Neo4j

Graph Databases

Agenda

- Overview
- Main Features
- Neo4j Architecture
- Neo4j Storage Strategy
- Neo4j based Application Architecture
- Benchmarks & Datasets

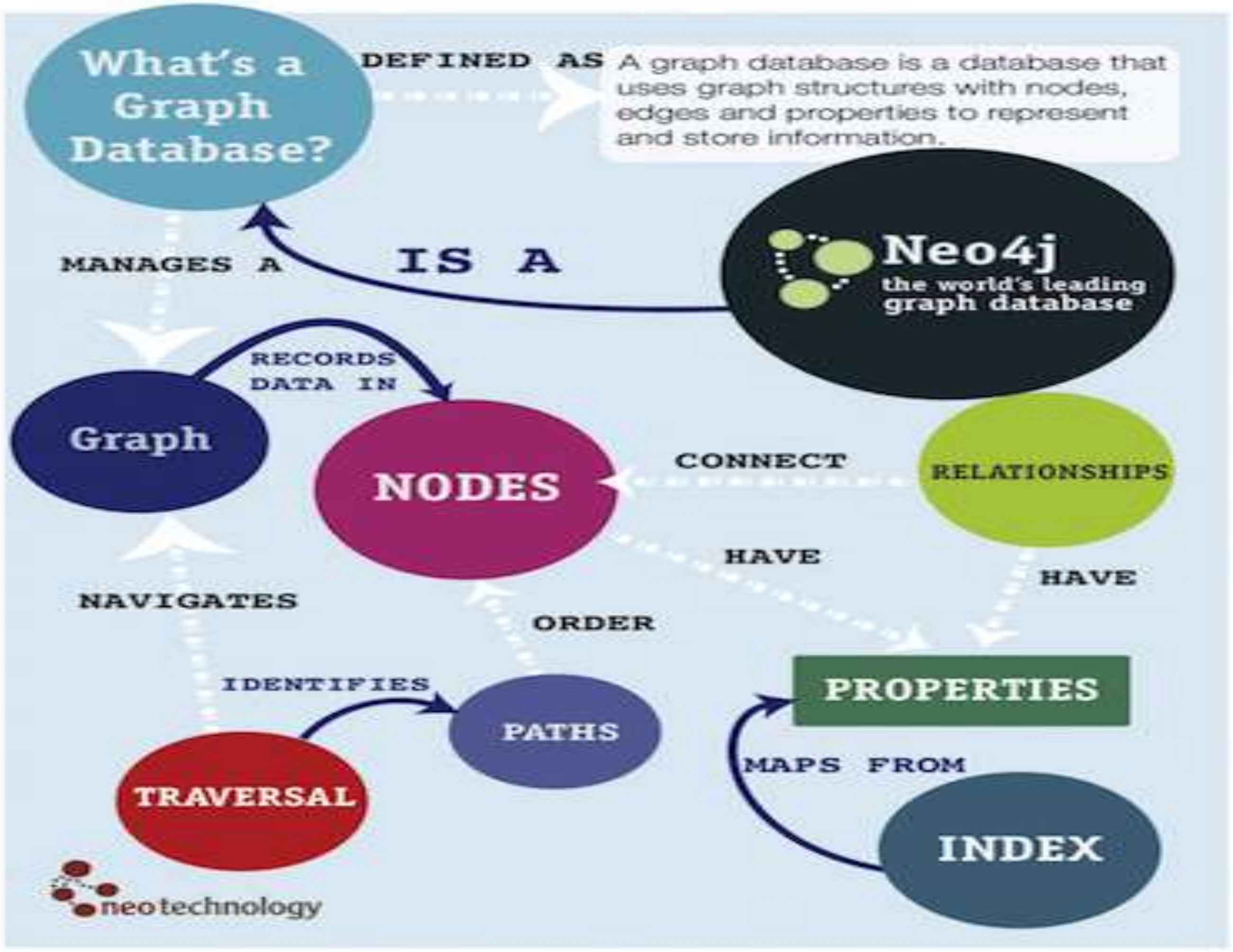
What is Graph Database?

- A graph database stores data in a graph, the most generic of data structures, capable of elegantly representing any kind of data in a highly accessible way.

