

I



Francis

I



B U S T L E

A highly pixelated, orange and brown version of the Reddit logo, featuring a stylized alien head with a antenna and a speech bubble.

@reconbot

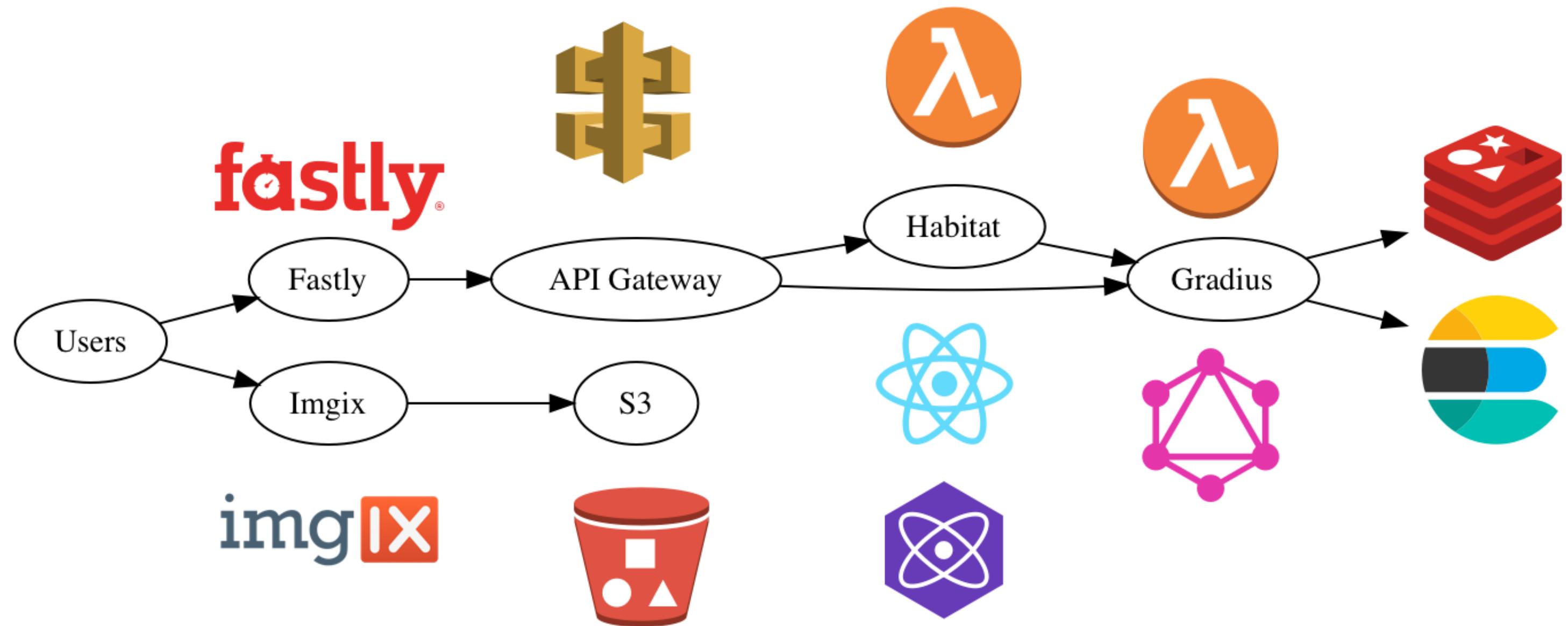
*Hey Francis, How  
did everyone at  
Bustle make the  
website so fast?*

