SPARQL 1.1 Update

INSERT DATA:

```
INSERT DATA {
  GRAPH <http://example.org/g> {
    :Bob rdf:type :Person .
  }
}
```

DELETE DATA:

```
DELETE DATA {
  GRAPH <http://example.org/g> {
    :Bob rdf:type :Person .
  }
}
```

DELETE/INSERT with WHERE:

```
DELETE { ... }
INSERT { ... }
WHERE  { ... }
```

Complex Example

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

SELECT ?film (COUNT(?actor) AS ?numActors)
       (GROUP_CONCAT(?actorLabel; SEPARATOR=", ") AS ?actors)
WHERE {
  ?film a dbo:Film ;
        dbo:starring ?actor .
  OPTIONAL {
    ?actor rdfs:label ?actorLabel .
    FILTER(lang(?actorLabel) = "en")
  }
}
GROUP BY ?film
HAVING(COUNT(?actor) > 3)
ORDER BY DESC(?numActors)
LIMIT 5
```

SPARQL: Additional Clauses & Features

Some less-known SPARQL clauses and features:

- **VALUES:** Inline data tables to parameterize queries
- **BIND:** Assign an expression result to a new variable
- **GRAPH:** Query named graphs in a dataset
- **SERVICE:** Federated queries to remote SPARQL endpoints
- **MINUS vs. FILTER NOT EXISTS:** Both can exclude matches, but differ in scoping
- **Group Concatenation:** `GROUP_CONCAT(?var; SEPARATOR=";")`
- **Property Paths with +, *, ? :** Repeat or optional steps in the graph

Remember: Not all endpoints implement every feature fully.

SPARQL: VALUES & BIND Examples

VALUES Example: Define a small data table inline.

```
VALUES ?city { <http://example.org/CityA>
               <http://example.org/CityB> }

SELECT ?city ?pop
WHERE {
  ?city ex:population ?pop .
}
```

BIND Example: Compute an expression once, store in new var.

