Neo4j + Haskell + hasbolt

Кольцов Максим
Мирзоев Денис

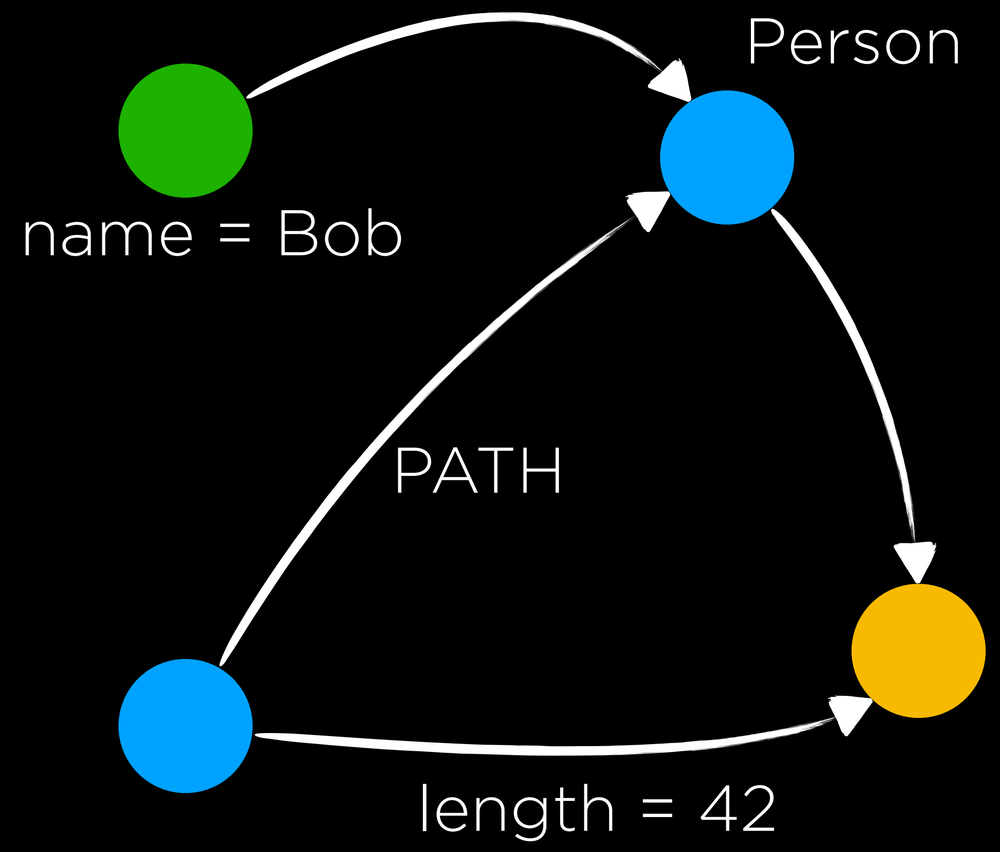
25 мая 2019 года

BIOCAD
Biotechnology Company

neo4j

Что такое графовая БД

- Сущности (вершины, узлы)
- Связи (рёбра, отношения)
- Свойства
- Метки



Установка

```
git clone
https://github.com/biocad/neo4j-workshop
```

Установка

- Docker
  - https://docs.docker.com/docker-for-mac/install/
  - https://docs.docker.com/install/linux/docker-ce/ubuntu/

