

Installation and Setup Cheat Sheet (Windows)

Java Environment Variables for Windows

JAVA\_HOME C:\Program Files\Java\jre-10.0.1

PATH ;C:\Program Files\Java\jre-10.0.1\bin\server

Download

neo4j-community-3.4.0-windows.zip

Create Operating Folder

C:\graph

Copy in Zip contents

Start Server from CMD Run as Administrator

cd /.

cd graph/bin

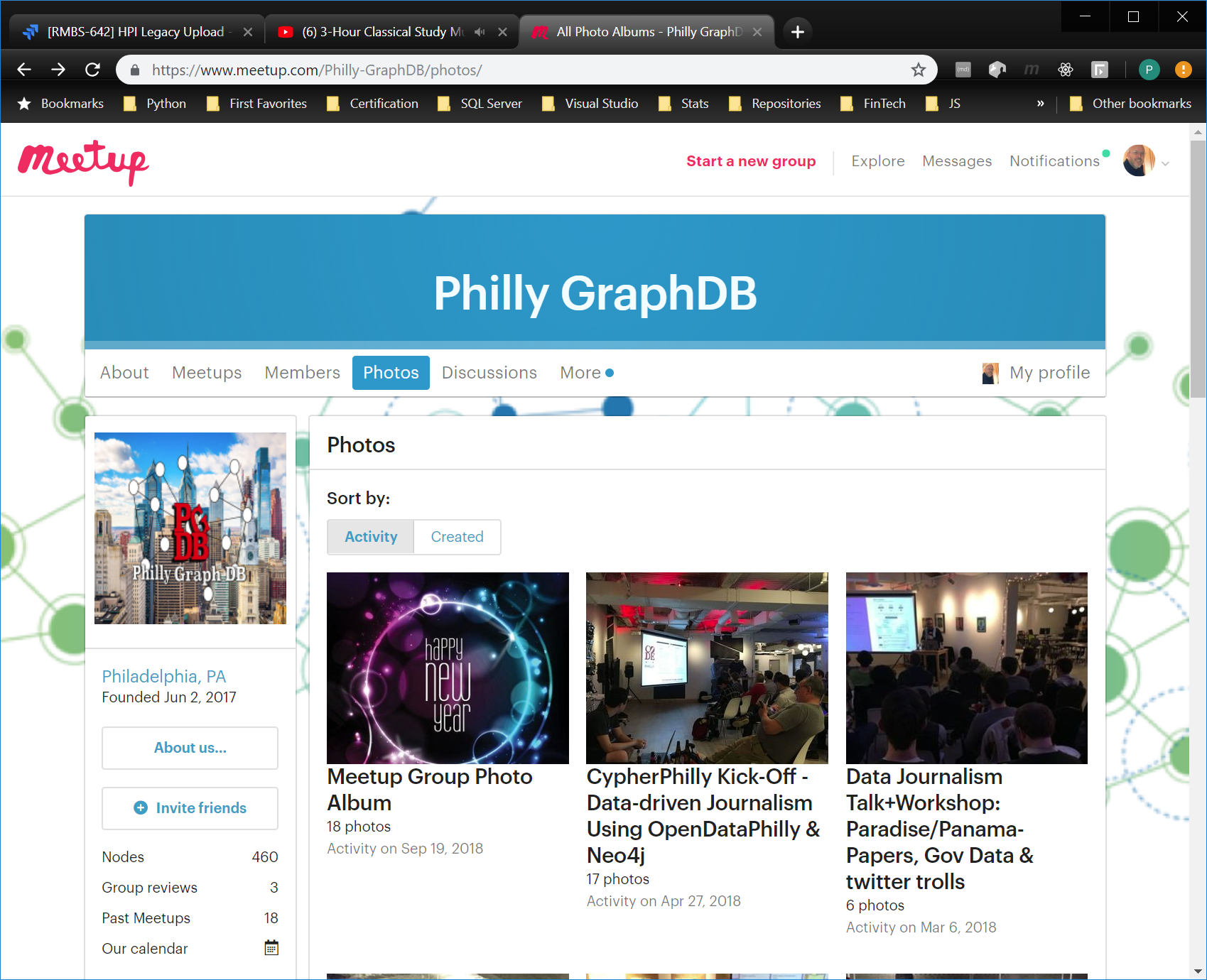
neo4j console

Open Browser

localhost:7474

Username: neo4j

Password: neo4j



//Add a deal node (edit and repeat for deal 2 and 3)  
CREATE (n:Deal {dealId:"1", dealName:"SQ101"})

//Add loan nodes from CSV Note: file must be in %Neo4j\_Home%/Import  
LOAD CSV WITH HEADERS FROM "file:///C:Loan.csv"AS row  
CREATE (n:Loan) SET  
n.loanId = row.LoanId, n.dealId = row.DealId, n.maturityDate = row.MaturityDate,  
n.originalAmount = toInteger(row.OriginalAmount)

//Add loanPeriod nodes from CSV  
LOAD CSV WITH HEADERS FROM "file:///C:LoanPeriod.csv" AS row  
CREATE (n:LoanPeriod) SET  
n.periodId = row.PeriodId, n.loanId = row.LoanId, n.Balance = toFloat(row.Balance),   
n.delinquencyStatus = toInteger(row.DelinquencyStatus)

//Add Deal:Loan relationships  
MATCH (d:Deal), (l:Loan) WHERE d.dealId = l.dealId CREATE (d)-[:CONTAINS]->(l)

//Add Loan:LoanPeriod relationships  
MATCH (l:Loan), (p:LoanPeriod) WHERE l.loanId = p.loanId CREATE (l)-[:HAS\_BALANCE]->(p)

//Query everything  
MATCH(s) RETURN s

//Update a loan property  
MATCH (n { loanId: '2' }) SET n.maturityDate = '2037-01-01' RETURN n

//Update loan add property  
MATCH (n { loanId: '2' }) SET n += {rating: 'AAA'} RETURN n

//Returns table of matches  
MATCH (d:Deal)-[:CONTAINS]->(l:Loan) RETURN d.dealName, l.loanId ORDER BY d.dealName , l.loanId

//Returns table of nonmatches  
MATCH (d:Deal) WHERE NOT (d:Deal)-[:CONTAINS]->() RETURN d.dealName

//Returns Average balance of Deals in Period 2  
MATCH (d:Deal)-[:CONTAINS]->(l:Loan)-[:HAS\_BALANCE]->(p:LoanPeriod) WHERE p.periodId = "2"   
RETURN d.dealName AS `Deal`, AVG(p.loanBalance) AS `Average Balance`

//Returns list of loans with change in delinquencyStatus from Period 1 to Period2  
MATCH (p1:LoanPeriod), (p2:LoanPeriod)  
WHERE p1.loanId = p2.loanId  
AND p1.periodId = "1"  
AND p2.periodId = "2"  
AND p1.delinquencyStatus <> p2.delinquencyStatus  
RETURN p2.loanId  
ORDER BY p2.loanId

//Delete relationships  
MATCH ()-[r:CONTAINS]->() DELETE r

//Delete all nodes  
MATCH(s) DELETE s