数据仓库与数据挖掘
Data Warehouse & Data Mining

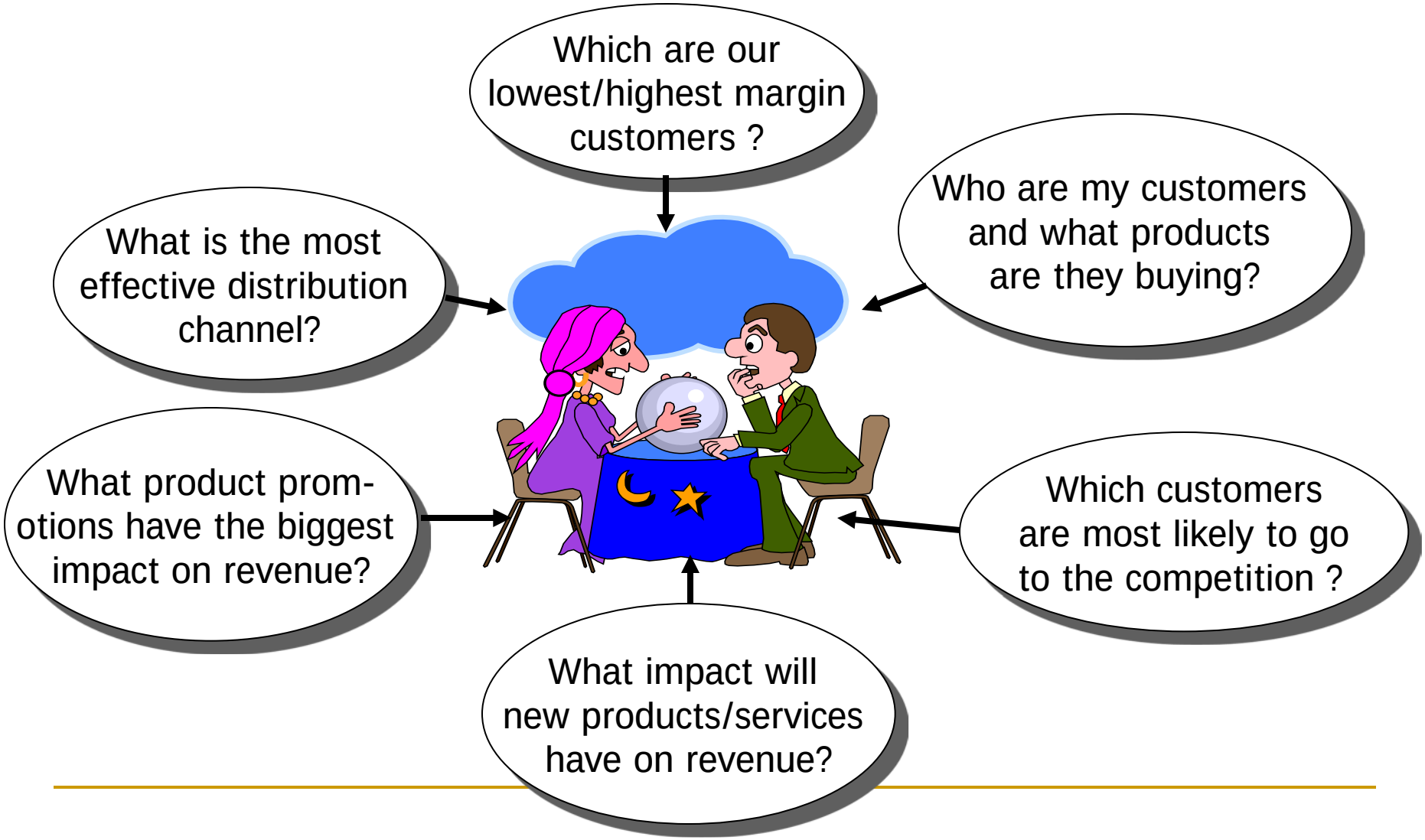
宋 杰
Song Jie
东北大学 软件学院
Software College, Northeastern University

Personal Introduction

- Born in 1980;
- Ph.D in Computer Science (3 years);
- Post-doctoral work in France (2 years);
- Associate professor of Software College, NEU;
- Papers, more than 40;
- Projects, more than 15;
- **Current research area: Energy Efficient Computing, Data Insensitive Computing, Cloud Computing**

1. Data Warehouse: Basic Concepts

A producer wants to know....



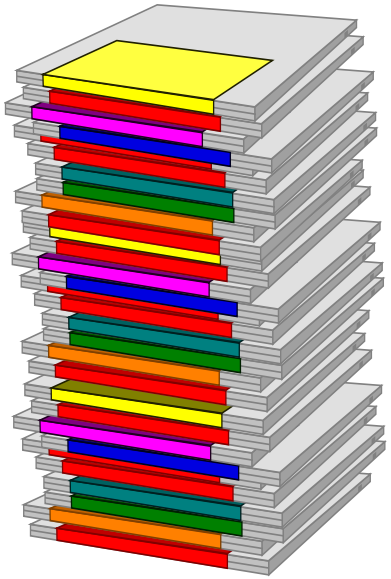
Data, Data everywhere yet ...

- I can't find the data I need
  - data is scattered over the network
  - many versions, subtle differences
- I can't get the data I need
  - need an expert to get the data
- I can't understand the data I found
  - available data poorly documented
- I can't use the data I found
  - results are unexpected
  - data needs to be transformed from one form to another

What are the users saying...

- Data should be integrated across the enterprise;
- Summary data has a real value to the organization;
- Historical data holds the key to understanding data over time;
- What-if capabilities are required.

What is data warehouse?



- Data warehouse is a
  - subject-oriented,
  - integrated,
  - time-variant (时变), and
  - nonvolatile (非易失)

nonvolatile
[nɔn'vɔlətail]
adj.
(尤指物质)非挥发性的; 不挥发的

collection of data in support of management and decision making process

- by W. H. Inmon 's definition (1992)

- 数据仓库是支持管理和决策过程的、面向主题的、集成的、时变的、稳定的数据集合

Subject-Oriented

- Organized around major subjects, such as customer, product, sales;
- Focusing on the modeling and analysis of data for decision makers, not on daily operations or transaction processing;
- Provide a simple view around particular subject issues by excluding data that are not useful in the decision support process;

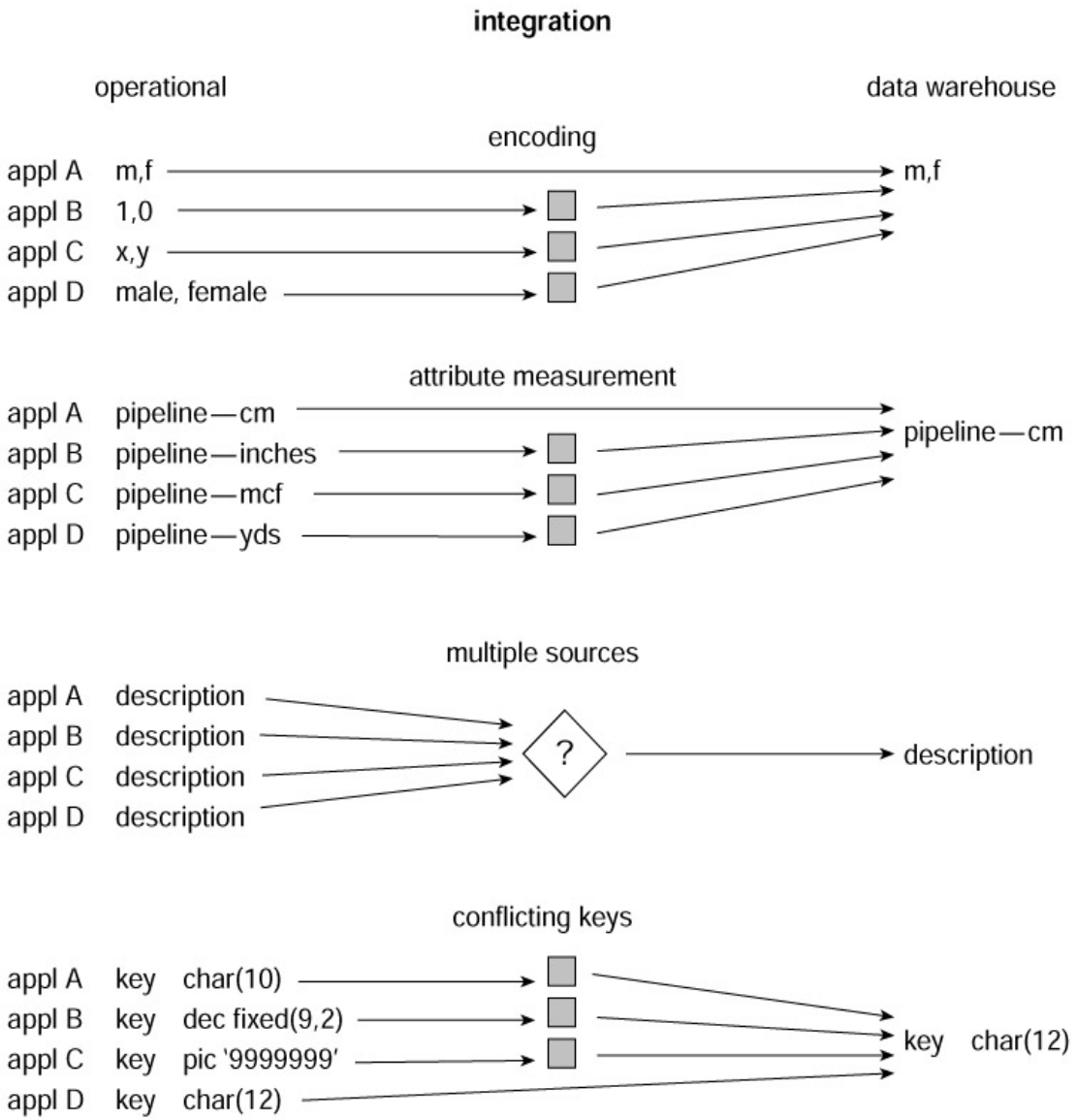
Application-Oriented vs. Subject-Orientated

- For a university
  - Applications: entrance examination, teaching courses, supervise classes, arrange classrooms, employment;
  - Subjects: students (employment, average score, innovation, mater abroad), teacher, research;
- For a retailer
  - Applications: purchase, payment, account;
  - Subjects : product, sale, vendor.

Integrated

- Constructed by integrating multiple, heterogeneous data sources
  - Relational databases, flat files, on-line transaction records.
- Data cleaning and data integration techniques are applied
  - Ensure consistency in naming conventions, encoding, structures, attribute measures, etc. among different data sources;
  - When data is moved to the warehouse, it is converted.

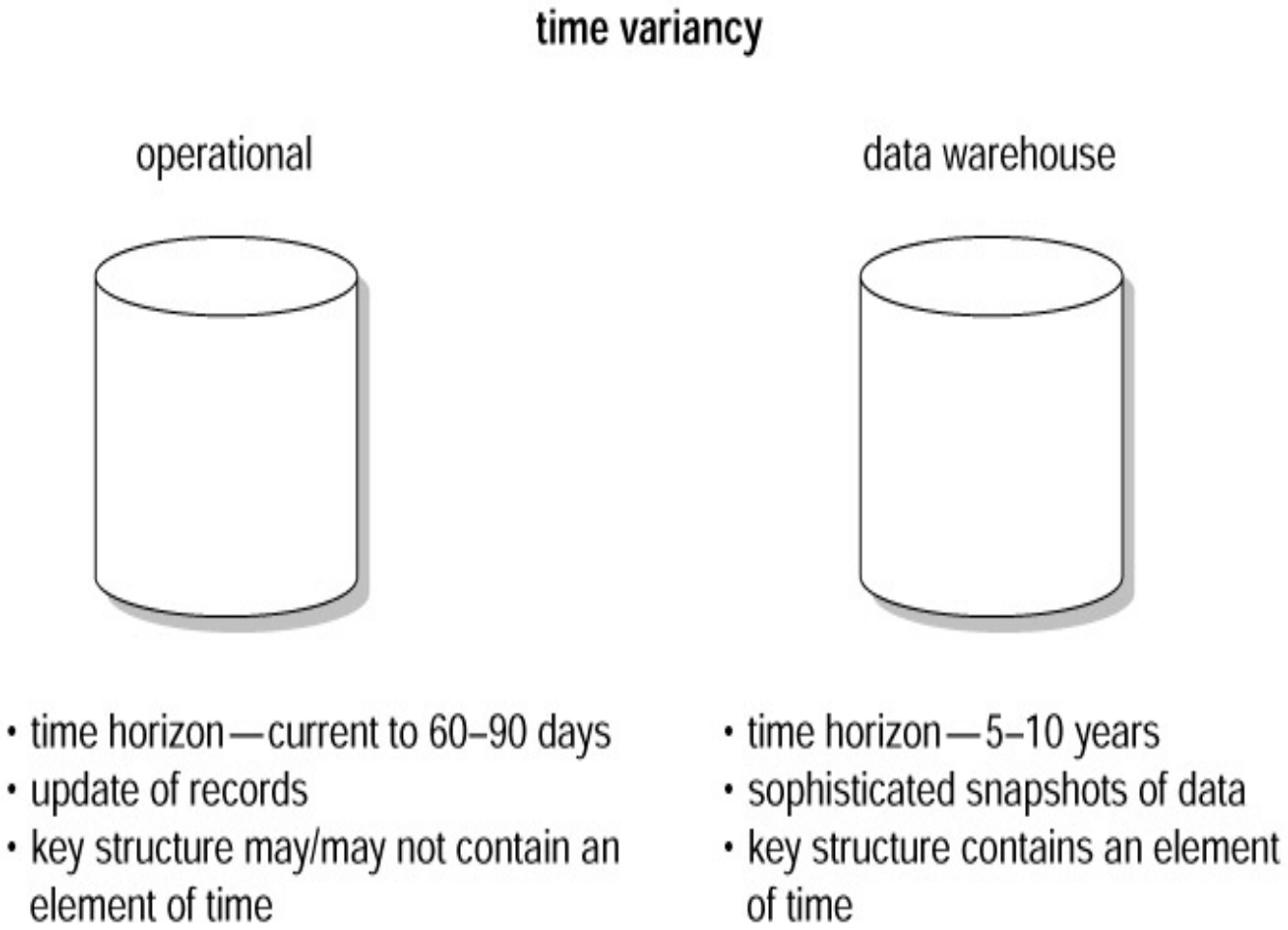
Integrated


integration
operational
data warehouse
encoding
appl A m,f
appl B 1,0
appl C x,y
appl D male, female
m,f
attribute measurement
appl A pipeline—cm
appl B pipeline—inches
appl C pipeline—mcf
appl D pipeline—yds
pipeline—cm
multiple sources
appl A description
appl B description
appl C description
appl D description
?
description
conflicting keys
appl A key char(10)
appl B key dec fixed(9,2)
appl C key pic '9999999'
appl D key char(12)
key char(12)

Time Variant

- The time horizon for the data warehouse is significantly longer than that of operational systems
  - Operational database: current value data;
  - Data warehouse: provide information from a historical perspective (e.g., past 5-10 years).
- Every key structure in the data warehouse
  - Contains an element of time, explicitly or implicitly;
  - But the key of operational data may or may not contain "time element".

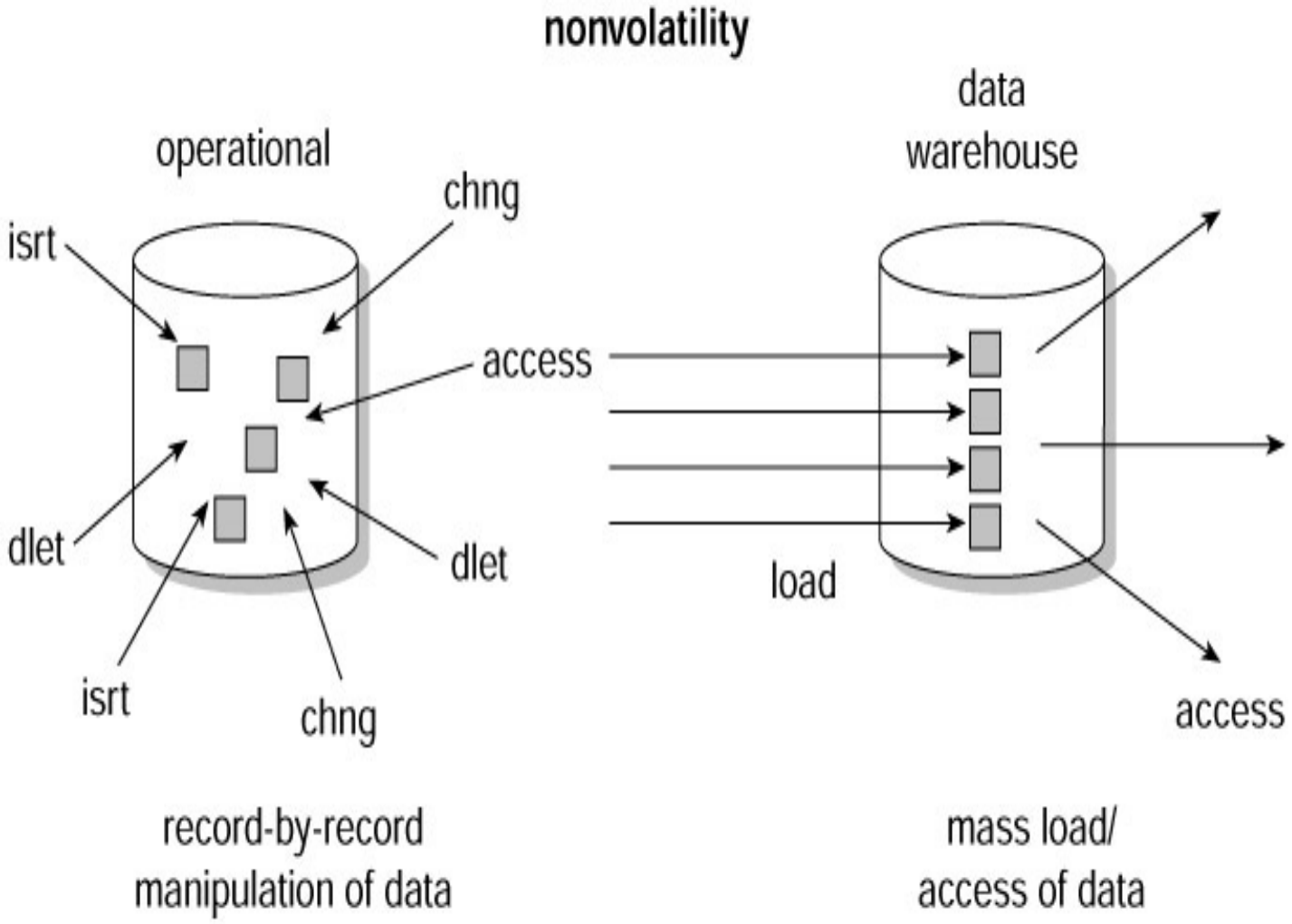
Time Variant



Nonvolatile

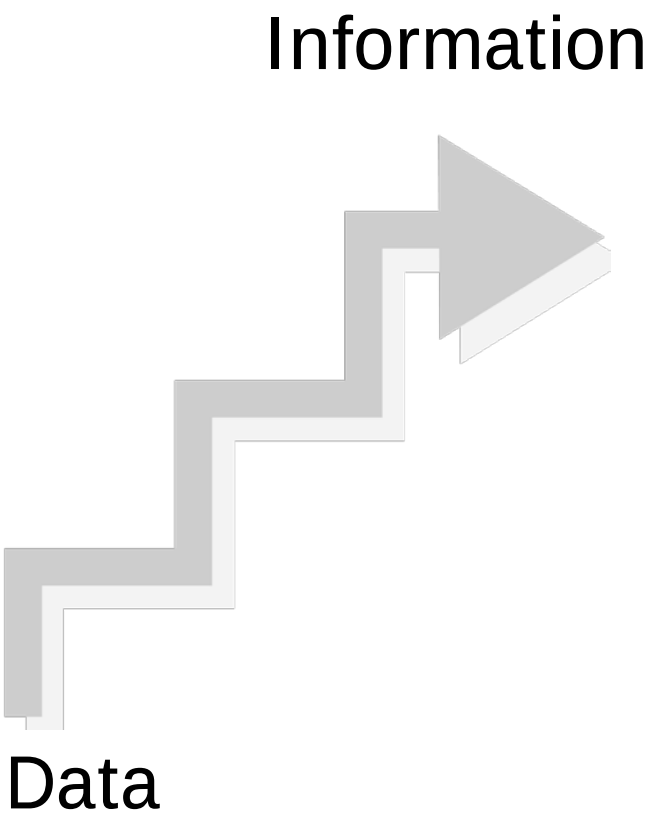
- A physically separate store of data transformed from the operational environment
- Operational update of data does not occur in the data warehouse environment
  - Does not require transaction processing, recovery, and concurrency control mechanisms
  - Requires only two operations in data accessing:
    - initial loading of data
    - access of data

Nonvolatile



What is Data Warehousing?

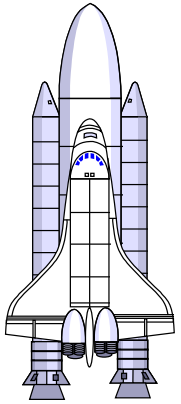
- A process of transforming data into information and making it available to users in a timely enough manner to make a difference
  - By - Forrester Research
    April 1996



Explorers, Farmers and Tourists

Tourists: Browse information harvested by farmers

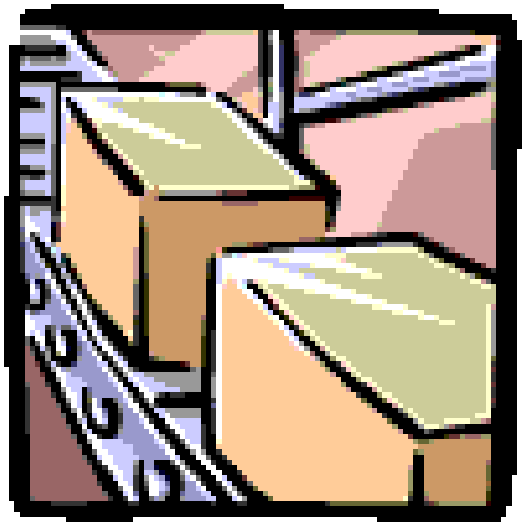
Farmers: Harvest information from known access paths



Explorers: Seek out the unknown and previously unsuspected information hiding in the detailed data

Very Large Databases

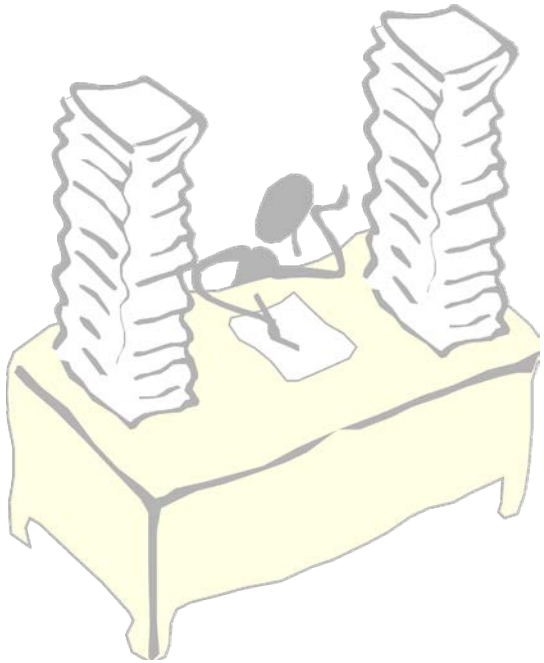
- Terabytes -- 10^12 bytes:
  - Walmart -- 24 Terabytes
- Petabytes -- 10^15 bytes:
  - Geographic Information Systems
- Exabytes -- 10^18 bytes:
  - National Medical Records
- Zettabytes -- 10^21 bytes:
  - Weather images
- Zottabytes -- 10^24
  - Intelligence Agency (情报局) Videos



Decision Support

- Used to manage and control business;
- Data is historical or point-in-time;
- Optimized for inquiry rather than update;
- Used by managers and end-users to understand the business and make the judgments;

Data Mining



- Data Warehousing provides the Enterprise with a memory

- Data Mining provides the Enterprise with intelligence

We want to know ...

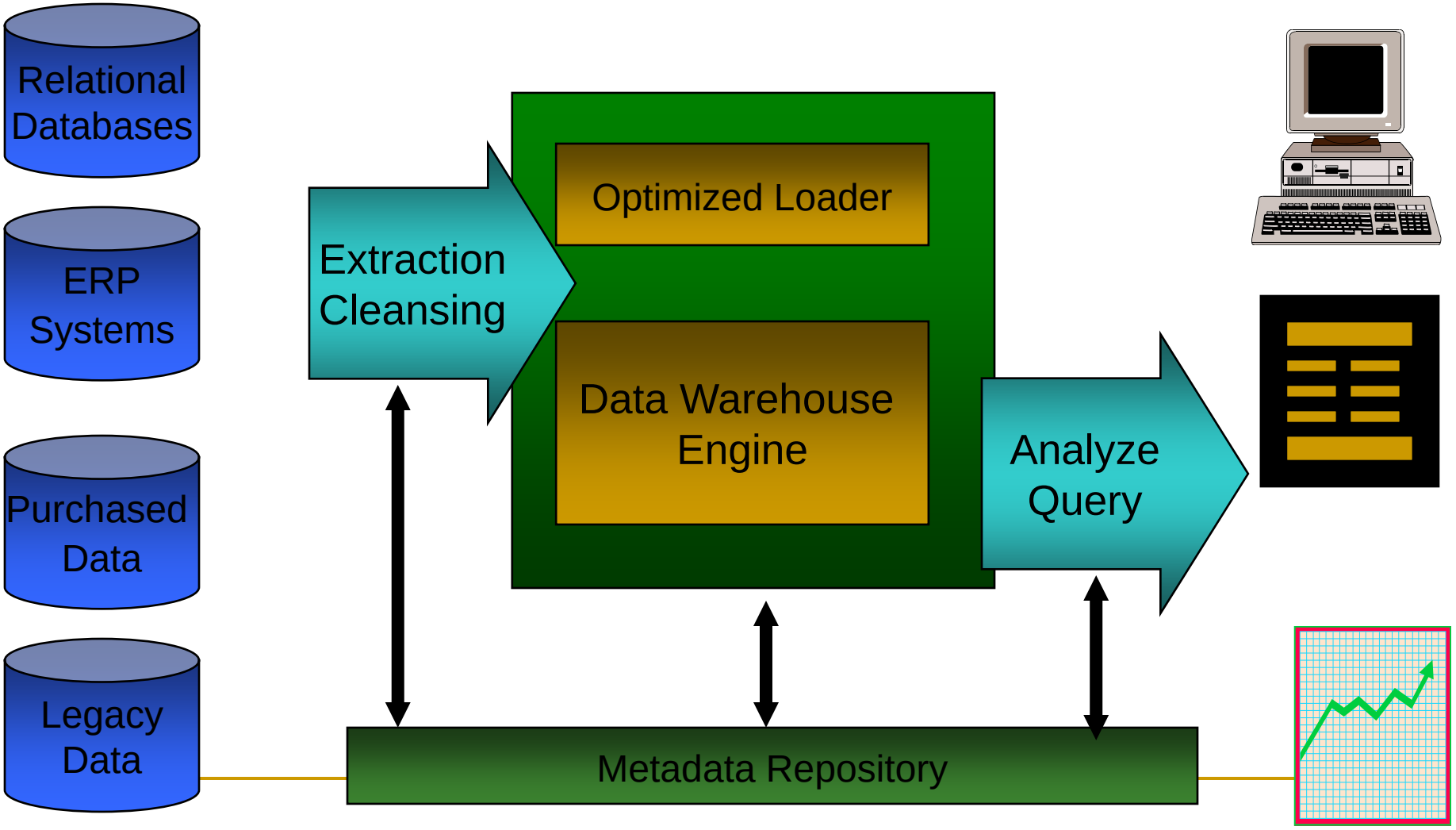
- Given a database of 100,000 names, which persons are the least likely to default on (拖欠) their credit cards?
- Which types of transactions are likely to be fraudulent (期骗性的挪用) given the demographics (人口统计数据) and transactional history of a particular customer?
- If I raise the price of my product, what is the effect on my ROI?
- If I emphasize ease-of-use of the product as opposed to its technical capabilities, what will be the net effect on my revenues?
- Which of my customers are likely to be the most loyal?

Data Mining helps extract such information

Data Mining in Use

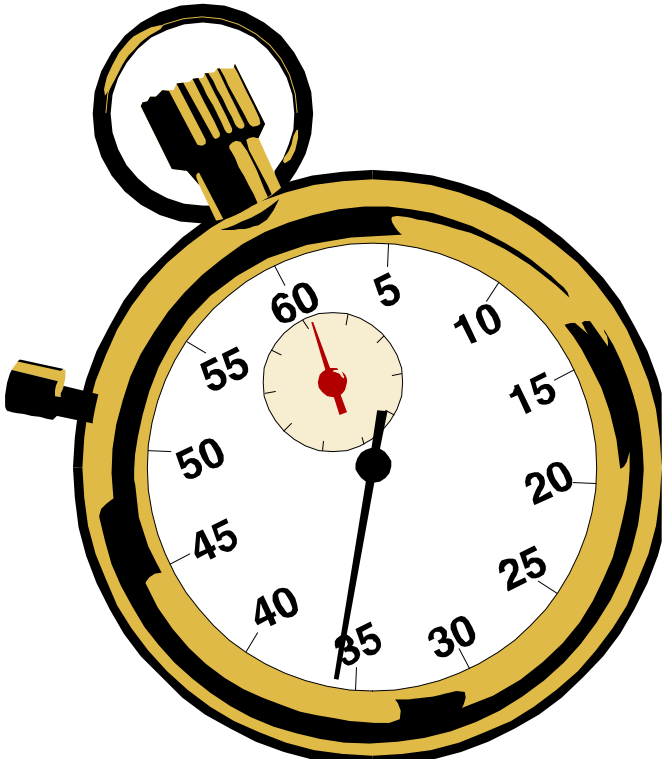
- The US Government uses Data Mining to track fraud;
- Basketball teams use it to track game strategy;
- A Supermarket becomes an information broker:
  - Cross Selling;
  - Holding on to Good Customers;
  - Weeding out Bad Customers;

Data Warehouse Architecture



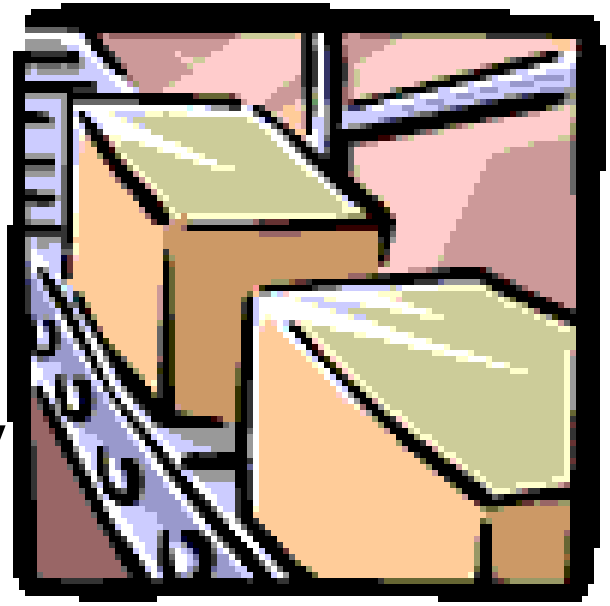
Operational Systems

- Run the business in real time
- Based on up-to-the-second data
- Optimized to handle large numbers of simple read/write transactions
- Optimized for fast response to predefined transactions
- Used by people who deal with customers, products, clerks, salesman etc.

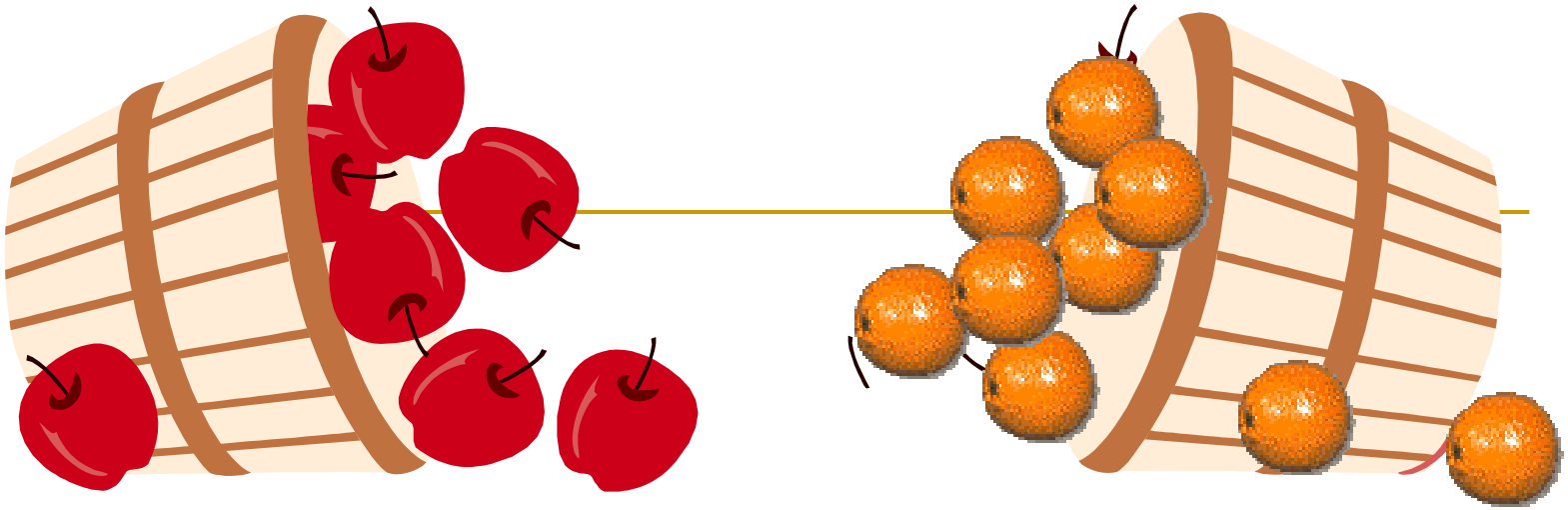


RDBMS used for OLTP (Online Transaction Processing)

- Database systems have been used traditionally for OLTP (Online Transaction Processing)
  - Daily data processing tasks
  - Detailed, up to date data
  - Structured repetitive tasks
  - Read/Update a few records
  - Transaction, isolation, recovery are critical



So, what's different?



Application-Orientation vs. Subject-Orientation

Application-Orientation

Subject-Orientation

Operational Database

Data Warehouse

Loans

Credit Card

Trust

Savings

Customer

Vendor

Product

Activity

RDBMS vs Data Warehouse

- RDBMS
  - Application Oriented
  - Used to run business
  - Detailed data
  - Current up to date
  - Isolated Data
  - Repetitive access
  - Clerical User
- Warehouse
  - Subject Oriented
  - Used to analyze business
  - Summarized and refined
  - Snapshot data
  - Integrated Data
  - Ad-hoc access
  - Knowledge User (Manager)

RDBMS vs Data Warehouse

- RDBMS
  - Performance Sensitive
  - Few Records accessed at a time (tens)
  - Read/Update Access
  - No data redundancy
  - Database Size 100MB - 100 GB
- Data Warehouse
  - Performance relaxed
  - Large volumes accessed at a time(millions)
  - Mostly Read (Batch Update)
  - Redundancy present
  - Database Size 100 GB - few terabytes

RDBMS vs Data Warehouse

- RDBMS
  - Transaction throughput is the performance metric
  - Thousands of users
  - Managed in entirety
- Data Warehouse
  - Query throughput is the performance metric
  - Hundreds of users
  - Managed by subsets

To summarize ...

- RDBMS are used to "run" a business

- The Data Warehouse helps to "optimize" the business

To summarize ...
OLTP: Online Transaction Processing
OLAP: Online Analytical Processing

	OLTP	OLAP
users	clerks, IT professionals	managers, executives
function	day to day operations	decision support
DB design	application-oriented	subject-oriented
data	current, up-to-date detailed, flat relational isolated	historical, summarized, multidimensional integrated, consolidated
access	read/write	mostly read
unit of work	short, simple transaction	complex query
number of accessed records	tens	millions
number of users	thousands	hundreds
DB size	100MB-GB	100GB-TB
metric	transaction throughput	Query throughput, response time

Let's go to the next …

数据仓库与数据挖掘
Data Warehouse & Data Mining

宋 杰
Song Jie
东北大学 软件学院
Software College, Northeastern University

1. Multidimensional Data Model (MDM)

Multidimensional data model

- Multidimensional data model is a data model that organizes data from the view of multiple dimensions.
- Dimensions are the perspectives(透视) or entities with respect to which an organization wants to keep records.

Example

We sell products in various markets, and we measure our performance over time

Business Manager

We sell ***Products*** in various ***Markets***, and we measure our performance over ***Time***

Data Warehouse Designer

Multidimensional data model

products	markets	time	…
……	……		
……			

Business Manager

Products key	Products name	...
…	…	…
…	…	…

Markets key	Markets name	…
…	…	…
…	…	…

Time key	year	month	…
…	…	…	…
…	…	…	…

Data Warehouse Designer

Products key	Markets key	Time key	sale
…	…	…	…
…	…	…	…

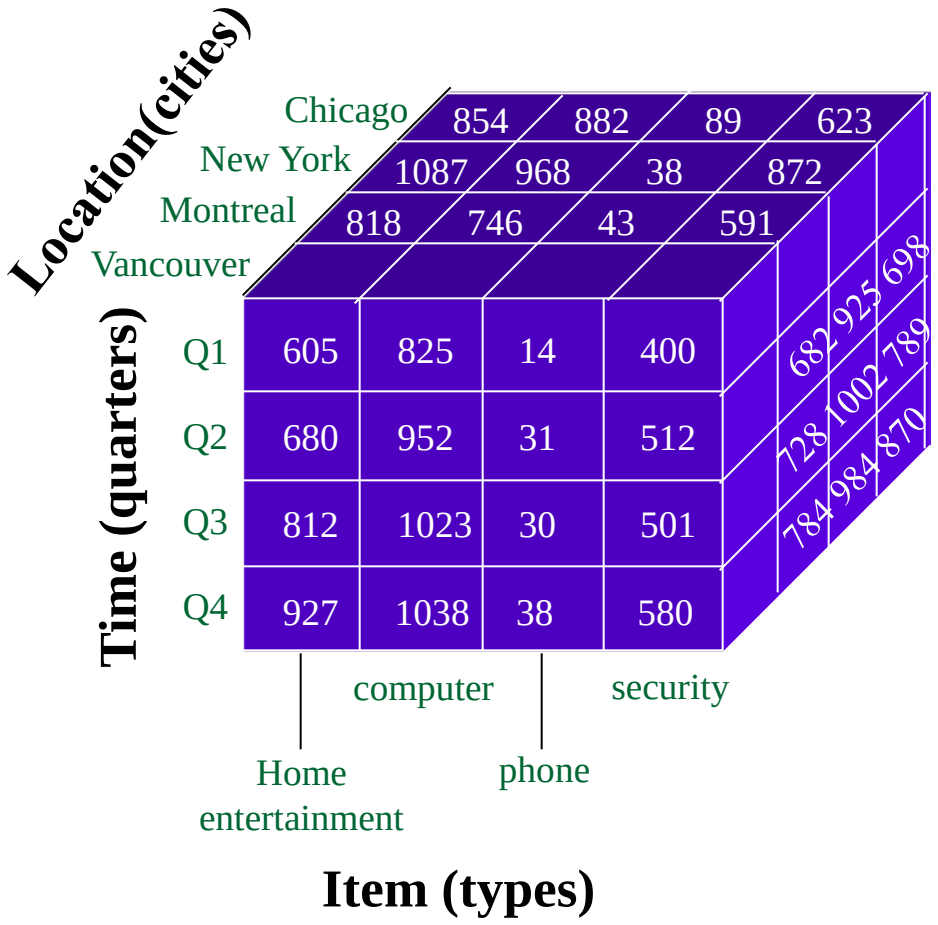
What is data cube? (I)

- let's see an example:

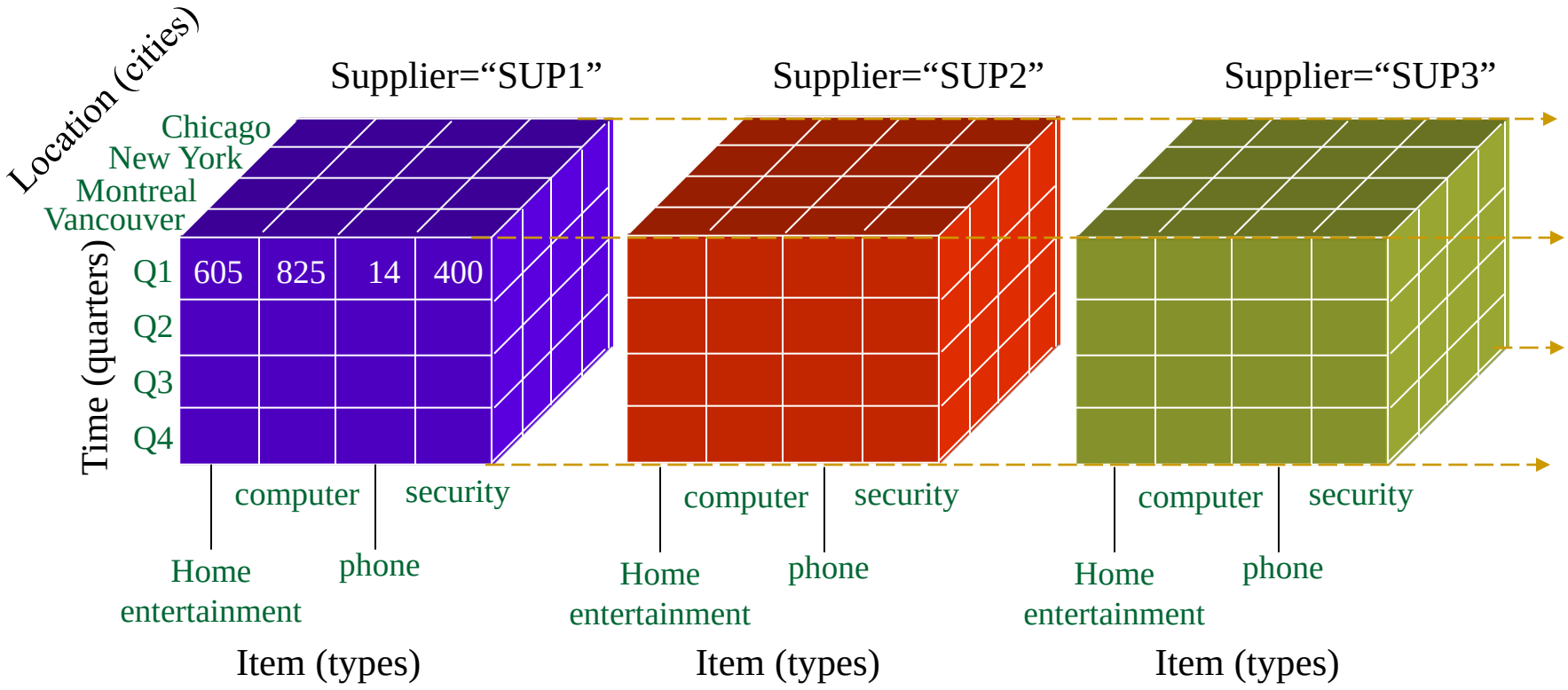
time	location = "Vancouver"				location = "Montreal"			
	item				item			
	home ent.	comp.	phone	sec.	home ent.	comp.	phone	sec.
Q1	605K	825K	14K	400K	818K	746K	43K	591K
Q2	680K	952K	31K	512K	894K	769K	52K	682K
Q3	812K	1023K	30K	501K	940K	795K	58K	728K
Q4	927K	1038K	38K	580K	978K	864K	59K	784K

time	location = "New York"				location = "Chicago"			
	item				item			
	home ent.	comp.	phone	sec.	home ent.	comp.	phone	sec.
Q1	1087K	968K	38K	872K	854K	882K	89K	623K
Q2	1130K	1024K	41K	925K	943K	890K	64K	698K
Q3	1034K	1048K	45K	1002K	1032K	924K	59K	789K
Q4	1142K	1091K	54K	984K	1129K	992K	63K	870K

What is data cube? (II)



