Chapter 2

Data Warehouse Philosophy

Introduction

A data warehouse is an asset of an enterprise and exists for the benefit of an entire enterprise. It does not exist for the benefit of a single entity (e.g., business unit, individual customer, etc.) to the exclusion of all others in the enterprise. As such, data in a data warehouse does not conform specifically to the preferences of any single enterprise entity. Instead, a data warehouse is intended to provide data to the entire enterprise in such a way that all members can use the data in the warehouse throughout its lifespan.

Traditionally, an information system succeeds by satisfying specific requirements of a specific customer. A data warehouse, however, succeeds by satisfying the data needs of an entire enterprise, not just one entity. The “one size fits all” approach to data positions a data warehouse to fail in its mission to provide data to the whole enterprise. All data warehouses would fail in this mission were it not for the foundational principles created by the data warehousing pioneers and visionaries Ralph Kimball and Bill Inmon.

In the 1990s, Kimball and Inmon created and documented the concepts and principles of data warehouses, which today are the foundation of all data warehouses. These concepts and principles will not immediately equip a reader to design and develop a data warehouse; however, they will equip a reader to understand the reasons and intentions underlying data warehouse design. For that reason, these
concepts and principles are collectively known as the data warehouse philosophy. The concepts and principles within the data warehouse philosophy guide the design and development of a data warehouse.

Inclusion of all elements of the data warehouse philosophy is not mandatory for the success of a data warehouse. Awareness of the elements of this philosophy, however, increases its success and value. A data warehouse designer may choose to include or exclude elements of the data warehouse philosophy. Such decisions should be made from the context of cognitive understanding of the philosophy.

The elements of the data warehouse philosophy are explained in the following sections. Those elements are:

- Enterprise Data
- Subject Orientation
- Integration
- Nonvolatility
- Time Variant
- Single Version of the Truth
- Long-Term Investment and Return on Investment (ROI)

**Enterprise Data**

A data warehouse should include data that is applicable to the enterprise. The value and relevance of a data warehouse is rooted in that data. If members of the enterprise perceive as superfluous or irrelevant the data in a data warehouse, those same members will cast that perception onto the data warehouse. This principle is not as restrictive as it seems. Frequently, data that is acted on by a single business unit is also relevant to the remainder of the enterprise. For example:

- The Accounting Department uses tax codes; members of other departments understand the relevance of tax codes to the enterprise.
- The Manufacturing Department uses part numbers; members of other departments understand the relevance of part numbers to the enterprise.

Tax codes are not directly applicable to the Manufacturing Department; however, they understand tax codes are relevant to the enterprise. Part numbers may not be directly applicable to the Accounting Department; however, they understand that part numbers are relevant to the enterprise.

Data that is not relevant to the enterprise is localized in its relevance. Localized relevance renders data irrelevant to the enterprise. For example:

- The Accounting Department maintains a list of notary publics in its office.
Data Warehouse Philosophy

- The Manufacturing Department keeps a list of machinists who own and use their own tools, including a designation for United States and metric tools.

The names and availability of notary publics are handy in the Accounting Department, but irrelevant to the rest of the enterprise. Likewise, a list of machinists and their tools helps the shop foreman assign individual tasks, but has little relevance to the rest of the enterprise.

Subject Orientation

Data in a data warehouse is organized around the business subjects of the enterprise. Operational data is organized by its physical manifestations, including file names, job schedules, and application dependencies. A data warehouse does not present data, which reflects the physical manipulation of operational data. Instead, a data warehouse presents data, which reflects major subject areas within the enterprise.

- **Business Entities**
  - Customers
  - Vendors
  - Agents

- **Business Processes**
  - Sales
  - Receiving
  - Manufacturing
  - Distribution

Subject orientation of data allows a data warehouse to maintain its overall architecture throughout its lifespan. Individual data elements may change in the enterprise and in the data warehouse. The subject orientation of a data warehouse enables a data warehouse to absorb inevitable changes without drastic changes to its architecture.

Data Integration

The data in a data warehouse is presented in a uniform manner. Uniformity allows data warehouse customers to query data across subject areas without traversing through data translations or look-ups from other data sources. By integrating its data, a data warehouse presents a consistent and seamless statement of the enterprise, which relieves data warehouse customers of the need to reconcile differences.
and inconsistencies within data from disparate business areas. Data integration occurs in multiple ways, which can be combined into the following three groups:

- **Form**
- **Function**
- **Grain**

**Form**

Data form includes the types and layouts of data.\(^5\) These are the way data is expressed. Disparate business units may express similar data elements in different ways. For example:

- Money can be expressed as currency or integer data types.
- Phone numbers can be expressed as (123) 123-1234 or 123-123-1234 or 123.123.1234.
- Names can be expressed as First Last or Last, First.

In a data warehouse, the disparate expressions of similar data elements (e.g., money, phone number, names, etc.) are integrated into a single form, which creates consistency within a data warehouse. This helps data warehouse customers query across business subjects.

