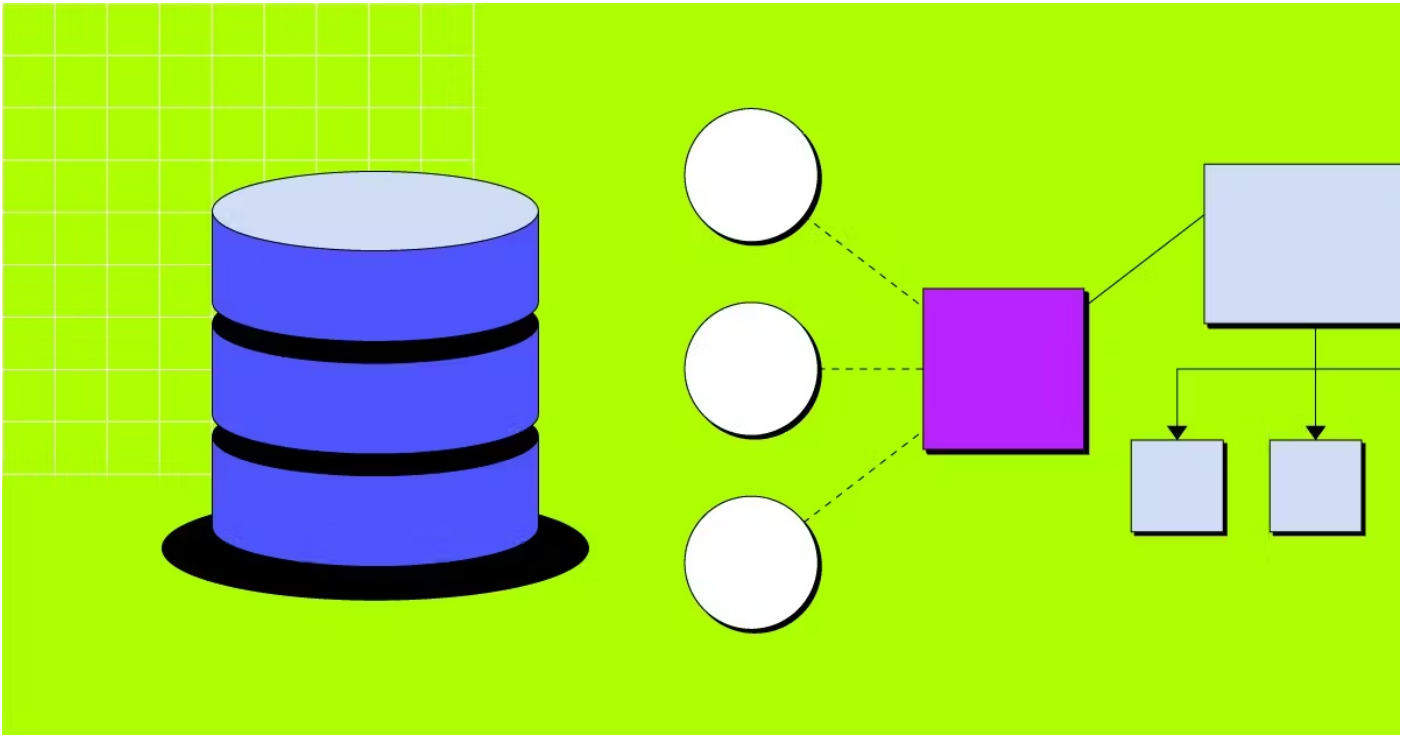
Data Science

The fundamentals of data warehouse architecture

by Sonny Rivera, Senior Analytics Evangelist at ThoughtSpot

Published Sep 19, 2023, last updated Sep 15, 2023



By the end of this two-part series, we will dive into what data warehouse architecture is and how to implement one for your organization. Part one will look at architectural layers and common data warehouse components, while part two dives into multi-tiered data warehouse architecture.

What is data warehouse architecture?

But first, it's essential to define exactly what a data warehouse is before diving into the data warehouse architecture. Bill Inmon is widely recognized as "the father of data warehousing" and the author of possibly the most important book about the subject, "Building the Data Warehouse." He defines a data warehouse as:

> **"Data warehouse architecture refers to a subject-oriented, integrated, time-variant, and non-volatile collection of data in support of management's decision-making process."**
>
> Bill Inmon

This definition, of less than 20 words, has an enormous impact on the world of data. Let me break it down for you with apologies to my hero, Bill Inmon.

