

# Guidelines for Designing New Data Architectures



**Rick F. van der Lans**  
Industry analyst



CONSULTANCY

Copyright © 2026 R20/Consultancy B.V.,  
The Netherlands. All rights reserved. No  
part of this material may be reproduced,  
stored in a retrieval system, or transmitted  
in any form or by any means, electronic,  
mechanical, photographic, or otherwise,  
without the explicit written permission of  
the copyright owners.

# Rick F. van der Lans

---



**Rick F. van der Lans** is a highly-respected independent analyst, consultant, author, and internationally acclaimed lecturer specializing in data warehousing, business intelligence, big data, and database technology. He is managing director of R20/Consultancy BV.

He has presented countless seminars, webinars, and keynotes at industry-leading conferences. Rick helps clients worldwide to design their data warehouse, big data, and business intelligence architectures and solutions and assists them with selecting the right products. He has been influential in introducing the new logical data warehouse architecture worldwide which helps organizations to develop more agile business intelligence systems.

He is the author of several books on computing, including his new *Data Virtualization: Selected Writings* and *Data Virtualization for Business Intelligence Systems*. Some of these books are available in different languages. Books such as the popular *Introduction to SQL* is available in English, Dutch, Italian, Chinese, and German and is sold world wide. He also authored numerous whitepapers for vendors.

In 2018 he was selected the sixth most influential BI analyst worldwide by [analytica.com](http://analytica.com).

You can get in touch with Rick van der Lans via:

Email: [rick@r20.nl](mailto:rick@r20.nl)

Website: [www.r20.nl](http://www.r20.nl)

LinkedIn: <http://www.linkedin.com/pub/rick-van-der-lans>

# Agenda and Subjects

---



- 1. Introduction: Why a New Data Architecture?**
- 2. Terminology and Concepts**
- 3. Introduction to Data Architectures**
- 4. New Technologies for Data Storage, Processing, and Analytics**
- 5. Steps 1-3: Setting the Stage**
- 6. Step 4: Architectural Design Principles**
- 7. Step 5: Reference Data Architectures**
- 8. Step 6-7: Designing New Data Architectures**
- 9. Steps 8-9: Final Steps**
- 10. Closing Remarks**

A photograph of a desert landscape featuring large, rounded, light-colored boulders. The ground is sandy and covered with sparse, low-lying green and brown vegetation. The sky is a clear, bright blue. A semi-transparent white box with rounded corners is overlaid on the upper portion of the image, containing the title text.

# **Part 1: Introduction: Why a New Data Architecture?**



# Unlocking success in digital transformations

October 2018 | Survey



Digital transformations are even more difficult than traditional change efforts to pull off. But the results from the most effective transformations point to five factors for success.



# 4 Steps to Help You Become a Data Driven Company in 2019



CLEMENT RENÉ

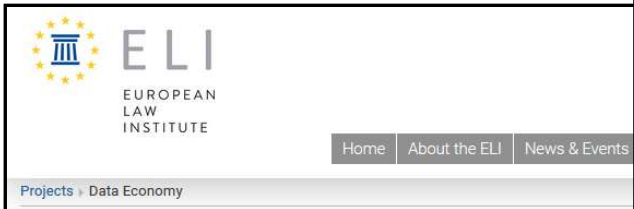
Inc.

INC. 5000 CONFERENCE

TECHNOLOGY

# 3 Ways Your Company Should Be Like Google, and 1 Way it Definitely Shouldn't

You don't have to be the world's most popular search engine in order to innovate. Here are four things you can learn from



## Principles for a Data Economy (with the ALI)

With the rise of an economy in which data is a tradeable asset globally, more certainty is needed with regard to the legal rules that are applicable to the transactions in which data is an asset. Critical questions arise such as who has which right with regard to the data generated by connected devices? They need to be answered urgently, as lack of clarity in this field potentially hinders innovation and growth and, more importantly, troubles consumers, data-driven industries, and start-ups.

For an overview of past and upcoming meetings of this project, please click here.

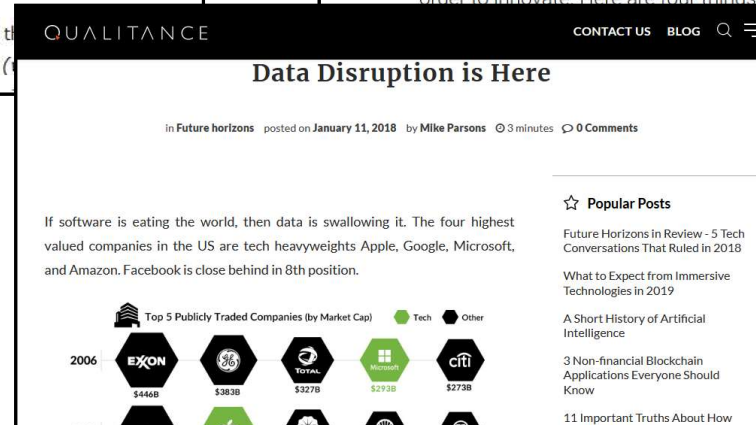
ELI Members, who are interested in actively contributing to the development of this project are invited to

## Data is a business asset beyond imagination – here is why (and where)

It has almost become embarrassing to say that data is a business asset and should be treated as one (*the same goes for information*).

The 'data is an asset' or a 'data is a business asset' message is not new. It goes back over two decades. However, despite the fact that so many people have said it so often before, we still see that there is a difference between preach and practice.

It's not that organizations fail to understand the value of data, information and actionable intelligence (



QUALITANCE

CONTACT US BLOG

## Data Disruption is Here

in Future horizons posted on January 11, 2018 by Mike Parsons 3 minutes 0 Comments

If software is eating the world, then data is swallowing it. The four highest valued companies in the US are tech heavyweights Apple, Google, Microsoft, and Amazon. Facebook is close behind in 8th position.



### Popular Posts

Future Horizons in Review - 5 Tech Conversations That Ruled in 2018

What to Expect from Immersive Technologies in 2019

A Short History of Artificial Intelligence

3 Non-financial Blockchain Applications Everyone Should Know

11 Important Truths About How

# Examples of 'Doing More' with Data

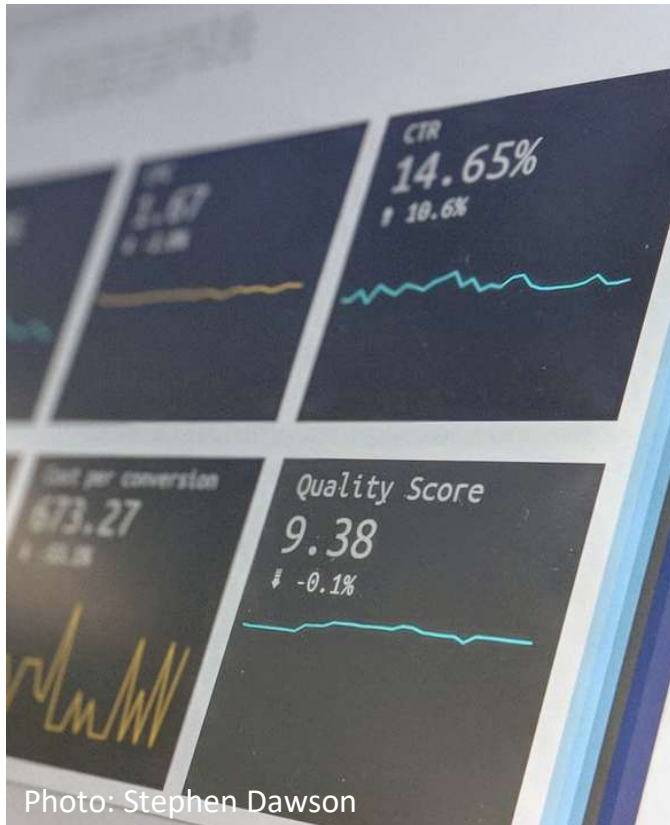
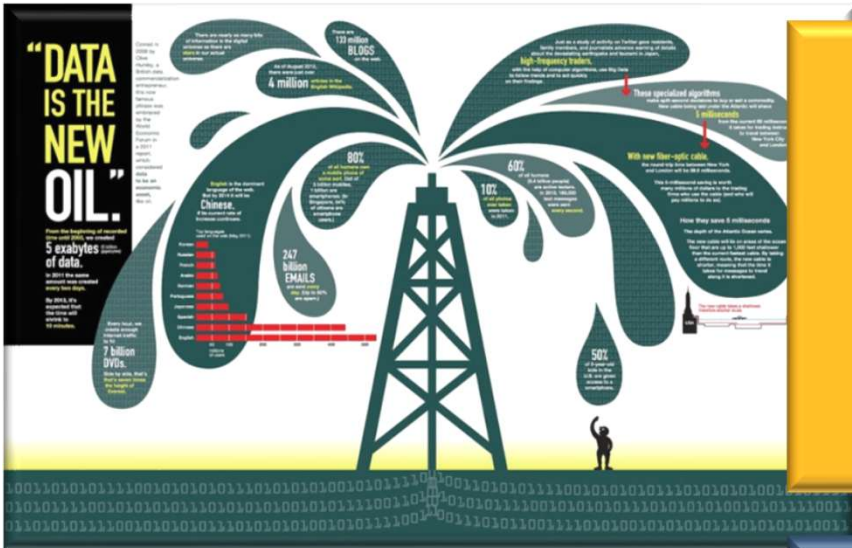


Photo: Stephen Dawson

- Enabling self-service reporting, dashboards, and for analytics to work with (near) real-time data
- Combining internal with external data to enrich analytical capabilities
- Accelerating AI/machine learning initiatives to discover new patterns or trends in the data
- Simplifying deployment of IoT technology to generate more data on the machines and business processes
- Offering edge analytics in which real-time data is analyzed continuously and near the place where the data is produced by the sensor or business process



# BIG DATA





**Data hasn't changed,  
it's just more of the same**

# **Data *consumption* has changed**

**Self-service BI**

**Embedded BI**

**Supplier- and Customer-driven BI**

**Applied AI in Text, Image, Video Analysis**

**Edge Analytics**

**Data Marketplace**

**Data Science**

**Automated decisions**

**...**

# Evolution of the Data Processing Landscape (1)

---

Source  
systems



# Evolution of the Data Processing Landscape (2)

Source systems



Analysis systems

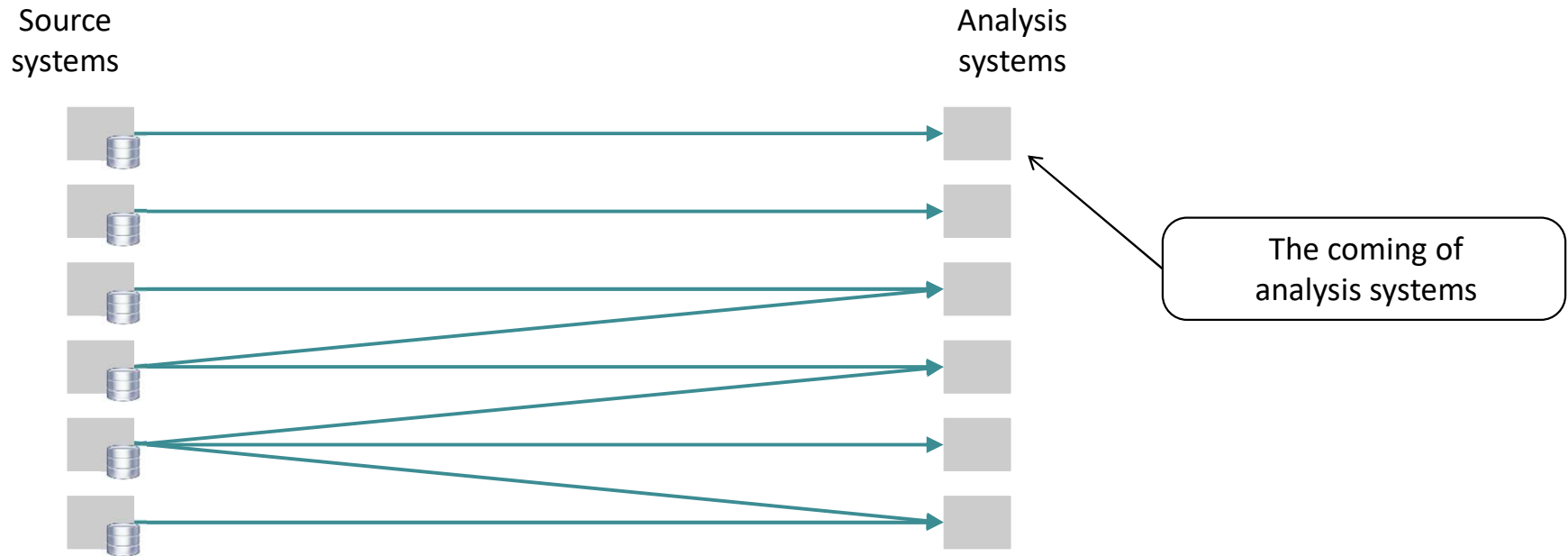


The coming of analysis systems

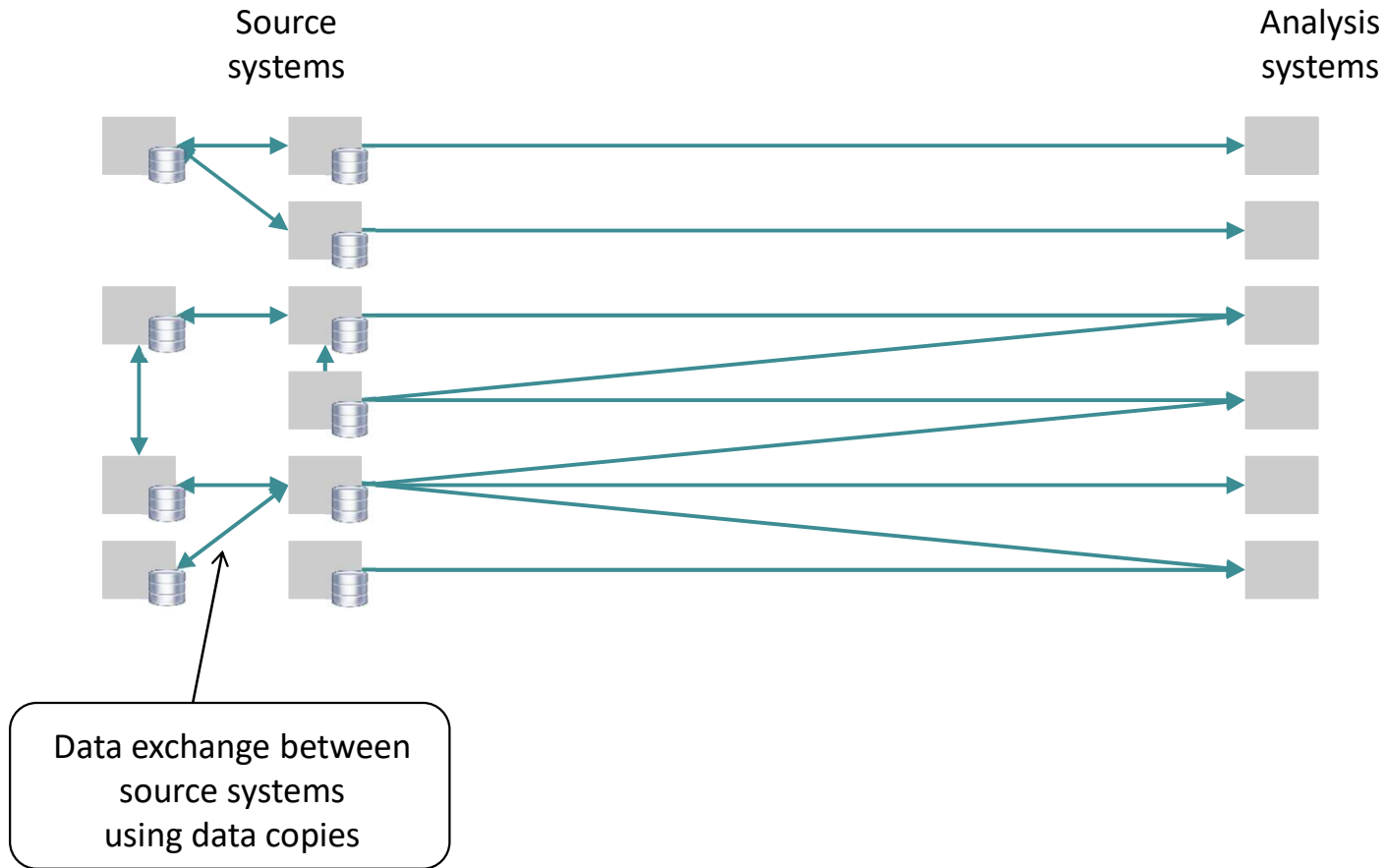
# Data for the Happy Few Only



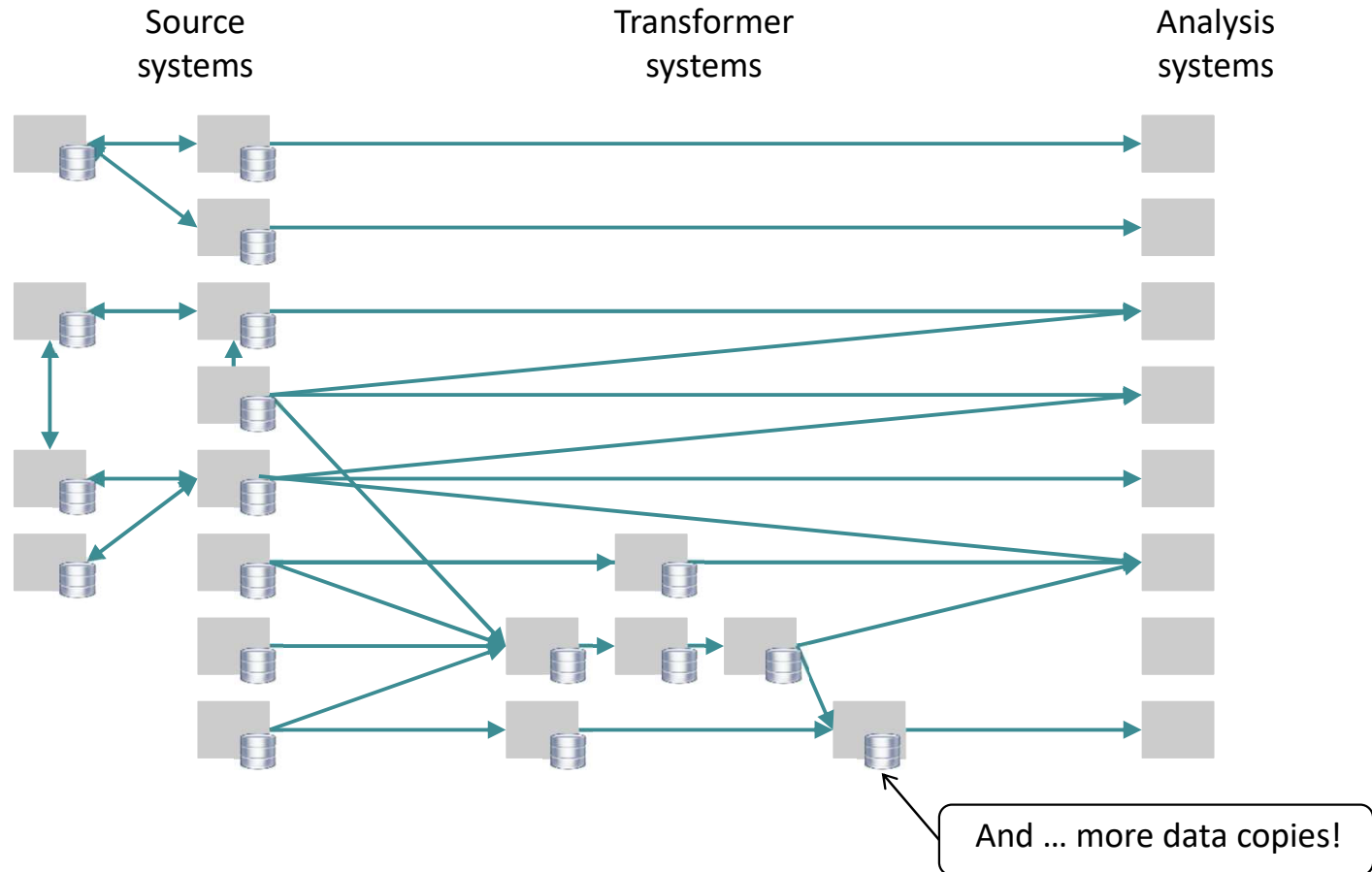
# Evolution of the Data Processing Landscape (2)



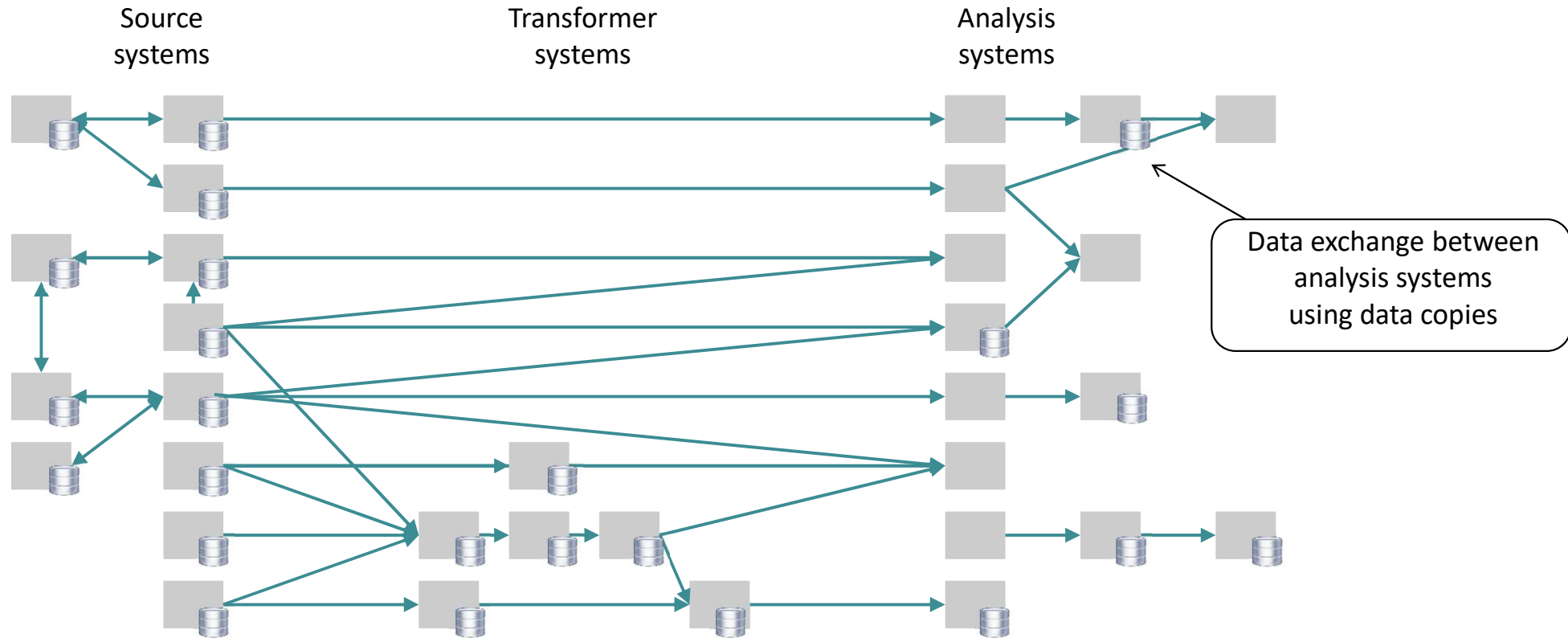
# Evolution of the Data Processing Landscape (3)



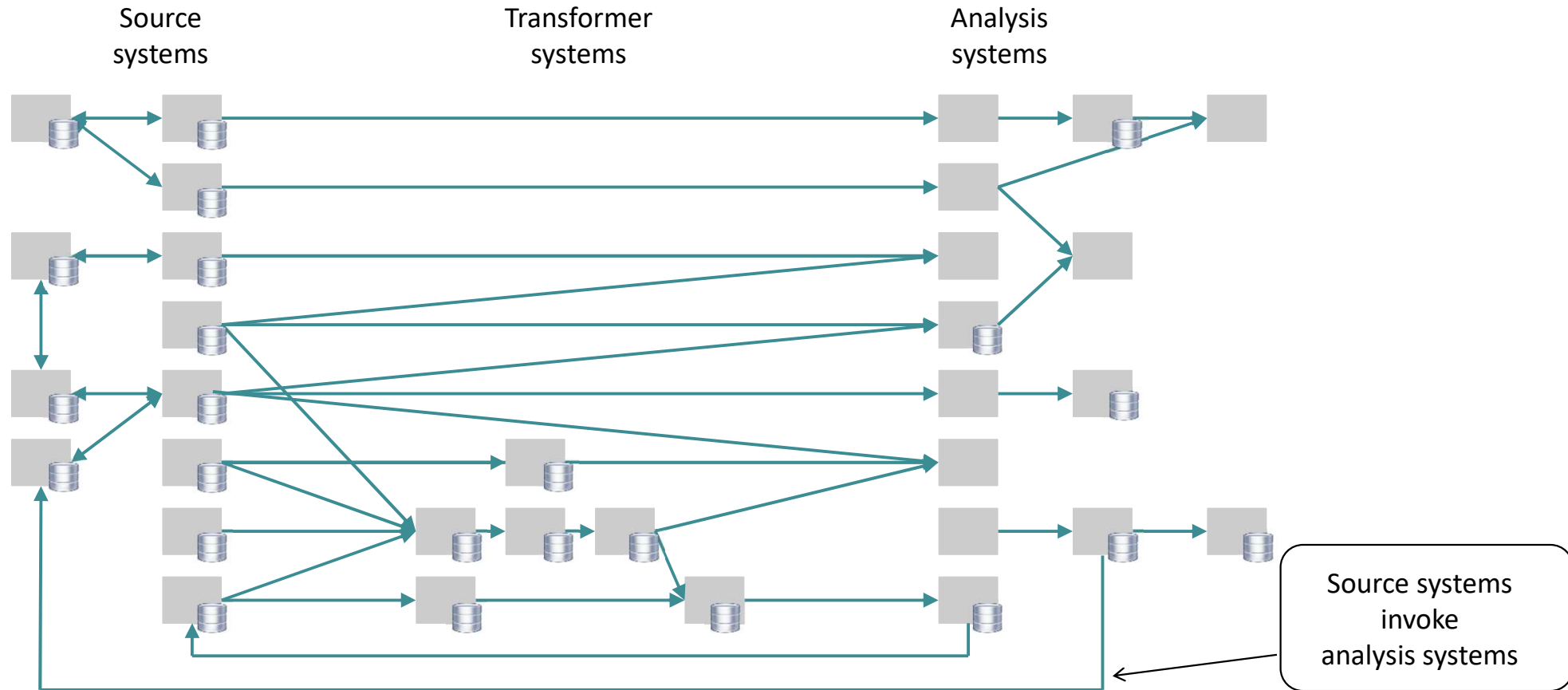
# Evolution of the Data Processing Landscape (4)



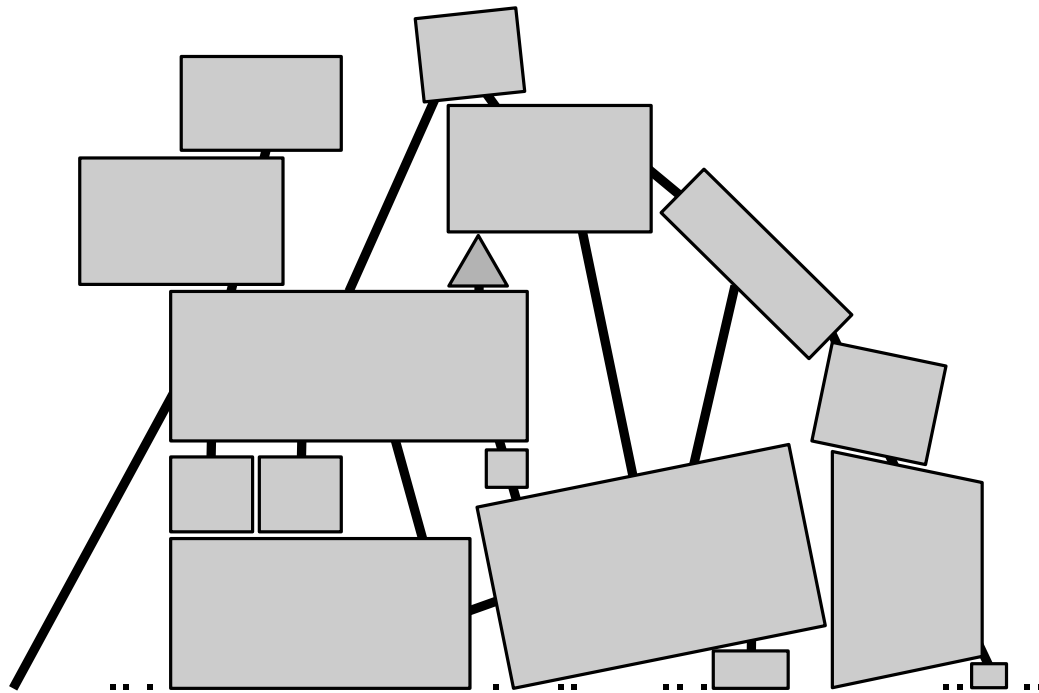
# Evolution of the Data Processing Landscape (5)



# Evolution of the Data Processing Landscape (6)



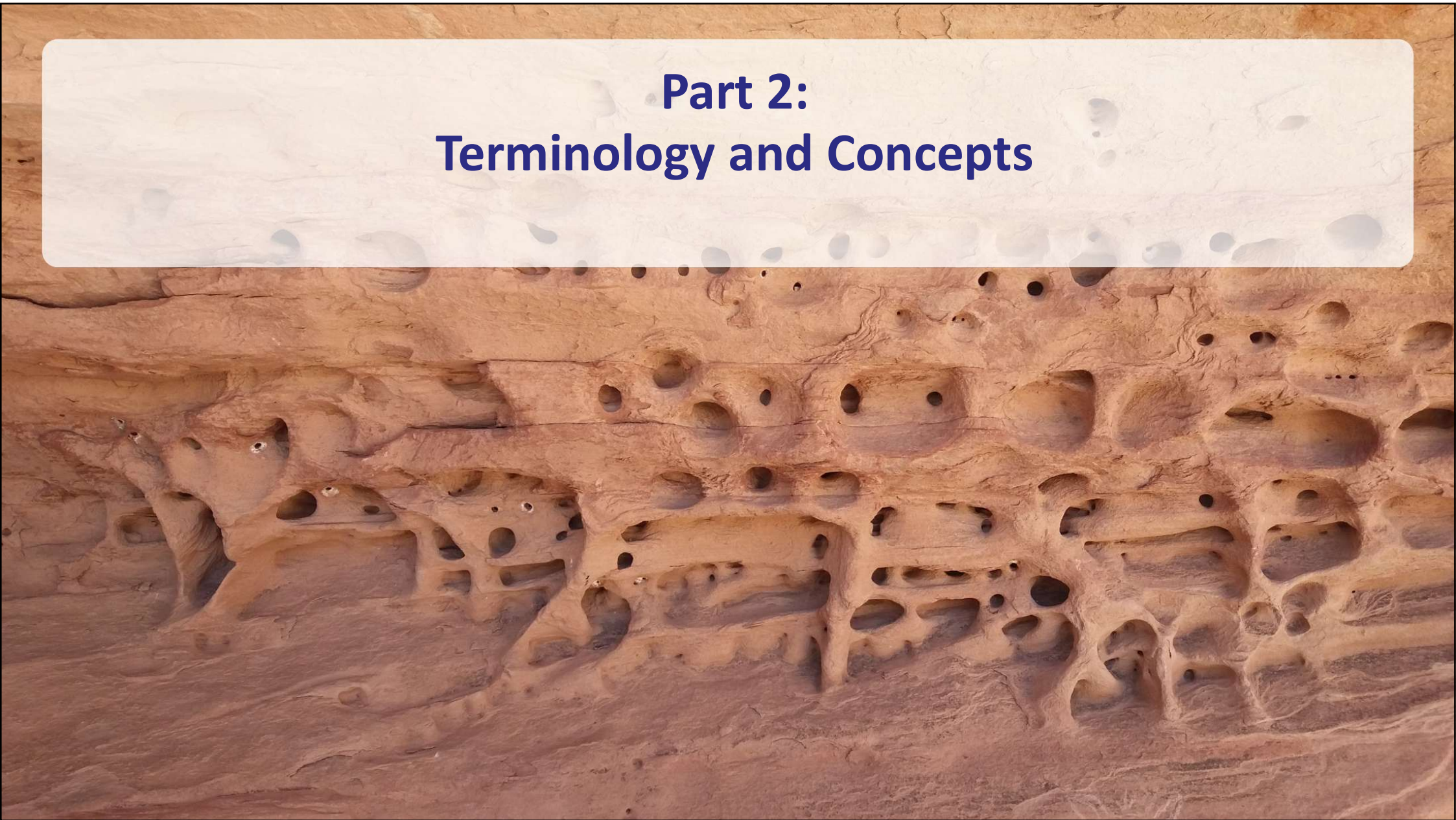
# “Wobbly” Foundation of the Data Processing Landscape



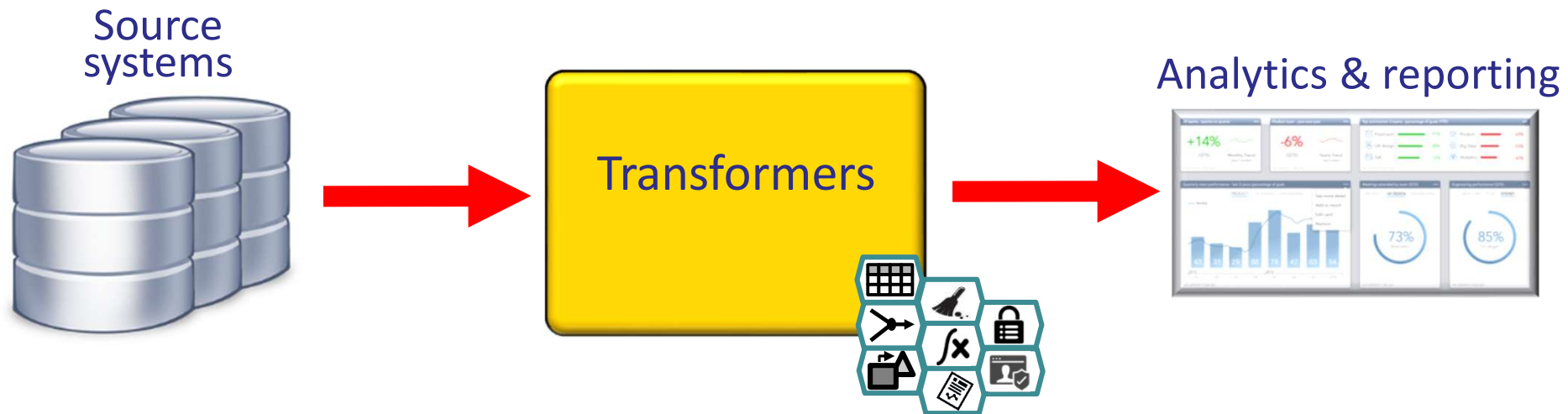


**Raising the Bar for ICT systems**

## Part 2: Terminology and Concepts



# The Transformers



- Data structure
- Integration
- Transformation
- Data security

- Data cleansing
- Analytical
- Visualization
- Data privacy

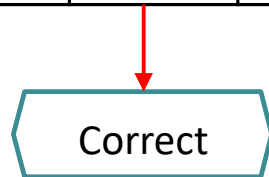
# Examples of Transformers

---

- Data value transformations
- Data structure transformations
- Aggregations
- Filters
- Groupings
- Calculations
- Integrations
- Technical corrections
- Functional corrections
- Anonymizations
- Historizations
- ...
- Summarize
- Keywords extraction
- Translate
- Named Entity Recognition (NER)
- Optical Character Recognition (OCR)
- Text-to-speech
- Speech-to-text
- Data tagging
- Sentiment analysis
- Pattern recognition
- ...
- Clustering
- Normalization
- Interpolation
- Extrapolation
- Binning
- Sampling
- Regression
- ...
- Vector-to-Raster Conversion
- Containment
- Clipping
- Spatial intersection
- Buffering
- ...

# Transformer for Correcting Data

Custid	Lastname	Gender	City	Zipcode
12345	Jansen	Man	Rotterdam, ZH	2651 BJ
23324	Visser	V	Delft, ZH	2289 BG
57657	Dekker	Vrouw	Zwolle, OV	4011 AE
65461	Brouwer	M	Breda, NB	4811 AD



Custid	Lastname	Gender	City	Zipcode
12345	Jansen	M	Rotterdam, ZH	2651 BJ
23324	Visser	V	Delft, ZH	2289 BG
57657	Dekker	V	Zwolle, OV	
65461	Brouwer	M	Breda, NB	4811 AD

# Transformer for Grouping Data

Custid	Transactionid	Datetime	Productid	Quantity	Price
12345	13463	202603121145	P34	3	3,50
12345	13463	202603121145	P66	2	2,75
12345	13463	202603121145	P87	1	1,95
12345	29365	202603130927	P66	2	2,75
57657	37721	202603131650	P12	1	4,00
57657	37721	202603131650	P66	2	2,75
65461	43846	202603141012	P73	4	0,75
65461	57290	202603151430	P19	5	2,40
65461	57290	202603151430	P66	3	2,75

Group,  
calculate

Transactionid	Datetime	Total amount
13463	202603121145	17,95
29365	202603130927	5,50
37721	202603131650	9,50
43846	202603141012	3,00
57290	202603151430	20,25

# Transformer for Integrating and Grouping Data

Custid	Lastname	Gender	City	Zipcode
12345	Jansen	Man	Rotterdam, ZH	2651 BJ
23324	Visser	V	Delft, ZH	2289 BG
57657	Dekker	Vrouw	Zwolle, OV	4011 AE
65461	Brouwer	M	Breda, NB	4811 AD

Custid	Transactionid	Datetime	Productid	Quantity	Price
12345	13463	202603121145	P34	3	3,50
12345	13463	202603121145	P66	2	2,75
12345	13463	202603121145	P87	1	1,95
12345	29365	202603130927	P66	2	2,75
57657	37721	202603131650	P12	1	4,00
57657	37721	202603131650	P66	2	2,75
65461	43846	202603141012	P73	4	0,75
65461	57290	202603151430	P19	5	2,40
65461	57290	202603151430	P66	3	2,75

Integrate,  
correct,  
transform,  
group,  
calculate

Custid	Lastname	Gender	City	Province	Zipcode	Number of trans	Total
12345	Jansen	M	Rotterdam	Zuid-Holland	2651 BJ	2	23,45
23324	Visser	V	Delft	Zuid-Holland	2289 BG	0	0,00
57657	Dekker	V	Zwolle	Overijssel	4011 AE	1	9,50
65461	Brouwer	M	Breda	Noord-Brabant	4811 AD	2	23,25

# Transformer for Determining Keywords

Veel organisaties zijn zich ervan bewust dat actie noodzakelijk is, maar ze weten niet welke stappen ze moeten nemen. Het bouwen van weer een nieuw transformatorsysteem, zelfs met de nieuwste technologieën en inzichten, lost het onderliggende probleem niet op. Zoals een bekende uitspraak, die vaak aan Albert Einstein toegeschreven wordt, luidt: "Het is waanzin om steeds hetzelfde te doen en toch een ander resultaat te verwachten."

Natuurlijk kan elke organisatie een eigen data-architectuur ontwikkelen die alle genoemde problemen aanpakt. Maar waarom alles vanaf nul uitvinden? Waarom niet een bestaande referentie data-architectuur als uitgangspunt nemen, deze aanvullen met ontbrekende onderdelen en overbodige elementen weglaten? Het wiel telkens opnieuw uitvinden kost onnodig veel tijd en geld. Het hergebruiken van de beste ervaringen, tips en do's en don'ts die in een referentie data-architectuur zijn vastgelegd, bespaart enorm veel tijd en kosten en verkleint de kans op een minder effectieve architectuur.

Daarnaast wordt data-uitwisseling veel eenvoudiger wanneer organisaties vergelijkbare architecturen hanteren. Dit is te vergelijken met het gemak dat ontstaat doordat alle landen in Europa dezelfde stopcontacten en voltspanning gebruiken.

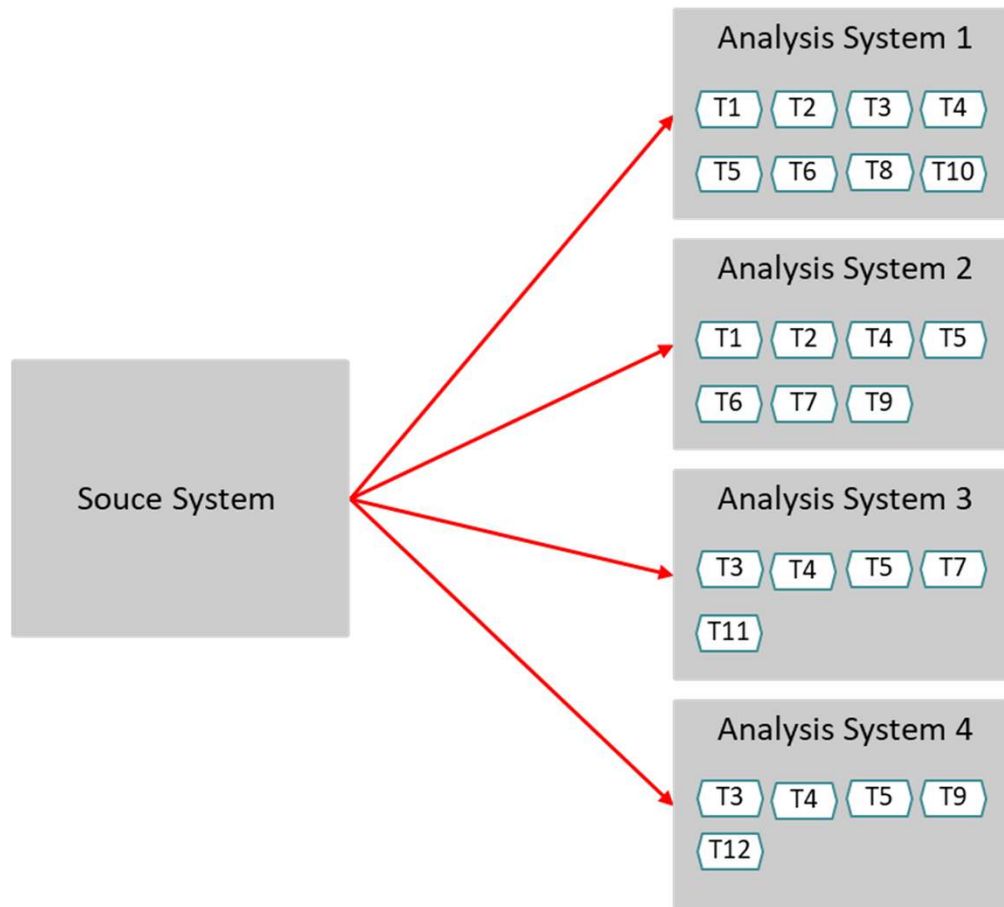
Uiteraard zijn er organisaties die zo uniek zijn in hun dataverwerking dat een referentie architectuur minder relevant is. Echter, dergelijke gevallen zijn zeldzaam. Vooral organisaties binnen dezelfde sector hebben over het algemeen vergelijkbare eisen en wensen en kunnen prima uit de voeten met sterk overeenkomende data-architecturen. Denk bijvoorbeeld aan gemeenten, retailbedrijven, ziekenhuizen en transportbedrijven. Waarom alles zelf uitvinden?

Om te voorkomen dat organisaties alles zelf uitdenken en ontwikkelen, is de referentie data-architectuur Delta gedefinieerd. Enkele uitgangspunten van Delta zijn: ten eerste, transformatorfunctionaliteit verplaatsen naar de bronsystemen en de beheerders van de bronsystemen er verantwoordelijk voor maken. Ten tweede, het gebruik van datakopieën moet geminimaliseerd worden. Ten derde, data moet eenvoudig vindbaarder zijn. Ontwikkelaars en gebruikers moeten snel de benodigde gegevens kunnen terugvinden, zodat ze direct kunnen starten met het ontwikkelen van analysesystemen en er zeker van zijn dat ze de juiste data gebruiken.

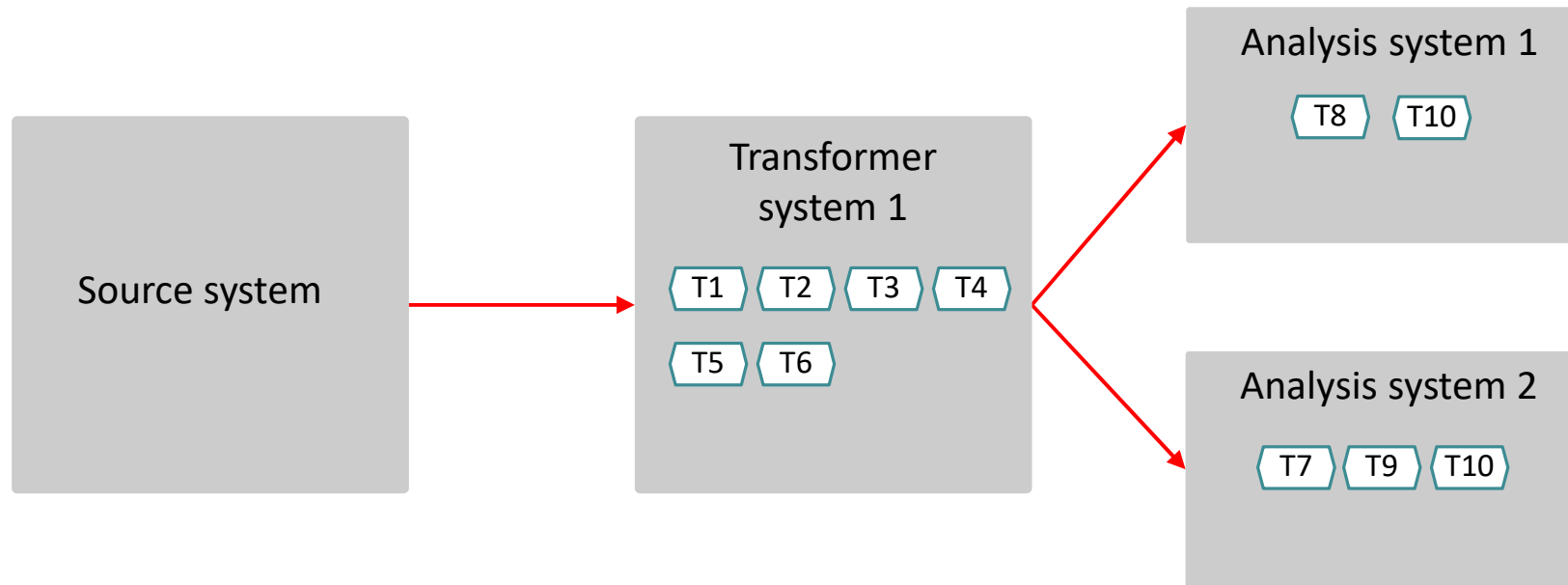
Determine  
keywords

Referentiearchitectuur  
Hergebruik  
Transformatiesystemen  
Data-uitwisseling  
Efficiëntie  
Standaardisatie  
Bronsystemen  
Datakopieën  
Vindbaarheid

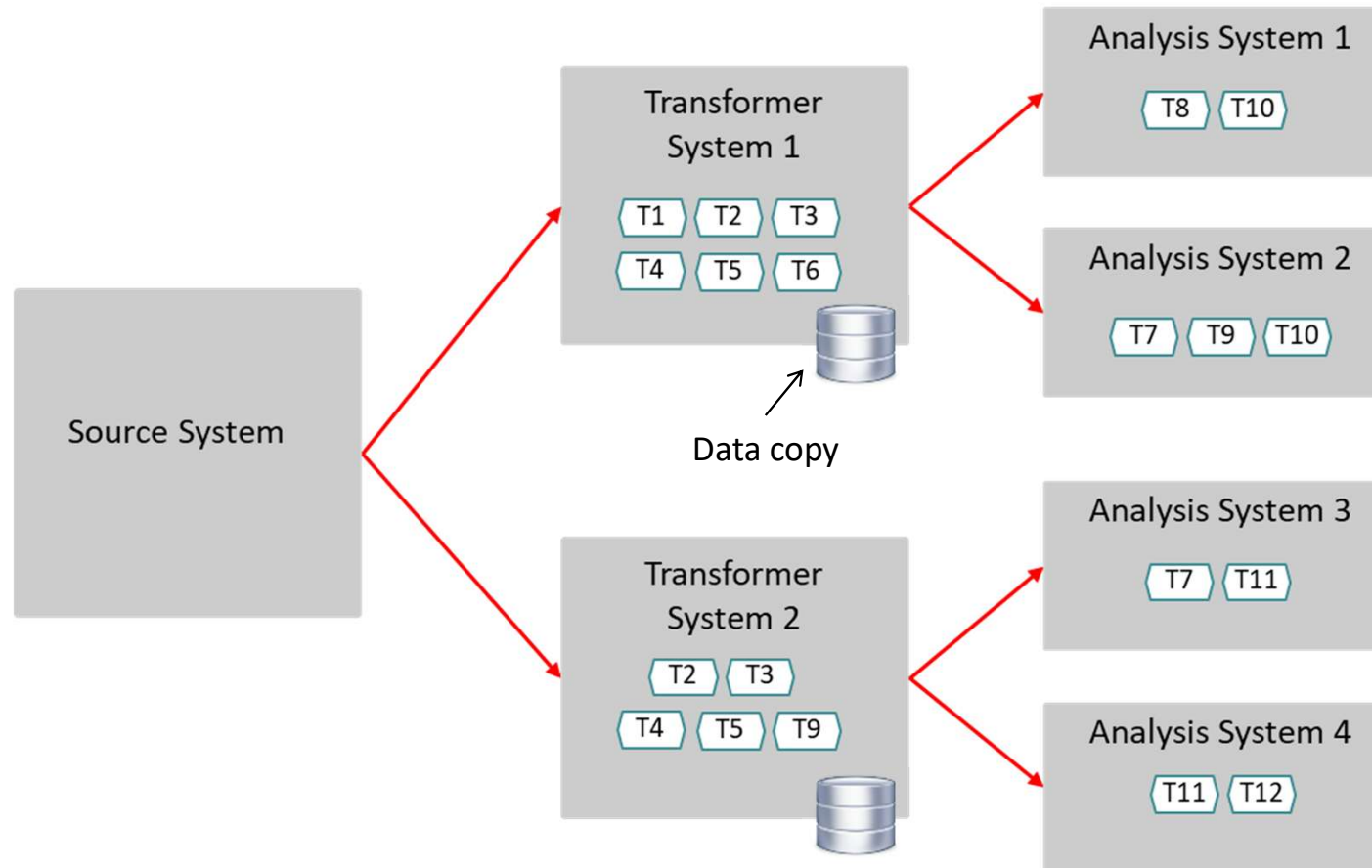
# Duplicate Transformers in Analysis Systems



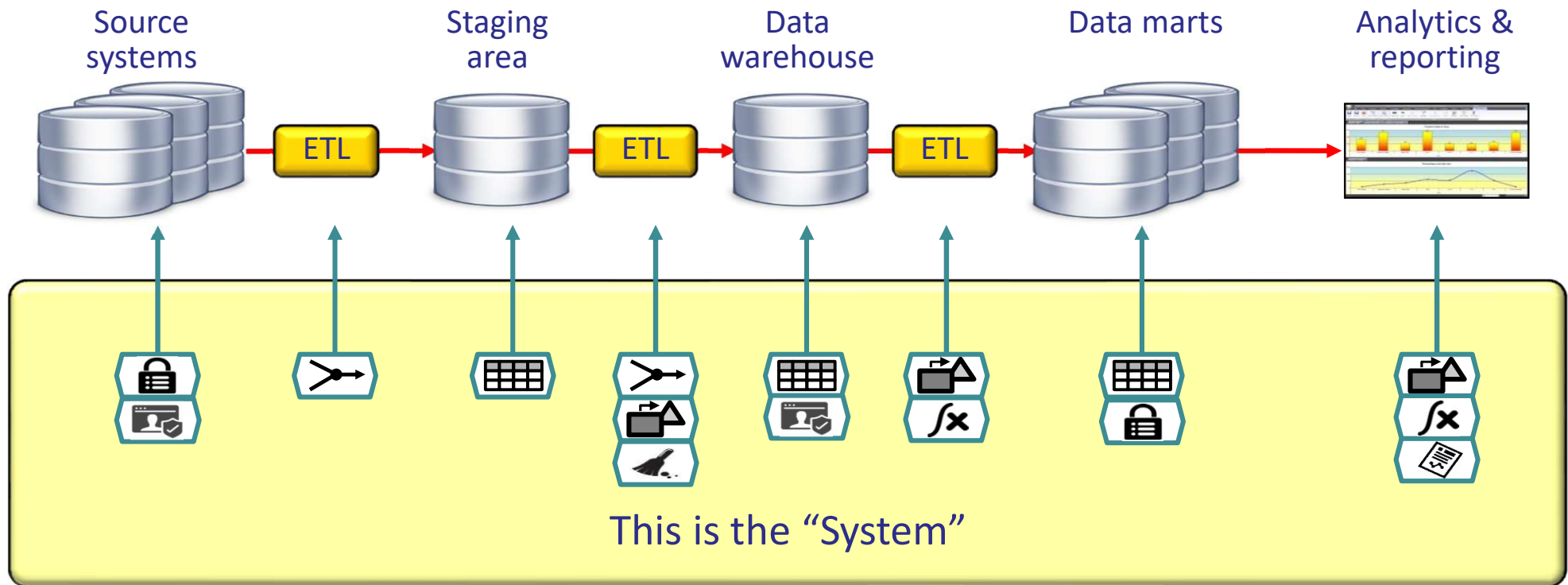
# Centralizing Transformers



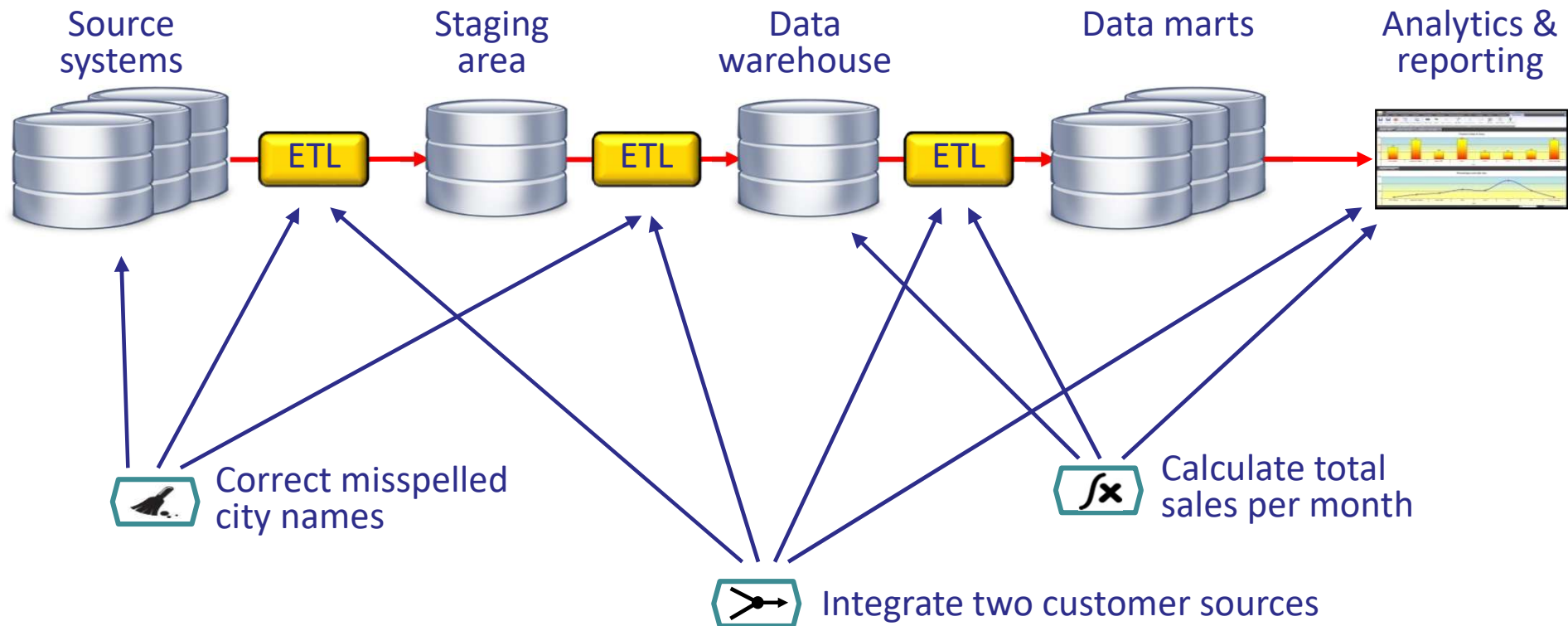
# Duplicate Transformers in Transformer Systems



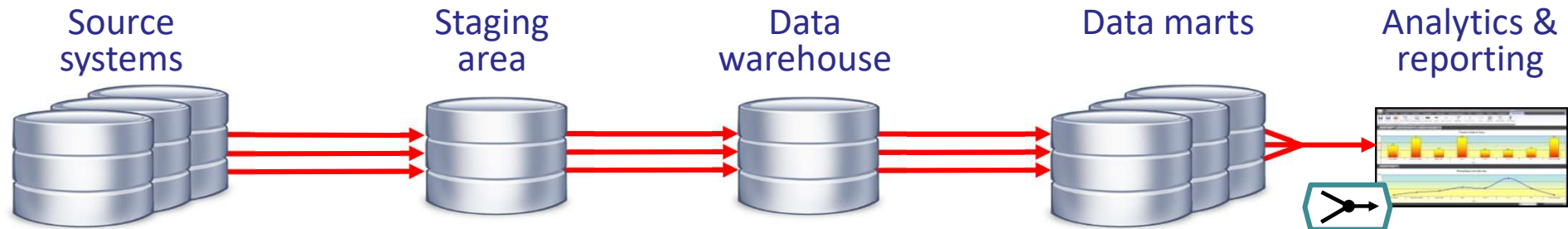
# Transformers in Data Warehouse Environments



# Where to Implement Transformers?

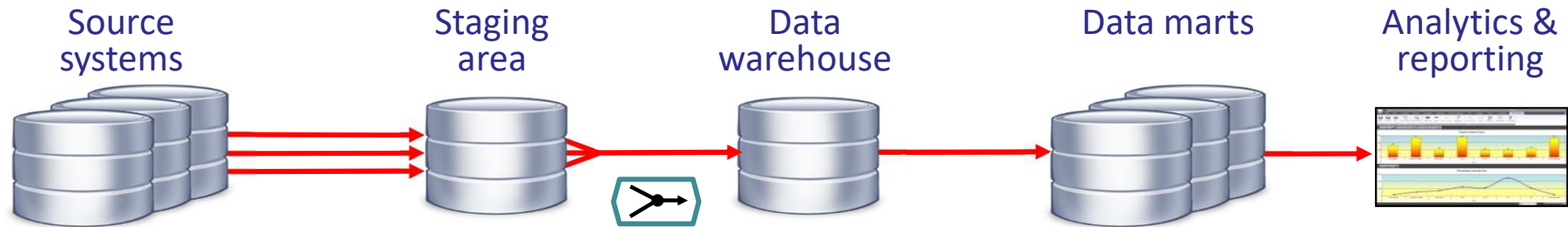


# Example: Integration and Aggregation (1)



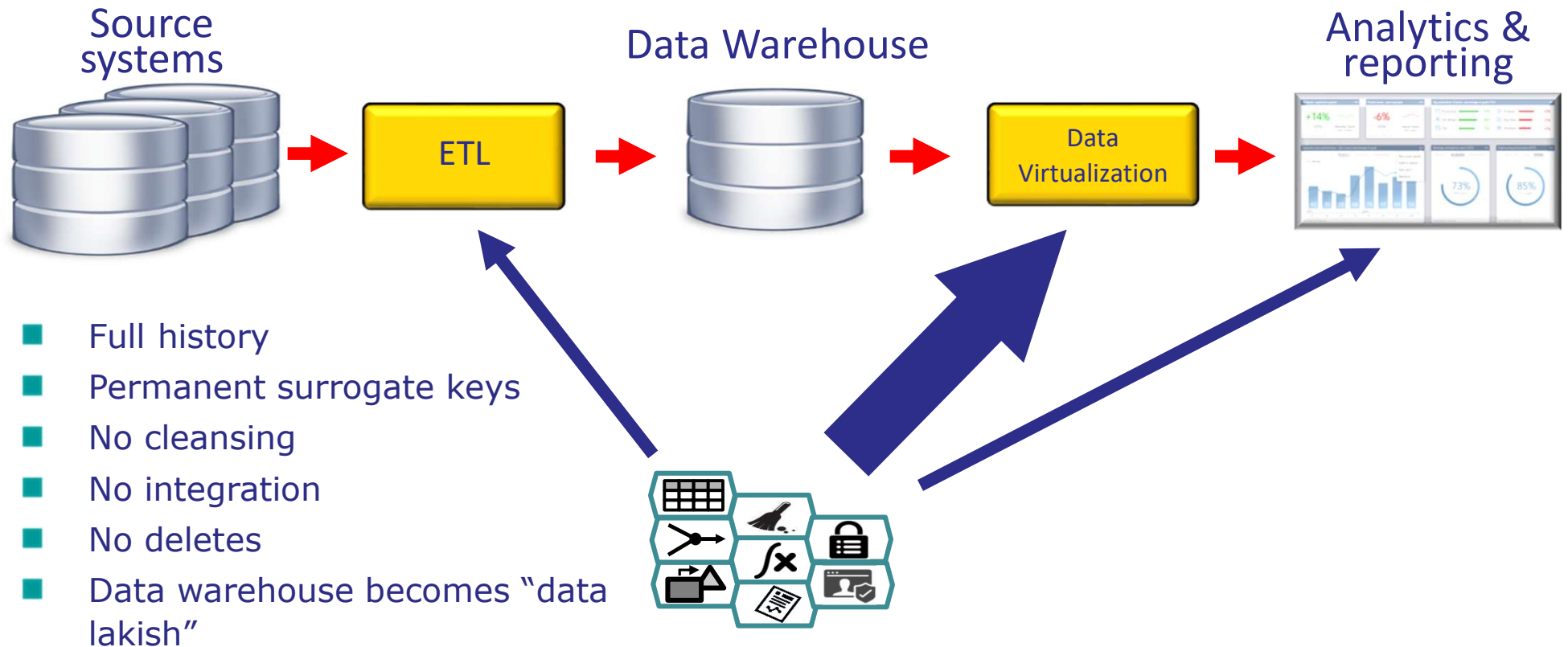
- Slow processing of integration logic in reports
- Complex queries in reports
- No sharing of integration logic across reports and tools
- Potential errors and inconsistencies in reports
- Fast copying and lower data latency
- Data structure of source database determines all data structures
- Use of original raw data possible
- What about integration errors?

## Example: Integration and Aggregation (2)



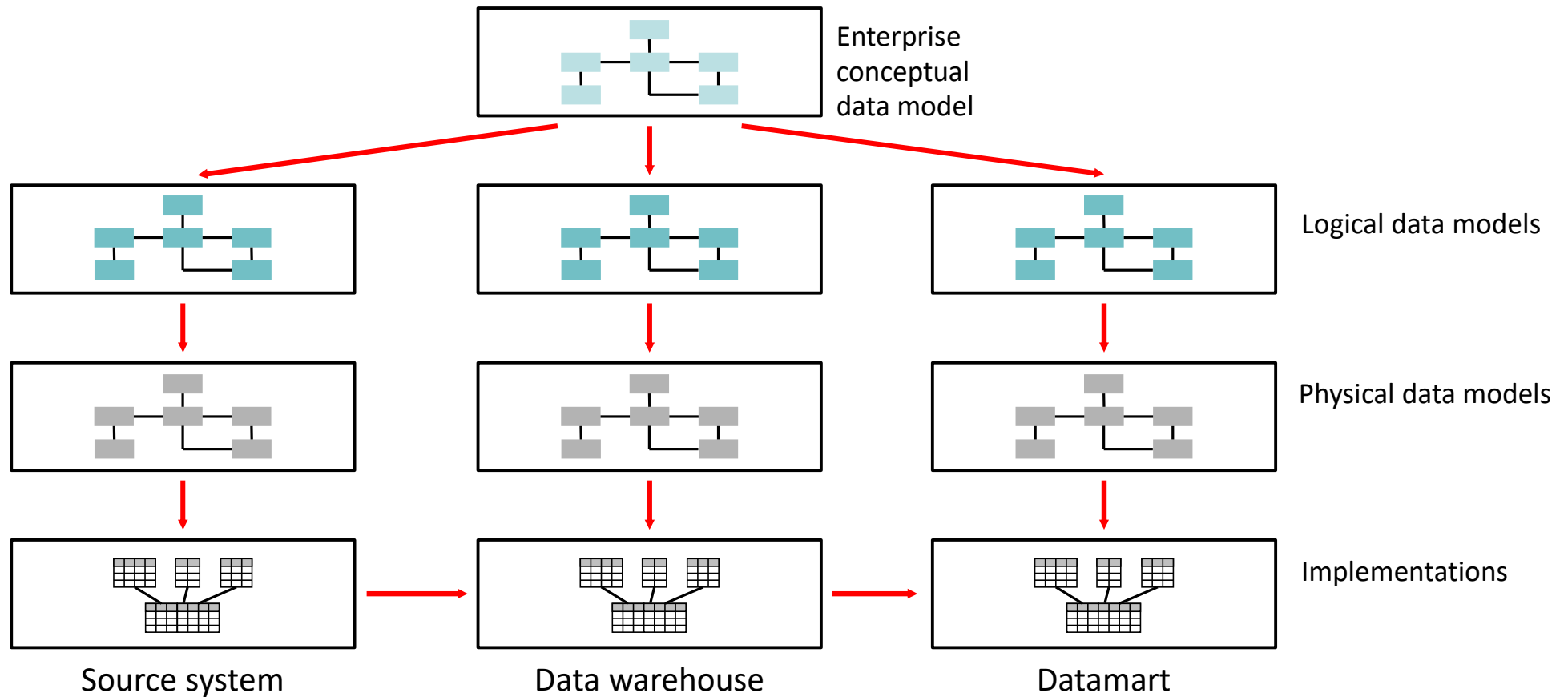
- Fast processing of integration logic in reports
- Simpler queries in reports
- Sharing of integration logic across reports and tools
- Potential errors and inconsistencies in ETL
- Slower copying and higher data latency
- Data structure of source database does not determine all data structures
- Use of original raw data possible
- Integration errors easier to fix

# Implementing the Transformers

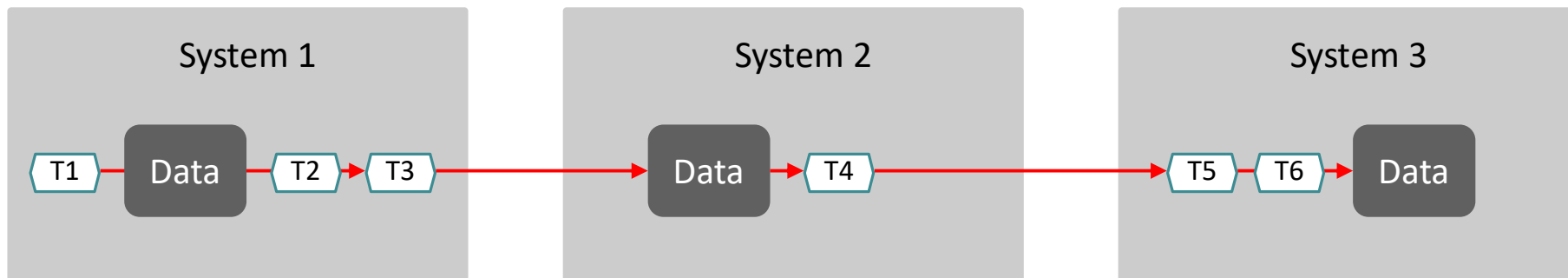


- Full history
- Permanent surrogate keys
- No cleansing
- No integration
- No deletes
- Data warehouse becomes "data lakish"

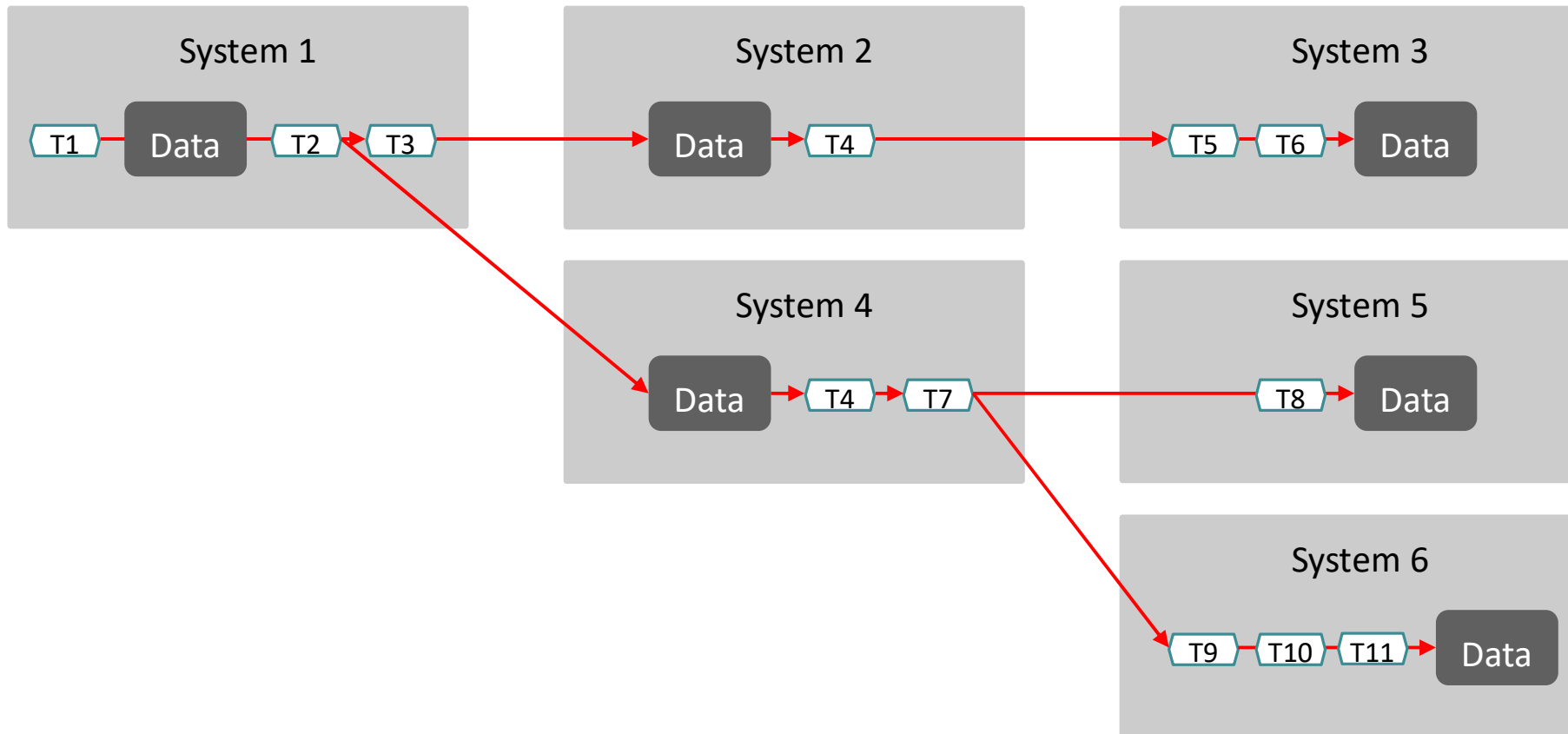
# Vertical and Horizontal Lineage



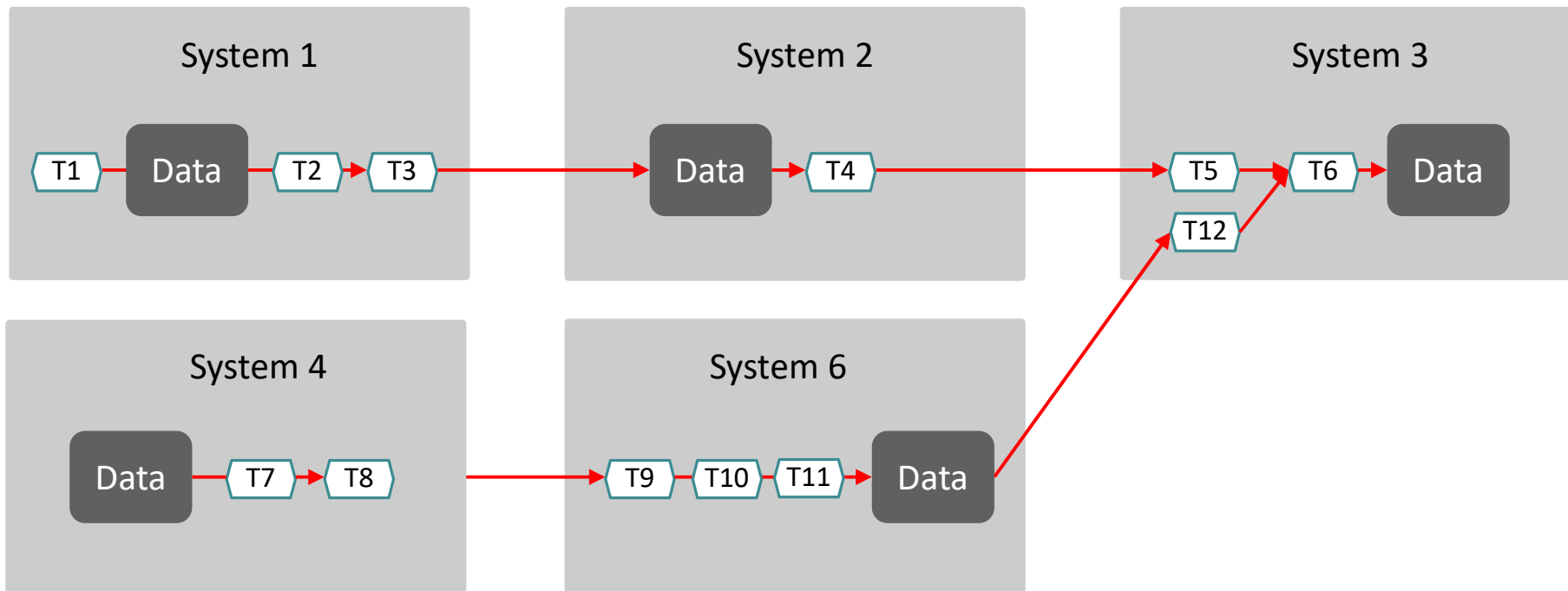
# Example of a Data Path with Transformers



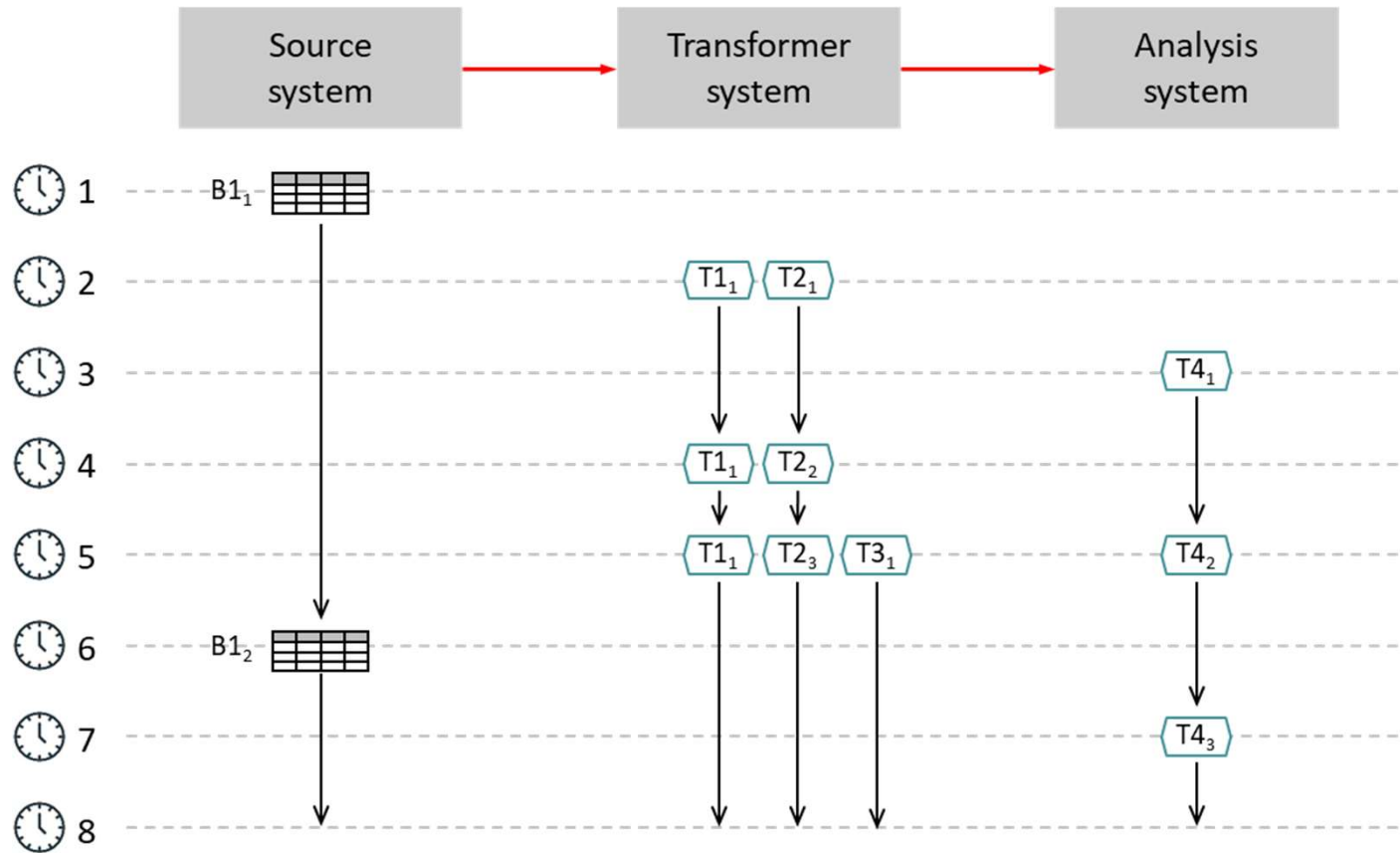
# Data Path with Splits



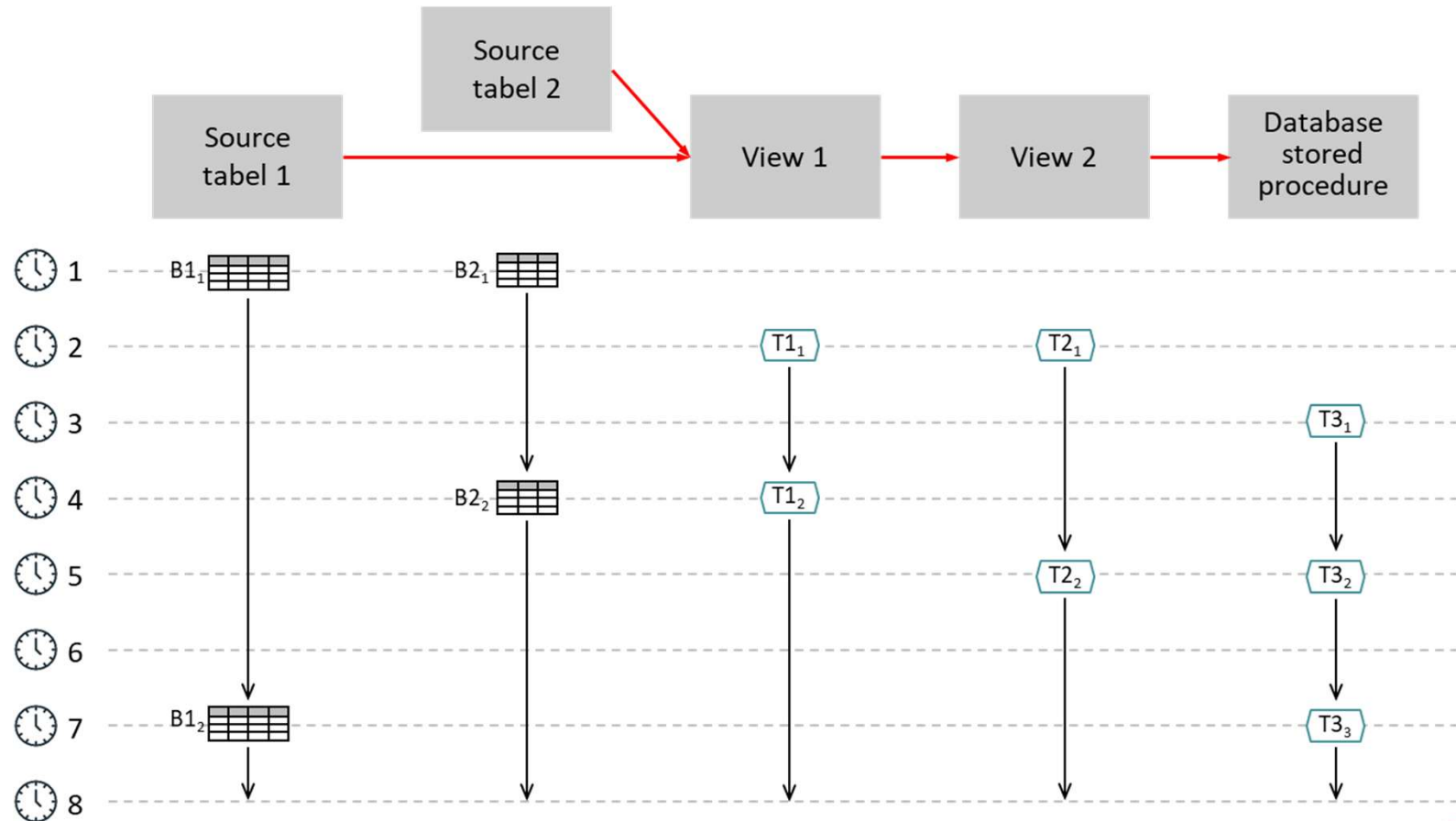
# Data Path with Joins



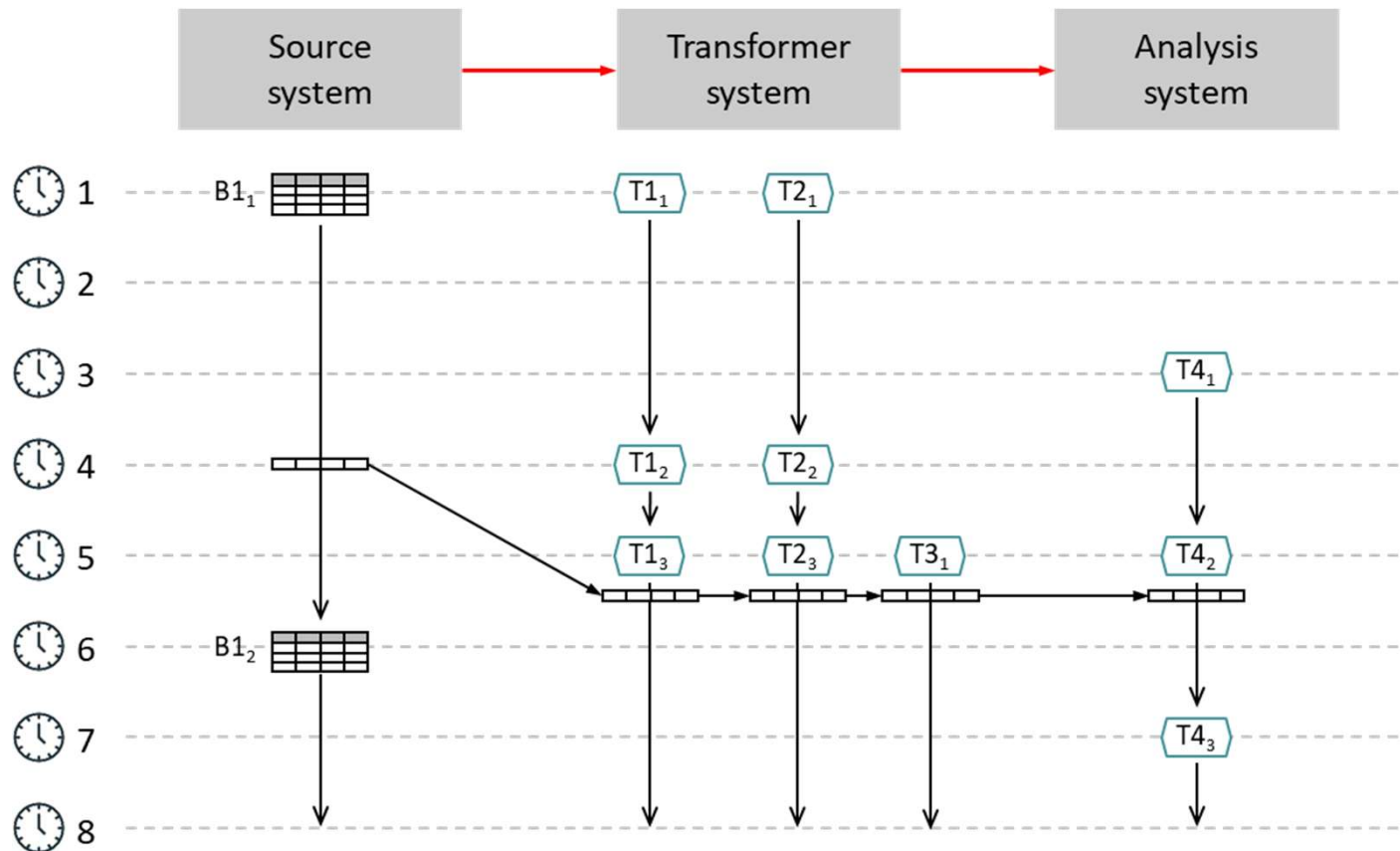
# Example Horizontal Lineage (1)



