Facts and Dimensions in a Data Warehouse?

Facts and dimensions are the fundamental elements that define a data warehouse. They record relevant events of a subject or functional area (facts) and the characteristics that define them (dimensions).

Data warehouses are data storage and retrieval systems (i.e., databases) specifically designed to support business intelligence (BI) and OLAP (online analytical processing) activities. They are different from databases designed to support transactional systems – e.g., e-commerce sites – whose function is primarily OLTP (online transactional processing). You can read about WHAT A DATA WAREHOUSE IS to get a broad idea.

It is common for data warehouses to contain large volumes of historical information for performing queries and analyses. It is also common for the information in a data warehouse to come from a diverse range of sources and to be used in queries that cross-reference information from different sources for discoveries and gaining insights.

The main benefit of a data warehouse is for extracting significant value from information accumulated over time. Organizations depend heavily on their data warehouses for fundamental support in decision-making. The design of a data warehouse should not be taken lightly due to its value to the organization, and once it starts to be populated with data, modifying its structure is highly risky and potentially costly.

The Four Pillars of the Data Warehouse

William Inmon, one of the precursors of the data warehouse concept, points out that data warehouses are characterized by four fundamental conditions:

- Oriented to a single subject or a particular functional area. For example, it is oriented to company sales.
- They unify and create consistency among data from disparate sources.
- Persistent and immutable. Once data enters a data warehouse, it stays there and does not change.
- Structured in time intervals. To provide information from a historical perspective, data warehouses record information over different intervals, such as weekly, monthly, quarterly, etc.

A well-designed data warehouse is high-performance and responsive to queries. It also provides flexibility so that business analysts (or any other end user of the data warehouse) can query from different points of view. Users can alternate between a high-level overview and deep queries at the greatest level of detail as they wish.

The data warehouse serves as the source of information for BI visualization tools. It provides end-users with the ability to easily generate reports, dashboards, graphs, and other forms of data inquiry.

An X-Ray of a Data Warehouse

From a technical point of view, a data warehouse is a database. Therefore, it is composed of tables, fields, relations, keys – just like any other database. And as such, you can model it using a database design tool such as VERTABELO.

However, the tables of a data warehouse have some peculiarities that differentiate them from those commonly used for transactional processing. In particular, data warehouse tables are divided into two main categories: fact tables and dimension tables.

Facts and dimensions in a data warehouse should form a layout that responds to a particular topology. There are two main topologies:

the STAR SCHEMA and the SNOWFLAKE SCHEMA. In a star schema, individual dimensions surround a single fact table, while a snowflake schema has a hierarchy of dimensions.

In Data Warehouse Modeling, a **star schema** and a **snowflake schema** consists of **Fact** and **Dimension** tables.
Fact Table:

- It contains all the primary keys of the dimension and associated facts or measures(is a property on which calculations can be made) like quantity sold, amount sold and average sales.

Dimension Tables:

- Dimension tables provides descriptive information for all the measurements recorded in fact table.
- Dimensions are relatively very small as comparison of fact table.
- Commonly used dimensions are people, products, place and time.