Subject-oriented

The data in the warehouse is organized around subjects or topics rather than the applications or source systems that generate the data. For example, all sales data, regardless of where it comes from, is logically organized and grouped in the data warehouse. Assembling data around subject areas allows for a unified view of the subject—in our case, sales—rather than disparate views from each system.

Integrated

The data from each source system (e.g. CRM, ERP, Behavioral Data, or e-commerce platforms) is brought together and made consistent in the data warehouse. For instance, if one system uses region names like "North Carolina" and another uses abbreviations such as "NC," an integrated data warehouse would reconcile the region names to produce a consistent coding system.

Time-variant

s historical reporting. It's not just a snapshot of the current moment but a timeline of data that

Non-volatile

Data written into the warehouse doesn't ever get overwritten or deleted, ensuring the stability and reliability of the data, which is crucial for trustworthy analysis. Data can be added to the warehouse, but existing data typically remains unchanged.

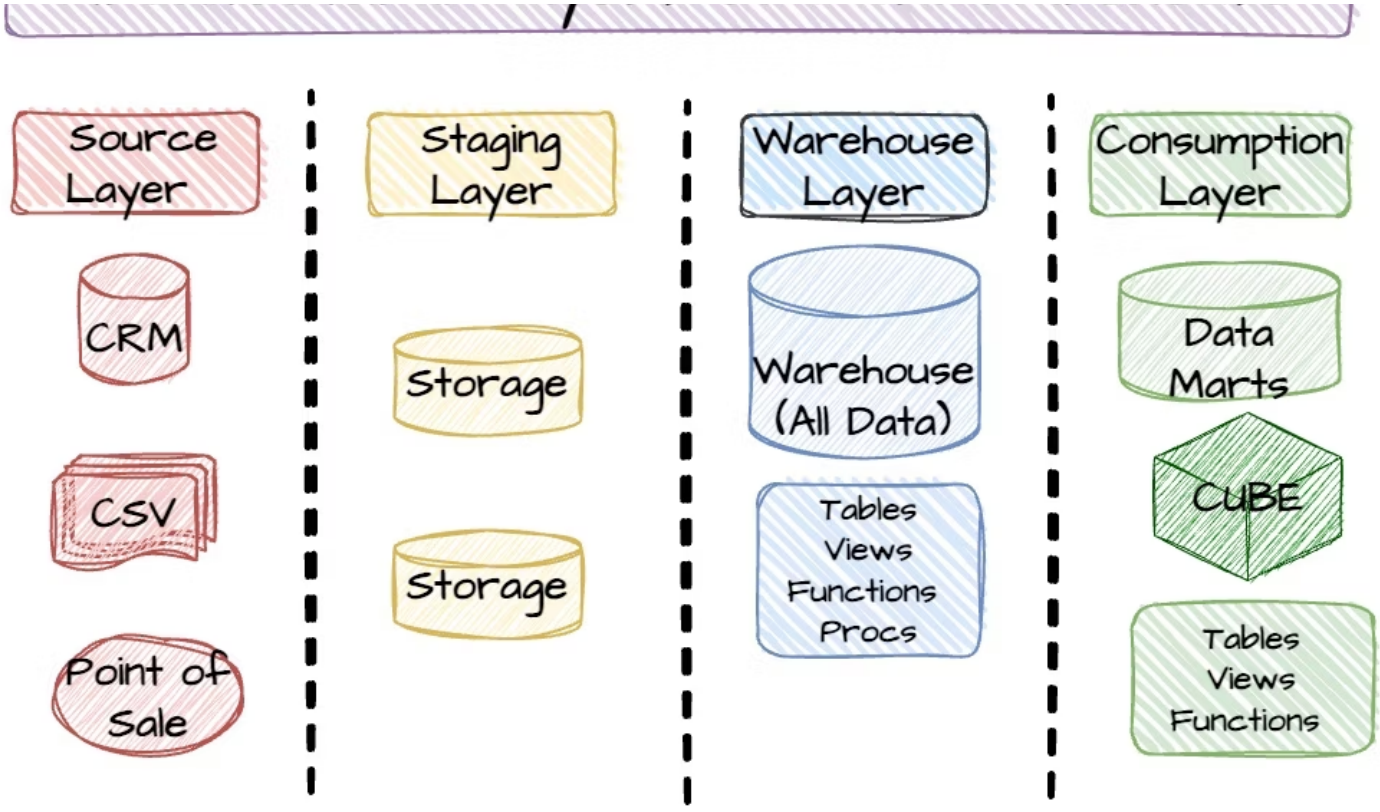
The role of the data warehouse architecture