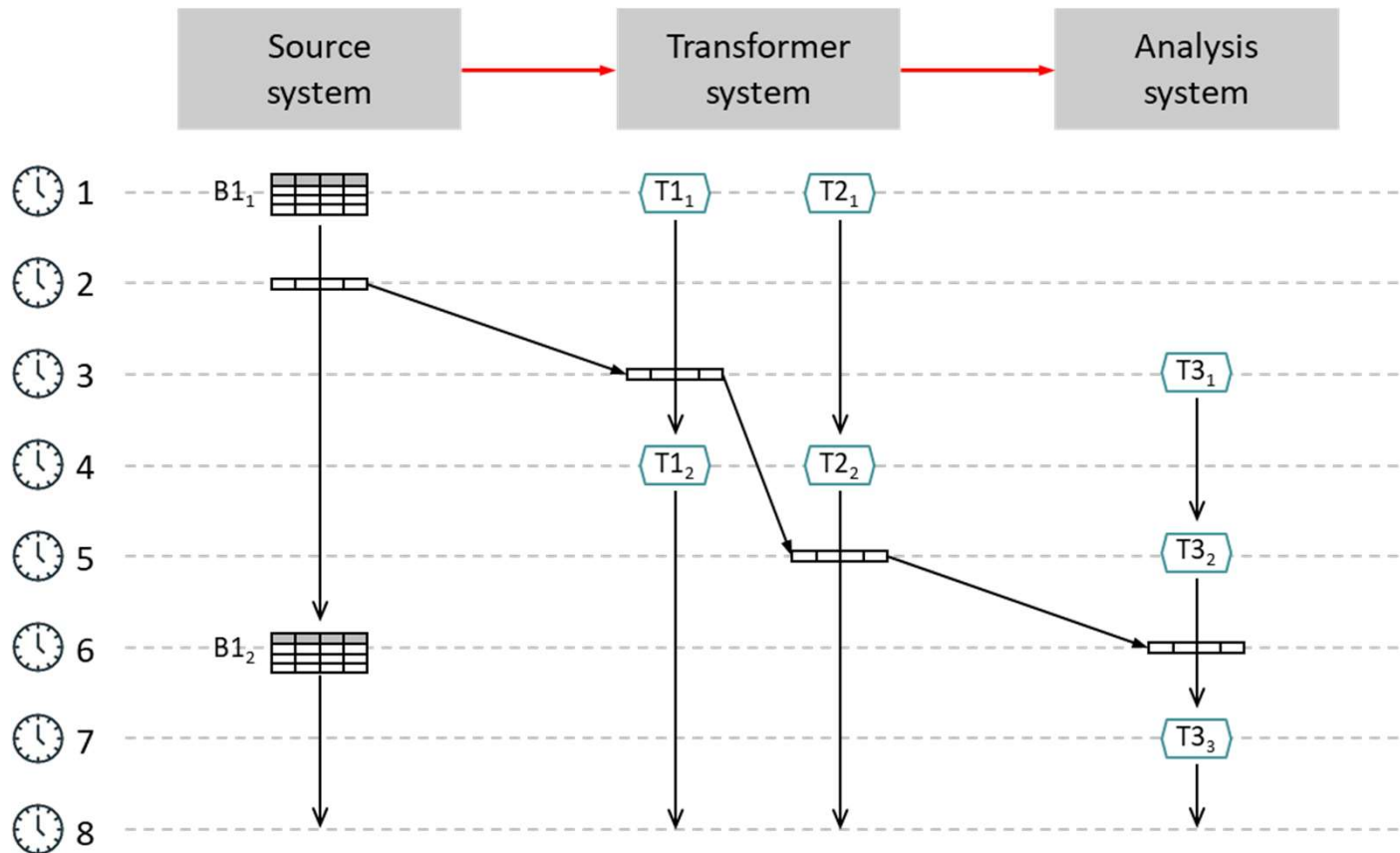
# Example Horizontal Lineage (2) Within a Database



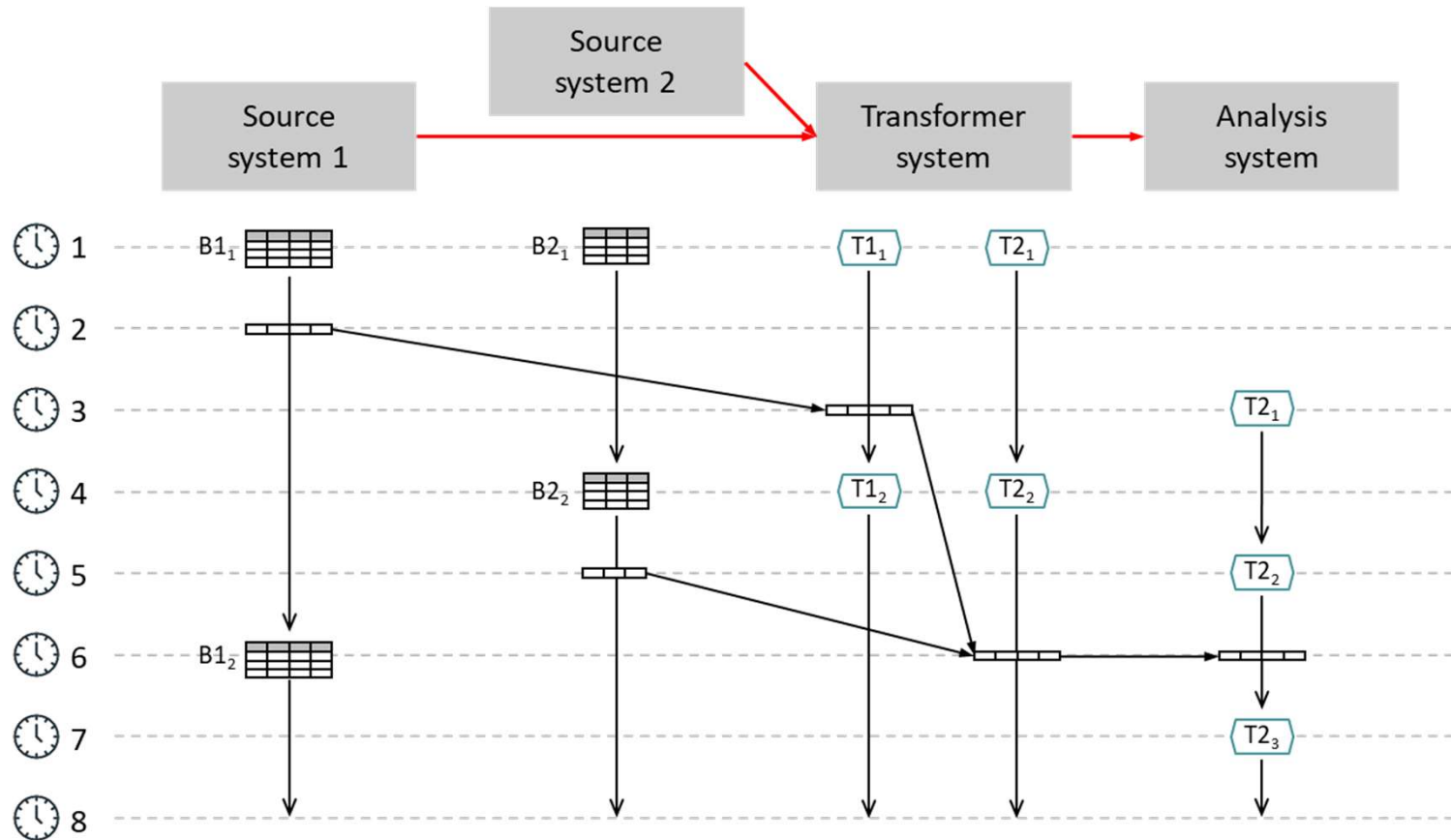
# Operational Lineage (1) Simple Realtime Query



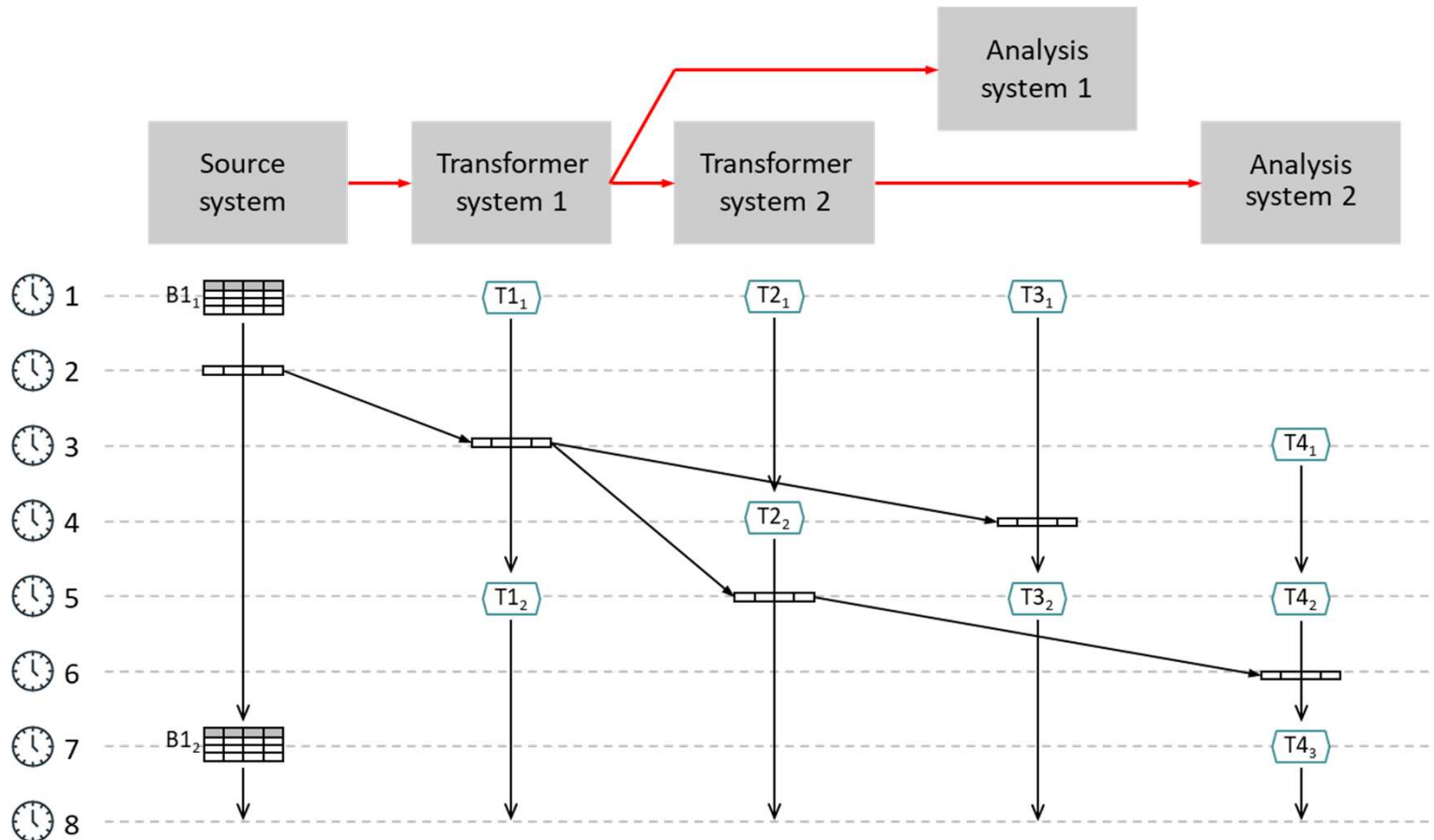
# Operational Lineage (2) With Data Copies



# Operational Lineage (3) Integration



# Operational Lineage (4) Split



# Master Data: Two Customer Tables

Customer table in **Sales System**

ID	Name	Initials	Date Entered	City	State
12345	Young	N	Aug 4, 2008	San Francisco	CA
23324	Stills	S	Sep 10, 2009	New Orleans	LA
57657	Furay	R	Oct 16, 2010	Yellow Springs	OH
65461	Palmer	B	Nov 22, 2011	Boston	MA
...	...	...	...	...	...

Customer table in **Finance System**

ID	Name	Initials	Date Entered	City	State
C5729	Young	N	Sep 16, 2007	San Francisco	CA
LA781	Stils	S	Dec 8, 2010	New Orleans	LA
J7301	Furay	R	Jan 10, 2008	Yellow Springs	OH
K8839	Palmer	B	Feb 11, 2009	New York	NY
...	...	...	...	...	...

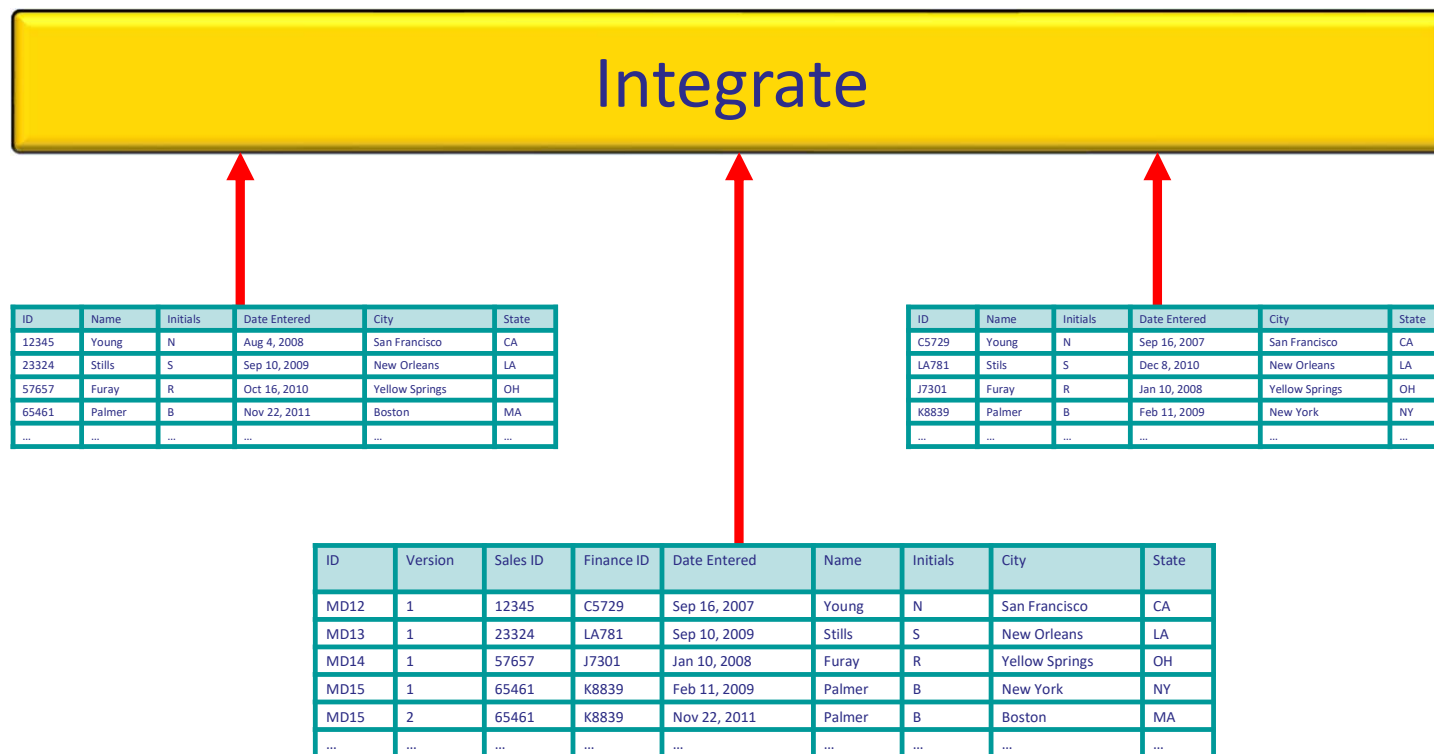
# The Master Customer Table

---

Master Customer table

ID	Version	Sales ID	Finance ID	Date Entered	Name	Initials	City	State
MD12	1	12345	C5729	Sep 16, 2007	Young	N	San Francisco	CA
MD13	1	23324	LA781	Sep 10, 2009	Stills	S	New Orleans	LA
MD14	1	57657	J7301	Jan 10, 2008	Furay	R	Yellow Springs	OH
MD15	1	65461	K8839	Feb 11, 2009	Palmer	B	New York	NY
MD15	2	65461	K8839	Nov 22, 2011	Palmer	B	Boston	MA
...	...	...	...	...	...	...	...	...

# Joining Needs Master Data



# Example: Cohelion (Master Data-driven BI)

Accounts Payable	30,809	35,972	37,370	37,668	37,346	33,360	34,141	42,505
<b>Profit - Loss Accounts</b>								
Gross profit	79,150	56,450	61,467	77,134	<b>Actual</b>	<b>185,254.00</b>		60,400
EBIT	11,327	-732	-1,247	11,101	Manual	20,248.00		
					SalesForce - Feed	30,855.00		-374
					SAP - Feed	134,151.00		
Sales Income	201,736	167,789	170,810	222,655	199,000	185,254	7,600	131,000
Rental Income	0	0	0	0	0	0	0	0
Other income	0	0	0	0	0	0	0	0
<b>Expense Accounts</b>								
Office Expense	812	681	790	532	960	372	1,010	455

A wide-angle photograph of a desert canyon with layered red rock formations and a winding river. The sky is blue with scattered white clouds. A semi-transparent white box with rounded corners is overlaid on the top half of the image, containing the title text.

# **Part 3: Introduction to Data Architectures**

# What is a Data Architecture?

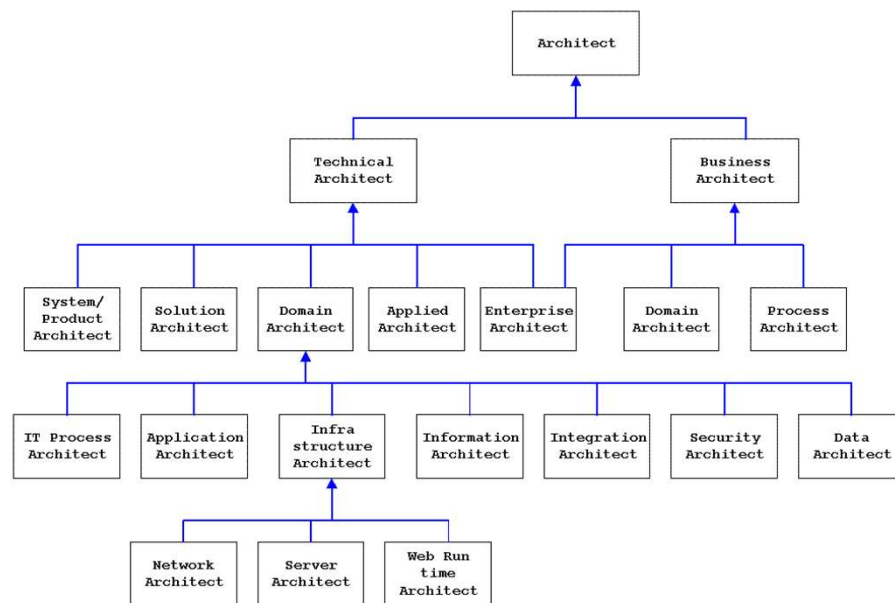


- Wikipedia: A *data architecture* is composed of models, policies, rules or standards that govern which data is collected, and how it is stored, arranged, integrated, and put to use in data systems and in organizations.
- Examples of data architectures:
  - Data warehouse architecture
  - Data streaming architecture
  - Transactional system

# Data Architects versus Solutions Architects

Data Architects	Solutions Architects
... focus on how information moves across the system from one application to another	... look at the overall technological environment of the company
... collaborate with clients to determine the specifications of the project, as well as the business goals that will align with the collected data	... meet with their clients and establish their specific technology needs based on their business objectives
... design the data model for the organization; where to store the customer data, how to retrieve the data; who can read the data	... has a more technical point of view. Do we select a cloud solution, or on premise? What will the network look like? How will everything be connected without failures?

# So Many Different Types of Architects



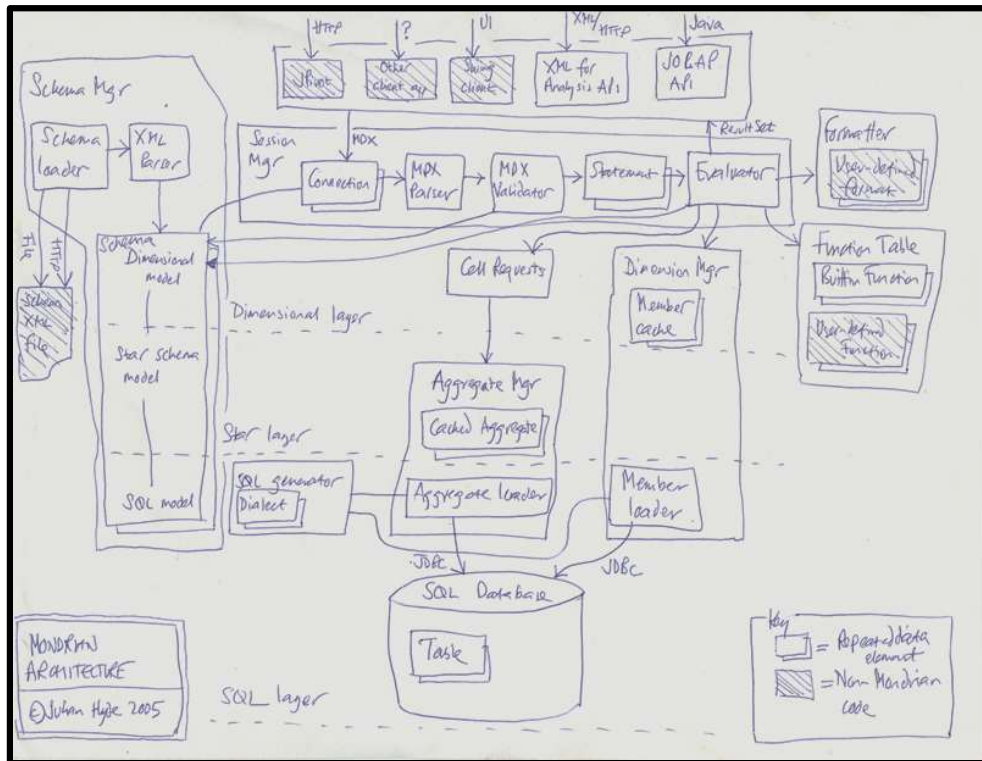
Paul Catalin Tomoiu : In [the] IT world, an architect is a person with enough knowledge to find a high level solution to a challenge in an IT environment.

The challenge type, defines the nature of the architecture role.

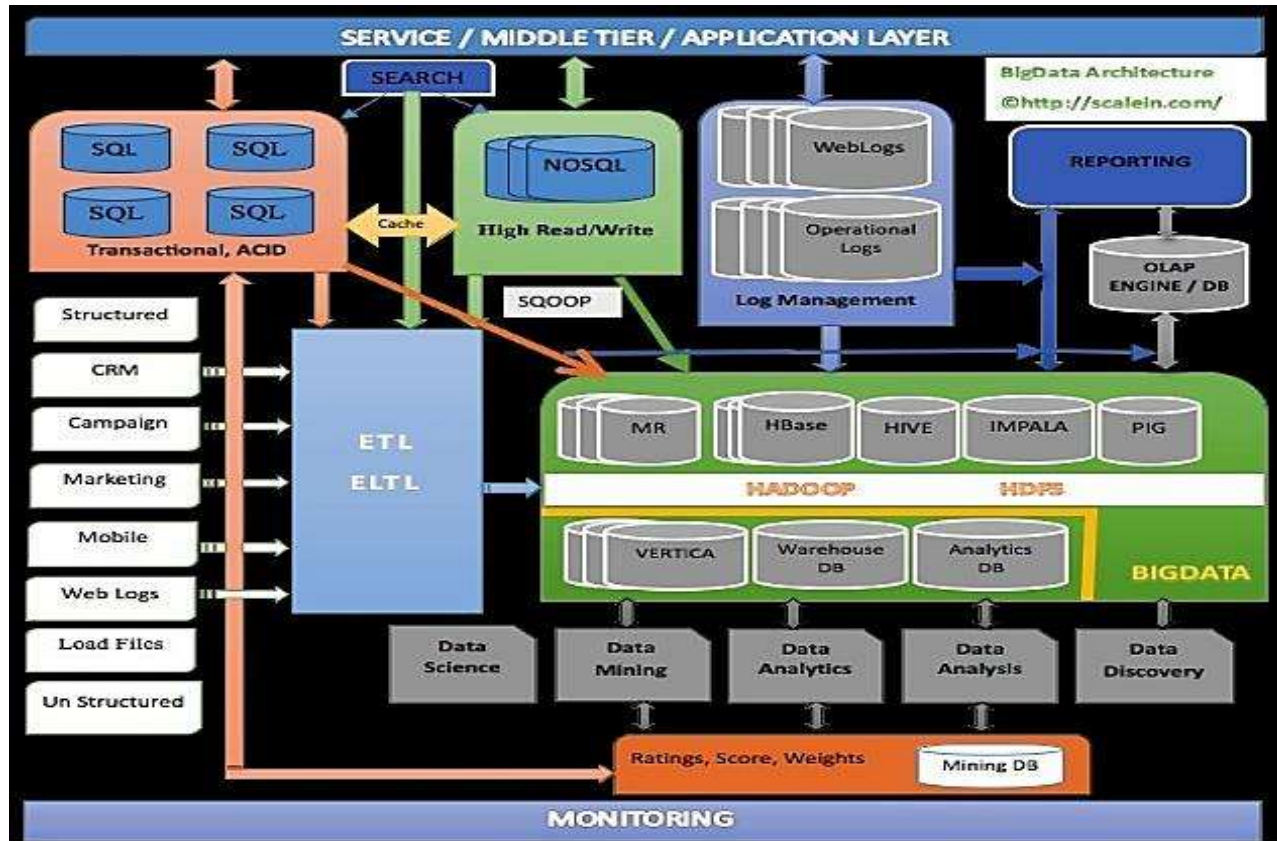
We can speak about an Enterprise Architecture, a Business Architecture, Data Architecture, Solution Architecture, IT Technical Architecture, Application Architecture, Software Architecture, Security Architecture, etc.

Source: <https://blog.prabasiva.com/2008/08/21/different-types-of-architects/>

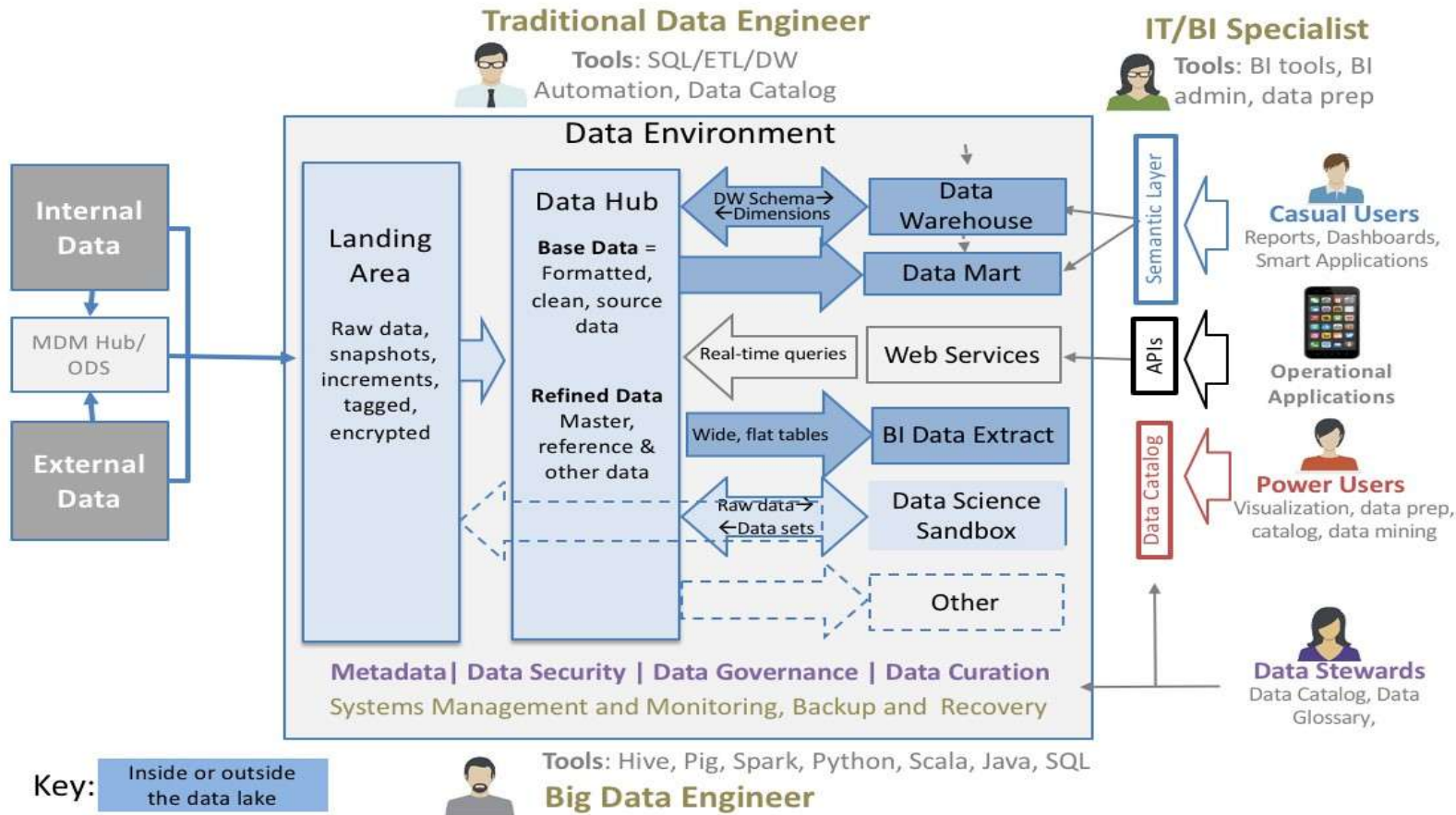
# The Birth of a Data Architecture



# Example Data Architecture (1)

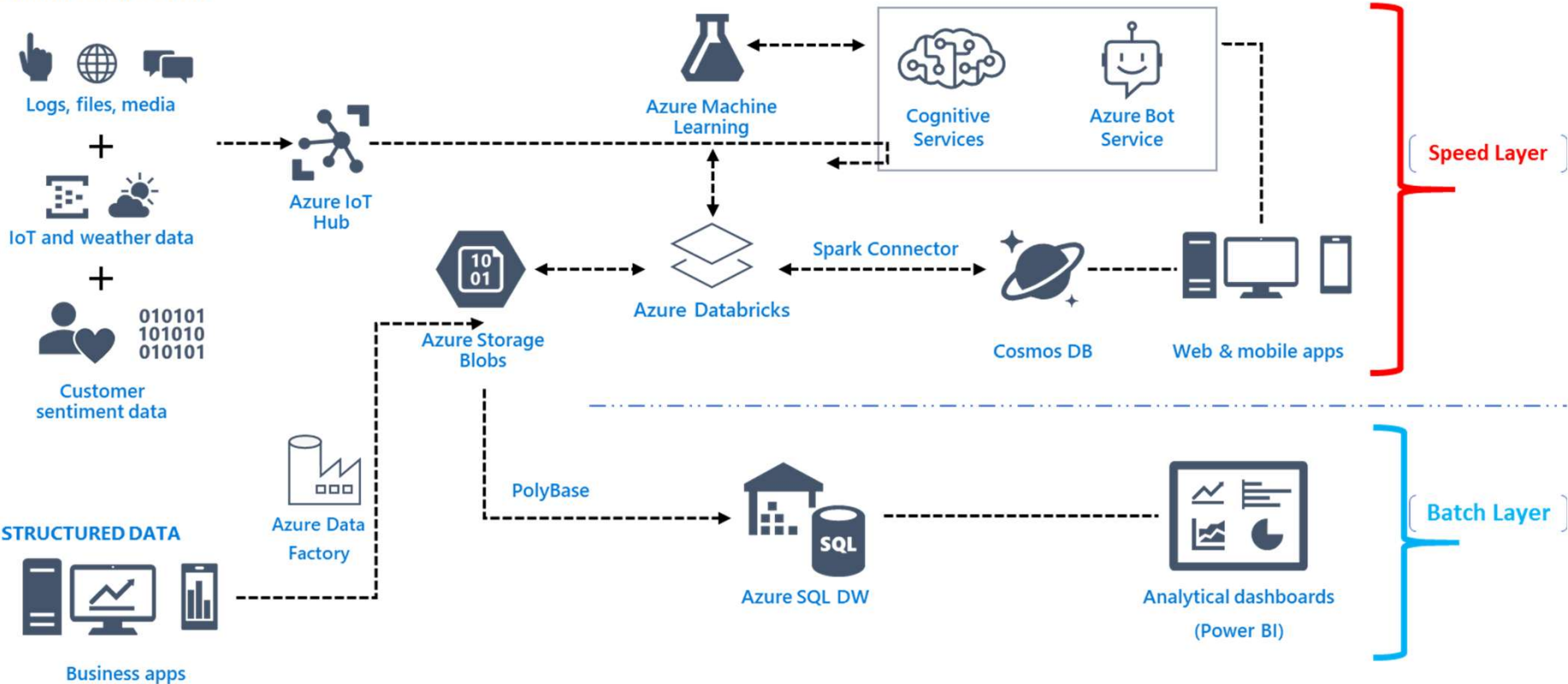


# Example Data Architecture (2)

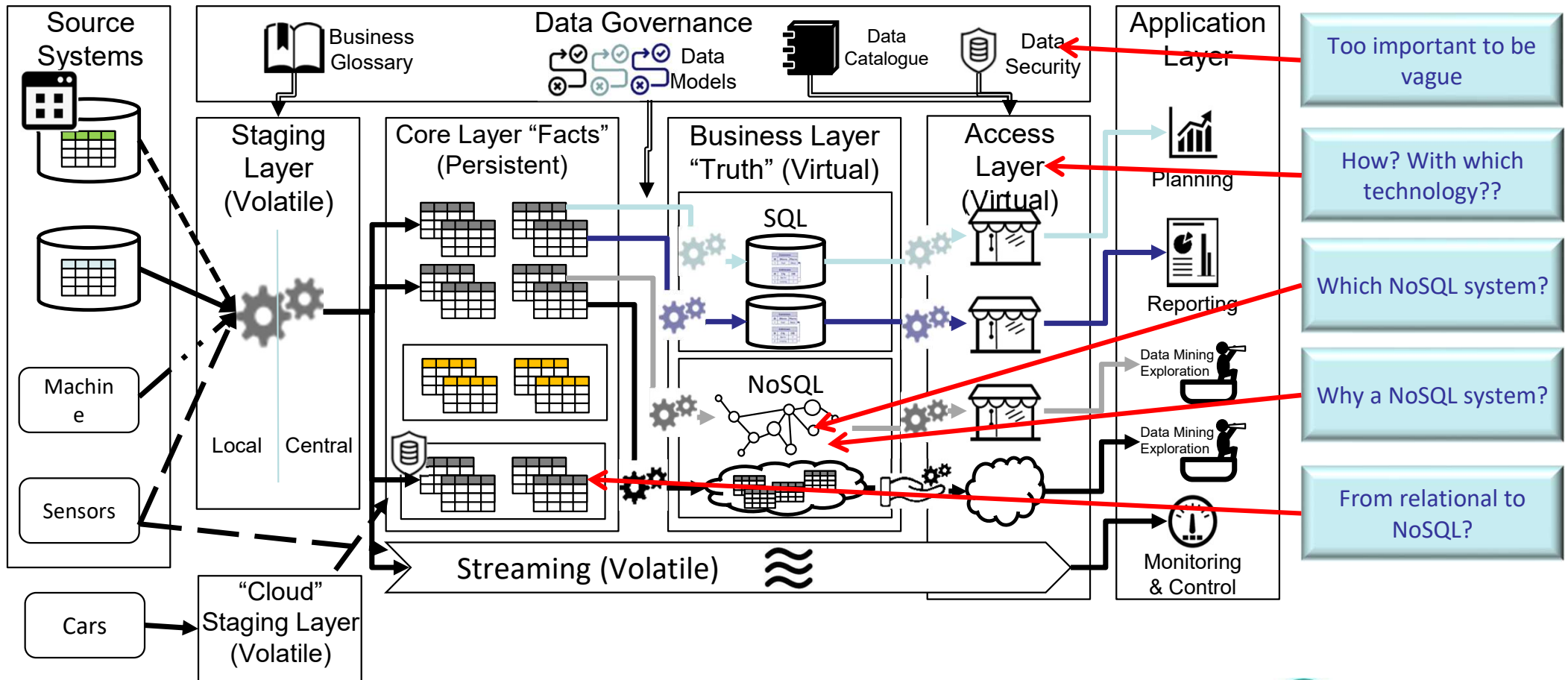


# Example Data Architecture (3)

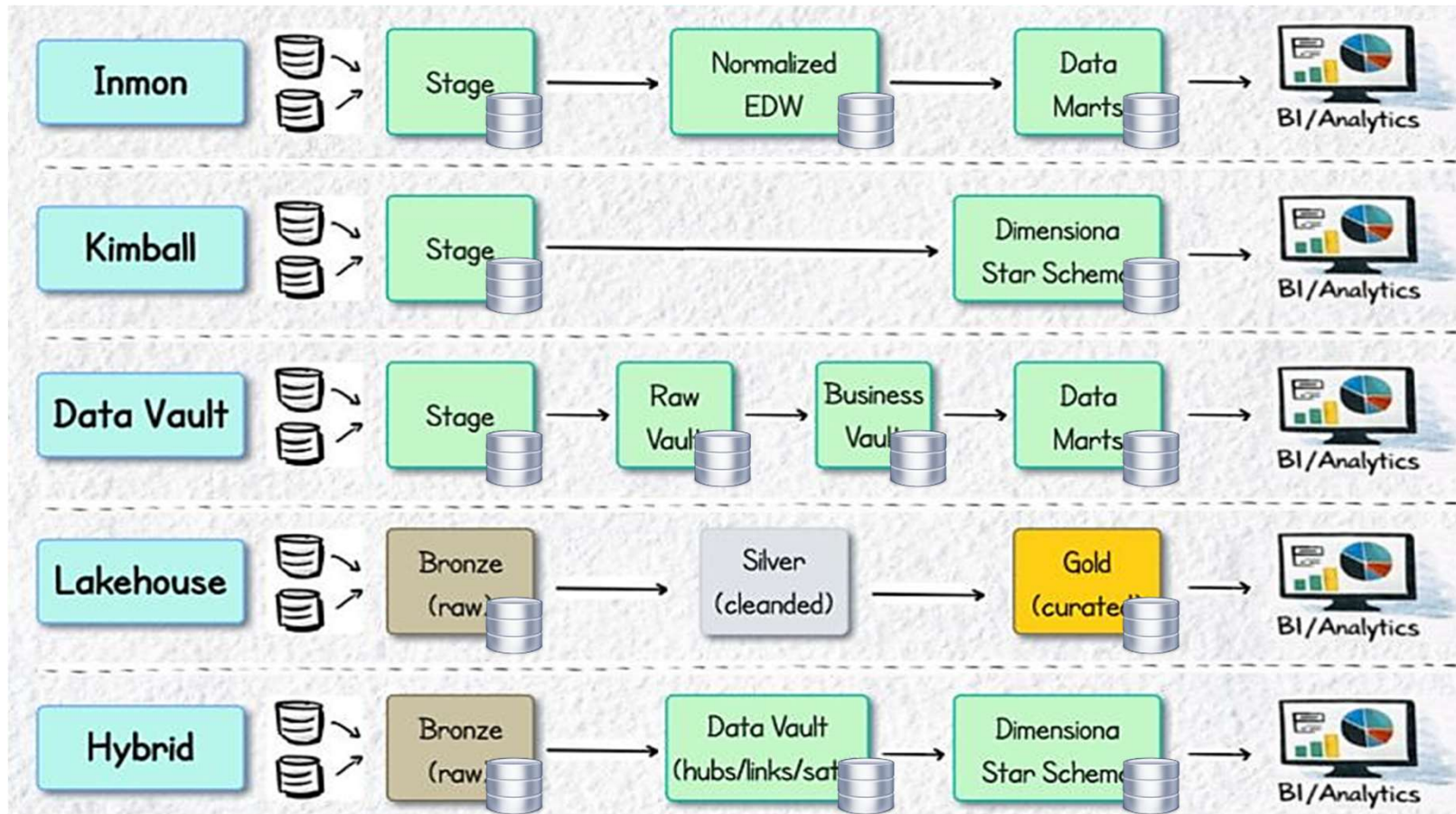
## UNSTRUCTURED DATA



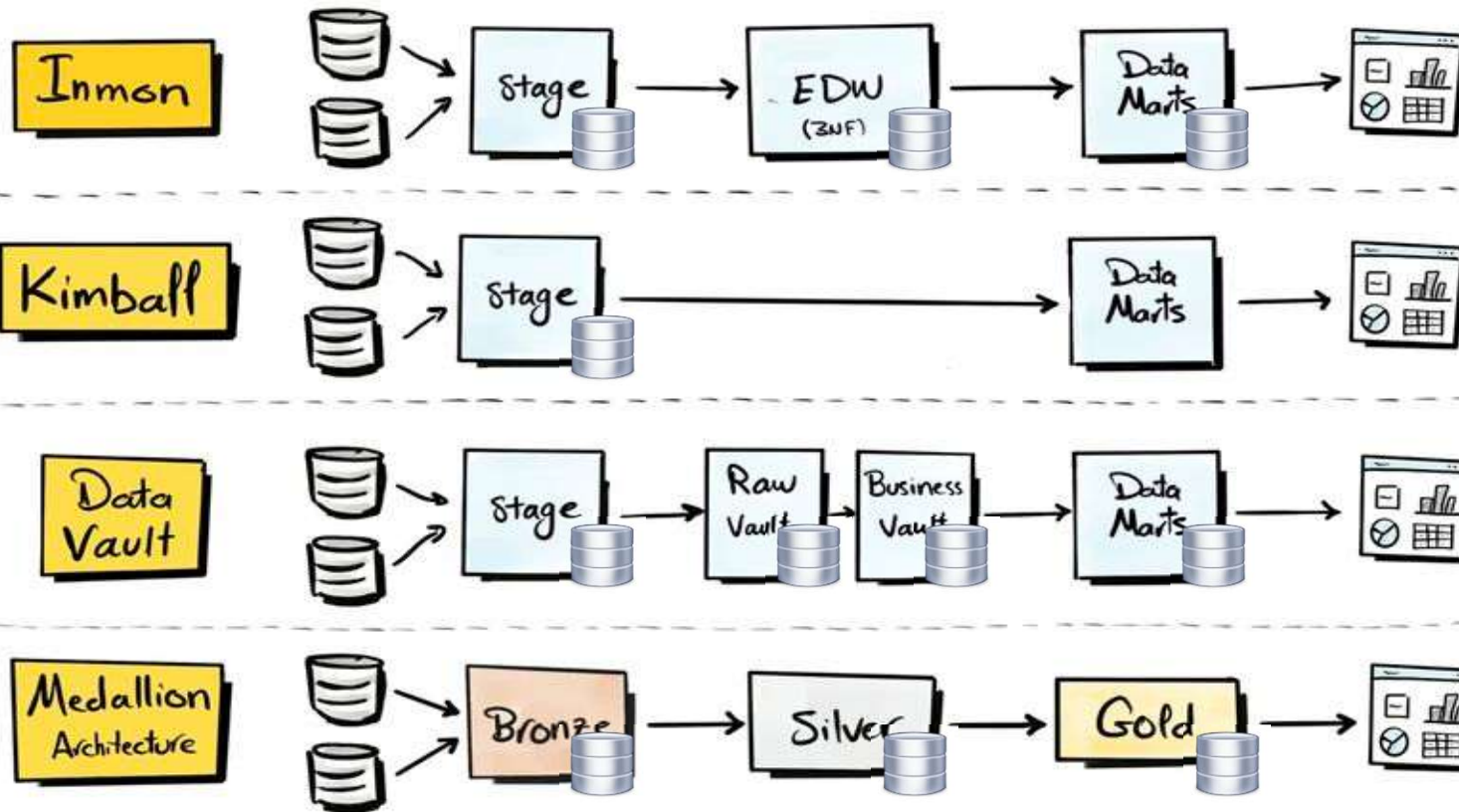
# Example Data Architecture (4)



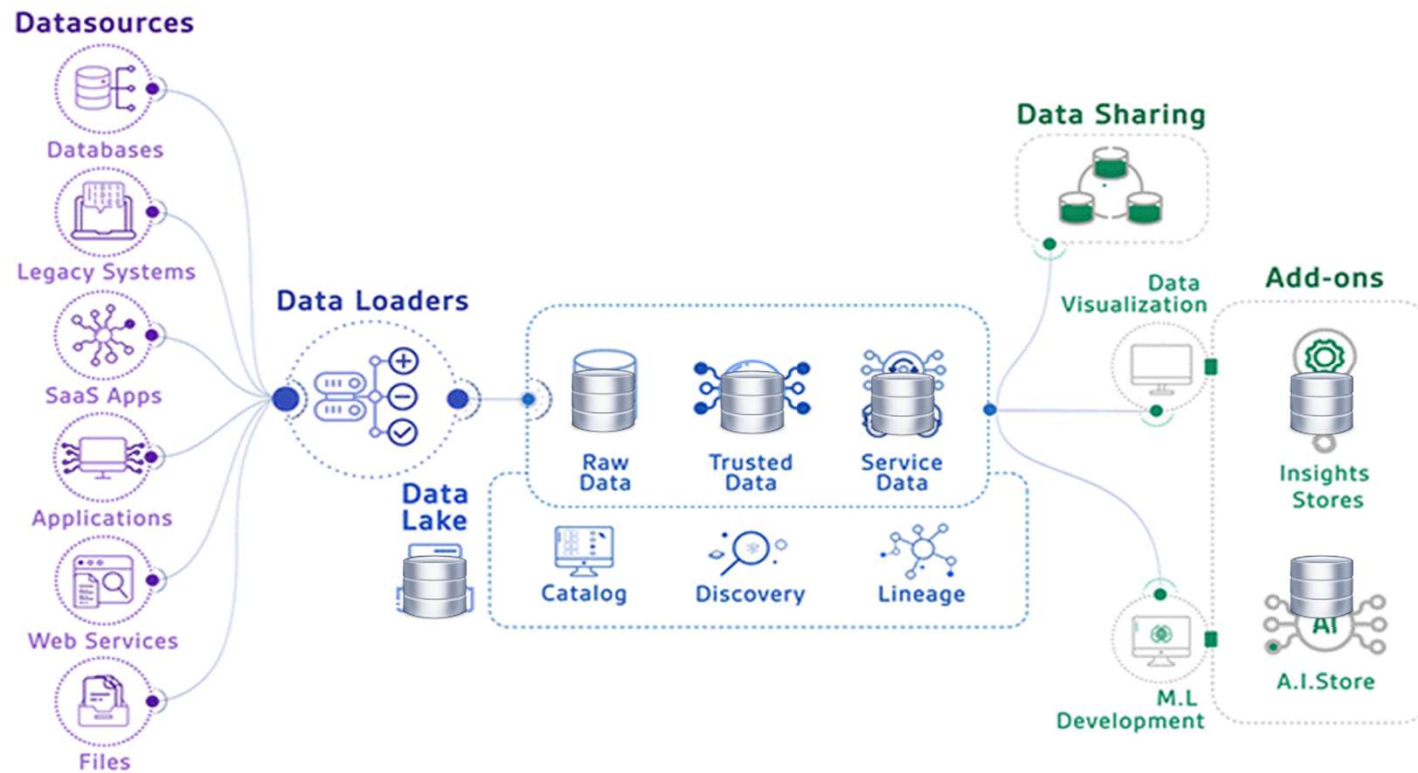
# Data-copy Rich, Partial Data Architectures (1)



# Data-copy Rich, Partial Data Architectures (2)

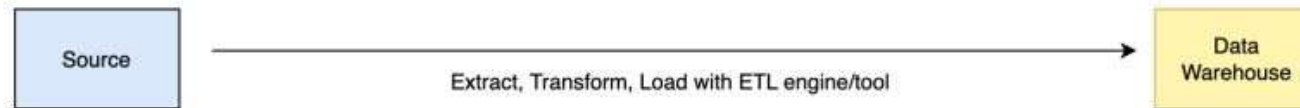


# Data-copy Rich, Partial Data Architectures (3)

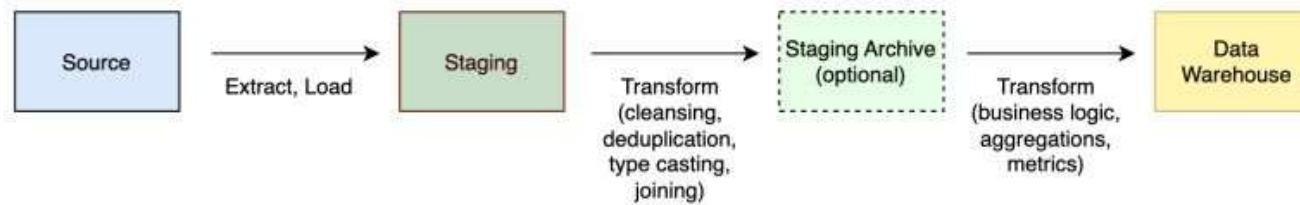


# Data-copy Rich, Partial Data Architectures (4)

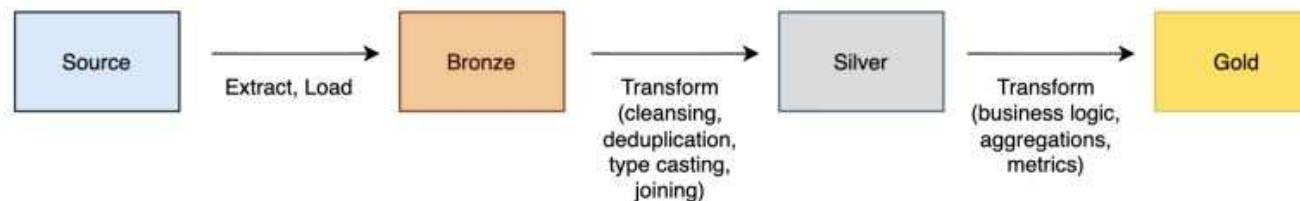
## ETL



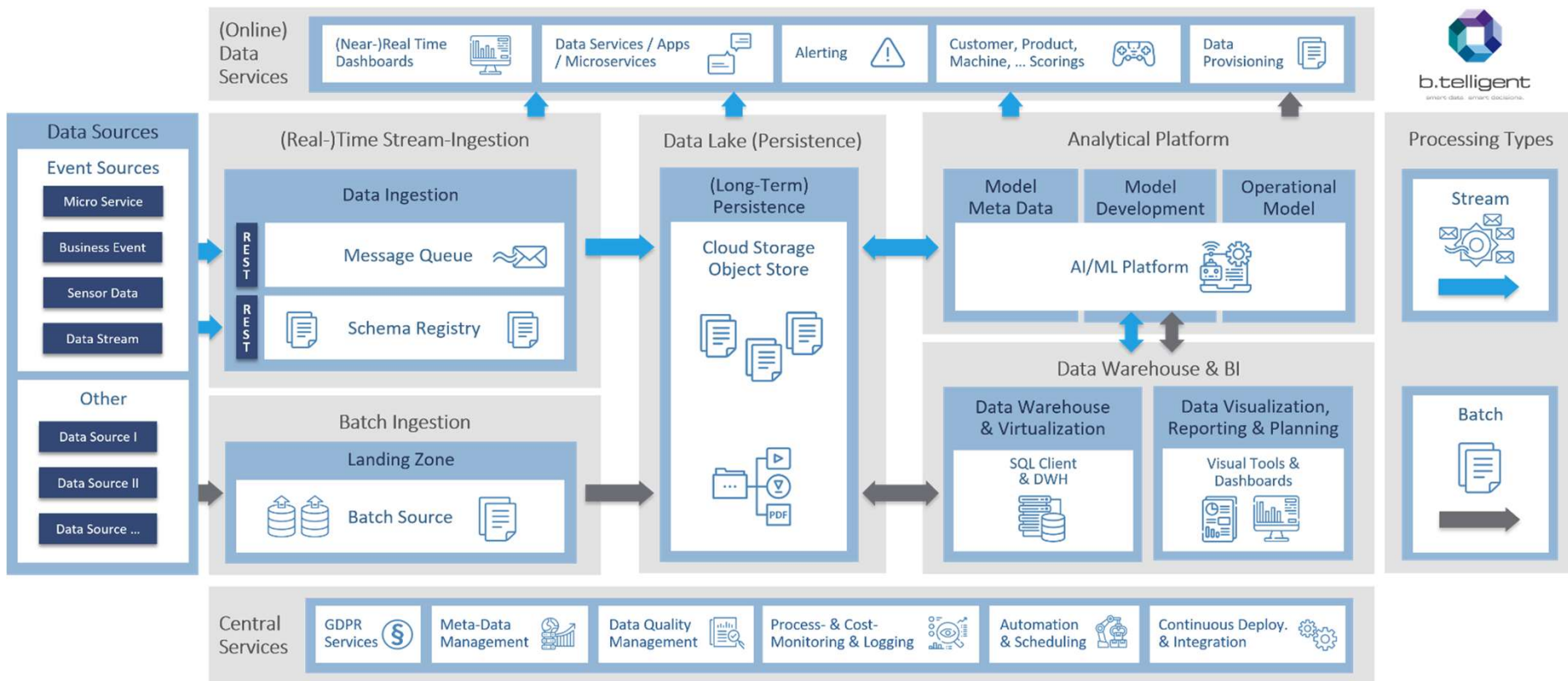
## ELT (Data Warehouse)



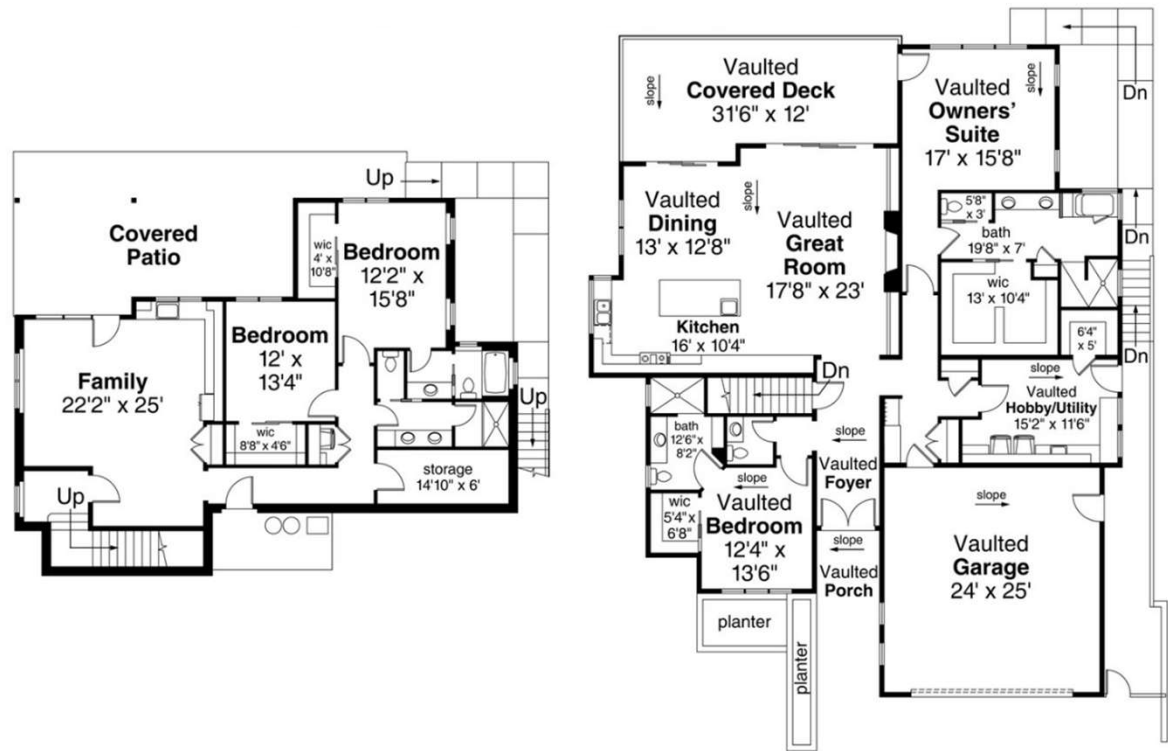
## ELT (Lakehouse)



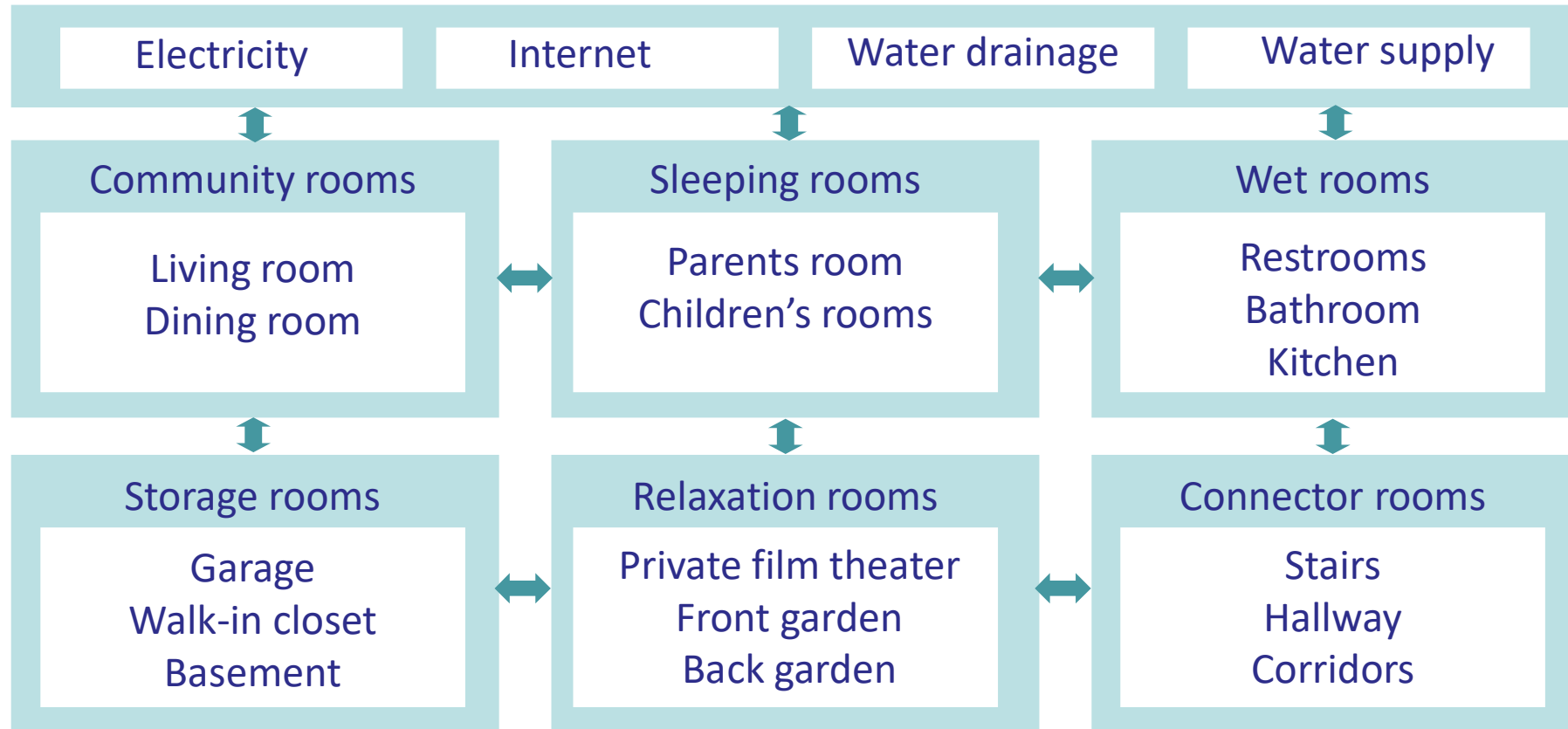
# Data-copy Rich Data Architectures



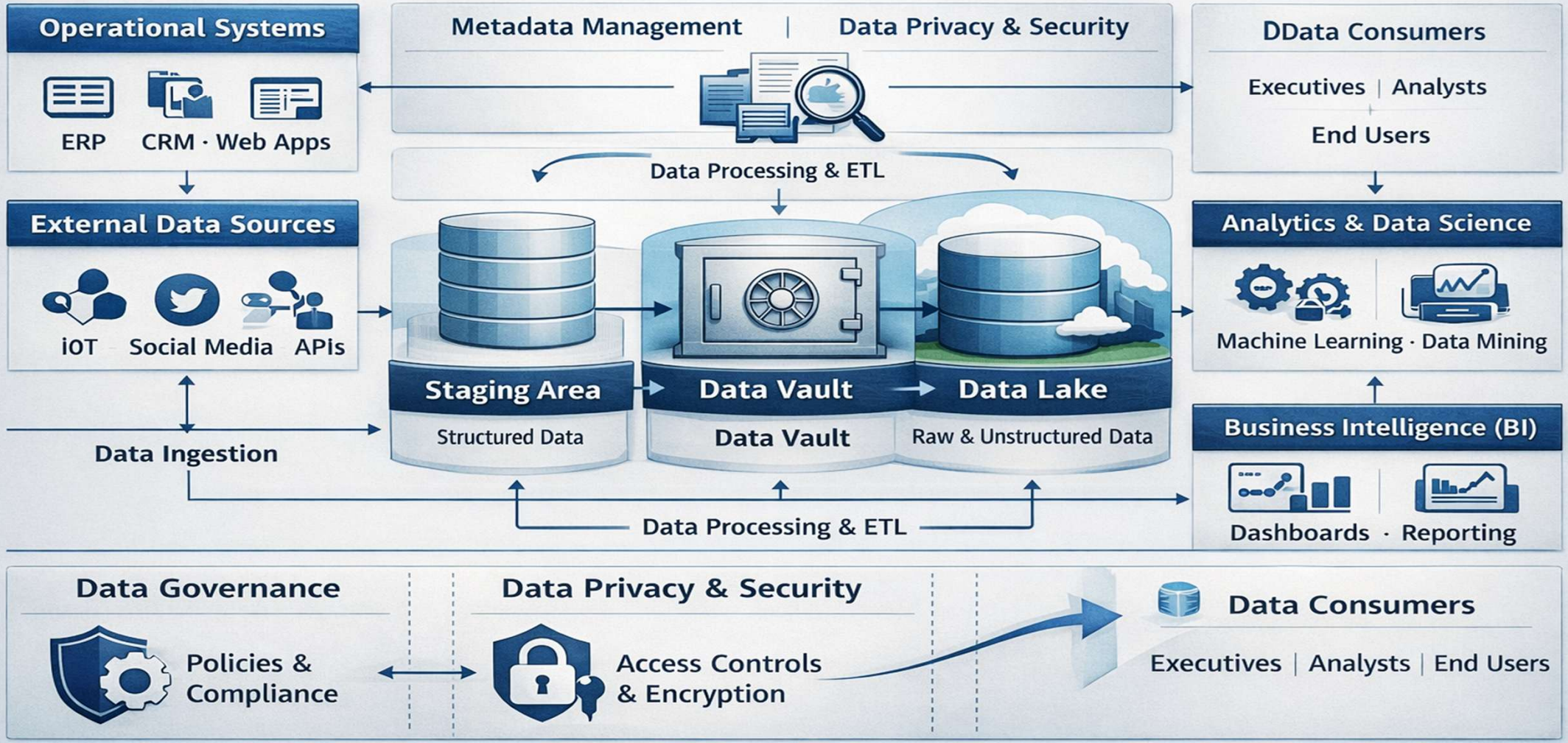
# Architecture of a House



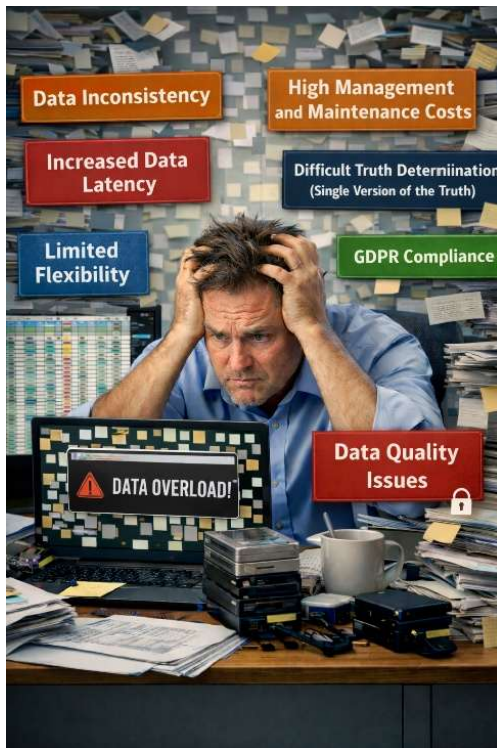
# Architecture of a House (IT Style)



# Data Architecture



# Disadvantages of Data Copying



- Data inconsistency
- Difficult determination of the truth (“single version of the truth”)
- Limited flexibility and agility
- High management, development, and maintenance costs
- More complex compliance with General Data Protection Regulation (GDPR)
- Complex data synchronization issues
- Data quality challenges
- More complex data security
- Outdated data (increased data latency)
- ...

The background features a dark green field with a pattern of glowing green binary code (0s and 1s) arranged in a grid. In the center, there is a semi-transparent, light-colored rectangular button with the word "Copy" written on it in a dark font. The text of the slide is overlaid on this background.

## Patients Suffering From:

**Acute Copying Urge Disorder**

**Pathological Data Accumulation Disorder**

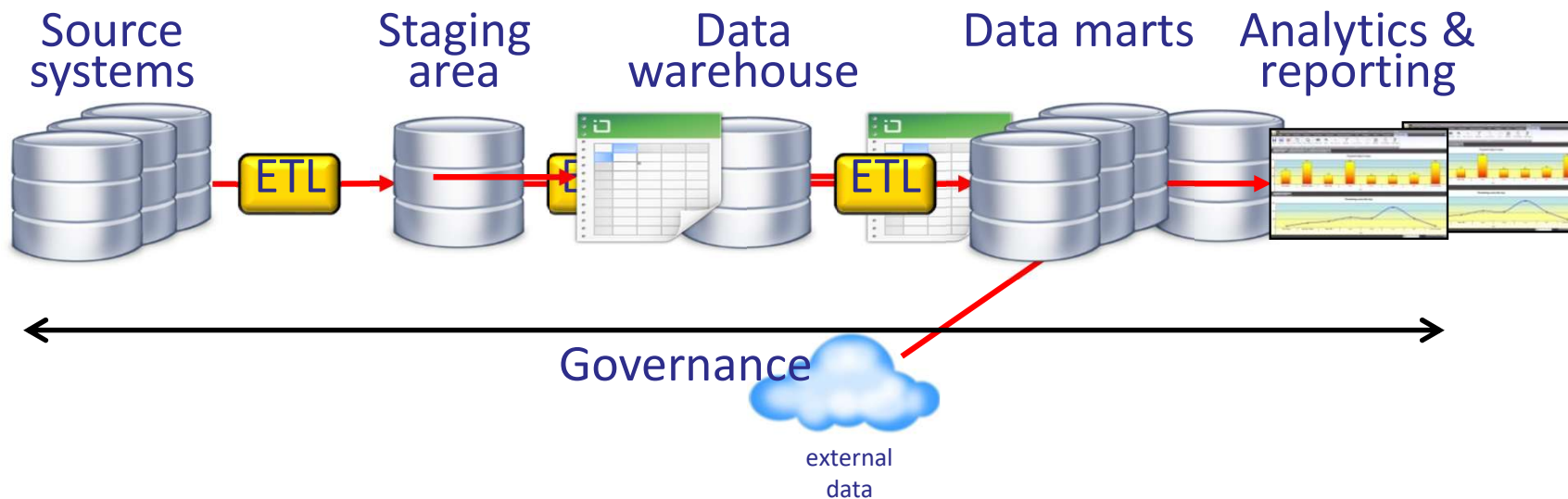
**Excessive Digital Reproduction Condition**

**Type II Redundancy Syndrome**

**Post-Storage Compulsive Replication**

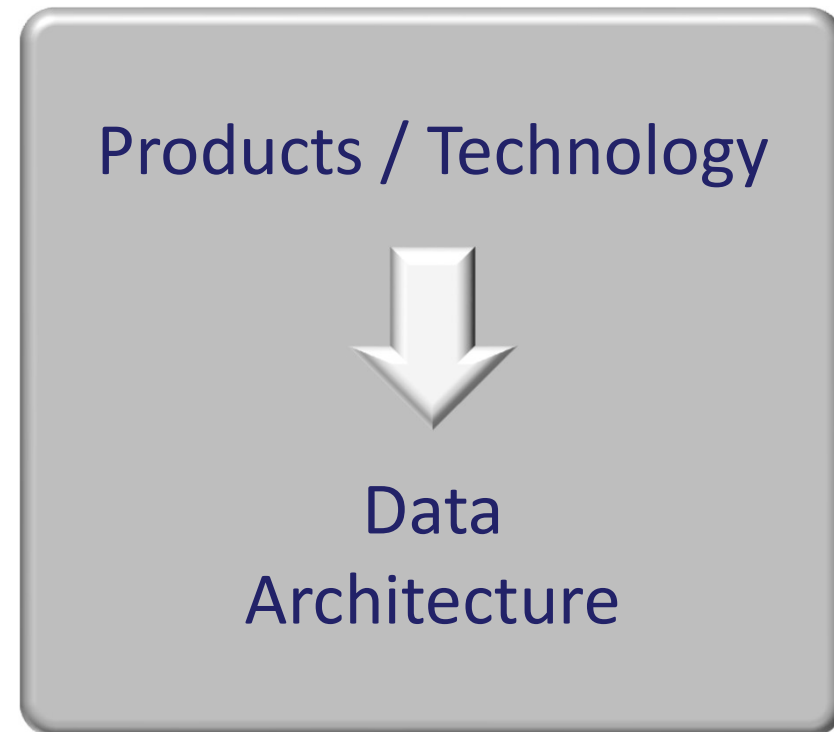
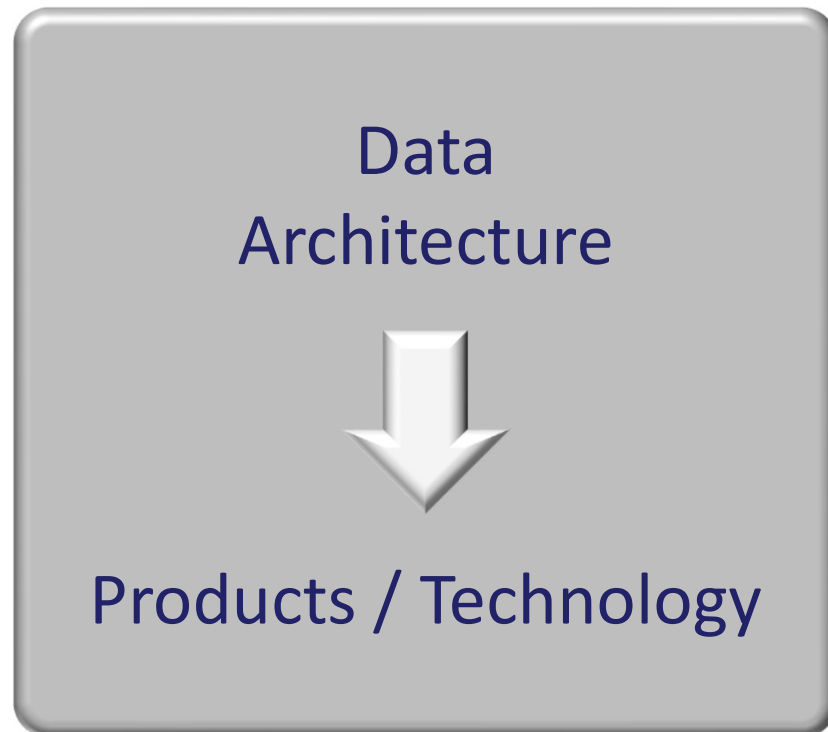
**Duplicitis Chronica**

# Is This Really the Entire Data Architecture?



# What Comes First?

---



# Roadmap for Designing Data Architectures

---

1. Determine business motivations

2. Determine new requirements

3. Analyze the existing environment

4. Define architectural design principles

5. Select a reference data architecture

6. Design the new data architecture

7. Determine the implementation approach

8. Select new products and technologies

9. Introduce the data architecture within the organization



**Part 4:**  
**Analyze New Technologies for  
Data Storage, Processing, and Analytics**



## Part 4.1: Data Storage

# Trends in Database Market

---

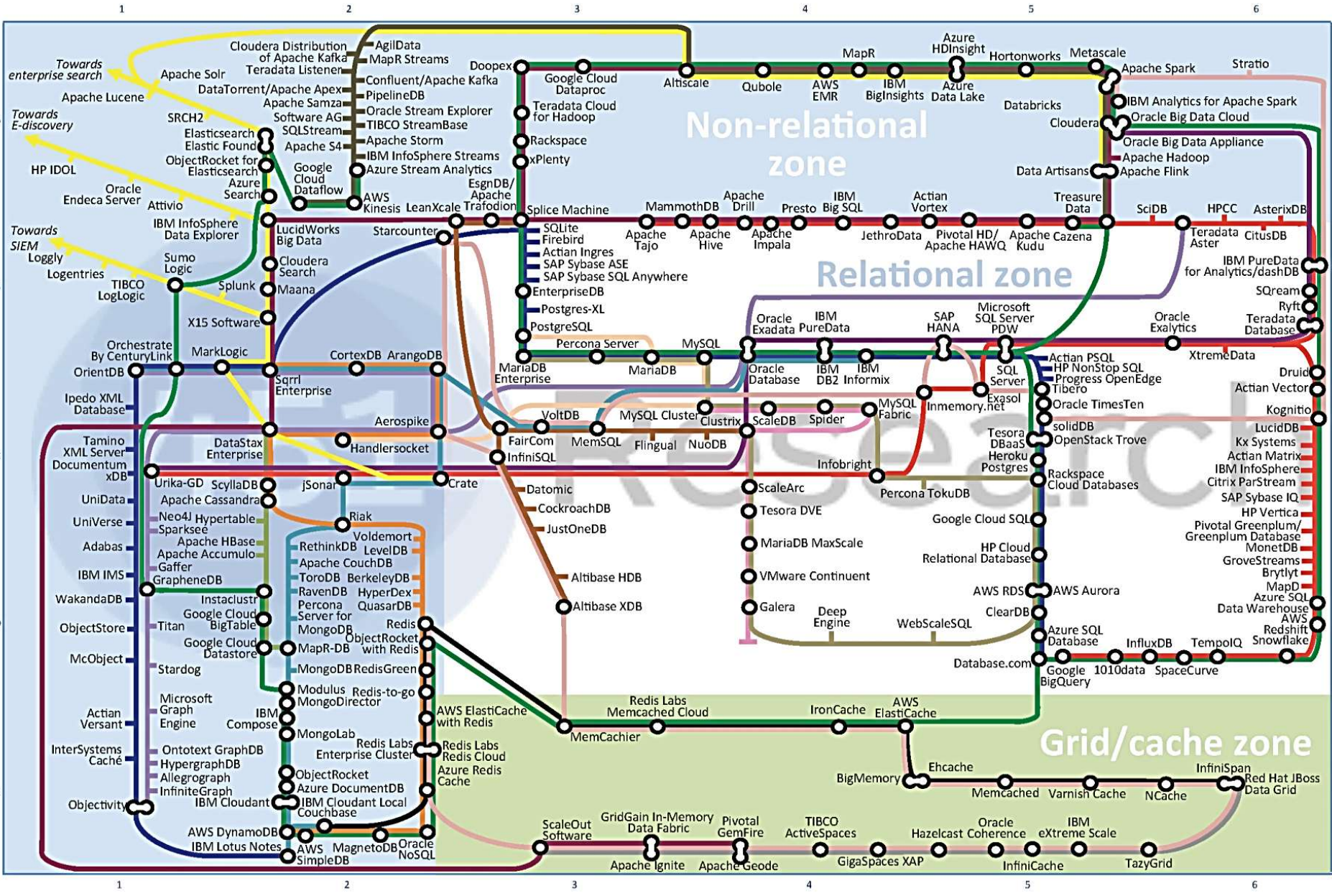


- More data, more queries, more transactions, more concurrent users, more complex queries, ...
- Coming and going of products
- Less standards
  - Less portability
- No interchangeable products
- Specialization of products
  - Limited use cases

# Data Platforms Map

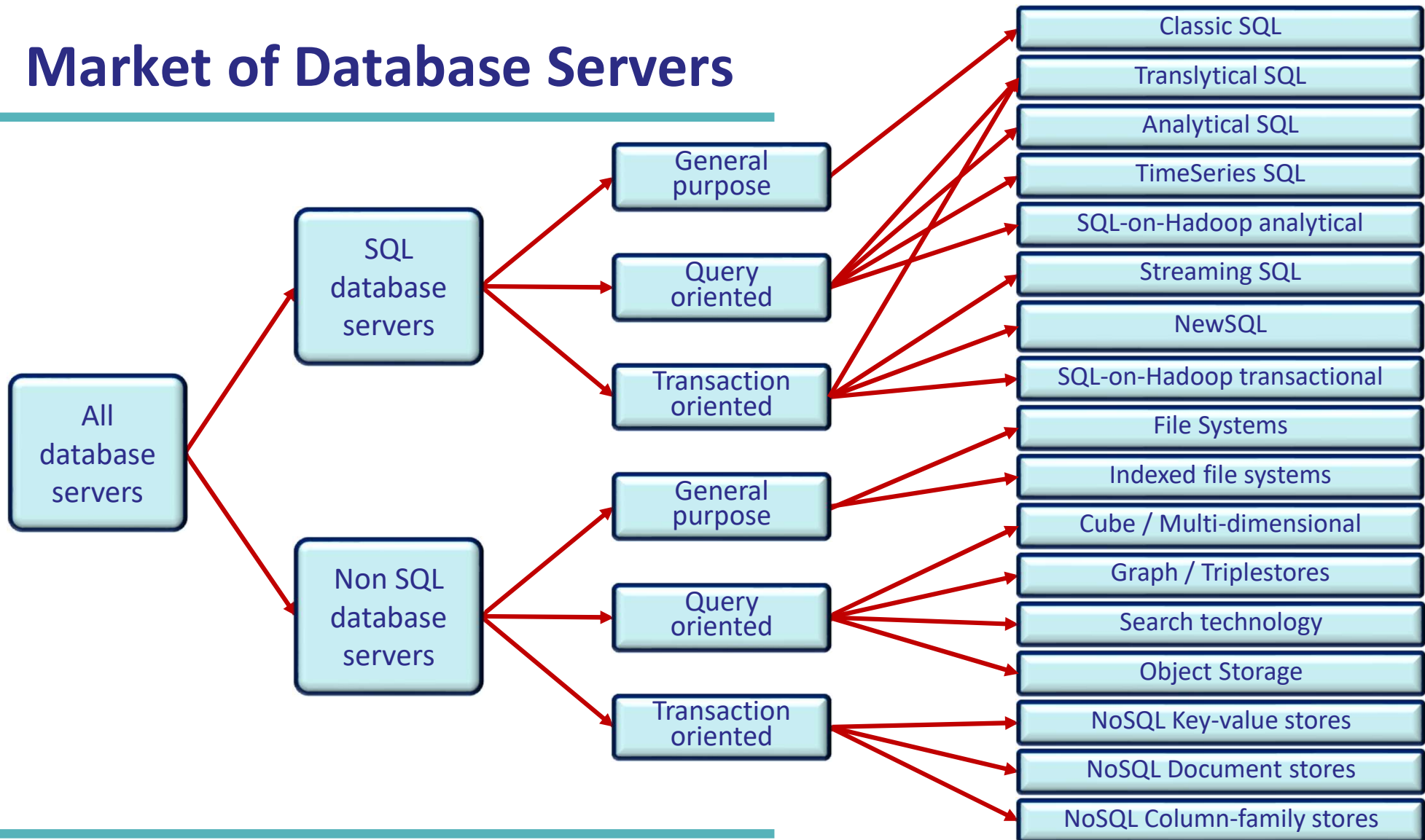
January 2016

- Key:**
- General purpose
  - Specialist analytic
  - -as-a-Service
  - BigTables
  - Graph
  - Document
  - Key value stores
  - Key value direct access
  - Hadoop
  - MySQL ecosystem
  - Advanced clustering/sharding
  - New SQL databases
  - Data caching
  - Data grid
  - Search
  - Appliances
  - In-memory
  - Stream processing