– People

*We put everything  
in Redis.*

– Francis

We Live in Memory



Value distribution for Duration

Value: 24.4

Count: 497,969

500k

400k

300k

200k

100k

0

# API Response Time

1

5

10

50

100

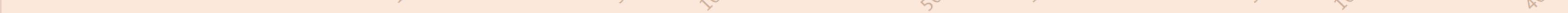
500

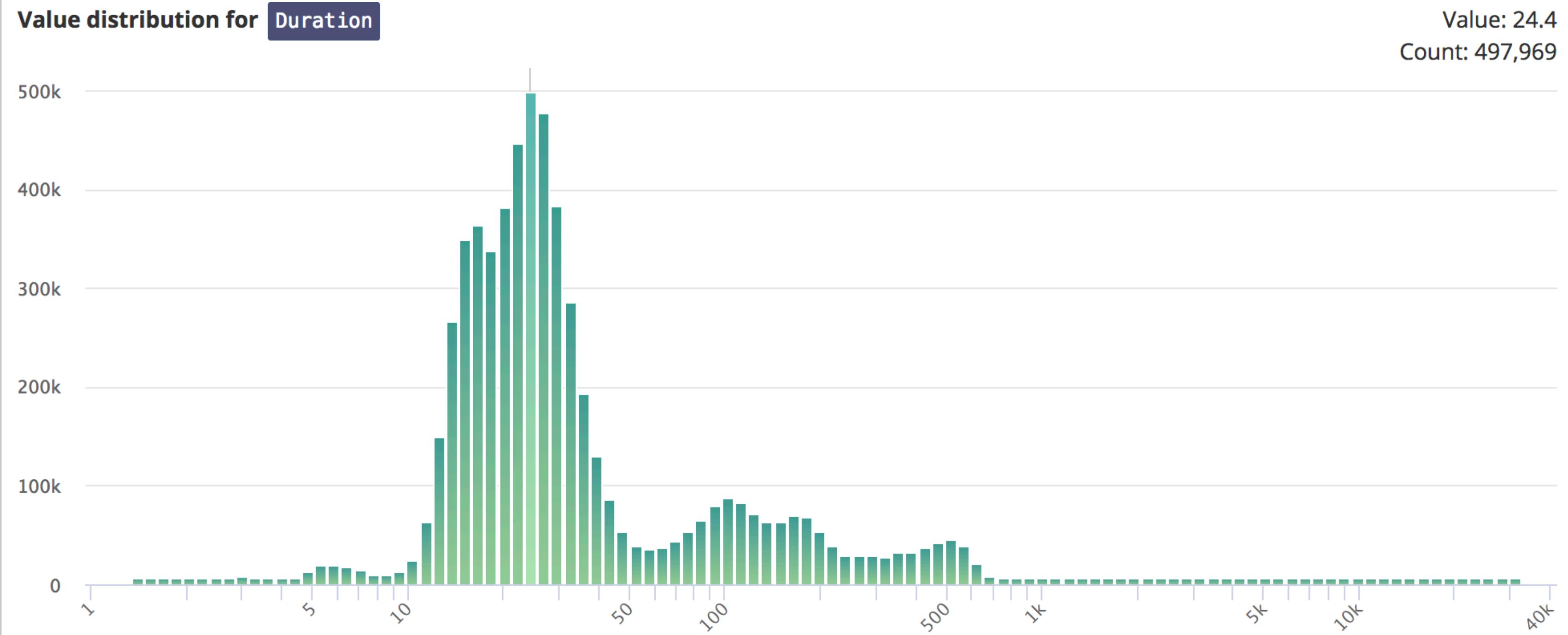
1k

5k

10k

40k





# Redis is our Primary Data Store

But that doesn't make it fast

*Redis is an **in-memory data structure store**, used as a database, cache and message broker. It supports data structures such as strings, hashes, lists, sets, sorted sets with range queries...*

*– Redis.io(ish)*

*Hey Francis, isn't  
that really  
dangerous?*

– 50/50 chance you'll say this

*No*

– 100% chance I'll say this

*I don't believe you*

**- 100% chance you're thinking this**

# Redis Is our Primary Data Store

# Redis Is our Primary Data Store

- 1s fsync of AOF

# Redis Is our Primary Data Store

- 1s fsync of AOF
- 1 hour snapshot RDB

# Redis Is our Primary Data Store

- 1s fsync of AOF
- 1 hour snapshot RDB
- Read replicas ready to take over really fast

# Redis Is our Primary Data Store

- 1s fsync of AOF
- 1 hour snapshot RDB
- Read replicas ready to take over really fast
- Good enough for Bustle's read heavy load

# Modeling GraphQL data in a redis GraphDB

☝️ is why we're fast



how can I pull off dad shoes?

Google Search

I'm Feeling Lucky

how can I pull off dad shoes? - X +

https://www.google.com/search?source=hp&ei=ZJg9W8OVGaKQ\_Qb: ...

