Neo4j use case:

Description of the Use Case:

Organization Hierarchy – The typical hierarchical view of the org chart of ABC Corp is shown below.

In this org chart, each worker reports to “a” supervisor, and holds (/assigned to) “one” position & “one” cost center. The **relational** database designed for this requirement works great. [Assignment3/4]

However, say, if the organization's requirement is to have some workers in multiple positions, and possibility of a worker reporting to more than 1 supervisor at a given point in time (say, a trainee manager double stacking in a given position etc.), the current relational model need to be modified. Also, changing the hierarchy when there is a change in worker's position also gets tedious. Let's design the above hierarchy in graphical database's nodes and relationships.

Nodes:

Worker

hiredate	1995-10-05
emplid	E1001
firstname	JOHN
userid	JMARK01
lastname	MARK
gender	Male

Position

positionid	1
title	President & CEO

Cost Center

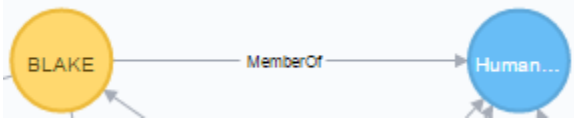
costcenterid	100
costcentername	ExecutiveMgmt
businessunit	ADM

Relationships:

AssignedTo – Worker is assigned to 1 or more positions

IS607 Project 5

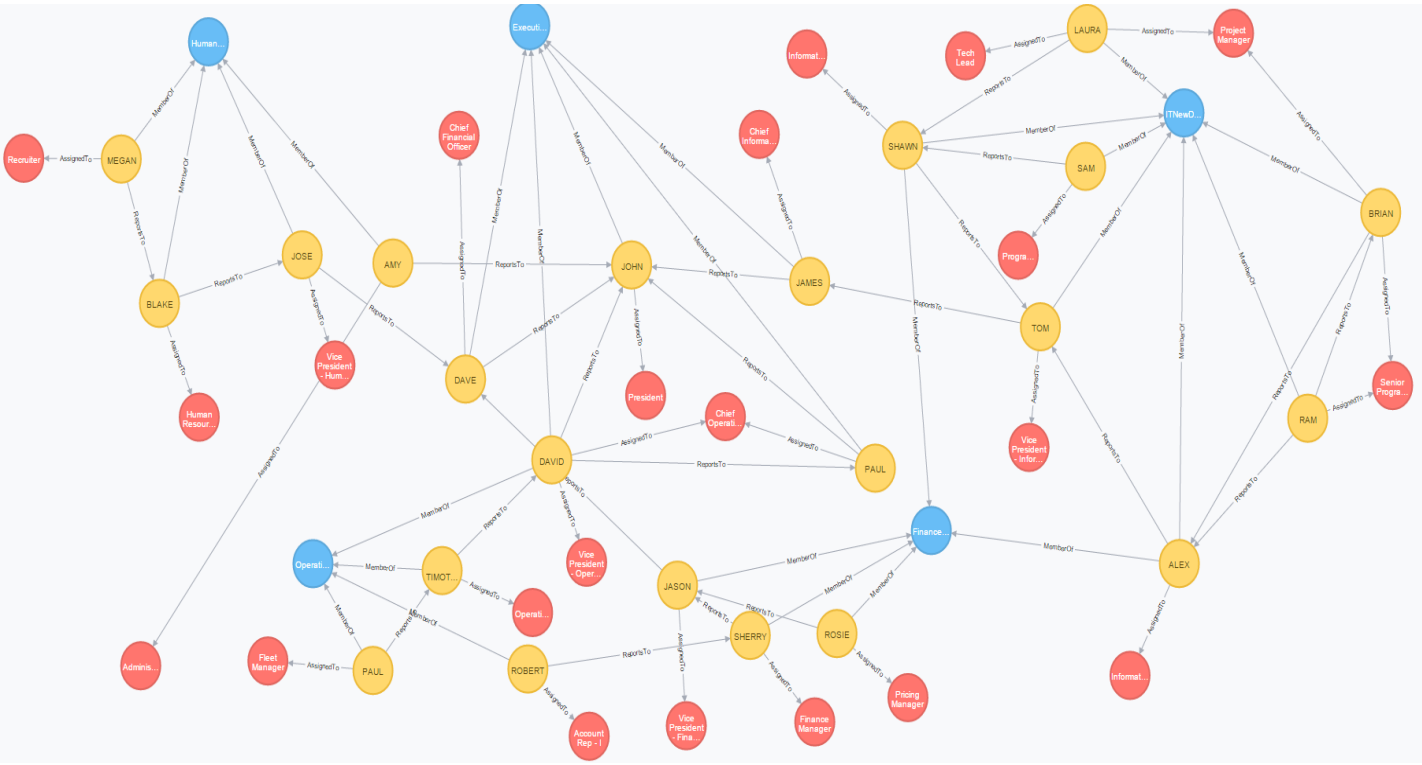
MemberOf – Worker is a member of 1 or more Cost Centers