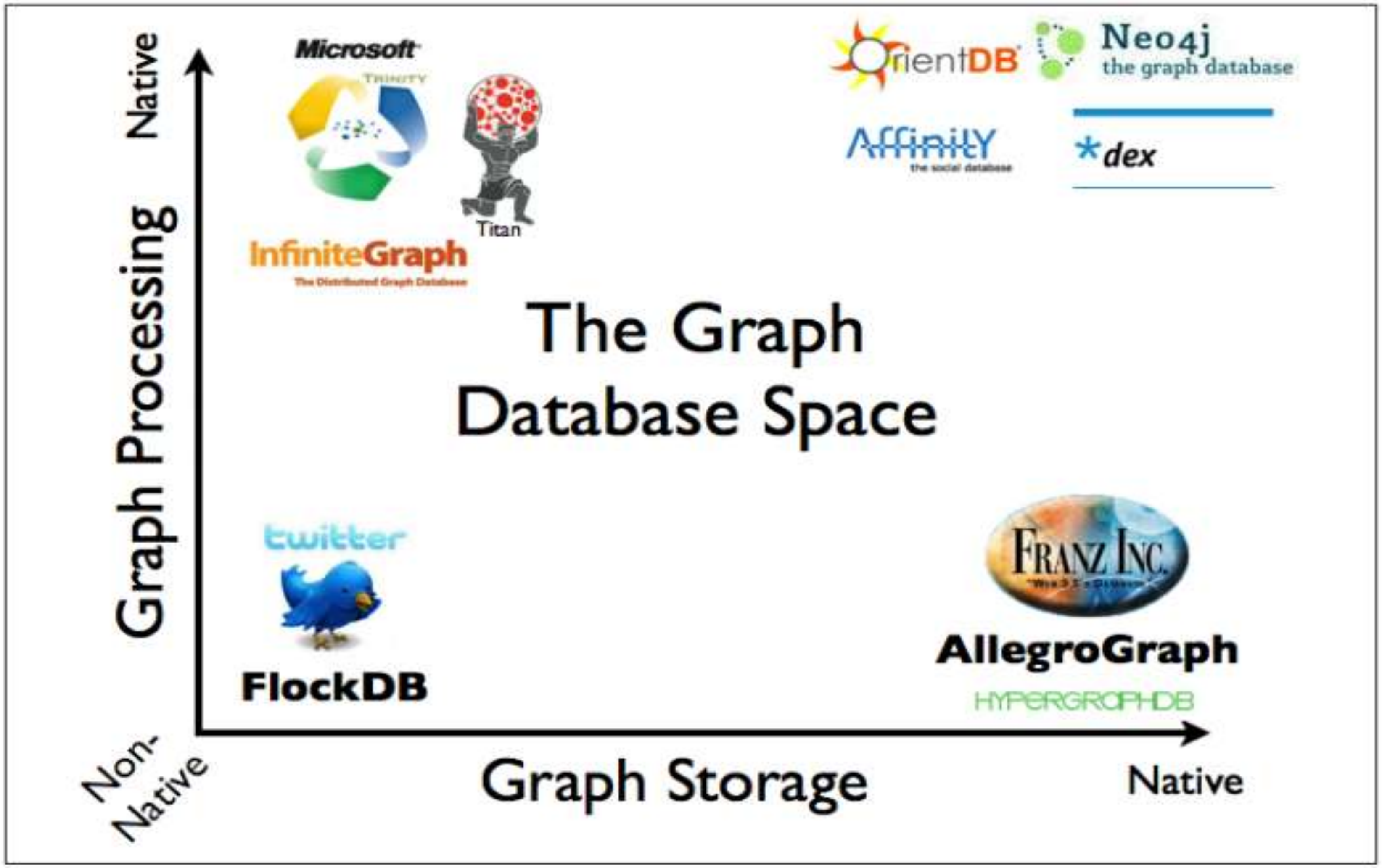
Main features: Neo4j

- ***intuitive***, using a graph model for data representation
- ***reliable***, with full ACID transactions
- ***durable and fast***, using a custom disk-based, native storage engine
- ***massively scalable***, up to several billion nodes/relationships/properties
- ***highly-available***, when distributed across multiple machines
- ***expressive***, with a powerful, human readable **graph query language**.
- ***fast,*** with a powerful traversal framework for high-speed graph queries
- ***embeddable***, with a few small jars
- ***simple***, accessible by a convenient **REST interface** or an object-oriented Java **API**

