05

Multidimensional Modeling - Inventory

Notice

- **Author**
  - **João Moura Pires (jmp@di.fct.unl.pt)**

- **This material can be freely used for personal or academic purposes without any previous authorization from the author, only if this notice is maintained with.**

- **For commercial purposes the use of any part of this material requires the previous authorization from the author.**

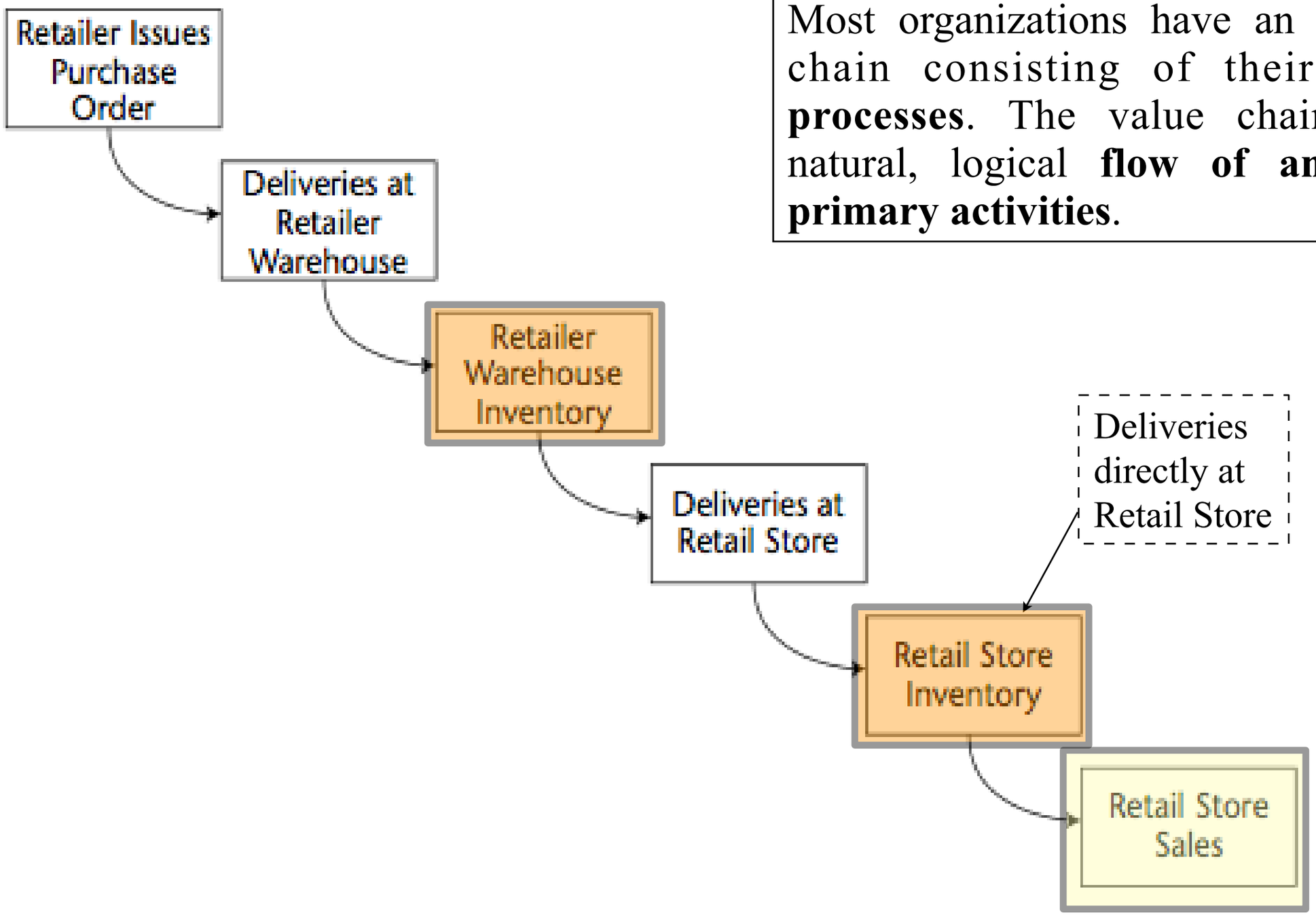
Bibliography

- **Many examples are extracted and adapted from**
  - **The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling (Second Edition) - Ralph Kimball, Margy Ross**

Table of Contents

Introduction

Value Chain



Most organizations have an underlying value chain consisting of their **key business processes**. The value chain identifies the natural, logical **flow of an organization's primary activities**.

Inventory Models

- **Inventory Periodic Snapshot**
  - Every day (or at some other regular time interval), we **measure the inventory levels of each product and place them as separate rows** in a fact table. These periodic snapshot rows appear over time as a series of data layers in the dimensional model.
- **Inventory Transactions**
  - We **record every transaction** that has an impact on inventory levels as products move through the warehouse.
- **Inventory Accumulating Snapshot**
  - We build one fact table row for each product delivery and update the row until the product leaves the warehouse.

[Kimball, 2002]

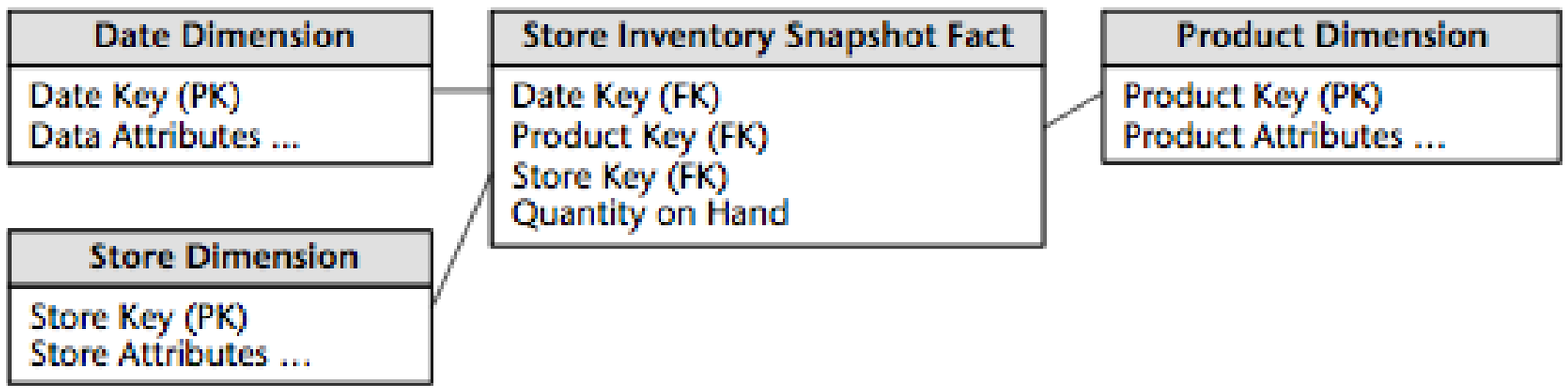
Inventory Periodic Snapshot

Four-step process for designing dimensional models

- **Motivation**
  - Optimized inventory levels in the stores can have a major impact on chain profitability. Making sure the **right product** is in the **right store** at the **right time** minimizes out-of-stocks (where the product isn't available on the shelf to be sold) and reduces overall inventory carrying costs.
- **Process and granularity**
  - The retailer needs the ability to analyze **daily quantity-on-hand inventory levels** by **product** and **store**.
  - **Dimensions**: Date, Product, Store
  - **Facts:** Quantity on hand

