Semantic Data

Chapter 9 : Data integration and ontology-based data access

Jean-Louis Binot

Sources and recommended readings

- The following paper is part of the course material as case study :
  - *Ontology Based Data Access in Statoil (Kharlamov et al. 2017).*
- Sources and useful additional readings :
  - *RDF Database Systems, (Curé and Blin 2015)* : comprehensive review of the state of the art on RDF stores.

  OBDA is a recent discipline led by a few scientific teams. Its presentation here is mostly based on :
  - ***Ontology-based Data Access: Theory and Practice,*** *(Xiao and Kontchakov)*, IJCAI 2018 tutorial.
  - *Query Answering and Rewriting in Ontology-Based Data Access, (Rosati)*, KR 2014 tutorial.
  - *Ontology-Based Data Access: From Theory to Practice (Calvanese 2012).*
- University courses having partially inspired ideas and examples for this chapter :
  - *Grundlagen von Ontologien und Datenbanken für Informationssysteme (CS5130), Özçep, Universität Lübeck.*
  - *Ontology languages (COMP321), Wolter, University of Liverpool.*

Agenda

Slides 17, 28, 34, 35, 58 are not in the material for the exam.

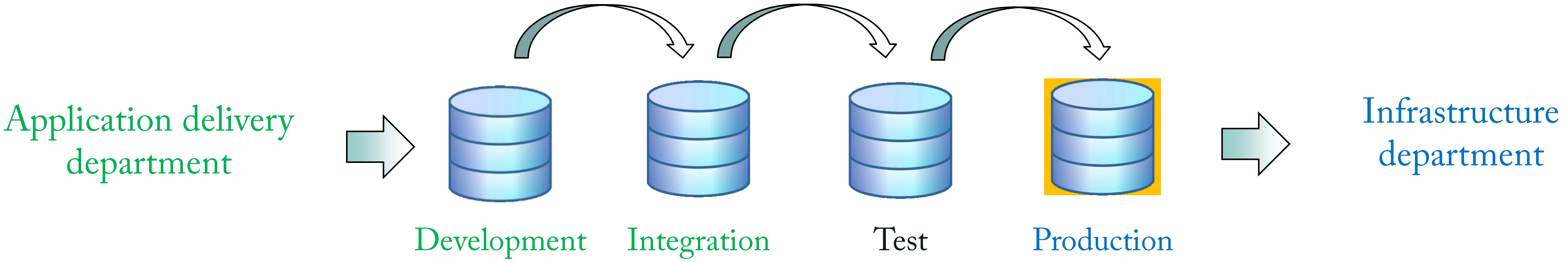
1. RDBMS challenges (why NoSQL)
2. Data integration challenges
3. RDF based data integration
4. Graph DBs and triple stores
5. Case Study in Oil & Gaz
6. Ontology based data access

Background questions

1. Why not use a relational database (why NoSQL) ?
2. To cope with change, why can't we just change the RDBMS schema ?
3. Why is data integration so difficult ?
4. What is a triple store – or a graph database ?

The IT environments

In a normal IT environment, Application Delivery is not allowed to push directly programs or data in production !



- **Development** : hosts the development of applications.
- **Integration** : hosts integration testing of all components of a solution.
- **Test** : hosts final tests of functional and non-functional aspects of applications. Ideally a mirror of production.
- **Production** : runs business applications, servers, databases. Must “keep the lights on”.

What do we find in production?

Production

- Hardware :
  - Servers, networking, storage.
- Software :
  - Business applications.
  - Databases (usually large, relational DBs).
  - Job scripts and other small programs.
  - IT management applications :

    schedulers, load balancers, monitoring, help desk, security systems…

  …
- Data.

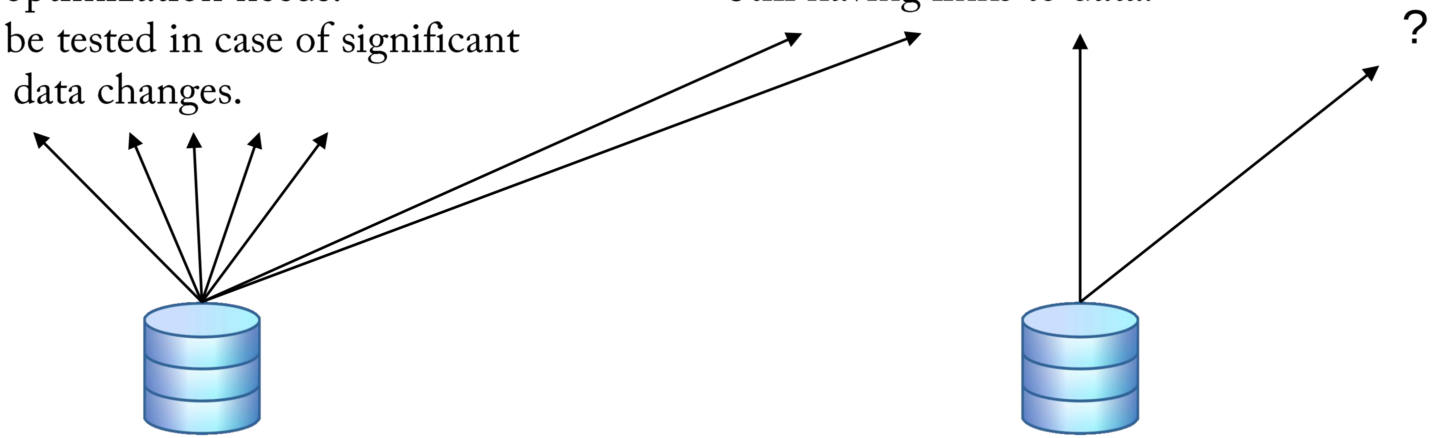
Operational complexity

Multiple live applications

Sharing the same RDBMS, but
different optimization needs.
All must be tested in case of significant
RDBMS data changes.

Multiple old applications

No longer used but not retired.
Still having links to data.



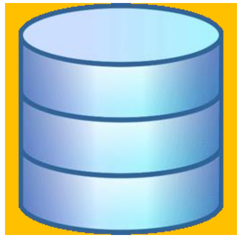
Live data

Dead data

Only accessed by applications no longer used.
What application is accessing them may be unknown.

- Potentially thousands of applications to consider.
- Cleaning that situation is very expensive and does not bring new business benefits.

Performance and velocity challenges in production



- ❑ Objective 1 : fast, reliable, integrated access to data.
- ❑ Objective 2 : fast velocity to cope with data changes.

Constant fight to optimize :

- ❑ Overall infrastructure performance.
- ❑ Database performance.
  - Optimized for transactional performance (OLTP).
  - Hardware / database / network setup and parameters.
  - Data schema, indexes.
  - Size of tables and queries.
  - Query complexity optimization…

Constraint : production must be on for business !

A reality check :

- ✗ SQLs may reach thousands of lines !
- ✗ Developing SQLs / interfaces may take weeks.
- ✗ Many SQL developers cannot optimize queries.
- ✗ Old data is locked in legacy systems.
- ✗ Dead data accumulate without archiving.
- ✗ Overgrown tables threaten database limits.
- ✗ Large changes only made during technical week-ends.

RDBMS challenges summary

- The RDBMS is the de facto standard for all large IT organizations :
  - Clear and precise formal models.
  - Mature technology with efficient optimizations (query execution planning …)
  - Rigorous management of transactions (Atomicity, Coherence, Isolation, Durability)
  - Capable to serve multiple applications.
- It has however several drawbacks :
  - Cost : maintaining stability and performance of a large RDBMS system is very costly.
  - Rigidity : significant data schema changes require heavy work and are only allowed at specific times.
  - Scaling up : big data now requires extremely large quantities of data (exabytes, zettabytes …).
  - Not very good at handling relationships between multiple data.
- Big data needs :
  - Huge amount of data, evolving data schema, capability to replicate in large parallel clusters …

  => Adoption of NoSQL models.

Agenda

1. RDBMS challenges (why NoSQL)
2. Data integration challenges
3. RDF based data integration
4. Graph DBs and triple stores
5. Case Study in Oil & Gaz
6. Ontology based data access

Data integration and system interoperability

Billions of dollars lost every year in integration and interoperability costs !

Many domains impacted : biomedical sciences, energy, engineering, aerospace…

- Genetics. "We have these giant piles of data and no way to connect them" said H. Steven Wiley, a biologist at the Pacific Northwest National Laboratory. "I'm sitting in front of a pile of data that we've been trying to analyze for the last year and a half." *(DNA Sequencing Caught in Deluge of Data, Pollack 2011).*

- Automotive. Concurrent design and engineering in the supply chain are vital ... However, these innovative … processes are hampered if product data cannot be exchanged seamlessly across the supply chain… imperfect interoperability costs the US automotive industry about $1 billion per year *(Brunnermeier and Martin 2002).*

- Construction. $15.8 billion in **annual** interoperability costs were quantified for the capital facilities (construction) industry in 2002 *(Gallaher et al. 2004).*

- Pharmacy. The increased generation of data in the R&D process has failed to generate the expected returns in terms of enhanced productivity and pipelines … The big business challenges … all rely heavily on integrating a broad range of information in a more meaningful way than the current industry norm *(Gardner 2005).*

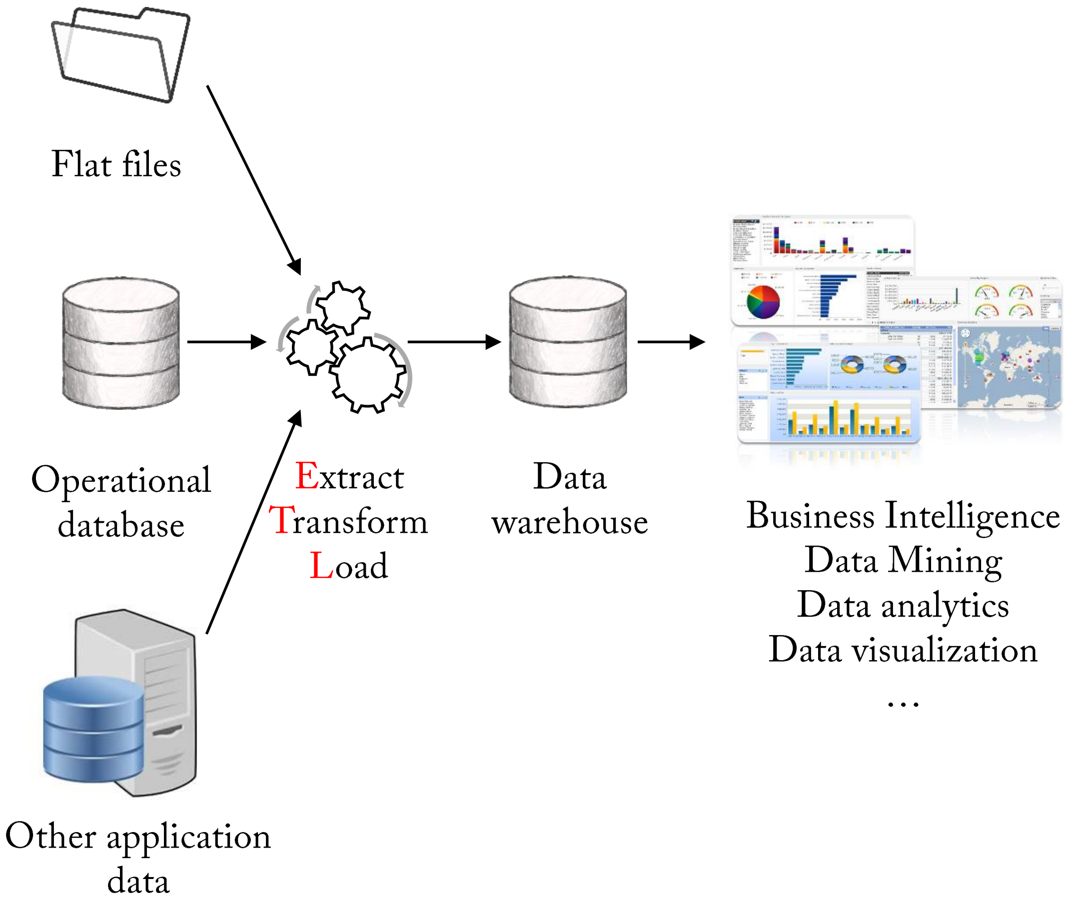
Data integration ?

Data integration (one definition among many) :

the process of combining data from different sources into a single, unified view. Integration begins with the ingestion process, and includes steps such as cleansing, ETL mapping, and transformation. Data integration ultimately enables analytics tools to produce effective, actionable business intelligence. (https://www.talend.com/resources/what-is-data-integration/).

- ETL (Extract Transform Load) ?
- Business intelligence ?

Is OLAP the solution for data integration ?



- OLAP: Online Analytical Processing.
  - Data is moved into a data warehouse through an Extract, Transform, Load (ETL) process.
  - Data organization in the data warehouse is optimized for analytics (e.g., data cubes).
- OLAP is a support for Business Intelligence.
  - Data analysis of business information (historical, current and predictive views of business operations, KPIs).
  - Normally performed by business users, with powerful tools.
- Another reality check :
  - ✗ ETL development takes time (days or weeks).
  - ✗ Business intelligence is not always a business priority (!).
  - ✗ OLAP encourages end user computing (data is sitting on a user PC, in Excel, somewhere…)

Integration challenges : data silos

(source of image https://www.protegrity.com/silos-causes-overcome/)

Data silo: situation where only one group in an organization can access specific data.

Causes ?

- Technical.
- Structural (organizational).
  - Each business unit wants its own data(bases).
  - Who is ready to pay for interfaces ?
  - Proliferation of projects and databases.
  - Broken end to end processes.
  - Manual re-encoding of data; low data quality.
- Cultural (knowledge is power).

