Introduction, characteristics of data warehouse architecture, goals

Data Warehouse Architecture

A data warehouse architecture is a method of defining the overall architecture of data communication processing and presentation that exist for end-clients computing within the enterprise. Each data warehouse is different, but all are characterized by standard vital components.

Production applications such as payroll accounts payable product purchasing and inventory control are designed for online transaction processing **(OLTP)**. Such applications gather detailed data from day to day operations.

Data Warehouse applications are designed to support the user ad-hoc data requirements, an activity recently dubbed online analytical processing (OLAP). These include applications such as forecasting, profiling, summary reporting, and trend analysis.

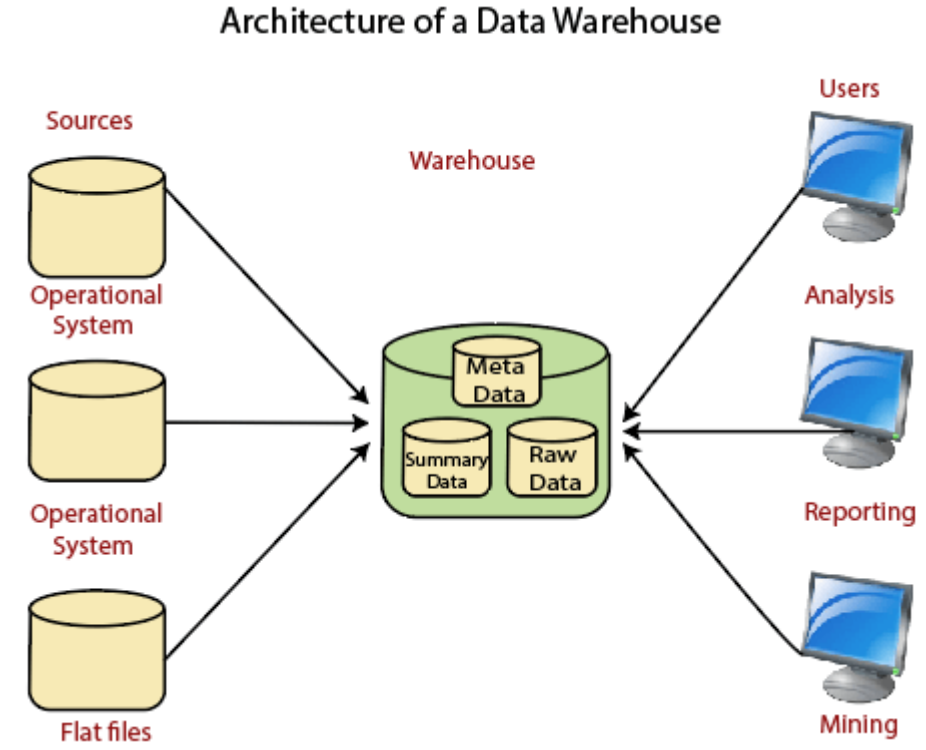
Production databases are updated continuously by either by hand or via OLTP applications. In contrast, a warehouse database is updated from operational systems periodically, usually during off-hours. As OLTP data accumulates in production databases, it is regularly extracted, filtered, and then loaded into a dedicated warehouse server that is accessible to users. As the warehouse is populated, it must be restructured tables de-normalized, data cleansed of errors and redundancies and new fields and keys added to reflect the needs to the user for sorting, combining, and summarizing data.

Data warehouses and their architectures very depending upon the elements of an organization's situation.

Three common architectures are:

- Data Warehouse Architecture: Basic
- Data Warehouse Architecture: With Staging Area
- Data Warehouse Architecture: With Staging Area and Data Marts

Data Warehouse Architecture: Basic



Operational System

An **operational system** is a method used in data warehousing to refer to a **system** that is used to process the day-to-day transactions of an organization.

Flat Files

A **Flat file** system is a system of files in which transactional data is stored, and every file in the system must have a different name.

Meta Data

A set of data that defines and gives information about other data.

Meta Data used in Data Warehouse for a variety of purpose, including:

Meta Data summarizes necessary information about data, which can make finding and work with particular instances of data more accessible. For example, author, data build, and data changed, and file size are examples of very basic document metadata.

Metadata is used to direct a query to the most appropriate data source.

Lightly and highly summarized data

The area of the data warehouse saves all the predefined lightly and highly summarized (aggregated) data generated by the warehouse manager.

The goals of the summarized information are to speed up query performance. The summarized record is updated continuously as new information is loaded into the warehouse.

End-User access Tools

The principal purpose of a data warehouse is to provide information to the business managers for strategic decision-making. These customers interact with the warehouse using end-client access tools.

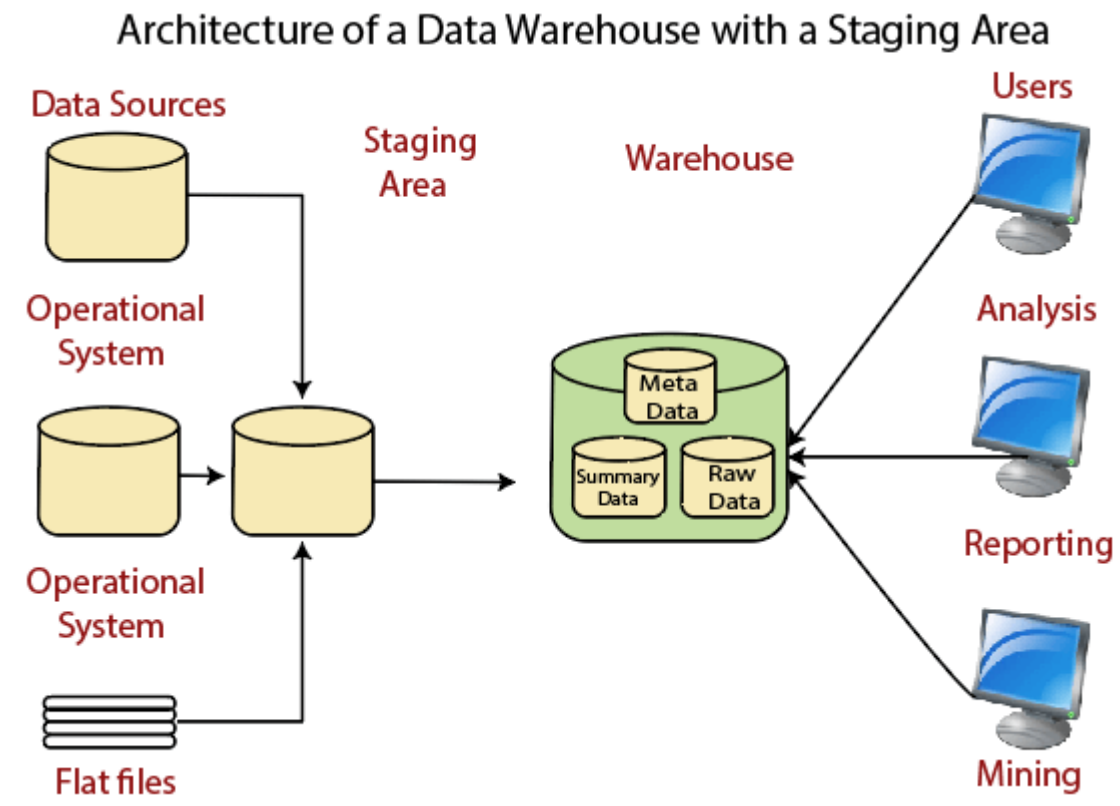
The examples of some of the end-user access tools can be:

- Reporting and Query Tools
- Application Development Tools
- Executive Information Systems Tools
- Online Analytical Processing Tools
- Data Mining Tools

