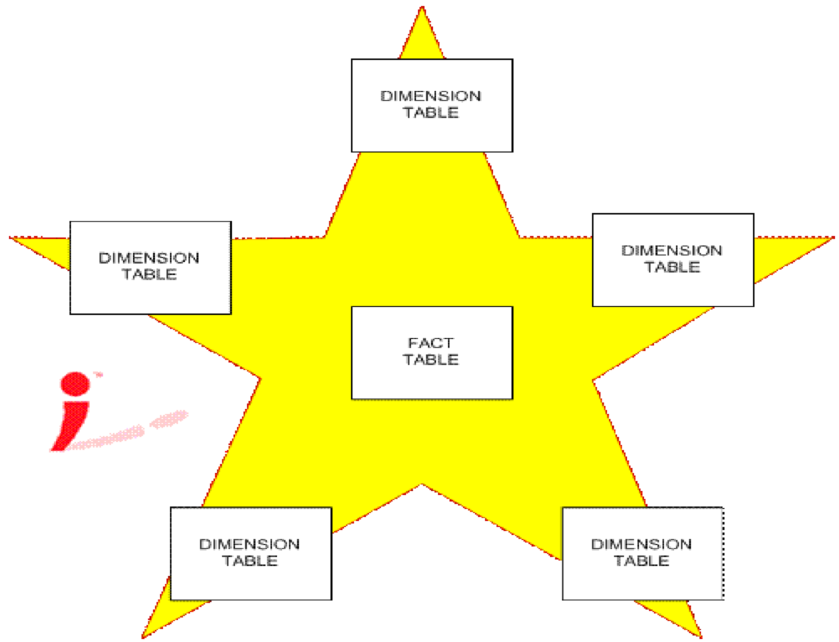
Advantages of Dimensional Data Modeling



SYSTEMS SEMINAR CONSULTANTS, INC.

2997 Yarmouth Greenway Drive Madison, WI 53711

(608) 278-9964 www.sys-seminar.com

Top Ten Reasons Why Your Data Model Needs a Makeover

1. **Ad hoc queries are difficult to construct for end-users or must go through database “gurus.”**
2. **Even standard reports require considerable effort and detail knowledge of the database**.
3. Data is not integrated or is inconsistent across sources.
4. Changes in data values or in data sources cannot be handled gracefully.
5. The structure of the data does not mirror business processes or business rules.
6. The data model limits which BI tools can be used.
7. There is no system for maintaining change history or collecting metadata.
8. Disk space is wasted on redundant values.
9. Users who might benefit from the data don’t use it.
10. Maintenance is tedious and ad hoc.

Advantages of

Dimensional Data Modeling

Part 1

Part 1 - Data Model Overview

- What is data modeling and why is it important?

- Three common data models:
    - de-normalized (SAS data sets)
    - normalized
    - dimensional model

- Benefits of the dimensional model

What is data modeling?

- The generalized logical relationship among tables
- Usually reflected in the physical structure of the tables
- Not tied to any particular product or DBMS
- A critical design consideration

Why is data modeling important?

- Allows you to optimize performance
- Allows you to minimize costs
- Facilitates system documentation and maintenance

- *The <u>dimensional data model</u> is the foundation of a well designed data mart or data warehouse*

Common data models

Three general data models we will review:

De-normalized

Expected by many SAS procedures

Normalized

Often used in transaction based systems such as order entry

Dimensional

Often used in data warehouse systems and systems subject to ad hoc queries.

The dimensional model may be used for any reporting or query data even if not a “data warehouse”

The dimensional model is our focus here.

De-normalized Data

Sales Transaction Table

Each row represents a sale transaction line.

Transaction line table
Transaction number
Customer Name
Customer Street Address
Customer City
Customer State
Customer Zip
Multi-state region
Product Category
Product Number
Product Name
Calendar day
Day of week
Month
Year
Season
Annual product cycle number
Sale quantity
Sale dollar amount

Attributes of the sale: customer and product info, date, etc.

All attributes of the sale are included with each transaction line row.

Sale facts: number of items and dollars

De-normalized Data in SAS Procedures

- SAS data sets are commonly structured as de-normalized data
- Many SAS procedures that do grouping expect de-normalized data
- Low cardinality attributes are CLASS variables – often character type
- High cardinality measures or facts are VAR variables – numeric type

```
PROC MEANS DATA=TRANSACTION SUM;
   CLASS STATE;
   VAR AMOUNT;
RUN;
```

Sale attribute

Sale fact

De-Normalized Data

A single row contains:

- Numeric facts or measurements and…
- All attributes related to that measurement

All data is in a single table.