Silo issues summary

- ✘ Silos defeat data integration and collaboration.
- ✘ When data is in silos, no one has the big picture.
- ✘ No single source of truth.
- ✘ Data quality issues.
- ✘ Huge interoperability costs (interfaces, manual re-encoding…).

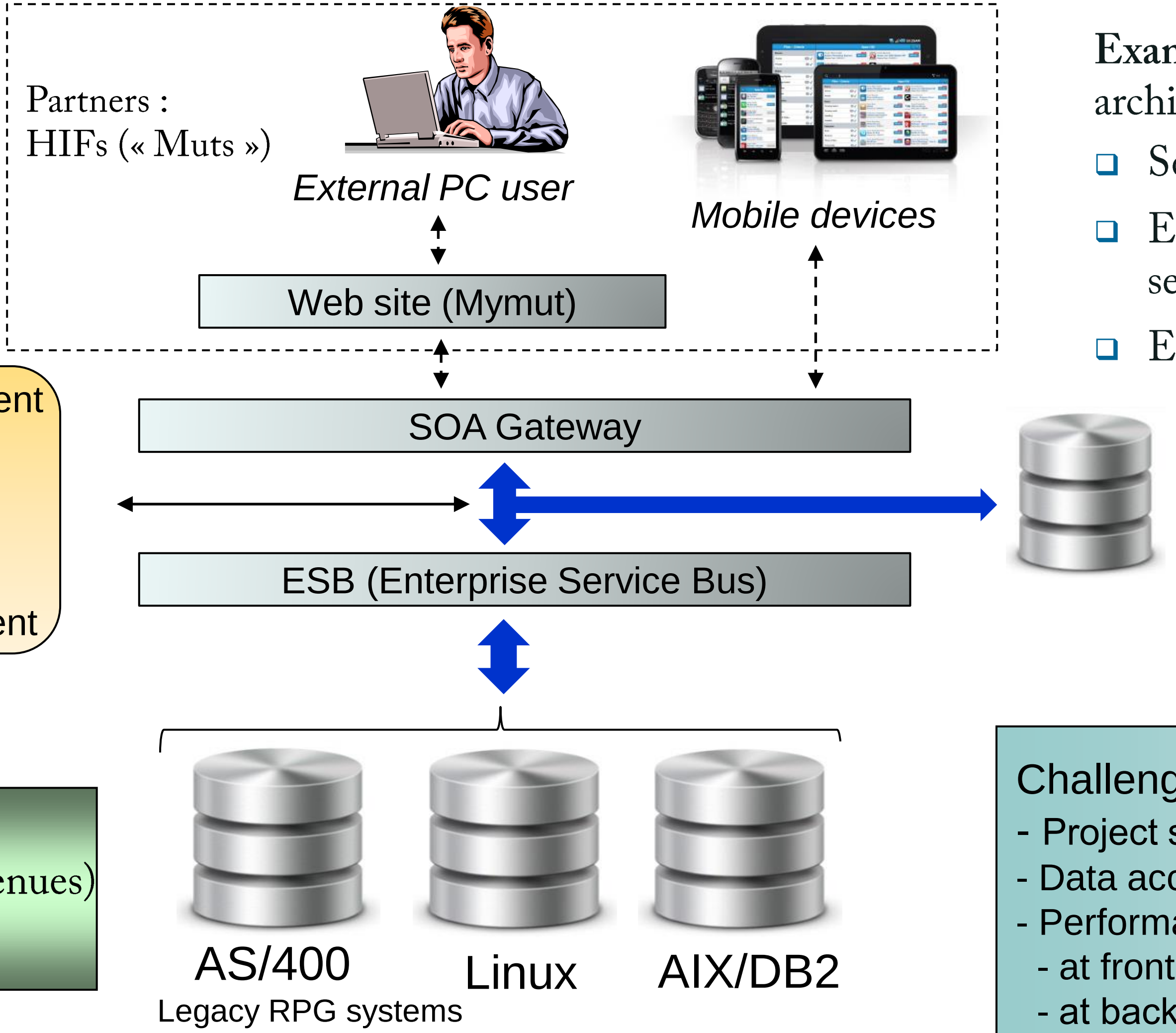
Agenda

1	RDBMS challenges (why NoSQL)
2	Data integration challenges
3	RDF based data integration
4	Graph DBs and triple stores
5	Case Study in Oil & Gaz
6	Ontology based data access

Data integration option 1 : SOA + ESB

Example : state of the art nationwide service architecture for health insurance (2016).

- Service oriented architecture.
- Enterprise service bus (XML, EDI, web services …).
- Encapsulation of legacy.



Data isolation layer (cache database)

Identity and access management
Monitoring and logging
Tracking and tracing
Statistics
Error mapping
Transactional state management

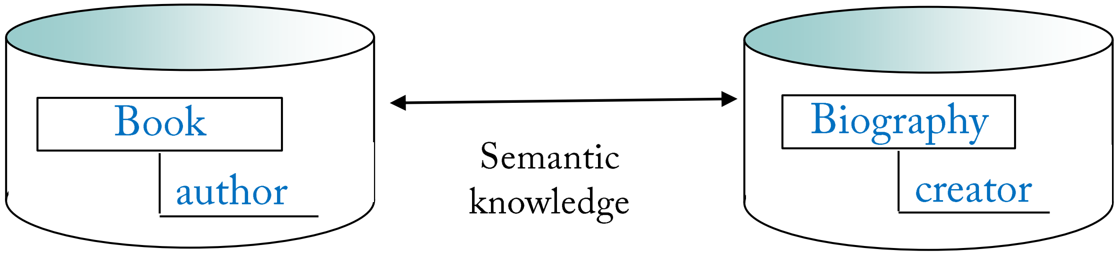
Benefits:
- increase time to market (revenues)
- reduce costs (IT)

Challenges
- Project size (>5M€; 5 “Muts”, several 100K users)
- Data accuracy and confidentiality
- Performance and availability (8000 t/m peak)
  - at front end (caching, optimization)
  - at backend (DB, ESB, network, server …)
- Data consistency between cache and backend

Data integration and system interoperability

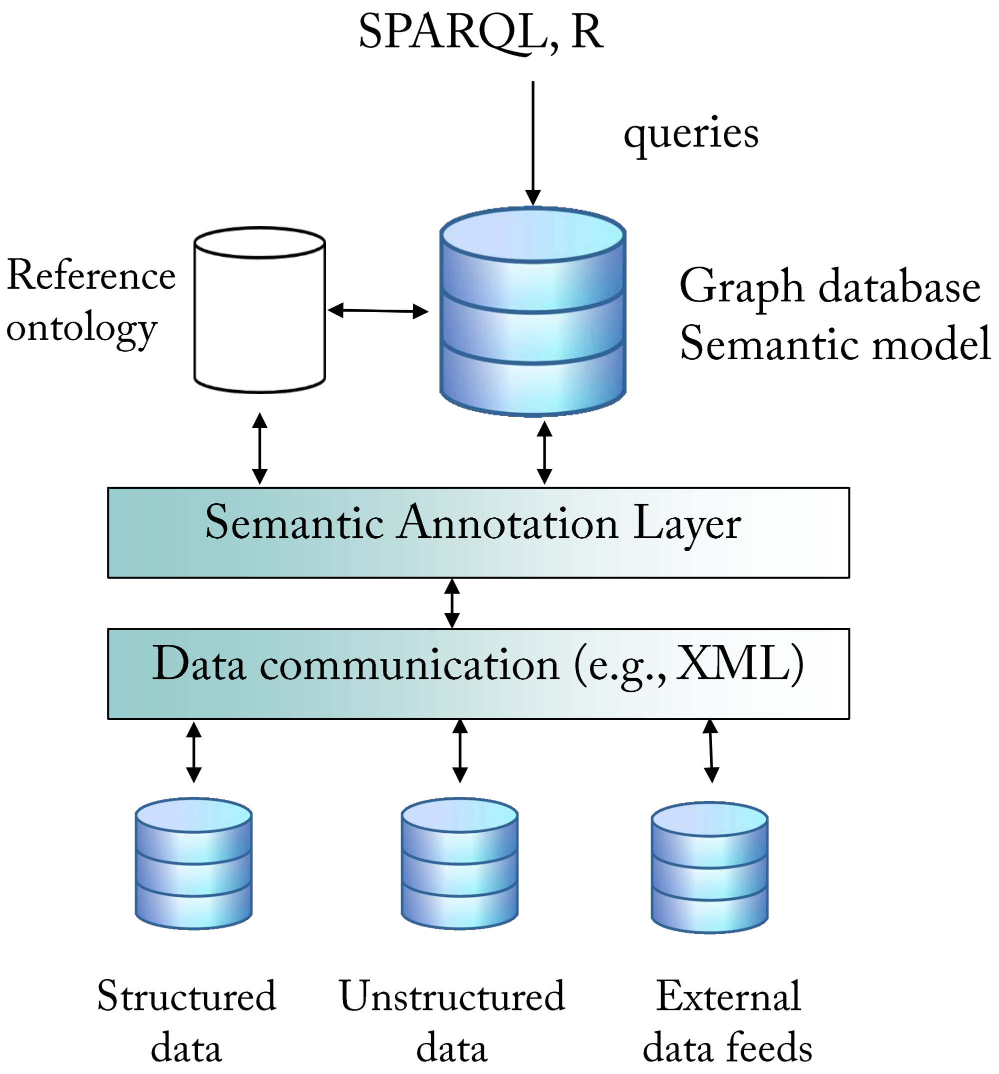
Three types of interoperability *(from the EU eHealth Governance Initiative (eHGI, 2012))*

- Technical interoperability : the ability of information and communication technology applications to accept data from each other and perform a given task in an appropriate manner without the need for extra operator intervention.
- Legal interoperability : refers to the environment of laws, policies, procedures and cooperation agreements needed to allow the seamless exchange of information.
- Semantic interoperability : refers to the ability to ensure that the precise meaning of exchanged information is unambiguously interpretable by any other system, service or user.



Biographies are books (but there are other books).
Authors are creators (but there are other creators).

Data integration option 2 : adding semantics



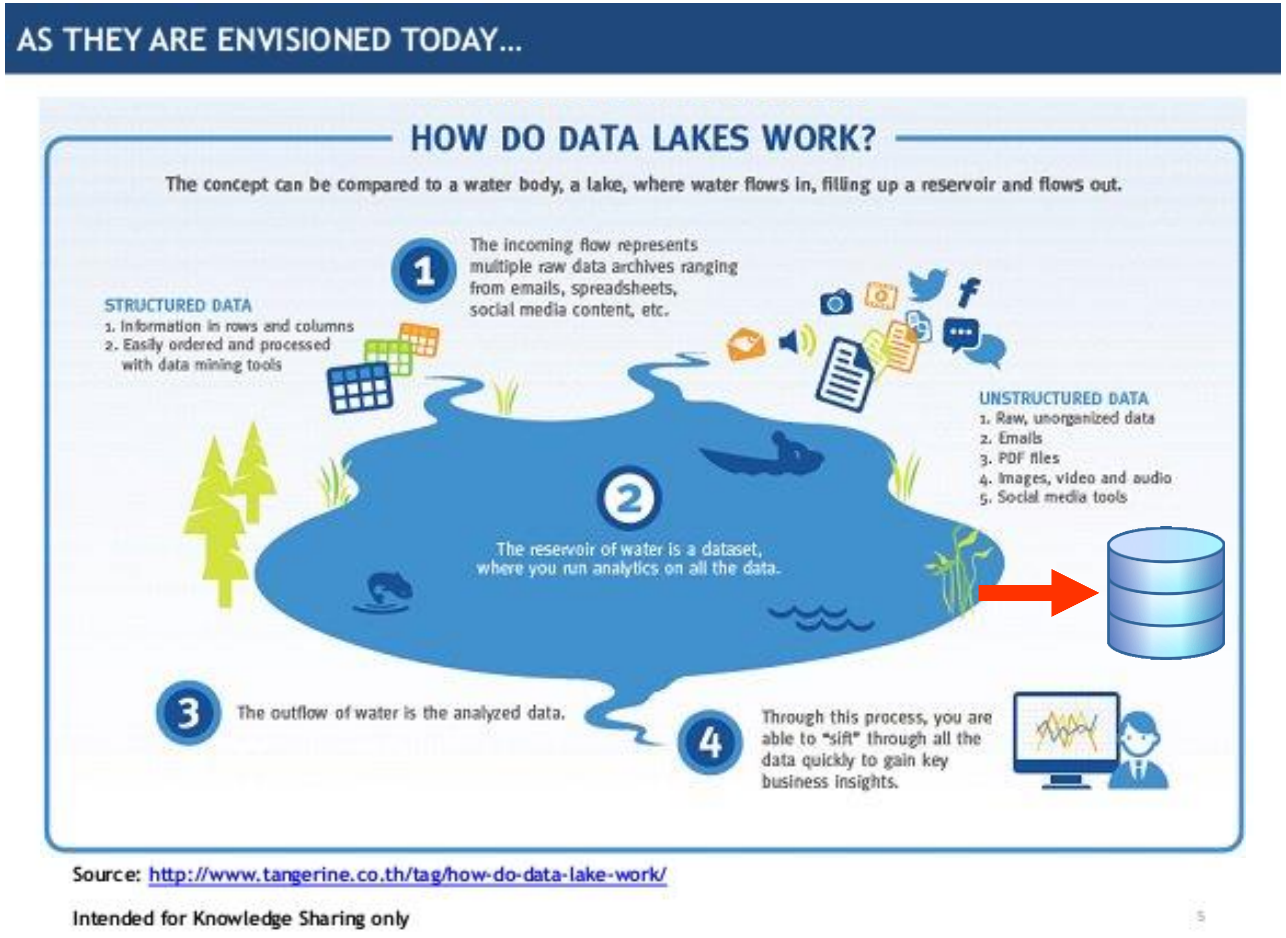
Adding a semantic layer can solve a number of data integration problems.

