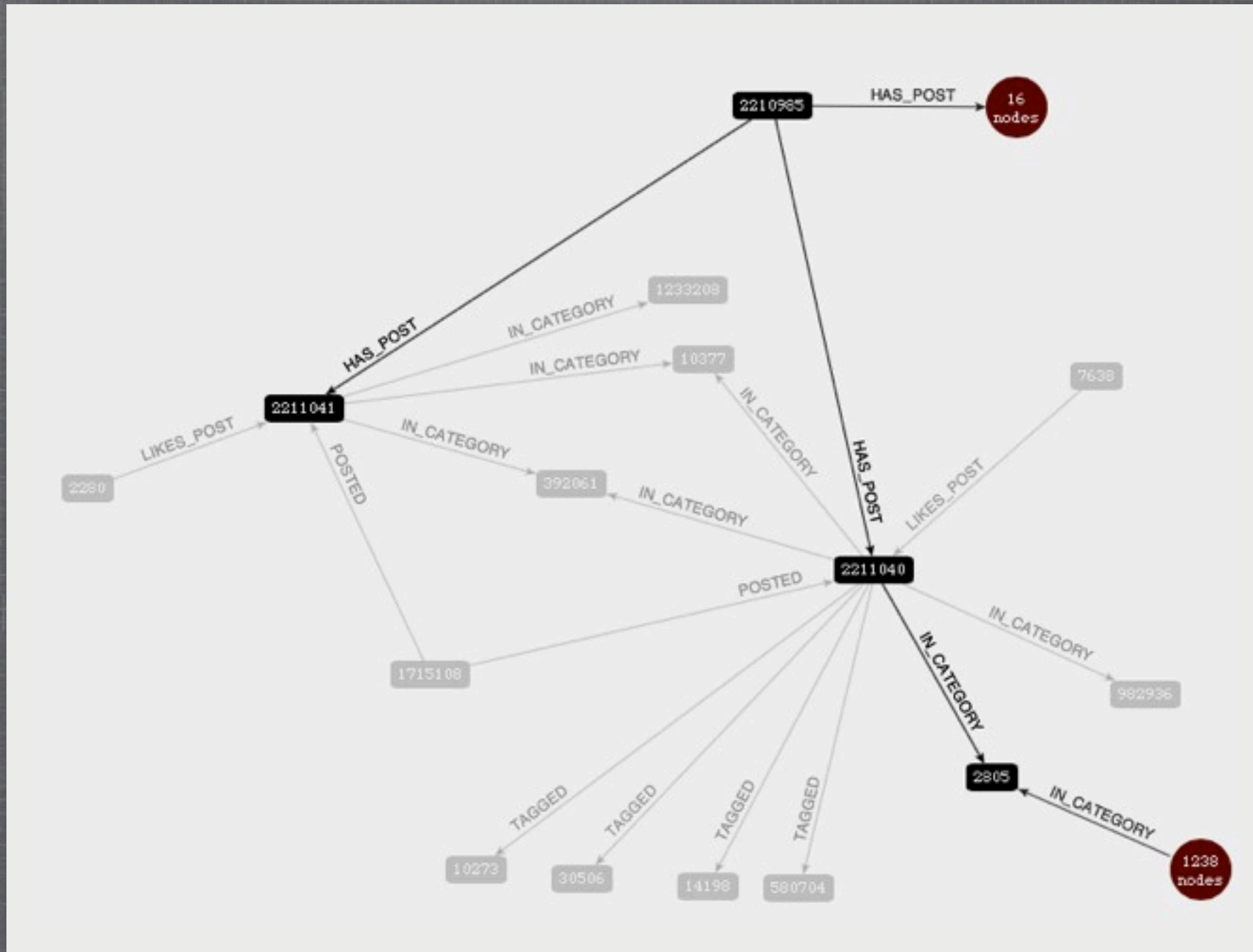# EXPLORING A GRAPH

Modeling, Querying, and Visualizing Graph Data

THE RAW DATA

- 5.1 GB txt file

- 1,196,111 Lines

- JSON Encoded, 1 Post/Line

- 5 weeks of Posts and "likes"

THE RAW DATA

{ "date_gmt":"2012-03-28 19:59:45",
 "language": "en",
 "author": "5",
 "url": "http://memes.com/5",
 "title": "Memes You Love",
 "blog": "4",
 "post_id": "1329369",
 "tags": ["cats","honey badgers","kids","bears"],
 "blogname": "The Internet Culture",
 "date": "2012-03-28 19:59:45",
 "content": "<h1>MEMES</h1><p>Are hilarious</p>",
 "categories": ["memes"],
 "likes": [{"dt": "2012-03-28 21:05:37", "uid": "31367867"}]
}

