



# Practical Knowledge Graph Example



Protege,  
Stardog and  
Peeps

# Today's exercise

1. Look at a simple ontology for information about people and their relations in Protégé
2. Look at some instance data in Protégé
3. Run the DL and rule reasoner in Protégé
4. Load the ontology and data into Stardog
5. Browse and query the resulting knowledge graph in Stardog

# Preliminaries

- On your own computer (Windows, Mac, Linux)
  - Download and install Protégé
  - Download, install and configure the community edition of Stardog 5
  - Clone the 691 peeps repository

# Peeps files

- The peeps repo has five files
  - **README.md**
  - **catalog-v001.xml** – protégé config file
  - **load\_peeps.sh** – bash script to load peeps into stardog
  - **mypeeps.ttl** – data encoded using peeps ontology
  - **peeps.ttl** – the peeps ontology
  - **prefixes.ttl** – list of prefixes, used by stardog’s query component

# Separate ontology and data?

- An ontology is a knowledge graph schema
  - peeps:Man owl:disjointWith peeps:Woman .
- We talk about populating it with instance data
  - :janeDoe a peeps:Woman; foaf:givenName “Jane” .
- Good practice for real applications is to keep the ontology and data separate
  - i.e., in different files
- Hence, peeps.ttl and mypeeps.ttl

# Why separate ontology and data?

- It really depends on the usecase
- Some facts are part of an ontology if they're important, unchanging knowledge
- Maybe the ontology is a one-off, and will never be used with any other data
- Maybe you added data while developing the ontology for testing and debugging
- But many ontologies are intended for reuse or to represent datasets that change frequently

# Namespaces

- Promoting reuse also entails giving the ontology and a knowledge graph that uses it with different namespaces
- Namespace = uri = unique identifier
- Example
  - <http://dbpedia.org/resource/>
  - <http://dbpedia.org/ontology/>
- BTW, lookup prefixes at <http://prefix.cc>
- Ideally, the uris are ones you control and no one else will use

# Namespace best practice

- Ideally, the namespace should resolve to a file containing the ontology or data
  - Maybe not the data if it's big or proprietary
- Enables other ontologies to **import and use** yours just from its URI
- If you don't control a long-lived URI ...
  - You might use a file on github
  - You might use purl to create a “permanent url” that redirects to the current location

# Peeps.ttl in Protégé

The screenshot shows the Protégé 5.5 interface with the following details:

- Title Bar:** peeps.ttl (https://raw.githubusercontent.com/UMBC-CMSC-491-691-F18-Knowledge-Graphs/peeps/master/peeps.ttl) : [/Users/finin/Desktop...]
- Toolbars:** Data Properties, Annotation Properties, Individuals by class, DL Query, SWRLTab.
- Ontology Header:** Ontology IRI: https://raw.githubusercontent.com/UMBC-CMSC-491-691-F18-Knowledge-Graphs/peeps/master/peeps.ttl
- Annotations Tab:** Active Ontology, Entities, Object Properties.
- Annotations Panel:** rdfs:label, An example ontology for people created in Protege OWL 5.5".
- OWL/XML rendering:** OWL functional syntax rendering.
- Imports:** Ontology imports, General axioms, RDF/XML rendering.
- Imported ontologies:** Direct Imports, Indirect Imports.
- Rules Panel:** Rules:
  - hasParent(?p1, ?p2), Woman(?p2) -> hasMother(?p1, ?p2)
  - hasParent(?p1, ?p2) -> youngerThan(?p1, ?p2)
  - hasAge(?p1, ?a1), hasAge(?p2, ?a2), lessThan(?a1, ?a2) -> youngerThan(?p1, ?p2)
- Footer:** Git: master, To use the reasoner click Reasoner > Start reasoner, Show Inferences checked.

# Mypeeps.ttl

The screenshot shows the Protégé ontology editor interface for the file "mypeeps.ttl".