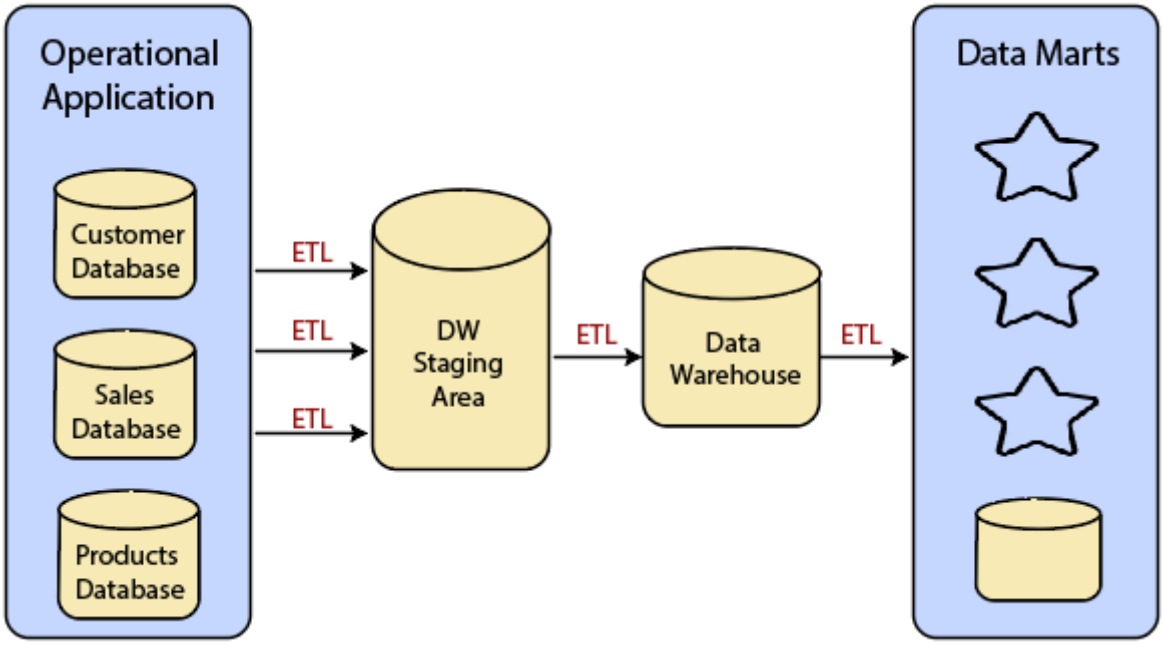
Data Warehouse Architecture: With Staging Area

We must clean and process your operational information before put it into the warehouse.
We can do this programmatically, although data warehouses uses a **staging area** (A place where data is processed before entering the warehouse).

A staging area simplifies data cleansing and consolidation for operational method coming from multiple source systems, especially for enterprise data warehouses where all relevant data of an enterprise is consolidated.



Data Warehouse Staging Area is a temporary location where a record from source systems is copied.

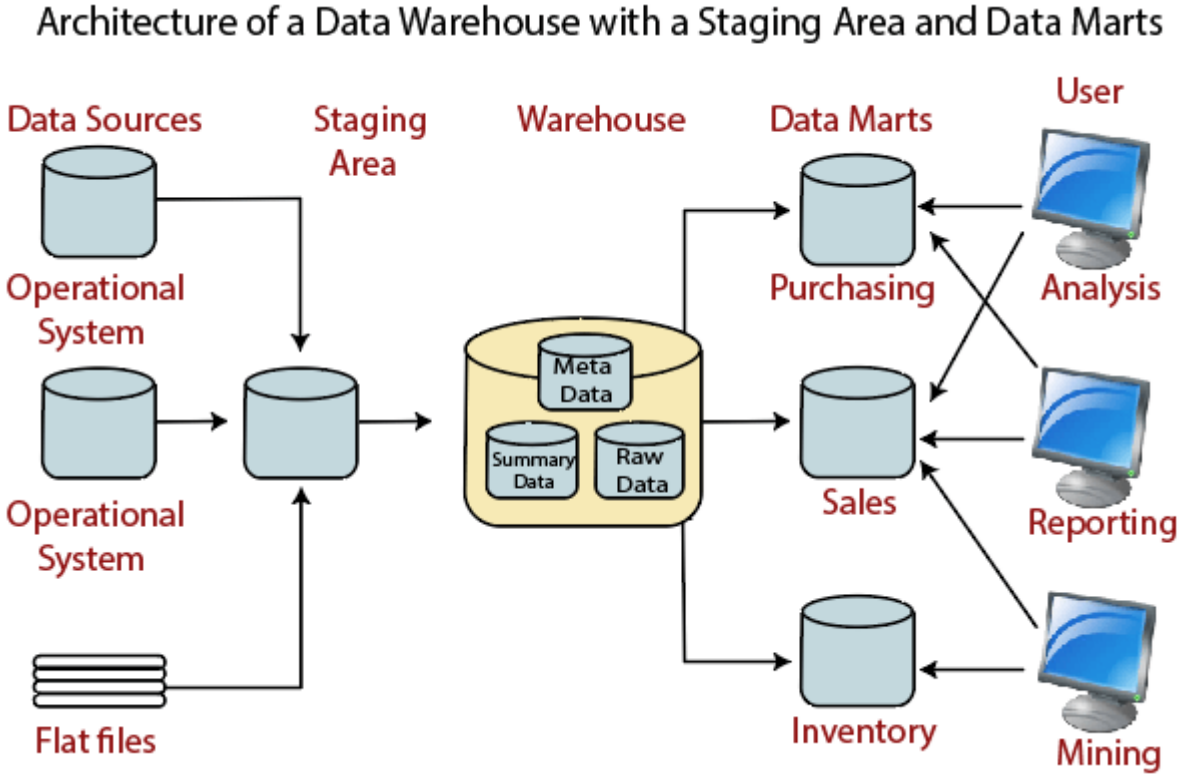


Data Warehouse Architecture: With Staging Area and Data Marts

We may want to customize our warehouse's architecture for multiple groups within our organization.

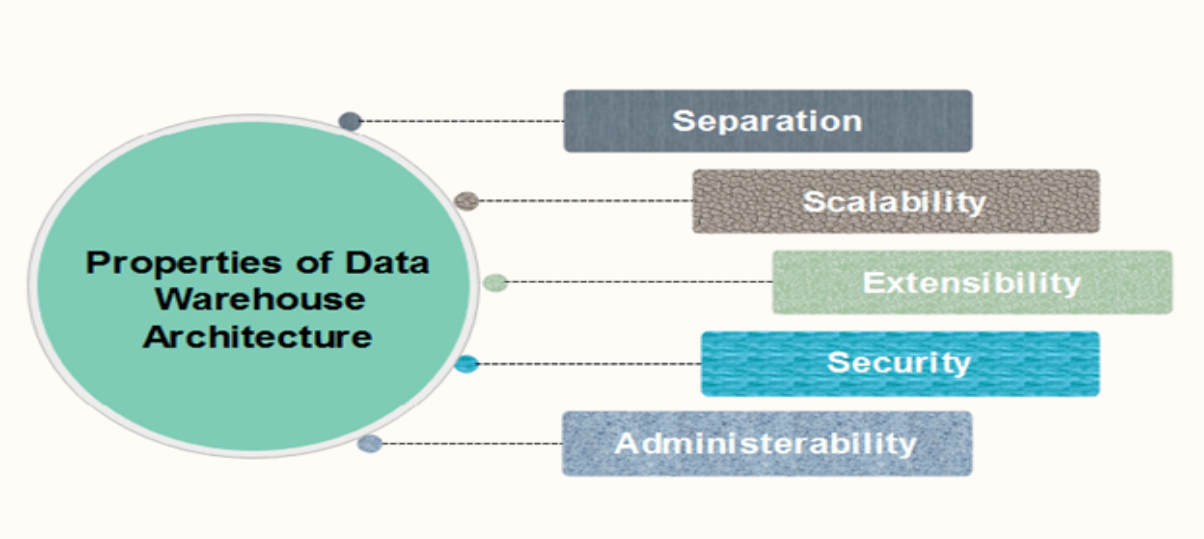
We can do this by adding **data marts**. A data mart is a segment of a data warehouses that can provided information for reporting and analysis on a section, unit, department or operation in the company, e.g., sales, payroll, production, etc.

The figure illustrates an example where purchasing, sales, and stocks are separated. In this example, a financial analyst wants to analyze historical data for purchases and sales or mine historical information to make predictions about customer behavior.



