Knowledge-Based Data-Management with Graal V2

Federico Ulliana
Univ. Montpellier, INRIA, LIRMM

INRAE Semantic Web Seminar
October 13, 2022

This talk's goal

Introduce you to a novel platform we are currently building : Graal V2

1. The aim : Knowledge-Based Data Management with Existential Rules

2. Zoom on the project

Knowledege-Based Data Management

Exploiting Data can be Challenging

Many examples have been presented during this seminar, and there are even more.

Heterogeneity

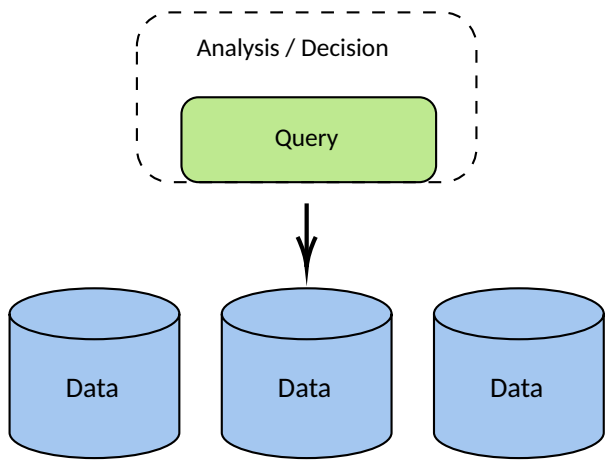
- data sources can be different
- models, platforms : API, languages, access
- refinement, dimensions : query constructs
- dynamicity, confidence : validity of answers

Challenges

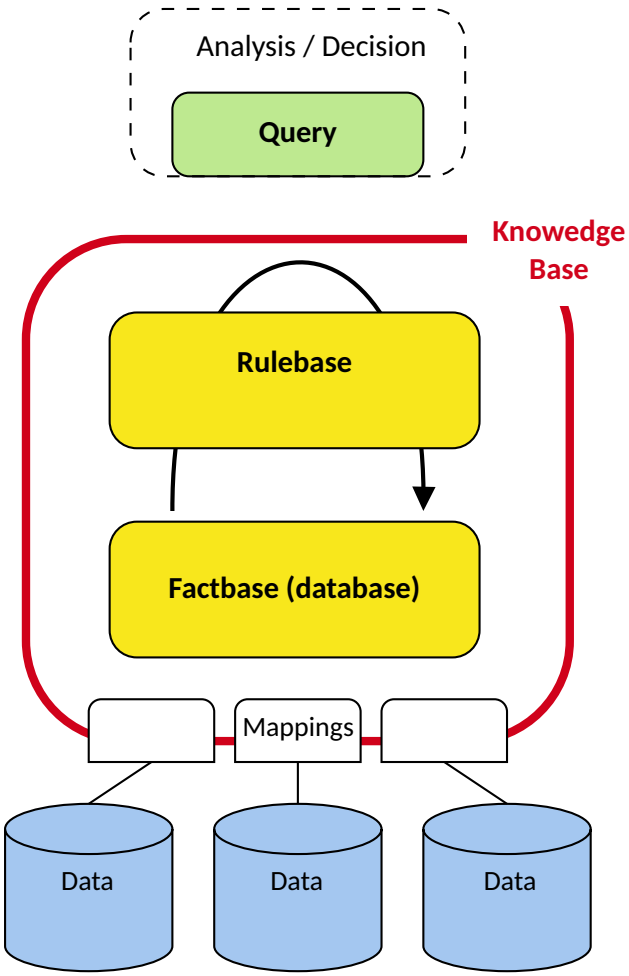
1. Express & maintain data-services
2. Decision-making : cross-querying
3. Inconsistencies are endemic
4. Explanations for answers & errors

Federation

- number of sources grows as problem scales
- independent systems



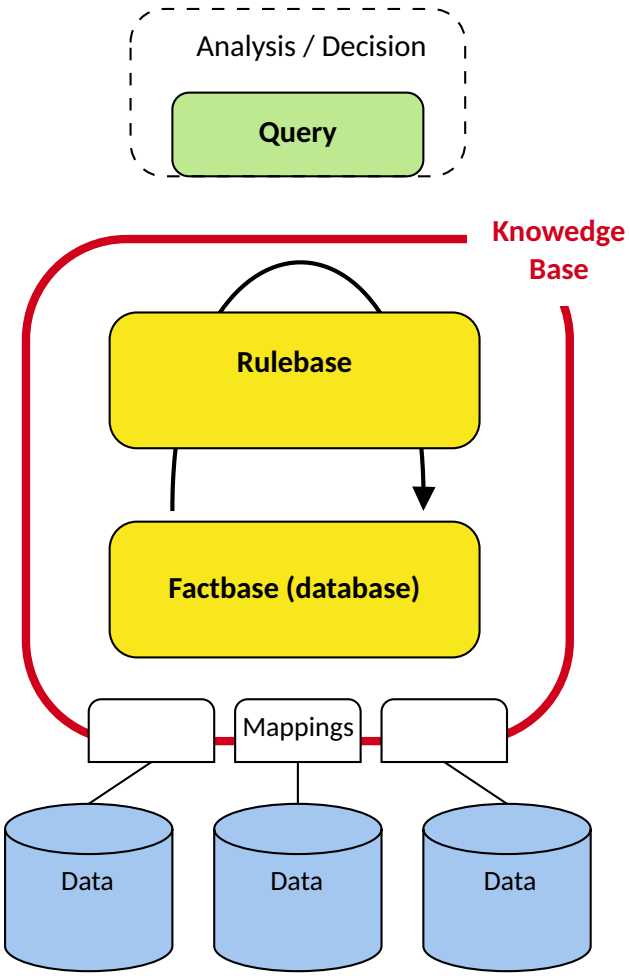
Knowledge-Based Data Management with Rules