ReportsTo – Worker reports to 1 or more supervisors.

The overall graph appears like below, once we have defined all Nodes and Relationships:

```
match(n) return(n)
```



Data Acquisition/Loading of sample data

Load Cost Centers

load csv with headers from "file:C:/Users/Suman/Dropbox/Sem-I/Prog/github/DAMgmt/IS607 Project 5/data-costcenters.csv" as costcenters create (a:CostCenter {costcenterid:costcenters.costcenterid, costcentername: costcenters.costcentername, businessunit: costcenters.businessunit })

Load Positions

load csv with headers from "file:C:/Users/Suman/Dropbox/Sem-I/Prog/github/DAMgmt/IS607 Project 5/data-positions.csv" as positions create (a:Position {positionid:positions.positionid, title: positions.title })

Load Workers

load csv with headers from "file:C:/Users/Suman/Dropbox/Sem-I/Prog/github/DAMgmt/IS607 Project 5/data-workers.csv" as workers create (a:Worker {emplid:workers.emplid, userid: workers.userid, firstname: workers.firstname, lastname: workers.lastname, gender: workers.gender, hiredate: workers.hiredate })

Establish relationship between Worker <--- AssignedTo ---> Position(s)

load csv with headers from "file:C:/Users/Suman/Dropbox/Sem-I/Prog/github/DAMgmt/IS607 Project 5/data-workers-positions.csv" as workerposns match (a: Worker {emplid: workerposns.emplid}), (b: Position {positionid: workerposns.positionid}) create (a) - [r:AssignedTo {positionid: workerposns.positionid}] -> (b)

Establish relationship between Worker <--- MemberOf ---> CostCenter

load csv with headers from "file:C:/Users/Suman/Dropbox/Sem-I/Prog/github/DAMgmt/IS607 Project 5/data-worker-costcenters.csv" as workerccs match (a: Worker {emplid: workerccs.emplid}), (b: CostCenter {costcenterid: workerccs.costcenterid}) create (a) - [r:MemberOf {costcenterid: workerccs.costcenterid}] -> (b)

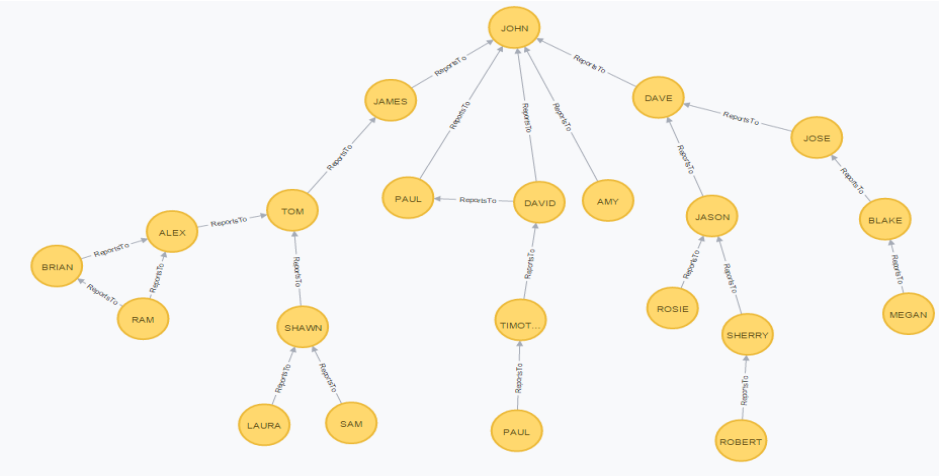
Establish relationship between Worker <--- ReportsTo ---> Supervisors

load csv with headers from "file:C:/Users/Suman/Dropbox/Sem-I/Prog/github/DAMgmt/IS607 Project 5/data-workers-supervisors.csv" as supervisors match (a: Worker {emplid: supervisors.emplid}), (b: Worker {emplid: supervisors.supervisorid}) create (a) - [r:ReportsTo {supervisorid: supervisors.supervisorid}] -> (b)

Data Analysis

Find Worker Nodes

match (a:Worker) return(a)



Notice that RAM now reports to both BRIAN and ALEX. Similarly, DAVID reports to JOHN and PAUL.

