University of Technology
الجامعة التكنولوجية

Computer Science Department
قسم علوم الحاسوب

Data Warehouse & Data Mining
تعدين البيانات ومستودعات البيانات

Asst. Prof. Dr. Khalil I. Ghathwan
أ.م.د. خليل ابراهيم غثوان

cs.uotechnology.edu.iq

1- History of Data

The past couple of decades have seen a dramatic increase in the amount of information or data being stored in electronic format. This accumulation of data has taken place at an explosive rate. It has been estimated that the amount of information in the world doubles every 20 months and the sizes as well as number of databases are increasing even faster. There are many examples that can be cited. Point of sale data in retail, policy and claim data in insurance, medical history data in healthcare, financial data in banking and securities, are some instances of the types of data that is being collected.

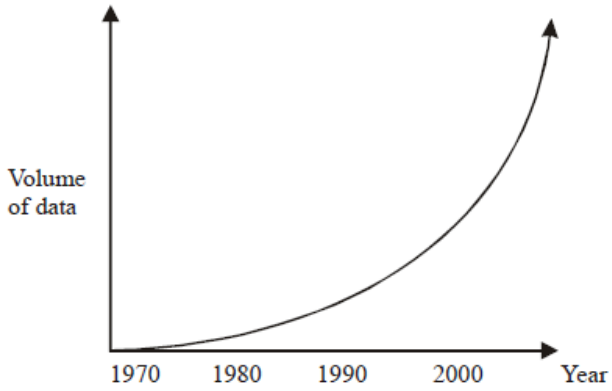


Fig. 1.1 The growing base of data

2- History of data warehousing

The concept of data warehousing dates back to the late 1980s when IBM researchers Barry Devlin and Paul Murphy developed the "business data warehouse". In essence, the data warehousing concept was intended to provide an architectural model for the flow of data from **operational systems** to **decision support environments**. The concept attempted to address the various problems associated with this flow, mainly the high costs associated with it. In the absence of a data warehousing architecture, an

enormous amount of **redundancy** was required to **support multiple decision support** environments. In larger corporations it was typical for multiple decision support environments to operate independently. Though each environment served different users, they often required much of the same stored data. The process of gathering, cleaning and integrating data from various sources, usually from long-term existing operational systems (usually referred to as legacy systems), was typically in part replicated for each environment.

3- What is a Data Warehouse?

A data warehouse is a **relational database** that is designed for query and analysis rather than for transaction processing. It usually contains historical data derived from transaction data, but it can include data from other sources.

In addition to a relational database, a data warehouse environment includes an extraction, transportation a transformation, and an online analytical processing (OLAP) engine, client analysis tools, and other applications that manage the process of gathering data and delivering it to business users.

A common way of introducing data warehousing is to refer to the characteristics of a data warehouse as set forth by William In mon:

- Subject Oriented
- Integrated
- Nonvolatile
- Time Variant

3-1 Subject Oriented

Data warehouses are designed to help you analyze data. For example, to learn more about your company's sales data, you can build a warehouse that concentrates on

sales. Using this warehouse, you can answer questions like "Who was our best customer for this item last year?" This ability to define a data warehouse by subject matter, sales in this case, makes the data warehouse subject oriented.

3-2 Integrated

Integration is closely related to subject orientation. Data warehouses must put data from disparate sources into a consistent format. They must resolve such problems as naming conflicts and inconsistencies among units of measure. When they achieve this, they are said to be integrated. For example, gender field might be represented differently from one transactional system to another. i.e. X/Y in human resources system or 0/1 in the salary system, while the data warehouse has one consistent format like M/F as illustrated in figure 2.

OLTP systems		DW system
Gender in Human Resources (HR) system: X/Y	Gender in Salary System: 0/1	Gender in data Warehouse (DW) system: M/F

Figure 2: Same attribute with different formats in dissimilar systems.

3-3 Nonvolatile

The final crucial characteristic of the data warehouse is Non-Volatile, data inside operational systems are updated (Insert, Delete, or Read and Write) also known as “CRUD”. While, data warehouse data are considered static or not up-to-date but only for Read. DW refreshed from time to time in a batch mode. This property enables DW

to be heavily optimized for query processing; figure 3 illustrates a simple comparison of this feature between operational and informational systems.

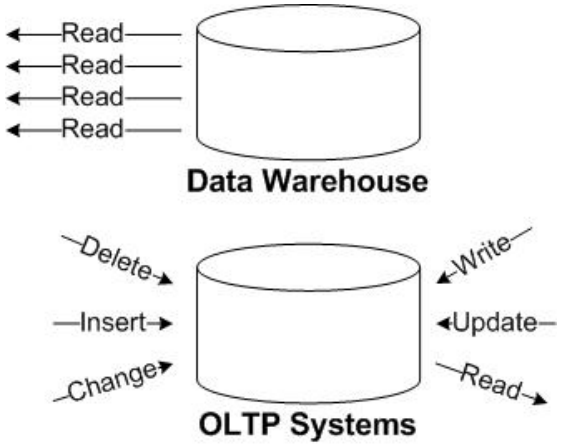


Figure 3: Simple comparison between OLTP and DW.

3-4 Time Variant

Operational systems store current up-to-date data for fast and atomic transactions. Old data transferred to archives since it affects performance, while the data warehouse contains a large amount of historical information that helps the analysts to discover trends in business. Every entry in the data warehouse is time-stamped through time-dimension. Noteworthy, data in OLTP systems maintained for few months or less than a year, while DW data kept for over three years (3 - 10).

4- Reasons for building Data warehouse

1. Improving integration

An organization registers data in different systems, which support the various business processes. In order to create an overall picture of business operations, customers and suppliers thus creating a single version of the truth the data must come together in one place and made compatible.

2. Speeding up response times

The source systems are fully optimized in order to process many small transactions, such as orders, in a short time.

3. Faster and more flexible reporting

The structure of both data warehouses and data marts enables end users to report in a flexible manner and to quickly perform interactive analyses on the basis of various predefined angles (dimensions).

4. Recording changes to build history

Source systems do not usually keep a history of certain data.

5. Increasing data quality

Stakeholders and users frequently overestimate the quality of data in the source systems. Unfortunately, source systems quite often contain data of poor quality. When we use a data warehouse, we can greatly improve the data quality,

6. Unburdening the IT department

A data warehouse and Business Intelligence tools allow employees within the organization to create reports and perform analyses independently.

7. Increasing recognisability

8. Increasing findability

When we create a data warehouse, we make sure that users can easily access the meaning of data. With a data warehouse users can find data more quickly and thus establish information and knowledge faster.

5- Granularity

Granularity refers to the level of detail or summarization of the units of data in the data warehouse. The more detail there is, the lower the level of granularity. The less detail there is, the higher the level of granularity.

Granularity is the major design issue in the data warehouse environment because it profoundly affects the volume of data that resides in the data warehouse and the type of query that can be answered. The volume of data in a warehouse is traded off against the level of detail of a query.

In almost all cases, data comes into the data warehouse at too high a level of granularity. This means that the developer must spend a lot of resources breaking the data apart. Occasionally, though, data enters the warehouse at too low a level of granularity. An example of data at too low a level of granularity is the Web log data generated by the Web-based e-business environment. Web log click stream data must be edited, filtered, and summarized before its granularity is fit for the data warehouse environment.

5-1 The Benefits of Granularity

- Reusability.
- Ability to reconcile data.
- Flexibility.
- It contains a history of activities and events across the corporation.

5-2 An Example of Granularity

Figure below shows an example of the issues of granularity. The left side shows a low level of granularity. Each activity in this case, a phone call is recorded in detail. At the end of the month, each customer has, on average, 200 records (one for each recorded

phone call throughout the month) that require about 40,000 bytes collectively. The right side of the figure shows a higher level of granularity. A high level of detail refers to a low level of granularity. A low level of detail refers to a high level of granularity. The data shown in Fig below is at a high level of granularity. It represents the summary information stet. Each record summarizes one month's activity for one customer, which requires about 200 bytes.

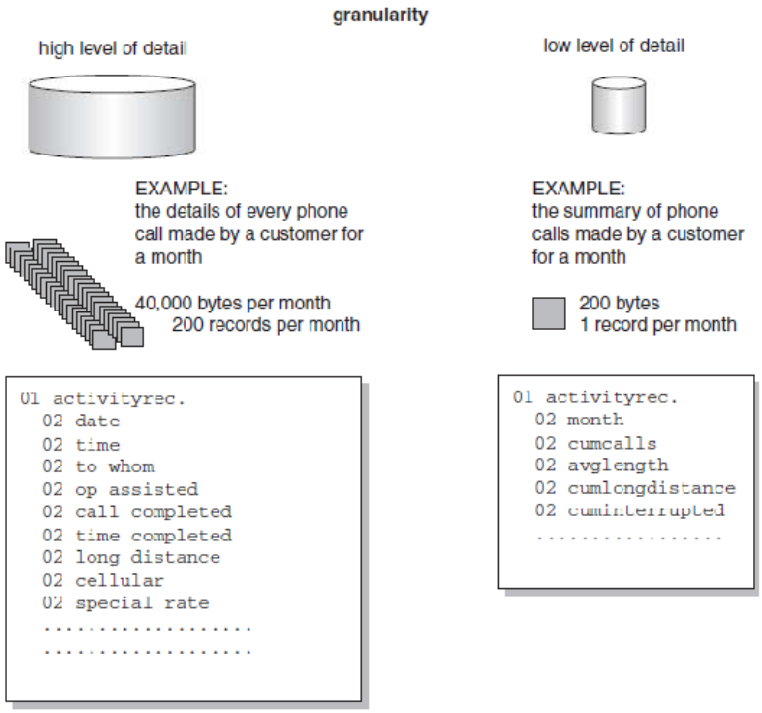


Figure 2.12 Determining the level of granularity is the most important design issue in the data warehouse environment.

Figure 4: Granularity example

6- Differences between Operational Database Systems and Data Warehouses

The major task of online operational database systems is to perform online transaction and query processing. These systems are called **online transaction processing (OLTP)** systems. They cover most of the day-to-day operations of an organization such as purchasing, inventory, banking, payroll, registration, and accounting. Data warehouse systems, on the other hand, serve users or knowledge workers in the role of data analysis and decision making. These systems are known as **online analytical processing (OLAP)** systems. The major distinguishing features of OLTP and OLAP are summarized as follows:

1. **Users and system orientation**: An OLTP system is *application-oriented* and is used for transaction and query processing by clerks, clients, and information technology professionals. An OLAP system is *subject-oriented* and is used for data analysis by knowledge workers, including managers, executives, and analysts.

2. **Data contents**: An OLTP system manages current data that, typically, are too detailed to be easily used for decision making, while OLAP system manages large amounts of historic data, provides facilities for summarization and aggregation, these features make the data easier to use for informed decision making.

3. **Database design**: An OLTP system usually adopts an entity-relationship (ER) data model and an application-oriented database design. An OLAP system typically adopts either a *star* or a *snowflake* model and a *subject-oriented* database design.

4. **View**: An OLTP system focuses mainly on the current data within an enterprise or department, without referring to historic data or data in different organizations. In contrast, an OLAP system often spans multiple versions of a database schema. OLAP systems also deal with information that originates from different organizations, integrating information from many data stores.

5. **Access patterns**: The access patterns of an OLTP system consist mainly of short, atomic transactions. Such a system requires concurrency control and recovery mechanisms. However, accesses to OLAP systems are mostly read-only operations (because most data warehouses store historic rather than up-to-date information), although many could be complex queries.

Other features that distinguish between OLTP and OLAP systems include database size, frequency of operations, and performance metrics. These are summarized in Table 1.

Table 1 Comparison of OLTP and OLAP Systems

Feature	OLTP	OLAP
Characteristic	operational processing	informational processing
Orientation	transaction	analysis
User	clerk, DBA, database professional	knowledge worker (e.g., manager, executive, analyst)
Function	day-to-day operations	long-term informational requirements decision support
DB design	ER-based, application-oriented	star/snowflake, subject-oriented
Data	current, guaranteed up-to-date	historic, accuracy maintained over time
Summarization	primitive, highly detailed	summarized, consolidated
View	detailed, flat relational	summarized, multidimensional
Unit of work	short, simple transaction	complex query
Access	read/write	mostly read
Focus	data in	information out
Operations	index/hash on primary key	lots of scans
Number of records accessed	tens	millions
Number of users	thousands	hundreds
DB size	GB to high-order GB	≥ TB
Priority	high performance, high availability	high flexibility, end-user autonomy
Metric	transaction throughput	query throughput, response time

7- A Multidimensional Data Model

Data warehouses and OLAP tools are based on a multidimensional data model. This model views data in the form of a data cube. In this section, you will learn how data cubes model n-dimensional data. Various multidimensional models are shown: star schema, snowflake schema, and fact constellation.

7-1 Data Cube: From Tables and Spreadsheets to Data Cubes

"What is a data cube?" A **data cube** allows data to be modeled and viewed in multiple dimensions. It is defined by dimensions and facts. In general terms, **dimensions** are entities with respect to which an organization wants to keep records. For example, *AllElectronics* may create a *sales* data warehouse in order to keep records of the store's sales with respect to the dimensions *time*, *item*, *branch*, and *location*. These dimensions allow the store to keep track of things like monthly sales of items and the branches and locations at which the items were sold.

Each dimension may have a table associated with it, called a **dimension table**. For example, a dimension table for *item* may contain the attributes *item name*, *brand*, and *type*. Dimension tables can be specified by users or experts, or automatically generated and adjusted based on data distributions.

