openCypher

An open standard for graph queries:
the Cypher contribution

Cypher: is the query language

openCypher:
a project for open collaboration on Cypher

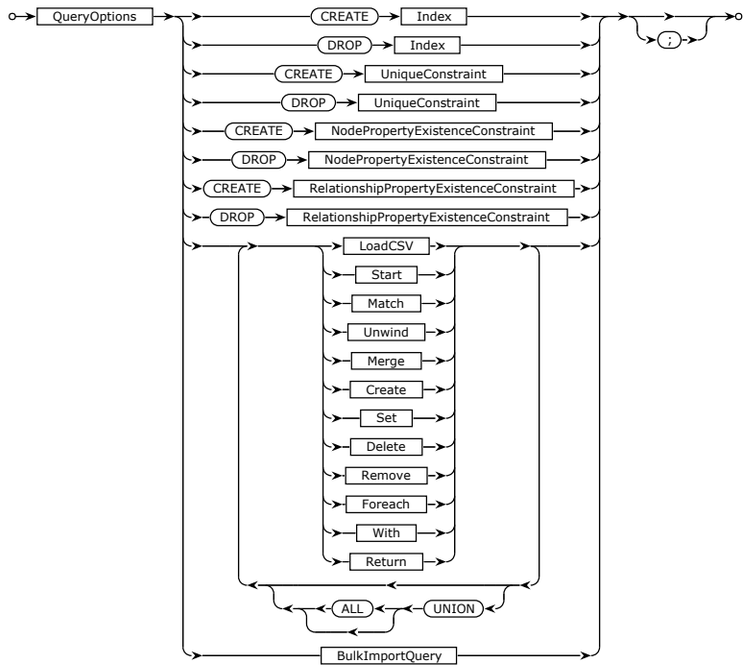
Neo4j provides an implementation of Cypher

```
(openCypher)-[:DEFINES]->(nextCypher)

(nextCypher{version:"pre 1.0"})
        -[:SUBSET_OF]->
  (neo4jCypher{version:"3.0"})

(neo4j{version:"3.x"})-[:IMPLEMENTS]->(nextCypher)
```

Status report

- openCypher announced in October 2015
- Neo4j have opened up resources that were internal
  - Grammar
    - Current Neo4j Grammar Complete
  - Tests - as a *Technology Compatibility Kit*
    - Current Neo4j Tests ported: ETA July
  - Cypher Improvement Proposals (CIPs)
    - New proposals public - existing body being ported
- Evolving both language and resources under openCypher

Cypher: (grammar)



Railroad diagram for the top-level of the Neo4j 3.0 Cypher grammar:
https://s3.amazonaws.com/artifacts.opencypher.org/railroad/Cypher.html

Cypher Technology Compatibility Kit (TCK)

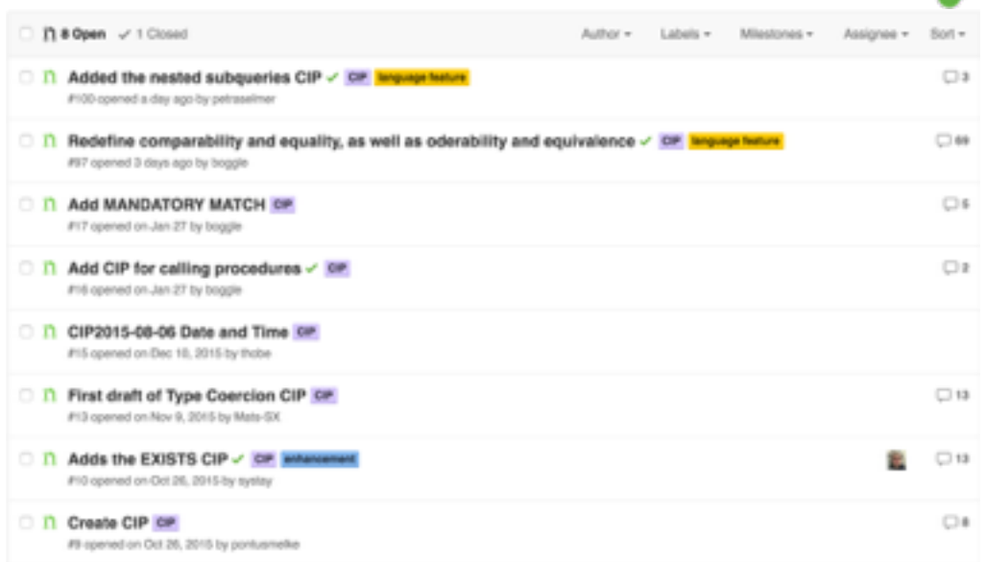
```
Scenario: Merge node with label when it exists
  Given an empty graph
  And having executed:
    """
    CREATE (:One{alpha:"foo",beta:"bar"})
    """
  When executing query:
    """
    MERGE (a:One{alpha:"foo"})
    RETURN a.beta
    """
  Then the result should be:
    | a.beta |
    | 'bar'  |
  And no side effects
```


Example of a test scenario from the Cypher TCK, using *Cucumber feature files*.
An implementation of Cypher would implement a test runner that consumes files.

