Getting Started with *Neo4j*

As of January 2020

Jeff Tallman
jeffrey.tallman@neo4j.com

neo4j

Agenda

Installing Neo4j (aka Infrastructure)

Causal Clustering (aka High Availability)

Backups & Monitoring (aka Operations)

Loading Data into Neo4j (aka Integration)

Use Cases for Neo4j (aka Applicability)

Modeling for Neo4j (aka Design)

Moving to Cypher (aka Porting)

Caveat

This is not a full class on any one of these topics

The purpose of this presentation is to give you an introduction to the technology, techniques and capabilities of Neo4j that are common areas of focus for new customers.

There are training classes that separately address many of these topics

Installing Neo4j

neo4j

Installation Requirements

Supported Platforms

- MS Windows
- Most Linux distros (RHEL, SUSE, Debian, Ubuntu, CentOS, …)
- Apple MacIntosh
- VM's, Docker, Public Cloud (AWS, GCP, Azure), DBaaS (Neo4j Aura)

What is needed from OS

- A few ports opened (4 for single instance; 7 for clustering)
- A high number of allowed open files (linux kernel settings/limits.conf)
- A JRE – can be OpenJRE/OpenJDK
  - ✓ *JDK recommended for compiling procs/UDF's*
  - ✓ *Desktop versions ship with JRE – Enterprise versions typically don't*
  - ✓ *Neo4j 3.5 → JRE/JDK 1.8*
  - ✓ *Neo4j 4.0 → JRE/JDK 11, 12+*
- A fast filesystem (Neo4j uses standard OS files for storage)

See Neo4j Operations Manual for Install Instructions (especially Docker)

Sizing

It all depends

- See item #1
- See item #1

If you know nothing….

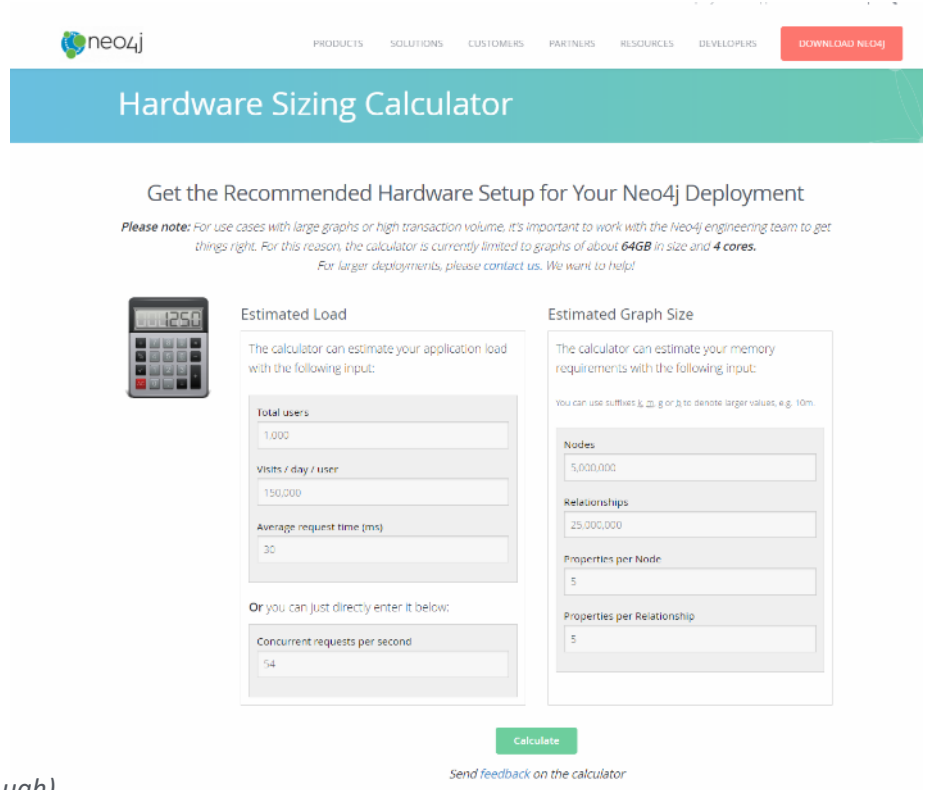
- Assume 1 core per 10 concurrent users
  - ✓ *Based on 100ms SLA and 10ms query exec time*
- Assume 4-16GB of RAM per core

If you know # of nodes, etc. …..

- Neo4j sizing calculator
  - ✓ *Just Google search for it*

If you'd rather do the math….

- Nodes occupy 15 bytes of space
  - ✓ *Labels are stored separately and use minimal space*
- Relationships occupy 31 bytes of space
  - ✓ *Relationship Type names are stored separately*
- Properties occupy 41 bytes of space
  - ✓ *Plus space for each property value (e.g Strings…text ..ugh)*



Sizing & Memory

Page Cache

- Cache of data (nodes & relationships & properties)
- Try to have as much memory as database size
  - ✓ *100% - perfect*
  - ✓ *80% - optimal*
  - ✓ *50% - usable*
  - ✓ *<50% - uh oh*

Heap Memory

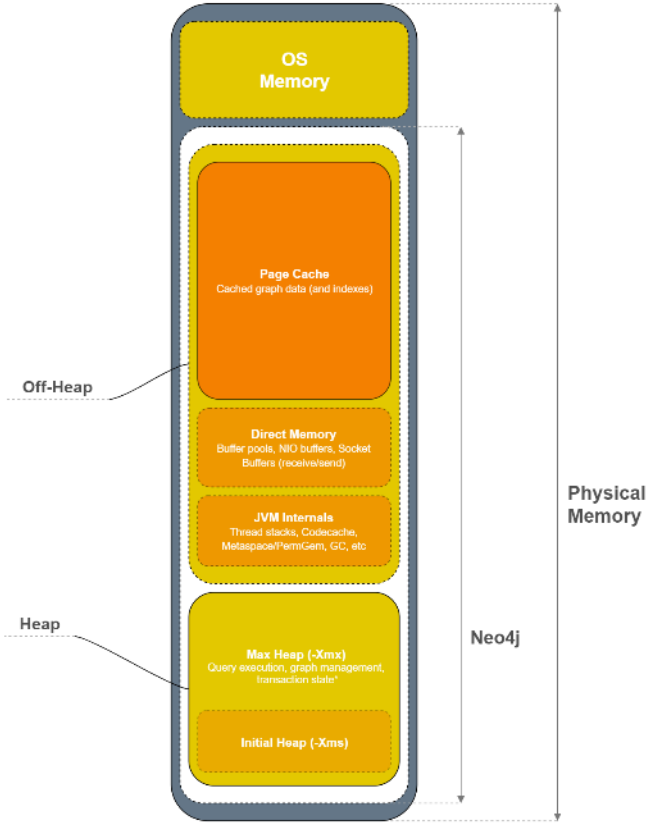
- Caches results for users, etc.
- Probably need at least 2-4x page cache

File System Buffer/Page Cache

- Probably uses ~20%+ of physical RAM
- So don't allow Neo4j to use more than 70% of physical RAM

Other OS processes

- Add 1-2GB



Linux tips

Use XFS filesystem vs. ext4

- Ext4 is a lot slower – likely due to journaling being slower than xfs journaling

Use RHEL 8 vs. RHEL 7 (kernels 3.16+)

- Adds multi-queue block driver for SSD support, etc.
- Defaults to none for IO Scheduler…which bypasses IO scheduler (yay!)

Add Neo4j ports to firewall

- http, https, bolt, backup ports – plus raft ports for cluster (3)

Tune OS

- Increase file limits (kernel & limits.conf)
- Increase networking memory (kernel) & device queue
- Change IO Scheduler to noop (RHEL 7) or none (RHEL 8)
  - ✓ *Everything else is for JBOD spinning disks (…no SAN, no SSD, no RAID)*
  - ✓ *For example, deadline*
    - sorts reads into 4 queues to increase sequential vs. random io
    - massively delays writes (assumes write cached in file system….but what about txn logs?)
- Tune file system cache flushing (kernel)

Create neo4j user

- Give sudo permissons if possible

Linux Kernel Tuning Caveats

Other than number of files…may not be necessary

- E.g. IO tuning is mainly to ensure writes are as fast as possible for low latency/high speed writes
- Network tuning is mainly a capacity issue with large numbers of users
- File system tuning is mainly to prevent neo4j & file system cache from hitting
- If none of these are a concern, then other than open files…..meh….don't sweat it.
  - ✓ *Even then, the default open files is 1M – large enough for a cluster with 3.5….*

What about read speed ?

- If IO tuning is all about writes, what about reads?
- That's what the data cache is for
  - ✓ *Ideally, we don't want to be doing a lot of physical reads*
- File system already has some support → e.g. the read-ahead

In other words

- …for most folks, the next 4 slides is just good background info

Comparing AHCI (hard disk) vs. NVMe (SSD)

AHCI vs. NVMe

- No real support for SAN/RAID with native queues
- No real support for VM (MSI-X)
  - ✓ *E.g. IO completion might result in VM swapping or delay in processing IO (until VM is scheduled)*
- Wreaks havoc via IO schedulers

AHCI Queue Depth

- Many HW controllers come with a queue depth of 254 or 255
  - ✓ `cat /sys/block/<device>/device/queue_depth`
- Unfortunately Linux comes with a request limit of ….128!!!
  - ✓ `cat /sys/block/<device>/queue/nr_requests`
- You can monitor this via iostat –x….check avgqu-sz column
  - ✓ *Make sure you monitor during peak IO activity – e.g. database dumps, bulk loads, etc.*

To change:

- `echo 1024 > /sys/block/<device>/queue/nr_requests`
- May not work on boot device/root volume
- To make this persistent – add to rc.local
- Some tests have shown a 2x performance gain just with dd

High-level comparison of AHCI and NVMe[5]

	AHCI	NVMe
Maximum queue depth	One command queue; 32 commands per queue	65535 queues;[30] 65536 commands per queue
Uncacheable register accesses (2000 cycles each)	Six per non-queued command; nine per queued command	Two per command
MSI-X and interrupt steering	A single interrupt; no steering	2048 MSI-X interrupts
Parallelism and multiple threads	Requires synchronization lock to issue a command	No locking
Efficiency for 4 KB commands	Command parameters require two serialized host DRAM fetches	Gets command parameters in one 64-byte fetch

https://en.wikipedia.org/wiki/NVM_Express

Note – will require blk-mq IO driver in Linux (kernel 3.16 or higher) to bypass IO Scheduler. Current enterprise Linux kernels are ~3.10

Linux Kernel Tuning – Not Needed…but…. (1)

```
fs.file-max = 5000000
fs.nr_open = 5000000
```

} Strongly Recommended

```
net.core.rmem_default =   4194304
net.core.rmem_max = 134217728
net.core.wmem_default =   4194304
net.core.wmem_max = 134217728
net.ipv4.tcp_wmem = 4194304 67108864 134217728
net.ipv4.tcp_rmem =  4194304 67108864 134217728

net.ipv4.tcp_max_syn_backlog = 1280
net.core.netdev_max_backlog = 30000
```

Linux Kernel Tuning – Not Needed…but…. (1)

Tune for high open files

- Especially with 4.0 and multi-db in cluster

Tune network memory

- Defaults are way too low for a multi-user DBMS

Tune receive backlog

- Tuning send backlog requires using ip link set
  - ✓ *Needs to be put in a script executed by NetworkManager dispatcher*

```
#!/bin/bash
if [ "$1" == "ens160" ] && [ "$2" == "up" ];
then
    ip link set ens160 txqueuelen 5000
fi
```

```
fs.file-max = 5000000
fs.nr_open = 5000000

net.core.rmem_default =   4194304
net.core.rmem_max = 134217728
net.core.wmem_default =   4194304
net.core.wmem_max = 134217728
net.ipv4.tcp_wmem = 4194304 67108864 134217728
net.ipv4.tcp_rmem =  4194304 67108864 134217728

net.ipv4.tcp_max_syn_backlog = 1280
net.core.netdev_max_backlog = 30000
```

Linux Kernel Tuning – Not Needed…but…. (2)

Java apps have to do standard file IO

- No DIRECTIO, CONCURRENTIO nor ASYNCIO

Result: OS filesystem cache will get used extensively

- Filesystem cache can NOT be swapped, so net, net – as it expands, your process gets swapped

Some key controls

- vm.dirty_ratio
  - ✓ *Absolute maximum percent of physical memory that can be consumed by filesystem cache*
  - ✓ *On larger systems, adjust down to 15 or 10*
- dirty_background_ratio
  - ✓ *How much file system cache consumes before the flushers kick in to write to disk - Set it to 5 (or less?)*
- vm.dirty_expire_centisecs
  - ✓ *How long a dirty disk page sits in cache before flusher will try to write it to disk – reduce to 1000 (10 seconds)*
- vm.dirty_writeback_centisecs
  - ✓ *Flusher wake up interval – reduce to 300 (3 seconds)*
- vm.swappiness
  - ✓ *How aggressive kernel swaps*

```
vm.dirty_background_bytes = 0
vm.dirty_background_ratio = 5
vm.dirty_bytes = 0
vm.dirty_expire_centisecs = 1000
vm.dirty_ratio = 20
vm.dirty_writeback_centisecs = 500
vm.swappiness = 10
```

Now you understand why in the sizing we said not to plan for neo4j to have more than 70% of the physical memory – even at 20%, the other 10% will be used by other processes, etc.

/etc/security/limits.conf – this is ***REQUIRED***

```
# /etc/security/limits.conf
…
#Each line describes a limit for a user in the form:
#
#<domain>        <type>  <item>  <value>
#

#@student        hard    nproc           20
#@faculty        soft    nproc           20
…

#setting the file limit to unlimited may prevent logging in
#so use a setting greater than 400000 …especially in 4.0 cluster multi-db
#make sure to tune the kernel first (fs.file-max, fs.nr_open)
neo4j    soft    nofile  2000000
neo4j    hard    nofile  2000000
```

RPM or TGZ → Personal/Organization Preference

RPM

- The good
  - ✓ *Installs neo4j and services*
  - ✓ *Allows to be stopped/started as a service (systemctl [start | stop | restart] neo4j)*
- The bad & ugly
  - ✓ *File locations are a bit scattered (/var/log; /var/lib; etc. – see docs)*
  - ✓ *Installs/execs as root (can be a security issue)*

TGZ

- The good
  - ✓ *Simple install – just untar*
  - ✓ *Files are all under $NEO4J_HOME*
  - ✓ *Owned by neo4j user – started by neo4j user*
    - No root required – although sudo helps during initial install
- The bad
  - ✓ *Neo4j not a service that can be stopped/started via service manager*
  - ✓ *But…how hard is "neo4j start"…. "neo4j stop"*

Linux Steps – OS prep

Create neo4j account & create directories

- Grant sudo
- Create neo4j home directory (/opt/neo4j ?)

Add ports to firewall

- http (7474), https (7473), bolt (7687), backup (6362)
- Plus 3 ports for cluster (5000,6000,7000 by default)

Tune OS kernel

- Number of files, network, filesystem cache flushing
- Set higher limit for neo4j in /etc/security/limits.conf

Change IO scheduler to noop/none

Install JRE or JDK

- JDK if planning on creating stored procs or user-defined functions
- OpenJRE/JDK fine
- JRE 1.8 for 3.5; JRE 11 or 12+ for 4.0

Linux Steps – Install neo4j

Download software & untar

- Download APOC & GraphAlgos for same version
- Copy APOC & GraphAlgos to plug-ins

Set env's, etc. in .bashrc

- JAVA_HOME, PATH, etc.

Change/tune ./conf/neo4j.conf

- Set page cache & heap sizes
- Set ports/ipaddresses for instance & cluster (remember to uncomment default)
- Set transaction log retention (not in file by default)
- Config for APOC/GraphAlgos

Set the initial password

- ./bin/neo4j-admin set-initial-password mysecretpassword

Let'er rip

- ./bin/neo4j start (console on Windows)

Neo4j Download Center - Server

Download what you want/need

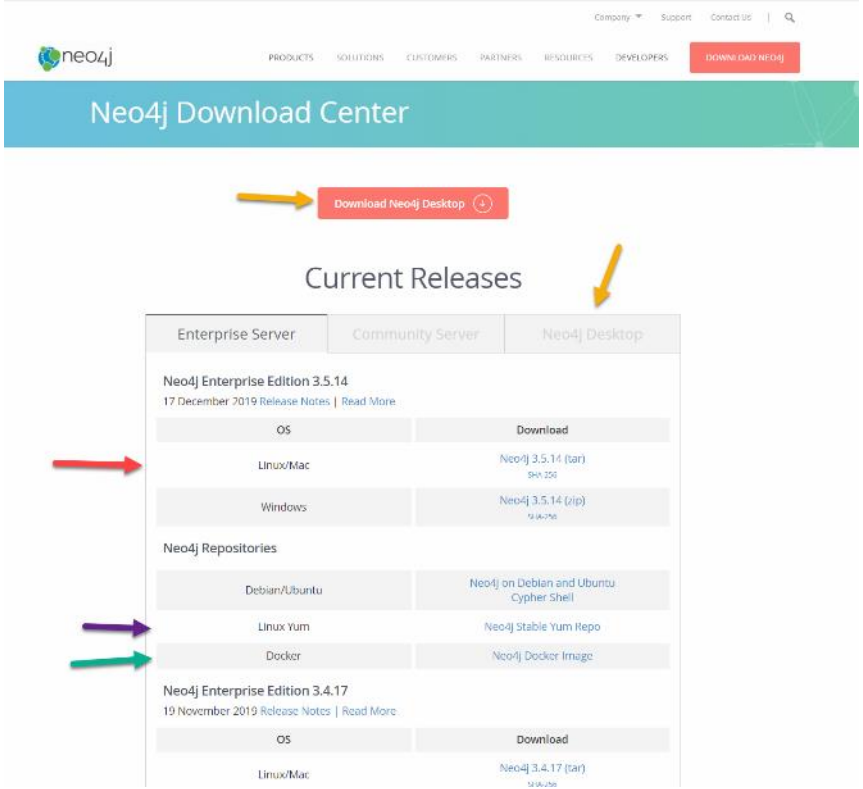
- Tgz, repo, docker

Download desktop for developers

- Includes browsers, plugins
- Includes libs for compiling procs/udfs

Cypher-shell?

- If downloading desktop, it should come with it
- ….but if you want a command line on another host (e.g. remote backup), you might want this to test connections, etc.



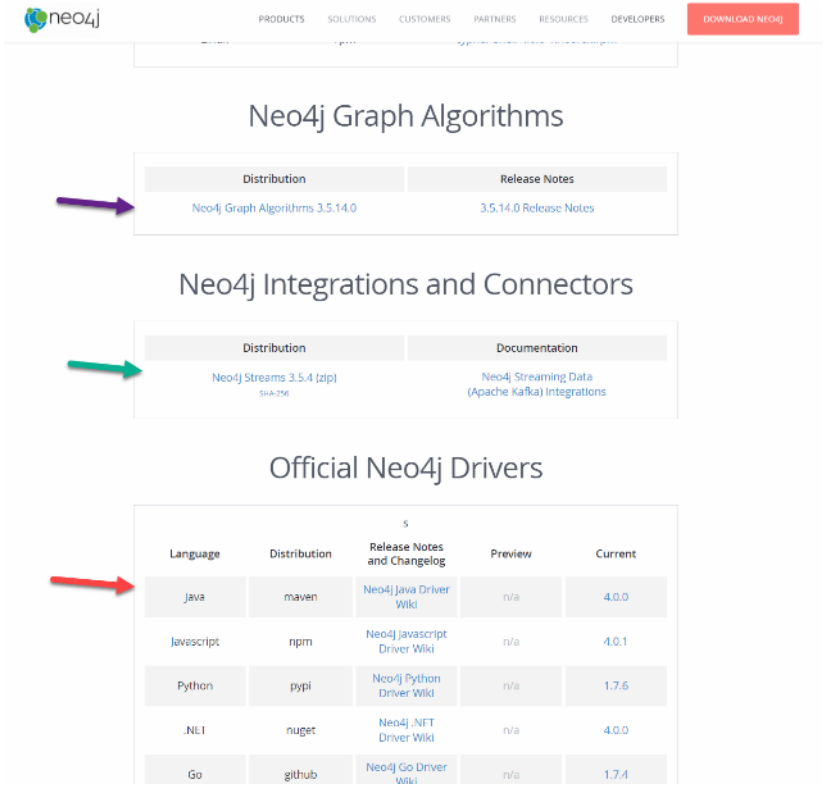
Neo4j Download Center – Algos & Drivers

Graph Algos

- If you want to run these

Official drivers

- Grab the ones for your favorite language
- Some are “official” → supported by Neo4j
- Others are “community”



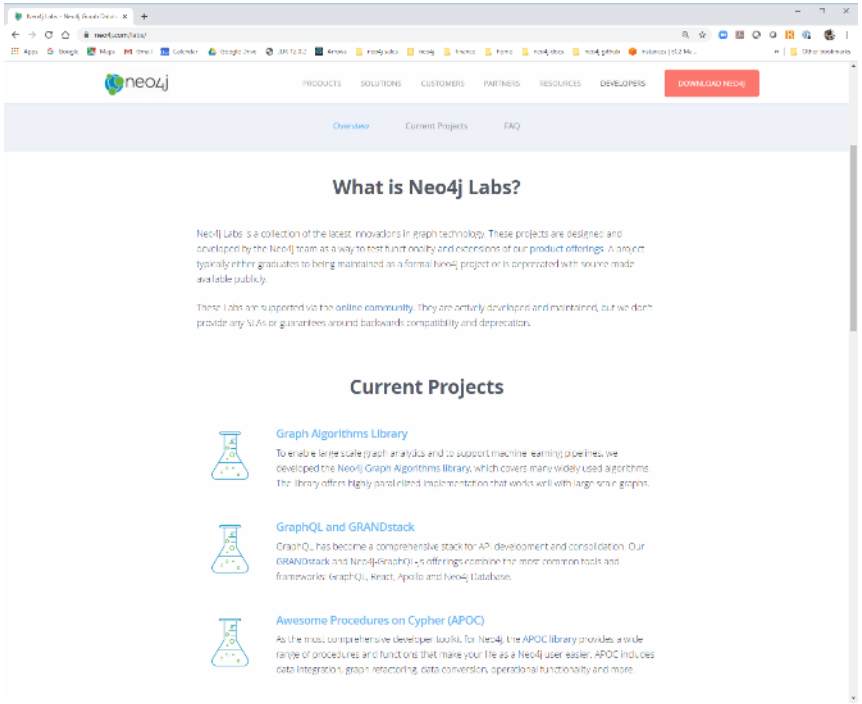
Neo4j Labs Downloads

APOC

Browser Plugins

- Halin - Monitoring
- ETL-Tool
  - ✓ *Quick & dirty conversion from ER/SQL to Graph*

Neosemantics (if coming from RDF)



Common configs to change in ./conf/neo4j.conf

```
dbms.memory.heap.initial_size=32768m
dbms.memory.heap.max_size=32768m
dbms.memory.pagecache.size=12288m
```

64GB host * 0.7 = 44GB for neo4j

```
dbms.connectors.default_listen_address=0.0.0.0          #uncomment
dbms.backup.address=0.0.0.0:6362                        #network backups

dbms.connector.bolt.listen_address=:7687
dbms.connector.http.listen_address=:7474
dbms.connector.https.listen_address=:7473

# Number of Neo4j worker threads.
#dbms.threads.worker_count=
```

Network worker threads – if you have a lot of clients or need very low latency

```
#dbms.tx_log.rotation.retention_policy=7 days    # shorten to 1 day??
dbms.security.procedures.unrestricted=apoc.*,algo.*,bloom.*

# OTHERS: → LDAP, SSL, Causal Clustering, Query Logging
```

What's Else Is In the Box?

Neo4j & Custom Plug-Ins (./plugin

- APOC & Graph Algos
- User-defined functions & stored procs
- Optionally bloom (if server mode)

Neo4j-admin script (./bin/neo4j-admin)

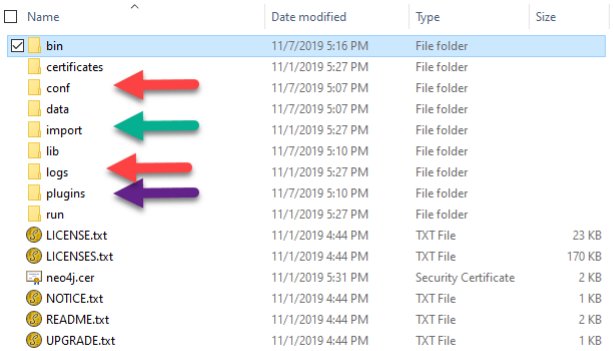
- Used for backups, changing admin passwords,etc.

