LTIMindtree

WHITEPAPER

Data Vault 2.0

Modern Analytical Ecosystems

Executive Summary

As cloud technology continues to evolve at breakneck speed, organizations are no longer limited by traditional infrastructure constraints. The ability to **dynamically scale compute and storage** has transformed the data landscape—enabling not just faster processing, but **real-time, intelligent decision-making.** Businesses are now tapping into both **structured and unstructured data** to deliver smarter, faster, and more personalized client experiences.

However, this agility introduces a new imperative: the need for a **trusted, unified source of truth.** Without it, real-time decisions risk being fragmented, and even the most advanced machine learning models can falter due to inconsistent or incomplete data.

Simultaneously, the shift from rigid, waterfall-style development to **agile, iterative product cycles** demands a data architecture that is equally flexible and future-ready. Traditional modeling approaches like **Kimball,** while foundational, often struggle to keep pace with the demands of **modern analytical ecosystems.**

This is where **Data Vault 2.0** emerges as a catalyst. Purpose-built for the cloud era, it offers a **modular, scalable, and auditable framework** that aligns seamlessly with cloud-native platforms like **Snowflake and Databricks.** Despite its potential, adoption has been cautious—largely due to perceived complexity and operational overhead.

This whitepaper aims to **break down those barriers.** It presents a real-world case study of how I successfully implemented **Data Vault 2.0** for a leading **North American logistics client** that underwent a digital transformation from a legacy, batch-based, on-premise warehouse to a **real-time, cloud-native data platform** on Snowflake.

The fundamental difference between Data Vault and Data Vault 2.0 lies in how they handle record identification. While the original Data Vault approach relies on surrogate or system-generated keys to uniquely identify records, Data Vault 2.0 introduces the use of hash keys derived from natural business keys. This shift enhances consistency, scalability, and performance, especially in distributed and big data environments.

In this document, the term object is used interchangeably as either a table or a view. A table is a physical object, whereas a view is a logical object.

In the interest of the client confidentiality, the North American Logistics client is referred as "X".

Business Context and Drivers

In an era defined by speed, scale, and seamless experiences, logistics innovators like **X** are reengineering their digital core to stay ahead of the curve. At the center of this transformation is a strategic imperative: **real-time analytics** that serve as an engine powering intelligent, responsive, and customer-centric operations.

To lead in a volatile and competitive market, X needed to unlock real-time intelligence to:

Dynamically optimized pricing based on live demand, competitive landscape, and vehicle availability across both internal fleets and trusted vendor networks.

React instantly to market fluctuations, enabling agile decision-making and operational resilience.

React instantly to market fluctuations, enabling agile decision-making and operational resilience.

Continuously evaluate offer performance in real-time, adapting strategies to shifting market dynamics.

Smartly route orders to the most efficient fulfillment centers, maximizing speed and service quality.

X's operational triad, comprising **offer, order, and transportation**, was orchestrated through a complex ecosystem of platforms. Orders originate from systems like **Siebel and Mercury Gate**, depending on whether they are direct or third-party. Transportation was split between X's internal fleet (managed via **OTM Assets**) and external logistics partners via **TMS**).

As X migrated from legacy on-premises systems to **cloud-native platforms**, including **Oracle Fusion Cloud CX** for centralized order management and a future-ready unified transportation system, the need for a **modular, scalable, and future-proof data architecture** became non-negotiable. This architecture had to support:

Parallel system operations- allowing legacy and modern platforms to coexist during transition.

Incremental onboarding/offboarding- enabling seamless migration of orders and data without disruption.

Real-time synchronization- ensuring consistency across systems even when orders were split between old and new environments.

Beyond integration, X was laying the foundation for **enterprise-wide data unification**. A key aspect of this was the creation of **enterprise data objects**—such as the **enterprise person object, enterprise offer object,** and **enterprise order object**. These objects would consolidate fragmented roles and records into a **single source of truth**. They are designed to scale, supporting everything from operational dashboards to predictive analytics and AI-driven insights.

Business Context and Drivers

In an era defined by speed, scale, and seamless experiences, logistics innovators like **X** are reengineering their digital core to stay ahead of the curve. At the center of this transformation is a strategic imperative: **real-time analytics** that serve as an engine powering intelligent, responsive, and customer-centric operations.