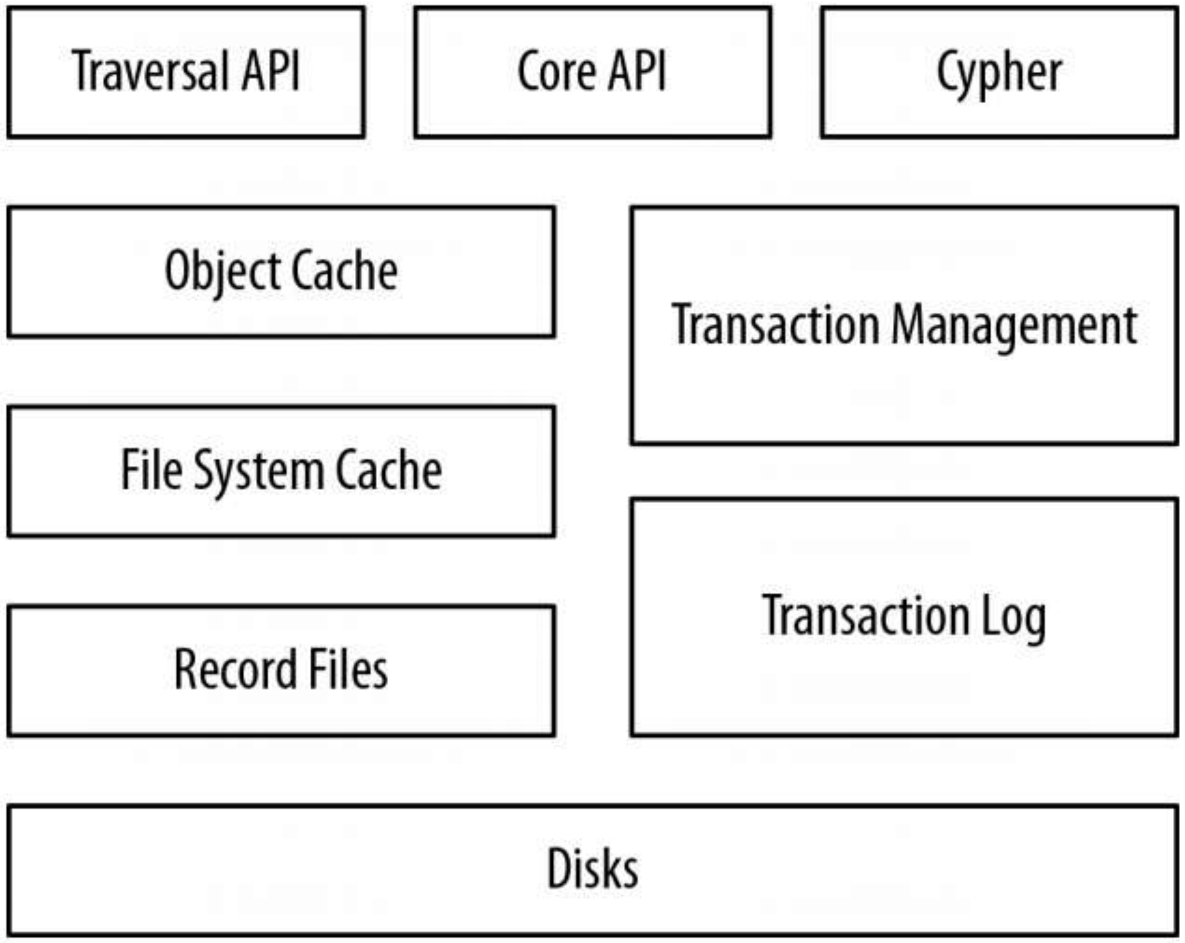
Neo4j in graph



An overview of graph database space:



Neo4j Architecture



Store files

- Neo4j stores graph data in a number of different store files.
- Each store file contains the data for a specific part of the graph (e.g., nodes, relationships, properties)
  - neostore.nodestore.db
  - neostore.relationshipstore.db
  - neostore.propertystore.db
  - neostore.propertystore.db.index
  - neostore.propertystore.db.strings
  - neostore.propertystore.db.arrays

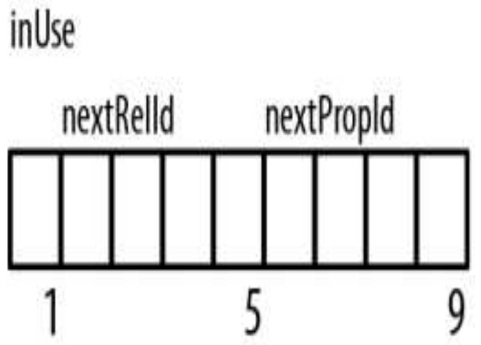
Node store

- neostore.nodestore.db
- Size: 9 bytes
  - 1st byte: in-use flag
  - Next 4 bytes: ID of first relationship
  - Last 4 bytes: ID of first property of node
- Fixed size records enable fast lookups

