TU WIEN Informatics

192.161 Management of Graph Data

(4.0 VU / 6.0 ECTS)

2025W

Advanced Topics

Katja Hose

Maxime Jakubowski

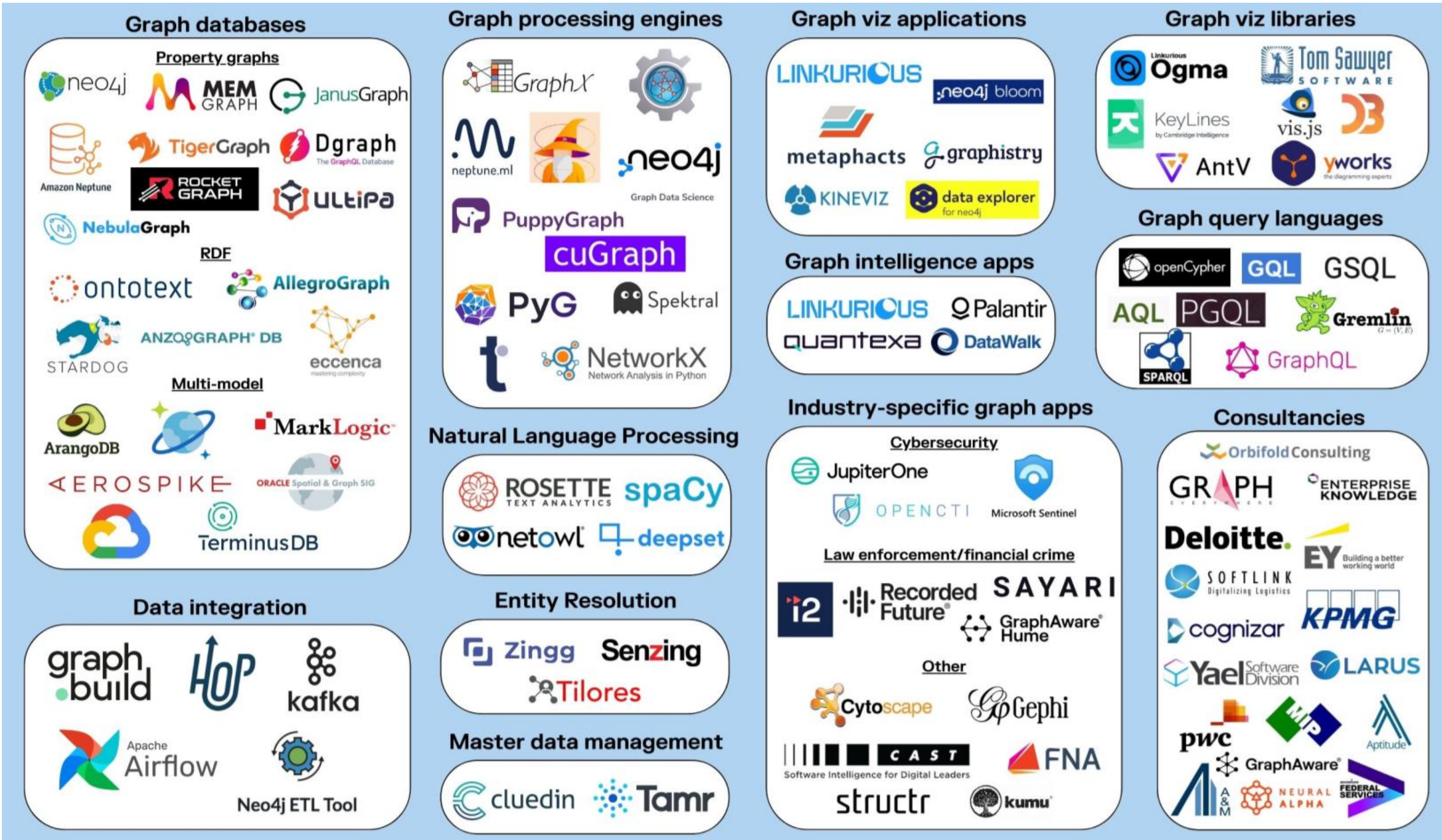
mogda@list.tuwien.ac.at

Lecture Outline

- Advanced topics in graph data management
- Course feedback
- Exam information

TU WIEN Informatics

Graph technology landscape



https://linkurious.com/blog/introduction-graph-technology-landscape/

TU WIEN Informatics

Knowledge graph challenges for GDBMS and AI

Data Modeling and Interoperability

- alternative graph data models (RDF, property graphs) and query languages (SPARQL, Cypher, GQL)
- schema modeling (ontologies, PG-schema)
- multimodal data
- data integration (OBDA, entity resolution, virtual KGs, etc.)

Knowledge graph challenges for GDBMS and AI

TU WIEN Informatics

Scalability and Querying

- efficiently handling large-scale graphs
- indexing, statistics, query optimization
- centralized/cluster/federated
- example-driven analytics and exploration

Data Modeling and Interoperability

- alternative graph data models (RDF, property graphs) and query languages (SPARQL, Cypher, GQL)
- schema modeling (ontologies, PG-schema)
- multimodal data
- data integration (OBDA, entity resolution, virtual KGs, etc.)

TU WIEN Informatics

Knowledge graph challenges for GDBMS and AI

Scalability and Querying

- efficiently handling large-scale graphs
- indexing, statistics, query optimization
- centralized/cluster/federated
- example-driven analytics and exploration

Data Modeling and Interoperability

- alternative graph data models (RDF, property graphs) and query languages (SPARQL, Cypher, GQL)
- schema modeling (ontologies, PG-schema)
- multimodal data
- data integration (OBDA, entity resolution, virtual KGs, etc.)

Quality, Provenance, and Dynamics

- ensuring accuracy, completeness, and consistency (SHACL, SHeX, PG-schema)
- Knowledge evolution, temporal knowledge graphs
- provenance, lineage

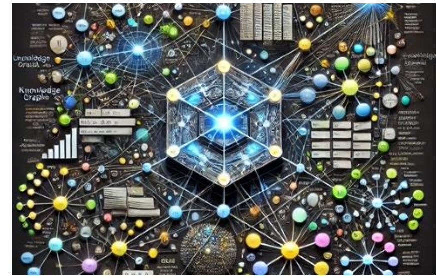
Knowledge graph challenges for GDBMS and AI

TU WIEN Informatics

Scalability and Querying

- efficiently handling large-scale graphs
- indexing, statistics, query optimization
- centralized/cluster/federated
- example-driven analytics and exploration

Reasoning and Neurosymbolic AI



- combining logical reasoning with machine learning
- embeddings, vector representations
- graph neural networks
- multi-modal ML

Data Modeling and Interoperability

- alternative graph data models (RDF, property graphs) and query languages (SPARQL, Cypher, GQL)
- schema modeling (ontologies, PG-schema)
- multimodal data
- data integration (OBDA, entity resolution, virtual KGs, etc.)

Quality, Provenance, and Dynamics

- ensuring accuracy, completeness, and consistency (SHACL, SHeX, PG-schema)
- Knowledge evolution, temporal knowledge graphs
- provenance, lineage

TU WIEN Informatics

Knowledge graph challenges for GDBMS and AI

Scalability and Querying

- efficiently handling large-scale graphs
- indexing, statistics, query optimization
- centralized/cluster/federated
- example-driven analytics and exploration

Reasoning and Neurosymbolic AI

- combining logical reasoning with machine learning
- embeddings, vector representations
- graph neural networks
- multi-modal ML

- Semantic Data Lakes
- information retrieval
- AutoML
- data catalogs
- explainable AI
- digital twins

Data Modeling and Interoperability

- alternative graph data models (RDF, property graphs) and query languages (SPARQL, Cypher, GQL)
- schema modeling (ontologies, PG-schema)
- multimodal data
- data integration (OBDA, entity resolution, virtual KGs, etc.)

Quality, Provenance, and Dynamics

- ensuring accuracy, completeness, and consistency (SHACL, SHeX, PG-schema)
- Knowledge evolution, temporal knowledge graphs
- provenance, lineage

KGs as Enabling Technology

TU WIEN Informatics

Knowledge graph challenges for GDBMS and AI

Scalability and Querying

- efficiently handling large-scale graphs
- indexing, statistics, query optimization
- centralized/cluster/federated
- example-driven analytics and exploration

Reasoning and Neurosymbolic AI

- combining logical reasoning with machine learning
- embeddings, vector representations
- graph neural networks
- multi-modal ML

Generative AI

- LLMs and hallucinations
- Conversational agents
- Agentic AI
- GraphRAG

Data Modeling and Interoperability

- alternative graph data models (RDF, property graphs) and query languages (SPARQL, Cypher, GQL)
- schema modeling (ontologies, PG-schema)
- multimodal data
- data integration (OBDA, entity resolution, virtual KGs, etc.)

