neo4j 高级应用技术专题系列

# APOC Java存储过程库

## 实现复杂和高性能的图遍历

Fanghua(Joshua) Yu
Field Engineering, APAC.
joshua.yu@neotechnology.com
https://www.linkedin.com/in/joshuayu/

俞方桦, 博士
2730625048, QQ群： Neo4j中文社区
Neo4j中文社区：neo4j.com.cn, 用户: GraphWay

By Fanghua Yu, Dec. 2017

# 1、概述

# APOC是什么东东？

从Neo4j 3.0开始，引入了用户定义的存储过程这一概念。简单地说，存储过程：

- ✓ 用Java实现
- ✓ 可以在Neo4j数据库启动时加载
- ✓ 可以方便地在Cypher中调用
- ✓ 实现用Cypher很难实现的任何功能

非常类似于关系数据库中的存储过程概念。


告诉过你们，图数据库是俺的表亲。。。

对的，任何功能。比如说：

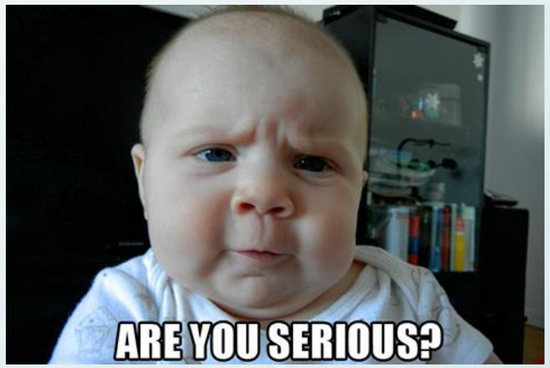
- 特殊的遍历逻辑
- Cypher不提供的函数
- 用SQL查询Neo4j (?!)
- 在Cypher里面访问微信朋友圈
- 。。。 。。。

# 那么，APOC到底是什么东东？

APOC是一个包含了300+强大函数和过程的用户存储过程库。

APOC的名字最早来自2009年的Neo4j，代表“A Package Of Components” – 组件包。是的，一点也不酷。

后来，变成“Awesome Procedures On Cypher” – 超级棒的Cypher存储过程。有点自卖自夸的感觉吧？



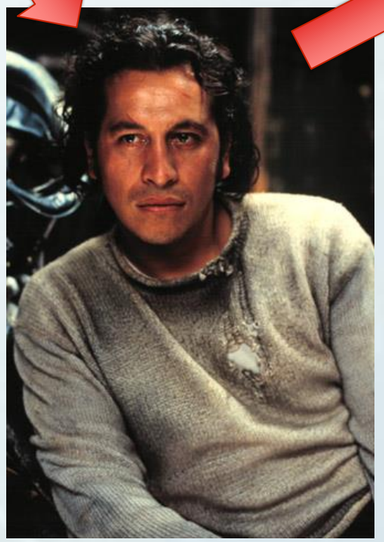
# APOC的真正含义…

如果你也和Neo4j的创始人一样是《黑客帝国》(Matrix)的粉丝，你应该想道了…


Neo是这位帅哥+人类英雄

Cypher是这个背叛人类投靠机器世界的家伙(看上去也不像什么好人)。

Apoc是他们的(程序员)队友，后来被Cypher谋杀。



是不是很像咱们的偶像大叔？

By Fanghua Yu, Dec. 2017

好了，不说八卦了，来点干货。

# 安装APOC(其实非常简单)

1、github: https://github.com/neo4j-contrib/neo4j-apoc-procedures/
2、下载JAR文件；
3、复制到Neo4j的plugins目录下；
4、(可选)某些情况下需要修改neo4j.con中的配置；
5、重启动Neo4j服务；
6、在Neo4j浏览器中，输入：

```
RETURN apoc.version()
```

apoc和Neo4j版本对应

| apoc版本 | Neo4j版本 |
|---|---|
| 3.3.0.1 | 3.3.x |
| 3.2.3.5 | 3.2.x |
| 3.2.0.4 | 3.2.2 |
| 3.2.0.3 | 3.2.0 |
| 3.1.3.9 | 3.1.x |
| … | … |

# 这就好了？是不是少了些什么？

要注册吗？NO！

要付费吗？NO！


要激活吗？NO！

要告诉Neo4j吗？
MAYBE, ONLY IF YOU DON’T LIKE IT.