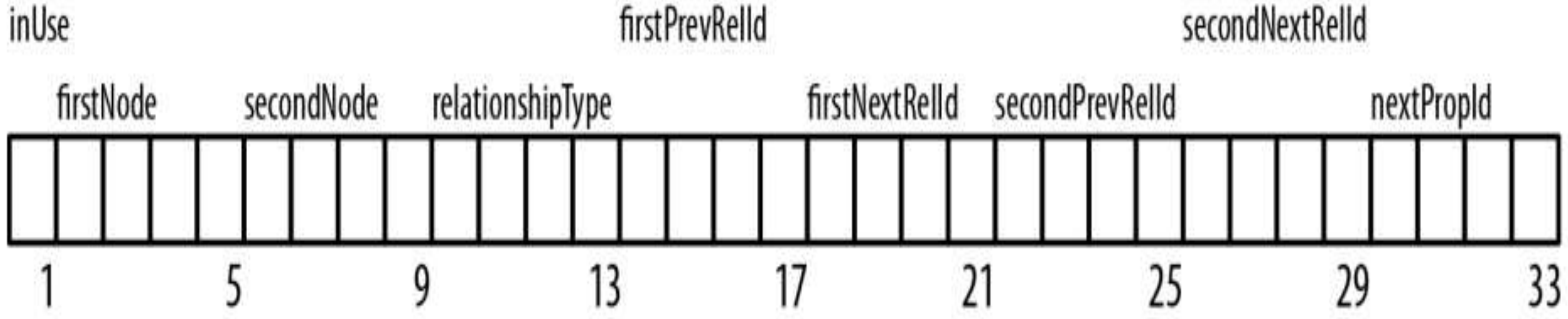
Relationship store

- neostore.relationshipstore.db
- Size: 33 bytes
  - 1st byte: In use flag
  - Next 8 bytes: IDs of the nodes at the start and end of the relationship
  - 4 bytes: Pointer to the relationship type
  - 16 bytes: pointers for the next and previous relationship records for each of the start and end nodes. (property chain)
  - 4 bytes: next property id

