DATA ANALYTICS AND BIG DATA

A HARD ENOUGH EXPLANATION FOR NON-TECHNICAL ROLES

PART 3

BIG DATA DOESN'T HURT

OR HOW TO UNDERSTAND SOMETHING ABOUT A COMPLEX GEEK CONCEPT WHITHOUT FEEL THEY ARE SPEAKING GIBBERISH

By TECHBIZDESIGN.COM

Big data doesn't hurts is...

About easy concepts

Myths, basic definitions, importance for society…

About change in Data paradigm

Datawarehousing vs Big Data, history, differences…

About Architecture and components

Hadoop, spark, hive and some other swear-words …

About Big data process

Ingestion, storage…

About the State of the art

Vendors offering, trends…

What is Big Data? Which are its main characteristics?

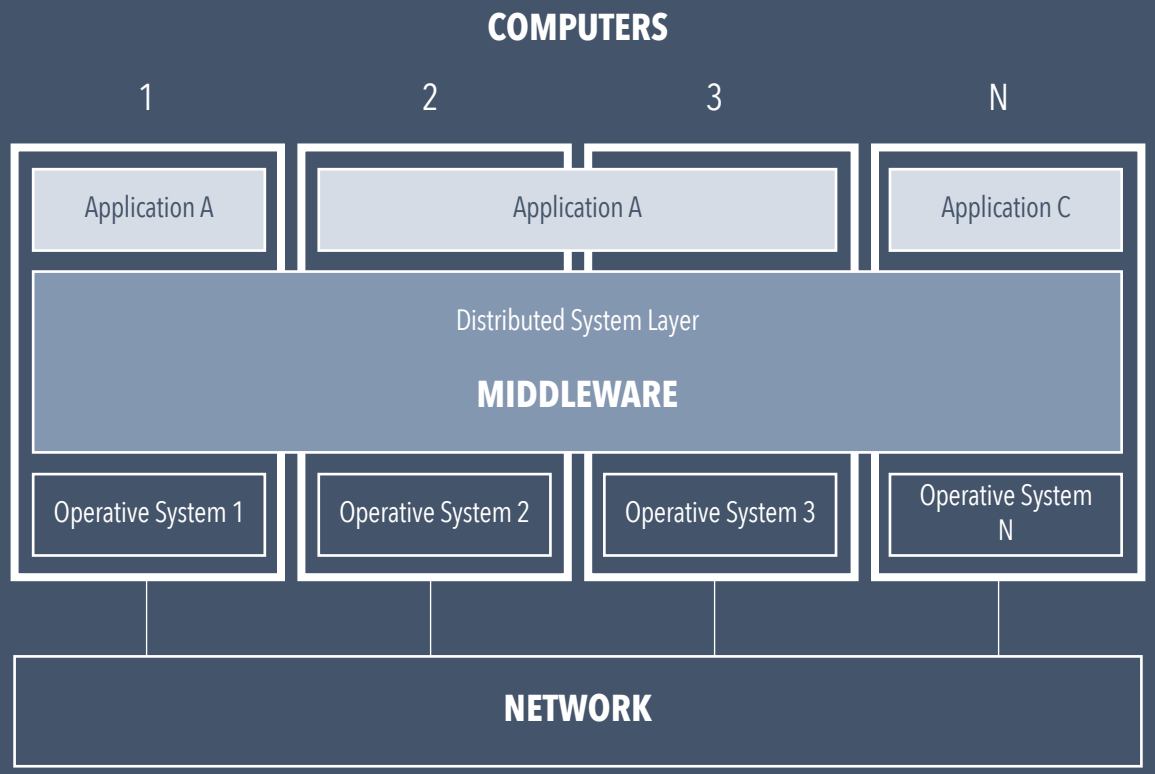
BIG DATA relevant concepts: It's a distributed system

Big Data technology is a **distributed system** composed by N machines, depending of Application needs

They are composed by **independent nodes** connected by the **same network**

But for an external user, they look as a **single machine** and act as well

They are **complex** systems where concepts as redundancy, availability or scalability are crucial



BIG DATA relevant concepts: The famous 3 Vs

VOLUME

Volume is probably the **best known characteristic** of big data; this is no surprise, considering more than 90 percent of all today's data was created in the past couple of years. The current amount of data can actually be quite staggering

VELOCITY

Velocity refers to the speed at which data is being **generated, produced, created, or refreshed.**

Sure, it sounds impressive that Facebook's data warehouse stores upwards of 300 petabytes of data, but the velocity at which new data is created should be taken into account. Facebook claims 600 terabytes of incoming data per day.

VARIETY

When it comes to big data, we don't only have to handle structured data but also semi-structured and mostly unstructured data as well. As you can deduce from the above examples, most big data seems to be unstructured, but besides audio, image, video files, social media updates, and other text formats there are also log files, click data, machine and sensor data, etc

Icons by H Alberto Gongora from the Noun Project

BIG DATA relevant concepts: Two other Vs come later

VERACITY

This is one of the unfortunate characteristics of big data. As any or all of other Vs increase, the veracity (**confidence or trust in the data**) drops. This is similar to, but not the same as, validity or volatility. Veracity refers more to the provenance or reliability of the data source, its context, and how meaningful it is to the analysis.

VALUE

Maybe the **most important of all**, is value. The other characteristics of big data are meaningless if you don't derive business value from the data.

Substantial value can be found in big data, including understanding your customers better, targeting them accordingly, optimizing processes, and improving machine or business performance

Icons by H Alberto Gongora from the Noun Project

BIG DATA relevant concepts: And lastly five more were included

VALIDITY

Similar to veracity, validity refers to how **accurate and correct the data is for its intended use**.

The benefit from big data analytics is only as good as its underlying data, so you need to adopt good data governance practices to ensure consistent data quality, common definitions, and metadata

VULNERABILITY

Big data brings new security concerns. After all, a data breach with big data is a big breach. Does anyone remember the infamous AshleyMadison hack in 2015?

But, unfortunately there have been many big data breaches as the one starred by a hacker called Peace posted data on the dark web to sell information on 167 million LinkedIn accounts in May 2016.

VOLATILITY

How old does your data need to be before it is considered irrelevant, historic, or not useful any longer? How long does data need to be kept for?

Before big data, organizations tended to store data indefinitely -- a few terabytes of data might not create high storage expenses; it could even be kept in the live database without causing performance issues

VISUALIZATION

Another characteristic of big data is **how challenging it is to visualize**.

Current big data visualization tools face technical challenges due to limitations of in-memory technology and poor scalability, functionality, and response time. You can't rely on traditional graphs when trying to plot a billion data points, so you need different ways of representing data such as data clustering or similar techniques.

VARIABILITY

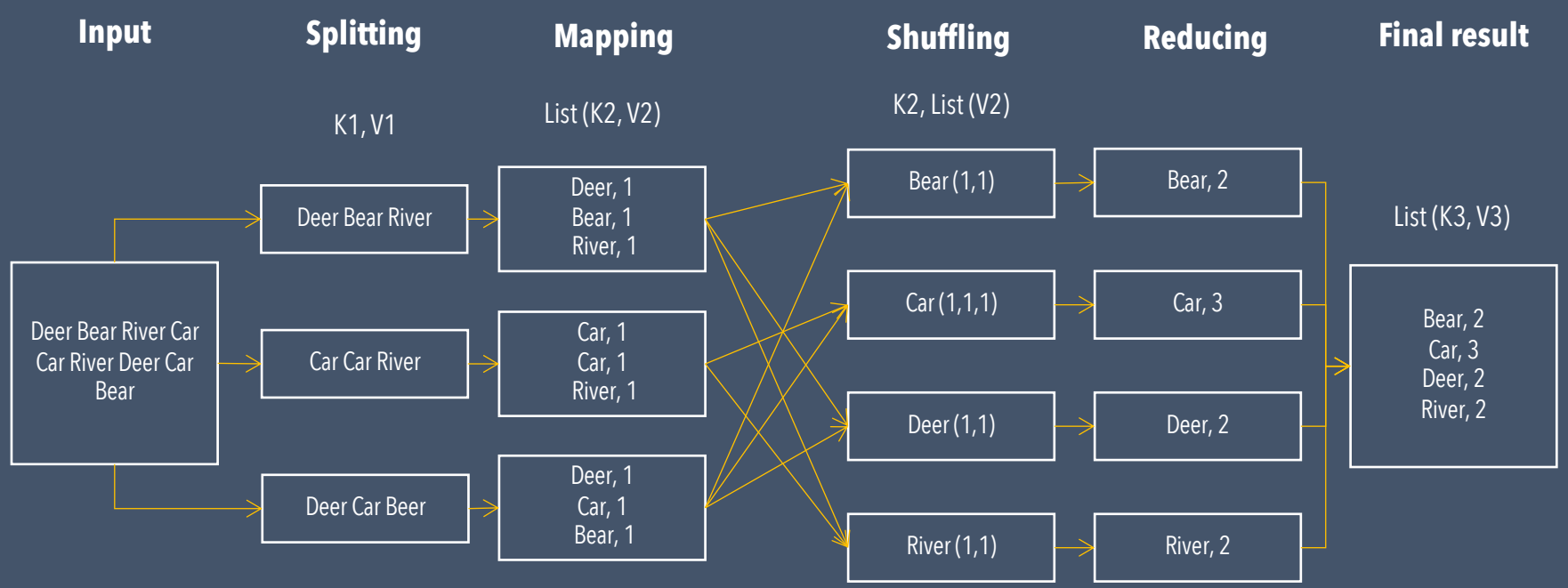
Variability in big data's context refers to a few different things. One is the **number of inconsistencies** in the data.

Big data is also variable because of the multitude of **data dimensions** resulting from multiple disparate data types and sources. Variability can also refer to the inconsistent speed at which big data is loaded into your database

BIG DATA relevant concepts: Map Reduce

This concept could be easily called "**divide and conquer**"

For a very complex and time-consuming problem, the approach is simple. We split it in a small problems and act over them to get small solutions. Then we aggregate them to get an overall solution.



BIG DATA relevant concepts: CAP theorem

It's impossible for a distributed data store to simultaneously provide more than two out of the following three guarantees:

C for **Consistency**

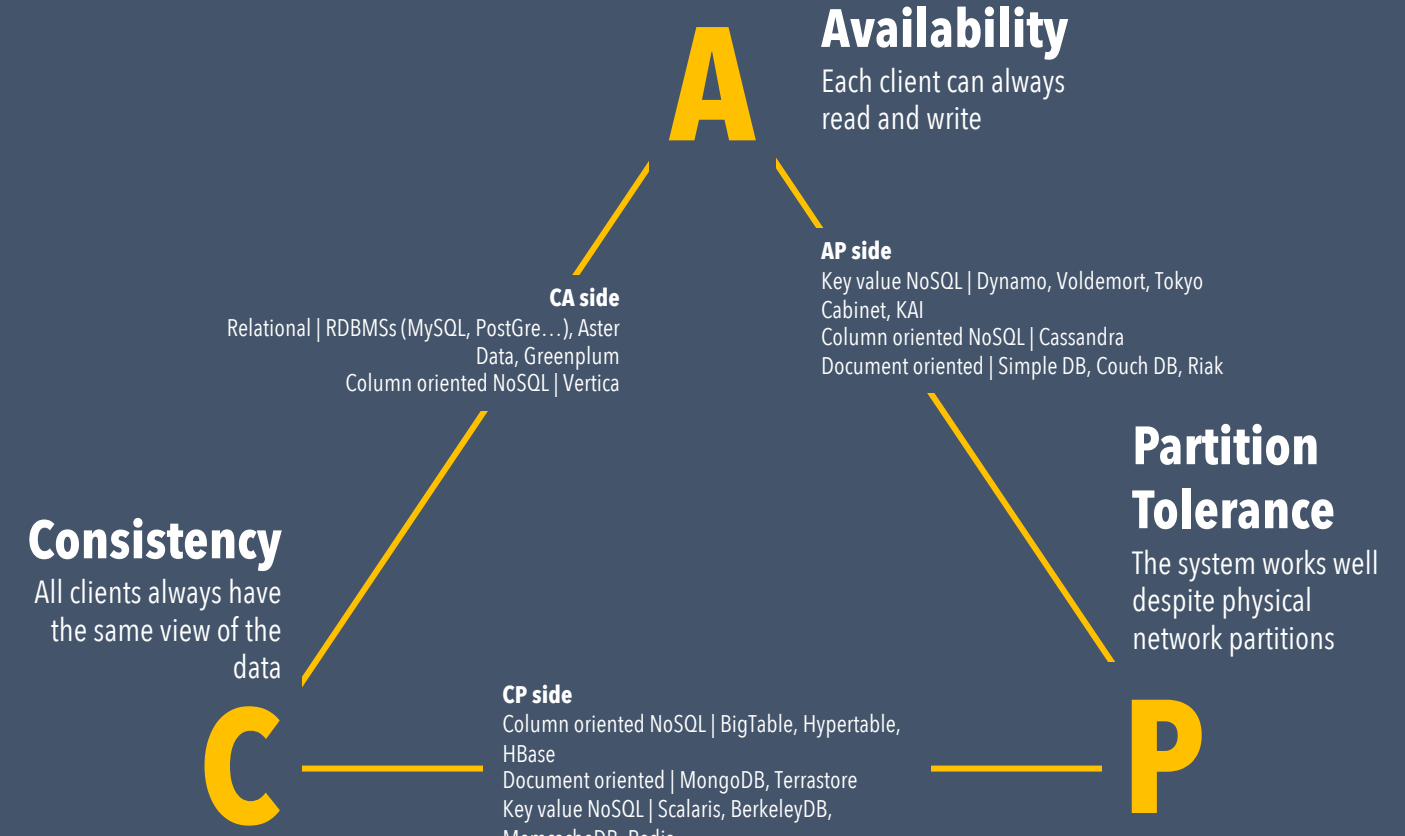
Every read receives the most recent write or an error

A for **Availability**

Every request receives a (non-error) response, without the guarantee that it contains the most recent write

P for **Partition Tolerance**

Partition tolerance: The system continues to operate despite an arbitrary number of messages being dropped (or delayed) by the network between nodes



