<https://www.bankinghub.eu/banking/technology/big-data-in-banking-data-lake-instead-of-data-warehouse>

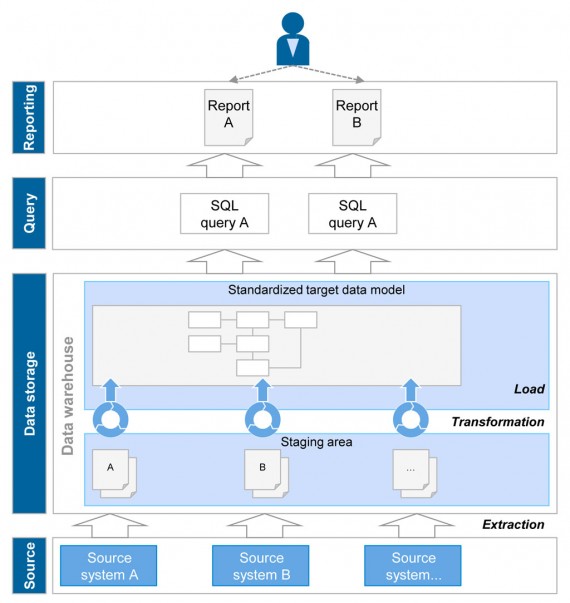
# BIG DATA IN BANKING: DATA LAKE INSTEAD OF DATA WAREHOUSE?Requirement-compliant provision of operational data

IT departments of banks are increasingly facing the challenge that the need for settlement system-related information and data and their availability for analyses and evaluations is steadily growing. Previously, settlement systems mostly worked independently and transferred data via defined interfaces or reports in an agreed schedule, e.g. for controlling purposes or sales. Nowadays there are ever more requirements demanding real-time and flexible analyses of the entire operational data warehouse. Examples are the increasing regulatory initiatives and requirements for timely and more detailed information or requirements of the digitalization strategy of many institutions assuming the availability of real-time settlement information in the sales applications.

A direct access to the settlement systems is usually not possible for reasons of stability and performance of the settlement systems. Thus, for example, turnover bookings in the daily business aren’t to be influenced by analyses for the customer portal. But also technical hurdles are a point against a direct access to the operational systems if they, for instance, are located on the host and decentralized applications are to be serviced or cost implications for transaction-oriented settlement models.

## TRADITIONAL DWH: EXTRACT, TRANSFORM, LOAD

Setting up a data warehouse (DWH) in which the required operational data is saved, prepared and analyzed used to be a tried and tested solution pattern for fulfilling settlement-oriented information needs.

*Figure 1: Draft of principles of traditional DWH architecture*

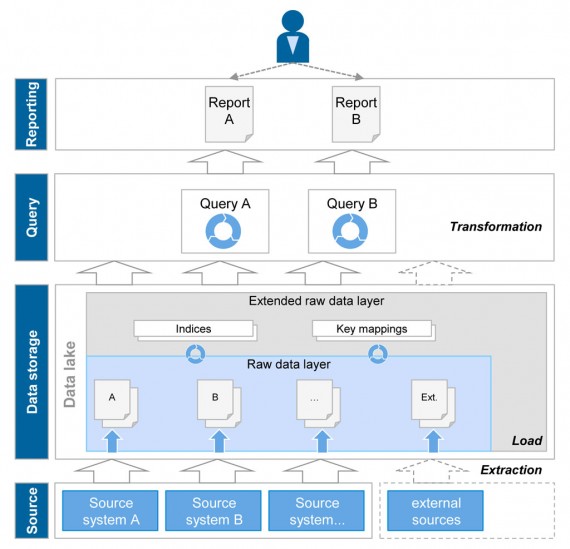
The required operational data is selectively extracted from the source systems and buffered on the entry level of the DWH (staging area). The extraction is usually made at regular intervals (daily, monthly, …) and at fixed dates (e.g. after final daily processing). Data is incrementally prepared, standardized and transformed in the target data structure within the DWH. The target data model is afterwards persisted whereas the initial data is usually overwritten with the next loading. Analyses exclusively access the target data model that comprises a consolidated view on the corresponding data warehouse. A relational database management system with a standardized SQL access for queries serves as a technical basis for a traditional data warehouse.

