

# Introduction

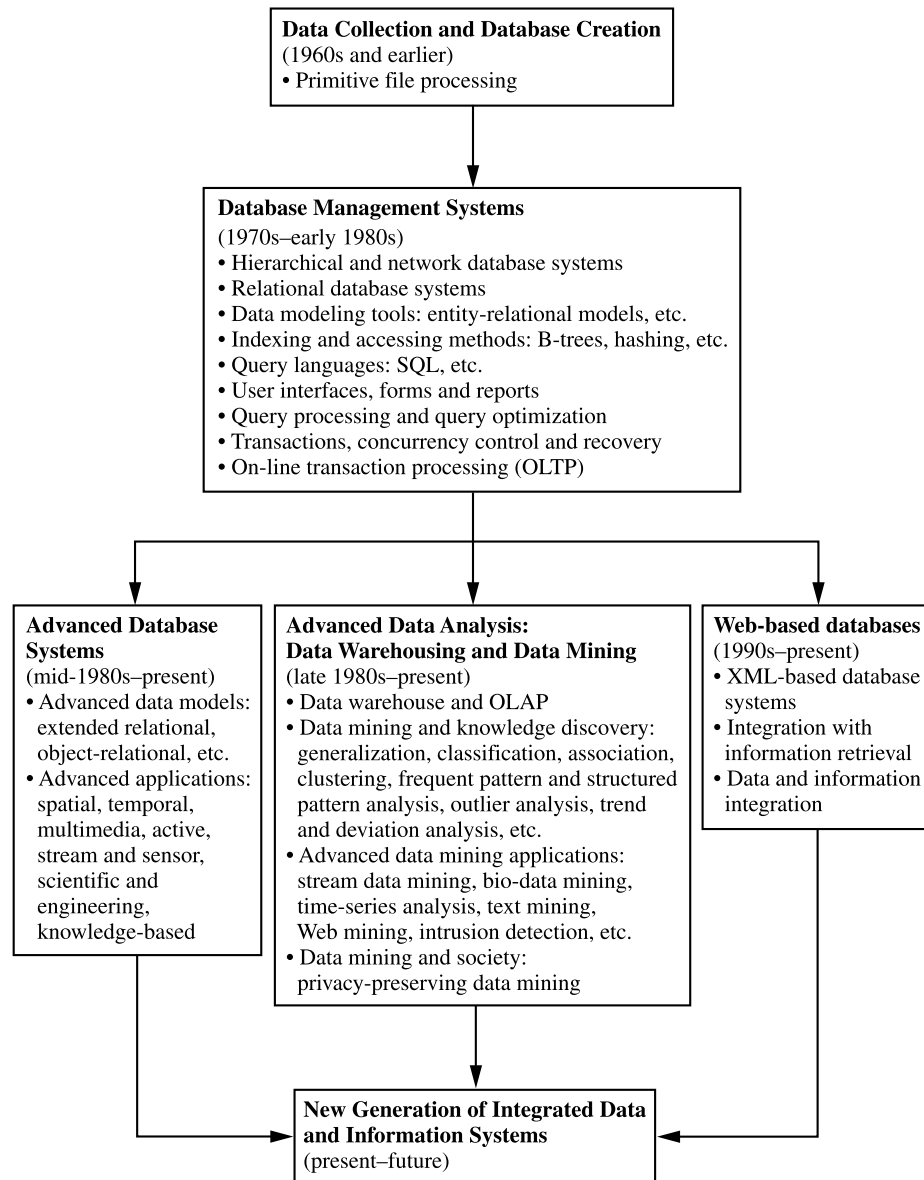
**This book is an introduction** to a young and promising field called *data mining* and *knowledge discovery from data*. The material in this book is presented from a database perspective, where emphasis is placed on basic data mining concepts and techniques for uncovering interesting data patterns hidden in *large data sets*. The implementation methods discussed are particularly oriented toward the development of *scalable* and *efficient* data mining tools. In this chapter, you will learn how data mining is part of the natural evolution of database technology, why data mining is important, and how it is defined. You will learn about the general architecture of data mining systems, as well as gain insight into the kinds of data on which mining can be performed, the types of patterns that can be found, and how to tell which patterns represent useful knowledge. You will study data mining primitives, from which data mining query languages can be designed. Issues regarding how to integrate a data mining system with a database or data warehouse are also discussed. In addition to studying a classification of data mining systems, you will read about challenging research issues for building data mining tools of the future.

## What Motivated Data Mining? Why Is It Important?

Necessity is the mother of invention. —Plato

Data mining has attracted a great deal of attention in the information industry and in society as a whole in recent years, due to the wide availability of huge amounts of data and the imminent need for turning such data into useful information and knowledge. The information and knowledge gained can be used for applications ranging from market analysis, fraud detection, and customer retention, to production control and science exploration.

Data mining can be viewed as a result of the natural evolution of information technology. The database system industry has witnessed an evolutionary path in the development of the following functionalities (Figure 1.1): *data collection and database creation*, *data management* (including data storage and retrieval, and database



**Figure 1.1** The evolution of database system technology.

transaction processing), and *advanced data analysis* (involving data warehousing and data mining). For instance, the early development of data collection and database creation mechanisms served as a prerequisite for later development of effective mechanisms for data storage and retrieval, and query and transaction processing. With numerous database systems offering query and transaction processing as common practice, advanced data analysis has naturally become the next target.

Since the 1960s, database and information technology has been evolving systematically from primitive file processing systems to sophisticated and powerful database systems. The research and development in database systems since the 1970s has progressed from early hierarchical and network database systems to the development of relational database systems (where data are stored in relational table structures; see Section 1.3.1), data modeling tools, and indexing and accessing methods. In addition, users gained convenient and flexible data access through query languages, user interfaces, optimized query processing, and transaction management. Efficient methods for **on-line transaction processing (OLTP)**, where a query is viewed as a read-only transaction, have contributed substantially to the evolution and wide acceptance of relational technology as a major tool for efficient storage, retrieval, and management of large amounts of data.

Database technology since the mid-1980s has been characterized by the popular adoption of relational technology and an upsurge of research and development activities on new and powerful database systems. These promote the development of advanced data models such as extended-relational, object-oriented, object-relational, and deductive models. Application-oriented database systems, including spatial, temporal, multimedia, active, stream, and sensor, and scientific and engineering databases, knowledge bases, and office information bases, have flourished. Issues related to the distribution, diversification, and sharing of data have been studied extensively. Heterogeneous database systems and Internet-based global information systems such as the World Wide Web (WWW) have also emerged and play a vital role in the information industry.

The steady and amazing progress of computer hardware technology in the past three decades has led to large supplies of powerful and affordable computers, data collection equipment, and storage media. This technology provides a great boost to the database and information industry, and makes a huge number of databases and information repositories available for transaction management, information retrieval, and data analysis.

Data can now be stored in many different kinds of databases and information repositories. One data repository architecture that has emerged is the **data warehouse** (Section 1.3.2), a repository of multiple heterogeneous data sources organized under a unified schema at a single site in order to facilitate management decision making. Data warehouse technology includes data cleaning, data integration, and **on-line analytical processing (OLAP)**, that is, analysis techniques with functionalities such as summarization, consolidation, and aggregation as well as the ability to view information from different angles. Although OLAP tools support multidimensional analysis and decision making, additional data analysis tools are required for in-depth analysis, such as



**Figure 1.2** We are data rich, but information poor.

data classification, clustering, and the characterization of data changes over time. In addition, huge volumes of data can be accumulated beyond databases and data warehouses. Typical examples include the World Wide Web and *data streams*, where data flow in and out like streams, as in applications like video surveillance, telecommunication, and sensor networks. The effective and efficient analysis of data in such different forms becomes a challenging task.

The abundance of data, coupled with the need for powerful data analysis tools, has been described as a *data rich but information poor* situation. The fast-growing, tremendous amount of data, collected and stored in large and numerous data repositories, has far exceeded our human ability for comprehension without powerful tools (Figure 1.2). As a result, data collected in large data repositories become “data tombs”—data archives that are seldom visited. Consequently, important decisions are often made based not on the information-rich data stored in data repositories, but rather on a decision maker’s intuition, simply because the decision maker does not have the tools to extract the valuable knowledge embedded in the vast amounts of data. In addition, consider expert system technologies, which typically rely on users or domain experts to *manually* input knowledge into knowledge bases. Unfortunately, this procedure is prone to biases and errors, and is extremely time-consuming and costly. Data mining tools perform data analysis and may uncover important data patterns, contributing greatly to business

strategies, knowledge bases, and scientific and medical research. The widening gap between data and information calls for a systematic development of *data mining tools* that will turn data tombs into “golden nuggets” of knowledge.

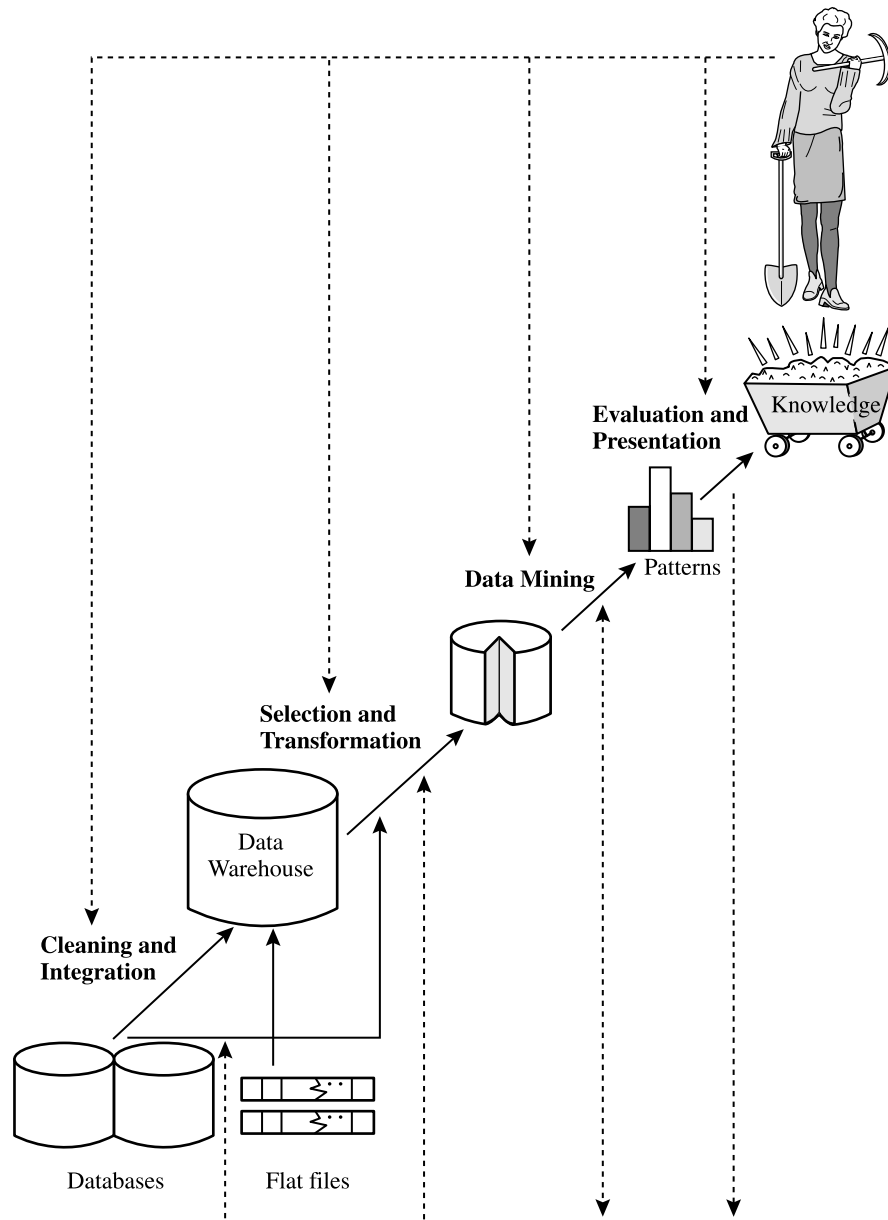
## 1.2 So, What Is Data Mining?

