UNIVERSITY OF Southampton
School of Electronics and Computer Science

SPARQL

Dr Nicholas Gibbins
nmg@ecs.soton.ac.uk

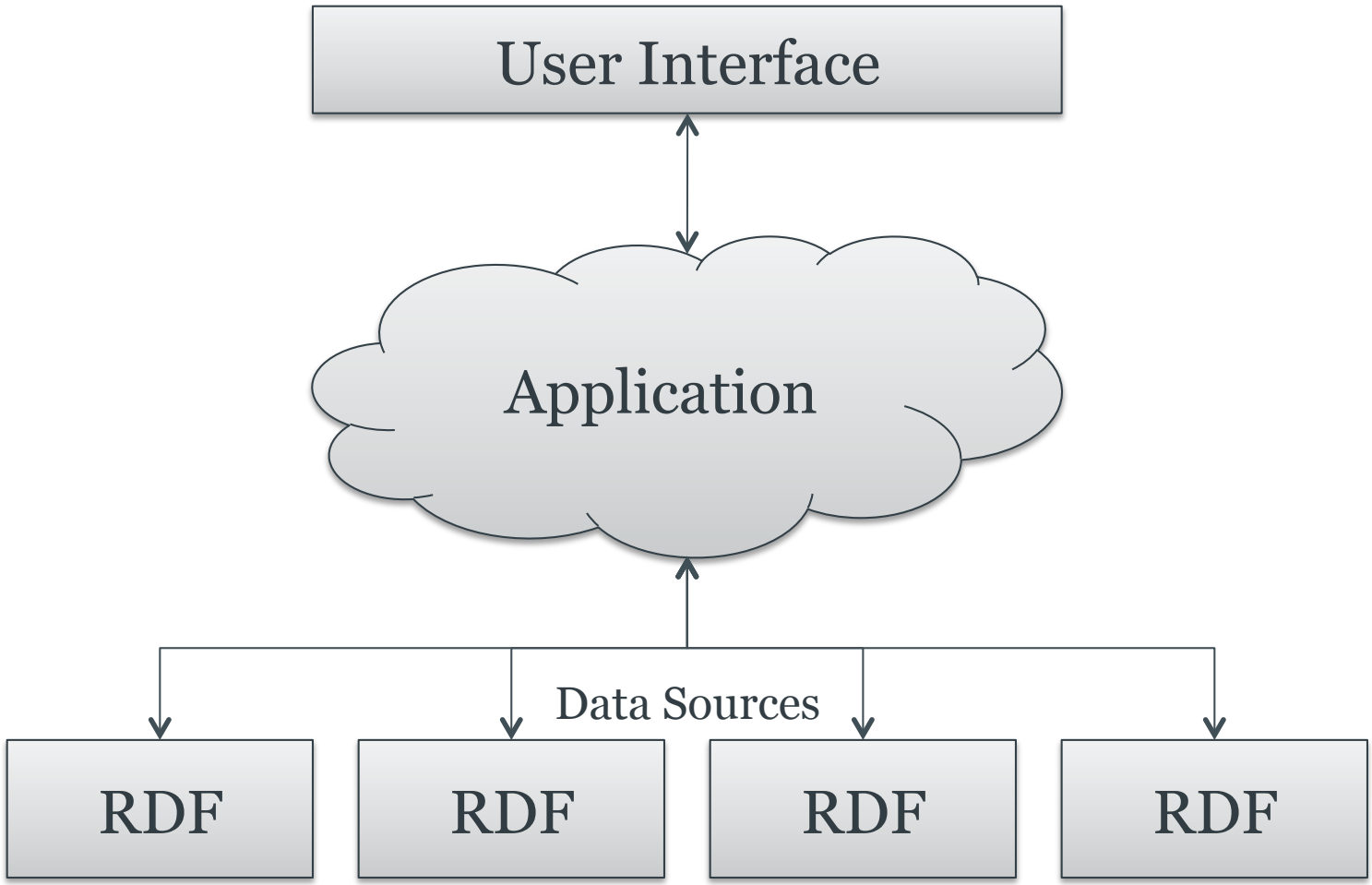
Semantic Web Applications

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- Technologies considered so far allow us to create representation schemes (RDFS, OWL) and to represent data (RDF)
- We can put data in – how do we get it back out?