However, this concept reaches its limits with an increasing scope of requirements.

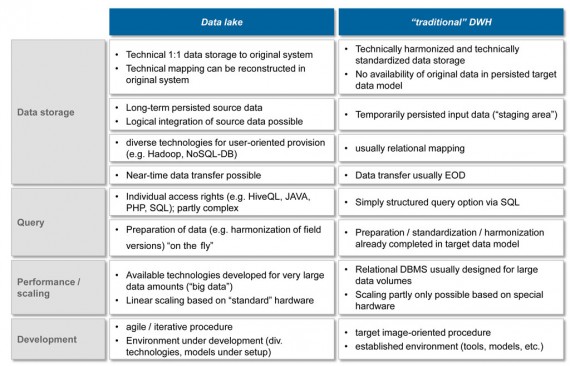
* **High level of implementation efforts:**The transformation and standardization of the initial data in a harmonized data model represent the core of the DWH concept. This procedure leads to a comparatively high level of implementation efforts since the target data model has first to be elaborated and discussed at large. Afterwards, each transformation has to be described in terms of its concept, implemented and tested before specific analysis can be conducted.
* **Restricted flexibility:** Solely the target data model defined beforehand is available for the analysis in the traditional data warehouse. A flexible integration of additional data sources, e.g. for ad-hoc analyses is usually not possible and has to be conducted outside the DWH environment.
* **No raw data:**The aim of a data warehouse is the standardization and similar mapping of comparable issues in order to simplify analyses at a later stage. This often makes sense for regulatory issues or in the fields of bank management (e.g. standardized mapping of all credits) but contradicts more operational requirements where detailed information on individual products is required (e.g. specific conditions for construction financing).
* **Data volume:**Basically, data warehouses can deal with huge amounts of data, however, need special database and storage hardware if the scope increases. Thus, the costs typically grow exponentially in case of major data volumes.

## DATA LAKE: FIRST LOAD, THEN TRANSFORM

The data lake concept provides a remedy by making all required data for analysis purposes available at one place in a central and unchanged manner. In the next step only, a selective data preparation within specific analyses takes place. In this context, performing, flexible and cost-efficient processing is to be ensured by using big data technologies.

*Figure 2: Draft of principles of data lake architecture*

The initial data of the source systems is directly loaded into a raw data layer pursuant to the data lake concept without any logical transformations. The technology is usually not based on a relational database but on file system structures (Hadoop) or key value structures due to the aforementioned restrictions. Both approaches offer the advantage to dynamically deal with structural changes on the part of the source system and to process heterogeneous data formats (also unstructured). The raw data is, if needed, supplemented by key mappings or indices in an expanded raw data layer in order to allow for performing access, but doesn’t change its structure. Queries and analyses are directly made on the raw data layer but are much more complex in comparison with the traditional DWH since the harmonization and consolidation of information (transformation) have to be made as part of the queries in this context. Various big data technologies are available for defining and processing these queries that ensure a high and cost-efficient scalability via distribution mechanisms based on standard hardware. Since the data transformation is only made at the term of the queries, a flexible integration of further data (e.g. external information) is quite easy. Usually external data is not persisted in the data lake but only published via a link in the metadata.

*Figure 3: Comparison of data lake and traditional DWH*

## PREREQUISITES FOR A DATA LAKE APPROACH NOT TO BE UNDERESTIMATED

The data lake approach is a promising alternative apart from the traditional data warehouse concept, however, it requires the fulfillment of a series of framework conditions.

The metadata and its management play a pivotal role because they are indispensable for the dynamic and proper access to the raw data. In addition to a subject-related and technical description of the data lake content, the metadata is to contain also information whether the raw data can be merged and transformed. The creators of queries and/or the BI tool used have to be able to find out from the metadata if and how the customer data from system A can be combined with the product data from system B. A consolidated key relationship such as in the traditional DWH ensuring a standardized combination for all analyses is not provided in the data lake.

As a result, the overall requirements for the users of a data lake are on the increase. Free analyses of data should only be carried out by experts (data scientists) who know the relationships of the raw data very well. Predefined queries and reports or sub-data portfolios with a comprehensive metadata description, however, can be used by several users. Thus, the data lake often differentiates between an exploratory expert area and a secured user area. All in all, it can be concluded that complexity and expenses are shifted from data storage to report and analysis setup.

