Big Data Integration

1. Data integration
2. Data lake
3. Big data integration in a DW
4. Big data integration approaches
5. Operational data lake
6. Trends

Objectives

- Integration of (structured) data from IS
  - Customer information, transactions, etc.
- With big data
  - Sensor data
    - Call recordings, weblogs, GPS positions, trading data, intelligent sensors
  - Social data
    - Customer feedback, microblogs (Twitter), social networks (Facebook)
- For fine-grain, precise data analysis
  - Real-time information and high-value knowledge

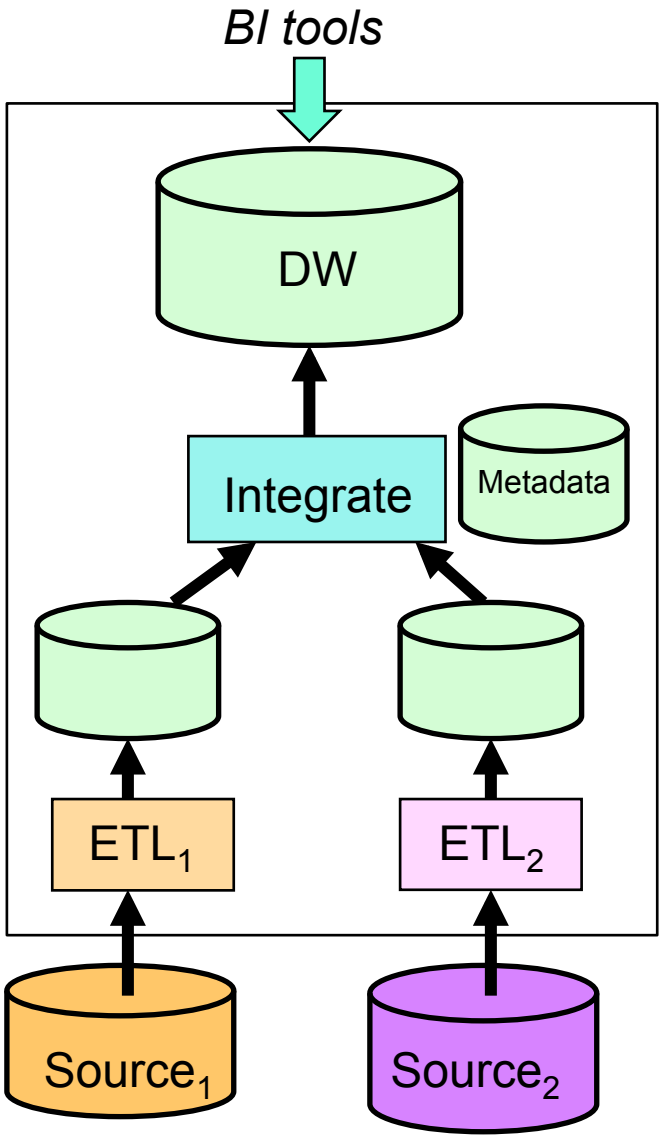
1. Data Integration

1. Data warehouse (DW)
   - Real integration
     - Data storage from various data in a database
     - BI queries on database
2. Data federator
   1. Virtual integration
      - No storage
      - Queries to data sources

- In both approaches, need of *metadata* to represent integrated data

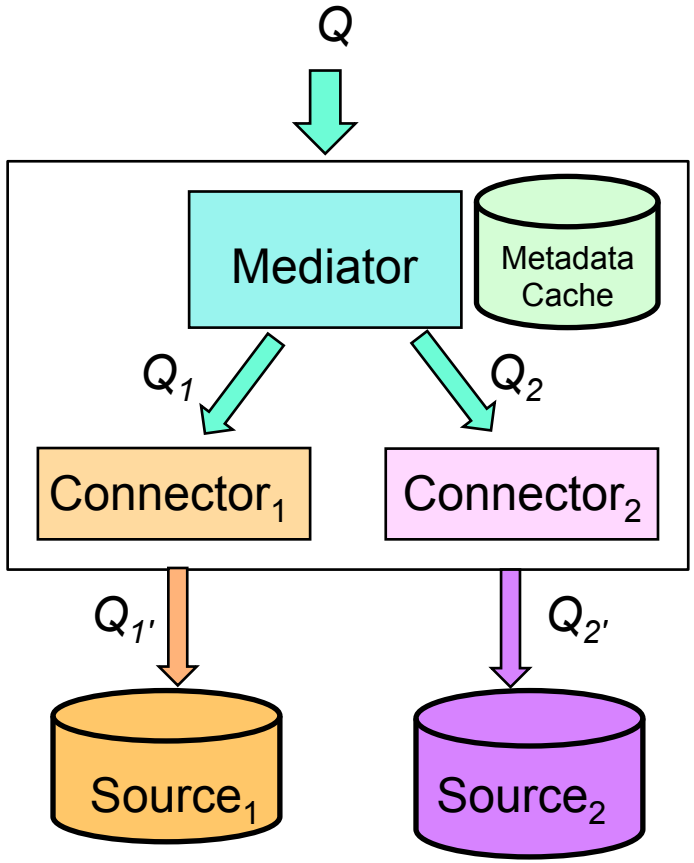
DW Architecture

- DW
  - *Enterprise DW* to integrate all data
  - *Data marts* to adapt the data to BI
- ETL (Extract-Transform-Load)
  - Extraction from source
  - Transformation in a common model and cleaning
  - Loading in database



Data Federator Architecture

- Mediator
  - Transforms queries into sub-queries for connectors
  - Integrates (computes) the results of sub-queries
- Connector
  - Transforms sub-queries into sources' languages
  - Transforms the results in the mediator format