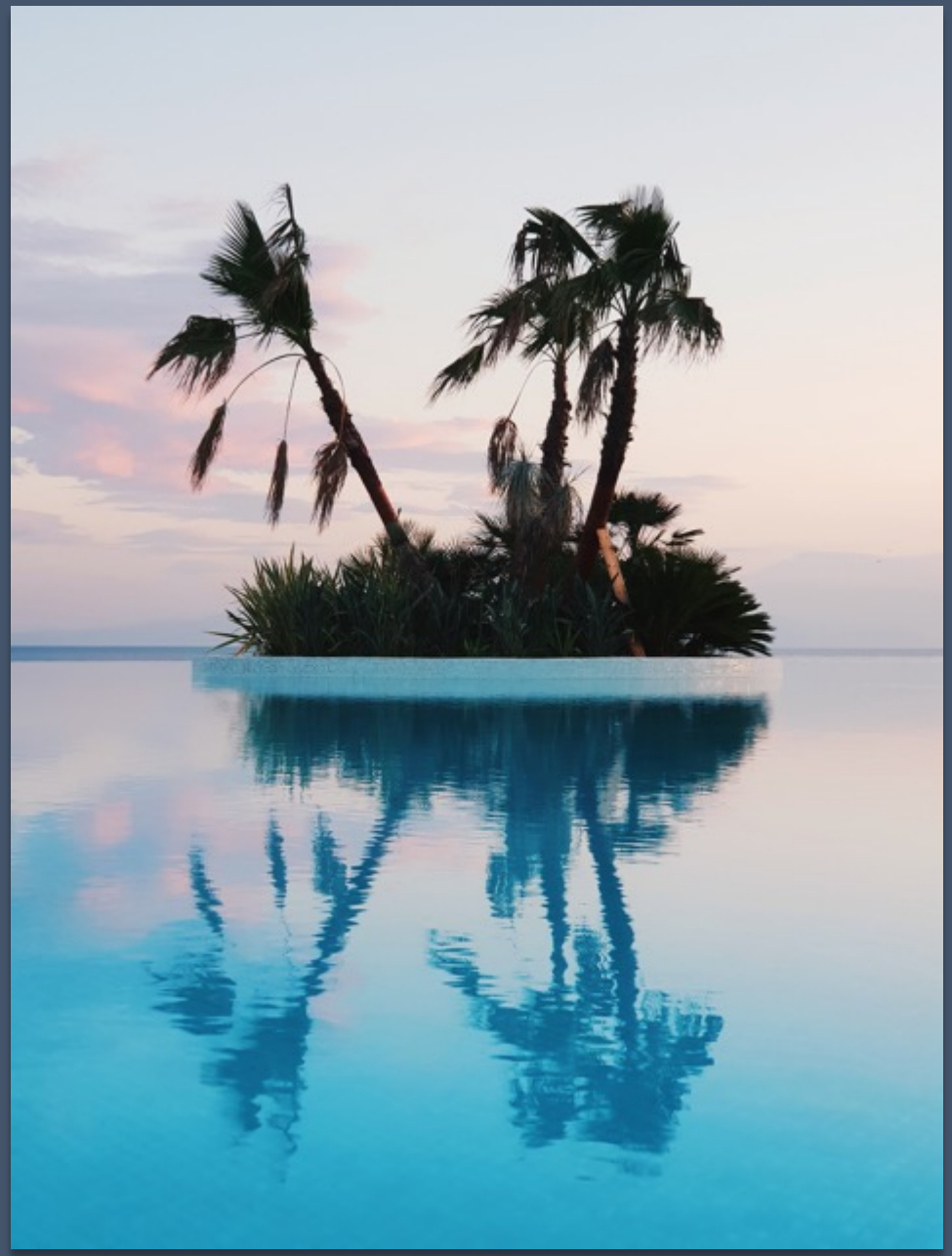
Why did Big Data appear? What is its origin?

Islands of data

We've evolved from centralized mainframes and first Relational Data Base Management Systems (RDBMS) to distributed and decentralized computing. Today we use intensively PCs, laptops and mobile devices into our enterprise applications.

The result: we've wound up with an islands of data problem, because critical data we need for reports and analysis is **scattered among numerous different applications and systems**

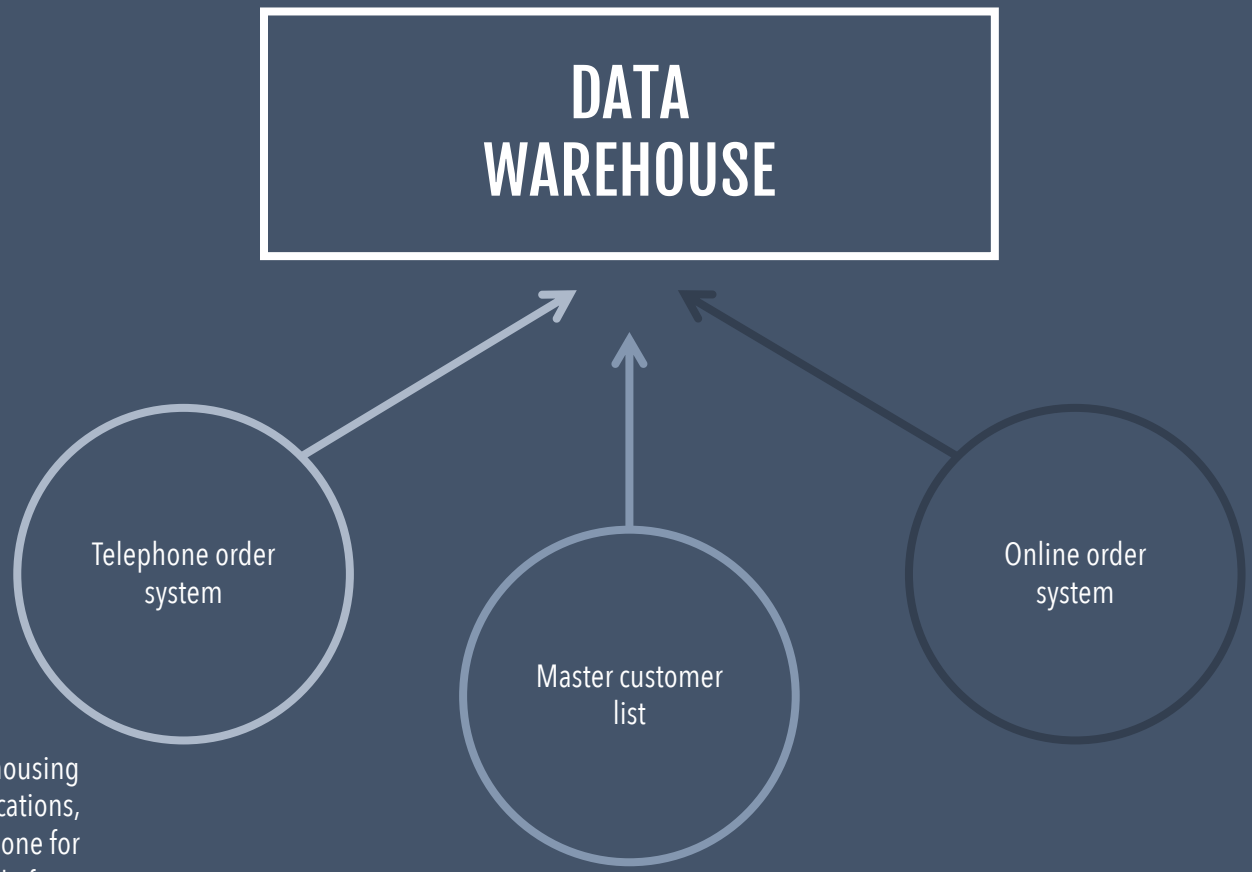
The consequence of our islands of data problem is that when we need to build reports that require data from more than one different source, it's typically been, over the years, a very **difficult and time-consuming effort**.



The beginning: Datawarehousing

We used to work with our traditional RDBMS. Then, at early 1990s, introduced us to the concept of data warehousing

We've attempted to break through these islands of data and bring selected data from each of our different applications together into a **single store of information**, and that's where we will run our reports and do our analysis.



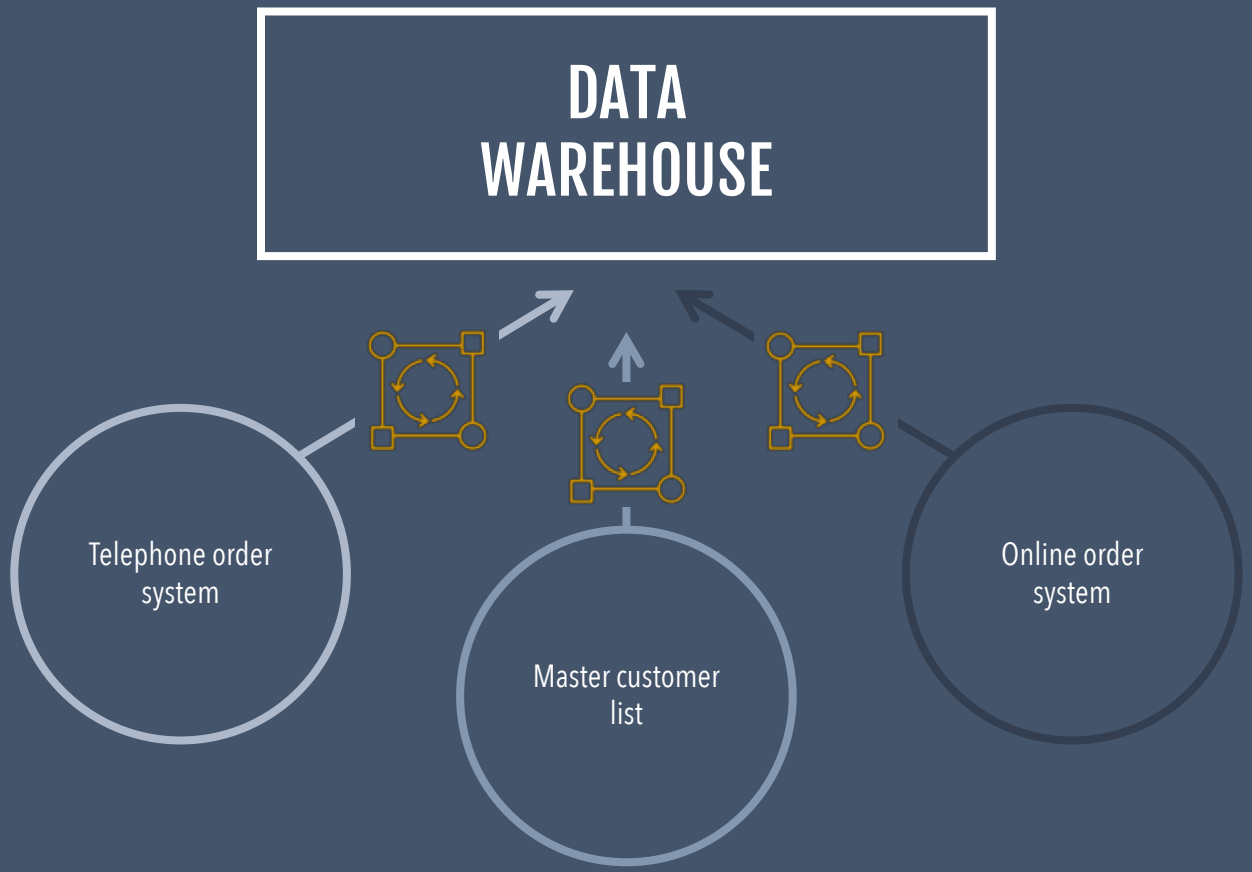
This picture shows a very simple data warehousing architecture where we have three different applications, one for taking orders over the telephone, another one for doing orders online, as well as a master list of all of our customers. And selected data from each of those systems is fed and copied into the data warehouse

Datawarehousing - DWH

We pursued the **transformation of external data**, or in other words, the unification of data. To do that, we apply new Data **selection techniques** based on a very rigorous requirements analysis process according to the reporting and analytics needs.

There was a **continuous update** to keep relevance and **Data consolidation** periodic processes by subjects and structured internally based on a strong intentionality.

The reason for Datawarehousing? To provide **one-stop shopping for our data** within the enterprise.