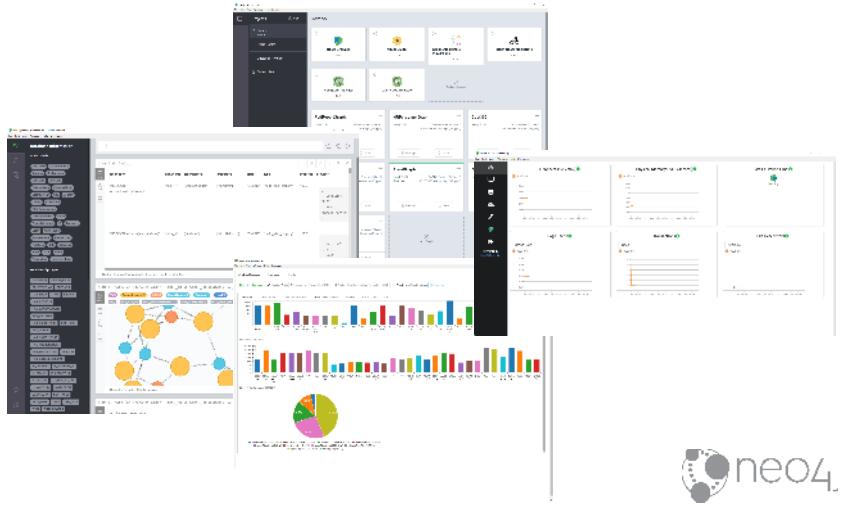
Cypher-shell (./bin/cypher-shell)

- Command line utility for entering cypher

Neo4j Desktop (Browser & Plugins)

- Neo4j Browser
  - ✓ *cypher development/visualization*
  - ✓ *Example databases/tutorials (:play movies, etc.)*
- Bloom – client-side visualization/explore & discover
- Halin – Monitoring
- Query Log Analyzer
- DB Analyzer





Some Lessons I've Learned

If it doesn't start – check NEOJ_HOME/log/server.log

- If APOC is mentioned in stack – probably have more than one APOC jar → delete 1
- If complains about memory – large txn in tran log → increase heap and try again

It's JAVA – if performance rrrreeallyyy matters – use Oracle JRE/JDK

- Internal benchmark for customer showed it 2x faster
- Probably due to better JIT compiler

The transaction logs can take up a lot of space

- Especially when initially loading – demo db load required 23GB of tran log space
- Reason why is default retention is 30 days
  - ✓ *You need to shorten this to 2x your incremental backup interval*
  - ✓ *For development servers, consider turning off retention (demo db then <3GB)*
  - ✓ *You may want to "prune" the logs after large bulk loads (see Ops Manual)*

Clustering & High Availability

neo4j

Neo4j Cluster Concepts – Core Servers

A cluster consists of 3 or more “core” servers

A core server

- Maintained synchronously via RAFT protocol
- Any of the “core” servers could be R/W

Leaders vs. Followers

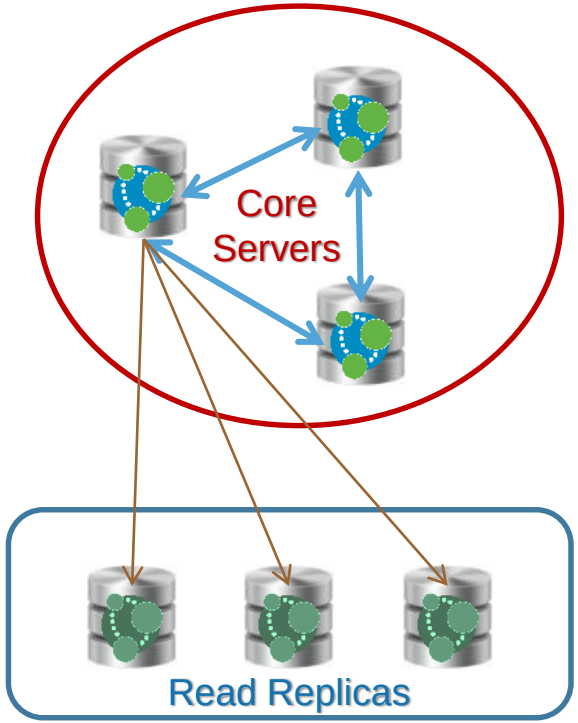
- Each database in a core server has a “Leader”
- Leaders handle ALL write transactions
  - ✓ *….and may handle read transactions as well*
- Followers can handle read transactions for load balancing

A cluster may also have 0 or more “read replicas”

- Read replicas are for read-only
- Useful for scaling-out workloads
- Maintained asynchronously via RAFT protocol via polling

Breaking News: 4.0 Differences

- Each DB in a cluster has its own leader
- Technically each DB could have its leader on different node
- But….
  - ✓ *No way to control this (manually move the leader on-demand)*
  - ✓ *Likely all on the same one due to first node at startup*



Neo4j Cluster Concepts – Number of Nodes

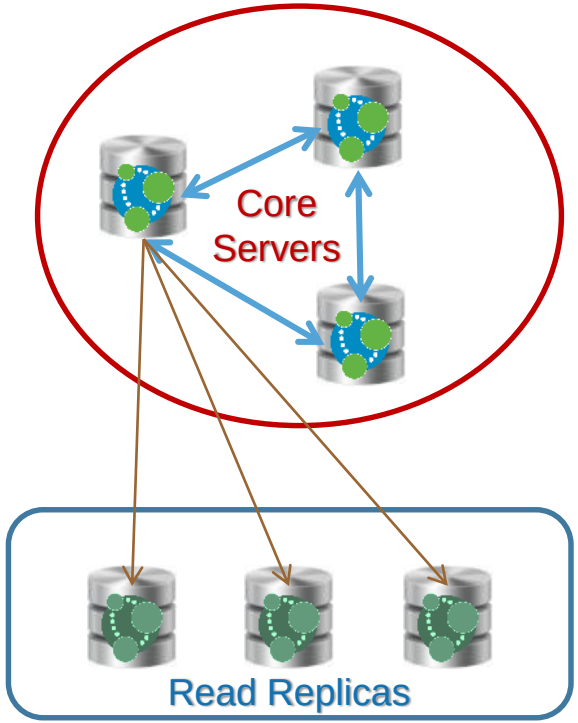
The minimum suggested is 3 “core” servers

Transaction commits & replication

- Transaction starts to commit at Leader
  - ✓ *RAFT log is written*
- Transaction is replicated to all other cores
- When the majority (n/2+1) of cores accept txn, commit is allowed
  - ✓ *“Accept” means raft log is written*
  - ✓ *Note that “voting” is among core servers only*
- Transaction is replicated to read replicas after commit
  - ✓ *Via read replica polling*

Planning for Fault Tolerance

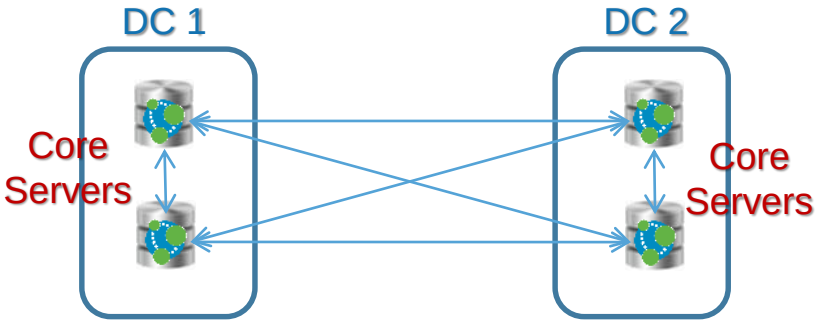
- If the majority of the nodes are not available, the cluster becomes read-only
- For example, in a 3 node cluster, if 2 fail → read-only
  - ✓ *To sustain 2 failures you need 5 core servers*
- Theoretically, you need 2*F + 1 servers
  - ✓ *F is the number of failures you think you will have*
  - ✓ *This tends to make people think you always need an odd number of core servers – but in some cases, an even number may be desirable*



Neo4j Cluster Concepts – Multiple Data Centers

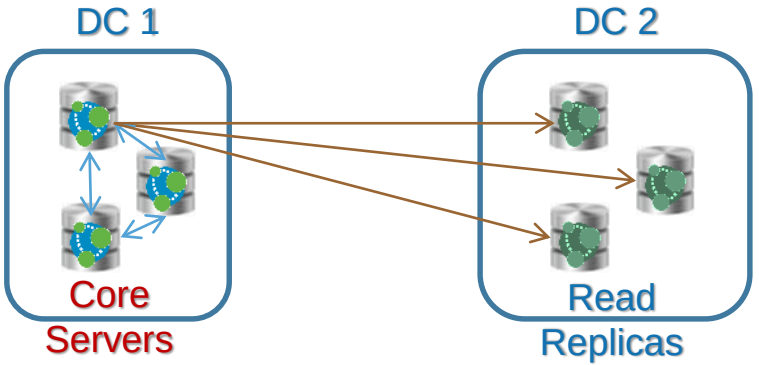
2 Data Centers (fairly close) → HA/DR

- Use an even number of core servers
- This forces that at least one of the "remote" DC core servers is in sync (due to majority voting rules)
- If primary DC fails, all data is at standby DC
- However, cluster is initially read-only (due to lack of majority)
  - ✓ *Can be reconfigured to a stand-alone cluster*
  - ✓ *When DC 1 comes back online -*



2 Data Centers (far apart) → DR Only

- Use an odd number of core servers in primary DC
- Have 1 or more read-replicas in standby DC
- If primary DC fails, reconfigure read-replicas to be a cluster
  - ✓ *Work with neo4j support on this*
  - ✓ *Need to evaluate which one is most caught up to become master*



Neo4j Cluster Concepts – Multiple Clusters

2 Data Centers (far apart) → Autonomy

- Use an odd number of core servers for each cluster in each data center
- Use Kafka to replicate data between clusters
  - ✓ *Neo4j → Kafka Connector*

There are several gotchas

- Conflict resolution
- Transactional integrity

Conflict Resolution

- Any bi-directional replication with any DBMS hits this problem
- Need to logically separate workloads at each site from the other site

Transactional Integrity

- Neo4j's Kafka CDC doesn't understand transactional boundaries
- Therefore a 10 row insert will become 10 inserts

Neo4j Cluster Concepts – Cluster Setup

Disgustingly Easy

- Just edit the conf for each node

Optionally seed the cluster (large DB)

- Copy DB files to the other nodes
- Cluster needs to be down
- Section 5.3 of Operations Manual

What happens

- First node started is leader
- When other nodes join cluster, the leader copies the database to it

Breaking News: 4.0

- The system & neo4j database are created by the first node brought online initially
  - ✓ *They are then sync'd to each other node as nodes are added to the cluster*
- Each new database "leader" is defined on where it is created
  - ✓ *Other nodes will be sync'd*

```
#*****************************************************************
# Causal Clustering Configuration
#*****************************************************************

…
dbms.mode=CORE

…
causal_clustering.minimum_core_cluster_size_at_formation=3

…
causal_clustering.minimum_core_cluster_size_at_runtime=3

…
causal_clustering.initial_discovery_members=host-1:5000,host-2:5000,host-3:5000

…
#causal_clustering.discovery_listen_address=host-2:5000

…
#causal_clustering.transaction_listen_address=host-2:6000

…
#causal_clustering.raft_listen_address=host-2:7000

…
#causal_clustering.server_groups=
```

Neo4j Cluster Concepts – Workload Management

Cluster supports automatic workload management

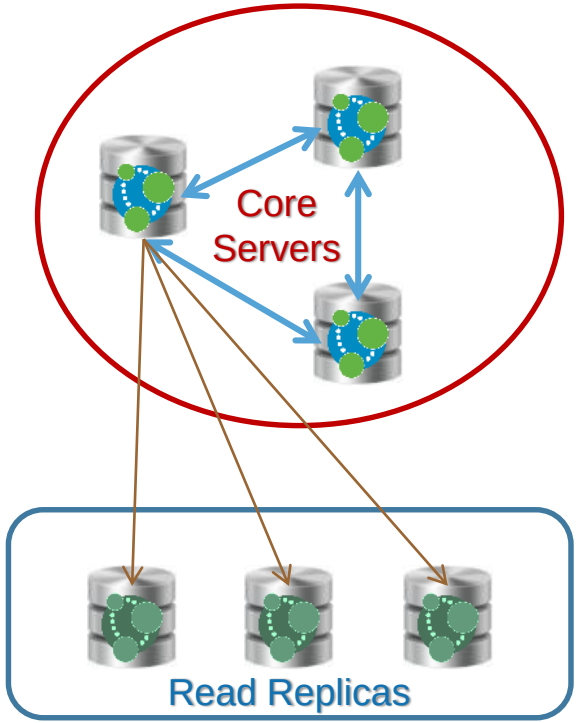
- …depending on protocol
- …http, https, bolt stay connected to requested server
- Bolt+routing (aka neo4j in 4.) will be routed

What happens

- Write transactions are routed to leader – regardless
- Read transactions can be routed to any other core server or read replica depending on workload/load balancing

Read-your-own-writes

- Neo4j has concept of “bookmarks”
- Write transaction creates a “bookmark”
- Subsequent read transactions routed to a follower server
  - ✓ *…wait for their bookmark to arrive → then read*
  - ✓ *Note that since we are replicating the log, commit order is maintained. Therefore you essentially are waiting for all txns prior to yours to arrive as well (if lagging)*
  - ✓ *Think about it – if follower is member of voting majority, effectively, there is no wait – only time there is a wait is if the follower is not in majority and lagging…or a read-replica*



Advice for running clusters…from the trenches

Keep transaction sizes small (100's to 1000 at most)

- Those 1M node CREATE scripts will use a LOT of memory for the RAFT protocol and take a long time to sync….
- Use periodic commit and LIMIT clause

Don't run graph algos

- Loading a graph into memory …..takes a lot of memory (memrec)
- Writing graph algos back….is a rrrreaaalllly *HUGE* txn (see rule #1)
- Having said that, you *might* be able to run small graphs
- Don't try read-replica either (what part of Read Only did you not understand?)

A lot of admin commands are asynchronous

- Wait for them to complete before going to the next step
- E.g. create database (4.0)

Backups & Recovery

neo4j

Before we begin: What are your requirements?

How often does data change?

How much can you stand to lose (if any)?

How large is the database?

What policies do we need to adhere to?

How much data is changed overnight?

- E.g. if bulk loading, is incremental backups really going to help?

How quickly do you need to recover?

- Are storage snapshots a better option?
- Remember, for DBMS backups
  - ✓ *DBMS backup must read all the data from data devices*
  - ✓ *….and then write all the data to filesystem devices (guaranteed to fill filesystem cache)*
  - ✓ *DBMS restore does the reverse – reads from filesystem (with filesystem readahead)*
  - ✓ *….and then must write to data devices*
- …a lot of CPU and OS/HW IO processing

Backup

Hot backup (recommended) (Enterprise only)

- Make a backup while the database is online
- Non-blocking, triggers a checkpoint
- Allows for incremental backups on top of a full backup
- Uses neo4j-admin utility program

Cold backup (only option in Community)

- Requires stopping the database completely first
- Simply copy the neo4j data directory
- Reasonable approach when using Read Replicas
- Make sure to run manual consistency check afterwards!

Backup non-database files periodically

- Neo4j.conf
- SSL certificates
- ./plugins (APOC + custom procs, etc.)
- Etc.

Syntax

```
neo4j-admin backup --backup-dir=<backup-path> --name=<graph.db-backup>
                   [--from=<address>] [--protocol=<any|catchup|common>]
                   [--fallback-to-full[=<true|false>]]
                   [--pagecache=<pagecache>]
                   [--timeout=<timeout>]
                   [--check-consistency[=<true|false>]]
                   [--additional-config=<config-file-path>]
                   [--cc-graph[=<true|false>]]
                   [--cc-indexes[=<true|false>]]
                   [--cc-label-scan-store[=<true|false>]]
                   [--cc-property-owners[=<true|false>]]
                   [--cc-report-dir=<directory>]
```

Options

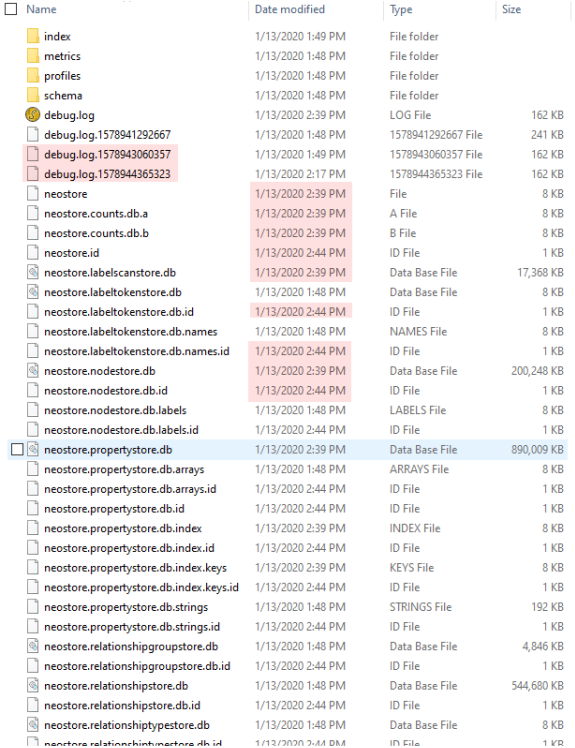
Option	Default	Description
--backup-dir		Directory to place backup in.
--name		Name of backup. If a backup with this name already exists an incremental backup will be attempted.
--from	localhost:6362	Host and port of Neo4j.
--protocol	any	Protocol over which to perform backup. If set to `any`, then `catchup` will be tried first. If that fails, then it will attempt to fall back to `common`. It is recommended to set this option explicitly. Set it to `catchup` for Causal Cluster backups, and to `common` for HA or single-instance backups. For more information, see Backup scenarios and examples.
--fallback-to-full	true	If an incremental backup fails backup will move the old backup to <name>.err.<N> and fallback to a full backup instead.
--pagecache	8M	The size of the page cache to use for the backup process.

Backup Example

```
C:\Neo4j\Data\neo4jDatabases\database-Fraud\installation-3.5.12\bin> .\neo4j-admin.bat backup --backup-dir "C:\neo4j\backups" --name="fraud_13jan2020.db-bck"
2020-01-13 18:48:12.691+0000 INFO [o.n.b.i.BackupOutputMonitor] Start receiving store files
2020-01-13 18:48:13.216+0000 INFO [o.n.b.i.BackupOutputMonitor] Start receiving store file C:\Neo4j\backups\fraud_13jan2020.db-bck\temp-copy\neostore.nodestore.db.labels
2020-01-13 18:48:13.228+0000 INFO [o.n.b.i.BackupOutputMonitor] Finish receiving store file C:\Neo4j\backups\fraud_13jan2020.db-bck\temp-copy\neostore.nodestore.db.labels
...
2020-01-13 18:48:51.463+0000 INFO [o.n.b.i.BackupOutputMonitor] Start receiving store file C:\Neo4j\backups\fraud_13jan2020.db-bck\temp-copy\neostore
2020-01-13 18:48:51.465+0000 INFO [o.n.b.i.BackupOutputMonitor] Finish receiving store file C:\Neo4j\backups\fraud_13jan2020.db-bck\temp-copy\neostore
2020-01-13 18:48:51.467+0000 INFO [o.n.b.i.BackupOutputMonitor] Finish receiving store files, took 38s 771ms
2020-01-13 18:48:51.592+0000 INFO [o.n.b.i.BackupOutputMonitor] Start recovering store
2020-01-13 18:48:57.205+0000 INFO [o.n.b.i.BackupOutputMonitor] Finish recovering store, took 5s 613ms
2020-01-13 18:49:02.203+0000 INFO [o.n.b.i.BackupOutputMonitor] Finished, took 49s 513ms
Full Consistency Check
..................  10%
..................  20%
..................  30%
..................  40%
..................  50%
..................  60%
..................  70%
..................Checking node and relationship counts
..................  10%
..................  20%
..................  30%
..................  40%
..................  50%
..................  60%
..................  70%
..................  80%
..................  90%
.................. 100%
Backup complete.
PS C:\Neo4j\Data\neo4jDatabases\database-Fraud\installation-3.5.12\bin>
```

Incremental Backup Example

```
PS C:\Neo4j\Data\neo4jDatabases\database-Fraud\installation-3.5.12\bin> .\neo4j-admin.bat backup --backup-dir "C:\neo4j\backups" --
name="fraud_13jan2020.db-bck"     Full Consistency Check
.....................  10%
.....................  20%
.....................  30%
.....................  40%
.....................  50%
.....................  60%
.....................  70%
..................Checking node and relationship counts
.....................  10%
.....................  20%
.....................  30%
.....................  40%
.....................  50%
.....................  60%
.....................  70%
.....................  80%
.....................  90%
..................... 100%
Backup complete.
PS C:\Neo4j\Data\neo4jDatabases\database-Fraud\installation-3.5.12\bin>
```



Name	Date modified	Type	Size
index	1/13/2020 1:49 PM	File folder	
metrics	1/13/2020 1:48 PM	File folder	
profiles	1/13/2020 1:48 PM	File folder	
schema	1/13/2020 1:48 PM	File folder	
debug.log	1/13/2020 2:39 PM	LOG File	162 KB
debug.log.1578941292667	1/13/2020 1:48 PM	1578941292667 File	241 KB
debug.log.1578943060357	1/13/2020 1:49 PM	1578943060357 File	162 KB
debug.log.1578944365323	1/13/2020 2:17 PM	1578944365323 File	162 KB
neostore	1/13/2020 2:39 PM	File	8 KB
neostore.counts.db.a	1/13/2020 2:39 PM	A File	8 KB
neostore.counts.db.b	1/13/2020 2:39 PM	B File	8 KB
neostore.id	1/13/2020 2:44 PM	ID File	1 KB
neostore.labelscanstore.db	1/13/2020 2:39 PM	Data Base File	17,368 KB
neostore.labeltokenstore.db	1/13/2020 1:48 PM	Data Base File	8 KB
neostore.labeltokenstore.db.id	1/13/2020 2:44 PM	ID File	1 KB
neostore.labeltokenstore.db.names	1/13/2020 1:48 PM	NAMES File	8 KB
neostore.labeltokenstore.db.names.id	1/13/2020 2:44 PM	ID File	1 KB
neostore.nodestore.db	1/13/2020 2:39 PM	Data Base File	200,248 KB
neostore.nodestore.db.id	1/13/2020 2:44 PM	ID File	1 KB
neostore.nodestore.db.labels	1/13/2020 1:48 PM	LABELS File	8 KB
neostore.nodestore.db.labels.id	1/13/2020 2:44 PM	ID File	1 KB
neostore.propertystore.db	1/13/2020 2:39 PM	Data Base File	890,009 KB
neostore.propertystore.db.arrays	1/13/2020 1:48 PM	ARRAYS File	8 KB
neostore.propertystore.db.arrays.id	1/13/2020 2:44 PM	ID File	1 KB
neostore.propertystore.db.id	1/13/2020 2:44 PM	ID File	1 KB
neostore.propertystore.db.index	1/13/2020 2:39 PM	INDEX File	8 KB
neostore.propertystore.db.index.id	1/13/2020 2:44 PM	ID File	1 KB
neostore.propertystore.db.index.keys	1/13/2020 2:39 PM	KEYS File	8 KB
neostore.propertystore.db.index.keys.id	1/13/2020 2:44 PM	ID File	1 KB
neostore.propertystore.db.strings	1/13/2020 1:48 PM	STRINGS File	192 KB
neostore.propertystore.db.strings.id	1/13/2020 2:44 PM	ID File	1 KB
neostore.relationshipgroupstore.db	1/13/2020 1:48 PM	Data Base File	4,846 KB
neostore.relationshipgroupstore.db.id	1/13/2020 2:44 PM	ID File	1 KB
neostore.relationshipstore.db	1/13/2020 1:48 PM	Data Base File	544,680 KB
neostore.relationshipstore.db.id	1/13/2020 2:44 PM	ID File	1 KB
neostore.relationshiptypestore.db	1/13/2020 1:48 PM	Data Base File	8 KB

Hot Backup – Recommended Policy

Wrap the neo4j-admin backup in a script, and schedule via cron or similar

- Full backup daily (or as appropriate)
- Incremental backup hourly (or as appropriate)

In a cluster, run against Read Replicas, or Followers, only

Consistency check run automatically

- For stores under XX, leave it as is
- For stores over XX, skip it and run manually
- Our script goes for the latter as its simple enough when wrapped in a script

Backup locally and then copy the backup to remote storage

Hot Backup – Wrapper Template

We provide a wrapper for you to use or modify to suit your needs

- Ask support/customer success for current location

You need to customize some of the variables for your environment:

- `NEO4J_HOME="/usr/share/neo4j" #CHANGE ME`
- `backupFileDir="$NEO4J_HOME/backups" #POSSIBLY CHANGE ME`
- `bname=graph.db-backup-$timestamp #POSSIBLY CHANGE ME`
- `fromHost=localhost #CHANGE ME IF RUNNING REMOTELY`
- `heapSize=2G #POSSIBLY CHANGE ME`
- `pageCache=1G #POSSIBLY CHANGE ME`

Also, you may want the backup log written to the Neo4j logs directory instead of to bin:

- `logFile=$NEO4J_HOME/logs/backupDatabase_$timestamp.log`

backupDatabase.sh – How does it work?

Step 1: Place `backupDatabase.sh` in `NEO4J_HOME/bin`

This is designed to run via a regular job (ex. `cron`)

- Example cron entry to run every hour on the hour (will result in 1 full then 23 incrementals)
- Assumes `NEO4J_HOME=/usr/share/neo4j`
  `0 * * * * ="/usr/share/neo4j/bin/backupDatabase.sh`

This will backup to `NEO4J_HOME/backups` each time the script is run

Because the backup target includes the current date stamp, it will run incremental backups if run more than once per day

- `graph.db-backup-$timestamp`
- If you just want to do a single full backup and then always incremental after that, you will modify this setting to be just: `bname=graph.db-backup`

It skips the consistency check, then runs it manually after the backup completes.

There are many options for extending its capabilities, lets explore some…

backupDatabase.sh – Extending the wrapper

Copy the backup to external storage

- Most policies require this.
- You may simply need to copy to a mounted filesystem, or to an S3 bucket or similar.

You will likely want to compress this to save space (remember to un-compress before restoring!)

- Replace this line with custom commands:

Insert copy to external storage here...

Assuming external mounted storage is `/mnt/neo4jbackups`

- `cp –R "${backupFileDir}/${bname}" /mnt/neo4jbackups`

*OR to compress **(advised)***

- `tar –czf "/mnt/neo4jbackups/${backupFileDir}/${bname}.tgz" "${backupFileDir}/${bname}"`

backupDatabase.sh – Extending the wrapper

Send an email alert upon completion

- Note: Requires sendmail be available
- Add these lines to the beginning of the script:

```
# Mail variables
#
SENDMAIL=/usr/sbin/sendmail
export DBA_EMAIL=dbaemail@acme.com
export DBA_PAGER=dbapager@acme.com
export BACKUP_FROM=backup.job@neo4jcc-01
EMAIL_HDR="To:${DBA_EMAIL}\nFrom:${BACKUP_FROM}"
SUBJ_F="Subject: BACKUP of Neo4j instance at `date` --- FAILED\n"
SUBJ_S="Subject: BACKUP of Neo4j instance at `date` --- SUCCESS\n"
SUBJ_W="Subject: BACKUP of Neo4j instance at `date` --- SUCCESS WITH WARNING\n"
```

Uncomment the sendmail commands throughout the script

backupDatabase.sh – Advanced Best Practice

De-centralize the backup job

- It is advised to de-centralize the backup script, in the event that the cluster member that is tasked to run the backup routine is offline at the time
- Simply untar the Neo4j binaries on a separate server, and call remotely

Even better:

- Build a check into the script to iterate through a list of IPs/Hostnames in the event the first one isn't available
  - ✓ *That list should be Read Replicas and then Followers*

Cold Backup – Considerations

Make sure the database is fully stopped before making a copy

Make sure the copy is fully completed before starting the database

- Failure to do both will likely result in a corrupt copy

If you have Neo4j Enterprise, then run a manual consistency check after the manual copy is complete to make sure it is clean!

Backup Non-database Files

Periodically backup the following files/directories

- conf/neo4j.conf
- data/dbms
- logs/
- plugins/

Restore – Best Practices

Single Instance

- After stopping the database, copy **data/databases/graph.db** in case you need it later

Causal Cluster

- After stopping the database, copy both **data/databases/graph.db** and **data/cluster_state** in case you need them later
- Run **`neo4j-admin restore`** on each core cluster member
- Run **`neo4j-admin unbind`** on each core member

Important: You cannot restore to a Docker container running Neo4j!