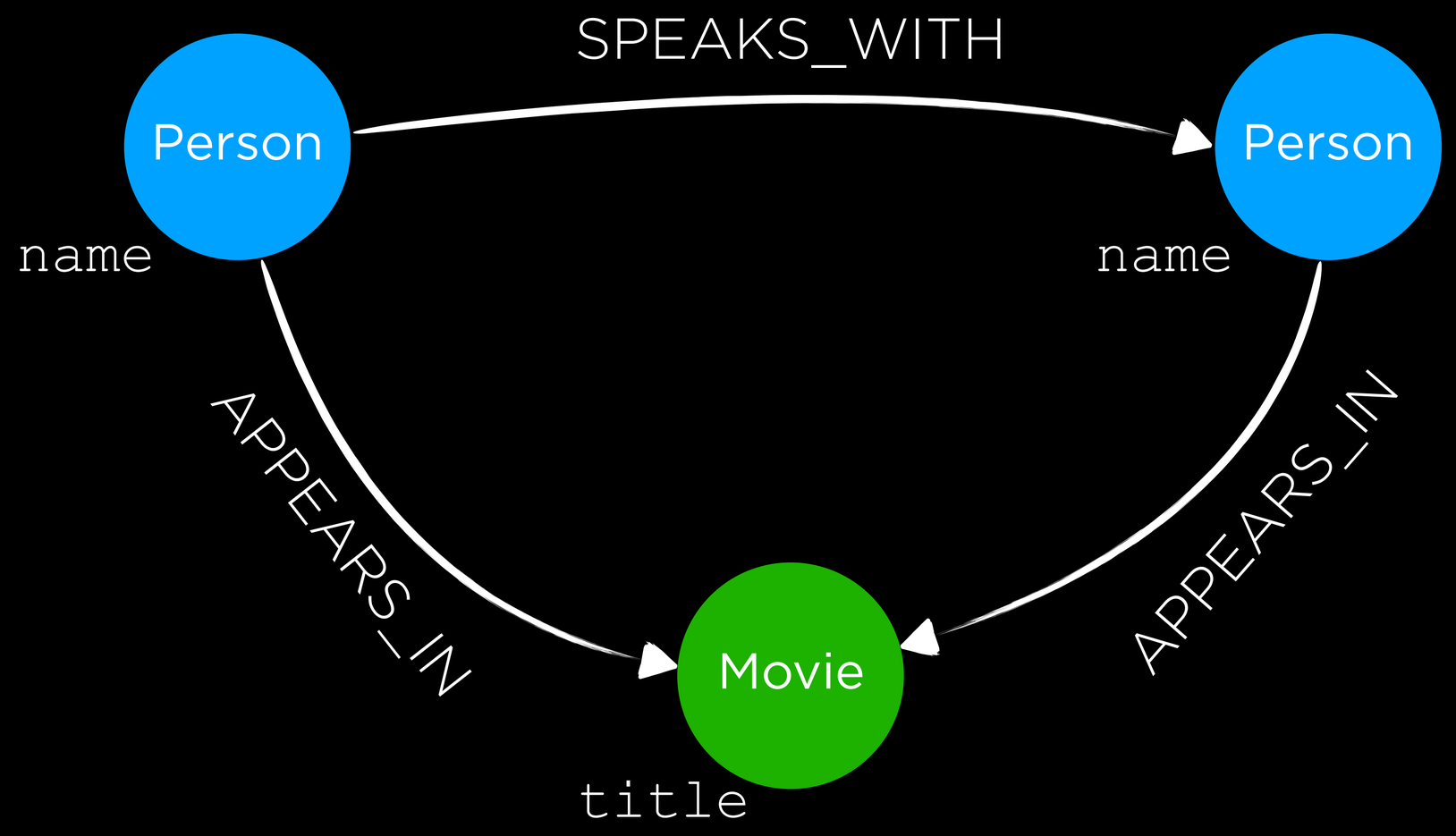
- stack
  - curl -sSL https://get.haskellstack.org/ | sh

- Neo4j
  - docker pull maksbotan/neo4j-fpure
  - docker run -d -p 7474:7474 -p 7687:7687 maksbotan/neo4j-fpure
  - http://localhost:7474

∝ Репозиторий

```
src/
   |-- Config.hs
   `-- Types.hs
tasks/
   |-- Sample.hs
   |-- Task1.hs
   `-- Task2.hs
package.yaml
stack.yaml
```

Датасет


SPEAKS_WITH
Person
Person
name
name
APPEARS_IN
APPEARS_IN
Movie
title

Cypher

- Узлы
  - `(), (n)`
  - `(m: Movie), (:Person)`
- Отношения
  - `[], [r]`
  - `[s: SPEAKS_WITH]`
- Пути
  - `(m) -> (p)`
  - `(m) -- (p)`
  - `(m) -[s]- (p) -> ()`

Cypher — поиск в базе

В каких фильмах появлялся Anakin?

