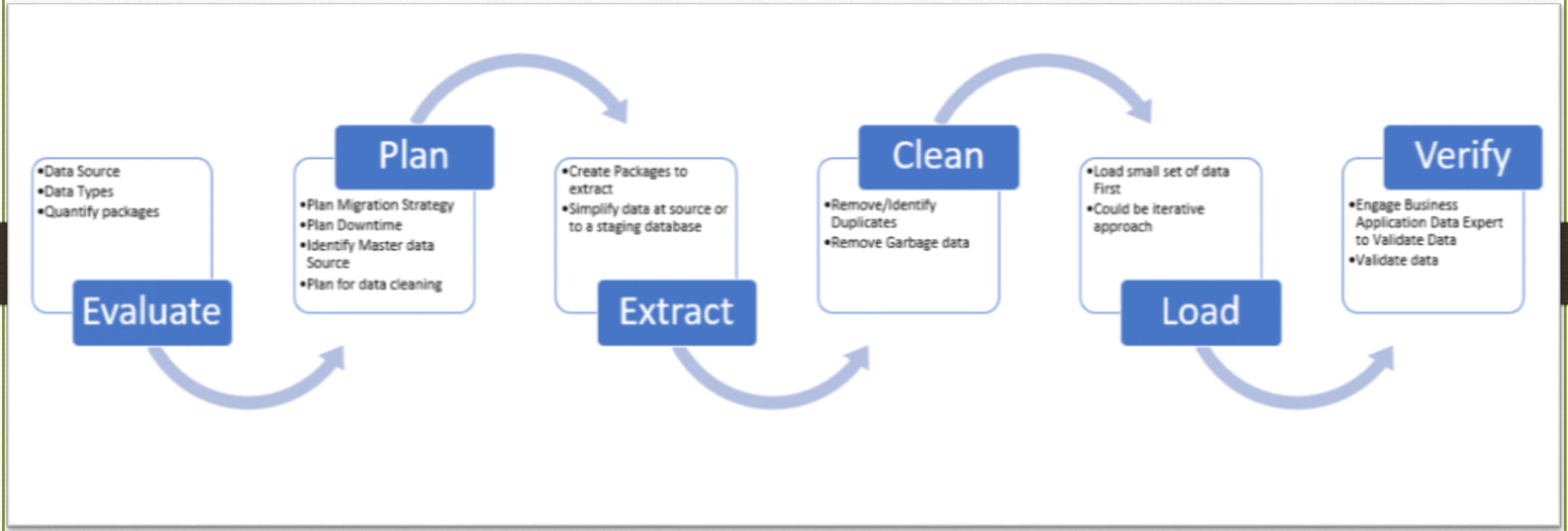
YouTube: Techlake

Data Warehousing Concepts

1

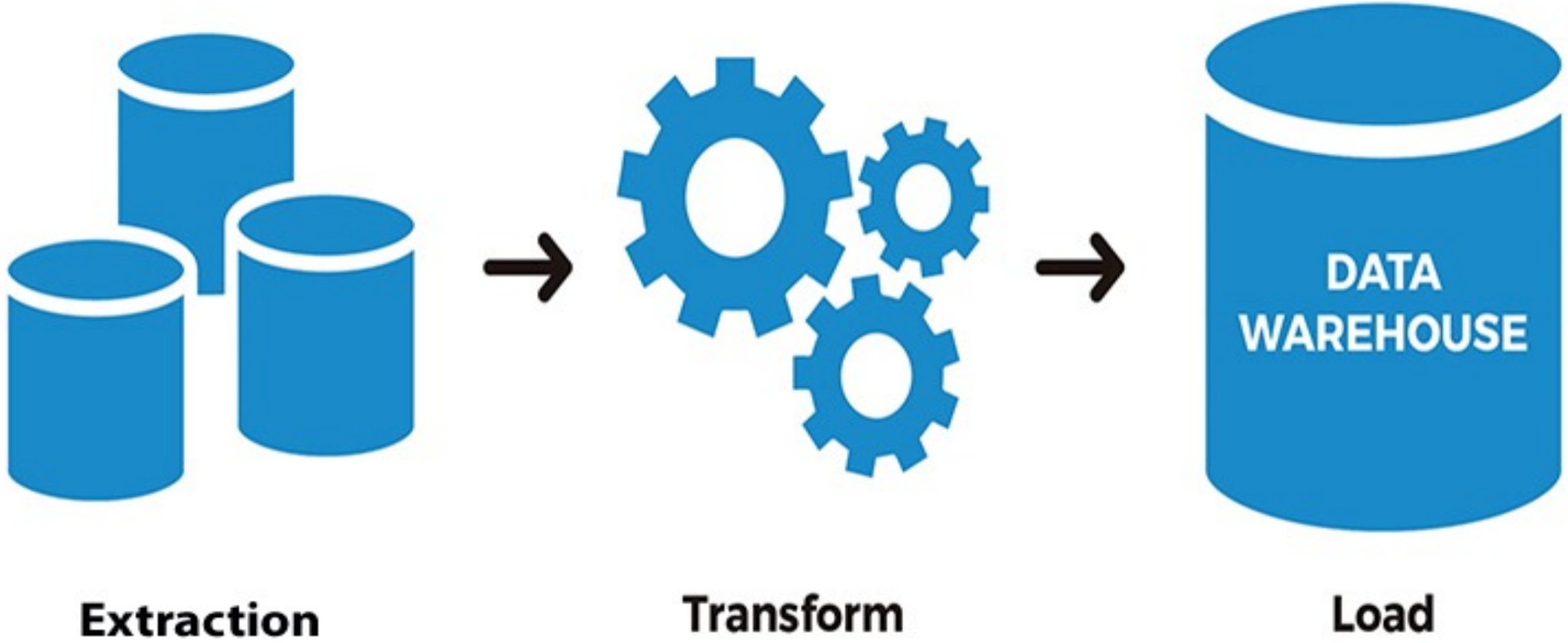
What is Data Migration?

YouTube: Techlake



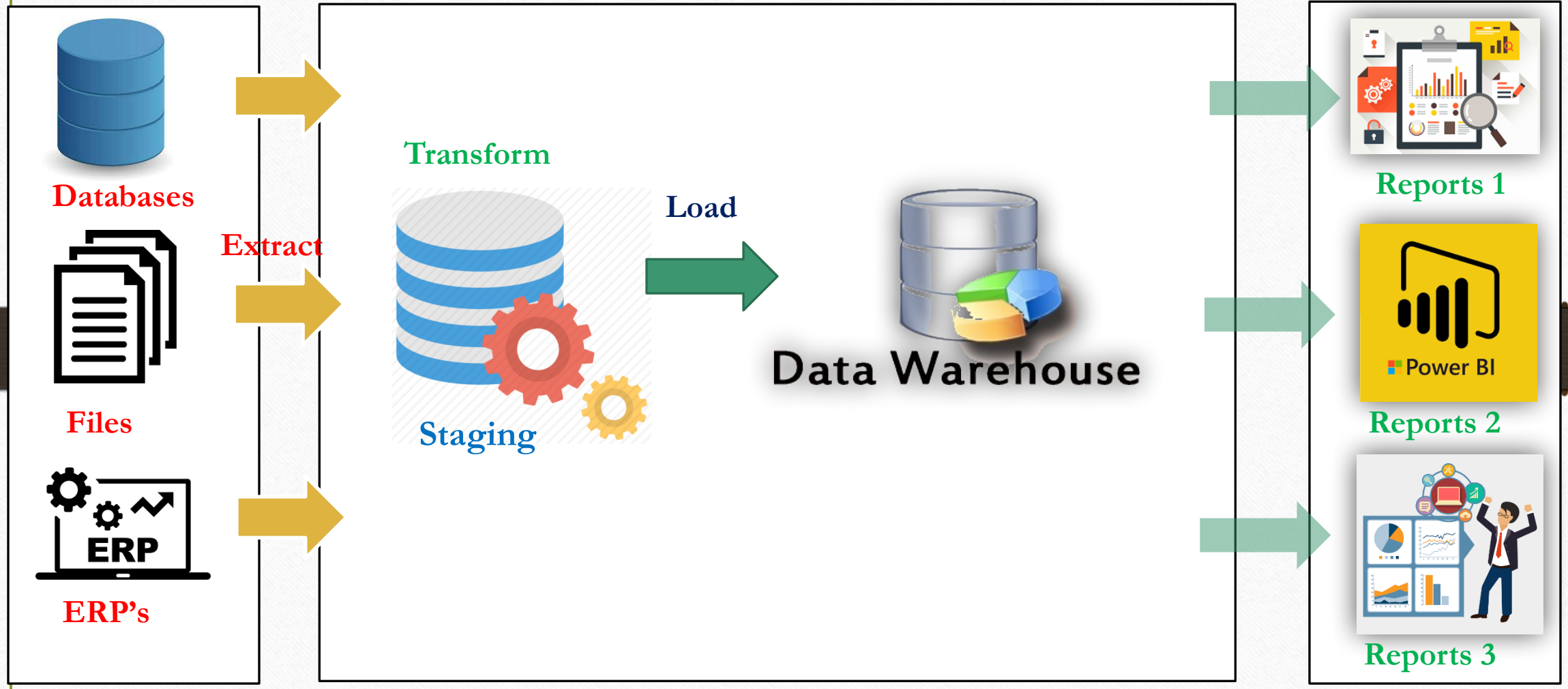
What is ETL

YouTube: Techlake



Traditional DataWarehouse Flow

YouTube: Techlake



ETL Side Major Transformations flow

YouTube: Techlake

Data Extraction

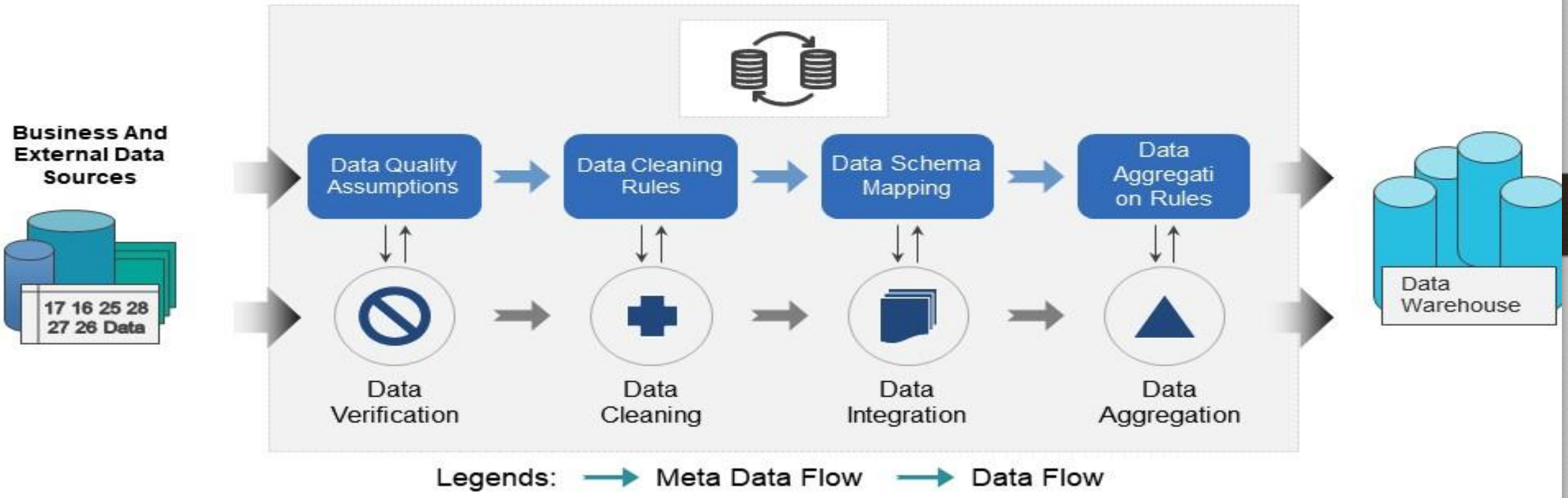
This slide is 100% editable. Adapt it to your needs and capture your audience's attention.

Data Transformation

This slide is 100% editable. Adapt it to your needs and capture your audience's attention.

Data Load

This slide is 100% editable. Adapt it to your needs and capture your audience's attention.



What is OLTP System?

- Database Systems have been used traditionally for OLTP
  - clerical data processing tasks
  - detailed, up to date data
  - structured repetitive tasks
  - read/update a few records
  - isolation, recovery and integrity are critical

ERP Process Flow and OLTP Transactions data

YouTube: Techlake

Customer ordering at online store

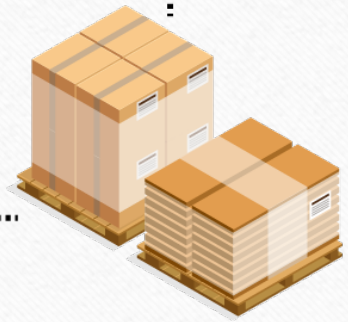
E-commerce operator ERP system

NGL Warehouse management system

Customer

Fedex / UPS Pick up

Ship out



Pick / Pack

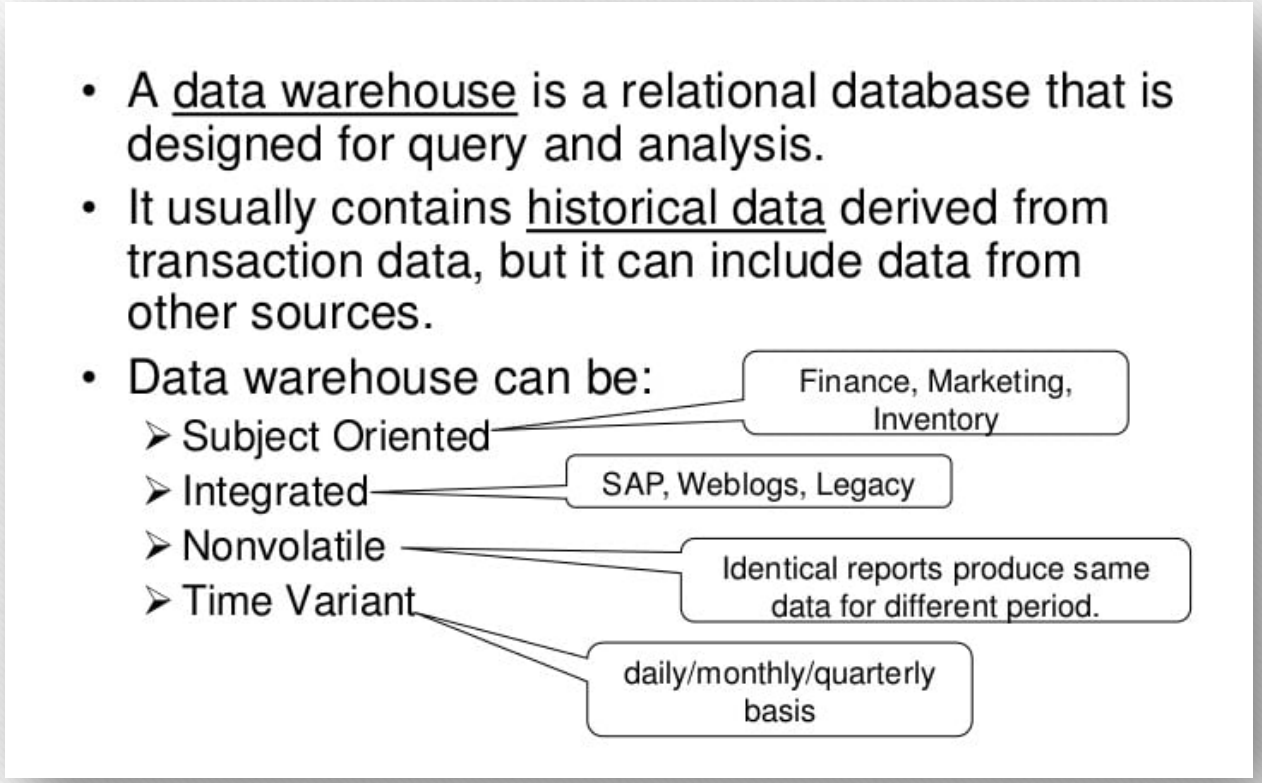
What is a OLAP?