[Kimball, 2002]

Inventory Periodic Snapshot Schema - **Store Level**

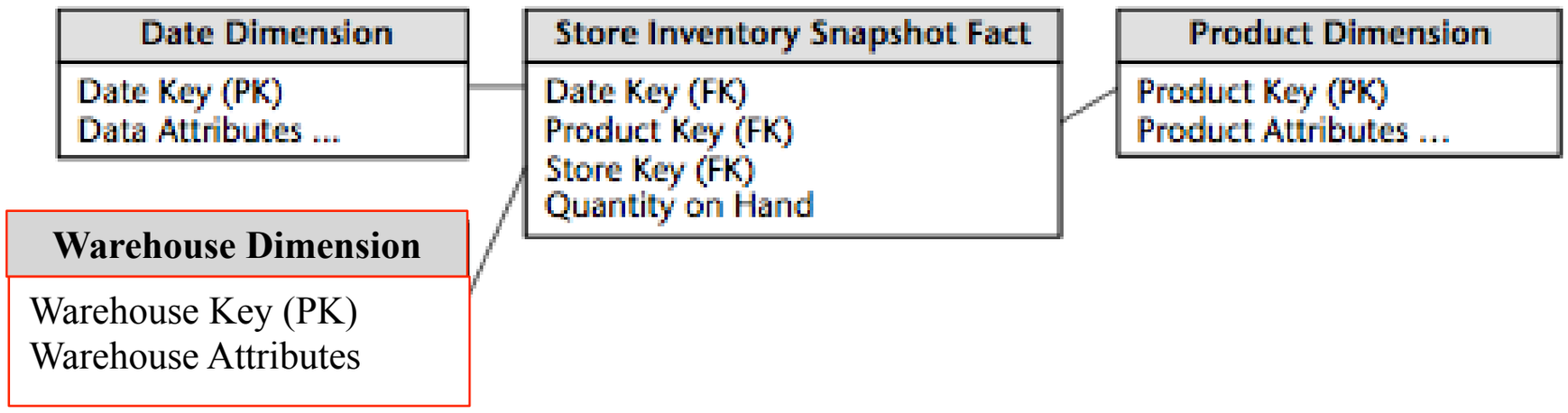


- **To analyze daily quantity-on-hand inventory levels by product and store.**

- **The dimensions are the same on previous schema.** We may want to further decorate these dimension tables with additional attributes that would be useful for inventory analysis.
  - **Product:** Minimum reorder quantity, assuming that they are constant and discrete descriptors of each product stock keeping unit (SKU).
  - **Store**: Frozen and refrigerated storage square footages.

[Kimball, 2002]

Inventory Periodic Snapshot Schema - **Warehouse Level**



- When monitoring inventory levels at the warehouse, normally we do not retain the store dimension as a fourth dimension unless the warehouse inventory has been allocated to a specific store.
- Both schemas may be used

[Kimball, 2002]

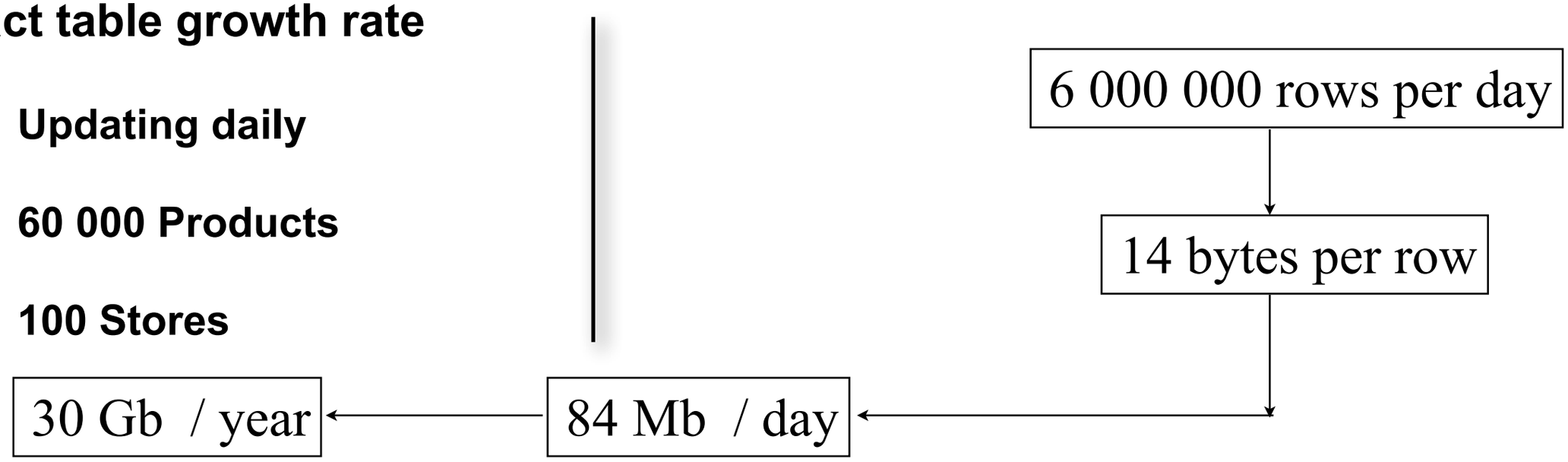
Sparsity and growth rates

- **Sparsity**
  - The **sales** fact table was **reasonably sparse** because only about 10 percent of the products in each of our hypothetical stores actually sold each day. If a **product didn't sell** in a store on a given day, then there was **no row in the fact table** for that combination of keys
  - **Inventory**, on the other hand, **generates dense snapshot tables**. Since the retailer strives to avoid out-of-stock situations where the product is not available for sale, there is a row in the fact table for virtually every product in every store every day:
    - at warehouse level is almost certain that there are movements of products (in or out) for most of products
    - at store level we can choose to register the level for unchanged products

[Kimball, 2002]

Sparsity and growth rates

- **Fact table growth rate**
  - **Updating daily**
  - **60 000 Products**
  - **100 Stores**



- **Compromises**
  - **Reduce the snapshot frequencies** over time. It may be acceptable to keep the last 60 days of inventory at the **daily level** and then revert to less granular **weekly snapshots for historical data**. In this way, instead of retaining 1,095 snapshots during a 3-year period, the number could be reduced to 208 total snapshots (60 daily + 148 weekly snapshots in two separate fact tables given their unique periodicity). Factor of reduction of 5.

