



Dogfooding the graph ecosystem

Learnings from building Neo4j Ops Manager

About Neo4j

- Creators of *Neo4j Graph Plattform* and *Neo4j* - World's leading native graph database
- Company founded 2007 in Sweden
- Today 700+ employees in San Mateo, London, Malmö and remote





October 26, 2023

- 3 tracks:
 - Building Applications & APIs
 - ML & AI with Graphs
 - Powerful Visualizations
- 24 hours
- 100 talks

[Register for free! - neo4j.com/nodes](https://neo4j.com/nodes)

NOM and dog food

Confused cookie monster approves

Eating your own dog panda food

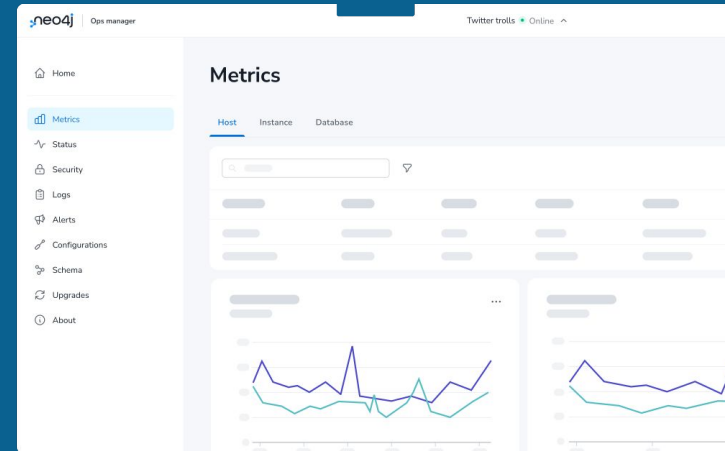


© Lea Peuckert

What are Neo4j Ops Managers objectives?

- To provide an improved experience for Neo4j Administrators through a set of UI based functional management tools, making Neo4j simple to run
- To reduce operational costs and improve productivity of the Administration team and reliability of the Neo4j implementation
- To manage all customer's Neo4j deployments regardless of modality

**For paid Enterprise Edition only*



NOM Architecture (and this talk)

1. What is Neo4j & Cypher? How to use

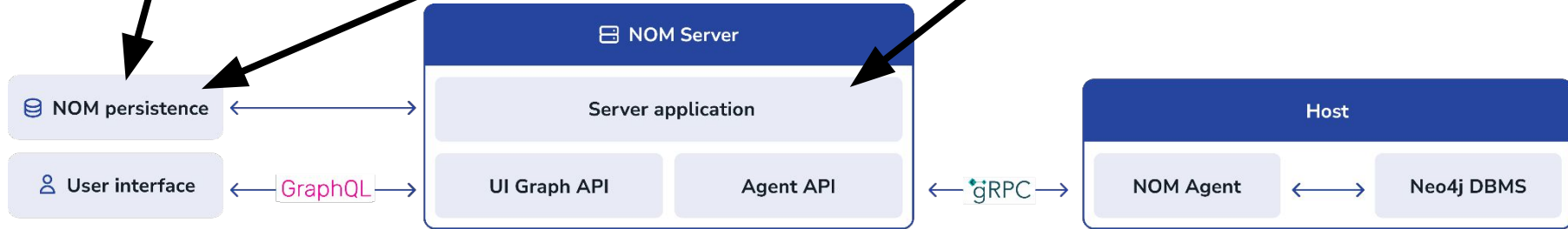
2. Using Neo4j to store metrics

3. Neo4j-Migrations

5. Neo4j Cypher-DSL

4. Spring Data Neo4j

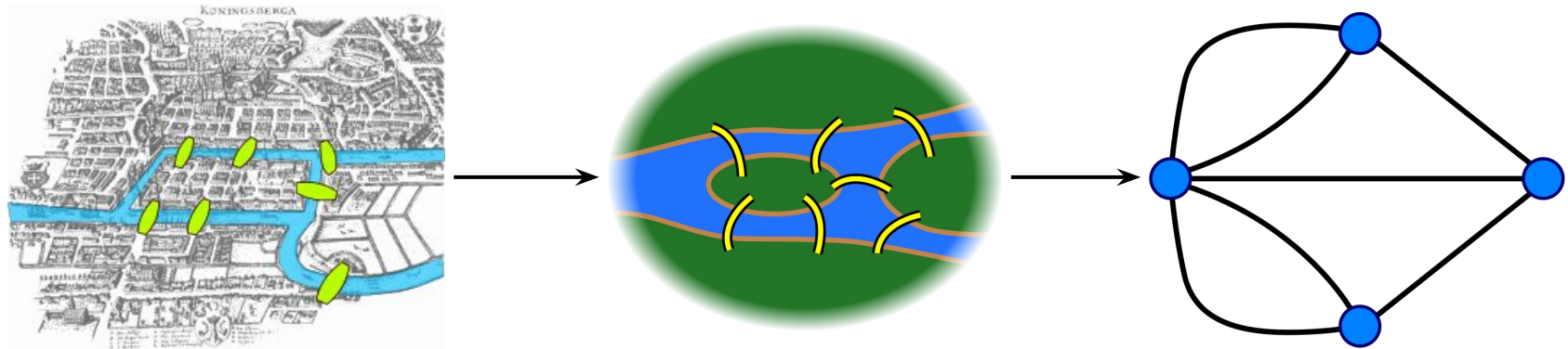
6. Special surprise!