- Defined in many different ways, but not rigorously.
  - A decision support database that is maintained separately from the organization's operational database
  - Support information processing by providing a solid platform of consolidated, historical data for analysis.
- "**A data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management's decision-making process**."—W. H. Inmon

What is Data Warehouse?

YouTube: Techlake

A data warehouse is a type of data management system that is designed to enable and support business intelligence (BI) activities, especially analytics. Data warehouses are solely intended to perform queries and analysis and often contain large amounts of historical data. The data within a data warehouse is usually derived from a wide range of sources such as application log files and transaction applications.

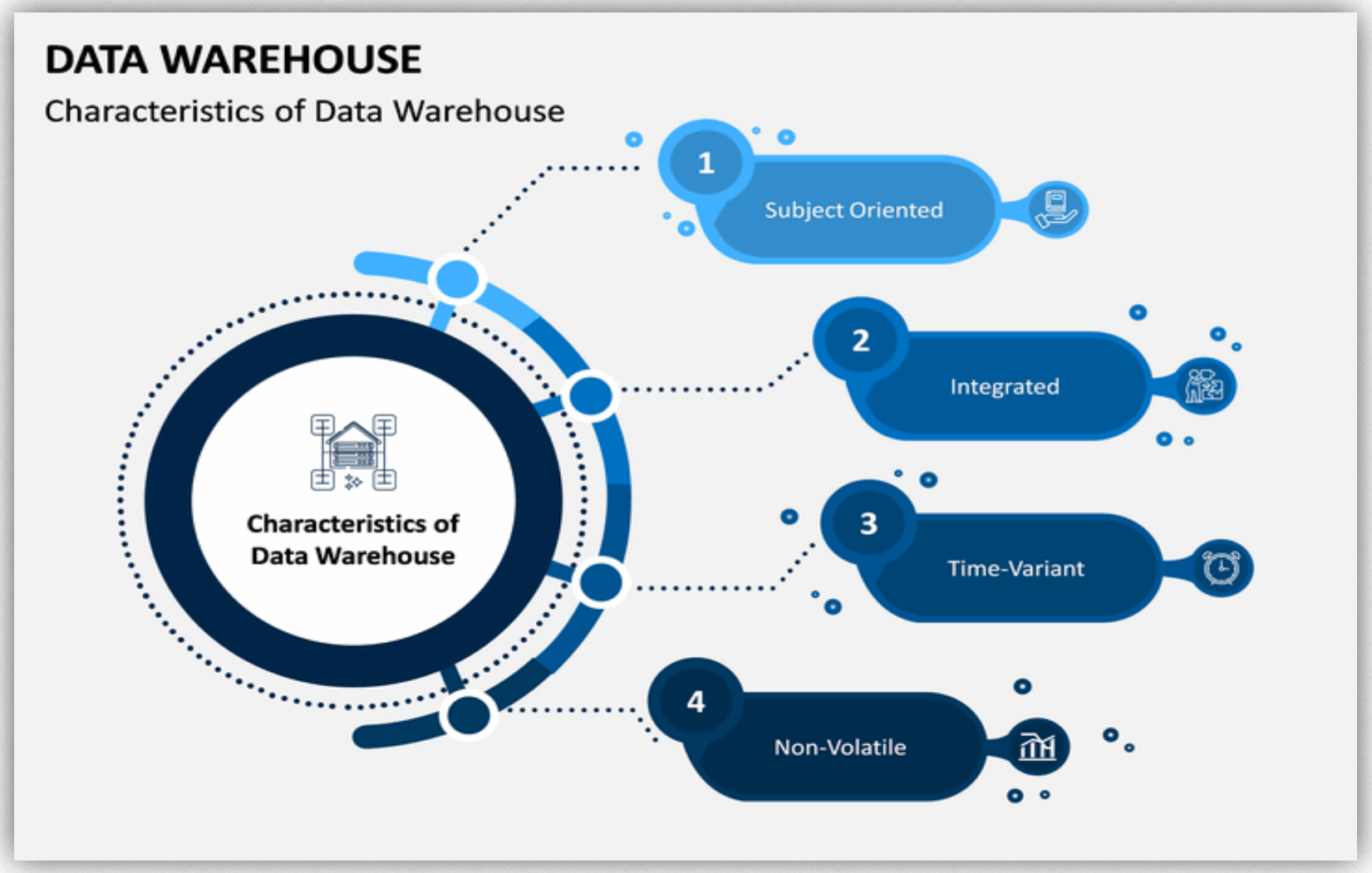


9

Top 10 Benefits of a Data Warehouse

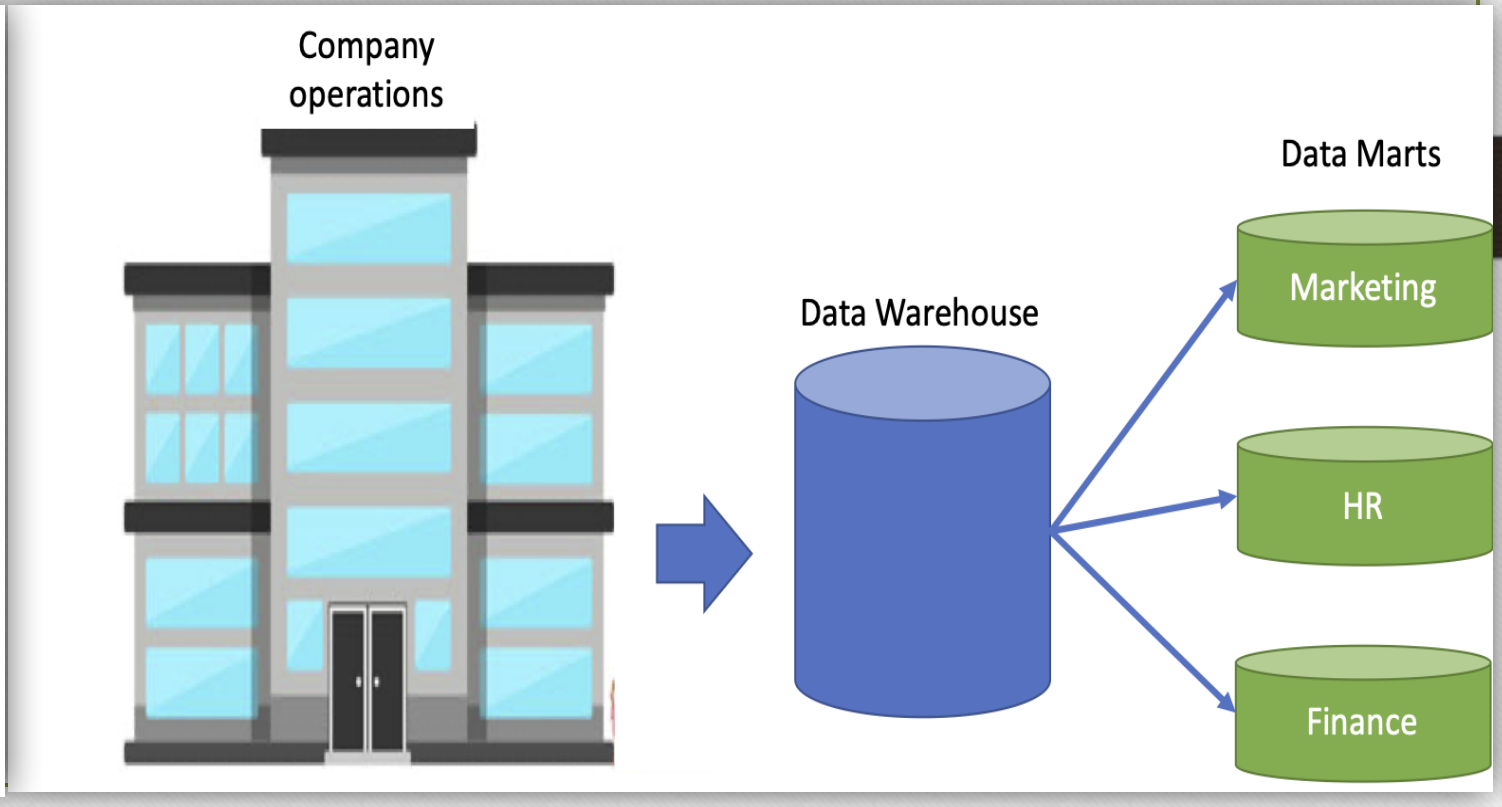
1.  Enables Historical Insight
2.  Enhances Conformity and Quality of Data
3.  Boosts Efficiency
4.  Increase the Power and Speed of Data Analytics
5.  Drives Revenue in Increasing Levels
6.  High Scalability
7.  Interoperates with On-Premise and Cloud
8.  Boosts Data Security
9.  Higher Query Performance and Insight
10.  Provides Major Competitive Advantage

A data warehouse is a **subject-oriented**, **integrated**, **time-variant** and **non-volatile** collection of data in support of management's decision making process.



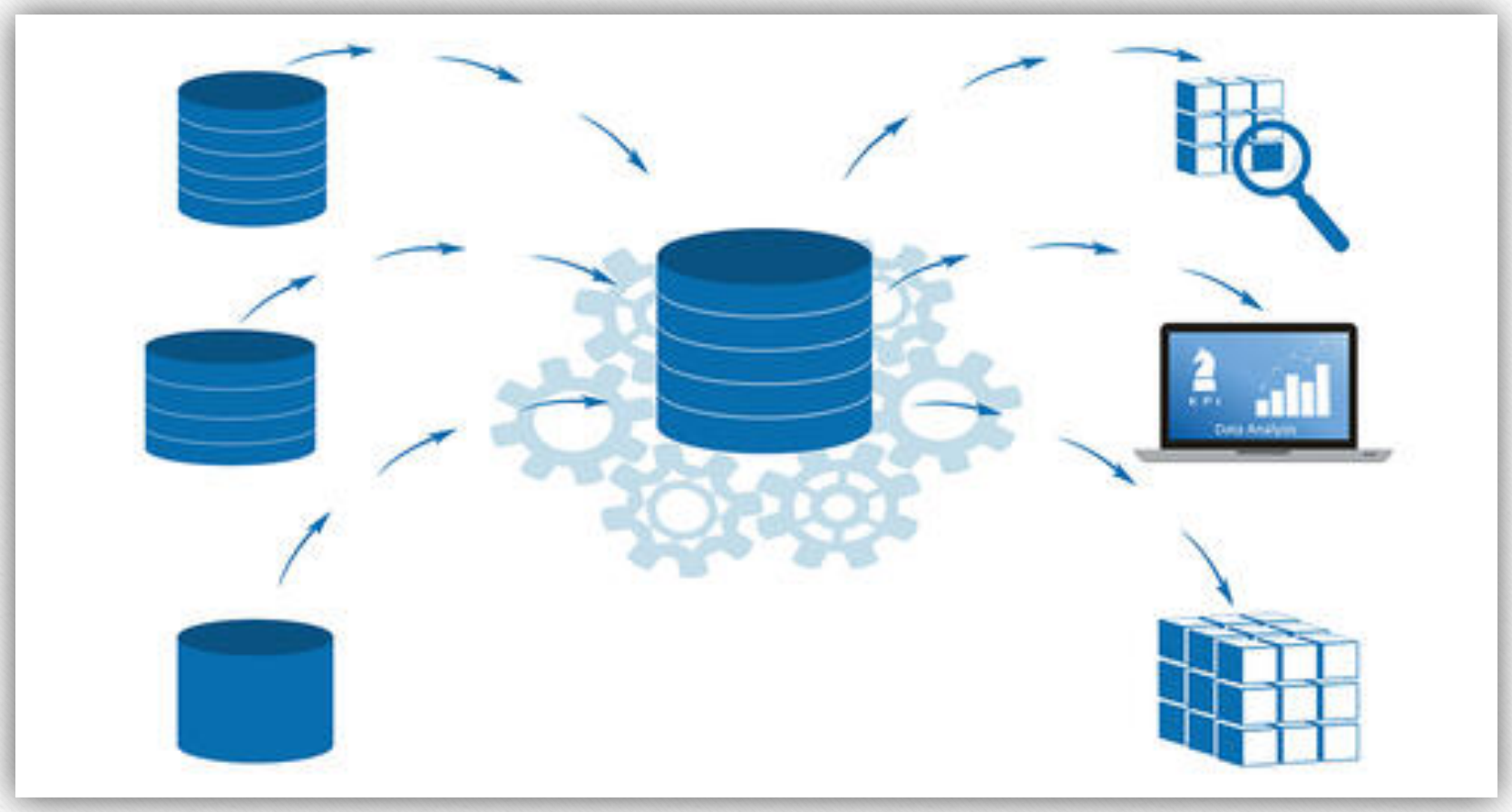
Subject-Oriented

A data warehouse can be used to analyze a particular subject area. For example, "sales" can be a particular subject.



Integrated

A data warehouse integrates data from multiple data sources. For example, source A and source B may have different ways of identifying a product, but in a data warehouse, there will be only a single way of identifying a product.



Time-Variant

Historical data is kept in a data warehouse. For example, one can retrieve data from 3 months, 6 months, 12 months, or even older data from a data warehouse. This contrasts with a transactions system, where often only the most recent data is kept. For example, a transaction system may hold the most recent address of a customer, where a data warehouse can hold all addresses associated with a customer.

Non-volatile

Once data is in the data warehouse, it will not change. So, historical data in a data warehouse should never be altered. Ralph Kimball provided a more concise definition of a data warehouse:

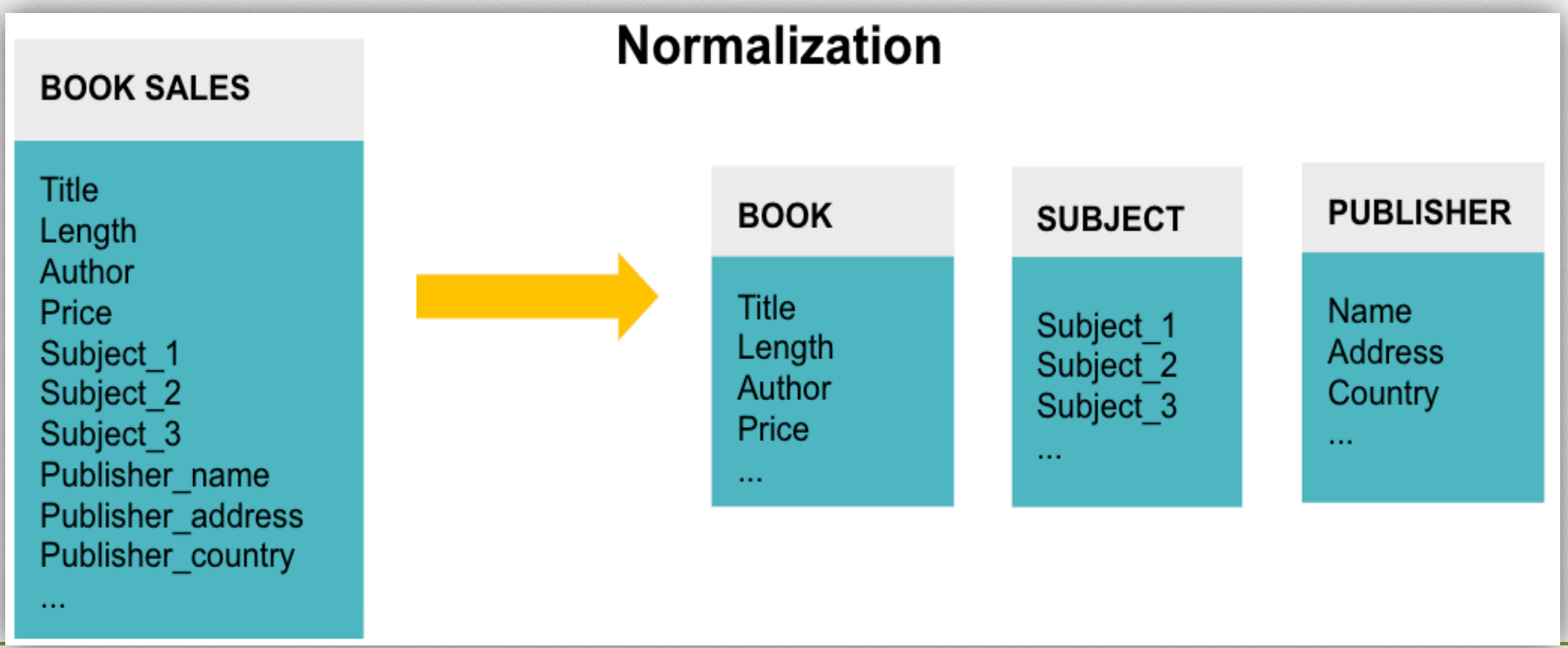
No- Update

Only- Analysis

Normalization

Database normalization is the process of structuring a database, usually a relational database, in accordance with a series of so-called normal forms in order to reduce data redundancy and improve data integrity. It was first proposed by **E.F Codd** as part of his relational model.. We call a relation normalized if:

1. it does not contain any redundancy
2. it does not cause maintenance problems
3. it is an accurate representation of the data

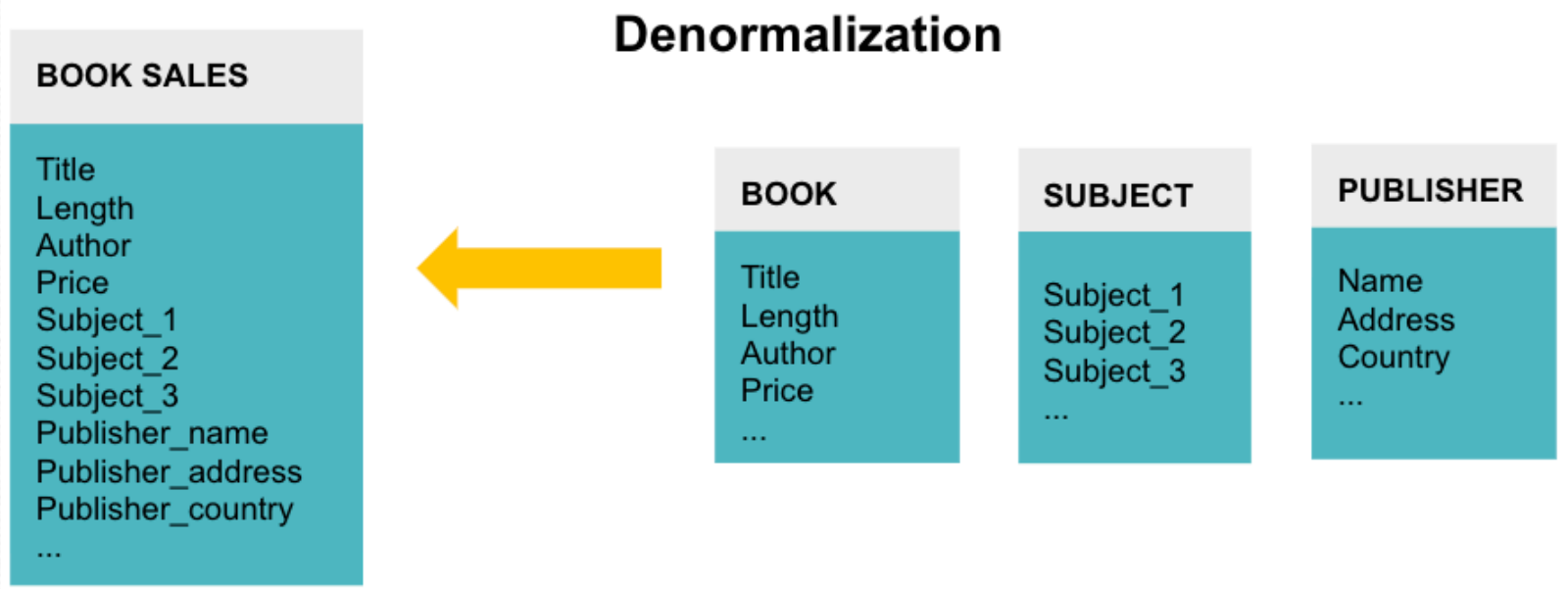


16

Denormalization

Denormalization is a technique used by database administrators to optimize the efficiency of their database infrastructure. This method allows us to add redundant data into a normalized database to alleviate issues with database queries that merge data from several tables into a single table. The denormalization concept is based on the definition of normalization that is defined as arranging a database into tables correctly for a particular purpose.

When we normalize tables, we break them into multiple smaller tables. So when we want to retrieve data from multiple tables, we need to perform some kind of join operation on them. In that case, we use the denormalization technique that eliminates the drawback of normalization.

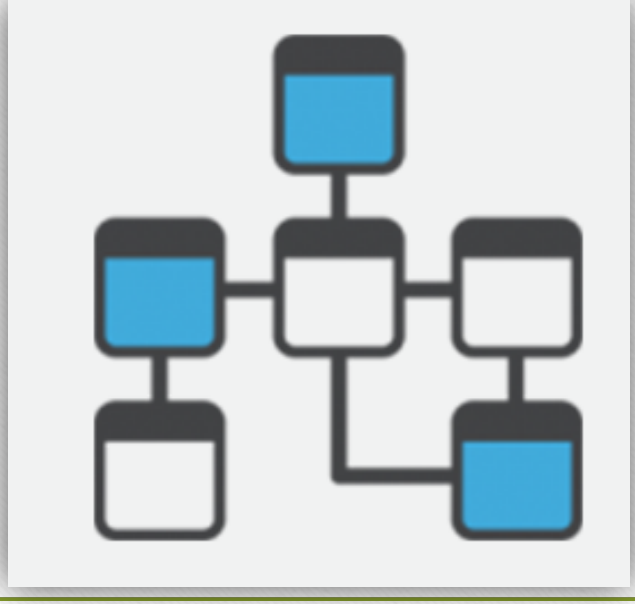


17

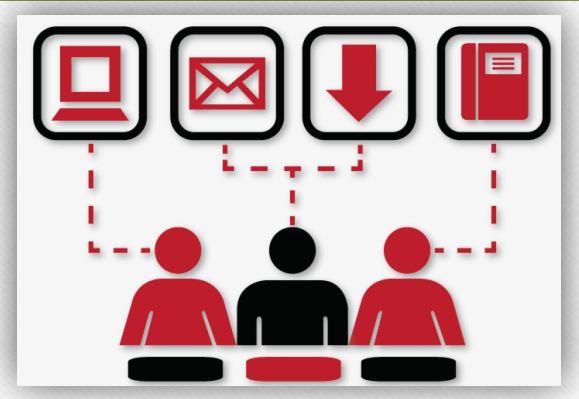
Data Models

It would be wildly inefficient to store all your data in one massive table. So, your data warehouse contains many tables that you can join together to get specific information. The main tables is called a **fact table**, and **dimension tables** surround it.

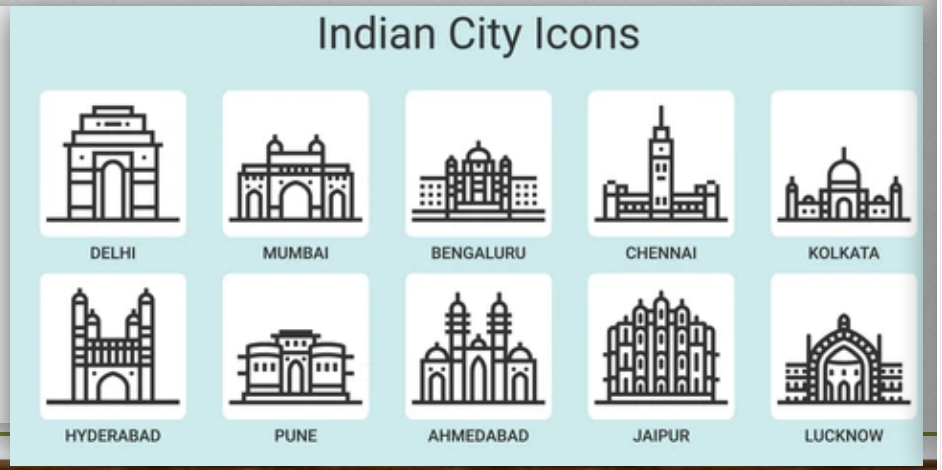
The first step in designing a data warehouse is to build a conceptual data model that defines the data you want and the high-level relationships between them.



Dimension table



1) A dimension table contains dimensions of a fact.
2) They are joined to fact table via a foreign key.
3) Dimension tables are de-normalized tables.
4) The Dimension Attributes are the various columns in a dimension table
5) No set limit set for given for number of dimensions
6) The dimension can also contain one or more hierarchical relationships
7) Dimensions offers descriptive characteristics of the facts with the help of their attributes



What is Fact Table?

YouTube: Techlake

1. Fact table contains measurements, metrics, and facts about a business process while the Dimension table is a companion to the fact table which contains descriptive attributes to be used as query constraining.
2. Fact table is located at the center of a star or snowflake schema, whereas the Dimension table is located at the edges of the star or snowflake schema.
3. Fact table is defined by their grain or its most atomic level whereas Dimension table should be wordy, descriptive, complete, and quality assured.
4. Fact table helps to store report labels whereas Dimension table contains detailed data.
5. Fact table does not contain a hierarchy whereas the Dimension table contains hierarchies.

Types of Dimensions and Fact tables

YouTube: Techlake

Types of Dimensions

1. Junk Dimensions
2. Conformed Dimensions
3. Degenerate Dimensions
4. Rapidly Changing Dimensions
5. Role Playing Dimensions
6. Inferred Dimensions
7. Shrunken Dimensions
8. Static Dimensions
9. Slowly Changing Dimensions

Types of Facts

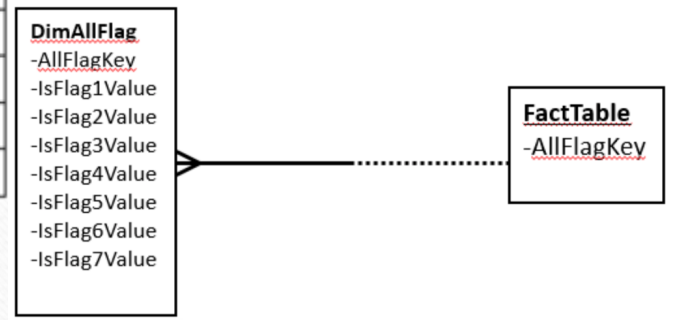
1. Transaction Fact
2. Snapshot Fact
3. Accumulated Fact
4. Fact-less Fact

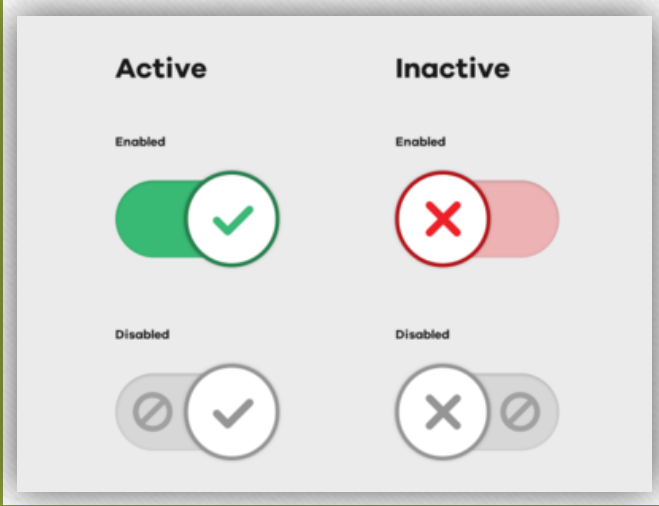
YouTube: Techlake

Junk Dimensions

A single table that includes a compilation of unrelated and different attributes is referred as the junk dimension table. This table is meant to avoid including a huge amount of foreign keys in the fact table. This table is often generated to handle foreign keys generated by the rapidly changing dimensions.

AllFlagKey	IsFlag1Value	IsFlag2Value	IsFlag3Value	IsFlag4Value	IsFlag5Value	IsFlag6Value	IsFlag7Value
1	Y	Y	Y	Y	Y	Y	Y
2	N	Y	Y	Y	Y	Y	Y
3	Y	N	Y	Y	Y	Y	Y
4	Y	Y	N	Y	Y	Y	Y

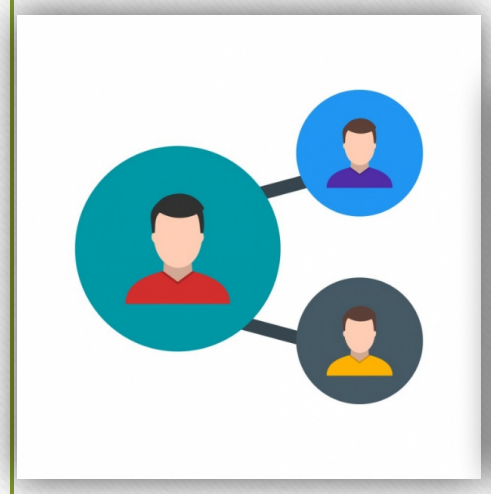


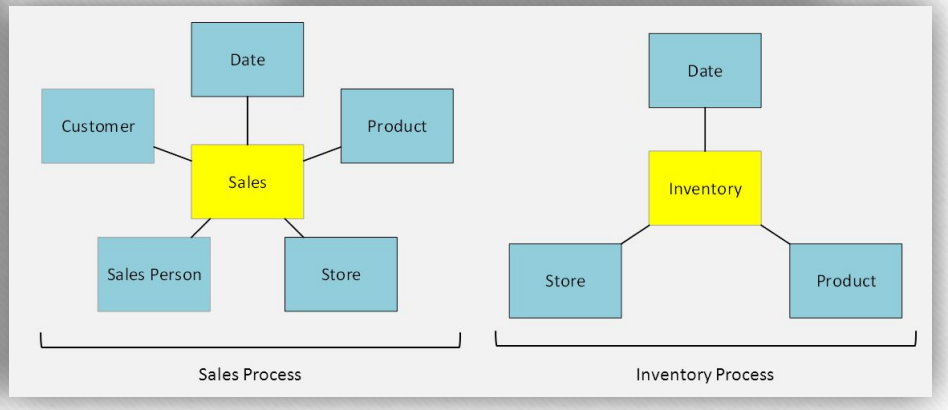


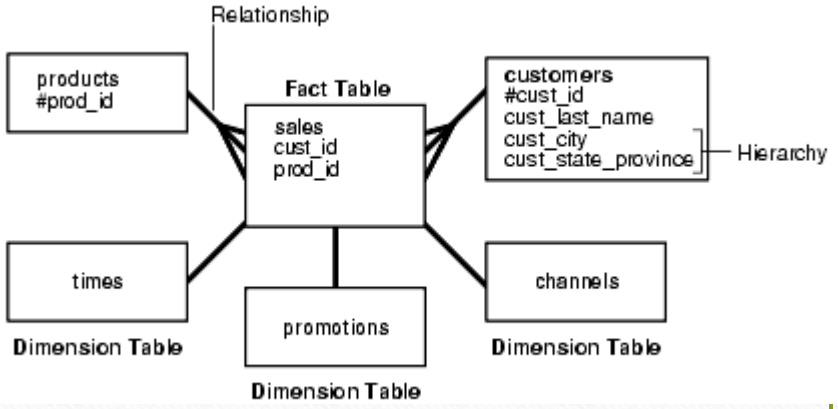
Conformed Dimensions

YouTube: Techlake

A conformed dimension connects to multiple fact tables
A conformed dimension allow a consistent analysis of different business process metrics used across multiple dimensional models. A product dimension may join to the inventory, sales, procurement, and other fact tables. It is important to closely collaborate with data governance team to have a consistent definition for this dimension.



