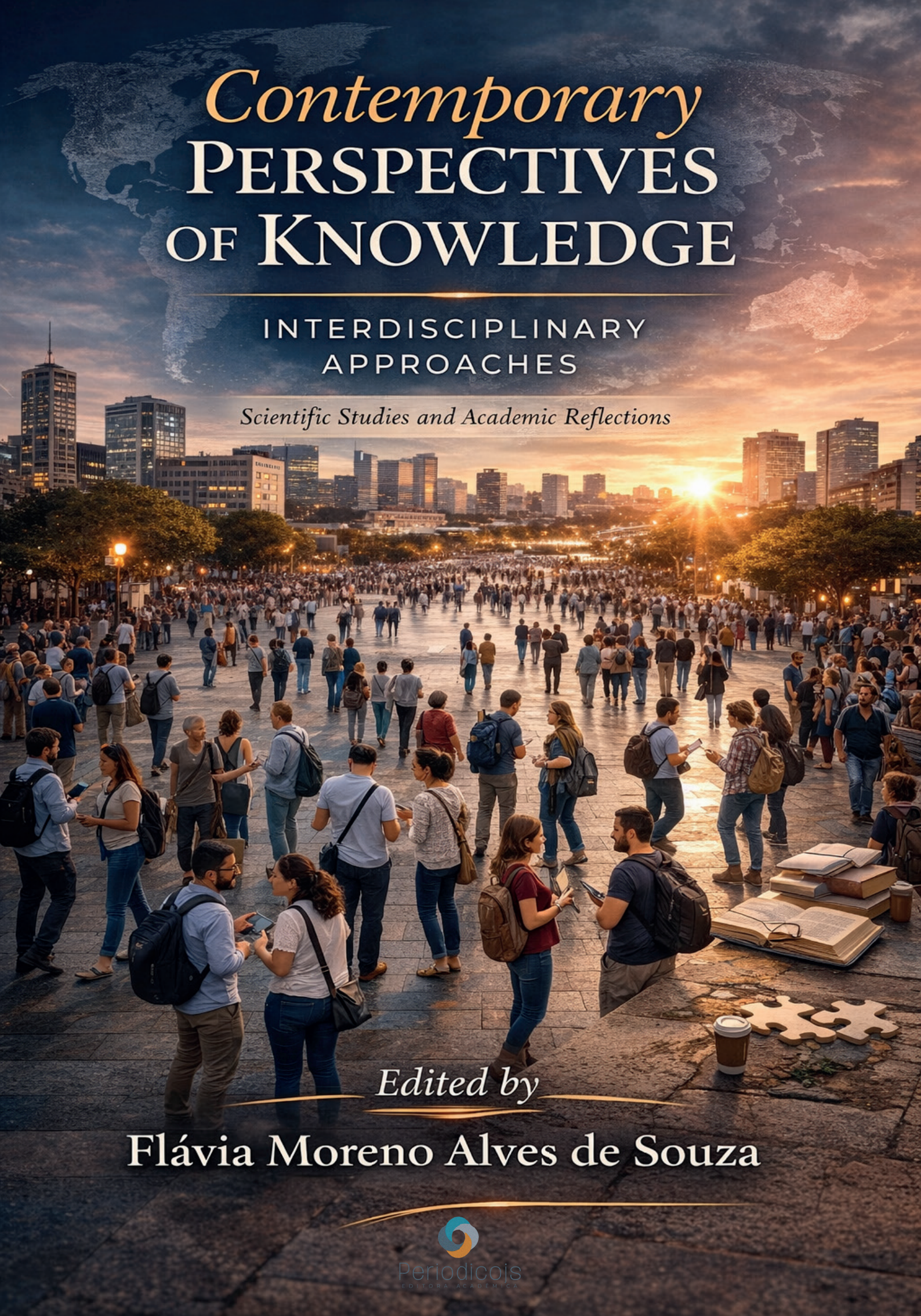




Contemporary PERSPECTIVES OF KNOWLEDGE

INTERDISCIPLINARY
APPROACHES

Scientific Studies and Academic Reflections

Edited by
Flávia Moreno Alves de Souza

Editorial Team

Abas Rezaey	Izabel Ferreira de Miranda
Ana Maria Brandão	Leides Barroso Azevedo Moura
Fernado Ribeiro Bessa	Luiz Fernando Bessa
Filipe Lins dos Santos	Manuel Carlos Silva
Flor de María Sánchez Aguirre	Renísia Cristina Garcia Filice
Isabel Menacho Vargas	Rosana Boullosa

Graphic Design, Layout and Cover

Academic Publisher Periodicojs

Language

Portuguese and English

International Cataloging-in-Publication Data (CIP)

(Brazilian Book Chamber, SP, Brazil)

C761	Contemporary Perspectives of Knowledge: Interdisciplinary Approaches/ Flávia Moreno Alves de Souza (org) – João Pessoa: Periodicojs publisher, 2026. E-book: il. color. Includes bibliography ISBN: 978-65-6010-196-8 1. Free themes. I. Souza, Flávia Moreno Alves. II. Title
CDD 370	

Prepared by Dayse de França Barbosa CRB 15-553

Index for systematic catalog:

Indexes for systematic catalog:

1. Education: 370

Work without funding from public or private bodies.

The published works have been submitted to peer review and evaluation (double-blind), with respective acceptance letters in the publisher's system.

The work is the result of studies and research from the Interdisciplinary Studies in Human Sciences section of the Humanities in Perspective book collection.



