Cypher is the declarative query language for Neo4j, the world’s leading graph database.

Key principles and capabilities of Cypher are as follows:

- Cypher matches patterns of nodes and relationship in the graph, to extract information or modify the data.
- Cypher has the concept of identifiers which denote named, bound elements or parameters.
- Cypher can create, update, and remove nodes, relationships, labels, and properties.
- Cypher manages indexes and constraints.

You can try Cypher snippets live in the Neo4j Console at console.neo4j.org or read the full Cypher documentation at neo4j.com/developer. Live graph models using Cypher check out GraphGo.

The Cypher Refcard is also available in PDF format.

### Syntax

#### Read Query Structure

- **MATCH**
  - `MATCH (n:Label)`
  - Matches any pattern that can be used in `LOAD CSV`.
- **UNION**
  - `UNION (n:Label)`
  - Optional pattern; will be used for missing parts.
- **WHERE**
  - `WHERE expr` or `expr = true`
  - For the planner to use a label scan to solve the query (for manual performance tuning).

#### Write-Only Query Structure

- **CREATE**
  - `CREATE (n:Label)`
  - Create a node with the given properties.
  - `CREATE (n:Label) WITH n.properties` or `create n.labels = label, n.properties = prop`
  - Create nodes with the given properties.
- **MATCH**
  - `MATCH (n:Label)`
  - Create a relationship with the given type and direction; bind a variable to it.
  - `MATCH (n:Label) WITH n.labels = label, n.properties = prop`
  - Create a relationship with the given type, direction, and properties.

#### FOR each

- `FOR m IN n` or `FOR m IN n(COUNT)`
  - Execute a mutating operation for each relationship of a path.
- `FOR m IN c` or `FOR m IN c(COUNT)`
  - Execute a mutating operation for each element in a list.

#### DELETE

- `DELETE n` or `DELETE n labels, n.properties`
  - Delete a node and all relationships connected to it.
- `DELETE n:Label` or `DELETE n:Label WITH n.properties`
  - Delete all nodes and relationships from the database.

### Patterns

**Node patterns**
- `n` or `n:`
  - A pattern for a node.
- `n:Label`
  - A pattern for a node with the given label.
- `n:Label` WITH `n.properties = prop`
  - A pattern for a node with the given label and properties.

**Relationship patterns**
- `n r Label:Label n2`
  - A pattern for a relationship of type `Label` between nodes `n` and `n2`.
- `n r Label:Label n2:Label`
  - A pattern for a relationship of type `Label` between nodes `n` and `n2` with labels.

#### Indexes

- `CREATE INDEX r:Label property` or `create index r:Label property` or `CREATE INDEX r:Label property ON n:Label (property)`
  - Create an index on the value of a property `property` for the specific label `Label`.
- `CREATE INDEX r:Label property` or `create index r:Label property`
  - Create a unique property constraint on the label `Label` for property `property`.

### Constraints

- `CREATE CONSTRAINT r:Label property` or `create constraint r:Label property`
  - A constraint that enforces uniqueness on the value of a property `property` for the specific label `Label`.

### Operations

- `RETURN *`
  - Return the value of all variables.
- `RETURN value
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr`
  - Use alias for result column name.
- `RETURN DISTINCT n
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr`
  - Return unique variables.
- `ORDER BY property`
  - Sort the results.
- `ORDER BY expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr`
  - Sort the result in descending order.

### Labs

- **Relationship lab**
  - `n:Label`
  - Node pattern.
- **Create lab**
  - `CREATE (n:Label)`
  - Create a node with given properties.
- **Update lab**
  - `MATCH (n:Label) WITH n.properties = prop
  SET n.properties = prop
  RETURN n
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr`
  - Update or create a property.

### Try It Out

- **Neo4j Cypher Refcard 3.0.0**
  - Neo4j is the world’s leading graph database.
  - Neo4j properties can be strings, numbers, booleans, or arrays thereof.
  - Neo4j also supports maps and lists.

#### Neo4j Cypher Refcard 3.0.0

- **CREATE**
  - `CREATE (n:Label)`
  - Create a node with given properties.
  - `CREATE (n:Label) WITH n.properties = prop`
  - Create nodes with the given properties.
- **MATCH**
  - `MATCH (n:Label)`
  - Create a relationship with the given type and direction; bind a variable to it.
  - `MATCH (n:Label) WITH n.labels = label, n.properties = prop`
  - Create a relationship with the given type, direction, and properties.

### Indexes

- `CREATE INDEX r:Label property` or `create index r:Label property` or `CREATE INDEX r:Label property ON n:Label (property)`
  - Create an index on the value of a property `property` for the specific label `Label`.
- `CREATE INDEX r:Label property` or `create index r:Label property`
  - Create a unique property constraint on the label `Label` for property `property`.