Quality, Provenance, and Dynamics

- ensuring accuracy, completeness, and consistency (SHACL, SHeX, PG-schema)
- Knowledge evolution, temporal knowledge graphs
- provenance, lineage

KGs as Enabling Technology

- Semantic Data Lakes
- information retrieval
- AutoML
- data catalogs
- explainable AI
- digital twins

TU WIEN Informatics

Knowledge graph challenges for GDBMS and AI

Scalability and Querying

- efficiently handling large-scale graphs
- indexing, statistics, query optimization
- centralized/cluster/federated
- example-driven analytics and exploration

Reasoning and Neurosymbolic AI

- combining logical reasoning with machine learning
- embeddings, vector representations
- graph neural networks
- multi-modal ML

Generative AI

- LLMs and hallucinations
- Conversational agents
- Agentic AI
- GraphRAG

Data Modeling and Interoperability

- alternative graph data models (RDF, property graphs) and query languages (SPARQL, Cypher, GQL)
- schema modeling (ontologies, PG-schema)
- multimodal data
- data integration (OBDA, entity resolution, virtual KGs, etc.)

Quality, Provenance, and Dynamics

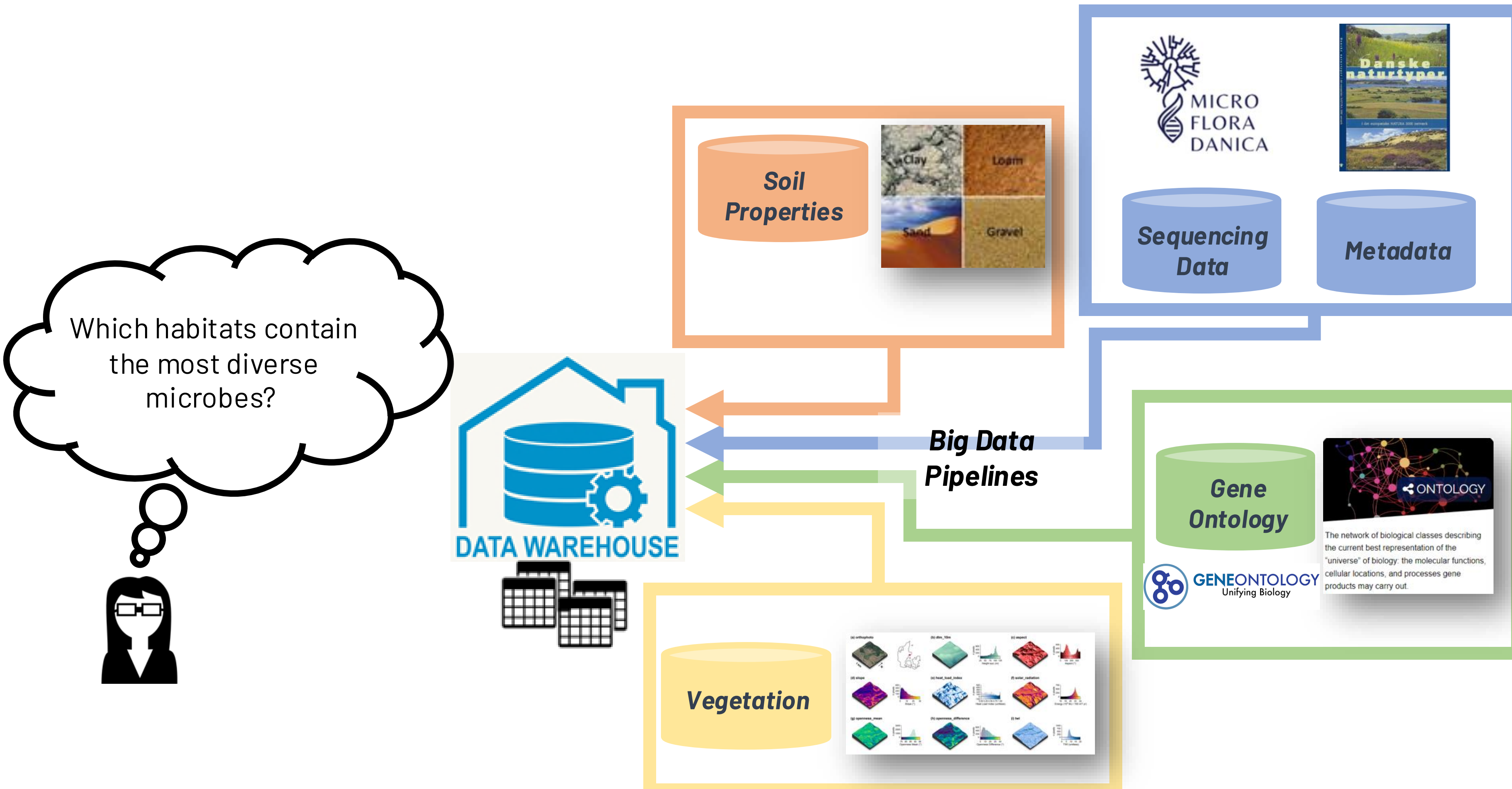
- ensuring accuracy, completeness, and consistency (SHACL, SHeX, PG-schema)
- Knowledge evolution, temporal knowledge graphs
- provenance, lineage

KGs as Enabling Technology

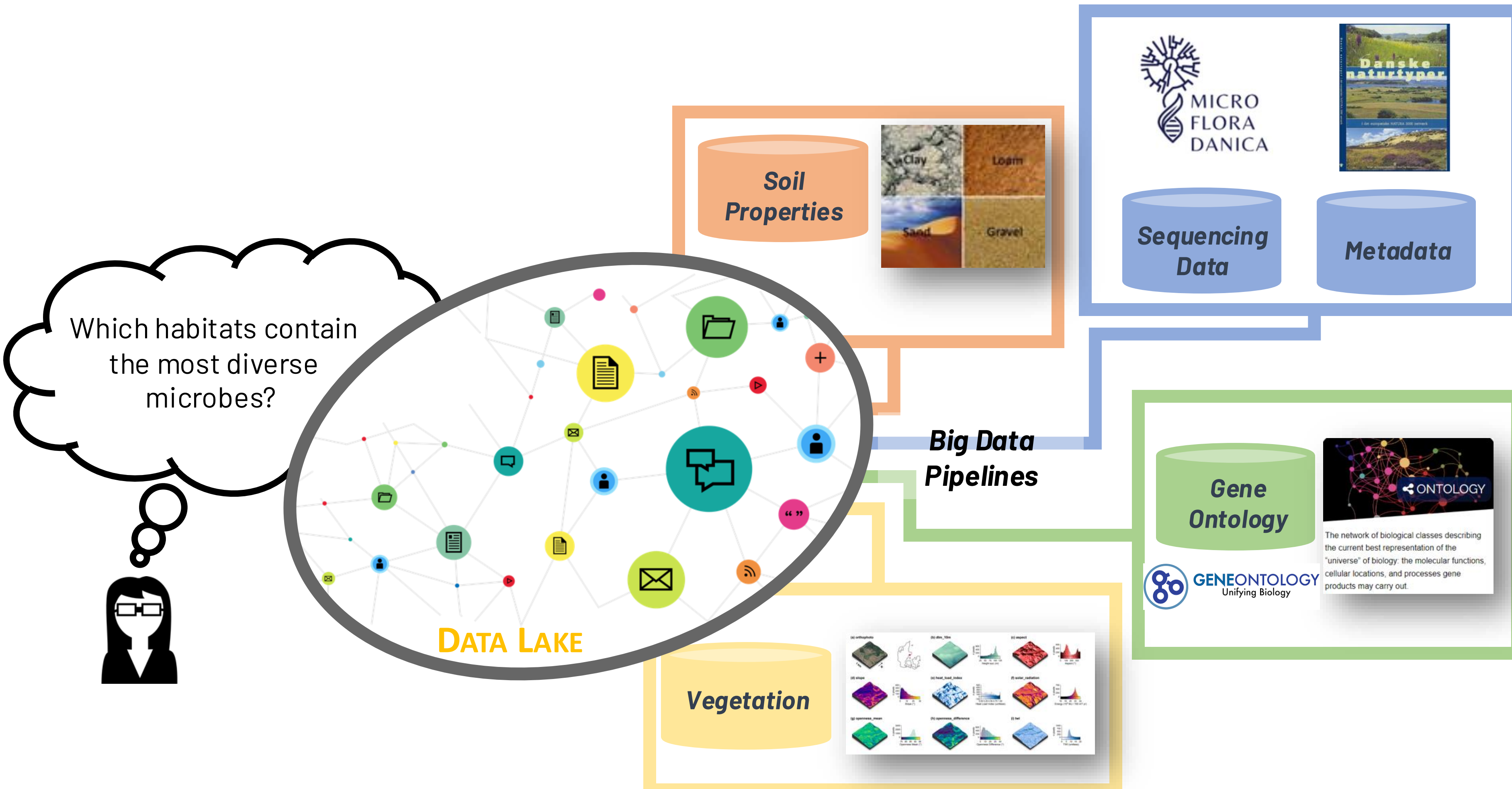
- Semantic Data Lakes
- information retrieval
- AutoML
- data catalogs
- explainable AI
- digital twins

