Low-cost Open Data As-a-Service

Marin Dimitrov, Alex Simov, Yavor Petkov

May 31st, 2015
• Use cases & requirements
• Cloud architecture for a RDF DBaaS
• Lessons learned
Use Cases & Requirements
Why an RDF DBaaS?

Grafter

- RDF UTENSIL

Open Data Portal

- Transform **tabular data** into RDF
- Publish (Linked) **data services**, instead of static datasets
  - Lower-cost & easier data publishing process

Grafterizer

- Edit prefixes...
- Create custom function...
- Preview Grafter pipeline...
- Modify RDF mapping...
- Register pipeline and mapping...

RDF DBaaS

Low-cost Open Data as-a-Service / SemDev’2015
Why an RDF DBaaS?

- Transform **textual data** into RDF
- Linked **data services**
- Low-cost & easy to use

Knowledge Graphs

RDF Data

Text Analytics

- Career
- Person
- Organization
- Location

Deputy Defence Minister Kostas Isichos told the BBC northern Europe must do more to rescue and shelter migrants.

Low-cost Open Data as-a-Service / SemDev’2015

May 2015

#5
DBaaS requirements

- Elastic
  - dynamically adapt to growing data & query volumes
- High availability & resilience
  - no SPFs, “graceful degradation” upon failures
- Cost efficient
- Host a large number of data services (databases)
  - But probably of low/moderate data & query volume
- Isolation of the multi-tenant databases

Not easy to achieve all three!
Cloud Architecture
• AWS based
  – Network storage, compute & autoscaling, load balancing, integration services, ...
• Ontotext GraphDB as the RDF DB engine
  – OpenRDF REST API
• Docker for containerisation
• An RDF DBaaS is...
  – A GraphDB instance...
  – Running within a Docker container...
  – Storing its data on a private NAS volume
DBaaS architecture on AWS

Elasticity vs
High Availability vs
Cost Efficiency
Dealing with failures

Low-cost Open Data as-a-Service / SemDev’2015

CSP responsibility
• Elastic
  – Routing nodes, data nodes + NAS storage grow as usage grows

• High availability & resilience
  – Strategies for dealing with failures in data, routing, Coordinator nodes
  – Planned: multi-DC deployment with replication

• Cost efficient
  – Cloud native architecture -> cost savings
  – Multi-tenant model -> cost savings
  – Elastic: return underutilised or unused resources back to CSP
Lessons Learned
• Cloud-native architecture
  – Improved scalability, reliability, cost savings

• A microservice architecture will continuously evolve

• Assume that failures will happen on all levels
  – Design for “graceful degradation”

• A good DevOps process is essential
Discussion
• Use it for free!
  – http://s4.ontotext.com (available NOW)
  – http://dapaas.eu (end of June)
• Send us questions, comments, criticism, suggestions for improvements, ...
• Are you measuring the TCO of your on-premise RDF databases?
  – Important for many Open Data scenarios

• What is your #1 concern for using an RDF DBaaS

• Do you have use cases where your productivity will increase by using an RDF DBaaS
  – Experiment & prototype faster; focus on building apps, don’t worry about infrastructure; provision new DBs instantly...
  – Real world example: training courses by Ontotext switching from local deployments to the RDF DBaaS
Thank you!