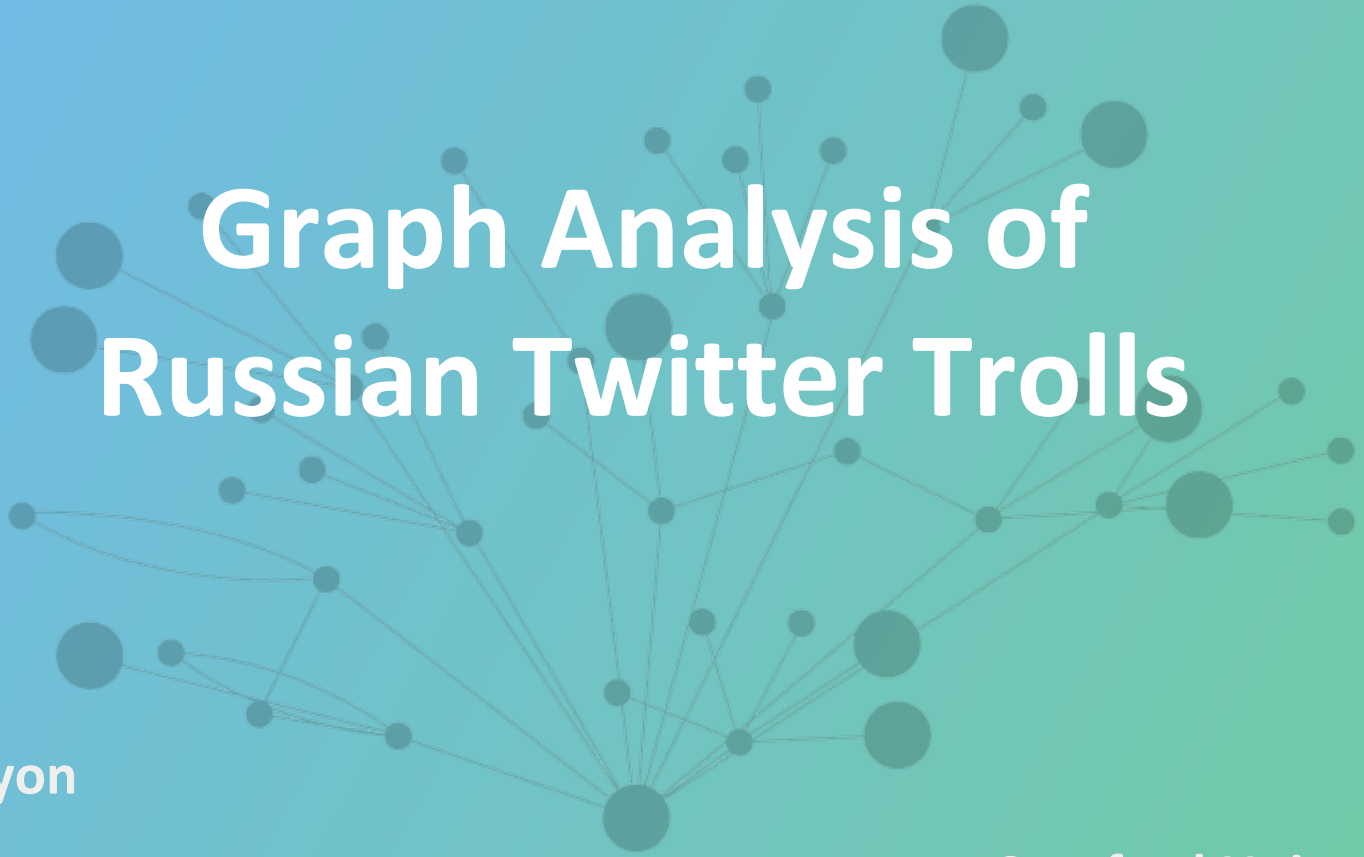
Graph Analysis of Russian Twitter Trolls



William Lyon
@lyonwj
lyonwj.com

Stanford University
Feb 2018

Agenda

Overview of Neo4j and graph databases in the context of Russian Twitter Troll data

William Lyon

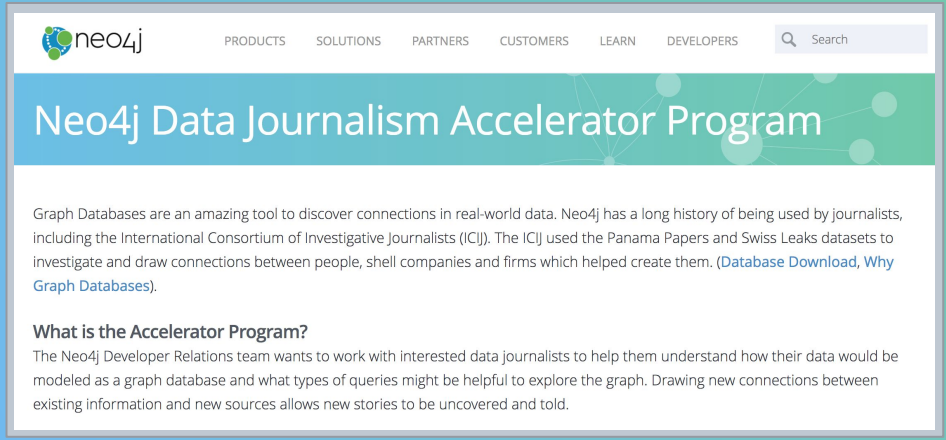
Software Developer, Neo4j

(Developer Relations, integrations, GraphQL)

will@neo4j.com

@lyonwj

lyonwj.com



TECH DEC 20 2017, 11:11 AM ET

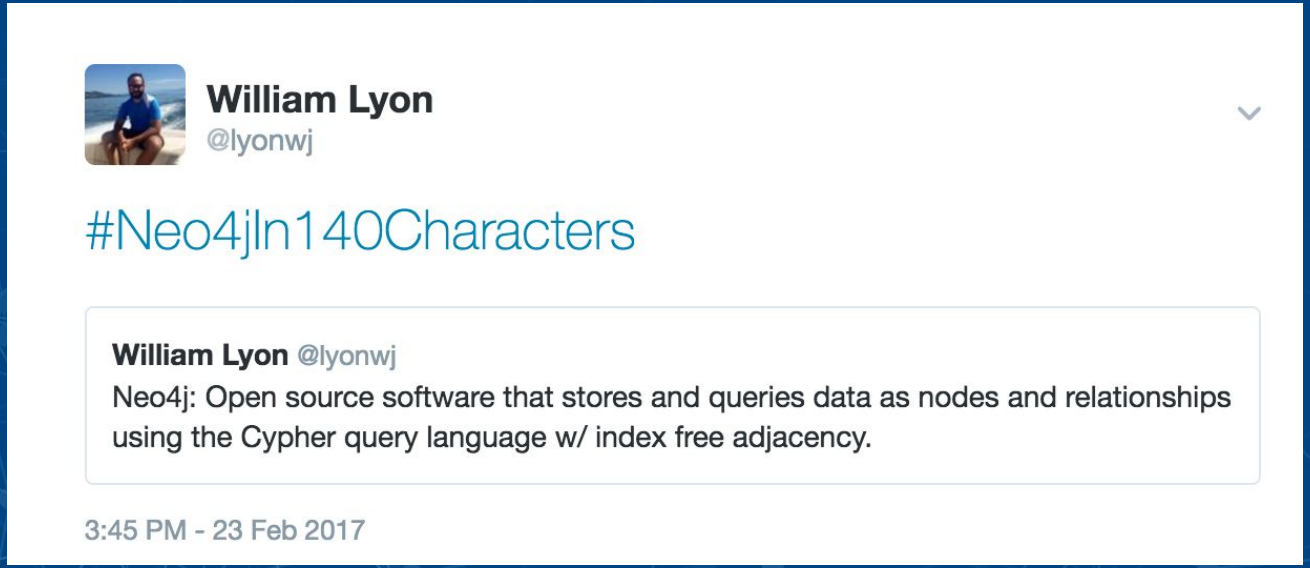
Russian trolls went on attack during key election moments

by BEN POPKEN

https://www.nbcnews.com/tech/social-media/russian-trolls-went-attack-during-key-election-moments-n827176

What’s a Graph Database???

What's a Graph Database???



neo4j

William Lyon
@lyonwj

Neo4j: Open source software that stores and queries data as nodes and relationships using the Cypher query language w/ index free adjacency.

3:44 PM - 23 Feb 2017 from Dunlap, Seattle

Reply to @lyonwj

This repository Search Pull requests Issues Gist

neo4j / neo4j

Watch 326 Unstar 3,502 Fork 1,079

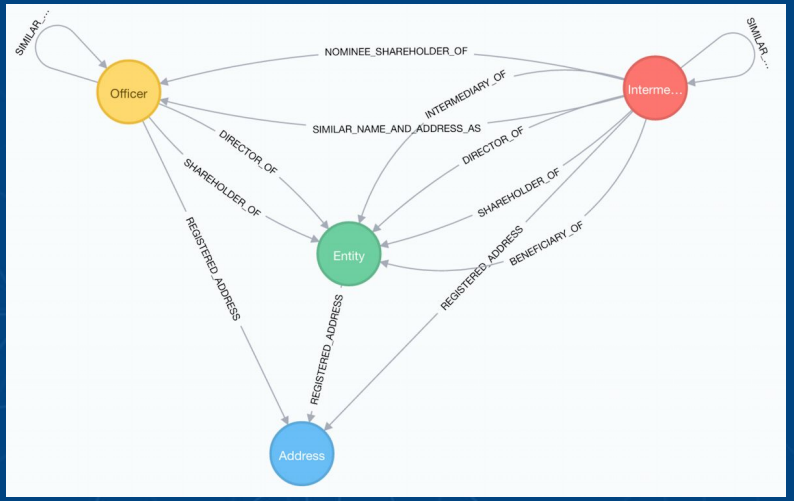
Code Issues 379 Pull requests 33 Projects 0 Wiki Pulse Graphs

Graphs for Everyone http://neo4j.com

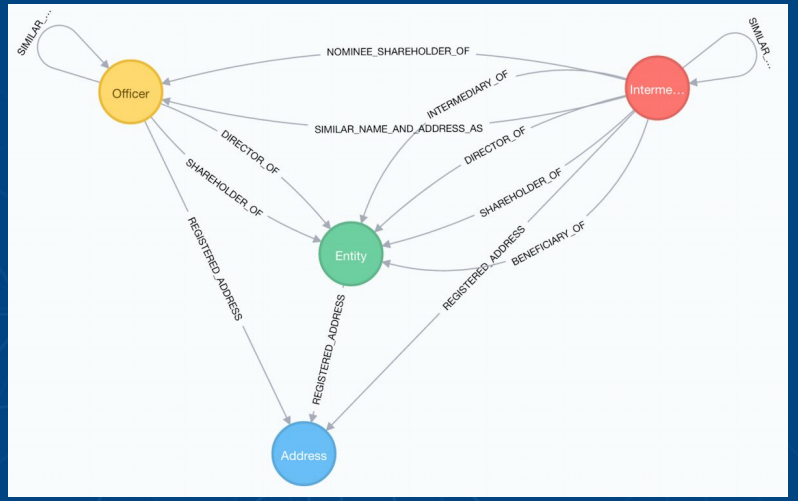
45,336 commits 18 branches 173 releases 141 contributors

- Data durability
- ACID transactions

William Lyon
@lyonwj
Neo4j: Open source software that stores and queries data as nodes and relationships using the Cypher query language w/ index free adjacency.
3:44 PM - 23 Feb 2017 from Dunlap, Seattle
Reply to @lyonwj


SIMILAR...
NOMINEE_SHAREHOLDER_OF
SIMILAR...
Officer
INTERMEDIARY_OF
Interme...
SIMILAR_NAME_AND_ADDRESS_AS
DIRECTOR_OF
DIRECTOR_OF
SHAREHOLDER_OF
SHAREHOLDER_OF
REGISTERED_ADDRESS
Entity
REGISTERED_ADDRESS
BENEFICIARY_OF
REGISTERED_ADDRESS
Address

neo4j



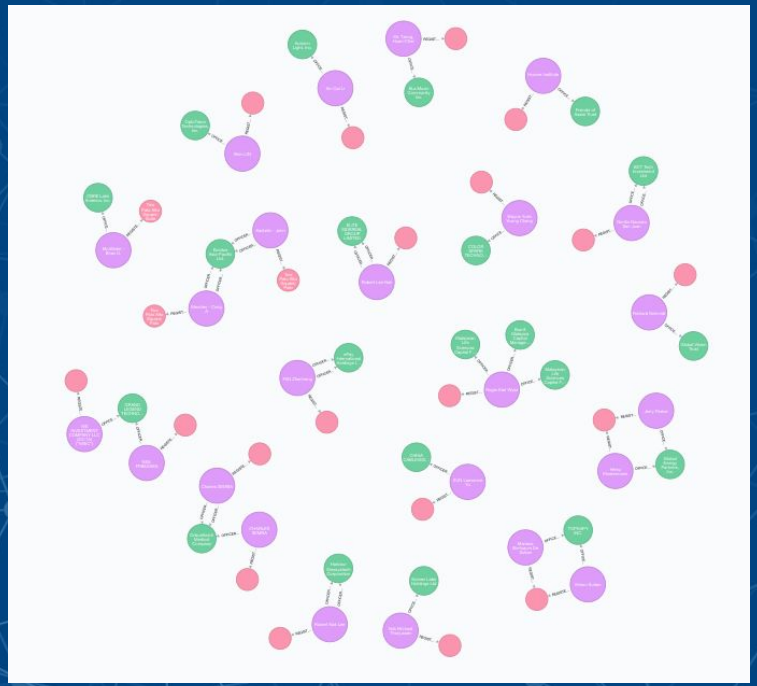
https://offshoreleaks.icij.org/pages/database

neo4j

```
1 MATCH (a:Address)<-[r:REGISTERED_ADDRESS]-(o:Officer)-->(e:Entity)
2 WHERE a.address CONTAINS "Palo Alto"
3 RETURN *
```

http://www.opencypher.org/

```
MATCH (a:Address)<-[r:REGISTERED_ADDRESS]-(o:Officer)-->(e:Entity)
WHERE a.address CONTAINS "Palo Alto"
RETURN *
```



neo4j

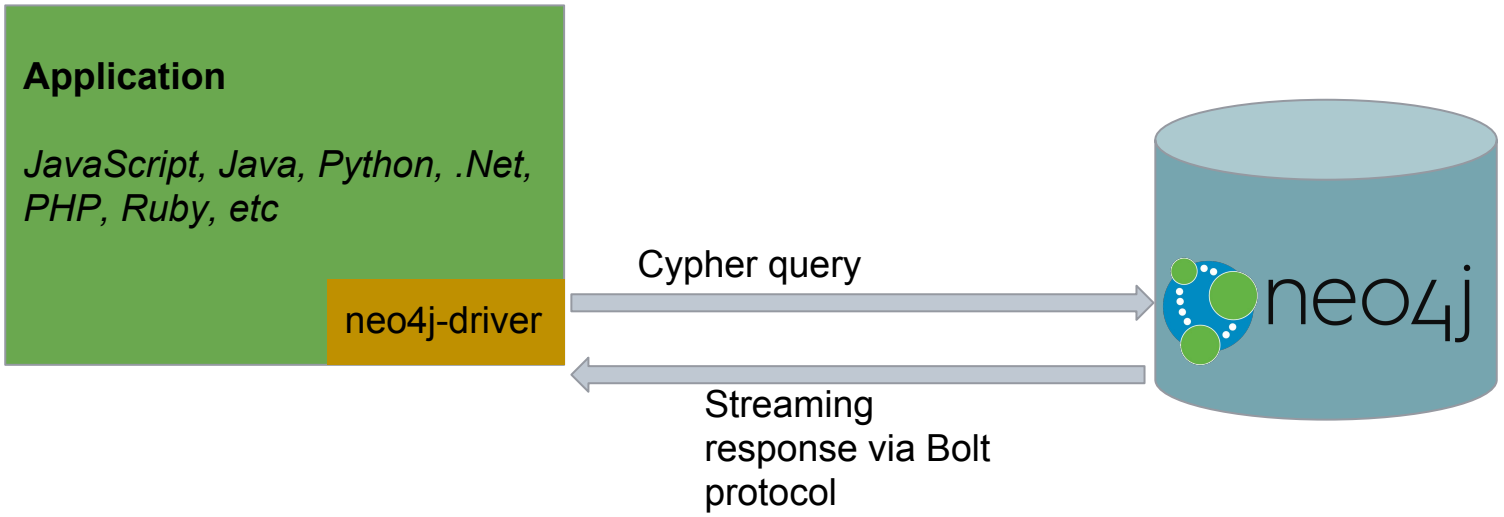
```
1 MATCH (a:Address)<-[r:REGISTERED_ADDRESS]-(o:Officer)-->(e:Entity)
2 WHERE a.address CONTAINS "Palo Alto"
3 RETURN e.jurisdiction_description AS description, COUNT(*) AS num
4 ORDER BY num DESC
```

```
1 MATCH (a:Address)<-[r:REGISTERED_ADDRESS]-(o:Officer)-->(e:Entity)
2 WHERE a.address CONTAINS "Palo Alto"
3 RETURN e.jurisdiction_description AS description, COUNT(*) AS num
4 ORDER BY num DESC
```

"description"	"num"
"British Virgin Islands"	18
"Bermuda"	4
"Undetermined"	3
"Labuan"	3
"Cook Islands"	2
"Cayman"	1
"Cayman Islands"	1
"Samoa"	1

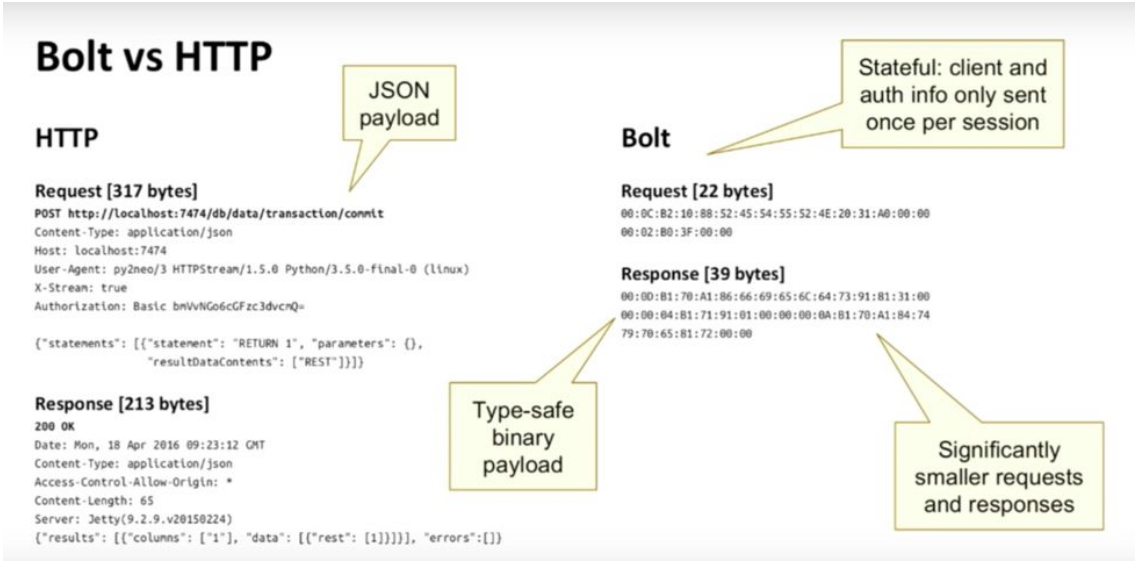
Using Cypher With Client Drivers

Client driver libraries use Cypher over Bolt protocol



https://neo4j.com/developer/language-guides/

Bolt Protocol

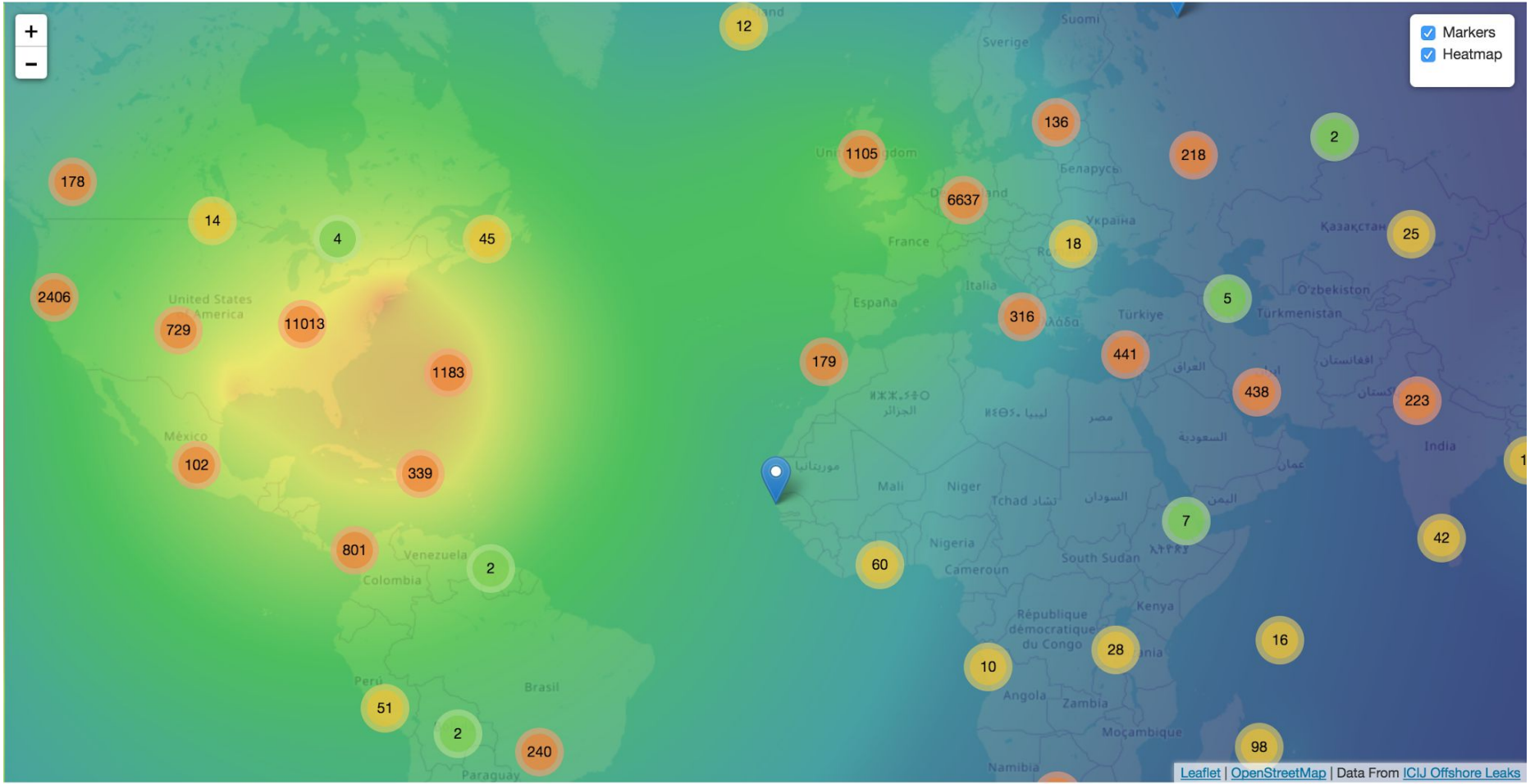


- Unified type system
- Stateful (auth)
- Less data over wire
- Efficient serialization

```
var driver = neo4j.v1.driver("bolt://localhost:7687", neo4j.v1.auth.basic("neo4j", "MYPASSWORD"));
var session = driver.session();
session
  .run(`MATCH (a:Address)<-[:REGISTERED_ADDRESS]-(o:Officer)-[:CONNECTED_TO|OFFICER_OF]-(e:Entity)
        WHERE exists(a.latitude) and exists(a.longitude)
        RETURN a.name AS address, a.latitude AS latitude, a.longitude AS longitude,
          COLLECT(DISTINCT o.name) AS officers, COLLECT(DISTINCT e.name) AS entities,
          COLLECT(DISTINCT e.jurisdiction_description) AS jurisdictions, 1.0*COUNT(*) AS strength`)
  .subscribe({
    onNext: function (record) {
      //console.log(record);
      var marker = new L.Marker(new L.LatLng(record.get('latitude'), record.get('longitude')));
      marker.bindPopup('<b>Address:</b> ' + record.get('address') + '<br>' + '<b>Officers: </b>' + record.get('officers').toString()
      markerLayers.addLayer(marker);
      heatmap.pushData(record.get('latitude'), record.get('longitude'), record.get('strength')*0.01);
    },
    onCompleted: function () {
      var overlayMaps = {'Markers': markerLayers, 'Heatmap': heatmap};
      var controls = L.control.layers(null, overlayMaps, {collapsed: false, autoZIndex: true});
      map = new L.Map('map', {center: new L.LatLng(51.505, -0.09), zoom: 3, layers: [baseLayer]});
      map.addLayer(markerLayers);
      map.addLayer(heatmap);
      controls.addTo(map);
      session.close();
    },
    onError: function (error) {
      console.log(error);
    }
  });
```

http://www.lyonwj.com/2017/11/28/geocoding-paradise-papers-neo4j-spatial-visualization/

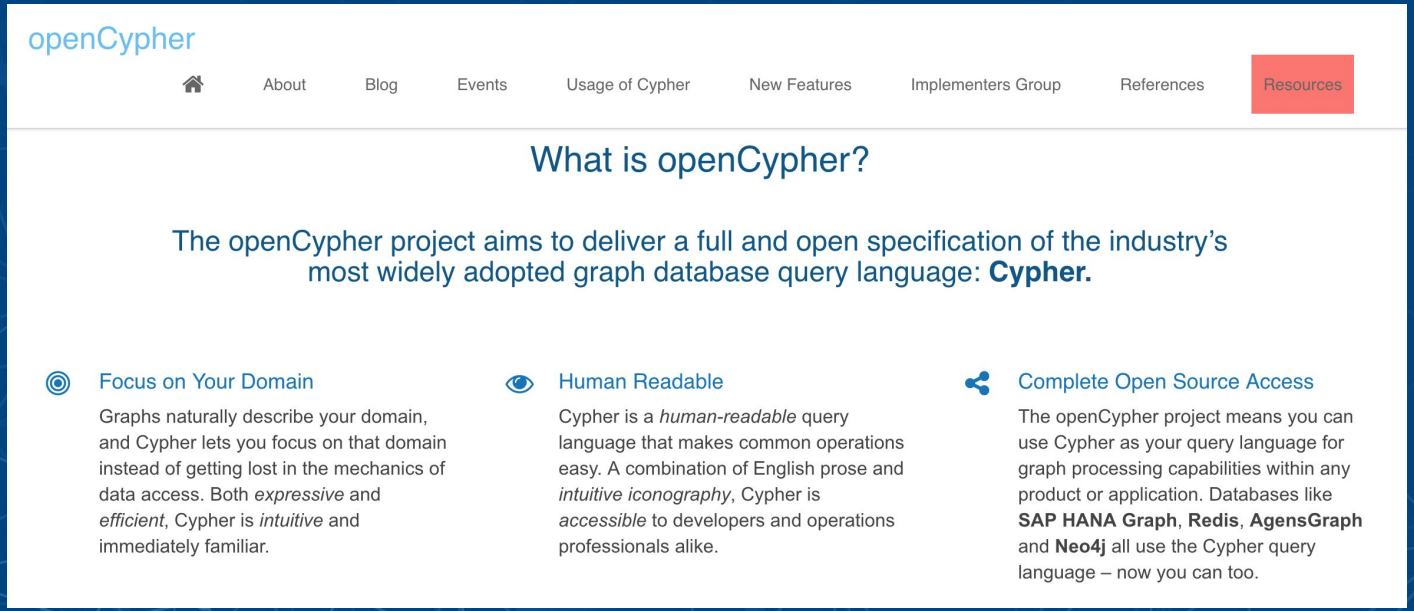
http://www.lyonwj.com/pp-viz/heatmap/



http://www.lyonwj.com/2017/11/28/geocoding-paradise-papers-neo4j-spatial-visualization/

http://www.lyonwj.com/pp-viz/heatmap/

openCypher



http://www.opencypher.org/

neo4j

Cypher For Apache Spark

```
val CITYFRIENDS_NA = SN_NA.cypher(
  """
    | MATCH (a:Person)-[:IS_LOCATED_IN]->(city:City)<-[:IS_LOCATED_IN]-(b:Person),
    |       (a)-[:KNOWS*1..2]->(b)
    | WHERE city.name = "New_York" OR city.name = "San_Francisco"
    | RETURN GRAPH result OF (a)-[r:SIMILAR_CIRCLE]->(b)
  """.stripMargin).graphs("result").cache

CITYFRIENDS_NA.asZeppelinGraph
```

Nodes 37: Person

Relationships 99: SIMILAR_CIRCLE

https://github.com/opencypher/cypher-for-apache-spark

neo4j

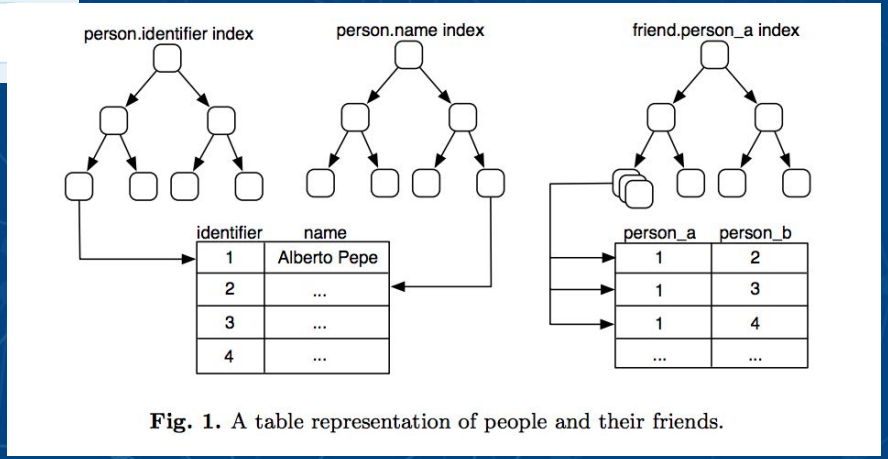


Fig. 1. A table representation of people and their friends.

https://arxiv.org/pdf/1004.1001.pdf

Neo4j Clustering Architecture

Raft-based architecture

Consensus commits via “Core” servers

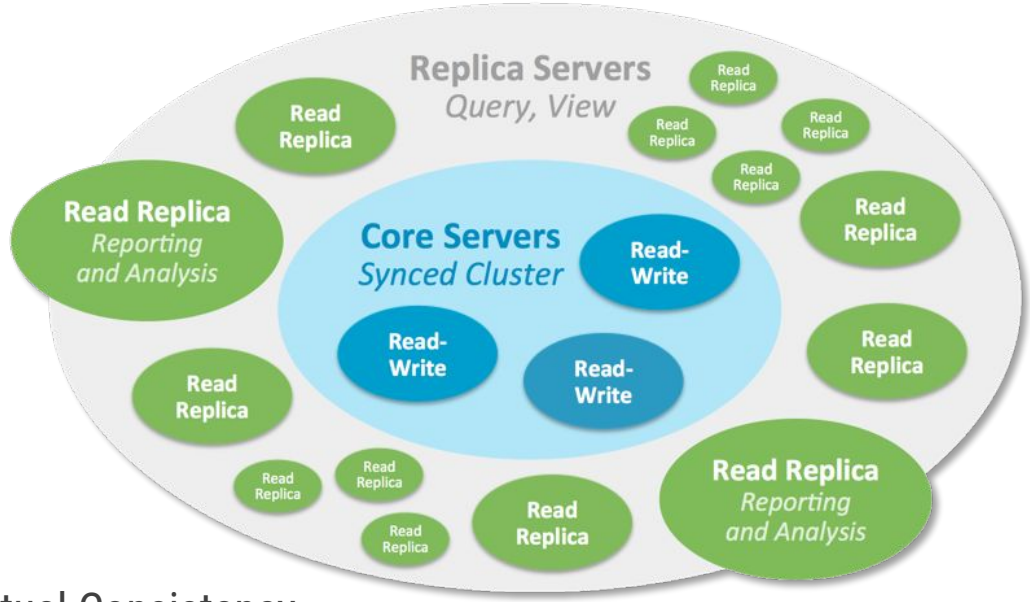
Cluster-aware drivers

No need for external load balancer

Stateful, cluster-aware sessions

Causal Consistency

Stronger consistency model than Eventual Consistency



Common Use Case

- Real time recommendations
- Fraud Detection
- Network & IT Management
- Social Networks
- Bill of Materials / Supply Chain
- Knowledge Graphs
- Master Data Management
- Access Management
- Microservices Analysis
- IoT
- Investigative Journalism
- ...