Three layer architecture (common to DI and OBDA)

Sources Mappings Knowledge Base (KB)

Knowledge-Based Data Management with Rules



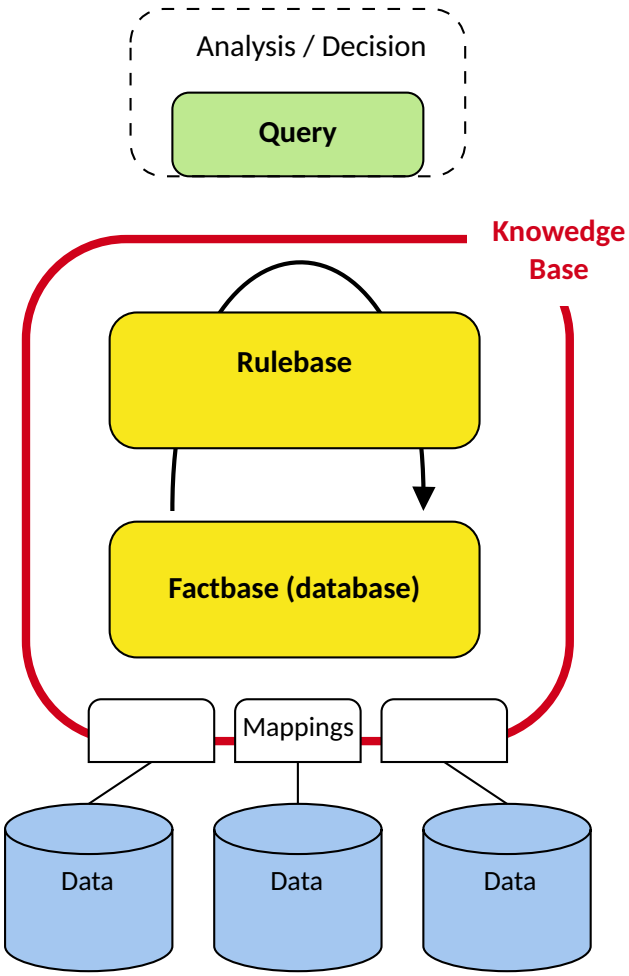
Three layer architecture (common to DI and OBDA)

Sources Mappings Knowledge Base (KB)

Domain Knowledge "the glue"

- Start with a conceptual model of the domain
- Semantics: map sources to KB
- Abstraction: translate KB services towards data

Knowledge-Based Data Management with Rules



Three layer architecture (common to DI and OBDA)

Sources Mappings Knowledge Base (KB)

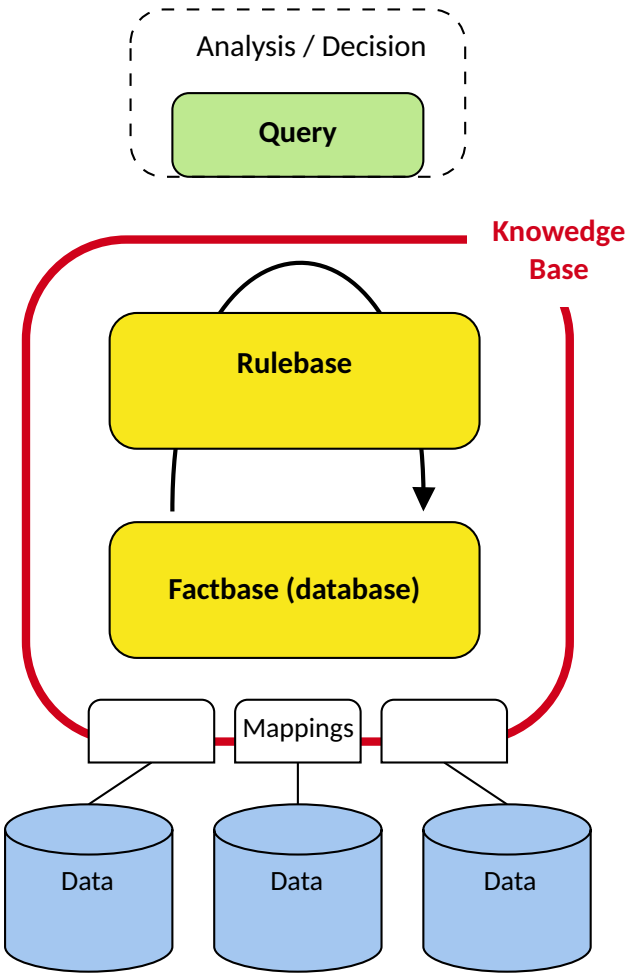
Domain Knowledge "the glue"

- Start with a conceptual model of the domain
- Semantics: map sources to KB
- Abstraction: translate KB services towards data