Properties of Data Warehouse Architectures

The following architecture properties are necessary for a data warehouse system:



1. Separation: Analytical and transactional processing should be keep apart as much as possible.

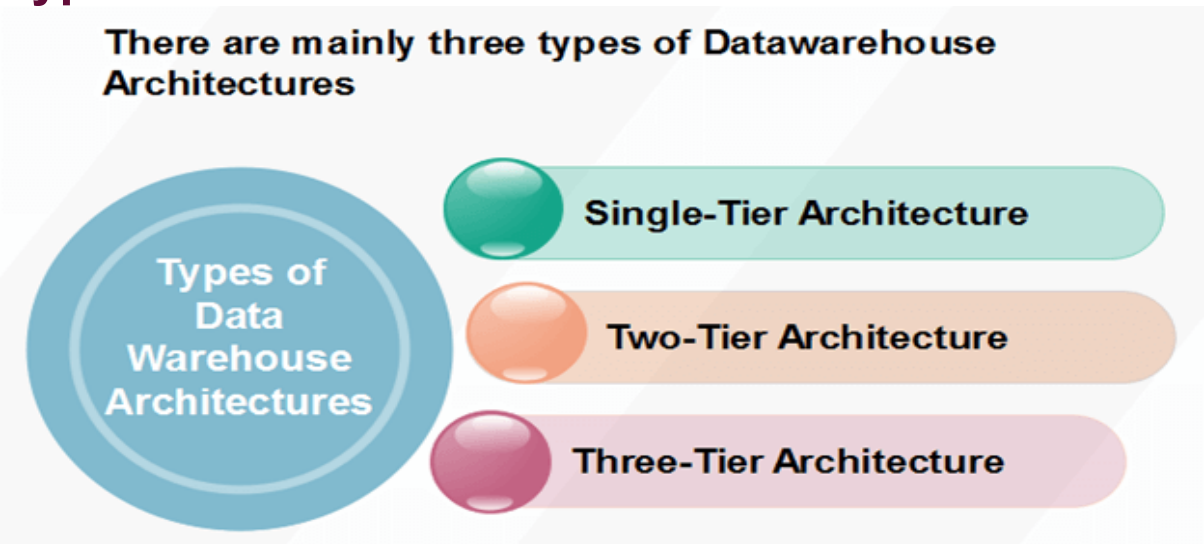
2. Scalability: Hardware and software architectures should be simple to upgrade the data volume, which has to be managed and processed, and the number of user's requirements, which have to be met, progressively increase.

3. Extensibility: The architecture should be able to perform new operations and technologies without redesigning the whole system.

4. Security: Monitoring accesses are necessary because of the strategic data stored in the data warehouses.

5. Administerability: Data Warehouse management should not be complicated.

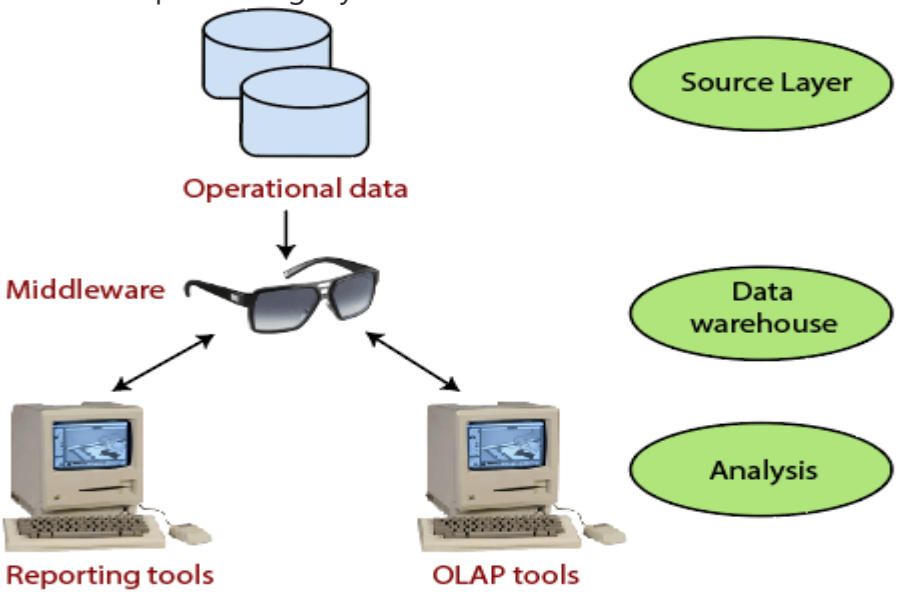
Types of Data Warehouse Architectures



Single-Tier Architecture

Single-Tier architecture is not periodically used in practice. Its purpose is to minimize the amount of data stored to reach this goal; it removes data redundancies.

The figure shows the only layer physically available is the source layer. In this method, data warehouses are virtual. This means that the data warehouse is implemented as a multidimensional view of operational data created by specific middleware, or an intermediate processing layer.

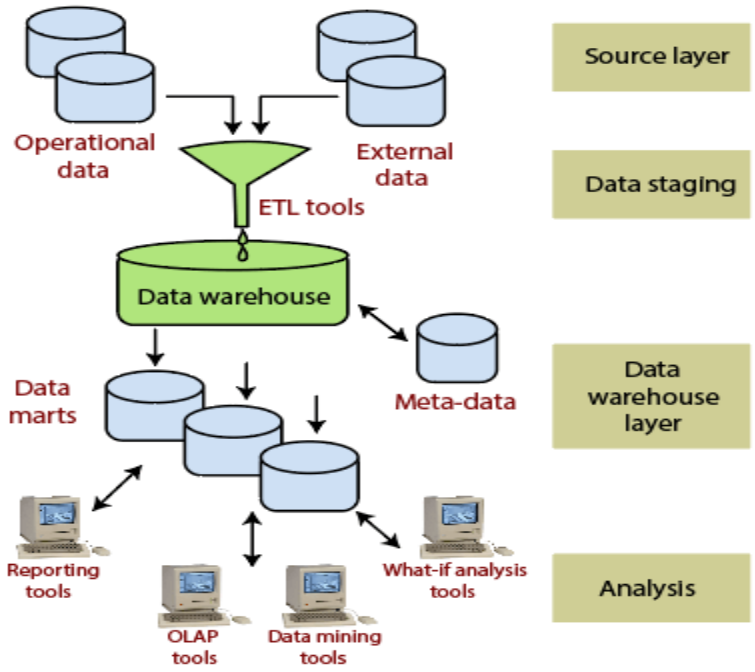


Single-Tier Data Warehouse Architecture

The vulnerability of this architecture lies in its failure to meet the requirement for separation between analytical and transactional processing. Analysis queries are agreed to operational data after the middleware interprets them. In this way, queries affect transactional workloads.

Two-Tier Architecture

The requirement for separation plays an essential role in defining the two-tier architecture for a data warehouse system, as shown in fig:



Two-Tier Data Warehouse Architecture

Although it is typically called two-layer architecture to highlight a separation between physically available sources and data warehouses, in fact, consists of four subsequent data flow stages:

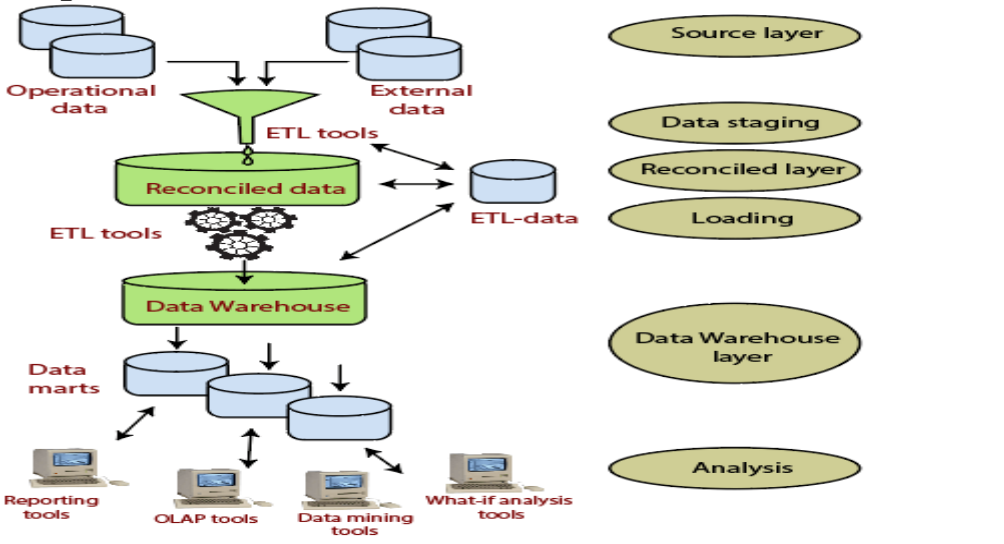
1. **Source layer:** A data warehouse system uses a heterogeneous source of data. That data is stored initially to corporate relational databases or legacy databases, or it may come from an information system outside the corporate walls.
2. **Data Staging:** The data stored to the source should be extracted, cleansed to remove inconsistencies and fill gaps, and integrated to merge heterogeneous sources into one standard schema. The so-named **Extraction**, **Transformation**, and **Loading Tools (ETL)** can combine heterogeneous schemata, extract, transform, cleanse, validate, filter, and load source data into a data warehouse.
3. **Data Warehouse layer:** Information is saved to one logically centralized individual repository: a data warehouse. The data warehouses can be directly accessed, but it can also be used as a source for creating data marts, which partially replicate data warehouse contents and are designed for specific enterprise departments. Meta-data repositories store information on sources, access procedures, data staging, users, data mart schema, and so on.
4. **Analysis:** In this layer, integrated data is efficiently, and flexible accessed to issue reports, dynamically analyze information, and simulate hypothetical business scenarios. It should feature aggregate information navigators, complex query optimizers, and customer-friendly GUIs.