<https://451research.com/state-of-the-database-landscape>

# Market of Database Servers



# Specialization of Cars



# Database Technology has Changed

---

Generic Database  
Solutions



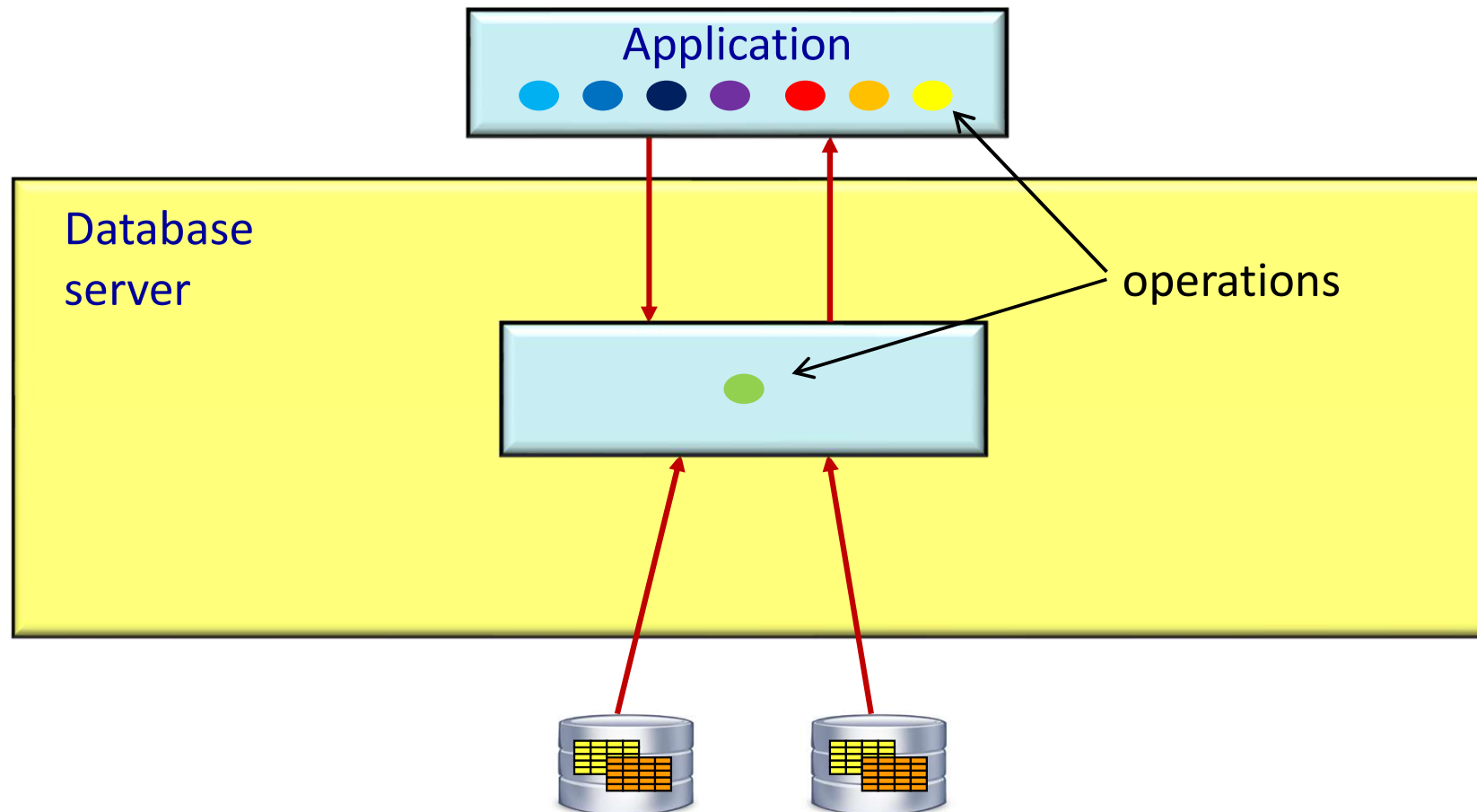
Specialized Database  
Solutions

Use Case  
Agnostic

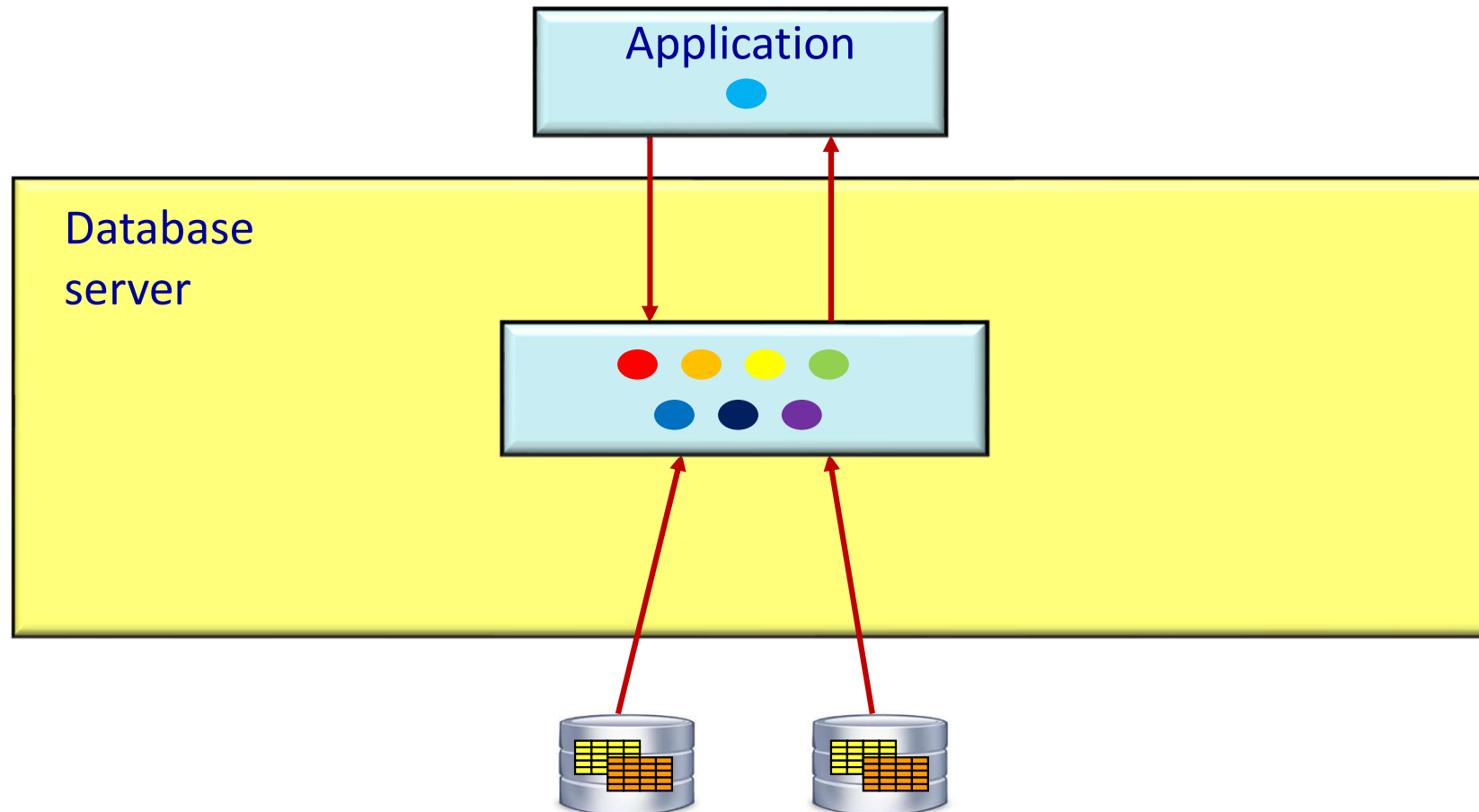


Use Case  
Specific

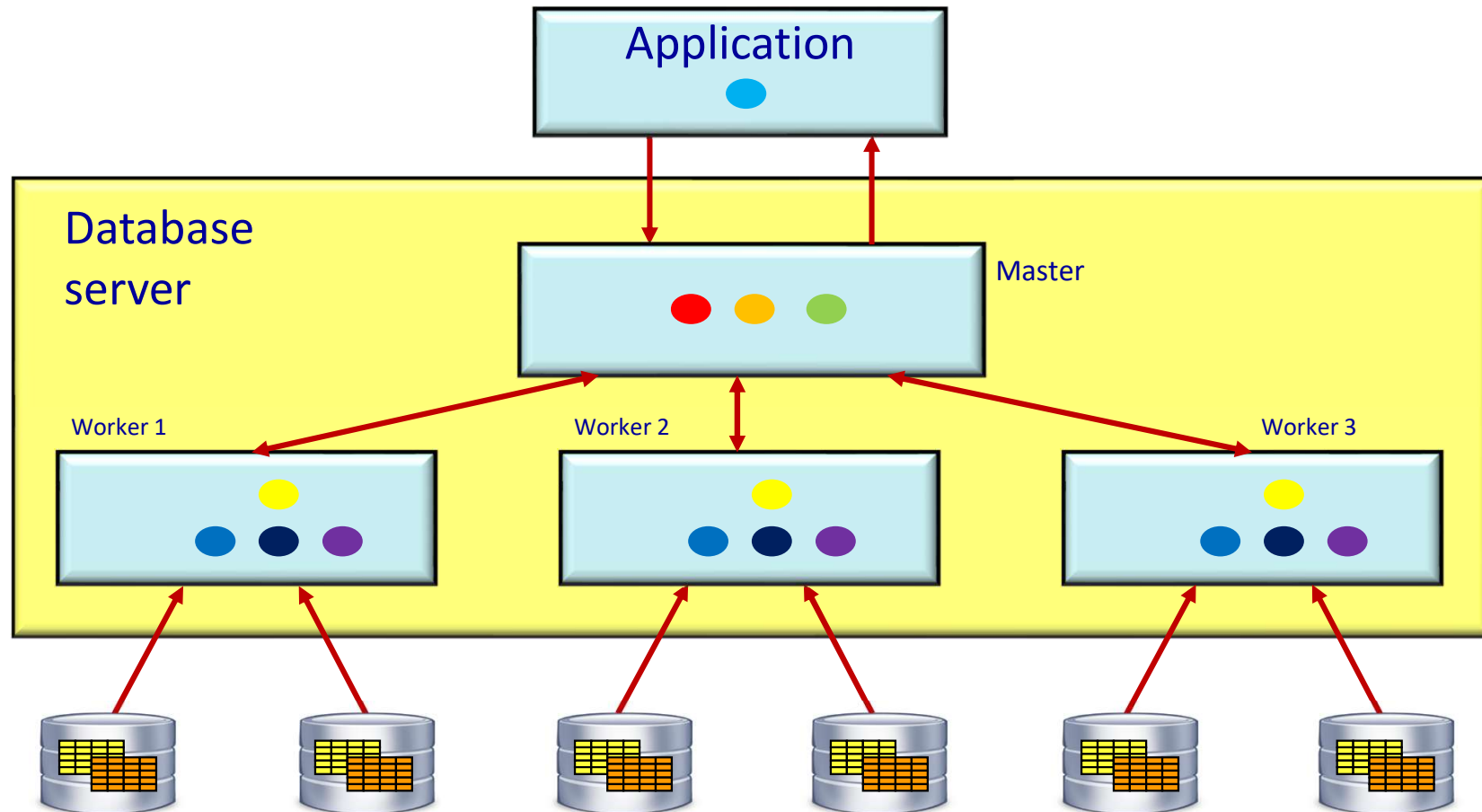
# Application-based Analytics



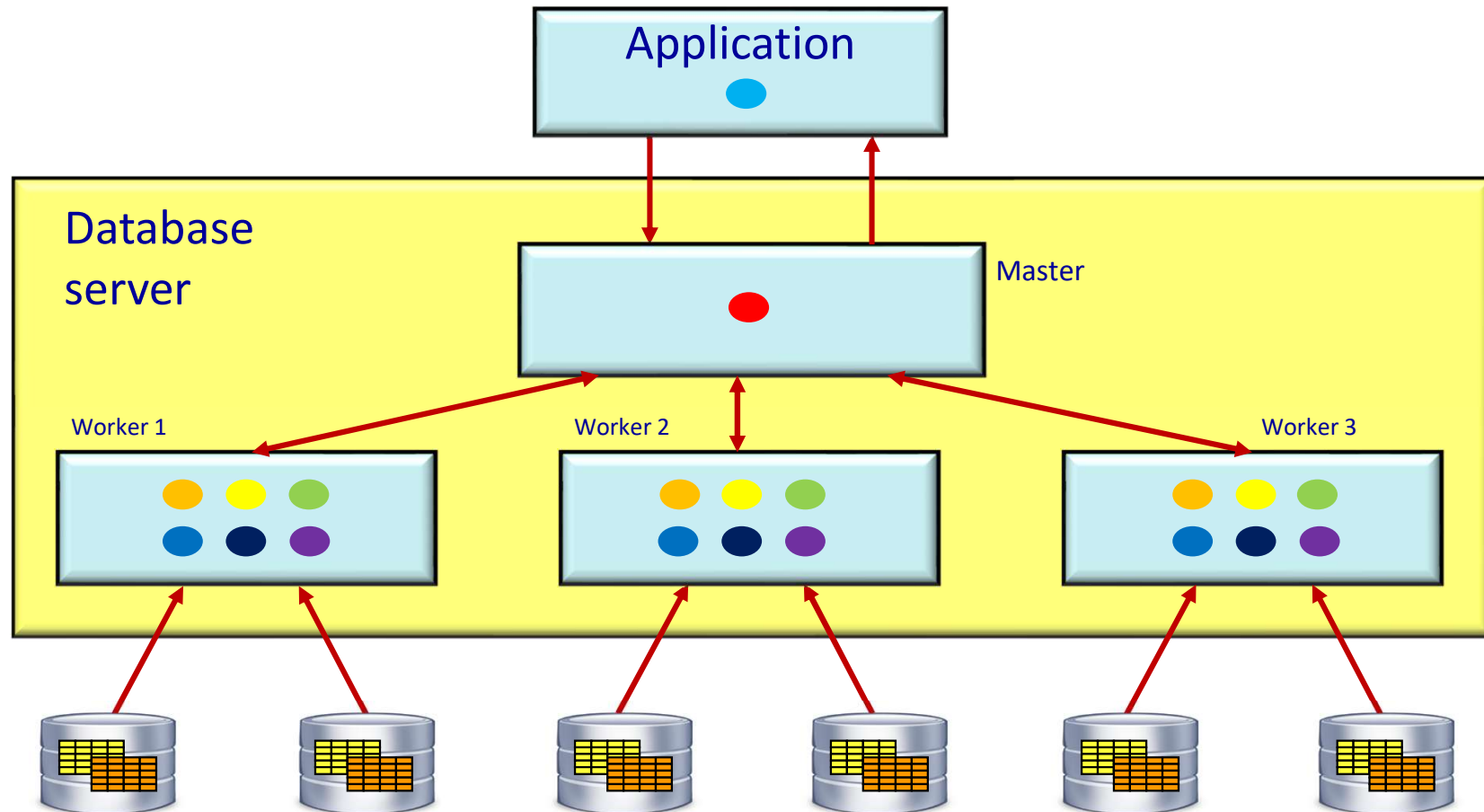
# In-Database Analytics



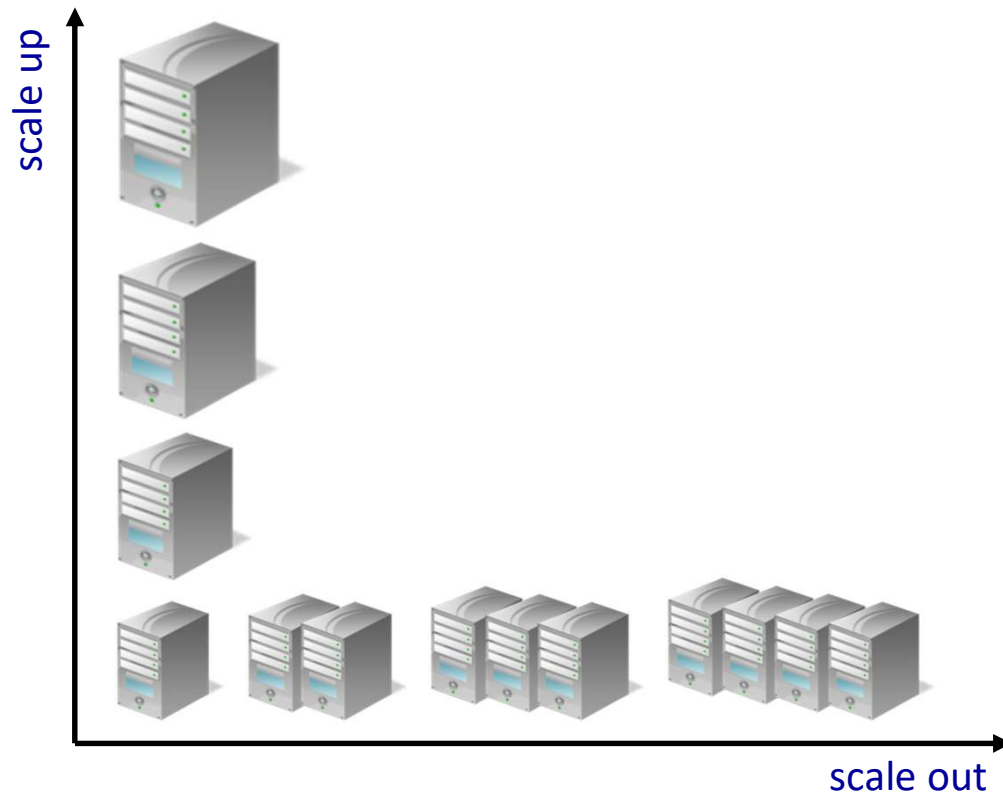
# Partial Parallel Analytics



# Full Parallel Analytics

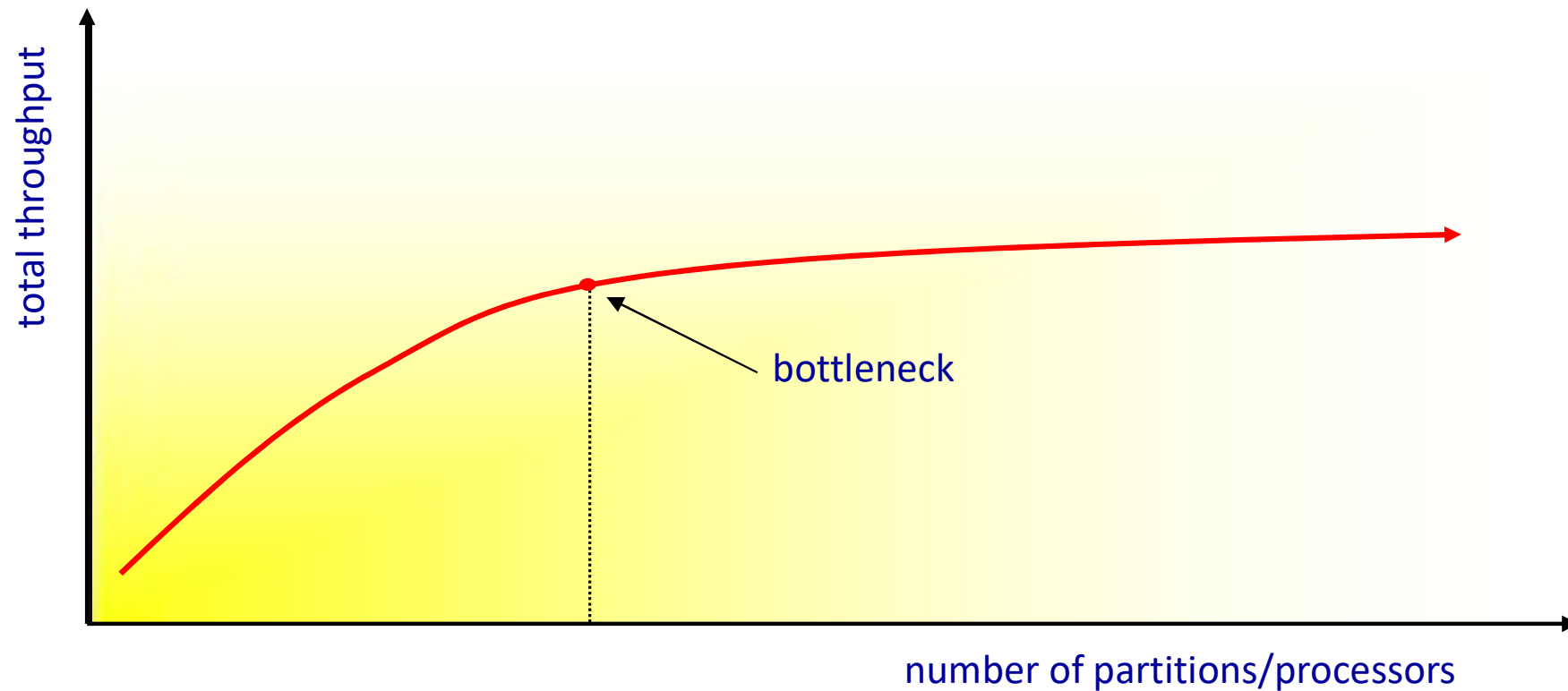


# Scale Up versus Scale Out



- Scale up (vertical scaling) means adding more resources to one node in a system
- Scale out (horizontal scaling) means adding more nodes to a system
  - Continuous availability/redundancy
  - Cost/performance flexibility
  - Contiguous upgrades
  - Geographical distribution

# Effect of Partitions on Query Response



# NoSQL Database Servers

---

APACHE  
**HBASE**

 **mongoDB**

 **redis**

  
**Cassandra**

**Google**  
**BigTable**

 **Azure Cosmos DB**

  
**CouchDB**  
*relax*

  
**LevelDB**

  
**RAVENDB**

 **riak**

**UpscaleDB**

  
**Couchbase**

 **HYPERTABLE**

 **emCached**

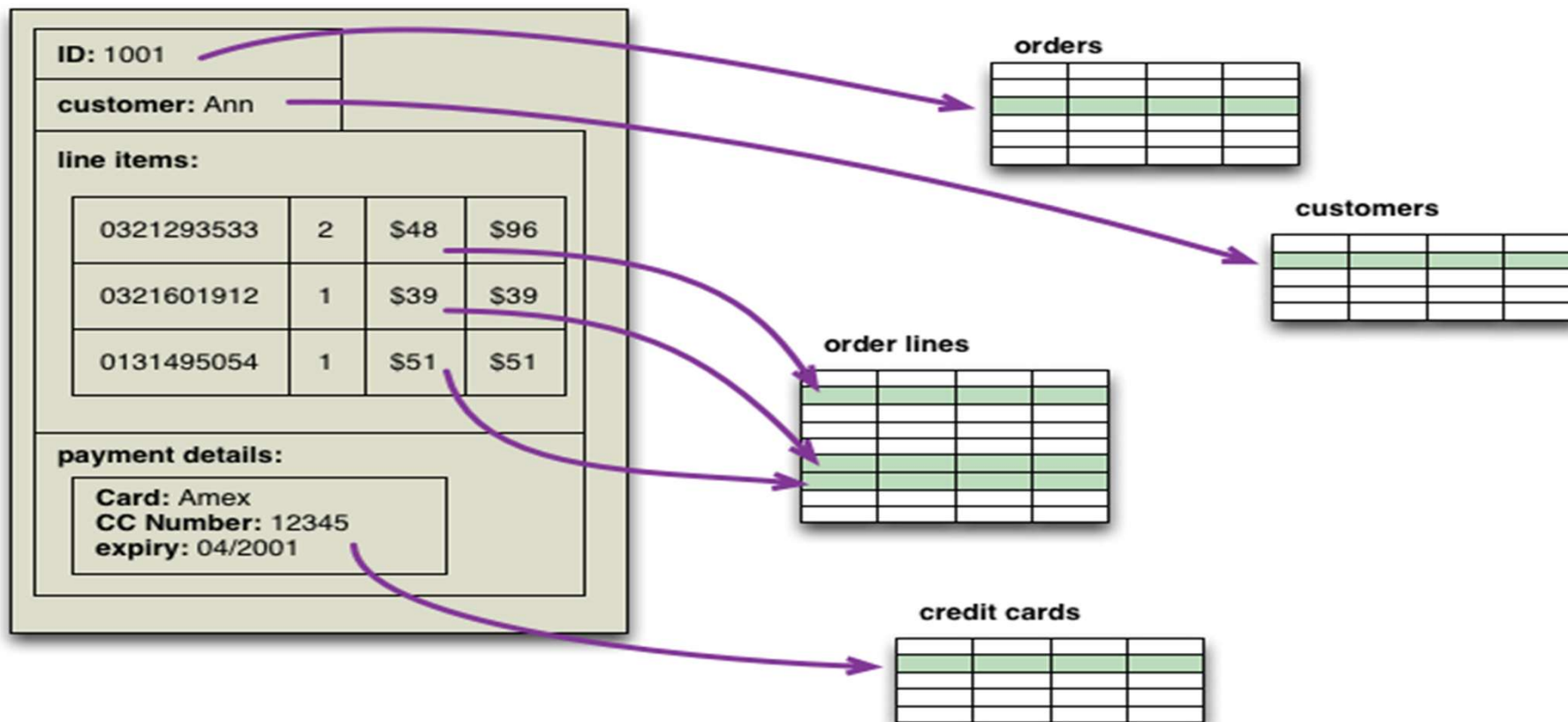
# Tricks to Improve Performance

---



- Aggregate data model
  - To remove the impedance mismatch
- Design architecture to scale-out
  - Sharding
- Reduce functionality (security, query power, data integrity, ...)
- Lower consistency
- Give developers full control over internal processing
- “Push down” complex operations

# NoSQL: Aggregate Data Model



# Typical NoSQL Use Cases

APACHE  
**HBASE**



- Transactional
- Big transactional workload
- Single record/document transactions
- Massive data ingestion
- Simple reporting – point queries
- Dynamic data structures
- Complex data structures
- “Narrow” data model

**Couchbase**

Google  
**BigTable**

Neo4j  
the graph database



**RAVENDB**

LevelDB

HAYSTACK DB



**hazelcast**

**riak**

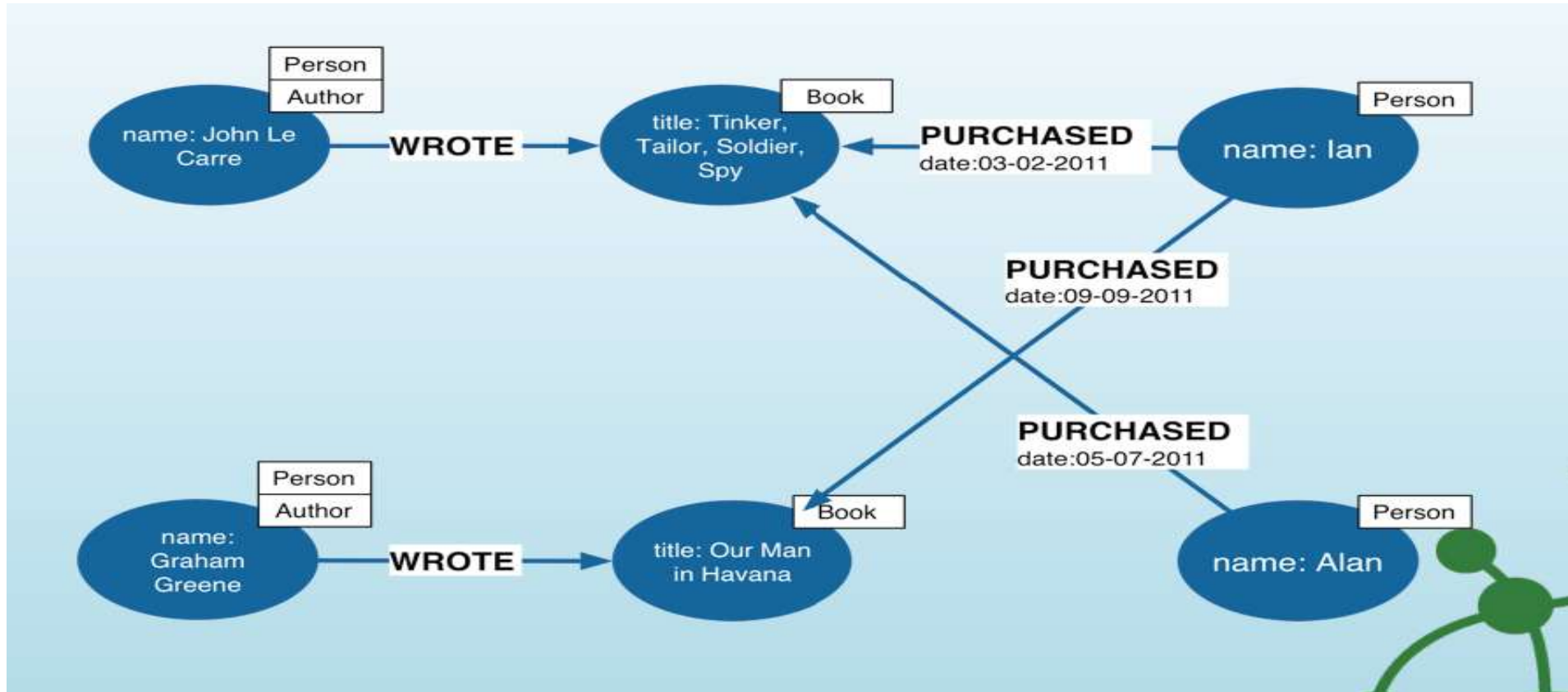


**CouchDB**  
relax

**emCached**

**HYPERTABLE**

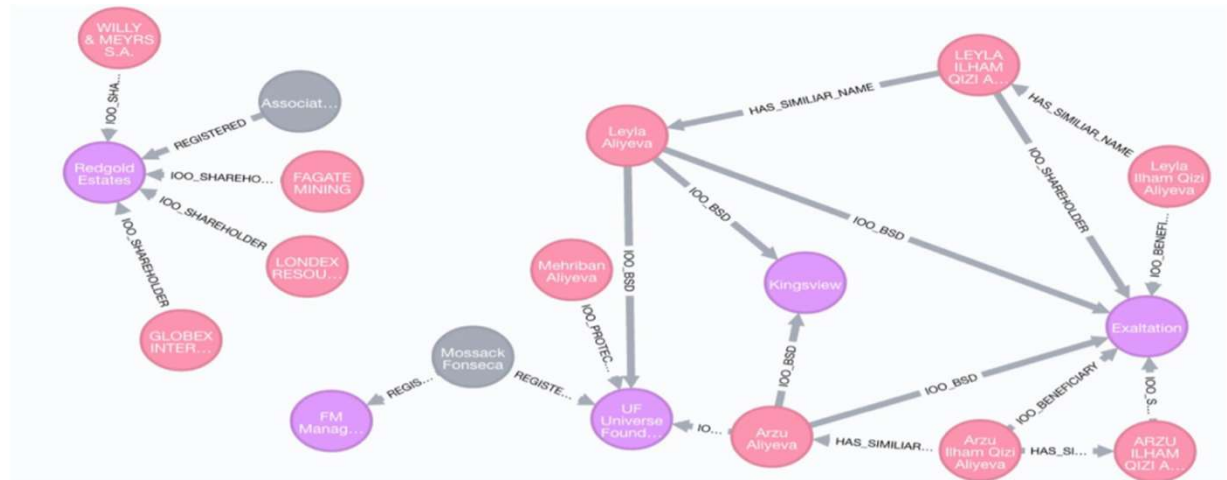
# Graph Database Servers



Source: M. Hunger, Neo Technology, Data Modeling with Neo4j, Aug 2013

# Application Areas Graph Databases

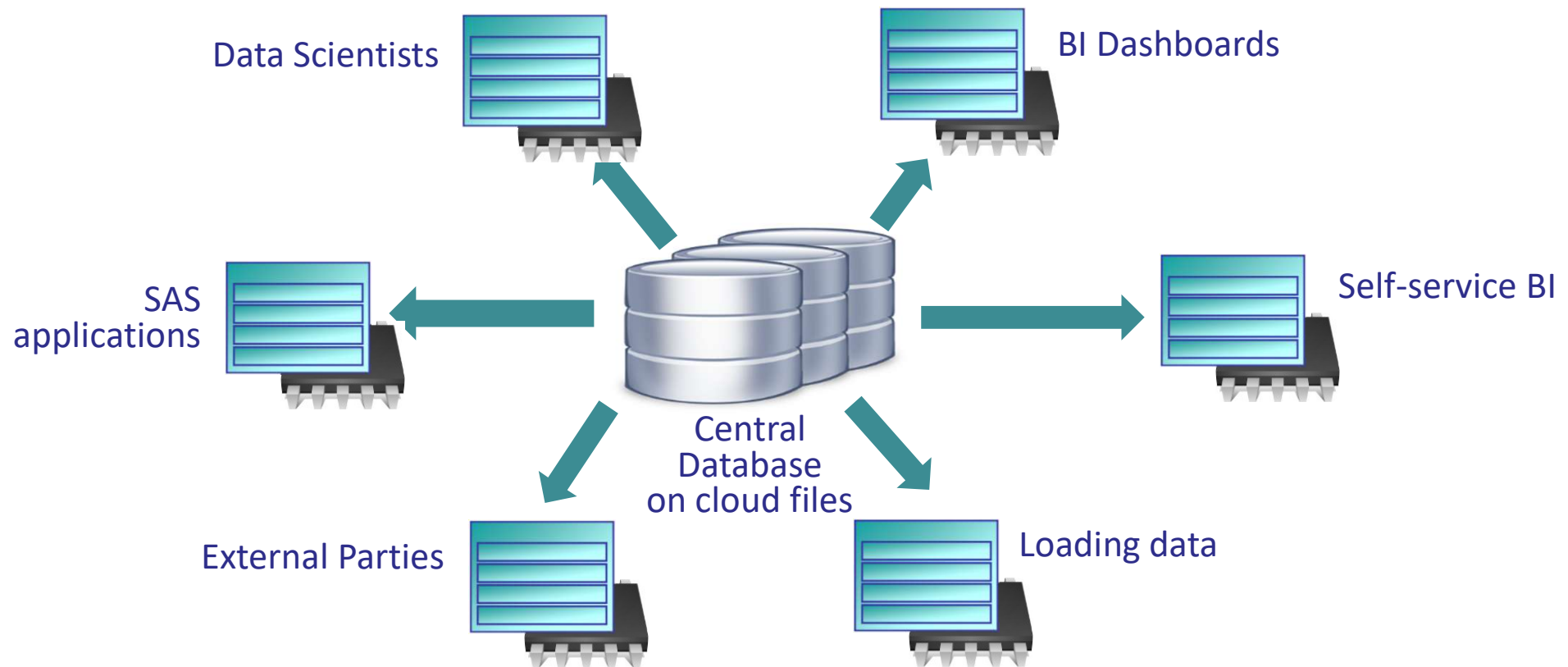
- Social network analysis
- Network impact analysis
- Optimal route determination
- Internet retail recommendations
- Logistics
- Fraud analysis
- Securities and debts
- “Panama papers”
- And many more



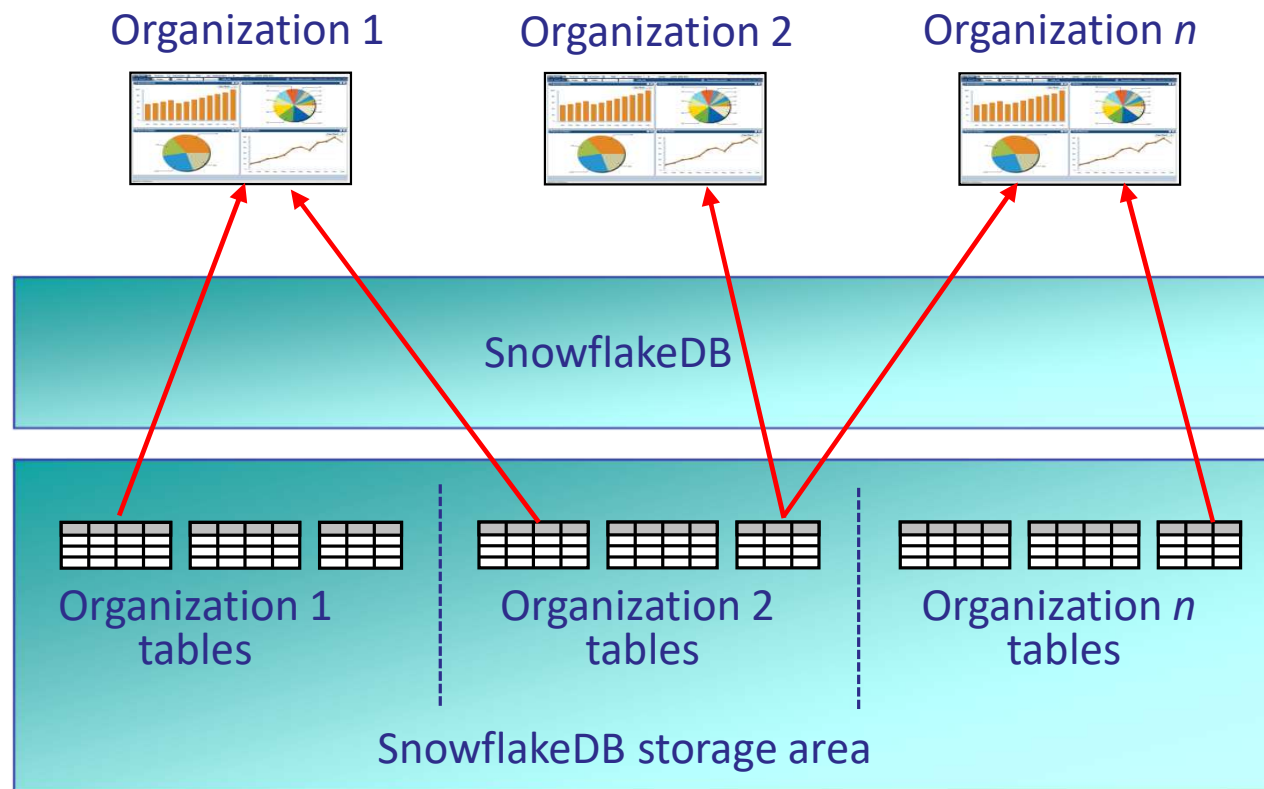
# Analytical SQL Database Servers

Examples	
1010Data	Microsoft Azure Synapse
Amazon Redshift and Athena	OmniSci (MapD)
Apache HAWQ	Oracle Database In-Memory
BlazingSQL	SAP HANA
CitusDB (PostgreSQL)	SAP Sybase IQ
ClickHouse SQL	SnowflakeDB
Databricks Delta Lake	Splice Machine
Edge Intelligence	SQream
Exasol	Starburst (Trino formerly Presto)
Google BigQuery	Teradata Vantage
Greenplum	XTremeData dbX
IBM DB2 Warehouse on Cloud	Several SQL-on-Hadoop engines
Ignite InfoBright DB	Vertica
Kinetica	And many others ...
Kognitio WX2	

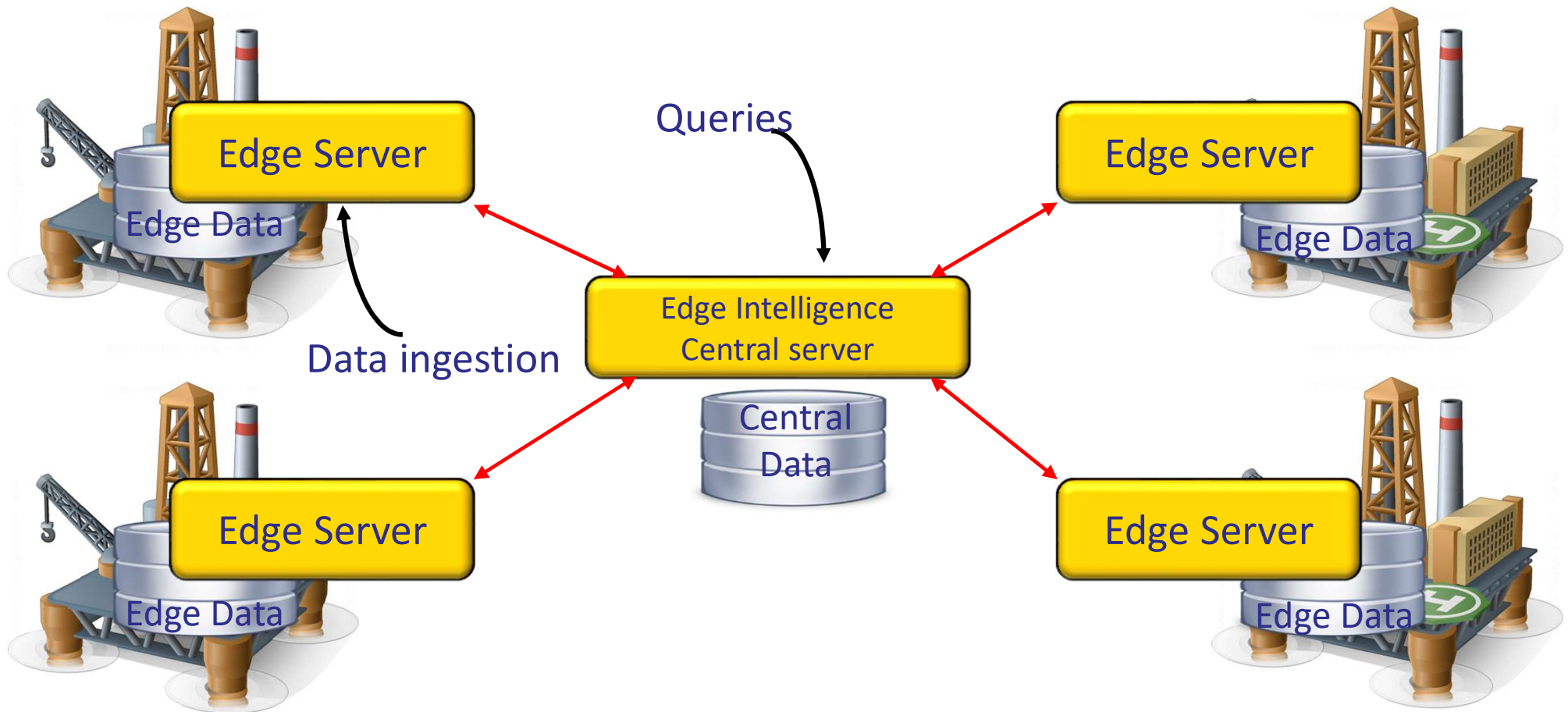
# Example 1: Snowflake



# Example 1: Snowflake



# Example 2: Edge Intelligence



# Example 2: Edge Intelligence

Non-replicated Data at Edge 1

Store_id	Customer	Datetime	...
1	Metheny	2017-11-01 12:00:08	...
1	Johnson	2017-11-01 12:10:18	...
1	Young	2017-11-01 12:12:33	...
1	Morrison	2017-11-01 12:50:09	...
1	Harris	2017-11-01 12:55:45	...

Replicated Data at Edge 1

Customer	...
Metheny	...
Johnson	...
Young	...
Morrison	...
Harris	...
Mitchell	...
Stills	...
Dylan	...
Brown	...

Non-replicated Data at Edge 2

Store_id	Customer	Datetime	...
2	Metheny	2017-11-01 12:01:32	...
2	Mitchell	2017-11-01 12:05:42	...
2	Stills	2017-11-01 12:11:39	...
2	Dylan	2017-11-01 12:12:30	...
2	Brown	2017-11-01 12:40:19	...

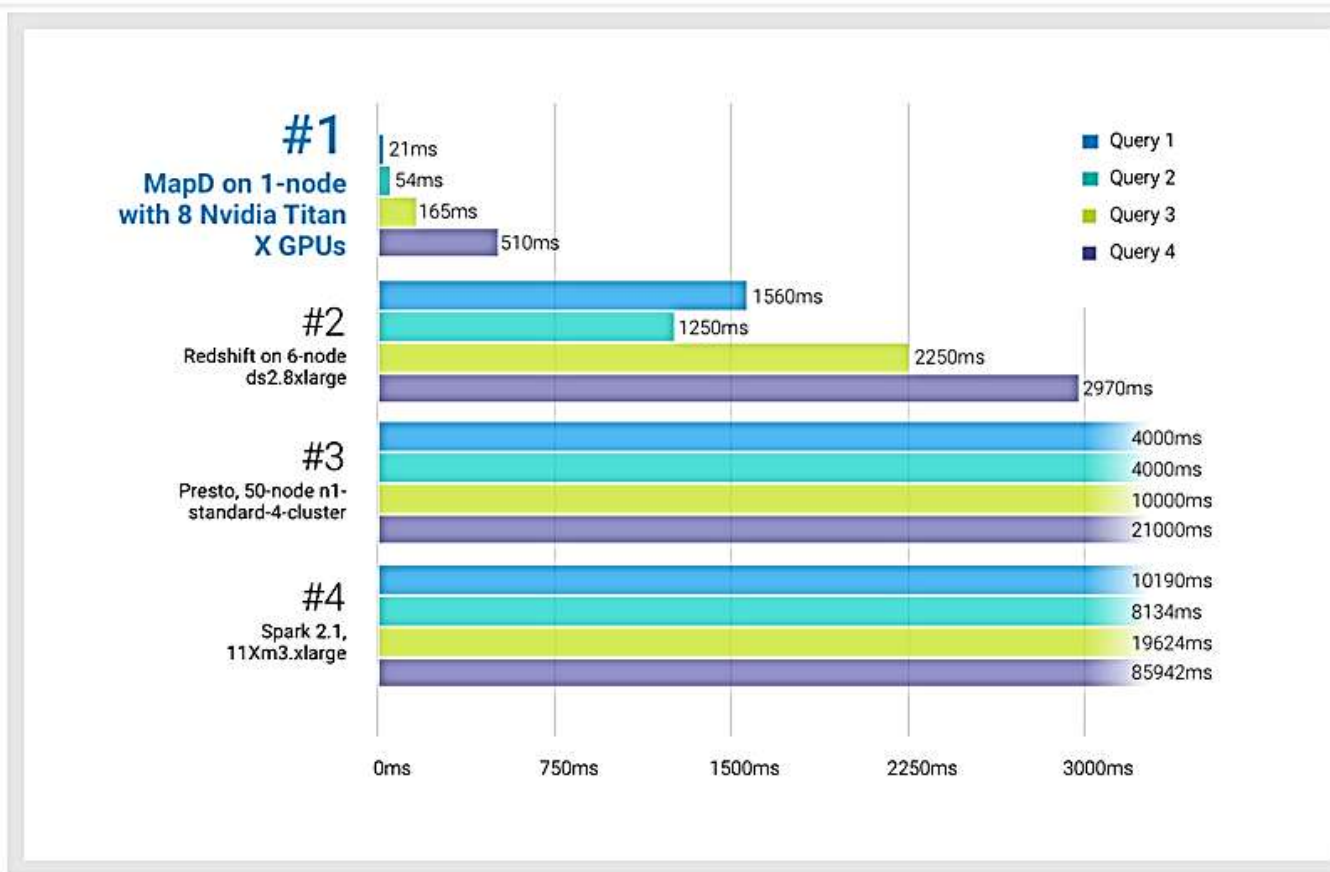
Replicated Data at Edge 2

Customer	...
Metheny	...
Johnson	...
Young	...
Morrison	...
Harris	...
Mitchell	...
Stills	...
Dylan	...
Brown	...

## **NVIDIA TITAN V: GPU With More Than 5,000 Cores**

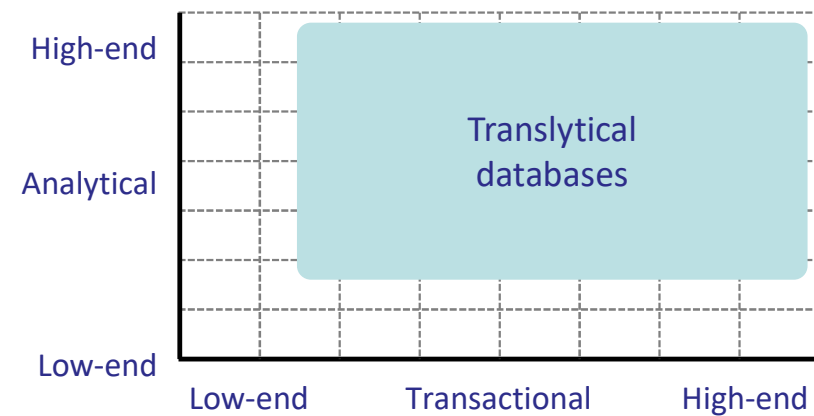
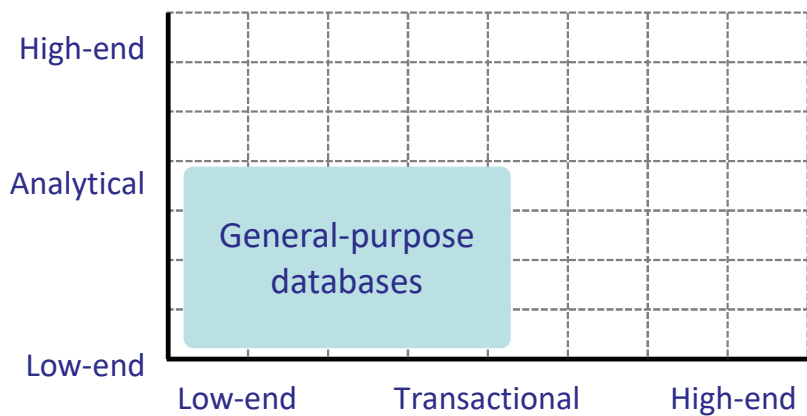
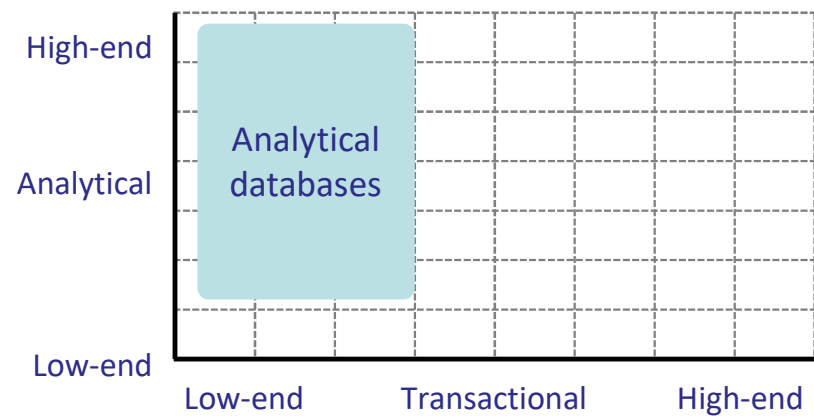
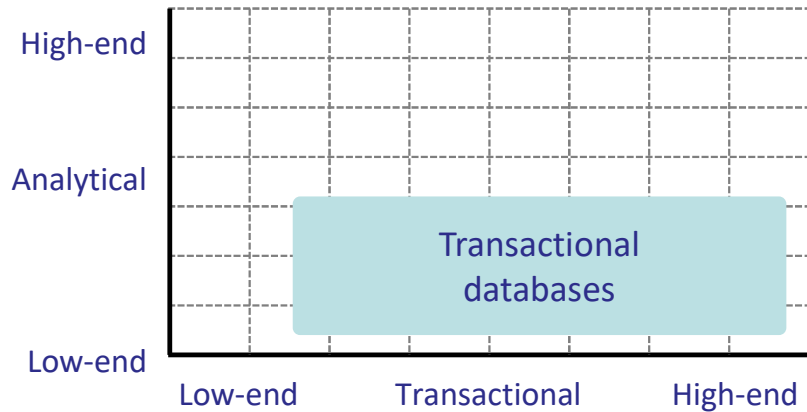


# Comparison of GPU-based Analytical SQL Databases



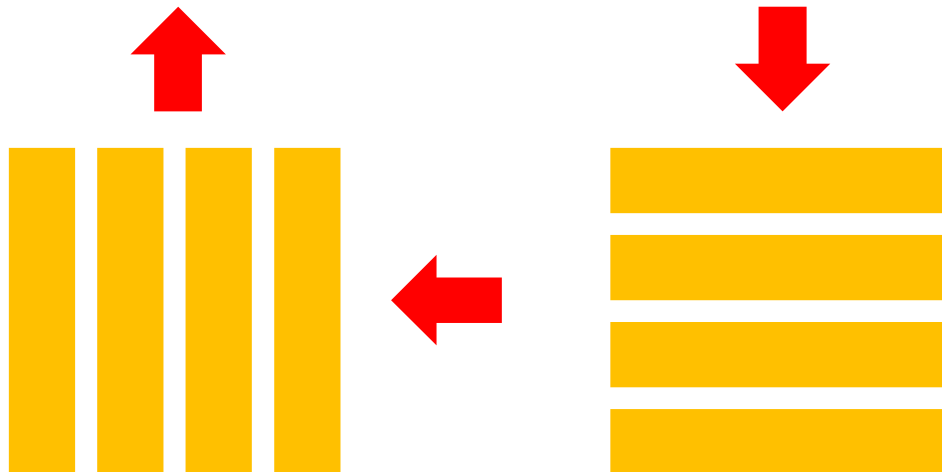
- Products: BlazingSQL, Kinetica, HeavyDB (OmniSciDB, MapD), SQream
- They make use of the parallel power of GPU's
- Long-term data persistency is not their core business

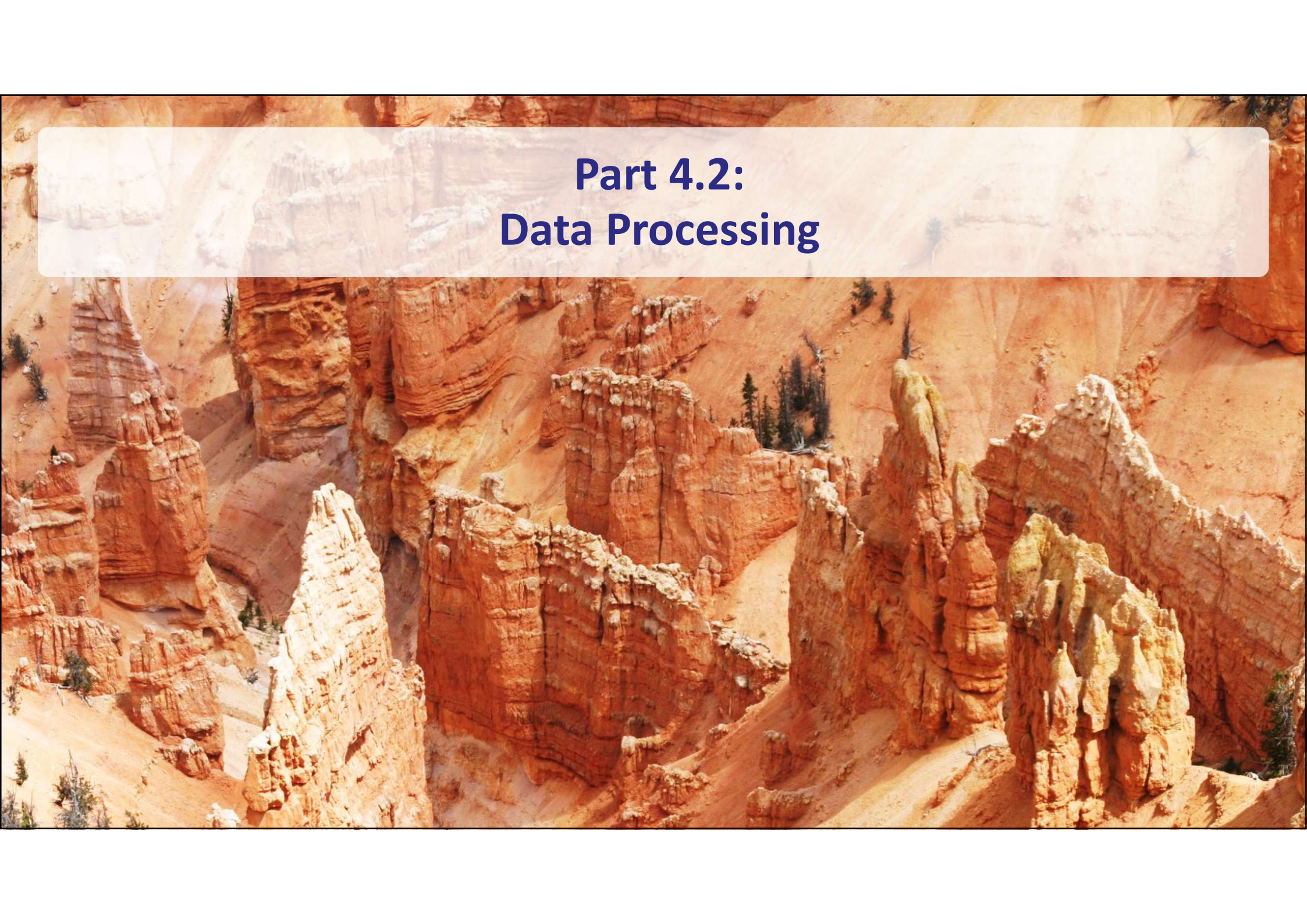
# Four Categories of SQL Databases



# Example: SingleStore (translytical)

ID	Name	Initials	Date Entered	City	State
12345	Young	N	Aug 4, 2008	San Francisco	CA
23324	Stills	S	Sep 10, 2009	New Orleans	LA
57657	Furay	R	Oct 16, 2010	Yellow Springs	OH
65461	Palmer	B	Nov 22, 2011	Boston	MA
...	...	...	...	...	...

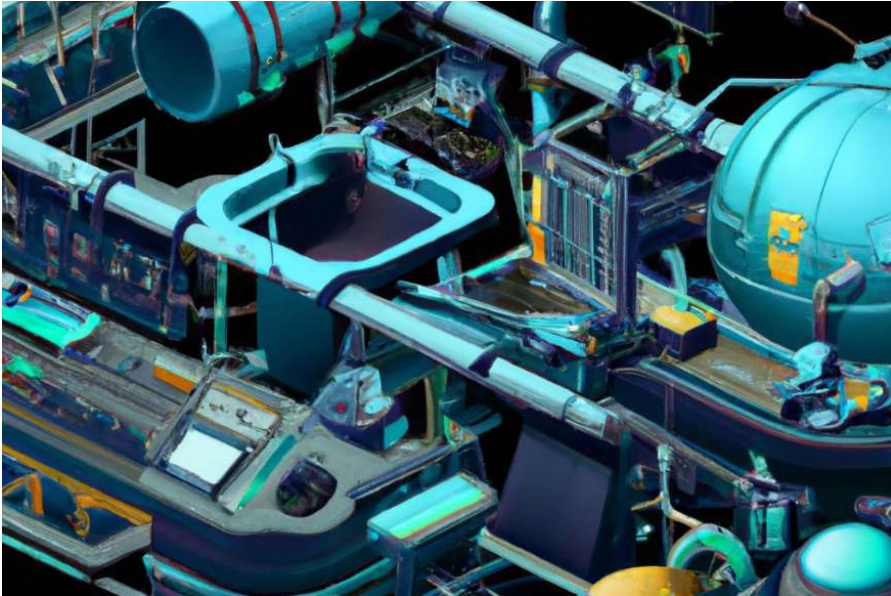




## Part 4.2: Data Processing

# Categories for Data Processing

---

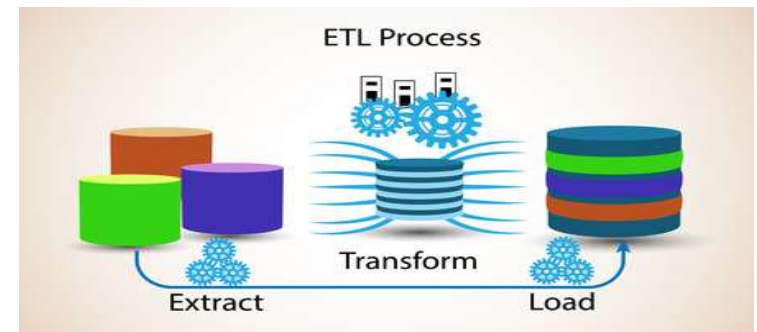


- ETL (Extract Transform Load)
- Data Replication (Change Data Capture)
- ESB (Enterprise Service Bus)
- Data Virtualization

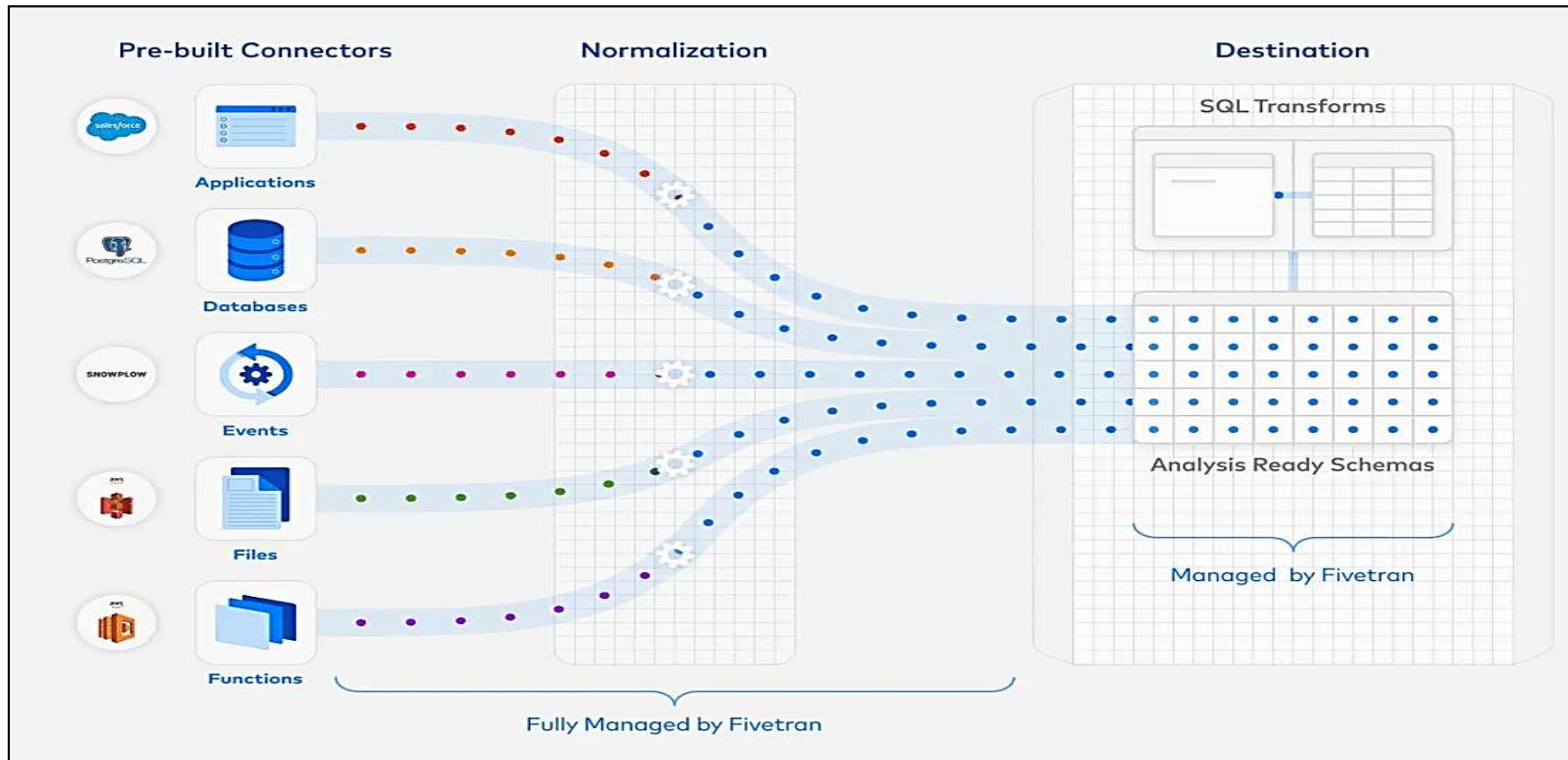
# ETL = Extract Transform Load

---

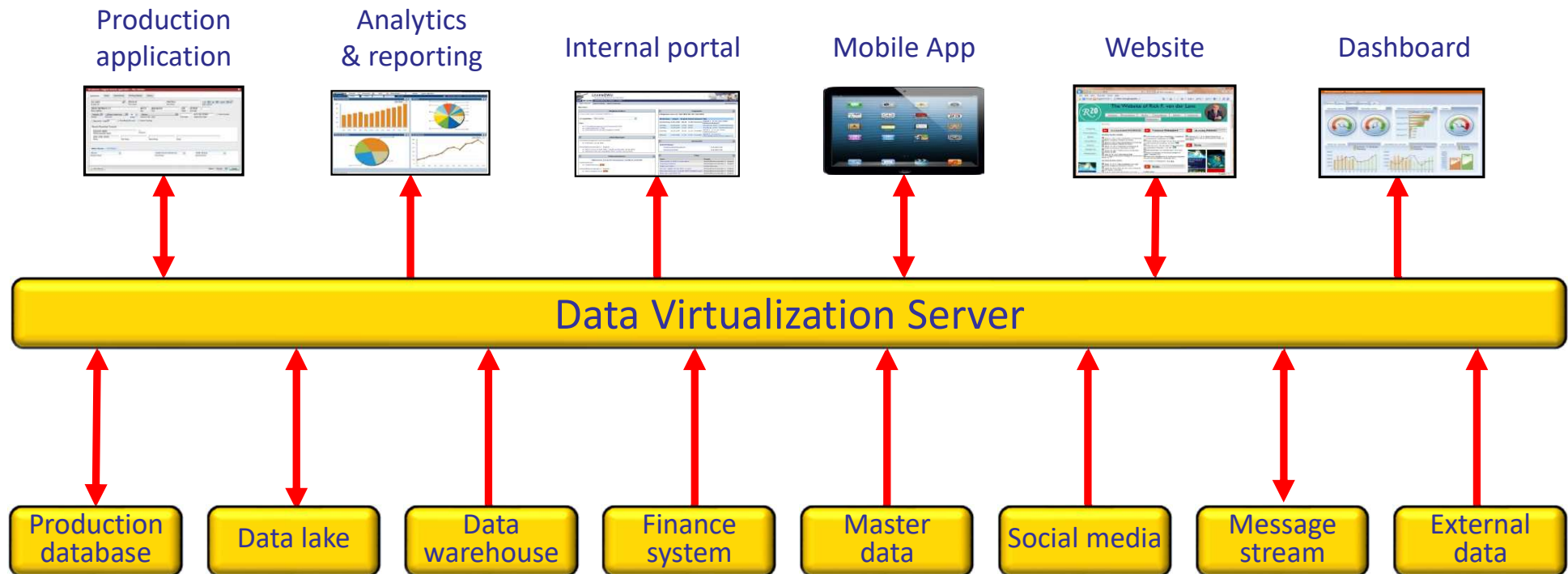
- Transforming of data structures
  - To a data structure suitable for reporting and analysis
- Cleansing of data
- Integration of data from production systems
- Transforming data
  - Filtering, aggregating, projecting, joining, splitting, ...
- Scheduling the ETL process
  - Batch-oriented
- Managing the ETL process



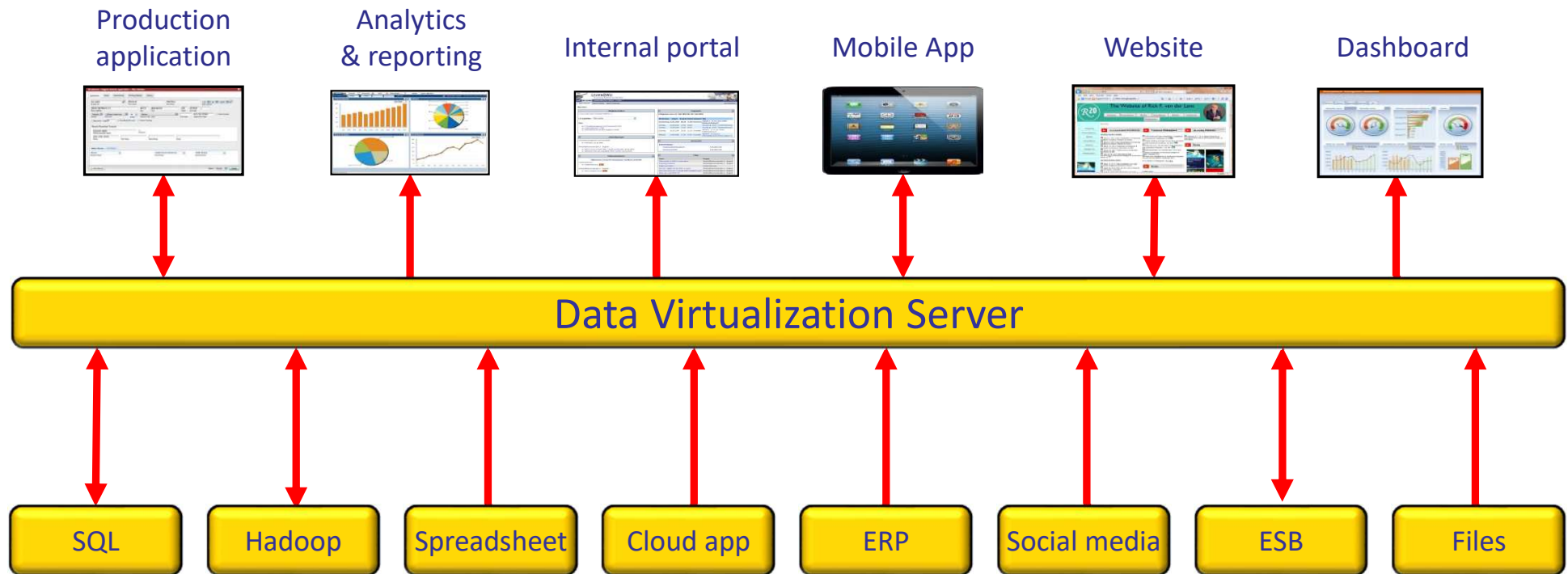
# Example: Fivetran



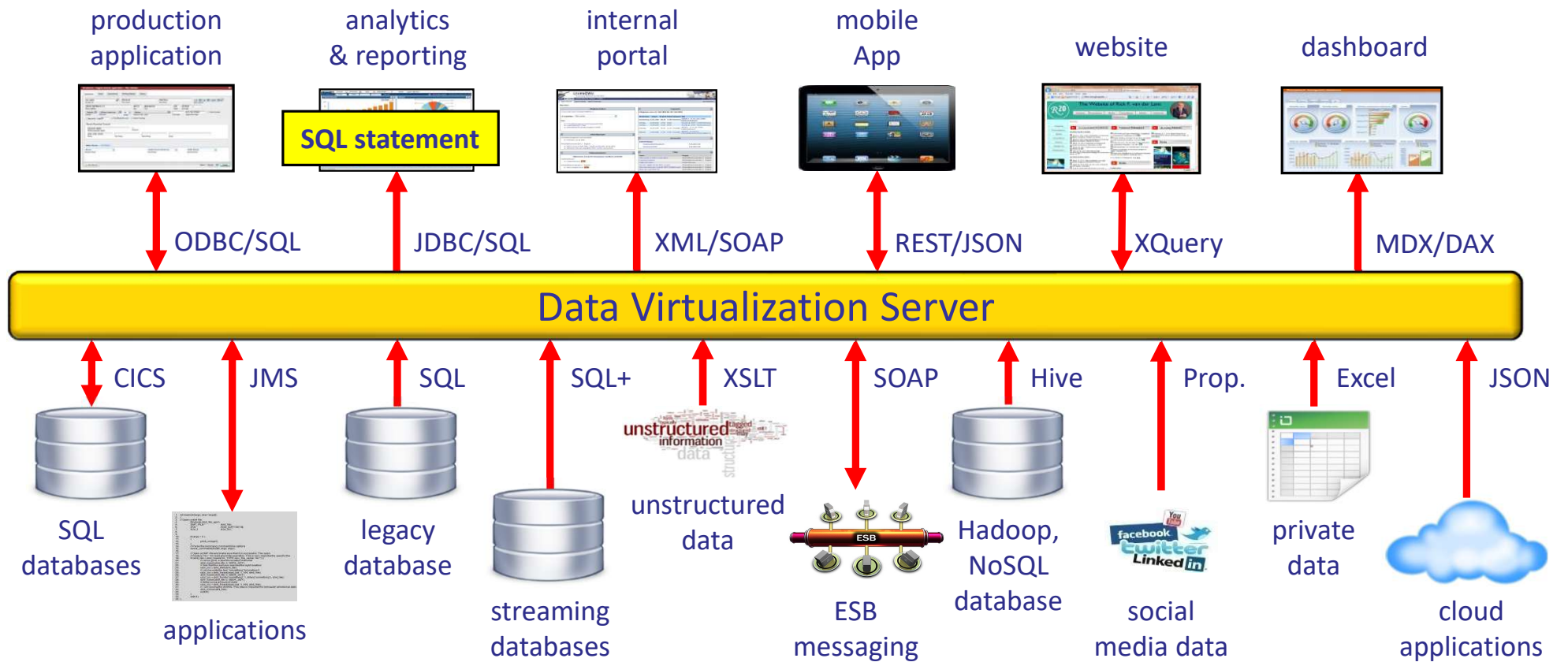
# Data Virtualization Overview (1)



# Data Virtualization Overview (2)



# Data Virtualization Overview (3)

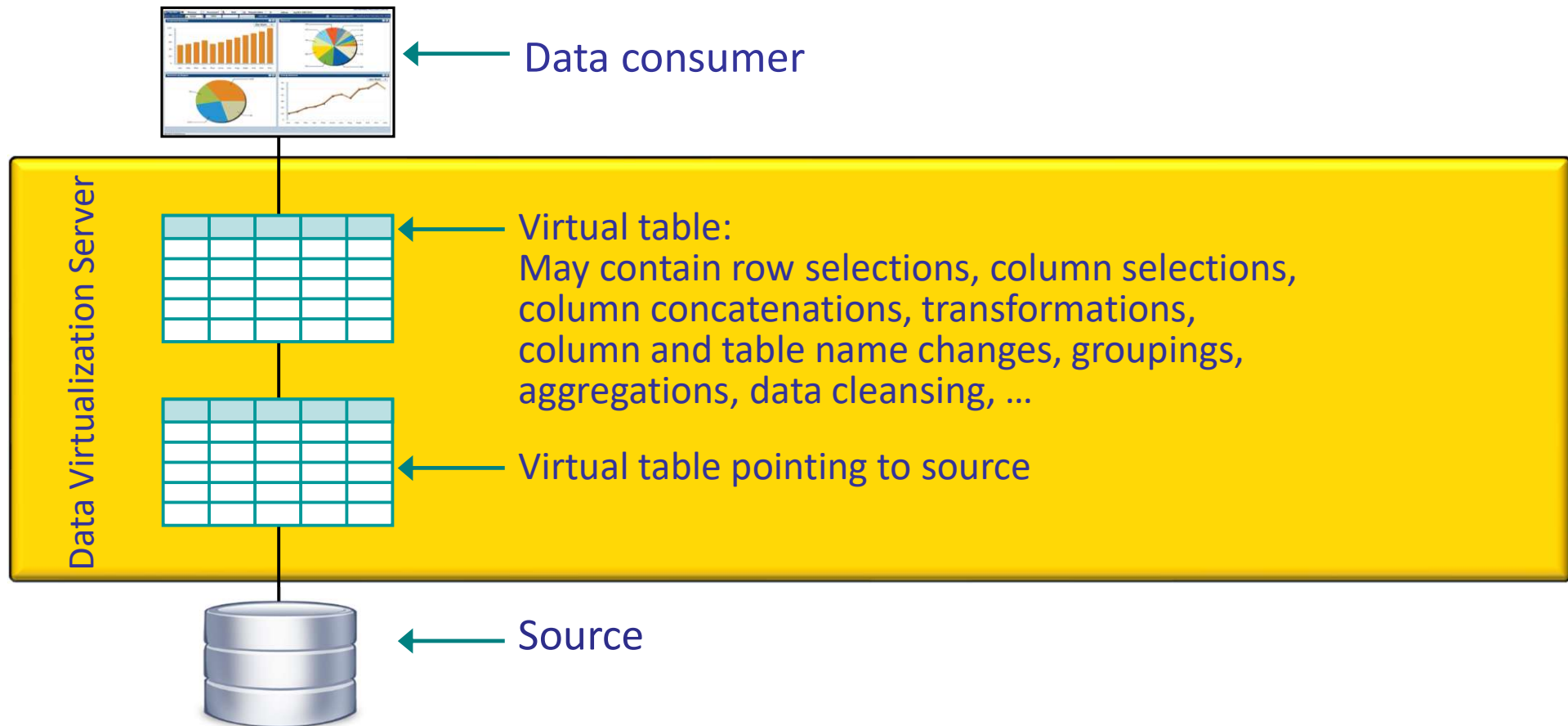


# The Market of Data Virtualization Servers

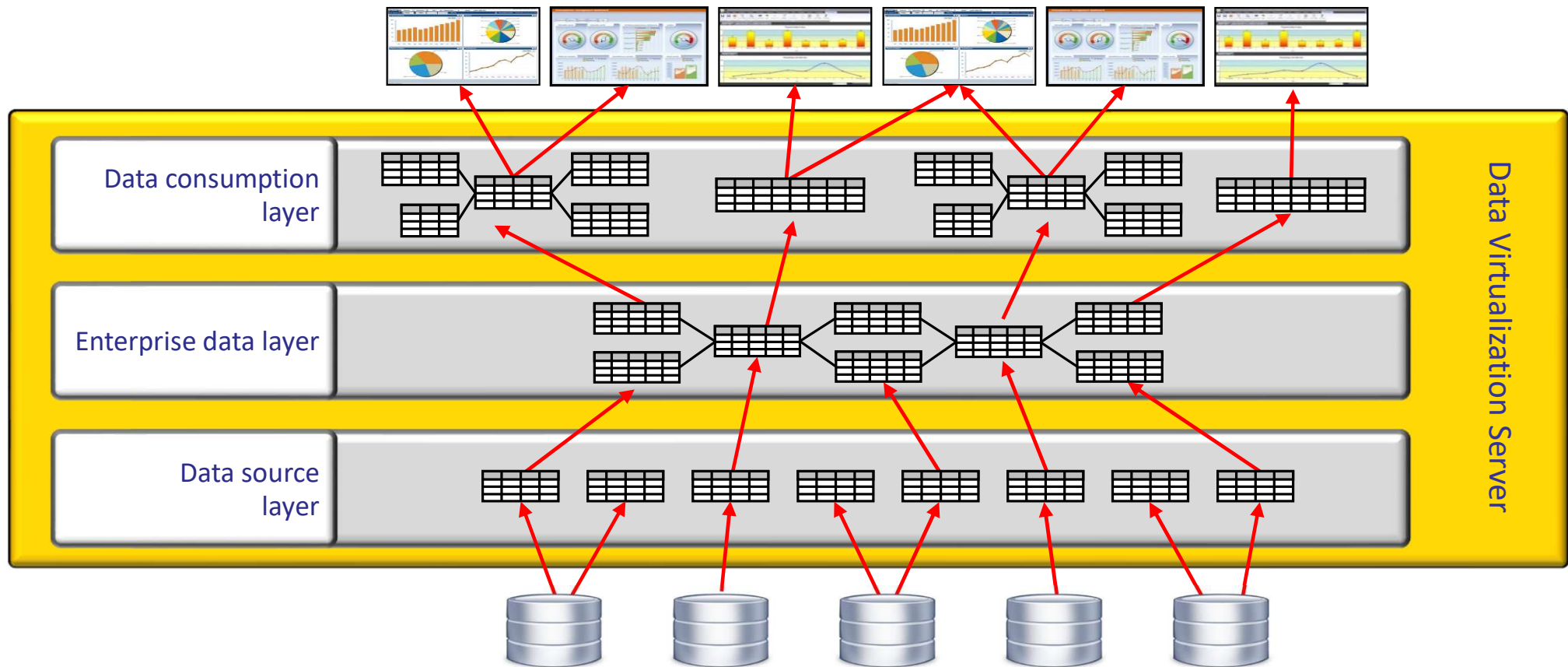


- AtScale
- DataVirtuality (Pipes, Pipes Prof, LDW)
- Denodo Platform
- Dremio
- Fraxses
- IBM InfoSphere Federation Server & IBM Data Virtualization Manager for z/OS (formerly Rocket Data Virtualization)
- Red Hat JBoss Data Virtualization (Teiid) ??
- Stone Bond Enterprise Enabler Virtuoso
- TIBCOData Virtualization (formerly Cisco & Composite)
- Zetaris
- And many more ...