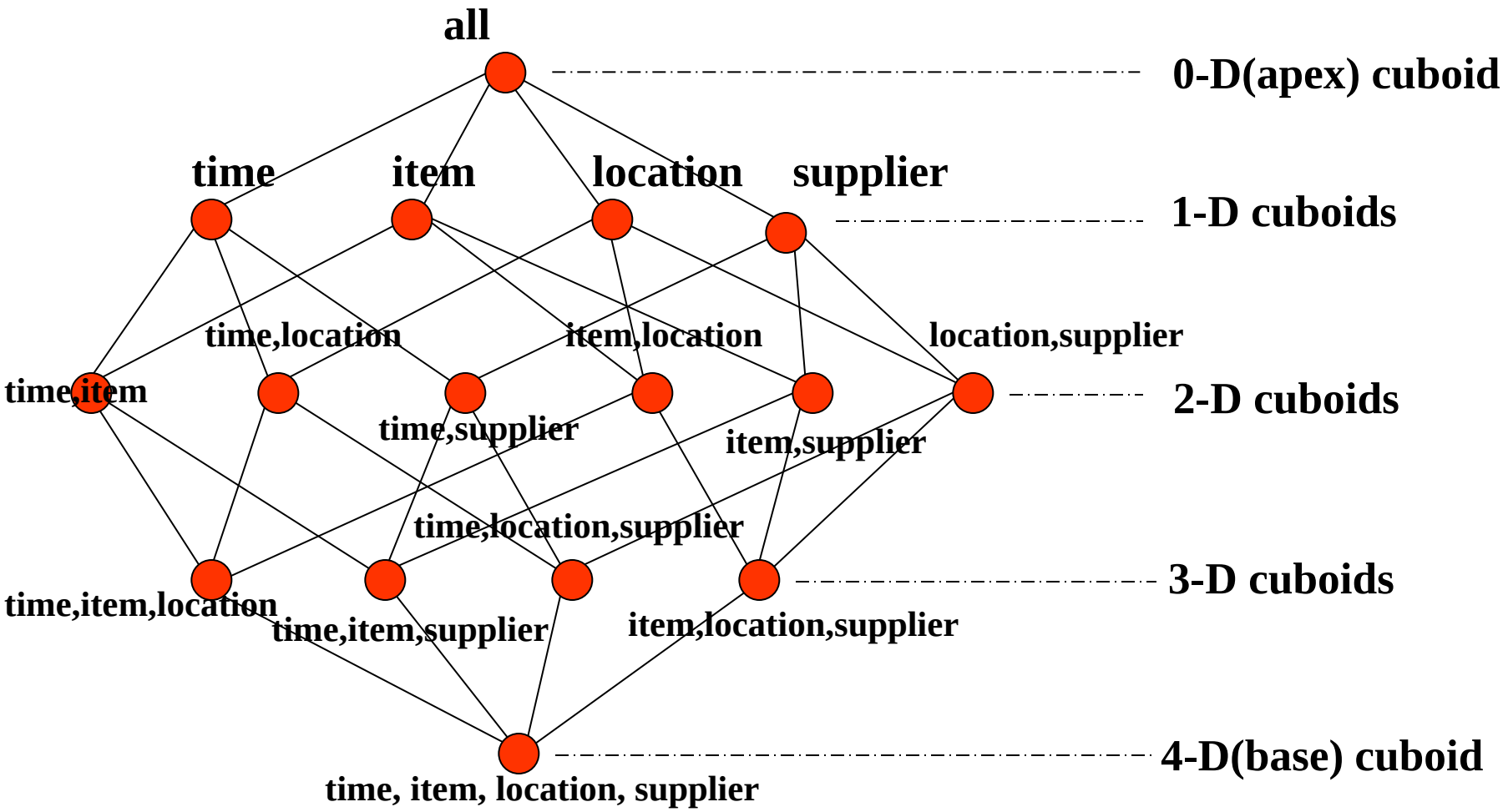
What is data cube? (IV)



***n*-D data can be represented as a series of (*n*-1)-D "cubes"**

What is data cube? (V)

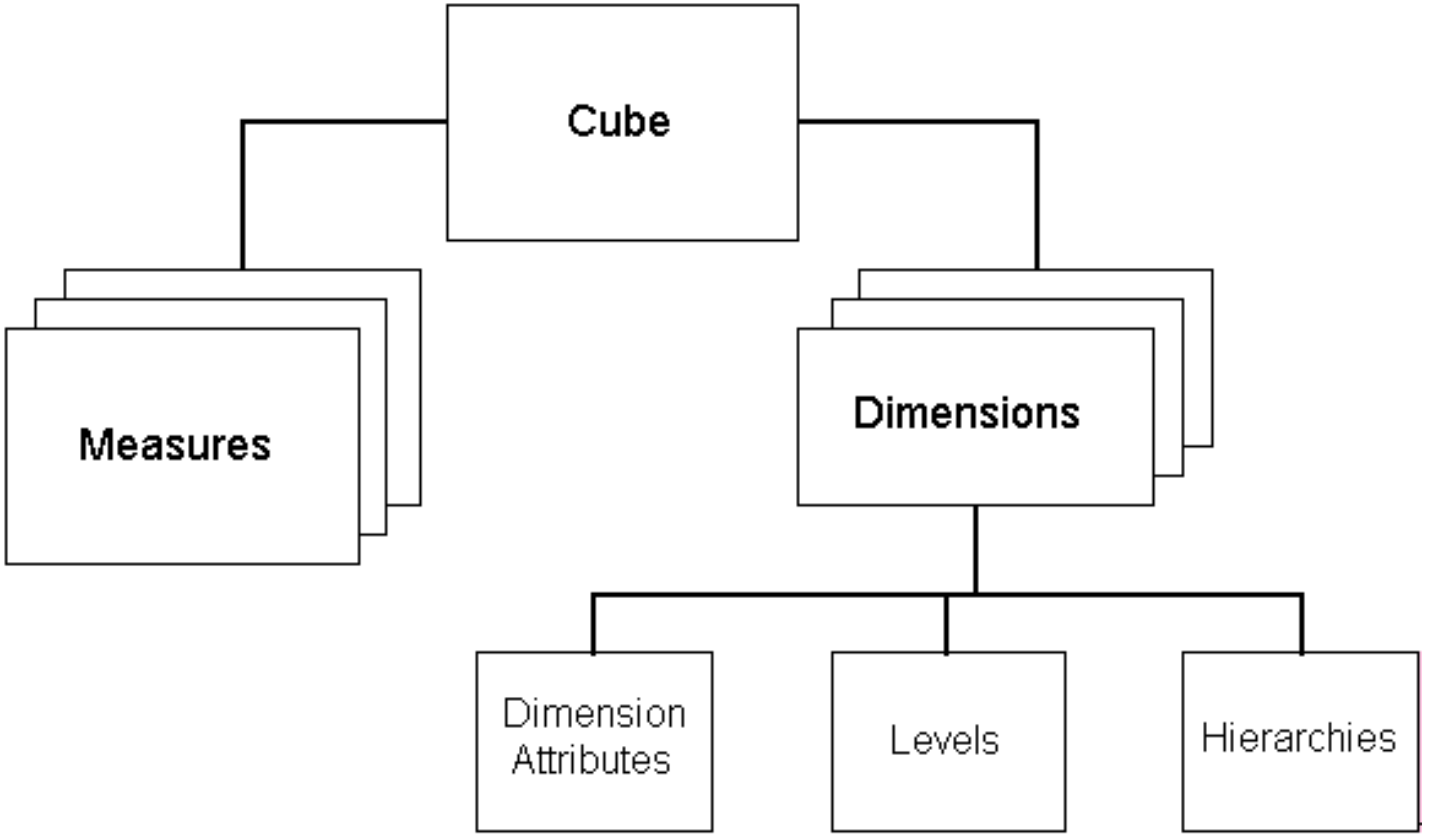
Given a set of dimensions, we can construct a lattice of cuboids



Warehouse Models & Operators

- Data Models
  - Data cubes
  - Relations
  - Stars & snowflakes
- Operators
  - slice & dice
  - roll-up, drill down
  - pivoting
  - other

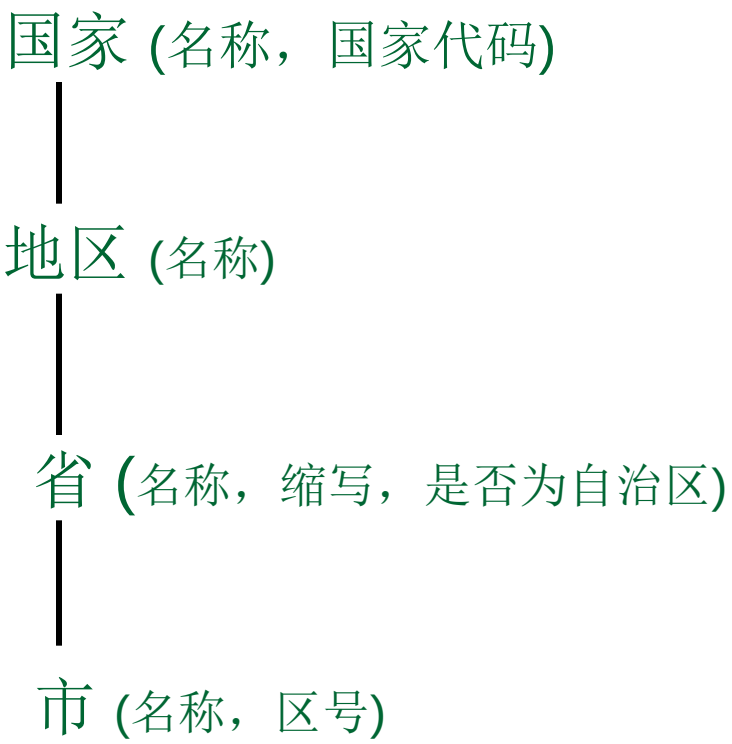
Terms


Cube
Measures
Dimensions
Dimension Attributes
Levels
Hierarchies

Dimension

- Dimension(维)：
  - Dimensional Hierarchy (维层次)
  - Dimensional Level (维级别)
  - Dimensional Attribute (维属性)
  - Dimensional Attribute Values (维属性值，维值)
  - 维是一个抽象的概念，是最高层的；维层次是在一个维上的分析路径；一个维可以有一个(或多个)维层次，每个维层次上有若干个维级别；比如年、月、日，而每个维级别上会有若干个维属性；比如一个维级别的描述信息等；每个维属性有若干维属性值；

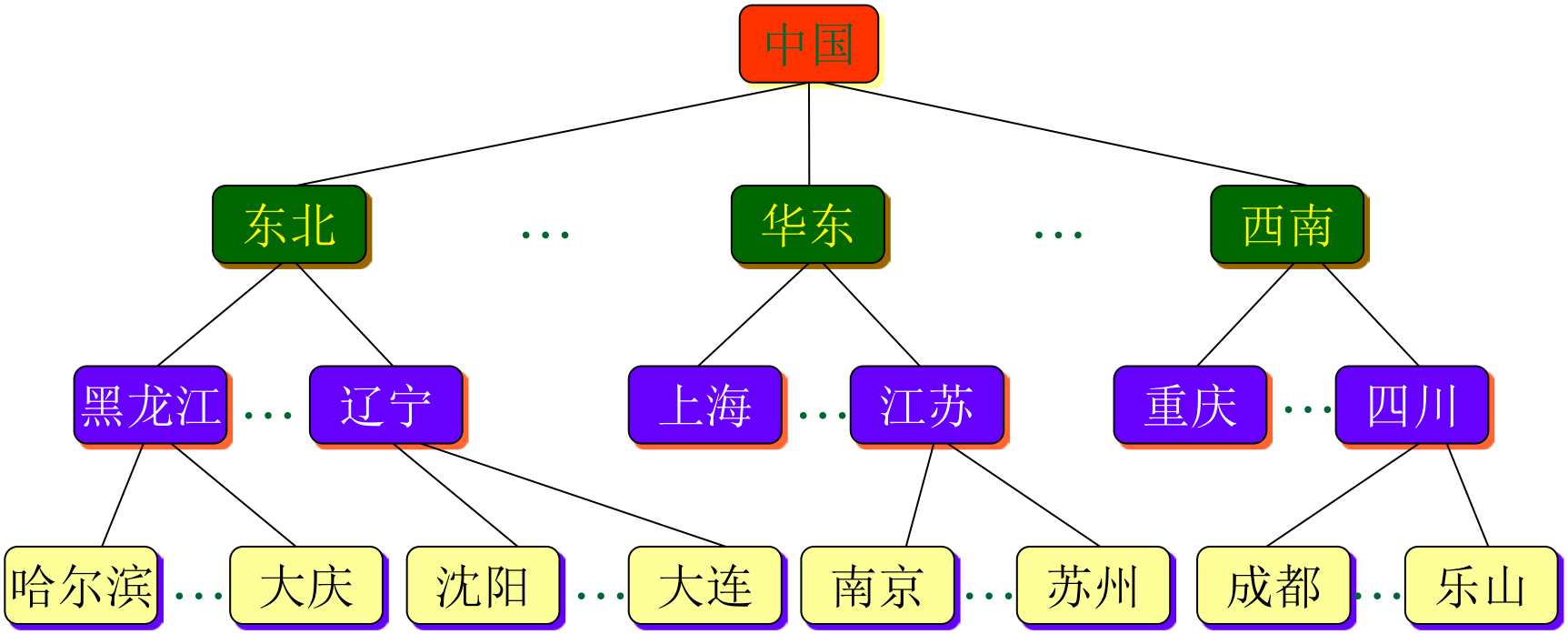
Location Dimension



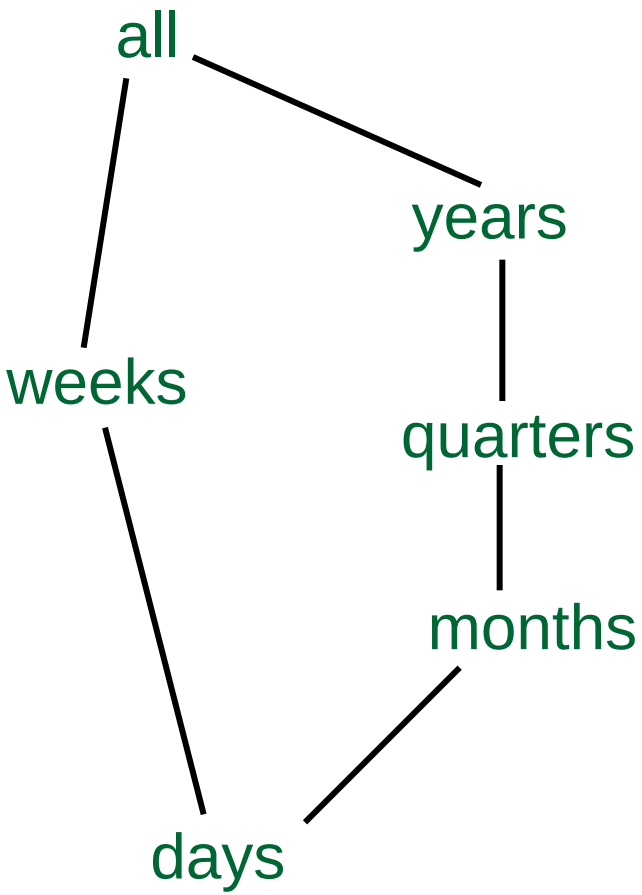
Dimensional Hierarchy
(维层次)
Dimensional Level
(维级别)
Dimensional Attribute
(维属性)
Dimensional Attribute Values
(维属性值，维值)

Location Dimension

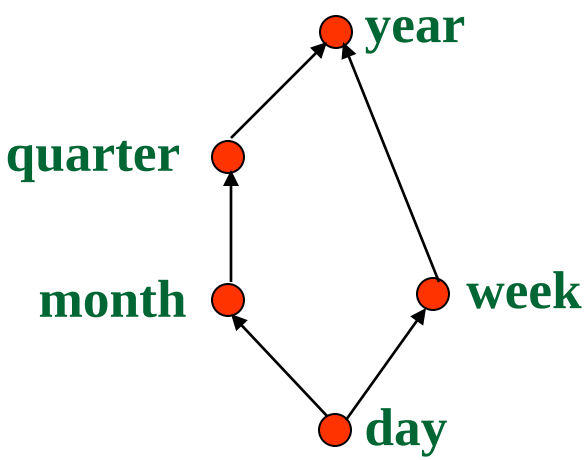
- A dimension hierarchy defines a sequence of mappings from a set of low dimension level to higher dimension level, more general concepts



Time Dimension

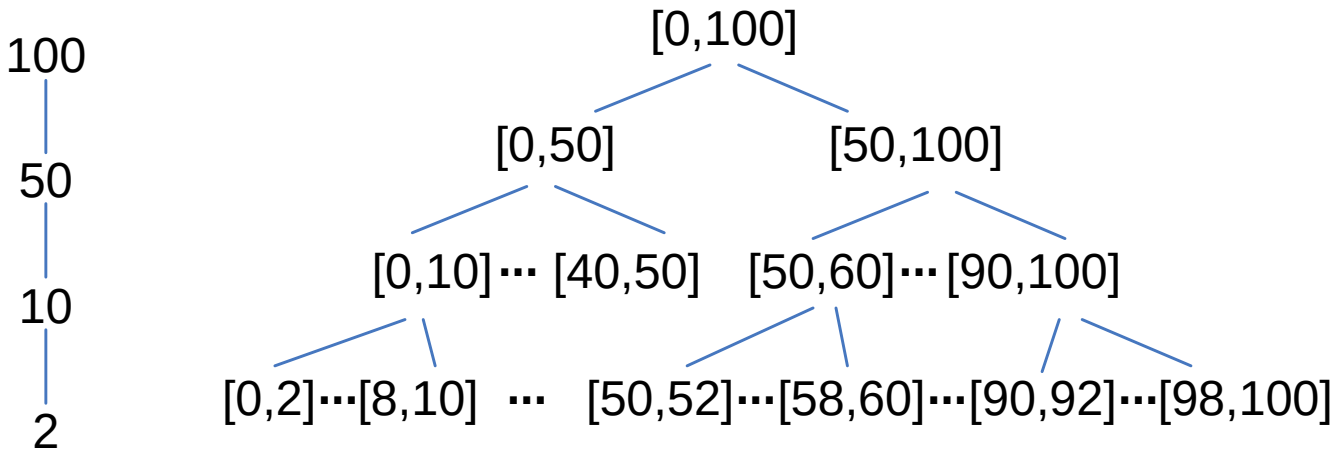


time	day	week	month	quarter	year
	1	1	1	1	2000
	2	1	1	1	2000
	3	1	1	1	2000
	4	1	1	1	2000
	5	1	1	1	2000
	6	1	1	1	2000
	7	1	1	1	2000
	8	2	1	1	2000



day < {month < quarter; week} < year

Price Dimension



Fact

- Fact (事实)
  - Fact Attribute (事实属性)
  - Measure (度量)
  - Aggregation (聚集)
- Measures populate the cells of a logical cube with the facts collected about business operations;
- Measures are organized by dimensions;
- Measure is the lowest level of detail; Users may never view this base level data, but it determines the types of analysis that can be performed;
- Values of measures are aggregated;

Fact

- Under the dimensions of *Time*, *Location*, *Production*:
  - Measure (度量)
    - 销量(个数)，销量(金额)，库存(个数)，利润(金额)
  - Aggregation (聚集)
    - 平均，最大，最小

相关知识（1）

- 定义1：设R为定义在集合X上的二元关系，如果对于每个$x \in X$，有xRx，则称二元关系R是自反的。例：实数集合中的“≤”。
- 定义2：设R为定义在集合X上的二元关系，如果对于每个$x, y \in X$，每当xRy，就有yRx，则称集合X上关系R是对称的。例：平面上三角形的相似关系。
- 定义3：设R为定义在集合X上的二元关系，如果对于任意$x, y, z \in X$，每当xRy，yRz时就有xRz，则关系R在X上是传递的。例：实数集合中的 “<”和“=”。

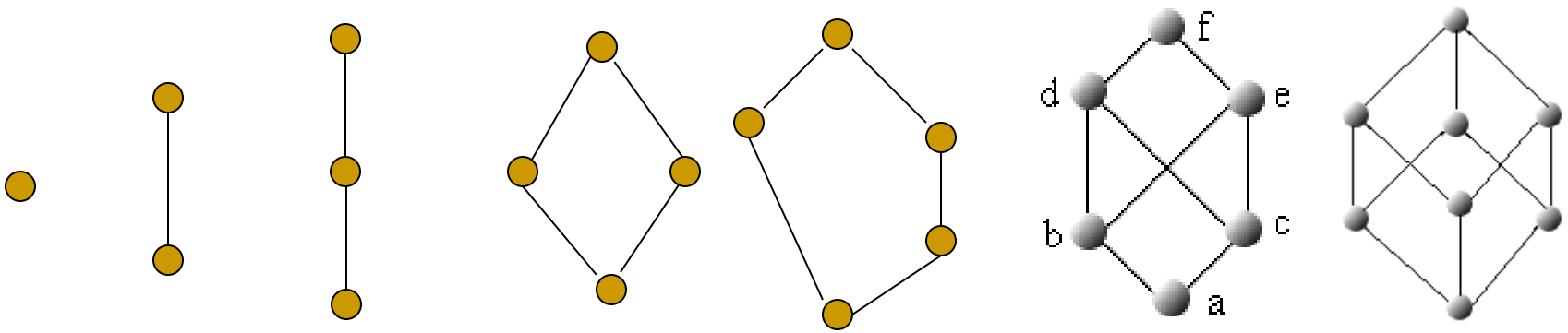
相关知识（2）

- 定义4：设R为定义在集合X上的二元关系，如果对于每个$x, y \in X$，每当xRy和yRx必有$x=y$，则称R在X上是反对称的。例：实数集合中的“≤”。
- 定义5：设A是一个集合，如果A上的一个关系R，满足自反性、反对称性和传递性，则称R是A上的一个偏序关系，记为“$\preccurlyeq$”。序偶$\langle A, \preccurlyeq \rangle$称作偏序集。
- 定义6：设$\langle A, \preccurlyeq \rangle$是一个偏序集合，在$A$的一个子集中，如果每两个元素都是有关系的，则称这个子集为链。

相关知识（3）

- 定义7：在偏序集〈$A, \preccurlyeq$〉中，如果A是一个链，则称〈$A, \preccurlyeq$〉为全序集(线序集)，在这种情况下，二元关系$\preccurlyeq$称为全序关系(称线序关系)。
- 定义8：设〈$A, \preccurlyeq$〉是一个偏序集，如果A中任意两个元素都有最小上界和最大下界，则称〈$A, \preccurlyeq$〉为格。

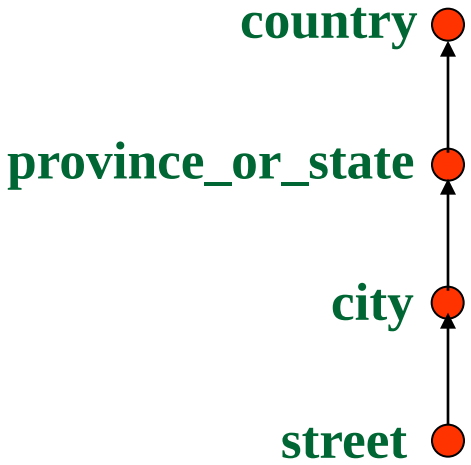
例：



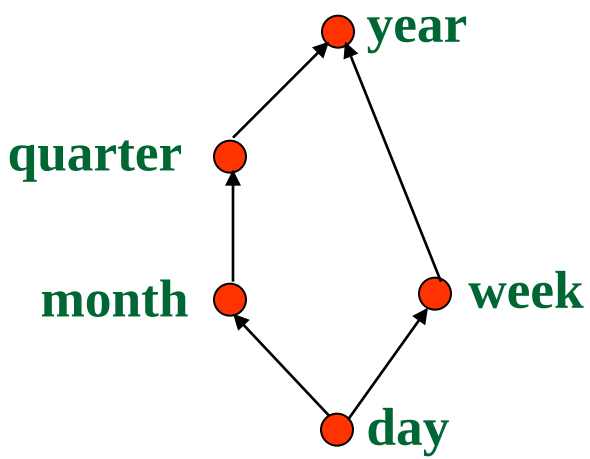
Specification of hierarchies (I)

- Schema hierarchy
  - A dimension hierarchy that is a total or partial order among attributes in a database schema

total order



lattice (partial order)



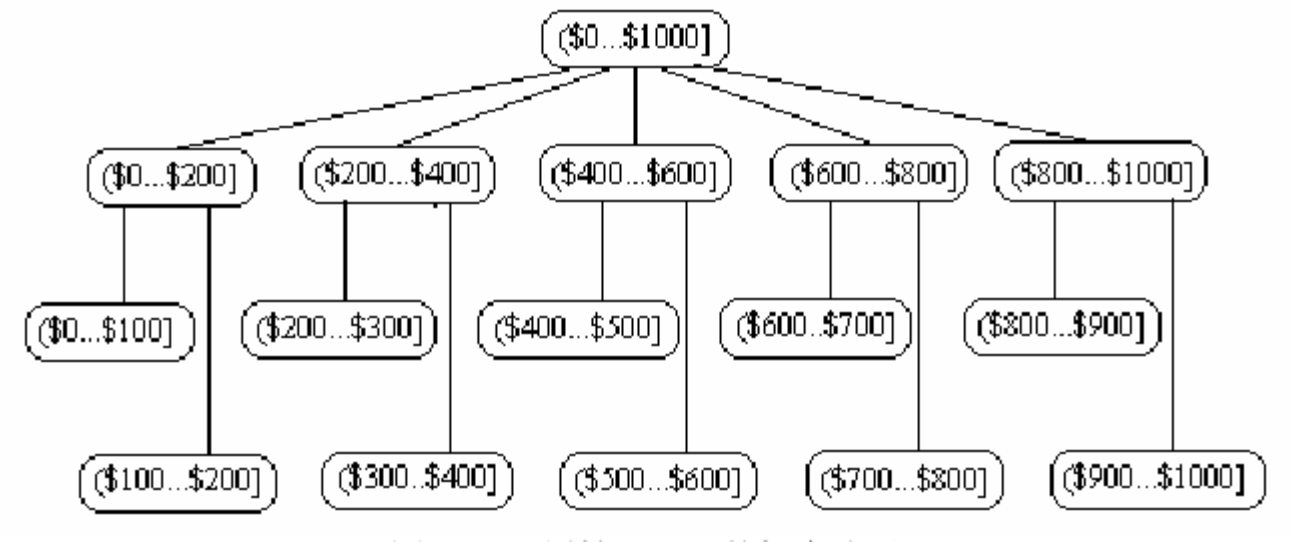
street < city < province < country

day < {month < quarter; week} < year

A dimension hierarchy may involve a single attribute or several attributes

Specification of hierarchies (II)

- Set-grouping hierarchy
  - A dimension hierarchy that is defined by discretizing or grouping dimension values for a given dimension attribute

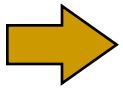


There may be more than one dimension hierarchy for a given dimension, based on different user viewpoints

Aggregates

- Add up amounts for day 1
- In SQL: SELECT sum(amt) FROM SALE
- WHERE date = 1

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4

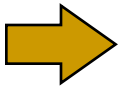


81

Aggregates

- Add up amounts by day
- In SQL: SELECT date, sum(amt) FROM SALE GROUP BY date

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4

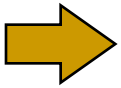


ans	date	sum
	1	81
	2	48

Aggregates

- Add up amounts by day, product
- In SQL: SELECT date, sum(amt) FROM SALE GROUP BY date, prodId

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4



sale	prodId	date	amt
	p1	1	62
	p2	1	19
	p1	2	48

rollup →

← drill-down

Aggregates

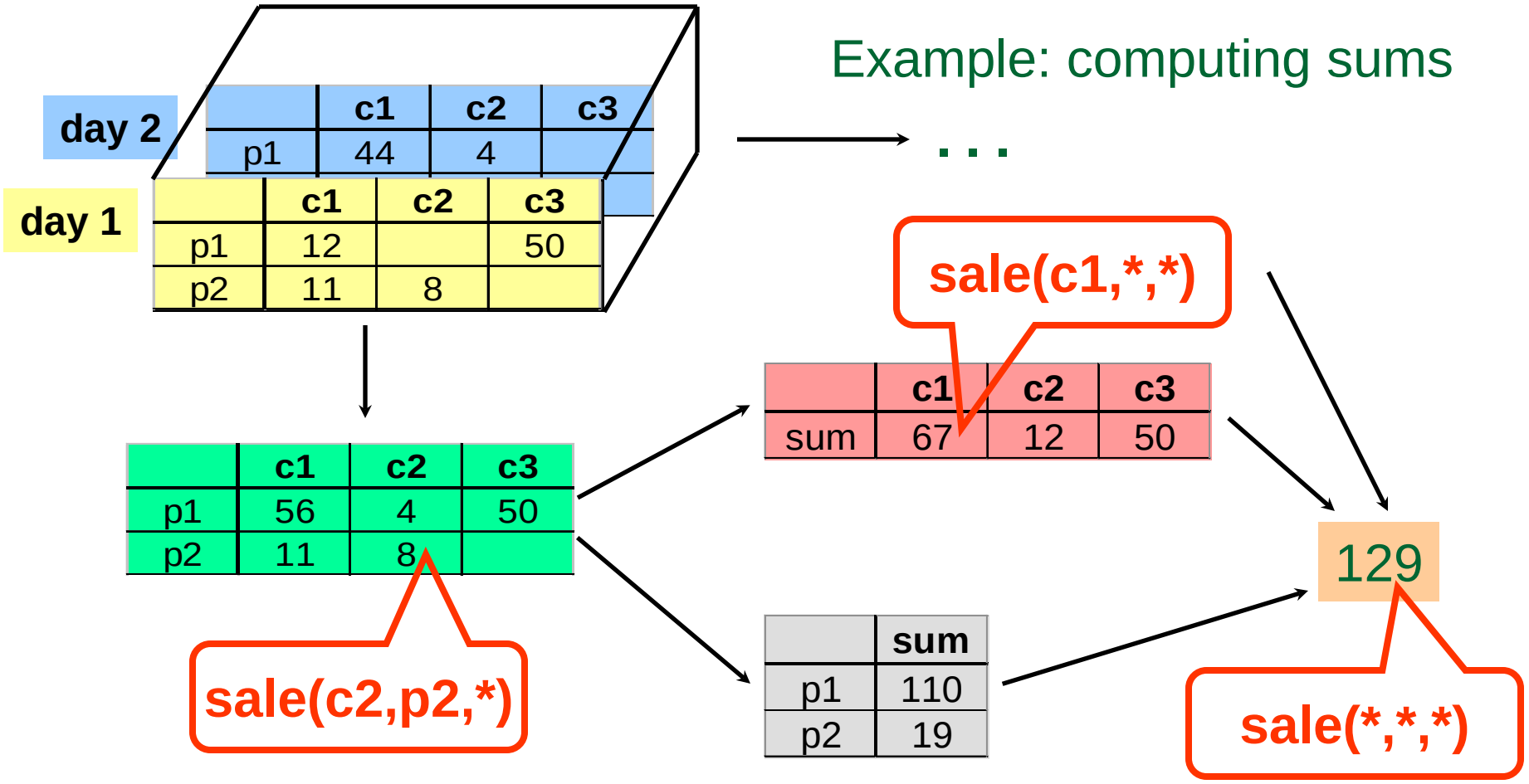
- Operators: sum, count, max, min, median, average(mean)
- “Having” clause
- Using dimension hierarchy
  - average by region (within store)
  - maximum by month (within date)

Cube Aggregation

rollup

drill-down

Example: computing sums



Aggregation

- Distributive: if the result derived by applying the function to *n* aggregate values is the same as that derived by applying the function on all the data without partitioning
  - E.g., count(), sum(), min(), max()
- Algebraic: if it can be computed by an algebraic function with *M* arguments (where *M* is a bounded integer), each of which is obtained by applying a distributive aggregate function
  - E.g., avg(), min_N(), standard_deviation()
- Holistic: if there is no constant bound on the storage size needed to describe a subaggregate.
  - E.g., median() 中位数, mode() 众数, rank() 秩

Aggregation

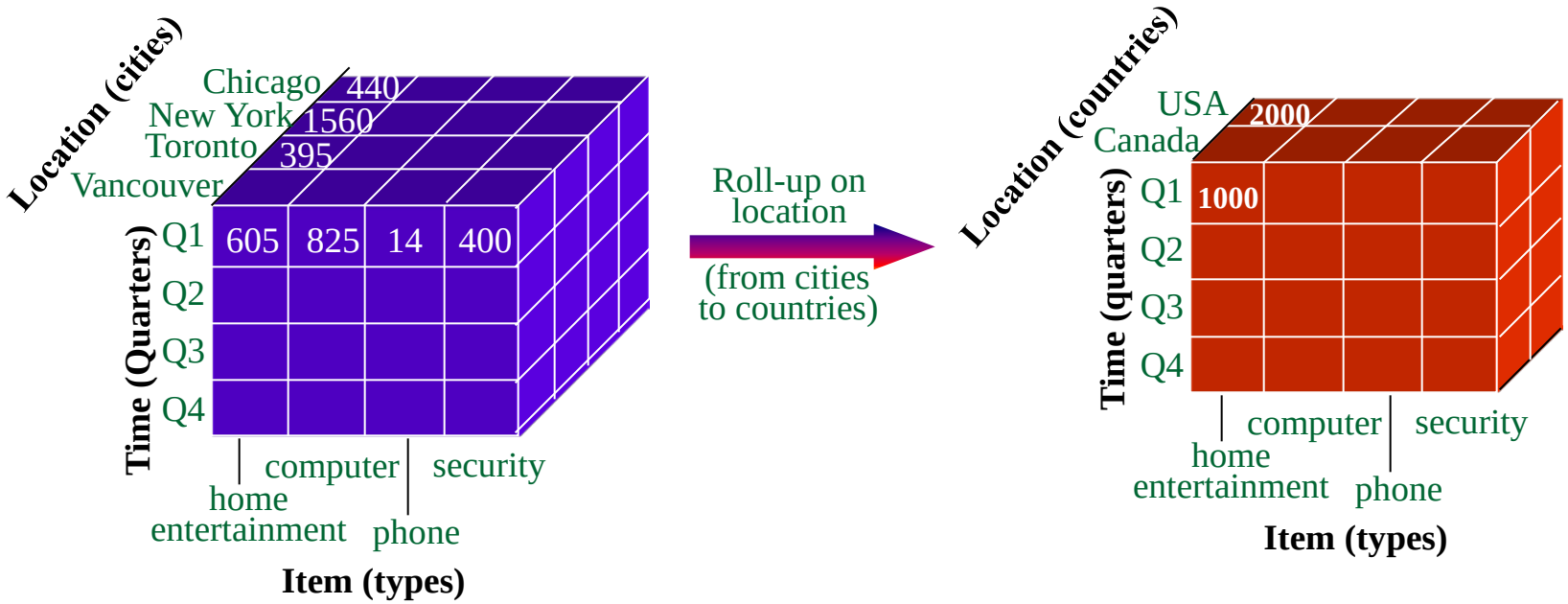
分布的：一个聚集函数是分布的，如果它能以如下分布方式进行计算：设数据被划分为 n 个集合，函数在每一部分上的计算得到一个聚集值。如果将函数用于 n 个聚集值得到的结果，与将函数用于所有数据得到的结果一样，则该函数可以用分布方式计算。例如，count()可以这样计算：首先将数据方分割成子方的集合，对每个子方计算 count()，然后对这些子方得到的计数求和。因此，count()是分布聚集函数。同理，sum(), min()和 max()是分布聚集函数。一个度量是分布的，如果它可以用分布聚集函数得到。

代数的：一个聚集函数是代数的，如果它能够由一个具有 M（其中，M 是一个整数界）个参数的代数函数计算，而每个参数都可以用一个分布聚集函数求得。例如，avg()可以由 sum()/count()计算，其中 sum()和 count()是分布聚集函数。类似地，可以表明 min_N(), max_N()和 standard_deviation()是代数聚集函数。一个度量是代数的，如果它可以用代数聚集函数得到。

整体的：一个聚集函数是整体的，如果描述它的子聚集所需的存储没有一个常数界。即，不存在一个具有 M 个（其中，M 是常数）参数的代数函数进行这一计算。整体函数的常见例子包括 median(), mode()（即，最常出现的项），和 rank()。一个度量是整体的，如果它可以用整体聚集函数得到。

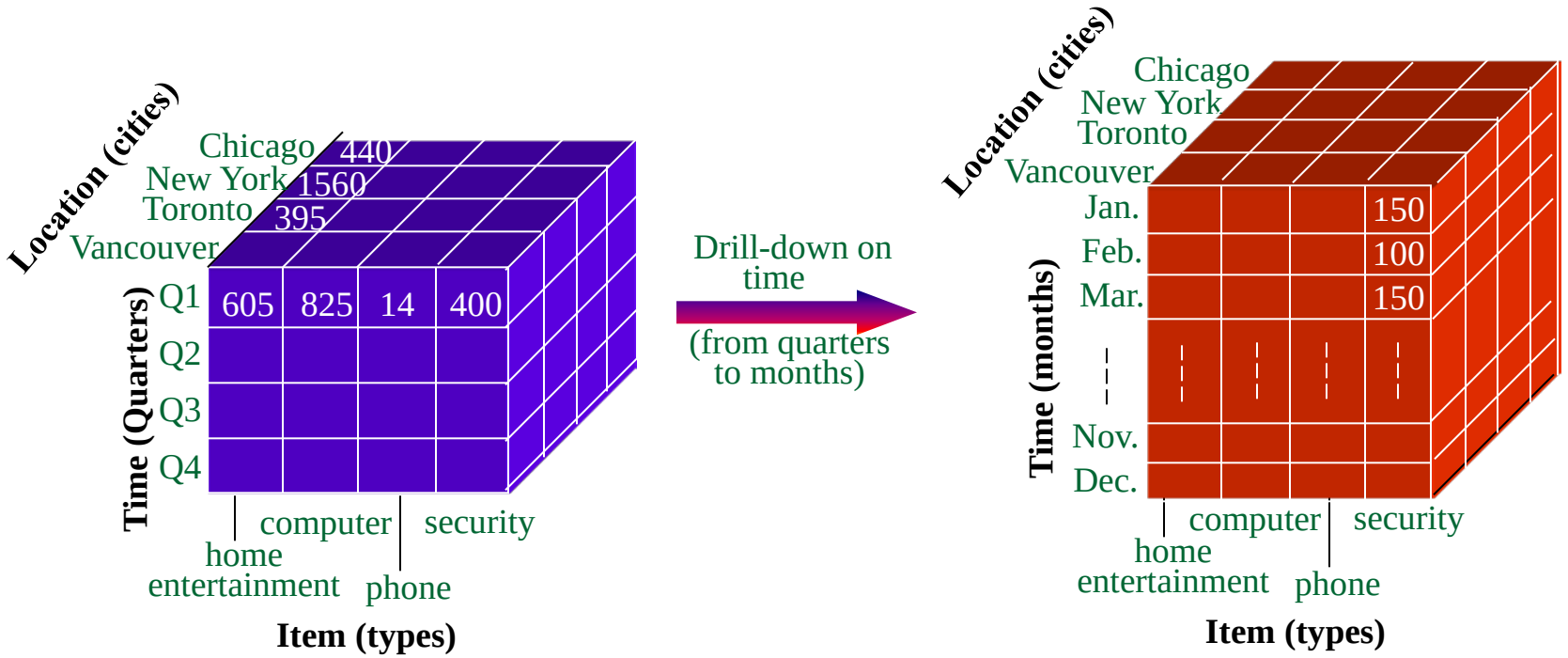
OLAP operations on data cube (I)

- **OLAP: Online Analytical Processing**
- **Roll-up:** performs aggregation on a data cube, either by climbing up a dimension hierarchy for a dimension or by dimension reduction



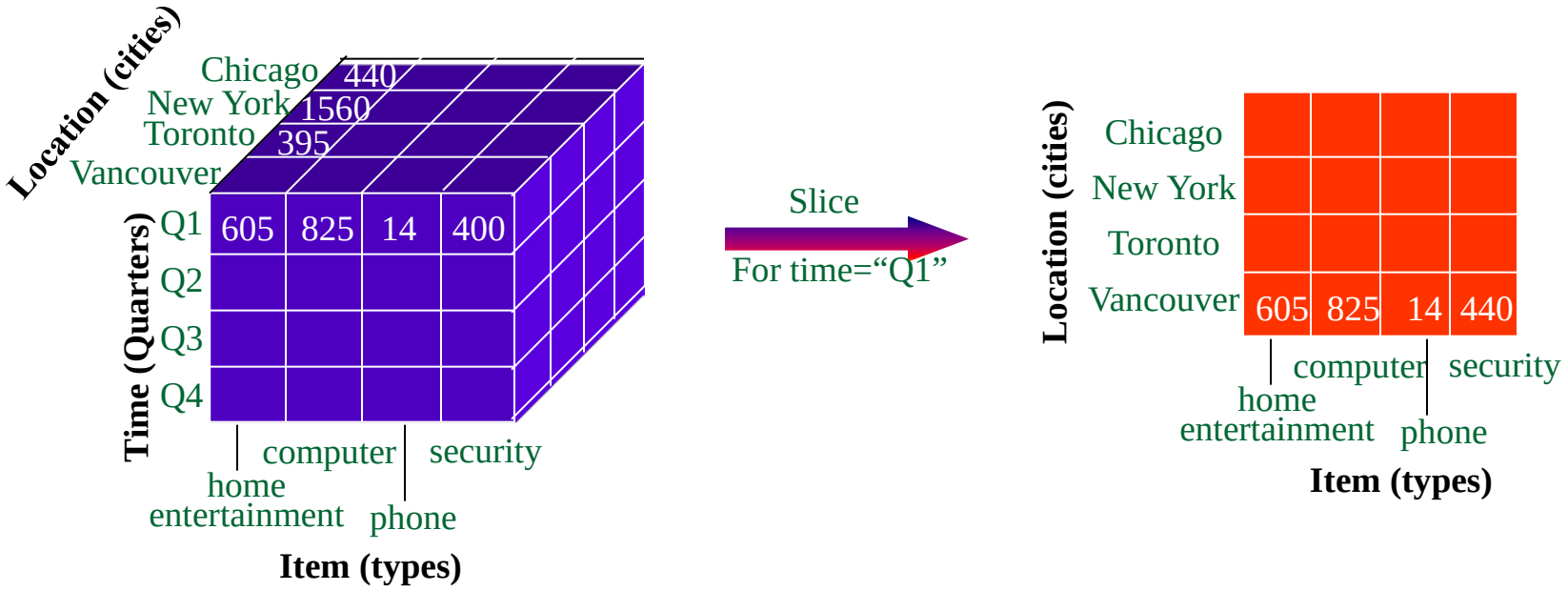
OLAP operations on data cube (II)

- **Drill-down**: navigates a data cube from less detailed data to more detailed data, either by stepping-down a dimension hierarchy for a dimension or by introducing additional dimensions



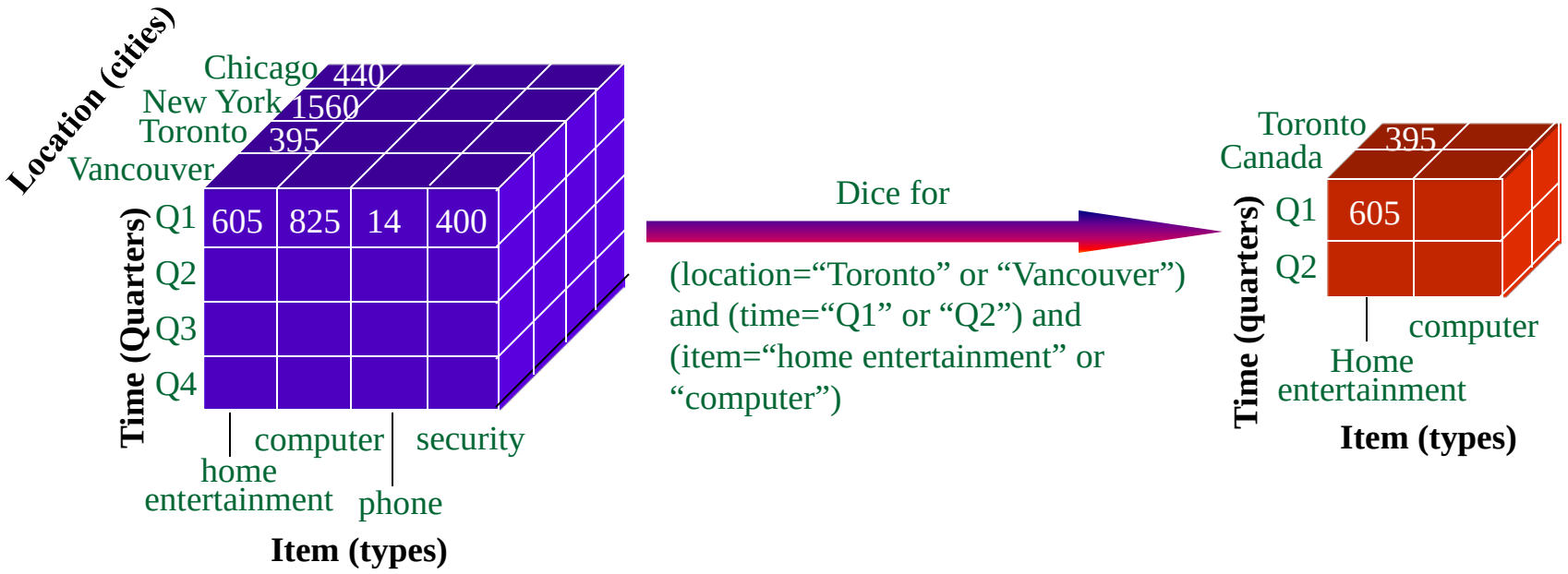
OLAP operations on data cube (III)

- **Slice**: performs a selection on one dimension of a data cube in a specified dimension level.



