

# Data Integration and ETL Process

---

ผู้ช่วยศาสตราจารย์ ดร.สมเกียรติ โกศลสมบัติ

สาขาวิชาวิทยาศาสตร์และนวัตกรรมข้อมูล

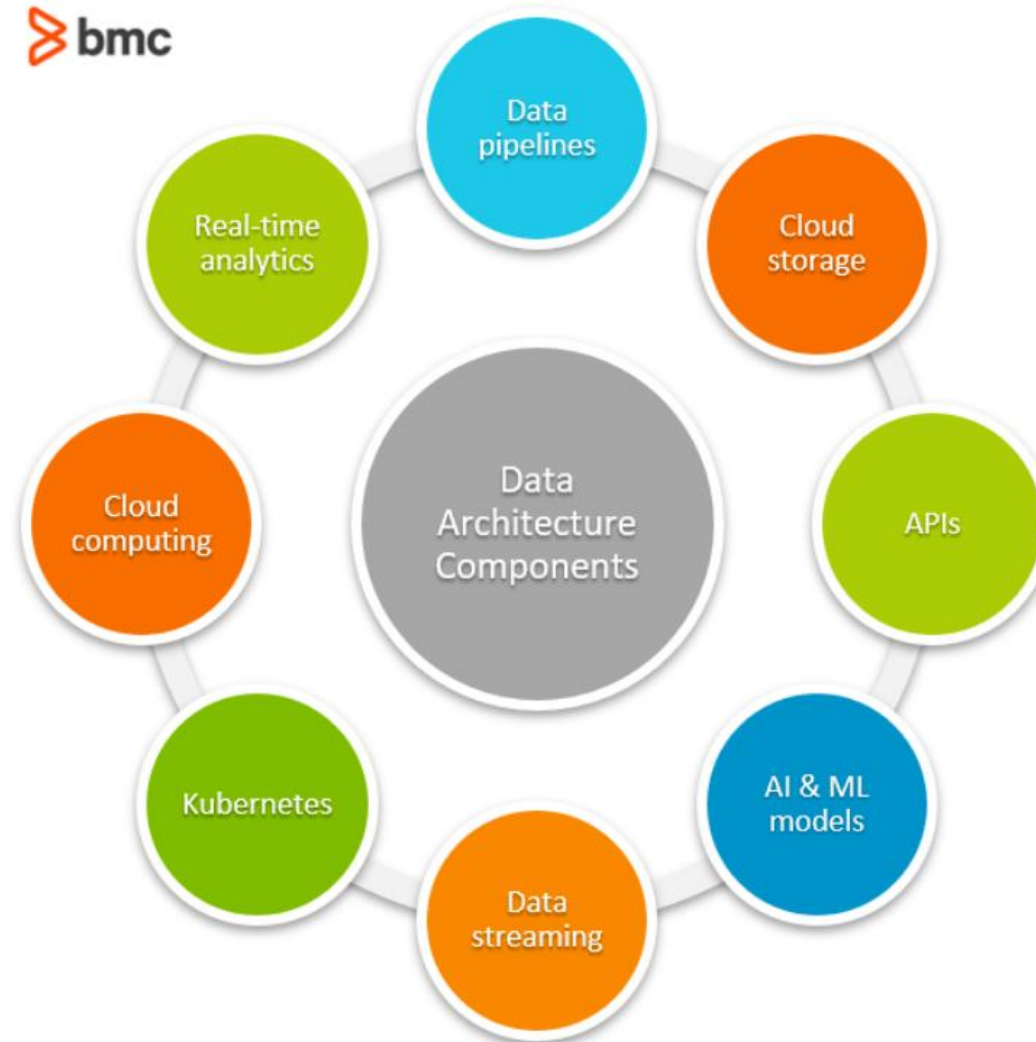
วิทยาลัยสหวิทยาการ มหาวิทยาลัยธรรมศาสตร์

# Contents

---

- Data Architecture Components
- Data Standards
- New Data Architecture
- Data Pipeline
- Case Study
- References

# Data Architecture Components



# Data Standards

---

- Data Schemas
  - The architecture is responsible for setting the data standards that define what kinds of data will pass through it.
- Data Security
  - Data standards also help set the security rules for the architecture. These can be visualized in the architecture and schema by showing what data gets passed where, and, when it travels from point A to point B

# Data Standards

---

- Data Schemas
  - Each entity that should be collected.
  - Schema for contact info, for example, might include name, phone number, email, and place of work.
  - The type of data each piece should be. For example, name is text data, phone number is integer data, email is text data, place of work is text data.
  - The relationship of that entity to others in the database, such as where it comes from and where it's going.