## I Wore 'Dad Shoes' For A Week & They Were SO Much Cooler Than I ...

<https://www.bustle.com/.../i-wore-dad-shoes-for-a-week-they-were-so-much-cooler-th...> ▾

Mar 15, 2018 - My two reference points for the perfect **dad shoes** were a stylish Ryan ... I ended up pulling a blazer from the back of my closet and wearing a ...

## 'Dad' trainers are everywhere, but would you wear them?

<https://www.harpersbazaar.com/uk/fashion/.../chunky-dad-grandpa-trainers-trend/> ▾

Feb 7, 2018 - Spongy gym **shoes** are fast becoming the fashion trainer of choice. Here's how to pull them off.

## 8 "Dad Style" Moves That Are Actually Really Cool | GQ

<https://www.gq.com/story/dad-style-is-actually-cool> ▾

Jun 15, 2016 - These days, add "dad" to anything—dad jeans, dad bod, dad style—and you're implying that it's not hip, hopelessly **out of touch**. Then again ...

# I Wore 'Dad Shoes' For A Week & They Were SO Much Cooler Than I Thought They'd Be

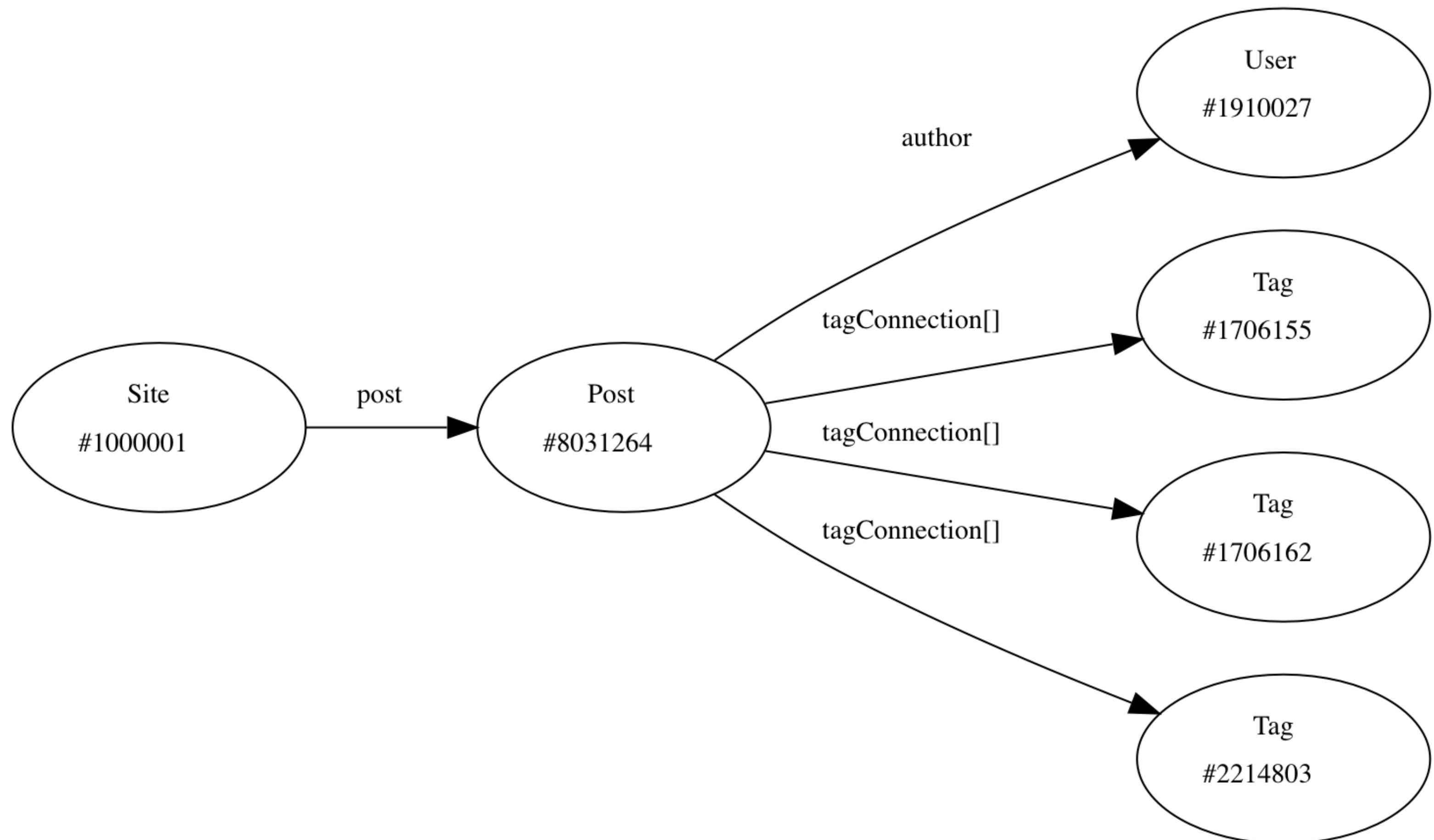
By DALE ARDEN CHONG | Mar 15 2018

f

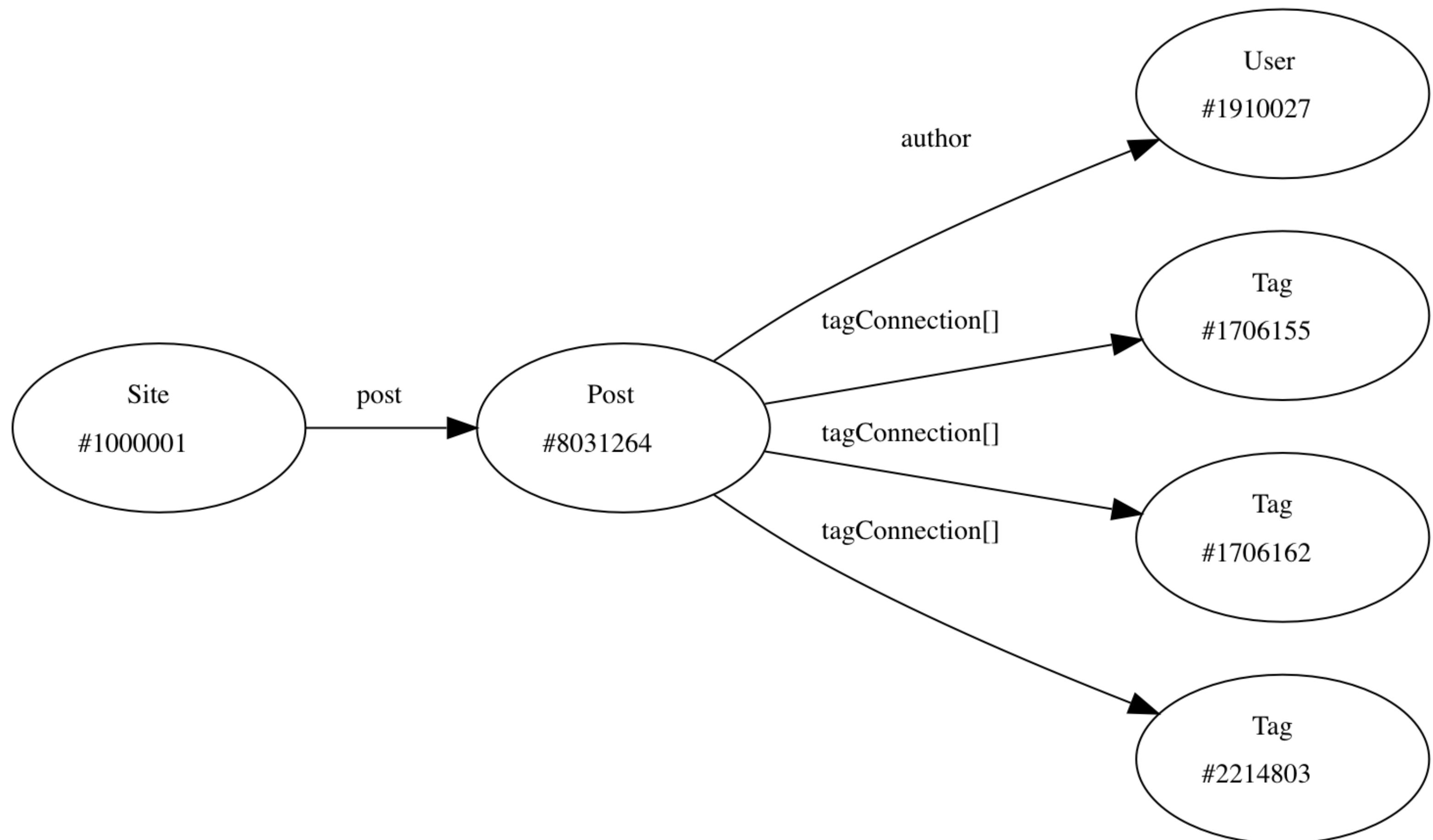


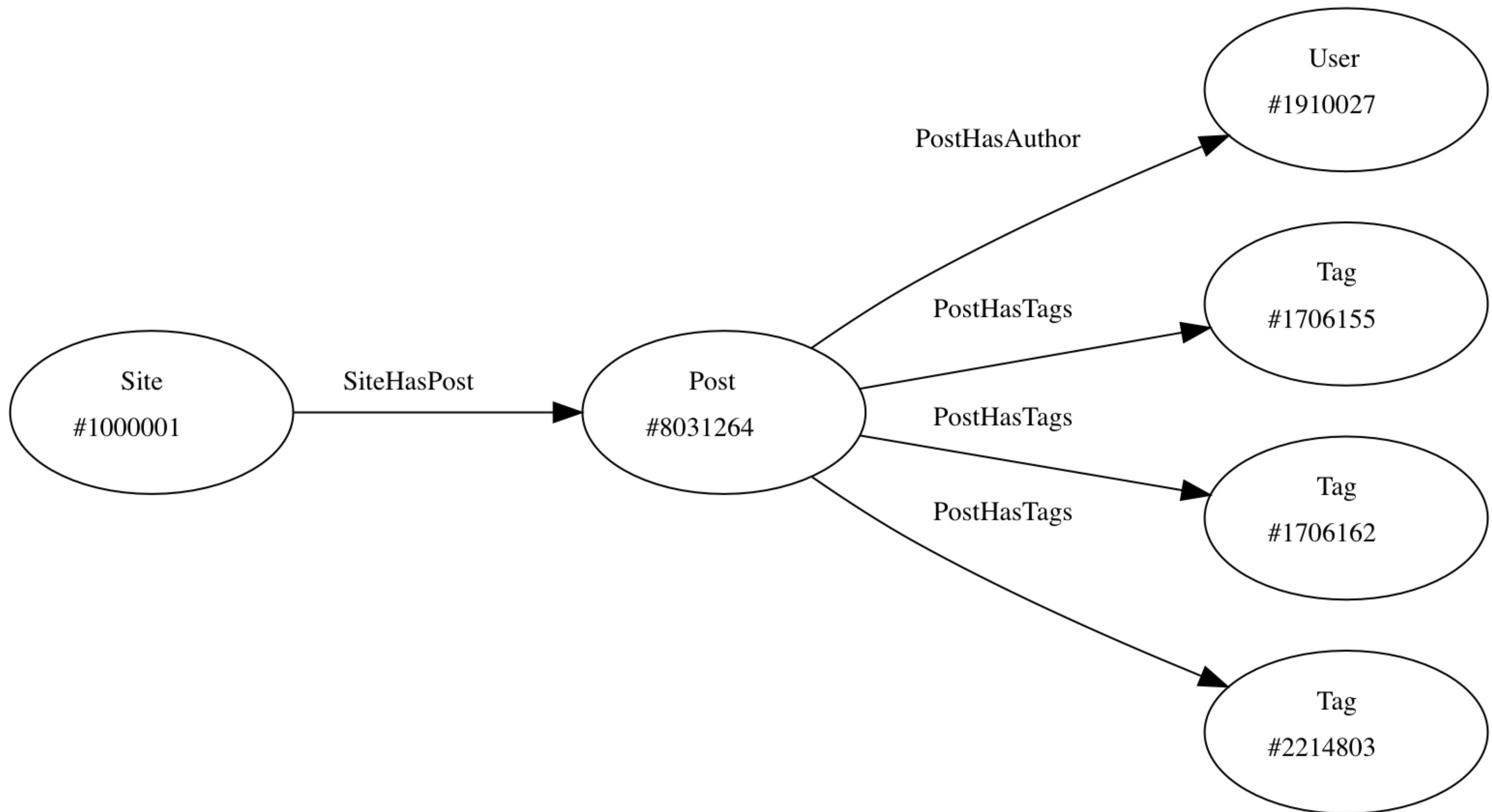
```
{  
  "data": {  
    "site": {  
      "name": "BUSTLE",  
      "post": {  
        "id": "8031264",  
        "__typename": "ArticlePost",  
        "title": "I Wore 'Dad Shoes' For A Week & They Were SO Much Cooler Than I Thought They'd Be",  
        "path": "/p/i-wore-dad-shoes-for-a-week-they-were-so-much-cooler-than-i-thought-theyd-be-8031264",  
        "publishedAt": 1521160559405,  
        "updatedAt": 1524672008862,  
        "author": {  
          "id": "1910027", "name": "Dale Arden Chong" },  
        "tagConnection": {  
          "nodes": [  
            { "id": "1706155", "name": "homepage" },  
            { "id": "1706162", "name": "fashion" },  
            { "id": "2214803", "name": "Freelancer" }  
          ]  
        }  
      }  
    }  
  }  
}
```

```
{  
  "data": {  
    "site": {  
      "id": "1000001",  
      "__typename": "Site"  
    },  
    "post": {  
      "id": "8031264",  
      "__typename": "ArticlePost",  
      "author": { "id": "1910027", "__typename": "User" },  
      "tagConnection": [  
        { "id": "1706155", "__typename": "Tag" },  
        { "id": "1706162", "__typename": "Tag" },  
        { "id": "2214803", "__typename": "Tag" }  
      ]  
    }  
  }  
}
```