The purpose of these core characteristics is to enable a fact-driven, decision-making process that can be utilized by everyone in the organization, especially as organizations look to take advantage of self-service analytics.

Data warehouse architecture is an intentional design of data services and subsystems that consolidates disparate data sources into a single repository for business intelligence (BI), AI/ML, and analysis. The architecture itself is a set of logical services that makes up the backbone of a data warehouse system, offering a structured and coherent way of storing, managing, and retrieving massive amounts of data.

Even if you haven't purposefully planned your data warehouse architecture, you still have one—just potentially a poorly designed and inefficient one.

Layers of data warehouse architecture



Data warehouses have several functional layers, each with specific capabilities. The most common data warehouse architecture layers are the source, staging, warehouse, and consumption.

Source layer

The logical layer of all systems of record (SOR) that feed data into the warehouse. They could include point-of-sale, marketing automation, CRM, or ERP systems. Each source SOR has a specific data format and may require a different data capture method based on that data format.

Staging layer

A landing area for data from the source SOR. A data staging best practice is to ingest data from the SOR without applying business logic or transformations. It's also critical to ensure that staging data is not used in production data analysis; data in the staging area has yet to be cleansed, standardized, modeled, governed, and verified.

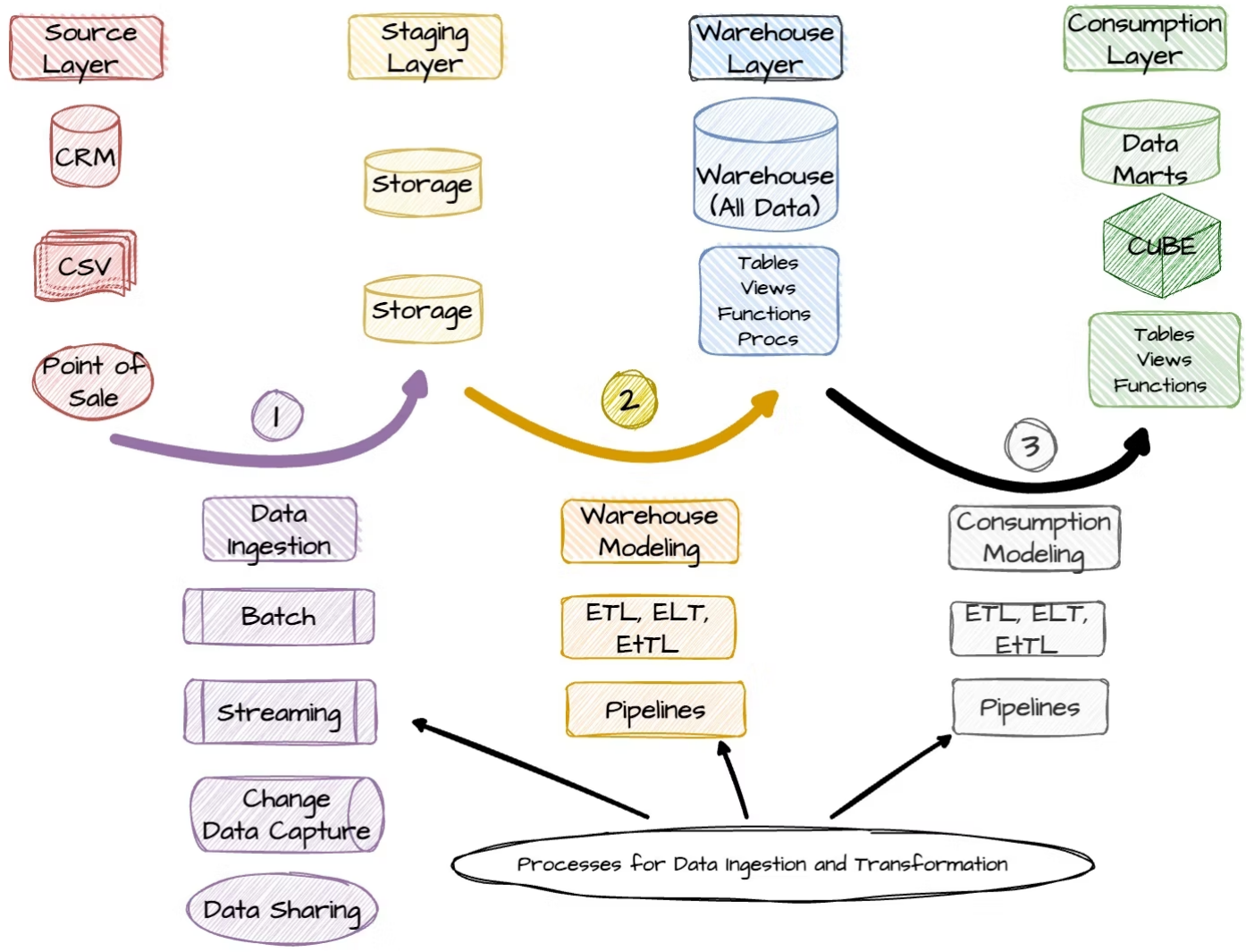
Warehouse layer

