P. Bellini, M. Di Claudio, P. Nesi, N. Rauch

Slides from: M. Di Claudio, N. Rauch

Big Data Solutions

and applications

Slide del corso:

Sistemi Collaborativi e di Protezione

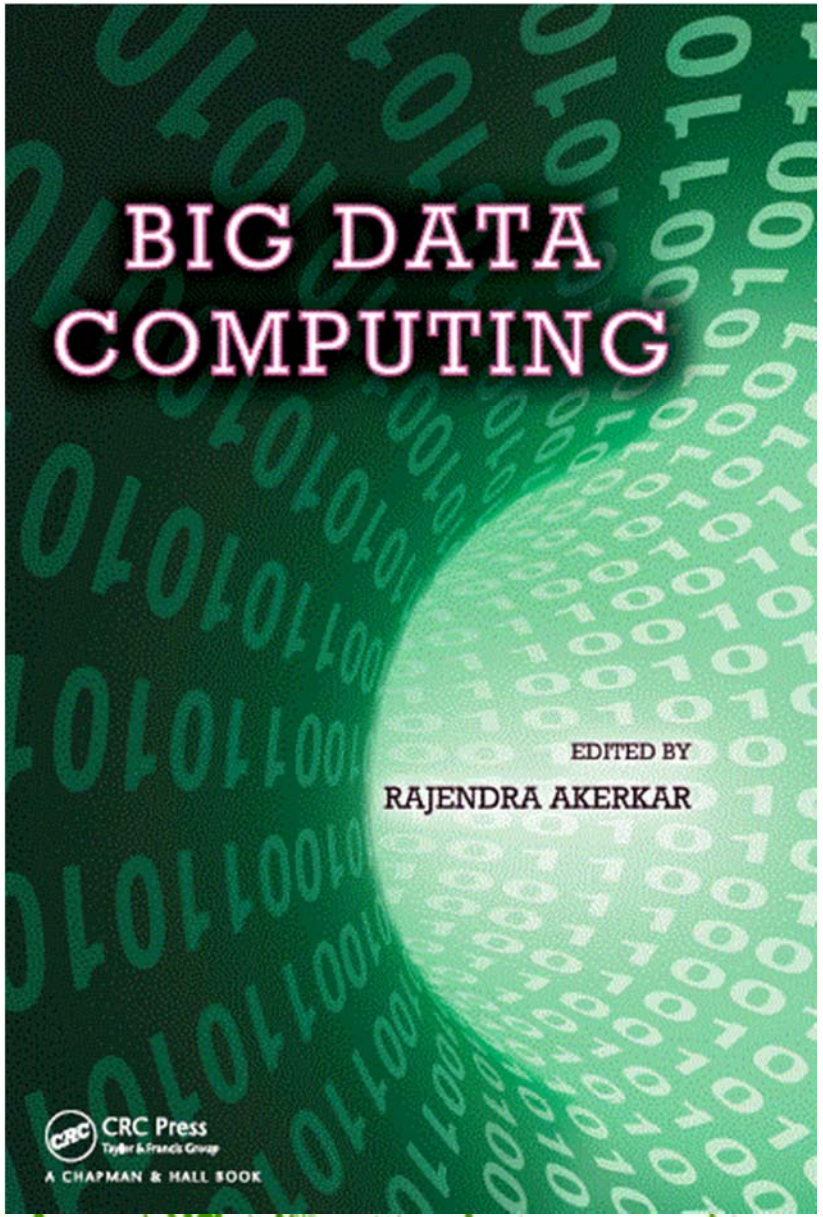
Corso di Laurea Magistrale in Ingegneria 2013/2014

http://www.disit.dinfo.unifi.it

Distributed Data Intelligence and Technology Lab

Related to the following chapter in the book:

- P. Bellini, M. Di Claudio, P. Nesi, N. Rauch, "Tassonomy and Review of Big Data Solutions Navigation", in "Big Data Computing", Ed. Rajendra Akerkar, Western Norway Research Institute, Norway, Chapman and Hall/CRC press, ISBN 978-1-46-657837-1, eBook: 978-1-46-657838-8, july 2013, in press. http://www.tmrfindia.org/bigdata.html



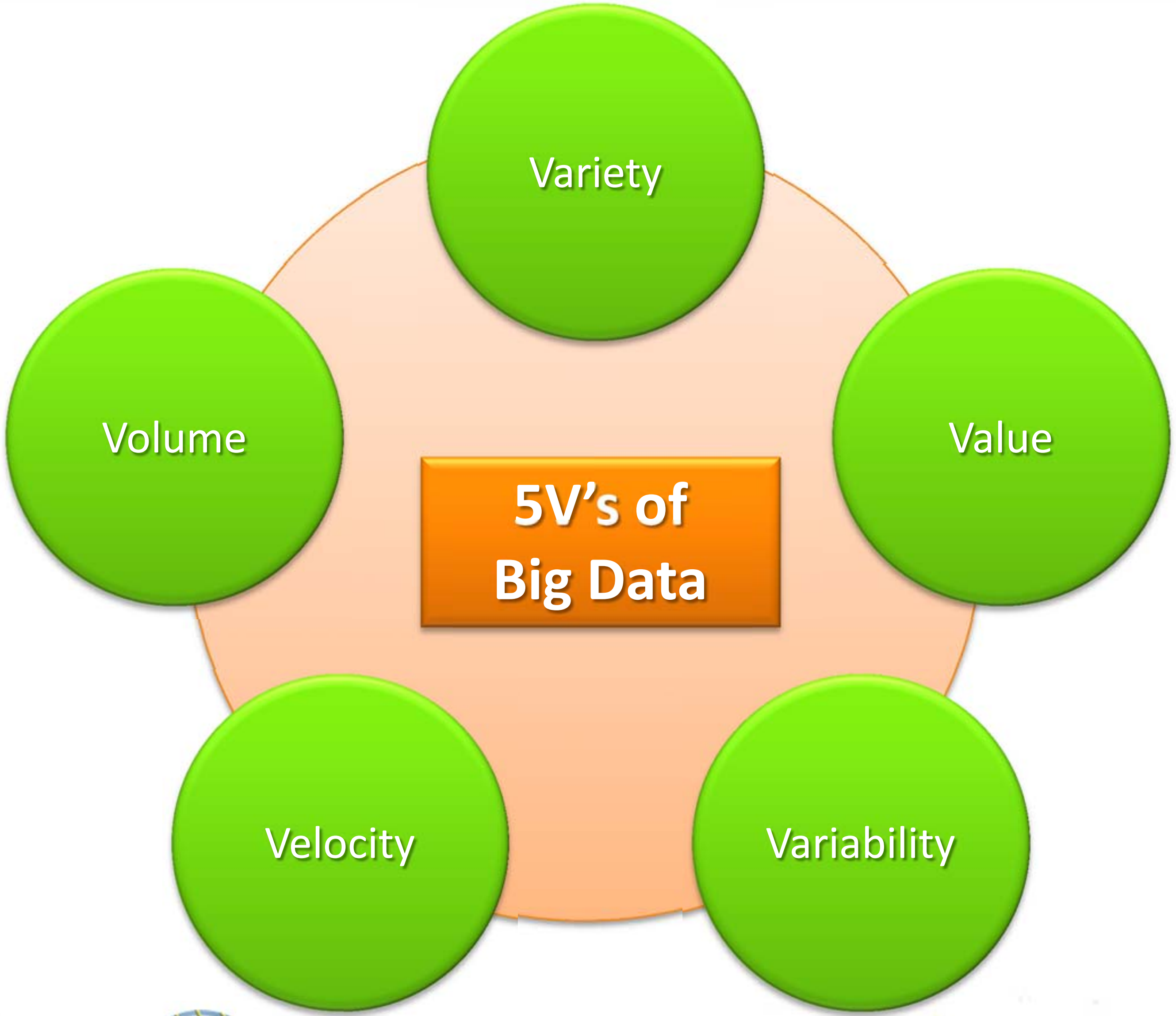
Index

Index

The Big Data Problem.

Index

What is Big Data?



5V's of Big Data (1/2)

- **Volume**: Companies amassing terabytes/petabytes of information, and they always look for faster, more efficient, and lower-cost solutions for data management.
- **Velocity**: How quickly the data comes at you. For time-sensitive processes, big data must be used as streams of Data in order to maximize its value.
- **Variety**: Big data is any type of data: structured and unstructured data such as text, sensor data, audio, video, click streams, log files and more.

5V's of Big Data (2/2)

- **Variability:** Refers to variance in meaning and in lexicon, but also to the variability in data structure.
- **Value**: Big Data hiding a great value. With the primary use you can extract only a part, the remaining value remains "dormant" until their secondary use. The value is all that you can gain from all possible modes of use of the data, the sum of many small parts that are slowly discovered. Once the data were eliminated after the first use.

Major problems of Big Data

How is it possible to discover their "**value**"?

- Using complex modeling process: hypotheses are formulated, statistical, visual and semantic models are implemented to validate them;
- Applying several techniques and analysis so much different to the data collected.

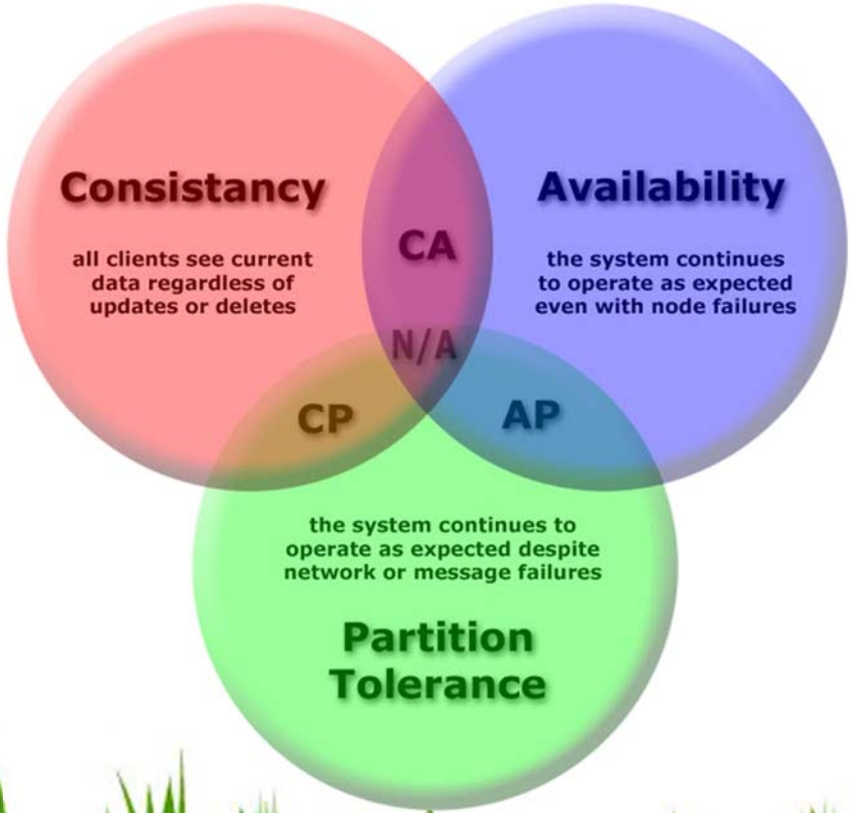
Problems:

- Identify a unique architecture adaptable to all possible area;
- The large number of application areas, so different from each other;
- The different channels through which data are daily collected.

Brewer’s Theorem (CAP’s Principle)

- **Consistency**: if the system is in a consistent state after an operation, all clients see the same data.
- **Availability**: every request received by a non-failing node in the system, must result in a response.
- **Partition tolerance**: the system continues to function even when split into disconnected subsets (by a network disruption).

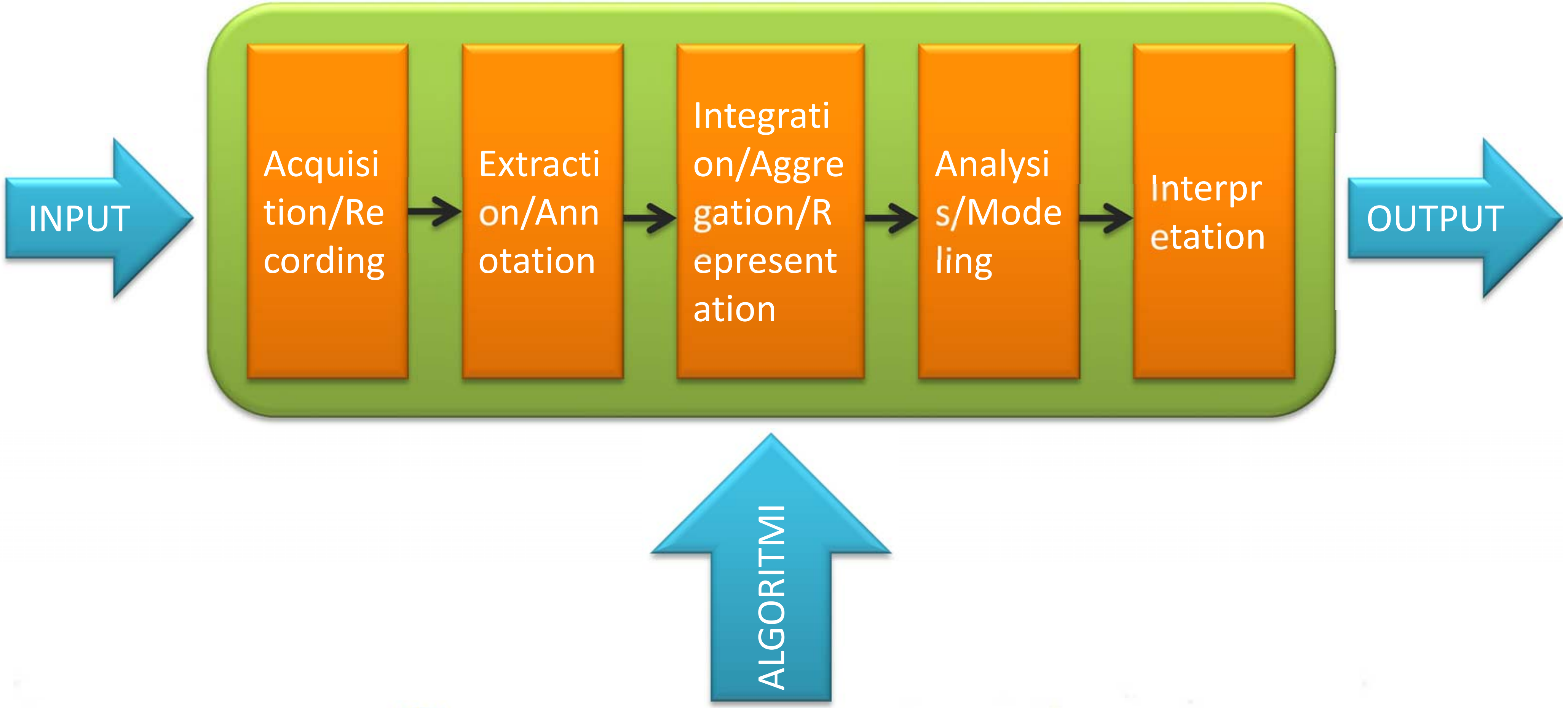
<u>You can satisfy **at most 2** out of this 3 requirements!</u>


Consistancy
all clients see current data regardless of updates or deletes
Availability
the system continues to operate as expected even with node failures
CA
N/A
CP
AP
the system continues to operate as expected despite network or message failures
Partition Tolerance

Possible Combinations with C, A and P

- **CA**: a database is able to provide distributed transactional semantics, only in the absence of network partition that separates server nodes;
- **CP**: if we are dealing with partitions, transactions to an ACID database may be blocked until the partition perfectly works, in order to avoid risk of inconsistency.
- **AP**: system is still available under partitioning, but some of the data returned may be inaccurate.

Big Data Analysis Pipeline



Pipeline: Data Acquisition and Recording

- Huge amount of data can be filtered and compressed by orders of magnitude.
  - **Challenge**: define these filters in such a way that they do not discard useful information.
- Detail regarding experimental conditions, procedures may be required to interpret the results correctly.
  - **Challenge**: automatically generate the right metadata.
- Must be possible to research both into metadata and into data systems.
  - **Challenge**: create optimized data structures that allow searching in acceptable times.

Pipeline: Information Extraction and Cleaning

- The information collected will not be in a format ready for analysis (surveillance photo VS picture of the stars).
  - **Challenge**: realize an information extraction process that pulls out information, and express it in a form suitable for analysis.
- Big Data are incomplete and errors may have been committed during Data Acquisition phase.
  - **Challenge**: define constraints and error models for many emerging Big Data domains.

Pipeline: Data Integration, Aggregation, Representation

- Data is heterogeneous, it is not enough throw it into a repository.
  - **Challenge**: create a data record structure that is suitable to the differences in experimental details.
- Many ways to store the same information: some designs have advantages over others, for certain purposes.
  - **Challenge**: create tools to assist in database design process and developing techniques.

Pipeline: Query Processing, Data Modeling, and Analysis

- Methods for querying and mining Big Data are different from traditional statistical analysis.
  - **Challenge**: create a scaling complex query processing techniques to terabytes while enabling interactive response times.
- Interconnected Big Data forms large heterogeneous networks, with which information redundancy can be explored to compensate for missing data, to crosscheck conflicting cases and to uncover hidden relationships and models.
  - **Challenge**: add coordination between database systems and provide SQL querying, with analytics packages that perform various forms of non-SQL processing (data mining, statistical analysis).

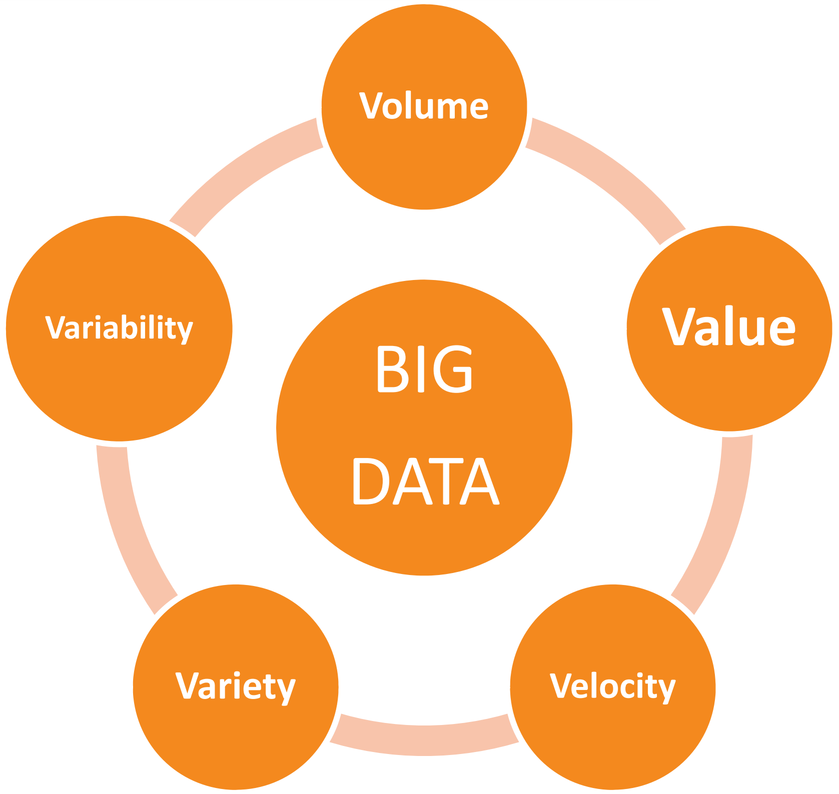
New concepts

- **Datification**: to taking information about all things and transforming it into a data format to make it quantified.
- Use this information in new ways to unlock the implicit, latent value of this information.
- When the data were “**few**”, it was desirable they were **accurate** (Random Sampling). BigData have changed the expectations of precision: to deal with these **large quantities** of data as something **imprecise and imperfect** allows us to make superior forecasts (**Predictive Analisys**).

Big Data Application Fields

Application Fields

“Big Data” problems one referred to the combination of large volume of data to be treated in short time.



There are **many areas** where big data are currently used with remarkable results and excellent future prospects to fully deal with the main challenges like data analysis, modeling, organization and retrieval.

Application Fields

- A major investment in Big data can lay the foundations for the next generation of advances in medicine, science, business and information technology…
- **Healthcare and Medicine**
- Data Analysis – Scientific Research
- Educational
- Energy and Transportation
- Social Network – Internet Service – Web Data
- Financial/Business
- Security

Healthcare and Medicine

- In Healthcare/Medical field large amount of information is collected about :
  - **Electronic Patient Record (EPR)**
  - **Symptomatology**
  - **Diagnoses**
  - **Therapies**
  - **Responses to treatments**
- In just **12 days** approximately **5000 patients** entered the emergency department.
- In Medical Research two main application are:
  - **Genomic Sequence Collections** (A single sequencing experiment yield 100 milion short sequences)
  - **Analysis of neuroimging data** (Intermidiate data stored ~ 1.8PetaBytes)

V
Volume
Velocity
Variability

Healthcare and Medicine

- Healthcare processes are characterized by the fact that several organizational units can be involved in the treatment process of patients and that these organizational units often have their own specific IT applications.
- At the same time – especially in the case of public hospitals - there is growing pressure from governmental bodies to refactor clinical processes in order to improve efficiency and reduce costs.

There is the need, in medical field, to use run time data
to support the analysis of existing processes.

Healthcare and Medicine

- **Data mining techniques** might be implemented to **derive knowledge** from this data in order to either identify new interesting pattern in infection control data or to examine reporting practices.
- **Process Mining** is accomplished through techniques of analysis and evaluation of event stored in log files. In hospitals with the Electronic Patient Record (EPR) have investigated techniques to the fast access and **extraction of information from event's log**, to produce easily interpretable models, using partitioning, clustering and preprocessing techniques.
- By **building a predictive model**, it could be possible either to provide **decision support** for specific triage and diagnosis or to produce effective plans for chronic disease management, enhancing the quality of healthcare and lower its cost.

Healthcare and Medicine

- At the base of genomics there are techniques of **gene cloning** and **sequencing of DNA** with the objective to **know the entire genome** of organisms.
- The knowledge of the entire genome allows to more easily **identify the genes involved** and to observe **how these interact**, particularly in the case of complex diseases such as **tumors**.
- **Huge amount of data**, **genetic knowledge, clinical practice, appropriateDB**

http://www.atgc-montpellier.fr/gkarrays

perform predictive studies on the incidence of certain diseases

A new opportunity is the use of **GkArrays** (combination of three arrays), an intelligent solution for indexing huge collections of short reads.

Application Fields

- A major investment in Big data can lay the foundations for the next generation of advances in medicine, science, business and information technology…
- Healthcare and Medicine
- **Data Analysis – Scientific Research**
- Educational
- Energy and Transportation
- Social Network – Internet Service – Web Data
- Financial/Business
- Security

Data Analysis – Scientific Research

- There are **several areas of science and research** in which the aim of Big Data analysis is to extract meaning from data and detemine what actions take.
  - **Astronomy** (Automated sky survey)
    - Today, **200 GB** of new high resolution optical data are being captured each evening by charge-coupled devices (CCDs) attached to telescopes.
  - **Biology** (Sequencing and encoding genes)
  - **Sociology** (Web log analysis of behavioral data)
    - With up to **15 million people** world wide accessing the internet each day (**nearly 10 hours per week on line**.).
  - **Neuroscience** (genetic and neuro-imaging data analysis)
- Scientific research is **highly collaborative** and involves scientists from different disciplines and from different countries.

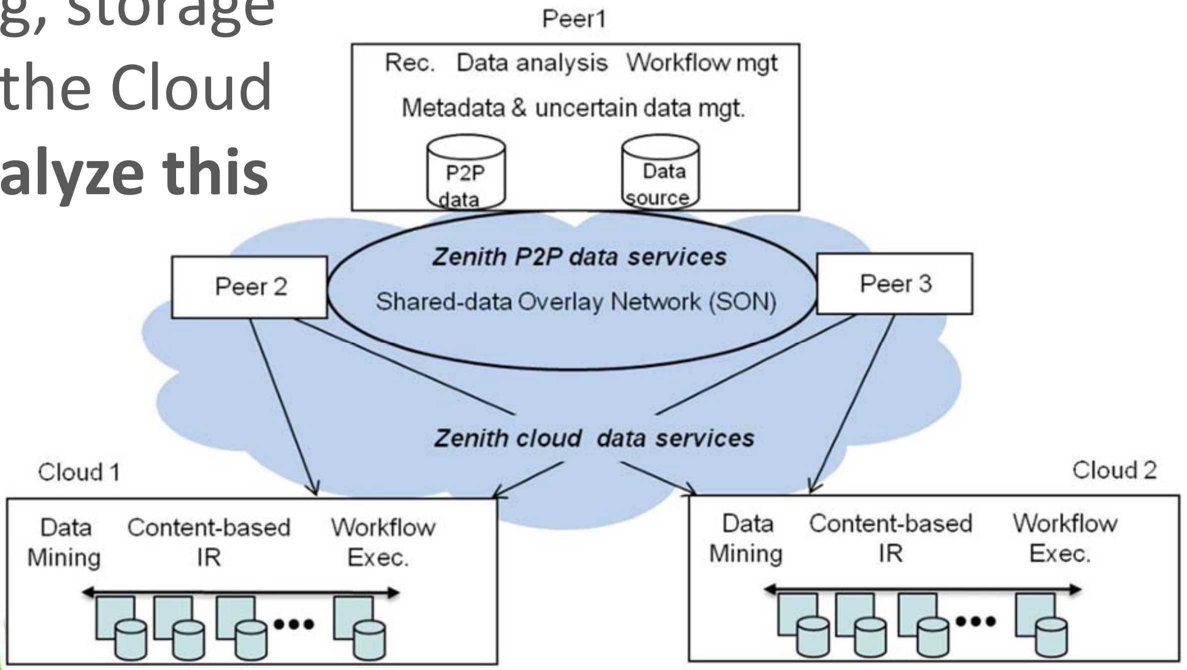
V
Volume
Variety
Variability

Data Analysis – Scientific Research

- To cope with the large amount of experimental data produced by modern disciplines, the University Montpellier started the **ZENITH** project.
- Zenith adopt a **hybrid architecture p2p/cloud**.