[Kimball, 2002]

Semi-additive Facts

- **Quantity on hand (stock level)**
  - Given a product and a date the **Quantity on hand** (**qh**) can be summed-up by stores (or by warehouses)

$$\sum_{store} qh(date, product, store)$$

  - Given a date and a store (or an warehouse) the Quantity on hand (qh) can be summed-up by products:

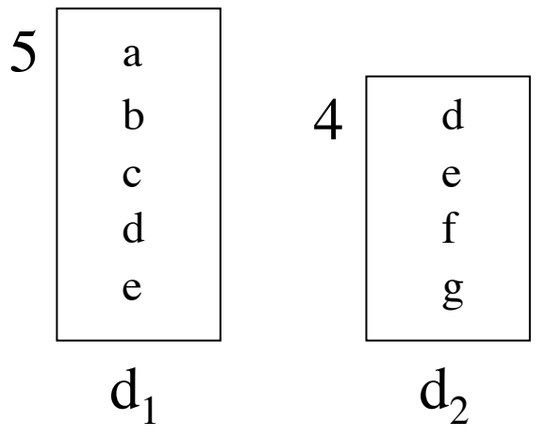
$$\sum_{product} qh(date, product, store)$$

  - However, given a product and a store we cannot summed-up by date. Why?

[Kimball, 2002]

Semi-additive Facts

- **Quantity on hand (stock level)**
  - However, given a product and a store we cannot summed-up by date. Why?



$$5 + 4 = 9$$

The actual number of products in the two days is 7

  - Possible aggregations
    - Average (4, 5) = 4.5
    - Min, Max

> **All measures that record a static level (inventory levels, financial account balances, and measures of intensity such as room temperatures) are inherently nonadditive across the date dimension and possibly other dimensions. In these cases, the measure may be aggregated usefully across time, for example, by averaging over the number of time periods.**

[Kimball, 2002]

Semi-additive Facts: SQL AVG trap

- Unfortunately, you cannot use the SQLAVG function to calculate the average over time. The SQL AVG function **averages over all the rows received by the query**, not just the number of dates.
- For example, if a query requested the average inventory for a cluster of three products in four stores across seven dates (that is, what is the average daily inventory of a brand in a geographic region during a given week), the SQLAVG function would divide the summed inventory value by 84 (3 products x 4 stores x 7 dates). Obviously, the correct answer is to divide the summed inventory value by 7, which is the number of daily time periods.
- The application must **divide the final summed inventory value by the date constraint cardinality**. This can be done with an embedded SQL call within the overall SQL statement or by querying the date dimension separately and then storing the resulting value in an application that is passed to the overall SQL statement.

[Kimball, 2002]

Enhanced Inventory Facts

- Quantity on hand needs to be used in conjunction with additional facts to measure the velocity of inventory movement and develop other interesting metrics
  - **Quantity sold** (or equivalently, quantity depleted or shipped if we're dealing with a warehouse location)
  - **Number of turns** (replacement ratio)
    - **Daily**:
      - Turns = Quantity sold / Quantity on hand
    - **For a period of time**:
      - Turns = Sum (Quantity sold) / Average(Quantity on hand)

[Kimball, 2002]

Enhanced Inventory Facts (cont)

- **Number of days´s supply**
  - The number of estimated days that the actual Quantity on hand is enough assuming the actual Quantity sold per day
  - **Daily:**
    - Number of days´s supply = Quantity on hand / Quantity sold
  - **For a period of time:**
    - Number of days´s supply

      = Quantity on hand at the end of the period / Daily Average(Quantity sold)

[Kimball, 2002]

Enhanced Inventory Facts (cont)

- **To measure the stock financial value**
  - Extended value of the inventory at cost - EVIC. (supplied do the DW)
  - The value at the latest selling price - LSP. (supplied do the DW)
  - Then it will be possible to calculate:
    - Gross Profit = LSP - EVIC ; Selling Price - Cost
    - Gross Margin = Gross Profit / LSP
    - Gross margin return on inventory - GMROI

[Kimball, 2002]

Enhanced Inventory Facts (cont)

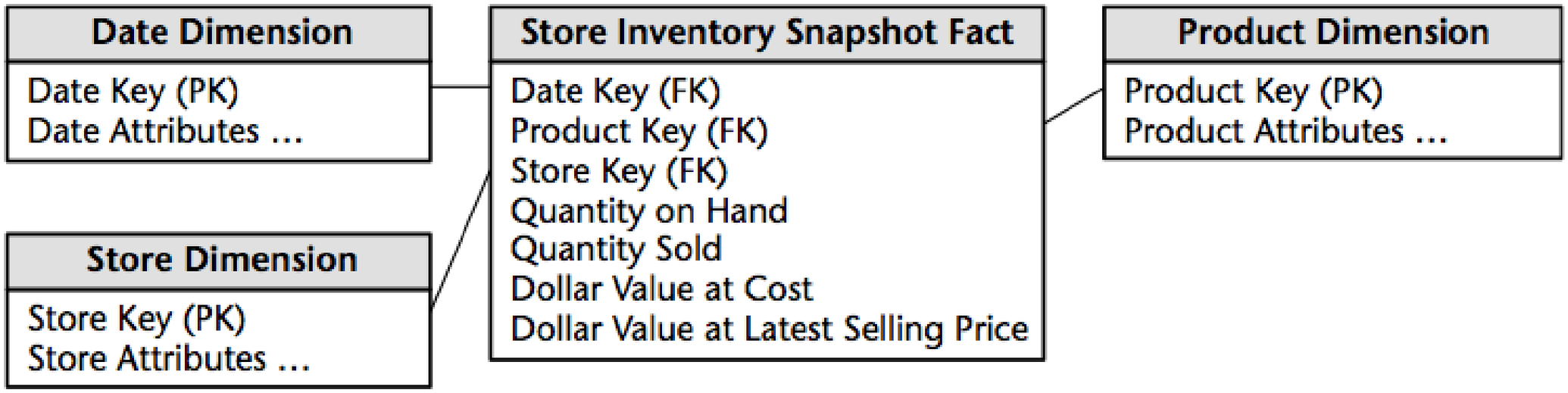
- Gross margin return on inventory - GMROI
  - GMROI = Turns x Gross Margin

$$GMROI = \frac{\text{total quantity sold x (value at latest selling price – value at cost)}}{\text{daily average quantity on hand x value at the latest selling price}}$$

  - A measure of the **effectiveness of our inventory investment**.
  - A **high GMROI** means that we are moving the product through the store quickly (**lots of turns**) and are making good money on the sale of the product (**high gross margin**).
  - A low GMROI means that we are moving the product slowly (low turns) and aren't making very much money on it (low gross margin). The GMROI is a standard metric used by inventory analysts to judge a company's quality of investment in its inventory.

[Kimball, 2002]

Enhanced Inventory Facts - Schema



Notice that quantity on hand is semiadditive but that the other measures in our advanced periodic snapshot are all fully additive across all three dimensions. The quantity sold amount is summarized to the particular grain of the fact table, which is daily in this case. The value columns are extended, additive amounts. We do not store GMROI in the fact table because it is not additive. We can calculate GMROI from the constituent columns across any number of fact rows by adding the columns up before performing the calculation, but we are dead in the water if we try to store GMROI explicitly because we can't usefully combine GMROIs across multiple rows.