A configuration :

- Structured and unstructured data.
- Data marked with semantic annotations.
- Data imported into a graph database (schema easy to extend) with semantic linkage.
- Queries through semantic web standards (SPARQL).
- Interfaces to programs and analytical tools (R).

This is illustrated by the Norwegian oil & gas example.

Data integration option 3 : data lakes



Still needs semantics ! To be discussed in ch. 11

Main principles:

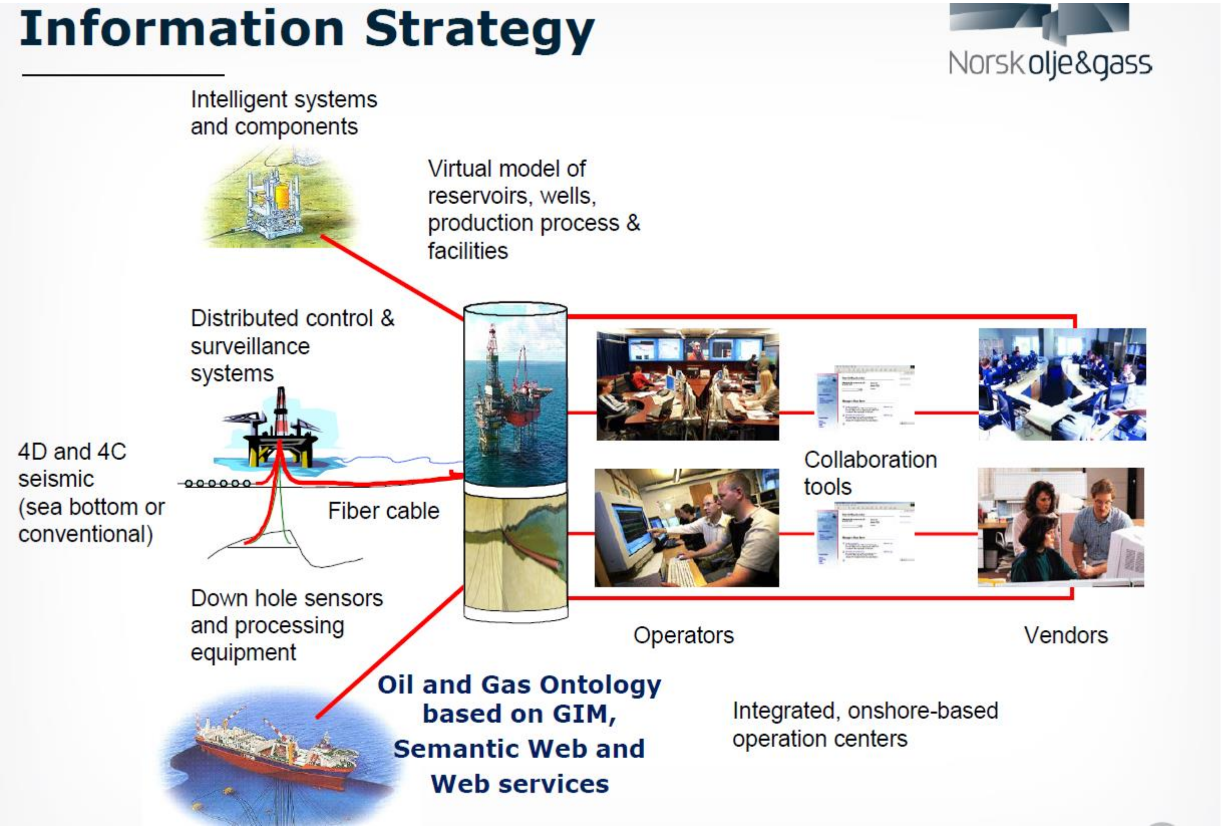
- Data is gathered from various sources
  - Both structured and unstructured.
- Distributed file system architecture.
  - Typically, Hadoop / Spark.
  - Cloud services: Amazon S3, Microsoft Azure…
- There is no effort to structure the data at the time of capture (schema on read approach):
  - Data is stored in its initial raw format.
  - Data consumers will set up their analysis applications to perform specific data exploration.
  - Less up-front costs, more flexibility, less optimization.
  - The data lake can feed a data warehouse.

Oil and Gaz data context

- Many actors (operators, vendors, authorities).
- Multiple applications (reporting, logistics, environmental data mgt …) evolving over time.
- Many sources of data (downhole sensors, surface facilities, logistics …).
- Data analysis is a bottleneck; reporting is expensive.
- 30-70% of time is spent looking for and assessing the quality of the data found.

Cf. a.o. *Keynote address to semantic web in Oil & Gaz workshop (Crompton 2008).*

A data integration example : EPIM

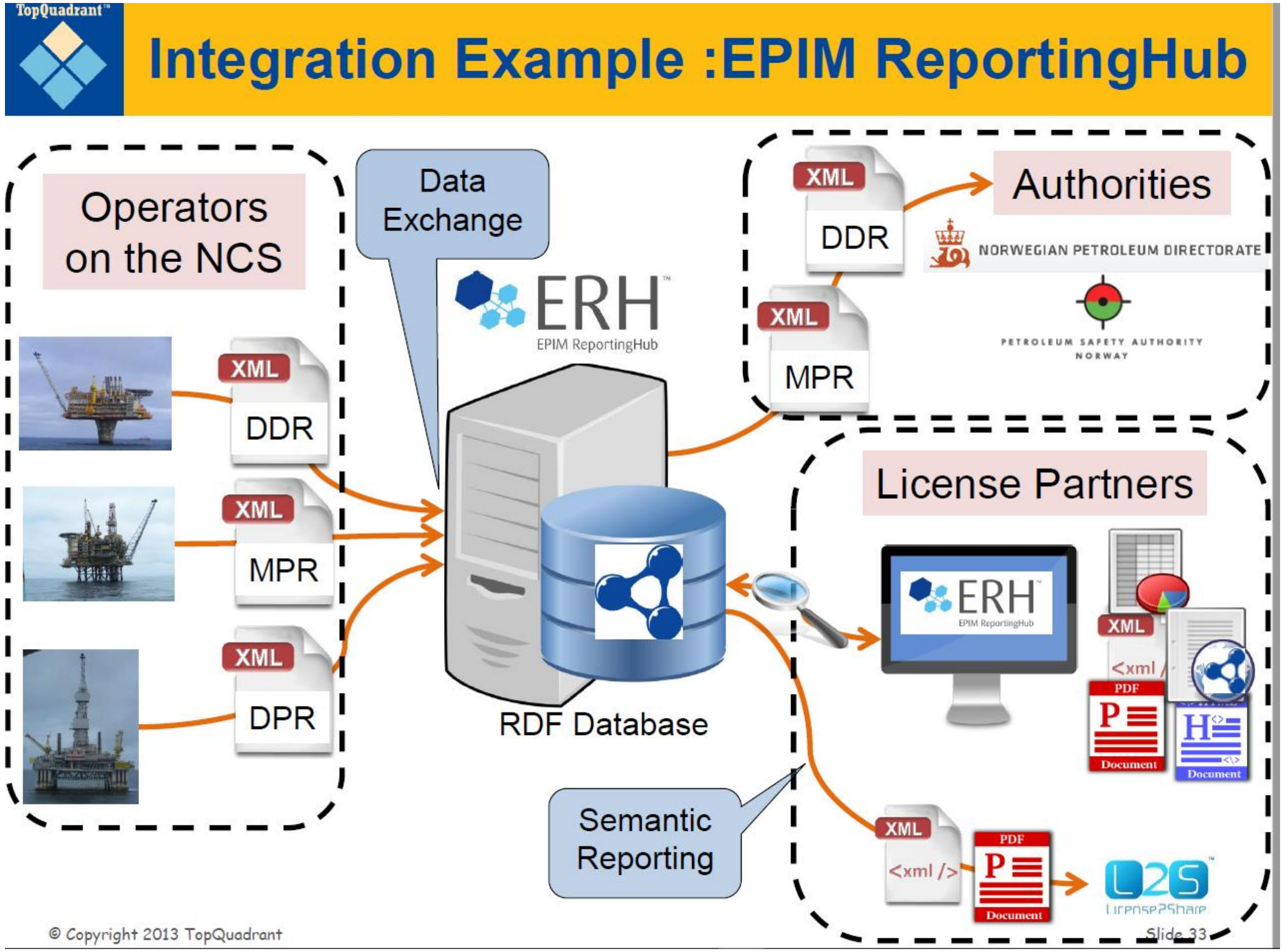


(Source for image : Langeland 2013).

- Realization reported in several sources :
  - Presentation by T. Langeland *(2013).*
  - Case study sheet *(TopQuadrant 2013).*
- Exploration and Production Information Management (EPIM) :

  Non-profit association of oil and gas operators and partners on the Norwegian Continental Shelf (now absorbed in the Norwegian Oil and Gas Association).

Solution : RDF database for data integration

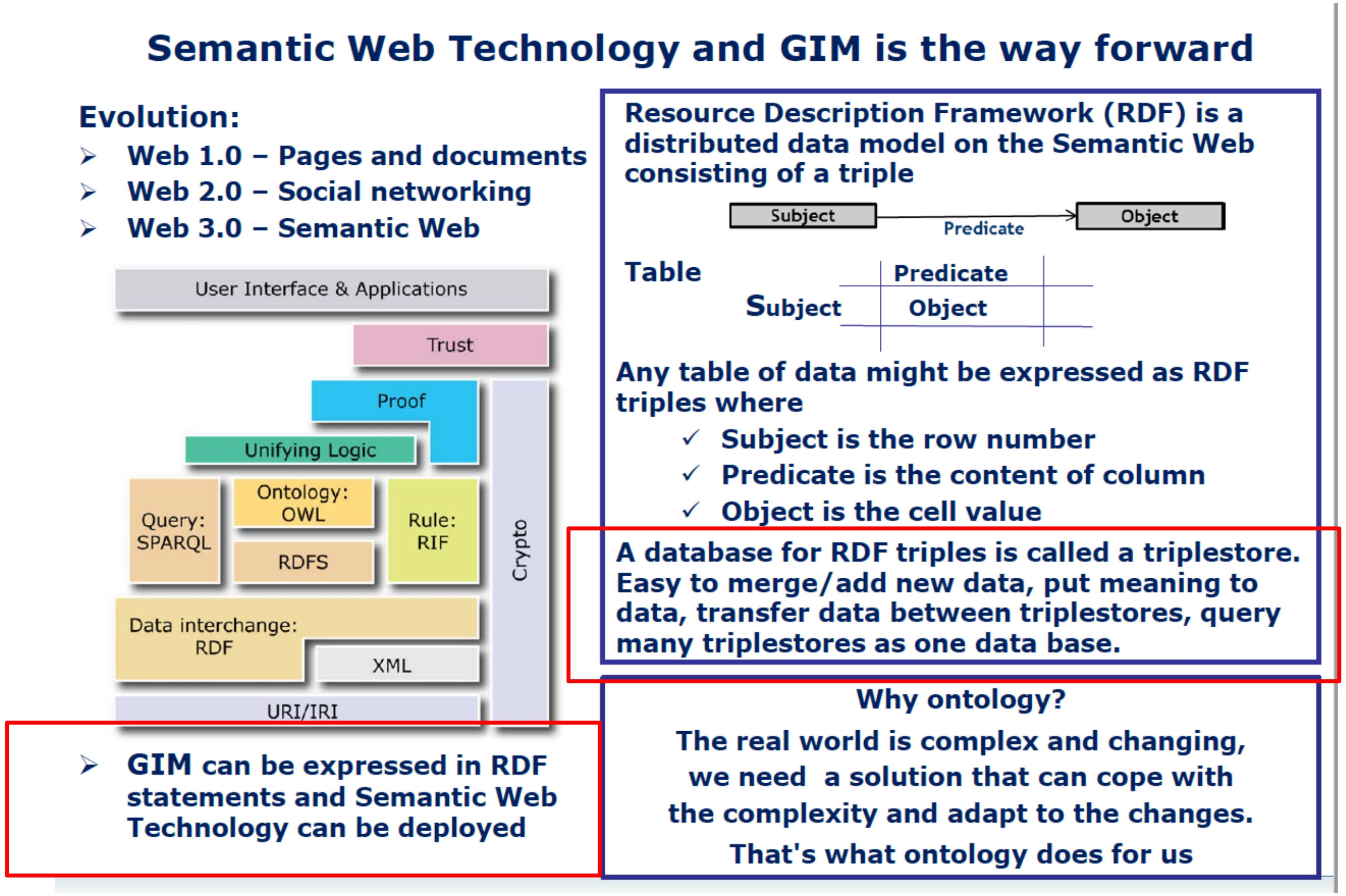


(Source for image : TopQuadrant fact sheet)

Solution provider point of view : RDF facilitates integration through :

- Easy import/export through XML compatibility.
- Flexible database schema (NoSQL triple store).
- SPARQL queries (W3C standard) allowing querying in a compact intuitive way.
- Semantic data model (light ontology) allowing semantic tagging of data and discovery of implicit facts (inferences).

Solution : RDF database for data integration



Customer point of view: key points :

1. A Generic Information Modeling (GIM) standard for data integration.
   - International standard *ISO15926* for lifecycle data integration and interoperability (cf. chapter 10).
2. Semantic web technologies.
3. An ontology.
4. A triple store.

Cost savings estimated at several billions euro per year.

(Source : Langeland 2013).

Agenda

1	RDBMS challenges (why NoSQL)
2	Data integration challenges
3	RDF based data integration
4	Graph DBs and triple stores
5	Case Study in Oil & Gaz
6	Ontology based data access

NoSQL databases

- NoSQL databases are usually divided in the following categories :
  - Key-value stores.
  - Document stores.
  - Wide column stores (also called column family stores).
  - Graph stores.
- A triple store is a special kind of graph database dedicated to storing RDF triples.
- NoSQL transactions typically focus less on ACID requirements, but rather on BASE

  Basically **A**vailable, **S**oft state, **E**ventual consistency.