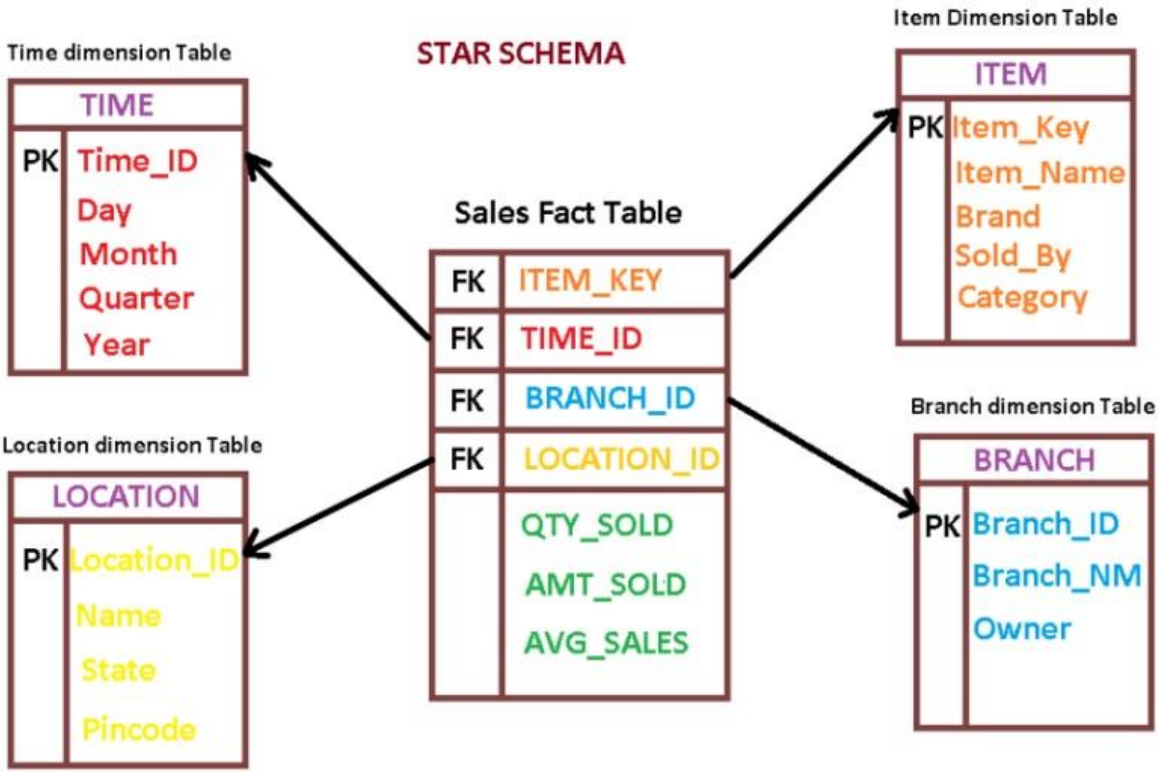
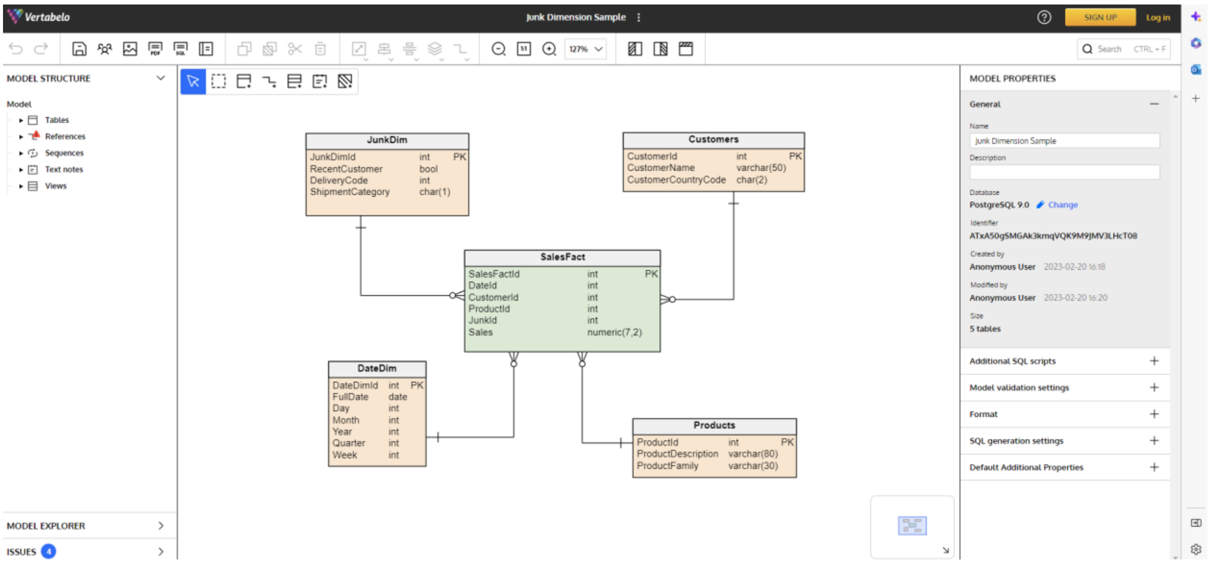


image source



JunkDim		
JunkDimId	int	PK
RecentCustomer	bool	
DeliveryCode	int	
ShipmentCategory	char(1)	

Customers		
CustomerId	int	PK
CustomerName	varchar(50)	
CustomerCountryCode	char(2)	

SalesFact		
SalesFactId	int	PK
DateId	int	
CustomerId	int	
ProductId	int	
JunkId	int	
Sales	numeric(7,2)	

DateDim		
DateDimId	int	PK
FullDate	date	
Day	int	
Month	int	
Year	int	
Quarter	int	
Week	int	

Products		
ProductId	int	PK
ProductDescription	varchar(80)	
ProductFamily	varchar(30)	

General

Name: Junk Dimension Sample

Description:

Database: PostgreSQL 9.0 Change

Identifier: ATxA50gSMGAk3kmqVQK9M9JMV3LHcT08

Created by: Anonymous User 2023-02-20 16:18

Modified by: Anonymous User 2023-02-20 16:20

Size: 5 tables

EDIT MODEL IN YOUR BROWSER

A typical star-shaped data warehouse schema: the fact table sits in the middle, surrounded by the dimension tables.

