Abstrakt zum Vortrag im Oberseminar Graphdatenbanken und Graph-Frameworks

Duy Hung Nguyen

7. Juni 2018

1 Grundlagen

1.1 Aufbau von Graphen

Ein Graph besteht aus zwei Elementen: einem Knoten und einer Beziehung. Jeder Knoten repräsentiert eine Entität (eine Person, ein Ort, eine Sache oder ein anderes Datenelement), und jede Beziehung stellt dar, wie zwei Knoten verknüpft sind.

Diese allgemeine Struktur erlaubt alle Arten von Szenarien zu modellieren, das durch Beziehungen definiert wird.

1.2 Graphdatenbank

Eine Graphdatenbank ist eine Datenbank, die Graphen benutzt, um stark vernetzte Informationen darzustellen und abzuspeichern. Sie ist ein Online - Datenbankverwaltungssystem mit Operationen zum Erstellen, Lesen, Aktualisieren und Löschen (CRUD), die an einem Graphdatenmodell arbeiten. Graphdatenbanken werden in der Regel für die Verwendung mit OLTP-Systemen (Online Transaction Processing) erstellt. Sie werden normalerweise für die Transaktionsleistung optimiert und unter Berücksichtigung der Transaktionsintegrität und der betrieblichen Verfügbarkeit entwickelt.

Im Gegensatz zu anderen Datenbanken haben Beziehungen in Graphdatenbanken erste Priorität. Das bedeutet, dass Ihre Anwendung keine Datenverbindungen mit Fremdschlüsseln oder Out-of-Band-Verarbeitung wie MapReduce ableiten muss.

Indem wir die einfachen Abstraktionen von Knoten und Beziehungen in verbundene Strukturen zusammenfügen, ermöglichen uns Graphendatenbanken, anspruchsvolle Modelle zu erstellen, die sich eng an unsere Problemdomäne anlehnen.

Es gibt drei dominante Diagrammdatenmodelle:

- Property-Graph.
- Resource Description Framework (RDF) Triple.
- Hypergraph.

Die meisten der auf dem Markt gebräuchlichen Graphendatenbanken verwenden eine Variante des Property-Graph.

2 Abgrenzung gegenüber relationalen Datenbank

Relationale Datenbanken speichern hochgradig strukturierte Daten in Tabellen mit vorgegebenen Spalten bestimmter Typen und vielen Zeilen derselben Art von Informationen. Die Daten sind streng strukturiert, die in bestimmten Anwendungen verwendet werden.

In relationalen Datenbanken werden die Referenz auf andere Zeilen und Tabellen durch die Referenz auf ihre (Primär-) Schlüsselattribute über Fremdschlüsselspalten angegeben. Diese Verfahren hat jedoch eigene Einschränkungen. Joins werden zur Abfragezeit berechnet, indem Primär- und Fremdschlüssel der vielen potenziell indizierten Zeilen der zu verknüpfenden Tabellen abgeglichen werden. Diese Operationen sind rechen- und speicherintensiv und haben exponentielle Kosten.

Wenn Sie viele Beziehungen verknüpfen, müssen Sie eine JOIN-Tabelle einführen, die Fremdschlüssel der beiden beteiligten Tabellen enthält, was die Join-Betriebskosten weiter erhöht. Diese teuren Join-Operationen werden normalerweise durch Denormalisierung von Daten, um die Anzahl der notwendigen Joins zu reduzieren.

Diese folgende Merkmale von Graphdatenbank: Intuitivität, Geschwindigkeit, Agilität macht sie speziell gegen relationalen Datenbank.

3 Abgrenzung gegenüber RDF Triple Stores

Triple-Stores stammen aus der Semantic-Web, bei der Forscher an großflächigen Wissen interessiert sind, indem sie den Links, die Webressourcen verbinden, semantisches Markup hinzufügen. Mit RDF können wir Aussagen über Ressourcen treffen. Eine Anweisung hat immer folgende Struktur:

<subject> <predicate> <object>

Triple-Stores fallen unter die allgemeine Kategorie von Graph-Datenbanken, weil sie sich damit befassen Daten, die - einmal verarbeitet - tendenziell logisch verknüpft sind. Sie sind jedoch nicht "native" Graph-Datenbanken, da sie keine indexfreie Umgebung unterstützen, und ihre Speicher-Engines sind

auch nicht zum Speichern von Eigenschaftsgraphen optimiert. Triple Stores speichert die Tripel als unabhängige Artefakte, wodurch sie horizontal für die Speicherung skaliert werden können, sie jedoch daran hindern, Beziehungen schnell zu durchlaufen. Zur Durchführung von Graphabfragen müssen Triple Stores zusammenhängende Strukturen aus unabhängigen Fakten erstellen, wodurch jeder Abfrage eine Latenz hinzugefügt wird.

4 Typische Graph-Algorithmen

Graph-Algorithmen werden verwendet, um Metriken für Graphen, Knoten oder Beziehungen zu berechnen. Viele Graphalgorithmen sind iterative Ansätze, die Berechnung häufig durch die Graphen durchlaufen, wobei Random Walks,Pattern-Matching oder Breiten-, Tiefen-Suchen verwendet werden. Aufgrund des exponentiellen Wachstums von möglichen Wegen mit zunehmender Entfernung weisen viele der Ansätze auch eine hohe algorithmische Komplexität auf. Es existiert jedoch die optimierte Algorithmen, die bestimmte Strukturen des Graphen nutzen und die Operationen parallelisieren.