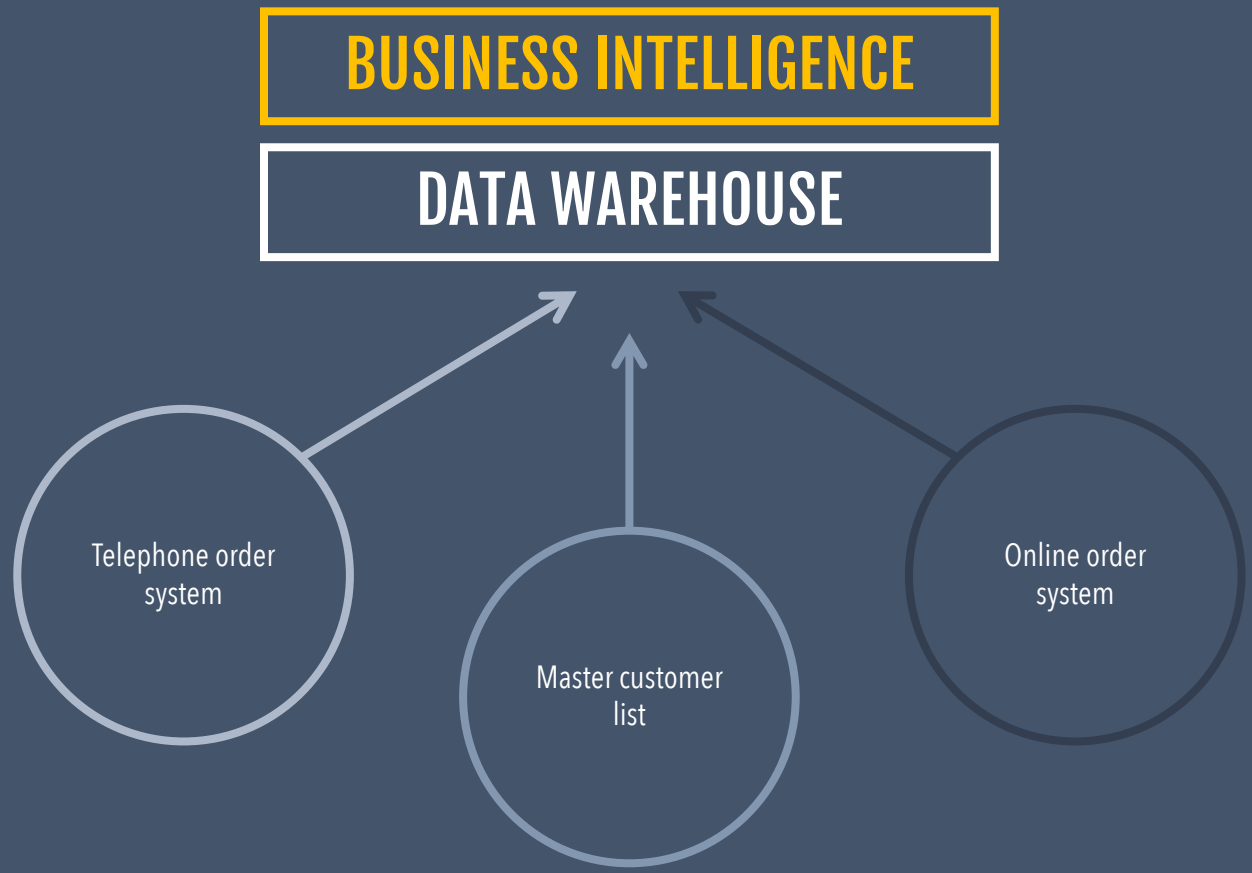


Datawarehousing and Business Intelligence (BI) connection

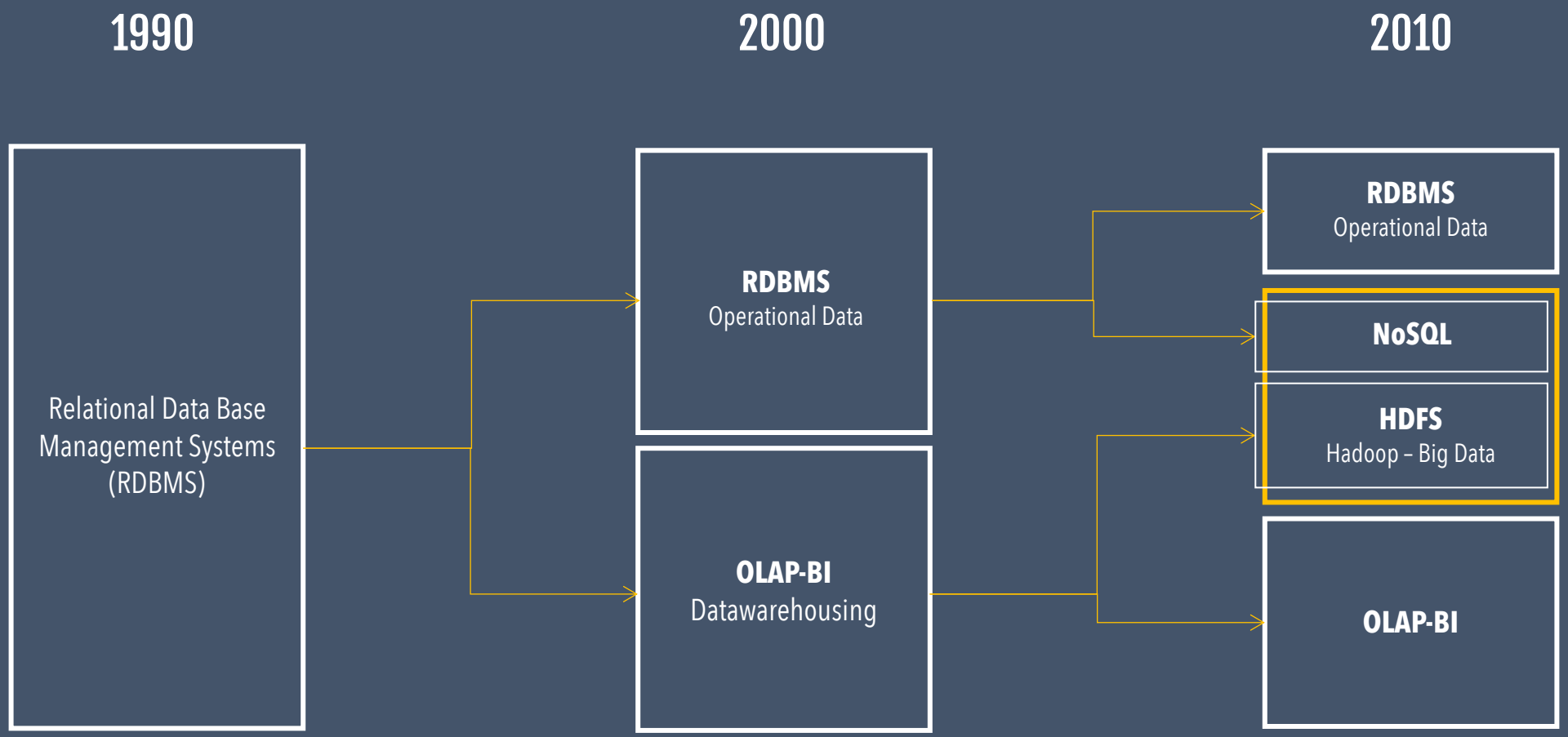
We need more than just that an integrated set of data for the insights we're after, and that's where BI fits in. We define BI as our attempts to draw critical insights from our data.

So, Business intelligence, or BI, began around the same time that data warehousing did in the late 1980s and early 1990s.

The idea behind business intelligence is that while a data warehouse should provide one-stop shopping for our data, the BI systems will then provide the **one-stop shopping for those data-driven insights**.



A consistent evolution towards Big Data



Main shortcomings with traditional Datawarehousing and Business Intelligence

In DWH there's a strong predominance of **backwards-looking**, a kind of rearview mirror. It has limited capabilities for understanding what is happening right now and rarely provides predictive or discovery analytics. This provokes a dimensional analysis of facts and lack of hypotheses that makes **DWH not suitable for prescriptive analytics**.

Some-but not all-data

The limitations of the data management technologies that we've had to work with require us to bring in some, but not all, of the data from within our enterprise and from different sources outside.

It seems **limited to structured data**

Significant upfront business requirements analysis

What that means, then, is that we need to engage in this **significant upfront effort** before we can begin building a data warehouse.

And as our requirements and needs change over time, it's increasingly **difficult to keep our DWH and our BI capabilities totally up to date** with the things we need them to provide.

Batch-oriented updates from sources

In some cases, the currency of our data within our data warehouse is not necessarily where it needs to be for the most timely, most critical business decisions we need to make.

So, **real time and agile decisions are far** from the capabilities of DWH.

Other pain points with traditional DWH and BI

About source systems

It can deal with a subset of available resources. We have to select data from each source with a significant upfront analysis. This means that **adding new data source is time-consuming**.

About DWH

It's based originally on a set of **architectural rules limited**, essentially read-only copy of some data.

It's heavily driven by business rules often impacted by a lack of clarity and agreement.

In addition, **data quality often an issue**.

About methodology

There are two main methodologies to design an build a DWH: Data first or functionality oriented. Both needs a deep understanding of organizational data, a general idea of how organization will use data. That issue makes **DWH are normally built slowly**.

In the other hand, **enhancing DWH is always time-consuming** and often leads to separate, faster and fragmented solutions.

About data feeds - ETL

DWH are traditionally batch-oriented. You need to build **different architecture for real-time purposes**.

As is heavily driven by business rules, performance is difficult to manage by **inflexibility**.

But Datawarehousing and Business Intelligence still have a strong role

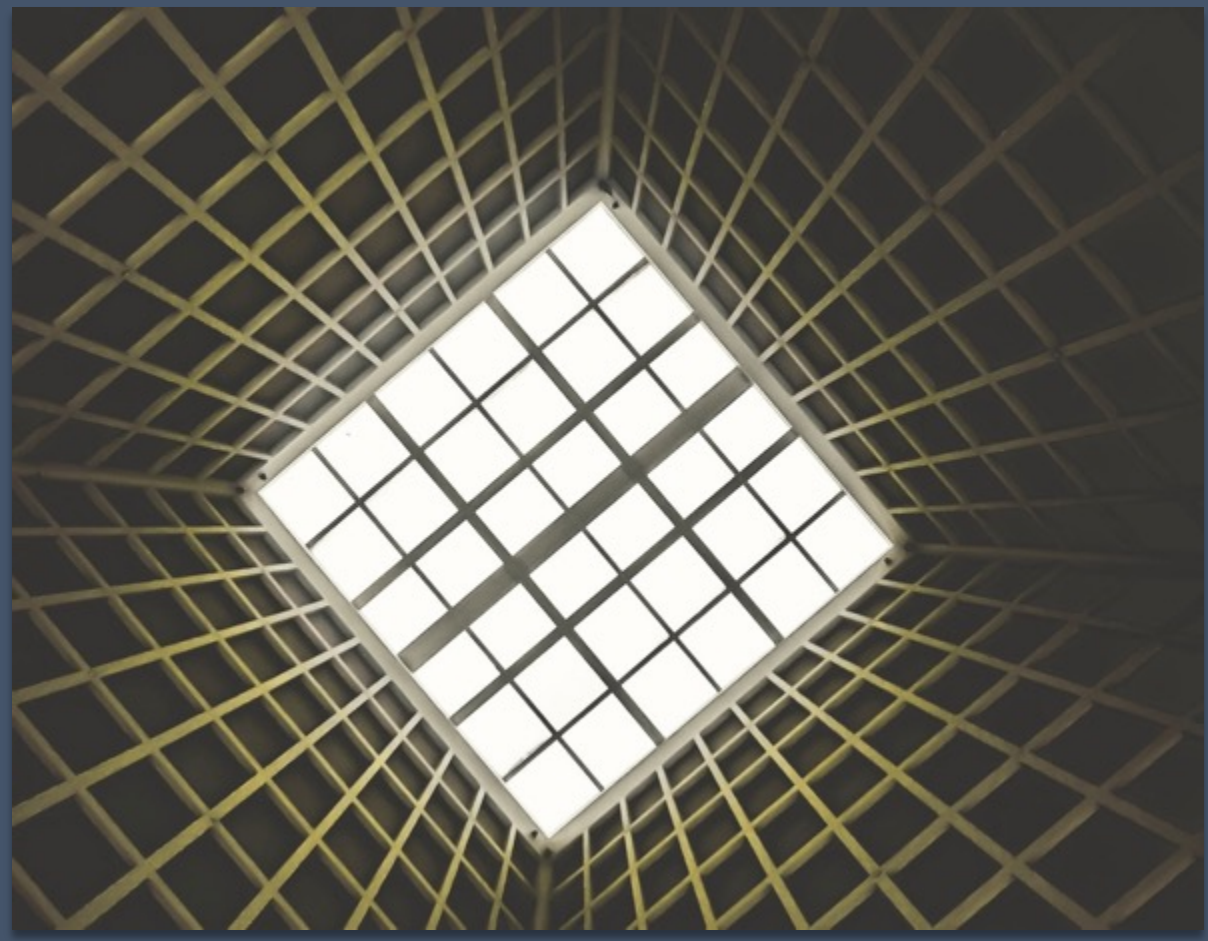
Because it's perfect for dimensional "**slice and dice**" analysis

Because many business decisions are made from BI-driven insights

Because it's a **mature proven technology**

Because there are many skilled DW and BI professionals

And because it's part of a a **emerging and broader analytical architecture**



Which are the differences with Big Data? What is the process now?

The Big Data paradigm – Mantras to follow

1

Ingest and store now

2

Organize later

3

We don't need to follow detailed requirements

4

We're beyond relational database rules

A paradigm change – From DWH ETL to Big Data ELT

ETL process in Datawarehousing

With a data warehouse in ETL, our objective is to **have clean, standardized data only that is closely aligned with specific requirements** and that's what makes its way into the data warehouse

ELT process in Big Data

With Big Data and ELT, our objective here is different, it's to **have as much data as possible as quickly as possible**, even if we don't necessarily have specific requirements or know that, in fact, we may ever even use that data

A paradigm change – From DWH ETL to Big Data ELT

ETL

EXTRACT

Dependent on specific requirements | Limited data set | Quick ingestion | No data quality worries | Updated on a schedule

STAGING AREA

TRANSFORMATION

Driven by business rules | Driven by specific requirements | Significant data cleansing | Objective: cleansed and normed data ready to load into DW

LOAD

Simply write on the proper space inside the DW

ELT

EXTRACT

Theoretically, all source data can be extracted | Practically, will still do some selection of source data | Don't need deep requirements to rive extraction | Mixture of bulk load and streaming

LOAD

Direct pipeline to extract process| Load the data that has been extracted | No cleansing or norming before loading

TRANSFORMATION

Transform as needed | Data mining can still use "dirty" data | Downstream relational EDW? Transformation then | Infrastructure serving as EDW? Set aside "clean data" area

STAGING AREA ANALYTICS SANDBOX

↓

TRANSFORM

↓

STANDARDIZED DATA SECTOR

ETL

Relational Data (Super sized staging strategy)

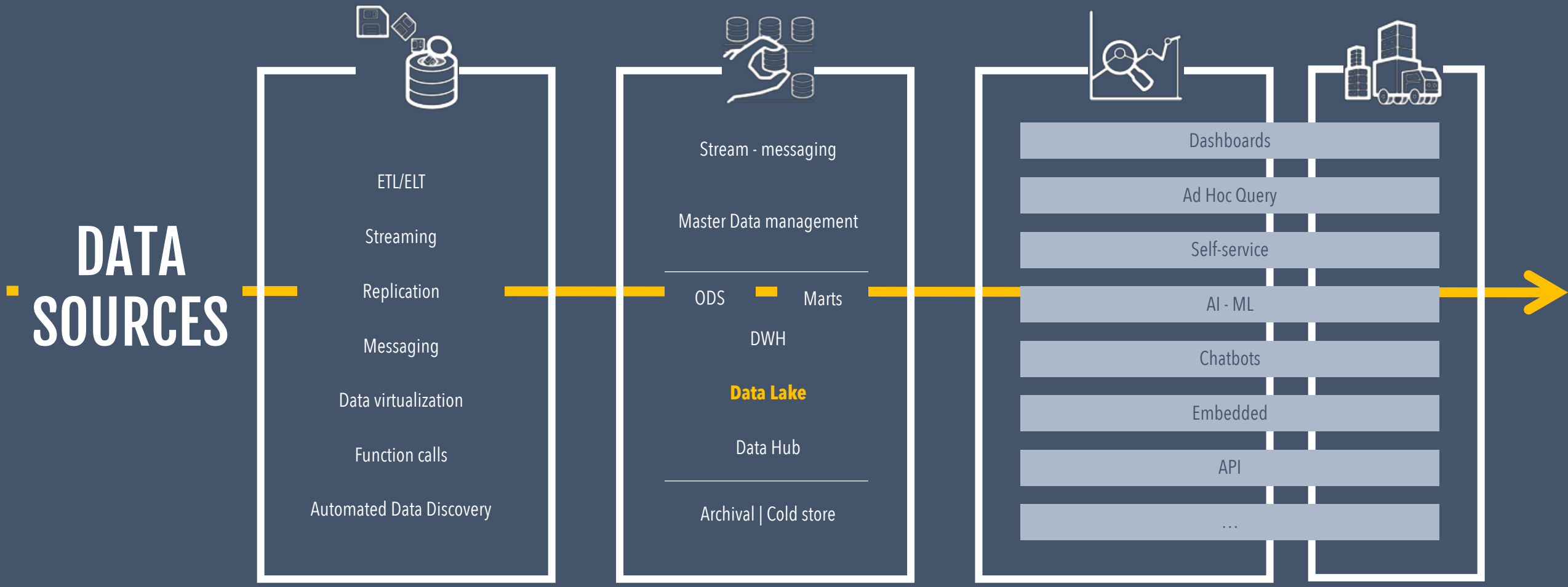
User (Non relational EDW strategy)

The Data Analytics continuum

DATA → INSIGHT → ACTION

Acquire → Organize → Analyze → Deliver

End to End Data and Analytics modern architecture basis



What is a "Data Lake"?