Simply stated, **data mining** refers to *extracting or “mining” knowledge from large amounts of data*. The term is actually a misnomer. Remember that the mining of gold from rocks or sand is referred to as *gold mining* rather than rock or sand mining. Thus, data mining should have been more appropriately named “knowledge mining from data,” which is unfortunately somewhat long. “Knowledge mining,” a shorter term, may not reflect the emphasis on mining from large amounts of data. Nevertheless, mining is a vivid term characterizing the process that finds a small set of precious nuggets from a great deal of raw material (Figure 1.3). Thus, such a misnomer that carries both “data” and “mining” became a popular choice. Many other terms carry a similar or slightly different meaning to data mining, such as **knowledge mining from data**, **knowledge extraction**, **data/pattern analysis**, **data archaeology**, and **data dredging**.

Many people treat data mining as a synonym for another popularly used term, **Knowledge Discovery from Data**, or KDD. Alternatively, others view data mining as simply an



**Figure 1.3** Data mining—searching for knowledge (interesting patterns) in your data.



**Figure 1.4** Data mining as a step in the process of knowledge discovery.

essential step in the process of knowledge discovery. Knowledge discovery as a process is depicted in Figure 1.4 and consists of an iterative sequence of the following steps:

1. **Data cleaning** (to remove noise and inconsistent data)
2. **Data integration** (where multiple data sources may be combined)<sup>1</sup>
3. **Data selection** (where data relevant to the analysis task are retrieved from the database)
4. **Data transformation** (where data are transformed or consolidated into forms appropriate for mining by performing summary or aggregation operations, for instance)<sup>2</sup>
5. **Data mining** (an essential process where intelligent methods are applied in order to extract data patterns)
6. **Pattern evaluation** (to identify the truly interesting patterns representing knowledge based on some **interestingness measures**; Section 1.5)
7. **Knowledge presentation** (where visualization and knowledge representation techniques are used to present the mined knowledge to the user)

Steps 1 to 4 are different forms of data preprocessing, where the data are prepared for mining. The data mining step may interact with the user or a knowledge base. The interesting patterns are presented to the user and may be stored as new knowledge in the knowledge base. Note that according to this view, data mining is only one step in the entire process, albeit an essential one because it uncovers hidden patterns for evaluation.

We agree that data mining is a step in the knowledge discovery process. However, in industry, in media, and in the database research milieu, the term data mining is becoming more popular than the longer term of knowledge discovery from data. Therefore, in this book, we choose to use the term data mining. We adopt a broad view of data mining functionality: data mining is the process of discovering interesting knowledge from large amounts of data stored in databases, data warehouses, or other information repositories.

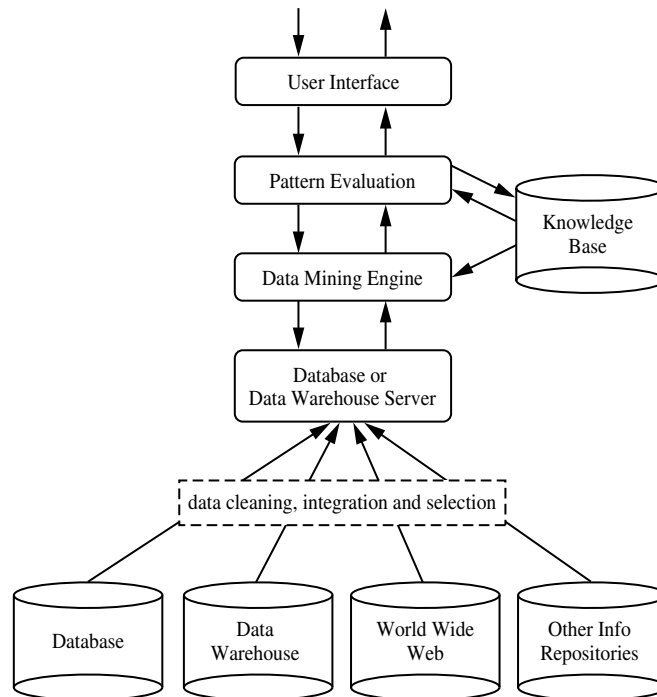
Based on this view, the architecture of a typical data mining system may have the following major components (Figure 1.5):

- **Database, data warehouse, World Wide Web, or other information repository:** This is one or a set of databases, data warehouses, spreadsheets, or other kinds of information repositories. Data cleaning and data integration techniques may be performed on the data.
- **Database or data warehouse server:** The database or data warehouse server is responsible for fetching the relevant data, based on the user's data mining request.

---

<sup>1</sup> A popular trend in the information industry is to perform data cleaning and data integration as a preprocessing step, where the resulting data are stored in a data warehouse.

<sup>2</sup> Sometimes data transformation and consolidation are performed before the data selection process, particularly in the case of data warehousing. *Data reduction* may also be performed to obtain a smaller representation of the original data without sacrificing its integrity.



**Figure 1.5** Architecture of a typical data mining system.

- **Knowledge base:** This is the domain knowledge that is used to guide the search or evaluate the interestingness of resulting patterns. Such knowledge can include **concept hierarchies**, used to organize attributes or attribute values into different levels of abstraction. Knowledge such as user beliefs, which can be used to assess a pattern's interestingness based on its unexpectedness, may also be included. Other examples of domain knowledge are additional interestingness constraints or thresholds, and metadata (e.g., describing data from multiple heterogeneous sources).
- **Data mining engine:** This is essential to the data mining system and ideally consists of a set of functional modules for tasks such as characterization, association and correlation analysis, classification, prediction, cluster analysis, outlier analysis, and evolution analysis.
- **Pattern evaluation module:** This component typically employs interestingness measures (Section 1.5) and interacts with the data mining modules so as to *focus* the search toward interesting patterns. It may use interestingness thresholds to filter out discovered patterns. Alternatively, the pattern evaluation module may be integrated with the mining module, depending on the implementation of the data mining method used. For efficient data mining, it is highly recommended to push

the evaluation of pattern interestingness as deep as possible into the mining process so as to confine the search to only the interesting patterns.

- **User interface:** This module communicates between users and the data mining system, allowing the user to interact with the system by specifying a data mining query or task, providing information to help focus the search, and performing exploratory data mining based on the intermediate data mining results. In addition, this component allows the user to browse database and data warehouse schemas or data structures, evaluate mined patterns, and visualize the patterns in different forms.

From a data warehouse perspective, data mining can be viewed as an advanced stage of on-line analytical processing (OLAP). However, data mining goes far beyond the narrow scope of summarization-style analytical processing of data warehouse systems by incorporating more advanced techniques for data analysis.

Although there are many “data mining systems” on the market, not all of them can perform true data mining. A data analysis system that does not handle large amounts of data should be more appropriately categorized as a machine learning system, a statistical data analysis tool, or an experimental system prototype. A system that can only perform data or information retrieval, including finding aggregate values, or that performs deductive query answering in large databases should be more appropriately categorized as a database system, an information retrieval system, or a deductive database system.

Data mining involves an integration of techniques from multiple disciplines such as database and data warehouse technology, statistics, machine learning, high-performance computing, pattern recognition, neural networks, data visualization, information retrieval, image and signal processing, and spatial or temporal data analysis. We adopt a database perspective in our presentation of data mining in this book. That is, emphasis is placed on *efficient* and *scalable* data mining techniques. For an algorithm to be *scalable*, its running time should grow approximately linearly in proportion to the size of the data, given the available system resources such as main memory and disk space. By performing data mining, interesting knowledge, regularities, or high-level information can be extracted from databases and viewed or browsed from different angles. The discovered knowledge can be applied to decision making, process control, information management, and query processing. Therefore, data mining is considered one of the most important frontiers in database and information systems and one of the most promising interdisciplinary developments in the information technology.

## 1.3 Data Mining—On What Kind of Data?

In this section, we examine a number of different data repositories on which mining can be performed. In principle, data mining should be applicable to any kind of data repository, as well as to transient data, such as data streams. Thus the scope of our examination of data repositories will include relational databases, data warehouses, transactional databases, advanced database systems, flat files, data streams, and the

World Wide Web. Advanced database systems include object-relational databases and specific application-oriented databases, such as spatial databases, time-series databases, text databases, and multimedia databases. The challenges and techniques of mining may differ for each of the repository systems.

Although this book assumes that readers have basic knowledge of information systems, we provide a brief introduction to each of the major data repository systems listed above. In this section, we also introduce the fictitious *AllElectronics* store, which will be used to illustrate concepts throughout the text.

### 1.3.1 Relational Databases

A database system, also called a **database management system (DBMS)**, consists of a collection of interrelated data, known as a **database**, and a set of software programs to manage and access the data. The software programs involve mechanisms for the definition of database structures; for data storage; for concurrent, shared, or distributed data access; and for ensuring the consistency and security of the information stored, despite system crashes or attempts at unauthorized access.

A **relational database** is a collection of **tables**, each of which is assigned a unique name. Each table consists of a set of **attributes** (*columns* or *fields*) and usually stores a large set of **tuples** (*records* or *rows*). Each tuple in a relational table represents an object identified by a unique *key* and described by a set of attribute values. A semantic data model, such as an **entity-relationship (ER)** data model, is often constructed for relational databases. An ER data model represents the database as a set of entities and their relationships.

Consider the following example.

**Example 1.1** A relational database for *AllElectronics*. The *AllElectronics* company is described by the following relation tables: *customer*, *item*, *employee*, and *branch*. Fragments of the tables described here are shown in Figure 1.6.

- The relation *customer* consists of a set of attributes, including a unique customer identity number (*cust.ID*), customer name, address, age, occupation, annual income, credit information, category, and so on.
- Similarly, each of the relations *item*, *employee*, and *branch* consists of a set of attributes describing their properties.
- Tables can also be used to represent the relationships between or among multiple relation tables. For our example, these include *purchases* (customer purchases items, creating a sales transaction that is handled by an employee), *items\_sold* (lists the items sold in a given transaction), and *works\_at* (employee works at a branch of *AllElectronics*). ■

Relational data can be accessed by **database queries** written in a relational query language, such as SQL, or with the assistance of graphical user interfaces. In the latter, the user may employ a menu, for example, to specify attributes to be included in the query, and the constraints on these attributes. A given query is transformed into a set of

<i>customer</i>							
<u>cust_ID</u>	name	address	age	income	credit_info	category	...
C1	Smith, Sandy	1223 Lake Ave., Chicago, IL	31	\$78000	1	3	...
...	...	...	...	...	...	...	...

<i>item</i>								
<u>item_ID</u>	name	brand	category	type	price	place_made	supplier	cost
I3	hi-res-TV	Toshiba	high resolution	TV	\$988.00	Japan	NikoX	\$600.00
I8	Laptop	Dell	laptop	computer	\$1369.00	USA	Dell	\$983.00
...	...	...	...	...	...	...	...	...

<i>employee</i>						
<u>empl_ID</u>	name	category	group	salary	commission	
E55	Jones, Jane	home entertainment	manager	\$118,000	2%	
...	...	...	...	...	...	...

<i>branch</i>		
<u>branch_ID</u>	name	address
B1	City Square	396 Michigan Ave., Chicago, IL
...	...	...