Rule-based Languages

- Declarativity, Modularity, Reusability
- Expressivity: *"one language to rule them all"* ontologies, data-processing and data-quality
- Explainability

Knowledge-Based Data Management with Existential Rules



Existential Rules *[Baget+ IJCAI 09][Cali+ PODS 09]*

$$\varphi(XY) \longrightarrow \exists Z.\psi(XZ)$$

positive function-free conjunctions

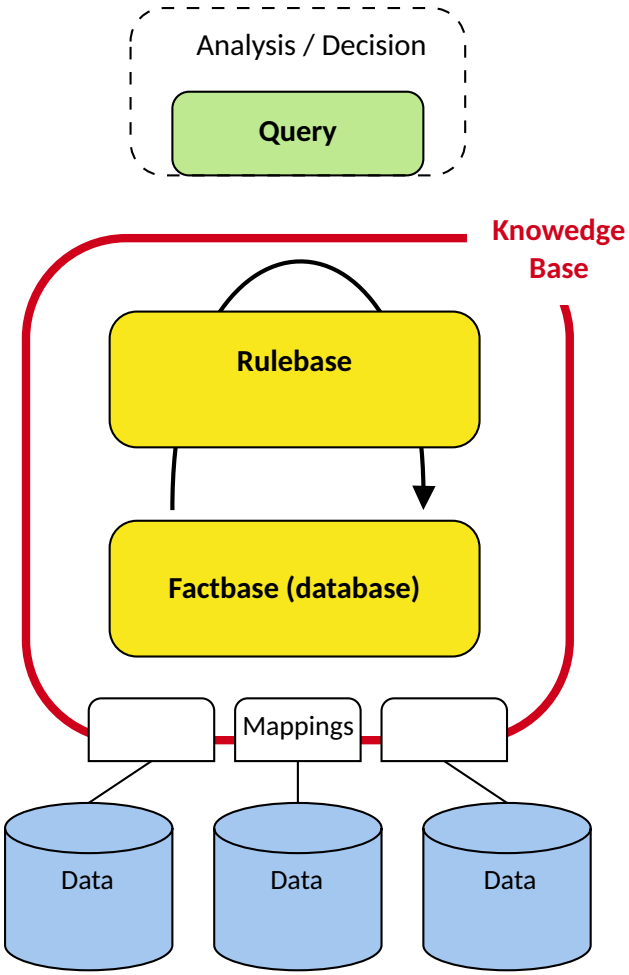
$$\forall x.\ Plant(x) \wedge Contamined(x) \longrightarrow \exists z.affectedBy(x,z) \wedge Agent(z)$$

Advantages

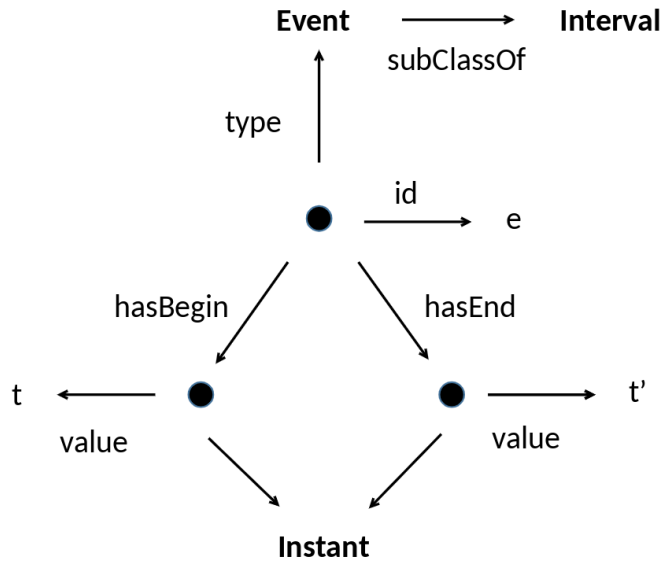
1. infer existence of unknown entities & values
   - reasoning in open domains, incomplete data
2. model complex relationships & n-ary relations
   - key for declarative data-processing & quality
3. generalize KR (DL) & DB (Datalog) languages

Data Models and Storage

Models and Storage



- Can go beyond RDF/graphs as a data-model for integration
  - but can still export and import RDF



Event(e,t,t')

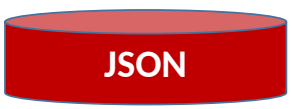
- Reuse existing single/multi-store DBMS
  - both external and internal storage

Data Access

Data Access

Query :
“retrieve all researchers name”

```
{ name : Patrice,
  UMR : IATE }
```




```
emp name Pascal
 emp UMR Mistea
```


```
(Christian,ABSys)
```

Source to KB Mappings

Researcher(NAME,INSTITUTE,LAB)

```
collection().
[name:$x1,UMR:$x2]
```

—>

Researcher($x1,'INRAE',$x2)

```
{ name : Patrice,
  UMR : IATE }
```

JSON

```
SELECT ?v1 ?v2
WHERE {?x name ?v1 ,
       ?x UMR  ?v2 }
```

—>

Researcher(?v1,'INRAE',?v2)

```
emp name Pascal
 emp UMR Mistea
```

RDF

```
SELECT nom, umr
   FROM emp
```

—>

Researcher(nom,'INRAE',umr)