Graphs, Neo4j & Cypher

A graph is...

...a set of discrete entities, each of which has some set of relationships with the other entities

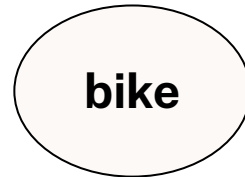
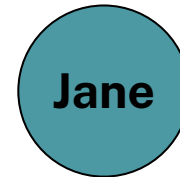


Seven Bridges of Königsberg problem. Leonhard Euler, 1735

Graph components

Node (Vertex)

- The main data element from which graphs are constructed



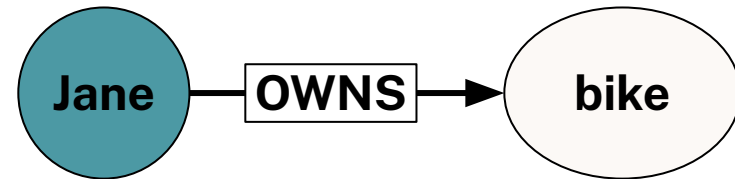
Graph components

Node (Vertex)

- The main data element from which graphs are constructed

Relationship (Edge)

- A link between two nodes. Has:
 - Direction
 - Type
- *A node without relationships is permitted. A relationship without nodes is not*



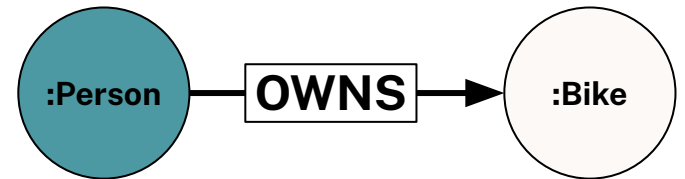
Property graph database

Node (Vertex)

Relationship (Edge)

Label

- Define node role (optional)



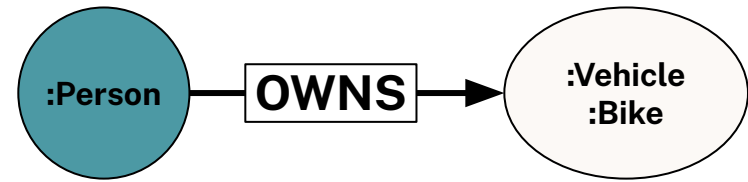
Property graph database

Node (Vertex)

Relationship (Edge)

Label

- Define node role (optional)
- Can have more than one



Property graph database

Node (Vertex)

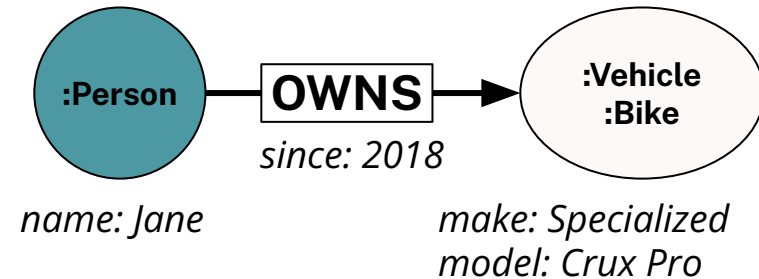
Relationship (Edge)

Label

- Define node role (optional)
- Can have more than one

Properties

- Enrich a node or relationship
- No need for nulls!



Cypher

A pattern matching query language made for graphs

- Declarative
- Expressive
- Pattern-Matching

Cypher

A pattern matching query language made for graphs

- Declarative
- Expressive
- Pattern-Matching

With ASCII ART 

Use MATCH to retrieve nodes

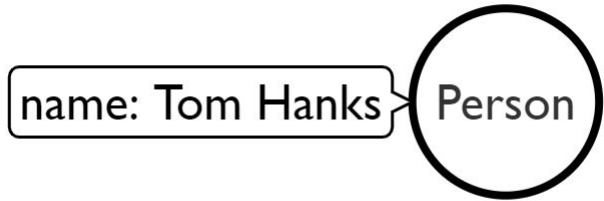
```
//Match all nodes  
MATCH (n)  
RETURN n;
```



Use MATCH to retrieve nodes

```
//Match all nodes  
MATCH (n)  
RETURN n;
```

```
//Match all nodes with a Person label  
MATCH (n:Person)  
RETURN n;
```



Use MATCH to retrieve nodes

```
//Match all nodes with a Person label  
MATCH (n:Person)  
RETURN n;
```



```
//Match all nodes with a Person label and property name is "Tom Hanks"  
MATCH (n:Person {name: "Tom Hanks"})  
RETURN n;
```

Use MATCH and properties to retrieve nodes

```
//Return nodes with label Person and name property is "Tom Hanks" - Inline  
MATCH (p:Person {name: "Tom Hanks"}) //Only works with exact matches  
RETURN p;
```