The layer where all of the data is stored. The warehouse data is now subject-oriented, integrated, procedures, and functions needed to access the warehouse-modeled data.

Also known as the analytics layer, is where you model data for consumption using analytics tools like ThoughtSpot, data analysts, data scientists, and business users.

Data ingestion and transformation

How does data travel from the source to the consumption layer? Three key processes are typically needed to move, clean, and transform data into warehouse and consumption layers. Let's take a look at them.



Data ingestion is the process of ingesting data from each system of record or even from files. The most common ingestion techniques can be broadly categorized into several prominent methods, such as

understood. However, it has some drawbacks, including data latency, lack of scheduling flexibility, intense CPU and disk usage, and integration difficulties.

Data streaming

This is a method of processing data in real-time or near-real-time as it is generated or received. Unlike batch processing, which handles data in large chunks at specified intervals, streaming continuously ingests, processes, and analyzes data, allowing immediate insights and actions based on the incoming data flow. This approach is especially beneficial for applications requiring timely responses, such as real-time analytics, monitoring systems, and fraud detection.

Change data capture (CDC)

CDC typically uses the database transaction logs of the SOR to capture any data that has been created, modified, or deleted. This approach ensures that only the changed data is ingested. The CDC process is typically executed periodically, such as every hour or every day. It's great for syncing operational data stores (ODS) and staging data from transactional systems of record.