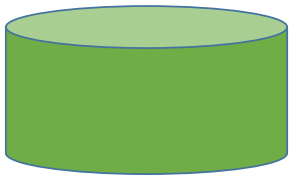
```
(Christian,ABSys)
```

SQL

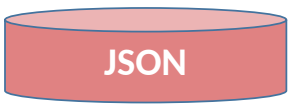
Strategies

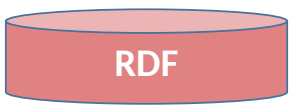
Materialization

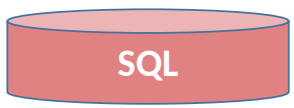
Query :
“retrieve all researchers name”



Researcher(Patrice, INRIA, IATE)
Researcher(Pascal, INRIA, Mistea)
Researcher(Christian, INRIA, ABSys)

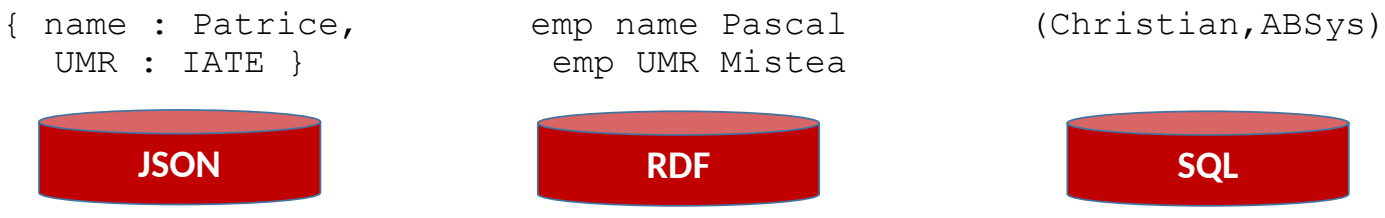






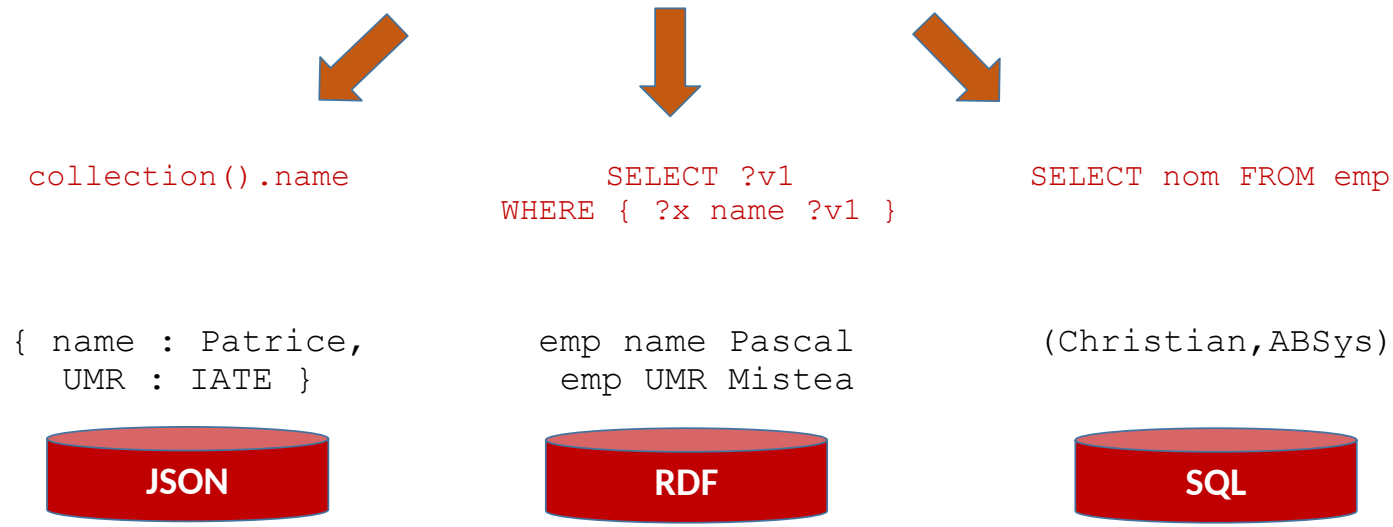
Virtualization

Query :
“retrieve all researchers name”



Virtualization

Query :
“retrieve all researchers name”



Reasoning

Rules for Data Processing

(example from Elie's Najm PhD on the analysis of agro-ecological systems)

contre(fongicideBio,oidium)

agitApres(fongicideBio, 0)

active(T,J+X) if agitApres(T,X) applicationTechnique(T,J)

vigneTotalementProtegeeContre(J,M) if technique(T) contre(T,M) maladie(M)
efficacite(T, totale) active(T,J)