Data redundancy:

Directly correlated attributes, such as product number and product category, are repeated in each row

Sale Number	Product Number	Product Category
1	S3200	Software
2	S3223	Software
3	H7005	Hardware

Normalized Data

Transaction table
- Transaction number
- Customer name

Customer table
- Customer name
- Customer street address
- Customer zip code

Zip table
- Zip code
- City
- State

State table
- State
- Multi-state region

Transaction line table
- Product number
- Transaction number
- Date
- Sale quantity
- Sale dollar amount

Product table
- Product number
- Product category
- Product name

Data is broken up into related tables

Month table
- Month
- Season

Date table
- Date
- Calendar day
- Month
- Year

Year table
- Year
- Annual product cycle number

Normalized Data

•*Insert Optimized*
A new transaction line involves gathering only the five data items in the Transaction Line table. No other attribute look up is required.

•*Redundancy is reduced:*

For example, Product Category is not repeated for each transaction

•*Changes have less impact on the database:*

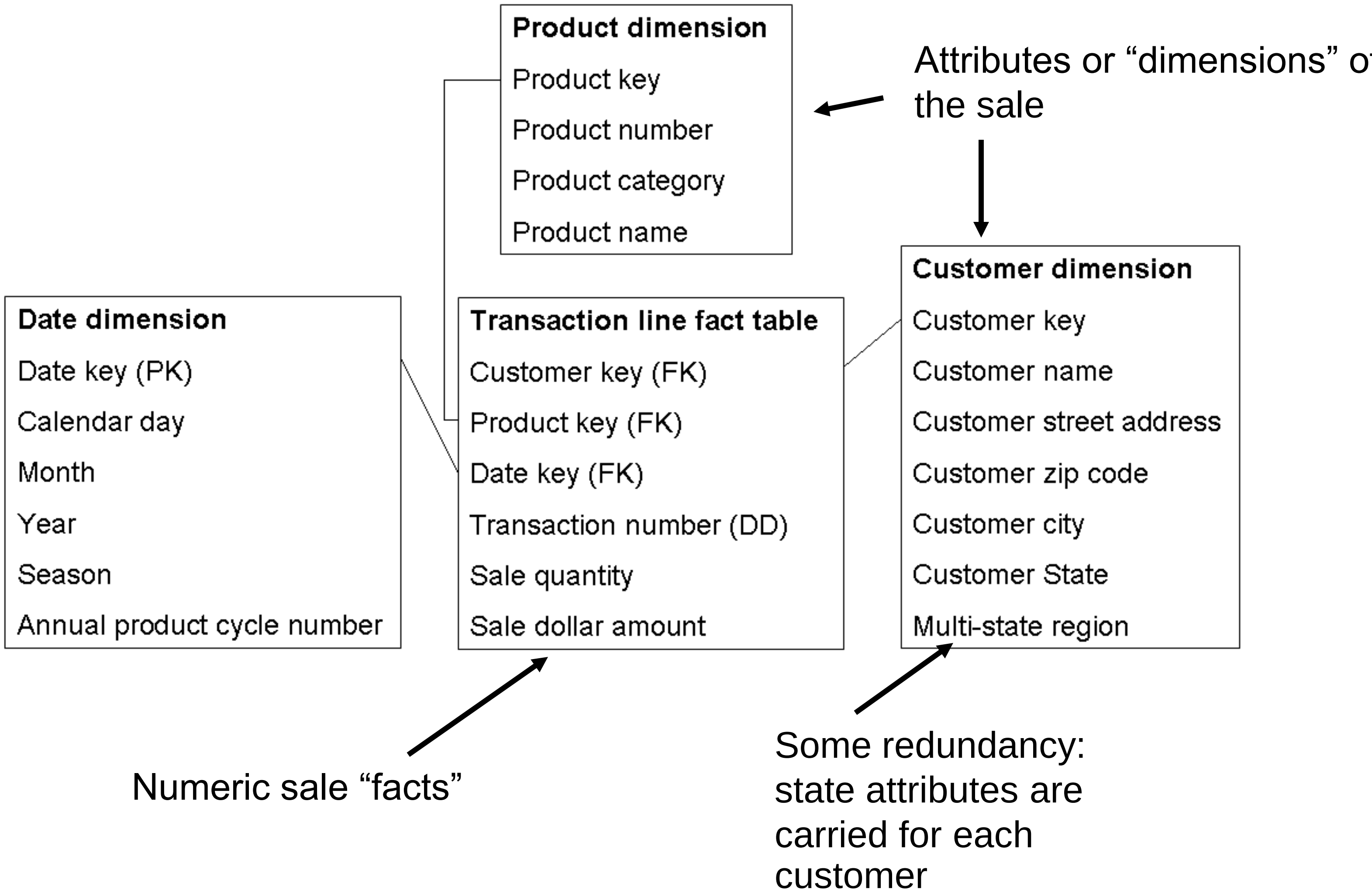
If a product category changes, only the Product table needs to be changed