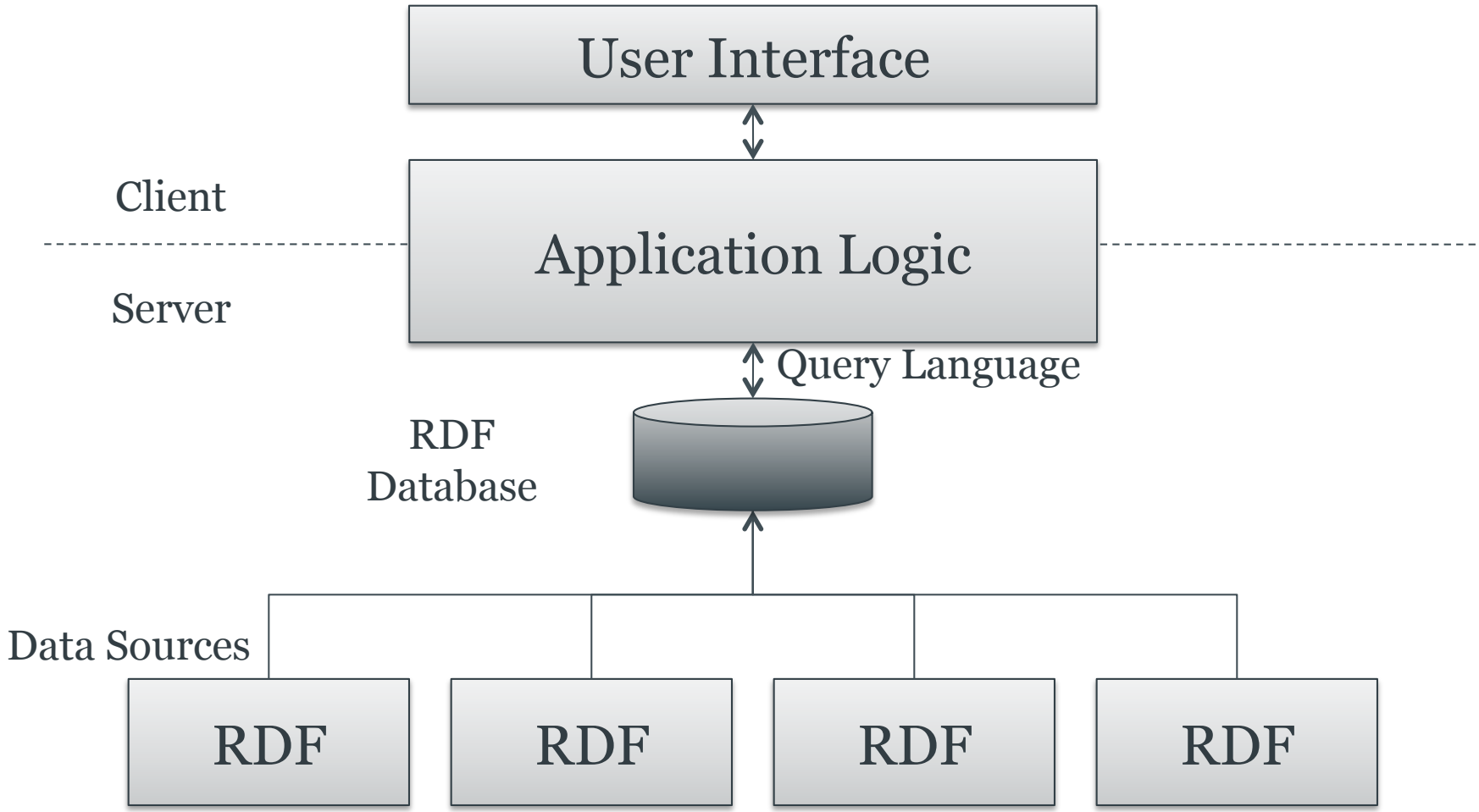
The Semantic Web Application

UNIVERSITY OF Southampton
School of Electronics and Computer Science



The Semantic Web Application

UNIVERSITY OF Southampton
School of Electronics and Computer Science



RDF Triplestores

UNIVERSITY OF Southampton
School of Electronics and Computer Science

