

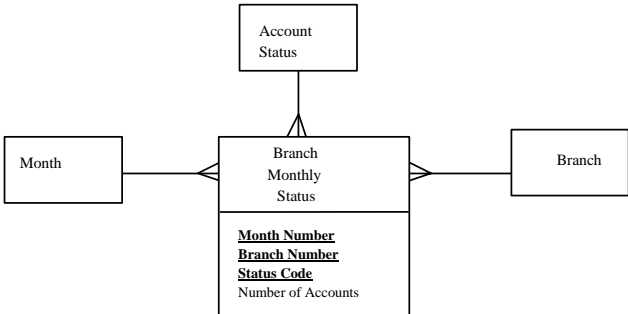
# Dimensional Modeling

DAMA Philadelphia  
May 11, 2005  
Tom Haughey  
President  
InfoModel LLC  
868 Woodfield Road  
Franklin Lakes, NJ 07417  
201 337 9094  
[tom.haughey@InfoModelUSA.com](mailto:tom.haughey@InfoModelUSA.com)  
(cell) 201 755 3350



## Sample Query Exercise

*Give me the number of accounts by month by branch by status.*





## Strategic Versus Tactical Decisions

• Strategic Decisions	• Tactical Decisions
How many orders, by product, were shipped more than 30 days late over the past year?	What backorders from the 30 day backlog are scheduled to be shipped today?
Is there a pattern, based on average household income, of students who default on loans?	What is the list of names and the associated addresses of the students currently more than 30 days late on their loans? (For mailing overdue notices)
What color car has typically sold the best by sales region over the last 5 years?	What other dealers have a white Product XYZ in stock right now?
What is the seasonal pattern in sales by model, by geographic area?	Is my dealership on target for this month compared to my objectives?



## Operational Vs. Informational Data

### Operational Data

- detailed
- normalized
- current
- volatile
- transaction oriented
- performance sensitive
- high availability
- critical to daily operations

### Informational Data

- detailed/summarized
- normalized/denormalized
- historical
- mostly read-only
- analysis oriented
- not performance sensitive
- not high availability
- used to analyze operations





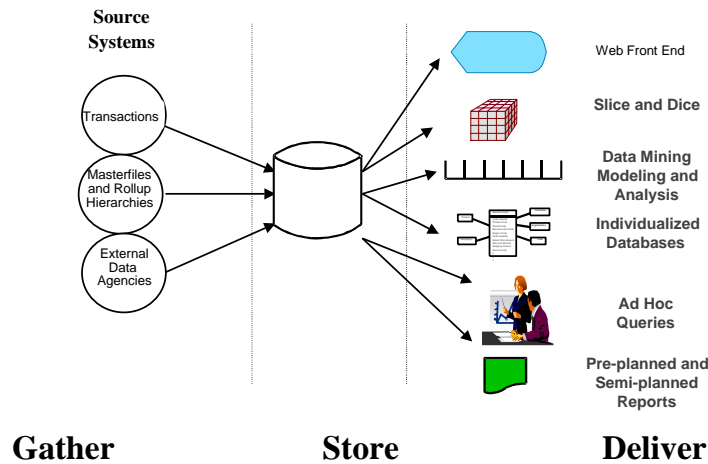
## Three Major Trends Today

- **The data warehouse**
  - A strategic system for the control and presentation of informational data for the purpose of managing the business
- **The operational data store**
  - A tactical system containing the near real-time committed transactions for tactical purposes for a limited period of time
  - Built for three main purposes:
    - To consolidate operational data systems that are non-integrated or distributed
    - For tactical consolidated reporting or
    - For integrated update processing
- **Data mining**
  - The use of mathematical algorithms to determine hidden relationships in apparently unrelated data

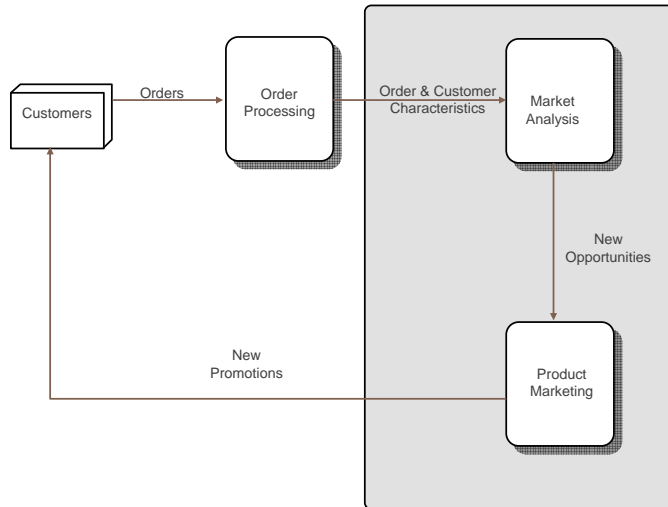


## What Does a Data Warehouse Look Like?

- A data warehouse is an environment and system for gathering data from multiple sources, enhancing and storing the data in a integrated database, and delivering information to business people.