**Function**

Function includes the substance and meaning of the data within the data element. Codes and cryptic values often differ between business units and must be reconciled so the entire organization can leverage these codes and values. For example:

- Part Status = 32B: In the Manufacturing Department, 32B means the manufacturing part is on back-order. This translation needs to be provided to others in the enterprise.
- Closing Code = 32B: In the Accounting Department, 32B means the financial statement is out of balance and cannot be closed. This translation needs to be provided to others in the enterprise.

Business units may use the same code or value with two distinct and separate meanings. For example, in the Accounting Department, 32B means “Out of Balance,” while in the Manufacturing Department, 32B is a jacket size. Business units may use different codes or values with the same meaning. For example, the Marketing Department refers to a 30-second TV spot as a promotion, while the Distribution and Logistics Department refers to the same 30-second TV spot as an
Data Warehouse Philosophy

advertisement. In such cases, disparate data from business units must be integrated into one data element, which expresses the function of both.

**Grain**

Grain refers to the unit of measurement at which data is expressed. Business units may store data using different units of measurement:

- Purchasing measures product by the barrel.
- Transportation measures product by the shipload.
- Sales measures product by the gallon.

In this scenario, a data warehouse will reconcile these different units of measurement, which will allow the integration of data from the Purchasing, Transportation, and Sales business units.

Grain can also refer to a hierarchical level. Individual people, objects, and events are organized into hierarchical groups. Business units may store data using different hierarchical groupings of people, objects, and events:

- A captain commands a vessel in the Third Fleet.
- Captain Roy P. Jones commands a vessel in the Third Fleet.
- A captain commands the USS Hawkins.
- Captain Roy P. Jones commands the USS Hawkins.

In this scenario, a data warehouse will reconcile these different levels of hierarchical grouping, which will allow the integration of data from the Personnel Department and the Third Fleet.

Grain of data has two physical implications for a data warehouse. First, fine grain data expresses more detailed information, but at a cost. The increased detail consumes increased resources to capture, store, and retrieve. Second, a data warehouse cannot provide data to customers at a grain lower than the grain at which it is stored.

A data warehouse must integrate the Form, Function, and Grain of data from disparate business units. Once integrated, data warehouse customers can traverse data within business subjects from across the enterprise.

**Nonvolatility**

Data, once written to a data warehouse, is never deleted or updated. Operational applications manipulate data to reflect only the current state of a business unit. A data warehouse reflects both the historical and current state of the enterprise by
inserting new rows. A data warehouse retains historical rows as well as the most recent rows, which allows a data warehouse to present data in the context of the past and the present. Nonvolatility allows a data warehouse to express the enterprise across time, by retaining that data.

**Time Variant**

A data warehouse expresses the events of the enterprise across time. Nonvolatile historical data allows a data warehouse to express historical enterprise events in their historical context. For example:

- During the month of January, Fred was the manager of store #1024. January net profit for the store totaled $140,000.
- During the month of February, Alice was the manager of store #1024. February net profit for the store totaled $70,000.
- During the present month, George is the manager of store #1024.

The presence of historical data allows analysis and comparison of these three store managers, even though they occurred at different times. An analyst can ask such questions as:

- What was the profitability of store #1024 when Fred was the manager?
- What was the profitability of store #1024 when Alice was the manager?
- What is the profitability of store #1024 now that George is the manager?

These questions can be answered by translating these questions into a surrogate question based on a simultaneous and coincidental event (e.g., the month).

- What was the profitability of store #1024 in January?
- What was the profitability of store #1024 in February?
- What is the profitability of store #1024 in the current month of March?

Stores will not normally (in fact, rarely) change store managers on a schedule that coincides with the change of the month. So, a business analyst cannot expect to track the performance of store managers by tracking months, expecting each month to represent a different store manager. A data warehouse facilitates the answers to the real questions (e.g., How profitable was each manager?) by allowing a business analyst to track the performance of the managers, regardless of the historical context. The historical data in a data warehouse provides answers to questions of historical events and conditions in this context, based on the events or conditions. A data warehouse does not require its customers to translate historical
questions into their historical context because the data in a data warehouse has already framed its data in its historical context.

Time variant data allows a data warehouse to express the enterprise as of a moment in time. That moment in time has a grain. A moment in time can be expressed as a millisecond, minute, hour, day, week, month, year, etc. In the context of digital versus analog, Time is analog. Information systems, however, can only capture Time digitally. Every expression of Time, therefore, is a digital representation of analog Time; hence, Time expressed as a millisecond, minute, hour, day, week, month, year, etc.

Historical data allows a data warehouse to express the enterprise from three different historical contexts:

- **As It Was**: In this context, a data warehouse can express states of the enterprise at the moment they occurred, including the moment the state began and ended. For example:
  - Fred was the manager of store #1024 during the month of January.
  - Alice was the manager of store #1024 during the month of February.
  - George is the manager of store #1024 now.