Use MATCH and properties to retrieve nodes

```
//Return nodes with label Person and name property is "Tom Hanks" - Inline  
MATCH (p:Person {name: "Tom Hanks"}) //Only works with exact matches  
RETURN p;
```

```
//Return nodes with label Person and name property equals "Tom Hanks"  
MATCH (p:Person)  
WHERE p.name = "Tom Hanks"  
RETURN p;
```

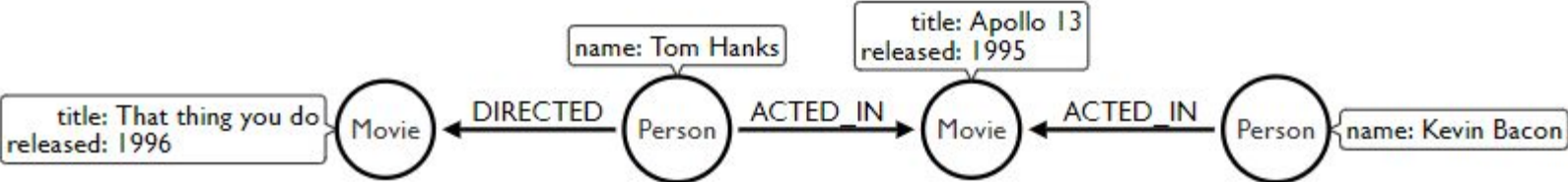
Use MATCH and properties to retrieve nodes

```
//Return nodes with label Person and name property is "Tom Hanks" - Inline  
MATCH (p:Person {name: "Tom Hanks"}) //Only works with exact matches  
RETURN p;
```

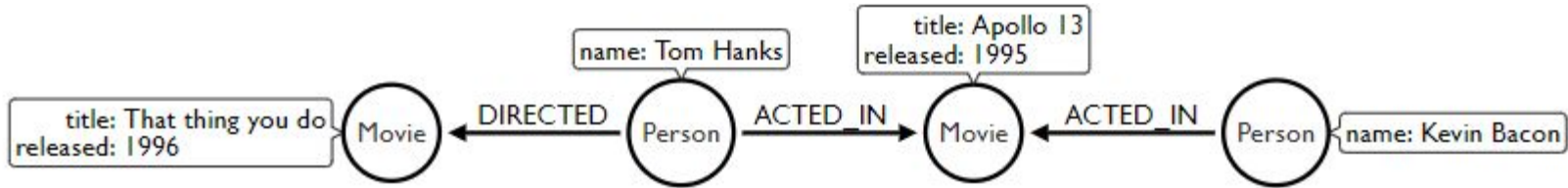
```
//Return nodes with label Person and name property equals "Tom Hanks"  
MATCH (p:Person)  
WHERE p.name = "Tom Hanks"  
RETURN p;
```

```
//Return nodes with label Movie, released property is between 1991 and 1999  
MATCH (m:Movie)  
WHERE m.released > 1990 AND m.released < 2000  
RETURN m;
```

Extending the MATCH

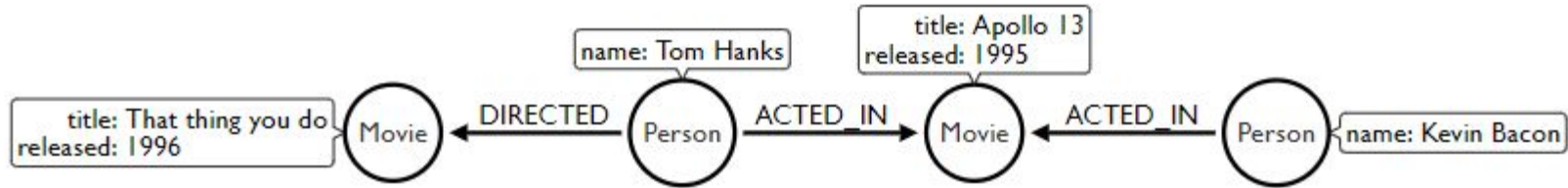


Extending the MATCH



```
//Find all the movies Tom Hanks is connected to  
MATCH (:Person {name:"Tom Hanks"})--(m:Movie)  
RETURN m.title;
```

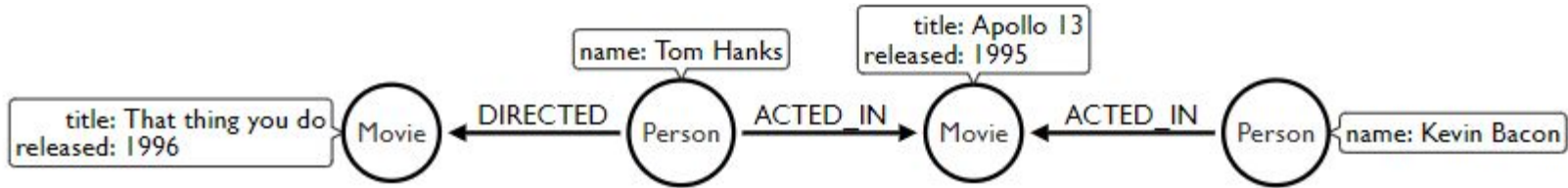

Extending the MATCH



```
//Find all the movies Tom Hanks is connected to  
MATCH (:Person {name:"Tom Hanks"})--(m:Movie)  
RETURN m.title;
```

```
//Find all the movies Tom Hanks directed and order by latest movie  
MATCH (:Person {name:"Tom Hanks"})-[:DIRECTED]->(m:Movie)  
RETURN m.title, m.released ORDER BY m.released DESC;
```