Filipe Lins dos Santos
President and Senior Editor of Periodicojs

CNPJ: 39.865.437/0001-23

Rua Josias Lopes Braga, n. 437, Bancários, João Pessoa - PB - Brazil
website: www.periodicojs.com.br
instagram: @periodicojs

Chapter 29

DATA MODELING FOR BUSINESS INTELLIGENCE IN TRANSPORTATION LOGISTICS



DATA MODELING FOR BUSINESS INTELLIGENCE IN TRANSPORTATION LOGISTICS

Gleudson Rondinelli Cardoso¹

Abstract: This chapter aims to lay the conceptual and technical groundwork needed to build a Business Intelligence (hereafter, BI) environment for transportation logistics. The goal is not to offer a superficial review of generic information technology concepts artificially transplanted into the transportation industry. Rather, drawing on the established literature on data architecture and dimensional modeling, it builds a framework that speaks directly to the operational realities of freight and passenger transportation—namely, the dispersion of data sources, the heterogeneity of data formats and capture frequencies, and the structural fragility of integration between legacy systems and enterprise resource planning (ERP) platforms. The chapter covers five complementary areas. First, it discusses why transportation logistics is a particularly complex data domain, in light of the information systems and supply chain literature (CHRISTOPHER, 2016; BALLOU, 2006). Second, it presents a layered data architecture—from the operational source to the analytical dashboard—grounded in the seminal contributions of Inmon (2005) and Kimball and Ross (2013) on data warehousing and dimensional modeling. Third, it details the Extract, Transform, and Load (ETL) processes, with particular attention to the data quality issues discussed by Wang and Strong (1996) and Redman (2001). Fourth, it examines integration with enterprise management systems and the viable strategies for dealing with legacy systems that lack a structured application programming interface. Finally, it consolidates best practices in data modeling and data governance, aligned with the DAMA International (2017) body of knowledge, and presents a real, anonymized case study that illustrates the practical application of the concepts discussed. The decision to address these topics in an integrated way—rather than as isolated blocks of technical content—reflects an underlying

¹ Bachelor's degree in Mechanical Engineering from the Federal University of Uberlândia.



conviction that guides this entire chapter, one shaped by the author's hands-on experience building fleet control departments and implementing quality management systems (ISO 9001 and SiAC/PBQP-H): BI projects fail far more often because of flawed architectural decisions at the outset than because of limitations in the visualization tools used at the end of the process. A visually sophisticated dashboard built on a poorly modeled database is a façade—and that façade is exactly what this chapter sets out to prevent.

Keywords: data model, transportation, business intelligence

Introduction: Why Transportation Logistics Is a Particularly Complex Data Domain

The claim that “every business domain has its own data challenges” is, to some extent, trivial. This chapter advances a more specific thesis: transportation logistics presents a particular combination of characteristics that makes it structurally harder to model analytically than comparable domains such as retail or financial services. This difficulty does not stem from any single technological limitation, but from a systemic condition of the business process itself: transportation operations are, by definition, physically distributed, carried out by multiple actors (drivers, fueling stations, maintenance shops, regulatory agencies), and mediated by information systems that were rarely designed to talk to one another.

In industries such as retail, much of the relevant transactional data originates within a single system—the point of sale—and reaches the organization in a relatively structured format that is immediately usable for analytical purposes. In transportation logistics, the opposite is true: data originates scattered across vehicle tracking systems, mobile driver-hours apps, fuel-log spreadsheets filled out by hand at partner fueling stations (and also by hand when the company runs its own on-site station), preventive and corrective maintenance systems, and the financial modules of enterprise ERPs, which treat entities such as “vehicle” and “route” as secondary attributes of accounting entries



rather than as first-class objects in the data model. This dispersion is not incidental; it is a direct consequence of the physical and organizational nature of transportation itself, which Christopher (2016) describes as one of the links in the supply chain with the greatest operational variability and the greatest dependence on uncontrollable external factors—traffic conditions, weather, and the availability of road infrastructure.

Volume, Variety, Velocity, and Veracity of Operational Data

The big data literature typically characterizes the complexity of a data domain along three classic dimensions—volume, variety, and velocity—to which a fourth is now added with reasonable consensus: veracity, that is, the degree of reliability and accuracy of the captured data (REDMAN, 2001). When applied to fleet operations, these four dimensions reveal why transportation logistics demands above-average architectural care.

The volume of data generated by a mid-sized transportation operation is often underestimated by managers who still think in terms of “one row per trip.” A fleet of fifty vehicles running eight hours a day, with telematics systems capturing geographic position every thirty seconds, easily generates tens of millions of location records per month—not counting fueling data, maintenance logs, driver-hours records, and route incidents. Beyond a certain level of operational maturity, this volume alone rules out any attempt at manual consolidation in spreadsheets and calls for a matching storage and processing infrastructure—a point we will return to in Section 1.2 when we discuss the layered architecture.

Variety is perhaps the most underestimated dimension among non-technical managers. Different sources produce data in radically different formats: comma-separated values (CSV) files exported by tracking systems; spreadsheets filled out by hand at partner fueling stations with no standardized fields whatsoever; maintenance checklist records filed as PDFs or even on paper and scanned later; geospatial coordinates from global positioning systems; and work-hour logs kept in



human resources systems entirely separate from the fleet management system. Each of these sources has its own internal logic for identifying vehicles, drivers, and dates—a semantic heterogeneity that, as we will discuss in Section 1.4, is the root cause of most of the integration problems faced by mid-sized and large operations.

The velocity at which data is generated also varies enormously within the same domain. Very high-frequency data—telematics captured in real time, every few seconds—coexists with very low-frequency data, such as a monthly physical tire audit or a quarterly scheduled preventive maintenance service. A data model that ignores these differences in velocity by forcing all information into a single, artificial level of granularity tends to produce either an unnecessary explosion in volume (when trying to “flatten” everything to the finest grain) or a loss of relevant information (when trying to simplify everything to the coarsest grain). We will return to this problem, with a concrete proposed solution, in the discussion of mixed granularities in Section 1.2.

Finally, the veracity—or reliability—of data captured in transportation logistics environments is systematically lower than in controlled transactional environments such as a retailer’s point of sale. This stems in part from the physical environment in which capture takes place: GPS signals suffer interference in tunnels and dense urban areas or become temporarily unavailable in low-coverage regions, onboard telematics sensors can have calibration failures, and much of the fueling and maintenance data still depends on manual entry, which is prone to human typing errors and inconsistent criteria across different operators. Wang and Strong (1996), in one of the foundational works of the data quality literature, propose a taxonomy of quality dimensions—accuracy, completeness, consistency, timeliness, and accessibility—that proves especially useful for diagnosing, in a structured way, exactly where the reliability problem lies in each data source of a specific transportation operation, instead of treating “bad data” as a generic, undifferentiated problem.



The Distinction Between Operational Data and Analytical Data

One of the most common conceptual mistakes among teams starting BI projects—and one that, in the author’s hands-on experience, accounts for much of the rework and frustration in the early phases of a project—is confusing operational data with analytical data. The data warehousing literature has treated this distinction as fundamental since Inmon’s seminal work (2005), yet it is rarely spelled out clearly enough in real-world implementation settings.