### Constraints

- `CREATE CONSTRAINT r:Label property` or `create constraint r:Label property`
  - A constraint that enforces uniqueness on the value of a property `property` for the specific label `Label`.

### Operations

- `RETURN *`
  - Return the value of all variables.
- `RETURN value
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr`
  - Use alias for result column name.
- `RETURN DISTINCT n
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr`
  - Return unique variables.
- `ORDER BY property`
  - Sort the results.
- `ORDER BY expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr`
  - Sort the result in descending order.

### Labs

- **Relationship lab**
  - `n:Label` Node pattern.
- **Create lab**
  - `CREATE (n:Label)` Create a node with given properties.
- **Update lab**
  - `MATCH (n:Label) WITH n.properties = prop
  SET n.properties = prop
  RETURN n
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr`
  - Update or create a property.

### Try It Out

- **Neo4j Cypher Refcard 3.0.0**
  - Neo4j is the world’s leading graph database.
  - Neo4j properties can be strings, numbers, booleans, or arrays thereof.
  - Neo4j also supports maps and lists.

#### Neo4j Cypher Refcard 3.0.0

- **CREATE**
  - `CREATE (n:Label)`
  - Create a node with given properties.
  - `CREATE (n:Label) WITH n.properties = prop`
  - Create nodes with the given properties.
- **MATCH**
  - `MATCH (n:Label)`
  - Create a relationship with the given type and direction; bind a variable to it.
  - `MATCH (n:Label) WITH n.labels = label, n.properties = prop`
  - Create a relationship with the given type, direction, and properties.

### Indexes

- `CREATE INDEX r:Label property` or `create index r:Label property` or `CREATE INDEX r:Label property ON n:Label (property)`
  - Create an index on the value of a property `property` for the specific label `Label`.
- `CREATE INDEX r:Label property` or `create index r:Label property`
  - Create a unique property constraint on the label `Label` for property `property`.

### Constraints

- `CREATE CONSTRAINT r:Label property` or `create constraint r:Label property`
  - A constraint that enforces uniqueness on the value of a property `property` for the specific label `Label`.

### Operations

- `RETURN *`
  - Return the value of all variables.
- `RETURN value
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr`
  - Use alias for result column name.
- `RETURN DISTINCT n
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr`
  - Return unique variables.
- `ORDER BY property`
  - Sort the results.
- `ORDER BY expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr`
  - Sort the result in descending order.

### Labs

- **Relationship lab**
  - `n:Label` Node pattern.
- **Create lab**
  - `CREATE (n:Label)` Create a node with given properties.
- **Update lab**
  - `MATCH (n:Label) WITH n.properties = prop
  SET n.properties = prop
  RETURN n
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr
  OR expr`
  - Update or create a property.
Cypher is the declarative query language for Neo4j, the world’s leading graph database.

Key principles and capabilities of Cypher are as follows:

- Cypher matches patterns of nodes and relationship in the graph, to extract information or modify the data.
- Cypher has the concept of identifiers which denote named, bound elements or parameters.
- Cypher can create, update, and remove nodes, relationships, labels, and properties.
- Cypher manages indexes and constraints.

You can try Cypher snippets live in the Neo4j Console at console.neo4j.org or read the full Cypher documentation in the Neo4j Manual for live graph models using Cypher check out Graphite.

The Cypher Refcard is also available in PDF format.

Note: (value) denotes either literals, for all but Cypher queries; or parameters, which is the best practice for applications. Use (value) properties can be strings, numbers, booleans or arrays thereof. Cypher also supports maps and lists.

Syntax

### Read Query Structure

**MATCH**

MATCH (n:Person) WHERE n:Person ORDER BY n:Person

### Write Only Query Structure

CREATE n:

### Read Write Query Structure

CREATE n:

### UPDATE or create a property

UPDATE n:

### DELETE a node and a relationship

DELETE n:

### DELETE all nodes and relationships from the database

### FOR Each

FOR n IN COL

FOR n, p IN COL

### CALL

CALL db:Nodes

### CREATE

CREATE (n:Person) WHERE n:Person

### INDEX

CREATE INDEX ON :Person(name)

Cypher is the declarative query language for Neo4j, the world’s leading graph database.

Key principles and capabilities of Cypher are as follows:

- Cypher matches patterns of nodes and relationship in the graph, to extract information or modify the data.
- Cypher has the concept of identifiers which denote named, bound elements or parameters.
- Cypher can create, update, and remove nodes, relationships, labels, and properties.
- Cypher manages indexes and constraints.

You can try Cypher snippets live in the Neo4j Console at console.neo4j.org or read the full Cypher documentation in the Neo4j Manual for live graph models using Cypher check out Graphite.

The Cypher Refcard is also available in PDF format.

Note: (value) denotes either literals, for all but Cypher queries; or parameters, which is the best practice for applications. Use (value) properties can be strings, numbers, booleans or arrays thereof. Cypher also supports maps and lists.

Syntax