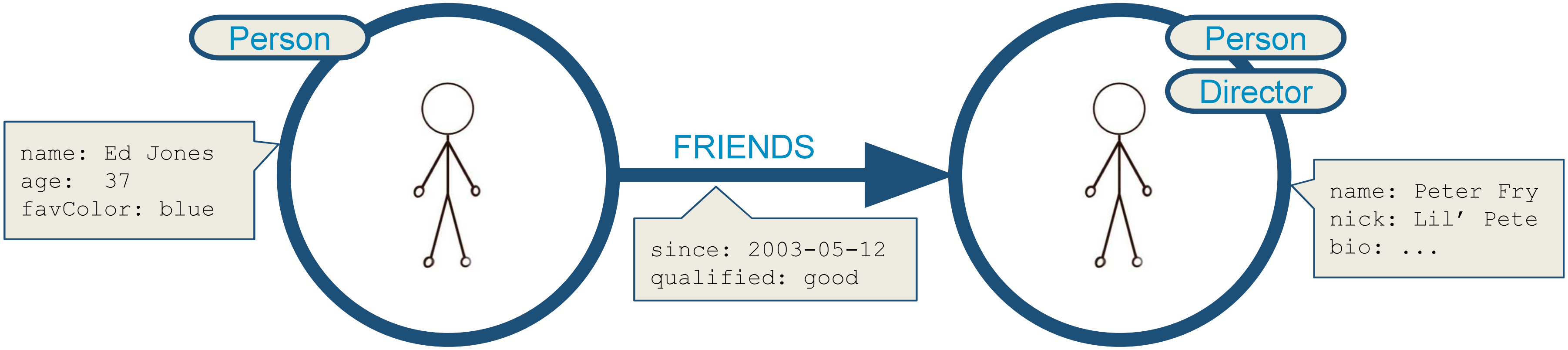
Property Graphs



Person

name: Ed Jones
age: 37
favColor: blue

FRIENDS

since: 2003-05-12
qualified: good

Person

Director

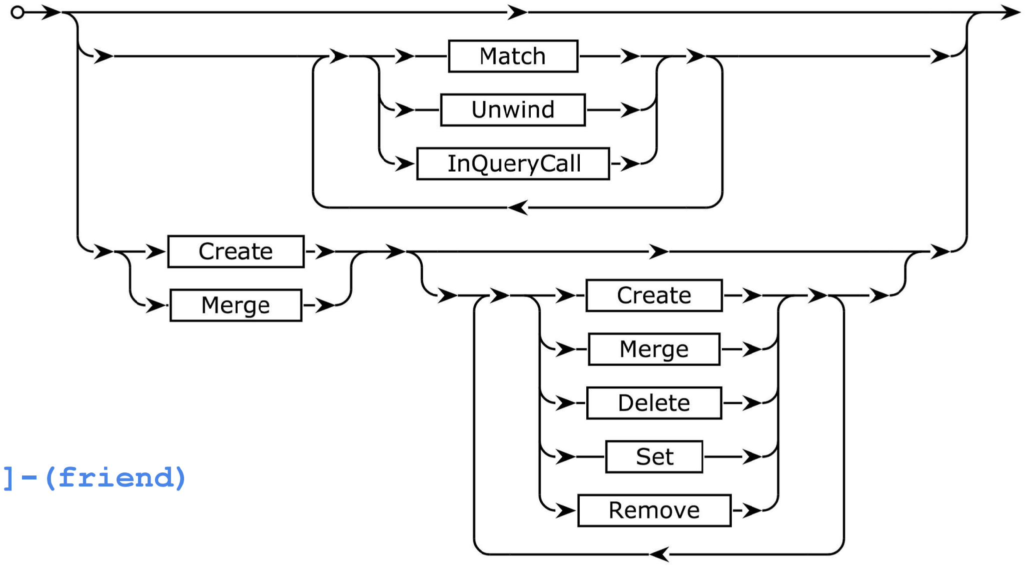
name: Peter Fry
nick: Lil’ Pete
bio: ...