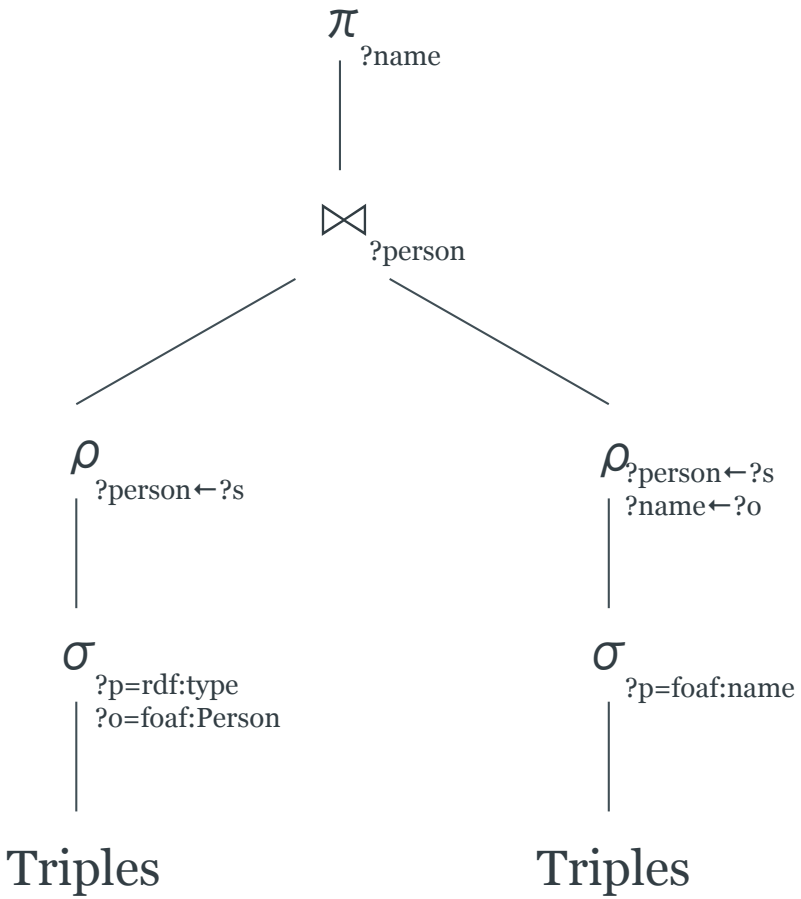
- Specialised databases for storing RDF data
- Can be viewed as a database containing a single table
  - Table contains subject/predicate/object as minimum
  - Many triplestores store quads to maintain provenance
  - May store other ancillary information