A multidimensional data model is typically organized around a central theme, such as *sales*. This theme is represented by a fact table. **Facts** are numeric measures. Think of them as the quantities by which we want to analyze relationships between dimensions. Examples of facts for a sales data warehouse include *dollars sold* (sales amount in dollars) and *units sold* (number of units sold). The **fact table** contains the names of the *facts*, or measures, as well as keys to each of the related dimension tables.

We usually think of cubes as 3-D geometric structures, in data warehousing the data cube is *n*-dimensional. To gain a better understanding of data cubes and the multidimensional data model, let's start by looking at a simple 2-D data cube that is, in fact, a table or spreadsheet for sales data from *AllElectronics*. In particular, we will look at the *AllElectronics* sales data for items sold per quarter in the city of Vancouver. These data are shown in Table 2. In this 2-D representation, the sales for Vancouver are

shown with respect to the *time* dimension (organized in quarters) and the *item* dimension (organized according to the types of items sold). The fact or measure displayed is *dollars sold* (in thousands).

Table 2. 2-D View of Sales Data for *AllElectronics* According to *time* and *item*

location = "Vancouver"				
	item (type)			
time (quarter)	home entertainment	computer	phone	security
Q1	605	825	14	400
Q2	680	952	31	512
Q3	812	1023	30	501
Q4	927	1038	38	580

Note: The sales are from branches located in the city of Vancouver. The measure displayed is *dollars_sold* (in thousands).

Now, suppose that we would like to view the sales data with a third dimension. For instance, suppose we would like to view the data according to *time* and *item*, as well as *location*, for the cities Chicago, New York, Toronto, and Vancouver. These 3-D data are shown in Table 3. The 3-D data in the table are represented as a series of 2-D tables. Conceptually, we may also represent the same data in the form of a 3-D data cube, as in Figure 5.

Table 3. 3-D View of Sales Data for *AllElectronics* According to *time*, *item*, and *location*.

	location = "Chicago"				**location** = "New York"				**location** = "Toronto"				**location** = "Vancouver"			
	item				**item**				**item**				**item**			
	home				home				home				home			
time	ent.	comp.	phone	sec.	ent.	comp.	phone	sec.	ent.	comp.	phone	sec.	ent.	comp.	phone	sec.
Q1	854	882	89	623	1087	968	38	872	818	746	43	591	605	825	14	400
Q2	943	890	64	698	1130	1024	41	925	894	769	52	682	680	952	31	512
Q3	1032	924	59	789	1034	1048	45	1002	940	795	58	728	812	1023	30	501
Q4	1129	992	63	870	1142	1091	54	984	978	864	59	784	927	1038	38	580

Note: The measure displayed is *dollars_sold* (in thousands).

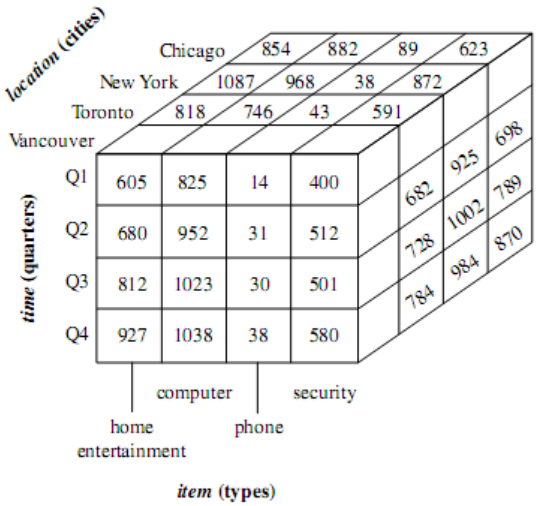


Figure 5. A 3-D data cube representation of the data in Table 3, according to *time*, *item*, and *location*.

Suppose that we would now like to view our sales data with an additional fourth dimension such as *supplier*. Viewing things in 4-D becomes tricky. However, we can think of a 4-D cube as being a series of 3-D cubes, as shown in Figure 6.

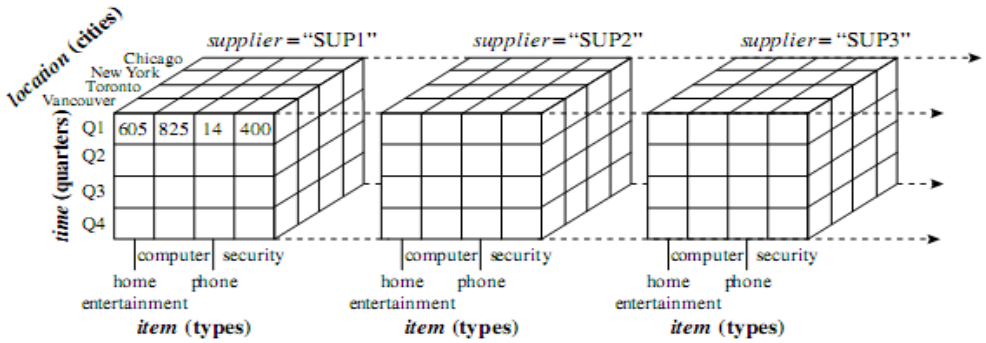


Figure 6. A 4-D data cube representation of sales data, according to *time*, *item*, *location*, and *supplier*. For improved readability, only some of the cube values are shown.

If we continue in this way, we may display any *n*-dimensional data as a series of (*n*-1) dimensional "cubes." The data cube is a metaphor for multidimensional data storage. The actual physical storage of such data may differ from its logical representation. Tables 2 and 3 show the data at different degrees of summarization. In the data warehousing research literature, a data cube like those shown in Figures 5 and 6 is often referred to as a **cuboid**. Given a set of dimensions, we can generate a cuboid for each of the possible subsets of the given dimensions. The result would form a *lattice* of cuboids, each showing the data at a different level of summarization, or **group-by**. The lattice of cuboids is then referred to as a data cube. Figure 7 shows a lattice of cuboids forming a data cube for the dimensions *time*, *item*, *location*, and *supplier*.

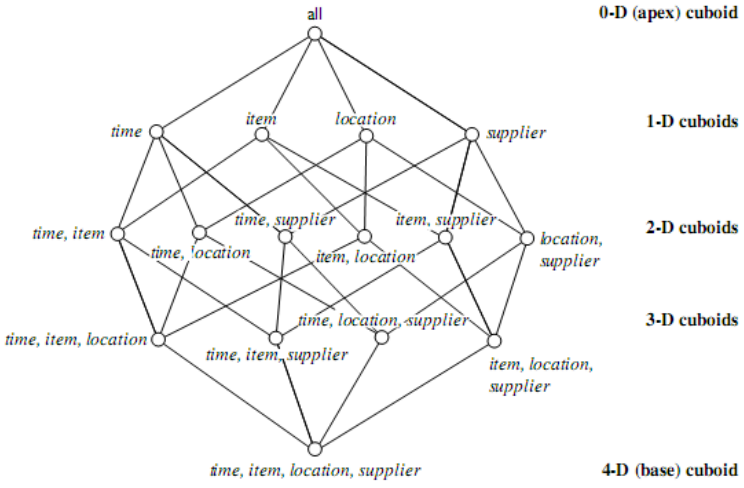


Figure 7. Lattice of cuboids, making up a 4-D data cube for *time*, *item*, *location*, and *supplier*. Each cuboid represents a different degree of summarization.

The cuboid that holds the lowest level of summarization is called the **base cuboid**. For example, the 4-D cuboid in Figure 6 is the base cuboid for the given *time*, *item*, *location*, and *supplier* dimensions. Figure 5 is a 3-D (nonbase) cuboid for *time*, *item*, and *location*, summarized for all suppliers. The 0-D cuboid, which holds the highest

level of summarization, is called the **apex cuboid**. In our example, this is the total sales, summarized over all four dimensions. The apex cuboid is typically denoted by all.

7-2 Stars, Snowflakes, and Fact Constellations: Schemas for Multidimensional Data Models

The entity-relationship data model is commonly used in the design of relational databases, where a database schema consists of a set of entities and the relationships between them. Such a data model is appropriate for online transaction processing. A data warehouse, however, requires a concise, subject-oriented schema that facilitates online data analysis. The most popular data model for a data warehouse is a **multidimensional model**, which can exist in the form of a **star schema**, a **snowflake schema**, or a **fact constellation schema**.

1. **Star schema:** The most common modeling paradigm is the star schema, in which the data warehouse contains (1) a large central table (**fact table**) containing the bulk of the data, with no redundancy, and (2) a set of smaller attendant tables (**dimension tables**), one for each dimension. The schema graph resembles a starburst, with the dimension tables displayed in a radial pattern around the central fact table.

Example

A star schema for *AllElectronics* sales is shown in Figure 8. Sales are considered along four dimensions: *time*, *item*, *branch*, and *location*. The schema contains a central fact table for *sales* that contains keys to each of the four dimensions, along with two measures: *dollars sold* and *units sold*. Notice that in the star schema, each dimension is represented by only one table, and each table contains a set of attributes. For example, the *location* dimension table contains the attribute set *location key*, *street*, *city*, *province*

or state, country. This constraint may introduce some redundancy. For example, "Urbana" and "Chicago" are both cities in the state of Illinois, USA. Entries for such cities in the *location* dimension table will create redundancy among the attributes *province or state* and *country*; that is ..., Urbana, IL, USA and ..., Chicago, IL, USA. Moreover, the attributes within a dimension table may form a hierarchy.

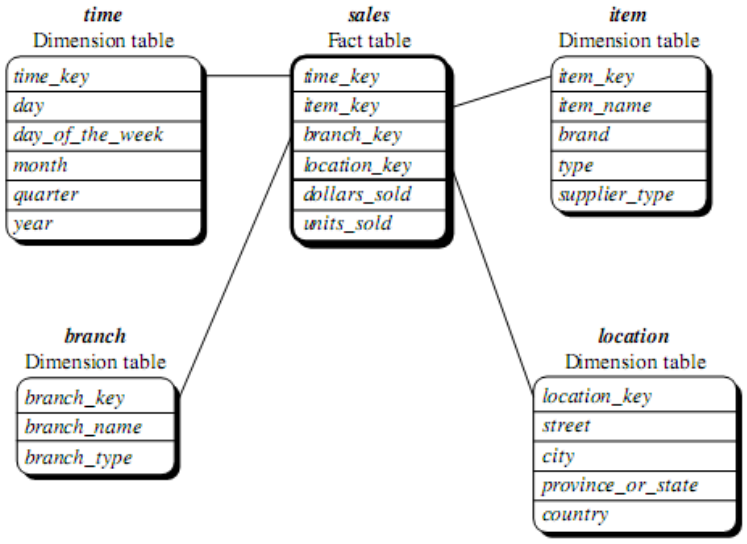


Figure 8. Star schema of *sales* data warehouse.

2. **Snowflake schema:** The snowflake schema is a variant of the star schema model, where some dimension tables are *normalized*, thereby further splitting the data into additional tables. The major difference between the snowflake and star schema models is that the dimension tables of the snowflake model may be kept in normalized form to reduce redundancies. Such a table is easy to maintain and saves storage space. However, this space savings is negligible in comparison to the typical magnitude of the fact table. Furthermore, the snowflake structure can reduce the

effectiveness of browsing, since more joins will be needed to execute a query. Consequently, the system performance may be adversely impacted.

Example

A snowflake schema for *AllElectronics* sales is given in Figure 9. Here, the *sales* fact table is identical to that of the star schema in Figure 8. The main difference between the two schemas is in the definition of dimension tables. The single dimension table for *item* in the star schema is normalized in the snowflake schema, resulting in new *item* and *supplier* tables. For example, the *item* dimension table now contains the attributes *item key*, *item name*, *brand*, *type*, and *supplier key*, where *supplier key* is linked to the *supplier* dimension table, containing *supplier key* and *supplier type* information.

Similarly, the single dimension table for *location* in the star schema can be normalized into two new tables: *location* and *city*. The *city key* in the new *location* table links to the *city* dimension. Notice that, when desirable, further normalization can be performed on *province or state* and *country* in the snowflake schema.

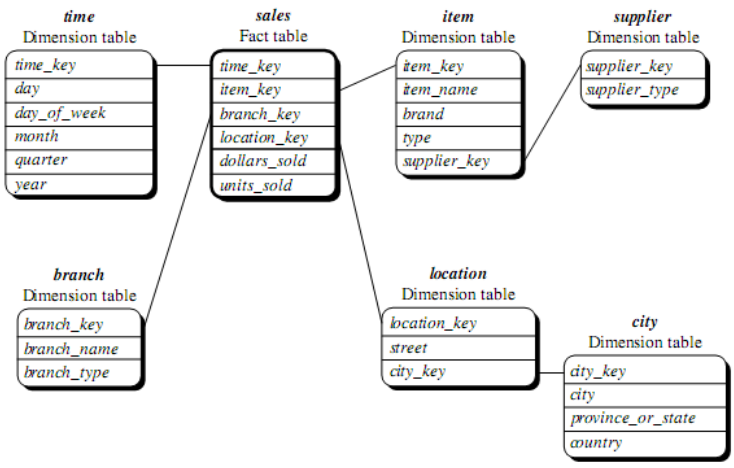


Figure 9. Snowflake schema of a *sales* data warehouse.

3. **Fact constellation:** Sophisticated applications may require multiple fact tables to *share* dimension tables. This kind of schema can be viewed as a collection of stars, and hence is called a **galaxy schema** or a **fact constellation**.

Example

A fact constellation schema is shown in Figure 10. This schema specifies two fact tables, *sales* and *shipping*. The *sales* table definition is identical to that of the star schema (Figure 8). The *shipping* table has five dimensions, or keys—*item key*, *time key*, *shipper key*, *from_location*, and *to_location*—and two measures—*dollars cost* and *units shipped*. A fact constellation schema allows dimension tables to be shared between fact tables. For example, the dimensions tables for *time*, *item*, and *location* are shared between the *sales* and *shipping* fact tables.