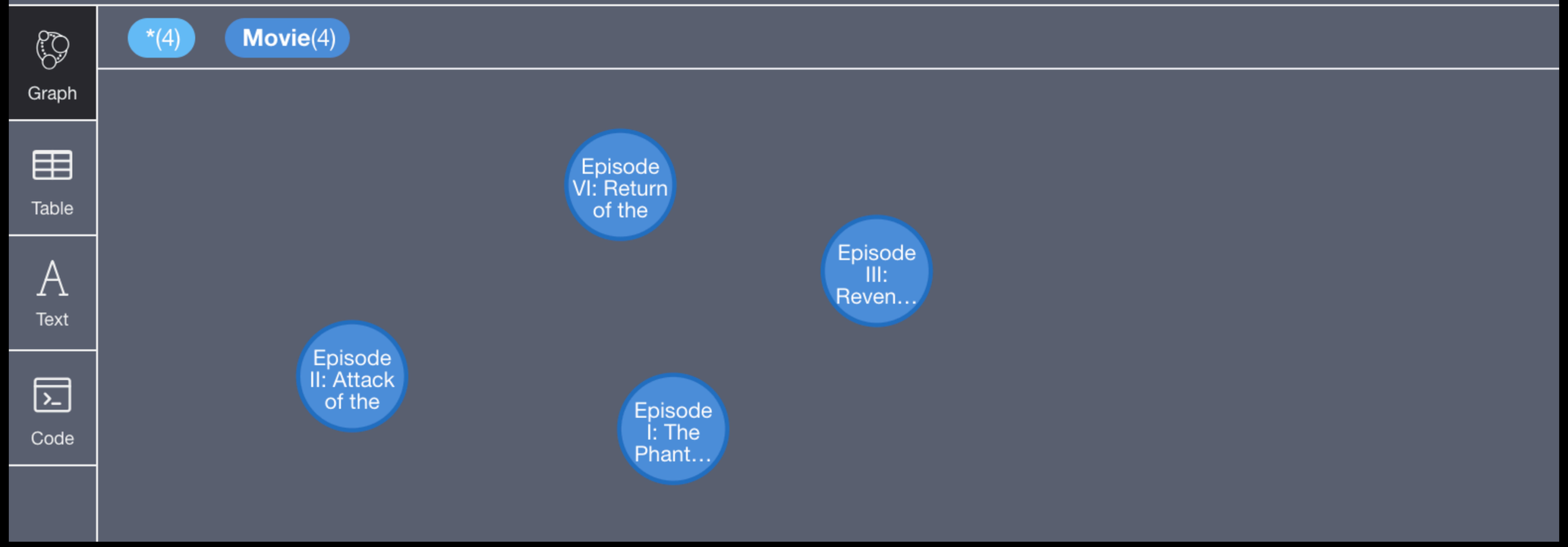

```
MATCH (anakin: Person) -[:APPEARS_IN]- (movie: Movie)
WHERE anakin.name = 'ANAKIN'
RETURN movie
```

Cypher — что можно вернуть

- Узлы и отношения
  - `RETURN m, s, p`
- Свойства
  - `RETURN m.title, p.name`
  - `RETURN s.name AS person_name`
- Функции
  - `RETURN count(m)`

Пример

```
$ MATCH (anakin: Person) -[:APPEARS_IN]- (movie: Movie) WHERE anakin.name = 'ANAKIN' RETURN movie
```



Пример

Сколько персонажей в каждом фильме?

```
MATCH
   (m: Movie) -[:APPEARS_IN]- (character)
RETURN
    m.title AS movie,
    count(character) AS characters
```

Пример

```
$ MATCH (m: Movie) -[:APPEARS_IN]- (character) RETURN  m.title AS movie, count(*) AS charac…
```