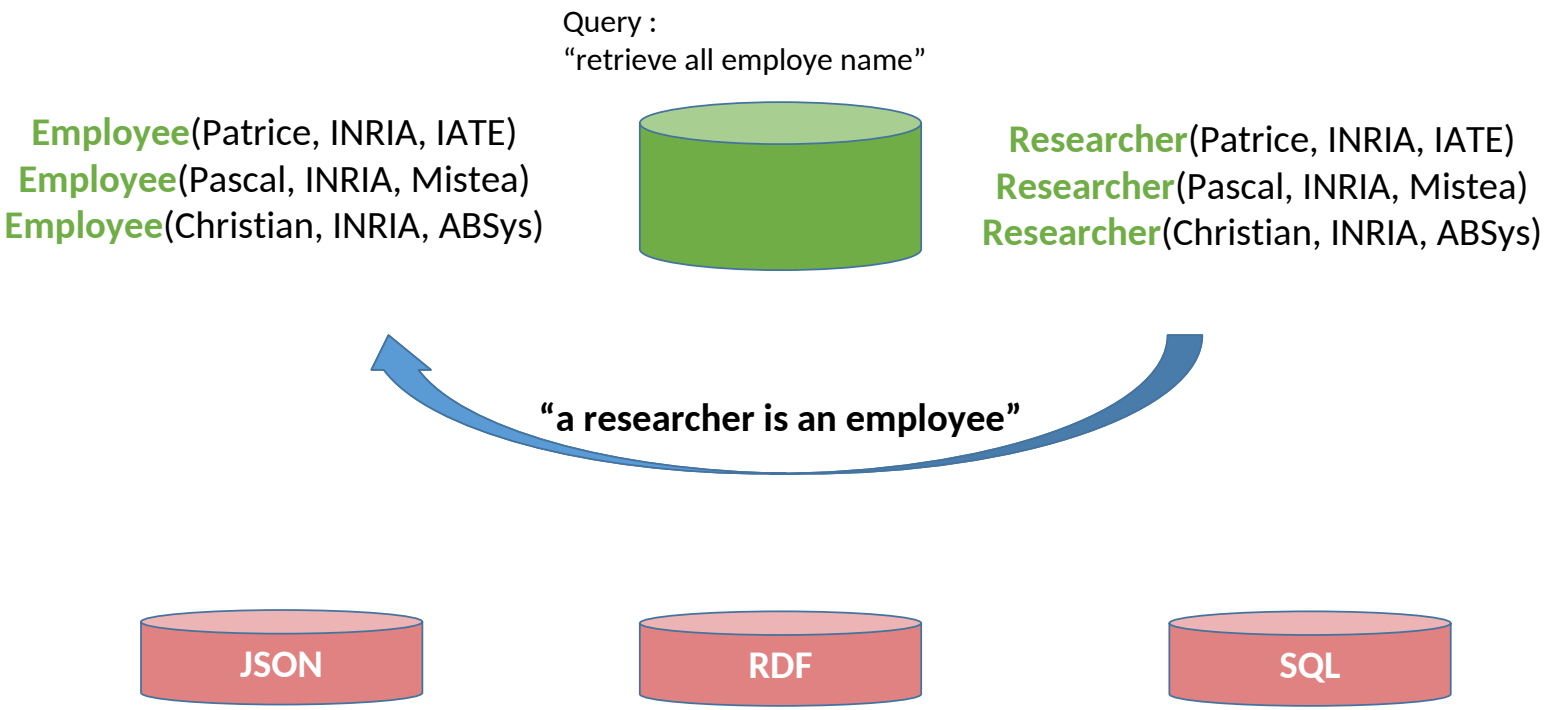
dangerEleve(J, M) if risqueEleve(J, M) ¬ vigneProtegeeContre(J,M)

Knowledge and Rules

Query :
“retrieve all employe name”

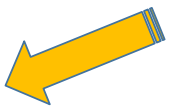
“a researcher is an employee”

Saturation


Query :
“retrieve all employe name”
Employee(Patrice, INRIA, IATE)
Employee(Pascal, INRIA, Mistea)
Employee(Christian, INRIA, ABSys)
Researcher(Patrice, INRIA, IATE)
Researcher(Pascal, INRIA, Mistea)
Researcher(Christian, INRIA, ABSys)
“a researcher is an employee”
JSON
RDF
SQL

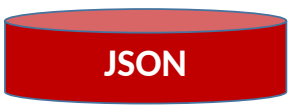
Reformulation

Query :
“retrieve all employe name”

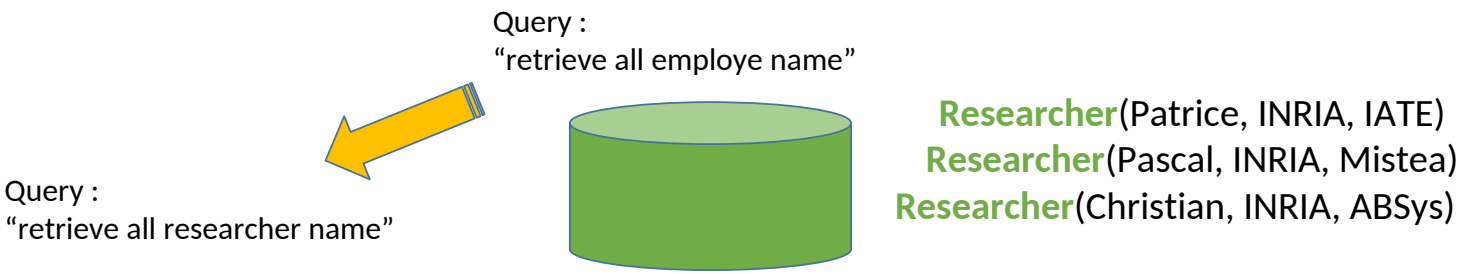


Query :
“retrieve all researcher name”