要告诉老板吗？你自己看着办！

# APOC装好了，然后就下班了？

先不急，让我们来看看里面到底有什么东东。。。

注：

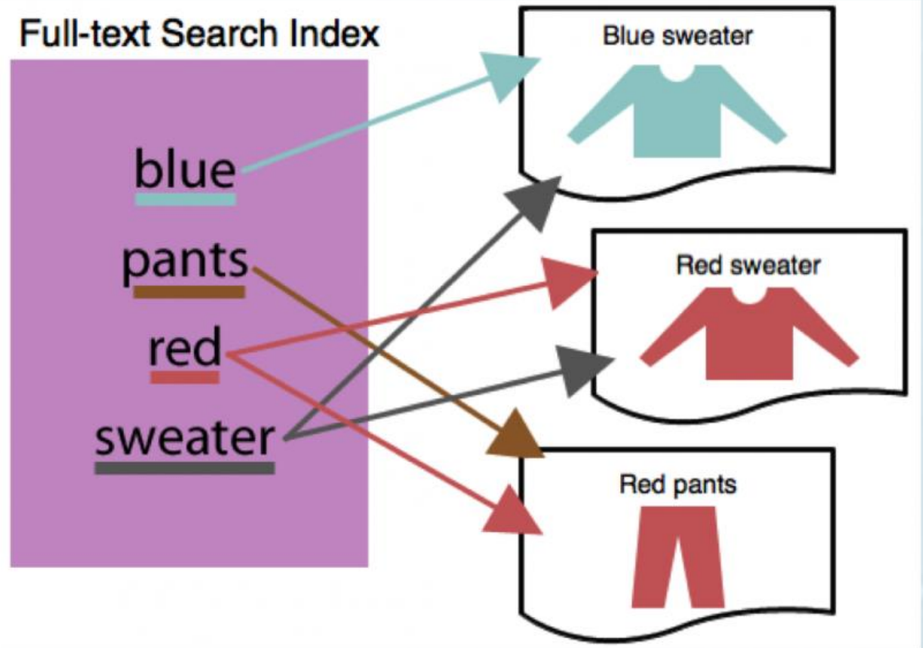
- 本材料中使用的是版本3.3.0.1的apoc，适用于Neo4j 3.3.x版本。
- 在线文档：https://neo4j-contrib.github.io/neo4j-apoc-procedures/
- 最后更新日期：2017年12月

# APOC: 文本和查找索引

CALL apoc.index.*

索引？！不是说图数据库不需要索引的吗？需要吗？不需要吗？

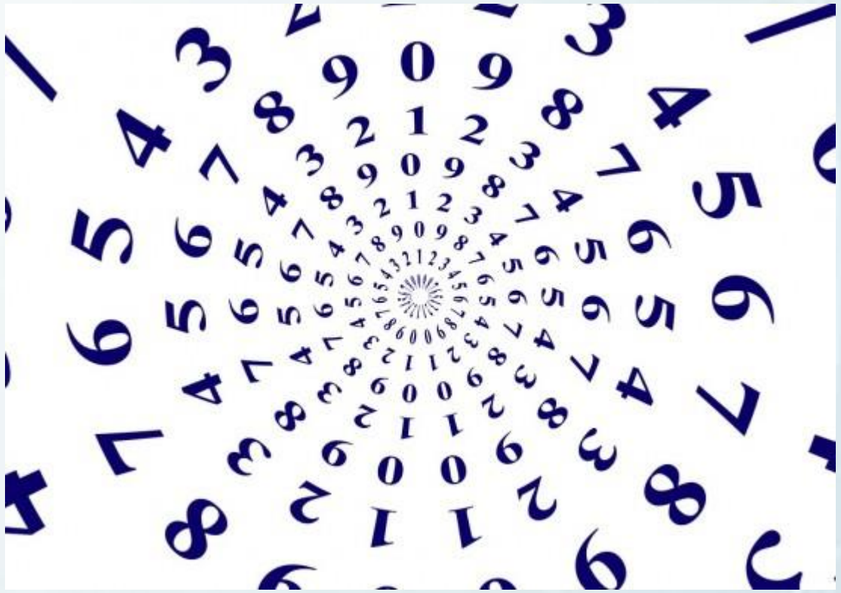
- Neo4j使用 Apache的Lucene库来进行文本处理
- 文本索引过程用来对属性的文本内容进行自然语言处理并创建索引
- 支持快速的对节点和关系属性值的全文本查询
- 手工索引方式，需要随数据更新而定期更新
- 如果加载中文分词库，也能够实现中文文本的索引