•*<u>Query complexity is increased</u>:*

Several tables must be related to each other in order to answer simple questions.

What is the sum of sale amount for each state?

Dimensional Data



Dimensional Data

•*The central fact table is surrounded by dimension tables*

Star schema

•*Table relationships are only one level deep*

No more than two tables need to be joined together for common business questions and aggregations

What is the sum of sale amount for each state?

Facts and Dimensions

•Key terms: Fact and Dimension

•*Fact:*

High cardinality, numeric measure of some event such dollars for a sale.

Typically many rows, one per business event.

•*Dimension:*

Low cardinality, typically character, attribute of a fact.

Typically many columns, one per attribute of interest.

The dimensional model is made up of facts and dimensions

What can dimensional modeling do for your organization?

- Bring together data from many different sources and create a **single**, **consistent** user view.
- Support the **ad hoc queries** that arise from **real business questions**.
- Maximize **flexibility** and **scalability.**
- Optimize the **end-user** experience.

What's Next?

Part 2 – The Dimensional Data Model

- Facts and dimensions explained
- Granularity
- Why use surrogate keys?
- Drill down and drill across queries in dimensional data
- Introduction to slowly changing dimensions
- Benefits of dimensional modeling

Advantages of

Dimensional Data Modeling

Part 2

Fact and Dimensions

Dimensional modeling implies two distinct types of data:

1. Facts
2. Dimensions

These data are stored two types of tables:

1. Fact tables
2. Dimension tables

Facts and Dimensions

A fact is…

- A business measurement, amount, or event
- Typically numeric, continuously valued, and additive
- Something we analyze: “What were total <u>sales</u> by state?”

Some facts:

revenue dollars, unit counts, event counts

A dimension is…

- Context surrounding a fact: who the fact applies to; when, where, and under what conditions the fact was measured
- Usually a discrete character or numeric value
- Static or slowly changing
- Something we use to identify or group data: “What were total sales by <u>state</u>?”

Some dimensions:

customer, date, time, location

Fact Table

Elements of a fact table:

- **Fact:** the measure(s) of interest
- **Dimension foreign key:** Key to a row in a dimension table

Sales Transaction Fact Table

Date key (FK)

Product key (FK)

Channel key (FK)

Promotion key (FK)

Customer ID (FK)

Sales quantity

Sales dollar amount

Cost dollar amount

Dimension Table

The dimension table represents an entity of interest to the business: Customer, product, vendor, promotion, etc.

Elements:

- **Primary key (PK):** Unique for each row in the table. It should be a surrogate key, i.e., have no inherent meaning. The value of the dimension key is what's stored in the fact table.
- **Dimension attributes**: A set of variables that encompass what is known about the business entity.

Customer Dimension

Customer ID (PK)

Customer Name

City

State

Zip

Date of first contact

Fact-Dimension Data Model

Date Dimension
- Date key (PK)
- Date
- Day of week
- Calendar month
- Calendar year
- Holiday

Channel Dimension
- Channel ID (PK)
- Channel name
- Channel description
- Channel type

Sales Transaction Fact Table
- Date key (FK)
- Product key (FK)
- Channel key (FK)
- Promotion key (FK)
- Customer ID (FK)
- Sales quantity
- Sales dollar amount
- Cost dollar amount

Product Dimension
- Product key (PK)
- Product Description
- Category Description

Promotion Dimension
- Promotion key (PK)
- Promotion name
- Promotion type
- Promotion start date
- Promotion end date

Dimension
- Customer ID (PK)
- Customer Name
- City
- State
- Zip

This relationship pattern is called a star schema.