# Data Standards

---

- Data Security
  - Encrypting data during travel
  - Restricting access to individuals
  - Anonymizing data to decrease the value of the information upon receipt by receiving party
  - Additional actions

# New Architecture

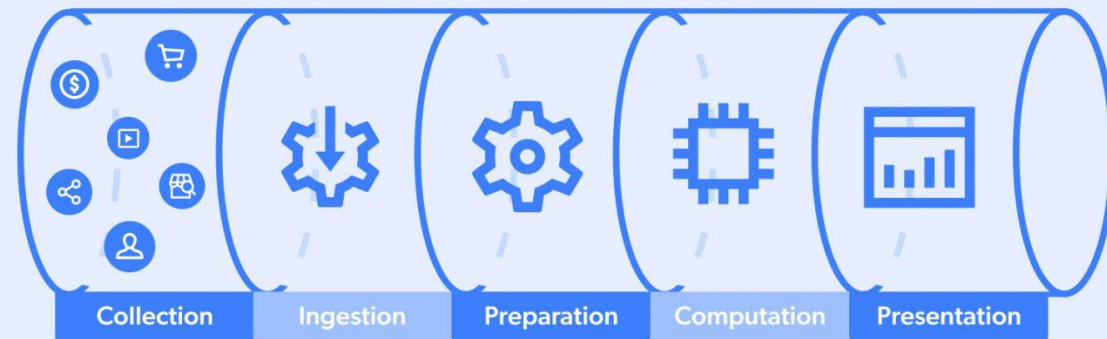
---

- From on-premise to **cloud-based** data platforms
- From batch **to real-time** data processing
- From pre-integrated commercial solutions to **modular**, best-of-breed platforms
- From point-to-point to **decoupled** data access
- From an enterprise warehouse to **domain-based** architecture
- From rigid data models toward **flexible, extensible data schemas**

# Data Pipeline

---

- A data pipeline is the process that data undergoes.
- Typically, a full cycle takes place between a ‘target site’, and a ‘data lake or pool’ that services a team in their decision-making process or an algorithm in its Artificial Intelligence (AI) capabilities.
- A typical flow looks something like this:
  - Collection
  - Ingestion
  - Preparation
  - Computation
  - Presentation





# Data Pipeline

---

- A data pipeline is a method in which raw data is ingested from various data sources and then ported to data store, like a data lake or data warehouse, for analysis.
- Before data flows into a data repository, it usually undergoes some data processing.
- This is inclusive of data transformations, such as filtering, masking, and aggregations, which ensure appropriate data integration and standardization.
- This is particularly important when the destination for the dataset is a relational database.
- This type of data repository has a defined schema which requires alignment—i.e. matching data columns and types—to update existing data with new data.

# Data Pipeline

---

- **Predictive analytics**: Algorithms are able to make predictions in terms of the stock market or product demand, for example.
- These capabilities require ‘data training’ using historical data sets that enable systems to understand human behavioral patterns in order to predict potential future outcomes.
- **Real-time market capture**: This approach understands that current consumer sentiment, for example, can shift sporadically.
- Therefore, aggregating large amounts of information from multiple sources such as collecting social media data, eCommerce marketplace data, and competitor advertisement data on search engines. By cross-referencing these unique data points at scale they are able to make better decisions resulting in higher market share capture.

# Data Pipeline

---

- **Scalability** – Data volumes tend to fluctuate often, and systems need to be equipped with the ability to activate/deactivate resources on command.
- **Fluidity** – When collecting data at scale from multiple sources, big data processing operations need the wherewithal to deal with data in many different formats (e.g., JSON, CSV, HTML) as well as the know-how to clean, match, synthesize, process, and structure unstructured target website data.
- **Concurrent request management** – As Bright Data's CEO, or Lenchner, likes to put it: 'Data collection at scale is like waiting for beer online at a music festival. Concurrent requests are short, quick lines that get service quickly/simultaneously. Whereas the other line is slow/consecutive. When your business operations depend on it, which line would you prefer to be standing on?'

# Types of Data Pipeline

---

- Batch Processing
  - For example, one command may kick off data ingestion, the next command may trigger filtering of specific columns, and the subsequent command may handle aggregation. This series of commands will continue until the data is completely transformed and written into data repository.

# Types of Data Pipeline

---

- Streaming Data

- For example, apps or point of sale systems need real-time data to update inventory and sales history of their products; that way, sellers can inform consumers if a product is in stock or not.
- A single action, like a product sale, is considered an “event”, and related events, such as adding an item to checkout, are typically grouped together as a “topic” or “stream.” These events are then transported via messaging systems or message brokers, such as the open-source offering, Apache Kafka.
- Since data events are processed shortly after occurring, streaming processing systems have lower latency than batch systems, but aren’t considered as reliable as batch processing systems as messages can be unintentionally dropped or spend a long time in queue. Message brokers help to address this concern through acknowledgements, where a consumer confirms processing of the message to the broker to remove it from the queue.

# Data Pipeline Examples

---

- A batch-based data pipeline: This is a more simple/straightforward architecture. It typically consists of one system/source that generates a large quantity of data points, which are then delivered to one destination (i.e., a data storage/analysis ‘facility’). A good example of this would be a financial institution that collects large amounts of data regarding investor buys/sells/volume on the Nasdaq. That information gets sent for analysis and is then used to inform portfolio management.

# Data Pipeline Examples

---

- A **streaming data pipeline**: This data pipeline is for more real-time applications. For example, an Online Travel Agency (OTA) that collects data on competitor pricing, bundles, and advertising campaigns. This information is processed/formatted, and then delivered to relevant teams/systems for further analysis, and decision-making (e.g., an algorithm in charge of repricing tickets based on competitor price drops).