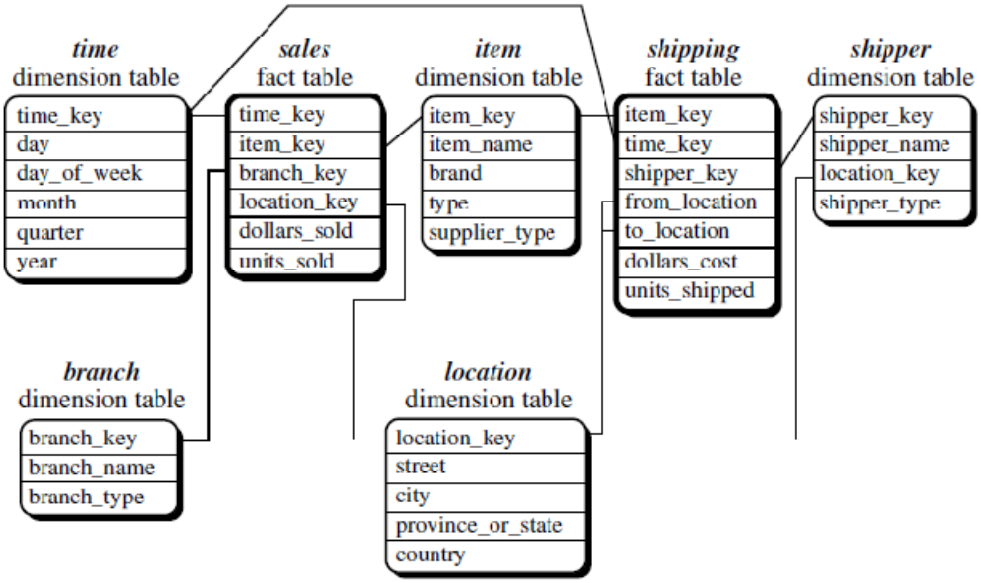


Figure 10: A fact constellation schema

7-3 OLAP Operations on a multidimensional data model

"How are concept hierarchies useful in OLAP?" In the multidimensional model, data are organized into multiple dimensions, and each dimension contains multiple levels of abstraction defined by concept hierarchies. This organization provides users with the flexibility to view data from different perspectives. A number of OLAP data cube operations exist to materialize these different views, allowing analysis of the data. Hence, OLAP provides a user-friendly environment for interactive data analysis.

Example

Let's look at some typical OLAP operations for multidimensional data. Each of the following operations described is illustrated in Figure 13. At the center of the figure is a data cube for *AllElectronics* sales. The cube contains the dimensions *location*, *time*, and *item*, where *location* is aggregated with respect to city values, *time* is aggregated with respect to quarters, and *item* is aggregated with respect to item types. To aid in our explanation, we refer to this cube as the central cube. The data examined are for the cities Chicago, New York, Toronto, and Vancouver.

Roll-up: The roll-up operation (also called the *drill-up* operation by some vendors) performs aggregation on a data cube, either by *climbing up a concept hierarchy* for a dimension or by *dimension reduction*. Figure 13 shows the result of a roll-up operation performed on the central cube by climbing up the concept hierarchy for *location* given in Figure 10. This hierarchy was defined as the total order "*street<city<province or state<country*." The roll-up operation shown aggregates the data by ascending the *location* hierarchy from the level of *city* to the level of *country*. In other words, rather than grouping the data by city, the resulting cube groups the data by country.

When roll-up is performed by dimension reduction, one or more dimensions are removed from the given cube. For example, consider a sales data cube containing only the *location* and *time* dimensions. Roll-up may be performed by removing, say, the *time* dimension, resulting in an aggregation of the total sales by location, rather than by location and by time.

Drill-down: Drill-down is the reverse of roll-up. It navigates from less detailed data to more detailed data. Drill-down can be realized by either *stepping down a concept hierarchy* for a dimension or *introducing additional dimensions*. Figure 13 shows the result of a drill-down operation performed on the central cube by stepping down a concept hierarchy for *time* defined as "*day*<*month*<*quarter*<*year*." Drill-down occurs by descending the *time* hierarchy from the level of *quarter* to the more detailed level of *month*. The resulting data cube details the total sales per month rather than summarizing them by quarter. Because a drill-down adds more detail to the given data, it can also be performed by adding new dimensions to a cube.

Slice and dice: The *slice* operation performs a selection on one dimension of the given cube, resulting in a subcube. Figure 13 shows a slice operation where the sales data are selected from the central cube for the dimension *time* using the criterion *time*="Q1." The *dice* operation defines a subcube by performing a selection on two or more dimensions. Figure 13 shows a dice operation on the central cube based on the following selection criteria that involve three dimensions: (*location*="Toronto" or "Vancouver") and (*time*="Q1" or "Q2") and (item="home entertainment" or "computer").

Pivot (rotate): *Pivot* (also called *rotate*) is a visualization operation that rotates the data axes in view to provide an alternative data presentation. Figure 11 shows a pivot operation where the *item* and *location* axes in a 2-D slice are rotated.

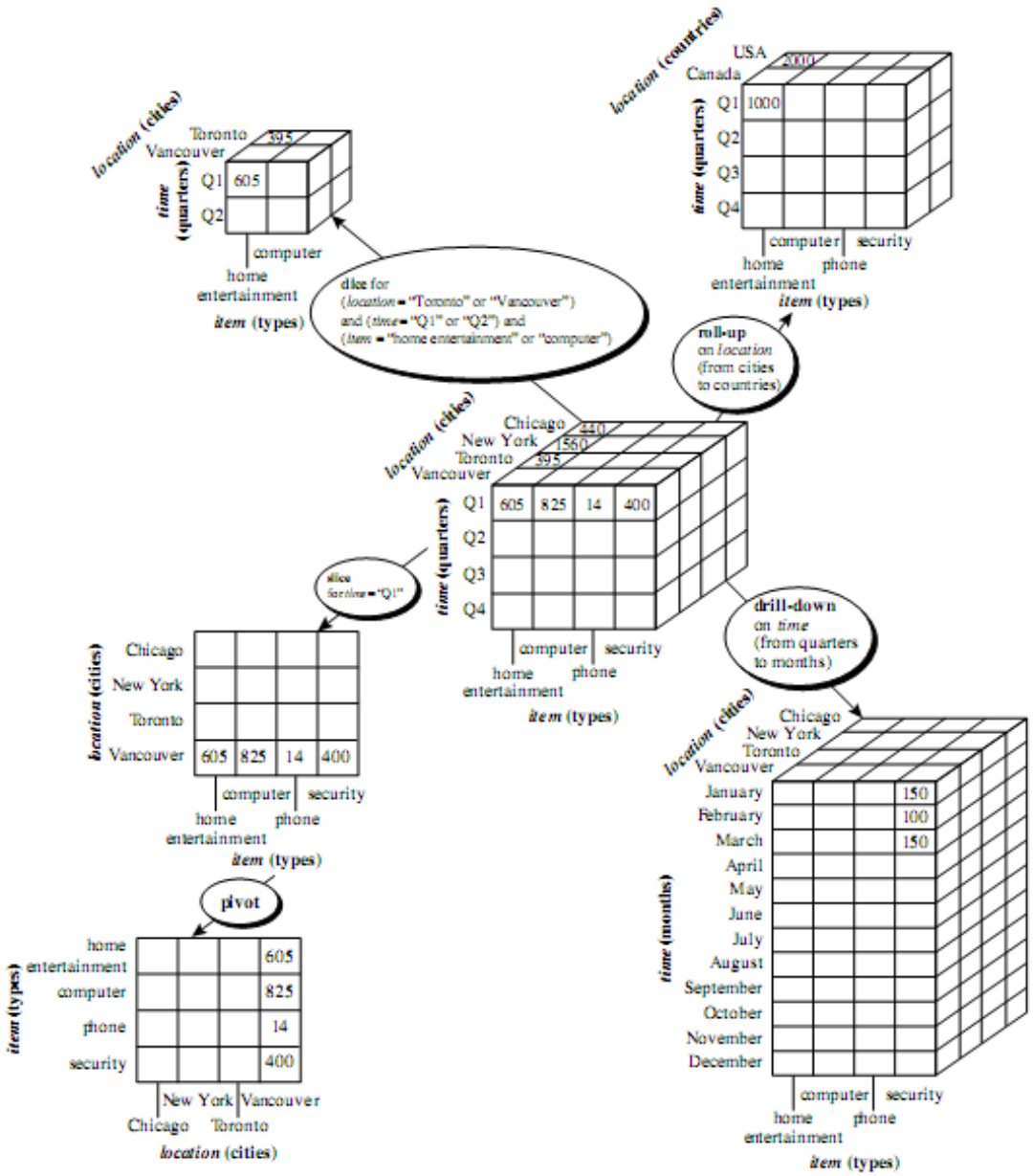


Figure 11. Examples of typical OLAP operations on multidimensional data.

8- Data Warehouse Architecture

Basically, there are two thought of schools concerning the architecture for the data warehouse, the Top-Down approach, and Bottom-Up approach for Inmon and Kimball, respectively. Data warehouse building is based upon number of components.

8-1 Data Warehouse Components

Building data warehouse involves grouping a number of essential components like building a school or a hospital can be seen as building blocks consists of class rooms, offices, etc. but with a specific arrangements that makes the best interest. Same way, data warehouses consist of a number of blocks and the structure of these components called architecture. Building blocks consist of source data, data staging, data storage, and information delivery.

1. Source Data Component

- **Production Data:** This sort of data comes from different operational systems and based on the information needed by data warehouse requirements, choosing segments of data from these systems. These data reside on multiplatform hardware systems with different formats supported by different operating systems.
- **Internal Data:** Inside organization users may keep their private spreadsheets, customer profiles, and documents. This data may be a part of the data warehouse.
- **Archived Data:** Transactional databases are designated to do current businesses; some of the data are archived in files. Many different archiving methods do exist. The first, storing data on archival databases that may be still in use on-line. Whilst, the second storing older data to flat files on disk storage. And finally, archiving data in cartridges kept off-line (Long period data).
- **External Data:** Executives depend mostly on information from external sources such as statistics relating to their business by external agencies may be used.

2- Data Staging Component (DSA)

After data extraction from disparate operational systems, whether it was internally or externally, a conversion takes place before loading data into the data warehouse. By conversion means data must first be cleansed, reconciled, derived, deduplicated, standardized, and make data in a format such that to be ready for further analysis and queries. Three major processes or functions need to be done in this area, which are data extraction, data transformation, and data loading.

- **Data Extraction:** This function deals with numerous amounts of data from different sources. Dedicated techniques may need to be adopted depending on the source of data. Data may be in relational databases, legacy systems, flat files, or also internally in spreadsheets. The complexity of technique depends on the complexity of the source data to be extracted.
- **Data Transformation:** Data transformation is an important process, such that converting data from its prior systems to new state depending on DW system requirements. Again, as mentioned in the previous paragraph the extraction complexity depends on data source systems, data conversion will even raise greater challenges. Transformation involves a number of tasks need to take place like cleansing data, standardization, and combining pieces of related data from a single source or different systems. When data transformation done, a collection of integrated data that is cleaned, standardized, and summarized are ready to load into the data warehouse.
- **Data Loading:** The last process or function in Data Staging Area (DSA) is data loading. After data warehouse design and construction, then go live for the first time, initial loading is performed to load data into DW storage, which moves a large amount of data. After DW functioning, extractions of data

changes process continue to work and transformation of data revision incrementally keeps going in a regular basis.

Data staging area is the central repository for ETL (Extraction, Transformation, and Loading) processing. Figure 12 exhibits DSA within data warehouse.

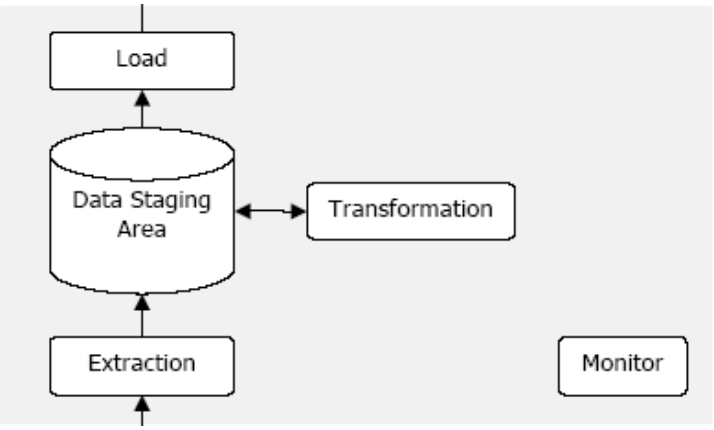


Figure 12. ETL Process within DSA.

3- Data Storage Component

Data resides in the data warehouse within a separate repository unlike operational systems for essential reasons. As known, data in transactional systems support day-to-day transaction processing. As an alternative, data in a warehouse repository are snapshots at a specified time (history data) and the most important thing is that operational systems structured in a highly normalized format for fast retrieval of data and efficient processing while the data warehouse repository must be structured properly for further analysis. Analysts need the data in storage area to be stable. In contrast, to the operational systems how and when transactions occurring stay unknown. Also, data aggregated, summarized, and derived in subject purpose groups known as data marts for a better information delivery.