```
SELECT ?city ?pop ?density
WHERE {
  ?city ex:population ?pop ;
        ex:area ?area .
  BIND(?pop / ?area AS ?density)
}
```

SPARQL: GRAPH & Named Datasets

GRAPH Clause:

- Many triplestores split data into "named graphs."
- `GRAPH <IRI> {...}` to query a specific subgraph.
- Example: `GRAPH <http://example.org/graph1> { ?s ?p ?o }`.

Default Graph vs. Named Graphs:

- Some queries only see the "default graph."
- Others explicitly reference a named graph with `GRAPH`.

Why Named Graphs?

- Keep data sets separate in one store (provenance, ownership).
- Selectively query relevant subsets.

Additional Resources

- **W3C SPARQL 1.1**: sparql11-query and sparql11-update
- **Learning SPARQL (2nd Ed.) by Bob DuCharme**
- **DBpedia:** dbpedia.org/sparql
- **Wikidata:** query.wikidata.org
- **EU Publications / Europeana** for open data

DBpedia: Architecture & Extraction

What is DBpedia?

- A community project extracting structured data from Wikipedia.
- Publishes data as RDF, forming a large "DBpedia knowledge graph."
- Relies on "infoboxes" (the side panels in Wikipedia) to gather facts.

Extraction Framework:

- Periodically runs scripts to parse Wikipedia's "infobox" templates.
- Produces RDF triples using DBpedia Ontology or external vocabularies.
- Multi-language: not only English Wikipedia but also 100+ language versions.

Result:

- Over 6 million resources (English DBpedia alone).
- Linked to many external data sets (Geonames, MusicBrainz, etc.).

DBpedia Ontology

Ontology Highlights:

- Classes: `dbo:Person`, `dbo:Place`, `dbo:Organisation`, etc.
- Properties: `dbo:birthDate`, `dbo:capital`, `dbo:populationTotal`, etc.
- Evolved from manual mapping of Wikipedia infobox templates.

Namespace:

- `http://dbpedia.org/ontology/` (often aliased as `dbo:`)
- `http://dbpedia.org/property/` for raw properties from Wikipedia
- `http://dbpedia.org/resource/` for actual "subjects"

Hierarchy:

- `dbo:Place` has sub-classes like `dbo:Country`, `dbo:City`, etc.
- `dbo:Event` has sub-classes like `dbo:SportsEvent`, `dbo:SocietalEvent`.

Accessing DBpedia

Public SPARQL Endpoint:

- `https://dbpedia.org/sparql`
- Typically uses Virtuoso UI with a query box.
- Rate-limited; heavy queries might be throttled.

Downloads & Dumps:

- `http://wiki.dbpedia.org/Downloads`
- Offers RDF dumps (Turtle, N-Triples) for offline use.

Faceted Browser:

- `http://dbpedia.org/fct/`
- Explore data via a faceted search interface (like a "point-and-click").

DBpedia Lookup:

- `https://wiki.dbpedia.org/projects/dbpedia-lookup`
- Autocomplete service for typed entity names.

DBpedia: Extended Architecture & Background

Origins:

- Launched in 2007, derived from Wikipedia "infoboxes" (those side panels).
- Maintained by researchers at institutions like the University of Mannheim, Leipzig, etc.

Multi-language DBpedia:

- Separate extracts for each major Wikipedia language edition (English, German, French, etc.).
- Each language can have distinct classes/properties from local "infobox" templates.
- Interconnected via owl:sameAs across different language chapters.

DBpedia Live:

- A near real-time version that updates as Wikipedia changes.
- Built with a continuous extraction pipeline.
- Not guaranteed always stable or comprehensive, but more current.

DBpedia Extraction Process (Detailed)

Stages of Extraction:

1. *Download Wikipedia dumps* (XML) from `https://dumps.wikimedia.org/`.
2. *Parse Infoboxes* using DBpedia's custom mapping-based extractor.
3. *Generate RDF Triples*:
   - Use the DBpedia Ontology for standardized classes & properties.
   - Fallback to "raw" properties when a mapping is missing.
4. *Clean & Post-processing*: remove duplicates, unify data types, etc.

Mapping-based Extraction:

- Community-driven "Mapping Wiki" that defines how each infobox template maps to DBpedia classes/properties.
- E.g. `Infobox person` in Wikipedia → `dbo:Person`, `dbo:birthDate`, etc.

Resulting Data:

- The "en" dump alone yields millions of resources with structured RDF

DBpedia Ontology Mappings

Ontology Growth:

- Initially, a small subset of classes & properties mapped from common infobox templates.
- Grown over time to cover hundreds of classes (dbo:Film, dbo:Country, dbo:City, etc.).

Mapping Wiki:

- `https://mapping.dbpedia.org/`
- Volunteers can add or edit mappings for new infobox types.
- Aim: unify the wide variety of Wikipedia templates into a consistent structure.

Example Mappings:

- `Infobox book` → dbo:Book, with fields like "author," "publisher," etc.
- `Infobox settlement` → dbo:Place or dbo:PopulatedPlace.

DBpedia: Data Types & Language Tags

Data Types:

- `xsd:date`, `xsd:integer`, `xsd:double`, etc.
- Sometimes Wikipedia stores numeric data as text; DBpedia tries to convert them.
- Some fields remain strings if no consistent format is found.

Language Tags:

- Often see `rdfs:label` or `dbo:abstract` with `@en`, `@de`, `@fr` etc.
- DBpedia's best-known property: `dbo:abstract` is the "short summary" from Wikipedia, available in many languages.

Implication for Queries:

- Always `FILTER(lang(?abstract) = "en")` (or whichever language) to get the correct version.
- For numeric fields, confirm data type is numeric, or do a cast if needed.

DBpedia Linking & External Data

Links to Other Datasets:

- DBpedia is central in the "Linked Open Data Cloud."
- Often uses `owl:sameAs` or `schema:sameAs` to link a resource to external IDs (e.g., Wikidata, VIAF, MusicBrainz).

Examples:

- `dbr:Paris` might link to `<http://www.geonames.org/2988507>` via sameAs.
- People often link to VIAF, GND, or ISNI IDs.

Why Linking Matters:

- It merges knowledge from multiple sources: e.g., DBpedia + Wikidata + Geonames.
- Helps build richer knowledge graphs & cross-reference info.

DBpedia Quality & Future

Data Quality:

- Varies by Wikipedia article & template completeness.
- Some infobox fields are missing or incorrectly mapped.
- Multi-language chapters often have different coverage rates.

Limitations:

- No strong "official editorial board." It's community-driven, so some data might be outdated or incomplete.
- Wikipedia vandalism can appear in DBpedia until corrected upstream.

Future Plans:

- DBpedia **Live** expansions, more frequent updates.
- Better alignment with **Wikidata**, merging complementary data.

DBpedia Example 1: Filter by Language

Goal: Retrieve the English abstract of Slovakia.

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbr: <http://dbpedia.org/resource/>

SELECT ?abstract
WHERE {
  dbr:Slovakia dbo:abstract ?abstract .
  FILTER (lang(?abstract) = "en")
}
```

DBpedia Example 2: List Books by an Author

Goal: Return 5 books (`dbo:Book`) by George R. R. Martin.