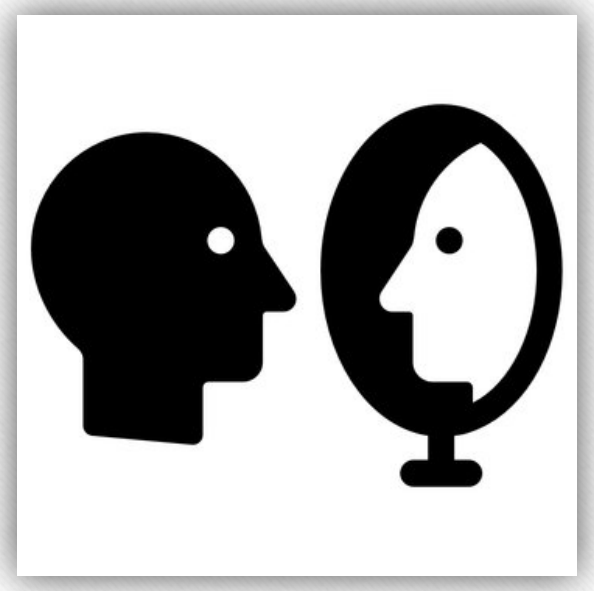


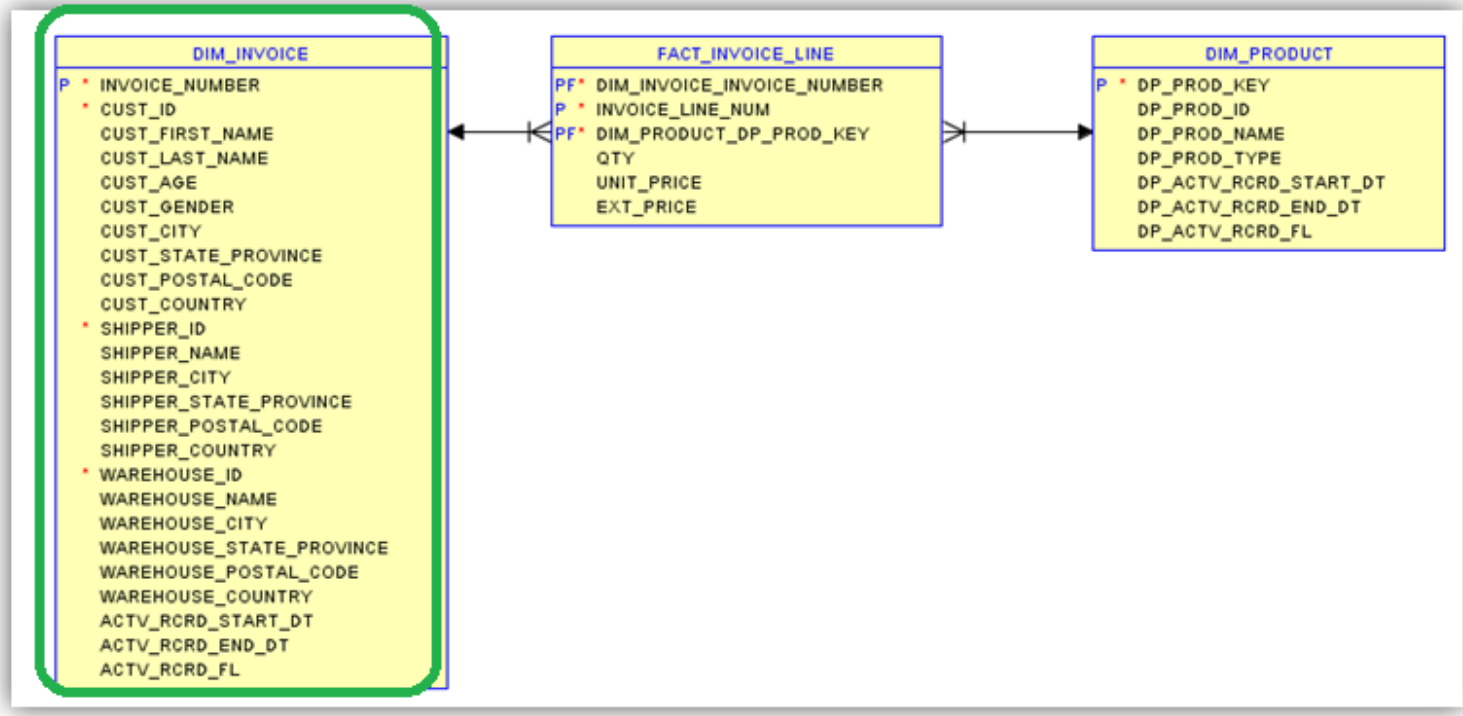


YouTube: Techlake

Degenerate Dimensions

A degenerate dimension is a special dimension like invoice number, check number that is an identifier for a transaction. However, it does not have attributes linked to it as all the important attributes are already part of their respective dimensions. So an invoice may have a customer name attribute but it's already a part of the customer dimension. A degenerate dimension is a part of the fact table but it's not a measure, it is still a dimension.

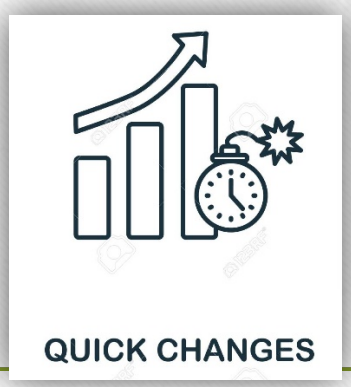




Rapidly Changing Dimensions

YouTube: Techlake

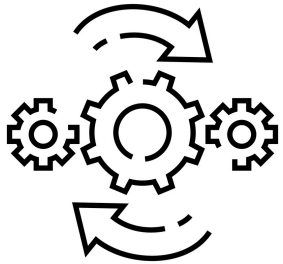
An attribute in the dimension, which is changing frequently, is referred as the rapidly changing attribute. Handling rapidly changing dimension in data warehouse is very difficult because of many performance implications. As we know **slowly changing dimension type 2** is used to maintain the history for the changes. But the problem with type 2 is, with each and every change in the dimension attribute, it adds new row to the table. If in case there are dimensions that are changing a lot, table become larger and may cause serious performance issues. Hence, use of the type 2 may not be the wise decision to implement the rapidly changing dimensions.



DIM_CUSTOMER		
P *	CUST_KEY	NUMBER
	CUST_NAME	VARCHAR2
	CUST_CITY	VARCHAR2
	CUST_STATE	VARCHAR2
	CUST_AGE	NUMBER
	CUST_INCOME	NUMBER
	CUST_LIFETIME_PURCHASES	NUMBER
	CUST_RATING	VARCHAR2
	CUST_ACCOUNT_STATUS	VARCHAR2
	CUST_CREDIT_SCORE	NUMBER
	CUST_GENDER	VARCHAR2
	CUST_ACTV_RCRD_FL	NUMBER
	CUST_ACTV_RCRD_START_DT	DATE
	CUST_ACTV_RCRD_END_DT	DATE
DIM_CUSTOMER_PK (CUST_KEY)		

DIM_CUSTOMER		
P *	CUST_KEY	NUMBER
	CUST_NAME	VARCHAR2
	CUST_CITY	VARCHAR2
	CUST_STATE	VARCHAR2
	CUST_ATTRIBUTE_KEY	NUMBER
	CUST_GENDER	VARCHAR2
	CUST_ACTV_RCRD_FL	NUMBER
	CUST_ACTV_RCRD_START_DT	DATE
	CUST_ACTV_RCRD_END_DT	DATE
DIM_CUSTOMER_PK (CUST_KEY)		

DIM_CUSTOMER_ATTRIBUTE		
P *	CUST_ATTR_KEY	NUMBER
	CUST_ATTR_AGE	VARCHAR2
	CUST_ATTR_INCOME	VARCHAR2
	CUST_ATTR_LIFETIME_PURCHASES	VARCHAR2
	CUST_ATTR_RATING	VARCHAR2
	CUST_ATTR_ACCOUNT_STATUS	VARCHAR2
	CUST_ATTR_CREDIT_SCORE	VARCHAR2
	CUST_ATTR_ACTV_RCRD_FL	NUMBER
	CUST_ATTR_ACTV_RCRD_START_DT	DATE
	CUST_ATTR_ACTV_RCRD_END_DT	DATE
DIM_CUSTOMER_ATTRIBUTE_PK (CUST_ATTR_KEY)		



Role Playing Dimensions

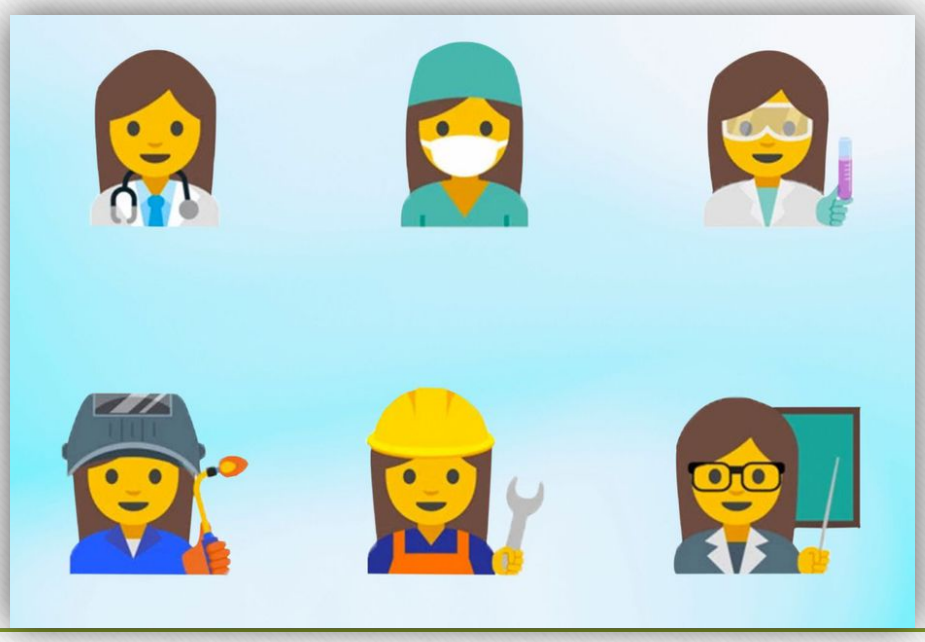
YouTube: Techlake

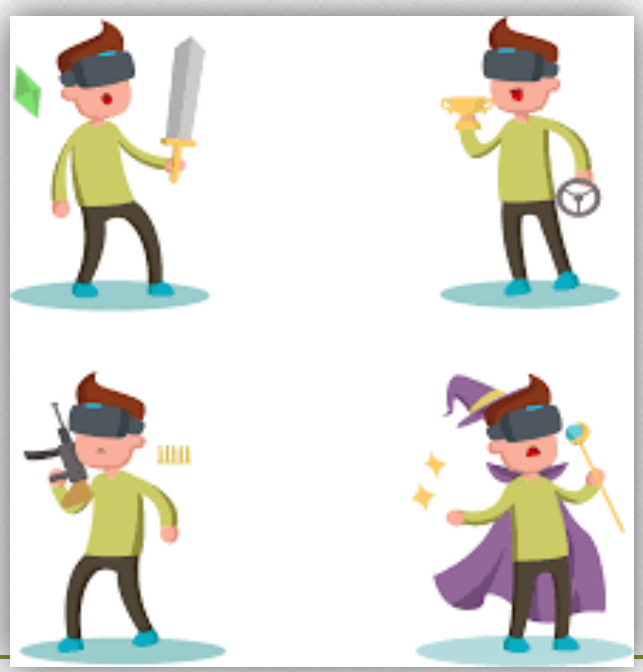
Role Playing Dimension

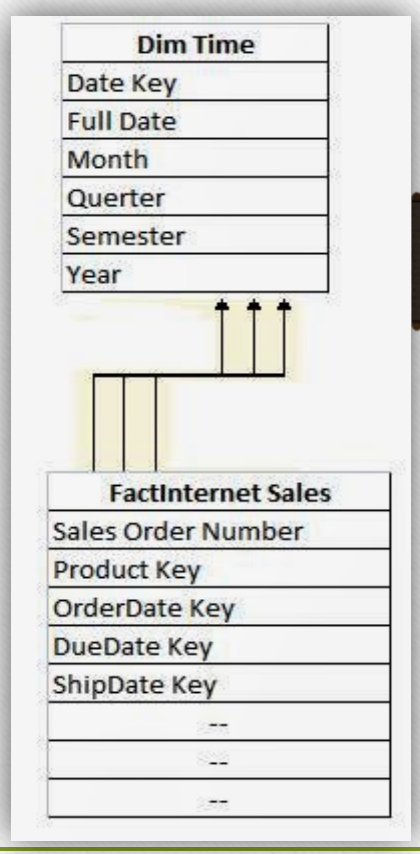
A database Dimension that acts as multiple dimensions with in a cube is called as Role Playing Dimension. From the same table if we have multiple foreign keys in fact tables then the table acts differently for each key attribute.

Example:

Time Dimension is one of the best Example of Role playing Dimension, you can have one Time Dimension called Date and then you can add **ShipDate**, **DueDate** and **OrderDate** as Cube Dimensions.







Inferred Dimensions

YouTube: Techlake

Inferred Dimension, also referred to as **Early coming Fact** or **Late Arrival dimension**.

Sometimes, dimension records might not be ready while loading the fact records. A solution to handle this situation is creating a surrogate key with a null value for the entire other attributes. This process is technically referred as the inferred member or inferred dimension.

Dim_Promo_Code

Promo_Code_PKEY	Promo_Code	Promotion Type	Discount Type	Inferred
1	P1	Marketing	Cashback	0
2	P2	Employee	Instant	0
3	PSpecialOrg	Unknown	Unknown	1

Fact _Tickets

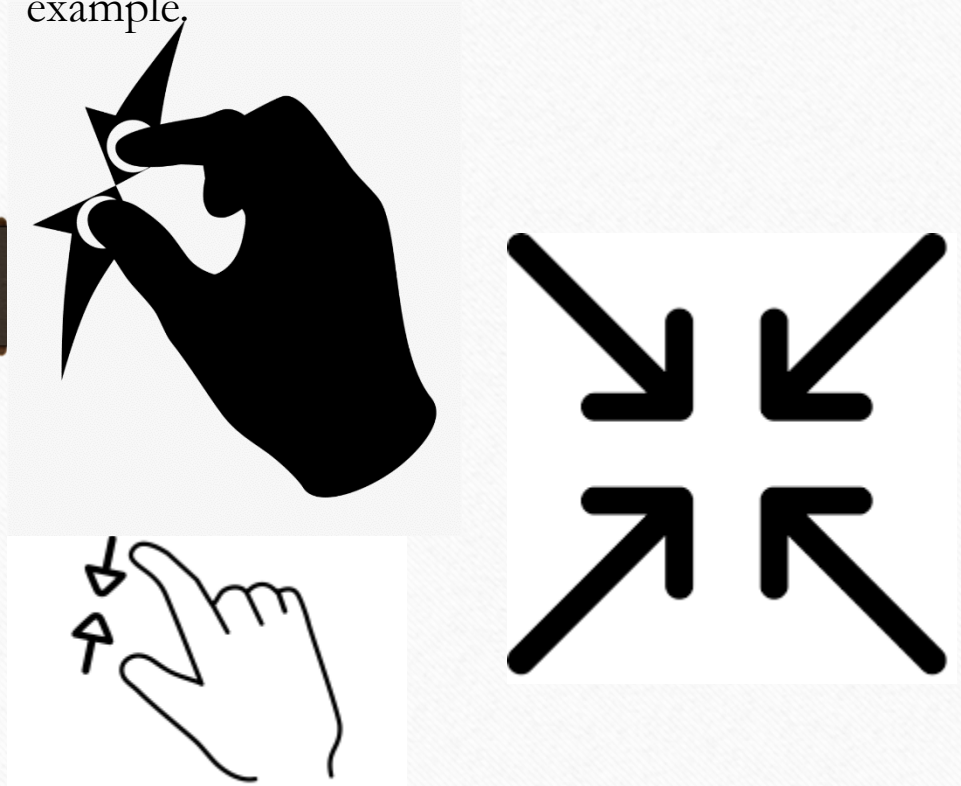
Ticket_ID	TransactionDate_FKey	Promo_Code	Promo_Code_FKEY
T1	20180819	P1	1
T2	20180819	P2	2
T3	20180819	PSpecialOrg	-1