- ***See: https://support.neo4j.com/hc/en-us/articles/115015674407-Stopping-the-Neo4j-docker-image-in-order-to-restore-from-a-backup***

Better yet, use our Restore Wrapper Template (next slide)

Restore – Wrapper Template

We provide a wrapper for you to use or modify to suit your needs

- Ask support/customer success for current location

You need to customize some of the variables for your environment:

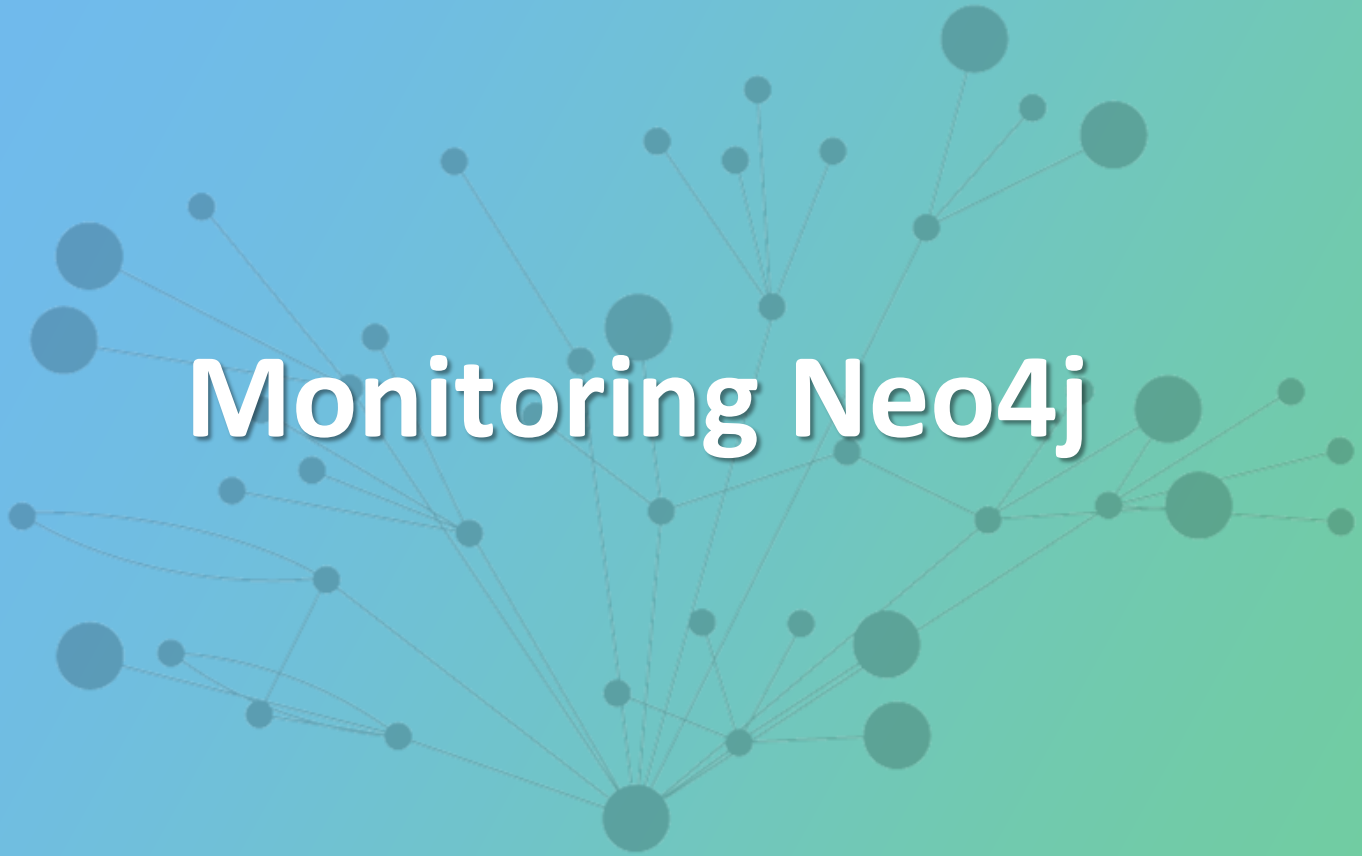
- `NEO4J_HOME="/usr/share/neo4j" #CHANGE ME`
- `backupFileDir="$NEO4J_HOME/backups" #PROBABLY CHANGE ME`
- `backupDbName=graph.db-backup #PROBABLY CHANGE ME`
- `dbName=graph.db #POSSIBLY CHANGE ME`

Restore – Wrapper Template

How does it work?

- Simplifies the process and prompts for required actions:
  - ✓ *Lets you know if DB is still running*
  - ✓ *Lets you know it is overwriting the existing graph*
  - ✓ *Asks you to tar gzip the current graph (press y to do it)*
  - ✓ *Backs up cluster_state in Causal Cluster*
  - ✓ *Unbinds the instance in Causal Cluster*
  - ✓ *Restores the database*

In a Causal Cluster, run it on each Core member (and optionally Read Replicas, unless you want them to pull from a core member)



Monitoring Neo4j

neo4j

Different levels of monitoring

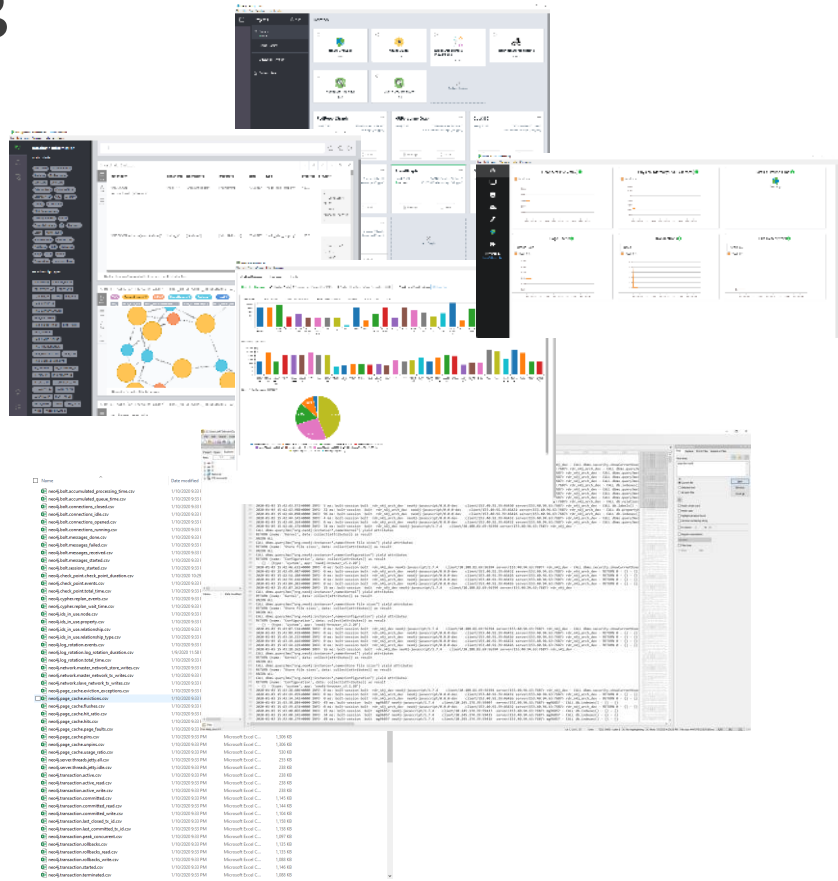
Monitoring neo4j.log & debug.log

Long running queries

Metrics

- Cache health/memory
- Checkpoint frequency/duration
- GC frequency/duration
- ….and lots more

Query Plan/Diagnostics



Neo4j.log

Mostly useful if server stack traces…

Otherwise a few tidbits

- Logins failing due to authentication failure
- Whether metrics are enabled
- Memory statistics (seems to be more prevalent when GC – but maybe anecdotal)
- Some ‘you were stupid things’ …e.g. can’t drop a constraint that doesn’t exist

Usually file is really small….

- Unless you are experiencing memory pressure or stack tracing…

What you DON’T want to see….

```
2019-11-26 07:42:51.779+0000 ERROR Client triggered an unexpected error [Neo.DatabaseError.Transaction.TransactionStartFailed]:
        The database has encountered a critical error, and needs to be restarted. Please see database logs for more details., reference 6e24014a-52b4-460c-8e5b-5e253f522a68.
2019-11-26 07:42:51.798+0000 ERROR Client triggered an unexpected error [Neo.DatabaseError.Transaction.TransactionStartFailed]:
        The database has encountered a critical error, and needs to be restarted. Please see database logs for more details., reference fe89987e-3330-40f4-b15e-822c29a4d0e0.
2019-11-26 07:42:51.824+0000 ERROR Client triggered an unexpected error [Neo.DatabaseError.Transaction.TransactionStartFailed]:
        The database has encountered a critical error, and needs to be restarted. Please see database logs for more details., reference 57582f6e-df00-4a02-90ca-9b8d563716de.
```

(This usually means you have to kill the JVM to kill the DB)

Monitor debug.log for JRE Garbage Collections

The Good & Bad

- Neo4j is written in Java – so easily extendable
- Java doesn't manage memory well – hence the GC requirements
  - ✓ *Unlike C which can preallocate memory pools and app can use memory from pool*

Neo4j invokes a GC every 10 seconds

- This is a "stop-the-world"- all processing halts (ouch!)
- Puts an entry in debug.log only if (by default) GC exceeds 100ms
- If you see a bunch with ~100-200ms, this can be normal….
  - ✓ *During loads with MERGE*
  - ✓ *During complex queries*
- What you DON'T want to see is GC taking multiple seconds (especially 10)
  - ✓ *gcCount >1 or gcTime >1000ms*
  - ✓ *If you do – likely too large of heap needed to hold transaction state/results*

Debug.log & Stop The World GC's…meh – normal?

```
2020-01-02 20:07:04.320+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=512, gcTime=520, gcCount=1}
2020-01-02 20:07:11.449+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=255, gcTime=281, gcCount=1}
2020-01-02 20:07:14.923+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=129, gcTime=194, gcCount=1}
2020-01-02 20:07:20.285+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=146, gcTime=205, gcCount=1}
2020-01-02 20:07:23.505+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=125, gcTime=182, gcCount=1}

2020-01-02 20:48:25.376+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=221, gcTime=225, gcCount=1}
2020-01-02 20:48:35.397+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=442, gcTime=437, gcCount=1}
2020-01-02 20:48:39.834+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=394, gcTime=461, gcCount=1}
2020-01-02 20:48:43.528+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=142, gcTime=165, gcCount=1}
2020-01-02 20:48:46.065+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=121, gcTime=186, gcCount=1}
2020-01-02 20:48:48.749+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=183, gcTime=196, gcCount=1}
2020-01-02 20:48:51.570+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=105, gcTime=159, gcCount=1}

2020-01-02 21:02:10.562+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=128, gcTime=160, gcCount=1}
2020-01-02 21:08:34.085+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=148, gcTime=137, gcCount=1}
2020-01-02 22:15:50.540+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=151, gcTime=150, gcCount=1}
2020-01-02 23:42:10.541+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=111, gcTime=149, gcCount=1}
2020-01-03 01:40:20.679+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=244, gcTime=267, gcCount=1}
2020-01-03 01:59:30.911+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=451, gcTime=474, gcCount=1}
2020-01-03 02:18:33.046+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=139, gcTime=138, gcCount=1}
2020-01-03 02:56:22.666+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=145, gcTime=161, gcCount=1}
2020-01-03 04:33:40.528+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=106, gcTime=124, gcCount=1}
```

Intermittent…mostly ~150ms …only a few north of 250ms, all gcCounts=1 – probably due to large data sets/low filtering, which unfortunately can happen when pattern matching to try to find trends

Debug.log & Stop The World GC's….Uh Oh!!!!!!

```
2020-01-09 20:32:30.130+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=6657, gcTime=6679, gcCount=1}
2020-01-09 20:32:44.356+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=10299, gcTime=10322, gcCount=1}
2020-01-09 20:32:51.821+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=5452, gcTime=5480, gcCount=1}
2020-01-09 20:33:04.191+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=11351, gcTime=11419, gcCount=1}
2020-01-09 20:33:18.537+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=13541, gcTime=13582, gcCount=1}
2020-01-09 20:33:29.868+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=11030, gcTime=11058, gcCount=1}
2020-01-09 20:33:41.006+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=10937, gcTime=10975, gcCount=1}
2020-01-09 20:33:46.292+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=5185, gcTime=5202, gcCount=1}
2020-01-09 20:33:56.727+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=10335, gcTime=10384, gcCount=1}
2020-01-09 20:34:07.509+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=10681, gcTime=10766, gcCount=1}
2020-01-09 20:34:11.790+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=4180, gcTime=4272, gcCount=1}
2020-01-09 20:34:15.948+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=4058, gcTime=4149, gcCount=1}
2020-01-09 20:34:24.967+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=8919, gcTime=9012, gcCount=1}
2020-01-09 20:34:29.400+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=4332, gcTime=4419, gcCount=1}
2020-01-09 20:34:34.189+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=4688, gcTime=4781, gcCount=1}
2020-01-09 20:34:38.922+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=4632, gcTime=4730, gcCount=1}
2020-01-09 20:34:43.665+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=4644, gcTime=4737, gcCount=1}
2020-01-09 20:34:53.560+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=4835, gcTime=4934, gcCount=1}
2020-01-09 20:35:03.205+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=14503, gcTime=14601, gcCount=2}
2020-01-09 20:35:12.704+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=9399, gcTime=9497, gcCount=1}
2020-01-09 20:35:23.462+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=10658, gcTime=10756, gcCount=1}
2020-01-09 20:35:32.814+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=9251, gcTime=9349, gcCount=1}
2020-01-09 20:35:46.741+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=9221, gcTime=9320, gcCount=1}
2020-01-09 20:35:54.962+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=12726, gcTime=12824, gcCount=2}
2020-01-09 20:37:52.682+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=21701, gcTime=21798, gcCount=4}
2020-01-09 20:44:02.998+0000 WARN [o.n.k.i.c.VmPauseMonitorComponent] Detected VM stop-the-world pause: {pauseTime=306212, gcTime=310979, gcCount=62}
```

AAAAAAAAAAAHHHHHHHHHHHHHHHHHH!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! (SPLAT!)

Capturing Slow Running Queries (My Fav!)

Turn on query logging

- Long running queries above a threshold will write out to a series of files (configurable)
- Setting the threshold to 0 gets everything….that means everything…

Extremely useful

- ….especially since bad queries is 90%+ of any DB performance issues
- Gets the EXACT query – with parameters (if configured)….all 1000+ values if necessary
  - ✓ *Helps for repro*
- Can report how much memory was used and query exec times

Tales from the trenches

- Queries are logged when they COMPLETE execution
  - ✓ *….so….if still running – then you need "call dbms.listQueries()"*
- Tools such as neo4j browser are kinda chatty…..okay, they are *extremely* chatty
  - ✓ *So….if you use 0……*
- Look for the client IP/pid (avoid confusing bad guy with just annoying minutia)

Just set it up already!!!

Configuring for query logging

```
# Log executed queries that takes longer than the configured threshold. Enable by uncommenting this line.
dbms.logs.query.enabled=true

# If the execution of query takes more time than this threshold, the query is logged. If set to zero then all queries
# are logged.
dbms.logs.query.threshold=10

# The file size in bytes at which the query log will auto-rotate. If set to zero then no rotation will occur. Accepts a
# binary suffix "k", "m" or "g".
#dbms.logs.query.rotation.size=20m

# Maximum number of history files for the query log.
#dbms.logs.query.rotation.keep_number=7

# Include parameters for the executed queries being logged (this is enabled by default).
dbms.logs.query.parameter_logging_enabled=true

# Uncomment this line to include detailed time information for the executed queries being logged:
dbms.logs.query.time_logging_enabled=true

# Uncomment this line to include bytes allocated by the executed queries being logged:
dbms.logs.query.allocation_logging_enabled=true

# Uncomment this line to include page hits and page faults information for the executed queries being logged:
dbms.logs.query.page_logging_enabled=true
```

Example Output (mostly thanks to neo4j browser)

```
2020-01-03 16:50:10.730+0000 INFO  0 ms: bolt-session  bolt  …    neo4j-javascript/1.7.4     client/10.108.82.69:56757   server/153.40.94.63:7687>   … –
                CALL dbms.security.showCurrentUser() - {} - {type: 'system', app: 'neo4j-browser_v3.2.20'}
2020-01-03 16:50:11.857+0000 INFO  0 ms: bolt-session  bolt  …    neo4j-javascript/0.0.0-dev client/153.40.92.39:47008   server/153.40.94.63:7687>   … -
                RETURN 0 - {} - {}
2020-01-03 16:50:21.871+0000 INFO  0 ms: bolt-session  bolt  …    neo4j-javascript/0.0.0-dev client/153.40.92.39:47008   server/153.40.94.63:7687>   … -
                RETURN 0 - {} - {}
2020-01-03 16:50:27.755+0000 INFO  10 ms: bolt-session bolt  …    neo4j-javascript/1.7.4     client/10.245.174.39:59657  server/153.40.94.63:7687>   … –
CALL dbms.queryJmx("org.neo4j:instance=*,name=Kernel") yield attributes
RETURN {name: 'Kernel', data: collect(attributes)} as result
UNION ALL
CALL dbms.queryJmx("org.neo4j:instance=*,name=Store file sizes") yield attributes
RETURN {name: 'Store file sizes', data: collect(attributes)} as result
UNION ALL
CALL dbms.queryJmx("org.neo4j:instance=*,name=Configuration") yield attributes
RETURN {name: 'Configuration', data: collect(attributes)} as result - {} - {type: 'system', app: 'neo4j-browser_v3.2.20'}
2020-01-03 16:50:27.786+0000 INFO  24 ms: bolt-session           bolt  …         neo4j-javascript/1.7.4      client/10.245.174.39:59661
                server/153.40.94.63:7687>   … –
CALL db.labels() YIELD labelRETURN {name:'labels', data:COLLECT(label)[..1000]} as result
UNION ALL
CALL db.relationshipTypes() YIELD relationshipTypeRETURN {name:'relationshipTypes', data:COLLECT(relationshipType)[..1000]} as result
UNION ALL
CALL db.propertyKeys() YIELD propertyKey
RETURN {name:'propertyKeys', data:COLLECT(propertyKey)[..1000]} as result
UNION ALL
CALL dbms.functions() YIELD name, signature, descriptionRETURN {name:'functions', data: collect({name: name, signature: signature, description:
description})} AS result
UNION ALL
CALL dbms.procedures() YIELD name, signature, description
RETURN {name:'procedures', data:collect({name: name, signature: signature, description: description})} as result
UNION ALL
MATCH () RETURN { name:'nodes', data:count(*) } AS result
UNION ALL
MATCH ()-[]->() RETURN { name:'relationships', data: count(*)} AS result - {} - {type: 'system', app: 'neo4j-browser_v3.2.20'}
```

Long running queries

There are two reasons why a Cypher query may take a long time:

The query returns a lot of data.

- The query completes execution in the graph engine, but it takes a long time to create the result stream.

```
MATCH (a)--(b)--(c)--(d)--(e)--(f) RETURN a
```

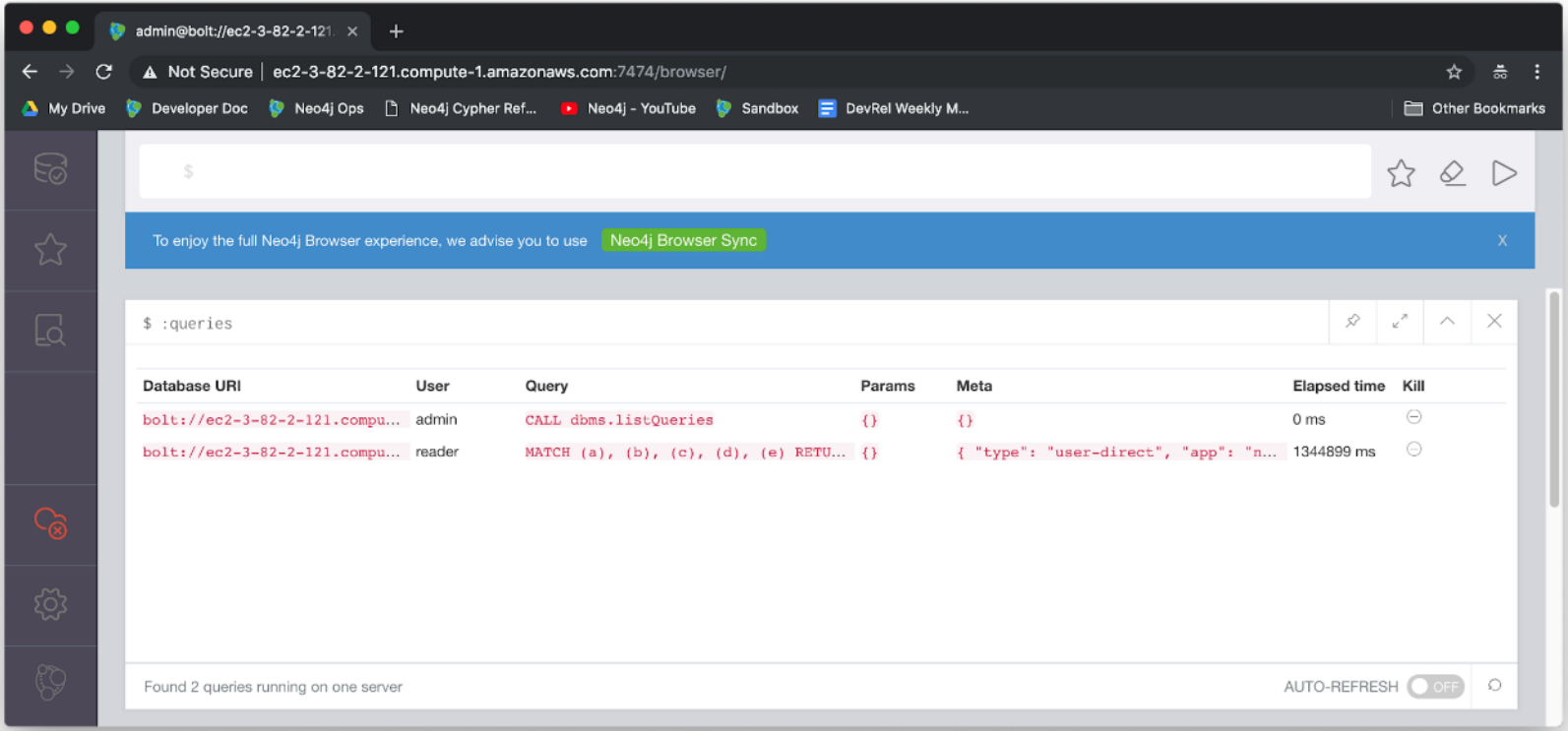
The query takes a long time to execute in the graph engine.

- Has to scan a lot of data

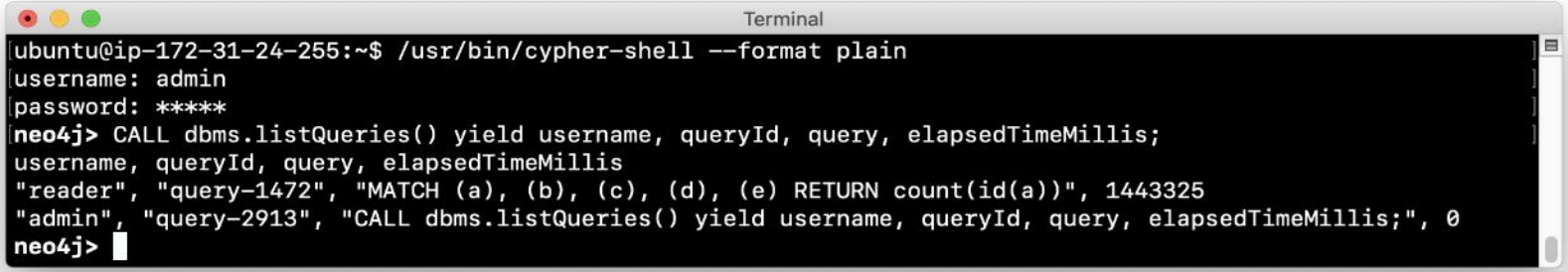
```
MATCH (a), (b), (c), (d), (e) RETURN count(id(a))
```

- Obviously, a Cartesian – and can take a long time
- A more common reason is “tree explosion” – every hop expands by 100’s of nodes

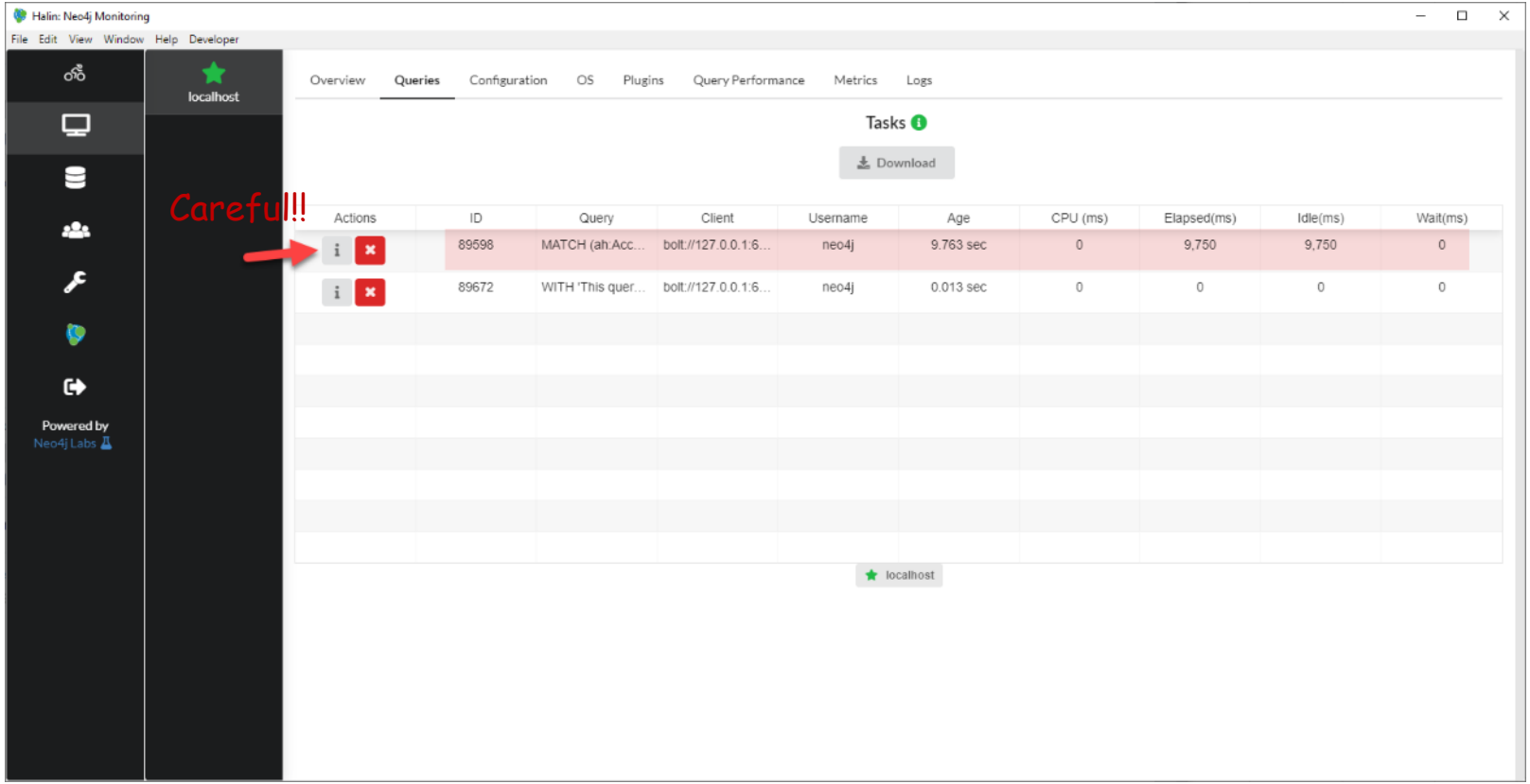
Viewing currently running queries (Neo4j Browser)