TU WIEN Informatics

Data Warehouses



Knowledge-Graph based Integration, Data Lakes, Data Fabrics, Enterprise Knowledge Graphs



TU WIEN Informatics

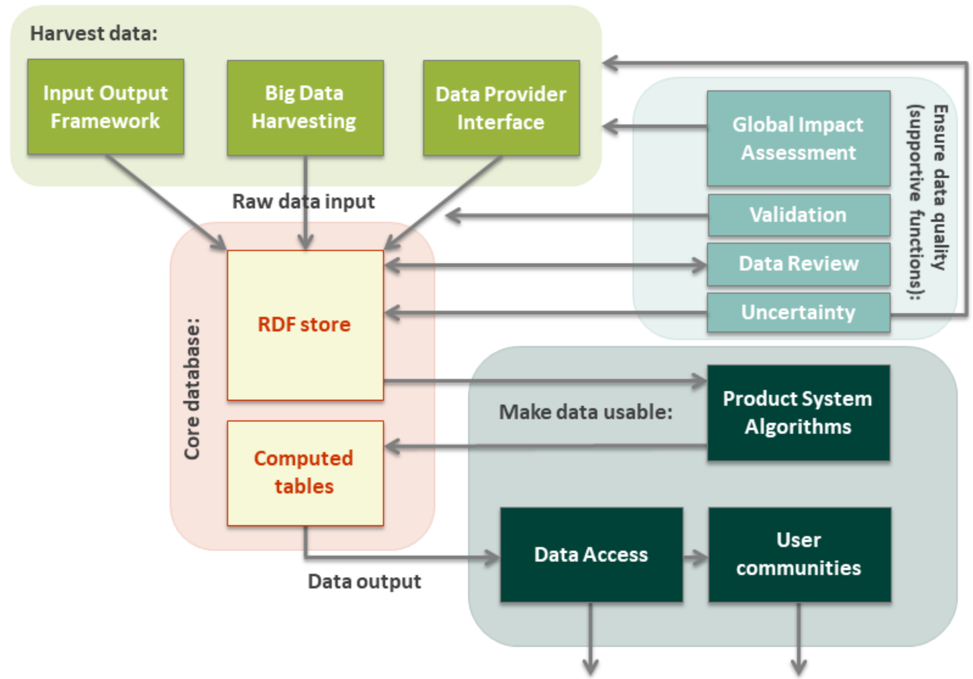
Data integration and sharing

Use case
Lifecycle sustainability assessment

Goal
Overcoming closed data silos

Framework
- Harvest data: data extraction
- Core database: pipeline for data integration, provenance for tracing data
- Ensuring data quality (LCA reasoning)
- Capturing provenance (PROV-O)
- Make data usable (user interfaces and APIs)

140M triples of instance-level data
2K triples of schema-level data



E. R. Hansen, M. Lissandrini, A. Ghose, S. Løkke, C. Thomsen, K. Hose: Transparent Integration and Sharing of Life Cycle Sustainability Data with Provenance. ISWC 2020

TU WIEN Informatics

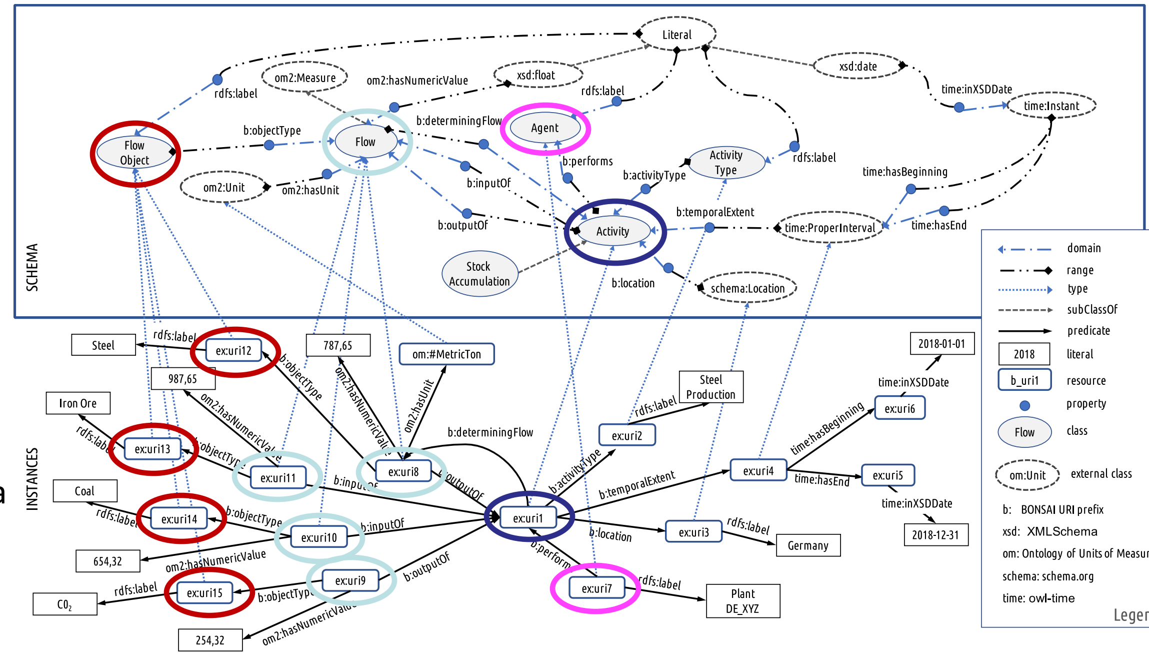
Data integration and sharing

Use case
Lifecycle sustainability assessment

Goal
Overcoming closed data silos

Framework
- Harvest data: data extraction
- Core database: pipeline for data integration, provenance for tracing data
- Ensuring data quality (LCA reasoning)
- Capturing provenance (PROV-O)
- Make data usable (user interfaces and APIs)

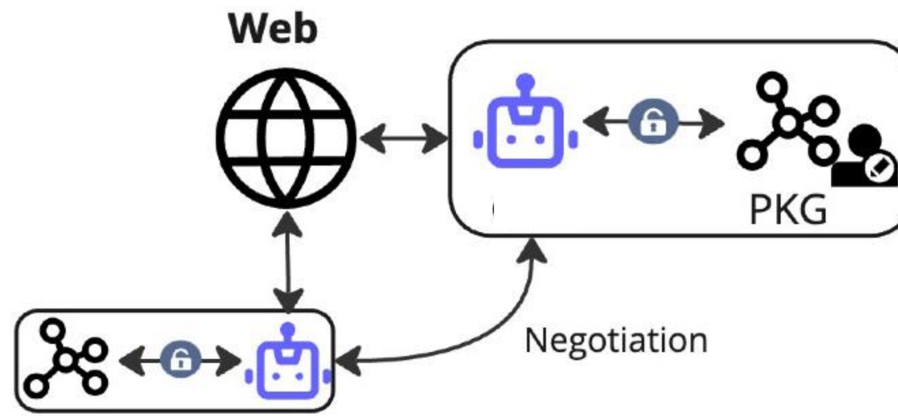
140M triples of instance-level data
2K triples of schema-level data



A. Ghose, K. Hose, M. Lissandrini, B. Weidema
An Open Source Dataset and Ontology
for Product Footprinting. In: ESWC 2019