Operational data exists to support the execution of operations in the present moment. It lives inside transactional systems—the vehicle tracking system, the ERP fleet module, the mobile app used by the driver—and its primary function is to record discrete events as they happen: a vehicle departing, a fueling being confirmed, a delivery being completed. From a software engineering standpoint, systems that store this kind of data are designed to optimize write speed and guarantee immediate transactional consistency (properties known in the database literature by the acronym ACID—atomicity, consistency, isolation, and durability), not to support complex analytical queries involving aggregations over large volumes of historical data (ELMASRI; NAVATHE, 2015).

Analytical data, in turn, is the result of a deliberate process of reorganizing, cleansing, and remodeling operational data, with the explicit purpose of answering management questions that rarely match the original structure in which the data was captured. Questions such as “What is the cost per kilometer driven per vehicle last quarter?”, “Which routes show the greatest variability in travel time?”, or “Which driver has had the highest rate of hard-braking events over the past six months?” require data to be organized in a way that favors reading, aggregation, and cross-referencing across multiple dimensions of analysis—the exact opposite of the write-optimized design that characterizes operational data.

The most common mistake among teams new to logistics BI is trying to answer analytical questions by querying the source system’s operational tables directly, whether through ad hoc queries against the transactional database or through manual exports and spreadsheet cross-referencing.



This approach works—seemingly well enough—as long as data volumes are small and the business questions are simple. It stops working—abruptly, and often traumatically for the team involved—as soon as the operation scales up or management questions gain even a modest degree of sophistication, such as combining three or more different sources. The layered data architecture discussed in the next section exists precisely to solve this problem structurally rather than with stopgap fixes.

Chapter roadmap. The following sections follow a logical progression intended to mirror the order in which these decisions should actually be made in a real-world project. Section 1.2 presents the layered architecture that organizes the flow of data from its source to the analytical dashboard, including a discussion of dimensional modeling and the problem of multiple coexisting granularities. Section 1.3 details the ETL processes that move and transform data between these layers, with attention to the dimensions of data quality. Section 1.4 examines integration with enterprise management systems, including strategies for dealing with legacy systems that lack structured programming interfaces—a far more common scenario than the enterprise BI literature tends to acknowledge. Section 1.5 consolidates best practices for naming conventions, documentation, and data governance. Section 1.6 presents a real, anonymized case study that illustrates how the concepts discussed work together in practice, with quantified results. Finally, Section 1.7 wraps up the chapter with a data architecture maturity checklist, a tool that teams at any stage of maturity in their BI environment can use for self-assessment.

Data Architecture: From Source to Dashboard

Typical Layers of a Mature Data Architecture

With some variation in terminology, the data warehousing literature converges on an architectural model organized into successive layers, each with a specific and clearly defined responsibility (INMON, 2005; KIMBALL; CASERTA, 2004). For the transportation logistics context, this chapter proposes a four-layer architecture: operational sources, staging, analytical model, and



visualization.

The operational sources layer comprises the systems where the data actually originates—the enterprise ERP, the vehicle tracking system, the fuel-log spreadsheets, the maintenance systems, and the driver-hours apps. It is important to stress that this layer should never, under any circumstances, be queried directly by the end visualization tools, for two complementary reasons: first, because heavy analytical queries against transactional databases can degrade the performance of the operational systems themselves, posing a real risk to day-to-day operations; and second, because, as discussed in Section 1.1, the format in which data resides in this layer rarely matches the structure needed to answer analytical questions.

The staging layer—sometimes called the “landing zone”—is an intermediate area where raw data is copied exactly as it appears at the source, without any transformation. This layer serves a twofold purpose. On the one hand, it preserves a faithful copy of the original data, acting as an audit trail that makes it possible to reconstruct the exact state of the information at any given moment and to fully reprocess the data if a transformation rule applied downstream turns out to be wrong. On the other hand, it technically decouples extraction from transformation, so that an isolated failure in either step—for example, a source API being temporarily unavailable—does not compromise the integrity of the process as a whole. Vassiliadis (2009), in a comprehensive review of the ETL literature, singles out this “failure-buffering” role as one of the strongest technical justifications for maintaining a separate staging layer, even in smaller-scale architectures.

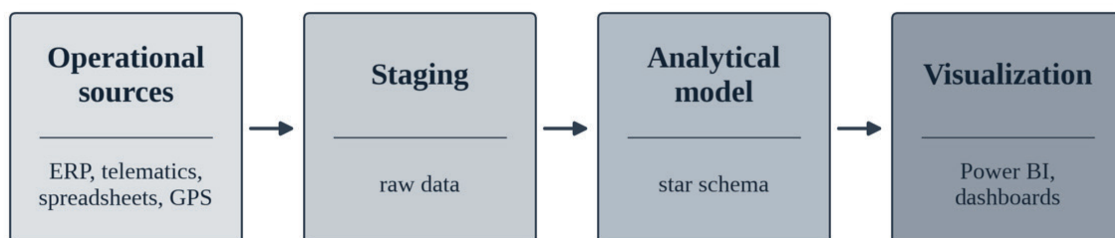
The analytical model layer is where data—already cleansed, standardized, and reorganized—takes on a query-optimized structure, typically a star schema, whose logic is detailed later in this section. It is in this layer that business concepts such as “trip,” “vehicle,” “route,” and “driver” stop being fragmented and potentially conflicting representations across different source systems and take on a single, consistent, auditable definition—what the data governance literature calls dimension “conformance” (KIMBALL; ROSS, 2013).

Finally, the visualization layer is where managers actually interact with the data, whether



through tools such as Power BI and Tableau or through custom dashboards built in-house. An architectural principle that admits no exceptions is that this layer must consume only the analytical model and never the operational sources directly—a principle that, however obvious it may sound when stated, is frequently violated in practice by teams under deadline pressure who opt for direct-integration “shortcuts” that, over time, turn into technical debt that is hard to unwind.

Figure 1 – Layered data architecture, from the operational source to the BI dashboard



Source: prepared by the author.