Subject	Predicate	Object	Graph
URI	URI	URI/Literal	URI
…	…	…	…

Querying Triplestores

UNIVERSITY OF Southampton
School of Electronics and Computer Science

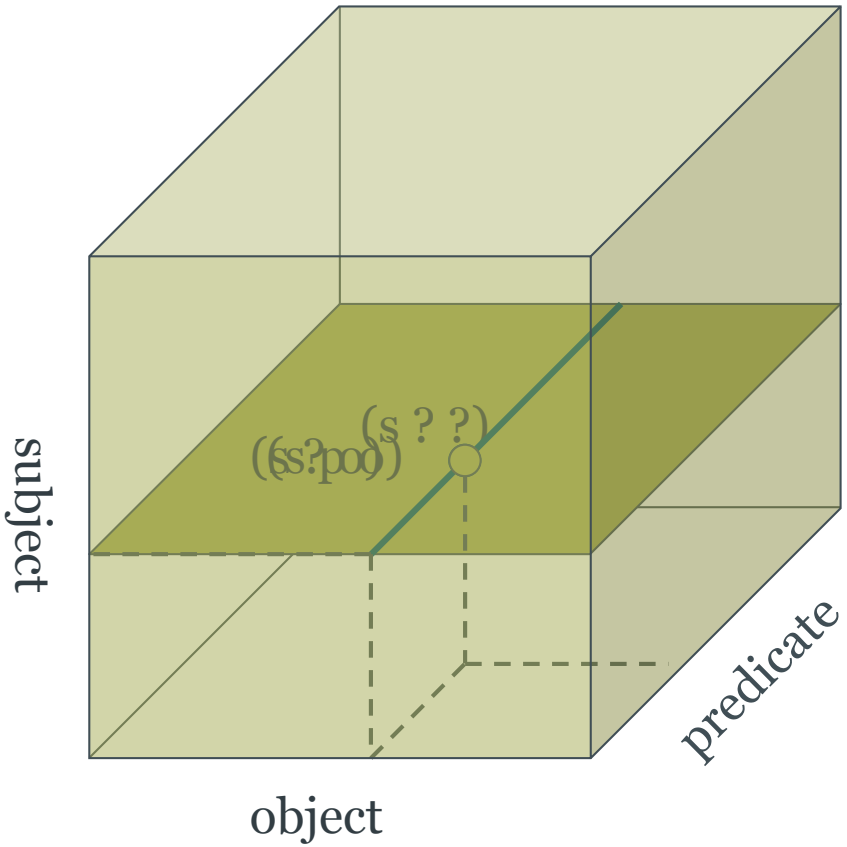
- Constructing queries for RDF data can be viewed as writing queries over this triple table
- “get all the names of all the people”



The Triplespace

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- The ranges of the subjects, predicates and objects define a space
- A triple is a point within this space
- Query patterns define regions within the space
  - (s p ?), (s ? o), (? p o) define lines
  - (s ? ?), (? p ?), (? ? o) define planes
- Answering queries = finding points in these regions



RDF Query Languages

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- RDF query languages were not standardised until recently
- Large number of different query languages around:
  - SerQL
  - SquishQL
  - RDQL
  - XsRQL
  - RQL
  - RDFQL

RDF Query Languages

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- Existing query languages mostly represent queries in the same way, as a set of triple patterns: (?s ?p ?o)
- Typical query structure:
  <list of variables to be bound and returned>
  <list of triple patterns and constraints>
- W3C has now specified a single query language which combines the best features of its predecessors: SPARQL

Introducing SPARQL

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- The SPARQL Protocol and RDF Query Language
- SQL-like syntax
  SELECT <variables>
  FROM <triple patterns>
- Triple patterns expressed in N3-like syntax
- Variables prefixed with '?':
  ?who foaf:knows http://example.org/person/fred .
- Queries return a result set of bindings for the variables

Example Query

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
_:a  foaf:name   "Johnny Lee Outlaw" .
_:a  foaf:mbox   <mailto:jlow@example.com> .
_:b  foaf:name   "Peter Goodguy" .
_:b  foaf:mbox   <mailto:peter@example.org> .
_:c  foaf:mbox   <mailto:carol@example.org> .
```