Extending the MATCH



```
//Find all the movies Tom Hanks is connected to  
MATCH (:Person {name:"Tom Hanks"})--(m:Movie)  
RETURN m.title;
```

```
//Find all the movies Tom Hanks directed and order by latest movie  
MATCH (:Person {name:"Tom Hanks"})-[:DIRECTED]->(m:Movie)  
RETURN m.title, m.released ORDER BY m.released DESC;
```

```
//Find all of the co-actors Tom Hanks have worked with  
MATCH (:Person {name:"Tom Hanks"})-->(:Movie)<-[:ACTED_IN]-(coActor:Person)  
RETURN coActor.name;
```

Developing with Neo4j

Neo4j drivers

- Binary protocol called Bolt
- Officially supported Neo4j drivers and community drivers

✔ Java	✔ Spring	✔ Neo4j-OGM	✔ .NET	✔ JavaScript
✔ Python	✔ Go	Ruby	PHP	Erlang/Elixir
Perl	C/C++	Clojure	Haskell	R

Neo4j Desktop

Experience Neo4j on Your Local Desktop. For Free.

*Includes Neo4j Enterprise License for Developers

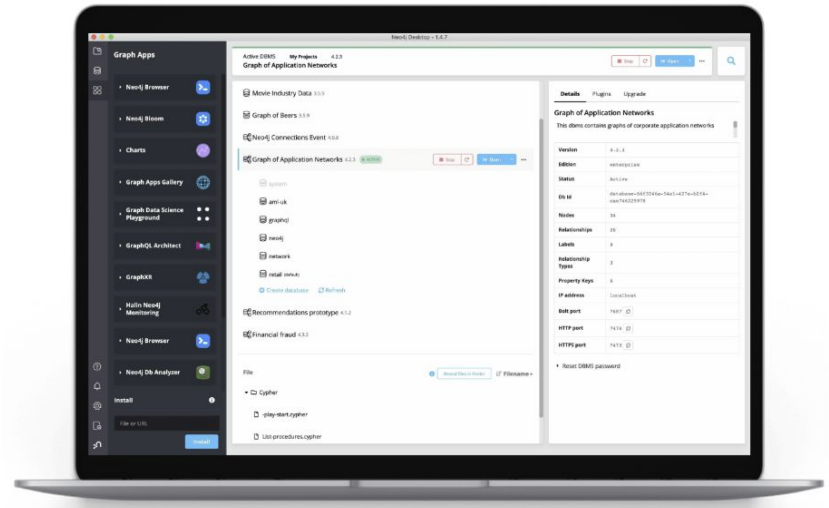
Download Now

System Requirements

MacOS 10.10 (Yosemite)+, Windows 7+, Ubuntu 12.04+,

Fedora 21, Debian 8

- Develop with Neo4j Enterprise features for advanced capabilities
- Unlimited local databases in a project-based workspace
- 1-click access to Graph Apps and plugins
- Connect and query even remote Neo4j databases



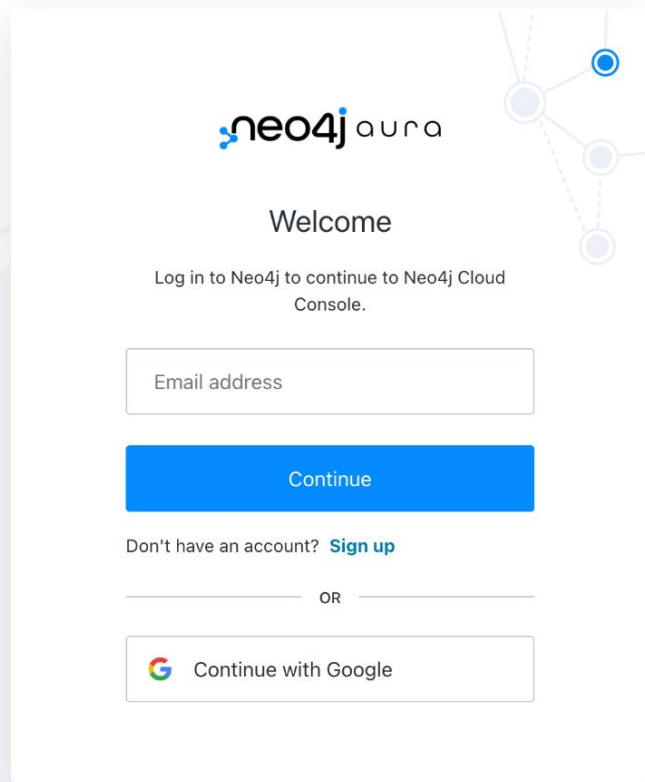
Welcome to Neo4j Aura

A fully managed, cloud-native graph data platform.