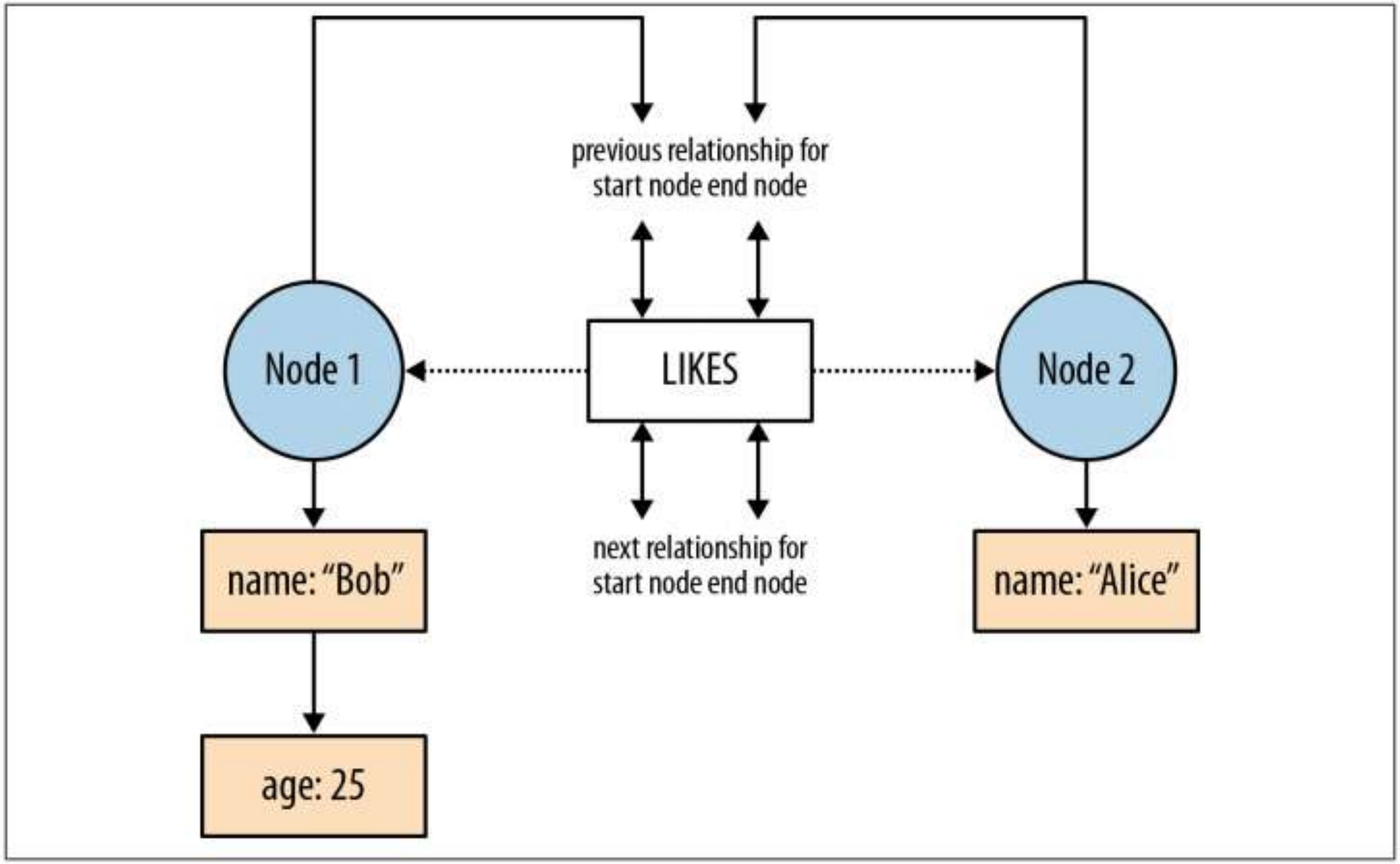
Node/property record structure



Relationship (33 bytes)



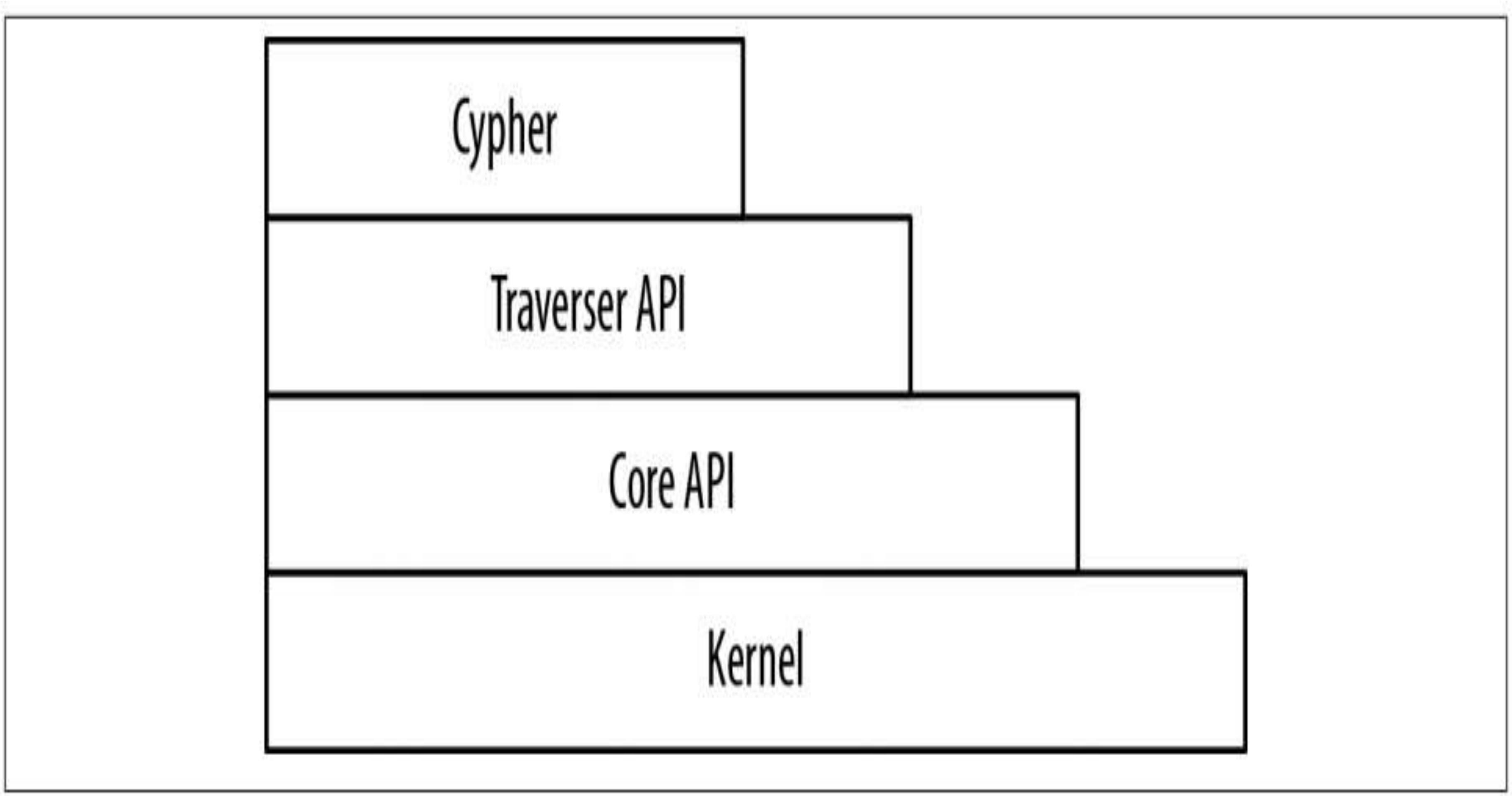
How a graph is physically stored



Neo4j: Data Size

nodes	2^{35} (~ 34 billion)
relationships	2^{35} (~ 34 billion)
properties	2^{36} to 2^{38} depending on property types (maximum ~ 274 billion, always at least ~ 68 billion)
relationship types	2^{15} (~ 32 000)