TECH DEC 20 2017, 11:11 AM ET

Russian trolls went on attack during key election moments

by BEN POPKEN

https://www.nbcnews.com/pages/author/ben-popken

Russia Twitter Trolls

- **2752 Twitter accounts** tied to Russia's Internet Research Agency
- Accounts **suspended by Twitter**
  - Data deleted
- **What were they tweeting about?**

user id	handle
905874659358453760	10_gop
2528776985	1488reasons
2535564756	1D_Nicole_
79793901	1DRussianFNDM
1860330774	1Erik_Lee
838742761515991041	1lorenafava1
2537164155	2lauragibson
823484279787294720	2ndHalfOnion
80712878	2oliverbaker
80723578	30ToMarsFandom
3644473036	459Jisalge
2261091126	4claireevans
748870305280647168	4ever1937
4036537452	4MySquad
713665792550645760	666steverogers
2534875924	6Druz
3936735149	71bilaljamil1
2533199949	756shawn
2619792207	___Judith___D_
2556610923	___S0phia___
2595466901	__alshaw
2544663209	_Amy_Mccann_
2753352910	_AnnaSwanson
2540856370	_Becker_Gerald
2426927403	_beglov
2559290935	_Ben_Santos

https://democrats-intelligence.house.gov/uploadedfiles/exhibit_b.pdf

Wayback API

user id	handle
905874659358453760	10_gop
2528776985	1488reasons
2535564756	1D_Nicole_
79793901	1DRussianFNDM
1860330774	1Erik_Lee
838742761515991041	1lorenafava1
2537164155	2lauragibson
823484279787294720	2ndHalfOnion
80712878	2oliverbaker
80723578	30ToMarsFandom
3644473036	459Jisalge
2261091126	4claireevans
748870305280647168	4ever1937
4036537452	4MySquad
713665792550645760	666steverogers
2534875924	6Druz
3936735149	71bilaljamil1
2533199949	756shawn
2619792207	___Judith___D_
2556610923	___S0phia___
2595466901	__alshaw
2544663209	_Amy_Mccann_
2753352910	_AnnaSwanson
2540856370	_Becker_Gerald
2426927403	_beglov
2559290935	_Ben_Santos
2498964074	_Billy_Moyer_
2586365248	_Edward_Connor_
2644685369	_Gabriel_Pope_
2592814610	_GeorgeSchultz_
2449237434	_GregorVan
2622070817	_Howard_Good_

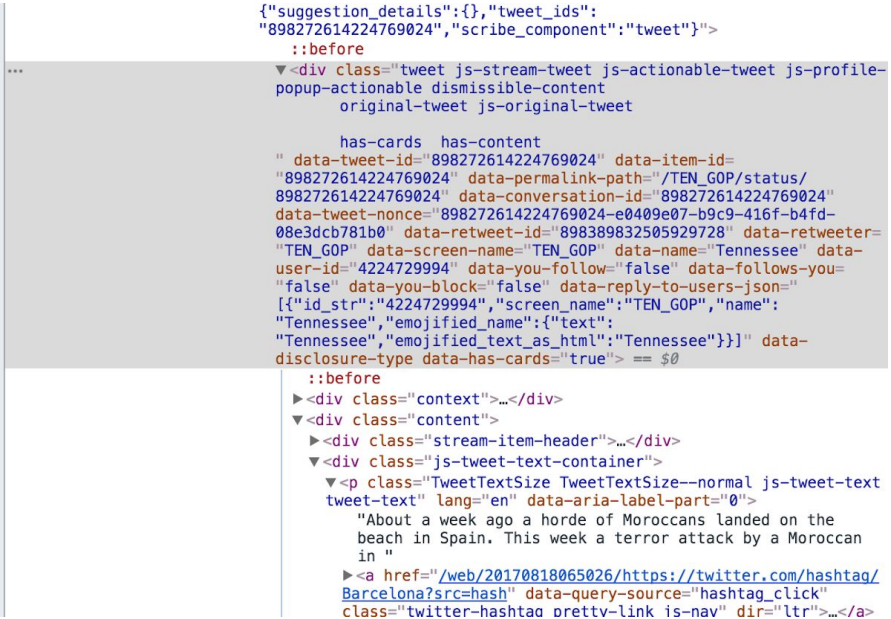
http://archive.org/wayback/available?url=http://twitter.com/TEN_GOP

```
{
    url: "http://twitter.com/TEN_GOP",
  - archived_snapshots: {
      - closest: {
            status: "200",
            available: true,
            url: "http://web.archive.org/web/20170818065026/https://twitter.com/TEN_GOP",
            timestamp: "20170818065026"
        }
    }
}
```


INTERNET ARCHIVE

http://www.lyonwj.com/2017/11/12/scraping-russian-twitter-trolls-python-neo4j/

Scraping Internet Archive...



http://www.lyonwj.com/2017/11/12/scraping-russian-twitter-trolls-python-neo4j/

Scraping Internet Archive...


```
[
    {
        "tweet_id": 561931644785811457,
        "screen_name": "AlwaysHungryBae",
        "permalink": "/AlwaysHungryBae/status/561931644785811457",
        "tweet_text": "Happy Super Bowl Sunday \n#superbowlfood pic.twitter.com/s6rwMtdLom",
        "user_id": "2882130846",
        "timestamp": 1422809918000,
        "hashtags": [
            {
                "tag": "superbowlfood",
                "archived_url": "/web/20150603004258/https://twitter.com/hashtag/superbowlfood?src=hash"
            }
        ],
        "links": [
            {
                "url": "pic.twitter.com/s6rwMtdLom",
                "archived_url": "http://web.archive.org/web/20150603004258/http://t.co/s6rwMtdLom"
            },
            {
                "url": "https://pbs.twimg.com/media/B8xh2fFCQAE-vxU.jpg:large",
                "archived_url": "//web.archive.org/web/20150603004258/https://twitter.com/AlwaysHungryBae/status/561931644785811457/photo/1"
            }
        ]
    }
]
```


http://www.lyonwj.com/2017/11/12/scraping-russian-twitter-trolls-python-neo4j/

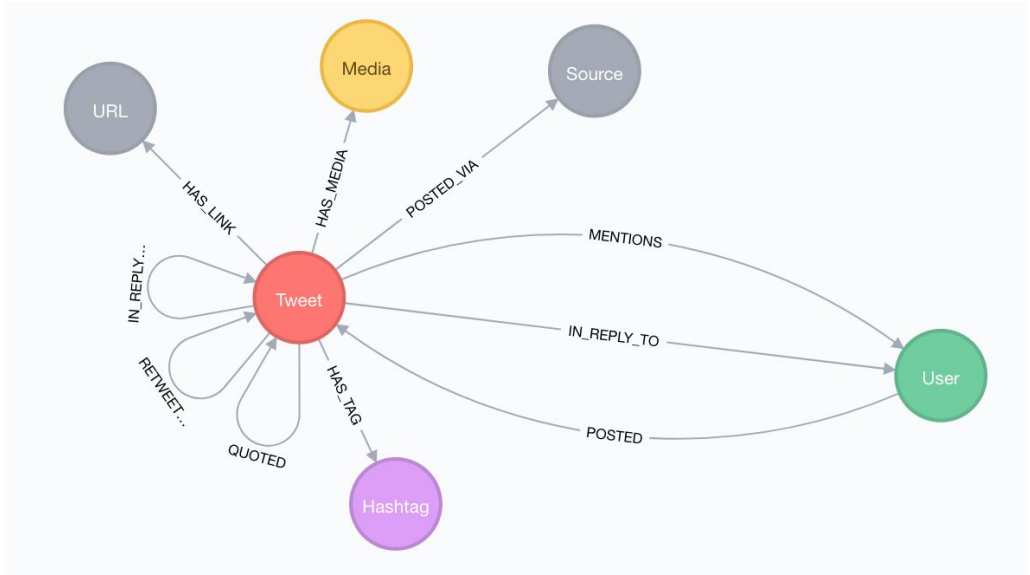
Scraping Internet Archive...

```
WITH $tweetArr AS tweets
UNWIND tweets AS tweet

MERGE (u:User {user_id: tweet.user_id})
ON CREATE SET u.screen_name = tweet.screen_name

MERGE (t:Tweet {tweet_id: tweet.tweet_id})
ON CREATE SET t.text = tweet.tweet_text,
              t.permalink = tweet.permalink

MERGE (u)-[:POSTED]->(t)

FOREACH (ht IN tweet.hashtags |
  MERGE (h:Hashtag {tag: ht.tag })
  ON CREATE SET h.archived_url = ht.archived_url
  MERGE (t)-[:HAS_TAG]->(h)
)

FOREACH (link IN tweet.links |
  MERGE (l:Link {url: link.url})
  ON CREATE SET l.archived_url = link.archived_url
  MERGE (t)-[:HAS_LINK]->(l)
)
```

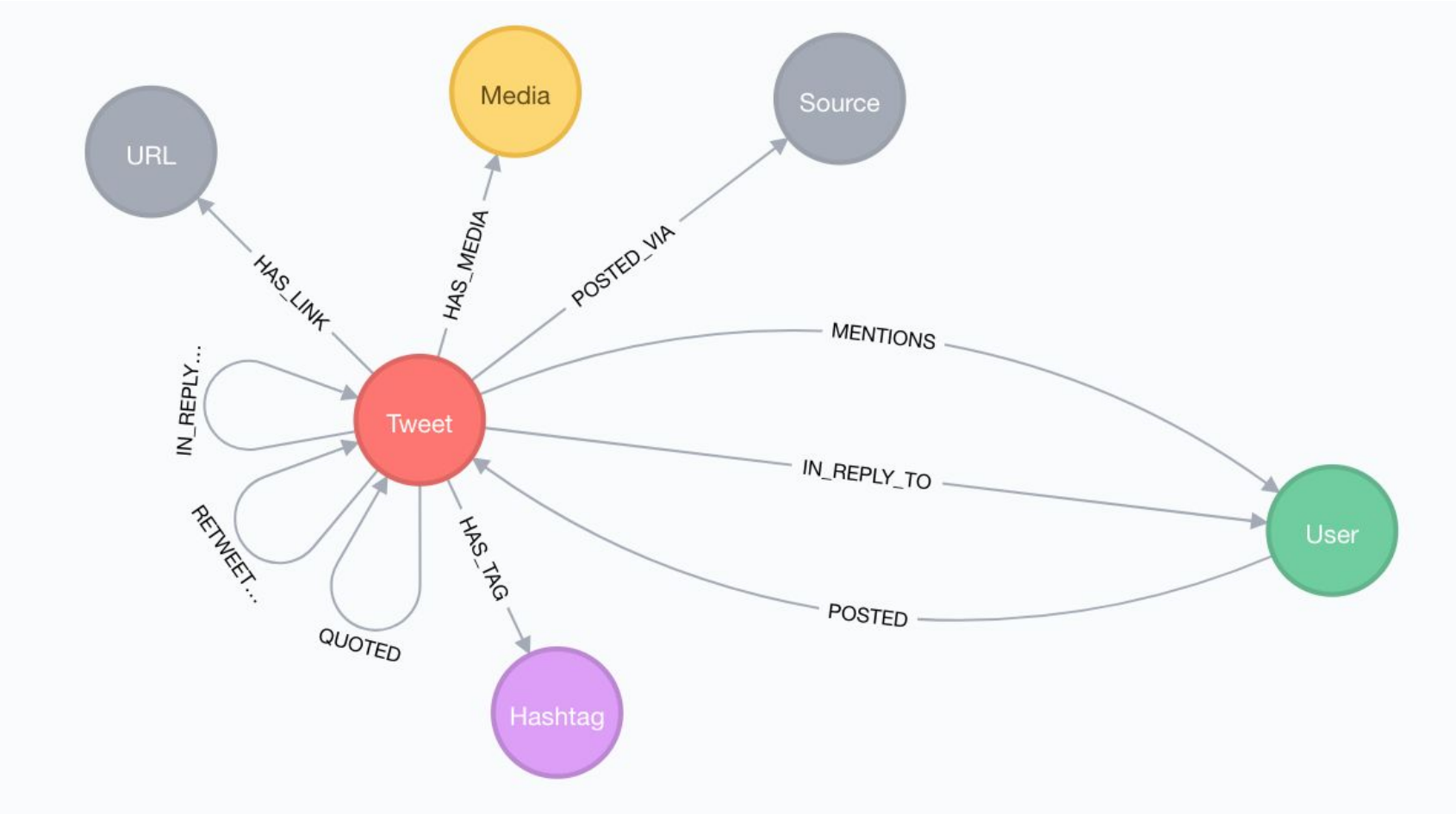
http://www.lyonwj.com/2017/11/12/scraping-russian-twitter-trolls-python-neo4j/

Scraping Internet Archive...

```
WITH $tweetArr AS tweets
UNWIND tweets AS tweet

MERGE (u:User {user_id: tweet.user_id})
ON CREATE SET u.screen_name = tweet.screen_name

MERGE (t:Tweet {tweet_id: tweet.tweet_id})
ON CREATE SET t.text = tweet.tweet_text,
              t.permalink = tweet.permalink

MERGE (u)-[:POSTED]->(t)

FOREACH (ht IN tweet.hashtags |
  MERGE (h:Hashtag {tag: ht.tag })
  ON CREATE SET h.archived_url = ht.archived_url
  MERGE (t)-[:HAS_TAG]->(h)
)

FOREACH (link IN tweet.links |
  MERGE (l:Link {url: link.url})
  ON CREATE SET l.archived_url = link.archived_url
  MERGE (t)-[:HAS_LINK]->(l)
)
```



http://www.lyonwj.com/2017/11/12/scraping-russian-twitter-trolls-python-neo4j/

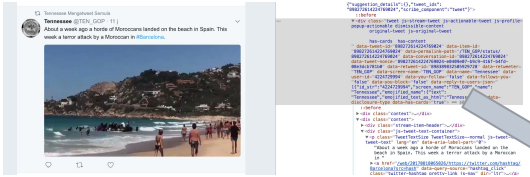


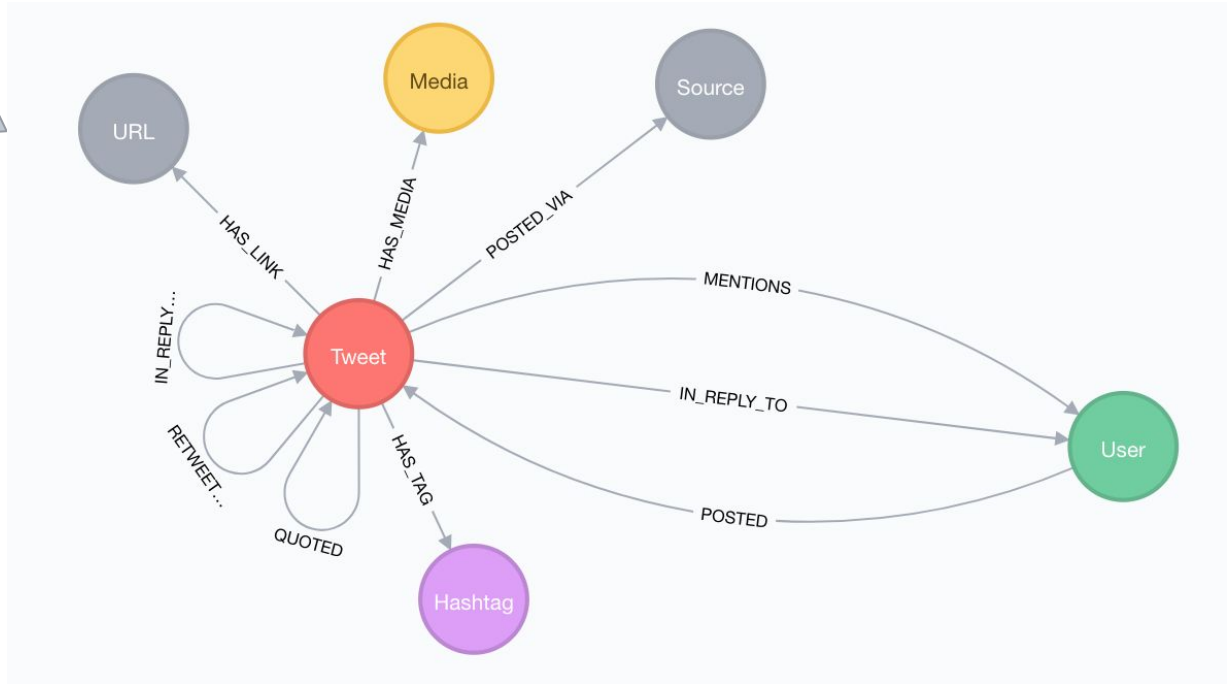
http://www.lyonwj.com/2017/11/12/scraping-russian-twitter-trolls-python-neo4j/

Passive Capture From Twitter API

```
[
  {
    "contributors": null,
    "coordinates": null,
    "created_at": "Tue Oct 25 09:04:03 +0000 2016",
    "display_text_range": [
      0,
      140
    ],
    "entities": {
      "hashtags": [],
      "symbols": [],
      "urls": [
        {
          "display_url": "twitter.com/i/web/status/7…",
          "expanded_url": "https://twitter.com/i/web/status/790841341643198464",
          "indices": [
            117,
            140
          ],
          "url": "https://t.co/J437581mKq"
        }
      ],
      "user_mentions": []
    },
    "extended_tweet": {
      "display_text_range": [
        0,
        137
      ],
      "entities": {
        "hashtags": [],
        "symbols": [],
        "urls": [
          {
            "display_url": "twitter.com/danielradosh/s…",
            "expanded_url": "https://twitter.com/danielradosh/status/790568116023615488",
            "indices": [
              138,
              161
            ],
            "url": "https://t.co/CAApSbra5y"
          }
        ],
        "user_mentions": []
      },
      "full_text": "I'm sure parents concerned about the profanity (?) in Fahrenheit 451 are also Trump supporters who h
ave no qualms about Trump's vulgarity https://t.co/CAApSbra5y"
    },
    "favorite_count": 0,
    "favorited": false,
    "filter_level": "low",
    "geo": null,
    "id": 790841341643198464,
    "id_str": "790841341643198464",
```

345k **Tweets**, 41k **Users** (454 Russian **Trolls**)





Your typical Russian Twitter Troll?

Your typical American Citizen?

@LeroyLovesUSA

Cleveland Online
@OnlineCleveland

Breaking news, weather, traffic and more for Cleveland. DM us anytime. RTs not endorsements

City of Cleveland, USA

@ClevelandOnline

Your typical Local News Publication?

Your typical Local Political Party?

@TEN_GOP

Your typical **Russian Troll**

@LeroyLovesUSA

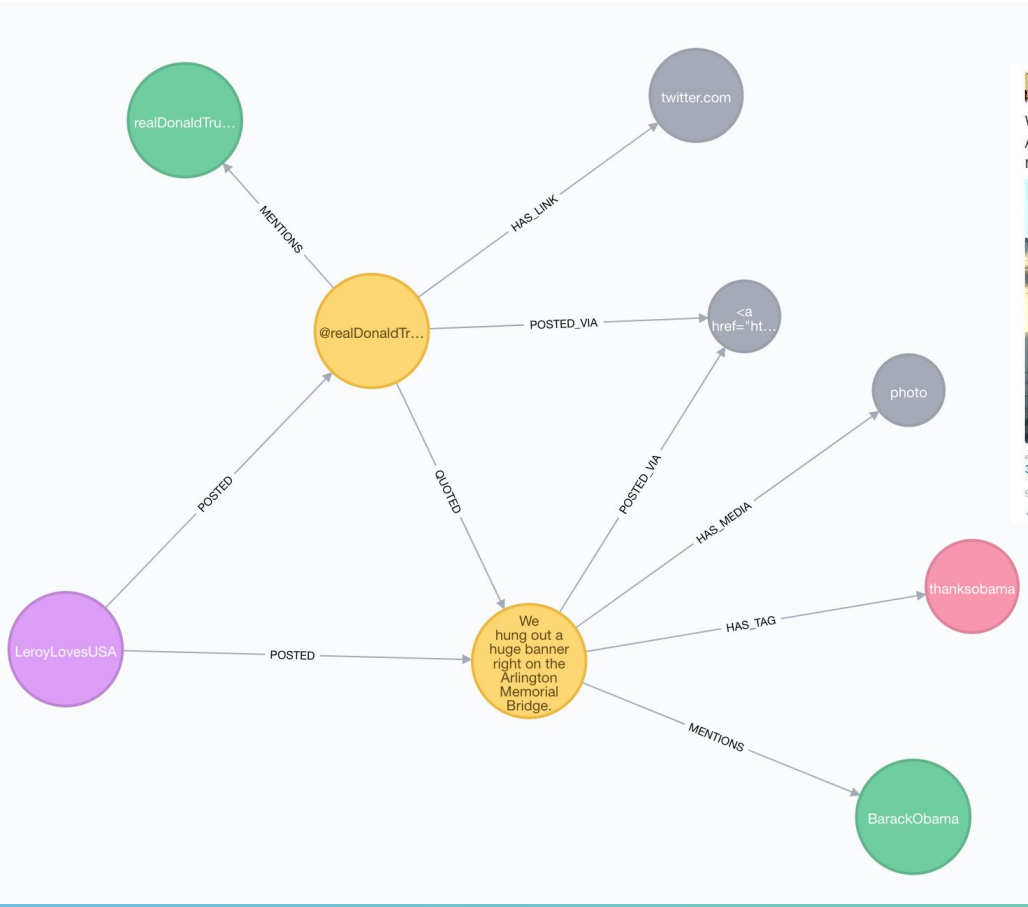
@ClevelandOnline

Your typical **Russian Troll**

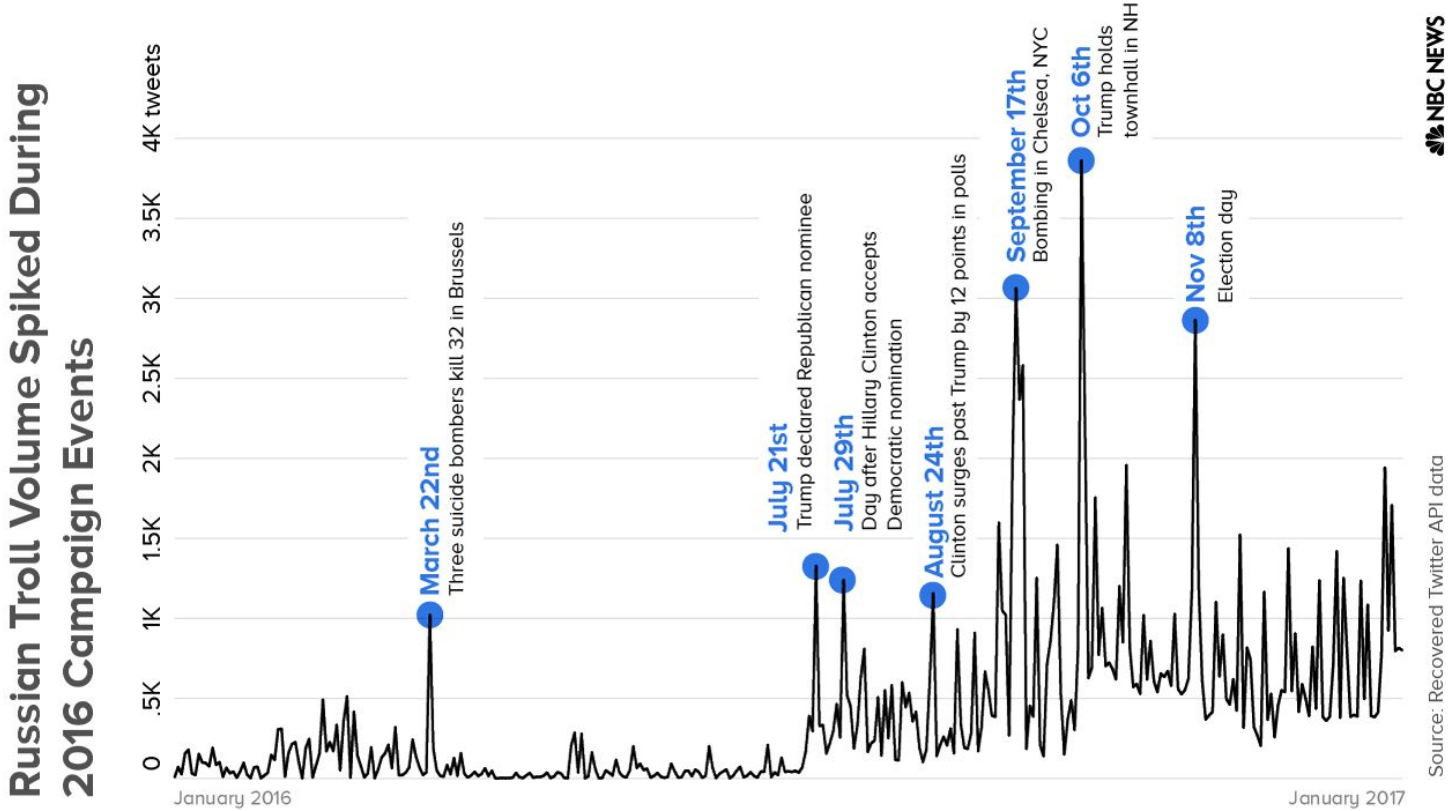
Your typical **Russian Troll**

@TEN_GOP

```
MATCH
(u:User {screen_name: "LeroyLovesUSA"})-[:POSTED]->(t:Tweet)-[:HAS_TAG]->(ht:Hashtag {key: "thanksobama"})
RETURN *
```



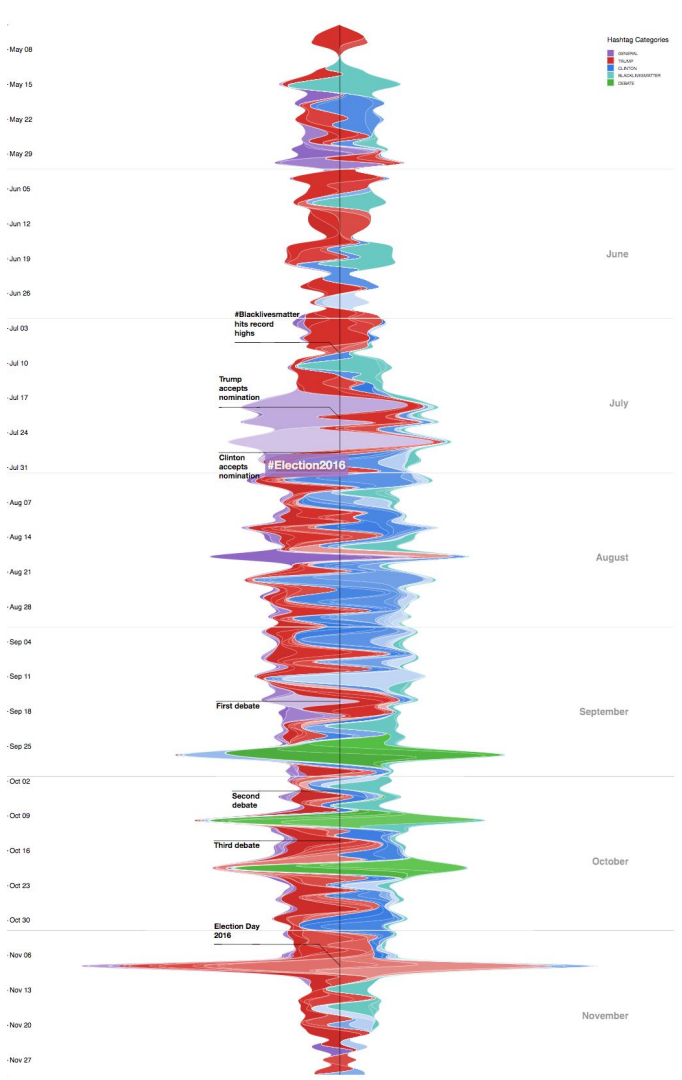
```
MATCH (t:Tweet)<-[:POSTED]-(u:Troll)
RETURN t.dayOfYear AS day, COUNT(*) AS num ORDER BY day
```



https://www.nbcnews.com/tech/social-media/russian-trolls-went-attack-during-key-election-moments-n827176

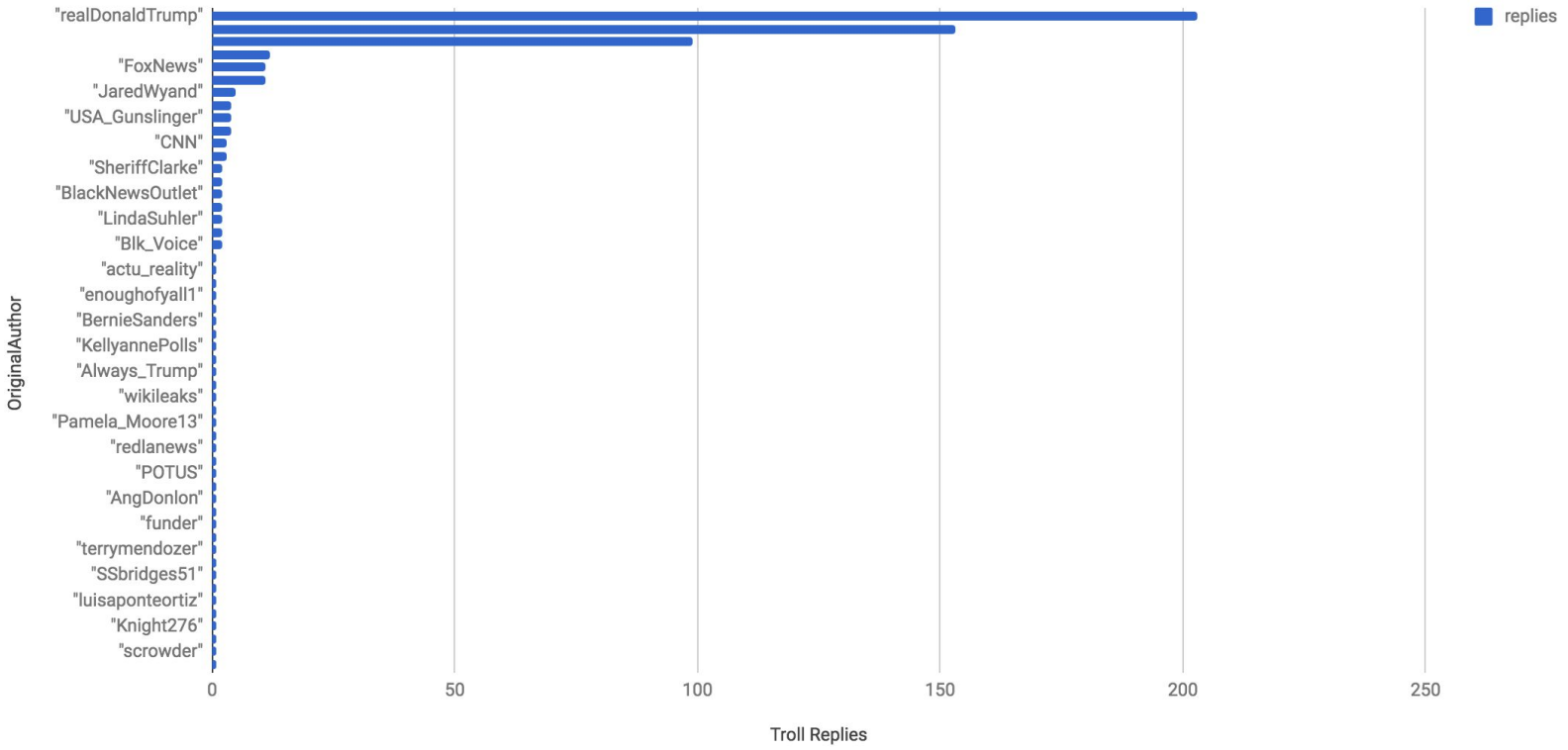
Hashtags

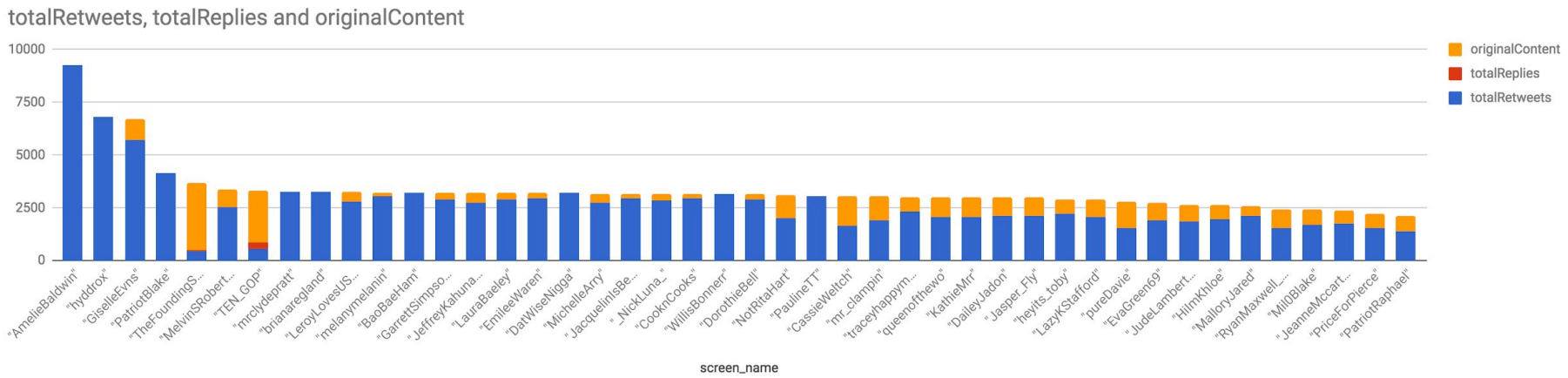
- Use of hashtags to gain visibility and insert into conversation
- **@WorldOfHashtags**
  - #RejectedDebateTopics

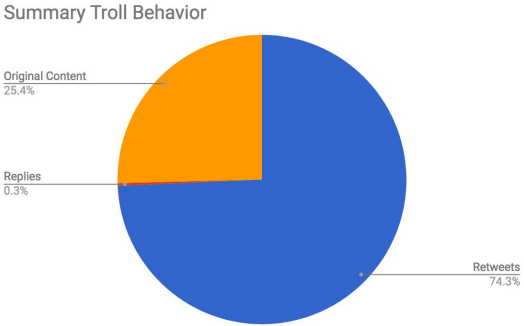


https://www.nbcnews.com/tech/social-media/russian-trolls-went-attack-during-key-election-moments-n827176

Replies vs. Original Author







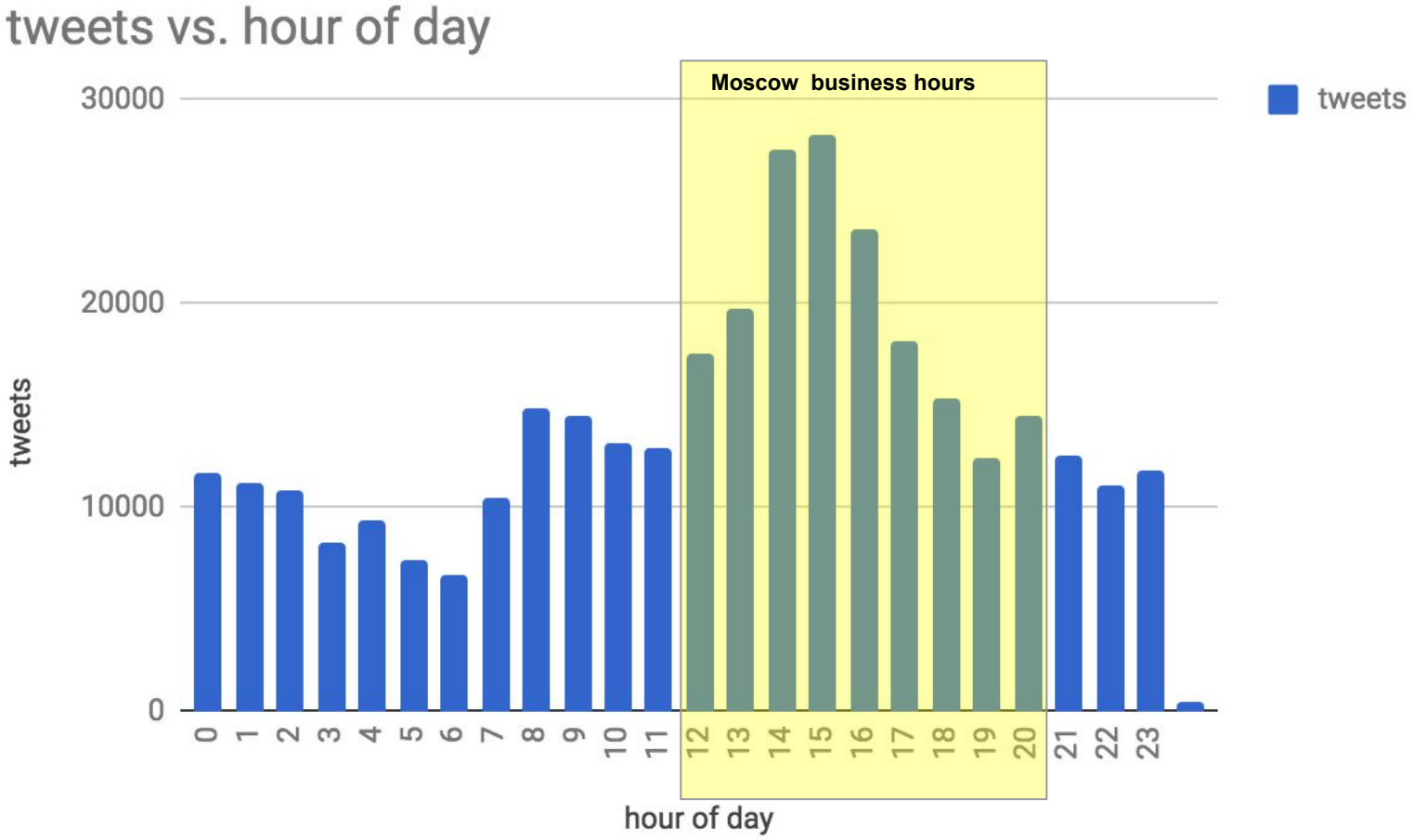

```
MATCH (tr:Troll)-[:POSTED]->(tw:Tweet) WITH tr, tw
OPTIONAL MATCH (tw)-[:RETWEETED]-(rt:Tweet)
OPTIONAL MATCH (tw)-[:IN_REPLY_TO]-(irp:Tweet)
RETURN distinct tr.screen_name as screen_name, count(tw) as totalTweets,
  count(rt) as totalRetweets, count(irp) as totalReplies,
  (count(tw) - (count(rt) + count(irp))) as originalContent
ORDER BY totalTweets DESC;
```

```
1 MATCH (:Troll)-[:POSTED]->(tw:Tweet)-[:POSTED_VIA]->(s:Source)
2 RETURN DISTINCT s.name as source, COUNT(tw) as tweets
3 ORDER BY tweets DESC
```

Source	Tweets
Twitter Web Client	42684
twitterfeed	6926
TweetDeck	6409
Twibble.io	1491
dlvr.it	243
Twitter for iPhone	152
Twitter for Android	94
Twitter for iPad	21
Medium	17
Buffer	13
OS X	8
Wordpress.com	6
IFTTT	5
Google	3
Tweetbot for iOS	3
Put your button on any page!	3
pourtwi	2
Twitter for Android Tablets	1
NovaPress Publisher	1

```
1 MATCH (:Troll)-[:POSTED]->(oc:Tweet)-[:HAS_LINK]-(u:URL)
2 RETURN DISTINCT u.domain as DomainOfTrollLink, COUNT(u) as LinksPosted
3 ORDER BY LinksPosted desc;
```

