LDBC SPB: Semantic Publishing Benchmark
Outline

- Introduction to LDBC SPB
- Workloads
- Software
- Sample results
- Next steps
**BBC’s DSP Approach**

- The SPB was inspired by the Dynamic Semantic Publishing
  - First applied at BBC’s FIFA Worldcup 2010 website
  - Next at website of BBC and the official one for the 2012 Olympics
  - Now continuously used at BBC Sport and by many others

- The BBC’s primary use case: rich, deep, dynamic websites
  - Text-mining automatically annotates articles with entities
  - Editor curates the metadata before storing it in a triplestore
  - Thematic web pages are generated on-the-fly through SPARQL
“The goal is to be able to more easily and accurately aggregate content, find it and share it across many sources. From these simple relationships and building blocks you can dynamically build up incredibly rich sites and navigation on any platform.”

John O’Donovan
Chief Technical Architect, BBC
Publishing & Media Domain

- Constantly generating new content
- Constantly updating existing content
- Constantly consuming content

Semantic technologies in the publication pipeline

- Annotation of content; often through text mining
  - Also know as “metadata enrichment” or just “tagging”
- Content multi-purposing
  - Multiple media (press/Tv, web, mobile), multiple products (specific channels, data feeds, etc.)
LDBC SPB

LDBC-Semantic Publishing Benchmark [1]

• A benchmark for RDF Databases
• Simulates a media organization which maintains a catalogue of meta-data for its assets
• Simulates operations performed by real users:
  • consumption of meta-data
  • management of meta-data
• Measures the performance of both types of operations

LDBC SPB Requirements

- Support for quadruples; triples plus context/named graphs
  - Support for TRIG or NQ RDF serialization syntax
- SPARQL Query 1.1 support
- SPARQL Update 1.1 support
  - READ COMMITTED transaction isolation
  - Consistent handling of updates
- SPARQL Protocol 1.1 (known as “SPARQL End-point”)
- Inference support – RDFS semantics is sufficient
Outline

- Introduction to LDBC-SPB
- Workloads
- Software
- Sample results
- Next steps
WORKLOADS

Editorial agents
• Simulate the work performed by journalists or editors with the system
  • E.g. enriching journalistic assets with meta-data: description, creation date, location etc.
• Run simultaneously
• Provide a constant stream of update operations
• Editorial operations:
  • INSERT, UPDATE, DELETE
WORKLOADS

Aggregation agents
  • Simulate the interactions of end-users or semi-automated tools with the system
    • E.g. queries generated by an application that dynamically generates web pages for wide range of topics (teams, players, events, etc.) at BBC Sport website
  • Run simultaneously
  • Provide a constant workload of queries from two available query mixes:
    • Basic query mix of 9 queries
    • Advanced query mix of 25 queries
WORKLOADS

- Basic (interactive) query mix (9 Queries):
  - Search queries
  - Full-text search queries
  - Aggregation queries
  - Geospatial queries

- Advanced (interactive + analytical) query mixes (25 Queries):
  - Analytical queries
  - Faceted search queries
  - Drill-down queries
Outline

- Introduction to LDBC SPB
- Workloads
- Software
- Sample results
- Next steps
DATA GENERATOR

- Real ontologies provided by the BBC
- Real reference datasets provided by the BBC, DBpedia and Geonames
- Parallel data generation
- Deterministic
Scalable
• Scales to billions of triples

Generated datasets simulate the activity of a publishing organization for a period of time
• Media assets are enriched by metadata called ‘Creative Works’
The SPB test driver:
- Open Source
- Available on GitHub: https://github.com/ldbc/ldbc_spb_bm

Runs Editorial and Aggregation agents simultaneously
- Parallel execution
- Provide a steady update stream during query workload

Validates Results
- Query results
- Update operations; batching is not allowed

Gathers and reports performance metrics
Outline

- Introduction to LDBC SPB
- Workloads
- Software
- Sample results
- Next steps
## SPB Sample Results

<table>
<thead>
<tr>
<th>Engine</th>
<th>Variant</th>
<th>Scale</th>
<th>R/W Agents</th>
<th>Hardware</th>
<th>Load time (sec)</th>
<th>Reads /sec</th>
<th>Updates /sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>GraphDB SE 6.0</td>
<td>Inter. Basic</td>
<td>50M</td>
<td>14/2</td>
<td>2xXeon, 256GB, SSD</td>
<td>2 045</td>
<td>32</td>
<td>12</td>
</tr>
<tr>
<td>GraphDB SE 6.0</td>
<td>Inter. Basic</td>
<td>50M</td>
<td>14/2</td>
<td>AWS c3.4xlarge (16GB, SSD)</td>
<td>2 045</td>
<td>27</td>
<td>11</td>
</tr>
<tr>
<td>GraphDB SE 6.1</td>
<td>Inter. Basic</td>
<td>50M</td>
<td>14/2</td>
<td>2xXeon, 256GB, SSD</td>
<td>2 045</td>
<td>34</td>
<td>17</td>
</tr>
<tr>
<td>GraphDB SE 6.0</td>
<td>Inter. Basic</td>
<td>1B</td>
<td>14/2</td>
<td>2xXeon, 256GB, SSD</td>
<td>41 400</td>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>GraphDB SE 6.1</td>
<td>Inter. Basic</td>
<td>1B</td>
<td>14/2</td>
<td>2xXeon, 256GB, SSD</td>
<td>41 400</td>
<td>10</td>
<td>6</td>
</tr>
</tbody>
</table>
Outline

- Introduction to LDBC SPB
- Workloads
- Software
- Sample results
- Next steps
Next Steps

- Generate rich set of public results
  - Ranging across different scales, variants, engines, hardware, etc.
- Richer reference knowledge; more links between entities
- Queries that explore links between entities
- Experiment with simple OWL inference
Next Steps

- Extension/variant for testing enterprise features
  - Online backup
  - Various **failover scenarios**, e.g. seamless failover, hardware or network failures
  - Various **stress tests**, e.g. reaction of the cluster to a “Disk Full” event
  - Various **workload tests**, i.e. failover and stress tests passing under ongoing LDBC-SPB interactive workload
Questions?

Thank You