Logical view of neo4j user API

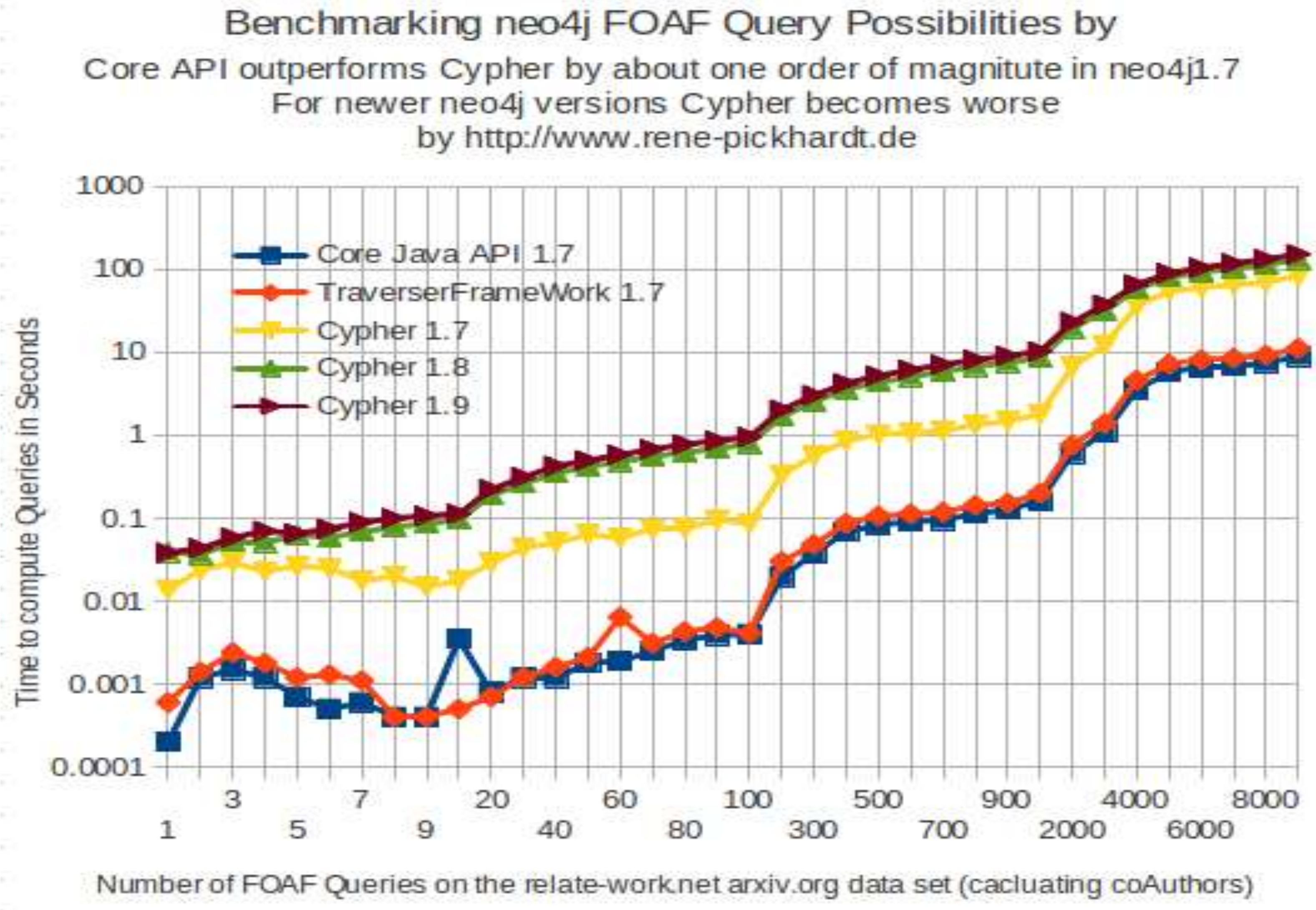


Application Architecture

- Embedded Neo4j
  - Low latency
  - Choice of APIs
  - Explicit transactions
  - JVM only
  - GC behavior
  - Database life cycle