## Where Is The Greatest Opportunity?



- A Data Warehouse makes good business sense

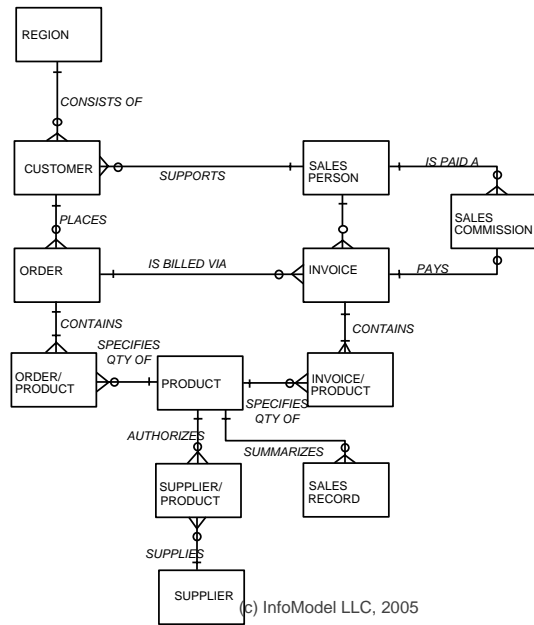


## Characteristics Of A Warehouse

- A data warehouse is a:
  - Management-oriented
  - Subject-oriented
  - Integrated
  - Historical (time-variant)
  - Read-only (non-volatile)
  - **Controlled**
- collection of data in support of management's decision making process

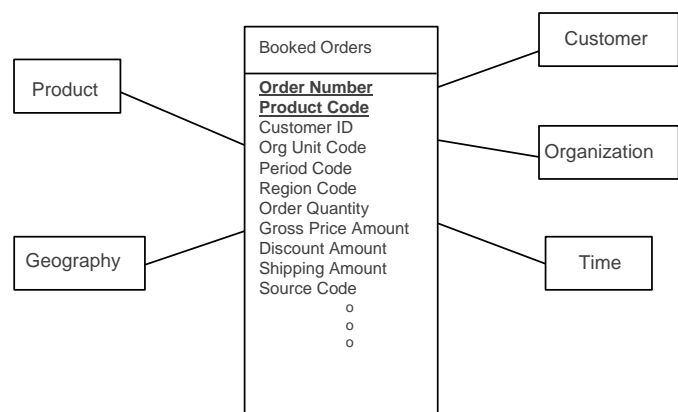


## An Operational Model



(c) InfoModel LLC, 2005

## An Analytical Model



(c) InfoModel LLC, 2005



## What Is A Dimensional Model?

---

- A dimensional model is a model in which the data is structurally classified as fact or dimension.
- General characteristics:
  - Query oriented
  - Structured around data usage not business rules
  - Organized roughly into base facts and dimensions of those facts
  - Based on identification of key grains of data
  - Consisting usually of pre-joined data
  - Looks to reduce the number and depth of joins
- Levels of dimensionality can differ:
  - From very atomic to highly aggregated
  - From few dimensions to many
  - Facts usually contain three or more dimensions



## Two General Types of Tables

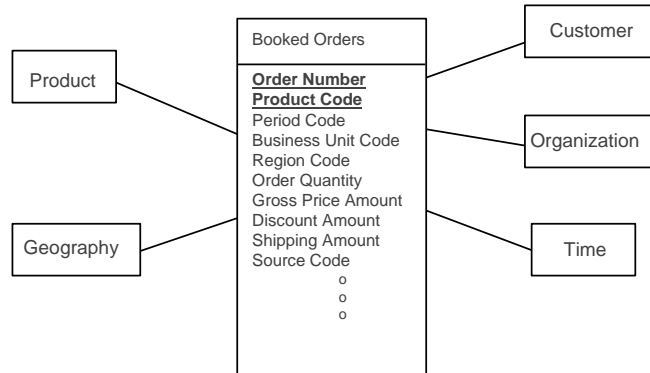
---

- Fact Tables
  - Contain the numeric values of interest to the business analyst
  - Represent the natural facts found in the business and the dimensions by which they are identified and measured
  - Are the base values used in analyzing the business
  - Are derivable - i.e., summarization of base amounts from detailed data
- Dimension Tables
  - Consists of the constraints used in forming and examining the fact table
  - They contain mostly descriptive elements used individually or in various combinations to identify the facts



## The Fact Table

- The central (major) table in a Star Schema
- Largest table in the warehouse database.
- Dimension tables share primary/foreign key relationships with the fact table
- The Fact Table is highly normalized since it resides at the intersection of a many-to-many relationship among uncorrelated dimensions.



(c) InfoModel LLC, 2005

13

## Does the DW Model Have to be Dimensional?

- The DW model has to satisfy information requirements and must do so within performance expectations.
  - It must use whatever data compromises are necessary to achieve this
  - Many factors play into this
    - All of them tangible
    - None of them emotional
  - Because of the vast amounts of data in the data warehouse, we will see that there are many levels of data and potentially many levels of optimization
  - We start with a logical model of the information requirements
    - Business oriented
    - Independent of technology
    - Independent of implementation
  - Since the DW is to support reporting/analysis, we start with dimensional modeling concepts because they aid comprehension of the analytical/reporting problem domain



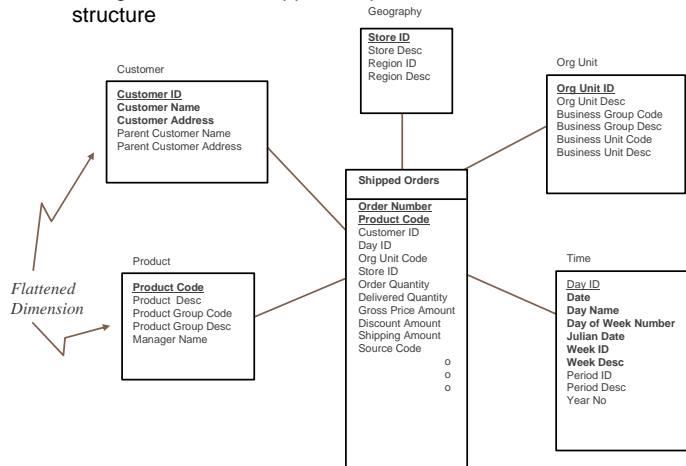
(c) InfoModel LLC, 2005

14



## Classic Star

- A fact surrounded by a **single circle** of dimensions
  - ↳ Multi-levelled dimensions are flattened
  - ↳ Designed for direct support of queries that have an inherent dimension-fact structure