5 Graph-Datenbanksysteme: Neo4j

5.1 Einleitung

Neo4j ist ein Open-Source Graphdatenbank-Managementsystem, das von Neo Technology, Inc. entwickelt wurde. Es wurde entwickelt, um die schnelle Verwaltung, Speicherung und Durchquerung von Knoten und Beziehungen zu optimieren. Es ist eine hochskalierbare, native Graphdatenbank, die auf der Nutzung von Daten und ihrer Beziehung basiert.

Die Abfragesprache von relationalen Datenbanken ist SQL. Sie ist eine deklarative Abfragesprache, die sowohl eine einfache Ad-hoc-Abfrage in einem Datenbank-Tool als auch die Angabe von Use-Case-bezogenen Abfragen in Ihrem Code ermöglicht.

Cypher, die deklarative Graphabfragesprache von Neo4j, baut auf den grundlegenden Konzepten und Klauseln von SQL auf, bietet aber viele zusätzliche graphspezifische Funktionen, um die Arbeit mit Ihrem Rich-Graph-Modell zu vereinfachen, ohne zu ausführlich zu sein. Sie können die Graphenstrukturen mit übersichtlichen Anweisungen abfragen und aktualisieren. Cypher ist um die Diagrammmuster zentriert, die für Ihre Anwendungsfälle zentral sind, und stellt sie visuell als Teil der Abfragesyntax dar.

Wenn Sie jemals versucht haben, eine SQL-Anweisung mit einer großen Anzahl von Joins zu schreiben, wissen Sie, dass Sie aufgrund des technischen Rauschens schnell den Überblick verlieren, was die Abfrage tatsächlich macht.

In Cypher bleibt die Syntax sauber und konzentriert sich auf Domänenkonzepte, da die strukturellen Verbindungen zum Suchen oder Erstellen visuell ausgedrückt werden. Die anderen Klauseln neben dem Mustervergleich sollten für alle, die SQL schon früher benutzt haben, sehr vertraut sein.

5.2 Importieren von Daten aus einer relationalen Datenbank

Wenn Sie die Form Ihres Diagrammmodells ausreichend verstanden haben, d.h. welche Daten als Knoten oder Beziehungen dargestellt werden und wie die Bezeichnungen, Beziehungstypen und Attribute benannt werden, können Sie loslegen.

Die einfachste Möglichkeit, Daten aus Ihrer relationalen Datenbank zu importieren, besteht darin, einen CSV Datei einzelner Entitätstabellen und Join-Tabellen zu erstellen. Mehr Details können Sie hier finden:
`https://neo4j.com/developer/guide-import-csv/`

6 Graph-DB Angragesprache: Cypher für Neo4j

6.1 Einleitung

Cypher ist eine deklarative, SQL-inspirierte Sprache zum visuellen Beschreiben von Mustern in Graphen unter Verwendung einer ASCII-Syntax. Damit können wir angeben, was wir aus unseren Diagrammdaten auswählen, einfügen, aktualisieren oder löschen möchten, ohne dass wir genau beschreiben müssen, wie das geht.

Um die Leistungsfähigkeit unserer Graphendatenbank voll auszunutzen, möchten wir komplexere Muster zwischen unseren Knoten ausdrücken. Beziehungen sind im Grunde ein Pfeil $-$ $>$ zwischen zwei Knoten. Zusätzliche Informationen können in eckigen Klammern innerhalb des Pfeils platziert werden.

Knoten und Beziehungsausdrücke sind die Bausteine für komplexere Muster. Muster können fortlaufend oder durch Kommas getrennt geschrieben werden. Sie können sich auf zuvor deklarierte Variablen beziehen oder neue Variablen einführen.

6.2 Beispiel: The Movie Database

Die Filmdatenbank ist ein traditioneller Datensatz für Graphendatenbanken, der ähnlich wie die IMDB Filme und Schauspieler, Regisseure, Produzenten usw. enthält. Er könnte um Genre, Publisher, Ratings und mehr erweitert werden.

Beispiel für Abfragen:

Schauspieler, die in irgendeinem Film gespielt haben:

```
MATCH (m:Movie {title: 'Forrest Gump'})<-[:ACTS_IN]-(a:Actor)
RETURN a.name, a.birthplace
```

Finde Schauspieler, die in weniger als 3 Filmen waren:

```
MATCH (a:Actor)-[:ACTS_IN]->(m:Movie)
WITH a, count(m) AS movie_count
WHERE movie_count < 3
RETURN a, movie_count
ORDER BY movie_count DESC LIMIT 5;
```

7 Literatur

1. Graphdatenbanken URL: http://www.datenbanken-verstehen.de/lexikon/graphdatenbanken/
2. What is a Graph Database? URL: https://neo4j.com/developer/graph-database/
3. From Relational to Neo4j URL: https://neo4j.com/developer/graph-db-vs-rdbms/
4. W3C - RDF 1.1 Primer URL: https://www.w3.org/TR/2014/NOTE-rdf11-primer-20140624/
5. RDF Triple Stores vs. Labeled Property Graphs: What's the Difference? URL: https://neo4j.com/blog/rdf-triple-store-vs-labeled-property-graph-difference/
6. Graphalgorithmen URL: http://www.datenbanken-verstehen.de/lexikon/graphalgorithmen/
7. Intro to Cypher URL: https://neo4j.com/developer/cypher-query-language/
8. Dataset: Movie Database URL: https://neo4j.com/developer/movie-database/
9. Bryce Merkl Sasaki, Joy Chao Rachel Howard. Graph Databases For Beginners " O'Reilly Media, Inc.", 2013. URL: https://neo4j.com/whitepapers/graph-databases-beginners-ebook/?ref=home
10. Ian Robinson, JimWebber und Emil Eifrem. Graph databases. " O'Reilly Media, Inc.", 2013. URL: http://na-sjl.marketo.com/rs/neotechnology/images/Graph_Databases_2e_Neo4j.pdf