# APOC: 功能函数

CALL apoc.text.* / date.* / number.*

- 字符串处理
- 时间戳
- 数字类型及其格式
- 日期
- 大数/科学计数法

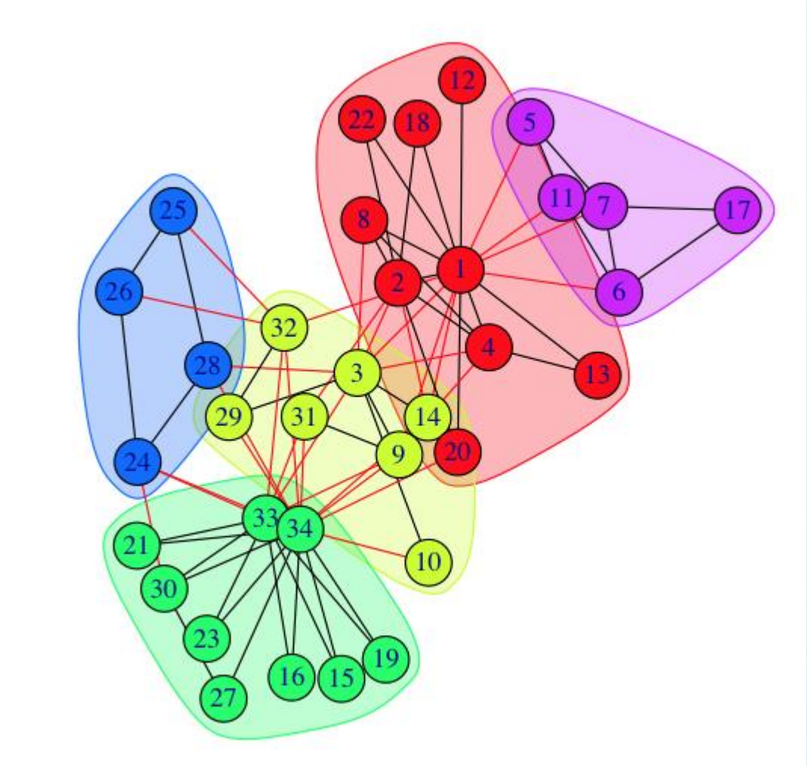


# APOC: 图论算法(1)

CALL apoc.algo.community()

- 社区检测/社团划分(Community Detection)
  - 标签传播(Label Propagation)
  - 可自定义的迭代层数和权重
  - 对网络实施分区(Partition)

```
CALL
apoc.algo.community(25,null,'partition','X',
'OUTGOING','weight',10000)
```

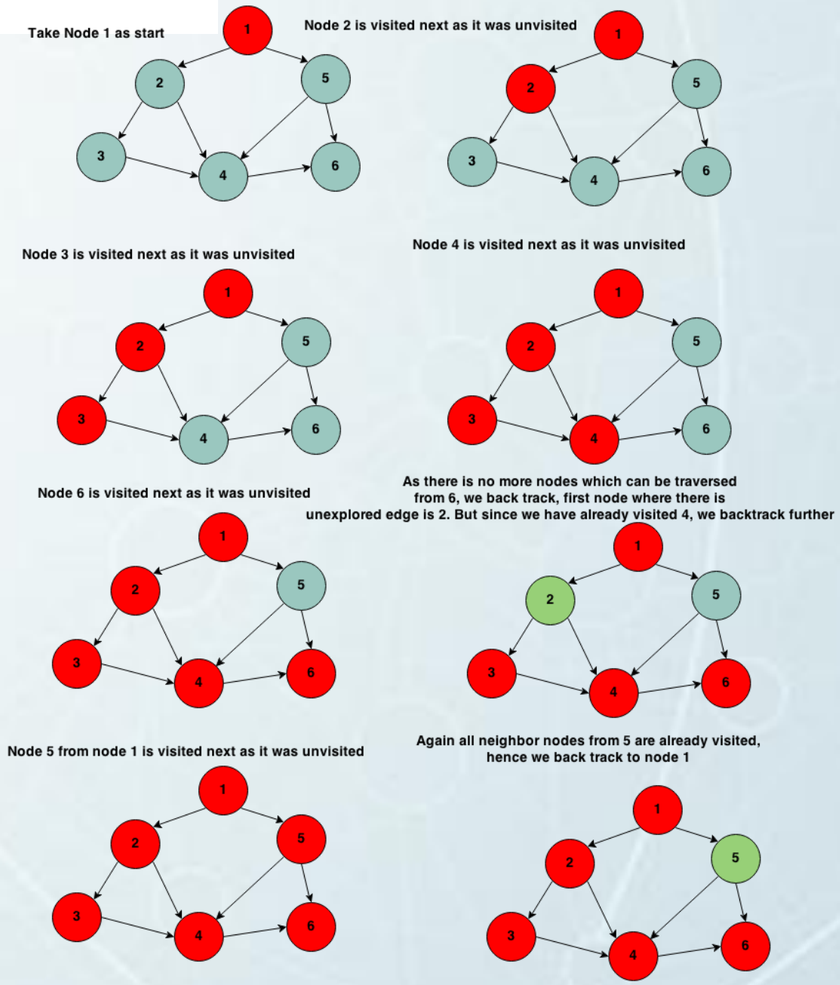


# APOC: 图论算法(2)

CALL apoc.path.*

❖ 路径扩展 / 图的遍历

- 宽度优先 vs 无重复的关系路径
- 可自定义遍历规则：起始节点，层级，包含关系及方向等等
- 按照节点类型进行过滤：排除(blacklist)、终止 (termination)、结束(end)、包含 (whitelist)
- 最大节点/关系数限制
- 子图遍历
- 生成树(spanning tree)遍历

