Modelling and Thinking in Graphs (Neo4J)

Faculty: Michael Enudi

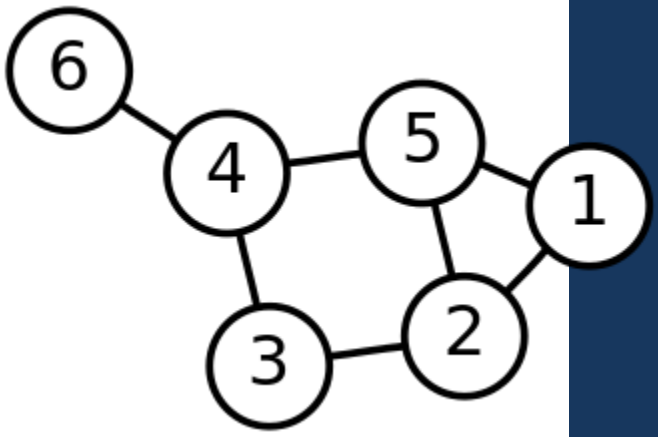
About Me.

- Lives and works in Johannesburg, South Africa
- Senior Software engineer with over 10 years of working experience writing enterprise java applications, architecting data solutions.
- Cloudera Certified Spark and Hadoop Dev.
- Oracle Certified SQL Expert
- Oracle Certified Java Master
- Sun Certified Java Business Component Dev.
- Sun Certified Java Programmer
- Big data enthusiast

LinkedIn → https://www.linkedin.com/in/michaelenudi

What are Graphs??

An abstract representation of a set of objects where some pairs are connected by links.



It is inspired by concept or subject of mathematics called graph theory.

The objects or entities are called Vertex/Vertices or Node/Nodes while the relationship are called Edge/Edges or Relationship/Relationships.

Types of Graphs

- Simple graphs
- Bipartite graphs
- Directed and Undirected graphs
- Multigraphs
- Property graphs
- See more ...

lines or curves for the edges. Graphs are one of the objects of study in discrete mathemati

The edges may be directed or undirected. For example, if the vertices represent people at
hands with a person *B* only if *B* also shakes hands with *A*. In contrast, if any edge from a p
The former type of graph is called an *undirected graph* and the edges are called *undirecte*

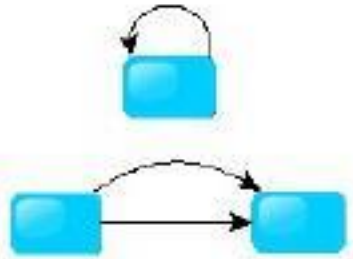
Graphs are the basic subject studied by graph theory. The word "graph" was first used in th

Contents [hide]