To lead in a volatile and competitive market, X needed to unlock real-time intelligence to:

01 **Bill Inmon:** Top-Down Approach

Philosophy	Inmon, known as the "Father of Data Warehousing," advocates building a centralized, normalized enterprise data warehouse (EDW) first.
Structure	Data is stored in 3rd normal form (3NF) to reduce redundancy and ensure consistency.
Use case	Ideal for organizations prioritizing data integrity, governance, and a single version of the truth.
Pros	Strong data quality, scalable for enterprise-wide analytics.
Cons	Slower to deliver business value initially due to complexity and time to build.

LTIMindtree

02 Ralph Kimball: Bottom-Up Approach

Philosophy	Kimball promotes building data marts first, focused on specific business processes, and integrating them into a dimensional data warehouse.
Structure	Uses star schemas for ease of use and performance.
Use case	Best for quick wins and business-driven analytics.
Pros	Faster time-to-value, user-friendly for reporting and BI tools.
Cons	Can lead to data silos and integration challenges over time.

03 Data Vault 2.0: Hybrid and Agile Approach

Philosophy	Created by Dan Linstedt, Data Vault 2.0 is designed for agility, scalability, and auditability in modern, complex data environments.
Structure	Separates data into hubs (business keys), links (relationships), and satellites (context/history/fact).
Use case	Ideal for cloud-native, real-time, and big data ecosystems with evolving requirements and scalable for enterprise-wide analytics.
Pros	Supports parallel development, historical tracking, and flexible integration of new sources.
Cons	More complex modelling.

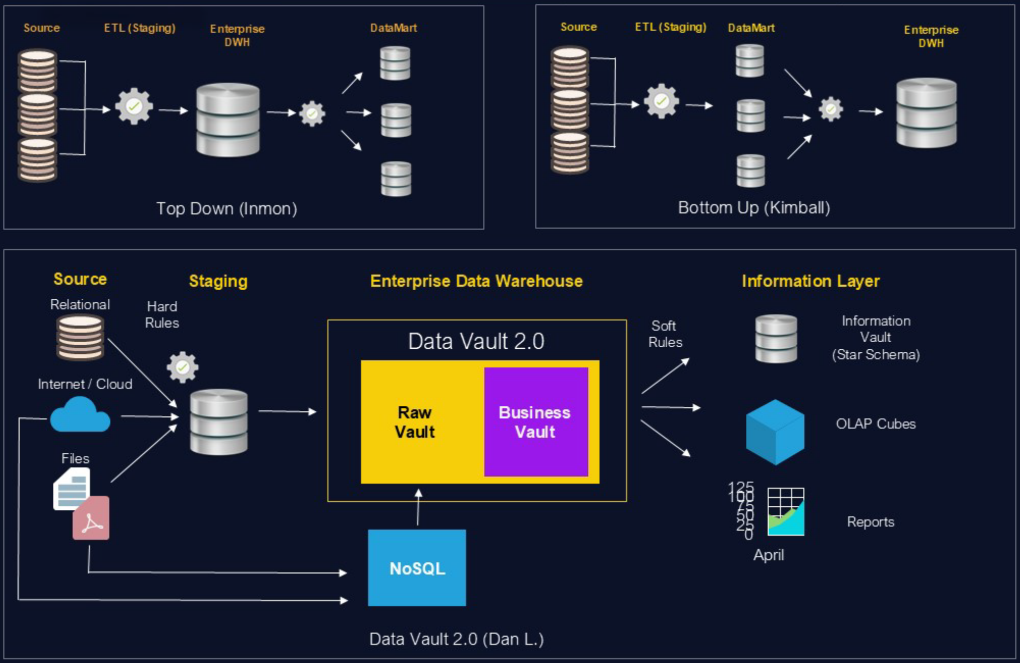


Figure 1: Architecture of Data Vault 2.0

Why Data Vault 2.0?

Unlocking Enterprise Agility with Data Vault 2.0

The Data Vault 2.0 architecture is more than just a modelling technique—it's a strategic enabler for enterprise agility, scalability, and historical traceability. At its core, the triad of hubs, links, and satellites forms a resilient foundation that supports dynamic business needs without compromising data integrity or requiring disruptive redesigns.

Let's explore how each component contributes to this transformative capability.

Embracing Change: Cardinality and Granularity Without Redesign

Traditional data models often falter when business rules evolve—requiring costly redesigns and retesting. Data Vault 2.0, however, is inherently adaptive:

01

Cardinality shifts

Whether a business rule changes from "one person can place many orders" to "multiple people can contribute to the same order," the Data Vault 2.0 model absorbs this shift seamlessly. No structural changes are needed; just a new entry in the link object.