Find Position Nodes

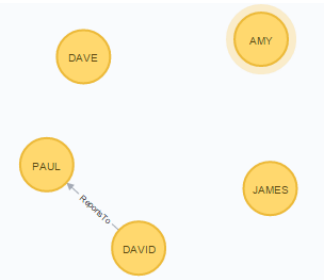
match (b:Position) return(b)

Find Cost Center Nodes

match (c:CostCenter) return(c)

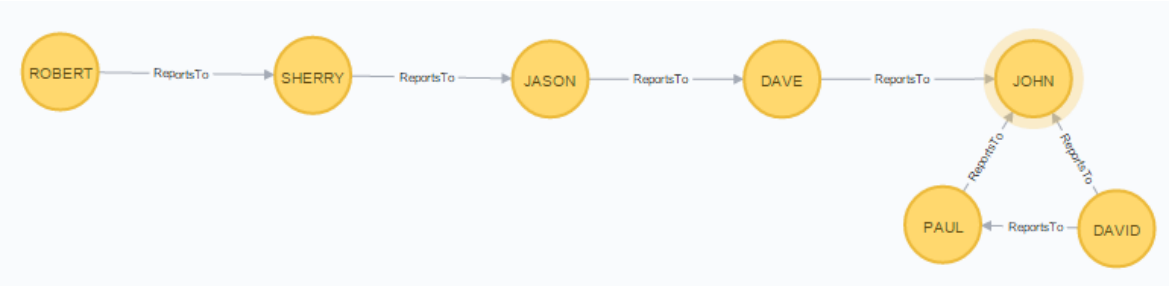
Find All workers that directly reports to John Mark (President & CEO , emplid:E1001)

match (a:Worker)-[r:ReportsTo]->(b:Worker{emplid: 'E1001'}) RETURN a



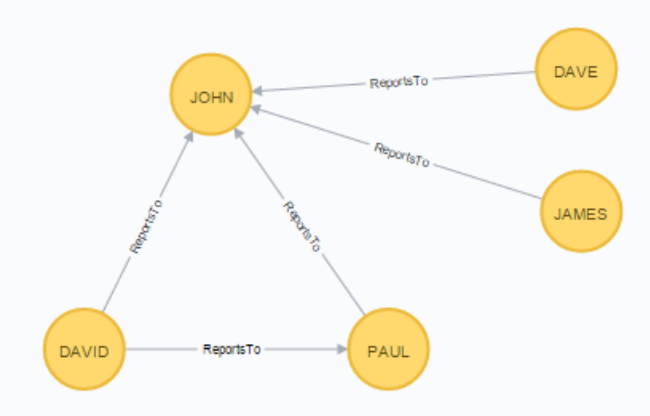
Find the hierarchy of worker [E2200,ROBERT, WELCH] [bottom-up]

*match (a:Worker {emplid: 'E2200'}), (b:Worker {emplid: 'E1001'}), hierarchy = ((a)-[:ReportsTo*0..]-(b)) RETURN hierarchy*



Find the executives (memeber of ExecutiveMgmt cost center)

match (a:Worker) - [cc:MemberOf] -> (c:CostCenter) where c.costcenterid = '100' return a



