Basics of Dimensional Modeling

Data warehouse and OLAP tools are based on a dimensional data model. A dimensional model is based on dimensions, facts, cubes, and schemas such as star and snowflake.

Dimension

Developing an operational system requires interviewing users who perform day-to-day business operations of a company. In developing a data warehouse, users are generally unable to define their requirements. However, they can provide some important insights of the business, such as the parameters that determine the success in a company or department.

A dimension is a data structure that categorizes facts and measures in order to enable users to answer business questions through important business parameters. Commonly used dimensions are customers, products, locations and time.

Managers think of the business in terms of business dimensions. Thus dimensions are parameters that define the success of a business. For example, a marketing vice president is interested in the revenue numbers by month, in a certain division, by customer demographic, by sales office, relative to the previous product version.

So the business dimensions are month, division, demographics, sales office, and product version. The revenue is the fact that the vice president wants to know.

For a retail store, the important measurement or fact is the sales units. The business dimensions might be time, promotion, product, and store. For an insurance company, the important measurement or fact might be claims, and the business dimensions are agent, policy, insured party, status, and time.

These examples show that business dimensions are different and they are relevant to the industry and to the subject for analysis. Also the time dimension is common to all industry – almost all business analyses are performed over time.
**Dimensional Table**

When a business dimension is abstracted and represented in a database table, it is called a dimensional table. Thus, a dimension can be viewed as an entity (provide a definition of an entity as learned in the database class!). A dimensional table provides the textual descriptions of a business dimension through its attributes. Some characteristics of the dimension tables are:

- **Dimension tables tend to be relatively shallow in terms of the number of rows, but are wide with many columns.**
- **A dimensional table always has a single primary key.**
- **Dimensional tables are also typically highly denormalized.**

Dimensional table attributes play a vital role in query processing and in report labels. In many ways, the power of the data warehouse is directly proportional to the quality and depth of the dimension attributes.

**Fact**

A fact is a measurement captured from an event (transaction) in the marketplace. It is the raw materials for knowledge – observations. A customer buys a product at a certain location at a certain time. When the intersection of these four dimensions occur, a sale is made. The sale is describable as amount of dollars received, number of items sold, weight of goods shipped, etc. – a **quantity** that can be added to other sales similar in definition. Thus, a meaningful and measurable event of significance to the business occurs at the intersection point of business dimensions. It is the fact, which happened. We use the fact to represent a business measure.

A data warehouse fact is defined as an intersection of the dimensions constituting the basic entities of the business transaction. It is not easy to show the intersection of more than three dimensions in a diagram, but facts in a data warehouse may originate from many dimensions.
**Fact Table**

A fact table is the primary table in a dimensional model where the **numerical performance measurement** of the business are stored. There can be many performance measurements or facts in a fact table.

A row in a fact table corresponds to a measurement. The most useful facts in a fact table are numeric and additive. Characteristics of fact Table are:

- Fact tables tend to be deep in terms of the number of rows but narrow in terms of the number of columns.
- Thus, fact tables usually make up to 90 percent or more space of a dimensional database.

All fact tables have two or more foreign keys that connect to the dimension tables' primary keys. When all the keys in the fact table match their respective primary keys correctly in the corresponding dimension tables, we say that the tables satisfy **referential integrity**. We access the fact table via the dimension tables joined to it.

The fact table itself generally has its **primary key** made up of a subset of the foreign keys. This key is called a **composite or concatenated key**. Every fact table in a dimensional model has a composite key, and conversely, every table that has a composite key is a fact table.

Another way to say this is that in a dimensional model, every table that expresses a many-to-many relationship must be a fact table. All other tables are dimension tables.

This is a good time to view the sample **FoodMart** data warehouse dimensions and facts, including their attributes and values.
The Dimensional Model: Star Schema

The model that brings the dimensions and facts together is termed as the dimensional model. In this model, the fact table consisting of numeric measurements is joined to a set of dimension tables filled with descriptive attributes.

In the model, the fact table is at the center and the dimension tables are hung around like a star. Hence, this characteristic structure is often termed as *star schema*.

When a customer_id, a product_id, and a time_id are used to determine which rows are selected from the fact table, this way of collecting data is called the *star schema join*.
The dimensional model is simple and symmetric. The data is easier to understand and navigate. Every dimension is equivalent; all dimensions have symmetrically equal entry points into the fact table. The logical model has no built-in bias regarding expected query patterns.