MODELING

RDBMS vs. GraphDB

```
{ "date_gmt":"2012-03-28 19:59:45",
  "language": "en",
 "author": "5",
 "url": "http://memes.com/5",
 "blog": "4",
 "post_id": "1329369",
 "tags": ["cats","honey badgers","kids","bears"],
 "date": "2012-03-28 19:59:45",
 "content": "<h1>MEMES</h1><p>Are hilarious</p>",
 "categories": ["memes"],
 "likes": [ {"dt": "2012-03-28 21:05:37", "uid": "31367867"}]
}
```

MODELING - RDBMS

post_id	blog	author	language	...
1329369	4	1	1	...

tag_id	value
1	cats

category_id	value
1	Memes

blog_id	Name
4	...

user_id	value
1	David

lang_id	value
1	en

post_id	tag_id
1329369	1

post_id	categ_id
1329369	1

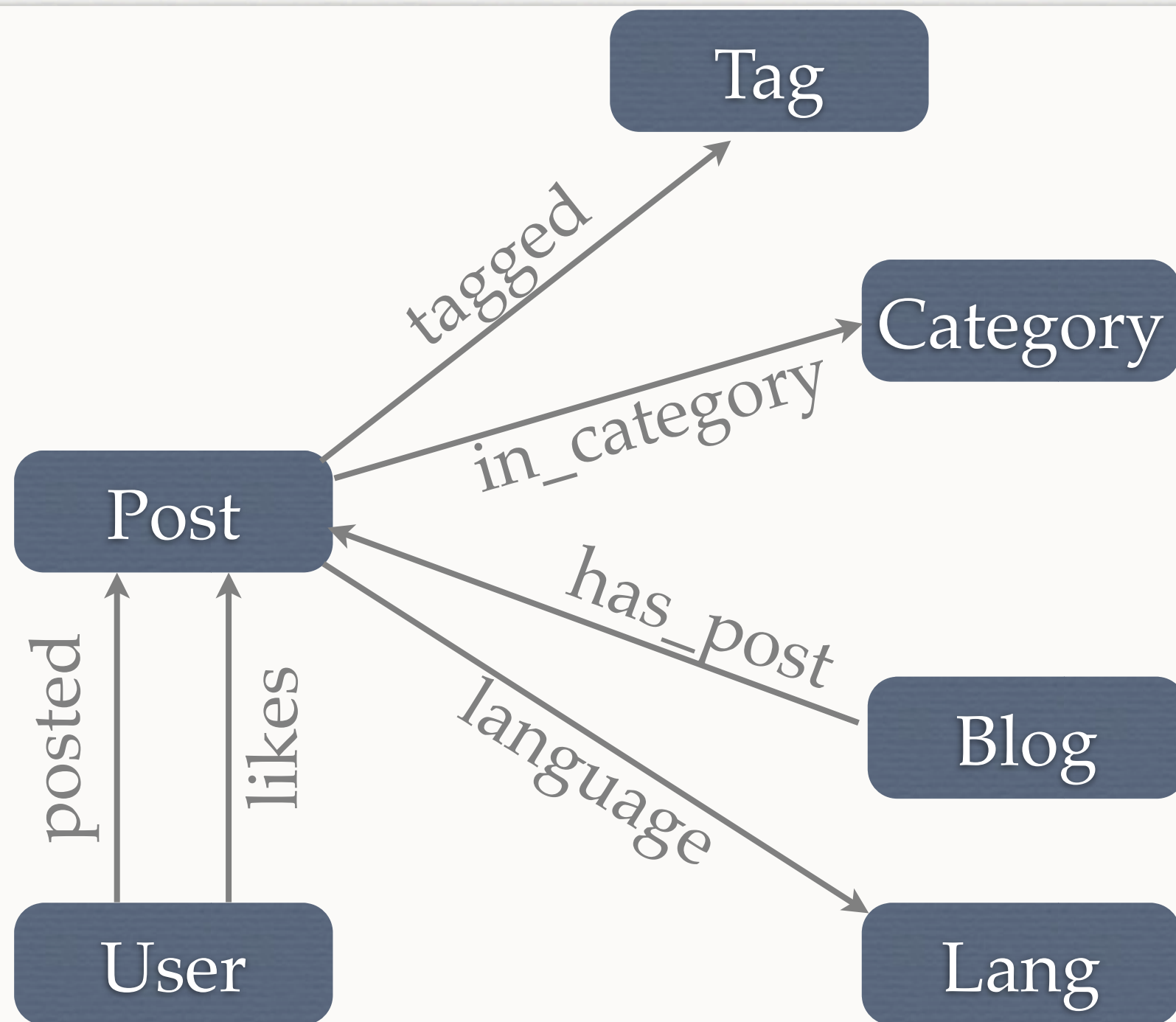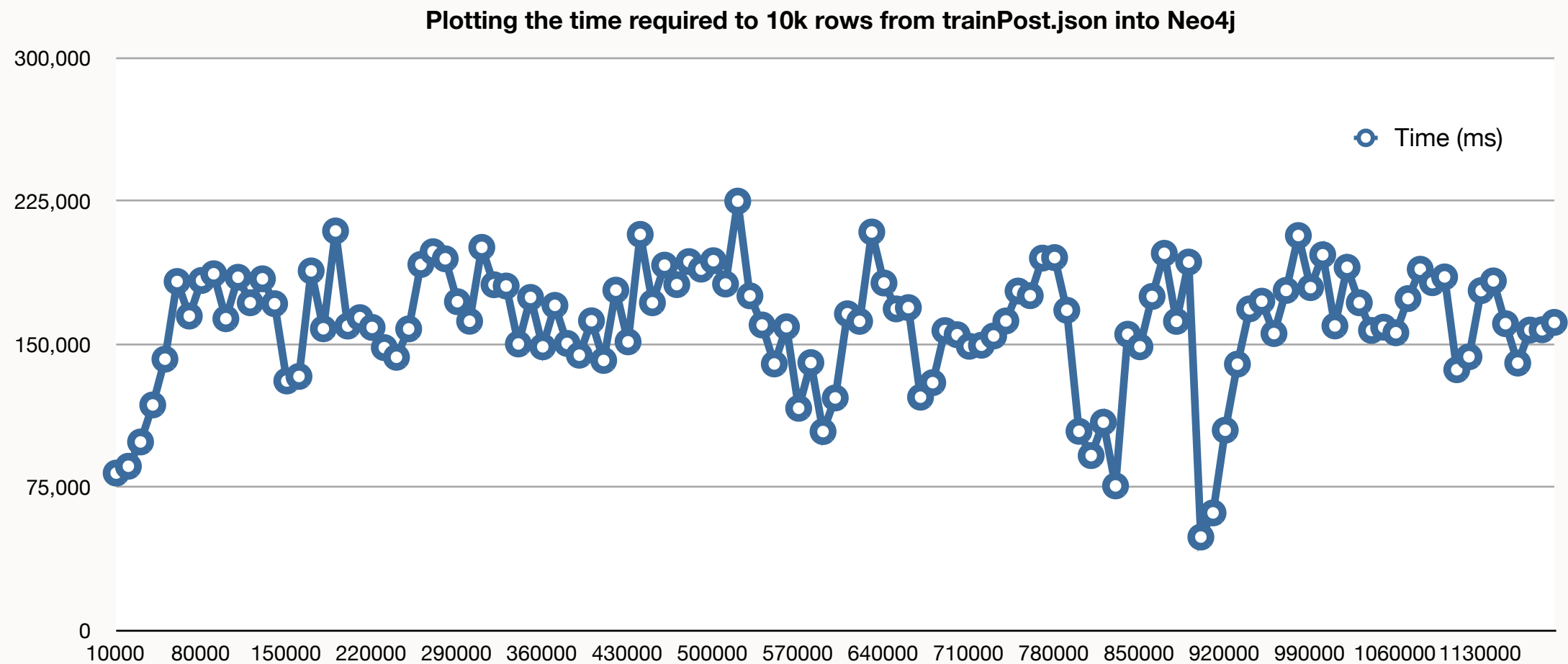| user_id	post_id
42	1329369