(c) InfoModel LLC, 2005

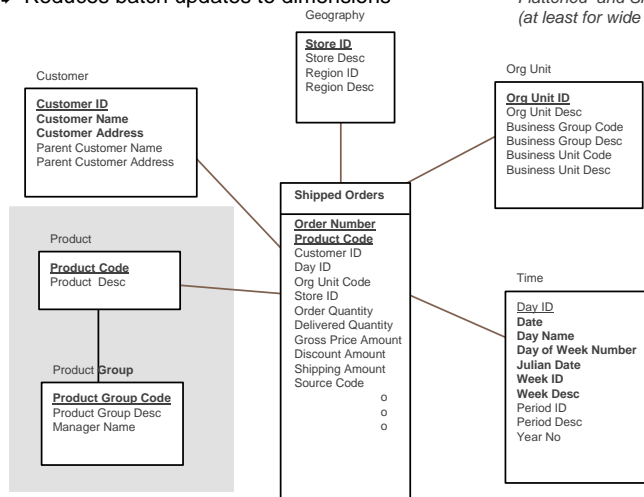
15



## Snowflake

- Separate levels of a dimension are kept separate\*
  - ↳ More flexible
  - ↳ Reduces batch updates to dimensions

\* Tho always said to be slower than a star, some tests have revealed no difference in performance between Flattened and Snowflaked Dimensions (at least for wide dimensions).



\* To simplify the visual, only one dimension is shown as a snowflake. Any or all dimensions could be snowflaked.



(c) InfoModel LLC, 2005

16



## Snowflake Schema

---

- Suitable for many-to-many and one-to-many relationships among related dimension levels
- Required for many-to-many fact-dimension relationships (e.g., customer-policy)
- **Pros**
  - ✦ Less storage space. However, normalizing the Star based solely on amount of storage saved is not warranted unless the volume is huge
  - ✦ Could improve performance, flexibility, and maintainability under some circumstances.
- **Cons**
  - ✦ May complicate the user understanding of the warehouse database.
  - ✦ Makes the database design seems more complex. Some hard-core OLTP database designers see it as a natural.
  - ✦ Could impact browsing performance due to extra joins required to access the Facts.



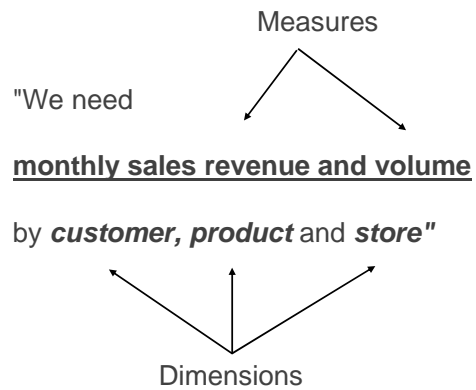
## Steps for Data Modeling for the DW

---

- 1. Identify the business process or question
- 2. Declare the necessary grain of data
- 3. Define the dimensions
- 4. Define the facts
- 5. Determine the summary levels



## 1. Identify the Business Process or Question



## 2. Declare the Grain

- Transaction
  - Singular transaction - Buy 100 shares of IBM
  - Composite transaction - Order and its Line Items
- Periodic Status
  - A fact that represents some business measure determined at the end of each regular, predetermined interval
  - Periodic status typical of balance forward businesses like banking and insurance
- Accumulating Snapshot
  - A fact that stores the complete timeline of a process, such as:
  - The complete lifecycle of an order
  - Rolling summaries



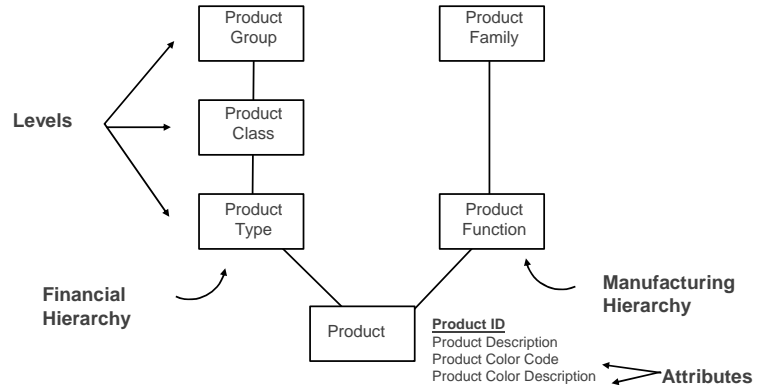
### 3. Identify the Dimensions

- Dimensions can have:

- Attributes
- Levels
- Multiple hierarchies

- Dimensions can be:

- Symmetric dimensions
- Asymmetric dimensions



(c) InfoModel LLC, 2005

21

### Conformed Dimensions

- Conformed dimensions means that when multiple copies of the same dimension exist they are consistent with one another in that they have:

- The same key
- The same values for the key
- The same granularity
- Attributes in each conformed dimension can be a subset and can overlap

- Must be built at a common level of granularity (or be a rollup of the base dimension).

- Generally identified by a surrogate primary key.

- Must establish an inviolable policy to reuse the conformed dimension whenever that subject is required for a data mart

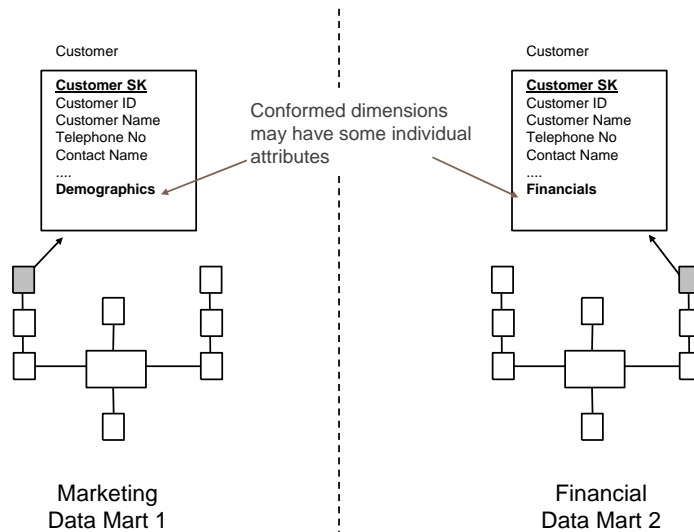
- Also called dependent dimensions or architected dimensions.