The idea of Zenith is to exploit p2p because the **collaborative nature of scientific data**, centralized control, and use the potentialities of computing, storage and network resources in the Cloud model, **to manage and analyze this large amount of data**.

http://www.sop.inria.fr/teams/zenith



Data Analysis – Scientific Research

- **Europeana** is platform for the management and sharing of multimedia content
- **Millions of content are indexed** and retrieved in real time
- Each of them was early modeled with a **simple metadata model (ESE)**
- A new more complex models with a **set of semantic relationships (EMD)**, is going to be adopted in short time

europeana
think culture

www.europeana.eu

Application Fields

- A major investment in Big data can lay the foundations for the next generation of advances in medicine, science, business and information technology…
- Healthcare and Medicine
- Data Analysis – Scientific Research
- **Educational**
- Energy and Transportation
- Social Network – Internet Service – Web Data
- Financial/Business
- Security

Educational

- Big Data has the potential to **revolutionize** not just research, but also **education**.
- Main educational data are
  - **students' performance** (Project KDD 2010 *)
  - **mechanics of learning**
  - **answers to different pedagogical strategies**

V Volume
Variability

A **new approach of teaching** can be defined by exploiting the big data management

- These data can be used to define **models to understand** what students actually know, their progresses and how to enrich this knowledge.

*** https://pslcdatashop.web.cmu.edu/KDDCup/**

Educational

- The proposed **new models of teaching** exploits the potential of **computer science** and **technology** combined with **techniques for data analysis** with the aim of clarifying issues taking into account pedagogical, psychological and learning mechanisms, to define **personalized instruction**, that meets the different needs of different individual students or groups.
- Another sector of interest, in this field, is the **e-learning domain**, where are defined two mainly kinds of users: **the learners** and the **learning providers**.
  - All personal details of learners and the online learning providers' information are stored in specific db, so applying **data mining** with e-learning can be able to realize **teaching programs targeted to particular interests** and needs through an efficient **decision making**.

Application Fields

- A major investment in Big data can lay the foundations for the next generation of advances in medicine, science, business and information technology…
- Healthcare and Medicine
- Data Analysis – Scientific Research
- Educational
- **Energy and Transportation**
- Social Network – Internet Service – Web Data
- Financial/Business
- Security

Energy and Transportation

- The movement of people, public transport and private vehicles within and between metropolitan cities is critical to economic strength and quality of life and involves a great variety of data:
  - **Trains and Bus schedules**
    - **About 33.000 tourists visit Rome every day**
    - **About 1.600.000 workers move to Rome at day**
  - **GPS information**
  - **Traffic interruptions**
  - **Weather**

V Volume
Variety
Velocity
Variability

- A **data-centric approach** can also help for enhancing the efficiency and the dependability of a transportation system.
  - The optimization of **multimodal transportation infrastructure** and their **intelligent use** can improve traveller experience and operational efficiencies.

Energy and Transportation

- Through the **analysis and visualization** of detailed road network data and **the use of a predictive model** it is possible to achieve an intelligent transportation environment.
- Furthermore, through the merging of high-fidelity geographical stored data and **real-time sensor networks** scattered data, it can be made an efficient urban planning system that **mix public and private transportation**, offering people more flexible solutions (**Smart Mobility**).
  - Amsterdam – Copenaghen – Berlino – Venezia – Firenze – Bologna …
  - **http://www.smartcityexhibition.it/**
    - This new way of travelling has interesting implications for energy and environment.

Energy and Transportation

- In the field of energy resources optimization and environmental monitoring, very important are the data related to the consumption of electricity.
- The analysis of a set of **load profiles and geo-referenced information**, with appropriate data mining techniques, and the construction of predictive models from that data, could define **intelligent distribution strategies** in order to lower costs and improve the quality of life.

Energy and Transportation

- Researchers have shown that instrumenting a home with just three sensors for :
  - **Electricity, Power, and Water** – possible to determine the resource usage of individual appliances.
  - There is an opportunity to transform homes and larger **residential areas into rich sensor networks**, and to integrate information about personal usage patterns with information about energy availability and energy consumption, in support of evidence-based power management.

Application Fields

- A major investment in Big data can lay the foundations for the next generation of advances in medicine, science, business and information technology…
- Healthcare and Medicine
- Data Analysis – Scientific Research
- Educational
- Energy and Transportation
- **Social Network – Internet Service – Web Data**
- Financial/Business
- Security

Social Networks Big Data

2012:

- **Facebook**: more than 10 million photos uploaded per hour, 3billions “like” button/comment per day;
- **Google Youtube**: 800 million users upload ~1h of video per second;
- **Twitter**: more than 400 million tweets per day.
- **Instagram:** 7.3 million unique users per day

2009:

- **Facebook**: 3 billion photos uploaded per month; “like” button was implemented.
- **Google Youtube**: all users upload 24h of video per minute;
- **Twitter**: 50 million tweets per day.
- **Instagram** was created in 2010.

Social Network – Internet Service – Web Data

- The **volume of data** generated by **internet services**, **websites**, **mobile applications** and **social network** is great, but speed of production is variable, due to **human factor**.
- From these large amounts of data collected through social networks, researchers try to **predict the collective behavior**, or for example through the trend of the Twitter hash tag are able to **identify models of influence**.

V Volume, Variety
Velocity, Variability

- In a broader sense by all this information is possible to **extract knowledge** and data relationships, by improving the **activity of query answering**.

Social Network – Internet Service – Web Data

- Some researcher have proposed an **alternative use of these data** to **create a novel form of urban living**, in an initiative called ConnectiCity. Key aspect are:
- Create a set of tools to capture in real-time various forms of city-relevant user generated content from a variety of types of sources:
  - Social networks – websites – mobile applications…

  To interrelate it to the territory using Geo-referencing, Geo-Parsing and Geo-Coding techniques, and to analyze and classify it using **Natural Language Processing techniques** to identify:
  - Users' emotional approaches
  - Networks of attention and of influence and trending issues
  - Topics of interest

Social Network – Internet Service – Web Data

- Some researcher have proposed an **alternative use of these data** to **create a novel form of urban living**, in an initiative called ConnectiCity. Key aspect are:
- To **make this information available** and **accessible** both centrally and peripherally, to enable the creation of **novel forms of decision-making** processes as well as to experiment innovative models for participated, peer to peer, user-generated initiatives.

Project link - http://www.connecticity.net/
http://www.opendata.comunefi.it

Application Fields

- A major investment in Big data can lay the foundations for the next generation of advances in medicine, science, business and information technology
- Healthcare and Medicine
- Data Analysis – Scientific Research
- Educational
- Energy and Transportation
- Social Network – Internet Service – Web Data
- **Financial/Business**
- Security

Financial and Business

- The task of **finding patterns in business data** is not new. Traditionally business analysts use statistical techniques.
- Nowadays **widespread use of pc and networking** technologies has created **large electronic repositories** that store numerous business transaction.
  - the magnitude of data is ~ **50-200 PBs** at day
  - **Internet audience** in Europe is **about 381,5 milion unique visitors**
  - **40% of European citizens** do online shopping
- These data can be analyzed in order to define:
  - **Prediction about the behavior of users**
  - **Identify buying pattern of individual/group customers**
  - **Provide new custom services**.

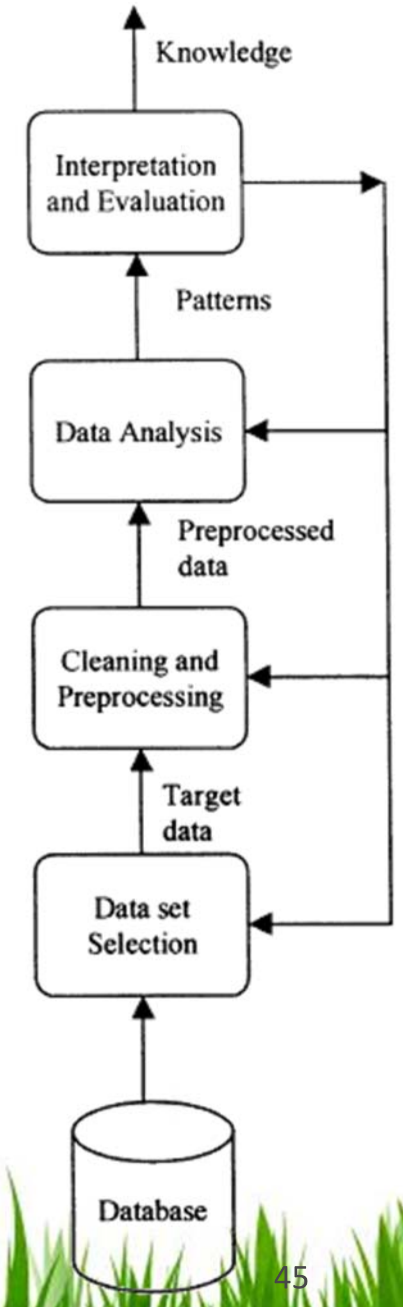
V Volume Velocity Variability

Using Data warehousing technology and mature machine learning techniques.

Financial and Business

- In financial field, instead, investment and **business plans may be created thanks to predictive models** derived using techniques of reasoning and used to discover meaningful and interesting patterns in business data.

1. **Selection** of data for analysis (from a network of DB)
2. **Cleaning operation** removes discrepancies and inconsistencies
3. Data set **is analyzed** to identified patterns (models that show the relationship among data).
4. It should be possible to **translate the model** into actionable business plan, that help the organization achieve its goal.
5. A model/pattern that satisfies these condition become **business knoweldge**.



Application Fields

- A major investment in Big data can lay the foundations for the next generation of advances in medicine, science, business and information technology…
- Healthcare and Medicine
- Data Analysis – Scientific Research
- Educational
- Energy and Transportation
- Social Network – Internet Service – Web Data
- Financial/Business
- **Security**

Security

- **Intelligence, Surveillance, and Reconnaissance (ISR)** define topics that are well suited for data-centric computational analyses,
  - close to **one zettabyte (10^{21}bytes or one billion terabytes)** of digital data are generated each year.
- Important data sources for intelligence gathering include:
  - **Satellite and UAVs Image**
    - **Intercepted communications**: civilian and military, including voice, email, documents, transaction logs, and other electronic data - **5 billion mobile phones in use worldwide.**

V
Volume
Variety
Velocity
Variability

Security

- **Radar tracking data**.
- **Public domain sources** (web sites, blogs, tweets, and other Internet data, print media, television, and radio).
- **Sensor data** (meteorological, oceanographic, security camera feeds).
- **Biometric data** (facial images, DNA, iris, fingerprint, gait recordings).
- **Structured and semi-structured information** supplied by companies and organizations: airline flight logs, credit card and bank transactions, phone call records, employee personnel records, electronic health records, police and investigative records.

Security

- **The challenge** for intelligence services is to **find, combine, and detect patterns and trends in the traces of important information**.
- **The challenge becomes one of finding meaningful evolving patterns in a timely manner among diverse, potentially obfuscated information across multiple sources**. These requirements greatly increase the need for very sophisticated methods to detect subtle patterns within data, **without generating large numbers of *false positives*** *so that we* do not find conspiracies where none exist.
- As models of how to effectively exploit large-scale data sources, we can look to Internet companies, such as Google, Yahoo, and Facebook.

Security

- Within the world of intelligence, we should consider **computer technology** and a consolidated **machine learning technology** as a way to augment the power of human analysts, rather than as a way to replace them.
- The key idea of machine learning is:
  - **First apply structured statistical analysis** to a data set to generate a ***predictive model*** .
  - ***Then to apply this model*** *to different data streams to* ***support*** **different forms of analysis**.

Overview of Big Data solutions

Index

4 classes of Main Requirements

- **Architecture aspects:** processing a very huge dataset is important to optimize workload, for example with a parallel architecture or providing a dynamic allocation of computation resources.
- **Data Management aspects:** the main solutions available are moving in the direction of handle increasing amounts of data.
- **Access/Data Rendering aspects:** is important to be able to analyze Big Data with Scalable Display tools.
- **Data Analysis|Mining/Ingestion aspects:** Statistical Analysis, Facet Query, answer queries quickly and quality data in terms of update, coherence and consistency are important operation that an ideal architecture must support or ensure.

Data Management Aspects

Index

Data Management Aspects (1/2)

- **Scalability**: Each storage system for big data must be scalable, with common and cheap hardware (increase the number of storage discs.)
- **Tiered Storage**: To optimize the time in which we want the required data.
- **High availability**: Key requirement in a Big Data architecture. The design must be distributed and optionally lean on a cloud solution.
- **Support to Analytical and Content Applications**: analysis can take days and involve several machines working in parallel. These data may be required in part to other applications.
    - **Workflow automation**: full support to creation, organization and transfer of workflows.

Data Management Aspects (2/2)

- **Integration with existing Public and Private Cloud systems**: Critical issue: transfer the entire data. Support to existing cloud environments, would facilitate a possible data migration.
- **Self Healing**: The architecture must be able to accommodate component failures and heal itself without customer intervention. Techniques that automatically redirected to other resources, the work that was carried out by failed machine, which will be automatically taken offline.
- **Security**: The most big data installations are built upon a web services model, with few facilities for countering web threats, while it is essential that data are protected from theft and unauthorized access.

NoSQL Definition

- “Not Only SQL”.
- **Definition**: Name for a subset of structured storage software, designed for increased optimization for high-performance operations on large dataset.

Why NoSQL?

- ACID (**A**tomicity, **C**onsistency, **I**solation, **D**urability) doesn’t scale well.
- Web apps have different needs: **High Availability**, Low Cost **Scalability** & **Elasticity**, Low Latency, Flexible Schemas, Geographic Distributions.
- Next Generation Databases mostly addressing some of this needs being **non-relational, distributed, open-source** and **horizontally scalable.**

NoSQL Characteristics

PROs:

- schema-free;
- High Availability;
- Scalability;
- easy replication support;
- simple API;
- eventually consistent / **BASE** (not ACID);

CONs:

- Limited query capabilities;
- hard to move data out from one NoSQL to some other system (but 50% are JSON-oriented);
- no standard way to access a NoSQL data store.

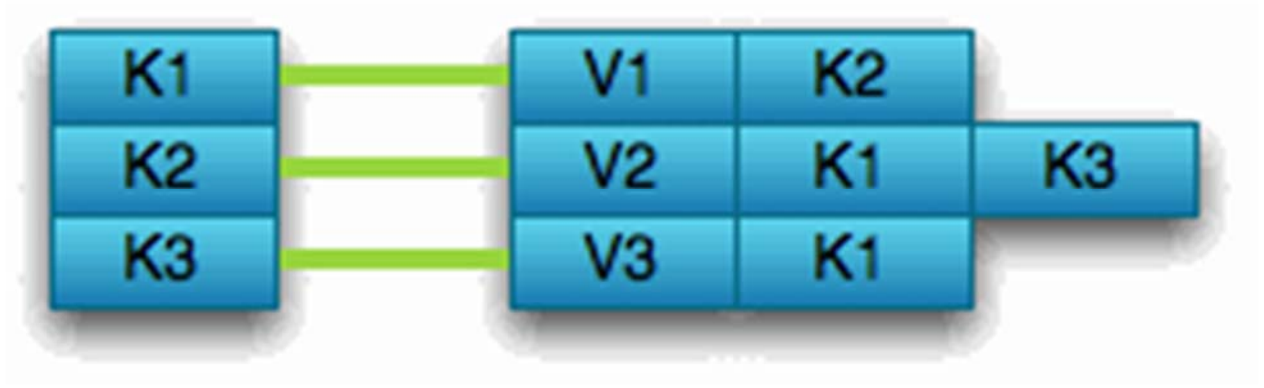
Eventually consistent

- A consistency model used in distributed computing;
- The storage system guarantees that if no new updates are made to the object, eventually (after the inconsistency window closes) all accesses will return the last updated value;
- BASE (**B**asically **A**vailable, **S**oft state, **E**ventual consistency) approach;
- Known under the moniker of **Optimistic Replication**.

Types of NoSQL Database

- Key-Value DB
- Col- Family/Big Table DB
- Document DB
- XML DB
- Object DB
- Multivalue DB
- ACID NoSQL

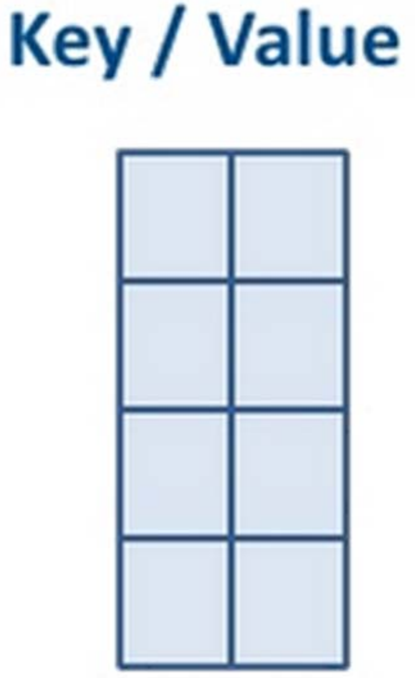
Key-Value Database (1/2)

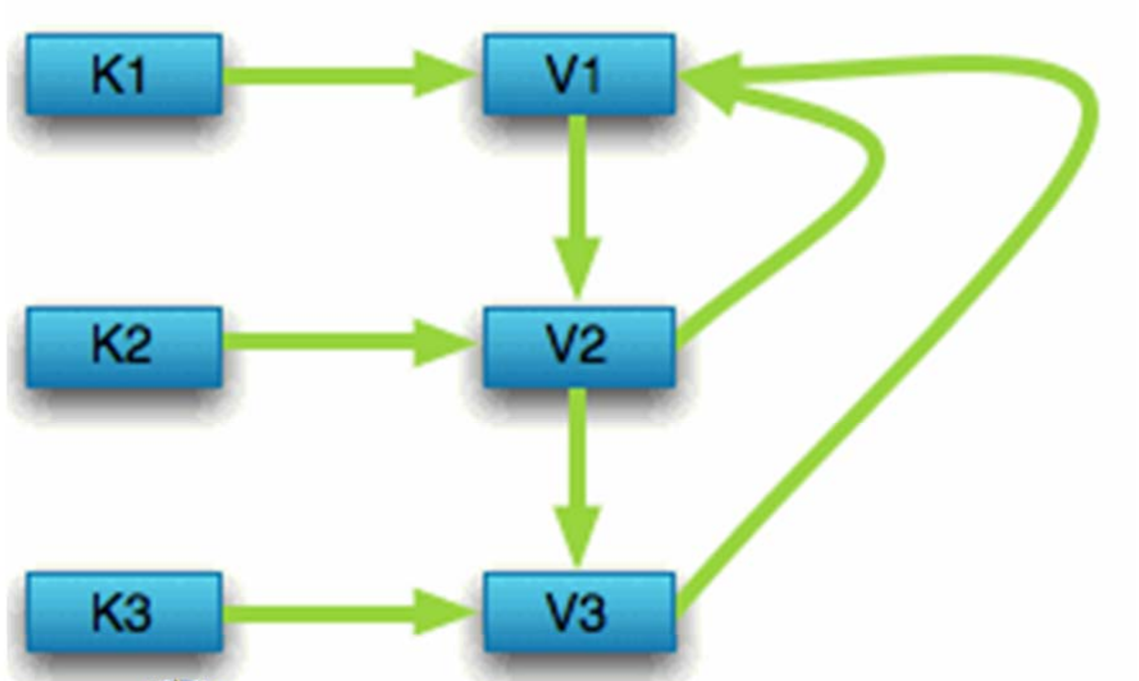


- High scalable database, not suitable for large data sets.
- Solution that allows to obtain good speed and excellent scalability.
- Different permutations of key-value pairs, generate different database.
- This type of NoSQL databases are used for large lists of elements, such as stock quotes.

Key-Value Database (2/2)

- Data Model: collection of kay-value pairs
- Based on Amazon’s Dynamo Paper
- Example: Dynomite, Voldemort, Tokyo



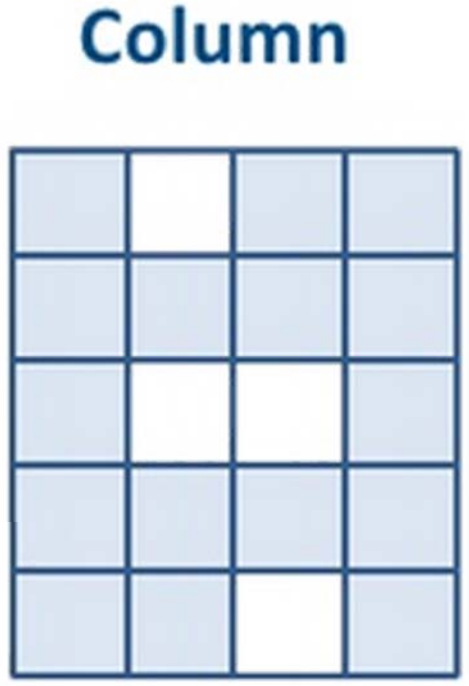


Column Family/Big Table Database (1/2)

- Key-value stores can be column-family database (group columns);
- Most advanced type where keys and values can be composed;
- Very useful with time series or with data coming from multiple sources, sensors, device and website, with high speed;
- They require good performance in reading and writing operations ;
- They are not suitable for datasets where data have the same importance of relationship.

Column Family/Big Table Database (2/2)

- Based on Google BigTable Paper
- Data Model: Big Table, Column Family
- Example: Hbase, Hypertable



Row-oriented

1,Smith,Joe,40000;

2,Jones,Mary,50000;

3,Johnson,Cathy,44000;

→

Column-oriented

1,2,3;

Smith,Jones,Johnson;

Joe,Mary,Cathy;

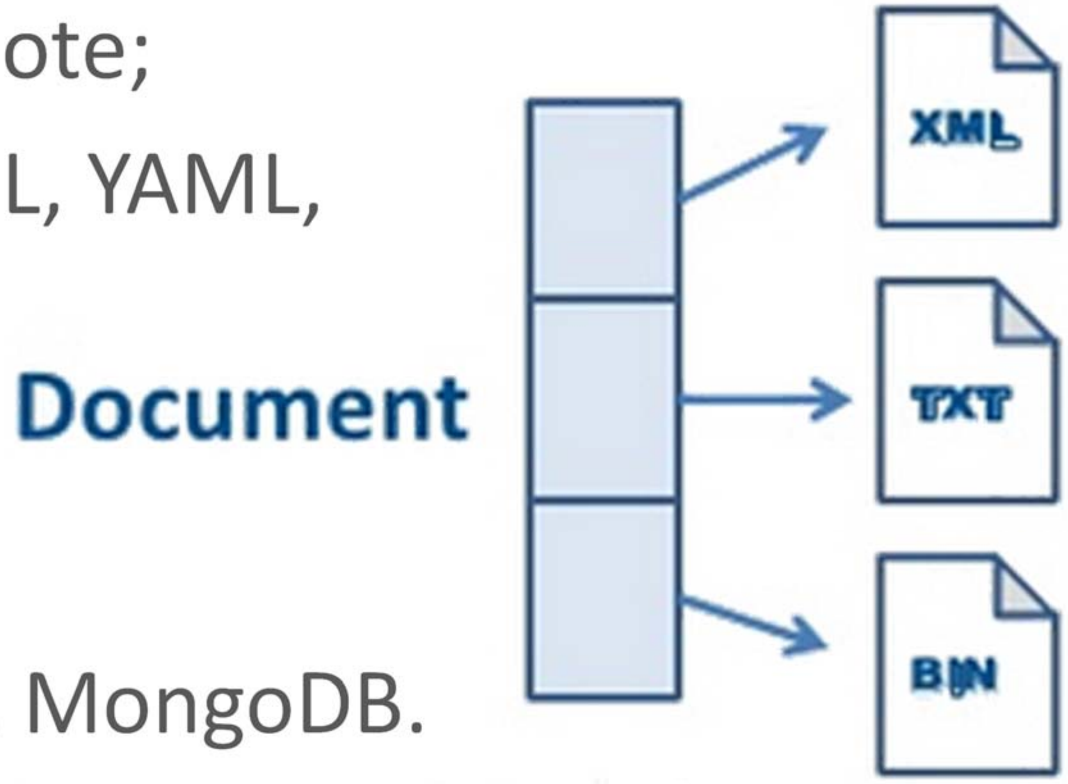
40000,50000,44000;

Document Database (1/2)

- Designed for storing, retrieving, and managing semi-structured data.
- Fit perfectly to the OO programming
- Have the same behavior of key-value, but the contest of this document is understood and taken into account.
- Useful when data are hardly representable with a relational model due to high complexity
- Used with medical records or with data coming from social networks.