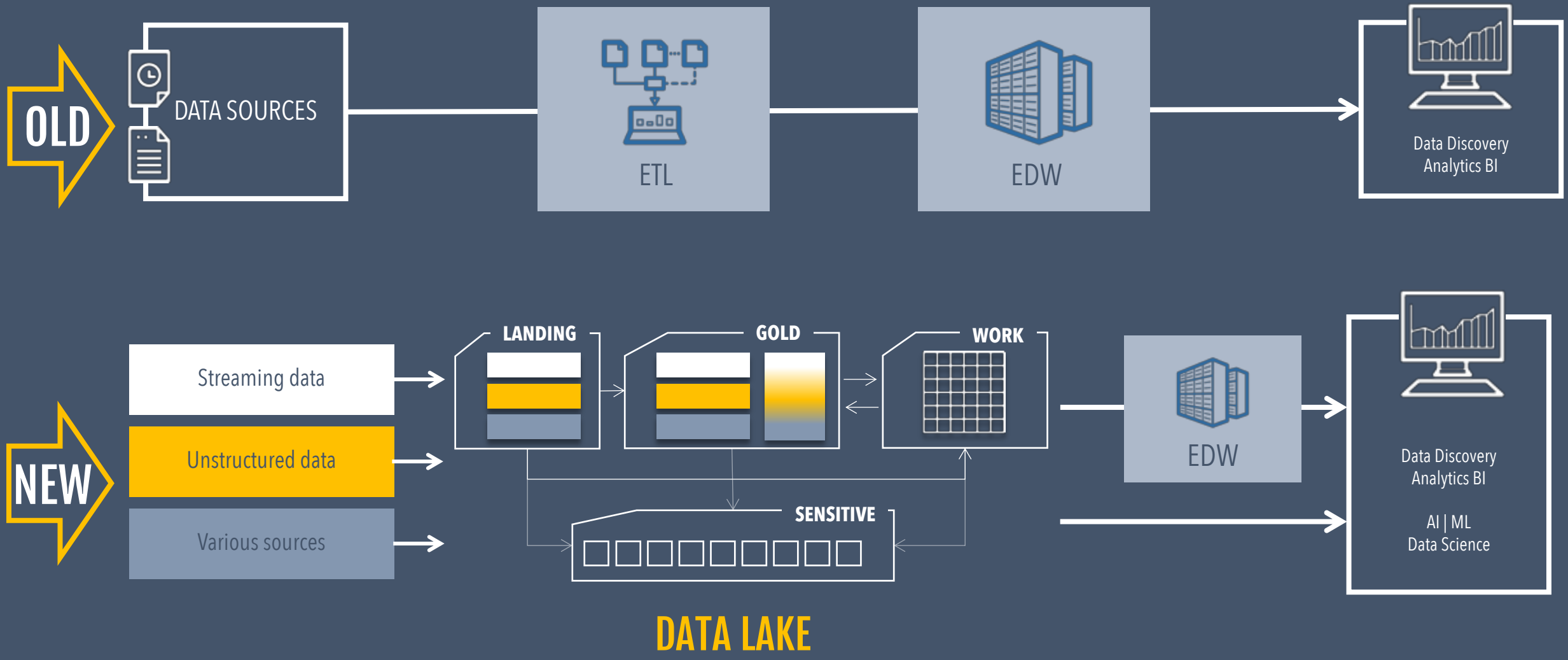
A data lake is a **centralized repository** that allows you to store all your **structured and unstructured data at any scale**.

You can store your data as-is, without having to first structure the data, and run different types of analytics–from dashboards and visualizations to big data processing, real-time analytics, and machine learning to guide better decisions

Data lake manages shared **data repositories oriented to Data analytics**. These are dedicated areas where we can work on a function or a specific operative.

So, data can be replicated in multiple repositories with different meanings and be used with different purposes

Data Lake – Tradition vs modern strategies

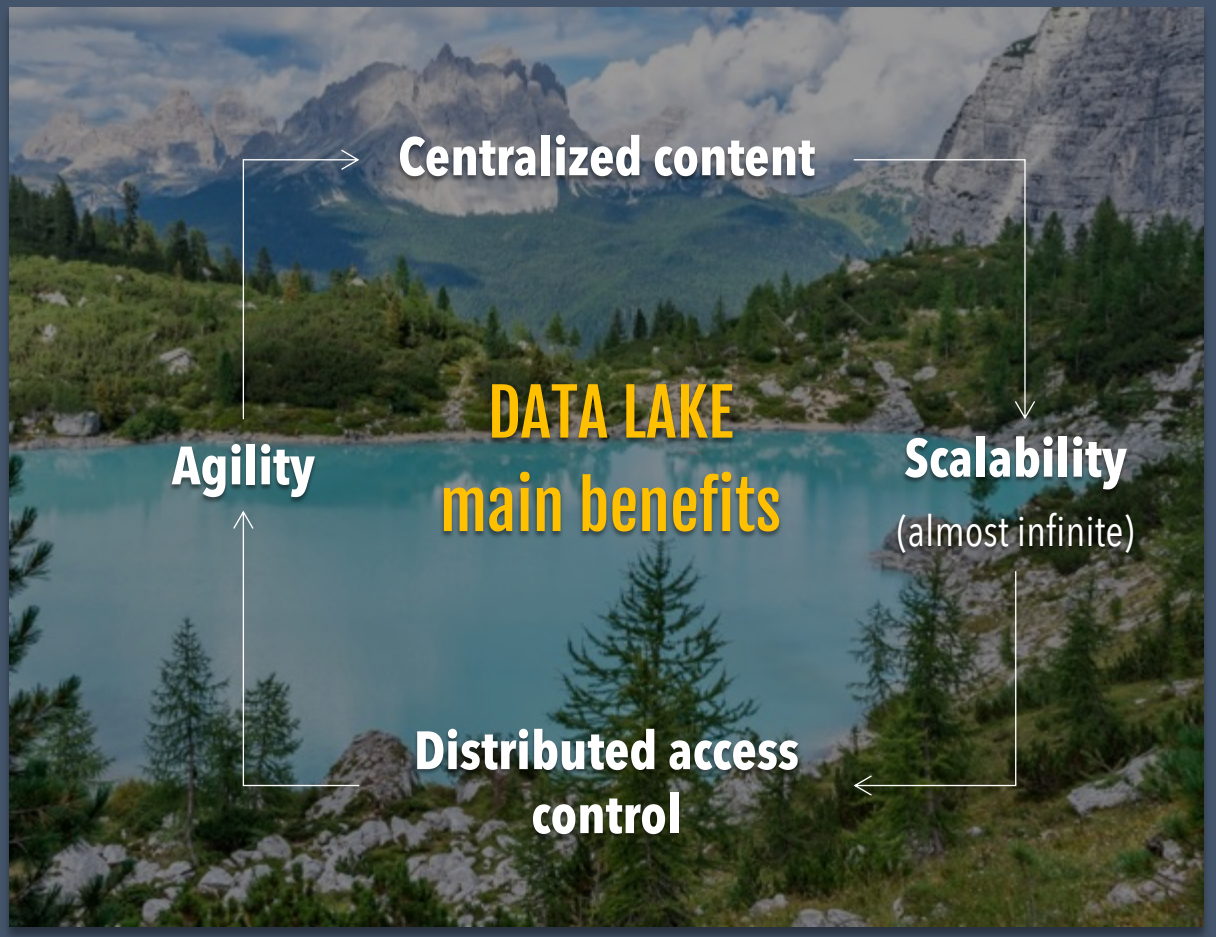


Data Lake – Main characteristics

Data lake is **absolutely linked with Data Analytics** politics and strategies into the organization

The key characteristics and challenges are:

. Security
. Organizational alignment
. Data lineage
. Data governance



Data Lake – Structure

LANDING

Also called the **raw zone, bronze zone or even the swamp**, is a place that contains the source data as is, with no transformation, such as a raw log file or a binary file coming from a mainframe.

The initial landing zone is often managed by the IT organization which automates the **data lake ingestion process**. However, project that would be business driven may also bring their raw data (external data) in the raw zone for future usage

STAGING

Also called the **silver zone, the pond, the sandbox or the exploration zone**, is the place where data can be discovered, explored and experimented with for hypothesis validation and tests.

It usually includes private zones for each individual user and a shared zone for team collaboration. It is **often seen as a sandbox** with minimal security constraints where end users can access and process the data they want with light automation.

PRODUCTION

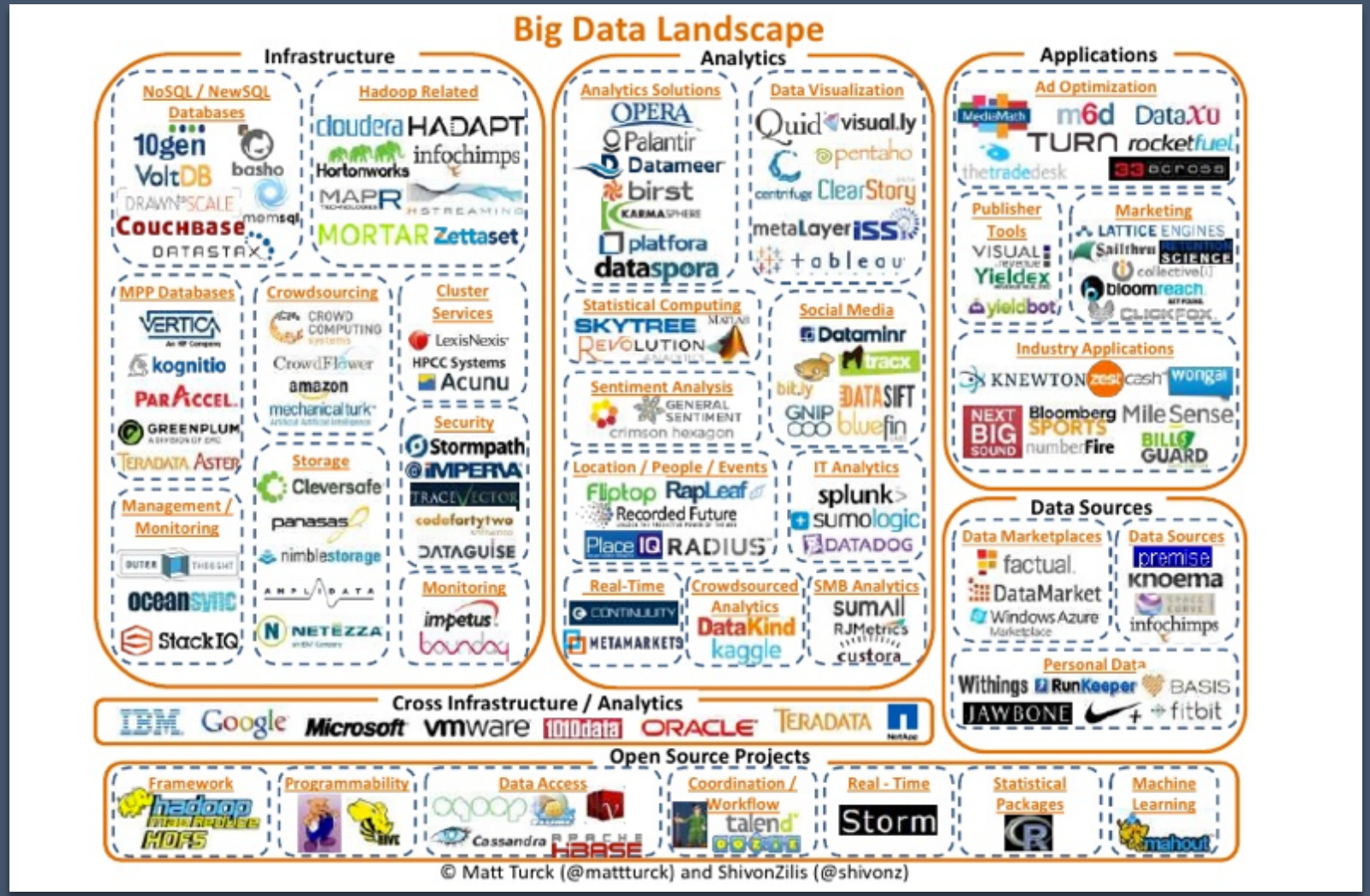
Also called the **gold zone, the refined zone, the lagoon or operationalization zone**, is where clean, well structured data is stored in the optimal format to inform critical business decisions and drive efficient operations.

It often includes an **operational data store that feeds traditional data warehouses and data marts**. This is a zone that has strict security restrictions for data access and automated provisioning of data where end users only have a read access

How it look a Big Data architecture? What's the famous Hadoop?