# Data Pipeline Examples

---

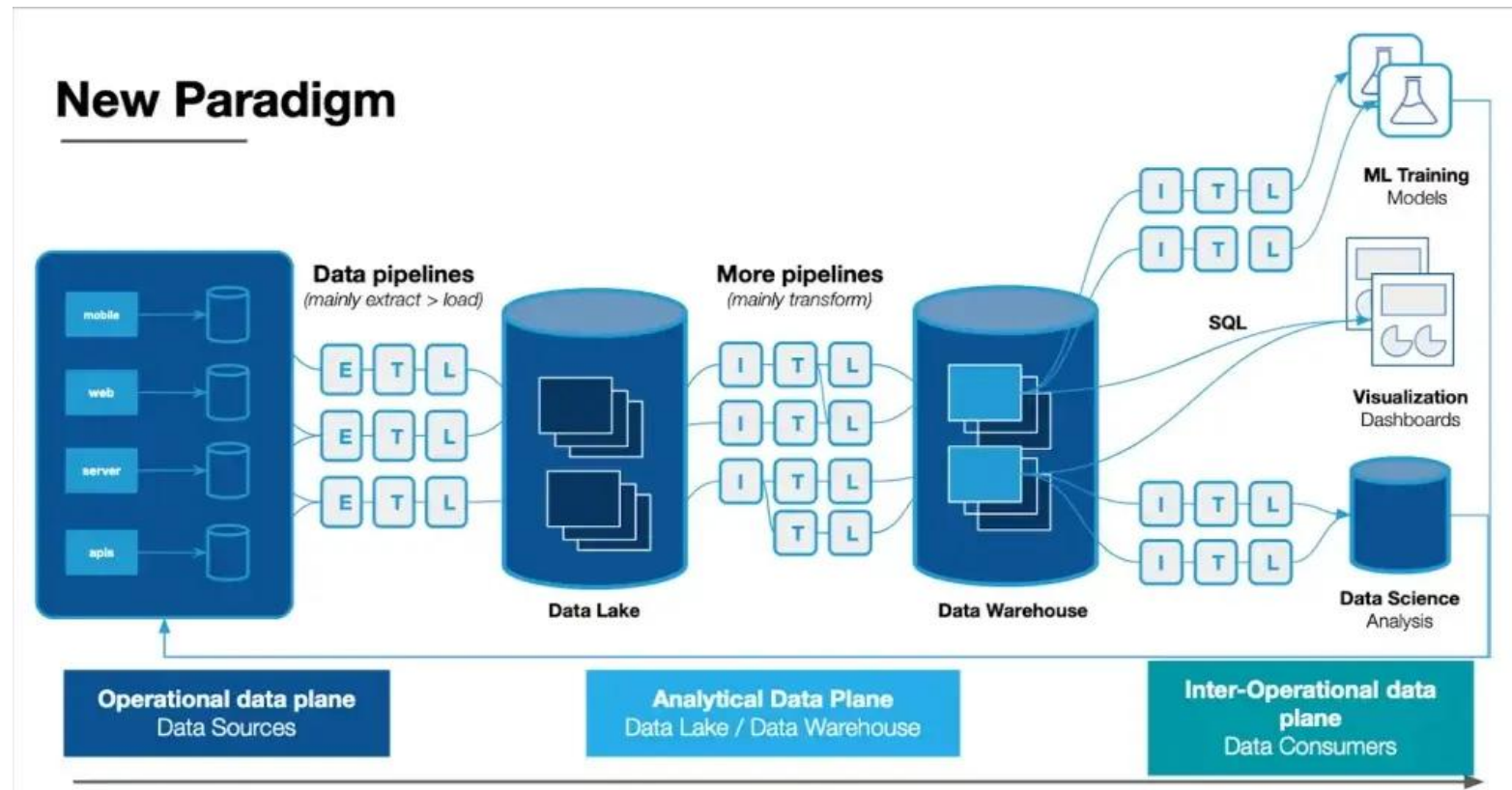
- A hybrid data pipeline: This type of approach is popular with very large companies/environments enabling real-time insights as well as batch processing/analysis. Many corporations that opt for this approach prefer keeping data in raw formats in order to enable increased future versatility in terms of new queries/pipeline structural changes.



# Data Pipeline Architecture

---

- Data pipeline architecture is the process of designing how data is surfaced from its source system to the consumption layer.

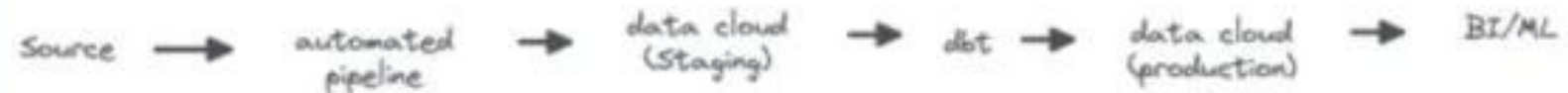


Raw ——— Clean/normalized ——— Transformed ——— Consumed

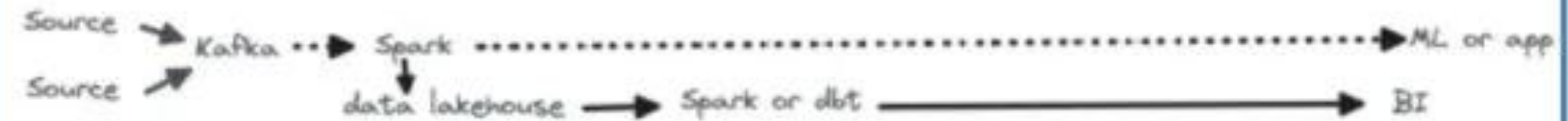
ETL  
(Hadoop Era)