Besides, a major challenge is the security issue. The benefits of the data lake increases the more detailed and varied the available data is. At the same time, not all users should be granted access to the entire data warehouse of an institution, in particular from compliance perspective. This requires respective security mechanisms restricting the access as far as needed, but as little as possible.

## REQUIREMENTS DEFINE THE RIGHT CONCEPT

The question of the better concept for providing operational data for analyses cannot be answered in a general way but depends on the specific requirements. If the aforementioned framework conditions for setting up a data lake can be fulfilled, this concept can be used for the following requirements:

* **Access to original data structures:**The provision of raw data is a core element of the data lake concept. Depending on the historicization and replication concept, raw data with a long history and/or single changes of the state can be made available. Typical use cases can be found, for instance, in the fields of Compliance and Auditing.
* **Low standardization degree:** The advantage of this concept is the high level of flexibility for queries and analyses on a broad data basis that is, for example, required for issues in the fields of data mining and data exploration or for ad-hoc queries in the regulatory context (QIS, AQR). The concept can also be used for standard queries. Here, the traditional data warehouse offers specific advantages due to the transformation of the data beforehand.
* **Major data volume:** The implementation of the data lake concept is based on the use of big data technologies that were especially designed for dealing with major amounts of data. Apart from the high performance via distribution mechanisms, the cost advantages from the usability of standard hardware become apparent, too.
* **Neartime availability:** Raw data can be timely provided thanks to the lacking transformation necessities. A nearly simultaneous data provision of the data lake is made possible via specific replication mechanisms, if necessary. Consequently, the data lake can be used as a data source for online banking, for example, without increasing the burden for the operational systems.

The traditional DWH still has its strengths if the analysis focus has a defined scope and if standardized queries are made oftentimes. It is also the preferred concept if a consolidated overall view is required that has to be consistently queried from different users.

Both concepts can be combined in a sensible way. Thus, a data lake can serve as a staging area for a traditional DWH since the raw data only has to be provided once for different requirements and is available for a long time. At the same time, the target data model of a DWH can be made available in the data lake which provides its consolidated data warehouse for more flexible analyses.

A data lake can be set up as addition to the existing DWH and can be incrementally integrated in the existing system landscape.

<http://www.toadworld.com/platforms/oracle/w/wiki/11576.modern-data-pipeline-architectures>

Modern data pipeline architectures

* [Introduction](http://www.toadworld.com/platforms/oracle/w/wiki/11576.modern-data-pipeline-architectures" \l "introduction)
* [Traditional data warehouse architecture](http://www.toadworld.com/platforms/oracle/w/wiki/11576.modern-data-pipeline-architectures" \l "traditional-data-warehouse-architecture)
* [Modern pipeline architectures](http://www.toadworld.com/platforms/oracle/w/wiki/11576.modern-data-pipeline-architectures" \l "modern-pipeline-architectures)
* [Conclusion](http://www.toadworld.com/platforms/oracle/w/wiki/11576.modern-data-pipeline-architectures" \l "conclusion)

Written by [*Juan Carlos Olamendy Turruellas*](http://www.toadworld.com/members/juan-carlos-olamendy)

## **Introduction**

In this article, I want to talk about the evolution of data pipeline architectures from the traditional centralized, batch-oriented and report-only data warehouses towards the modern one based on distributed data stores, distributed computing, near real-time processing and the used of machine learning and analytics to support decision making process in today fast-changing business environment.

Big data has matured and become in one of the pillar of any business strategy today, specifically in the area of sales and digital marketing in order to increase the revenue and customer loyalty and expectation. In a highly competitive and regulated environment, businesses are required every day to make decisions based on data, instead of intuition. In order to make good decisions, it’s necessary to process a huge amount of data in an efficient way (the less possible computing resources and a minimum processing latency), add new data sources (structure, semi-structure and unstructured ones such as UI activities, logs, performance events, sensor data, emails, documents, social media, etc) and support the decisions using machine learning algorithms and visualization techniques.

Some companies such as *Netflix* are publicly declared data-oriented because their core business, products and services are based on insights derived from data analysis.