Big Data architectures and components have grown rapidly

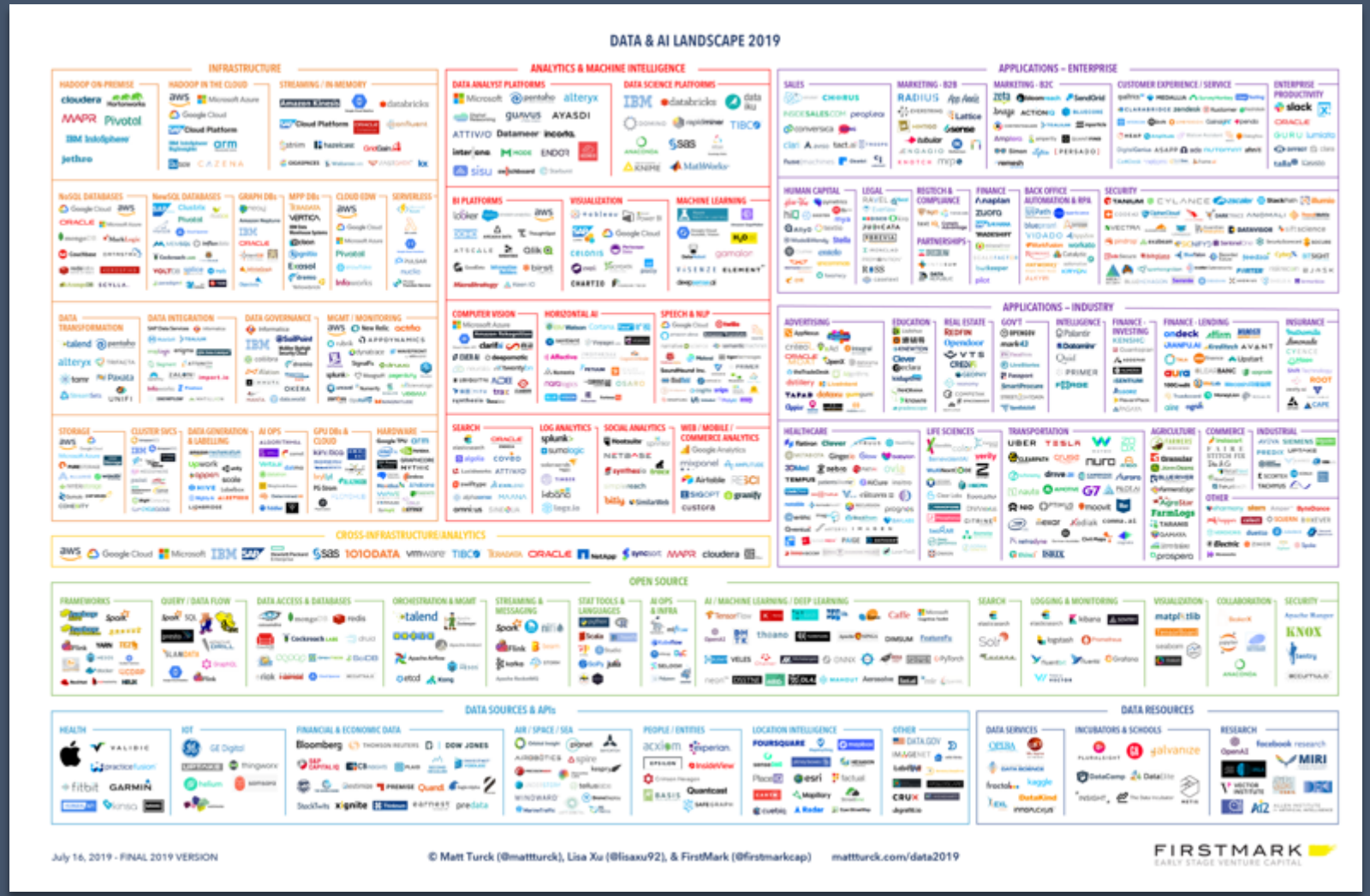
2012



Source: https://mattturck.com/a-chart-of-the-big-data-ecosystem/

Big Data architectures and components have grown rapidly

2019



Source: https://mattturck.com/data2019/

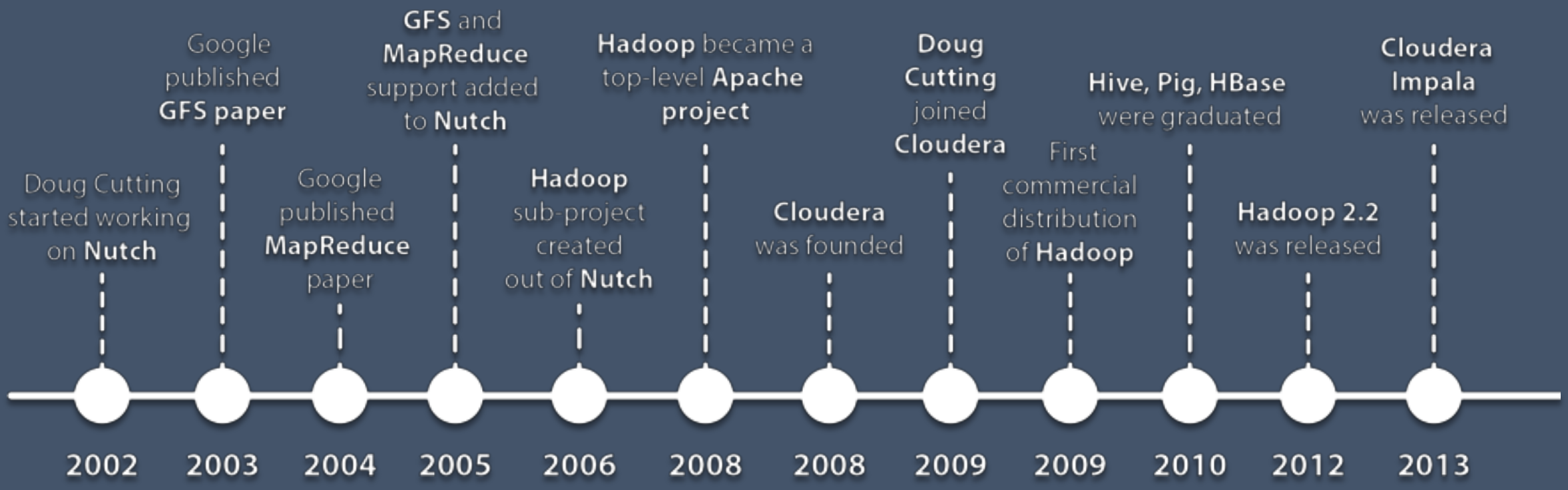
Hadoop as the start of Big Data revolution - Ecosystem

When it comes to applying big data technology into the world of BI and DWH, Hadoop is definitely a game-changer for managing enterprise data far beyond what we've ever been able to do with traditional data warehousing.

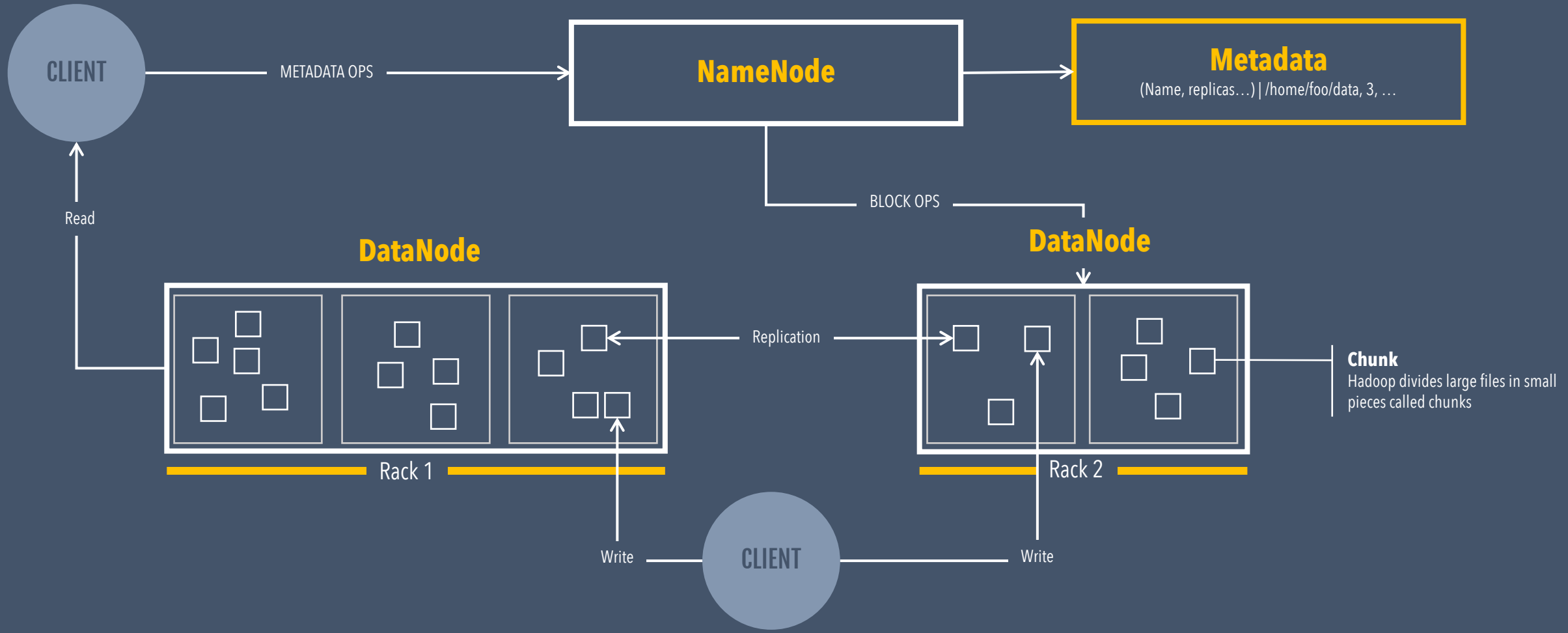
Hadoop is best thought of as an **entire Big Data ecosystem**.

We find a data storage environment within Hadoop, we also have a number of different languages and tools and APIs. As well as the vendors that bring Hadoop, which is an open source environment, to the market place, they add their own enhancements and extensions

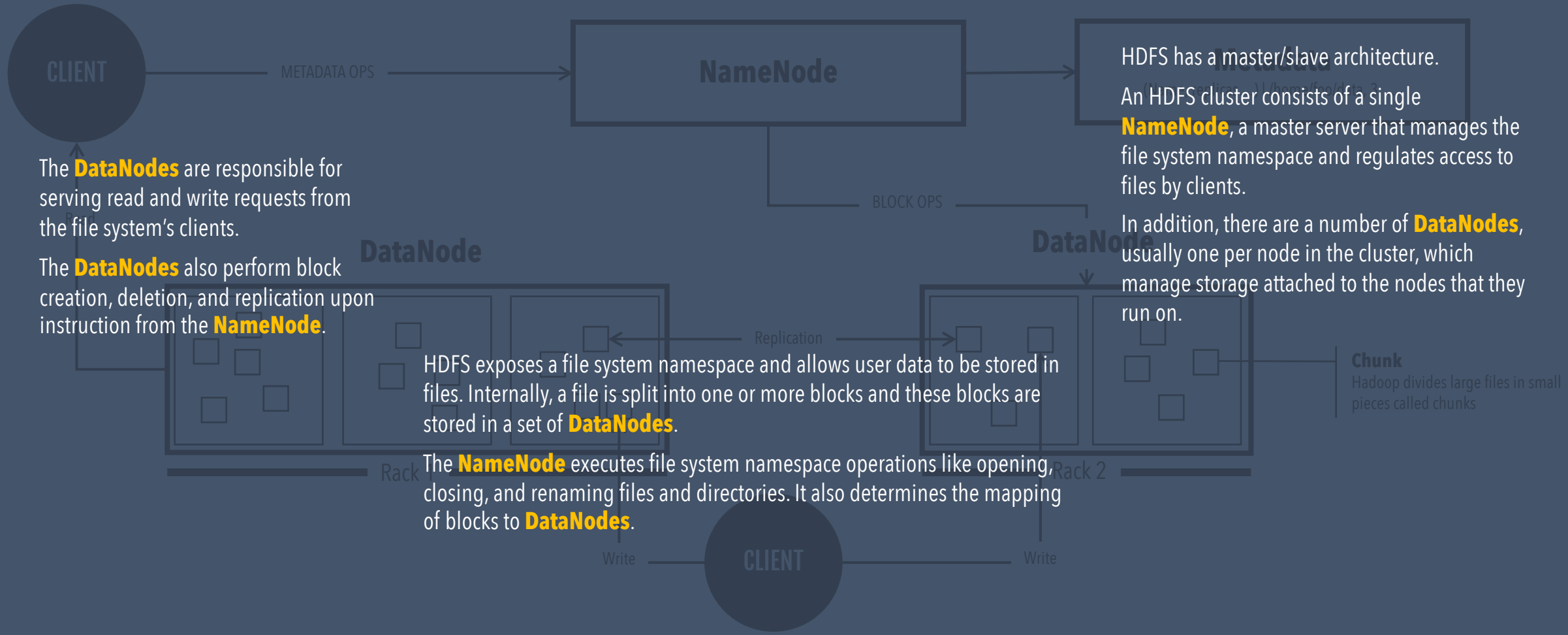
Hadoop as the start of Big Data revolution - History



HDFS – The core of Hadoop



HDFS – The core of Hadoop



The **DataNodes** are responsible for serving read and write requests from the file system's clients.

The **DataNodes** also perform block creation, deletion, and replication upon instruction from the **NameNode**.

HDFS exposes a file system namespace and allows user data to be stored in files. Internally, a file is split into one or more blocks and these blocks are stored in a set of **DataNodes**.

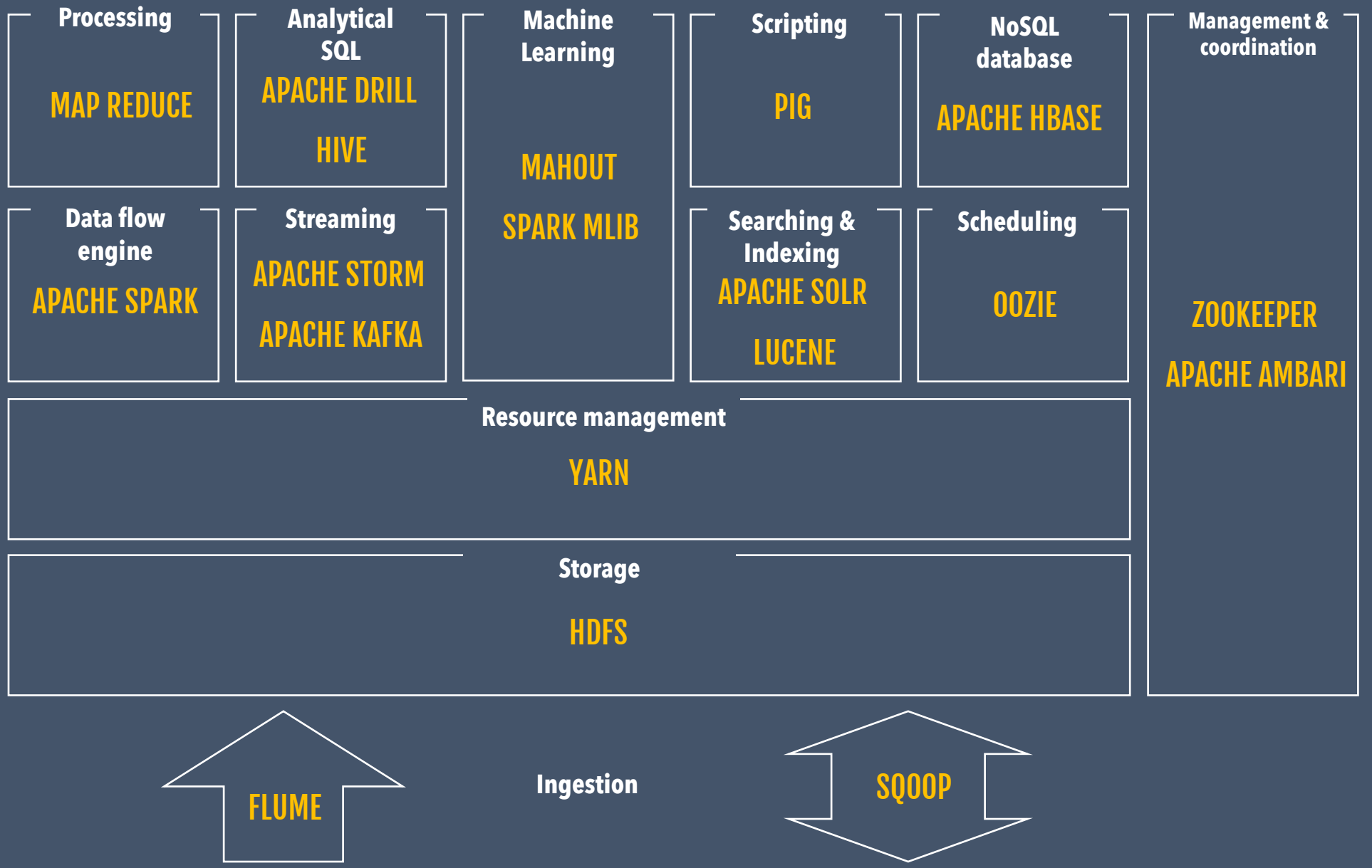
The **NameNode** executes file system namespace operations like opening, closing, and renaming files and directories. It also determines the mapping of blocks to **DataNodes**.