DomainOfTrollLink	LinksPosted
"bit.ly"	3431
"twitter.com"	1597
"twibble.io"	945
"wapo.st"	394
"www.wcvb.com"	210
"sh.st"	203
"www.nytimes.com"	136
"politics.blog.ajc.com"	104
"www.huffingtonpost.com"	98
"dailym.ai"	84
"www.cbsnews.com"	77
"cbsloc.al"	68
"politi.co"	64
"www.chron.com"	61
"www.usatoday.com"	56
"bsun.md"	56
"www.breitbart.com"	54
"www.washingtonpost.com"	54
"www.cbs8.com"	53
"www.wbaltv.com"	50
"www.nbcchicago.com"	49
"www.reuters.com"	49
"www.chicagotribune.com"	48
"www.local10.com"	47
"www.youtube.com"	44
"www.azcentral.com"	41


tweets vs. hour of day
Moscow business hours
tweets
30000
20000
10000
0
tweets
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
hour of day

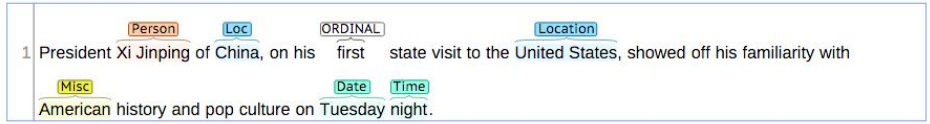
Natural Language Processing

With Cypher and Neo4j

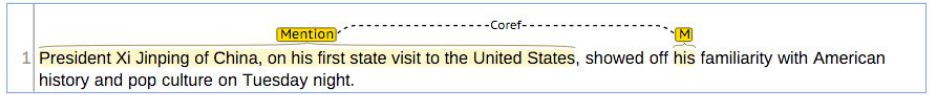
Common NLP Tasks

- Language detection
- Part of speech tagging
- Word similarities
- Keyword extraction
- Topic extraction
- Concept hierarchy
- Sentiment analysis
- Entity extraction
- Entity merging
- Word associations
- Summarization
- Opinion mining

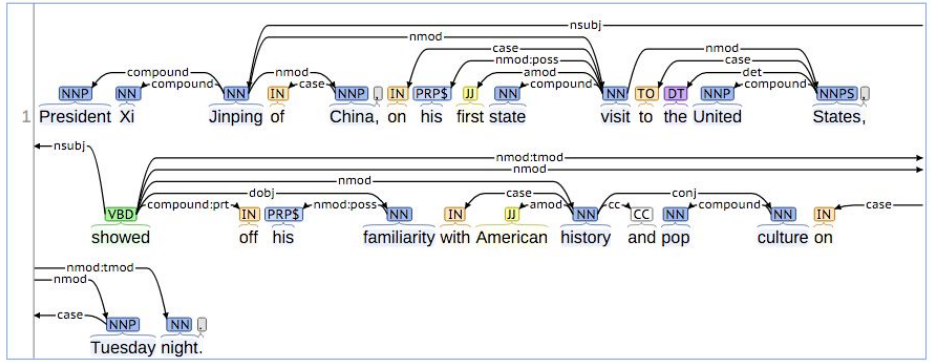
Named Entity Recognition:



Coreference:



Basic Dependencies:



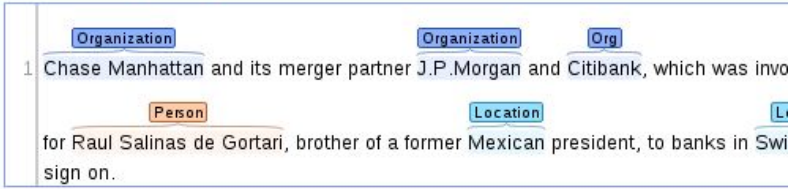
NLTK
Textblob
Polyglot
TextRank

ConceptNet
An open, multilingual knowledge graph

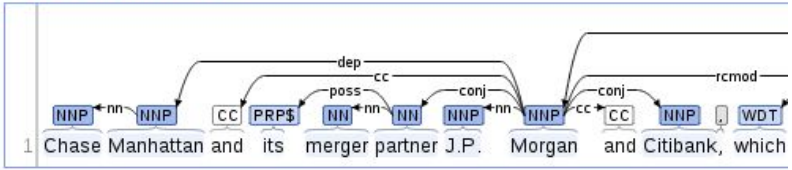
NLP w/ Graph Databases

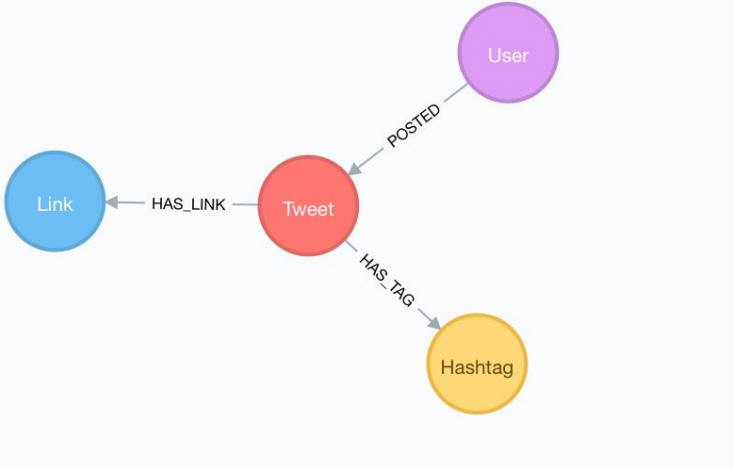
Annotations

Named Entity Recognition:

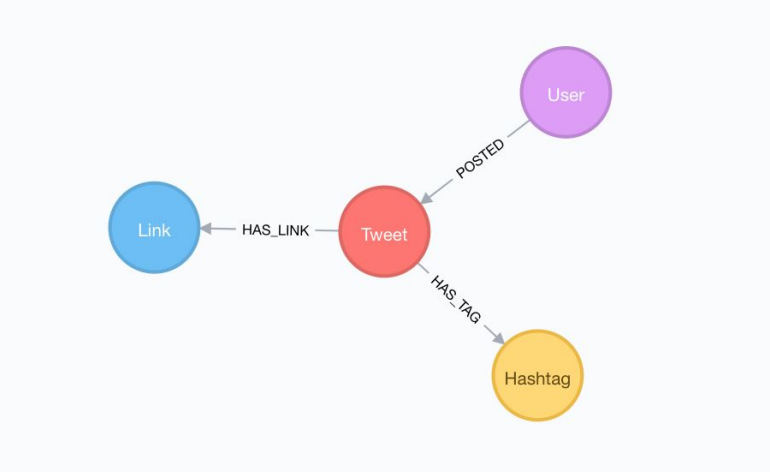


Basic dependencies:

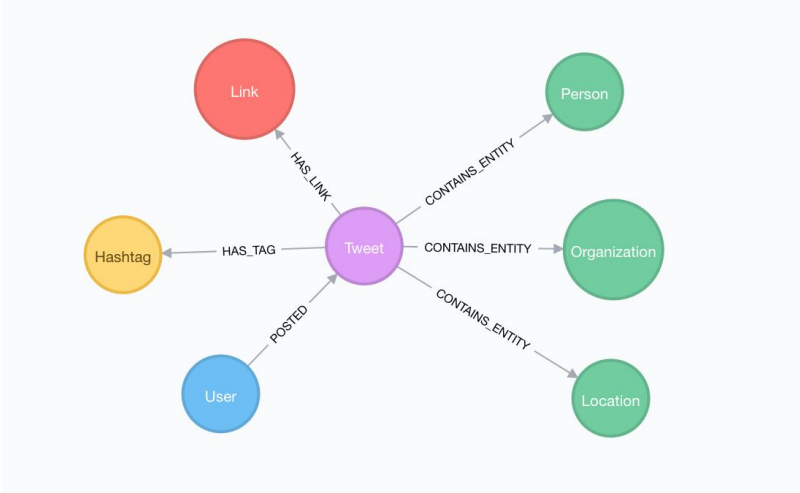




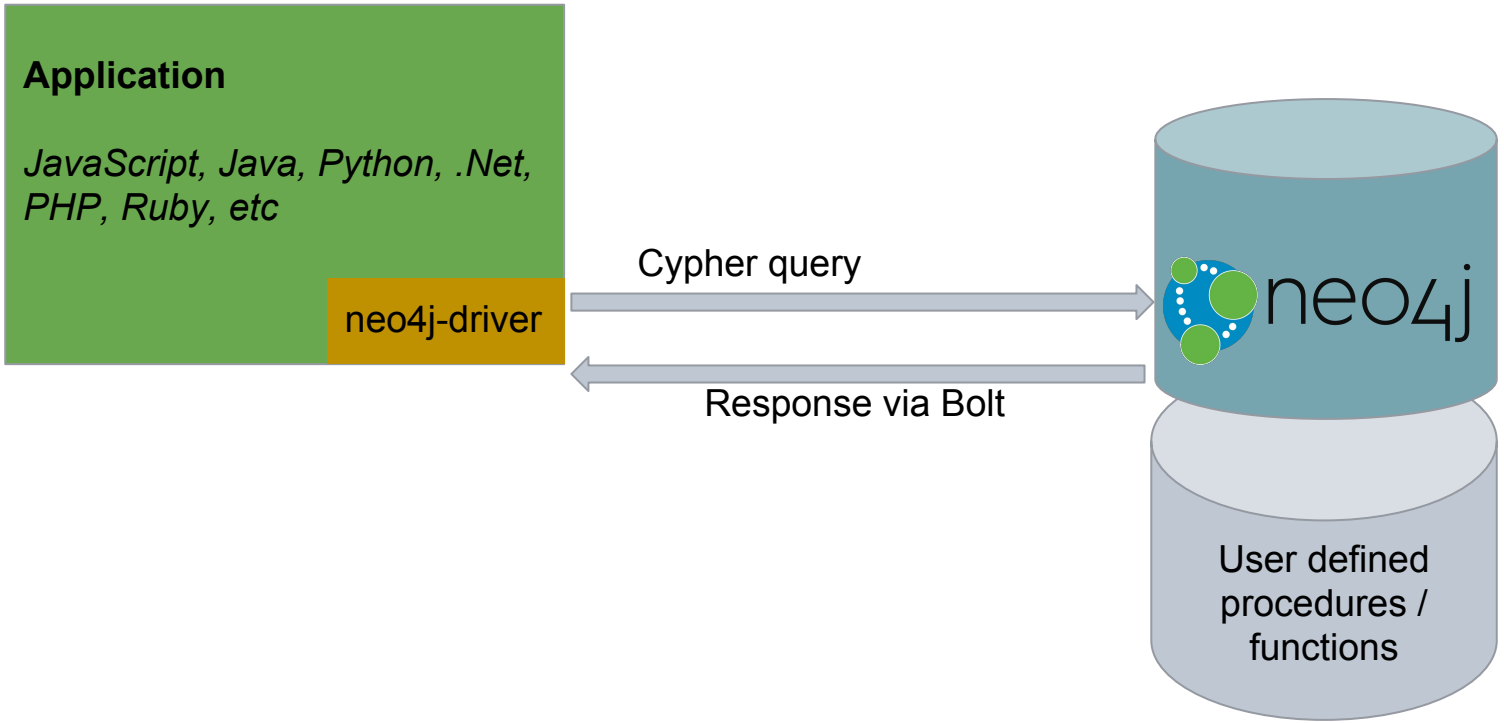
NLP w/ Graph Databases *Annotations*



NLP Process



How to combine open source NLP tools w/ Neo4j?

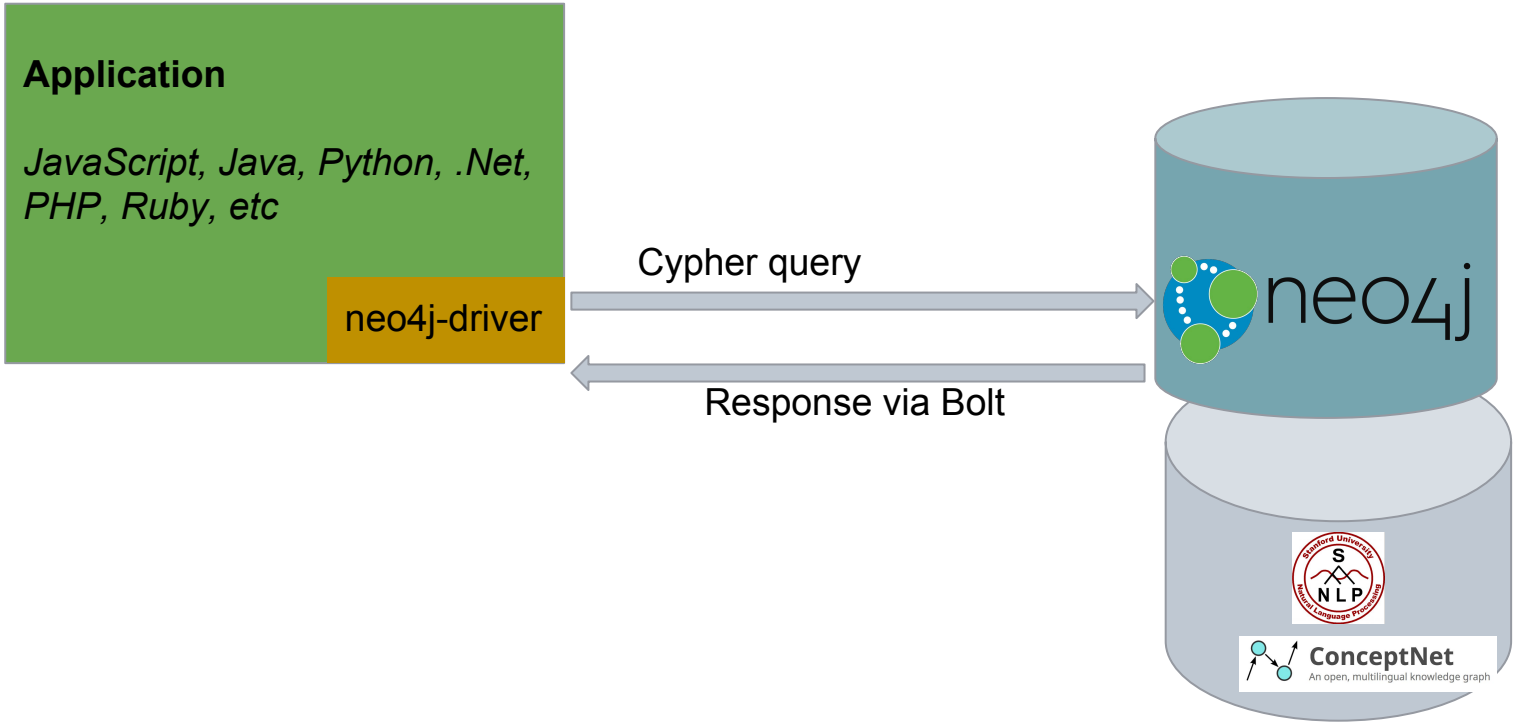


How to combine open source NLP tools w/ Neo4j?

```
public class YourProcedureClass {
     @Context
     public GraphDatabaseService db;  // Inject the database for easy access

     @Procedure
     @Description("<explain yourProcedure here>")
     public Stream<OutputRecord> yourProcedure(@Name("arg1") SomeType arg1, … )
     {
          // TODO: Your implementation
     }
}

public class OutputRecord {
     public final SomeType value; // Your result fields
     ...

     public OutputRecord(SomeType value)
     {
          this.value = value;
     }
}
```

Name of Procedure Output

How to combine open source NLP tools w/ Neo4j?



How to combine open source NLP tools w/ Neo4j?

GraphAware Natural Language Processing

build passing This Neo4j plugin offers Graph Based Natural Language Processing capabilities.

```
1 MATCH (tw:Tweet {lang: "en"})
2 CALL ga.nlp.annotate({text: tw.text, id: id(tw)})
3 YIELD result
4 MERGE (tw)-[:HAS_ANNOTATED_TEXT]->(result)
5 RETURN count(result)
```

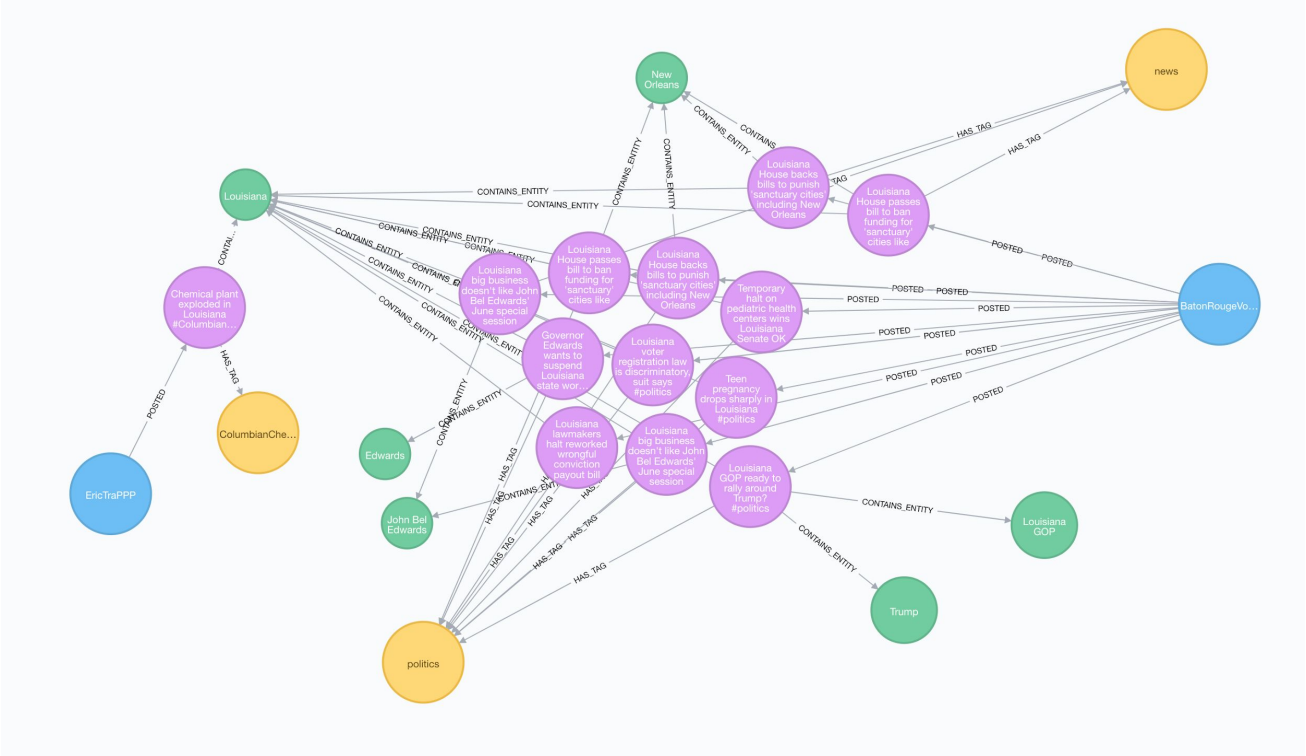
https://github.com/graphaware/neo4j-nlp

PEOPLE MENTIONED	
Person	**Num**
"donald trump"	5676
"hillary clinton"	4792
"bill clinton"	800
"mike pence"	376
"tim kaine"	370
"michelle obama"	358
"bernie sanders"	218
"barack obama"	190
"rudy giuliani"	188
"david clarke"	170
"ted cruz"	164
"ben carson"	138
"sheriff clarke"	136
"newt gingrich"	136
"topnews obama"	128
"george soros"	114
"colin powell"	106
"topvideo trump"	88
"trump foundation"	86
"paul ryan"	86
"chelsea clinton"	84
"topvideo obama"	80
"reagan"	78

LOCATIONS MENTIONED	
Location	**Num**
"white house"	760
"new york"	232
"united states"	188
"milwaukee"	142
"north carolina"	138
"munich"	126
"saudi arabia"	104
"baltimore"	104
"philippines"	84
"paris"	84
"north korea"	78
"st louis"	76
"pittsburgh"	68
"trump tower"	68
"utah"	68
"baton rouge"	56
"middle east"	56
"new york city"	56
"laos"	52
"atlantic city"	44
"new jersey"	44

ORGANIZATIONS MENTIONED	
Organization	**Num**
"white house"	760
"new york"	232
"north carolina"	138
"hillarys health"	106
"democratic party"	106
"supreme court"	100
"republican party"	98
"state department"	92
"democratic national convention"	84
"gop convention"	76
"trump tower"	68
"trump university"	58
"justice department"	56
"islamic state"	52
"secret service"	46
"walk of fame"	44
"house gop"	40
"air force"	40
"rust belt"	36
"nba"	36

```
// Tweets, hashtags, and entities around Louisiana
MATCH (u:User)-[:POSTED]->(t:Tweet)-[:CONTAINS_ENTITY]->(l:Location {name: "Louisiana"})
OPTIONAL MATCH (t)-[:HAS_TAG]->(ht:Hashtag)
OPTIONAL MATCH (t)-[:CONTAINS_ENTITY]->(e)
RETURN *
```



http://www.lyonwj.com/2017/11/15/entity-extraction-russian-troll-tweets-neo4j/

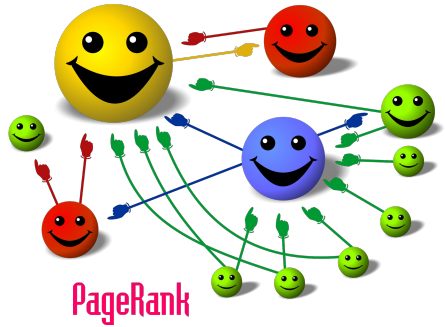
Graph Algorithms

Centrality

Measure of importance

PageRank

- Recursive
- Importance and number of connected nodes

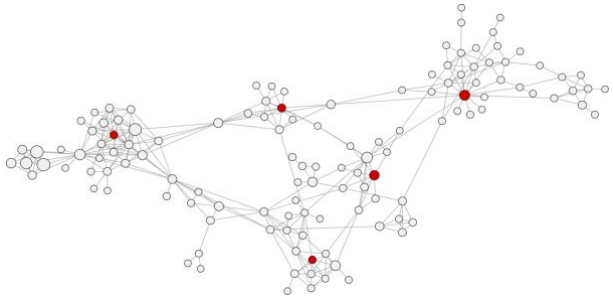


Betweenness Centrality

- Number of shortest paths connecting all pairs in the network

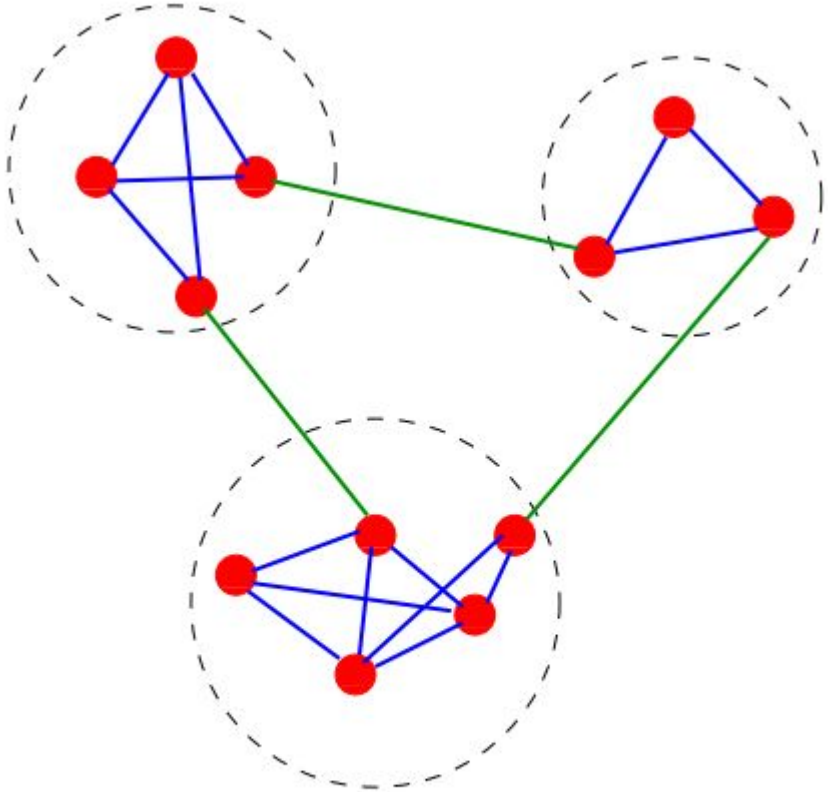
Closeness Centrality

- Inverse of distance to all other nodes in the network

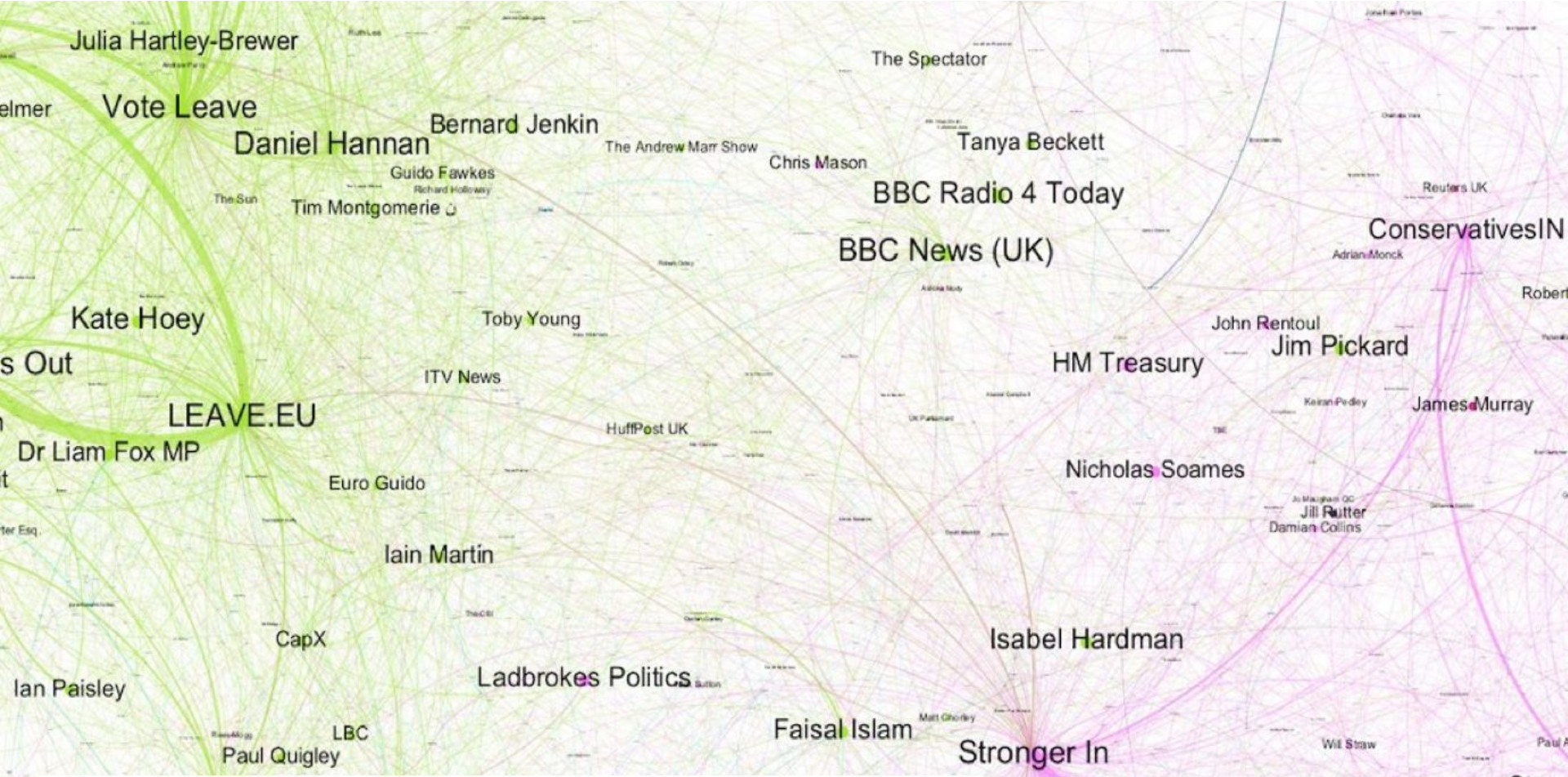


Community Detection / Clustering

- Segment graph
- Minimize modularity
- Iterative Algorithms
  - Louvain
  - Union Find
  - Walktrap

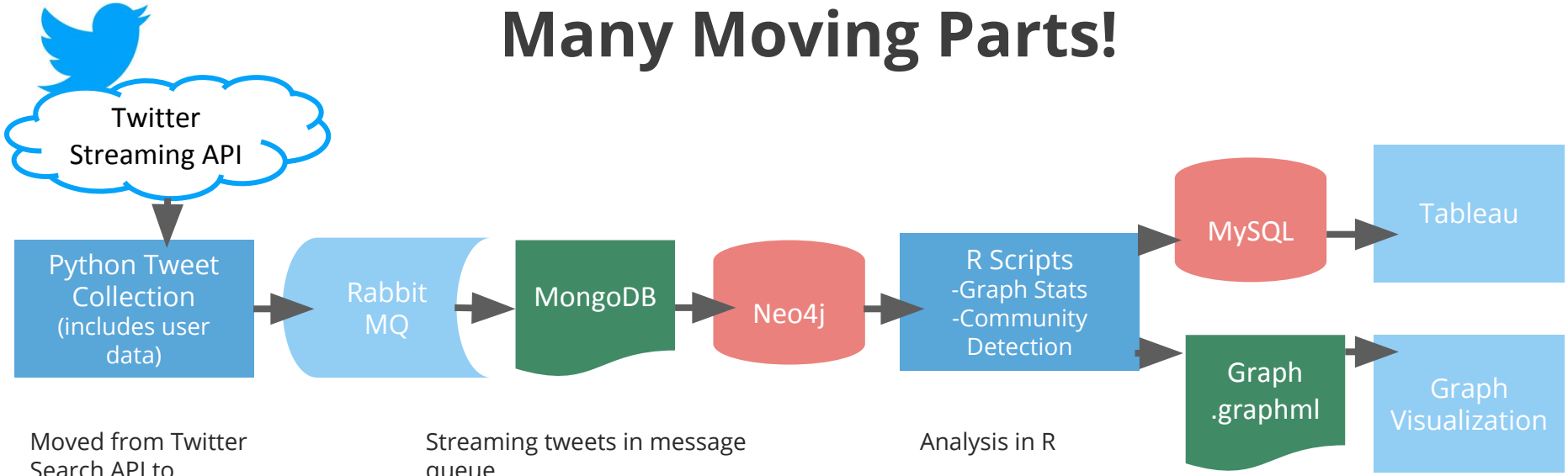


How to run graph algorithms in Neo4j?



Source: John Swain - Twitter Analytics Right Relevance Talk

Many Moving Parts!