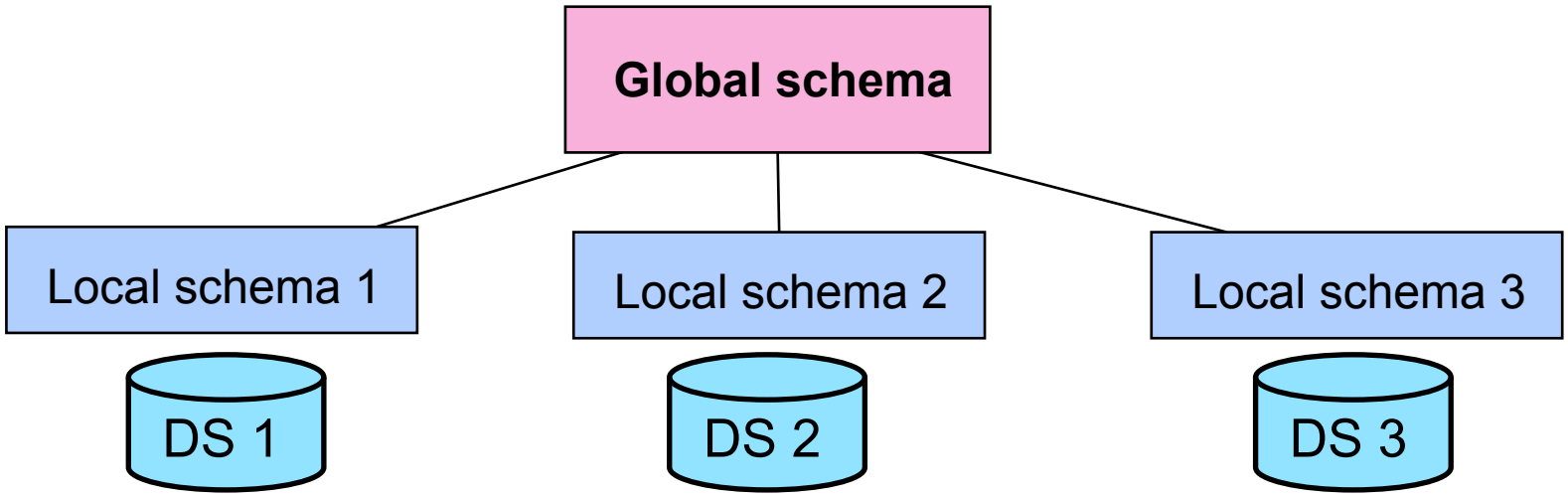


DW versus Federator

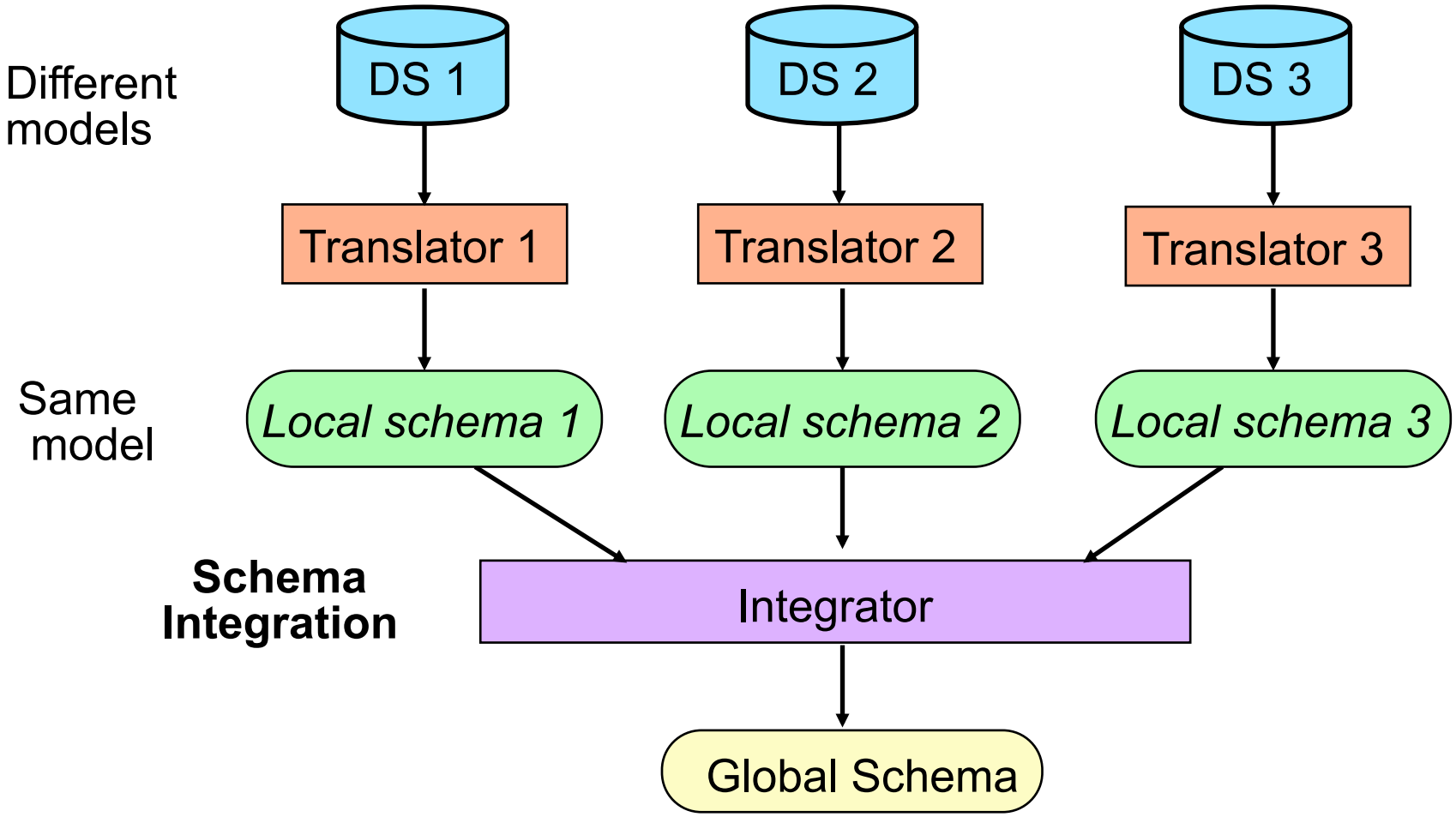
- DW
  - + Rapid and complex analysis (e.g. OLAP cubes)
  - + Strong integration (cleaning, quality)
  - – No real-time access to data sources
  - – Refreshment time
  - – Complex and costly architecture
- Federator
  - + Real-time access to data sources
  - + Access to very many sources
  - + No data storage
  - – High, impredictable response time
  - – Difficult to deal with data quality in real-time

Schema Management

- Local schema
  - Describes the data source in a common model (e. g. relational)
- Global schema
  - Gives a global and unified description of all data sources (e. g. relational views)
  - Mapping rules with local schemas



Schema Integration



Schema Integration: steps

1. Pre-integration
   - *Identification* of related elements and establishment of conversion rules
     - E.g. 1 inch = 2,54 cm, 1€=x$
2. Comparison
   - *Identification* of name conflicts (e. g. synonyms, homonyms) and structural conflicts (e. g. types, keys)
3. Conformance
   - *Resolution* of conflicts (type conversions, key modifications, etc.)
4. Creation of global schema
   - Definition of integrated data
   - Conversion rules from local schemas