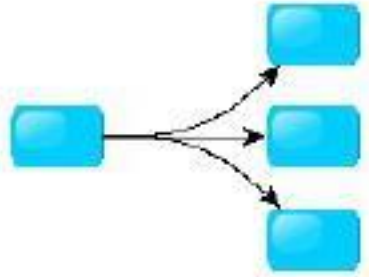
Different Kinds of Graphs

- Undirected Graph
- Directed Graph

- Pseudo Graph
- Multi Graph

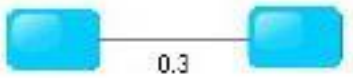


- Hyper Graph

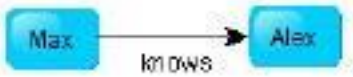


More Kinds of Graphs

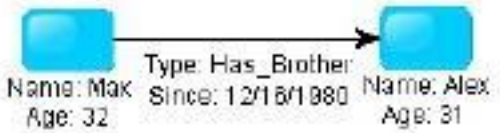
- Weighted Graph



- Labeled Graph



- Property Graph

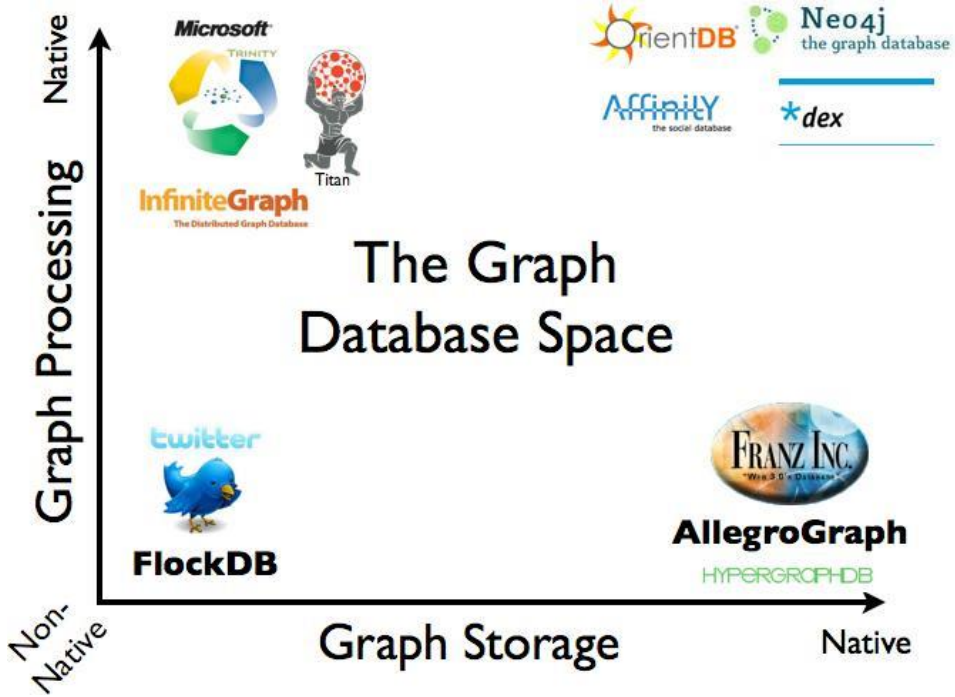


Graph storage or graph databases

A family of NoSQL databases that provide full transactional CRUD support for data modelled as graphs.

They

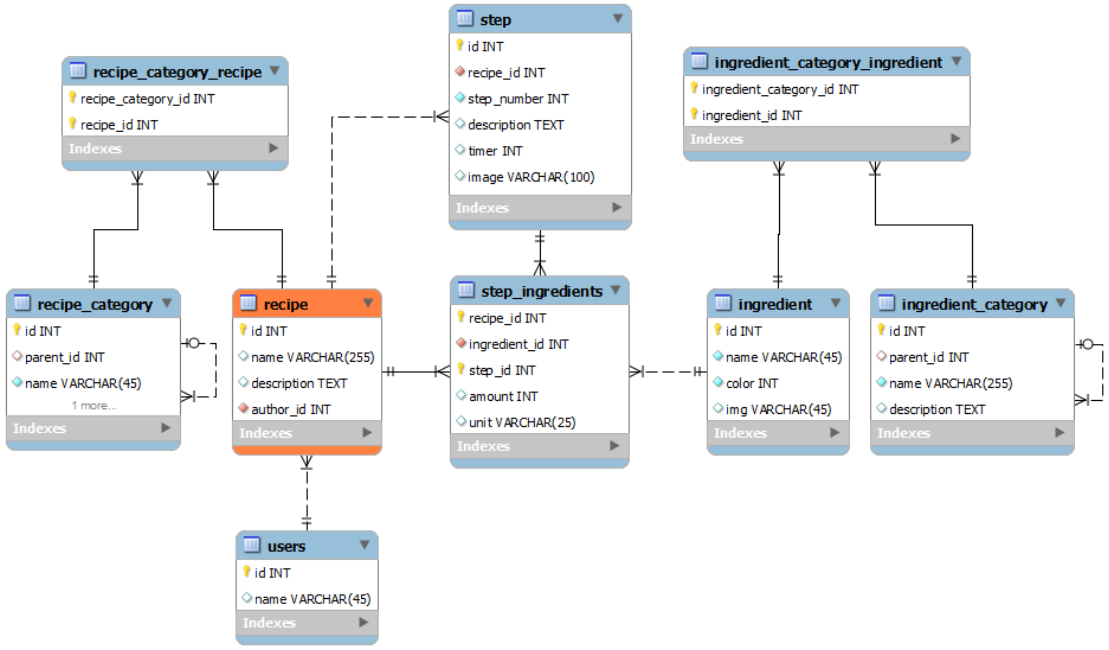
- Offer real time query support
- Are engineered for transactional integrity
- Can served as data sources for big data real time systems.



Why are RDBMS not enough

RDBMS are great and still most widely deployed today but but

- They lack relationship
- When relationship is enforced, scalability suffers as data grows.
- When you lose relationship, you also loose insight
- Gives us only CA portion of the CAP theorem. (we will discuss this in more details later)
- They are a step or two more abstract than the natural or business world



Other NoSQL databases ??

But we have

- Document-oriented databases (MongoDB, CouchDB)
- Key-value stores (Memcache, Redis)
- Columnar (BigTable-like) databases (HBase, Cassandra)

……but

1. Relationships are still an after-thought.
2. Lack of native graph processing feature (a.k.a, index-free adjacency).
3. Using various implementation of linking (eg. ObjRef in MongoDB) will perform badly.