Moved from Twitter Search API to Streaming API

Replaced Python Twitter libraries (Tweepy) with raw API calls

Streaming tweets in message queue

Full tweets and user data stored in MongoDB

Built graph for analysis in Neo4j from tweets persisted in MongoDB

Analysis in R

iGraph libraries for algorithms

Some text analysis e.g. LDA topics

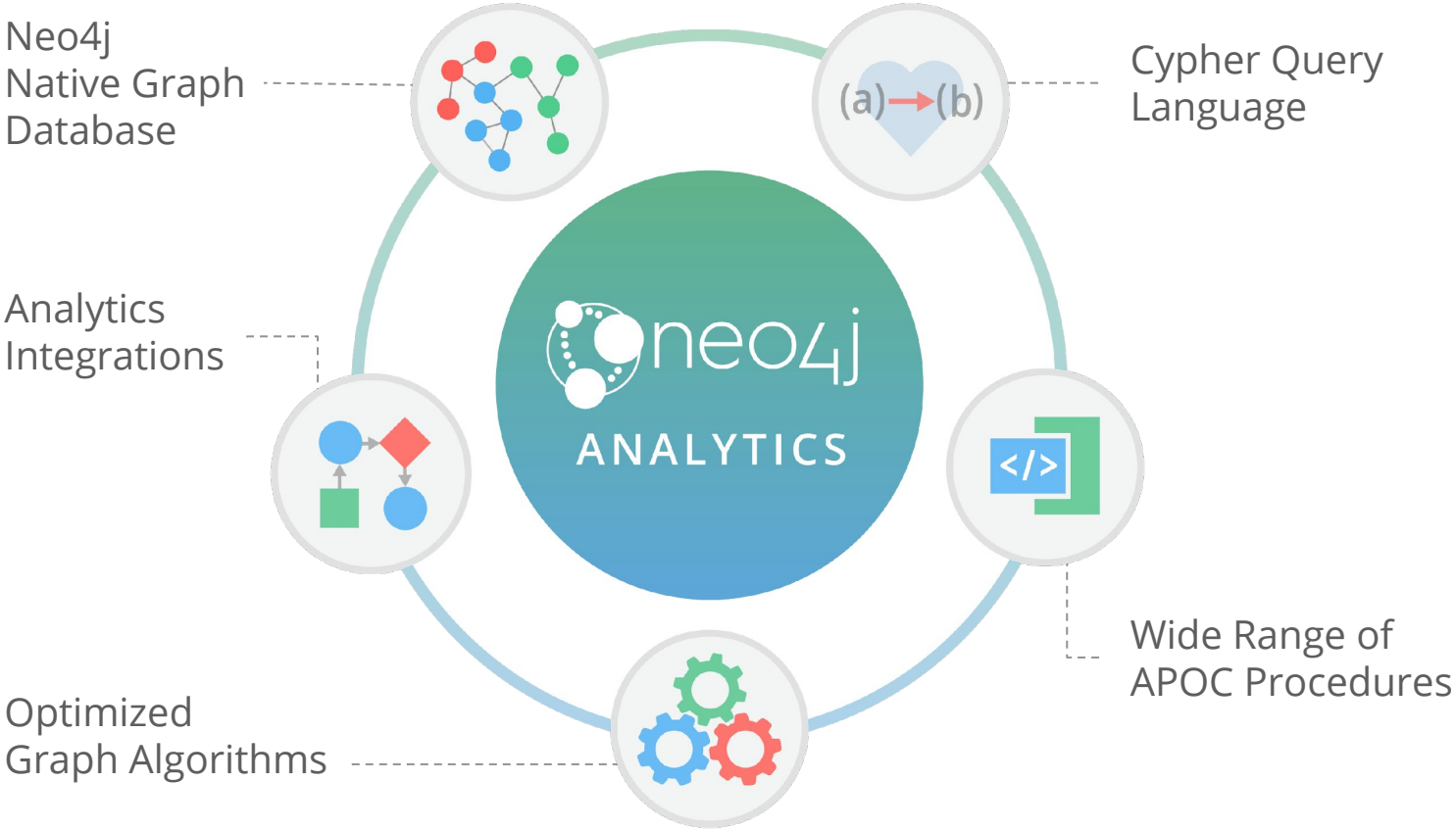
Results published in MySQL for Tableau

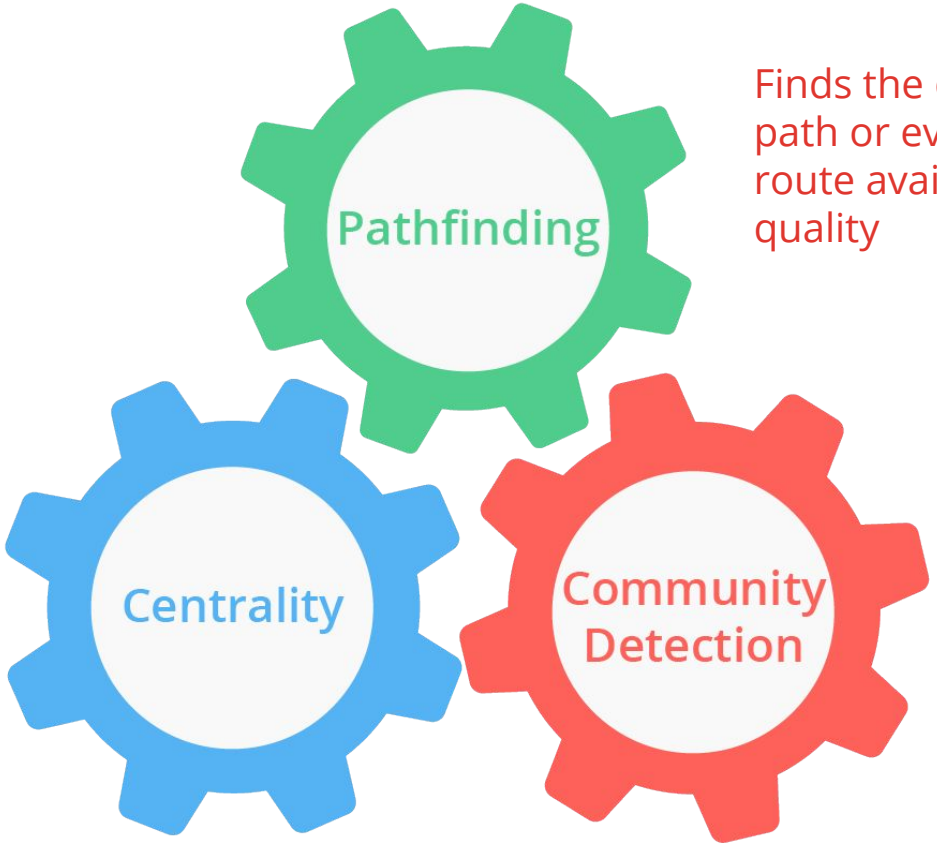
Graphml for import to Gephi with stats precalculated

Example Workflow Pipeline

Our Goal

Example Workflow Pipeline


Neo4j
Native Graph
Database
Cypher Query
Language
Analytics
Integrations
neo4j
ANALYTICS
Wide Range of
APOC Procedures
Optimized
Graph Algorithms



Finds the optimal path or evaluates route availability and quality

Determines the importance of distinct nodes in the network

Evaluates how a group is clustered or partitioned

Usage

1. Call as Cypher procedure
2. Pass in specification (Label, Prop, Query) and configuration
3. ~.stream variant returns (**a lot**) of results

```
CALL algo.<name>.stream('Label','TYPE',{conf})
YIELD nodeId, score
```

4. non-stream variant writes results to graph returns statistics

```
CALL algo.<name>('Label','TYPE',{conf})
```

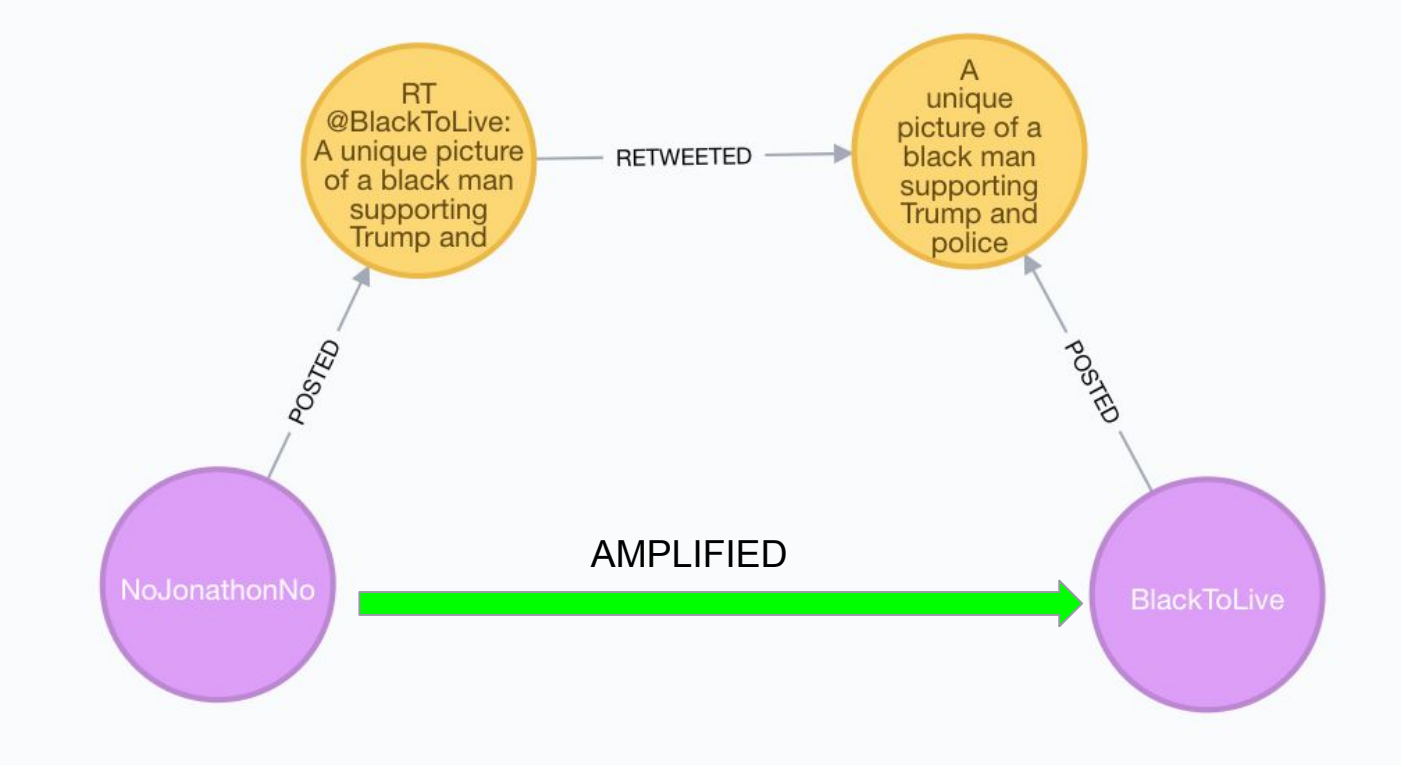
Cypher Projection

Pass in Cypher statement for node- and relationship-lists.

```
CALL algo.<name>(
  'MATCH ... RETURN id(n)',
  'MATCH (n)-->(m)
   RETURN id(n) as source,
          id(m) as target',   {graph:'cypher'})
```

Inferred Relationships

```
1 MATCH (r1:Troll)-[:POSTED]->(t1:Tweet)<-[:RETWEETED]-(t2:Tweet)<-[:POSTED]-(r2:Troll)
```

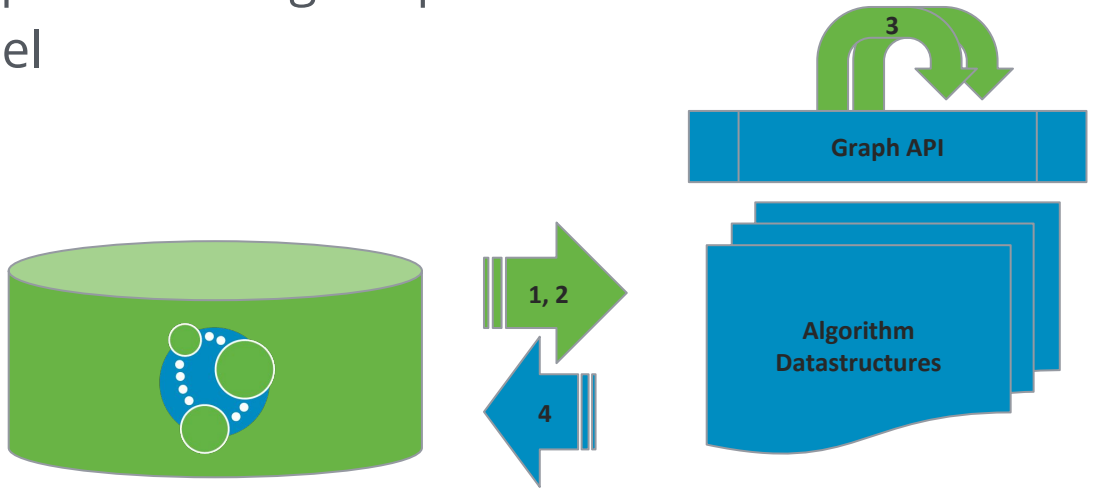


PageRank on Inferred AMPLIFIED Graph

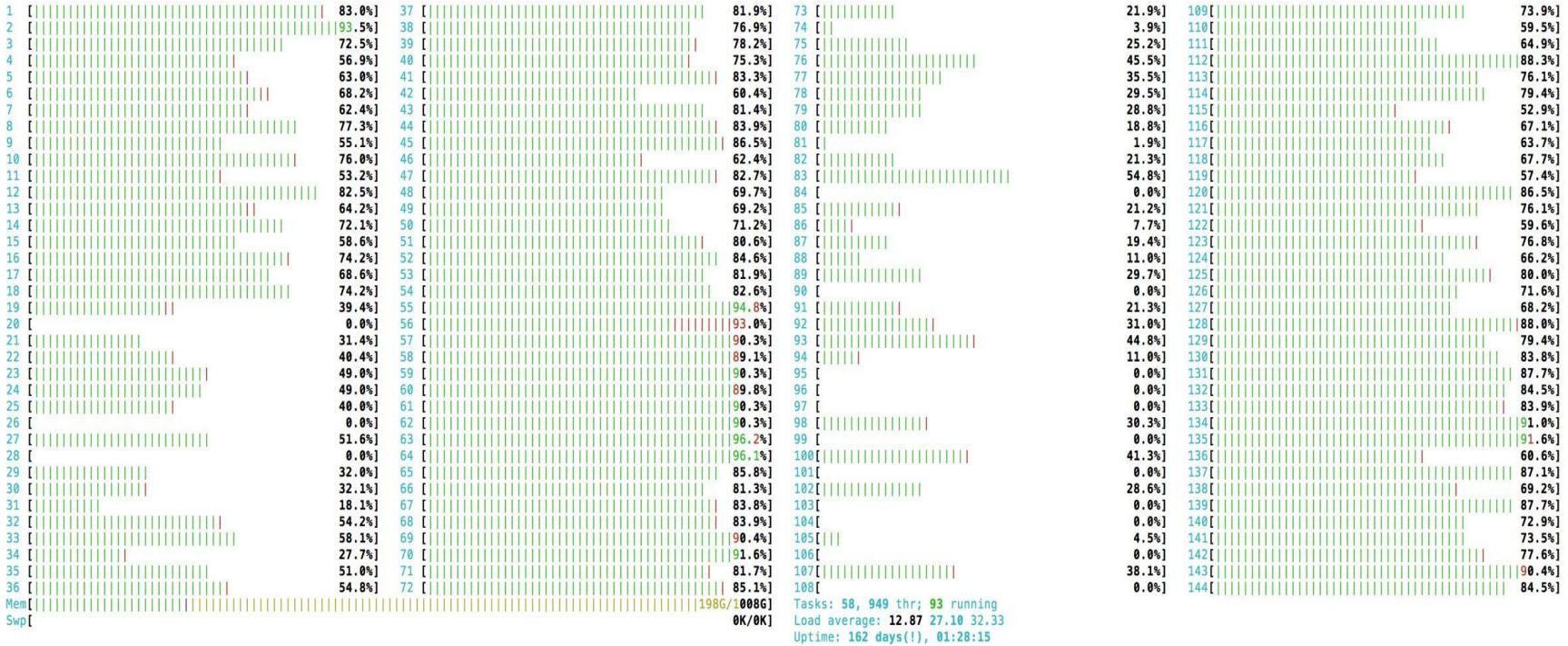
```
CALL algo.pageRank("MATCH
(r1:Troll)-[:POSTED]->(:Tweet)<-[:RETWEETED]-(:Tweet)<-[:POSTED]-(r2:Troll)
RETURN id(r2) as source, id(r1) as target",   {graph:'cypher'})
```

Architecture

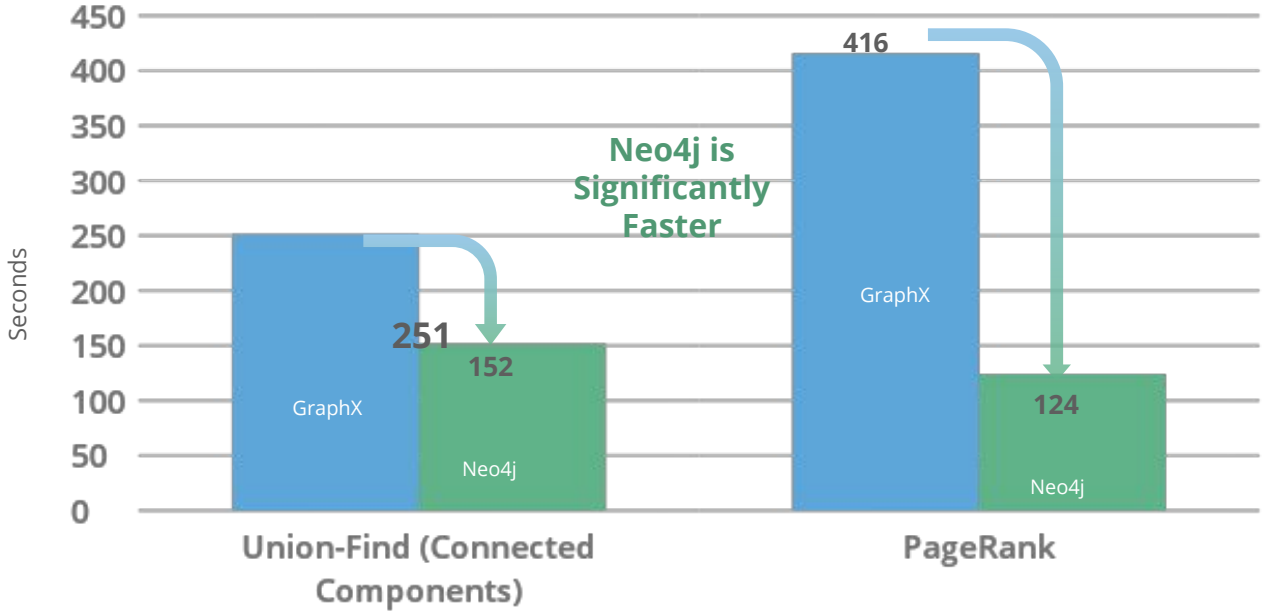
1. Load Data in parallel from Neo4j
2. Store in efficient data structures
3. Run Graph Algorithm in parallel using Graph API
4. Write data back in parallel



Scale: 144 CPU



Neo4j Graph Platform with Neo4j Algorithms vs. Apache Spark's GraphX



Twitter 2010 Dataset
- 1.47 Billion Relationships
- 41.65 Million Nodes

Spark GraphX results publicly available
- Amazon EC2 cluster running 64-bit Linux
- 128 CPUs with 68 GB of memory, 2 hard disks

Neo4j Configuration
- Physical machine running 64-bit Linux
- 128 CPUs with 55 GB RAM, SSDs

Compute At Scale – Payment Graph

3,000,000,000 nodes and 18,000,000,000 relationships (600G)
PageRank (20 iterations) on 1 machine, 20 threads, 900G RAM

```
call algo.pageRank('Account','SENT',
                   {graph:'huge',iterations:20,write:true,concurrency:20});

+-------------------------------------------------------------------+
| nodes     | iterations | loadMillis | computeMillis | writeMillis |
+-------------------------------------------------------------------+
| 300000000 | 20         | 401404     | 6024994       | 47106       |
+-------------------------------------------------------------------+
1 row 6473526 ms -> 1h 47min
```

Graph Visualization

Graph Visualization

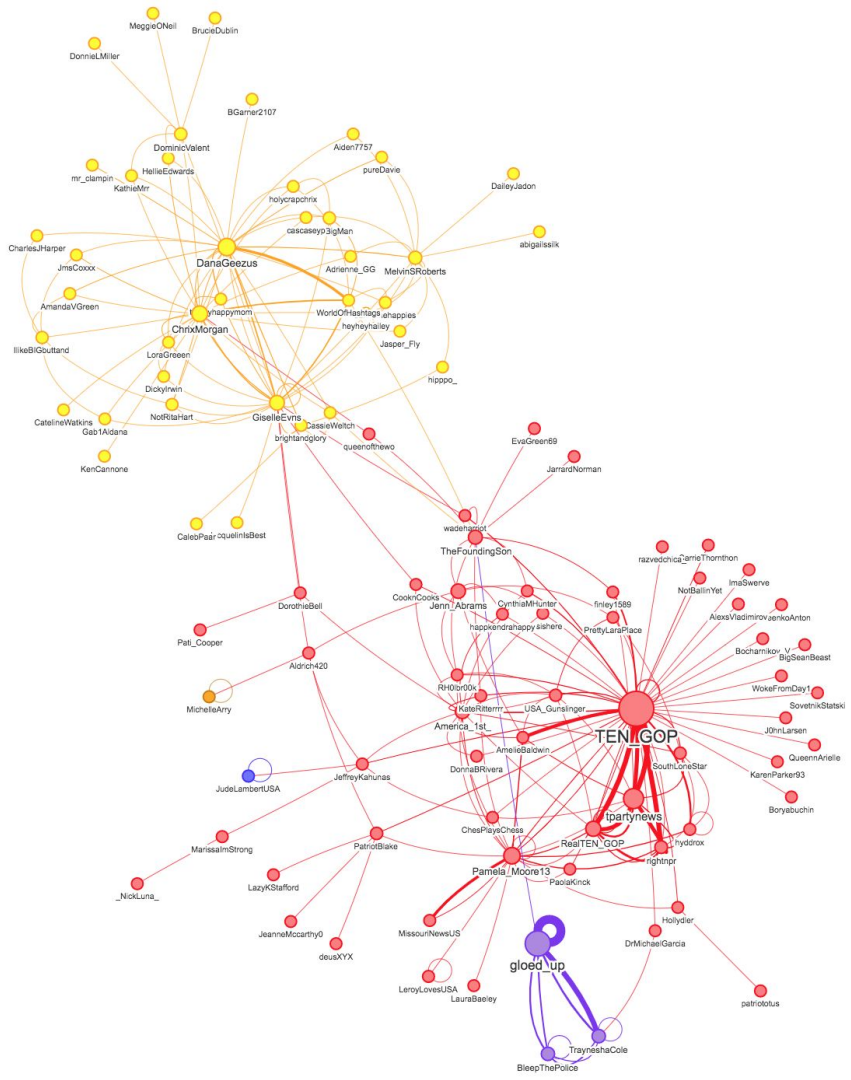
Centrality & community detection

AMPLIFIED relationships

Node size → PageRank

Color → community detection

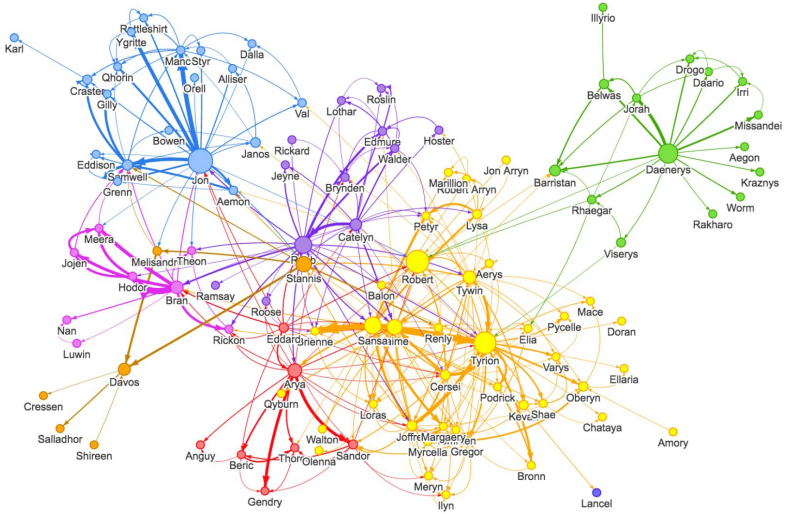
Rel Thickness → weight



Graph Visualization

neovis.js

Graph visualizations powered by vis.js with data from Neo4j.




```
var config = {
                container_id: "viz",
                server_url: "bolt://localhost:7687",
                server_user: "neo4j",
                server_password: "sorts-swims-burglaries",
                labels: {
                    //"Character": "name",
                    "Character": {
                        "caption": "name",
                        "size": "pagerank",
                        "community": "community"
                        //"sizeCypher": "MATCH (n) WHERE id(n) = {id} MATCH (n)-[r]-() RETURN sum(r.weight)
                    }
                },
                relationships: {
                    "INTERACTS": {
                        "thickness": "weight",
                        "caption": false
                    }
                },
                initial_cypher: "MATCH (n)-[r:INTERACTS]->(m) RETURN n,r,m"
            };

            viz = new NeoVis.default(config);
            viz.render();
```

https://github.com/johnymontana/neovis.js

Feb 14:

NEWS

Twitter Deleted 200,000 Russian Troll Tweets. Read Them Here.

GET THE DATA:

- *Regular reader?* Download streamlined spreadsheet (29 mb) with just usernames, tweet and timestamps. We recommend you right click on links and select "save link as" or similar, otherwise it may take a long time to load in your browser.
- View full data for ten influential accounts in Google Sheets
- *Researcher?* Download tweets.csv (50 mb) and users.csv with full underlying data
- Explore a graph database in Neo4j

Feb 16:

IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF COLUMBIA

UNITED STATES OF AMERICA	*	
	*	CRIMINAL NO.
v.	*	
	*	(18 U.S.C. §§ 2, 371, 1349, 1028A)
INTERNET RESEARCH AGENCY LLC A/K/A MEDIASINTEZ LLC A/K/A GLAVSET LLC A/K/A MIXINFO LLC A/K/A AZIMUT LLC A/K/A NOVINFO LLC,	*	
CONCORD MANAGEMENT AND CONSULTING LLC,	*	
CONCORD CATERING,	*	
YEVGENIY VIKTOROVICH PRIGOZHIN,	*	
MIKHAIL IVANOVICH BYSTROV,	*	
MIKHAIL LEONIDOVICH BURCHIK A/K/A MIKHAIL ABRAMOV,	*	
ALEKSANDRA YURYEVNA KRYLOVA,	*	
ANNA VLADISLAVOVNA BOGACHEVA,	*	
SERGEY PAVLOVICH POLOZOV,	*	
MARIA ANATOLYEVNA BOVDA A/K/A MARIA ANATOLYEVNA BELYAEVA,	*	
ROBERT SERGEYEVICH BOVDA,	*	
DZHEYKHUN NASIMI OGLY ASLANOV A/K/A JAYHOON ASLANOV A/K/A JAY ASLANOV,	*	
VADIM VLADIMIROVICH PODKOPAEV,	*	
GLEB IGOREVICH VASILCHENKO,	*	
IRINA VIKTOROVNA KAVERZINA,	*	
and	*	
VLADIMIR VENKOV.	*******	

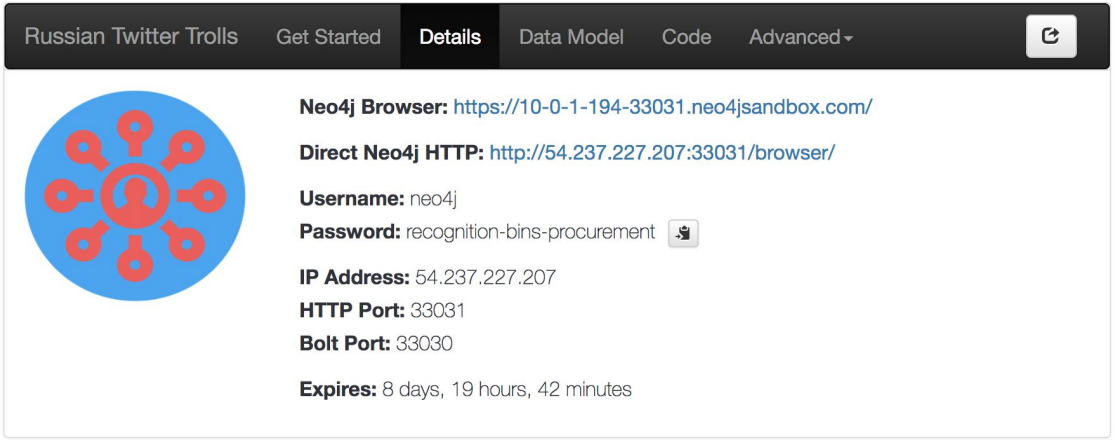
https://www.nbcnews.com/tech/social-media/now-available-more-200-000-deleted-russian-troll-tweets-n844731

Surprising Takeaways

- Amplifying w/ retweets
- Used social media automation tools
  - Not necessarily live responses
- Meddling in elections is just another 9-5 job
- **Data availability**

- See lyonwj.com for code, etc.
- https://www.nbcnews.com/tech/social-media/russian-trolls-went-attack-during-key-election-moments-n827176

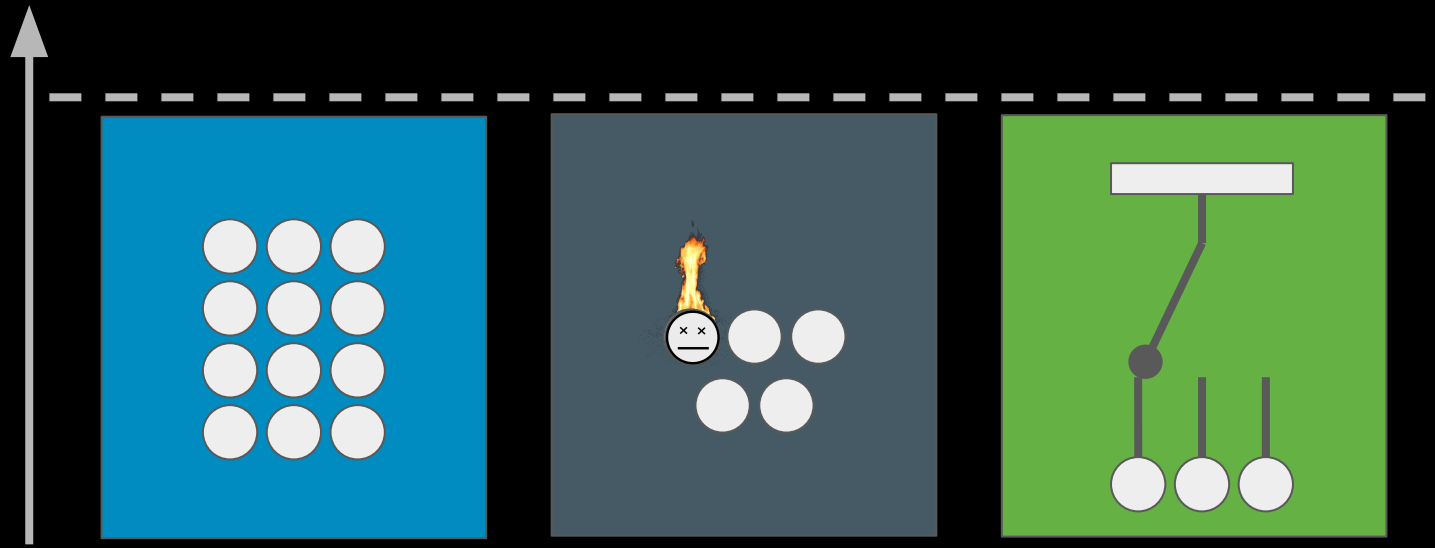
neo4jsandbox.com



https://hackernoon.com/six-ways-to-explore-the-russian-twitter-trolls-database-in-neo4j-6e52394c38f1

(you)-[:HAVE]->(questions)<-[:ANSWERS]-(will)

Neo4j Causal Clustering

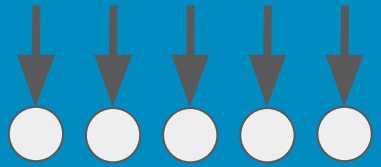


Massive Throughput

Data Redundancy

High Availability

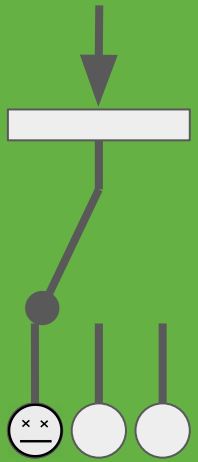
Massive Throughput



Data Redundancy

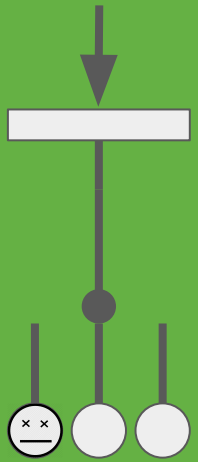
High Availability

Error!
503: Service Unavailable



High Availability

Error!
503: Service Unavailable



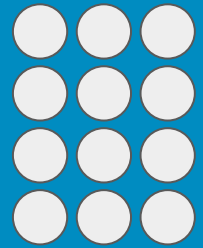
neo4j
3.1
Causal
Clustering

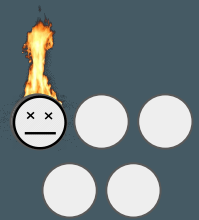
Bigger Clusters

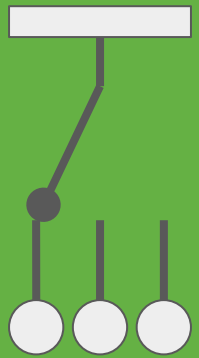
Consensus Commit

Built-in load balancing

neo4j
3.0



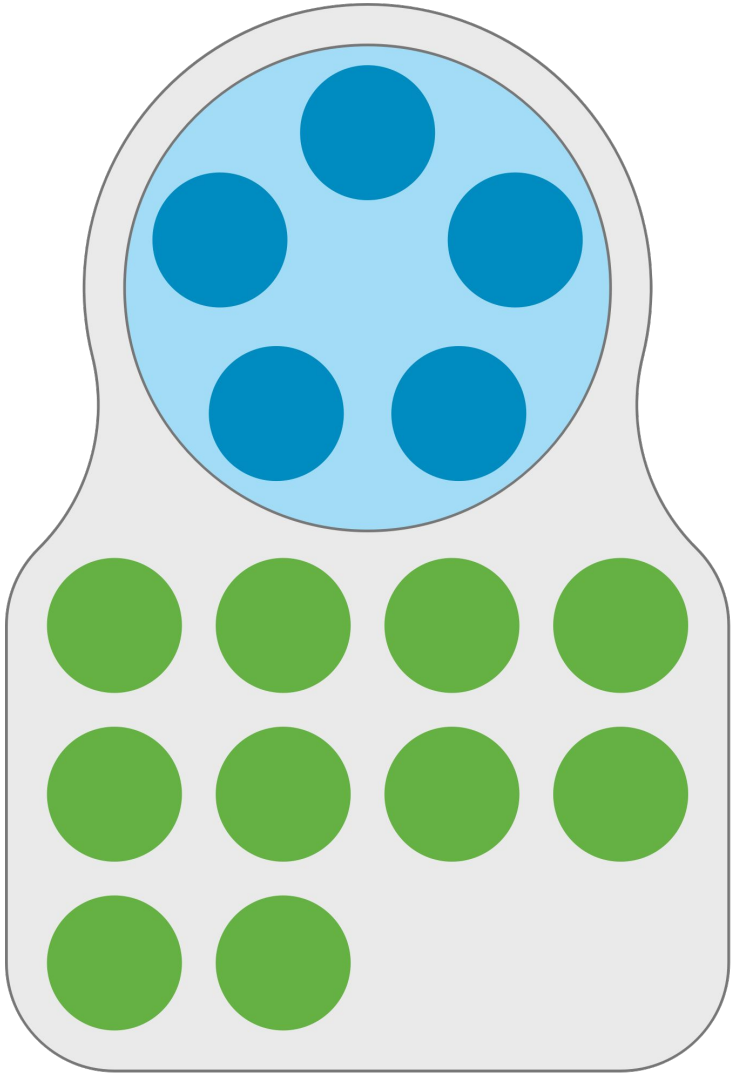




Massive Throughput

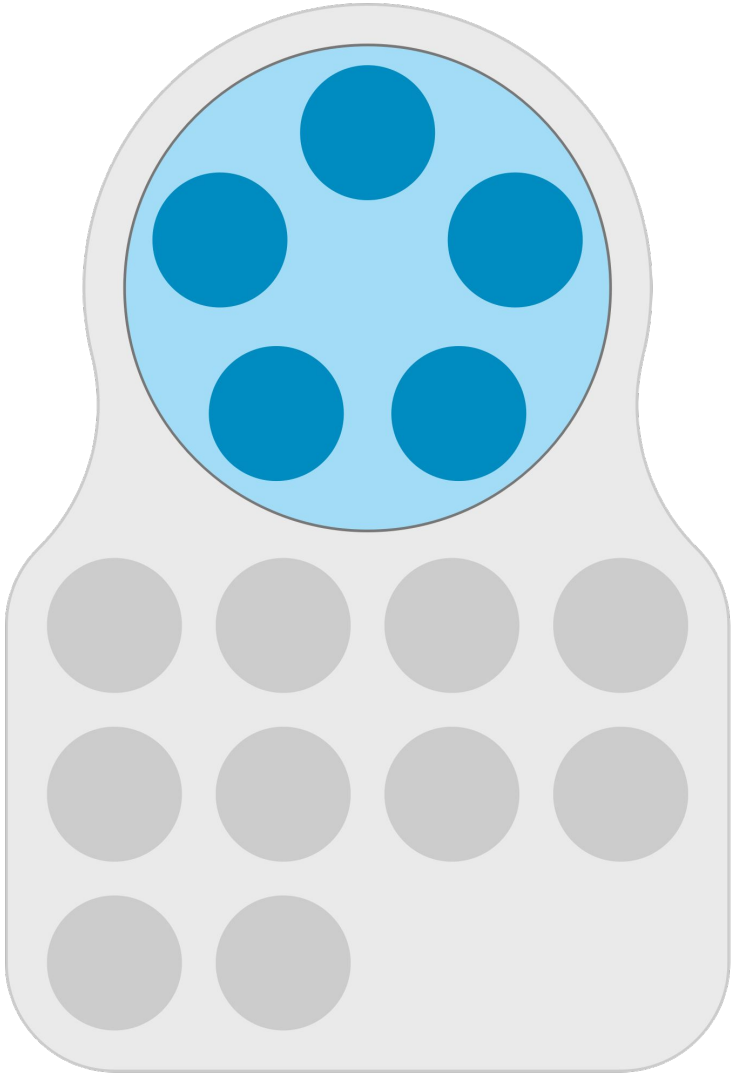
Data Redundancy

High Availability



Core

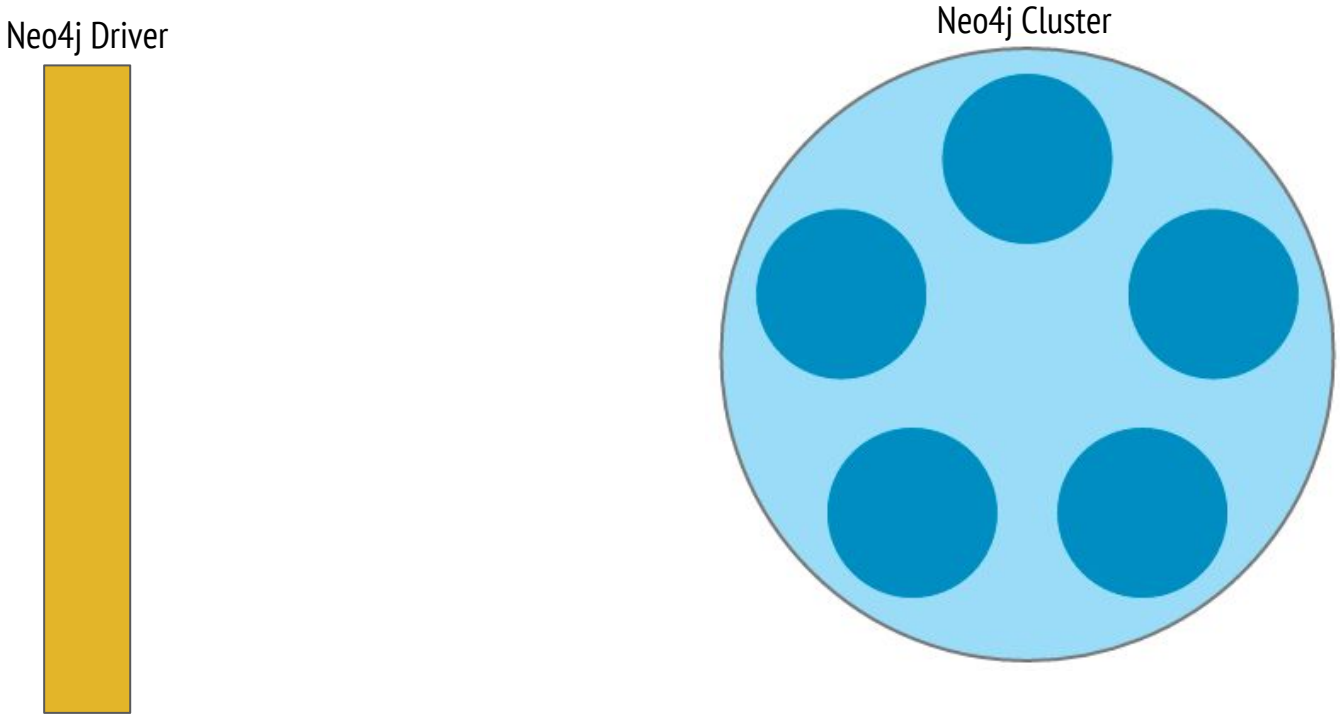
Read Replicas



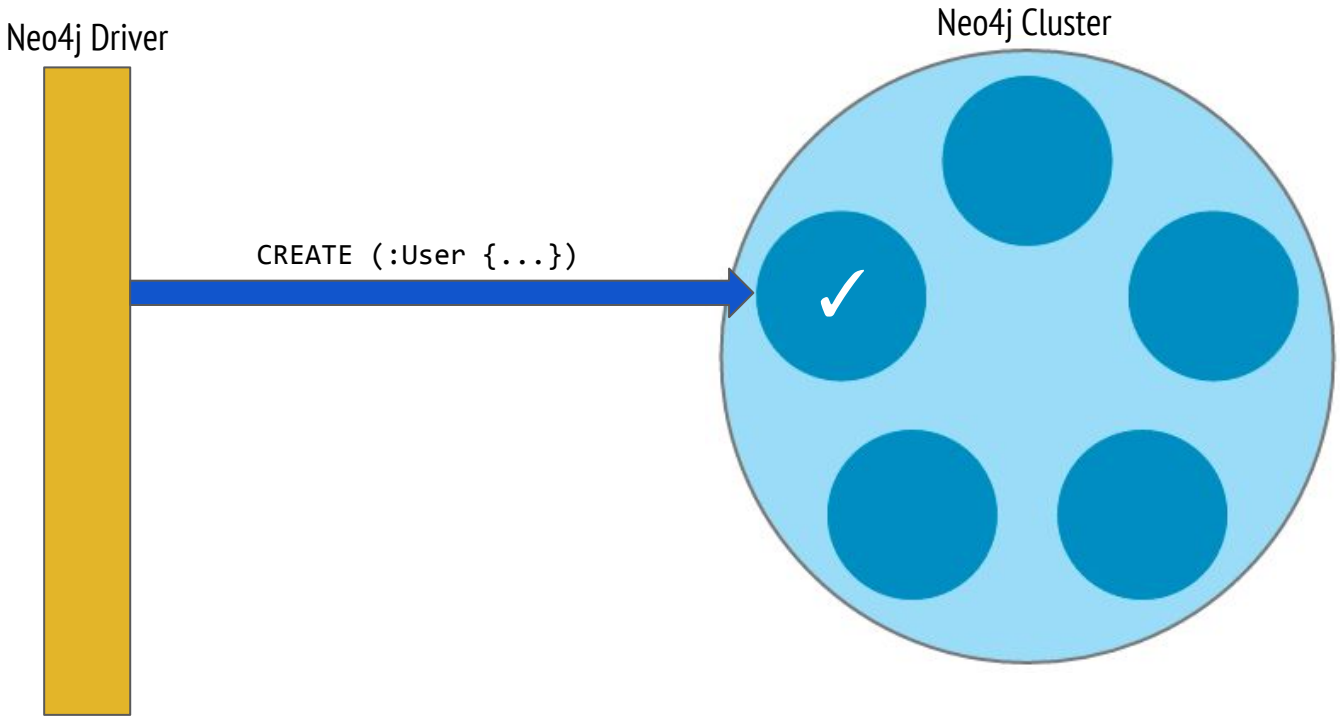
Core

- Small group of Neo4j databases
- Fault-tolerant Consensus Commit
- Responsible for data safety