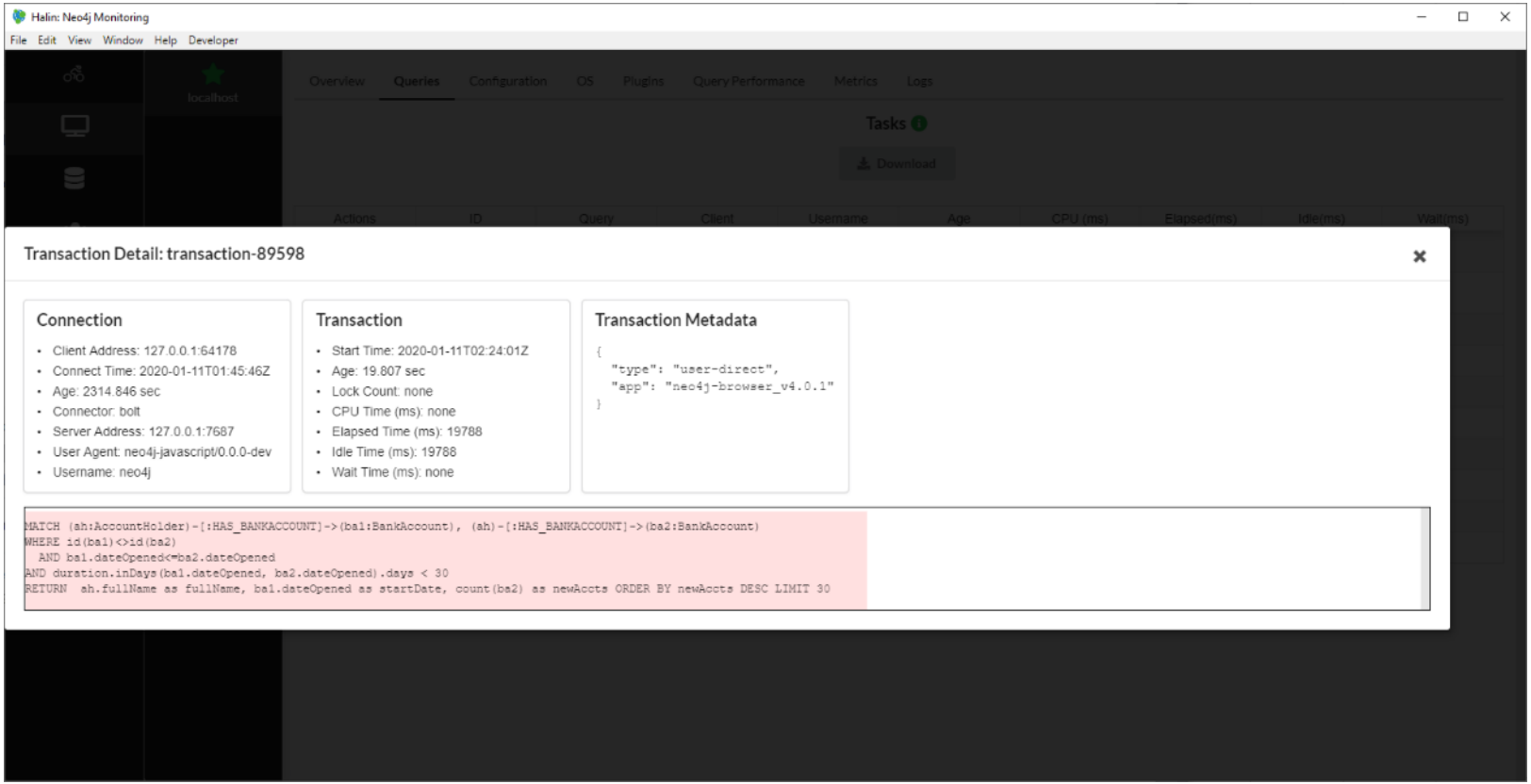
Viewing currently running queries (cypher-shell)



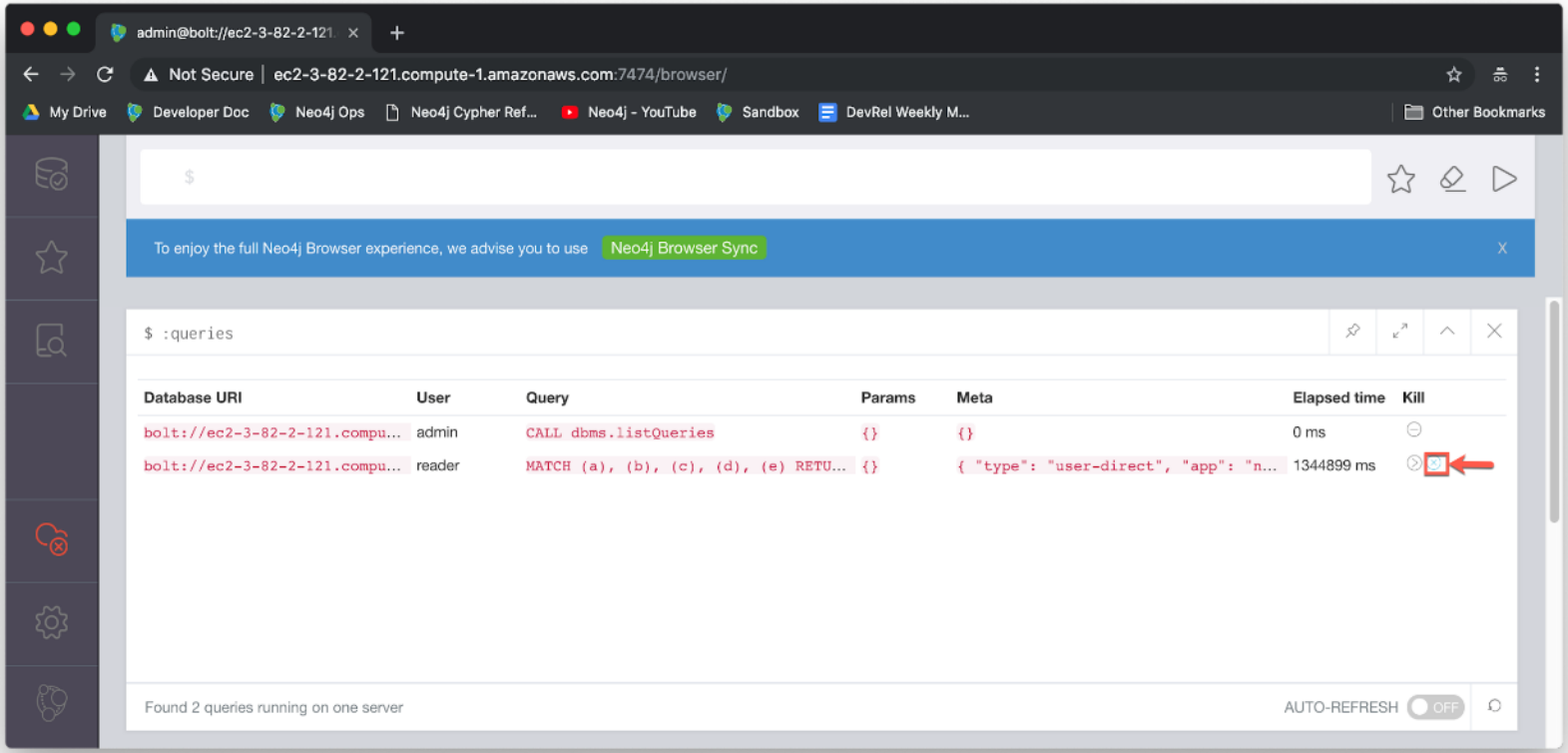
Monitoring with Halin (1)



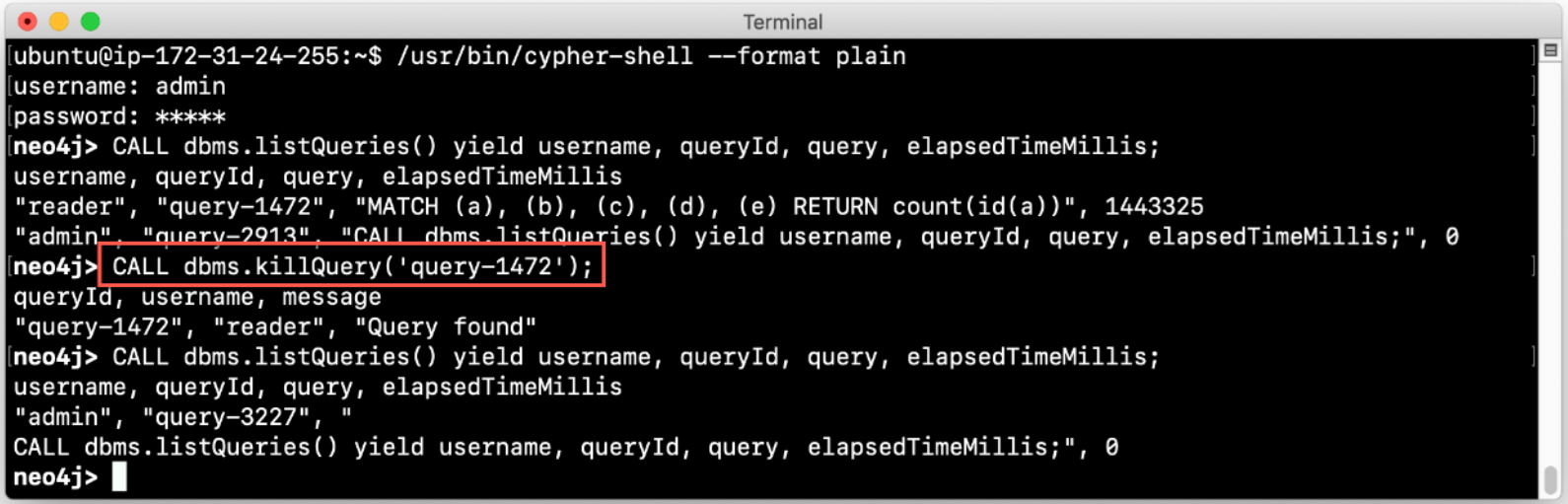
Monitoring with Halin (2)



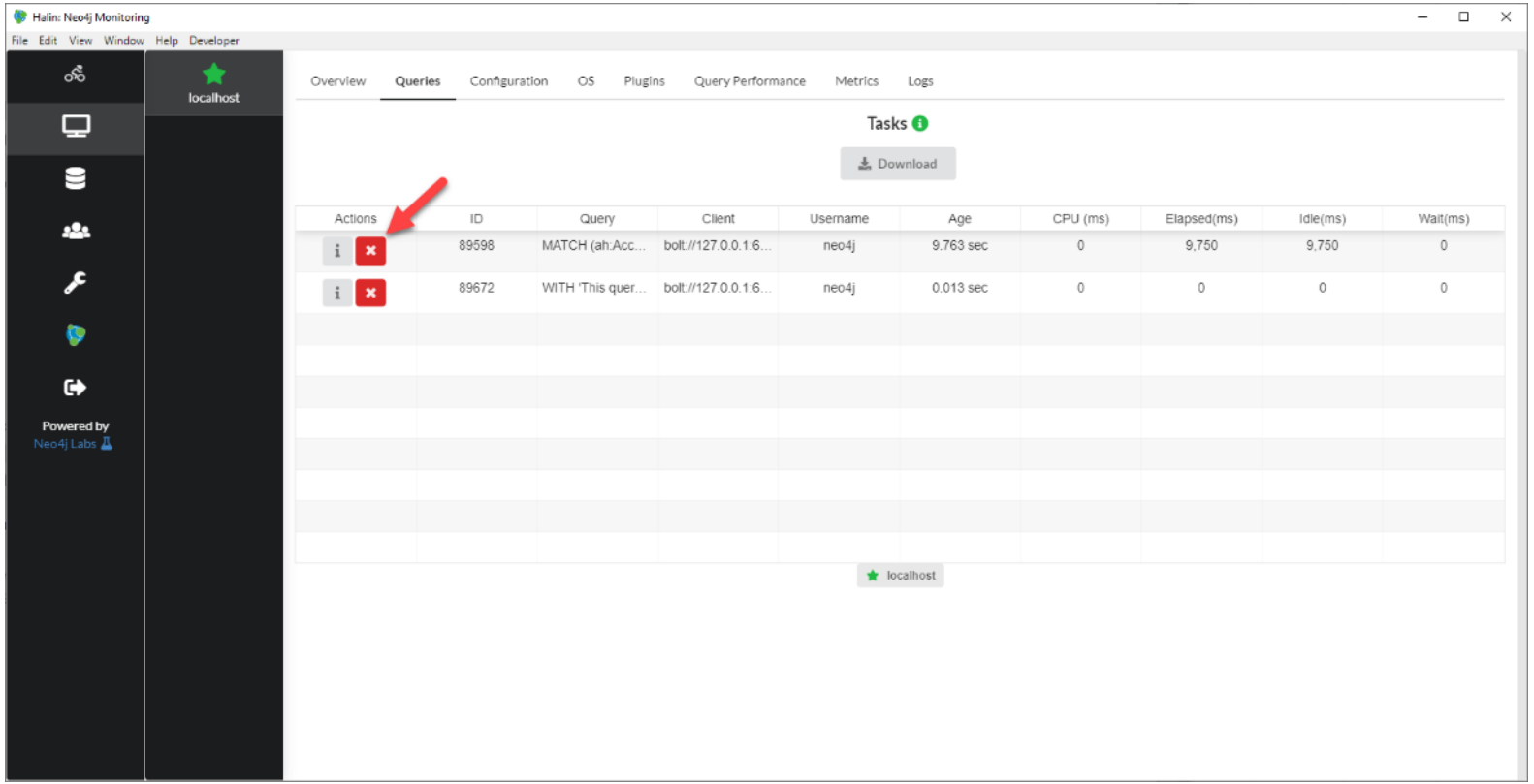
Killing a long-running query (browser)



Killing a long-running query (cypher-shell)



Killing with Halin



Query Tuning

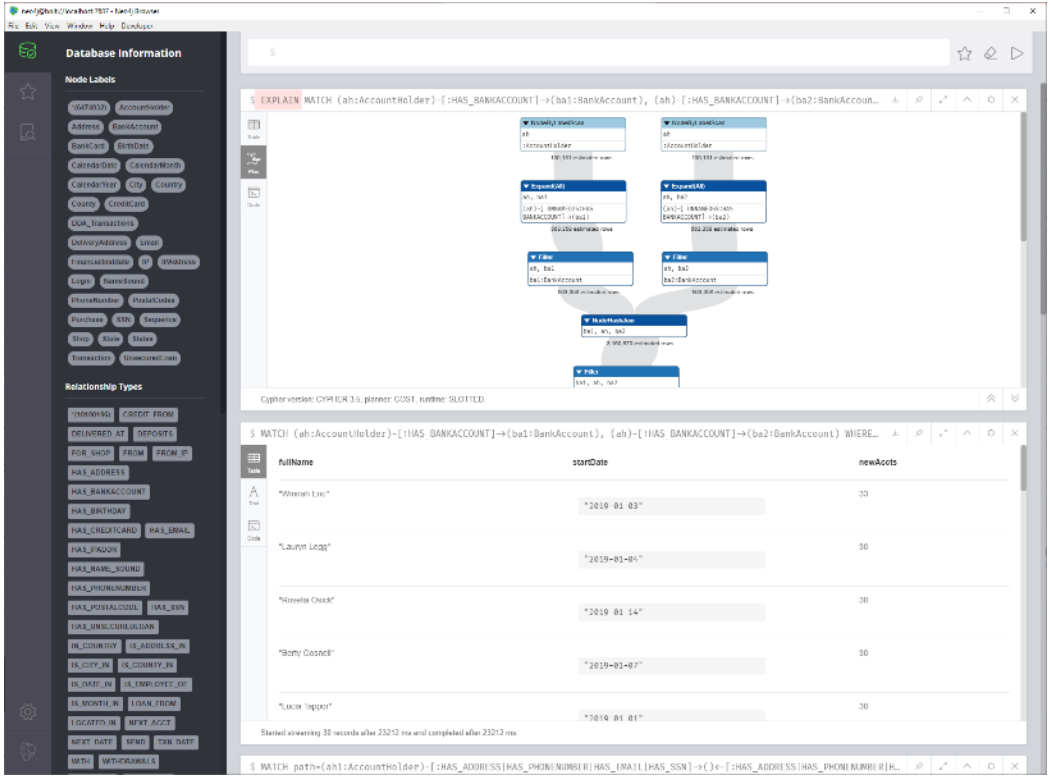
EXPLAIN <cypher>

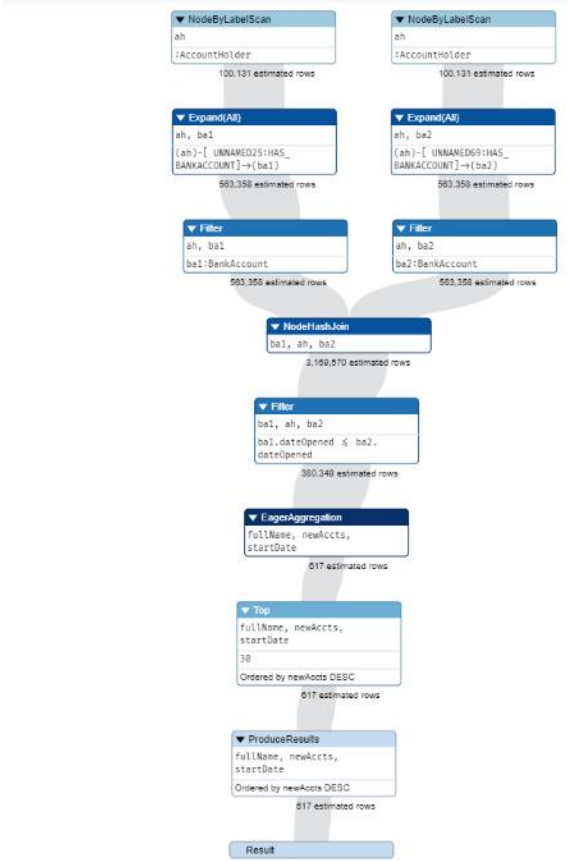
- Shows query plan without running query
- Gives estimates

PROFILE <cypher>

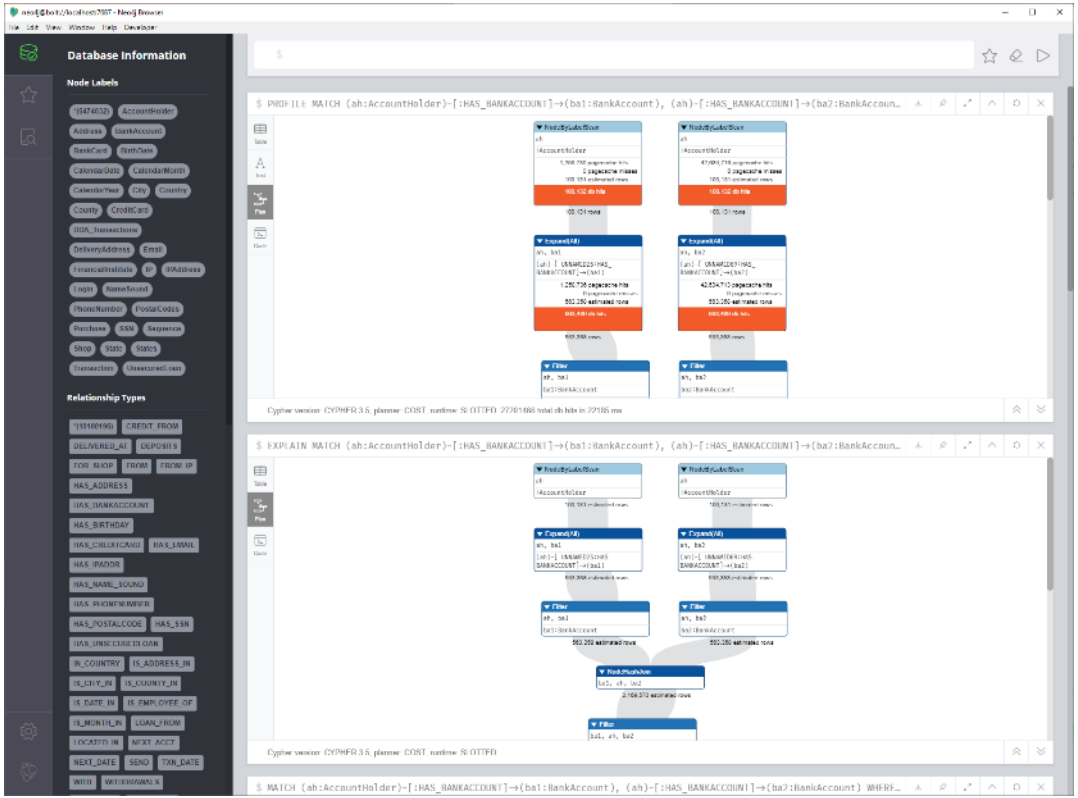
- Similar to above, but post exec
- Shows times & actuals

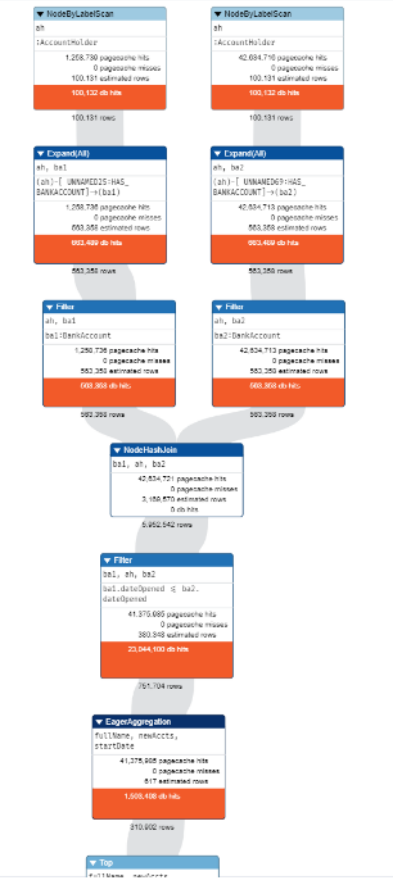
Explain output





Profile output





Comparing the two

Really, only PROFILE is needed…

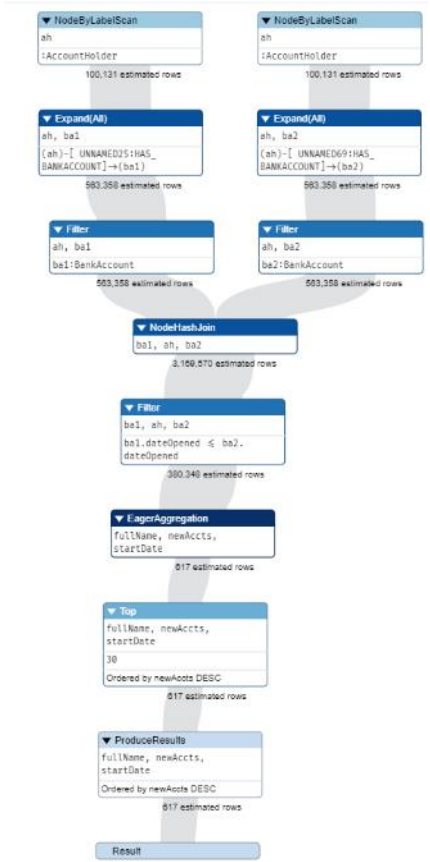
- It has both estimates & actuals

If estimates are off….

- …and index used, update the index stats

Look for

- Node label scans (when SARGS are present)
- Starting with wrong node
- Page cache misses



Collecting metrics

Metrics collection is on by default

- Careful!! They can take a LOT of space in the file system
- Metrics are related to events that are also logged

Can be viewed in Halin (!)

Since CSV, you can load in MS Excel and graph

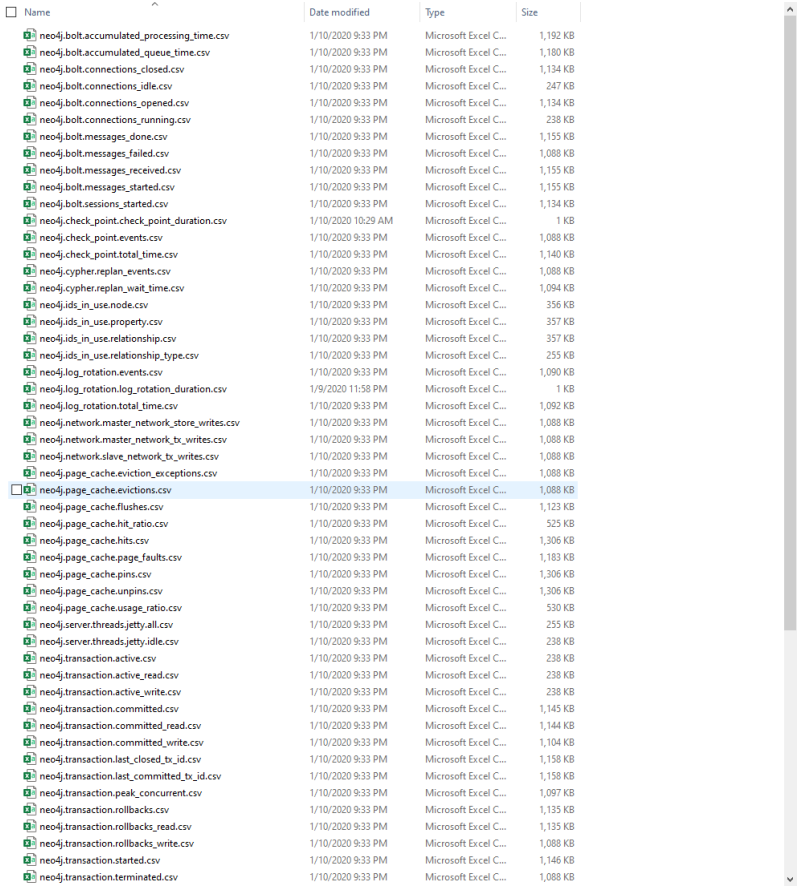
- Can allow you to overlay plots to spot trends/cause & effect (e.g. if this goes up, that goes down)

You can import into tools of your choice

- Graphite, Prometheus protocols for collecting metrics
- Works with Grafana & Nagios

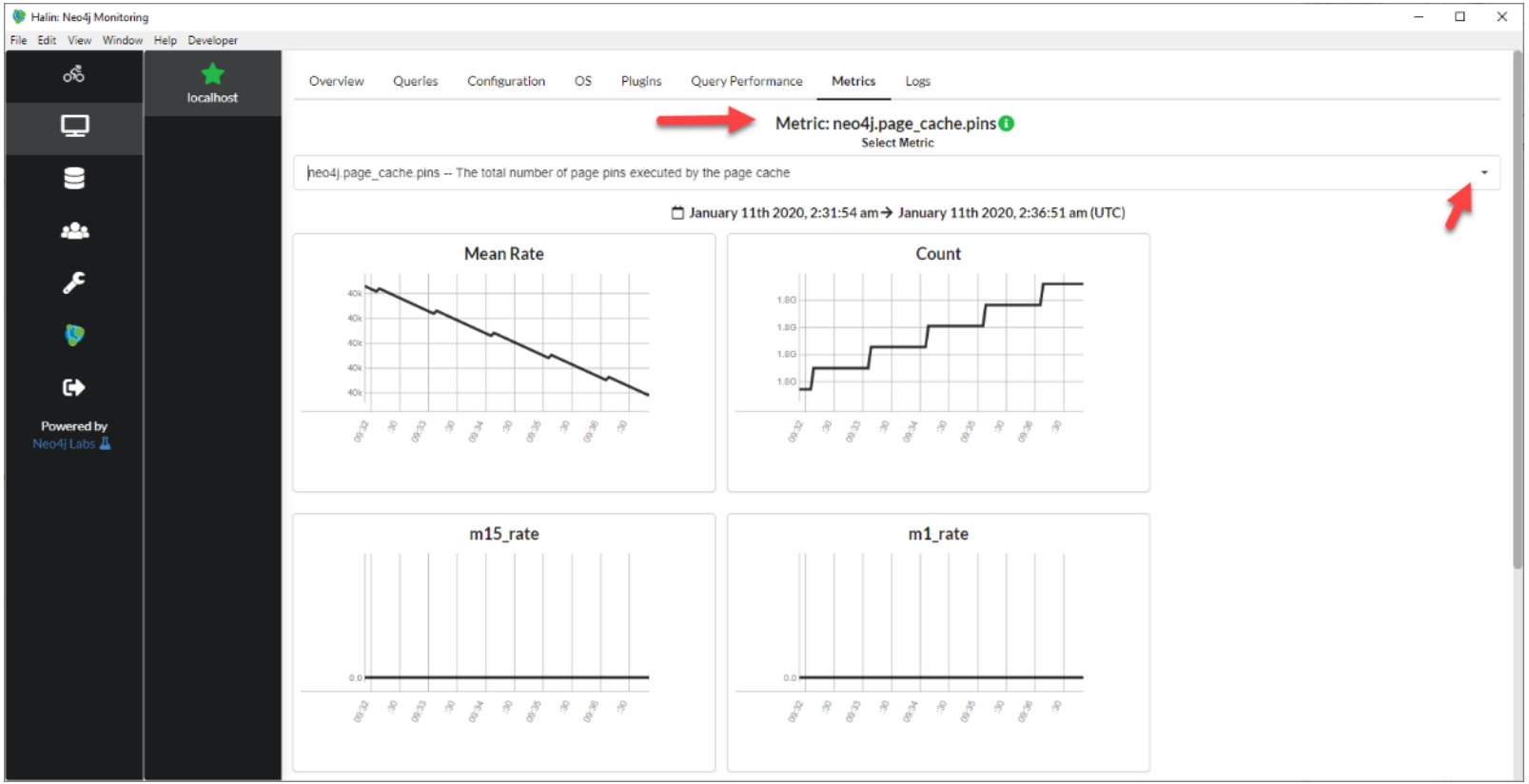
The bad news

- Server level resource stats only
- No per node/relationship stats (e.g. ## traversals/sec)
- No per user/session/query stats
- Not quite as fine grained as Oracle V$, ASE MDA, etc.