Table | Text | Code

movie	characters
"Episode I: The Phantom Menace"	38
"Episode II: Attack of the Clones"	33
"Episode III: Revenge of the Sith"	25
"Episode IV: A New Hope"	22
"Episode V: The Empire Strikes Back"	21
"Episode VI: Return of the Jedi"	20
"Episode VII: The Force Awakens"	27

hasbolt

hasbolt

```
MATCH
  (person: Person) -[:APPEARS_IN]- (movie: Movie)
RETURN movie, person.name
```

Node

Value

Record

∝ Типы

```haskell
data Value
  = N ()
  | B Bool
  | I Int
  | F Double
  | T Text
  | L [Value]
  | M (Map Text Value)
  | S Structure
```

```haskell
data Node = Node
  { nodeIdentity :: Int
  , labels       :: [Text]
  , nodeProps    :: Map Text Value
  }
data Relationship = ...
data URelationship = ...
```

```haskell
type Record = Map Text Value
```

Подключение

```
import Database.Bolt
import Data.Default

cfg :: BoltCfg
cfg = def
  { host = "...", user = "...",
  , password = "..."
  }

main :: IO ()
main = do
  pipe <- connect cfg
  ...
```

src/Config.hs

Запросы

```
query :: Text -> BoltActionT m [Record]
run :: Pipe -> BoltActionT m a -> m a
```

```
result <- run pipe $ query "..."
```

Разбор ответа

```
at :: Monad m => Record -> Text -> m Value
exact :: (RecordValue a, Monad m) => Value -> m a
```

Разбор ответа

```
let anakinQ =
  "MATCH (anakin: Person) -[:APPEARS_IN]- (movie: Movie)\
  \WHERE anakin.name = 'ANAKIN'\
  \RETURN movie"

result <- run pipe $ query anakinQ

forM_ result $ \record -> do
    movie     <- rec `at` "movie" >>= exact @Node
    movieName <- exact @Text $ nodeProps movie ! "title"
    putStrLn $ unpack movieName
```

Разбор ответа с линзами

```
field :: RecordValue a => Text -> Fold Record a
prop  :: RecordValue a => Text -> Fold Node   a
```

```
forM_ result $ \record -> do
    let movieName =
          record ^?! field "movie" . prop "title"
    putStrLn $ unpack movieName
```

Задача 1

- Сколько персонажей в каждом фильме? Запрос дан, запустить его и прочитать ответ
- С кем говорит Люк в каждом фильме? Написать запрос и запустить его

hasbolt-extras

Пробле

```
MATCH
   (anakin: Pe                                    ovie: Movie)
WHERE anakin.
RETURN movie
```

2 x String

```
forM_ result
    let movieN
          reco                                    prop "title"
    putStrLn S
```

∞ Типы

```
data Movie = Movie
  { title :: Text }

data Person = Person
  { name :: Text }

data SPEAKS_WITH = SPEAKS_WITH
data APPEARS_IN = APPEARS_IN
```

src/Types.hs

Типы

Partial

```
class NodeLike a where
  toNode :: a -> Node
  fromNode :: Node -> a
```

```
makeNodeLike ''Movie
makeNodeLike ''Person

makeURelationLike ''SPEAKS_WITH
makeURelationLike ''APPEARS_IN
```