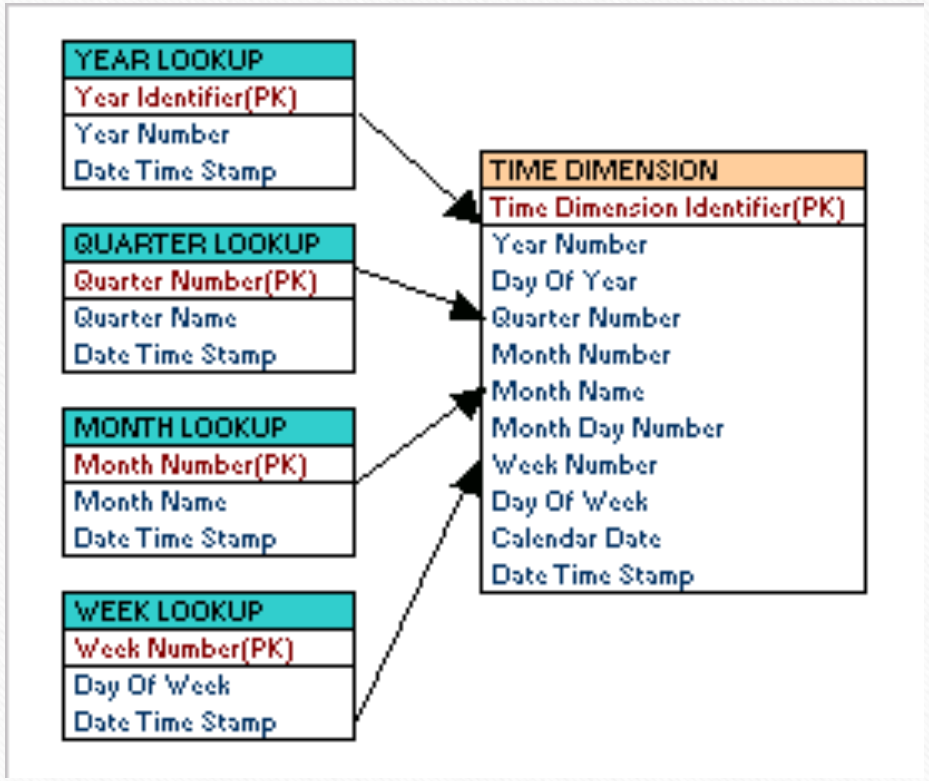
Shrunken Dimensions

YouTube: Techlake

The shrunken dimension table is referred as the subset of alternative dimension.

Month could be a shrunken dimension of date(time or calendar) table, which could be connected to a fact which has grain of month like monthly sales. But this is a very classic example, let us deep dive a little with another example.





Static Dimensions

YouTube: Techlake

Static Dimensions : (no-source)

A static dimension is a dimension that is not loaded from the source system.
The static dimensions are generated using a SQL script or a stored procedure and are manually loaded. The time dimension is a classic example of a static dimension. Also, status codes dimension is mostly a static dimension.

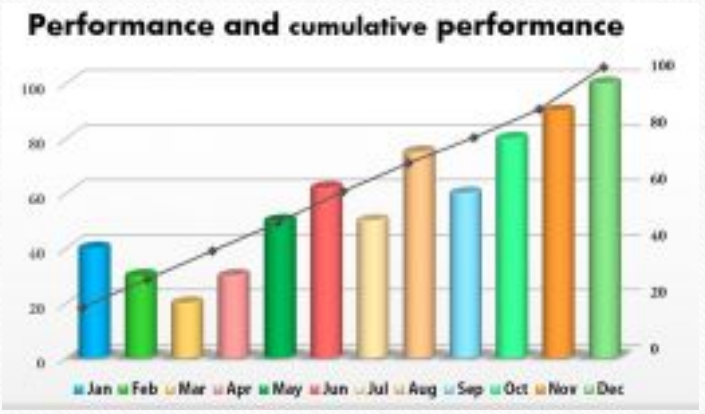
A few common examples of the static dimensions would be
"DIM_SOURCE_SYSTEMS",

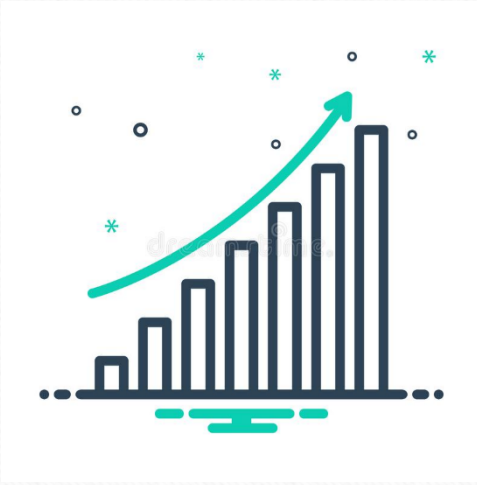
id	source_name	type	status
1	SAP	ERP	Y
2	Oracle	Database	Y
3	DB2	Database	Y
4	SalesForce	ERP	Y
5	systemlogs	file	Y

Types of Facts

YouTube: Techlake

1. **Additive**: Additive facts can be used with any aggregation function like Sum(), Avg() etc. Example is Quantity, sales amount etc
2. **Semi-Additive**: Semi-additive facts are facts that can be summed up for some of the dimensions in the fact table, but not the others. For example, Consider bank account details. You cannot apply the Sum() on the bank balance that does not give useful results but min() and max() function may return useful information
3. **Non-Additive:** Non-additive facts are facts that cannot be summed up for any of the dimensions present in the fact table. For example of non-additive fact is any kind of ratio or percentage. Non numeric facts can also be a non-additive facts



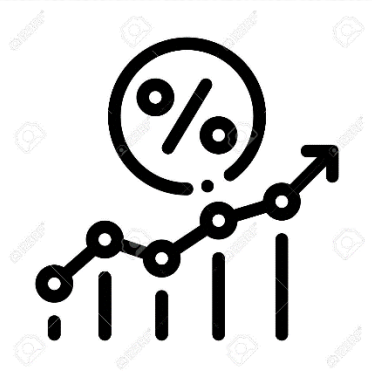


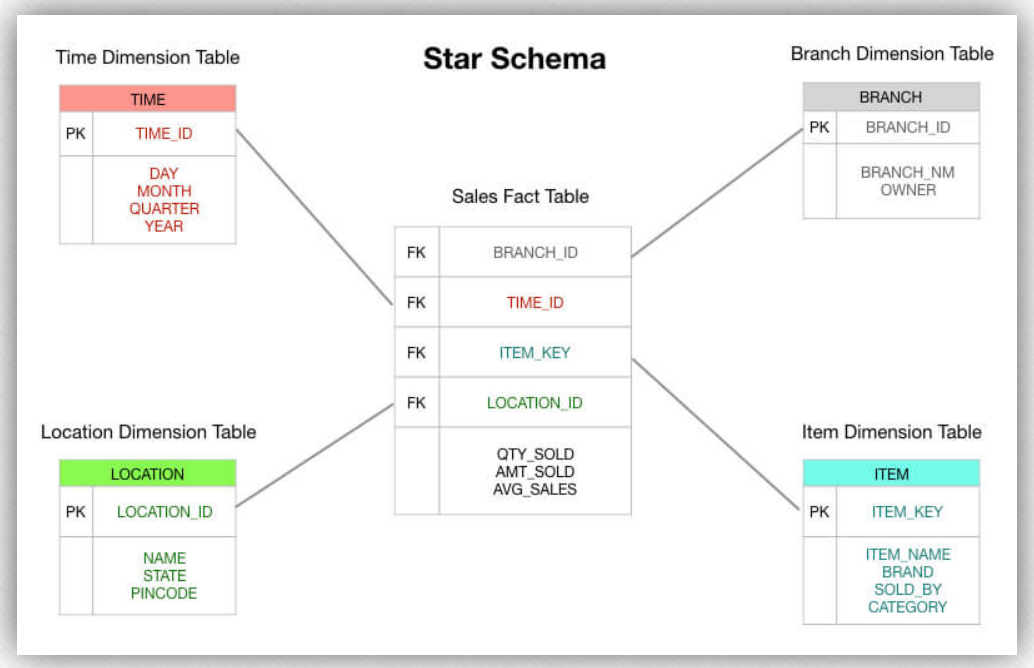
Transaction Fact

YouTube: Techlake

Transaction Fact

Transaction fact tables are easy to understand: a customer or business process does some thing; you want to capture the occurrence of that thing, and so you record a transaction in your data warehouse and you're good to go.



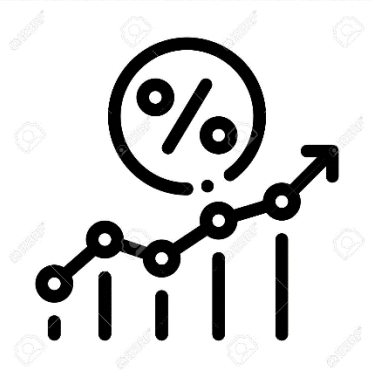


Cumulative Fact

YouTube: Techlake

Cumulative

This type of fact table describes what has happened over a period of time. For example, this fact table may describe the total sales by product by store by day. The facts for this type of fact tables are mostly additive facts. The first example presented here is a cumulative fact table.

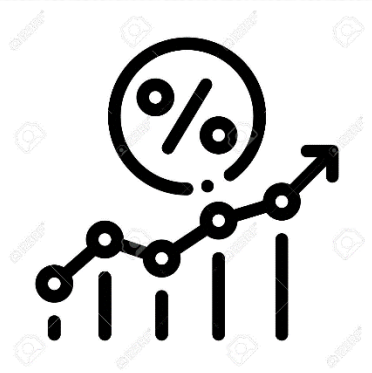


FACT_CLAIM_PROCESSING		
P *	CLAIM_KEY	NUMBER
P *	CUSTOMER_KEY	NUMBER
P *	POLICY_KEY	NUMBER
*	CLAIM_DATE	DATE
	INVESTIGATION_DATE	DATE
	DAYS_TO_INVESTIGATION	NUMBER
	REVIEW_DATE	DATE
	DAYS_TO_REVIEW	NUMBER
	DESCISION_DATE	DATE
	DAYS_TO_DESCISION	NUMBER
	PAYMENT_DATE	DATE
	DAYS_TO_PAYMENT	NUMBER

Snapshot Fact

YouTube: Techlake

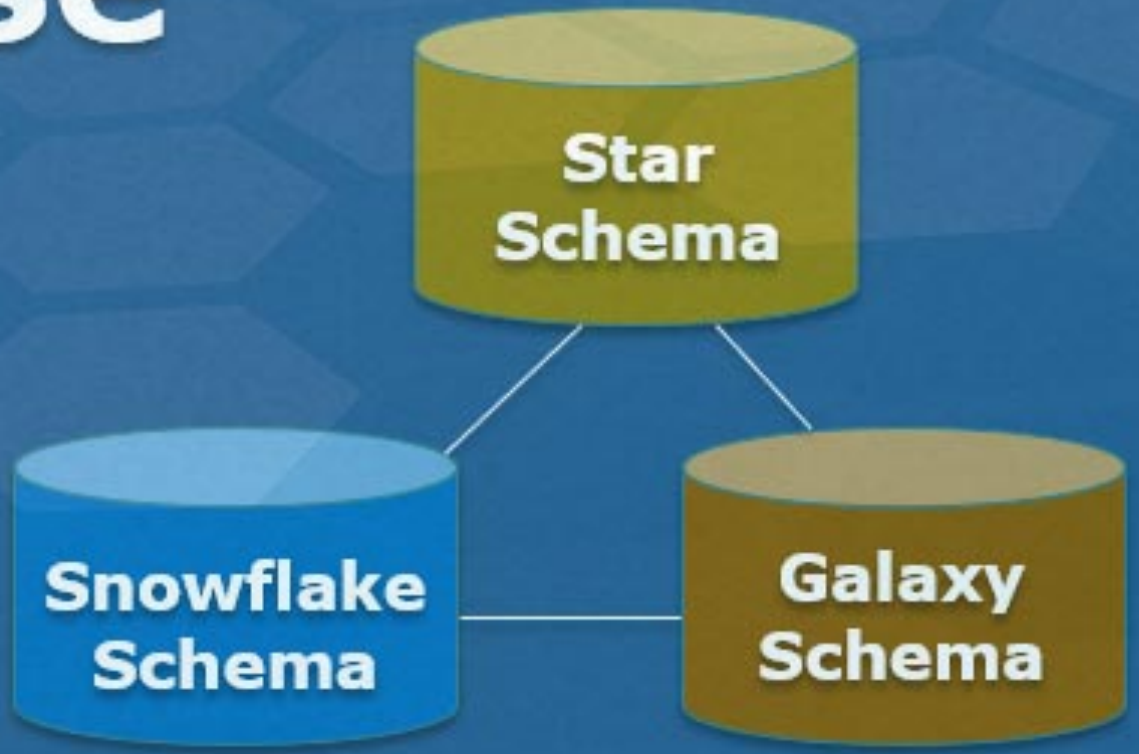
Snapshot: (Periodic) Snapshot fact tables capture the state of the measures based on the occurrence of a status event or at a specified point-in-time or over specified time intervals (week, month, quarter, year, etc.).



FACT_MONTHLY_SALES		
P *	SALE_MONTH_DATE_KEY	NUMBER
P *	SALE_CUSTOMER_KEY	NUMBER
P *	SALE_PRODUCT_KEY	NUMBER
P *	SALE_AUTHOR_GROUP_KEY	NUMBER
P *	SALE_STORE_KEY	NUMBER
	SALE_QTY	NUMBER
	SALE_AMT	NUMBER