4- Information Delivery Component

Data delivered from the data warehouse to a wide range of users. For example, novice users who in this case need premade-up report and predefined queries. Casual users who need information once in awhile, this type also needs prepackaged information. Business analysts who perform complex analysis. Likewise, powerful users who navigate through the data warehouse drill down. There are different methods of feeding information to the end-users as listed below:

- Ad hoc reports: predefined reports intended to novice users.
- Executive Information System: information is feed to executives.
- Data mining: help in discovering new key trends and pattern of data. Also, known as Knowledge Discovery in Database (KDD), and finally
- Complex queries, Multidimensional (MD), and statistical analysis: this type is meant to powerful users for business analysis.

5- Metadata Component

The most important aspect of the data warehouse is metadata also known as data dictionary. Many definitions had been conducted concerning metadata some of them listed below:

- Data about the data in the data warehouse.
- Table of contents for the data.
- Catalog for the data
- Data warehouse atlas, and
- The nerve center.

Further discussion concerning metadata will be introduced in section 8. Figure 13 depicts data warehouse components.

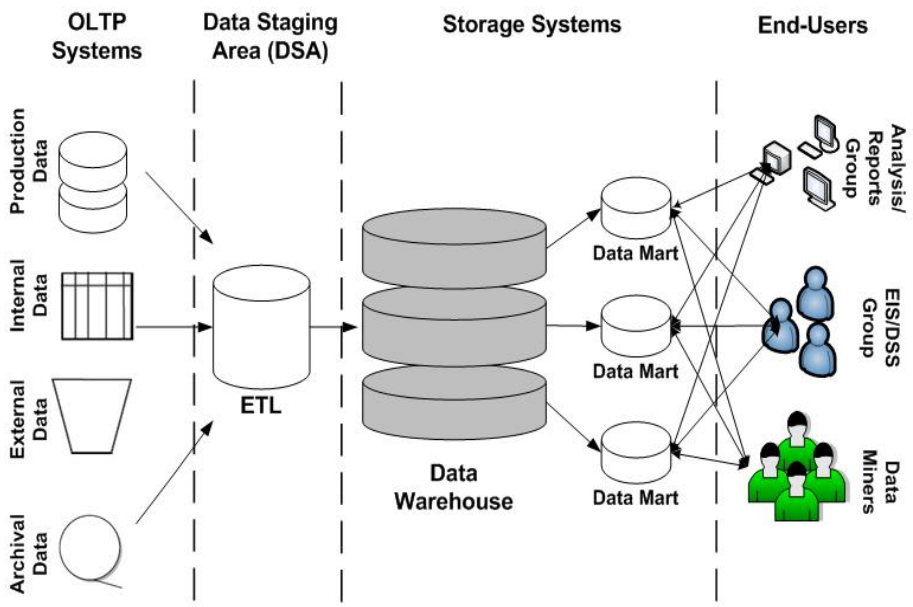


Figure 13. Data Warehouse Components.

8- Meta Data

For Example: In order to store data, over the years, many application designers in each branch have made their individual decisions as to how an application and database should be built. So source systems will be different in naming conventions, variable measurements, encoding structures, and physical attributes of data. Consider a bank that has got several branches in several countries, has millions of customers and the lines of business of the enterprise are savings, and loans. The following example explains how the data is integrated from source systems to target systems.

Example of Source Data

System name	*Attribute name*	*Column name*	*Datatype*	*Values*
Source System 1	Customer Application Date	CUSTOMER_ APPLICATION_DATE	NUMERIC (8, 0)	11012005
Source System 2	Customer Application Date	CUST_APPLICATION_ DATE	DATE	11012005
Source System 3	Application Date	APPLICATION_DATE	DATE	01NOV2005

In the aforementioned example, attribute name, column name, datatype and values are entirely different from one source system to another. This inconsistency in data can be avoided by integrating the data into a data warehouse with good standards

Example of Target Data (Data Warehouse)

Target system	*Attribute name*	*Column name*	*Datatype*	*Values*
Record #1	Customer Application Date	CUSTOMER_ APPLICATION_DATE	DATE	01112005
Record #2	Customer Application Date	CUSTOMER_ APPLICATION_DATE	DATE	01112005
Record #3	Customer Application Date	CUSTOMER_ APPLICATION_DATE	DATE	01112005

9- Data Mart

A data warehouse is a cohesive data model that defines the central data repository for an organization. A data mart is a data repository for a specific user group. It contains summarized data that the user group can easily understand, process, and apply. A data mart cannot stand alone; it requires a data warehouse. Because each data warehousing effort is unique, your company's data warehousing environment may differ slightly from what we are about to introduce. Each data mart is a collection of tables organized according to the particular requirements of a user or group of users. Retrieving a collection of different kinds of data from a "normalized" warehouse can be complex and time-consuming. Hence the need to rearrange the data so they can be retrieved more easily. The notion of a "mart" suggests that it is organized for the

ultimate consumers with the potato chips, and video tapes all next to each other. This organization does not have to follow any particular inherent rules or structures. Indeed, it may not even make sense. And however the marts are organized initially, the requirements are almost certain to change once the user has seen the implications of the request. This means that the creation of data marts requires:

- Understanding of the business involved.
- Responsiveness to the user's stated objectives.
- Sufficient facility with database modeling and design to produce new tables quickly.
- Tools to convert models into data marts quickly.

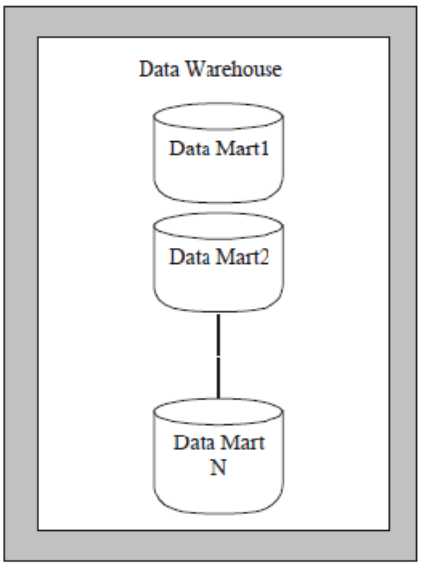


Figure 14: data warehouse and data mart

10- Steps for the Design and Construction of Data Warehouse (A business analysis framework)

Four different views regarding a data warehouse design must be considered:

- T**op-down view** allows the selection of the relevant information necessary for the data warehouse based on current and future needs
- **Data source view** exposes the information being captured, stored, and managed by operational systems (E/R models, CASE, etc). This information may be documented at various levels of detail.
- **Data warehouse view**
  - Includes fact tables and dimension tables, pre-calculated totals and counts.
  - Source information, date, and time provide historical context.
- B**usiness query view** is the data perspective in the data warehouse from the end-user's viewpoint.

10-1 Data Warehouse Design Process

- **Top-Down approach:** Starts with overall design and planning (technology is mature; business problems that must be solved are clear and well understood). Derived by Bill Inmon (Father of data warehouse).
- **Bottom-Up approach:** Starts with experiments and prototypes (rapid development, and less expense). Derived by Ralph Kimball (Co-founder).

From software engineering point of view the design and construction of a data warehouse may consist of the following steps: *planning, requirement study, problem analysis, warehouse design, data integration and testing, and finally deployment of the data warehouse.*

Methodologies used to construct large systems (Data Warehouse):

- **Waterfall:** structured and systematic analysis at each step.
- **Spiral:** rapid generation of increasingly functional systems, short turnaround time, quick turnaround.

Typical data warehouse design process consist the following steps:

- Choose a ***business process*** to model, e.g., orders, invoices, shipments, sales, etc.
- Choose the ***grain* (*atomic level of data*)** of the business process.
- Choose the ***dimensions*** that will apply to each fact table record.
- Choose the ***measure*** that will populate each fact table record.

11- Three-Tier Data Warehouse Architecture

Data warehouses often adopt three-tier architecture, as presented in Figure 15.

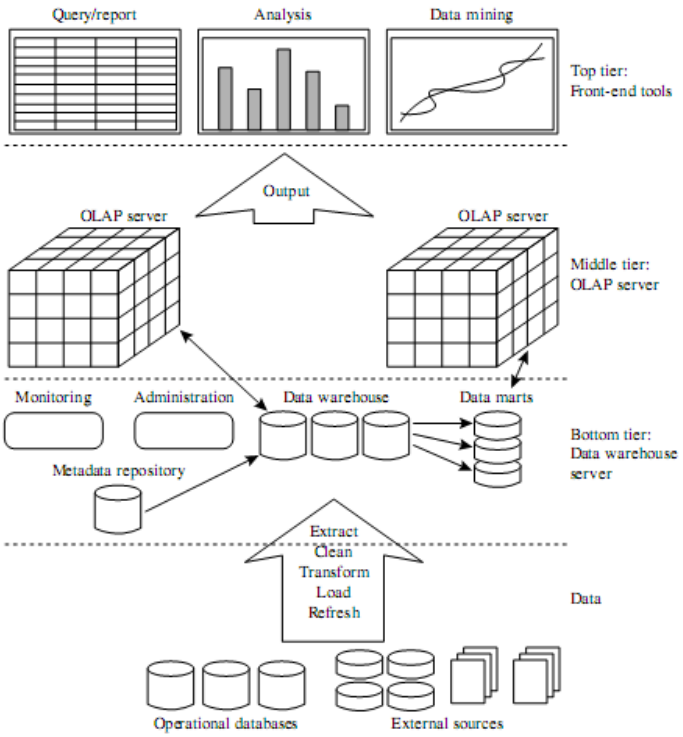


Figure 15. Three-Tier Data Warehouse Architecture.

- The bottom tier is a **warehouse database server** that is almost always a relational database system. Back-end tools and utilities are used to feed data into the bottom tier from operational databases or other external sources (e.g., customer profile information provided by external consultants). These tools and utilities perform data extraction, cleaning, and transformation (e.g., to merge similar data from different sources into a unified format), as well as load and refresh functions to update the data warehouse. This tier also contains a metadata repository, which stores information about data warehouse and its contents.

- The middle tier is an **OLAP server** that is typically implemented using either (1) a **relational OLAP (ROLAP)** model; or (2) a **multidimensional OLAP (MOLAP)** model.

- The top tier is a **front-end client layer**, which contains query and reporting tools, analysis tools, and/or data mining tools (e.g., trend analysis, prediction, and so on).

12- Data Homogeneity/Heterogeneity

At first glance, it may appear that the data found in the data warehouse is homogeneous in the sense that all of the types of records are the same. In truth, data in the data warehouse is very heterogeneous. The data found in the data warehouse is divided into major subdivisions called subject areas. Figure 16 shows that a data warehouse has subject areas of product, customer, vendor, and transaction. The first division of data inside a data warehouse is along the lines of the major subjects of the corporation. But with each subject area there are further subdivisions. Data within a subject area is divided into tables. Figure 17 shows this division of data into tables for the subject area product.

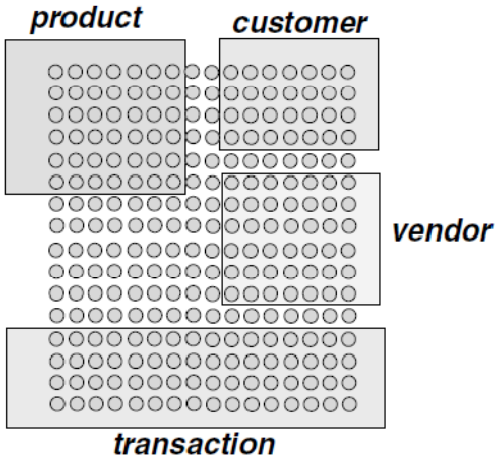


Figure 16: The data in the different parts of the data warehouse are grouped by subject area.

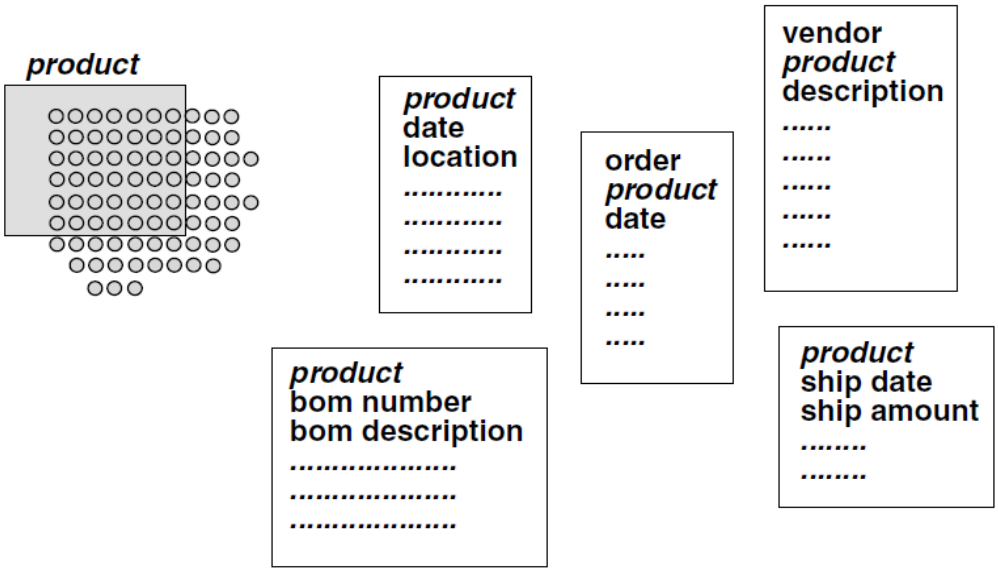


Figure 17: Within the product subject area there are different types of tables, but each table has a common product identifier as part of the key.