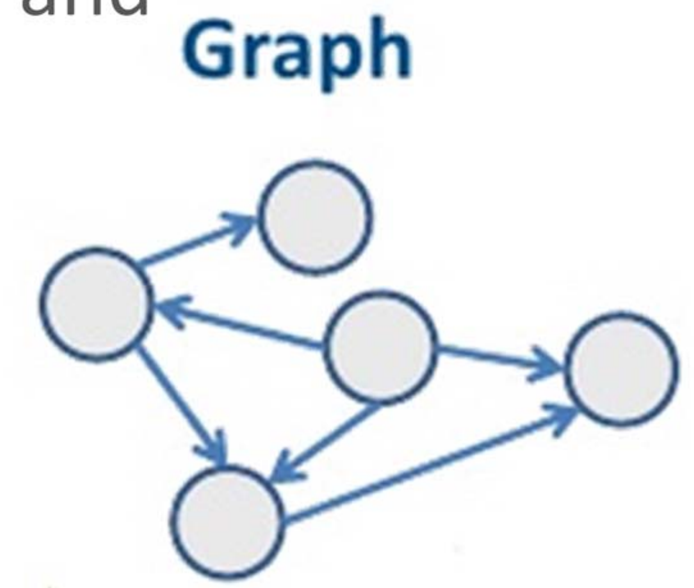
Document Database (2/2)

- Has become the most popular type of NoSQL;
- Data Model: Collection of K-V collection;
- Inspired on Lotus Note;
- Used encoding: XML, YAML, JSON, e BSON;
- Example: CouchDB, MongoDB.



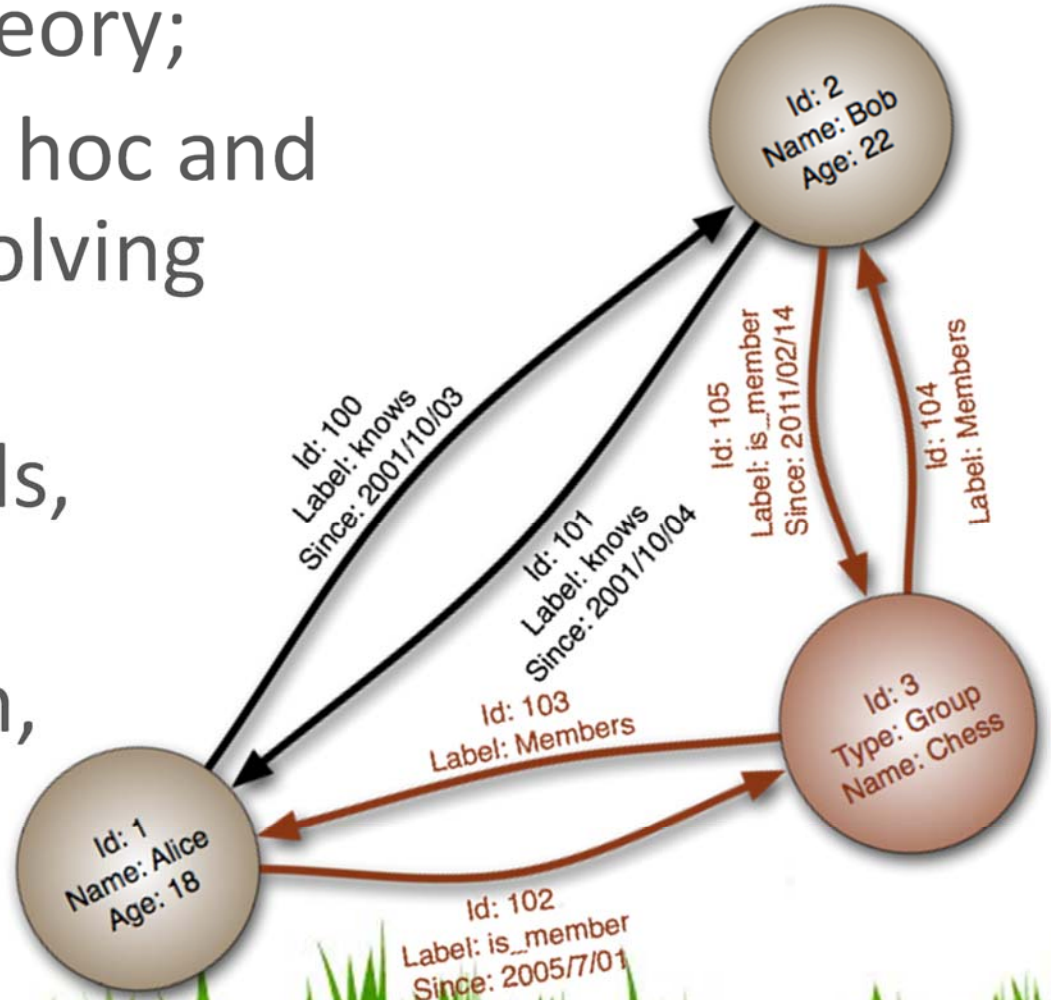
Graph Database (1/2)

- They take up less space than the volume of data with which they are made and store a lot of information on relationship between data.
- Used when dataset is strongly interconnected and not tabular.
- The access model is transactional and suitable for all those applications that need transactions.
- Used in field like geospatial, bioinformatics, network analysis and recommendation engines.



Graph Database (2/2)

- Not easy execute query on this database type;
- Inspired by Graph's Theory;
- Suitable to manage ad hoc and changing data with evolving schemas;
- Data Model: nodes, rels, K-V on both;
- Example: AllegroGraph, Objectivity.



Object Database

- Allow the creation of very reliable storage system.
- It should be a DBMS, and it should be an object-oriented system;
- They directly integrate the object model of programming language in which they were written (not much popularity);
- They cannot provide a persistent access to data, due to the strong binding with specific platforms;
- Pros: persistence, secondary storage management, concurrency, recovery and an ad hoc query facility;
  - Example: Objectivity, Versant.

XML Database

- Large XML document optimized to contain large quantities of semi-structured data;
- They are usually associated with document-oriented databases;
- The data model is flexible;
- Queries are simple to perform, but not very efficient;
- XML-enabled (map XML to traditional database), Native XML (uses XML documents as the fundamental unit of storage);
- Example: eXist, BaseX.

Multivalue Database (1/2)

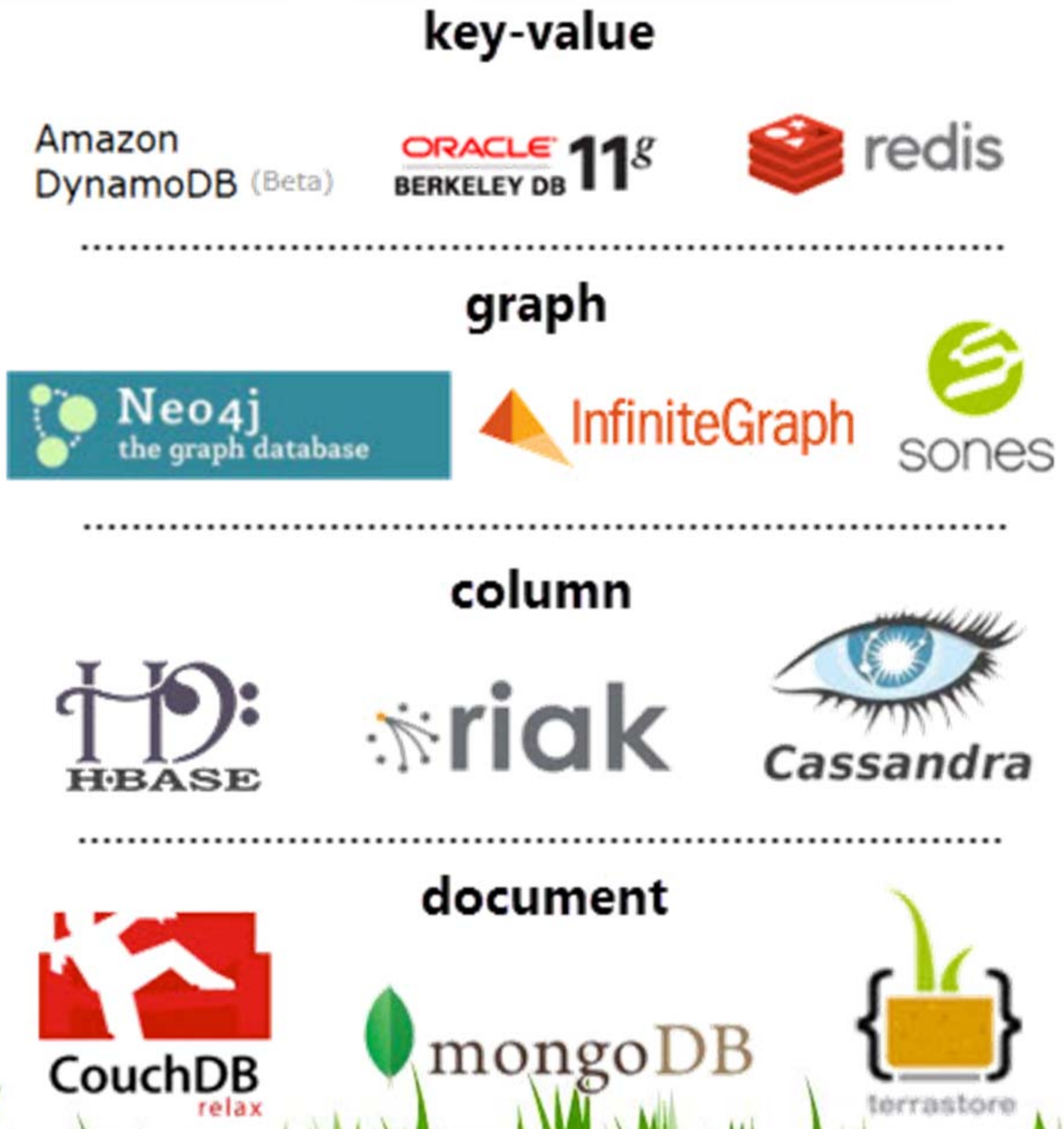
- Synonymous with Pick Operating System;
- Support use of attributes that can be a list of value, rather than a single value (RDBMS);
- The data model is well suited to XML;
- They are classified as NoSQL but is possible to access data both with or without SQL;
- They have not been standardized, but simply classified in pre-relational, post-relational, relational and embedded.

Multivalue Database (2/2)

- Database = “Account”, Table = “File”, Column = “Attribute” (row) or “Dictionary“ (trasformed row).
- In the “person” File there is a Dictionary called “emailAddress” where we can store a variable number of email address values in the single record.
- Data Model: simil relational
- Example: OpenQM, Jbase.

Products Overview

Most used NoSQL Database



ACID NoSQL

- Try to combine the more desirable features of NoSQL databases and RDBMS
- **FoundationDB** and **OracleNoSQL**: distributed, replicated key-value store with a shared-nothing architecture;
- All reads and writes in **FoundationDB** provide ACID guarantees (Atomic, Consistent, Isolated, and Durable);
- **OracleDB** supports atomic operations on the same key, and allows atomic transactions on sets of keys that share the same major key path.

MongoDB (www.mongodb.org)

- Open source NoSQL;
- Use of JSON document;
- Built-in replication (high availability);
- Auto-Sharding (horizontal scalability);
- **Document database** with dynamic schemas (flexible index support and rich queries);
- Developed to achieve high performance in reading and writing, ability to be easily scalable with automatic failover.

MongoDB Features (1/4)

- **Query ad hoc:** support for search by field, range queries, regular expression searches that return specific fields of documents or user defined function;
- **Indexing:** include support for many types of indexes on any field in the document; indices are conceptually similar to RDBMS ;
- **Replication:** Master-Slave, Master can perform reads and writes, Slave copies data from the master and can only be used for reads or backup;

MongoDB Features (2/4)

- **Load balancing**: can run over multiple servers, balancing the load and/or duplicating data to keep the system up in case of hardware failure;
- **File storage:** could be used as a file system (GridFS), files can be distributed and copied multiple times between machines transparently;
- **Aggregation:** built-in MapReduce framework (not based on Hadoop) used for batch processing of data and aggregation operations (GROUP BY);

MongoDB Features (3/4)

- **Capped collections**: support for fixed-size collections that maintains insertion order and behaves like a circular queue.
- **In Place document update:** disk space reserved to store document at the same place (< disk write and disk allocation cost);
- **MMAP**: maps the whole data to memory;
- **Global lock**: for write or multiple reads (less efficient concurrency);

MongoDB Features (4/4)

- **Document Search:** flexible and power search command to find the document by its inner attribute (that need to be indexed);
- **Sharding:** each secondary node has a full data of primary node. Add more sharding partition later is difficult and time consuming;
- **JSON format**: data are not stored in rows and columns, but in a binary form of JSON documents (BSON). Schema can vary across documents and change quickly;

MongoDB index type (1/2)

- **Unique Indexes:** reject inserts of new documents or the update of a document with an existing value for the field for which the unique index has been created;
- **Compound Indexes:** useful for queries that specify multiple predicates (customers example);
- **Array Indexes:** for fields that contain an array, each array value is stored as a separate index entry (recipes example);
- **Geospatial Indexes:** to optimize queries related to location within a two dimensional space;

MongoDB index type (2/2)

- **Time To Live (TTL) Indexes**: allow users to specify a period of time after which the data will automatically be deleted (example users clickstream);
- **Sparse Indexes:** smaller, more efficient indexes when fields are not present in all documents (flexibility);
- **Text Search Indexes:** specialized index for text search that uses advanced, language-specific linguistic rules for stemming, tokenization and stop words.

MongoDB Advantages

- **In-memory performance with on-disk capacity**. MongoDB makes extensive use of RAM to speed up database operations; all data is read and manipulated through memory-mapped files; data that is not accessed is not loaded into RAM.
- **Flexibility**. MongoDB provides tremendous flexibility to the development thanks to document data model, multi-datacenter deployments, tunable consistency, operation-level availability options.

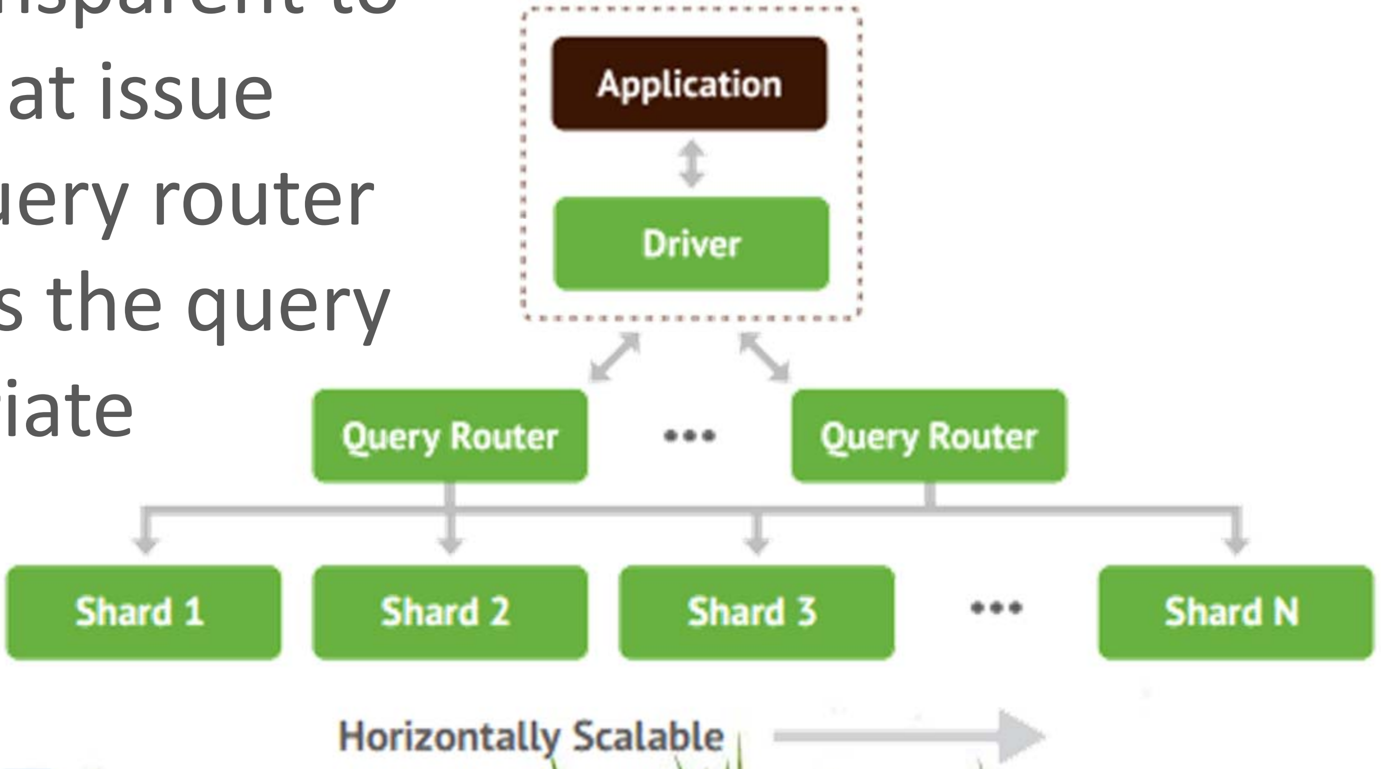
MongoDB Advantages

- **Simple and rich query model**. Can be defined different types of indices that allow to query data in many ways; this feature makes MongoDB fit for a wide variety of applications.
- **Different Type of Sharding.** To reduce limitations of a single server (bottlenecks in RAM or disk I/O) without adding complexity. Range-based Sharding (“close” key), Hash-based Sharding (MD5 hash), Tag-aware Sharding (user configuration).

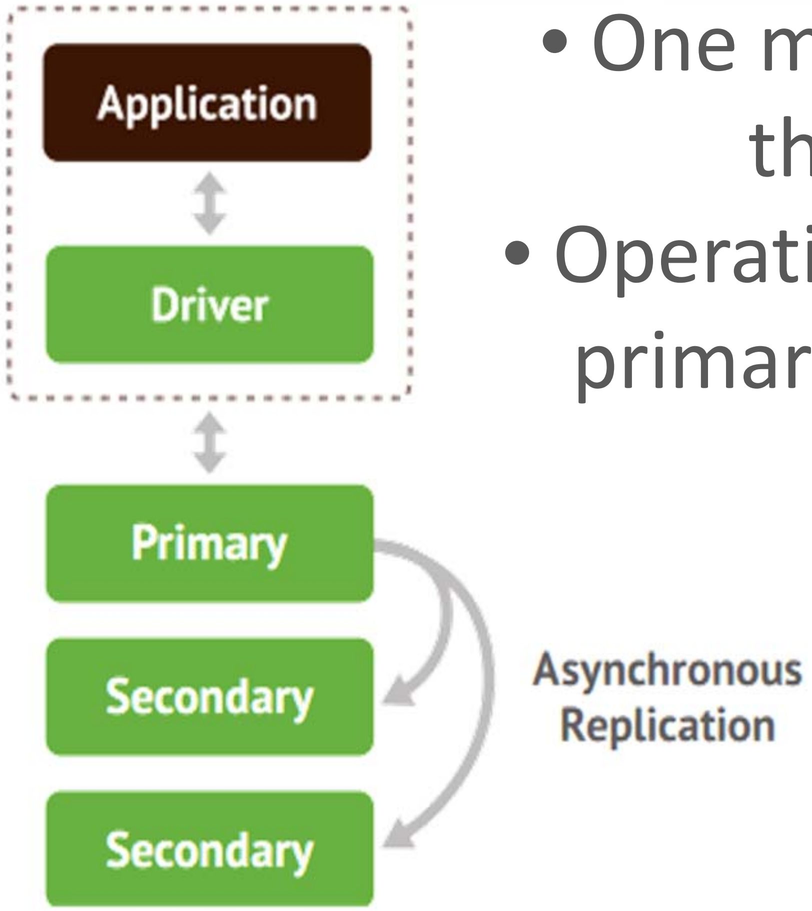
MongoDB – Automatic Sharding

- Automatically balances the data in the cluster as the data grows or the size of the cluster increases /decreases

- Sharding is transparent to applications that issue queries to a query router that dispatches the query to the appropriate shards.



MongoDB – Easy Replication



- One member acts as the primary replica, the others act as secondary replicas.
- Operations that modify a database on the primary are replicated to the secondaries with a log (oplog).
- Reads and writes are issued to a primary copy; if it fails, one of the secondary members is elected.
- More replicas increase data durability, reduce operational overhead and improve system availability.
- Configurable number of replicas

MongoDB Queries Type

- **Key-value queries:** return results based on any field in the document (primary key);
- **Range queries:** return results based on values defined as inequalities (>, < or =);
- **Geospatial queries:** return results based on proximity criteria, intersection and inclusion as specified by a point, line, circle or polygon;
- **Text Search queries:** return results in relevance order based on text arguments (AND, OR, NOT);
- **Aggregation Framework queries:** return aggregations of values returned (GROUP BY);
- **MapReduce queries:** execute complex data processing (JavaScript) across data.

Query example MongoDB (1/2)

Select All Documents in a Collection

- db.inventory.find()

Exact Match on an Array

- db.inventory.find({ tags: ['fruit', 'food', 'citrus'] })

Specify AND Conditions

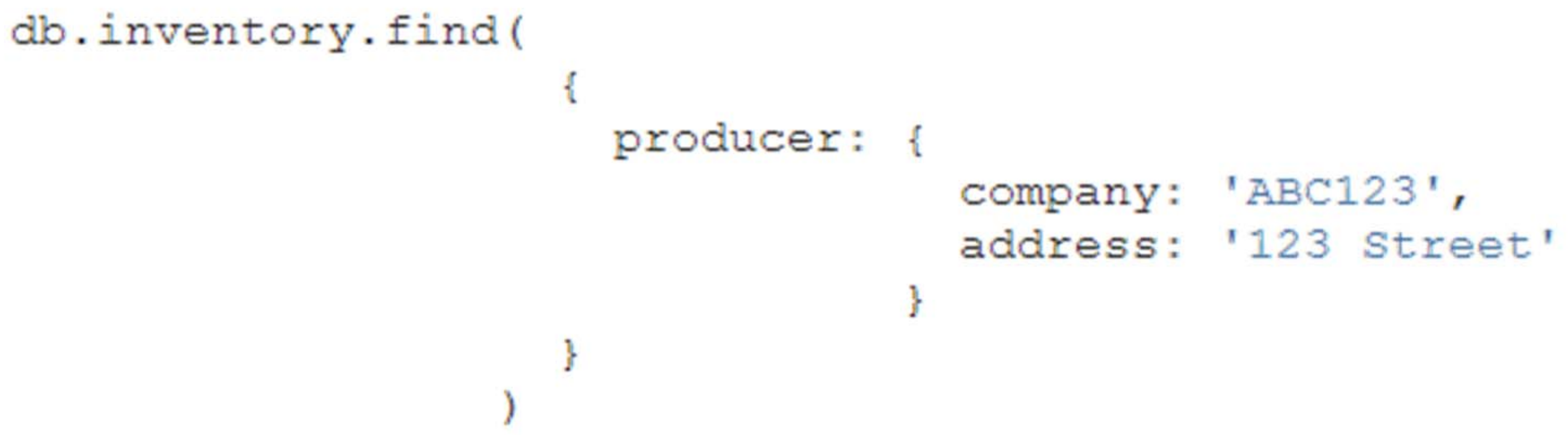
- db.inventory.find({ type: 'food', price: { $lt: 9.95 } })

Equality Match on Fields within Subdocument

- db.inventory.find({ 'producer.company': 'ABC123' })

Query example MongoDB (2/2)

Exact Match on Subdocument



```
db.inventory.find(
                   {
                     producer: {
                                 company: 'ABC123',
                                 address: '123 Street'
                               }
                   }
                 )
```

- The db.collection.findOne() method is the db.collection.find() method with a limit of 1.