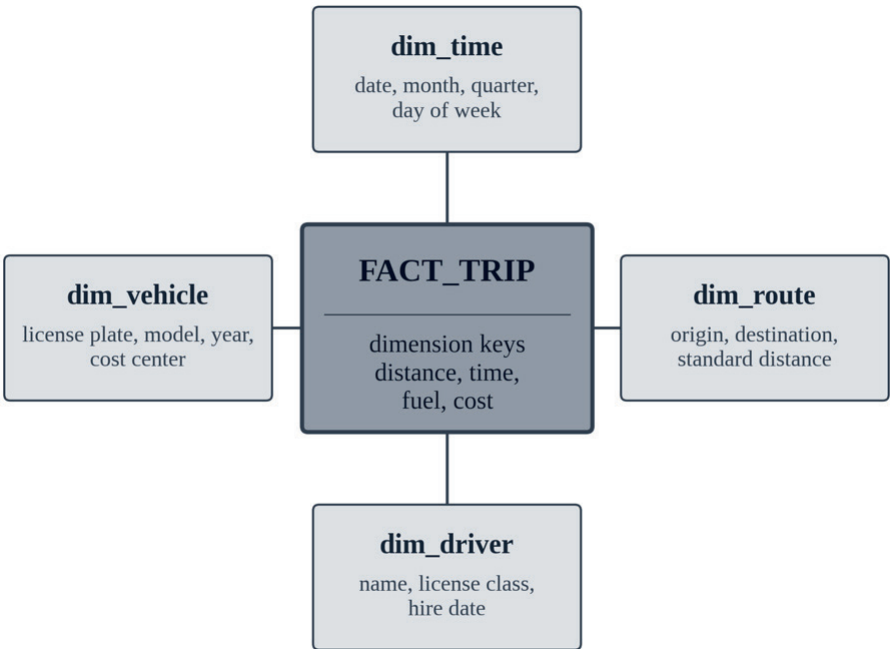
The Star Schema as the Structure of the Analytical Model

Within the analytical model layer, the predominant way of organizing tables in relation to one another is the so-called star schema, formalized and widely popularized by Ralph Kimball beginning in the 1990s and established in the literature by Kimball and Ross (2013) as the reference approach for dimensional modeling geared toward high-performance analytical queries. Choosing the star schema over alternatives—such as the snowflake schema, a normalized variation, or a purely relational model along the lines originally proposed by Codd (1970) for transactional systems—is not arbitrary: it reflects a deliberate prioritization of query simplicity and read performance over storage efficiency, a priority that makes sense precisely because the analytical layer is meant to be read frequently and written relatively rarely (once or a few times a day, typically through overnight batch loads).



The logic of the star schema can be summarized as follows: a central fact table holds the quantitative metrics of the business process—in the case of a transportation operation, distance traveled, trip time, fuel consumption, and associated cost—at the grain of detail discussed below. Surrounding this fact table, a set of dimension tables describes the qualitative, descriptive attributes of the business entities involved: the vehicle dimension carries license plate, model, model year, and cost center; the driver dimension carries name, license class, and hire date; the route dimension carries origin, destination, and standard distance; and the time dimension carries date, month, quarter, and day of the week, enabling time-based analysis without the need for recalculation at query time. Figure 2 illustrates this arrangement in simplified form:

Figure 2 – Simplified representation of a star schema



Source: prepared by the author.

The FACT_TRIP table, at the center of the schema, carries only identifiers—known in the



literature as surrogate keys—for each of the associated dimensions, plus the quantitative metrics themselves. The dimension tables, in turn, carry the descriptive attributes that give these keys their business meaning. It is this design—rather than a single “flattened” table combining all attributes and metrics into one structure—that allows business users, within the BI tool, to build analyses that combine any mix of dimensions and metrics (“cost per route per month,” “consumption per driver per quarter”) without the data analyst having to rewrite specific queries for every new question that comes up. The comparison presented in Section 1.7 systematically details the advantages of the star schema over the alternative of “flat” (single-table) modeling.

Another modeling issue that is often overlooked in introductions to the subject also deserves mention: how to handle dimension attributes that change over time—for example, a vehicle that is moved to a different cost center, or a driver who is upgraded to a higher license class. Kimball and Ross (2013) call this phenomenon “slowly changing dimensions” (SCD) and propose three main strategies for handling it: Type 1, in which the old value is simply overwritten by the new one, appropriate when the history of the change has no analytical relevance; Type 2, in which a new row is inserted into the dimension table with each change, preserving the full history through effective dates; and Type 3, in which an additional column is kept for the previous value, useful when only the most recent change matters. In the transportation logistics context, Type 2 is usually the best fit for the vehicle dimension—ensuring, for example, that a historical cost-by-cost-center analysis correctly reflects the allocation in effect in each period, rather than retroactively applying the current allocation to the entire history.

Criteria for Choosing Granularity

One of the most foundational decisions in any dimensional modeling project is defining the granularity—or grain—of the fact table, that is, precisely defining what a single row in that table represents. Kimball and Ross (2013) are emphatic that this decision must be made explicitly and



documented before any other modeling step, precisely because granularity decisions made implicitly or ambiguously are the most common cause of rework in data warehouse projects.

In the specific context of transportation logistics, four granularity options are most commonly adopted, each with its own uses and limitations, as summarized in Table 1.

Table 1 – Fact table granularity options in transportation logistics

Granularity	When to use	Limitation
Per trip	Route efficiency, cycle time, and cost-per-trip analyses	Does not capture in-trip variations (stops, detours)
Per day/vehicle	Consolidated management KPIs, executive dashboards	Loses the detail needed for root-cause investigation
Per telematics event	Driving behavior, safety, and predictive maintenance analyses	Very high data volume; requires robust infrastructure
Per route (fixed segment)	Performance comparison across recurring routes	Not well suited to operations with dynamic routing

Source: prepared by the author.

The general recommendation, supported both by the literature and by the hands-on experience reported in this chapter, is to adopt per-trip granularity as the primary grain of the central fact table and to build later aggregations—by day, by vehicle, by period—from that finer grain, not the other way around. This choice avoids having to fully reprocess the database every time a new management question calls for a different time or dimensional slice than originally planned—a principle that Kimball and Ross (2013) sum up in the maxim that “the most atomic grain available should always be preferred,” precisely because aggregations can always be derived from atomic data, whereas the reverse—reconstructing detail from already-aggregated data—is structurally impossible.