Neo4j AuraDB and AuraDS make it easy to build fast, scalable, and intelligent applications in the cloud, without managing complex infrastructure. Aura solutions offer:

- ✔ Lightning-fast query performance
- ✔ Fully-managed updates and patches
- ✔ Easy scalability, on-demand
- ✔ Robust security, reliability and ACID compliance
- ✔ Built-in tools to learn, build, and visualize

Start Free with AuraDB and join the largest graph community.
No Credit Card Required.

The image shows a screenshot of the Neo4j Aura welcome page. At the top right, there is a decorative graphic of a graph with nodes and edges. The main content area has the 'neo4j aura' logo, followed by the heading 'Welcome'. Below this is the instruction 'Log in to Neo4j to continue to Neo4j Cloud Console.' There is a text input field for 'Email address' and a prominent blue 'Continue' button. Below the button, it says 'Don't have an account? Sign up' with a link. A horizontal line with 'OR' in the center separates this from a 'Continue with Google' button, which features the Google logo.

Using Neo4j to store metrics

Using Neo4j to store metrics

They said:

"Neo4j is not optimized for metric data."

We said:

"Challenge accepted!"



Using Neo4j to store metrics (1st data model)

Example:

```
{
  labels: [
    "Gauge",
    "Metric"
  ],
  properties: {
    "agentId": "<Some NOM agent id>",
    "instanceId": "",
    "name": "memory.free",
    "value": 1253912576.0,
    "labels": "{}",
    "timestamp": datetime("1963-11-23T17:16:00.691000000Z")
  }
}
```

- 1 Node per data point
- Metric label on all, specific label (e.g. Gauge) where it applies
- Lots of repeated data (name, agent, instance) on all metric nodes
- Grows infinitely*
*until we run out of storage space

Using Neo4j to store metrics (2nd data model)

Example:

```
{
  labels: [
    "Metric",
    "Gauge",
    "agent:<Some NOM agent id>",
    "name:memory.free"
  ],
  properties: {
    "value": 12502708224.0,
    "timestamp": 1667833467
  }
}
```

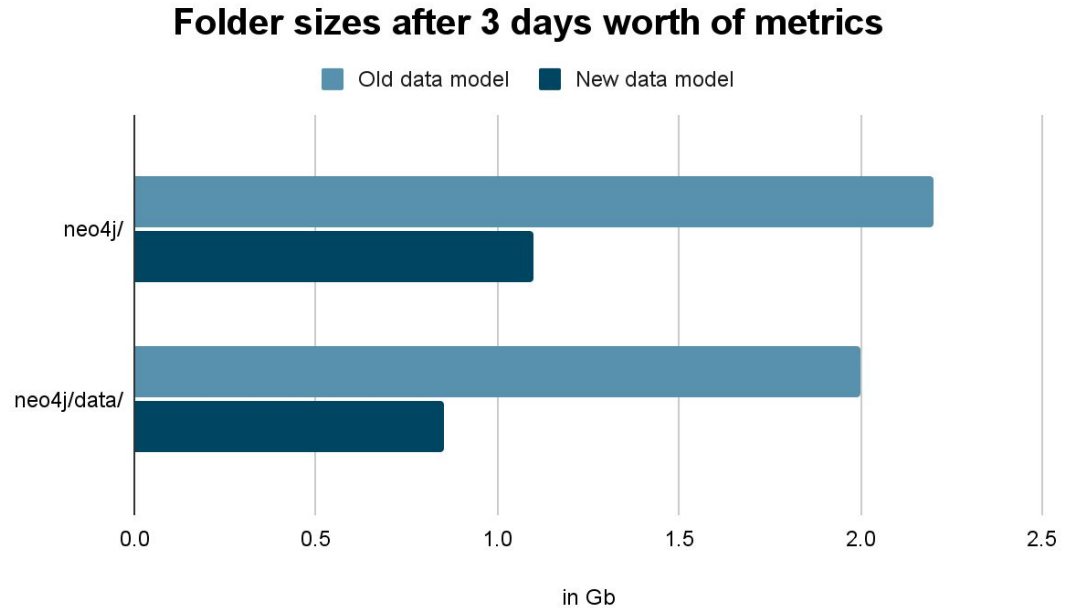
- 1 Node per data point
- Timestamps are now epoch second integers
- Get rid of unused property (where applies)
- Moved static information into labels so that we only have a reference instead of the full size of property on each node
- Grows infinitely*

*until we run out of storage space

Using Neo4j to store metrics (2nd data model)

Example:

```
{
  labels: [
    "Metric",
    "Gauge",
    "agent:<Some NOM agent id>",
    "name:memory.free"
  ],
  properties: {
    "value": 12502708224.0,
    "timestamp": 1667833467
  }
}
```



Using Neo4j to store metrics (2nd data model)

Example:

```
{
  labels: [
    "Metric",
    "Gauge",
    "agent:<Some NOM agent id>",
    "name:memory.free"
  ],
  properties: {
    "value": 12502708224.0,
    "timestamp": 1667833467
  }
}
```

- 1 Node per data point
- **Grows infinitely***
*until we run out of storage space

Using Neo4j to store metrics ("3rd" data model)