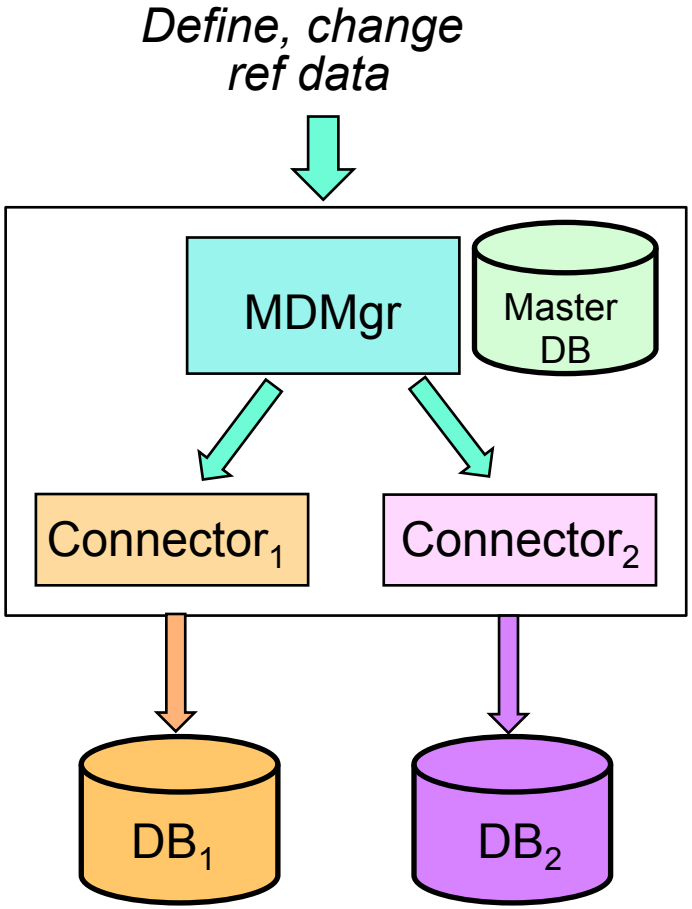
- Tools for integration
  - Dictionaries, thesaurus, ontologies
  - Mapping tools, ex. MapForce d'Altova

Data Quality and MDM

- Referential data
  - Data shared by all processes that support the company's business
  - E.g. customer, product, supplier,...
- Problem
  - This data is stored redundantly in operational or OLAP databases, which generates inconsistencies following the changes
  - E.g. customer address different from one database to another
- Solution: master data management (MDM)
  - Objective: definition, access and publication of a unified vision of reference data within the IS, independently of their storage in the various databases

MDM: main functions

- Administration
  - Rights of the different actors (owner, user,...)
- Management of the repository data life cycle
- Metadata management
- Storage in a database
- Upload and publication of data
- Guiding rules
  - E.g. an owner for a referential data



Products & Services

- DW, MDM & federator
  - All major software vendors: Oracle, IBM, Microsoft, SAP, …
- DW & MDM specialized vendors
  - Teradata, Dell, HP, EMC Greenplum, ParAccel, Talend, Tibco Orchestra, …

2. Data Lake

- Term coined by James Dixon, CTO Pentaho
- Data lake: collection of raw data in native format
  - Each element has a unique identifier and metadata
  - For each business question, you can find the relevant data set to analyze it
- Originally based on Hadoop
  - Enterprise Hadoop

Problems with DW

- Very long development process (2-3 years)
  - Precise definition of data needs and organization
- Schema on write
  - Write the data according to schema, then read them
  - Difficult and costly evolution to adapt to changes in the business environment
- Expensive development with ETL
  - The global schema involves data cleaning, complex ETL programs and tracking data sources to manage refreshment

Schema on Write versus Schema on Read

- Schema on write (RDBMS)
- Prescriptive modeling
  - Create schema S for DB
  - Write data according to S
  - Query data in S format
- New columns must be defined before new data can be added
- Efficient querying but difficult evolution

- Schema on read (Hadoop)
- Descriptive modeling
  - Write data in native format
  - Define schema S when needed
  - Query data in native format according to S (on the fly ETL)
- New data can be added at any time
- Agility and flexibility

Advantages of a Data Lake

- Schema on read
  - Write the data as they are, read them according to a diagram (e.g. code of the Map function)
  - More flexibility, multiple views of the same data
- Multi-workload data processing
  - Different types of processing on the same data
  - Interactive, batch, real time
- Cost-effective data architecture
  - Excellent cost/performance and ROI ratio with SN cluster and open source technologies