02

Granularity adjustments

In case you need to track data at a finer or coarser level, just introduce a new satellite and associate it to the existing link. The existing model remains untouched, preserving stability while enabling flexibility.

Parallel Processing for Real-Time Analytics

Data Vault 2.0's use of hashed natural keys enables decoupled, parallel loading of contextual (dimensional) and transactional (fact) data. Unlike traditional models that rely on surrogate keys and enforce strict load dependencies, Data Vault 2.0 allows:

01 Independent processing of facts and dimensions as fact is not dependent on dimension for surrogate key

02 Faster delivery of analytical insights

Enterprise-First Design Philosophy

Unlike Kimball's "Mart-first" approach, Data Vault 2.0 aligns more closely with Inmon's "Enterprise-first" philosophy without inheriting the stringent normalization constraint. It builds enterprise-wide objects before downstream marts, ensuring a single version of the truth and enabling consistent, scalable analytics across the organization.

Built-In Historization and Incremental Loading

Every hub, link, and satellite in a Data Vault 2.0 model includes a load date, inherently supporting historical tracking. This design:

01 Facilitates SCD Type 2: As the load date is maintained in the hub, link, and satellite entities, records are generally not deleted from these structures unless a specific data retention policy is in place.

02 Supports incremental loads and reprocessing by injecting keys into hubs.

Natural Key Advantage in Big Data Ecosystems

In big data environments, where semi-structured data resides in a staging layer, the reliance on natural keys becomes a strategic asset. These keys allow:

- Seamless integration of staging data at any processing stage
- Consistent referencing across structured and semi-structured domains
- Enhanced traceability and auditability
- Centralized encryption

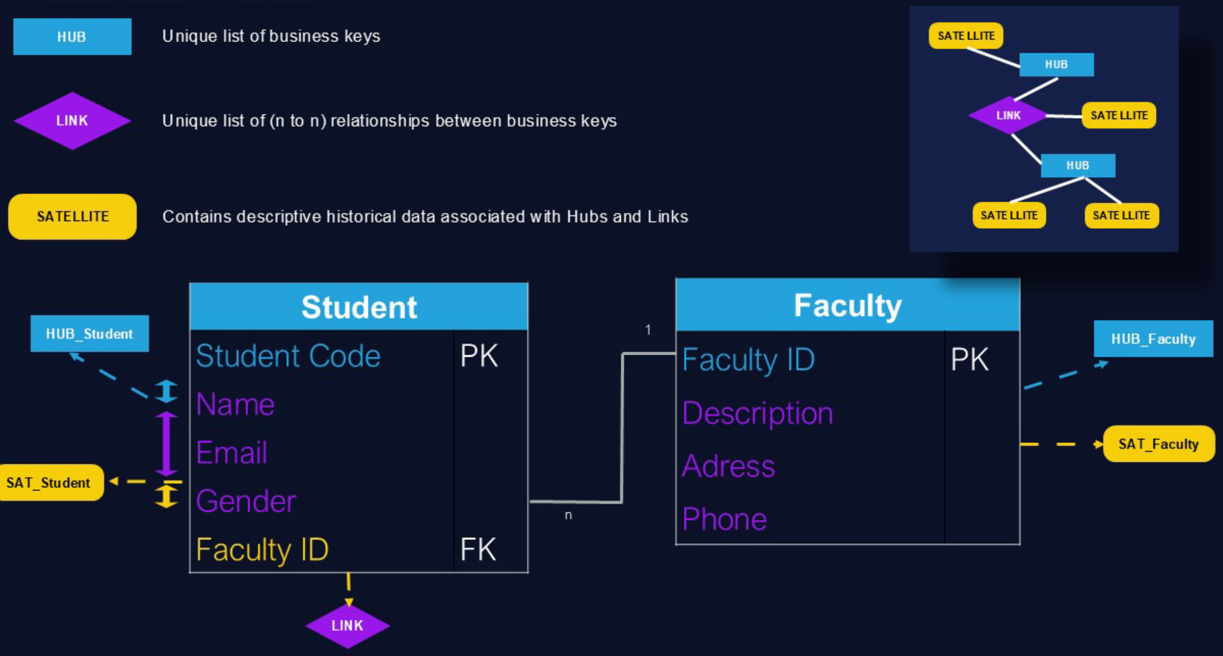


Figure 2: Data Vault 2.0 Basics

Use Case

1. Client Overview

The client X was a leading logistics service provider in North America, operating a hybrid transportation model that leverages both an owned fleet of trucks and third-party vendors. Their business was centered around three core pillars: Offer management, order fulfillment, and shipment execution.