<i>purchases</i>						
<u>trans_ID</u>	cust_ID	empl_ID	date	time	method_paid	amount
T100	C1	E55	03/21/2005	15:45	Visa	\$1357.00
...	...	...	...	...	...	...

<i>items_sold</i>		
<u>trans_ID</u>	<u>item_ID</u>	qty
T100	I3	1
T100	I8	2
...	...	...

<i>works_at</i>	
<u>empl_ID</u>	<u>branch_ID</u>
E55	B1
...	...

**Figure 1.6** Fragments of relations from a relational database for *AllElectronics*.

relational operations, such as join, selection, and projection, and is then optimized for efficient processing. A query allows retrieval of specified subsets of the data. Suppose that your job is to analyze the *AllElectronics* data. Through the use of relational queries, you can ask things like “Show me a list of all items that were sold in the last quarter.” Relational languages also include aggregate functions such as sum, avg (average), count, max (maximum), and min (minimum). These allow you to ask things like “Show me the total sales of the last month, grouped by branch,” or “How many sales transactions occurred in the month of December?” or “Which sales person had the highest amount of sales?”

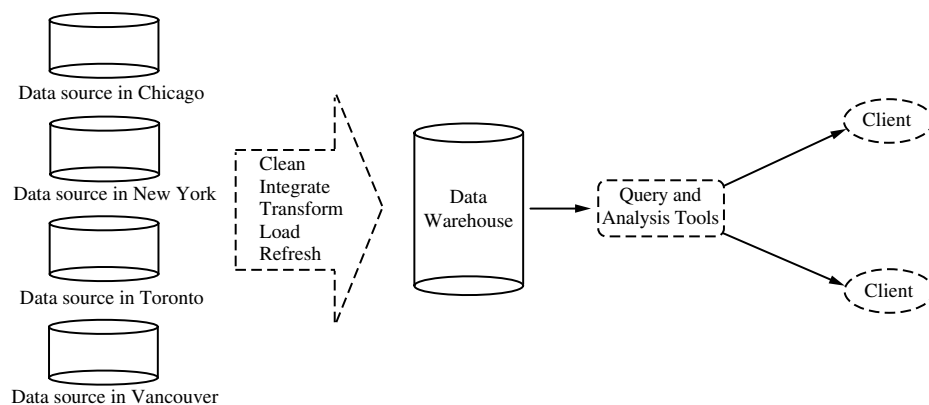
When data mining is applied to relational databases, we can go further by *searching for trends or data patterns*. For example, data mining systems can analyze customer data to predict the credit risk of new customers based on their income, age, and previous credit information. Data mining systems may also detect deviations, such as items whose sales are far from those expected in comparison with the previous year. Such deviations can then be further investigated (e.g., has there been a change in packaging of such items, or a significant increase in price?).

Relational databases are one of the most commonly available and rich information repositories, and thus they are a major data form in our study of data mining.

### 1.3.2 Data Warehouses

Suppose that *AllElectronics* is a successful international company, with branches around the world. Each branch has its own set of databases. The president of *AllElectronics* has asked you to provide an analysis of the company's sales per item type per branch for the third quarter. This is a difficult task, particularly since the relevant data are spread out over several databases, physically located at numerous sites.

If *AllElectronics* had a data warehouse, this task would be easy. A **data warehouse** is a repository of information collected from multiple sources, stored under a unified schema, and that usually resides at a single site. Data warehouses are constructed via a process of data cleaning, data integration, data transformation, data loading, and periodic data refreshing. This process is discussed in Chapters 2 and 3. Figure 1.7 shows the typical framework for construction and use of a data warehouse for *AllElectronics*.



**Figure 1.7** Typical framework of a data warehouse for *AllElectronics*.

To facilitate decision making, the data in a data warehouse are *organized around major subjects*, such as customer, item, supplier, and activity. The data are stored to provide information from a *historical perspective* (such as from the past 5–10 years) and are typically *summarized*. For example, rather than storing the details of each sales transaction, the data warehouse may store a summary of the transactions per item type for each store or, summarized to a higher level, for each sales region.

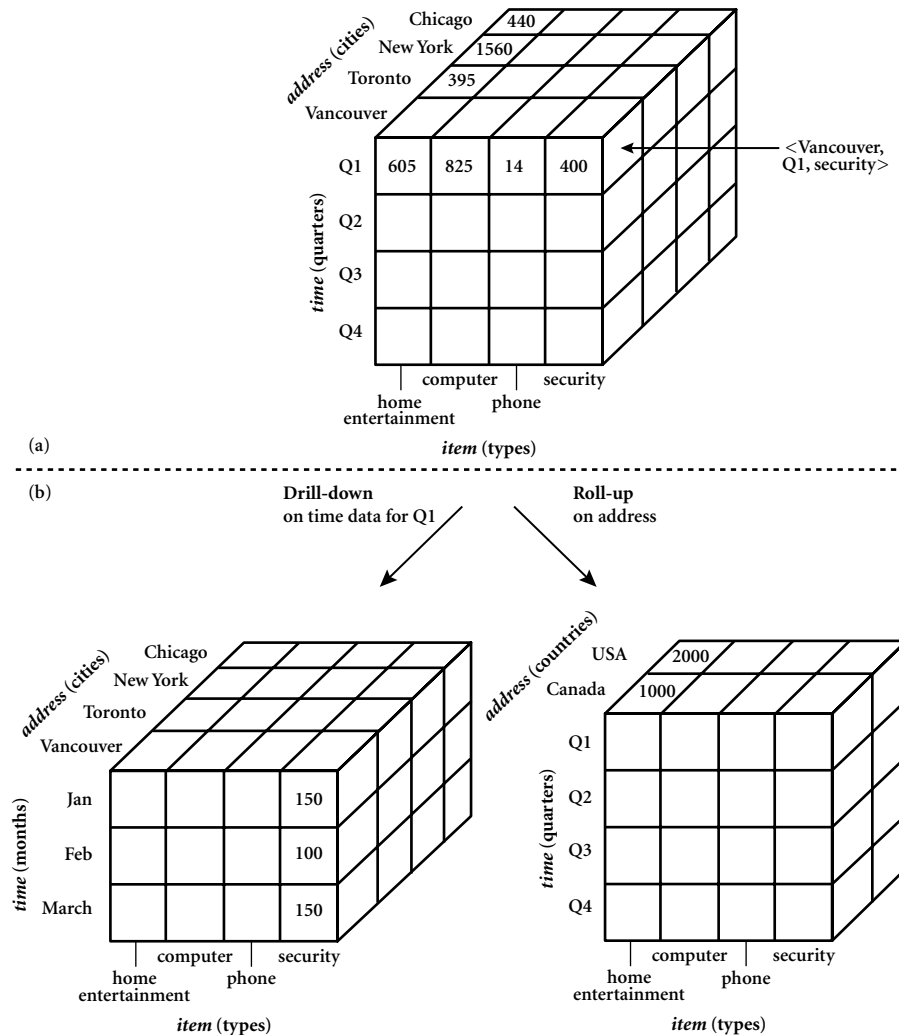
A data warehouse is usually modeled by a multidimensional database structure, where each **dimension** corresponds to an attribute or a set of attributes in the schema, and each **cell** stores the value of some aggregate measure, such as *count* or *sales\_amount*. The actual physical structure of a data warehouse may be a relational data store or a **multidimensional data cube**. A data cube provides a multidimensional view of data and allows the precomputation and fast accessing of summarized data.

**Example 1.2** A data cube for *AllElectronics*. A data cube for summarized sales data of *AllElectronics* is presented in Figure 1.8(a). The cube has three dimensions: *address* (with city values *Chicago*, *New York*, *Toronto*, *Vancouver*), *time* (with quarter values *Q1*, *Q2*, *Q3*, *Q4*), and *item* (with item type values *home entertainment*, *computer*, *phone*, *security*). The aggregate value stored in each cell of the cube is *sales\_amount* (in thousands). For example, the total sales for the first quarter, *Q1*, for items relating to security systems in Vancouver is \$400,000, as stored in cell  $\langle \text{Vancouver}, \text{Q1}, \text{security} \rangle$ . Additional cubes may be used to store aggregate sums over each dimension, corresponding to the aggregate values obtained using different SQL group-bys (e.g., the total sales amount per city and quarter, or per city and item, or per quarter and item, or per each individual dimension). ■

“I have also heard about data marts. What is the difference between a data warehouse and a data mart?” you may ask. A data warehouse collects information about subjects that span an *entire organization*, and thus its scope is *enterprise-wide*. A **data mart**, on the other hand, is a department subset of a data warehouse. It focuses on selected subjects, and thus its scope is *department-wide*.

By providing multidimensional data views and the precomputation of summarized data, data warehouse systems are well suited for **on-line analytical processing**, or **OLAP**. OLAP operations use background knowledge regarding the domain of the data being studied in order to allow the presentation of data at *different levels of abstraction*. Such operations accommodate different user viewpoints. Examples of OLAP operations include **drill-down** and **roll-up**, which allow the user to view the data at differing degrees of summarization, as illustrated in Figure 1.8(b). For instance, we can drill down on sales data summarized by *quarter* to see the data summarized by *month*. Similarly, we can roll up on sales data summarized by *city* to view the data summarized by *country*.

Although data warehouse tools help support data analysis, additional tools for data mining are required to allow more in-depth and automated analysis. An overview of data warehouse and OLAP technology is provided in Chapter 3. Advanced issues regarding data warehouse and OLAP implementation and data generalization are discussed in Chapter 4.



**Figure 1.8** A multidimensional data cube, commonly used for data warehousing, (a) showing summarized data for *AllElectronics* and (b) showing summarized data resulting from drill-down and roll-up operations on the cube in (a). For improved readability, only some of the cube cell values are shown.

### 1.3.3 Transactional Databases

In general, a **transactional database** consists of a file where each record represents a transaction. A transaction typically includes a unique transaction identity number (*trans.ID*) and a list of the **items** making up the transaction (such as items purchased in a store).

<i>trans_ID</i>	<i>list of item_IDs</i>
T100	I1, I3, I8, I16
T200	I2, I8
...	...

**Figure 1.9** Fragment of a transactional database for sales at *Allelectronics*.

The transactional database may have additional tables associated with it, which contain other information regarding the sale, such as the date of the transaction, the customer ID number, the ID number of the salesperson and of the branch at which the sale occurred, and so on.

**Example 1.3** A transactional database for *Allelectronics*. Transactions can be stored in a table, with one record per transaction. A fragment of a transactional database for *Allelectronics* is shown in Figure 1.9. From the relational database point of view, the *sales* table in Figure 1.9 is a nested relation because the attribute *list of item\_IDs* contains a set of *items*. Because most relational database systems do not support nested relational structures, the transactional database is usually either stored in a flat file in a format similar to that of the table in Figure 1.9 or unfolded into a standard relation in a format similar to that of the *items\_sold* table in Figure 1.6. ■

As an analyst of the *Allelectronics* database, you may ask, “Show me all the items purchased by Sandy Smith” or “How many transactions include item number I3?” Answering such queries may require a scan of the entire transactional database.

Suppose you would like to dig deeper into the data by asking, “Which items sold well together?” This kind of *market basket data analysis* would enable you to bundle groups of items together as a strategy for maximizing sales. For example, given the knowledge that printers are commonly purchased together with computers, you could offer an expensive model of printers at a discount to customers buying selected computers, in the hopes of selling more of the expensive printers. A regular data retrieval system is not able to answer queries like the one above. However, data mining systems for transactional data can do so by identifying *frequent itemsets*, that is, sets of items that are frequently sold together. The mining of such frequent patterns for transactional data is discussed in Chapter 5.