The Practical Problem of Mixed Granularities

In most real-world mid-sized and large operations, however, a single grain is rarely enough to meet all of the organization’s analytical needs. It is quite common for the same BI environment to



contain both a relatively coarse-grained fact table—the `fact_trip` table, with one row per completed trip—and a much finer-grained fact table—the `fact_telematics_event` table, with one row for each location event captured every thirty seconds. Treating these two tables as independent structures, with no formal, documented relationship between them, is one of the most common modeling mistakes seen in practice: the data analyst ends up being forced to compare “cost per trip” with “hard-braking events per trip” through manual joins, built ad hoc and often inconsistently each time a new business question comes up.

The most robust solution to this problem—and the one actually adopted in the case study presented in Section 1.6—is to create an intermediate aggregate fact table that rolls up the fine-grained telematics events to the same grain as the trip fact table. A table such as `fact_trip_telematics_summary`, containing one row per trip with aggregated metrics such as the count of hard-braking events, the maximum speed reached, and the total idle time during the trip, shares the same dimension keys as the original `fact_trip` table—vehicle, driver, route, and time—allowing the two to be combined natively in any BI tool without requiring business users to work directly with the raw volume of telematics events. When a deeper root-cause analysis calls for the fine-grained detail of a specific event, the data analyst can go back to the original events table—but the day-to-day management dashboard never needs to touch it directly, preserving both query performance and ease of use for the end user.

ETL Processes in the Transportation Industry

The acronym ETL—Extract, Transform, and Load—refers to the set of processes responsible for moving data from the operational sources to the analytical model, passing along the way through the staging layer discussed in the previous section. Although the acronym suggests a strictly linear sequence, Vassiliadis (2009) notes that in real-world implementations these three steps often overlap and feed back into one another—decisions made during transformation, for example, may require



retroactive adjustments to how extraction is performed. In this section, we discuss each of these steps with attention to the particularities of the transportation industry.

Extract

The extraction step consists of collecting raw data from the operational sources while preserving, as far as possible, its fidelity to the original format—fidelity that will later serve as an audit reference in the staging layer. In a typical transportation operation, the main extraction sources differ from one another in their characteristics and technical challenges, which means each one must be handled with its own extraction strategy rather than a single, one-size-fits-all approach.

Telematics and vehicle tracking systems typically expose their data through REST application programming interfaces (APIs) or, alternatively, through batch exports in formats such as CSV or JSON. A technical detail often overlooked at this stage is the handling of time zones and sampling rates, which vary considerably across telematics vendors—an inconsistency that, if not explicitly addressed during extraction, silently propagates through every subsequent step of the process, undermining comparability between vehicles equipped with tracking systems from different vendors.

Fuel-log spreadsheets remain, even today, an extremely common source in operations that rely on partner fueling stations with no direct electronic integration into the fleet management system. This source necessarily requires a manual or semi-automated extraction process—for example, a scheduled job that reads a shared folder where files are dropped periodically—and that process must include basic structural validations (checking for required columns and expected data types) before a file is accepted into the staging layer; otherwise, manual data-entry errors will propagate to later stages of the pipeline.

ERP systems, for their part, generally provide financial and maintenance data through directly accessible database tables or batch-exportable reports. It is particularly common for there to be a time lag between the operational event itself—the vehicle maintenance actually took place on



a given date—and its accounting entry in the ERP, which may occur weeks later depending on the service provider’s billing cycle. This lag, discussed in more detail in Section 1.4, must be handled explicitly in the data model; otherwise, it will systematically distort cost metrics tied to the period in which the event actually occurred.

Finally, driver-hours and checklist apps are an increasingly important source, both for regulatory compliance and for analytical cross-referencing with telematics data. In the Brazilian context, this regulatory relevance stems in particular from Law No. 13,103/2015—known as the Truck Driver Law—which governs working hours, maximum continuous driving time, and mandatory rest periods for professional drivers (BRASIL, 2015). Integrating driver-hours data with telematics data makes it possible, for example, to identify high-risk situations in which a driver shows signs of fatigue (continuous driving time beyond the legal limit) combined with hard-braking or speeding events—an analytical cross-reference that is only possible when both sources converge into the same dimensional model.

Transform

The transformation step is where most of the technical effort in an ETL project is concentrated—and, correspondingly, where most of the analytical value is added to the raw data. Redman (2001) argues that the quality of the final analytical data is determined far more by the quality of the transformation rules applied than by the sophistication of the visualization tool used at the end of the process, an argument this chapter fully endorses and one that motivates the detailed attention given to this step.

Standardizing units of measure is a classic, recurring problem in logistics environments: the same physical quantity may arrive in different units depending on the source—liters and gallons in fueling records from different regions or suppliers, kilometers and miles in tracking systems from international manufacturers, metric tons and kilograms in cargo records. The entire database must



converge on a single standard unit, centrally defined and documented, before any aggregation is performed; otherwise, the unit error propagates silently and is often detected only when the final metric shows an order of magnitude that is clearly out of line with the manager's expectations.

Handling atypical values, or outliers, calls for a more careful discussion than simply applying automatic discard rules. Cases such as physically impossible speeds (typically caused by GPS capture errors), negative or abnormally high fuel consumption (often caused by typos in manually filled spreadsheets), and zero trip times require clearly defined business rules for identifying and handling them—whether by discarding the record or by flagging it for manual review by an analyst. From a statistical standpoint, methods such as the z-score (identifying values that deviate significantly from the mean in terms of standard deviations) or the interquartile range method are widely used in the data quality literature to systematically detect these cases (REDMAN, 2001). It is essential, however, to sound an important warning: not every outlier is an error. A vehicle on a route test, a genuine operational emergency, or a systematic defect in a specific model of tracking device—one that produces artificially inflated speed readings across the entire fleet equipped by that manufacturer—are all examples of outliers that carry genuine, valuable analytical information; discarding them automatically, without prior investigation, risks masking real operational problems instead of simply removing statistical noise. The most defensible rule of thumb, adopted in the case study in Section 1.6, is that a one-off, isolated outlier is likely a capture error, whereas a recurring, concentrated outlier—tied to the same vehicle, the same driver, or the same equipment model—is likely a legitimate signal that deserves dedicated investigation rather than automatic removal.

Record deduplication becomes necessary when the same piece of business information reaches the BI environment through two or more different channels—for example, a fueling event recorded simultaneously by the partner station's system and by the driver's mobile app. The keys used to identify duplicates usually combine the vehicle's license plate, the date and time of the event, and the monetary amount involved, applying a time tolerance window (typically a few minutes) to accommodate small clock-synchronization discrepancies between systems.



