Chapter 6 More on Dimensions

- Assigning attributes to dimensions
- Splitting a dimension
- Role playing
- Handling NULLs
- Behavioral dimensions

Assigning attributes to dimensions

Dimensions

On one hand, organizing dimensions is simple because they have significant and obvious business value.
E.g. Student, Instructor, Section, …

On the other hand, difficulties may arise due to the contrast of normalized **OLTP databases where relationships are explicitly shown** vs a dimensional design where relationships are not so directly exposed

Assigning attributes to dimensions

Assigning attributes to dimensions:

The type of relationship between two attributes is a determining factor:

If a relationship depends on *context* this leads to the use of fact tables to capture that relationship
-relationship between student number and course number

If a relationship is one of *natural affinity* then the attributes are placed in the same dimension
-department code, course number

Assigning attributes to dimensions

Explicit Relationships
Fact tables hold FK references to dimensions.
These FKs provide the dimensional context for the facts, and can be considered as providing information about relationships between the dimensions.

Recall from ACS-3902 : Fact tables are examples of what ER modelers call intersection tables … useful for many-to-many-to-…-to-many relationships.

Assigning attributes to dimensions

Implicit Relationships
Relationships between dimension attributes can be implied through their coexistence in a table.

These relationships tend to exist only in a single context, representing a natural affinity rather than one based on processes e.g. Products are organized into categories

The relationships among attributes in a dimension table may change over time but are less volatile than the explicit relationships (via fact table)

  implicit relationships change
  … history can be preserved via type 2 technique.

Assigning attributes to dimensions

If we snowflake a design then we are making an implicit relationship explicit.

Assigning attributes to dimensions

Implicit Relationships

In a dimensional model, dimensions are grouped into tables based on natural affinity.

e.g. Departments and courses, for example, are related to one another regardless of whether a student registers.

A transaction is not required to establish this relationship.

Splitting a dimension

Generally a large set of dimension attributes facilitates analysis.

Dimensions can be so wide that database administrators become concerned about its effect on the database. A technical but valid concern.
e.g. wide rows may impact the way that the DBA allocates space or designates block size.
Large dimensions can also become a concern for ETL

Splitting a dimension

1st technique:

• simple separation of attributes generating two tables – a split dimension

• excessive row length is split across the two tables

• dimensions have a 1-1 relationship

• same surrogate key values used

• one FK from fact table to … – issues with RI

  • e.g figure 6-3

•unless one has 2 FKs with same value referencing the two different tables (representing the split dimension)

Splitting a dimension

2nd technique:
If there are distinct dimensions, then the dimension is redesigned as two tables, each with its own surrogate key.

These two dimensions will then participate in explicit relationships via a fact table

Splitting a dimension

3rd technique:
Outrigger for Free-Form Text Fields

As free-form text fields can lead to excessive row length, they can be relocated to a separate table and replaced with a foreign key reference.

Splitting a dimension

4th technique:
Recognizing subtypes

Create a supertype (base) dimension with the shared attributes, and separate subtype dimensions
Supertype and subtypes share the same surrogate key values

The base dimension is used when analyzing all subtypes, such as products, and the custom dimensions are used when studying only one particular subtype, such as books, subscriptions, or compact discs.

See figure 13-5,-6

Splitting a dimension

5th technique: (suppose the dimension is growing too fast)
Mini-dimensions

Isolate a subset of the dimension’s attributes and use them as the basis for a new dimension called a mini-dimension.

Similar to a junk dimension as mini-dimension attributes do not represent a single concept.

New dimension table can alleviate size problems at the expense of limited browsability.

Can have an unexpected impact on table growth.
See figures 6-4,-5,-6

Splitting a dimension

A mini-dimension is created by removing the more volatile attributes from the dimension in question and placing them in a new table with its own surrogate key.

These attributes share no direct relationship to one another, and there is no natural key.

A one-time-only process can populate this table with data by creating a row for each combination of values.