### 1.3.4 Advanced Data and Information Systems and Advanced Applications

Relational database systems have been widely used in business applications. With the progress of database technology, various kinds of advanced data and information systems have emerged and are undergoing development to address the requirements of new applications.

The new database applications include handling spatial data (such as maps), engineering design data (such as the design of buildings, system components, or integrated circuits), hypertext and multimedia data (including text, image, video, and audio data), time-related data (such as historical records or stock exchange data), stream data (such as video surveillance and sensor data, where data flow in and out like streams), and the World Wide Web (a huge, widely distributed information repository made available by the Internet). These applications require efficient data structures and scalable methods for handling complex object structures; variable-length records; semistructured or unstructured data; text, spatiotemporal, and multimedia data; and database schemas with complex structures and dynamic changes.

In response to these needs, advanced database systems and specific application-oriented database systems have been developed. These include object-relational database systems, temporal and time-series database systems, spatial and spatiotemporal database systems, text and multimedia database systems, heterogeneous and legacy database systems, data stream management systems, and Web-based global information systems.

While such databases or information repositories require sophisticated facilities to efficiently store, retrieve, and update large amounts of complex data, they also provide fertile grounds and raise many challenging research and implementation issues for data mining. In this section, we describe each of the advanced database systems listed above.

## Object-Relational Databases

**Object-relational databases** are constructed based on an object-relational data model. This model extends the relational model by providing a rich data type for handling complex objects and object orientation. Because most sophisticated database applications need to handle complex objects and structures, object-relational databases are becoming increasingly popular in industry and applications.

Conceptually, the object-relational data model inherits the essential concepts of **object-oriented databases**, where, in general terms, each entity is considered as an **object**. Following the *AllElectronics* example, objects can be individual employees, customers, or items. Data and code relating to an object are *encapsulated* into a single unit. Each object has associated with it the following:

- A set of **variables** that describe the objects. These correspond to attributes in the entity-relationship and relational models.
- A set of **messages** that the object can use to communicate with other objects, or with the rest of the database system.
- A set of **methods**, where each method holds the code to implement a message. Upon receiving a message, the method returns a value in response. For instance, the method for the message *get\_photo(employee)* will retrieve and return a photo of the given employee object.

Objects that share a common set of properties can be grouped into an **object class**. Each object is an **instance** of its class. Object classes can be organized into class/subclass

hierarchies so that each class represents properties that are common to objects in that class. For instance, an *employee* class can contain variables like *name*, *address*, and *birth-date*. Suppose that the class, *sales\_person*, is a subclass of the class, *employee*. A *sales\_person* object would **inherit** all of the variables pertaining to its superclass of *employee*. In addition, it has all of the variables that pertain specifically to being a salesperson (e.g., *commission*). Such a class inheritance feature benefits information sharing.

For data mining in object-relational systems, techniques need to be developed for handling complex object structures, complex data types, class and subclass hierarchies, property inheritance, and methods and procedures.

## Temporal Databases, Sequence Databases, and Time-Series Databases

A **temporal database** typically stores relational data that include time-related attributes. These attributes may involve several timestamps, each having different semantics. A **sequence database** stores sequences of ordered events, with or without a concrete notion of time. Examples include customer shopping sequences, Web click streams, and biological sequences. A **time-series database** stores sequences of values or events obtained over repeated measurements of time (e.g., hourly, daily, weekly). Examples include data collected from the stock exchange, inventory control, and the observation of natural phenomena (like temperature and wind).

Data mining techniques can be used to find the characteristics of object evolution, or the trend of changes for objects in the database. Such information can be useful in decision making and strategy planning. For instance, the mining of banking data may aid in the scheduling of bank tellers according to the volume of customer traffic. Stock exchange data can be mined to uncover trends that could help you plan investment strategies (e.g., when is the best time to purchase *AlIElectronics* stock?). Such analyses typically require defining multiple granularity of time. For example, time may be decomposed according to fiscal years, academic years, or calendar years. Years may be further decomposed into quarters or months.

## Spatial Databases and Spatiotemporal Databases

**Spatial databases** contain spatial-related information. Examples include geographic (map) databases, very large-scale integration (VLSI) or computed-aided design databases, and medical and satellite image databases. Spatial data may be represented in **raster format**, consisting of  $n$ -dimensional bit maps or pixel maps. For example, a 2-D satellite image may be represented as raster data, where each pixel registers the rainfall in a given area. Maps can be represented in **vector format**, where roads, bridges, buildings, and lakes are represented as unions or overlays of basic geometric constructs, such as points, lines, polygons, and the partitions and networks formed by these components.

Geographic databases have numerous applications, ranging from forestry and ecology planning to providing public service information regarding the location of telephone and electric cables, pipes, and sewage systems. In addition, geographic databases are

commonly used in vehicle navigation and dispatching systems. An example of such a system for taxis would store a city map with information regarding one-way streets, suggested routes for moving from region A to region B during rush hour, and the location of restaurants and hospitals, as well as the current location of each driver.

“What kind of data mining can be performed on spatial databases?” you may ask. Data mining may uncover patterns describing the characteristics of houses located near a specified kind of location, such as a park, for instance. Other patterns may describe the climate of mountainous areas located at various altitudes, or describe the change in trend of metropolitan poverty rates based on city distances from major highways. The relationships among a set of spatial objects can be examined in order to discover which subsets of objects are spatially auto-correlated or associated. Clusters and outliers can be identified by spatial cluster analysis. Moreover, spatial classification can be performed to construct models for prediction based on the relevant set of features of the spatial objects. Furthermore, “spatial data cubes” may be constructed to organize data into multidimensional structures and hierarchies, on which OLAP operations (such as drill-down and roll-up) can be performed.

A spatial database that stores spatial objects that change with time is called a **spatiotemporal database**, from which interesting information can be mined. For example, we may be able to group the trends of moving objects and identify some strangely moving vehicles, or distinguish a bioterrorist attack from a normal outbreak of the flu based on the geographic spread of a disease with time.

## Text Databases and Multimedia Databases

**Text databases** are databases that contain word descriptions for objects. These word descriptions are usually not simple keywords but rather long sentences or paragraphs, such as product specifications, error or bug reports, warning messages, summary reports, notes, or other documents. Text databases may be highly unstructured (such as some Web pages on the World Wide Web). Some text databases may be somewhat structured, that is, *semistructured* (such as e-mail messages and many HTML/XML Web pages), whereas others are relatively well structured (such as library catalogue databases). Text databases with highly regular structures typically can be implemented using relational database systems.

“What can data mining on text databases uncover?” By mining text data, one may uncover general and concise descriptions of the text documents, keyword or content associations, as well as the clustering behavior of text objects. To do this, standard data mining methods need to be integrated with information retrieval techniques and the construction or use of hierarchies specifically for text data (such as dictionaries and thesauruses), as well as discipline-oriented term classification systems (such as in biochemistry, medicine, law, or economics).

**Multimedia databases** store image, audio, and video data. They are used in applications such as picture content-based retrieval, voice-mail systems, video-on-demand systems, the World Wide Web, and speech-based user interfaces that recognize spoken commands. Multimedia databases must support large objects, because data objects such

as video can require gigabytes of storage. Specialized storage and search techniques are also required. Because video and audio data require real-time retrieval at a steady and predetermined rate in order to avoid picture or sound gaps and system buffer overflows, such data are referred to as **continuous-media** data.

For multimedia data mining, storage and search techniques need to be integrated with standard data mining methods. Promising approaches include the construction of multimedia data cubes, the extraction of multiple features from multimedia data, and similarity-based pattern matching.

## Heterogeneous Databases and Legacy Databases

A **heterogeneous database** consists of a set of interconnected, autonomous component databases. The components communicate in order to exchange information and answer queries. Objects in one component database may differ greatly from objects in other component databases, making it difficult to assimilate their semantics into the overall heterogeneous database.

Many enterprises acquire legacy databases as a result of the long history of information technology development (including the application of different hardware and operating systems). A **legacy database** is a group of *heterogeneous databases* that combines different kinds of data systems, such as relational or object-oriented databases, hierarchical databases, network databases, spreadsheets, multimedia databases, or file systems. The heterogeneous databases in a legacy database may be connected by intra- or inter-computer networks.

Information exchange across such databases is difficult because it would require precise transformation rules from one representation to another, considering diverse semantics. Consider, for example, the problem in exchanging information regarding student academic performance among different schools. Each school may have its own computer system and use its own curriculum and grading system. One university may adopt a quarter system, offer three courses on database systems, and assign grades from A+ to F, whereas another may adopt a semester system, offer two courses on databases, and assign grades from 1 to 10. It is very difficult to work out precise course-to-grade transformation rules between the two universities, making information exchange difficult. Data mining techniques may provide an interesting solution to the information exchange problem by performing statistical data distribution and correlation analysis, and transforming the given data into higher, more generalized, conceptual levels (such as *fair*, *good*, or *excellent* for student grades), from which information exchange can then more easily be performed.

## Data Streams

Many applications involve the generation and analysis of a new kind of data, called **stream data**, where data flow in and out of an observation platform (or window) dynamically. Such data streams have the following unique features: *huge or possibly infinite volume, dynamically changing, flowing in and out in a fixed order, allowing only one or a small*

*number of scans, and demanding fast (often real-time) response time.* Typical examples of data streams include various kinds of scientific and engineering data, time-series data, and data produced in other dynamic environments, such as power supply, network traffic, stock exchange, telecommunications, Web click streams, video surveillance, and weather or environment monitoring.

Because data streams are normally not stored in any kind of data repository, effective and efficient management and analysis of stream data poses great challenges to researchers. Currently, many researchers are investigating various issues relating to the development of data stream management systems. A typical query model in such a system is the *continuous query model*, where predefined queries constantly evaluate incoming streams, collect aggregate data, report the current status of data streams, and respond to their changes.

Mining data streams involves the efficient discovery of general patterns and dynamic changes within stream data. For example, we may like to detect intrusions of a computer network based on the anomaly of message flow, which may be discovered by clustering data streams, dynamic construction of stream models, or comparing the current frequent patterns with that at a certain previous time. Most stream data reside at a rather low level of abstraction, whereas analysts are often more interested in higher and multiple levels of abstraction. Thus, multilevel, multidimensional on-line analysis and mining should be performed on stream data as well.

## The World Wide Web

The World Wide Web and its associated distributed information services, such as Yahoo!, Google, America Online, and AltaVista, provide rich, worldwide, on-line information services, where data objects are linked together to facilitate interactive access. Users seeking information of interest traverse from one object via links to another. Such systems provide ample opportunities and challenges for data mining. For example, understanding user access patterns will not only help improve system design (by providing efficient access between highly correlated objects), but also leads to better marketing decisions (e.g., by placing advertisements in frequently visited documents, or by providing better customer/user classification and behavior analysis). Capturing user access patterns in such distributed information environments is called **Web usage mining** (or **Weblog mining**).

Although Web pages may appear fancy and informative to human readers, they can be highly unstructured and lack a predefined schema, type, or pattern. Thus it is difficult for computers to understand the semantic meaning of diverse Web pages and structure them in an organized way for systematic information retrieval and data mining. Web services that provide keyword-based searches without understanding the context behind the Web pages can only offer limited help to users. For example, a Web search based on a single keyword may return hundreds of Web page pointers containing the keyword, but most of the pointers will be very weakly related to what the user wants to find. Data mining can often provide additional help here than Web search services. For example, **authoritative Web page analysis** based on linkages among Web pages can help rank Web pages

based on their importance, influence, and topics. **Automated Web page clustering and classification** help group and arrange Web pages in a multidimensional manner based on their contents. **Web community analysis** helps identify hidden Web social networks and communities and observe their evolution. Web mining is the development of scalable and effective Web data analysis and mining methods. It may help us learn about the distribution of information on the Web in general, characterize and classify Web pages, and uncover Web dynamics and the association and other relationships among different Web pages, users, communities, and Web-based activities.