Granularity

- Granularity is the level of detail in a fact table
- Granularity is the combination of all dimensions

The grain of the previous table is:

Date

Product

Channel

Promotion

Customer

- Only facts with the same grain (i.e. described by the same dimensions) can coexist in a fact table.
- Granularity can always be reduced through aggregation, but can never be increased.

Granularity

- The fact tables represent two different business processes.
- The fact tables each have a unique set of foreign keys, though some foreign keys match (red).

Sales Transaction Fact Table

Date key (FK)

Product key (FK)

Channel key (FK)

Promotion key (FK)

Customer ID (FK)

Sales quantity

Sales dollar amount

Cost dollar amount

Each line is a sales transaction– one customer buying some quantity of one product.

Promotion Event Fact Table

Date key (FK)

Promotion key (FK)

Medium key (FK)

Customer ID (FK)

Count variable

Each line is a promotion event– one customer being offered one promotion.

Surrogate Key

Each row in a dimension table should be identified by a surrogate primary key. A surrogate key has no inherent meaning.

Surrogate key

Natural key

Channel ID (PK)	Channel Name	Channel Description	Channel Type
1042	Store #0720	St. Louis Retail Store	Retail Store
1043	Store #0721	Albuquerque Street Kiosk	Kiosk
1044	Store #0722	Scranton Retail Store	Retail Store
1045	Store #0720	St. Louis Outlet Store	Outlet Store

- The two records for Store #0720 (natural key) can coexist without conflict because each has a unique surrogate key.

Surrogate Key

Benefits of using a surrogate key:

- Surrogate keys make it possible to integrate data from sources that use different forms of a natural key.
- Allow the use of legitimate unknown and null natural keys, or natural keys with special meanings.
- Natural keys may be reused. For example, transaction numbers may be recycled six months after the transaction. A unique surrogate key value distinguishes between two like-numbered transactions.

Drill Down and Up

Drill down means displaying facts at a lower level of granularity.

When you drill down you add dimensional attributes.

Example:

I am viewing sales by state and I want to drill down to the zip code level within state.

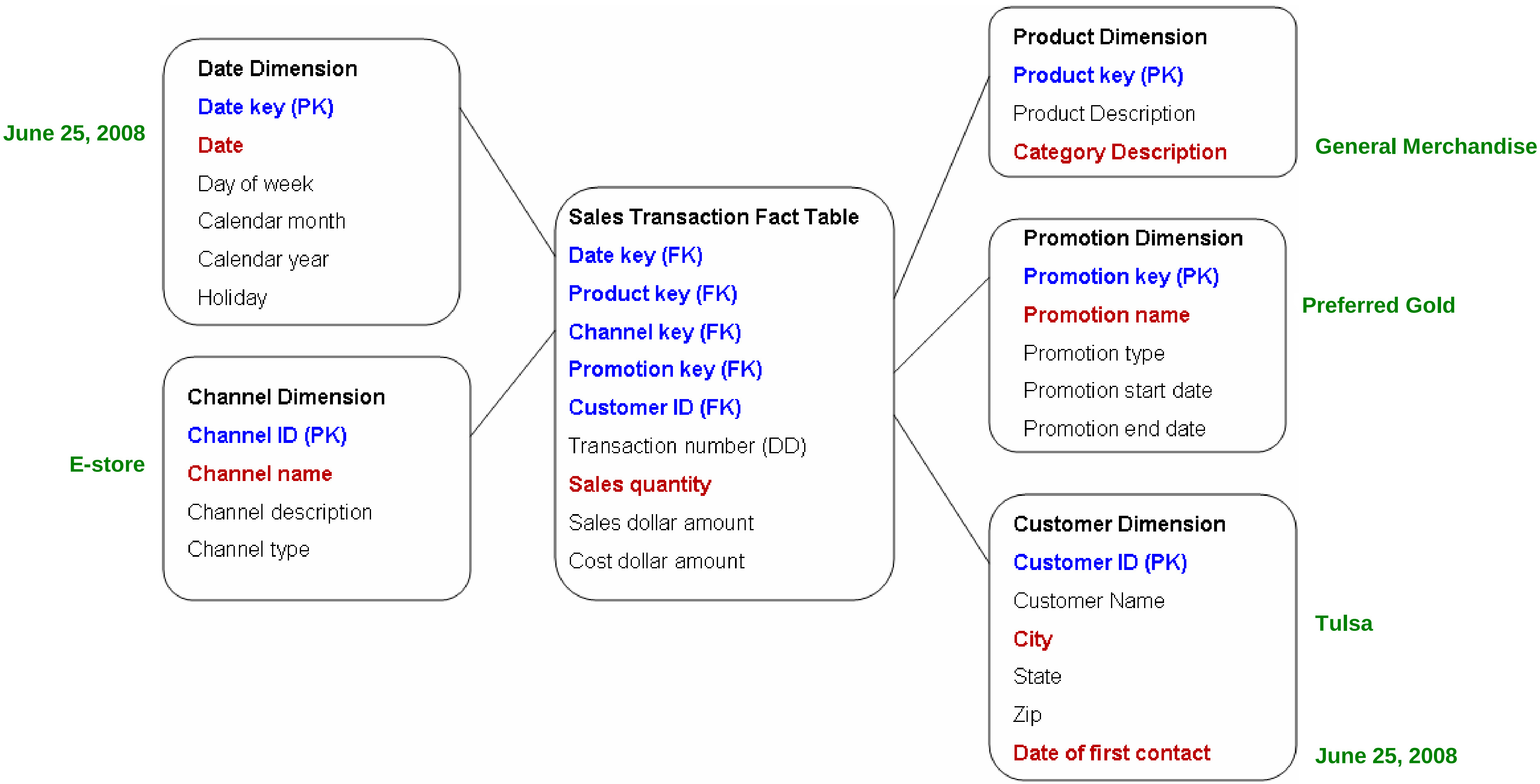
Drill up is the reverse. Drill up reduces the number of dimensional attributes.