Splitting a dimension

The dimension table and the mini-dimension are only related via facts.

Limited browsability between a dimension and mini-dimension can be alleviated by adding a foreign key to the dimension table that refers to the mini-dimension.

Splitting a dimension

If needed, the full history of the relationships between a dimension and a mini-dimension can be preserved by designing an additional fact table.

Each row will identify a row in the dimension, a row in the mini-dimension, and the time frame during which they were associated … probably a *factless fact table*

Role playing

Measurement of a business process can involve more than one instance of a dimension.
When an auto dealer sells a car, for example, two employees are associated with the transaction: the salesperson who sold the car and the manager who approved the sale.

In a fact table, they are represented by multiple foreign key references to the same dimension table.

In Northwind an order has 3 dates: order date, requested date, shipped date

See figure 6-7,-8
Assignment 2 includes: modify the fact table to include FKs for requested date and shipped date

Nulls

NULL values cause complications

For reasons that are purely pragmatic, it is best to avoid the use of NULLs.

Attributes:
For dimension attributes, the inelegant but practical solution is to store a specific value such as 0 or “N/A” when data is not available.

Nulls

Relationships

If a relationship to a dimension is optional, or
the information is not available or incorrect,

then
NULL values can be avoided by establishing **special rows in the dimension** to represent not available, not applicable, incorrect, …

As mentioned Assignment 2 includes shipped date but not all orders have been shipped … need a ‘special’ row for ‘date not available’

Nulls

Future Events

When a fact table row represents something that may expire, it is useful to record a pair of dates: the date it became effective and the date it expired.
But if it hasn't expired yet …
See figure 6-12
note Dec 31, 9999

Consider Northwind and the shipped date …

Consider a schema that tracks a student's progress through university …

Behavioral Dimensions

A *behavioral* question is one that groups or filters facts based on the past behavior of members of a dimension.

Behavioral dimensions transform facts into dimensions, enabling analysis without complex queries or intensive processing

e.g. Are customers who generate over $1 million in orders receiving better discounts than those who generate $500,000 or less?”

We need to determine who those customers are first, then …

Behavioral Dimensions

Answering a question that groups or filters facts based on past behaviour goes beyond the capabilities of the basic SELECT/GROUP BY query.

To answer these questions, two major steps are required:
1. Identify past behavior for each member of the dimension in question.
2. Use this information as part of a query studying current behavior.

Suggested Technique: add columns to a dimension to track past behaviour.

Behavioral Dimensions

e.g.
analysts in a marketing group may want to be able to use the date of a customer's most recent order (*last_order_date*) to filter queries. Storing this date in the customer dimension eliminates the necessity to query an orders fact table to find the dates on which each customer last placed an order.

(Figure 6-13)

Behavioral Dimensions

Using an *Historic Fact*:
A behavioral attribute can also capture a fact of historic significance for storage in the dimension table.

Figure 6-13 contains an example of a historic fact stored in a dimension: *annual_sales*

Universities could keep
… a student's current gpa in the student dimension
… date of last registration
… date of first registration

Behavioral Dimensions

Categorizing Facts

e.g.
we might wish to group facts into three categories:
low annual sales, medium annual sales, high annual sales.

Each customer can be placed in one of those buckets, based on historic sales.

Behavioral Dimensions

Categorizing Facts

Figure 6-13 … behavioral attribute *annual_sales_group* attribute identifies three ranges, or "buckets," into which past history is sorted.
ETL needs to determine the bucket for each customer

Another approach (not in chapter 6)

Use a **Banding** table for reporting purposes

e.g. …search for "banding" in

http://books.google.ca/books?id=XoS2oy1IcB4C&pg=PT235&lpg=PT235&dq=kimball+banding+table&source=bl&ots=1AGhpCiKbB&sig=2eRzA19gEgsF9UomrPyZREnEBts&hl=en&sa=X&ei=wYQxT-mTLaX10gH6_4DxBw&ved=0CD8Q6AEwAw#v=onepage&q&f=false