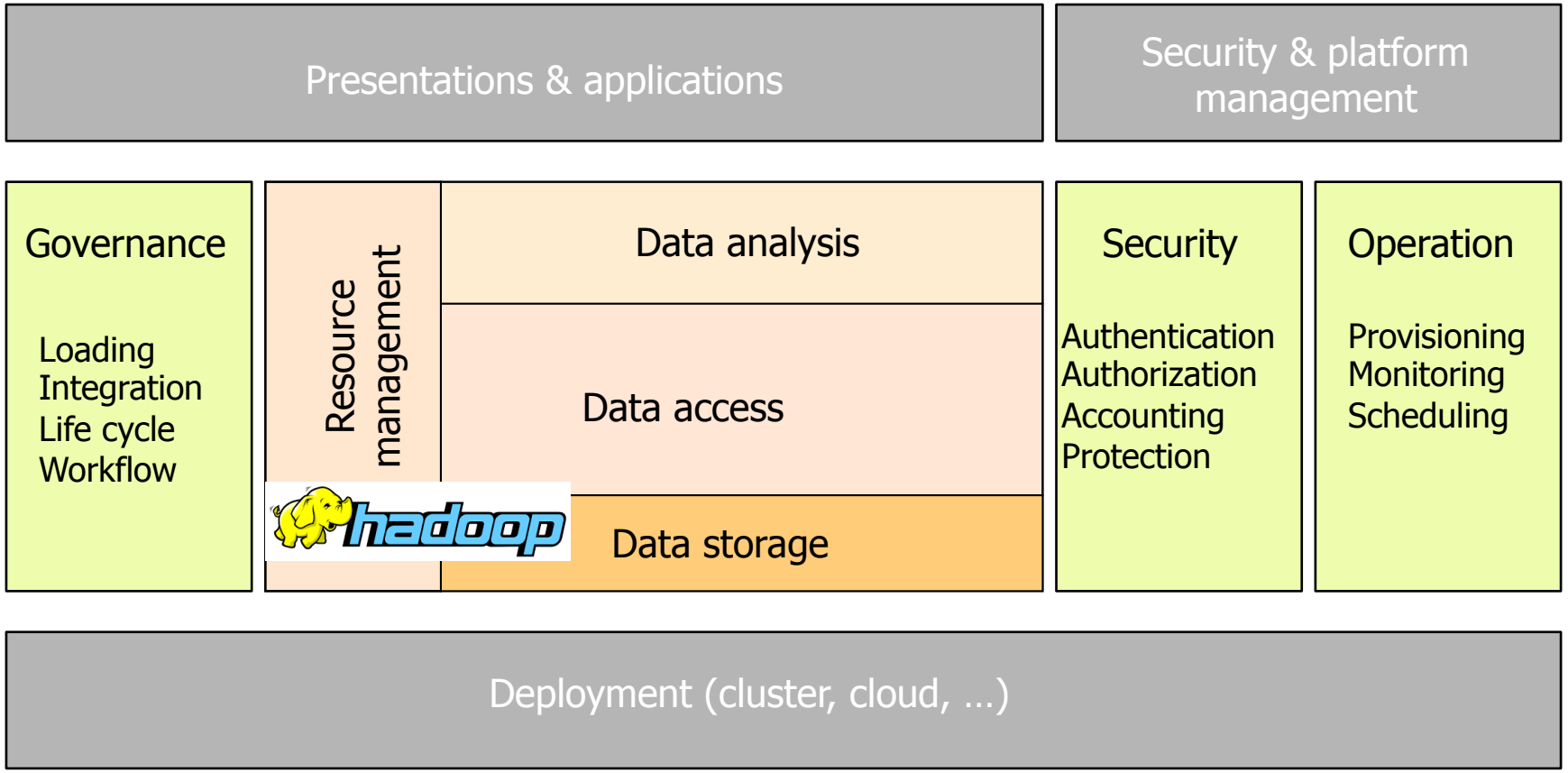
Principles of Data Lake

- Collect all useful data
  - Raw data, transformed data
- Dive from anywhere
  - Users from different business units can explore and enrich the data
- Flexible access
  - Different access paths to shared infrastructure
    - Batch, interactive (OLAP and BI), real-time, search,.....

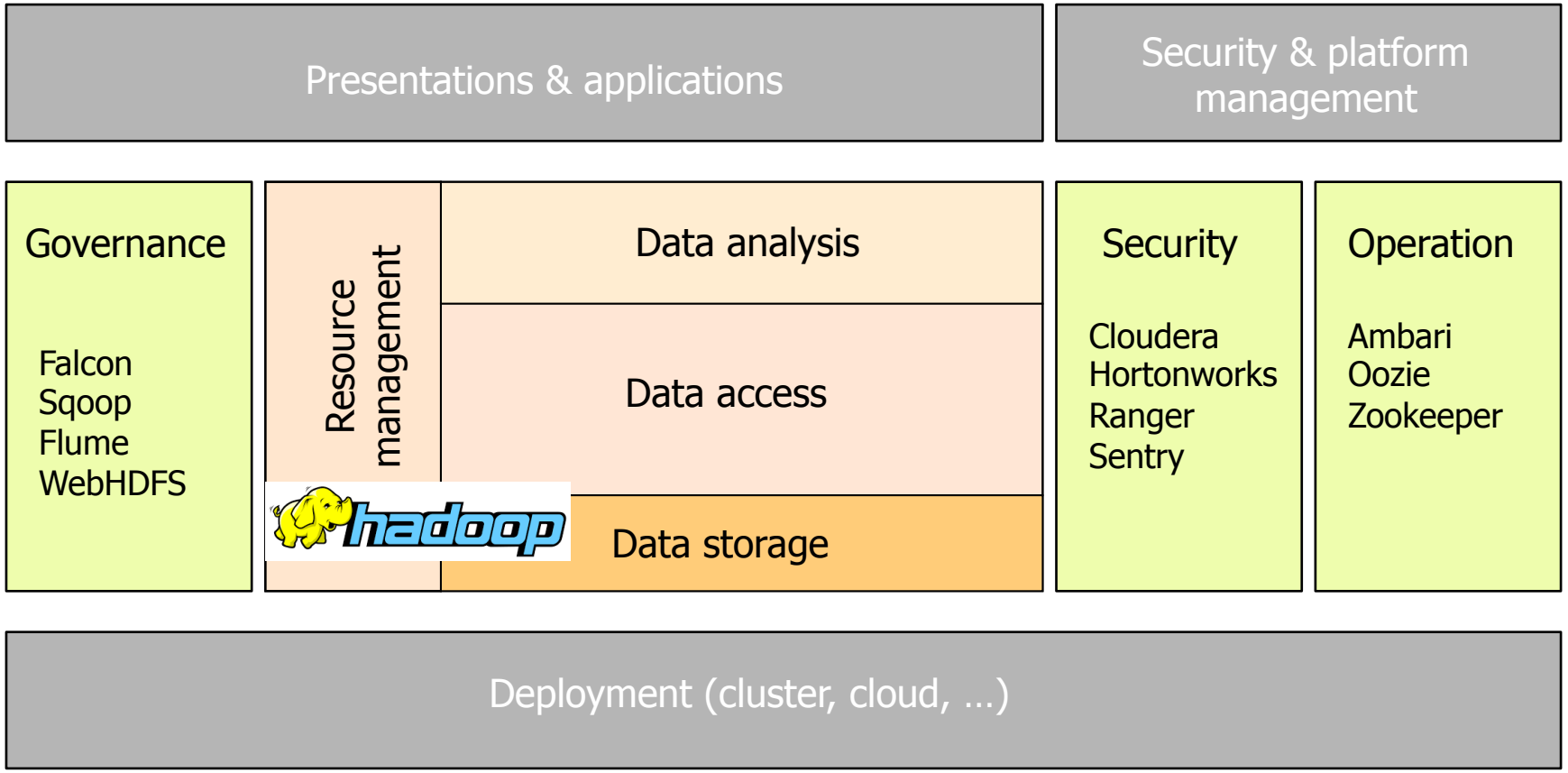
Platform for Data Lake

- Main functions
  - Data management, to store and process large amounts of data
  - Data access: interactive, batch, real time, streaming
  - Governance: load data easily, and manage it according to a policy implemented by the *data steward*
  - Security: authentication, access control, data protection
  - Platform management: provision, monitoring and scheduling of tasks (in a cluster)