(c) InfoModel LLC, 2005

22

## Conformed Dimensions

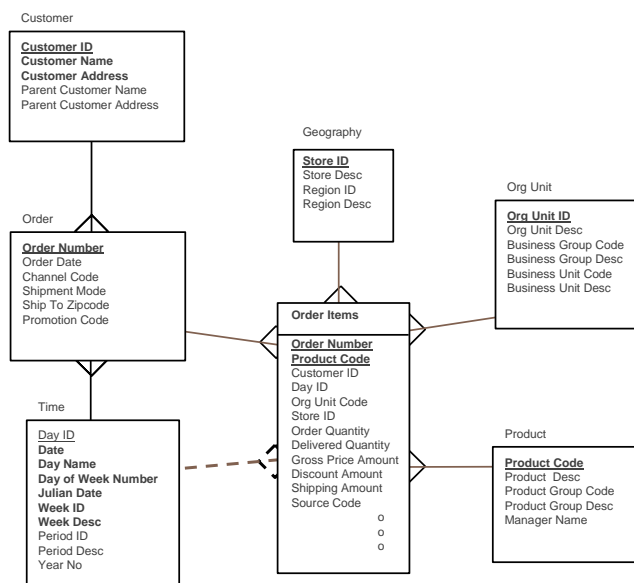


(c) InfoModel LLC, 2005

23

## 4. Define the Facts

- Define the most detailed grain necessary to support the business requirement
- This is an example of a Line Item Fact:
  - Very detailed
  - Most granular



(c) InfoModel LLC, 2005

24

## 5. Determine the Summary Levels

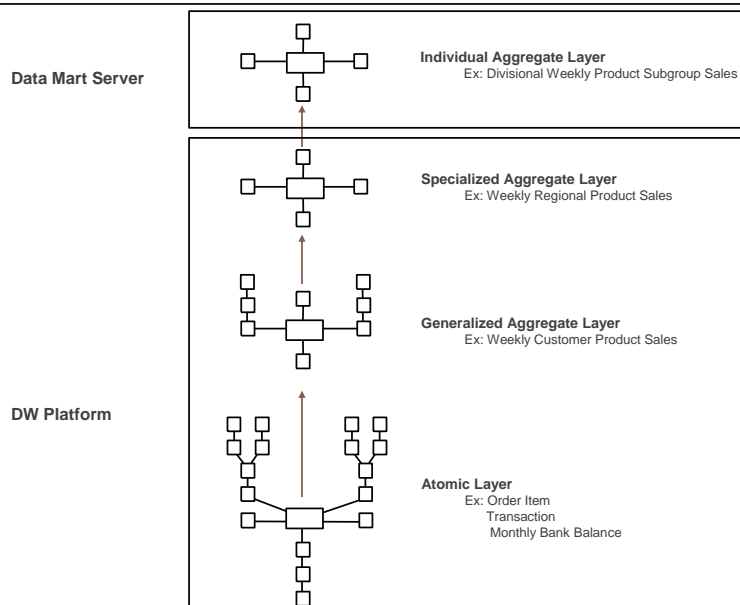
- The need for stored summaries can be determined in one of two ways:
  - By reacting to poor performance and
  - By analyzing queries in advance and predicting whether summaries are needed or not
- Which queries are required is determined by analyzing query data usage.
- There are three major steps for deciding on what summary levels to create:
  - First, build more generalized aggregates
  - If that does not give adequate performance, build more specialized aggregates
  - If that still does not provide adequate performance, then offload the aggregates to a separate server