The simplicity also has performance benefits. Fewer joins are necessary for query processing. A database engine can make strong assumptions about first constraining the heavily indexed dimension tables, and then attacking the fact table all at once with the Cartesian product of the dimension table keys satisfying the user's constraints.

With dimensional models, we can add completely new dimensions to the schema as long as a single value of that dimension is defined for each existing fact row. Likewise, we can add new, unanticipated facts to the fact table, assuming that the level of detail is consistent with the existing fact table. We can also supplement preexisting dimension tables with rows down to a lower level granularity from a certain point in time forward.

In all of the cases above, existing data access applications will continue to run without yielding different results. Data would not have to be reloaded.

Another way of thinking about the simplistic nature of star schema is to see how the dimensions and facts contribute to the report. The dimension table attributes supply the report labeling, whereas the fact tables supply the report's numeric values.
The Data Cube

Another approach to look at the multi-dimensional data model is through a data cube. It allows data to be modeled and viewed in multiple dimensions. It is developed on the basis of dimensions and facts as well.

The cube is a metaphor or a concept that is suitable for OLAP processing (slicing and dicing along a business dimension), as compared to the star schema which is suitable for query processing. The data cube can be defined as the intersection of dimensions that provide some facts of interest to the business.

Thus data cubes can be translated into star schema and vice versa. However, high level aggregation of data is efficiently stored as cubes; having been pre-calculated; alternative roll-ups across changing dimensions are more efficiently and flexibly performed by star schema, based on available details.

The classic cube is the sale of a product by location by time, and it is a three-dimensional (3-D) cube.
Although we usually think of cubes as 3-D geometric structures, in data warehouse the data cube can be n-dimensional. To gain a better understanding of data cubes, let us start with an example of a 2-D data cube that is, in fact, a table or spreadsheet for sales data per quarter (time dimension) for various items (product dimension) for a particular location (location dimension). The fact or measure is the dollar amount sold.

In order to view the sales data in a third dimension (the location), we include additional 2-D sales data for other locations. Conceptually, we may view these data in the form of a 3-D data cube as shown below.
Suppose, we would like to view our sales data in a fourth dimension, such as supplier. Viewing this in 4-D becomes tricky; however, we can think of a 4-D cube as being a series of 3-D cubes, as shown below. If we continue this way, we may display any n-D data in a series of (n-1)D cubes.

The data cube is a metaphor or concept for multidimensional data storage. The actual physical storage of such data may differ from its logical representation.

In the data warehouse literature, 1-D, 2-D, 3-D cube and so on are in general referred to as a cuboid. Given a set of dimensions, we can construct a set of cuboids, each showing the data at a different level of summarization. The cuboid that holds the lowest level of summarization is called the base cuboid. For example, the 4-D cuboid below is the base cuboid for the given time, item, location, and supplier dimensions. The apex cuboid is typically denoted by all.
Hierarchies in Dimensions

In a data warehouse or data mart, measures are stored in the fact table in such details that users can roll-up in various levels of summarization. This is called aggregation.

For example, if sales data in a grocery store are kept in the level of a single customer buying a particular item in a particular day in a particular store, then we can summarize or aggregate the data for various days, weeks, months, quarters, and years; and all of these for a store, zone, state, and country; as well as by products, product group, department, and so on.

Only the sales data in the lowest level are kept in the fact table, but the descriptions of various levels of data are kept in the dimension tables, so that appropriate tools can be used to summarize data in various levels.

A hierarchy defines a sequence of mappings from a set of low-level concepts to higher-level, more general level concepts. Consider a hierarchy for the dimension Location. If City is in the lowest level of hierarchy, then all cities can be mapped to a higher level of State, and all states can be mapped to a higher level of Country, and so on.

The dimensional levels form a tree-like structure, and the members in the lowest level of the hierarchy are called leaf members. There is only one member in the topmost level. A dimension can not exist without leaf members, but it is possible to have a dimension with nothing but leaf members – that is, with only one level, like city in the diagram below.
**Balanced and Unbalanced Hierarchy**

The example above of city -> state -> country -> continent forms a complete or balanced hierarchy. Some hierarchies do not form a complete order, such as day -> week -> month -> quarter -> year.