HDFS has a master/slave architecture.

An HDFS cluster consists of a single **NameNode**, a master server that manages the file system namespace and regulates access to files by clients.

In addition, there are a number of **DataNodes**, usually one per node in the cluster, which manage storage attached to the nodes that they run on.

The Hadoop ecosystem



The Hadoop ecosystem

Processing
MAP REDUCE

Analytical SQL
APACHE DRILL
HIVE

Machine Learning
MAHOUT
SPARK MLIB

Scripting
PIG

NoSQL database
APACHE HBASE

Management & coordination
ZOOKEEPER
APACHE AMBARI

Data flow engine
APACHE SPARK

Streaming
APACHE STORM
APACHE KAFKA

Searching & Indexing
APACHE SOLR
LUCENE

Resource management
YARN

Storage
HDFS

Ingesting data is an important part of our Hadoop Ecosystem.

The **Flume** is a service which helps in **ingesting unstructured and semi-structured data into HDFS**.

It gives us a solution which is reliable and distributed and helps us in collecting, aggregating and moving large amount of data sets.

It helps us to ingest online streaming data from various sources like network traffic, social media, email messages, log files etc. in HDFS.

Now, let us talk about another data ingesting service i.e. **Sqoop**. The major difference between Flume and Sqoop is that:

Flume only ingests unstructured data or semi-structured data into HDFS.

While Sqoop can **import as well as export structured data** from RDBMS or Enterprise data warehouses to HDFS or vice versa

Ingestion

The Hadoop ecosystem

HDFS is the one, which makes it possible to store different types of large data sets. It creates a level of abstraction over the resources, from where we can see the whole **HDFS** as a single unit.

HDFS has two core components, i.e. NameNode and DataNode.

The NameNode is the main node and it doesn't store the actual data. It contains metadata, just like a log file or you can say as a table of content.

On the other hand, all your data is stored on the DataNodes and hence it requires more storage resources. These DataNodes are commodity hardware in the distributed environment. That's the reason, why Hadoop solutions are very cost effective.

Consider **YARN** as the **brain of your Hadoop Ecosystem**. It performs all your processing activities by allocating resources and scheduling tasks.

It has two major components, i.e. ResourceManager and NodeManager.

ResourceManager is again a main node in the processing department. It receives the processing requests, and then passes the parts of requests to corresponding NodeManagers accordingly, where the actual processing takes place.

NodeManagers are installed on every DataNode. It is responsible for execution of task on every single DataNode

The Hadoop ecosystem

Apache **Zookeeper** is the **coordinator of any Hadoop job** which includes a combination of various services in a Hadoop Ecosystem.

Apache Zookeeper coordinates with various services in a distributed environment.

Before Zookeeper, it was very difficult and time consuming to coordinate between different services in Hadoop Ecosystem.

The Apache **Ambari** project is aimed at making Hadoop management simpler by developing software for **provisioning, managing, and monitoring Apache Hadoop clusters**. Ambari provides an intuitive, easy-to-use Hadoop management web UI backed by its RESTful APIs.

The Hadoop ecosystem

Other interesting NoSQL databases are:

. **CASSANDRA** | Key value based | Very fast and used nowadays

. **MONGODB** | Document based | Oriented to docs and JSON | Not very scalable but used today

. **REDIS** | Key value based | Optimized for data performance and speed | Normally used for cache

. **ELASTIC DB** | Documental storage, indexation and search solution | It comes with a powerful data visualization call Kibana

. **NEOJ4** | Graph based | One the best DB oriented to graphs

. **COUCHDB** | Oriented to cache and large storage | Optimized for DB that lose connectivity often

HBase is an open source NoSQL database that supports all types of data and that is why, it's capable of handling anything and everything inside a Hadoop ecosystem.

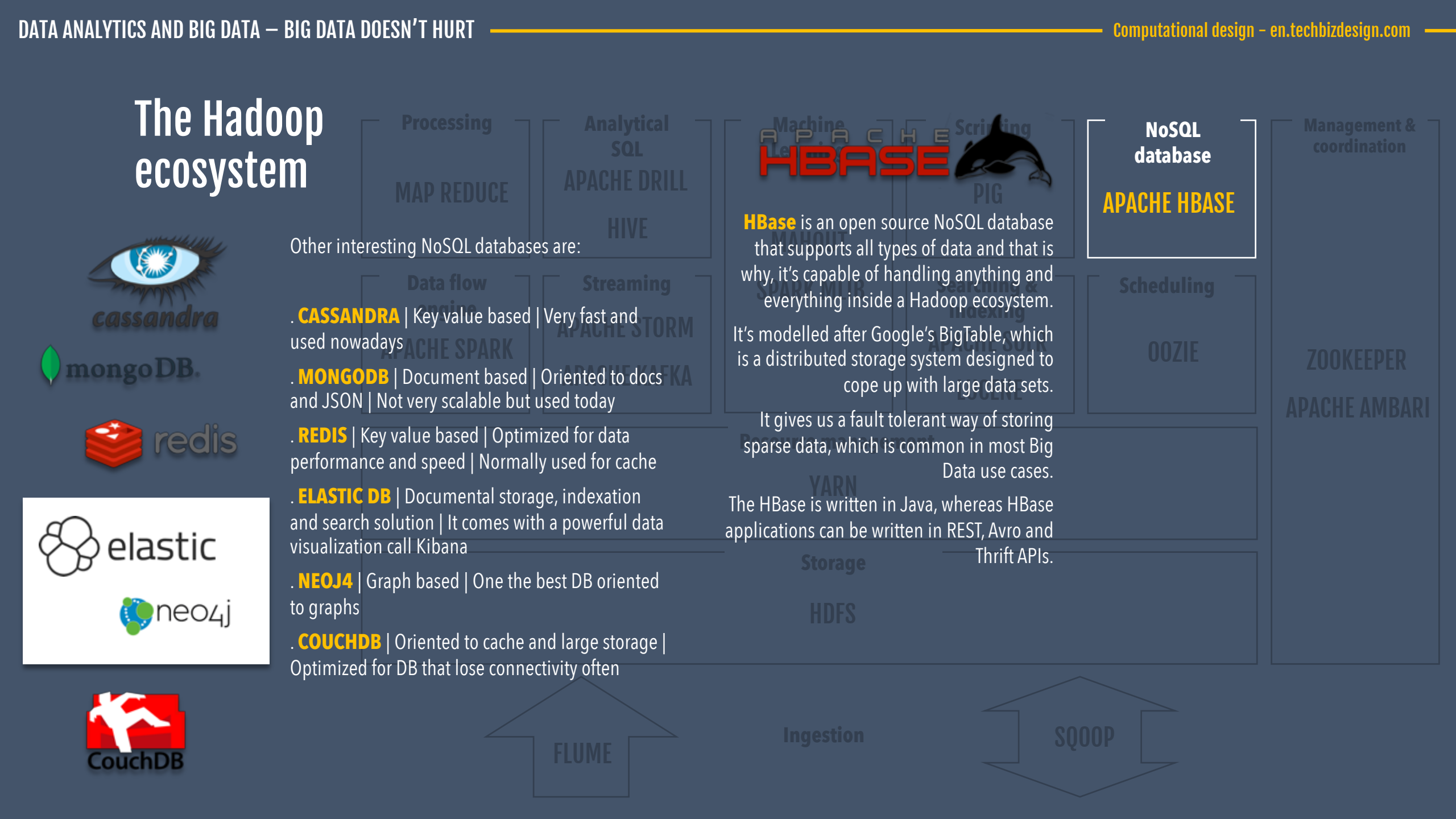
It's modelled after Google's BigTable, which is a distributed storage system designed to cope up with large data sets.

It gives us a fault tolerant way of storing sparse data, which is common in most Big Data use cases.

The HBase is written in Java, whereas HBase applications can be written in REST, Avro and Thrift APIs.

NoSQL database

APACHE HBASE



The Hadoop ecosystem

Apache **Pig** is a platform for **analyzing large data sets** that consists of a high-level language for expressing data analysis programs, coupled with infrastructure for evaluating these programs.

The salient property of Pig programs is that their structure is amenable to substantial parallelization, which in turns enables them to handle very large data sets.

At the present time, Pig's infrastructure layer consists of a compiler that produces sequences of Map-Reduce programs, for which large-scale parallel implementations already exist (e.g., the Hadoop subproject)

Consider Apache **Oozie** as a **clock and alarm service** inside Hadoop Ecosystem. For Apache jobs, Oozie has been just like a scheduler.

It schedules Hadoop jobs and binds them together as one logical work.

The Hadoop ecosystem

Processing

Analytical SQL

APACHE DRILL

Machine Learning

MAHOUT

SPARK MLIB

Scripting

PIG

NoSQL database

Management & coordination

Data flow

Streaming

Searching & Indexing

APACHE SOLR

LUCENE

Resource management

YARN

Storage

HDFS

Ingestion

FLUME

SQOOP

Mahout provides an environment for **creating machine learning applications** which are scalable

It performs collaborative filtering, clustering and classification. Some people also consider frequent item set missing as Mahout's function

Spark MLIB is part of Apache Spark

Apache **Solr** and Apache **Lucene** are the two services which are used for searching and indexing in Hadoop Ecosystem.

Apache Lucene is based on Java, which also helps in spell checking.

If Apache Lucene is the **engine**, Apache Solr is the **car built** around it. Solr is a complete application built around Lucene

The Hadoop ecosystem

Processing
MAP REDUCE

Analytical SQL
APACHE DRILL
HIVE

Scripting
PIG

NoSQL database
APACHE HBASE

Management & coordination
ZOOKEEPER
APACHE AMBARI

Data flow engine
APACHE SPARK

Streaming
APACHE STORM
APACHE KAFKA

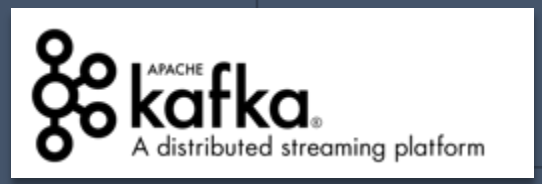
Searching & Indexing
APACHE SOLR
LUCENE

Scheduling
OOZIE

Resource management
YARN

FLUME

Apache **Storm** is a free and open source distributed realtime computation system. Apache Storm makes it easy to **reliably process unbounded streams of data**, doing for realtime processing what Hadoop did for batch processing.



Apache **Kafka** is an open-source stream-processing software platform developed by LinkedIn and donated to the Apache Software Foundation, written in Scala and Java. The project aims to provide a unified, high-throughput, low-latency platform for handling **real-time data feeds**. Kafka can connect to external systems (for data import/export) via Kafka Connect and provides Kafka Streams, a Java stream processing library.

Facebook created **HIVE** for people who are fluent with SQL. Thus, HIVE makes them feel at home while working in a Hadoop Ecosystem.

Basically, **HIVE** is a **data warehousing component which performs reading, writing and managing large data sets** in a distributed environment using **SQL-like interface**.

HIVE + SQL = HQL

The query language of Hive is called Hive Query Language(HQL), which is very similar like SQL.

Apache **Drill** is an open-source software framework that supports **data-intensive distributed applications for interactive analysis** of large-scale datasets.

Drill is the open source version of Google's Dremel system which is available as an infrastructure service called Google BigQuery. One explicitly stated design goal is that Drill is able to scale to 10,000 servers or more and to be able to process petabytes of data and trillions of records in seconds.

The Hadoop ecosystem

Processing

MAP REDUCE

Data flow engine

APACHE SPARK

It is the **core component of processing in a Hadoop Ecosystem** as it provides the logic of processing. In other words, MapReduce is a software framework which helps in writing applications that processes large data sets using distributed and parallel algorithms inside Hadoop environment.

In a MapReduce program, Map() and Reduce() are two functions: The **Map** function performs actions like filtering, grouping and sorting. While **Reduce** function aggregates and summarizes the result produced by map function.

The result generated by the Map function is a key value pair (K, V) which acts as the input for Reduce function

Apache **Spark** is a **framework for real time data analytics in a distributed computing environment**, written in Scala and originally developed at the Berkeley University

It executes **in-memory computations** to increase speed of data processing over Map-Reduce. It' s 100x faster than Hadoop for large scale data processing by exploiting in-memory computations and other optimizations.

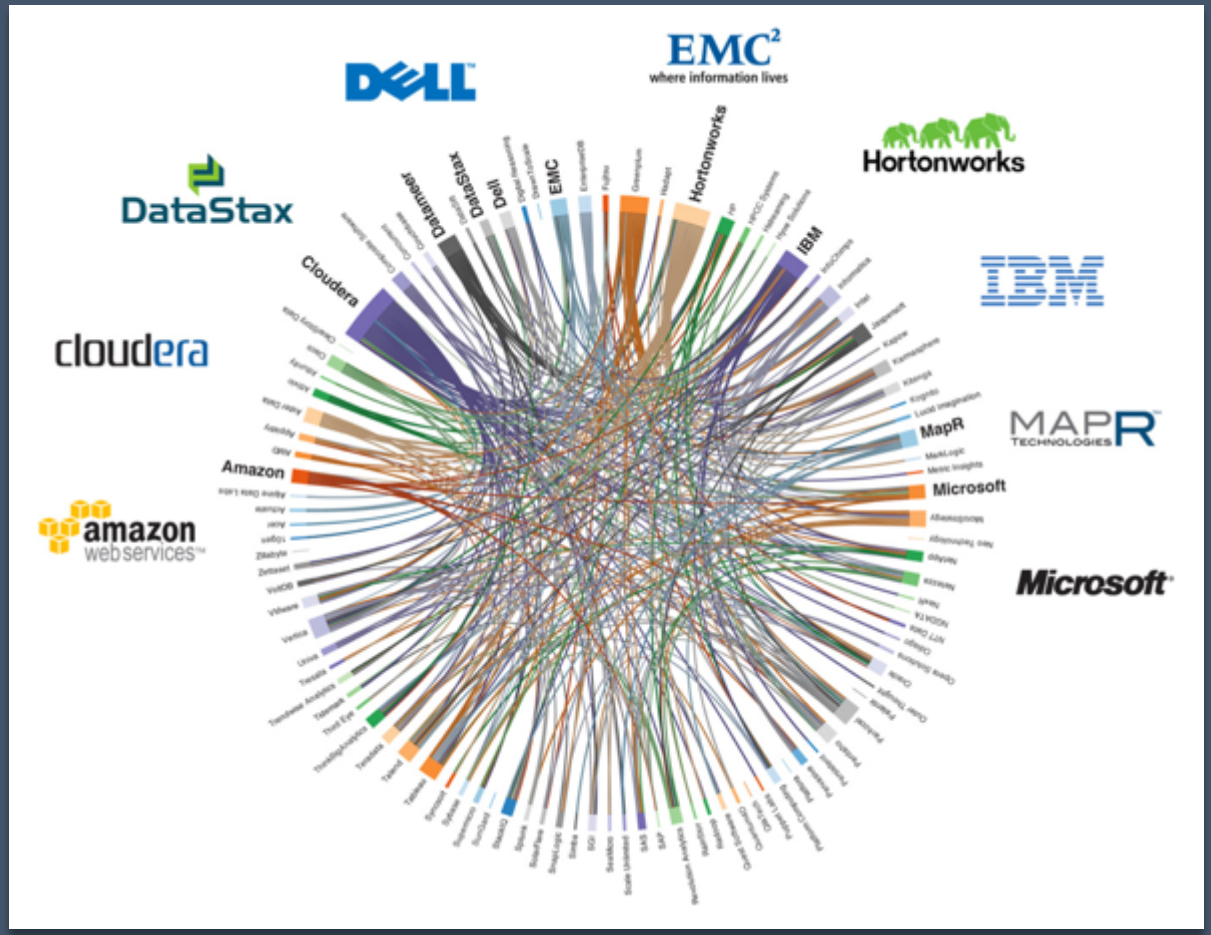
Spark comes packed with **high-level libraries** (MLlib, GraphX, SQL + Data Frames, Streaming services…), including support for R, SQL, Python, Scala, Java…

Hadoop distributions

To meet the needs of enterprises that deploy Hadoop, and help them with their big data requirements, vendors have developed commercial distributions of Hadoop and related open source technologies.