# GraphDB





*Oh you mean, like  
Neo4j right?*

**– 99% of you**

*Sure, but it's  
faster and doesn't  
do any of the same  
things.*

– Francis

*Databases like trains aren't slow,  
the queries like the passengers are slow.*

- Ikai (a DBA) who I guess never took the subway

They're complaining  
about the database being  
slow but it's their  
queries that are slow!

They don't know how  
databases work! Where do  
they get off accusing  
the database servers!

Some database are slow, some  
queries are slow, it's all  
about tradeoffs

Bustle Traded  
Query Flexibility  
for Speed

# No slow queries<sup>1</sup>

<sup>1</sup>Ok it's just very hard

*Ok Francis, but how  
does it work?*

**– Get to it already**

# Nodes , Edges , and Indexes

# Nodes (future)

- GUID (int32)
- Node Type
- **data** as a Compressed Protocol Buffer
- Redis Hash

```
{ id, data, updateclock, ...metadata }
```
- Update count with a Lua Script to prevent update clobbering

# Nodes (now)

- GUID (int32)
- Node Type
- `JSON.stringify()` fields
- Redis Hash
  - { id, ...data, ...metadata }
- `HGETALL`, `HMSET`

# Node JSON

```
{  
  id: 1,  
  _nodeType: 'User',  
  name: 'Ein',  
  bigDeal: true,  
  createdAt: 1531334120311,  
  updatedAt: 1531334113447  
}
```

# Edges

# Edges

- An edge is Subject -> Predicate -> Object

# Edges

- An edge is Subject -> Predicate -> Object
- An edge is User -> UserHasPost -> Post

# Edges

- An edge is Subject -> Predicate -> Object
- An edge is User -> UserHasPost -> Post
- An edge can have weights  
 $User \rightarrow UserHasPost(42) \rightarrow Post$

# Edge JSON

```
{  
  subject: 5001,  
  predicate: 'UserHasPost',  
  object: 40238,  
  weight: 42  
}
```

# Sorted Sets

- Unique strings sorted by a score, then by member
- Think: hash with keys (members) sorted by their numeric values (scores) then by key
- Very fast read/write operations  $O(\log(N))$
- Can search by member or score

# Edges Hexastore

- 1 big Sorted Set with each edge in 6 orderings
- Pick an ordering and use `zrangebylex` to search

`ops:ObjectID:Predicate:SubjectID => 0`

`osp:ObjectID:SubjectID:Predicate => 0`

`pos:Predicate:ObjectID:SubjectID => 0`

`pso:Predicate:SubjectID:ObjectID => 0`

`sop:SubjectID:ObjectID:Predicate => 0`

`spo:SubjectID:Predicate:ObjectID => 0`

```
★ gradius git:(master) redis-cli
127.0.0.1:6379> ZADD hexastore 0 "1:HasAuthor:2"
(integer) 1
127.0.0.1:6379> ZADD hexastore 0 "1:HasPost:3"
(integer) 1
127.0.0.1:6379> ZRANGEBYLEX hexastore "[1:HasAuthor" "[1:HasAuthor\xff"
1) "1:HasAuthor:2"
127.0.0.1:6379> ZRANGEBYLEX hexastore "[1:" "[1:\xff"
1) "1:HasAuthor:2"
2) "1:HasPost:3"
127.0.0.1:6379> _
```