Drill up is aggregation.

Example:

I am viewing sales state but want sales aggregated by multi-state region.

Drill Up and Down



Drill Across

Drill Across means:

Join two or more facts that share the same dimensions.

Consider the question…

"How many customers who purchased products this December were notified of the Year End Clearance promotion by e-mail?"

The answer involves two different facts:

1. Sale events
2. Promotion events

Sales facts and promotion facts can be joined on their common dimension: customer

Drill Across Query

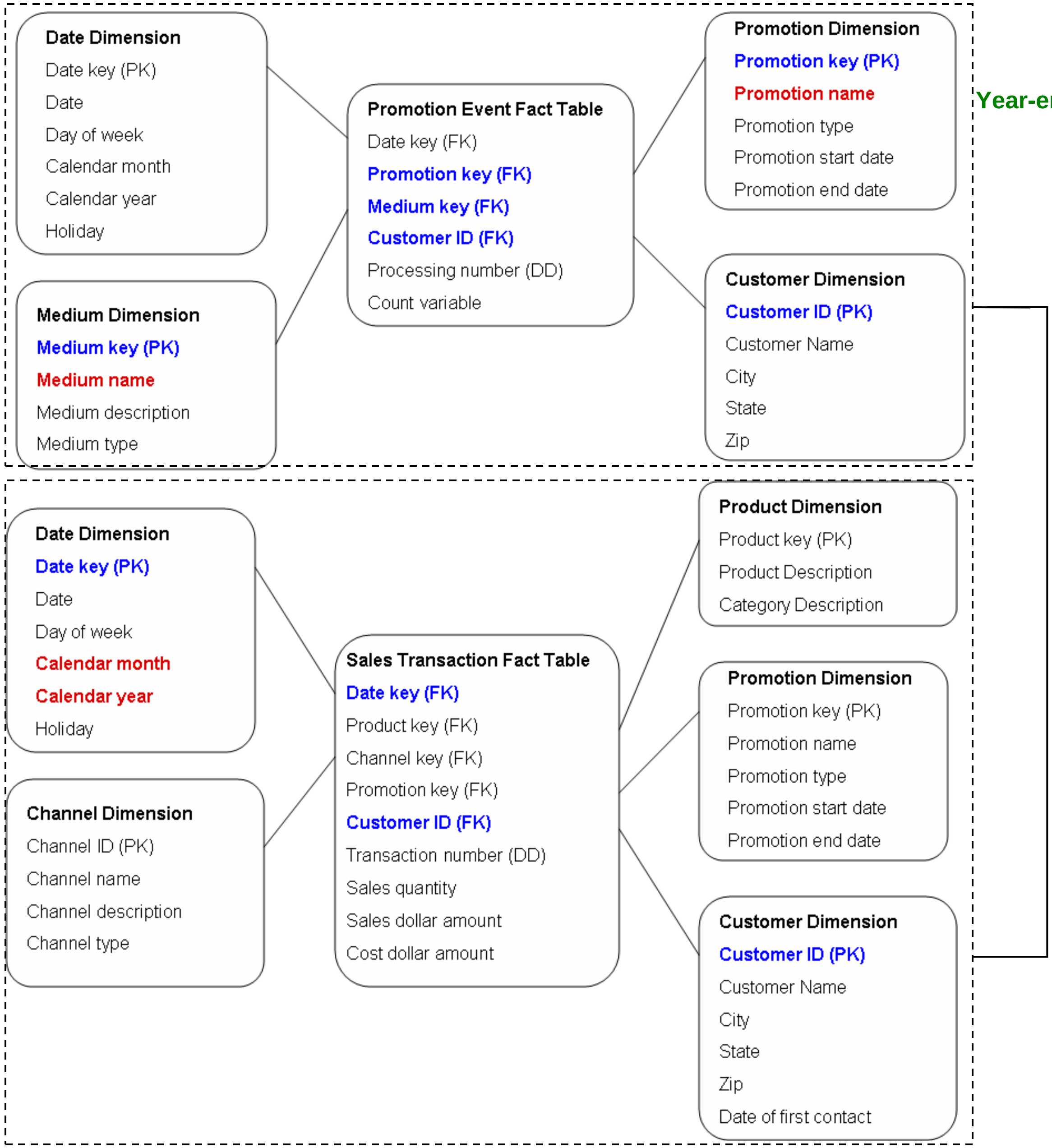
Promotion Event Schema

Year-end Clearance

E-mail

December 2007

Sales Transaction Schema



The shared customer dimension allows for a join on Customer ID

Conformed Dimensions

Criteria for conformed dimensions:

- Like-entities represented in different tables have the same primary key
- One set of dimension attributes may be a subset of the another
- Like-named attributes are equivalent– they have the same meaning and the same range of values.

Customer Dimension

Customer ID (PK)

Customer Name

City

State

Zip

Date of first contact

Customer dimension from the Sales Transaction schema

Customer Dimension

Customer ID (PK)