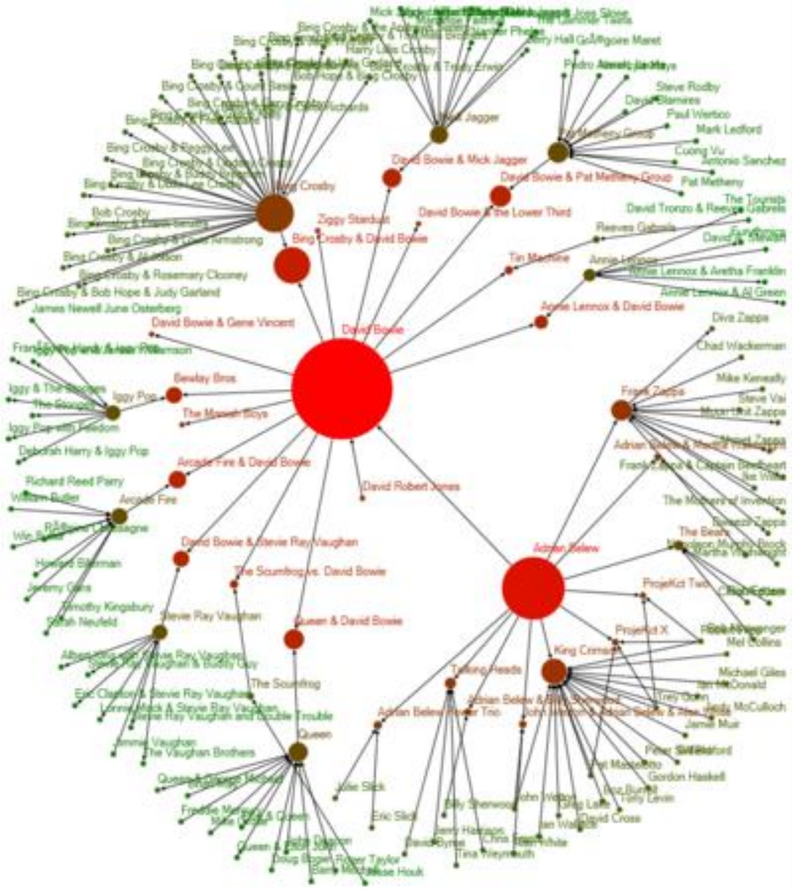



By Fanghua Yu, Dec. 2017

# APOC: 图论算法(3)

CALL apoc.algo.closeness() / betweenness()

- 中心性(Centrality algorithm)
  - 紧密中心性(Closeness Centrality)
  - 间接中心性(Betweenness Centrality)
- 计算节点在网络中处于核心地位的程度
- 发现社交网络中的重要人物
- 发现欺诈团伙中的核心/老大

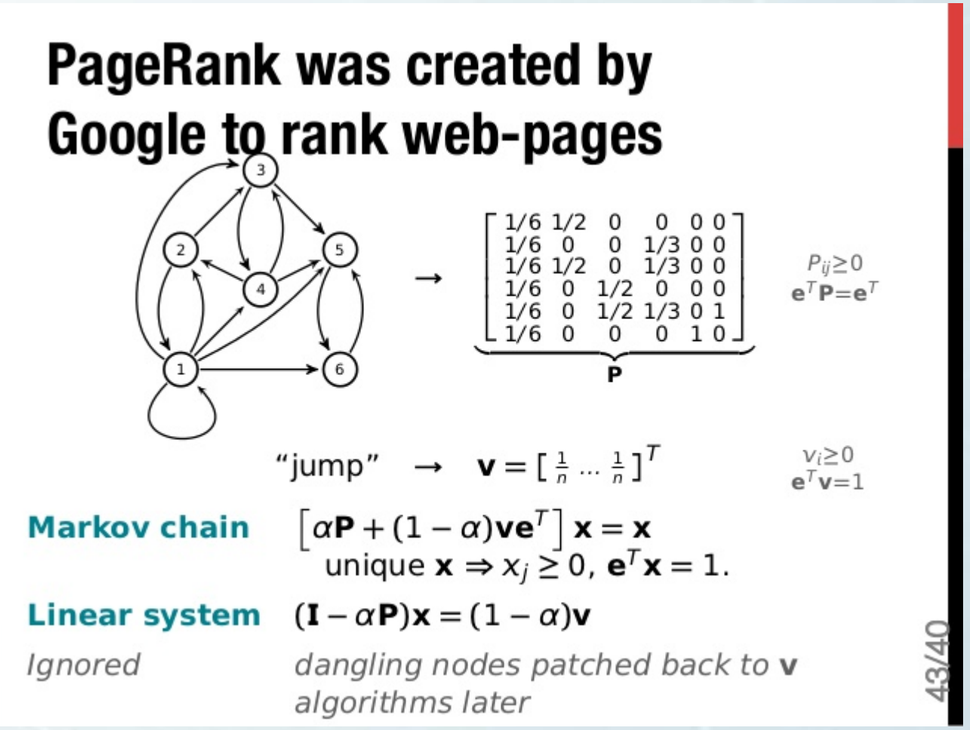


# APOC: 图论算法(4)

CALL apoc.algo.pageRank()