# Edges Hexastore

- 6 big Sorted Sets one for each ordering
- 4 bytes === 32bit GUID
- 4 bytes === sha256(predicate).slice(0, 4)
- 12 Byte Binary packed key **SSSSPPPP0000**
- Binary Packing saved 70% of memory
- still use `zrangebylex` to searching the members

# Edges Weighted Store

- Can store a weight on an edge
- Site -> HasPublishedPost(publishedAt) -> Post
- Can only search SP\* or OP\*
- Can search ranges of weights (eg, posts published today)
- Lots of tiny Sorted Sets
- Faster because of the smaller "n" in  $O(\log(n))$

# Edges Weighted Store

Each edge is stored two different ways

Key	Member	Score
"edge:s:\${subjectId}:\${predicate}"	objectId	weight
"edge:o:\${objectId}:\${predicate}"	subjectId	weight

```
★ gradius git:(master) redis-cli
127.0.0.1:6379> ZADD "edge:s:1:HasAuthor" 42 2
(integer) 1
127.0.0.1:6379> ZADD "edge:o:2:HasAuthor" 42 1
(integer) 1
127.0.0.1:6379> ZRANGEBYSCORE "edge:s:1:HasAuthor" -Inf Inf WITHSCORES LIMIT 0 1
1) "2"
2) "42"
127.0.0.1:6379> _
```

# Indexes

- `node:id` Sorted set of all node IDs
- `${type}:id` Sorted set of all node IDs of a type
- `${type}:${field}` Sorted set of unique values to  
ids

*How fast is it?*

– Me

# Loading Nodes is fast

```
while (true) {  
  const start = Date.now()  
  await graph.Site.find(1000001)  
  console.log(` ${Date.now() - start}ms`)  
}
```

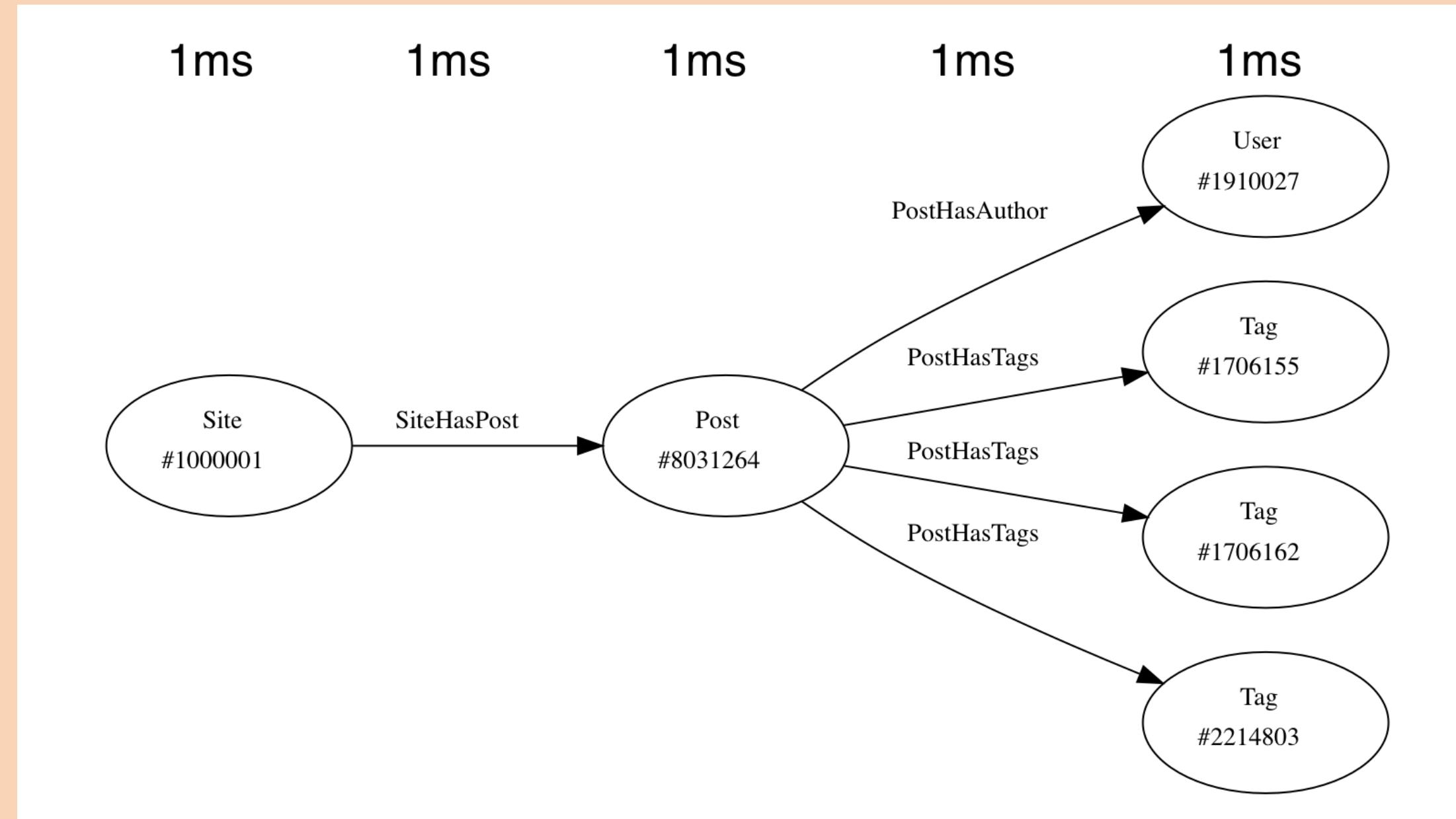
```
0ms  
1ms  
0ms  
0ms  
0ms  
0ms  
0ms  
0ms  
1ms  
0ms  
0ms
```

