UNIVERSITÀ DEGLI STUDI DI TRENTO

Dipartimento di Ingegneria e Scienze dell'Informazione
Corso di laurea in informatica

Elaborato finale

TITOLO ELEBORATO

Supervisore:
Prof. Bouquet Paolo

Laureando:
Corte Pause Manuela

Anno Accademico 2021/2022

Indice

1 Introduzione **1**

2 Virtual Knowledge Graphs **3**
2.1 Cos'è un Virtual Knowledge Graph 3
2.2 Il Virtual Knowledge Graph system Ontop 3
2.2.1 Intermediate Query language 3
2.2.2 Esempi di utilizzo 3

3 Progetto **5**
3.1 Ontopic 5
3.2 Il modulo bi-connector 5
3.2.1 Creazione database 5
3.2.2 Parsing di query SQL 5
3.3 Stato iniziale bi-connector 5
3.4 Analisi prerequisiti 6
3.5 Costrutti implementati 6
3.5.1 Distinct, Limit e Offset 6
3.5.2 Ordinamento righe 6
3.5.3 Combinazione tabelle 6
3.5.4 Operazioni insiemistiche 6
3.5.5 Aggregazione 6
3.6 Risultati ottenuti 6

4 Conclusioni **9**
4.1 Possibili sviluppi futuri 9

Capitolo 1

Introduzione

Riassunto introduttivo di quello che sarà l'argomento dell'elaborato

Capitolo 2

Virtual Knowledge Graphs

2.1 Cos'è un Virtual Knowledge Graph

Invenzione VKG da parte di Google
Vantaggi di un VGK
Struttura di un vkg: ontology, mapping, schema

2.2 Il Virtual Knowledge Graph system Ontop

VKG system open source che segue gli standard W3C

2.2.1 Intermediate Query language

Ontop era inizialmente basato su Datalog poi passato ad un proprio linguaggio interno (Intermediate Query)

IQ: rappresentazione usata per tradurre le query degli utenti in SPARQL nelle query SQL dei mapping

2.2.2 Esempi di utilizzo

Esempi di utilizzo di Ontop in ambienti aziendali e accademici

Capitolo 3

Progetto

3.1 Ontopic

Breve descrizione dell'azienda e di cosa si occupa (non so bene se mettere questa sezione qui o nel capitolo introduttivo)

3.2 Il modulo bi-connector

Bi-connector: progetto con lo scopo di poter usare strumenti di BI come Tableau su ontologie. (a differenza di altre soluzioni NoSQL come MongoDB non esistono connettori pre-forniti)

3.2.1 Creazione database

Dai file dell'ontologia (.ttl) vengono estratte delle viste e con queste viste viene creato in locale un database PostgreSQL. (Per visualizzarlo ho usato DBeaver con la classe ProfJDBC.java per eseguire i test)

3.2.2 Parsing di query SQL

Su questo database creato inizialmente è possibile fare query SQL. Viene fatto il parsing di queste query e vengono tradotte in un albero di nodi IQ che viene poi usato dall'ontologia per rispondere alla query.

3.3 Stato iniziale bi-connector

Quando ho iniziato io il modulo già esisteva ed era funzionante per query del tipo SELECT-JOIN-WHERE e veniva utilizzato il parser SQL interno di Ontop che era però molto limitato essendo stato pensato inizialmente per le query usate nei mapping che sono tipicamente semplici (union of conjunctive queries). Avendo un insieme di query riconosciute molto limitato, molte delle query erano o riscritte in una forma semplificata o non supportate.

3.4 Analisi prerequisiti

Passaggio dal parser di Ontop a JSqlParser una volta uscita la versione 4.

Analisi di quali fossero i costrutti usati nelle query automaticamente generate da Tableau (ad esempio viene usato moltissimo il GROUP BY) e analisi di quale fosse il comportamento specifico di PostgreSQL su queste keyword (e.g. in PostgreSQL la funzione CONCAT non è null-rejecting, MINUS non è supportato, tipo di NULL ordering di default, ...)

3.5 Costrutti implementati

3.5.1 Distinct, Limit e Offset

Distinct implementato tramite un semplice IQ node Distinct Limit e Offset implementati con un filtro Interessanti principalmente in quanto sono stati un primo approccio sia alle struttura generale del progetto / IQTree che a JSqlParser più che come funzionalità complesse da implementare.

3.5.2 Ordinamento righe

Order by più complesso in quanto richiede la creazione di comparatori, operazioni di sostituzione per la proiezione delle variabili e la gestione del NULL ordering.

3.5.3 Combinazione tabelle

Cross e inner join già presenti, implementazione left join (scontato di conseguenza il right join) Problematiche sorte su colonne con stessi nomi

3.5.4 Operazioni insiemistiche

Operazioni su insiemi (unione e sottrazione) implementate con alcune restrizioni. L'implementazione della sottrazione è interessante (implementata come filtro su un left join).

3.5.5 Aggregazione

Funzioni di aggregazione (SUM, COUNT, MIN, MAX, AVG) per cui è stato necessario introdurre una funzionalità che ritardasse l'assegnazione del tipo alla funzione. (Questo perché SPARQL a differenza di SQL usa una tipizzazione dinamica). Costrutto Group by e having (interessante l'implementazione per funzioni con la sostituzione dei functional term con variabili)

3.6 Risultati ottenuti

Con l'introduzione dell'aggregazione (e anche dell'order by) è stato possibile rimuovere buona parte delle query automaticamente create da Tableau -> accesso a Tableau

e prime dashboard create su dataset non banali (chiedere a Benjamin se ha qualche screenshot)

Capitolo 4

Conclusioni

Importanza tirocinio dal punto di vista formativo e personale (prima esperienza di lavoro con un gruppo di programmatori, esperienza su una codebase di una certa dimensione, importanza testing in ambito aziendale, interesse personale nel mondo del Data Integration, ...)

4.1 Possibili sviluppi futuri

Allargare l'insieme delle query supportate (funzioni su date, cursori, meccanismi di autenticazione, ...)

Supportare altri strumenti di BI (PowerBI, Qlik, ...)

Bibliografia

[1] Elena Botoeva, Diego Calvanese, Benjamin Cogrel, Julien Corman, and Guohui Xiao. Ontology-based data access – beyond relational sources. *Intelligenza Artificiale*, 13:21–36, 2019. 1.

[2] Diego Calvanese, Benjamin Cogrel, Sarah Komla-Ebri, Roman Kontchakov, Davide Lanti, Martin Rezk, Mariano Rodriguez-Muro, and Guohui Xiao. Ontop: Answering sparql queries over relational databases. *Semantic Web*, 8(3):471–487, 2017.

[3] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, et al. Knowledge graphs. *Synthesis Lectures on Data, Semantics, and Knowledge*, 12(2):1–257, 2021.

[4] JSqlParser. Jsqlparser sourceforge. `http://jsqlparser.sourceforge.net/`.

[5] Ontop. Intermediate query (iq). `https://ontop-vkg.org/dev/internals/iq.html`.

[6] Guohui Xiao, Linfang Ding, Benjamin Cogrel, and Diego Calvanese. Virtual knowledge graphs: An overview of systems and use cases. *Data Intelligence*, 1(3):201–223, 2019.

[7] Guohui Xiao, Davide Lanti, Roman Kontchakov, Sarah Komla-Ebri, Elem Güzel-Kalaycı, Linfang Ding, Julien Corman, Benjamin Cogrel, Diego Calvanese, and Elena Botoeva. The virtual knowledge graph system ontop. In *International Semantic Web Conference*, pages 259–277. Springer, 2020.