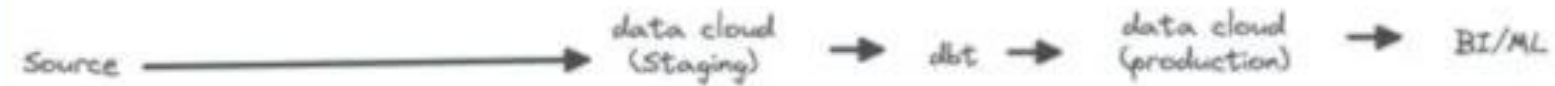
ELT  
(MDS)



Streaming

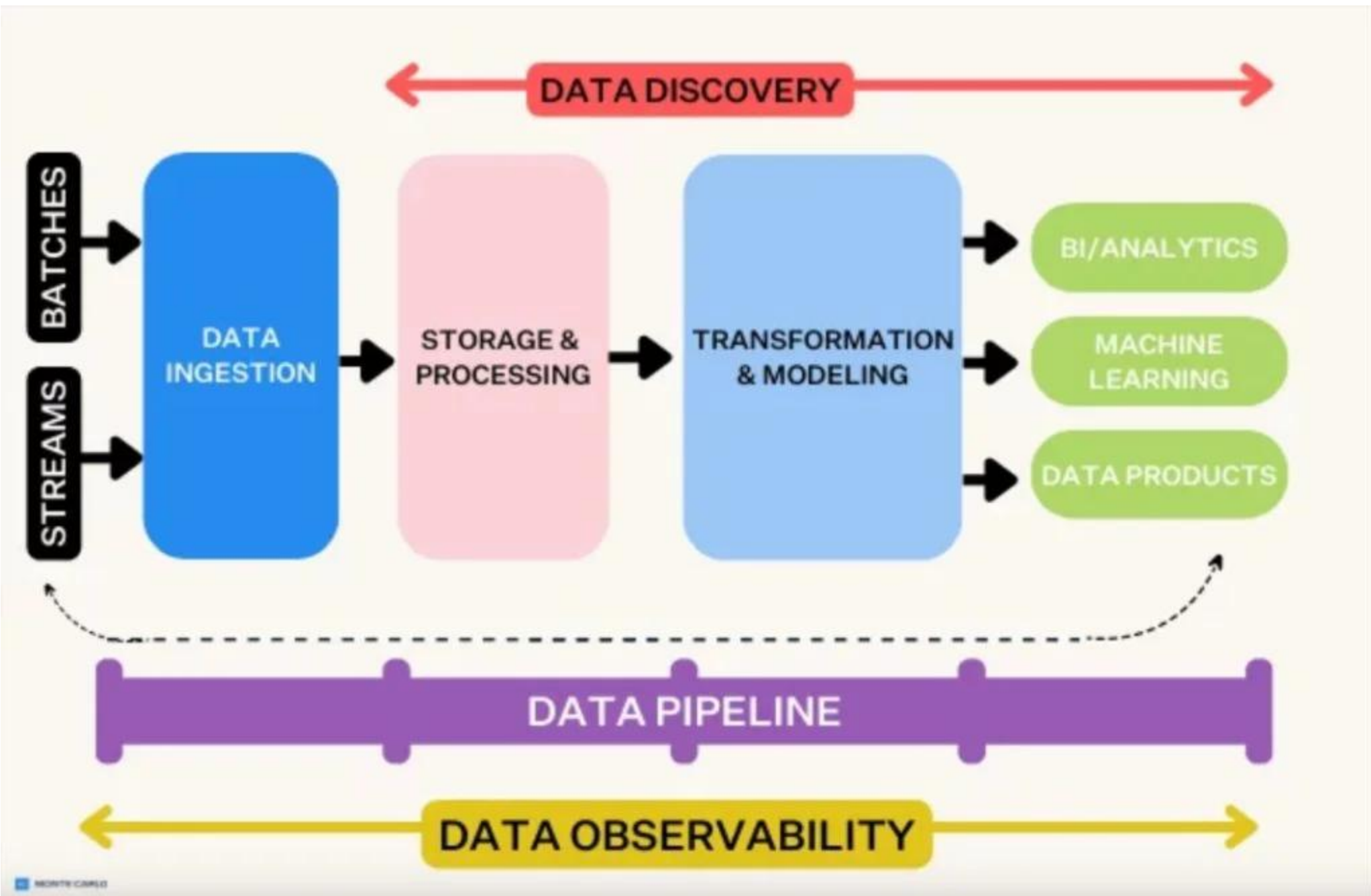


Zero ETL  
(emerging)



Data Sharing  
(emerging)
















































































































































# State of Data Engineering 2023

[Explore the table →](#)