Dimensions Versus Facts

It is fundamental to understand the differences between facts and dimensions in a data warehouse for creating a flawless database design. Facts are the measurable events related to the functional area covered by a data warehouse. For example, sales are facts in a data warehouse for the sales and distribution area of a company. If the data warehouse contains information about patient care in a hospital, patient visits to the hospital are facts.

Fact tables are the core tables of a data warehouse. They contain quantitative information, commonly associated with points in time. They are used in trends, comparisons, aggregations, and groupings. They feed analysis and visualization tools to allow insights to be discovered about the functional area.

Dimensions, on the other hand, are collections of reference information about the facts in a data warehouse. Dimensions categorize and describe the facts recorded in a data warehouse to provide meaningful, categorized, and descriptive answers to business questions.

In the design of a data warehouse, it is common to create dimension tables first and then create fact tables by relating them to the dimension tables through foreign keys. For this reason, let’s first see what dimension tables are and how they are classified.

Defining Dimension Tables

Just like most any table in a relational model, dimension tables have a primary key, which may be natural or surrogate, and a set of attributes. However, dimension tables do not need to be in the 3rd normal form to do their job. In fact, to improve query performance, dependencies are allowed to exist among non-key attributes of a dimension table to minimize the number of tables that need to be joined in a query.

Dimension tables have low cardinality and a relatively small number of rows so that their impact on performance is minimal. They also have numerous attributes that are mainly textual or temporal. The dimensions of a data warehouse, together with their attributes, serve as the guidelines for data analysts to view the information from different angles and apply different filtering criteria.

Types of Dimensions

The dimensions of a data warehouse are classified into different types according to their behavior and use. When designing tables in a data warehouse, knowing the type of each dimension helps you make the right design decisions.

There are many dimension types. Below, we will see just four of them: conformed dimensions, role-playing dimensions, slowly changing dimensions, and junk dimensions. For more information on dimension types, take a look at THE MOST COMMON TYPES OF DIMENSIONS IN DATA WAREHOUSING.

Conformed Dimensions

A conformed dimension can be associated with different fact tables, maintaining the same meaning with all of them. In constellation-type data warehouse designs with multiple fact tables, conformed dimensions make cross-domain queries possible.

A typical conformed dimension is the date. Its meaning does not vary by fact table. For this reason, most data warehouses have a single date dimension shared by all fact tables.

Other conformed dimensions are not as obvious as the date and may pose design challenges as they need to provide consistency across different domains. Data sources in a conformed dimension may have structural differences between each other, such as missing/additional columns, columns with different data types, and differently named columns representing the same data.

An example is an SKU dimension shared by a purchase fact table and a sales fact table. Using SKU as a conformed dimension requires gathering the dimension attributes coming from both domains in a single table and making the data types and attribute names common to both.

Role-Playing Dimensions

Role-playing dimensions are used by different fact tables, just like conformed dimensions. But unlike conformed dimensions, they have different meanings depending on the fact table or even the field within the fact table.

The date dimension, mentioned earlier as a conformed dimension, can be a role-playing dimension if we relate the same table to different date attributes. For example, in the sales fact table, we may have an order date, a delivery date, and a payment date, all related to the same date dimension.

Slowly-Changing Dimensions (SCDs)

One difference between events and dimensions is that events occur once and leave a record that they have occurred. That record stays there forever with no chance of it changing.

Dimensions, on the other hand, may change. SKU attributes, some customer data, and even the names of seemingly immutable things like locations or institutions may change. Dimensions that are susceptible to change are called slowly changing dimensions (SCDs). Here, “slowly” is subjective; it would be more accurate to call them simply changing dimensions.

When designing a data warehouse, it is necessary to think about how those changes impact the dimension tables since changes are inevitable in this type of dimension. Fortunately, a series of strategies to keep SCDs up to date have been developed by those who have already faced this dilemma.

When defining an SCD, we must decide which update strategy to apply. The most common update strategies are:

- No update. This strategy is applied to immutable dimensions such as most date-type dimensions.
- Attributes are always updated to the most recent value. All events referring to the dimension are associated with the most recent values of its attributes, no matter when they occur.

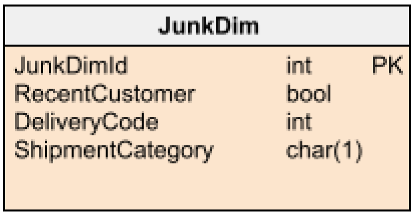
- Changes to a dimension are applied by generating new versions of the changed elements, either by adding records that include an effective date, by using history tables, or by adding fields to differentiate the new data from the original data.