**Header:**

- Active Ontology: mypeeps.ttl
- Annotations, Selected entailments, Rules, Ontology prefixes
- Ontology header:
  - Ontology IRI: <https://raw.githubusercontent.com/UMBC-CMSC-491-691-F18-Knowledge-Graphs/peeps/master/mypeeps.ttl>
  - Ontology Version IRI: e.g. <https://raw.githubusercontent.com/UMBC-CMSC-491-691-F18-Knowledge-Graphs/peeps/master/mypeeps.ttl/1.0.0>

**Annotations:** +

**Imports:**

- Ontology imports, General axioms, RDF/XML rendering, OWL/XML rendering, OWL functional syntax rendering
- Imported ontologies:
  - Direct Imports +
    - <<https://raw.githubusercontent.com/UMBC-CMSC-491-691-F18-Knowledge-Graphs/peeps/master/peeps.ttl>>
    - peeps.ttl
      - Ontology IRI: <<https://raw.githubusercontent.com/UMBC-CMSC-491-691-F18-Knowledge-Graphs/peeps/master/peeps.ttl>>
      - Location: <https://raw.githubusercontent.com/UMBC-CMSC-491-691-F18-Knowledge-Graphs/peeps/master/peeps.ttl>

**Bottom Status:**

- Git: master
- To use the reasoner click Reasoner > Start reasoner
- Show Inferences

# When to import an ontology

- In Protégé, we import an ontology if we want a reasoner to understand its vocabulary
- It does not add the ontology to the file that will be saved
- Plus: the knowledge may be important or essential in testing
- Minus: big ontologies may add a lot of useless data
- Here mypeeps.ttl imports peeps, but not foaf or schema

# Stardog Graph Platform

The screenshot shows the official website for Stardog. At the top, there's a navigation bar with a logo of a blue and orange dog, a search bar, and links for 'PLATFORM', 'CUSTOMERS', 'RESOURCES', 'ABOUT', and a prominent orange 'DOWNLOAD' button. Below the navigation is a large headline: 'The Knowledge Graph Platform for the Enterprise'. Underneath this, a sub-headline reads: 'With Stardog you can unify, query, search, and analyze all your data. Say goodbye to data silos forever.' A red 'Try Stardog' button is located on the left. On the right side of the page, there's a photograph of a brown and white dog wearing a white motorcycle helmet, sitting on a tiled floor. At the bottom right, there's a 'LEAVE A MESSAGE' button.

Stardog: The Enterprise Knowle X

https://www.stardog.com

PLATFORM CUSTOMERS RESOURCES ABOUT DOWNLOAD

The Knowledge Graph Platform for the Enterprise

With Stardog you can unify, query, search, and analyze all your data. Say goodbye to data silos forever.

Try Stardog

LEAVE A MESSAGE

# Stardog Graph Platform

- Stardog is easy to install and use, but rich in features
- It has a Web interface, good command-line tools and a Java API
- We'll look at how to
  - Load the peeps example files
  - Browse the results
  - Query the graph via the Web console

# Start Stardog

- This command will start Stardog listening to its default port (5820) and disable security