2. Current Operational Landscape

The client initiated transportation offers based on incoming leads. Upon client acceptance, orders were placed through systems such as Siebel and Mercury Gate. Shipment execution was managed via Oracle Transportation Management (OTM) applications:

- OTM assets: Handled shipments using in-house fleet
- OTM non-assets: Managed third-party logistics providers

In addition to these core systems, the organization relied on approximately 24 auxiliary applications, including human capital management (HCM), marketing, and lease management.

3. Strategic Vision and Roadmap

The client was embarking on a modernization journey to consolidate its fragmented application ecosystem into a unified platform. The roadmap included:

- Phase 1: Integrating legacy applications with the new enterprise data warehouse (offer management, Siebel, Mercury Gate, OTM assets, and OTM non-assets).

- Phase 2: Enabling seamless integration between Mastermind once the legacy application moved to Mastermind.

- Phase 3: Decommissioning legacy application connections from the warehouse.

4. Warehouse Modernization Goals

To support this transformation, the client aimed to modernize its data warehouse with the following capabilities:

- **Real-time analytics:** Empower business teams to make timely, data-driven decisions and offer competitive pricing.
- **Enterprise data objects (EDOs):** Establish a single source of truth for key entities such as person, order, and offer.
- **Data security architecture:** Implement robust data access controls at the staging data layer, ensuring consistent security across all downstream systems and tools. Avoid data duplication to maintain integrity and reduce risk.
- **Scalable KPI integration:** Allow for the dynamic addition of new KPIs and contextual data with minimal or no rework.
- **Flexible data reloading:** Enable history to reload without increasing compute resources or causing downtime.
- **Historization:** The requirement was to have SCD 2 without an end date.
- **Accelerating Time-to-Value Through Phased Warehouse Deployment.**
  The client was committed to delivering rapid business impact through a phased rollout of its modernized data warehouse. Rather than waiting for full implementation, the organization sought to unlock early value by enabling sponsors and stakeholders through a phased approach so they could experience the benefits incrementally.

This approach ensured:

- **Real-time analytics:** Sponsors could begin leveraging enhanced analytics and insights from initial phases, fostering confidence and momentum.
- **Enterprise data objects (EDOs):** Business units could gradually adapt to new capabilities, reducing change fatigue and improving user engagement.
- **Data security architecture:** Early releases allowed for real-time feedback, enabling course correction and optimization throughout the implementation lifecycle.
- **Scalable KPI integration:** Smaller, manageable releases helped identify and resolve issues early, minimizing disruption and ensuring smoother transitions.

Challenges

Architecture evolution

For transition from the legacy application to the advanced Mastermind platform, a strategic integration into the warehouse ecosystem was essential. This migration was not just a technical shift—it was a transformation in business logic and operational flow. For instance, while the legacy Siebel application supported order versioning, Mastermind introduced a streamlined model where orders are governed by rolled-out offers, eliminating the need for versioning.

During this transitional phase, both systems could operate in parallel. This dual-system approach ensured continuity, allowing existing orders to flow through the legacy system while new orders were onboarded via Mastermind. Eventually, as the new system stabilized and fully absorbed operational demand, the legacy platform was gracefully decommissioned—ensuring a smooth, disruption-free offboarding process.

Enterprise object evolution

Onboarding of applications into the data warehouse was planned in phases, and the availability of enterprise data objects (EDOs) was also staggered. To support this phased approach, the data architecture had to enable incremental development of these enterprise objects. Each EDO integrated data from multiple source systems, which could differ in terms of available information and unique identifiers.

For example, the "person" EDO needed to consolidate employee data from both HCM and Siebel systems. In HCM, employee records are identified by person ID and may not include driver license information, as it is not mandatory. In contrast, Siebel contains data on drivers—who are also employees—but uses driver license as the unique identifier. Therefore, the person EDO must be designed to accommodate and unify employee details from both systems, despite differences in data structure and unique fields.

Metrics evolution

New decision matrices were planned for later phases, so the data warehouse needed to be designed with a flexible approach that supported the seamless addition of new metrics and contextual information—without requiring changes to existing objects. This was necessary to ensure scalability without rework.

Data architecture needed to support real-time analytics.

Phase-wise deployment:
Phase-wise deployment posed a challenge when integrating a new source system and incorporating its content into existing objects. This process often necessitates changes to object metadata and business rules, along with comprehensive end-to-end testing. Without a structured approach, it can lead to significant rework. The client needed an approach that avoided rework with phase-wise deployment.