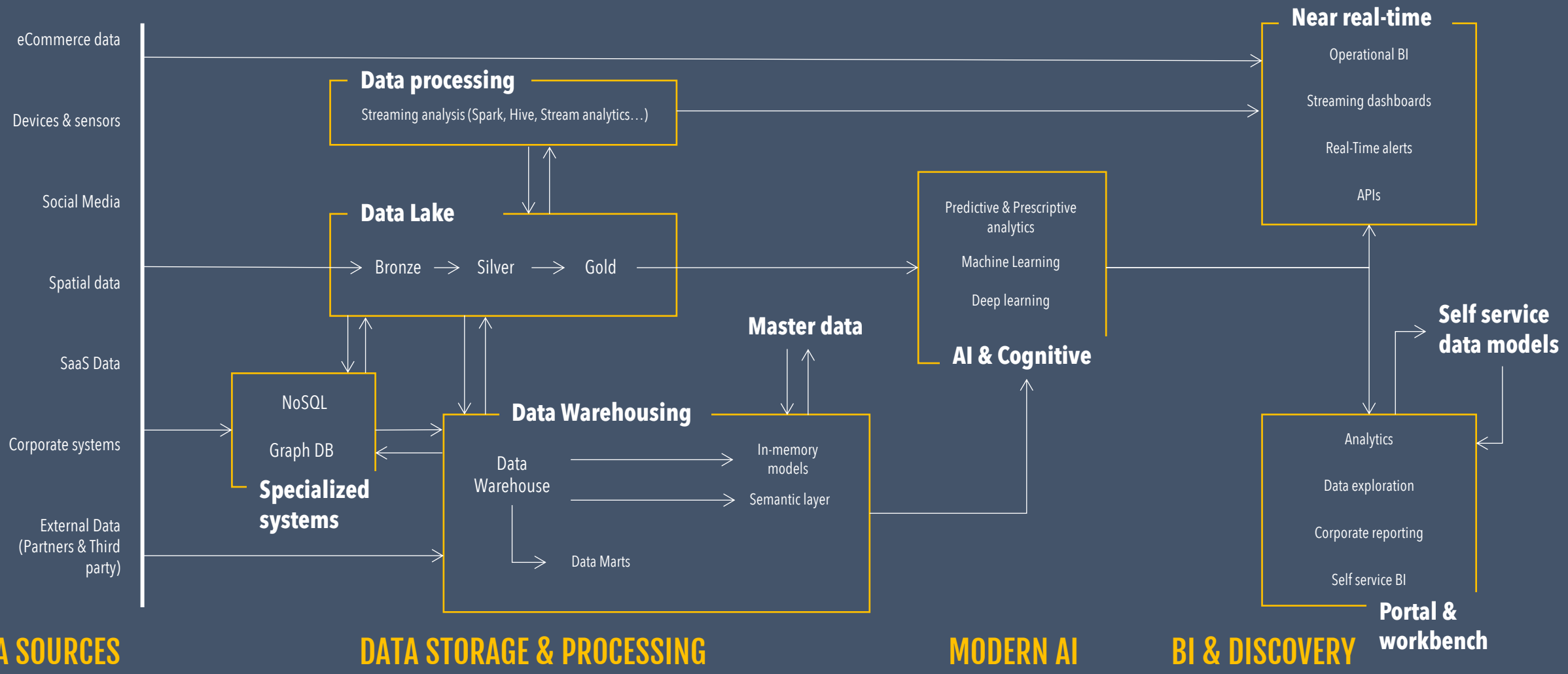
In May 2019, the top ones are:

- ALIBABA CLOUD E-MAPREDUCE
- AMAZON EMR
- AZURE HDINSIGHT
- CLOUDERA CDH
- GOOGLE CLOUD DATAPROC
- HORTONWORKS DATA PLATFORM
- MAPR
- QUBOLE



What does it looks and end-to-end Data Analytics architecture? What is the role of Big Data?

An integrated Big Data architecture example



The Big Data architecture most critical decisions

The SQL engines for Big Data workloads adoption

Enterprises are moving to create data lakes and logical data warehouses, with Apache Hadoop, Amazon S3, Azure Data Lake Storage, Google Cloud Storage and NoSQL data stores both on- premises and in the cloud.

These data hubs act as a central repository for data from various sources, for different workloads and different data formats. Big data tools and frameworks are then used to manage and run analytics to build data-driven products and gain actionable insights from this data.

The rapid adoption of the next-generation data storage and data processing technologies has triggered development of a plethora of SQL engines on big data technologies. This enables existing tools and systems – such as business intelligence (BI), data warehouses, ETL (extraction, transformation and loading) and data marts – to work seamlessly with the next-generation data stores.

The optimal Data storage selection

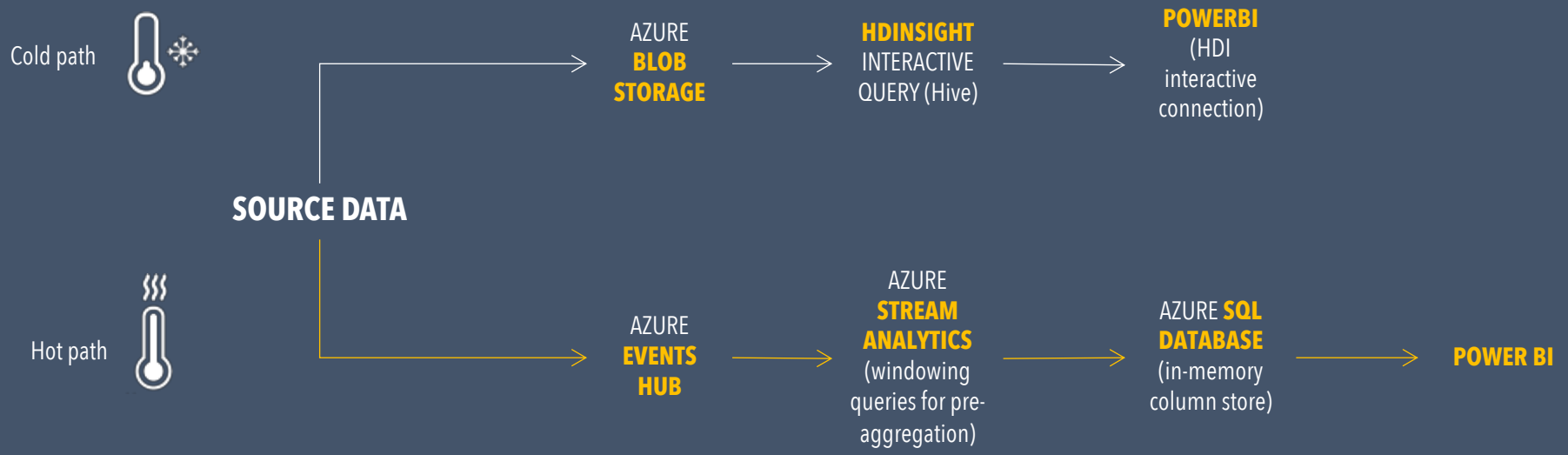
Modern data architectures are built to address scalability. They need to handle very high numbers of concurrent reads or writes, with ultralow latency (millisecond or even microsecond) and high throughput. At the same time, they must still accommodate exploratory and historical workloads. In modern data architectures, users may originate from across the world.

Hence, zero to very little downtime is permitted. Data stores need to accommodate multistructured data types that are ingested or egressed through various query languages and APIs. In addition, the volume of data is in the terabyte, petabyte or even exabyte range. Deployment models for modern data architectures include microservices architectures (MSAs), often orchestrated by containers.

So, your data storage election in order to fulfill your business needs can be the key for your project success

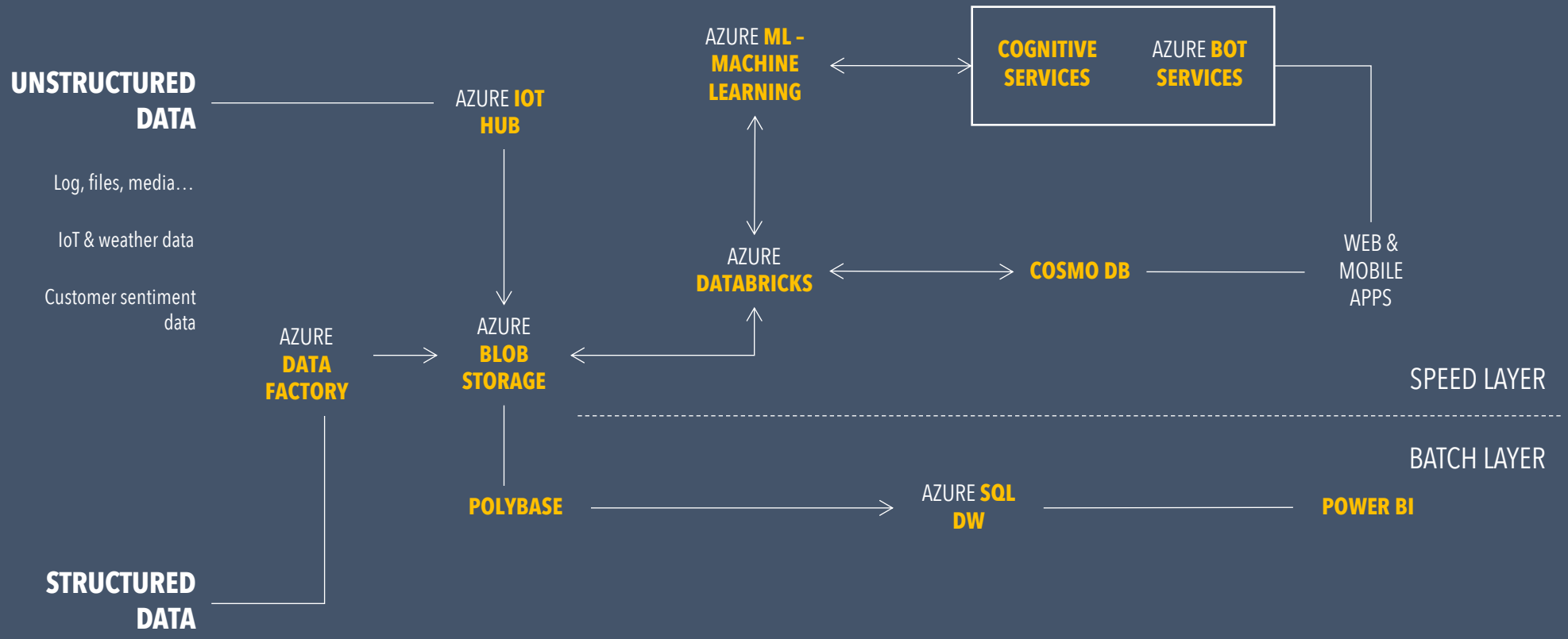
Top tech giants Big Data architecture approach – MICROSOFT AZURE