**stardog-admin server start --disable-security**

- Enter the URL <http://localhost:5820> to access the Web console

Use admin for both the user and password

Stardog Admin Web Console    X    +

localhost:5820

Stardog Admin    Databases    Security    Query Management    admin

**Server**

**Stardog Home:** /Users/finin/stardog    **Stardog Version:** 5.3.5

**Databases**

| Name | Status |
|------|--------|
|------|--------|

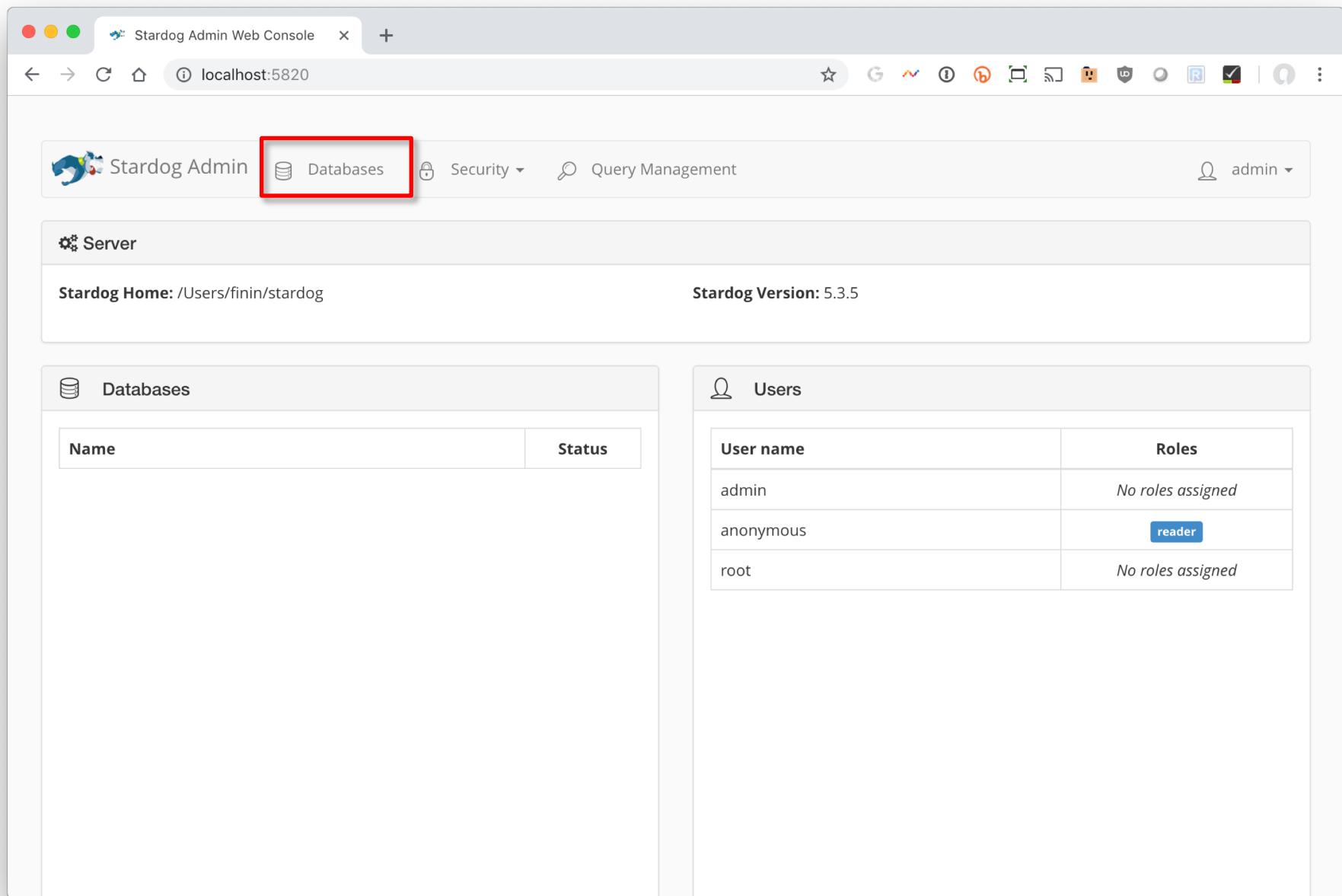
**Users**

| User name | Roles             |
|-----------|-------------------|
| admin     | No roles assigned |
| anonymous | reader            |
| root      | No roles assigned |

# Stardog script

- load\_peeps.sh is a bash script for loading the peeps data and ontology
- Use variations for other systems or shells
- Once loaded go to <http://localhost:5820/> to use Stardog's web interface

# Stardog's web interface



The screenshot shows the Stardog Admin Web Console interface running in a browser window. The title bar reads "Stardog Admin Web Console" and the address bar shows "localhost:5820". The top navigation bar includes links for "Databases", "Security", "Query Management", and a user dropdown set to "admin". A red box highlights the "Databases" link.

**Server**

**Stardog Home:** /Users/finin/stardog      **Stardog Version:** 5.3.5

**Databases**

| Name | Status |
|------|--------|
|      |        |

**Users**

| User name | Roles             |
|-----------|-------------------|
| admin     | No roles assigned |
| anonymous | reader            |
| root      | No roles assigned |

# Create a database

The screenshot shows the Stardog Admin Web Console interface. At the top, the title bar reads "Stardog Admin Web Console" and the address bar shows "localhost:5820/#/databases". The main header includes the Stardog logo, "Databases", "Security", "Query Management", and a user dropdown for "admin". Below the header is a search bar with a magnifying glass icon and the placeholder "Type to search by database name". A table follows, with columns labeled "Status", "Name", "Last Activity", "Features", and "Actions". In the top right corner of the table area, there is a green button labeled "New DB" which is highlighted with a red rectangular border.