Three-Tier Architecture

The three-tier architecture consists of the source layer (containing multiple source system), the reconciled layer and the data warehouse layer (containing both data warehouses and data marts). The reconciled layer sits between the source data and data warehouse.

The main advantage of the **reconciled layer** is that it creates a standard reference data model for a whole enterprise. At the same time, it separates the problems of source data extraction and integration from those of data warehouse population. In some cases, the **reconciled layer** is also directly used to accomplish better some operational tasks, such as producing daily reports that cannot be satisfactorily prepared using the corporate applications or generating data flows to feed external processes periodically to benefit from cleaning and integration.

This architecture is especially useful for the extensive, enterprise-wide systems. A disadvantage of this structure is the extra file storage space used through the extra redundant reconciled layer. It also makes the analytical tools a little further away from being real-time.



Three-Tier Architecture for a data warehouse system

Three-Tier Data Warehouse Architecture

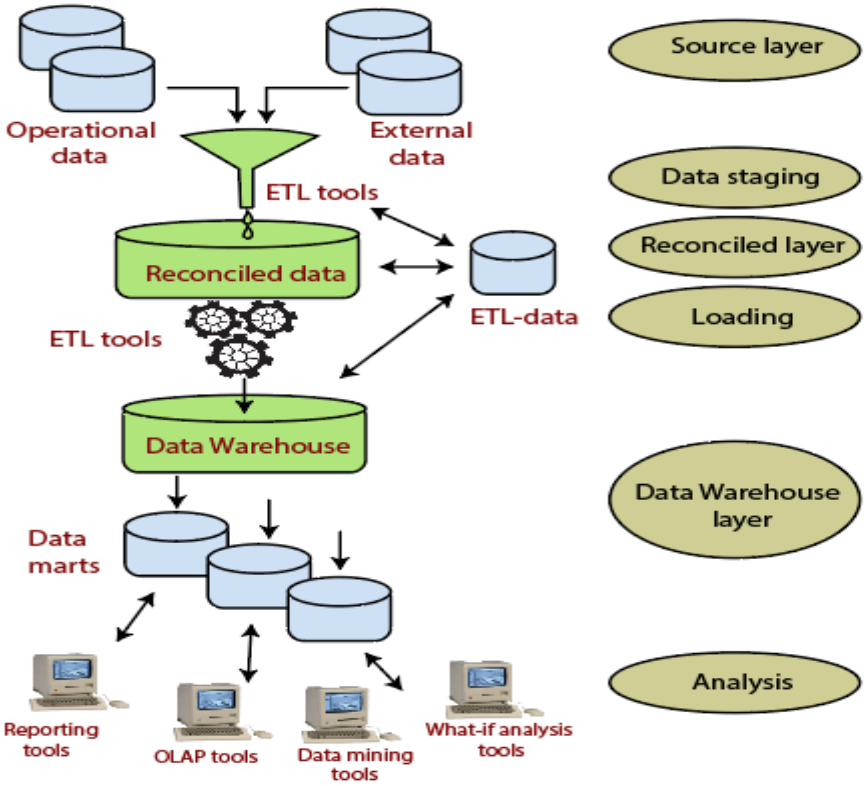
Data Warehouses usually have a three-level (tier) architecture that includes:

1. Bottom Tier (Data Warehouse Server)
2. Middle Tier (OLAP Server)
3. Top Tier (Front end Tools).

A **bottom-tier** that consists of the **Data Warehouse server**, which is almost always an RDBMS. It may include several specialized data marts and a metadata repository.

Data from operational databases and external sources (such as user profile data provided by external consultants) are extracted using application program interfaces called a gateway. A gateway is provided by the underlying DBMS and allows customer programs to generate SQL code to be executed at a server.

Examples of gateways contain **ODBC** (Open Database Connection) and **OLE-DB** (Open-Linking and Embedding for Databases), by **Microsoft**, and **JDBC** (Java Database Connection).



Three-Tier Architecture for a data warehouse system

A **middle-tier** which consists of an **OLAP server** for fast querying of the data warehouse.

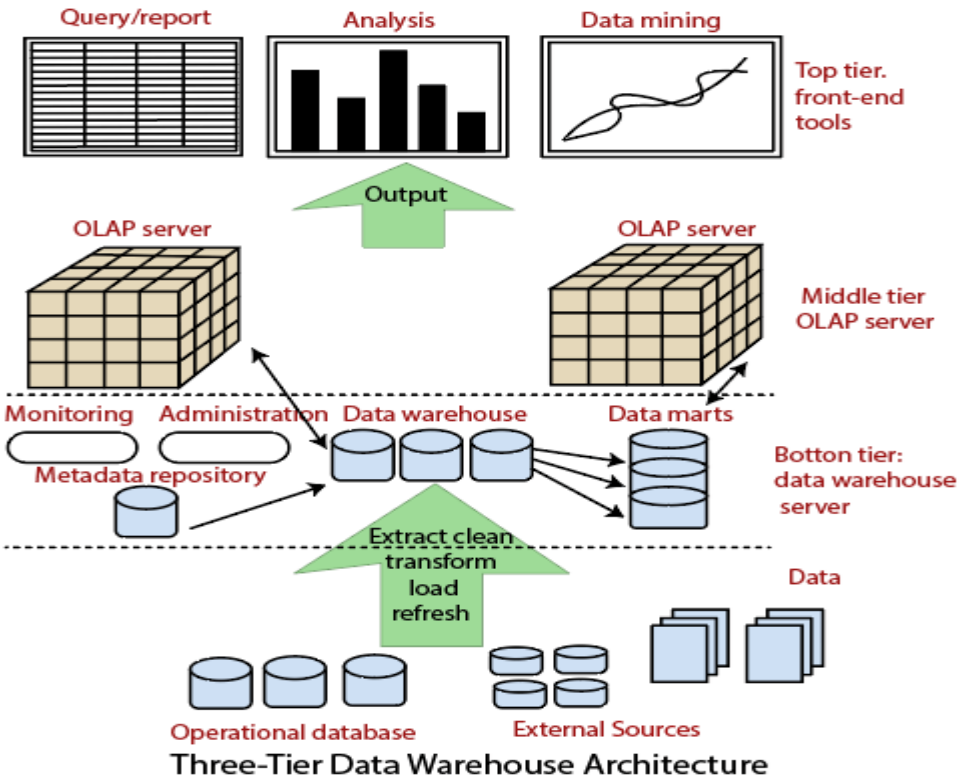
The OLAP server is implemented using either

(1) A **Relational OLAP (ROLAP) model**, i.e., an extended relational DBMS that maps functions on multidimensional data to standard relational operations.

(2) A **Multidimensional OLAP (MOLAP) model**, i.e., a particular purpose server that directly implements multidimensional information and operations.

A **top-tier** that contains **front-end tools** for displaying results provided by OLAP, as well as additional tools for data mining of the OLAP-generated data.

The overall Data Warehouse Architecture is shown in fig:

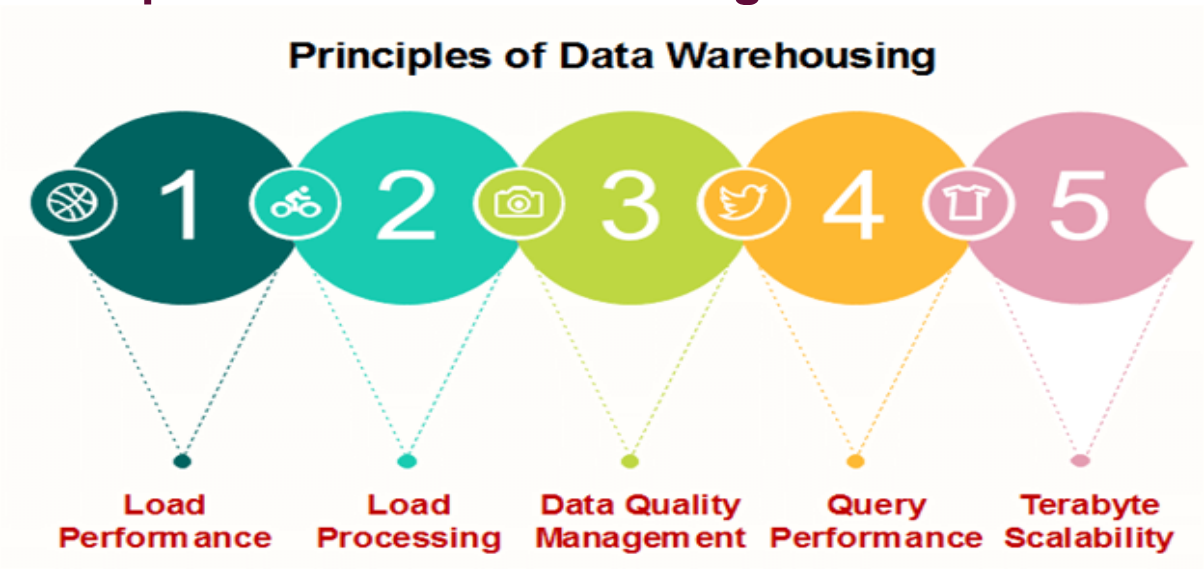


Three-Tier Data Warehouse Architecture

The **metadata repository** stores information that defines DW objects. It includes the following parameters and information for the middle and the top-tier applications:

1. A description of the DW structure, including the warehouse schema, dimension, hierarchies, data mart locations, and contents, etc.
2. Operational metadata, which usually describes the currency level of the stored data, i.e., active, archived or purged, and warehouse monitoring information, i.e., usage statistics, error reports, audit, etc.
3. System performance data, which includes indices, used to improve data access and retrieval performance.
4. Information about the mapping from operational databases, which provides source **RDBMSs** and their contents, cleaning and transformation rules, etc.
5. Summarization algorithms, predefined queries, and reports business data, which include business terms and definitions, ownership information, etc.

Principles of Data Warehousing



Load Performance

Data warehouses require increase loading of new data periodically basis within narrow time windows; performance on the load process should be measured in hundreds of millions of rows and gigabytes per hour and must not artificially constrain the volume of data business.

Load Processing

Many phases must be taken to load new or update data into the data warehouse, including data conversion, filtering, reformatting, indexing, and metadata update.

Data Quality Management

Fact-based management demands the highest data quality. The warehouse ensures local consistency, global consistency, and referential integrity despite "dirty" sources and massive database size.

Query Performance

Fact-based management must not be slowed by the performance of the data warehouse RDBMS; large, complex queries must be complete in seconds, not days.

Terabyte Scalability

Data warehouse sizes are growing at astonishing rates. Today these size from a few to hundreds of gigabytes and terabyte-sized data warehouses.

Difference between OLAP and OLTP

Online Analytical Processing (OLAP) –
Online Analytical Processing consists of a type of software tools that are used for data analysis for business decisions. OLAP provides an environment to get insights from the database retrieved from multiple database systems at one time.

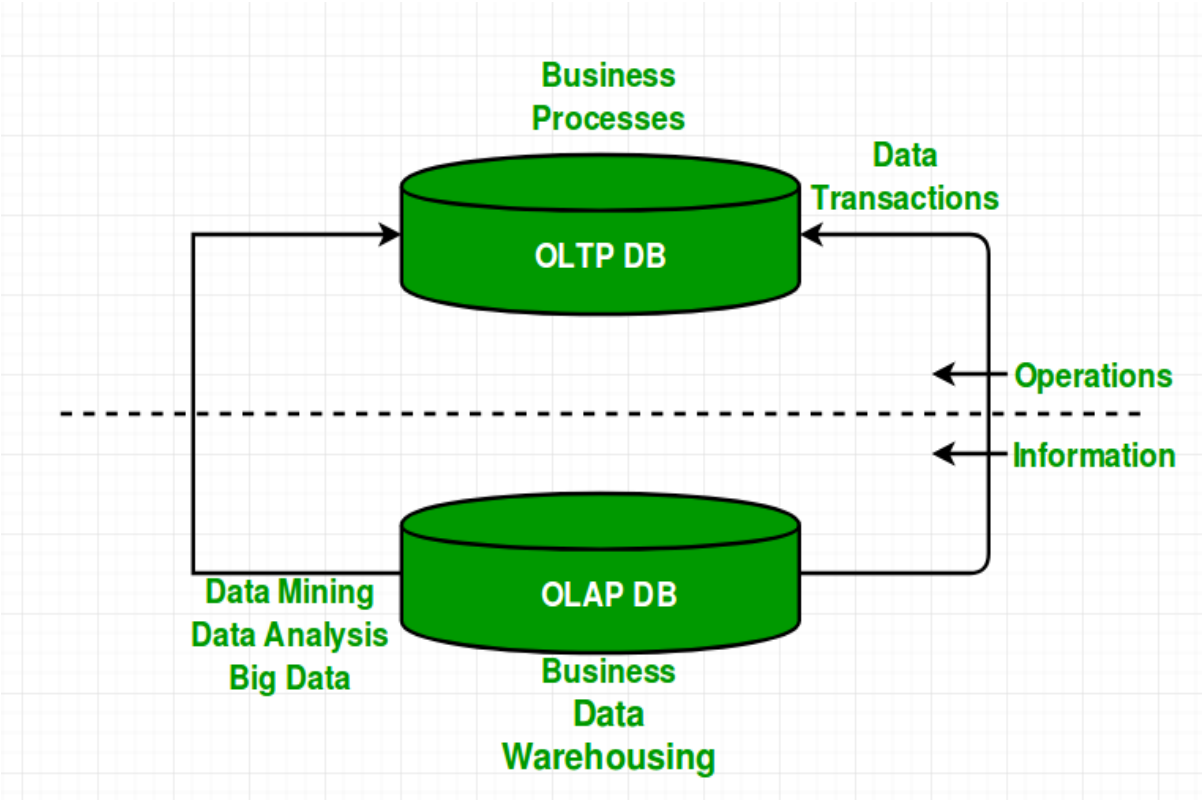
Examples – Any type of Data warehouse system is an OLAP system. Uses of OLAP are as follows:

- Spotify analyzed songs by users to come up with the personalized homepage of their songs and playlist.
- Netflix movie recommendation system.

Online transaction processing (OLTP) –
Online transaction processing provides transaction-oriented applications in a 3-tier architecture. OLTP administers day to day transaction of an organization.

Examples – Uses of OLTP are as follows:

- ATM center is an OLTP application.
- OLTP handles the ACID properties during data transaction via the application.
- It's also used for Online banking, Online airline ticket booking, sending a text message, add a book to the shopping cart.



- **Comparisons of OLAP vs OLTP –**

OLAP (Online analytical processing)	OLTP (Online transaction processing)
Consists of historical data from various Databases.	Consists only operational current data.
It is subject oriented. Used for Data Mining, Analytics, Decision making,etc.	It is application oriented. Used for business tasks.
The data is used in planning, problem solving and decision making.	The data is used to perform day to day fundamental operations.
It reveals a snapshot of present business tasks.	It provides a multi-dimensional view of different business tasks.
Large amount of data is stored typically in TB, PB	The size of the data is relatively small as the historical data is archived. For ex MB, GB
Relatively slow as the amount of data involved is large. Queries may take hours.	Very Fast as the queries operate on 5% of the data.
It only need backup from time to time as compared to OLTP.	Backup and recovery process is maintained religiously
This data is generally managed by CEO, MD, GM.	This data is managed by clerks, managers.
Only read and rarely write operation.	Both read and write operations.

OLAP

What is OLAP? Cube, Operations & Types in Data Warehouse

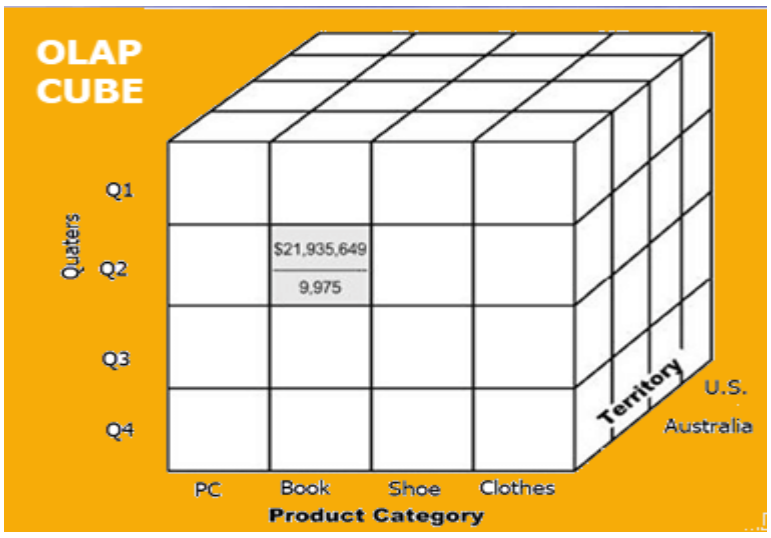
What is OLAP?

Online Analytical Processing (OLAP) is a category of software that allows users to analyze information from multiple database systems at the same time. It is a technology that enables analysts to extract and view business data from different points of view.