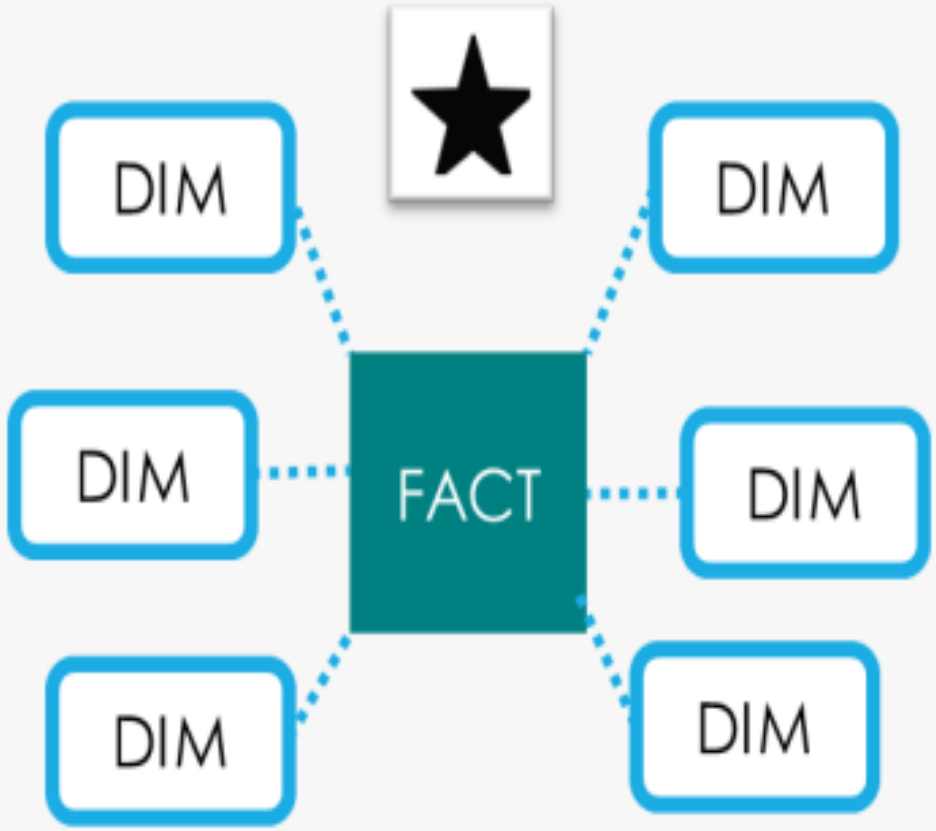
Data Warehouse Schema

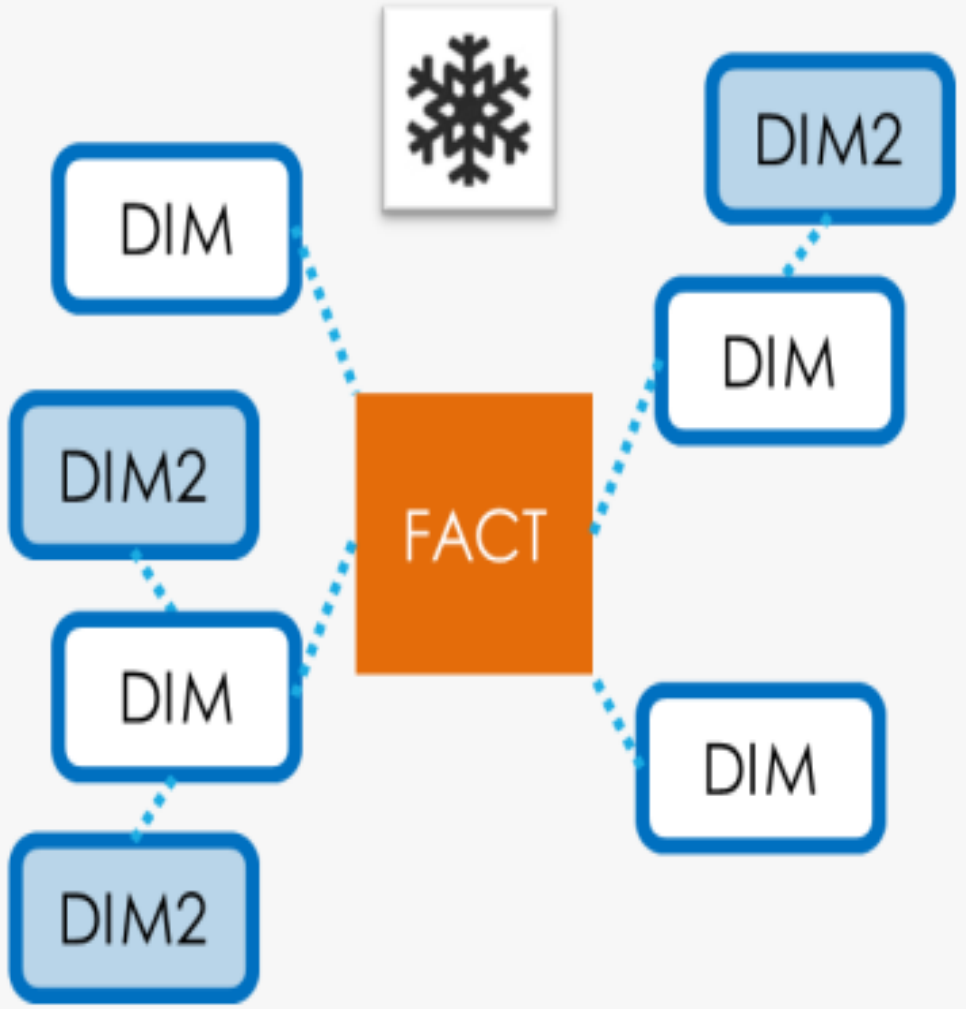
YouTube: Techlake



Star Schema And Snowflake Schema

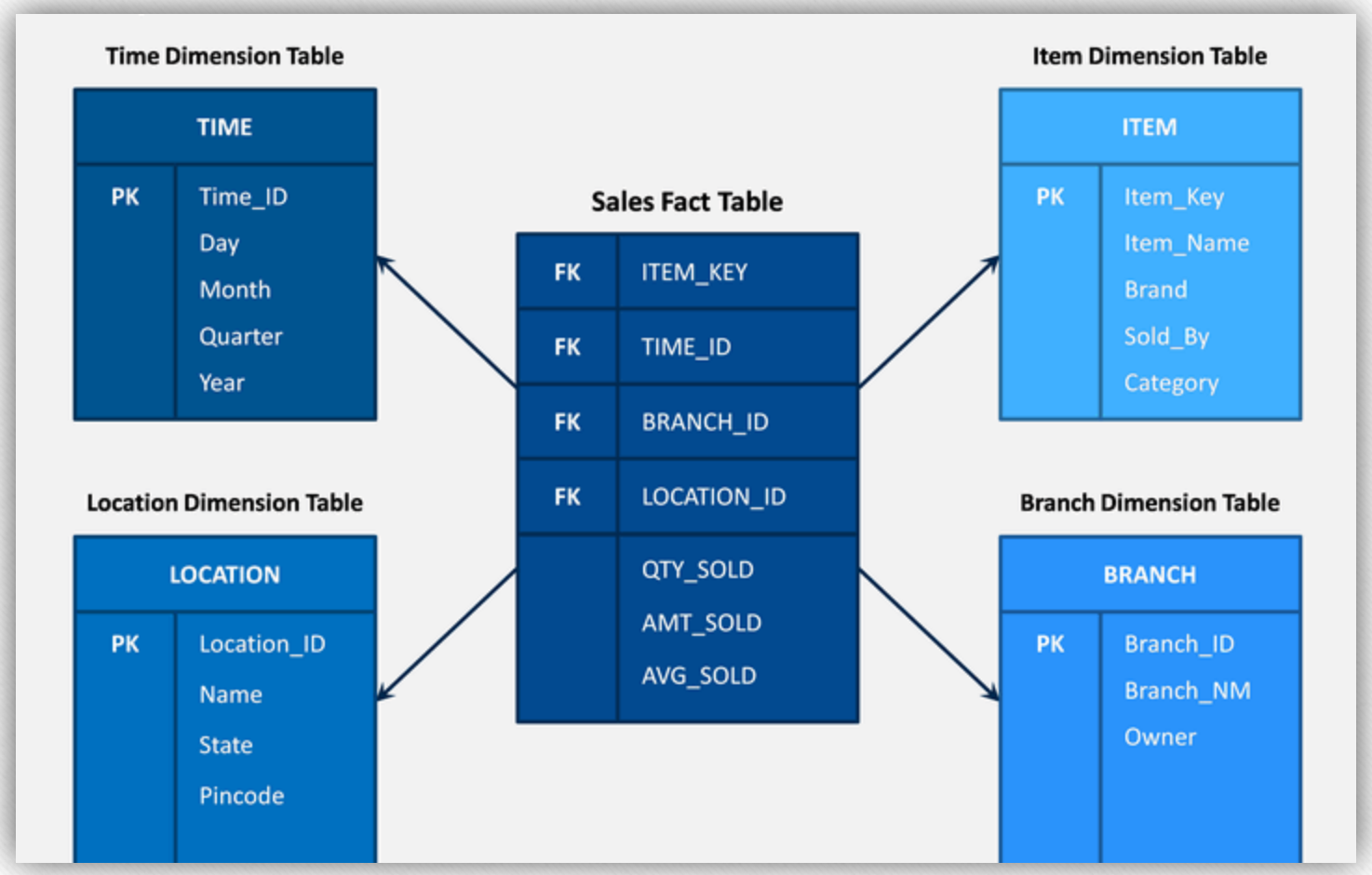
YouTube: Techlake





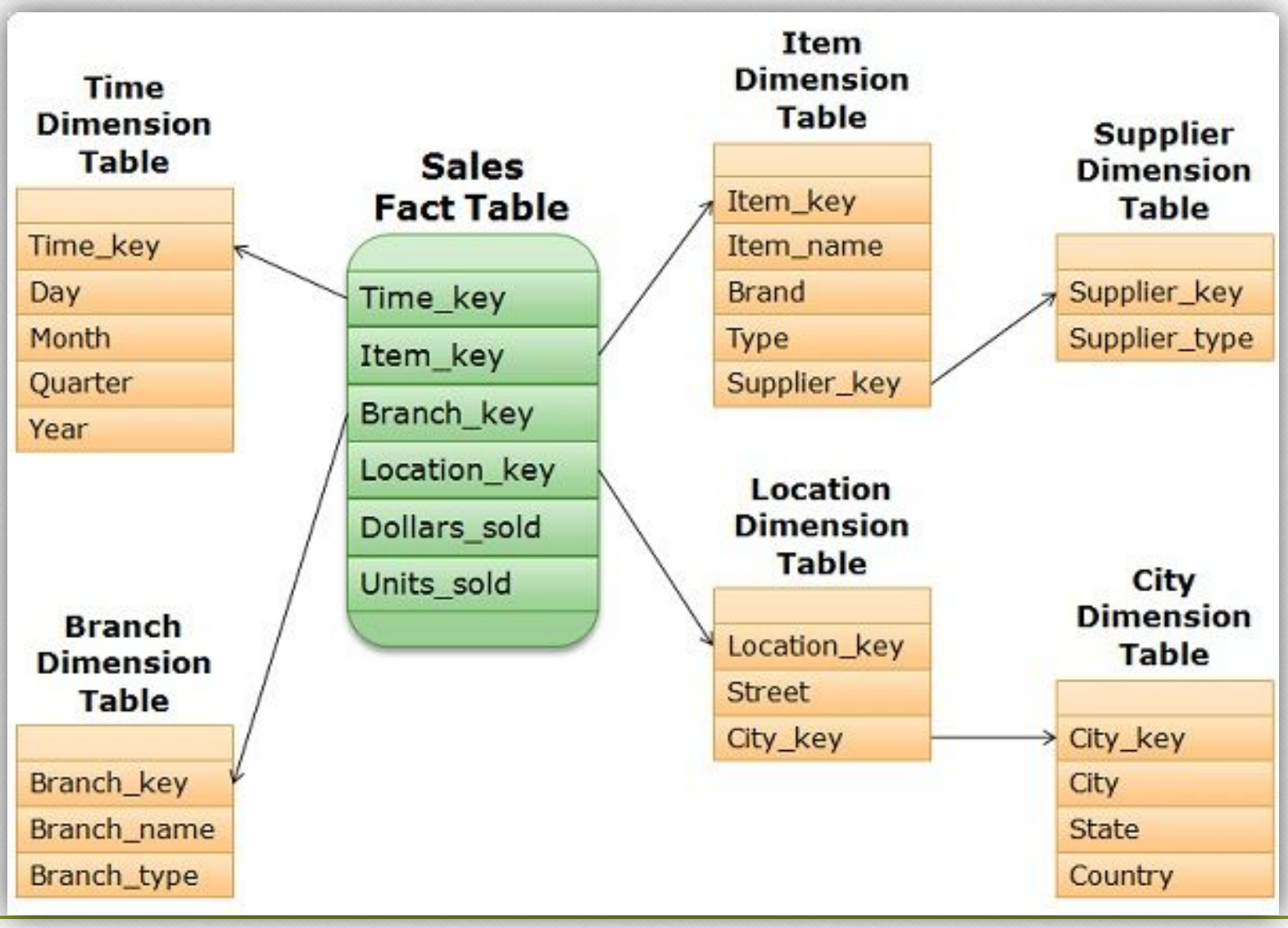
Star Schema

YouTube: Techlake



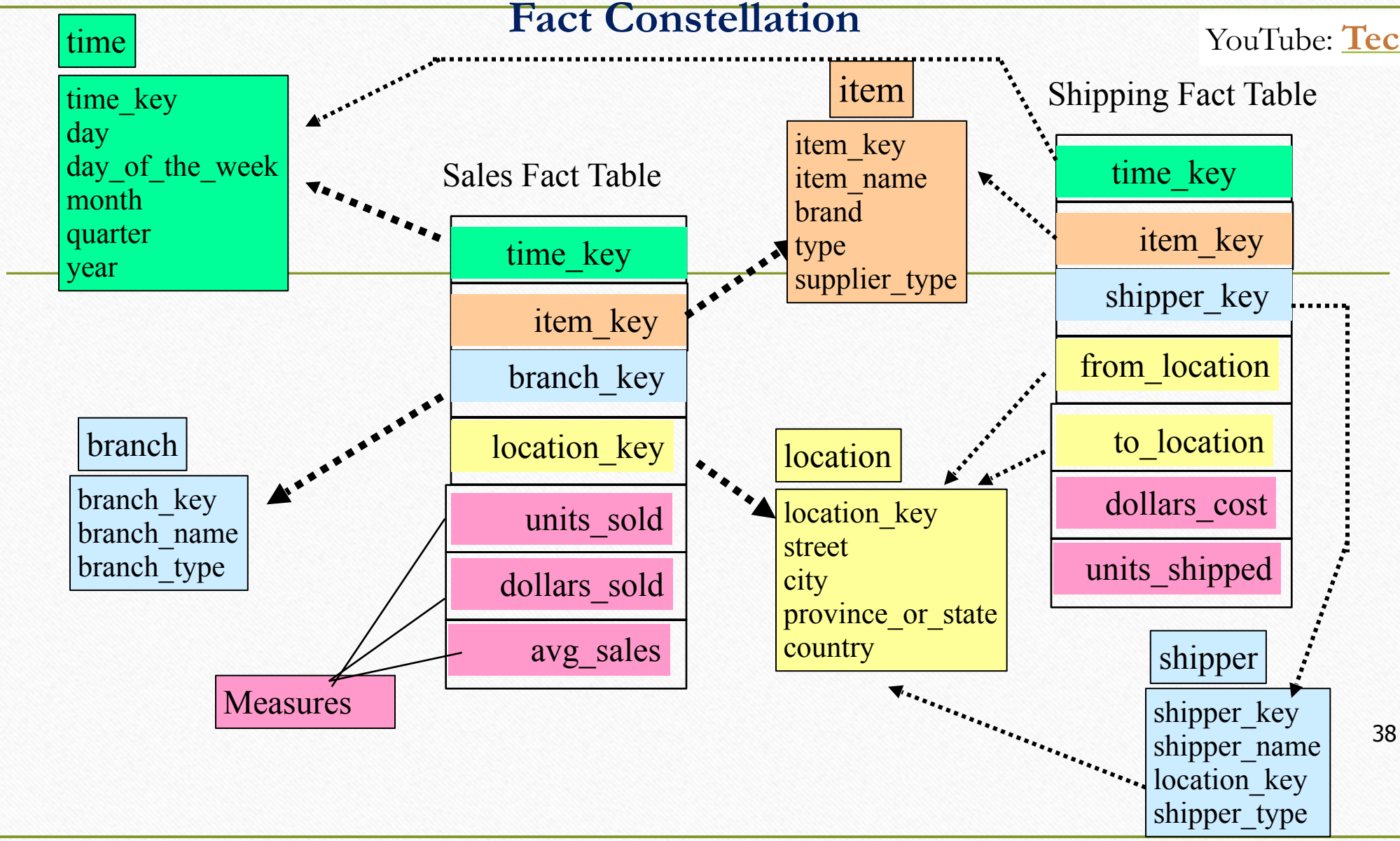
Snowflake Schema

YouTube: Techlake



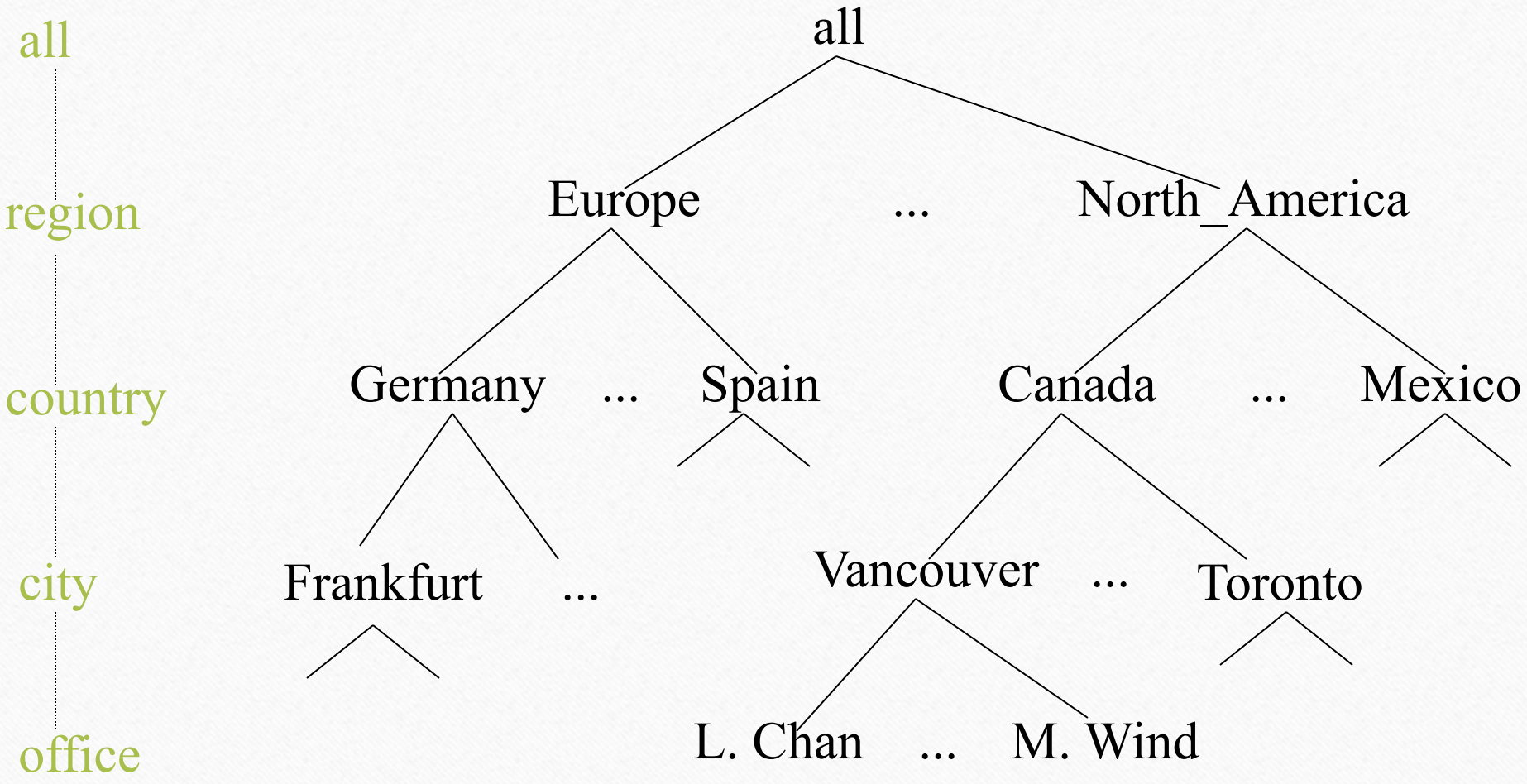
Fact Constellation

YouTube: Techlake



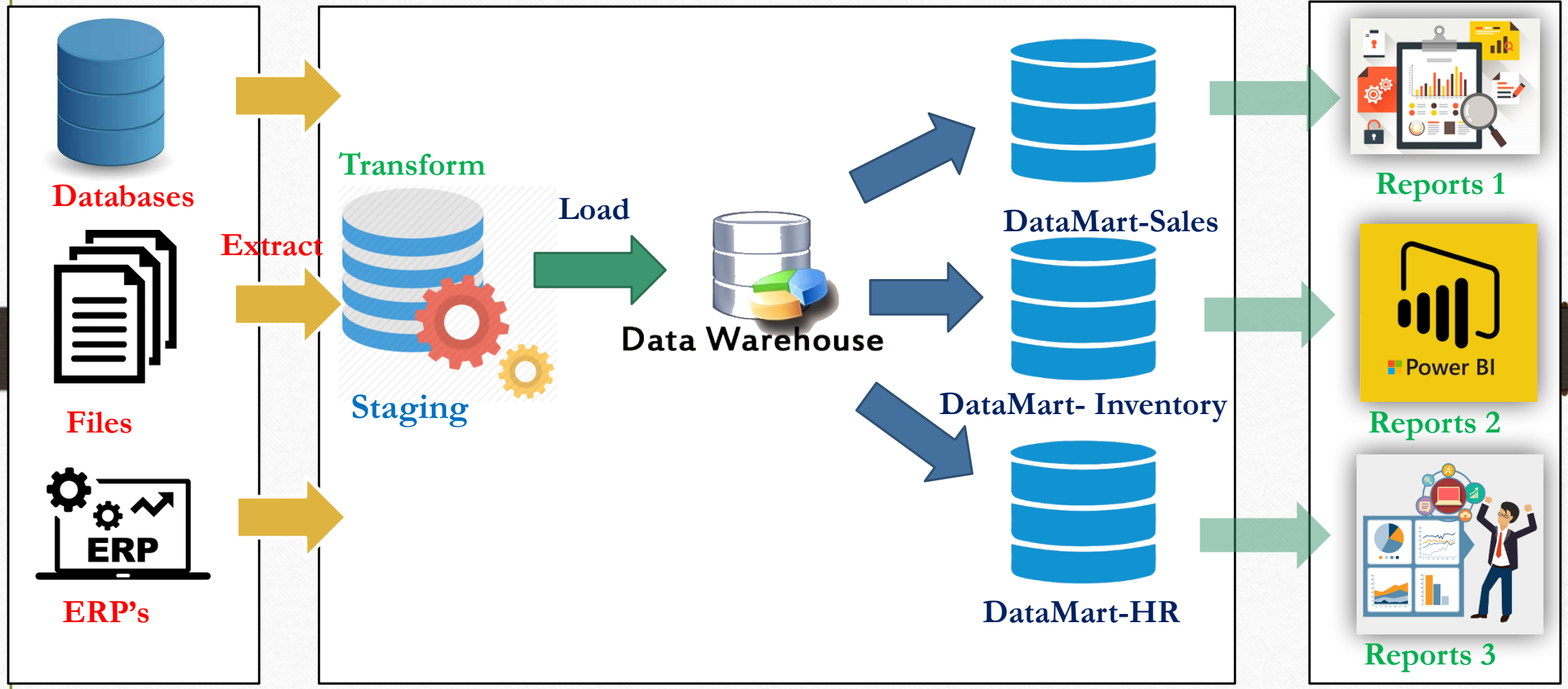
OLTP	OLAP
♦ Current data	♦ Current data as well as history.
♦ Used to support transaction processing	♦ Used to support the business interests
♦ Clerical data processing tasks	♦ Decision support tasks
♦ Simple and known queries	♦ Ad hoc, complex, and iterative queries which access million records and perform a lot of joins and aggregates
♦ A few tables involved and unlikely to be scanned	♦ Multiple tables involved and likely to be scanned
♦ Small foundset	♦ Large foundset
♦ Short transactions	♦ Long transactions
♦ Update/Select	♦ Select (Read only)
♦ Real time update	♦ Batch update
♦ Unique index	♦ Multiple index
♦ Known access path	♦ Do not know access path until users start asking queries
♦ Detail row retrieval	♦ Aggregation and group by
♦ High selectivity queries	♦ Low selectivity queries
♦ Low I/O and processing	♦ High I/O and processing
♦ Response time does not depend on database size	♦ Response time depends on database size
♦ Data model: entity relational	♦ Data model: multidimensional

