NATURAL INTELLIGENCE

Neptune Graph Database

0 to Production

Ohad Israeli
Platform Architect @ NI

Ohad.Israeli@naturalint.com

Geektime
<CODE>

Jun 2021

NATURAL INTELLIGENCE | Top10.com

Graphs are all around us

NATURAL INTELLIGENCE

Ohad Israeli
Platform Architect @ NI
20+ Years of Software Architecture

NATURAL INTELLIGENCE

What are we going to talk about today?

Sharing my experience of mapping entities @ NI using AWS Neptune / Kafka / NestJS

90% 5% 5%

1. **Why** graph database ?
2. **How** did we use it ?
3. **What** we learned ?
4. **What** you need to know ?

setTimeout(presentationEnd, 30min, ‘lets start');

NATURAL INTELLIGENCE | Top10.com

Natural Intelligence in a Nutshell

Household

Shopping

Tech

Lifestyle

Family

Health & Wellness

Finance

Entertainment

Personal Growth

NATURAL INTELLIGENCE | Top10.com

Natural Intelligence in a Nutshell

6M

Leads generated annually

9M

Monthly Unique Visitors

130+

Countries Reached

10

Years of Expertise

320+

Talented Employees

NATURAL INTELLIGENCE | Top10.com

Natural Intelligence R&D in a Nutshell

NodeJS / NestJS

Serving platform + Backend Microservices

Java / Python

Data science / Machine learning

KAFKA / Kinesis

Event Bus

AWS

Fully invested

meetup.com/CodeNaturally

www.naturalint.com/jobs Full Stack, Frontend, Data Scientists, QA Lead, R&D Directors, Senior DevOps ...

NATURAL INTELLIGENCE

OK, let’s get started

Chapter 1

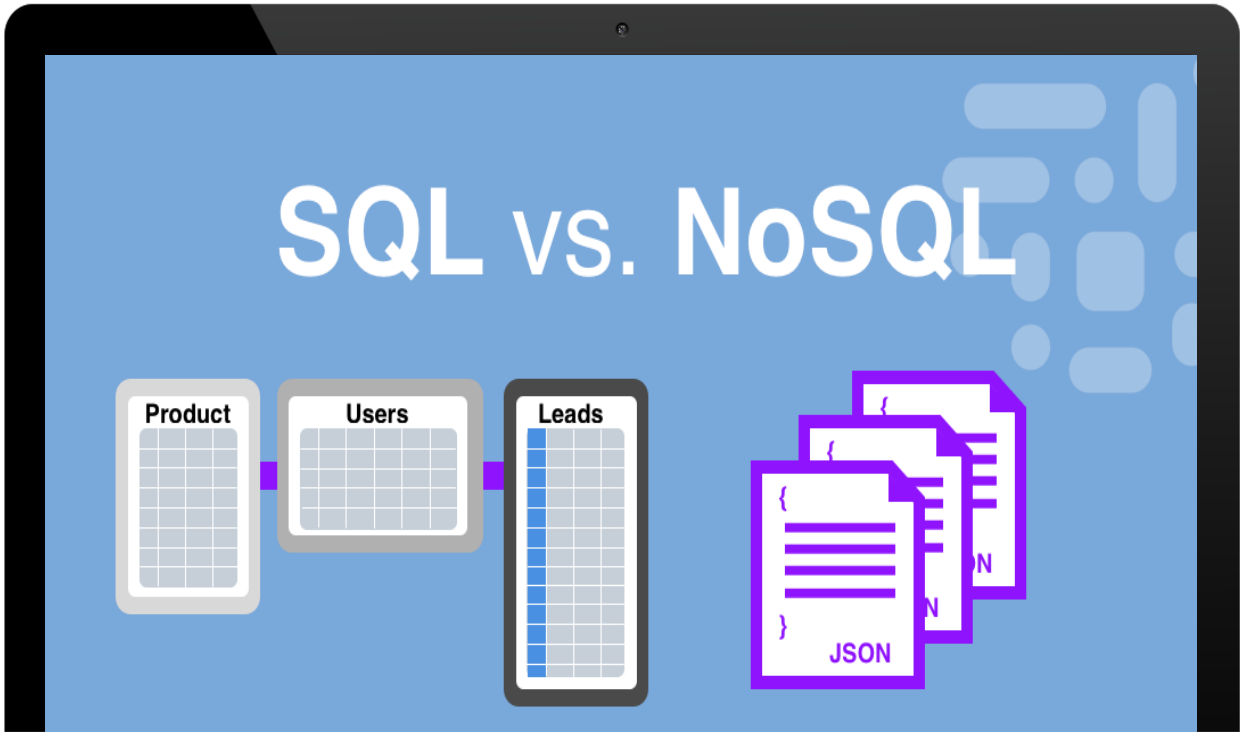
Graph Data What ?

