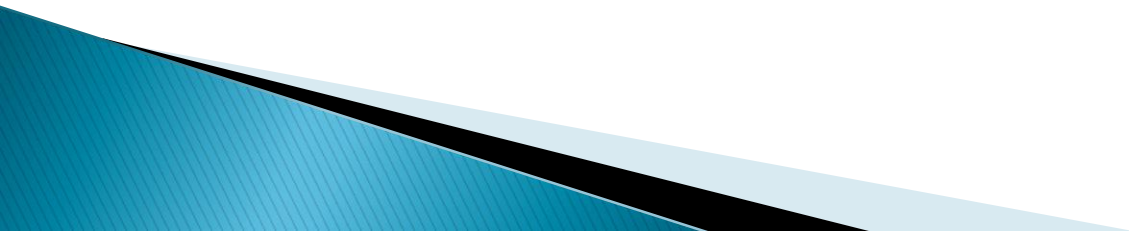


Data Mining, Data ware Housing



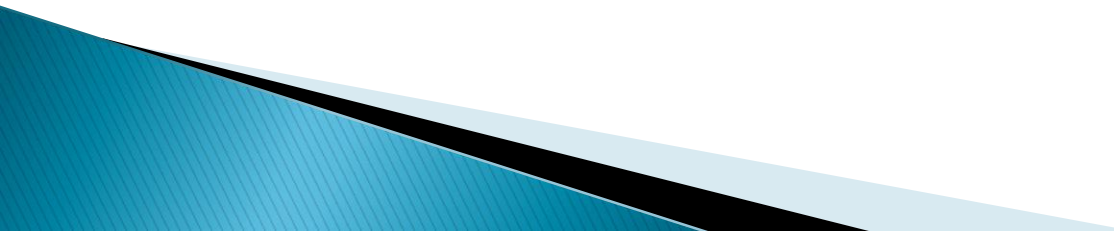
Introduction

- ▶ Data mining refers loosely to the process of semi automatically analyzing large data bases to find useful patterns Data ware house is a repository of information gathered from multiple sources , stored under a unified schema , at a single site

Applications

- ▶ Multimedia Data Mining
 - ▶ Mining Raster Databases
 - ▶ Mining Associations in Multimedia Data
 - ▶ Audio and Video Data Mining
 - ▶ Text Mining
 - ▶ Mining the World Wide Web
- 

Scope of research

- ▶ In data mining we can design Data Mining Models.
 - ▶ Can develop data mining algorithms.
 - ▶ Add privacy and security features in data mining.
 - ▶ Scaling up for high dimensional data and high speed data streams.
- 

Data Analysis and Mining

- ▶ Decision Support Systems
- ▶ Data Analysis and OLAP
- ▶ Data Warehousing
- ▶ Data Mining

Decision Support Systems

- ▶ Decision–support systems are used to make business decisions, often based on data collected by on–line transaction–processing systems.
- ▶ Examples of business decisions:
 - What items to stock?
 - What insurance premium to change?
 - To whom to send advertisements?
- ▶ Examples of data used for making decisions
 - Retail sales transaction details
 - Customer profiles (income, age, gender, etc.)

Decision–Support Systems: Overview

- ▶ **Data analysis** tasks are simplified by specialized tools and SQL extensions
 - Example tasks
 - For each product category and each region, what were the total sales in the last quarter and how do they compare with the same quarter last year
 - As above, for each product category and each customer category
- ▶ **Statistical analysis** packages (e.g., : S++) can be interfaced with databases
 - Statistical analysis is a large field, but not covered here
- ▶ **Data mining** seeks to discover knowledge automatically in the form of statistical rules and patterns from large databases.
- ▶ A **data warehouse** archives information gathered from multiple sources, and stores it under a unified schema, at a single site.
 - Important for large businesses that generate data from multiple divisions, possibly at multiple sites
 - Data may also be purchased externally

Data Analysis and OLAP

- ▶ **Online Analytical Processing (OLAP)**
 - Interactive analysis of data, allowing data to be summarized and viewed in different ways in an online fashion (with negligible delay)
- ▶ Data that can be modeled as dimension attributes and measure attributes are called **multidimensional data**.
 - **Measure attributes**
 - measure some value
 - can be aggregated upon
 - e.g. the attribute *number* of the *sales* relation
 - **Dimension attributes**
 - define the dimensions on which measure attributes (or aggregates thereof) are viewed
 - e.g. the attributes *item_name*, *color*, and *size* of the *sales* relation

Cross Tabulation of *sales* by *item-name* and *color*

size:

		<i>color</i>			
		dark	pastel	white	Total
<i>item-name</i>	skirt	8	35	10	53
	dress	20	10	5	35
	shirt	14	7	28	49
	pant	20	2	5	27
	Total	62	54	48	164

- ▶ The table above is an example of a **cross-tabulation (cross-tab)**, also referred to as a **pivot-table**.
 - Values for one of the dimension attributes form the row headers
 - Values for another dimension attribute form the column headers
 - Other dimension attributes are listed on top
 - Values in individual cells are (aggregates of) the values of the dimension attributes that specify the cell.