```
SELECT ?name ?mbox
WHERE {
      ?x foaf:name ?name .
      ?x foaf:mbox ?mbox .
}
```

Example Query

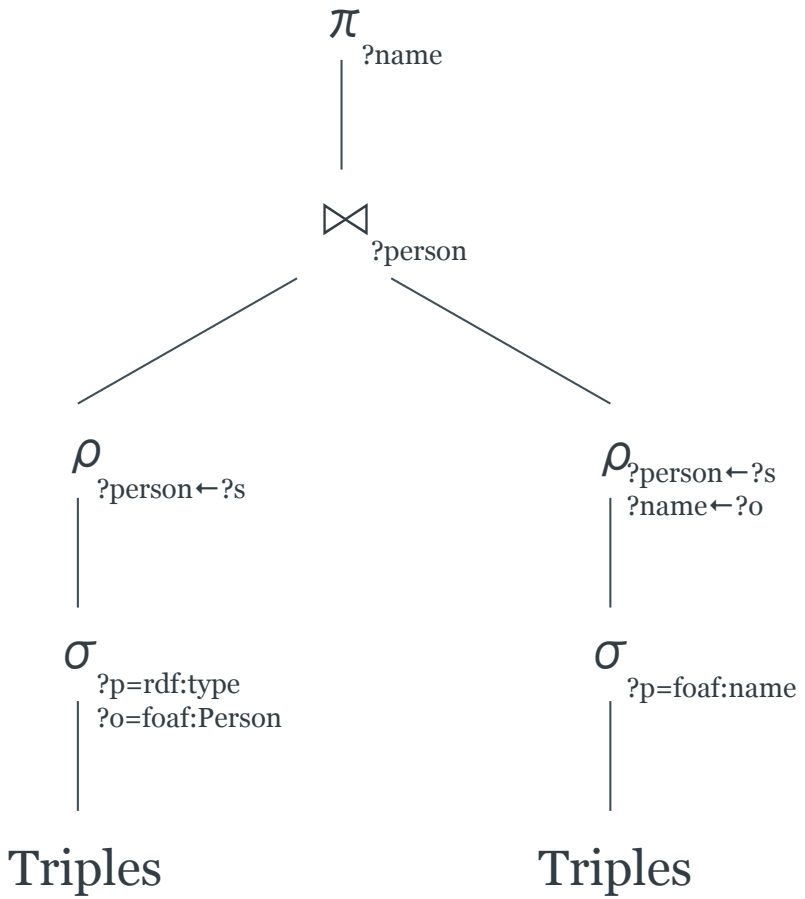
```
_:a foaf:name  "Johnny Lee Outlaw" .
_:a foaf:box   <mailto:jlow@example.com> .

_:b foaf:name  "Peter Goodguy" .
_:b foaf:box   <mailto:peter@example.org> .
```

?name	?mbox
"Johnny Lee Outlaw"	<mailto:jlow@example.com>
"Peter Goodguy"	<mailto:peter@example.org>

SPARQL Query Plan

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
SELECT ?name
WHERE {
    ?person rdf:type foaf:Person .
    ?person foaf:name ?name .
}
```



Namespaces in Queries

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- The query given in the previous example was over-simplified
- Need to define namespaces for vocabulary terms used:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name ?mbox
WHERE {      ?x foaf:name ?name .
             ?x foaf:mbox ?mbox }
```

Matching RDF Literals

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
@prefix ns:   <http://example.org/ns#> .
@prefix :     <http://example.org/ns#> .
@prefix xsd:  <http://www.w3.org/2001/XMLSchema#> .

:x   ns:p     "cat"@en .
:y   ns:p     "42"^^xsd:integer .
```

Matching RDF String Literals

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
SELECT ?v WHERE { ?v ?p "cat" }
```

- Returns no bindings
- “cat” must have language tag, as in graph: “cat”@en

```
SELECT ?v WHERE { ?v ?p "cat"@en }
```

- Returns a single binding

Matching RDF Numeric Literals

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
SELECT ?v WHERE { ?v ?p 42 }
```

- Returns a single binding
- Integers in SPARQL queries are implicitly typed as xsd:integer

```
"42"^^<http://www.w3.org/2001/XMLSchema#integer>
```

Blank Nodes

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- Blank nodes in a result set are scoped to that result set
- No guarantee that blank node labels will be unchanged over repeated queries

```
_:a  foaf:name   "Alice" .
_:b  foaf:name   "Bob" .

SELECT ?x ?name
WHERE  { ?x foaf:name ?name }
```