Architectural Aspects

Index

Architectural aspects (1/2)

- **Clustering size**: the number of nodes in each cluster, affects the completion times of each job: a greater number of nodes, correspond to a less completion time of each job.
- **Input data set**: increasing the size of the initial data set, the processing time of the data and the production of results, increase.
- **Data node**: greater computational power and more memory, is associated with a shorter time to completion of a job.
- **Date locality**: it is not possible to ensure that the data is available locally on the node. we need to retrieve the data blocks to be processed, and the time of completion be significantly higher.

Architectural aspects (2/2)

- **Network**: The network affects the final performance of a Big Data management system; connections between clusters make extensive use during read and write operations. Highly available and resiliency network, which is able to provide redundancy.
- **Cpu**: more processes to be carried out are CPU-intensive, greater will be the influence of the CPU power on the final performance of the system.
- **Memory**: For applications memory-intensive is good to have the amount of memory on each server, able to cover the needs of the cluster. 2-4GB of memory for each server, if memory is insufficient performance would suffer a lot.

Riak (basho.com/riak)

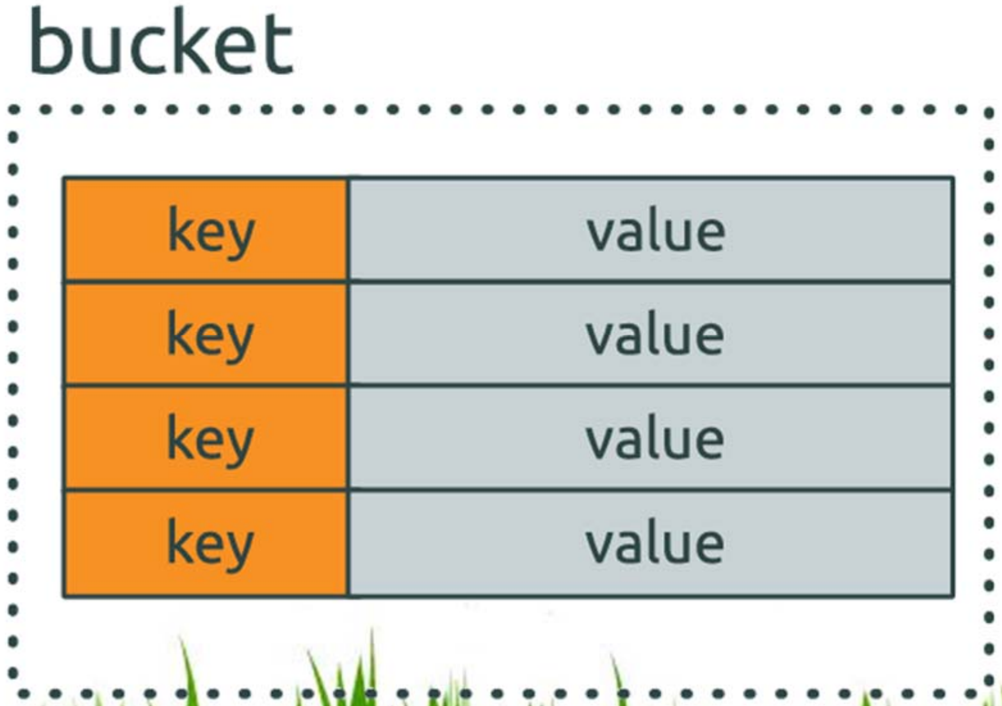
- Open source NoSQL;
- **Key/Value DB** implementing the principles from Amazon's Dynamo paper;
- **Masterless** system (eventually consistent);
- data is automatically distributed across nodes using **consistent hashing**;
- **Riak Control**, an open source graphical console for monitoring and managing Riak clusters;
- **RiakCS** (cloud storage built on a distributed database Riak).

Riak Features (1/4)

- **Scalability:** data is rebalanced automatically with no downtime, when add/remove machine; data is automatically distributed around the cluster and yields a near-linear performance increase as capacity is added.
- **Availability:** a neighboring node will take over write and update responsibilities for a node becoming unavailable.
- **Operational Simplicity**: Add new machines to a Riak cluster is easily without larger operational burden.

Riak Features (2/4)

- **Simple Data Model:** key/value pairs stored in flat namespace (bucket). All object are stored on disk as binaries. Developing code for this model is simpler and more efficient; perfect for applications that require rapid interactions;
- **Masterless design:** any node can serve any incoming request, because all data is replicated across multiple nodes;



Riak Features (3/4)

- **MapReduce:** allows operation like filtering documents by tag, counting words in documents, extracting links to related data (javascript support);
- **Riak Search**: distributed full-text search engine that provide support for various MIME type and robust querying (exact matches, wildcards, range queries, proximity search).
- **Secondary Indexing (2i):** each object can be tagged with 1 ore more queryable values, integers or strings (exact matches, range queries).

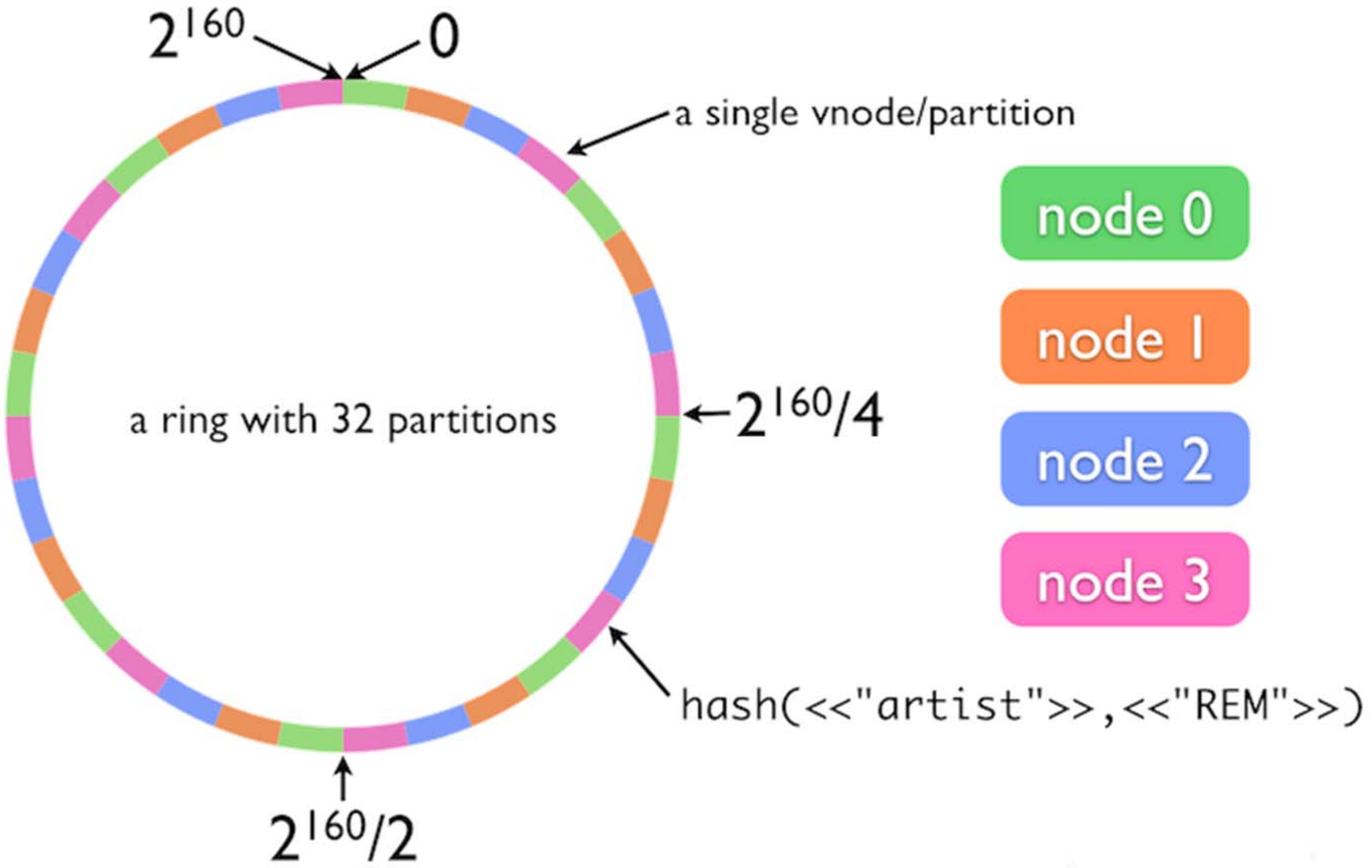
Riak Features (4/4)

- **Fault-Tolerance:** due to network partition or hardware failure, access can be lose to many nodes without losing data;
- **Consistent hashing:** ensures data is distributed *evenly* around the cluster. New nodes can be added with automatic, minimal reshuffling of data.
- **Active anti-entropy:** self-healing property in background; it uses a hash tree exchange to compare replicas of objects and automatically repairs/update any divergence.

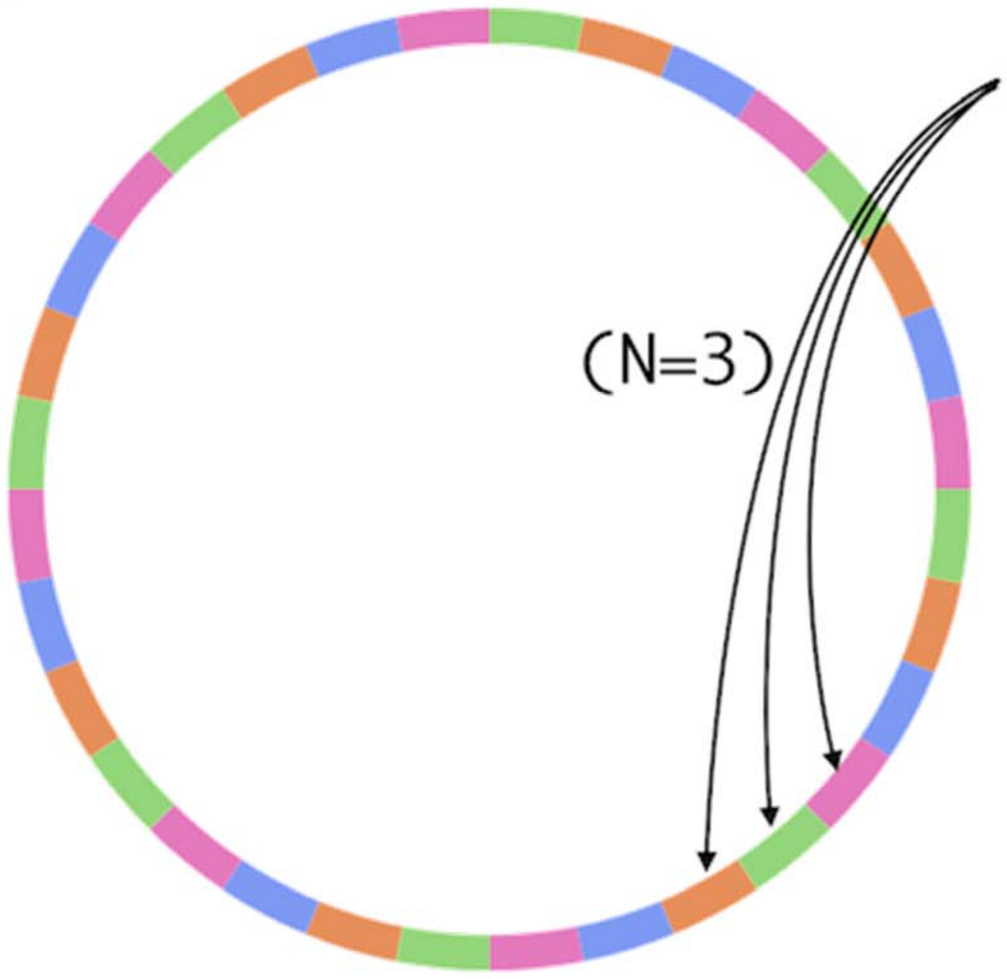
Riak – Consistent Hashing (1/3)

- Makes possible the **automatic redistribution** of data across nodes;
- Ensures data is evenly distributed;
- Bucket and key combination is hashed to an **hash maps** onto a 160-bit integer space (ring)
- The ring is used to determine **what data** to put on **which** physical **machines**
- The integer space is divided into **equally-sized partitions**. Each partition correspond to a **range of values** on the ring, and is responsible for all buckets/keys couple that, when hashed, fall into that range.

Riak – Consistent Hashing (2/3)



Riak – Consistent Hashing (3/3)



```
put(<<"artist">>,<<"REM">>)
```

- Each partition is managed by a virtual node
- Set a replication number, “n”: when a key is mapped onto a given partition, data will automatically replicated onto the next n-1 partitions.

Riak - Adding/removing node

- A running node can be added to an existing cluster. To stage a join request:

```
riak-admin cluster join riak@192.168.2.2
```

- If successful, node receives the **new cluster state** and starts claiming partitions until even distribution (primary replica for the partition);
- It recalculates a new cluster state and **gossips** it to a random node;
- **Partition handoff** starts to transferring data from existing nodes to the new one (already ready to serve requests).

Riak - Vector Clock

- A method for keeping track of which version of a value is current;
- When a value is stored, it is **tagged** with a vector clock, that it is **extended** for each update;
- **Auto-repair** out-of-sync data.
- Enable clients to **always write** to the database in exchange for consistency conflicts being resolved at read time by either application or client code.
- It can be configured to store copies of a given datum based on size and age.
- It can be disable to fall back to simple **time-stamp** based “last-write-wins”.

Riak - Conflict Resolution

Conflict:

- Two simultaneous writes from clients with the **same vector clock** value; the object is given two **siblings**;
- Writes to an existing object **without** a vector clock (missing).

Resolution:

- Automatically with an **application**;
- by presenting the conflicting objects to the end user. (VectorClock -> Siblings -> Update with the current value).

Riak – Query data (1/2)

Primary key: (GET, PUT, DELETE, UPDATE)

MapReduce:

- Adds the capability to perform **more powerful queries** over stored data;
- Spread the processing to take advantage of **parallel processing power**;
- Dividing the query into **several steps**, dividing the dataset into **several chunks**, and then running those **step/chunk pairs** on separate physical hosts;
- **More efficient** to take the computation to the data than to bring the data to the computation (10 KB).

Riak – Query data (2/2)

Secondary Indexes:

- To find data based on terms **other** than bucket/key pair
- Allows **streaming** and **pagination** of results.

Riak Search:

- Support various **MIME types** (JSON, XML, plain text, Erlang) for automatic data extraction;
- **Scoring** and **ranking** for most relevant results;
- **Exact match queries** (Wildcards, Range queries, Grouping, Prefix matching, Proximity searches, Term boosting);

→ Search queries as **input** for MapReduce jobs.

Riak - Query example

- Riak Search via **command line** tool:

  bin/search-cmd search books "title:\"See spot run\""

- **HTTP Solr**-compatible interface:

  http://hostname:8098/solr/select

  Parameter: index=INDEX, q=QUERY, df=FIELDNAME, start=N, rows=N, wt=FORMAT, sort=FIELDNAME

- **Wildcard** Searches: “bus*” “bus?”
- **Proximity** Searches: "See spot run"~20
- **Range** Searches: "field:{red TO rum}"

Access/Data Rendering Aspects

Index

Access and Data Rendering aspects

- **User Access:** thanks to an **Access Management System**, is possible to define **different types of users** (normal users, administrator, etc.) and assign to each, access rights to different parts of the Big data stored in the system.
- **Separation of duties:** using a combination of **authorization**, **authentication and encryption**, may be separate the duties of different types of users.

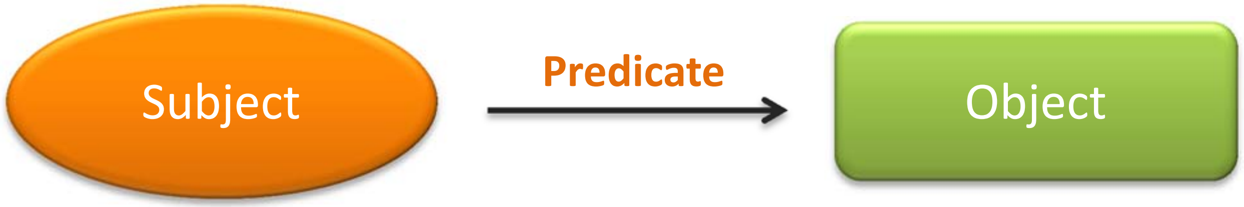
These separation provides a strong contribution to safeguard the privacy of data, which is a fundamental feature in some areas such as health/medicine o government.

Access and Data Rendering aspects

- **Efficient Access**: defining of standard interfaces (specially in business and educational application), managing **concurrencies issues**, **multi-platform and multi-device** access it is possible not decrease data's availability and to improve user experience.
- **Scalable Visualization:** because a query can give a **little or enormous set of result**, is important a **scalable display tools**, that allow a clear vision in both cases (i.e. 3D adjacency matrix for RDF)

RDF-HDT Library

- **RDF** is a standard for describing resources on the web, using statements consist of **subject** - **predicate** - **object**
- An **RDF model** can be represented by a **directed graph**



- An RDF graph is represented physically by a serialization

RDF/XML - N-Triple - Notation3

RDF-HDT Library - Scalability drawbacks in Web of Data

Publication

- No Recommendations/methodology to publish at large scale
- Use of Vocabulary of Interlinked Data and Semantic Map

Exchange

- Main RDF format (RDF/XML – Notation3 – Turtle…) have a **document-centric view**
- Use of universal compressor (gzip) to reduce their size

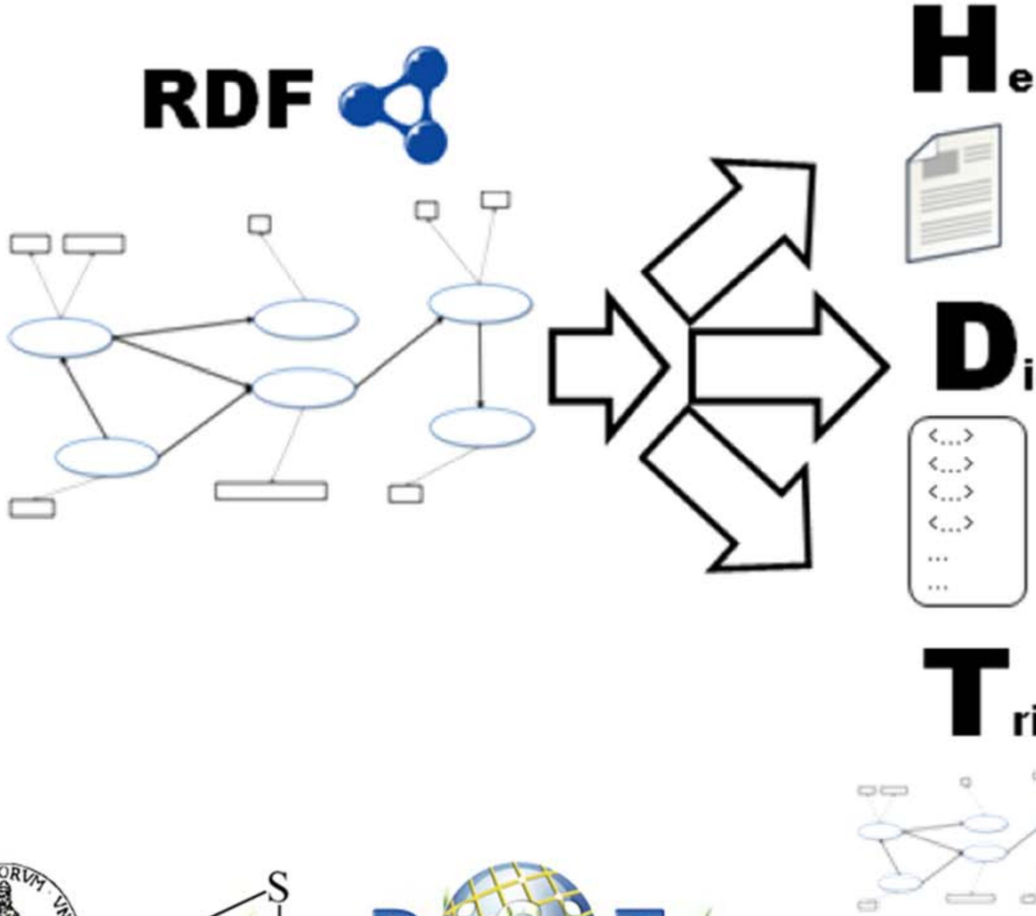
Consumption (query)

- Costly Post-processing due to **Decompression** and **Indexing** (RDF store) issues

RDF-HDT Library

- **HDT** (Header, Dictionary, Triples) is a **compact data representation** for RDF dataset to reduce its verbosity.
- RDF graph is represented with 3 logical components.
  - <u>**H**eader</u>
  - <u>**D**ictionary</u>
  - <u>**T**riples</u>
- This makes it an ideal format for storing and sharing RDF datasets on the Web.

RDF-HDT Library



Header — Logical and physical metadata describing the RDF data set. It serves as an entrance point to the information.

Dictionary — Mapping between elements in the data set and unique IDs, thus contributing to compactness.

Triples — Structure of the data after the ID replacement, in a compressed form.

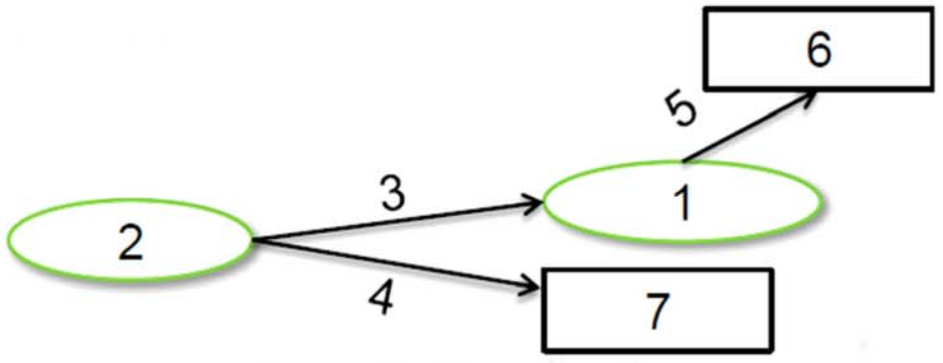
RDF-HDT Library - Header

- HDT improve the value of metadata;
- Header is itself an RDF graph containing information about:
  - **provenance** (provider, publication dates, version),
  - **statistics** (size, quality, vocabularies),
  - **physical organization** (subparts, location of files)
  - **other types of information** (intellectual property, signatures).

RDF-HDT Library – Dictionary + Triples

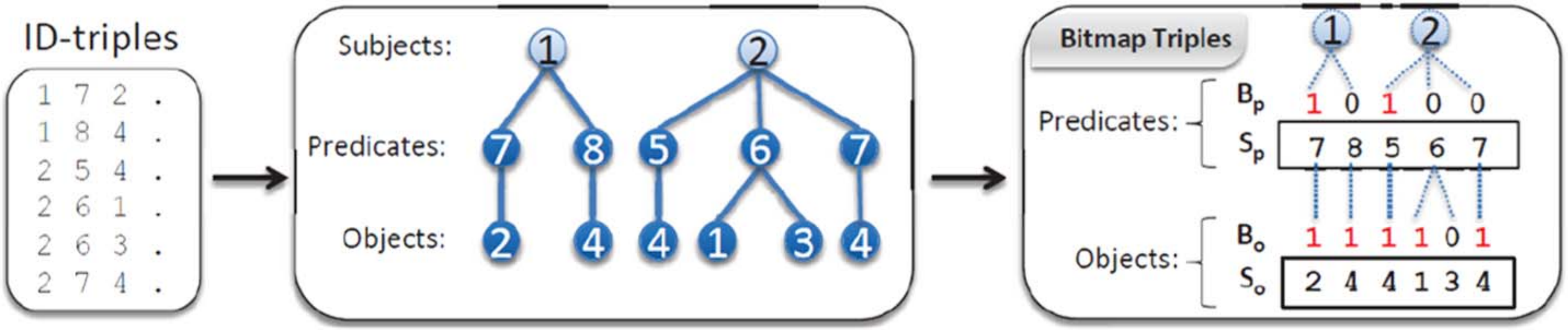
- **Mapping** between **each term** used in a dataset and **unique IDs.**
- **Replacing** long/redundant **terms** with their **corresponding IDs.** So, Graph structures can be indexed and managed as integer-steam.
- **Compactness** and **consumption performance** with an advanced dictionary serialization.