Finally, data enrichment consists of matching the extracted raw data against reference tables that add further business context—for example, mapping a raw geographic coordinate to the name of a previously registered route, or mapping a vehicle’s license plate to an organizational cost center. This step is what allows data originally captured in a purely technical format (coordinates, codes) to take on a meaning that business users can immediately interpret in the final visualization layer.

Load

The final step of the ETL process consists of loading the transformed data into the final structure of the analytical model—the star schema discussed in detail in Section 1.2, a dimensional modeling format established by Kimball and Ross (2013) and widely adopted in both the literature and industry practice for its combination of conceptual simplicity and high analytical query performance. In this step, the central fact table receives the quantitative metrics of the business process at the previously defined grain, while the surrounding dimension tables receive the corresponding qualitative attributes, already conformed—that is, already semantically unified across the different source systems, so that, for example, the same vehicle is represented by a single row in the vehicle dimension, regardless of how many different sources contributed data about it.

This architectural design allows business users, within the BI tool, to freely build analyses combining any mix of dimensions and metrics—“cost per route per month,” “average consumption per driver per quarter”—without the data analyst having to rewrite specific queries or build new intermediate tables for every business question that arises over time. This query flexibility, discussed in greater theoretical depth by Chaudhuri and Dayal (1997) in the broader context of online analytical processing (OLAP) systems, is the main technical and economic justification for the comparatively higher up-front investment required to build a star schema rather than simpler modeling alternatives such as a single “flat” table—whose structural limitations are detailed in the comparison in Section 1.7.



Integration with Enterprise Resource Planning (ERP) Systems

The Most Common Integration Points and Risks of Inconsistency

Integration between the BI environment and the enterprise management system usually centers on three main areas of data: financial data, which includes maintenance costs and fuel costs when these are billed centrally to the company; cost center data, used to allocate expenses across different business units; and master records describing vehicles, drivers, and service providers. Although conceptually simple, this integration is one of the highest technical-risk stages of any logistics BI project and therefore deserves a dedicated discussion of its own, along the lines of what the enterprise application integration (EAI) literature has discussed since the early work on enterprise middleware architectures.

The most common risk in this area of integration is inconsistent identifiers across systems—a problem that the discipline of master data management (MDM) treats as one of its central concerns. It is extremely common for the ERP system to identify a given vehicle by an internal asset number used for accounting purposes, while the telematics system identifies the same vehicle by its license plate, and the fuel-log spreadsheet identifies it by an informal nickname used by the drivers themselves—such as “Truck 12.” Without a well-maintained crosswalk table—that is, a mapping table that links, for each relevant business entity, the different identifiers used by each source system—every new integration becomes a silent source of error: records that cannot be correctly matched across systems simply disappear from the final analyses, without any visible alert being raised for the team in charge. That is what makes this type of error particularly dangerous—it does not produce a visible processing failure, but rather a systematic, undetected understatement of the final metrics.

A second recurring risk, of a different nature, is the time lag between an operational event and its formal entry in the ERP system. A maintenance expense is often recorded in the ERP on the date of the corresponding accounting entry—which may occur several weeks after the service



was actually performed, depending on the contracted service provider’s billing cycle. If this time difference is not handled explicitly in the data model—for example, by keeping two separate dates in the fact table, one for the operational date and another for the posting date—cost metrics such as “maintenance cost per trip” can become systematically skewed, assigning to a given month expenses that operationally belong to an earlier period.

Strategies for Dealing with Legacy Systems That Lack a Structured Programming Interface

Not every legacy system that a mid-sized or large transportation operation needs to integrate offers an application programming interface (API) suitable for automated data extraction—and in many mid-sized operations, this unfavorable scenario is the rule rather than the exception, especially when it comes to management systems deployed more than a decade ago, often with no active support from the original vendor. Given this scenario, three practical strategies present themselves as viable alternatives, each with a different profile in terms of cost, robustness, and long-term sustainability.

Scheduled batch export is the least technically complex strategy: when the legacy system offers no API but does allow reports to be exported in formats such as CSV or spreadsheets, this export can be automated through a scheduled job or a dedicated script, treating the resulting file as a formal data source in the ETL pipeline, subject to structural validation on every new load. It is a quick solution to implement, but it depends on the export format of the legacy system remaining stable over time.

Reading directly from the underlying database, when authorized by the legacy system’s vendor, is a technically more robust alternative than manual export, since it eliminates dependence on an intermediate report-generation process. This strategy does, however, require extra care not to overload the transactional database—which is still in active use by operations—with potentially heavy analytical queries, which once again underscores the importance of the staging layer discussed in Section 1.2 as a buffer between extraction and downstream analytical processing.

Robotic process automation (RPA) is a third alternative, reserved for cases in which the legacy system offers only a visual interface designed for human use, with no data export mechanism



or direct database access. In this strategy, a software robot mimics human navigation of the system's screens to extract the desired information. This solution should be deliberately treated as temporary rather than structural, for three practical reasons well documented in the process automation literature: first, its fragility—any visual update to the source system, such as a button moving or a screen layout changing, tends to break the robot without warning; second, the cost of ongoing maintenance—unlike an API or database integration, which tends to remain stable over time, an RPA bot requires recurring monitoring and adjustment, shifting the data team's effort toward infrastructure maintenance and away from value-adding analytical work; and third, limited auditability—it is structurally harder to demonstrate and guarantee the integrity of data extracted by simulating screen interactions than of data extracted through a structured, deterministic query. For these reasons, any use of RPA should always come with an explicit plan to replace it with a more robust form of integration as soon as the legacy system's vendor makes a suitable technical alternative available.

Regardless of which of these three extraction strategies is adopted, maintaining a centralized, version-controlled crosswalk table—that is, one kept under change control, so that changes to identifier mappings can be tracked over time—is, in the hands-on experience reported in this chapter, the highest-return investment for achieving a sustained reduction in integration inconsistencies over time, outweighing in impact any one-off improvement to a single, isolated extraction source.