[Kimball, 2002]

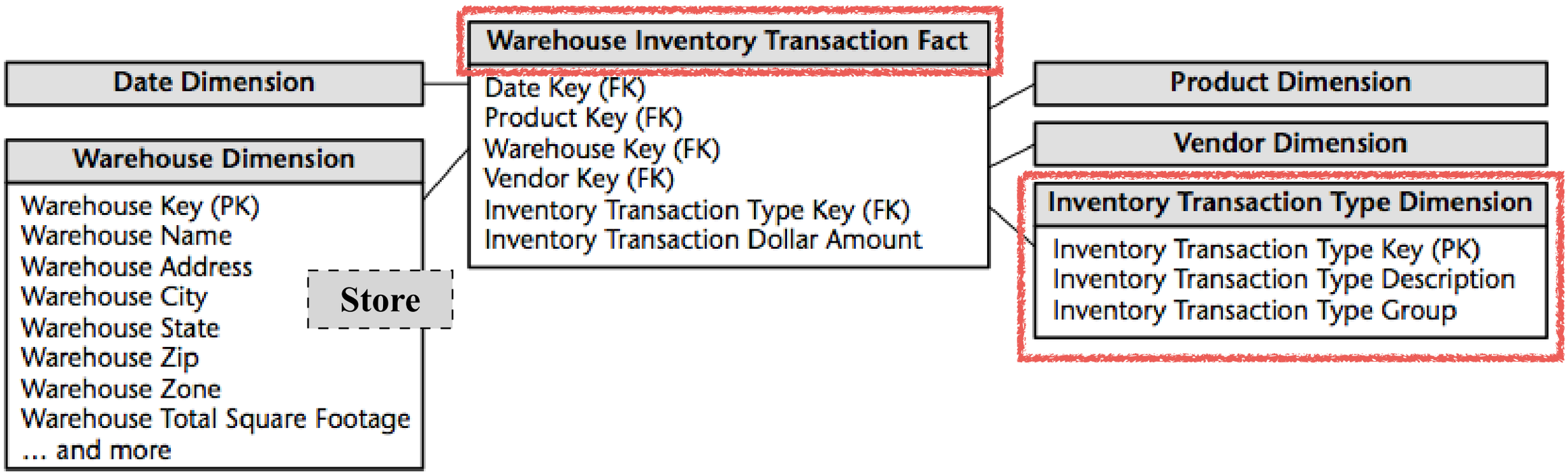
Inventory Transactions

Inventory Transactions

- A second way to model an inventory business process is to **record every transaction that affects inventory**. Some examples:
  - Receive product
  - Place product into inspection hold
  - Release product from inspection hold
  - Return product to vendor due to inspection failure
  - Place product in bin
  - Authorize product for sale
  - Pick product from bin
  - Package product for shipment
  - Ship product to customer
  - Receive product from customer
  - Return product to inventory from customer return
  - Remove product from inventory

[Kimball, 2002]

Inventory Transactions - Schema



Each inventory transaction identifies the date, product, warehouse, vendor, **transaction type**, and in most cases, a single amount representing the inventory quantity impact caused by the transaction. We are assuming that the granularity of our fact table is one row per inventory transaction

[Kimball, 2002]

Inventory Transactions - What for?

Even though the transaction-level fact table is again very simple, it contains the most detailed information available about inventory because it mirrors fine-scale inventory manipulations. The transaction-level fact table is useful for measuring the frequency and timing of specific transaction types. For instance, only a transaction-grained inventory fact table can answer the following questions:

- How many times have we placed a product into an inventory bin on the same day we picked the product from the same bin at a different time?
- How many separate shipments did we receive from a given vendor, and when did we get them?
- On which products have we had more than one round of inspection failures that caused return of the product to the vendor?

Even so, it is impractical to use this table as the sole basis for analyzing inventory performance. Although it is theoretically possible to reconstruct the exact inventory position at any moment in time by rolling all possible transactions forward from a known inventory position

[Kimball, 2002]

Inventory Accumulating Snapshot

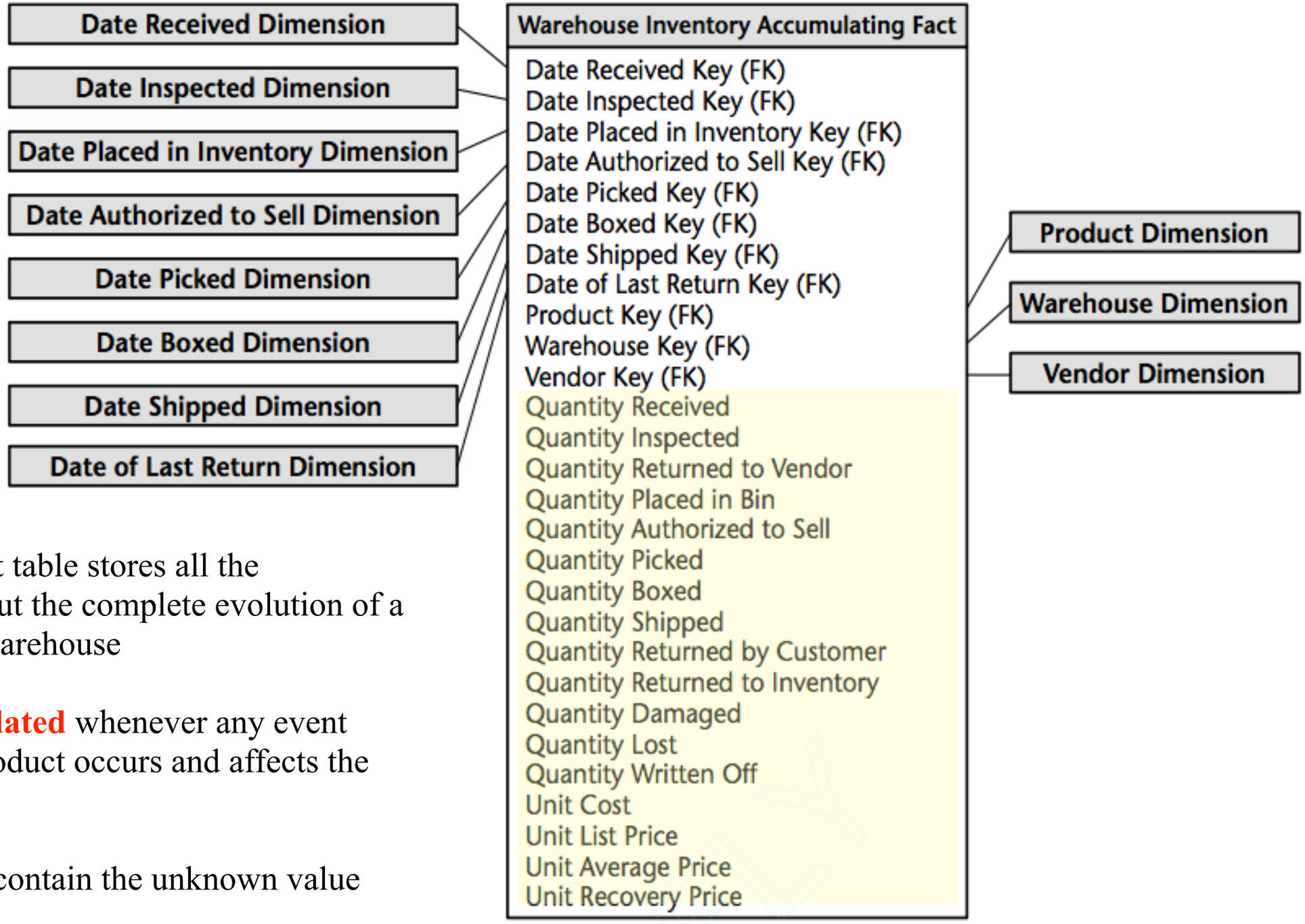
Inventory Accumulating Snapshot

- In this model we place one row in the fact table for a shipment of a particular product to the warehouse. In a single fact table row we **track the disposition of the product shipment until it has left the warehouse**. The accumulating snapshot model is **only possible if we can reliably distinguish products delivered in one shipment from those delivered at a later time**. This approach is also appropriate if we are tracking disposition at very detailed levels, such as by product serial number or lot number.