1	<http://books/author33>
2	<http://books/book21>
3	dc:author
4	dc:title
5	foaf:name
6	"Pablo Neruda"
7	"Spain in the Heart"



RDF-HDT Library – Triples

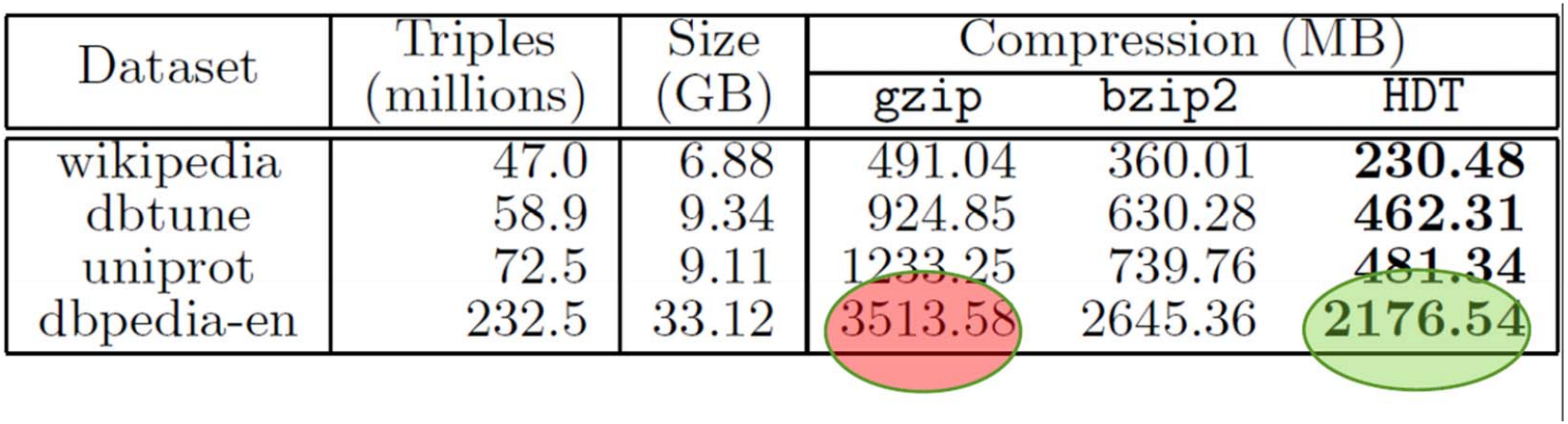
- These **ID-triples** component **compactly** represent **the RDF graph.**
- **ID-triple** is the key component to accessing and querying the RDF graph.
- **Triples component** can provide a succinct index for some basic queries.



optimizes space - provides efficient performance in primitive operations

RDF-HDT Library - Pro

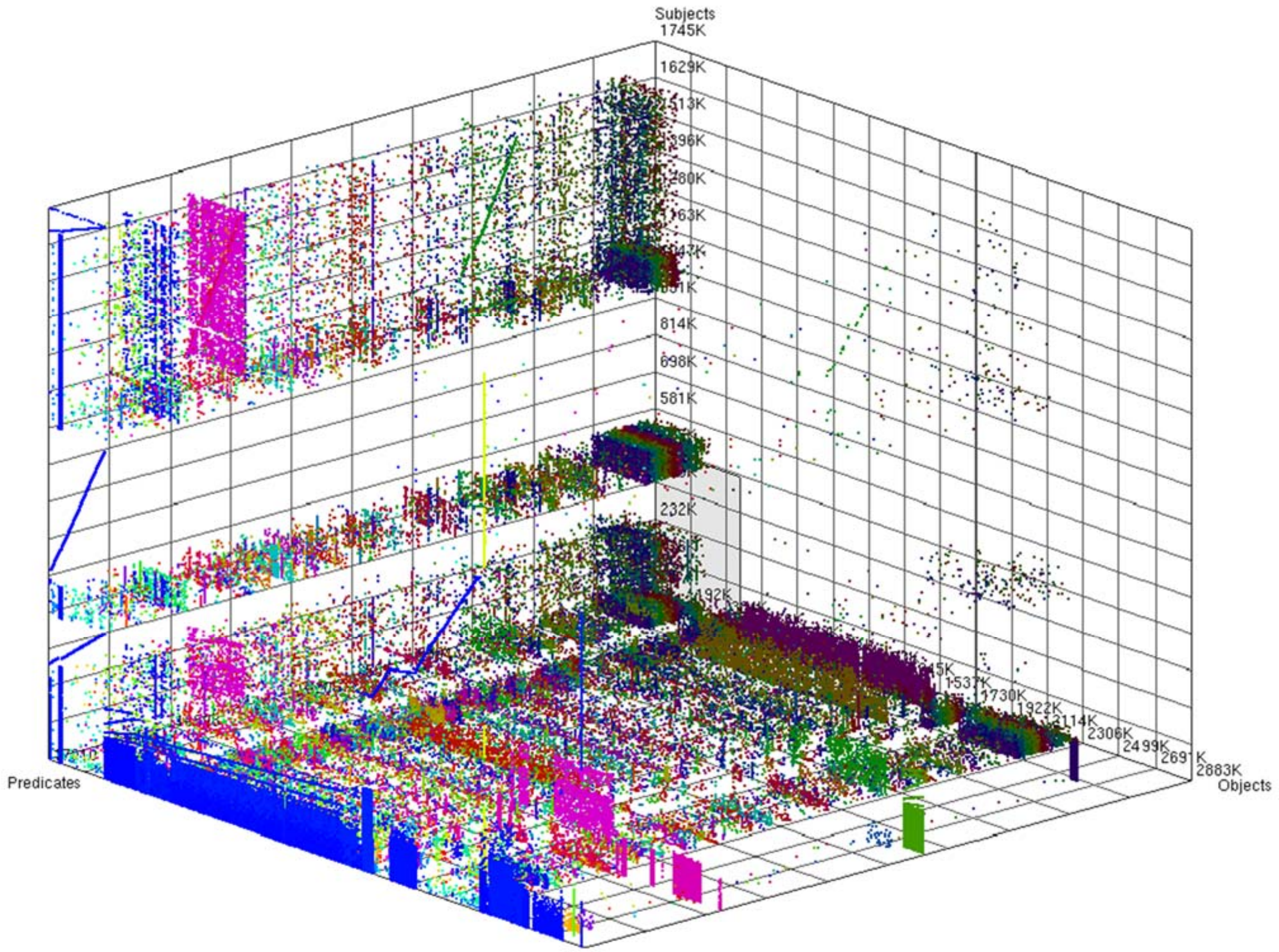
- HDT for exchanging



Dataset	Triples (millions)	Size (GB)	Compression (MB)		
			gzip	bzip2	HDT
wikipedia	47.0	6.88	491.04	360.01	**230.48**
dbtune	58.9	9.34	924.85	630.28	**462.31**
uniprot	72.5	9.11	1233.25	739.76	**481.34**
dbpedia-en	232.5	33.12	3513.58	2645.36	**2176.54**

RDF-HDT Library

HDT-it!



Part 7

Data Analysis and Mining/Ingestion Aspects

Index

Mining and Ingestion

- Mining and Ingestion are two key features in the field of big data, in fact there is a tradeoff between
  - The speed of data ingestion
  - The ability to answer queries quickly
  - The quality of the data in terms of update, coherence and consistency.
- This compromise impacting the design of any storage system (i.e. OLTP vs OLAP).
- For instance, some file-systems are optimized for reads and others for writes, but workloads generally involve a mix of both these operations

Mining and Ingestion

- **<u>Type of indexing:</u>** to speed data ingest, records could be written to the **cache** or apply advanced **data compression techniques**, meanwhile the use of a **different method of indexing** can improve speed of data retrieval operations at only cost of an increased storage space.
- **<u>Management of data relationship</u>**: in some context, data and **relationships between data**, have the same importance. Furthermore new types of data, in the form of highly interrelated content, need to manage multi-dimensional relationships in real-time. A possible solution it is to store relationships in apposite data structures which ensure good ability to access and extraction in order to adequately support predictive analytics tools.
- **<u>Temporary data</u>**: some kinds of data analysis are themselves big: computations and analyses that create **enormous amounts of temporary data** that must be opportunely managed to avoid **memory problems**. In other cases, however, make some statistics on the information that is accessed more frequently, it is possible to use techniques to create well-defined **cache system or temporary files then optimize the query process**.

Hadoop

- Hadoop is an **open source framework** for processing, storing and analyzing massive amounts of distributed, unstructured data.
- It is designed to **scale up from a single server to thousands of machines**, with a very high degree of **fault tolerance**.

Hadoop

- Hadoop was inspired by MapReduce, a user-defined function developed by Google in early 2000s for indexing the Web.
- Hadoop is now a project of the Apache Software Foundation, where hundreds of contributors continuously improve the core technology;
- **Key Idea**: Rather than banging away at one, huge block of data with a single machine, Hadoop breaks up Big Data into multiple parts so each part can be processed and analyzed at the same time.

Hadoop

Hadoop changes the economics and the dynamics of large scale computing, defining a processing solution that is:

- **Scalable –** New nodes can be added as needed, and added without needing to change data formats, how data is loaded, how jobs are written, or the applications on top.
- **Cost effective –** Hadoop brings massively parallel computing to commodity servers. The result is a sizeable decrease in the cost per terabyte of storage, which in turn makes it affordable to model all data.
- **Flexible –** Hadoop is schema-less, and can absorb any type of data, structured or not, from any number of sources. Data from multiple sources can be joined and aggregated in arbitrary ways enabling deeper analyses than any one system can provide.
- **Fault tolerant –** When you lose a node, the system redirects work to another location of the data and continues processing without missing a beat.

Hadoop main components

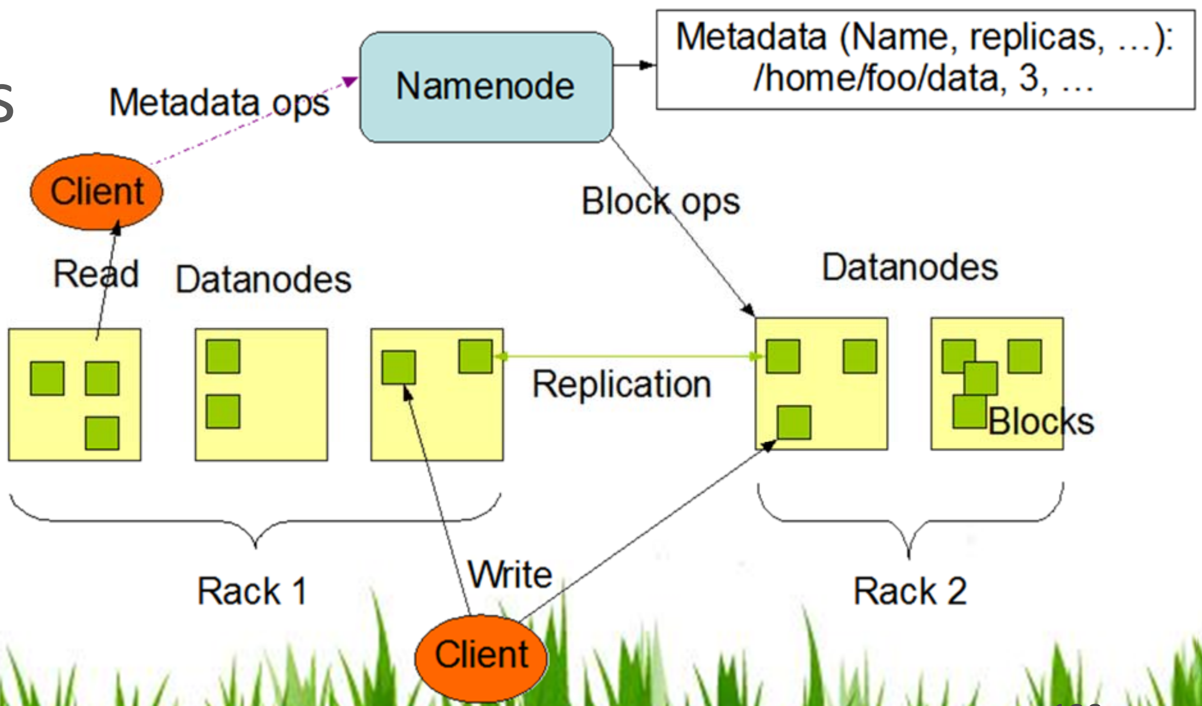
- Hadoop goal is to **scan large data set to produce results** through a **distribute** and **highly scalable** batch processing **systems**.

- Apache Hadoop has two main components**:**
  - **HDFS**
  - **MapReduce**

Hadoop - HDFS

- **HDFS** is a file system that spans all the nodes in a Hadoop cluster for data storage - default block size in **HDFS is 64MB**
- It links together the file systems on many local nodes to make them into one big file system.
- **HDFS** assumes nodes will fail, so it achieves reliability by replicating data across multiple nodes.

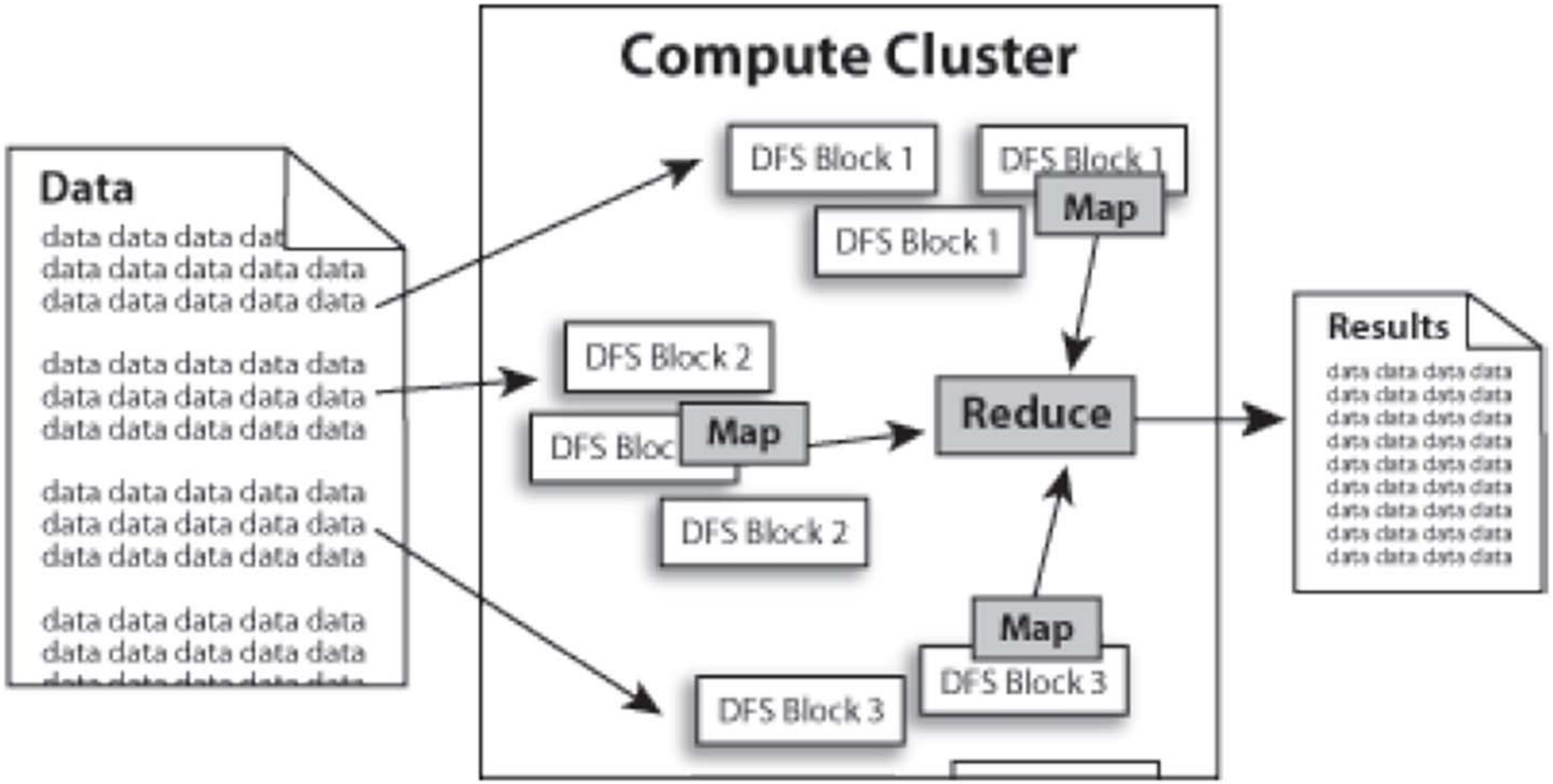
HDFS Architecture



Hadoop - MapReduce

- **MapReduce** is the heart of Hadoop. It is this programming paradigm that allows for **massive scalability** across hundreds or thousands of servers in a Hadoop cluster.
- The **Map function** in the master node takes the input, partitions it into smaller sub-problems, and distributes them to operational nodes.
  - Each operational node could do this again, creating a multi-level tree structure.
  - The operational node processes the smaller problems, and returns the response to its root node.
- In the **Reduce function**, however, the root node, once took the answers of all the sub-problems, combining them to get the answer to the problem it is trying to solve.

Hadoop main features



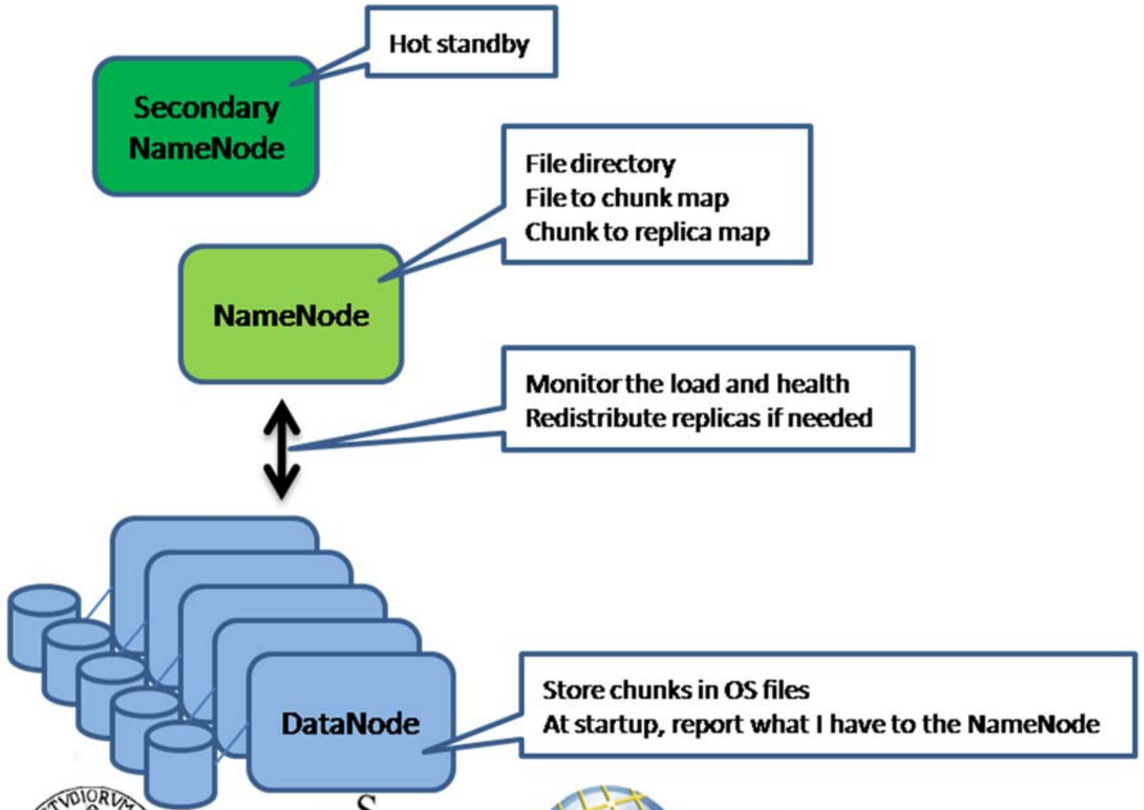
Map-Reduce Paradigm

How Hadoop Works

1. A client accesses **unstructured** and **semi-structured data** from **different sources** including log files, social media feeds and internal data stores.
2. It **breaks the data up into "parts“**, which are then loaded into a file system made up of multiple nodes running on commodity hardware.
3. File systems such as HDFS are adept at storing large volumes of unstructured and semi-structured data as they **do not require data to be organized into relational rows and columns**.

How Hadoop Works

4. Each "part" is **replicated multiple times** and loaded into the file system so that **if a node fails, another node has a copy** of the data contained on the failed node.
5. A **Name Node acts as facilitator**, communicating back to the client information such as which nodes are available, where in the cluster certain data resides, and which nodes have failed.



How Hadoop Works

6. Once the data is loaded into the cluster, it is ready to be analyzed via the MapReduce framework.
7. The client submits a **"Map" job** - usually a query written in Java – to one of the nodes in the cluster known as the **Job Tracker**.
8. The **Job Tracker refers to the Name Node** to determine which data it needs to access to complete the job and where in the cluster that data is located.

How Hadoop Works

9. Once determined, the Job Tracker submits the query to the relevant nodes.
10. **Rather than bringing all the data back into a central location for processing**, processing then occurs at each node simultaneously, or in parallel. This is an essential characteristic of Hadoop**.**
11. When **each node has finished processing** its given job, it stores the results.

How Hadoop Works

12. The **client initiates a "Reduce" job** through the Job Tracker in which **results of the map phase** stored locally on individual nodes **are aggregated to determine the “answer”** to the original query, then loaded on to another node in the cluster.
13. The client accesses these **results**, which can then be loaded into an **analytic environments** for analysis.
14. The **MapReduce** job has now been completed.

Hadoop Stack

- A Hadoop "stack" is made up of a number of components. They include:
  - **Hadoop Distributed File System** (HDFS): The default storage layer in any Hadoop cluster.
  - **Name Node**: The node in a Hadoop cluster that provides the client information on where in the cluster particular data is stored and if any nodes fail;
  - **Secondary Node**: A backup to the Name Node, it periodically replicates and stores data from the Name Node should it fail;
  - **Job Tracker**: The node in a Hadoop cluster that initiates and coordinates MapReduce jobs, or the processing of the data.
  - **Slave Nodes**: The grunts of Hadoop cluster, slave nodes store data and take direction to process it from the Job Tracker.

Hadoop complementary subproject

H·BASE

Cassandra

HIVE

Hadoop Pros & Cons

- **Main Benefit**: it allows enterprises to process and analyze **large volumes** of unstructured and semi-structured data, in **a cost and time-effective** manner.
- Hadoop clusters can scale to **petabytes** (and exabytes) of data; **enterprises can process** and analyze **all** relevant data (no sample data sets).
- Developers can **download** the Apache Hadoop distribution **for free** and begin experimenting with Hadoop in less than **a day**.

Hadoop Pros & Cons

- **Data Scientists** can apply an iterative approach to analysis, **continually refining** and testing queries to uncover previously unknown insights. **Inexpensive** to **get started** with Hadoop.
- **Main downside**: Hadoop and its components are immature and still developing.
- Implementing and managing Hadoop clusters and performing advanced analytics on large volumes of unstructured data, requires significant **expertise**, **skill** and **training**.