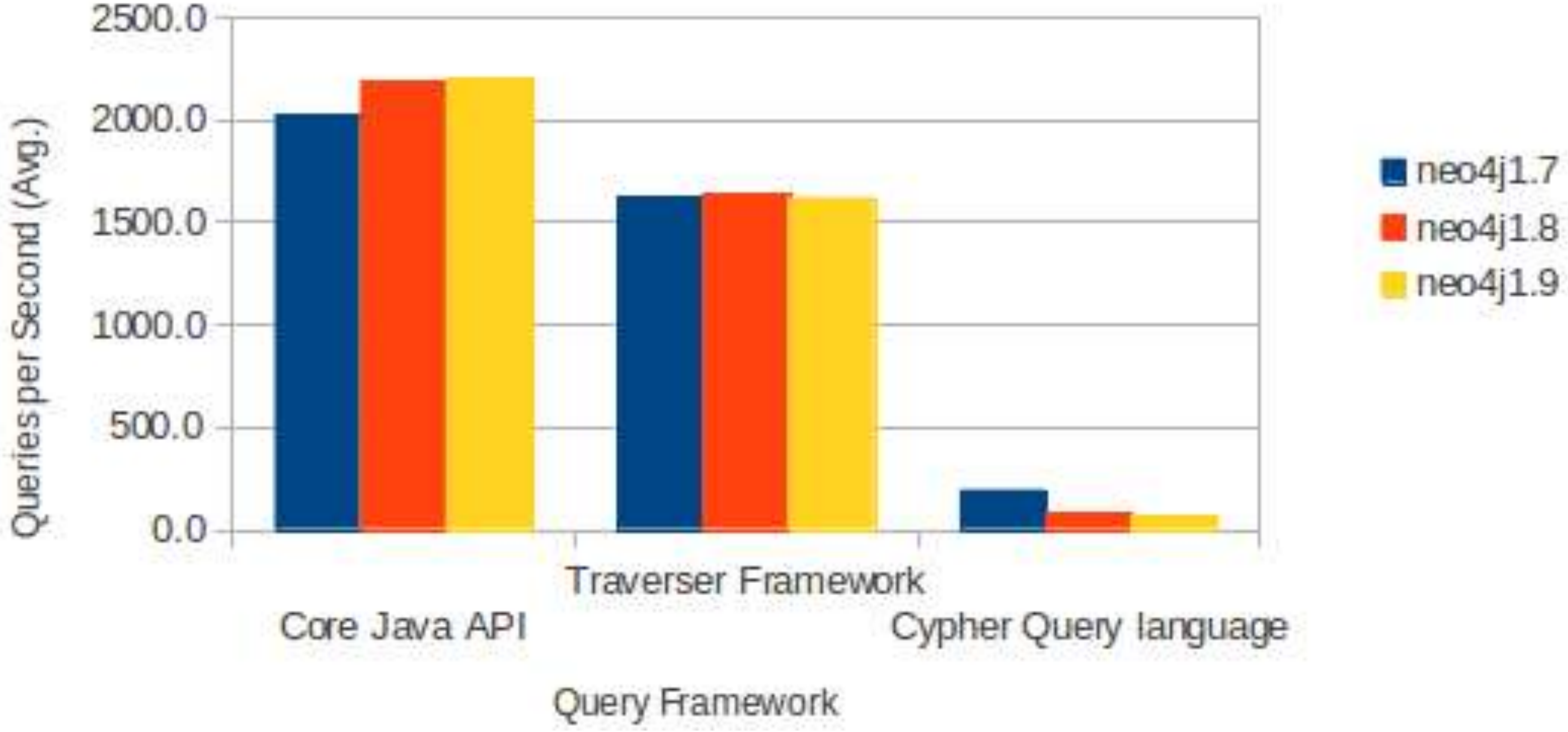
Application Architecture (contd..)

- Server Mode
  - Using same embedded instance of neo4j
  - REST API
  - Platform independent
  - Isolation from application GC behavior
  - Network overhead
  - Support for server extensions


Benchmarking neo4j FOAF Query Possibilities by
Core API outperforms Cypher by about one order of magnitute in neo4j1.7
For newer neo4j versions Cypher becomes worse
by http://www.rene-pickhardt.de
Core Java API 1.7
TraverserFrameWork 1.7
Cypher 1.7
Cypher 1.8
Cypher 1.9
Time to compute Queries in Seconds
1000
100
10
1
0.1
0.01
0.001
0.0001
1
3
5
7
9
20
40
60
80
100
300
500
700
900
2000
4000
6000
8000
Number of FOAF Queries on the relate-work.net arxiv.org data set (cacluating coAuthors)

FOAF like Queries per Second with various neo4j Query frame works

by http://www.rene-pickhardt.de
averages are built on experimants with more than 1k queries



