Dimensional Modeling
CS386, Introduction to Database Systems
Jay Urbain, PhD

Dimensional Modeling for Data Mining

DOGBERT CONSULTS

YOU NEED TO DO DATA MINING TO UNCOVER HIDDEN SALES TRENDS.

IF YOU MINE THE DATA HARD ENOUGH, YOU CAN ALSO FIND MESSAGES FROM GOD.

...SALES TO LEFT-HANDED SQUIRRELS ARE UP...AND GOD SAYS YOUR TIE DOESN'T GO WITH THAT SHIRT.
Data Modeling

• A **data model** is an abstract model that describes how data is represented and accessed.

• Data models formally define **data objects** and **relationships** among data objects for a domain of interest.

• Typical applications of data models include supporting the development of databases and enabling the exchange of data for a particular area of interest.

• Data models are specified in a **data modeling language**.
Dimensional Data Model

• *Dimensional modeling* is a design technique that puts the data into a *standard framework* for *efficient aggregation* of related facts.

• A *dimensional model* consists of one *fact table* and multiple *dimension tables* (which describe attributes & are smaller).

• Two kinds of schemas are typically used when designing dimensional models, either a *star schema* or a *snowflake schema*.

• Commonly used in data warehousing / data mining systems.
Concepts of Dimensional Modeling

- *Dimensional modeling* (DM) is the name of a logical design technique often used for data warehouses.
- It is different from, and contrasts with, *entity-relation modeling* (ER).
- DM is the primary technique used for databases that are designed to support end-user queries in a data warehouse to support OLAP (On Line Analytical Processing).
- ER is very useful for the transaction capture and the data administration phases of constructing a data warehouse, but it should be avoided for end-user delivery.
What is ER?

• ER is a **logical design technique** that seeks to *remove the redundancy* in data.

• Initially capture a sales order as a single flat record with many fields.

• The line items of the order were probably represented as a *repeating group of fields* embedded in the master record (non-normalized).
Problems?

• Having this data on the computer is very useful, but we quickly learned some basic lessons about storing and manipulating data…
Lessons Learned with non-Normalized Data

• One of the lessons we learned was that data in this form was difficult to keep consistent because each record stood on its own.

• The customer's name and address appeared many times, because this data was repeated whenever a new order was taken.

• Inconsistencies in the data become rampant, because all of the instances of the customer address were independent.

• Updating the customer's address was a messy transaction (update, insert, & delete anomalies).
Relational database revolution

• We learned to separate out the redundant data into distinct tables, such as a customer and product, or student, course, instructor -- *but* we pay a price.

• Software systems for retrieving and manipulating the data became complex and inefficient because they required careful attention to the processing algorithms for joining (linking) these sets of tables together.

• We needed a database system that was very good at linking tables.

• This paved the way for the relational database revolution, where the database was devoted to just this task.
Relational database revolution

- The relational database revolution bloomed in the 1980s.
- *ER modeling and normalization* were developed in later years as the industry shifted its attention to *transaction processing*.
- The ER modeling technique is used to illuminate *relationships* among data elements.
ER modeling

• The highest art form of ER modeling is to remove all redundancy in the data.
• This is beneficial to transaction processing because transactions are made very simple and deterministic.
• The transaction of updating a customer's address may resolve to a single record lookup in a customer address table.
• Lookup is controlled by a customer key, which defines uniqueness of the customer address record and allows an indexed lookup that is extremely fast.
• Success of transaction processing in relational databases is in large part due to ER modeling.
Transaction processing

• In our zeal to make transaction processing efficient, we have lost sight of one of our most important goals.
• We have created databases that cannot be easily or efficiently queried!
• In a simple order-taking example a database of dozens of tables that are linked together by a bewildering spider web of joins.
ER model taken for sales app
ER model for the enterprise

- All of us are familiar with the big chart on the wall of the DBA’s cubicle.
- The *ER model for an enterprise* can have *hundreds* of logical entities!
- High-end systems such as SAP have *thousands* of entities.
- Each of these entities usually turns into a physical table when the database is implemented.
- End users cannot understand or remember an ER model.
- Software cannot usefully query a general ER model.
A simpler design

- **Simpler** designs all look very similar! Almost all of these simpler designs can be thought of as "*dimensional.*"
- In a natural, almost unconscious way, many designers returned to the roots of the original relational model because they know the database cannot be used unless it is packaged simply.
What is DM?

• DM is a logical design technique that seeks to present the data in a **standard, intuitive framework** that allows for **high-performance access**.