(c) InfoModel LLC, 2005

25

## Layers of Data in the Data Warehouse



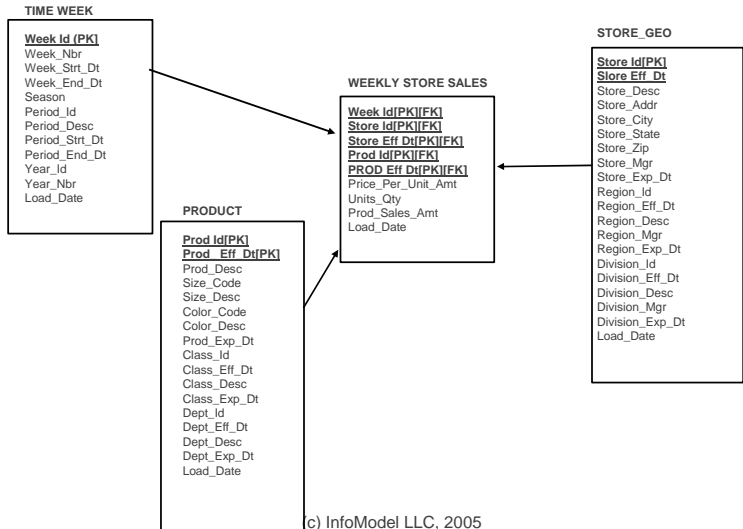
(c) InfoModel LLC, 2005

26



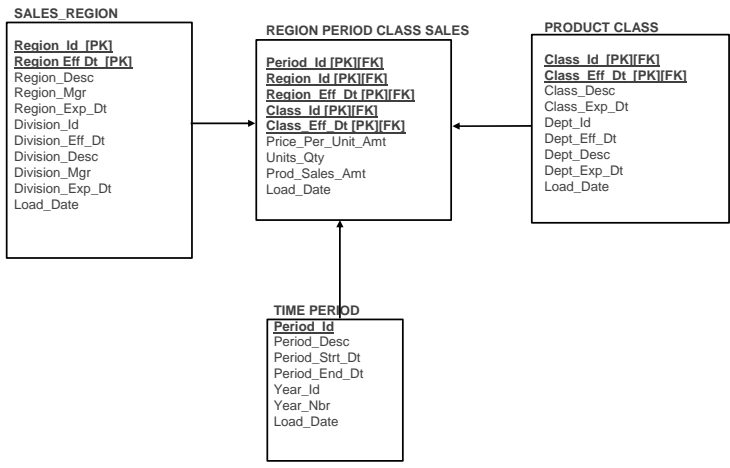
## First Summary Level Grain

- Decide on granularity of first summary level that you intend to store
  - In this example, Sales by Product by Store by Week.



## Second Summary Level Grain

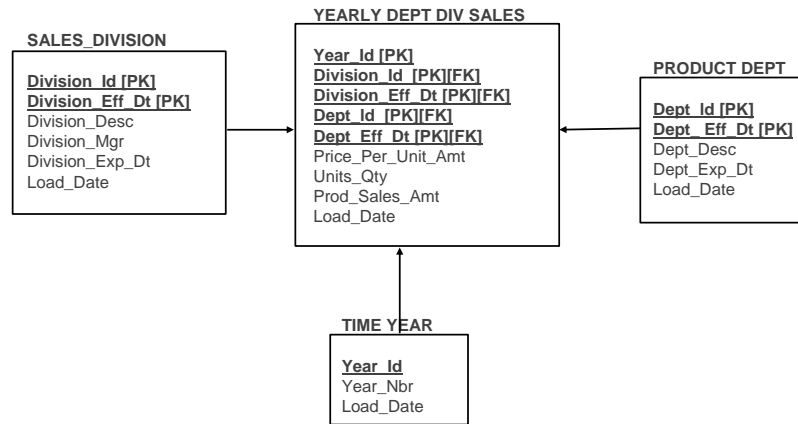
- Decide on granularity of the second summary level that you need to store
  - In this example, Sales by Class by Region by Period.





## Third Summary Level Grain

- Decide on granularity of the third summary level
  - In this example, Sales by Department by Division by Year.



## Reasoning Behind Aggregates

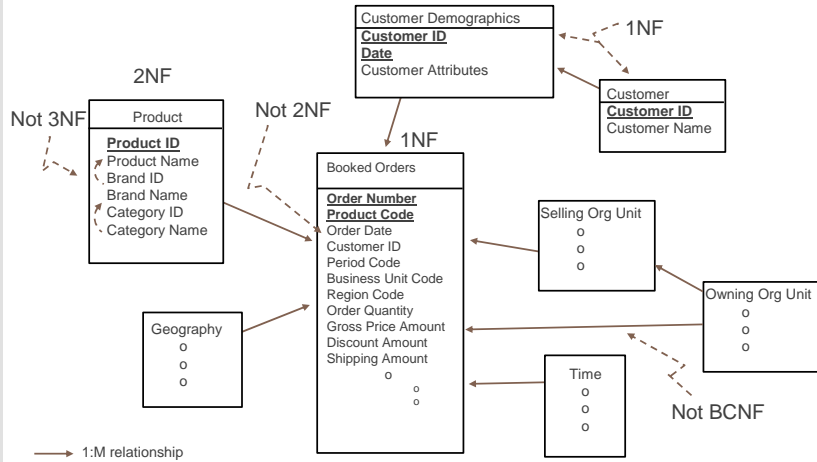
- Two reasons for aggregates:
  - To improve performance
  - To provide a consistent basis for answers
- Aggregates are a two-edge sword
  - They improve query performance
  - They prolong batch duration
- Create aggregates where:
  - Data is abundant
  - For frequently running queries
  - The compression ratio (from original to aggregate) is at least 10: 1





## Denormalization and the Dimensional Model

- A typical dimensional model uses denormalization in the following way:
  - ✦ Violates 3NF in dimensions by collapsing dimensions
  - ✦ Violate 2NF in facts by collapsing common data into the transaction
  - ✦ But, preserves 1NF in changing dimensions



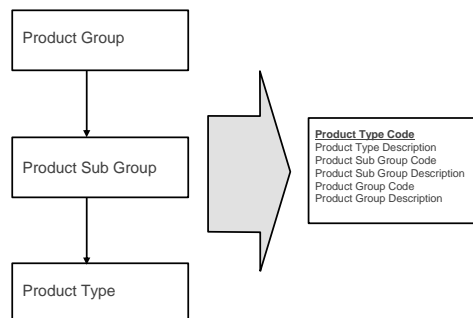
(c) InfoModel LLC, 2005

31



## Denormalization of Dimension Hierarchies

- Used carefully in OLTP systems because of the UPDATE nature of the data
- Highly normalized model most suitable for OLTP applications
- Advantage:
  - ✦ Fewer joins and better performance
- Disadvantage:
  - ✦ Redundancy
  - ✦ The cost of additional storage
  - ✦ Integrity problems and extra work when applying changes to duplicated data
  - ✦ Reduces intelligence about higher levels in the hierarchy



(c) InfoModel LLC, 2005

32



## Summary of Ways to Keep Historical Data

### **Current Occurrence**

- Contains the present state of the data only
- No history

### **Continuous History**

- Adds a date to any occurrence
- Continuous history
- No gaps or overlaps

### **Delimited History**

- Requires an effective and expiration date to any occurrence
- Can support gaps and overlaps
- Can be discontinuous history

### **Complex History (not covered in this presentation)**

- Permits forward and backward history changes and out of sequence changes
- Can be discontinuous history

### **Incomplete History**

- Keeping in a single row only a limited amount of history such as this time, last time



## 1. Current Occurrence

### **Current Occurrence**

- Contains the present state of the data
- A date is not absolutely necessary because a change event causes a new occurrence to be added and the old occurrence to be removed.