Data mining in advanced database and information systems is discussed in Chapters 8 to 10.

## 1.4 Data Mining Functionalities—What Kinds of Patterns Can Be Mined?

We have observed various types of databases and information repositories on which data mining can be performed. Let us now examine the kinds of data patterns that can be mined.

Data mining functionalities are used to specify the kind of patterns to be found in data mining tasks. In general, data mining tasks can be classified into two categories: **descriptive** and **predictive**. Descriptive mining tasks characterize the general properties of the data in the database. Predictive mining tasks perform inference on the current data in order to make predictions.

In some cases, users may have no idea regarding what kinds of patterns in their data may be interesting, and hence may like to search for several different kinds of patterns in parallel. Thus it is important to have a data mining system that can mine multiple kinds of patterns to accommodate different user expectations or applications. Furthermore, data mining systems should be able to discover patterns at various granularity (i.e., different levels of abstraction). Data mining systems should also allow users to specify hints to guide or focus the search for interesting patterns. Because some patterns may not hold for all of the data in the database, a measure of certainty or “trustworthiness” is usually associated with each discovered pattern.

Data mining functionalities, and the kinds of patterns they can discover, are described below.

### 1.4.1 Concept/Class Description: Characterization and Discrimination

Data can be associated with classes or concepts. For example, in the *AllElectronics* store, classes of items for sale include *computers* and *printers*, and concepts of customers include *bigSpenders* and *budgetSpenders*. It can be useful to describe individual classes and concepts in summarized, concise, and yet precise terms. Such descriptions of a class or a concept are called **class/concept descriptions**. These descriptions can be derived via (1) *data characterization*, by summarizing the data of the class under study (often called

the **target class**) in general terms, or (2) *data discrimination*, by comparison of the target class with one or a set of comparative classes (often called the **contrasting classes**), or (3) both data characterization and discrimination.

**Data characterization** is a summarization of the general characteristics or features of a target class of data. The data corresponding to the user-specified class are typically collected by a database query. For example, to study the characteristics of software products whose sales increased by 10% in the last year, the data related to such products can be collected by executing an SQL query.

There are several methods for effective data summarization and characterization. Simple data summaries based on statistical measures and plots are described in Chapter 2. The data cube-based OLAP roll-up operation (Section 1.3.2) can be used to perform user-controlled data summarization along a specified dimension. This process is further detailed in Chapters 3 and 4, which discuss data warehousing. An *attribute-oriented induction* technique can be used to perform data generalization and characterization without step-by-step user interaction. This technique is described in Chapter 4.

The output of data characterization can be presented in various forms. Examples include **pie charts**, **bar charts**, **curves**, **multidimensional data cubes**, and **multidimensional tables**, including crosstabs. The resulting descriptions can also be presented as **generalized relations** or in rule form (called **characteristic rules**). These different output forms and their transformations are discussed in Chapter 4.

**Example 1.4 Data characterization.** A data mining system should be able to produce a description summarizing the characteristics of customers who spend more than \$1,000 a year at *AllElectronics*. The result could be a general profile of the customers, such as they are 40–50 years old, employed, and have excellent credit ratings. The system should allow users to drill down on any dimension, such as on *occupation* in order to view these customers according to their type of employment. ■

**Data discrimination** is a comparison of the general features of target class data objects with the general features of objects from one or a set of contrasting classes. The target and contrasting classes can be specified by the user, and the corresponding data objects retrieved through database queries. For example, the user may like to compare the general features of software products whose sales increased by 10% in the last year with those whose sales decreased by at least 30% during the same period. The methods used for data discrimination are similar to those used for data characterization.

“How are discrimination descriptions output?” The forms of output presentation are similar to those for characteristic descriptions, although discrimination descriptions should include comparative measures that help distinguish between the target and contrasting classes. Discrimination descriptions expressed in rule form are referred to as **discriminant rules**.

**Example 1.5 Data discrimination.** A data mining system should be able to compare two groups of *AllElectronics* customers, such as those who shop for computer products regularly (more

than two times a month) versus those who rarely shop for such products (i.e., less than three times a year). The resulting description provides a general comparative profile of the customers, such as 80% of the customers who frequently purchase computer products are between 20 and 40 years old and have a university education, whereas 60% of the customers who infrequently buy such products are either seniors or youths, and have no university degree. Drilling down on a dimension, such as *occupation*, or adding new dimensions, such as *income\_level*, may help in finding even more discriminative features between the two classes. ■

Concept description, including characterization and discrimination, is described in Chapter 4.

### 1.4.2 Mining Frequent Patterns, Associations, and Correlations

**Frequent patterns**, as the name suggests, are patterns that occur frequently in data. There are many kinds of frequent patterns, including itemsets, subsequences, and substructures. A *frequent itemset* typically refers to a set of items that frequently appear together in a transactional data set, such as milk and bread. A frequently occurring subsequence, such as the pattern that customers tend to purchase first a PC, followed by a digital camera, and then a memory card, is a (*frequent*) *sequential pattern*. A substructure can refer to different structural forms, such as graphs, trees, or lattices, which may be combined with itemsets or subsequences. If a substructure occurs frequently, it is called a (*frequent*) *structured pattern*. Mining frequent patterns leads to the discovery of interesting associations and correlations within data.

**Example 1.6 Association analysis.** Suppose, as a marketing manager of *AllElectronics*, you would like to determine which items are frequently purchased together within the same transactions. An example of such a rule, mined from the *AllElectronics* transactional database, is

$$\text{buys}(X, \text{"computer"}) \Rightarrow \text{buys}(X, \text{"software"}) \quad [\text{support} = 1\%, \text{confidence} = 50\%]$$

where  $X$  is a variable representing a customer. A **confidence**, or certainty, of 50% means that if a customer buys a computer, there is a 50% chance that she will buy software as well. A 1% **support** means that 1% of all of the transactions under analysis showed that computer and software were purchased together. This association rule involves a single attribute or predicate (i.e., *buys*) that repeats. Association rules that contain a single predicate are referred to as **single-dimensional association rules**. Dropping the predicate notation, the above rule can be written simply as “*computer*  $\Rightarrow$  *software* [1%, 50%]”.

Suppose, instead, that we are given the *AllElectronics* relational database relating to purchases. A data mining system may find association rules like

$$\begin{aligned} \text{age}(X, \text{"20...29"}) \wedge \text{income}(X, \text{"20K...29K"}) &\Rightarrow \text{buys}(X, \text{"CD player"}) \\ &[\text{support} = 2\%, \text{confidence} = 60\%] \end{aligned}$$

The rule indicates that of the *AllElectronics* customers under study, 2% are 20 to 29 years of age with an income of 20,000 to 29,000 and have purchased a CD player

at *AllElectronics*. There is a 60% probability that a customer in this age and income group will purchase a CD player. Note that this is an association between more than one attribute, or predicate (i.e., *age*, *income*, and *buys*). Adopting the terminology used in multidimensional databases, where each attribute is referred to as a dimension, the above rule can be referred to as a **multidimensional association rule**. ■

Typically, association rules are discarded as uninteresting if they do not satisfy both a **minimum support threshold** and a **minimum confidence threshold**. Additional analysis can be performed to uncover interesting statistical **correlations** between associated attribute-value pairs.

*Frequent itemset mining* is the simplest form of frequent pattern mining. The mining of frequent patterns, associations, and correlations is discussed in Chapter 5, where particular emphasis is placed on efficient algorithms for frequent itemset mining. Sequential pattern mining and structured pattern mining are considered advanced topics. They are discussed in Chapters 8 and 9, respectively.

### 1.4.3 Classification and Prediction

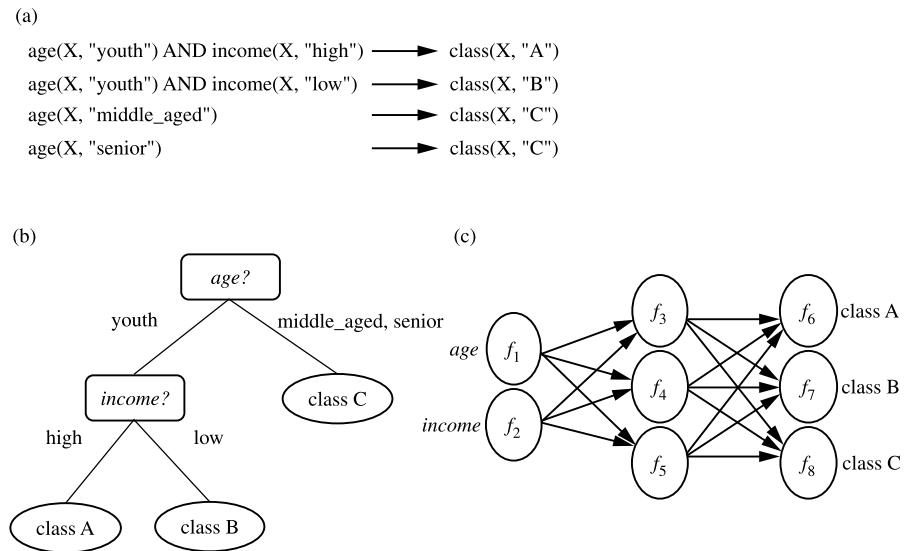
**Classification** is the process of finding a **model** (or function) that describes and distinguishes data classes or concepts, for the purpose of being able to use the model to predict the class of objects whose class label is unknown. The derived model is based on the analysis of a set of **training data** (i.e., data objects whose class label is known).

“How is the derived model presented?” The derived model may be represented in various forms, such as *classification (IF-THEN) rules*, *decision trees*, *mathematical formulae*, or *neural networks* (Figure 1.10). A **decision tree** is a flow-chart-like tree structure, where each node denotes a test on an attribute value, each branch represents an outcome of the test, and tree leaves represent classes or class distributions. Decision trees can easily be converted to classification rules. A **neural network**, when used for classification, is typically a collection of neuron-like processing units with weighted connections between the units. There are many other methods for constructing classification models, such as naïve Bayesian classification, support vector machines, and *k*-nearest neighbor classification.

Whereas classification predicts categorical (discrete, unordered) labels, **prediction** models continuous-valued functions. That is, it is used to predict missing or unavailable *numerical data values* rather than class labels. Although the term *prediction* may refer to both numeric prediction and class label prediction, in this book we use it to refer primarily to numeric prediction. **Regression analysis** is a statistical methodology that is most often used for numeric prediction, although other methods exist as well. Prediction also encompasses the identification of distribution *trends* based on the available data.

Classification and prediction may need to be preceded by **relevance analysis**, which attempts to identify attributes that do not contribute to the classification or prediction process. These attributes can then be excluded.

**Example 1.7** Classification and prediction. Suppose, as sales manager of *AllElectronics*, you would like to classify a large set of items in the store, based on three kinds of responses to a



**Figure 1.10** A classification model can be represented in various forms, such as (a) IF-THEN rules, (b) a decision tree, or a (c) neural network.