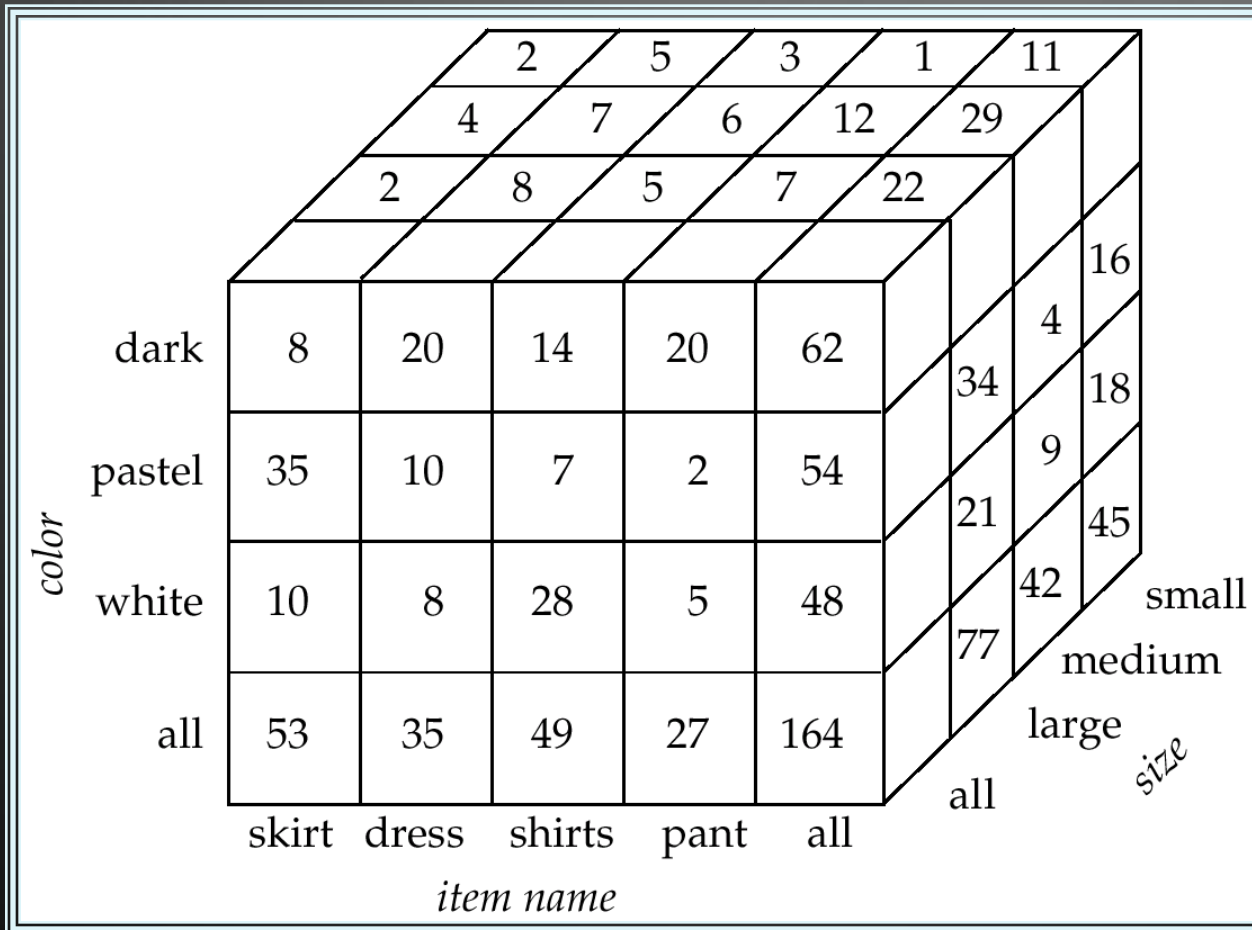
Cross-tabs

- n Cross-tabs can be represented as relations
 - n We use the value **all** is used to represent aggregates
 - n The SQL:1999 standard actually uses null values in place of **all** despite confusion with regular null values

<i>item-name</i>	<i>color</i>	<i>number</i>
skirt	dark	8
skirt	pastel	35
skirt	white	10
skirt	all	53
dress	dark	20
dress	pastel	10
dress	white	5
dress	all	35
shirt	dark	14
shirt	pastel	7
shirt	white	28
shirt	all	49
pant	dark	20
pant	pastel	2
pant	white	5
pant	all	27
all	dark	62
all	pastel	54
all	white	48
all	all	164