Personalized agents

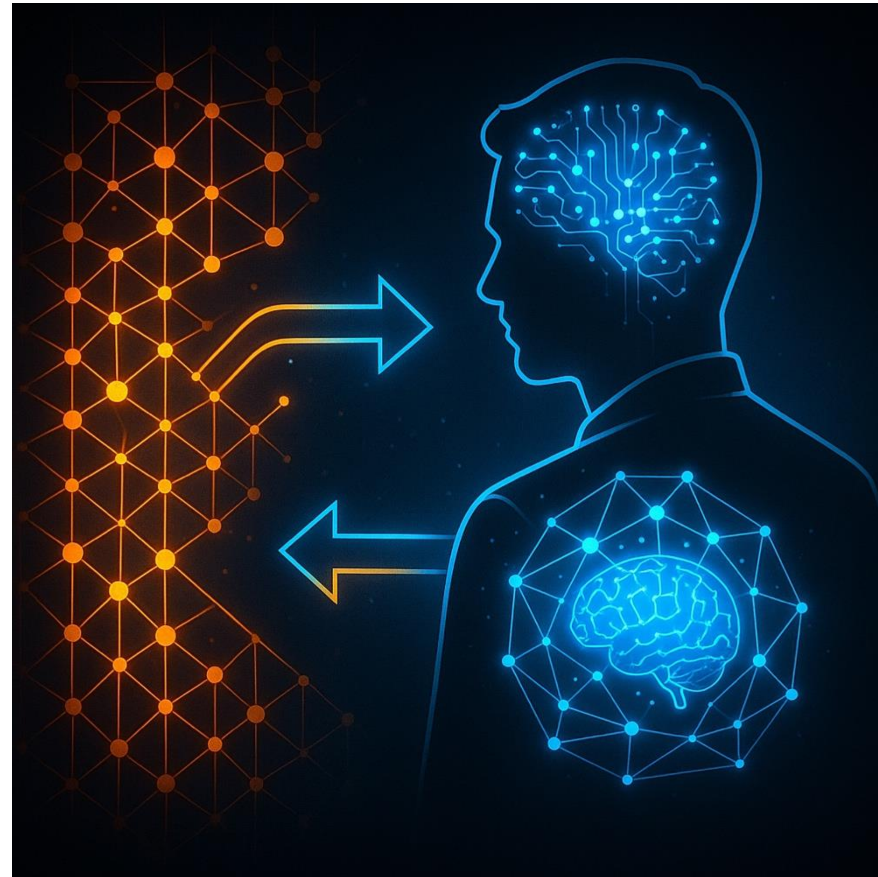
- Advancing Agentic AI
- Combine LLMs + Personal Knowledge Graphs (PKG)
- Goal
    - Personal automation with trust , structure, and control



Piero A. Bonatti, J. Domingue, A. L. Gentile, A. Harth, Olaf. Hartig, A. Hogan, K. Hose, E. Jimenez-Ruiz, D. L. McGuinness, I. Pickup, C. Sun, R. Verborgh, J. Wright: Towards Computer-Using Personal Agents, arxiv.org, 2025.

Use case: planning a dinner date

- Agent assists in meal choice and shopping
- Coordinates between users‘ agents (diet, allergies, schedule)
- Learns from outcomes (e.g., heartburn → avoid next time)
- Challenge: autonomy + privacy + policy-aware reasoning

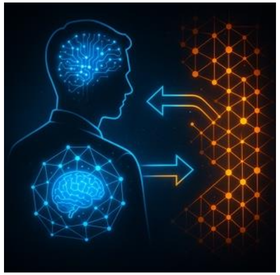


Piero A. Bonatti, J. Domingue, A. L. Gentile, A. Harth, Olaf. Hartig, A. Hogan, K. Hose, E. Jimenez-Ruiz, D. L. McGuinness, I. Pickup, C. Sun, R. Verborgh, J. Wright: Towards Computer-Using Personal Agents, arxiv.org, 2025.

Use case: planning a dinner date

Sam: Jane is coming over for dinner, propose Thai food recipes.

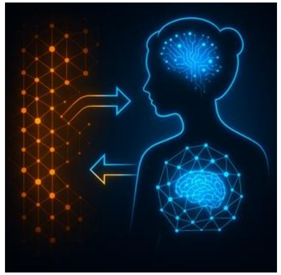
Sam + Agent



recipes →

← No coriander, no shellfish (allergy)

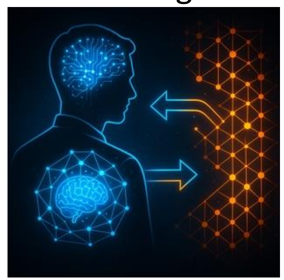
Jane + Agent



Use case: planning a dinner date

Sam: Jane is coming over for dinner, propose Thai food recipes.
→ agent proposes recipes (Sam is pre-diabetic)
→ find out which ones Jane probably likes

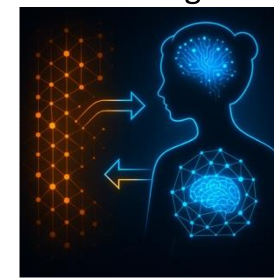
Sam + Agent



recipes →

← No coriander, no shellfish (allergy)

Jane + Agent

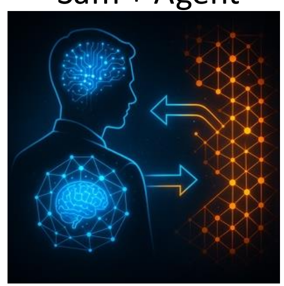


Use case: planning a dinner date

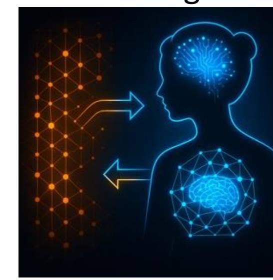
Sam: Jane is coming over for dinner, propose Thai food recipes.
→ agent proposes recipes (Sam is pre-diabetic)
→ find out which ones Jane probably likes

Sam: picks one of the proposed options, asks agent to order ingredients
Sam: dinner at 7pm?

Sam + Agent



Jane + Agent



recipes

No coriander, no shellfish (allergy)

dinner at 7pm

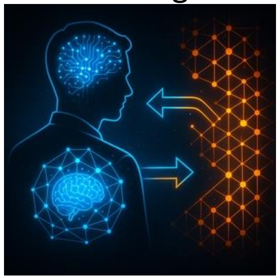
TU WIEN Informatics

Use case: planning a dinner date

Sam: Jane is coming over for dinner, propose Thai food recipes.
→ agent proposes recipes (Sam is pre-diabetic)
→ find out which ones Jane probably likes

Sam: picks one of the proposed options, asks agent to order ingredients
Sam: dinner at 7pm?

Sam + Agent



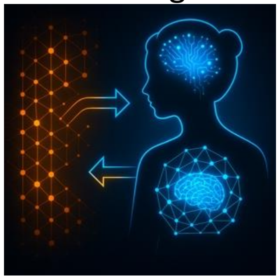
recipes →

← No coriander, no shellfish (allergy)

dinner at 7pm →

← 8pm, traffic jam because of an accident

Jane + Agent



TU WIEN Informatics

Use case: planning a dinner date

Sam: Jane is coming over for dinner, propose Thai food recipes.
→ agent proposes recipes (Sam is pre-diabetic)
→ find out which ones Jane probably likes

Sam: picks one of the proposed options, asks agent to order ingredients
Sam: dinner at 7pm?