13-Types of Distributed Data Warehouses

The three types of distributed data warehouses are as follows:

- Business is distributed geographically or over multiple, differing product lines. In this case, there is what can be called a local data warehouse and a global data warehouse. The local data warehouse represents data and processing at a remote site, and the global data warehouse represents that part of the business that is integrated across the business.
- The data warehouse environment will hold a lot of data, and the volume of data will be distributed over multiple processors. Logically there is a single data warehouse, but physically there are many data warehouses that are all tightly related but reside on separate processors. This configuration can be called the technologically distributed data warehouse.
- The data warehouse environment grows up in an uncoordinated manner first one data warehouse appears, then another. The lack of coordination of the growth of the different data warehouses is usually a result of political and organizational differences. This case can be called the independently evolving distributed data warehouse.

13-1 Local and Global Data Warehouses

When a corporation is spread around the world, information is needed both locally and globally. The global needs for corporate information are met by a central data warehouse where information is gathered. But there is also a need for a separate data warehouse at each local organization that is, in each country. In this case, a distributed data warehouse is needed. Data will exist both centrally and in a distributed manner.

A second case for a local/global distributed data warehouse occurs when a large corporation has many lines of business.

In some cases part of the data warehouse exists centrally (i.e., globally), and other parts of the data warehouse exist in a distributed manner (i.e., locally).

13-2 The Local Data Warehouse

A form of data warehouse, known as a local data warehouse, contains data that is of interest only to the local level. There might be a local data warehouse for Brazil, one for France, and one for Hong Kong. Or there might be a local data warehouse for car parts, motorcycles, and heavy trucks. Each local data warehouse has its own technology, its own data, its own processor, and so forth.

13-3 The Global Data Warehouse

Global data warehouse has as its scope the corporation or the enterprise, while each of the local data warehouses within the corporation has as its scope the local site that it serves. For example, the data warehouse in Brazil does not coordinate or share data with the data warehouse in France, but the local data warehouse in Brazil does share data with the corporate headquarters data warehouse in Chicago. Or the local data warehouse for car parts does not share data with the local data warehouse for motorcycles, but it does share data with the corporate data warehouse in Detroit. The scope of the global data warehouse is the business that is integrated across the corporation. In some cases, there is considerable corporate integrated data; in other cases, there is very little. The global data warehouse contains historical data, as do the local data warehouses.

14- External/Unstructured Data and the Data Warehouse

Most organizations build their first data warehouse efforts on data whose source is existing systems (i.e., on data internal to the corporation). In almost every case, this data

can be termed internal, structured data. The data comes internally from the corporation and has been already shaped into a regularly occurring format. A whole host of other data is of legitimate use to a corporation that is not generated from the corporation's own systems. This class of data is called external data and usually enters the corporation in an unstructured, unpredictable format. Figure 18 shows external and unstructured data entering the data warehouse. The data warehouse is the ideal place to store external and unstructured data.

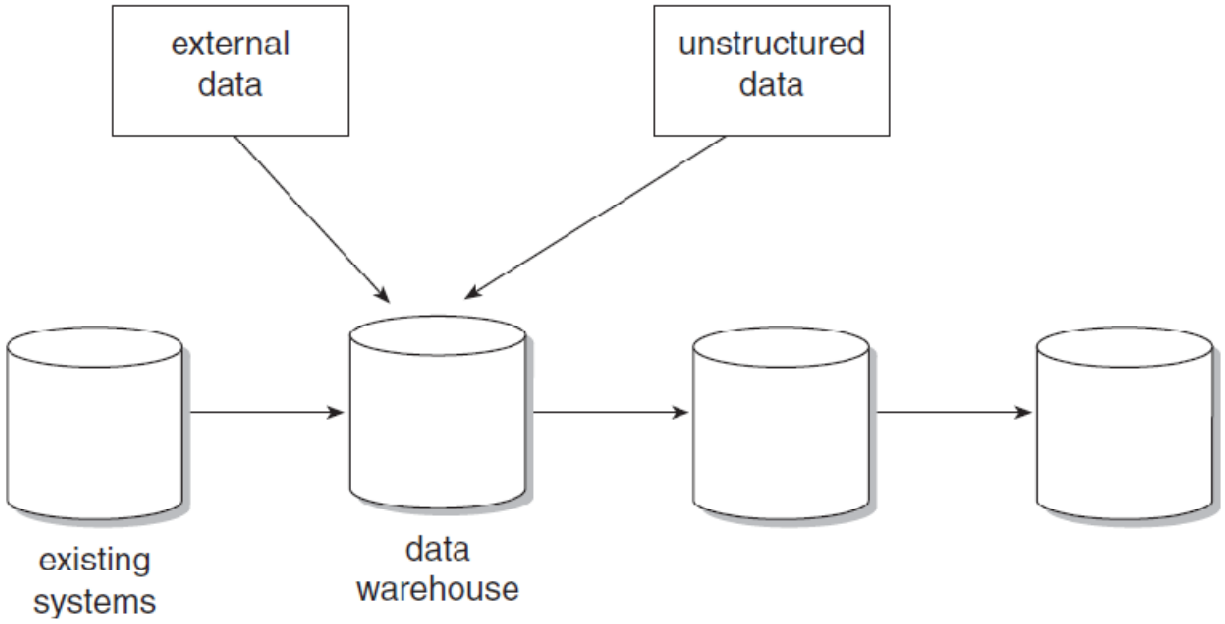


Figure 18: External and unstructured data both belong in the data warehouse.

Several issues relate to the use and storage of external and unstructured data in the data warehouse.

- One problem of unstructured data is the frequency of availability. Unlike internally appearing data, there is no real fixed pattern of appearance for external data.

- The second problem with external data is that it is totally undisciplined. To be useful, and for placement in the warehouse, a certain amount of reformatting of external data is needed to transform it into an internally acceptable and usable form.
- In some cases, the level of granularity of the external data will not match that of the internal systems of the corporation. For example, suppose a corporation has individual household information.

14-1 Storing External/Unstructured Data

External data and unstructured data can actually be stored in the data warehouse if it is convenient and cost-effective to do so. But in many cases it will not be possible or economical to store all external data in the data warehouse. Instead, an entry is made in the meta data of the warehouse describing where the actual body of external data can be found. The external data is then stored elsewhere, where it is convenient, as shown in Figure 19. External data may be stored in a filing cabinet, on fiche, on magnetic tape, and so on. However it is done, storing external data and unstructured data requires considerable resources. By associating external data and unstructured data with a data warehouse, the external data and the unstructured data become available for all parts of the organization, such as finance, marketing, accounting, sales, engineering, and so forth.

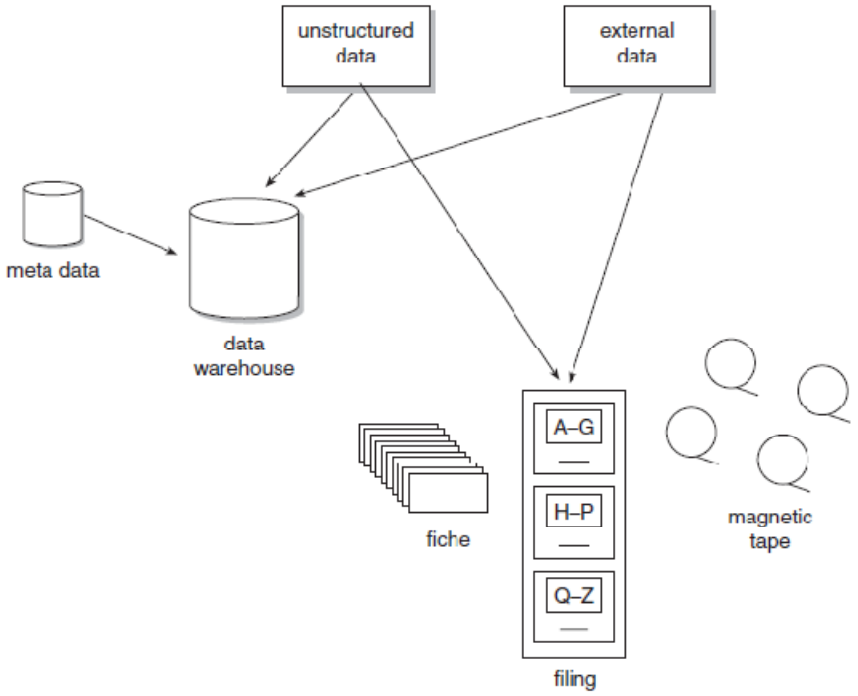


Figure 19: In every case, external/unstructured data is registered with meta data, but the actual data may or may not be stored in the data warehouse based on the size of the data and the probability of access.

15- The Data Warehouse and the Web

One of the most widely discussed technologies is the Internet and its associated environment-the World Wide Web. Embraced by Wall Street as the basis for the new economy, Web technology enjoys wide popular support among business people and technicians alike.

The Web environment interacts with corporate systems in two basic ways. One interaction occurs when the Web environment creates a transaction that needs to be executed an order from a customer, for example. The transaction is formatted and shipped to corporate systems, where it is processed just like any other order. In this

regard, the Web is merely another source for transactions entering the business. But the Web interacts with corporate systems another way as well through the collection of Web activity in a log.

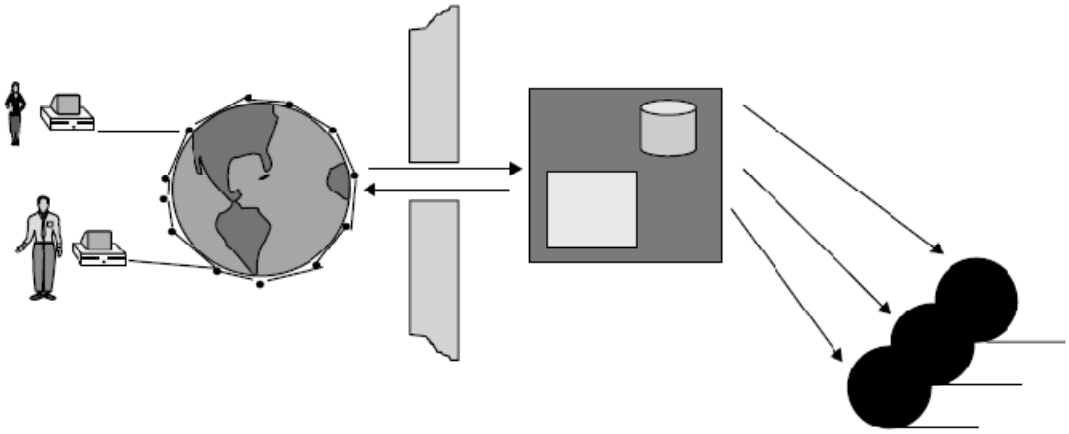


Figure 20: The activity of the Web environment is spun off into Web logs in records called clickstream records.

Figure 20 shows the capture of Web activity and the placement of that activity in a log.

- The Web log contains what is typically called clickstream data. Each time the Internet user clicks to move to a different location, a clickstream record is created.
- As the user looks at different corporate products, a record of what the user has looked at, what the user has purchased, and what the user has thought about purchasing is compiled. Equally important, what the Internet user has not looked at and has not purchased can be determined. In a word, the clickstream data is the key to understanding the stream of consciousness of the Internet user.

Figure 21 shows that Web log clickstream data is passed through software that is called a Granularity Manager before entry into the data warehouse environment.

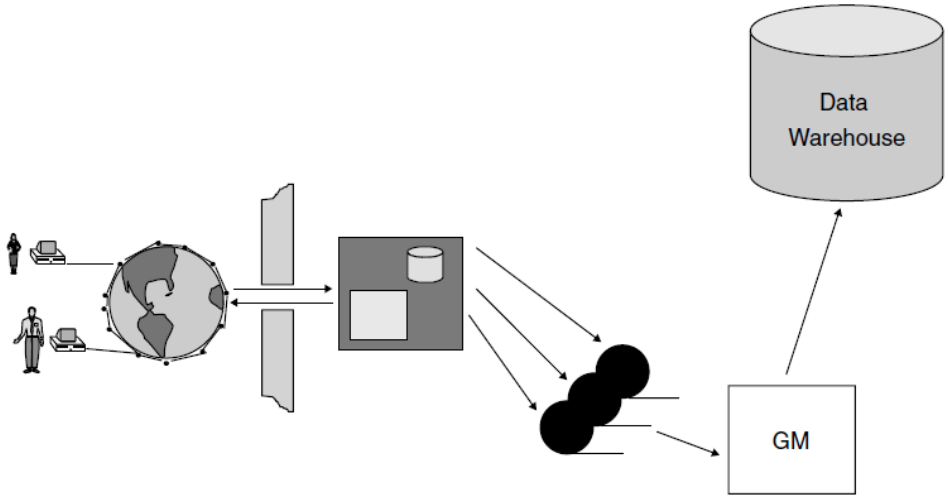


Figure 21: Data passes through the Granularity Manager before entering the data warehouse.

A lot of processing occurs in the Granularity Manager, which reads clickstream data and does the following:

- Edits out extraneous data
- Creates a single record out of multiple, related clickstream log
- Edits out incorrect data
- Converts data that is unique to the Web environment, especially key data that needs to be used in the integration with other corporate data
- Summarizes data
- Aggregates data

In summary, the process of moving data from the Web into the data warehouse involves these steps:

- Web data is collected into a log.
- The log data is processed by passing through a Granularity Manager.

- The Granularity Manager then passes the refined data into the data warehouse.

15-1 Data Warehouse into the Corporate Operational Data Store (ODS),

Figure 22 shows that data passes out of the data warehouse into the corporate operational data store (ODS), where it is then available for direct access from the Web. There are some very good reasons for this positioning.