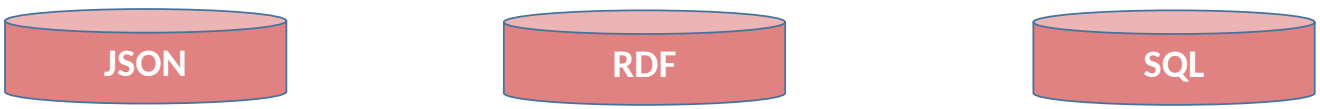
“a researcher is an employee”



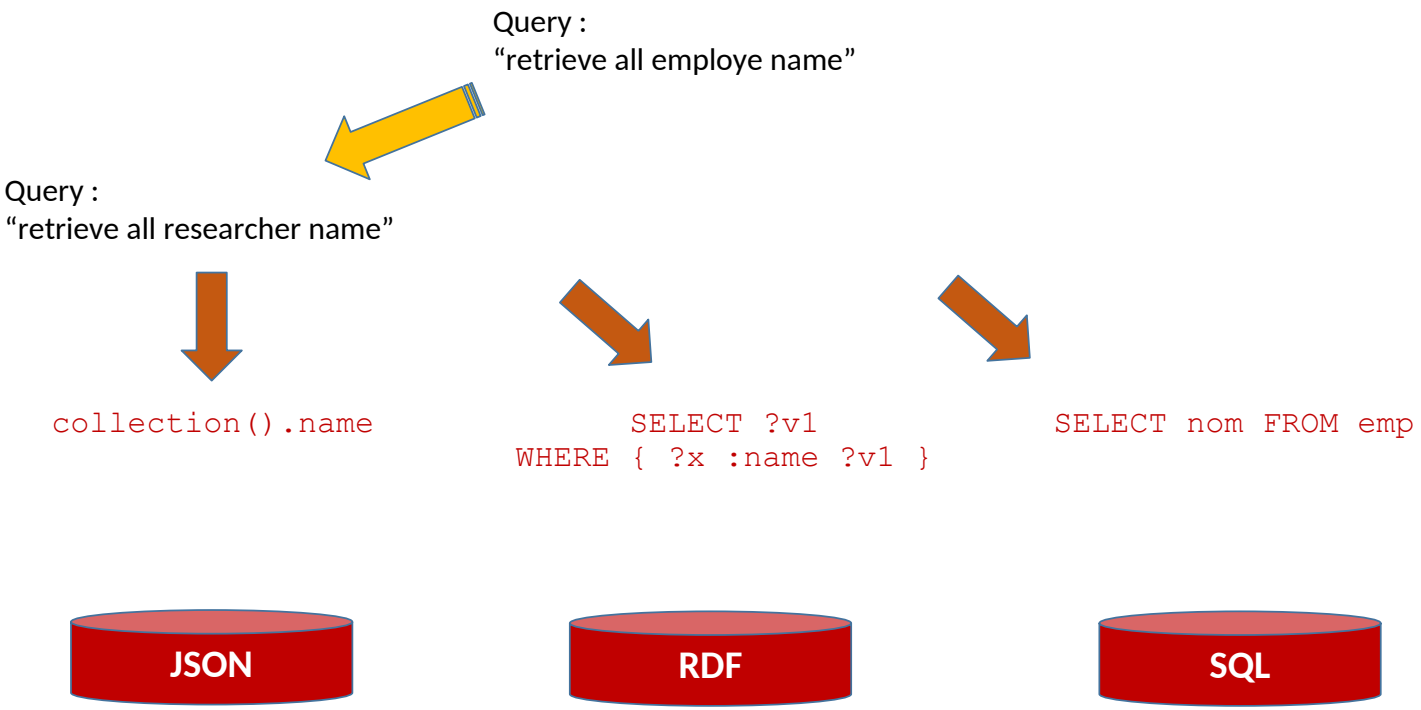
Reformulation+Saturation



“a researcher is an employee”

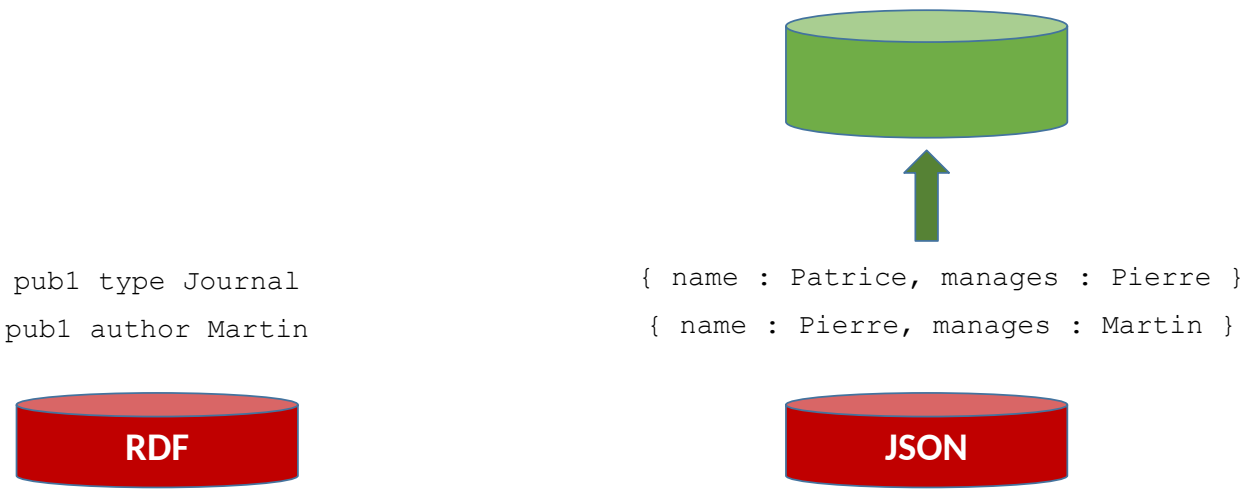


Reformulation+Virtualization



Combining the Approaches

Query :
“retrieve all publications by a researcher in the team headed by Patrice”