- 页面排行(Page Rank)
  - 用来计算节点在整个网络中的重要性
  - 可以指定参与计算的节点
  - 目前计算是基于全网中的所有指定节点

```
CALL apoc.algo.pageRank(nodes) YIELD node,score
```



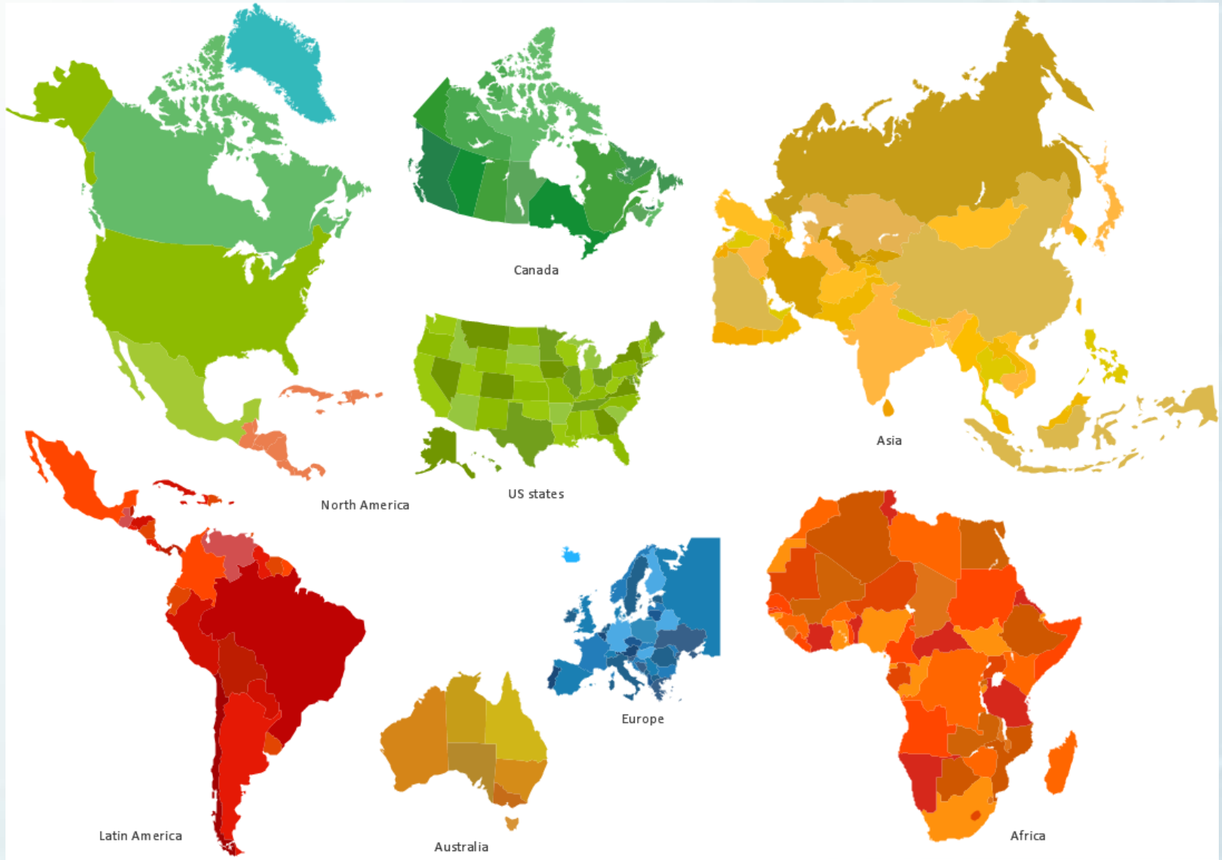

By Fanghua Yu, Dec. 2017

# APOC: 地理空间函数

CALL apoc.spatial.*

- 根据地址返回地球坐标
- 计算直线距离
- 按照距离远近排序节点

```
CALL
apoc.spatial.feocodeOnce(node.address)
YIELD location
```