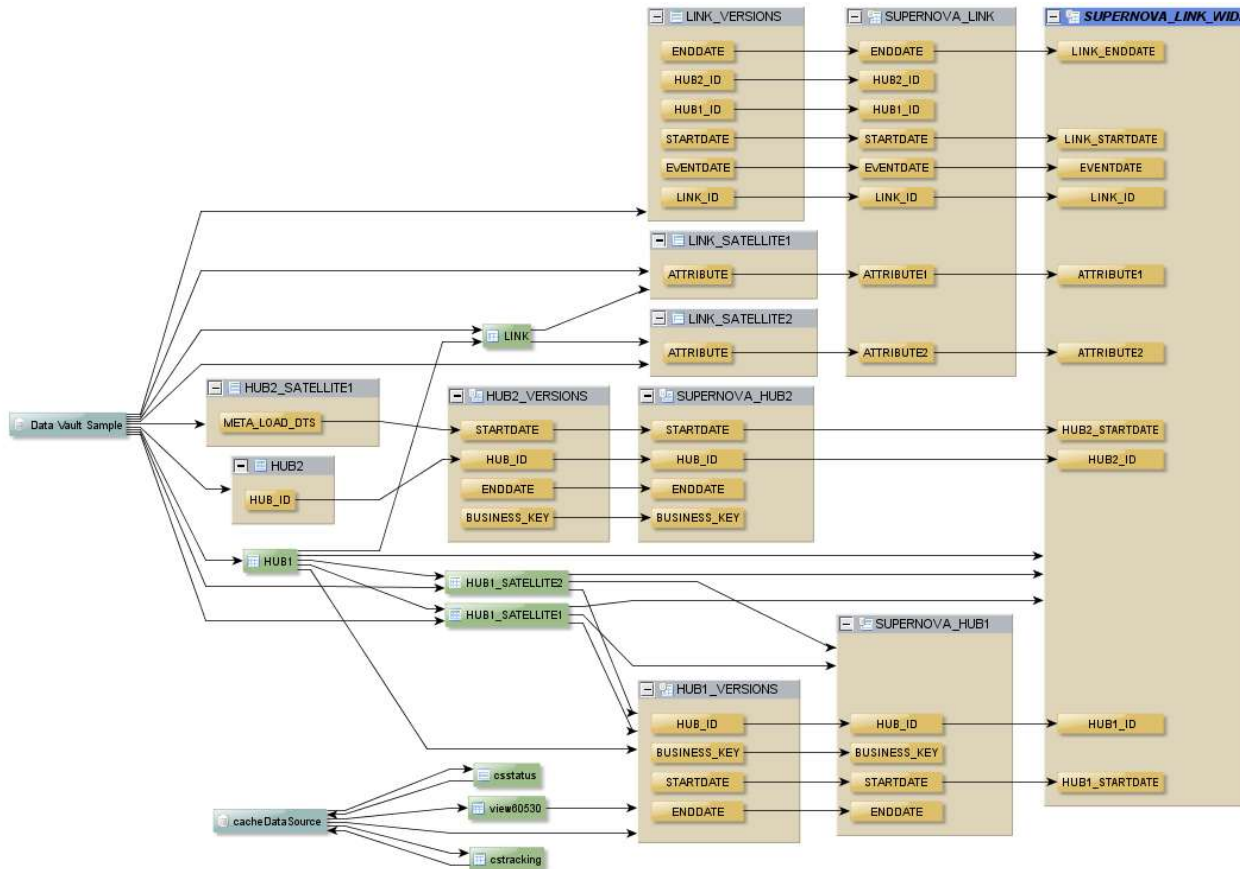
# Developing Virtual Tables



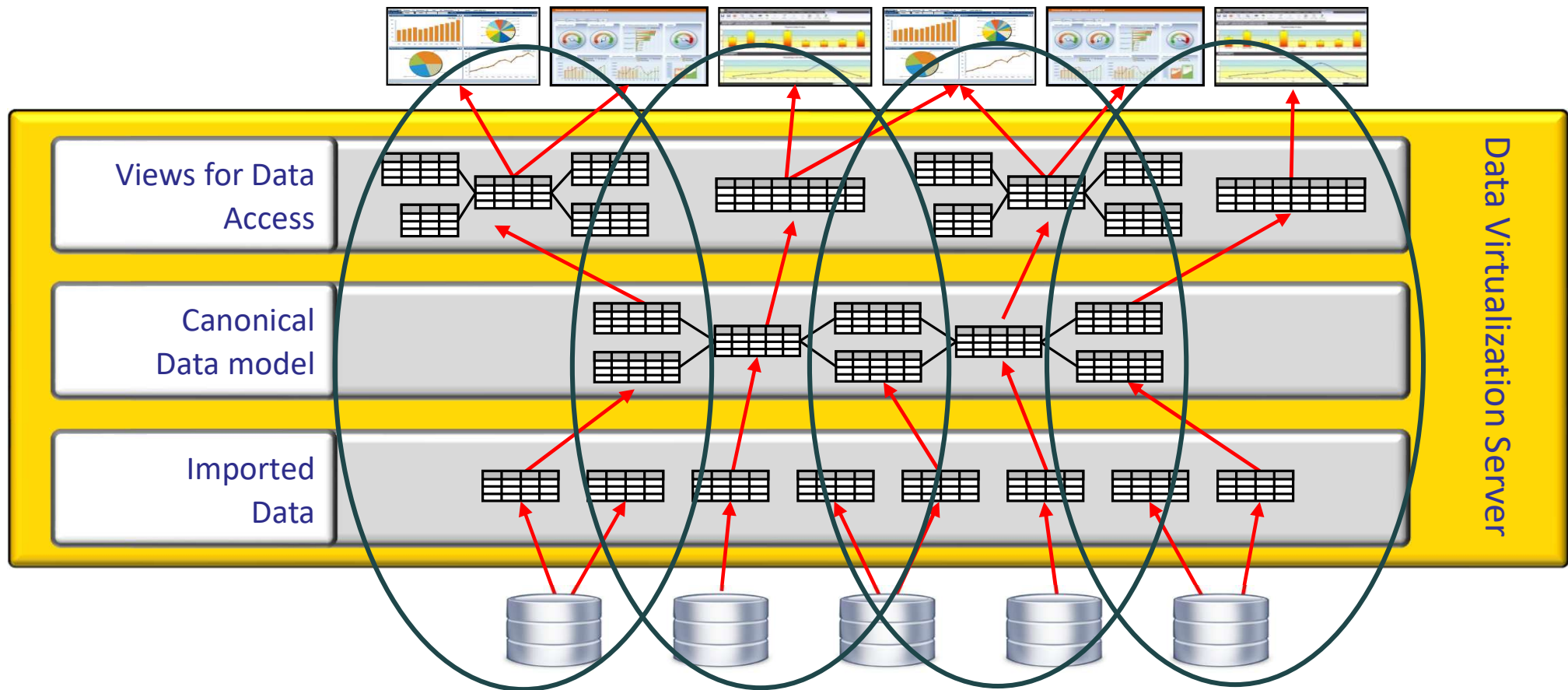
# Layers of Virtual Tables



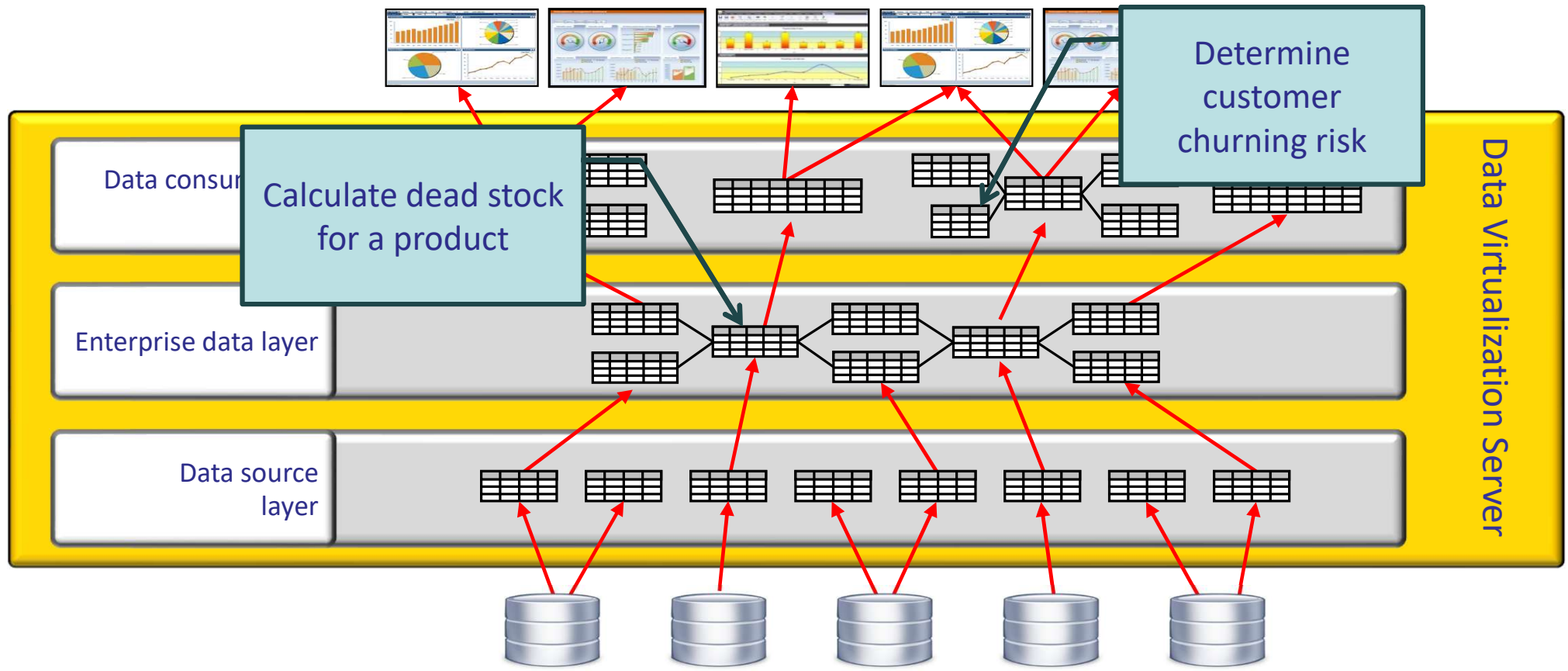
# Lineage and Impact Analysis



# Evolutionary Development Approach



# Improved Productivity Through Sharing



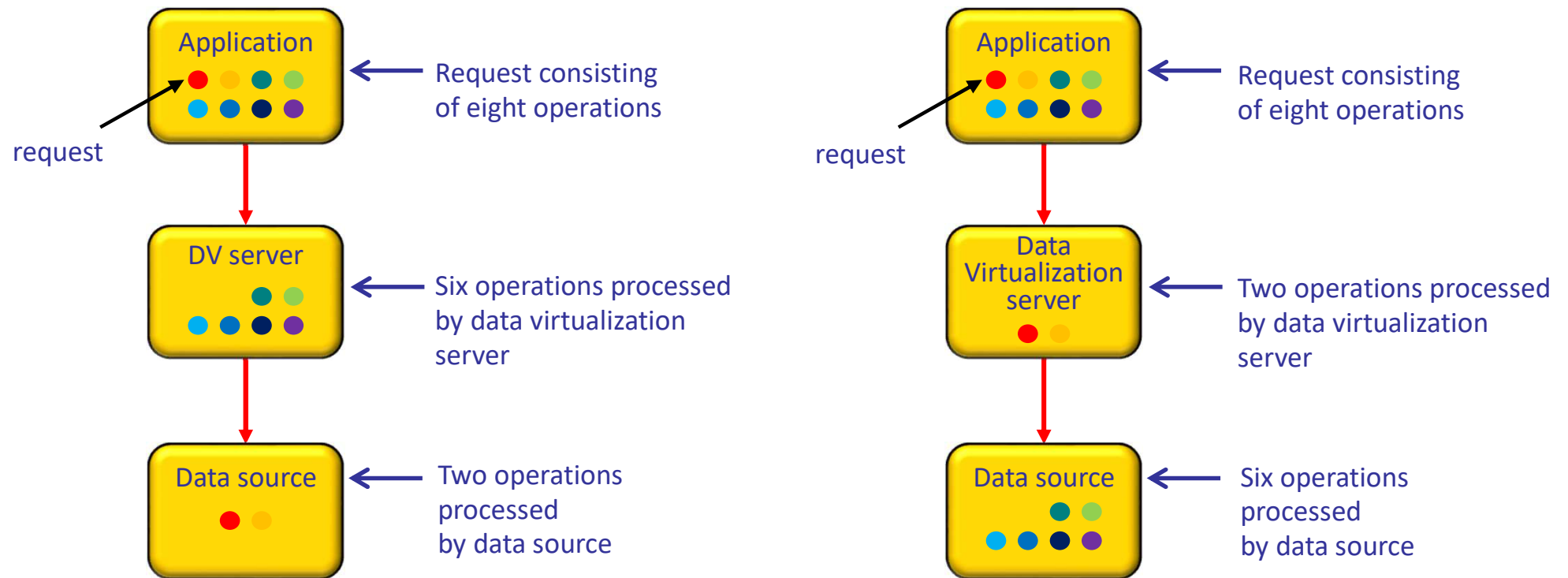
# Performance Improving Features

---

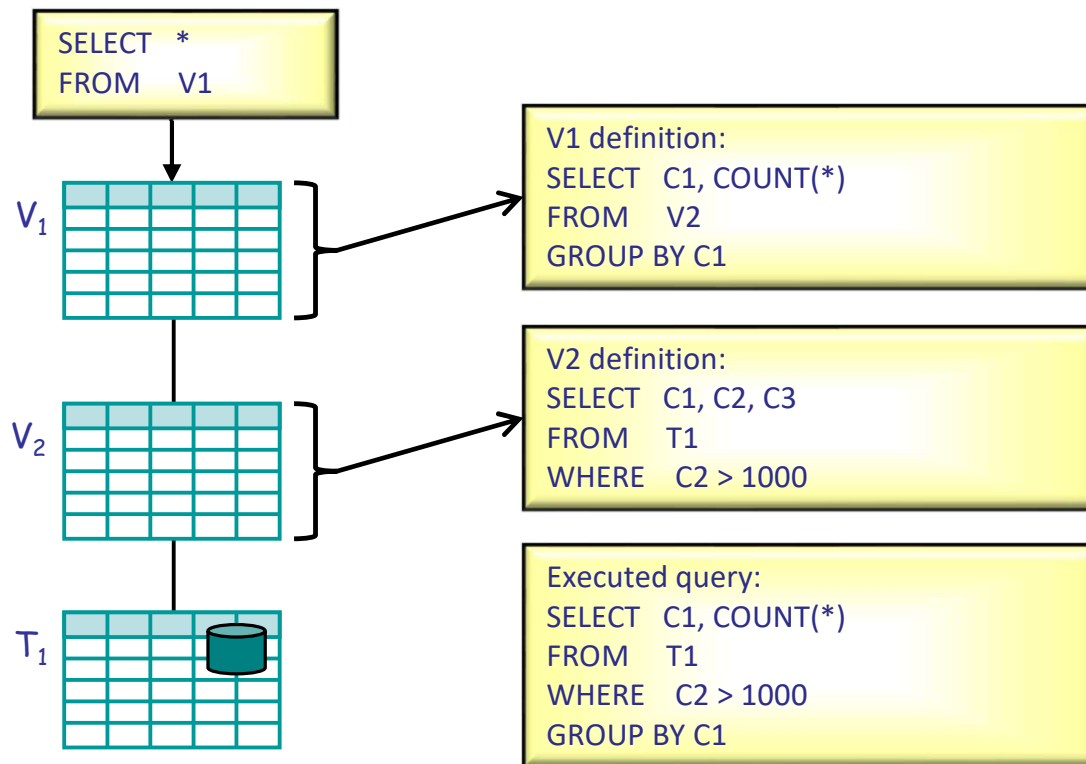


- Easy-to-optimize queries
- Environment setup
- Query optimization
- Parallel processing and parallel pushdown
- Caching virtual tables
- The network
- Efficient drivers and connectors

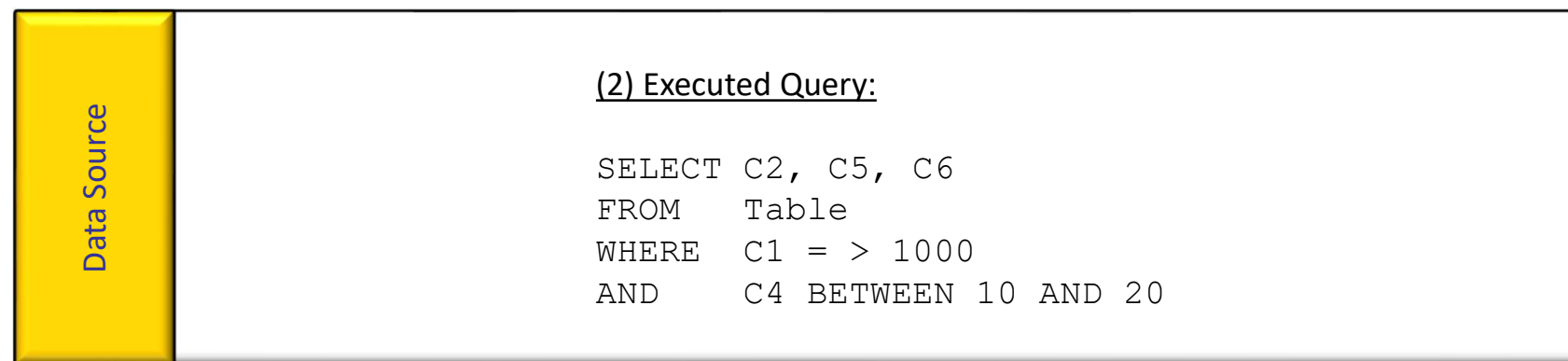
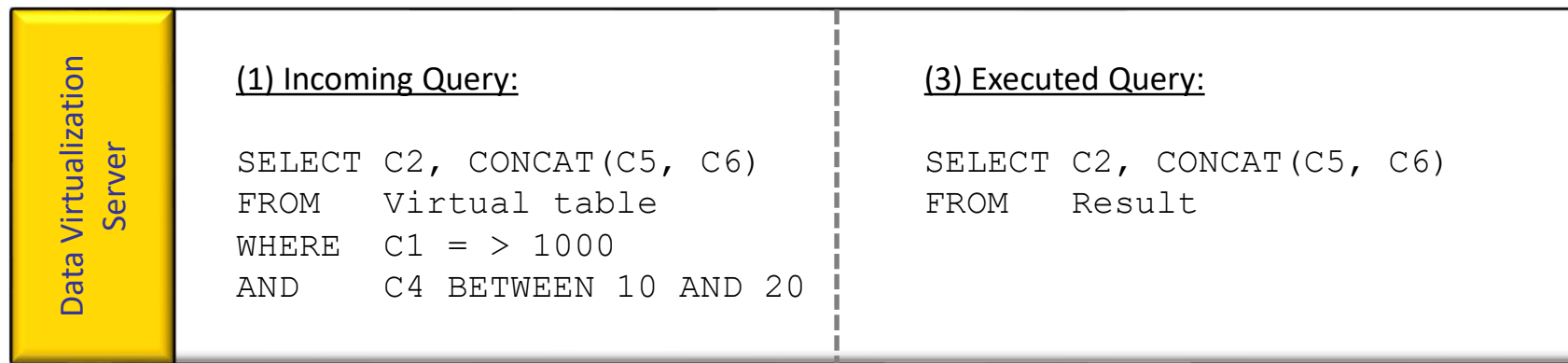
# Improved Performance Through Query Pushdown



# Many Levels and One Query



# Push Down Query Processing



# Accessing Files

Data Virtualization  
Server

## (1) Incoming Query:

```
SELECT C2, CONCAT(C5, C6)
FROM   Virtual table
WHERE  C1 = > 1000
AND    C4 BETWEEN 10 AND 20
```

## (3) Executed Query:

```
SELECT C2, CONCAT(C5, C6)
FROM   Result
WHERE  C1 = > 1000
AND    C4 BETWEEN 10 AND 20
```

Data Source

## (2) Executed Query:

```
SELECT C1, C2, C4, C5, C6
FROM   File
```

# Inferred Filters

Data Virtualization  
Server

## (1) Incoming Query:

```
SELECT T1.C1, T1.C2, T2.C2
FROM   T1, T2
WHERE  T1.C1 = T2.C1
AND    T1.C1 => 1000
AND    T2.C2 BETWEEN 10 AND 20
```

## (3) Executed Query:

```
SELECT T1.C1, T1.C2, T2.C2 FROM
T1, T2
WHERE  T1.C1 = T2.C1
```

Data Source

## (2a) Executed Query:

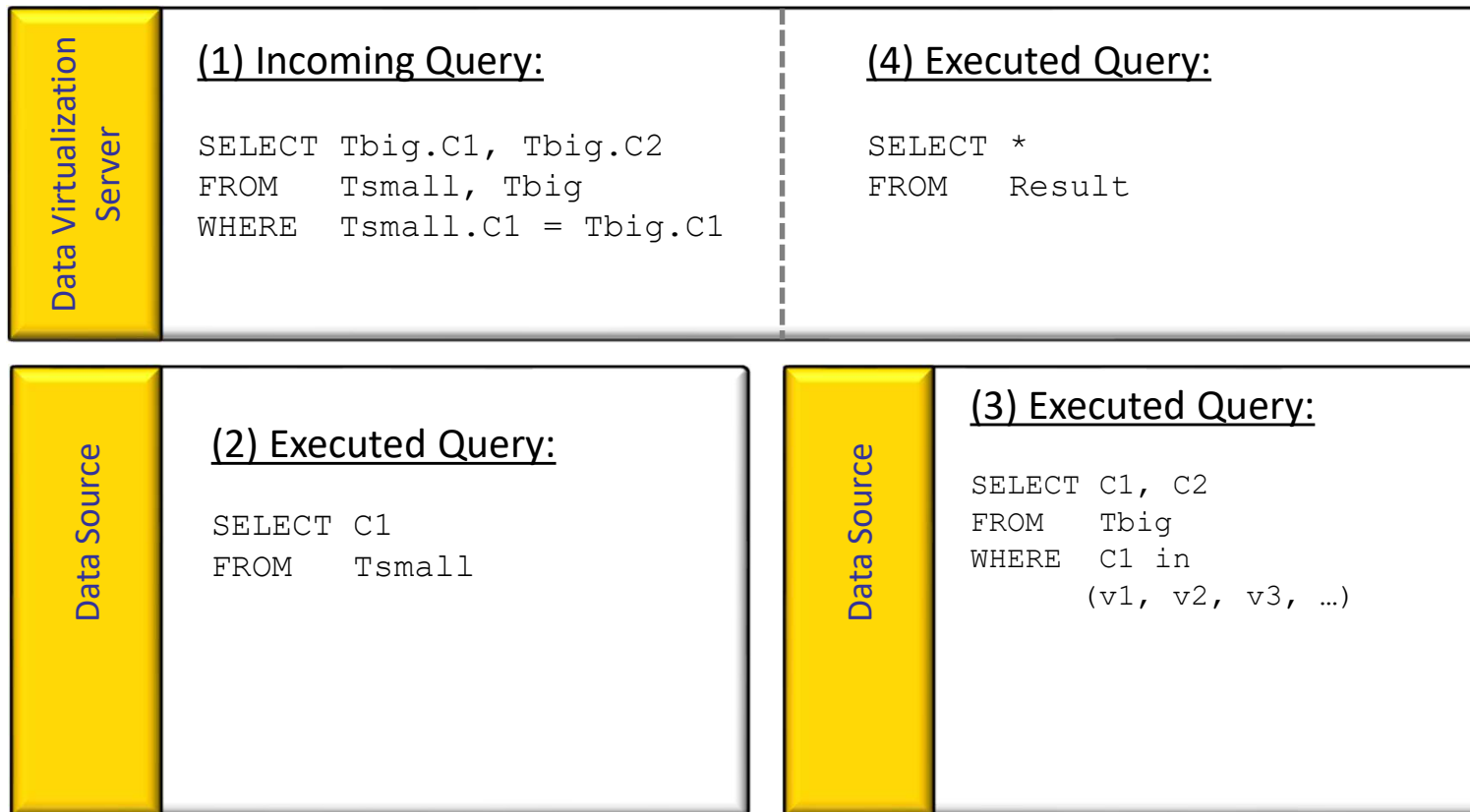
```
SELECT C1, C2
FROM   T1
WHERE  C1 => 1000
```

Data Source

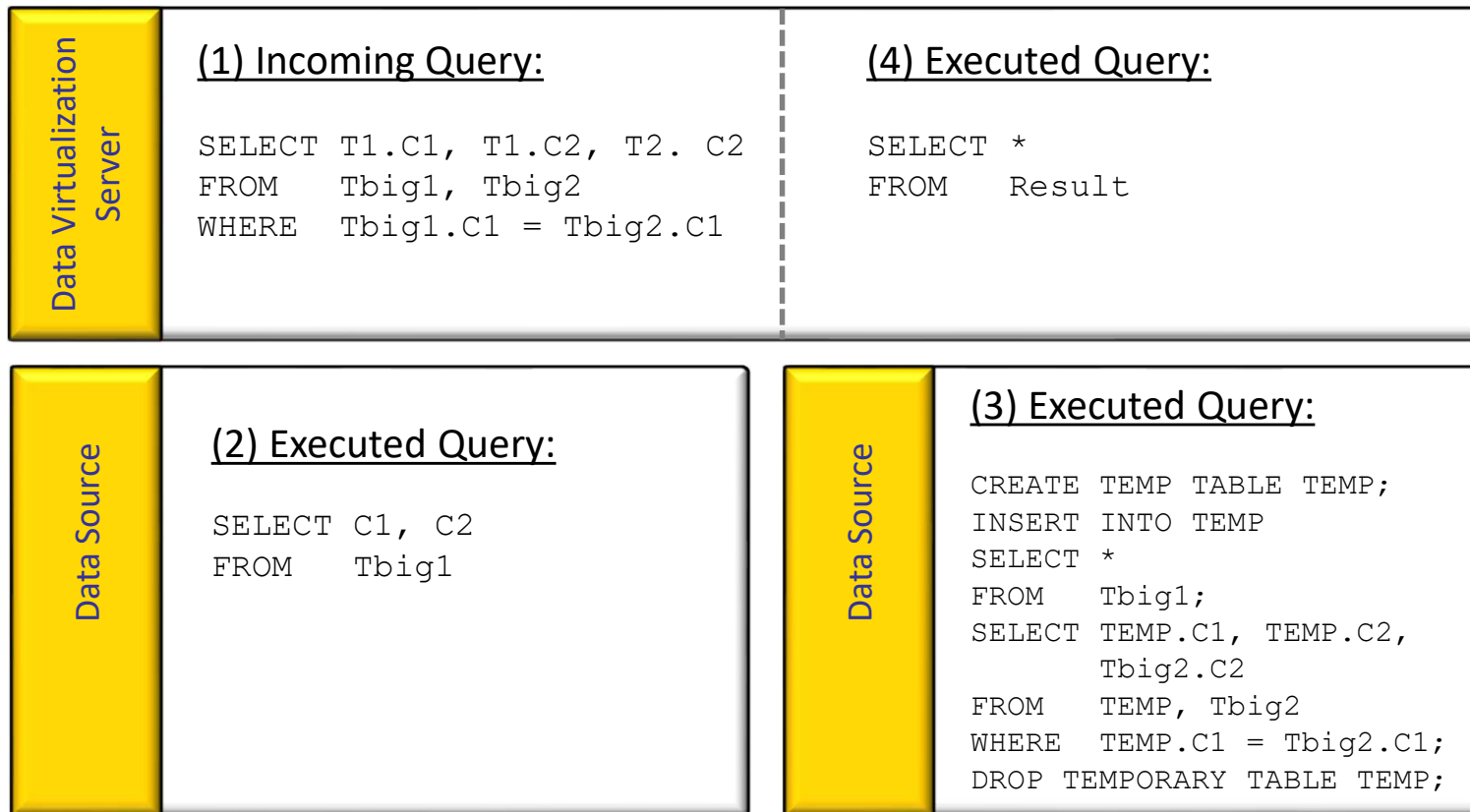
## (2b) Executed Query:

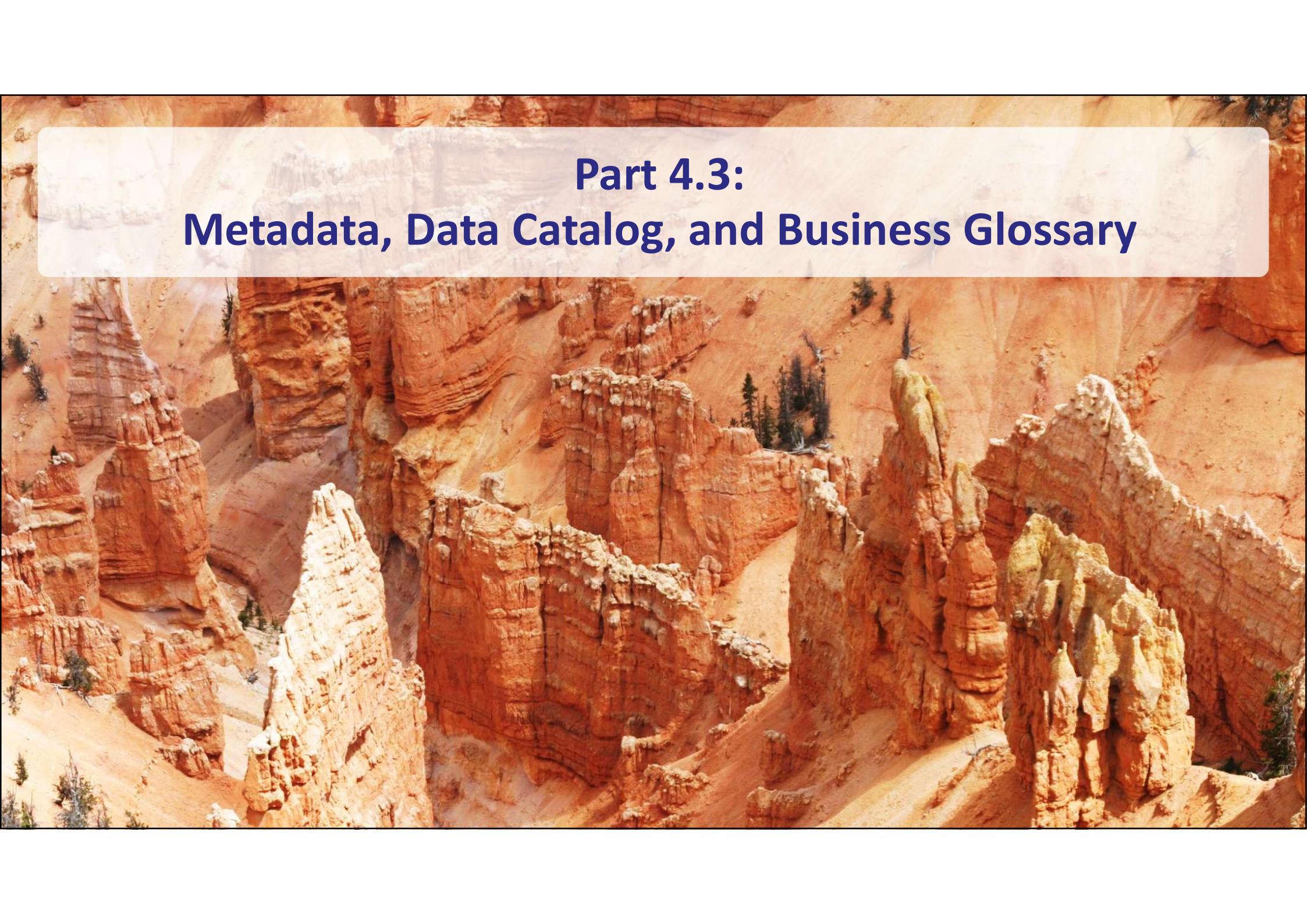
```
SELECT C1, C2
FROM   T2
WHERE  T2.C1 => 1000
AND    T2.C2 BETWEEN 10
AND 20
```

# Join of Data Sources with Query Injection



# Join of Data Sources with Ship Join



A photograph of a desert canyon with tall, layered rock spires and a semi-transparent text box overlaid on top. The text box contains the title "Part 4.3: Metadata, Data Catalog, and Business Glossary" in a bold, dark blue font. The background shows the intricate, orange-brown rock formations of a canyon, with some sparse green trees and shrubs scattered throughout.

## **Part 4.3: Metadata, Data Catalog, and Business Glossary**

# Types of Metadata

## Metadata on Data

- Textual definition
- Description
- Annotations by users and IT specialists
- Data lineage including transformations
- Retention information
- Qualifications: trustworthiness, completeness, data quality, ...
- Value descriptions
- Original or masked/anonymized
- Ontology
- Owner and support
- ...

## Metadata on Metadata

- Retention information on metadata
- History of metadata
- Qualifications: trustworthiness, completeness, data quality, ...
- Metadata value descriptions
- Original or masked/anonymized metadata
- Owner and support
- ...

# Metadata in 1976

---

## DE DATA DICTIONARY/DIRECTORY (DD/D)

---

door L. Delpont

### 1.1 Wat is een DD/D? (Data Dictionary/Directory) (gegevenskataloog)

Zeer algemeen kunnen we een DD/D als volgt bepalen:

Een DD/D is een katalogus die de omschrijving bevat van alle informatie-elementen die in een bedrijf bestaan.' Deze bepaling laat natuurlijk de weg open

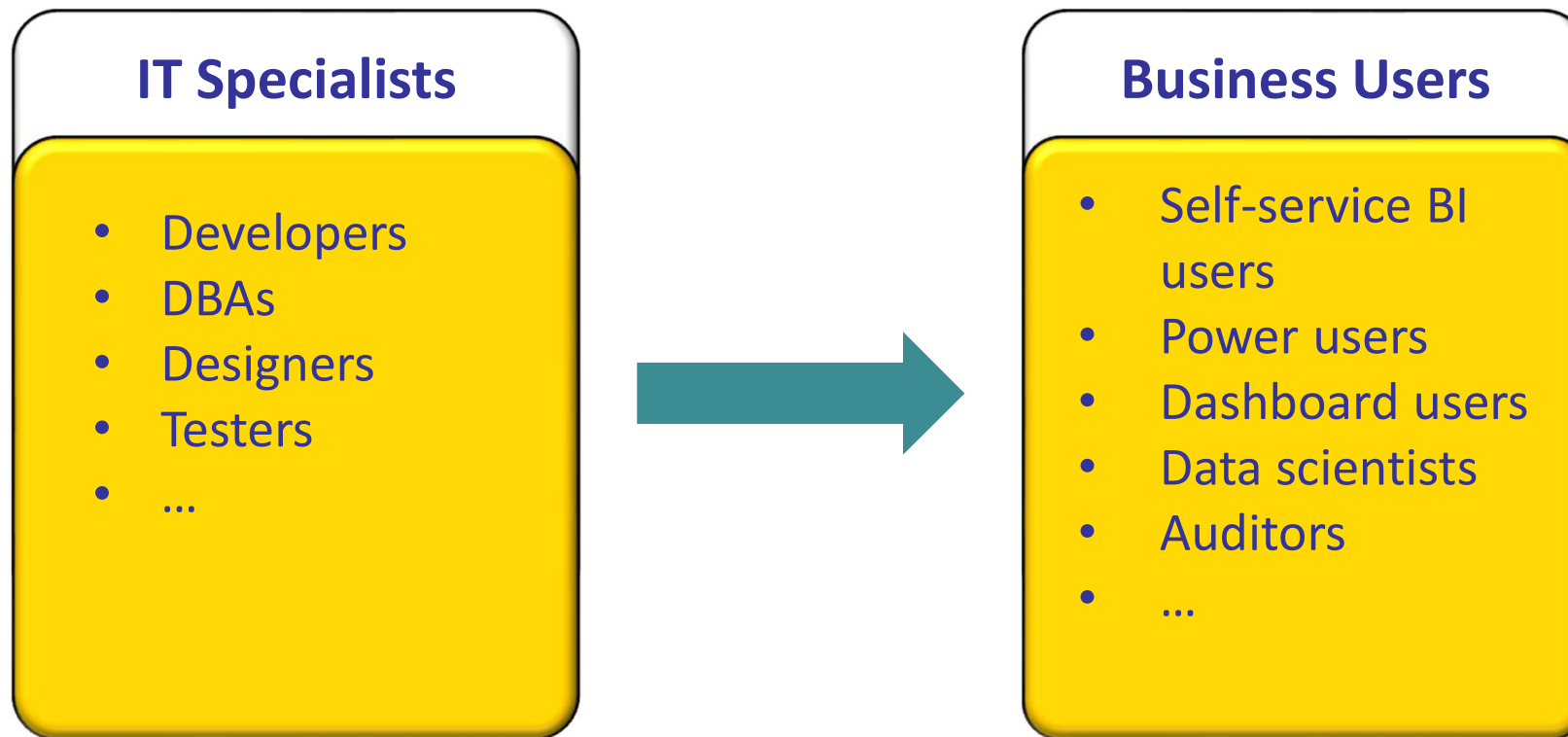
### 1.2 Waarom hebben we een DD/D nodig?

Centralisatie, het vermijden van dubbele gegevens en het opzoeken en vinden van informatie langs allerlei wegen en door middel van allerlei sleutels zijn technieken eigen aan M.I.S. (7) en systemen van gegevensbanken.

*Informatie jaargang 18 nr. 7/8 pag. 430 t/m 491 Amsterdam juli/augustus 1976*

# A Target Audience Shift of Metadata

---



# Numerous Solutions that Manage Metadata

---

- Home made metadata systems
- Professional data catalogs and business glossaries: Alation, Apache Atlas, Collibra, Informatica, TIBCO EBX, ...
- Scraping and linking: ASG, Manta, SQLdep (Collibra), ...
- Data warehouse automation: Astera, Attunity Compose, BiGenius, TimeXtender, WhereScape, ...
- BI tools: semantic layers
- ETL tools
- Data profiling tools
- Data quality tools
- Data virtualization servers: Data Virtuality, Denodo, Fraxses, Tibco DV, ...
- And many more ...

# Instant Metadata

Stock overview

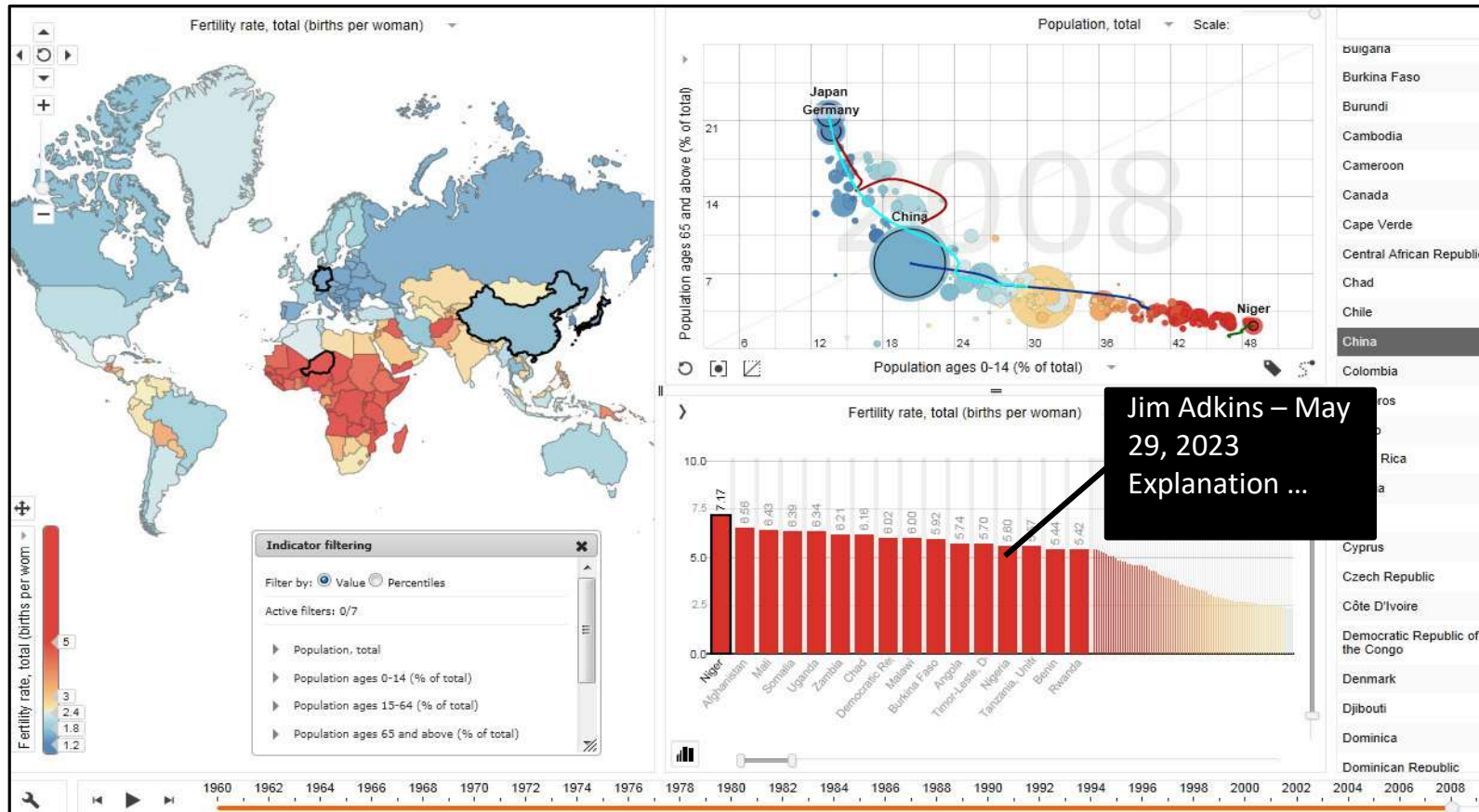
## Cohelion

Note: This is an aggregated report. Most data is first calculated at the native resolution (per group and pool), and the results are summed or averaged for an aggregated report.

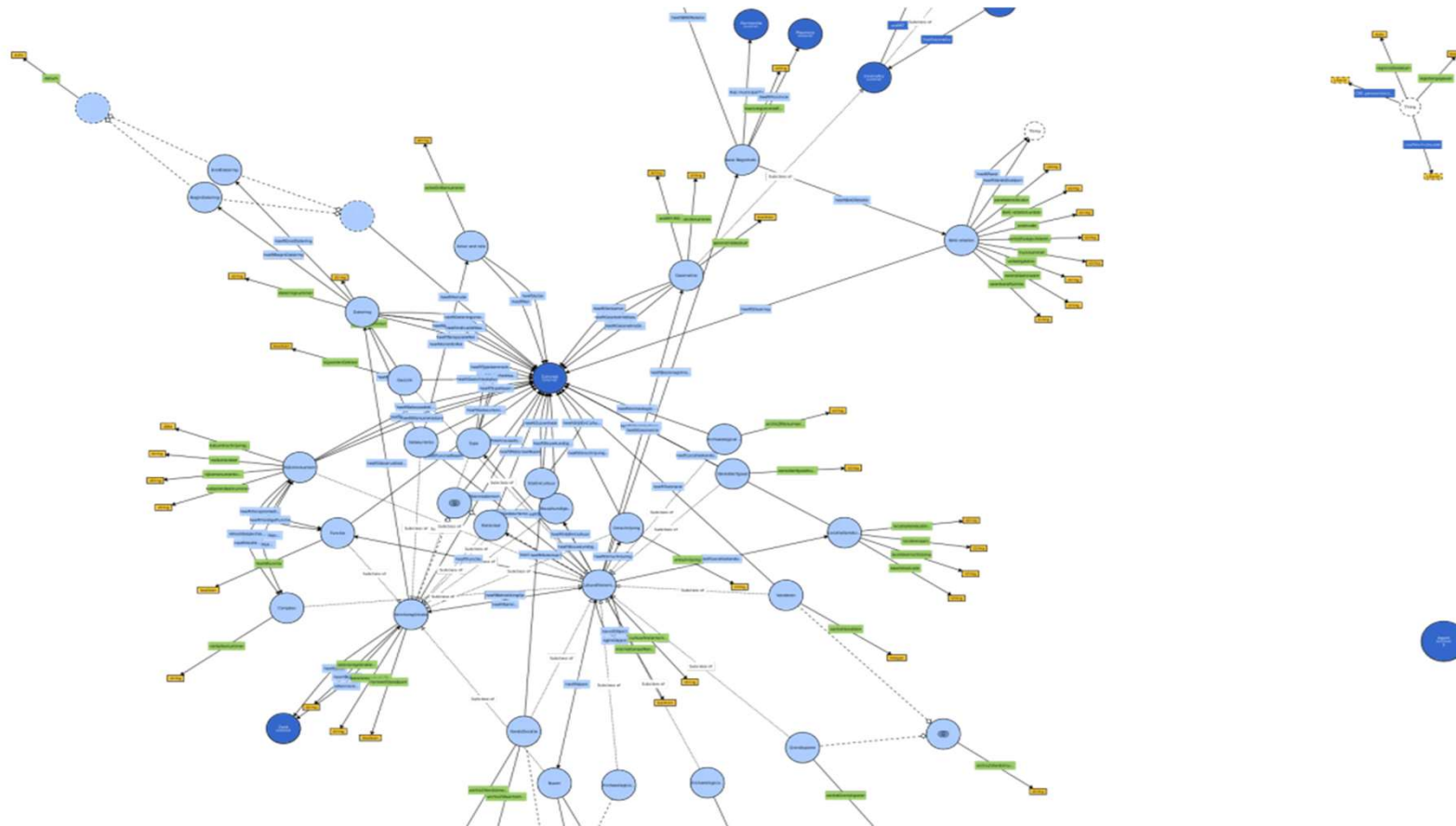
	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May	Trend vs last year	Overall
<b>Total demand</b>	128,213	127,098	117,923	116,033	115,431	108,253	114,681	117,610	115,882	115,554	115,122	116,106	116,849		117,289
<b>Total inventory</b>	145,401	144,654	144,753	143,668	142,977	142,224	141,258	139,011	138,414	137,690	137,108	136,660	136,191		140,770
<b>Total available inventory</b>	111,562	111,719	113,504	114,175	114,342	112,682	113,106	112,081	110,908	109,726	108,631	107,575	106,545		111,274
<b>Available Bicycles</b>	100,653	101,344	102,974	103,659	103,766	101,010	101,180	100,155	98,982	97,800	96,705	95,649	94,619		99,884
<b>Rented Bicycles</b>	10,909	10,375	10,530	10,516	10,576	11,672	11,926	11,926	11,926	11,926	11,926	11,926	11,926		11,389
<b>Unservicable Bicycles</b>	6,490	6,528	7,793	6,931	7,094	7,716	7,990	7,040	7,040	7,040	7,040	7,040	7,158		7,142
<b>In repair Bicycles</b>	2,445	2,508	2,635	2,809	2,607	2,749	2,499	2,744	2,744	2,744	2,744	2,744	2,667		2,652
<b>Storage Bicycles</b>	1,223	1,652	1,526	3,836	4,205	4,568	4,934	2,515	2,515	2,515	2,515	2,515	3,072		2,981
<b>Lost / Stolen Bicycles</b>	23,681	22,247	19,295	15,917	14,729	14,509	12,729	14,631	14,631	14,631	14,631	14,631	16,749		16,721
<b>Rebrand In</b>	0	0	0	0	0	0	0	-	-	-	-	-	-		0
<b>Rebrand Out</b>	0	0	0	0	0	0	0	-	-	-	-	-	-		0
<b>Lease In</b>	0	0	9	20	0	35	116	-	-	-	-	-	-		26

**Available Bicycles**  
**Jan**  
 Forecast: 98,982  
 Actual is entered manually. Forecast is the last month repeated, but adjusted for all inventory increase and decrease of that month. Formula: Available Bicycles + Rebrand - Scrap - Loss - Sales Capex Volume. All values are taken from the previous month.

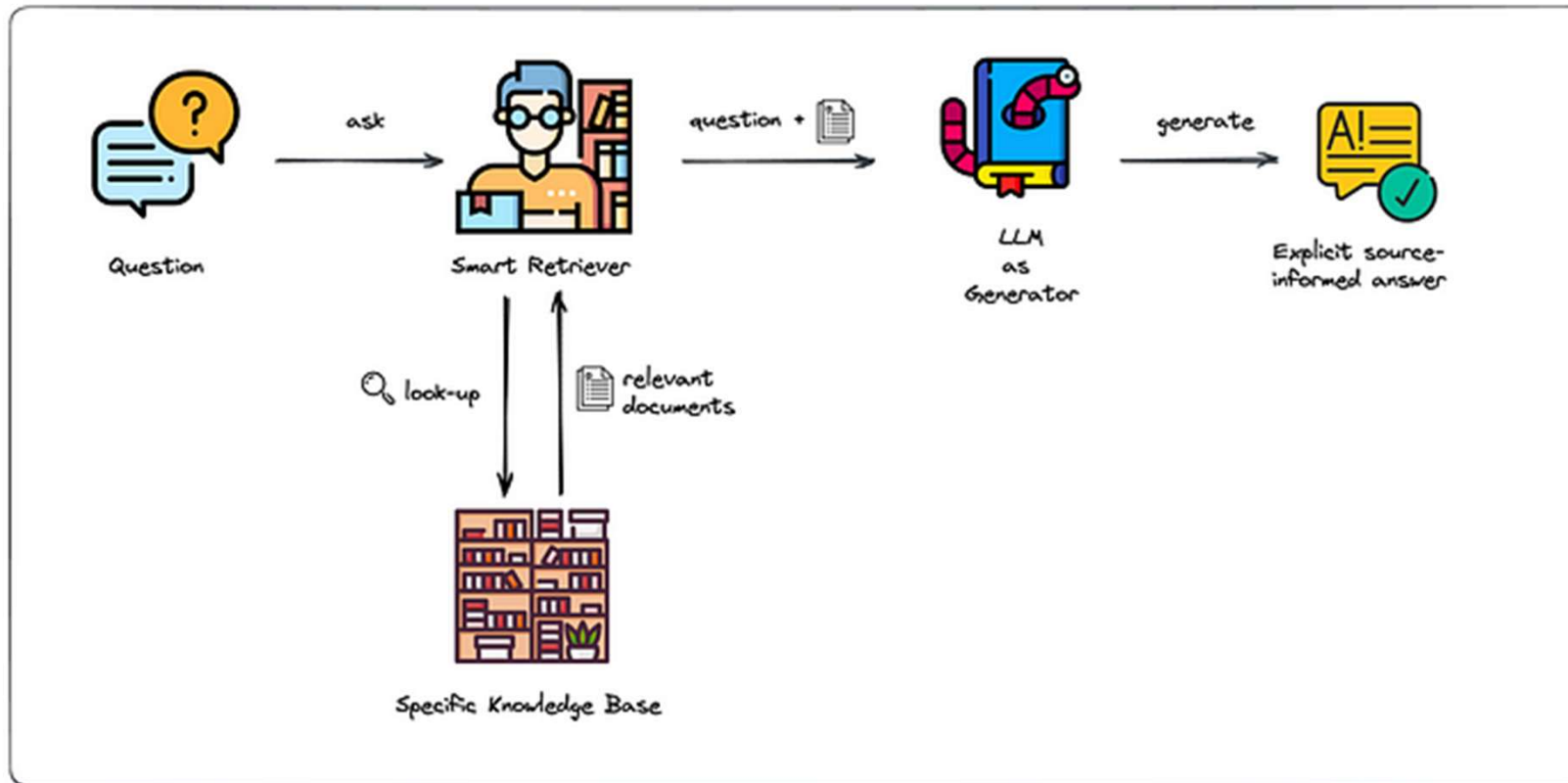
# Adding Annotations on Data Points



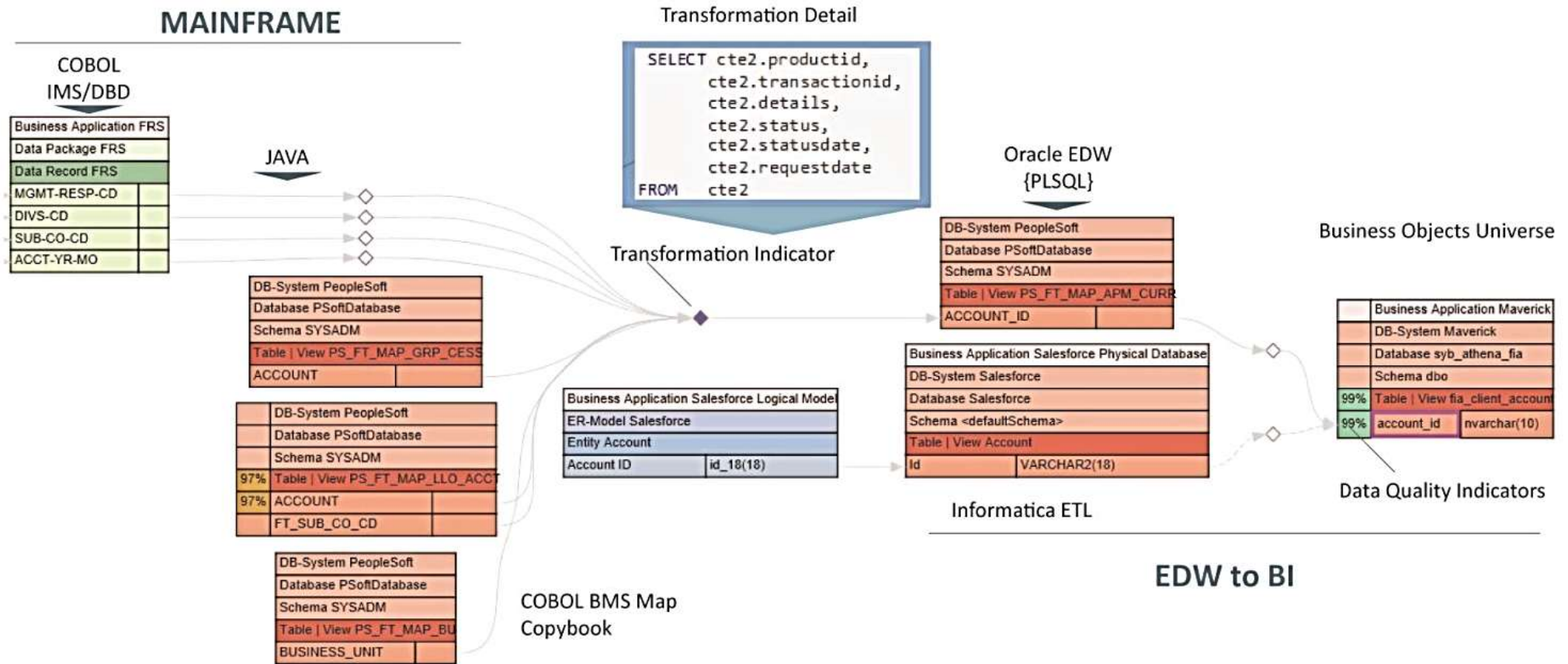
# Generating an Ontology (for AI?)



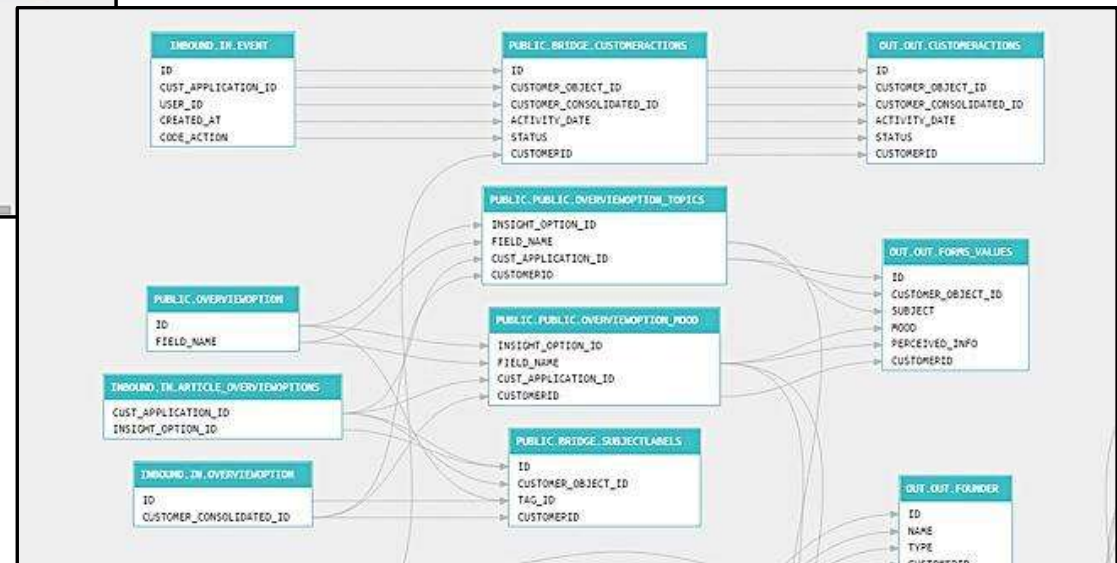
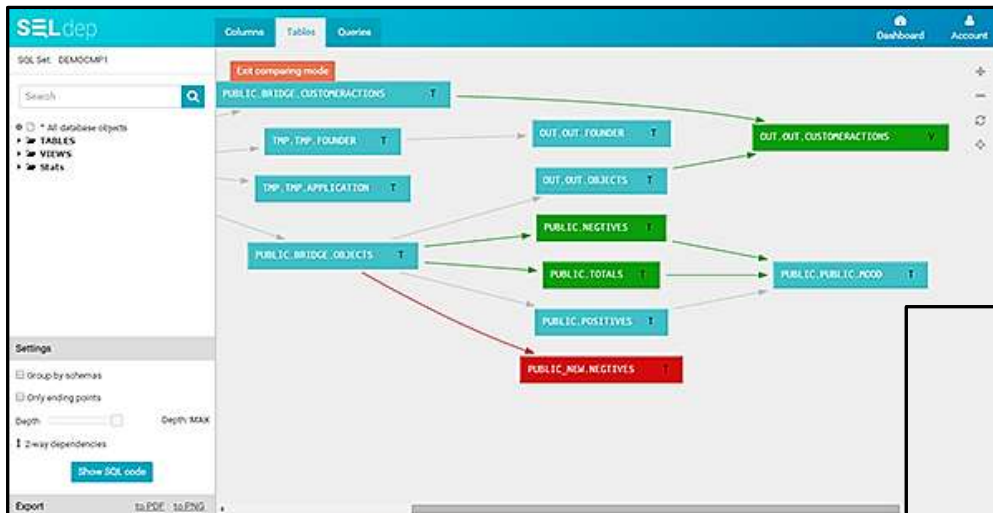
# Retrieval Augmented Generation

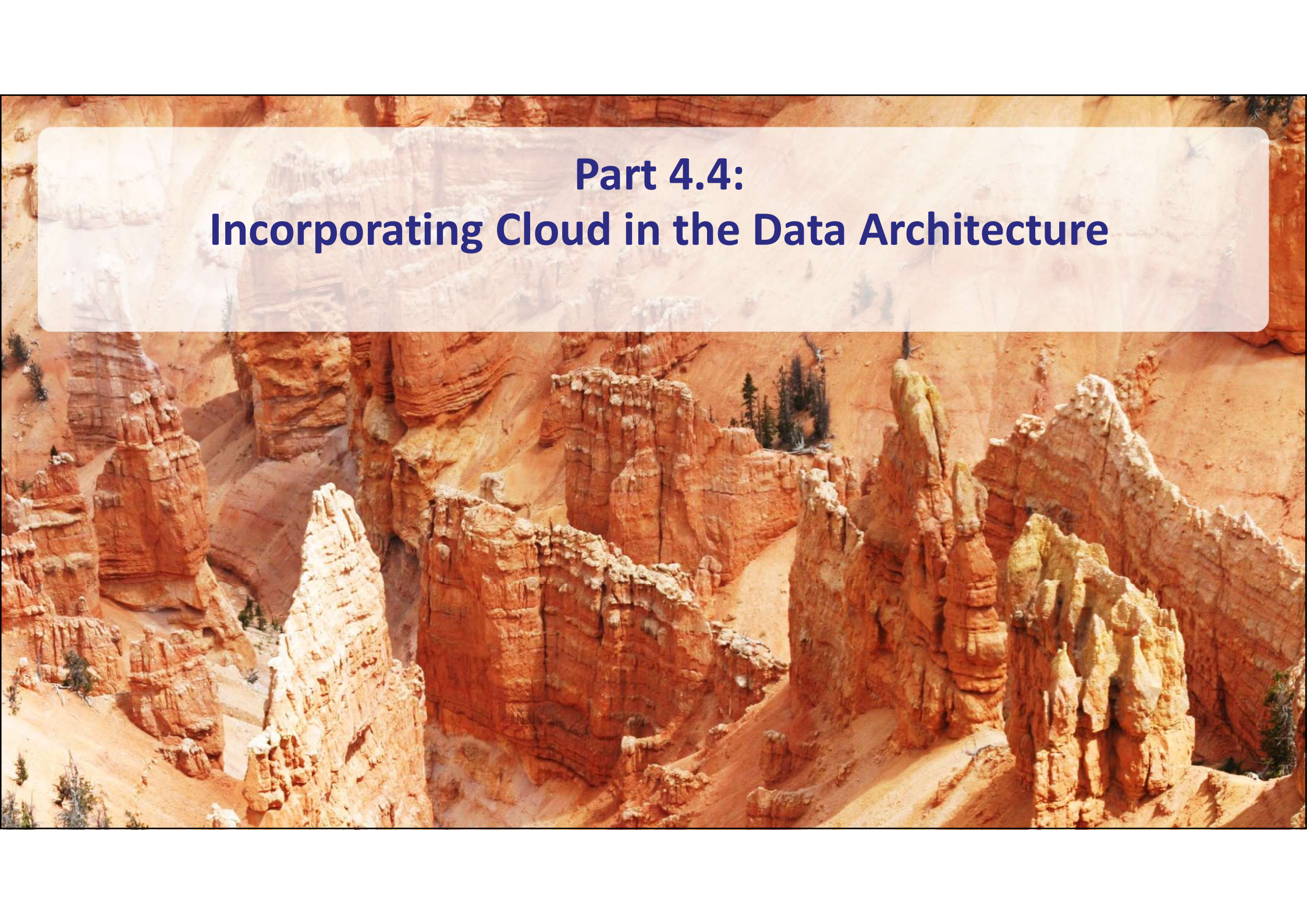


# Example: Lineage Through Scraping by ASG



# Example: Lineage Through Scraping by SQLdep (Collibra)





## **Part 4.4: Incorporating Cloud in the Data Architecture**

# Cloud Platforms are Becoming the New Mainframes



# Mainframe = Lock In

---



- Proprietary operating systems
- Proprietary system management software
- Proprietary database servers
- Proprietary security systems
- Proprietary development environments
- Proprietary JCLs
- Proprietary ...

# Cloud Platform = Lock In?

---



- Proprietary operating systems
- Proprietary management software
- Proprietary database servers
  - E.g. Amazon: RDS, RedShift (SQL), S3, ...
- Proprietary security systems
- Proprietary development environments
  - E.g. Microsoft Azure: Reporting Services, Analytics services, Data Management Services, ...
- Proprietary ...

# Data Storage Technologies Available on Cloud Platforms





Cloud Platform	Data Storage Technology
Amazon AWS	Aurora DocumentDB DynamoDB Elasticache for Redis RDS Redshift S3 Timestream
Google	BigQuery Cloud Bigtable Cloud Firestore Cloud Spanner Cloud SQL
Microsoft Azure	Cache for Redis Cosmos DB Data Lake SQL Database Synapse Analytics

**Stay Cloud Platform Independent  
(IT sovereignty)**

**=**

**Design to Migrate**

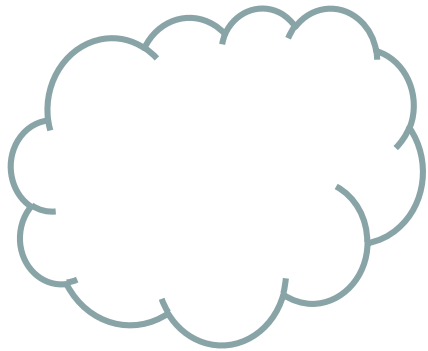
# Watch Out For Egress Costs!

	Public Cloud	Typical Data Egress Charge (per GB)	Cost to move 10 TB, per month	Discounted Data Egress Charge (per GB) for 100 TB	Cost to move 100 TB, per month
	Azure	\$0.08	\$800	\$0.07	\$7,000
	AWS	\$0.02	\$200	\$0.02	\$2,000
	Google Cloud Platform	\$0.11	\$1100	\$0.08	\$8,000
	Oracle	Free up to 10TB	free	\$0.0085-\$0.050 depending on geography	\$850 to \$5,000

Source: <https://www.factioninc.com/blog/it-challenges/egress-charges-how-to-prevent-costs/>

# Cloud Platform Fees

---



- Fees can have influence on data architecture
- Example:
  - SnowflakeDB: pay for data usage (queries)
    - Store more derived data
  - Exasol: pay for environment size (queries for free)
    - Work with views in stead of physical data marts
- How well can the technology exploit the cloud platform?
  - E.g. cloud is endless MPP, what about the database server?
- Pushing processing into the cloud, close to where data is produced

A photograph of a massive, layered rock formation in a desert landscape. The rock is reddish-brown and shows distinct horizontal strata. The formation is set against a clear blue sky with a few wispy white clouds. The foreground is a sandy, rocky plain with sparse, dry vegetation. A semi-transparent white box with rounded corners is overlaid on the upper portion of the image, containing the text.

**Part 5:**  
**Steps 1-3: Setting the Stage**

A large, rugged rock formation in a desert landscape under a blue sky with light clouds. The rock is reddish-brown and has a layered, weathered appearance. The foreground is a rocky, sandy area with sparse green vegetation.

## **Part 5.1:**

# **Step 1: Determine Business Motivations**

# Poor Examples of Business Motivations

---



- Change insights and requirements
- Deployment of self-service BI
- Optimization of existing data architecture
- The platform on which the current BI system is hosted externally is old and needs to be replaced
- Move to the cloud
- Data science is not very well supported by current data warehouse environment
- We want to do more with the data we have, but it's hard to get to it

# Proper Business Motivations

---



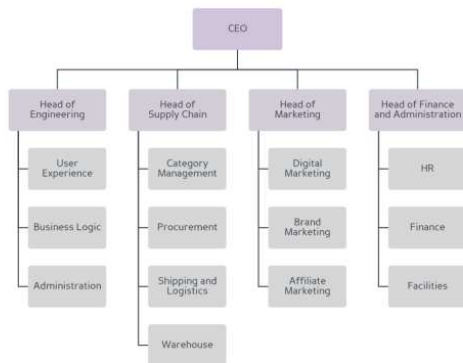
- Competitive improvement
  - Improving reaction speed to customer requests
- Support for customer journey and valuestream
- New business model
  - Allow customers real-time access to data
- Lower costs of specific business processes to improve margin
- Organization under threat
  - New competitor
- Comply with new laws and regulations
  - E.g. GDPR, CCPA, PSD2

# Challenges, Wishes and Requirements

---

- Speeding up development of *cross-system analysis systems*
- Speeding up development of *cross-organization analysis systems*
- Simplifying the *synchronization of source systems*
- Increasing data processing transparency for *process/decision reconstruction*
- Increasing the *discoverability* and clarity/meaning of data
- Improving *data quality*
- Strengthening *data security* and *data privacy*
- Strengthening *data history*
- Handling unstructured data: video, sound, text, and images
- Simplifying the data processing landscape
- Smarter uses of new technology
- ...

# Business Strategy and Data Strategy



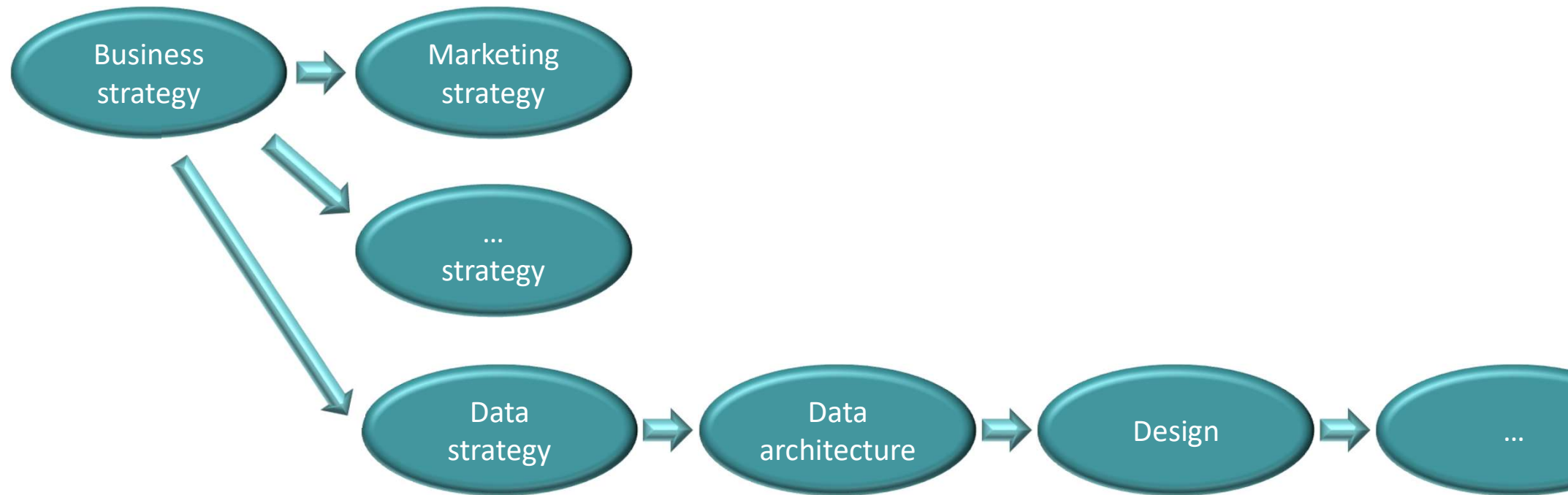
## ■ Business Strategy

- The challenges of top executives
  - New regulations, competitors, ...
- The main concerns for current business processes
- Future business developments
  - New business domains

## ■ Data Strategy

- New data architecture has to “fit” the data strategy
- New demands for data delivery

# From Strategy to Data Architecture and Onwards



A photograph of a massive, layered rock formation in a desert landscape. The rock is reddish-brown and shows distinct horizontal strata. The foreground is a rocky, sandy area with sparse green and yellow vegetation. The sky is bright blue with a few wispy white clouds. A semi-transparent white box with rounded corners is overlaid on the top half of the image, containing the text.

## **Part 5.2: Step 2: Determine New Requirements**

# Determine New Requirements (1)



- New analytical functionality
- Lower latency for reports
- More users
- More access to metadata
- Migration to cloud platform
- More data
- More transparency of architecture
- Better security
- Deployment of data science
- ...

## Determine New Requirements (2)



- Reconstruction of processes and decisions
- Data-shop to discover and describe data
- Migrate/redevelop source systems
- Minimize data copies
- Sharing data with other organizations
- Making data AI-ready
- ...

# Determine Constraints

---



Photo: Martin Sanchez

- Laws and regulations
  - GDPR, CCPA, PSD2, ...
- Budget restrictions
- Software limitations
  - One-stop shopping, open source preferred, company preferences, ...
- Hardware limitations
  - No easy processing, memory or storage scalability, ...
- Current legacy systems
  - Mainframe-based, proprietary applications, plain old, out-of-date/obsolete development environments
- Internal ICT skills

A photograph of a massive, layered rock formation in a desert landscape. The rock is reddish-brown and shows distinct horizontal strata. The foreground is a rocky, sandy area with sparse green and yellow vegetation. The sky is bright blue with a few wispy white clouds. A semi-transparent white box with a blue border is overlaid on the top half of the image, containing the text.

**Part 5.3:**  
**Step 3: Analyze the Existing Environment**

# Determine Current ICT Bottlenecks

---



- Performance
- Report latency
- Productivity - backlog
- Functionality
- Costs too high
- Business – ICT cooperation
- Non-professional IT organization
- Not IT savvy
- ...

# Analyze Existing Applications

---

## ■ Data producers

- Can we access the database directly or through an API?
- Current workload?
- Homemade or application?

## ■ Data transformers and transporters

- Home-made or professional (e.g. ETL, bus, data virtualization)?
- Implementation style?

## ■ Data Consumers

- Homemade?
- Internal or external?

# Analyze Technology and Products in Use

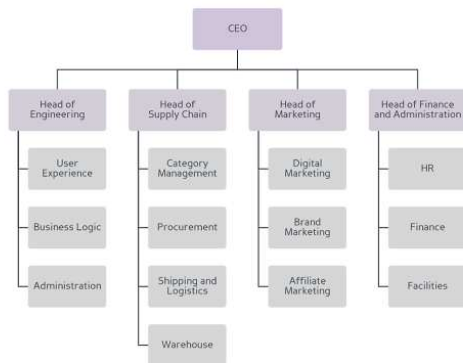
---



Photo: Maxine Rossignol

- Selected products and versions
- Selected (cloud) platforms
- License costs
- Infrastructure
- Potential migration challenges

# Determine the Culture of the IT Organization



- Traditional?
- Risk evasive?
- No experience with modern technologies?
- Cynical towards new developments?

# Determine IT Maturity Level of Organization (1)



Photo: Shridhar Gupta

## ■ Data processing checks

- Is data primarily stored to support business processes and to conform to reporting regulations?
- Can DBAs see the data?
- Are ETL processes started manually?
- Is ETL crash automatically fixed?
- Are transformers scattered across all modules?
- Is metadata available and kept up to date?
- Are “old” reports reproducible?

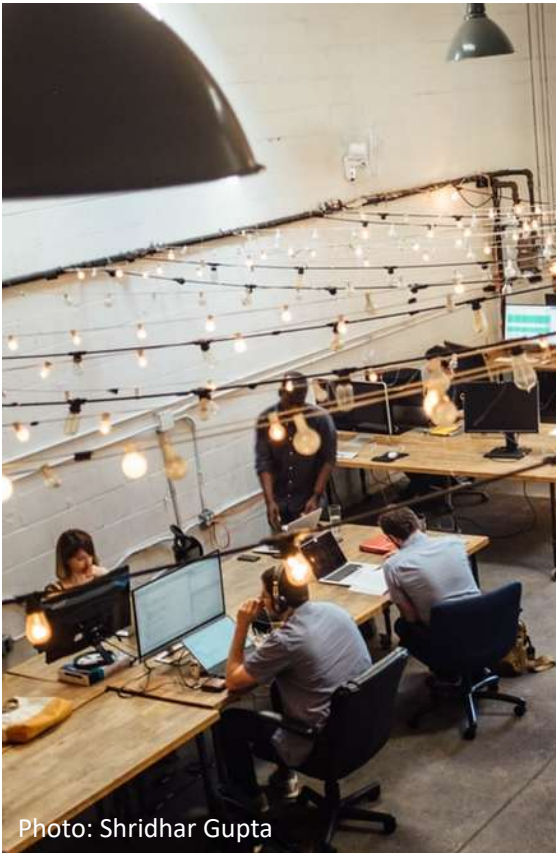
# Determine IT Maturity Level of Organization (2)



- Data consumption
  - Do reports primarily show what has happened within business processes?
  - High data latency?
  - Do they use predictive analytics to optimize business processes and decision-making processes?
- Data management
  - Ownership of data assigned?
  - Is there focus on data quality?
  - Are there procedures in place to fix incorrect data?

# Determine IT Maturity Level of Organization (3)

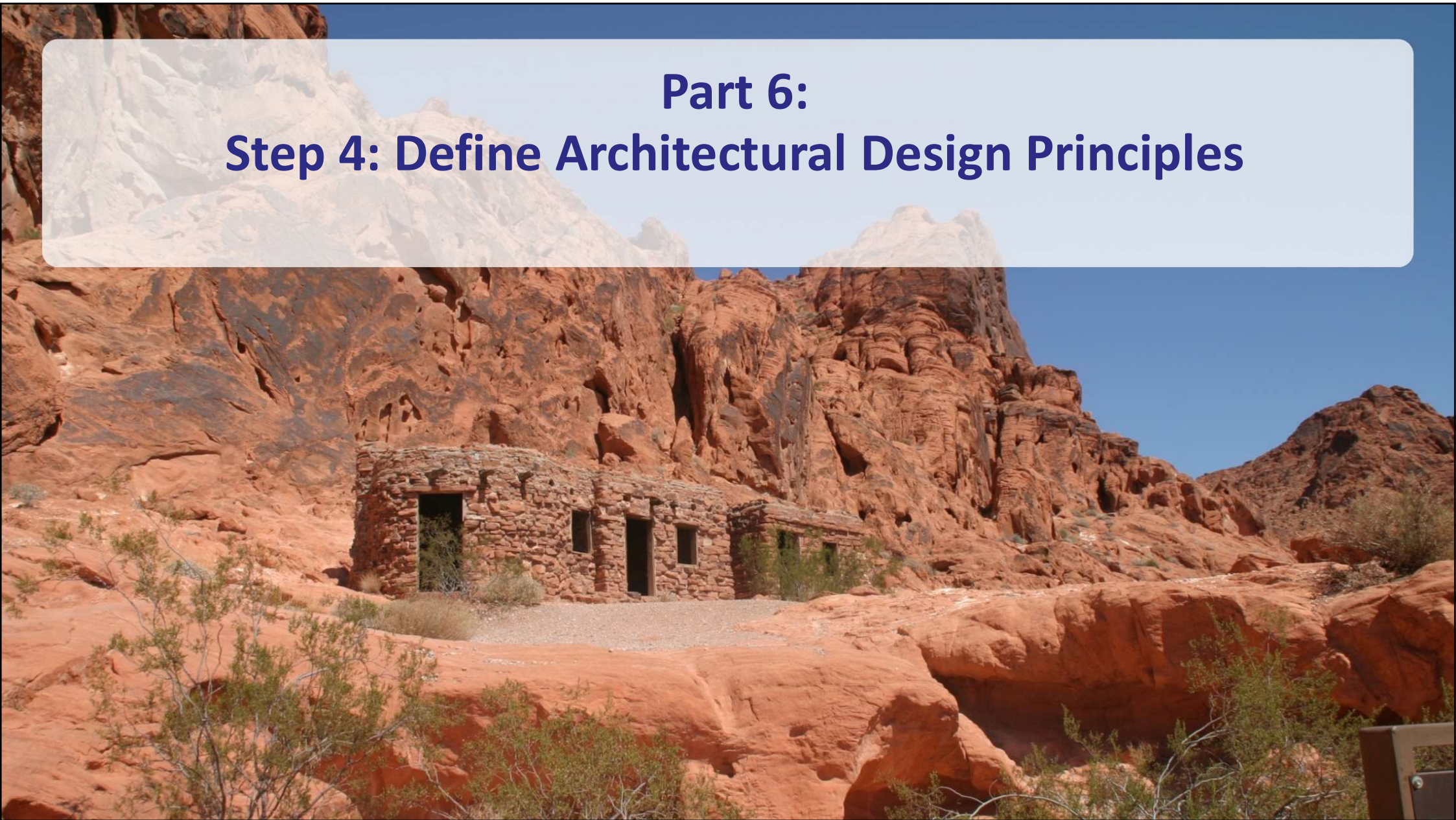
---



## ■ ICT skills

- All development outsourced?
- Many tool-jockeys?
- Performance anxiety?
- Minimal knowledge of new technologies?

**Part 6:**  
**Step 4: Define Architectural Design Principles**



# Forget Old Architectural Design Principles

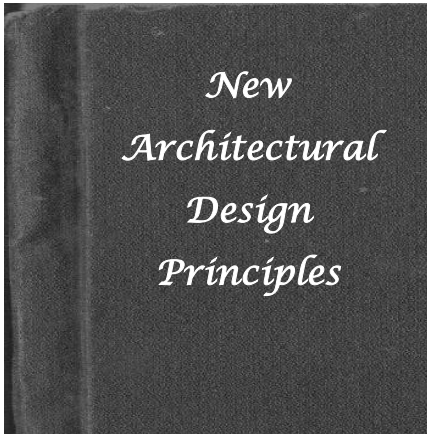
---



- No reporting on the production database
  - Reporting and transaction workloads clash
- *Physical* data marts are needed to improve reporting performance
- Data marts need a star schema design to speed up analytical queries
- ETL is used to transform data
  - Batch oriented
- When SQL databases are used
  - Indexes are required to improve query performance
  - Use locking for concurrency management
  - Not ideal for MPP
  - Need constant tuning by DBA
- ...

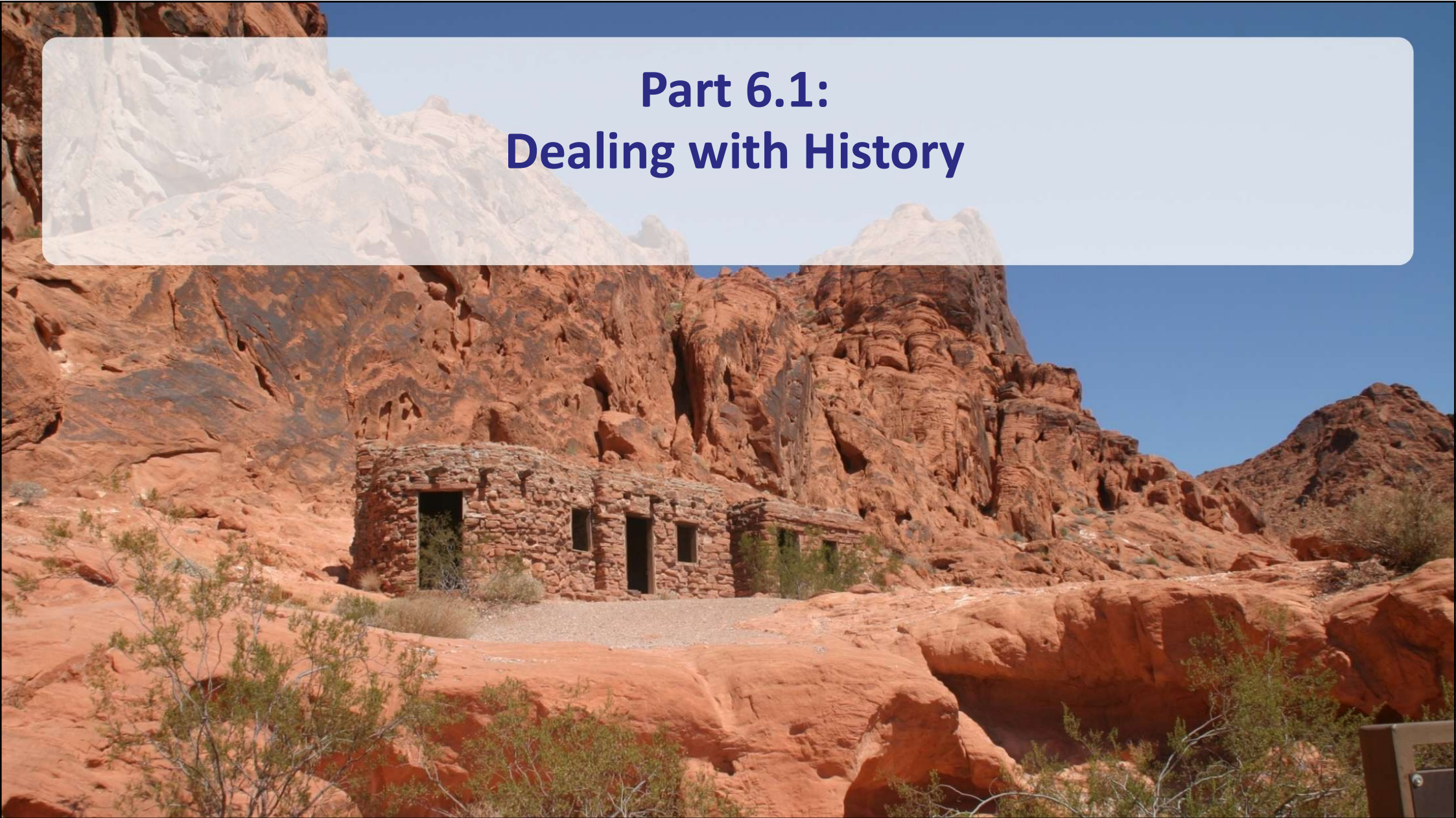
# Examples of Architectural Design Principles

---



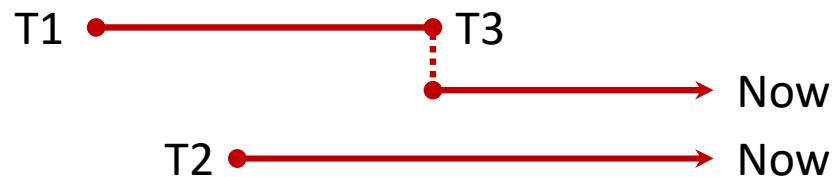
- Centralized and active transformers
  - Searchable definitions and descriptions for technical and business users
  - Lineage and impact analysis
- One universal architecture for all forms of data consumption
  - Standard reporting, self-service BI, apps, data science, ...
- Data storage and access technology agnostic
  - Hadoop, SQL, cubes, ...
  - Abstraction
- Push the processing to the data, not the data to the processing
  - Decentralized data production
  - Edge analytics
  - Hyper-decentralized data production and storage
- Generator-driven
- ...

## Part 6.1: Dealing with History



# Modeling History: Simple History for Updates

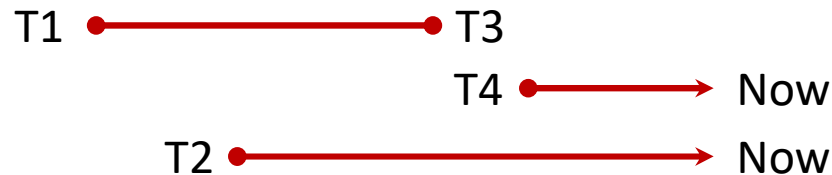
Cid	City	Start	End
C1	London	T1	T3
C1	Leicester	T3	Now
C2	Paris	T2	Now



- No gaps in history
- Only one value for an object on a specific datetime
- Supports following queries:
  - What is the current value – Where End = Now
  - What was the value on a specific datetime - Where date between Start and End

# Modeling History: Simple History for Updates with Gaps

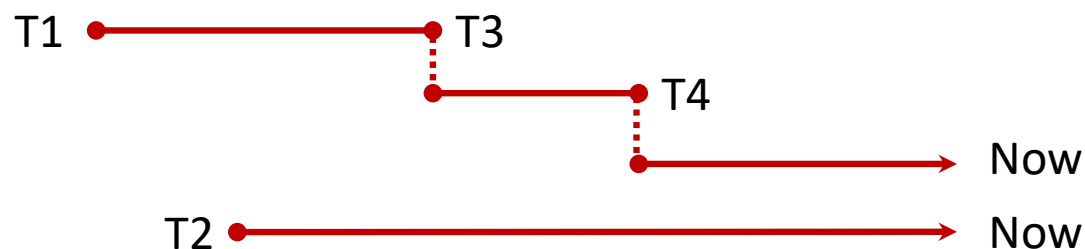
Cid	City	Start	End
C1	London	T1	T3
C1	Leicester	T4	Now
C2	Paris	T2	Now



- No gaps in history
- Only one value for an object on a specific datetime
- Supports following queries:
  - What is the current value – Where End = Now
  - What was the value on a specific datetime - Where date between Start and End - may return no values

# Modeling History: Simple History for Updates Without Gaps

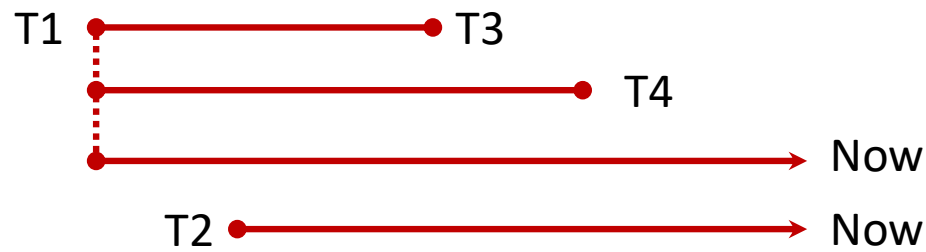
Cid	City	Start	End
C1	London	T1	T3
C1	Gap	T3	T4
C1	Leicester	T4	Now
C2	Paris	T2	Now



- No gaps in history
- Only one value for an object on a specific datetime
- Supports following queries:
  - What is the current value – Where End = Now
  - What was the value on a specific datetime - Where date between Start and End – always returns a value; sometimes nothing

# Modeling History: Corrections

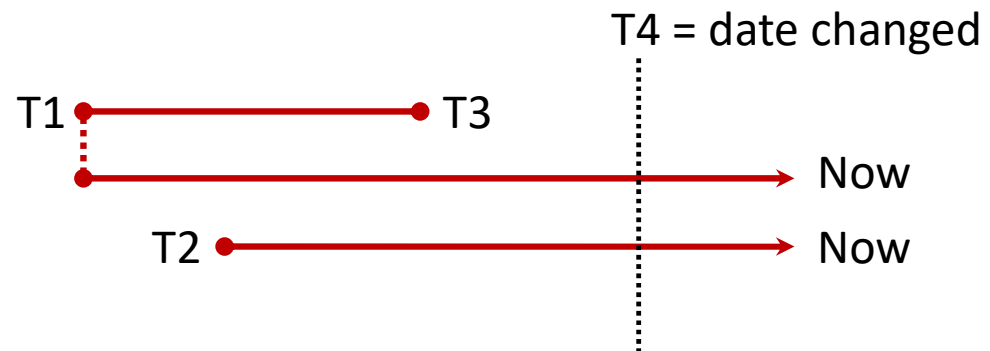
Cid	City	Start	End
C1	Londdon	T1	T3
C1	Londn	T1	T4
C1	London	T1	Now
C2	Paris	T2	Now



- No gaps in history
- Multiple values for an object on a specific datetime
- Supports following queries:
  - What is the current value – Where End = Now
  - What was the value on a specific datetime – Get the oldest where date between Start and End

# Modeling History: Delayed Corrections

Cid	City	Start	End	Changed
C1	Londdon	T1	T3	T4
C1	London	T1	Now	
C2	Paris	T2	Now	



- Extra column required
- No gaps in history
- Multiple values for an object on a specific datetime
- Supports following queries:
  - What is the current value – Where End = Now
  - What was the value on a specific datetime – Get the oldest where date between Start and End

# Modeling History: Logging Updates and Corrections

Cid	City	Start	End	Changed	Insert id	Change id
C1	London	T1	T2		Insert1	Update1
C1	Leicester	T2	Now		Insert2	
C2	Paris	T3	T4		Insert3	Update2
C2	Lyon	T4	T5		Insert4	Delete1

Change id	Who	When	Where	...
Insert1	User1	T1	...	...
Insert2	User2	T2	...	...
Insert3	User1	T3	...	...
Insert4	User3	T4	...	...
Update1	User2	T2	...	...
Update2	User4	T4	...	...
Delete1	User5	T5	...	...

- Log table for auditing purposes
- Batch inserts, updates, and deletes

# Modeling Streaming Data: Single Values

Incoming Stream
T1, S1, Temp=50
T2, S1, Temp=52
T3, S2, Temp=51
T4, S3, Temp=49
T5, S1, Temp=52;
T5, S2, Temp=53

Key	Start	End	Sensor	Temp	Avg Temp
1		T1	S1	50	50
2	T1	T2	S1	52	51
5	T2	T5	S1	52	51,3
3		T3	S2	51	51
6	T3	T5	S2	53	52
4		T4	S3	49	49

- Key is unique artificial value
- Measurement is considered as temperature since previous measurement
- Sensor data is arriving in the right order

# Modeling Streaming Data: Multiple Values

Incoming Stream
T1, S1, Temp=50
T2, S1, Temp=52
T3, S2, Temp=51
T4, S3, Temp=49
T5, S1, Temp=52; S2, Temp=53

Key	Start	End	Sensor	Temp	Avg Temp
1		T1	S1	50	50
2	T1	T2	S1	52	51
5	T2	T5	S1	52	51,3
3		T3	S2	51	51
6	T3	T5	S2	53	52
4		T4	S3	49	49

- Key is unique artificial value
- Stream records are flattened
- Measurement is considered as temperature since previous measurement
- Sensor data is arriving in the right order

# Modeling Streaming Data: Delta Values

Incoming Stream
T1, S1, Temp=50
T2, S1, Temp=+2
T3, S2, Temp=51
T4, S3, Temp=49
T5, S1, Temp=+0
T5, S2, Temp=+2

Key	Start	End	Sensor	Temp	Avg Temp
1		T1	S1	50	50
2	T1	T2	S1	52	51
5	T2	T5	S1	52	51,3
3		T3	S2	51	51
6	T3	T5	S2	53	52
4		T4	S3	49	49

- Key is unique artificial value
- Measurement is considered as change in temperature
- Sensor data is arriving in the right order

# Modeling Streaming Data: Log Data

Incoming Stream
T1, Insert, C1, London
T2, Update, C1, Leicester
T3, Insert, C2, Paris
T4, Insert, C3, Berlin
T5, Update, C1, Manchester
T5, Update, C3, Munich

Cid	City	Start	End
C1	London	T1	T2
C1	Leicester	T2	T5
C1	Machester	T5	Now
C2	Paris	T3	Now
C3	Berlin	T4	T5
C3	Munich	T5	Now

- Business key used
- Stream is seen as data entry
- Careful with parallel inserts; order not unimportant
  - Loading with hashed keys?