A Concept Hierarchy



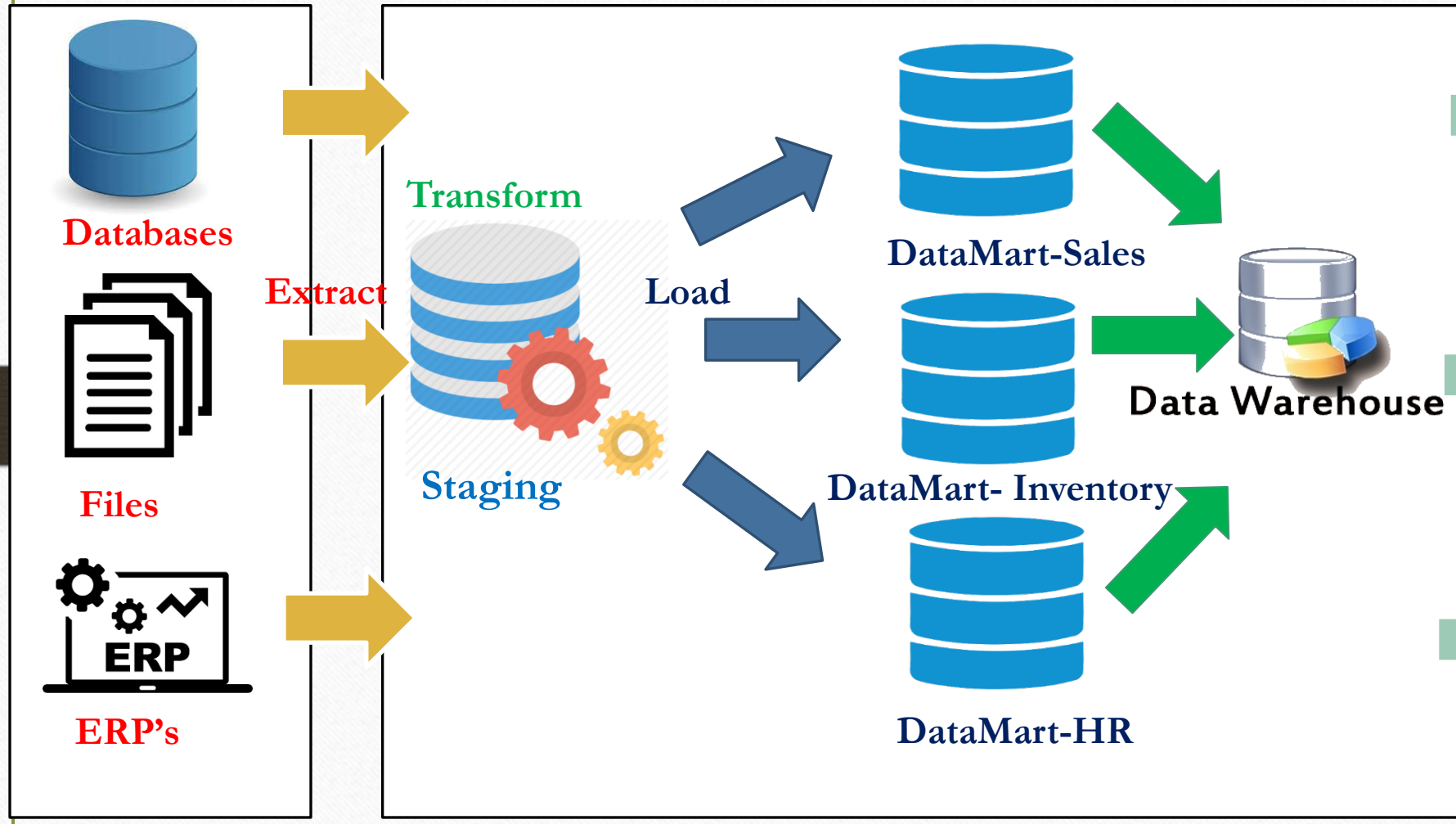
Top-down Approach

YouTube: Techlake

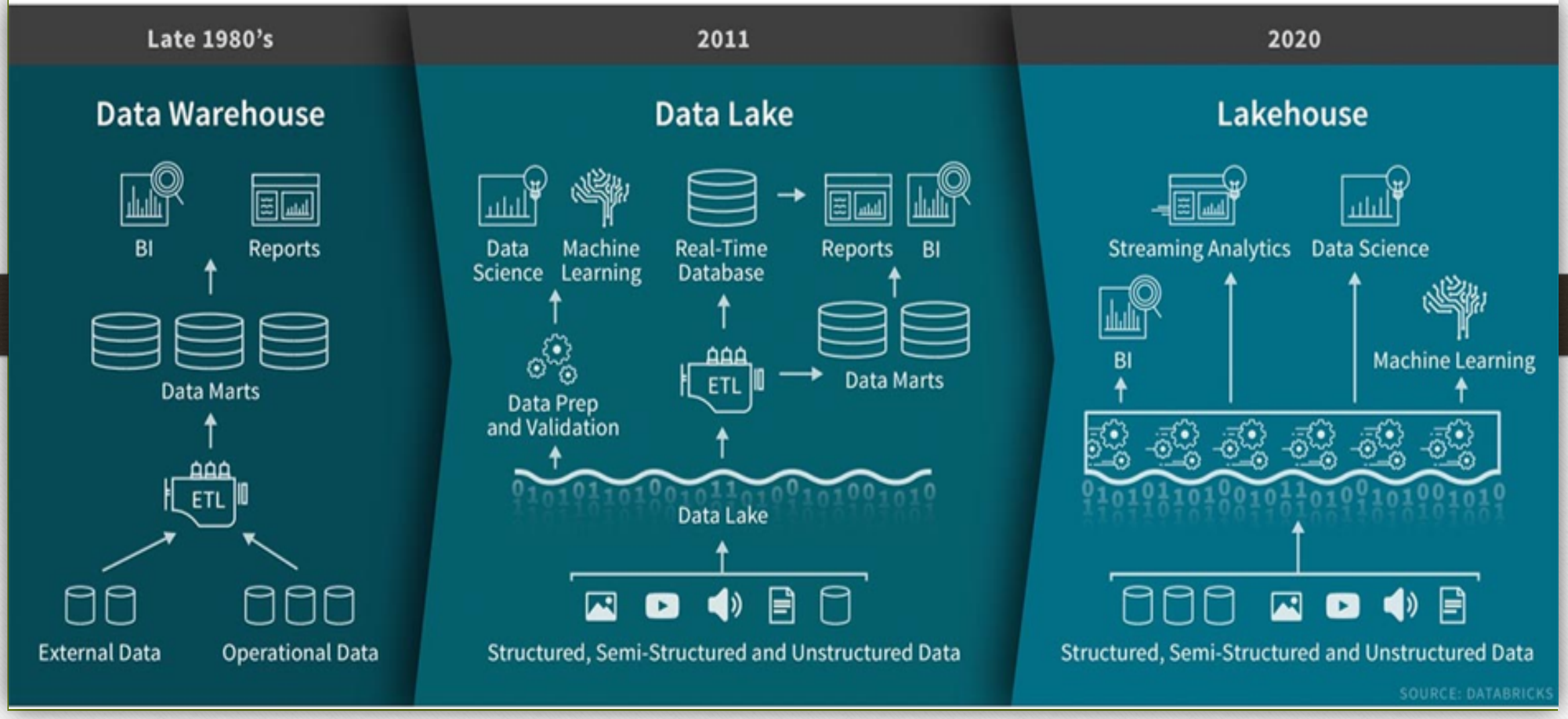


Bottom-up approach

YouTube: Techlake

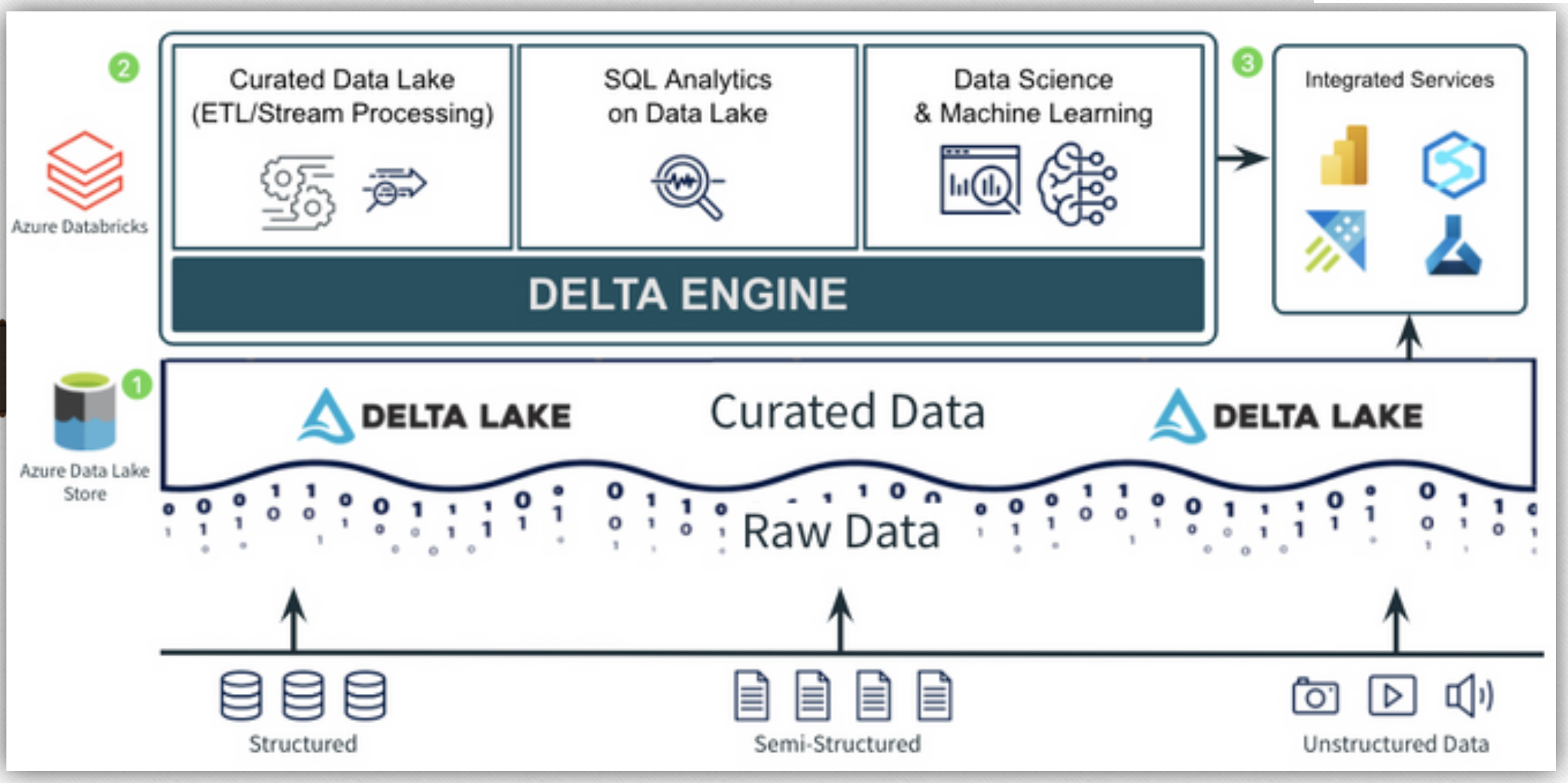


Data Warehouse in Traditional and big data architecture



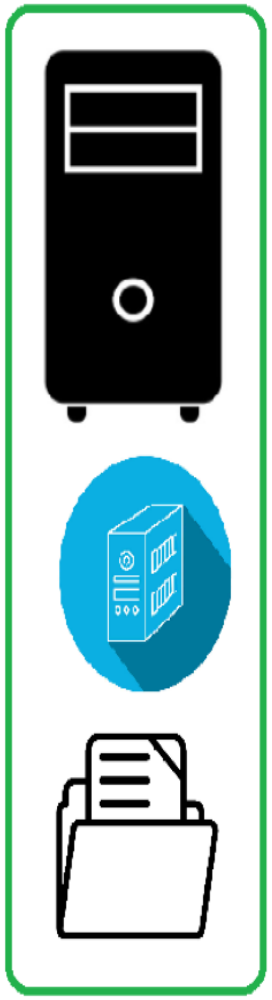
Azure Cloud Data Lake Architecture

YouTube: Techlake

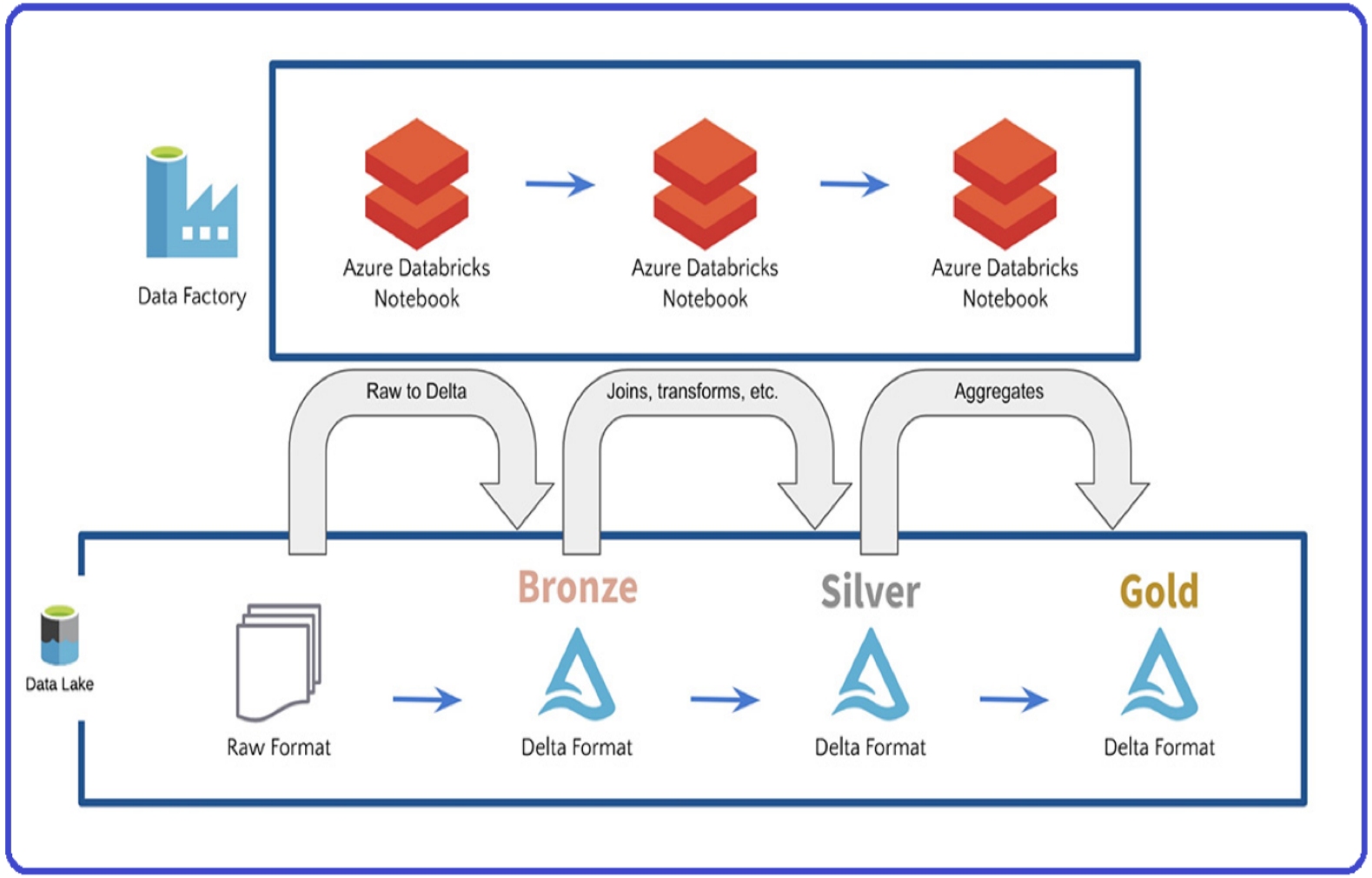


Multi-Hop Architecture (LAKE HOUSE)

YouTube: Techlake



ON PREMISES



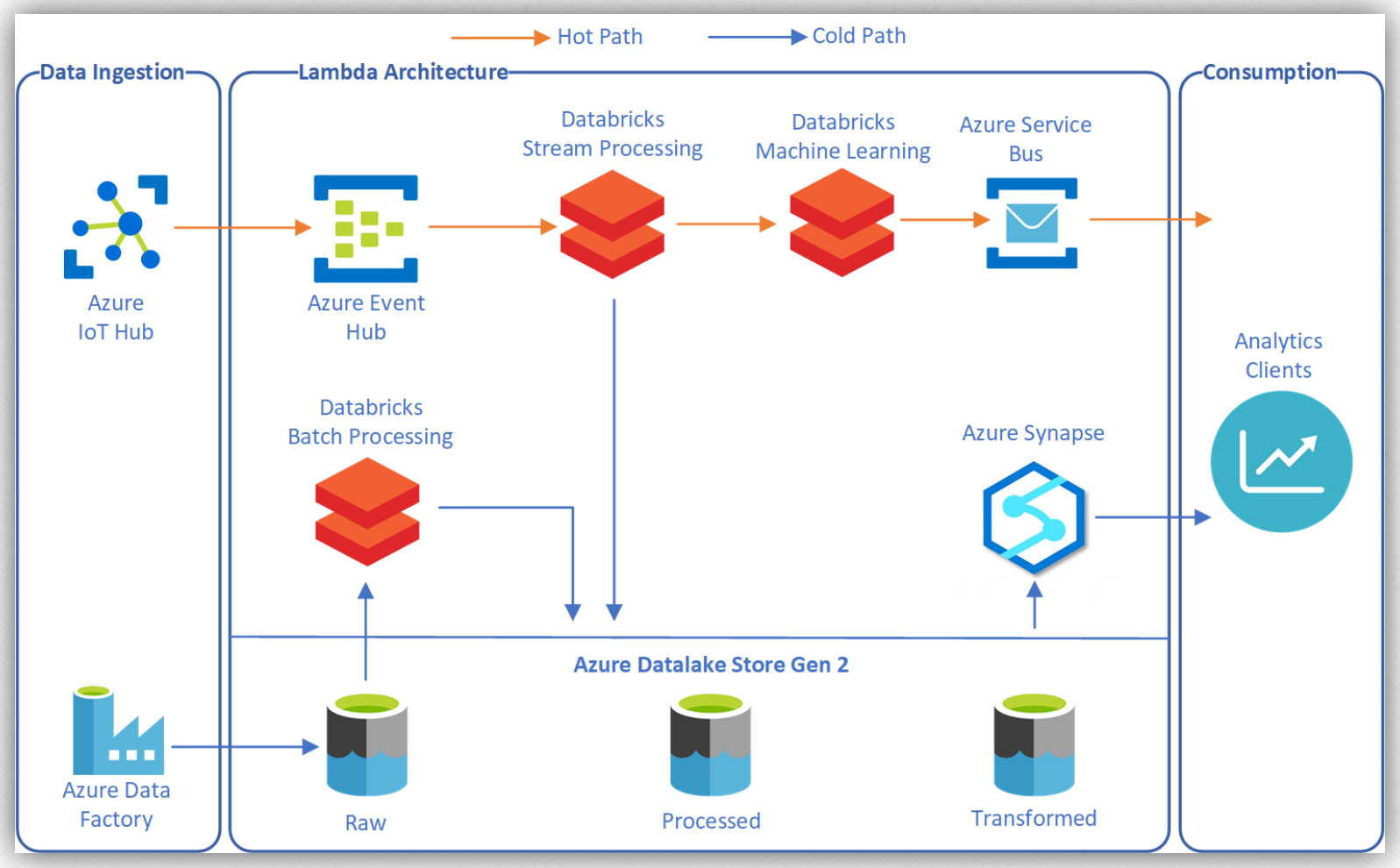
Cloud

Analytics

Lambda Architecture with Azure Databricks

YouTube: Techlake

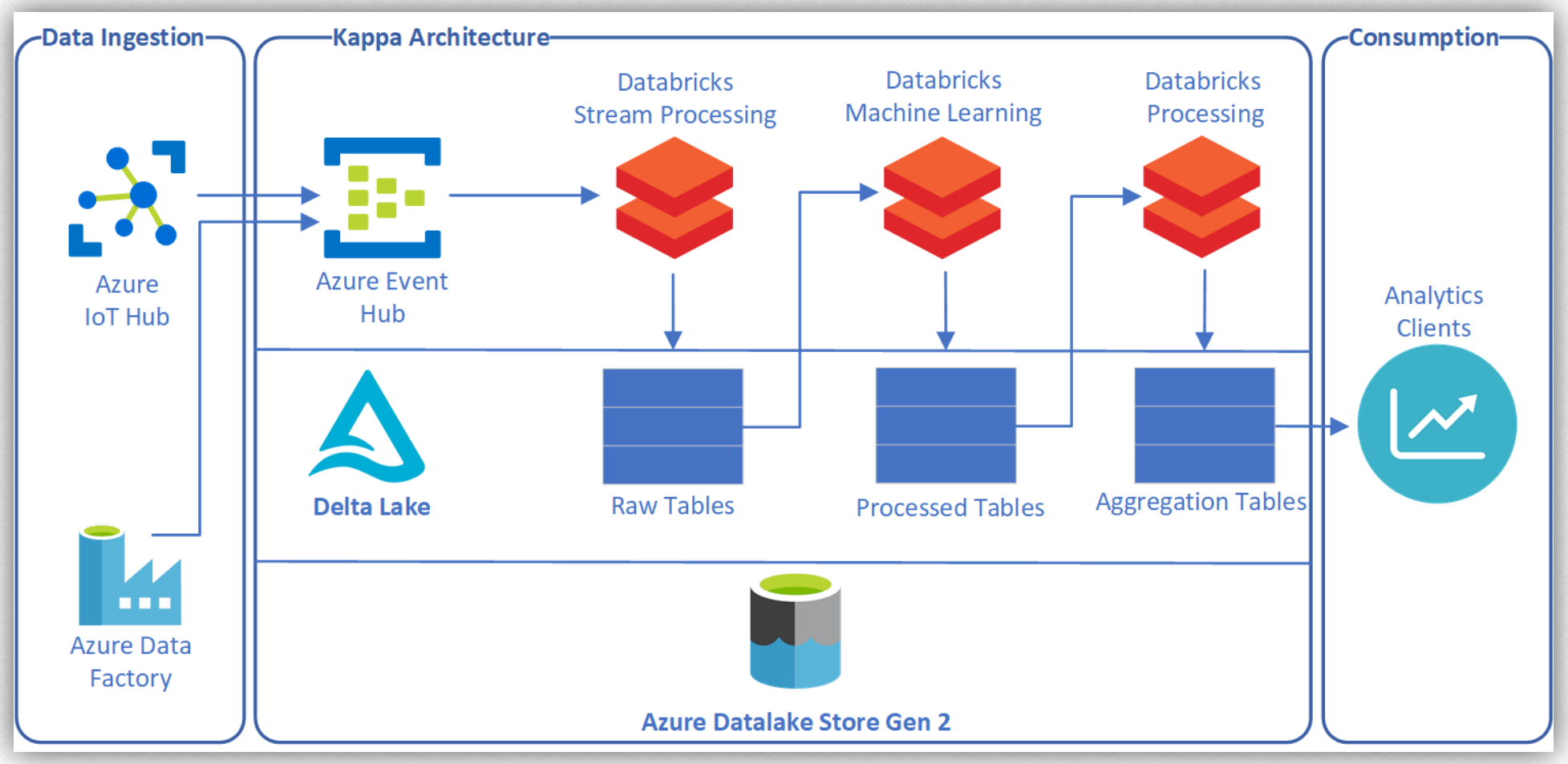
In proposed Lambda Architecture implementation, the Databricks is a main component as shown in the below diagram.



Kappa Architecture with Databricks

YouTube: Techlake

The Kappa Architecture suggests to remove the cold path from the Lambda Architecture and allow processing in near real-time.



YouTube: Techlake

Differences between Data Lake , Cloud Warehouses and On-Premises Databases

	Data Lake	Cloud Data Warehouse	On-Premises Databases
Unstructured data (schema-less data)	Yes	No	No
Semi-Structured data (Self-describing schema)	Yes	Yes	No
Structured Data (Relational)	Yes	Better	Better
Independently scale storage and compute	Yes	Yes	No
Schema-on-Read	Yes	Yes (semi-structured)	No
Schema-on-Write	No	Yes	Yes


All the Best