Sam: I have a heartburn
→ agent for the future

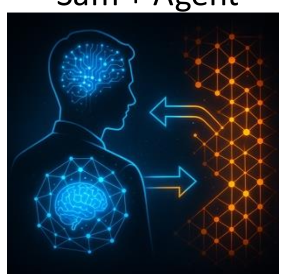
recipes

No coriander, no shellfish (allergy)

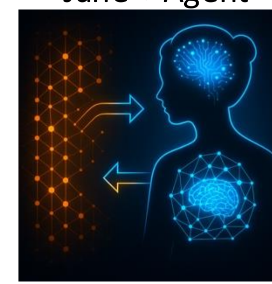
dinner at 7pm

8pm, traffic jam because of an accident

Sam + Agent



Jane + Agent



Some Topics Around SHACL

One of our research focus points

TU WIEN Informatics

Bridging Gaps in RDF Validation

A community survey on the use of schema languages for RDF

Data quality with SHACL

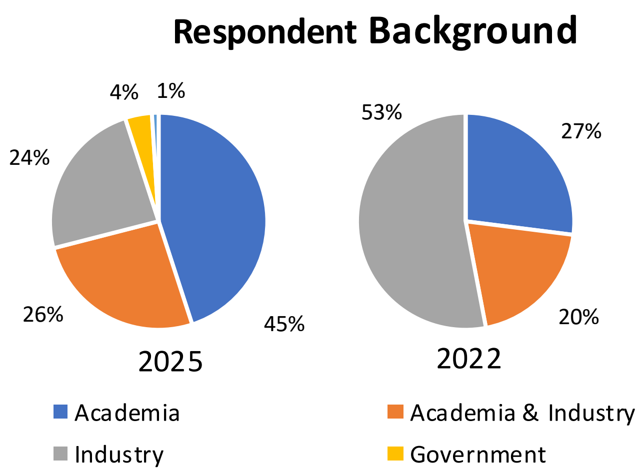
Language for validating RDF graphs against a set of conditions

W3C recommendation since July 2017: https://www.w3.org/TR/shacl/

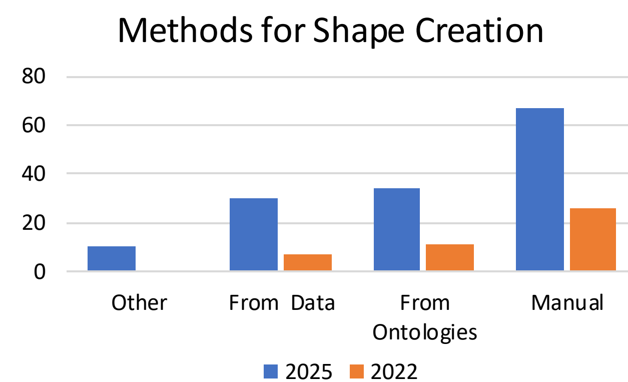
SHACL allows validation of knowledge graphs in RDF by defining a shapes graph to validate a data graph.

TU WIEN Informatics

Validating RDF Data



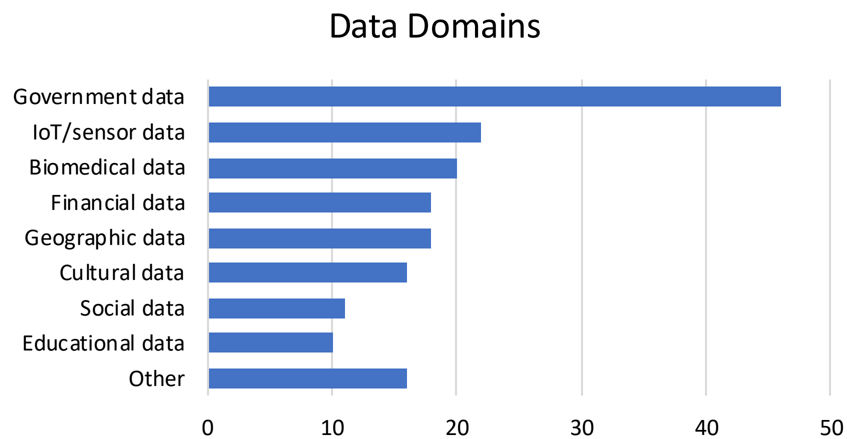
- Online survey from 2025
- Comparison to a survey from 2022

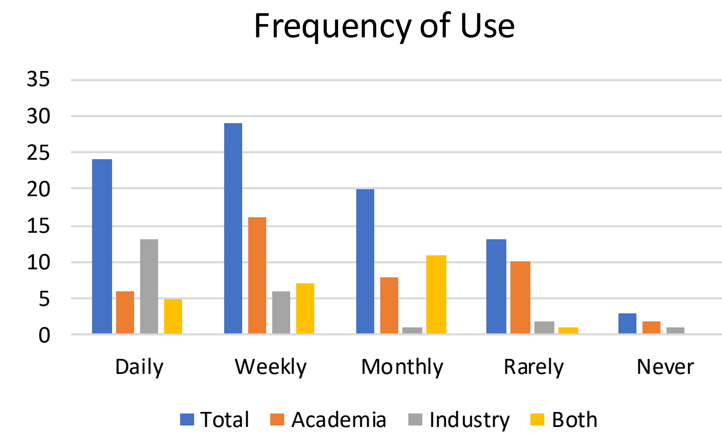


- Overwhelmingly manual
- Often created from other artifacts
- OWL Ontologies
- JSON Schema
- Conceptual Models

M. Jakubowski, D. Tomaszuk, K. Hose: Bridging Gaps in RDF Validation – Insights and Innovation Opportunities in RDF Validation Practices. SEMANTiCS 2025

Data Domains and Frequency of Use





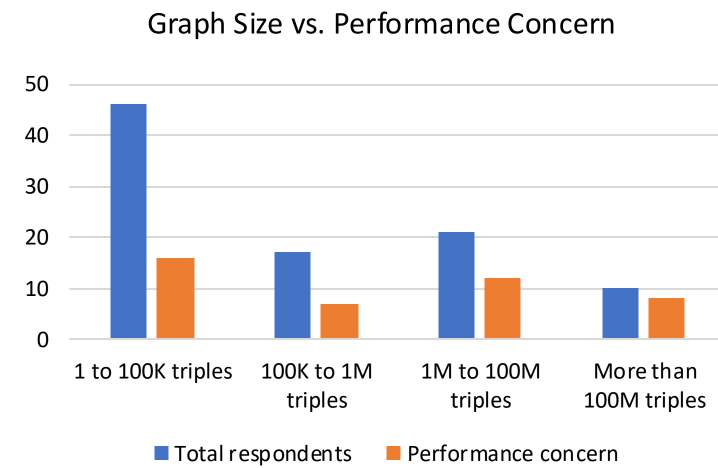
Challenges

Issue / Limitation	Academia	Industry	Both
Documentation and tutorials	22	8	11
Community and tool support	20	11	8
Performance and scalability	17	9	11
Features or functionality	14	11	8
Missing constraint types	11	11	10

TU WIEN Informatics

Challenges

Issue / Limitation	Academia	Industry	Both
Documentation and tutorials	22	8	11
Community and tool support	20	11	8
Performance and scalability	17	9	11
Features or functionality	14	11	8
Missing constraint types	11	11	10



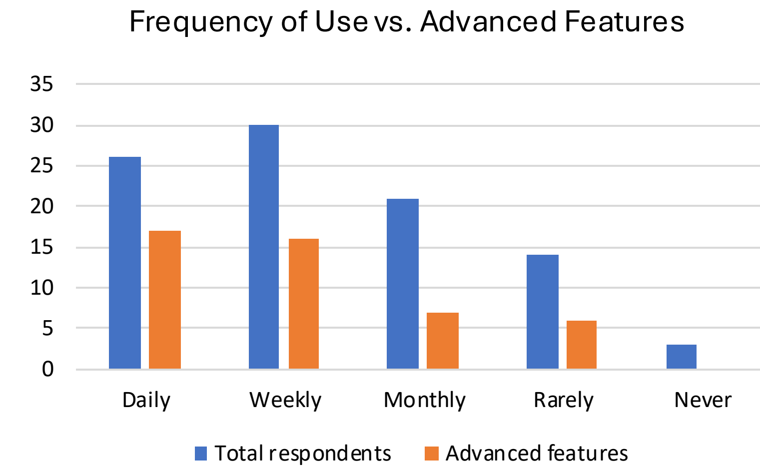
- Performance bottleneck for large datasets
- Few validators focus on performance
- Comparatively little academic literature
  - Idea: bring known techniques for query processing to SHACL/ShEx
  - Our work: [Compiling SHACL Into SQL] at ISWC 2024

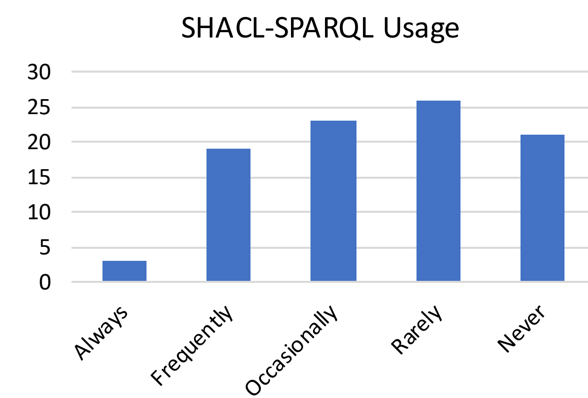
TU WIEN Informatics

Challenges

Issue / Limitation	Academia	Industry	Both
Documentation and tutorials	22	8	11
Community and tool support	20	11	8
Performance and scalability	17	9	11
Features or functionality	14	11	8
Missing constraint types	11	11	10