Data Cube

- n A **data cube** is a multidimensional generalization of a cross-tab
- n Can have n dimensions; we show 3 below
- n Cross-tabs can be used as views on a data cube



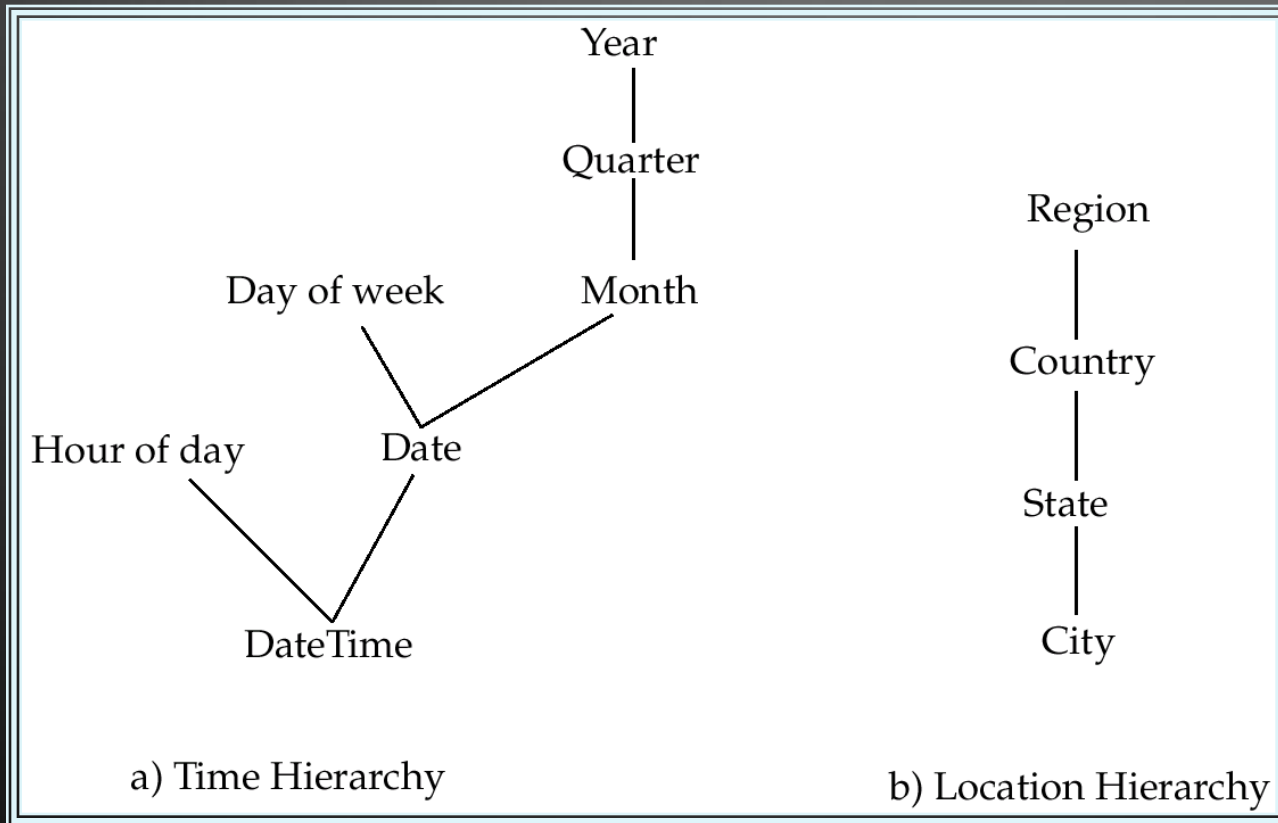
Online Analytical Processing

- ▶ **Pivoting:** changing the dimensions used in a cross-tab is called
- ▶ **Slicing:** creating a cross-tab for fixed values only
 - Sometimes called **dicing**, particularly when values for multiple dimensions are fixed.
- ▶ **Rollup:** moving from finer-granularity data to a coarser granularity
- ▶ **Drill down:** The opposite operation – that of moving from coarser-granularity data to finer-granularity data

Hierarchies on Dimensions

n **Hierarchy** on dimension attributes: lets dimensions to be viewed at different levels of detail

H E.g. the dimension DateTime can be used to aggregate by hour of day, date, day of week, month, quarter or year



Cross Tabulation With Hierarchy

- n Cross-tabs can be easily extended to deal with hierarchies
 - H Can drill down or roll up on a hierarchy

<i>category</i>	<i>item-name</i>	dark	pastel	white	total	
womenswear	skirt	8	8	10	53	
	dress	20	20	5	35	
	subtotal	28	28	15		88
menswear	pants	14	14	28	49	
	shirt	20	20	5	27	
	subtotal	34	34	33		76
total		62	62	48		164

OLAP Implementation

- ▶ The earliest OLAP systems used multidimensional arrays in memory to store data cubes, and are referred to as **multidimensional OLAP (MOLAP)** systems.
- ▶ OLAP implementations using only relational database features are called **relational OLAP (ROLAP)** systems
- ▶ Hybrid systems, which store some summaries in memory and store the base data and other summaries in a relational database, are called **hybrid OLAP (HOLAP)** systems.