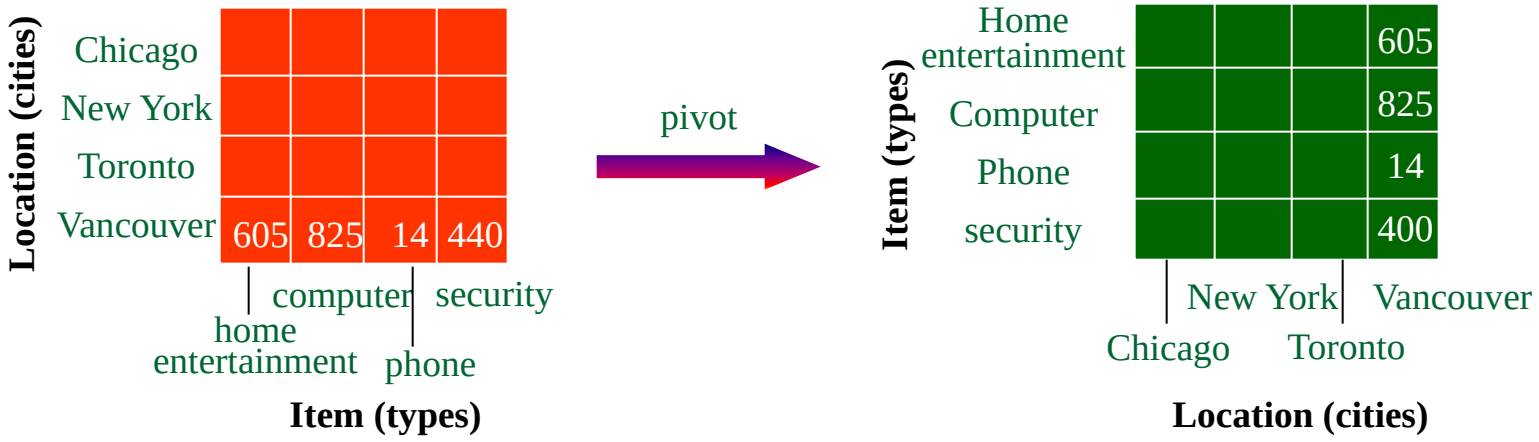
OLAP operations on data cube (IV)

- **Dice**: performs a selection on multiple dimensions of a data cube in a specified dimension levels.



OLAP operations on data cube (V)

- **Pivot**: rotates the data axes in view, also called rotate

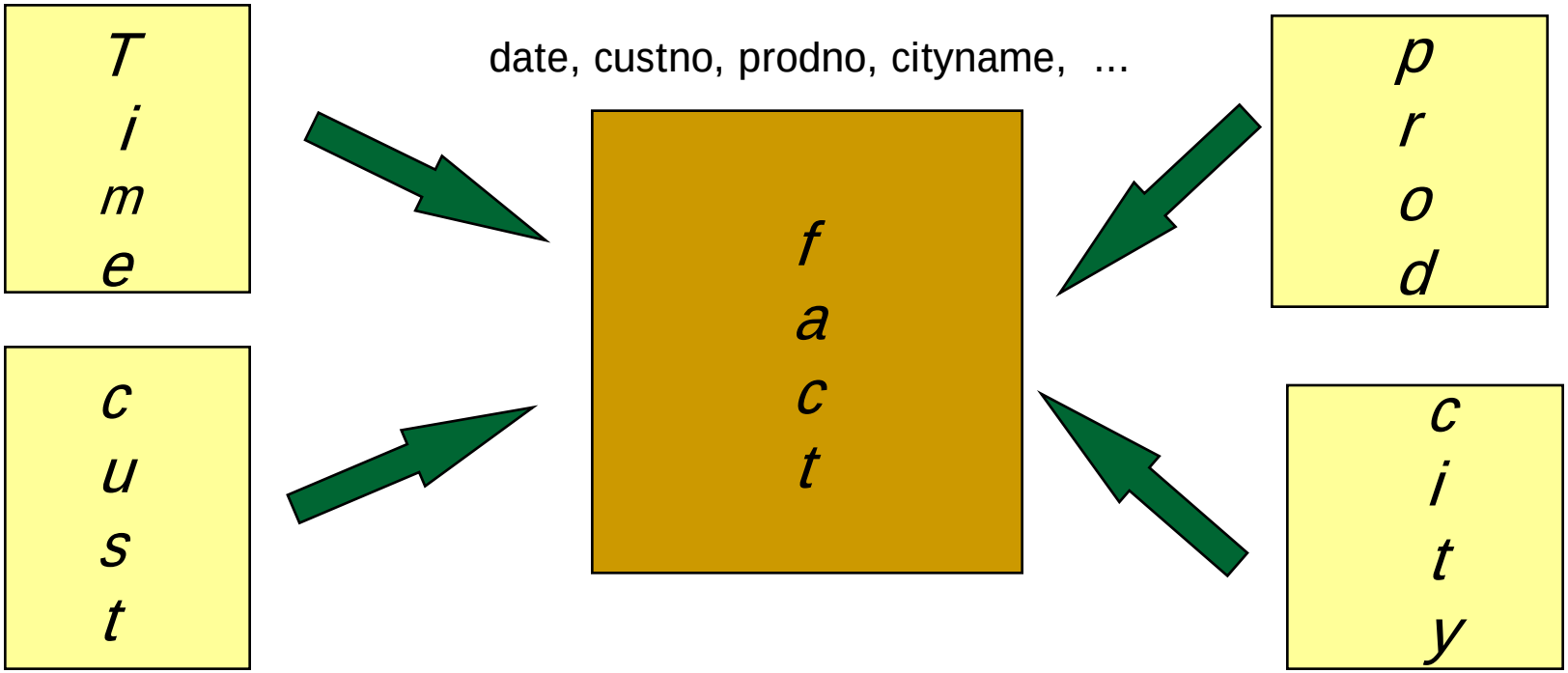


Multidimensional database schemas

- The multidimensional data model for data warehouse can exist in the form of Star Schema, Snowflake Schema, or Fact Constellation Schema
- In those schemas:
  - each dimension is represented by a dimension table that describes the dimension
  - Numerical measures are called facts. A fact table contains the facts as well as keys to each of the related dimension tables.

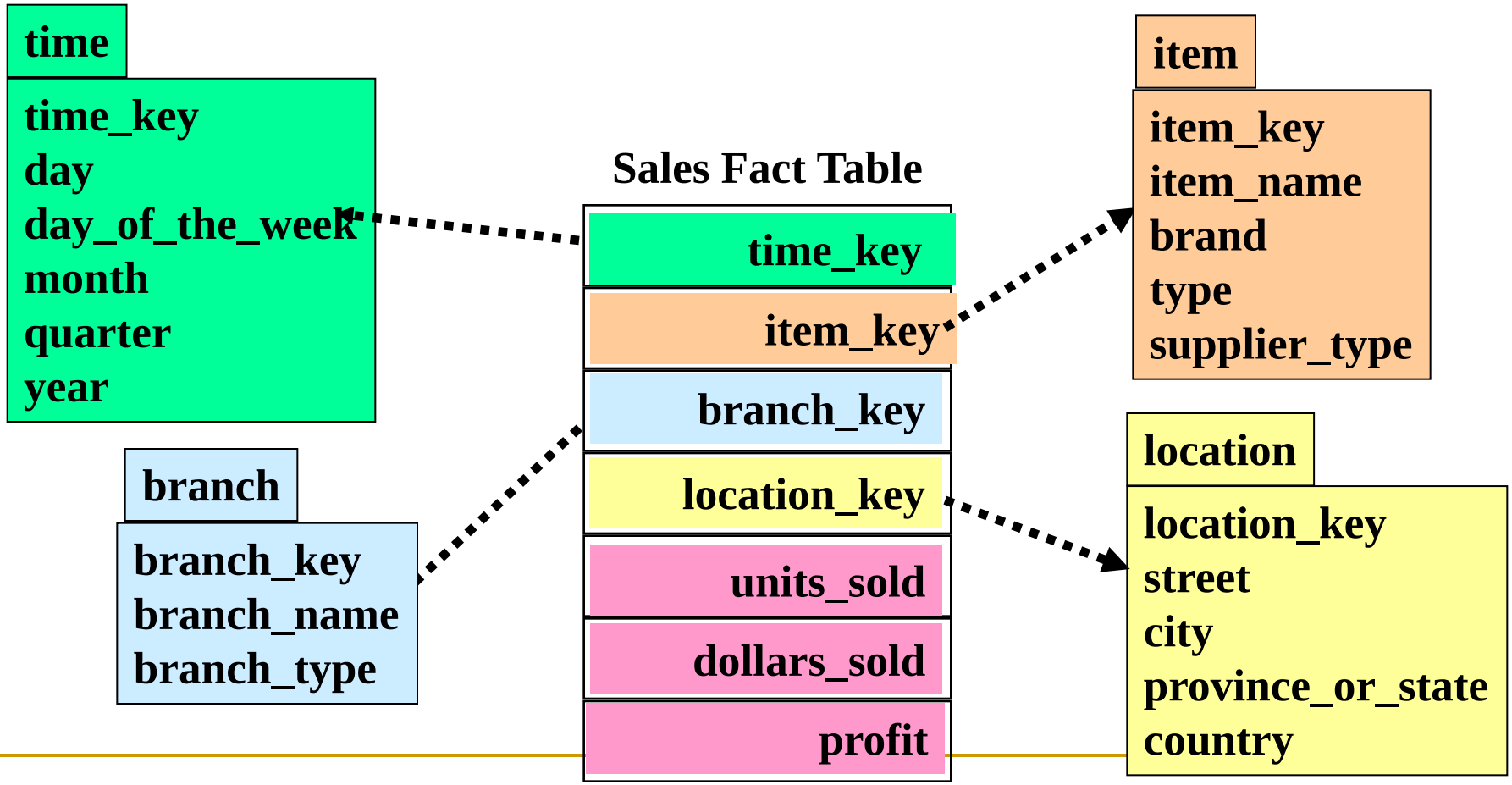
Star Schema

- A single fact table and for each dimension one dimension table
- Does not capture hierarchies directly



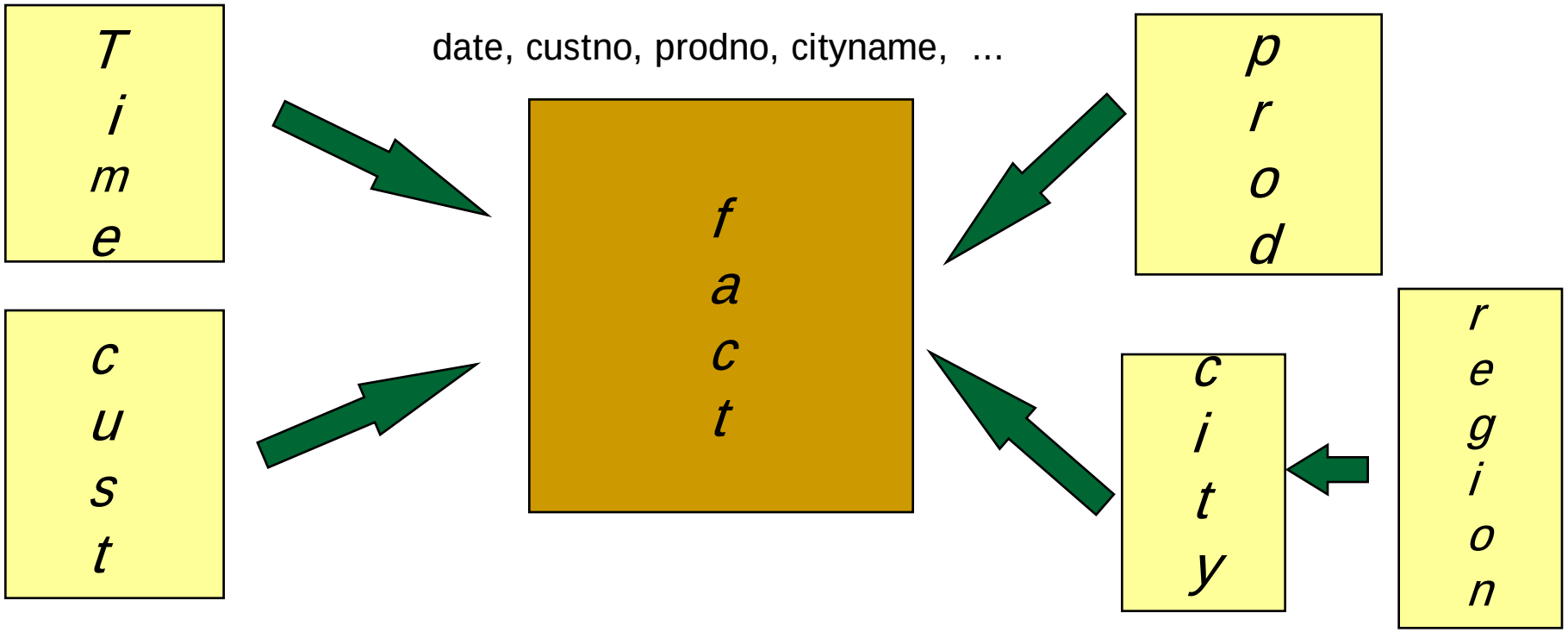
Star schema

- A fact table in the middle connected to a set of dimension tables



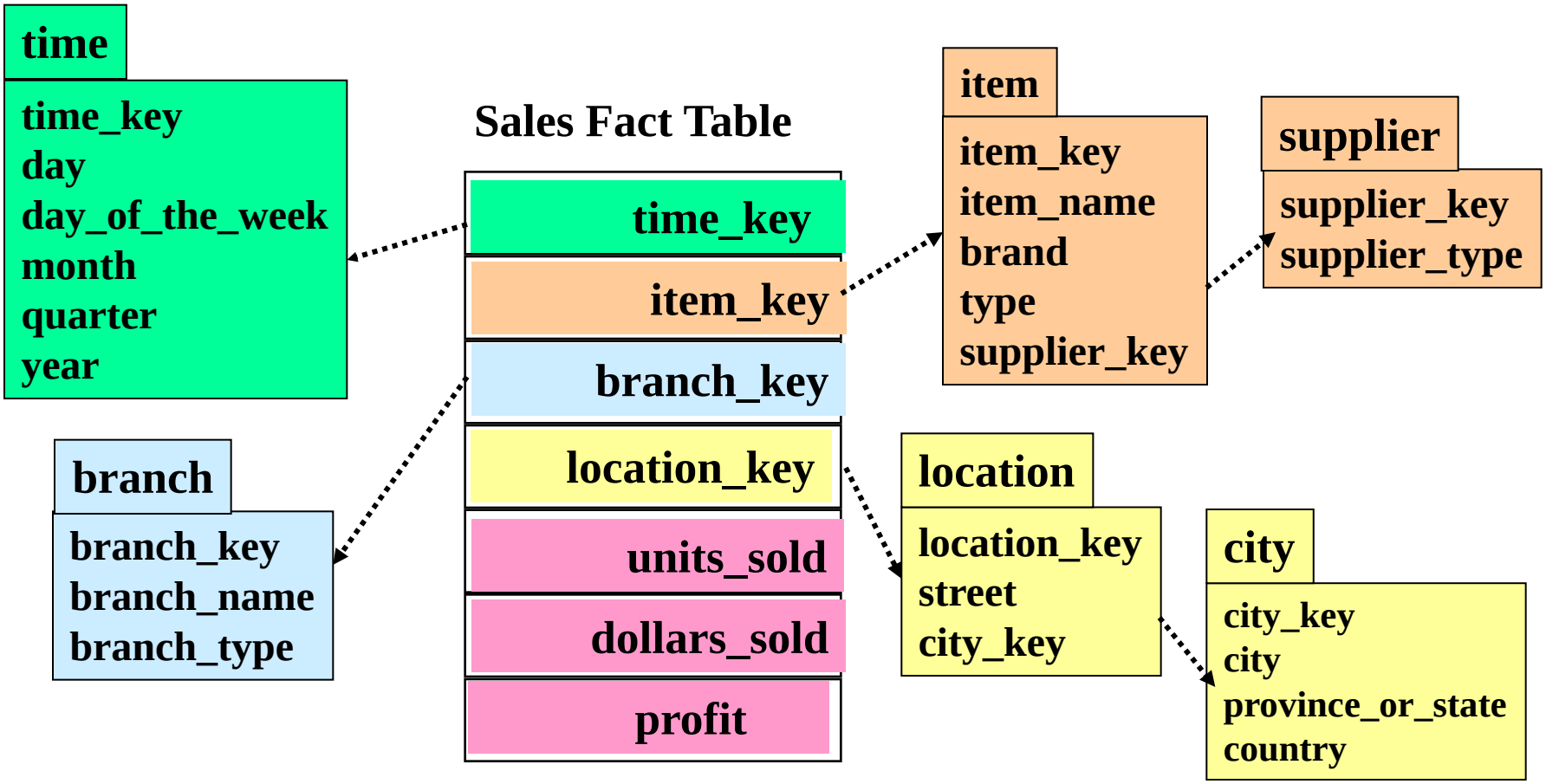
Snowflake schema

- Represent dimension hierarchy directly by normalizing tables.
- Easy to maintain and saves storage



Snowflake schema

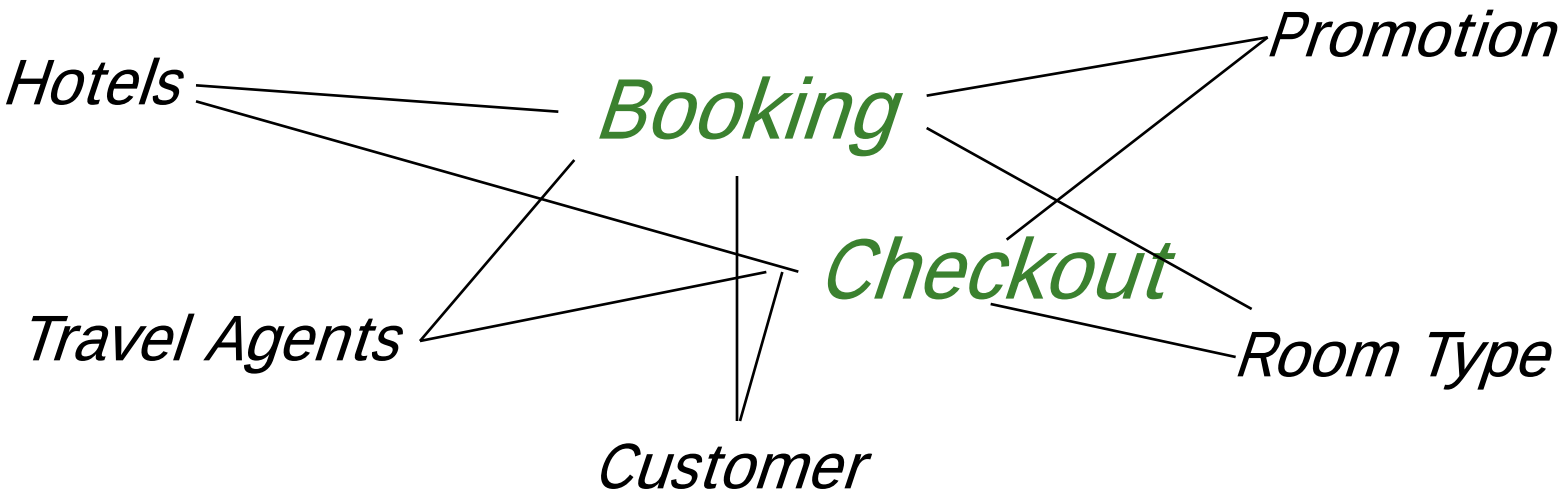
- A variant of star schema where some dimension tables are normalized into a set of smaller dimension tables



A compromise between star schema and snowflake schema is to adopt a mixed schema where only the very large dimension tables are normalized

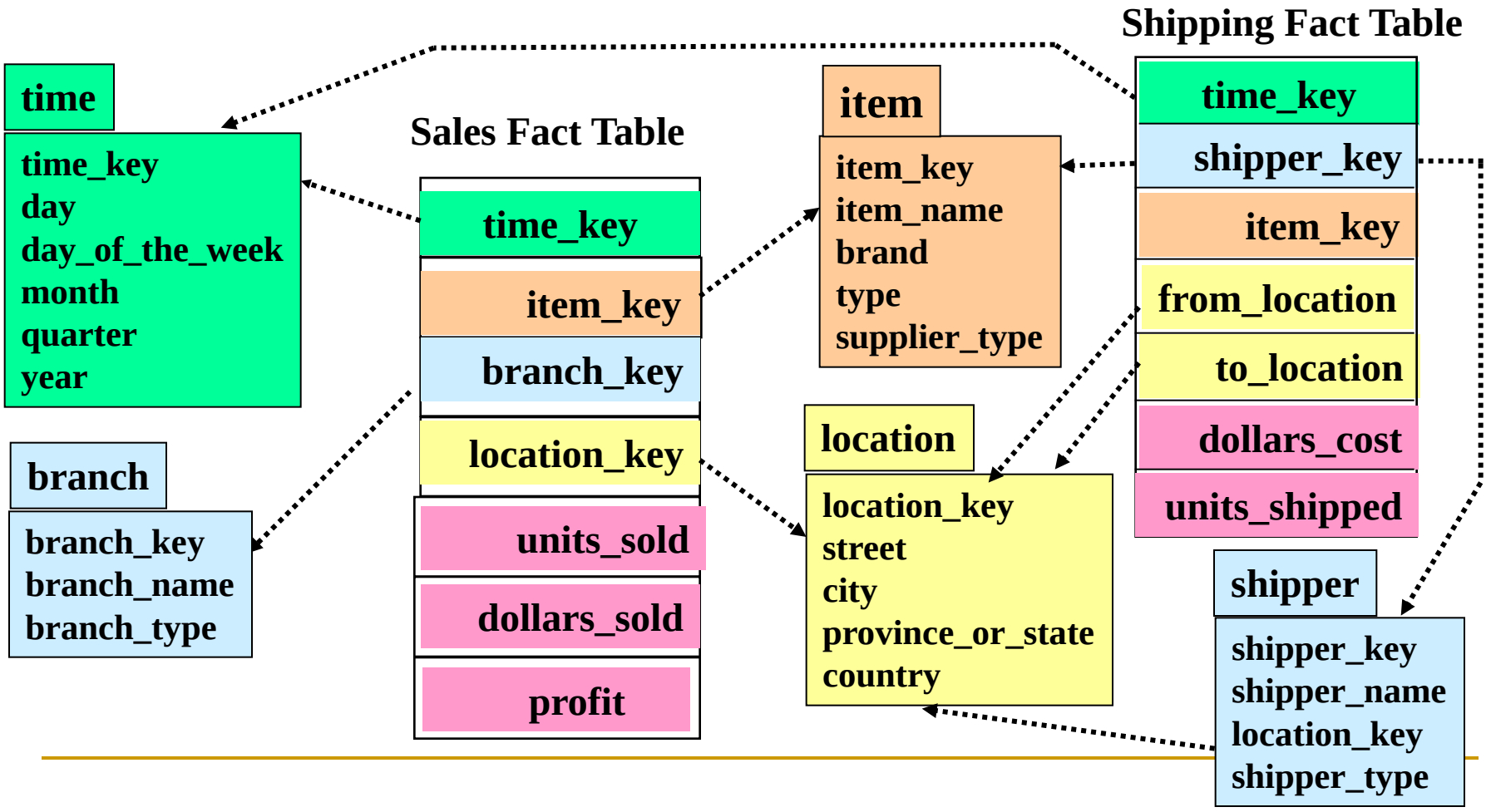
Fact Constellation

- Fact Constellation
  - Multiple fact tables that share many dimension tables
  - Booking and Checkout may share many dimension tables in the hotel industry



Fact constellation schema

- also called galaxy schema
- time_key Multiple fact tables share some dimension tables



Dimension Tables

- Dimension tables
  - Define business in terms already familiar to users
  - Wide rows with lots of descriptive text
  - Small tables (about a million rows)
  - Joined to fact table by a foreign key
  - heavily indexed
  - typical dimensions
    - time periods, geographic region (markets, cities), products, customers, salesperson, etc.

Fact Table

- Central table
  - mostly raw numeric items
  - narrow rows, a few columns at most
  - large number of rows (millions to a billion)
  - Access via dimensions

De-normalization

- Normalization in a data warehouse may lead to lots of small tables
- Can lead to excessive I/O's since many tables have to be accessed
- De-normalization is the answer especially since updates are rare

Selective Redundancy

- Dimension attribute can be stored redundantly in fact table
  - Most often dimension attribute is also accessed with fact table
- Updates have to be careful

An Example of Creating Data Warehouse

Time dimension table

Time-key	day	mouth	quarter	year
T1	1	1	1	2003
T2	2	1	1	2003
……	……	……	……	……

fact table

Time-key	Item-key	Location-key	price	number
……	……	……	……	……
T1	I2	L2	300	5

Item dimension table

item-key	name	brand	type	supplier
I1	TV	sonny	SN1	S1
I2	DVD	厦新	NC2	S2
……	……	……	……	……

销售事实：2009年1月1日销售5台中国福建生产、由S2供应的厦新DVD

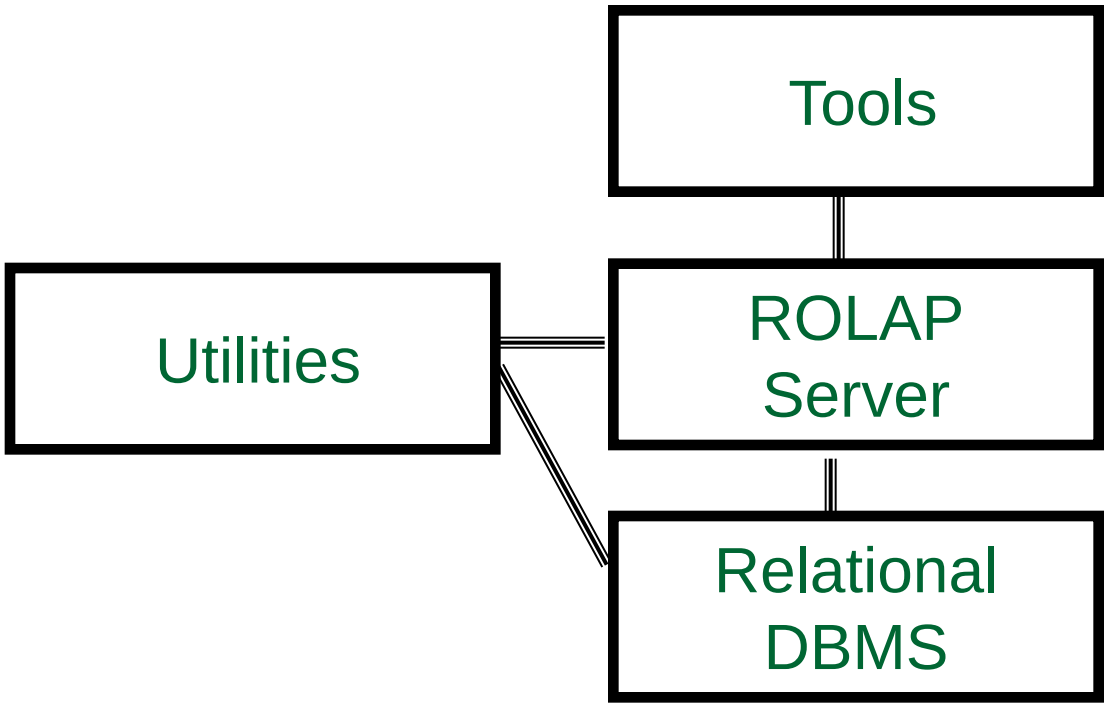
Location dimension table

location-key	street	city	province	country
L1	为民	大连	辽宁	中国
L2	清水	厦门	福建	中国
……	……	……	……	……

ROLAP Server

- Relational OLAP Server

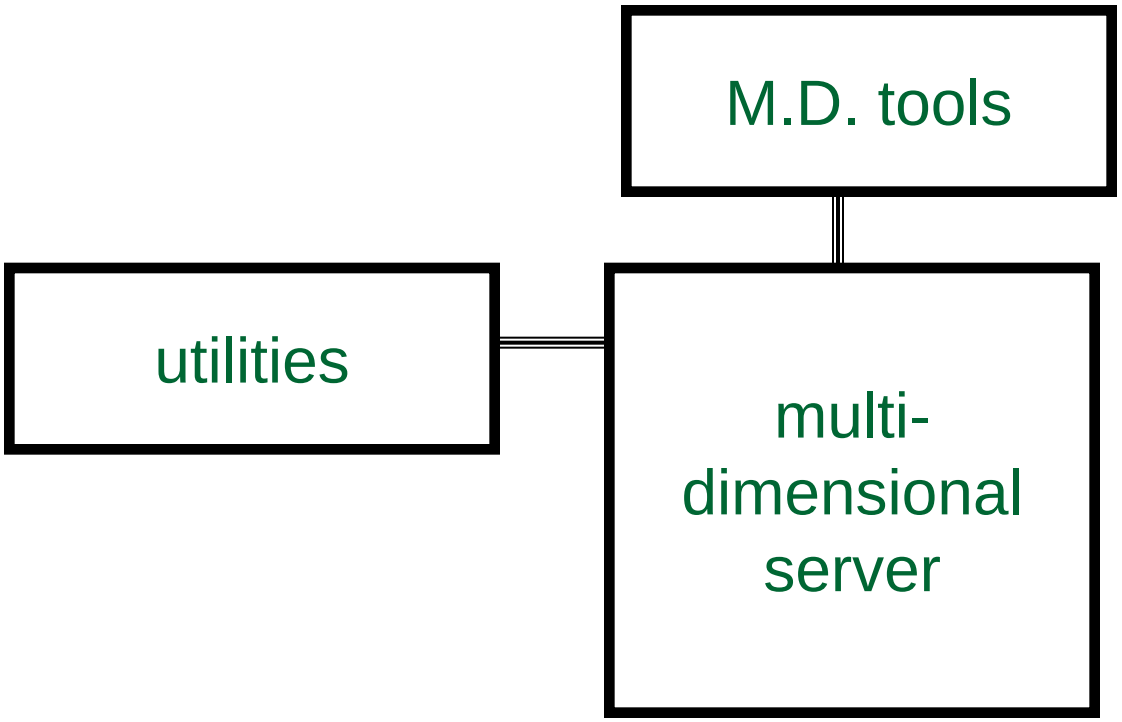
sale	prodId	date	sum
	p1	1	62
	p2	1	19
	p1	2	48

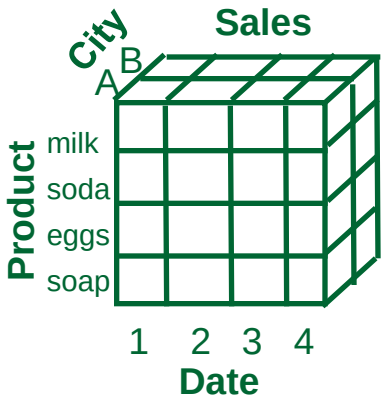


Special indices, tuning;
Schema is “denormalized”

MOLAP Server

- Multi-Dimensional OLAP Server





could also sit on relational DBMS

OLAP Server Architectures

- ROLAP (Relational OLAP)
  - Use relational or extended-relational DBMS to store and manage warehouse data and OLAP middle ware to support missing pieces
  - Include optimization of DBMS backend, implementation of aggregation navigation logic, and additional tools and services
  - Good scalability
- MOLAP (Multidimensional OLAP)
  - Array-based multidimensional storage engine (sparse matrix techniques)
  - Fast indexing to pre-computed summarized data
- HOLAP (Hybrid OLAP)
  - User flexibility e.g. low level: relational, high-level: array
  - Specialized SQL servers
  - Specialized support for SQL queries over star/snowflake schemas

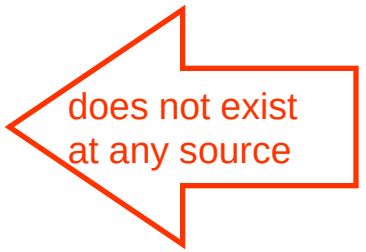
Materialized Views

- Define new relations using SQL expressions

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4

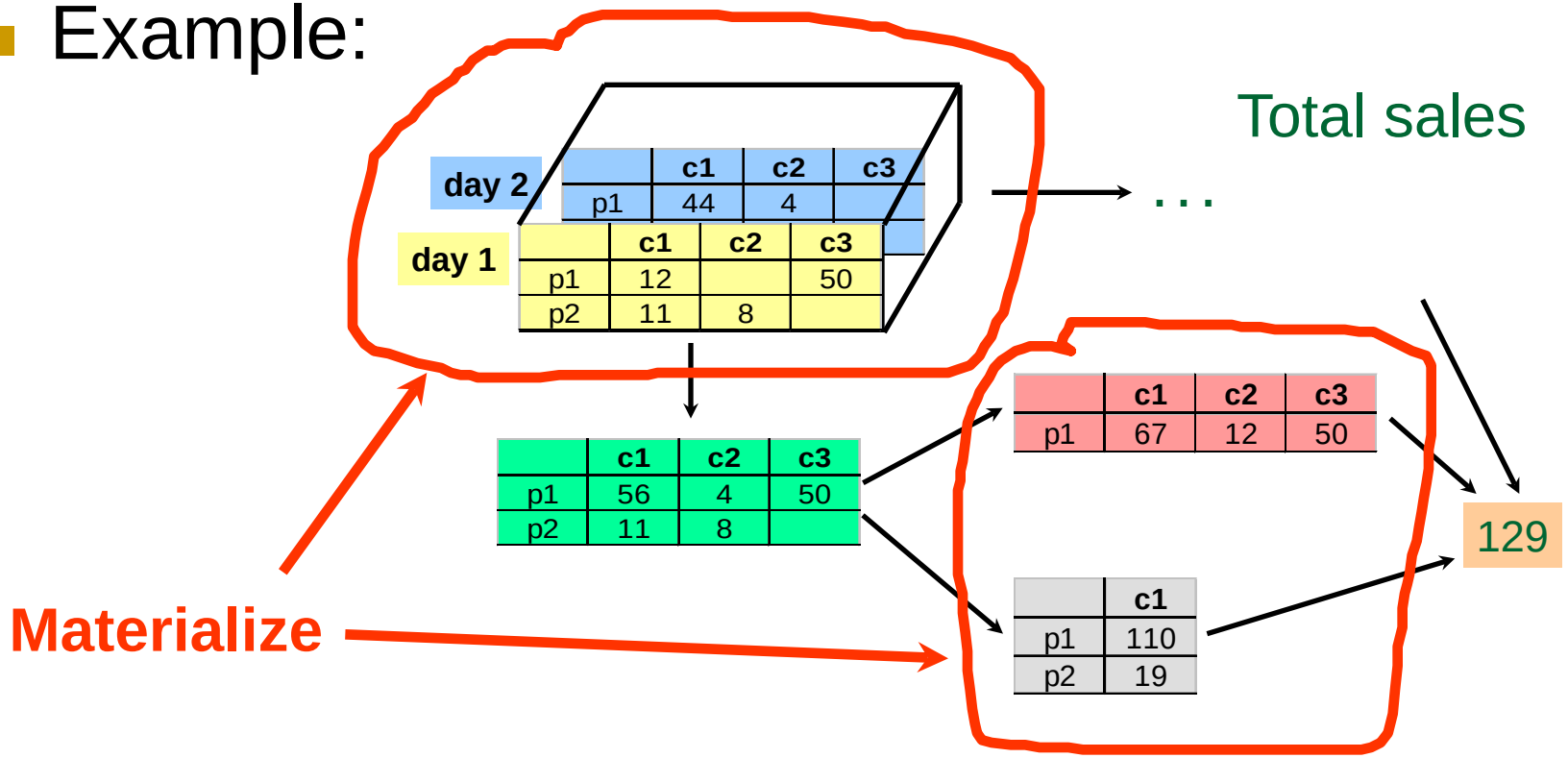
product	id	name	price
	p1	bolt	10
	p2	nut	5

joinTb	prodId	name	price	storeId	date	amt
	p1	bolt	10	c1	1	12
	p2	nut	5	c1	1	11
	p1	bolt	10	c3	1	50
	p2	nut	5	c2	1	8
	p1	bolt	10	c1	2	44
	p1	bolt	10	c2	2	4



How to Materialize?

- Results which are useful for common queries
- Store in warehouse
- Example:



Why precomputation?

- OLAP engines demand that decision support queries be answered in the order of seconds
- Pre-computation of a data cube leads to fast response time and avoids some redundant computation
- Data cube can be viewed as a lattice of cuboids
- How many cuboids are there in an n-D data cube?
  - If there are no dimension hierarchies

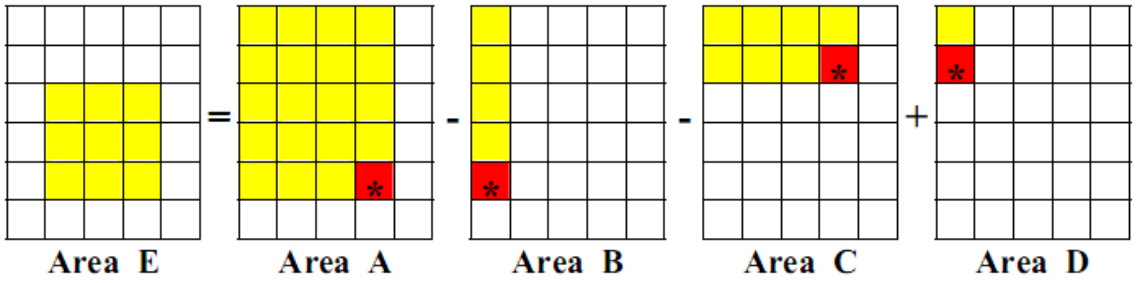
$$T = 2^n$$

  - If each dimension is associated with L_i levels of hierarchy

$$T = \prod_{i=1}^{n} (L_i + 1)$$

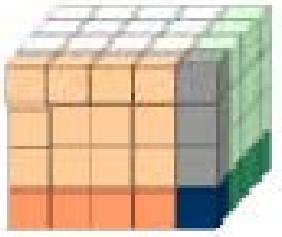
What's the best way of precomputation?

- **No materialization**
  - Pre-compute only the base cuboid
  - Slow response
- **Full materialization**
  - Pre-compute all the cuboids
  - Huge storage
- **Partial materialization**
  - Pre-compute some of the cuboids
  - Trade-off
- Selection of the cuboids to be materialized based on size, access frequency, sharing, etc.
- A popular approach: materialize the set of cuboids having relatively simple structure or always be used.



Efficient cube computation

- Since ROLAP and MOLAP employ different data structures (tuples and relational tables vs. multi-dimensional array), they utilize different cube computation techniques
- ROLAP-based cube computation
  - Based on some optimization techniques
  - Relatively slow
  - (Instead of cubing a table directly, it is even faster to convert the table to an array, cube the array, then convert the result back to a table)
- MOLAP-based cube computation
  - Based on multiway array aggregation
  - Relatively fast



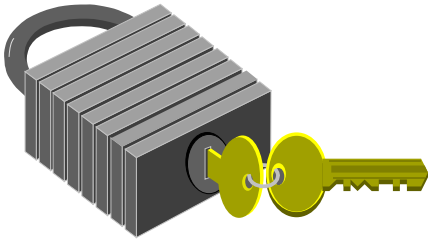
Let's go to the next …

数据仓库与数据挖掘
Data Warehouse & Data Mining

宋 杰
Song Jie
东北大学 软件学院
Software College, Northeastern University

3. Metadata and Data Marts

Metadata



Metadata is data about data

- **Metadata** is loosely defined as data about data
- Metadata definition provides information about the distinct items, such as:
  - Means of creation,
  - Purpose of the data,
  - Time and date of creation,
  - Creator or author of data,
  - Placement on a network (electronic form) where the data was created,
  - What standards used
  - etc.

Categories of data warehouse metadata

- Information on potential data source
- Information on data model
- Information on the mapping between operational data structure and warehouse data structure
- Information on the usage of warehouse contents (so that the performance of warehouse can be tuned)

Metadata repository

- A metadata repository should contain:
  - Description of the structure of the warehouse
    - Schema, view, dimensions, hierarchies, derived data definitions, data mart locations and contents
- Operational metadata
  - Data lineage (history of migrated data and transformations applied to it), currency of data (active, archived, or purged), monitoring information (warehouse usage statistics, error reports, audit trails)
- The algorithms used for summarization
- The mapping from operational environment to the data warehouse
- Data related to system performance
  - Indices, profiles, rules for the timing and scheduling of refresh, update, and replication cycles
- Business data
  - Business terms and definitions, data ownership information, charging policies

Usage of Metadata

- The backgrounds of the data
- The potential relationships of data from difficult “subjects”
- The index of the data
- The storage of additional data
- The integrated rules for heterogeneous data
- The bridge of data exchanging and sharing
- The mapping relationships of the codes and values

CRUD

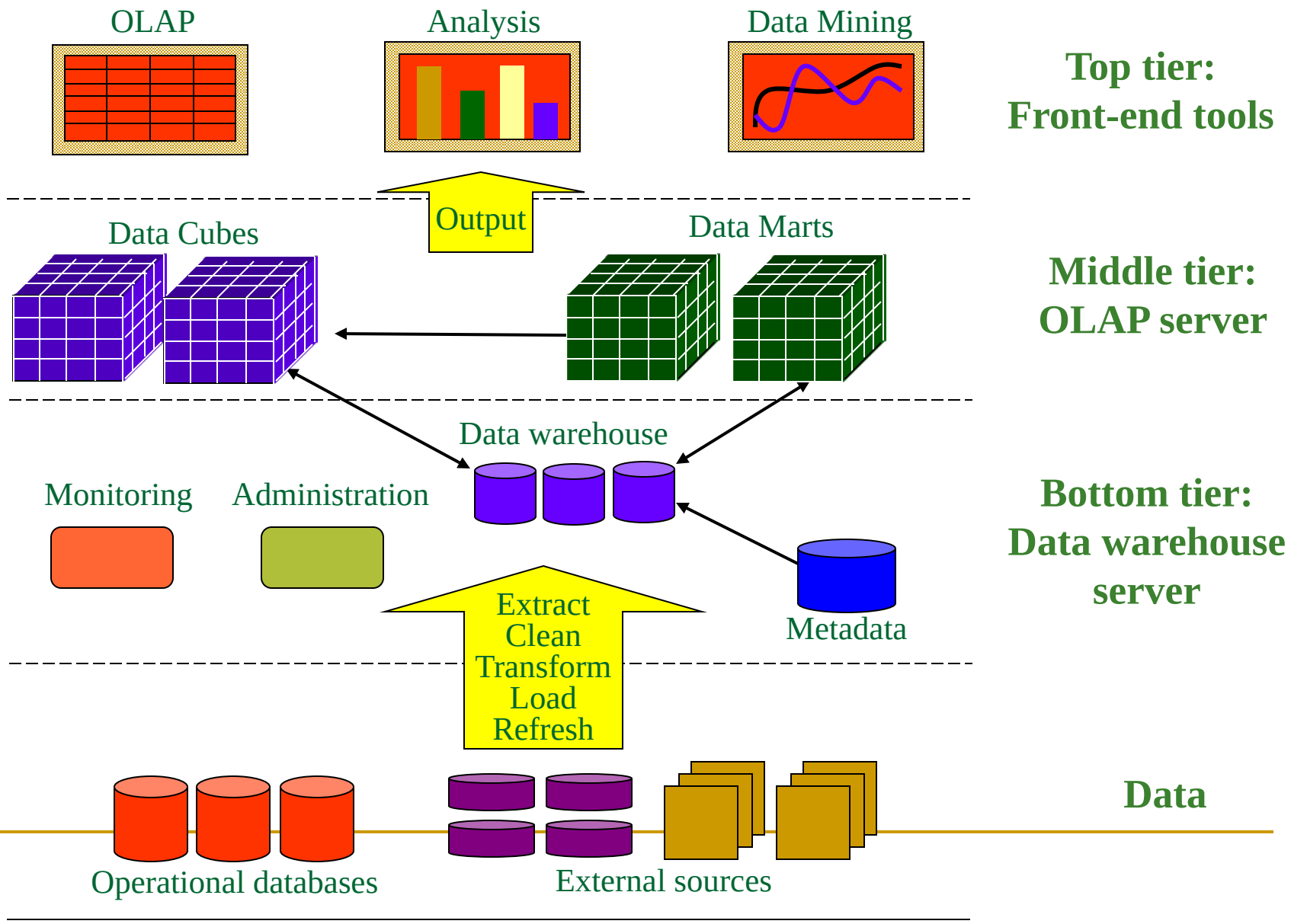
- In computing, CRUD is an acronym for create, retrieve, update, and delete. It is used to refer to the basic functions of a database or persistence layer in a software system.
  - Create new records
  - Read (Retrieve) existing records
  - Update existing records
  - Delete (Destroy) existing records.