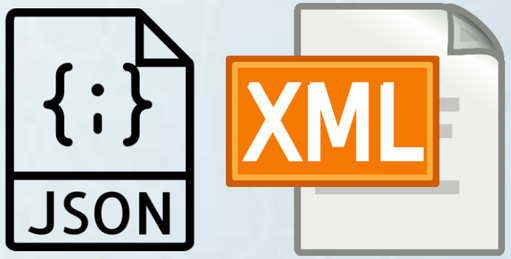


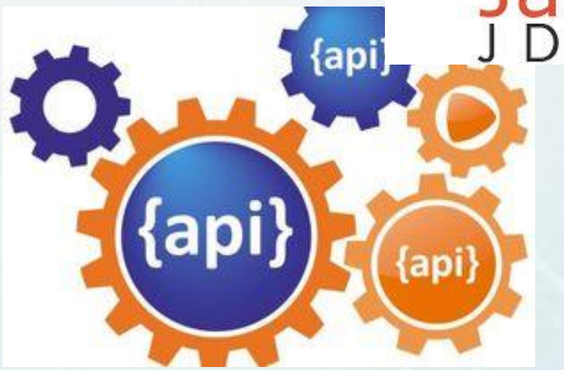

By Fanghua Yu, Dec. 2017

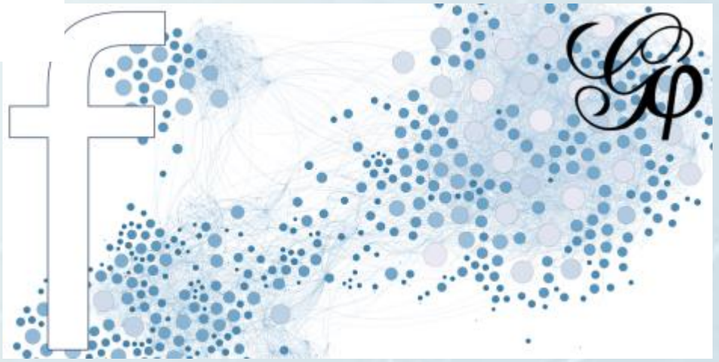
# APOC: 数据集成

CALL apoc.load.*

- 加载JSON数据：调用RESTful API
- 加载关系数据库数据：通过JDBC
- 流式化数据到Gephi
- 集成 Elastic Search
- 加载XML文档

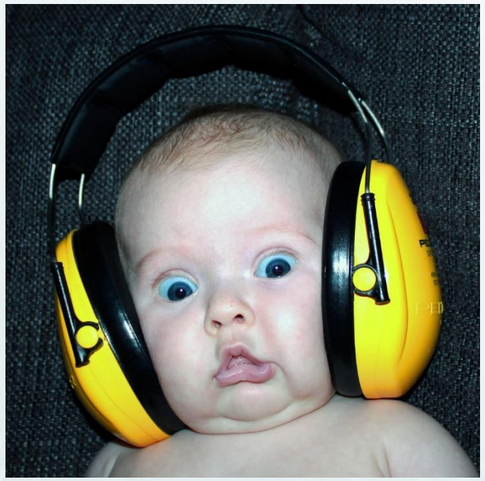
# APOC: Cypher查询

CALL apoc.cypher.*

对的！你可以在Cypher里面调用apoc过程，然后在过程里面使用Cypher查询。

*&#*J%><@P(!5+$(*M<>?!