OLAP Implementation (Cont.)

- ▶ Early OLAP systems precomputed *all* possible aggregates in order to provide online response
 - Space and time requirements for doing so can be very high
 - 2^n combinations of **group by**
 - It suffices to precompute some aggregates, and compute others on demand from one of the precomputed aggregates
 - Can compute aggregate on *(item-name, color)* from an aggregate on *(item-name, color, size)*
 - For all but a few “non-decomposable” aggregates such as *median*
 - is cheaper than computing it from scratch
- ▶ Several optimizations available for computing multiple aggregates
 - Can compute aggregate on *(item-name, color)* from an aggregate on *(item-name, color, size)*
 - Can compute aggregates on *(item-name, color, size)*, *(item-name, color)* and *(item-name)* using a single sorting of the base data

Extended Aggregation in

SQL:1999

- ▶ The cube operation computes union of **group by**'s on every subset of the specified attributes

- ▶ E.g. consider the query

```
select item-name, color, size, sum(number)  
from sales  
group by cube(item-name, color, size)
```

This computes the union of eight different groupings of the *sales* relation:

```
{ (item-name, color, size), (item-name, color),  
  (item-name, size),      (color, size),  
  (item-name),          (color),  
  (size),              ( ) }
```

where () denotes an empty **group by** list.

- ▶ For each grouping, the result contains the null value for attributes not present in the grouping.

Extended Aggregation (Cont.)

- ▶ Relational representation of cross-tab that we saw earlier, but with *null* in place of *all*, can be computed by

```
select item-name, color, sum(number)  
from sales  
group by cube(item-name, color)
```

- ▶ The function **grouping()** can be applied on an attribute
 - Returns 1 if the value is a null value representing all, and returns 0 in all other cases.

```
select item-name, color, size, sum(number),  
       grouping(item-name) as item-name-flag,  
       grouping(color) as color-flag,  
       grouping(size) as size-flag,  
from sales  
group by cube(item-name, color, size)
```

- ▶ Can use the function **decode()** in the **select** clause to replace such nulls by a value such as *all*
 - E.g. replace *item-name* in first query by
decode(grouping(*item-name*), 1, 'all', *item-name*)

Extended Aggregation (Cont.)

- ▶ The **rollup** construct generates union on every prefix of specified list of attributes

- ▶ E.g.

```
select item-name, color, size, sum(number)  
from sales  
group by rollup(item-name, color, size)
```

Generates union of four groupings:

```
{ (item-name, color, size), (item-name, color), (item-name), (  
) }
```

- ▶ Rollup can be used to generate aggregates at multiple levels of a hierarchy.
- ▶ E.g., suppose table *itemcategory*(*item-name*, *category*) gives the category of each item. Then

```
select category, item-name, sum(number)  
from sales, itemcategory  
where sales.item-name = itemcategory.item-name  
group by rollup(category, item-name)
```

would give a hierarchical summary by *item-name* and by *category*.

Ranking

- ▶ Ranking is done in conjunction with an order by specification.
- ▶ Given a relation `student-marks(student-id, marks)` find the rank of each student.

```
select student-id, rank( ) over (order by marks desc) as s-rank  
from student-marks
```

- ▶ An extra `order by` clause is needed to get them in sorted order

```
select student-id, rank ( ) over (order by marks desc) as s-rank  
from student-marks  
order by s-rank
```
- ▶ Ranking may leave gaps: e.g. if 2 students have the same top mark, both have rank 1, and the next rank is 3
 - `dense_rank` does not leave gaps, so next dense rank would be 2

Ranking (Cont.)

- ▶ Ranking can be done within partition of the data.
- ▶ “Find the rank of students within each section.”

```
select student-id, section,  
       rank ( ) over (partition by section order by marks desc)  
       as sec-rank  
from student-marks, student-section  
where student-marks.student-id = student-section.student-id  
order by section, sec-rank
```

- ▶ Multiple **rank** clauses can occur in a single **select** clause
- ▶ Ranking is done *after* applying **group by** clause/aggregation

Ranking (Cont.)