A photograph of ancient stone ruins built into a red rock cliffside in a desert landscape. The ruins consist of several rectangular structures with dark, rectangular openings, likely doorways or windows. The surrounding environment is arid, with sparse green shrubs and a clear blue sky. The rock formations are layered and textured, showing signs of erosion. A semi-transparent white box with rounded corners is overlaid on the top left of the image, containing the text 'Part 6.2: Supporting GDPR' in a bold, dark blue font.

## **Part 6.2: Supporting GDPR**

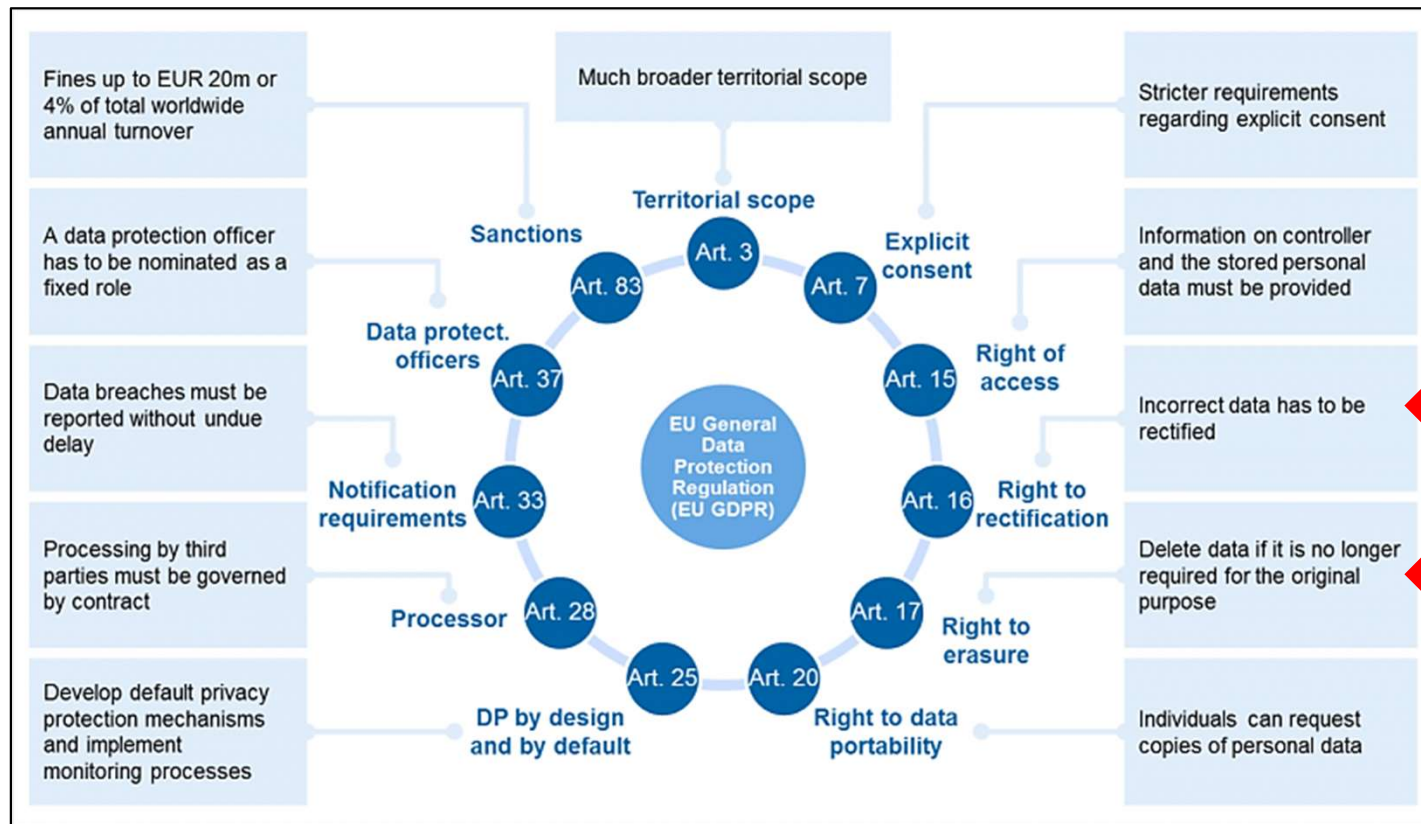
# GDPR – The Right to be Forgotten

---



- Art. 17 GDPR Right to erasure (right to be forgotten)
  - The rule primarily regulates erasure obligations
- According to this, personal data must be erased immediately where the data are no longer needed for their original processing purpose, or the data subject has withdrawn his consent and there is no other legal ground for processing, the data subject has objected and there are no overriding legitimate grounds for the processing, or erasure is required to fulfill a statutory obligation under the EU law or the right of the Member States.
- In addition, data must naturally be erased if the processing itself was against the law in the first place
- A data subject should have the right to have personal data concerning him or her rectified

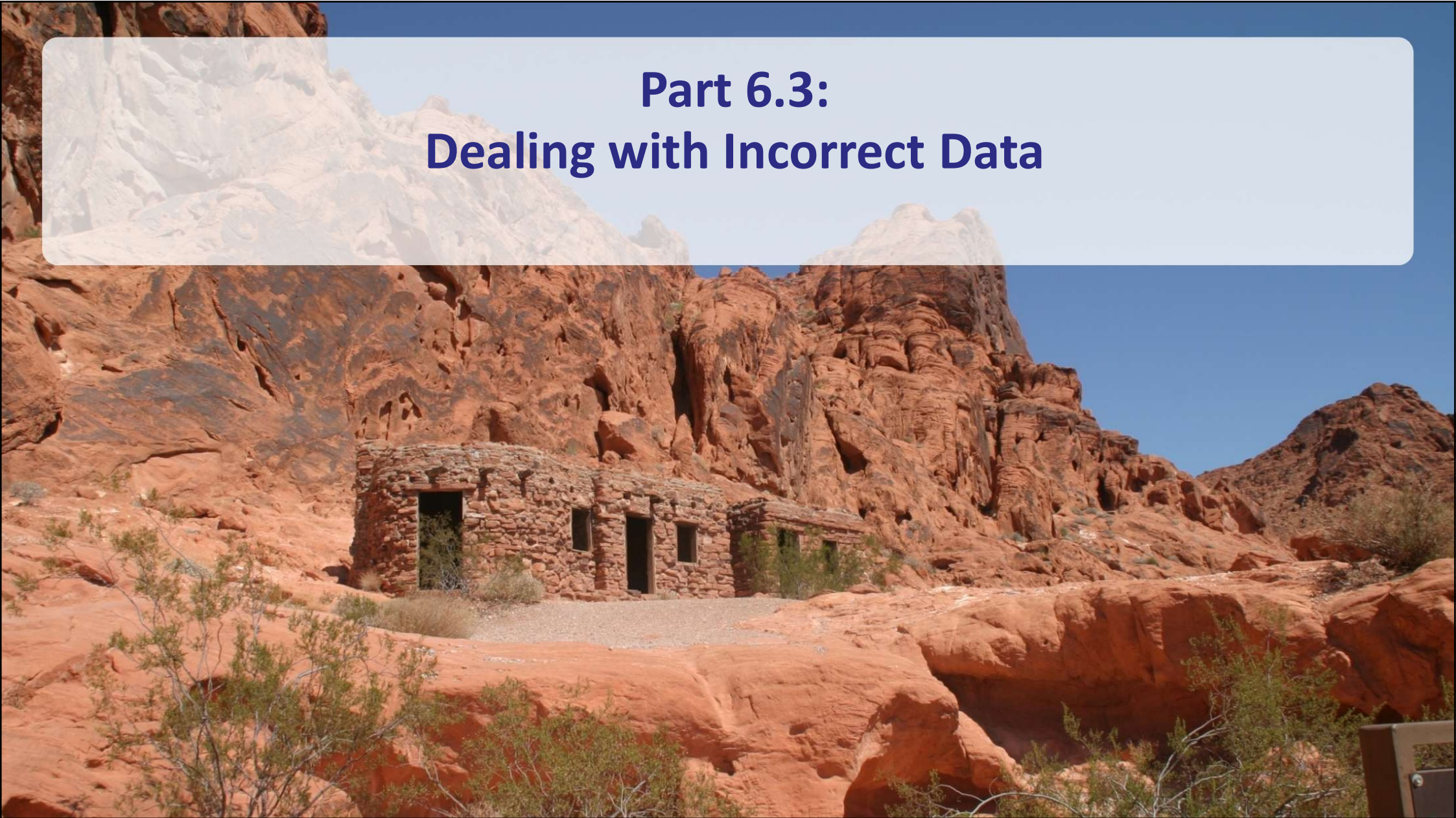
# Requirements of GDPR



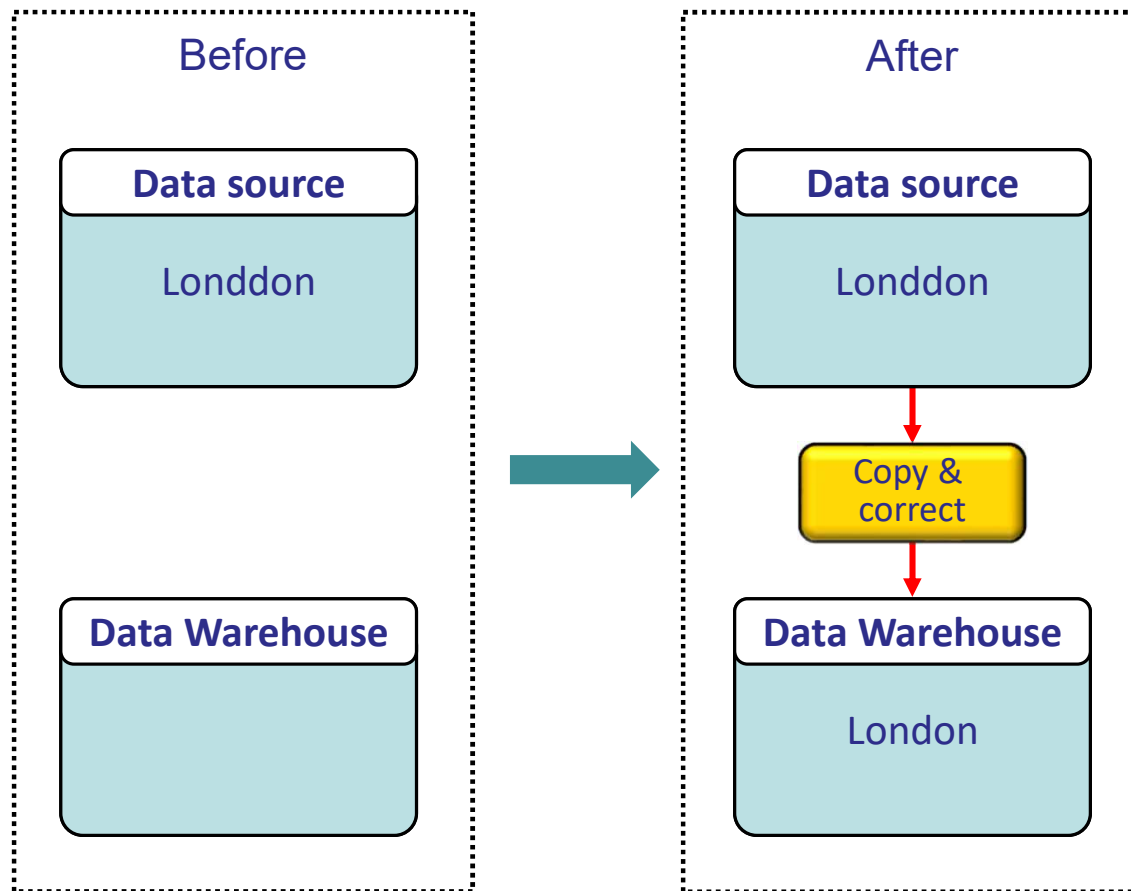
Source: Banking Hub, November 2017;

see <https://www.bankinghub.eu/banking/finance-risk/gdpr-deep-dive-implement-right-forgotten>

## Part 6.3: Dealing with Incorrect Data

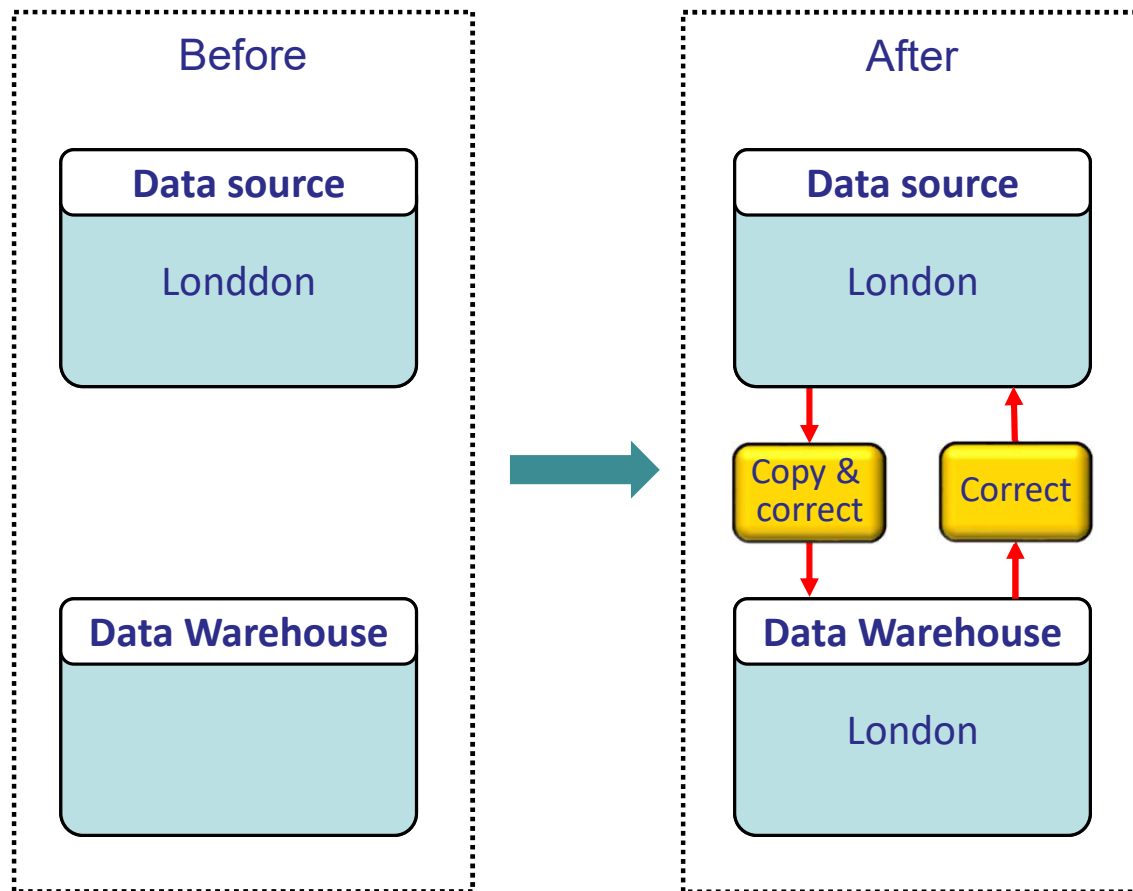


# Data Correction Strategies: Simple



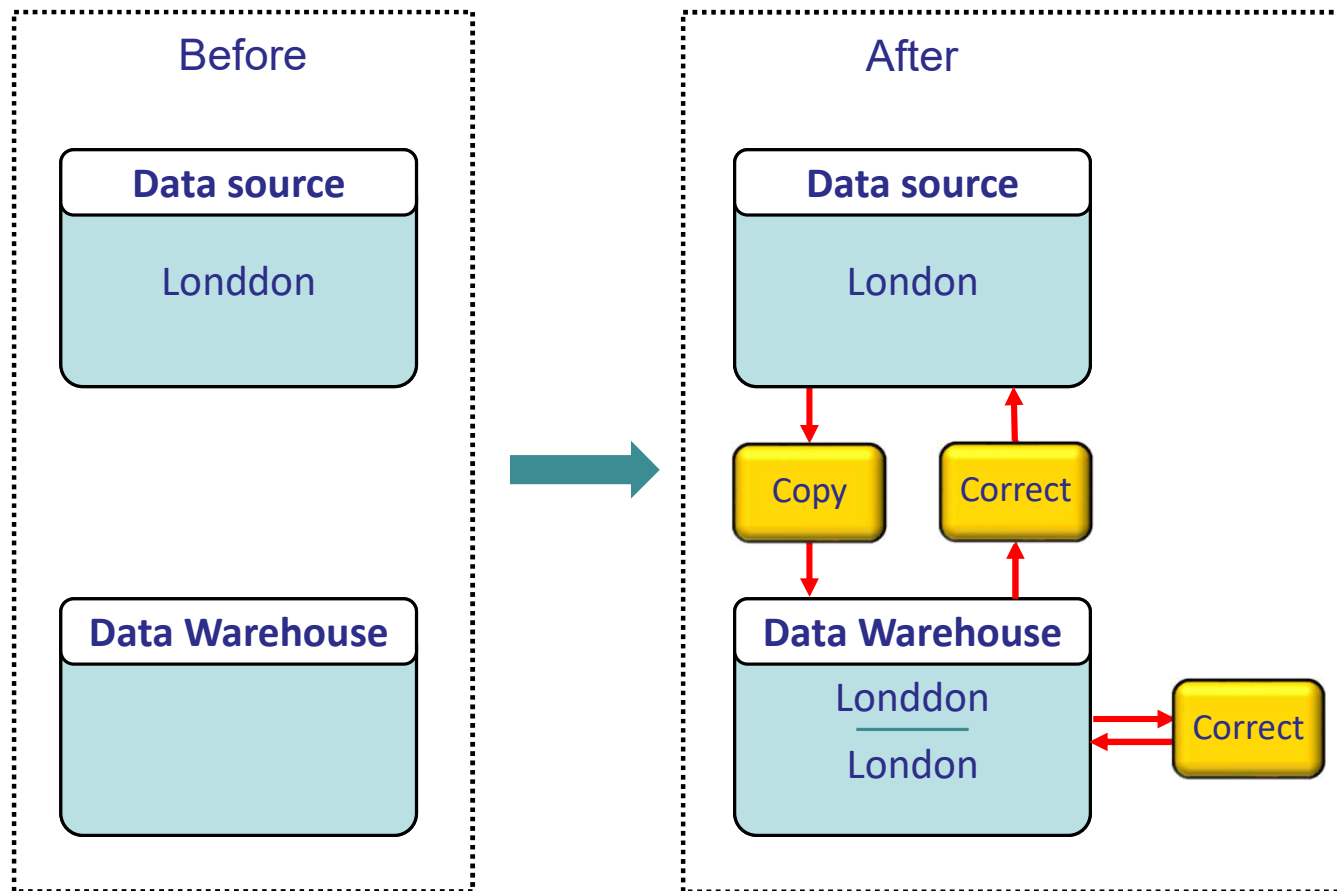
- Restricted to programmable cleansing operations
- Easy to implement
- Source doesn't benefit from cleansing
- Source and data warehouse inconsistent
- No impact on organization
- No time travel supported

# Data Correction Strategies: Synchronize



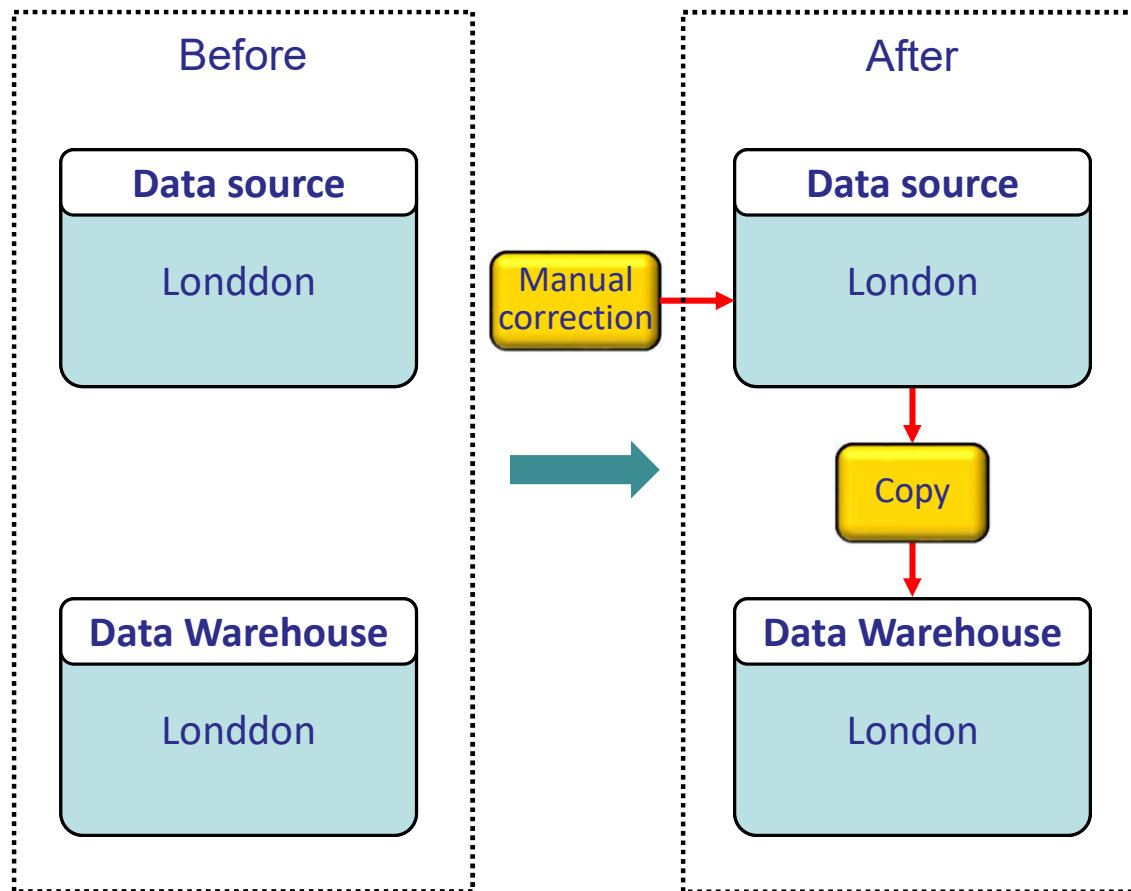
- Manual or automated process?
- Source benefits from cleansing
- Source and data warehouse consistent
- Impact on organization
- No time travel supported

# Data Correction Strategies: Time Travel



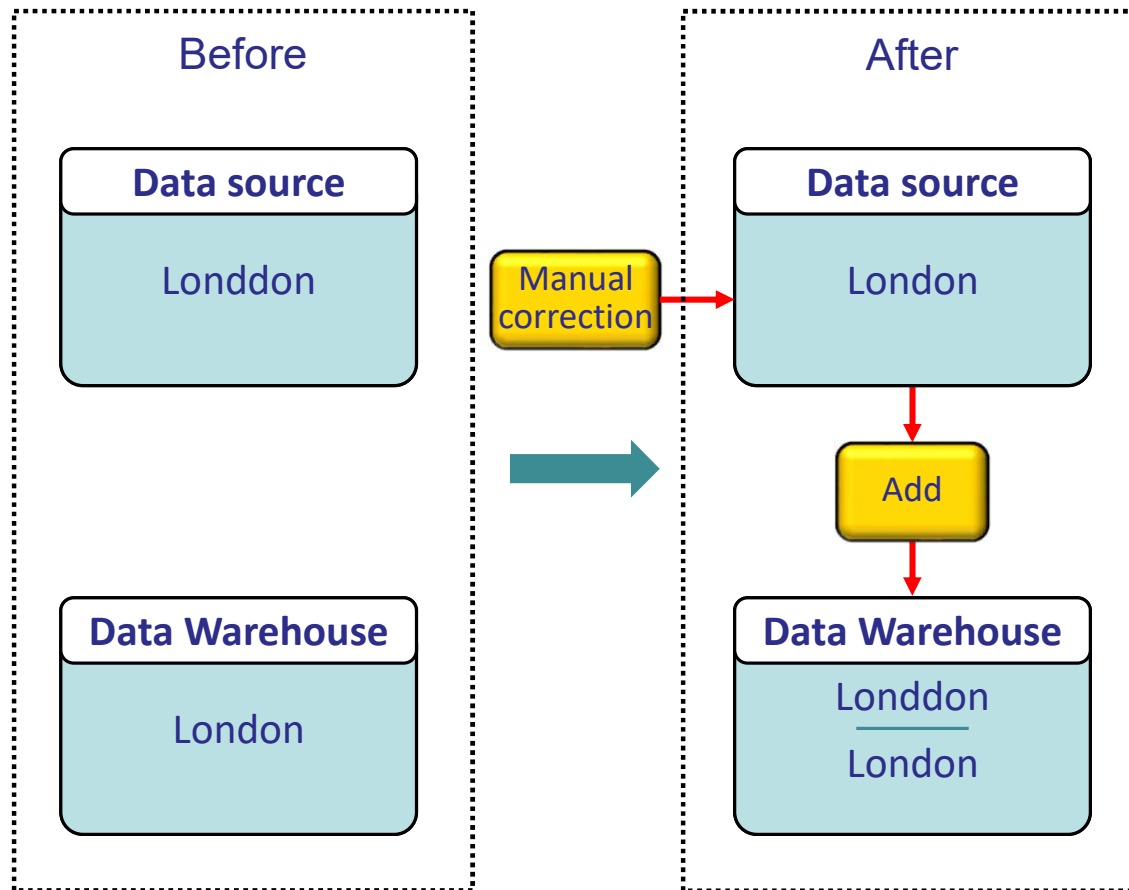
- Restricted to programmable cleansing operations
- Source benefits from cleansing
- Source and data warehouse consistent
- Time travel supported

# Data Correction Strategies: Manual Corrections



- User corrections
- Source benefits from cleansing
- Source and data warehouse consistent
- Impact on organization
- No time travel supported

# Data Correction Strategies: Manual Corrections + Time Travel



- User corrections
- Source benefits from cleansing
- Source and data warehouse consistent
- Impact on organization
- Time travel supported

**Part 7:**  
**Step 5: Select a Reference Data Architecture**



# Roadmap for Designing Data Architectures

---

1. Determine business motivations

2. Determine new requirements

3. Analyze the existing environment

4. Define architectural design principles

5. Select a reference data architecture

6. Design the new data architecture

7. Determine the implementation approach

8. Select new products and technologies

9. Introduce the data architecture within the organization

# Common Challenges

---

- Source data must be queryable
- Developers and data consumers can't find data easily
- Every insert, update, delete and query should be logged for reconstruction purposes and transparency
- Horizontal and operational lineage
- CRUD interface for real-time synchronization
- Centralized, reusable, versioned business logic
- Proper authorization when integrating data

# Part 7.1: The Classic Data Warehouse Architecture



# The Classic Data Warehouse Architecture

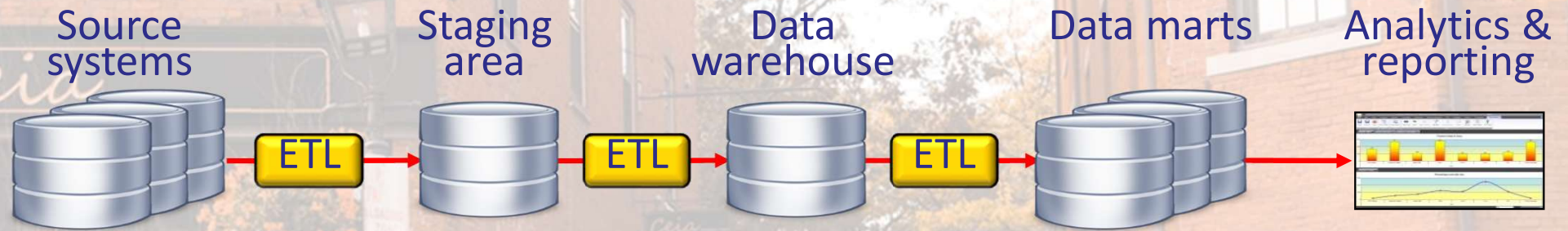
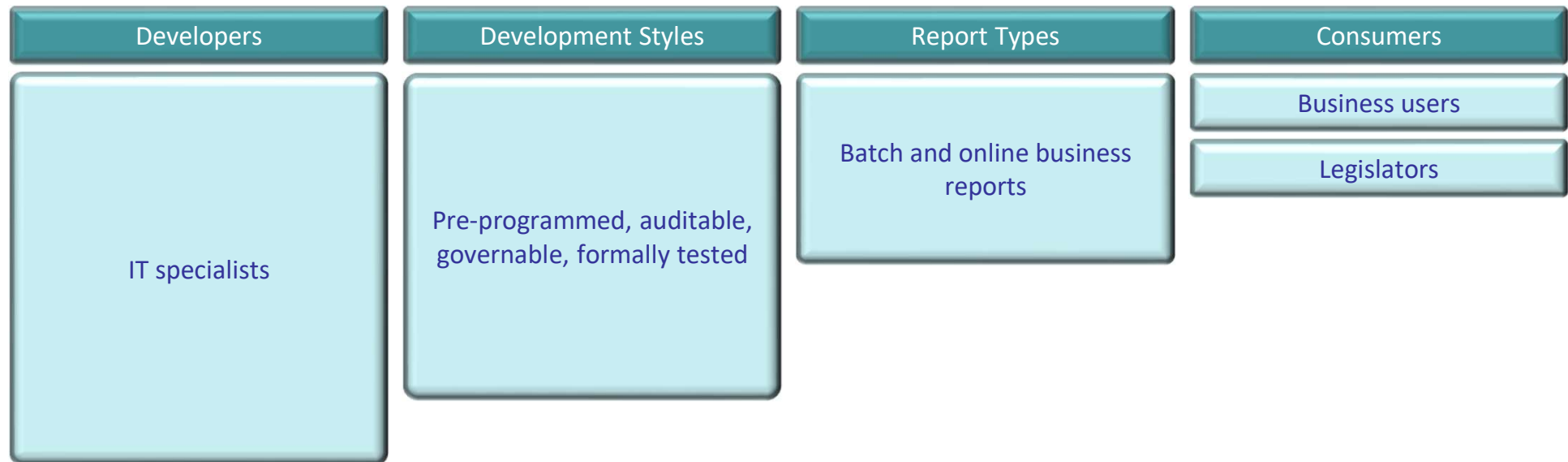


Photo: Alex Iby

# Yesterday: Data Warehouse and Data Consumption



# Today & Tomorrow: Data Warehouse and Data Consumption

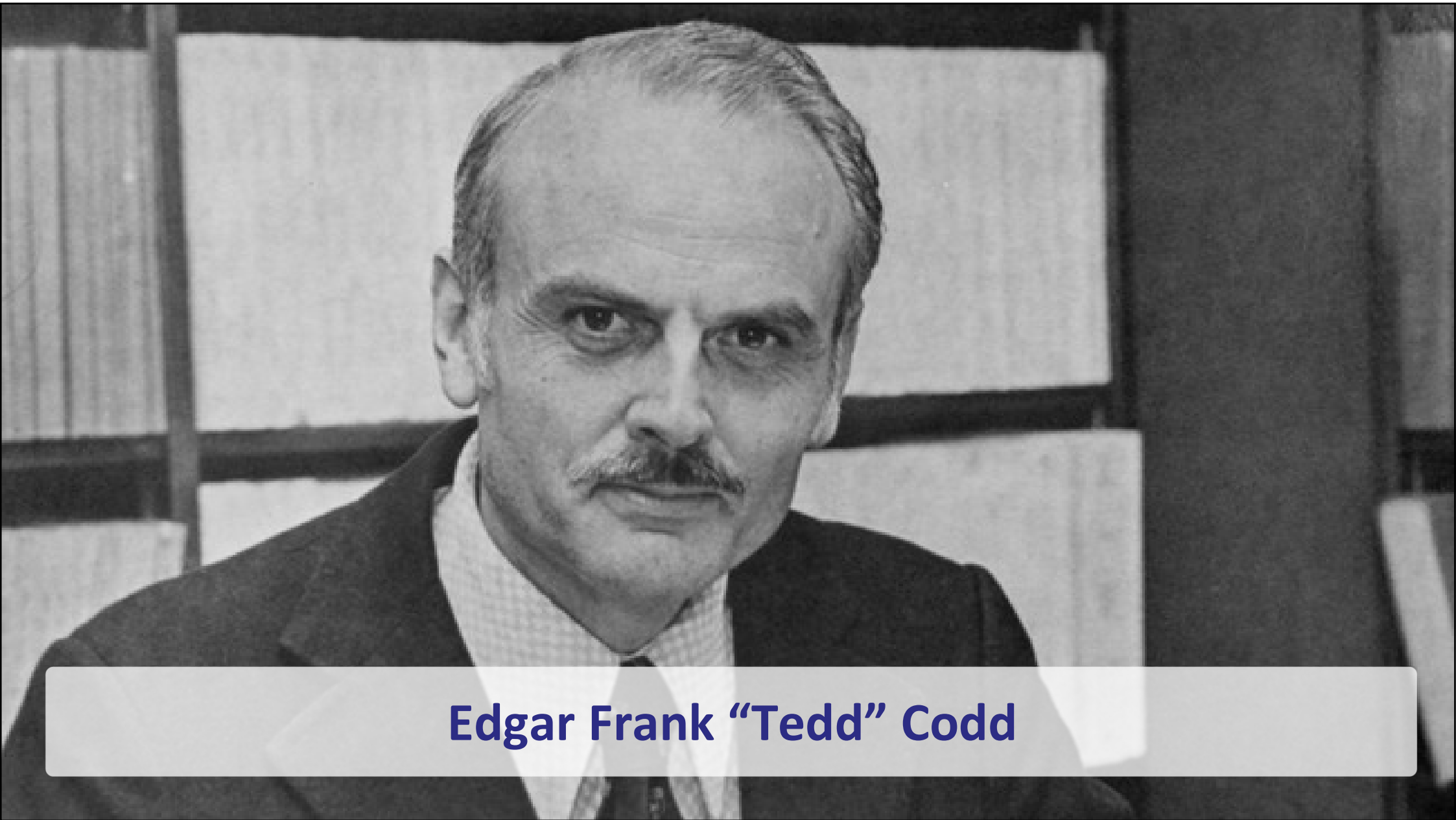
Developers	Development Styles	Report Types	Consumers
IT specialists	Pre-programmed, auditable, governable, formally tested	Batch and online business reports	Business users
		Customer-facing apps	Legislators
		Streaming analytics	External parties
		Ad-hoc reports	Consumers
Business Users	Self-service, investigative	Simple data retrieval	Business users, machines
	Pre-programmed	Ad-hoc reports	Business users
	Self-service, investigative	Ad-hoc reports	Business users
		Data mining, statistics	Data scientists
		Dark data analysis	Business users and IT

# The Classic Data Warehouse Architecture is Like a Rigid Assembly Line



## **Part 7.2: The Logical Data Warehouse Architecture**

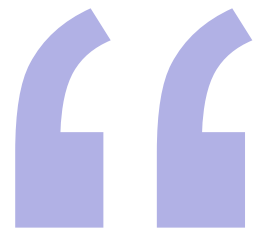




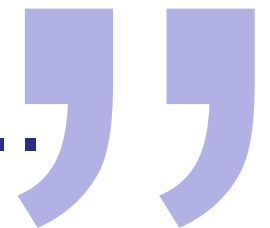
**Edgar Frank "Tedd" Codd**

## Ted Codd – June 1970

---



Future users of large data banks must be protected from having to know how the data is organized (...) application programs should remain unaffected when the internal representation of data is changed ...



Source: <https://cs.uwaterloo.ca/~david/cs848s14/codd-relational.pdf>

# Ted Codd on Data Independence

## The 1981 ACM Turing Award Lecture

Delivered at ACM '81, Los Angeles, California, November 9, 1981



The 1981 ACM Turing Award was presented to Edgar F. Codd, an IBM Fellow of the San Jose Research Laboratory, by President Peter Denning on November 9, 1981 at the ACM Annual Conference in Los Angeles, California. It is the Association's foremost award for technical contributions to the computing community.

Codd was selected by the ACM General Technical Achievement Award Committee for his "fundamental and continuing contributions to the theory and practice of database management systems." The originator of the relational model for databases, Codd has made further important contributions in the development of relational algebra, relational calculus, and normalization of relations.

Edgar F. Codd joined IBM in 1949 to prepare programs for the Selective Sequence Electronic Calculator. Since then, his work in computing has encompassed logical design of computers (IBM 701 and Stretch), managing a computer center in Canada, heading the development of one of the first operating systems with a general multiprogramming capability, contributing to the logic of self-reproducing automata, developing high level techniques for software specifica-

tion, creating and extending the relational approach to database management, and developing an English analyzing and synthesizing subsystem for casual users of relational databases. He is also the author of *Cellular Automata*, an early volume in the ACM Monograph Series.

Codd received his B.A. and M.A. in Mathematics from Oxford University in England, and his M.Sc. and Ph.D. in Computer and Communication Sciences from the University of Michigan. He is a Member of the National Academy of Engineering (USA) and a Fellow of the British Computer Society.

The ACM Turing Award is presented each year in commemoration of A. M. Turing, the English mathematician

## 2. Motivation

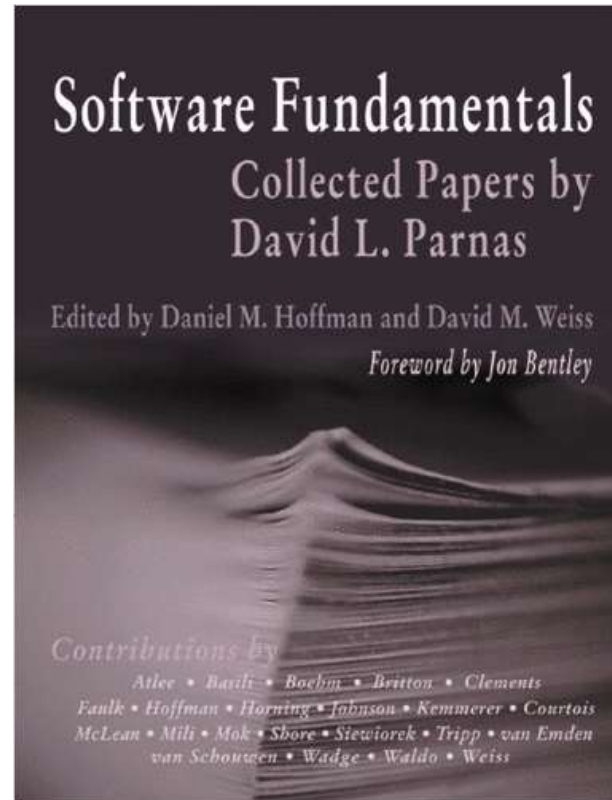
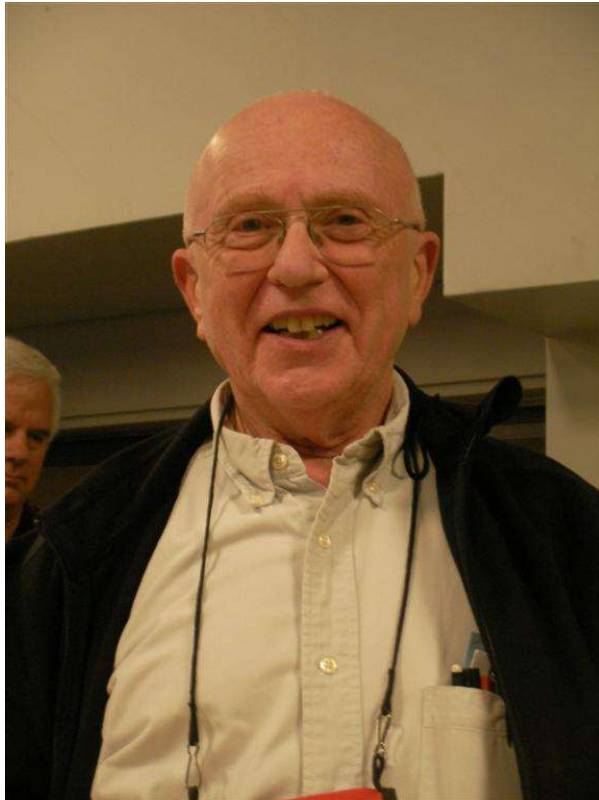
The most important motivation for the research work that resulted in the relational model was the objective of providing a sharp and clear boundary between the logical and physical aspects of database management (including database design, data retrieval, and data manipulation). We call this the *data independence objective*.

A second objective was to make the model structurally simple, so that all kinds of users and programmers could have a common understanding of the data, and could therefore communicate with one another about the database. We call this the *communicability objective*.

A third objective was to introduce high level language concepts (but not specific syntax) to enable users to express operations upon large chunks of information at a time. This entailed providing a foundation for set-oriented processing (i.e., the ability to express in a single statement the processing of multiple sets of records at a time). We call this the *set-processing objective*.

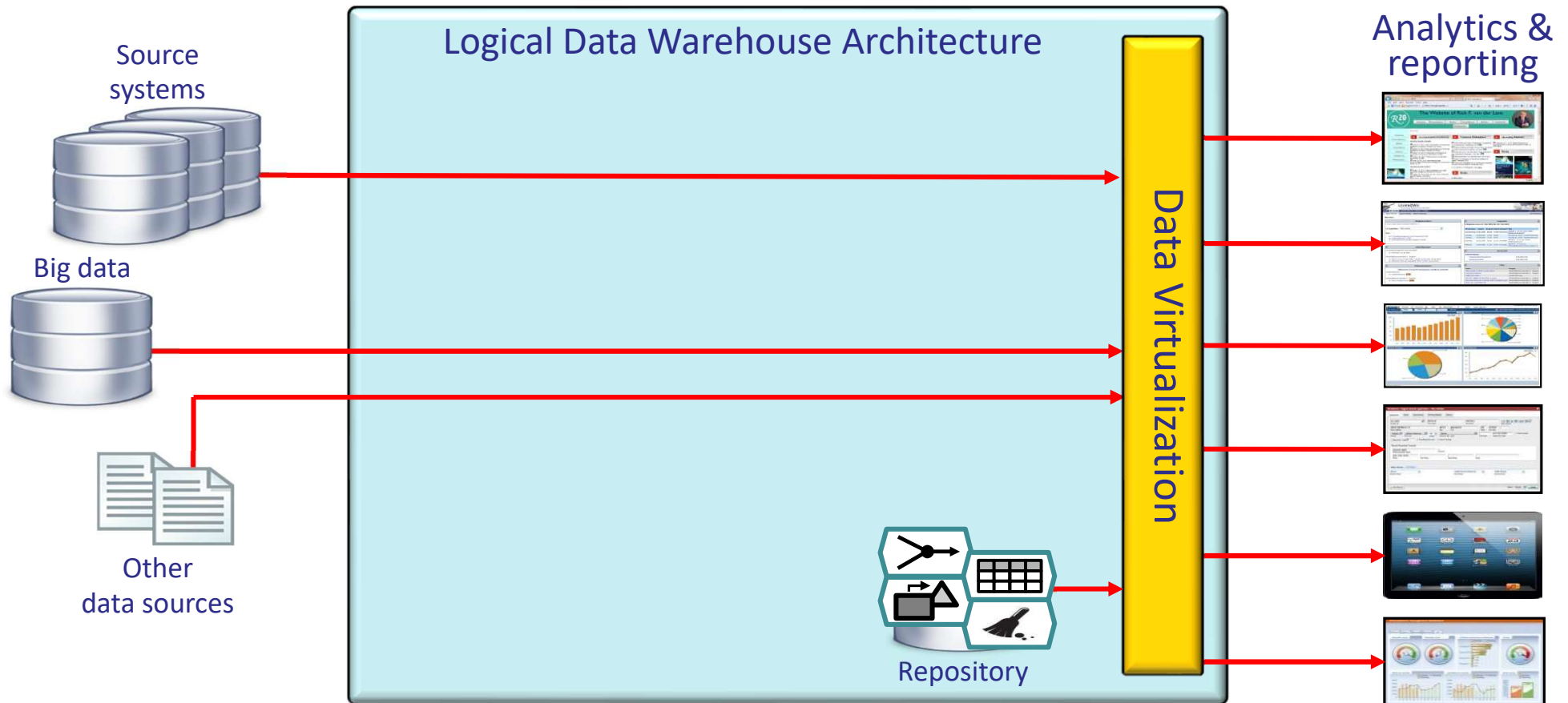
# David Parnas - Information Hiding - 1972

---

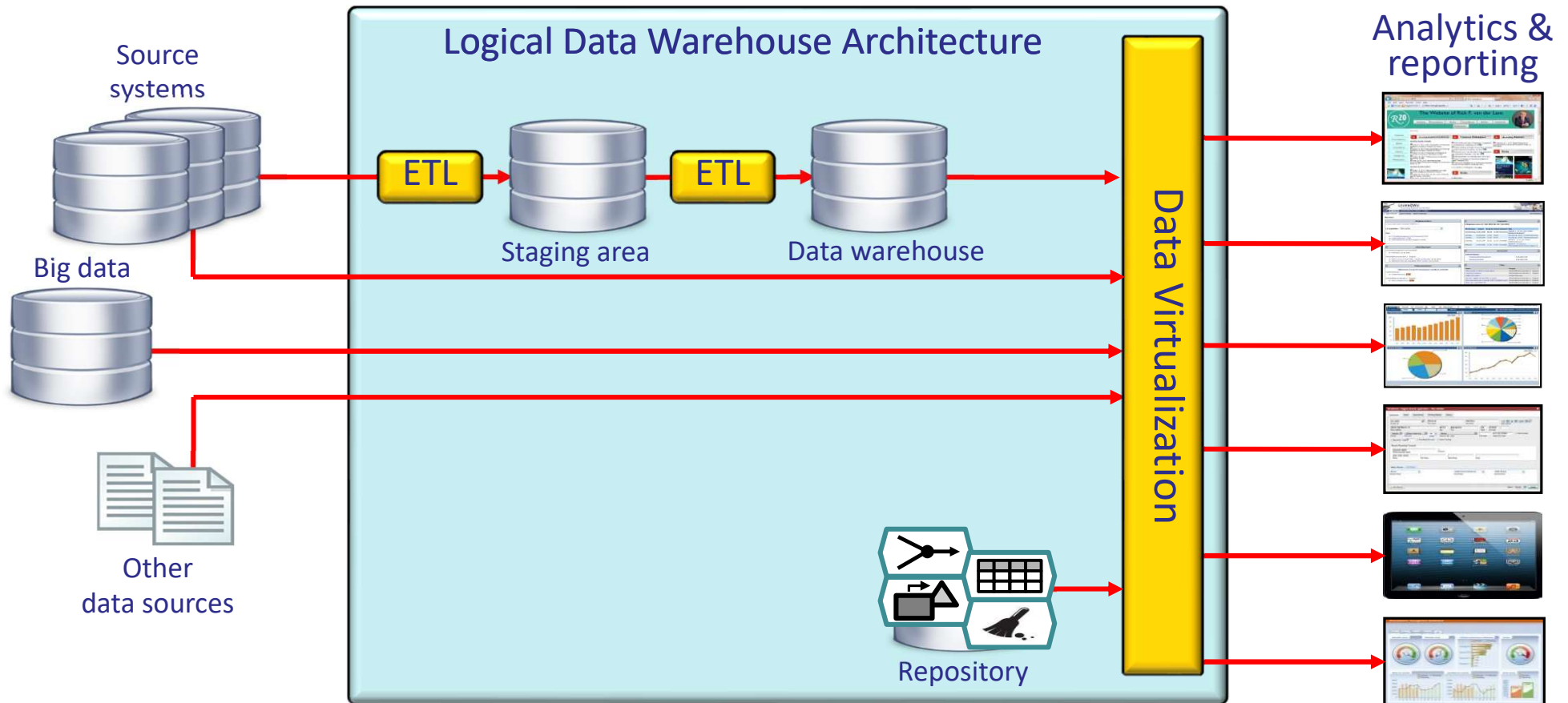


<http://www.cs.umd.edu/class/spring2003/cmsc838p/Design/criteria.pdf>

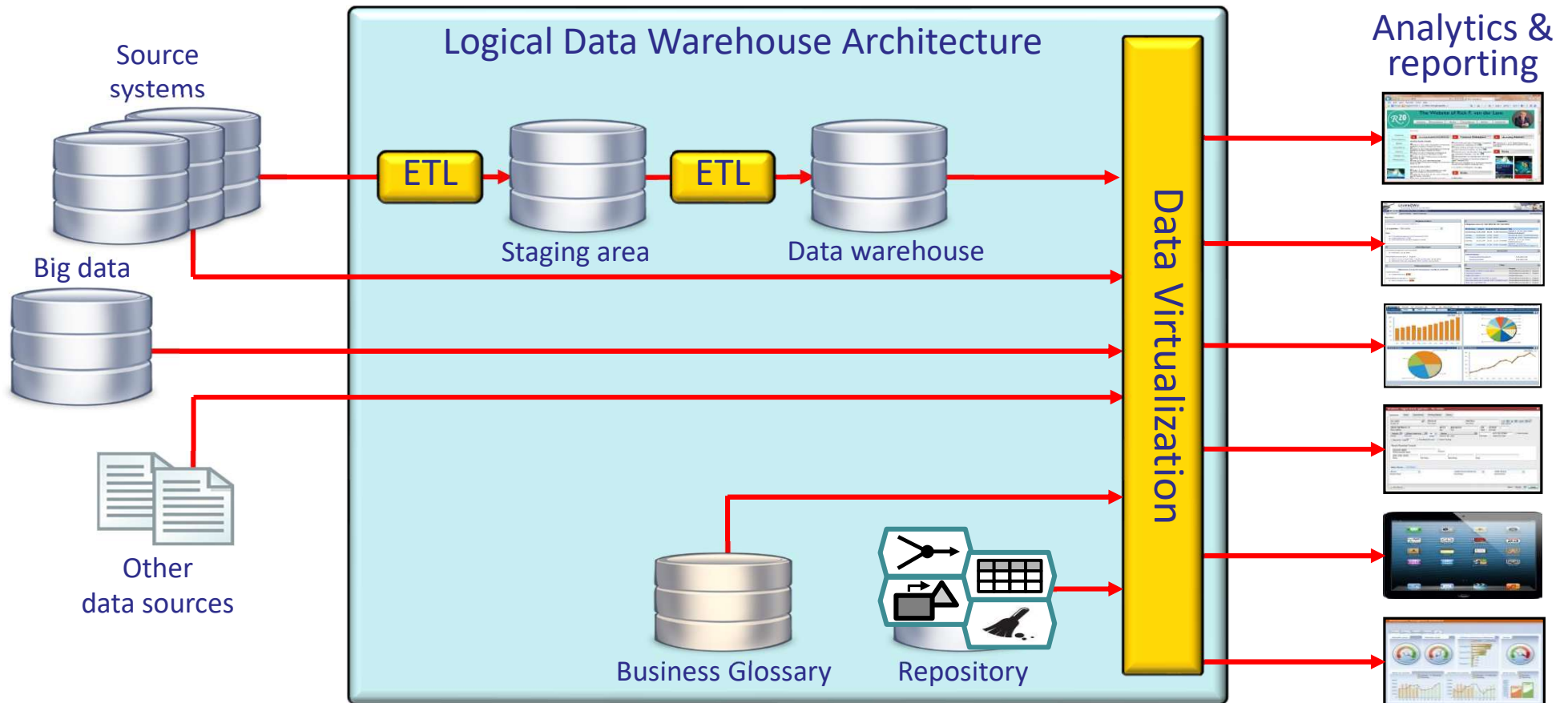
# The Logical Data Warehouse Architecture (1)



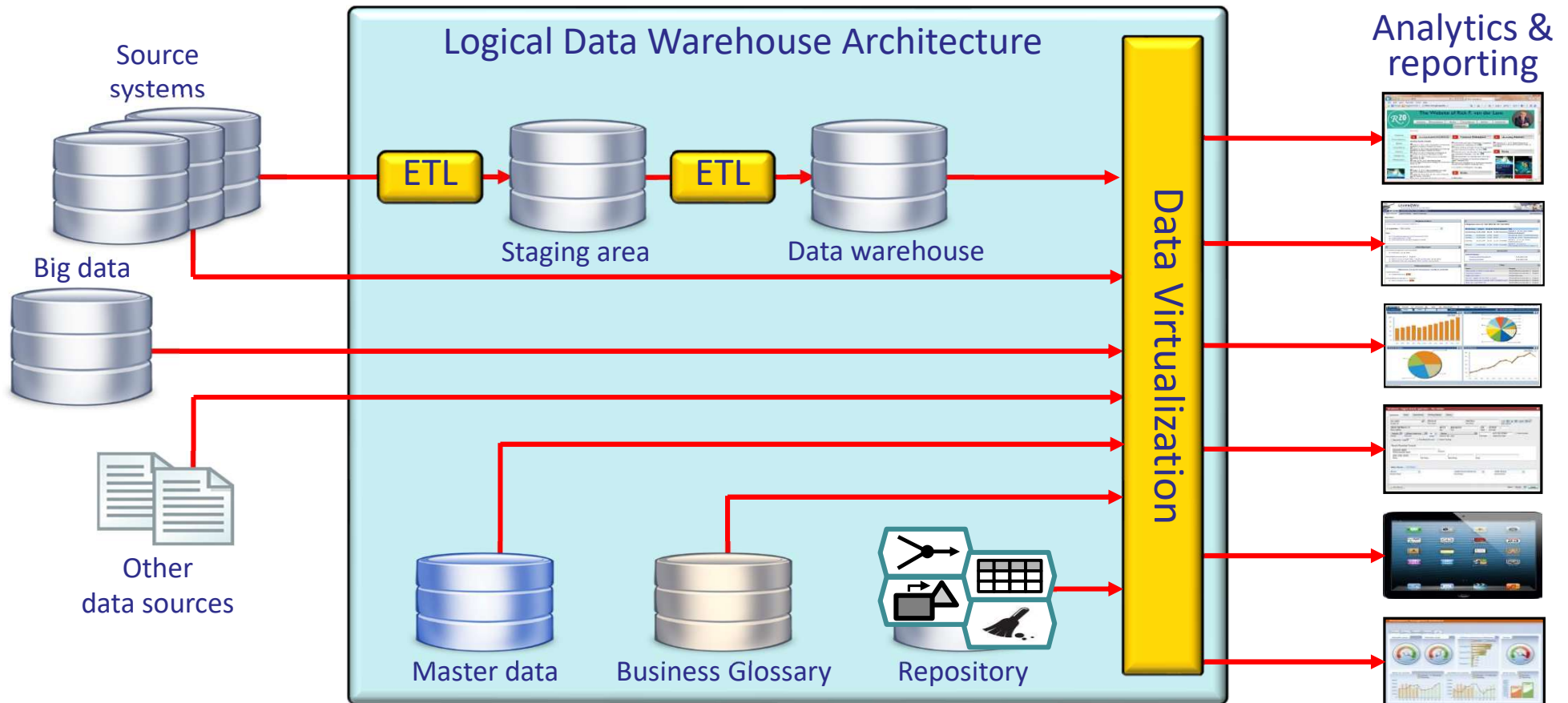
# The Logical Data Warehouse Architecture (2)



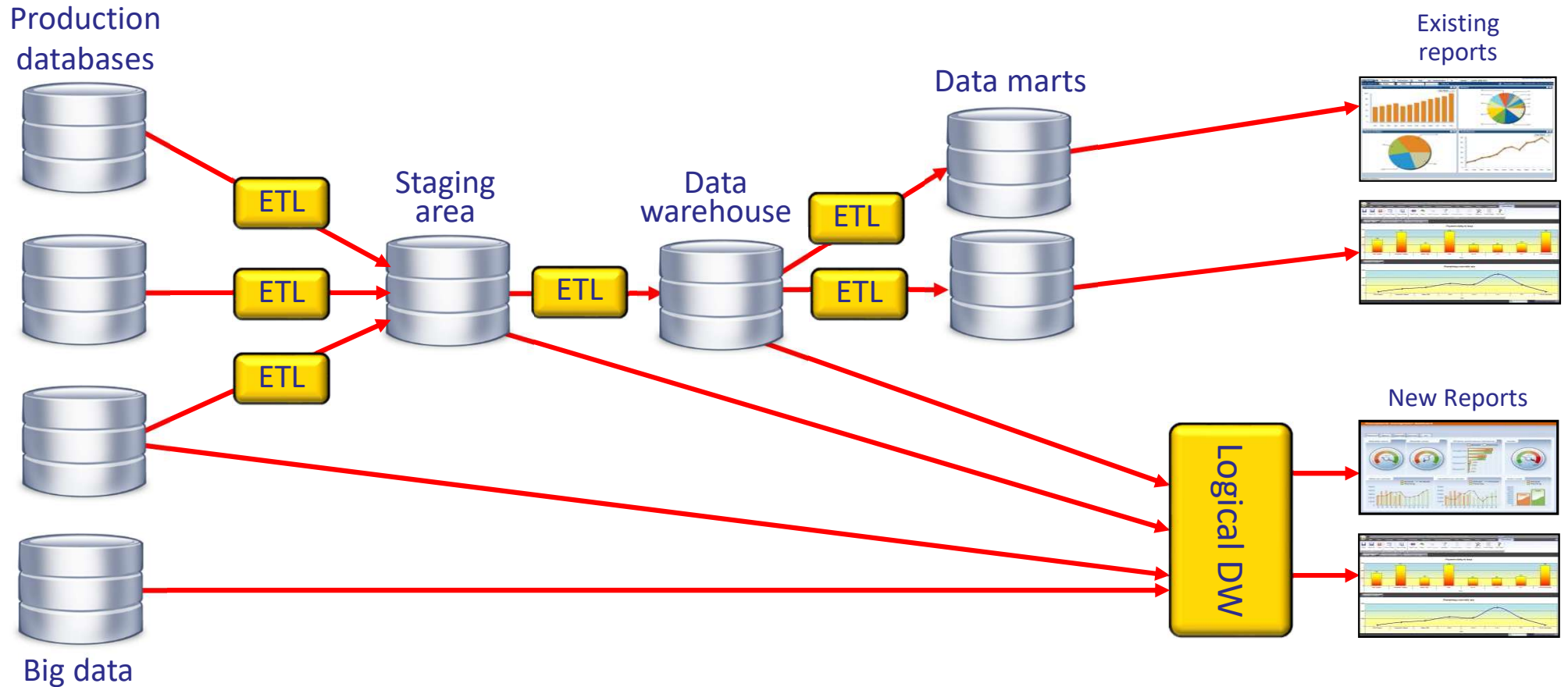
# The Logical Data Warehouse Architecture (3)



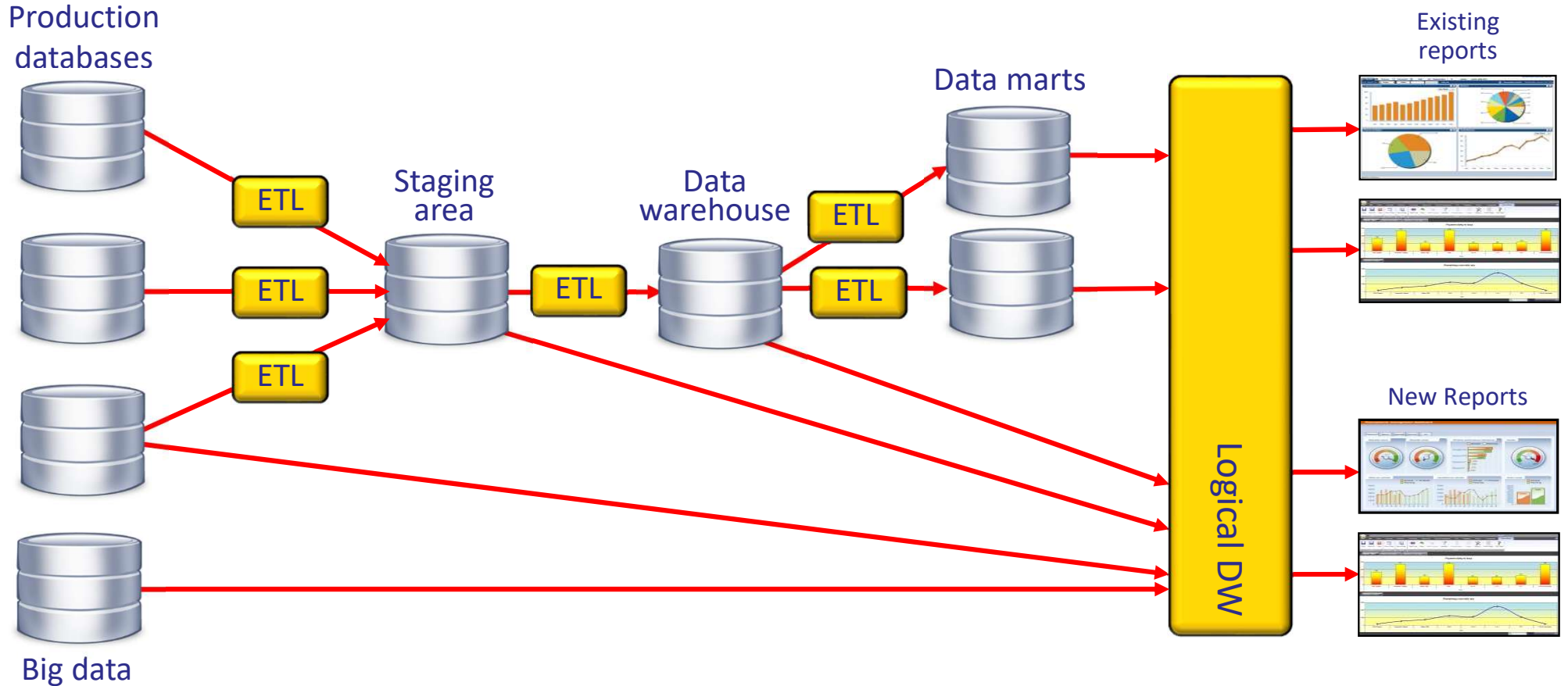
# The Logical Data Warehouse Architecture (4)



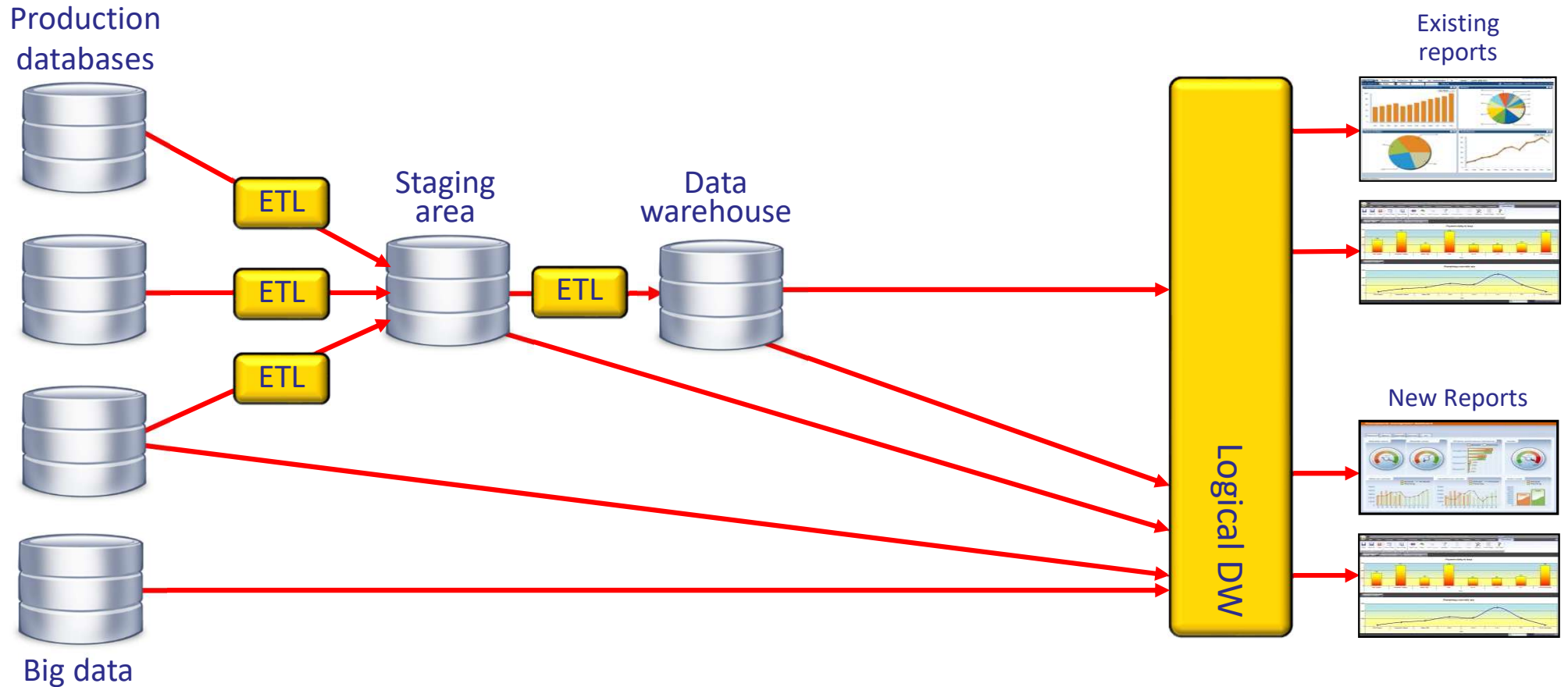
# Wrap the Old Data Warehouse (1)

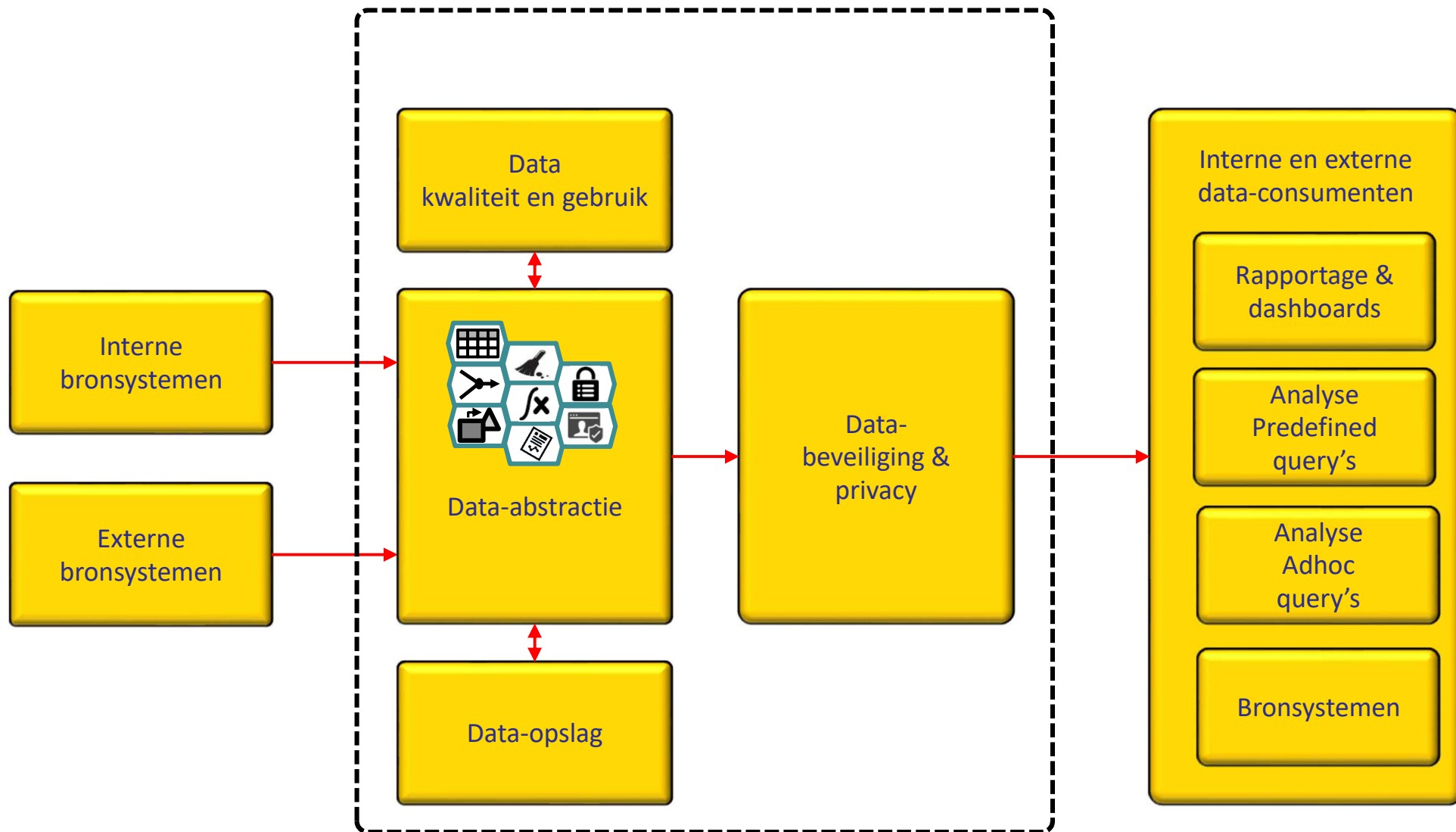


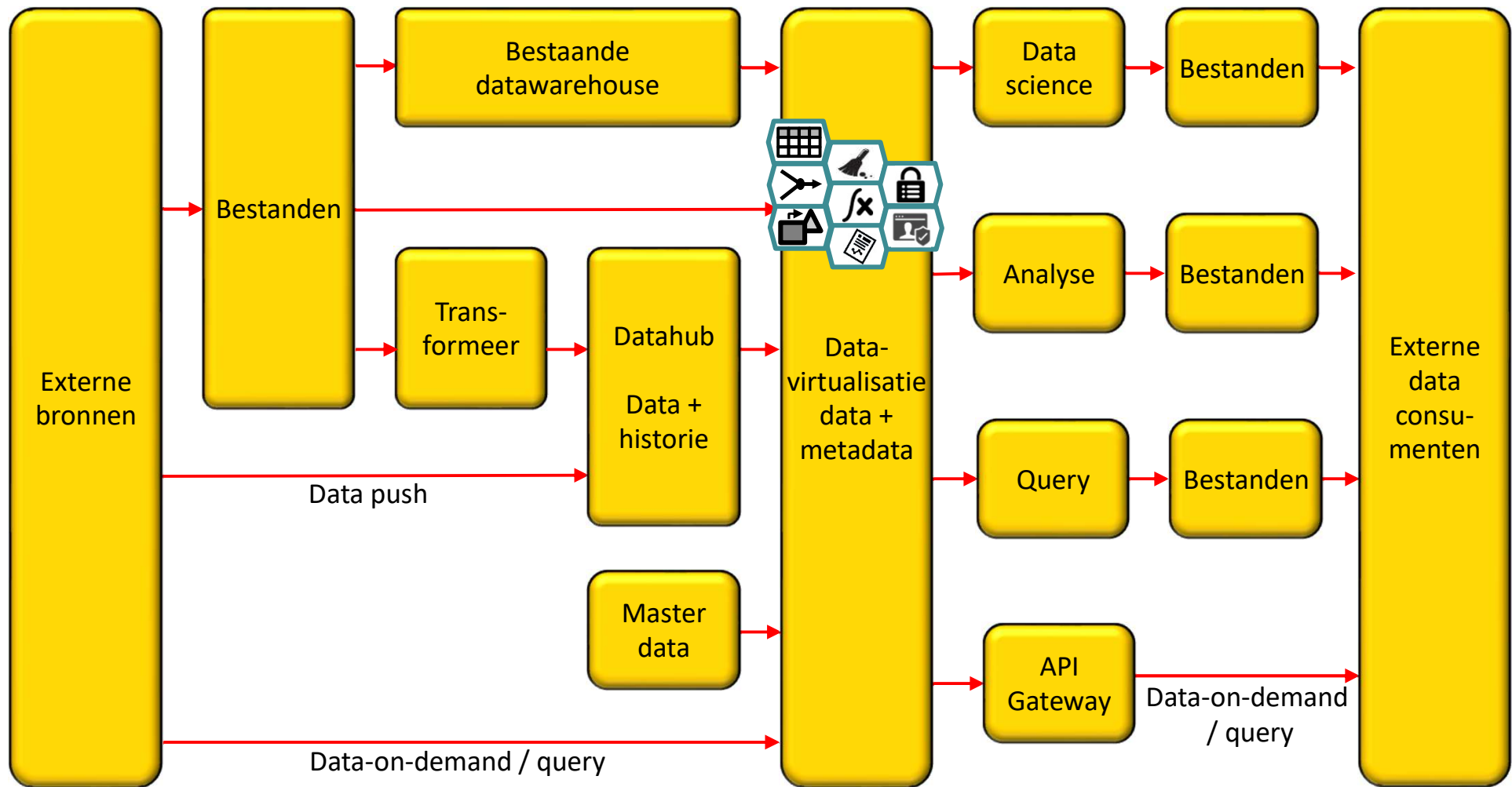
# Wrap the Old Data Warehouse (2)

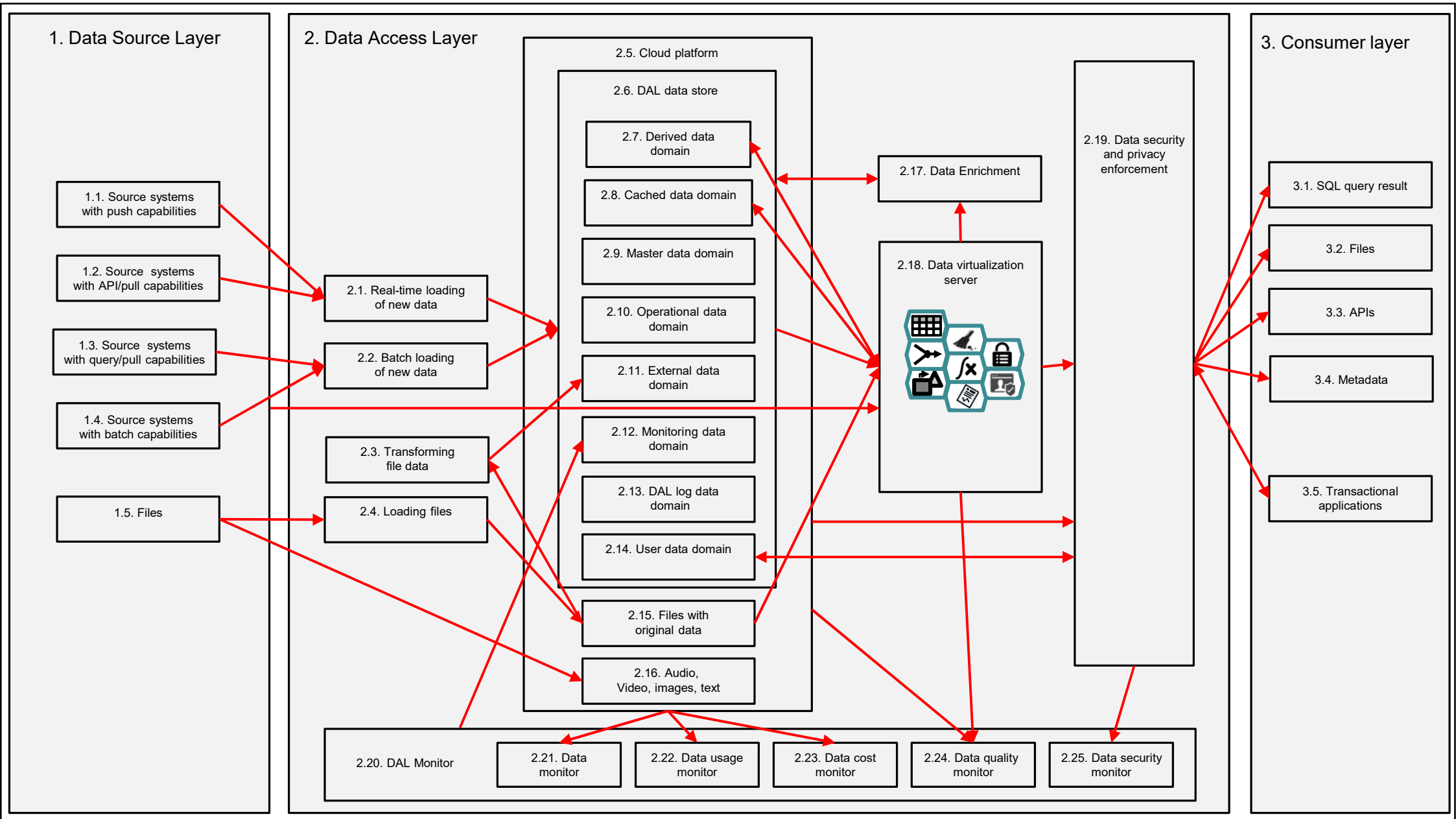


# Wrap the Old Data Warehouse (3)





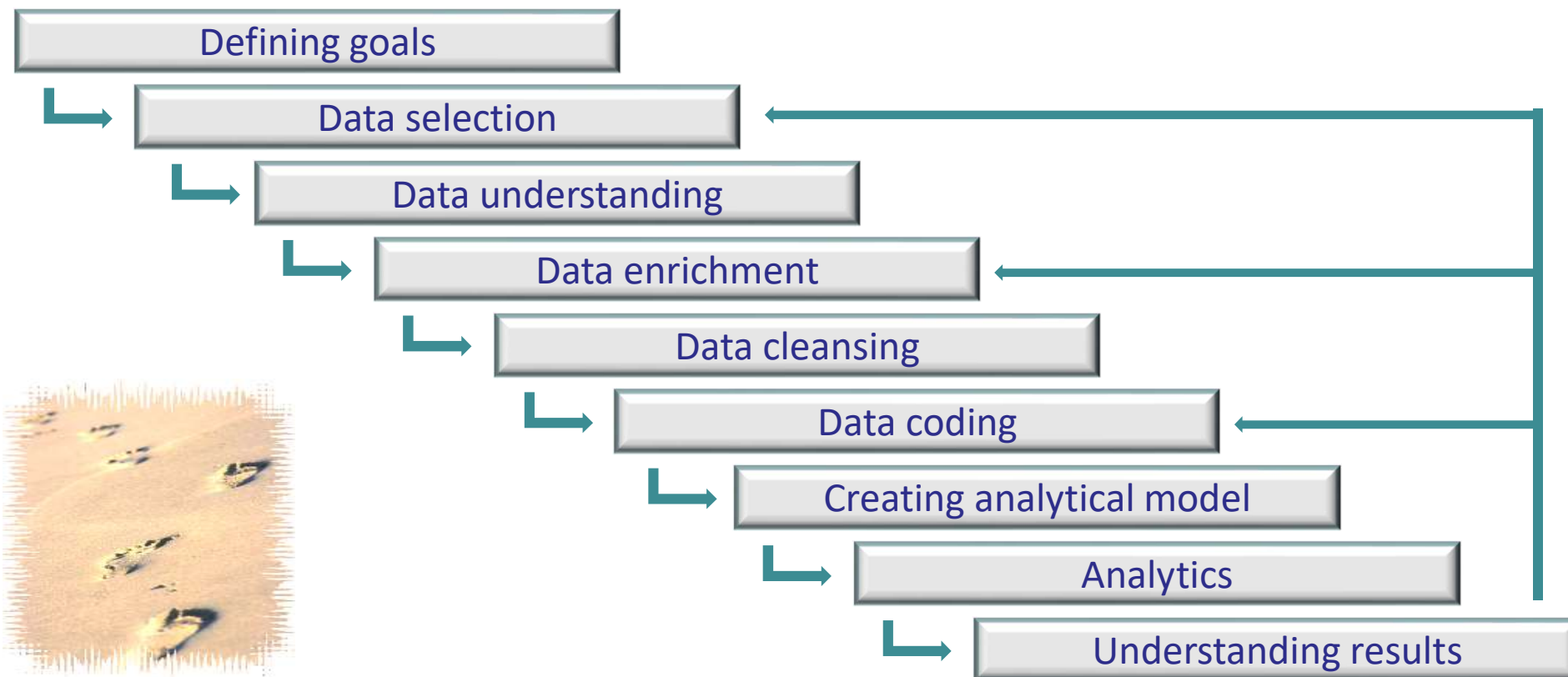




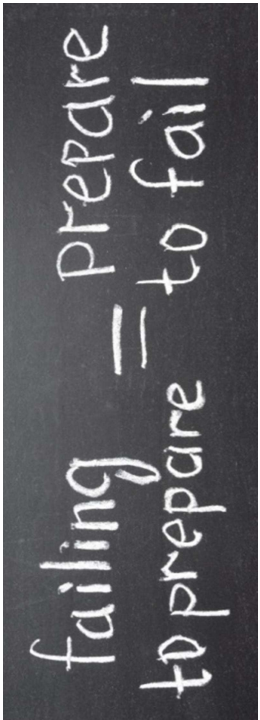
## **Part 7.3: The Data Lake**



# Data Science Steps



# Data Coding



failing = to prepare  
to prepare = to fail

- Computation
  - examples: divide all monthly salaries by 1000; round all prices
- Grouping continuous values
  - example: all transaction between 08:00 and 10:30 will belong to group 1, all transactions between 10:31 and 12:00 will belong to group 2
  - do groups need equal sizes (with respect to ranges)?
  - do groups need equal numbers of values?
- Scaling
  - most neural networks accept numeric data only in the range 0.0 to 1.0 or -1.0 to 1.0; used for continuous values, such as salary and weight
- Normalizing
  - sum all elements, and divide each element by the sum
  - value represents the percentage of contribution
- Symbolic to numeric transformations
  - example: the string "yes" becomes 1, and "no" becomes 0
- Coding discrete values
  - transform a column with fixed set of values (F) into F columns with yes/no values

## Common Definition of Data Lake

“

James Serra:

A “data lake” is a storage repository, usually in Hadoop, that holds a vast amount of raw data in its native format until it is needed. It’s a great place for investigating, exploring, experimenting, and refining data, in addition to archiving data.