很搞脑子吧？



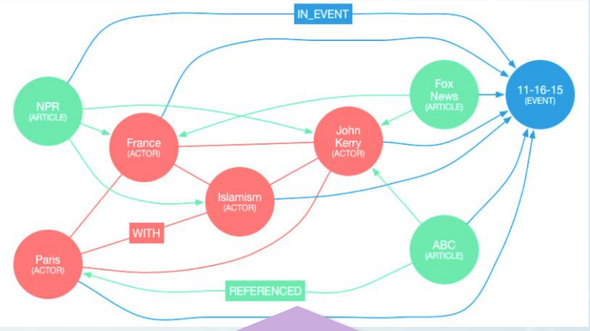
使用apoc来执行Cypher查询的好处：

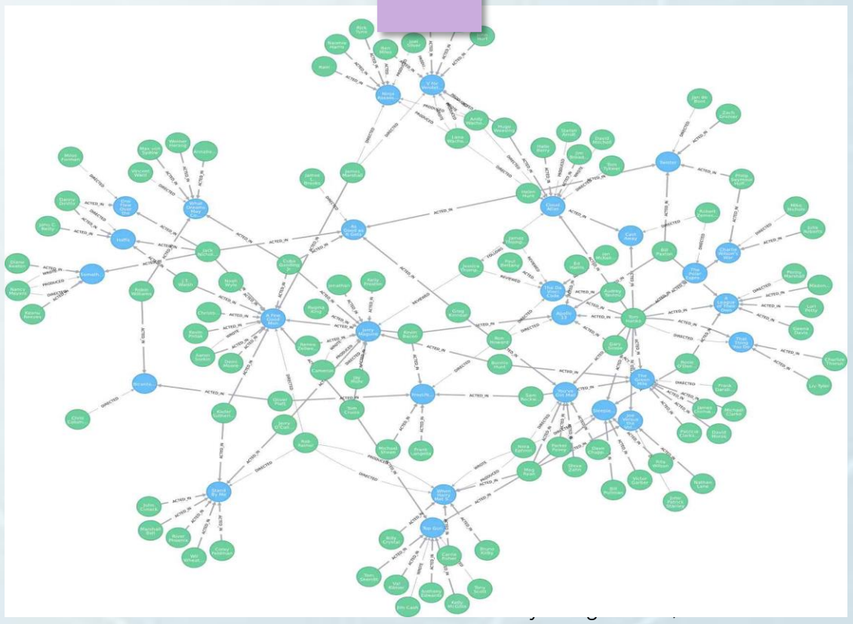
- ✓ 可以动态构造查询语句
- ✓ 控制查询的执行时间
- ✓ 条件化查询分支：when, case
- ✓ 更灵活的查询执行任务控制：批次大小，并行执行，重试等等

# APOC: 虚拟图

CALL apoc.create.*

- apoc支持创建虚拟(Virtual)的节点和关系，从而构成虚拟路径和子图；
- 虚拟图类似关系数据库中视图(View)的概念：它们可以被查询并返回数据，但是并不物理地存储在数据库中；
- 虚拟图使某些查询更加灵活和高效：
  - 创建数据库中并不存在的节点和关系
  - 缩小查询的相关子图规模
  - 控制遍历的路径
- 虚拟节点和关系的ID都是负数
- 内存管理



