(Optional) Wikidata as an Example of the Semantic Web

A Collaborative Knowledge Base

What is Wikidata?

Wikidata is a free, collaborative, multilingual knowledge base that can be read and edited by both humans and machines.

- Think of it as Wikipedia's structured data counterpart - while Wikipedia provides articles for humans to read, Wikidata provides structured data that computers can process.

One-Line Summary

Wikidata = A real-world Semantic Web knowledge graph

- Public
- RDF-based
- Queryable with SPARQL
- Linked to external datasets

Why Wikidata Fits the Semantic Web

Semantic Web Principle	Wikidata Implementation
URI identifiers	Q-IDs, P-IDs
RDF triples	Statement structure
Ontology/schema	Classes & properties
Linked Data	External database links
SPARQL queries	Query Service

Example: Triple Representation

Natural form:

- Douglas Adams — instance of — Human
- Douglas Adams — occupation — Writer

RDF form:

```
wd:Q42  wdt:P31  wd:Q5 .
wd:Q42  wdt:P106 wd:Q36180 .
```

Same Subject–Predicate–Object model.

SPARQL Query Example

```
SELECT ?person WHERE {
  ?person wdt:P31 wd:Q5 .
}
```

Meaning:

Find all humans.

Ontology Role in Wikidata

Wikidata provides lightweight ontology:

- Class hierarchy
- Property definitions
- Type constraints
- Domain/range hints

Not full OWL, but semantically structured.

Linked Open Data Connections

Wikidata links to:

- Wikipedia
- DBpedia
- VIAF
- Library of Congress
- MusicBrainz

Forms part of the Linked Open Data cloud.

Semantic Web Stack Mapping

Layer Wikidata Equivalent

RDF Triple storage

RDFS Class/property hierarchy

OWL-lite Some constraints

SPARQL Query endpoint

Linked Data External IDs

Why Wikidata Matters

Before Wikidata

```
Wikipedia Article (Human-readable text):
"Douglas Adams was an English writer, born in 1952.
He wrote The Hitchhiker's Guide to the Galaxy."
```

Problem: Computers can't easily extract or query this information.

After Wikidata

```
Structured Data (Machine-readable):
Douglas Adams (Q42)
  - instance of: human (Q5)
  - occupation: writer (Q36180)
  - date of birth: 1952-03-11
  - country: United Kingdom (Q145)
  - notable work: The Hitchhiker's Guide to the Galaxy (Q3107329)
```

Benefit: Computers can query "Show me all British writers born in 1952" instantly!

Key Features:

- **Free and Open**: Anyone can access and contribute
- **Multilingual**: Labels and descriptions in 300+ languages
- **Machine-readable**: Structured data using semantic web standards
- **Collaborative**: Community-driven like Wikipedia
- **Linked Data**: Connects to other knowledge bases

Core Concepts

1. Entities

Every "thing" in Wikidata has a unique identifier called a **Q-number**.

Examples:

- Q42 = Douglas Adams
- Q5 = Human
- Q145 = United Kingdom
- Q36180 = Writer
- Q1299 = The Beatles

Why Q-numbers?

- Language-independent (Q42 is the same in all languages)
- Unique and permanent (won't change even if the name changes)
- Easy for computers to process

2. Properties

Properties define relationships between entities. They use **P-numbers**.

Common Properties:

- P31 = instance of (what type of thing is it?)
- P106 = occupation (what does this person do?)
- P569 = date of birth (when were they born?)
- P570 = date of death (when did they die?)
- P27 = country of citizenship
- P50 = author (of a book)
- P800 = notable work

3. Statements (Triples)

Knowledge in Wikidata is expressed as **statements** in the form:

```
Subject – Property – Value
```

This is called the **RDF triple** structure:

- **Subject**: What we're talking about
- **Predicate**: The relationship/property
- **Object**: The value

Example:

Subject	Property	Value
Douglas Adams (Q42)	occupation (P106)	writer (Q36180)
Douglas Adams (Q42)	date of birth (P569)	1952-03-11
Douglas Adams (Q42)	country (P27)	United Kingdom (Q145)

Simple Example: Modeling a Sentence

Let's model: **"Douglas Adams was an English writer, born in 1952."**

Step 1: Identify Entities

- Douglas Adams → Q42
- Writer → Q36180
- United Kingdom → Q145

Step 2: Identify Properties

- "was a writer" → P106 (occupation)
- "born in 1952" → P569 (date of birth)
- "English" → P27 (country of citizenship)

Step 3: Create Statements

```
Q42 --P31--> Q5                (Douglas Adams is a human)
Q42 --P106--> Q36180           (Douglas Adams's occupation is writer)
Q42 --P569--> "1952-03-11"     (Douglas Adams was born on March 11, 1952)
Q42 --P27--> Q145              (Douglas Adams's country is United Kingdom)
```

Visual Representation

```
          Douglas Adams (Q42)
               /    |    \
              /     |     \
             /      |      \
    instance of  occupation  born
           |         |         |
       Human     Writer    1952-03-11
        (Q5)   (Q36180)
```

Querying Wikidata with SPARQL

SPARQL is the query language for RDF data (like SQL for databases).

Example 1: Basic Query

Question: What is Douglas Adams's birth date?

```
SELECT ?birthDate
WHERE {
  wd:Q42 wdt:P569 ?birthDate .
}
```

Result: `1952-03-11`

Explanation:

- `wd:Q42` = Douglas Adams
- `wdt:P569` = date of birth property
- `?birthDate` = variable to store the result

Example 2: Get Labels

Question: What are Douglas Adams's occupations (in English)?

```
SELECT ?occupation ?occupationLabel
WHERE {
  wd:Q42 wdt:P106 ?occupation .

  SERVICE wikibase:label {
    bd:serviceParam wikibase:language "en" .
  }
}
```

- We can ask the service to add labels in English for any entity with "wikibase:label".

