Enterprise Big Data Platforms

+ Big Data research @ Roma Tre

Antonio Maccioni
maccioni@dia.uniroma3.it

19 April 2017
Outline

- Polystores
  - QUEPA project

- Data Lakes
  - KAYAK project
No one size fits all
Polyglot Persistence

02. POLYGLOT PERSISTENCE: How many persistent storage models do your applications typically use?
Typical polyglot persistence scenario
Typical polyglot persistence scenario

What if I need data spread over several sources?
Integration of different database systems

- Tightly-coupled architectures
  - Multi-model: ArangoDB, OrientDB,
  - BDMS: Apache AsterixDB

- Loosely-coupled integration (Polystores):
  - Uniform interfaces
    - *query language*: UnQL
    - *primitives*: ODBC
    - *model*: Apache Metamodel
  - Federated databases
    - *data mediators*: R*, Ingres*, Garlic
    - *islands of information*: BigDAWG
Multi-model: ArangoDB

**KEY-VALUE**

- `pID` -> `x34`
- `name` -> `John`
- `surname` -> `Doe`

**DOCUMENT**

```
{ 
  "pID" : "x34",
  "name" : "John",
  "surname" : "Doe"
}
```

**GRAPHS**

- `pID : x34`
- `name : John`
- `surname : Doe`

**AQL**

```sql
FOR meetup IN meetups
  FILTER 'NOSQL' IN meetup.topics
  FOR city IN OUTBOUND meetup held_in
    FOR programmer IN INBOUND city lives_in
      FILTER programmer.notify
      FOR cname IN city_names
        FILTER cname.city == city._key AND cname.lang == programmer.lang
        INSERT { email: programmer.email, meetup: meetup._key, city: cname.name } INTO invitations
```
BDMS: Apache AsterixDB

Diagram showing the architecture of BDMS, with categories such as Data Model, Metadata, Functions, Ingestion, Transaction, Replication, Language, and REST API. Below the categories, there is a section labeled "General-purpose Query Optimizer" with further details on Operator Library (join, sort, group-by, etc.), Storage Library (LSM B-Tree, R-Tree, etc.), Connector Library (m-to-n, broadcast, etc.), and HDFS Utilities. At the bottom, it states "General-purpose Distributed DAG Execution Engine."
A polystore architecture is a loosely coupled integration of heterogeneous data sources that allows the direct access, with the local language, to each specific storage engine to exploit its distinctive features.
Apache Metamodel

“a uniform connector and query API to many very different datastore types”

```java
DataContext dataContext = DataContextFactory.create[TypeOfDatastore](...);
DataSet dataSet = dataContext.query()
  .from("libraries")
  .select("name")
  .where("language").eq("Java")
  .and("enhances_data_access").eq(true)
  .execute();
```
BigDAWG

- **Island of informations**: a collection of storage engines accessed with a single query language
- Support conjunct analytics
  - cross-db joins based on copies between back ends (cast)

[The BigDAWG Polystore System, Duggan et al., Sigmod Record 2015]
A user of a database wants to access the polystore but she doesn’t know how to query other databases and, sometimes, does not even know their existence.
Augmentation is a new construct for data access automatic enrichment of data extracted from one database with data stored elsewhere.

SELECT *  
FROM inventory  
WHERE name like 'wish%'

[QUerying and Exploring a Polystore by Augmentation, Maccioni, Basili, Torlone, Sigmod 2016]
Query Augmentation

```
SELECT *
FROM inventory
WHERE name like '%wish%'

< a32, Cure, Wish >
```
Query Augmentation

```
SELECT *
FROM inventory
WHERE name like '%wish%'
⇒
< a32, Cure, Wish > ⇒ (discounts: 40%)
⇒
(catalogue: { title: Wish,
artist id: a1,
artist: The Cure,
year: 1992,
... })
```
Architecture of QUEPA

1. USER INTERFACE
   - $Q^s(n)$

2. COLLECTOR
   - $Q^s$

3. VALIDATOR
   - $\alpha^n$

4. AUGMENTER
   - $\overline{Q}^s$

5. DBMS CONNECTORS
   - similar-items, catalogue, transactions, discount

6. \[ A^+ INDEX \]
   - $\overline{Q}^s$
   - $Q^i$
   - $\alpha$
   - $Q^s$

   - v1: catalogue.customers.c1
   - v2: transactions.inventory.a42
   - v3: transactions.sale-details.i1
   - v4: transactions.sale-details.i4
   - v5: transactions.sales.s8
   - v6: similar-items.ties.n4
   - v7: catalogue.albums.d1
   - v8: discount.drop.k1:cur:ish
   - v9: transactions.inventory.a32
Optimizations in QUEPA

- **Network-based** optimization:
  - Batch Augmenter;

- **CPU-based** optimization:
  - Inner Augmenter;
  - Outer Augmenter;
  - Outer+Inner Augmenter;
  - Outer+Batch Augmenter;

- **Memory-based** optimization:
  - Caching objects from Polystore.
Batch Augmentation

SEQUENTIAL

BATCH
CPU-based optimization

Polystores are inherently distributed
CPU-based optimization

INNER

OUTER-INNER

BATCH

OUTER

OUTER-INNER

OUTER-BATCH
Query Optimization

- **Adaptive Augmenter**
  - Rule-based optimizer for query augmentation;
  - Based on **machine learning**
    - rules are created with 4 decision/regression trees
    - training based on logs and other parameters (e.g., polystore RTT)
Benchmarking vs COMPETITORS