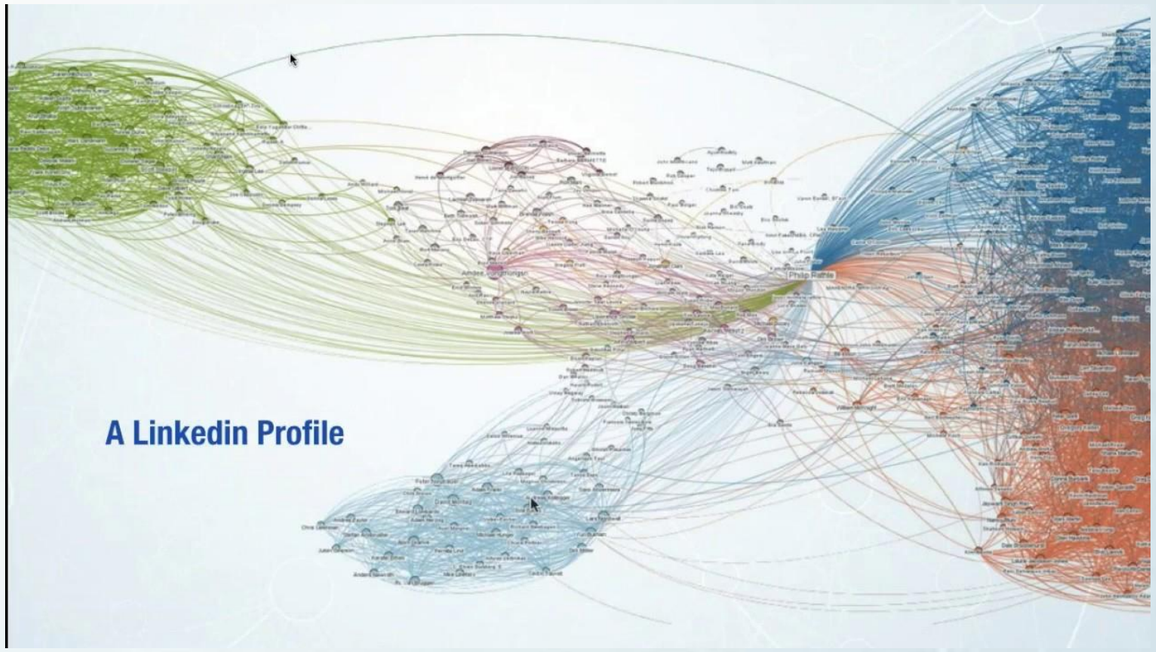



# APOC: 重构/优化图

CALL apoc.refactoring.*

❖ 对已有的图进行转换操作以实现重构(Refactoring)，包括：

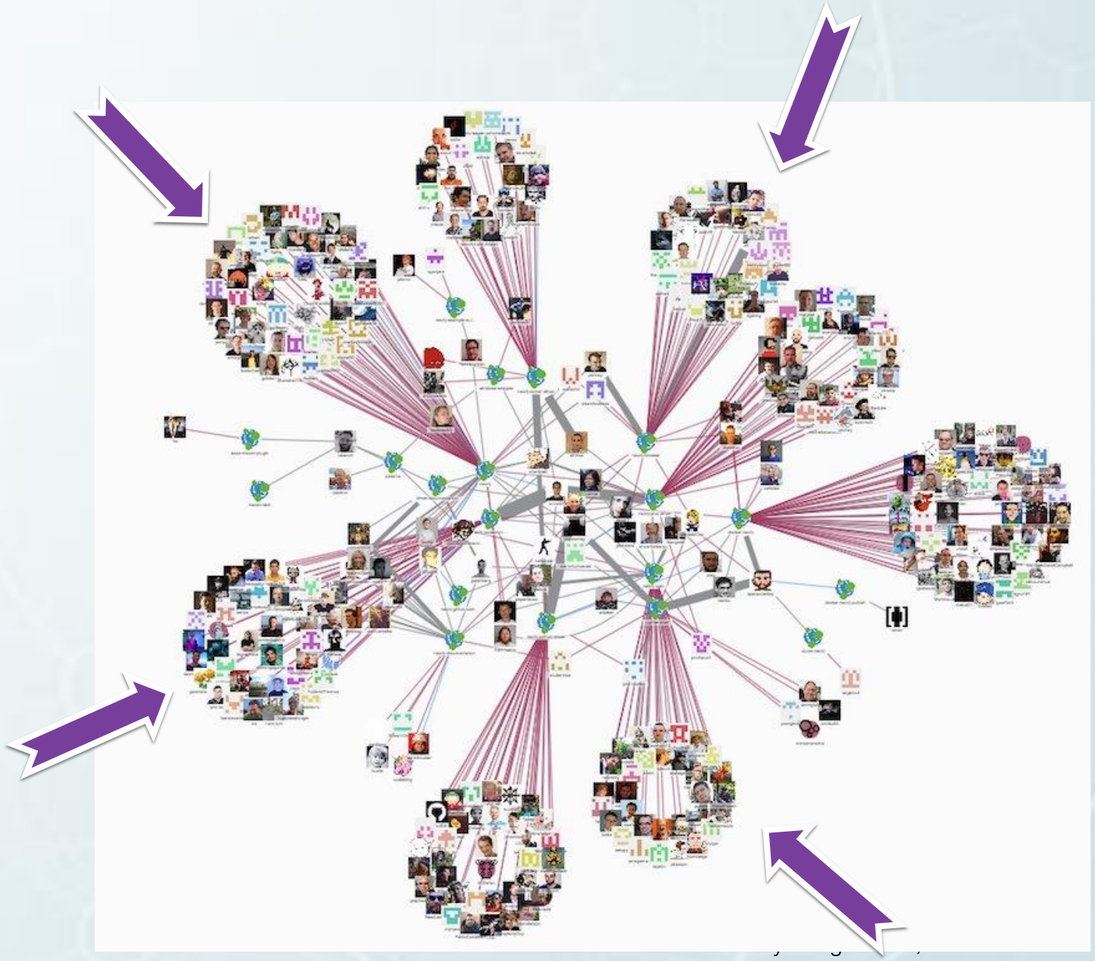
- 复制节点及其属性，包括/不包括关系
- 合并节点
- 重建关系到新的节点
- 改变关系类型
- 将关系转换成节点
- 将节点转换成关系
- 将属性转换成分类节点，并与相关的节点建立关系



# APOC: 并行节点查询

CALL apoc.search.*

- 在可能的情况下并行查找结点
- 结果可以是全部匹配节点，或者去除重复后的节点
- 可以使用JSON格式定义要查询节点的属性集
- 支持多种匹配类型："<", ">", "=", "<>", "⇐", ">=", "=~"



# APOC: 其他数据库特性

- 触发器(Trigger)
- 写入锁(Write lock)
- 显示数据库元模型(metadata)
- 数据轮廓(Data profiling)
- 管理索引和限制
- 对节点和关系并发操作的支持：原子性

# APOC: 展望

自从问世以来，几年间APOC迅速发展、扩充，不断丰富功能。从起初的一个社区开发项目，到Neo4j 3.3发布时，它已经成为官方版本中包含的存储过程库。未来，更多、更丰富的特性会不断添加进来。已经在开发的图遍历算法将增加：

- ✓ Dijkstra 遍历
- ✓ A* 遍历
- ✓ allSimplePaths
- ✓ 对单个节点的图算法
- ✓ 最大（权）团(Cliques)搜索
- ✓ 余弦相似度(Cosine similarity)
- ✓ 欧氏距离和相似度(Euclidean distance and similarity)

# 感谢阅读！

## 欢迎提出问题、意见和建议。

Neo4j中文社区： http://neo4j.com.cn
QQ群：Neo4j中文社区 / 547190638
个人QQ号：Neo4j-APAC技术支持 / 2730625048