- Let's assume that the inventory goes through a series of well-defined events or milestones as it moves through the warehouse, such as **receiving**, **inspection**, **bin placement**, **authorization to sell**, **picking**, **boxing**, and **shipping**. The philosophy behind the accumulating snapshot fact table is to provide an **updated status of the product shipment as it moves through these milestones**. Each fact table row will be updated until the product leaves the warehouse.

[Kimball, 2002]

Inventory Accumulating Snapshot - Schema



Each row of fact table stores all the information about the complete evolution of a product in the warehouse

Each row is **updated** whenever any event concerning a product occurs and affects the counters

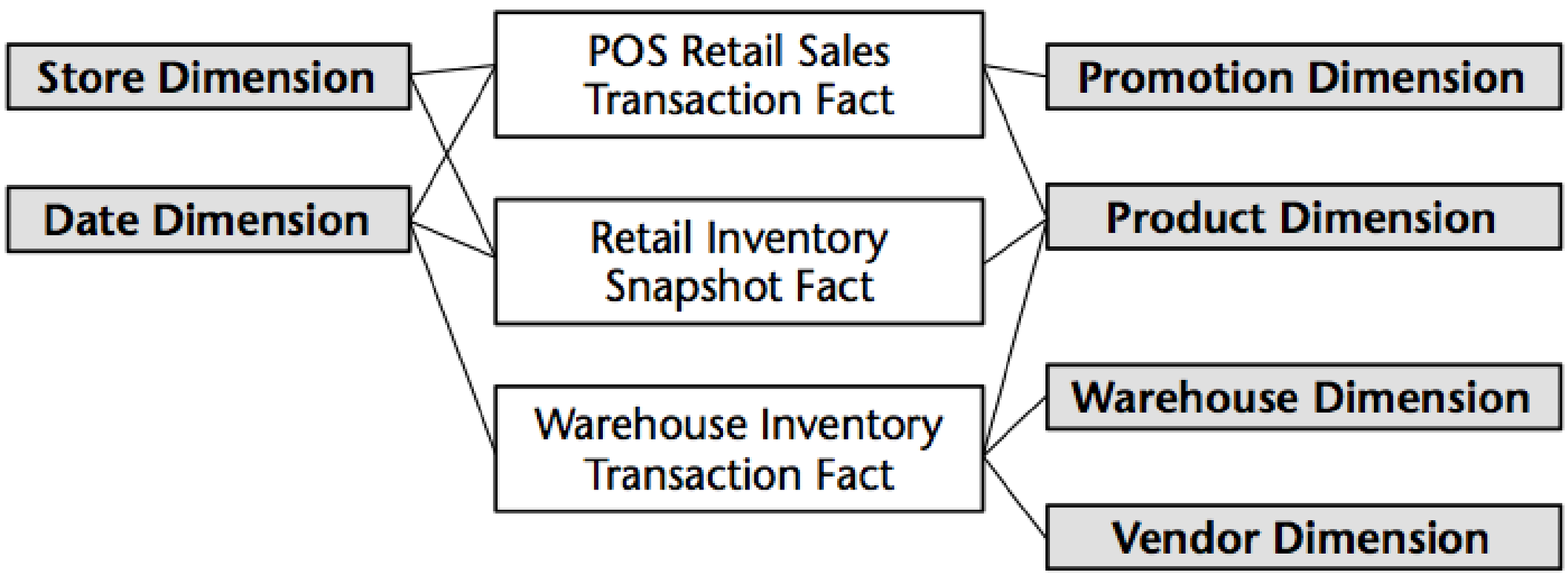
The date views contain the unknown value

[Kimball, 2002]

Integration

Integration

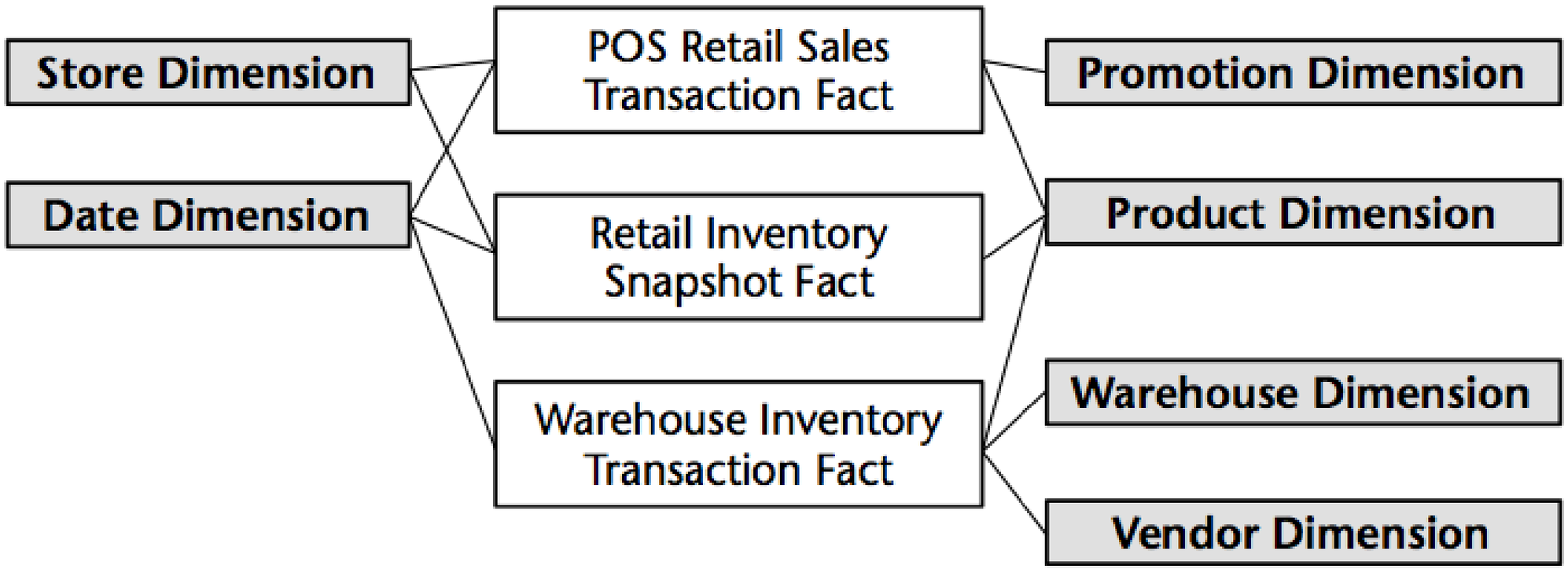
- While separate fact tables in separate data marts represent the data from each process, **the models share several common business dimensions**, namely, date, product, and store.
- Using **shared, common dimensions is absolutely critical to designing data marts that can be integrated**. They allow us to combine performance measurements from different processes in a single report.