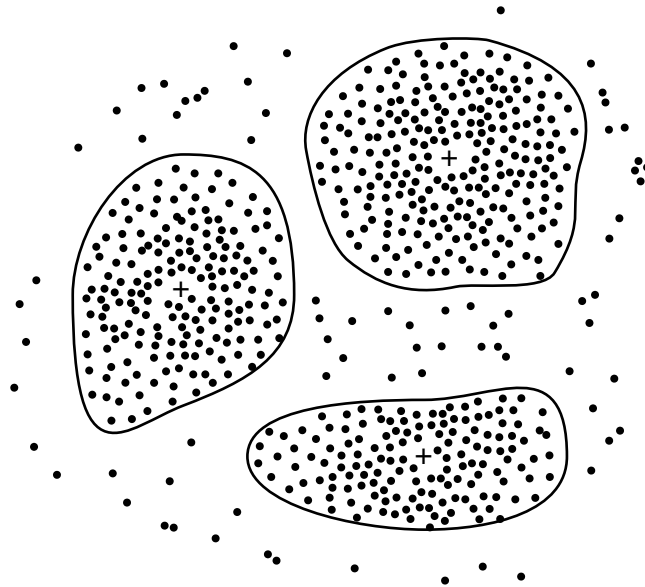
sales campaign: *good response*, *mild response*, and *no response*. You would like to derive a model for each of these three classes based on the descriptive features of the items, such as *price*, *brand*, *place\_made*, *type*, and *category*. The resulting classification should maximally distinguish each class from the others, presenting an organized picture of the data set. Suppose that the resulting classification is expressed in the form of a decision tree. The decision tree, for instance, may identify *price* as being the single factor that best distinguishes the three classes. The tree may reveal that, after *price*, other features that help further distinguish objects of each class from another include *brand* and *place\_made*. Such a decision tree may help you understand the impact of the given sales campaign and design a more effective campaign for the future.

Suppose instead, that rather than predicting categorical response labels for each store item, you would like to predict the amount of revenue that each item will generate during an upcoming sale at *AlIElectronics*, based on previous sales data. This is an example of (numeric) prediction because the model constructed will predict a continuous-valued function, or ordered value. ■

Chapter 6 discusses classification and prediction in further detail.

#### 1.4.4 Cluster Analysis

“What is cluster analysis?” Unlike classification and prediction, which analyze class-labeled data objects, **clustering** analyzes data objects without consulting a known class label.



**Figure 1.11** A 2-D plot of customer data with respect to customer locations in a city, showing three data clusters. Each cluster “center” is marked with a “+”.

In general, the class labels are not present in the training data simply because they are not known to begin with. Clustering can be used to generate such labels. The objects are clustered or grouped based on the principle of *maximizing the intraclass similarity and minimizing the interclass similarity*. That is, clusters of objects are formed so that objects within a cluster have high similarity in comparison to one another, but are very dissimilar to objects in other clusters. Each cluster that is formed can be viewed as a class of objects, from which rules can be derived. Clustering can also facilitate **taxonomy formation**, that is, the organization of observations into a hierarchy of classes that group similar events together.

**Example 1.8** **Cluster analysis.** Cluster analysis can be performed on *AllElectronics* customer data in order to identify homogeneous subpopulations of customers. These clusters may represent individual target groups for marketing. Figure 1.11 shows a 2-D plot of customers with respect to customer locations in a city. Three clusters of data points are evident. ■

Cluster analysis forms the topic of Chapter 7.

### 1.4.5 Outlier Analysis

A database may contain data objects that do not comply with the general behavior or model of the data. These data objects are **outliers**. Most data mining methods discard

outliers as noise or exceptions. However, in some applications such as fraud detection, the rare events can be more interesting than the more regularly occurring ones. The analysis of outlier data is referred to as **outlier mining**.

Outliers may be detected using statistical tests that assume a distribution or probability model for the data, or using distance measures where objects that are a substantial distance from any other cluster are considered outliers. Rather than using statistical or distance measures, deviation-based methods identify outliers by examining differences in the main characteristics of objects in a group.

**Example 1.9 Outlier analysis.** Outlier analysis may uncover fraudulent usage of credit cards by detecting purchases of extremely large amounts for a given account number in comparison to regular charges incurred by the same account. Outlier values may also be detected with respect to the location and type of purchase, or the purchase frequency. ■

Outlier analysis is also discussed in Chapter 7.

### 1.4.6 Evolution Analysis

Data **evolution analysis** describes and models regularities or trends for objects whose behavior changes over time. Although this may include characterization, discrimination, association and correlation analysis, classification, prediction, or clustering of *time-related* data, distinct features of such an analysis include time-series data analysis, sequence or periodicity pattern matching, and similarity-based data analysis.

**Example 1.10 Evolution analysis.** Suppose that you have the major stock market (time-series) data of the last several years available from the New York Stock Exchange and you would like to invest in shares of high-tech industrial companies. A data mining study of stock exchange data may identify stock evolution regularities for overall stocks and for the stocks of particular companies. Such regularities may help predict future trends in stock market prices, contributing to your decision making regarding stock investments. ■

Data evolution analysis is discussed in Chapter 8.

## 1.5 Are All of the Patterns Interesting?

A data mining system has the potential to generate thousands or even millions of patterns, or rules.

“So,” you may ask, “*are all of the patterns interesting?*” Typically not—only a small fraction of the patterns potentially generated would actually be of interest to any given user.

This raises some serious questions for data mining. You may wonder, “*What makes a pattern interesting? Can a data mining system generate all of the interesting patterns? Can a data mining system generate only interesting patterns?*”

To answer the first question, a pattern is **interesting** if it is (1) *easily understood* by humans, (2) *valid* on new or test data with some degree of *certainty*, (3) potentially *useful*,

and (4) *novel*. A pattern is also interesting if it validates a hypothesis that the user *sought to confirm*. An interesting pattern represents **knowledge**.

Several **objective measures of pattern interestingness** exist. These are based on the structure of discovered patterns and the statistics underlying them. An objective measure for association rules of the form  $X \Rightarrow Y$  is rule **support**, representing the percentage of transactions from a transaction database that the given rule satisfies. This is taken to be the probability  $P(X \cup Y)$ , where  $X \cup Y$  indicates that a transaction contains both  $X$  and  $Y$ , that is, the union of itemsets  $X$  and  $Y$ . Another objective measure for association rules is **confidence**, which assesses the degree of certainty of the detected association. This is taken to be the conditional probability  $P(Y|X)$ , that is, the probability that a transaction containing  $X$  also contains  $Y$ . More formally, support and confidence are defined as

$$\text{support}(X \Rightarrow Y) = P(X \cup Y).$$

$$\text{confidence}(X \Rightarrow Y) = P(Y|X).$$

In general, each interestingness measure is associated with a threshold, which may be controlled by the user. For example, rules that do not satisfy a confidence threshold of, say, 50% can be considered uninteresting. Rules below the threshold likely reflect noise, exceptions, or minority cases and are probably of less value.

Although objective measures help identify interesting patterns, they are insufficient unless combined with subjective measures that reflect the needs and interests of a particular user. For example, patterns describing the characteristics of customers who shop frequently at *AllElectronics* should interest the marketing manager, but may be of little interest to analysts studying the same database for patterns on employee performance. Furthermore, many patterns that are interesting by objective standards may represent common knowledge and, therefore, are actually uninteresting. **Subjective interestingness measures** are based on user beliefs in the data. These measures find patterns interesting if they are **unexpected** (contradicting a user's belief) or offer strategic information on which the user can act. In the latter case, such patterns are referred to as **actionable**. Patterns that are **expected** can be interesting if they confirm a hypothesis that the user wished to validate, or resemble a user's hunch.

The second question—“*Can a data mining system generate all of the interesting patterns?*”—refers to the **completeness** of a data mining algorithm. It is often unrealistic and inefficient for data mining systems to generate all of the possible patterns. Instead, user-provided constraints and interestingness measures should be used to focus the search. For some mining tasks, such as association, this is often sufficient to ensure the completeness of the algorithm. Association rule mining is an example where the use of constraints and interestingness measures can ensure the completeness of mining. The methods involved are examined in detail in Chapter 5.

Finally, the third question—“*Can a data mining system generate only interesting patterns?*”—is an optimization problem in data mining. It is highly desirable for data mining systems to generate only interesting patterns. This would be much more efficient for users and data mining systems, because neither would have to search through the patterns generated in order to identify the truly interesting ones. Progress has been made in this direction; however, such optimization remains a challenging issue in data mining.

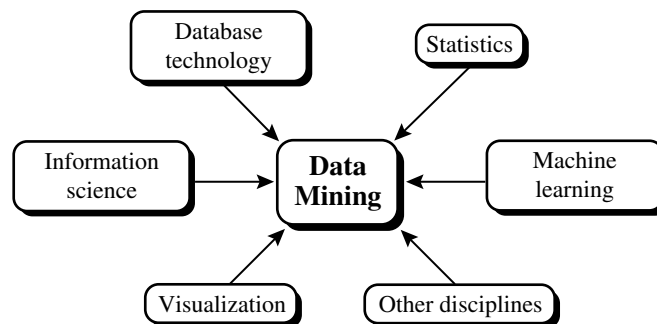
Measures of pattern interestingness are essential for the efficient discovery of patterns of value to the given user. Such measures can be used after the data mining step in order to rank the discovered patterns according to their interestingness, filtering out the uninteresting ones. More importantly, such measures can be used to guide and constrain the discovery process, improving the search efficiency by pruning away subsets of the pattern space that do not satisfy prespecified interestingness constraints. Such constraint-based mining is described in Chapter 5 (with respect to association mining) and Chapter 7 (with respect to clustering).

Methods to assess pattern interestingness, and their use to improve data mining efficiency, are discussed throughout the book, with respect to each kind of pattern that can be mined.

## 1.6 Classification of Data Mining Systems

Data mining is an interdisciplinary field, the confluence of a set of disciplines, including database systems, statistics, machine learning, visualization, and information science (Figure 1.12). Moreover, depending on the data mining approach used, techniques from other disciplines may be applied, such as neural networks, fuzzy and/or rough set theory, knowledge representation, inductive logic programming, or high-performance computing. Depending on the kinds of data to be mined or on the given data mining application, the data mining system may also integrate techniques from spatial data analysis, information retrieval, pattern recognition, image analysis, signal processing, computer graphics, Web technology, economics, business, bioinformatics, or psychology.

Because of the diversity of disciplines contributing to data mining, data mining research is expected to generate a large variety of data mining systems. Therefore, it is necessary to provide a clear classification of data mining systems, which may help potential users distinguish between such systems and identify those that best match their needs. Data mining systems can be categorized according to various criteria, as follows:



**Figure 1.12** Data mining as a confluence of multiple disciplines.

Classification according to the *kinds of databases* mined: A data mining system can be classified according to the kinds of databases mined. Database systems can be classified according to different criteria (such as data models, or the types of data or applications involved), each of which may require its own data mining technique. Data mining systems can therefore be classified accordingly.

For instance, if classifying according to data models, we may have a relational, transactional, object-relational, or data warehouse mining system. If classifying according to the special types of data handled, we may have a spatial, time-series, text, stream data, multimedia data mining system, or a World Wide Web mining system.

Classification according to the *kinds of knowledge* mined: Data mining systems can be categorized according to the kinds of knowledge they mine, that is, based on data mining functionalities, such as characterization, discrimination, association and correlation analysis, classification, prediction, clustering, outlier analysis, and evolution analysis. A comprehensive data mining system usually provides multiple and/or integrated data mining functionalities.

Moreover, data mining systems can be distinguished based on the granularity or levels of abstraction of the knowledge mined, including generalized knowledge (at a high level of abstraction), primitive-level knowledge (at a raw data level), or knowledge at multiple levels (considering several levels of abstraction). An advanced data mining system should facilitate the discovery of knowledge at multiple levels of abstraction.

Data mining systems can also be categorized as those that mine data regularities (commonly occurring patterns) versus those that mine data irregularities (such as exceptions, or outliers). In general, concept description, association and correlation analysis, classification, prediction, and clustering mine data regularities, rejecting outliers as noise. These methods may also help detect outliers.