Review: CAP Theorem

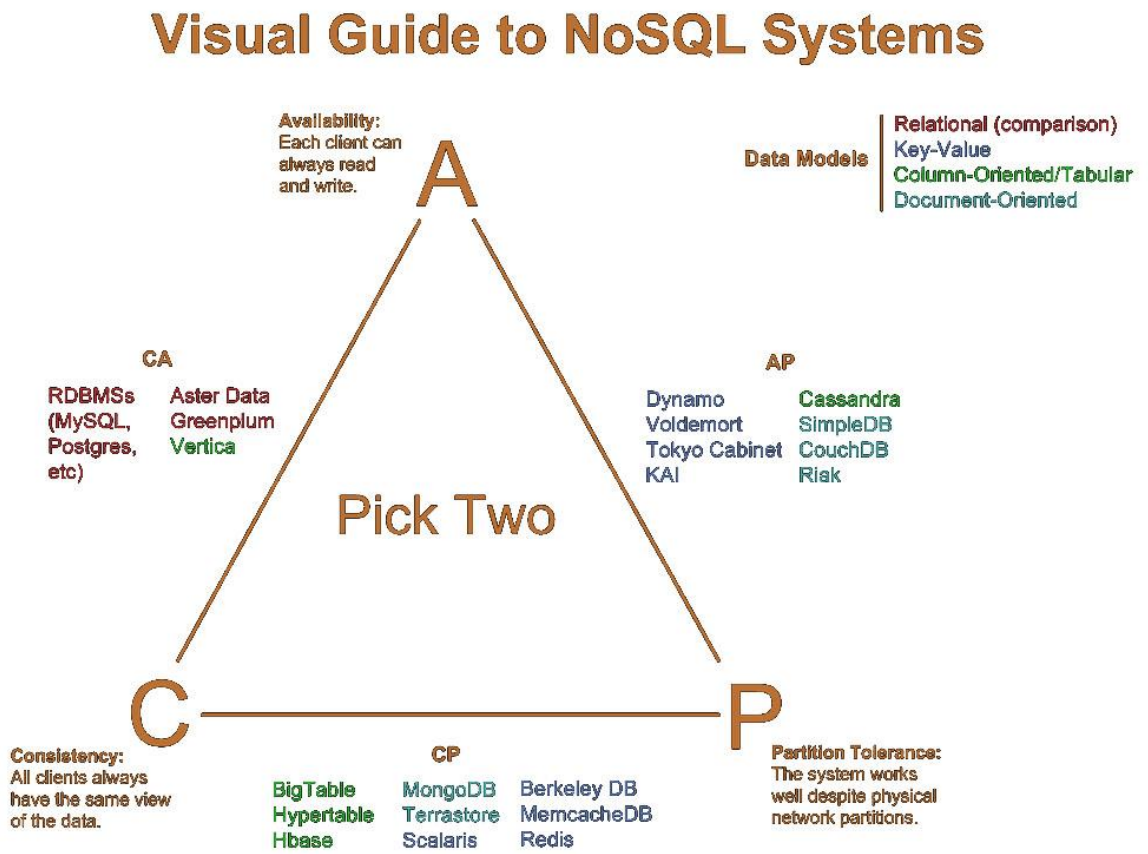
Three primary concerns you must balance when choosing a data management system: consistency, availability, and partition tolerance.

- ❖ Consistency means that each client always has the same view of the data.
- ❖ Availability means that all clients can always read and write.
- ❖ Partition tolerance means that the system works well across physical network partitions.

According to the CAP Theorem, you can only pick two.

A distributed system can satisfy any two of CAP guarantees at the same time but not all three:

- ✓ Consistency + Availability
- ✓ Consistency + Partition Tolerance
- ✓ Availability + Partition Tolerance



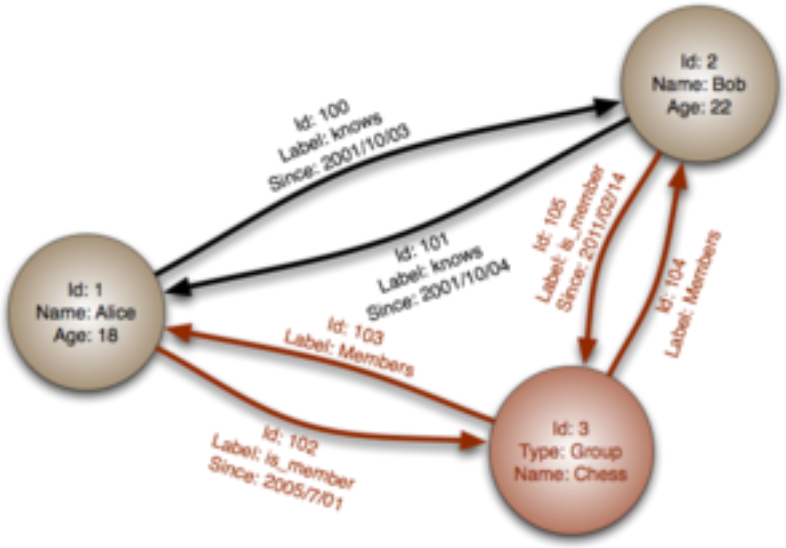
Review: Graph database and the CAP theorem

Consistency	???
Availability	???
Partitioning	???