Operation	SQL
Create	INSERT
Read (Retrieve)	SELECT
Update	UPDATE
Delete (Destroy)	DELETE

Applications on Metadata

- CRUD
- Auto-retrieve metadata from data
- Fast querying on metadata
- ETL on metadata

Multi-tiered Data Warehouse Architecture



Data Cube

- An Data Cube or OLAP Cube is a data structure that allows fast analysis of data.
- It can also be defined as the capability of manipulating and analyzing data from multiple perspectives.
- The arrangement of data into cubes overcomes some limitations of relational databases.

Data Marts

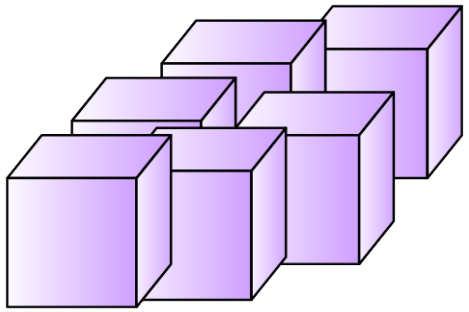
- A Data Mart (DM) is the access layer of the Data Warehouse (DW) environment that is used to get data out to the users. The DM is a subset of the DW, usually oriented to a specific business line or team.
  - Smaller warehouses
  - Spans part of organization
    - e.g., marketing (customers, products, sales)
  - Do not require enterprise-wide consensus

Reasons for creating a data mart

- Creates collective view by a group of users
- Improves end-user response time
- Ease of creation
- Lower cost than implementing a full data warehouse
- Potential users are more clearly defined than in a full data warehouse

Characteristics of the Departmental Data Mart

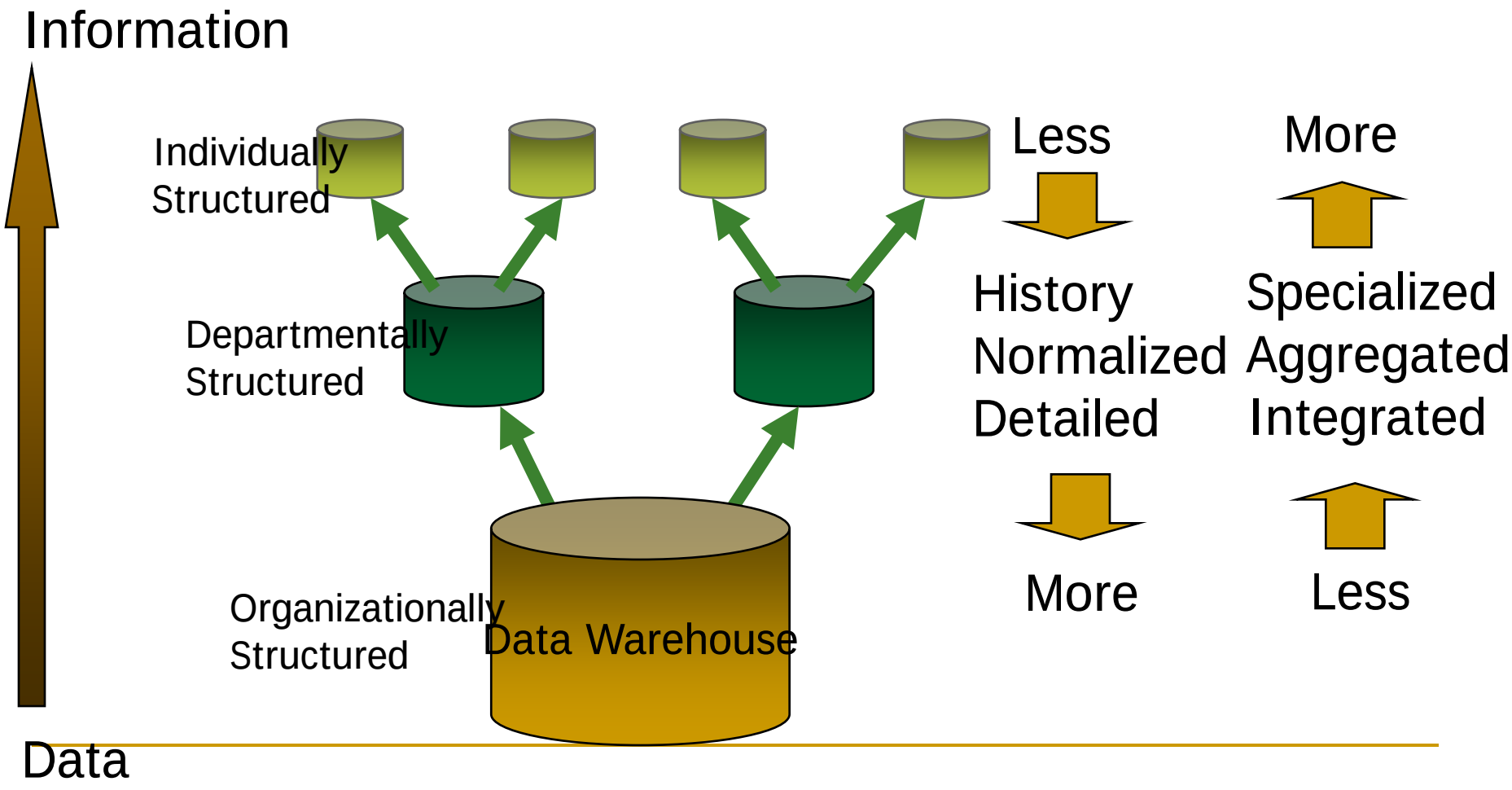
- OLAP
- Small
- Flexible
- Customized by department
- Source is departmentally structured data warehouse



Data Warehouse vs. Data Marts

- They are both
  - Multi-dimensional;
  - Clean
  - Aggregated
  - Integrated
  - Time-variant, and nonvolatile
- Which one comes first?

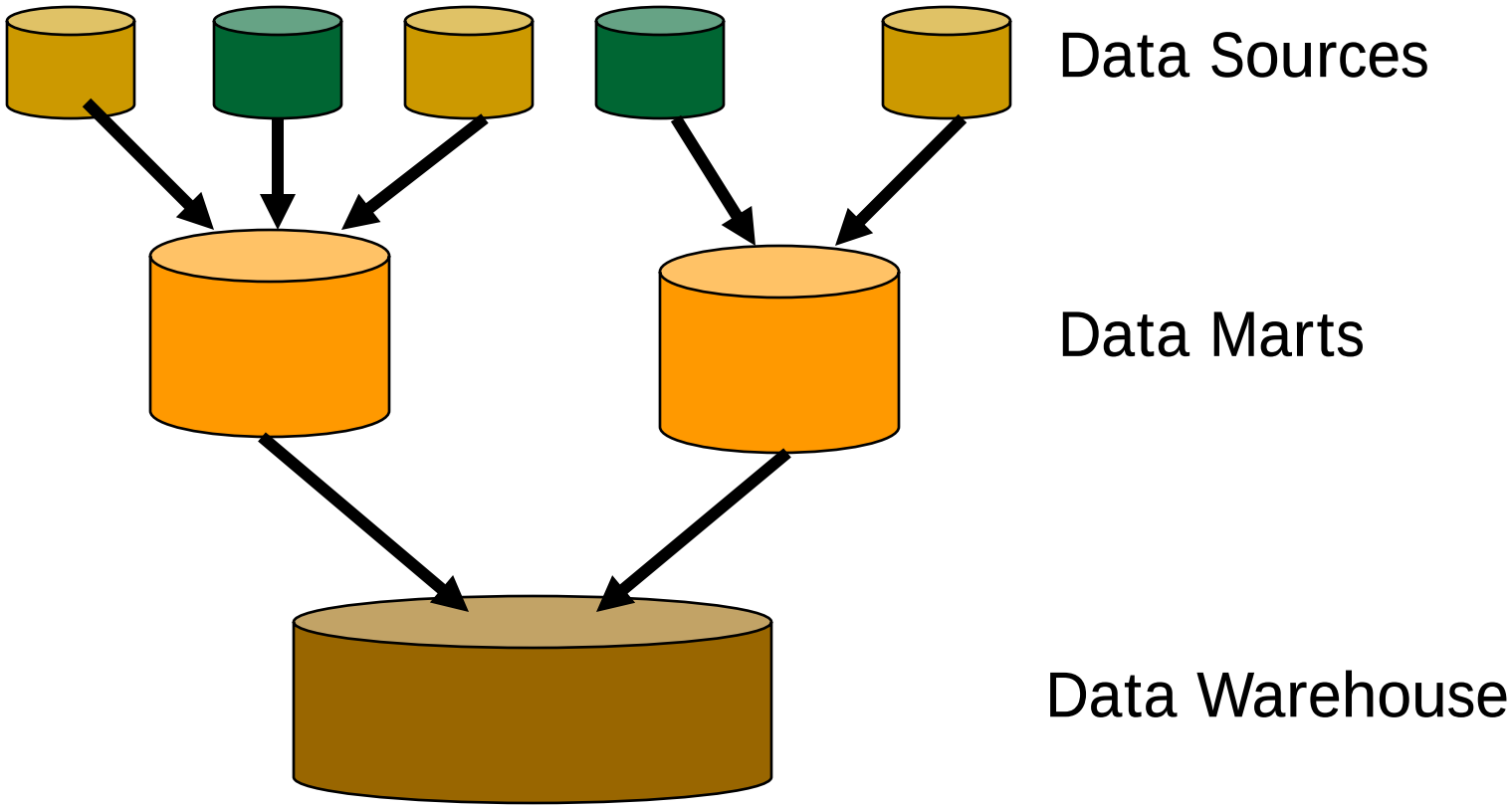
From the Data Warehouse to Data Marts (Bottom-up approach)



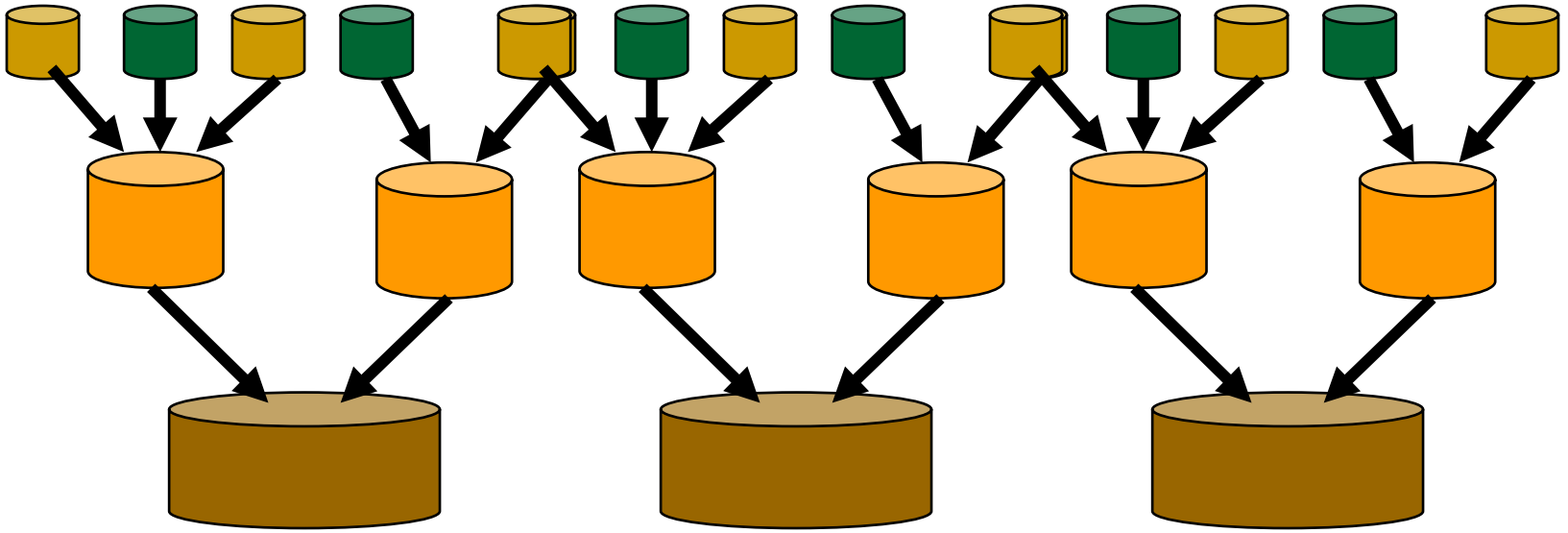
Problems of Bottom-up approach

- Data Warehouse level should be well defined according to the analysis which are really implemented in Data Marts level;
- To many unmanaged Data Marts which many caused duplication.

Data Mart Centric (Top-down approach)



Problems with Data Mart Centric Solution



If you end up creating multiple warehouses, integrating them is a problem

Let's go to the next …

数据仓库与数据挖掘
Data Warehouse & Data Mining

宋 杰
Song Jie
东北大学 软件学院
Software College, Northeastern University

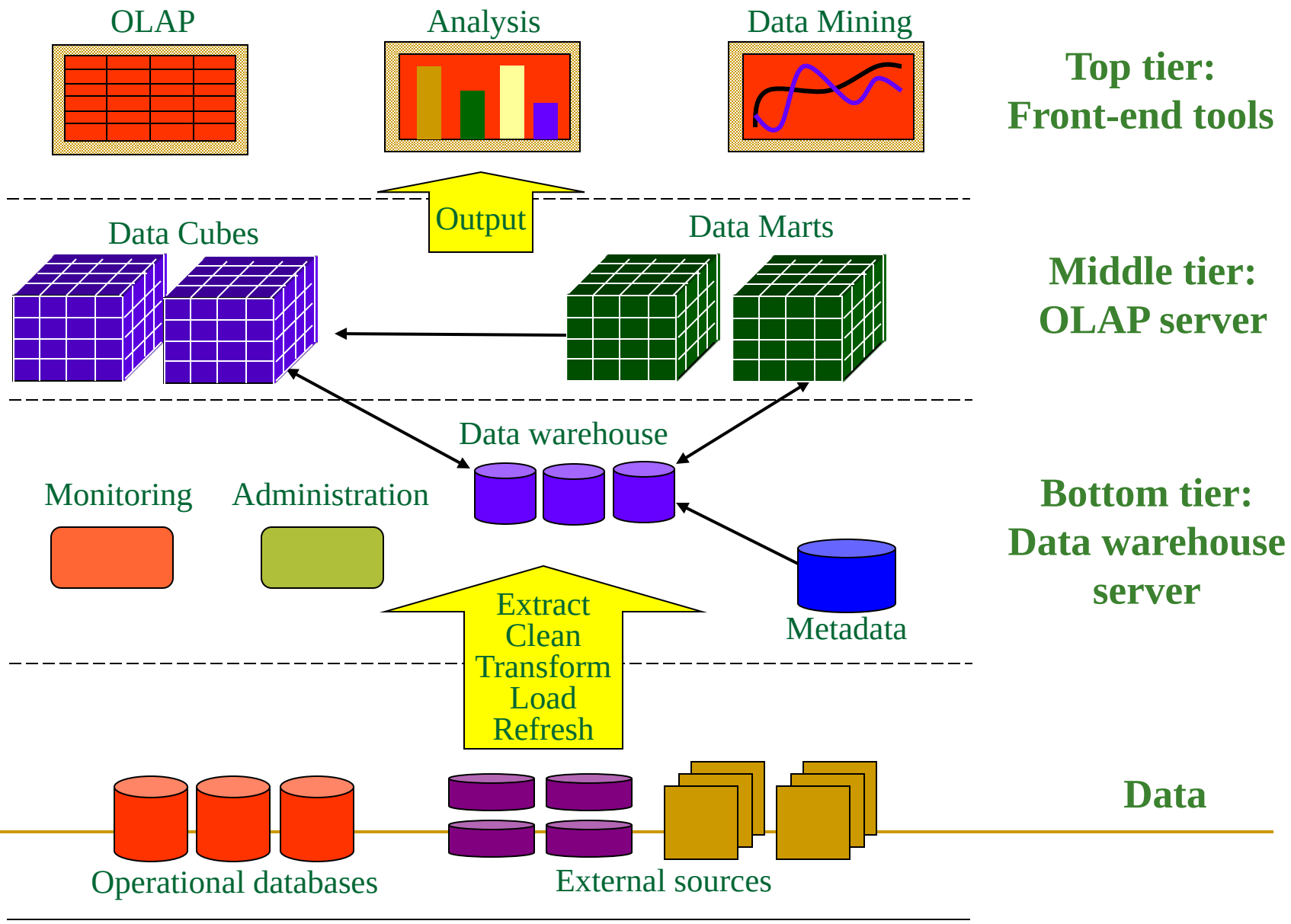
4. ETL

Cleaning the data before it is loaded

What is ETL

- **Extract, transform, and load** (**ETL**) is a process in database usage and especially in data warehousing that involves:
  - Extracting data from outside sources
  - Transforming it to fit operational needs (which can include quality levels)
  - Loading it into the end target (database or data warehouse)

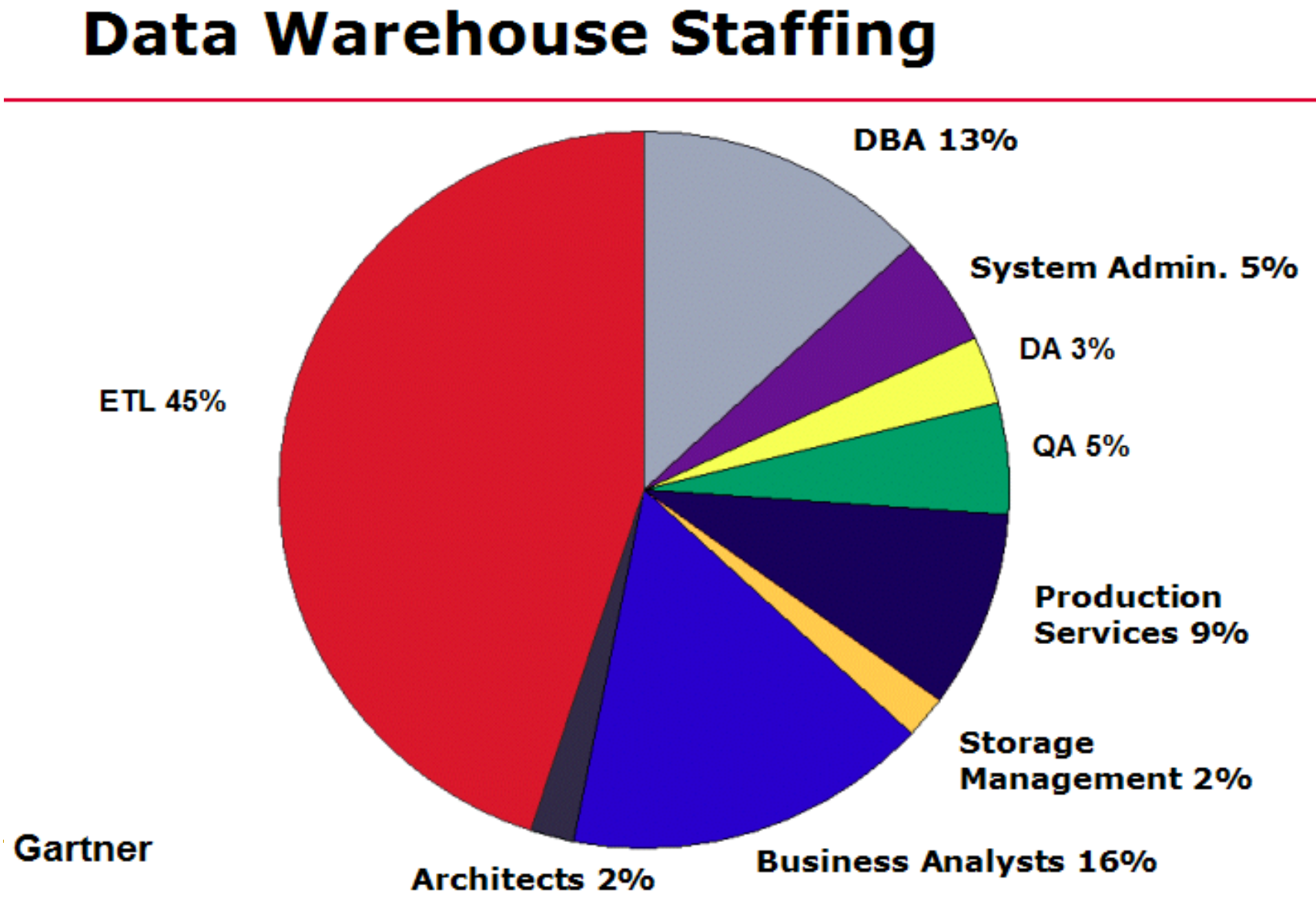
Multi-tiered Data Warehouse Architecture



What is ETL

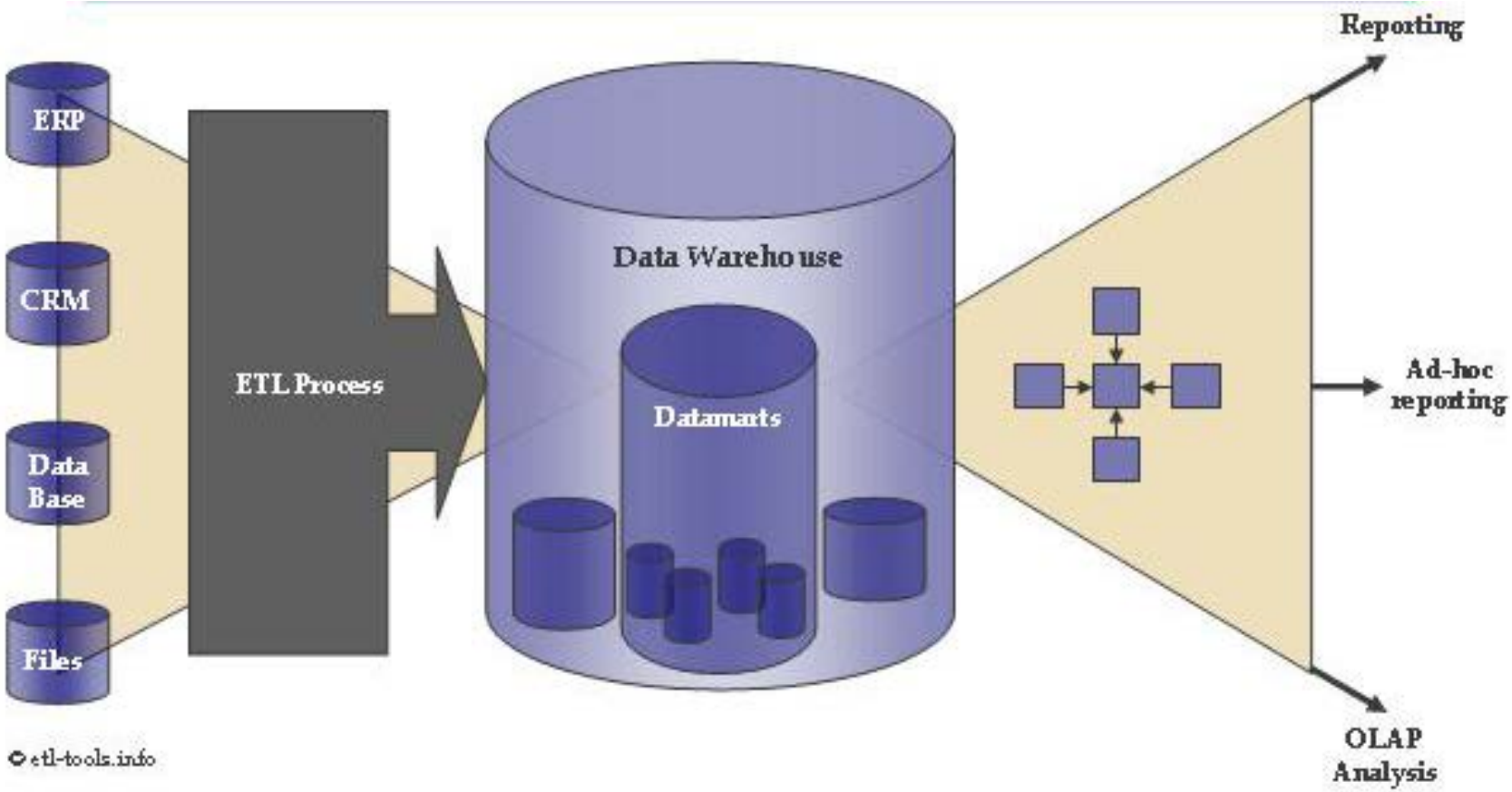
- Extraction Transformation Loading – ETL
- To get data out of the source and load it into the data warehouse – simply a process of copying data from one database to other
- Data is extracted from an OLTP database, transformed to match the data warehouse schema and loaded into the data warehouse database
- Many data warehouses also incorporate data from non-OLTP systems such as text files, legacy systems, and spreadsheets; such data also requires extraction, transformation, and loading
- When defining ETL for a data warehouse, it is important to think of ETL as a process, not a physical implementation

ETL Cost Analysis



ETL Overview

- ETL is often a complex combination of process and technology that consumes a significant portion of the data warehouse development efforts and requires the skills of business analysts, database designers, and application developers
- It is not a one time event as new data is added to the Data Warehouse periodically – monthly, daily, hourly
- Because ETL is an integral, ongoing, and recurring part of a data warehouse
  - Automated
  - Well documented
  - Easily changeable


Reporting
ERP
CRM
Data Base
Files
ETL Process
Data Warehouse
Datamarts
Ad-hoc reporting
OLAP Analysis
© etl-tools.info

Extraction

- Data needs to be taken from some data source so that it can be put into the Data Warehouse.
  - The integration of all of the disparate systems across the enterprise.
  - Data is extracted from heterogeneous data sources
  - Each data source has its distinct set of characteristics that need to be managed and integrated into the ETL system in order to effectively extract data.

Heterogeneous data sources

- Most data warehousing projects consolidate data from different source systems.
  - Relational databases
  - Flat files
  - Non-relational database
  - Fetching from outside sources such as through web spidering or screen-scraping.
- Goal of the extraction phase is to convert the data into a single format which is appropriate for transformation processing

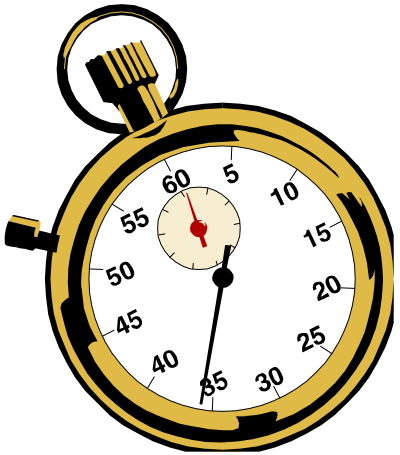
Logical Data Map

Target			Source			Transformation
Table Name	Column Name	Data Type	Table Name	Column Name	Data Type	

- Need to have a logical data map before the physical data can be transformed
- The logical data map describes the relationship between the extreme starting points and the extreme ending points of your ETL system usually presented in a table or spreadsheet

How To Detect Changes

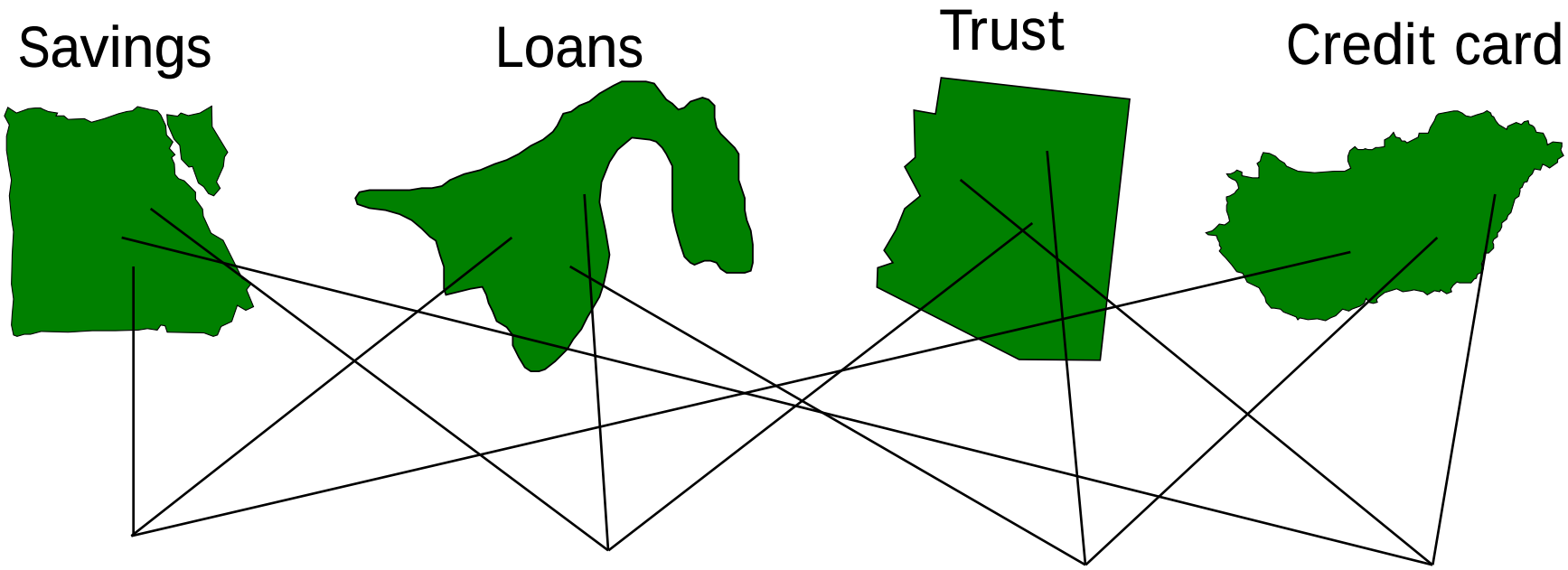
- Detect changes by:
  - In-time comparing with data in data warehouse;
  - Snapshot of id or timestamp;
  - Database triggers
  - Using regular transaction log to detect changes to source data;
  - Using metadata;
  - Using some natural attributes;
  - Application level monitoring
  - Doing it manually.



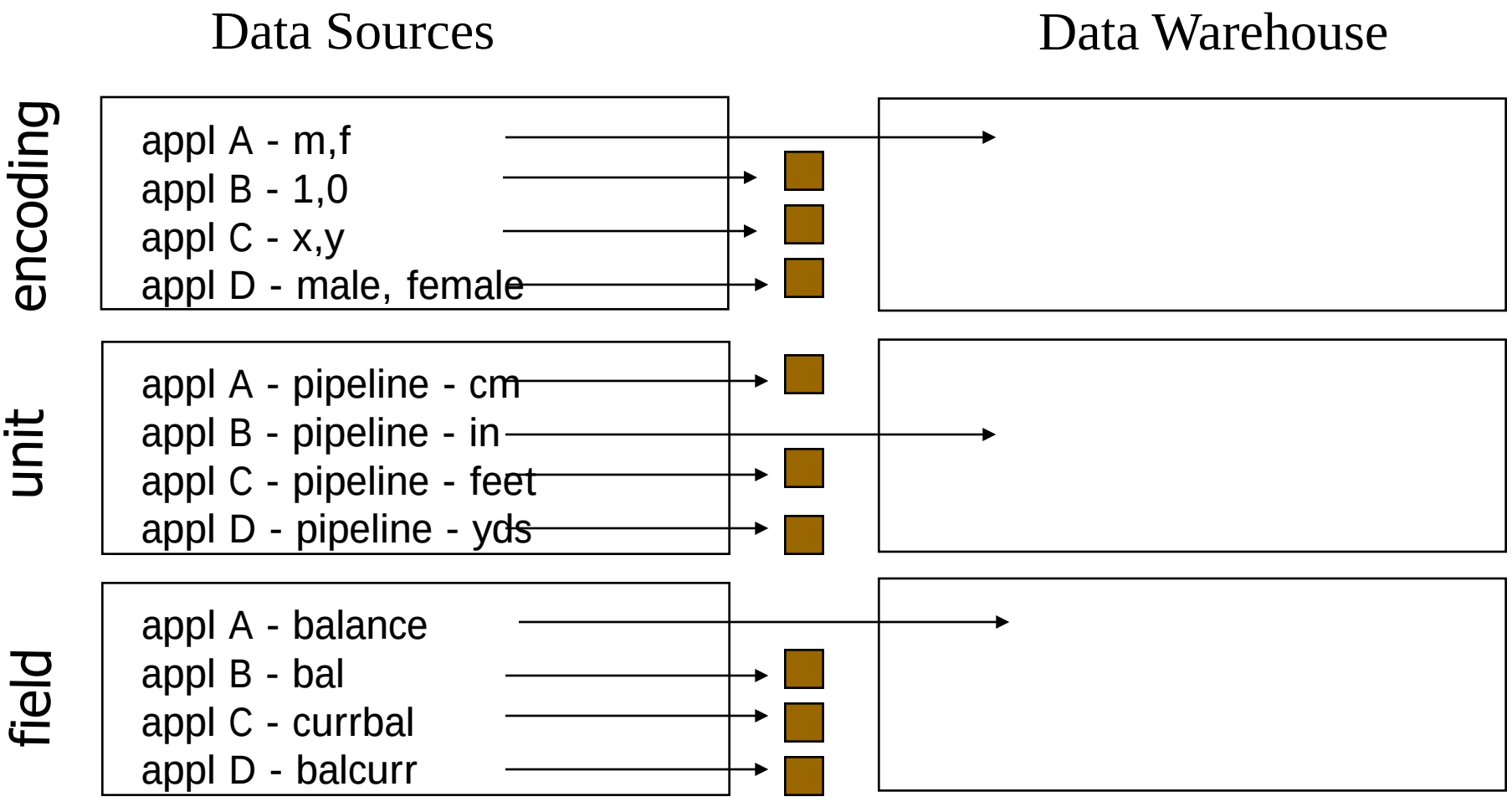
Transformation

- Three major steps of data transformation
  - Data Cleansing
  - Data Integration
  - Data Aggregation
  - Other Transformations
    - Replacement of codes
    - Process derived values
    - Process NULL value

Data Quality and Data Cleaning



Data Transformation Example

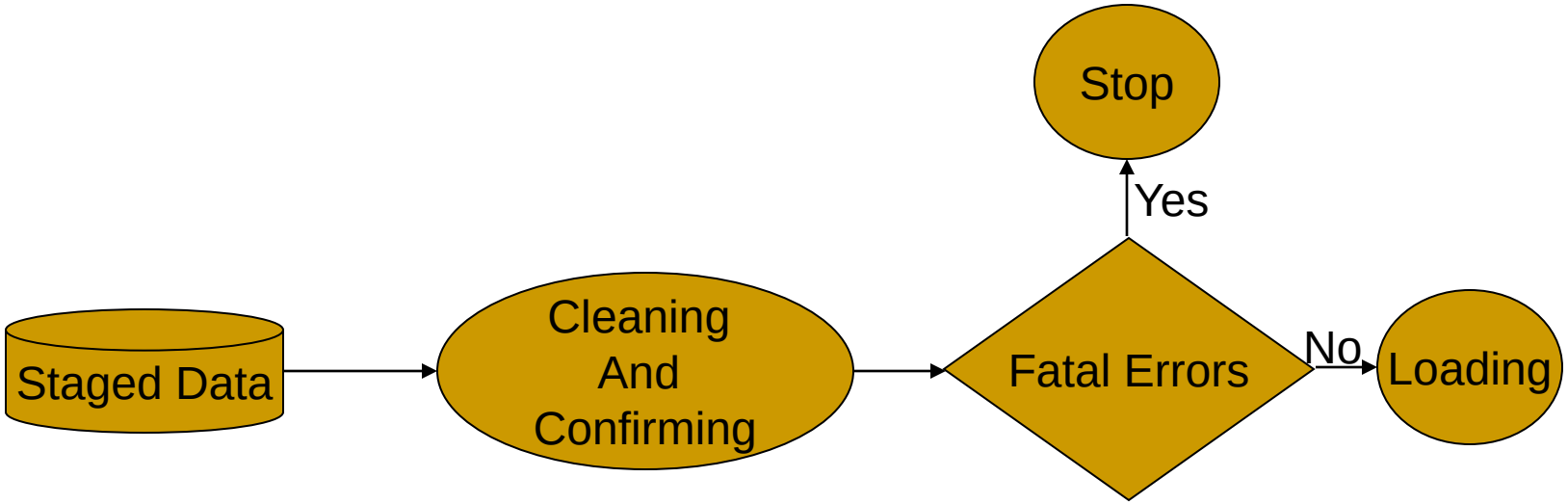


Data Cleansing

- Dirty Data
- Dummy Values
- Absence of Data
- Cryptic Data
- Contradicting Data
- Inconsistent Data
- Reused Primary Keys
- Non-unique Identifiers

- 脏数据
- 虚假数据
- 空缺数据
- 隐藏数据
- 矛盾数据
- 不一致数据
- 重复使用的主键
- 不唯一的标识符

Cleansing process



Data Integration

- Data integration involves combining data residing in different sources and providing users with a unified view of these data.

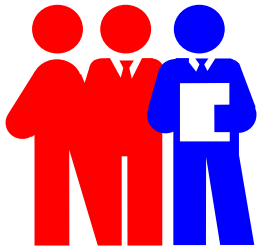
- Integrating data from several schema to an unify schema.