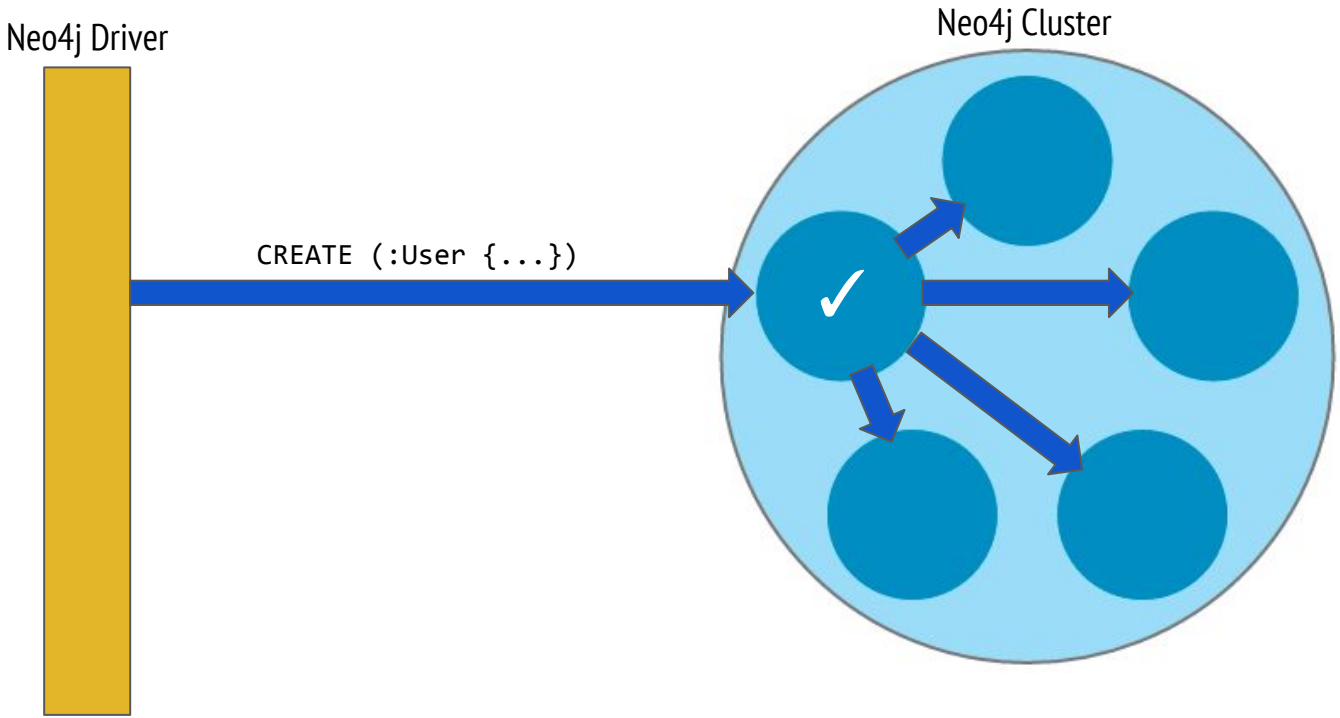
Writing to the Core Cluster



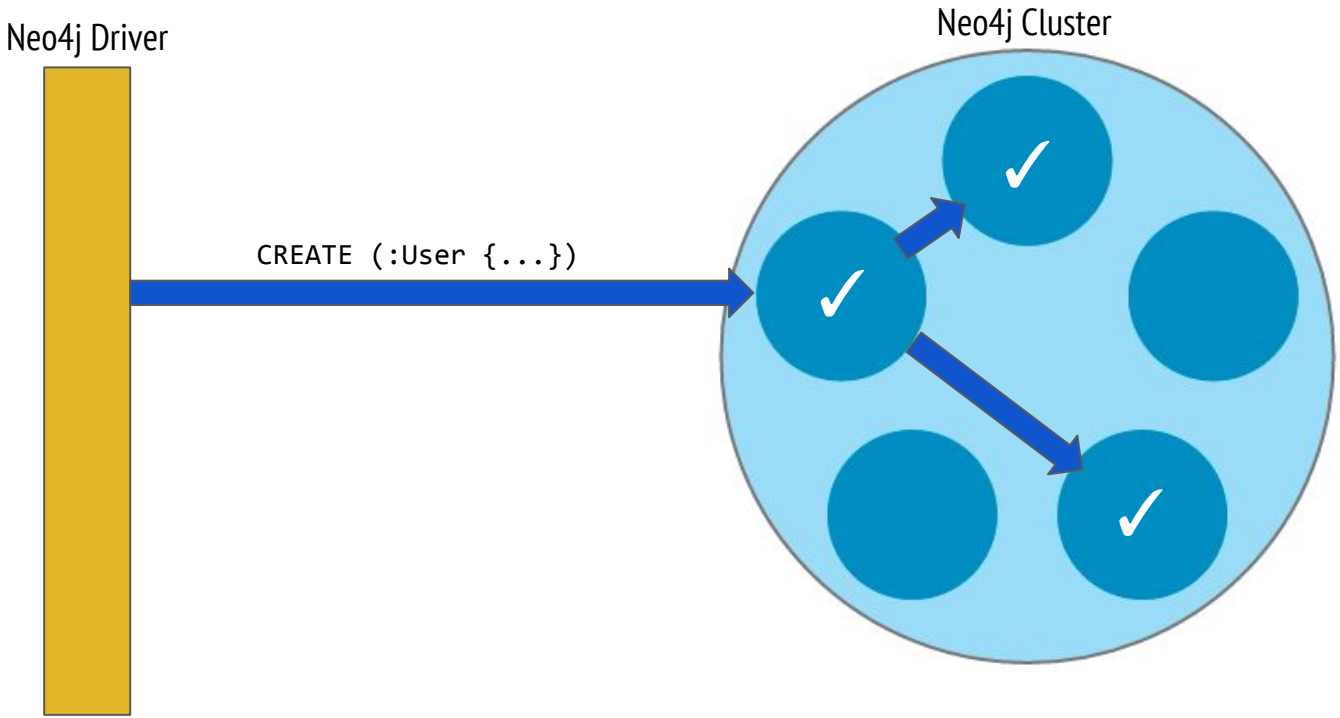
Writing to the Core Cluster



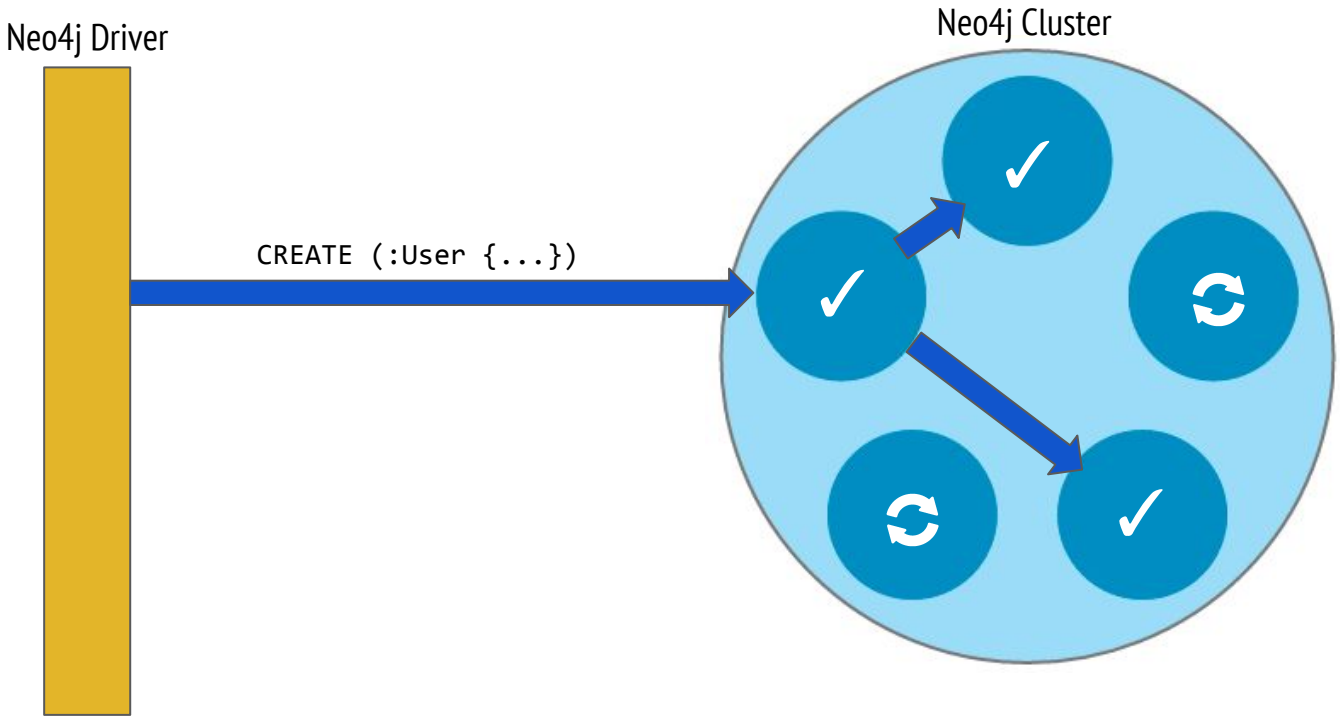
Writing to the Core Cluster



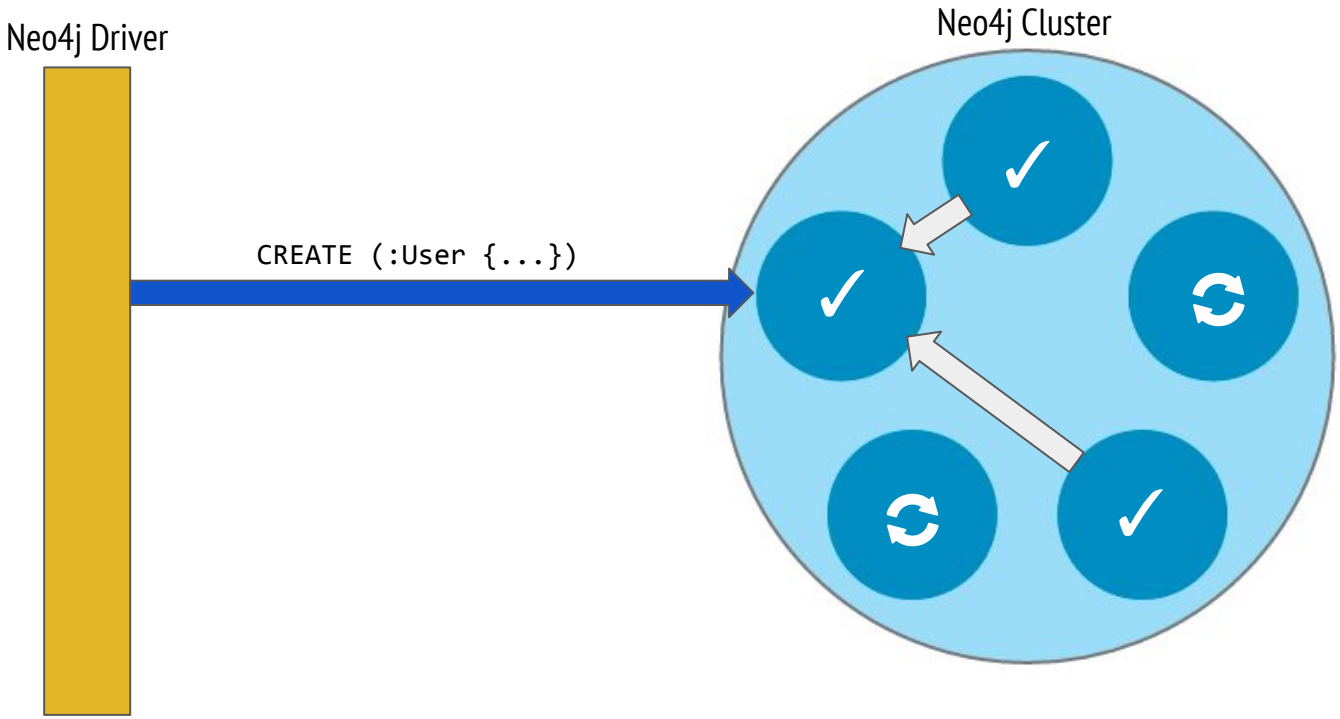
Writing to the Core Cluster



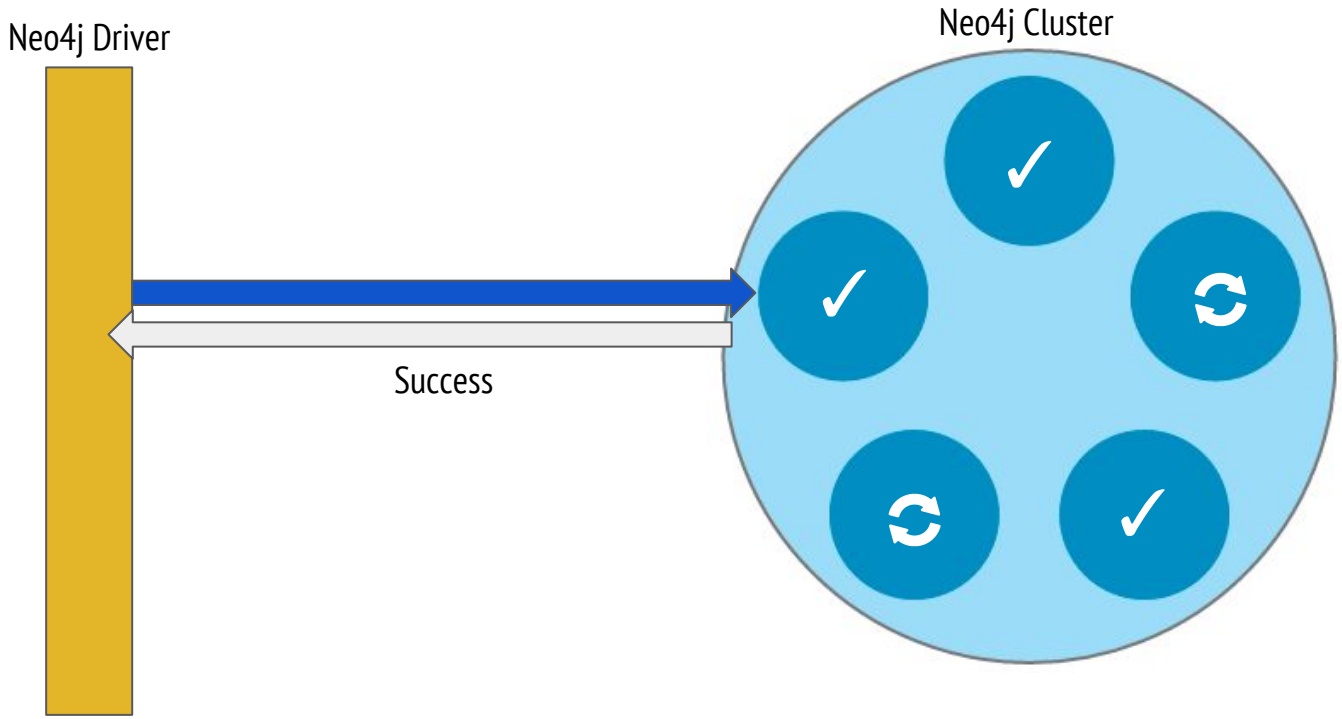
Writing to the Core Cluster



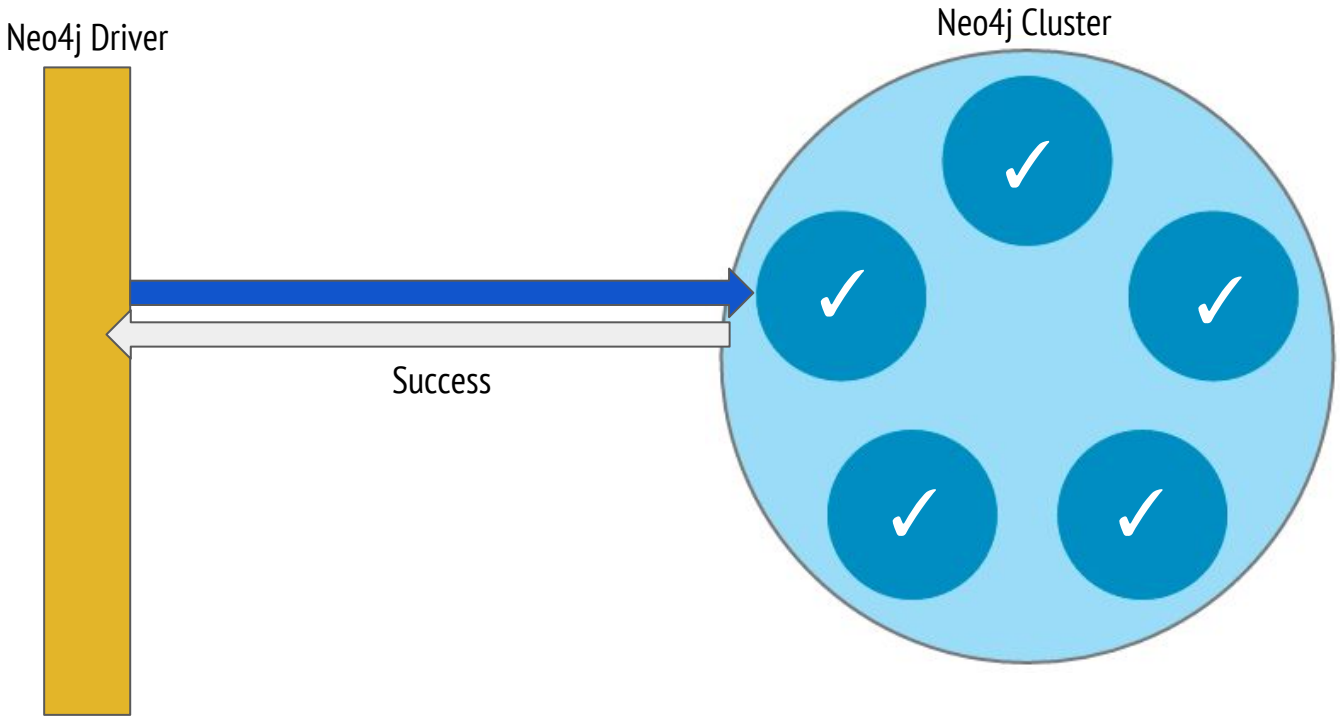
Writing to the Core Cluster



Writing to the Core Cluster



Writing to the Core Cluster



Raft Protocol

In Search of an Understandable Consensus Algorithm (Extended Version)

Diego Ongaro and John Ousterhout
Stanford University

Abstract

Raft is a consensus algorithm for managing a replicated log. It produces a result equivalent to (multi-)Paxos, and it is as efficient as Paxos, but its structure is different from Paxos; this makes Raft more understandable than Paxos and also provides a better foundation for building practical systems. In order to enhance understandability, Raft separates the key elements of consensus, such as leader election, log replication, and safety, and it enforces a stronger degree of coherency to reduce the number of states that must be considered. Results from a user study demonstrate that Raft is easier for students to learn than Paxos. Raft also includes a new mechanism for changing the cluster membership, which uses overlapping majorities to guarantee safety.

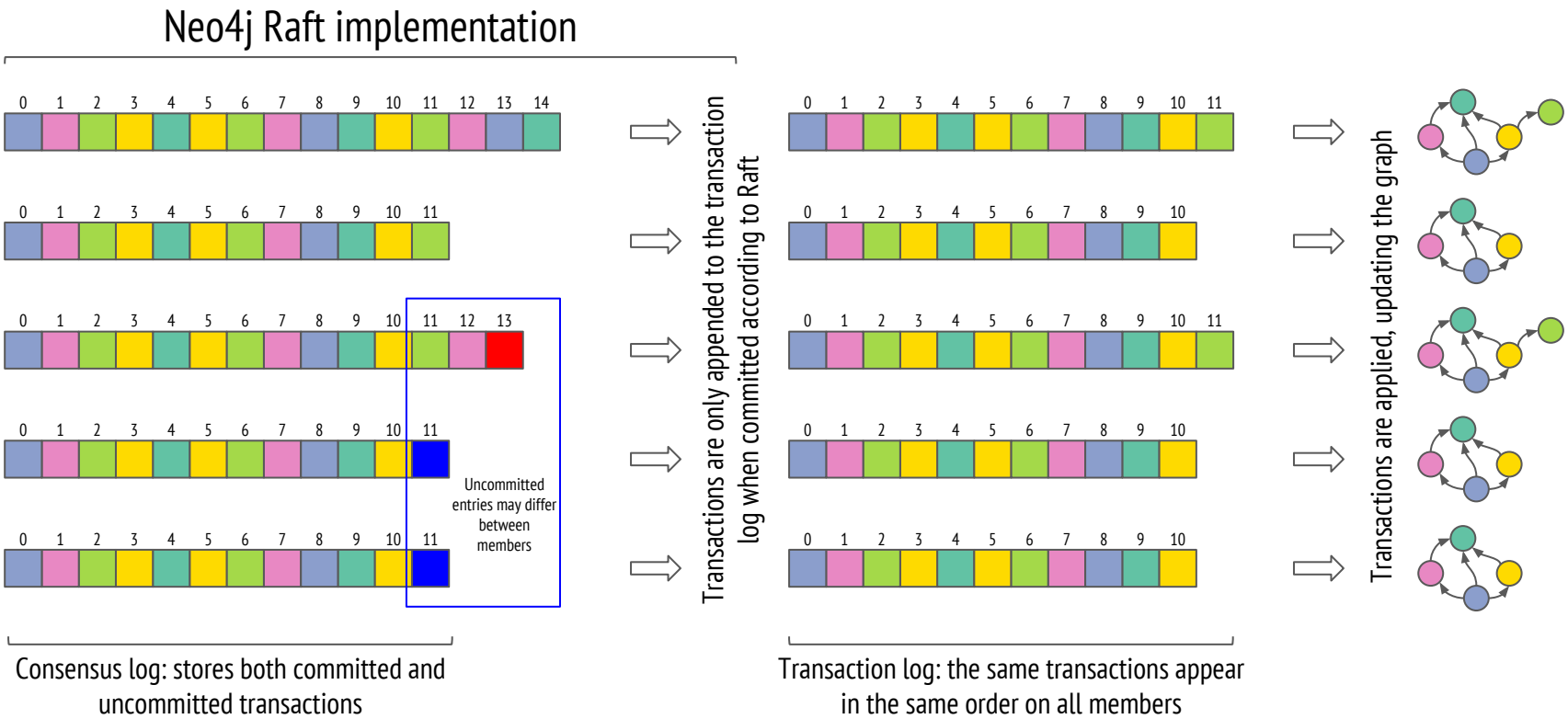
1 Introduction

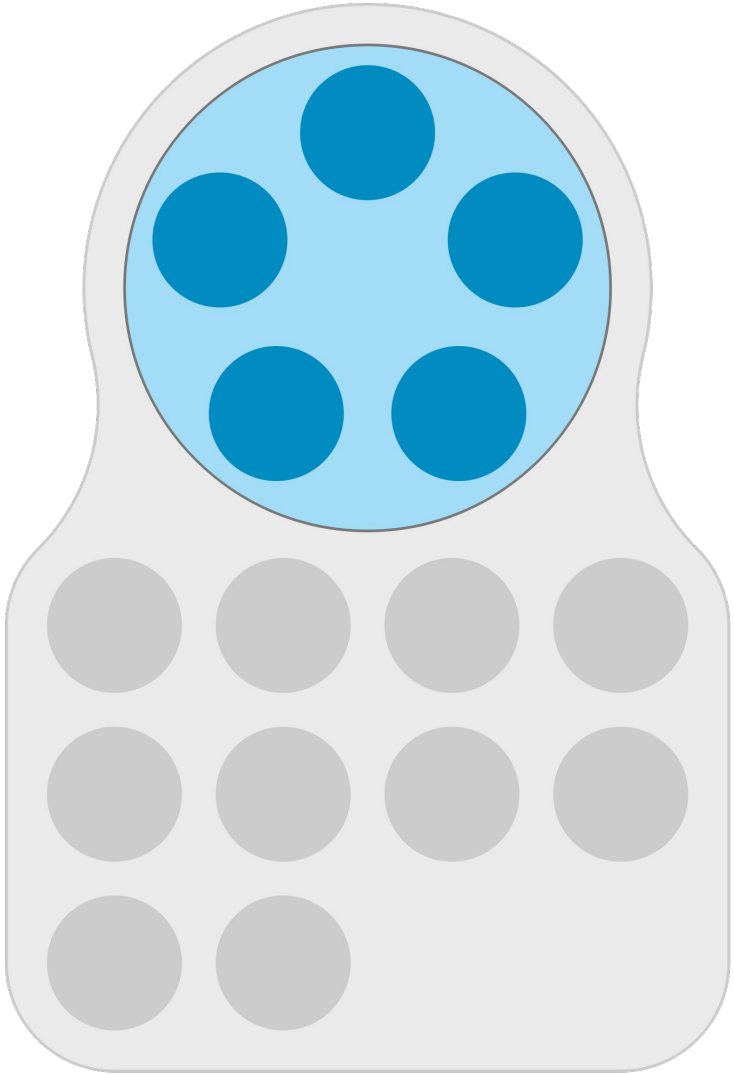
state space reduction (relative to Paxos, Raft reduces the degree of nondeterminism and the ways servers can be inconsistent with each other). A user study with 43 students at two universities shows that Raft is significantly easier to understand than Paxos: after learning both algorithms, 33 of these students were able to answer questions about Raft better than questions about Paxos.

Raft is similar in many ways to existing consensus algorithms (most notably, Oki and Liskov's Viewstamped Replication [29, 22]), but it has several novel features:

- **Strong leader:** Raft uses a stronger form of leadership than other consensus algorithms. For example, log entries only flow from the leader to other servers. This simplifies the management of the replicated log and makes Raft easier to understand.
- **Leader election:** Raft uses randomized timers to

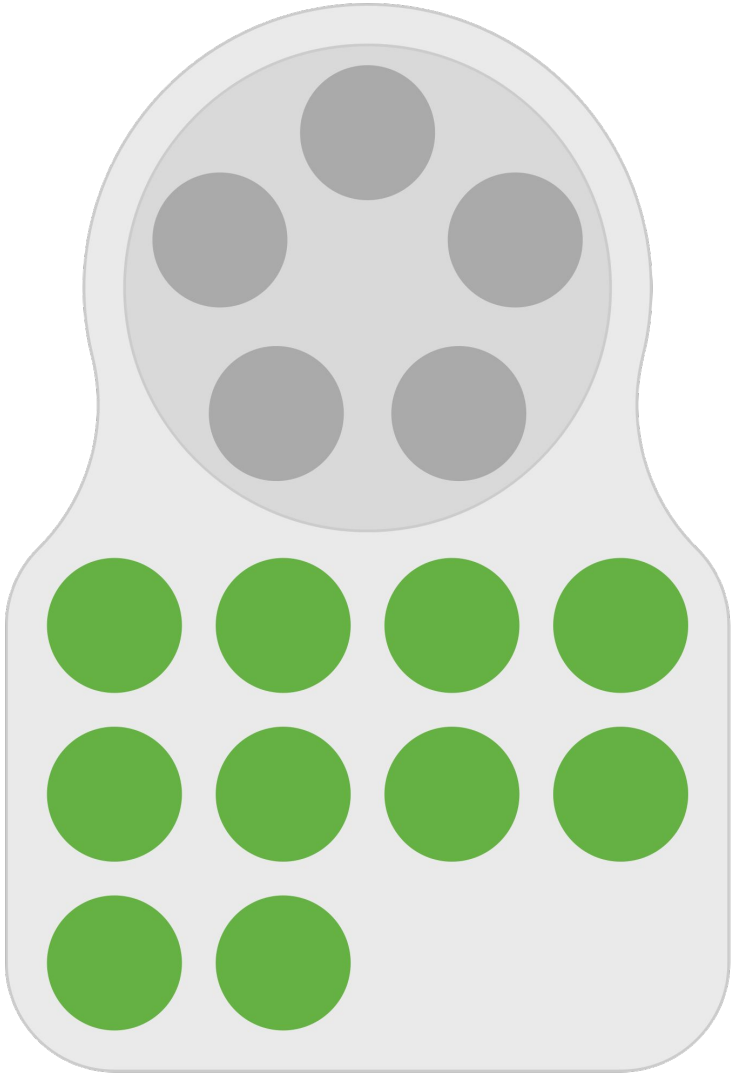
Consensus Log ⟶ Committed Transactions ⟶ Updated Graph





Core

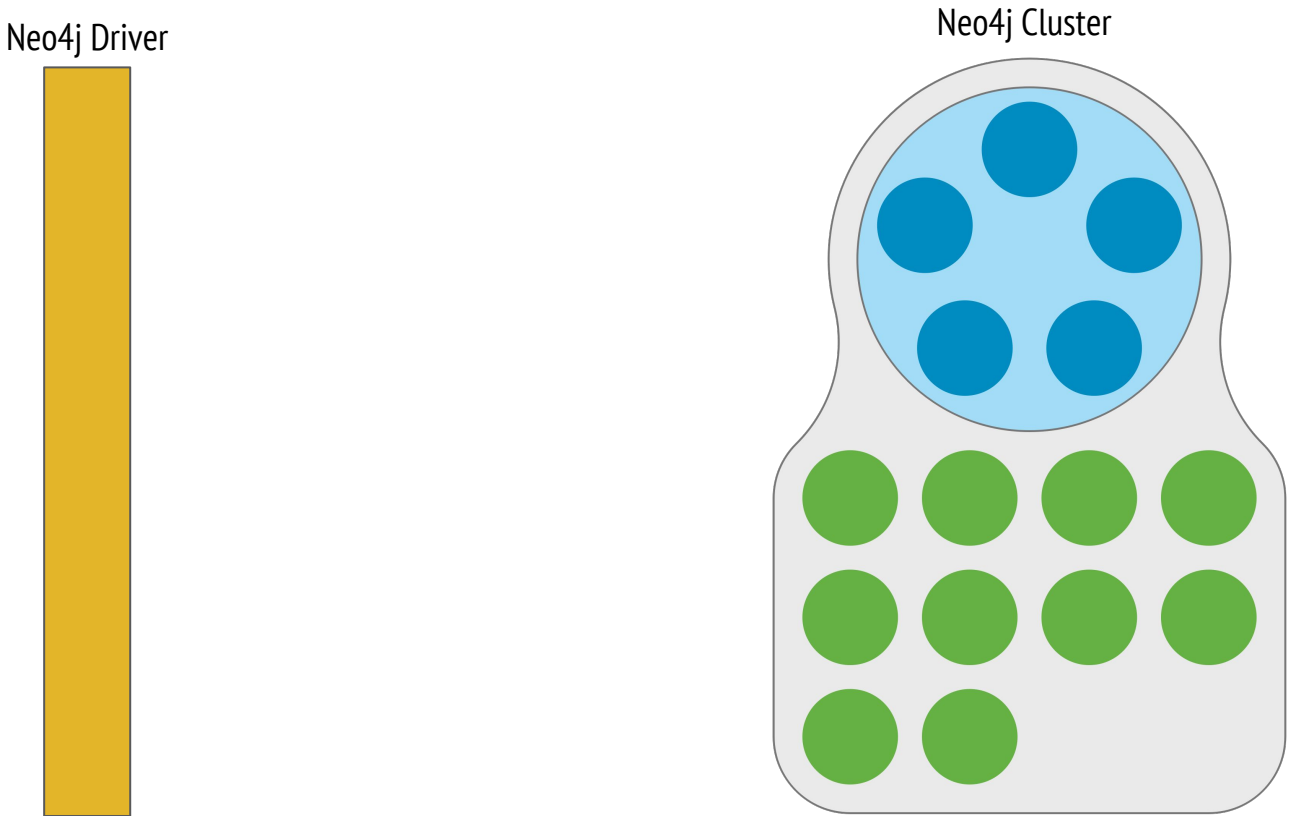
- Small group of Neo4j databases
- Fault-tolerant Consensus Commit
- Responsible for data safety