Reprocessing:
The expectation was to implement a streamlined approach that enabled the reload of historical data without causing downtime or consuming additional compute resources.

Centralized encryption:
Sensitive data needed to be encrypted in one place not in multiple layers.

Approach

01 Scalable Business Intelligence

To enable a robust and scalable data integration framework, our approach began with ingesting source application data into the **staging layer**, preserving the original schema to maintain data fidelity. Each staging metadata mirrored its source counterpart, ensuring seamless traceability with insert, update, delete flag. Corresponding **hub tables** were established to manage business keys, forming the foundational structure for downstream processing.

The hub table stores the business key, its corresponding hash key (generated using a hash function), and the load date. The hub of the driver entity serves as the starting point, which is then joined with other hubs based on business rules to populate intermediate entities. The general approach is to first include the business key in the intermediate layer, then add calculated values, and finally append contextual information before loading into the final entity—be it a **dimension, fact, enterprise object** or **bridge**. In Data Vault 2.0 terms, dimensions and facts are modeled as **satellite entities**, while **bridges** are represented as **link entities**. Satellite, hub, and link entities will each include at minimum a business key, a hash key, and a load date. Hash keys are created for both individual and combined business keys. Depending on the requirement, satellite, link, and intermediate entities can be implemented as physical objects (for batch processing) or logical objects (for real-time access).

The satellite and hub are considered part of the **Raw Vault** in the given context. When additional attributes are required beyond what the enterprise object provides, a link table can be introduced to support this expansion. Dimensions that are specific to a data mart should selectively expose only a limited subset of attributes from the enterprise object, ensuring targeted and efficient data representation. Moreover, link and satellite entities are to be treated as essential components of the **Business Vault**.

Finally, the **Information Vault Layer** orchestrates the integration of link entities, dimensions, and facts into a **star schema**, optimized for downstream consumption by BI tools and analytics platforms.

The following diagram illustrates the structural organization of entities across each layer.

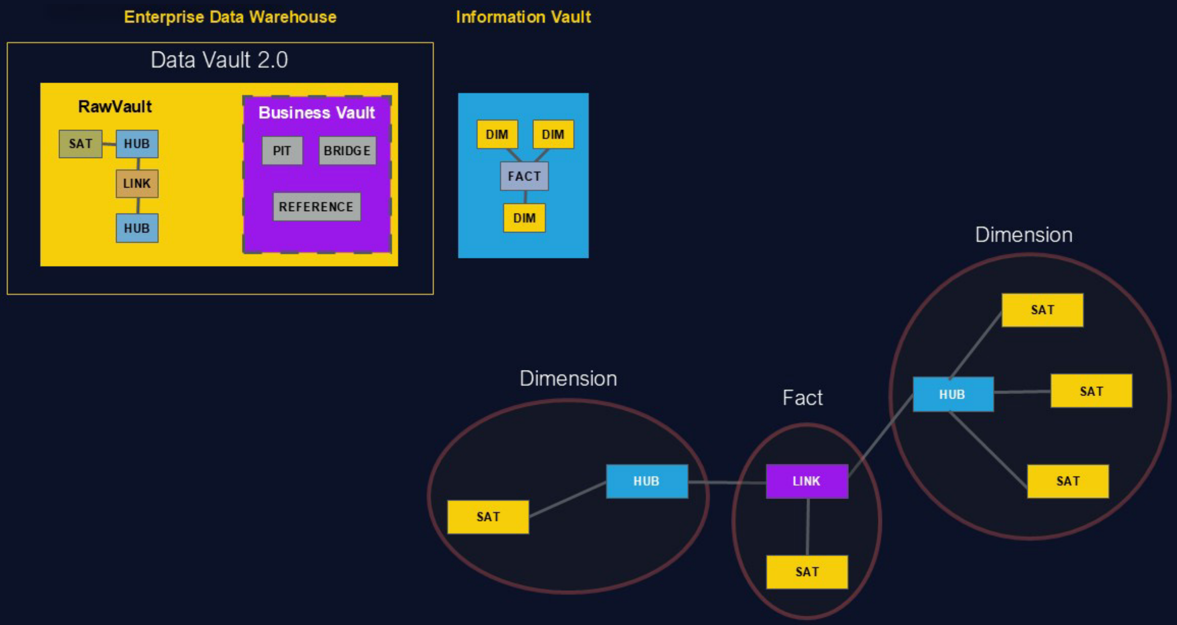


Figure 3: Information Vault

02 Enterprise Object: A Unified, Scalable approach to Data Integration

The purpose of an **enterprise object** is to serve as a centralized, authoritative source of information for logical business entities across the organization. It encapsulates both **contextual** data at different level of details, ensuring consistency and clarity in analytical and operational use cases.