Name	Date modified	Type	Size
neo4j.bolt.accumulated_processing_time.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,192 KB
neo4j.bolt.accumulated_queue_time.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,180 KB
neo4j.bolt.connections_closed.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,134 KB
neo4j.bolt.connections_idle.csv	1/10/2020 9:33 PM	Microsoft Excel C...	247 KB
neo4j.bolt.connections_opened.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,134 KB
neo4j.bolt.connections_running.csv	1/10/2020 9:33 PM	Microsoft Excel C...	238 KB
neo4j.bolt.messages_done.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,155 KB
neo4j.bolt.messages_failed.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,088 KB
neo4j.bolt.messages_received.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,155 KB
neo4j.bolt.messages_started.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,155 KB
neo4j.bolt.sessions_started.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,134 KB
neo4j.check_point.check_point_duration.csv	1/10/2020 10:29 AM	Microsoft Excel C...	1 KB
neo4j.check_point.events.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,088 KB
neo4j.check_point.total_time.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,140 KB
neo4j.cypher.replan_events.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,088 KB
neo4j.cypher.replan_wait_time.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,094 KB
neo4j.ids_in_use.node.csv	1/10/2020 9:33 PM	Microsoft Excel C...	356 KB
neo4j.ids_in_use.property.csv	1/10/2020 9:33 PM	Microsoft Excel C...	357 KB
neo4j.ids_in_use.relationship.csv	1/10/2020 9:33 PM	Microsoft Excel C...	357 KB
neo4j.ids_in_use.relationship_type.csv	1/10/2020 9:33 PM	Microsoft Excel C...	255 KB
neo4j.log_rotation.events.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,090 KB
neo4j.log_rotation.log_rotation_duration.csv	1/9/2020 11:58 PM	Microsoft Excel C...	1 KB
neo4j.log_rotation.total_time.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,092 KB
neo4j.network.master_network_store_writes.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,088 KB
neo4j.network.master_network_tx_writes.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,088 KB
neo4j.network.slave_network_tx_writes.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,088 KB
neo4j.page_cache.eviction_exceptions.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,088 KB
neo4j.page_cache.evictions.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,088 KB
neo4j.page_cache.flushes.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,123 KB
neo4j.page_cache.hit_ratio.csv	1/10/2020 9:33 PM	Microsoft Excel C...	525 KB
neo4j.page_cache.hits.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,306 KB
neo4j.page_cache.page_faults.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,183 KB
neo4j.page_cache.pins.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,306 KB
neo4j.page_cache.unpins.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,306 KB
neo4j.page_cache.usage_ratio.csv	1/10/2020 9:33 PM	Microsoft Excel C...	530 KB
neo4j.server.threads.jetty.all.csv	1/10/2020 9:33 PM	Microsoft Excel C...	255 KB
neo4j.server.threads.jetty.idle.csv	1/10/2020 9:33 PM	Microsoft Excel C...	238 KB
neo4j.transaction.active.csv	1/10/2020 9:33 PM	Microsoft Excel C...	238 KB
neo4j.transaction.active_read.csv	1/10/2020 9:33 PM	Microsoft Excel C...	238 KB
neo4j.transaction.active_write.csv	1/10/2020 9:33 PM	Microsoft Excel C...	238 KB
neo4j.transaction.committed.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,145 KB
neo4j.transaction.committed_read.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,144 KB
neo4j.transaction.committed_write.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,104 KB
neo4j.transaction.last_closed_tx_id.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,158 KB
neo4j.transaction.last_committed_tx_id.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,158 KB
neo4j.transaction.peak_concurrent.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,097 KB
neo4j.transaction.rollbacks.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,135 KB
neo4j.transaction.rollbacks_read.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,135 KB
neo4j.transaction.rollbacks_write.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,088 KB
neo4j.transaction.started.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,146 KB
neo4j.transaction.terminated.csv	1/10/2020 9:33 PM	Microsoft Excel C...	1,088 KB

Viewing Metrics with Halin



Stress Testing App

Gatling (www.gatling.io)

- Scala/JAVA based

Tips

- Run on a different host than neo4j
- Due to java client thread scaling in single JVM, use multiple Gatling instances
  - ✓ *Put different instances on different hosts*

Caveats

- There are limits – e.g. if you saturate a client NIC and server NIC is same size, adding multiple instances is not going to help

Loading Data into Neo4j

neo4j

Loading Data

Bulk loader

- Rarely used - Limited – e.g. can load into a single node/label – no merging, etc.
- Relationships have to be created after the fact via cypher

Cypher

- Obvious CREATE/MERGE commands
- More typical is LOAD CSV (from $NEO4J_HOME/import or http)
  - ✓ *Default/Implied syntax is CREATE*
  - ✓ *Mostly used with MERGE as most of the time the CSV file is loading more than one node/label*
  - ✓ *Allows relationships to be created at load time*

ETL – (Kettle)

- Non-Kettle (Informatica, etc.) → can get data to the doorstep (final step is generate CSV)
- Kettle
  - ✓ *Supported by neo4j – Integrated with neo4j – leverages cypher (load csv/create/merge)*

Streaming

- Kafka connector

LOAD CSV Example - CREATE

```
load csv with headers from 'file:///funds.csv' as line
create (:Fund {
    fund_name:          line.fund_name,
    ticker:             line.ticker,
    assets:             line.assets,
    manager:            line.manager,
    inception_date:     date(line.inception_date),
    company:            line.company,
    expense_ratio:      toFloat(line.expense_ratio)
 })
```

LOAD CSV Example - MERGE

```
USING PERIODIC COMMIT 1000
LOAD CSV WITH HEADERS FROM 'file:///customer.csv' AS input
MATCH (myZip:PostalCodes {postalCode: input.postal_code})
MATCH (myCity:City)-[:HAS_POSTALCODE]->(myZip)
MATCH (myState:States {stateName:input.state})
MATCH (myBank:FinancialInstitute {name: input.bank_name})
MERGE (myPhone:PhoneNumber {phone:input.phone_number})
                ON CREATE SET myPhone.provider="Verizon"
MERGE (mySSN:SSN {ssn: input.ssn})
MERGE (myStreet:Address {streetAddress: input.street_address, city: input.city, state: input.state, zip: input.postal_code})
MERGE (myStreet)-[:IS_ADDRESS_IN]->(myZip)
MERGE (myEmail:Email {email: input.email_address})
MERGE (myDOB:BirthDate {birthDate: input.birthdate})
MERGE (myIP:IPAddress {ipAddress: input.ip_address})
MERGE (ah:AccountHolder {UniqueId: input.customer_id})
                ON CREATE SET
                ah.lastName=input.last_name,
                                ah.firstName=input.first_name,
                                ah.fullName=input.first_name+" "+input.last_name,
                                ah.birthDate=input.birthdate,
                                ah.gender=input.gender
MERGE (ah)-[:HAS_PHONENUMBER]->(myPhone)
MERGE (ah)-[:HAS_SSN]->(mySSN)
MERGE (ah)-[:HAS_ADDRESS]->(myStreet)
MERGE (ah)-[:HAS_EMAIL]->(myEmail)
MERGE (ah)-[:HAS_BIRTHDAY]->(myDOB)
MERGE (ah)-[:HAS_IPADDR]->(myIP)
MERGE (myAcct:BankAccount {accountNumber: input.account_number})
                ON CREATE SET
                                myAcct.balance=toFloat(input.account_balance),
                                myAcct.dateOpened=date(input.account_date)
MERGE (ah)-[:HAS_BANKACCOUNT]->(myAcct)
MERGE (myAcct)-[:FROM]->(myBank)
```

Can be really slow!!!!
...and cause a lot of GC!!!!

LOAD CSV Example – Optimal (Multi-Step)

```
// Step 1 – Load the data – if we know this is “new” data, we can just call CREATE vs. MERGE
// Added 1500000 labels, created 1500000 nodes, set 7500000 properties, completed after 59321 ms.
USING PERIODIC COMMIT 1000
LOAD CSV WITH HEADERS FROM 'file:///atm_txns.csv' AS input
CREATE (:DDA_Transactions {
                transaction_id:                         input.transaction_id,
                transaction_date:                       input.transaction_date,
                amount:                                 toFloat(input.amount),
                accountNumber:                          input.account_number,
                type:                                   "DEPOSIT"
})


// Step 2 – Do the relationships by rescanning the file
USING PERIODIC COMMIT 1000
LOAD CSV WITH HEADERS FROM 'file:///atm_txns.csv' AS input
MATCH (myDate:CalendarDate {date: date(input.transaction_date)})
MATCH (myTxn:DDA_Transactions {transaction_id: input.transaction_id})
MATCH (myAcct:BankAccount {accountNumber: input.account_number})
MERGE (myTxn)-[:TXN_DATE]->(myDate)
MERGE (myAcct)-[:DEPOSITS]->(txn)
```

Much, much, much faster!!

Using ETL with Neo4j

Enterprise ETL don't support most no SQL

- Primarily just SQL and Hadoop

Kettle → OpenSource from Pentaho

- Supports Neo4j
- Also supports SQL, Hadoop/Hbase, Streaming, Mongo, etc.
- Google 'neo4j kettle remix' (or www.ibridge.be)

Deployment model

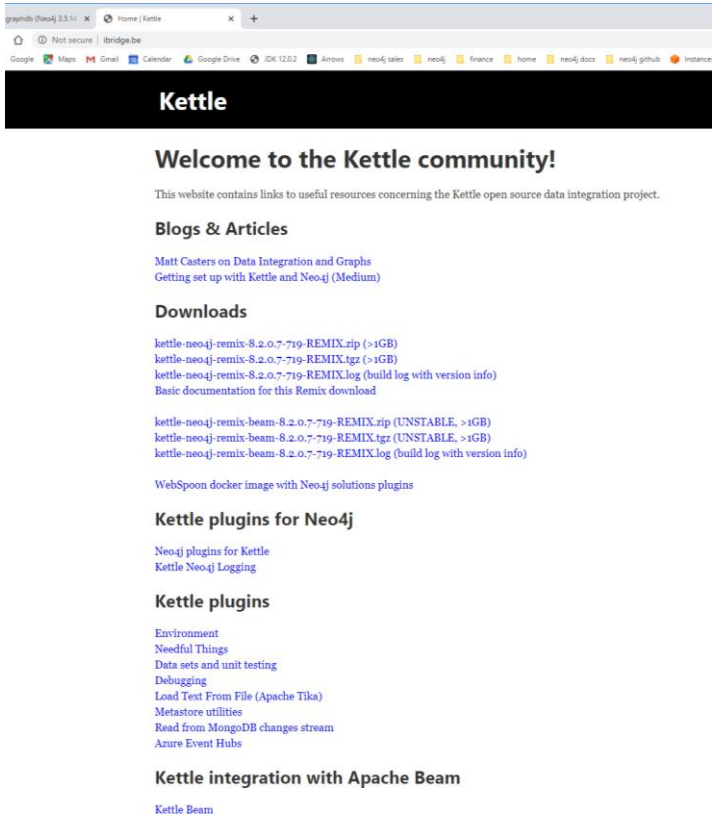
- Use existing ETL to get data to the "last mile"
- E.g. at end of existing ETL, create CSV files to load into neo4j
- Then use Kettle to pickup and load data into neo4j

Both a command line and GUI

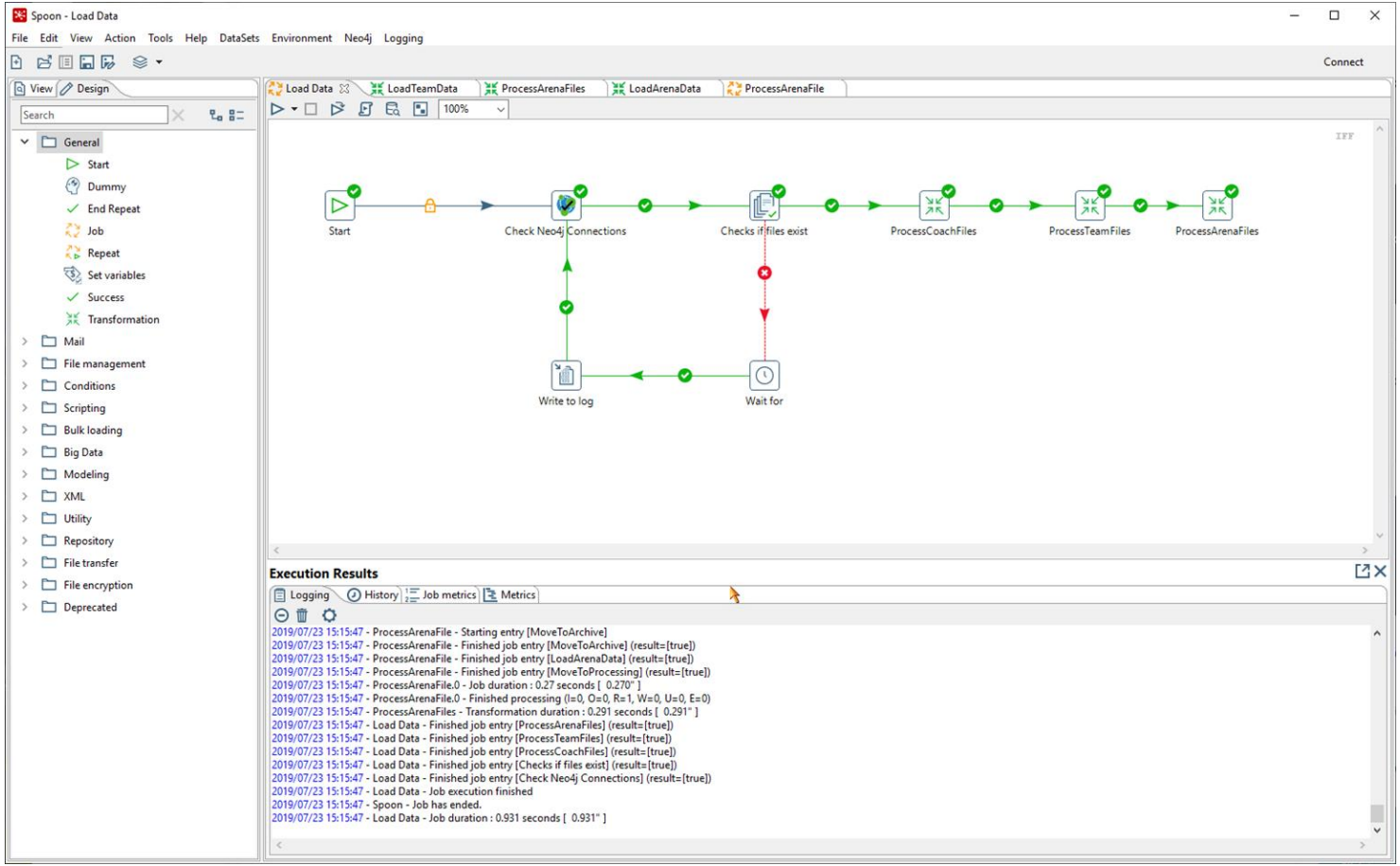
- Create jobs in GUI
- Run jobs (typically) via command line via scheduler, etc.

Is it supported by neo4j?

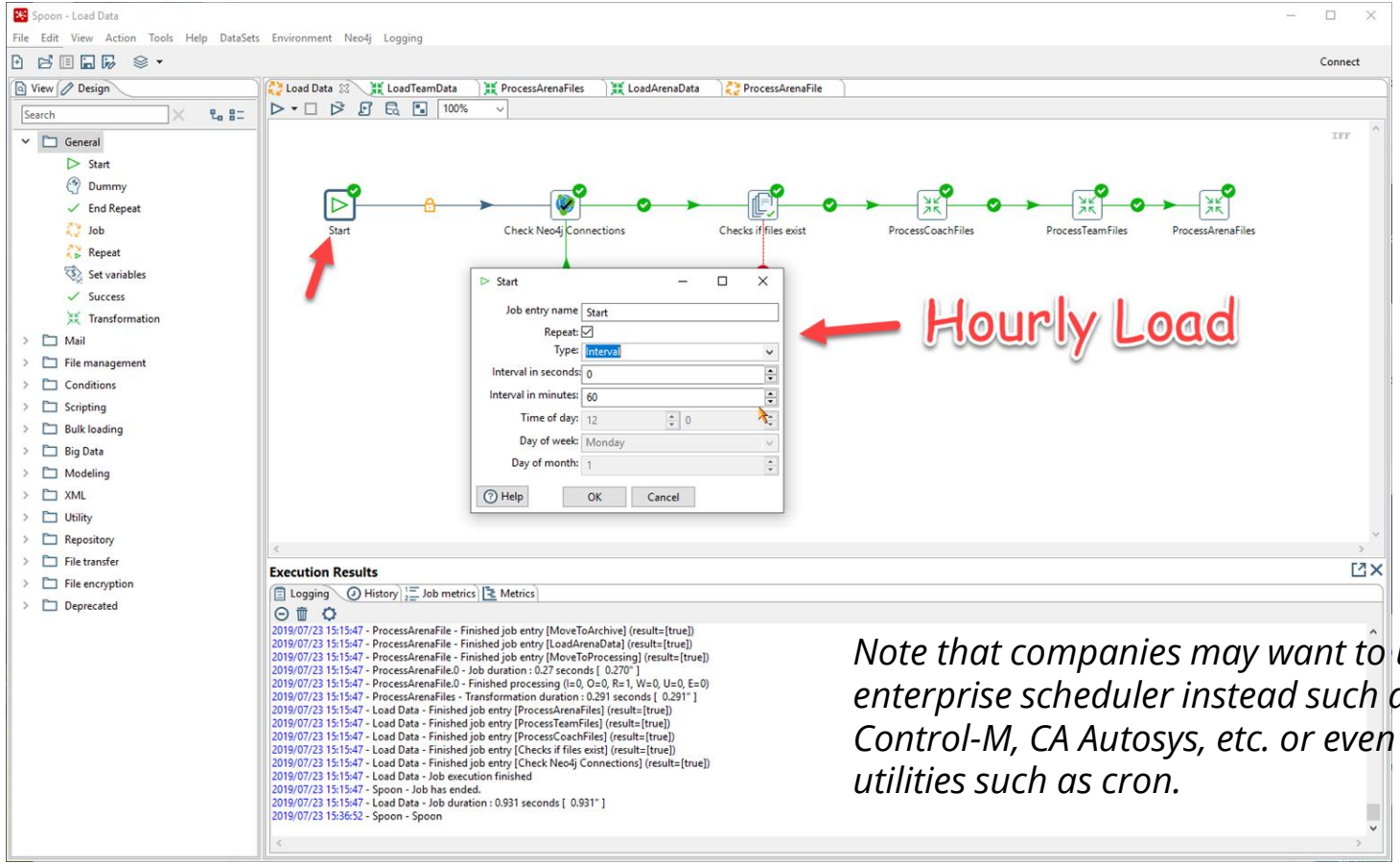
- You can use it for free with no support
- If you want neo4j support (e.g. call for bugs), then extra support fee per year



Initial & Incremental Load

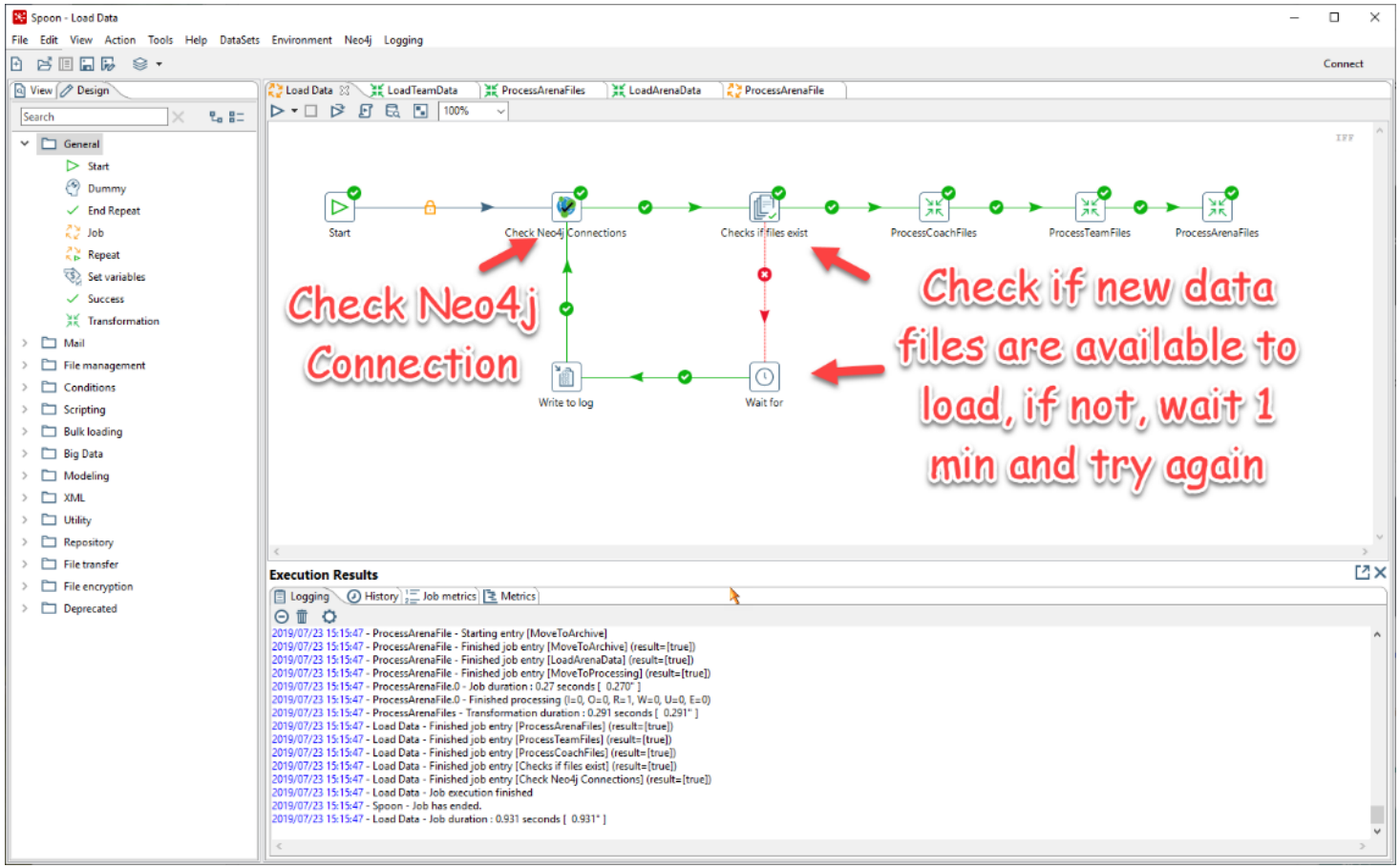


Interval, Batch Size, File Size, Etc.