Presented by  lakeFS

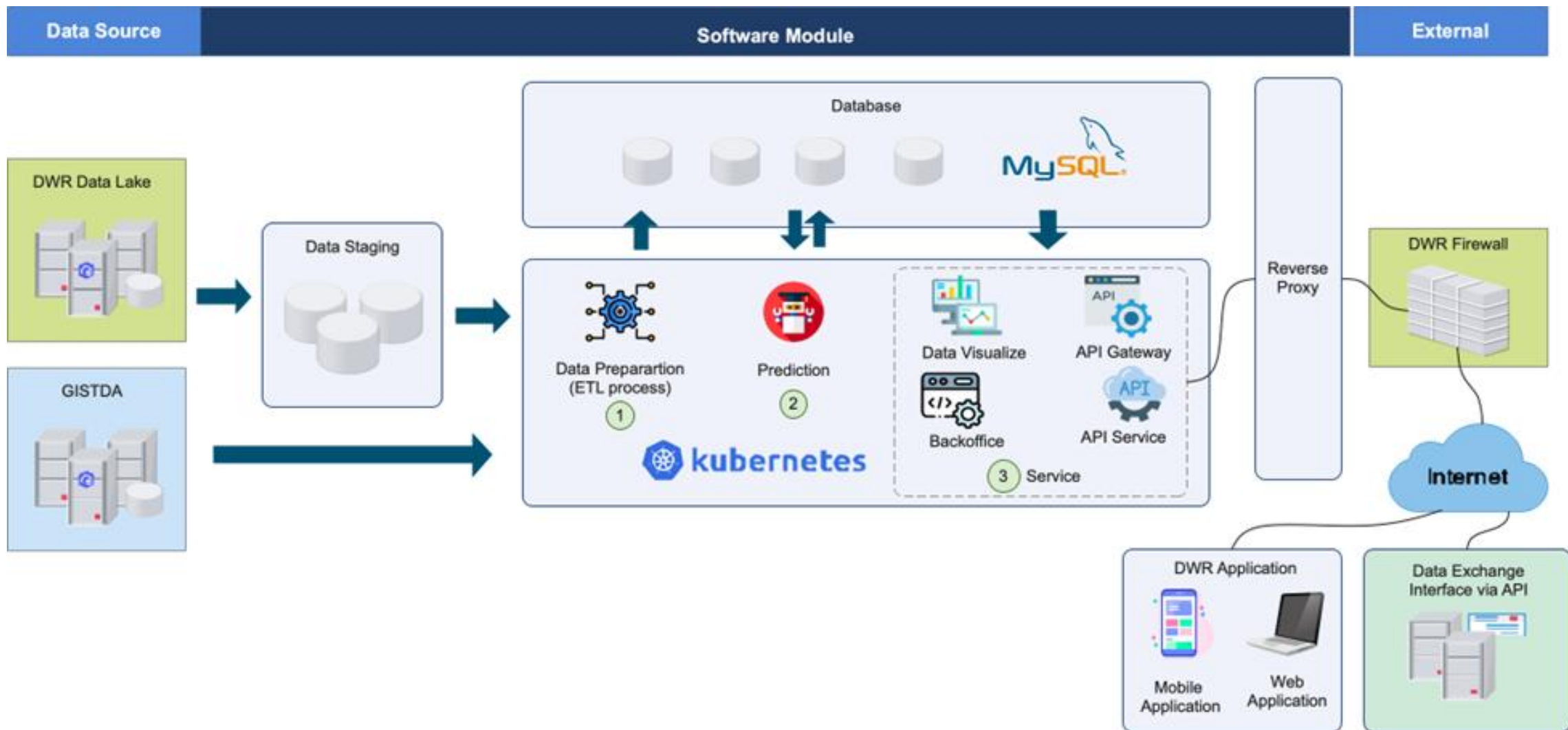
INGEST	DATA LAKE	METADATA	COMPUTE ENGINES	PIPELINES	PRACTITIONERS APPS		GOVERNANCE
<b>Ingest Tech</b>  Google Cloud Pub/Sub  Apache Storm  Kafka  Beam  Spark  Pulsar  Beneath  StreamNative  Upsolver  Confluent  Flink  Decodable <b>Ingest SaaS</b>  Airbyte  Fivetran  Segment  Rivery  Rudderstack  Datacoral  Matillion  Snowplow	<b>Object Storage</b>  Microsoft Azure Blob Storage  Google Cloud Storage  Amazon S3  Oracle Cloud  IBM Cloud Object Storage  Alibaba Cloud  Hadoop  Minio  Zadara  Cloudfiles  DigitalOcean  SwiftStack  Wasabi  Pure Storage  CrowdStorage  Vast  Filebase	<b>Metastore</b>  Databricks  Unity Catalog  Amazon Athena  Tabular  Google Cloud Data Catalog  Cloudera <b>Data Version Control Infra</b>  lakeFS  Nessie <b>Open Table Formats</b>  Onehouse  Hudi  Iceberg  Orc  Delta Lake	<b>Distributed Compute</b>  Databricks  Spark  Cloudwick  Amazon EMR  Azure Databricks  Cloudera  Bodai  Ray <b>Analytics Engines</b>  Data Fusion  Google Big Query  Amazon Redshift  Snowflake  Pinot  Databricks  StarRocks  Firebolt  Hazelcast  Imply  ClickHouse  Amazon Athena  Dremio  Rockset  StarTree  Bale  Druid  Starburst  Yellowbrick  Hydra  DuckDB  Boris	<b>Orchestration</b>  Airflow  Prefect  Luigi  Flyte  Dagster  Astronomer  Shipyard  Kestra  Kensu  Pandera  Great Expectations  Metaplane  LightUp  Soda  Datafold  Why Labs  Griffin  Monte Carlo  Databand  Unravel  Redata  Anomalo  SoloClean  AWS Labs / Deequ  Elementary  Timeseries AI	<b>MLOps End-to-End</b>  OctoML  ABACUS.AI  Metaflow  Baseten  Verta  Aible  DataRobot  Wallaroo  Cnvrg.to  Hugging Face  Mlflow  Seldon  Kedro  W&B  Graft  Kubeflow  ZenML  DotData  Iterative  Neptune.ai  Comet  BentoML  Deca  Modular  DAGsHub  Dataiku  Qwak  Valohai <b>Data Centric AI/ML</b>  DVC  ActiveLoop  Pachyderm  Graviti  Alectio  Stack AI  Galileo  Voxel51  Xethub  Dataq  Snorkel  Dstack  DataLoop  Labelbox  Scale <b>ML observability and monitoring</b>  Deepchecks  Mona  Superwise  Fiddler  DataBuck  Arize  Apres  Census  Evidently AI  Arthur  Why Labs  Giskard  Badook  Hydrosphere.io  Robust Intelligence  Truera  Apona  Gantry <b>Feature Stores</b>  Hopsworks  Redis  Scribble Data  Feast  Tecton <b>Notebooks</b>  Hex  Count  Noteable  Deepnote <b>Analytics Workflow</b>  Dataform  dbt  Querybook  Databricks		<b>Data Catalog/ Governance</b>  Ckan  Amundsen  BigD  DataHub  Boomi  Raito  Marquez  Atlan  Apache Atlas  Magda  Mimuta  Okera  IBM Watson  Data.world  Alation  SELECT STAR  Zeenea  Platform  Stemma  Collibra  Acryl Data