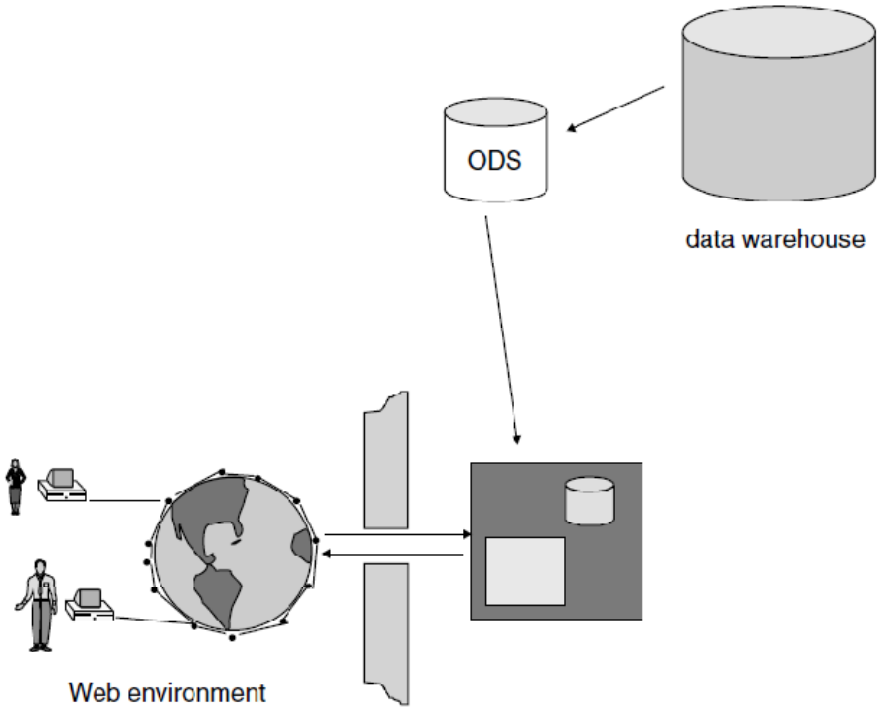


Figure 22: Data is passed to the ODS before it goes to the Web.

The ODS: is a hybrid structure that has some aspects of a data warehouse and other aspects of an operational system. The ODS contains integrated data and can support DSS processing. But the ODS can also support high-performance transaction processing. It is this last characteristic of the ODS that makes it so valuable to the Web.

At first glance it may seem that there is a lot of redundant data between the data warehouse and the ODS. After all, the ODS is fed from the data warehouse. But in truth there is very little overlap of data between the data warehouse and the ODS.

- The data warehouse contains detailed transaction data, while the ODS contains what can be termed “profile” data.
- The data warehouse contains all sorts of transaction data about past interactions between the customer and the business.
- The data warehouse maintains a detailed log, by customer, of the transactional interactions the customer has had with the business, regardless of the source of the interaction.

1-Data Mining philosophy

1-1 What Motivated Data Mining? Why Is It Important?

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, for production control and science exploration. Data mining can be viewed as a result of the natural evolution of information technology. Data warehouse technology includes data cleaning, data integration, and online 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 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.

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.

1-3 Alternative name of DM

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 essential step in the process of knowledge discovery. Knowledge discovery as a process is depicted in Figure 1.3 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)

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)

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;)

7. **Knowledge presentation** (where visualization and knowledge representation techniques are used to present the mined knowledge to the user)

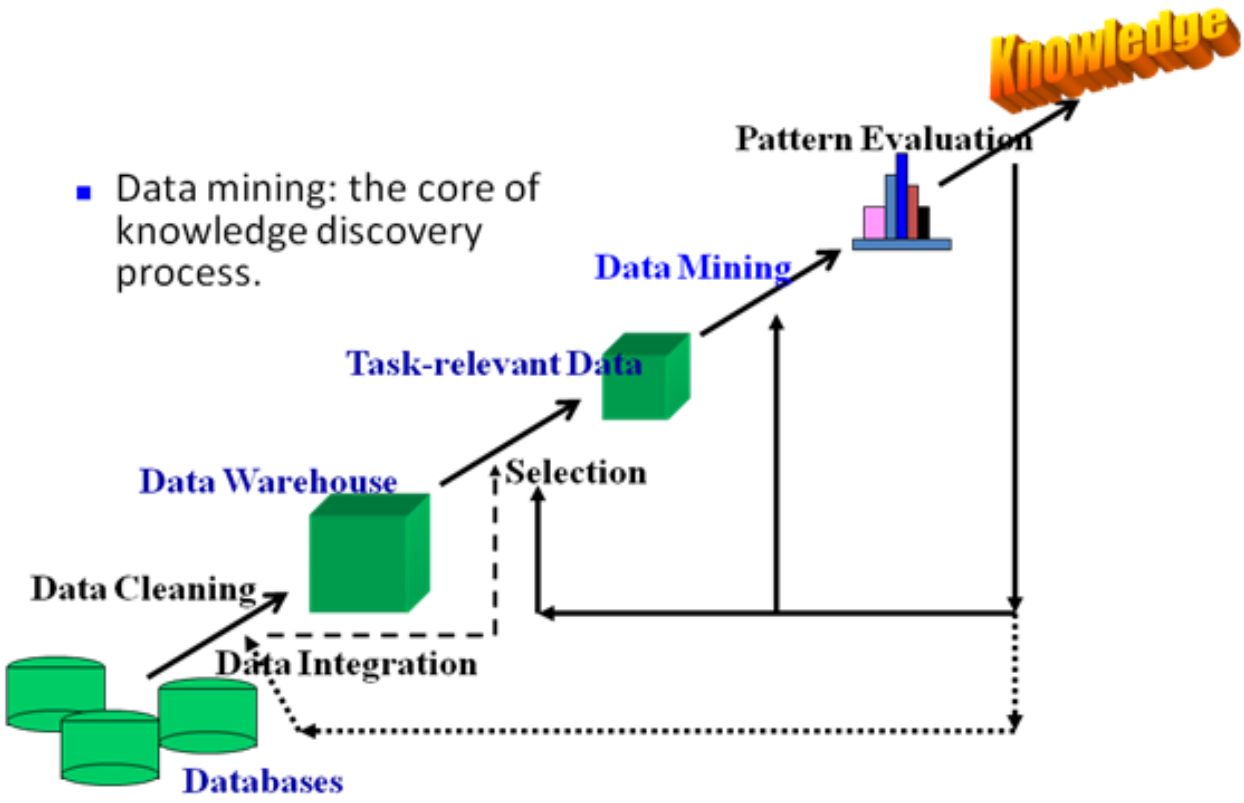


Figure 1.3. Data mining as a step in the process of knowledge discovery.

2- Architecture of data mining system

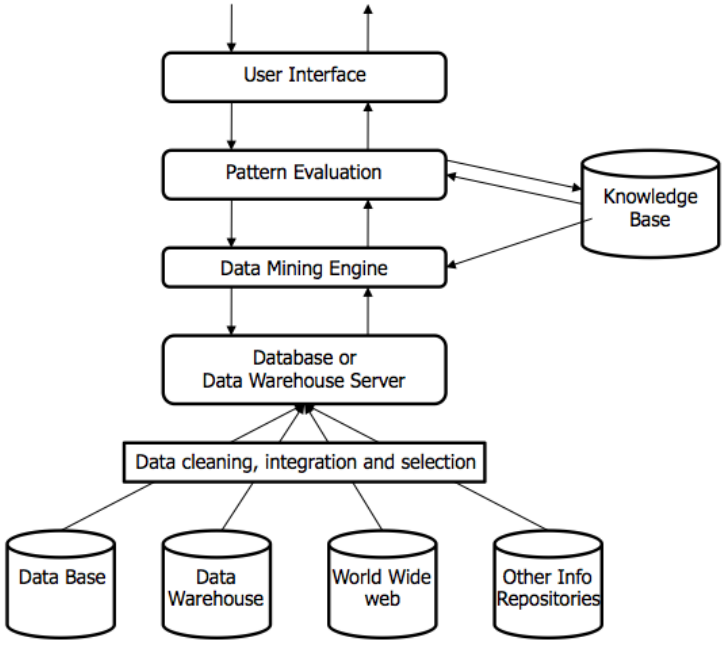


Fig 1.4 Architecture of data mining system

1- Knowledge base: This is the information of domain we are mining like concept hierarchies, to organize attributes onto various levels of abstraction, Also contains user beliefs, which can be used to access interestingness of pattern or thresholds

2- Data mining engine: Performs functionalities like characterization, association, classification, prediction etc.

3- Pattern evaluation module: Tests for interestingness of a pattern

4- User interface: Communicates between users and data mining system. Visualizes results or perform exploration on data and schemas.

3- Data Mining Applications

1- in Sales/Marketing

Data mining enables businesses to understand the hidden patterns inside historical purchasing transaction data, thus helping in planning and launching new marketing campaigns in prompt and cost effective way.

2- in Banking / Finance

Several data mining techniques, e.g., distributed data mining have been researched, modeled and developed to help credit card fraud detection.

3- in Health Care and Insurance

4- in Transportation

5-in Medicine

4- Advantages of Data Mining

1. Predict future trends, customer purchase habits
2. Help with decision making
3. Improve company revenue and lower costs
4. Market basket analysis
5. Fraud detection

5- Disadvantages of Data Mining

1. User privacy/security
2. Amount of data is overwhelming
3. Great cost at implementation stage
4. Possible misuse of information
5. Possible in accuracy of data

6-Data Mining Techniques

There are several major ***data mining techniques*** have been developing and using data mining projects recently, including *association*, *classification*, *clustering*, *prediction*, *sequential patterns* and *decision tree*. We will briefly examine those data mining techniques in the following sections.

6-1Data Mining: On What Kind of Data?

There are 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 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.

6-2General Data Mining Functionalities

Data Mining functionalities are used to specify the kind of patterns to be found in data mining tasks.

Data Mining tasks can be classified into two categories

- **Descriptive**: Characterize general properties of data in the database
- **Predictive**: perform inference on data to make predictions

Data Mining Functionalities: Characterization and Discrimination

Data can be associated with classes or concepts that can be described in summarized, concise, and yet precise, terms.
Such descriptions of a concept or class are called class/concept descriptions.
These descriptions can be derived via

- **Data Characterization**
- **Data 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 is typically collected by a query.

Data discrimination is a comparison of the target class data objects against the objects from one or multiple contrasting classes with respect to customers that share specified generalized feature(s).

7- A data mining algorithms

A data mining algorithm is a set of heuristics and calculations that creates a data mining model from data. To create a model, the algorithm first analyzes the data you provide, looking for specific types of patterns or trends.

Choosing the best algorithm to use for a specific analytical task can be a challenge. While you can use different algorithms to perform the same business task, each algorithm produces a different result, and some algorithms can produce more than one type of result.

Choosing an Algorithm by Type

Analysis Services includes the following algorithm types:

- **Classification algorithms** predict one or more discrete variables, based on the other attributes in the dataset.
- **Regression algorithms** predict one or more continuous variables, such as profit or loss, based on other attributes in the dataset.
- **Segmentation algorithms** divide data into groups, or clusters, of items that have similar properties.
- **Association algorithms** find correlations between different attributes in a dataset. The most common application of this kind of algorithm is for creating association rules, which can be used in a market basket analysis.
- **Sequence analysis algorithms** summarize frequent sequences or episodes in data, such as a Web path flow.

8- Overview of Association Rule algorithms

Association rule learning is a popular and well researched method for discovering interesting relations between variables in large databases. It is intended to identify strong rules discovered in databases using different measures of interestingness. Based on the concept of association rules for discovering regularities between products in large-scale transaction data recorded by point-of-sale (POS) systems in supermarkets. For example, the rule $\{\text{onions}, \text{potatoes}\} \Rightarrow \{\text{burger}\}$found in the sales data of a supermarket would indicate that if a customer buys onions and potatoes together, they are likely to also buy hamburger meat.

☐ **Basic terminology:**

1. Tuples are *transactions*, attribute-value pairs are *items*.
2. *Association rule*: {A,B,C,D,...} => {E,F,G,...}, where A,B,C,D,E,F,G,... are items.