Blank Nodes

UNIVERSITY OF Southampton
School of Electronics and Computer Science

?x	?name
_:a	“Alice”
_:b	“Bob”

?x	?name
_:x	“Alice”
_:y	“Bob”

SPARQL Constraints

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- Constraints can be applied to variables:

```
SELECT  ?title
WHERE {       ?x dc:title ?title .
              FILTER regex(?title, "^SPARQL")  }

SELECT  ?title ?price
WHERE {       ?x ns:price ?price .
              FILTER (?price < 30.5)
              ?x dc:title ?title . }
```

Group Graph Patterns

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- Set of patterns enclosed in { }
- Determines scoping for FILTER operators (and other operators)

Optional Graph Patterns

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
_:a  rdf:type       foaf:Person .
_:a  foaf:name       "Alice" .
_:a  foaf:mbox       <mailto:alice@example.com> .
_:a  foaf:mbox       <mailto:alice@work.example.com> .
_:b  rdf:type       foaf:Person .
_:b  foaf:name       "Bob" .
```

```
SELECT ?name ?mbox
WHERE {
        ?x foaf:name  ?name .
        OPTIONAL {
                ?x  foaf:mbox  ?mbox
        }
}
```

Optional Graph Patterns

UNIVERSITY OF Southampton
School of Electronics and Computer Science

?name	?mbox
“Alice”	<mailto:alice@example.com>
“Alice”	<mailto:alice@work.example.com>
“Bob”	

Union Graph Patterns

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
PREFIX dc10:  <http://purl.org/dc/elements/1.0/>
PREFIX dc11:  <http://purl.org/dc/elements/1.1/>
SELECT ?title
WHERE {
      { ?book dc10:title  ?title }
      UNION
      { ?book dc11:title  ?title }
}
```

Specifying Datasets

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- RDF data may be published as multiple graphs (serialised in multiple RDF/XML documents)
- Sometimes we wish to be able to restrict a query to certain sources (graphs or datasets)

Default Graph

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT  ?name
FROM <http://example.org/foaf/aliceFoaf>
WHERE   { ?x foaf:name ?name }
```

- http://example.org/foaf/aliceFoaf is the default graph
- Unless otherwise specified, all triples are taken from that graph

Named Graphs

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
SELECT ?who ?g ?mbox
FROM <http://example.org/dft.ttl>
FROM NAMED <http://example.org/alice>
FROM NAMED <http://example.org/bob>
WHERE {
        ?g dc:publisher ?who .
        GRAPH ?g { ?x foaf:mbox ?mbox }
}
```

Ordering

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
PREFIX foaf:    <http://xmlns.com/foaf/0.1/>
SELECT ?name
WHERE { ?x foaf:name ?name ; :empId ?emp }
ORDER BY ?name DESC(?emp)
```

- Order results by ?name, and then by ?emp in descending order
- ASC() sorts in ascending order

Limit and Offset

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
PREFIX foaf:    <http://xmlns.com/foaf/0.1/>
SELECT  ?name
WHERE   { ?x foaf:name ?name }
ORDER BY ?name
LIMIT   5
OFFSET  10
```

- Returns five results, after skipping the first ten results

Distinct

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- Removes duplicate results

```
PREFIX foaf:    <http://xmlns.com/foaf/0.1/>
SELECT DISTINCT ?name
WHERE { ?x foaf:name ?name }
```

Other SPARQL Verbs

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- SELECT is not the only verb
  - DESCRIBE
  - CONSTRUCT
  - ASK

CONSTRUCT

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- A CONSTRUCT query returns an RDF graph, specified by a graph template
  - For each row in the result set, substitute the variables in the template with the values in that row
  - Combine the resulting graphs into a single graph (set union of triples)