- ▶ Other ranking functions:
 - **percent_rank** (within partition, if partitioning is done)
 - **cume_dist** (cumulative distribution)
 - fraction of tuples with preceding values
 - **row_number** (non-deterministic in presence of duplicates)
- ▶ SQL:1999 permits the user to specify **nulls first** or **nulls last**

```
select student-id,  
       rank ( ) over (order by marks desc nulls last) as s-rank  
from student-marks
```

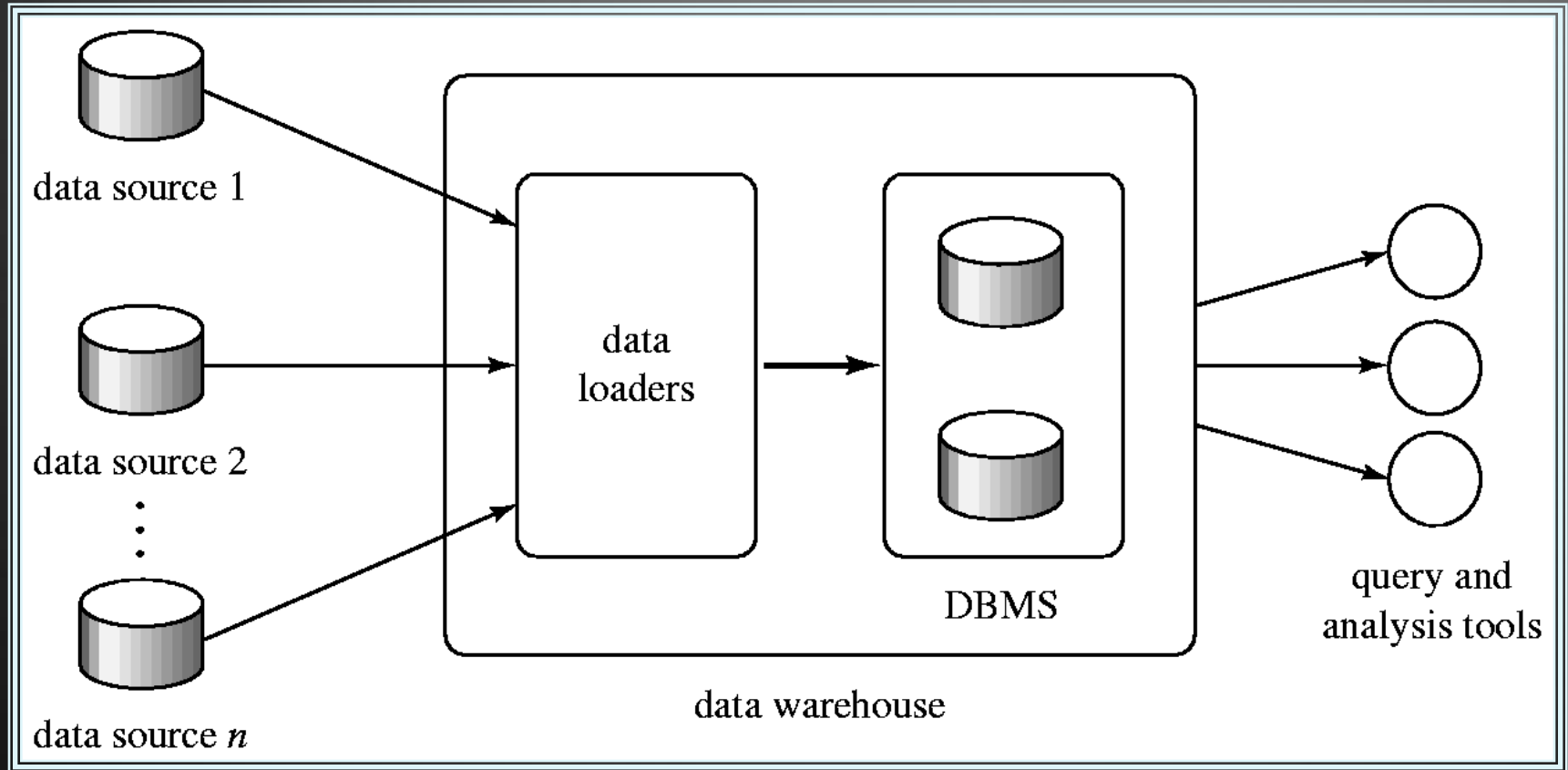
Ranking (Cont.)

- ▶ For a given constant n , the ranking the function $ntile(n)$ takes the tuples in each partition in the specified order, and divides them into n buckets with equal numbers of tuples.