The easy way…

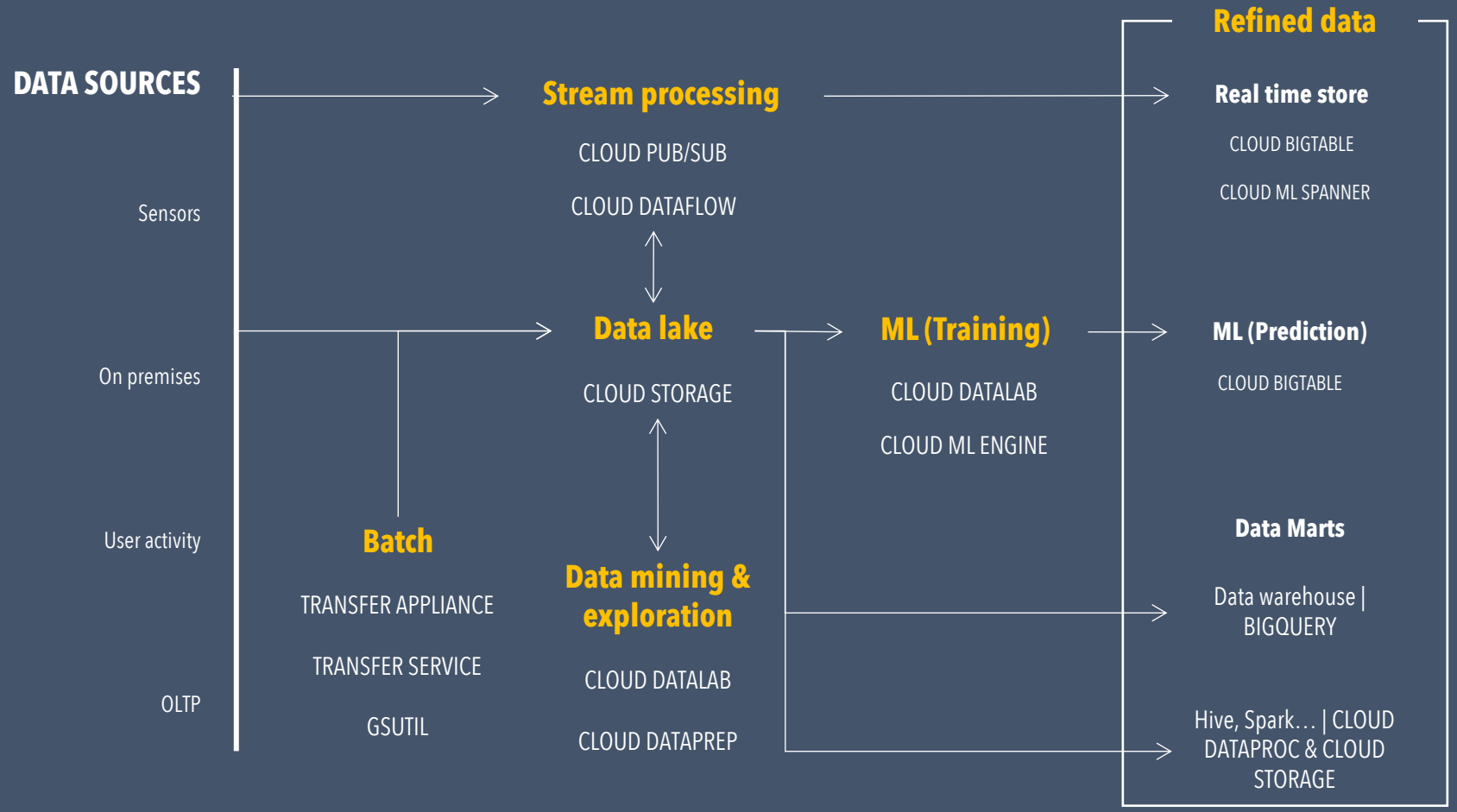


Top tech giants Big Data architecture approach – MICROSOFT AZURE

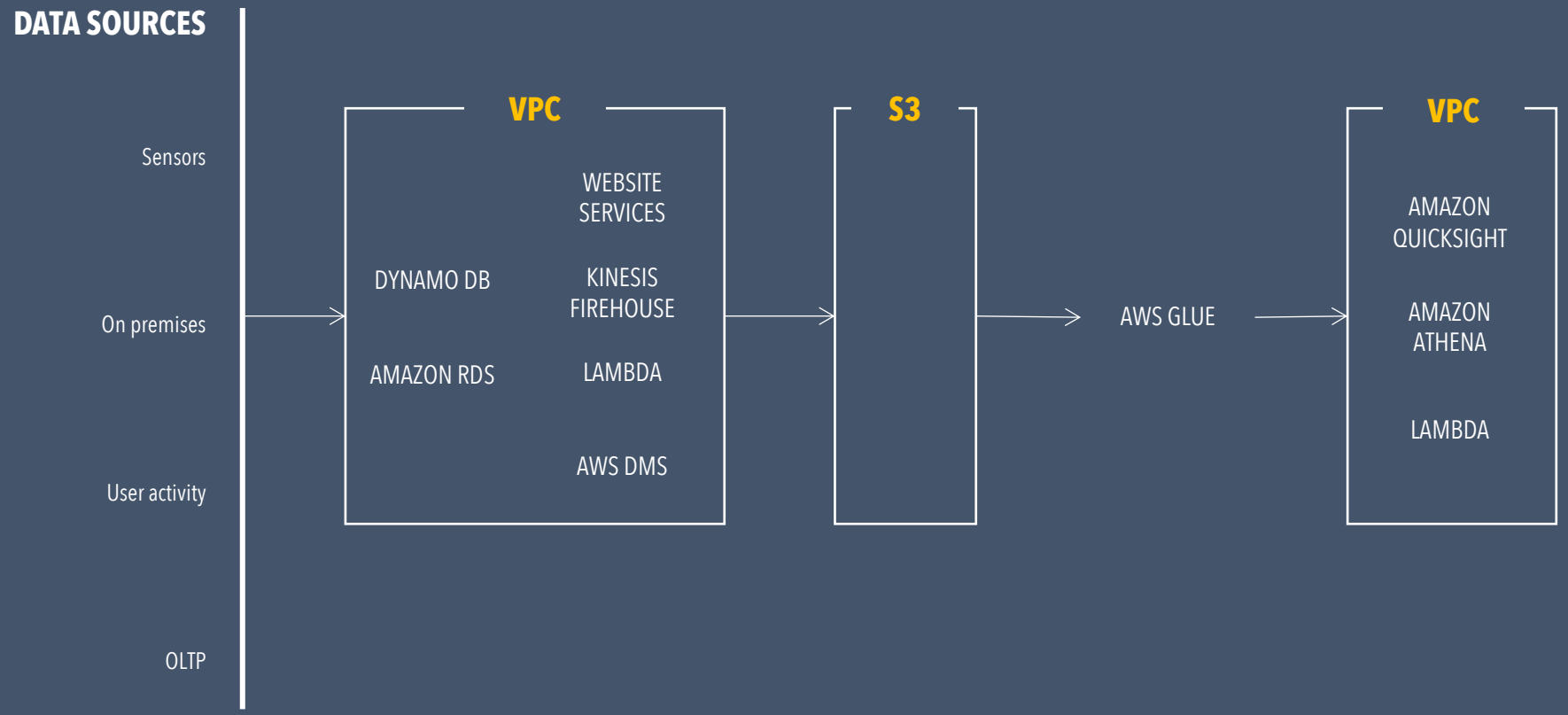
The complete way…



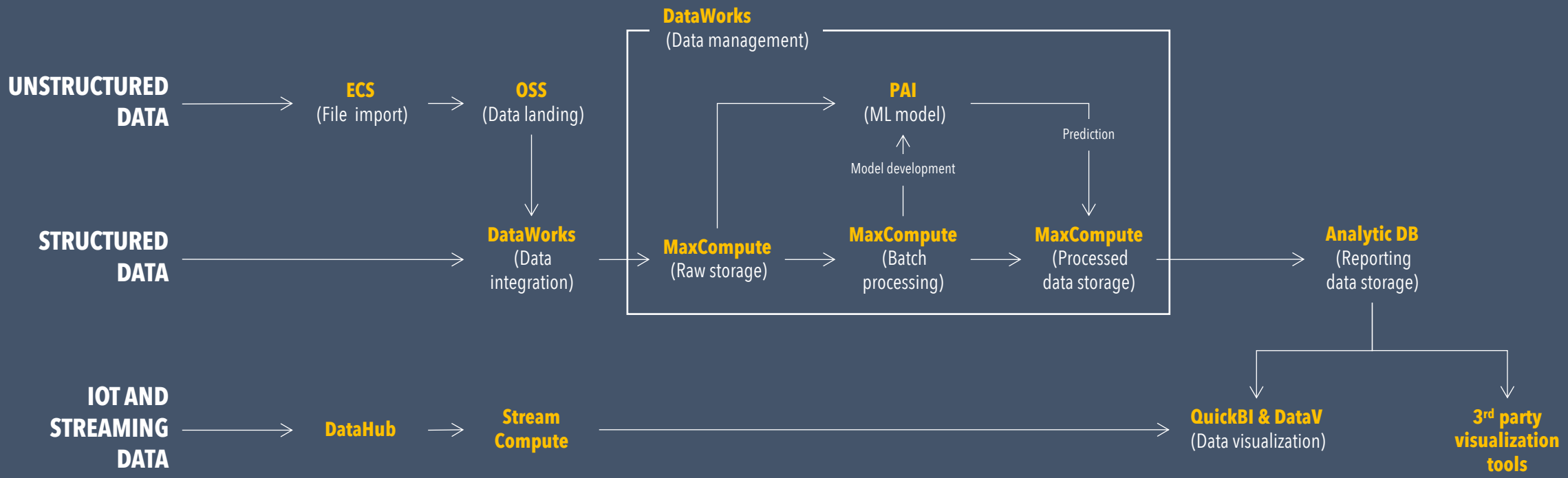
Top tech giants Big Data architecture approach – GOOGLE CLOUD PLATFORM



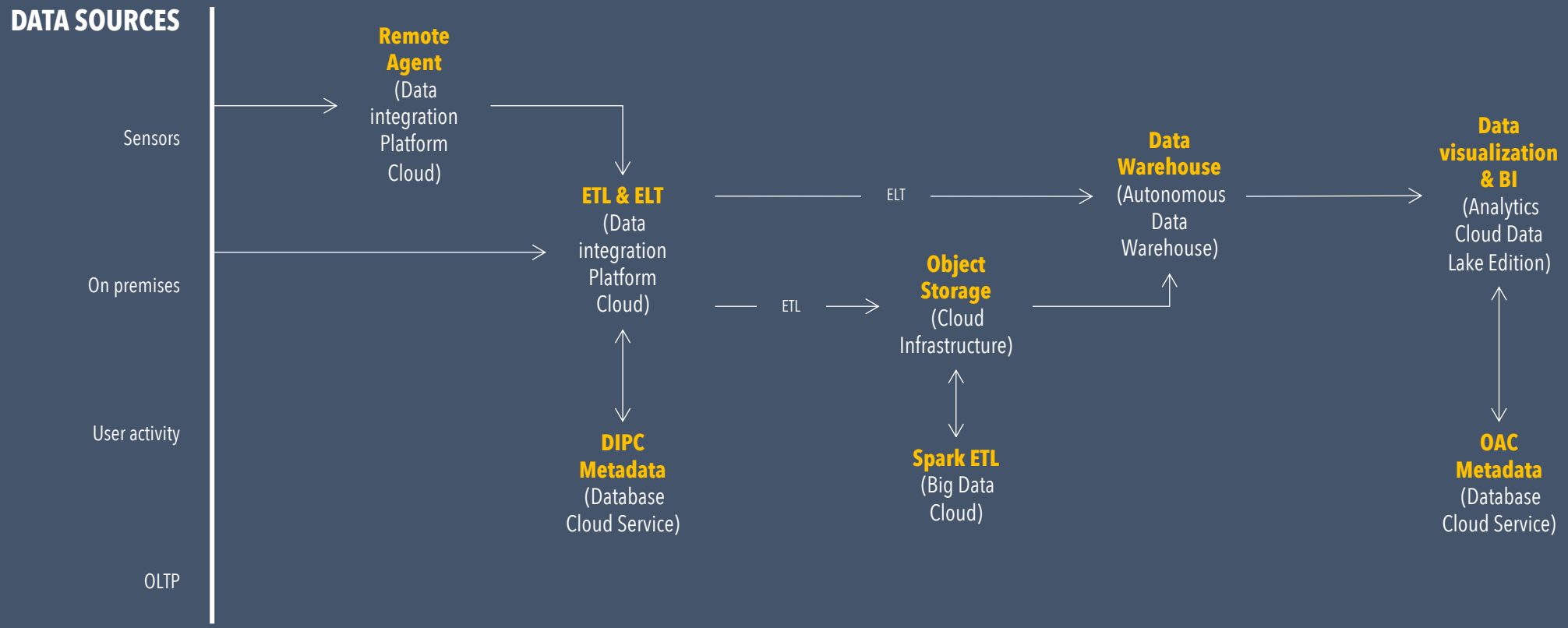
Top tech giants Big Data architecture approach – AMAZON WS



Top tech giants Big Data architecture approach – ALIBABA CLOUD



Top tech giants Big Data architecture approach – ORACLE



Any recommendation to implement a Big Data strategy in my organization? Anything to be aware with?

Be careful with the Big Data myths

80% of all data is NOT unstructured

All data has structure, although its structure may not be obvious or familiar. There wouldn't be any patterns to discover in data if it didn't have structure.

Be careful with the non Data Analytics experts that often believe **nonrelational data is unstructured because it doesn't fit neatly into the familiar structure of relational databases**

Advanced analytics is NOT an improved version of traditional analytics

Traditional analytics such as BI or corporate reporting is oriented to descriptive analytics, while advanced analytics solves problems using predictive analytics and prescriptive analytics.

Predictive analytics predicts future outcomes and behavior, such as a customer's shopping behavior or a machine's failure.

Prescriptive analytics goes further, suggesting actions to take based on the predictions.

So, **it's not a matter of evolution, they have just a different purpose**.

You need MORE than embedded analytics

Embedded analytics typically cover only the business process or function run by a transaction system. They provide additional reports and metrics that may be of use to only one business unit, but they lack predictive modeling capabilities.

There's an opportunity for organizations to **combine embedded analytics with other tools to consolidate and analyze data** to assess enterprise-wide operations and performance.

Improved analytics tools WON'T replace Data Scientists

Many vendors of advanced analytics solutions claim that their data science functions are easy enough for anyone to use, with no need for coding, knowledge of predictive algorithms or years of training.

Reality is there's a shortage of data scientists. IBM predicts that, by 2020, 2.7 million jobs globally will be needed to cope with big data, plus to current demand of 700.000 openings.

In despite of vendors sales oriented messages, **data scientist role will be among the roles in greater demand for may years**.

Great data scientist DON'T need a Ph.D. to do their job

They just need a well-rounded competency in statistics and optimization/ operations research. We prefer they also understand business processes and have a strong drive to understand the real world through data and are imaginative in creating surrogates for the real world.

We love they are curious, and have the ability to ask great questions and obtain answers to those questions from data.

Ph.D. is always valuable but not enough to be a good data scientist.

Be careful with the Big Data myths

Descriptive Analytics DOESN'T look to the past and Predictive to the future

All analytics is based on the past and most of it looks to the future. So, the **results of all forms of analytics are created by analyzing data that has been collected in the past**. When you use analytics, you're assuming that the future will "behave" in a similar way to the past.

Sometimes this assumption is wrong. When this happens, the past is no longer a good indicator of the future. This means that the results of analytics are no longer reliable and may lead to very poor – and even harmful – decision making.

Fast analytics ISN'T equal Real-Time analytics

Hadoop, no-SQL DBMSs, in-memory databases, in-memory data grids and other big data technologies enable you to run queries and analytic models much more quickly than traditional technologies. Some people call this real-time analytics, even if all the data is weeks old.

Analytics is not real-time – and not even near-real-time – unless some or all of the input data has been captured in the past few seconds or minutes.

We CANNOT predict almost anything with Big Data

We will, indeed, be able to predict many more things with the availability of more data and more data sources.

But **there will still be many things we can't predict, especially in complex domains** such as law and politics, and with natural phenomena, such as earthquakes.

Prediction is difficult even in homogeneous, fairly well-structured domains with endless streams of data like, for example, online marketing.

Unfortunately, Big Data IS biased

Data is always biased, regardless of its volume. Data is the result of certain measurements and was collected with a certain purpose. So, **approach biased data with great care**.

Social media, for example, provides analytics professionals with a vast dataset they can use for a variety of analyses, such as sentiment, trends, threats and key influences. However, the data produced by social media is itself a biased sample, as those who use social networking sites tend to be younger

Coming soon

DATA ANALYTICS AND BIG DATA – THE DATA SCIENCE CONNECTION