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbr: <http://dbpedia.org/resource/>

SELECT ?book
WHERE {
  ?book a dbo:Book ;
        dbo:author dbr:George_R._R._Martin .
}
LIMIT 5
```

DBpedia Example 3: Using Aggregation

Goal: List Spanish cities, counting references to them.

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbr: <http://dbpedia.org/resource/>

SELECT ?city (COUNT(*) AS ?countMentions)
WHERE {
  ?page dbo:location ?city .
  ?city dbo:country dbr:Spain .
}
GROUP BY ?city
ORDER BY DESC(?countMentions)
LIMIT 5
```

DBpedia Example 4: OPTIONAL + FILTER + Language

Goal: Retrieve films with English abstracts, plus an optional release date.

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbr: <http://dbpedia.org/resource/>

SELECT ?film ?abstract ?releaseDate
WHERE {
  ?film a dbo:Film ;
        dbo:abstract ?abstract .
  FILTER(lang(?abstract) = "en")

  OPTIONAL {
    ?film dbo:releaseDate ?releaseDate .
  }
}
LIMIT 5
```

DBpedia Exercises

Exercise 1: List 5 U.S. Cities + Population

- Hints:
  - `dbo:country dbr:United_States`
  - `dbo:populationTotal`
  - `LIMIT 5`

Exercise 2: Writers Born Before 1900

- Filter on `dbo:birthDate < "1900-01-01"^^xsd:date`.
- Possibly ensure `rdf:type dbo:Writer`.

Exercise 3: Count Actors per Film (One Specific Director)

- Use `dbo:director` to find a director.
- Then see who starred in those films, grouping by the film.
- Show how many distinct actors in each film.

SPARQL: ASK

- Used to get true/false reply for a query
- true is returned if there exists a possible solution (triple) for the query

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbr: <http://dbpedia.org/resource/>

ASK {
dbr:Bratislava dbo:type dbr:Capital_city.
}
```

SPARQL: ASK with FILTER

(dbo:Book) by George R. R. Martin.

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbr: <http://dbpedia.org/resource/>

ASK
{
dbr:Danube dbo:length ?dlzkaDunaj .
dbr:Váh dbo:length ?dlzkaVah .
FILTER(?dlzkaDunaj > ?dlzkaVah) .
}
```

SPARQL: CONSTRUCT

- Used to generate a directed graph from RDF triples
- The triples we want to see are specified in the CONSTRUCT clause

```
CONSTRUCT { ?s ?p ?o }
```

SPARQL: CONSTRUCT

- There are several formats for the serialized graph
- XML/RDF, Turtle, TriG...
- Service to generate graph image from serialized string: https://www.ldf.fi/service/rdf-grapher

SPARQL: CONSTRUCT example

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbr: <http://dbpedia.org/resource/>
CONSTRUCT {
?book a dbo:Book .
?book dbo:author dbr:George_R._R._Martin .
?book dbo:author dbr:Voltaire .
}
WHERE {
{
?book a dbo:Book .
?book dbo:author dbr:George_R._R._Martin .
}
UNION
{
?book a dbo:Book .
?book dbo:author dbr:Voltaire .
}
```

EU Court Data: Introduction

Key Sources:

- EU Publications Office: Manages and publishes official documents from EU institutions (legislation, case-law, etc.).
- Court of Justice of the European Union (CJEU) data: judgments, orders, opinions, etc.
- Hosted in the CELLAR repository with RDF describing legal documents.

Why RDF?

- Allows interoperability across EU legal data.
- Facilitates advanced queries (**SPARQL**) linking legislations, court decisions, references.

The Common Data Model (CDM)

Definition:

- A conceptual model used by the EU Publications Office to describe legislative documents, case-law, etc.
- Based on FRBR (Functional Requirements for Bibliographic Records) – "Work," "Expression," "Manifestation," etc.
- Extended for legal domain: references to treaties, directives, national case-law, etc.

Key Classes:

- `cdm:CaseLawWork`, `cdm:order_cjeu`, `cdm:Judgement`, etc.
- `cdm:Work` for abstract documents, `cdm:Expression` for language versions.

IRIs & Vocabulary:

- Many properties start with `cdm:`, e.g. `cdm:work_date_document`, `cdm:work_created_by_agent`.