”

# The Data Lake

Data sources



E<sub>T</sub>L



E<sub>T</sub>L



E<sub>T</sub>L

Data lake



Investigative analytics



E<sub>T</sub>

Data science



E<sub>T</sub>

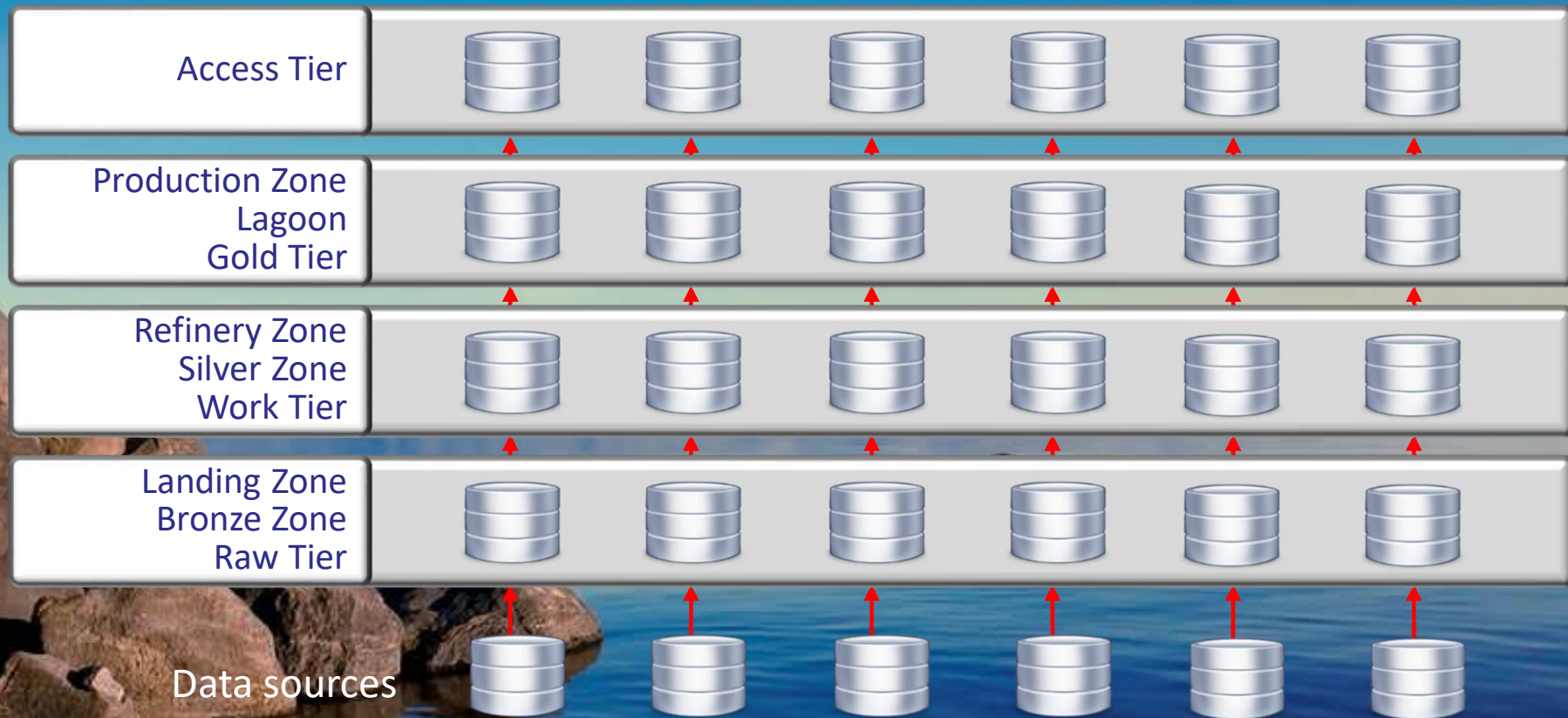
# Challenges of a Physical Data Lake

---



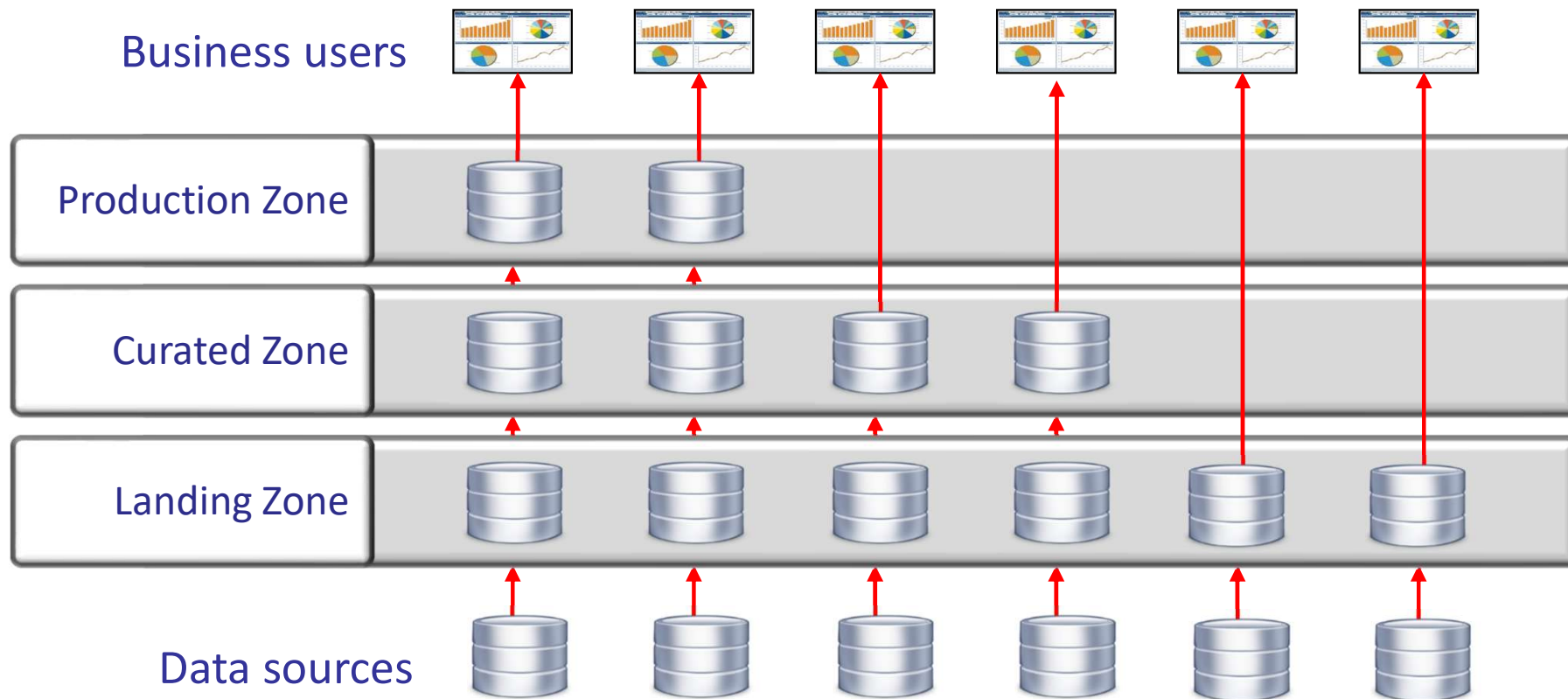
- Big data too big to move
  - Too slow to copy and bandwidth issues
- Complex “T” moved to data consumption
- Company politics
- Data privacy and protection regulations
- Data in data lake is stored outside original security realm
- Metadata to describe data
- Some sources are hard to copy
  - For example, mainframe data
- Refreshing of data lake
- Management of data lake required
- ...

# The Business Data Lake

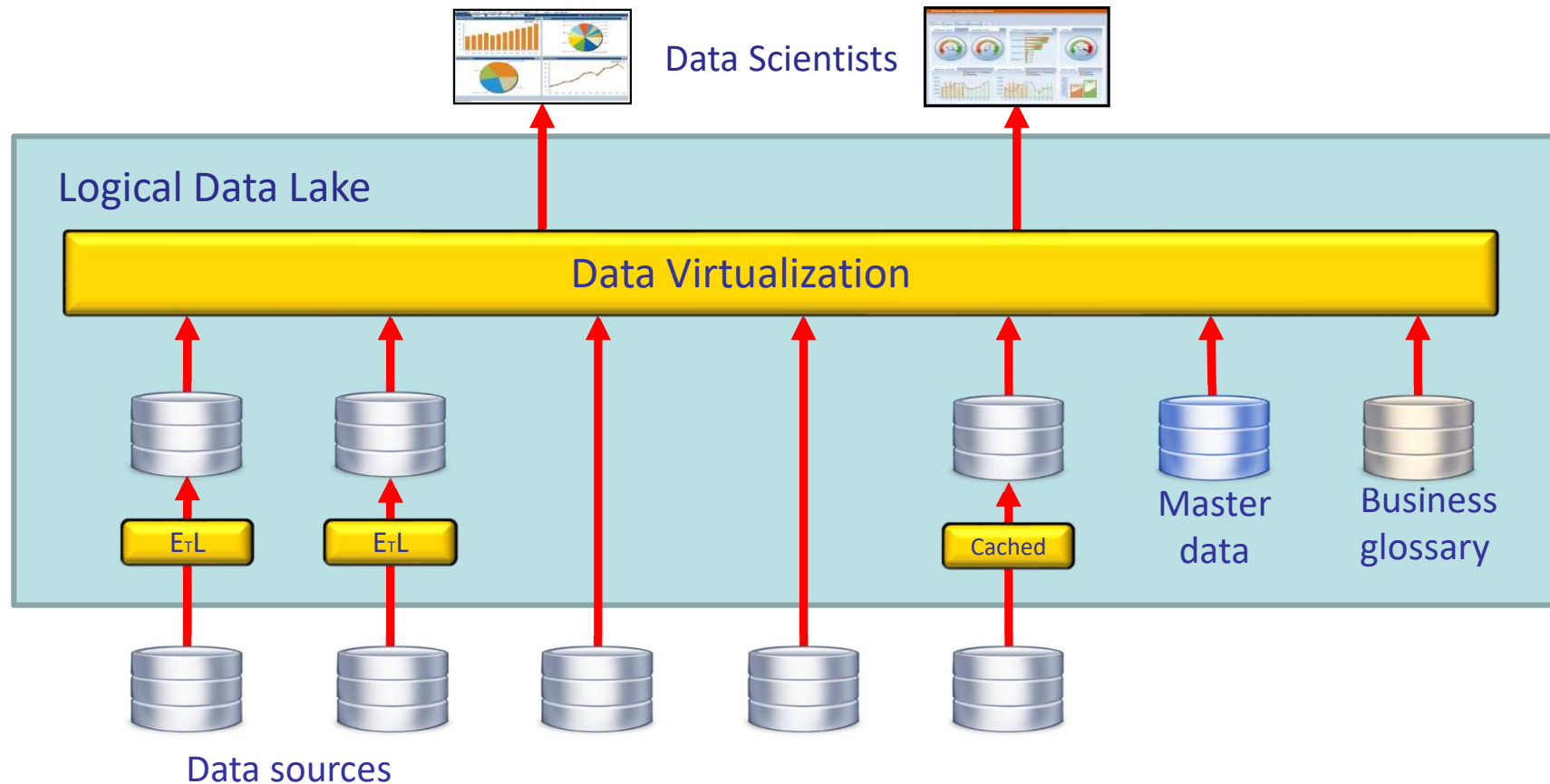


Data sources

# A Data Lake With Multiple Zones



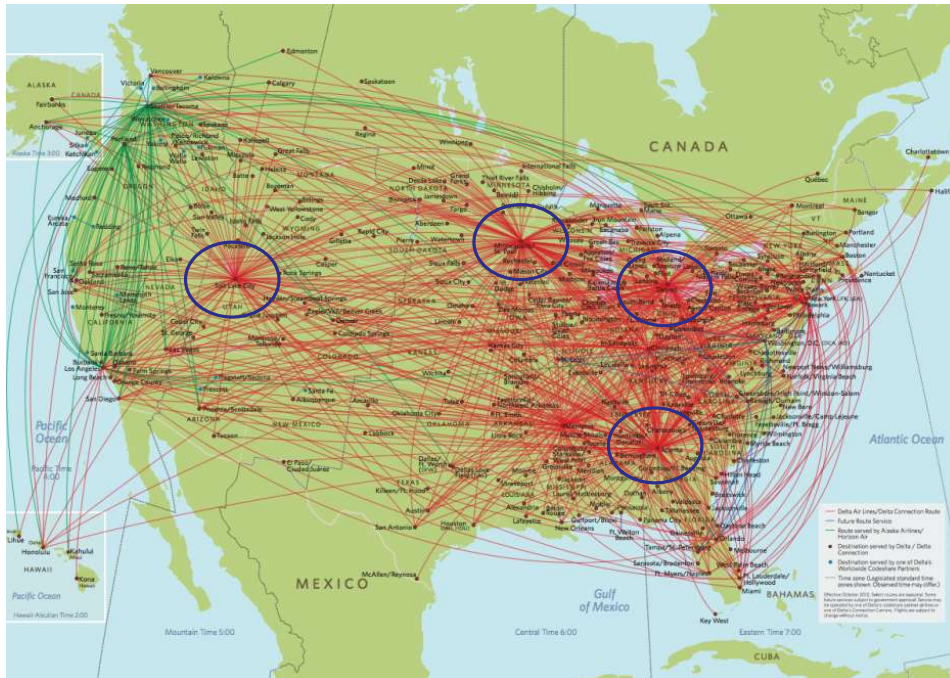
# The Logical (Virtual) Data Lake



## **Part 7.4: The Data Hub**

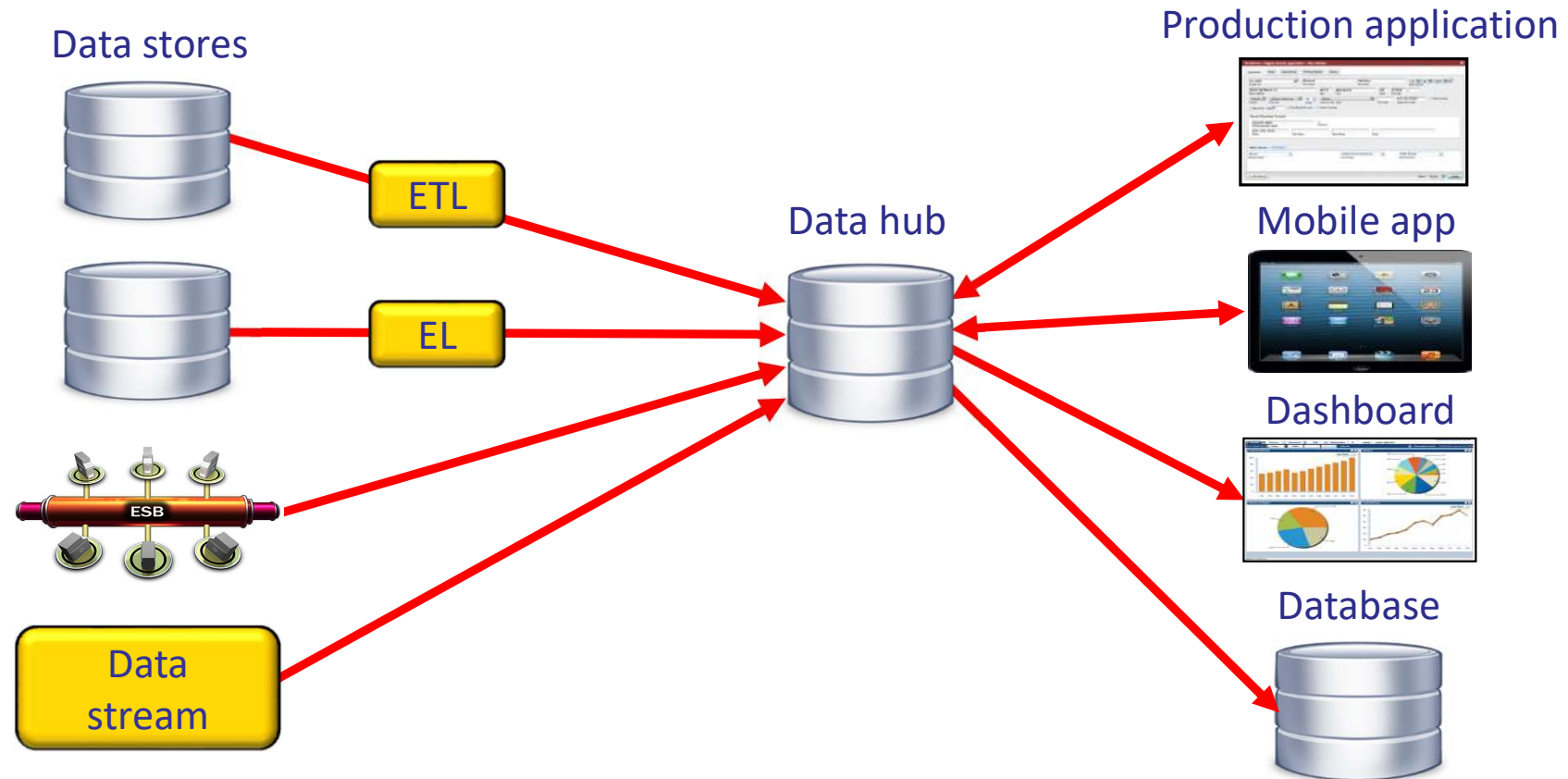


# What is a Data Hub?

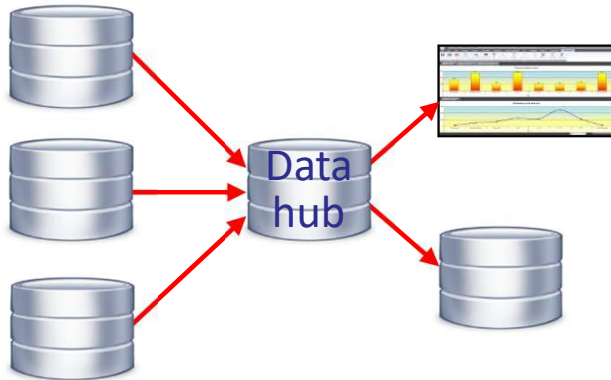


- Wikipedia: A data hub is a collection of data from multiple sources organized for distribution, sharing, and often subsetting and sharing. Generally this data distribution is in the form of a hub and spoke architecture

# The Data Hub (Sharing of Data)

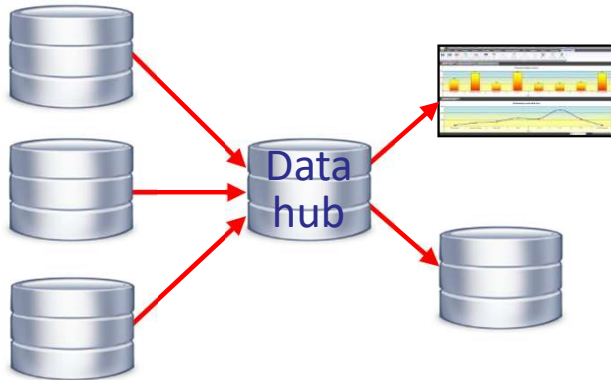


# Characteristics of the Data Hub






- The main goal of a data hub is to organize data efficiently, storing it in a cost-efficient manner and expose it towards key business functions
- It excels in easy integration, and enables de-duplication, security, quality and data standardization
- The data hub can be leveraged to enable data processing activities with the end use-case in mind, and typically has governance capabilities
- Although operationally focused, it can be trusted as an analytical data source

# Data Hub Versus the Rest Of the World



- Data hub versus *data warehouse*: a data hub is generally non-integrated and often at different grains
- Data hub versus *operational data store*: a data hub does not need to be limited to operational data
- Data hub versus *data lake*: a data lake tends to store data in one place for availability, and allow/require the consumer to process or add value to the data
- Data warehouses and data lakes may be endpoints, data hubs are not endpoints, they serve as points of intermediation and data exchange

# Comparison of Three Data Storage Environments

DATA WAREHOUSE	DATA LAKE	DATA HUB
		
<ul style="list-style-type: none"><li>• STRUCTURED FOR ANALYTICS</li><li>• CONSUMED BY PEOPLE AS A SELF-SERVICE</li><li>• FOCUSED ON DECISION MAKING</li></ul>	<ul style="list-style-type: none"><li>• (UN)STRUCTURED FOR DISCOVERY</li><li>• CONSUMED BY DATA PROFESSIONALS AND ALGORITHMS</li><li>• FOCUSED ON DEEP LEARNING, AI</li></ul>	<ul style="list-style-type: none"><li>• STRUCTURED FOR DATA PORTABILITY</li><li>• CONSUMED BY PEOPLE AND APPS</li><li>• FOCUSED ON DATA INTEGRITY AND SPEED FOR SHARING</li></ul>

Source: Talend; see <https://www.talend.com/resources/customer-360-data-hub/>

# Part 7.5: The Data Lakehouse

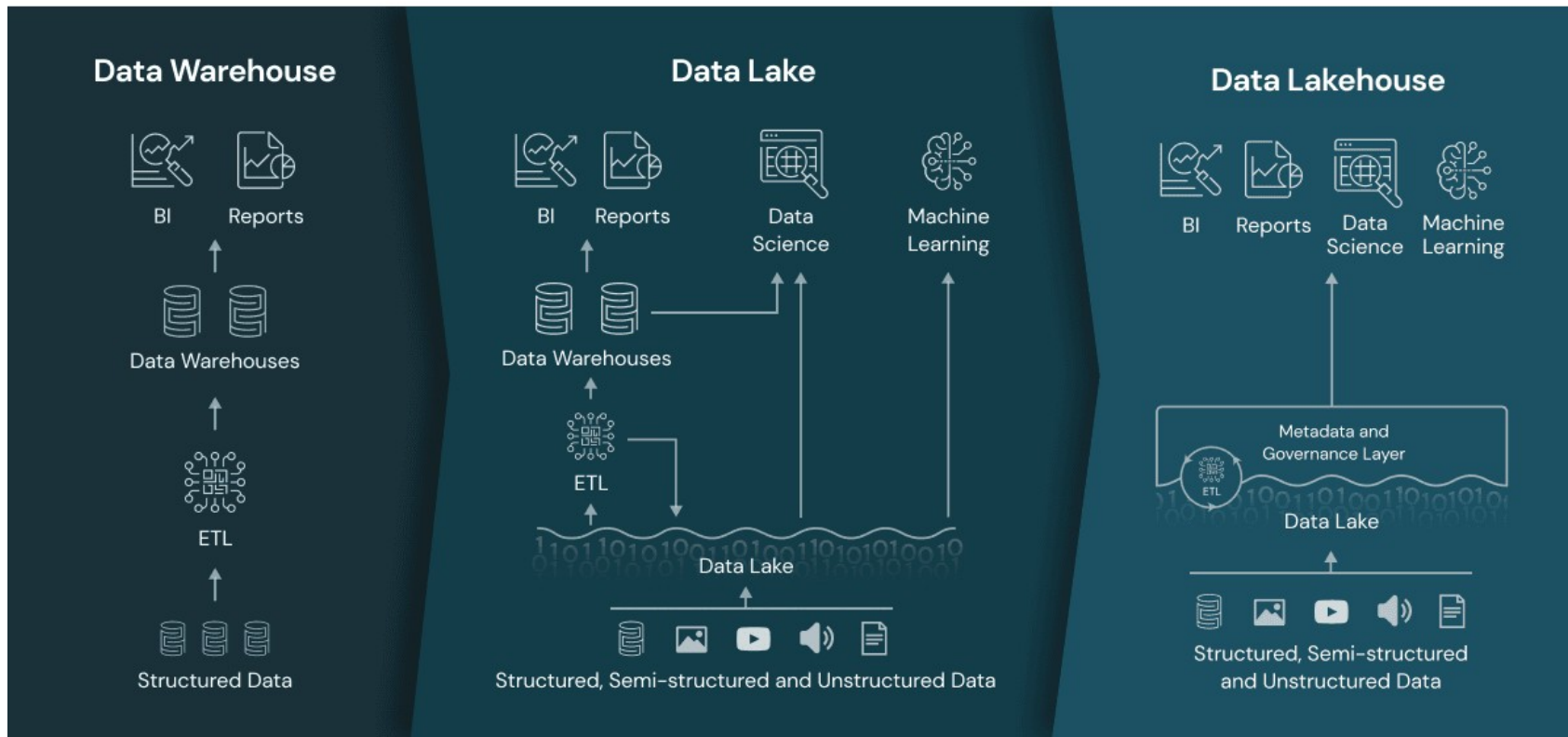


# Definitions of Data Lakehouse

---

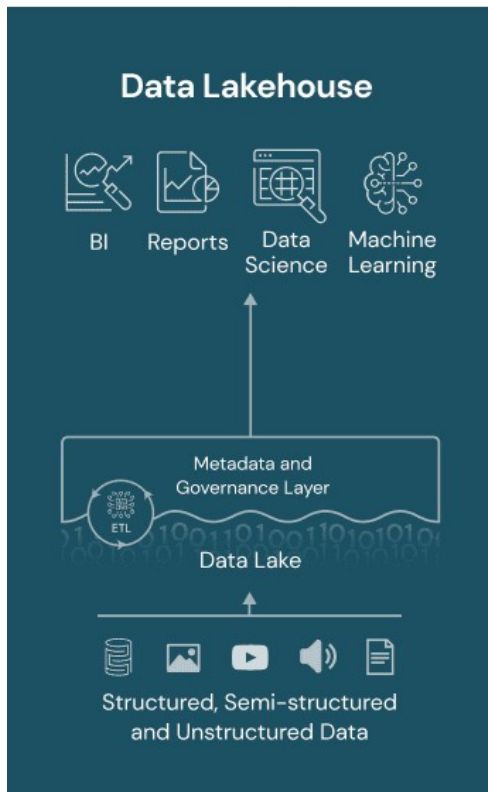
- DataBricks: “A data lakehouse is a [...] open data management architecture that combines the flexibility, cost-efficiency, and scale of data lakes with the data management and ACID transactions of data warehouses, enabling business intelligence (BI) and machine learning (ML) on all data.”
- Striim, John Kutay: “A data lakehouse is a new, big-data storage architecture that combines the best features of both data warehouses and data lakes. A data lakehouse enables a single repository for all your data (structured, semi-structured, and unstructured) while enabling best-in-class machine learning, business intelligence, and streaming capabilities.”
- Dremio, Deepa Sankar: “A [data] lakehouse has the performance and optimization of a data warehouse combined with the flexibility of a data lake.”
- Wikipedia: “Databricks develops and sells a cloud data platform using the marketing term lakehouse, a portmanteau based on the terms data warehouse and data lake.”

# A Comparison of Three Data Architectures



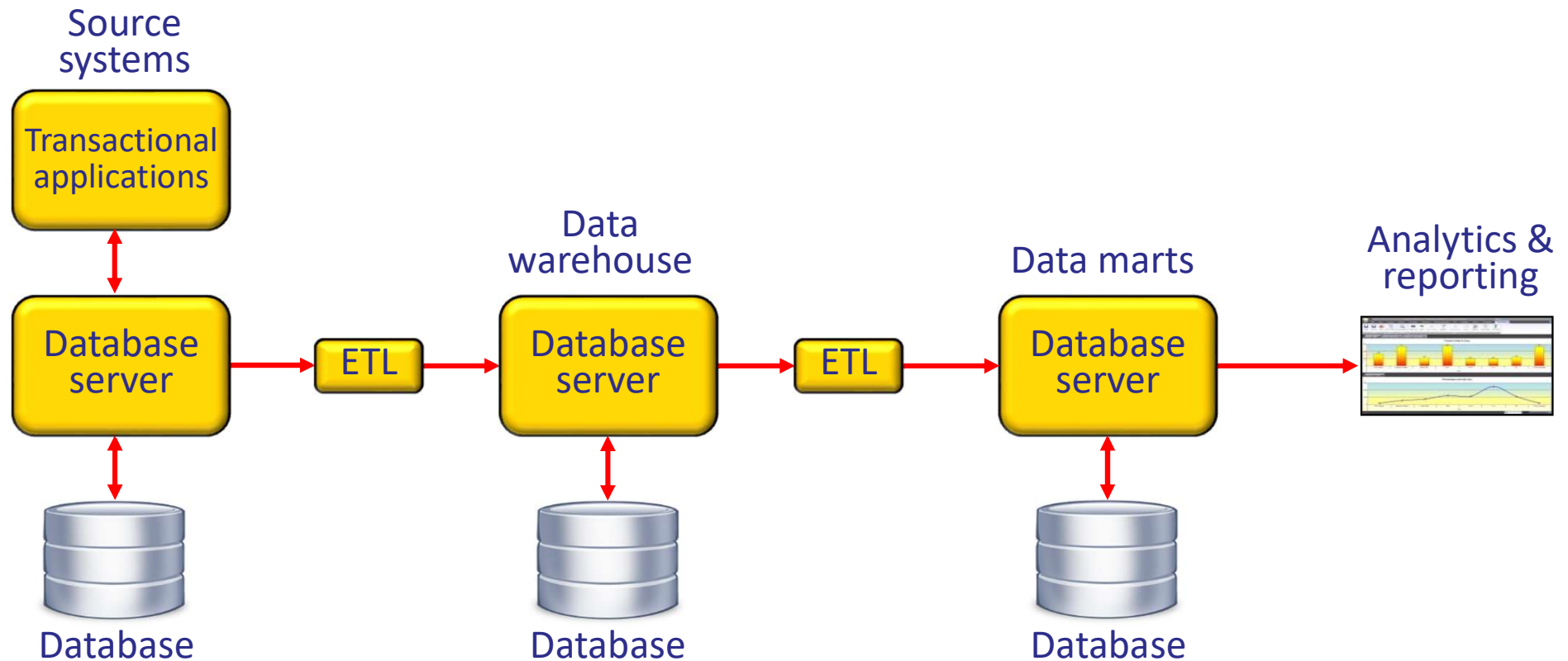
Source: Databricks.com

# Key Characteristics of a Data Lakehouse

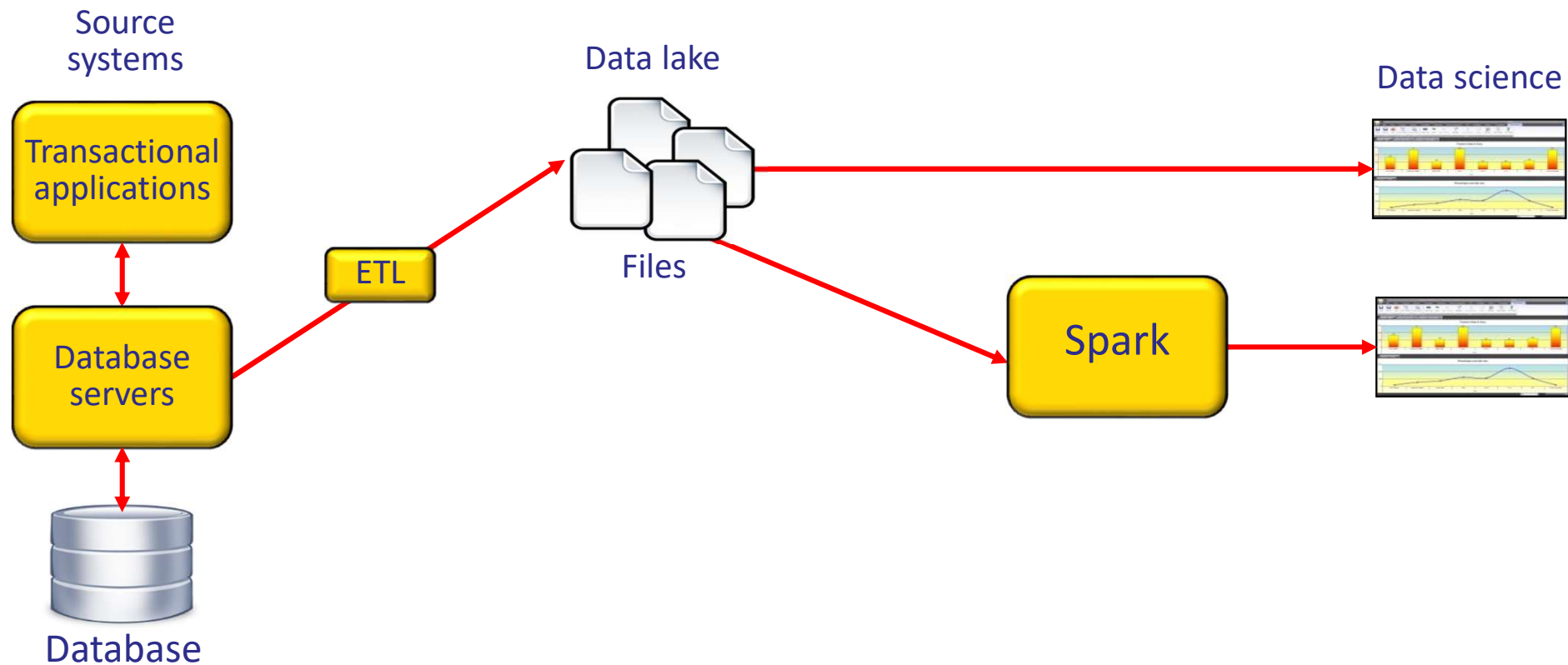


- Two use cases: BI and data science
- Data is stored once
- Supports structured and unstructured data
- Schema enforcement
- Open file formats
- Low-cost data storage
- ACID compliant

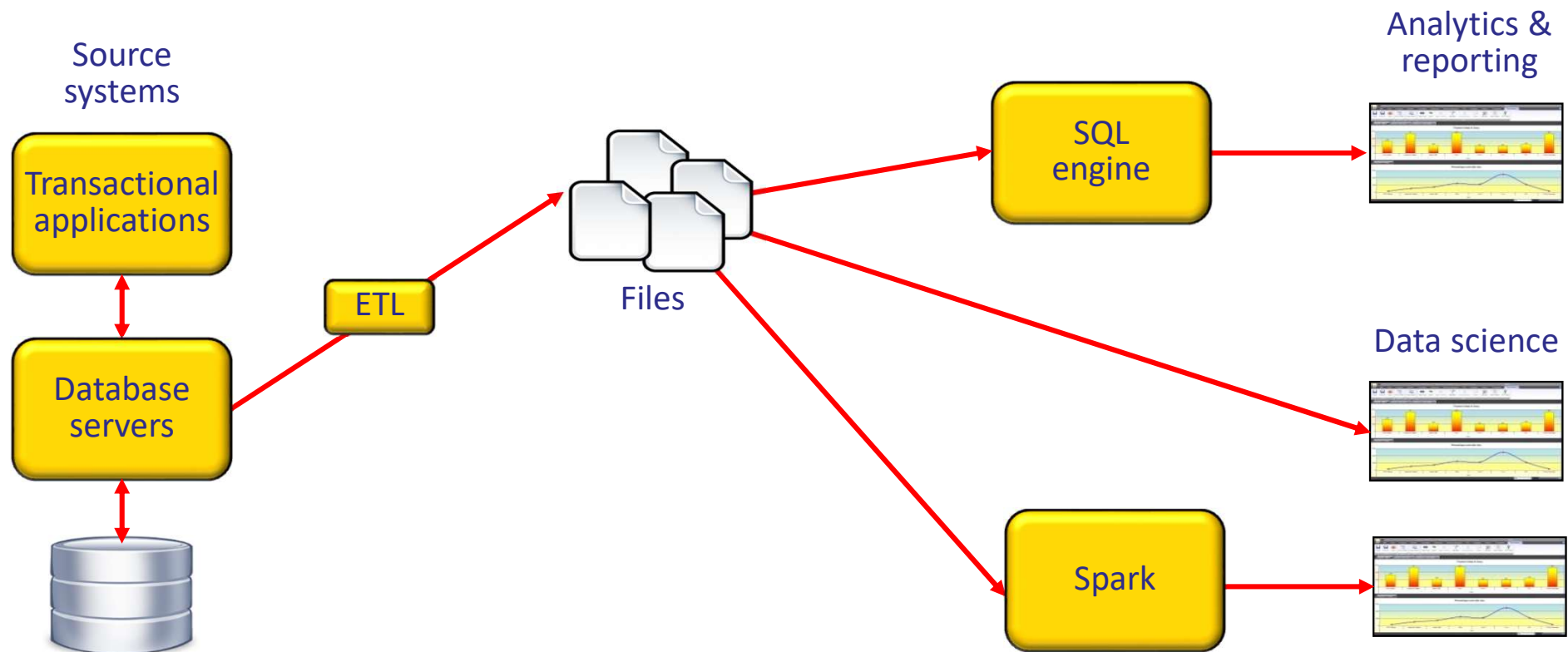
# The Data Warehouse Architecture in More Detail



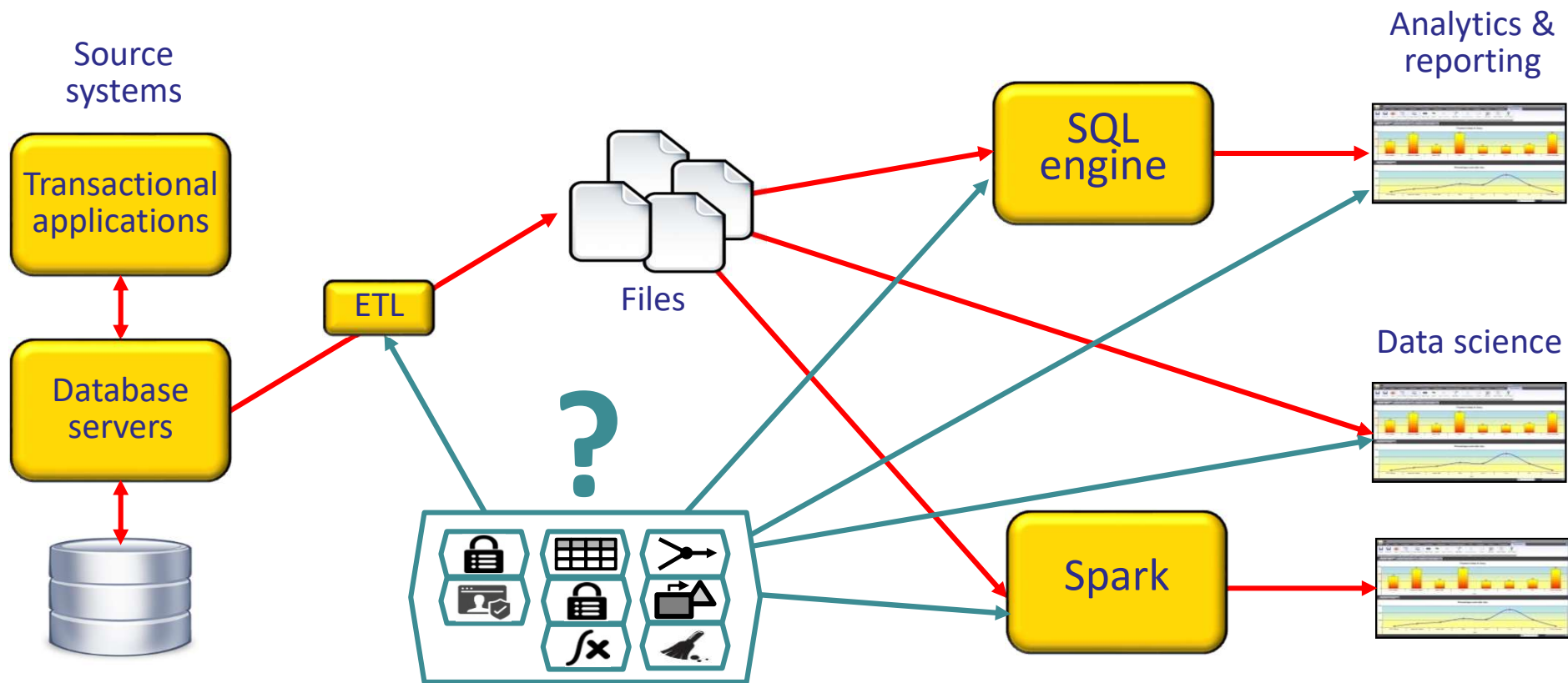
# The Data Lake Architecture in More Detail



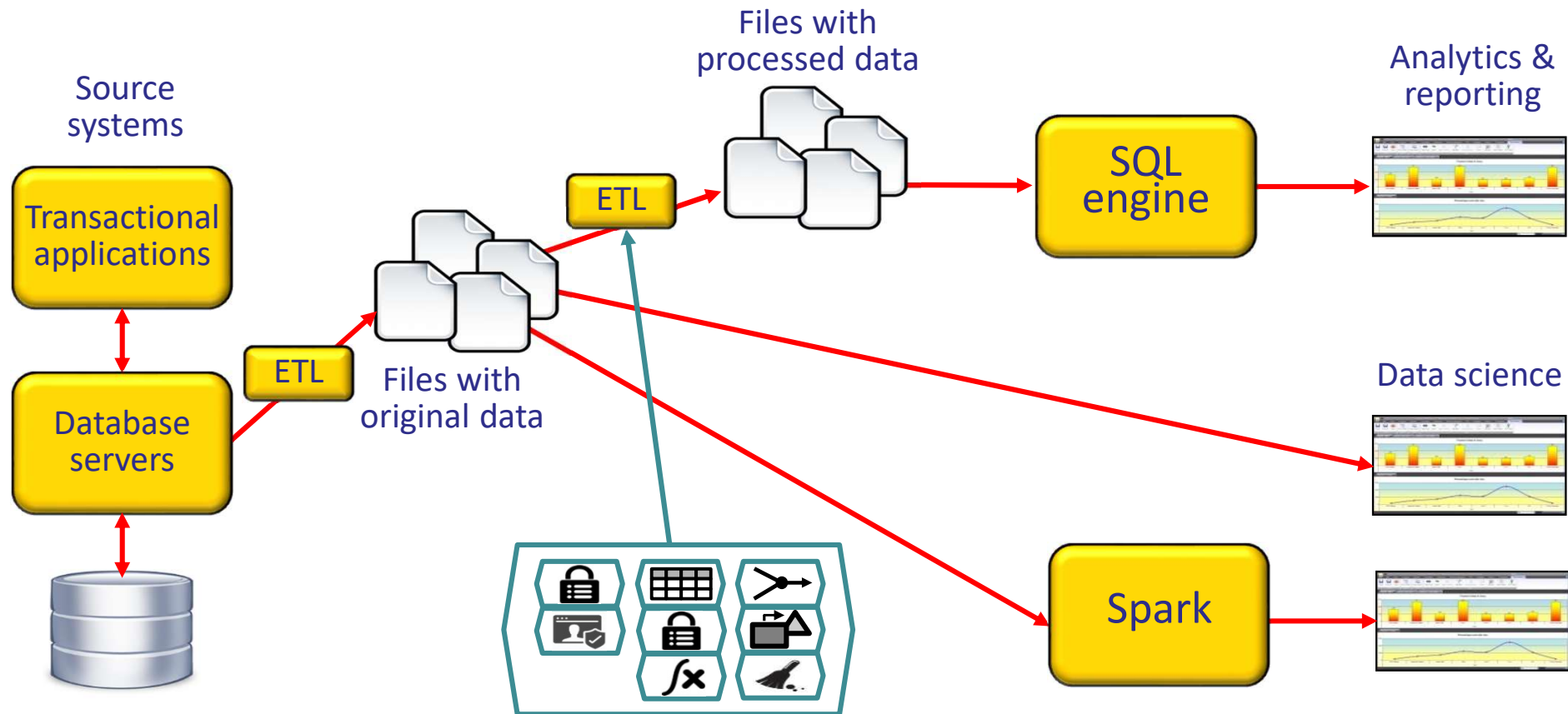
# The Data Lakehouse Architecture in More Detail



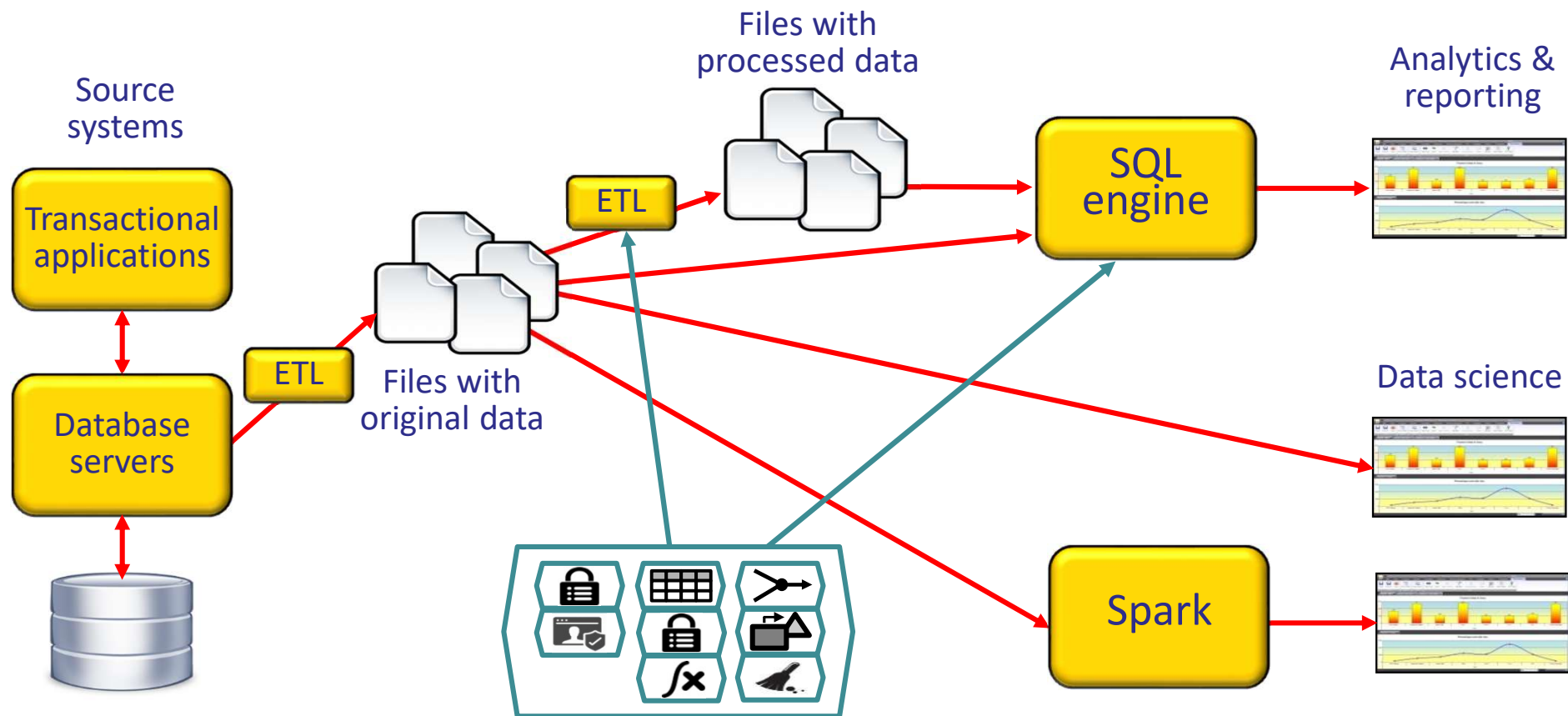
# Where to Implement Transformers?



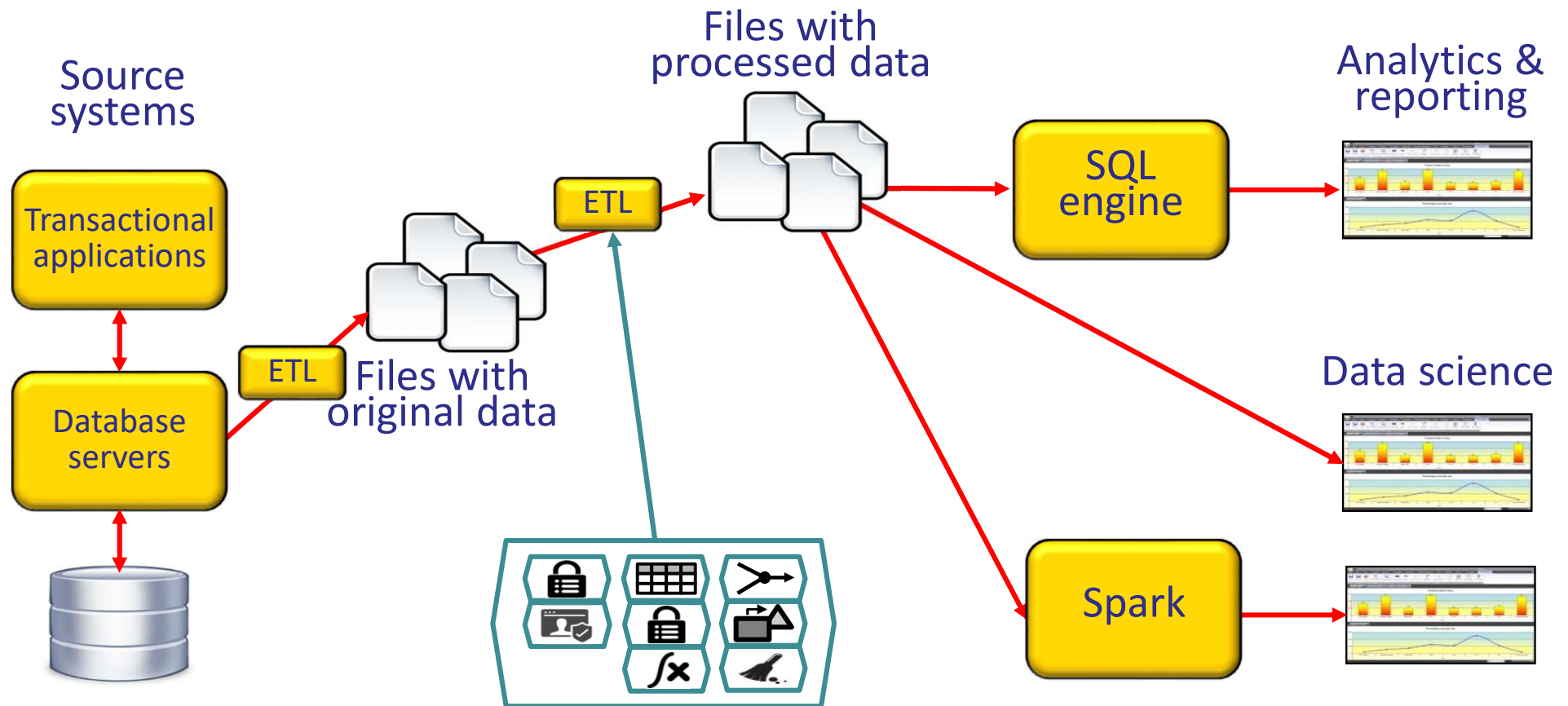
# Solution 1: All BI via Processed Data



## Solution 2: *Some* BI via Processed Data



# Solution 3: Both Use Cases via Processed Data



# High-Level Comparison of Three Architectures

Characteristic	Data Warehouse	Data lake	Data Lakehouse
Type of data	Structured	Structured and unstructured	Structured and unstructured
Use cases	BI, reporting, dashboarding	Experimental, investigative	Both
Schema enforcement	Yes	Optional	Optional
Open file format	No	Yes	Yes
Low-cost data storage	No	Yes	Yes
ACID-compliant	Yes	No	Yes
Near real-time data	No	Yes	Yes
Non-siloed	No	No	Yes
Data copies minimal	No	No	Yes
Anonymization	Yes	Depends	Depends
Auditable	Yes	Depends	Depends
Performance optimized for BI	Yes	No	?
Performance optimized for data science	No	Yes	Yes



Source: Various articles in the Internet



## **Part 7.6: The Data Fabric**



# What is the Data Fabric?

---

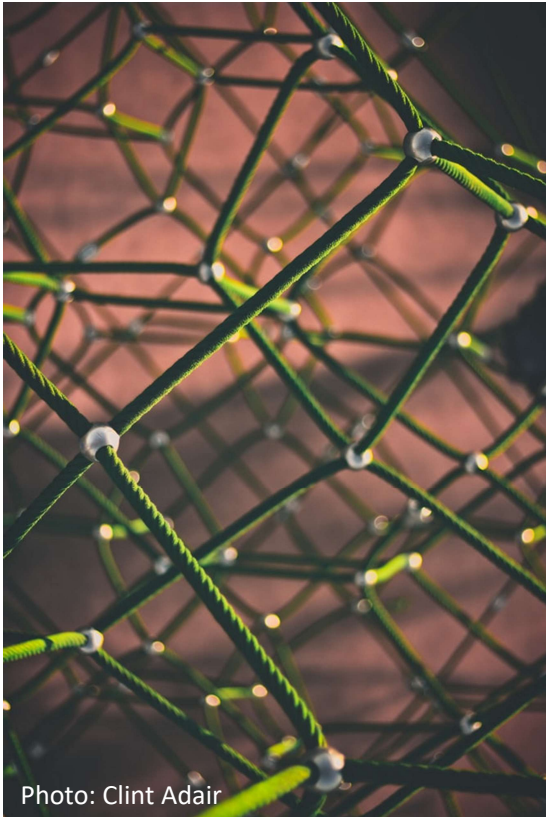
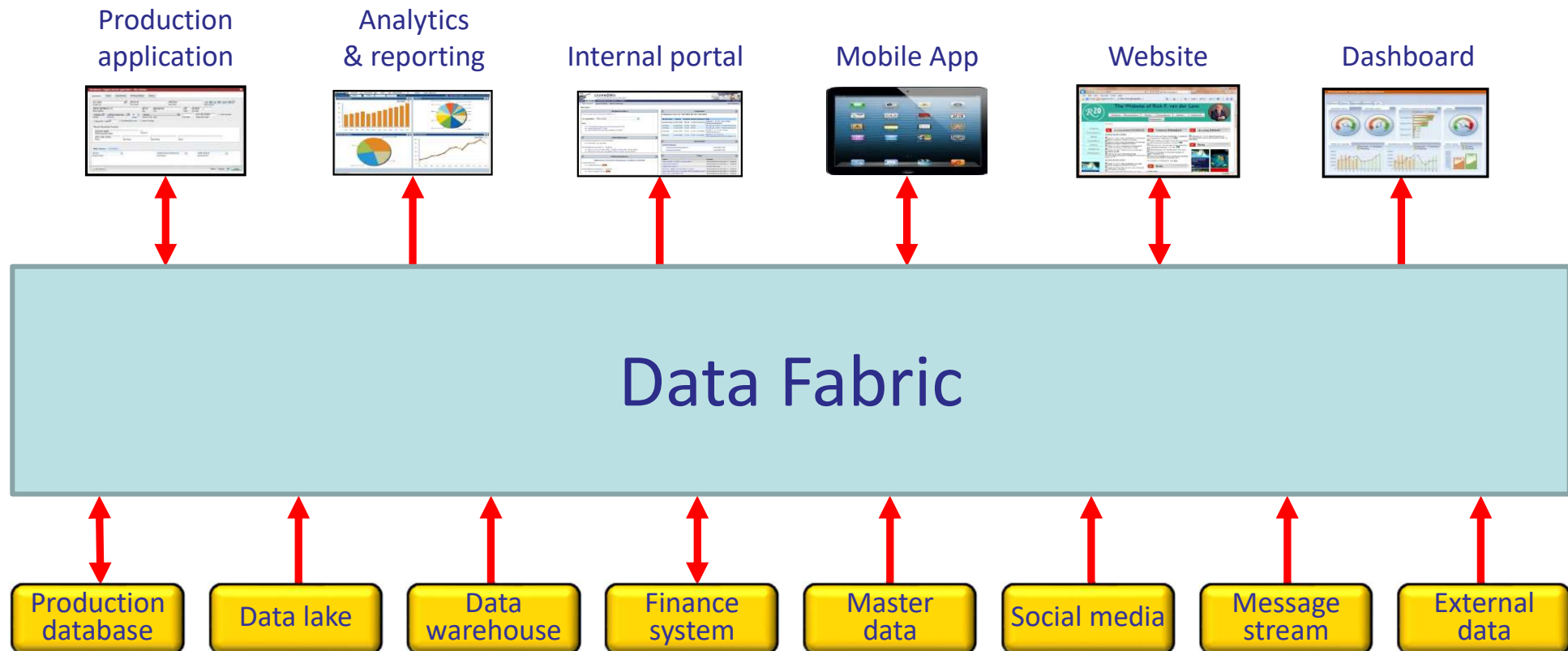


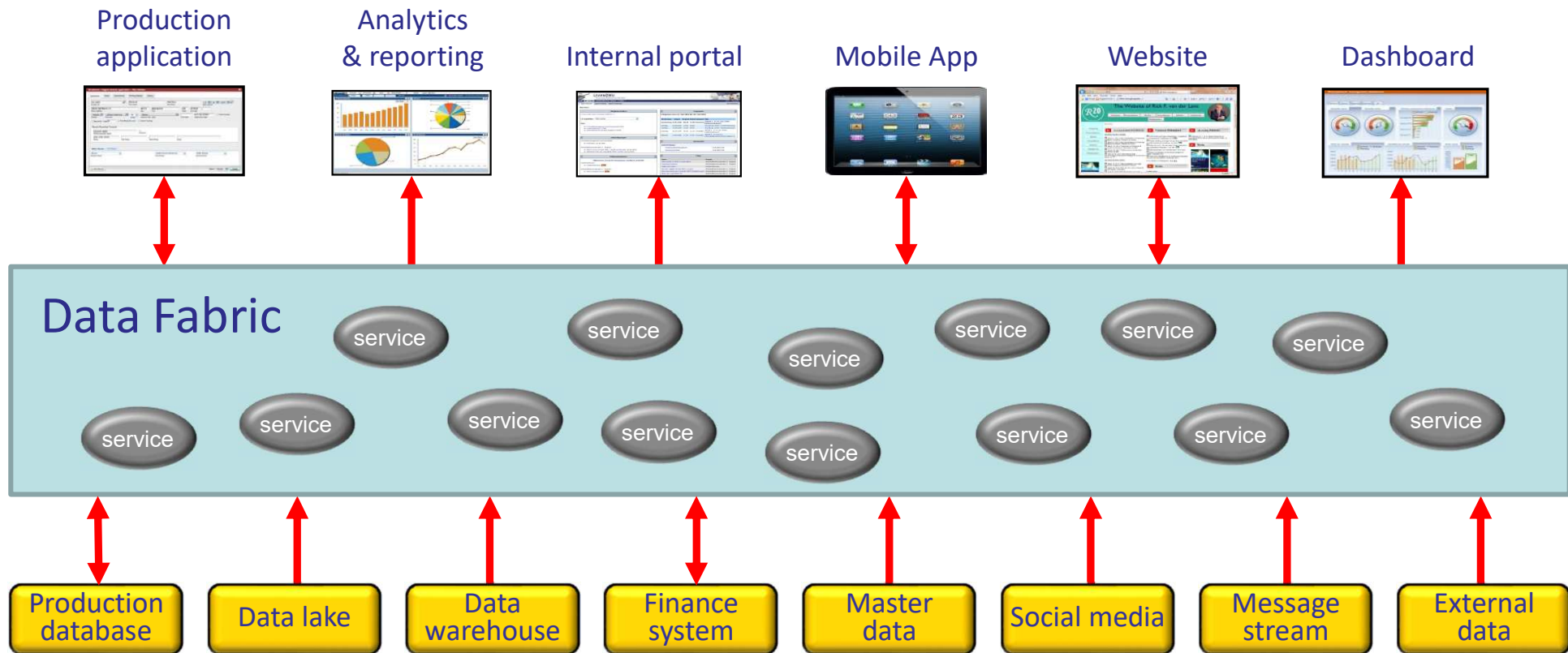
Photo: Clint Adair

- Gartner: A data fabric is generally a custom-made design that provides reusable data services, pipelines, semantic tiers or APIs via combination of data integration approaches in an orchestrated fashion.
- Gartner: Data fabric enables *frictionless access* and sharing of data in a distributed data environment. It enables a *single* and *consistent* data management framework, which allows seamless data access and processing by design across otherwise siloed storage.

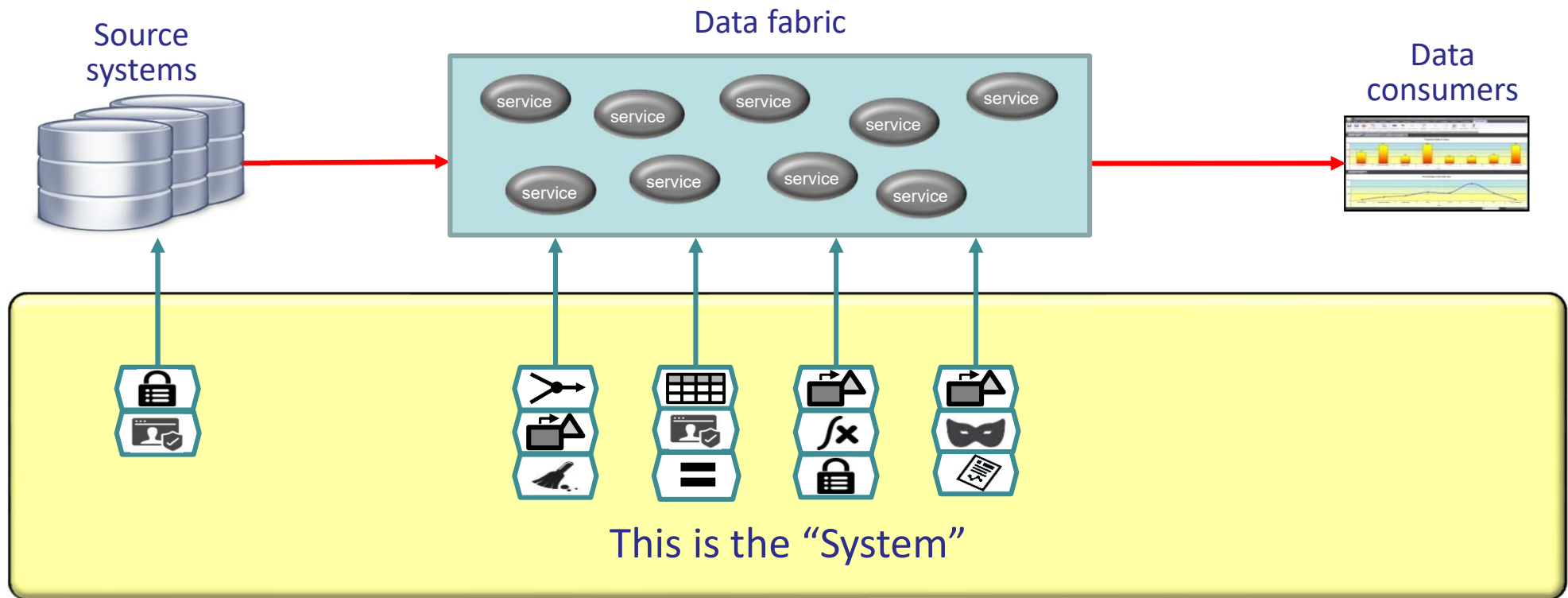
# The Data Fabric



# The Services of a Data Fabric

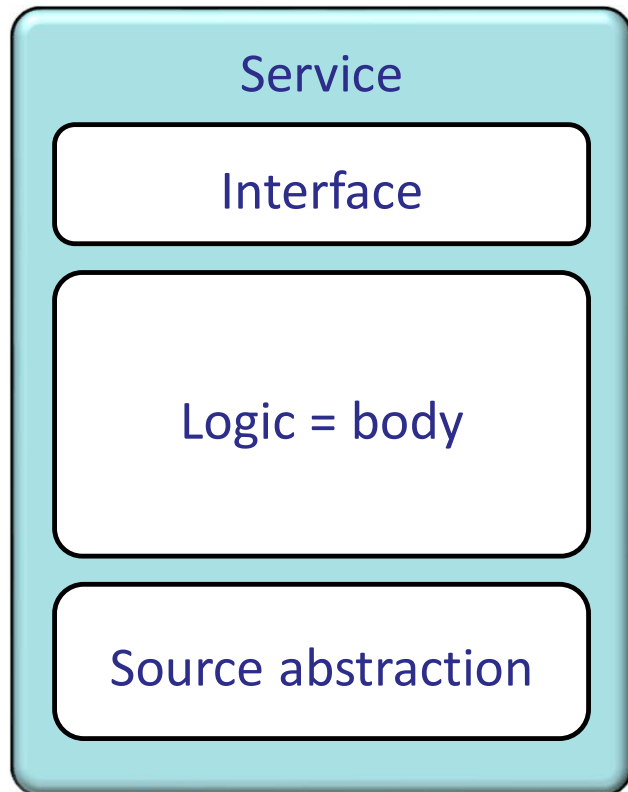


# Data Fabrics and Transformers



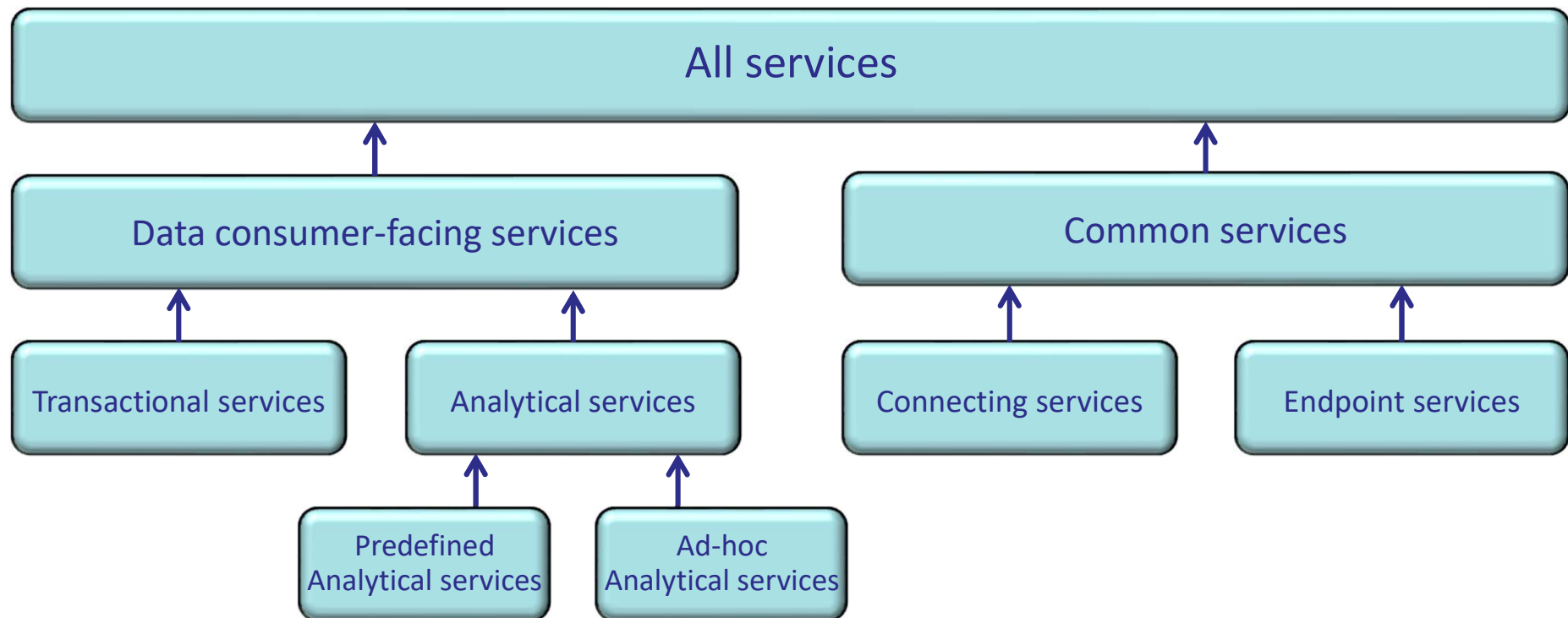
# The Components of Services

---

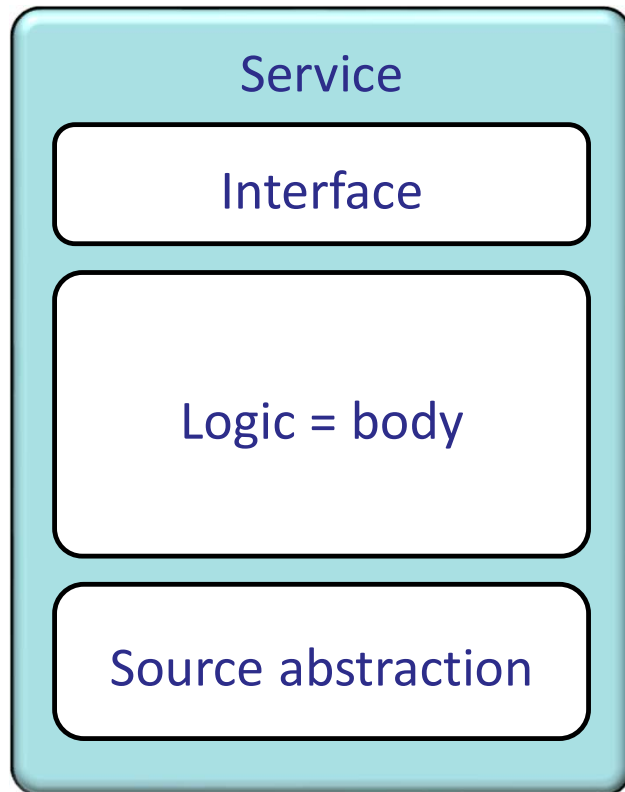


- The interface component is responsible for handling incoming parameters and outgoing results
- The logic of the service form the body
- The body deals with transformers
- Abstraction layer to make it independent of changes to the IT systems

# A Data Fabric Consists of Different Types of Services

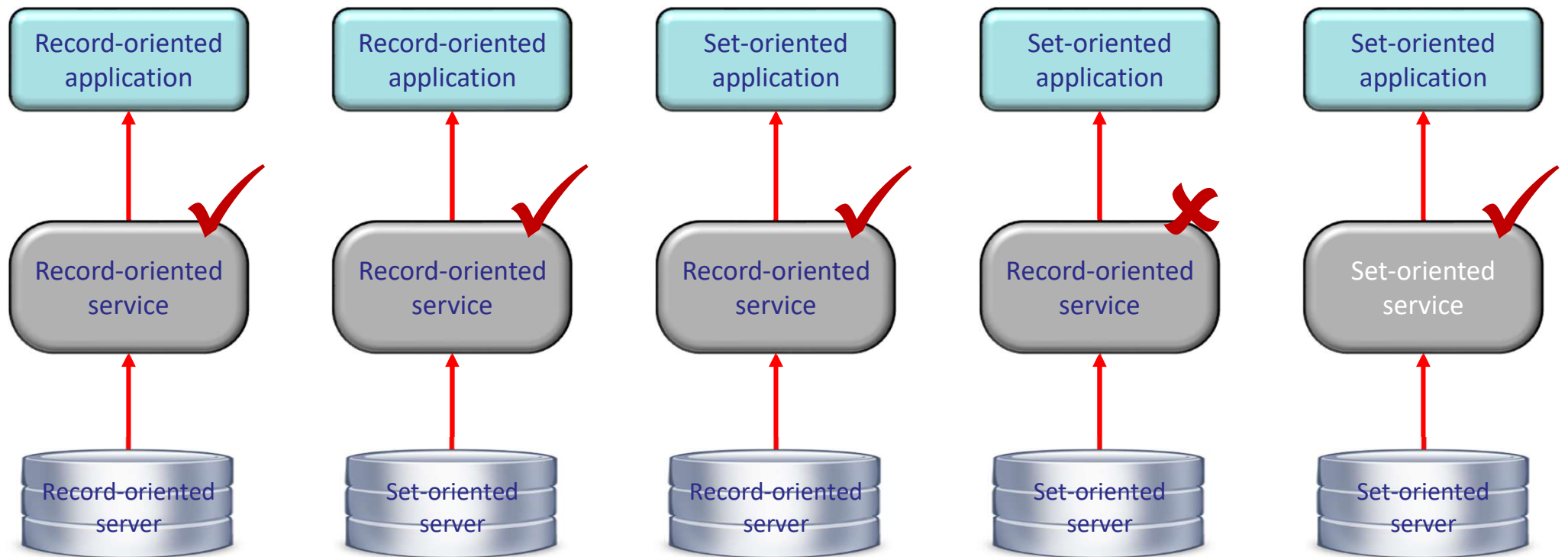


# 12 Capabilities for Frictionless Data Access

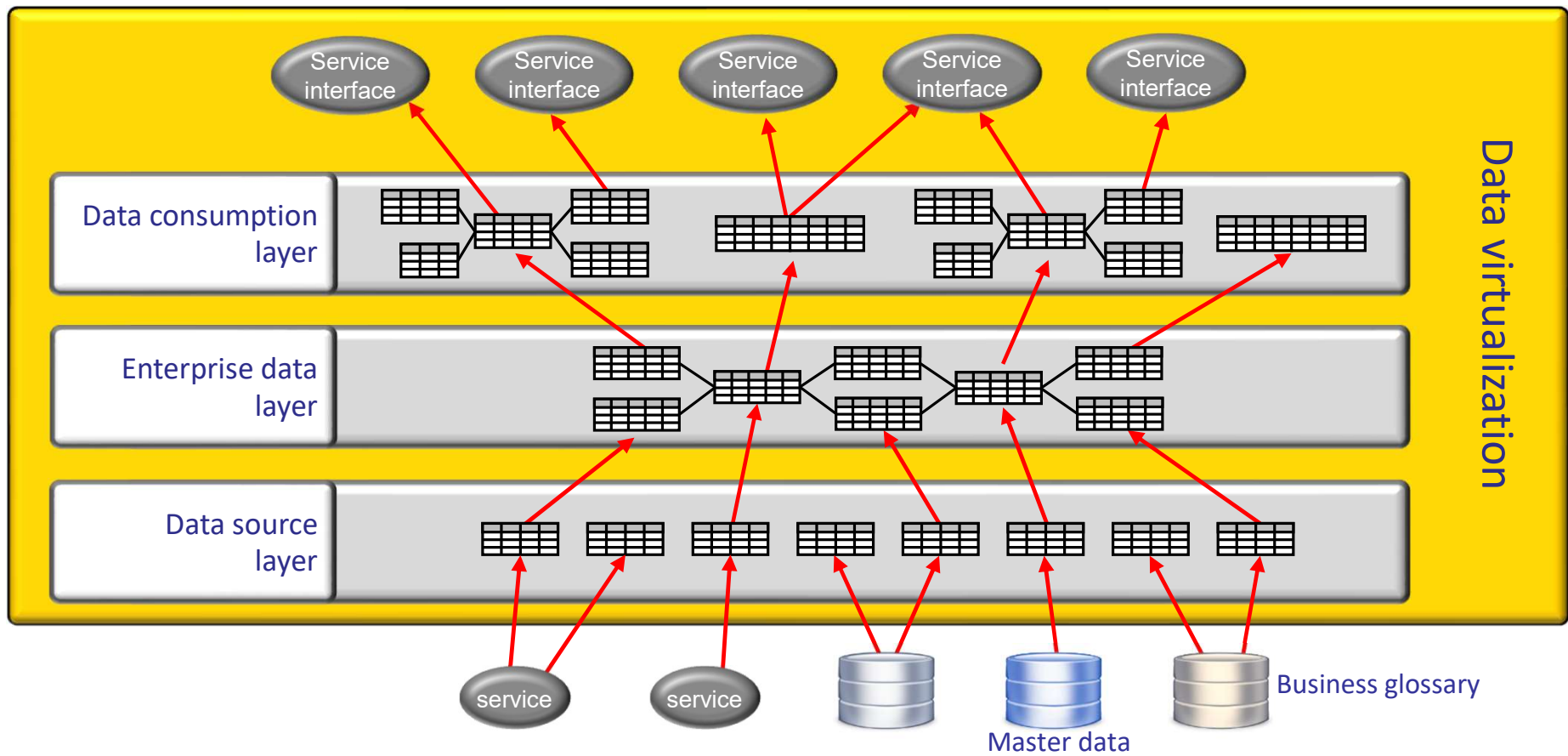


- Data preparation, such as transformations, calculations, aggregations, filters, joins, ...
- Adaptable logic
- High performance
- Data access by many data consumption forms
- Access to all the data sources
- Processing of all types of data
- Data security and privacy
- Real-time data access
- Read and write data access
- Data minimization
- Event processing
- Technical and business metadata management
- Master and reference data management

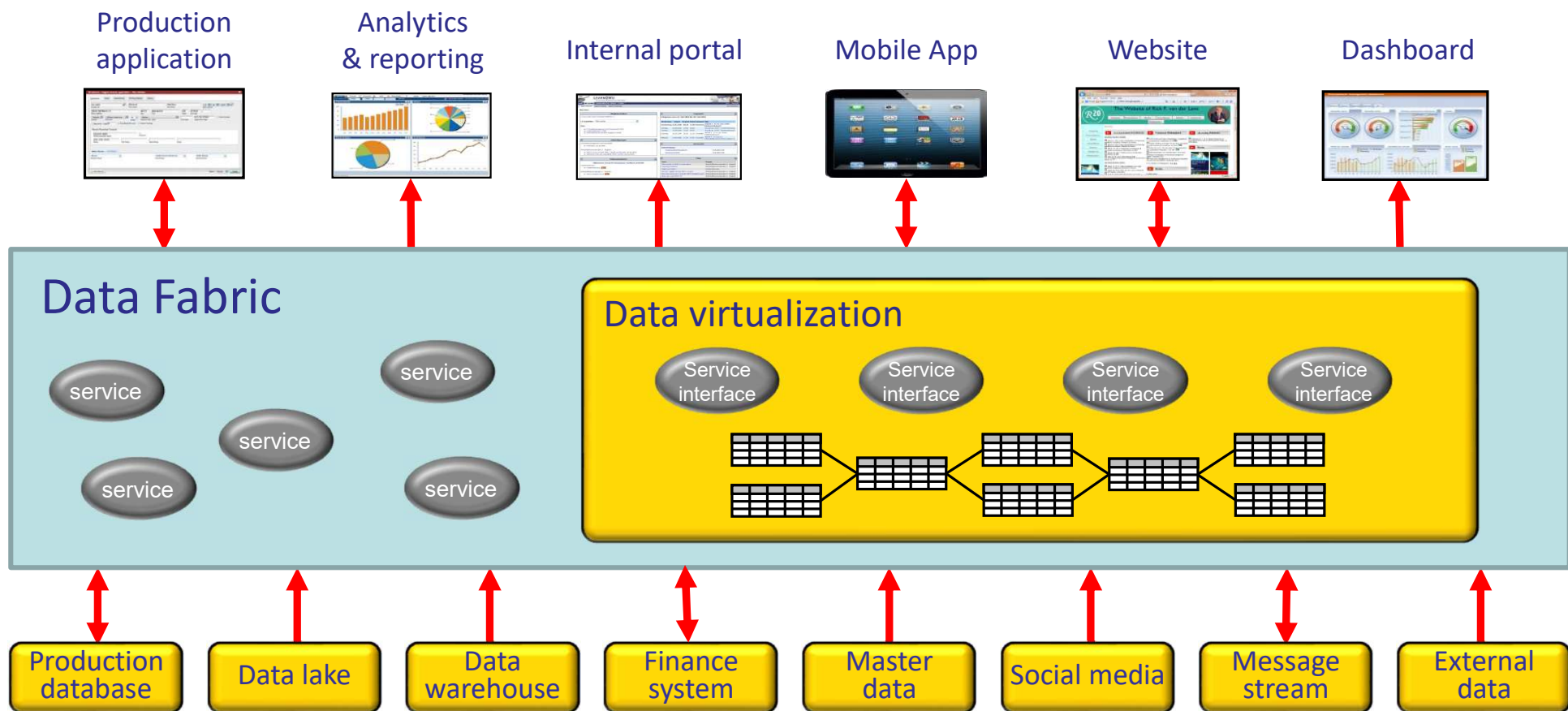
# Record-Oriented or Set-Oriented Interface?



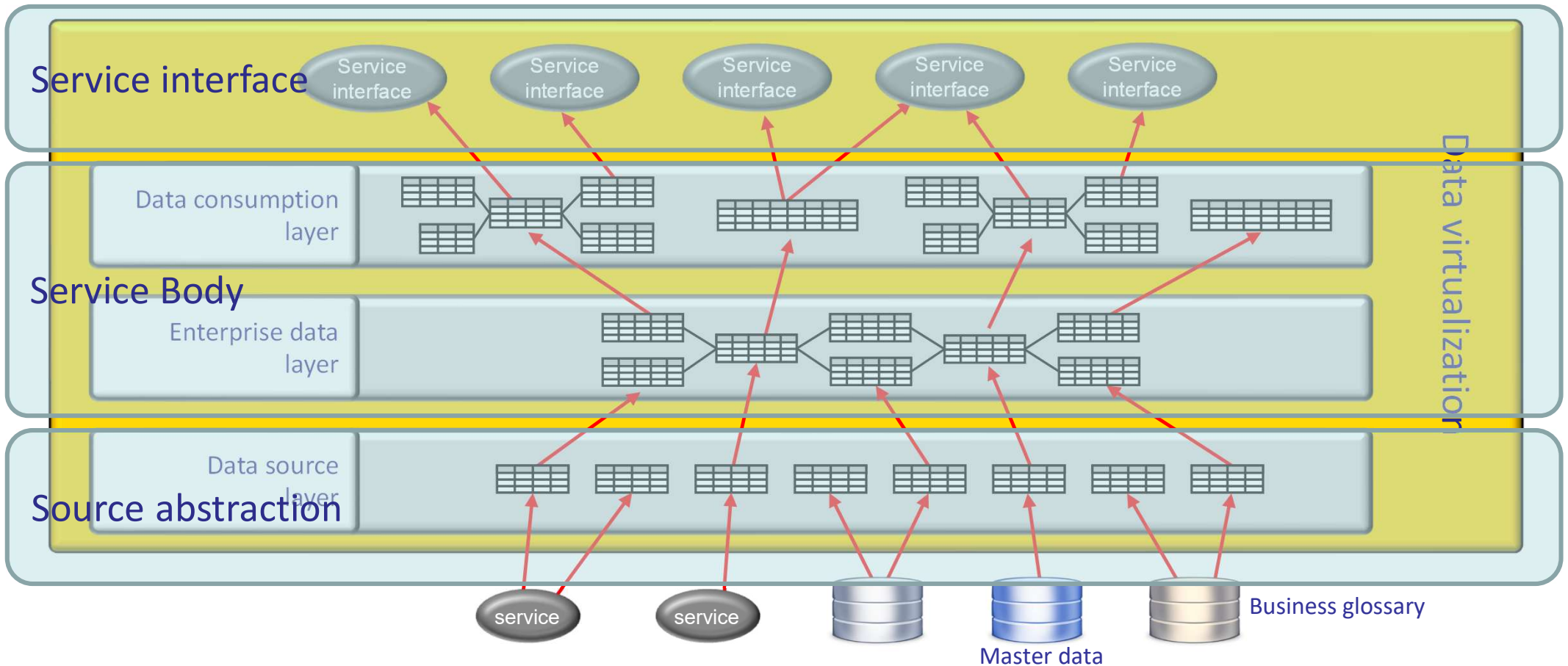
# Developing Data Fabric Services with Data Virtualization



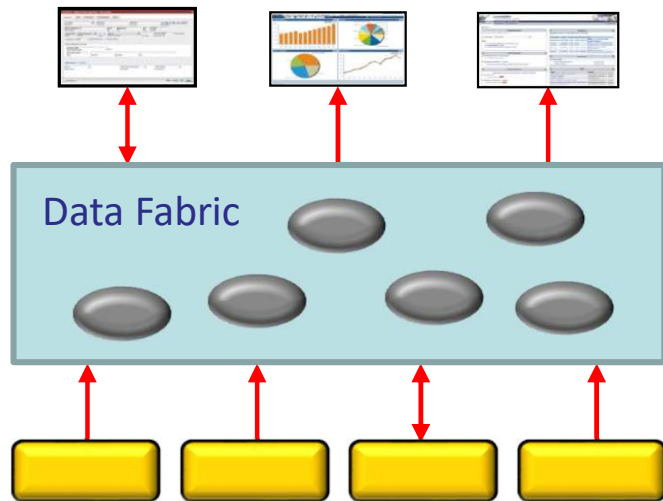
# The Data Fabric



# Developing Data Fabric Services with Data Virtualization

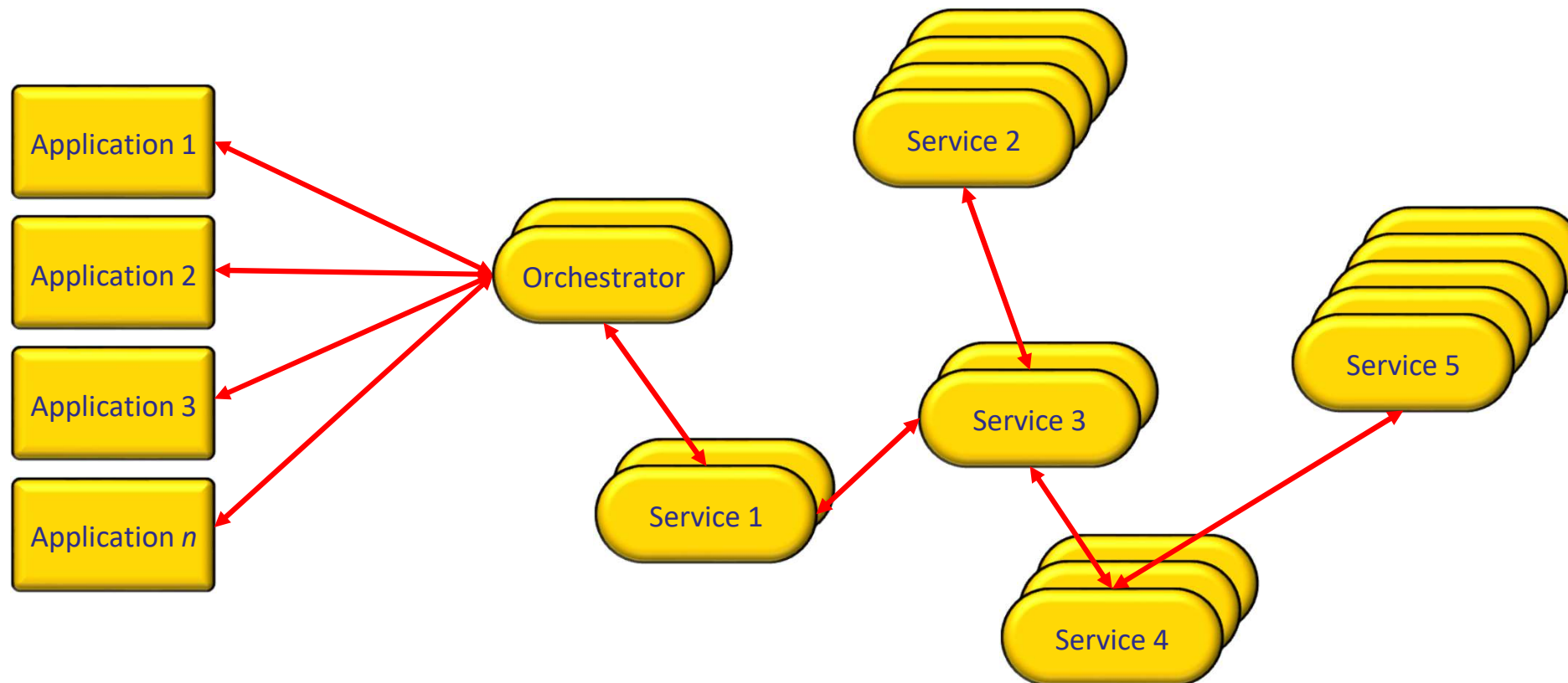


# Remarks on Data Fabric



- Poorly defined concept
- Metadata is required
- Master data needed to integrate data correctly
- A data fabric may contain a data warehouse, data lake, or data hub
- Dedicated tool market is small, e.g. Cinchy
- Data fabric must support transactional and analytical workloads

# Data Fabric $\neq$ Micro-Services Architecture



## **Part 7.7: The Data Mesh**



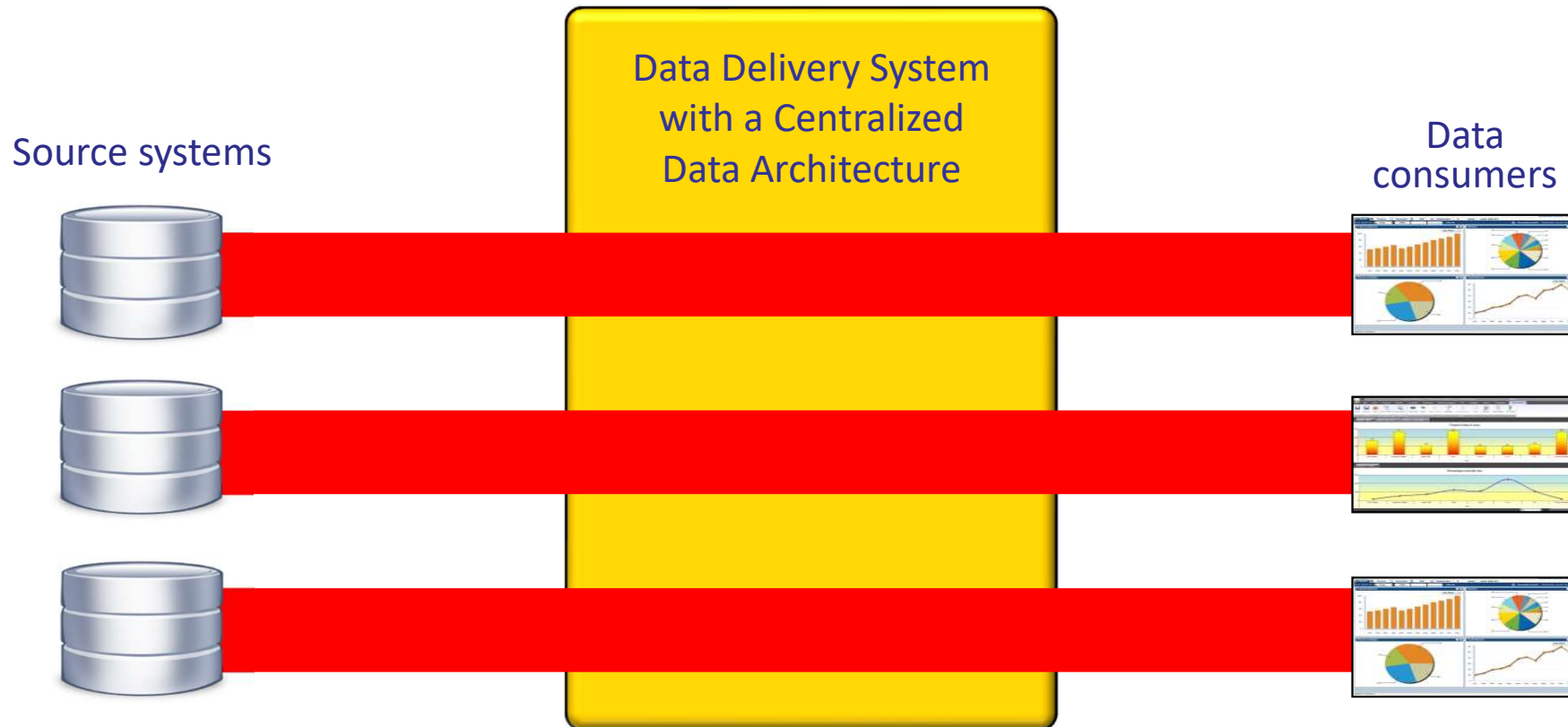
# What is a Data Mesh?