CONSTRUCT

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
PREFIX foaf:    <http://xmlns.com/foaf/0.1/>
PREFIX vcard:   <http://www.w3.org/2001/vcard-rdf/3.0#>
CONSTRUCT   {
        ?x vcard:FN ?name .
        ?x vcard:EMAIL ?mail .
}
WHERE {
        ?x foaf:name ?name .
        ?x foaf:mbox ?mail .
 }
```

CONSTRUCT and bNodes

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- CONSTRUCT can create graphs with bNodes
- bNode labels are scoped to the template for each solution
  - If the same label occurs twice in a template, then there will be one blank node created for each query solution
  - There will be different blank nodes for triples generated by different query solutions

CONSTRUCT and bNodes

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
CONSTRUCT {
        ?x  vcard:N _:v .
        _:v vcard:givenName ?gname .
        _:v vcard:familyName ?fname .
}
WHERE {
        { ?x foaf:firstname ?gname } UNION
        { ?x foaf:givenname   ?gname } .
        { ?x foaf:surname   ?fname } UNION
        { ?x foaf:family_name ?fname } .
}
```

CONSTRUCT and bNodes

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
_:a    foaf:givenname "Alice" .
_:a    foaf:family_name "Hacker" .
_:b    foaf:firstname "Bob" .
_:b    foaf:surname  "Hacker" .

_:v1 vcard:N _:x .
_:x vcard:givenName "Alice" .
_:x vcard:familyName "Hacker" .
_:v2 vcard:N _:z .
_:z vcard:givenName "Bob" .
_:z vcard:familyName "Hacker" .
```

CONSTRUCT is not a rule language

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- CONSTRUCT matches graph patterns and returns a new graph
- We could implement a simple rule-based system using CONSTRUCT queries to represent rules
- From the Data Access Working Group Charter:

  "While it is hoped that the product of the RDF Data Access Working Group will be useful in later development of a rules language, **development of such a rules language is out of scope for this working group**. However, any serializations of a query language must not preclude extension to, or inclusion in, a rules language. The group should expend minimal effort assuring that such an extension be intuitive and and consistent with any query language produced by the group."

Accessing graphs with CONSTRUCT

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
CONSTRUCT {
        ?s ?p ?o .
}
WHERE {
        GRAPH <http://example.org/aGraph> { ?s ?p ?o } .
}
```

Accessing graphs with CONSTRUCT

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
CONSTRUCT {
        ?s ?p ?o .
}
WHERE {
        GRAPH ?g { ?s ?p ?o } .
        { ?g dc:publisher <http://www.w3.org/> } .
        { ?g dc:date ?date } .
        FILTER ( ?date >
                "2005-02-28T00:00:00Z"^^xsd:dateTime ) .
}
```

ASK

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- An ASK query is used to check whether a query pattern has a solution.
- No information is returned about query solutions (no variable bindings), just a boolean value

ASK Example

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
_:a  foaf:name       "Alice" .
_:a  foaf:homepage   <http://work.example.org/alice/> .
_:b  foaf:name       "Bob" .
_:b  foaf:mbox       <mailto:bob@work.example> .

ASK  { ?x foaf:name  "Alice" }
```

(returns true)

DESCRIBE

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- Returns a single graph containing information about a resource or resources
- Nature of description left unspecified
  - Blank node closure
  - Concise Bounded Description
- Structure of returned data is determined by the SPARQL query processor

Describe Examples

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
DESCRIBE <http://example.org/>