# Loading all the nodes is fast

```
while (true) {  
  const start = Date.now()  
  await Promise.all([  
    graph.Site.find(1000001),  
    graph.Post.find(8031264),  
    graph.User.find(1910027),  
    graph.Tag.find(1706155),  
    graph.Tag.find(1706162),  
    graph.Tag.find(2214803)  
  ])  
  console.log(` ${Date.now() - start}ms`)  
}
```

```
0ms  
1ms  
0ms  
0ms  
0ms  
0ms  
0ms  
0ms  
0ms  
1ms  
0ms  
0ms
```

# Command Batching Makes Redis Fast



# Not covered: ORM & Schemas

```
import { edge, Post, User } from './graph'
const ein = await User.create({ name: 'Ein the Dog', meta: {} })
const post = await Post.findBy('path', '/p/10-reasons-why-dogs-are-the-best')
await edge.create({
  subject: post,
  predicate: 'HasAuthor'
  object: ein
})

const authors = edge.makeIterator({ subject: post, predicate: 'HasAuthor' })
for await (author in authors) {
  await giveTreat(author)
}
```

# Not covered:

- Interfaces and Mixin Inheritance
- Typed Edge validations
- Named Edges (unique text values between two nodes)
- The many different Failures and Bugs in production and how we mitigate them

In Conclusion



# The secret of Dad shoes is a flowy skirt

Dale Arden Chong

On Day 5, I did a complete 180. With a newfound determination to make these sensible shoes look like Fashion with a capital F, I looked to Instagram to find new inspiration. What I found were looks juxtaposing the shoe's chunkiness with flowing skirts.

# In Conclusion

- Don't make slow queries possible
- Round trip time is a killer so batch your requests
- Store your data in a way that matches your access patterns

# Related Open Source

- bluestream Streams for Async functions
- mobiledoc-kit A toolkit for building WYSIWYG editors with Mobiledoc
- SAMMIE Serverless Application Model Made Infinitely Easier
- shep A framework for building JS Applications with AWS API Gateway and Lambda
- streaming-iterables Replace your streams with async iterators
- redis-loader An ioredis-like object that batches commands via dataloader

We Live In Memory  
(the story)  
[rbrrtr.com/post/1911](http://rbrrtr.com/post/1911)

Thanks