Example for aggregated node:

```
{
  labels: [
    "Metric",
    "Gauge",
    "Aggregate"
    "agent:<Some NOM agent id>",
    "name:memory.free"
  ],
  properties: {
    "min": 12502707224.0,
    "max": 12502709224.0,
    "numberOfDataPoints": 10,
    "value": 12502708224.0,
    "timestamp": 1667833467
  }
}
```

- 1 Node per data point
- Metrics data older than 3 days is aggregated to hourly summary node
- After aggregation: Remove all nodes that were aggregated
- Remove all aggregated nodes older than 30 days

Using Neo4j to store metrics

A note about removing/adding data in bulk

```
MATCH (n:Metric)
WHERE n.timestamp <= $aggregateFrom
DELETE n;
```

```
MATCH (n:Metric)
WHERE n.timestamp <= $aggregateFrom
CALL {
  WITH n
  DELETE n
} IN TRANSACTIONS OF 5000 ROWS;
```

Neo4j-Migrations

Neo4j-Migrations

"Damn! I just fundamentally changed the data model! What to do with the data in the old format?"

"Putting some schema into the schema-free database"

- Schema migrations as easy as possible
- Track, manage and apply changes to the database
- Builds directly on top of Neo4j-Java-Driver
- Modules: Core, CLI, Spring-Boot-Starter, Quarkus, Maven-Plugin

Kinds of migrations:

- Cypher-based
- Catalog-based
- Java-based

Cypher-based migration examples

```
DROP INDEX metrics IF EXISTS;  
CREATE RANGE INDEX metrics FOR (n:Metric) ON (n.timestamp);
```

```
CREATE (m:License)  
SET m.id = "placeholder"  
SET m.text = "This is a placeholder for the license text."  
SET m.version = 0;
```

Cypher-based migrations with preconditions

Assertions

Preconditions starting with `// assert` are hard requirements

Assumptions

Preconditions starting with `// assume` are soft requirements (migration will be skipped if fail to satisfy)

Require a certain edition

```
// assume that edition is enterprise
```

Require a certain version

```
// assume that version is 4.3
```

Preconditions based on Cypher queries

```
// assume q' RETURN true
```

Catalog-based migration example (1)

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<migration xmlns="https://michael-simons.github.io/neo4j-migrations">
  <catalog>
    <constraints>
      <constraint name="dbms_uniqueness" type="key">
        <label>DBMS</label>
        <properties>
          <property>id</property>
        </properties>
      </constraint>
    </constraints>
  </catalog>
  <drop item="metrics_agentId_exists"/>
  <drop item="metrics_instanceId_exists"/>
  <drop item="metrics_name_exists"/>
</migration>
```

Catalog-based migration example (2)

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<migration xmlns="https://michael-simons.github.io/neo4j-migrations">
  <?assume that version is lt 5 ?>
  <refactor type="migrate.createFutureIndexes">
    <parameters>
      <parameter name="suffix">_5_version</parameter>
    </parameters>
  </refactor>
</migration>
```

Java-based migration example

```
import ac.simons.neo4j.migrations.core.JavaBasedMigration;
import ac.simons.neo4j.migrations.core.MigrationContext;
import org.neo4j.driver.Session;

public class V001__DeleteOldMetrics implements JavaBasedMigration {

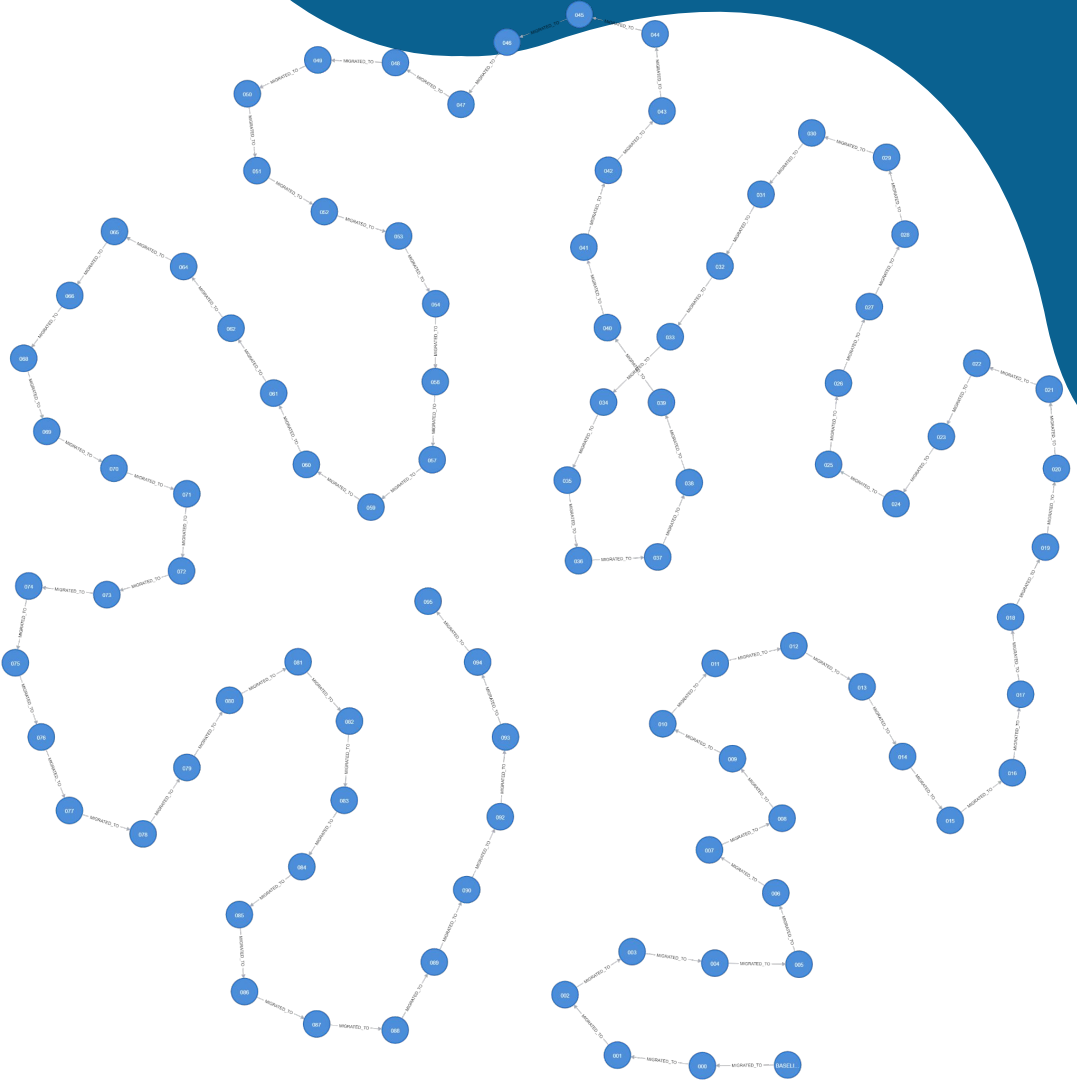
    @Override
    public void apply(MigrationContext context) {

        try (Session session = context.getSession()) {
            session.run("MATCH (m:Metric) CALL { WITH m DELETE m } IN
TRANSACTIONS OF 10000 ROWS");
        }
    }
}
```