Data modelling in a Graph database

1. A labelled property graph is made up of nodes, relationships, properties, and labels.
2. Nodes contains properties (key-value pairs)
3. Nodes can also contains more than one labels
4. Relationships is defined by two nodes they connect, a direction and a name
5. relationships can have properties

Pg.26, Graph Databases 2nd Edition by by Ian Robinson, Jim Webber, and Emil Eifrem



Data modelling in a Graph database

1. Use nodes to represent entities—that is, the things in our domain that are of interest to us, and which can be labelled and grouped.
2. Use relationships both to express the connections between entities and to establish semantic context for each entity, thereby structuring the domain.
3. Use relationship direction to further clarify relationship semantics. Many relationships are asymmetrical, which is why relationships in a property graph are always directed. For bidirectional relationships, we should make our queries ignore direction, rather than using two relationships.
4. Use node properties to represent entity attributes, plus any necessary entity metadata, such as timestamps, version numbers, etc.
5. Use relationship properties to express the strength, weight, or quality of a relationship, plus any necessary relationship metadata, such as timestamps, version numbers, etc.

Pg.67, Graph Databases 2nd Edition by by Ian Robinson, Jim Webber, and Emil Eifrem

Neo4J

- A Graph Database + Lucene Index
- Property Graph
- Full ACID (atomicity, consistency, isolation, durability)
- High Availability (with Enterprise Edition)
- 32 Billion Nodes, 32 Billion Relationships,64 Billion Properties (This limit has been removed - https://neo4j.com/blog/neo4j-3-0-massive-scale-developer-productivity/)
- Embedded Server
- REST API
- Open source project managed by Neo Technology
- Nodes and relationships are labelled property objects

Cypher: Neo4J's query language

Cypher is an expressive (yet compact) graph database query language.

It is declarative and based principally on pattern-matching. With PM, we can transverse graphs structures in any direction and at any depth.

```
(emil)<-[:KNOWS]-(jim)-[:KNOWS]->(ian)-[:KNOWS]->(emil)
```

Like SQL, cypher is made up of clauses, statements, functions and expressions.

Examples of clauses include MATCH, RETURN, WHERE, CREATE, CREATE UNIQUE, MERGE, DELETE, SET, FOREACH, UNION, WITH, etc.

```
MATCH (movie:Movie)
WHERE coalesce(movie.genres, "-") <> "-"
WITH SPLIT(movie.genres, "|") as parts, movie as m
UNWIND parts as x
MATCH (g: Genre {name: x})
MERGE (m)-[:IS_A]->(g)
REMOVE m.genres;
```

Cypher is currently only specific to Neo4J.

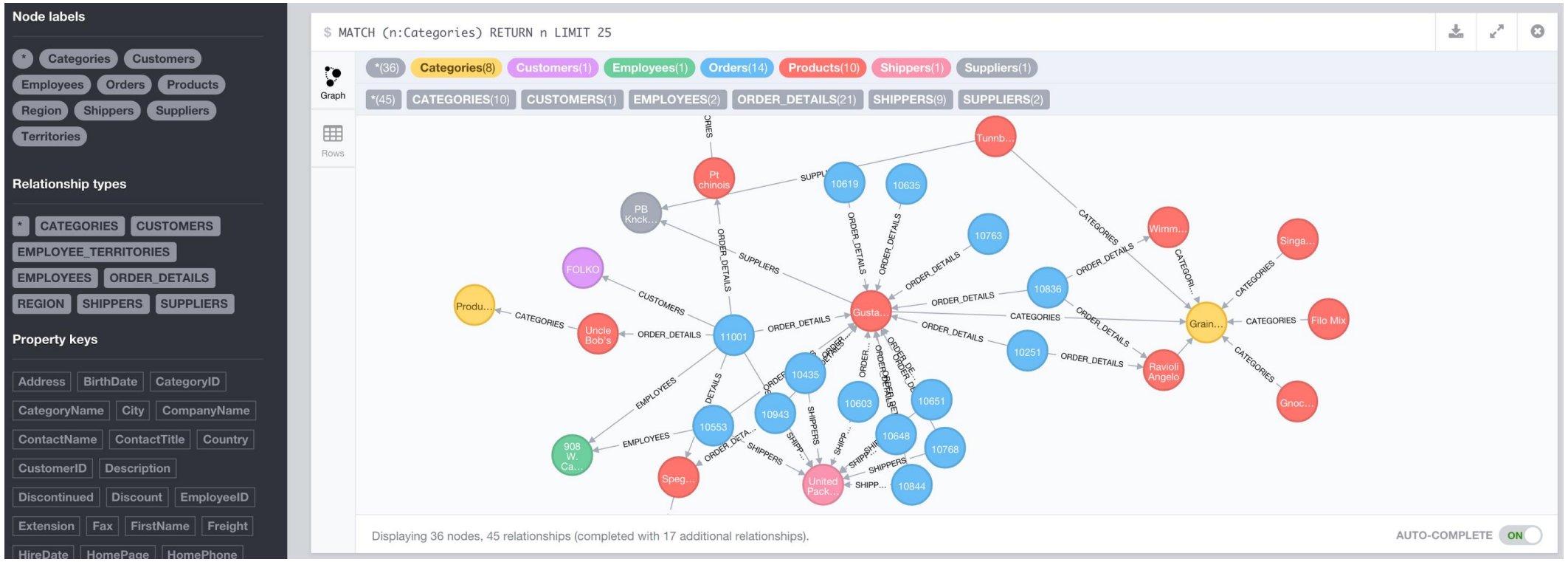
Neo4J and big data pipeline

- Neo4j-Spark-Connector
- Neo4j-Mazerunner
- Neo4j MongDB connector
- Neo4J Cassandra connector
- Neo4j Elastic search connector

- ✓ Sometimes, a some task are better processed outside the databases because of either the complexity or we don’t want to cause of denial of service on our database.
- ✓ Neo4J has connectors to load and store data to big data processing platforms like Apache Spark.
- ✓ It can be used as a store for final or intermediate result of graph or even non-graph processing frameworks.
- ✓ Also, Neo4J can be integrated to other NoSQL database to change the model for a totally different goal and perspective.