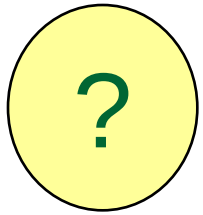
Two Major Problems of Data Integration

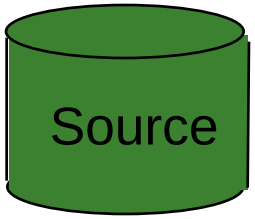
- Data that should be related but cannot be
  - May arise due to non-unique primary keys or more often, the absence of primary keys
- Data that is inadvertently(非故意的) related but should not be
  - Occurs when fields or records are reused for multiple purposes

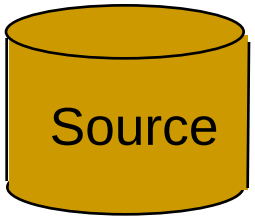
Data Integration

- Two Approaches:
  - Query-Driven (Lazy)
  - Data Warehouse (Eager)

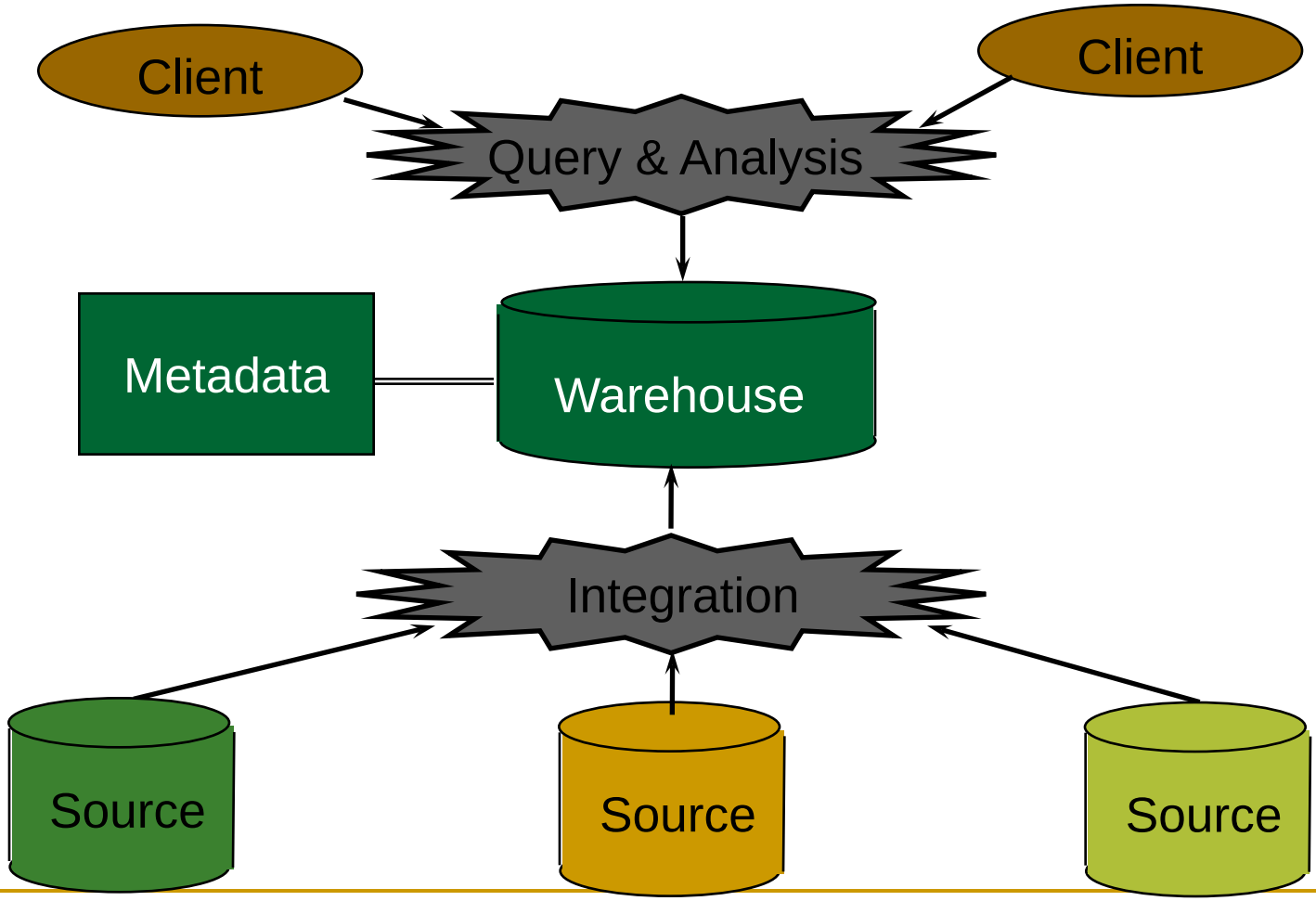




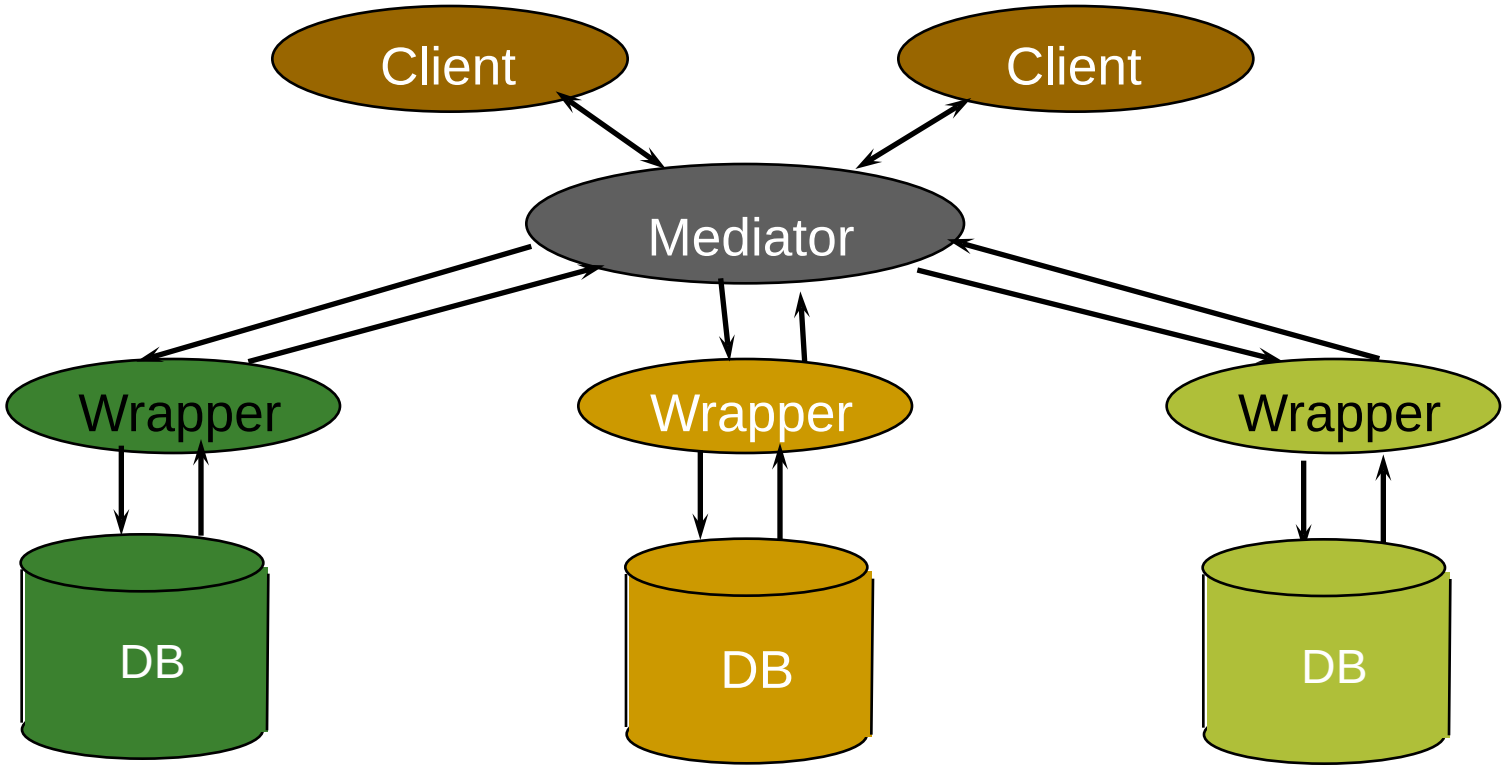




Warehouse Based Data Integration



Mediator based Query-Driven Data Integration



Advantages of Query-Driven

- No need to copy data
  - Less storage
  - No need to buy equipments
- More up-to-date data
- Query needs can be unknown at sources
- Only query interface needed at sources
- May be less leaking on sources

Advantages of Warehousing

- High query performance
- Queries not visible outside warehouse
- Local processing at sources unaffected
- Can operate when sources unavailable
- Can query data not stored in a DBMS
- Extra information at warehouse
  - Modify, summarize, aggregates
  - Add historical information

Data Transform process

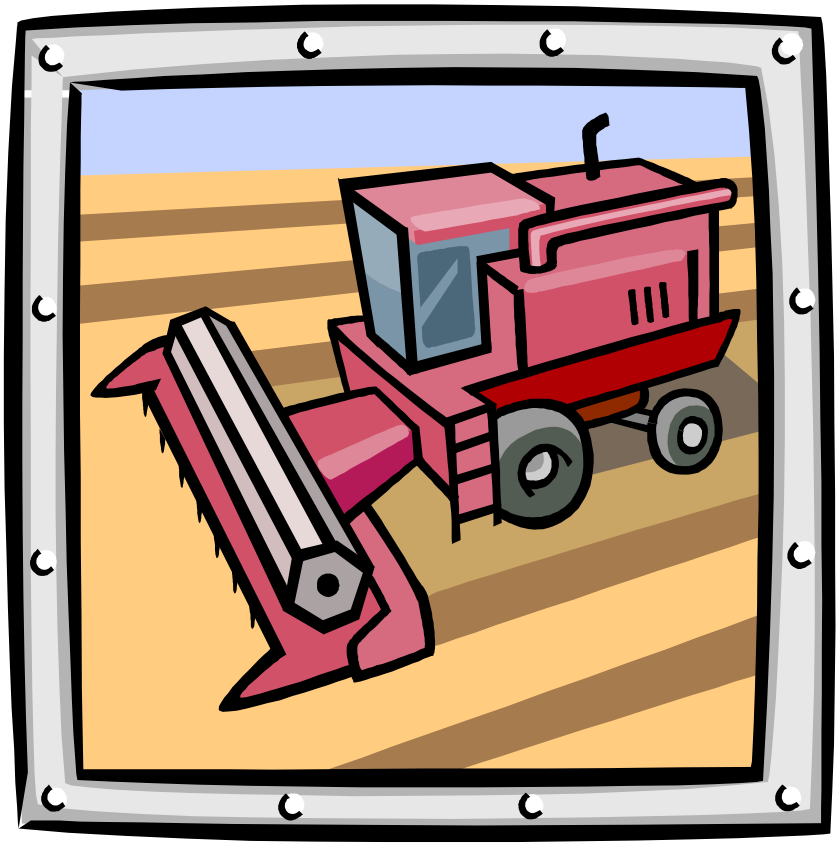
1. Selecting only certain columns to load
2. Translating coded values
3. Encoding free-form values
4. Deriving a new calculated value
5. Filtering
6. Sorting
7. Joining data from multiple sources (lookup and merge)
8. Aggregation
9. Generating surrogate-key values
10. Transposing or pivoting (turning multiple columns into multiple rows or vice versa)
11. Splitting a column into multiple columns
12. Applying any form of simple or complex data validation.

Loading

- The load phase loads the data into the end target, usually the data warehouse.
- Data warehouses may overwrite existing information with cumulative(渐增的) information, frequently updating extract data is done on daily, weekly or monthly basis.

Loading

- The loading process can be broken down into two different types:
  - Initial Load
  - Continuous Load
    (Refresh)
    (Increased Load)
    (Loading over time)



Initial Load

- Consists of populating tables in warehouse schema and verifying data readiness
- Examples:
  - DTS – Data Transformation Services
  - Bcp utility – Batch copy
  - SQL*Loader
  - Native Database Languages (T-SQL, PL/SQL, etc.)

Refresh

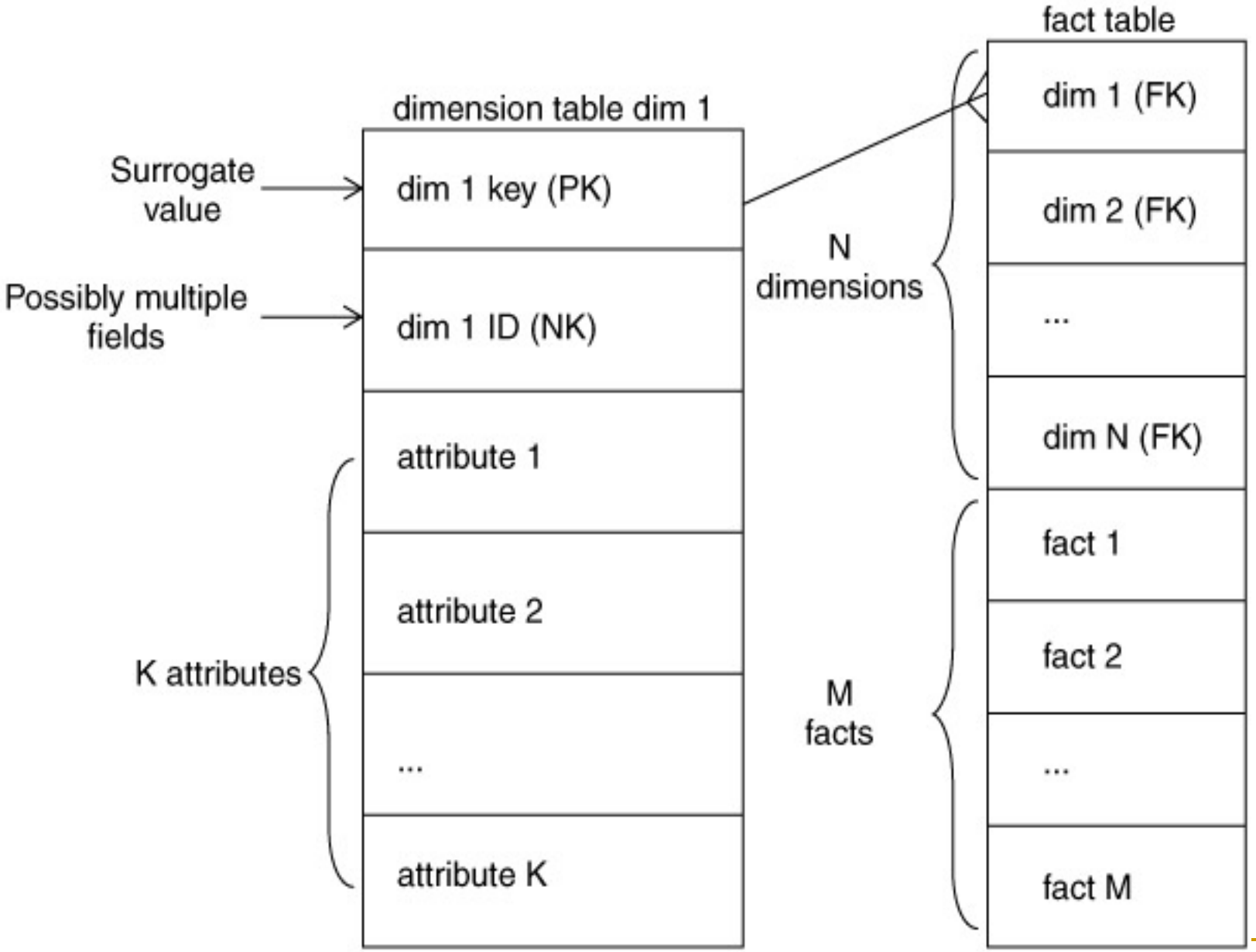
- Spread updates on source data to the warehouse
- Must be scheduled and processed in a specific order to maintain integrity, completeness
- Should be the most carefully planned step in data warehousing or can lead to:
  - Error duplication
  - Exaggeration of inconsistencies in data
- Issues:
  - When to refresh
  - How to refresh: refresh techniques

When to Refresh?

- Periodically (e.g., every night, every week)
- After significant events
- On every update
- When warehouse data require current data
- Refresh policy set by administrator based on user needs and traffic
- Possibly different policies for different sources

Loading Dimensions

- Physically built to have the minimal sets of components
- The primary key is a single field containing meaningless unique integer – Surrogate Keys
  - There may be a natural key in a dimension
  - The DW owns surrogate keys and never allows any other entity to assign them
- All attributes in a dimension must take on a single value in the presence of a dimension primary key
- Should possess one or more other fields that compose the natural key of the dimension


fact table
dimension table dim 1
Surrogate value
dim 1 key (PK)
Possibly multiple fields
dim 1 ID (NK)
attribute 1
attribute 2
...
attribute K
K attributes
N dimensions
dim 1 (FK)
dim 2 (FK)
...
dim N (FK)
M facts
fact 1
fact 2
...
fact M

Slowly Changing Dimensions (SCD)

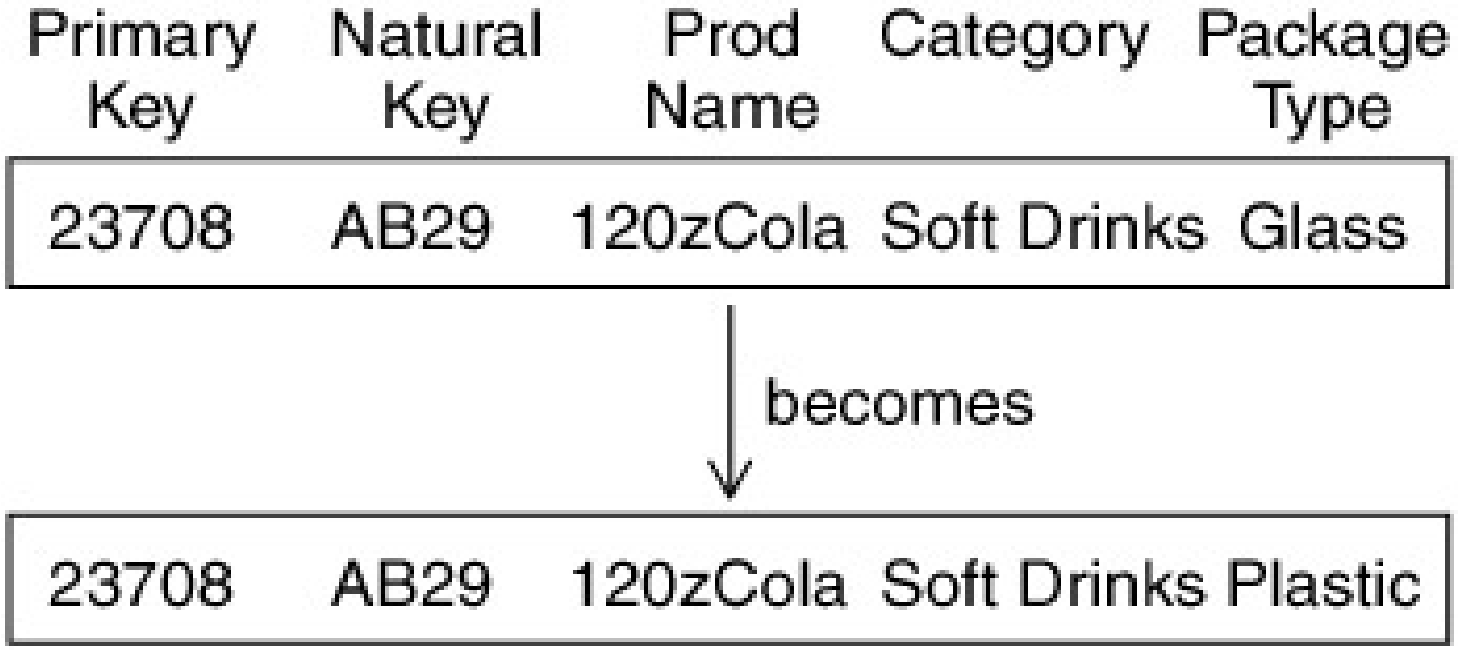
- The data loading module consists of all the steps required to administer slowly changing dimensions (SCD) and write the dimension to disk as a physical table in the proper dimensional format with correct primary keys, correct natural keys, and final descriptive attributes.
- Creating and assigning the surrogate keys occur in this module.
- The table is definitely staged, since it is the object to be loaded into the presentation system of the data warehouse.

Refresh SCD

- When DW receives notification that an existing row in dimension has changed
- It gives out three types of responses
  - Type 1
  - Type 2
  - Type 3

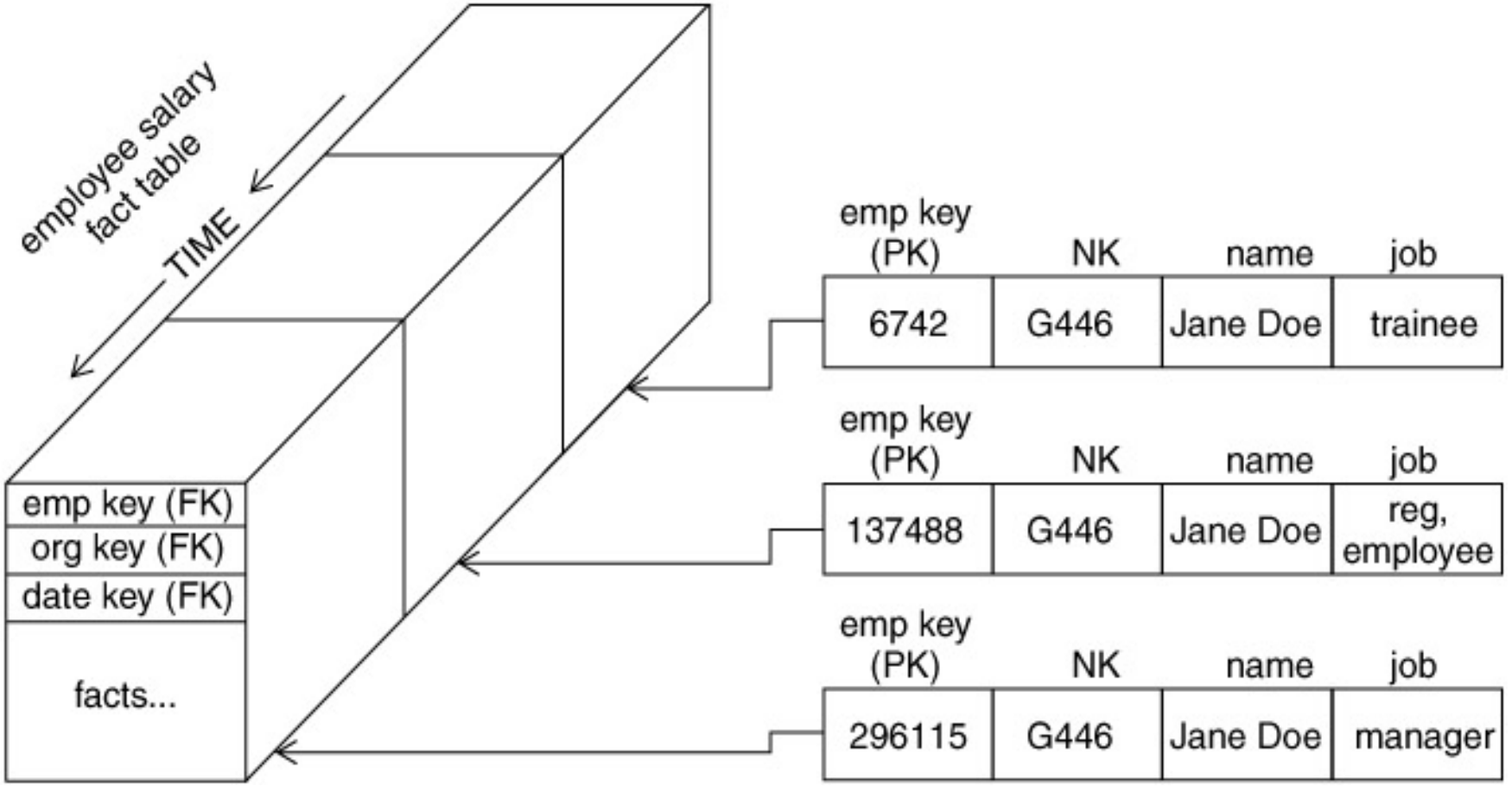
Type 1

- The dimension row is simply updated to match the current state of the source system. The warehouse does not capture history.



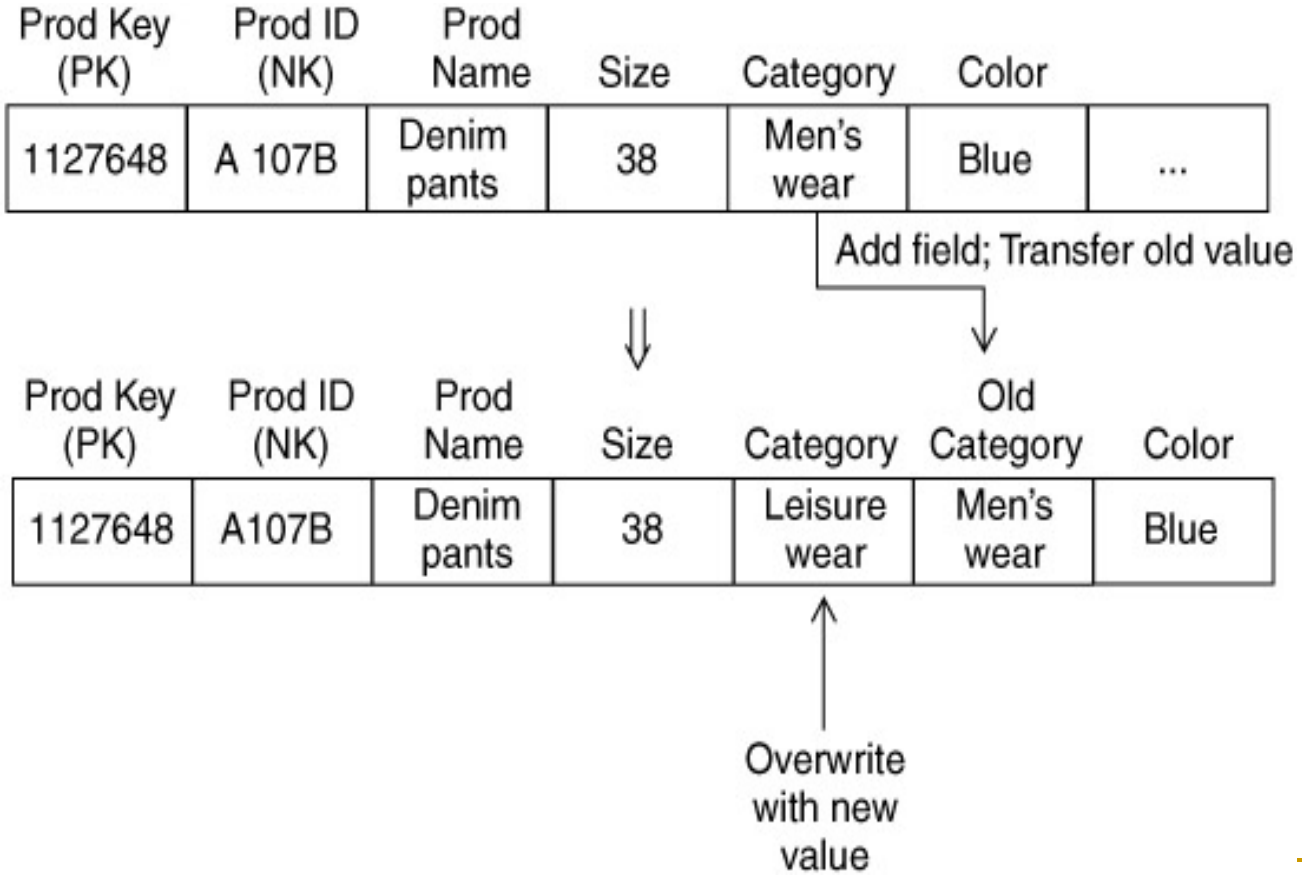
Type 2

- A new dimension row is added with the new state of the source system. A new surrogate key is assigned.



Type 3

- The old value is logged by adding a new attribute



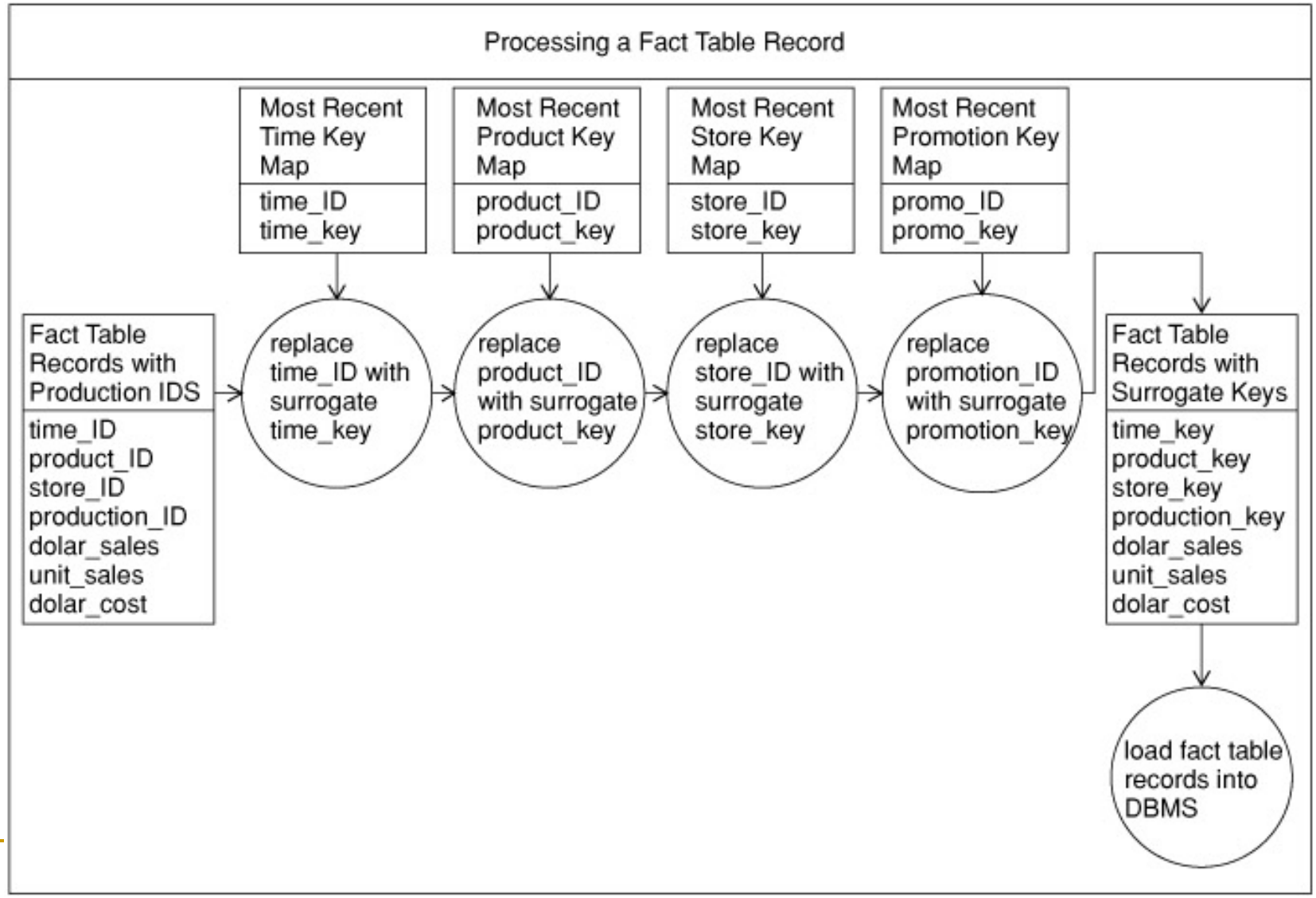
Loading Facts

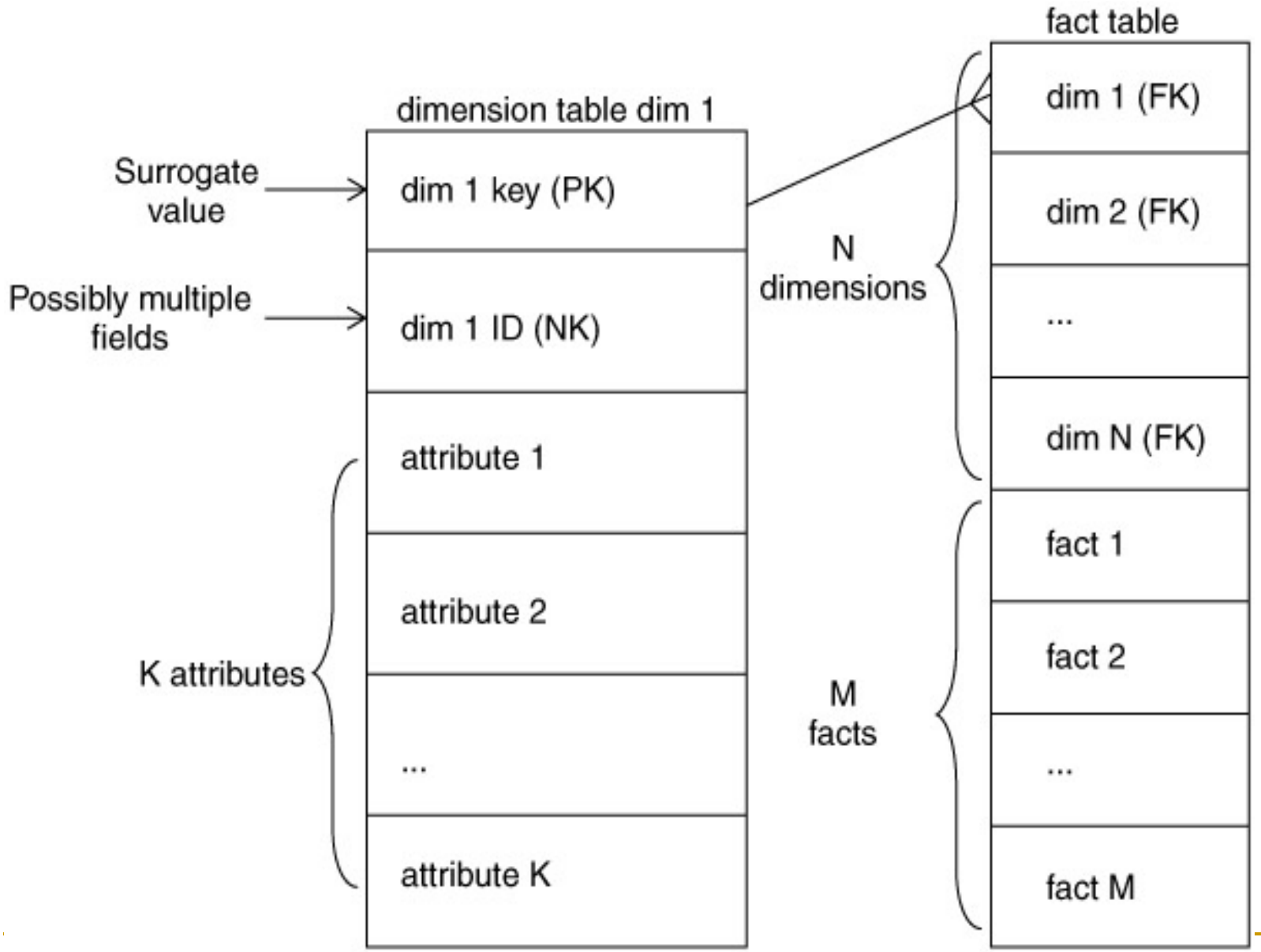
- Fact tables hold the measurements of an enterprise.
- The relationship between fact tables and measurements is extremely simple.
  - If a measurement exists, it can be modeled as a fact table row.
  - If a fact table row exists, it is a measurement

Key Building Process - Facts

- When building a fact table, the final ETL step is converting the natural keys in the new input records into the correct surrogate keys
- ETL maintains a special surrogate key lookup table for each dimension.
- All of the required lookup tables should be pinned in memory so that they can be randomly accessed as each incoming fact record presents its natural keys.

Key Building Process




fact table
dim 1 (FK)
dim 2 (FK)
...
dim N (FK)
fact 1
fact 2
...
fact M
N
dimensions
M
facts
dimension table dim 1
dim 1 key (PK)
dim 1 ID (NK)
attribute 1
attribute 2
...
attribute K
Surrogate
value
Possibly multiple
fields
K attributes

Load Techniques

- Use SQL to append or insert new data
  - Record at a time interface
  - Will lead to random disk I/O's
- Use batch load utility

Load Approach

- Online versus Offline loads
- Incremental versus Full loads
- Buy versus Build

Vendor ETL tools

- DataStage
- Informatica
- SAS ETL Server
- Information Builder's Data Migrator/ETL Manager tool suite
- Sunopsis

Open sources ETL

- Apatar
- CloverETL
- Flat File Checker
- Jitterbit 2.0
- Pentaho Data Integration (now[update] included in OpenOffice Base)
- RapidMiner
- Scriptella
- Talend Open Studio

Buy vs Build

- Performance
- Supported sources
- Standardization
- Reusability
- Extension
- Built-in features
- Studied capabilities
- Documenting
- Price and cost

Performance of ETL

- ETL vendors benchmark their record-systems at multiple TB (terabytes) per hour (or ~1 GB per second) using powerful servers with multiple CPUs, multiple hard drives, multiple gigabit-network connections, and lots of memory.

Performance issues

- Direct Path Extract method or bulk unload whenever is possible (instead of querying the database);
- Most of the transformation processing outside of the database;
- Use bulk load operations whenever possible;
- Partition original tables and indexes, try to keep partitions similar in size;
- Do all validation in the ETL layer before the load, disable integrity checking in the target database tables during the load.

Performance issues

- Disable triggers in the target database tables during the load.
- Generate IDs in the ETL layer, not in the database;
- Drop the indexes and partition of target table before the load;
- Use parallel bulk load when possible
  - Note: attempt to do parallel loads into the same table (partition) usually causes locks — if not on the data rows, then on indexes.

Four-layered approach for ETL architecture design

- **Functional layer**: Core functional ETL processing (extract, transform, and load).
- **Operational management layer**: Job-stream definition and management, parameters, scheduling, monitoring, communication and alerting.
- **Audit, balance and control (ABC) layer**: Job-execution statistics, balancing and controls, rejects- and error-handling, codes management.
- **Utility layer**: Common components supporting all other layers.

Qualities of a good ETL architecture design

- Performance
- Scalable
- Migratable
- Recoverable
- Operable
  - completion-codes for phases, re-running from checkpoints, etc.
- Auditable
  - Business requirements
  - Technical troubleshooting

Data Deduplication

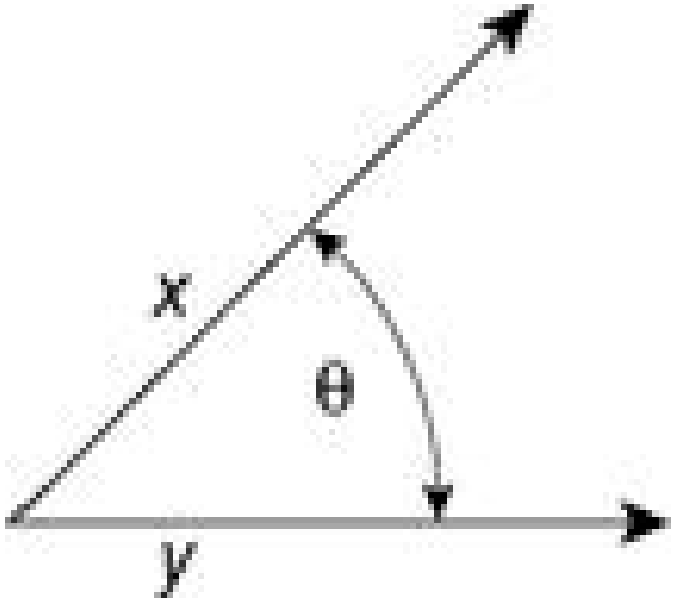
- **Data deduplication** is a specific form of compression where redundant data is eliminated, data is deleted, leaving only one copy of the data to be stored.
- **Record linkage** (RL) refers to the task of finding entries that refer to the same entity in two or more records. A data set that has undergone record linkage is said to be linked.

Data Deduplication

- Similarity
- Initial and Incremental
- Attribute Filtering
- Relationships of Tables

Similarity (Approximate)

- Dissimilarity Measure (值越大越相异)
  - Euclidean Distance (欧几里得距离)
- Similarity (值越大越相似)
  - SMC: Simple Matching Coefficient (简单匹配系数)
  - Jaccard Coefficient
  - Cosine Similarity



Jaccard Coefficient

- Given two objects, *A* and *B*, each with *n* binary attributes, the Jaccard coefficient is a useful measure of the overlap that *A* and *B* share with their attributes. Each attribute of *A* and *B* can either be 0 or 1. The total number of each combination of attributes for both *A* and *B* are specified as follows:
  - *M*11 represents the total number of attributes where *A* and *B* both have a value of 1.
  - *M*01 represents the total number of attributes where the attribute of *A* is 0 and the attribute of *B* is 1.
  - *M*10 represents the total number of attributes where the attribute of *A* is 1 and the attribute of *B* is 0.
  - *M*00 represents the total number of attributes where *A* and *B* both have a value of 0.
- Each attribute must fall into one of these four categories, meaning that
  - $M11 + M01 + M10 + M00 = n$.
- The Jaccard similarity coefficient, *J*, is given as
  - $J = M11 / (M01 + M10 + M11)$
  - $SMC = (M00 + M11) / (M01 + M10 + M00 + M11)$

Jaccard Coefficient Example

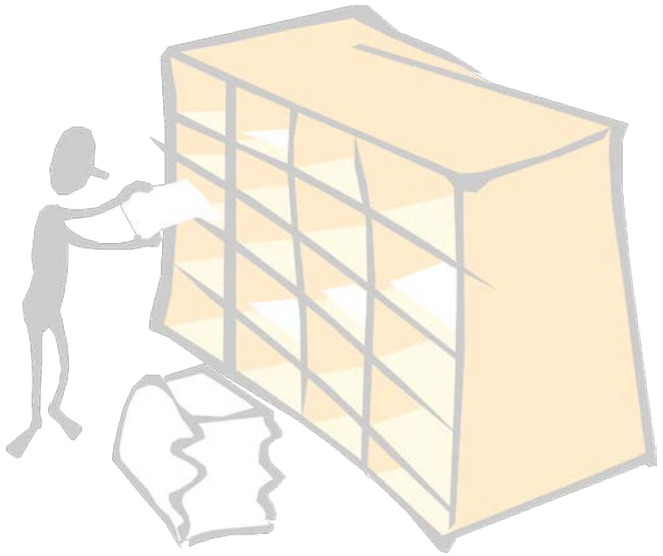
- **x** = (1, 0, 0, 0, 0, 0, 0, 0, 0, 0)
- **y** = (0, 0, 0, 0, 0, 0, 1, 0, 0, 1)
- $M01 = 2$ x=0 & y=1
- $M10 = 1$ x=1 & y=0
- $M00 = 7$ x=0 & y=0
- $M11 = 0$ x=1 & y=1
- $J = M11 / (M01 + M10 + M11)$
  $= 0/(2+1+0) = 0$
- $SMC = (M00 + M11) / (M01 + M10 + M00 + M11)$
  $= (7 + 0) / (2+1+7 + 0) = 0.7$

Let's go to the next …

数据仓库与数据挖掘
Data Warehouse & Data Mining

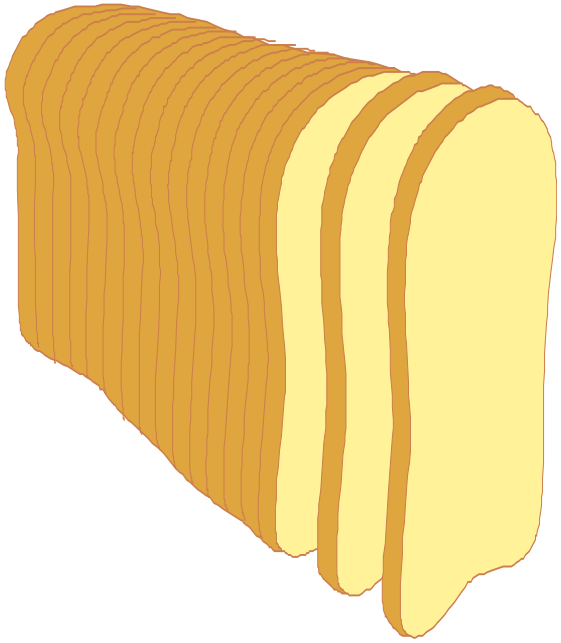
宋 杰
Song Jie
东北大学 软件学院
Software College, Northeastern University

5 . Partition and Index



Partitioning

- Breaking data into several physical units that can be handled separately
- Not a question of *whether* to do it but *how* to do it in data warehouses
- Granularity and partitioning are key to effective implementation of a warehouse



Partitioning

- A partition is a division of a logical database or its constituting elements into distinct independent parts.
- Database partitioning is normally done for manageability, performance or availability reasons.
- Horizontal partitioning
  - Range partitioning
  - List partitioning
  - Hash partitioning
- Vertical partitioning
- Composite partitioning

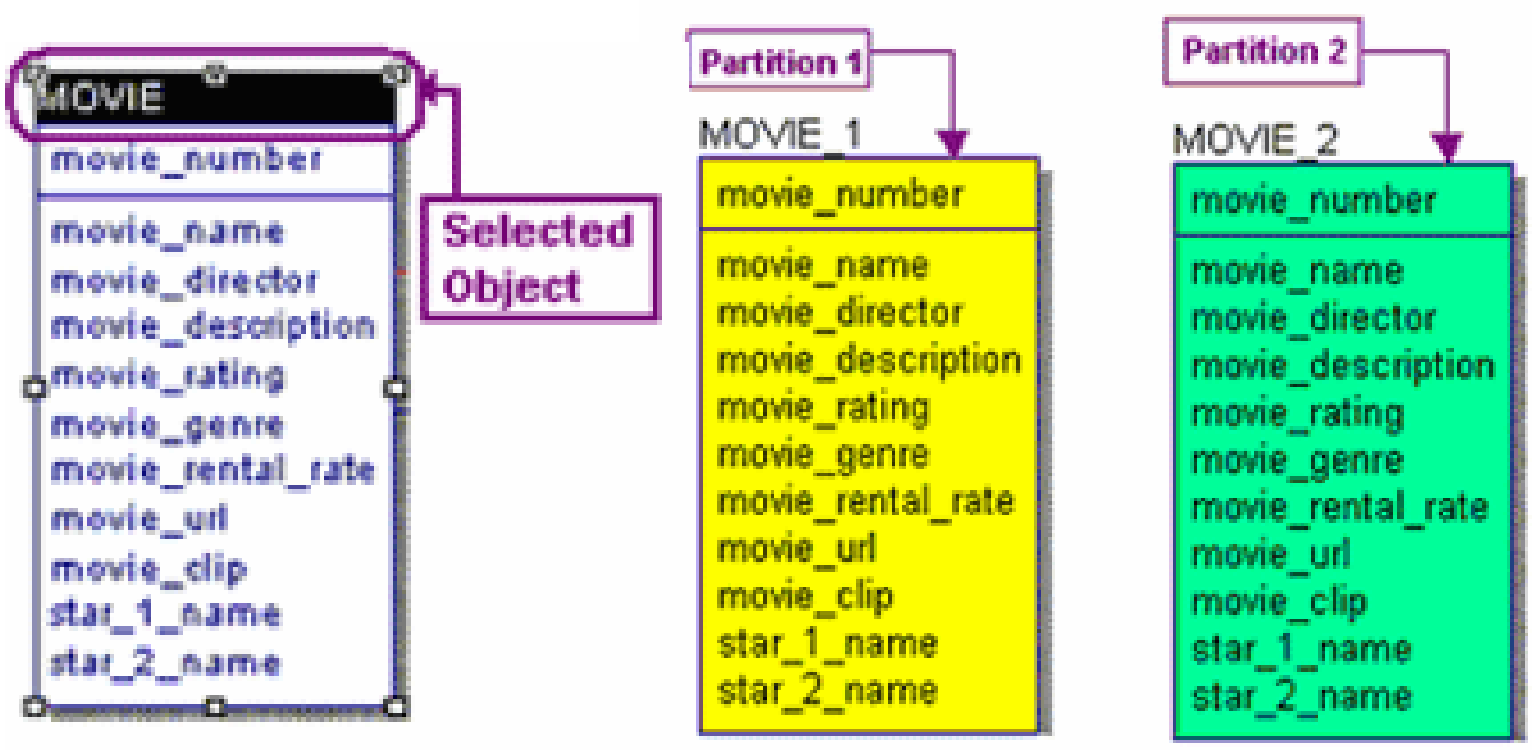
Why Partition?