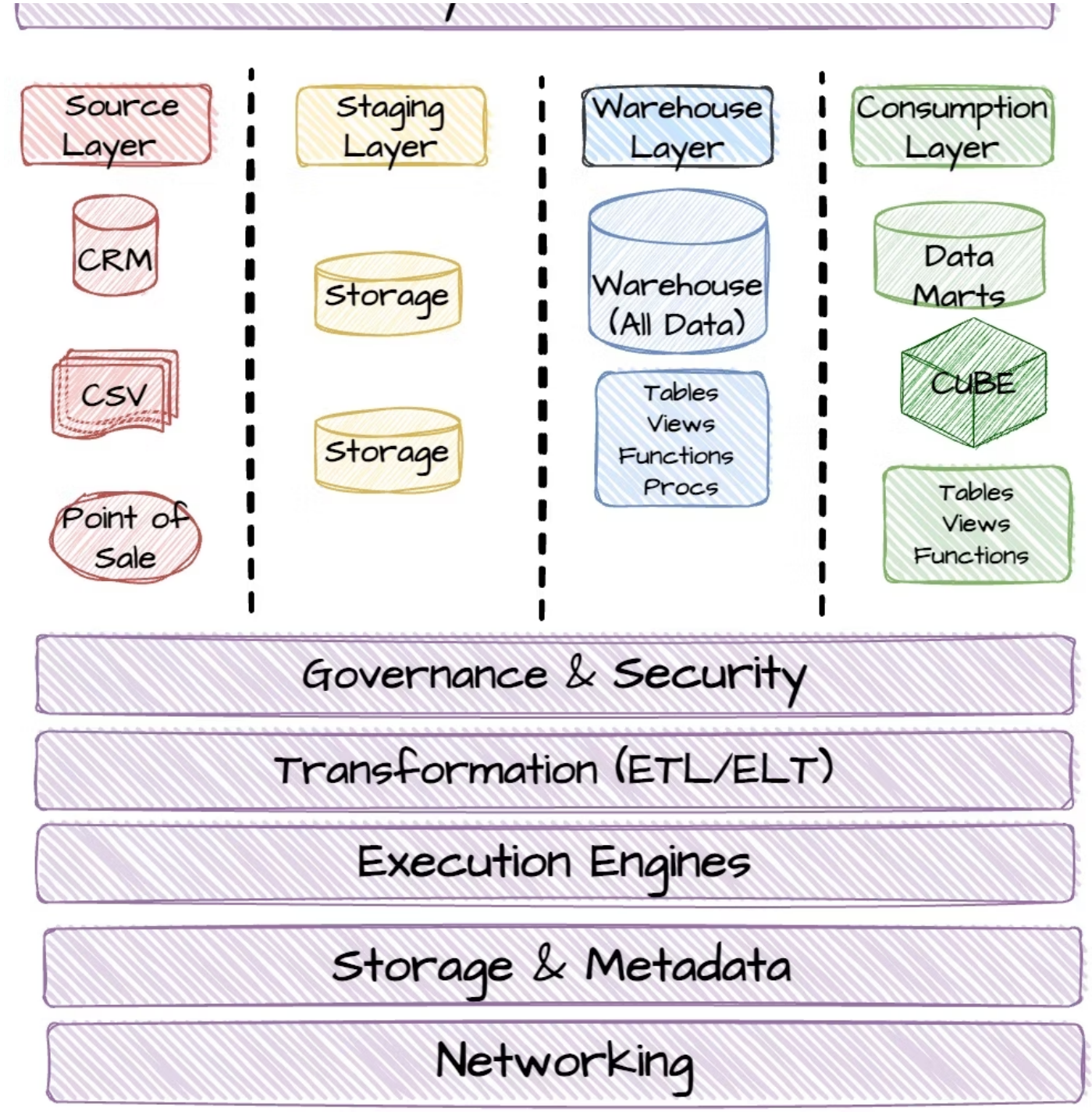
Data sharing

Unlike traditional methods of data ingestion, where data is duplicated or moved, data sharing from cloud data providers like Snowflake or Databricks doesn't require data duplication. This method provides real-time data sharing and is often used by organizations implementing Data Mesh.

Pro Tip: Each ingestion method has advantages and disadvantages, so evaluate the data source and business requirements to ensure you get the greatest value from the data.