[Kimball, 2002]

Integration

- Use multipass SQLto query each data mart separately, and then we outer join the query results based on a common dimension attribute. This linkage, often referred to as **drill across**, is straightforward if the dimension table attributes are identical.

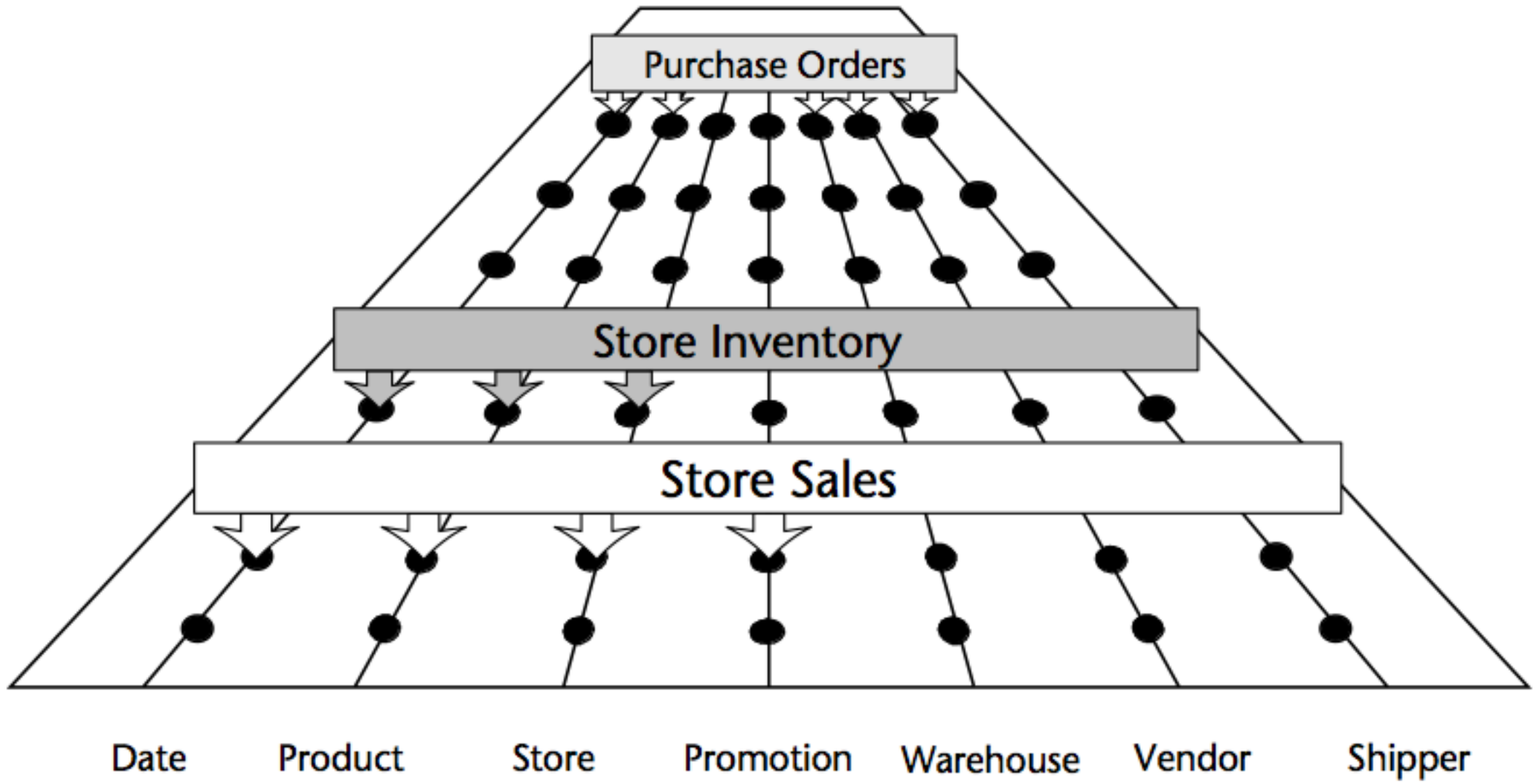


[Kimball, 2002]

Data Warehouse Bus Architecture

Data Warehouse Bus Architecture

- For long-term data warehouse success, we need to use an architected, incremental approach to build the enterprise's warehouse. The approach is the **data warehouse bus architecture**.



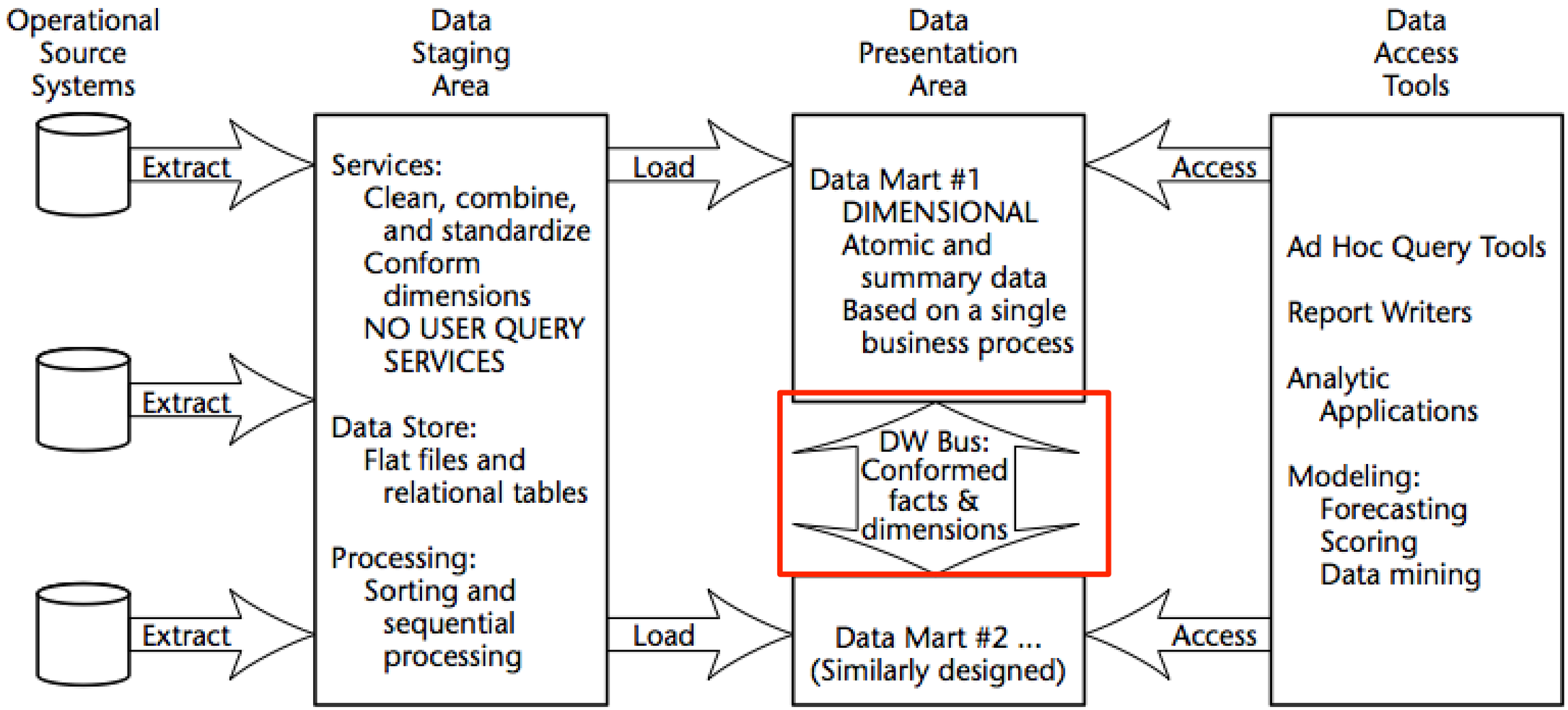
[Kimball, 2002]

