Federation of relational databases and data lakes

With Virtual Knowledge Graphs

Benjamin Cogrel, CTO and co-founder

Data-Centric Architecture Forum, Fort Collins, June 7, 2022

ONTOPIC

About me

- Former researcher at the Free University of Bozen-Bolzano (Italy) on Virtual Knowledge Graphs (aka Ontology-Based Data Access)
- Now CTO and co-founder of Ontopic (spin-off)
- Core developer of Ontop, an open-source VKG engine (now included in GraphDB and Allegrograph)

Agenda

- Database federators
- Virtual Knowledge Graphs
- Performance

Database federators

- Also called data virtualization platforms, data lake engines or data lakehouses
- Dremio, Apache Spark and Teiid (open-source), Denodo (commercial)
- SQL query processors
  - Hide dialects/No-SQL query languages and file formats
  - Plan, execute distributed joins
  - Can materialize parts of the data (on-demand)

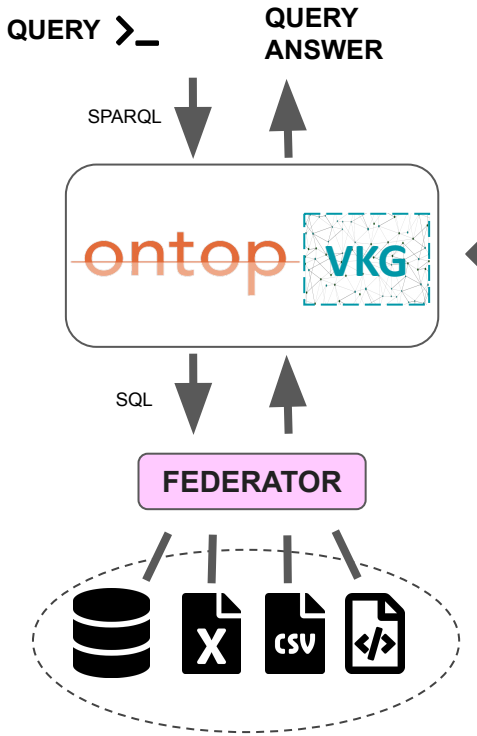
Dealing with files from data lakes

- Made accessible as relational views
  - Directories of files with the same structure can be merged into a single view

- Nested data
  - JSON arrays, multivalued fields (common in CSV)
  - Can be flattened into new views

This data becomes queryable using plain SQL over the flattened views

Virtual Knowledge Graphs (OBDA)



Specification:

- Mapping (R2RML)
- Ontology (OWL 2 QL)

ONTOPIC

Virtual Knowledge Graphs (OBDA)

- SPARQL queries are translated into SQL
- Relying on a mapping (typically R2RML or a dialect variant)

- Combined with database federators, can integrate multiple heterogeneous data sources

- VKG engines
  - Ontop (standalone or embedded in GraphDB or Allegrograph)
  - Stardog (virtual graph component)

Performance

1. Optimization at the VKG engine level

2. Acceleration at the database federator level

Optimization at the VKG level

Intensive use of structural information for generating efficient SQL queries

- Incompatible IRI templates helps pruning unions
  - "No point joining these 2 tables together"

- Information about the source structures
  - Integrity constraints (e.g. unique constraints, foreign keys)
  - Nullability information
  - Minimizes the number of inner and left joins coming from the SPARQL query

Acceleration at the database federator level

- Trade-off between materialization and federation

- Dremio
  - Different forms of materialization
    - Materialization of an existing view
    - Data-cube-like for accelerating aggregate queries
  - Often no need to change the SQL queries
    - No view creation
    - Keep the same mapping
    - Relies of the query planner of Dremio

Remarks on SPARQL federation

- Far less structural information than SQL federation
  - Much less optimization opportunities
  - Things could go better with SHACL

- I am not aware of acceleration capabilities comparable to those offered by Dremio

Thank you!

Ontop: https://ontop-vkg.org
Dremio: https://dremio.com

Ontopic: https://ontopic.ai

Reach me at
benjamin.cogrel@ontopic.ai