Enterprise Hadoop: functions



Enterprise Hadoop: components

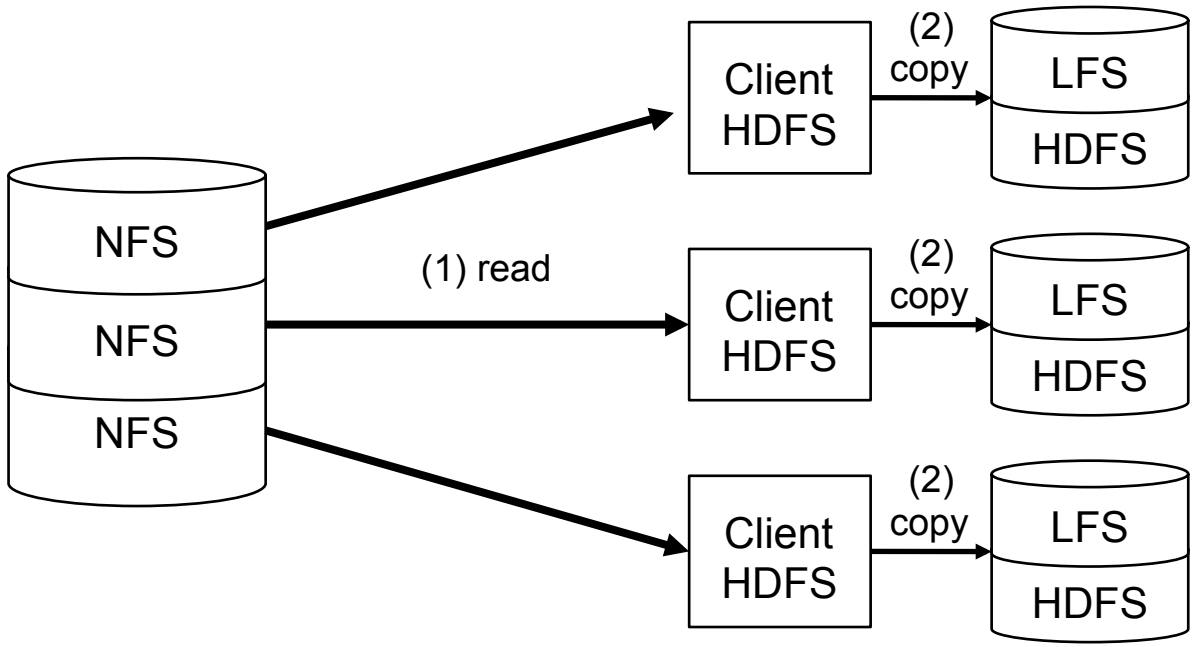


Loading in a Data Lake

- Loading very large volumes of data into a data lake can take a long time with a simple solution
  - E.g. Only one HDFS client that reads the data, e. g. from NFS, to write it to HDFS
- Solution: parallel loading
  - Similar to DW tools
    - E.g. Oracle SQL*Loader, Teradata Parallel Transporter

HDFS Parallel Loading

- One HDFS client on each node of the HDFS cluster
  1. Reading of the source data in the local file system (LFS)
  2. Parallel copy in HDFS
     - fs put oR copyFromLocalFile commands



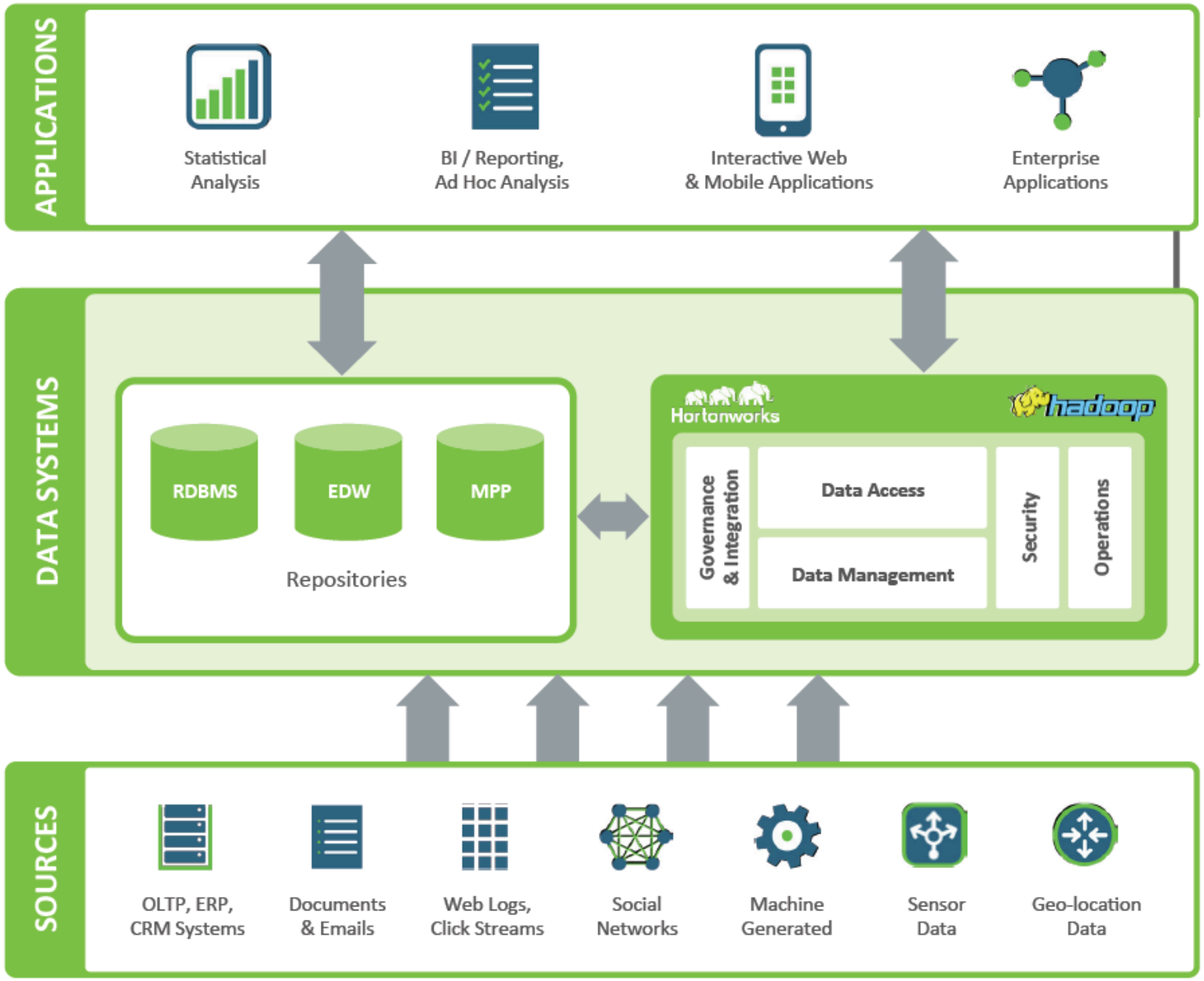
Hadoop Loading Tools

- Sqoop
  - Data import/export tool between reltional databases and Hadoop
  - Parallel import into HDFS
- Flume
  - Tool for collecting, aggregating and transferring of streams for Hadoop
  - Parallelism with multiple agents
- Flafka = Flume + Kafka
  - Tool for importing Kafka streams for Hadoop
  - Parallelism with multiple Flume agents

BI Tools for Hadoop

- New tools or extension of traditional BI tools
- Two approaches (can be combined)
  1. Hadoop SQL driver: HiveQL, SparkSQL, Impala, ...
     - Examples: Tableau, Platfora, Pentaho, Power BI, DB2 BigSQL, ...
  2. HDFS access via functions library
     - Examples: Datameer, Power BI, DB2 BigSQL, …