Data Warehouse Bus Architecture

- The data warehouse bus architecture provides a rational approach to decomposing the enterprise data warehouse planning task. During the limited duration architecture phase, the team designs a **master suite of standardized dimensions and facts that have uniform interpretation across the enterprise**. This establishes the data architecture framework.
- The bus architecture is **independent of technology and the database platform**. All flavors of relational and online analytical processing (OLAP)-based data marts can be full participants in the data warehouse bus if they are designed around **conformed dimensions and facts**.
- Data warehouses will inevitably consist of numerous separate machines with different operating systems and database management systems (DBMSs). If designed coherently, they will share a uniform architecture of conformed dimensions and facts that will allow them to be fused into an integrated whole.

[Kimball, 2002]

Data Warehouse Bus Architecture



[Kimball, 2002]

Data Warehouse Bus Matrix

- It is important to remember that we are identifying the **business processes** closely identified with **sources of data**, not the organization's business departments. The **matrix rows** translate into **data marts** based on the organization's primary activities. We begin by listing the data marts that are derived from a single primary source system, commonly known as first-level data marts.

the columns depict the interaction between the data marts and common dimensions.

BUSINESS PROCESSES	COMMON DIMENSIONS: Date	Product	Store	Promotion	Warehouse	Vendor	Contract	Shipper
Retail Sales	X	X	X	X				
Retail Inventory	X	X	X					
Retail Deliveries	X	X	X					
Warehouse Inventory	X	X			X	X		
Warehouse Deliveries	X	X			X	X		
Purchase Orders	X	X			X	X	X	X

[Kimball, 2002]

Data Warehouse Bus Architecture

- Once it is time to begin a data mart development project, we recommend starting the actual implementation with **first-level data marts** because they minimize the risk of signing up for an implementation that is too ambitious.
- Once we've fully enumerated the list of first-level data marts, then we can identify more complex **multi-source marts as a second step**. We refer to these data marts as consolidated data marts because they typically cross business processes. While consolidated data marts are immensely beneficial to the organization, they are more difficult to implement because the ETL effort grows alarmingly with each additional major source that's integrated into a single dimensional model.
- Example:
  - **Profitability** is a classic example of a consolidated data mart where separate **revenue** and **cost** factors are combined from different process marts to provide a complete view of profitability.

[Kimball, 2002]

Conformed Dimensions

Conformed Dimensions

- **Conformed dimensions** are either **identical** or **strict mathematical subsets** of the most granular, detailed dimension.
- **Conformed dimensions** have **consistent dimension keys**, **consistent attribute column names**, **consistent attribute definitions**, and **consistent attribute values** (which translates into consistent report labels and groupings).

- Dimension tables are **not conformed** if the **attributes are labeled differently** or **contain different values**.
  - If a customer or product dimension is deployed in a non-conformed manner, then either the separate data marts cannot be used together or, worse, attempts to use them together will produce invalid results.

[Kimball, 2002]

Basic level of conformed dimensions

- At the most basic level, conformed dimensions **mean the exact same thing** with every possible fact table to which they are joined.

- The date dimension table connected to the sales facts is identical to the date dimension table connected to the inventory facts. In fact, the conformed dimension **may be the same physical table within the database**.

- However, given the typical complexity of our warehouse's technical environment with multiple database platforms, it is more likely that the **dimensions are duplicated synchronously in each data mart**.
- In either case, the date dimensions in both data marts will have the **same number of rows**, **same key values**, **same attribute labels**, **same attribute definitions**, and **same attribute values**. There is consistent data content, data interpretation, and user presentation.

[Kimball, 2002]

Basic level of conformed dimensions

- Most conformed dimensions are **defined naturally at the most granular level** possible:
  - The grain of the customer dimension naturally will be the individual customer.
  - The grain of the product dimension will be the lowest level at which products are tracked in the source systems.
  - The grain of the date dimension will be the individual day.

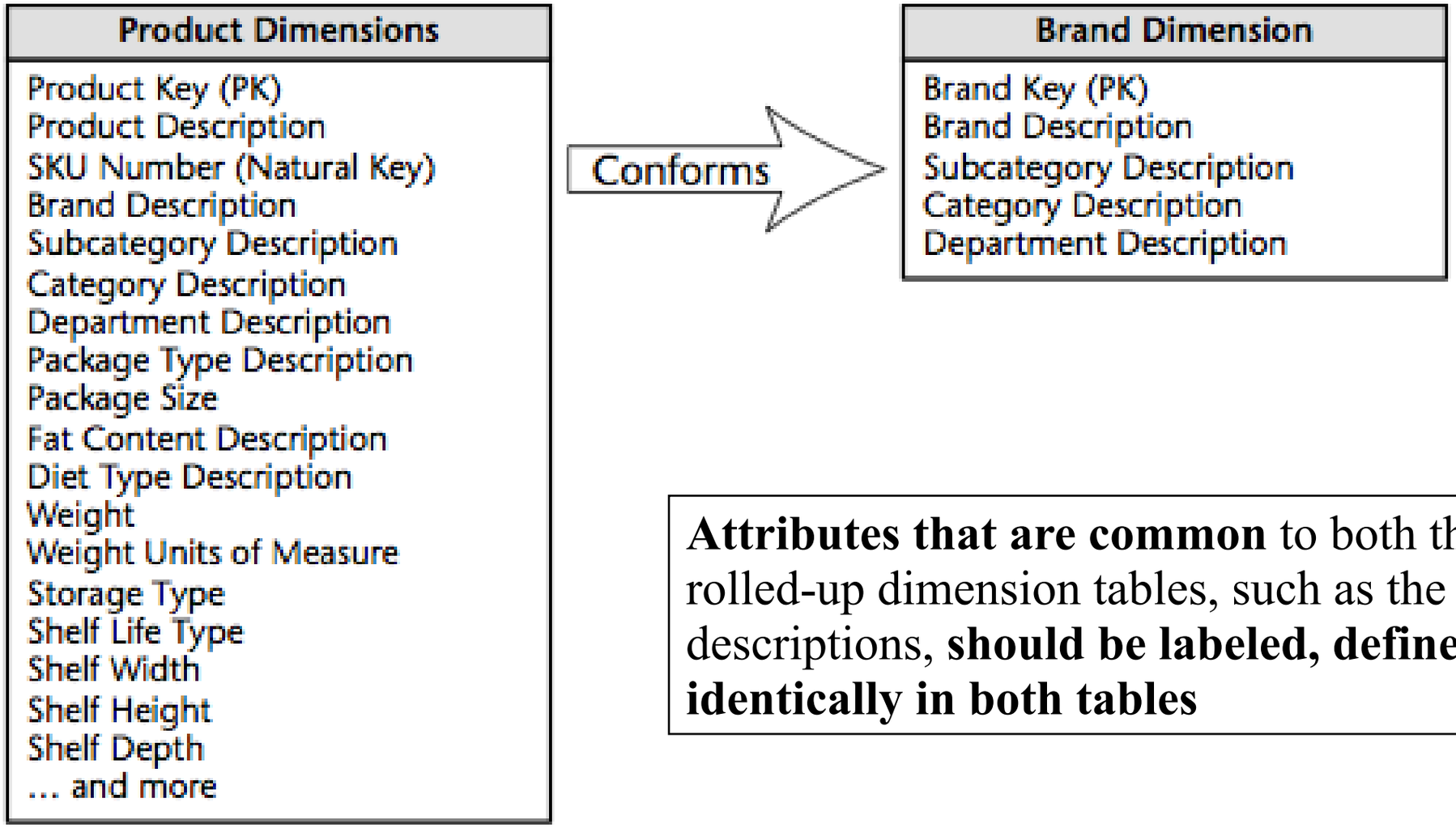
[Kimball, 2002]