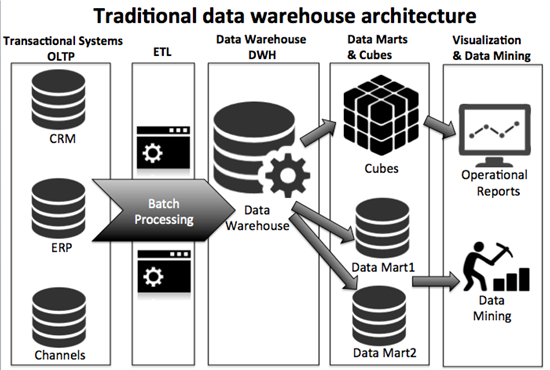
So, let’s see how we can transform our traditional data warehouse architecture into a modern one to support the challenges related to big data and high computing.

## **Traditional data warehouse architecture**

A traditional data warehouse architecture comprises of the following elements:

* Transactional systems (*OLTP*). Produce and record the transactional data/facts resulted from business operations
* A data warehouse (*DWH*). Centralized data stores that integrates and consolidates the transactional data
* *ETL* processes. Batch processes that move the transactional data from *OLTP* towards *DWH*
* Data marts and cubes. Representing basically a derived and aggregated view of the *DWH*
* Analytical and visualization tools. Enabling the visualization of data stored in the data marts and cubes for reporting and auditing purposes. They do first-generation analytics using data mining techniques

This kind of architecture can be illustrated in the following figure.



**Figure 01**

This architecture has some drawbacks as shown below:

* Data sources are limited only to transactional systems (*OLTP*).
* The major workload is based on *ETL* batch processing (jobs). It’s well-known that there is a loss of data in this step
* The integration and consolidation of data is very complex due to the rigid nature of *ETL* jobs.
* Data schema is not very flexible (in the OLTP, DWH, data marts and cubes sides) to be extended for new analytics use cases.
* It’s very complex to integrate semi- and un-structure data sources. So, we lose very important information in the form of log files, sensor data, emails and documents.
* It doesn’t support naturally real-time and interactive analytics. It’s thought to be batch-oriented.
* It’s very limited when we need to scale the solution.
* It’s designed to be used in on-premise environment, so it’s very complex to extend and deploy in cloud- and hybrid-based environments.

So, in order to overcome the limitations of the previous architecture, we need to think using new paradigms.

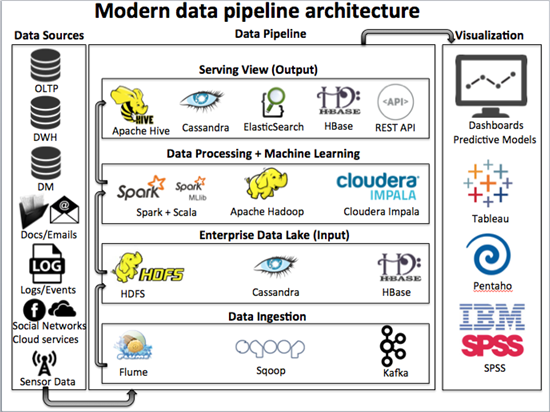
## **Modern pipeline architectures**

The modern pipeline architect is an evolution from the previous one integrating new data sources and using new computing paradigms as well as the integration of artificial intelligence, machine learning algorithm and cognitive computing.

In this new approach, we have a pipeline engine with the following features:

* Unified data architecture for all the data sources no matter the structure of the origin. Integration with existing data marts and data warehouses. It appears the concept of Enterprise Data Lake
* Flexible data schemas designed to be changed frequently. Use of *NoSQL*-based data stores
* Unified computing platform for processing any kind of workload (batch, interactive, real-time, machine learning, cognitive, etc). Use of distributed platforms such as*Hadoop*, *Spark*, *Kafka*, *Flume*, etc
* Deployable on hybric- and cloud-based environments
* Horizontal scalability, so we can process unlimited data volume by just adding new nodes to the cluster

Although, there are a huge amount (as big as the big data itself) of technologies related to big data and analytics, I’ll show a referential architecture for a modern data pipeline. I’ll illustrate the functional aspect of every layer using particular technologies for you to research further on this and learn more (see the figure 02).



**Figure 02**

From this referential architecture, we can derive specific use cases to be used in your business.