Lab

Exploring the Neo4J Web Console



Case study

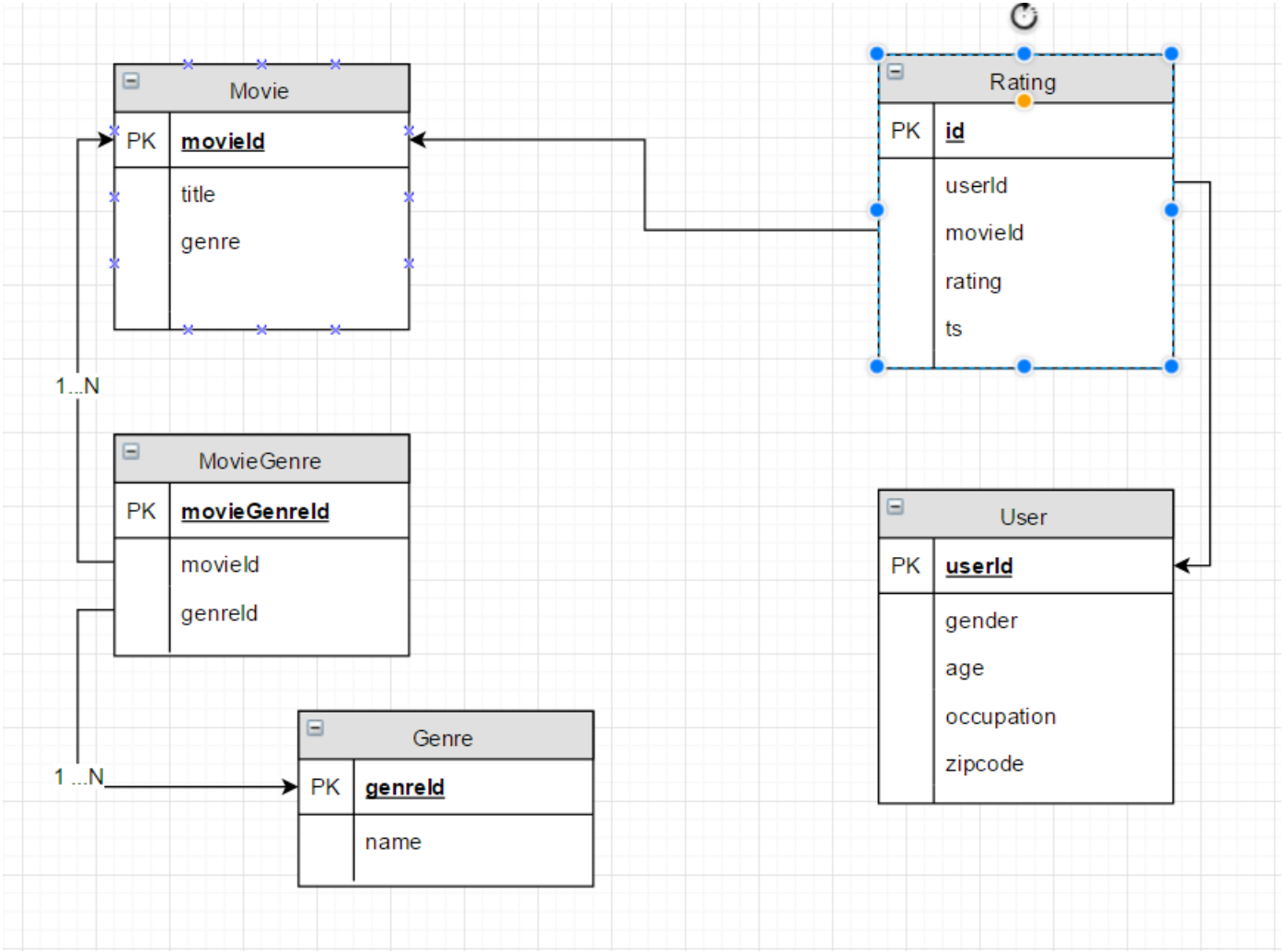
(Movielens)

Environment & tools

- Neo4J database server (Community edition)
- Any text editor
- (Optional) Java SDK installation
- (Optional) Maven
- (Optional) Java IDE

All of our tools for this lab are available for any OS platform.