Orthonworks data lake



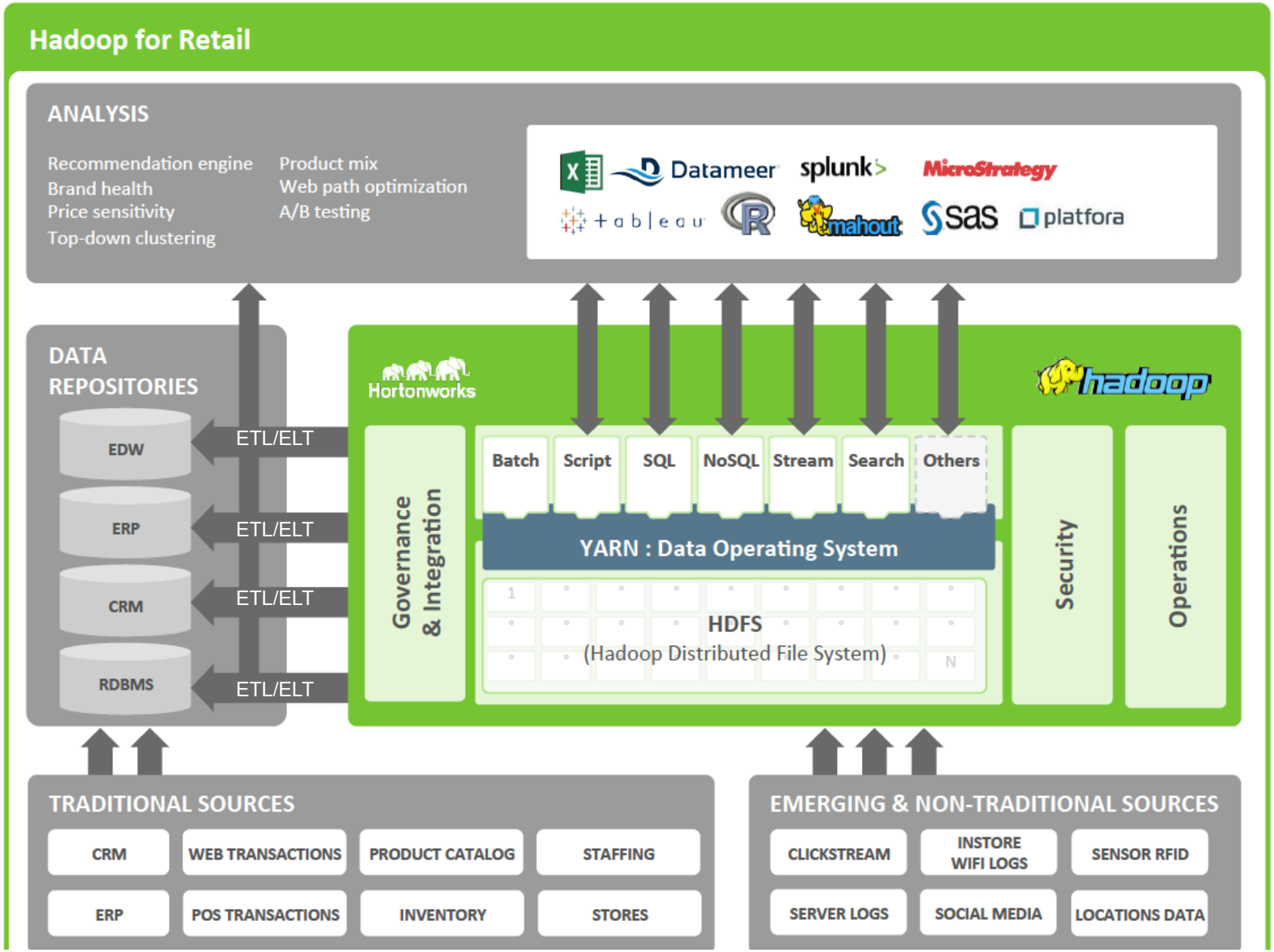
Best Practices for Data Lake

- List of business priorities and added value
  - In comparison with the company's DW
- Global vision of architecture
  - Use a POC to experiment with the first elements
- Include data governance and metadata
  - Security and confidentiality strategy
  - Important if sharing between business lines
- Calculation and storage model
  - Study of the scalability and scalability aspects to make technical choices
- Operational plan
  - Definition of SLAs in terms of uptime, volumes, variety and speed of data, etc.
- Long-term vision
  - To replace existing DWs

Case Study: retail

- Very large retail company in the USA
  - Per year: $1 billion in marketing, 100 million customer interactions, generating $74 billion in purchases
  - Objective: to improve marketing performance
- Need: 360° view of customer purchasing behaviour
  - Correlate transaction data (stored in isolated silos) with customers' marketing campaigns and online browsing experience
- Solution
  - Data lake with Hortonworks Data Platform
- Results
  - Better targeted campaigns, with personalized coupons, promotions and emails