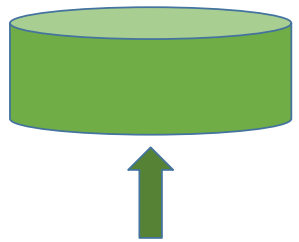


Combining the Approaches

Query :
“retrieve all publications by a researcher in the team headed by Patrice”

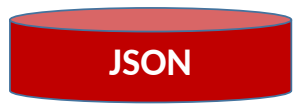
Journal < Publication

isTransitive(ManagerOf)



```
pub1 type Journal
pub1 author Martin
```

```
{ name : Patrice, manages : Pierre }
{ name : Pierre, manages : Martin }
```

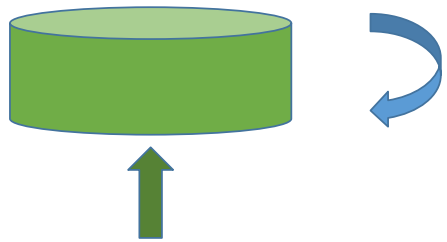


Combining the Approaches

Query :
“retrieve all publications by a researcher in the team headed by Patrice”

Journal < Publication

isTransitive(ManagerOf)



```
pub1 type Journal
pub1 author Martin
```

```
{ name : Patrice, manages : Pierre }
{ name : Pierre, manages : Martin }
```

RDF

JSON

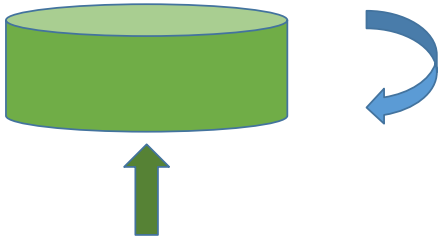
Combining the Approaches

Query :
“retrieve all publications by a researcher in the team headed by Patrice”

“retrieve all journals with author”

Journal < Publication

isTransitive(ManagerOf)



```
pub1 type Journal
pub1 author Martin
```

```
{ name : Patrice, manages : Pierre }
{ name : Pierre, manages : Martin }
```

RDF

JSON

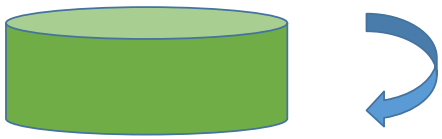
Combining the Approaches

Query :
"retrieve all publications by a researcher in the team headed by Patrice"

"retrieve all journals with author"

Journal < Publication

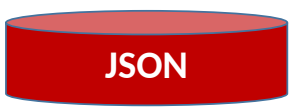
isTransitive(ManagerOf)



```
SELECT   ?pub ?auth
WHERE  { ?pub name ?auth.
         ?pub type Journal}
```

```
pub1 type Journal
pub1 author Martin
```

```
{ name : Patrice, manages : Pierre }
{ name : Pierre, manages : Martin }
```



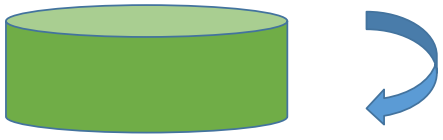
Combining the Approaches

Query :
"retrieve all publications by a researcher in the team headed by Patrice"

"retrieve all journals with author"

Journal < Publication

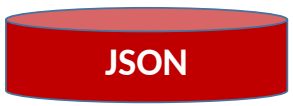
isTransitive(ManagerOf)



```
SELECT   ?pub ?auth
WHERE  { ?pub name ?auth.
         ?pub type Journal}
```

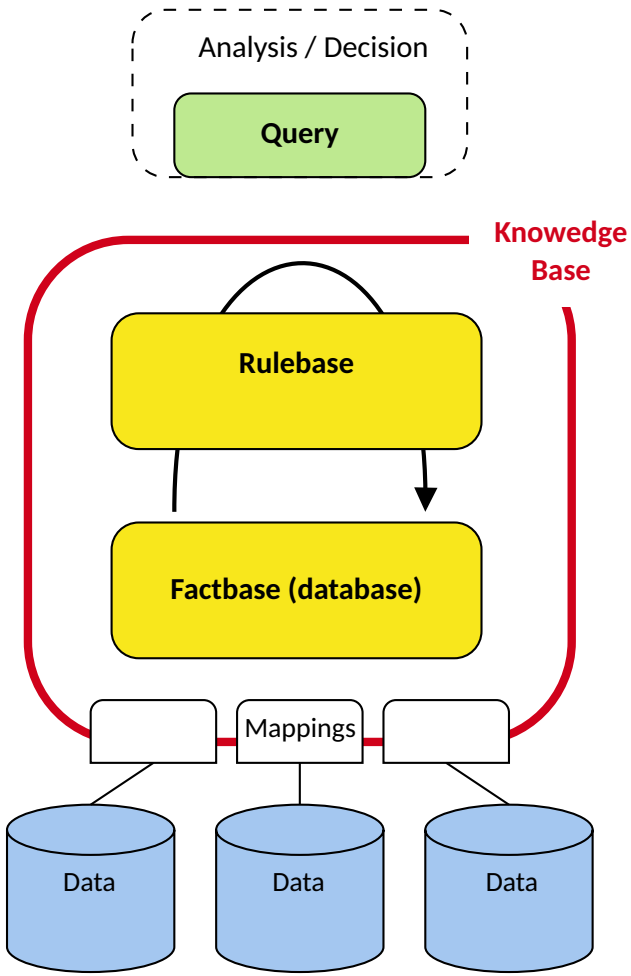
```
pub1 type Journal
pub1 author Martin
```

```
{ name : Patrice, manages : Pierre }
{ name : Pierre, manages : Martin }
```