Data Modeling Best Practices for Logistics

Standardized Naming for Fields and Metrics

A recurring problem in BI environments that grow organically, without a formal data governance process in place from the start, is the uncontrolled proliferation of different names for the same business concept—situations in which fields such as “km_driven,” “distance_km,” and “KM_TOTAL” coexist within the same organization, and not infrequently within the same data model, causing confusion over which field should actually be used in a given analysis and increasing the



risk that the same metric will be calculated in subtly different ways by different analysts. The data management body of knowledge compiled by DAMA International (2017) recommends defining, from the very beginning of any modeling initiative, a formally documented naming convention—one that establishes, for example, standardized prefixes for different field types, a requirement to include the unit of measure explicitly in the field name itself, and the adoption of a single reference language for the entire database—and applying that convention consistently across all dimension and fact tables in the model.

The Data Dictionary as an Operational Asset

The data dictionary—understood here as a document or metadata table that records, for each field in the model, its business definition, its source, its unit of measure, and the transformation rules applied to it throughout the ETL pipeline—should, in the view advanced in this chapter, be treated as a formal operational asset rather than an optional documentation by-product, which is often pushed to the back burner under the delivery-deadline pressure of a BI project. In a data team's day-to-day work, it is the data dictionary that allows a new analyst joining the team to understand the existing model without relying solely on the tacit knowledge accumulated by whoever originally built it—knowledge that, without formal documentation, tends to be lost entirely the moment that person leaves the organization. It is also the data dictionary that prevents the same business metric from being calculated in divergent ways by different people over time, one of the clearest symptoms of data governance immaturity in any organization, regardless of its size or industry.

Case Study — Building a Fleet Data Model Step by Step

The case study methodology, as formalized by Yin (2015), is particularly well suited to investigating contemporary organizational phenomena in their real-world context, especially when the



boundaries between phenomenon and context are not clearly evident—as is precisely the case when implementing a BI environment within a transportation operation that is up and running. The data presented below are real, drawn from a mid-sized transportation operation in which the author was directly involved, but they have been anonymized and partially scaled to protect the confidentiality of the company involved—names, absolute values, and specific identifiers have been changed, while preserving the order of magnitude and the actual structure of the problem, so that the conclusions drawn from the case remain valid and transferable to similar organizational settings.

The Starting Point

When the project began, the operation under study had forty light- and heavy-duty vehicles serving regional distribution routes and relied on three relevant data sources: a vehicle tracking system with its own programming interface, a fuel-log spreadsheet filled out by hand at partner fueling stations, and the maintenance and cost module of the enterprise ERP system. Before the restructuring described here, the monthly close of fleet metrics—cost per kilometer driven, average fuel consumption, vehicle availability—was done entirely by hand in spreadsheets, taking an average of six business days of an analyst’s dedicated time, with an estimated rework rate of around thirty percent, caused by recurring inconsistencies among the three data sources mentioned.

The Restructuring Process, Step by Step

The first step in the restructuring process was to formally define the grain of the main fact table. Following the recommendation discussed in Section 1.2, per-trip granularity was chosen as the primary grain (materialized in the `fact_trip` table), complemented by an aggregate per-trip telematics fact table (`fact_trip_telematics_summary`) to enable driving-behavior analyses without exposing the raw volume of location events to end users of the visualization layer.



The second step was to formally map the data sources and their respective identifiers, building a crosswalk table linking the vehicle's license plate—the key used by both the telematics system and the fuel-log spreadsheet—to the corresponding asset number used internally by the ERP system. This step alone turned out to be responsible for eliminating most of the monthly-close discrepancies previously identified: a retrospective analysis showed that roughly seventy percent of the cases historically classified as “vehicle not found” in the manual reports actually stemmed from mismatched identifiers across systems, not from any real absence of the underlying data—a finding that bears out the emphasis placed in Section 1.4 on the central importance of the crosswalk table as an integration governance tool.

The third step involved implementing a proper staging layer, in which each of the three sources began feeding its own area daily through incremental loads—that is, capturing only the records that were new or changed since the previous run, instead of fully reprocessing the database on every cycle.

The fourth step consisted of defining and implementing the transformation rules, including outlier handling—speed readings above one hundred kilometers per hour began to be flagged for manual review rather than discarded automatically, in direct keeping with the caveat discussed in Section 1.3 about the risk of prematurely discarding potentially informative outliers—and the standardization of units of measure, with all fuel consumption converted to liters and all distances converted to kilometers, regardless of the original unit of capture.

The fifth step was the star schema modeling itself, building the fact_trip fact table—containing the distance, time, consumption, and cost metrics—surrounded by the dim_vehicle, dim_driver, dim_route, and dim_time dimensions, fully following the design presented in Section 1.2.

The sixth and final step was connecting the analytical model to the visualization tool, building two dashboards tailored to different user profiles: an operational dashboard, refreshed daily at the vehicle level, for the fleet operations team; and a management dashboard, refreshed monthly at the cost-center level, for the operation's senior leadership.



Observed Results

An evaluation carried out three months after the new environment went live—a period deliberately chosen to allow the process to stabilize and the team to get past the initial learning curve—revealed a significant improvement in the efficiency metrics of the monthly close: the time required dropped from six business days to less than one business day, and the rework rate due to inconsistencies fell from roughly thirty percent to below five percent, concentrated mostly in legitimate cases of outlier review—and no longer in cross-source integration errors, which had been the predominant cause of rework in the previous scenario.

As the analytical takeaway of this case study, it is important to emphasize that the gains observed did not come from the visualization tool itself, but from the data architecture built behind it. Without the crosswalk table and the formal separation into layers discussed throughout this chapter, the very same dashboard would have remained structurally dependent on a manual reconciliation process repeated every monthly close cycle. This result empirically illustrates the central argument of this chapter: investment in data architecture must come before investment in visualization tools, not the other way around—a reversal of priorities that, in the author’s experience, accounts for a substantial share of the BI projects that fail to deliver sustainable value despite having technically sophisticated visualization tools.

Summary and Data Architecture Maturity Checklist

This chapter has covered the fundamentals that underpin any sound Business Intelligence initiative in transportation logistics: an understanding of the structural complexity inherent in the industry’s operational data, grounded in the information systems and supply chain literature; the layered architecture that deliberately separates raw data from analytically usable data, grounded



in Inmon’s (2005) seminal work on data warehousing and Kimball and Ross’s (2013) work on dimensional modeling; the ETL processes that put this transition between layers into practice, with special attention to the data quality dimensions discussed by Wang and Strong (1996) and Redman (2001); the specific precautions required when integrating with ERP systems and with legacy systems lacking structured interfaces; and the naming, documentation, and governance best practices that keep the BI environment sustainable as the operation grows in volume and complexity.