Data warehouse architecture components

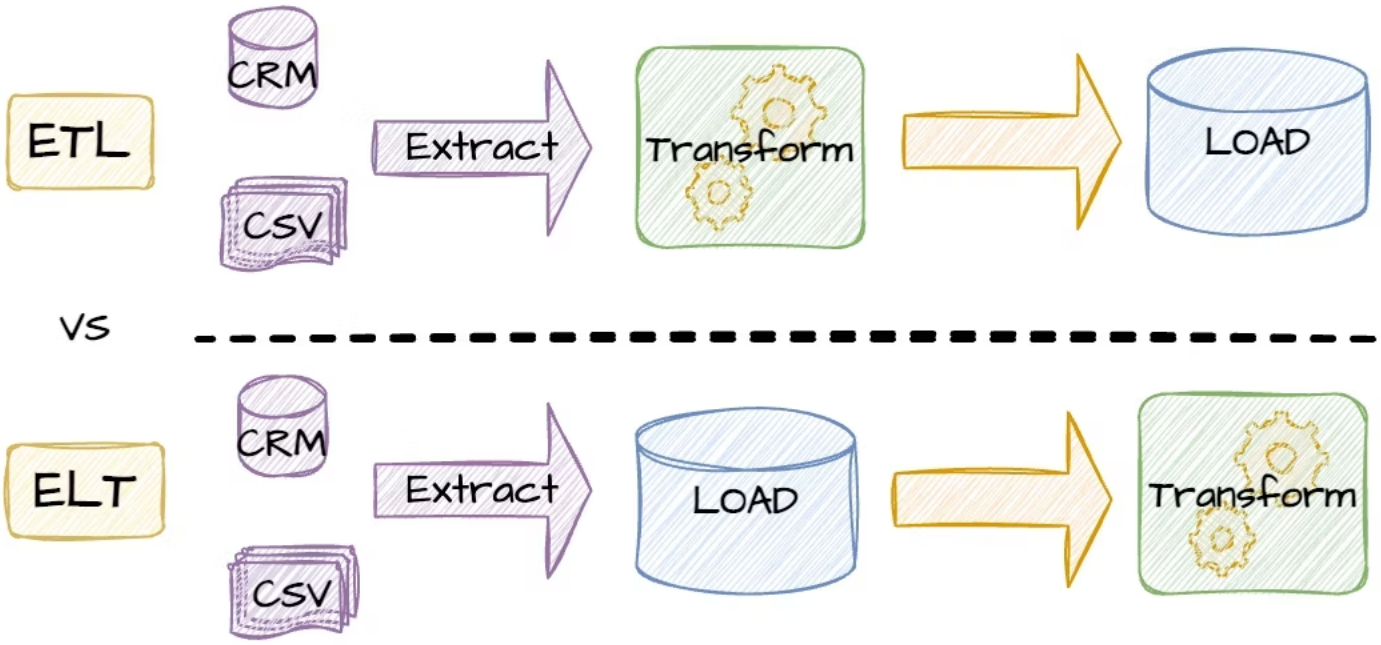
Every data warehouse architecture consists of architectural layers, processes for data ingestion, and transformations. Those layers and processes also rely on shared components of the data architecture.



ETL and ELT

Extract, Transform, and Load (ETL) is the process in which data is moved or transformed from source to target. As the name states, data is extracted from the source, transformed by applying hard and soft business rules and loaded into the destination. The ETL approach often suffers from poor performance and scalability issues because the process moves data from the

This approach extracts data from the source, is loaded into a staging area, and then transformed (business rules applied). This approach is prevalent in most modern cloud data platforms, like Snowflake and Databricks, that can ‘push down’ the processing to the data platform and offer virtually unlimited scale and performance, but at a cost. ELT is the preferred approach by modern data warehousing methodologies like Data Vault 2.0 (DV2).



It’s worth noting a common variant of ELT called EtTL. In this variant, the ‘lowercase t’ stands for small transformations, such as fixing data types but not applying business rules or re-shaping the data structures. DV2 advocates this approach, calling these small transformations “hard rules” and business transformations “soft rules.”

Metadata services

Metadata refers to the "data about data." It provides context, meaning, and information about data, helping users understand the source, lineage, history, and data characteristics. It also enables tools for data engineering, generative AI, large language models, and business intelligence tools to quickly transform raw data into modeled data. Common use cases for metadata include data modeling, data lineage, documentation, catalogs, knowledge bases, and observability.

Execution engines

Execution engines, sometimes called query engines, are the computing resources for data

cost-effective. Modern execution engines serve the needs of data engineers, analytics engineers, and the analytical needs of businesses.

Modern cloud data platforms have created proprietary, highly performant, and scalable execution engines that support data transformations and machine learning with R, Python, and SQL, as well as their proprietary query languages, like DAX.

Storage services

Storage services are at the core of a data warehouse, where all the consolidated data resides, including raw, staged, and modeled data. Storage services are responsible for storage, data partitioning, compression, replication, back and recovery, life cycle management, and data integrity.

Whether in the context of cloud-based data platforms like Amazon Redshift, Google BigQuery, or Snowflake, these storage services are managed, scalable, and optimized to offer maximum performance with minimum manual intervention at a low cost. They are essential to the efficient and well-functioning of a data warehouse.

Governance and security

Governance and security of a data warehouse are paramount due to the sensitive and critical nature of the consolidated data. These services include authentication, authorization, role-based access controls (RBAC), encryptions, network security, data classification, and even data masking. Don't worry; most organizations have dedicated experts for ensuring your data is well-governed and protected.

What's next?

In this first post, you have seen the common layers and shared components of a well-formed data warehouse architecture. But, that's not all there is in a data warehouse architecture. Part two of this series will cover building multi-tiered warehouse architecture on the data warehousing foundations described in this post.