Customer Name

City

State

Zip

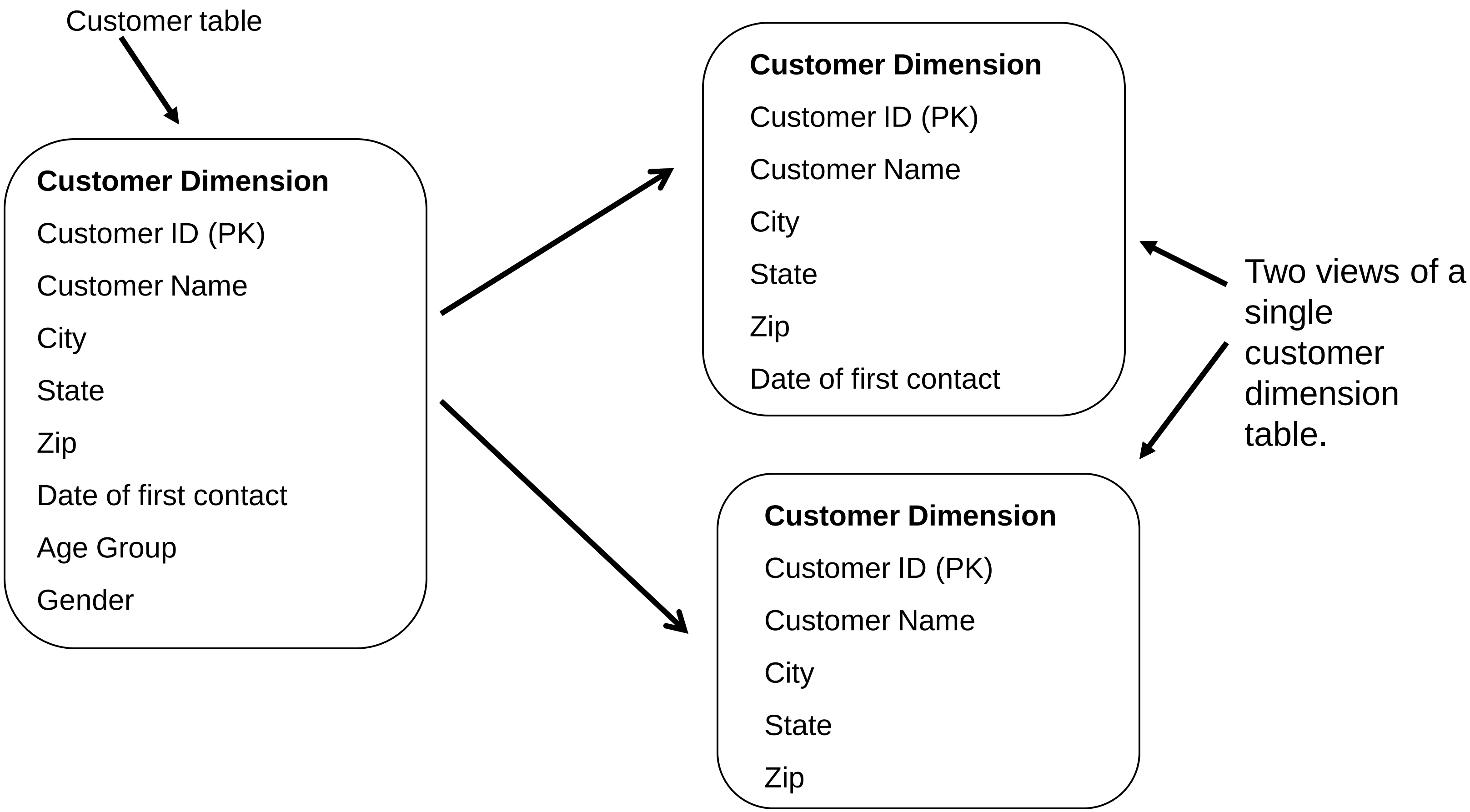
Customer dimension from the Promotion Event schema.

A subset of transaction customer

Conformed Dimensions

"Conformed dimension" is a logical concept.

The conformed dimension that is shared between two dimensional models may be a single physical table.



Slowly Changing Dimensions

The value of a dimension attribute may change.

For example, the Channel Type for Store #0720 may change:

Channel ID (PK)	Channel Name	Channel Description	Channel Type
1042	Store #0720	St. Louis Retail Store	Retail Store

Channel ID (PK)	Channel Name	Channel Description	Channel Type
1042	Store #0720	St. Louis Outlet Store	Outlet Store

Slowly Changing Dimensions

There are three generally accepted ways to handle slowly changing dimensions (SCD):

Type 1 – simply replace the value old attribute value with the new.

Type 2 – insert a new dimension row, with a new key, representing the changed attribute. The old version of the dimension, with its original key, remains.

Type 3 – Design the dimension table with columns that hold previous values of the attribute anticipated to change.

SCD Type 1

Channel ID (PK)	Channel Name	Channel Description	Channel Type
1042	Store #0720	St. Louis Retail Store	Retail Store

Channel ID (PK)	Channel Name	Channel Description	Channel Type
1042	Store #0720	St. Louis Outlet Store	Outlet Store

SCD Type 2

Channel ID (PK)	Channel Name	Channel Description	Channel Type
1042	Store #0720	St. Louis Retail Store	Retail Store

Channel ID (PK)	Channel Name	Channel Description	Channel Type
1042	Store #0720	St. Louis Retail Store	Retail Store
1099	Store #0720	St. Louis Outlet Store	Outlet Store

SCD Type 3

Channel ID (PK)	Channel Name	Current Channel Type	Previous Channel Type
1042	Store #0720	Outlet Store	Retail Store

•SCD Type 3 tables often include columns indicating when a change happened.

•SCD Type 3 designers must anticipate then number of change events to store.