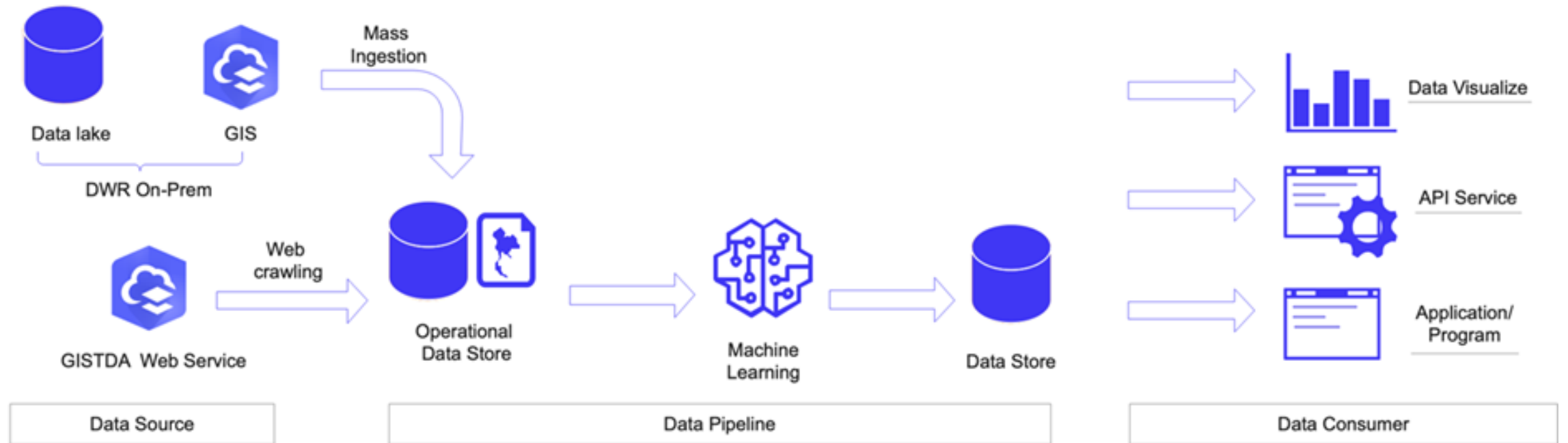


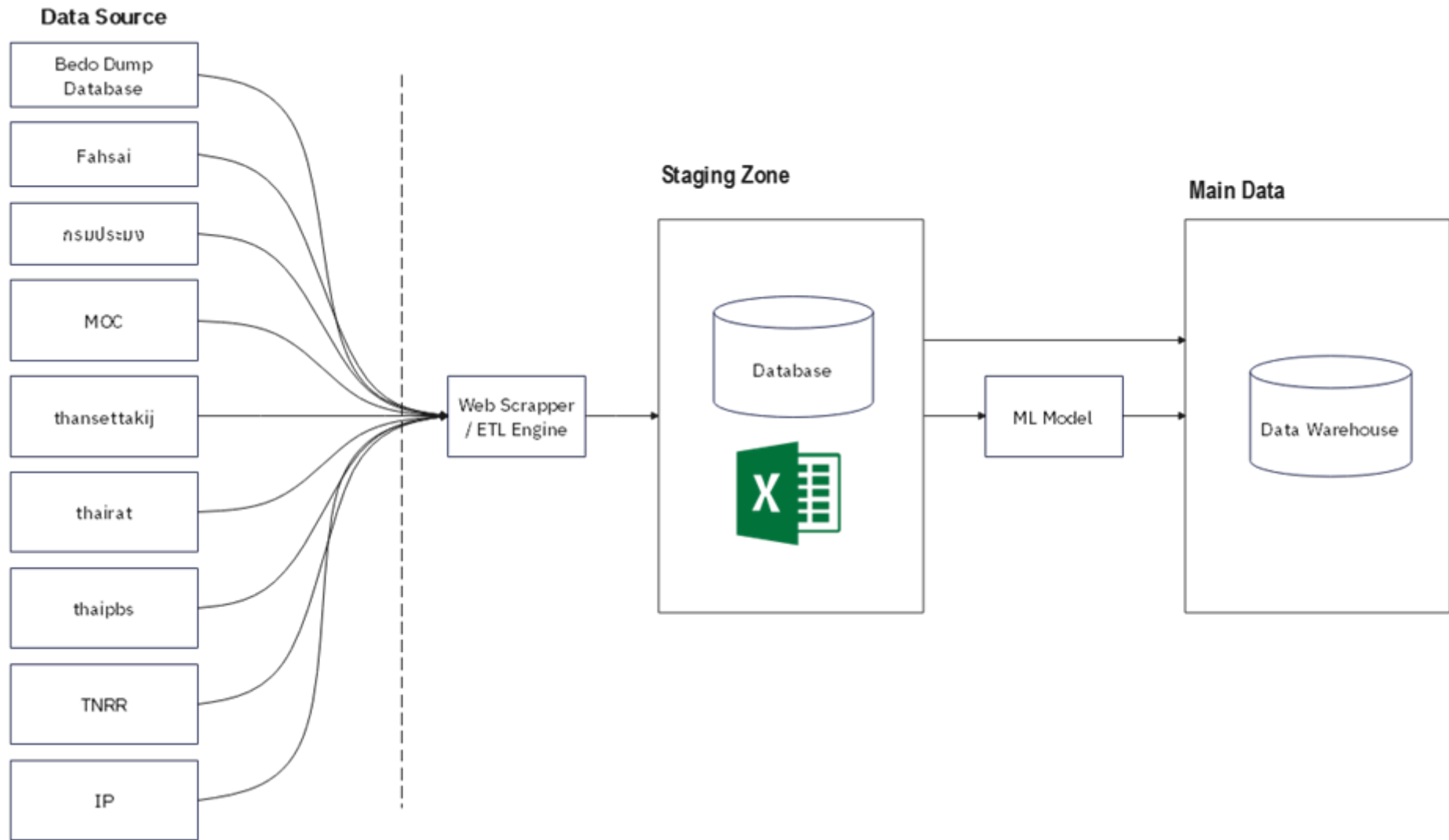
# Data Pipeline Architecture Best Practices

---

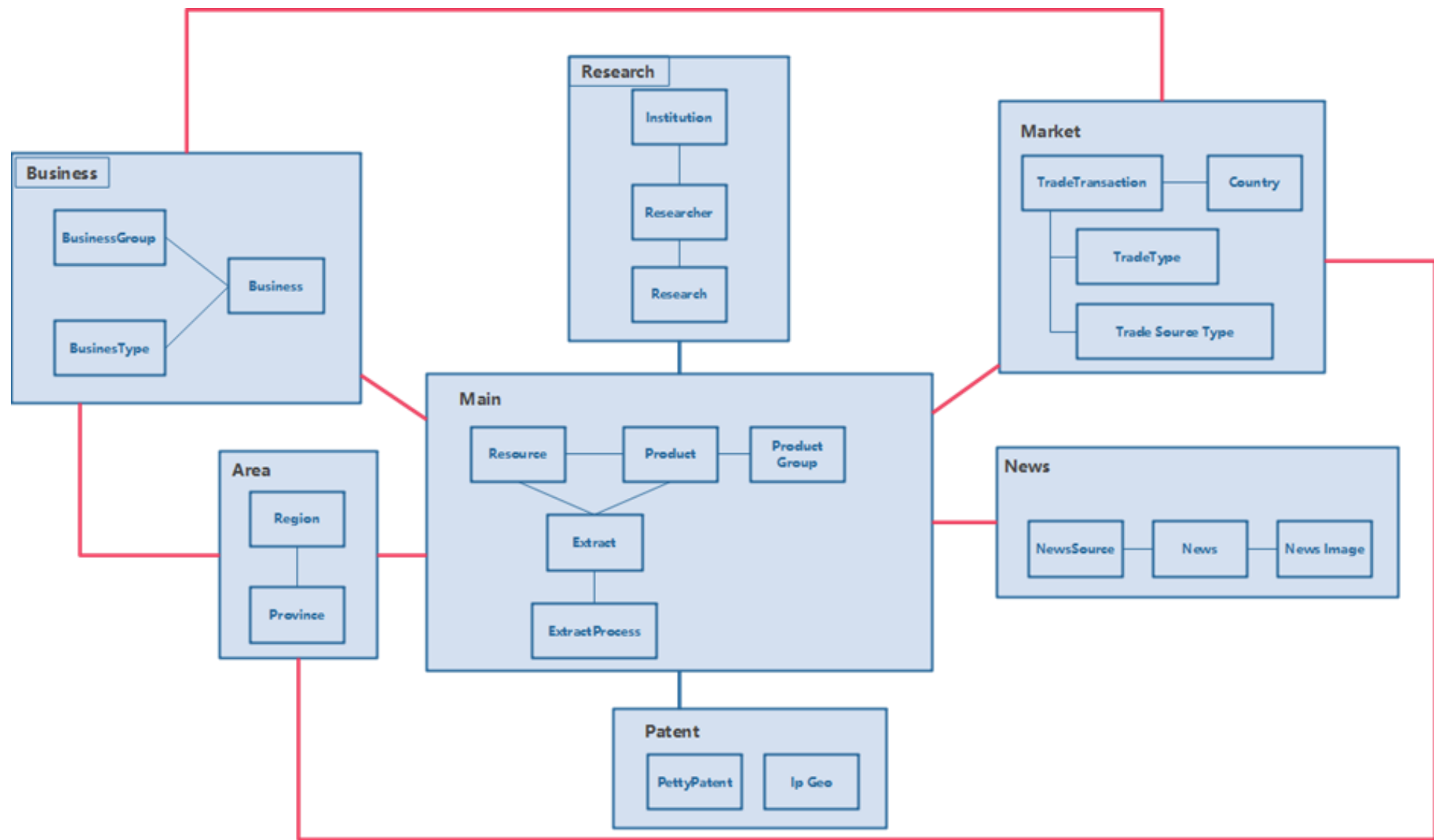
- Map and understand the dependencies
- Design your data pipeline so it is modular and automated
- Create data pipeline SLAs (Service Level Agreements)
- Let the data drive the data pipeline architecture
- Create data products
- Continuously review and optimize costs
- Make pipelines idempotent