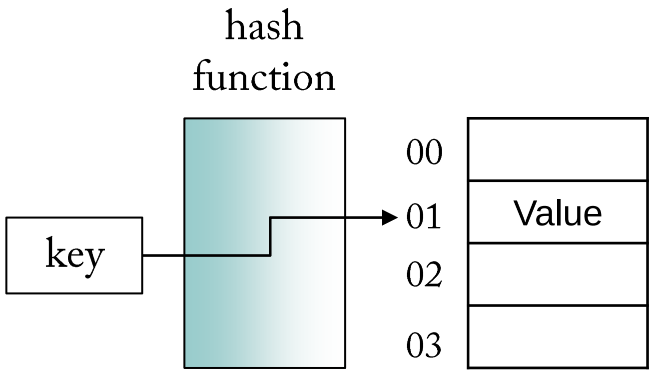
Aggregate NoSQL databases

Aggregate NoSQL databases are all based on key-value associations :

- They differ on what they use as values.
- They are sometimes referred as distributed hash tables.

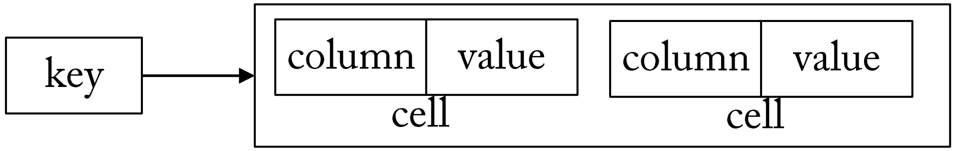
❑ Key value stores :

- A key is mapped to a value through a hashing function.
  The value can be complex (list, JSON object, image, video…).



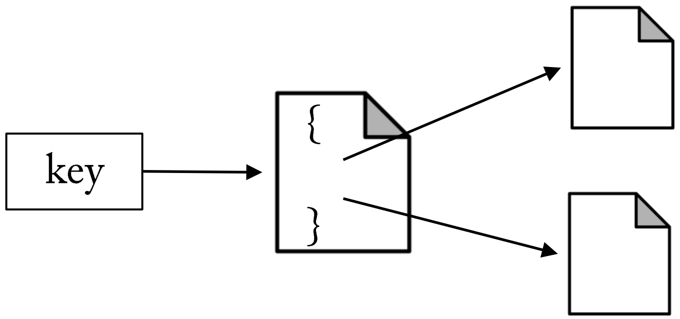
❑ Column family stores :

- A row index allows to access a column family (group of columns where each cell can have a name and value).



❑ Document stores :

- The database is organized as a collection of documents, each accessible by an ID.
- Each document is represented in a specific format (e.g., JSON) and can embed subdocuments, forming a hierarchy.



Aggregate NoSQL databases

The DB-Engines Ranking ranks DBMSs according to their popularity.

Key value stores

Rank Feb 2021	Rank Jan 2021	Rank Feb 2020	DBMS
1.	1.	1.	Redis
2.	2.	2.	Amazon DynamoDB
3.	3.	3.	Microsoft Azure Cosmos DB
4.	4.	4.	Memcached
5.	6.	6.	etcd

Colum family stores

Rank Feb 2021	Rank Jan 2021	Rank Feb 2020	DBMS
1.	1.	1.	Cassandra
2.	2.	2.	HBase
3.	3.	3.	Microsoft Azure Cosmos DB
4.	4.	4.	Datastax Enterprise
5.	5.	5.	Microsoft Azure Table Storage

Document stores

Rank Feb 2021	Rank Jan 2021	Rank Feb 2020	DBMS
1.	1.	1.	MongoDB
2.	2.	2.	Amazon DynamoDB
3.	3.	4.	Microsoft Azure Cosmos DB
4.	4.	3.	Couchbase
5.	6.	6.	Firebase Realtime Database

Graph databases

- Aggregate NoSQL databases :
  - Can represent links, but those are typically unidirectional, and may require processing steps to be identified.
  - Are not optimized for fast graph traversal.

- Graph databases focus on storing and accessing graphs.

They are part of the NoSQL family but not aggregate databases.

Advantages :

- Efficient data retrieval as graph traversal through connected data.
- Flexibility (NoSQL approach).
- Handling relations as first-class citizens.

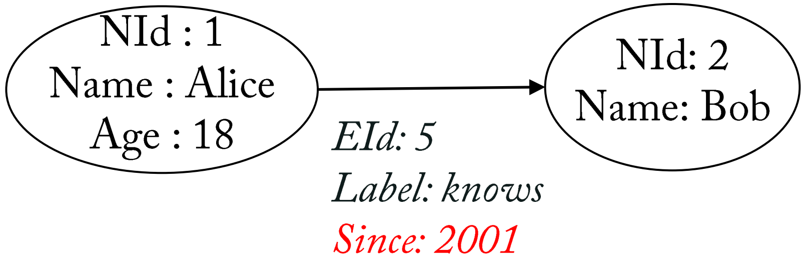
Trend 10: Relationships form the foundation of data and analytics value

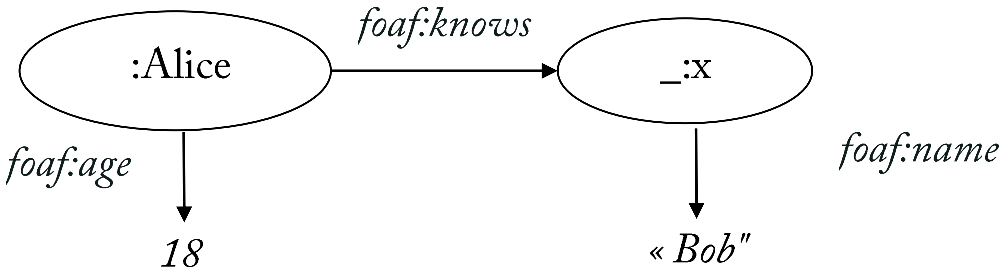
By 2023, graph technologies will facilitate rapid contextualization for decision making in 30% of organizations worldwide. Graph analytics is a set of analytic techniques that allows for the exploration of relationships between entities of interest such as organizations, people and transactions.

It helps data and analytics leaders find unknown relationships in data and review data not easily analyzed with traditional analytics.

https://www.gartner.com/smarterwithgartner/gartner-top-10-trends-in-data-and-analytics-for-2020/

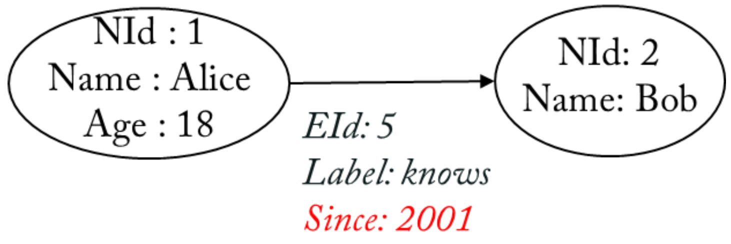
Property graph databases versus RDF triple stores

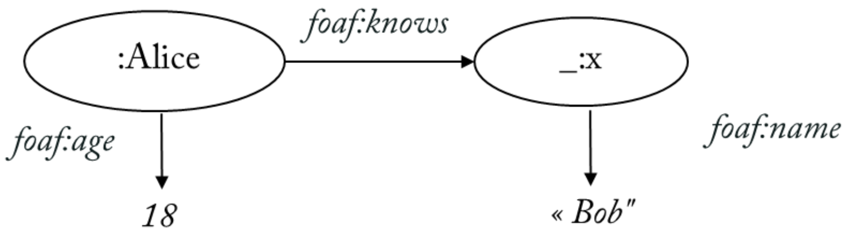




- Property graph databases
  - Data model : labelled property graphs (both nodes and links have properties).
  - A link may have e.g. a timestamp and/or a weight.
- RDF stores or triple stores :
  - Data model : RDF triples – quads if we add RDF graph names : <s, p, o> or <s, p, o, g>.
  - Relations can only have properties through reification.

Property graph databases versus RDF triple stores



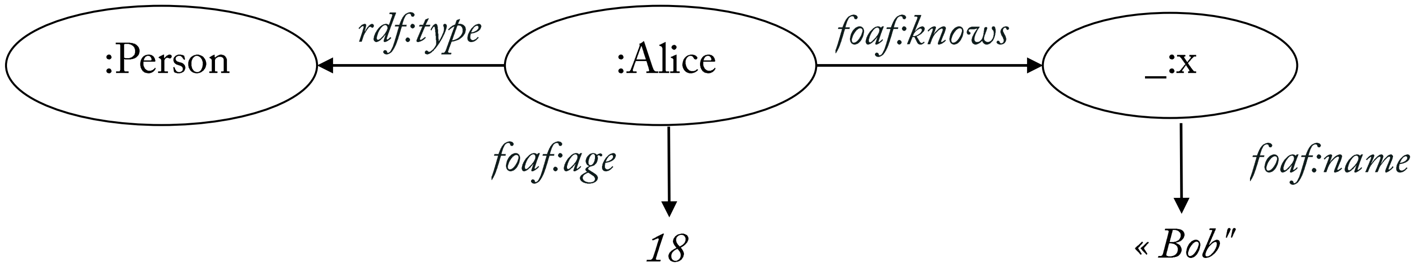


- Property graph databases
  - Are typically node centric.
  - Will typically have their own proprietary query language (e.g. Neo4J Cypher).
  - Have ad-hoc proprietary semantics.
  - May store more conveniently varied types of graphs (hypergraphs, weighted graphs …).

- RDF stores or triple stores :
  - Are typically edge-centric.
  - Support the standards of the semantic web (RDF, possibly RDFS, OWL), and SPARQL as query language.
  - Have clear logical formal semantics and may support logical inferences (depending on the DBMS engine).
  - Well suited for integration with XML, for web-based data integration, large scale knowledge graphs.

Implementation of RDF stores

- Simplest implementation : a relational table
  - No restructuring required if the ontology changes.
  - Inserts are easy.
  - Joins are expensive.



Subject	Predicate	Object
<http://www.example.org/Alice>	<http://xmlns.com/foaf/0.1/knows>	_:blanknode001
<http://www.example.org/Alice>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<http://www.example.org/Person>
_:blanknode001	<http://xmlns.com/foaf/0.1/name>	Bob
<http://www.example.org/Alice>	<http://xmlns.com/foaf/0.1/age>	18^^<http://www.w3.org/2001/XMLSchema#int>

Implementation of RDF stores ./.

❑ Improvements

SELECT ?entity WHERE
{?entity a :Person.
?entity foaf:knows ?x .
?x foaf:name "Bob" . }

Prefix	URI
:	<http://www.example.org/>
foaf:	<http://xmlns.com/foaf/0.1/>

Prefix table
Small simple data structure.

Id	String
1	<http://www.example.org/Alice>
2	<http://xmlns.com/foaf/0.1/knows>
3	<http://www.example.org/Person>
4	<http://xmlns.com/foaf/0.1/name>
5	Bob
6	_:blanknode001
7	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

Dictionary

id-to-string

string-to-id

Subject	Predicate	Object
1	2	6
1	7	3
6	4	5

Triple store
Short identifiers (integers) – gain of space.
Native (Jena) or non-native (Marklogic uses an underlying document store).
Specific indexing techniques.

string-to-id : efficient search structure (e.g. B+trees) sometimes combined with string compression techniques.
id-to-string : constant time direct access (e.g. array).

A ranking of graph databases and RDF stores

Rank Feb 2021	Rank Jan 2021	Rank Feb 2020	DBMS	Database Model
1.	1.	1.	Neo4j	Graph
2.	2.	2.	Microsoft Azure Cosmos DB	Multi-model
3.	3.	3.	OrientDB	Multi-model
4.	4.	4.	ArangoDB	Multi-model
5.	5.	↑ 7.	JanusGraph	Graph
6.	↑ 7.	↓ 5.	Virtuoso	Multi-model
7.	↑ 8.	↑ 9.	GraphDB	Multi-model
8.	↓ 6.	↓ 6.	Amazon Neptune	Multi-model
9.	9.	↑ 10.	FaunaDB	Multi-model
10.	↑ 11.	↑ 12.	Stardog	Multi-model

Rank Feb 2021	Rank Jan 2021	Rank Feb 2020	DBMS	Database Model
1.	1.	1.	MarkLogic	Multi-model
2.	2.	↑ 3.	Apache Jena - TDB	RDF
3.	↑ 4.	↓ 2.	Virtuoso	Multi-model
4.	↑ 5.	↑ 5.	GraphDB	Multi-model
5.	↓ 3.	↓ 4.	Amazon Neptune	Multi-model
6.	6.	6.	Stardog	Multi-model
7.	7.	7.	AllegroGraph	Multi-model
8.	8.	8.	Blazegraph	Multi-model
9.	9.	↑ 11.	RDF4J	RDF
10.	↑ 11.	10.	4store	RDF

(source DB-Engines Ranking : not including secondary database models)

The main commercial systems are also active

The top 5 commercial systems, February 2021

Rank	System	Score	Overall Rank
1.	Oracle	1317	1.
2.	Microsoft SQL Server	1023	3.
3.	IBM Db2	158	6.
4.	Microsoft Access	114	11.
5.	Splunk	89	13.

The top 5 open source systems, February 2021

Rank	System	Score	Overall Rank
1.	MySQL	1243	2.
2.	PostgreSQL	551	4.
3.	MongoDB	459	5.
4.	Redis	153	7.
5.	Elasticsearch	151	8.