Cypher

- Pattern Matching:
```
MATCH (me:Person)-[:FRIEND]->(friend)
```
- Filtering:
```
WHERE me.name = "Frank Black"
AND   friend.age > me.age
```
- Projection:
```
RETURN friend.name, friend.title
```

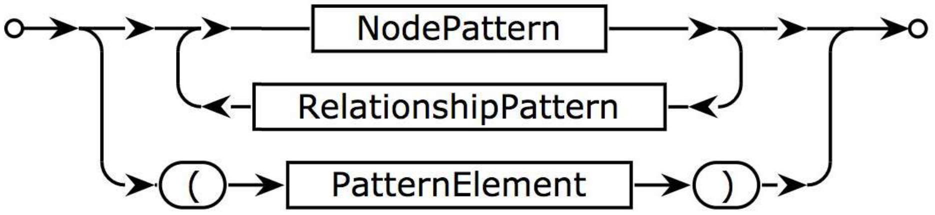
- Data creation and manipulation:
```
CREATE (you:Person)
SET you.name = "Aaron Fletcher"
CREATE (you)-[:FRIEND]->(me)
```

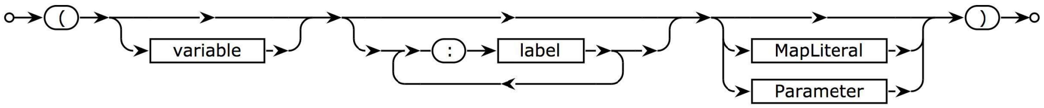
- Sequential Query Composition:
```
MATCH (me:Person{name:$name})-[:FRIEND]-(friend)
WITH me, count(friend) AS friends
MATCH (me)-[:ENEMY]-(enemy)
RETURN friends, count(enemy) AS enemies
```

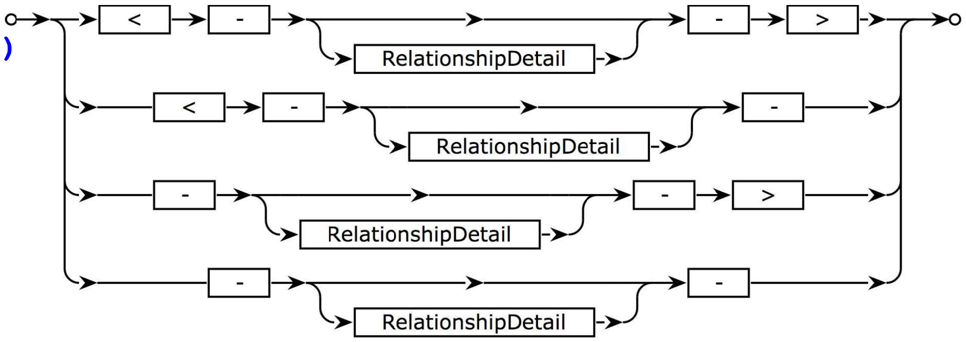


Patterns

```
MATCH (), (node), (node:Node), (:Node),
      (node {type:"NODE"})

MATCH ()-->(), ()-[edge]->(),
      ()-[edge:RELATES]->(),
      ()-[:RELATES]->(),
      ()-[edge {score:5}]->(),
      (a)-[edge]->(b)
      (a)<-[edge]-(b), (a)-[edge]-(b)

MATCH p = (a)-[:ONE]-()-[:TWO]-()-[:THREE]-()

MATCH p = (me)-[:FRIEND*]-(foaf)

MATCH (me)-[:FRIEND*1..3]-(foaf)
```


NodePattern
RelationshipPattern
(
PatternElement
)


(
variable
:
label
MapLiteral
Parameter
)


<
-
RelationshipDetail
-
>
<
-
RelationshipDetail
-
-
RelationshipDetail
-
>
-
RelationshipDetail
-