Classification according to the *kinds of techniques* utilized: Data mining systems can be categorized according to the underlying data mining techniques employed. These techniques can be described according to the degree of user interaction involved (e.g., autonomous systems, interactive exploratory systems, query-driven systems) or the methods of data analysis employed (e.g., database-oriented or data warehouse-oriented techniques, machine learning, statistics, visualization, pattern recognition, neural networks, and so on). A sophisticated data mining system will often adopt multiple data mining techniques or work out an effective, integrated technique that combines the merits of a few individual approaches.

Classification according to the *applications adapted*: Data mining systems can also be categorized according to the applications they adapt. For example, data mining systems may be tailored specifically for finance, telecommunications, DNA, stock markets, e-mail, and so on. Different applications often require the integration of application-specific methods. Therefore, a generic, all-purpose data mining system may not fit domain-specific mining tasks.

In general, Chapters 4 to 7 of this book are organized according to the various kinds of knowledge mined. In Chapters 8 to 10, we discuss the mining of complex types of

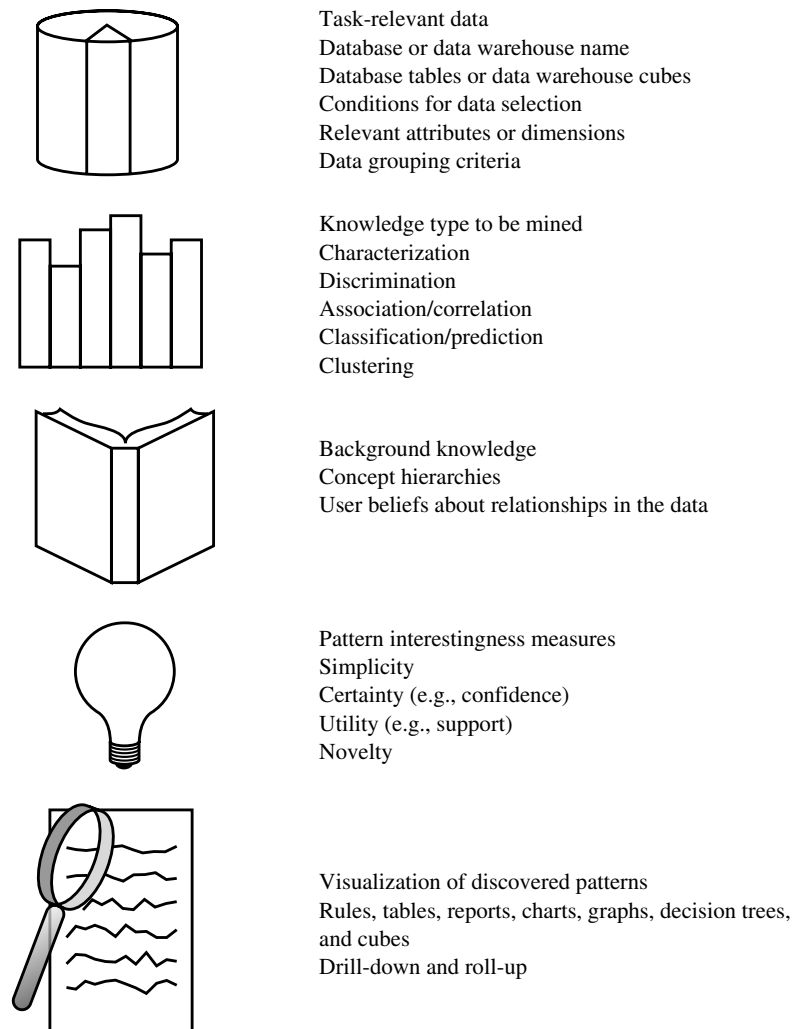
data on a variety of advanced database systems. Chapter 11 describes major data mining applications as well as typical commercial data mining systems. Criteria for choosing a data mining system are also provided.

## 1.7 Data Mining Task Primitives

Each user will have a **data mining task** in mind, that is, some form of data analysis that he or she would like to have performed. A data mining task can be specified in the form of a **data mining query**, which is input to the data mining system. A data mining query is defined in terms of **data mining task primitives**. These primitives allow the user to *interactively* communicate with the data mining system during discovery in order to direct the mining process, or examine the findings from different angles or depths. The data mining primitives specify the following, as illustrated in Figure 1.13.

- The set of *task-relevant data* to be mined: This specifies the portions of the database or the set of data in which the user is interested. This includes the database attributes or data warehouse dimensions of interest (referred to as the *relevant attributes or dimensions*).
- The *kind of knowledge* to be mined: This specifies the *data mining functions* to be performed, such as characterization, discrimination, association or correlation analysis, classification, prediction, clustering, outlier analysis, or evolution analysis.
- The *background knowledge* to be used in the discovery process: This knowledge about the domain to be mined is useful for guiding the knowledge discovery process and for evaluating the patterns found. *Concept hierarchies* are a popular form of background knowledge, which allow data to be mined at multiple levels of abstraction. An example of a concept hierarchy for the attribute (or dimension) *age* is shown in Figure 1.14. User beliefs regarding relationships in the data are another form of background knowledge.
- The *interestingness measures and thresholds* for pattern evaluation: They may be used to guide the mining process or, after discovery, to evaluate the discovered patterns. Different kinds of knowledge may have different interestingness measures. For example, interestingness measures for association rules include *support* and *confidence*. Rules whose support and confidence values are below user-specified thresholds are considered uninteresting.
- The expected *representation for visualizing* the discovered patterns: This refers to the form in which discovered patterns are to be displayed, which may include rules, tables, charts, graphs, decision trees, and cubes.

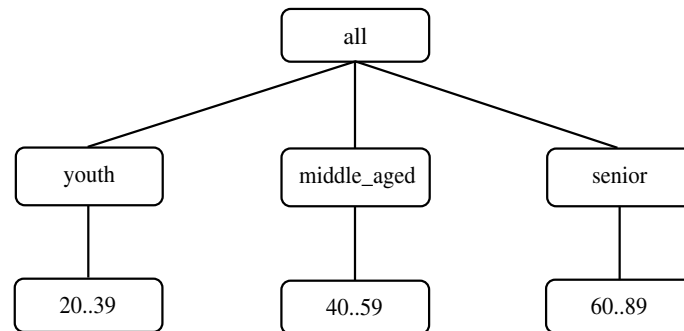
A **data mining query language** can be designed to incorporate these primitives, allowing users to flexibly interact with data mining systems. Having a data mining query language provides a foundation on which user-friendly graphical interfaces can be built.



**Figure 1.13** Primitives for specifying a data mining task.

This facilitates a data mining system's communication with other information systems and its integration with the overall information processing environment.

Designing a comprehensive data mining language is challenging because data mining covers a wide spectrum of tasks, from data characterization to evolution analysis. Each task has different requirements. The design of an effective data mining query language requires a deep understanding of the power, limitation, and underlying mechanisms of the various kinds of data mining tasks.



**Figure 1.14** A concept hierarchy for the attribute (or dimension) *age*. The root node represents the most general abstraction level, denoted as *all*.

There are several proposals on data mining languages and standards. In this book, we use a data mining query language known as **DMQL** (Data Mining Query Language), which was designed as a teaching tool, based on the above primitives. Examples of its use to specify data mining queries appear throughout this book. The language adopts an SQL-like syntax, so that it can easily be integrated with the relational query language, SQL. Let's look at how it can be used to specify a data mining task.

**Example 1.11 Mining classification rules.** Suppose, as a marketing manager of *AllElectronics*, you would like to classify customers based on their buying patterns. You are especially interested in those customers whose salary is no less than \$40,000, and who have bought more than \$1,000 worth of items, each of which is priced at no less than \$100. In particular, you are interested in the customer's age, income, the types of items purchased, the purchase location, and where the items were made. You would like to view the resulting classification in the form of rules. This data mining query is expressed in DMQL<sup>3</sup> as follows, where each line of the query has been enumerated to aid in our discussion.

- (1) **use database** AllElectronics\_db
- (2) **use hierarchy** location\_hierarchy for T.branch, age\_hierarchy for C.age
- (3) **mine classification as** promising\_customers
- (4) **in relevance to** C.age, C.income, I.type, I.place\_made, T.branch
- (5) **from** customer C, item I, transaction T
- (6) **where** I.item\_ID = T.item\_ID and C.cust\_ID = T.cust\_ID  
and C.income  $\geq$  40,000 and I.price  $\geq$  100
- (7) **group by** T.cust\_ID

<sup>3</sup>Note that in this book, query language keywords are displayed in sans serif font.

- (8) **having** sum(I.price)  $\geq$  1,000
- (9) **display as rules**

The data mining query is parsed to form an SQL query that retrieves the set of task-relevant data specified by lines 1 and 4 to 8. That is, line 1 specifies the *All-Electronics* database, line 4 lists the relevant attributes (i.e., on which mining is to be performed) from the relations specified in line 5 for the conditions given in lines 6 and 7. Line 2 specifies that the concept hierarchies *location\_hierarchy* and *age\_hierarchy* be used as background knowledge to generalize branch locations and customer age values, respectively. Line 3 specifies that the kind of knowledge to be mined for this task is classification. Note that we want to generate a classification model for “promising\_customers” versus “non\_promising\_customers.” In classification, often, an attribute may be specified as the **class label attribute**, whose values *explicitly* represent the classes. However, in this example, the two classes are *implicit*. That is, the specified data are retrieved and considered examples of “promising\_customers,” whereas the remaining customers in the customer table are considered as “non-promising\_customers.” Classification is performed based on this training set. Line 9 specifies that the mining results are to be displayed as a set of rules. Several detailed classification methods are introduced in Chapter 6. ■

There is no standard data mining query language as of yet; however, researchers and industry have been making good progress in this direction. Microsoft’s OLE DB for Data Mining (described in the appendix of this book) includes DMX, an XML-styled data mining language. Other standardization efforts include PMML (Programming data Model Markup Language) and CRISP-DM (CRoss-Industry Standard Process for Data Mining).

## 1.8 Integration of a Data Mining System with a Database or Data Warehouse System

Section 1.2 outlined the major components of the architecture for a typical data mining system (Figure 1.5). A good system architecture will facilitate the data mining system to make best use of the software environment, accomplish data mining tasks in an efficient and timely manner, interoperate and exchange information with other information systems, be adaptable to users’ diverse requirements, and evolve with time.

A critical question in the design of a data mining (DM) system is how to integrate or *couple* the DM system with a database (DB) system and/or a data warehouse (DW) system. If a DM system works as a stand-alone system or is embedded in an application program, there are no DB or DW systems with which it has to communicate. This simple scheme is called *no coupling*, where the main focus of the DM design rests on developing effective and efficient algorithms for mining the available data sets. However, when a DM system works in an environment that requires it to communicate with other information system components, such as DB and DW systems, possible integration schemes include

*no coupling*, *loose coupling*, *semitight coupling*, and *tight coupling*. We examine each of these schemes, as follows:

- **No coupling:** *No coupling* means that a DM system will not utilize any function of a DB or DW system. It may fetch data from a particular source (such as a file system), process data using some data mining algorithms, and then store the mining results in another file.

Such a system, though simple, suffers from several drawbacks. First, a DB system provides a great deal of flexibility and efficiency at storing, organizing, accessing, and processing data. Without using a DB/DW system, a DM system may spend a substantial amount of time finding, collecting, cleaning, and transforming data. In DB and/or DW systems, data tend to be well organized, indexed, cleaned, integrated, or consolidated, so that finding the task-relevant, high-quality data becomes an easy task. Second, there are many tested, scalable algorithms and data structures implemented in DB and DW systems. It is feasible to realize efficient, scalable implementations using such systems. Moreover, most data have been or will be stored in DB/DW systems. Without any coupling of such systems, a DM system will need to use other tools to extract data, making it difficult to integrate such a system into an information processing environment. Thus, no coupling represents a poor design.