---

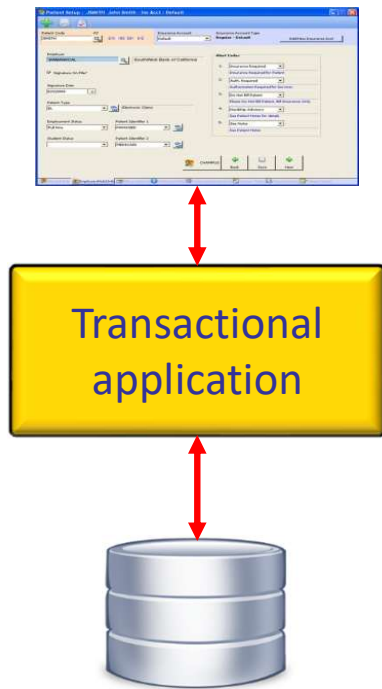


- Introduced by Zhamak Dehghani:
- “Data platforms based on the data lake architecture have common failure modes that lead to *unfulfilled promises* at scale.
- To address these failure modes we need to shift from the *centralized paradigm* of a lake, or its predecessor data warehouse.
- We need to shift to a paradigm that draws from *modern distributed architecture*: considering *domains* as the first class concern, applying platform thinking to create self-serve data infrastructure, and *treating data as a product*.”

# Single-Domain Data Consumers

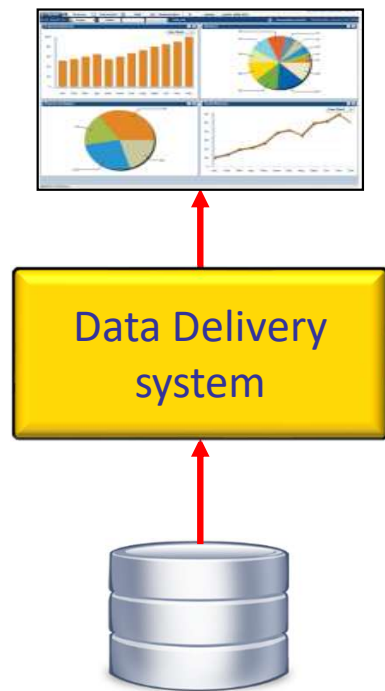


# Data Engineers for Transactional Applications



- They are single-domain experts
- They focus only on data requirements of transactional applications
  - Not on other forms of data consumption, such as BI
- They do not make the data easily consumable
- They implement all the business rules
- They know about all the exceptions
- They know when changes are implemented

# Data Engineers for Data Delivery Systems



- They need to understand the data of all the domains
  - Hyper-domain experts
- They need to transform the data into consumable data
- They need to work with data not designed to be integrated
- They need to understand all the business rules that need to be applied
  - Complex ETL processes
- They need to understand the data requirements of the data consumers
- They need to deal with SLAs of the data consumers

# The Wall between Two Groups of Data Engineers

---

Data Engineers for  
Transactional  
Applications

**The Wall**

Data Engineers for  
Data Delivery  
Systems

- Different groups of data engineers
- Different tool sets
- Different responsibilities
- Changes of data or data structures not always communicated
- Who owns the data and who is responsible for data quality?
- How to implement “the right to be forgotten/corrected”?

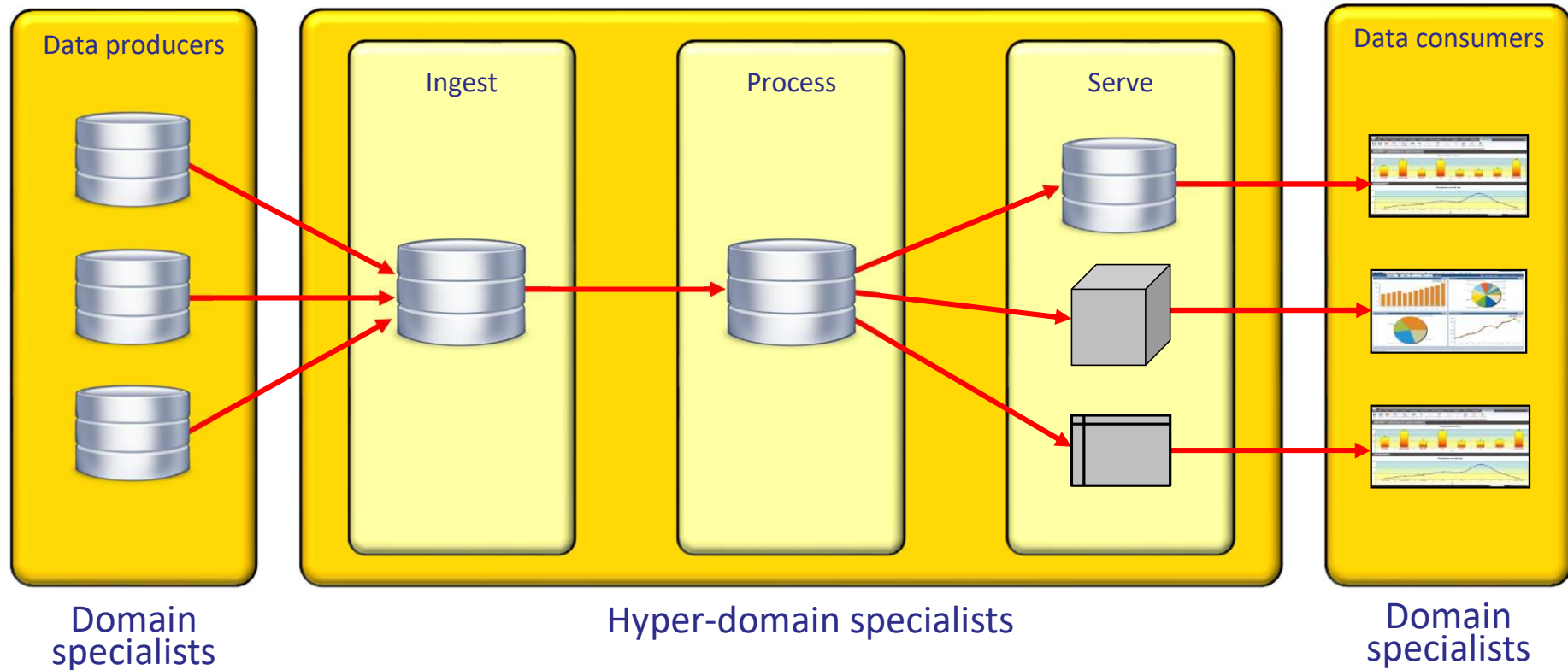
# Interfaces of Transactional Applications

---

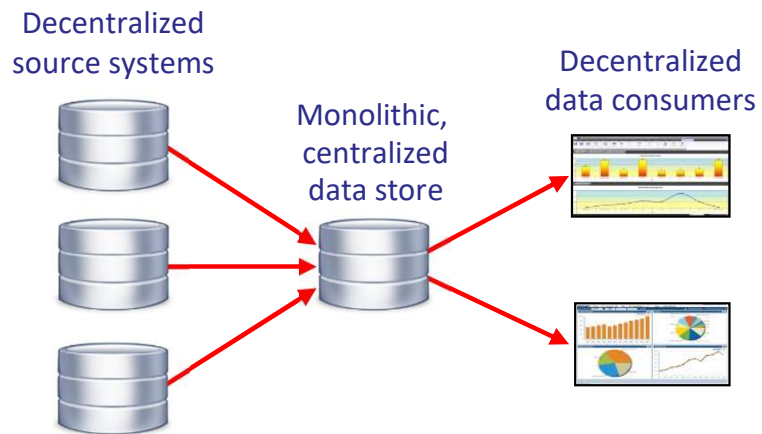


- Most are not developed to offer an interface
- Interfaces that do exist, are commonly developed for record-oriented access
- Direct database access complex or not always allowed
- Rarely support for historic data
- Risky to bypass multi-tenant systems

# Current Centralized, Monolithic Data Architectures

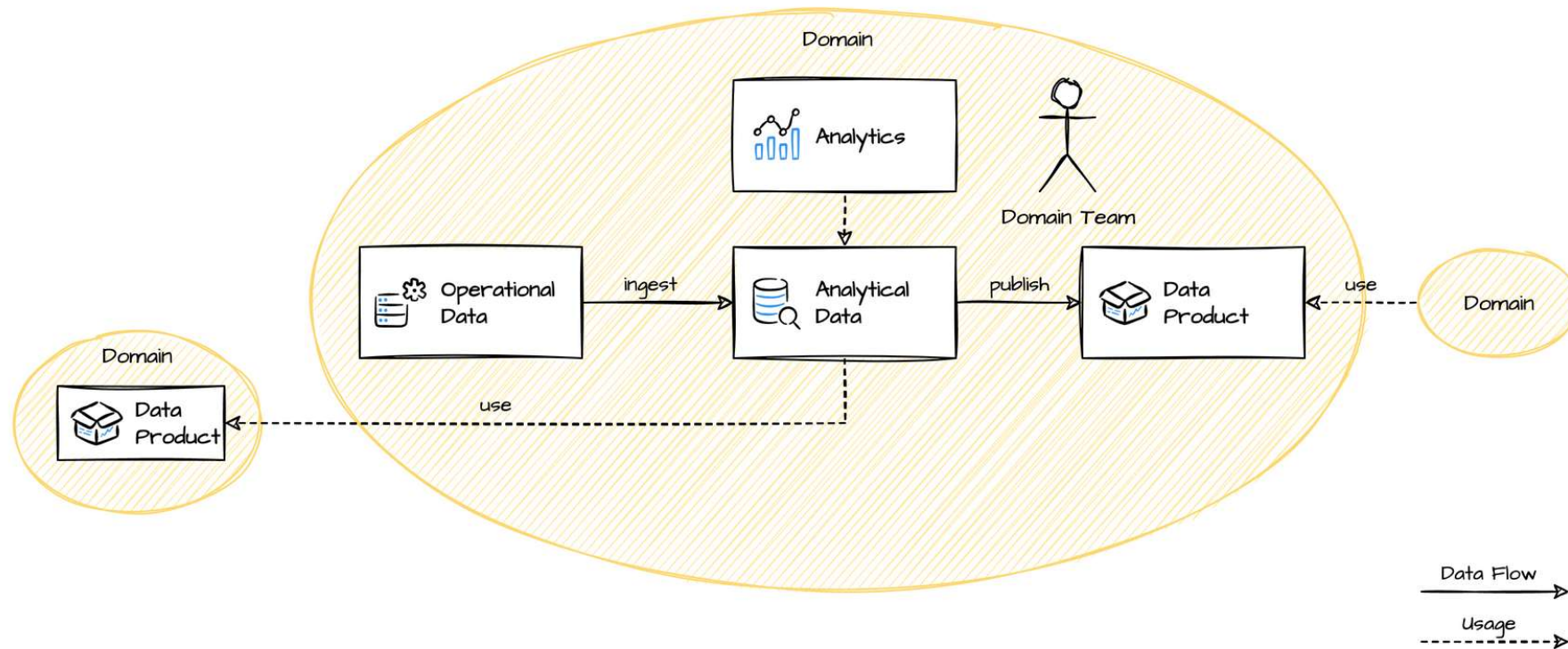


# Traditional Centralized, Monolithic Architectures



- Large and complex monolithic solutions
- Single-domain versus multi-domain experts
  - Application developers = single-domain experts
  - Data engineers = multi-domain experts
  - Data consumers = single-domain experts
- Data engineers need to understand all the business logic
- Who owns the data in the central database?
- Storage of redundant data

# A Domain-Oriented Architecture

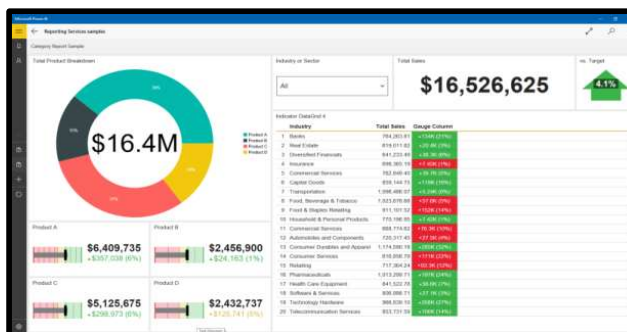


# Potential Data Products

## Data as file

SampleID	Common Name	Description	Keypad	DigitHand	Hand	Individual
M0Key217.1410	keyboard	Akey	Left	NA	Left	M0
M0Bkey217.141032	keyboard	Bkey	Ambiguous	NA	Ambiguous	M0
M0Ckey217.1410	keyboard	Ckey	Left	NA	Left	M0
M0Dkey217.1410	keyboard	Dkey	Left	NA	Left	M0
M0Ekey217.141044	keyboard	Ekey	Left	NA	Left	M0
M0Fkey217.1410	keyboard	Fkey	Right	NA	Right	M0
M0Gkey217.141085	keyboard	Gkey	Left	NA	Left	M0
M0Hkey217.1409	keyboard	Hkey	Left	NA	Left	M0
M0Ikey217.141066	human_skin	finger_tip	NA	Left	Right	M0
M0Jkey217.1410	keyboard	Ikey	Left	NA	Left	M0
M0Kkey217.141043	human_skin	finger_tip	NA	Left	Right	M0
M0Lkey217.141060	human_skin	finger_tip	NA	Right	Right	M0
M0Mkey217.1410	keyboard	Mkey	Right	NA	Right	M0
M0Nkey217.1410	keyboard	Nkey	Right	NA	Right	M0
M0Okey217.1410	keyboard	Okey	Right	NA	Right	M0
M0Pkey217.141035	human_skin	finger_tip	NA	Left	Left	M0
M0Qkey217.141002	human_skin	finger_tip	NA	Right	Right	M0
M0Rkey217.141096	keyboard	Pkey	Right	NA	Right	M0
M0Skey217.1410	keyboard	Qkey	Left	NA	Left	M0
M0Tkey217.141020	human_skin	finger_tip	NA	Left	Right	M0
M0Ukey217.141080	human_skin	finger_tip	NA	Right	Right	M0
M0Vkey217.141004	keyboard	key	Left	NA	Left	M0
M0Wkey217.1411	keyboard	space_bar	Ambiguous	NA	Ambiguous	M0
M0Xkey217.1410	human_skin	finger_tip	NA	Left	Right	M0
M0Ykey217.1410	human_skin	finger_tip	NA	Right	Right	M0
M0Zkey217.1410	keyboard	Vkey	Left	NA	Left	M0
M0Akey217.1410	keyboard	Wkey	Left	NA	Left	M0
M0Bkey217.141023	keyboard	Xkey	Left	NA	Left	M0
M0Ckey217.141023	keyboard	Ykey	Right	NA	Right	M0

## Report



## Service

```

{
  "people": [
    {
      "friends": [
        "/people/2/",
        "/people/3/"
      ],
      "username": "steveluscher",
      "email": "steveluscher@fb.com",
      "last_name": "Luscher",
      "id": "1",
      "first_name": "Steven"
    },
    {
      "friends": [
        "/people/1/",
        "/people/4/"
      ],
      "username": "aholovaty",
      "email": "a.holovaty@django.com",
      "last_name": "Holovaty",
      "id": "2",
      "first_name": "Adam"
    }
  ]
}
    
```

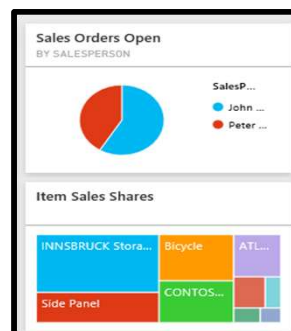
## Data via SQL



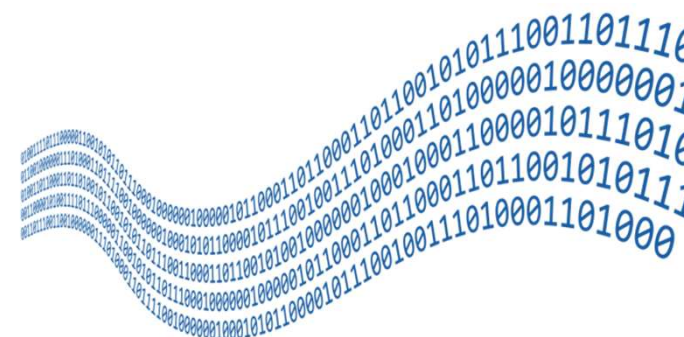
## Apps



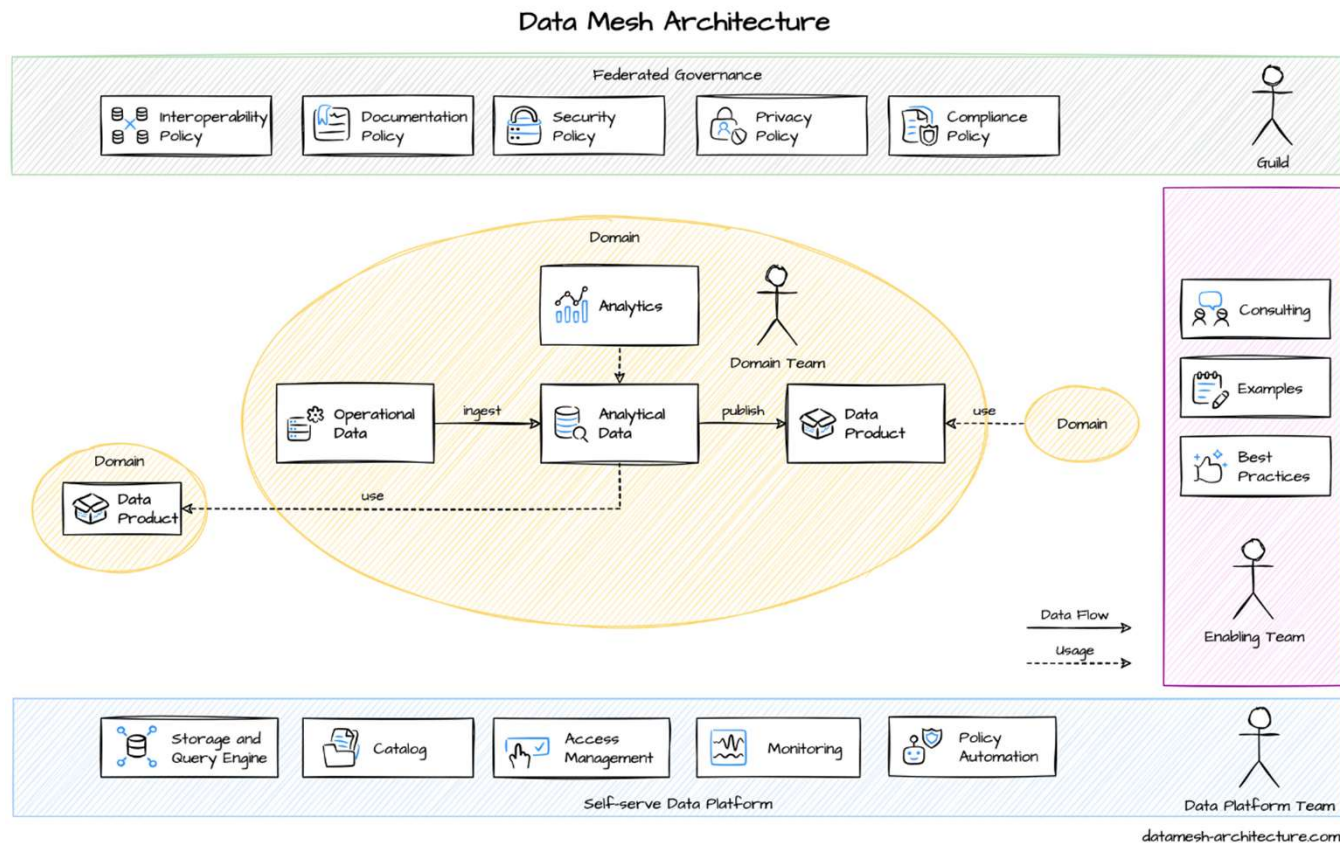
## Embeddable KPI



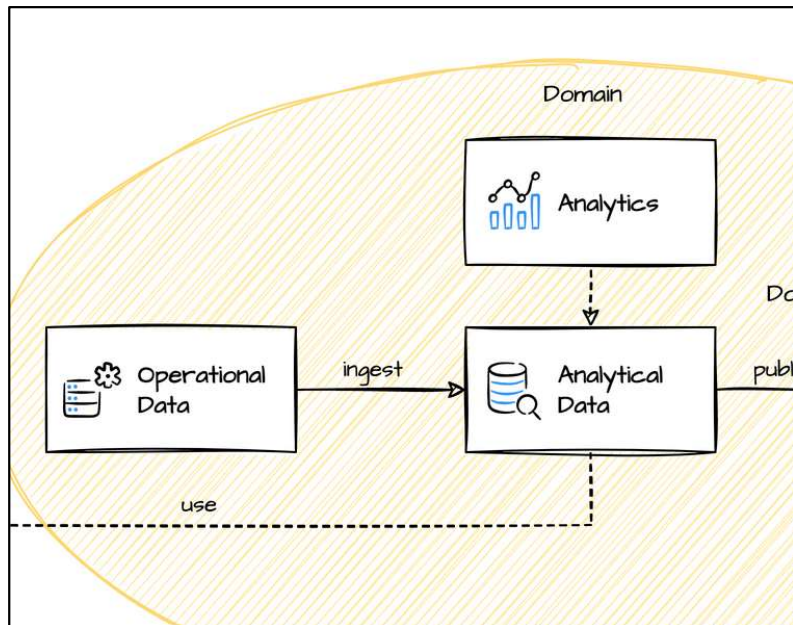
## Stream of Data



# The Periphery of a Domain



# What is Ingest?



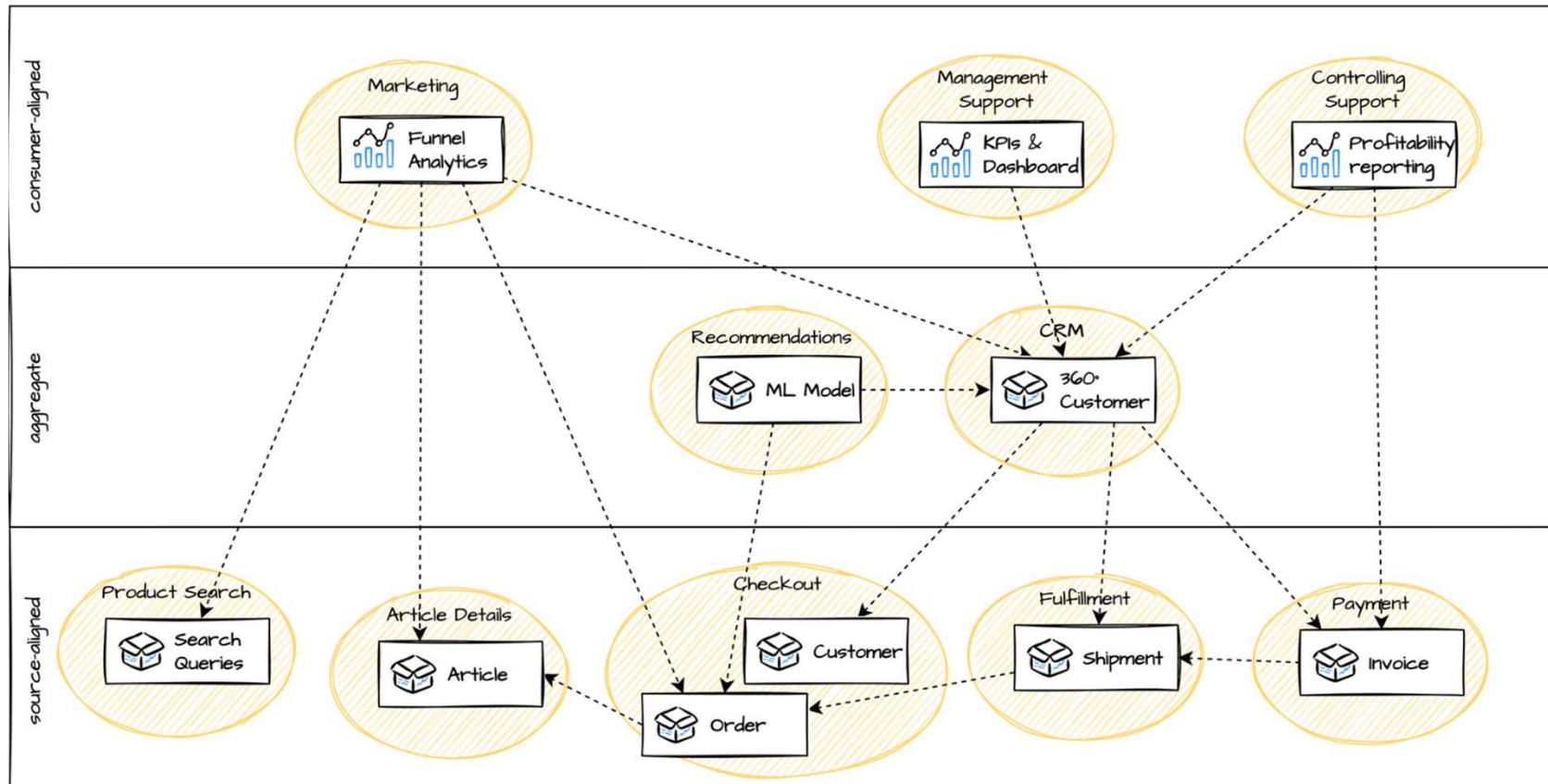
## ■ Potential technologies:

- ETL
- Data replication (Change Data Capture)
- ESB (Enterprise Service Bus)
- Messaging
- Database triggers
- Data virtualization

## ■ Involves transformers

- Transforming data values and structure
- Masking
- Cleansing
- Calculations
- ...

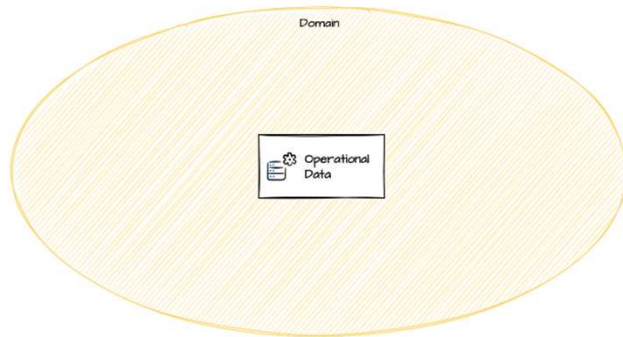
# The Mesh Itself



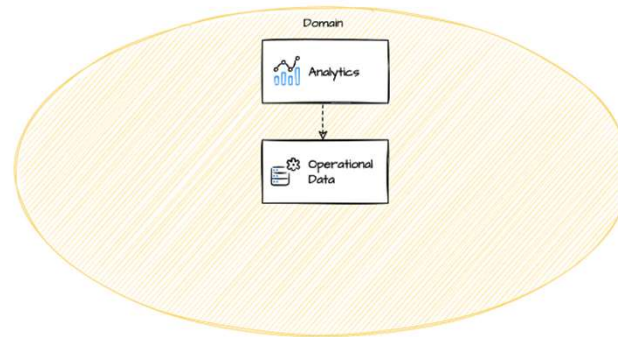
datamesh-architecture.com

# The Domain Team's Journey

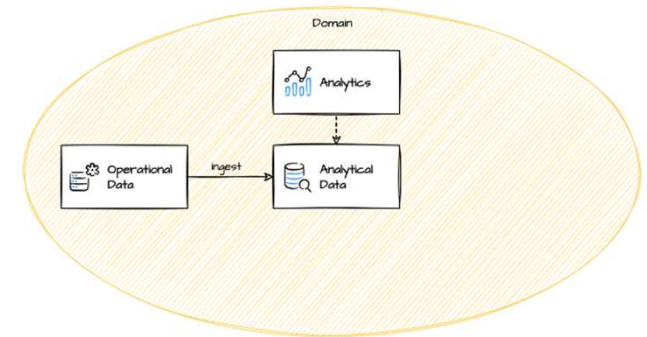
No data analytics



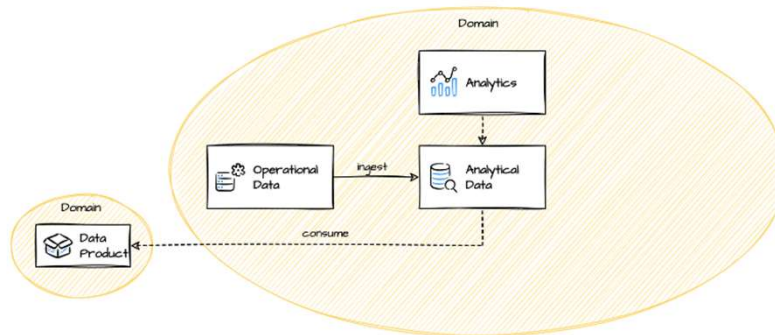
Operational database queries



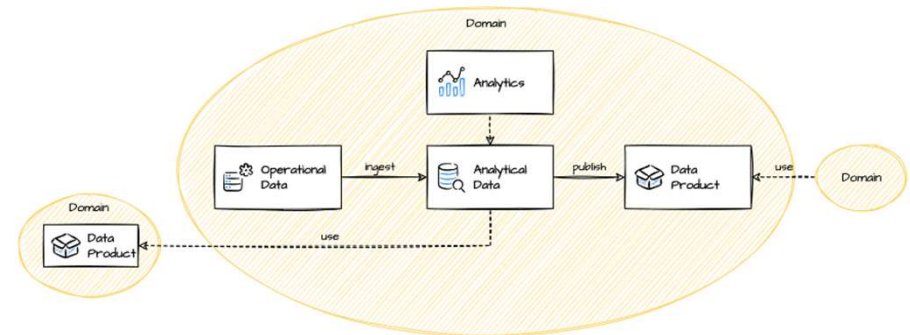
Analyze own data



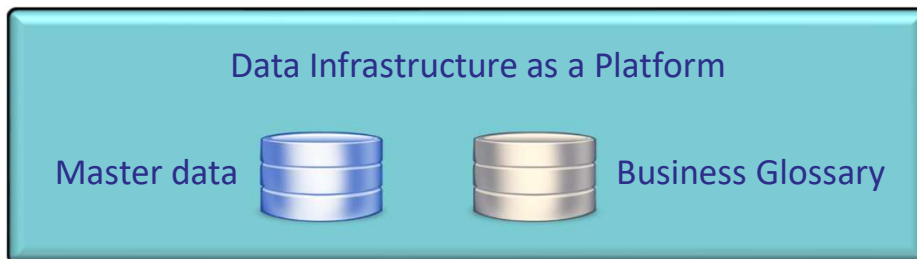
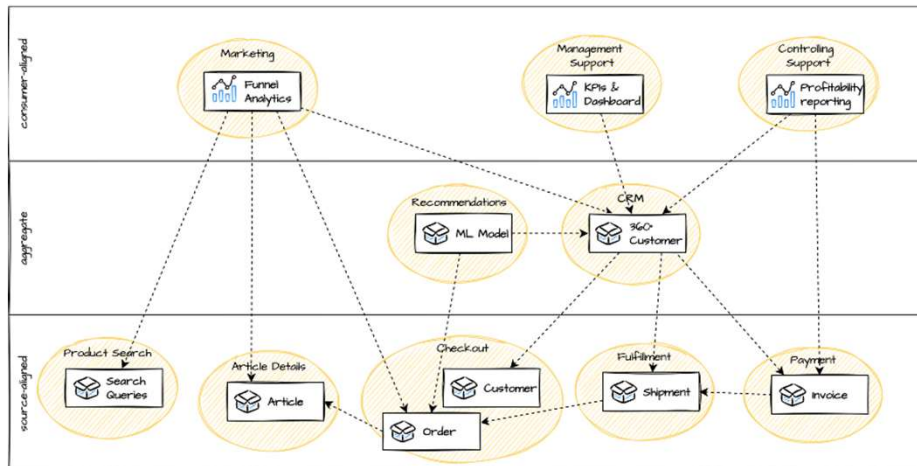
Analyze Cross-domain data



Publish data as a product

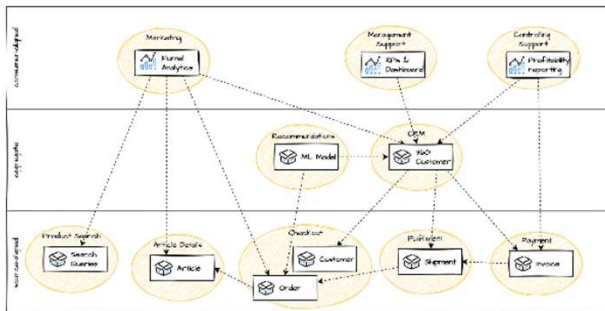


# The Foundation: Data Infrastructure as a Platform



- Scalable polyglot big data storage
- Encryption for data at rest and in motion
- Data product versioning
- Data product schema
- Data product de-identification
- Unified data access control and logging
- Data pipeline implementation and orchestration
- Data product discovery, catalog registration and publishing
- Data governance and standardization
- Data product lineage
- Data product monitoring/alerting/log
- Data product quality metrics (collection and sharing)
- In memory data caching
- Federated identity management
- Compute and data locality
- ...

# Data Mesh Challenges

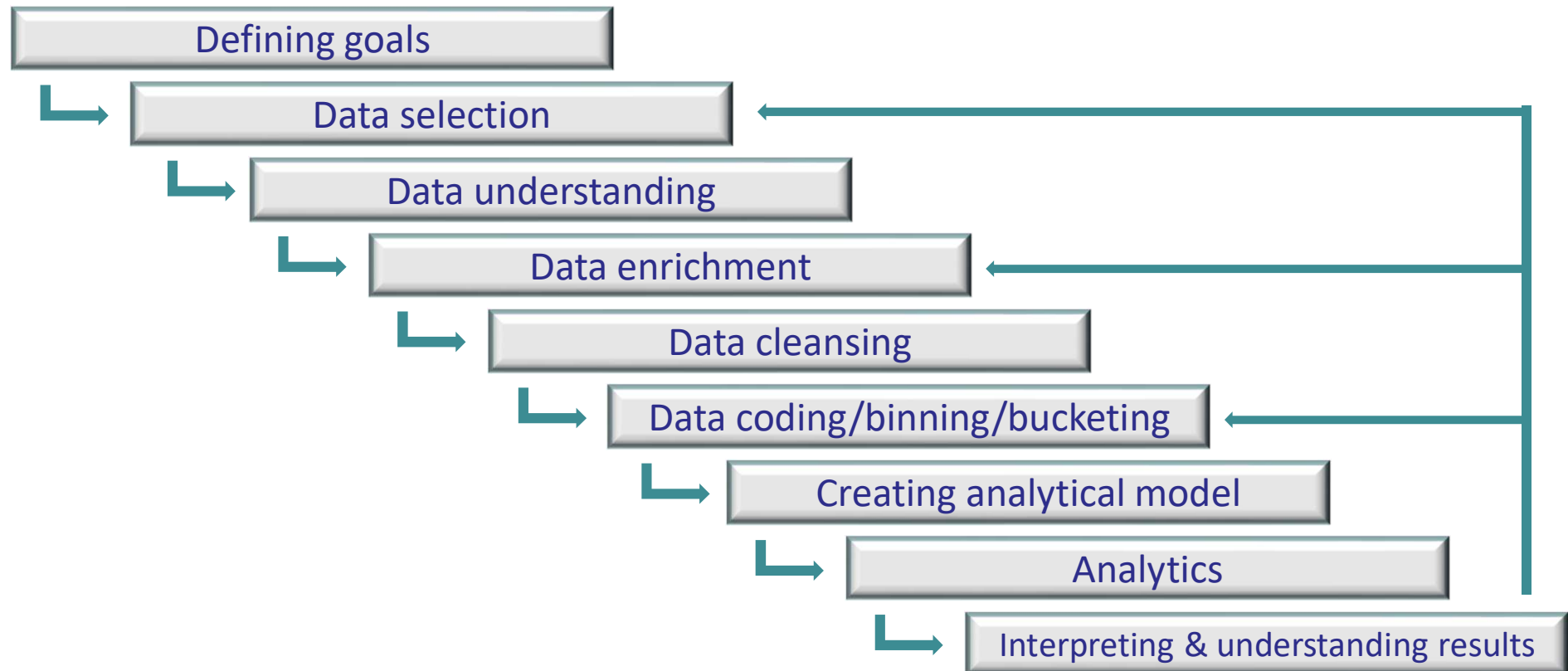


- Massive organizational change for IT
- Standardization of interfaces required
- How clear are BI requirements when new transactional applications are developed?
- Development of transactional applications more complex
- What about transactional applications that are bought?
  - Do they need to be wrapped?
- What about multi-domain transactional applications?
- Distribution of knowledge of data delivery technologies

**Part 7.8:  
Embedding Data Science Models in  
Data Architectures**

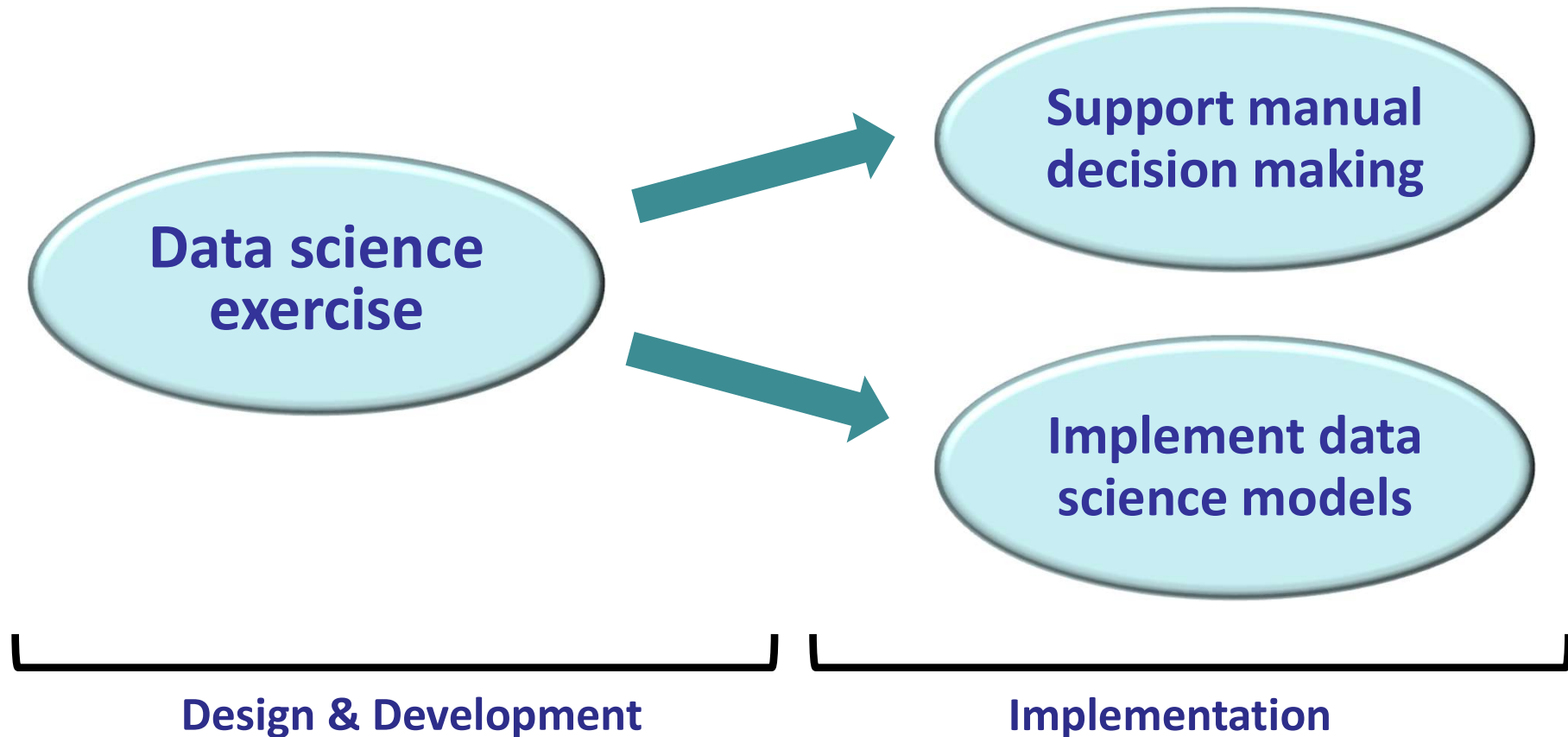


# Development Steps for Data Science



# Operationalization of Data Science Models

---



# Operationalization of Data Science Models

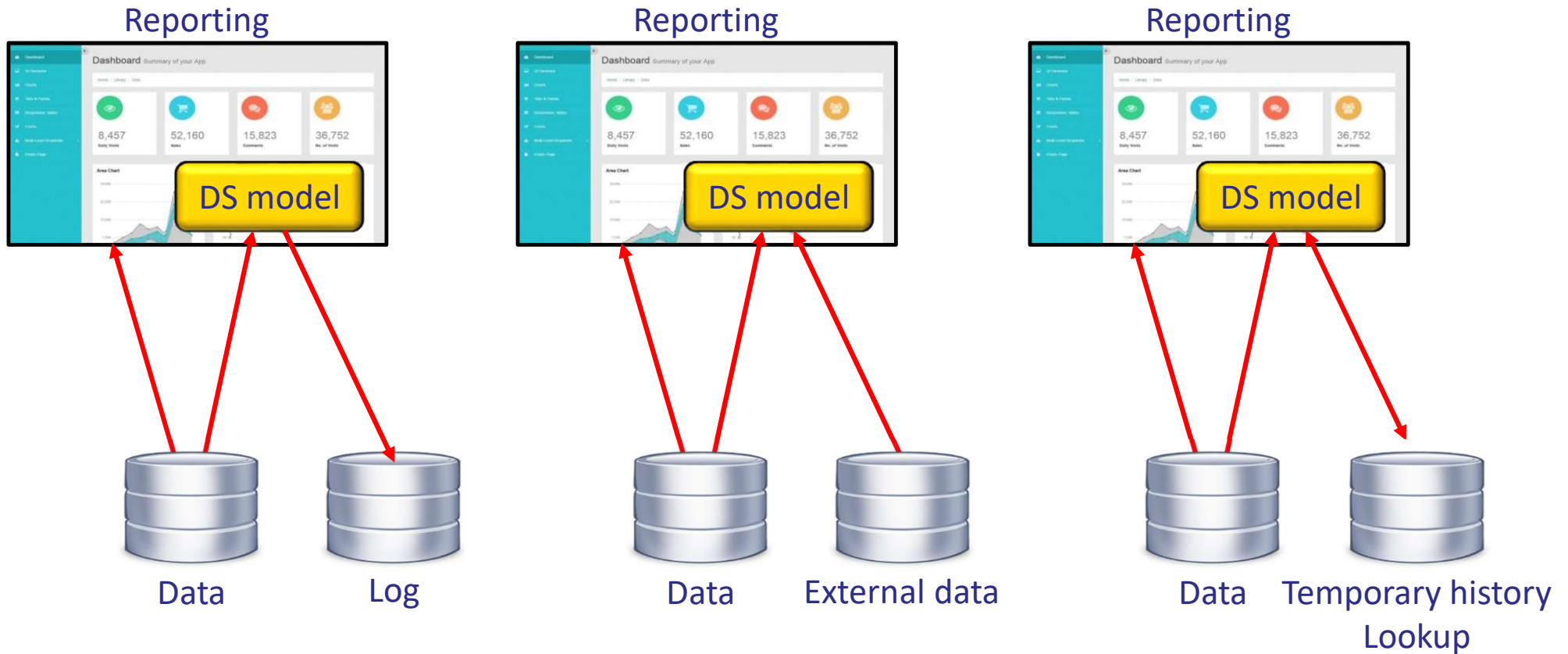
Type of Decision	Example
Singular manual decision	What will be the impact on total sales of acquiring company X?
Repeatable manual decision	Does a specific location have the right characteristics for opening a new shop?
Partial automated decision	In a call center: What is the churn risk for a customer? What should we offer?
Full automated decision without automated reaction	When credit card payment is dubious, send message to operator
Full automated decision with automated reaction	When sensor indicates the component is heating up too fast, switch off machine
...	...

# Requirements for a Supporting Data Architecture

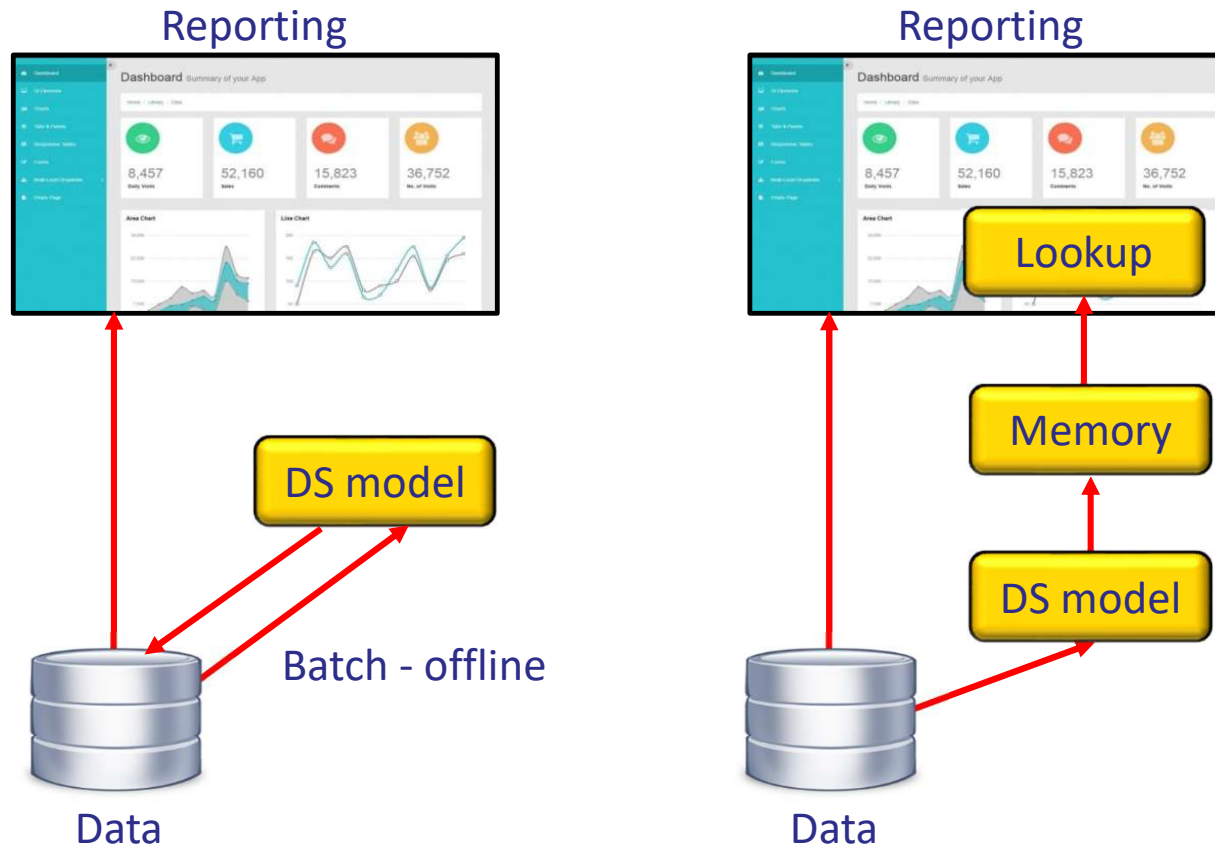
---

- Versioning of data science models
  - Immutable models
- Auditable data science models
  - Reproducible data for reproducible models
  - Transparency of models
- Different codings must be easy and quick to apply
- Self-learning models or not?
- Delivering metadata
  - Descriptions, definitions, tags, relationships, searchable
- Fast evaluation of models
  - Max time to execute model, SLAs
- And many more ...

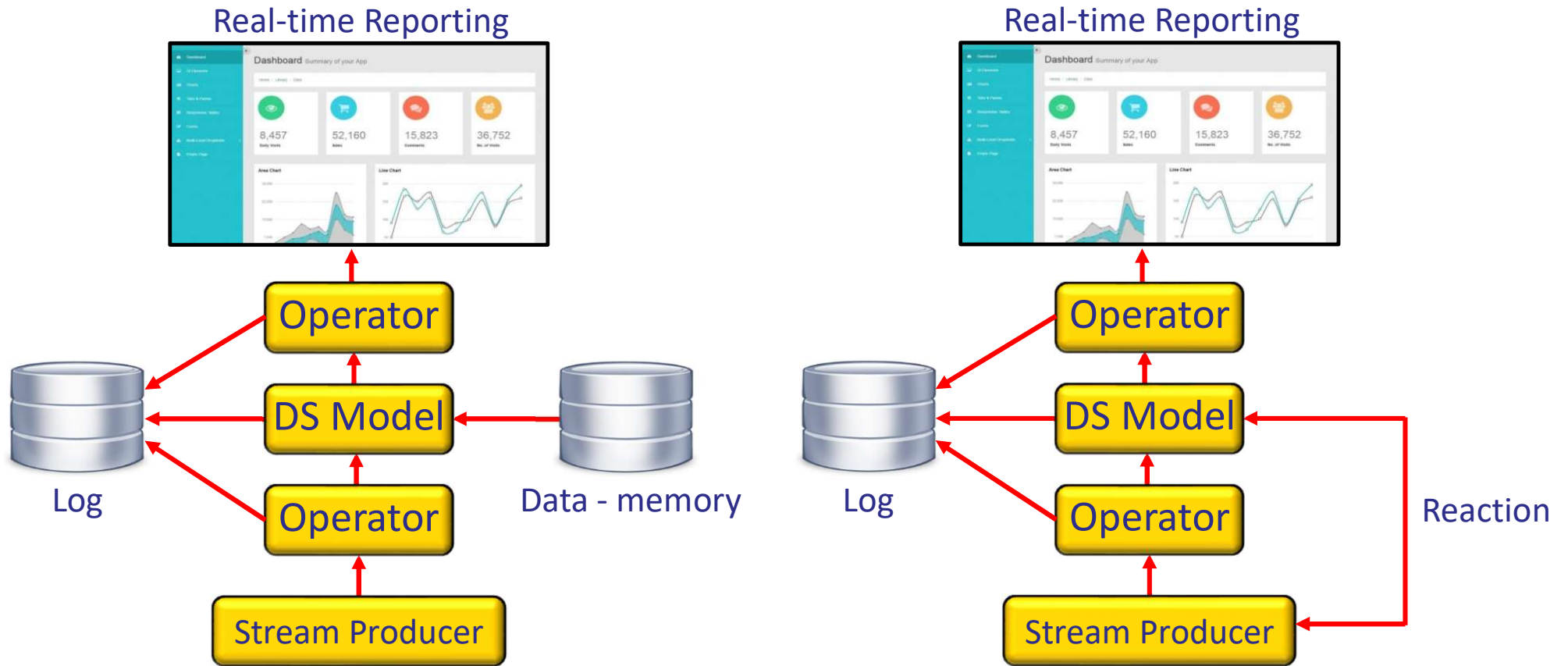
# Architectural Aspects (1)



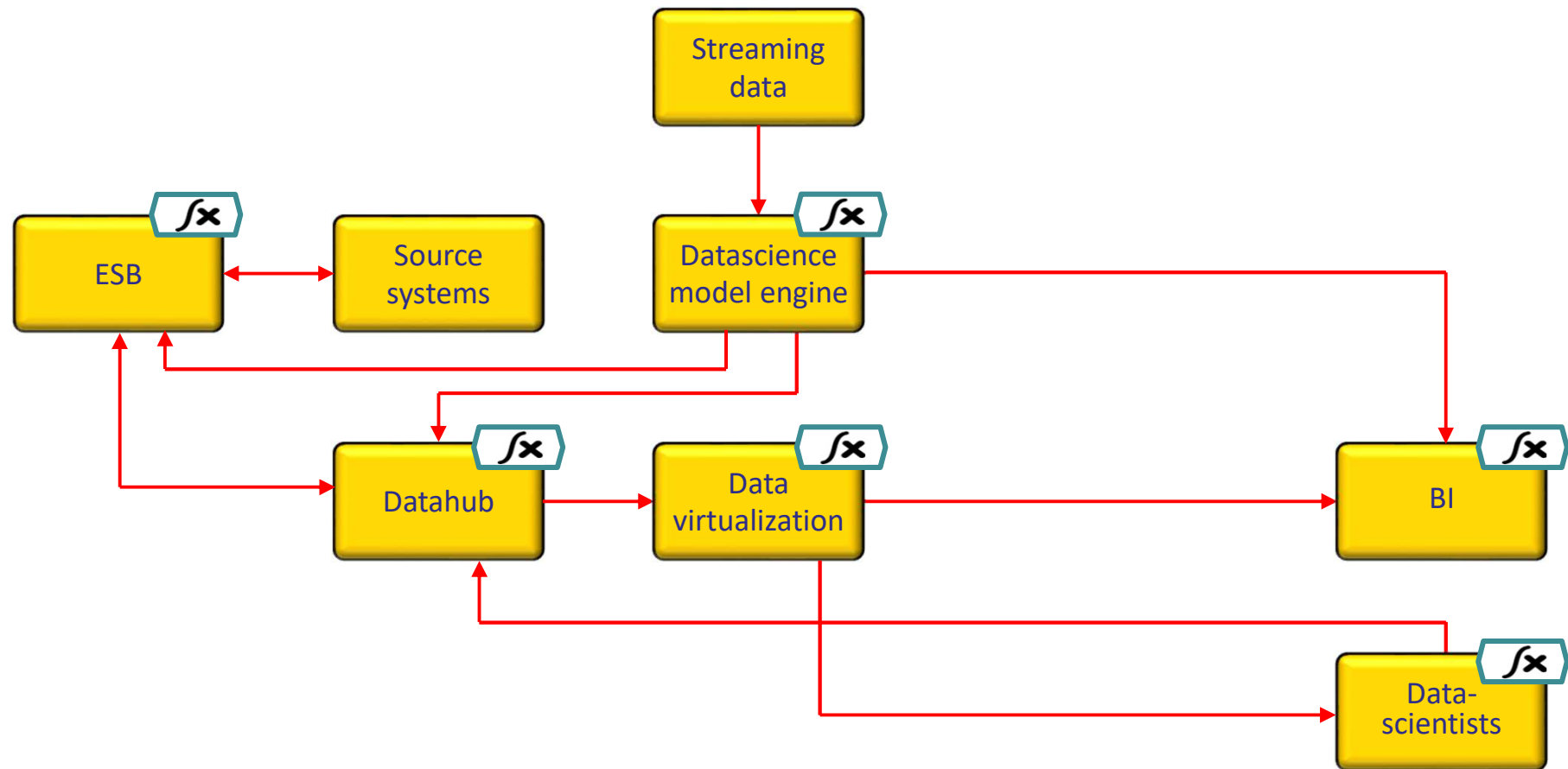
# Architectural Aspects (2)



# Architectural Aspects (3)



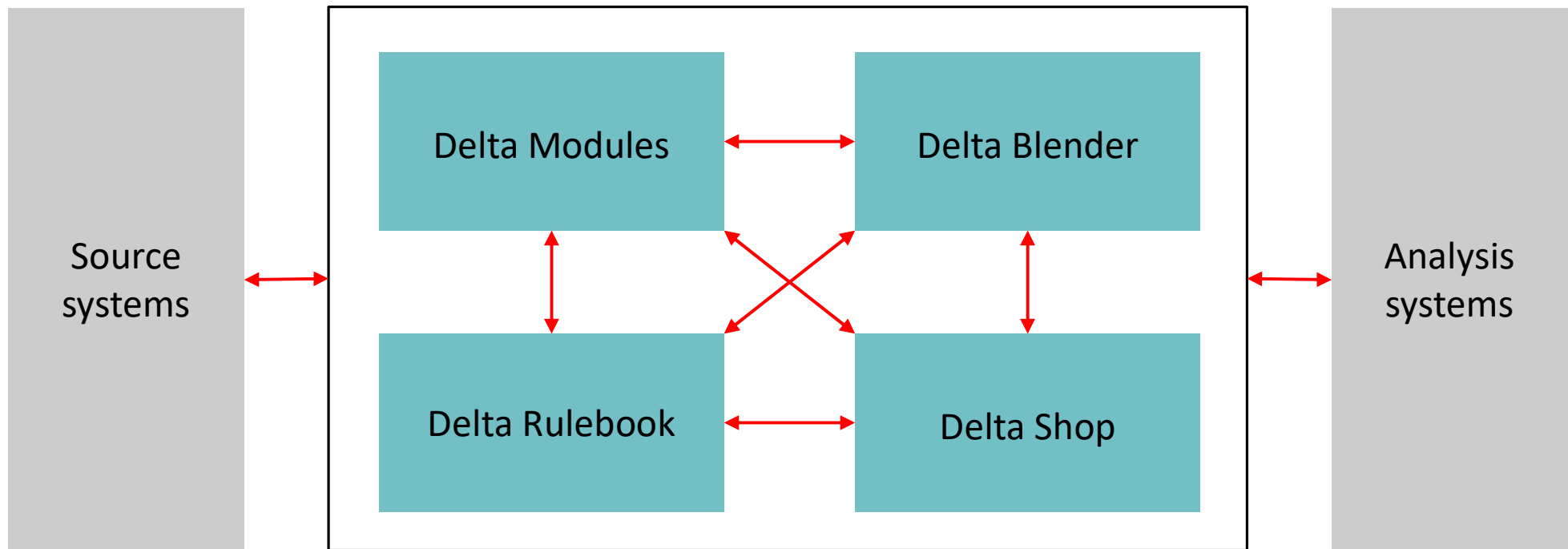
# Example of a Data Architecture



**Part 7.9:  
The Delta Data Architecture  
Under Development**



# Delta: A Modern, Enterprise Data Architecture

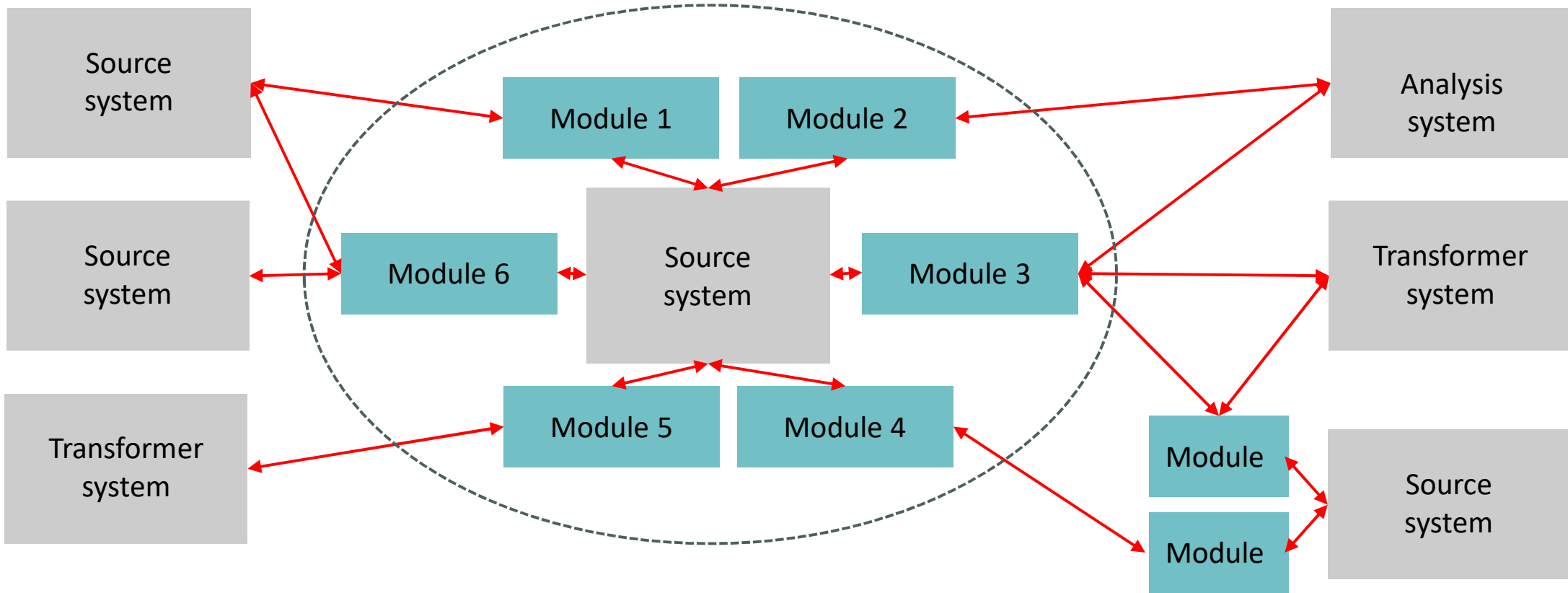


# Functions of Source Systems

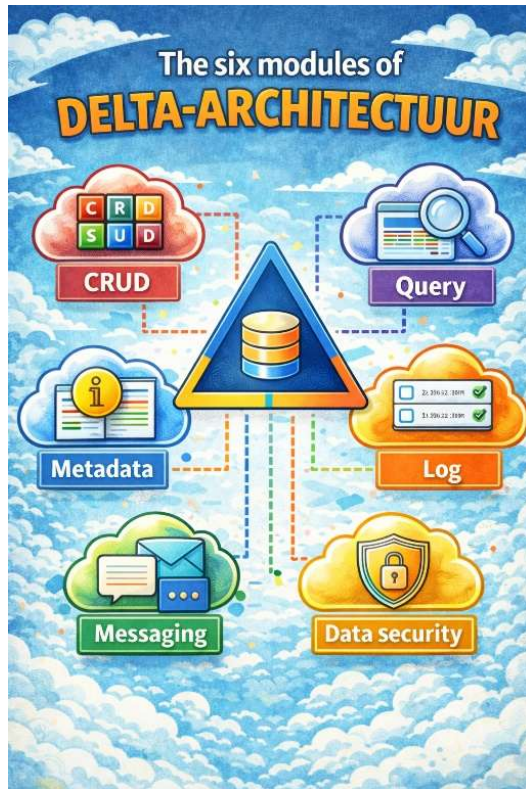
---

- Entering, modifying, and deleting data
  - For many business objects
- Retrieving data
- Securing data
- Data quality
- Backup and recovery tasks
- Calculating results
- Handling workflow and processes
- Triggering events
- *Connections with other systems*
- ...

# Wrapping the Source Systems - Abstraction

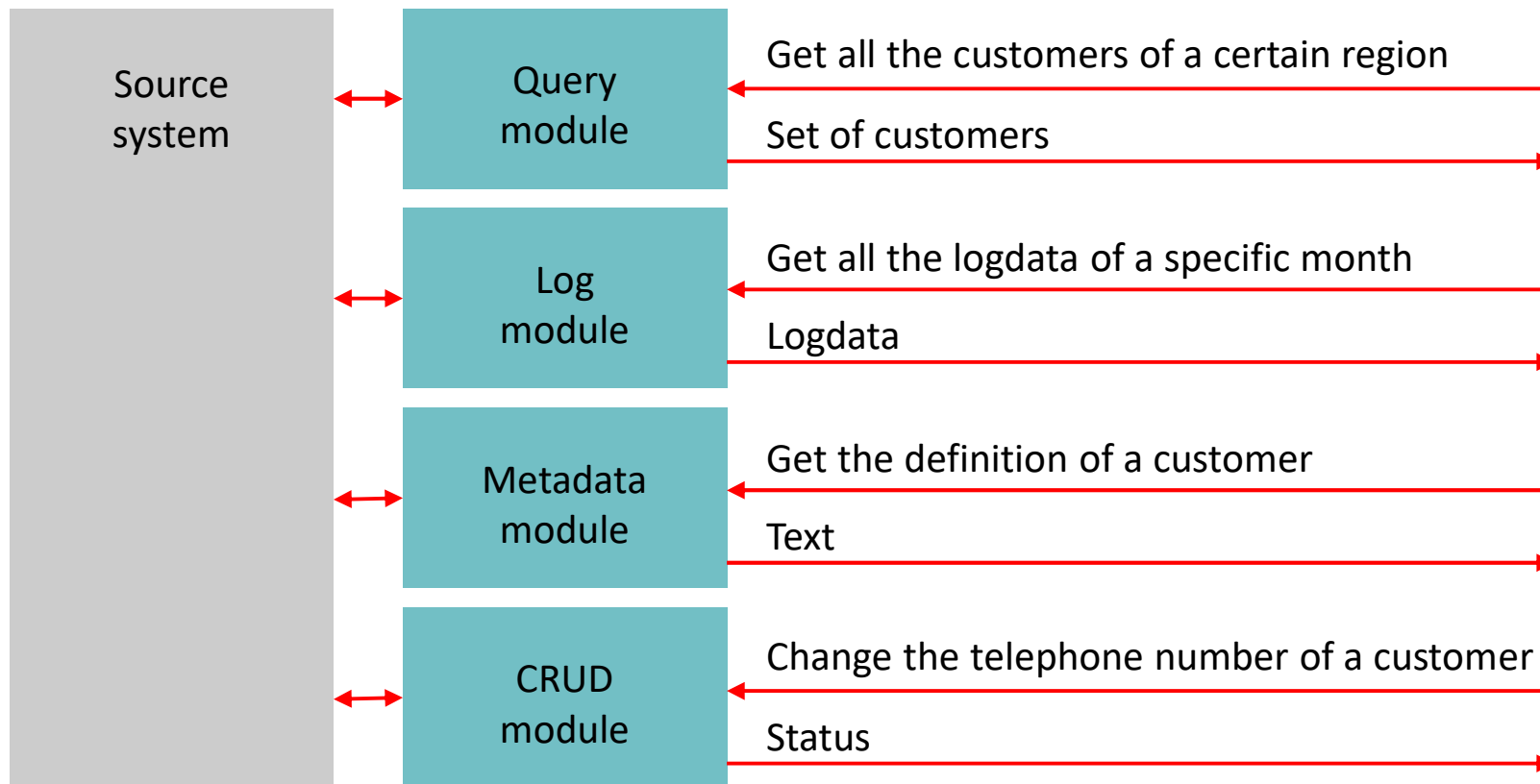


# Six Delta Modules to Wrap Source Systems



- CRUD
- Query
- Metadata
- Log
- Messaging
- Data security

# Modules and Functions

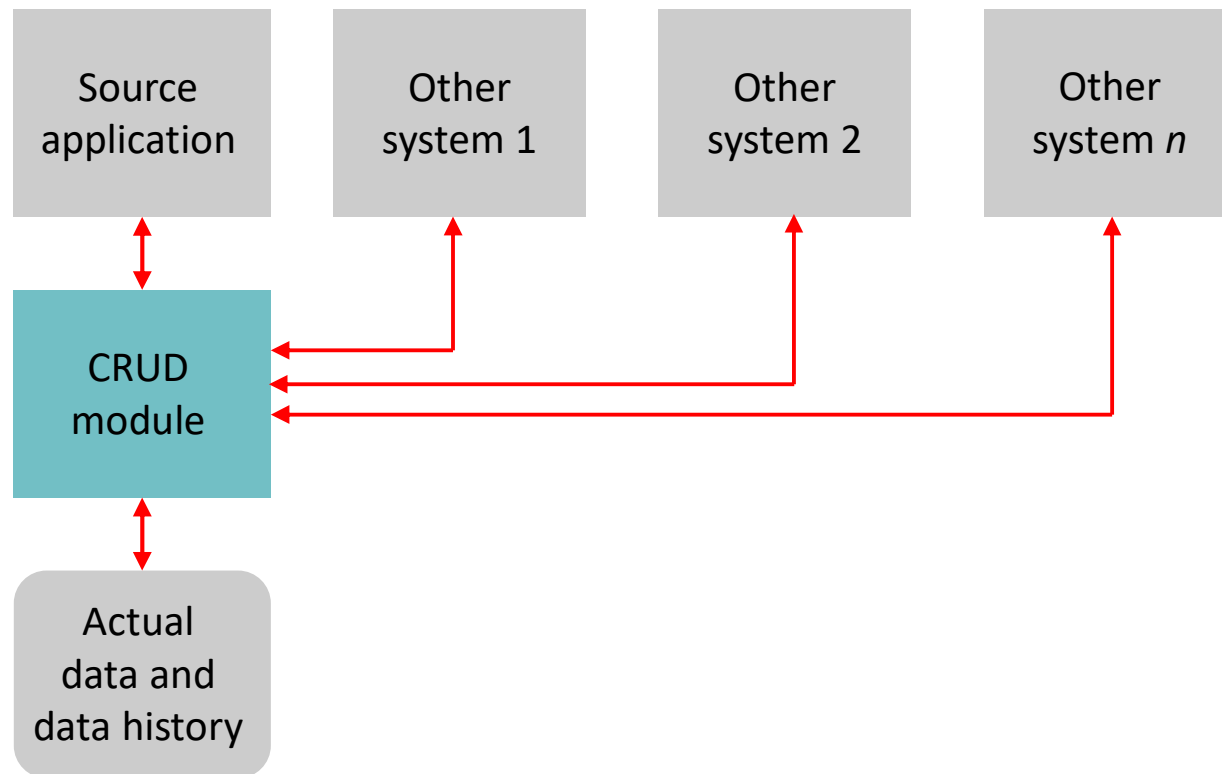


# Functions of CRUD modules

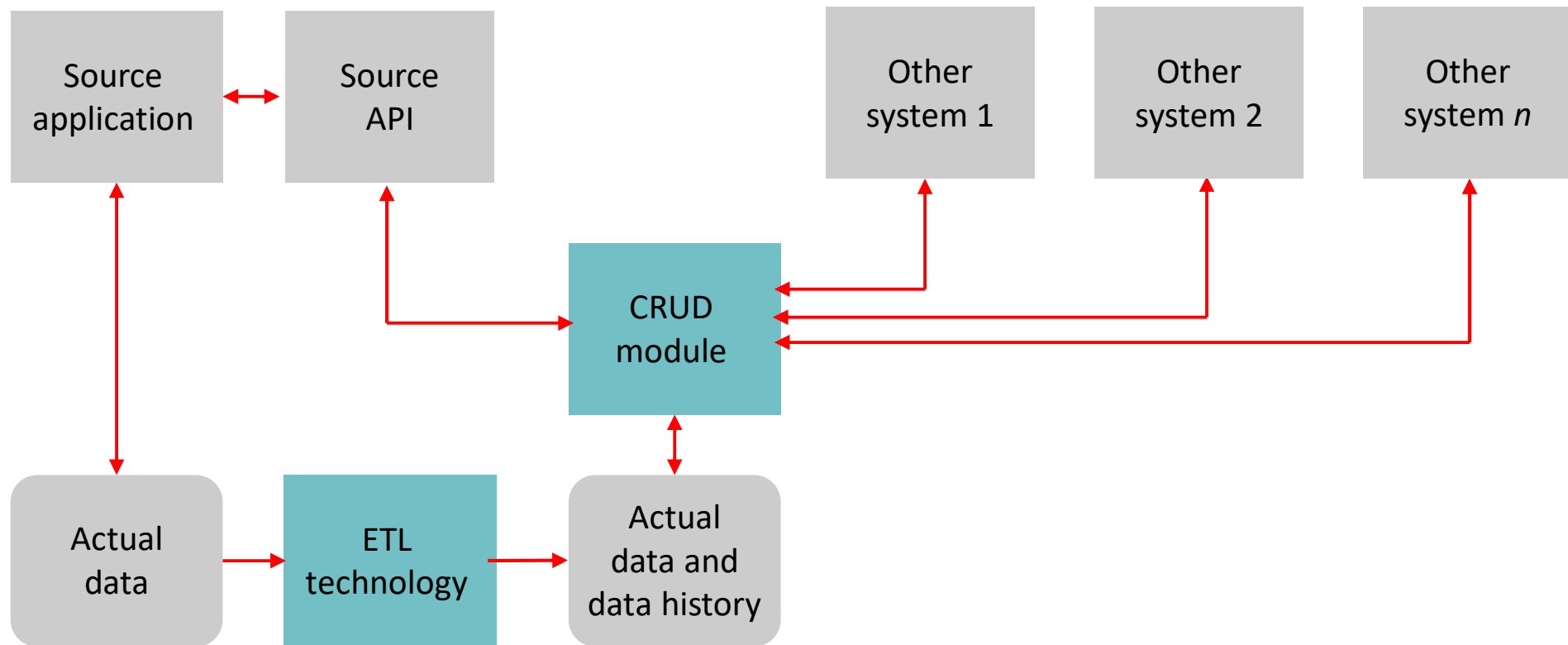
---

- Offer functions to enter, view, modify, and delete data
- Hide internal and external source systems and files
- Data complies with the Enterprise Data Model
- Support time travel
  - Maintains data history if the source system does not
  - Two time dimensions: transaction time and real time
- Work with individual business objects
- Enforce data quality rules
- Involved in source system synchronization
- Replaces data exchange with data sharing
- ...