3. *Confidence* (accuracy) of A => B : P(B|A) = (# of transactions containing both A and B) / (# of transactions containing A).
4. *Support* (coverage) of A => B : P(A,B) = (# of transactions containing both A and B) / (total # of transactions)
5. We looking for rules that exceed pre-defined support (*minimum support*) and have high confidence.

$$support = \frac{(X \cup Y).count}{n} \qquad confidence = \frac{(X \cup Y).count}{X.count}$$

t1: Beef, Chicken, Milk
t2: Beef, Cheese
t3: Cheese, Boots
t4: Beef, Chicken, Cheese
t5: Beef, Chicken, Clothes, Cheese, Milk
t6: Chicken, Clothes, Milk
t7: Chicken, Milk, Clothes

- Transaction data
- Assume:
  minsup = 30%
  minconf = 80%
- An example frequent *itemset*:

{Chicken, Clothes, Milk} [sup = 3/7]

- Association rules from the itemset:

Clothes → Milk, Chicken [sup = 3/7, conf = 3/3]
…
Clothes, Chicken → Milk, [sup = 3/7, conf = 3/3]

8-1 Apriori Algorithm

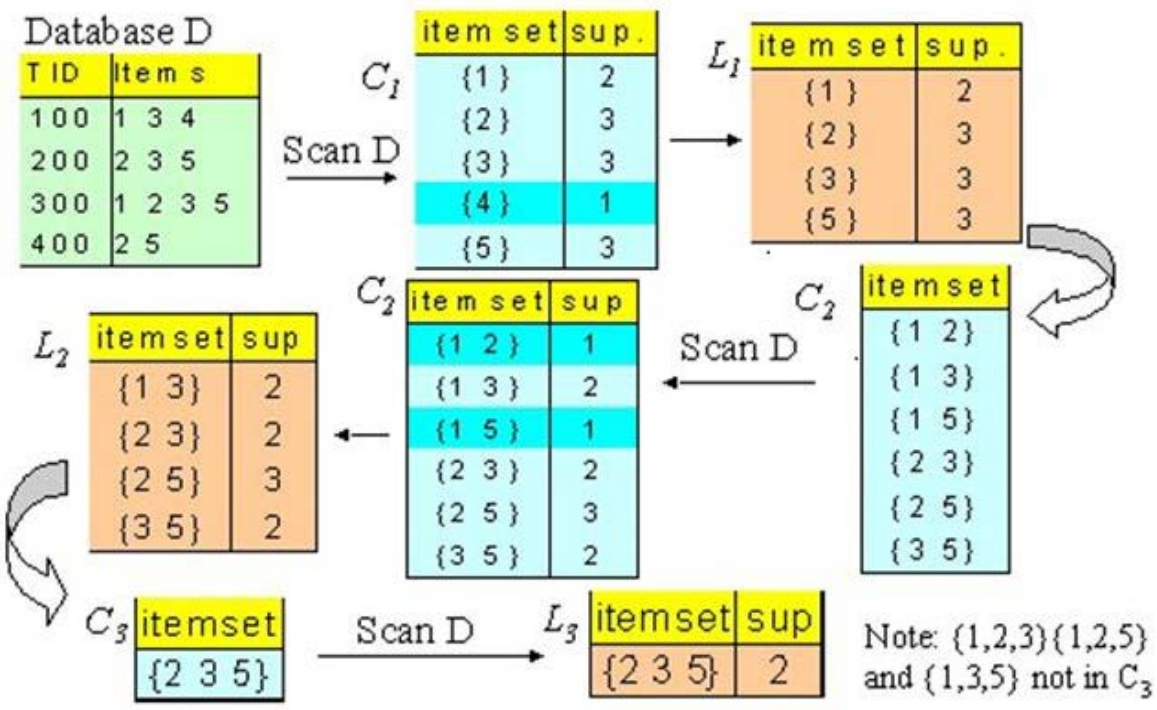
1. Candidate itemsets are generated using only the large itemsets of the previous pass without considering the transactions in the database.

2. The large itemset of the previous pass is joined with itself to generate all itemsets whose size is higher by 1.

3. Each generated itemset that has a subset which is not large is deleted. The remaining itemsets are the candidate ones.

Example: The database of transactions consist of the sets {1,3,4},{2,3,5},{1,2,3,5},{2,5}. We appointed the minimum support level as "1" for this example.

Solution: The first step of Apriori is to count up the frequency (support) of each number (item) separately. Therefore, {4} is not frequent. So, we remove {4} from the first candidate item-set C1. The next step is to generate C2 that is the item-set of all 2-pairs of the frequent items. In the same way, we remove {1,2}, {1,5} whose frequencies are "1". And then, we generate the tree of all possible sets. The third candidate item-set is {{2,3,5}}. Therefore, because of {2,3,5}'s support is 2, the last frequent item-set L3 is {{2,3,5}}. As a note, in the last scan, we did not include {1,2,3}, {1,2,5} and {1,3,5} in the C3 because in the previous scan we identified {1,2}, {1,5} items as infrequent, and $\{1,2\}\subset\{1,2,3\}$, $\{1,2\}\subset\{1,2,5\}$, $\{1,5\}\subset\{1,2,5\}$, and $\{1,5\}\subset\{1,3,5\}$. Now we can identify a association rule between the items of the last item-set.



9- Rough Set Theory

Rough Set Theory, proposed in 1982 by Zdzisl aw Pawlak, is in a state of constant development. Its methodology is concerned with the classification and analysis of imprecise, uncertain or incomplete information and knowledge, and is considered one of the first non-statistical approaches in data analysis (Pawlak, 1982). The fundamental concept behind Rough Set Theory is the approximation of lower and upper spaces of a set, the approximation of spaces being the formal classification of knowledge regarding the interest domain.

9-1 Information Systems Tables

IS is a pair (U, A) , U is a non-empty finite set of objects, A is a non-empty finite set of attributes such that $a:U \to V_a$ for every is called the value set of a.

	Age	LEMS	Walk
x1	16-30	50	yes
x2	16-30	0	no
x3	31-45	1-25	no
x4	31-45	1-25	yes
x5	46-60	26-49	no
x6	16-30	26-49	yes
x7	46-60	26-49	no

9-2 Indiscernibility:

- The equivalence relation

  A binary relation $R \subseteq X \times X$ which is reflexive (*xRx* for any object *x*) , symmetric (if *xRy* then *yRx*), and transitive (if *xRy* and *yRz* then *xRz*).
- The equivalence class $[x]_R$ of an element $x \in X$ consists of all objects $y \in X$ such that *xRy.*
- Let *IS = (U, A)* be an information system, then with any $B \subseteq A$ there is an associated equivalence relation:

$$IND_{IS}(B) = \{(x, x') \in U^2 \mid \forall a \in B, a(x) = a(x')\}$$

  where $IND_{IS}(B)$ is called the *B-indiscernibility relation.*

- If $(x, x') \in IND_{IS}(B)$, then objects *x* and *x'* are indiscernible from each other by attributes from *B.*
- The equivalence classes of the *B-indiscernibility relation* are denoted by $[x]_B$.

- The non-empty subsets of the condition attributes are *{Age}*, *{LEMS}*, and *{Age, LEMS}*.
- *IND({Age}) = {{x1,x2,x6}, {x3,x4}, {x5,x7}}*
- *IND({LEMS}) = {{x1}, {x2}, {x3,x4}, {x5,x6,x7}}*
- *IND({Age,LEMS}) = {{x1}, {x2}, {x3,x4}, {x5,x7}, {x6}}.*

9-3 Set Approximation

- Let $T = (U, A)$ and let $B \subseteq A$ and $X \subseteq U$. We can approximate X using only the information contained in B by constructing the *B-lower* and *B-upper* approximations of X, denoted $\underline{B}X$ and $\overline{B}X$ respectively, where

$$\underline{B}X = \{x \mid [x]_B \subseteq X\},$$

$$\overline{B}X = \{x \mid [x]_B \cap X \neq \phi\}.$$

- *B-boundary region* of X, $BN_B(X) = \overline{B}X - \underline{B}X$, consists of those objects that we cannot decisively classify into X in B.
- *B-outside region* of X, $U - \overline{B}X$,

consists of those objects that can be with certainty classified as not belonging to X.

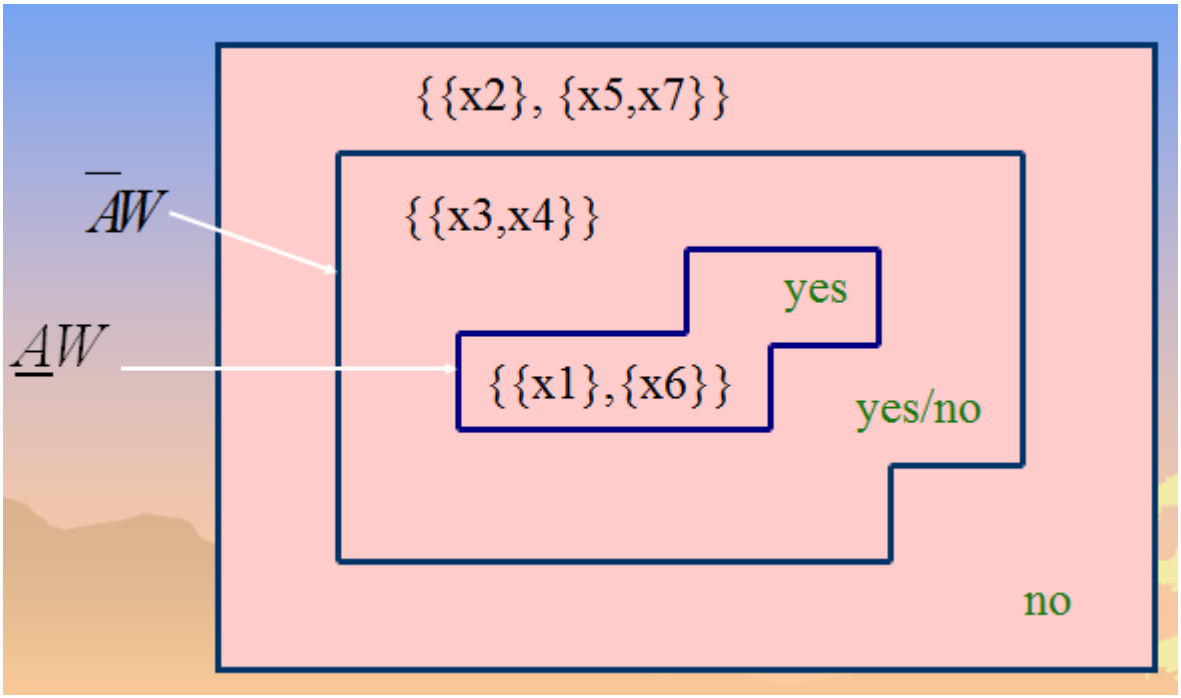
- A set is said to be *rough* if its boundary region is non-empty, otherwise the set is crisp.

- **Let W = {x | Walk(x) = yes}.**

- $\underline{A}W = \{x1, x6\},$
- $\overline{A}W = \{x1, x3, x4, x6\},$
- $BN_A(W) = \{x3, x4\},$

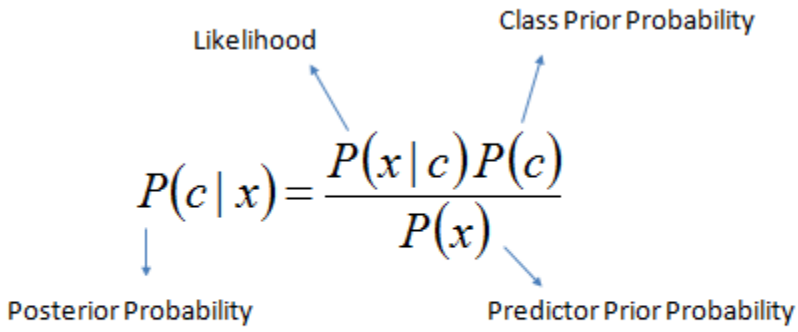
  $U - \overline{A}W = \{x2, x5, x7\}.$

- The decision class, *Walk*, is rough since the boundary region is not empty.



10- Naive Bayesian

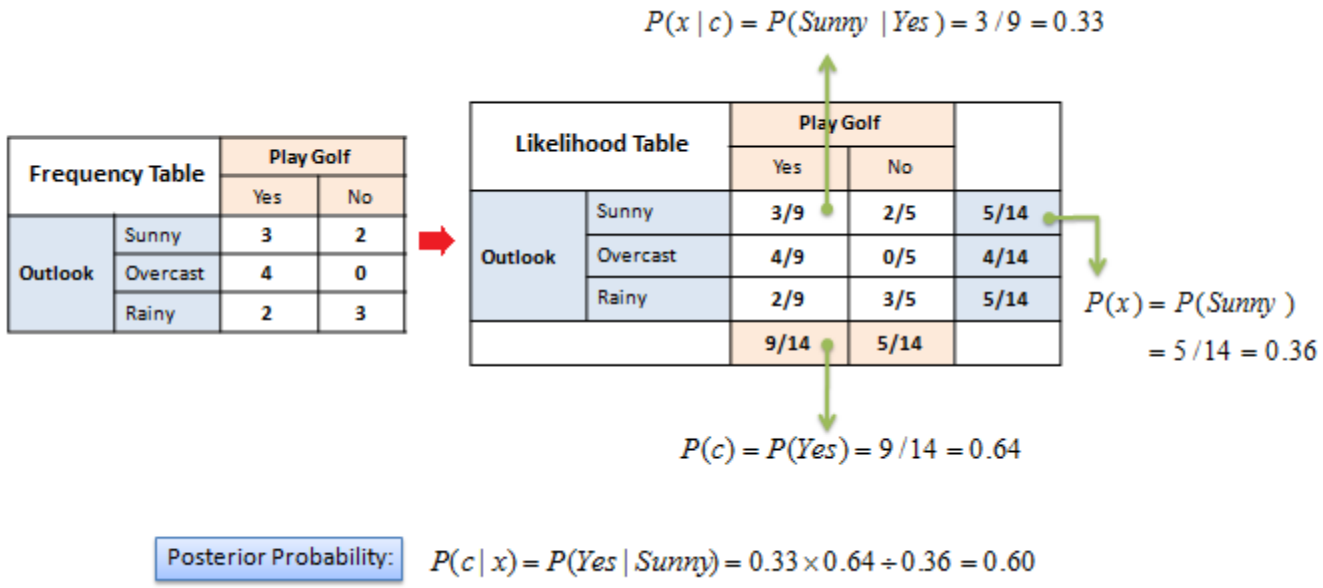
The Naive Bayesian classifier is based on Bayes' theorem with independence assumptions between predictors. A Naive Bayesian model is easy to build, with no complicated iterative parameter estimation which makes it particularly useful for very large datasets. Despite its simplicity, the Naive Bayesian classifier often does surprisingly well and is widely used because it often outperforms more sophisticated classification methods. Algorithm Bayes theorem provides a way of calculating the posterior probability, P(c|x), from P(c), P(x), and P(x|c). Naive Bayes classifier assume that the effect of the value of a predictor (x) on a given class (c) is independent of the values of other predictors. This assumption is called class conditional independence.



$$P(c \mid \mathrm{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

- $P(c|x)$ is the posterior probability of *class* (*target*) given *predictor* (*attribute*).
- $P(c)$ is the prior probability of *class*.
- $P(x|c)$ is the likelihood which is the probability of *predictor* given *class*.
- $P(x)$ is the prior probability of *predictor*.