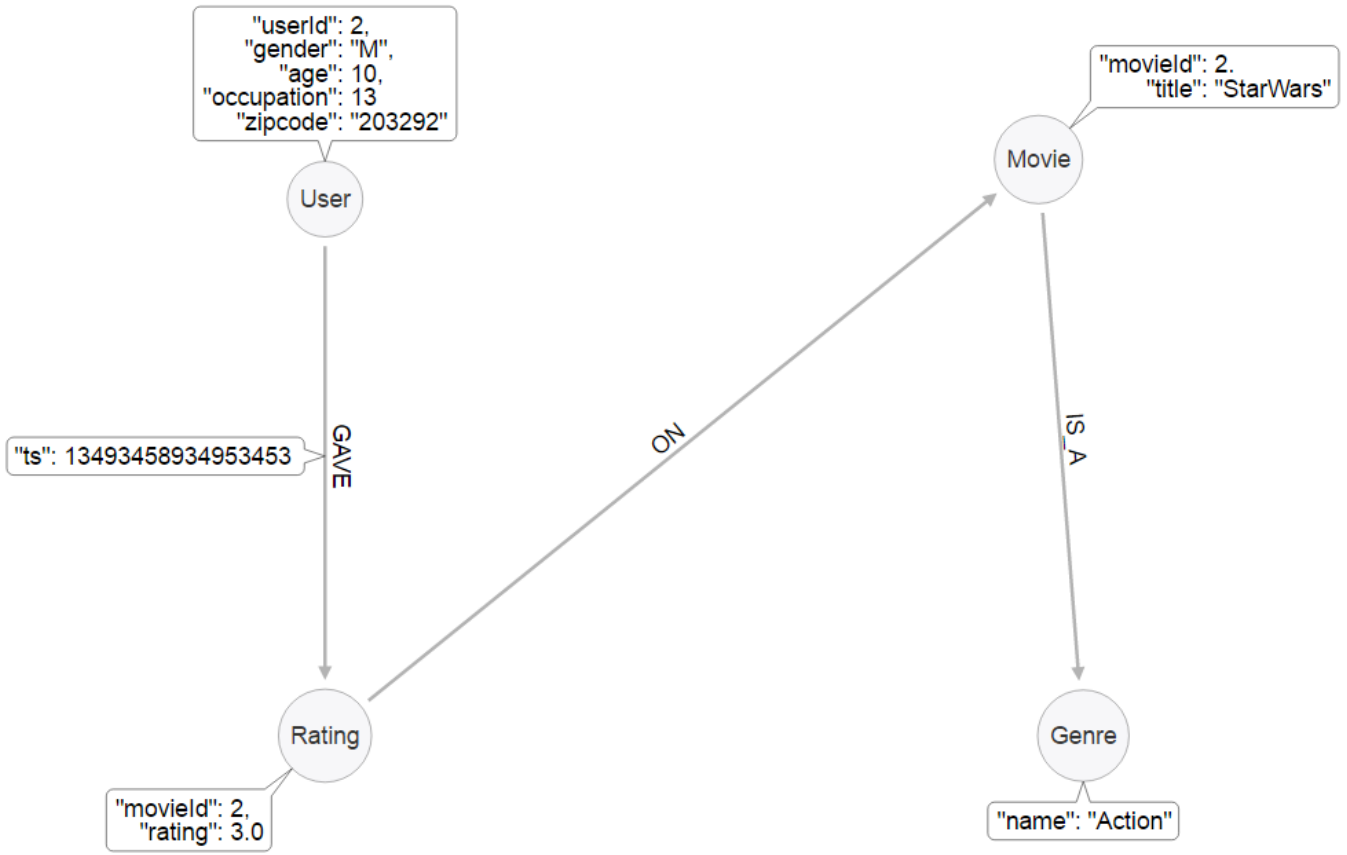
Movielens in RDBMS



- There are four entities in our business case but 5 in our relational model
- The MovieGenre has only one purpose in the model – a many to many relationship store.
- It is possible to still implement the design without the relationships. Some application manage the relationship in the code
- The relationship is clearly an after-thought

Modelling tool - https://www.draw.io/

Movielens in Graphs



- The Relationships are first class citizens of the model.
- There are 4 entities in our business case. Same as our design.
- The GAVE relationship has a timestamp to tell when the rating was done. It is a property or a metadata of the relationship.
- Any development on this data model must treat the relationship with the same attention as would be need for the nodes.