A landscape of data-access possibilities, each with a different cost.

Knowledge-Based Data Management with Existential Rules

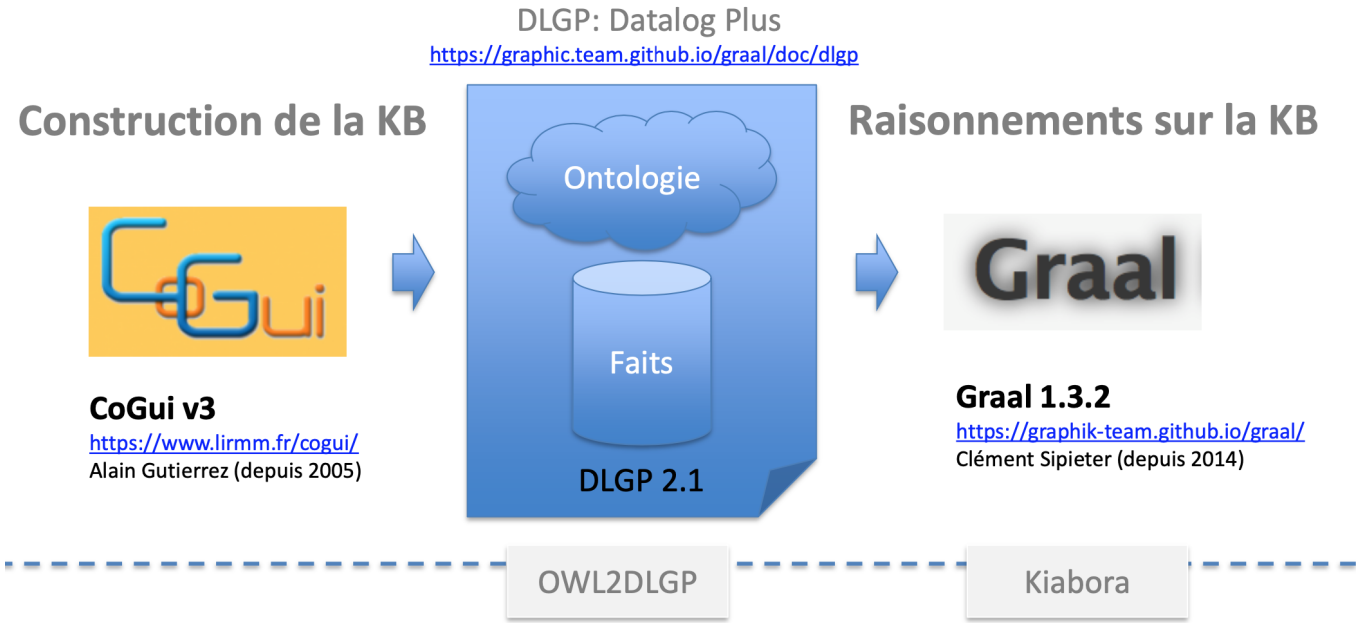


Wrapping up the main ideas:

1. three layers architecture
2. can go beyond triples
3. reuse database technology
4. rules for data-processing
5. intelligent access to data
6. (and more) KBDM data-flows

Zoom on Graal V2

Software Ecosystem of the Inria GraphIK Team



- Graal V2 : KBDM with Extended ER — Florent Tornil (2021-2023)
- Use cases : (1) implementing and testing algorithms (2) building applications
- First release beginning 2022 — AIC T-CALIS-FAIR

Graal V2 : What's New ?

- Architecture
  - Genericity, Reusability, Extensibility (objects, storage, algorithms)
- Extended Existential Rules
  - filters & functions $\big(infected/population\big) \geq 0.7$
  - negation $Contamined(plant) \wedge \neg Treated(plant, pesticide)$
  - (next) aggregates, disjunction
- Semantic Web
  - RDF(S), SPARQL (BGP), OWL 2 (Horn theories)
- Mappings and Internal Storage
  - SQL (Postgres, HSQL, MySQL), Triplestores (Virtuoso, Jena), Graphs (Neo4J)
  - (next) Key-value stores, CSV, Mediators/Polystores
- Data Access and Reasoning Strategies
  - Materialization, Virtualization, Saturation, Reformulation
  - (next) compilation, mixed approaches
- More
  - Rulebase analysis (KIABORA), Automatic benchmarking (BRUNNER)
  - (next) Python bridge (PyGraal)