NATURAL INTELLIGENCE | Top10.com

Databases Types

Relational Databases

Sql server
Oracle
DB2
MySQL
Postgresql
...
...



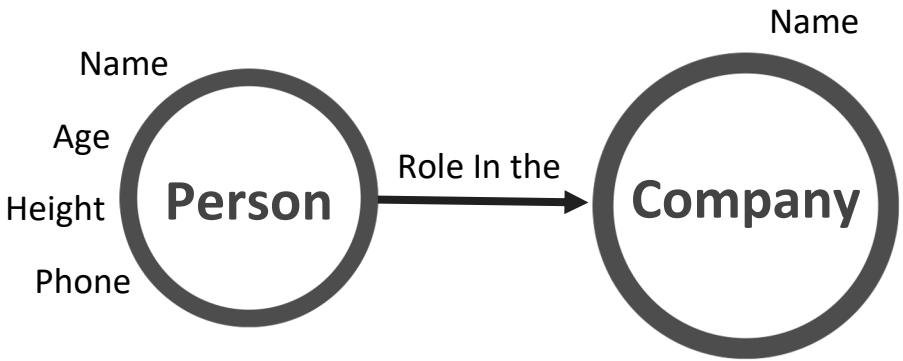
Document Databases

Aerospike
CouchDB
MongoDB
Elasticsearch
Solr
...
...

NATURAL INTELLIGENCE

What is Graph Database ?

- No pre-defined model
- Data modeled like a drawing on a whiteboard
- Relationships between data as equally important to the data itself



NATURAL INTELLIGENCE

Graph Databases

Graph Databases	Query Language
AWS Neptune	Gremlin
NEO4J	SparQL
DataStax	Cypher
JanusGraph	nGQL
OrientDB	Proprietary
ArangoDB	…
Dgraph	…
NabulaGraph	
…	
…	