The model was built using the Arrow tool - http://www.apcjones.com/arrows/

Migrating data from CSV (delimiter separated value) files to Neo4J database

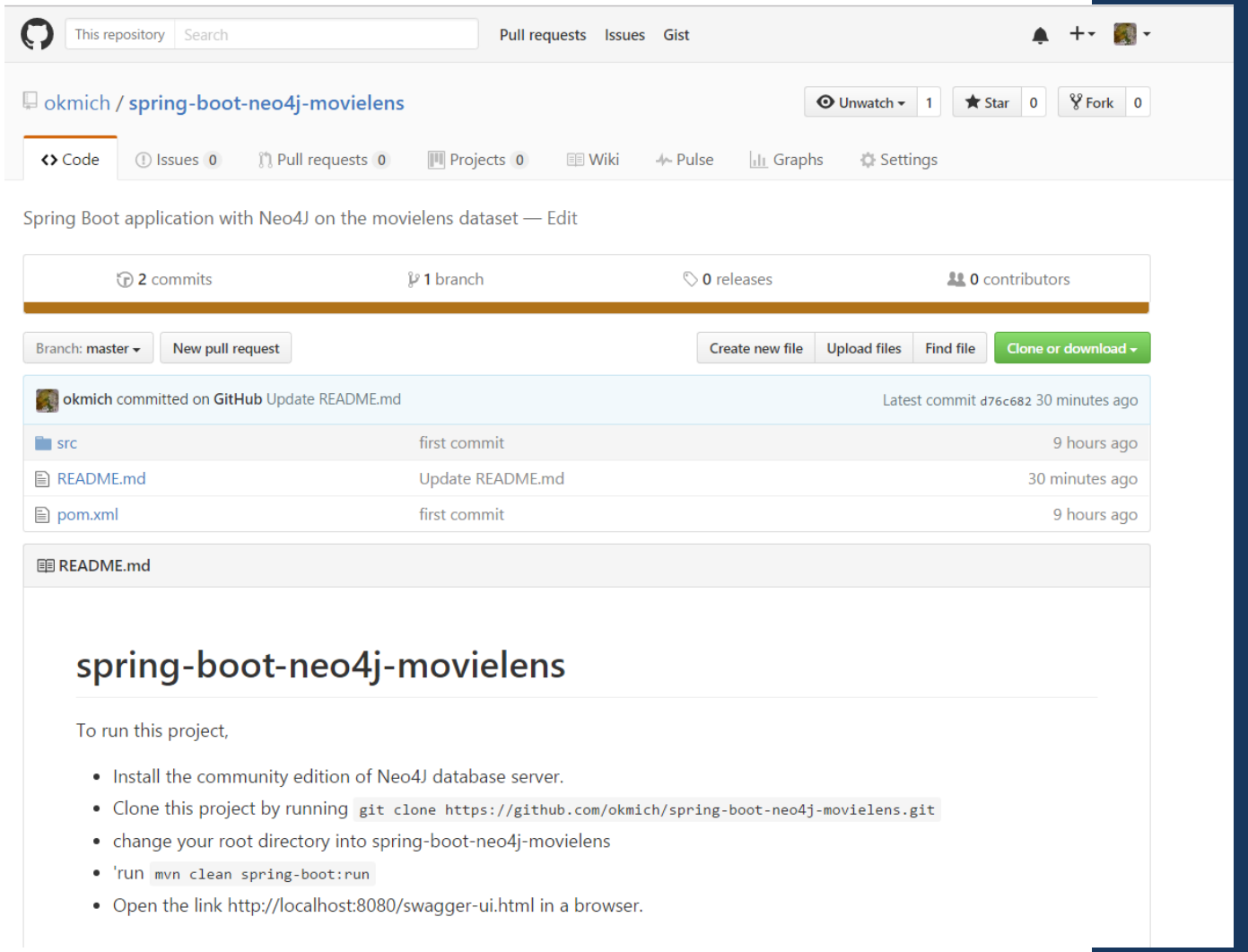
Writing queries in Cypher in Neo4J database

Movielens Application

Our movielens application is a backend service that receives request from a front-end client to perform transactional functions on our neo4j databases.
It functions includes

1. CRU (no D) for User and Movies domain objects
2. It returns a list of all Genre in our database.
3. It returns recommended movies for a user based on their demographic details.
4. It returns recommended movies for a movie based on its ratings.