Cypher Improvement Proposals

- Created as Pull Requests on the openCypher github repository https://github.com/opencypher/openCypher/pulls
- Document describing the new feature
- Updates to the specification
  - Updating relevant document
  - Adding and changing TCK test cases
  - Updating the Reference Implementation (when available)

Cypher Improvement Proposals



Proposed in the open, discussed in public using GitHub comments

Our thinking for what is next for openCypher

- Reference Implementation (for the Java platform)
- Textual (informal) semantic specification
- Building community - engaging partners
- Composition with other languages
  - Scripting and expressing algorithms
  - SQL embedding
- Ideas for new features for Cypher (unordered):
  - temporal types and functions
  - subqueries (Improvement Proposal available as of today)
  - Conjunctive Regular Path Queries (CRPQs)
  - enhanced schema support
  - et.c.

Community input is crucial for prioritisation

Making it easier for other projects & vendors to implement Cypher

- Open Source reference implementation
- Liberal License (Apache Software License)
- Java based
  - Open to contributions of reference implementations for other platforms
- Language agnostic Test Suite - Cypher Technology Compatibility Kit (TCK)

Different profiles of Cypher

- Based on different *vocabularies* of the language
- Read-only / Read+write
- Transactional / Non-transactional
- With / without schema

- Vendor specific extensions
  - functions
  - datatypes
  - aggregations
  - (named) infix operators
- Extension procedures

What have we learned from the LDBC Query Language task force?

as members of the LDBC Query Language Task Force
we are folding good ideas from the Task Force into cypher

What have we learned from the LDBC Query Language task force?

Shortest path first

What have we learned from the LDBC Query Language task force?

(Conjunctive) Regular Path Queries **(CRPQs)**
based on
Regular expressions with memory **(REMs)**

Who should get involved in openCypher?

- Graph System vendors
- Tool vendors
- Graph researchers
- Query Language researchers
- Enthusiasts

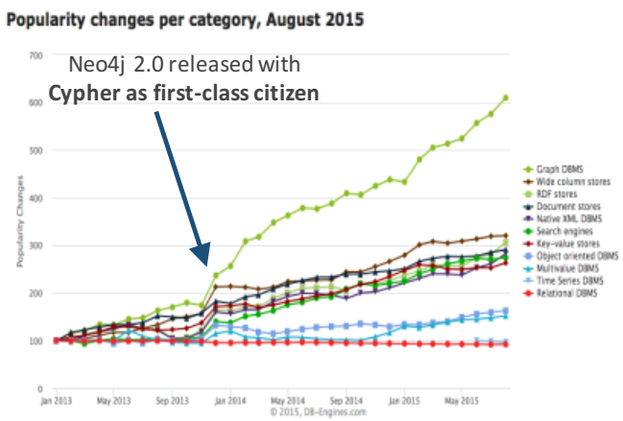
Cypher History

- Pragmatic background
  - Built on experience from building applications with Graph Data
- Battle tested
  - Has been in use for a long time
  - Most wrinkles have been straightened out
    (of what is currently in the language)
- Supports both Read and Write
- Large existing user base
  - the largest graph query language measured by user adoption

Cypher Adoption

- 2M+ downloads of Neo4j
  - 50-100k per month
  - 95% of Neo4j users use Cypher
- Available learning resources:
  - Online training, docs, O'Reilly Graph Databases book, …
  - 30,000+ registrants to Neo's online training
- Ecosystem of tooling providers
- Most users really love Cypher

Most property graph use is with Cypher
70+% according to DB-Engines (Nov 2015)



How to get involved in openCypher!

- Get on the mailing list!
  https://groups.google.com/forum/#!forum/opencypher
- Contribute code on GitHub
  https://github.com/opencypher/openCypher
- Contribute specification proposals on GitHub

- The way to get more involved is by being more active

- Tell us (the existing community) how you would like to contribute!

Cypher is open for collaboration

- All development on http://github.com/openCypher
  - Grammar
  - TCK - Compatibility tests
  - Specification (some there - more coming)
  - Reference Implementation (coming)
  - Improvement Proposals (CIPs) as Pull Requests
- Currently managed by 4 Cypher developers - the *Cypher Language Group*
- Please get involved - comment, propose, contribute!

we are open to suggestions on how to improve the process

Thank you!

Questions?