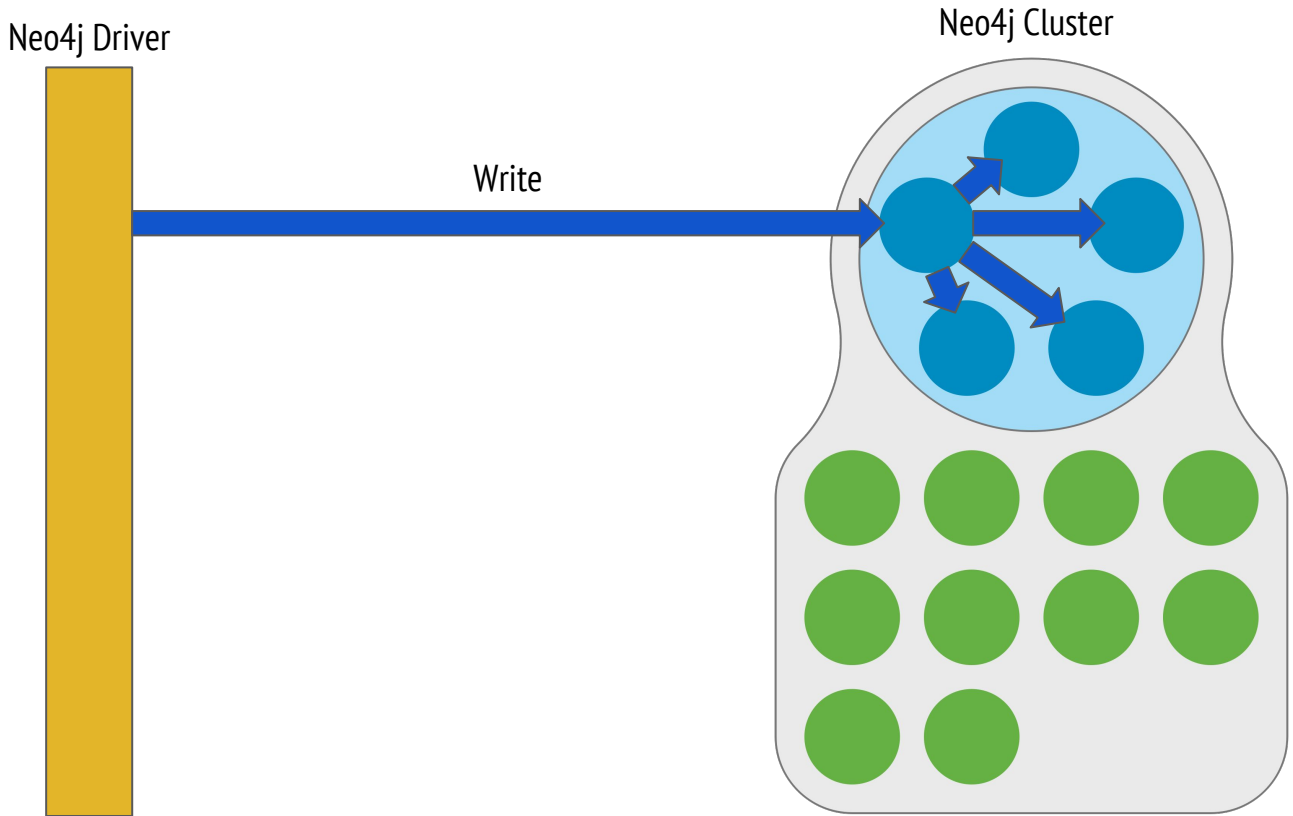
Read Replicas

- For massive query throughput
- Read-only
- Not involved in Consensus Commit
- Disposable, suitable for auto-scaling

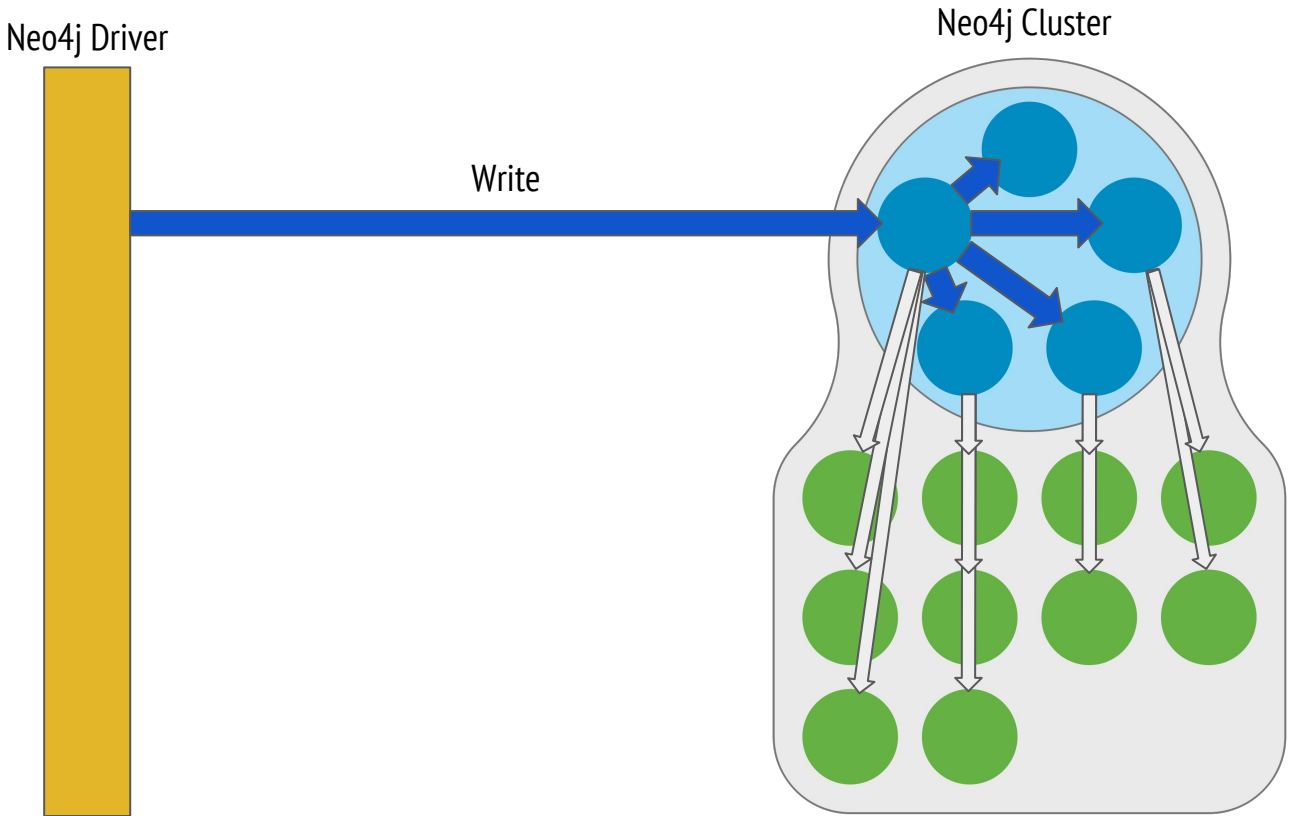
Propagating updates to the Read Replicas



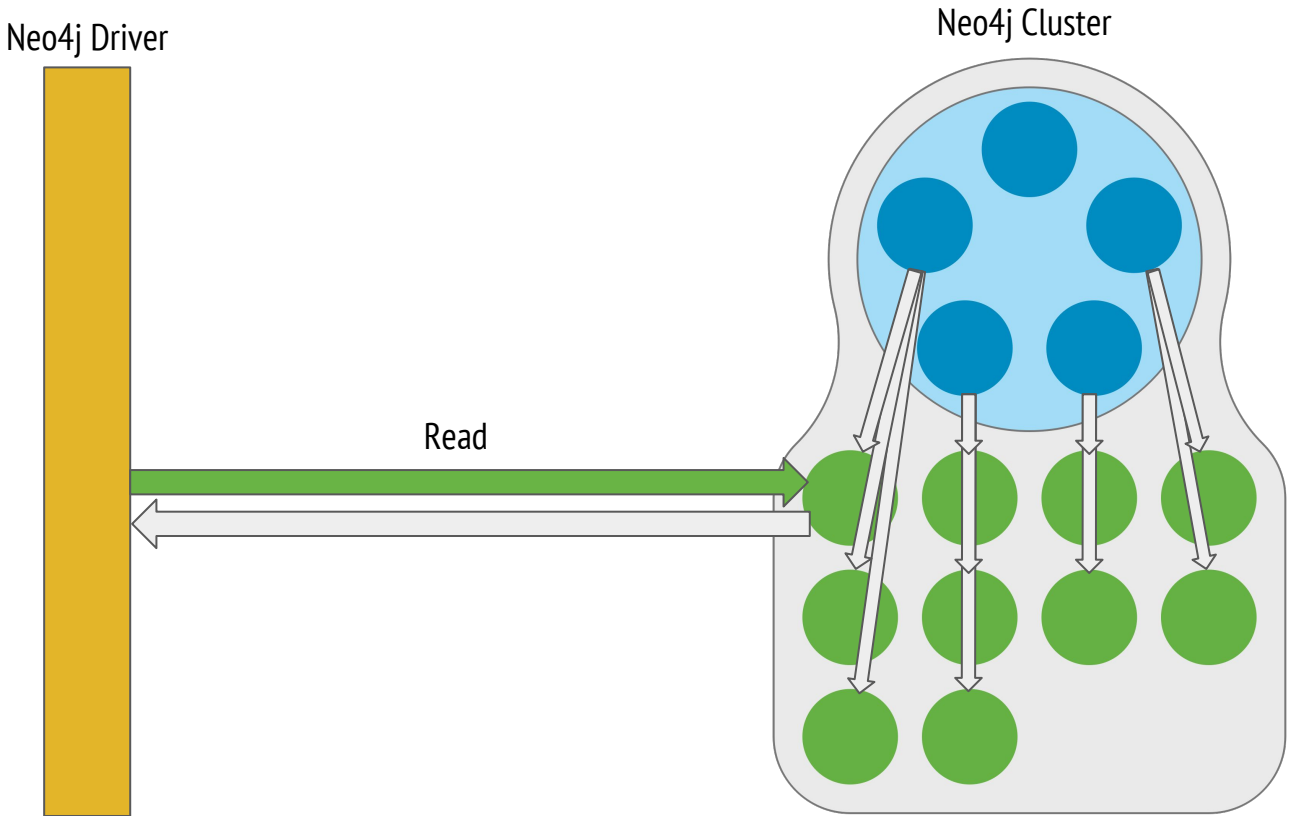
Propagating updates to the Read Replicas

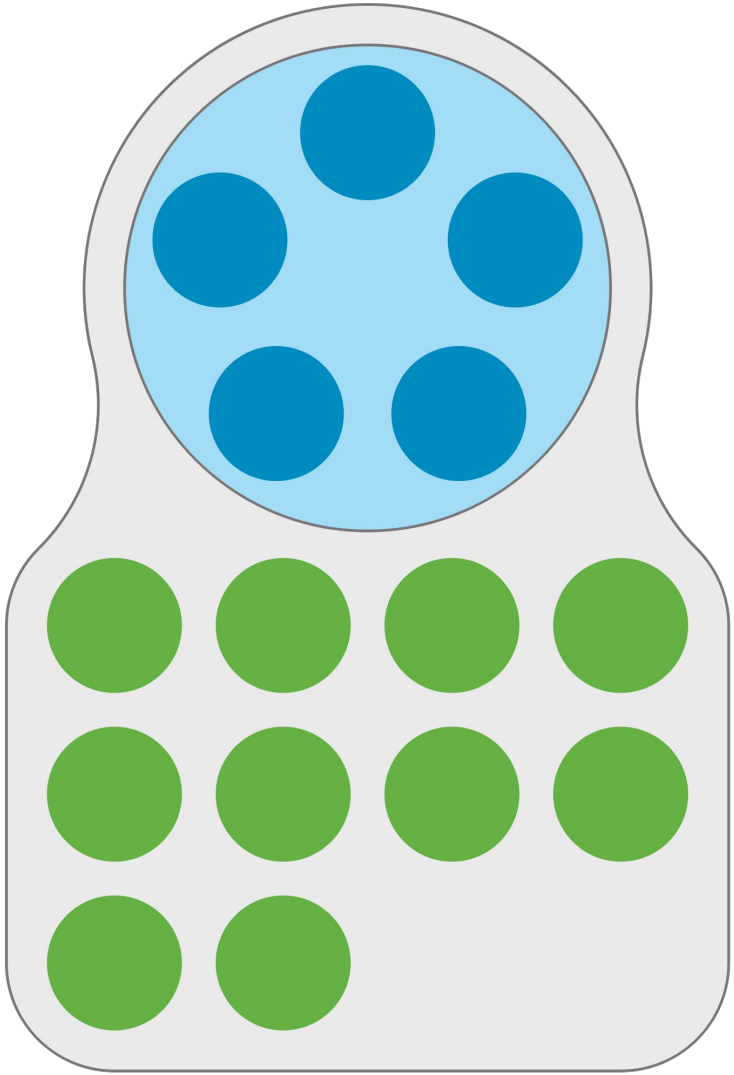


Propagating updates to the Read Replicas



Reading from the Edge





Core

Updating the graph

Read Replicas

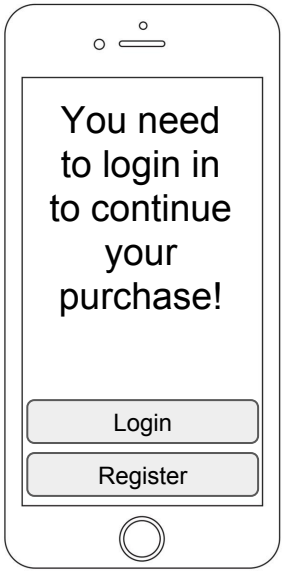
Queries, analysis, reporting

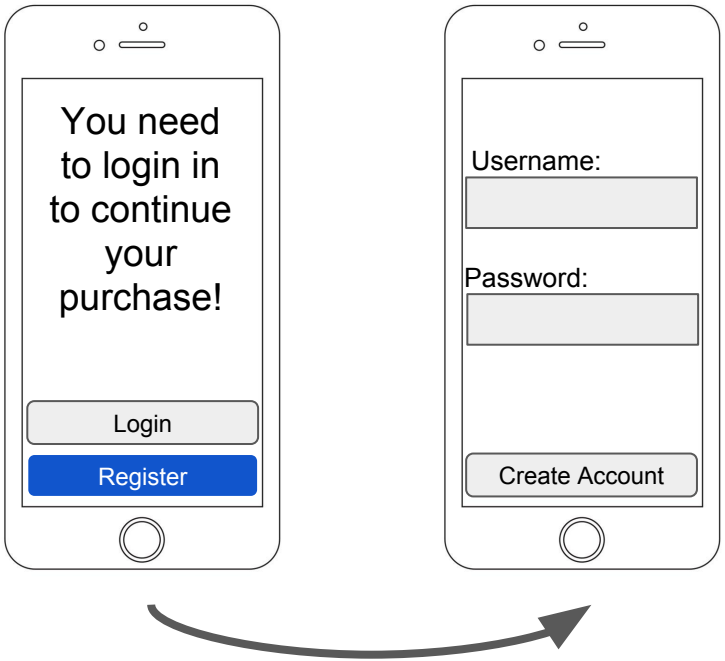
bolt+routing://

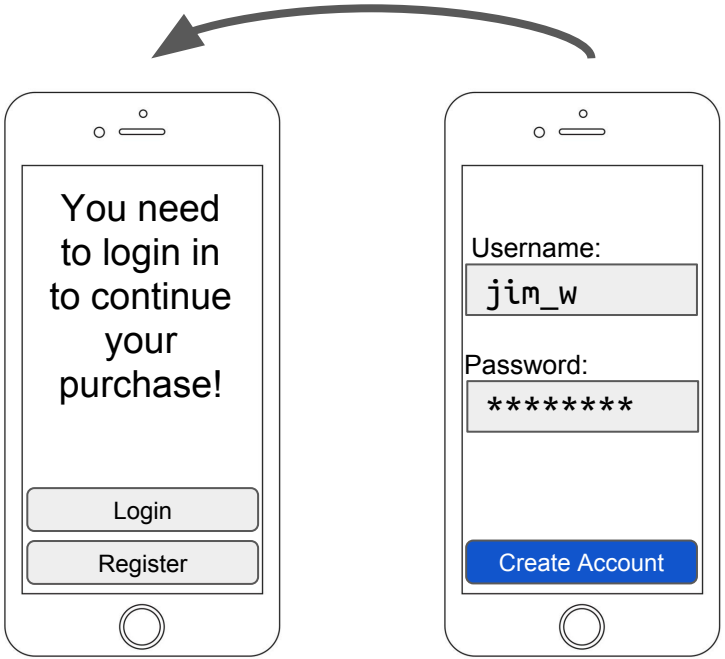
```
GraphDatabase.driver( "bolt+routing://aCoreServer" )
```

Bootstrap: specify any core server to route load across the whole cluster

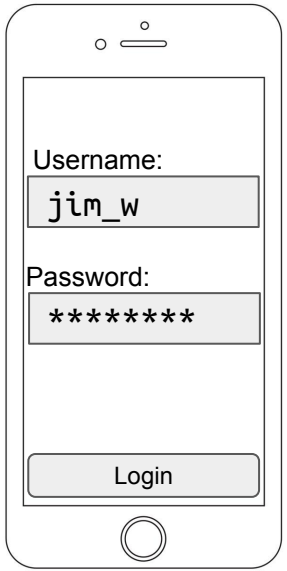
Consistency

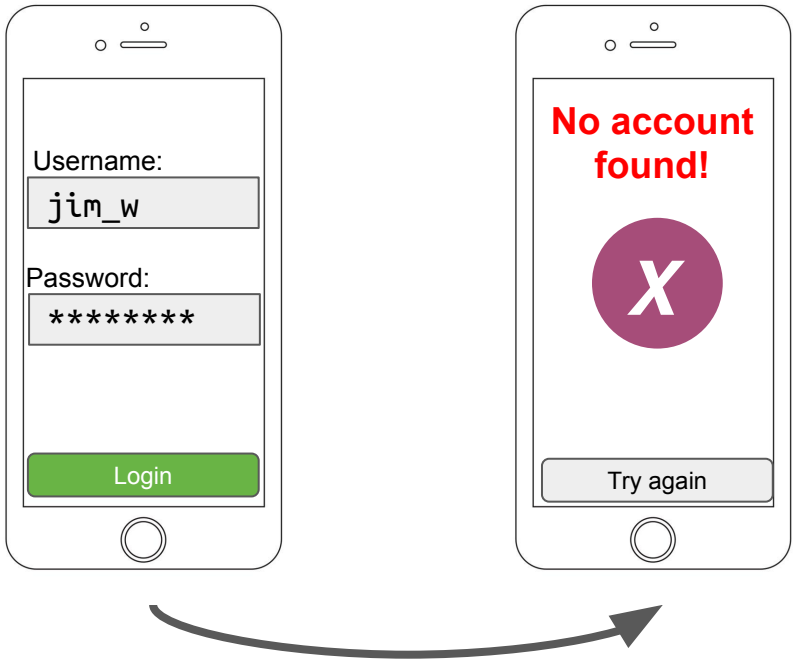

You need to login in to continue your purchase!
Login
Register


You need to login in to continue your purchase!
Login
Register
Username:
Password:
Create Account

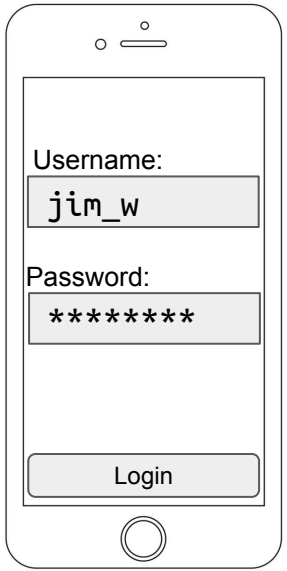

You need to login in to continue your purchase!
Login
Register
Username:
jim_w
Password:
****
Create Account

You need to login in to continue your purchase!
Login
Register
Username:
Password:
Login

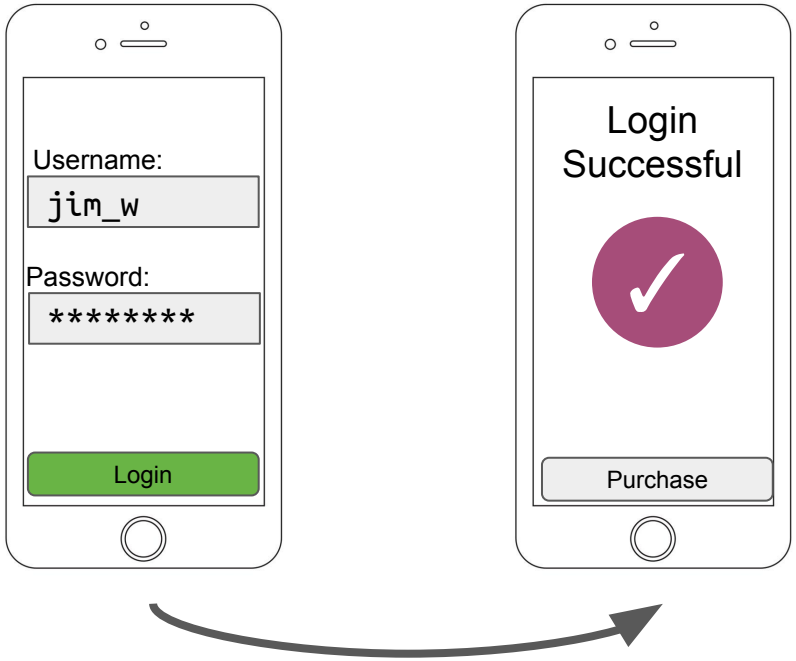

Username:
jim_w
Password:
****
Login


Username:
jim_w
Password:
****
Login
No account found!
X
Try again

A few moments later...

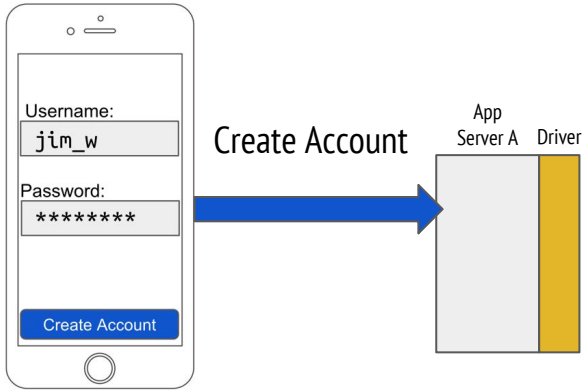


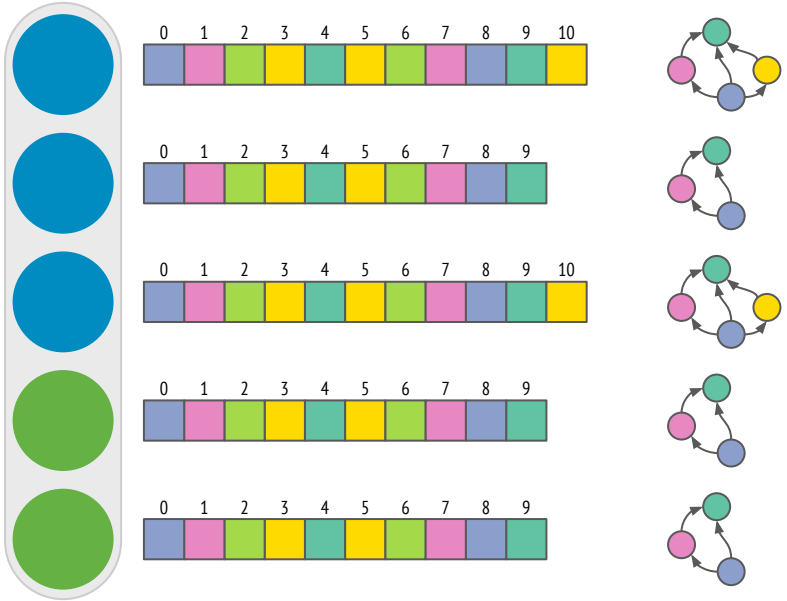
A few moments later...

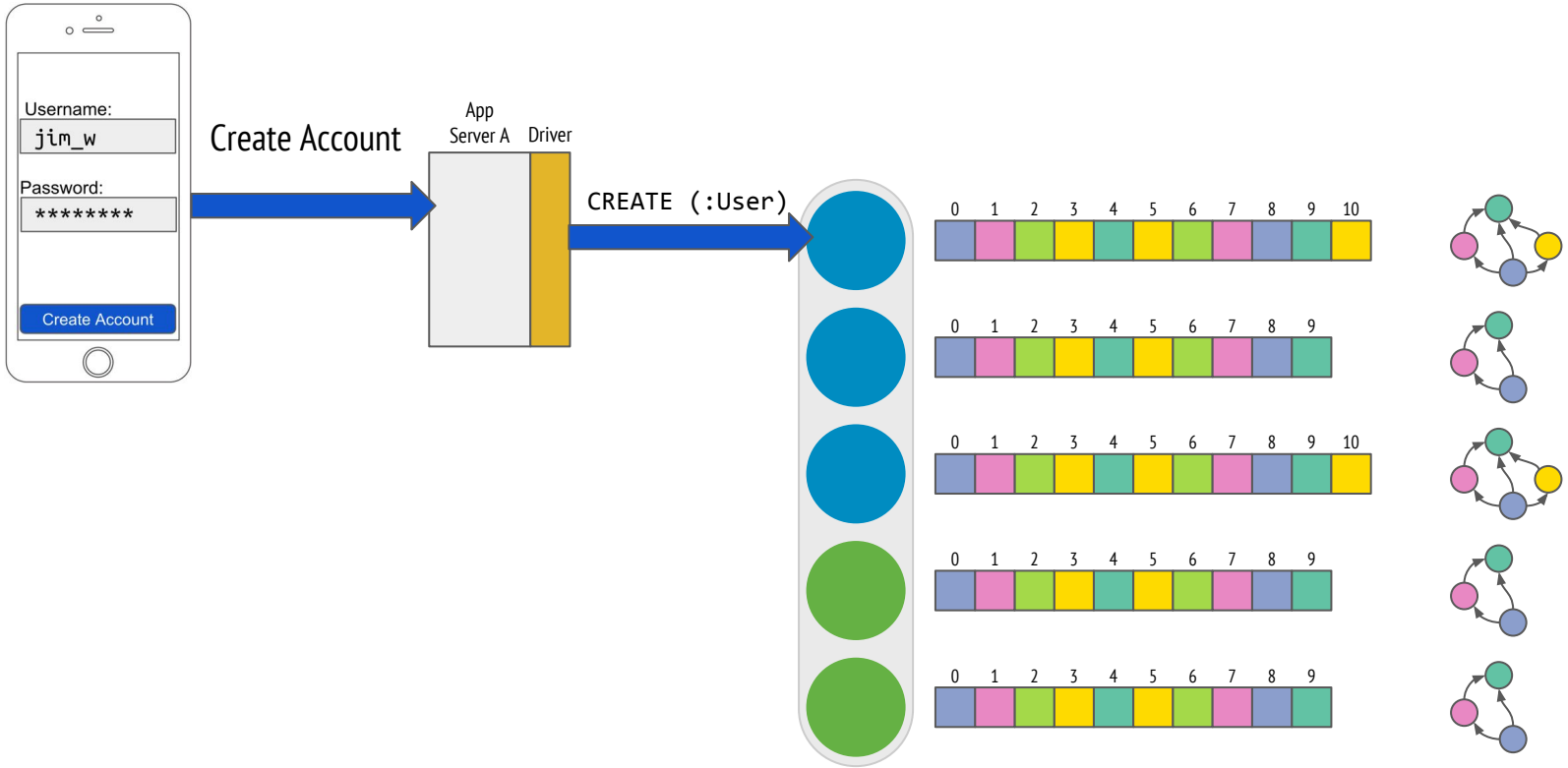


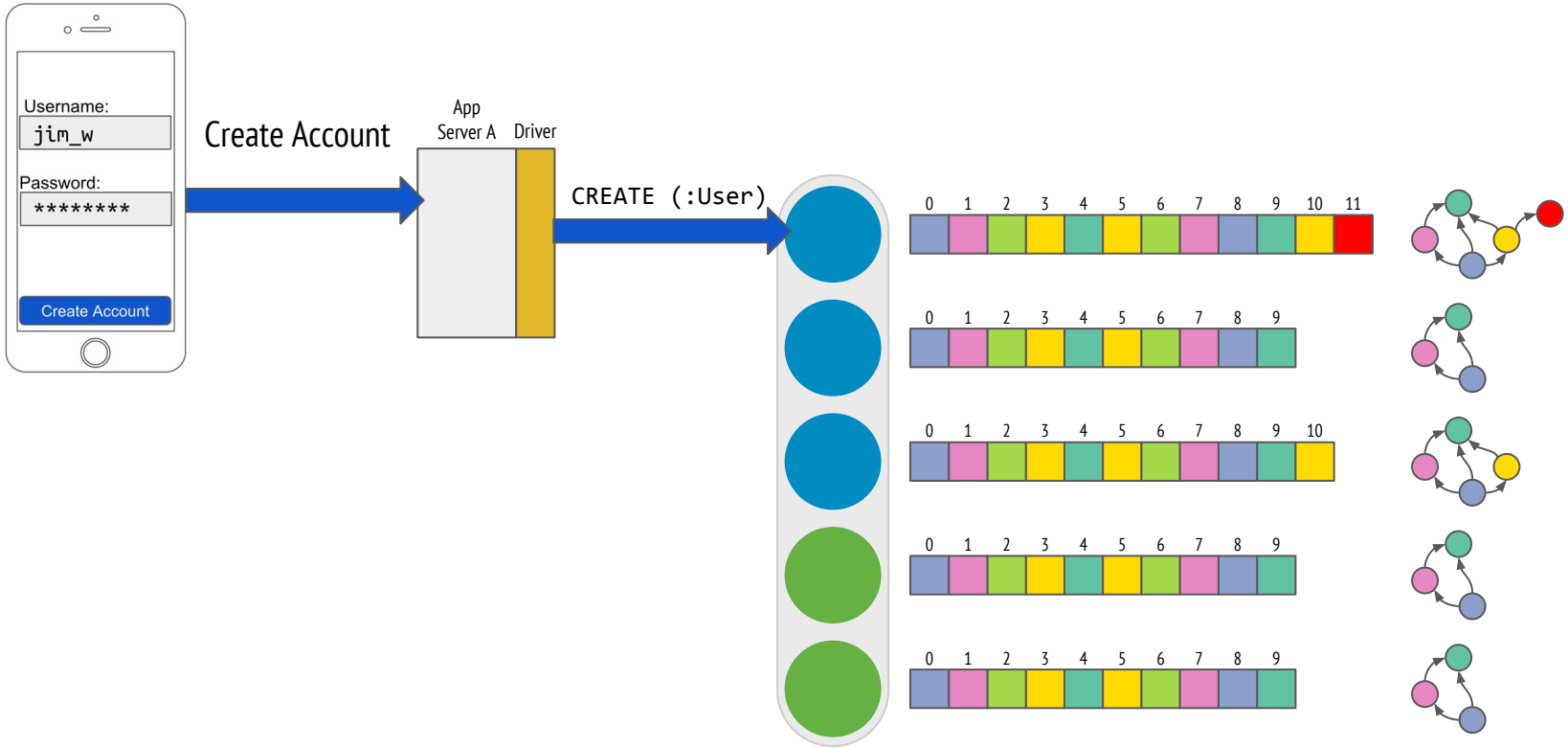
Q Why didn’t this work?