MODELING

RDBMS vs. GraphDB

{ "date_gmt":"2012-03-28 19:59:45",
  "language": "en",
 "author": "5",
 "url": "http://memes.com/5",
 "blog": "4",
 "post_id": "1329369",
 "tags": ["cats","honey badgers","kids","bears"],
 "date": "2012-03-28 19:59:45",
 "content": "<h1>MEMES</h1><p>Are hilarious</p>",
 "categories": ["memes"],
 "likes": [{"dt": "2012-03-28 21:05:37", "uid": "31367867"}]
}

MODELING - GRAPHDB

IMPORTING JSON -> NEO4J

Plotting the time required to 10k rows from trainPost.json into Neo4j

Total Import Time: 5.3 hrs

INDEXING

- Neo4j Auto-Indexing

- Per Type Indexing

- Reference Nodes

IMPORTING - STATS

Blogs	84,714
People	159,690
Categories	271,127
Posts	1,196,111
Tags	1,349,725
Total Nodes	3,061,367

Likes	2,452,001

*1 minute 52 seconds

VISUALIZATION

Neo4j Web Console

Gephi

QUERIES

- ## Top "Cat" Author

  ```
  start tag = node:tags(UID='cat')
  match tag<-[:TAGGED]-post<-[:POSTED]-author
  return distinct author.UID as CatAuthor, count(post) as
  count
  order by count desc
  limit 10
  ```

- ## Top "Cat" Liker

  ```
  start tag=node:tags(UID='cat')
  match tag<-[:TAGGED]-post<-[:LIKES_POST]-liker
  return distinct liker.UID as CatLiker, count(liker) as
  count
  order by count desc
  limit 10
  ```

QUERIES

- People who like the same posts as user 25564502

```
start person = node:people('UID:25564502')
match person-[:LIKES_POST]->()<-
[:LIKES_POST]-similar
return distinct similar
```