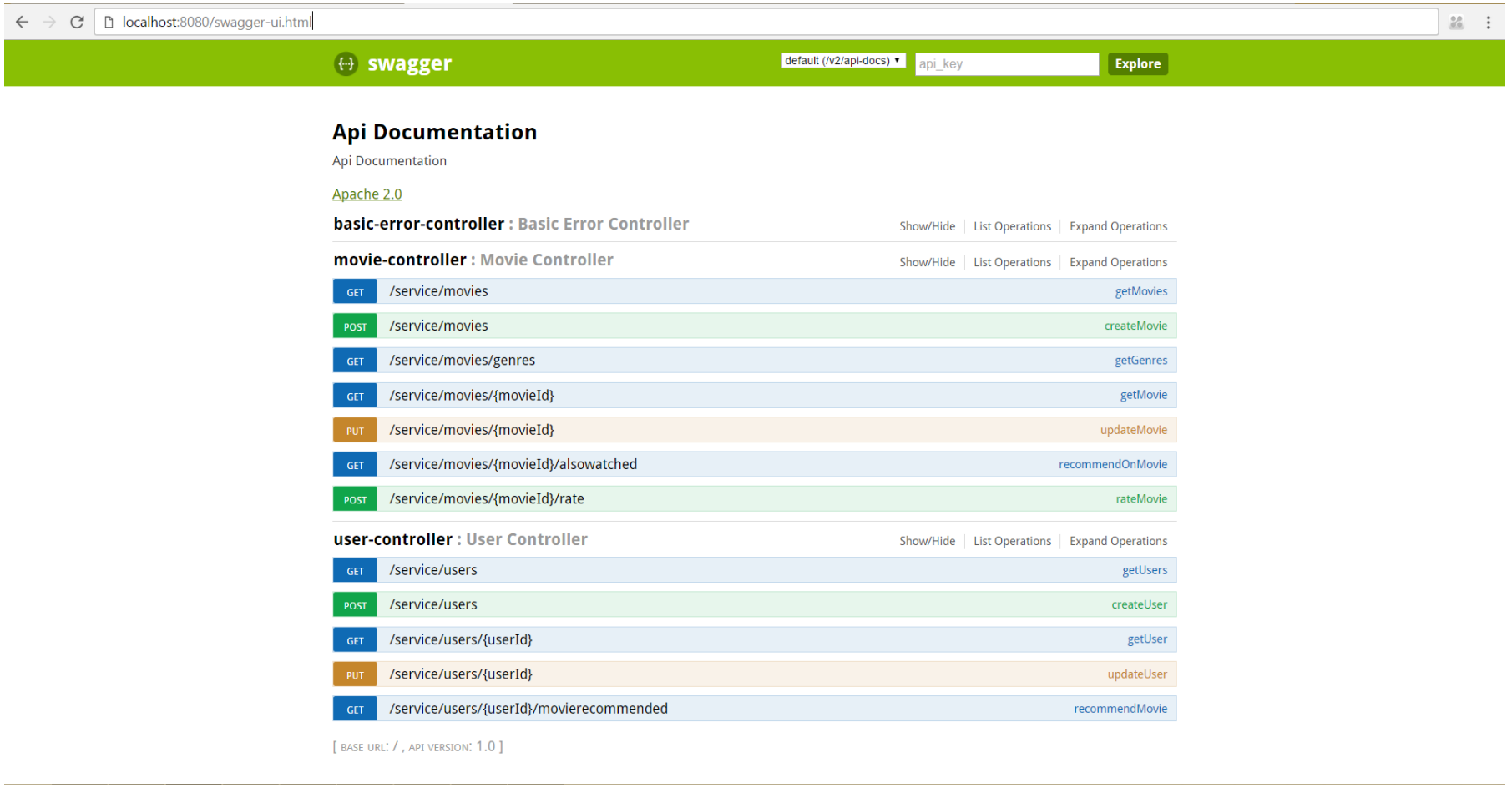
Our application is a back end system that exposes it functionality using RESTful web services.



The initial source code for this project is in the github directory - https://github.com/okmich/spring-boot-neo4j-movielens

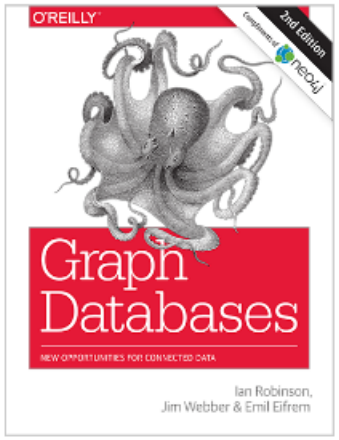
Movielens Application

The swagger documentation for our application's web services.



Lets test the functionalities

Read more

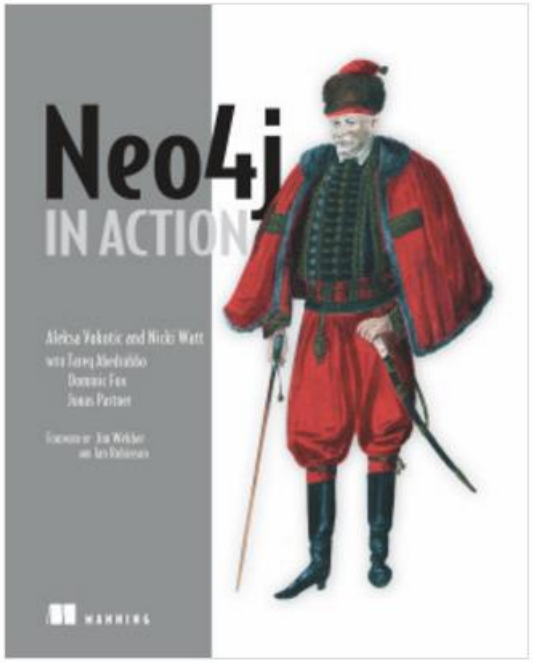


"This book significantly helps in understanding what graph databases are and how to use them properly... I really liked reading it!"

– *Krzysztof Ropiak, Customer*

Print Length: 224 Pages
Available Device Formats: PDF, Kindle, iBooks
Publisher: O'Reilly Media

https://neo4j.com/graph-databases-book/



https://www.manning.com/books/neo4j-in-action

Neo4j 3.0 Docs
https://neo4j.com/docs/

The Neo4j Developer Manual
https://neo4j.com/docs/developer-manual/current/

Thank You