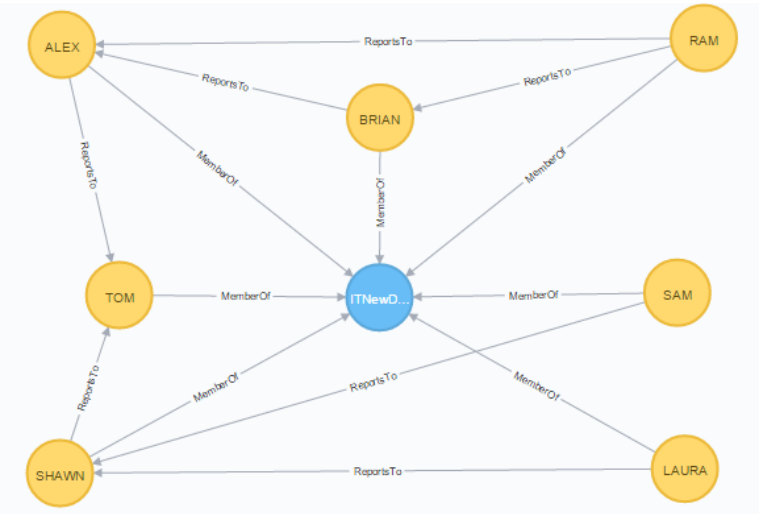
Find the cost center that has got max number of workers.

match (a:Worker) - [r:MemberOf] -> (c:CostCenter)
return c.costcentername, count(r.costcenterid) as memberCount
order by memberCount desc
limit 1

c.costcentername	memberCount
ITNewDev	7

Look up the Cost Center ITNewDev

*match (a:Worker) - [r:MemberOf] -> (c:CostCenter {costcenterid:'104'}) return **



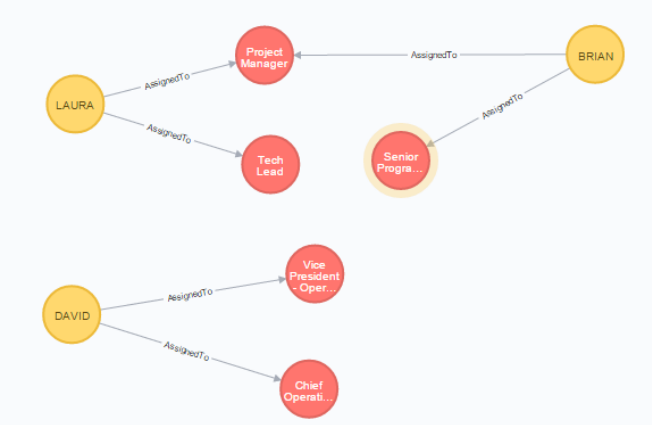
IS607 Project 5

*Find **all** workers with more than one position assigned*

```
match (a:Worker) - [r:AssignedTo] -> (b:Position)
with a.emplid as emplid, count(r.positionid) as positionCount
where positionCount > 1
return emplid, positionCount
```

emplid	positionCount
E3001	2
E1004	2
E2901	2

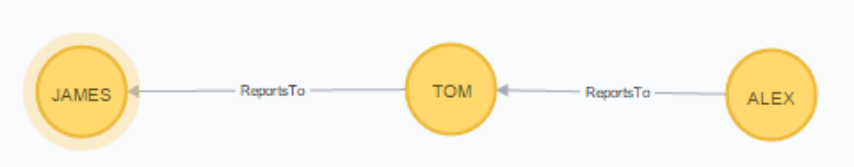
```
match (a:Worker)- [r:AssignedTo] -> ()  where a.emplid IN ['E3001','E1004','E2901']  return *
```



Modify hierarchy, to make the worker [Alex Miller, E2018] directly reporting to CIO [Jay Hudson, E2001]

The current reporting hierarchy of James is

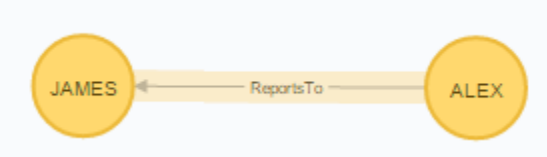
```
MATCH (a: Worker {emplid: 'E2018'}) - [r:ReportsTo] -> (b: Worker), (c : Worker {emplid: 'E2001'})
return r,c
```



IS607 Project 5

MATCH (a: Worker {emplid: 'E2018'}) - [r:ReportsTo] -> (b: Worker), (c : Worker {emplid: 'E2001'}) create (a) - [r2:ReportsTo{supervisorid: c.emplid}] -> (c) DELETE r

After the relationship change:



With the above change, the ITNewDev cost center (non-executives) looks like below:

*match (a:Worker) - [r:MemberOf] -> (c:CostCenter {costcenterid:'104'}) return **