• It is inherently *dimensional*, and it adheres to a discipline that uses the relational model with some important restrictions.

• Every dimensional model is composed of one table with a **multipart key**, called the **fact table**, and a set of smaller tables called **dimension tables**.
What is DM?

• Each dimension table has a **single-part primary key** that corresponds exactly to **one of the components of the multipart key** in the fact table.

• This characteristic "**star-like**" structure is often called a **star join**. The term star join dates back to the earliest days of relational databases
ER model taken for sales app
Dimensional model for retail sales
Fact Table

• A fact table, because it has a multipart primary key made up of two or more foreign keys, always expresses a many-to-many relationship.

• The most useful fact tables also contain one or more numerical measures, or "facts," that occur for the combination of keys that define each record.

• Additivity is crucial because data warehouse applications almost never retrieve a single fact table record;

• Rather, they fetch back hundreds, thousands, or even millions of these records at a time, and the only useful thing to do with so many records is to add them up.
Dimension Tables

• Dimension tables contain descriptive textual information.
• Dimension attributes are used as the source of constraints in data warehouse queries, and they are virtually always the source of the row headers in the SQL result/answer set.
  – Ex. we constrain the Lemon flavored products via the Flavor attribute in the Product table, and on Radio promotions via the AdType attribute in the Promotion table.
• The power of the database is proportional to the quality and depth of the dimension tables.
• The database design is highly recognizable to the end users in the particular business.
How many TV's sold on Monday?

```
select p.description, f.dayofweek, count(*) as n
from productdimension p, timedimension t, fact f
where f.productkey=p.productkey
    and p.product='HDTV'
    and f.timekey=t.timekey
    and t.dayofweek='Sunday'
group by p.description, f.dayofweek;
```
DM vs. ER

• The key to understanding the relationship between DM and ER is that a single ER diagram breaks down into multiple DM diagrams.
  – Think of a large ER diagram as representing every possible business process in the enterprise.

• The second step is to select those many-to-many relationships in the ER model containing numeric and additive nonkey facts and to designate them as fact tables.

• The third step is to denormalize all of the remaining tables into flat tables with single-part keys that connect directly to the fact tables. These tables become the dimension tables.
Strengths of DM

1. DM is a *predictable, standard framework*.
   - Report writers, query tools, and user interfaces can all make strong assumptions about the dimensional model to make the user interfaces more understandable and to make processing more efficient.

2. DM star join schema *withstands unexpected changes* in user behavior.
   - All dimensions can be thought of as symmetrically equal entry points into the fact table.
Strength of DM

3. DM is gracefully *extensible* to accommodate unexpected new data elements and new design decisions.

4. There is a body of standard approaches for handling *common modeling situations* in the business world.

5. Growing body of administrative utilities and software processes that manage and use *aggregates*. 

• The **star schema** consists of one or more fact tables referencing any number of dimension tables.
The **snowflake schema** is a variation of the star schema, featuring normalization of dimension tables.

"**Snowflaking**" is a method of normalizing the dimension tables in a STAR schema.

When it is completely normalized along all the dimension tables, the resultant structure resembles a snowflake.
Dimensional Genomics Search Engine

- Multi-modal dimensional data model capable of document, paragraph, sentence, and passage level retrieval.

- Composite retrieval models that combine multiple levels of contextual evidence.

- All modes of retrieval can benefit from different levels of evidence.
Dimensional Index Model
Dimensional Data Model – Concept and Term Index
select p.docid, max(d.docnum) docnum,
    sum( ln((s.ndocs-i.df+0.5)/(i.df+0.5))*
    (((k1+1)*p.tf)/(k1*((1-b)+b*(d.len/s.avgdoclen))+p.tf))*
    ((k3+1)*q.tf/(k3+q.tf)) ) as sc
from index i, documents d, query q, indexstats s,
( select p2.docid, p2.termid, sum(p2.tf) tf
from postinglist p2, invertedindex i2, query q2
where i2.termid=p2.termid
and i2.term=q2.term
    group by p2.docid, p2.termid ) p
where p.docid=d.docid
and i.termid=p.termid
and i.term=q.term
    group by p.docid
DM Summary of Advantages

- Adding new unanticipated facts -
  - New additive numeric fields in the fact table, as long as they are consistent with the fundamental *grain* of the existing fact table.

- Adding completely new dimensions, as long as there is a single value of that dimension defined for each existing fact record.

- Adding new, unanticipated dimensional attributes.

- Breaking existing dimension records down to a lower level of granularity from a certain point in time forward.