- **Loose coupling:** *Loose coupling* means that a DM system will use some facilities of a DB or DW system, fetching data from a data repository managed by these systems, performing data mining, and then storing the mining results either in a file or in a designated place in a database or data warehouse.

Loose coupling is better than no coupling because it can fetch any portion of data stored in databases or data warehouses by using query processing, indexing, and other system facilities. It incurs some advantages of the flexibility, efficiency, and other features provided by such systems. However, many loosely coupled mining systems are main memory-based. Because mining does not explore data structures and query optimization methods provided by DB or DW systems, it is difficult for loose coupling to achieve high scalability and good performance with large data sets.

- **Semitight coupling:** *Semitight coupling* means that besides linking a DM system to a DB/DW system, efficient implementations of a few essential data mining primitives (identified by the analysis of frequently encountered data mining functions) can be provided in the DB/DW system. These primitives can include sorting, indexing, aggregation, histogram analysis, multiway join, and precomputation of some essential statistical measures, such as sum, count, max, min, standard deviation, and so on. Moreover, some frequently used intermediate mining results can be precomputed and stored in the DB/DW system. Because these intermediate mining results are either precomputed or can be computed efficiently, this design will enhance the performance of a DM system.
- **Tight coupling:** *Tight coupling* means that a DM system is smoothly integrated into the DB/DW system. The data mining subsystem is treated as one functional

component of an information system. Data mining queries and functions are optimized based on mining query analysis, data structures, indexing schemes, and query processing methods of a DB or DW system. With further technology advances, DM, DB, and DW systems will evolve and integrate together as one information system with multiple functionalities. This will provide a uniform information processing environment.

This approach is highly desirable because it facilitates efficient implementations of data mining functions, high system performance, and an integrated information processing environment.

With this analysis, it is easy to see that a data mining system should be coupled with a DB/DW system. Loose coupling, though not efficient, is better than no coupling because it uses both data and system facilities of a DB/DW system. Tight coupling is highly desirable, but its implementation is nontrivial and more research is needed in this area. Semitight coupling is a compromise between loose and tight coupling. It is important to identify commonly used data mining primitives and provide efficient implementations of such primitives in DB or DW systems.

## 1.9 Major Issues in Data Mining

The scope of this book addresses major issues in data mining regarding mining methodology, user interaction, performance, and diverse data types. These issues are introduced below:

**Mining methodology and user interaction issues:** These reflect the kinds of knowledge mined, the ability to mine knowledge at multiple granularities, the use of domain knowledge, ad hoc mining, and knowledge visualization.

- *Mining different kinds of knowledge in databases:* Because different users can be interested in different kinds of knowledge, data mining should cover a wide spectrum of data analysis and knowledge discovery tasks, including data characterization, discrimination, association and correlation analysis, classification, prediction, clustering, outlier analysis, and evolution analysis (which includes trend and similarity analysis). These tasks may use the same database in different ways and require the development of numerous data mining techniques.
- *Interactive mining of knowledge at multiple levels of abstraction:* Because it is difficult to know exactly what can be discovered within a database, the data mining process should be *interactive*. For databases containing a huge amount of data, appropriate sampling techniques can first be applied to facilitate interactive data exploration. Interactive mining allows users to focus the search for patterns, providing and refining data mining requests based on returned results. Specifically, knowledge should be mined by drilling down, rolling up,

and pivoting through the data space and knowledge space interactively, similar to what OLAP can do on data cubes. In this way, the user can interact with the data mining system to view data and discovered patterns at multiple granularities and from different angles.

- *Incorporation of background knowledge:* Background knowledge, or information regarding the domain under study, may be used to guide the discovery process and allow discovered patterns to be expressed in concise terms and at different levels of abstraction. Domain knowledge related to databases, such as integrity constraints and deduction rules, can help focus and speed up a data mining process, or judge the interestingness of discovered patterns.
- *Data mining query languages and ad hoc data mining:* Relational query languages (such as SQL) allow users to pose ad hoc queries for data retrieval. In a similar vein, high-level data mining query languages need to be developed to allow users to describe ad hoc data mining tasks by facilitating the specification of the relevant sets of data for analysis, the domain knowledge, the kinds of knowledge to be mined, and the conditions and constraints to be enforced on the discovered patterns. Such a language should be integrated with a database or data warehouse query language and optimized for efficient and flexible data mining.
- *Presentation and visualization of data mining results:* Discovered knowledge should be expressed in high-level languages, visual representations, or other expressive forms so that the knowledge can be easily understood and directly usable by humans. This is especially crucial if the data mining system is to be interactive. This requires the system to adopt expressive knowledge representation techniques, such as trees, tables, rules, graphs, charts, crosstabs, matrices, or curves.
- *Handling noisy or incomplete data:* The data stored in a database may reflect noise, exceptional cases, or incomplete data objects. When mining data regularities, these objects may confuse the process, causing the knowledge model constructed to overfit the data. As a result, the accuracy of the discovered patterns can be poor. Data cleaning methods and data analysis methods that can handle noise are required, as well as outlier mining methods for the discovery and analysis of exceptional cases.
- *Pattern evaluation—the interestingness problem:* A data mining system can uncover thousands of patterns. Many of the patterns discovered may be uninteresting to the given user, either because they represent common knowledge or lack novelty. Several challenges remain regarding the development of techniques to assess the interestingness of discovered patterns, particularly with regard to subjective measures that estimate the value of patterns with respect to a given user class, based on user beliefs or expectations. The use of interestingness measures or user-specified constraints to guide the discovery process and reduce the search space is another active area of research.

**Performance issues:** These include efficiency, scalability, and parallelization of data mining algorithms.

- *Efficiency and scalability of data mining algorithms:* To effectively extract information from a huge amount of data in databases, data mining algorithms must be efficient and scalable. In other words, the running time of a data mining algorithm must be predictable and acceptable in large databases. From a database perspective on knowledge discovery, efficiency and scalability are key issues in the implementation of data mining systems. Many of the issues discussed above under *mining methodology and user interaction* must also consider efficiency and scalability.
- *Parallel, distributed, and incremental mining algorithms:* The huge size of many databases, the wide distribution of data, and the computational complexity of some data mining methods are factors motivating the development of **parallel and distributed data mining algorithms**. Such algorithms divide the data into partitions, which are processed in parallel. The results from the partitions are then merged. Moreover, the high cost of some data mining processes promotes the need for **incremental** data mining algorithms that incorporate database updates without having to mine the entire data again “from scratch.” Such algorithms perform knowledge modification incrementally to amend and strengthen what was previously discovered.

**Issues relating to the diversity of database types:**

- *Handling of relational and complex types of data:* Because relational databases and data warehouses are widely used, the development of efficient and effective data mining systems for such data is important. However, other databases may contain complex data objects, hypertext and multimedia data, spatial data, temporal data, or transaction data. It is unrealistic to expect one system to mine all kinds of data, given the diversity of data types and different goals of data mining. Specific data mining systems should be constructed for mining specific kinds of data. Therefore, one may expect to have different data mining systems for different kinds of data.
- *Mining information from heterogeneous databases and global information systems:* Local- and wide-area computer networks (such as the Internet) connect many sources of data, forming huge, distributed, and heterogeneous databases. The discovery of knowledge from different sources of structured, semistructured, or unstructured data with diverse data semantics poses great challenges to data mining. Data mining may help disclose high-level data regularities in multiple heterogeneous databases that are unlikely to be discovered by simple query systems and may improve information exchange and interoperability in heterogeneous databases. Web mining, which uncovers interesting knowledge about Web contents, Web structures, Web usage, and Web dynamics, becomes a very challenging and fast-evolving field in data mining.

The above issues are considered major requirements and challenges for the further evolution of data mining technology. Some of the challenges have been addressed in recent data mining research and development, *to a certain extent*, and are now considered *requirements*, while others are still at the research stage. The issues, however, continue to stimulate further investigation and improvement. Additional issues relating to applications, privacy, and the social impacts of data mining are discussed in Chapter 11, the final chapter of this book.

## 1.10 Summary

- **Database technology** has evolved from primitive file processing to the development of database management systems with query and transaction processing. Further progress has led to the increasing demand for efficient and effective advanced data analysis tools. This need is a result of the explosive growth in data collected from applications, including business and management, government administration, science and engineering, and environmental control.
- **Data mining** is the task of discovering interesting patterns from large amounts of data, where the data can be stored in databases, data warehouses, or other information repositories. It is a young interdisciplinary field, drawing from areas such as database systems, data warehousing, statistics, machine learning, data visualization, information retrieval, and high-performance computing. Other contributing areas include neural networks, pattern recognition, spatial data analysis, image databases, signal processing, and many application fields, such as business, economics, and bioinformatics.
- A **knowledge discovery process** includes data cleaning, data integration, data selection, data transformation, data mining, pattern evaluation, and knowledge presentation.
- The **architecture** of a typical data mining system includes a database and/or data warehouse and their appropriate servers, a data mining engine and pattern evaluation module (both of which interact with a knowledge base), and a graphical user interface. **Integration** of the data mining components, as a whole, with a database or data warehouse system can involve either no coupling, loose coupling, semitight coupling, or tight coupling. A well-designed data mining system should offer tight or semitight coupling with a database and/or data warehouse system.
- Data patterns can be mined from many different kinds of **databases**, such as relational databases, data warehouses, and transactional, and object-relational databases. Interesting data patterns can also be extracted from other kinds of **information repositories**, including spatial, time-series, sequence, text, multimedia, and legacy databases, data streams, and the World Wide Web.
- A **data warehouse** is a repository for long-term storage of data from multiple sources, organized so as to facilitate management decision making. The data are stored under

a unified schema and are typically summarized. Data warehouse systems provide some data analysis capabilities, collectively referred to as **OLAP (on-line analytical processing)**.

- **Data mining functionalities** include the discovery of concept/class descriptions, associations and correlations, classification, prediction, clustering, trend analysis, outlier and deviation analysis, and similarity analysis. Characterization and discrimination are forms of data summarization.
- A pattern represents **knowledge** if it is easily understood by humans; valid on test data with some degree of certainty; and potentially useful, novel, or validates a hunch about which the user was curious. Measures of **pattern interestingness**, either *objective* or *subjective*, can be used to guide the discovery process.
- **Data mining systems** can be **classified** according to the kinds of databases mined, the kinds of knowledge mined, the techniques used, or the applications adapted.
- We have studied five **primitives** for specifying a data mining task in the form of a **data mining query**. These primitives are the specification of task-relevant data (i.e., the data set to be mined), the kind of knowledge to be mined, background knowledge (typically in the form of concept hierarchies), interestingness measures, and knowledge presentation and visualization techniques to be used for displaying the discovered patterns.
- **Data mining query languages** can be designed to support ad hoc and interactive data mining. A data mining query language, such as DMQL, should provide commands for specifying each of the data mining primitives. Such query languages are SQL-based and may eventually form a standard on which graphical user interfaces for data mining can be based.
- Efficient and effective data mining in large databases poses numerous requirements and great challenges to researchers and developers. The issues involved include data mining methodology, user interaction, performance and scalability, and the processing of a large variety of data types. Other issues include the exploration of data mining applications and their social impacts.

## Exercises

1.1 What is *data mining*? In your answer, address the following:

- (a) Is it another hype?
- (b) Is it a simple transformation of technology developed from databases, statistics, and machine learning?
- (c) Explain how the evolution of database technology led to data mining.
- (d) Describe the steps involved in data mining when viewed as a process of knowledge discovery.