<u>Customer No</u>	<u>Customer Name</u>	<u>Address</u>
12345	Lee	160 W87 St., New York, NY
12345	Lee	161 W86 St., New York, NY

*old*  
*new*

- Not more than one occurrence may exist at any one time
- Date can be used, but is not part of key
- Changes either update the occurrence or delete it and add a new one





## 2. Continuous History

- Provides a continuous history (no gaps or overlaps)
- Adds a date to the (natural) key
- The simple date is used to indicate the start or effective date
- Identifies when the activity that initiated the occurrence became effective
- Allows us to retrieve a specific occurrence by supplying the primary entity key and a date \*

\* the question of using a surrogate key will be addressed later

<u>Customer No</u>	<u>Effective Date</u>	<u>Customer Name</u>	<u>Customer Address</u>
12345	07/27/92	Lee	160 W87 St., New York, NY
12345	06/25/93	Lee	161 W86 St., New York, NY



## Continuous History

- Since there are no gaps or overlaps, continuous history **requires** only an effective date
  - The expiration date for a previous row can be derived from the effective date of a new row
- Physically and for optimization, an expiration date is usually necessary to simplify query processing
  - Your SQL will still have to visit the rows twice, once for each date
- With both dates you can readily answer questions like:
  - "Search for all the open account transactions with a *begin* data occurring on or prior to the end of the timeframe, and with an *end* data occurring on or after the beginning of the time span
  - "Search for the single transaction with a *begin* date on or before the end of the time span and with an *end* date on or after the end of the time frame
  - "Search for the single transaction with a *begin* date of or before the arbitrary point in time and with an *end* date on or after the arbitrary point in time."



## Continuous History Augmented with End Date

Continuous history with both dates. Expiration Date is day before next Effective Date and Expiration Date is set to a future unreachable date

CustID	Current Ind	Eff Date	Expir Date	Org	Qty	Amount
370920	N	01/01/1996	05/12/1996	ZYX	009	\$357.00
370920	N	05/13/1996	03/15/1997	STU	009	\$357.00
370920	Y	03/16/1997	01/01/3000	STU	156	\$357.00

Last record of same Customer has its end date set to the distant future.

CustID	Current Ind	Eff Date	Expir Date	Org	Qty	Amount
370920	N	01/01/1996	05/12/1996	ZYX	009	\$357.00
370920	N	05/13/1996	03/15/1997	STU	009	\$357.00
370920	N	03/16/1997	01/01/3000	STU	156	\$357.00
370920	Y	07/08/1997	01/01/3000	STU	156	\$557.00

New record added, initially with same expiration date

CustID	Current Ind	Eff Date	Expir Date	Org	Qty	Amount
370920	N	01/01/1996	05/12/1996	ZYX	009	\$357.00
370920	N	05/13/1996	03/15/1997	STU	009	\$357.00
370920	N	03/16/1997	07/07/1997	STU	156	\$357.00
370920	Y	07/08/1997	01/01/3000	STU	156	\$557.00

End date of previous record updated to be the day before the start date of the new current record

(c) InfoModel LLC, 2005

37

## 3. Delimited History

- Is inherently discontinuous
- Logically **requires** a start and end date
- The data is meaningless without both dates
- Can potentially have gaps and overlaps (depending on business rules)
  - ✦ Effective and expiration dates for a marketing campaign
  - ✦ An employee leaving a company and the returning years later
- If overlaps are **not** allowed, either:
  - ✦ An interval structure may be necessary to prevent overlaps (with valid from and to dates)
  - ✦ ETL could be used to guarantee conformance to intervals

(c) InfoModel LLC, 2005

38



## Delimited History

- Requires an effective and expiration date to any occurrence
- Gaps and overlaps are (conceptually) possible
- Useful when you need to know which records were in force during a particular time period
- In the following example, the natural key is Customer No + Effective Date.

Customer No	Posting Date	Effective Date	Expiration Date	Cust Name	Customer Address
12345	07/27/92	01/22/93	12/29/93	Lee	160 W87 St., New York, NY
67890	06/25/93	06/25/93	12/23/93	Louis	161 W86 St., New York, NY



## 4. Incomplete History

- Usually includes a **current and previous value only**
  - Actually a matter of creating an array of values (in this case an array of two)
  - Must add the additional attributes to support the changes
  - May require a date to timestamp each change
  - If more history is needed, it can be handled as an array
- ✦ These arrays make sense only if the number of attributes is few and the number of repetitions is stable

### Original

Cust ID	Date	Tax Number	Name	Current Income
12345	1/1/95	101-34-0987	John Milton	\$100-125K

### Changed

Cust ID	Date	Tax Number	Name	Current Year Income	Last Year Income	Date
12345	1/1/98	101-34-0987	John Milton	\$250-275K	\$100-125K	1/1/95





## Types of Changing Dimensions

- Ralph Kimball has proposed a classification of changing dimensions, Often called (somewhat unnecessarily) "**slowly**" **changing dimensions**
- He has identified 3 types, and they are perfectly valid:
  - 1. Overwrite the history (Current Occurrence)
  - 2. Keep complete history of dimensions (Continuous History)
    - ✦ Add a new dimension record with either date as part of the key
    - ✦ Or a generalized key (such as a surrogate key)
  - 3. Keep current and last value (Incomplete History)
- Many DW implementation have identified a fourth:
  - 4. Keep *specific periodic snapshots* of a dimension



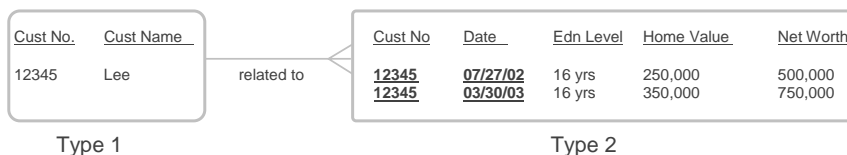
## Full vs. Partial History Records

- **Full History** - in this case, each time any change occurs, we keep a record of the entire new and old row

Customer No.	Date	Customer Name	Customer Address
<u>12345</u>	<u>07/27/92</u>	Lee	160 W87 St., New York, NY
<u>12345</u>	<u>06/25/93</u>	Lee	161 W86 St., New York, NY

Flattened

- **Partial History** - in this case, changes happen to a designated subset of attributes. Only the changing attributes are separated out into a new entity.