Junk Dimensions

In a data warehouse design, facts often have indicator attributes like flags, Boolean values, or some other set of values that do not make sense as a dimension because of their low cardinalities. To avoid creating small dimensions for each of these attributes and increasing the number and sizes of the fact tables unnecessarily, a junk dimension is often created to gather all these attributes into a single table.

It gathers all possible combinations of the attributes it encompasses, identifying each combination with a single surrogate primary key. In the fact table, it is sufficient to include a single foreign key to the junk dimension table instead of storing the value of each of the attributes. To clarify this idea, let’s look at an example.

Suppose you have a sales fact table with several indicator attributes, such as `FrequentCustomer` with values Yes/No, `DeliveryCode` with values 1 or 2, and `ShipmentCategory` with values A, B, or C. Instead of including these three attributes within the fact table, you can create a `JunkDim` table with this structure:



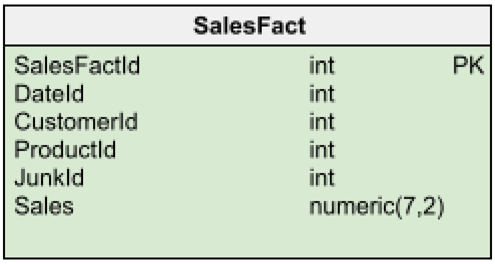
JunkDim		
JunkDimId	int	PK
RecentCustomer	bool	
DeliveryCode	int	
ShipmentCategory	char(1)	

Imagine this table has the following rows:

JunkDimId	RecentCustomer	DeliveryCode	ShipmentCategory
1	No	1	A
2	No	1	B
3	No	1	C
4	No	2	A
5	No	2	B
6	No	2	C
7	Yes	1	A
8	Yes	1	B
9	Yes	1	C
10	Yes	2	A
11	Yes	2	B

JunkDimId	RecentCustomer	DeliveryCode	ShipmentCategory
12	Yes	2	C

Then, our fact table may have a structure like this:



SalesFact		
SalesFactId	int	PK
DateId	int	
CustomerId	int	
ProductId	int	
JunkId	int	
Sales	numeric(7,2)	

Depending on the combination of junk dimensions for each fact, you assign the JunkId value that corresponds to the rows with the same combination of values.

Defining Fact Tables

In fact tables, there are two types of attributes: qualitative and quantitative. Qualitative attributes define the characteristics of a fact; they are commonly defined as a foreign key to a dimension table. A quantitative attribute defines a measure of a fact: an amount, a quantity, a length of time, or any other measurable (numerical) value on which you can apply statistical calculations like the sum, the average, and the variance.

Measures may be additive or non-additive depending on whether or not they can be summed across any dimension. They may also be semi-additive, meaning they can be summed across some of the dimensions of the data warehouse.

We can classify fact tables into three categories based on how measures are recorded:

- Transactional fact tables. In transactional fact tables, each row corresponds to an item in a transaction. For example, if the events recorded in the table are sales invoices, each row of the table contains the information of one item of an invoice. Transactional event tables aim to record as much detail as possible, so they usually include a large number of dimensions.
- Periodic snapshots: Periodic snapshot fact tables take snapshots of events for given time periods. They commonly use transactional type fact tables as the source of information. For example, they record total sales per day by summing data from a transactional sales fact table.
- Cumulative snapshots: Cumulative snapshot fact tables are used to mark the various milestones of a business process, usually with multiple date columns. Each row in a fact table summarizes the events that have occurred between the beginning and end of a process. This type of table is useful for data warehouses that gather information from pipeline processes such as claims processing and order fulfillment. Each row of a cumulative snapshot fact table corresponds to an event in the corresponding business process. As an example, take claim tickets – as the ticket changes status, the dates and times of the status changes are recorded in the corresponding record of the fact table, accompanied by measures of elapsed time and delays.

To learn more about fact tables in a data warehouse, read HOW TO ORGANIZE FACT TABLES IN DATA WAREHOUSE SYSTEMS.

Granularity of Facts

The data granularity of a fact table defines the greatest level of detail possible when analyzing the information in the data warehouse. More granular data allows for a greater level of detail, but it also implies a greater number of dimensions, a larger data warehouse, and greater complexity in queries and data-gathering processes.

The grain of fact tables is one of the most critical decisions in designing a data warehouse, as it determines the dimensions and how to record events in the fact tables. Once the data warehouse is designed and running, changing the grain of fact tables is practically impossible because of the effort, time, and cost implications.

Just as it is challenging to change how a large physical warehouse of materials and products is organized, it is challenging to change the structure of a large data warehouse. To avoid making insurmountable mistakes in a data warehouse, it is critical to define its two main elements, the facts and the dimensions, precisely and from the outset.