Analysts frequently need to group, aggregate and join data. These operations in relational databases are resource intensive. With OLAP data can be pre-calculated and pre-aggregated, making analysis faster.

OLAP databases are divided into one or more cubes. The cubes are designed in such a way that creating and viewing reports become easy. OLAP stands for Online Analytical Processing.

OLAP cube:



At the core of the OLAP concept, is an OLAP Cube. The OLAP cube is a data structure optimized for very quick data analysis.

The OLAP Cube consists of numeric facts called measures which are categorized by dimensions. OLAP Cube is also called the **hypercube**.

Usually, data operations and analysis are performed using the simple spreadsheet, where data values are arranged in row and column format. This is ideal for two-dimensional data. However, OLAP contains multidimensional data, with data usually obtained from a different and unrelated source. Using a spreadsheet is not an

optimal option. The cube can store and analyze multidimensional data in a logical and orderly manner.

How does it work?

A Data warehouse would extract information from multiple data sources and formats like text files, excel sheet, multimedia files, etc.

The extracted data is cleaned and transformed. Data is loaded into an OLAP server (or OLAP cube) where information is pre-calculated in advance for further analysis.

Basic analytical operations of OLAP

Four types of analytical operations in OLAP are:

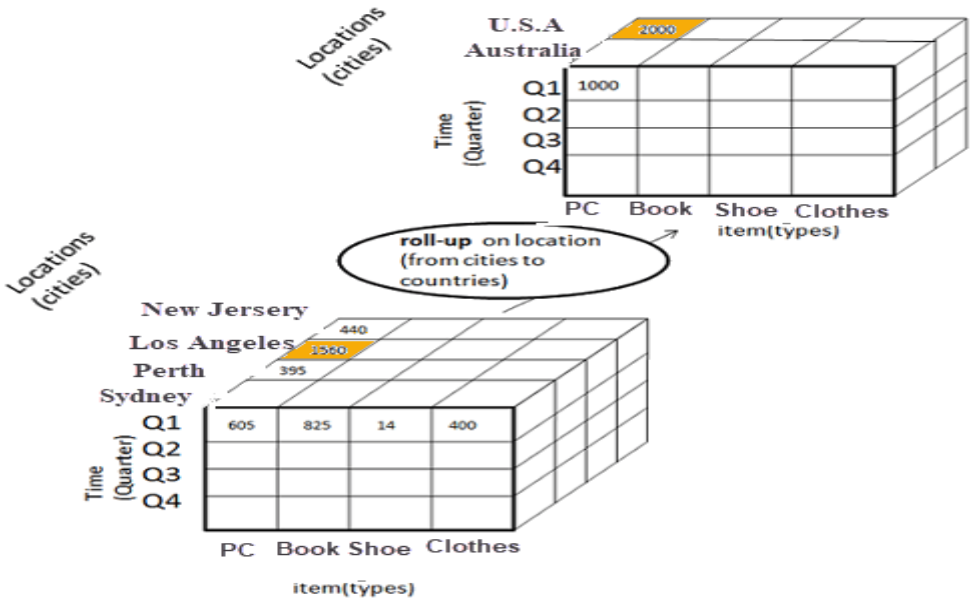
1. Roll-up
2. Drill-down
3. Slice and dice
4. Pivot (rotate)

1) Roll-up:

Roll-up is also known as "consolidation" or "aggregation." The Roll-up operation can be performed in 2 ways

1. Reducing dimensions
2. Climbing up concept hierarchy. Concept hierarchy is a system of grouping things based on their order or level.

Consider the following diagram

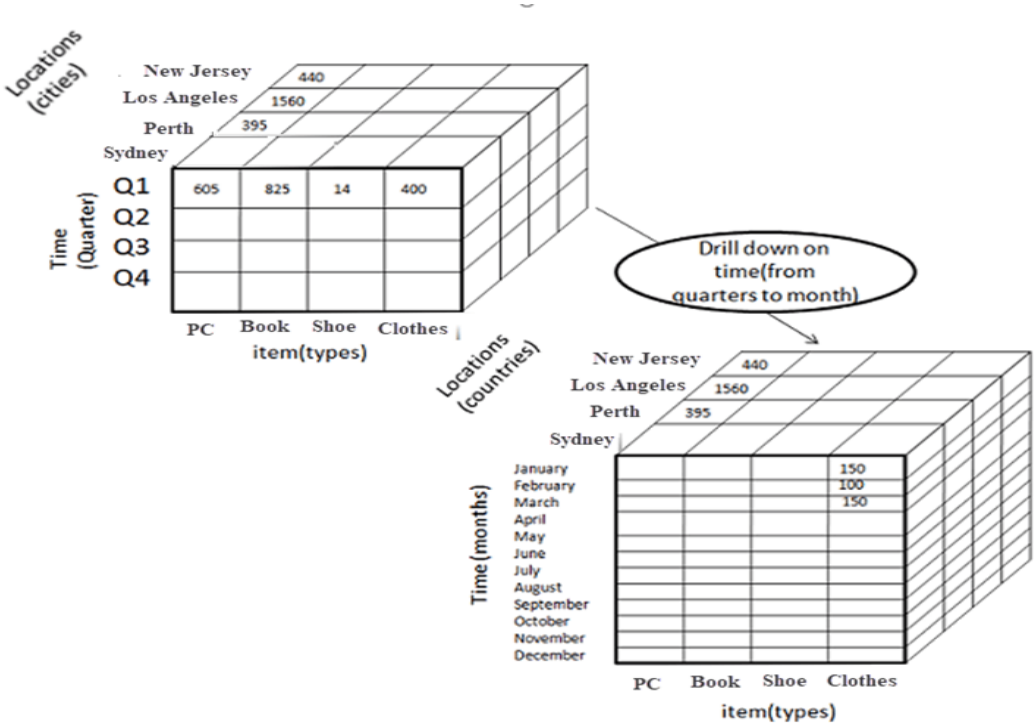


- In this example, cities New jersey and Lost Angles and rolled up into country USA
- The sales figure of New Jersey and Los Angeles are 440 and 1560 respectively. They become 2000 after roll-up
- In this aggregation process, data is location hierarchy moves up from city to the country.
- In the roll-up process at least one or more dimensions need to be removed. In this example, Quater dimension is removed.

2) Drill-down

In drill-down data is fragmented into smaller parts. It is the opposite of the rollup process. It can be done via

- Moving down the concept hierarchy
- Increasing a dimension



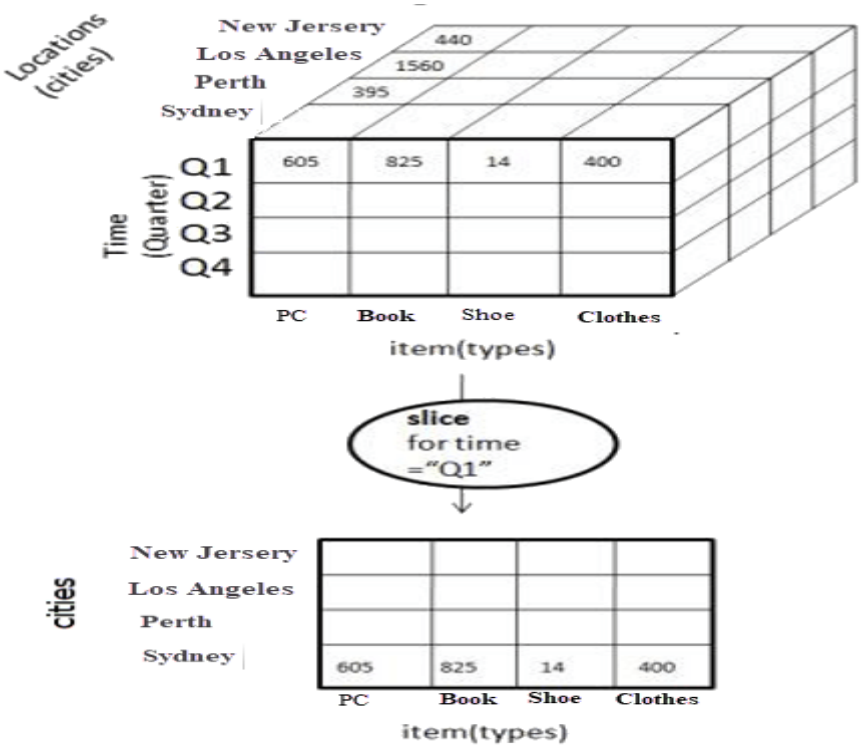
Consider the diagram above

- Quater Q1 is drilled down to months January, February, and March. Corresponding sales are also registers.
- In this example, dimension months are added.

3) Slice:

Here, one dimension is selected, and a new sub-cube is created.