One true chain

You can configure where the chain lives

Accidents can happen, you know?



The repair command

The repair operation behaves as follows:

- Check all the checksums (pairwise by migration version) and fix the recorded chain if necessary
- Check for missing local migrations and delete the missing ones in the database
- Check for inserted local migrations and create new chain entries with current timestamp

It does not apply the migrations!

It is not clean!

Example

Local migrations: v1*, v3 and v4

The chain recorded: (v1) → (v2) → (v3).

After repair: (v1*) → (v3)

Spring Data Neo4j

"Why spend time building graph-to-object mapping ourselves (and possibly fail) when SDN exist?"



- Reactive Programming
- 100% based on Neo4j Java Driver
- Immutable Mapping
- Spring Boot Integration
- Sensible Abstraction Layers
- Increases development speed

SDN: Node example (simplified)

```
import lombok.Builder;
import lombok.Value;
import org.springframework.data.annotation.Id;
import org.springframework.data.neo4j.core.schema.GeneratedValue;
import org.springframework.data.neo4j.core.schema.Node;
```

```
@Node("Metric")
@Value
@Builder(toBuilder = true)
public class Metric {
    @Id
    @GeneratedValue
    Long id;

    long timestamp; // as epoch time seconds

    double value;
}
```

SDN: create and save data example (simplified)

```
private final ReactiveNeo4jTemplate neo4j;
```

```
...
```

```
var metric = Metric.builder()  
    .timestamp(...)  
    .value(...)  
    .build();
```

```
neo4j.save(metric);
```

SDN: retrieve data example (simplified)

```
private final ReactiveNeo4jTemplate neo4j;
...
Mono<Metric> m = neo4j.findById("some id", Metric.class)

public interface MetricRepository
    extends ReactiveNeo4jRepository<Metric, String> {
    @Query("MATCH (m:Metric) WHERE $agentIdLabel in labels(m) RETURN m
           ORDER BY m.timestamp DESC")
    Flux<Metric> findByAgentId(String agentIdLabel);
}
...
@Autowired MetricRepository metrics;
...
Flux<Metric> m = metrics.findByAgentId("agent:smith")
```

SDN: projections

```
Mono<Metric> m = neo4j.findById("some id", Metric.class)
```

```
// interface based closed projection
```

```
interface ValuesOnly {  
    double getValue();  
}
```

```
...
```

```
interface MetricRepository extends Repository<Metric, String> {  
    Collection<ValuesOnly> findAllValuesBy(long timestamp);  
}
```

SDN: open projections (1)

```
interface ValuesAsLoglines {  
    @Value("#{ '[' + target.timestamp + ']: ' + target.value}")  
    String getLogLine();  
}
```

...

```
interface MetricRepository extends Repository<Metric, String> {  
    Collection<ValuesAsLoglines> findAllLoglinesBy(long timestamp);  
}
```

...

```
// example output of getLogLine()  
[-192696240]: 1.0
```

SDN: open projections (2)

```
interface ValuesAsLoglines {
    long getTimestamp();
    double getValue();

    @Value("#{ '[' + target.timestamp + ']: ' + target.value}")
    String getLogLine();

    default String getLogLine2() {
        return "[".concat(getTimestamp())
            .concat("]: ")
            .concat(getValue());
    }
}
```

"SDN Lifecycle" per feature

Answer to the question "Can we improve the performance?"