For example, an **order (ORD)** enterprise object would include comprehensive data such as **order** details. All analytical dimensions derived as **filtered subsets** of this enterprise object, containing only the relevant fields needed for specific reporting or analysis.
By first constructing enterprise objects within the **Business Vault**, and then layering dimensions on top of them, organizations can:

- Maintain a single source of truth
- Reduce data redundancy
- Simplify governance and access control
- Enable scalable, multi-purpose analytics

This approach ensures that dimensions remain lean, focused, and aligned with enterprise-wide data standards.

03 Enterprise Object:
Solving Data Diversity with Link and Satellite Entities

EDOs also tackle one of the most complex challenges in enterprise data integration: **heterogeneous data sources with inconsistent keys and structures**. Consider the **person EDO**, which must merge employee data from both HCM and Siebel systems:

- In **HCM**, the key is person ID, and driver license data may be absent
- In **Siebel**, the key is driver license number, and it includes attributes like years of driving experience, which HCM lacks

To harmonize these differences, we introduce a link entity that bridges the gap:

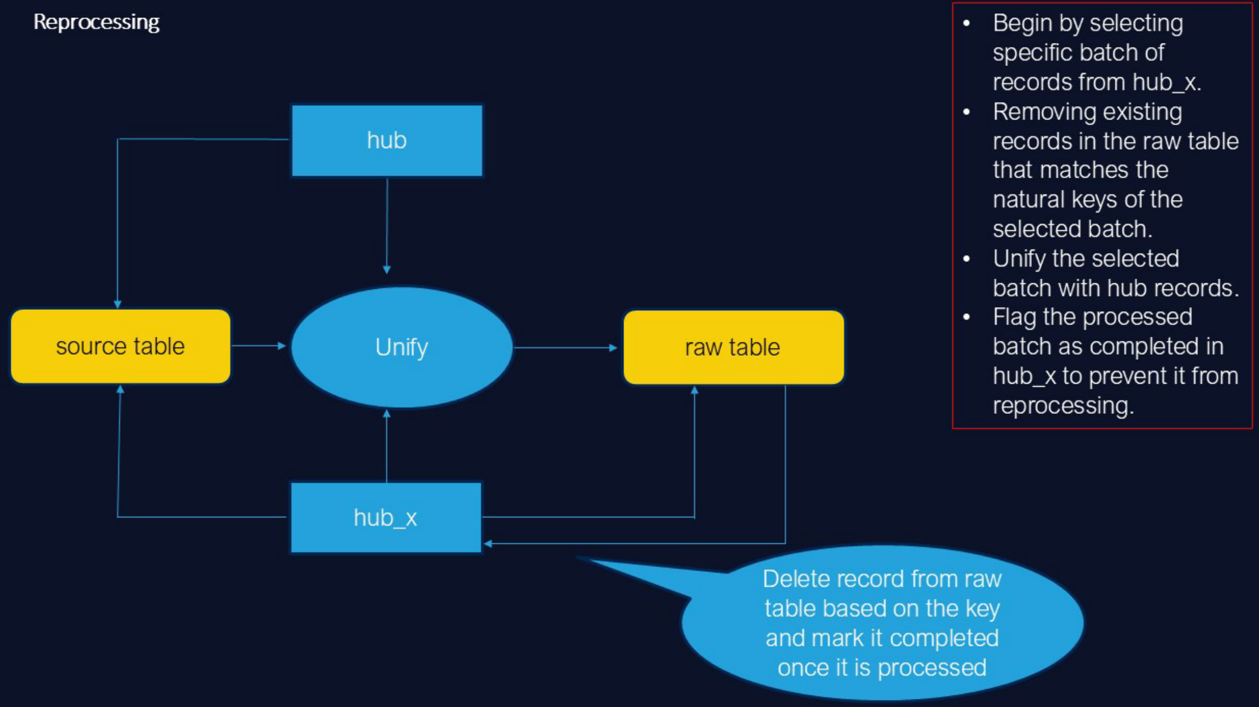
- **Key 1:** Person ID + source system code
- **Key 2:** Driver license number + source system code

If a lookup object exists that maps person ID to driver license, it can be used to map key 1 into key 2 in the link entity. If no such mapping is available, key 2 remains empty, and key 1 retains both identifiers, enabling connections to other entities using either key.
A fact entity may contain either key 1 or key 2, and it is linked to the corresponding dimension through a link entity, which is based on the enterprise object.