- ▶ E.g.:

```
select threetile, sum(salary)  
from (  
    select salary, ntile(3) over (order by salary) as threetile  
    from employee) as s  
group by threetile
```

Data Warehousing



Design Issues

- ▶ *When and how to gather data*
 - Source driven architecture: data sources transmit new information to warehouse, either continuously or periodically (e.g. at night)
 - Destination driven architecture: warehouse periodically requests new information from data sources
 - Keeping warehouse exactly synchronized with data sources (e.g. using two-phase commit) is too expensive
 - Usually OK to have slightly out-of-date data at warehouse
 - Data/updates are periodically downloaded from online transaction processing (OLTP) systems.
- ▶ *What schema to use*
 - Schema integration

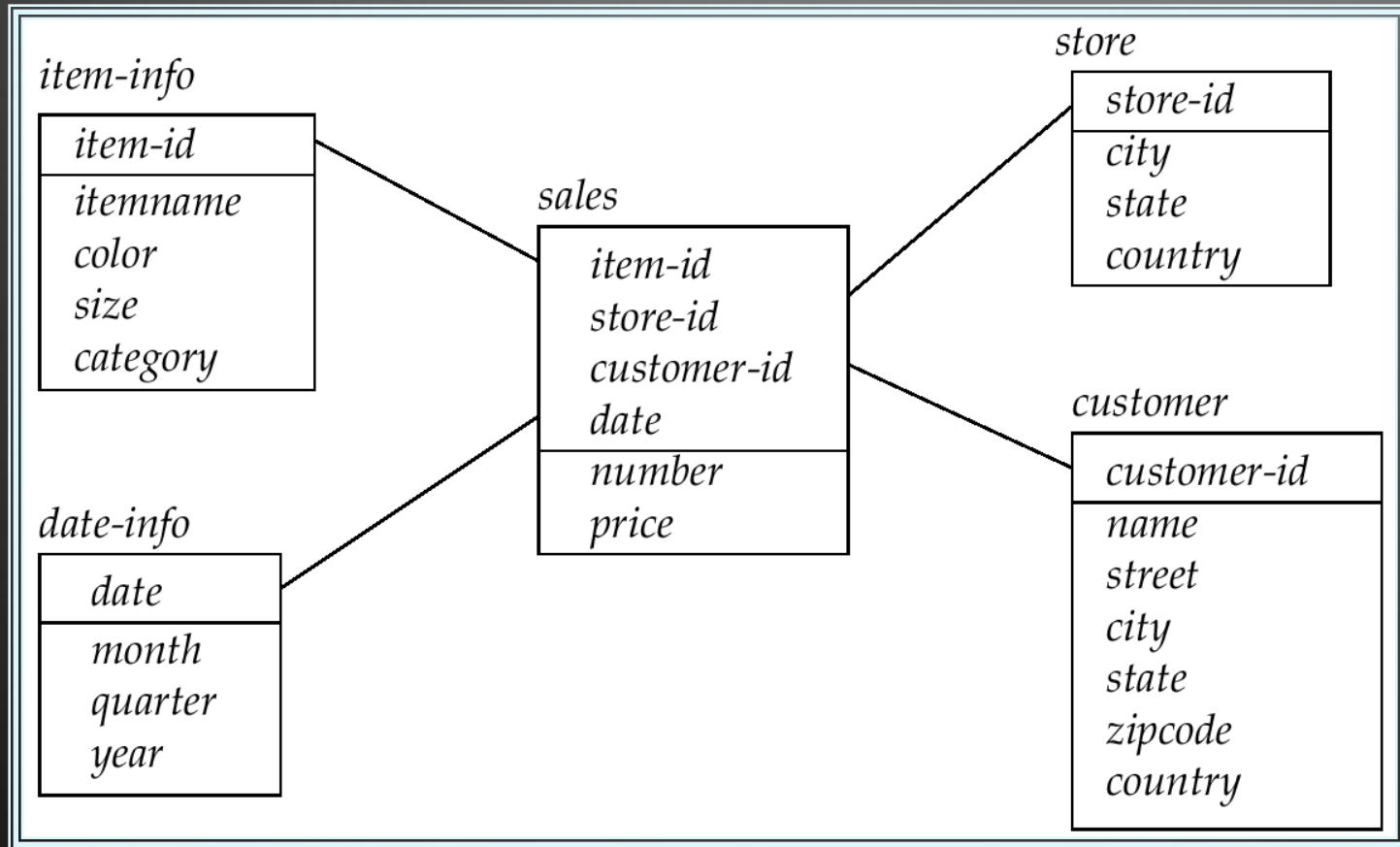
More Warehouse Design Issues

- ▶ *Data cleansing*
 - E.g. correct mistakes in addresses (misspellings, zip code errors)
 - Merge address lists from different sources and purge duplicates
- ▶ *How to propagate updates*
 - Warehouse schema may be a (materialized) view of schema from data sources
- ▶ *What data to summarize*
 - Raw data may be too large to store on-line
 - Aggregate values (totals/subtotals) often suffice
 - Queries on raw data can often be transformed by query optimizer to use aggregate values

Warehouse Schemas

- ▶ Dimension values are usually encoded using small integers and mapped to full values via dimension tables
- ▶ Resultant schema is called a **star schema**
 - More complicated schema structures
 - Snowflake schema: multiple levels of dimension tables
 - Constellation: multiple fact tables