Gremlin

```
g.V(person).valueMap();
```

SparQL

```
select ?p where {?p}
```

Cypher

```
MATCH (n:character {name:”person"}) RETURN properties(n)
```

nGQL

```
FETCH PROP ON character hash(”person");
```

Amazon Neptune neo4j DATASTAX OrientDB ArangoDB Dgraph NebulaGraph

NATURAL INTELLIGENCE

Coding time start in …

Gremlin $G = (V, E)$

jupyter

SCAN ME

Apache TinkerPop

Amazon Neptune

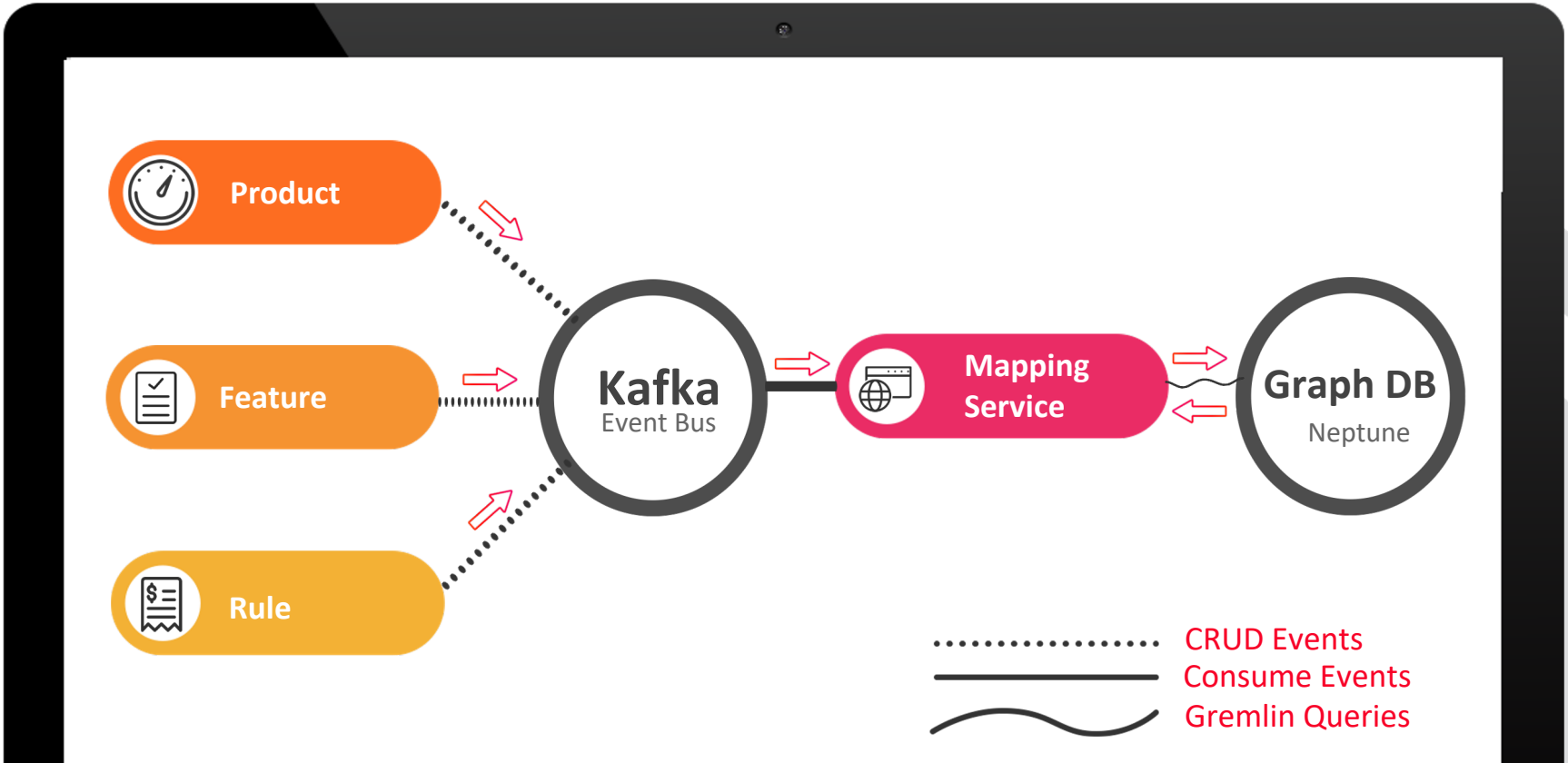
github.com / israelio / graphdb-intro

NATURAL INTELLIGENCE

Chapter 2

How did we use it ?

Entity relations mapping solution architecture



NATURAL INTELLIGENCE

Chapter 3

What did we learn ???

