGrams  ?Y.
          FILTER(?Y>150)
}
```

PUCPR

Query#15 Property Path

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO:<http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

SELECT  ?X  ?Y
WHERE { ?X   rdf:type       PO:Pizza.
            ?X   PO:hasBase/PO:hasWeightInGrams  ?Y.
            FILTER(?Y>150)
}
```

X	Y
PizzaID3	"180"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID7	"180"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID1	"310"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID6	"175"^^<http://www.w3.org/2001/XMLSchema#decimal>
PizzaID5	"250"^^<http://www.w3.org/2001/XMLSchema#decimal>

PUCPR

The SPARQL Query Structure

- DISTINCT
  - Syntax: SELECT **DISTINCT** ?variable
    - It ensures that duplicate solutions are eliminated from the solution set.
    - Only one solution that binds the same variables to the same RDF terms is returned from the query

QUERY #16 DISTINCT

- Find all the pizzas that have Vegetable Toppings
- Without duplicate solutions

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX
PO:<http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

SELECT   ?X
WHERE {   ?X   rdf:type PO:Pizza.
          ?X   PO:hasTopping/rdf:type PO:VegetableTopping
        }
ORDER BY ?X
```

X
PizzaID1
PizzaID2
PizzaID3
PizzaID4
PizzaID4
PizzaID5
PizzaID6
PizzaID6
PizzaID6
PizzaID7
PizzaID7
PizzaID7

Answer 1 for Query #16:

```
SELECT   DISTINCT ?X
WHERE {
          ?X   rdf:type PO:Pizza.
          ?X   PO:hasTopping/rdf:type PO:VegetableTopping
       }
ORDER BY ?X
```

PUCPR

QUERY #16 DISTINCT

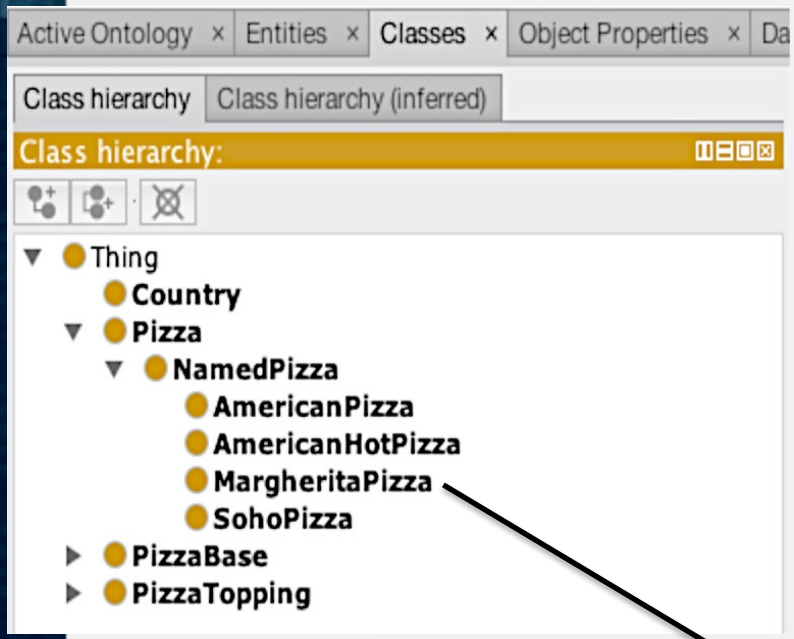
SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO:<http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

SELECT  DISTINCT ?X
WHERE {   ?X   rdf:type PO:Pizza.
          ?X   PO:hasTopping/rdf:type PO:VegetableTopping
        }
ORDER BY ?X
```