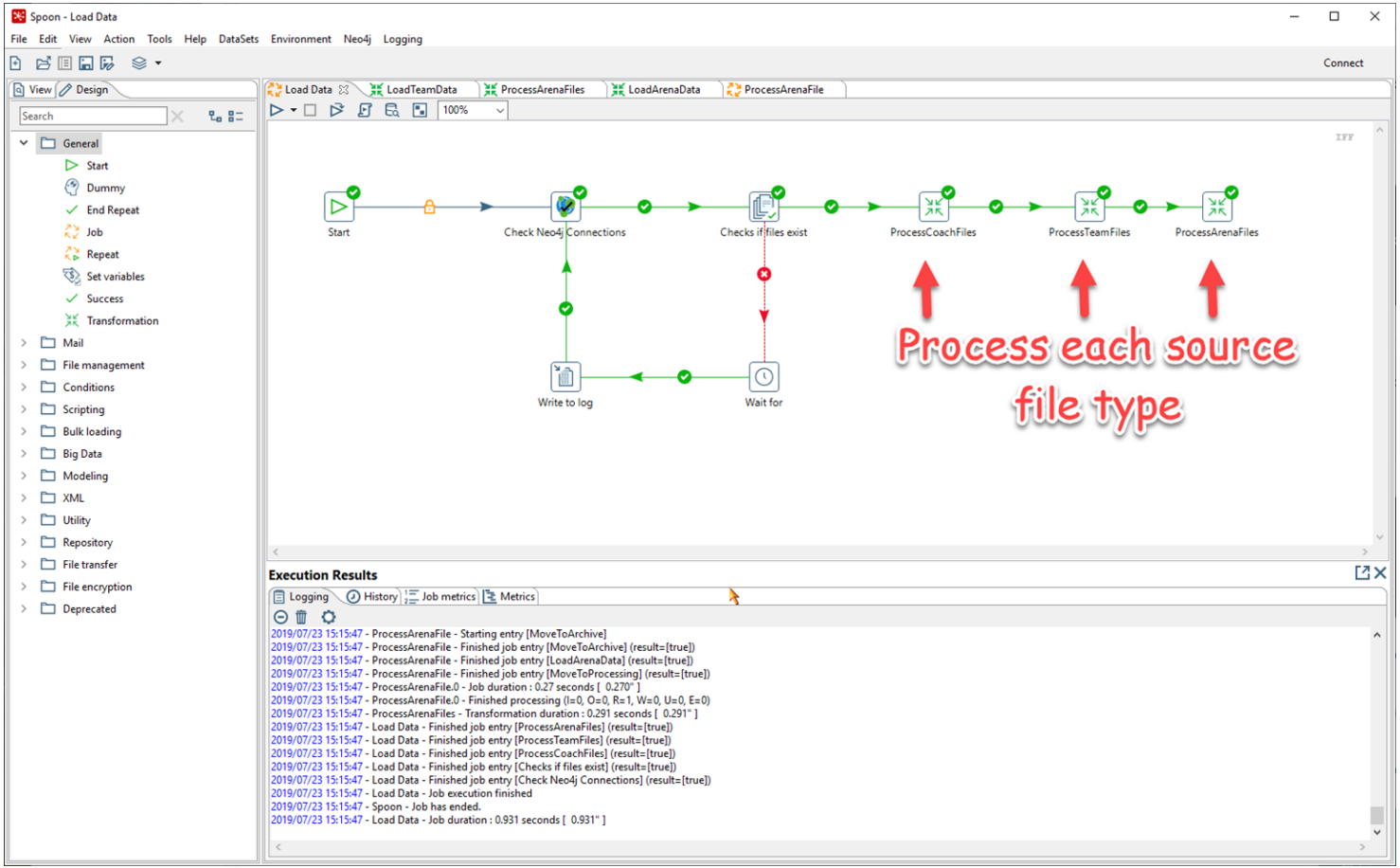


Note that companies may want to use an enterprise scheduler instead such as BMC Control-M, CA Autosys, etc. or even OS utilities such as cron.

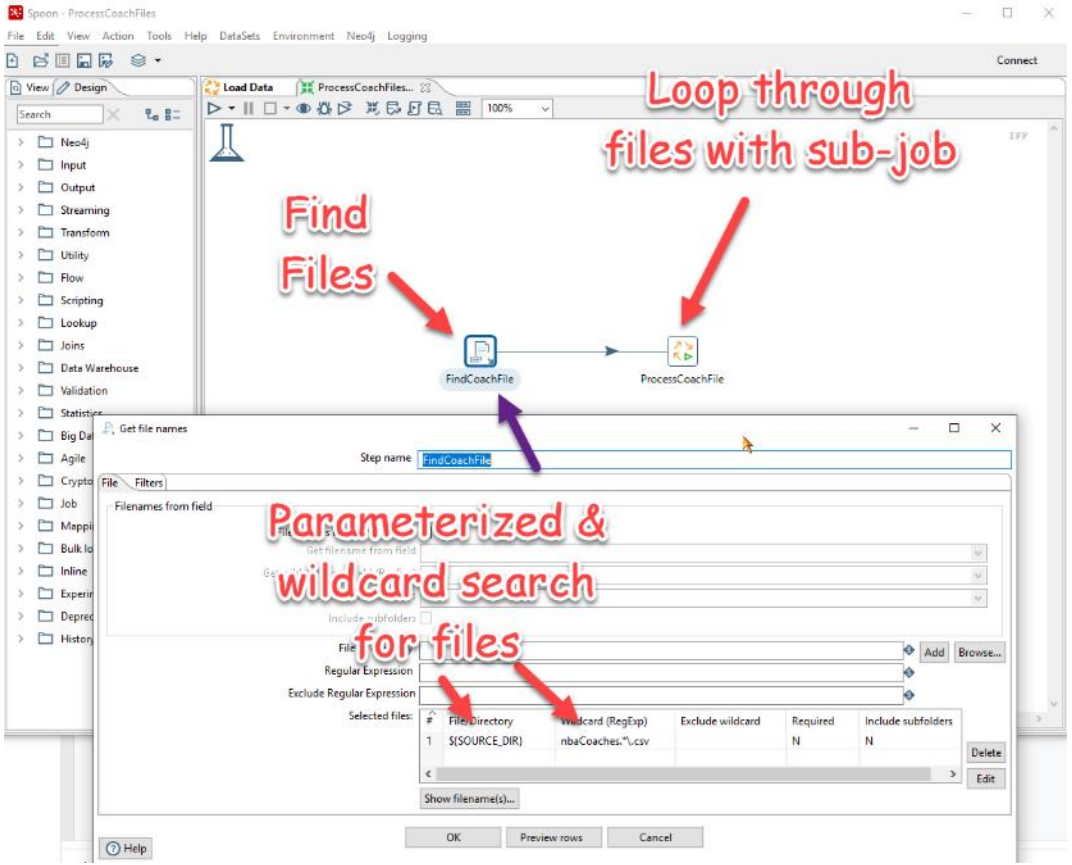
Sanity Checks for Go/No Go



Process Multiple Sources

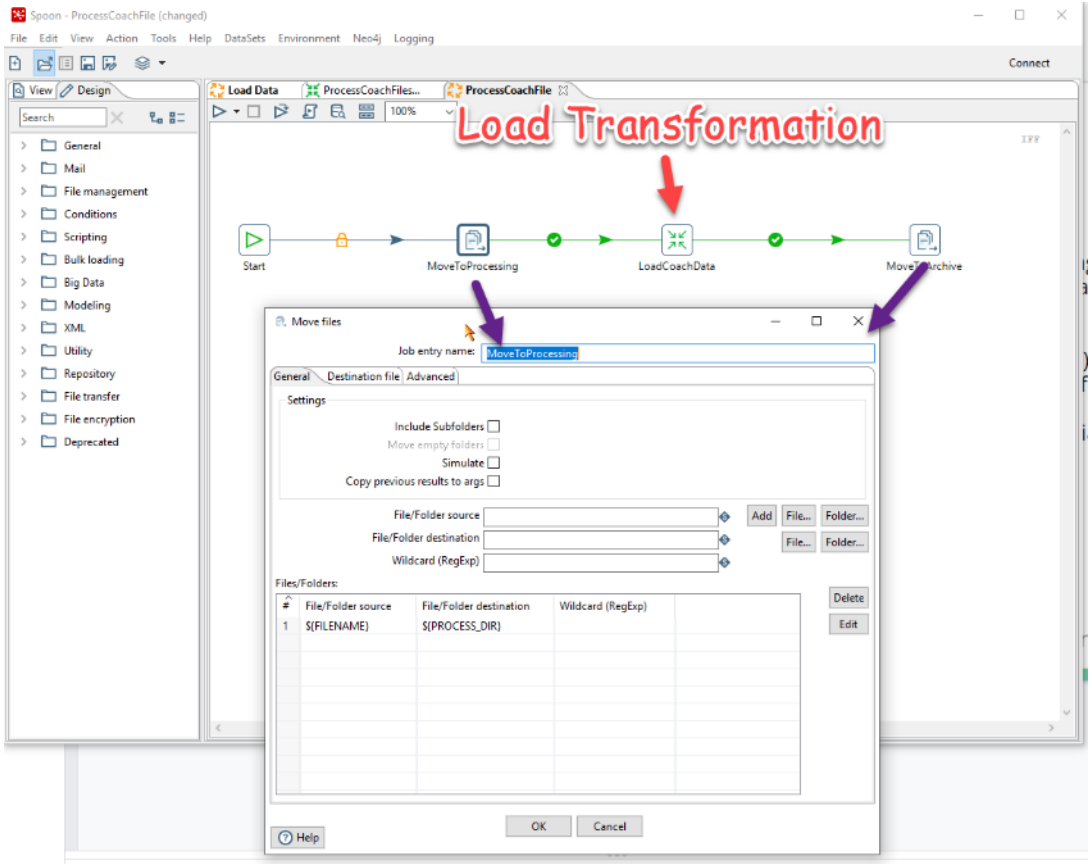


Sub-Processing Source/Type



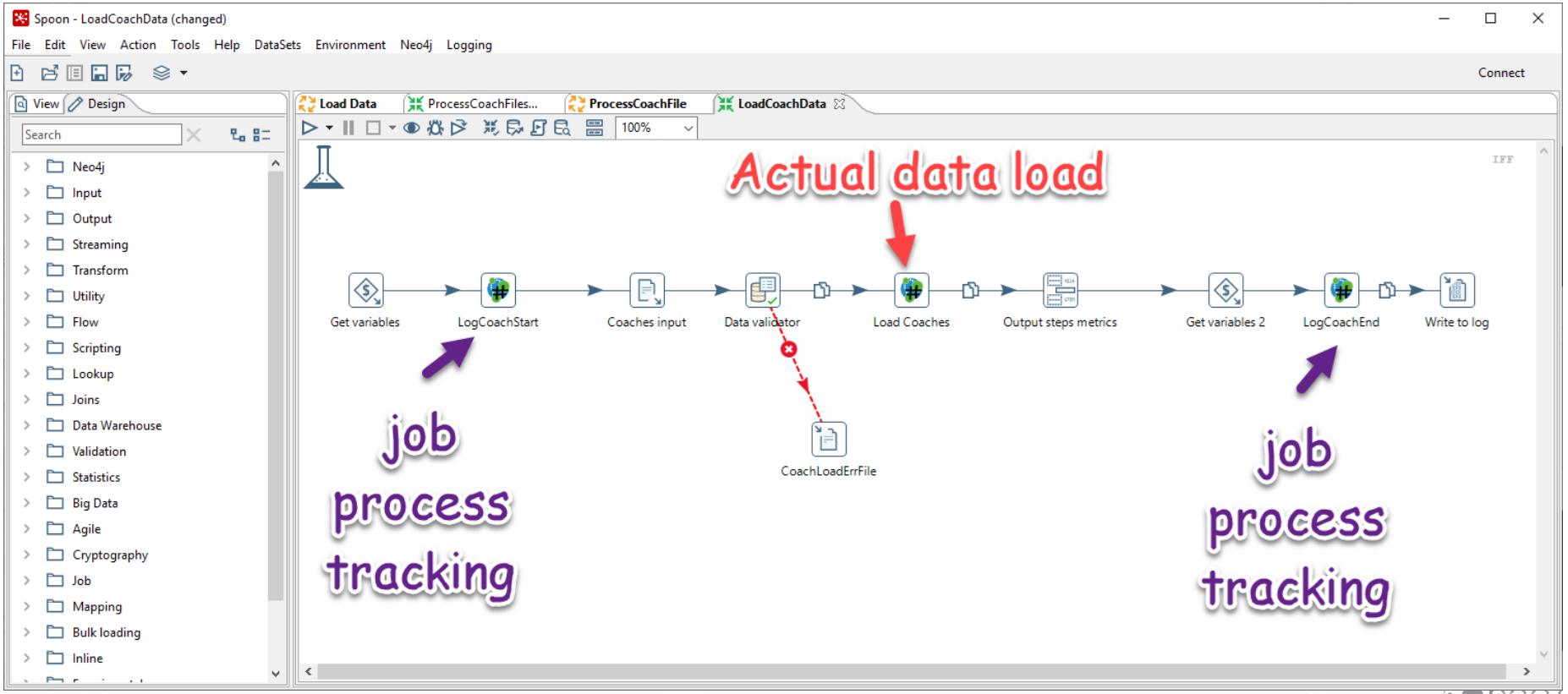
Due to external job processing frequency or issues, there may be more than one file (e.g. filename with timestamp ala mydata_yyyymmdd_hhmmss). To handle this, we get a list of files of the same format and process them one at a time via a subjob

Sub-Process from Source to Staging

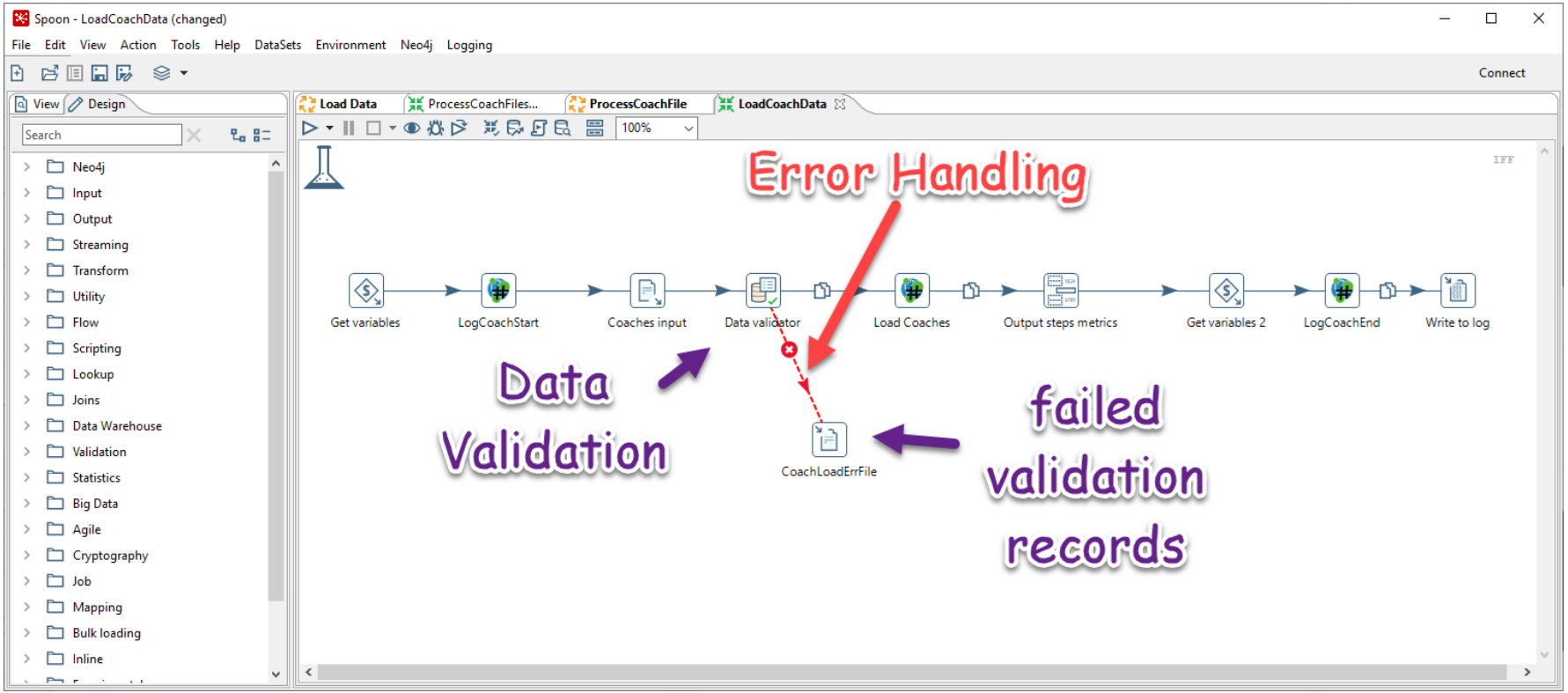


For a robust processing, we move the files from the source directory to a processing directory, execute a load transformation and then move the files to an archive folder