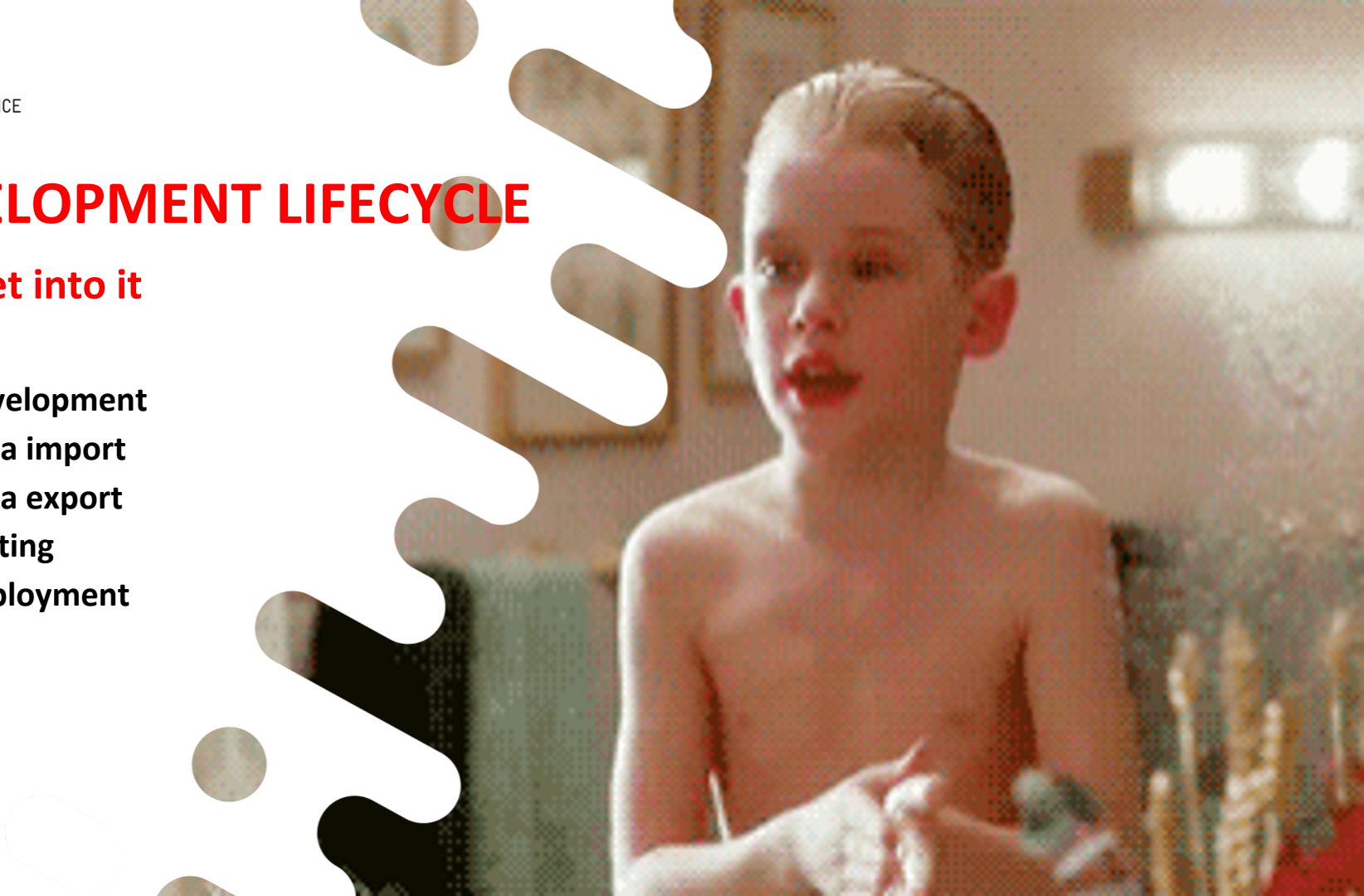
What do you need to know ?

NATURAL INTELLIGENCE

DEVELOPMENT LIFECYCLE

Let’s get into it

- Development
- Data import
- Data export
- Testing
- Deployment



NATURAL INTELLIGENCE

Mapping Service

- NodeJS MicroServices Framework (nestjs.com)
- Extensible
- Promote Design patterns
- Reusable services
- Lots of integrations
- Typescript enabled

- KafkaJS integration (already implemented – we extended it a bit)
- AWS Neptune repository service (we wrote it internally)

NATURAL INTELLIGENCE

NestJS + Kafka

npm i -g @nestjs/cli

nest new project-name

- **Use @EventPattern to handle Kafka events**

```
@EventPattern(topicName)
async handleEntityChangedEvent(payload: EntityRelationPayloadDto) {
  this.logger.info(`Event: ${topicName}, Payload: ${JSON.stringify(payload)}`);
  return this.statsDService.timer(
    () => this.handleEntityChanged(payload).catch((err) => this.getEntityProcessError(err, payload)),
    'handleKafkaEntityChanged_execution_time'
  )();
}
```

NATURAL INTELLIGENCE

Javascript + Gremlin


```
const {
  t: { id },
} = gremlin.process;
const {
  cardinality: { single },
} = gremlin.process;

const __ = gremlin.process.statics;
const __operators = gremlin.process.P;
```

1

- Use retry mechanism
- Use direct references
- Use UpSert pattern
- Add singularity