With this understanding of the architectural layers and shared components, you now have the ability to start evaluating your current architecture or plan out your upcoming architecture. You can start evaluating how these components align with your architecture and requirements. Take

- Are you staging your data? If so, where?
- Are you doing ETL? Will ELT be a better solution going forward?
- Can you better leverage ELT and push-down work loads to speed up development and increase performance and scalability?
- What types of execution engines are best for your requirements? SQL, Python, or something proprietary?
- How are these shared components, such as governance and security, utilized across all of the layers?

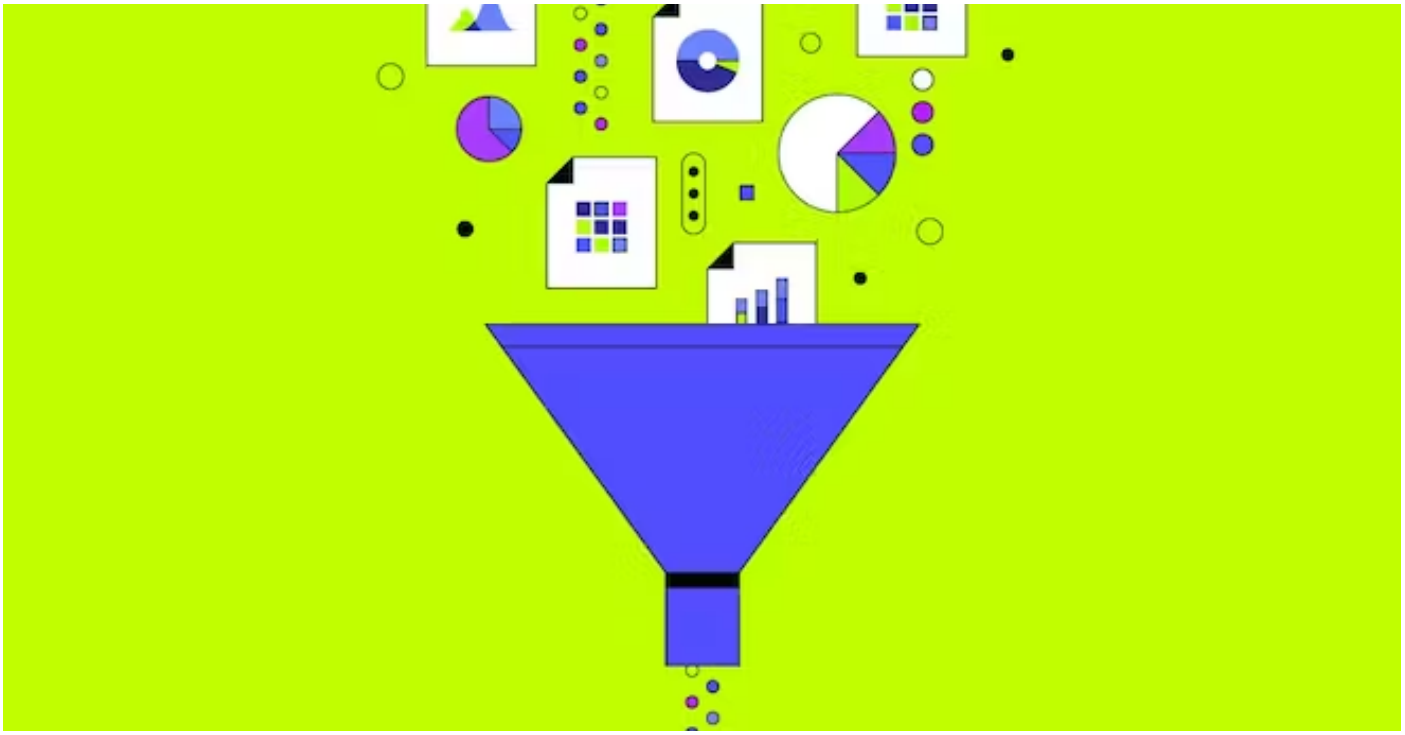
Stay tuned for part two on multi-tiered data warehouse architectures, where you'll see which tiers are best for your organization, and how those tiers promote analytics consumption and self-service analytics.

Related articles

Data Science

Multi-tiered architectures for data warehouse

Read more



Data Science

Get demo

Data Science

7 best practices of data collection for analytics

Read more