Architecture



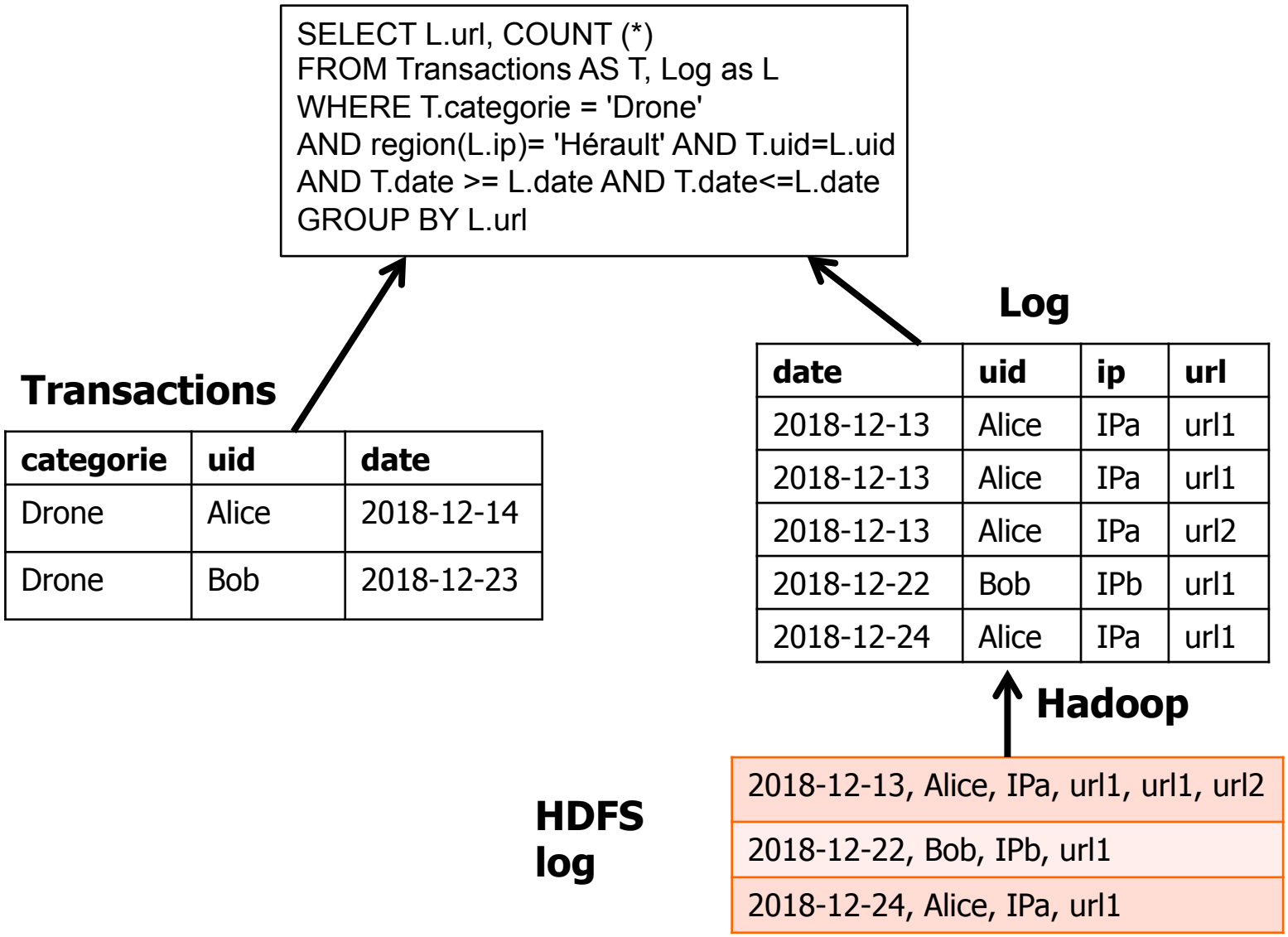
3. Big Data Integration in DW

- Objectives
  - Integrate data from big data platforms into the DW
  - Virtual integration for exploration
  - Real integration for analysis
- Enable analysis combining company data and big data

Application Example

- Consumer electronics store (online and in store)
  - All transactions are recorded in a relational database
  - Online navigation of customers registered in an HDFS log
- Objective: analyze the correlation between online behavior and sales
  - Requires transactions to be joined with the corresponding log data
- Example of a request
  - Number of views of URLs visited by customers with an IP address in Hérault who purchased a drone the next day of their visit

Example Integration Query



New Requirements for DW

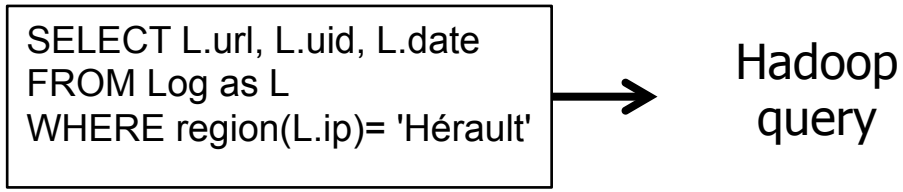
1. Acquisition of big data
   - Volume, Velocity, Variety
2. Data organization
   - Integration of big data and structured data
3. Data analysis
   - Interactive analysis, combining enterprise data and big data

Big Data Solutions for DW

1. Acquisition
   - Connectors: HDFS, NoSQL, CEP, …
   - Metadata extraction
   - Initial processing close to origin site
     - To avoid heavy data transfer
2. Organization
   - Integration of HDFS, NoSQL and SQL data with external tables
   - Integration of big data frameworks
   - Replication and synchronization between different databases
3. Analysis
   - Interactive analysis, by pushing queries directly on big data

SQL External Tables

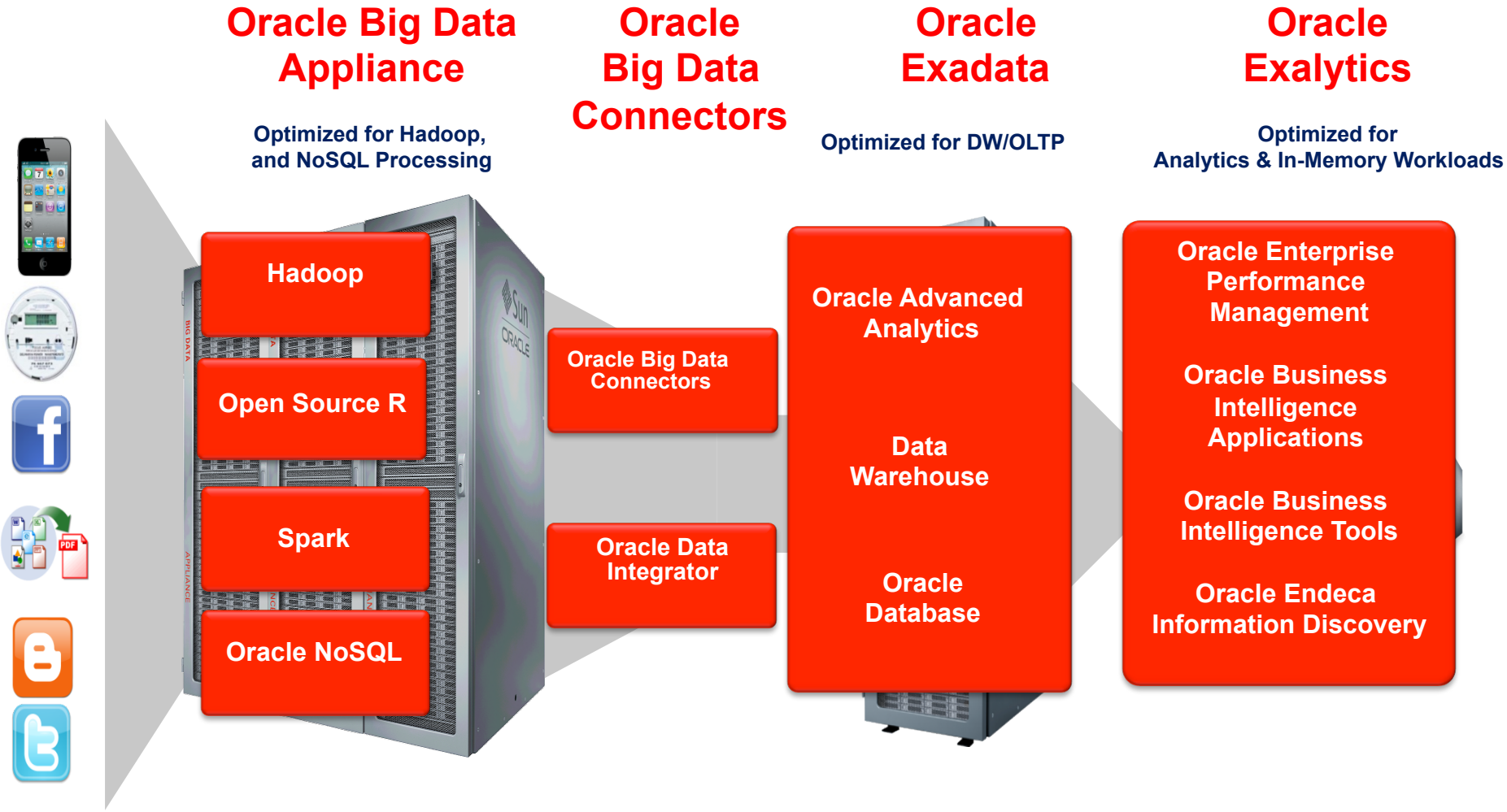
- External table
  - Definition of a table stored outside the database
    - HDFS, NoSQL,…
    - Location information, format (e.g. CSV, …), etc.
  - Use by the SQL compiler to generate the Hadoop code (e.g. MapReduce) to execute the query
    - Possibility to push predicates in Hadoop to select the useful data
    - E.g. Let L be an external table, the compiler produces:



Big Data Platforms for DW

- IBM Analytics Engine
  - Hadoop Hortonworks, Spark, outils
  - IBM Cloud Storage for data storage in a cluster
- Microsoft HDInsight (Azure Cloud)
  - Hadoop Hortonworks, Spark, Kafka, NoSQL, outils
  - Polybase: external HDFS tables in TransacSQL accessed from PDW
- Oracle Big Data Appliance

Oracle Big Data Platform



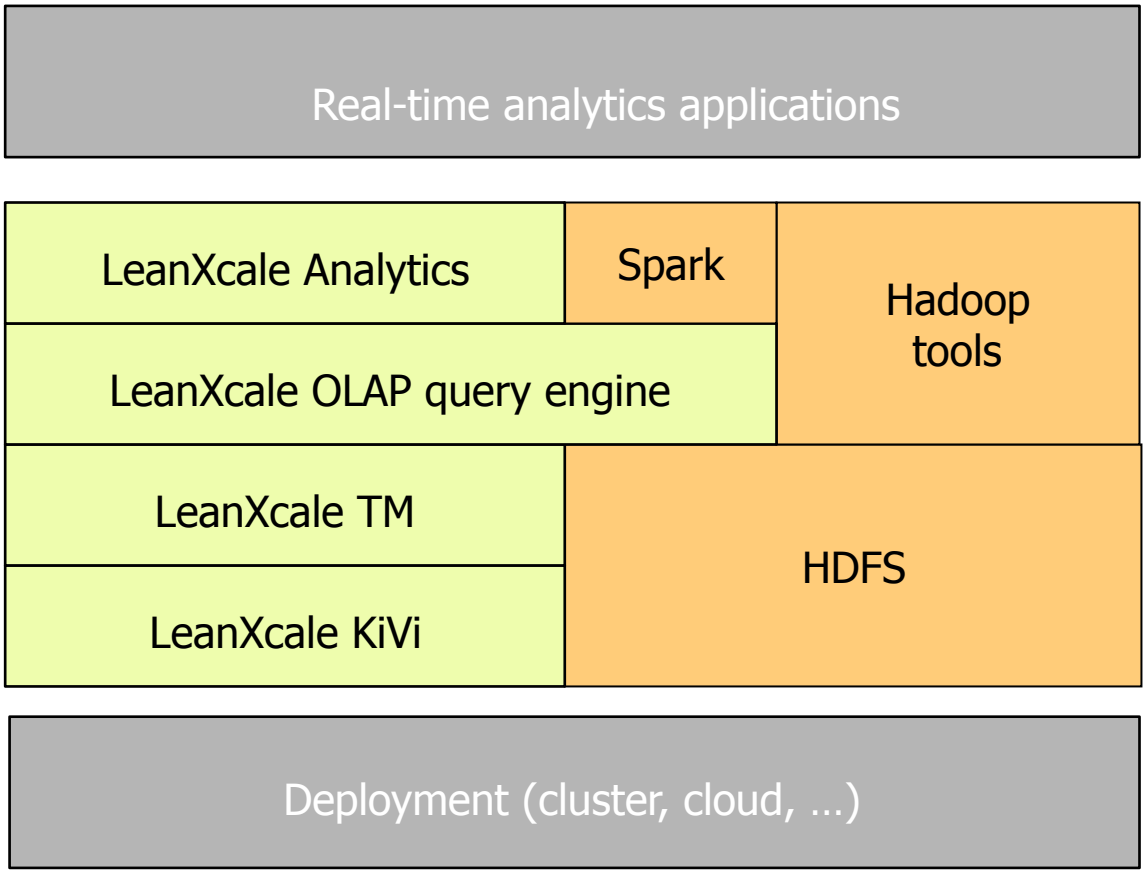
Big Data Integration Platforms

Vendor	Product	Comment
Amazon	Amazon Web Services	Hadoop, DynamoDB, Redshift, …
Cloudera	Impala	Enterprise Hadoop
Cisco	Data virtualization	Enterprise Hadoop
Google	Cloud platform	Big query, dataflow, Bigtable, F1, etc.
Horthonworks	Data platform	Enterprise Hadoop
IBM	Analytics Engine	DW - Hadoop
Microsoft	HDBigInsights	DW - Hadoop
Oracle	Big data	DW - Hadoop
Pentaho	Big data integration	Graphical ETL , visualization tools
SAP	HANA	DW – Hadoop, Smart Data Access
Talend	Data integration	Enterprise Hadoop, open source
Teradata	Integrated big data	DW - Hadoop, Aster (big data analysis)

5. Operational Data Lake

- A contradiction in terms?
  - Recall: the data lake allows operational data disseminated in the company's silos to be loaded and analyzed
- Objective: take advantage of the scale-out infrastructure to support transactional applications as well
  - Advantage: real-time analysis
- How: with HTAP
  - In a SN cluster

Operational Data Lake with LeanXcale



Case Study: Banking

- Banks use data lakes to store their historic data
  - Data coming from mobile devices and from the web coming with very high peaks
  - Use of ML to build predictive models over the historic data
  - Data copied from data lake into GPU-based clusters for doing ML
- Problem
  - During data loading, ML processes over that data must be paused since otherwise they would observe inconsistent data and would hamper the model they are building
  - Real-time analytics (e.g. real-time marketing) not possible

Objectives

- Business
  - Store data in real-time and deliver real-time analytics over operational data without creating data silos
- Technical
  - Ingest new operational data at high rates efficiently and cost-effectively (in terms of hardware and software)
  - Provide transactional guarantees over the data
  - Fast analytics over real-time and historic data
  - Ability to query in real-time other operational data sources such as NoSQL
- Operational
  - Scalable, yet simple infrastructure
  - Avoid a combination of open source systems
    - Hard to maintain and subject to arbitrary changes that can make them incompatible or force recoding applications

Solution with LeanXcale

- Single data manager for all big data
  - Fast data, operational data, historic data
  - Fast data loading, very high transaction throughput
- Data ingested directly through KiVi at very high rates
  - Very small hardware footprint, open source pricing
  - Readily accessible using Spark or other tools through JDBC driver
- Fast analytical queries through LeanXcale OLAP query engine over operational and historical data

6. Trends

- Big data boosts the growth of both approaches
- Market forecasts
  - Data lake: from $8 billion in 2019 to $14 billion in 2023 (source: Market Research Future)
  - DW: from $10 billion in 2019 to $20 billion in 2024 (source: Market Analysis)
- Strong scope for technological progress
  - Open source: Hadoop, NoSQL, NewSQL and HTAP, etc.
  - Analysis tools: statistics, machine learning, data mining, etc.