04 Optimized Reprocessing Strategy

Sometimes, past data needs to be reprocessed to add new source information or fix errors. To make this efficient, business keys for flagged records can be batched and fed into the regular pipeline with minimal system impact. This allows dynamic reloading without downtime and supports near real-time processing.

Here is a diagram to illustrate the reprocessing strategy.


Reprocessing
hub
source table
Unify
raw table
hub_x
Delete record from raw table based on the key and mark it completed once it is processed

- Begin by selecting specific batch of records from hub_x.
- Removing existing records in the raw table that matches the natural keys of the selected batch.
- Unify the selected batch with hub records.
- Flag the processed batch as completed in hub_x to prevent it from reprocessing.

Figure 4: Reprocessing Logic

05 Data Unification Strategy

To unify data from Siebel, Mercury Gate, and Oracle Fusion Cloud CX, each source should be processed independently and then unified in the Information Vault. . For example, Order data from Siebel,Mercury Gate and Oracle Fusion Cloud CX combined for a 360 degree view. This approach supports dynamic integration, allowing systems to be added or removed or cross-referenced without disruption, and ensures modular, scalable architecture as systems evolve.

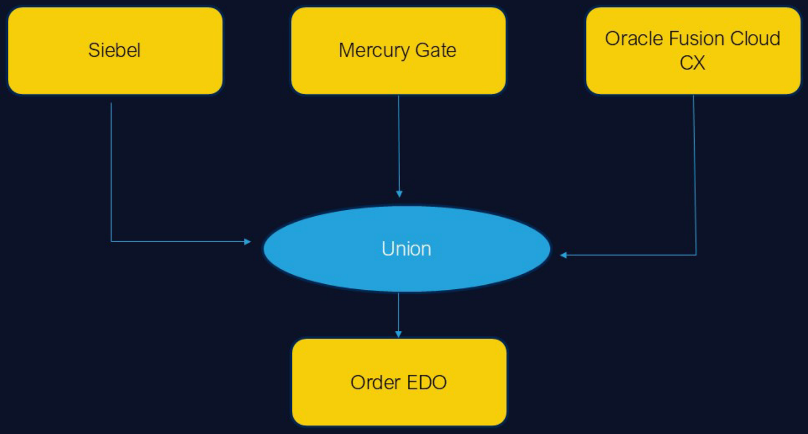


Figure 4: Reprocessing Logic

06 Textual/Semi-Structured Data

The final entity holds both a business key and a hash key. Final entity—whether a dimension, fact, or enterprise object—a lookup on the staging object using the business key retrieves textual or semi-structured data. Whether the final entity is logical or physical, it points to the staging physical object so that data is stored in one place with a centralized access policy.

07 Data Access and Security

To ensure robust data security and governance, access policies will be enforced at the **Staging layer**, which serves as the foundational data store. The **RAW Vault/Business Vault layer**, typically built as logical object over the staging layer, inherits these access controls unless standalone objects are created, where separate policies must be applied.

Even a standalone object can reference sensitive data in the staging layer through the business key. This enables a centralized approach for managing access to sensitive information. If sensitive data needs to be removed, deleting it from the staging layer will ensure it is removed system-wide.

Each business domain (e.g., **orders, offers, transportation**) will have a dedicated **data governance team** responsible for managing data visibility and sharing rulesThese teams will classify which data items are considered sensitive and identify the user groups eligible to access them. Based on this eligibility, access to sensitive data will be granted. Authorized users will be able to view the actual data, while others will see encrypted versions—ensuring secure and compliant data handling..

Access will be managed using **role-based access control (RBAC)**, enabling fine-grained permissions based on user roles and responsibilities. This approach not only strengthens data security but also simplifies governance across multi-domain environments.

Outcome

The following requirement traceability matrix demonstrates how the specified requirements were effectively addressed through the above approach.