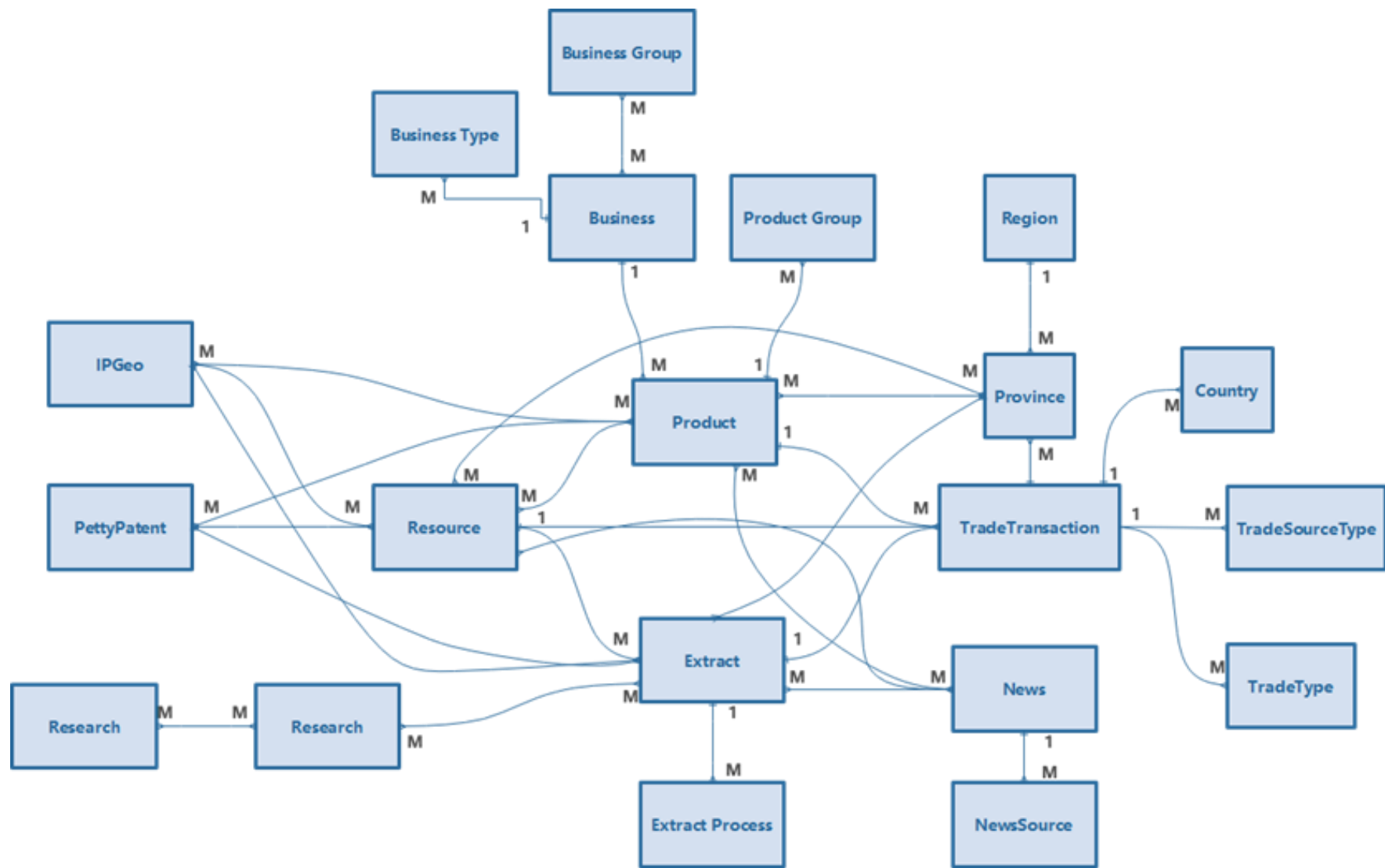












# References

---

- <https://www.bmc.com/blogs/data-architecture/>
- <https://brightdata.com/blog/proxy-101/data-pipeline-architecture>
- <https://www.ibm.com/topics/data-pipeline>
- <https://www.montecarlodata.com/blog-data-pipeline-architecture-explained/>
- <https://www.ibm.com/topics/data-pipeline>