2 3 4 5

```
@Retryable({
  maxAttempts: RETRY_MAX_ATTEMPTS,
  backOffPolicy: BackOffPolicy.ExponentialBackOffPolicy,
  backOff: RETRY_BACKOFF,
  exponentialOption: { maxInterval: RETRY_EXP_MAX_INTERVAL, multiplier: RETRY_EXP_MULTIPLIER },
  doRetry: (e: Error) =>
    e.message.includes('ConcurrentModificationException') || e.message.includes('ReadOnlyViolationException'),
})
async addEntity(entity: Entity) {
  return this.graphs.writer
    .V(entity.nodeId)
    .fold()
    .coalesce(
      __.unfold()
        .property(single, ENTITY_FIELDS.LAST_UPDATE, new Date().toUTCString())
        .property(single, ENTITY_FIELDS.ATTRIBUTES, JSON.stringify(entity.filteredAttributes)),
      __.addV(entity.entityType)
        .property(id, entity.nodeId)
        .property(ENTITY_FIELDS.NODE_ID, entity.nodeId)
        .property(ENTITY_FIELDS.ENTITY_ID, entity.entityId)
        .property(ENTITY_FIELDS.ENTITY_TYPE, entity.entityType)
        .property(ENTITY_FIELDS.VAR_ID, entity.varId)
        .property(ENTITY_FIELDS.VAR_TYPE, entity.varType)
        .property(single, ENTITY_FIELDS.IS_PUBLISHED, !!entity.isPublished)
        .property(single, ENTITY_FIELDS.LAST_UPDATE, new Date().toUTCString())
        .property(single, ENTITY_FIELDS.VERSION, this.VERTEX_VERSION)
        .property(single, ENTITY_FIELDS.ATTRIBUTES, JSON.stringify(entity.filteredAttributes))
    )
    .id()
    .next();
}
```

NATURAL INTELLIGENCE

Cost explorer

AWS
Neptune
Cloud

Production / Staging

- Price per instance / instance type

?
Local

Development / Testing

NATURAL INTELLIGENCE

Cost explorer

AWS
Neptune
Cloud

Production / Staging

- Price per instance / instance type

- In memory
- Large database is an issue in load time and memory consumption

Tinkerpop
Local
JanusGraph

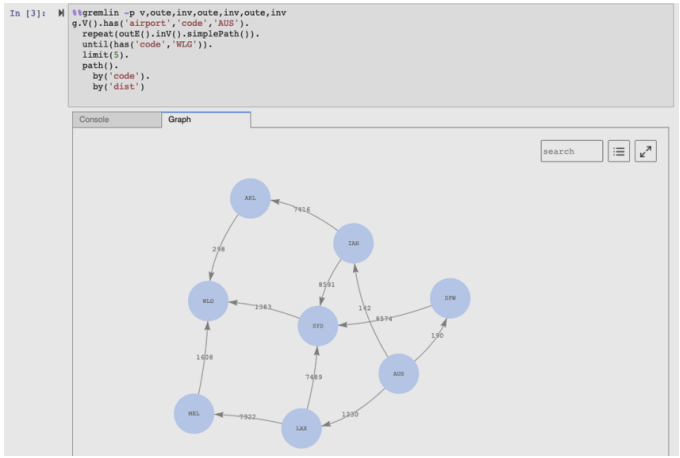
Development / Testing

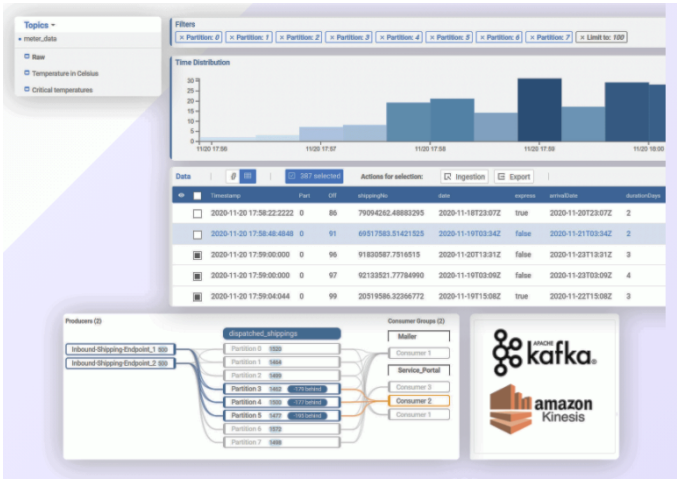
tinkerpop.apache.org / janusgraph.org

NATURAL INTELLIGENCE

Dev/Test - Tinkerpop + Kafka

- Use dockers to host Tinkepop and Kafka locally
- Configure Tinkerpop server yml to use persistency / Use JanusGraph
- Add all dockers to the same network
- Add dependencies between the dockers
- Add Xeotek KaDeckWeb docker – it will assist you in dev / test / prod with kafka
- Add AWS graph notebook docker - github.com/BarrMan/graph-notebook-docker