Dimensions at a rolled-up level of granularity

- Sometimes **dimensions are needed at a rolled-up level of granularity**
  - A roll-up dimension is required because **the fact table represents aggregated facts** that are associated with aggregated dimensions.
    - If we had a weekly inventory snapshot in addition to our daily snapshot.
  - **The facts** simply may be **generated** by another business process at a **higher level of granularity**.
    - Sales captures data at the atomic product level, whereas forecasting generates data at the brand level. The product and brand dimensions still would conform if the brand table were a strict subset of the atomic product table. Attributes that are common to both the detailed and rolled-up dimension tables, such as the brand and category descriptions, should be labeled, defined, and valued identically in both tables

[Kimball, 2002]

Dimensions at a rolled-up level of granularity



Attributes that are common to both the detailed and rolled-up dimension tables, such as the brand and category descriptions, **should be labeled, defined, and valued identically in both tables**

Roll-up dimensions conform to the base-level atomic dimension if they are a strict subset of that atomic dimension.

[Kimball, 2002]

Conformed dimensions at the same level of granularity

- **Two dimensions are identical with the exception of a few set of attributes**

  - In the inventory snapshot schema we added supplemental attributes to the product and store dimensions that may not be useful to the sales transaction schema.
    - The product dimension tables used in these two data marts **still conform if the keys and common columns are identical**.

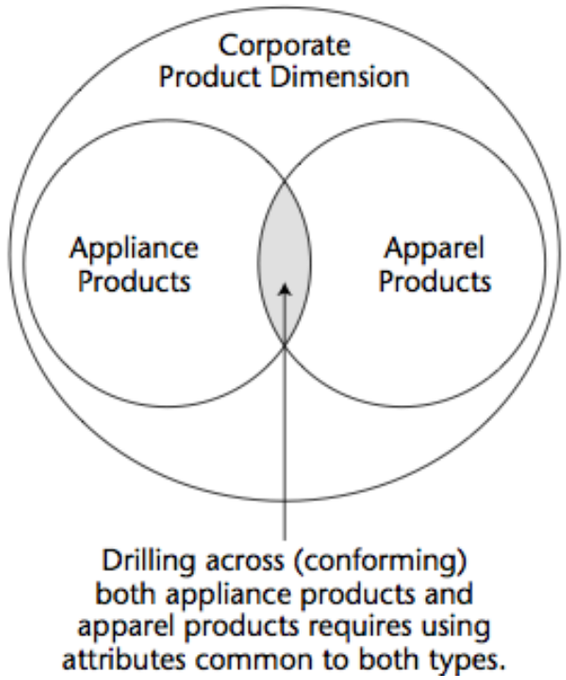
  - Given that the supplemental attributes were limited to the inventory data mart, we would be unable to look across processes using these add-on attributes.

[Kimball, 2002]

Conformed dimensions at the same level of granularity

- **Two dimensions are at the same level of detail but one represents only a subset of rows.**
  - A corporate product dimension that contains data for the full portfolio of products across multiple disparate lines of business
  - Analysts in the separate businesses may want to view only their subset of the corporate dimension, restricted to the product rows for their business. By using a subset of rows, they aren't encumbered with the entire product set for the organization.

the fact table joined to this sub-setted dimension must be limited to the same subset of products. If a user attempts to use a subset dimension while accessing a fact table consisting of the complete product set, he or she may encounter unexpected query results



[Kimball, 2002]

Both row and column dimension sub-setting.

- The conformed date dimension in our **daily** **sales** and **monthly** **forecasting** scenario is an example of both row and column dimension sub-setting.
  - We can't simply use the same date dimension table because of the **difference in roll-up granularity**.
  - However, the month dimension may consist of strictly the month-end daily date table rows with the exclusion of all columns that don't apply at the monthly granularity. Excluded columns would include daily date columns such as the date description, day number in epoch, weekday/weekend indicator, week-ending date, holiday indicator, day number within year, and others.
  - You might consider including a month-end indicator on the daily date dimension to facilitate creation of this monthly table.

[Kimball, 2002]

Managing conformed dimensions

- **Conformed dimensions will be replicated either logically or physically through out the enterprise; however, they should be built once in the staging area.**

- **The responsibility for each conformed dimension is vested in a group we call the dimension authority. The dimension authority has responsibility for defining, maintaining, and publishing a particular dimension or its subsets to all the data mart clients who need it.**

[Kimball, 2002]

Conformed Facts

Conformed Facts

- Revenue, profit, standard prices, standard costs, measures of quality, measures of customer satisfaction, and other key performance indicators (KPIs) are facts that must be conformed.
- In general, fact table data is not duplicated explicitly in multiple data marts. However, if facts do live in more than one location, such as in **first-level** and **consolidate data marts**, the underlying definitions and equations for these facts must be the same if they are to be called the same thing. If they are labeled identically, then they need to be defined in the same dimensional context and with the same units of measure from data mart to data mart.

[Kimball, 2002]

Conformed Facts

- Sometimes a fact has a natural unit of measure in one fact table and another natural unit of measure in another fact table. The correct solution is to carry the fact in both units of measure so that a report can easily glide down the value chain, picking off comparable facts.

[Kimball, 2002]

Further Reading and Summary

Notes

- It is important to analyze the growing rate of any fact table. It could be necessary retain historical data at different temporal granularity.
- The date dimension can be used to partitioning the fact tables.
- Any fact that measure static level (stocks, bank accounts, temperature, etc.) are non additive over the temporal dimensions. The average, min and max aggregators should be considered. Note that the SQL AVG is inappropriate to calculate the average over days.
- It is very common that the date dimension has many roles. Each role may be well implemented by using views with the appropriate renaming of some columns.

[Kimball, 2002]

Essential Concepts

- Data Warehouse Bus Matrix
- Conformed dimensions
- Dimension Management
- Drill-across

- **Further Readings**
  - The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling (Second Edition), Ralph Kimball, Margy Ross. 2002
    - From page 67 To 88

[Kimball, 2002]