Comparison and Analysis of Architectural Features

Index

Examined Product

- ArrayDBMS
- CouchBase
- eXist
- **Hadoop**
- Hbase
- MapReduce
- MonetDB
- Objectivity
- OpenQM
- **RdfHdt**
- RDF3x
- **MongoDB**
- **Riak**

Data Management (1/2)

	ArrayDBMS	CouchBase	Db4o	eXist	Hadoop	HBase	MapReduce	MonetDB	Objectivity	OpenQM	Rdf Hdt Library	RDF 3X	MongoDB	Riak
Data Dimension Huge	100 PB	PB	254G B	2^{31} doc.	10P B	PB	10P B	TB	TB	16TB	100mil Triples(TB)	50mi l Tripl es	PB	TB
Traditional / Not Traditional	NoS QL	NoS QL	NoSQ L	NoS QL	NoS QL	NoS QL	NoS QL	SQ L	XML/ SQL++	NoS QL	NoSQL	NoS QL	NoSQL	NoSQL
Partitioning	blob	blob 20M B	blob	N	chu nk	A	blo b	blo b	blob	Chun k 32KB	(N)	blob	Chunk 16MB	20MB (better 4-6MB)
Data Storage	Mul tidi m. Arra y	1 docu ment /con cept	object datab ase +B- tree	XML docu ment + tree	Big Tabl e	Big Tabl e	N	BA T	Classic Table in which define +model s	1file/ table (data +dict ionar y)	3struct ures,R DF graph for Header	1Tabl e + perm utati ons	Memor y- Mappe d Storag e Engine	modul ar, extensi ble local storage system

Data Management (2/2)

	ArrayDBMS	CouchBase	Db4o	eXist	Hadoop	HBase	MapReduce	MonetDB	Objectivity	OpenQM	Rdf Hdt Library	RDF 3X	MongoDB	Riak
Data Model	Multidimension	Document Store	ObjectDB	XML DB	Column Family	Column Family	CF, KV, OBJ, DOC	Ext SQL	Graph DB	Multivalue DB	RDF	RDF	Document Store	KV document oriented
Cloud	A	Y	A	Y/A	Y	Y	Y	A	Y	N	A	A	Y	Y
Amount of Memory	reduces	documentis+ Bidimensional index	Objects + index	Documents + index		Optimized Use	N	Efficient Use	Like RDBMS	No compression	-50% dataset	Less than data set	More than TDB, but hight compression with Fractal Tree Index	Google Snappy with Level DB
Metadata	Y	Y	Y	Y	Y	Y	Y/A	opt	Y/A	N	Y	Y	Y	Y

Data Management

- Managing of PB
- NoSQL
- Chunk 16MB (BSON)
- Auto-sharding and auto-failover
- Very easy transition from SQL
- Support to Cloud solution
- High compression with Fractal Tree index
- Spatial index and ad-hoc query support

Data Access and visual Rendering| Mining/Ingestion

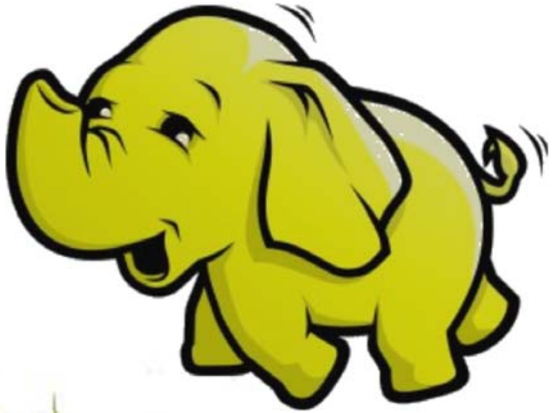
	ArrayDBMS	CouchBase	Db4o	eXist	Hadoop	HBase	MapReduce	MonetDB	Objectivity	OpenQM	Rdf Hdt Library	RDF 3X	MongoDB	Riak
Scalable visual rendering	N	N	N	N	A	N	N	N	N	Y	Y (3d adiacent matrix)	N	A	A
Visual rendering and visualization	N	N	Y	N	A	N	P	N	N	Y	Y	P	A	A
Access Type	Web-interface	Multiple point	Y (Interfaces)	Y (extend XPath sintax)	Sequential	Cache for frequent access	Sequential (no random)	Full SQL interfaces	Y (OID)	Y (concourrency)	Access on demand	SPARQL	http interface	http and solr-like interface

Data Access and visual Rendering| Mining/Ingestion

	ArrayDBMS	CouchBase	Db4o	eXist	Hadoop	HBase	MapReduce	MonetDB	Objectivity	OpenQM	Rdf Hdt Library	RDF 3X	MongoDB	Riak
Type of indexing	Multimensional-index	Y(incremental)	Y	B+-tree (XISS)	HDFS	h-files	Distributed Multilevel-tree Indexing	Hash index	Y (function objectivity/SQL++ interfaces)	B-tree based	RDF-graph	Y (efficient triple indexes)	Single Field, Compound, Multikey, Geospatial, Text, Hashed	Primary key, secodndary index, MapReduce, Fractal Tree
Data relationships	Y	N	Y	Y	A	N	Y	Y	Y	Y	Y	Y	Y	Y

Data Access and visual Rendering| Mining/Ingestion

- External tools for Rendering and Scalable Visualization.
- Sequential Access
- Smart Indexing with HDFS
- External tools for Data Relationship management



Data Analysis

	ArrayDBMS	CouchBase	Db4o	eXist	Hadoop	HBase	MapReduce	MonetDB	Objectivity	OpenQM	Rdf Hdt Library	RDF 3X	MongoDB	Riak
Statistical Support	Y	Y/A	N	N	Y	Y	N	A (SciQL)	Y	A	N	N	Y	Y
Log Analysis	N	N	N	N	Y	P	Y	N	N	N	N	Y	Y	Y
Semantic Query	P	N	P	A	N	N	N	N	N	N	Y	Y	N	N
Statistical Indicator	Y	N	N	N	Y	Y	N	N	Y	A	Y	Y(Performance)	A	Y
CEP (Active Query)	N	Y	N	N	N	N	N	N	N	N	N	N	P	N
Faceted Query	N	N	N	Y	N	A(filter)	Y	N	Y (Objectivity PQE)	N	Y	N	N	Y

Data Analysis

- No Statistical Support
- No Log Analysis
- Support to Semantic & Faceted Query
- Support to Statistical Indicator definition

Architecture (1/2)

	ArrayDBMS	CouchBase	Db4o	eXist	Hadoop	HBase	MapReduce	MonetDB	Objectivity	OpenQM	Rdf Hdt Library	RDF 3X	MongoDB	Riak
Distributed	Y	Y	Y	P (hard)	Y	Y	Y	Y	Y	(-)	A	Y	Y	Y
High Availability		Y	Y	N	Y	Y	Y	N	Y	Y			Y	Y
Fault Tolerance	A	Y	Y	Y	Y	Y	Y	N	Y	N	A	A	Y	Y
Workload	Computation intensitive	Auto distribution		So high for update of enteir files	configurable	Write intensitive	configurable	Read dominated (or rapidly change)	More Possibility	Divided among more processes		optimized	read-dominated	high write and read performance
Indexing Speed	more than RDBMS	non-optimal performance	5 – 10 times more than SQL	High speed with B+Tree	A	Y	Y/A	More than RDBMS	High Speed	Increased speed with alternate key	15 volte RDF	aerodynamic	Changing with differet type of index	low latency request times, high throughput

Architecture (2/2)

	ArrayDBMS	CouchBase	Db4o	eXist	Hadoop	HBase	MapReduce	MonetDB	Objectivity	OpenQM	Rdf Hdt library	RDF3X	MongoDB	Riak
Parallelism	Y	Y	Transactional	Y	Y	Y	Y	N	Y	N	N	Y	Y	Y
N. task configurable	N	Y	N	Y	Y	A	Y	N	Y	N	N	N	N	Y
Data Replication	N	Y	Y	Y	Y	Y	A	Y	Y	Y	A	A	Y	Y

Architecture

- Available for read and write operations, even in failure conditions
- Fault tollerant
- Horizontal scalability
- Very good single-site scalability
- RESTful interface (and HTTP)
- Add new machines to Riak cluster easily
- Automatically distributes data around the cluster

Data Management

	Healthcare	Educational	Energy/Transportation	Social Network Internet Service Web Data	Financial/Business	Security	Data Analysis Scientific Research (biomedical...)
Data Dimension Huge	PB	300PB	TB	1000EB	PB	1Bil TB	PB
Traditional / Not Traditional	NoSQL	SQL	Both	Both	Both	Both	Both
Partitioning	Blob		Y/ Cluster		Cluster		Chuncks
Cloud	Y	P	Y	Y	P	Y	Y
Metadata	Y	Y	Y	Y	N	Y	Y

Data Management

- Exchange of EB
- Involve Traditional / NoSQL DB
- Support to Cloud System
- High use of Metadata

Data Access and visual Rendering| Mining/Ingestion

	Healthcare	Educational	Energy/Transportation	Social Network Internet Service Web Data	Financial/Business	Security	Data Analysis Scientific Research (biomedical...)
Scalable visual rendering	P	Y	N	Y	N	Y	Y
Visual rendering and visualization	Y	N	A	Y	A	N	Y
Access Type	Y (Concurrency)	Standard Interfaces	Get API	Open ad hoc SQL access	A	N	AQL (SQL array version)

Data Access and visual Rendering| Mining/Ingestion

- Possible integration of scalable rendering tools
- Availability of visual reports
- Concurrency Access Type recommended

Data Analysis

	Healthcare	Educational	Energy/Transportation	Social Network Internet Service Web Data	Financial/Business	Security	Data Analysis Scientific Research (biomedical...)
Type of Indexing	N	Ontologies	Key Index Distribuited	Y (Inverted Index, MinHash)	N	N	Array MultiDim
Data relationships	Y	Y	N	Y	Y	N	Y
Statistical Analisys Tools	Y	Y	A	Y	P	N	Y
Log Analysis	Y	Y	A	Y	Y	Y	A
Semantic Query	N	N	N	Y	N	N	N
Statistical indicator	Y	Y	P	N	P	P	Y
CEP (Active Query)	N	N	N	N	N	Y	N
Facet Query	N	Y	N	Y	Y	P	Y

Data Analysis

- Semantics Indexing
- Data Relationship Management
- Benefit with Statistical Analysis Tools
- Benefit with Log Analysis
- High utilization of Faceted Query
- Statistical Indicator relevance

Architecture

	Healthcare	Educational	Energy/Transportation	Social Network Internet Service Web Data	Financial/Business	Security	Data Analysis Scientific Research (biomedical...)
Distributed	Y	P	Y	Y	Y	Y	Y
High reliable	Y	Y	N	N	Y	N	Y
FaultTollerance	N	Y	Y	N	Y	Y	N
Indexing Speed	N	N	N	P	N	N	Y
Parallelism	Y	N	Y	Y	N	N	Y

Architecture

- Distributed Architecture preferred
- Requires High Reliability
- High speed indexing
- Parallel computing required

Comparison Table

	ArrayDBMS	CouchBase	Db4o	eXist	Hadoop	HBase	MapReduce	MonetDB	Objectivity	OpenQM	RdfHdt Library	RDF 3X	MongoDB	Riak
Healthcare	Y		Y		Y	Y	Y	Y	Y	Y			Y	Y
Education				Y	Y								Y	
Energy/Transportation		Y	Y	Y				Y	Y					
Social Network	P	Y			Y	Y	Y	Y			Y	Y	Y	Y
Financial/Business	Y	Y	Y		Y			Y	Y				Y	Y
Security			Y	Y					Y	Y				Y
Data Analysis (Astronomy - Biology - Sociology)	Y		Y	Y			Y	Y	Y	Y	Y	Y	Y	Y

Contact us!

Nadia Rauch: **nadia.rauch@unifi.it**

Mariano Di Claudio: **mariano.diclaudi@unifi.it**

Nadia Rauch

Linked Data

Slide del corso:

Sistemi Collaborativi e di Protezione (Prof.Paolo Nesi)

Corso di Laurea Magistrale in Ingegneria 2012/2013

Index

Open Data

Open Data

- **OD:** types of data freely accessible to everyone, free of patents or other forms of control which limit their reproduction and copyright restrictions.
- **Obligations**: to mention the source, to release any data modification in the same form as the original.
- **Open Government (OG):** public administration should be open to the public, in terms of transparency and direct participation in the decision-making process, through the use of new information technologies and communication.

OD – Why?

- We have the technologies to permit worldwide availability and distributed process of scientific data, broadening collaboration and accelerating the pace and depth of discovery. **Why lock them?**
- The dataset interconnection would result to a data enrichment process and to a consequent increase of their value.
- Obtaining a strong impact on knowledge dissemination.

OD – Pros

- Data **belong** to the human race.
- The data produced by Public Administration (funded by public money) must be **returned to taxpayers** in the form of open data.
- **Restrictions** and re-use of data **limiting** the **progression** of the community.
- The data are needed to **facilitate** the implementation of **common** human **activities**.
- In science, the **rate of discovery** is **accelerated** through a better access to data.
- Scientific open data to obtain the **maximum benefit** from scientific research.

OD – Cons

- If the data will bring commercial benefits to a small number of users, this users should **reimburse** Public Administrations for **the cost** of providing data.
- **Privacy** concerns may require that access to data is limited to specific users or to sub-sets of the data.
- Whoever provides services to collecting, 'cleaning', managing and disseminating data should receive **fair remuneration** (labour and cost-intensive processes).

Where to find Open Data

- OpenData Italiani:

  www.datiopen.it
- OpenData Regione Toscana:

  dati.toscana.it
- OpenData Comune di Firenze:

  opendata.comune.fi.it
- OpenData dell’ISTAT:

  dati.istat.it

And much more….

Data Web VS Semantic Web

- **Data Web**: an intermediate step on the way to the Semantic Web.
- Combines lightweight **knowledge representation techniques** (RDF, RDF-schema, simple ontologies) with traditional **web technologies** (HTTP, REST) for publishing and **interlinking data**.
- Required **comprehensive dataset** and **vocabolaries** to enable the **disambiguation** and **aligment** of other data and information.

Linked Data

Linked Data

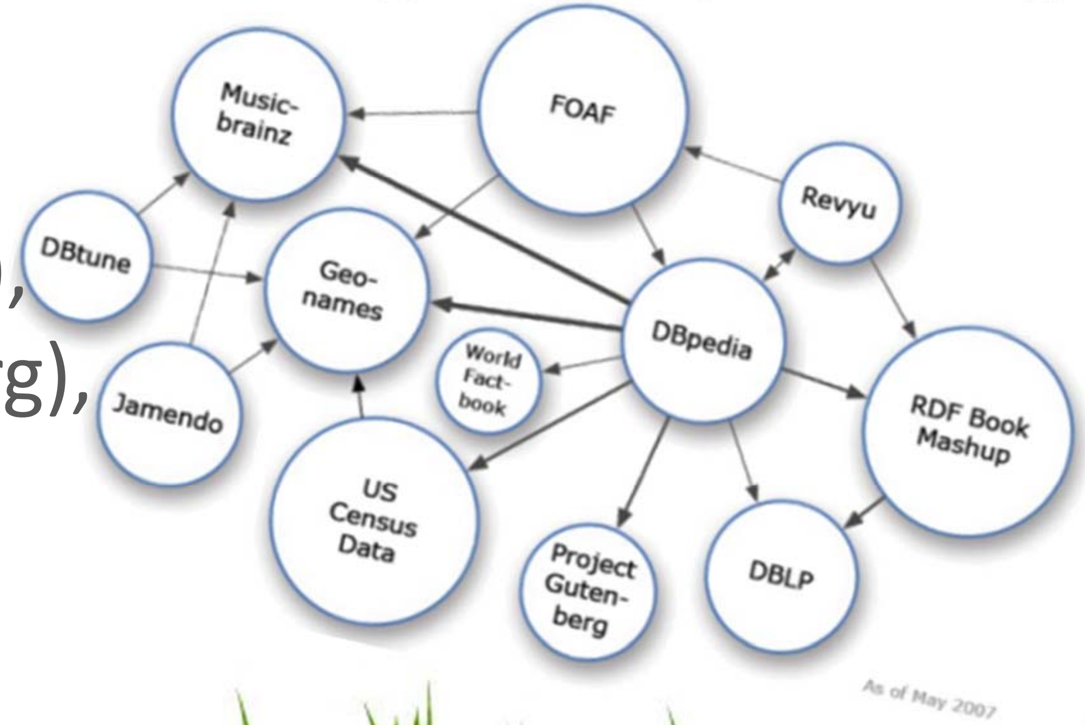
- **Wikipedia** def: a recommended best practice for exposing, sharing, and connecting pieces of data, information, and knowledge on the Semantic Web using URIs and RDF;
- A method of publishing structured data so that it can be interlinked and become **more useful**;
- Web technologies to build: **HTTP**, **RDF** and **URIs**.
- Used to share information in a way that can be **automatically read** by computers.

Linked Data – Best Practice

- Use standars such as **RDF**, **SPARQL**.
- Use **URIs** to denote things.

http://www.dsi.unifi.it/SiiMobility/RT04801501680TO

- Include links to other related things when publishing data.
- Famous dataset: **foaf** (www.foaf-project.org), **Dbpedia** (it.dbpedia.org), **GeoNames** (www.geonames.org)



Linked Data in Italy

- **OpenStreetMap** (2007-2010): freely available Street Map made by the municipal administrations of Merano, Vicenza, Schio and Montecchio Maggiore.
- **www.dati.gov.it** (2011): Open Data Italian portal.
- **www.datiopen.it** (2012): independent initiative that offers free services of signaling, loading and rendering.

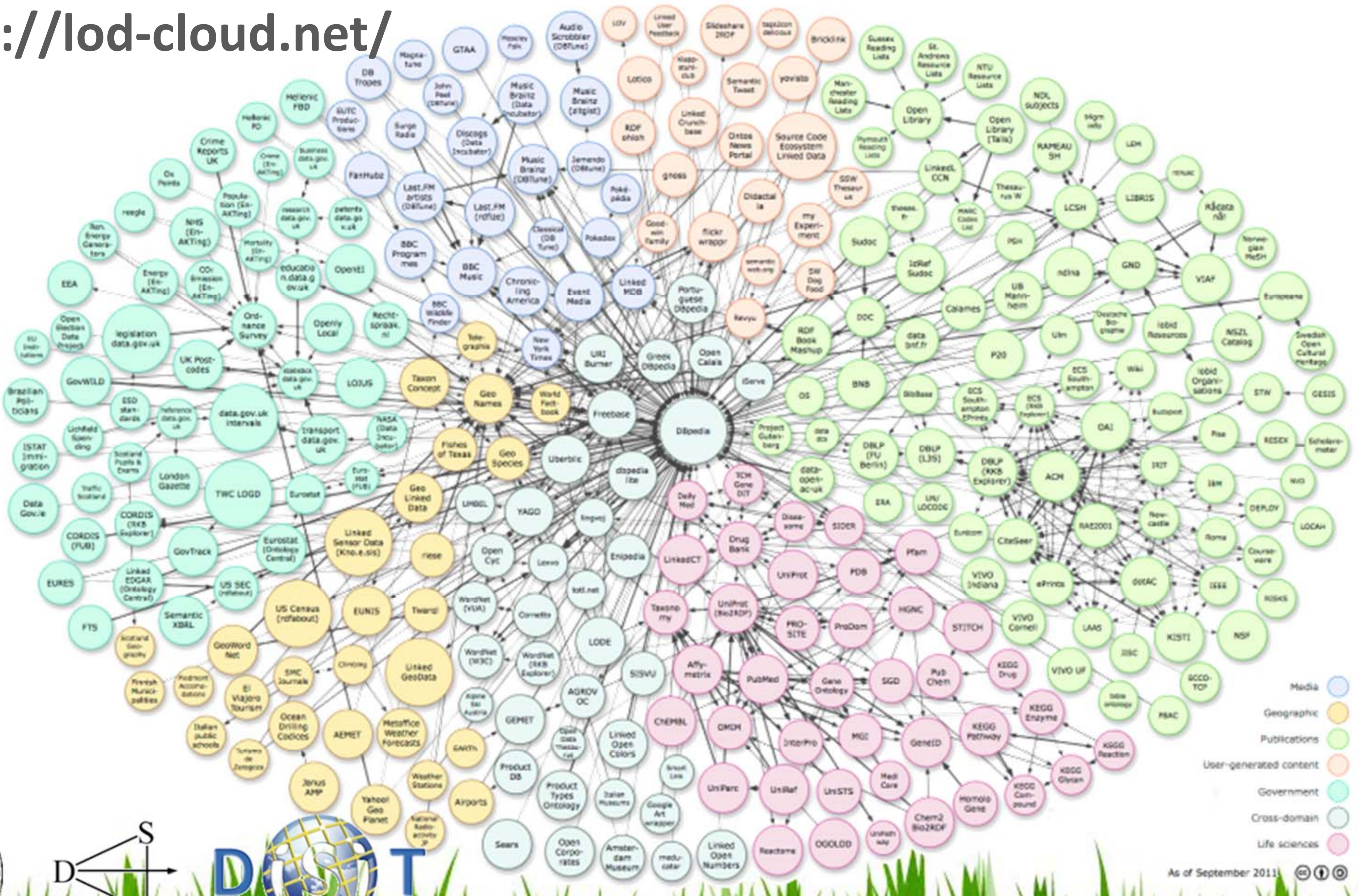
Progetto W3C – Linking Open Data

- The purpose of the **W3C Linking Open Data project** is to extend the Web publishing several open datasets as RDF on the Web and by setting RDF links between data from different resources.
- In **October 2007**, the dataset contained more than **2 billion RDF triples**, connected by more than **2 million RDF links**.
- In **September 2011**, there were **295** dataset with **31 billion RDF triples**, connected by about **504 million RDF links**.

http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData

LOD cloud diagram, by R.Cyganiak, A. Jentzsch

http://lod-cloud.net/



From Open Data to Linked Data the Sii-Mobility example:

http://www.disit.dinfo.unifi.it/siimobility.html

From Open Data to Linked Data

To map Open Data into Linked Data:

1. **Map** the data **to RDF**: selecting/writing a **domain ontology** with standard terminology, convert data to RDF according to this ontology;
2. **Link** to **external** source: find **links** from the **metadata** to **other repositories** (Dbpedia, GeoNames, etc),
3. **Curate** the **Linked Data**: to ensure that the published information/link to other source are accurate.

First step useful tools (Map the data to RDF)

Extract, Transform and Load (ETL): Process used in database and data warehousing that involves:

- **Extracting** data from outside sources;
- **Transforming** it to fit operational needs, which can include quality levels;
- **Loading** it into the end target (database, operational data store or data warehouse).

Useful tools to prepare data to RDF translation

Pentaho Data Integration (Kettle)

- Free, **open source** (LGPL) ETL (Extraction, Transformation and Loading) tool;
- **Powerful** Extraction, Transformation and Loading (ETL) capabilities;
- It use an innovative, **metadata-driven** approach;
- Graphical, **drag and drop** design environment;
- **Scalable**, standards-based architecture;
- K.E.T.T.L.E, a recursive acronym for "Kettle Extraction, Transport, Transformation and Loading Environment".

Pentaho Data Integration (Kettle)

Designed to:

- Collect data from a **variety of sources** (extraction);
- Move and modify data (transport and transform) while cleansing, denormalizing, aggregating and enriching it in the process;
- Frequently (daily) store data (loading) in the final target destination, usually a **large**, dimensionally modelled **database** (or **data warehouse**).

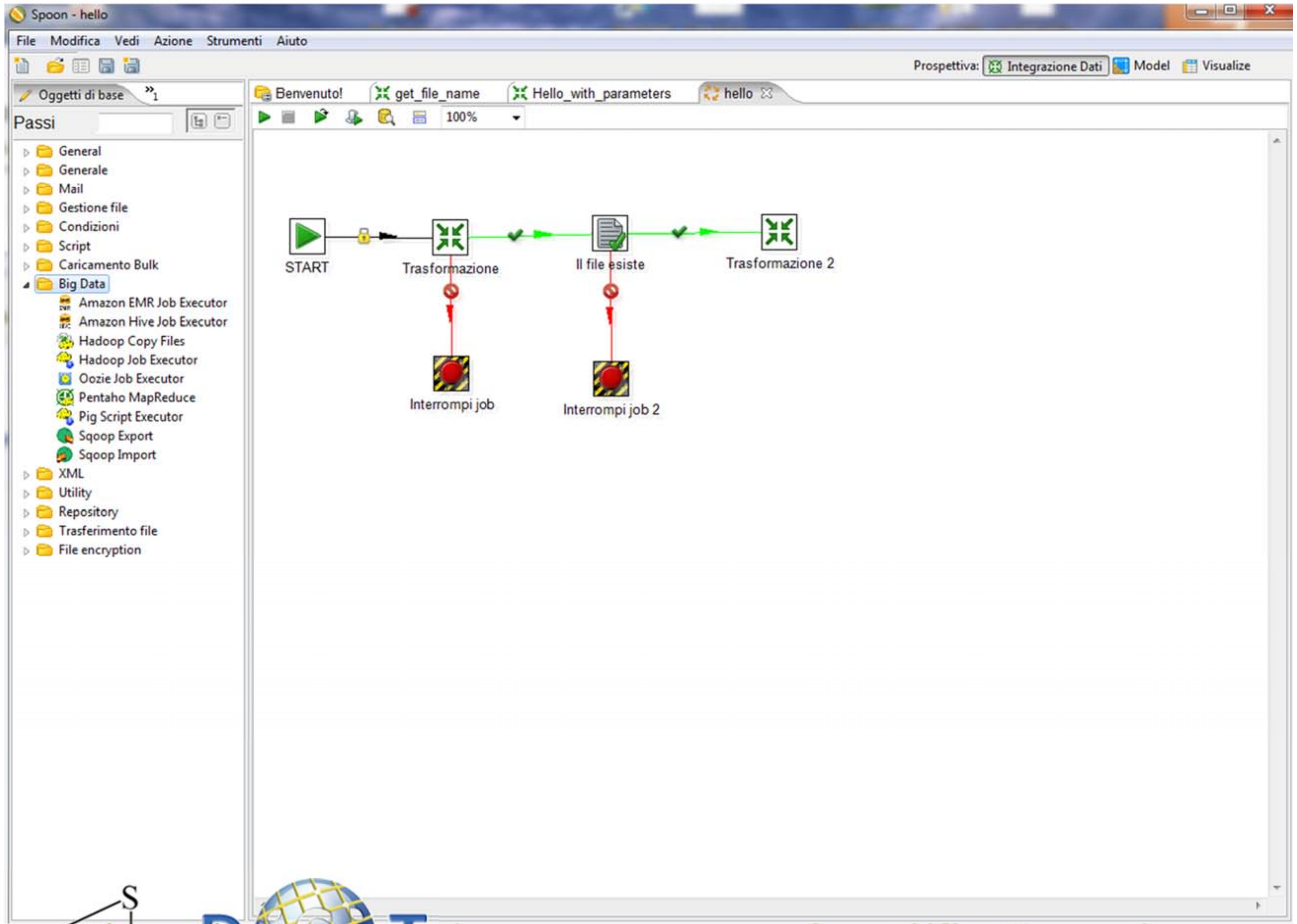
Kettle's 4 main programs

- **Spoon:** graphically oriented end-user tool to model the **flow of data** from input through transformation to output (**transformation**);
- **Pan** is a **command line tool** that executes transformations modelled with Spoon;
- **Chef**: a graphically oriented **end-user tool** used to model **jobs** (transformations, FTP downloads etc. placed in a flow of control);
- **Kitchen** is a **command line tool** to execute jobs created with Chef.