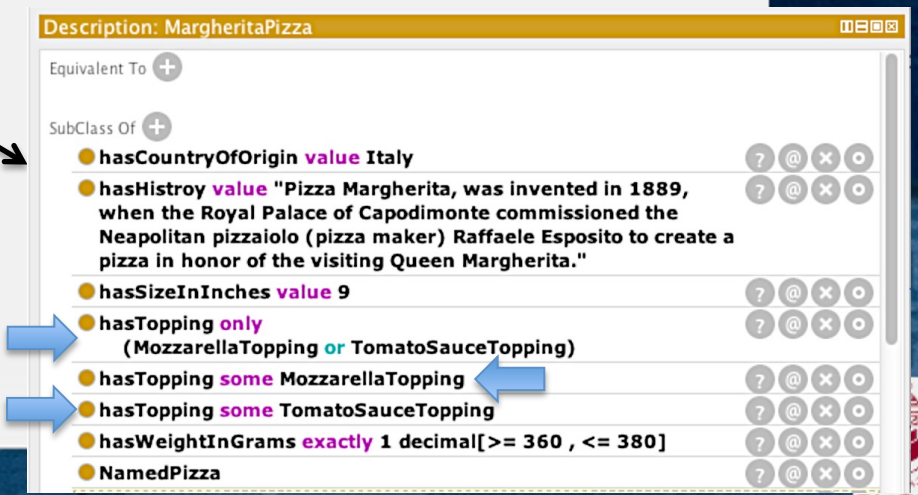
X
PizzaID1
PizzaID2
PizzaID3
PizzaID4
PizzaID5
PizzaID6
PizzaID7

PUCPR

Protégé Practices of Lecture 04-05



Query the information
in complex class expressions
about the pizza toppings



PUCPR

Subject, Predicate, and Object

- How to represent Complex Class Expressions?
  - A complex class expression is an anonymous class
    - It expresses its semantics through placing constrains on the existing named classes and properties
    - It has not IRI

(a resource that has an IRI)

(a resource that has an IRI)

(a resource that has an IRI or a data value)

  - Solution: RDF Blank Nodes
    - Itself does not contain any data, but serves as a parent node to a grouping of data.
    - Their identifiers are local identifiers

subject	object
MargheritaPizza	_:node1a2sag221x86
Pizza	_:node1a2sag221x92

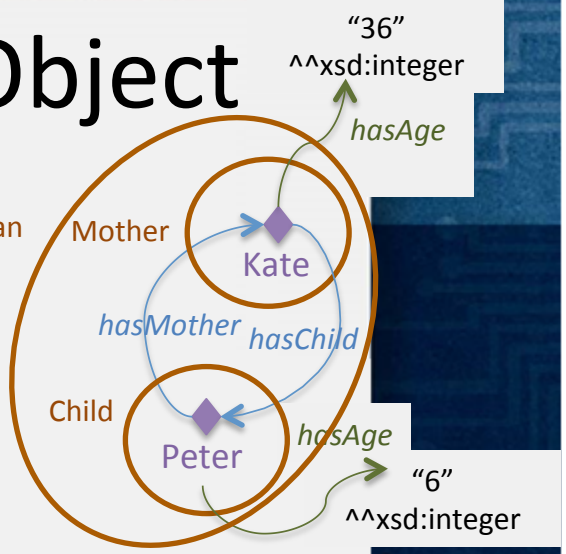
PUCPR

Subject, Predicate, and Object

- Each Triple Describes a Fact

Object Property Cardinality Restrictions

E.g. Child is subclass of hasMother exactly 1 Mother



Subject	Predicate	Object	
O1:Child	rdfs:subClassOf	[blank node x]	Blank nodes
[blank node x]	rdf:type	owl:Restriction	owl built-in classes
[blank node x]	owl:onProperty	O1:hasMother	Object Properties
[blank node x]	owl:onClass	O1:Mother	Classes
[blank node x]	owl:qualifiedCardinality	"1"^^xsd:nonNegativeInteger	

Classes

Blank nodes

rdfs built-in properties

rdf built-in properties

owl built-in properties

Data Values

```
PREFIX  O1:   <http://www.semanticweb.org/SPARQL-Example#>.
PREFIX  rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
PREFIX  rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
PREFIX  owl:  <http://www.w3.org/2002/07/owl#>.
```

