Recommended Data Modeling Practices

In building your LDM, the goal is to express your business events and processes so that you can easily measure them. However, this approach opposes techniques in traditional computer system development.

Contents

- Designing Fact Tables
- Practical Modeling Tips
- Data Modeling Bibliography

Your model design should reflect the way people think about business and the way people measure business. These measures are contained in the model's facts, while the attributes of the model are the parameters providing the context. Without context, business measures are meaningless: What are the units? Over what period? For which products?

A data model sits in the middle of the triangle between:

- Your business requirements (what's needed)
- Your data (what you have)
- Your BI tools (what's possible)

Particularly in the business intelligence space, data modeling is an art and science unto itself. While a technical background is helpful, it is no guarantee of success.

The data model is also an intersection between the end point of technical processing (where ETL ends) and the access limits of the business audience (where reporting starts). The art is to make your model accessible to both groups, technical and business, and the science is to make it perform well under your operating conditions.

Since BI datamarts are aimed at providing answers to business questions, your data model should tend to prioritize the business people.

To begin a data modeling project, you typically start by creating an enterprise bus matrix. This matrix identifies the relevant business processes that you are attempting to model. Each process must have its own fact table, which is used to store the numerical information to track the business process.

- A fact table includes raw numbers from your data sources (for example, order amount, sales price, quantity), which you intend to aggregate to measure the process.

  If the business process can be broken down into simpler components, you should probably create separate fact tables.

  If two facts do not occur at the same time during the same business process, they should probably be recorded in separate fact tables.

- All facts from the same fact table are recorded at exactly the same level of detail (Sales BY Product BY Region). This level of detail is called grain.

  The unique identifier for a row in a fact table is its grain. Thus, you cannot have two rows in a fact table with identical grain.

- Some combinations of facts may not be written to the fact table. The degree to which all possible combinations are not complete is called the table’s sparsity.
  - For example, if you have a fact table recording Sales BY Day of Month BY Product, you may have no data on the days of the month that correspond to weekend days (assuming the stores are closed) or for products that have been discontinued.
  - To examine sparsity, you must look at the data itself to determine the data footprint of the fact table.

For each business process, you must identify its dimensions, which are the contextual information used to aggregate your facts.

- Examples of dimensions for Sales are Customer, Date, and Product.
- A dimension consists of one or more attributes, columns from your data sources, which are grouped together because of their logical relationship. For example, the Customer dimension can include attributes such as Customer Id, Customer Name, and Customer City.
- In a dimension, one attribute is used as a reference point into all fact tables connected to the dimension. This attribute is called a connection point. A connection point functions like a key value. It must have unique values across all rows.
Dimensions are usually shared across fact tables to create the enterprise bus matrix.

When dimensions are shared across fact tables, you can aggregate the facts of individual tables by shared attributes. So, if the Customer dimension is shared across the Sales and Expenses fact tables, you can perform aggregations Sales and Expenses by the Customer Name attribute. As a result, these facts can be compared side-by-side in reports. This capability provides significant leverage in reporting from your data model.

All users of the data model must share a common understanding of what the attributes of the dimension mean. Documentation of the data model is important.

Without specific optimizations, your logical data model consists of datasets for identified dimensions and fact tables. Each fact table is connected to its dimensions, including specialized date dimensions. The result is called a star schema.

Designing Fact Tables

Depending on the type of underlying business process, fact tables can be designed in one of the following major ways:

- **Transactional fact table for discrete events.** One row corresponds to a single transaction, usually connected to one date dimension. Rows are added, not updated.
- **Accumulating fact table for evolving events.** This method models a process passing through several predefined stages, usually connected to multiple date dimensions. One row corresponds to an execution of the process, with individual stages of the process tracked in separate columns. Rows must be updated throughout the stages.
- **Snapshotted fact table for recurring events.** Multiple rows are required to capture the status of the business process at different moments, each of which is timestamped. The fact table is connected to a Snapshot date dimension, which is used for filtering, not for aggregations. Rows are added, not updated.

Practical Modeling Tips

- Verify that you understand the level of detail (the grain) in each fact table. In general, the dimensions indicate the level of detail.
- Where possible, avoid including non-additive facts, which cannot be used in metric computations.
  - Instead of storing a ratio or percentage, store the facts that can be used to calculate the percentage or ratio. The calculation become a metric.
  - Instead of storing unit prices, store the extended price (units * unit price). Put the calculation of unit price or average price into a metric.
- Avoid placing attributes in fact tables. Fact tables should contain facts and foreign keys to attributes stored in other dimensions.
- Facts placed in the same fact table must be at the same level of detail (grain) and from the same business process. If two events occur at different times, they belong in different fact tables.
- For example, ordered units and shipped units most likely belong in separate fact tables because the order is created before the shipment is made.
- Avoid snowflake structures, which are normalized dimension tables. Dimension tables should not have other dimensions are parents or children. For a hierarchy such as Division > Region > Sales Office, there are two easy solutions:
  - Collapse Division and Region into Sales Office, yielding one dimension.
  - Leave Division and Region as separate dimensions, but do not connect them to each other. Division, Region and Sales Office all become independent dimensions connected to the fact tables. This alternative is ideal when there are multiple fact tables that require the Division and Region dimensions.
- Do not connect fact tables directly to other fact tables. When combining facts from multiple fact tables in the GoodData Portal, you should connect them through a common dimension.
- Suppose your source file contains data that should become attributes and facts on a set of payment transactions. In addition to the payment amount, a payment transaction might also have a check number, payee name and payment description, all of which become attributes in your model. Ideally, you should create a dimension for Payment Info containing the attributes and a fact table containing only the facts of the transaction.
- Create common dimensions that can be reused (shared) when you create additional fact tables. For example, you should have only one dimension table for customer, one for product, one for an employee, and so on. These conformed dimensions ensure uniformity of data in the project and enable re-use of the associated contextual information.

Name your model objects in a consistent manner that is understandable by business users, since they will interact with them in project that uses your model. See LDM Naming Conventions.

Data Modeling Bibliography