Relazione MAADB - NoSQL lab project

Roberto Pegoraro (matr.981463) e Luca Schiavon (matr.987231)

Introduzione

In questa relazione viene descritto lo sviluppo di un'applicazione web per l'interrogazione del database LDBC (Linked Data Benchmark Council) utilizzando due database NoSQL: un database a grafo (Neo4J) e un database document-based (MongoDB).

Architettura del sistema

Il sistema è stato diviso in tre server differenti implementati in Node.js utilizzando il framework Express e axios per la comunicazione tra di essi. Ogni server ha compiti distinti con l'obiettivo di collaborare per rispondere alle interrogazioni dell'utente.

Il server principale è il MainServer, corrispondente alla porta 3000 del localhost, ha lo scopo di interfacciarsi con l'utente e coordinare le comunicazioni con gli altri due server. All'interno della cartella *javascript* contenuta nella cartella *public* troviamo il file *index.js* dove vengono dichiarate le funzioni richiamate quando l'utente decide di eseguire un'interrogazione. Queste funzioni comunicano con il file *index.js* presente nella cartella *routes* che si occupa della gestione delle richieste HTTP e del routing tra server.

Il server MongoDbServer, corrispondente alla porta 3001 del localhost, è il server utilizzato per la gestione dei dati statici del sistema. Per l'architettura di questo server si è seguito il pattern Model-View-Controller, infatti:

- i controller svolgono la funzione di monitorare la logica delle singole entità occupandosi di eseguire le operazioni sui dati e di monitorare che le regole applicative sui valori siano rispettate
- il file di routing *index.js* definisce gli endpoint REST, ovvero i punti di accesso che il MainServer può richiamare

Le istruzioni presenti nel file *database.js* permettono la connessione al database MongoDB denominato "MAADB_Project".

Il server Neo4jServer, corrispondente alla porta 3002 del localhost, si occupa della gestione degli altri dati. In questo server oltre alle routes e ai controller è presente anche un layer di services che permette lo svolgimento delle query Cypher. Le istruzioni presenti nel file *database.js* permettono la connessione al database Neo4j denominato "neo4j".

La decisione di creare un'architettura distribuita su tre server permette una miglior distribuzione del carico di lavoro, in modo che ogni server possa focalizzarsi sul compito principale per cui è stato creato. Questo approccio inoltre migliora anche la manutenzione del sistema.

Dataset

Il dataset utilizzato per il progetto è stato generato attraverso ldbc_snbdatagen_spark, una scelta motivata dalla maggiore semplicità di configurazione rispetto alla versione Hadoop. Il dataset LDBC simula un social network con diverse entità interconnesse che verranno illustrate in seguito.

Distribuzione dei dati

La distribuzione dei dati tra le due tipologie di database è stata progettata per utilizzare al massimo le caratteristiche di essi.

MongoDB gestisce le entità che presentano attributi descrittivi e relazioni statiche, quali:

- Entità:
    - Organisation
    - Place
    - Tag
    - TagClass
- Relazioni:
    - organisation_isLocatedIn_place
    - place_isPartOf_place
    - tag_hasType_tagClass
    - tagClass_isSubClassOf_tagClass

Neo4j gestisce le relazioni dinamiche e sono state salvate le entità principali utili ai fini della nostra applicazione. Troviamo anche alcuni set dove è presente solo il campo id, come le entità *Organisation*, *Tag* e *Place*, utile per facilitare il compito di interrogazione.

- Entità:
    - Comment
    - Forum
    - Organisation
    - Person
    - Place
    - Post
    - Tag
- Relazioni:
    - comment_hasCreator_person
    - comment_hasTag_tag
    - comment_replyOf_comment
    - comment_replyOf_post
    - forum_containerOf_post
    - forum_hasMember_person
    - forum_hasModerator_person
    - forum_hasTag_tag
    - person_hasInterest_tag
    - person_isLocatedIn_city
    - person_knows_person
    - person_likes_comment
    - person_likes_post
    - person_studyAt_university
    - person_workAt_company
    - post_hasCreator_person
    - post_hasTag_tag

Implementazione delle query

Le tre query lookup sviluppate sono:

1. “Trovare il top creator per un dato forum” solo su Neo4j
2. “Trovare le trending tagClass di una data nazione” cross-database
3. “Trovare i tag più frequenti di una data università” cross-database

Le tre query analitiche sviluppate sono:

1. “Fornire le conoscenze suggerite basandosi sulle conoscenze in comune” solo su Neo4j
2. “Fornire le conoscenze suggerite basandosi sugli interessi in comune” solo su Neo4j
3. “Trovare l’età media di una nazione” cross-database

Conclusioni

La realizzazione di questo progetto ci ha permesso di consolidare competenze sia tecniche sia metodologiche, mettendo in pratica concetti teorici in un contesto concreto.
Abbiamo approfondito le tecniche per integrare tra di loro database differenti, comprendendone punti di forza e limiti, e migliorato la nostra capacità di progettare schemi coerenti. Allo stesso tempo, il lavoro ci ha insegnato l'importanza della pianificazione, della collaborazione e della risoluzione progressiva dei problemi.