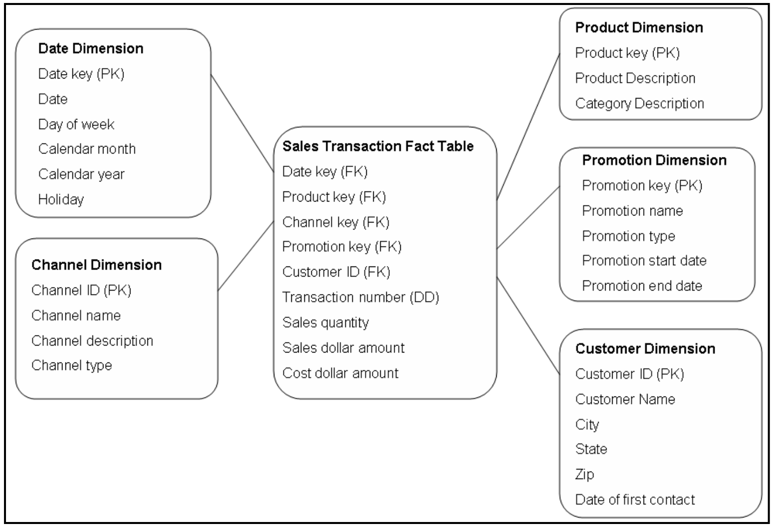
Using Views

•You can use views to “flatten” a dimensional model.

•This allows you to use many base SAS PROCs and may make more sense to some end users.

Using Views

Underlying model of five related tables:



User sees a single "table":

Date	Transaction #	Customer	Product	Quantity
06/25/08	02340982490A	Max Chen	Deluxe Widget	5
06/25/08	09283490238H	Norie Hamm	Regular Widget	1
06/25/08	82098349028C	Andy Henson	Super Deluxe Widget	89

Using Views

```
proc sql;
  create view transactions as
  select d.date, s.transactionNumber, cu.customerName,
         p.productDescription, s.quantity, *
  from db.sales    s                        left join
       db.date     d
         on s.dateID=d.dateID               left join
       db.channel c
         on s.channelID=c.channelID          left join
       db.product p
         on s.productID=p.productID          left join
       db.promotion pr
         on s.promotionID=pr.promotionID left join
       db.customer  cu
         on s.customerID=cu.customerID;
quit;

proc means data=transactions sum n mean;
  class customerName productDescription;
  var quantity;
run;
```

What can dimensional modeling do for your organization?

Bring together data from many different sources and create a **single**, **consistent** user view .

❖ **Single version of the truth**

The dimensional model applies business rules so the same fact or dimensional attribute always has the same definition.

❖ **Data integration**

The dimensional model is built around data integration. The dimensional modeling process reveals inconsistencies and allows (or forces) them to be reconciled.

What can dimensional modeling do for your organization?

Support the **ad hoc queries** that arise from **real business questions**.

- **Analyze on the fly**

  The dimensional model facilitates ad hoc queries and unanticipated business questions because it is generic and not tied to any specific report structure or view of the data.

- **Drill up or drill down to any level of detail contained in the data**

  The dimensional model is a natural for summary reports and drill down applications. Dimensions are added for drill down, removed for summaries. Commonly used summaries may be pre-aggregated for improved performance.

What can dimensional modeling do for your organization?

Maximize **flexibility** and **scalability.**

- **Enterprise-wide data warehouse or specialized data mart**

  The dimensional model works equally well with generalized corporate data warehouse schemes or “data marts” focused on specific departments or user groups. Small scale data marts can be expanded and large warehouse structures can be sub-setted to change your project scale in either direction.

- **Tool agnostic**

  Almost any BI tool supports dimensional models. You can use your favorite query tool while someone in the next department accesses the same data with a modeling application. SQL queries against a dimensional model all have the same general structure.

- **The data warehouse evolves with the organization**

  Adding new data sources and adapting to changes in current data sources is handled in a consistent, reproducible manner.

What can dimensional modeling do for your organization?

Optimize the **end-user** experience.

- ❖ **The dimensional model is all about queries**

  The dimensional model is designed to make queries consistent, understandable, and fast. The dimensional model makes business data available to more users because query structure is less of a mystery.

- ❖ **Understandable**

  In the dimensional model, data relationships are consistent and typically no more than one level deep. This makes the data structure more understandable for experts and casual users alike. It also facilitates documentation and meta-data set up.

Contact Us With Questions

SYSTEMS SEMINAR CONSULTANTS, INC.

SAS® Training, Consulting, & Help Desk Services

(608) 278-9964

train@sys-seminar.com

www.sys-seminar.com

2997 Yarmouth Greenway Drive

Madison, WI 53711

sas® | SAS Alliance *Affiliate Member*