Database.Bolt.Extras

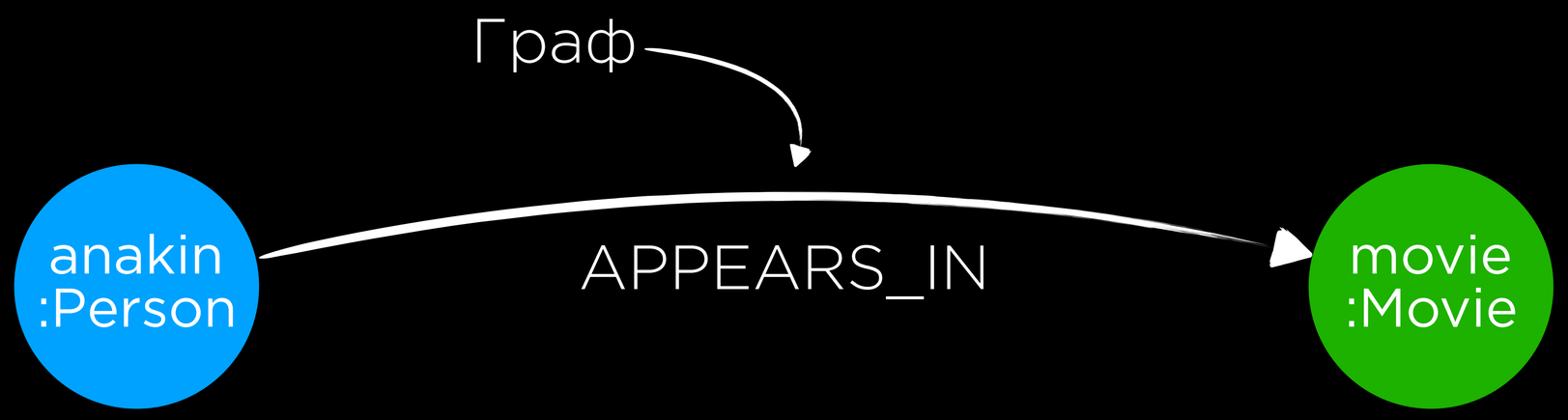
Разбор ответа

```
forM_ res $ \rec -> do
  let Movie movieName = fromNode $
        rec ^?! field "movie"
  putStrLn $ unpack movieName
```

Задача 2

- Воспользоваться `fromNode` для распаковки ответа

hasbolt-extras



Строка!

```
MATCH (anakin: Person) -[:APPEARS_IN]- (movie: Movie)
WHERE anakin.name = 'ANAKIN'
RETURN movie
```

Шаблон графа

Имена Узлы Отношения

```
data Graph n a b = Graph { _vertices, _relations }
emptyGraph :: Graph n a b
addNode ::
  n -> a -> Graph n a b -> Graph n a b
addRelation ::
  n -> n -> b -> Graph n a b -> Graph n a b
```

Шаблоны узлов

```
data NodeGetter
data RelGetter
class GetterLike a where
  withBoltId :: BoltId -> a -> a
  withLabel :: Text -> a -> a
  withLabelQ :: Name -> a -> a
  withProp, withReturn, isReturned
```

Пример запроса

?

```
anakinGraph :: GraphGetRequest
anakinGraph = emptyGraph
  & addNode "anakin" ...
  & addNode "movie" ...
  & addRelation "anakin" "movie" ...
```

Пример запроса

TemplateHaskell

```
defaultNodeReturn
  & withLabelQ ''Person
  & withProp ("name", T "ANAKIN")
  & withReturn allProps
```

Value

Всё вместе

```
anakinGraph = emptyGraph
  & addNode "anakin"
      (defaultNodeNotReturn
        & withLabelQ ''Person
        & withProp ("name", T "ANAKIN"))
  & addNode "movie"
      (defaultNodeReturn
        & withLabelQ ''Movie
        & withReturn allProps)
  & addRelation "anakin" "movie"
      (defaultRelNotReturn
        & withLabelQ ''APPEARS_IN)
```

Чтение из базы

```
makeRequest
  :: GraphQuery a
  => [Text] -> GraphGetRequest
  -> BoltActionT m [GraphResponse]

extractNode
  :: NodeLike a
  => Text -> GraphGetResponse -> a
```

Чтение из базы

```
gres <- run pipe $
  makeRequest @GetRequest [] anakinGraph

forM_ gres $ \graph -> do
  let Movie movieName = extractNode "movie" graph
  putStrLn $ unpack movieName
```

Задача 3

- Написать тот же запрос про Люка с помощью поиска графов

koltsov@biocad.ru
mirzoev@biocad.ru

@maksbotan
@nolanrus

Спасибо за внимание!

https://github.com/biocad/career