- Advanced Features extends SHACL
  - Expressive power
  - Widen the scope of SHACL
- Need for greater expressiveness
- Need for expansion of SHACL’s scope:
  - SHACL rules often cited





TU WIEN Informatics

Graph Subsetting and Neighborhoods

An alternative use for SHACL

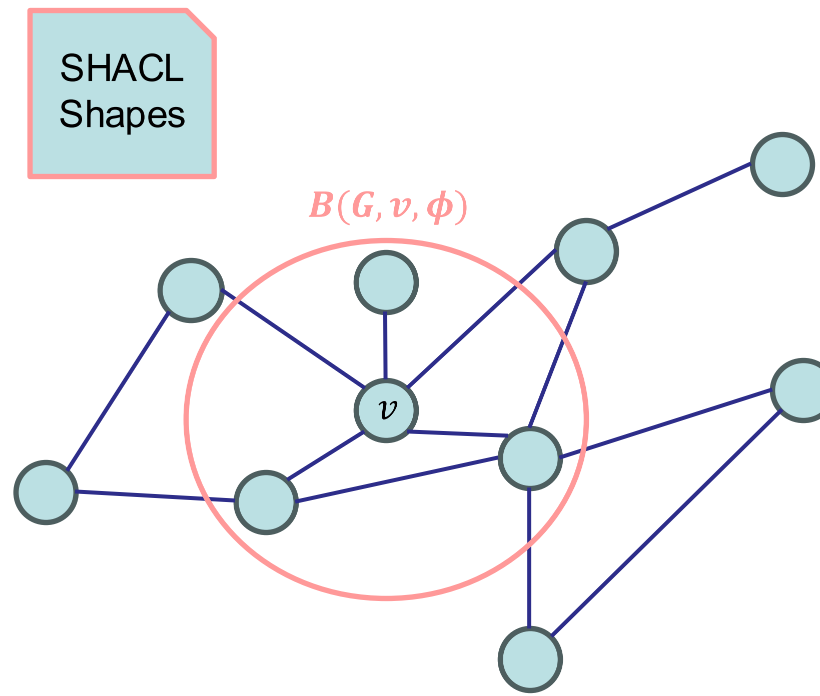
Neighborhoods

Goal:
Provide **a subgraph** that only contains triples that are "relevant"

We define the **neighborhood**: $B(G, v, \phi)$

- G a graph
- v a node
- ϕ a shape

What part of G is relevant to decide that v satisfies ϕ in G?



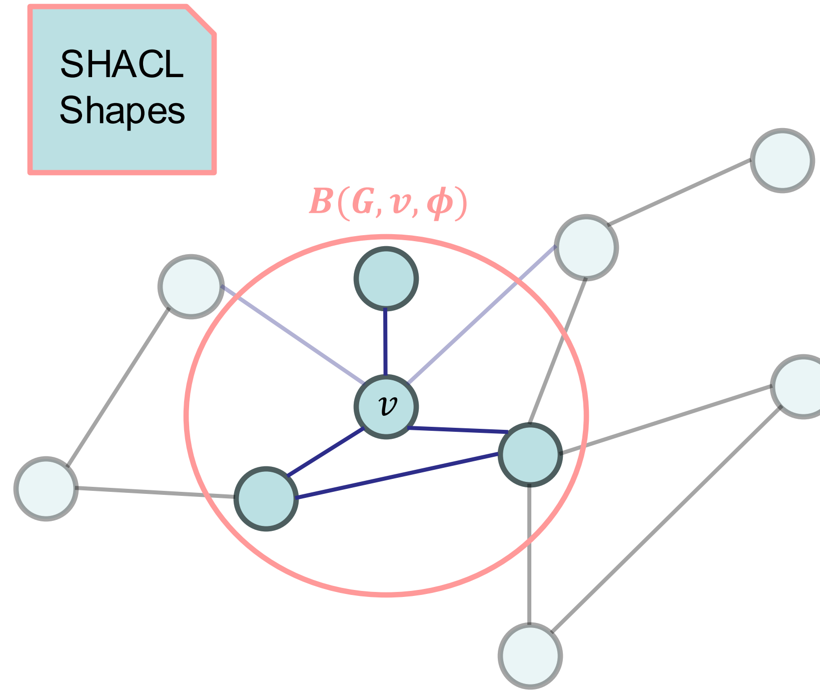
Neighborhoods

Goal:
Provide **a subgraph** that only contains triples that are “relevant”

We define the **neighborhood**: $B(G, v, \phi)$

- G a graph
- v a node
- ϕ a shape

What part of G is relevant to decide that v satisfies ϕ in G?



Sufficiency and design principles

Sufficiency Property	If a node v satisfies a shape ϕ in a graph G, then: v also satisfies ϕ in G' for any subgraph G' with $B(G, v, \phi) \subseteq G' \subseteq G$.

"Strong" sufficiency: it holds for all G' where $B(G, v, \phi) \subseteq G' \subseteq G$.

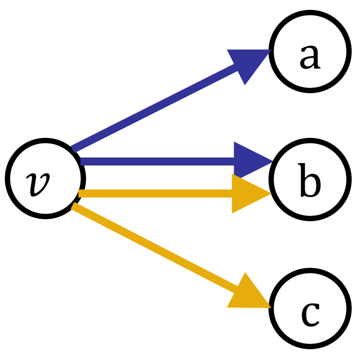
- Technical necessity
- Allows for leniency in implementations of neighborhoods

When defining neighborhoods we want to be both **deterministic** and **minimal**

Nonequality: $\neg eq(p, q)$

```
<myshape> sh:not [
  sh:path <p>;
  sh:equals <q>;
].
```

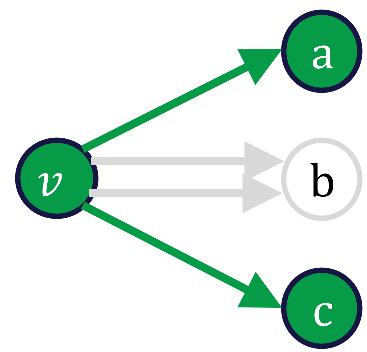
G



Options:
- Only edges to a and/or c
- All edges

$B(G, v, \mathrm{myshape})$

“symetric difference”

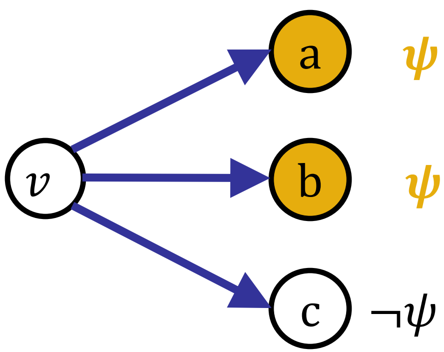


Reasons:
- Determinism
- Somehow minimal

Quantifiers: $\geq_1 p.\psi$

```
<myshape> sh:property [
  sh:path <p>;
  sh:qualifiedValueShape ψ ;
  sh:qualifiedMinCount 1 ;
].
```

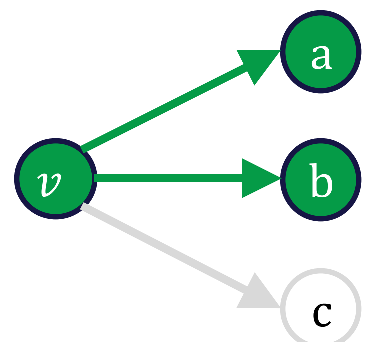
G



Options:

- Only edges to a and/or b
- All edges

$B(G, v, \phi)$



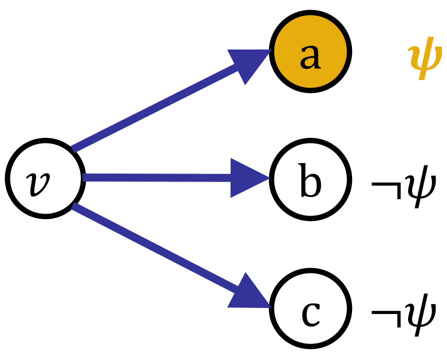
Reasons:

- Determinism
- Somehow minimal

Quantifiers: $\leq_1 p.\psi$

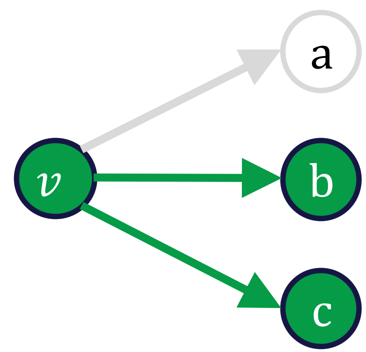
```
<myshape> sh:property [
  sh:path <p>;
  sh:qualifiedValueShape ψ ;
  sh:qualifiedMaxCount 1 ;
].
```

G



Options:
- No edges
- Edge to a
- All edges

$B(G, v, \phi)$



Reasons:
- Determinism
- Somehow minimal
- Adding edges to the neighborhood may not break sufficiency

Shape Fragments

We define $\mathbf{Frag}(\boldsymbol{G}, \boldsymbol{S})$ as the union of all neighborhoods of nodes satisfying the shapes from S in G.