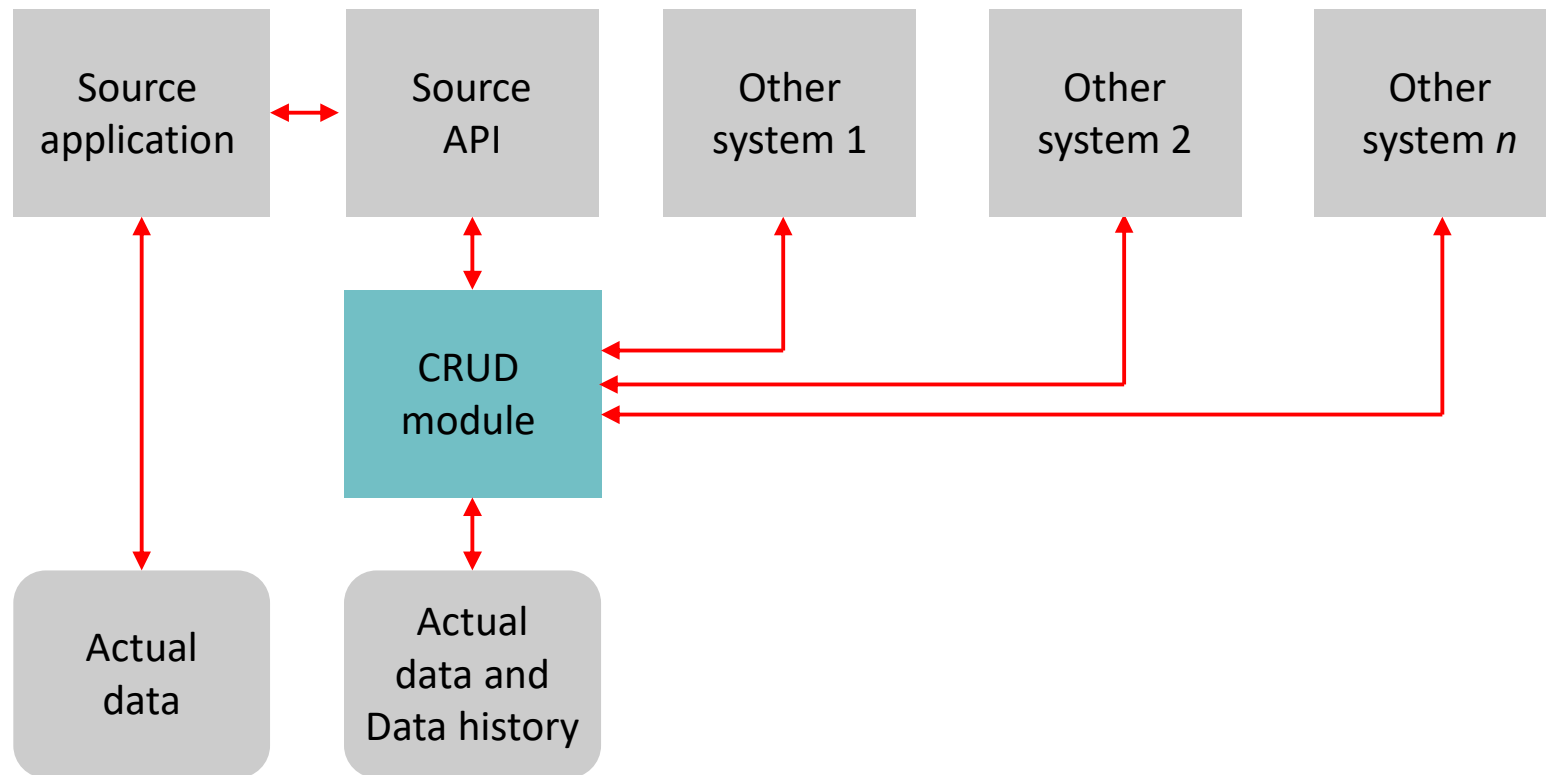
# CRUD Module (1) The Dream



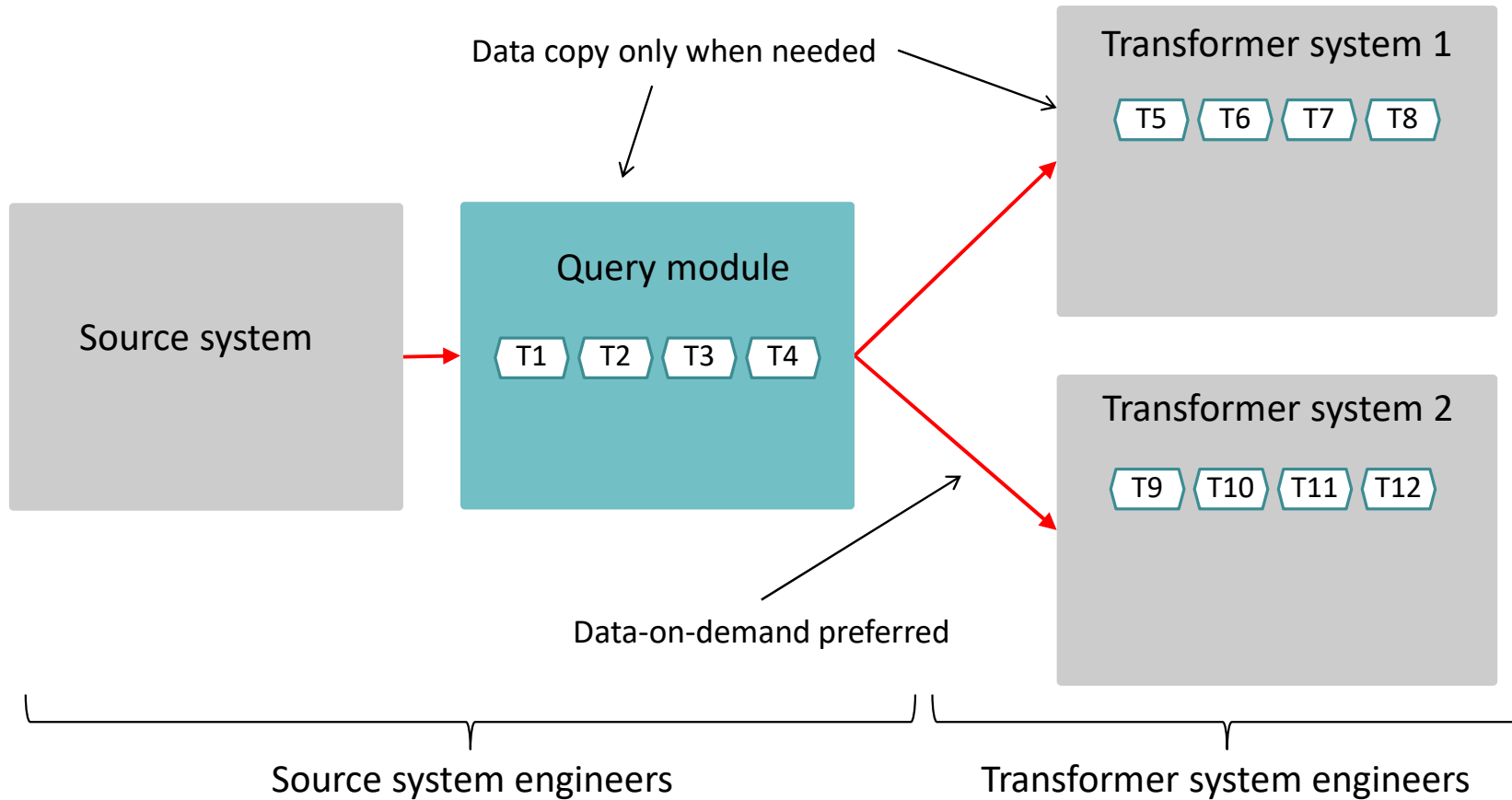
# CRUD Module (2) Via API of Source System and ETL



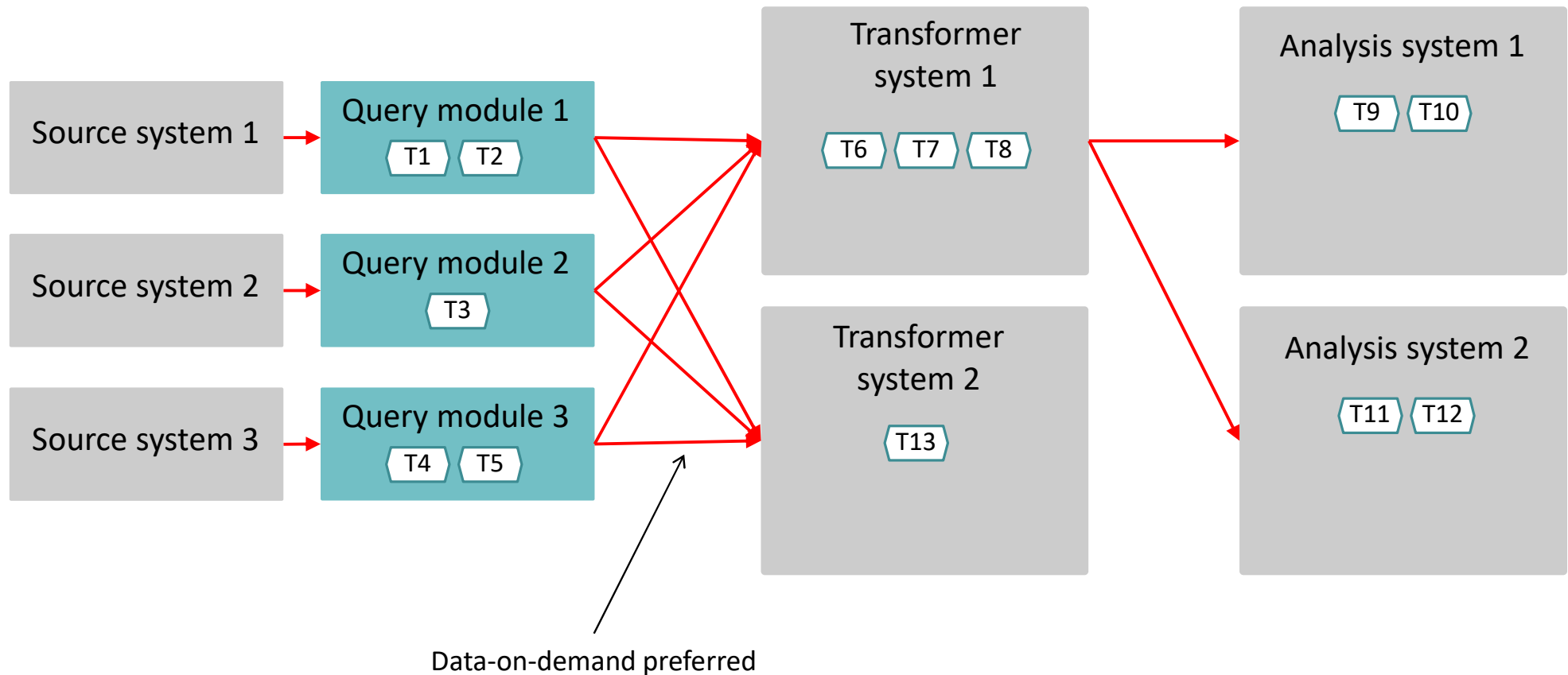
# CRUD Module (3) Via API of Source System



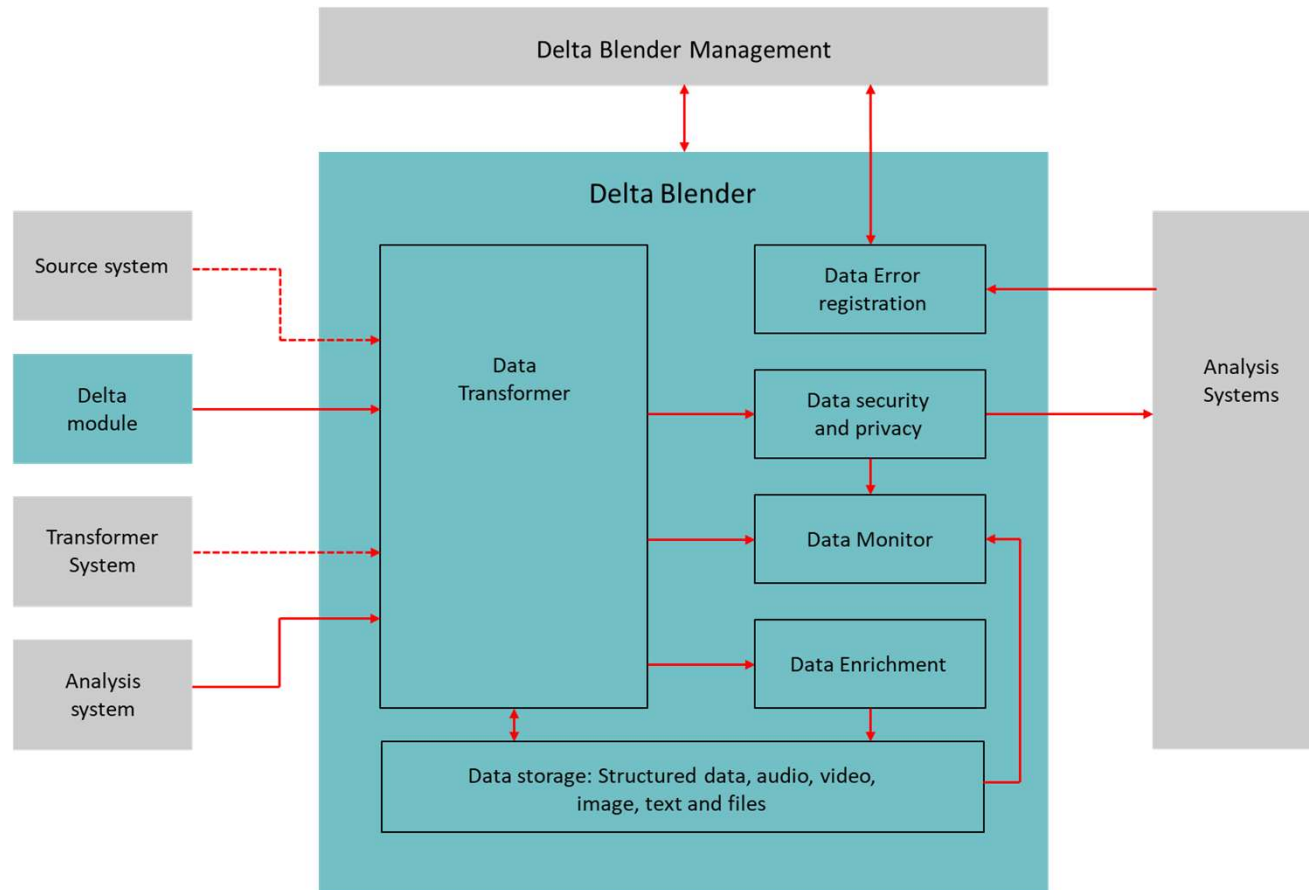
# Transformers Closer to the Source Systems



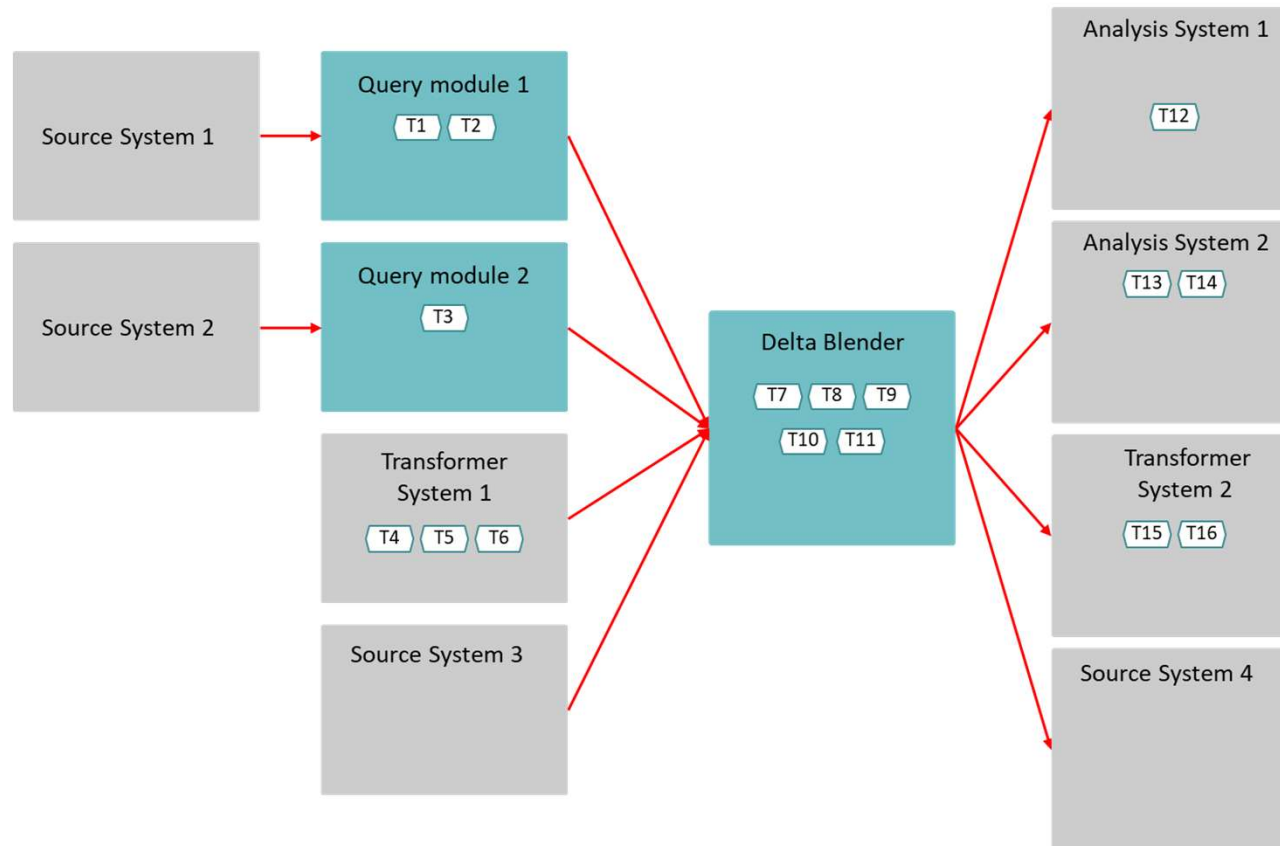
# Transforming Data in Query Modules



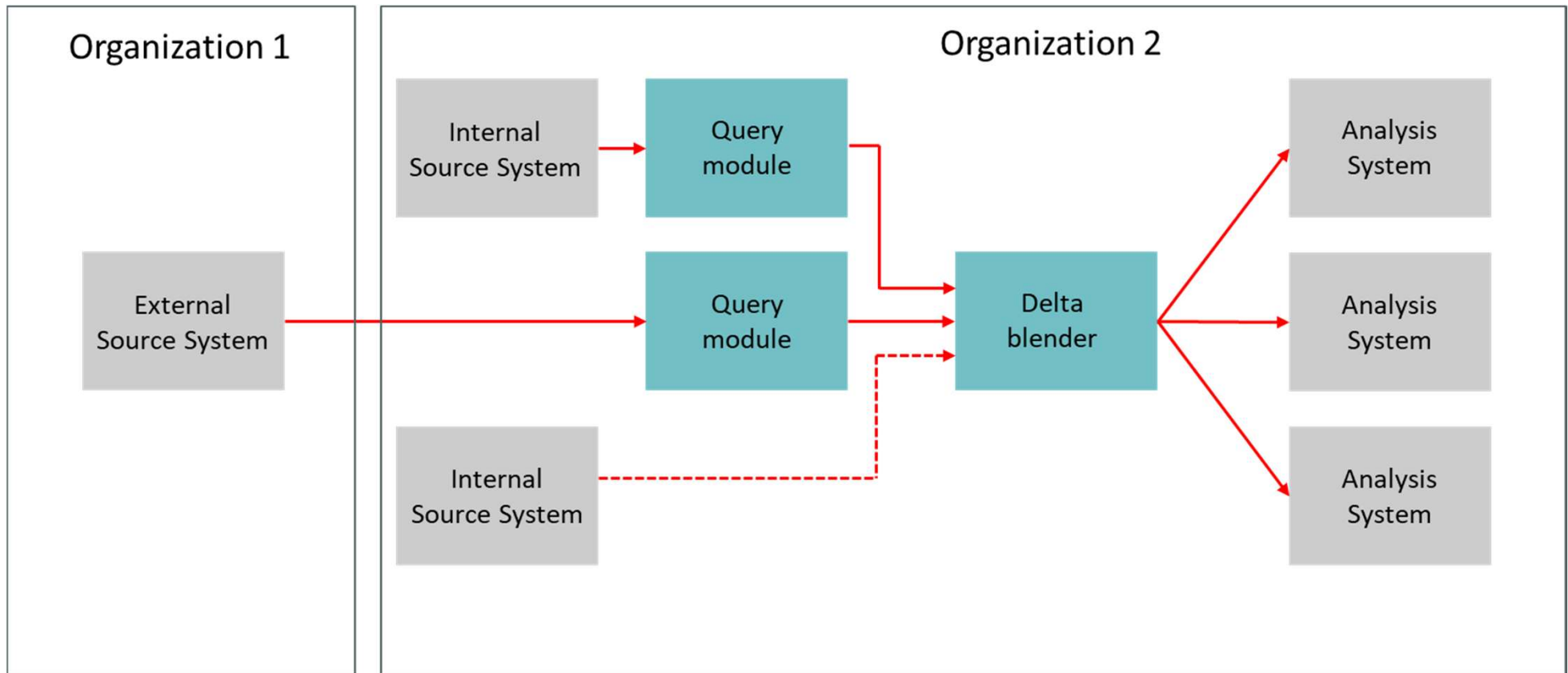
# Delta Blender: Produces Ready-to-use Data



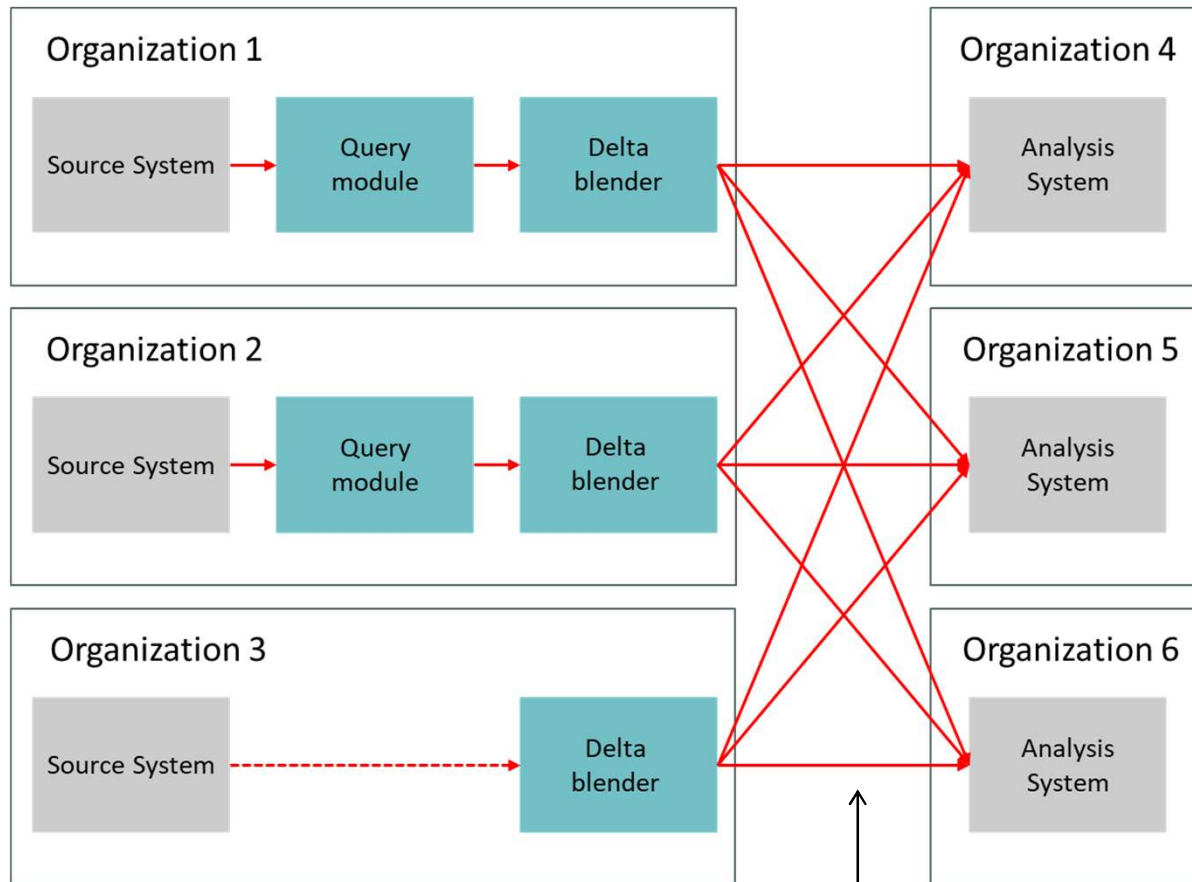
# Delta Blender: Transformers



# Local Federated Data Architecture

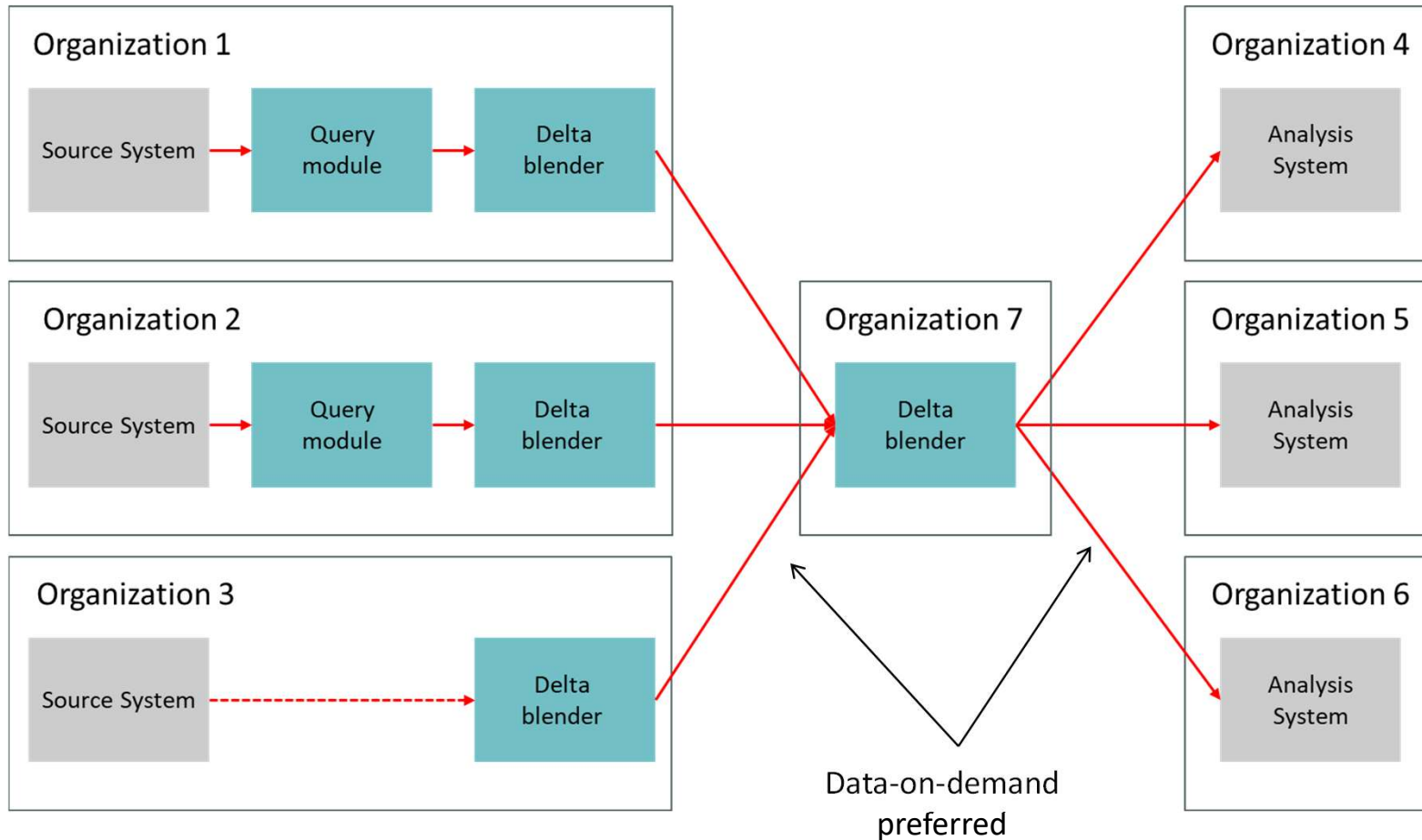


# No Federated Data Architecture

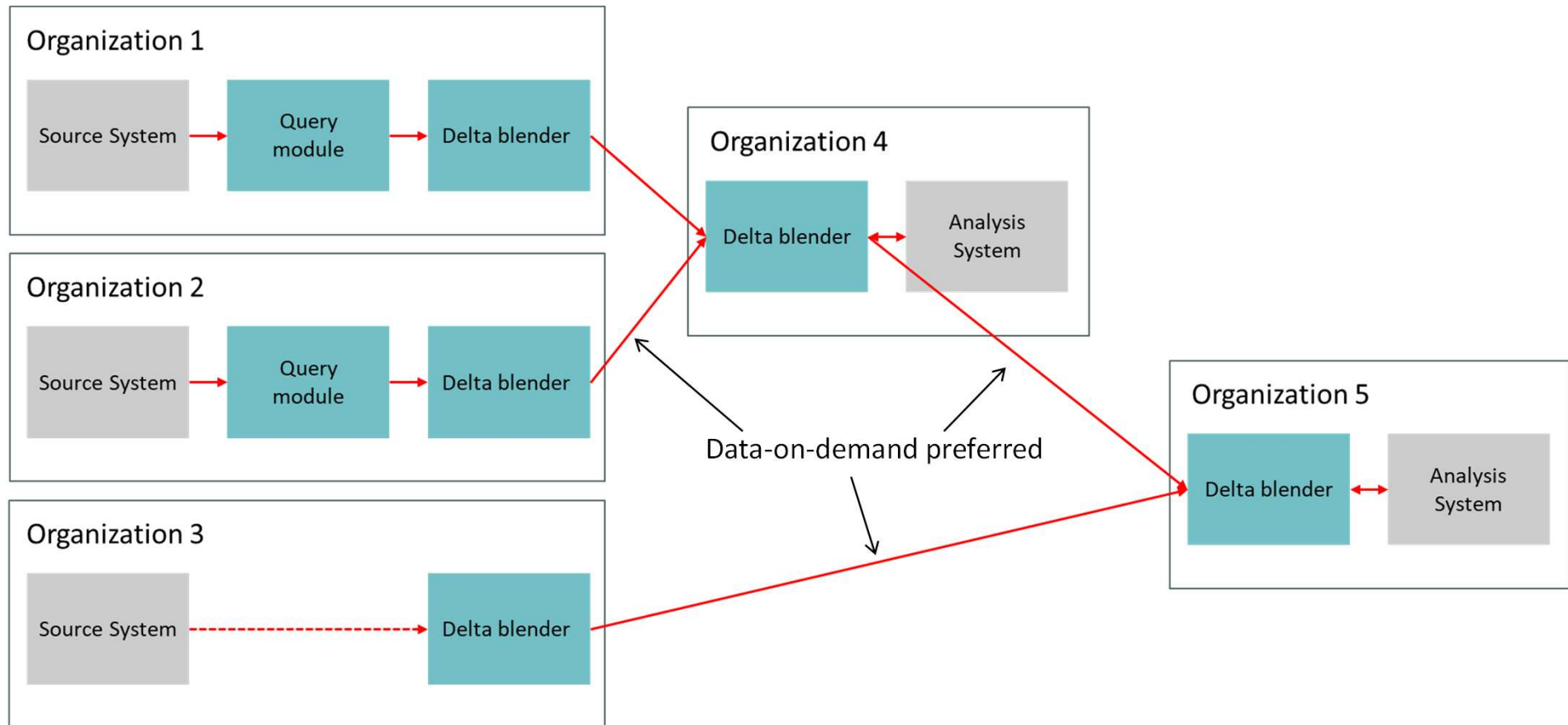


Data-on-demand preferred

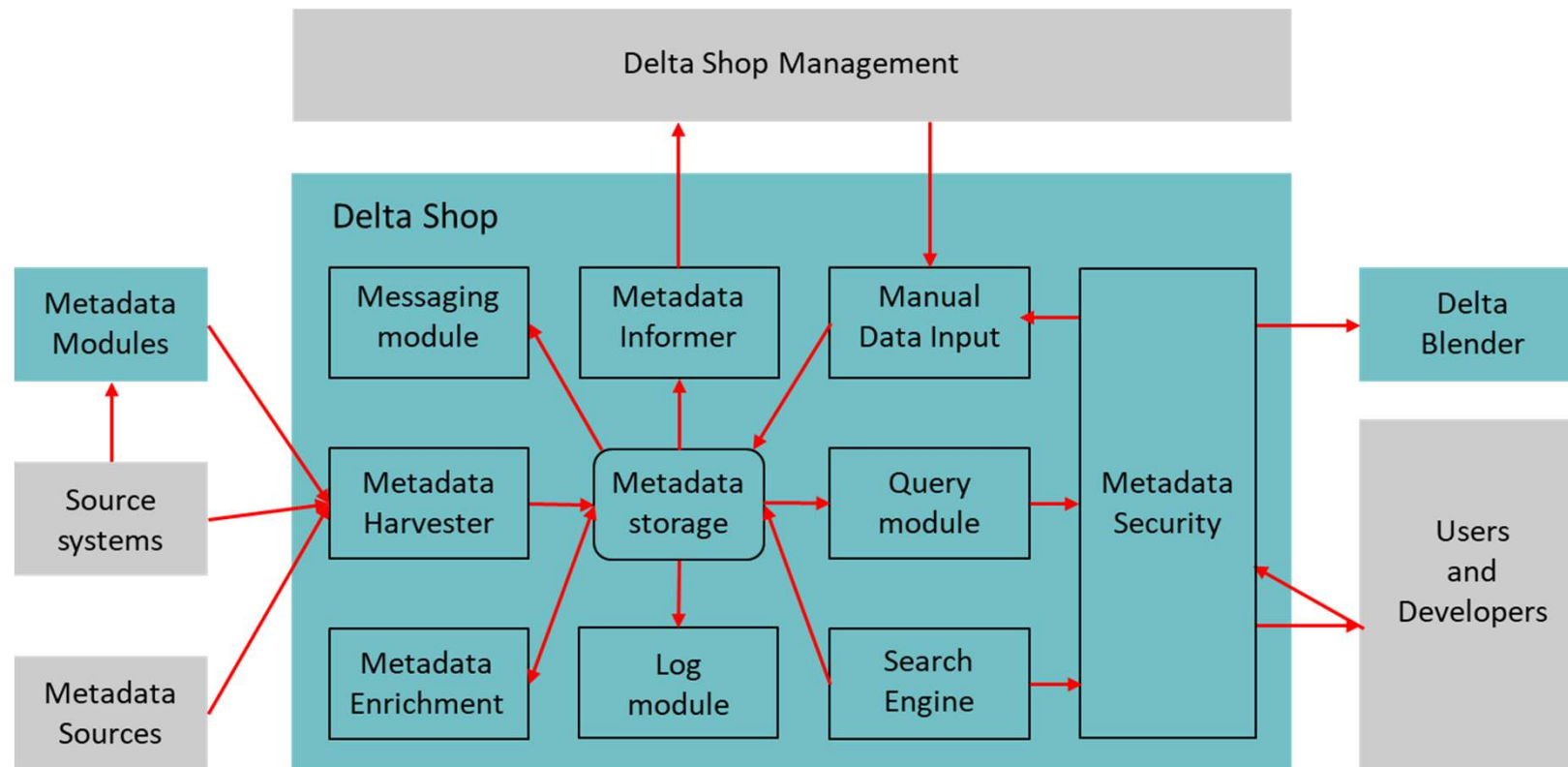
# Global Federated Data Architecture



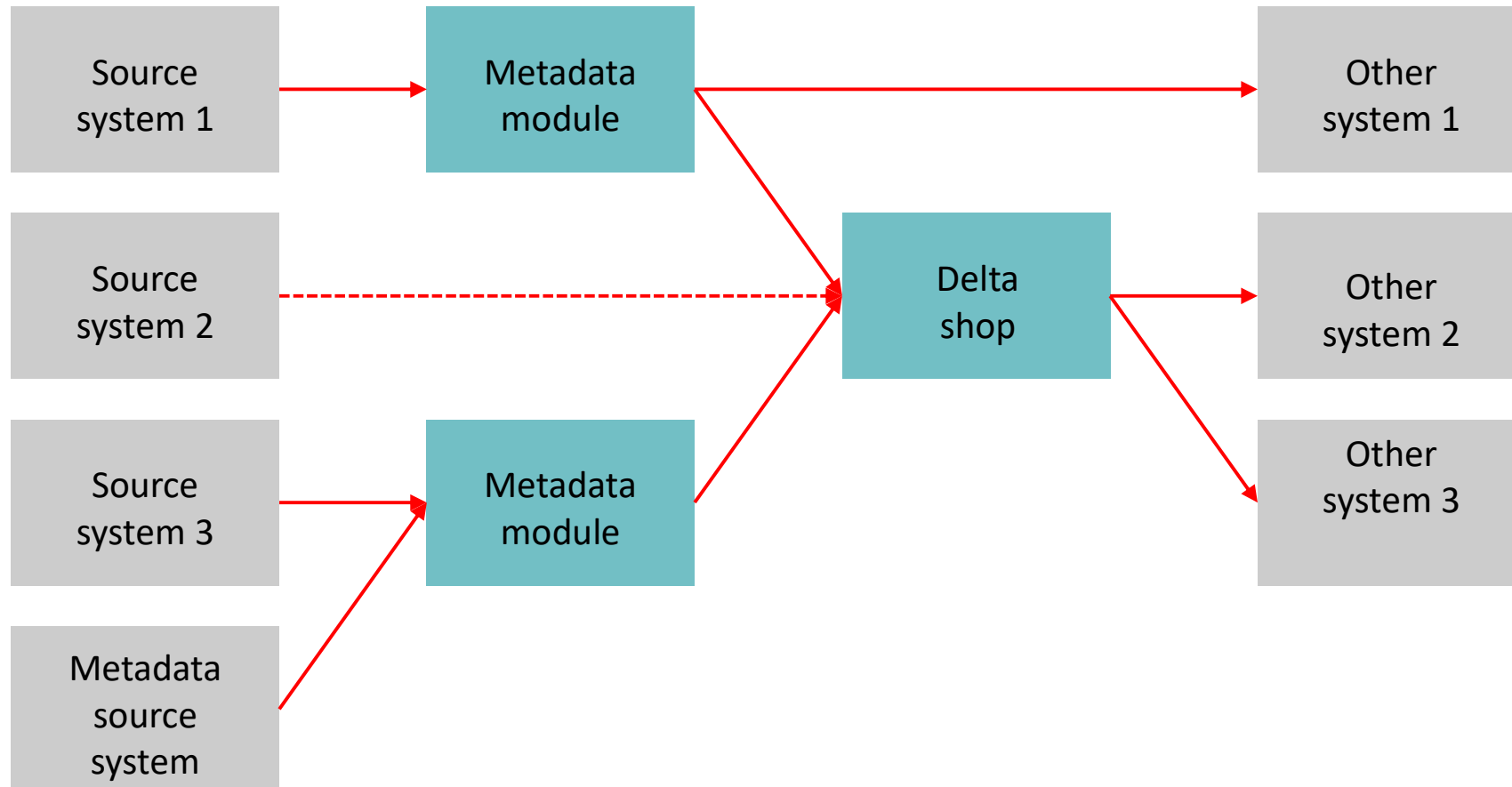
# Layered Global Federated Data Architecture



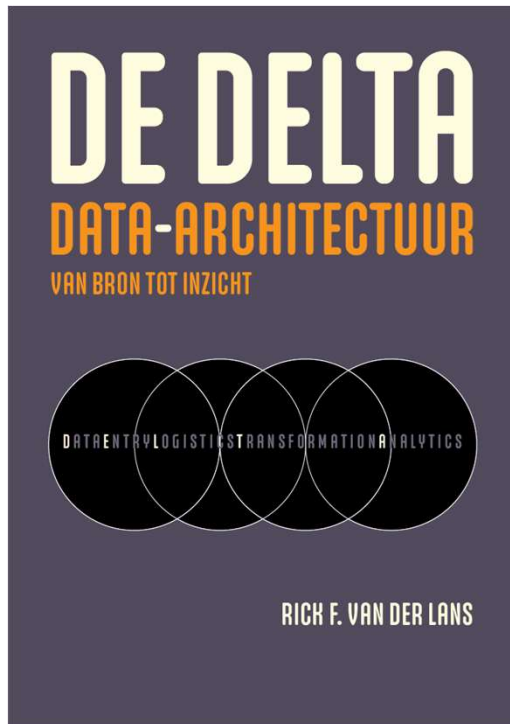
# Delta Shop Components - Metadata



# Metadata Modules and the Delta Shop



# Summary of the Delta Data Architecture



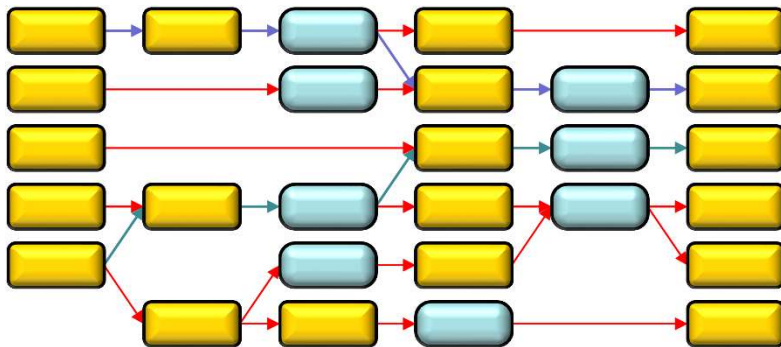
- Enterprise data architecture
  - From source to insight
  - Data must be ready-to-use, discoverable, and insightful
  - It's about the transformers, not the databases
  - Metadata for everyone
- Federated data architecture
  - Unbridled creation of data copies must stop
  - Data-on-demand, not data-by-copy
  - Avoid duplicate transformers
- Transparent data architecture
  - Operational lineage for reconstruction, transparency, and auditability
- Modular data architecture
  - Co-exist with existing solutions



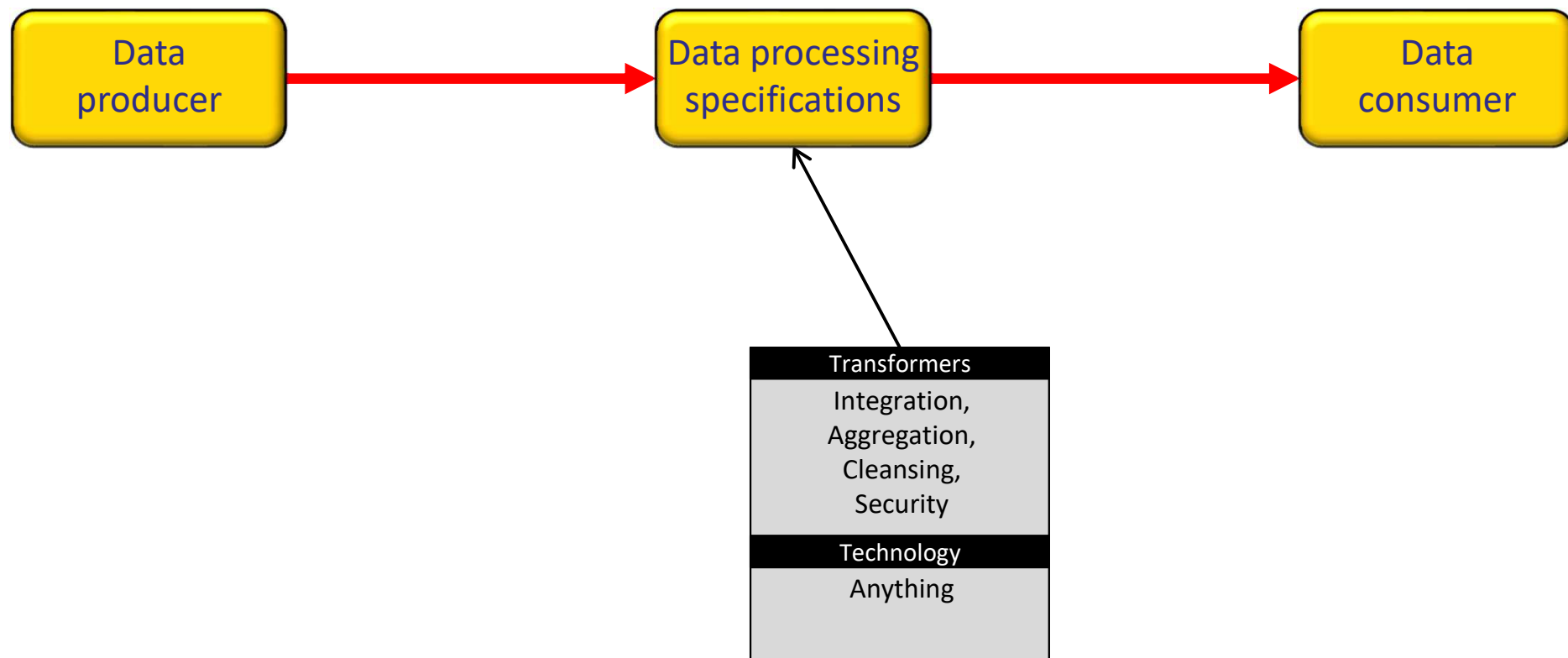
**Part 8:**  
**Steps 7-8: Design the New Data Architecture,  
Determine the Implementation Approach**

# What is a Data Track?

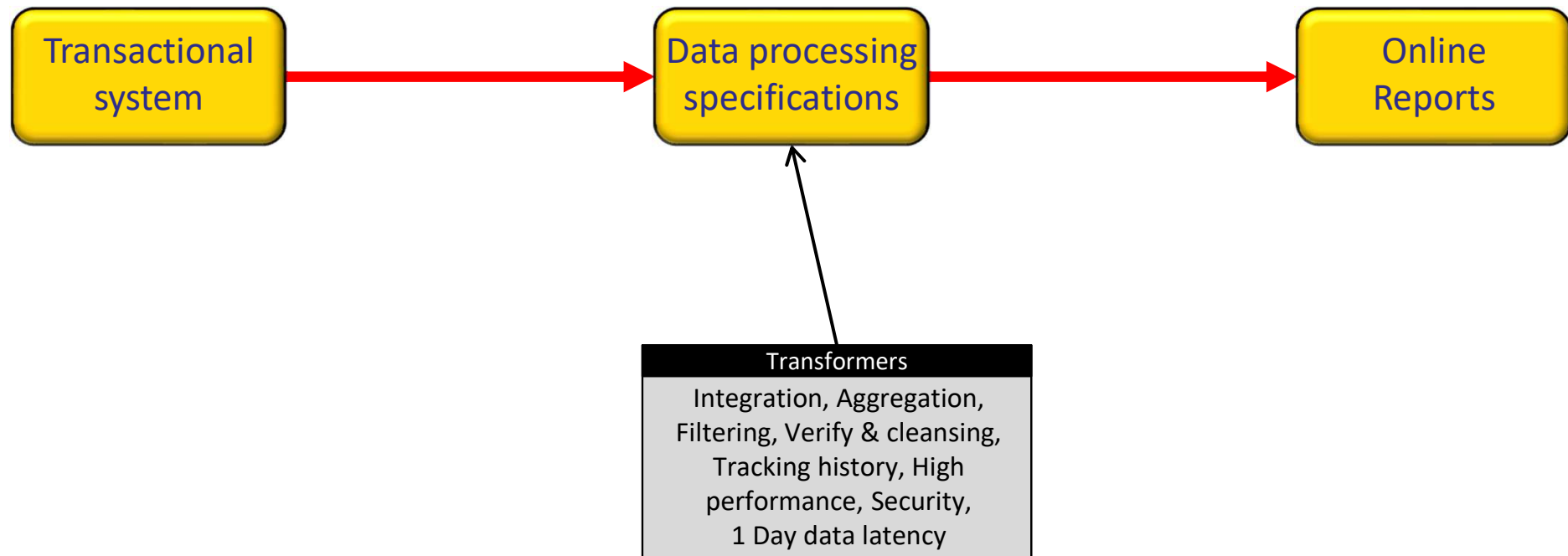
- A *data track* indicates how data “flows” from data producers to data consumers, and specifies the transformers to be applied and by which module.
- Multiple data consumers can share one data track.
- Data tracks may merge and split.



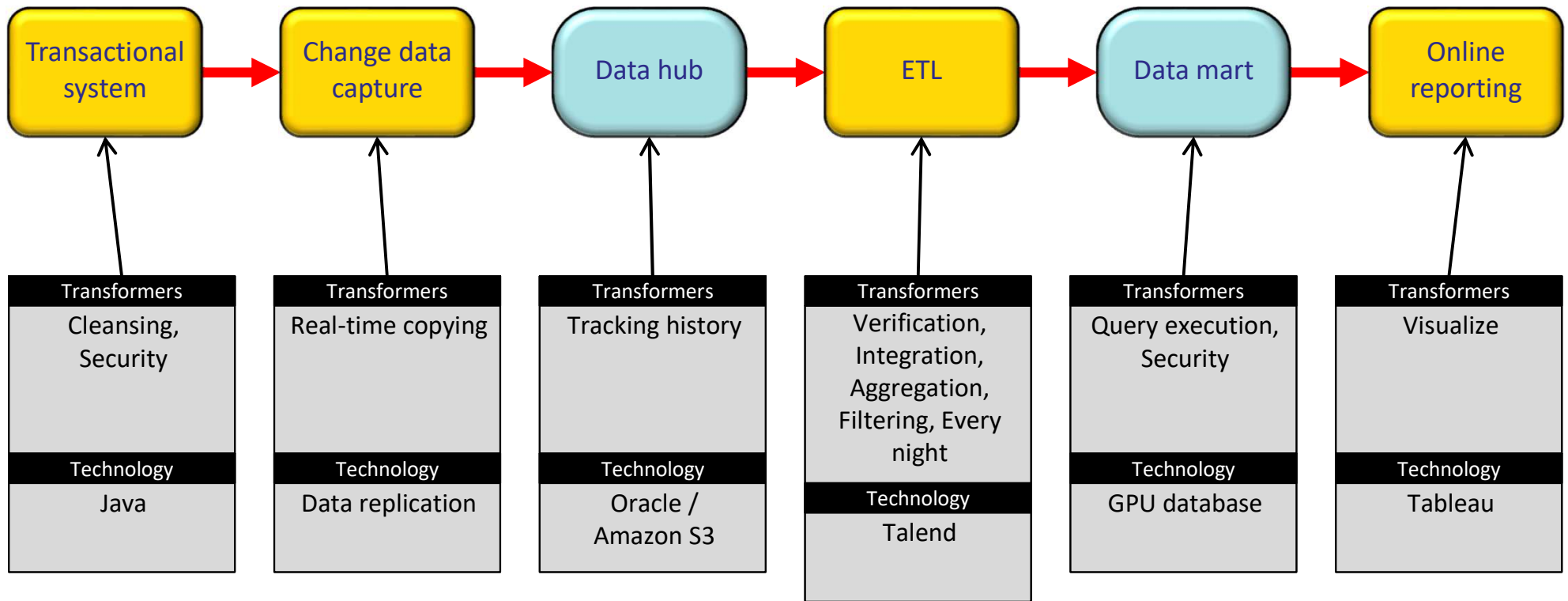
# A Data Track Diagram



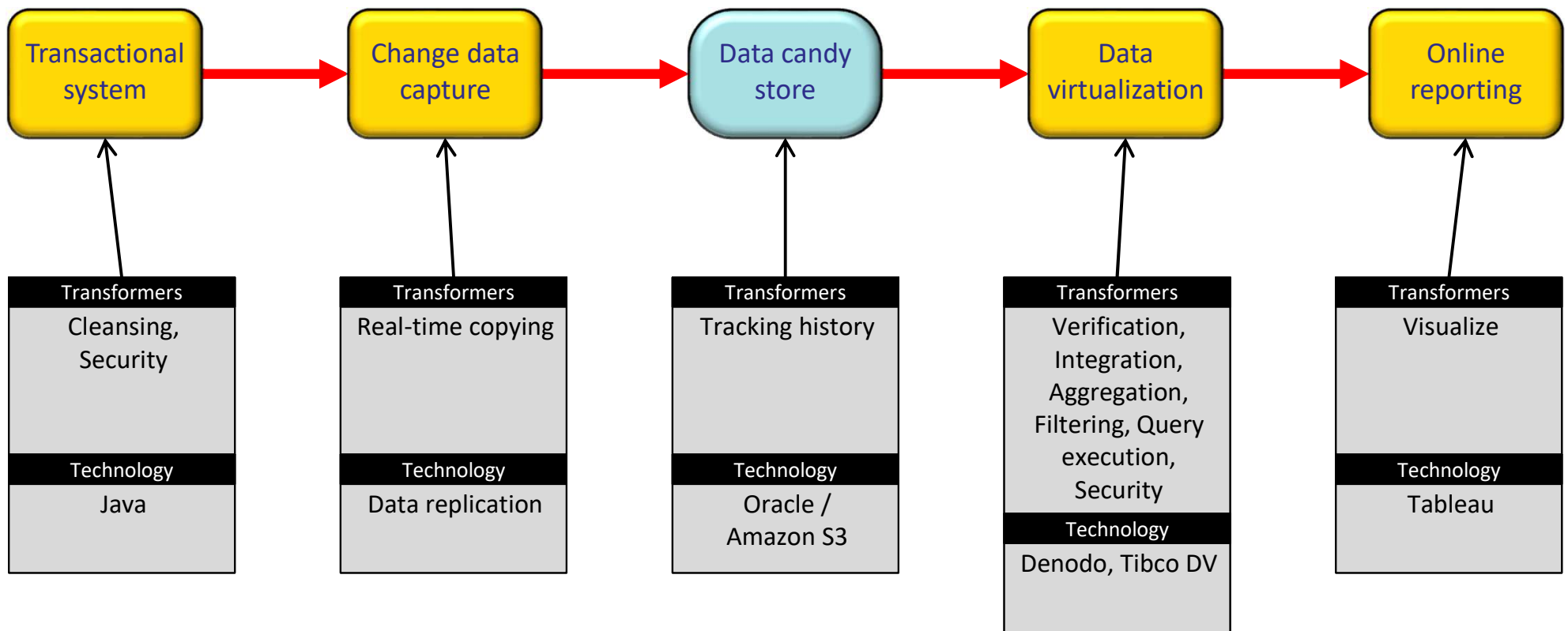
# Data Track Example: Standard Online Reporting (1)



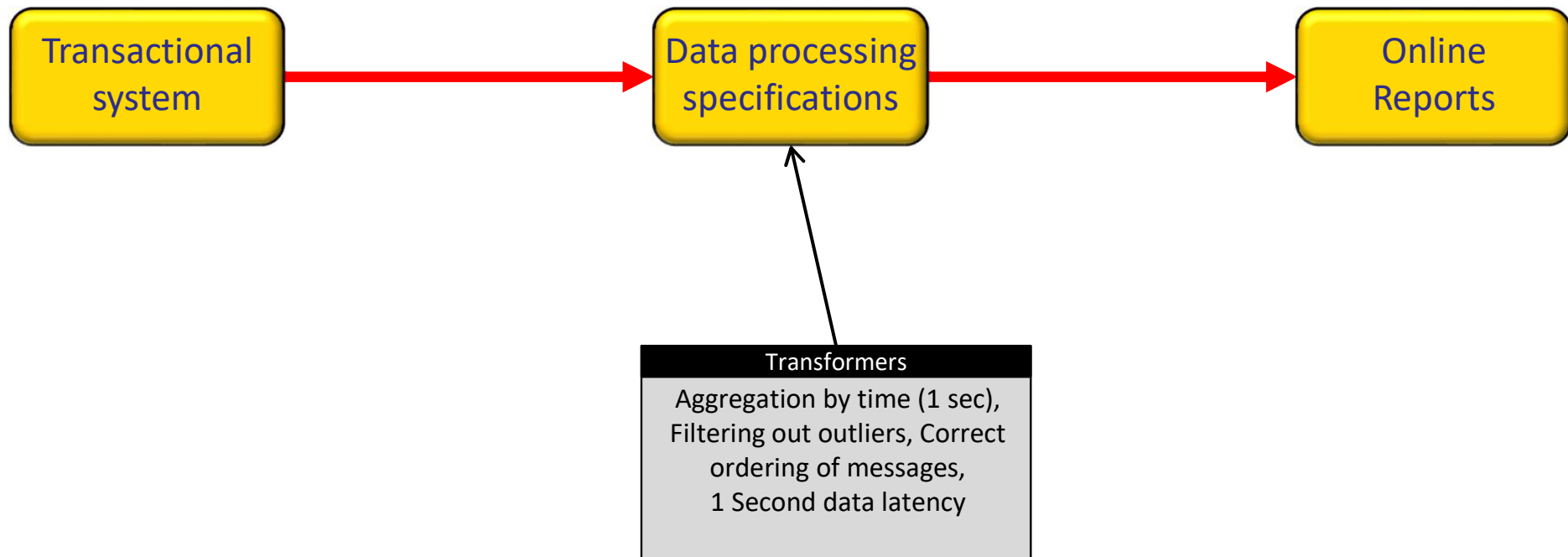
# Data Track Example: Standard Online Reporting (2)



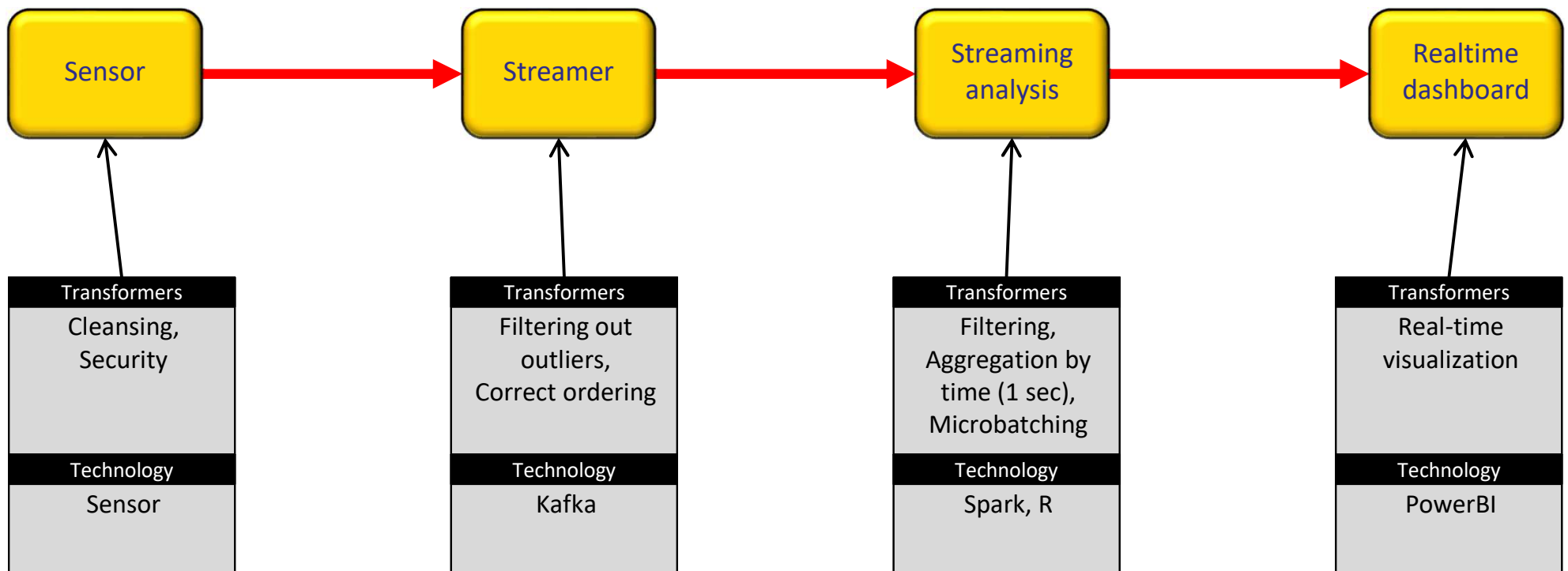
# Data Track Example: Standard Online Reporting (3)



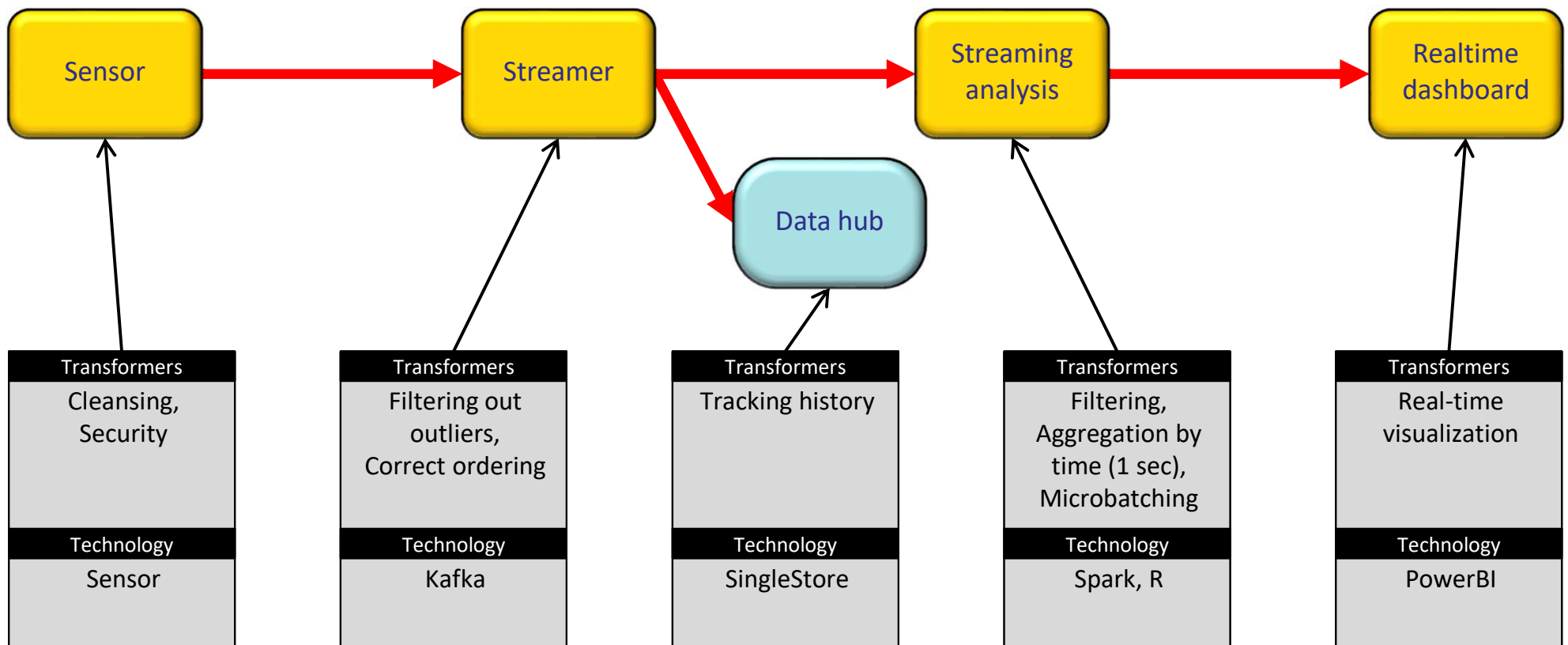
# Data Track Example: Streaming Real-time Dashboard (1)



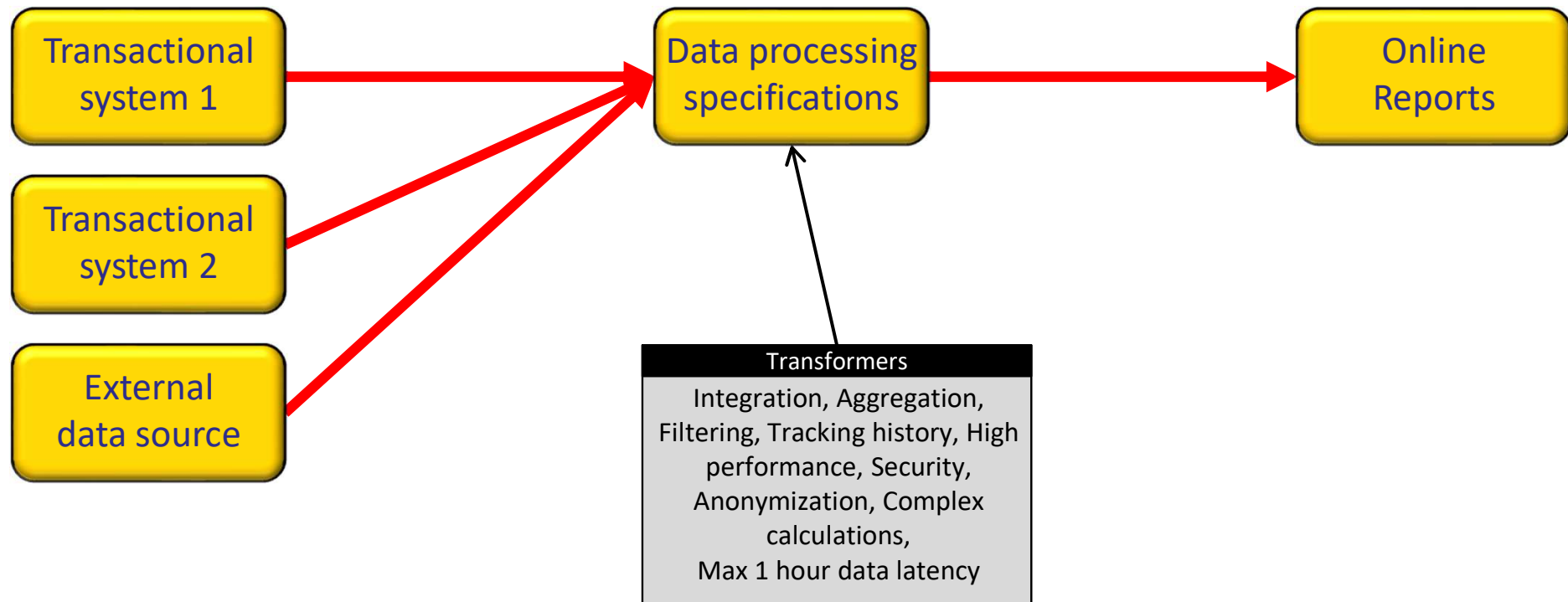
# Data Track Example: Streaming Real-time Dashboard (2)



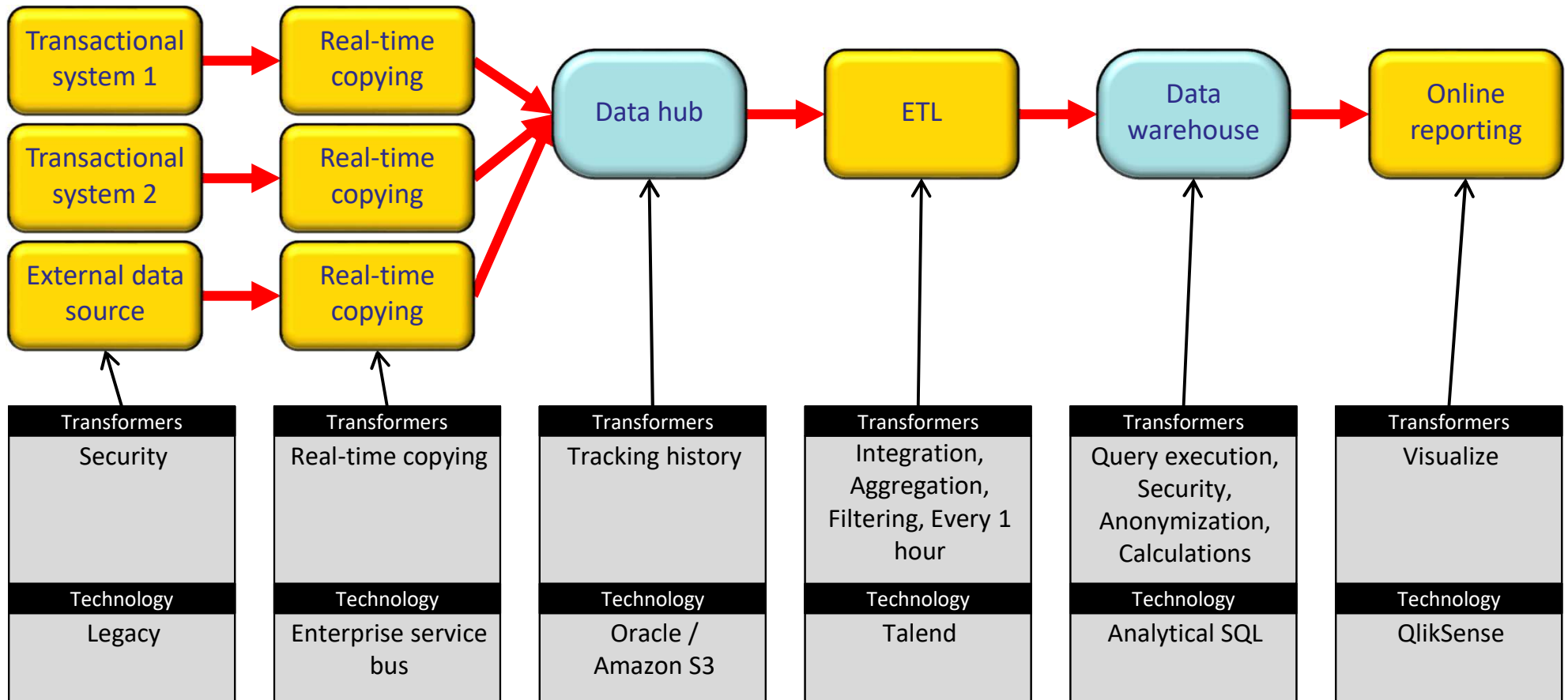
# Data Track Example: Streaming Real-time Dashboard (3)



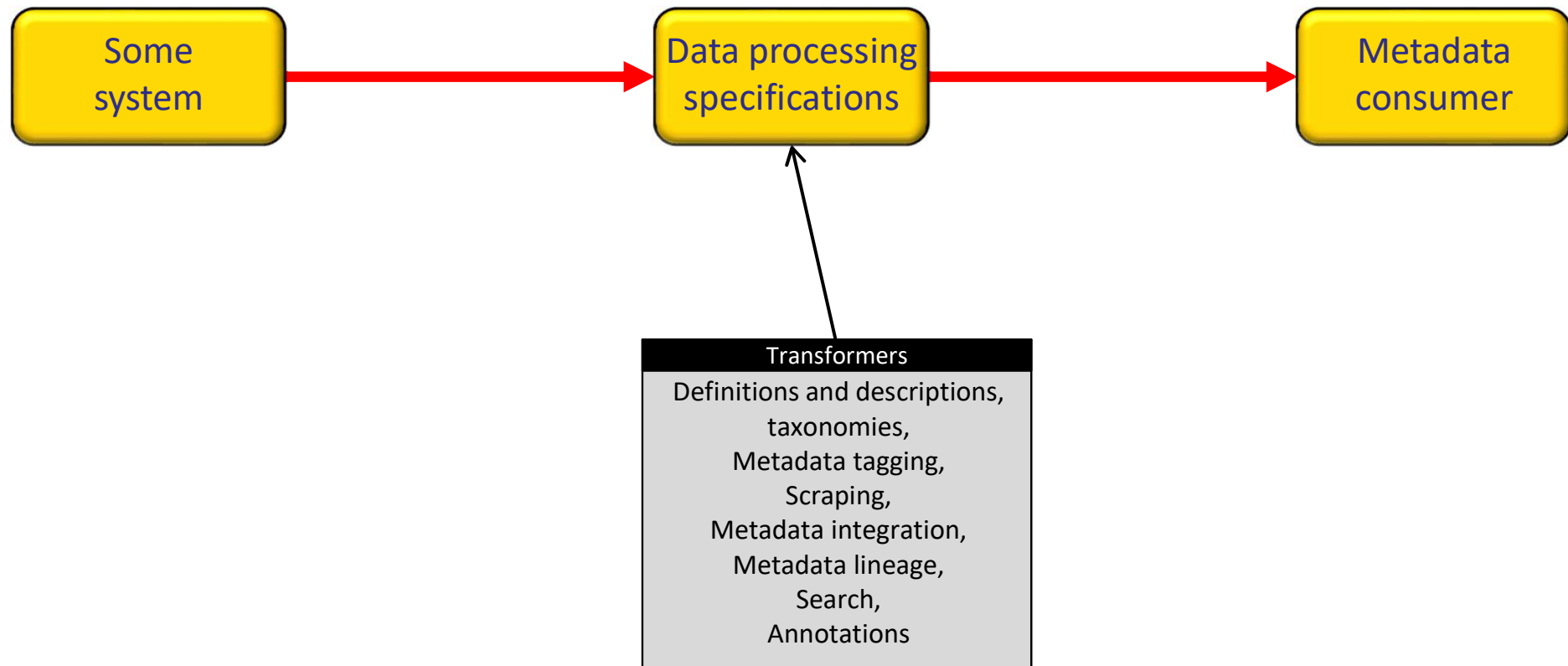
# Data Track Example: Integrated Online Reporting (1)



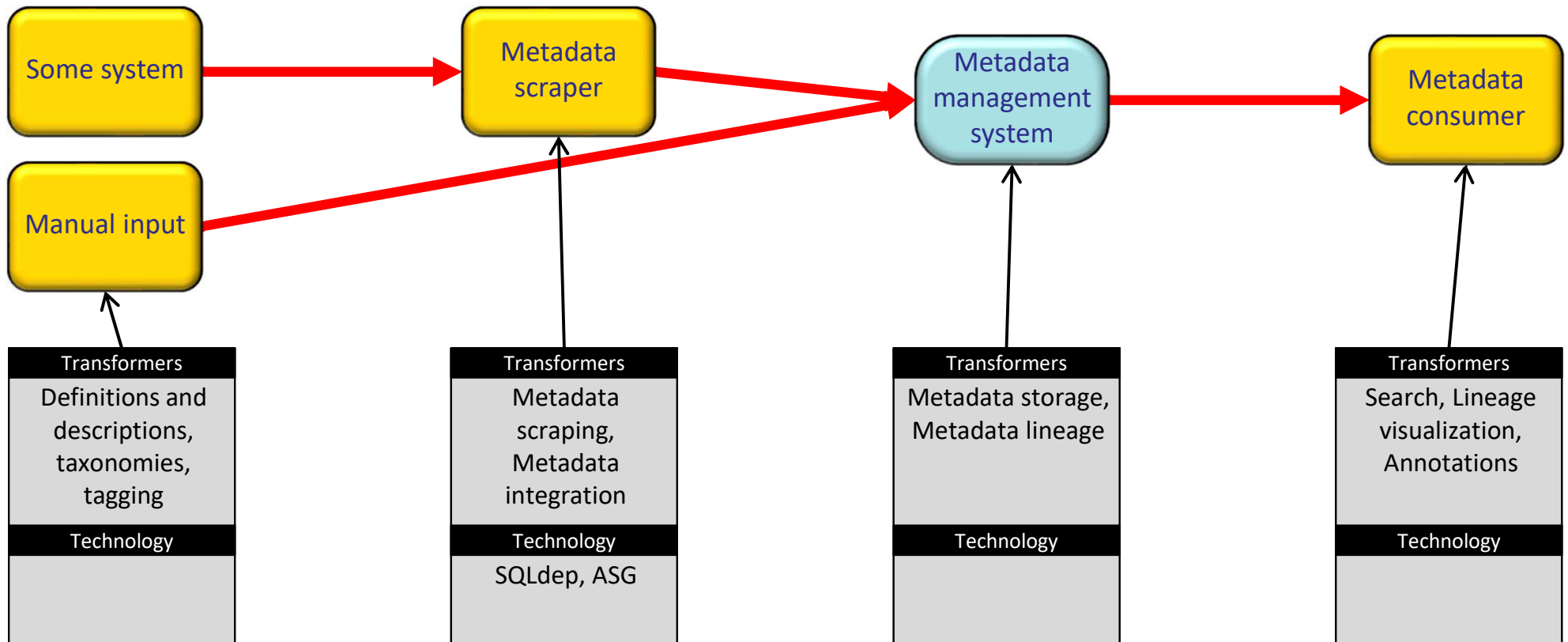
# Data Track Example: Integrated Online Reporting (2)



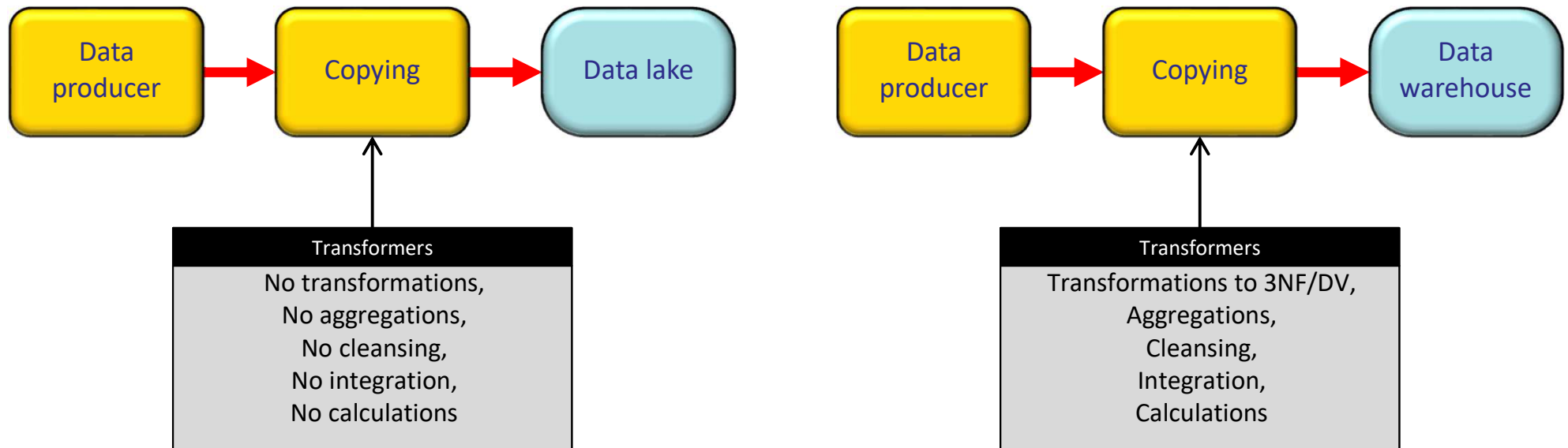
# Data Track Example: Metadata Delivery (1)



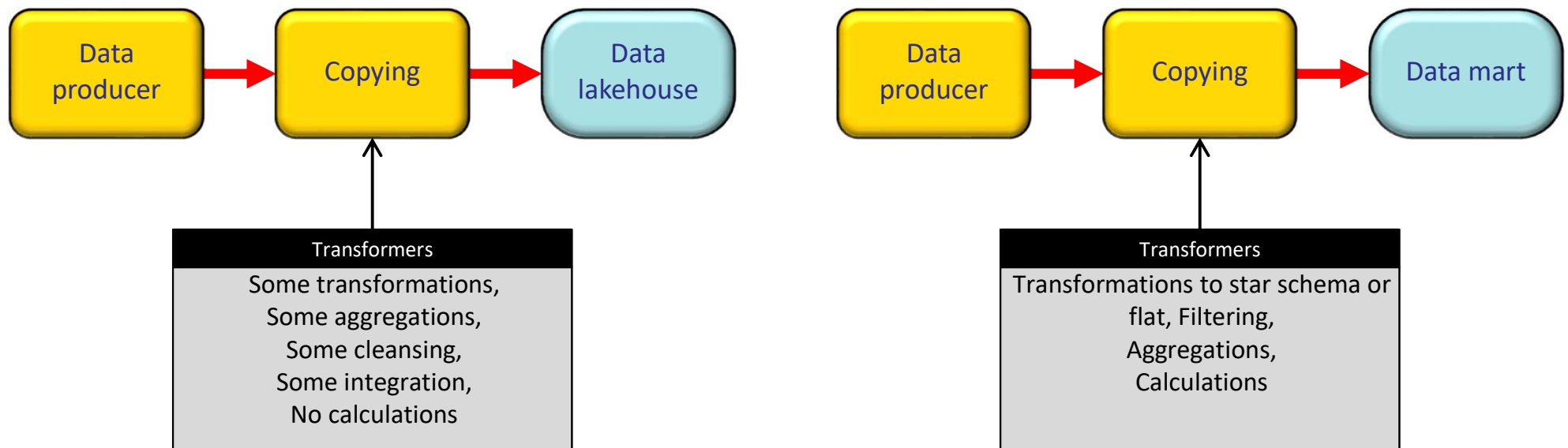
# Data Track Example: Metadata Delivery (2)



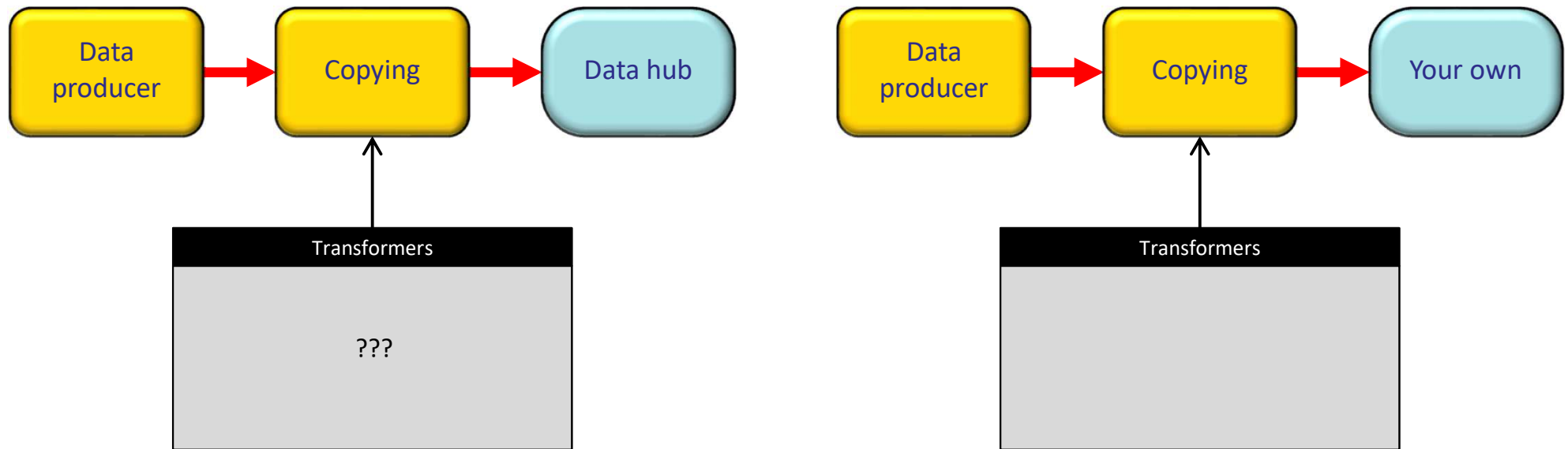
# What's in a Name? (1)



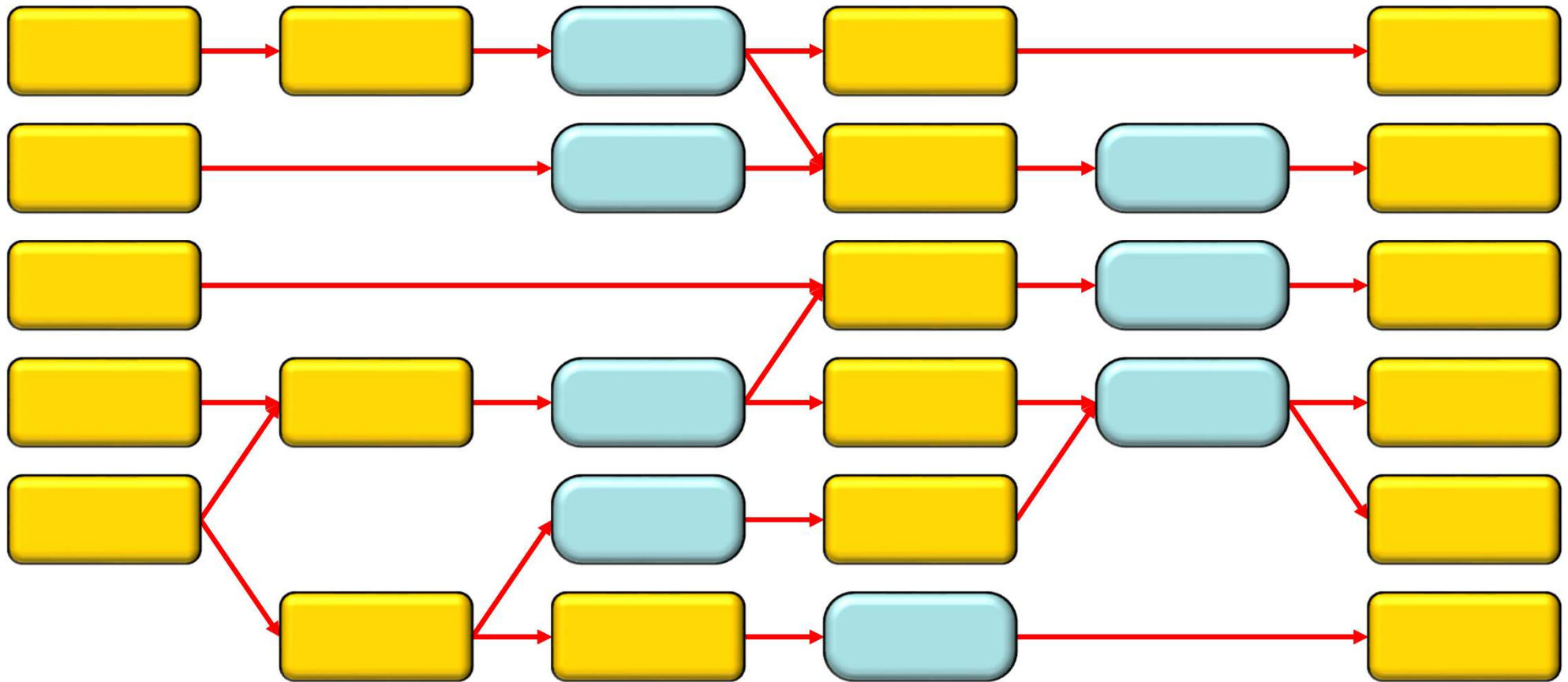
## What's in a Name? (2)



# What's in a Name? (3)

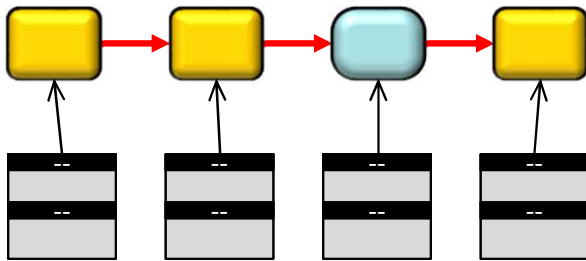


# High-Level View of the Tracks



# Recommendations for Designing Data Tracks

---



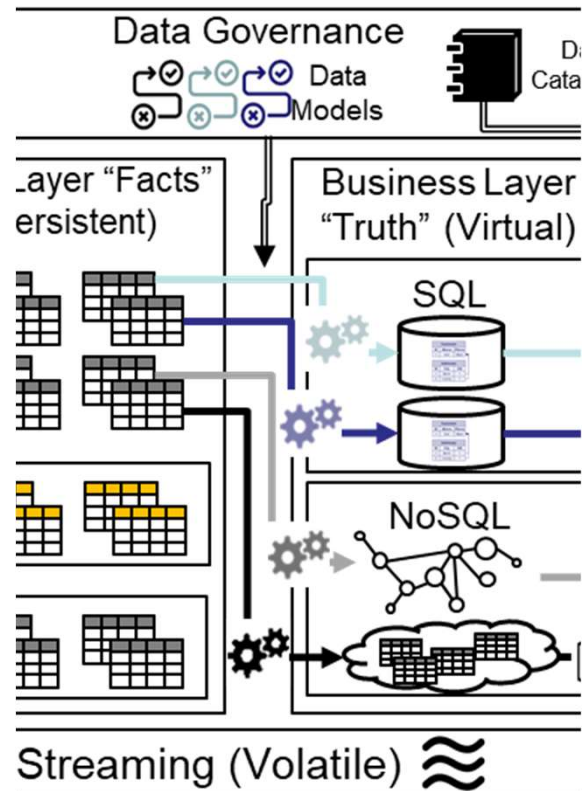
- Design them “backwards” (from consumer to producer)
- Identify transformers first, before assigning of the specs to modules
- One data track can support many comparable data consumers
- Define data track patterns!
  - Architectural design principle?
- Don't solve specific problems

# Some More Guidelines for Data Architectures

---

1. Treasure your transformers
2. Centralize implementation of transformers
3. Centralize technical and business metadata
4. Implement abstraction / decoupling
5. Make plug and play of technology possible
6. Store all data
7. Minimize stored data redundancy - compute over store
8. Choose productivity over performance
9. Minimize design exceptions
10. Implement cross checks
11. Don't send data, let them get it
12. Source systems responsible for data quality
13. Deploy a holistic design approach

# Determine the Intermediate Diagrams



- The current data architecture diagram
- The new data architecture diagram
  - The dream
  - Will never be reached
- The intermediate data architectures
  - The path from current to new
  - Make the steps as small as possible
  - Preferred: Each step leads to business value
- Think big, act small

## Part 9: Steps 8-9: Final Steps



# Roadmap for Designing Data Architectures

---

1. Determine business motivations
2. Determine new requirements
3. Analyze the existing environment
4. Define architectural design principles
5. Select a reference data architecture
6. Design the new data architecture
7. Determine the implementation approach
8. Select new products and technologies
9. Introduce the data architecture within the organization

**Part 9.1:**  
**Step 8: Select New Products and Technologies**



# Developing the Request for Information

---



- Get access to in-depth technological know-how
  - To ask the right questions
- Use distinguishing questions
- Weighs of requirements – explain why
- Closed questions – easier for comparisons
- Deliver sufficient info to let vendor provide details for pricing and products
- Are extra modules/versions required?
- Remember the new requirements!

# Product and Vendor Evaluation (1)

---



Photo: Clay Banks

- Evaluation of products
  - Features, performance, costs, market share
- Local support and partners
  - Experience?
- Extra software required
  - Master data management for complex integration
  - Data cleansing
  - Database server for reference tables and caches
  - Data security
  - Special connectors/drivers

## Product and Vendor Evaluation (2)

---



Photo: Clay Banks

- Products need to “fit” the architecture
- The intended use cases of the products must match use cases of organization
- Do you need the best tool?
  - Remember Betamax and quadrophonic records
- One-stop shopping or best-of-breed?
  - Minimize number of vendors
  - Never independent of zero vendors

# Product and Vendor Evaluation (3)

---



Photo: Clay Banks

- Standardize for back-end tools
  - If use cases allow
  - Not one BI tool for all forms of data consumption
- Open Source software
  - Open source ≠ Non-proprietary software
  - Standards = Non-proprietary software
  - Study how active the development group is
  - What if open source vendor goes commercial?
    - MySQL, Revolutionary Analytics (R), ...

# The Proof of Concept/Pilot



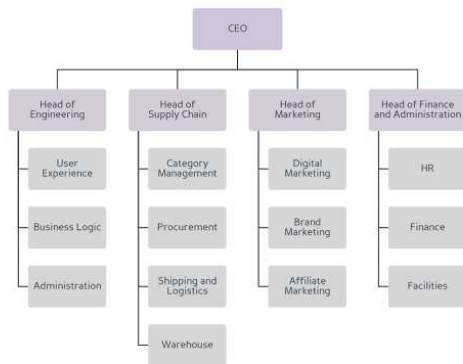
Photo: Haupes, Co

- The PoC must be representative of the new system
  - Data: size, characteristics (value distribution, uniqueness), anonymized data?
  - Applications and reports must have a representative complexity
- Performance
  - Multi-user tests
  - Experts required
- Tough SLAs must be tested!
  - Can be expensive
- Invite vendors to install and optimize software themselves
- Select ICT personnel for PoC
  - Developers who enjoy working with new technology and are willing to stay over the weekend

**Part 9.2:**  
**Step 9: Introduce the Data Architecture**  
**Within the Organization**

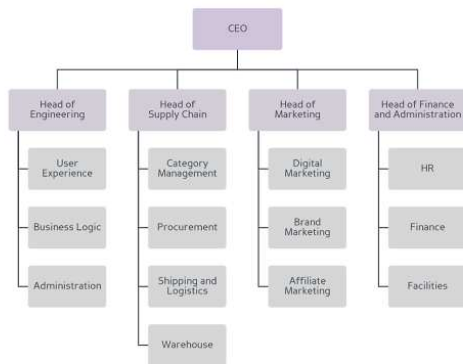


# Introduction Within the Organization (1)



- Three approaches for introduction
  - Via business
  - Via IT bottom up (Trojan horse)
  - Via IT top down
- Identify resistance
  - DBA, source owners, ...
- Educational/missionary program for everyone
  - From programmers to C-level management
  - De-mystify
  - Refute the mythical performance problem
  - Sell the new data architecture
  - Find a champion

# Introduction Within the Organization (2)



- Impact of new data architecture on organization
  - New roles and new responsibilities, examples
    - New roles related to data stewardship
    - Ownership of data
    - Data science models to support business users cooperating
    - New BI tools
  - Training

A photograph of a desert landscape featuring large, rounded, light-colored boulders. The ground is sandy and covered with sparse, low-lying green and brown vegetation. The sky is a clear, bright blue. A semi-transparent white box with rounded corners is overlaid on the upper portion of the image, containing the text "Part 10: Closing Remarks" in a bold, dark blue font.

## **Part 10: Closing Remarks**

# Time to Start!

---



- New data architectures are required
- Focus on transformers, before drawing the storage “boxes”
- Architects must be familiar with the strengths, weaknesses, and use cases of data storage and processing technologies
  - Without this knowledge:
    - Unnecessarily complex architecture
    - Incorrect use of technology
    - Not able to use the full power of a technology
- Design guidelines impact architecture
- Design a data architecture from source to insight

# From a Linear to a Holistic Approach

---

Data and Solutions  
Architectures

Design Principles

Data Storage and  
Processing Technology

Data Security  
and Privacy



The image features a background of concentric circles in shades of red and dark red, creating a tunnel-like effect. In the center, the text "That's all Folks!" is written in a white, elegant cursive font. The text is positioned over a dark blue circular area that serves as the focal point of the design.

*That's all Folks!*