NATURAL INTELLIGENCE

Import / Export

- AWS Bulk data loader
- Write your own code / scripts
  Java/NodeJS/Python

- Neptune-Export.jar
  github.com/awslabs/amazon-neptune-tools

Production / Staging

- Write your own code / scripts
  Java/NodeJS/Python

- Configure persistency

Development / Testing

NATURAL INTELLIGENCE

Export

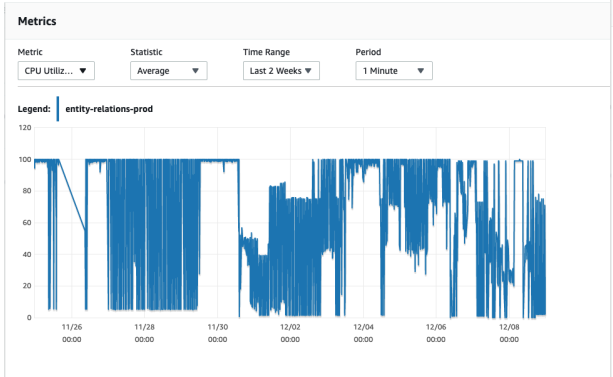
- **Neptune-Export.jar**
  github.com/awslabs/amazon-neptune-tools

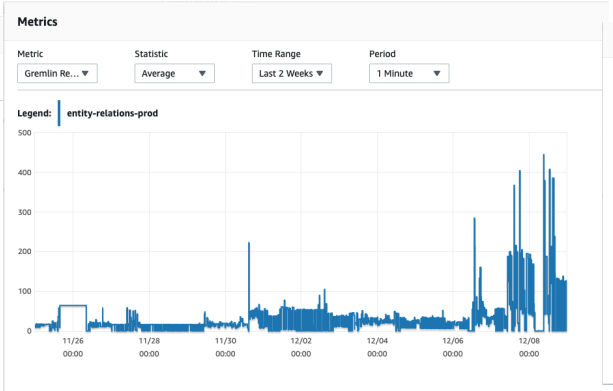
- **java -jar neptune-export.jar export-pg -e db-name.neptune.amazonaws.com**
  -d ./Neptune ← Dump to local folder
  --use-ssl
  --serializer GRAPHBINARY_V1D0 ← Compatibility with tinkerpop
  --clone-cluster ← Create new export cluster
  --clone-cluster-instance-type db.r5.larg

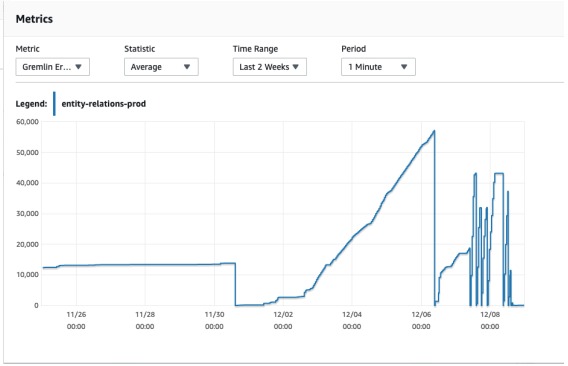
NATURAL INTELLIGENCE

Production

- Use AWS support if you need – they are great !
- We had performance issue with UpSert
  - It took 4.5 sec to complete
  - we got hotfix from AWS Support and it had been fixed in a later version
- We had high CPU for low numbers of gremlin transactions even on huge instances
  - Use direct references to nodes / edges when you can
- Do not forget to add retry on concurrency failures
- Monitor your queries, server load and gremlin errors







NATURAL INTELLIGENCE

Summary

- Don’t be afraid from using Graph Databases
- Use them only when / where it makes sense
  (3 – 4 joins | traversal path)
- Working with NodeJS ? - Give NestJS a try
- Event sourcing is great for system decoupling
  Kafka is awesome !

More info about this topic will be posted at:
medium.com/natural-intelligence

NATURAL INTELLIGENCE | Top10.com

thank you

Ohad Israeli
Platform Architect
Natural Intelligence

Ohad.israeli@naturalint.com

https://github.com/israelio/graphdb-intro