Dimension	Approach
Architecture Evolution	Scalable business intelligence
Historization	• Use of load date • Reprocessing strategy
Enterprise Object Evolution	• A unified, scalable approach to data unification • Solving data Diversity with link and satellite entities • Data unification strategy
Metrics Evolution	Scalable business intelligence
Real-Time Data	Scalable business intelligence
Centralized Data Access and Security	• Data access and security • Textual/Semi-Structured data
Phase-wise Deployment	Scalable business intelligence

Table 1: Requirement Traceability Matrix

Best Practices

1. Start with Business Keys

- Identify stable, unique business identifiers for hubs
- Avoid using surrogate keys unless absolutely necessary

2. Use Hash Keys for Consistency

- Generate hash keys for hubs and links to ensure uniqueness and simplify joins
- Use SHA-1 or SHA-256 for hash generation

3. Insert-Only Strategy

- Avoid updates and use insert-only logic to preserve historical data
- This supports auditability and simplifies ETL logic

4. Parallel Loading

- Load hubs, links, and satellites independently to maximize performance
- This is especially effective in cloud platforms like Snowflake or Databricks

5. Pattern-Based Modeling

- Use standardized templates for hubs, links, and satellites
- Promotes consistency and simplifies automation

Comparison Table

Aspect	Inmon	Kimball	Data Vault 2.0
Approach	Top-down	Bottom-up	Top-down
Architecture	Centralized	Distributed	Centralized
Modelling Style	3NF (Normalized)	Dimensional (Star/Snowflake)	Hub, link, satellite, bridge, reference tables
Historization	Not ideal	Preferred–Start date, end date and current indicator	Preferred – Start date
Key Strategy	Primary key and foreign key	Surrogate key	Hash key based on natural key
Real-Time Analytics	Not preferred	Not preferred	Preferred
Big Data Compatibility	Not preferred	Structured data	Structured/Semi-structured/ Unstructured
BI Reporting	Data mart	Data mart	Information Vault
Agility	Not preferred	Not preferred	Preferred

Table 2: Comparison of Three Different Data Modelling Approaches

Key Insights

1. Multi-granular Dimension and Fact

Model dimensions and facts as multi-granular satellite entities and use link entities to connect data across different levels of granularity. This approach minimizes the number of dimension and fact tables, streamlining the data architecture and reducing number of entities for maintenance.

2. Duplication of Staging Logic

This approach may result in some duplication of staging logic and lead to increased compute costs. It helps avoid rework and supports the dynamic scalability of the model.

3. Effort Estimation

The Data Vault 2.0 modelling methodology aligns seamlessly with agile development practices, supporting iterative delivery and scalable design. To estimate development effort effectively, a T-shirt sizing approach (e.g., small, medium, large) is recommended.

For multi-granular entities, treat each level of granularity as a distinct entity during estimation.

Conclusion

Adopting Data Vault 2.0 modelling has fundamentally reshaped our approach to data architecture—ushering in not just a technical enhancement, but a strategic shift in mindset. This methodology empowers us to build scalable, maintainable, and highly agile systems , while also streamlining compliance with data privacy regulations such as GDPR through centralized management of sensitive data. The impact is measurable: testing efforts are reduced to just (n–1) iterations per entity, and analysis time drops by 20–30%, particularly when refining entities to meet evolving business requirements. For organizations navigating cloud migrations or burdened by repetitive data model rebuilds, this approach offers a resilient, future-ready solution that reduces complexity and accelerates delivery. It is especially effective in industries where clients embrace a minimum viable product (MVP) approach, enabling faster iteration and value realization. Whether you're a data engineer, architect, or project manager, this framework is designed to empower your teams and drive meaningful outcomes. Let’s collaborate, exchange ideas, and continue evolving data engineering into a discipline that’s goes beyond mere efficiency and is truly transformative.

About the Author

Arijit Mitra
Associate Principal, Data Engineering, LTIMindtree

With two decades of experience in the IT industry, Arijit is a seasoned Data Architect who has consistently delivered innovative and scalable data solutions across global enterprises in the distribution, healthcare, and travel and transport sectors. His career spans a rich blend of roles including ETL designer, report designer, virtualization and business analysis, which have collectively shaped his holistic understanding of data ecosystems.

References

Modeling Data Warehouse with Data Vault 2.0, Esra Ekiz

https://www.udemy.com/course/modeling-data-warehouse-with-data-vault-for-beginners/?kw=Modeling+Data+Warehouse&src=sac

Prescriptive Guidance for Implementing a Data Vault Model on the Databricks Lakehouse Platform, Soham Bhatt, Tanveer Shaikh and Glenn Wiebe, June 24,2022

https://www.databricks.com/blog/2022/06/24/prescriptive-guidance-for-implementing-a-data-vault-model-on-the-databricks-lakehouse-platform.html

Tips for Optimizing the Data Vault Architecture on Snowflake, Kent Graziano and Keith Hoyle, Feb 3,2020

https://www.snowflake.com/en/blog/tips-for-optimizing-the-data-vault-architecture-on-snowflake/?_fsi=eLBYi14A