- Flexibility in managing data
- Smaller physical units allow
  - Easy restructuring
  - Free indexing
  - Sequential scans if needed
  - Easy reorganization
  - Easy recovery
  - Easy monitoring

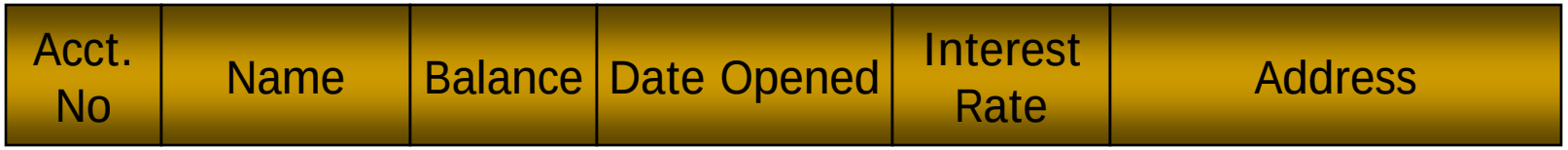
Criterion for Partitioning

- Typically partitioned by
  - Date
  - Geography
  - Organizational unit
  - Other business rule

Horizontal partitioning

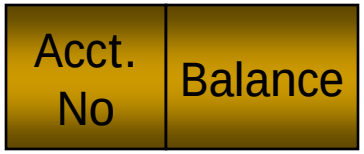


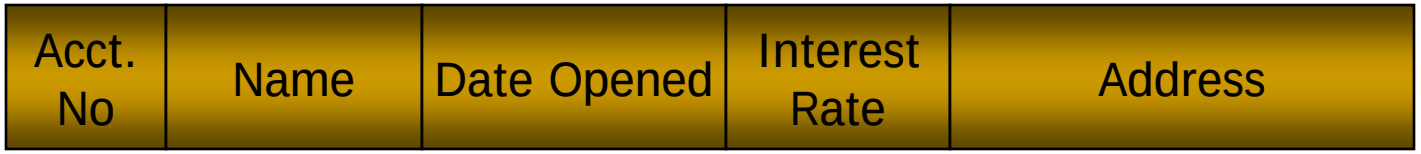
Vertical Partitioning



Frequently accessed

Rarely accessed



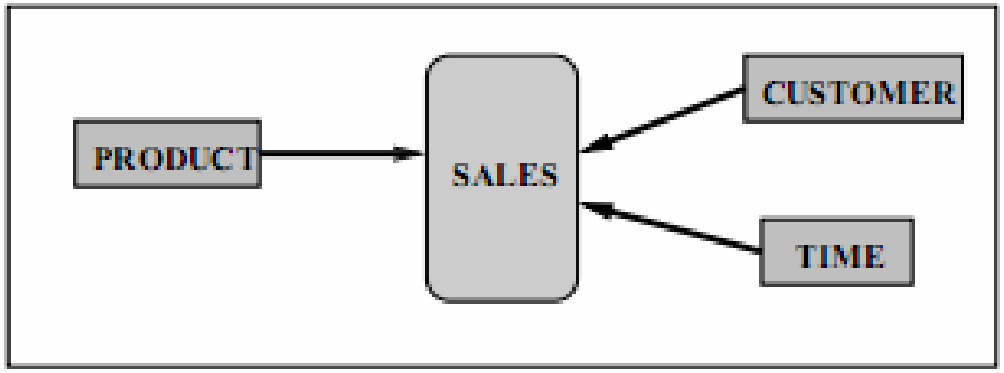


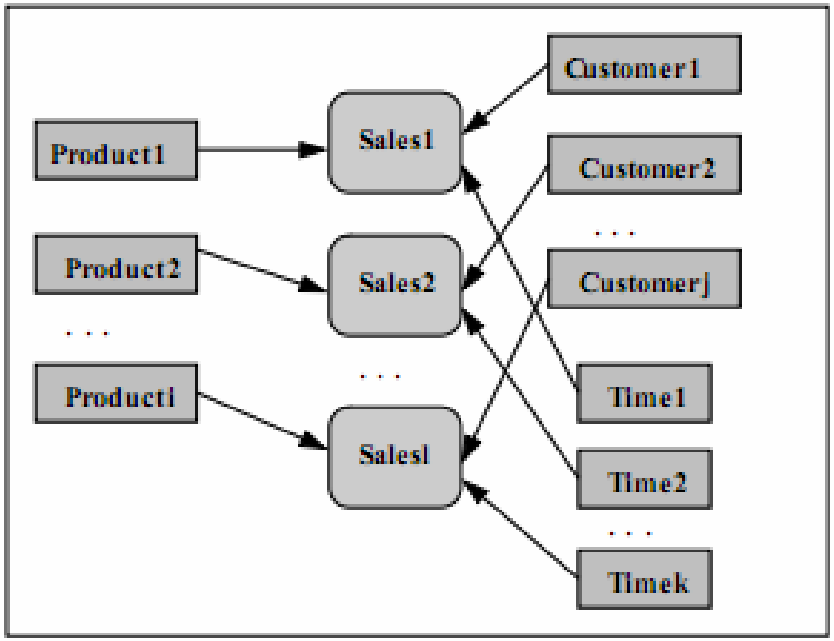
Smaller table
and so less I/O

Derived horizontal partition

- Primary HP of a relation is performed using attributes defined on that relation.
- Derived HP is the fragmentation of a relation using attribute(s) defined on another relation(s).
- Most data warehouse apply primary HP in dimension table and meanwhile the derived HP in fact table.

Partitioning on Star Schema





Managing Partitions

- Partitions allow a table (and its indexes) to be physically divided into *mini_tables* for administrative purposes and to improve query performance
- The most common partitioning strategy on fact tables is to partition the table by the date key. Because the date dimension is preloaded and static, you know exactly what the surrogate keys are
- Need to partition the fact table on the key that joins to the date dimension for the optimizer to recognize the constraint.
- The ETL team must be advised of any table partitions that need to be maintained.

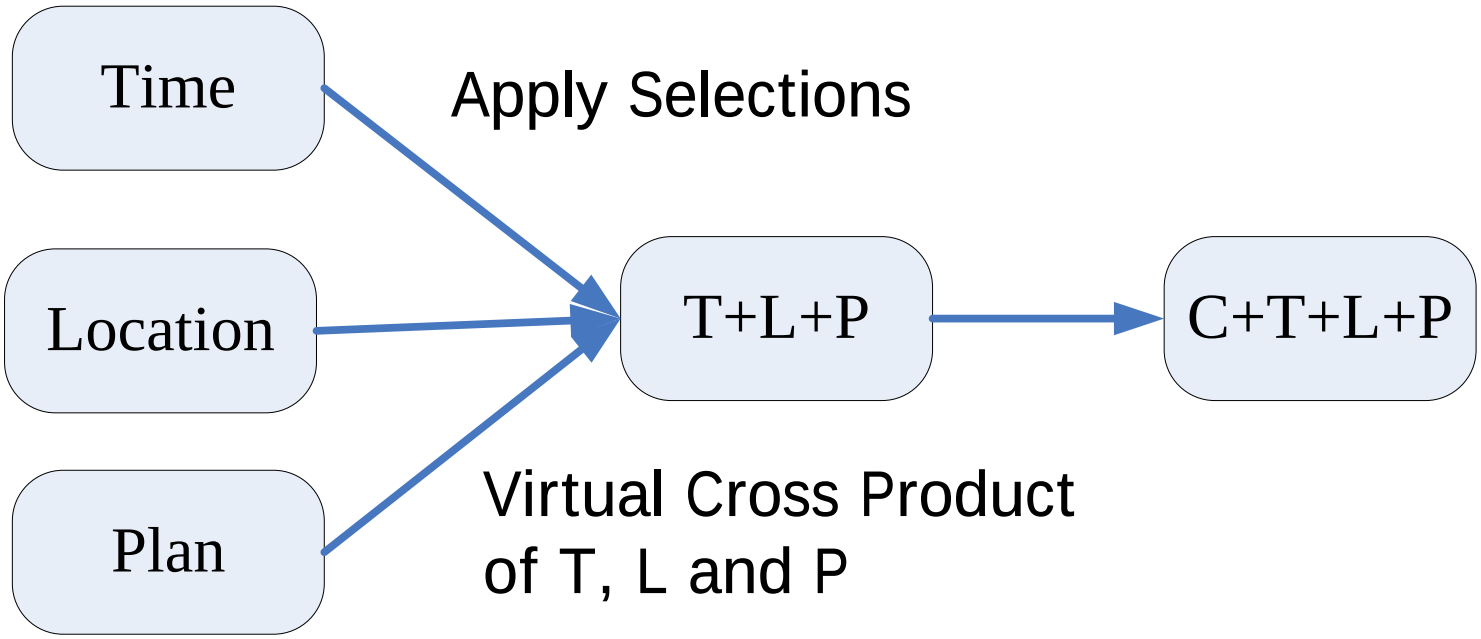
Indexing Techniques

- Hash index
- B+ trees index
- Bitmap index

Join Indexes

- A join index between a fact table and a dimension table correlates a dimension tuple with the fact tuples that have the same value on the common dimensional attribute
  - e.g., a join index on *city* dimension of *calls* fact table
  - CREATE BITMAP INDEX *city_calls_join_index*
    ON *calls*
    FROM *city* , *calls*
    WHERE *city.id* = *calls.city_id*;

Star Join Indexes



Managing Indexes

- Performance killers at load time
- Drop all indexes in pre-load time
- Segregate updates from inserts
- Load updates
- Rebuild indexes

ETL & Partitioning & Indexing

- Partitioning & Indexing before or after ETL?
- Local Indexes and Global Indexes?
- The costs of indexes and partitions?
- Sometime indexes and partitions are useless?

Let's go to the next …

数据仓库与数据挖掘
Data Warehouse & Data Mining

宋 杰
Song Jie
东北大学 软件学院
Software College, Northeastern University

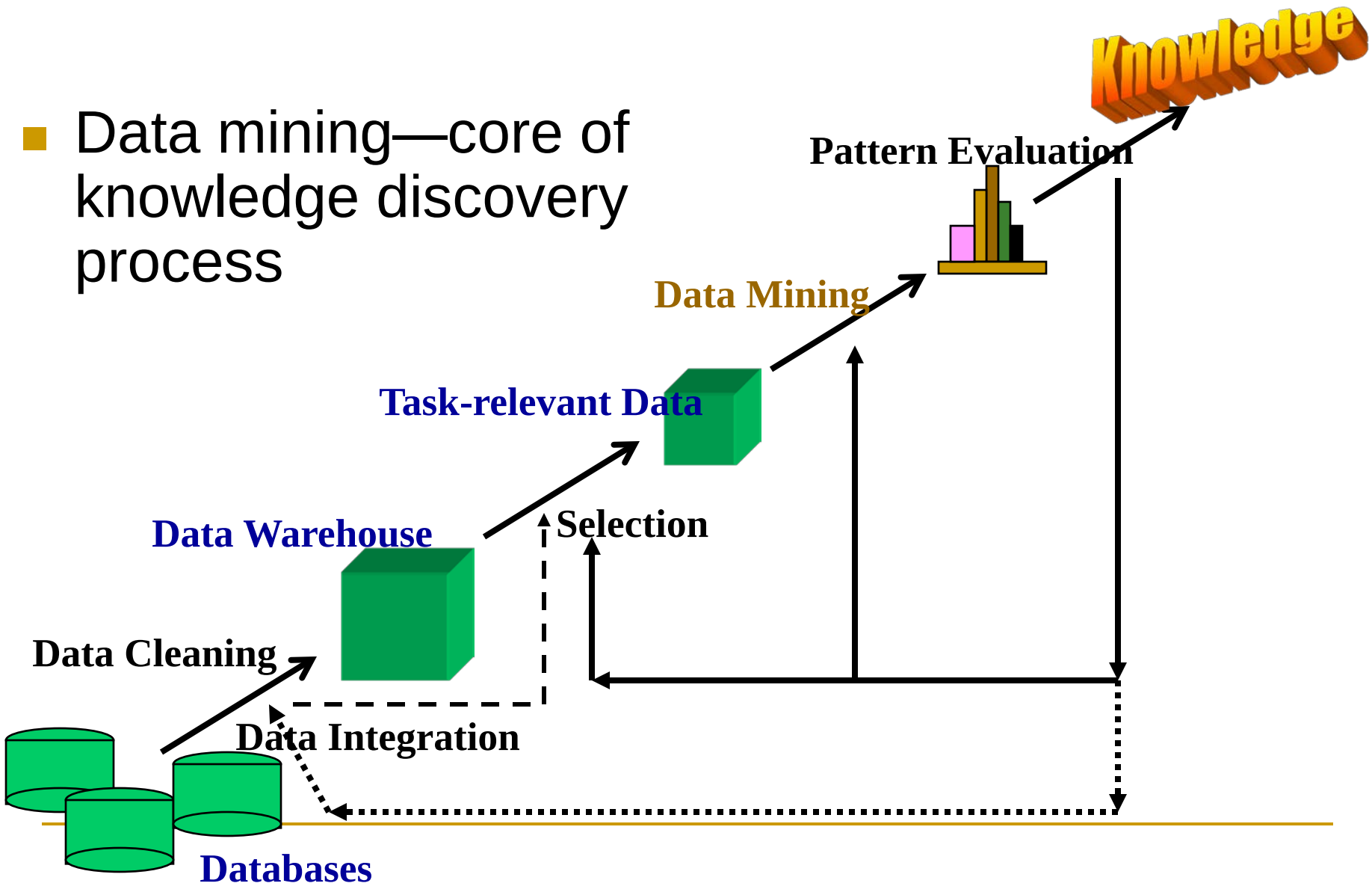
7. Data Mining (part one)

Summarization Relevance

Correlation Association

Knowledge Discovery (KDD) Process

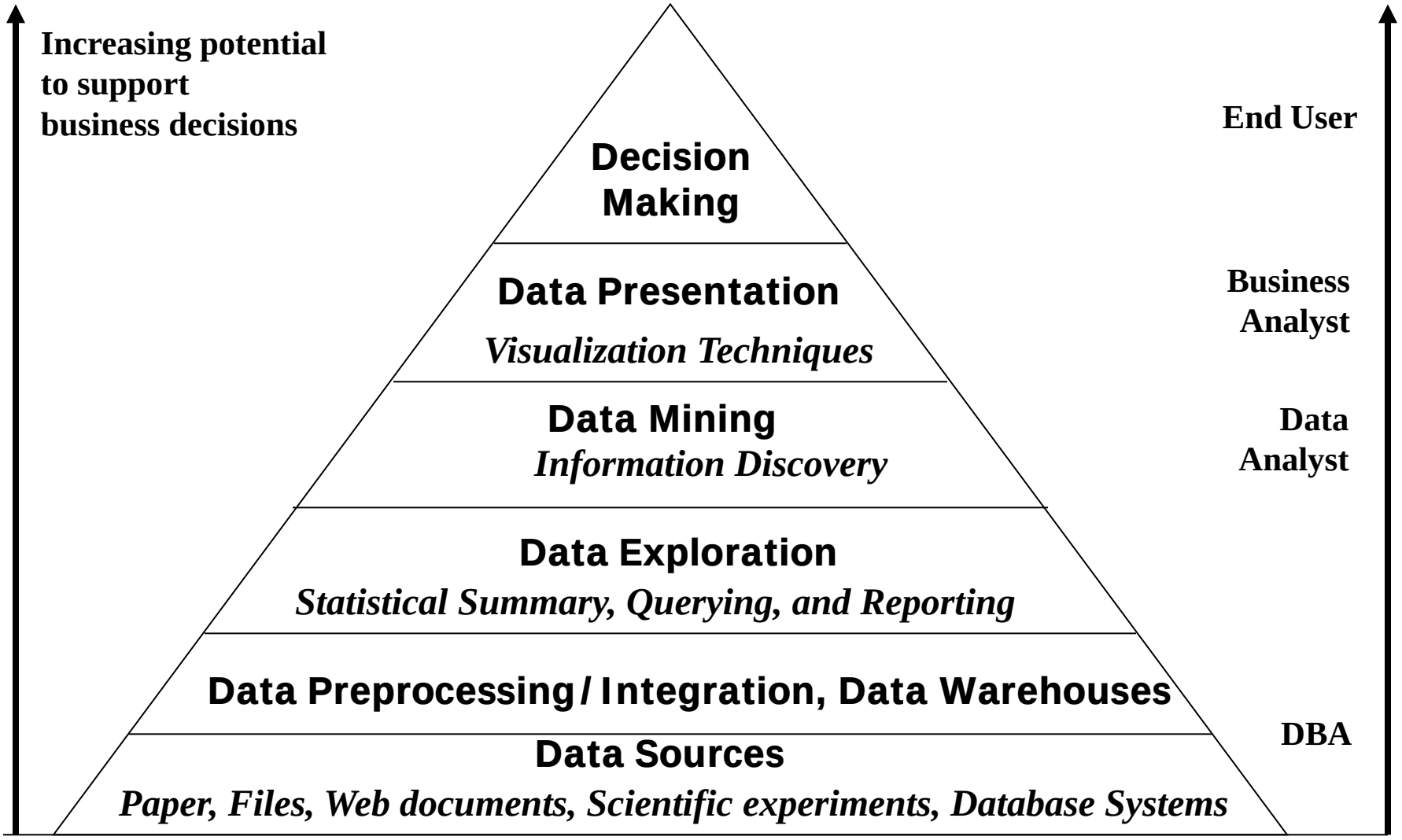
- Data mining—core of knowledge discovery process



KDD Process: Several Key Steps

- Learning the application domain
  - relevant prior knowledge and goals of application
- Creating a target data set: data selection
- Data cleaning and preprocessing: (may take 60% of effort!)
- Data reduction and transformation
  - Find useful features, dimensionality/variable reduction, invariant representation
- Choosing functions of data mining
  - summarization, classification, regression, association, clustering
- Choosing the mining algorithm(s)
- Data mining: search for patterns of interest
- Pattern evaluation and knowledge presentation
  - visualization, transformation, removing redundant patterns, etc.
- Use of discovered knowledge

Data Mining and Business Intelligence



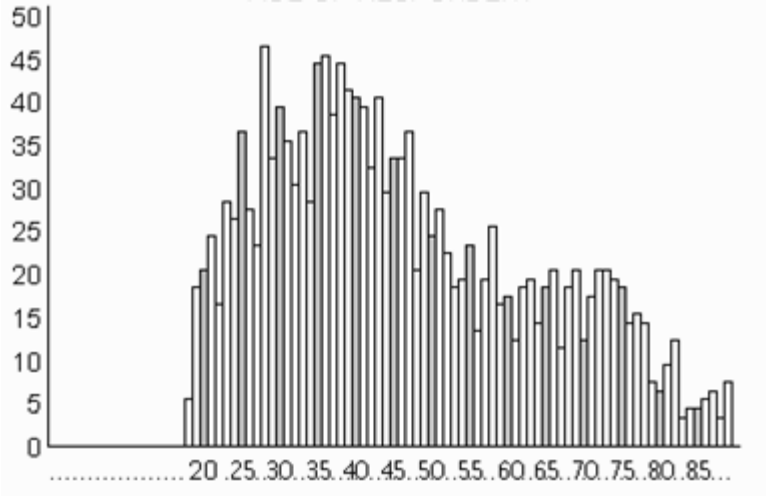
Data Mining Functionalities

- Multidimensional concept description: Characterization and discrimination (辨别)
  - Generalize, summarize, and contrast data characteristics, e.g., dry vs. wet regions
- Frequent patterns, association, correlation vs. causality
  - Diaper → Beer [0.5%, 75%] (Correlation or causality?)
- Classification and prediction
  - Construct models (functions) that describe and distinguish classes or concepts for future prediction
    - E.g., classify countries based on climate
  - Predict some unknown or missing numerical values

Data Mining Functionalities (2)

- Cluster analysis
  - Class label is unknown: Group data to new classes, e.g., cluster houses to find distribution patterns
- Outlier analysis
  - Outlier: Data object that does not comply with the general behavior of the data
  - Noise or exception? Useful in fraud detection, rare events analysis
- Trend and evolution analysis
  - Trend and deviation: e.g., regression analysis
  - Sequential pattern mining: e.g., digital camera, data stream
  - Periodicity analysis
  - Similarity-based analysis
- Other pattern-oriented or statistical analyses

Data Summarization



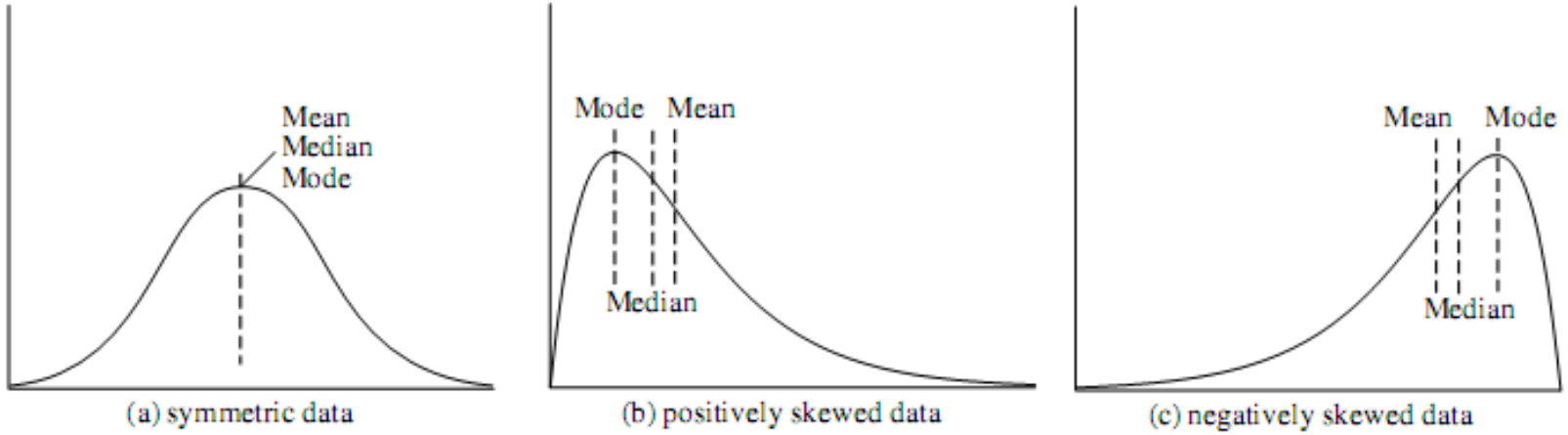
Measuring the Central Tendency

- Mean (algebraic) : 平均数 $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \quad \mu = \frac{\sum x}{N}$
  - Weighted arithmetic mean:
  - Trimmed mean: chopping extreme values $\bar{x} = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i}$
- Median (holistic): 中位数
  - Middle value if odd number of values, or average of the middle two values otherwise
- Mode(holistic) : 众数
  - Value that occurs most frequently in the data
  - Unimodal, bimodal, trimodal
- Empirical formula: 适度倾斜的单峰频率曲线

$$mean - mode = 3 \times (mean - median)$$

Central Tendency

- Symmetric versus positively and negatively skewed data

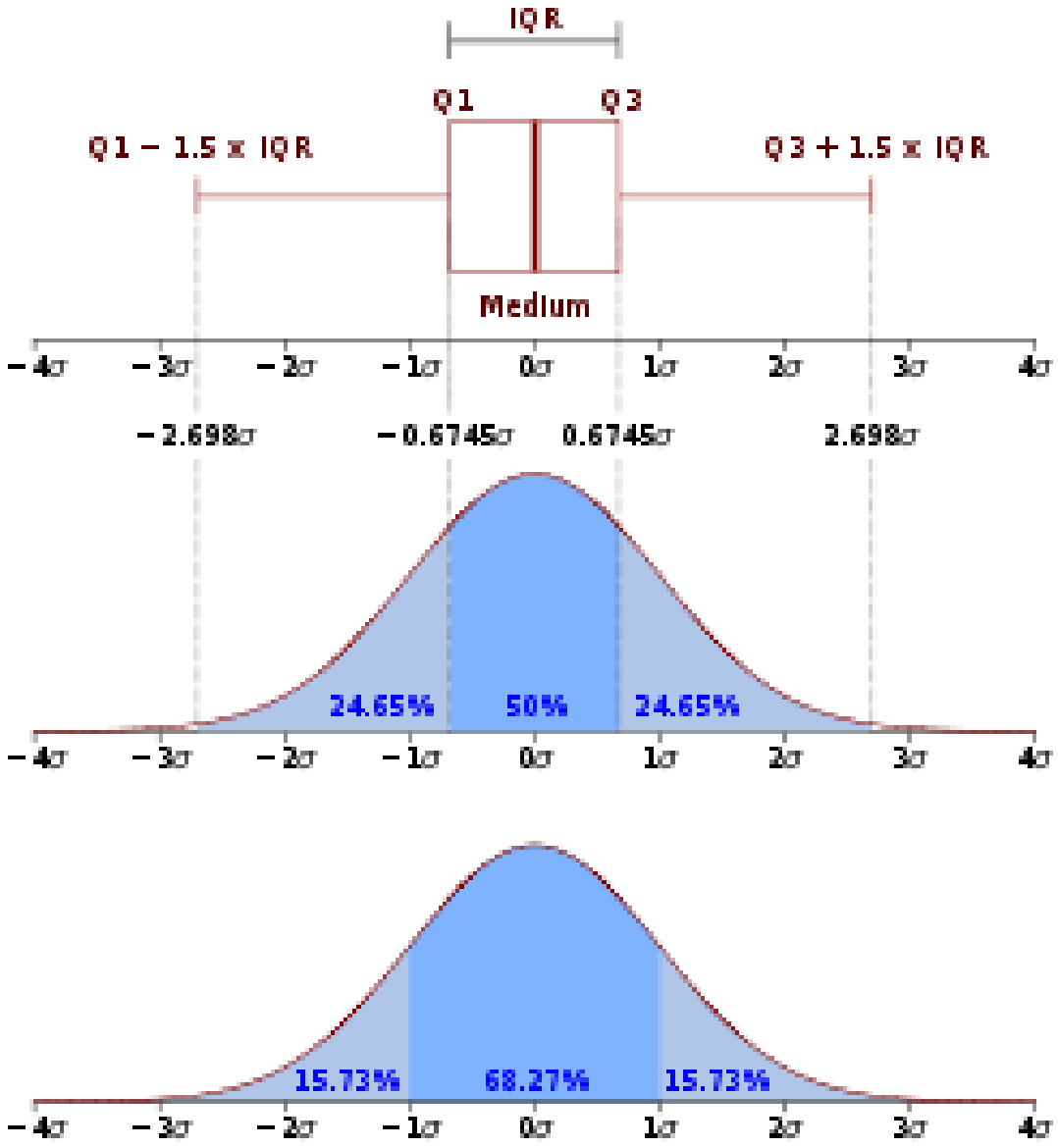


Measuring the Dispersion of Data

- The degree to which numerical data tend to spread is called the dispersion 离差, or variance方差 of the data.
- Percentile 分位数**:** The k^{th} percentile of a set of data in numerical order is the value x_i having the property that k percent of the data entries lies at or below x_i. The median is the 50th percentile.

['kwɔ:tail]

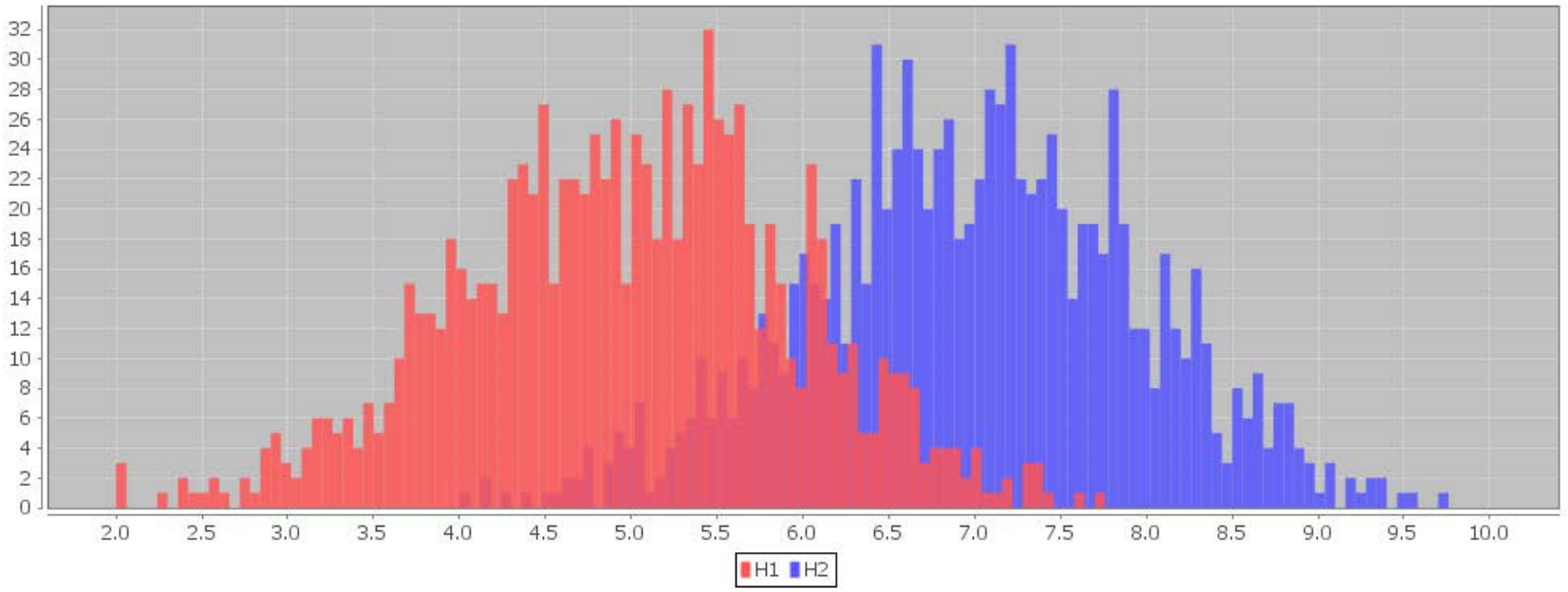
- Quartiles 四分位数: *Q1* (25th percentile), *Q2*, *Q3* (75th percentile)
- Interquartile range 四分位距: *IQR = Q3 – Q1*
- Five number summary: *min*, *Q1*, *median*, *Q3*, *max*
  - Outlier: Usually, a value higher/lower than 1.5 x *IQR*
- Variance方差 and standard deviation 标准差: (algebraic, scalable computation)


IQR
Q1
Q3
Q1 − 1.5 x IQR
Q3 + 1.5 x IQR
Medium
−4σ
−3σ
−2σ
−1σ
0σ
1σ
2σ
3σ
4σ
−2.698σ
−0.6745σ
0.6745σ
2.698σ
24.65%
50%
24.65%
15.73%
68.27%
15.73%

Plots Analysis

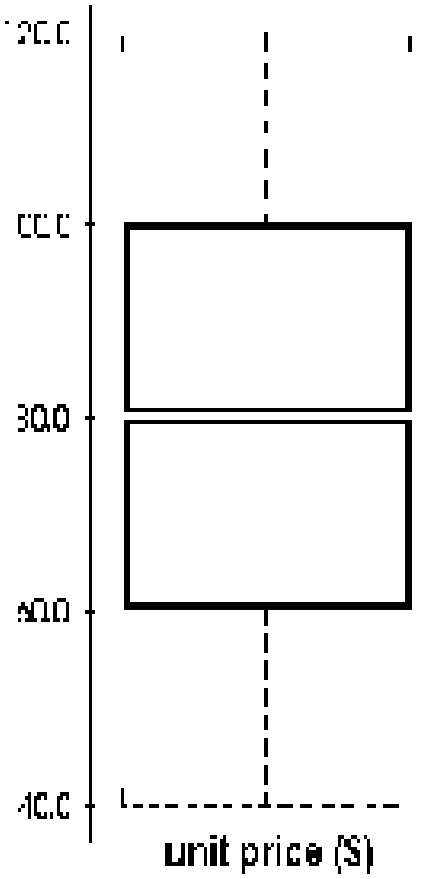
- Histograms
- Box Charts
- Quantile Charts
- Quantile-Quantile Charts
- Scatter Charts

Histograms

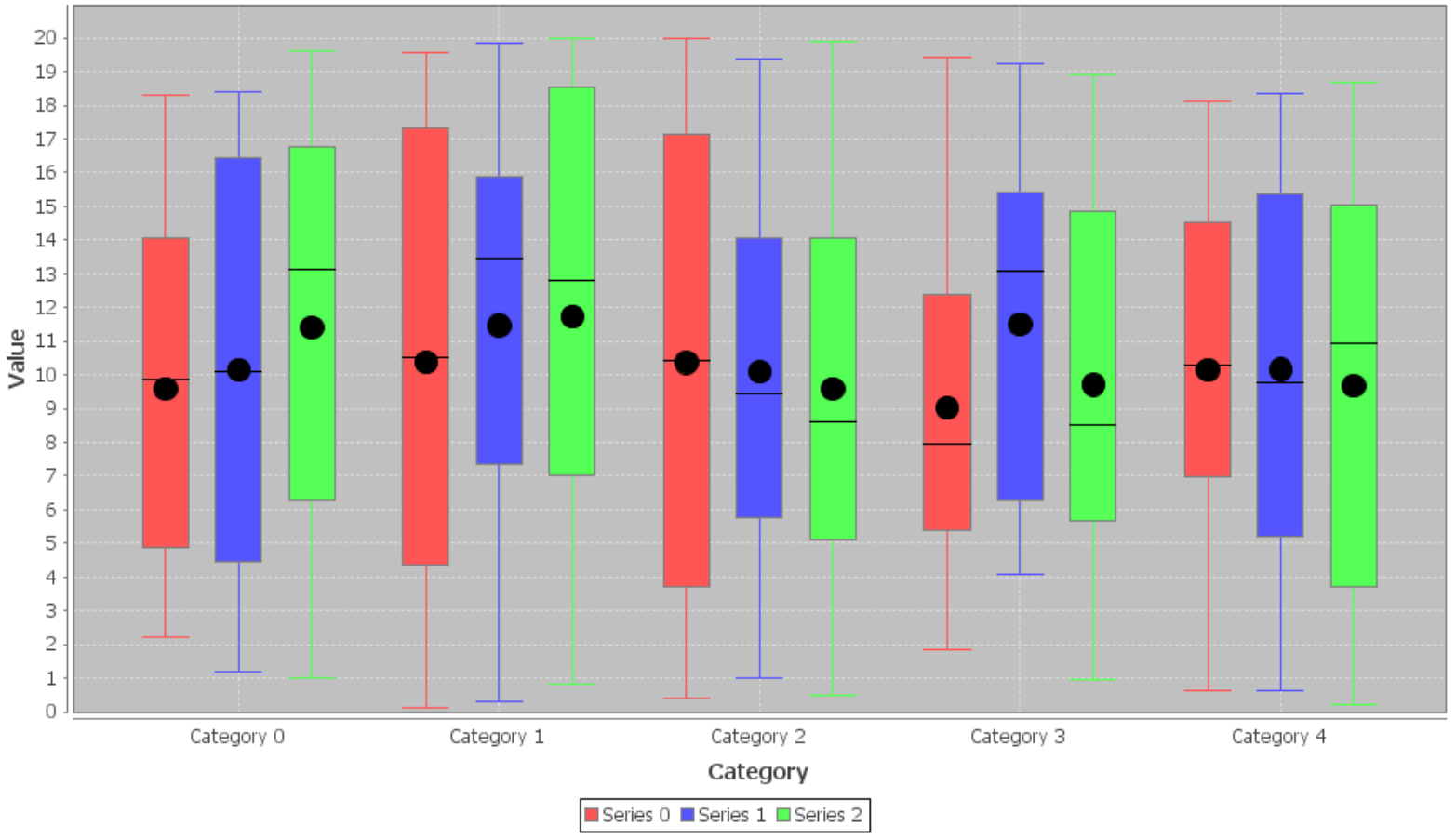

32
30
28
26
24
22
20
18
16
14
12
10
8
6
4
2
0
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
8.5
9.0
9.5
10.0
H1
H2

Boxplot Analysis

- Five-number summary of a distribution:
  - Minimum, Q1, Median, Q3, Maximum
- Boxplot
  - Data is represented with a box
  - The ends of the box are at the first and third quartiles, i.e., the height of the box is IQR
  - The median is marked by a line within the box
  - Whiskers: two lines outside the box extend to Minimum and Maximum

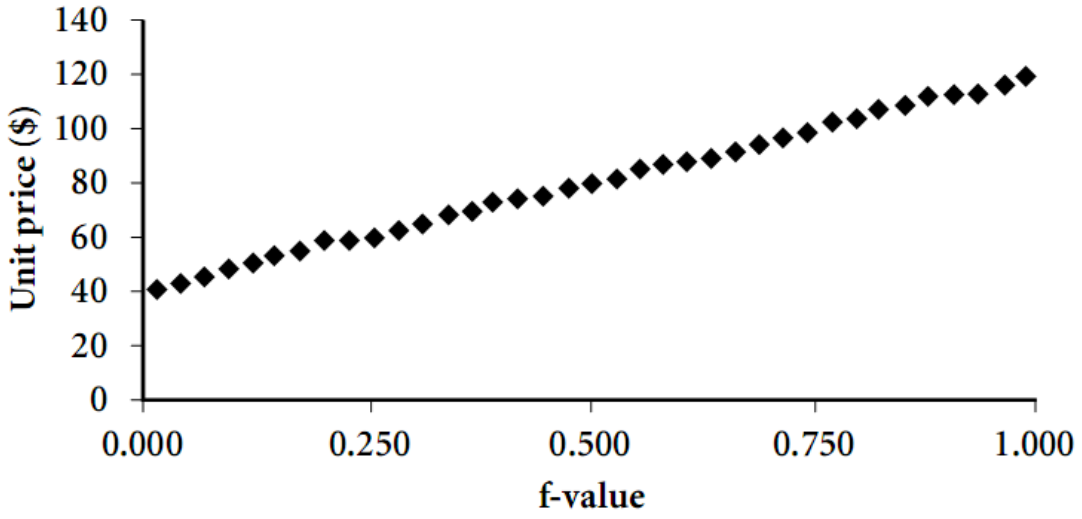


Boxplot


Value
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
0
Category 0
Category 1
Category 2
Category 3
Category 4
Category
Series 0
Series 1
Series 2

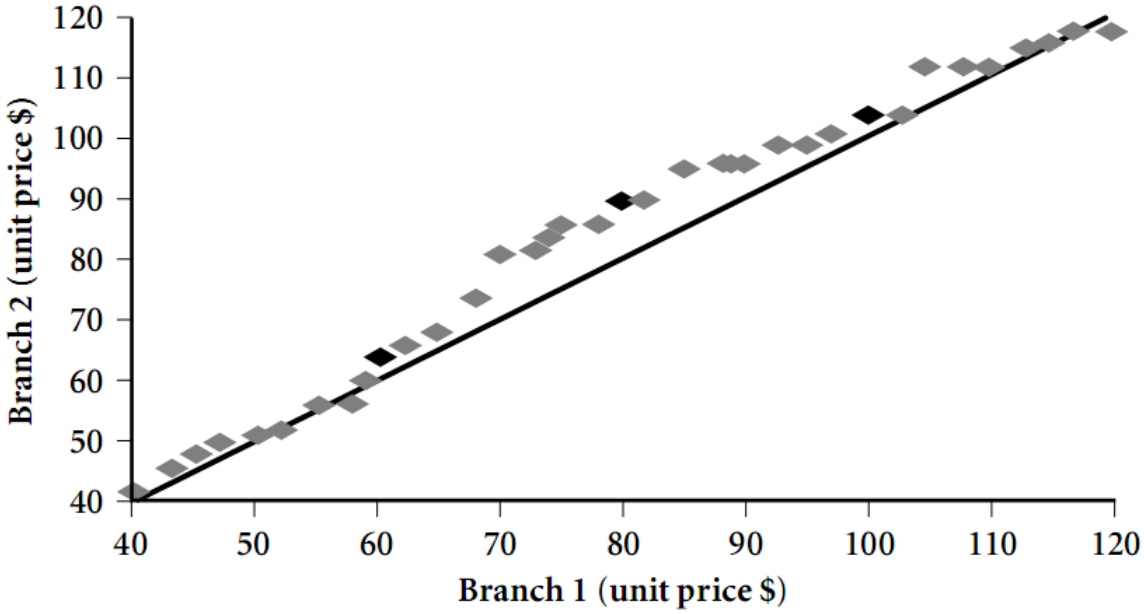
Quantile Plot

- Displays all of the data (allowing the user to plots quantile information)
  - For a data x_i data sorted in increasing order, f_i indicates that approximately 100 f_i% of the data are below or equal to the value x_i
  - If the distributions are linearly related, the points in the Q-plot will approximately lie on a line.

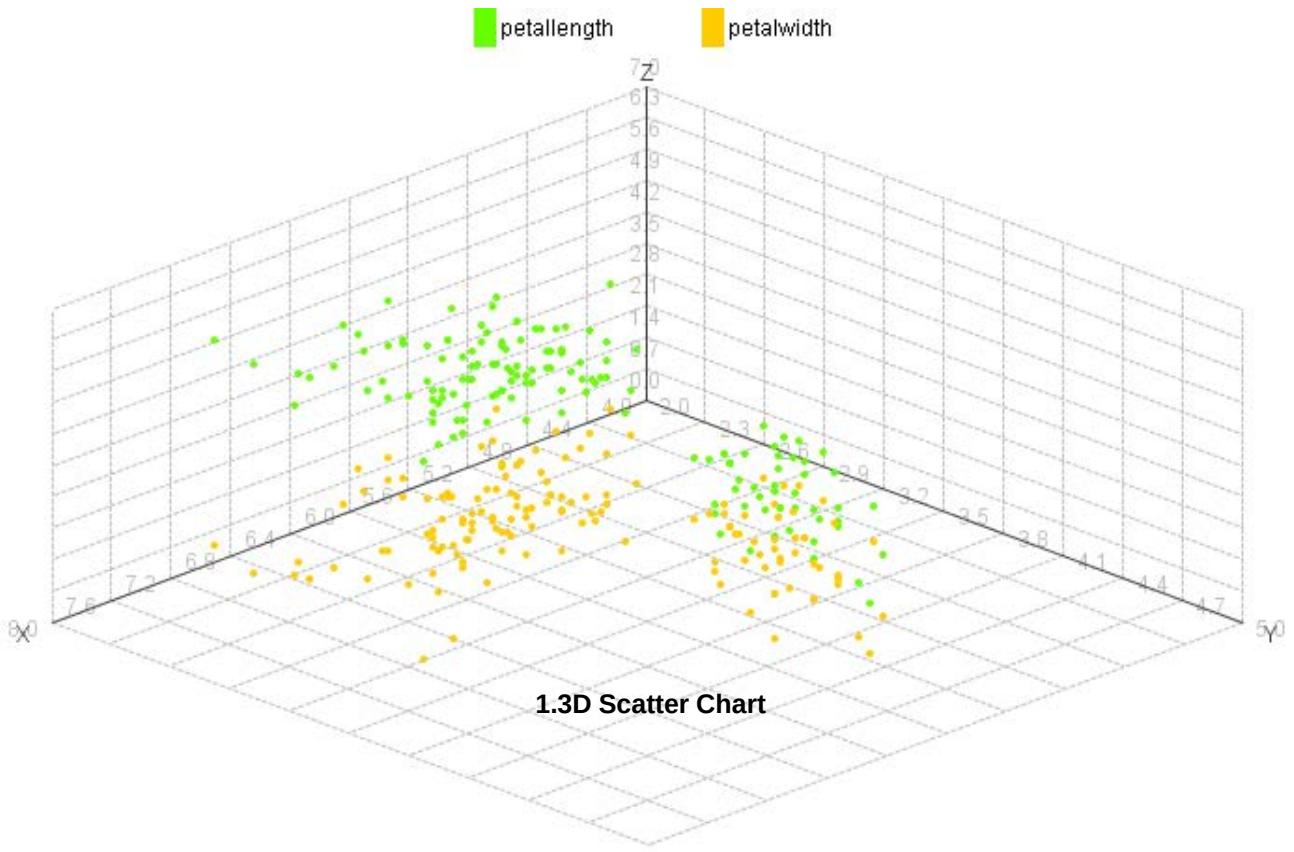


Quantile-Quantile (Q-Q) Plot

- Graphs the quantiles of one distribution against the corresponding quantiles of another
- Allows the user to view whether there is a shift in going from one distribution to another
  - If the two distributions being compared are similar, the points in the Q-Q plot will approximately lie on the line y = x.