Data Warehouse Schema



Data Mining

- ▶ Data mining is the process of semi-automatically analyzing large databases to find useful patterns
- ▶ Prediction based on past history
 - Predict if a credit card applicant poses a good credit risk, based on some attributes (income, job type, age, ..) and past history
 - Predict if a pattern of phone calling card usage is likely to be fraudulent
- ▶ Some examples of prediction mechanisms:
 - **Classification**
 - Given a new item whose class is unknown, predict to which class it belongs
 - **Regression formulae**
 - Given a set of mappings for an unknown function, predict the function result for a new parameter value

Data Mining (Cont.)

▶ Descriptive Patterns

◦ Associations

- Find books that are often bought by “similar” customers. If a new such customer buys one such book, suggest the others too.

◦ Associations may be used as a first step in detecting causation

- E.g. association between exposure to chemical X and cancer,

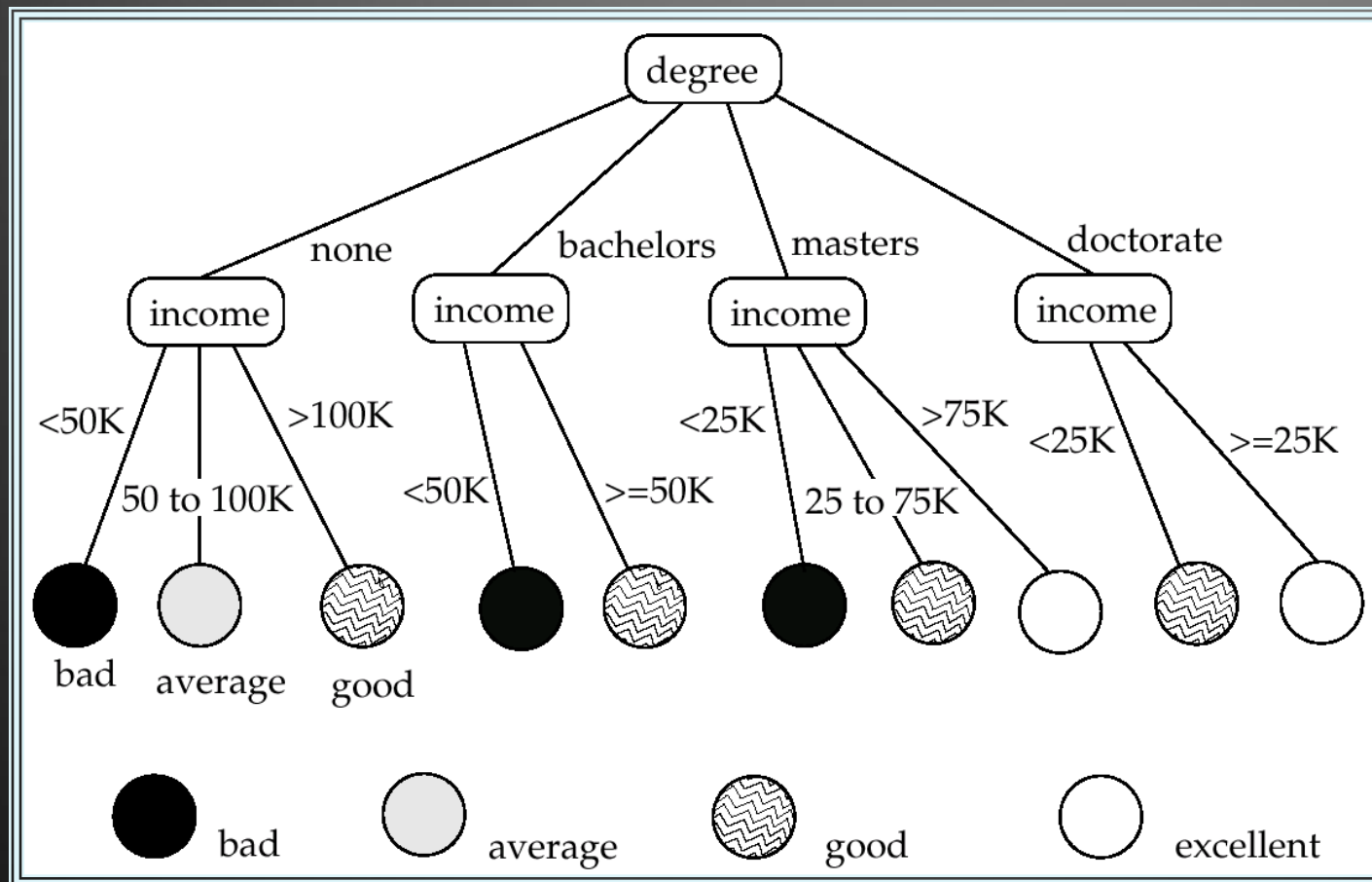
◦ Clusters

- E.g. typhoid cases were clustered in an area surrounding a contaminated well
- Detection of clusters remains important in detecting epidemics