- **As It Is**: In this context, a data warehouse can superimpose the current (i.e., now) state of the enterprise over the entire history of the enterprise. All historical data is still present in the data warehouse, but not used in the result set returned to the data warehouse customer. For example:
  - George (the current manager) has always been the manager of store #1024.

- **As If Nothing Changed**: In this context, a data warehouse can superimpose a historical state of the enterprise over subsequent periods of the enterprise. All current (i.e., now) data is still present in the data warehouse, but not used in the result set returned to the data warehouse customer. For example:
  - Fred was the manager of store #1024 during the month of January.
  - Alice was the manager of store #1024 during the month of February.
  - Alice (not George) is the manager of store #1024 now.

Ralph Kimball authored these three variations of Time Variance. He named them Type 1, Type 2, and Type 3. These three names have since become part of the data warehousing lexicon.

- **Type 1 (As it is)**: Cast all of history so the enterprise looks as it does now.
- **Type 2 (As it was)**: Express historical data as it was, with each data value as of its moment in history, retaining its context in time.
- **Type 3 (As if nothing changed)**: Cast the enterprise to look as if a change had not occurred.
The retention of nonvolatile historical data allows a data warehouse to express the enterprise within a historical context. This Time Variant principle is a significant difference from operational applications, which function in the now, rather than the past.

One Version of the Truth

For every question that can be answered by data, an enterprise will derive a myriad answers. For example:

Q: How many widgets were assembled?
   A: Total number of widgets assembled — 32,000
   A: Total widgets net of scrap — 31,195
   A: Total widgets adjusted by Activity Costing — 32,120
   A: Total widgets approved by Quality Control — 31,148

These different answers illustrate the confusion that occurs when business units look at a question (How many widgets were assembled?) and actually see more than just that one question.

Q: How many widgets were physically assembled?
Q: How many widgets were successfully assembled?
Q: How much assembly activity occurred in conjunction with the widgets?
Q: How many widgets were assembled and approved by Quality Control?

A data element stores the answer to a question. The question is the definition of that data element. A data warehouse must define every data element so that all members of the enterprise will associate one and only one question with that data element. Having narrowed a data element down to one and only one question, a data warehouse must also provide one and only one answer to the question posed by that data element. By doing so, a data warehouse provides the truth (i.e., the true answer to the question posed by a data element) and only one version of that truth.16

The One Version of the Truth principle allows a data warehouse to express the entire enterprise. When all members of an enterprise look at a data element with a single understanding of its meaning, then members of the enterprise can use a data warehouse as a shared point of communication across the enterprise.
Long-Term Investment

A data warehouse achieves its greatest ROI through longevity and stability. As the number of subject areas integrated into a data warehouse increases, a data warehouse increases its expression of the enterprise. As time variant history accumulates in a data warehouse, it increases its ability to answer historical questions. A data warehouse must, therefore, be designed and developed as a long-term investment.

A data warehouse team cannot build the entire data warehouse in a single project. The cost would be too high and the delivery schedule would be too slow. Instead, a data warehouse begins with one or two business subjects (e.g., sales, transportation, manufacturing, etc.). Then, each subsequent data warehouse development effort adds another business subject, or a subset of a large or complex subject. Each individual data warehouse project should last only six to nine months. When the duration of an individual data warehouse project exceeds nine months, management typically begins to question the ROI of the project. As multiple individual data warehouse projects integrate multiple business subjects into a single data warehouse, that data warehouse presents a picture of the enterprise, a picture which becomes more complete and comprehensive as each business subject is added to that data warehouse. A data warehouse, therefore, is a long-term investment with a long-term horizon. In fact, a data warehouse may never express the entire enterprise. The success of a data warehouse is not its ability to express the entire enterprise; rather, the success of a data warehouse is its ability to return value to the enterprise using the business subjects included in that data warehouse.

The very first data warehouse project of an enterprise defines the enterprise-level architecture of the data warehouse. The decisions made during the first data warehouse project will lay the foundation for all subsequent data warehouse projects within the enterprise. Physically, these decisions will lay the foundation for the platforms and infrastructures that will be the data warehouse. Logically, these decisions lay the foundation for the subject areas, entities, attributes, and processes as they are captured in a data warehouse. These architectural and foundational decisions will enable, or prevent, the data warehouse and its customers as they include new and additional subject areas in subsequent development efforts. The long-term nature of a data warehouse means the “return” of a data warehouse exists significantly beyond the “investment.” If done correctly, the investment should be of a short duration, and the return should extend for years, if not decades.

References

4. Inmon and Hackathorn, *Using the Data Warehouse*.
6. Inmon and Hackathorn, *Using the Data Warehouse*.
12. Inmon and Hackathorn, *Using the Data Warehouse*.
15. Ibid.