## "Slowly" Changing Dimensions

- Caveat
  - ✦ Not all dimensions are homogeneous, meaning not all the data in a dimension will change, can change, or will change at the same time
  - ✦ Most dimensions qualify for partial change such as where:
    - Some parts may not change (or if they do, you don't care)
    - Other parts may change and
    - Other parts still may change at different velocities
- What is conceptually one dimension, such as customer, may concurrently have some part that is Type 1 and other parts that are Type 2, even different Type 2s.
- There are times when you cannot completely flatten these into a single row
- There are times when it is best to keep them separate
  - ✦ Such as when there are other relationships pertinent only to the unchanging part



## Applying Changing Dimensions

- Adds a new row every time there is change.
- Requires specialized key, such as:
  - ✦ Surrogate Key
  - ✦ Customer ID + Version Number
  - ✦ Customer ID + Date
- The transaction always points to the instance of the dimension that was true when the transaction happened

HistSKey1				
HistSKey2				

Sales Fact Table

HistSKey1	1/1/95	101-34-0987	John Milton	\$100-125K



HistSKey1	1/1/95	101-34-0987	John Milton	\$100-125K
HistSKey2	1/1/98	101-34-0987	John Milton	\$250-275K

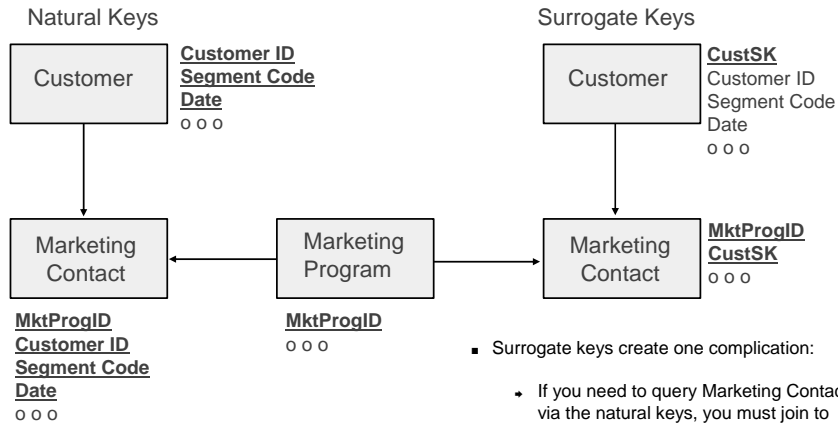
Customer Dimension Table





## Example of Surrogate Keys

- Surrogate key adds one attribute at the one-end
- It saves space at the many-end



## Surrogate Keys vs. Natural Keys

### ■ Pros of Surrogate Keys

- ✦ Reduced space by replacing a large combined primary key, especially when used as a foreign key (large natural keys)
- ✦ Isolation of the data from operational changes (such as reused order numbers)
- ✦ Ability to have multiple instances against a given entity (a changing dimension)
- ✦ Speed of access when the optimizer uses a simple, numeric index (the natural key could have a large composite index)

### ■ Cons of Surrogate Keys

- ✦ (Possible) Extra storage requirement due to an extra column (if both natural and surrogate are retained)
- ✦ Integrity having to be enforced some other way
- ✦ Still need access to the data using natural keys
- ✦ May have to supplement surrogate keys with a mapping table correlating surrogate key and its equivalent natural key.





## History Example

TransID	TransDate	Amount	SALES				Surr Key
			Count	OU No	OU Date	or	
X01	6/1/03	\$100	110	2	1/1/03		SK2
X02	6/1/04	\$550	60	2	1/1/04		SK5

OUSK	OU No	Start Date	End Date	ORG UNIT			or	Surr Key
				Desc	Parent OU	Parent Date		
SK1	1	1/1/03	12/31/03	OU1	--	--		
SK2	2	1/1/03	12/31/03	OU2	1	1/1/03		SK1
SK3	3	1/1/03	12/31/03	OU3	1	1/1/03		SK1
SK4	4	1/1/04	12/31/04	OU4	--	--		
SK5	2	1/1/04	12/31/04	OU2	4	1/1/04		SK4
SK6	3	1/1/04	12/31/04	OU3	4	1/1/04		SK4



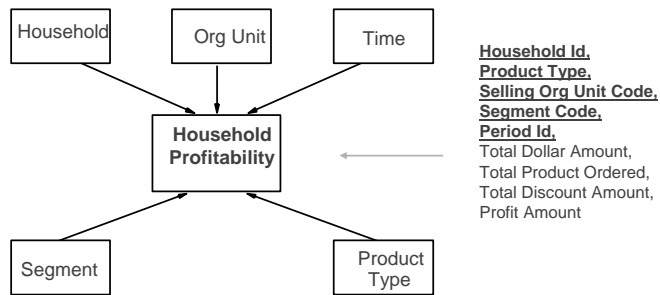
(c) InfoModel LLC, 2005

47



## A Data Mart

- A specialized set of related data designed to support the needs of a given set of knowledge workers, analysts or planners over a limited time frame



- Should be derived from the data warehouse, thus utilizing the central repository of strategic data
- May or may not be on separate physical platform from the data warehouse
- Will complicate the data architecture, especially if independent data marts



(c) InfoModel LLC, 2005

48

## Operational Data Store Versus The Data Warehouse

- The data warehouse cannot be used to help an organization struggling with unintegrated operational systems
- Enter the operational data store - it provides the foundation to achieve tangible integrated operational results in a short time frame
- It is important to note that there is a clear fire wall between the operational data store and the data warehouse.