Let H be a shape schema, we define:

$$\mathbf{Frag}(\boldsymbol{G}, \boldsymbol{H}) \coloneqq \text{Frag}(G, S)$$

where $S = \{\phi \wedge \tau \mid \tau \textit{ is the target of } \phi \textit{ in } H\}$

Conformance Property	If a graph G satisfies a schema H, then $\mathrm{Frag}(G, H)$ also conforms to H.

Shape Fragments can be used for…

- Explaining validation
- Graph subsetting
- A notion of "coverage"

Want to know more? https://openproceedings.org/2023/conf/edbt/paper-3.pdf
Want to know even more? https://dl.acm.org/doi/10.1007/978-3-031-72407-7_1

TU WIEN Informatics

Common Foundations of SHACL, ShEx and PG-Schema

They are very different in nature, but closer than it seems

Understand differences and commonalities between SHACL, ShEx and PG-Schema

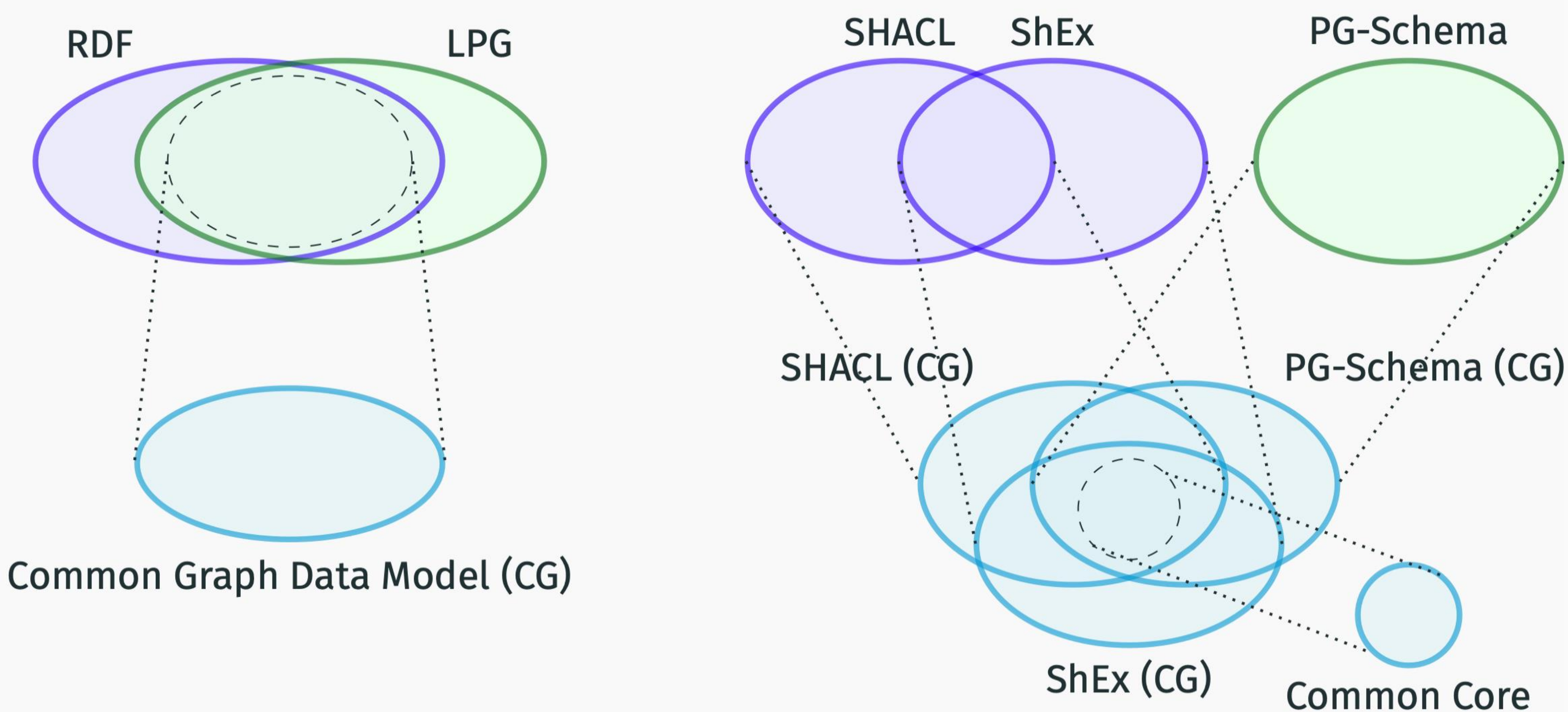
- **Users** are asking for this:
  - What are the fundamental differences? Is true interoperability possible?
- **Vendors** are asking for this:
  - What is actually used, and what is essential to support?
- **Designers** should be asking for this:
  - Why are there all these differences despite the similar objectives?
  - Do the superficial differences matter?

They certainly *look* different

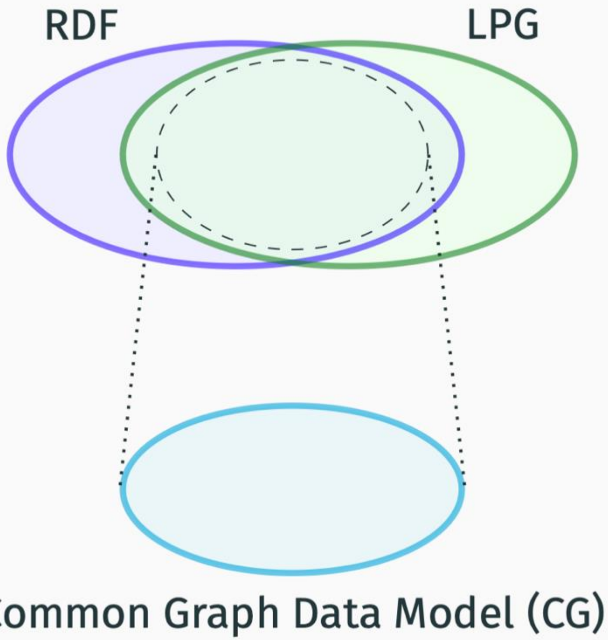
	SHACL	ShEx	PG-Schema
Data Model	RDF	RDF	LPG
Paradigm	Logical constraints	Regular expressions	Type-based + rich key constraints

- Their respective theoretical foundations seem only loosely related
- Defined for different data models
- Defined by different communities

Method


RDF
LPG
Common Graph Data Model (CG)
SHACL
ShEx
PG-Schema
SHACL (CG)
PG-Schema (CG)
ShEx (CG)
Common Core

Distilling the Common Graph Data Model



Features not present in both RDF and LPG are removed.