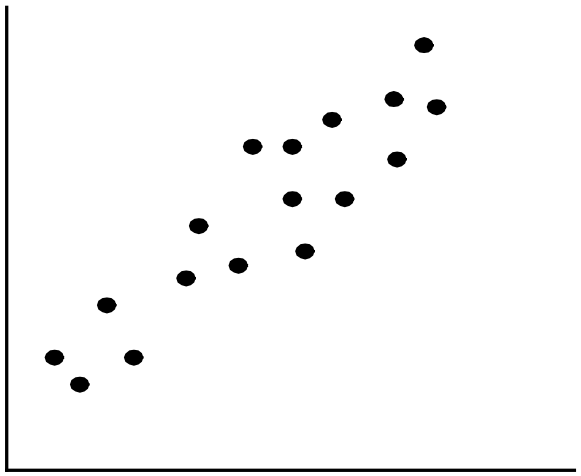


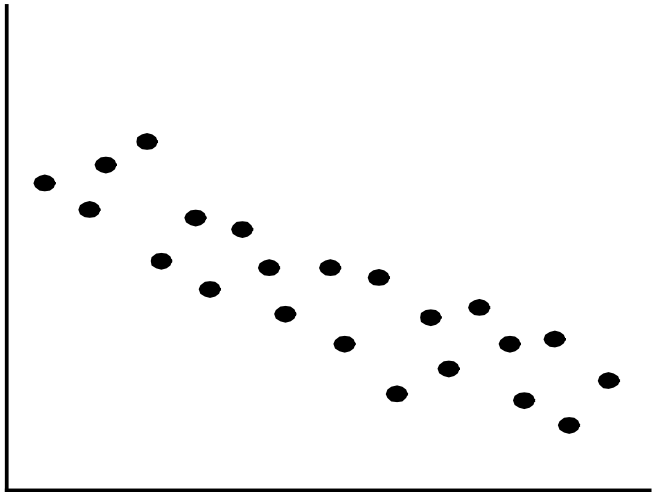
Scatter Charts

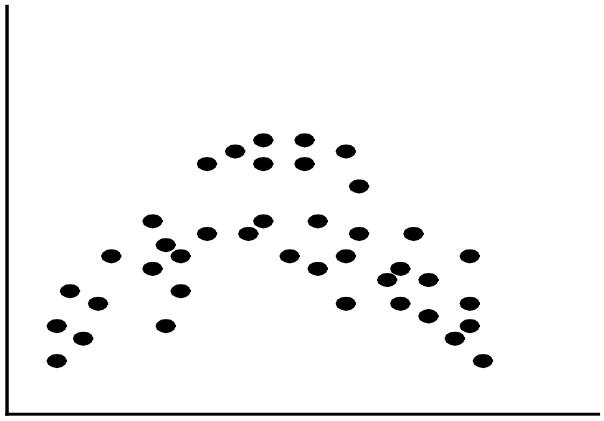


1.3D Scatter Chart

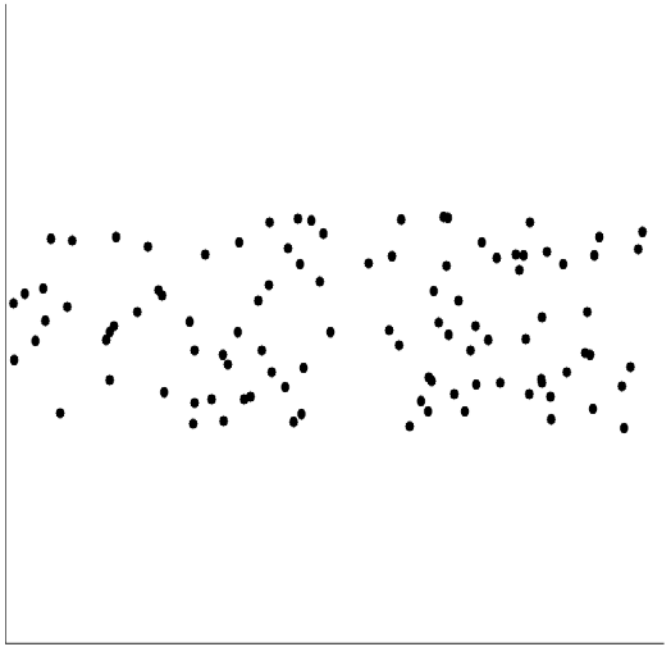
Positively and Negatively Correlated Data

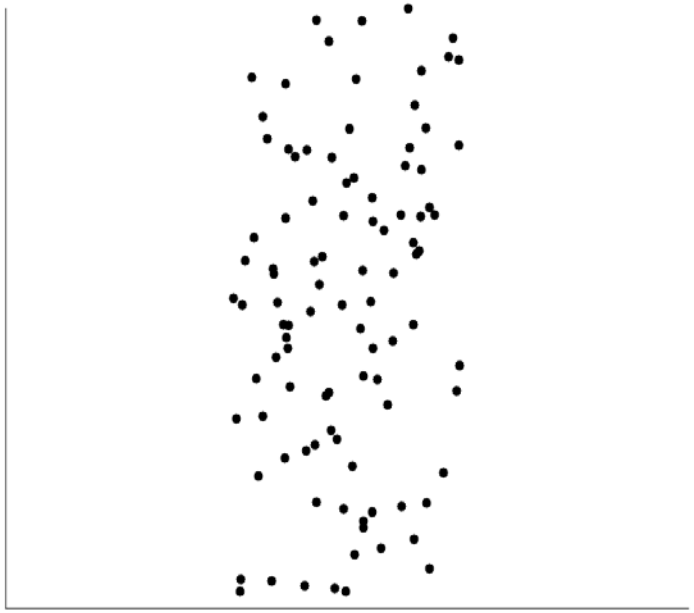




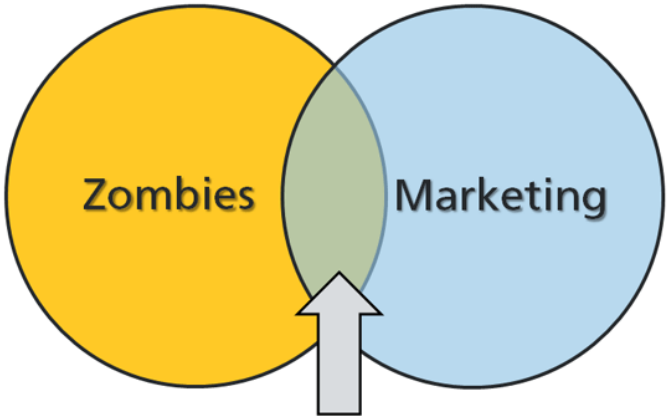


Not Correlated Data





Relevance Analysis



Relevance and Correlation

- Relevance (关联性)
  - Closely connected or appropriate to something
- Correlation (相关性)
  - A mutual relationship or connection between two or more things

Why attribute relevance analysis?

- It is difficult for users to determine which dimensions should be included
- Attribute relevance analysis is used to
  - Filter out statistically irrelevant or weakly relevant attributes
  - Retain or even rank the most relevant attributes

Relevance measures

- The general idea behind attribute relevance analysis is to compute some measure which is used to quantify the relevance of an attribute with respect to a given class
- Popular measures include:
  - Information Gain (信息增益)
  - Gain Ratio
  - Gini Index

Information gain measure

- S: Training set
- S_i: Training instances of class C_i (i = 1, …, m) based on class label
- a_j: Values of attribute A (j = 1, …, n)

The information needed to correctly classify the training set is

$$I(S_1, S_2, \ldots S_m) = -\sum_{i=1}^{m} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$

Suppose attribute A is selected to partition the training set into the subsets $\{S'_1, S'_2, \ldots, S'_n\}$, then the entropy 熵 of A, i.e. the information needed to classify all the instances in those subsets is

$$Ent(A) = \sum_{j=1}^{n} \frac{|S'_j|}{|S|} I(S'_{1j}, S'_{2j}, \ldots, S'_{mj}) = \sum_{j=1}^{n} \frac{|S'_j|}{|S|} \left[-\sum_{i=1}^{m} \frac{|S'_{ij}|}{|S'_j|} \log_2 \frac{|S'_{ij}|}{|S'_j|} \right]$$

Where S'_j is the instances of class C_i that are covered by S'_j then the information gain of selecting A is

$$Gain(A) = I(S_1, S_2, \ldots S_m) - Ent(A)$$

The bigger the information gain, the more relevant the attribute A

Example 1

- ***S*: Training set**

gender	major	birth_country	age_range	gpa	count	class
M	Science	Canada	20-25	Very_good	16	GS
F	Science	Foreign	25-30	Excellent	22	GS
M	Engineering	Foreign	25-30	Excellent	18	GS
F	Science	Foreign	25-30	Excellent	25	GS
M	Science	Canada	20-25	Excellent	21	GS
F	Engineering	Canada	20-25	Excellent	18	GS
M	Science	Foreign	<20	Very_good	18	S
F	Business	Canada	<20	Fair	20	S
M	Business	Canada	<20	Fair	22	S
F	Science	Canada	20-25	Fair	24	S
M	Engineering	Foreign	20-25	Very_good	22	S
F	Engineering	Canada	<20	Excellent	24	S

Example 2

- S_1: Graduate students (Σ=120)

gender	major	birth_country	age_range	gpa	count
M	Science	Canada	20-25	Very_good	16
F	Science	Foreign	25-30	Excellent	22
M	Engineering	Foreign	25-30	Excellent	18
F	Science	Foreign	25-30	Excellent	25
M	Science	Canada	20-25	Excellent	21
F	Engineering	Canada	20-25	Excellent	18

- S_2: Undergraduate students (Σ=130)

gender	major	birth_country	age_range	gpa	count
M	Science	Foreign	<20	Very_good	18
F	Business	Canada	<20	Fair	20
M	Business	Canada	<20	Fair	22
F	Science	Canada	20-25	Fair	24
M	Engineering	Foreign	20-25	Very_good	22
F	Engineering	Canada	<20	Excellent	24

Example 3

- The information needed to correctly classify the training set is
  - S_1: Graduate students (Σ=120)
  - S_2: Undergraduate students (Σ=130)

$$I(S_1, S_2, \ldots S_m) = -\sum_{i=1}^{m} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|} \qquad (m = 2, i = 1,2)$$

$$I(S_1, S_2) = I(120,130) = -\frac{120}{250} \log_2 \frac{120}{250} - \frac{130}{250} \log_2 \frac{130}{250} = 0.9988$$

Example 4

gender	major	birth_country	age_range	gpa	count	class
M	Science	Canada	20-25	Very_good	16	GS
F	Science	Foreign	25-30	Excellent	22	GS
M	Engineering	Foreign	25-30	Excellent	18	GS
F	Science	Foreign	25-30	Excellent	25	GS
M	Science	Canada	20-25	Excellent	21	GS
F	Engineering	Canada	20-25	Excellent	18	GS
M	Science	Foreign	<20	Very_good	18	S
F	Business	Canada	<20	Fair	20	S
M	Business	Canada	<20	Fair	22	S
F	Science	Canada	20-25	Fair	24	S
M	Engineering	Foreign	20-25	Very_good	22	S
F	Engineering	Canada	<20	Excellent	24	S

- Suppose attribute *major* is selected to partition the training set

for *major* = "*Science*":
$S_{11} = 84, S_{21} = 42$

for *major* = "*Engineering*":
$S_{12} = 36, S_{22} = 46$

for *major* = "*Business*":
$S_{13} = 0, S_{23} = 42$

i =1,2 m =2
j =1,2,3 n =3,
Number of S_{ij} is 6

Example 5

- The information needed to correctly classify the training set is

$$I(S_1, S_2) = I(120,130) = -\frac{120}{250}\log_2\frac{120}{250} - \frac{130}{250}\log_2\frac{130}{250} = 0.9988$$

- Suppose attribute major is selected to partition the training set

for *major* = "*Science*":
$S_{11} = 84, S_{21} = 42$

$$I(S_{11}, S_{21}) = -\frac{84}{126}\log_2\frac{84}{126} - \frac{42}{126}\log_2\frac{42}{126} = 0.9183$$

for *major* = "*Engineering*":
$S_{12} = 36, S_{22} = 46$

$$I(S_{12}, S_{22}) = -\frac{36}{82}\log_2\frac{36}{82} - \frac{46}{82}\log_2\frac{46}{82} = 0.9892$$

for *major* = "*Business*":
$S_{13} = 0, S_{23} = 42$

$$I(S_{13}, S_{23}) = 0$$

- Then the entropy of major is

$$Ent(major) = \frac{126}{250} I(S_{11}, S_{21}) + \frac{82}{250} I(S_{12}, S_{22}) + \frac{42}{250} I(S_{13}, S_{23}) = 0.7873$$

- Then the information gain of major is

$$Gain(major) = I(S_1, S_2) - Ent(major) = 0.2115$$

Example 6

- We can also get the information gain of other attributes:

$$Gain(gender) = 0.0003$$

$$Gain(birth\_country) = 0.0407$$

$$Gain(gpa) = 0.4490$$

$$Gain(age\_rabge) = 0.5971$$

- Now suppose we use an *attribute relevance threshold* of 0.1:
  - *gender* and *birth_country* are removed as weakly relevant attributes
  - *major*, *gpa*, and *age_range* are kept as strong relevant attributes

Correlation Analysis

Correlation Analysis (Numerical Data)

- Correlation coefficient

$$r_{A,B} = \frac{\sum (A - \overline{A})(B - \overline{B})}{(n-1)\sigma_A \sigma_B} = \frac{\sum (AB) - n\overline{A}\,\overline{B}}{(n-1)\sigma_A \sigma_B}$$

  - Where n is the number of tuples, $\overline{A}$ and $\overline{B}$ are the respective means of A and B, σ_A and σ_B are the respective standard deviation of A and B, and $\Sigma(AB)$ is the sum of the AB cross-product.
  - If $r_{A,B} > 0$, A and B are positively correlated (A's values increase as B's). The higher, the stronger correlation.
  - $r_{A,B} = 0$: independent; $r_{A,B} < 0$: negatively correlated

Correlation Analysis (Categorical Data)

- X^2 (chi-square)
  - Suppose *A* has *c* distinct values, namely a_1 to a_c. *B* has *r* distinct values, namely b_1 *to* b_r.

$$\chi^2 = \sum_{i=1}^{c} \sum_{j=1}^{r} \frac{(o_{ij} - e_{ij})^2}{e_{ij}}$$

  - Where o_{ij} is the observed frequency (actual count) of the joint event (A_i, B_j) and e_{ij} is the expected frequency of (A_i, B_j), which can be computed as

$$e_{ij} = \frac{count(A = a_i) \times count(B = b_j)}{N}$$

Correlation Analysis (Categorical Data)

- If the computed X^2 value is (more) above the pre-defined values, we can reject the hypothesis that *A* and *B* are independent and conclude that the two attributes are (strongly) correlated.
- (*r*-1)(*c*-1) is degree of freedom
- The cells that contribute the most to the X^2 value are those whose actual count is very different from the expected count

Chi-Square Calculation: An Example

	Play chess	Not play chess	Sum (row)
Like science fiction	250 (90)	200 (360)	450
Not like science fiction	50 (210)	1000 (840)	1050
Sum(col.)	300	1200	1500

- X^2(chi-square) calculation

$$\chi^2 = \frac{(250-90)^2}{90} + \frac{(50-210)^2}{210} + \frac{(200-360)^2}{360} + \frac{(1000-840)^2}{840} = 507.93$$

- It shows that *like_science_fiction* and *play_chess* are correlated in the group.

Upper critical values of chi-square distribution with ν degrees of freedom

ν	Probability of exceeding the critical value 0.10	0.05	0.025	0.01	0.001
1	2.706	3.841	5.024	6.635	10.828
2	4.605	5.991	7.378	9.210	13.816
3	6.251	7.815	9.348	11.345	16.266
4	7.779	9.488	11.143	13.277	18.467
5	9.236	11.070	12.833	15.086	20.515
6	10.645	12.592	14.449	16.812	22.458
7	12.017	14.067	16.013	18.475	24.322
8	13.362	15.507	17.535	20.090	26.125
9	14.684	16.919	19.023	21.666	27.877
10	15.987	18.307	20.483	23.209	29.588
11	17.275	19.675	21.920	24.725	31.264
12	18.549	21.026	23.337	26.217	32.910
13	19.812	22.362	24.736	27.688	34.528
14	21.064	23.685	26.119	29.141	36.123
15	22.307	24.996	27.488	30.578	37.697
16	23.542	26.296	28.845	32.000	39.252
17	24.769	27.587	30.191	33.409	40.790
18	25.989	28.869	31.526	34.805	42.312
19	27.204	30.144	32.852	36.191	43.820
20	28.412	31.410	34.170	37.566	45.315

Lower critical values of chi-square distribution with ν degrees of freedom

ν	Probability of exceeding the critical value 0.90	0.95	0.975	0.99	0.999
1.	.016	.004	.001	.000	.000
2.	.211	.103	.051	.020	.002
3.	.584	.352	.216	.115	.024
4.	1.064	.711	.484	.297	.091
5.	1.610	1.145	.831	.554	.210
6.	2.204	1.635	1.237	.872	.381
7.	2.833	2.167	1.690	1.239	.598
8.	3.490	2.733	2.180	1.646	.857
9.	4.168	3.325	2.700	2.088	1.152
10.	4.865	3.940	3.247	2.558	1.479
11.	5.578	4.575	3.816	3.053	1.834
12.	6.304	5.226	4.404	3.571	2.214
13.	7.042	5.892	5.009	4.107	2.617
14.	7.790	6.571	5.629	4.660	3.041
15.	8.547	7.261	6.262	5.229	3.483
16.	9.312	7.962	6.908	5.812	3.942
17.	10.085	8.672	7.564	6.408	4.416
18.	10.865	9.390	8.231	7.015	4.905
19.	11.651	10.117	8.907	7.633	5.407
20.	12.443	10.851	9.591	8.260	5.921

Correlation

- Correlation does not imply causality
  - number of *hospitals* and number of *car-theft* in a city are correlated
  - Both are causally linked to the third variable: *population*

Mining Association Rule

What is association rule mining?

- Association rule mining searches for interesting relationships among items in a given data set.
- Association rule mining is motivated for market basket analysis, where the customer buying habits are analyzed by finding associations between the different items that customers place in their shopping baskets.

Association rules

- Association rules are in the form of:
  - **Body⟶Head [support, confidence]**
- Example:
  - *buys*(*X*, “diapers”) ⇒ *buys*(*X*, “beers”) [0.5%, 60%]
  - *major*(*X*, “CS”) ∧ *takes*(*X*, “DB”)⇒*grade*(*X*, “A”) [1%, 75%]

Formal Definition of Association Rule

- $I = \{i_1, i_2, \ldots, i_m\}$ is an itemset (项集), a set of items
- D is the task relevant data, which is a set of transactions where each transaction T is an itemset such that $T \subseteq I$

- If A is an itemset, we say T contains A iff $A \subseteq T$
- An association rule is of the form $A \Rightarrow B$, where $A \subset I$, $B \subset I$, and $A \cap B = \Phi$
- **The rule holds in D with**
  - **support($A \Rightarrow B$) = Prob(AB)**
  - **confidence ($A \Rightarrow B$) = Prob($B|A$)**

TID	itemset
T_1	$i_1 i_3 i_6$
T_2	$i_1 i_5 i_8 i_{10}$
T_3	$i_1 i_2 i_3 i_4$
T_4	$i_1 i_2 i_5$
T_5	$i_1 i_4$
T_6	$i_2 i_4 i_5 i_6$

Support and confidence

- Support
  - The possibility of A and B are happened together.
  - Too small, can not find rules.
  - Too larger, may be the rules is well-known.

- Confidence
  - The possibility of A is happened meanwhile B is happened together.
  - Larger while rules are stronger

Rules satisfying both *min_sup* and *min_conf* are strong

Frequent Itemset

- An itemset contains *k* items is a *k*-itemset
- The number of transactions that contain an itemset is the frequency (or support count) of the itemset
- If an itemset satisfies *min_sup*, or its frequency satisfies *min_freq,* then it is a **frequent itemset**

Categories of association rules

- **Based on the types of values**
  - **Boolean association rule**

  $buys(X, \text{“diapers”}) \Rightarrow buys(X, \text{“beers”})$ [0.5%, 60%]
  - **Quantitative association rule**

  $age(X, \text{“30-39”}) \wedge income(X, \text{“42-48K”}) \Rightarrow buys(X, \text{“computer”})$ [1%, 75%]
- **Based on the dimensions of data**
  - **Single-dimensional association rule**

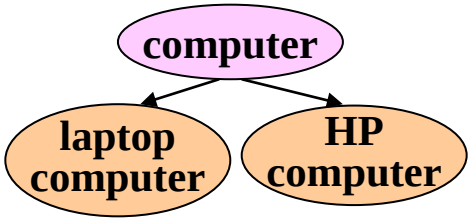
  $buys(X, \text{“diapers”}) \Rightarrow buys(X, \text{“beers”})$ [0.5%, 60%]
  - **Multidimensional association rule**

  $age(X, \text{“30-39”}) \wedge income(X, \text{“42-48K”}) \Rightarrow buys(X, \text{“computer”})$ [1%, 75%]
- **Based on the levels of abstractions**
  - **Single-level association rule**

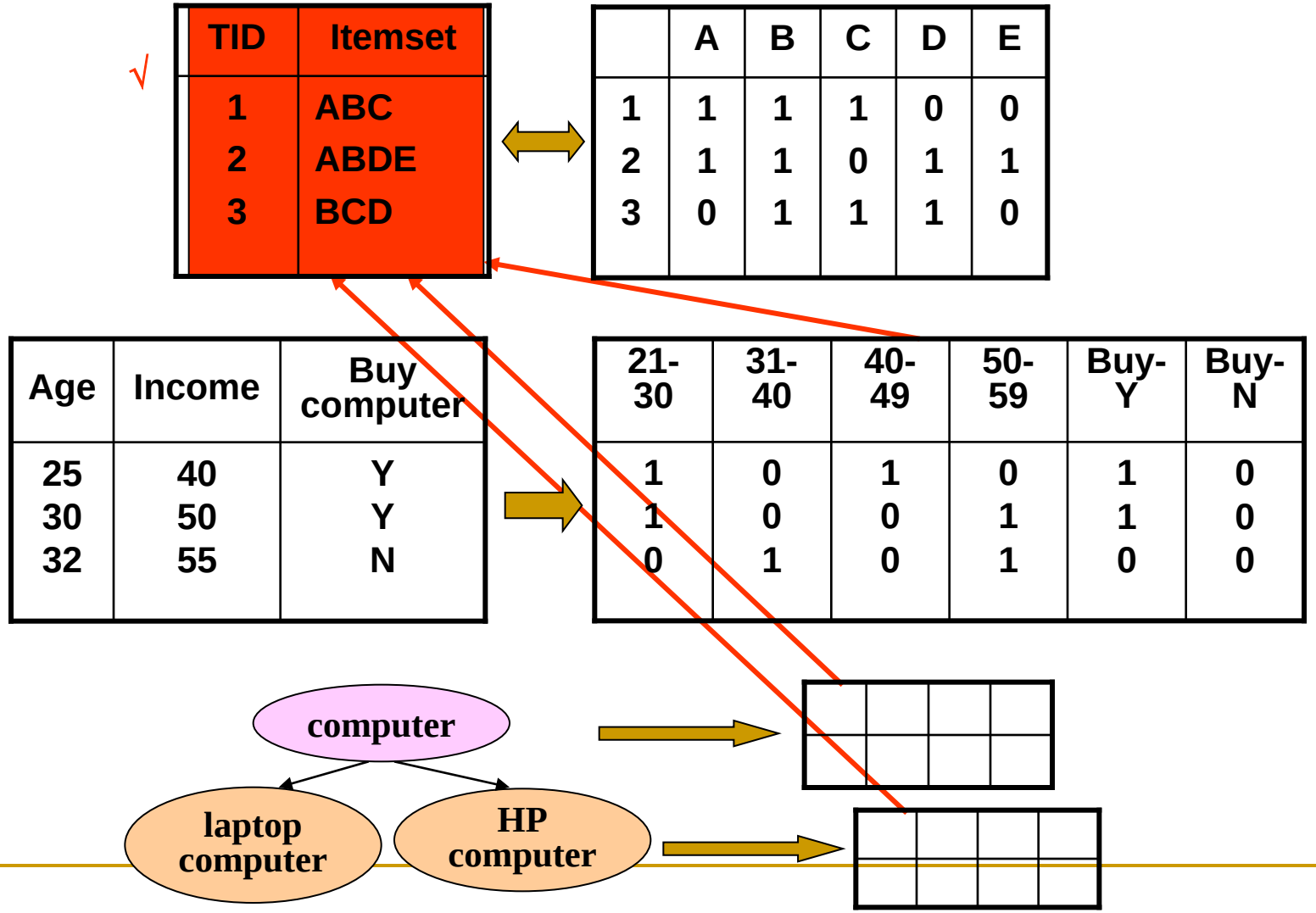
  $age(X, \text{“30-34”}) \Rightarrow buys(X, \text{“computer”})$ [1%, 75%]
  - **Multilevel association rule**

  $age(X, \text{“30-32”}) \Rightarrow buys(X, \text{“laptop computer”})$ [0.5%, 80%]

  $age(X, \text{“30-34”}) \Rightarrow buys(X, \text{“computer”})$ [1%, 75%]



Prepare Itemsets



Simple Discretization Methods: Binning

- Equal-width (distance) partitioning
  - Divides the range into *N* intervals of equal size: uniform grid
  - if *A* and *B* are the lowest and highest values of the attribute, the width of intervals will be: $W = (B - A)/N$.
  - The most straightforward, but outliers may dominate presentation
  - Skewed data is not handled well
- Equal-depth (frequency) partitioning
  - Divides the range into *N* intervals, each containing approximately same number of samples
  - Good data scaling
  - Managing categorical attributes can be tricky

Binning Methods for Data Smoothing

- Sorted data for price (in dollars): 4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34

* Partition into equal-frequency (equi-depth) bins:
  - Bin 1: 4, 8, 9, 15
  - Bin 2: 21, 21, 24, 25
  - Bin 3: 26, 28, 29, 34
* Smoothing by bin means:
  - Bin 1: 9, 9, 9, 9
  - Bin 2: 23, 23, 23, 23
  - Bin 3: 29, 29, 29, 29
* Smoothing by bin boundaries:
  - Bin 1: 4, 4, 4, 15
  - Bin 2: 21, 21, 25, 25
  - Bin 3: 26, 26, 26, 34

Two-step process of association rule mining

- Step1: find all frequent itemsets
- Step2: generate strong association rules from the frequent itemsets
  - Step2 is easier, therefore most works on association rule mining focus on step1

considering $\text{confidence}(A \Rightarrow B) = \text{Prob}(B \mid A) = \frac{\text{support}(AB)}{\text{support}(A)}$

- For each frequent itemset *l*, generate all non-empty subsets of *l*
- For every non-empty subset *s* of *l*, output the rule "*s* $\Rightarrow$ (*l* - *s*)" if $\frac{\textbf{support}(l)}{\textbf{support}(s)} \geq \textbf{min\_conf}$

Apriori

- In initial is for finding frequent itemsets for Boolean association rules
- **level-wise search, where *k*-itemsets are used to explore (*k*+1)-itemsets**
- **The key of Apriori is the *a priori* knowledge:**

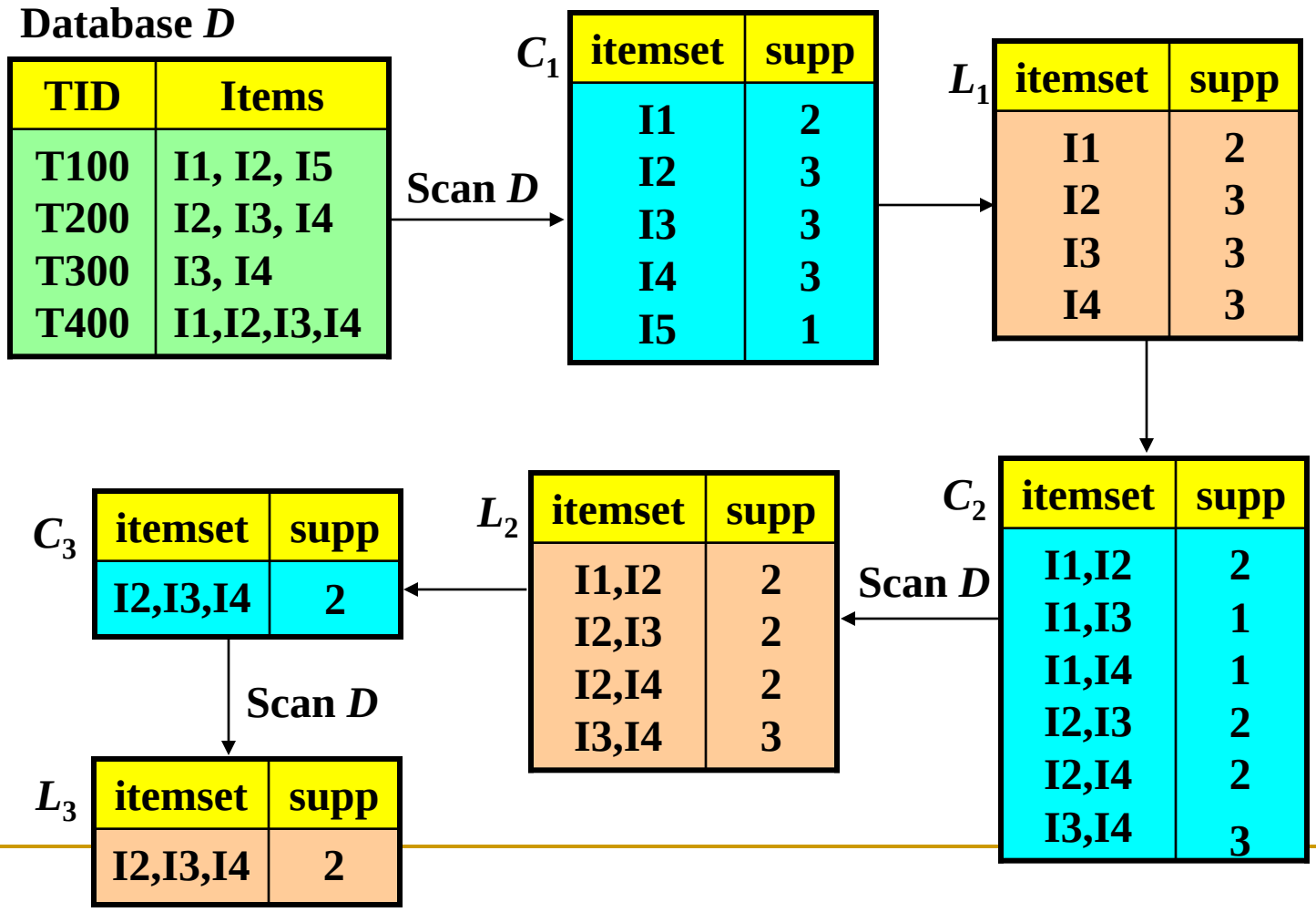
  All non-empty subsets of a frequent itemset must also be frequent
- From L_k to L_{k+1}:
  - Join: a set of candidate frequent (k+1)-itemsets, C_{k+1}, is generated by joining L_k with itself (the set of frequent k-itemsets is denoted Lk)

    $L_k \bowtie L_k = \{A \bowtie B \mid A, B \in L_k, |A) B| = k - 1\}$
  - Prune: a scan of the database for determining the count of each candidate in C_{k+1} would result in the determination of L_{k+1}

    a priori knowledge is used to reduce the size of C_{k+1}

Apriori (II)

- an example Suppose min_sup_count = 2



Apriori (III)

L_3

itemset	supp
I2,I3,I4	2

L_2

itemset	supp
I1,I2	2
I2,I3	2
I2,I4	2
I3,I4	3

L_1

itemset	supp
I1	2
I2	3
I3	3
I4	3

- I2^I3⇒ I4 Conf=2/2=1
- I3^I4⇒ I2 Conf=2/3=0.67
- I2^I4⇒ I3 Conf=2/2=1
- I4 ⇒ I2^I3 Conf=2/3=0.67
- I2 ⇒ I3^I4 Conf=3/3=0.67
- I3 ⇒ I2^I4 Conf=2/3=0.67

- I1⇒ I2 Conf=2/2=1
- I2⇒ I1 Conf=2/3=0.67
- I2⇒ I3 Conf=2/3=0.67
- I3⇒ I2 Conf=2/3=0.67
- I2⇒ I4 Conf=2/3=0.67
- I4⇒ I2 Conf=2/3=0.67
- I3⇒ I4 Conf=3/3=1
- I4⇒ I3 Conf=3/3=1

Suppose min_cof_count = 1

$$\text{confidence}(A \Rightarrow B) = \text{Prob}(B \mid A) = \frac{\text{support}(AB)}{\text{support}(A)}$$

Let's go to the next …

数据仓库与数据挖掘
Data Warehouse & Data Mining

宋 杰
Song Jie
东北大学 软件学院
Software College, Northeastern University

8. Data Mining (part two)

Classification and Prediction

Cluster Analysis

Classification vs. Prediction

- Classification
  - Predicts categorical class labels, classifies data based on the training set and the values (class labels) in a classifying attribute and uses it in classifying new data
- Prediction
  - Models continuous-valued functions, i.e., predicts unknown or missing values

- In some machine learning literature, both predicting categorical class labels and modeling continuous-valued functions are called prediction, where the former is called *classification* and the latter is called *regression estimation*

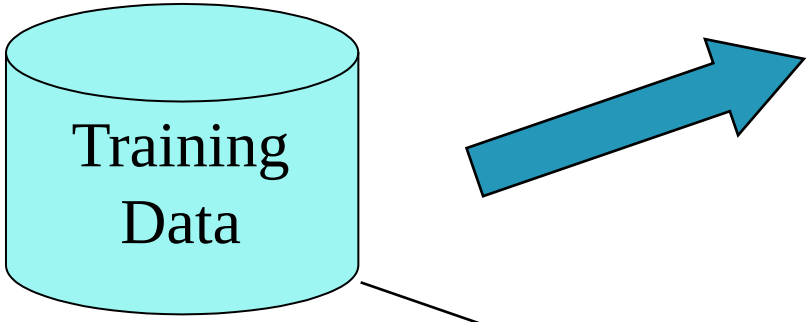
Applications

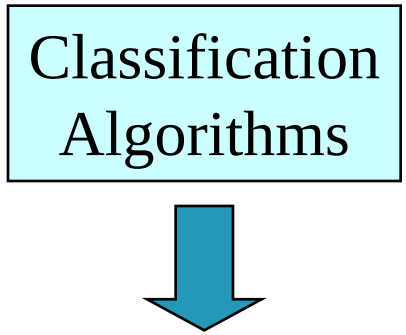
- Typical applications
  - Credit approval
  - Target marketing
  - Medical diagnosis
  - Fraud detection

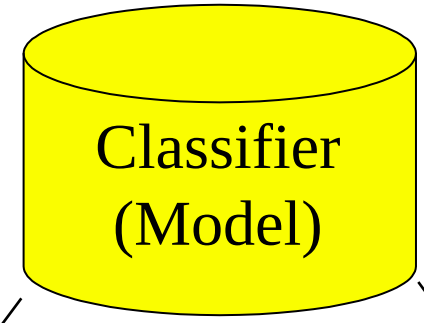
Classification—A Two-Step Process

- Model construction: describing a set of predetermined classes
  - Each sample is assumed to belong to a predefined class, as determined by the class label attribute
  - The data set used for model construction is training set
  - The model is represented as classification rules, decision trees, or mathematical formulae
- Model usage: for classifying future or unknown objects
  - Estimate accuracy of the model
    - The known label of test sample is compared with the classified result from the model
    - Accuracy rate is the percentage of test set samples that are correctly classified by the model
  - If the accuracy is acceptable, use the model to classify data tuples whose class labels are not known

Process (1): Model Construction



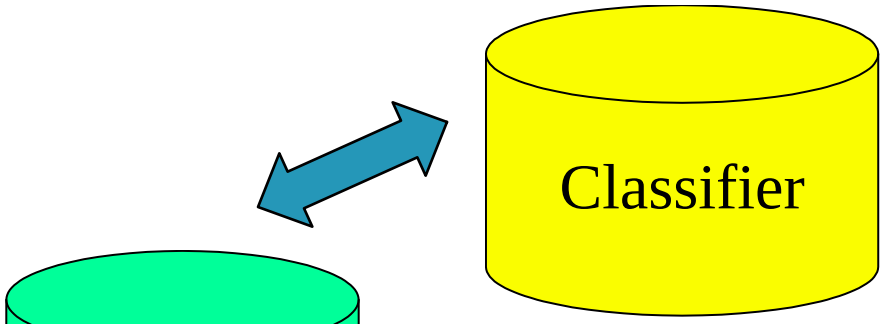


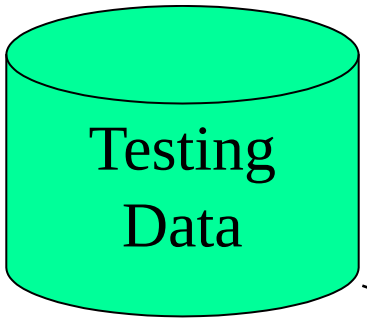


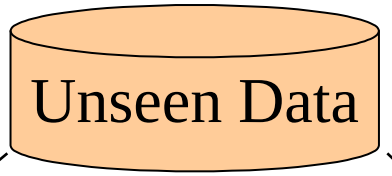
NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

IF rank = ‘professor’
OR years > 6
THEN tenured = ‘yes’

Process (2): Using the Model in Prediction



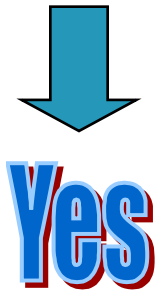




(Jeff, Professor, 4)

NAME	RANK	YEARS	TENURED
Tom	Assistant Prof	2	no
Merlisa	Associate Prof	7	no
George	Professor	5	yes
Joseph	Assistant Prof	7	yes

Tenured?



Supervised vs. Unsupervised Learning

- Supervised learning (classification)
  - Supervision: The training data (observations, measurements, etc.) are accompanied by labels indicating the class of the observations
  - New data is classified based on the training set
- Unsupervised learning (clustering)
  - The class labels of training data is unknown
  - Given a set of measurements, observations, etc. with the aim of establishing the existence of classes or clusters in the data

Issues: Evaluating Classification Methods

- Accuracy
  - classifier accuracy: predicting class label
  - predictor accuracy: guessing value of predicted attributes
- Speed
  - time to construct the model (training time)
  - time to use the model (classification/prediction time)
- Robustness: handling noise and missing values
- Interpretability
  - understanding and insight provided by the model
- Other measures, e.g., goodness of rules, such as decision tree size or compactness of classification rules

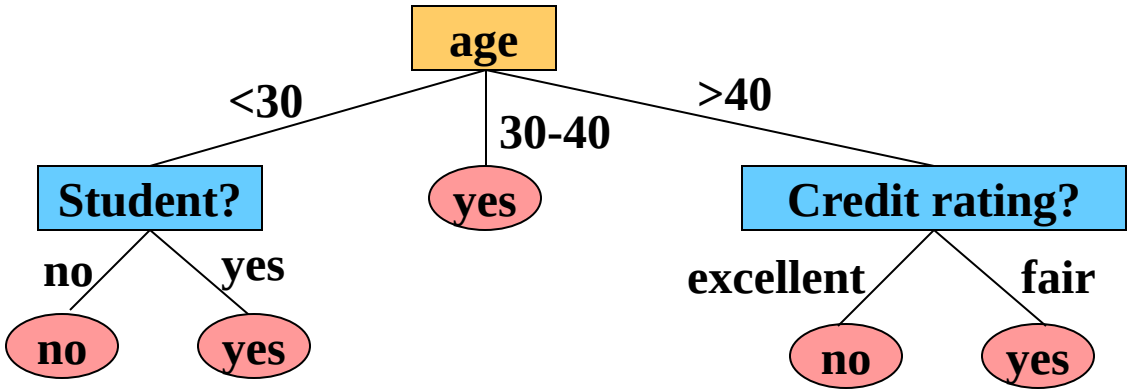
How to estimate accuracy?

- Two popular methods:
  - Hold-out
    - Partition the data set into two independent subsets, i.e. a training set and a test set.
    - Usually 2/3 of the data set are used for training while the rest 1/3 are used for test.
  - K-fold cross-validation
    - Partition the data set into *k* mutually exclusive subsets with approximately equal size.
    - Perform training and test for *k* times. In *i-th* time, the *i-th* subset is used for test while the rest subsets are collectively used for training
    - 10-fold cross-validation is often used

What is decision tree?