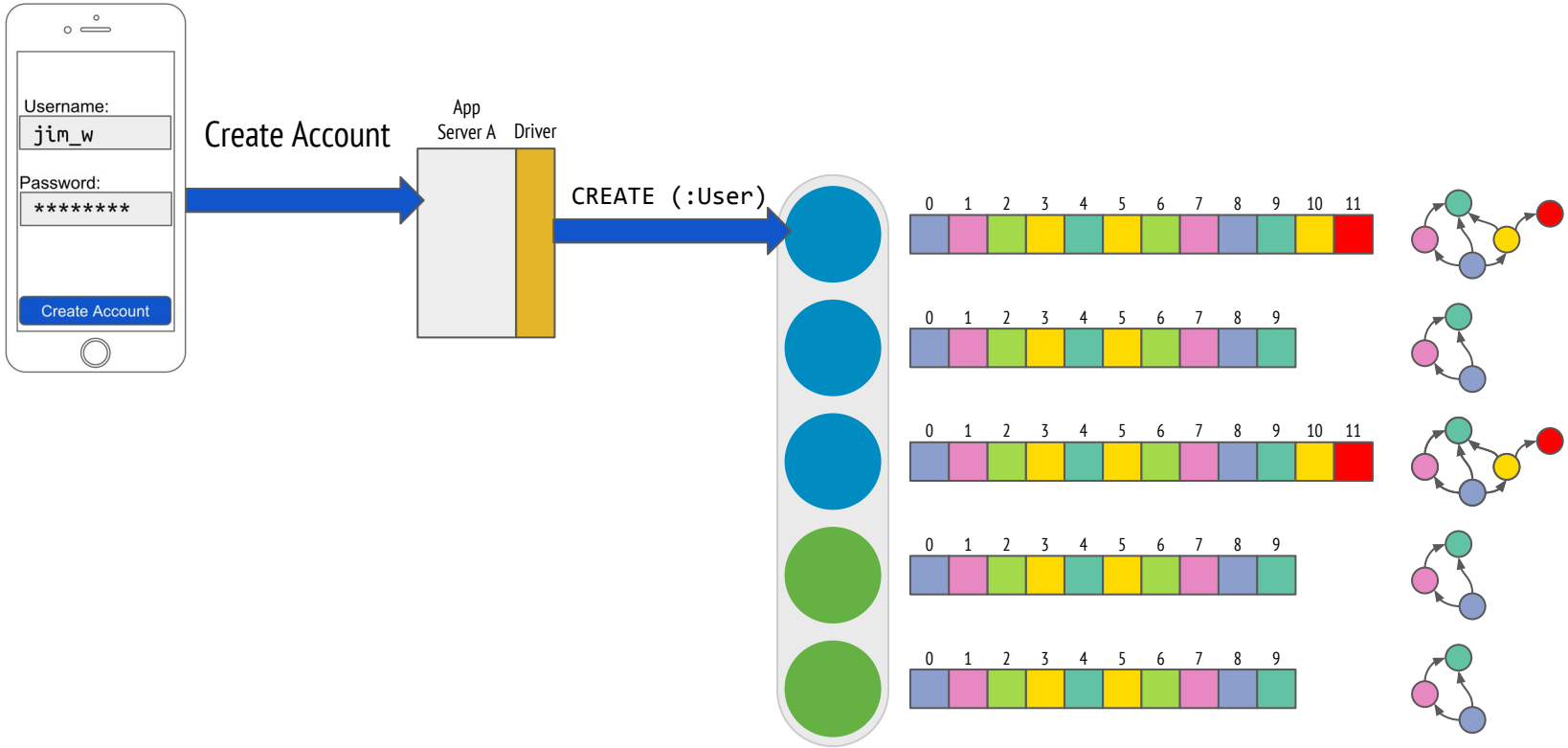
A Eventual Consistency

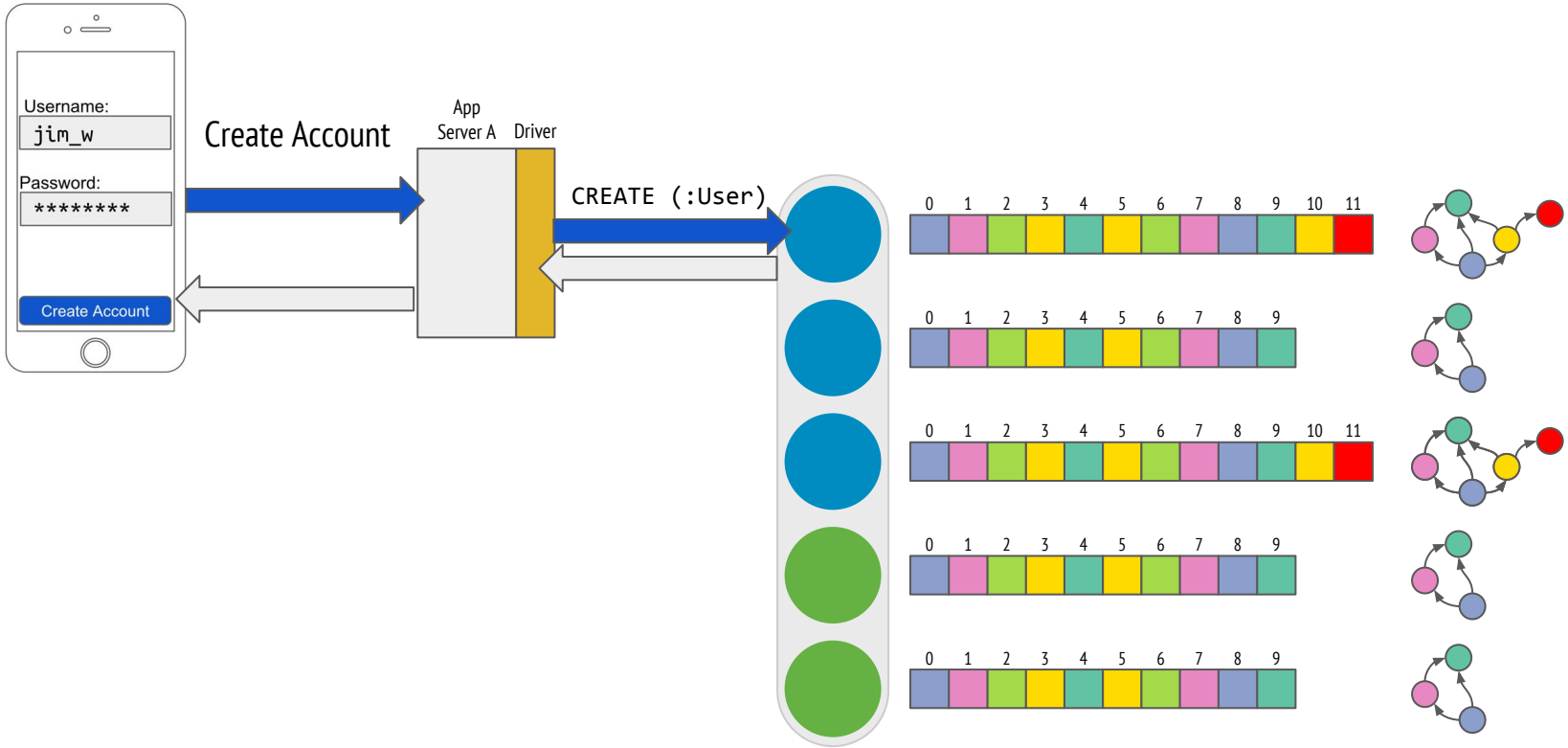

Username:
jim_w
Password:
****
Create Account
Create Account
App
Server A
Driver

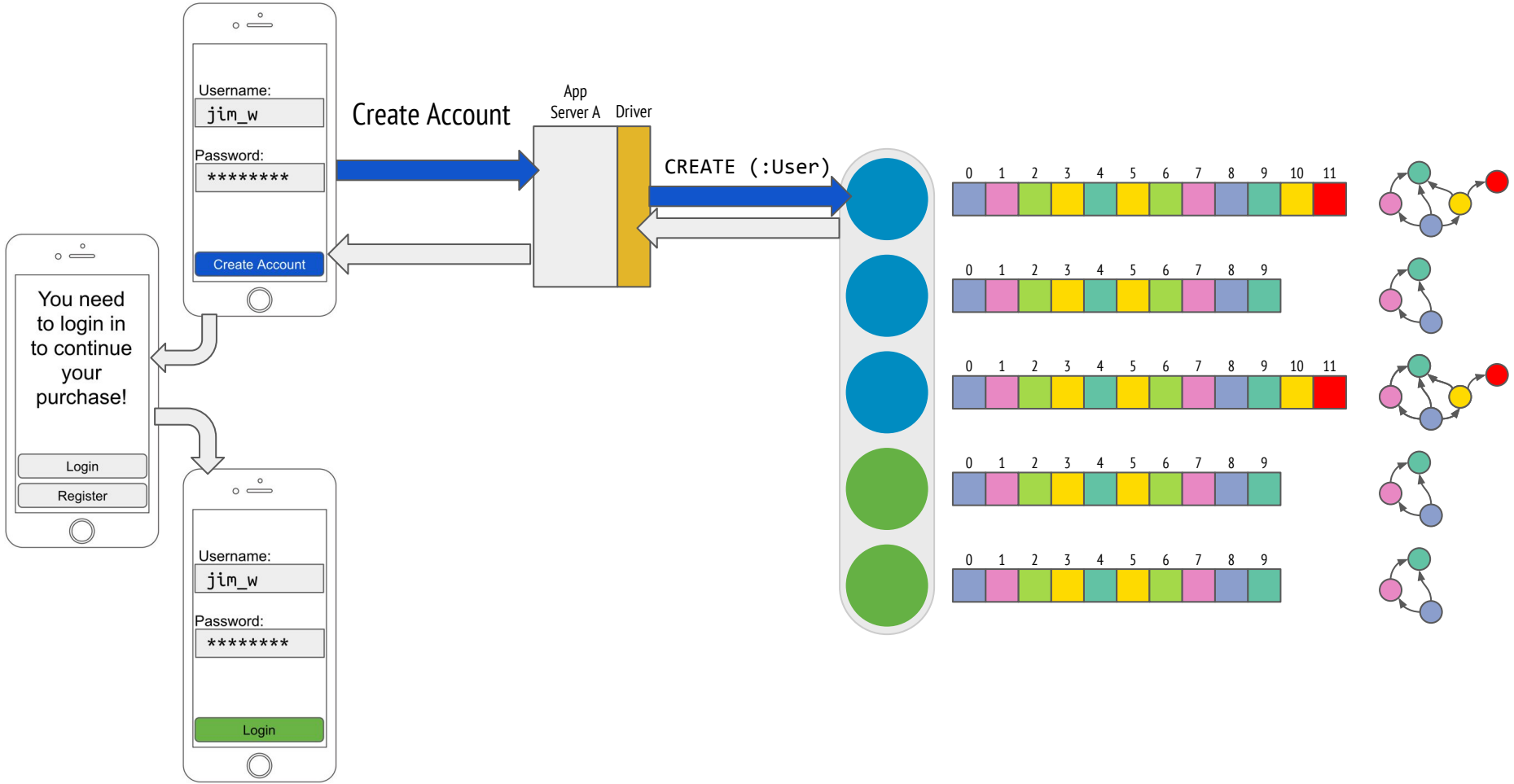

0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9

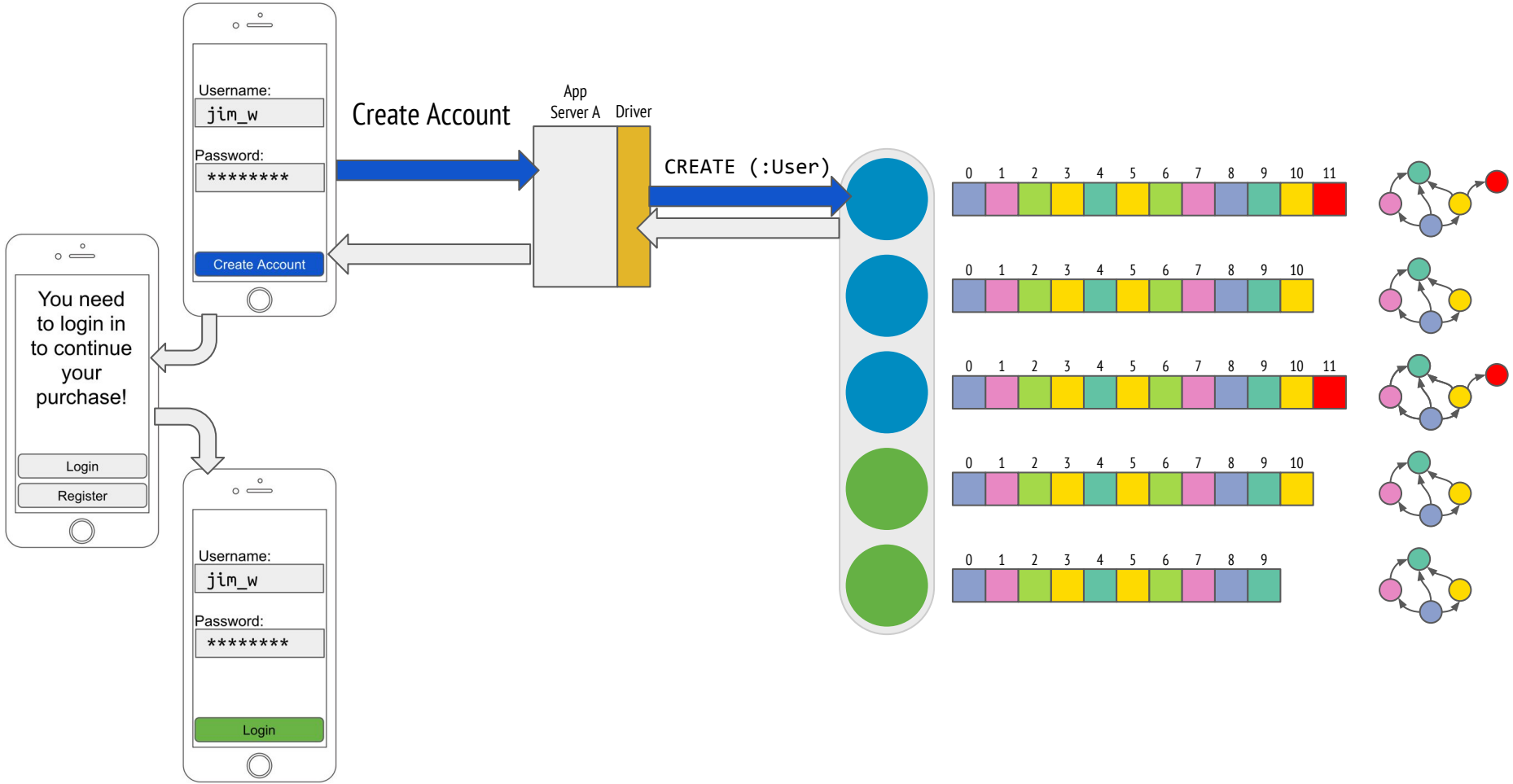

Username:
jim_w
Password:
****
Create Account
Create Account
App
Server A
Driver
CREATE (:User)
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9

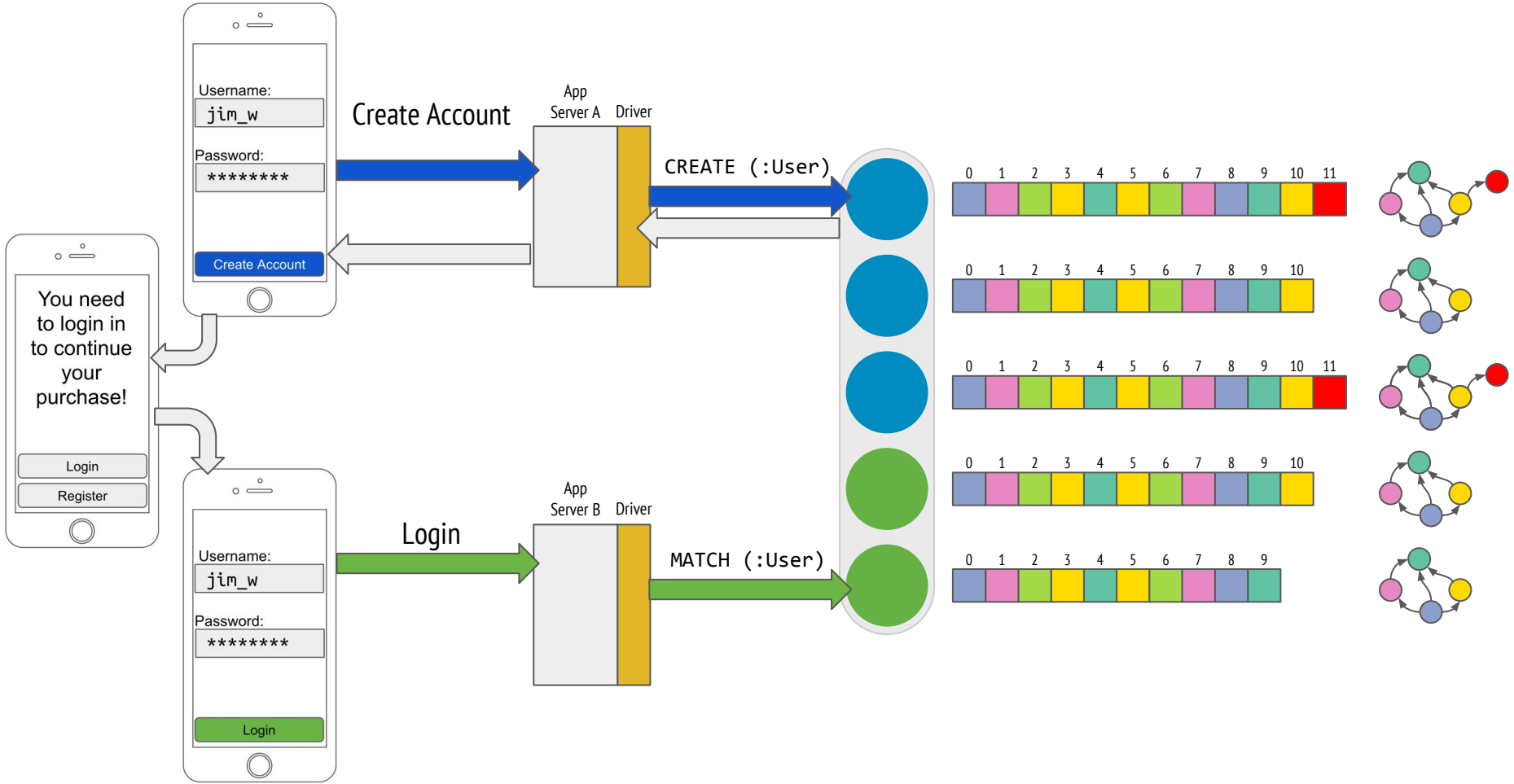

Username:
jim_w
Password:
****
Create Account
Create Account
App
Server A
Driver
CREATE (:User)
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9

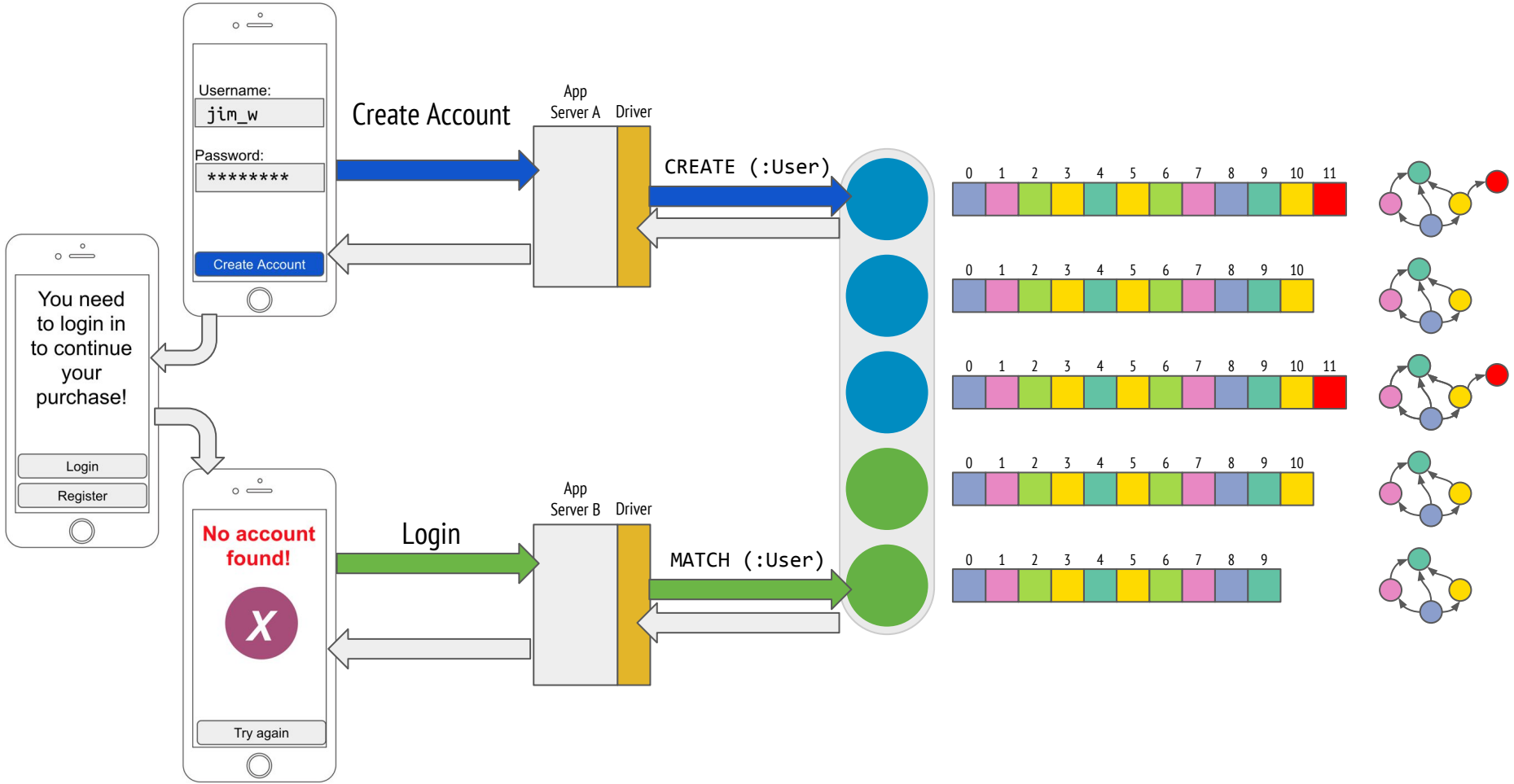

Username:
jim_w
Password:
****
Create Account
Create Account
App
Server A
Driver
CREATE (:User)
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9


Username:
jim_w
Password:
****
Create Account
Create Account
App
Server A
Driver
CREATE (:User)
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9


Username:
jim_w
Password:
****
Create Account
Create Account
App
Server A
Driver
CREATE (:User)
You need to login in to continue your purchase!
Login
Register
Username:
jim_w
Password:
****
Login
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9

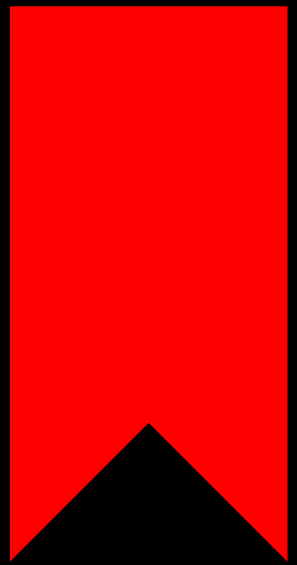

Username:
jim_w
Password:
****
Create Account
Create Account
App
Server A
Driver
CREATE (:User)
You need to login in to continue your purchase!
Login
Register
Username:
jim_w
Password:
****
Login
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9


Username:
jim_w
Password:
****
Create Account
Create Account
App Server A
Driver
CREATE (:User)
You need to login in to continue your purchase!
Login
Register
Username:
jim_w
Password:
****
Login
Login
App Server B
Driver
MATCH (:User)
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9

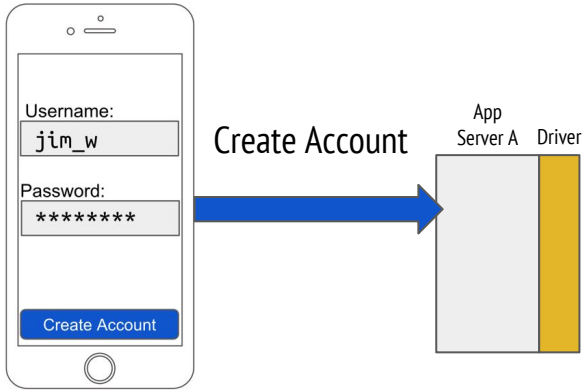

Username:
jim_w
Password:
****
Create Account
You need to login in to continue your purchase!
Login
Register
No account found!
X
Try again
Create Account
Login
App Server A
Driver
App Server B
Driver
CREATE (:User)
MATCH (:User)
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9

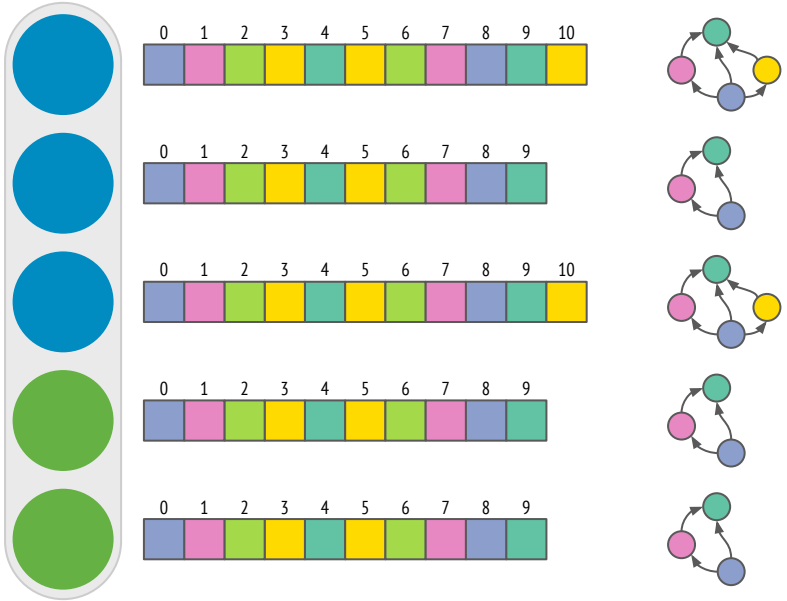
Let’s try again, with Causal Consistency

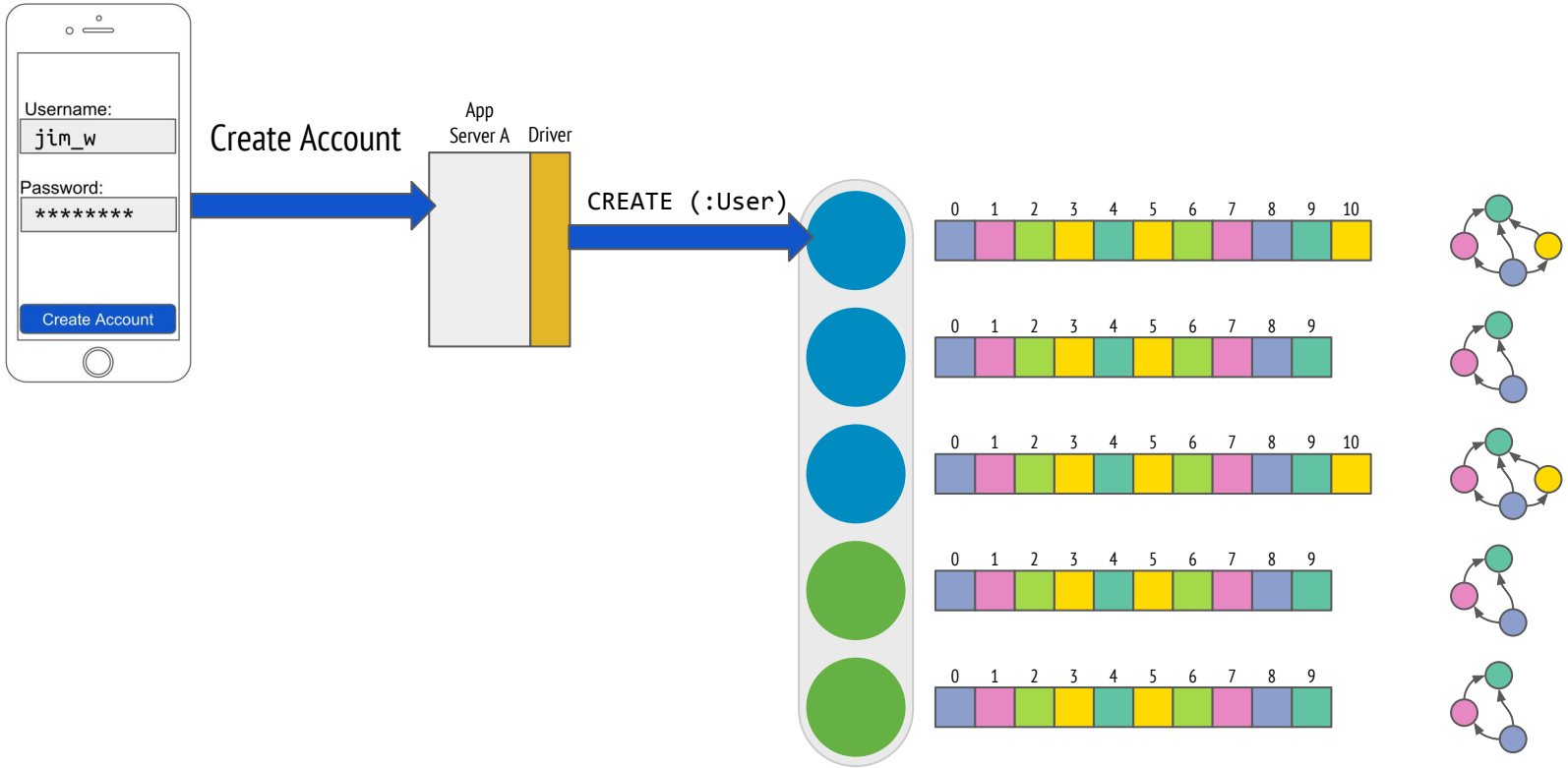
Bookmark

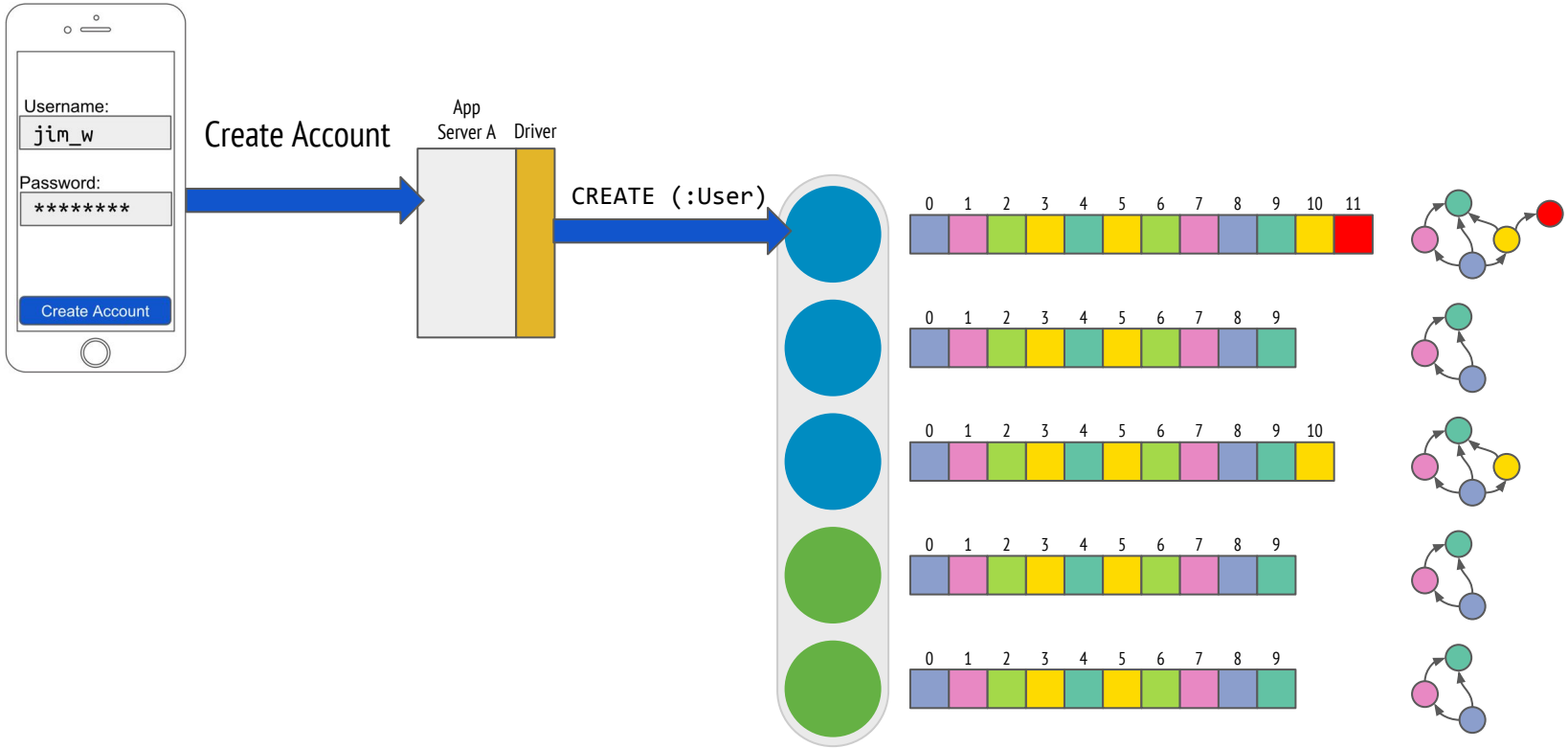


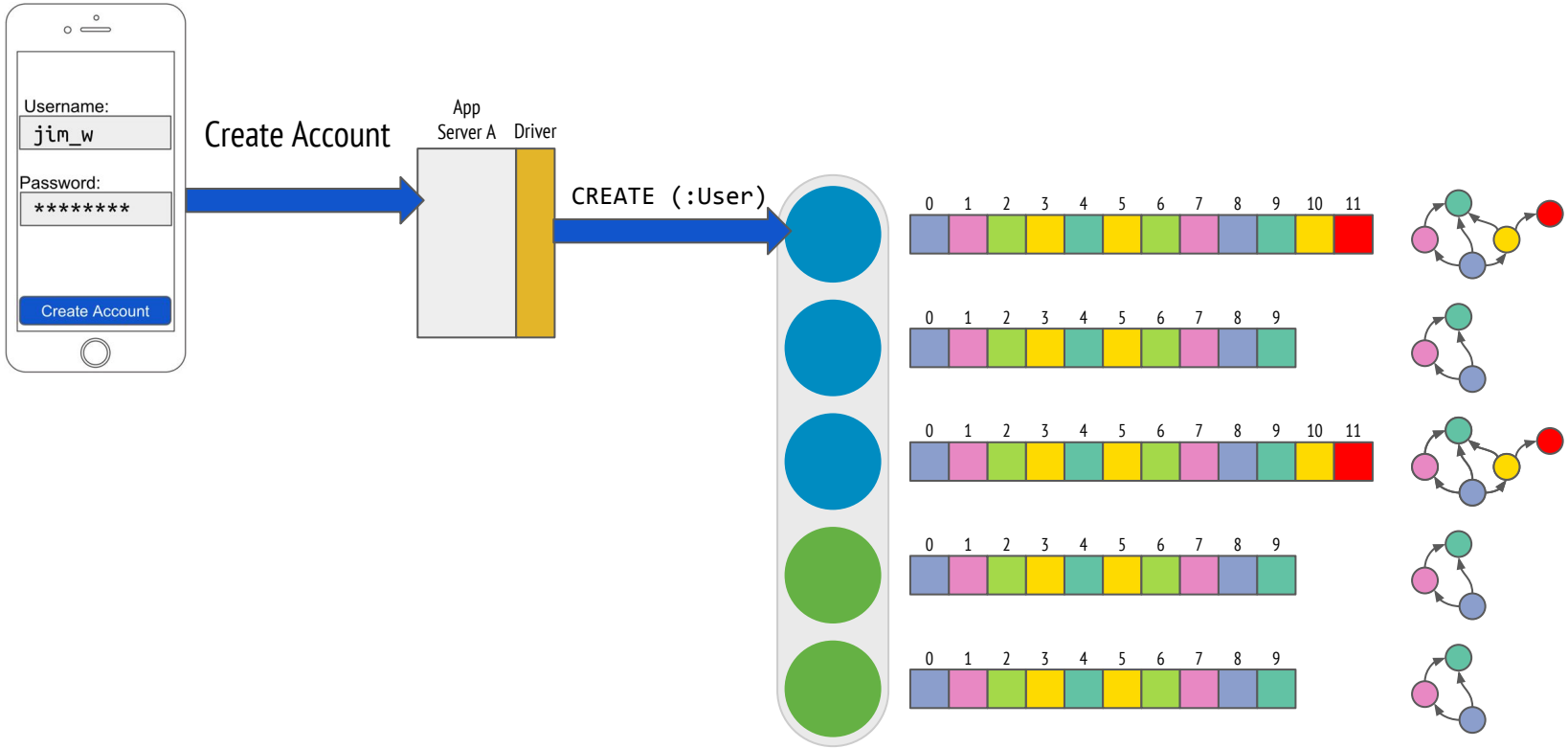
- Session token
- String (for portability)
- Opaque to application
- Represents ultimate user's most recent view of the graph
- More capabilities to come