Here day is in the lowest level of hierarchy, and there are two sets of hierarchies: day -> month -> quarter -> year and day -> week -> year. This type of hierarchy is called a partial or unbalanced hierarchy.

The multi-dimensional model requires the dimensional tree to be balanced; that is, there are equal number of members in each level. However, it is possible to have an unbalanced tree. For example, some of the states in the tree below do not aggregate to a higher level. However, for all practical purposes, the aggregation of facts must be maintained in each level – only that there may not be any dimensional attribute representing a particular level, or it is empty.
Implementing Dimensional Hierarchies

Dimensional hierarchies are stored as attributes in the dimension tables, and all related hierarchies are typically stored in a single dimension table. A description of each level of hierarchy is kept in the multidimensional metadata.

For example, *date, day, month, and year* are stored in a *Date* dimension; while *product, brand, category, and department* are stored in the *Product* dimension. The example below illustrates a *Retail Store* database schema and the associated *Date* and *Product* dimension tables.
Use of Dimensional Hierarchy

Hierarchies in dimensions are used for selecting and aggregating data at the desired level of detail. The fact table contains data only in the lowest level of the hierarchy. The higher-level data are obtained through aggregation of lowest-level fact data for the same instances of a dimensional level-attribute.

For the above example, if we want to find the total \textit{Sales Quantity} and the \textit{Sales Dollar Amount} for each of the two departments, \textit{Bakery} and \textit{Frozen Food}, we first select \textit{Bakery} and \textit{Frozen Food} from the \textit{Product Dimension} table and then add up all the values of \textit{Sales Quantity} and \textit{Sales Dollar Amount} from the Fact Table (not shown) corresponding to the two products.

This requires adding up separately, fact values for \textit{Product key} = 1, 2, 3, and 4, and \textit{Product key} = 5, 6, 7, 8, and 9, for all possible values of other keys in the Fact table. The result is shown below.

<table>
<thead>
<tr>
<th>Department Description</th>
<th>Sales Quantity</th>
<th>Sales Dollar Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bakery</td>
<td>5,088</td>
<td>$12,331</td>
</tr>
<tr>
<td>Frozen Food</td>
<td>15,565</td>
<td>$31,776</td>
</tr>
</tbody>
</table>

Instead of aggregation by \textit{Product Description}, if we want to go into details for the \textit{Brand Description} of the product, we project on the \textit{Product Description} and \textit{Brand Description} from the \textit{Product Dimension}, and then select all \textit{Sales Quantity} and \textit{Sales Dollar Amount} from the Fact Table, and add them up.
OLAP Operations: Querying Multidimensional Data

In the multidimensional model, data are organized into multiple dimensions, and each dimension contains multiple levels of abstraction defined by the hierarchies.

This organization provides users with the ability to view data from different perspectives. A number of data cube operations exist to materialize the different views, allowing interactive querying and analysis of the data. Following are some typical OLAP operations for multidimensional data.

Let us take an example of a cube containing the dimensions of location, time, and item, where location is aggregated with respect to city values, time is aggregated with respect to quarters, and item is aggregated with respect to types.

**Roll-Up:** The roll-up (or drill-up) operation performs aggregation on a data cube, either by climbing up a data hierarchy for a dimension or by dimension reduction. Roll-up by dimension reduction means that aggregation is performed up to the top level of a dimension. For example, if the location hierarchy contains three levels, city -> state -> country, then reduction of location dimension means, the resulting fact data will be summed over the city, and then over the states.

**Drill-Down:** Drill-down is the reverse of roll-up. It navigates from less detailed data to more detailed data. It can be done either stepping down a hierarchy for a dimension or introducing additional dimensions. Adding a new dimension means the fact table must contain (or be added) data in that dimension.

**Slice and Dice:** The slice operation performs a selection on one dimension of the given cube, resulting in a sub-cube. For example, we can select all sales data for various cities and items for a particular quarter = Q1.

The dice operation defines a sub-cube by performing a selection on two or more dimensions. For example, we can first slice on time to include sales for some quarters, and then on location to include sales of some cities.

**Pivot (Rotate):** Pivot is a visualization operation that rotates the data axes (in view) in order to provide an alternative presentation of the data.
Understanding Dimensional Model: Preview FoodMart Data Warehouse after downloading from the course website.