SCALABILITY

vs COMPETITORS
Augmented Exploration

- Provides an **interactive** way to access a polystore
- It is the **guided expansion** of the result of a local query
- The user finds her way by just **clicking** on the links
  - as soon as they are made available
  - similarly to Web surfing

```
< s8, John Doe, 20.0 > on-click ( { id: c1, name: John, surname: Doe, city: NYC, ... } )
```
How to rank polystore objects?
Diversity-Driven PageRank

\[ DDPR(v) = (1-d) + d \sum_{u \in I_v} DDPR(u) \times \text{diversification weight}(u,v) \]
Dynamic adjustment of ranking

Loop avoidance

Detour
"A data lake is a method of storing data within a system or repository, in its natural format, that facilitates the collocation of data in various schemata and structural forms, usually object blobs or files."

-- Wikipedia

“The idea of data lake is to have a single store of all data in the enterprise ranging from raw data ... to transformed data which is used for various tasks including reporting, visualization, analytics and machine learning.”

-- Wikipedia
Data Lake vs Data Warehouse

Data warehouse is **schema-on-write**

Data lake is **schema-on-read**

“In schema on read, data is applied to a plan or schema as it is pulled out of a stored location, rather than as it goes in”

-- Techopedia
“A logical data warehouse is a data system that follows the ideas of traditional EDW and includes, in addition to one (or more) core DWs, data from external sources”
Data preparation

“Data preparation is the process of gathering, combining, structuring and organizing data so it can be analyzed as part of business intelligence and business analytics programs. The components of data preparation include data discovery, profiling, cleansing, validation and transformation”
Data preparation

What data scientists spend the most time doing

- Building training sets: 3%
- Cleaning and organizing data: 60%
- Collecting data sets: 19%
- Mining data for patterns: 9%
- Refining algorithms: 4%
- Other: 5%

What's the least enjoyable part of data science?

- Building training sets: 10%
- Cleaning and organizing data: 57%
- Collecting data sets: 21%
- Mining data for patterns: 3%
- Refining algorithms: 4%
- Other: 5%

https://whatsthebigdata.com/2016/05/01/data-scientists-spend-most-of-their-time-cleaning-data/
Tools supporting data preparation

- **Data/Metadata Catalogs**
  - CKAN, GOODS, Data Ground, Apache Atlas, DataHub
  - *Help fishing a dataset, not preparing analysis*

- **Data Profilers**
  - Pandas, Metanome, Constance
  - *Involve hard-to-scale algorithms*

- **Data Curators**
  - Data Civilizer
  - *Focus on data cleaning and data transformations*
Example of Metadata Catalog
A data management framework for a data lake with the aim of:

1) Reducing the time that a data scientist spends on data preparation

2) Guiding the data scientist to the insights
Tasks, Primitives and Pipelines

- **Tasks** are atomic operations executed by the system.

- **Primitives** are operations exposed to users:
  - asynchronous
  - synchronous

- **Pipelines** are compositions of primitives.
Time-to-action: time elapsing between the submission of the primitive and when the data scientist is able to take an informed decision on how to move forward to the next step of the pipeline.

We observe that:

- Time-to-action is often unnecessarily long
- approximate results are often enough informative to move forward in the data preparation pipeline
Tolerance

15%
Incremental Execution

- An **approximation** of the final result is called a **preview**
- A preview is associated to a **confidence**
  - the uncertainty on the correctness of the result
- KAYAK produces, incrementally, one or more previews for the primitive within the user’s tolerance
  - previews are produced with an increasing confidence
- We have identified two **strategies** for incremental execution:
  - **GREEDY**: produces as many previews as possible
  - **BEST-FIT**: produces the most accurate preview only
Aspects involved in KAYAK

- Data profiling
- Data cataloguing
- Strategies for task generation
- Task dependencies management
- Scheduling of tasks
- Query recommendation
- Ad-hoc Access Management

- More aspects to be plugged in (later):
  - cleansing
  - visualization
  - ...

References

- Polyglot persistence / Polystores
  - J. Duggan et al., The BigDAWG Polystore System, SIGMOD Record 2015
  - Y. Papakonstantinou, Polystore Query Rewriting: The Challenges of Variety, EDBT/ICDT Workshops 2016
  - Apache AsterixDB ([paper, tool](http://metamodel.apache.org/))
  - CloudMdsQL: ([paper, tool](http://metamodel.apache.org/))
  - RHEEM ([paper, tools](http://metamodel.apache.org/))
  - MuSQLe ([paper, tool](http://metamodel.apache.org/))
  - Myria ([paper, tool](http://metamodel.apache.org/))
  - ArangoDB ([https://www.arangodb.com/](https://www.arangodb.com/))
References

● Data Lakes
  ○ A. Halevy et al., Goods: Organizing Google's Datasets, Sigmod 2016
  ○ Z. Abedjan et al., Profiling relational data: a survey. VLDB J. 2015
  ○ D. Deng et al., The Data Civilizer System, CIDR 2017
  ○ Data Ground (paper, tool)
  ○ Apache Atlas (http://atlas.apache.org/)
  ○ DataHub (paper, tool)
  ○ Metanome (paper, tool)