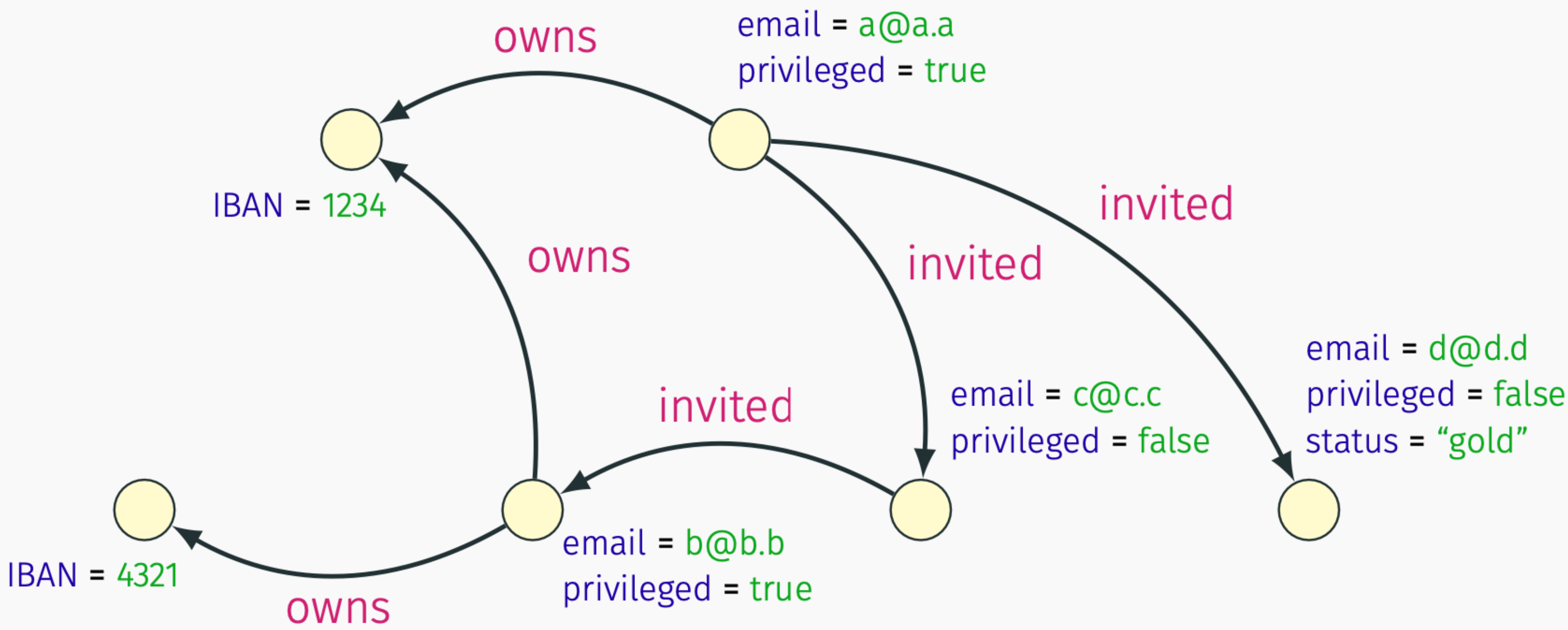
RDF Gives up:

- **Nodes** cannot be compared with an IRI
- **Predicates** …
  - associate at most one value with a node
  - cannot be compared with a node

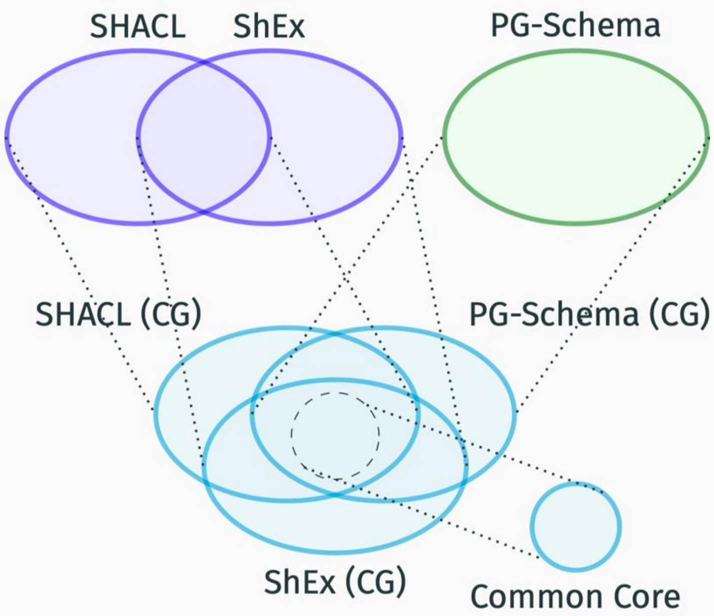
LPG Gives up:

- **Nodes** do not have labels
- **Edges** …
  - have exactly one label
  - have no properties
  - are identified by label and endpoints

Common Graph Example



Distilling the three schema languages

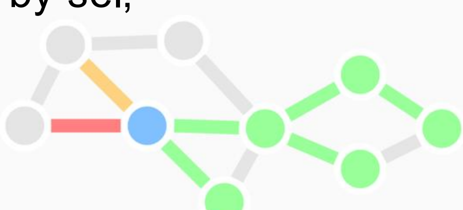


A **common shape** describes the graph's structure around a focus node or value.

A selector is a simple shape that indicates **focus** nodes or values.

A **schema** is a set of selector-shape pairs.

A graph G **conforms to** a schema S if,
for each selector-shape pair (sel, s) in S,
for every focus v indicated by sel,
v has shape s



Distilling the three schema languages

SHACL and ShEx give up

- disjunctions and negations,
- nesting of shapes.

```
<s> sh:or (
[ sh:path <invited>; sh:minCount 1 ]
[ sh:not [ sh:path <status> sh:hasValue “gold”]).
```

SHACL and PG-Schema give up

- unbounded path expressions.
- counting over multi-edge paths.

```
<s> sh:path [ sh:zeroOrMorePath <invited> ];
  sh:minCount 100 .
```

PG-Schema gives up

- closing properties and edges independently,
- testing properties under star.

ShEx gives up recursion.

TU WIEN Informatics

The Common Graph Schema Language (CoGSL)

Definition 12 (common shape). *A* common shape φ *is an expression given by the grammar*

$$\varphi ::= \exists\,\pi \mid \exists^{\leq n}\,\pi_1 \mid \exists^{\geq n}\,\pi_1 \mid \exists\,\mathbb{C} \land \nexists\,\neg P \mid \varphi \land \varphi\ .$$

$$\mathbb{C} ::= \{\} \mid \{k : \mathbb{V}\} \mid \mathbb{C}\ \&\ \mathbb{C} \mid \mathbb{C} \mid \mathbb{C}\ .$$

$$\pi_0 ::= [k = c] \mid \neg[k = c] \mid \mathbb{C}\ \&\ \top \mid \neg(\mathbb{C}\ \&\ \top) \mid \pi_0 \cdot \pi_0\ .$$

$$\pi_1 ::= \pi_0 \cdot p \cdot \pi_0 \mid \pi_0 \cdot p^- \cdot \pi_0 \mid \pi_0 \cdot k \mid k^- \cdot \pi_0\ .$$

$$\bar{\pi} ::= \pi_0 \mid p \mid \bar{\pi}^- \mid \bar{\pi} \cdot \bar{\pi} \mid \bar{\pi} \cup \bar{\pi}\ .$$

$$\pi ::= \bar{\pi} \mid \bar{\pi} \cdot k \mid k^- \cdot \bar{\pi} \mid k^- \cdot \bar{\pi} \cdot k'\ .$$

where $n \in \mathbb{N}$, $P \subseteq_{fin} \mathcal{P}$, $k, k' \in \mathcal{K}$, $c \in \mathcal{V}$, *and* $p \in \mathcal{P}$.

Want to know more? https://dl.acm.org/doi/10.1145/3696410.3714694

What have we gained?

- A common **framework** to talk about SHACL, ShEx, and PG-Schema, compare their core mechanisms and expressive power:
  - a high-level notion of a shape-based graph schema;
  - a uniform syntax for shapes and selectors.
- A **foundation** for automating interoperability:
  - a data model that can be supported by both RDF and LPG;
  - a schema language that can be compiled to SHACL, ShEx, and PG-Schema.

Master Thesis Topics

Master Thesis Topics

- Schema-related topics:
  - Creating a fast and scalable SHACL validator
  - Generating RDF data from SHACL shapes
  - Validating virtual graph data
  - Mining schemas from graph data
- Applied Data Management
  - Integrating biological datasets into graph data
  - Federated exploration of microbial and environmental data
- Evolving graph data
  - Visualizing evolving graph data
  - Benchmarking evolving property graphs
- Please contact us for more details, or to sit together and develop a topic!

Course Feedback

Please provide feedback to the course using the form on TUWEL (anonymous)

Exam Information

TU WIEN Informatics

Exam & Course Feedback

- Project hand-in: 18th of January 2026
- Written exam: 9th of January 2026
- Second chance: 2nd of March 2026
- Points: Project (60) + Exam (40)
  - You must pass both (at least 50%)
  - For the exam, the last attempt counts

- Exam structure
  - Theory: a set of yes/no questions
  - Understanding and interpreting queries/schemas
    - Either open questions;
    - Or complex multiple-choice questions