1. Use all/most of SDN
2. Start to use (closed) projections
3. Use repositories with custom Cypher
4. Run cypher via template directly and don't use object mapping

Cypher-DSL

Cypher-DSL

"There has to be an easier way to maintain these long/complex Cypher queries!"

- Avoid string concatenations in query generation
- Builder approach
- Great for complex queries
- Type-safe API for Cypher, checked at compile time
- Generates only valid Cypher constructs

Cypher-DSL example (1)

```
MATCH (m:Metric)
  WHERE $startTime <= m.timestamp AND m.timestamp < $endTime
RETURN m
ORDER BY m.timestamp
```

Cypher-DSL example (2)

```
Node metric = Cypher.node("Metric").named("m");
Condition condition = Cypher.parameter("startTime", startTime)
    .lte(metric.property("timestamp"))
    .and(
        metric.property("timestamp")
        .lt(Cypher.parameter("endTime", endTime)) );

var query = Cypher.match(metric)
    .where(condition)
    .returning(metric)
    .orderBy(metric.property("timestamp"))
    .build();

Flux<Metric> metrics = metricRepository.findAll(query);
```

Surprise!

I promised one, remember?



Sebastian Fabian 3:43 PM

One query fixed (`getMetricConditions`).. 😊

Screenshot 2023-05-11 at 15.42.48.png ▾



Jens Wollert Ehlers 3:47 PM

Using NOM to improve NOM. Loving it!

Using NOM to improve NOM

Insert meta joke here

NOM Architecture



Using NOM to improve NOM



So, what benefits did dogfooding get you?

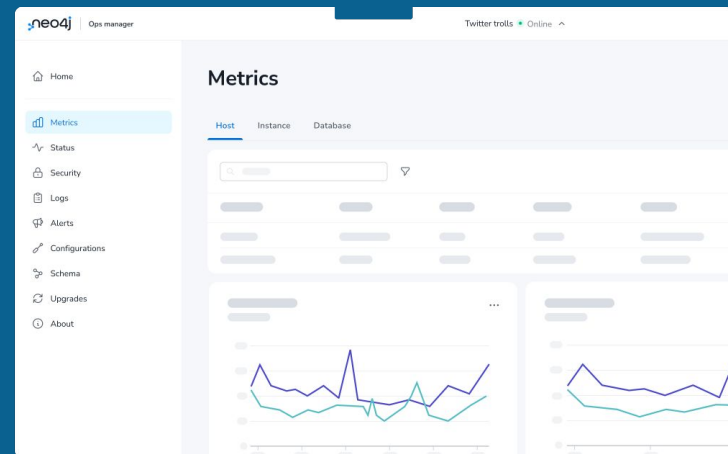
"But what have the ~~romans~~ developers ever done for us?!"

(Incomplete) List of dogfooding wins

- **Development speed!**
- **Bug fixes:**
 - Neo4j
 - Bug in Cypher planner
 - Issue found with the startup ignoring the server-logs.xml file unless specified in the neo4j.conf on an RPM install
 - Lots of documentation issues/enhancements (e.g. database role can be null with composite databases)
 - Got rid of wrong warnings in logs while preparing for this talk
 - SDN
 - Performance issue due to unnecessary ToString() on id()
- **New Features:**
 - Neo4j
 - SHOW SUPPORTED PRIVILEGES command
 - SHOW INDEXES to show last usage by a query
 - Introduced information about valid values for configuration options
 - Neo4j-Migrations
 - Repair command
 - Found breaking change in 1.13.2
 - Migrating BTREE indexes to "future" indexes
 - Detect invalid use of enterprise constraints against community edition
 - Wait for new indexes to get online
- **Coordination:**
 - Outside view on inconsistencies in product surface
 - Performance tips and consulting from our SDN team
- **Performance improvements of NOM queries**
- **This talk!**
 - Will be used as onboarding material for new team members and in company-wide knowledge sharing sessions

Dogfooding the graph ecosystem: Learnings from building NOM

- Data modeling is fun with a graph
- Neo4j can be used to store metrics
- Dogfooding: great source of feedback, for finding bugs and missing features
- The Neo4j ecosystem is very rich and helpful
 - Spring Data Neo4j
 - Neo4j-Migrations
 - Cypher-DSL
 - Neo4j drivers
 - Prometheus endpoint in Neo4j
 - Neo4j Aura
 - Neo4j Ops Manager
 - ...
 - So much more that we did not use (yet)





October 26, 2023

- 3 tracks:
 - Building Applications & APIs
 - ML & AI with Graphs
 - Powerful Visualizations
- 24 hours
- 100 talks

[Register for free! - neo4j.com/nodes](https://neo4j.com/nodes)

Thank you!

Questions?

Sascha Peukert

Software Engineer

sascha.peukert@neo4j.com

@SasPeuk