Classification Rules

- ▶ Classification rules help assign new objects to classes.
 - E.g., given a new automobile insurance applicant, should he or she be classified as low risk, medium risk or high risk?
- ▶ Classification rules for above example could use a variety of data, such as educational level, salary, age, etc.
 - \forall person P, P.degree = masters and P.income > 75,000
 \Rightarrow P.credit =
excellent
 - \forall person P, P.degree = bachelors and
(P.income \geq 25,000 and P.income \leq 75,000)
 \Rightarrow P.credit =
good
- ▶ Rules are not necessarily exact: there may be some misclassifications
- ▶ Classification rules can be shown compactly as a decision tree.

Decision Tree



Construction of Decision Trees

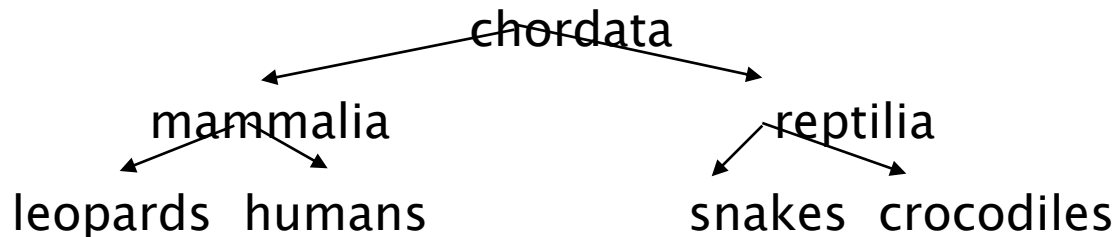
- ▶ **Training set:** a data sample in which the classification is already known.
- ▶ **Greedy top down generation of decision trees.**
 - Each internal node of the tree partitions the data into groups based on a **partitioning attribute**, and a **partitioning condition** for the node
 - **Leaf** node:
 - all (or most) of the items at the node belong to the same class, or
 - all attributes have been considered, and no further partitioning is possible.

Clustering

- ▶ Clustering: Intuitively, finding clusters of points in the given data such that similar points lie in the same cluster
- ▶ Can be formalized using distance metrics in several ways
 - Group points into k sets (for a given k) such that the average distance of points from the centroid of their assigned group is minimized
 - Centroid: point defined by taking average of coordinates in each dimension.
 - Another metric: minimize average distance between every pair of points in a cluster
- ▶ Has been studied extensively in statistics, but on small data sets
 - Data mining systems aim at clustering techniques that can handle very large data sets
 - E.g. the Birch clustering algorithm (more shortly)

Hierarchical Clustering

- ▶ Example from biological classification
 - (the word classification here does not mean a prediction mechanism)



- ▶ Other examples: Internet directory systems (e.g. Yahoo, more on this later)
- ▶ Agglomerative clustering algorithms
 - Build small clusters, then cluster small clusters into bigger clusters, and so on
- ▶ Divisive clustering algorithms
 - Start with all items in a single cluster, repeatedly refine (break) clusters into smaller ones

Clustering Algorithms

- ▶ Clustering algorithms have been designed to handle very large datasets
- ▶ E.g. the Birch algorithm
 - Main idea: use an in-memory R-tree to store points that are being clustered
 - Insert points one at a time into the R-tree, merging a new point with an existing cluster if it is less than some δ distance away
 - If there are more leaf nodes than fit in memory, merge existing clusters that are close to each other
 - At the end of first pass we get a large number of clusters at the leaves of the R-tree
 - Merge clusters to reduce the number of clusters

Collaborative Filtering

- ▶ Goal: predict what movies/books/... a person may be interested in, on the basis of
 - Past preferences of the person
 - Other people with similar past preferences
 - The preferences of such people for a new movie/book/...
- ▶ One approach based on repeated clustering
 - Cluster people on the basis of preferences for movies
 - Then cluster movies on the basis of being liked by the same clusters of people
 - Again cluster people based on their preferences for (the newly created clusters of) movies
 - Repeat above till equilibrium
- ▶ Above problem is an instance of **collaborative filtering**, where users collaborate in the task of filtering information to find information of interest

Other Types of Mining

- ▶ **Text mining:** application of data mining to textual documents
 - cluster Web pages to find related pages
 - cluster pages a user has visited to organize their visit history
 - classify Web pages automatically into a Web directory
- ▶ **Data visualization systems** help users examine large volumes of data and detect patterns visually
 - Can visually encode large amounts of information on a single screen
 - Humans are very good at detecting visual patterns