# Name it mypeeps and accept the defaults

The screenshot shows the Stardog Admin Web Console interface. The title bar reads "Stardog Admin Web Console" and the address bar shows "localhost:5820/#/databases/new". The main content area is titled "New database". A descriptive text states: "This wizard will help you create a new Stardog database. It will go through all the options available for setting up a new DB. All the options are filled up with the default values. If all you need are the default options, just go ahead and click Finish, otherwise click Next." On the left, there are several configuration sections with toggle switches: "Database name" (set to "mypeeps"), "Database archetypes" (set to "None selected"), "Database online" (set to "ON"), "Strict Parsing" (set to "ON"), and "Preserve BNode identifiers" (set to "ON"). On the right, there is a list of "Database namespaces" with red "X" buttons to remove them: "rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#", "rdfs=http://www.w3.org/2000/01/rdf-schema#", "xsd=http://www.w3.org/2001/XMLSchema#", "owl=http://www.w3.org/2002/07/owl#", and "stardog=tag:stardog:api:". Below this list is a blue "Add namespace" button. At the bottom left is a "Finish" button with a checked checkbox. At the bottom right is a "Next" button with a circular arrow icon, which is highlighted with a red rectangle.

Stardog Admin Web Console

localhost:5820/#/databases/new

Stardog Admin Databases Security Query Management admin

## New database

This wizard will help you create a new Stardog database. It will go through all the options available for setting up a new DB. All the options are filled up with the default values. If all you need are the default options, just go ahead and click Finish, otherwise click Next.

**Database name**  
mypeeps

**Database archetypes**  
None selected

**Database online**  
ON

**Strict Parsing**  
ON

**Preserve BNode identifiers**  
ON

**Database namespaces**

- rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#
- rdfs=http://www.w3.org/2000/01/rdf-schema#
- xsd=http://www.w3.org/2001/XMLSchema#
- owl=http://www.w3.org/2002/07/owl#
- stardog=tag:stardog:api:

Add namespace

Finish

Next

Stardog Admin Web Console X +

localhost:5820/#/databases/mypeeps

Stardog Admin Databases Security Query Management admin

Database created!  
Database **mypeeps** was created, go to **mypeeps console** to add data

Query Browse Edit Optimize Drop ON

## mypeeps

Database

|                            |  |
|----------------------------|--|
| Database archetypes        |  |
| Database name              | mypeeps  |
| Database namespaces        | rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#<br>rdfs=http://www.w3.org/2000/01/rdf-schema#<br>xsd=http://www.w3.org/2001/XMLSchema#<br>owl=http://www.w3.org/2002/07/owl#<br>stardog=tag:stardog:api:<br>=http://api.stardog.com/ |
| Database creation time     | Tuesday, October 30th 2018, 10:48:07 pm -04:00   |
| database modification time | Tuesday, October 30th 2018, 10:48:08 pm -04:00   |

Strict Drawing

# Click on *data* and select *+Add*

The screenshot shows the Stardog Admin Console interface at [localhost:5820/mypeeps#!/webconsole](http://localhost:5820/mypeeps#!/webconsole). The main area displays 'Database Metadata' for the database 'mypeeps'. A red box highlights the 'Data' dropdown menu in the top right corner, which includes options: '+ Add', 'Remove', and 'Export'.

**Add the files**

- **peeps.ttl**
- **mypeeps.ttl**

**Database Name** mypeeps

**Database Online** Yes

**Last Modified**

**Index Type** Disk

**Index Literals Canonical** Yes

**Index Statistics Update Automatic** Yes

**Index Differential Enable Limit (Triples)** 10000

**Database Namespaces**

- `rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#`
- `rdfs=http://www.w3.org/2000/01/rdf-schema#`
- `xsd=http://www.w3.org/2001/XMLSchema#`
- `owl=http://www.w3.org/2002/07/owl#`
- `stardog=tag:stardog:api:`
- `=http://api.stardog.com/`

**Database Time Creation** 2018-10-30T23:01:14.437-04:00

**Index Size (Triples)** 0

**Index Persist** Yes

**Index Named Graphs** Yes

# Go to Browse to explore the graph

The screenshot shows a web browser window with the URL `localhost:5820/mypeeps#/schema`. The browser's top navigation bar includes standard icons for back, forward, refresh, and search, along with specific icons for the application, such as a star, a gear, and a question mark.

The main content area is titled "Schema Browser". On the left, there are two tabs: "Classes" (which is selected and highlighted in blue) and "Properties".

The central part of the screen displays a hierarchical tree structure under the heading "Instructions". The instructions state: "Click on the + / - icons to expand or collapse node elements in the Schema Tree." A red box highlights the "Browse" button in the top navigation bar.