Example: The posterior probability can be calculated by first, constructing a frequency table for each attribute against the target. Then, transforming the frequency tables to likelihood tables and finally use the Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.



11- K mean Clustering

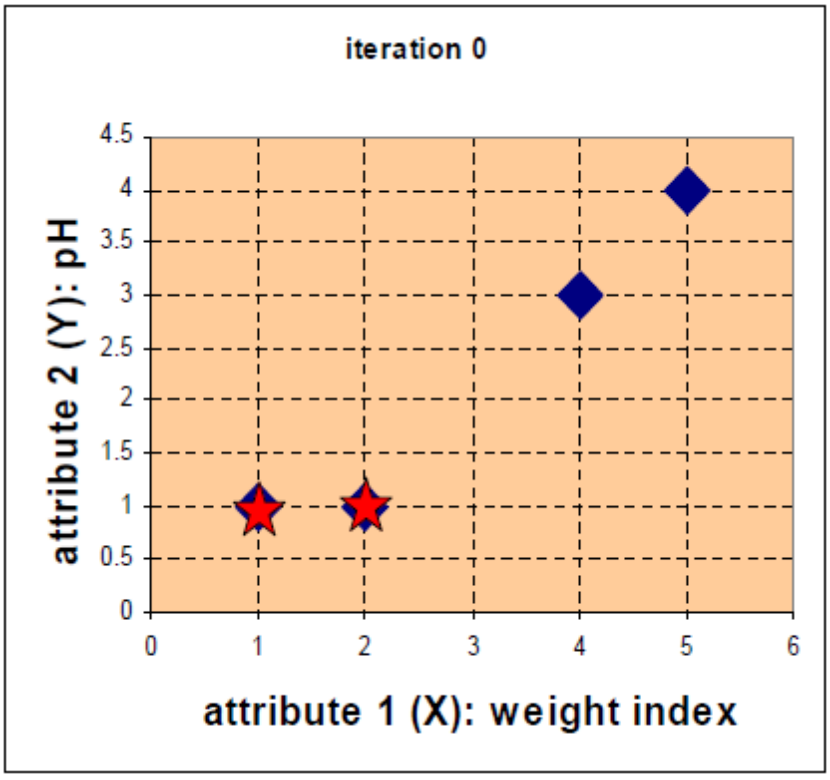
k-means clustering is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining. k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster.

Example: Suppose we have 4 objects as your training data points and each object have 2 attributes. Each attribute represents coordinate of the object.

Object	Attribute 1 (X):weight index	Attribute 2 (Y): pH
Medicine A	1	1
Medicine B	2	1
Medicine C	4	3
Medicine D	5	4

We also know before hand that these objects belong to two groups of medicine (cluster 1 and cluster 2). The problem now is to determine which medicines belong to cluster 1 and which medicines belong to the other cluster.

Our goal is to group these objects into K=2 group of medicine based on the two features (pH and weight index). Each medicine represents one point with two features (X, Y) that we can represent it as coordinate in a feature space as shown in the figure below.



1. *Initial value of centroids*: Suppose we use medicine A and medicine B as the first centroids. Let $\mathbf{c}_1$ and $\mathbf{c}_2$ denote the coordinate of the centroids, then $\mathbf{c}_1 = (1,1)$ and $\mathbf{c}_2 = (2,1)$
2. *Objects-Centroids distance*: we calculate the distance between cluster centroid to each object. Let us use Euclidean distance, then we have distance matrix at iteration 0 is

$$\mathbf{D}^0 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 1 & 0 & 2.83 & 4.24 \end{bmatrix} \quad \begin{matrix} \mathbf{c}_1 = (1,1) & group-1 \\ \mathbf{c}_2 = (2,1) & group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \quad \begin{matrix} X \\ Y \end{matrix}$$

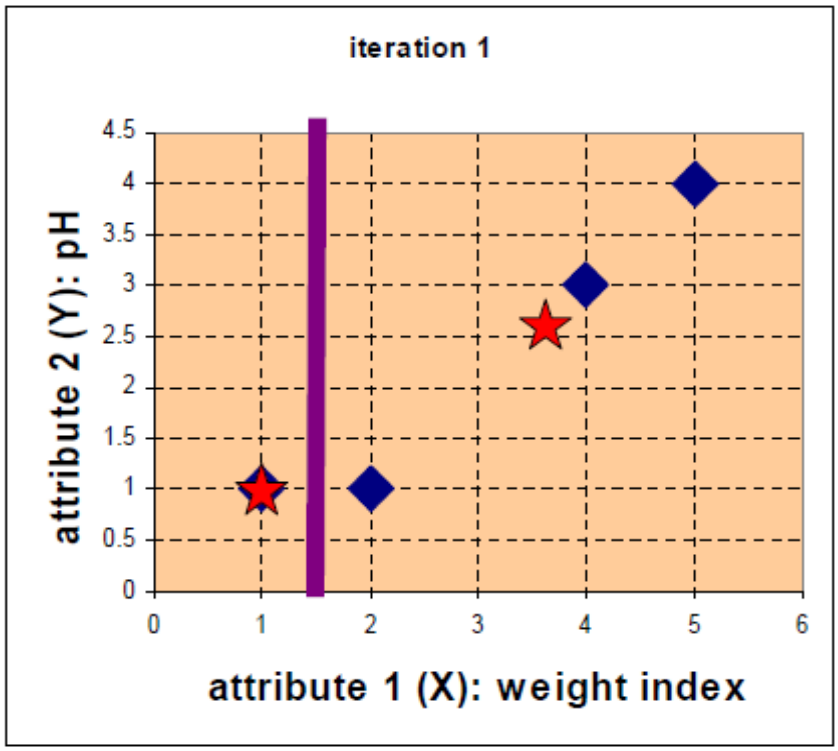
Each column in the distance matrix symbolizes the object. The first row of the distance matrix corresponds to the distance of each object to the first centroid and the second row is the distance of each object to the second centroid. For example, distance from medicine C = (4, 3) to the first centroid $\mathbf{c}_1 = (1,1)$ is $\sqrt{(4-1)^2 + (3-1)^2} = 3.61$, and its distance to the second centroid $\mathbf{c}_2 = (2,1)$ is $\sqrt{(4-2)^2 + (3-1)^2} = 2.83$, etc.

3. *Objects clustering*: We assign each object based on the minimum distance. Thus, medicine A is assigned to group 1, medicine B to group 2, medicine C to group 2 and medicine D to group 2. The element of Group matrix below is 1 if and only if the object is assigned to that group.

$$\mathbf{G}^0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \quad \begin{matrix} group-1 \\ group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

4. *Iteration-1, determine centroids*: Knowing the members of each group, now we compute the new centroid of each group based on these new memberships. Group 1 only has one member thus the centroid remains in $\mathbf{c}_1 = (1,1)$. Group 2 now has three members, thus the centroid is the average coordinate among the three members: $\mathbf{c}_2 = (\frac{2+4+5}{3}, \frac{1+3+4}{3}) = (\frac{11}{3}, \frac{8}{3})$.



5. *Iteration-1, Objects-Centroids distances*: The next step is to compute the distance of all objects to the new centroids. Similar to step 2, we have distance matrix at iteration 1 is

$$\mathbf{D}^1 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 3.14 & 2.36 & 0.47 & 1.89 \end{bmatrix} \quad \begin{matrix} \mathbf{c}_1 = (1,1) & group-1 \\ \mathbf{c}_2 = (\frac{11}{3}, \frac{8}{3}) & group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

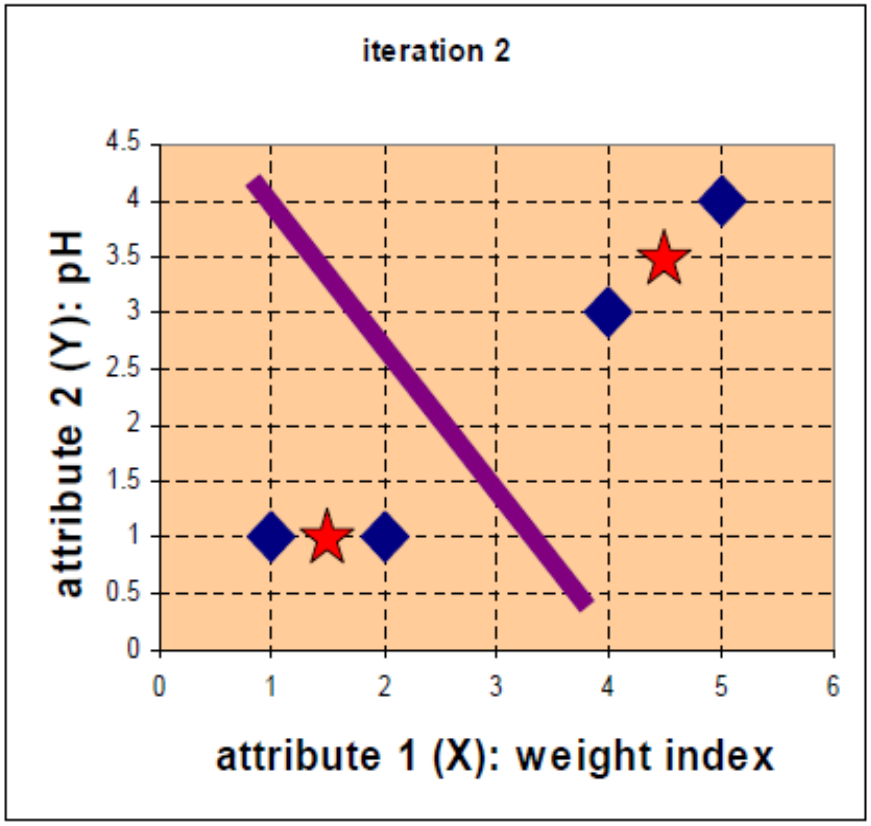
$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \quad \begin{matrix} X \\ Y \end{matrix}$$

6. *Iteration-1, Objects clustering:* Similar to step 3, we assign each object based on the minimum distance. Based on the new distance matrix, we move the medicine B to Group 1 while all the other objects remain. The Group matrix is shown below

$$\mathbf{G}^1 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad \begin{matrix} group-1 \\ group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

7. *Iteration 2, determine centroids:* Now we repeat step 4 to calculate the new centroids coordinate based on the clustering of previous iteration. Group1 and group 2 both has two members, thus the new centroids are $\mathbf{c}_1 = (\frac{1+2}{2}, \frac{1+1}{2}) = (1\frac{1}{2}, 1)$ and $\mathbf{c}_2 = (\frac{4+5}{2}, \frac{3+4}{2}) = (4\frac{1}{2}, 3\frac{1}{2})$



8. *Iteration-2, Objects-Centroids distances*: Repeat step 2 again, we have new distance matrix at iteration 2 as

$$\mathbf{D}^2 = \begin{bmatrix} 0.5 & 0.5 & 3.20 & 4.61 \\ 4.30 & 3.54 & 0.71 & 0.71 \end{bmatrix} \quad \begin{matrix} \mathbf{c}_1 = (1\frac{1}{2}, 1) & group-1 \\ \mathbf{c}_2 = (4\frac{1}{2}, 3\frac{1}{2}) & group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \quad \begin{matrix} X \\ Y \end{matrix}$$

9. *Iteration-2, Objects clustering:* Again, we assign each object based on the minimum distance.

$$\mathbf{G}^2 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad \begin{matrix} group-1 \\ group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

We obtain result that $\mathbf{G}^2 = \mathbf{G}^1$. Comparing the grouping of last iteration and this iteration reveals that the objects does not move group anymore. Thus, the computation of the k-mean clustering has reached its stability and no more iteration is needed. We get the final grouping as the results

Object	Feature 1 (X): weight index	Feature 2 (Y): pH	Group (result)
Medicine A	1	1	1
Medicine B	2	1	1
Medicine C	4	3	2
Medicine D	5	4	2

12- Multimedia Data Mining

- Multimedia data types
  - any type of information medium that can be represented, processed, stored and transmitted over network in digital form
  - Multi-lingual text, numeric, images, video, audio, graphical, temporal, relational, and categorical data.
  - Relation with conventional data mining term

12-1 Generalizing Spatial and Multimedia Data

- Spatial data:
  - Generalize detailed geographic points into clustered regions, such as business, residential, industrial, or agricultural areas, according to land usage
  - Require the merge of a set of geographic areas by spatial operations
- Image data:
  - Extracted by aggregation and/or approximation
  - Size, color, shape, texture, orientation, and relative positions and structures of the contained objects or regions in the image
- Music data:
  - Summarize its melody: based on the approximate patterns that repeatedly occur in the segment

- Summarized its style: based on its tone, tempo, or the major musical instruments played

13- Detecting Intrusions by Data Mining

Intrusion Detection: Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of intrusions, defined as attempts to bypass the security mechanisms of a computer or network (“compromise the confidentiality, integrity, availability of information resources”).

Misuse detection:

Predictive models are built from labeled data sets (instances are labeled as “normal” or “intrusive”) These models can be more sophisticated and precise than manually created signatures Unable to detect attacks whose instances have not yet been observed .

Anomaly detection:

Build models of “normal” behavior and detect anomalies as deviations from it Possible high false alarm rate - previously unseen (yet legitimate) system behaviors may be recognized as anomalies.

13-1 Basic steps in Data Mining for ID

1. Converting the data from the monitored system (computer network, host machine,...) into data (features) that will be used in data mining models.

- For misuse detection, labeling data examples into normal or intrusive may require enormous time for many human experts.

2. Building data mining models
    - Misuse detection models.
    - Anomaly detection models.
3. Analysis and summarization of results.