Spatial and Graph features in Oracle Database

In keeping with Oracle's mission to help people see data in new ways, discover insights, unlock endless possibilities, Oracle Database now includes the Machine Learning, Spatial and Graph features. If you have an Oracle Database license, you can use all the industry-leading Machine Learning, Spatial and Graph capabilities for development and deployment purposes on-premise and in Oracle Cloud Database Services.

IBM

IBM Graph

IBM Graph

Fully managed graph database-as-a-service that enables enterprise applications and is built on open source database technologies

Microsoft

Azure Cosmos DB

Fast NoSQL database with open APIs for any scale

Reasoning with RDF stores

- Entailment regimes :
  - RDF can support multiple entailment regimes (RDF, RDFS, OWL) (cf. chapter 7).
- Materialization or saturation :
  - The entailment rules are applied to generate tuples until the graph is saturated (cf. chapter 3).
  - This speeds queries up, but updates are more costly (saturation needs to be redone to maintain truth).
  - Some DMBSs use saturation as a basic approach with various optimization techniques.
  - For large scale knowledge graphs, link prediction (triple materialization) is typically achieved through statistical relational learning approaches (cf. chapter 10).
- Query rewriting :
  - The query is transformed until it can be executed against the DBMS without entailment rules.
  - Ontology-based data access, seen in this chapter, is a good example.

Agenda

1	RDBMS challenges (why NoSQL)
2	Data integration challenges
3	RDF based data integration
4	Graph DBs and triple stores
5	Case Study in Oil & Gaz
6	Ontology based data access

What do we mean by a case study ?

- Social Sciences : a method of analysis and a specific research design for examining a problem … in most circumstances to generalize across populations *(University South California)*.
- Business : a documented study of a specific real-life situation or imagined scenario, used as a training tool in business schools and firms *(business dictionary.com)*.
- IT : demonstrates the effective use of information technology resources. Illustrates information technology related experiences in organizations, with background information, project implementation successes and failures and lessons learned (*casestudyinc.com)*.

- We will use the IT definition. Analysis grid :
  - Quality of the sources (scientific papers, news, industry fact sheets …).
  - Business problem that the users / organisation is trying to solve.
  - Existing situation ("*before*", "*as-is*").
  - Solution proposed ("*after*", "*to be*").
  - Results (evaluation, deployment status, business benefits).

Statoil: case study paper

Recent.

Scientific journal.

Many authors. Results paper from a european project (Optique).

Customer is signing the paper.

Web Semantics: Science, Services and Agents on the World Wide Web 44 (2017) 3–36

Contents lists available at ScienceDirect

Web Semantics: Science, Services and Agents on the World Wide Web

ELSEVIER

journal homepage: www.elsevier.com/locate/websem

Ontology Based Data Access in Statoil

CrossMark

Evgeny Kharlamov[a], Dag Hovland[c], Martin G. Skjæveland[c], Dimitris Bilidas[b], Ernesto Jiménez-Ruiz[c], Guohui Xiao[d,*], Ahmet Soylu[f], Davide Lanti[d], Martin Rezk[d], Dmitriy Zheleznyakov[a], Martin Giese[c], Hallstein Lie[e], Yannis Ioannidis[b], Yannis Kotidis[g], Manolis Koubarakis[b], Arild Waaler[c]

[a] University of Oxford, Department of Computer Science, Wolfson Building, Parks Road, OX1 3QD, Oxford, UK
[b] National and Kapodistrian University of Athens, Panepistimiopolis, Ilissia, 15784, Athens, Greece
[c] Department of Informatics, University of Oslo, Blindern, 0316, Oslo, Norway
[d] KRDB Center, Free-University of Bozen-Bolzano, Pzza Domenicani 3, 39100, Bolzano, Italy
[e] Statoil ASA, Stavanger, Norway
[f] NTNU – Norwegian University of Science and Technology, Teknologiveien 22, 2815, Gjøvik, Norway
[g] Athens University of Economics and Business, 76 Patission Street, 10434, Athens, Greece

Looking at the abstract

OBDA : use ontology to abstract from schema details. Ontology connected to DB via mappings.

Present a real industrial experience (Statoil).

Key achievements:
- semi-automatic ontology and mappings creation.
- optimized translation in data queries over federated DBs.
- query interface for non IT users.

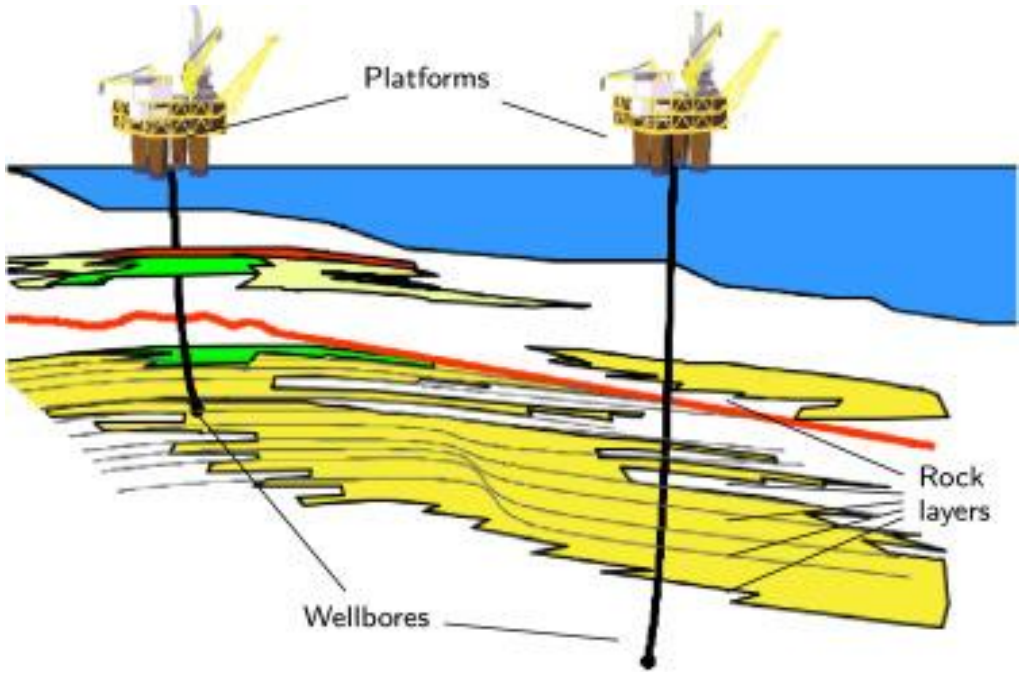
Deployed and evaluated.

Abstract summarizes the content (not just a table of content) !

A B S T R A C T

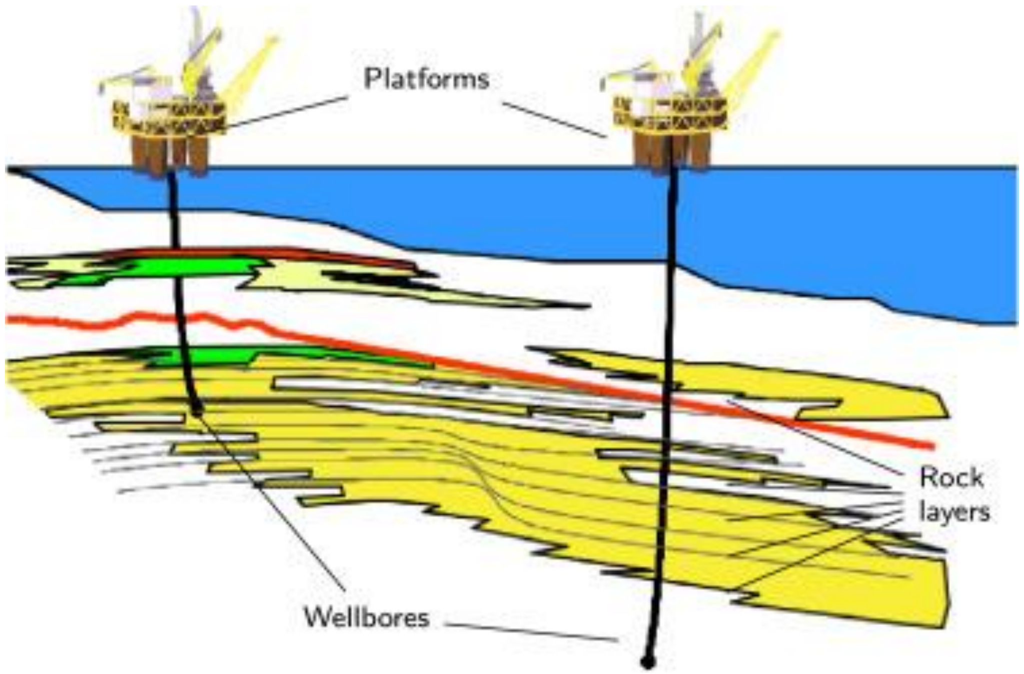
Ontology Based Data Access (OBDA) is a prominent approach to query databases which uses an ontology to expose data in a conceptually clear manner by abstracting away from the technical schema-level details of the underlying data. The ontology is 'connected' to the data via mappings that allow to automatically translate queries posed over the ontology into data-level queries that can be executed by the underlying database management system. Despite a lot of attention from the research community, there are still few instances of real world industrial use of OBDA systems. In this work we present data access challenges in the data-intensive petroleum company Statoil and our experience in addressing these challenges with OBDA technology. In particular, we have developed a deployment module to create ontologies and mappings from relational databases in a semi-automatic fashion; a query processing module to perform and optimise the process of translating ontological queries into data queries and their execution over either a single DB of federated DBs; and a query formulation module to support query construction for engineers with a limited IT background. Our modules have been integrated in one OBDA system, deployed at Statoil, integrated with Statoil's infrastructure, and evaluated with Statoil's engineers and data.

The problem



- Problem : find in a timely manner new exploitable accumulations of oil or gas in given areas by analyzing data about these areas.
- Data :
  - Seismic investigations.
    - Analyzing information from small explosive charges to estimate rock composition.
  - Rock samples / log curves from driller wellbores :
    - Rock samples taken during drilling;
    - Measurements from sensors along wellbore.
  - General geological knowledge.

Business needs



- Steps involved are :
  1. Find relevant wellbore, seismic, and other data in Statoil databases,
  2. Analyze these data with specialized analytical tools.
- Step 1 is too time-consuming :
  - Too complex for end-users, too lengthy with IT.
  - 30% - 70% of time spent on finding and analyzing the right data *(Crompton 2008)* .
- Background information : potential savings: €70,000,000 per year *(Source : Calvanese 2012)*.

Existing situation: architecture and data

	EPDS	GeoChemDB	Recall	CoreDB	OW	Compass
Overview						
Tables	1595	90	22	15	78	895
Mat. views	27	4				
Views	1703	41	12		1026	1004
Columns	8378	3396	430	63	16668	30638
Tables by no. rows						
0 rows	1130	3	2		15	512
1 row	1152	9	2		4	34
1 < rows ≤ 10	135	9	1	4	15	117
10 < rows ≤ 100	83	20	3	2	17	80
100 < rows ≤ 1 000	58	30	3	4	12	87
1 000 < rows ≤ 10 000	63	10	5	1	11	42
10 000 < rows ≤ 100 000	57	4	2	1	2	19
100 000 < rows ≤ 1000 000	35	3	4	2		4
1 000 000 < rows	12	3	2	1		
Tables by no. columns						
1 col	4				5	6
1 < cols ≤ 10	586	68	7	11	47	527
10 < cols ≤ 100	1032	23	13	4	26	353
100 < cols ≤ 1 000		3	2			9
1 000 < cols						
Mat. views by no. columns						
1 col						
1 < cols ≤ 10	23	1				
10 < cols ≤ 100	4	3				
100 < cols						
Views by no. columns						
1 col	3	2			3	2
1 < cols ≤ 10	526	12			555	509
10 < cols ≤ 100	1174	14	9		461	471
100 < cols ≤ 1 000		13	3		7	22
1 000 < cols						

- 6 internal databases + 1 external.
  - Relational databases from multiple vendors (Oracle for main DB).
  - Analytical tools from multiple vendors.
- Difficulty 1 : size of database.
  - Large (700 GB, 3000 tables, 57000 columns).
  - Complex schemas poorly or not documented.
- Difficulty 2 : SQL queries.
  - Queries for data extraction are typically large.