Table 2 systematically summarizes the structural differences between “flat” modeling—an approach that, despite its apparent initial simplicity, tends to prove unsustainable as the operation matures—and the star schema discussed in detail in Section 1.2.

Table 2 – Comparison of “flat” modeling and the star schema

Criterion	“Flat” modeling (single table)	Star schema
Initial ease of construction	High	Medium
Performance at high volumes	Low (excessive redundancy)	High
Flexibility for new analyses	Low	High
Long-term maintenance	Difficult (changes propagate errors)	More organized (isolated dimensions)
Fit with modern BI tools	Limited	Native

Source: prepared by the author.

The checklist presented in Table 3 can be used as a self-assessment tool for the data architecture maturity of a transportation operation, regardless of its size or current stage of technological maturity, serving both as a diagnostic tool for teams starting a BI project and as a periodic audit instrument for environments already in production.

Table 3 – Data architecture maturity checklist for logistics BI

#	Checklist item	Met?
1	There is a clear separation between the staging layer and the analytical model	
2	The grain of the main fact table is formally defined and documented	
3	There is a centralized crosswalk table for identifiers across systems	
4	Units of measure are standardized before loading into the analytical model	



5	There are documented rules for handling outliers	
6	The model follows a star schema (fact + dimensions), not a single “flat” format	
7	There is an up-to-date data dictionary accessible to the team	
8	Visualization tools consume only the analytical model, never the operational sources directly	
9	There is a load routine (manual or automated) with a defined, monitored frequency	
10	There is an audit/reprocessing process based on the staging layer	

Source: prepared by the author.

The more items on this checklist an operation actually meets, the less recurring rework its data team tends to face and the more reliable the management metrics it produces—and it is precisely this reliability, backed by a technically sound data architecture, that turns a visually appealing dashboard into an effective decision-support tool rather than a technological façade devoid of analytical substance. The following chapters of this book build directly on the architectural foundation laid here, moving on to define key performance indicators (KPIs) specific to transportation logistics and to build analytical dashboards geared toward management decision-making.

QUICK GLOSSARY

ETL: Extract, Transform, and Load—the process of moving and preparing data from the operational sources to the analytical model.

Staging: intermediate layer where raw data is stored without transformation before moving on to the analytical model.

Star schema: dimensional data model with a central fact table (metrics) surrounded by dimension tables (descriptive attributes).

Fact table: central table of a star schema, which stores the quantitative metrics at the grain defined for the model.

Dimension table: table that describes the qualitative attributes of a business entity (vehicle, driver,



route, time).

Grain (granularity): the level of detail that one row of the fact table represents (per trip, per day, per event).

Slowly changing dimension (SCD): modeling technique for handling dimension attributes that change over time, either preserving or discarding the history of the change.

Crosswalk table: mapping table between the different identifiers used by different systems for the same entity.

RPA: robotic process automation; mimics human interaction with a system's interface to extract or enter data.

MDM (Master Data Management): management discipline focused on unifying and governing master data shared across multiple systems.

REFERENCES

BALLOU, R. H. Gerenciamento da Cadeia de Suprimentos/Logística Empresarial [Business Logistics/ Supply Chain Management]. 5th ed. Porto Alegre: Bookman, 2006.

BRASIL. Lei nº 13.103, de 2 de março de 2015 [Law No. 13,103 of March 2, 2015]. Provides for the practice of the professional driver occupation. Diário Oficial da União [Official Gazette of the Federal Government], Brasília, DF, 2015.

CALDEIRA, J. Data Warehouse: conceitos e modelos com estudo de caso [Data Warehouse: Concepts and Models with a Case Study]. 2nd ed. Lisbon: Edições Sílabo, 2017.

CHAUDHURI, S.; DAYAL, U. An overview of data warehousing and OLAP technology. ACM SIGMOD Record, vol. 26, no. 1, pp. 65–74, 1997.

CHRISTOPHER, M. Logistics and Supply Chain Management. 5th ed. Harlow: Pearson, 2016.



CODD, E. F. A relational model of data for large shared data banks. *Communications of the ACM*, vol. 13, no. 6, pp. 377–387, 1970.

DAMA INTERNATIONAL. DAMA-DMBOK: Data Management Body of Knowledge. 2nd ed. Basking Ridge: Technics Publications, 2017.

DAVENPORT, T. H.; HARRIS, J. G. *Competing on Analytics: The New Science of Winning*. Boston: Harvard Business School Press, 2007.

ELMASRI, R.; NAVATHE, S. B. *Fundamentals of Database Systems*. 7th ed. Boston: Pearson, 2015.

GOLFARELLI, M.; RIZZI, S. *Data Warehouse Design: Modern Principles and Methodologies*. New York: McGraw-Hill, 2009.

INMON, W. H. *Building the Data Warehouse*. 4th ed. Indianapolis: Wiley, 2005.

KIMBALL, R.; CASERTA, J. *The Data Warehouse ETL Toolkit: Practical Techniques for Extracting, Cleaning, Conforming, and Delivering Data*. Indianapolis: Wiley, 2004.

KIMBALL, R.; ROSS, M. *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling*. 3rd ed. Indianapolis: Wiley, 2013.

BRAZILIAN ASSOCIATION OF TECHNICAL STANDARDS (ABNT). ABNT NBR ISO 9001:2015 — Sistemas de gestão da qualidade — Requisitos [Quality Management Systems — Requirements]. Rio de Janeiro: ABNT, 2015.

PBQP-H. SiAC — Sistema de Avaliação da Conformidade de Empresas de Serviços e Obras da Construção Civil [Conformity Assessment System for Civil Construction Service and Works Companies]. Brasília: Ministry of Cities, [n.d.].

REDMAN, T. C. *Data Quality: The Field Guide*. Boston: Digital Press, 2001.

VASSILIADIS, P. A survey of extract–transform–load technology. *International Journal of Data Warehousing and Mining*, vol. 5, no. 3, pp. 1–27, 2009.



WANG, R. Y.; STRONG, D. M. Beyond accuracy: what data quality means to data consumers. *Journal of Management Information Systems*, vol. 12, no. 4, pp. 5–33, 1996.

YIN, R. K. *Estudo de Caso: Planejamento e Métodos* [Case Study Research: Design and Methods]. 5th ed. Porto Alegre: Bookman, 2015.