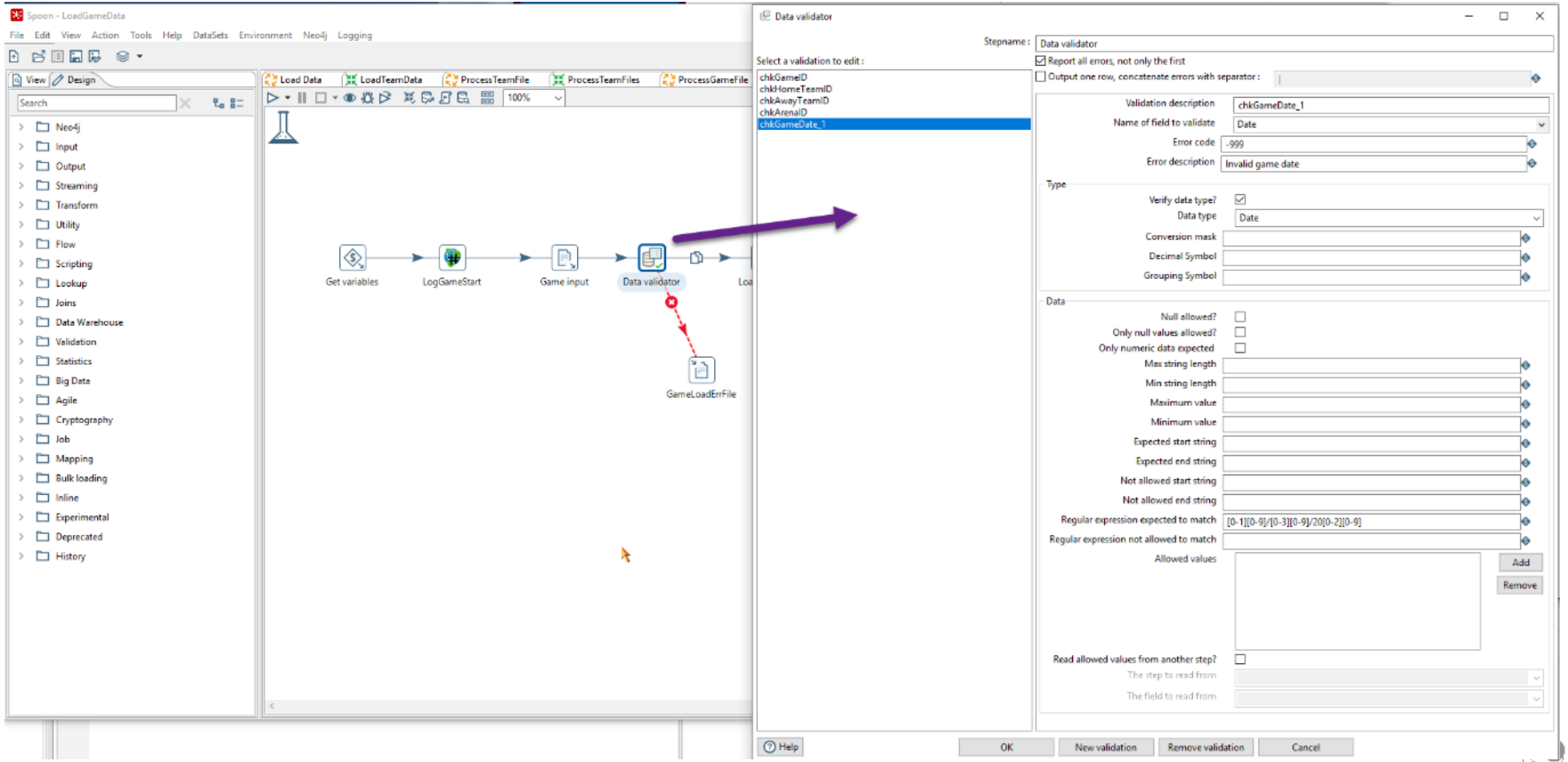
Load Transformation Metrics/Tracking



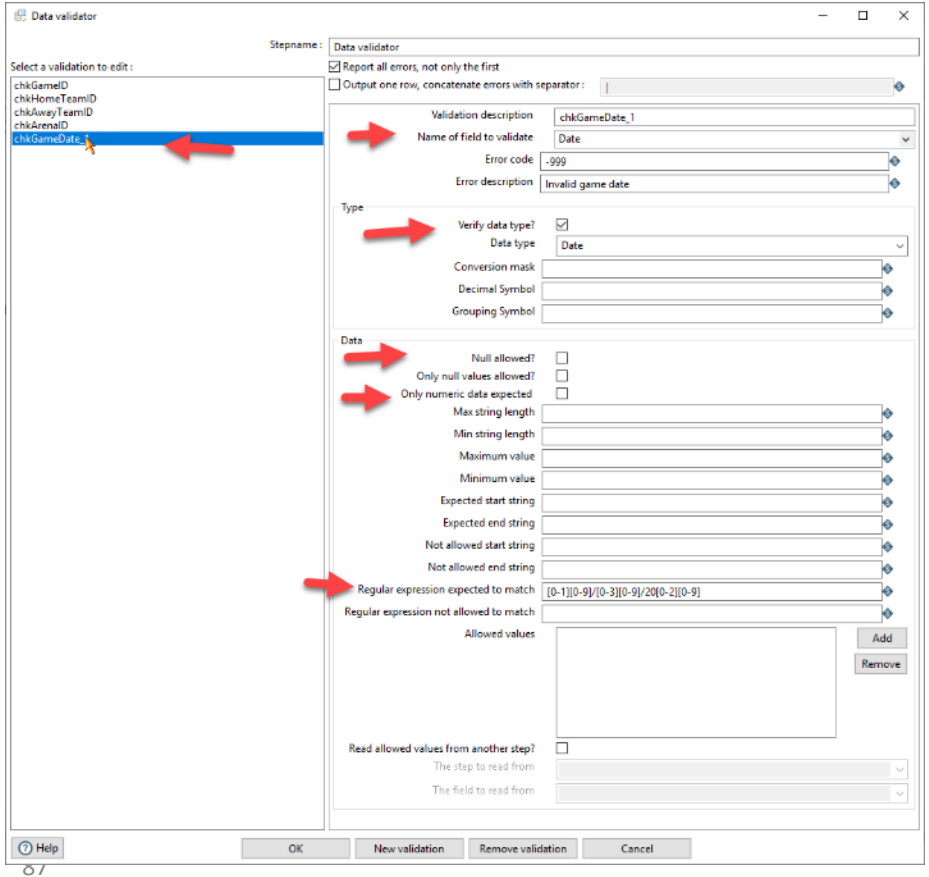
Data Validation & Error Handling

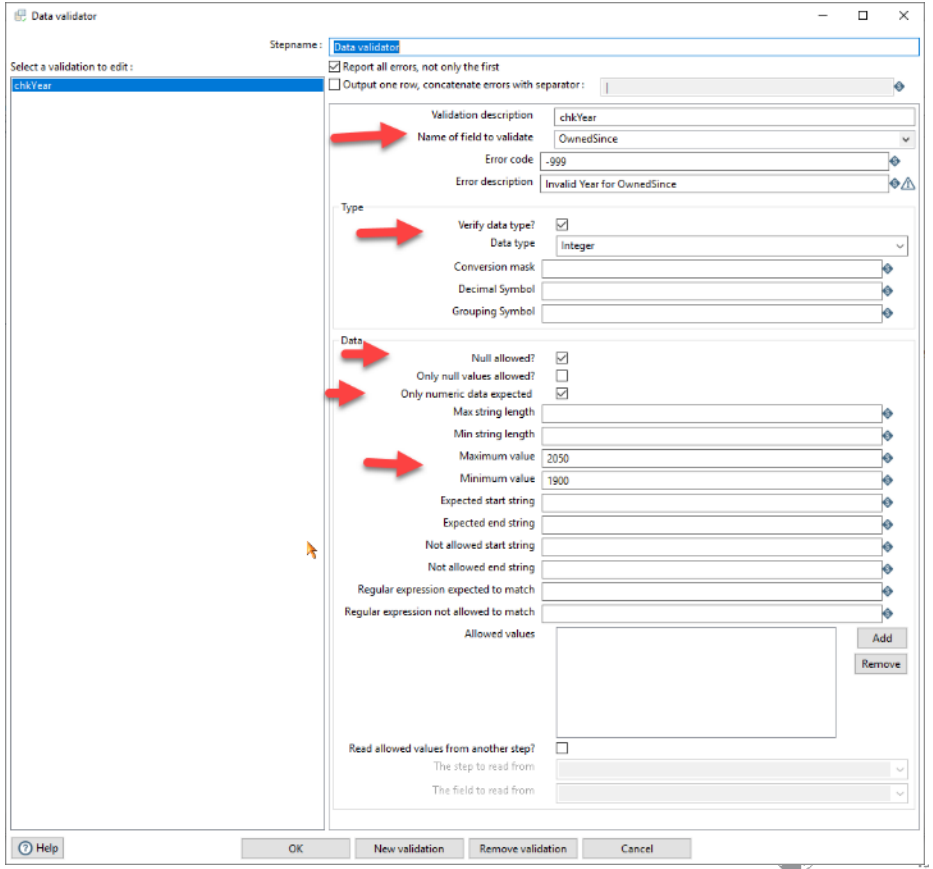


Data Validation Step

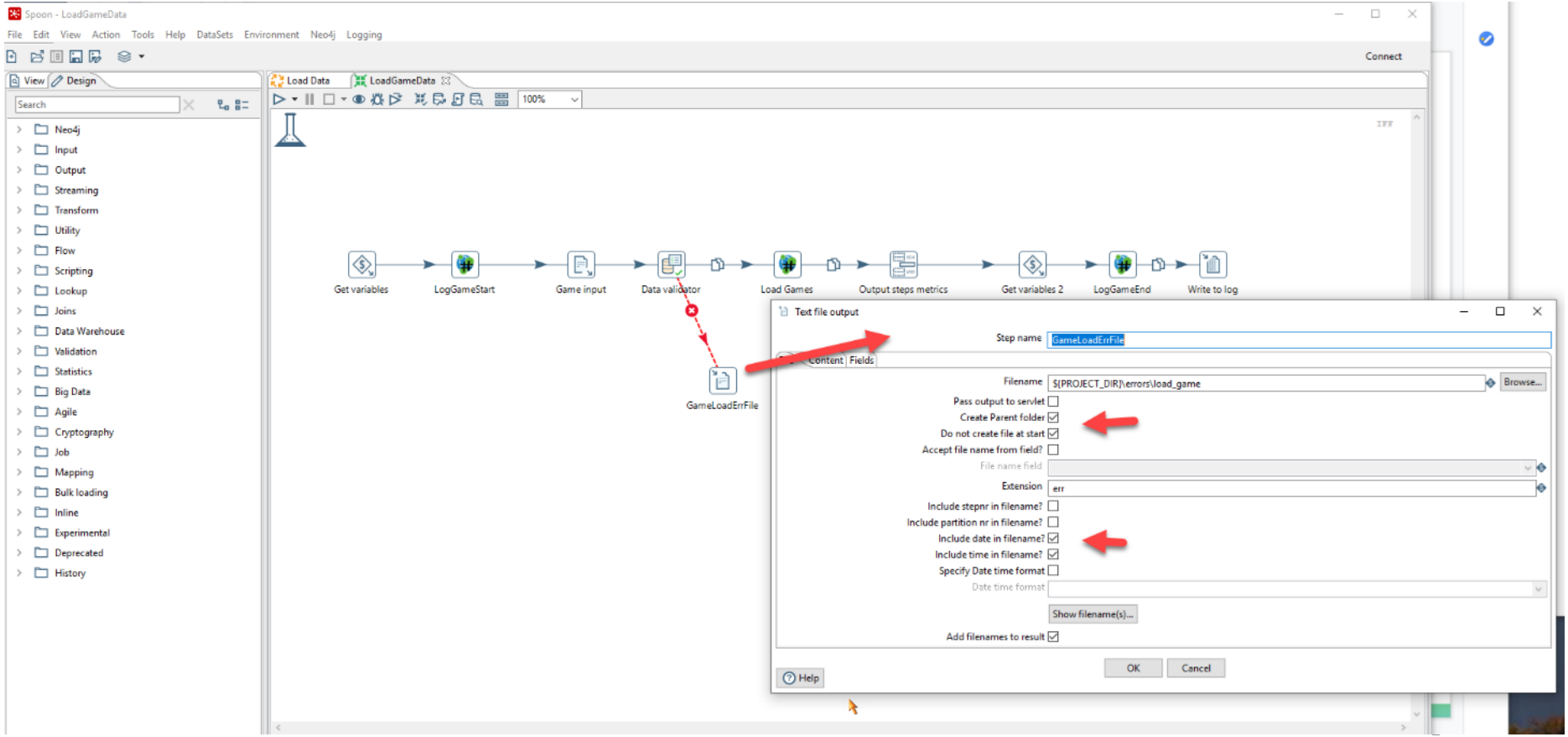


Data Validation Examples

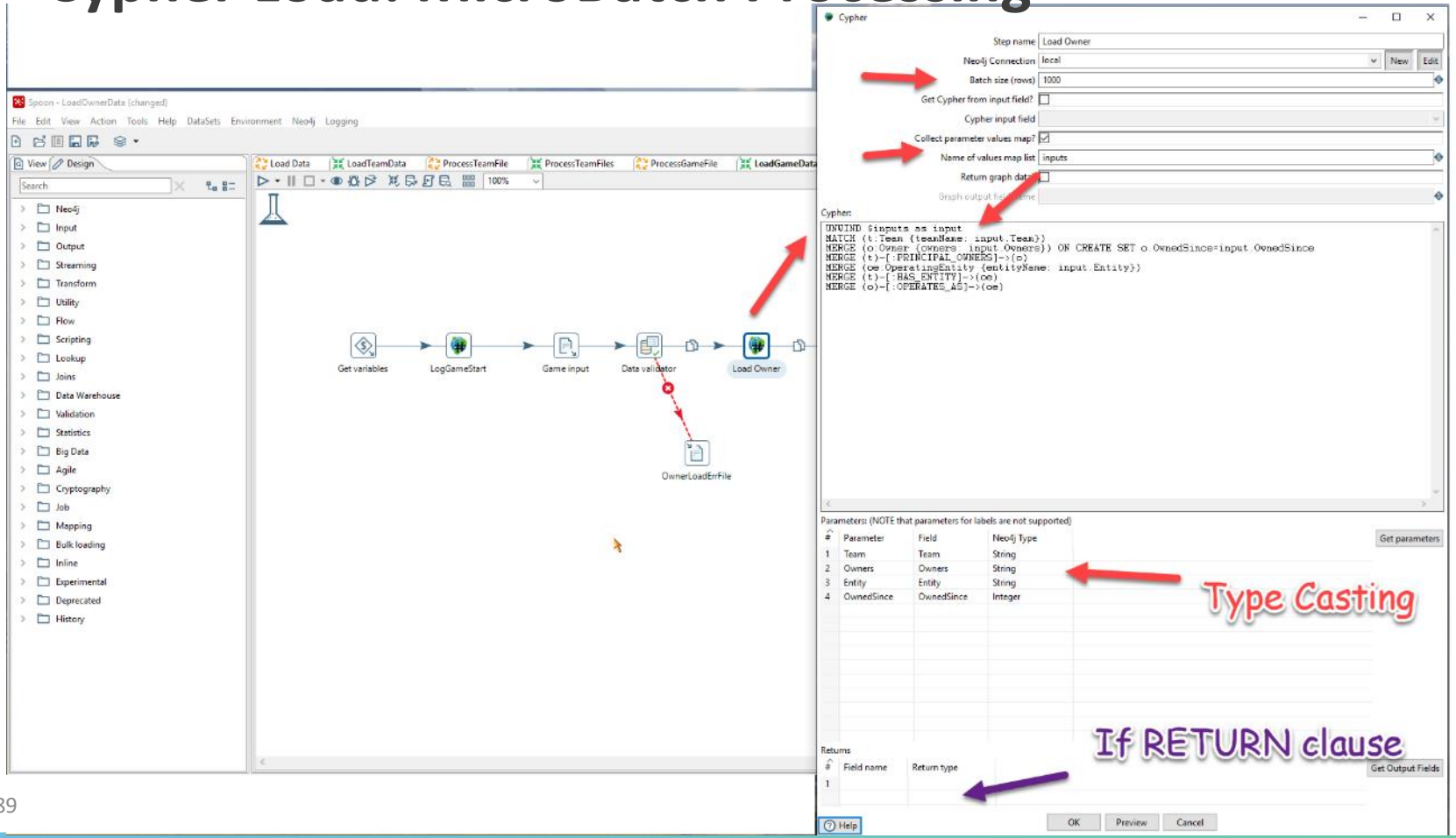




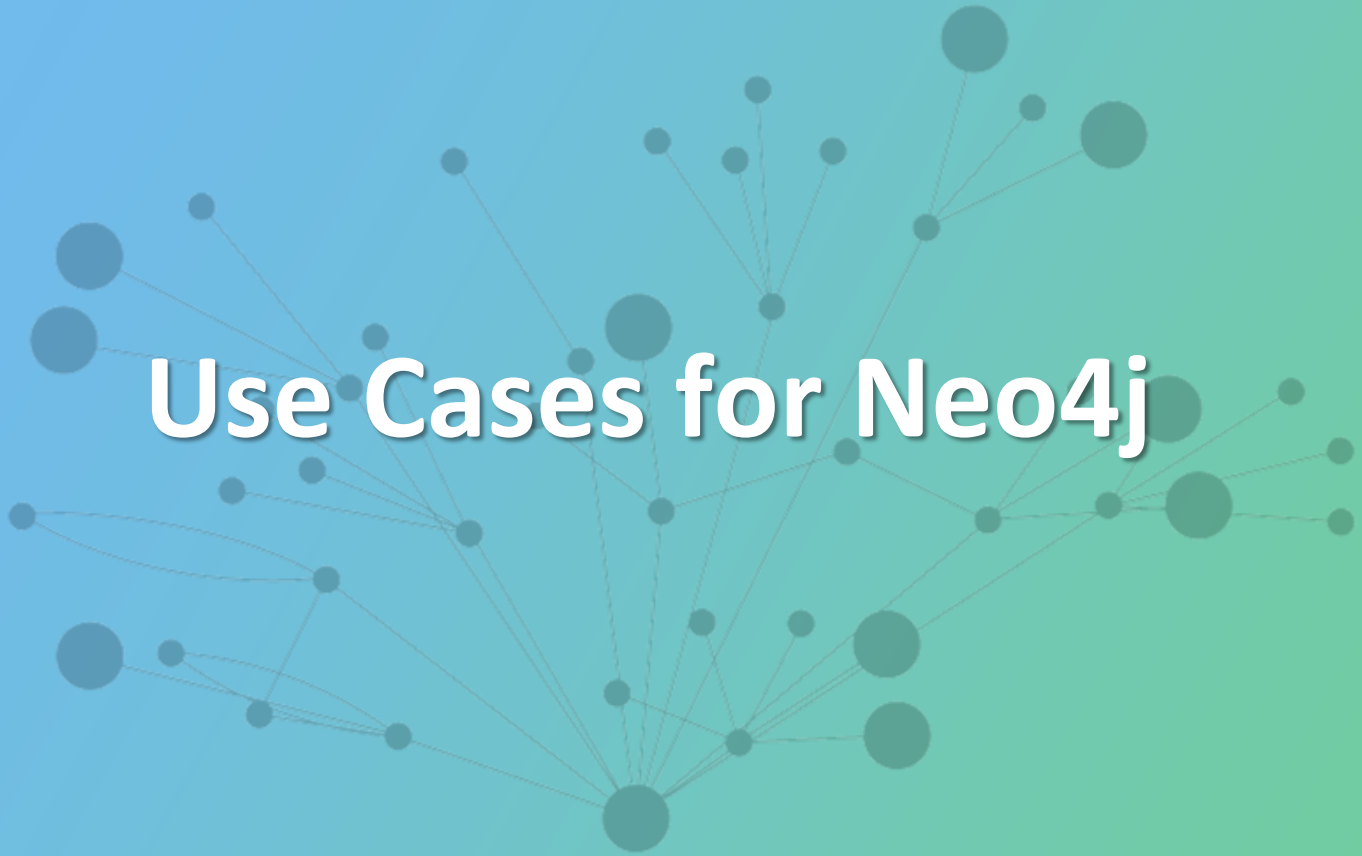
Logging Bad Data



Cypher Load: MicroBatch Processing



Use Cases for Neo4j

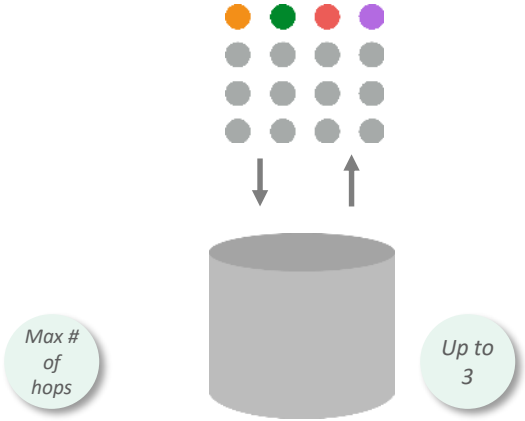


neo4j

IT Portfolio Perspective

TRADITIONAL
OLTP/RDBMS

Store and retrieve data

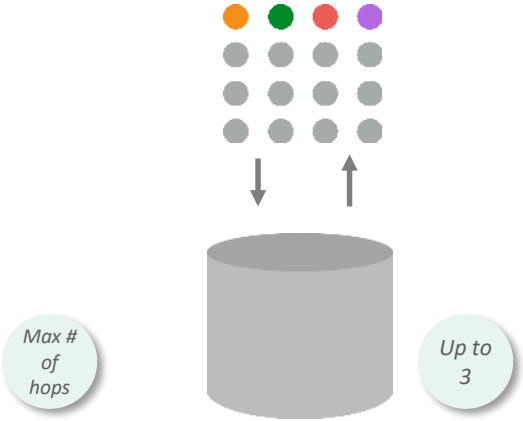


Real time storage & retrieval

IT Portfolio Perspective

TRADITIONAL OTLP/DBMS

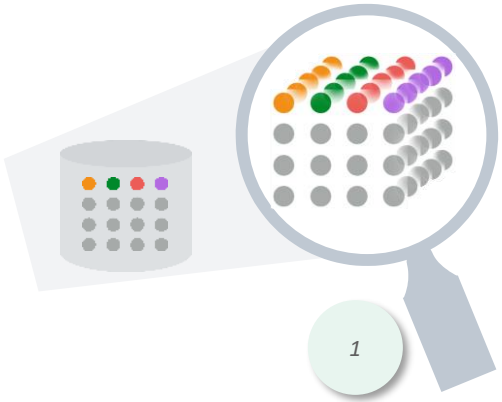
Store and retrieve data



Real time storage & retrieval

OLAP & COLUMN DBMS

Aggregate and filter data



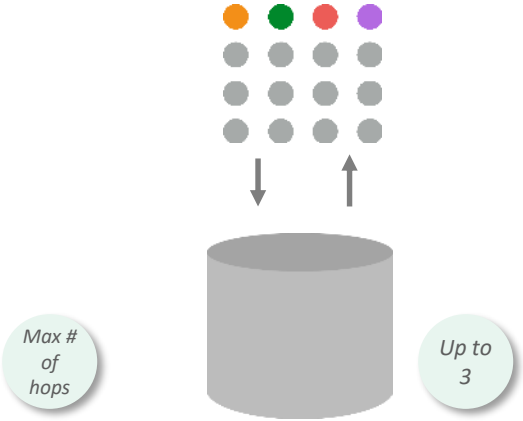
Long running queries
Aggregation & filtering

Started with Multi-Dimensional DB
Stabilized on Column Store DB

IT Portfolio Perspective

TRADITIONAL OTLP/DBMS

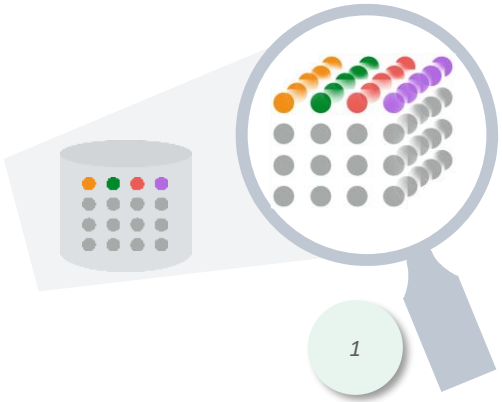
Store and retrieve data



Real time storage & retrieval

OLAP & COLUMN DBMS

Aggregate and filter data

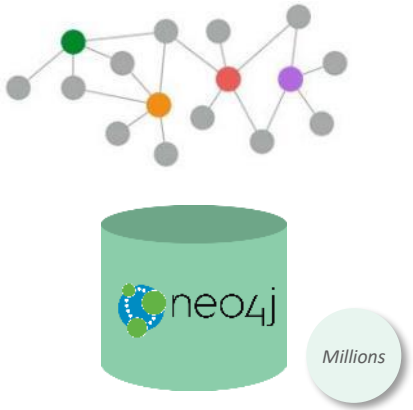


Long running queries
Aggregation & filtering

Started with Multi-Dimensional DB
Stabilized on Column Store DB

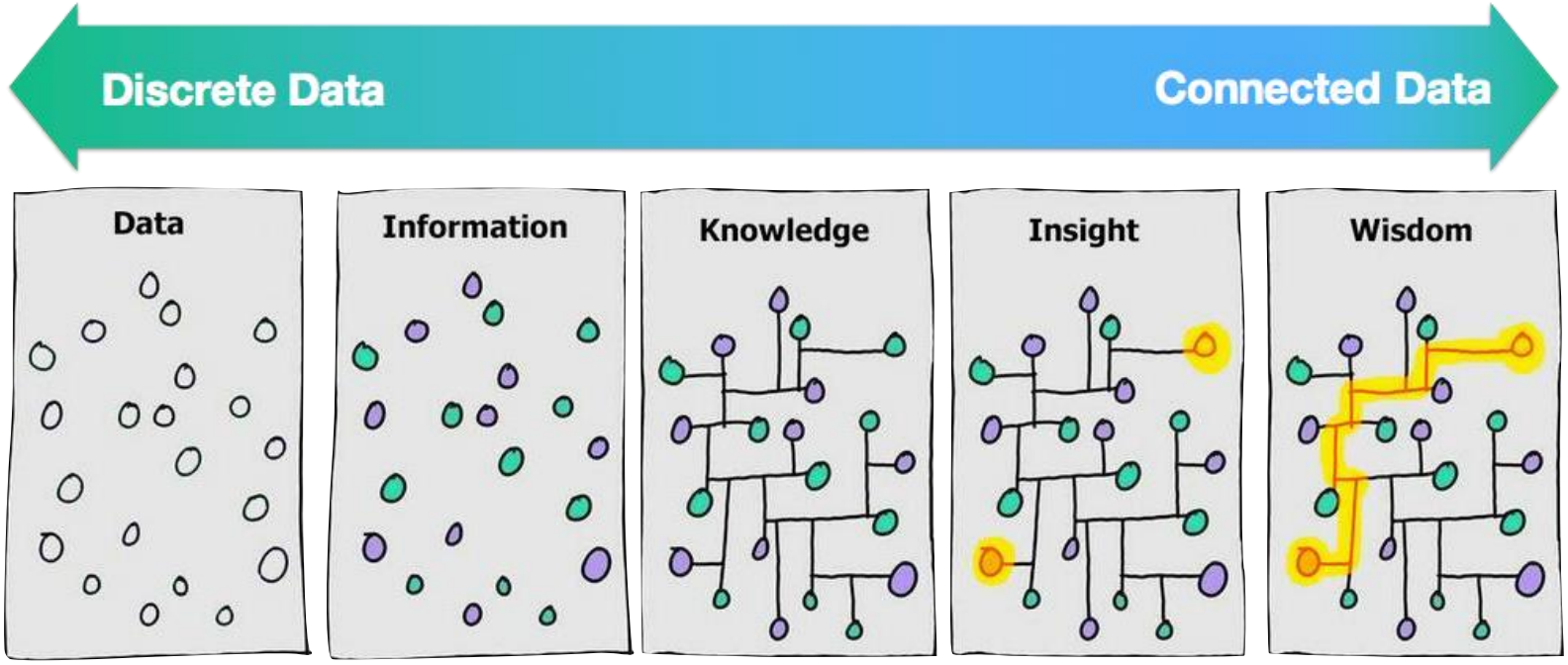
neo4j

Connections in data



Real-Time Connected Insights

"Our **Neo4j** solution is literally **thousands of times faster** than the prior MySQL solution, with queries that require **10-100 times less code**"
Volker Pacher, Senior Developer
ebay



RDBMS & Aggregate-Oriented NoSQL

Hadoop / EDW/ Columnar RDBMS

|<————————- Graph Database & ————————>|
Graph Compute Engine
(Graph Transactions & Analytics)

Illustration by David Somerville based on the original by Hugh McLeod (@gapingvoid)

Soooo….Where does Neo4j fit?

Not for high volume OLTP

- Yes, neo4j is ACID and fully transactional
- But it isn't designed around 10's of thousands of users with small transactions
- Has problems with complex queries on large data sets

Not for large scale aggregations/slicing & dicing

- This is where columnar databases/schemas shine
- Optimized storage and CPU hardware instructions
- Has problems with joins (other than Star Schema Dimensions → Fact table)

Data Exploration & Discovery → YES!!!

- 360-degree view queries
- Path/Journey/Lineage/Hierarchical queries
- Knowledge Graphs, Recommendation Engines
- Discover new relationships/communities/similarities (AI/ML pipeline)
- ….basically, if you see a lot of JOINs or UNIONs in a SQL query….

Neo4j Common Implementations

Identity Access Management

- Rapidly traverse 4+ hops in low ms

Fraud Detection

- Identify potential fraud based on patterns of behavior

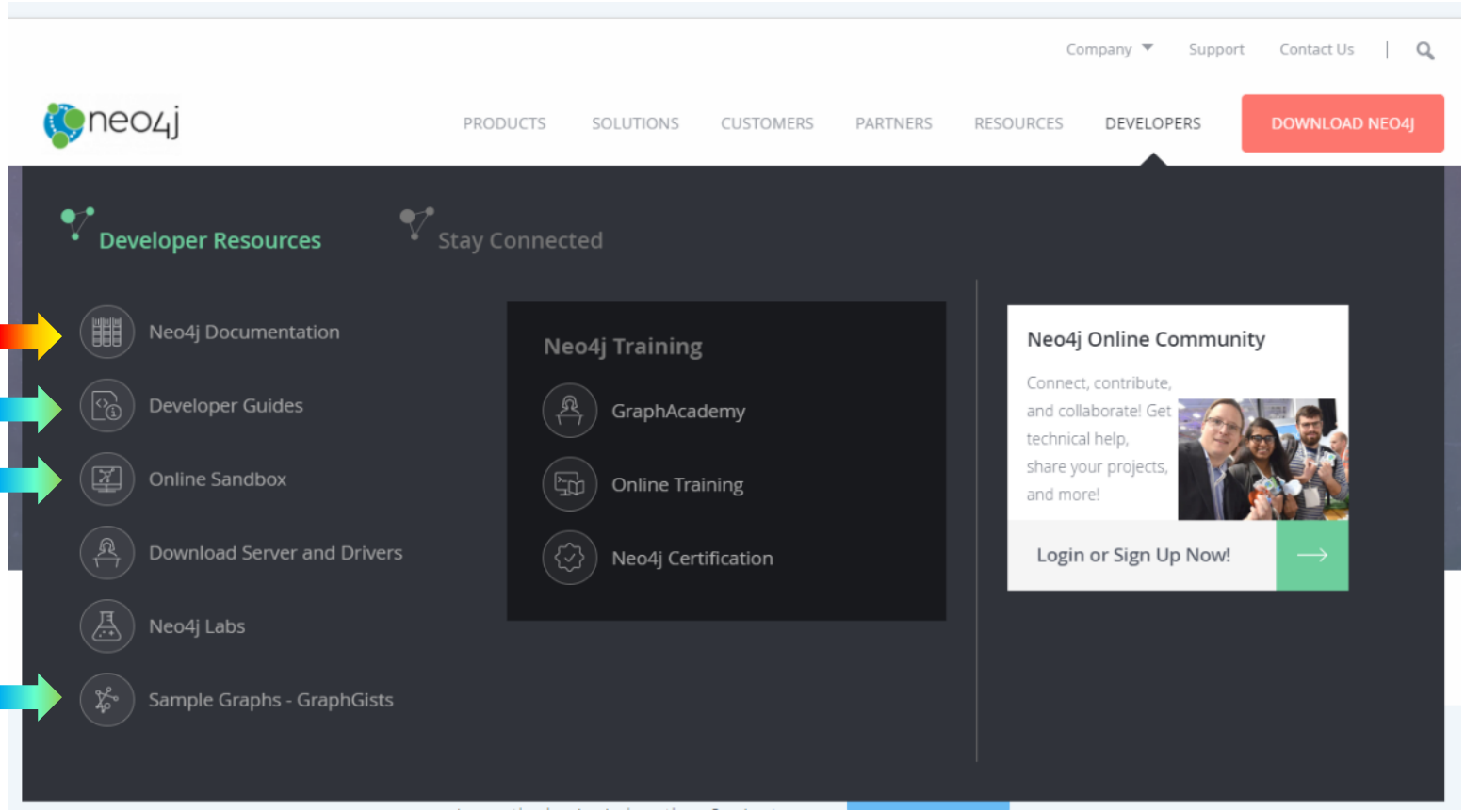
Knowledge Graph

- Fuse multiple data sets and provide cross data set views of knowledge
- Hierarchical knowledge context

Recommendation Engines

- Quickly identify related products in low ms
- Provide recommendations based on similarities with other customers, etc.

What's Online for help with neo4j



Online Sandboxes

Experience Neo4j in a click with the Sandbox

Pick a project and get started in less than 60 seconds. No download required.

Network and IT Management

Dependency and root cause analysis + more for network and IT management.

Launch →

Crime Investigation

Explore connections in crime data using the POLE - Person, Object, Location, Event - model.

Launch →

The ICIJ Panama Papers

Get to know politicians and criminals that hide their cash.

Launch →

Movies Recommendation

Generate personalized recommendations.

Launch →

Graph Gists

Explore By Use Case


GraphGist Challenge Entries


Sports and Recreation


Master Data Management


Real-Time Recommendations


Optimization


Fraud Detection


Pop Culture


Network and IT Operations


Holidays


Graph-Based Search


General Business


Graph Gist How-tos


Data Analysis


Public Web APIs

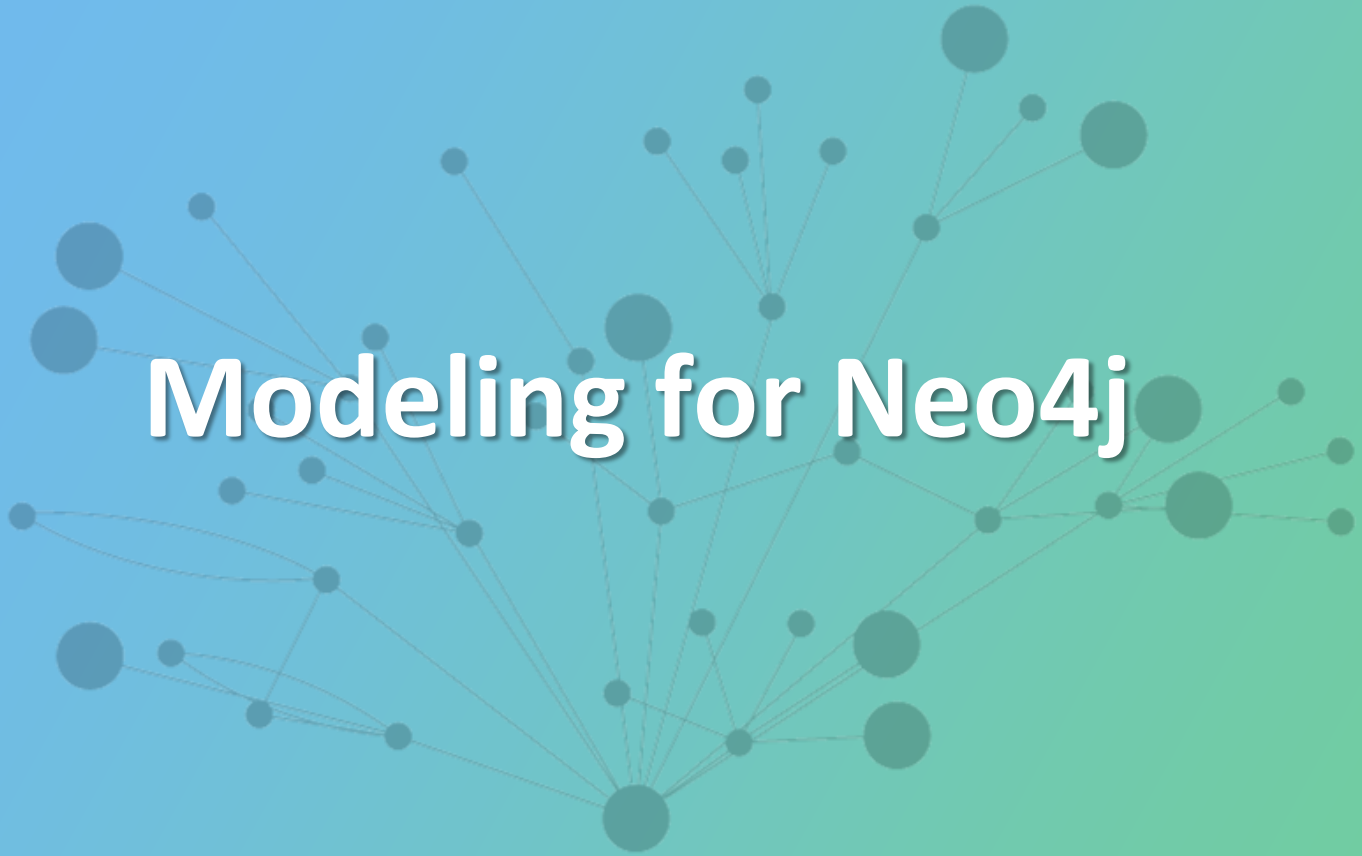

Internet of Things


Investigative Journalism


Open Government Data and Politics


Identity and Access Management

Modeling for Neo4j



Modeling for Neo4j

Neo4j is a labeled property graph database

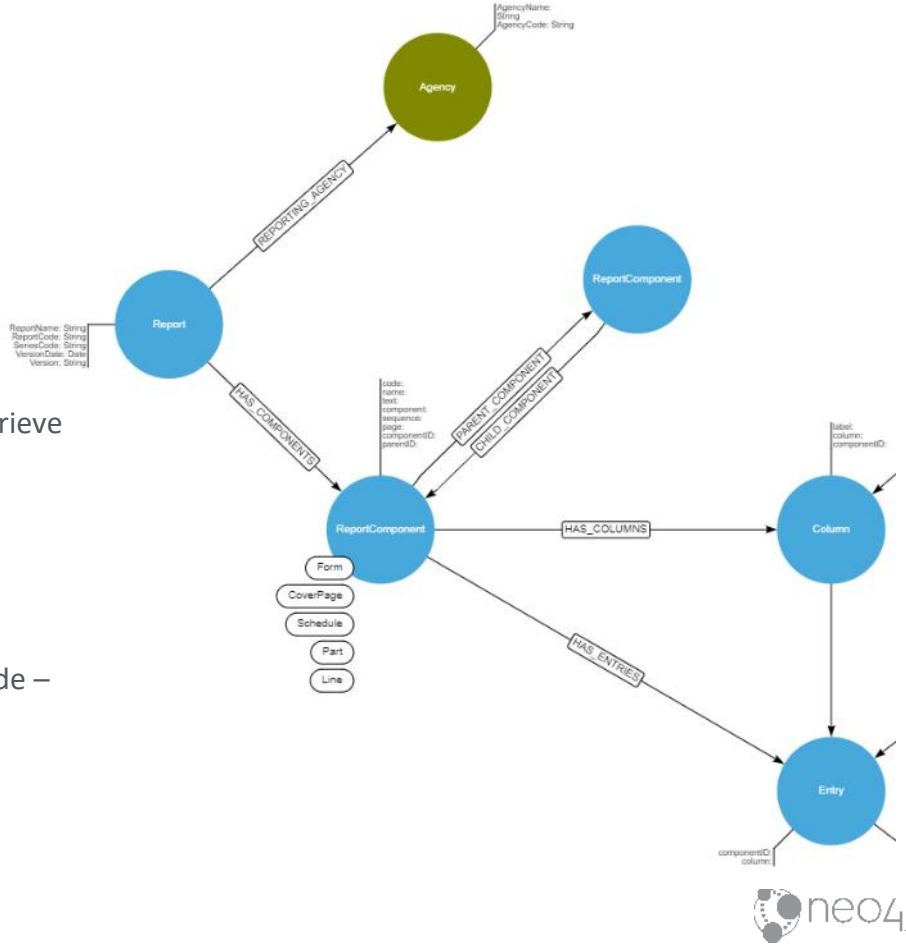
- This is different from Resource Data Framework (RDF) graph databases
- It is most likely closer to Relational ER DBMS