  Queries over main database EPDS may contain thousands of words and have 50–200 joins.
  - If predefined queries are not adequate, need to build new ones.

ask an **IT expert** to translate the information need into SQL

```
SELECT
    WELLBORE.IDENTIFIER,
    PTY_PRESSURE.PTY_PRESSURE_S,
    STRATIGRAPHIC_ZONE.STRAT_COLUMN_IDENTIFIER,
    STRATIGRAPHIC_ZONE.STRAT_UNIT_IDENTIFIER
FROM WELLBORE,
    PTY_PRESSURE,
    ACTIVITY FP_DEPTH_DATA
        LEFT JOIN (PTY_LOCATION_1D FP_DEPTH_PT1_LOC
            INNER JOIN PICKED_STRATIGRAPHIC_ZONES ZS
                ON ZS.STRAT_ZONE_ENTRY_MD <= FP_DEPTH_PT1_LOC.DATA_VALUE_1_O AND
                    ZS.STRAT_ZONE_EXIT_MD >= FP_DEPTH_PT1_LOC.DATA_VALUE_1_O AND
                    ZS.STRAT_ZONE_DEPTH_UOM = FP_DEPTH_PT1_LOC.DATA_VALUE_1_OU
            INNER JOIN STRATIGRAPHIC_ZONE
                ON ZS.WELLBORE = STRATIGRAPHIC_ZONE.WELLBORE AND
                    ZS.STRAT_COLUMN_IDENTIFIER = STRATIGRAPHIC_ZONE.STRAT_COLUMN_IDENTIFIER AND
                    ZS.STRAT_INTERP_VERSION = STRATIGRAPHIC_ZONE.STRAT_INTERP_VERSION AND
                    ZS.STRAT_ZONE_IDENTIFIER = STRATIGRAPHIC_ZONE.STRAT_ZONE_IDENTIFIER)
            ON FP_DEPTH_DATA.FACILITY_S = ZS.WELLBORE AND
                FP_DEPTH_DATA.ACTIVITY_S = FP_DEPTH_PT1_LOC.ACTIVITY_S,
    ACTIVITY_CLASS FORM_PRESSURE_CLASS
WHERE WELLBORE.WELLBORE_S = FP_DEPTH_DATA.FACILITY_S AND
    FP_DEPTH_DATA.ACTIVITY_S = PTY_PRESSURE.ACTIVITY_S AND
    FP_DEPTH_DATA.KIND_S = FORM_PRESSURE_CLASS.ACTIVITY_CLASS_S AND
    WELLBORE.REF_EXISTENCE_KIND = 'actual' AND
    FORM_PRESSURE_CLASS.NAME = 'formation pressure depth data'
```

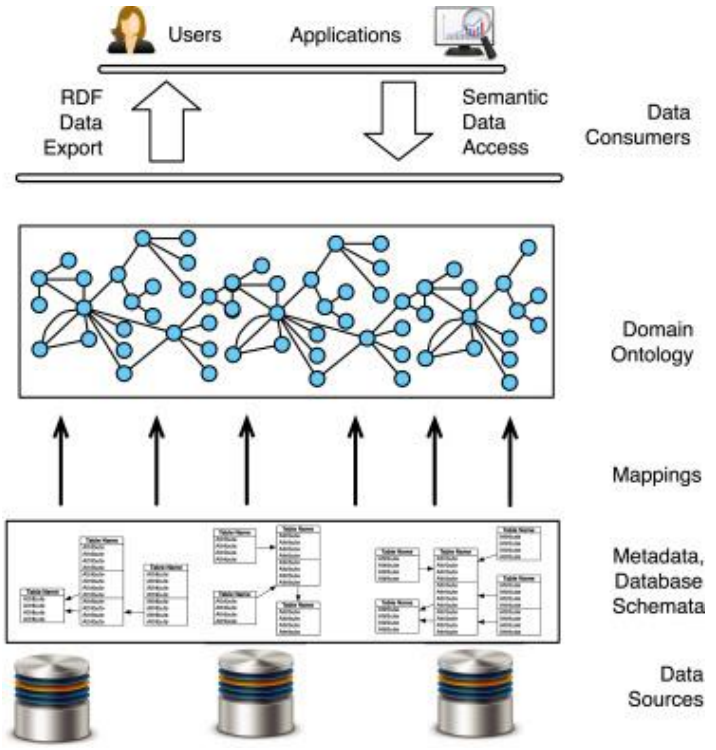
Existing situation : process + business objective

- Process :

1. Data access points fed by an ETL process (Extract, Transform, Load).
   - Transform: may steps involving projections, filtering, joins.
   - Setting up a new access point involves "a myriad of" data accesses and process steps !
2. Data access bottleneck :
   - 900 geologists and geophysicists.
   - Data analysis with existing access points is fast (hours).
   - But setting up of new access points may take up to 4 days.
   - IT must be involved and experienced IT personnel for theses tasks is scarce.

- Business objective : reduce setup time for new access points from days to hours.
  - If possible, without intervention of IT staff (!).

Solution architecture : ontology-based data access



$$\mathit{Class}(f_o(x)) \mapsto \mathtt{SQL}(x),\ \mathit{Property}(f_o(x), f_o(y)) \mapsto \mathtt{SQL}(x, y),$$
$$\mathit{Property}(f_o(x), f_v(y)) \mapsto \mathtt{SQL}(x, y),$$

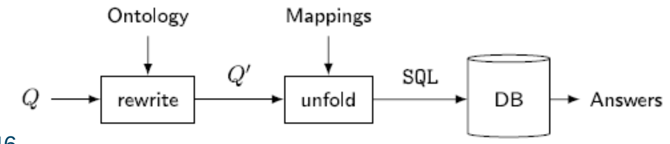
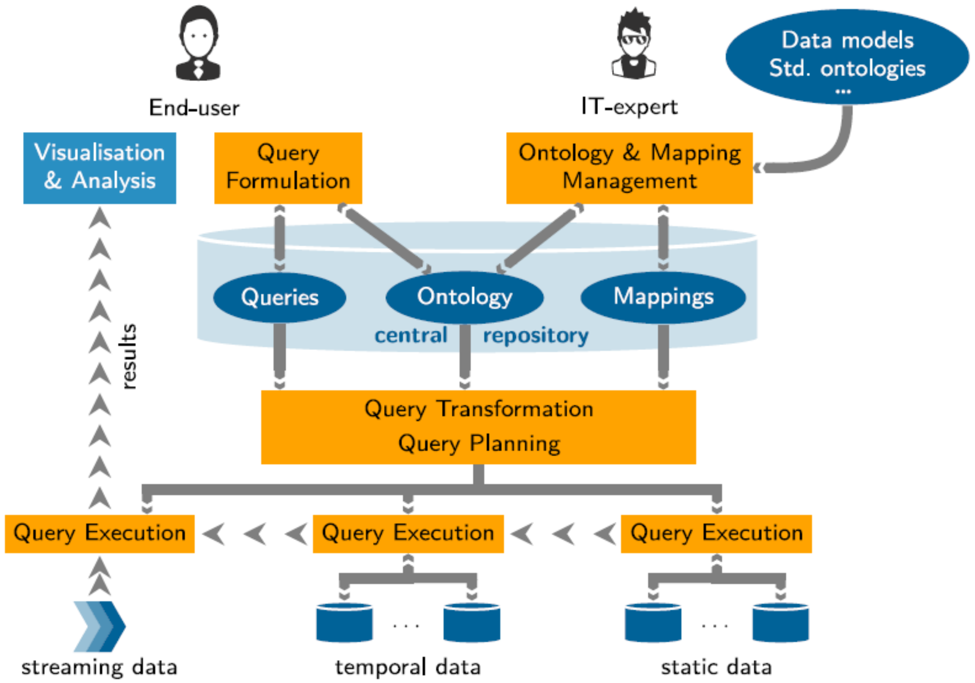


Fig. 5. Query processing in OBDA.

- A domain ontology mediates between users and data sources.
  - Export data in semantic format (RDF) or
  - Access database through ontology queries.
- Declarative mappings relate ontology to database schema.
  - Used to populate classes and properties.
- SPARQL is used for ontology queries.
  - Queries are handled in two steps : rewrite (query enriched on basis of axioms) and unfold into SQL.
- Advantages:
  - Declarative, smaller building blocks (map one class or 1 property), easier to maintain, reusable by any query.
  - OLTP and not OLAP.
  - Significantly reduced time to write queries.

Solution implemented: innovative aspects



❑ Challenges and solutions

1. Generating ontologies and mappings for large databases.
   - Semi-automatic bootstrapper extracting ontologies and mappings from relational schemas. Target is profile OLW 2 QL.
   - Ontology alignment techniques are used to complement bootstrapping.
   - A manually developed ontology covers general domain information.

2. Capacity to process semantic queries over massive amounts of data.
   - Optimized techniques for query rewriting, unfolding and execution.
   - Federated query over all the sources connected to the platform.
   - Results are fed back to visualization and analysis tools.

3. Difficulty for end users to use SPARQL.
   - Implemented a user-friendly query formulation tool.

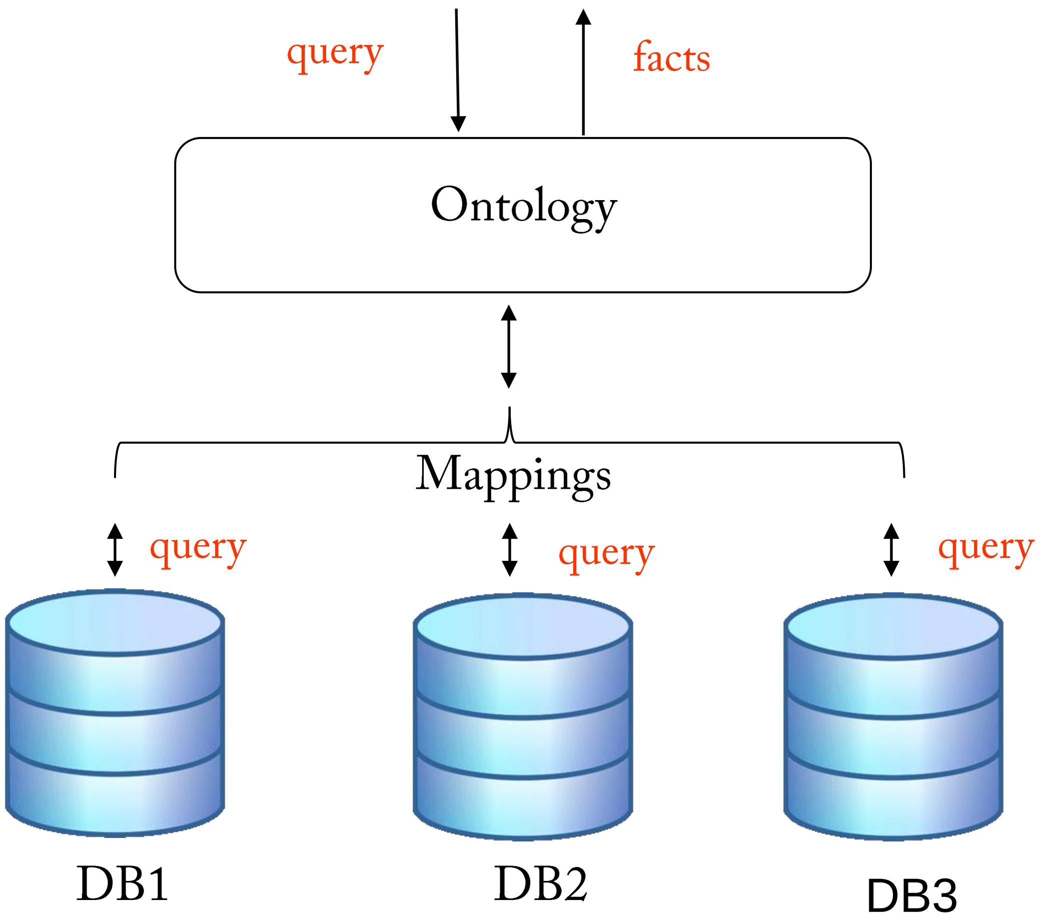
Evaluation

- Coverage :
  - Full catalogue of queries covered; results ok with manual addition of the general domain ontology.
  - Queries easier to formulate.
  - Wide range of vocabulary in the domain not fully achieved.
- Performance :
  - Execution time as good as before, even for federated queries; much better for distinct queries.
- Convenience :
  - Support broad range of users, task types, and interaction routines.
- Deployment :
  - Not deployed in production yet but tested in realistic Statoil environment with multiple servers.

Agenda

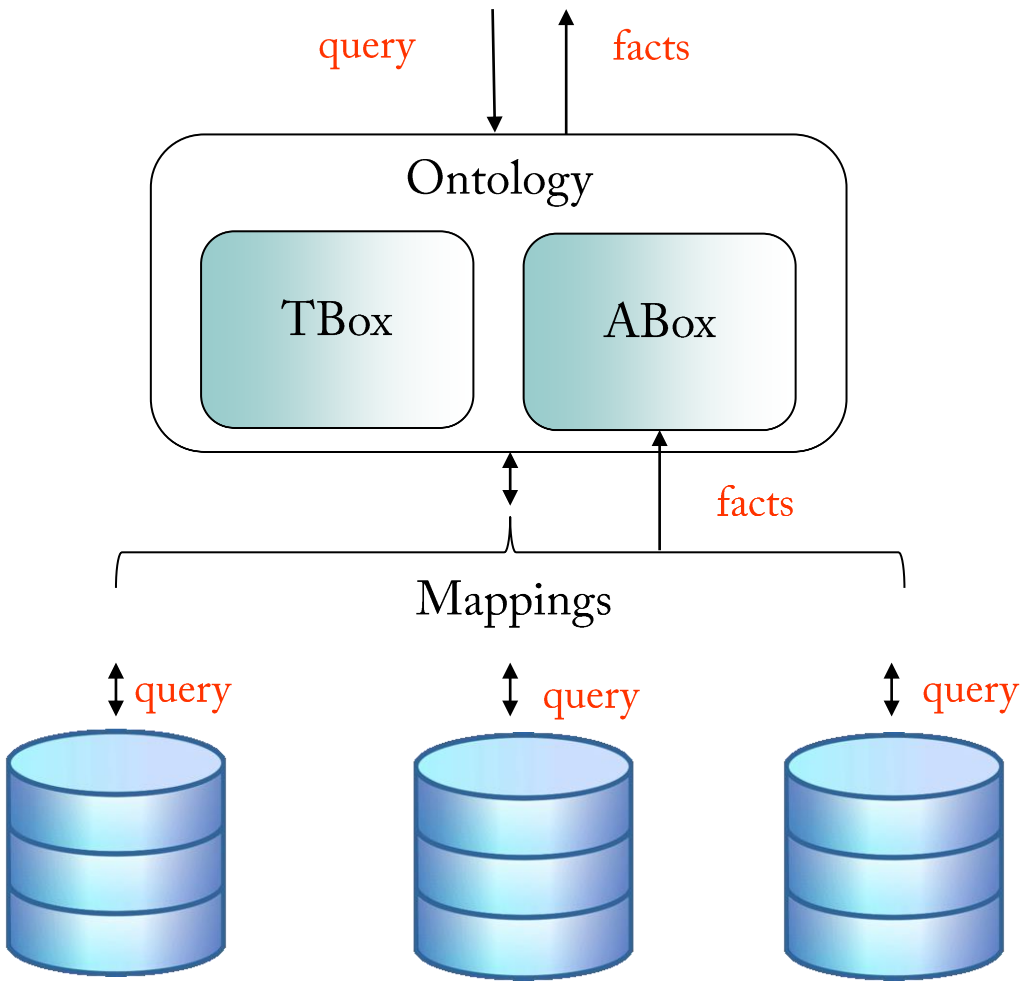
1. RDBMS challenges (why NoSQL)
2. Data integration challenges
3. RDF based data integration
4. Graph DBs and triple stores
5. Case Study in Oil & Gaz
6. Ontology based data access