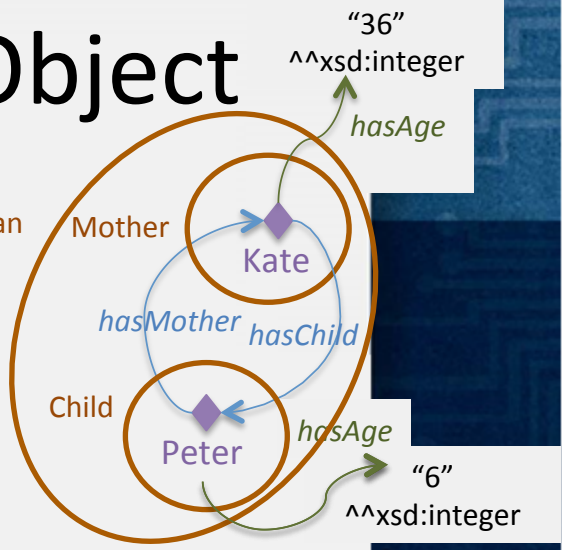
PUCPR

Subject, Predicate, and Object

- Each Triple Describes a Fact

Object Property Quantifier Restrictions

E.g. Mother is subclass of hasChild some Child



Subject	Predicate	Object	
O1:Mother	rdfs:subClassOf	[blank node y]	Blank nodes
[blank node y]	rdf:type	owl:Restriction	owl built-in classes
[blank node y]	owl:onProperty	O1:hasChild	Object Properties
[blank node y]	owl:someValuesFrom	O1:Child	Classes

Classes

Blank nodes

rdfs built-in properties

rdf built-in properties

owl built-in properties

PREFIX O1: <http://www.semanticweb.org/SPARQL-Example#>.
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
PREFIX owl: <http://www.w3.org/2002/07/owl#>.

PUCPR

QUERY #17 Select

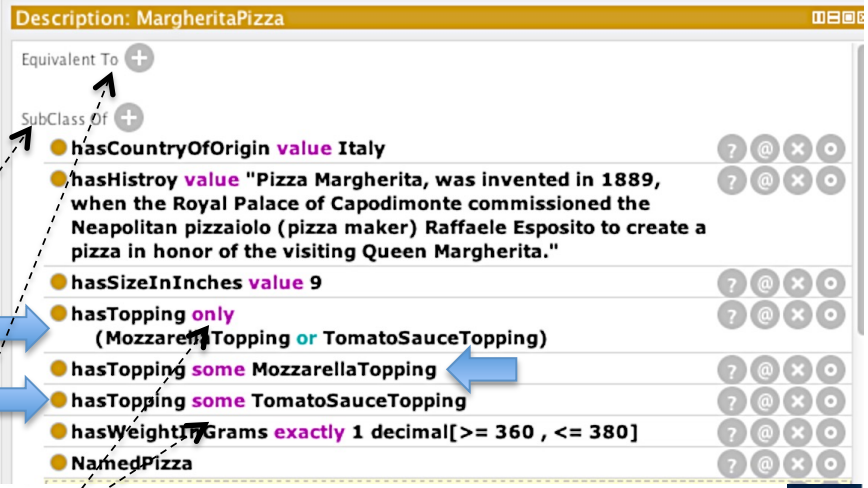
- Find what kinds of the pizza toppings that Margherita Pizzas have.
  - Query Complex Class Expressions of PO:MargheritaPizza (contain RDF Blank Nodes)
    - Query the subject (*PO:MargheritaPizza*) that is connected to all the objects (*?x*) by the predicate (*rdfs:subClassOf* or *owl:equivalentClass*).
    - Query all the subjects (?x) that are connected to the object (PO:hasTopping) by the predicate (owl:onProperty)
    - Query all the subjects (?x) that are connected to all the objects (?y) by the predicate (owl:someValuesFrom or owl:allValuesFrom)

Answer 1 for Query #2 (2/2):

```
SELECT ?x ?y
WHERE { PO:MargheritaPizza rdfs:subClassOf|owl:equivalentClass ?x.
        ?x owl:onProperty PO:hasTopping.
        ?x owl:someValuesFrom|owl:allValuesFrom ?y }
```

PUCPR

QUERY #17 Select (Multiple Variables)



SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX PO:<http://www.semanticweb.org/yongxinliao/ontologies/2015/6/Pizzas#>

SELECT ?x ?y
WHERE { PO:MargheritaPizza rdfs:subClassOf|owl:equivalentClass ?x.
        ?x owl:onProperty PO:hasTopping.
        ?x owl:someValuesFrom|owl:allValuesFrom ?y }
```

x	y
hasTopping some MozzarellaTopping	MozzarellaTopping
hasTopping some TomatoSauceTopping	TomatoSauceTopping
hasTopping only (MozzarellaTopping or TomatoSauceTopping)	MozzarellaTopping or TomatoSauceTopping

Thank you
for your attention!

Any Questions?

PUCPR