DESCRIBE ?x
WHERE    { ?x foaf:name "Alice" }

DESCRIBE ?x ?y <http://example.org/>
WHERE    {?x foaf:knows ?y}
```

Concise Bounded Description

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- Algorithm for extracting a subgraph from an RDF graph, given a resource of interest
- Produces a graph where the object nodes are either URI references, literals, or blank nodes not serving as the subject of any statement in the graph
- Asymmetric view of a resource – follows only outgoing links

Concise Bounded Description Algorithm

UNIVERSITY OF Southampton
School of Electronics and Computer Science

1. Include in the subgraph all statements in the source graph where the subject of the statement is the starting node;
2. Recursively, for all statements identified in the subgraph thus far having a blank node object, include in the subgraph all statements in the source graph where the subject of the statement is the blank node in question and which are not already included in the subgraph.
3. Recursively, for all statements included in the subgraph thus far, for all reifications of each statement in the source graph, include the concise bounded description beginning from the rdf:Statement node of each reification.

Concise Bounded Description Example

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
_:a  foaf:name       "Alice" .
_:a  foaf:homepage   <http://work.example.org/alice/> .
_:b  foaf:name       "Bob" .
_:b  foaf:mbox       <mailto:bob@work.example> .
<http://example.org/> dc:creator _:a .
<http://example.org/> dc:title “Example Inc. Website” .
<http://example.org/> dc:date “2005-05-23” .
```

Concise Bounded Description Example

UNIVERSITY OF Southampton
School of Electronics and Computer Science

DESCRIBE <http://example.org/>

```
_:a  foaf:name       "Alice" .
_:a  foaf:homepage   <http://work.example.org/alice/> .
_:b  foaf:name       "Bob" .
_:b  foaf:mbox       <mailto:bob@work.example> .
<http://example.org/> dc:creator _:a .
<http://example.org/> dc:title “Example Inc. Website” .
<http://example.org/> dc:date “2005-05-23” .
```

SPARQL Protocol

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- SPARQL Query Language is protocol-independent
- SPARQL Protocol defines the method of interaction with a SPARQL processor
- Two standard bindings:
  - HTTP
  - SOAP
- Standard XML results format

SPARQL over HTTP

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- Two methods of submitting a query to a processor:
  - HTTP GET
    - Query encoded as HTTP request URI
    - Limitation on query length
  - HTTP POST
    - Query encoded in HTTP request body
    - No limitation on query length

HTTP GET

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
GET /sparql/?query=EncodedQuery   HTTP/1.1
Host: www.example.org
User-agent: my-sparql-client/0.1
```

SPARQL Results Format

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
<?xml version="1.0"?>
<sparql xmlns="http://www.w3.org/2005/sparql-results#">
        <head>
                <variable name="book"/>
                <variable name="who"/>
        </head>
        <results distinct="false" ordered="false”>
                <result>
                        <binding name="book">
                                <uri>http://www.example/book/book5</uri>
                        </binding>
                        <binding name="who">
                                <bnode>r29392923r2922</bnode>
                        </binding>
                </result>
        …
```

SPARQL Results Format

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- CONSTRUCT and DESCRIBE queries return RDF data
  - application/rdf+xml or similar

- ASK queries return a boolean value:

```
<?xml version="1.0"?>
<sparql xmlns="http://www.w3.org/2005/sparql-results#">
        <head></head>
        <boolean>true</boolean>
</sparql>
```

SPARQL Update

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- SPARQL only provides a means for querying a database
- Other capabilities required by many applications:
  - Create
  - (Read)
  - Update
  - Delete
- SPARQL Update (SPARUL) is a proposed language that fills this gap

SPARUL Commands

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- MODIFY, DELETE, INSERT
  - Delete triples from a graph and add new triples
- LOAD
  - Loads all triple from a remote graph into the specified graph (or default graph if none specified)
- CLEAR
  - Removes all triples from the specified graph (or default graph if none specified)

INSERT Example

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
INSERT {
        <http://example/book3>      dc:title    "A new book" ;
                                    dc:creator  "A.N.Other" .
}
```

- Inserts triples into default graph

MODIFY Example

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
MODIFY <http://example/bookStore>
DELETE {
        <http://example/book3>  dc:title
                "Fundamentals of Compiler Desing”
}
INSERT {
        <http://example/book3>  dc:title
                "Fundamentals of Compiler Design”
}
```

DELETE Example

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
DELETE {
        ?book ?p ?v .
} WHERE {
        ?book dc:date ?date .
        FILTER ( ?date < "2000-01-01T00:00:00"^^xsd:dateTime )
        ?book ?p ?v .
}
```