OBDA main ideas



- The ontology provides :
  - A high-level global schema of multiple data sources;
  - A vocabulary for user queries.
- The OBDA system :
  - transforms queries into the vocabulary of the data sources through mappings.
  - delegates the actual query evaluation to a query answering system.
- Classical OBDA: relational DB, SQL, static data.
- Extensions :
  NoSQL data, temporal data, data streams…

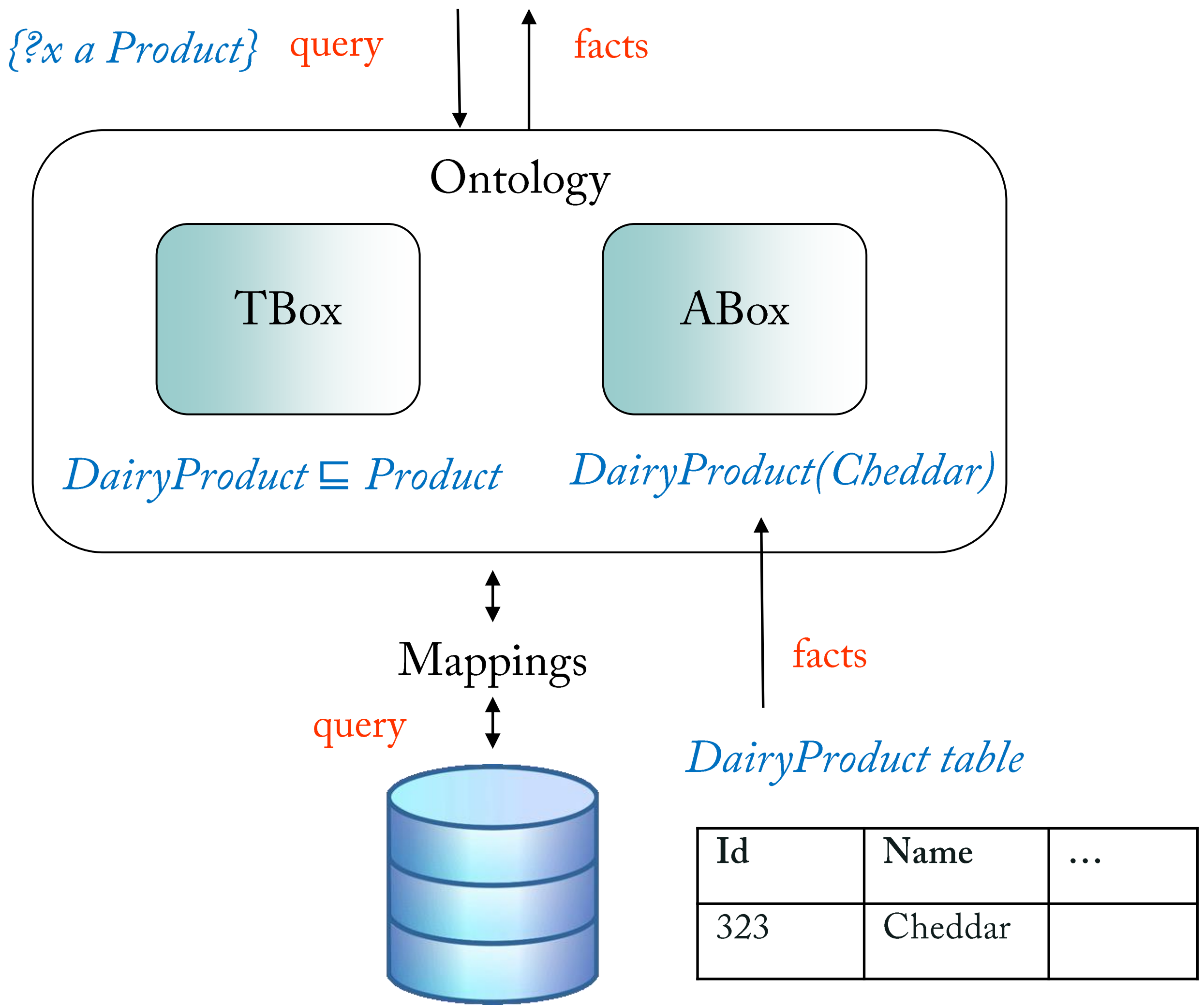
OBDA main ideas ./.



We consider description logics ontologies.

- Facts (assertions) should be in the ABox, however :
  - The ABox is virtual (too much data to populate it fully).
  - The relevant part of the ABox is materialized by executing queries over the data sources.
- The TBox captures the intensional information :
  - Concept descriptions; inclusion/equivalence axioms.
- The transformation of ontological queries into database queries uses the reasoning services of the ontology.

Differences between OBDA and databases



1. OBDA bridges two worlds :
   - The database world is based on the Closed World Assumption (CWA).

   If Cheddar is not stated as a Product, that is false.

   - The ontology world is based on the Open World Assumption (OWA).

   If Cheddar is not stated as a Product, that is unknown.

2. OBDA supports reasoning based on the intensional information in the TBox.

   The fact that Cheddar is a Product can be derived from the TBox by using the reasoning services.

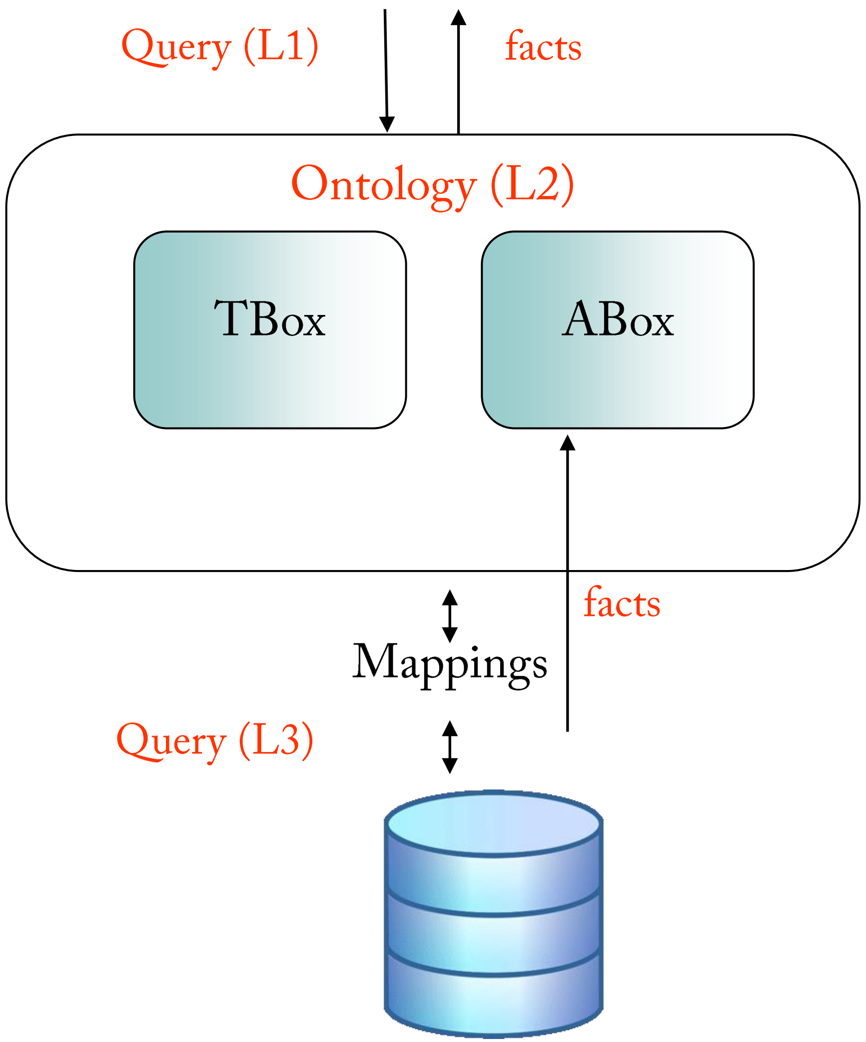
   (example from Optique project training material: http://optique-project.eu/training-programme/)

A quiz

- TBox
  - $\text{BelgianUniv} \sqsubseteq \text{University}$
  - $\text{NonBelgianUniv} \sqsubseteq \text{University} \sqcap \neg\text{BelgianUniv}$
  - $\exists\text{studentAt}.\top \sqsubseteq \text{Student}$
  - $\text{Student} \sqsubseteq \exists\text{studentAt}.\top$
- Database
  - BelgianUniv(ULiège), NonBelgianUniv(Oxford), Student(Jean), studentAt(André, ULiège), Institution(KUL).

Query	Database	ABox	Ontology
NonBelgianUniv(Oxford)	Yes	Yes	Yes
Student(André)	No	Don’t Know	Yes
University(ULiège)	No	Don’t Know	Yes
Institution ⊓ ¬University(KUL)	Yes	Don’t Know	Don’t Know

Questions to be adressed



1. What language L1 for querying the ontology ?
2. What language L2 (DL) for representing the ontology ?
3. What language L3 for querying the database ?

The relationships between these languages influence the possibility and complexity of rewriting queries and solving them against the DB.

4. How do we define and compute the answers ?

Choices of languages for OBDA 1 : ontology querying

1. To query the ontology, classical OBDA uses conjunctive queries :
   a fragment of FOL using only existential quantification and conjunction(*).

   Example: $q(x, y) = \exists z\ (Person(z) \wedge name(z, x) \wedge homepage(z, y))$

- In DL a conjunctive query CQ is of the form: $q(\vec{x}) = \exists(\vec{y})\ \varphi(\vec{x}, \vec{y})$ where
  - $\varphi(\vec{x}, \vec{y})$ is a conjunction of atoms such as $A(z)$ or $r(z_1, z_2)$;
    A being a concept name, r a role name, and z, z_1, z_2 individual names of variables from $\vec{x}$ or $\vec{y}$.
  - The free variables $\vec{x}$ are called distinguished variables, the bound variables $\vec{y}$ non-distinguished variables.
  - A solution is any assignment of values to the variables $\vec{x}$, part of an interpretation making the query *True*.
- Conjunctive queries can be expressed as SPARQL basic graph patterns.

  If x is renamed as ?name and y as ?site, the above example corresponds to the SPARQL query :

  SELECT ?name, ?site WHERE { ?person a :Person . ?person :name ?name . ?person :homepage ?site. }

  * : full FOL queries with ontologies are incomputable in the presence of incomplete information.

Choices of languages for OBDA 2 : database querying

2. The language for querying the database is SQL or …

   FOL, which is equally expressive as relational algebra, the core of SQL(*).

- An SQL engine can be used for the evaluation step.

** : at least the so-called "safe-range" fragment of FOL.*

Choices of languages for OBDA 3 : ontology representation

3. The ontology representation language is typically the OLW 2 QL profile.

- OLW 2 QL :
  - Covers the main features necessary to express conceptual models (UML, ER);
  - Is designed so that data stored in a database system can be queried through an ontology via a rewriting process, without any changes to the data;
  - Query answering is sound and complete in LOGSPACE(*) w.r.t. the size of the data.

    LOGSPACE ⊆ NLOGSPACE ⊆ P ⊆ NP ⊆ PSPACE ⊆ EXPTIME ⊆ EXPSPACE

  - This level of complexity is good for querying potentially large databases.

- OLW 2 QL is based on the DL Lite family of description logics. *(Calvanese 2012).*

* : actually, in AC0, a class from circuit complexity theory which is ⊆ LOGSPACE

Certain answers

- Databases handle a simple form of uncertain knowledge with NULL values.

  The semantics of NULL is however unclear :

  - A NULL can indicate that the value does not exist;
  - Or that the value is unknown at the time the database tuple was created.

- A certain answer denotes an answer which does not depend on uncertain data.

- At the ontology level, the certain answers $cert(q, \mathcal{O})$ to a query $q(\vec{x})$ for an ontology $\mathcal{O}$ contain those answers that hold for all models of $\mathcal{O}$.