The schema tree structure is as follows:

- Thing
  - Male person
    - Boy
    - Person
      - Person
        - Adult person
        - Minor
      - Person
        - Woman

# Go to Query to enter a SPARQL query

localhost:5820/mypeeps#!/query

Admin Console    Query    Browse    Data    Search

Reasoning OFF    Execute    Clear

Prefixes:

```
x rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>    x owl: <http://www.w3.org/2002/07/owl#>    x xsd: <http://www.w3.org/2001/XMLSchema#>  
x rdfs: <http://www.w3.org/2000/01/rdf-schema#>    x foaf: <http://xmlns.com/foaf/0.1/>
```

1 select \* where {?person foaf:givenName ?name}

The query

```
select * where {?person foaf:givenName ?name}
```

Finds variable assignments that satisfy the where clause

# Go to Query to enter a SPARQL query

The screenshot shows a web browser window with the URL `localhost:5820/mypeeps#!/query`. The page displays SPARQL results for a query about people, specifically looking for individuals named Alan, Bob, Carol, and Diana. The results are presented in a table format:

| person  | name   |
|---|--------|
| <a href="https://raw.githubusercontent.com/UMBC-CMSC-491-691-F18-Knowledge-Graphs/peeps/master/mypeeps.ttl#alan">https://raw.githubusercontent.com/UMBC-CMSC-491-691-F18-Knowledge-Graphs/peeps/master/mypeeps.ttl#alan</a>   | Alan   |
| <a href="https://raw.githubusercontent.com/UMBC-CMSC-491-691-F18-Knowledge-Graphs/peeps/master/mypeeps.ttl#bob">https://raw.githubusercontent.com/UMBC-CMSC-491-691-F18-Knowledge-Graphs/peeps/master/mypeeps.ttl#bob</a>     | Robert |
| <a href="https://raw.githubusercontent.com/UMBC-CMSC-491-691-F18-Knowledge-Graphs/peeps/master/mypeeps.ttl#carol">https://raw.githubusercontent.com/UMBC-CMSC-491-691-F18-Knowledge-Graphs/peeps/master/mypeeps.ttl#carol</a> | Carol  |
| <a href="https://raw.githubusercontent.com/UMBC-CMSC-491-691-F18-Knowledge-Graphs/peeps/master/mypeeps.ttl#diana">https://raw.githubusercontent.com/UMBC-CMSC-491-691-F18-Knowledge-Graphs/peeps/master/mypeeps.ttl#diana</a> | Diana  |

A red box highlights the "Export as..." button and a dropdown menu below it, which includes options for RDF/XML, JSON, TSV, and CSV.

**It found four solutions. The data can be exported to your computer as a file in any of several formats (e.g., rdf, json, csv, tsv)**

The screenshot shows a web browser window with a red box highlighting the error message area. The URL in the address bar is `localhost:5820/mypeeps#!/query`. The error message reads: "Error! Unknown prefix: peeps". Below this, the title "Query Panel" is visible, followed by a "SPARQL Editor" section with a "Prefixes:" list and a query editor containing the SPARQL code: `1 select * where {?person a peeps:Man}`.

Error!

Unknown prefix: peeps

## Query Panel

Hide SPARQL Editor

Explore

Reasoning  Execute

Prefixes:

```
1 select * where {?person a peeps:Man}
```

The query systems needs to know (independently) about any namespace prefixes you want to use (other than the common ones). Enter these when you create the database.

# Command line commands

Running a simple bash script will create or refresh the peeps knowledge graph example

```
#!/bin/bash
# loads peeps.ttl, mypeeps.ttl and associated namespaces into a Stardog database.

PORT="5820"
SERVER="http://localhost:$PORT"
DBNAME="mypeeps"
DBURL="$SERVER/$DBNAME"

# stop server in case one is already running
stardog-admin --server $SERVER server stop

# start server
stardog-admin server start --port $PORT --disable-security
# drop database $DBNAME in case it exists already
stardog-admin --server $SERVER db drop -n $DBNAME

# create database $DBNAME with reasoning and search enabled
stardog-admin --server $SERVER db create -o reasoning.sameas=FULL -o search.enabled=true -n $DBNAME

# load ontology and data
stardog data add $DBURL peeps.ttl mypeeps.ttl

# add namespace prefixes for the query system to use
stardog namespace import --verbose $DBURL prefixes.ttl
```

# Query from Python

- Stardog serves as a endpoint for SPARQL queries
- Use this URL to send queries to the mypeeps database  
<http://localhost:5820/mypeeps/query/>
- There are packages that help do this in many languages, including Python
- See [query.py](#) in the peeps repository