- As a result, instead of getting just Q-numbers, we get human-readable labels.

Result:

```
writer
novelist
screenwriter
science fiction writer
```

Example 3: Find Related People

Question: Who are other British science fiction writers?

```
SELECT ?person ?personLabel
WHERE {
  ?person wdt:P106 wd:Q36180 .        # occupation: writer
  ?person wdt:P136 wd:Q24925 .        # genre: science fiction
  ?person wdt:P27 wd:Q145 .           # country: United Kingdom

  SERVICE wikibase:label {
    bd:serviceParam wikibase:language "en" .
  }
}
LIMIT 10
```

Hands-On: Exploring Wikidata

Try These URLs:

1. **Douglas Adams Page**: https://www.wikidata.org/wiki/Q42
    - See all statements about him
    - Notice the Q-numbers and P-numbers
2. **Query Service**: https://query.wikidata.org/
    - Try example queries
    - Build your own queries
3. **Search for Entities**: https://www.wikidata.org/
    - Search for your favorite person, place, or thing
    - Note their Q-number

Real-World Applications

1. Digital Assistants

"Alexa, when was Douglas Adams born?"
→ Queries Wikidata: `Q42 –P569–> ?`

2. Knowledge Graphs

Google's Knowledge Panel uses Wikidata to show information boxes.

3. Data Integration

Connect data from different sources using common Wikidata identifiers.

4. Multilingual Applications

One query returns information in any language.

5. Research

Scientists use Wikidata for bibliometrics, social network analysis, etc.

Comparison: Traditional Database vs. Wikidata

Traditional Database (Relational)

```
TABLE: people
id  | name            | birth_date | occupation | country
----|-----------------|------------|------------|--------
1   | Douglas Adams   | 1952-03-11 | writer     | UK

SELECT name, birth_date
FROM people
WHERE occupation = 'writer' AND country = 'UK';
```

Limitations:

- Fixed schema (can't easily add new properties)
- Not linked to other data
- Language-dependent

Wikidata (Graph)

```
SELECT ?person ?personLabel ?birthDate
WHERE {
  ?person wdt:P106 wd:Q36180 .    # occupation: writer
  ?person wdt:P27 wd:Q145 .       # country: UK
  ?person wdt:P569 ?birthDate .   # get birth date

  SERVICE wikibase:label { bd:serviceParam wikibase:language "en" . }
}
```

Benefits:

- Flexible schema (easily add new properties)
- Linked to millions of entities
- Multilingual by design
- Semantic relationships

Key Takeaways

1. **Wikidata = Structured Wikipedia**: Machine-readable knowledge base
2. **Q-numbers = Entities**: Things in the world (people, places, concepts)
3. **P-numbers = Properties**: Relationships between entities
4. **RDF Triples**: Subject-Predicate-Object structure
5. **SPARQL**: Query language for semantic data
6. **Linked Data**: Everything is connected through URIs

Common Wikidata Properties

Property	P-number	Use	Example
instance of	P31	Type of entity	Q42 P31 Q5 (human)
occupation	P106	Person's job	Q42 P106 Q36180 (writer)
date of birth	P569	Birth date	Q42 P569 1952-03-11
date of death	P570	Death date	Q42 P570 2001-05-11
country	P27	Citizenship	Q42 P27 Q145 (UK)
author	P50	Book author	Q3107329 P50 Q42
notable work	P800	Important creation	Q42 P800 Q3107329
educated at	P69	University	Q42 P69 Q35794

Practice Exercise

Model this sentence in Wikidata format:

"J.K. Rowling is a British writer who wrote Harry Potter, born in 1965."

Entities:

- J.K. Rowling = Q34660
- Writer = Q36180
- United Kingdom = Q145
- Harry Potter series = Q8337

Statements:

```
Q34660 --P31--> Q5                  (J.K. Rowling is a human)
Q34660 --P106--> Q36180             (occupation: writer)
Q34660 --P27--> Q145                (country: United Kingdom)
Q34660 --P569--> "1965-07-31"       (birth date: July 31, 1965)
Q34660 --P800--> Q8337              (notable work: Harry Potter)
```

SPARQL Query:

```
SELECT ?property ?propertyLabel ?value ?valueLabel
WHERE {
  wd:Q34660 ?property ?value .
  FILTER(?property IN (wdt:P31, wdt:P106, wdt:P27, wdt:P569, wdt:P800))

  SERVICE wikibase:label {
    bd:serviceParam wikibase:language "en" .
  }
}
```

Next Steps

1. **Explore**: Visit https://www.wikidata.org/wiki/Q42 and click around
2. **Query**: Try queries at https://query.wikidata.org/
3. **Code**: Run the Python examples in `/code/wikidata/`
4. **Create**: Add missing data to Wikidata (create an account!)

Resources

- **Wikidata Homepage**: https://www.wikidata.org/
- **SPARQL Tutorial**: https://www.wikidata.org/wiki/Wikidata:SPARQL_tutorial
- **Query Service**: https://query.wikidata.org/
- **API Documentation**: https://www.wikidata.org/w/api.php
- **Code Examples**: See `/code/wikidata/` directory

Remember: Wikidata is about making knowledge machine-readable while keeping it human-editable. It bridges the gap between how humans think (natural language) and how computers process information (structured data).