Graphs have “Nodes”

- In Neo4j, a node can have one or more **labels**
  - ✓ *Hence “Labeled Property Graph”*
- Queries in Neo4j use the **labels** to specify which nodes to retrieve
- Nodes in neo4j can have one or more properties
  - ✓ *Similar to columns of a relational table*

Graphs have “Edges”

- In Neo4j, we refer to these as relationships
- More specifically, we call them “Relationship Types”
- Any node can have one or more relationship to any other node – whether same label or not
  - ✓ *E.g. (Person1) –[IS_PARENT_OF]->(Person2)*
  - ✓ *E.g. (Person1) –[OWNS]->(Car1)*
  - ✓ *E.g. (Person2) –[DRIVES]->(Car1)*
- Relationships can have properties as well
  - ✓ *However, often a clue that it really might be a Node instead*



How to Start Modeling

Focus on a few (1 or so) business questions

- Customer 360?
- Customer journey
- Customers at risk of committing fraud

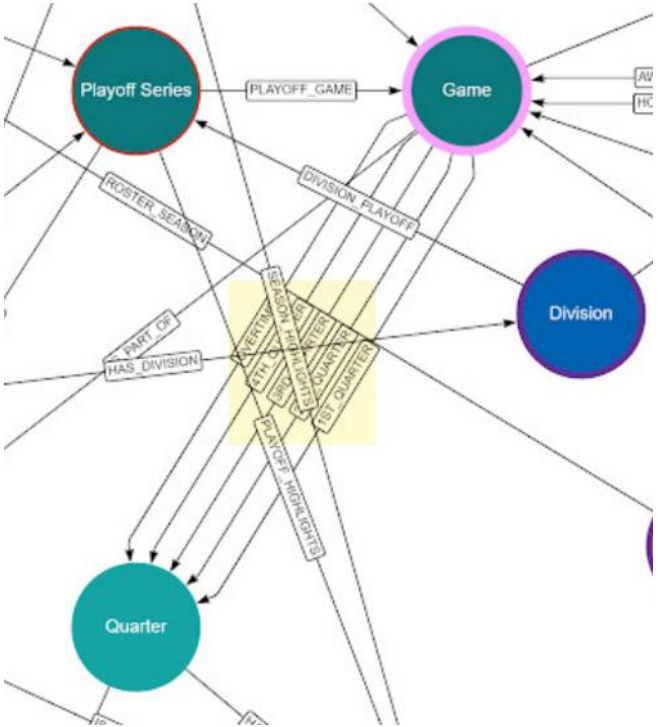
You can then think of it in ER terms ….but…

- You may split some “attributes” out into separate nodes
  - ✓ *E.g. customer addresses, SSN, etc*
  - ✓ *Things you might have indexed in RDBMS*
- This allows you to rapidly find & traverse relationships where people are sharing SSN’s or email (fraud)
- You may have more than one relationship
  - ✓ *In ER – typically there is only one Pkey to Fkey relationship*
  - ✓ *In graph you may want more than one to speed up traversals*
- This allows you to “filter on join” with low cardinality attributes vs. depending on the typical DBMS join then filter

Then add additional business questions

- Refactor model as necessary
  - ✓ *Make properties into nodes, relationships to nodes, etc.*
- Add additional properties where necessary

Rinse, Repeat



Tips & Things you will hear about

Avoid High Density Nodes

- What
  - ✓ *Nodes with 100's or 1000's of relationships*
  - ✓ *Common ones: "State", Demographics (e.g Gender, Race, etc.)*
- Why
  - ✓ *Adding a new relationship or updating the node could block other concurrent users*
    - Relationships are stored on the nodes in neo4j
- What to do instead
  - ✓ *Use a more discrete node in between – e.g. zipcode, demographic profile, etc.*

Use multiple relationships vs. properties on relationship

- What
  - ✓ *If a node has a lot of relationships, it can lead to "explosion" during traversal*
- What to do instead
  - ✓ *Use more descriptive relationships*
    - "Parts Supplier"/"Office Supplier"; "Sent_20190704", etc.

Modeling things to think about…

If it is indexed in SQL – consider making it a node

- For example, customers have addresses….usually with city, state, zip
- Pull city, state, zip out to a separate node
- (Customers)-[:LIVE_IN]->(PostalCodes)
- Note that city + state + zip are in a single node
  - ✓ *Postal Code may be node key (or not – choice is yours)*
  - ✓ *City/State can be indexed – composite index if desired*
- Pull State out even further

This may seem like it adds extra hops, but….

- Neo4j can do 5-10 million traversals per core on recent hardware
- It reduces the data size
  - ✓ *Storing all 10M inhabitants of NY city*
    - As properties → 10M * 41bytes * length(“New York” + “NY” + “12345”)=~6GB
    - As node → (10M * 41bytes) +15 = ~400MB

Add “NEXT_xxxx” Relationships – good for customer journey, etc.

- E.g. next_order, next_date

Data Properties or Nodes??? (hint: Nodes!)

City, State, Zip/Postal Code
- Beat that to death on last page

Dates of Birth
- Only 25K in last 70 yrs – likely highly repetitive
- Facilitates finding people of same year/birth month
  - ✓ *date().year, date().month, date().week*

SSN
- Yes – it's unique ….but…
- Helps to find fraud!!!

Phone
- Can be split area code, exchange easier and less overhead
- Allows multiple numbers per user (mobile + home)

Transaction Dates
- Again, highly repetitive
- Find all txns for same date/date range

State, Country (any region)
- Relationship to postal code not entity

Something_Type (fairly static)
- Although see next slide first about relationships
- Low cardinality – customer type
- Speeds query
  - ✓ *Traversal to node label is faster than property fetch*

Relationship

CustAcctMgr –[MANAGES]->Customer

- (CustAcctMgr)-[IS_PRIVATE_BANKER]->(PrivBankCust:Customer)
- (CustAcctMgr)-[IS_INSTITUTIONAL_MGR]->(InstCust:Customer)
- (Delco)-[IS_COMPONENT_SUPPLIER]->(Chevrolet:GM)
- (Staples)-[IS_OFFICE_SUPPLER]->(GM)

Property Types

- (Transaction)-[DEPOSITS]->(Account)
- (Transaction)-[WITHDRAWAL]->(Account)
- (Transaction)-[DEBITS]->(Account)

Common “Sub-graphs” you will *likely* have…(1)

Identity Access Management

- Most full time employees will/can have individual accounts….
- ….but you may have temp staff that needs access (e.g. equip repair) for just a few days
  - ✓ *Create “service” login account*
  - ✓ *As needed, specify in IAM subgraph what data they can see*

“Calendar Tree”

- (Year)-[]-(Month)-[]->Day
- (Day)-[NEXT_DAY]->(Day), (Month)-[NEXT_DAY]->(Month), …
- Rather than storing any dates as properties, have relationships to the Calendar Tree
  - ✓ *dateOpened, BirthDate, transactionDate, etc.*
- Really easy to do cross node date analysis
  - ✓ *Find everyone who made a withdrawal of >10000 within 60 days of retirement at age 65*

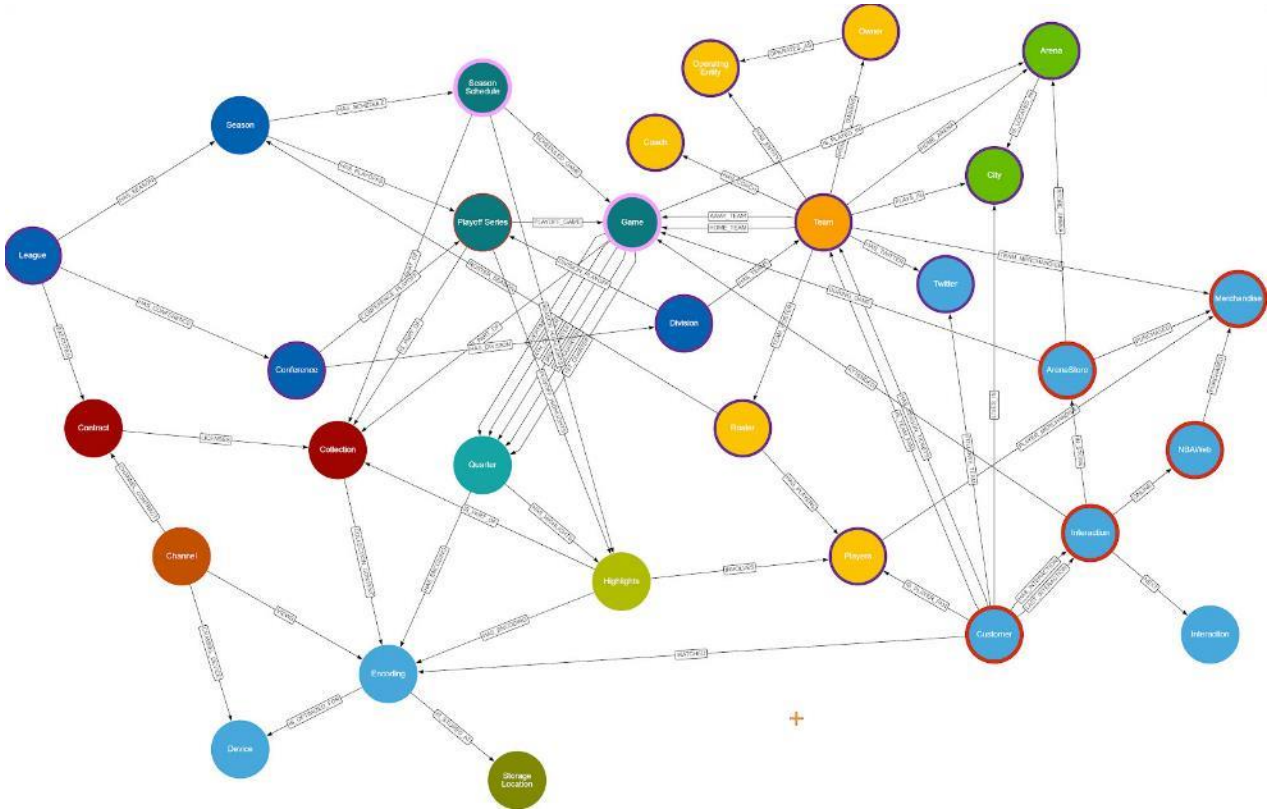
```
MATCH (cust)-[]->(dob)-[]->(day)-[NEXT_DAY*1..60]<-[XACT_DATE]-(trans)
WHERE dob.year=1955 AND trans.amount > 10000.0
```

Common “Sub-graphs” you will *likely* have…(2)

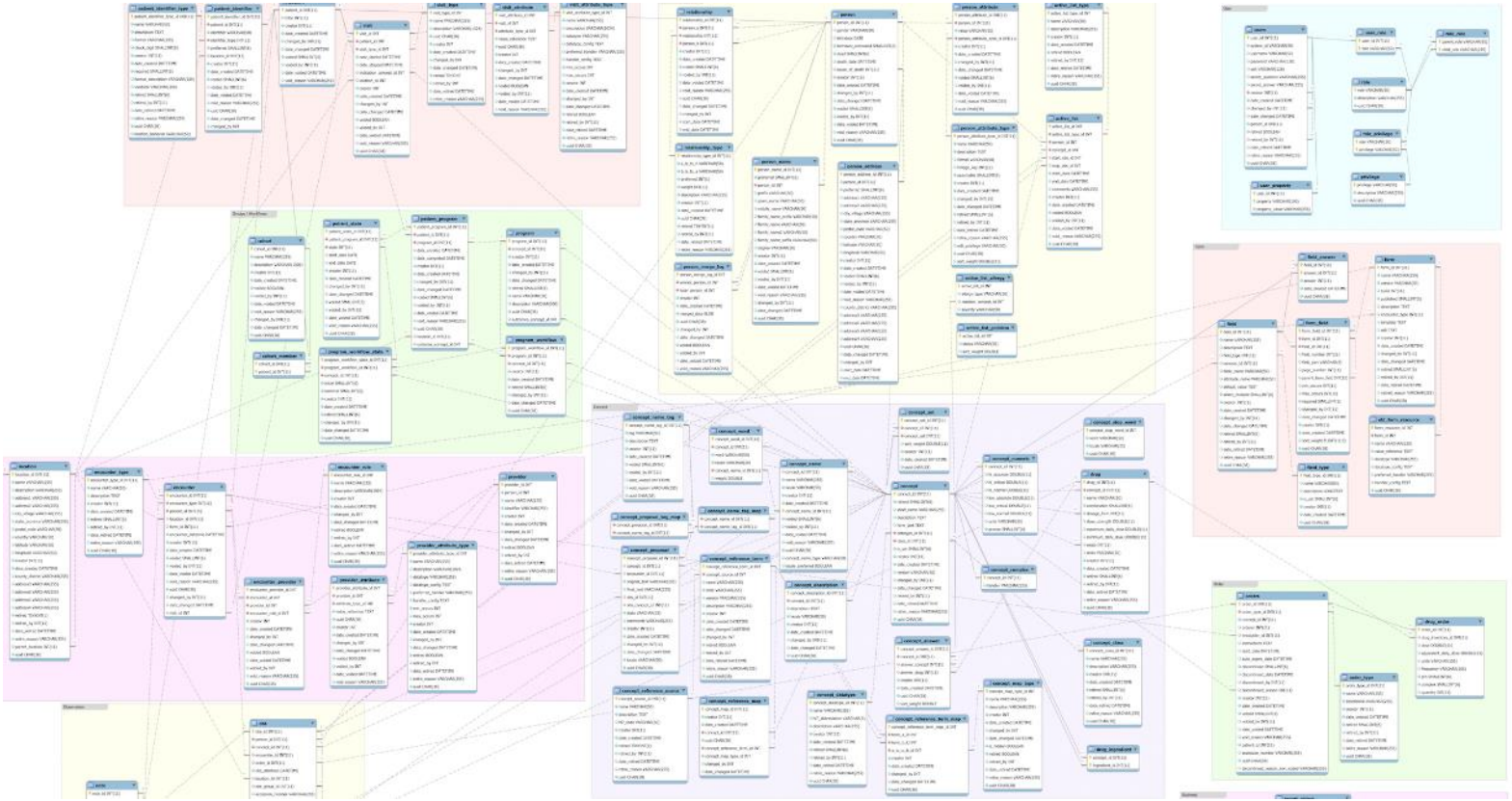
Hierarchical

- Data Lineage, Bill of Materials, Parts Explosion, etc.
- Locations (City)-[]->(State)-[]->(Country)-[]->(Region)
- If limitless (parts explosion) you want to have “generic” label/relationship type
  - ✓ *You can use multi-label for filtering, but the generic label & relationship type gives you the ability to quick traverse to any depth*
- If limited/low cardinality – keep separate (ala Locations)

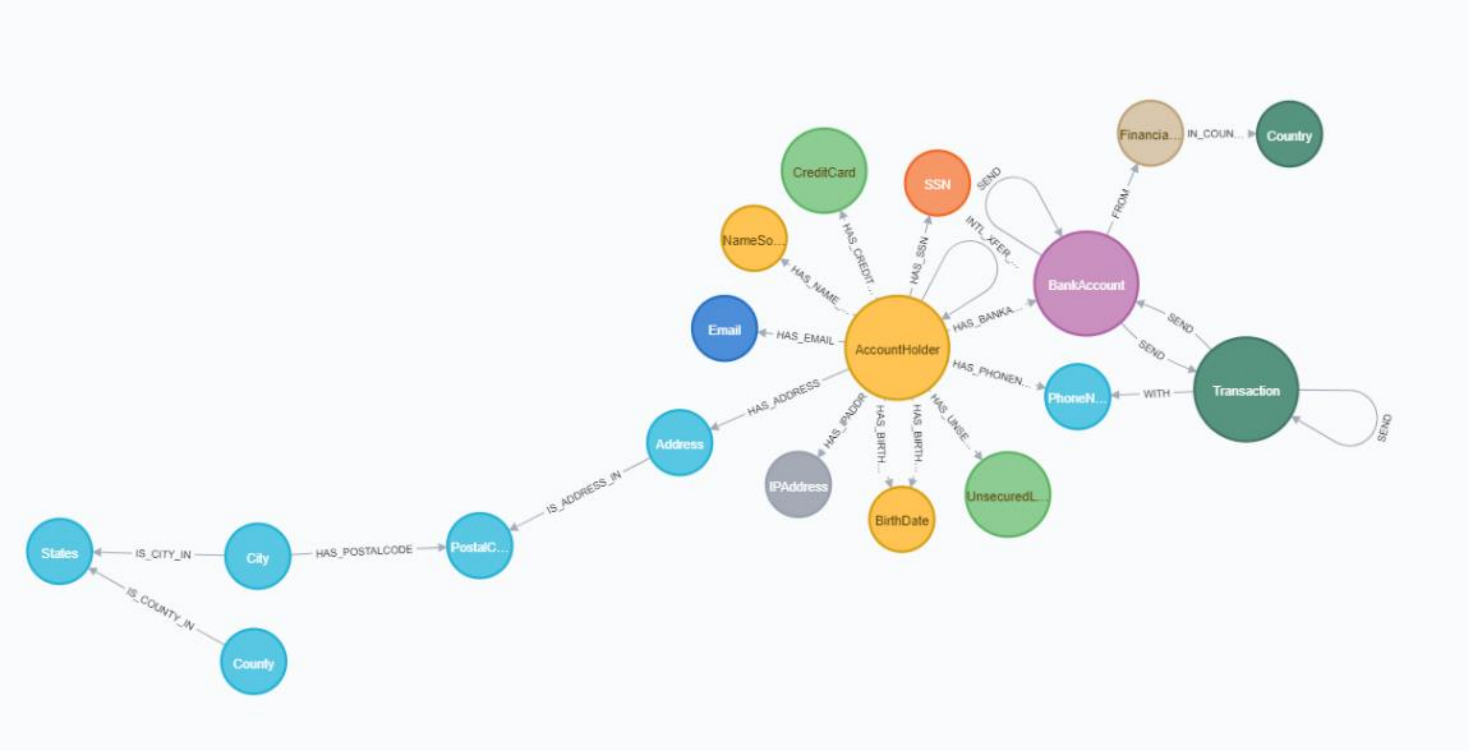
It can get messy…..



…but so can this



Avoiding High Density Nodes



Tools & Books

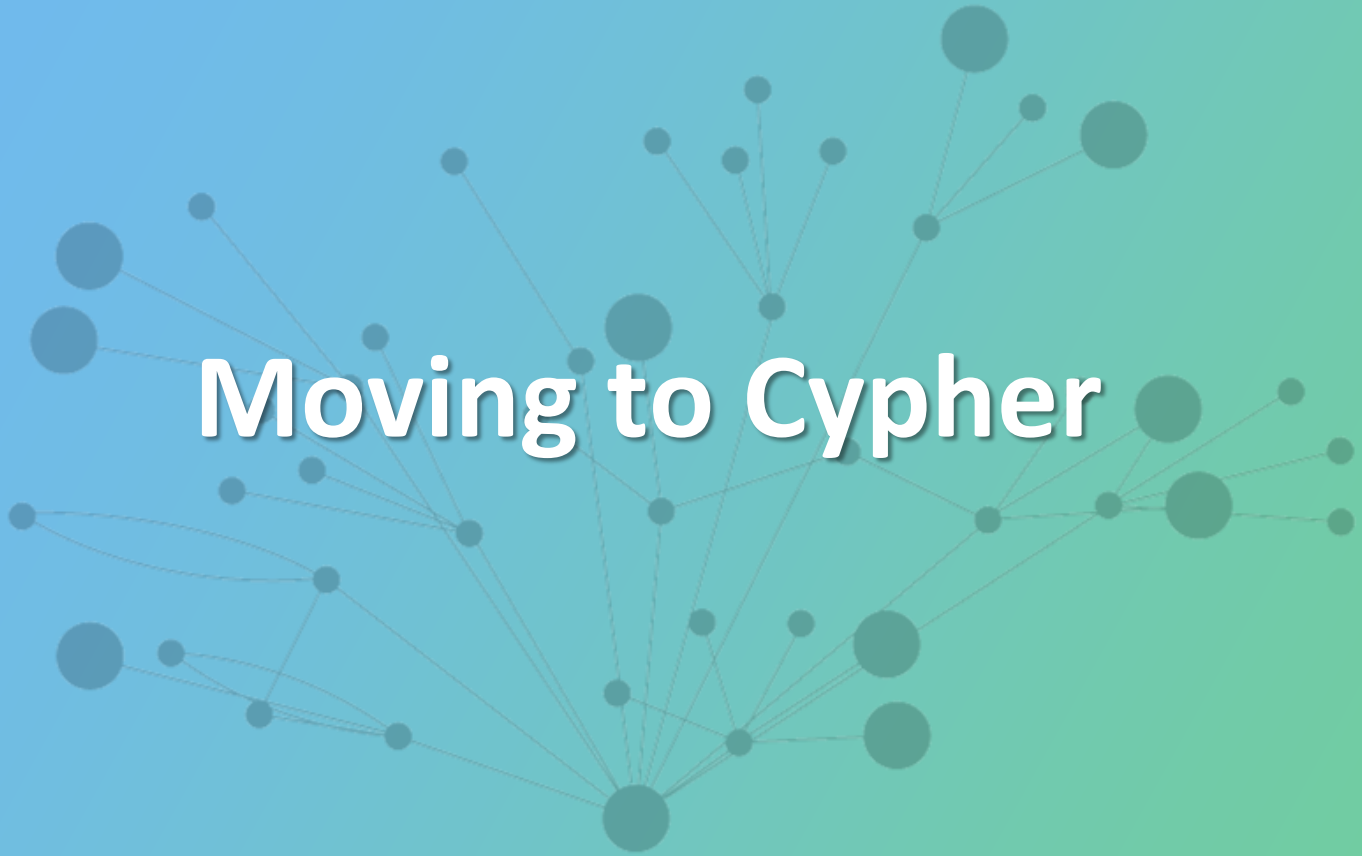
There are tools to help

- Because graph databases are new to industry
  - ✓ *…most enterprise modeling tools don't support – ER modeling for SQL only*
  - ✓ *….so the tooling tends to be a bit immature and sparse*
  - ✓ *…and some only on Mac vs. Windows*
- Some examples
  - ✓ *Arrows (OpenSource/Neo4j)*
  - ✓ *Grafo (paid version can generate Cypher - $8-16/pp/month), yEd, et al*
  - ✓ *Omnigraffle (Mac - $150-$250), Hackolade (~$1000/yr)*

Books

- Neo4j online developer's guide - https://neo4j.com/developer/data-modeling/
- Most Neo4j books have a section on modelig
  - ✓ *Graph Databases (Robinson, Webber, Eifrem)*
  - ✓ *Neo4j Graph Data Modeling*
- Graph Data Modeling for NoSQL and SQL (Thomas Frisendal)

Neo4j Training Class



Moving to Cypher

neo4j

Cypher & Schemas

Neo4j is a NoSQL/Schema-less/ “Schema-optional” DBMS

Classic schema components supported

- Unique constraints
  - ✓ *Although a bit strange – unique constraint is only for single column and only when non-NULL*
  - ✓ *In otherwords, you can’t have a multi-column unique constraint*
  - ✓ *….and a unique constraint will allow multiple nodes with NULL values for property*
  - ✓ *….so it really is “unique for non-NULL values”*
  - ✓ *This is an artifact of NoSQL – e.g. the property isn’t required AND if not present, a query will return a NULL value for that node’s property value*
- Node key constraints (aka “Primary Key”) – multi-column
  - ✓ *Strongly recommend that every label has a node key (only exception is multiple labels – tbd later)*
- Indexes (multiple-column) – but ***only*** on node properties
  - ✓ *Relationship properties cannot be indexed – therefore bad SARGs for searching*
- Exists constraints (aka “not null column”)

Labels, RelationshipTypes & PropertyNames are Strings

Properties can be one of the supported datatypes

Cypher Schema: Datatypes

Storage Types

- Very basic java types
  - ✓ *Integer, Float, String, Date, Datetime*
  - ✓ *The default type is String – you will need to cast to other types when loading*
    - toInteger, toFloat, date(), datetime()
- Storing of arrays of basic types allowed, but no maps

Dynamic Types

- 'Dynamic' types can be used in cypher, but not stored natively
- can be marshalled as String if you want to store intact
  - ✓ *Point {} for spatial support*
  - ✓ *JSON/maps*

Cypher Schema: Labels

A node has one or more labels

- CREATE (:Car {VIN: 123456})
- “Car” is the label above

Caution when using more than one label

- May be useful for some types of hierarchical data vs. creating nodes for hierarchy
- Allows flexibility in querying subclasses or generic classes when necessary
- Examples:
  - ✓ *CREATE (:Sedan:Automobile:Vehicle {VIN 123456})*
  - ✓ *CREATE (:Pickup:Automobile:Vehicle {VIN 234567})*
  - ✓ *CREATE (:Semi:Truck:Vehicle {VIN 345678})*
- The problem: Index qualification is only based on single label + properties
- The solution:
  - ✓ *Create index at parent class level (Vehicle)*
  - ✓ *Always reference desired node and parent node in query*
    - MATCH (:Sedan:Vehicle) …WHERE ….

Cypher is a “dataflow” language

Explicit transaction support

Multi-statement batches

- Statement terminator is semi-colon (;)

neo4j