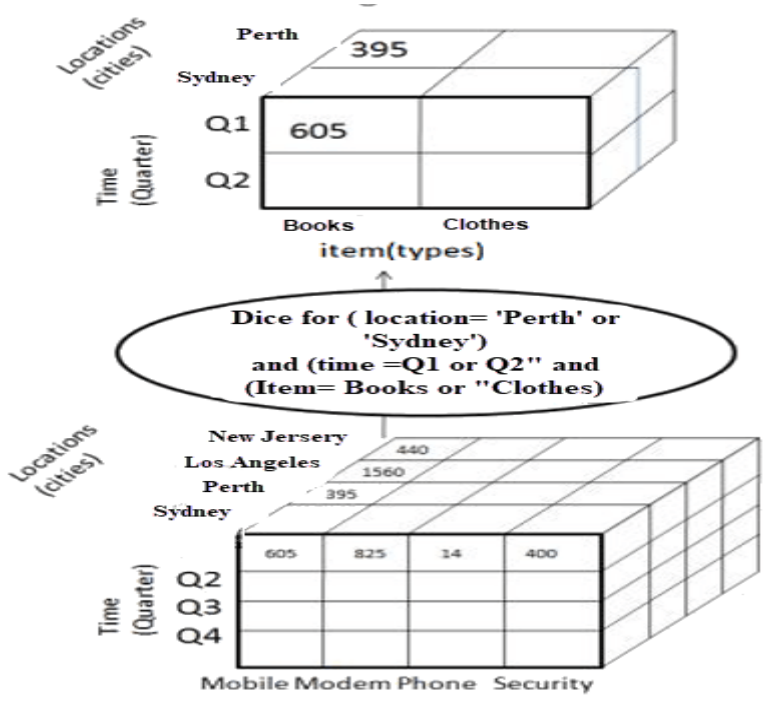
Following diagram explain how slice operation performed:



- Dimension Time is Sliced with Q1 as the filter.
- A new cube is created altogether.

Dice:

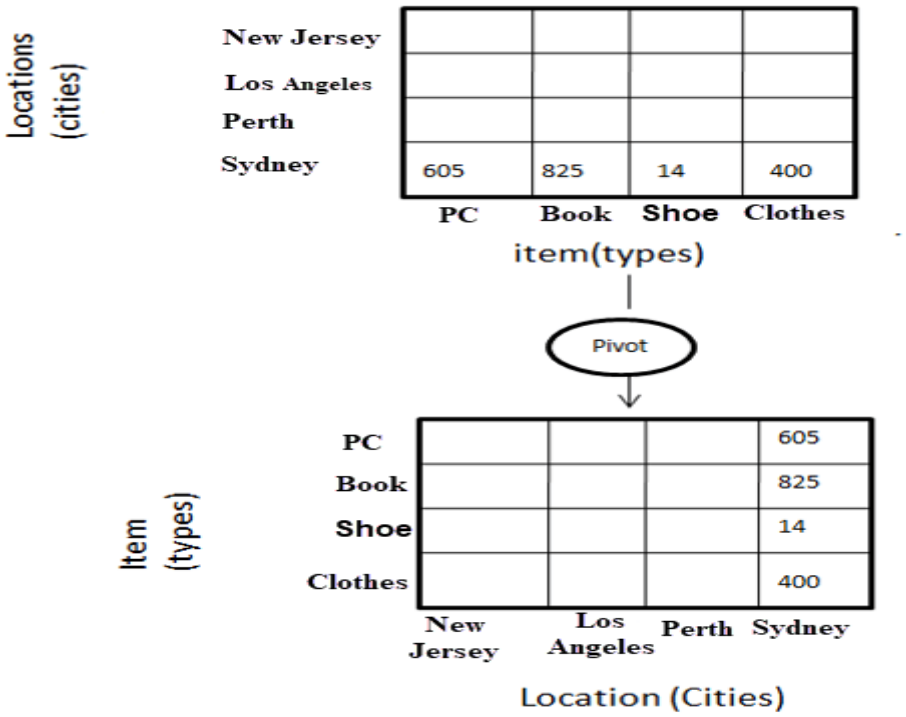
This operation is similar to a slice. The difference in dice is you select 2 or more dimensions that result in the creation of a sub-cube.



4) Pivot

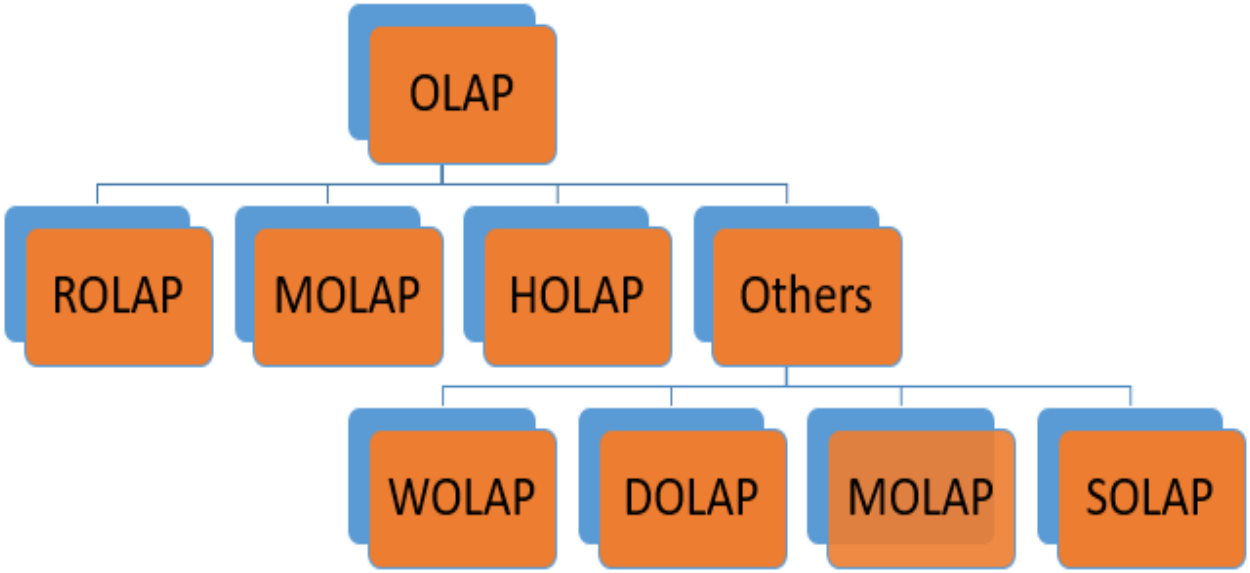
In Pivot, you rotate the data axes to provide a substitute presentation of data.

In the following example, the pivot is based on item types.



Types of OLAP systems

OLAP Hierarchical Structure



Type of OLAP	Explanation
Relational OLAP(ROLAP):	ROLAP is an extended RDBMS along with multidimensional data mapping to perform the standard relational operation.
Multidimensional OLAP (MOLAP)	MOLAP Implementes operation in multidimensional data.
Hybrid OnlineAnalytical Processing (HOLAP)	In HOLAP approach the aggregated totals are stored in a multidimensional database while the detailed data is stored in the relational database. This offers both data efficiency of the ROLAP model and the performance of the MOLAP model.
Desktop OLAP (DOLAP)	In Desktop OLAP, a user downloads a part of the data from the database locally, or on their desktop and analyze it. DOLAP is relatively cheaper to deploy as it offers very few functionalities compares to other OLAP systems.
Web OLAP (WOLAP)	Web OLAP which is OLAP system accessible via the web browser. WOLAP is a three-tiered architecture. It consists of three components: client, middleware, and a database server.
Mobile OLAP:	Mobile OLAP helps users to access and analyze OLAP data using their mobile devices
Spatial OLAP :	SOLAP is created to facilitate management of both spatial and non-spatial data in a Geographic Information system (GIS)

ROLAP

ROLAP works with data that exist in a relational database. Facts and dimension tables are stored as relational tables. It also allows multidimensional analysis of data and is the fastest growing OLAP.

Advantages of ROLAP model:

- **High data efficiency.** It offers high data efficiency because query performance and access language are optimized particularly for the multidimensional data analysis.
- **Scalability.** This type of OLAP system offers scalability for managing large volumes of data, and even when the data is steadily increasing.

Drawbacks of ROLAP model:

- **Demand for higher resources:** ROLAP needs high utilization of manpower, software, and hardware resources.
- **Aggregately data limitations.** ROLAP tools use SQL for all calculation of aggregate data. However, there are no set limits to the for handling computations.
- **Slow query performance.** Query performance in this model is slow when compared with MOLAP

MOLAP

MOLAP uses array-based multidimensional storage engines to display multidimensional views of data. Basically, they use an OLAP cube.