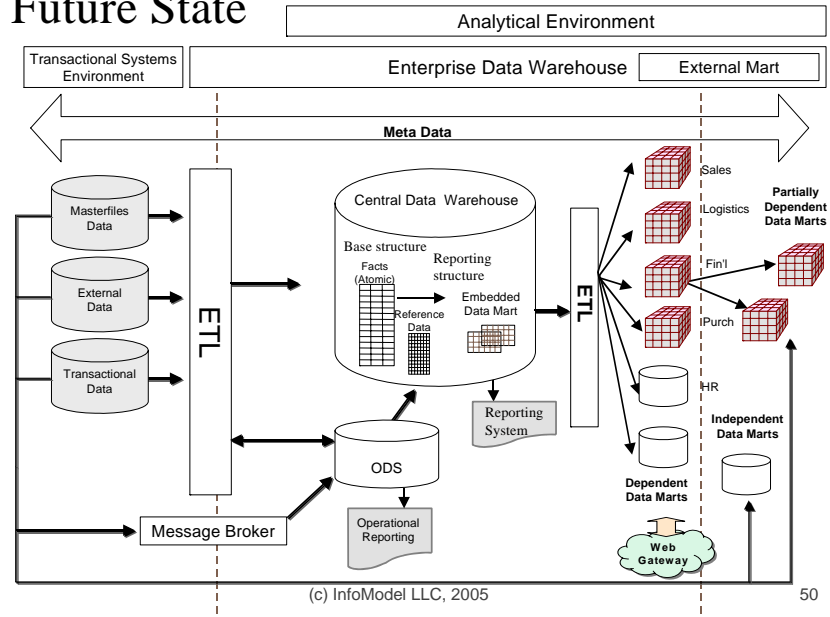


(c) InfoModel LLC, 2005

49

## Structure of a Data Warehouse

### Future State



(c) InfoModel LLC, 2005

50

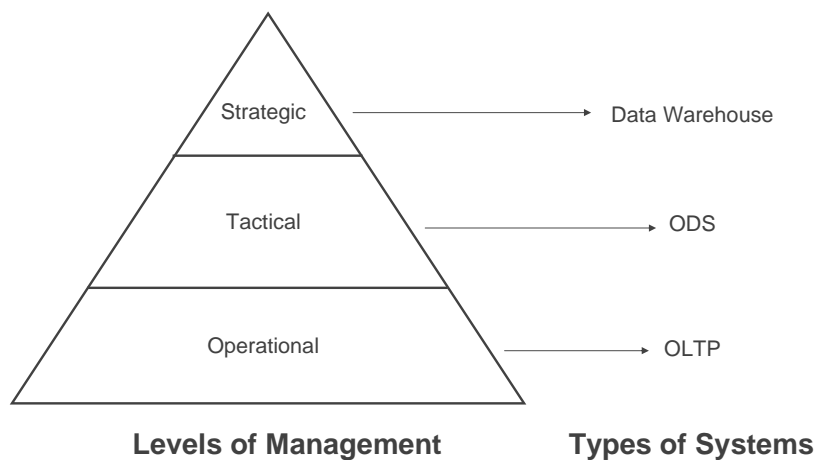


## Operational Data Store (ODS)

- A tactical environment which stores detailed, near-real time results of committed transactions for a certain period of time for immediate reporting needs and which can sometimes be updated by users
- An ODS is often created for one of three purposes
  - Integrating data from multiple sources (because modifying the source systems would be too costly)
  - Tactical reporting
  - Providing consolidated update processing
- While the ODS can be used to stage data for the DW, that is not its primary purpose



## The Anthony Triangle





## ODS Comparison

	Closely Related		Closely Related	
	OLTP	ODS	DW	DM
<b>Decision Horizon</b>	Day to Day	Day to Week	Week to Year	Week to Years
<b>Users/Usage</b>	Clerical Repetitive update, load, view Transactional	Functional management, analysis Tactical, functional, operational OLTP history	Management, analysis Strategic, directional Cross-functional	Management, analysis Strategic, directional Cross-functl
<b>Updates</b>	Insert/delete/update Real-time updates	Reflect changes in operational systems Periodic update (0.5-24 hr)	Non-volatile Periodic updates	Periodic Refresh
<b>Level of detail</b>	Detailed Aggregates rarely stored	Detailed Individual aggregates as req'd	Detailed & summaries	Summary only
<b>Data</b>	Functionally oriented May be unintegrated Structured for operational computations Highly normalized	Functionally oriented Minimal Integration Structure based on requirements (load, volume, usage)	Subject to Enterprise oriented Integrated Relational to dimensional	Subject oriented Multi-dimensional
<b>Data Retention</b>	Days to Weeks	6 Months to 1 Year	2-x Years**	2-x Years**



\*\* Data Retention requirements are based on business needs

(c) InfoModel LLC, 2005

53



## Important Terms

- Data warehouse
- Data mart
- Central data warehouse
- Gather-store-deliver
- OLAP
- OLTP
- ER, ERD
- Dimensional model
- Fact
- Dimension
- Conformed dimension
- Logical data model
- Physical data model
- Star schema
- Snowflake schema
- Aggregation
- Ad hoc
- Rollup
- Types of time support
- Types of time dimensions
- ODS



(c) InfoModel LLC, 2005

54