Datasets

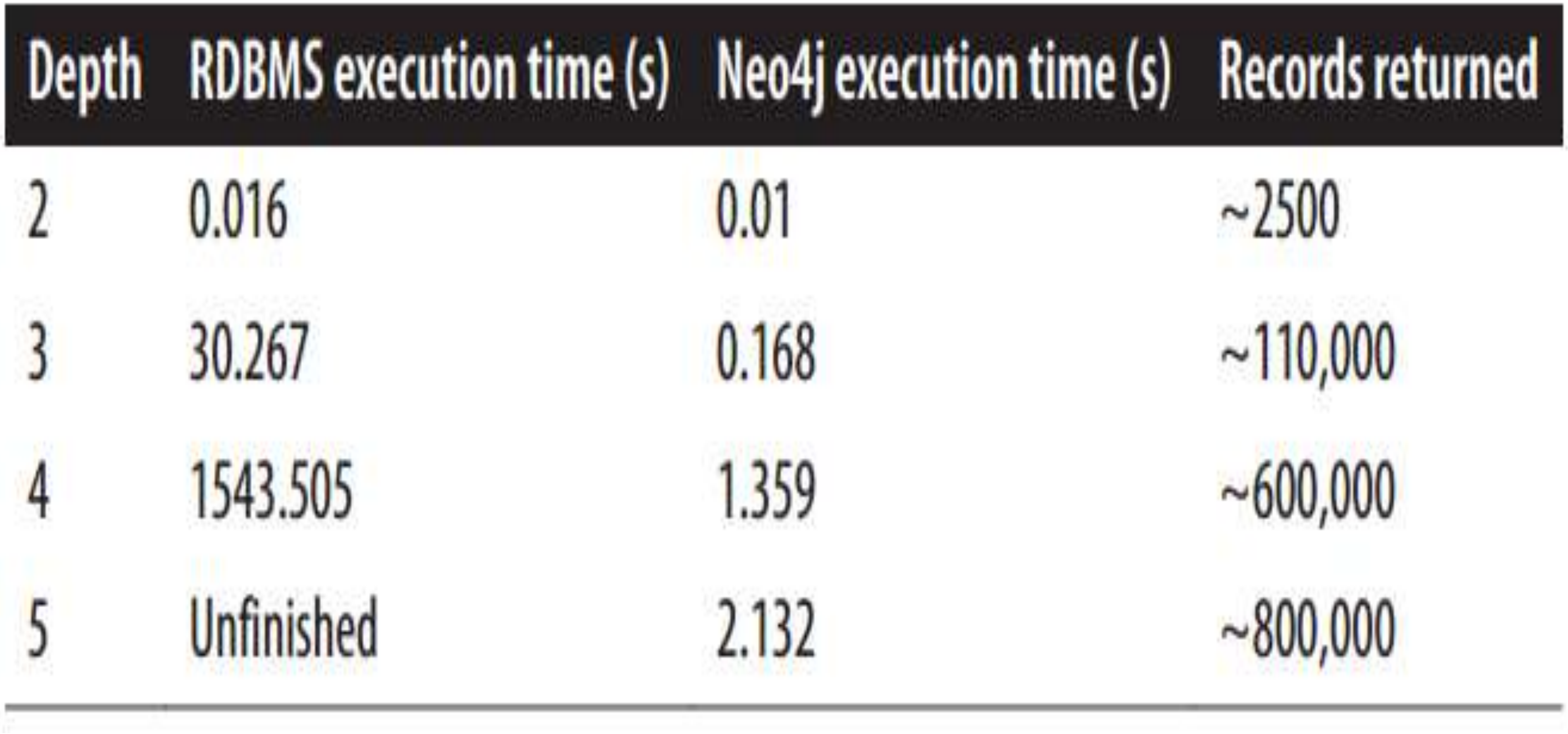
- Cineasts Movies & Actors
- Wordnet v:2.0
- Neo4j version 1.9 supports 10 of billions of nodes/relationships/properties
- The Neo4j team has publicly expressed the intention to support 100B+ nodes/relationships/properties in a single graph as part of its 2013 roadmap

benchmarks

- https://code.google.com/p/orient/wiki/GraphDBComparison
- http://nosql.mypopescu.com/post/1451025794/neo4j-and-orientdb-performance-compared
- https://groups.google.com/forum/#!msg/orient-database/9J5s4q3WKxY/d5XrIpiVG6gJ
- https://baach.de/Members/jhb/neo4j-performance-compared-to-mysql
- http://www.slideshare.net/thobe/nosqleu-graph-databases-and-neo4j
- http://java.dzone.com/articles/get-full-neo4j-power-using

Performance comparison: neo4j vs RDBMS (table 2-1 pg:20)

Supports index-free adjacency



Depth	RDBMS execution time (s)	Neo4j execution time (s)	Records returned
2	0.016	0.01	~2500
3	30.267	0.168	~110,000
4	1543.505	1.359	~600,000
5	Unfinished	2.132	~800,000