Decision tree is a flow-chart-like tree structure, where each *internal node* denotes a test on an attribute, each *branch* represents an outcome of the test, and each *leaf* represents a class or class distribution

example



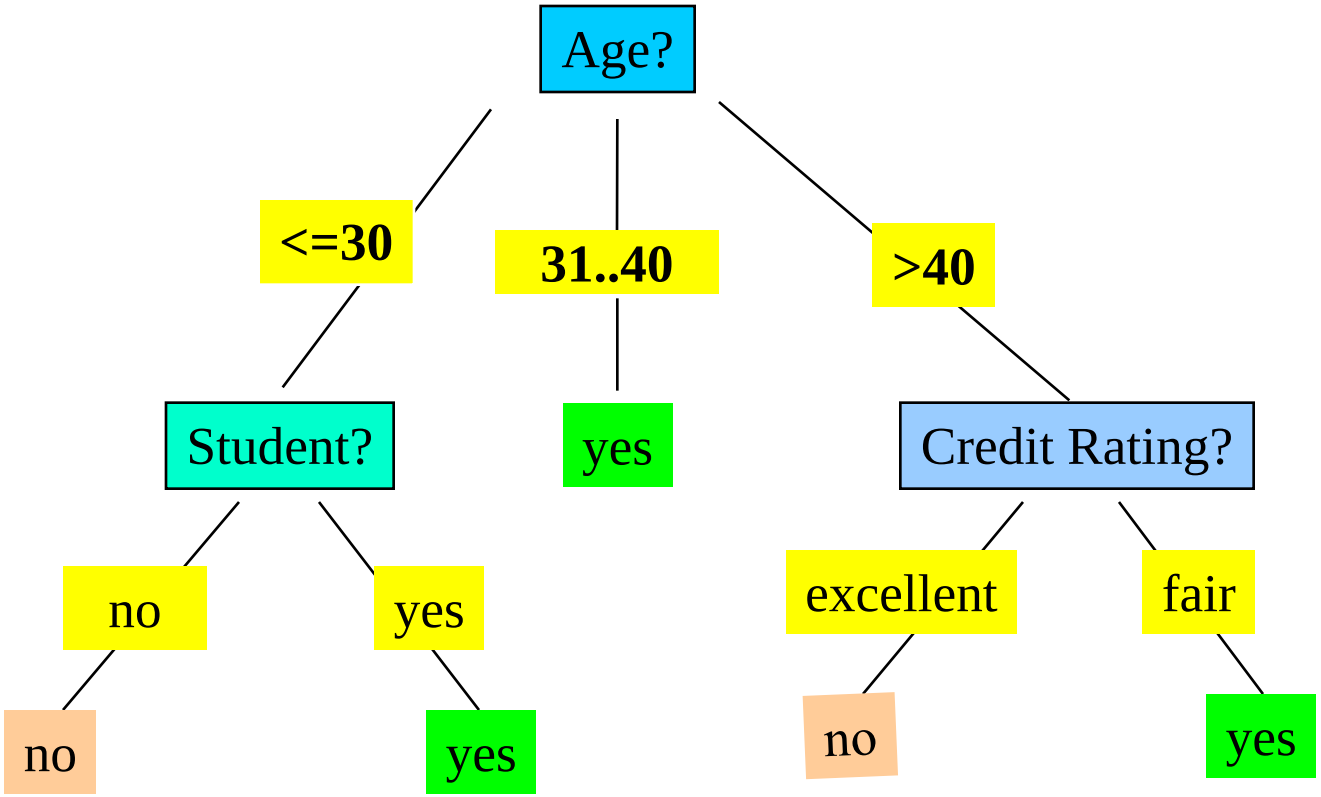
the topmost node in a tree is the root node

in order to classify an unseen instance, the attribute values of the instance are tested against the decision tree. A path is traced from the root to a leaf which holds the class prediction for the instance

Decision Tree Induction: Training Dataset

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Output: A Decision Tree for "*buys_computer*"



Algorithm for ID3 (Decision Tree Induction)

- Basic algorithm (ID3, a greedy algorithm)
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - At start, all the training examples are at the root
  - Attributes are categorical (if continuous-valued, they are discretized in advance)
  - Examples are partitioned recursively based on selected attributes
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)
- Conditions for stopping partitioning
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf
  - There are no samples left

Information gain

S: training set

S_i: training instances of class C_i (i = 1, …, m)

a_j: values of attribute A (j = 1, …, v)

- **the information needed to correctly classify all the instances**

$$I(S_1, S_2, \ldots, S_m) = -\sum_{i=1}^{m} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$

suppose attribute A is selected to partition the training set into the subsets $\{S'_1, S'_2, \ldots, S'_v\}$, then the entropy of A, i.e. the information needed to correctly classify all the instances in those subsets is

$$Ent(A) = \sum_{j=1}^{v} \frac{|S'_j|}{|S|} \left[-\sum_{i=1}^{m} \frac{|S'_{ij}|}{|S'_j|} \log_2 \frac{|S'_{ij}|}{|S'_j|} \right]$$

where S'_{ij} is the instances of class C_i that are covered by S'_j

then the information gain of selecting A is

$$Gain(A) = I(S_1, S_2, \ldots, S_m) - Ent(A)$$

the bigger the information gain, the better the attribute A

Example of information gain (1)

Training set:

S_1: *buys_computer* = "yes", S_2: *buys_computer* = "no"

rid	age	income	student	Credit-rating	Class: buys_computer
1	<30	High	No	Fair	No
2	<30	High	No	Excellent	No
3	30-40	High	No	Fair	Yes
4	>40	Medium	No	Fair	Yes
5	>40	Low	Yes	Fair	Yes
6	>40	Low	Yes	Excellent	No
7	30-40	Low	Yes	Excellent	yes
8	<30	Medium	No	Fair	No
9	<30	Low	Yes	Fair	Yes
10	>40	Medium	Yes	Fair	Yes
11	<30	Medium	Yes	Excellent	Yes
12	30-40	Medium	No	Excellent	Yes
13	30-40	High	Yes	Fair	Yes
14	>40	medium	No	Excellent	No

Example of information gain (2)

rid	age	income	student	Credit-rating	Class: buys_computer
1	<30	High	No	Fair	No
2	<30	High	No	Excellent	No
3	30-40	High	No	Fair	Yes
4	>40	Medium	No	Fair	Yes
5	>40	Low	Yes	Fair	Yes
6	>40	Low	Yes	Excellent	No
7	30-40	Low	Yes	Excellent	yes
8	<30	Medium	No	Fair	No
9	<30	Low	Yes	Fair	Yes
10	>40	Medium	Yes	Fair	Yes
11	<30	Medium	Yes	Excellent	Yes
12	30-40	Medium	No	Excellent	Yes
13	30-40	High	Yes	Fair	Yes
14	>40	medium	No	Excellent	No

the information needed to correctly classify the training set is

$$I(S_1, S_2) = I(9,5) = -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14} = 0.940$$

Example of information gain (3)

rid	age	income	student	Credit-rating	Class: buys_computer
1	<30	High	No	Fair	No
2	<30	High	No	Excellent	No
3	30-40	High	No	Fair	Yes
4	>40	Medium	No	Fair	Yes
5	>40	Low	Yes	Fair	Yes
6	>40	Low	Yes	Excellent	No
7	30-40	Low	Yes	Excellent	yes
8	<30	Medium	No	Fair	No
9	<30	Low	Yes	Fair	Yes
10	>40	Medium	Yes	Fair	Yes
11	<30	Medium	Yes	Excellent	Yes
12	30-40	Medium	No	Excellent	Yes
13	30-40	High	Yes	Fair	Yes
14	>40	medium	No	Excellent	No

suppose attribute *age* is selected to partition the training set

for *age*="<30"

S_{11}=2, S_{21}=3

$$I(S_{11}, S_{21}) = -\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5} = 0.971$$

Example of information gain (4)

The information needed to correctly classify the training set is

$$I(S_1, S_2) = I(9,5) = -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14} = 0.940$$

Suppose attribute *age* is selected to partition the training set

for *age*="<30"
S_{11}=2, S_{21}=3

$$I(S_{11}, S_{21}) = -\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5} = 0.971$$

for *age*="30-40"
S_{12}=4, S_{22}=0

$$I(S_{12}, S_{22}) = 0$$

for *age*=">40"
S_{13}=3, S_{23}=2

$$I(S_{13}, S_{23}) = -\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5} = 0.971$$

Then the entropy of *age* is

$$E(age) = \frac{5}{14}I(S_{11}, S_{21}) + \frac{4}{14}I(S_{12}, S_{22}) + \frac{5}{14}I(S_{13}, S_{23}) = 0.694$$

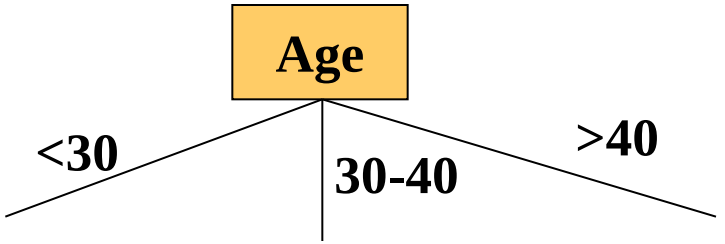
Example of information gain (5)

Then the information gain of selecting *age* is

$$Gain(age) = I(S_1, S_2) - E(age) = 0.940 - 0.694 = 0.246$$

We can also get the information gain of selecting other attributes:

$$Gain(income) = 0.029$$
$$Gain(student) = 0.151$$
$$Gain(credit - rating) = 0.048$$

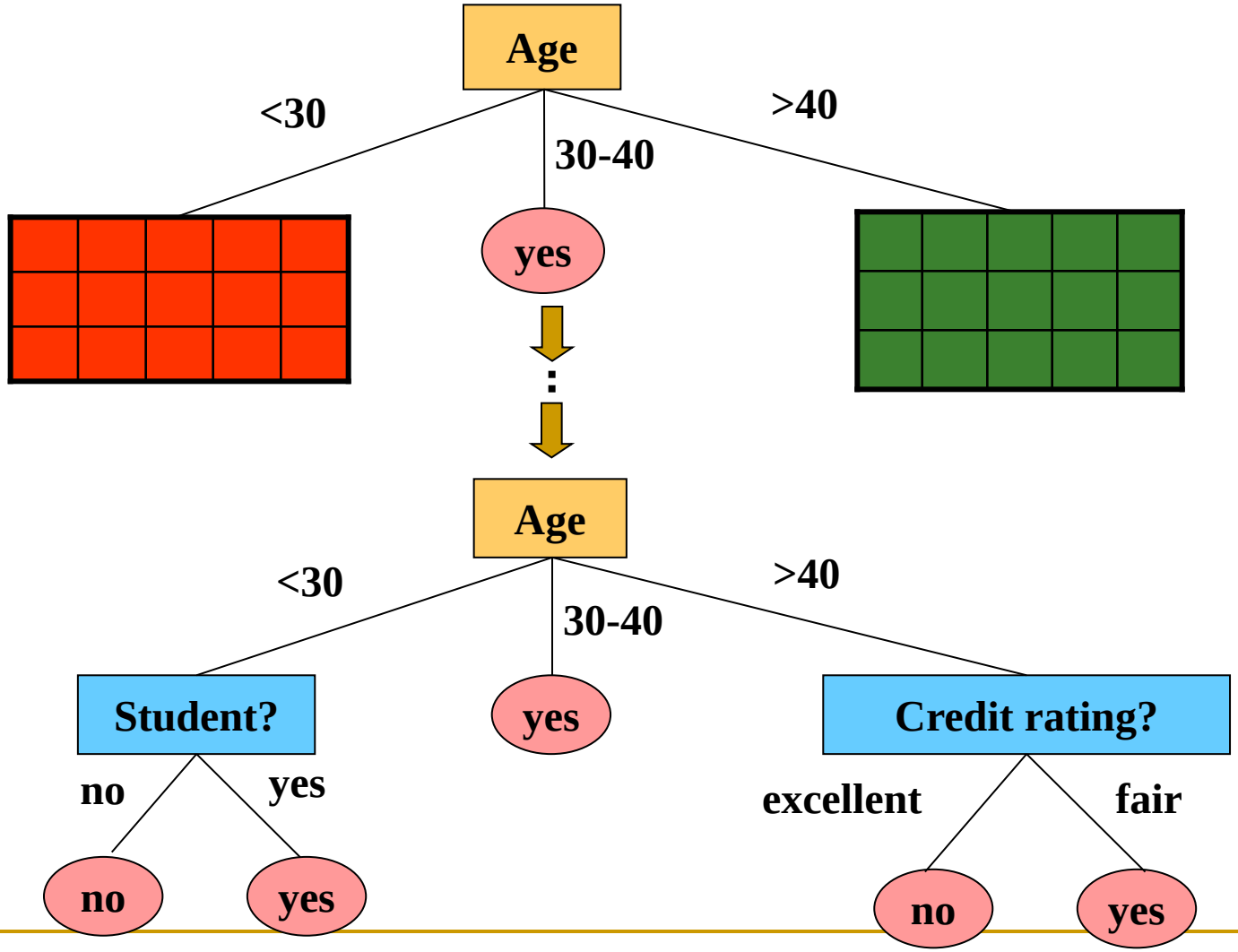


Since *age* has the highest information gain, it is selected as the split for partitioning the training set

Example of information gain (6)

rid	age	income	student	Credit-rating	Class: buys_computer
1	<30	High	No	Fair	No
2	<30	High	No	Excellent	No
3	30-40	High	No	Fair	Yes
4	>40	Medium	No	Fair	Yes
5	>40	Low	Yes	Fair	Yes
6	>40	Low	Yes	Excellent	No
7	30-40	Low	Yes	Excellent	yes
8	<30	Medium	No	Fair	No
9	<30	Low	Yes	Fair	Yes
10	>40	Medium	Yes	Fair	Yes
11	<30	Medium	Yes	Excellent	Yes
12	30-40	Medium	No	Excellent	Yes
13	30-40	High	Yes	Fair	Yes
14	>40	medium	No	Excellent	No

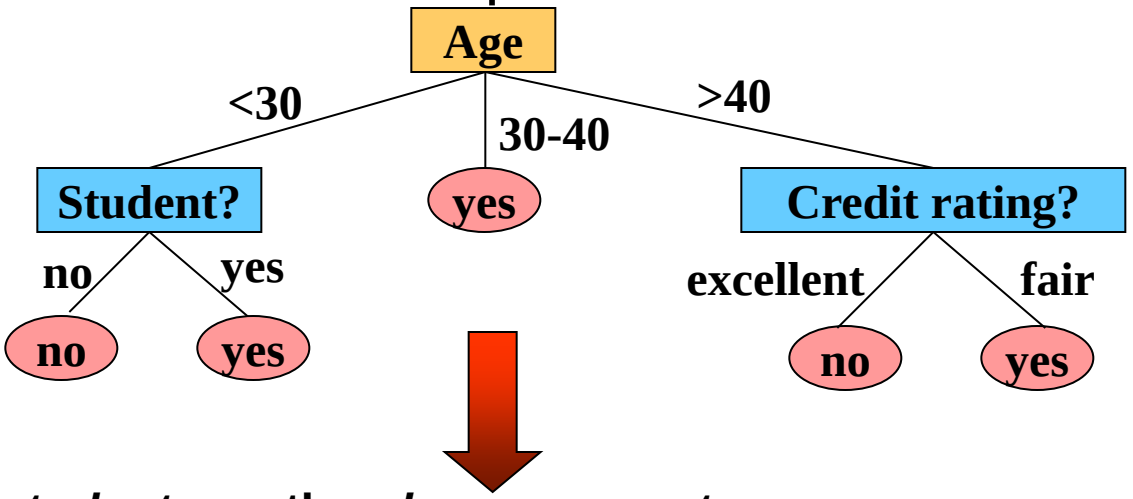
Example of information gain (7)



Map Decision Tree To Rules

- A rule is created for each path from the root to a leaf. Each attribute-value pair along a given path forms a conjunction in the rule antecedent. The class label held by the leaf forms the rule consequent

example



- **If *age*="<30" and *student=no* then *buys_computer=no***
- **If *age*="<30" and *student=yes* then *buys_computer=yes***
- **If *age*="30-40" then *buys_computer=yes***
- **If *age*=">40" and *credit_rating*="excellent" then *buys_computer=no***
- **If *age*=">40" and *credit_rating*="fair" then *buys_computer=yes***

Bayesian Theorem: Basics

- Let *X* be a data sample ("*evidence*"): class label is unknown
- Let *H* be a *hypothesis* that *X* belongs to class *C*
- Classification is to determine P(*H*|*X*), the probability that the hypothesis holds given the observed data sample *X*
- P(*H*) (*prior probability*), the initial probability
  - E.g., *X* will buy computer, regardless of age, income, ...
- P(*X*): probability that sample data is observed
- P(*X*|*H*) (*posteriori probability*), the probability of observing the sample *X*, given that the hypothesis holds
  - E.g., Given that *X* will buy computer, the prob. that *X* is 31..40, medium income

Bayesian Theorem

- Given training data *X*, *posteriori probability of a hypothesis H*, P(*H*|*X*), follows the Bayes theorem

$$P(H \mid \mathbf{X}) = \frac{P(\mathbf{X} \mid H)P(H)}{P(\mathbf{X})}$$

- Predicts *X* belongs to C_i iff the probability P(C_i|*X*) is the highest among all the P(C_k|*X*) for all the *k* classes

Towards Naïve Bayesian Classifier

- Let *D* be a training set of tuples and their associated class labels, and each tuple is represented by an n-*D* attribute vector $X = (x_1, x_2, ..., x_n)$
- Suppose there are *m* classes $C_1, C_2, ..., C_m$.
- Classification is to derive the maximum posteriori, i.e., the maximal $P(C_i|X)$
- This can be derived from Bayes' theorem

$$P(C_i|\mathbf{X}) = \frac{P(\mathbf{X}|C_i)P(C_i)}{P(\mathbf{X})}$$

- Since $P(X)$ is constant for all classes, only

$$P(C_i|\mathbf{X}) = P(\mathbf{X}|C_i)P(C_i)$$

needs to be maximized

Derivation of Naïve Bayes Classifier

- A simplified assumption: attributes are conditionally independent (i.e., no dependence relation between attributes):

$$P(\mathbf{X}|C_i) = \prod_{k=1}^{n} P(x_k|C_i) = P(x_1|C_i) \times P(x_2|C_i) \times ... \times P(x_n|C_i)$$

Naïve Bayesian Classifier: Training Dataset

Class:
C1:buys_computer = 'yes'
C2:buys_computer = 'no'

Data sample
X = (age <=30,
Income = medium,
Student = yes
Credit_rating = Fair)

age	income	student	redit_ratin	s_compu
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Naïve Bayesian Classifier: An Example

- $P(C_i)$: P(buys_computer = "yes") = 9/14 = 0.643
  P(buys_computer = "no") = 5/14= 0.357

- Compute $P(X|C_i)$ for each class
  P(age = "<=30" | buys_computer = "yes") = 2/9 = 0.222
  P(age = "<= 30" | buys_computer = "no") = 3/5 = 0.6
  P(income = "medium" | buys_computer = "yes") = 4/9 = 0.444
  P(income = "medium" | buys_computer = "no") = 2/5 = 0.4
  P(student = "yes" | buys_computer = "yes) = 6/9 = 0.667
  P(student = "yes" | buys_computer = "no") = 1/5 = 0.2
  P(credit_rating = "fair" | buys_computer = "yes") = 6/9 = 0.667
  P(credit_rating = "fair" | buys_computer = "no") = 2/5 = 0.4

- **X = (age <= 30 , income = medium, student = yes, credit_rating = fair)**

$P(X|C_i)$: P(X|buys_computer = "yes") = 0.222 x 0.444 x 0.667 x 0.667 = 0.044
P(X|buys_computer = "no") = 0.6 x 0.4 x 0.2 x 0.4 = 0.019

$P(X|C_i)*P(C_i)$: P(X|buys_computer = "yes") * P(buys_computer = "yes") = 0.028
P(X|buys_computer = "no") * P(buys_computer = "no") = 0.007

Therefore, X belongs to class ("buys_computer = yes")

Avoiding the 0-Probability Problem

- Naïve Bayesian prediction requires each conditional prob. be non-zero. Otherwise, the predicted prob. will be zero

$$P(X \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i)$$

- Ex. Suppose a dataset with 1000 tuples, income=low (0), income= medium (990), and income = high (10),
- Use Laplacian correction (or Laplacian estimator)
  - Adding 1 to each case

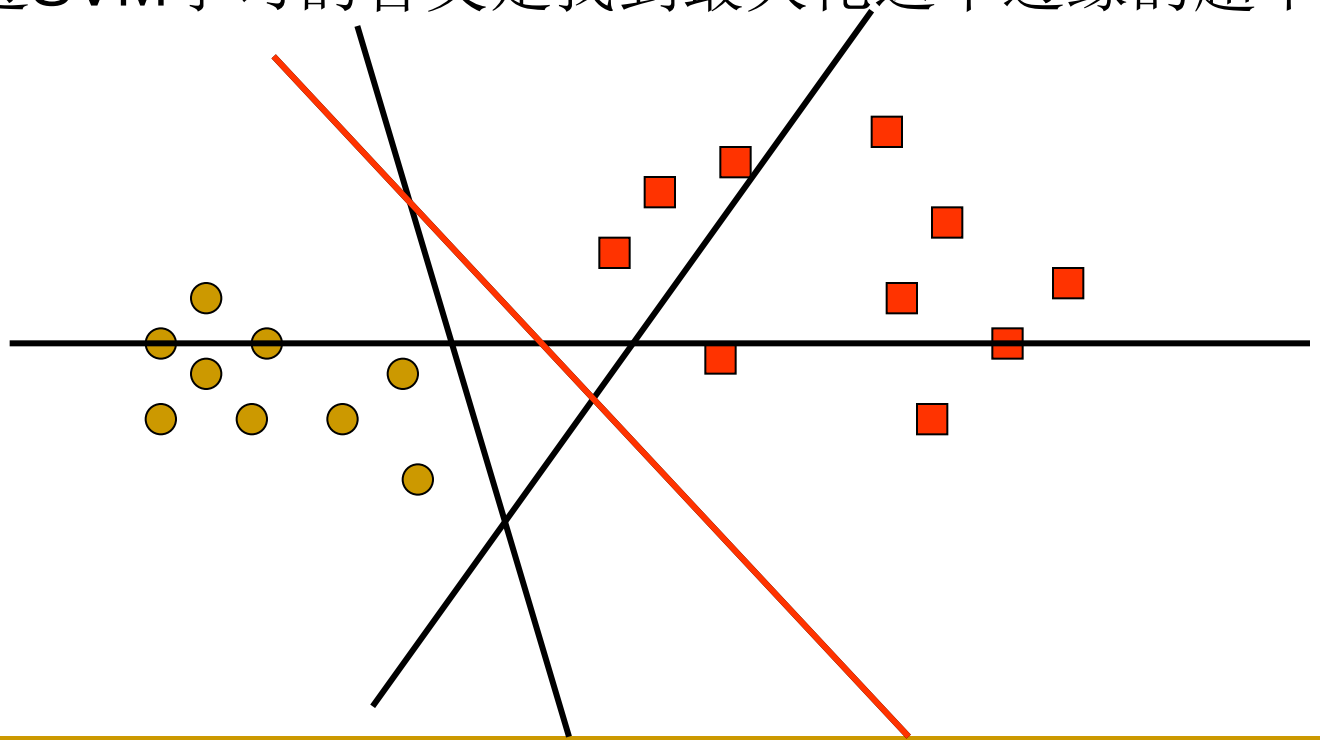
    Prob(income = low) = 1/1003

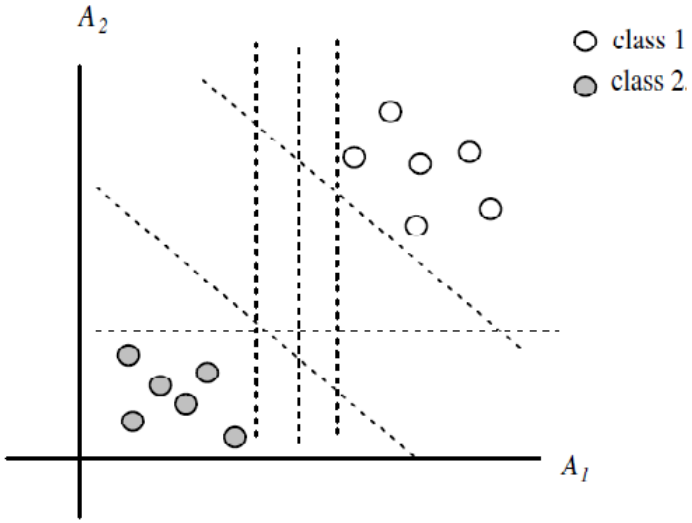
    Prob(income = medium) = 991/1003

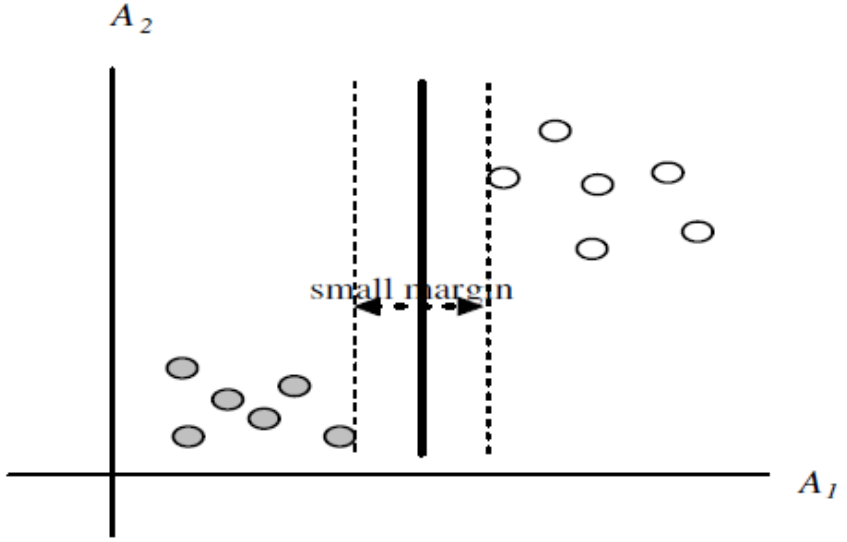
    Prob(income = high) = 11/1003
  - The "corrected " prob. estimates are close to their "uncorrected " counterparts

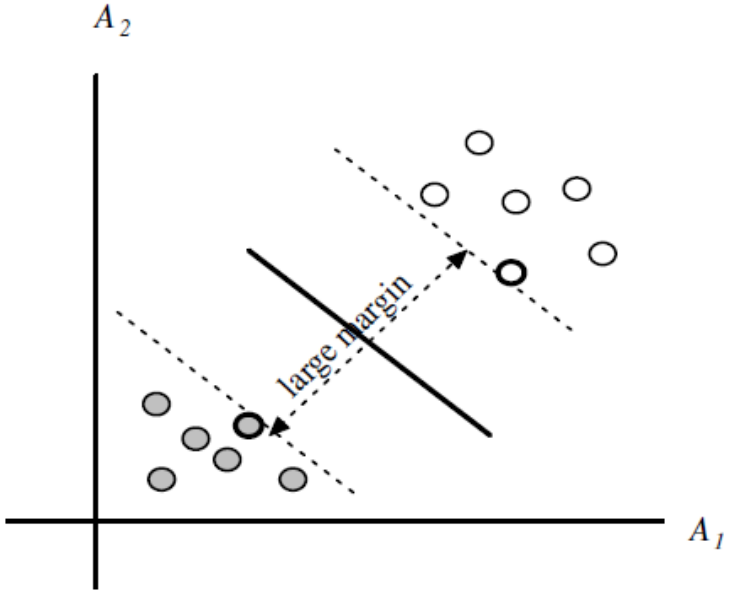
支持向量机（SVM）

- SVM构建了一个分割两类的超平面，在构建过程中，SVM算法试图使两类之间的分隔达到最大化。
- 和分类器平面平行、分别穿过数据集中的一个或多个点的两个平面称为边界平面，这些边界平面的距离称为边缘。通过SVM学习的含义是找到最大化这个边缘的超平面




A_2
class 1
class 2
A_1


A_2
small margin
A_1

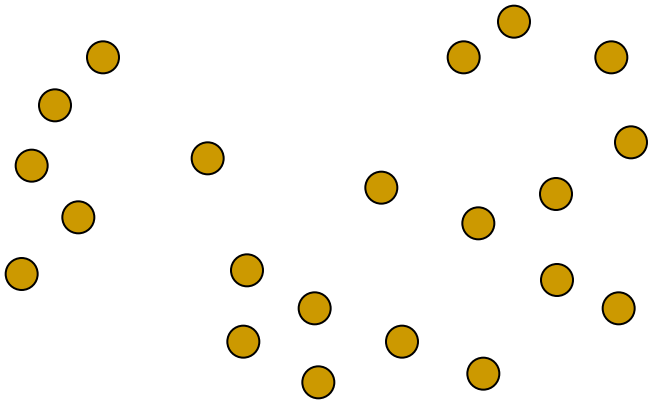

A_2
large margin
A_1

What is Cluster Analysis?

- Cluster: a collection of data objects
  - Similar to one another within the same cluster
  - Dissimilar to the objects in other clusters
- Cluster analysis
  - Finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters
- Unsupervised learning: no predefined classes

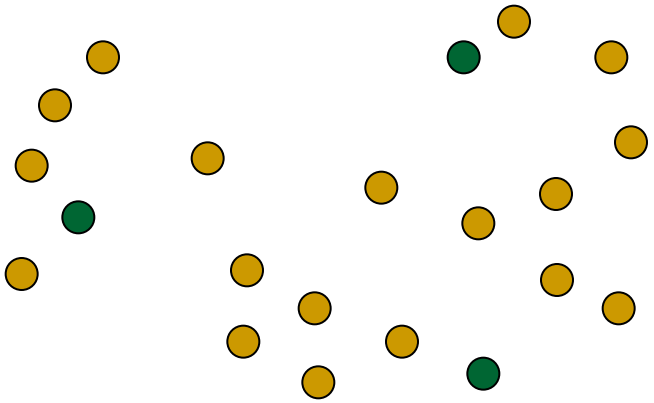
k-means method

- Each cluster is represented by the mean value of the data in the cluster
  - Step1: Randomly select *k* objects as the centers of the clusters
  - Step2: For each remaining object, assign it to the cluster whose center is the nearest to the object
  - Step3: Compute the new mean for each cluster
  - Step4: If there is no change, exit. Otherwise go to Step2



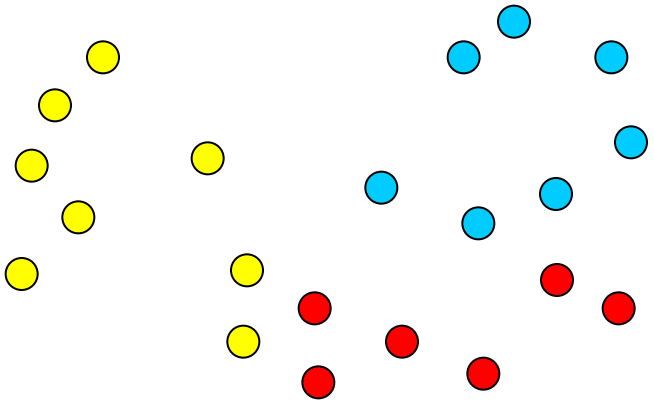
k-means method

- Each cluster is represented by the mean value of the data in the cluster
  - Step1: Randomly select k objects as the centers of the clusters
  - Step2: For each remaining object, assign it to the cluster whose center is the nearest to the object
  - Step3: Compute the new mean for each cluster
  - Step4: If there is no change, exit. Otherwise go to Step2



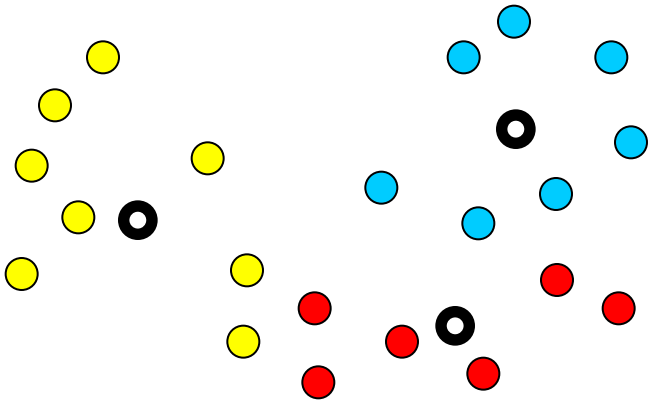
k-means method

- Each cluster is represented by the mean value of the data in the cluster
  - Step1: Randomly select k objects as the centers of the clusters
  - Step2: For each remaining object, assign it to the cluster whose center is the nearest to the object
  - Step3: Compute the new mean for each cluster
  - Step4: If there is no change, exit. Otherwise go to Step2



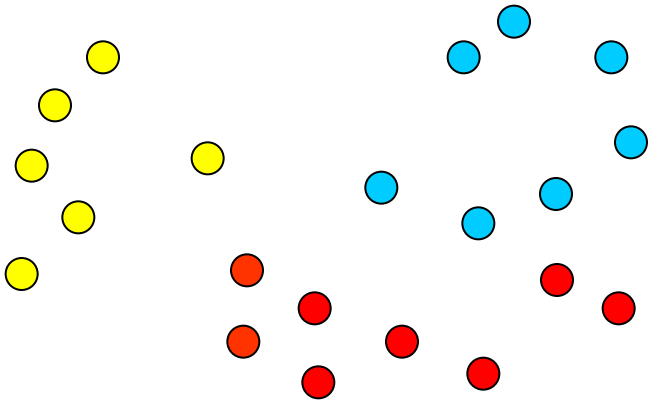
k-means method

- Each cluster is represented by the mean value of the data in the cluster
  - Step1: Randomly select k objects as the centers of the clusters
  - Step2: For each remaining object, assign it to the cluster whose center is the nearest to the object
  - Step3: Compute the new mean for each cluster
  - Step4: If there is no change, exit. Otherwise go to Step2



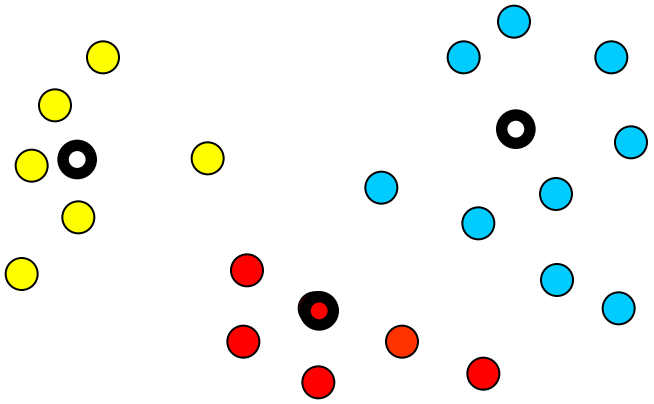
k-means method

- Each cluster is represented by the mean value of the data in the cluster
  - Step1: Randomly select k objects as the centers of the clusters
  - Step2: For each remaining object, assign it to the cluster whose center is the nearest to the object
  - Step3: Compute the new mean for each cluster
  - Step4: If there is no change, exit. Otherwise go to Step2



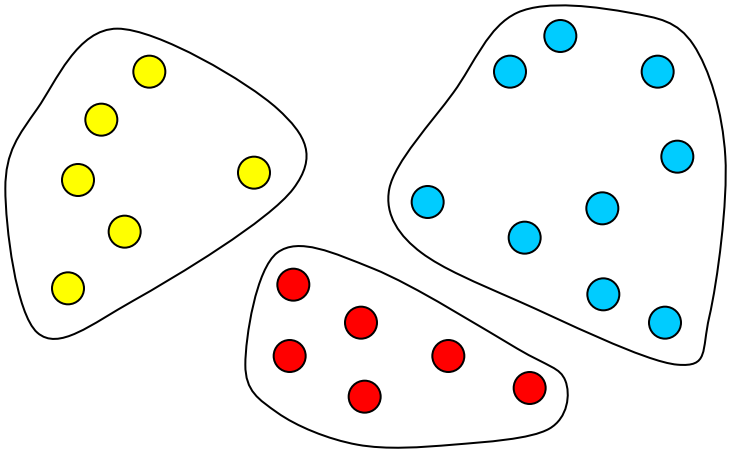
k-means method

- Each cluster is represented by the mean value of the data in the cluster
  - Step1: Randomly select k objects as the centers of the clusters
  - Step2: For each remaining object, assign it to the cluster whose center is the nearest to the object
  - Step3: Compute the new mean for each cluster
  - Step4: If there is no change, exit. Otherwise go to Step2



k-means method

- Each cluster is represented by the mean value of the data in the cluster
  - Step1: Randomly select k objects as the centers of the clusters
  - Step2: For each remaining object, assign it to the cluster whose center is the nearest to the object
  - Step3: Compute the new mean for each cluster
  - Step4: If there is no change, exit. Otherwise go to Step2



Similarity and Dissimilarity Between Objects

- Distances are normally used to measure the similarity or dissimilarity between two data objects
- Some popular ones include: *Minkowski distance*:

$$d(i,j) = \sqrt[q]{(|x_{i1} - x_{j1}|^q + |x_{i2} - x_{j2}|^q + ... + |x_{ip} - x_{jp}|^q)}$$

where $i = (x_{i1}, x_{i2}, ..., x_{ip})$ and $j = (x_{j1}, x_{j2}, ..., x_{jp})$ are two p-dimensional data objects, and q is a positive integer

- If $q = 1$, d is Manhattan distance

$$d(i,j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + ... + |x_{ip} - x_{jp}|$$

Similarity and Dissimilarity Between Objects (Cont.)

- *If q = 2*, *d* is Euclidean distance:

$$d(i,j) = \sqrt{(|x_{i_1} - x_{j_1}|^2 + |x_{i_2} - x_{j_2}|^2 + ... + |x_{i_p} - x_{j_p}|^2)}$$

  - Properties
    - $d(i,j) \geq 0$
    - $d(i,i) = 0$
    - $d(i,j) = d(j,i)$
    - $d(i,j) \leq d(i,k) + d(k,j)$
- Also, one can use weighted distance, parametric Pearson product moment correlation, or other disimilarity measures

Binary Variables

- A contingency table for binary data
- Distance measure for symmetric binary variables:
- Distance measure for asymmetric binary variables:
- Jaccard coefficient (*similarity* measure for *asymmetric* binary variables):

		Object *j*		
		1	0	*sum*
Object *i*	1	a	b	$a+b$
	0	c	d	$c+d$
	sum	$a+c$	$b+d$	p

$$d(i,j) = \frac{b+c}{a+b+c+d}$$

$$d(i,j) = \frac{b+c}{a+b+c}$$

$$sim_{Jaccard}(i,j) = \frac{a}{a+b+c}$$

Dissimilarity between Binary Variables

- Example

Name	Gender	Fever	Cough	Test-1	Test-2	Test-3	Test-4
Jack	M	Y	N	P	N	N	N
Mary	F	Y	N	P	N	P	N
Jim	M	Y	P	N	N	N	N

- Gender is a symmetric attribute
- The remaining attributes are asymmetric binary
- Let the values *Y* and *P* be set to 1, and the value *N* be set to 0

$$d(jack, mary) = \frac{0+1}{2+0+1} = 0.33$$

$$d(jack, jim) = \frac{1+1}{1+1+1} = 0.67$$

$$d(jim, mary) = \frac{1+2}{1+1+2} = 0.75$$

Nominal Variables

- A generalization of the binary variable in that it can take more than 2 states, e.g., red, yellow, blue, green
- Method 1: Simple matching
  - *m*: number of matches, *p*: total number of variables

$$d(i, j) = \frac{p - m}{p}$$

- Method 2: use a large number of binary variables
  - creating a new binary variable for each of the *M* nominal states

Attributes of Mixed Types

- Transform the dissimilar measures of different kinds of attributes to a same range, e.g. [0, 1], then combine the dissimilar measures

$$d(i, j) = \sum_{k} w_k d_{ij}^{A_k}$$

- Where w_k is the weight of attribute A_k, $d_{ij}^{A_k}$ is the dissimilarity of the *i-th* object and the *j-th* object on attribute A_k. $d_{ij}^{A_k}$ has already been normalized to [0, 1]

Vector Objects

- Vector objects: keywords in documents, gene features in micro-arrays, etc.
- Broad applications: information retrieval, biologic taxonomy, etc.
- Cosine measure

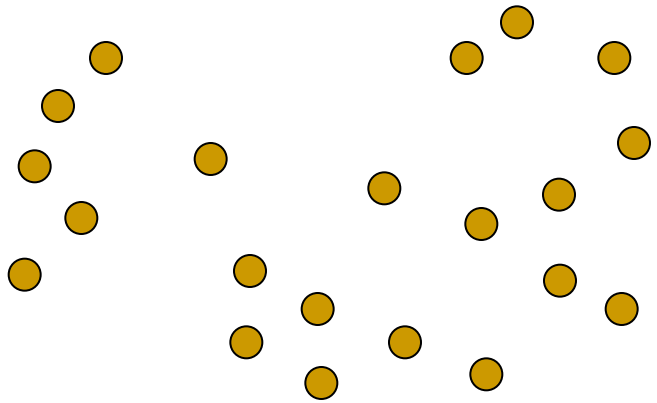
$$s(\vec{X}, \vec{Y}) = \frac{\vec{X}^t \cdot \vec{Y}}{|\vec{X}||\vec{Y}|},$$

k-medoids method

- Medoids are representative objects of a data set or a cluster with a data set whose average dissimilarity to all the objects in the cluster is minimal

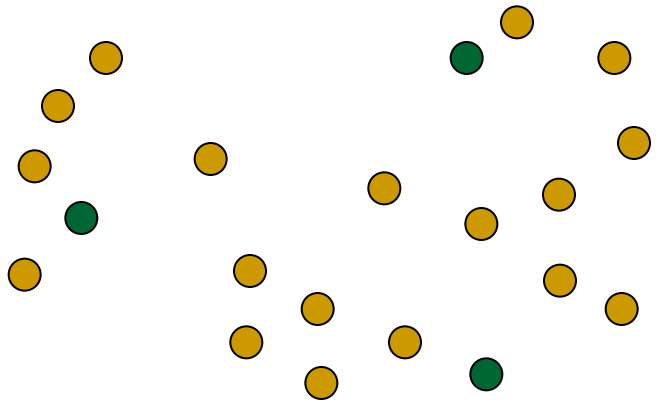
k-medoids method

- Each cluster is represented by one of the objects near the center of the cluster
  - Step1: randomly select k objects as the medoids of the clusters
  - Step2: for each remaining object, assign it to the cluster whose mediod is the nearest to the object
  - Step3: if the quality of the clustering can be improved by swapping a nonmedoid with a medoid, swapping them and then go to Step 2; otherwise exit



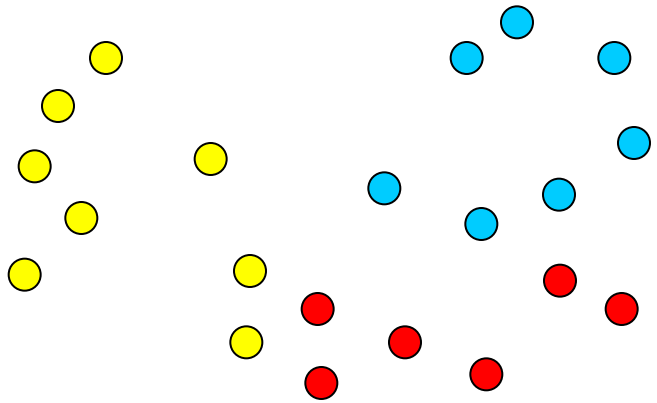
k-medoids method

- Each cluster is represented by one of the objects near the center of the cluster
  - Step1: randomly select k objects as the medoids of the clusters
  - Step2: for each remaining object, assign it to the cluster whose mediod is the nearest to the object
  - Step3: if the quality of the clustering can be improved by swapping a nonmedoid with a medoid, swapping them and then go to Step 2; otherwise exit



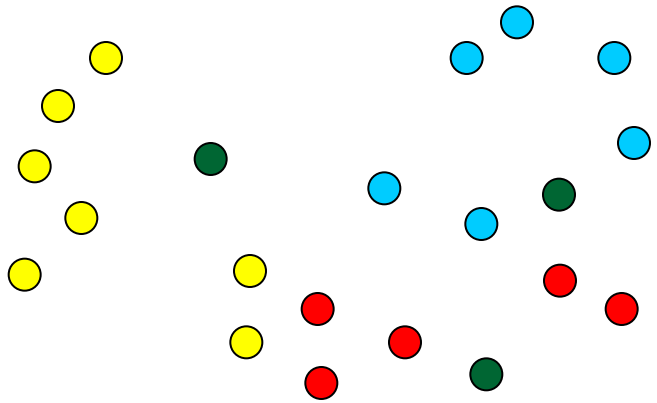
k-medoids method

- Each cluster is represented by one of the objects near the center of the cluster
  - Step1: randomly select k objects as the medoids of the clusters
  - Step2: for each remaining object, assign it to the cluster whose mediod is the nearest to the object
  - Step3: if the quality of the clustering can be improved by swapping a nonmedoid with a medoid, swapping them and then go to Step 2; otherwise exit



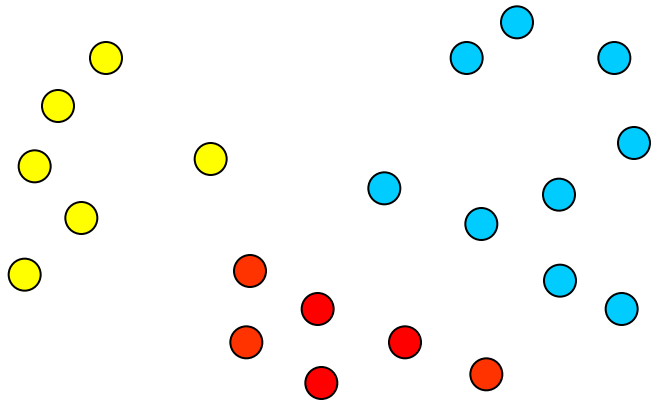
k-medoids method

- Each cluster is represented by one of the objects near the center of the cluster
  - Step1: randomly select k objects as the medoids of the clusters
  - Step2: for each remaining object, assign it to the cluster whose mediod is the nearest to the object
  - Step3: if the quality of the clustering can be improved by swapping a nonmedoid with a medoid, swapping them and then go to Step 2; otherwise exit



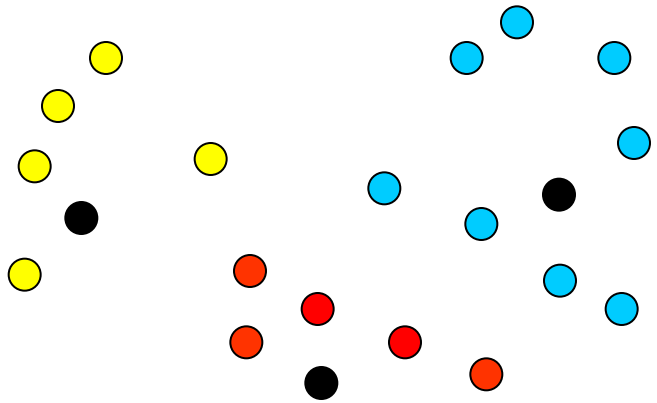
k-medoids method

- Each cluster is represented by one of the objects near the center of the cluster
  - Step1: randomly select k objects as the medoids of the clusters
  - Step2: for each remaining object, assign it to the cluster whose mediod is the nearest to the object
  - Step3: if the quality of the clustering can be improved by swapping a nonmedoid with a medoid, swapping them and then go to Step 2; otherwise exit



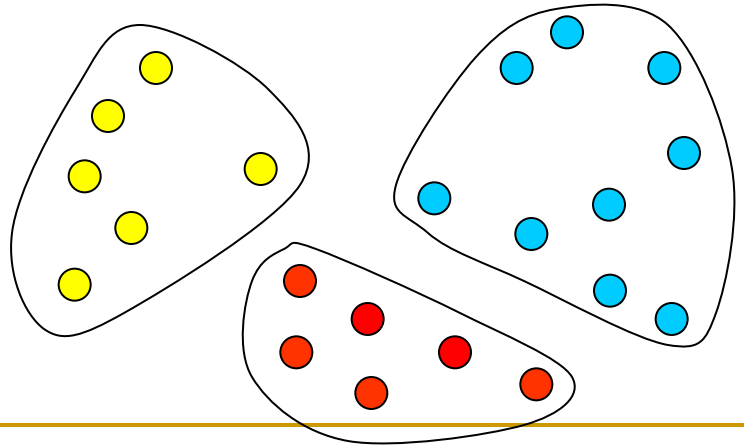
k-medoids method

- Each cluster is represented by one of the objects near the center of the cluster
  - Step1: randomly select k objects as the medoids of the clusters
  - Step2: for each remaining object, assign it to the cluster whose mediod is the nearest to the object
  - Step3: if the quality of the clustering can be improved by swapping a nonmedoid with a medoid, swapping them and then go to Step 2; otherwise exit



k-medoids method

- Each cluster is represented by one of the objects near the center of the cluster
  - Step1: randomly select k objects as the medoids of the clusters
  - Step2: for each remaining object, assign it to the cluster whose mediod is the nearest to the object
  - Step3: if the quality of the clustering can be improved by swapping a nonmedoid with a medoid, swapping them and then go to Step 2; otherwise exit



The End…