  An assignment $\vec{x}_1 = \{a_1 \dots a_n\}$ to the variables of $\vec{x}$ is a certain answer to $q(\vec{x})$ for $\mathcal{O}$ iff :

$$\mathcal{O} \vDash q(\vec{x}_1)$$

  Or, for all interpretations $\mathcal{I}$:

$$\mathcal{I} \vDash \mathcal{O} \rightarrow \mathcal{I} \vDash q(\vec{x}_1)$$

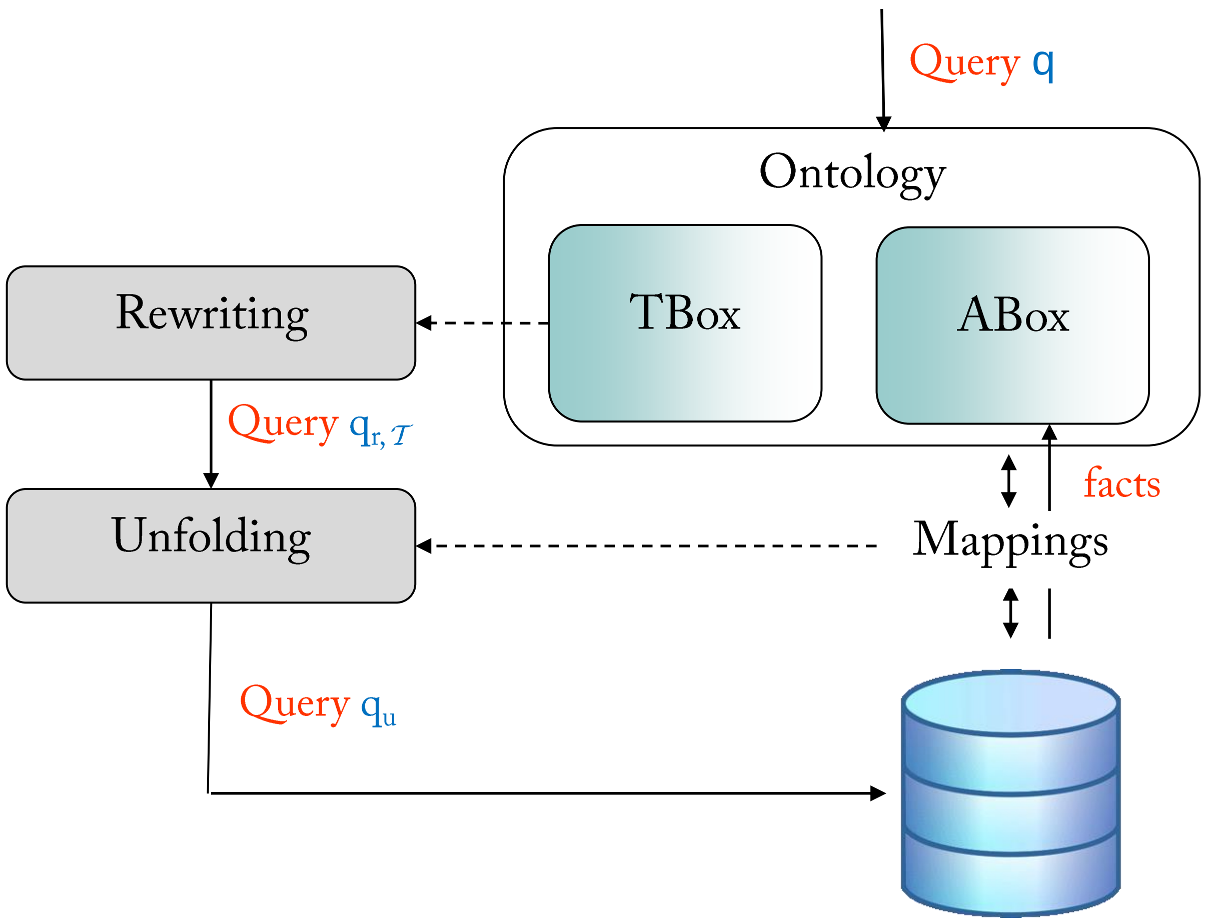
Dealing with the TBox

- Reminder
  - A database defines relations extensionally (listing the set of tuples which make the relation hold).
  - An ontology TBox defines concepts intensionally, through its concept definition axioms.
- Query answering requires to derive all extensional information from the TBox.
  - Example :

  With $\mathcal{A}$ = {Person(John), hasFather(John, Fred)} and $\mathcal{T}$ = {Person $\subseteq$ $\forall$hasFather.Person}

  a FOL query Person(x) must yield as certain answers : {John, Fred}.
- One could try to first compute all the extensional consequences of TBox + ABox.
- Unfortunately, for many DLs this is too expensive and can even be impossible.

Classical approach for dealing with the TBox: query rewrite



Aim : eliminate the TBox by transforming the query into another one which can be evaluated without the TBox with the same answers (perfect rewrite).

Given an ontology $\mathcal{O}$ = <$\mathcal{T}$, $\mathcal{A}$> and a conjunctive query q, there are three steps :

- Rewrite : produce from q a union of conjunctive queries $q_{r,\mathcal{T}}$ being the perfect rewrite of q w.r.t. $\mathcal{T}$.
- Unfold : using the mappings, transform the query into a query q_u which can be executed against the data source (SQL for RDBMS).
- Evaluate : execute the query q_u directly over the database, without considering $\mathcal{T}$.

Example of query rewriting from the case study

Database :

Location:

ID	Name
L1	Norway
L2	UK

Purpose:

ID	Name
P1	Shallow
P2	Injection

Wellbore:

ID	PurpID	Content	LocID
W1	P1	Dry	L1
W2	P2	Oil	L2

ExpWBore:

ID	Type
E1	Active
E2	Discovery

TBox T (in DL-Lite) :

ExplorationWellBore ⊆ WellBore

ShallowWellBore ⊆ WellBore

WellBore ⊆ ∃hasContent

- Query q(x) (in SPARQL)

SELECT ?x WHERE { ?x hasContent ?y}

- Query rewriten q_r :

SELECT ?x WHERE { ?x hasContent ?y}
UNION { ?x a WellBore}
UNION { ?x a ShallowWellBore}
UNION { ?x a ExplorationWellBore}

Example of query rewriting from the case study

- Mappings :
  - ExplorationWellBore(f(ID))
    →
    SELECT ID
    FROM ExpWBore
  - ShallowWellBore(f(W.ID))
    →
    SELECT W.ID
    FROM WellBore W, Purpose P
    WHERE W.PurpID = P.ID
    AND P.Name = "Shallow"
  - Haslocation(f(ID),f(LocID))
    →
    SELECT ID, LocID
    FROM WellBore

- Query rewritten q_r :

```
SELECT ?x WHERE { ?x hasContent ?y}
     UNION { ?x a WellBore}
     UNION { ?x a ShallowWellBore }
     UNION { ?x a ExplorationWellBore }
```

- Query unfolded q_u :

```
SELECT f(ID) AS x FROM ExpWBore
UNION
SELECT f(W.ID) AS x FROM WellBore W, Purpose P
WHERE W.PurpID = P.ID AND P.Name = "Shallow“
```

- Result : f(E1), f(E2), f(W1)

f is a constructor function which translates data instantiations to ontology instances.

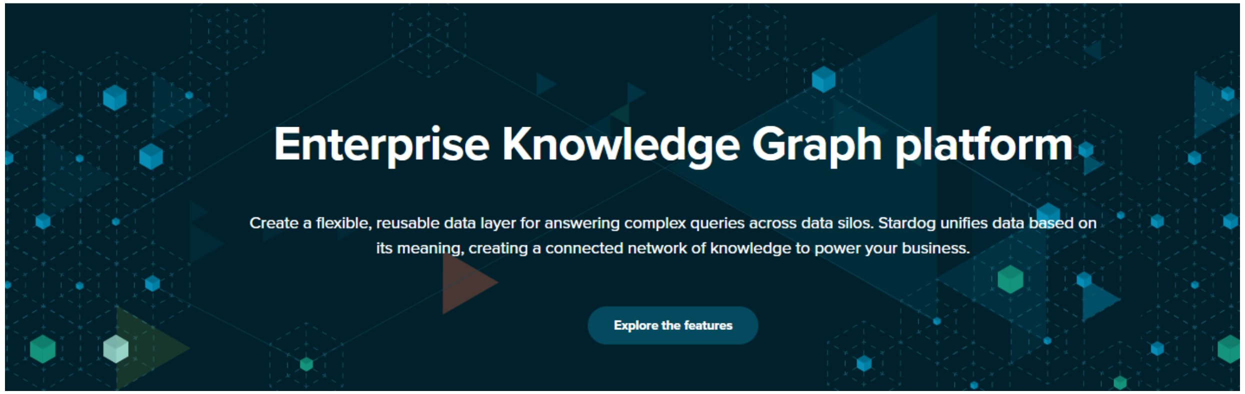
OBDA status

- State of the art OBDA system from university research : Ontop (Protégé plugin).
  - Resulting from work of EU project Optique.
  - Compliant with RDFS, OWL 2 QL, R2RML(*), and SPARQL.
  - Supports major RDBMSs : Oracle, DB2, MS SQL Server, Postgres, MySQL.
- Emerging industrial solutions (Stardog, Data.World, Oracle Spatial and Graph…).
- Ongoing research: Ontology based data integration (OBDI).
  - Heterogenous data sources and formats, including streaming and real-time data.
  - Continued optimizations of query processing and reasoning + explanation generation.

**: language for mapping from RDBs into RDF data sets*

Example of product solution

STARDOG Product Solutions Resources Company Learn Stardog Start for Free



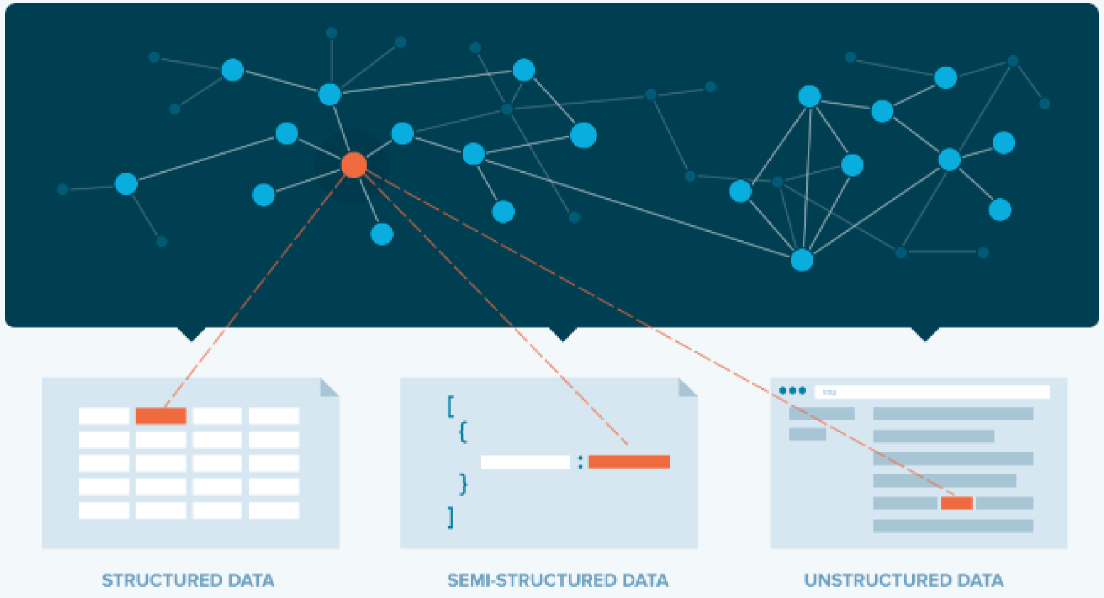
Based on open standards

Stardog offers first class support of the SPARQL, SPARQL*, and GraphQL query languages for interacting with your Knowledge Graph. At its core, Stardog is built on the RDF open standards of knowledge representation created by W3C, the same standards body that created the Web. This platform of open standards is designed to represent information —

Stardog's Enterprise Knowledge Graph platform is underpinned by a graph database that offers state-of-the-art performance for both storing **RDF data** and **executing SPARQL queries**. Stardog collects detailed statistics from the graph structure to compute accurate

Connect and query data of any structure

A Knowledge Graph connects to data sources within your company, enriches the data by finding connections across all sources, and creates a human- and machine-understandable output. Stardog accesses data with Connectors to all major SQL systems and the most popular NoSQL databases. In addition, our BITES pipeline extracts concepts from unstructured data like research papers, resumes, and regulatory documents, further enriching the Knowledge Graph. Access unified data via API or connect to analytics platforms using our BI/SQL Server. **Continue reading →**



(from company site https://www.stardog.com/)

Summary

- Relational DMBSs have limitations in coping with changes, both for OLTP and OLAP.
- NoSQL databases offer alternative approaches. Graph databases and RDF triple stores are (related) types of NoSQL databases.
- Data integration through RDF stores offers many advantages (natural fit with XML, semantic models, industrial level databases) and has seen significant life industrial applications.
- Reasoning with RDF stores is typically done by using saturation or query rewriting techniques.
- OBDA consists in querying databases at the ontology (knowledge) level, taking into account inferences and the intensional concept definitions from the TBox.
- Classical OBDA uses query rewriting techniques to handle the TBox, and is typically based on conjunctive queries, for ontologies of the OLW 2 QL profile.
- The industrial use of OBDA is less advanced than RDF integration. However real-life industry projects or pilots, as well as industrial product and service offerings are emerging.

References

- *[Calvanese 2012], D. Calvanese,* Ontology-Based Data Access: From Theory to Practice, *28e journees Bases de Donnees Avancees (BDA 2012), October 2012, Clermont-Ferrand, France.* http://www.inf.unibz.it/~calvanese/presentations/BDA-2012-obda-calvanese.pdf

- *[Crompton 2008] : J. Crompton,* Putting the FOCUS on Data, *W3C Workshop on Semantic Web in Oil & Gas Industry, Houston, Texas, 2008.*

- *[Curé and Blin 2015]:* RDF Database Systems, *(O. Curé and G. Blin, eds), Morgan Kaufmann, 2015.*

- *[Langeland 2013] : T. Langeland,* LogisticsHub – A Common Knowledgebase for Tracking Information for the Offshore Supply Chain, *PIXD Europe Autumn Conference, 2013.*

- *[Kharlamov et al. 2017]:* Ontology Based Data Access in Statoil, *Web Semantics: Science, Services and Agents on the World Wide Web, Elsevier, 2017.*

- *[Rosati 2014] : R. Rosati ,* Query Answering and Rewriting in Ontology-Based Data Access tutorial, *4th International Conference on Principles of Knowledge Representation and Reasoning, Vienna, Austria, July 2014.* http://www.dbai.tuwien.ac.at/kr2014/downloads/Rosati.pdf

- *[Xiao and Kontchakov 2018]: G. Xiaio and R, Knotchakov,* Ontology-based Data Access: Theory and Practice, *tutorial of the 27th International Joint Conference on Artificial Intelligence, Stockolm, 2018. http://ontop.inf.unibz.it/ijcai-2018-tutorial/*

THANK YOU