Kettle features

- Interesting feature: Kettle is **model-driven**;
- **Spoon** and **Chef** have a graphical user interface to define the ETL processes on a **high level**;
- **Pan** and **Kitchen** can read and interpret the models created by Spoon and Chef respectively;
- Models can be saved to a particular **XML format**, or they can be stored into a relational database (**repository**);
- Handling many models with repository: models are stored in a structured manner, arbitrary queries can be written against the repository.

Spoon Interface



Spoon Concepts: steps and hoops

- One **step** denotes a particular kind of **action** that is performed **on data**.
- Steps are easily created by **dragging** the icon from the treeview **and dropping** them on the graphical model view.
- Kettle provides a lot of different step types, and can be **extended with plugin**.
- Three different kinds of steps: **input**, **transform**, **output**.

Type of steps in Spoon (1/2)

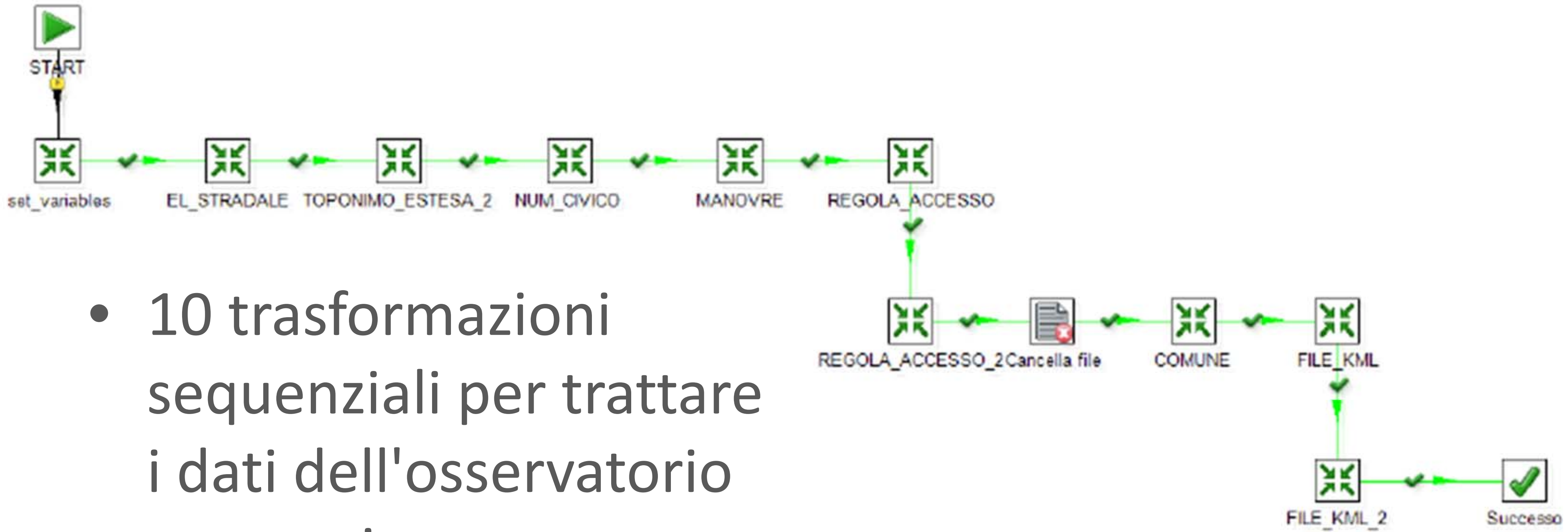
- **Input steps** process some kind of 'raw' resource (file, database query or system variables) and create an outputstream of records from it.
- **Transforming steps** process inputstreams and perform particular action on it (adding new fields/new records); This produce one or more outputstreams. Kettle offers many transformation steps out of the box, very simple tasks (renaming fields) and complex tasks (normalizing data, maintaining a slowly changing dimension in a data warehouse);

Type of steps in Spoon (2/2)

- **Output steps** (the reverse of input steps): accept records, and store them in some external resource (file, database table, etc).

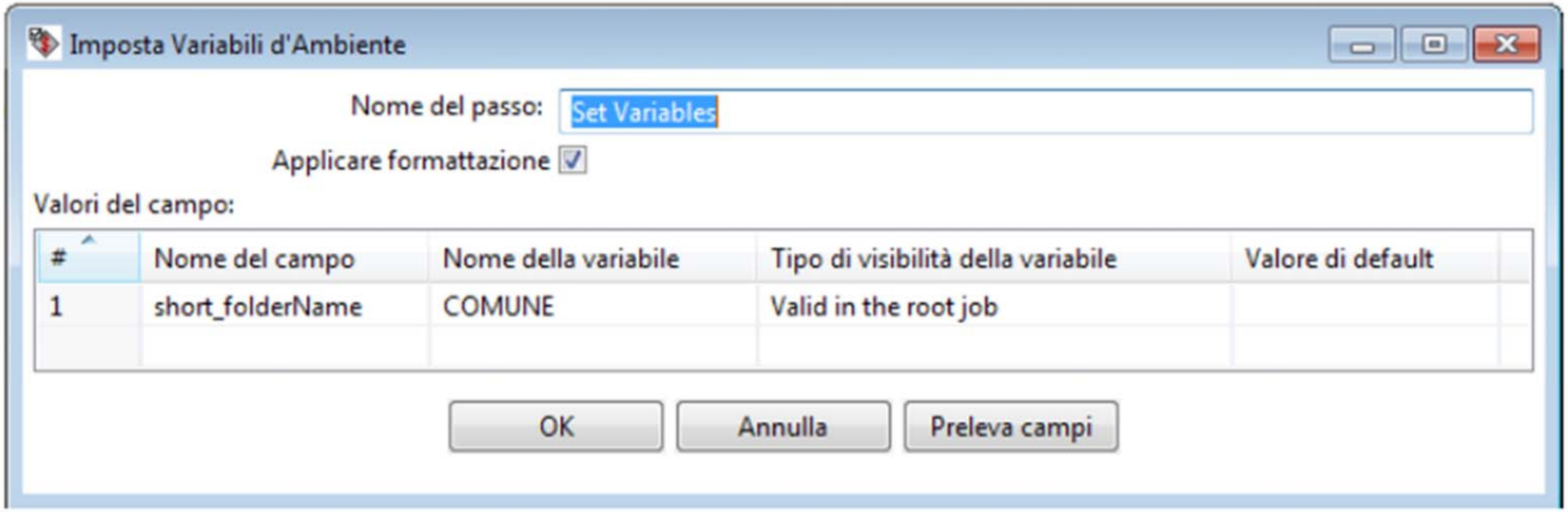
- Connections between the steps are called **hops.**
- Hops between steps behave like **pipelines**: records may flow through them from one step to the other.
- **Main.kjb** is the primary job.

Kettle - SiiMobility Example (1/8)



- 10 trasformazioni sequenziali per trattare i dati dell'osservatorio trasporti.
- Elementi stradali, Toponimi, Numeri Civici, Manovre, Regole di Accesso, Comuni, Giunzioni, Accessi, Cippi chilometrici.

Kettle - SiiMobility Example (2/8)


Imposta Variabili d'Ambiente

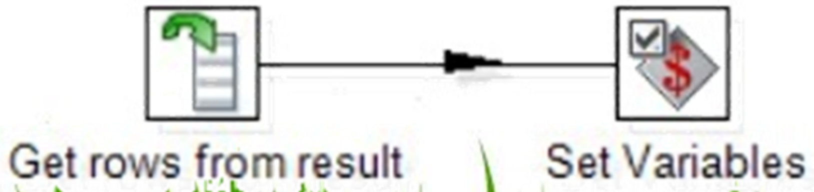
Nome del passo: Set Variables

Applicare formattazione ✓

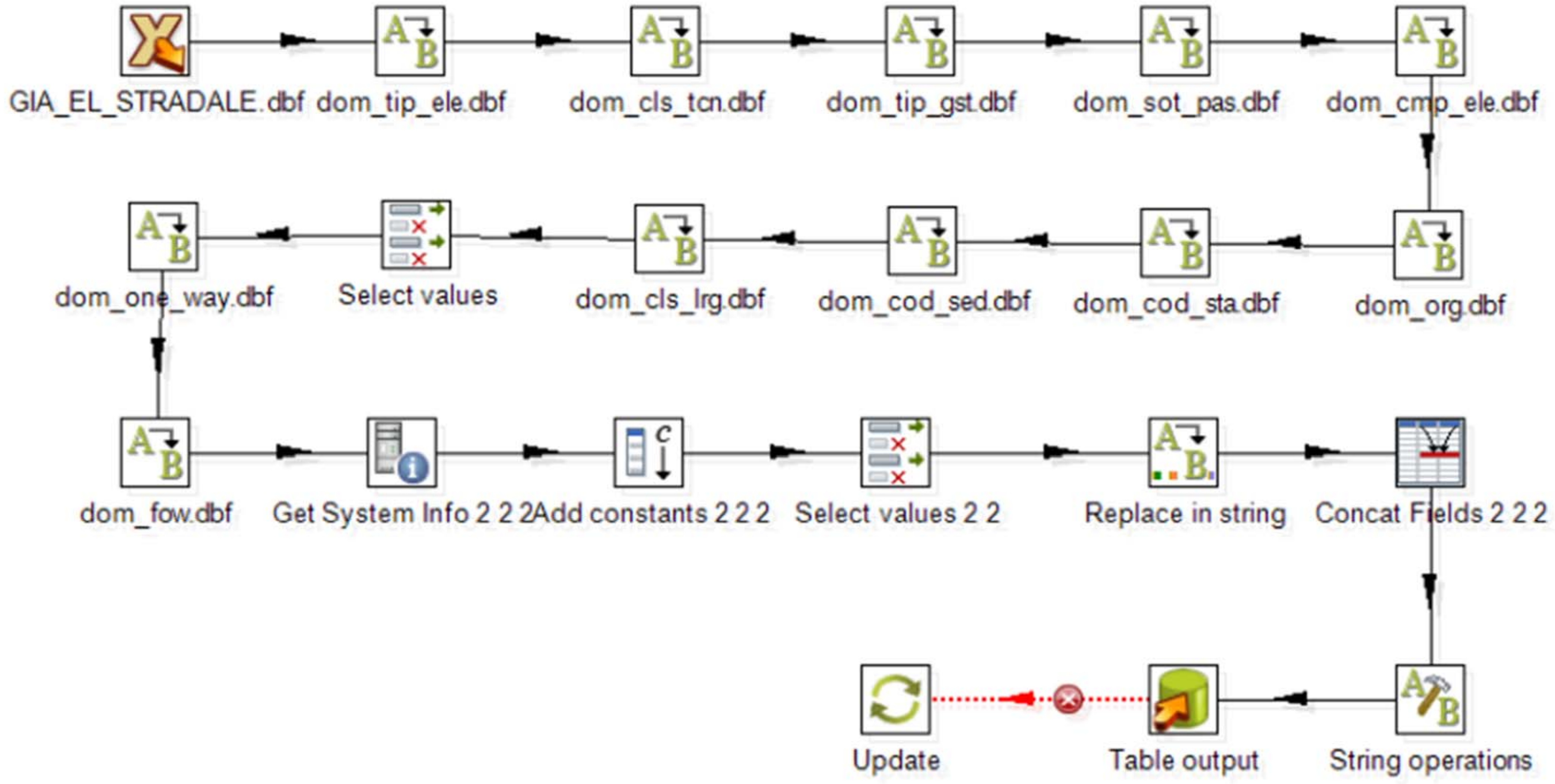
Valori del campo:

#	Nome del campo	Nome della variabile	Tipo di visibilità della variabile	Valore di default
1	short_folderName	COMUNE	Valid in the root job	

- **SET_VARIABLES**: copia in una variabile di ambiente il nome della sottocartella nella quale sono eseguite le operazioni. Nelle trasformazioni successive sarà possibile referenziarla tramite il comando ${COMUNE};



Kettle - SiiMobility Example (3/8)



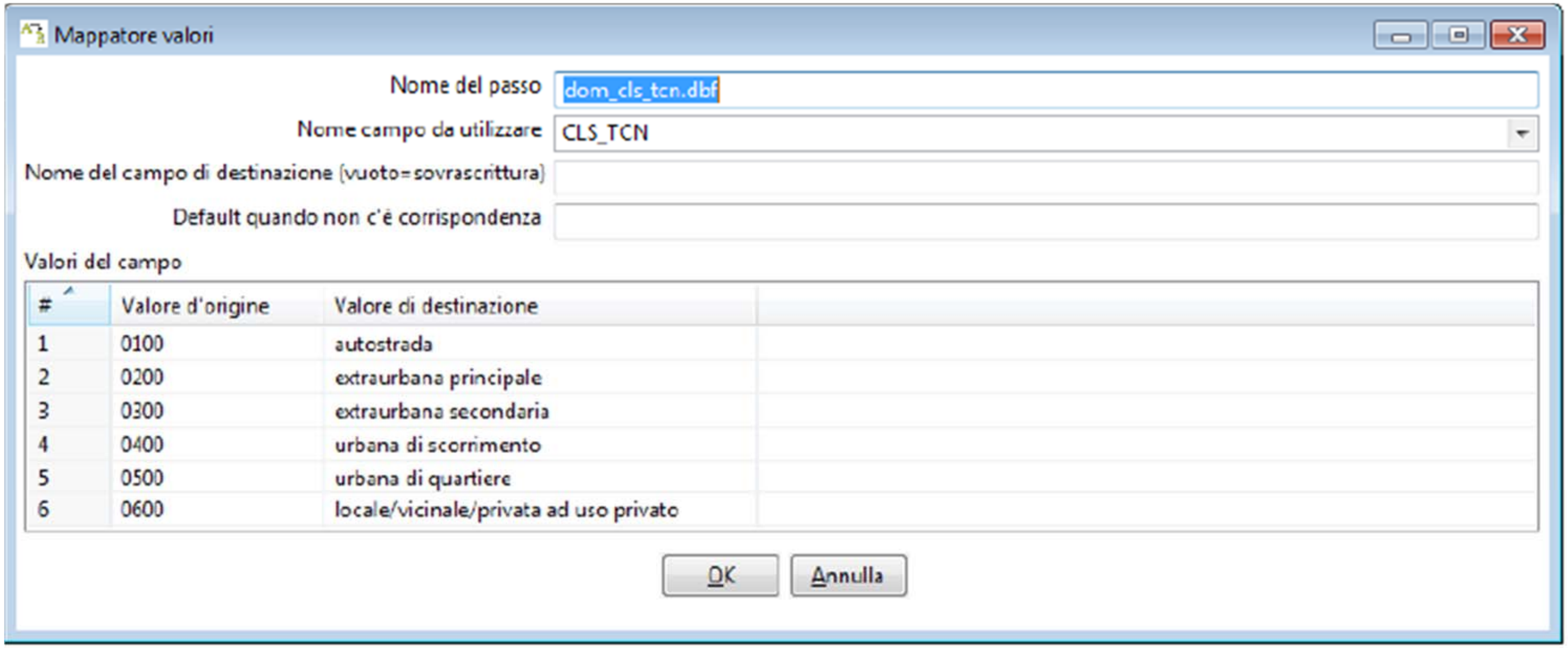
Kettle - SiiMobility Example (4/8)

- GIA_EL_STRADALE: Legge i dati da un file in formato Xbase (DBF);
- DOM_TIP_ELE: Permette di mappare i valori di una colonna attraverso una tabella;
- SELECT_VALUE: Seleziona o rimuove campi in una riga;
- GET_SYSTEM_INFO: Recupera informazioni dal sistema operativo come date, argomenti, ecc.
- ADD_CONSTANTS: Aggiunge una colonna ai dati e valorizza ogni riga con un valore costante;

Kettle - SiiMobility Example (5/8)

- REPLACE_IN_STRING: Funzione di sostituzione di sottostringhe;
- CONCAT_FIELDS: Concatena il valore di due diverse colonne e lo inserisce in una nuova colonna;
- STRING_OPERATIONS: Classiche operazioni sulle stringhe (trim, uppercase, rimozione caratteri);
- TABLE_OUTPUT: Salvataggio dei dati in una tabella di database;
- UPDATE: Aggiornamento di un record all'interno di una tabella di database.

Kettle - SiiMobility Example (6/8)



Nome del passo: dom_cls_tcn.dbf

Nome campo da utilizzare: CLS_TCN

Nome del campo di destinazione (vuoto=sovrascrittura):

Default quando non c'è corrispondenza:

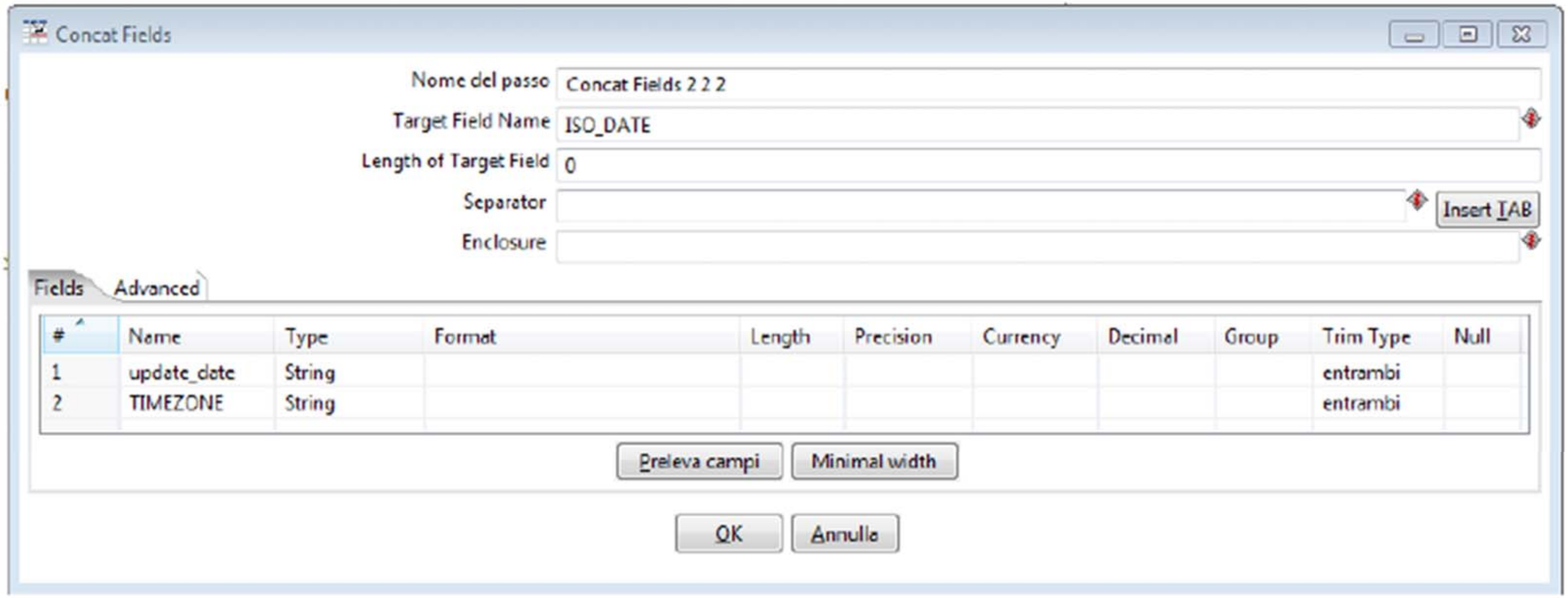
Valori del campo

#	Valore d'origine	Valore di destinazione
1	0100	autostrada
2	0200	extraurbana principale
3	0300	extraurbana secondaria
4	0400	urbana di scorrimento
5	0500	urbana di quartiere
6	0600	locale/vicinale/privata ad uso privato

Value Mapper:

- scegliere su quale campo andare ad eseguire la modifica
- definire la mappatura di tutti i possibili valori di input in altrettanti valori di output.

Kettle - SiiMobility Example (7/8)



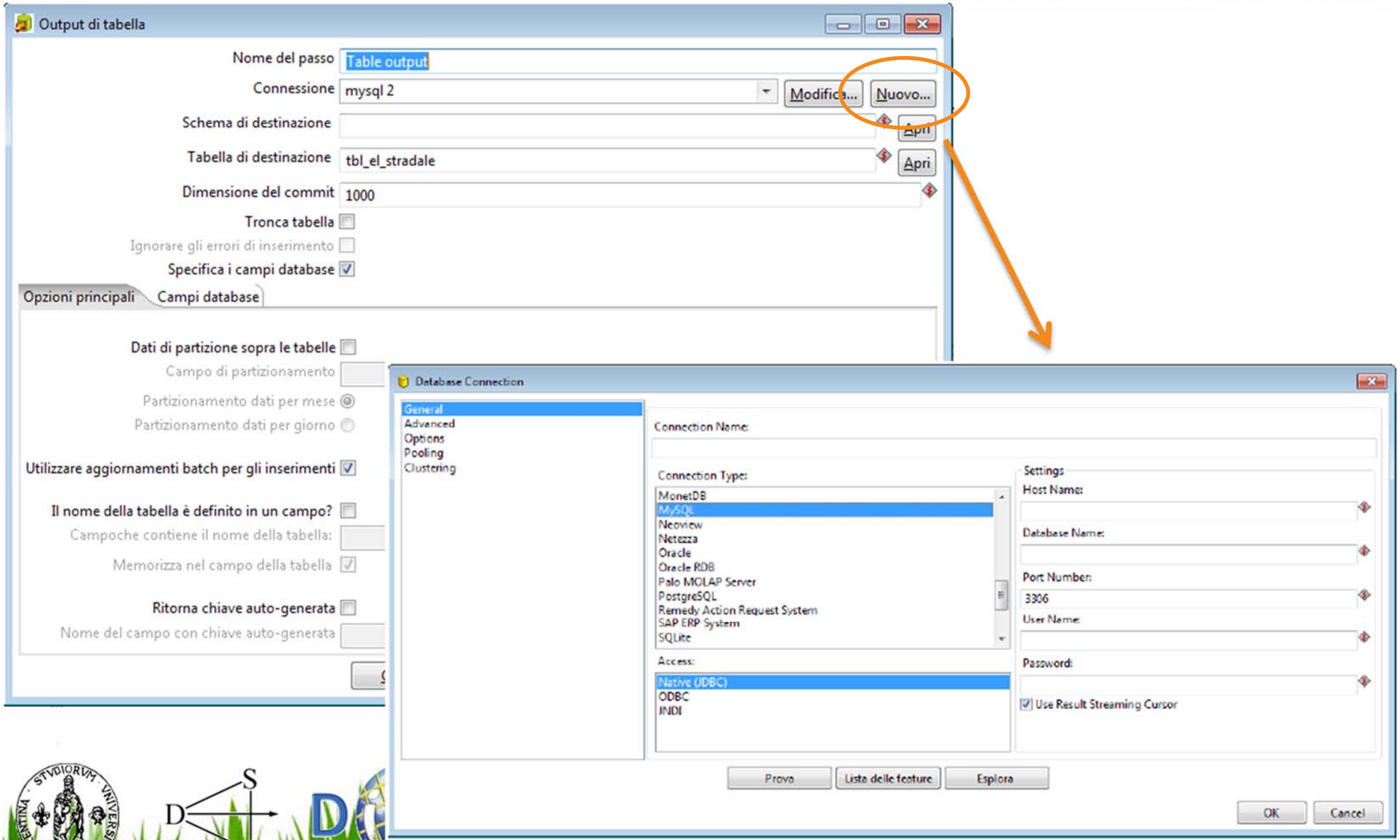
Nome del passo: Concat Fields 2 2 2
Target Field Name: ISO_DATE
Length of Target Field: 0
Separator:
Enclosure:

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim Type	Null
1	update_date	String							entrambi	
2	TIMEZONE	String							entrambi	

Concat Field:

- Concatenazione dei campi **update_date** e **TIMEZONE** per formare il nuovo campo **ISO_DATE.**

Kettle - SiiMobility Example (8/8)



Output di tabella

Nome del passo	Table output
Connessione	mysql 2
Schema di destinazione	
Tabella di destinazione	tbl_el_stradale
Dimensione del commit	1000
Tronca tabella	☐
Ignorare gli errori di inserimento	☐
Specifica i campi database	☑

