

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
1.1	Data integration	1
2	Knowledge Graphs	3
2.1	Cos'è un Knowledge Graph	3
2.2	Virtual Knowledge Graph	4
2.3	Il Virtual Knowledge Graph system Ontop	4
2.3.1	Intermediate Query language	4
2.3.2	Esempi di utilizzo	4
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

1.1 Data integration

I dati sono diventati una parte sempre più fondamentale della nostra vita e ancor di più in quella delle aziende al fine dell'assisterle nel processo di decision-making. Questi dati provengono da molteplici fonti come social network, tracking, sensori di IoT, ... e risultano in una mole enorme di dati non strutturati e di conseguenza complessi da sfruttare per ricavarne informazioni rilevanti. Proprio per questo il mondo della data integration è così importante [5].

Possiamo definire la data integration come il problema del combinare dati a provenienti da livelli diversi e fornire all'utente finale una visione unificata di questi dati [4]. E' facile vedere quindi come questo concetto si adatti bene in un'azienda che utilizza vari sistemi, applicazioni e piattaforme ognuna delle quali produce o raccoglie dati senza tenere conto degli altri applicativi (data silos). Oltre a questi applicativi, i dati che si vogliono utilizzare non devono essere necessariamente interni all'azienda, ma possono anche essere esterni come ad esempio dataset da internet.

Esistono approcci diversi alla data integration e quello più tradizionale è certamente quello dei data warehouse, ovvero tutti i dati sono combinati e memorizzati in un solo posto (tipicamente un database). Questo processo di "combinazione" è definito ETL (Extract, Transform, Load) e permette di rilevare e correggere inconsistenze tra i dati prima che venga fatto il merge di questi e permette inoltre di integrare diversi tipi di dati e visualizzarli poi con un'unica vista complessiva.

Questa soluzione risulta complessa da utilizzare per dataset che vengono modificati frequentemente e richiedono quindi che il processo di ETL, che risulta essere molto costoso, venga re-eseguito molte volte. Anche per questo si è quindi passati ad un paradigma basata sul *loose coupling* ovvero un'interfaccia sulla quale eseguire query che vengono poi mappate ed eseguite sulle sorgenti originali eliminando il problema dell'avere informazioni non aggiornate.

Oltre alla discussione sull'architettura del sistema di data integration anche l'aspetto semantico risulta importante al fine di evitare la collisione di termini uguali usati all'interno delle sorgenti con significati diversi. E' proprio da questa considerazione che nascono approcci basati su ontologie definiti come Ontology Based Data

Access (OBDA). Queste soluzioni mitigano il problema appena descritto fornendo un vocabolario comune da utilizzare.

Capitolo 2

Knowledge Graphs

2.1 Cos'è un Knowledge Graph

Si inizia a parlare di rappresentazione della conoscenza tramite l'aiuto di knowledge base già dalla fine degli anni 50 e nel 1980 ricercatori dell'università di Groningen e dell'università di Twente nei Paesi Bassi usarono per la prima volta il termine Knowledge Graph per descrivere il loro sistema basato sull'integrazione di molteplici sorgenti di dati per rappresentare il linguaggio naturale tramite una knowledge base. Questo primo momento di ricerca iniziale fu poi seguito all'inizio degli anni 2000 dall'affermazione degli standard W3C come RDF e OWL nell'ambito del Semantic Web e dal sorgere di varie ontologie pubbliche come DBPedia, YAGO e Freebase. [1] [3]

Il termine Knowledge Graph nasce però solo nel 2012 con il motore di ricerca di Google che introduce il termine per descrivere le nuove funzionalità di ricerca semantica del proprio motore di ricerca: le ricerche che vengono effettuate non sono più semplice string matching, ma viene aggiunta una componente di ragionamento di grado di riconoscere veri e propri "oggetti" del mondo reale. [1]

La definizione precisa di cosa sia un knowledge graph rimane nebulosa e definizioni differenti risultano essere a volte in contraddizione l'una con l'altra. In modo molto generale possiamo definire un knowledge graph come una struttura che rappresenta la conoscenza come un insieme di concetti e le relazioni fra essi. Se vogliamo invece dare una definizione più formale possiamo definire un knowledge graph come una struttura che acquisisce e integra informazioni in una knowledge base e applica un motore d'inferenza per ricavare nuova conoscenza come mostrato in figura 2.1. Molte definizioni indicano anche la dimensione del KG come fattore determinante, ma cosa questo significhi nel dettaglio non è mai specificato.

La knowledge base è spesso implementata praticamente tramite ontologia e può quindi sfruttare tutte le caratteristiche che rendono i grafi un'ottima scelta per rappresentare conoscenza. In particolare i grafi risultano essere ottimi per modellare domini complessi in quanto gli archi possono rappresentare relazioni tra diversi nodi, che rappresentano i soggetti, anche molto complesse. Inoltre la mancanza di uno schema definito a priori permette all'insieme dei dati di crescere in modo più flessibile e i linguaggi di interrogazione per sistemi basati su grafi sono molto espressivi e contengono la maggior parte dei costrutti usati da linguaggi di query più tradizionali come join, unioni, proiezioni, ... [2].

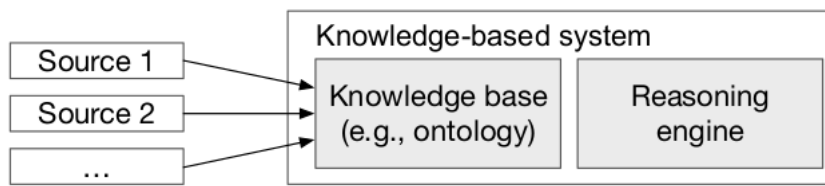


Figura 2.1. Struttura di un KG

2.2 Virtual Knowledge Graph

Specializzazione rispetto a un normale KG

Struttura di un vkg: ontology, mapping, schema

2.3 Il Virtual Knowledge Graph system Ontop

VKG system open source che segue gli standard W3C

2.3.1 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

2.3.2 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 alla 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] Lisa Ehrlinger and Wolfram Wöß. Towards a definition of knowledge graphs. *SEMANTiCS (Posters, Demos, SuCCESS)*, 48(1-4):2, 2016.
- [2] 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.
- [3] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 33(2):494–514, 2021.
- [4] Maurizio Lenzerini. Data integration: A theoretical perspective. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 233–246, 2002.
- [5] Bhavana Sayiram. Importance of data integration in the data decade. https://education.dellemc.com/content/dam/dell-emc/documents/en-us/2021KS_Sayiram-Data_Integration.pdf.