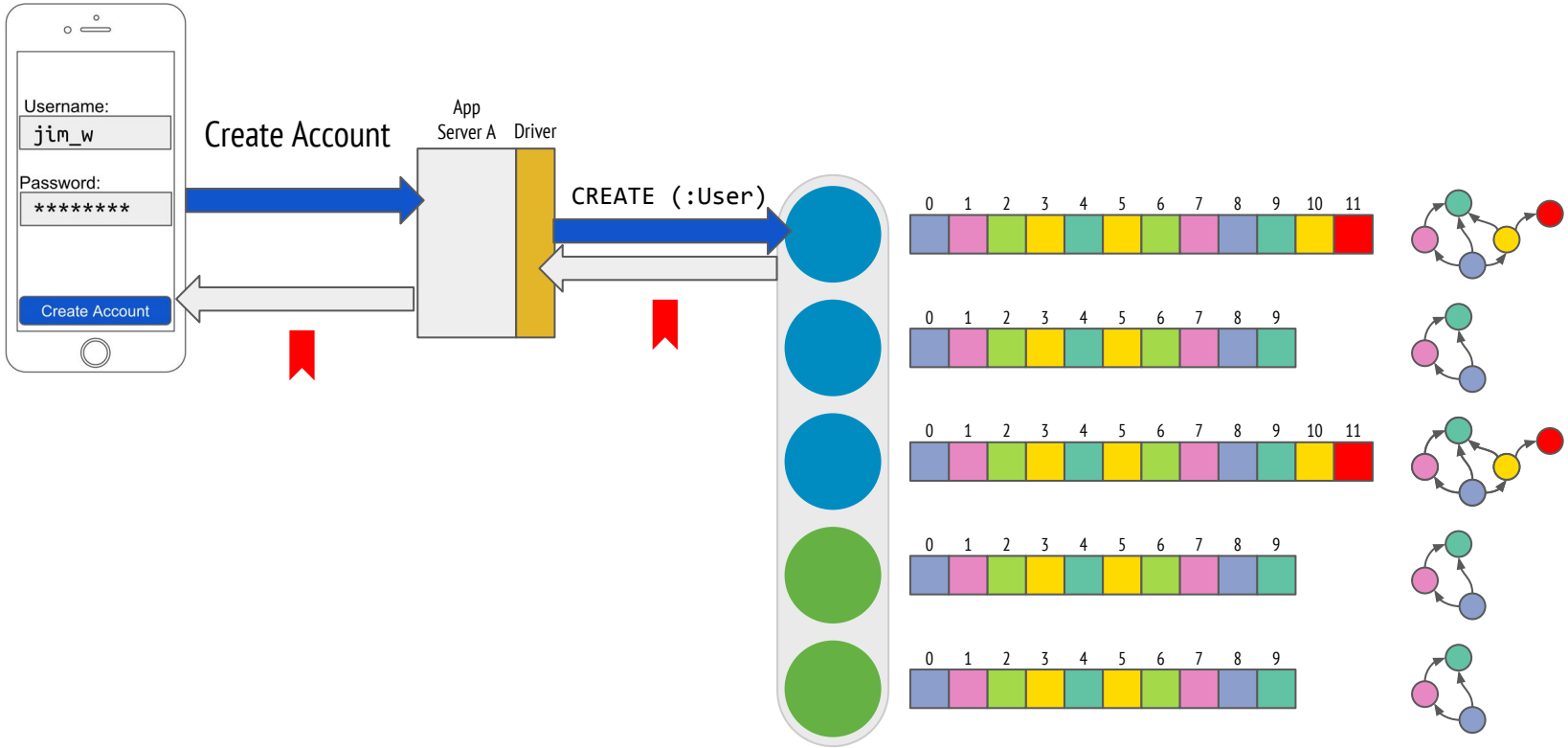

Username:
jim_w
Password:
****
Create Account
Create Account
App
Server A
Driver

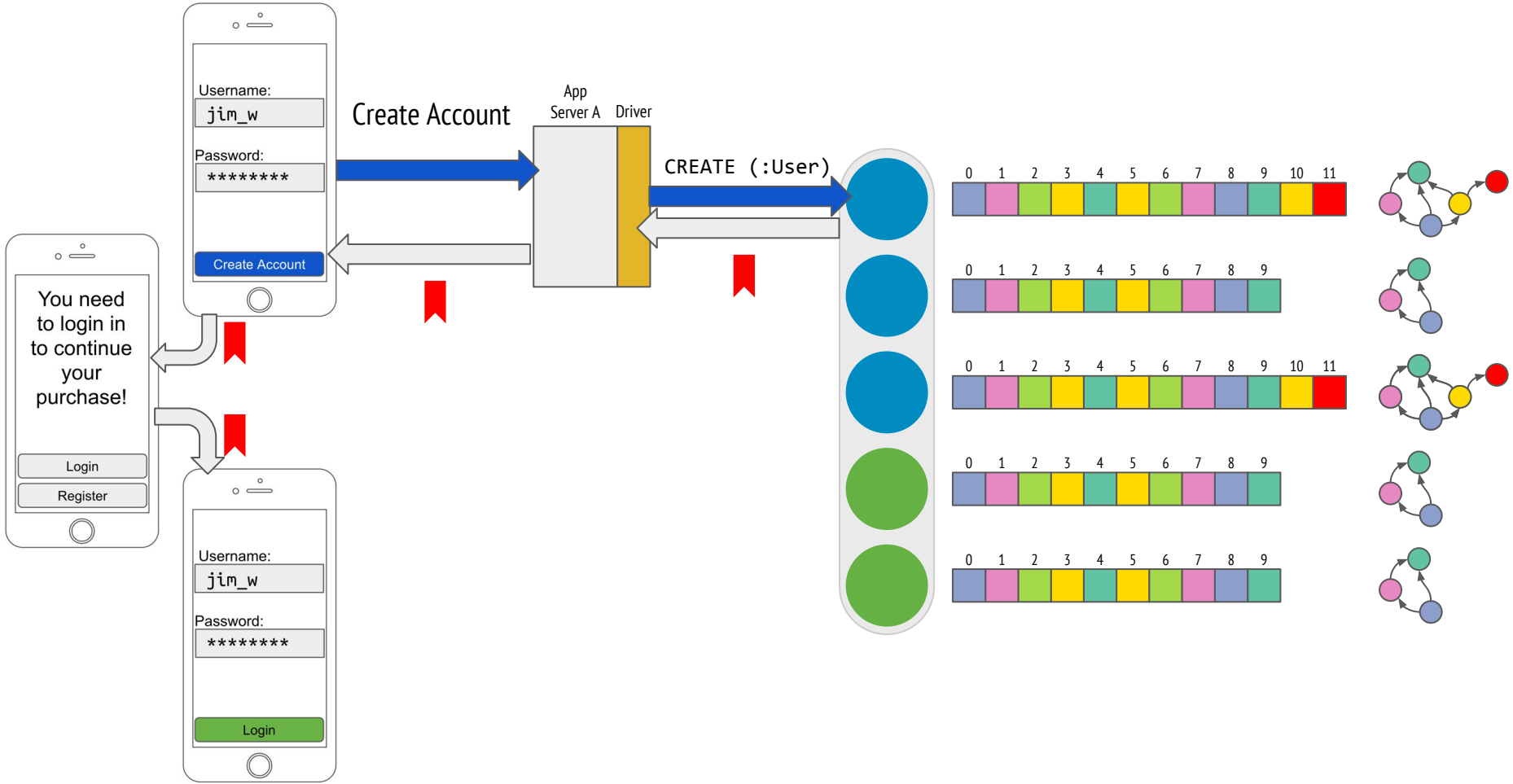

0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9

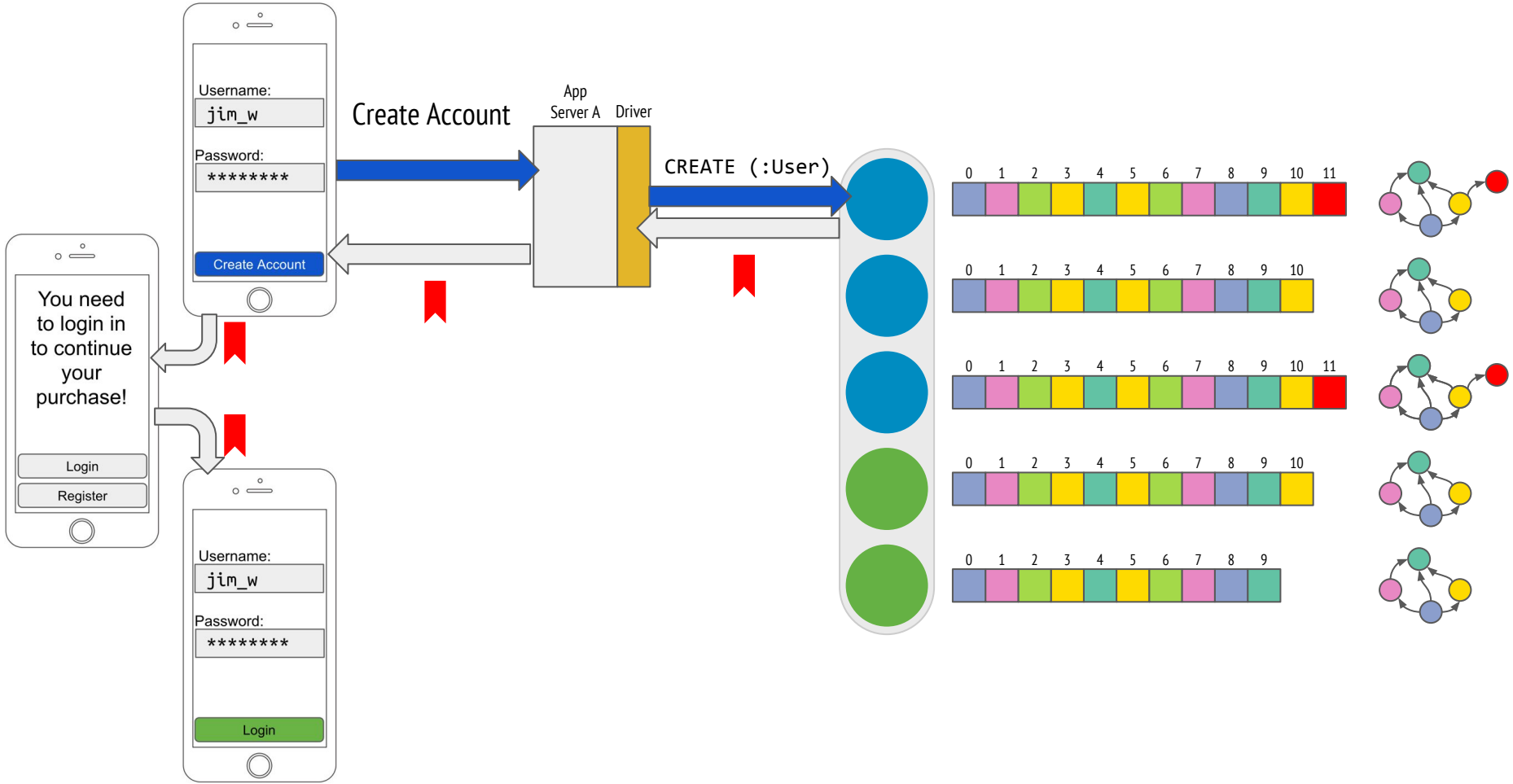

Username:
jim_w
Password:
****
Create Account
Create Account
App
Server A
Driver
CREATE (:User)
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9

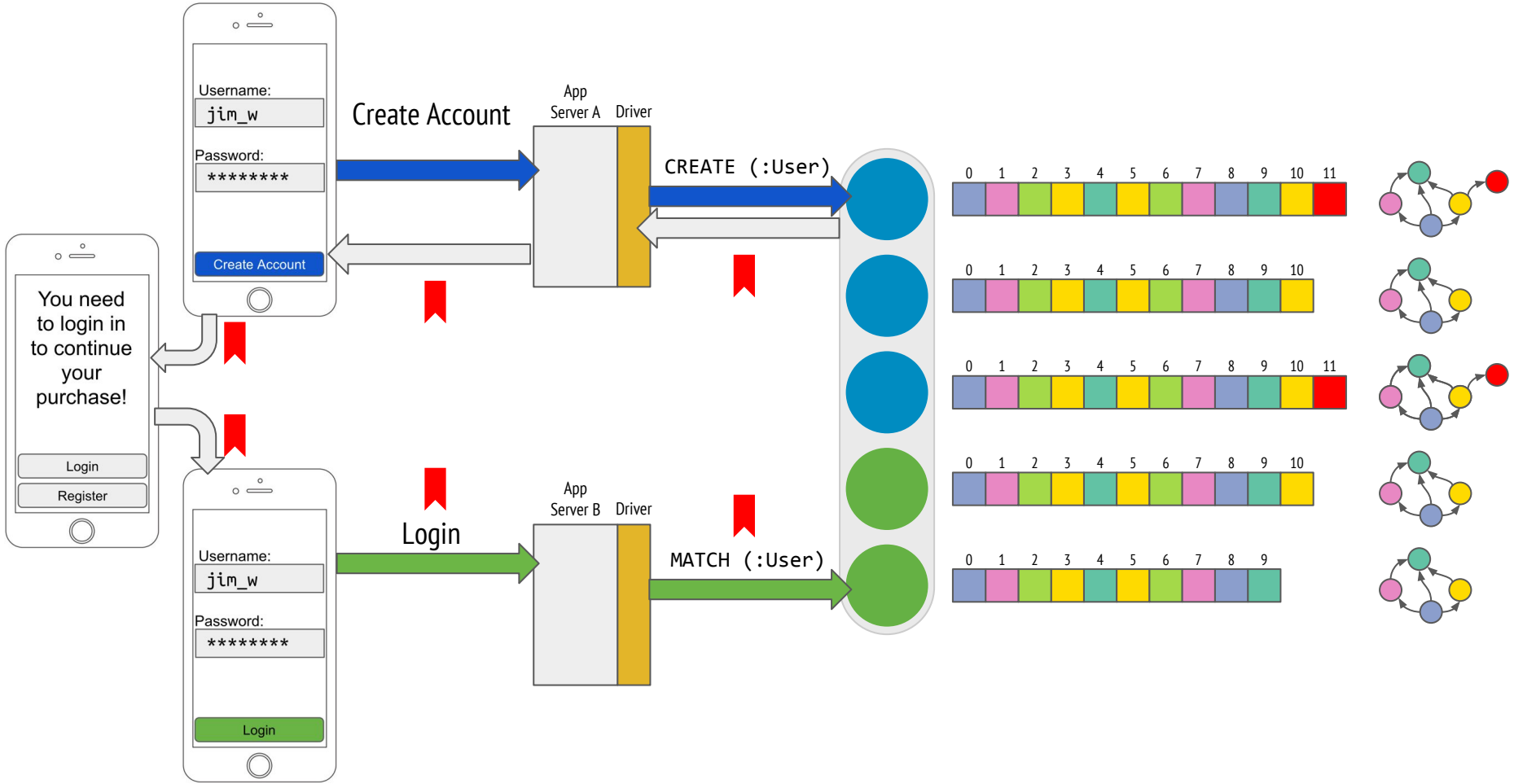

Username:
jim_w
Password:
****
Create Account
Create Account
App
Server A
Driver
CREATE (:User)
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9

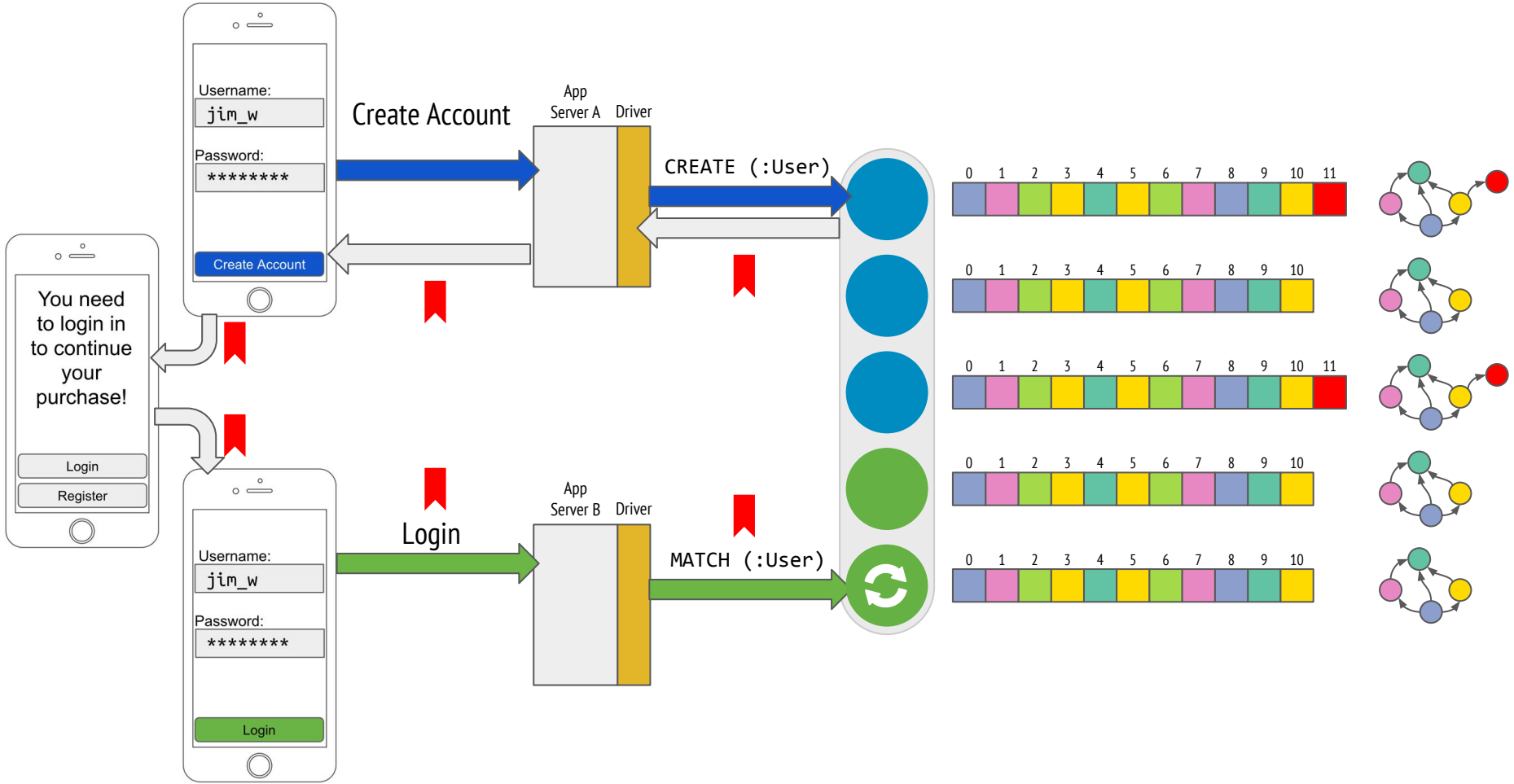

Username:
jim_w
Password:
****
Create Account
Create Account
App
Server A
Driver
CREATE (:User)
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9

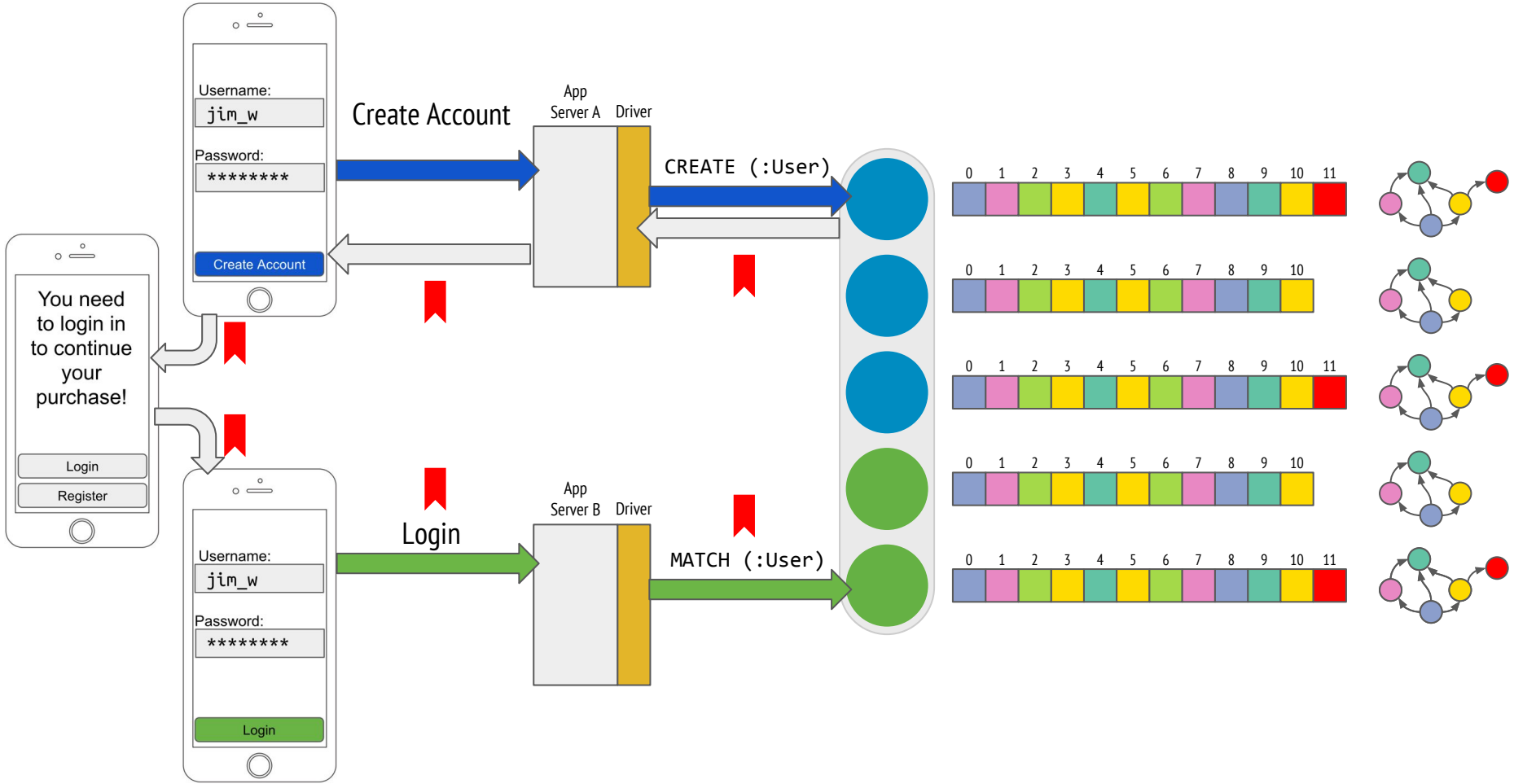

Username:
jim_w
Password:
****
Create Account
Create Account
App
Server A
Driver
CREATE (:User)
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9

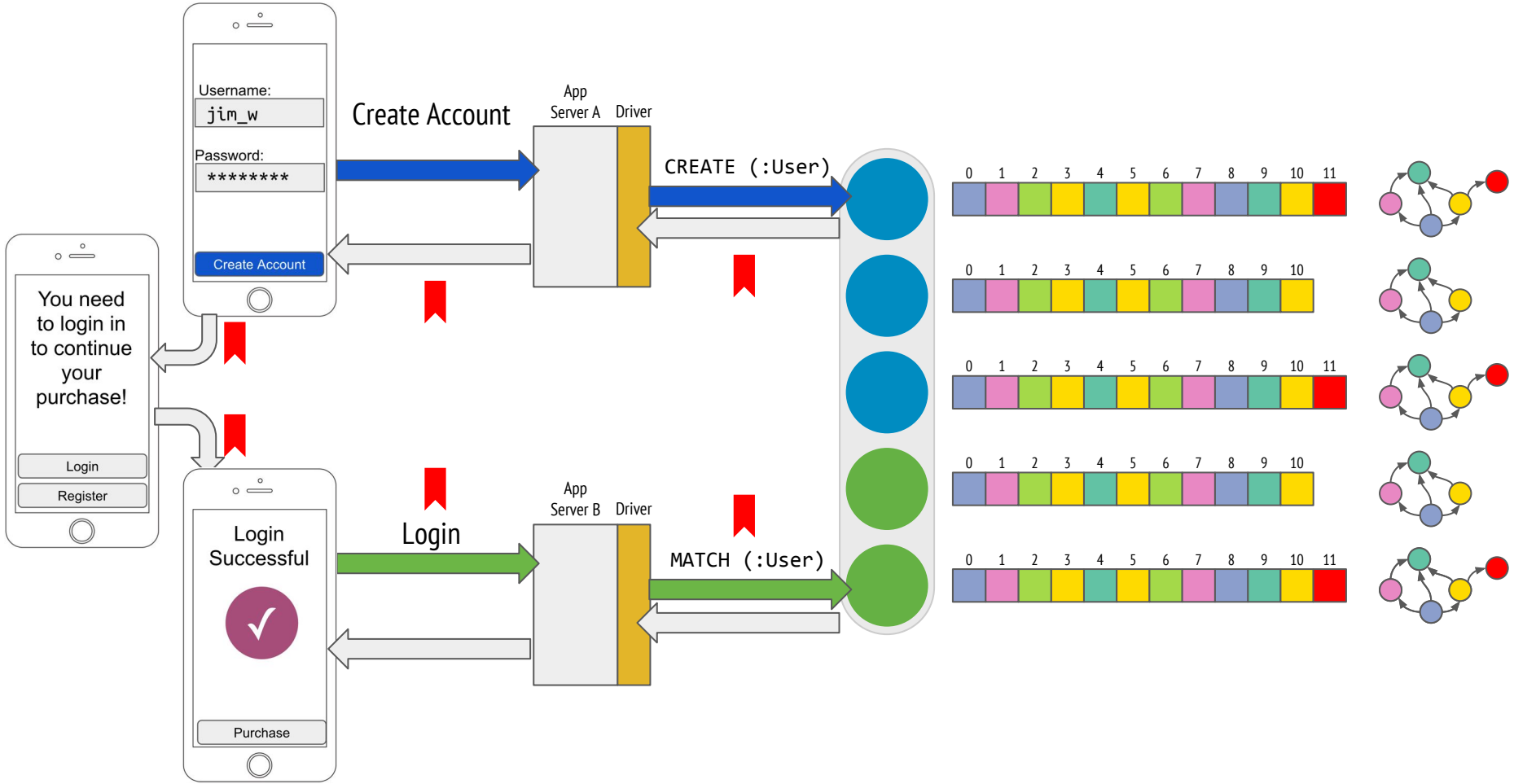

Username:
jim_w
Password:
****
Create Account
Create Account
App
Server A
Driver
CREATE (:User)
You need to login in to continue your purchase!
Login
Register
Username:
jim_w
Password:
****
Login
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9


Username:
jim_w
Password:
****
Create Account
Create Account
App
Server A
Driver
CREATE (:User)
You need to login in to continue your purchase!
Login
Register
Username:
jim_w
Password:
****
Login
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9


Username:
jim_w
Password:
****
Create Account
Create Account
App Server A
Driver
CREATE (:User)
You need to login in to continue your purchase!
Login
Register
Username:
jim_w
Password:
****
Login
Login
App Server B
Driver
MATCH (:User)
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9

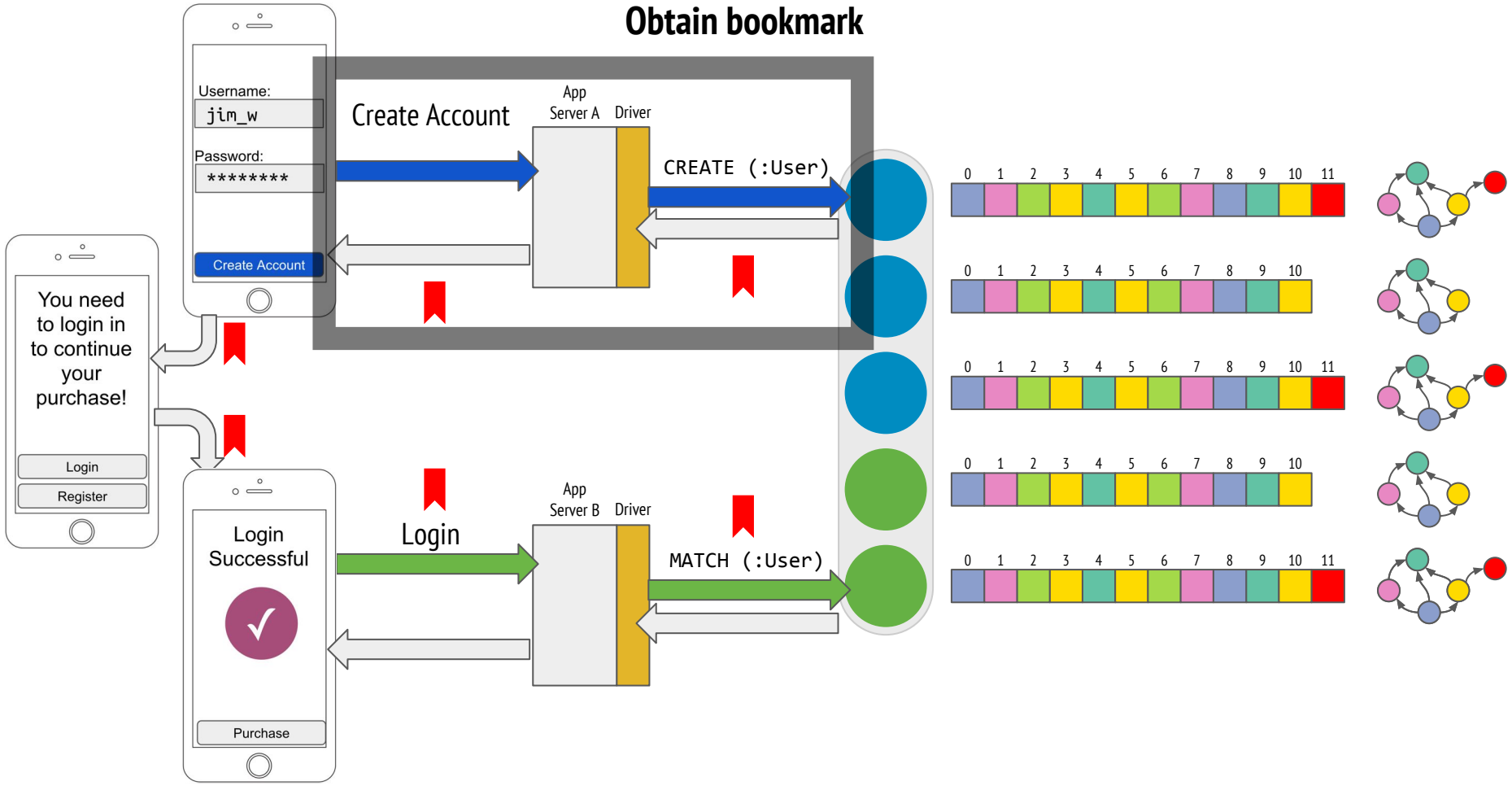

Username:
jim_w
Password:
****
Create Account
Create Account
App
Server A
Driver
CREATE (:User)
You need to login in to continue your purchase!
Login
Register
Username:
jim_w
Password:
****
Login
Login
App
Server B
Driver
MATCH (:User)
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9 10


Username:
jim_w
Password:
****
Create Account
Create Account
App
Server A
Driver
CREATE (:User)
You need to login in to continue your purchase!
Login
Register
Username:
jim_w
Password:
****
Login
Login
App
Server B
Driver
MATCH (:User)
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9 10 11


Username:
jim_w
Password:
****
Create Account
You need to login in to continue your purchase!
Login
Register
Login Successful
Purchase
Create Account
Login
App Server A
Driver
App Server B
Driver
CREATE (:User)
MATCH (:User)
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9 10 11

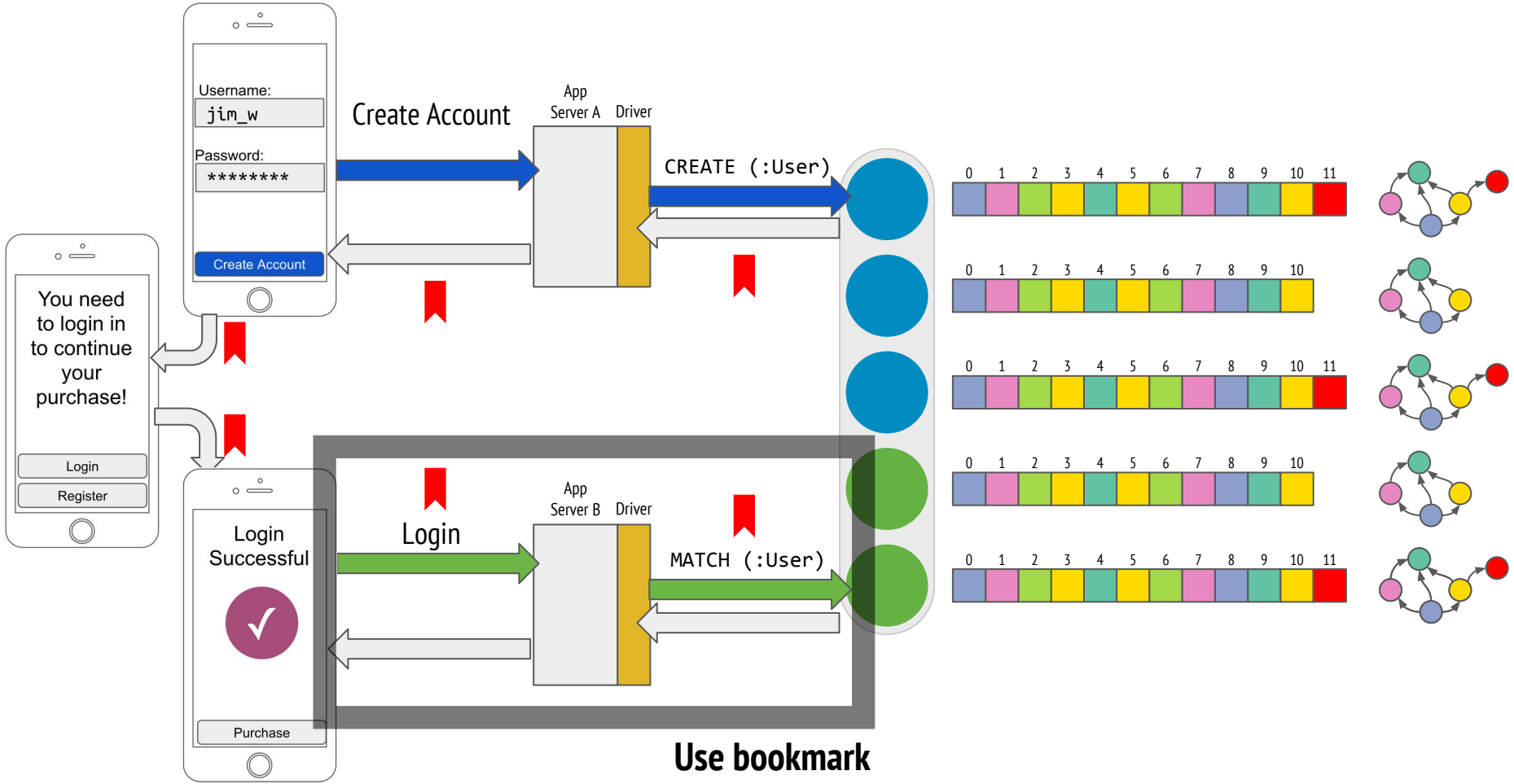
Obtain bookmark

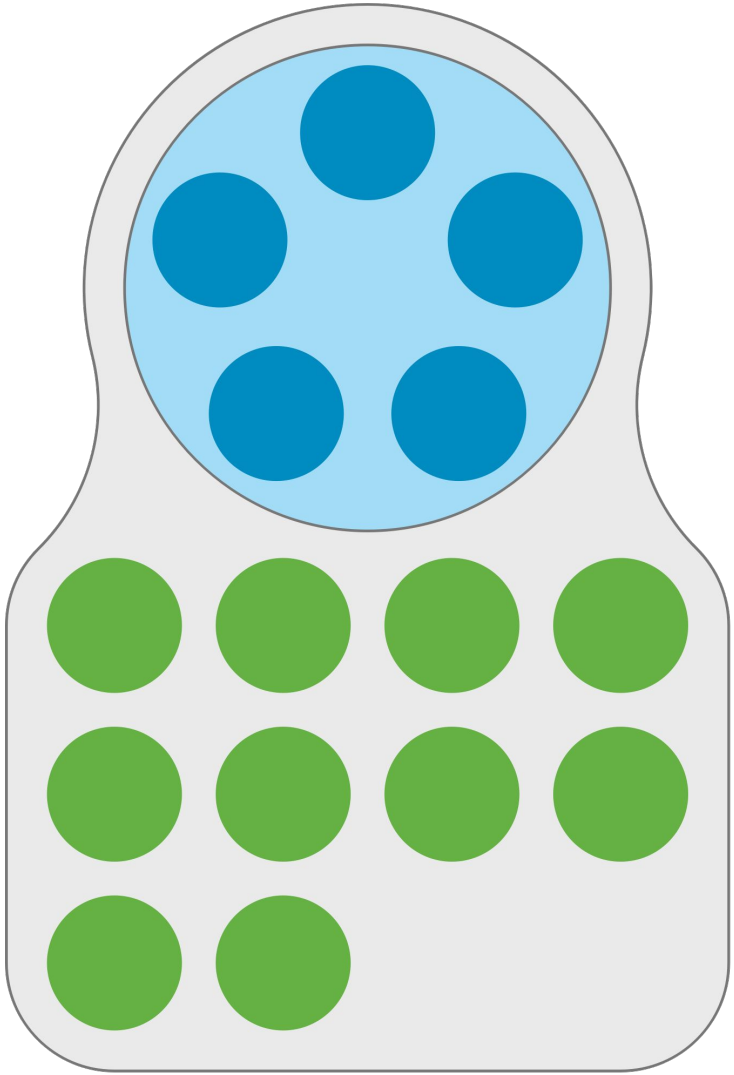
```
try ( Session session = driver.session( AccessMode.WRITE ) )
{
    try ( Transaction tx = session.beginTransaction() )
    {
        tx.run( "CREATE (user:User {userId: {userId}, passwordHash: {passwordHash})",
                parameters( "userId", userId, "passwordHash", passwordHash ) );

        tx.success();
    }

    String bookmark = session.lastBookmark();
}
```


Obtain bookmark
Username:
jim_w
Password:
****
Create Account
Create Account
App
Server A
Driver
CREATE (:User)
You need to login in to continue your purchase!
Login
Register
Login Successful
Purchase
Login
App
Server B
Driver
MATCH (:User)
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9 10 11

Use a bookmark

```
try ( Session session = driver.session( AccessMode.READ ) )
{
    try ( Transaction tx = session.beginTransaction( bookmark ) )
    {
        tx.run( "MATCH (user:User {userId: {userId}}) RETURN *",
                parameters( "userId", userId ) );

        tx.success();
    }
}
```


Username:
jim_w
Password:
****
Create Account
Create Account
App
Server A
Driver
CREATE (:User)
You need to login in to continue your purchase!
Login
Register
Login Successful
Purchase
Login
App
Server B
Driver
MATCH (:User)
Use bookmark



Core

Updating the graph

Continuous operation

Read Replicas

Queries, analysis, reporting

At large scale

Questions?