Opzioni principali | Campi database

Dati di partizione sopra le tabelle	☐
Campo di partizionamento	
Partizionamento dati per mese	●
Partizionamento dati per giorno	○
Utilizzare aggiornamenti batch per gli inserimenti	☑
Il nome della tabella è definito in un campo?	☐
Campoche contiene il nome della tabella:	
Memorizza nel campo della tabella	☑
Ritorna chiave auto-generata	☐
Nome del campo con chiave auto-generata	

Database Connection

General
Advanced
Options
Pooling
Clustering

Connection Name:

Connection Type:
MonetDB
MySQL
Neoview
Netezza
Oracle
Oracle RDB
Palo MOLAP Server
PostgreSQL
Remedy Action Request System
SAP ERP System
SQLite

Access:
Native (JDBC)
ODBC
JNDI

Settings

Host Name:	
Database Name:	
Port Number:	3306
User Name:	
Password:	

☑ Use Result Streaming Cursor

KARMA (1/2)

- Tool for **mapping** structured sources **to RDF according** to an **ontology;**
- Provides a **visual interface** to display the KARMA-proposed mapping (users can adjust them)
- Users can **work with** example **data** (not only schemas or ontologies);

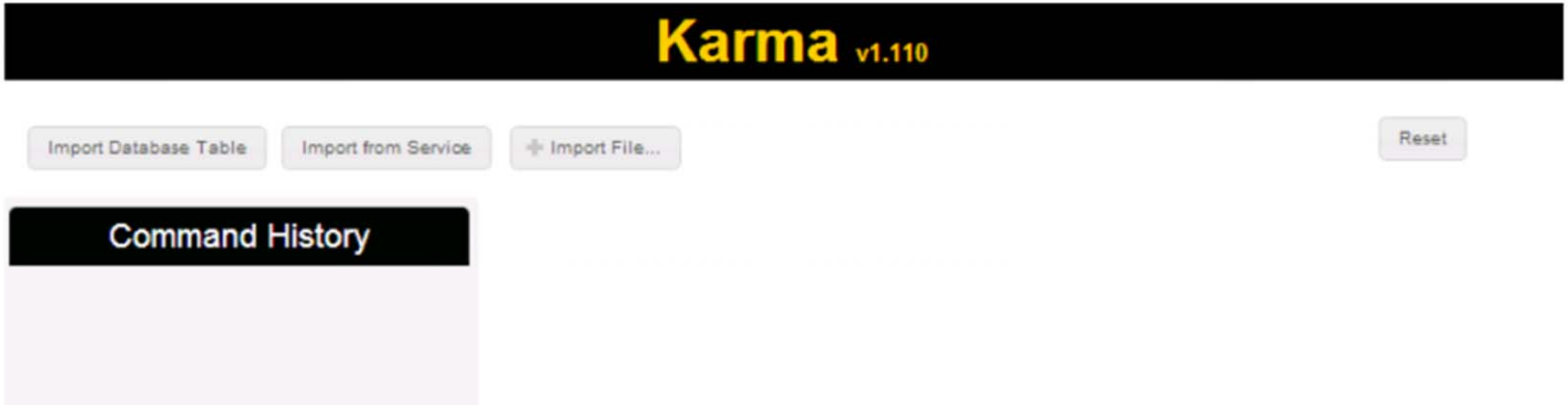
http://www.isi.edu/integration/karma

Karma v1.110

KARMA (2/2)

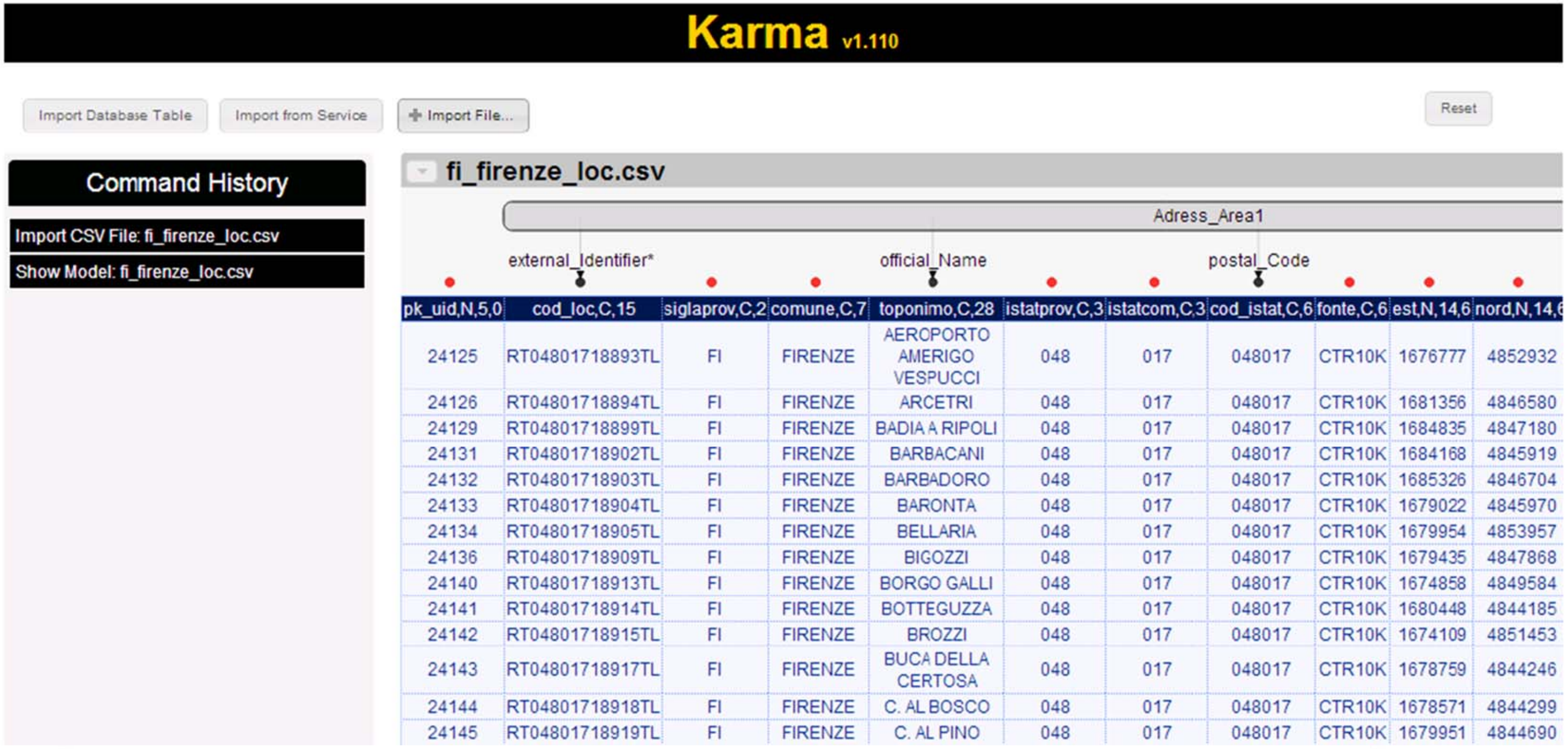
- Assigment of **semantic type** to data **columns**, specification of **relationship beetween** semantic **types** (an OWL class, a DataProperty, etc).
- Use a **Conditional Random Field** (CRF) model to **lear** the **assigment** semantic type/column;
- Thank to CRF, Karma can **suggest** a **semantic type** for unassigned data columns.
- Create a graph that defines the space of all possible mapping between the data source and the ontology. Node = class.

KARMA – SiiMobility Example (1/5)



- Loading at least one **ontology** and a **dataset**;
- After that it is possible to start **mapping**;
- **Command History** displays the sequence of the last steps performed.

KARMA – SiiMobility Example (2/5)



Command History

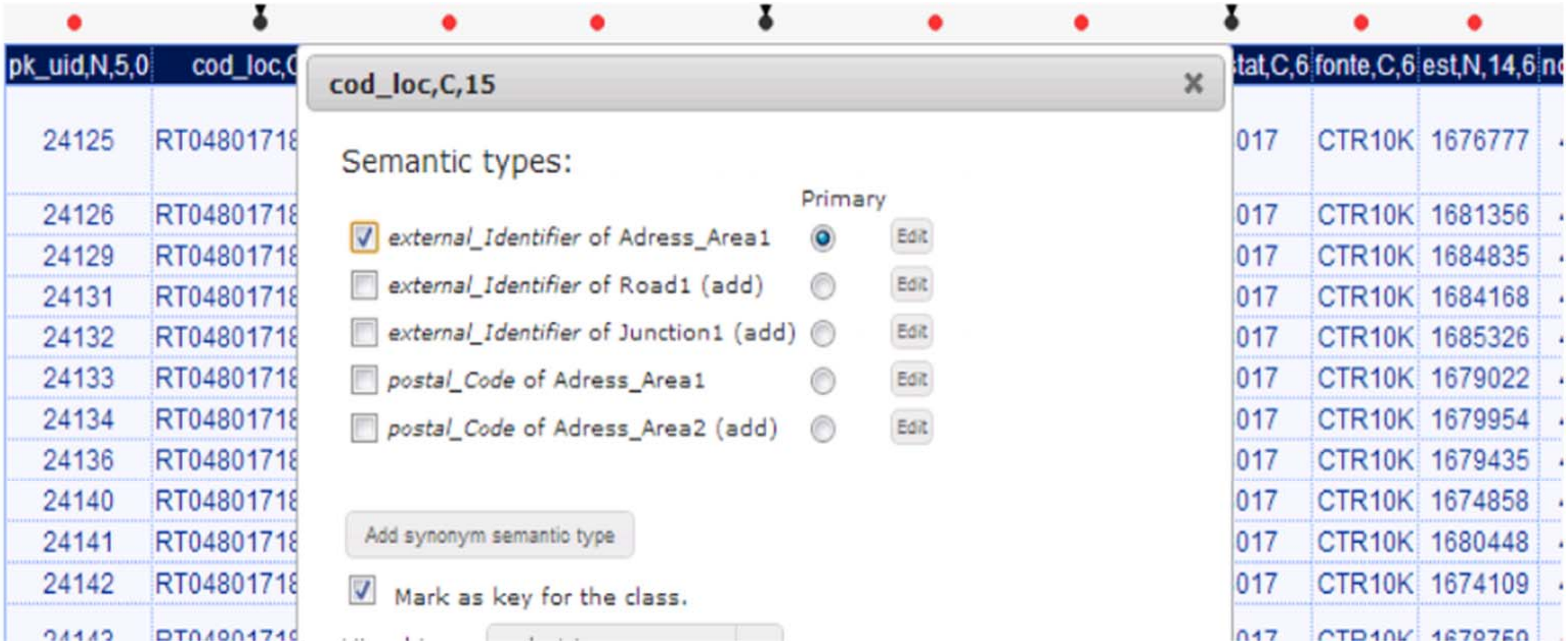
Import CSV File: fi_firenze_loc.csv

Show Model: fi_firenze_loc.csv

fi_firenze_loc.csv

pk_uid,N,5,0	cod_loc,C,15	siglaprov,C,2	comune,C,7	toponimo,C,28	istatprov,C,3	istatcom,C,3	cod_istat,C,6	fonte,C,6	est,N,14,6	nord,N,14,
24125	RT04801718893TL	FI	FIRENZE	AEROPORTO AMERIGO VESPUCCI	048	017	048017	CTR10K	1676777	4852932
24126	RT04801718894TL	FI	FIRENZE	ARCETRI	048	017	048017	CTR10K	1681356	4846580
24129	RT04801718899TL	FI	FIRENZE	BADIA A RIPOLI	048	017	048017	CTR10K	1684835	4847180
24131	RT04801718902TL	FI	FIRENZE	BARBACANI	048	017	048017	CTR10K	1684168	4845919
24132	RT04801718903TL	FI	FIRENZE	BARBADORO	048	017	048017	CTR10K	1685326	4846704
24133	RT04801718904TL	FI	FIRENZE	BARONTA	048	017	048017	CTR10K	1679022	4845970
24134	RT04801718905TL	FI	FIRENZE	BELLARIA	048	017	048017	CTR10K	1679954	4853957
24136	RT04801718909TL	FI	FIRENZE	BIGOZZI	048	017	048017	CTR10K	1679435	4847868
24140	RT04801718913TL	FI	FIRENZE	BORGO GALLI	048	017	048017	CTR10K	1674858	4849584
24141	RT04801718914TL	FI	FIRENZE	BOTTEGUZZA	048	017	048017	CTR10K	1680448	4844185
24142	RT04801718915TL	FI	FIRENZE	BROZZI	048	017	048017	CTR10K	1674109	4851453
24143	RT04801718917TL	FI	FIRENZE	BUCA DELLA CERTOSA	048	017	048017	CTR10K	1678759	4844246
24144	RT04801718918TL	FI	FIRENZE	C. AL BOSCO	048	017	048017	CTR10K	1678571	4844299
24145	RT04801718919TL	FI	FIRENZE	C. AL PINO	048	017	048017	CTR10K	1679951	4844690

KARMA – SiiMobility Example (3/5)



pk_uid,N,5,0	cod_loc,C	tat,C,6	fonte,C,6	est,N,14,6
24125	RT04801718	017	CTR10K	1676777
24126	RT04801718	017	CTR10K	1681356
24129	RT04801718	017	CTR10K	1684835
24131	RT04801718	017	CTR10K	1684168
24132	RT04801718	017	CTR10K	1685326
24133	RT04801718	017	CTR10K	1679022
24134	RT04801718	017	CTR10K	1679954
24136	RT04801718	017	CTR10K	1679435
24140	RT04801718	017	CTR10K	1674858
24141	RT04801718	017	CTR10K	1680448
24142	RT04801718	017	CTR10K	1674109

cod_loc,C,15 ×

Semantic types:

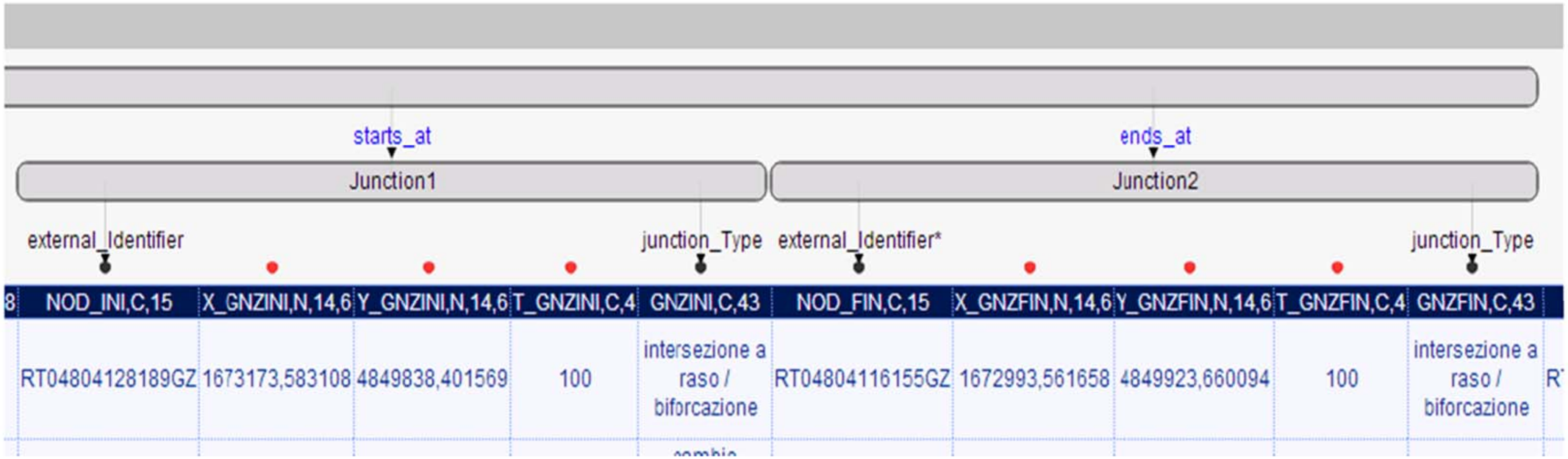
		Primary	
✓	*external_Identifier* of Adress_Area1	●	Edit
☐	*external_Identifier* of Road1 (add)	○	Edit
☐	*external_Identifier* of Junction1 (add)	○	Edit
☐	*postal_Code* of Adress_Area1	○	Edit
☐	*postal_Code* of Adress_Area2 (add)	○	Edit

Add synonym semantic type

✓ Mark as key for the class.

First step: to establish the **relationship** between the **columns** of the dataset and the **classes/properties** of the ontology

KARMA – SiiMobility Example (4/5)



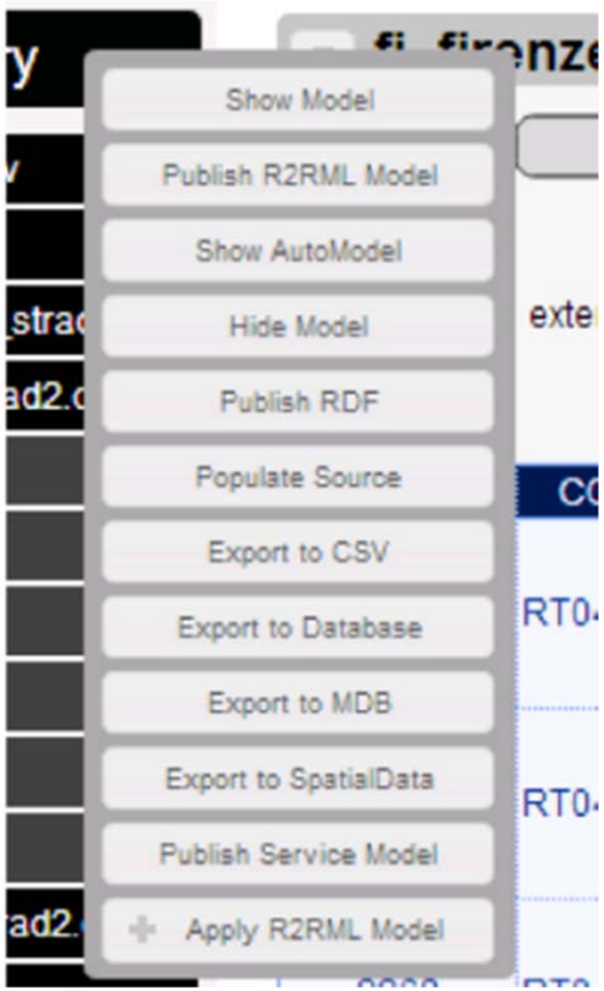
8	NOD_INI,C,15	X_GNZINI,N,14,6	Y_GNZINI,N,14,6	T_GNZINI,C,4	GNZINI,C,43	NOD_FIN,C,15	X_GNZFIN,N,14,6	Y_GNZFIN,N,14,6	T_GNZFIN,C,4	GNZFIN,C,43	
	RT04804128189GZ	1673173,583108	4849838,401569	100	intersezione a raso / biforcazione	RT04804116155GZ	1672993,561658	4849923,660094	100	intersezione a raso / biforcazione	R

- Is possible to define multiple instances of the same semantic class and bind each column to the correct instance;
- Through the check-box 'Mark as key for the class', specific URIs can be defined.

KARMA – SiiMobility Example (5/5)

After the data mapping, the resulting model can be exported in various formats:

- RDF Turtle
- R2RML model (usable to automatically map a relational data source in RDF)
- Notation 3 (N3) model
- CSV, Access, KML



Second step useful tools (Link to external source)

- **Dbpedia** uses the URI **http://dbpedia.org/resource/Berlin** to identify Berlin;
- **Geonames** uses the URI **http://sws.geonames.org/2950159** to identify Berlin;
- **URI aliases**: both URIs refer to the same non-information resource (common on the Web of Data);
- **Social function**: URI aliases allow different views and opinions to be expressed.
- **owl:sameAs** to link to URI aliases.

How to discovered URI Aliases

- **Manually**: **identify** particular **datasets** as suitable linking targets, manually **search** in these for the **URI** references you want to link to;
- **Automatically** : use some tools (**Silk Link Discovery Framework**, Mannheim University) for discovering **relationships** between data items within **different Linked Data** sources.
- Finally, set the built-in OWL property **owl:sameAs** for pairs of URIs identified.
- **PROV** Is a Suitable Technology for Curating the Links. In addition to supporting the user interface for human verfication of links.

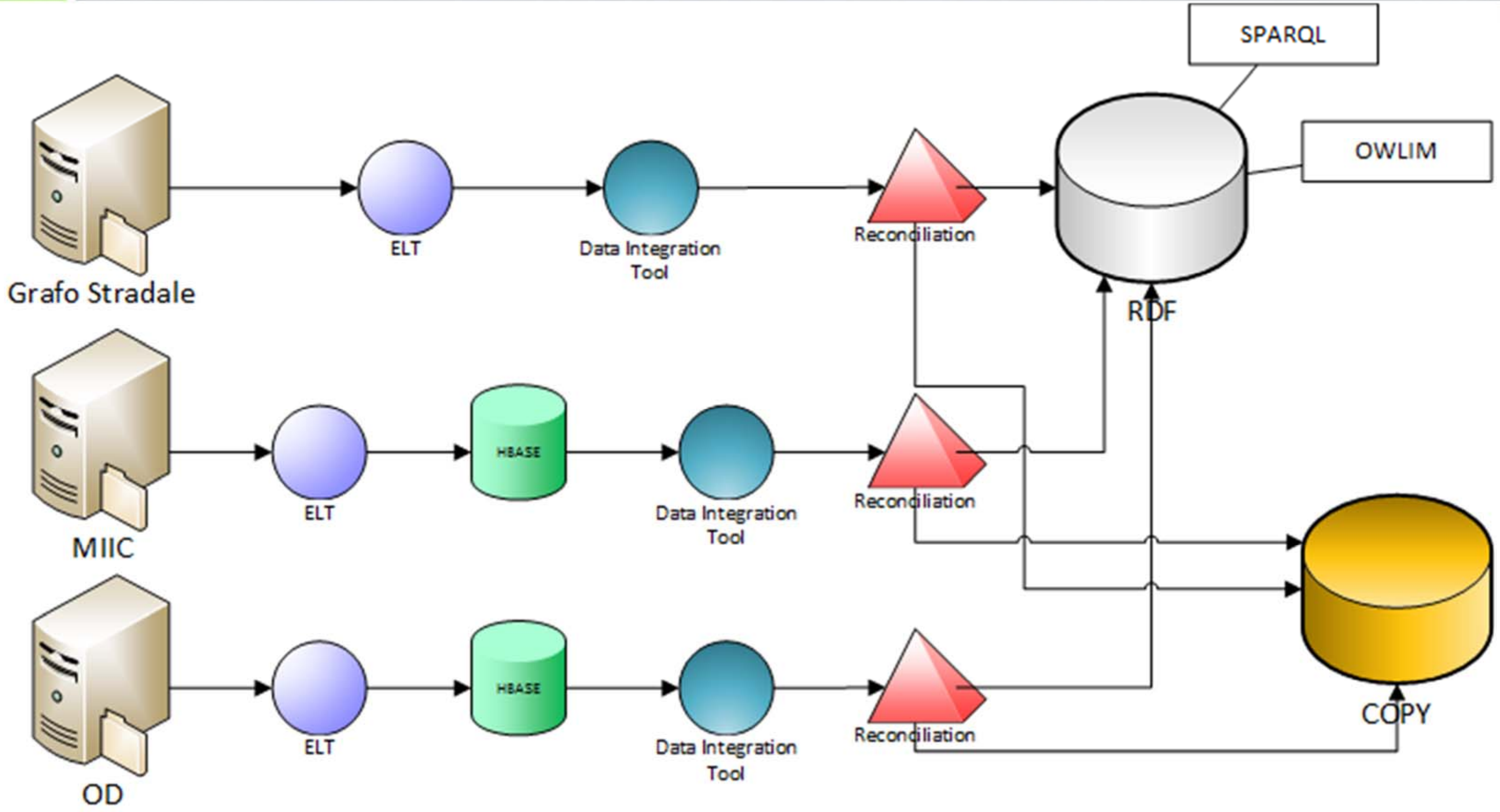
Third step useful tools (Curate the Linked Data)

- Linked Data will remain usable twenty years from now **only if** URIs persist and remain resolvable to documentation of their meaning;
- **Changes** or additions in **interlinked** datasets can **invalidate** existing **links** or imply the need to generate new ones;

Problems:

- Most vocabularies reside on a **single Web server**, representing a **single** point of **failure**.
- Vocabularies used today are developed and **curated** by **private maintainers** (individuals or institutions).

SiiMobility Project



Possibili elaborati/tesi

Progetto **SiiMobility:**

- **FLUCTUS:** IntelligentSensor System per la creazione e la gestione di Wireless Sensor Networks (**www.e-fluctus.com**);
- **UDOO:** mixa i migliori elementi di Raspberry Pi e Arduino in un unico mini PC compatibile anche con Android (**www.udoo.org**).

- Big data examples : http://blogs.wsj.com/drivers-seat/2012/01/18/forget-fingerprints-car-seat-ids-drivers-rear-end/
- http://www.focus.it/tecnologia/motori/il-sedile-antifurto-riconosce-il-guidatore_C12.aspx