FRBR Inspiration in CDM

FRBR Basics:

- *Work*: a distinct intellectual or artistic creation.
- *Expression*: a translation or version of the work.
- *Manifestation*: the physical or digital embodiment (like a PDF).

In EU Court Data:

- `cdm:CaseLawWork` can be the abstract "case."
- Different language expressions or forms: `cdm:Expression` or `cdm:Manifestation`
- This layered approach allows advanced queries about content vs. textual versions.

Benefit:

- Distinguish the "idea" of a judgment from its many translations or official forms.
- Link multiple references or expressions under one conceptual work.

EU Court Data: Properties & Agents

Properties:

- `cdm:work_date_document`: official date of the document.
- `cdm:work_created_by_agent`: which court or chamber created/issued it.
- `cdm:work_is_about_concept`: references to directives, treaties, subjects.

Agents (Courts, Chambers):

- `cdm:work_created_by_agent` could point to URIs for the Court of Justice (CJ), General Court, etc.
- Additional detail: `cdm:case-law_delivered_by_court-formation` for specific chambers.
- http://publications.europa.eu/resource/authority/court/CJ
- http://publications.europa.eu/resource/authority/formjug/CHAMB_01_

EU Publications Office

Role:

- Central point for publishing and disseminating EU official documents.
- Maintains the CELLAR repository, storing metadata in RDF.

SPARQL Endpoint:

- Typically a Virtuoso-based endpoint (URIs can vary).
- Provides read-only queries for legislation, court decisions, etc.

CELLAR Infrastructure:

- Contains multiple named graphs for different document types.
- Often uses CDM classes: `cdm:CaseLawWork`, `cdm:order_cjeu`, `cdm:Regulation`, `cdm:Directive`, etc.

Linking Legal Documents

Why Link?

- Helps track how a case interprets or applies certain EU laws.
- Facilitates queries across multiple levels: from a directive to all cases referencing it.

Cross-Dataset Potential:

- Could link to external resources (e.g. DBpedia or Eur-Lex URIs).
- Encourages a "web of legal data."

EU Court Data: Potential Queries

Examples of questions you can answer:

- "Show me all orders from the Court of Justice after 2010."
- "List cases referencing a particular directive (CELEX number)."
- "Count how many orders were created in a specific year or by a certain chamber."

Approaches:

- Simple filters (date range, agent/court).
- Aggregations (group by year, group by reference).
- Subqueries to refine or rank results.

Endpoint Reality:

- Some data might be incomplete or limited in the public endpoint.
- If a query yields no results, try removing or loosening filters.

EU Court Data Example 1: Basic Retrieval

Goal: Retrieve all objects of a type order_cjeu.

```
PREFIX cdm: <http://publications.europa.eu/ontology/cdm#>

  SELECT DISTINCT ?wc
  WHERE {
    ?wc a cdm:order_cjeu
    }
```

EU Court Data Example 2: All properties

Goal: All properties connected to object of a type order_cjeu.

```
PREFIX cdm: <http://publications.europa.eu/ontology/cdm#>

  SELECT DISTINCT ?p
  WHERE {
    ?wc a cdm:order_cjeu .
    ?wc ?p ?o
  }
```

EU Court Data Example 3: Aggregation by Year

Goal: Count how many cases appear per year (2019–2023).

```
PREFIX cdm: <http://publications.europa.eu/ontology/cdm#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

SELECT (YEAR(?date) AS ?yr) (COUNT(?case) AS ?caseCount)
WHERE {
  ?case a
        cdm:order_cjeu ;
         cdm:work_date_document ?date .
  FILTER(?date >= "2019-01-01"^^xsd:date && ?date < "2024-01-01"^^xs
}
GROUP BY YEAR(?date)
ORDER BY ?yr
```

EU Court Data Example 4: Cases by Agent (Court)

Goal: Retrieve 10 cases decided by specific chamber.

```
PREFIX cdm: <http://publications.europa.eu/ontology/cdm#>

SELECT ?case
WHERE {
  ?case a
        cdm:order_cjeu ;
         cdm:case-law_delivered_by_court-formation
        <http://publications.europa.eu/resource/authority/formjug/CH
}
LIMIT 10
```

EU Court Data Exercises

Exercise 1: List all CELEX numbers

- List all distinct CELEX numbers of documents of type cdm:order_cjeu.

Exercise 2: Cases after 2010, Sorted by Date

- Find all cdm:order_cjeu documents created after the year 2010.

Exercise 3: Decisions with their country of origin

- Find all cdm:order_cjeu documents and the country of their origin