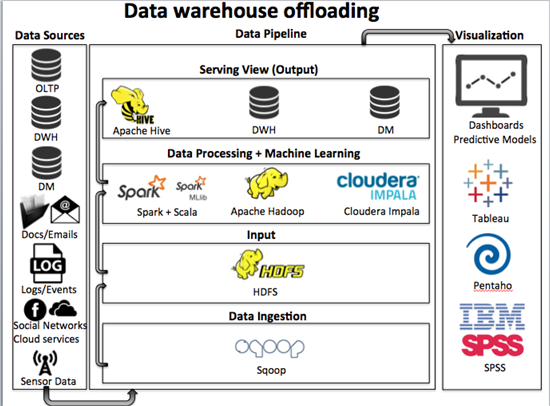
### ***Use Case 01. Data warehouse offloading***

Data warehouse has being in major companies for many years. With the exponential growth of data, the *DWH*s are reaching their capacity limits and batch windows are also increasing putting at risk the *SLA*. One approach is to migrate heavy *ETL* and calculation workloads into *Hadoop* in order to achieve faster processing time, lower costs per stored data and free *DWH* capacity to be used in other workloads.

Here we have two major options:

* One option is to load raw-data from *OLTP* into *Hadoop*, then transform the data into the required models, and finally move the data into the *DWH*. We can also extend this scenario by integrating semi-, un-structured and sensor-based data sources. In this sense, the *Hadoop* environment acts as an *Enterprise Data Lake*.
* Another option is moving data from the *DWH* into *Hadoop* using *Sqoop* in order to do pre-calculations, and then the result is stored in data marts to be visualized using traditional tools. Because the storage cost on *Hadoop* is much lower than on a *DWH*, we can save money and keep the data for longer time. We can also extend this use case by taking advantage of analytical power and creating predictive models using *Spark MLLib*, *Mahout* or *R* language or cognitive computing using *IBM Watson* in order support future decisions of the business.

We can visualize this scenario as shown in the figure 03.



**Figure 03**

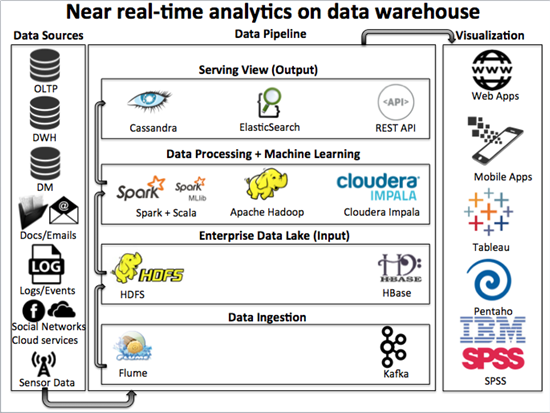
### ***Use Case 02. Near real-time analytics on data warehouse***

In this scenario, we have *Fume* agents installed on every data source for ingesting data into the pipeline. We can also use *Kafka* as a streaming data source (front-end for real-time analytics) to store the incoming data in the form of events/messages. As part of the ingestion process, the data is stored directly on the *Hadoop* file system, or some scalable, fault torelant and distributed database such as *HBase* or *Cassandra*. Then the data is computed and some predictive models are created using *Spark*, *Scala* and*MLLib* technologies. The result is stored in ElasticSearch for improving the searching capabilities of the platform, predictive models can be stored in *Hadoop* file system and the result of calculations can be stored in *Cassandra*. The data can be consumed by traditional tools as well as by Web and mobile applications via API Rest.

This architectural design has the following benefits:

* Add near real-time processing capabilities over batch-oriented processing
* Lower the latency for getting actionable insights, impacting positively on the agility of the business on making decisions
* Lower the storage cost significatively comparing to traditional data warehousing technologies because we can create commodity and low-cost cluster of data and processing nodes on-premise and in the cloud
* This architecture is prepared to be migrated to the cloud and take advantage of elasticity, so adding computing resources when the workload is increasing and relieving computing resources when they’re not needed

We can visualize this scenario as shown in the figure 04.



**Figure 03**

## **Conclusion**

In this post, I've talked about the evolution of data pipeline architecture towards modern ones today. Using the architectural patterns and strategies explained before, you can adapt your data pipeline architectures to be more scalable, resilient and help make better decisions in today changing world.