Hybrid OLAP

Hybrid OLAP is a mixture of both ROLAP and MOLAP. It offers fast computation of MOLAP and higher scalability of ROLAP. HOLAP uses two databases.

1. Aggregated or computed data is stored in a multidimensional OLAP cube
2. Detailed information is stored in a relational database.

Benefits of Hybrid OLAP:

- This kind of OLAP helps to economize the disk space, and it also remains compact which helps to avoid issues related to access speed and convenience.
- Hybrid HOLAP's uses cube technology which allows faster performance for all types of data.
- ROLAP are instantly updated and HOLAP users have access to this real-time instantly updated data. MOLAP brings cleaning and conversion of data thereby improving data relevance. This brings best of both worlds.

Drawbacks of Hybrid OLAP:

- Greater complexity level**:** The major drawback in HOLAP systems is that it supports both ROLAP and MOLAP tools and applications. Thus, it is very complicated.
- Potential overlaps**:** There are higher chances of overlapping especially into their functionalities.

Advantages of OLAP

- OLAP is a platform for all type of business includes planning, budgeting, reporting, and analysis.
- Information and calculations are consistent in an OLAP cube. This is a crucial benefit.
- Quickly create and analyze "What if" scenarios
- Easily search OLAP database for broad or specific terms.
- OLAP provides the building blocks for business modeling tools, Data mining tools, performance reporting tools.
- Allows users to do slice and dice cube data all by various dimensions, measures, and filters.
- It is good for analyzing time series.
- Finding some clusters and outliers is easy with OLAP.
- It is a powerful visualization online analytical process system which provides faster response times

Disadvantages of OLAP

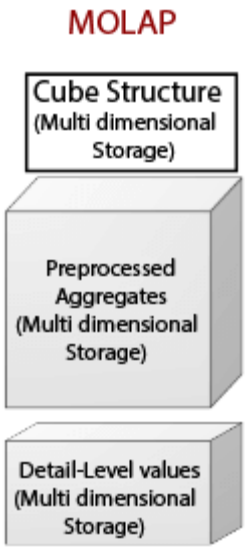
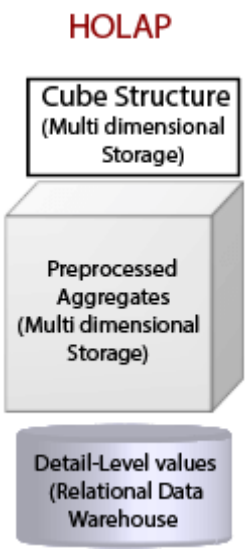
- OLAP requires organizing data into a star or snowflake schema. These schemas are complicated to implement and administer
- You cannot have large number of dimensions in a single OLAP cube
- Transactional data cannot be accessed with OLAP system.
- Any modification in an OLAP cube needs a full update of the cube. This is a time-consuming process

Summary:

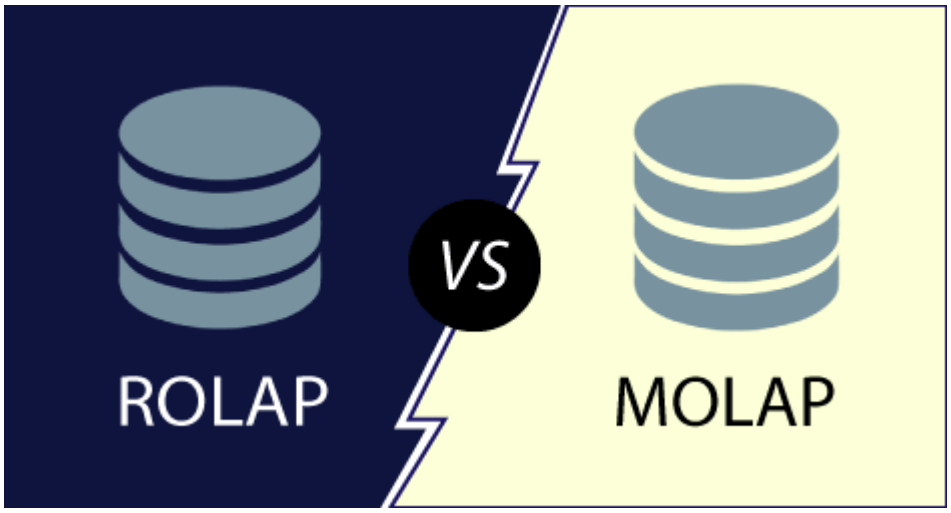
- OLAP is a technology that enables analysts to extract and view business data from different points of view.
- At the core of the OLAP concept, is an OLAP Cube.
- Various business applications and other data operations require the use of OLAP Cube.
- There are primary five types of analytical operations in OLAP 1) Roll-up 2) Drill-down 3) Slice 4) Dice and 5) Pivot
- Three types of widely used OLAP systems are MOLAP, ROLAP, and Hybrid OLAP.
- Desktop OLAP, Web OLAP, and Mobile OLAP are some other types of OLAP systems.

Difference between ROLAP, MOLAP, and HOLAP

ROLAP	MOLAP	HOLAP
ROLAP stands for Relational Online Analytical Processing.	MOLAP stands for Multidimensional Online Analytical Processing.	HOLAP stands for Hybrid Online Analytical Processing.
The ROLAP storage mode causes the aggregation of the division to be stored in indexed views in the relational database that was specified in the partition's data source.	The MOLAP storage mode principle the aggregations of the division and a copy of its source information to be saved in a multidimensional operation in analysis services when the separation is processed.	The HOLAP storage mode connects attributes of both MOLAP and ROLAP. Like MOLAP, HOLAP causes the aggregation of the division to be stored in a multidimensional operation in an SQL Server analysis services instance.
ROLAP does not because a copy of the source information to be stored in the Analysis services data folders. Instead, when the outcome cannot be derived from the query cache, the indexed views in the record source are accessed to answer queries.	This MOLAP operation is highly optimize to maximize query performance. The storage area can be on the computer where the partition is described or on another computer running Analysis services. Because a copy of the source information resides in the multidimensional operation, queries can be resolved without accessing the partition's source record.	HOLAP does not causes a copy of the source information to be stored. For queries that access the only summary record in the aggregations of a division, HOLAP is the equivalent of MOLAP.
Query response is frequently slower with ROLAP storage than with the MOLAP or HOLAP	Query response times can be reduced substantially by using aggregations. The record in the partition's	Queries that access source record for example, if we want to drill down to an atomic cube cell for which

storage mode. Processing time is also frequently slower with ROLAP.	MOLAP operation is only as current as of the most recent processing of the separation.	there is no aggregation information must retrieve data from the relational database and will not be as fast as they would be if the source information were stored in the MOLAP architecture.
		

Difference between ROLAP and MOLAP



ROLAP	MOLAP
ROLAP stands for Relational Online Analytical Processing.	MOLAP stands for Multidimensional Online Analytical Processing.
It usually used when data warehouse contains relational data.	It used when data warehouse contains relational as well as non-relational data.
It contains Analytical server.	It contains the MDDB server.
It creates a multidimensional view of data dynamically.	It contains prefabricated data cubes.
It is very easy to implement	It is difficult to implement.
It has a high response time	It has less response time due to prefabricated cubes.
It requires less amount of memory.	It requires a large amount of memory.

Data Warehouse Backup and Recovery

Security:

The objective of a data warehouse is to make large amounts of data easily accessible to the users, hence allowing the users to extract information about the business as a whole. But we know that there could be some security restrictions applied on the data that can be an obstacle for accessing the information. If the analyst has a restricted view of data, then it is impossible to capture a complete picture of the trends within the business.

The data from each analyst can be summarized and passed on to management where the different summaries can be aggregated. As the aggregations of summaries cannot be the same as that of the aggregation as a whole, it is possible to miss some information trends in the data unless someone is analyzing the data as a whole.

Security Requirements

Adding security features affect the performance of the data warehouse, therefore it is important to determine the security requirements as early as possible. It is difficult to add security features after the data warehouse has gone live.

During the design phase of the data warehouse, we should keep in mind what data sources may be added later and what would be the impact of adding those data sources. We should consider the following possibilities during the design phase.

- Whether the new data sources will require new security and/or audit restrictions to be implemented?
- Whether the new users added who have restricted access to data that is already generally available?

This situation arises when the future users and the data sources are not well known. In such a situation, we need to use the knowledge of business and the objective of data warehouse to know likely requirements.

The following activities get affected by security measures −

- User access
- Data load
- Data movement
- Query generation

User Access

We need to first classify the data and then classify the users on the basis of the data they can access. In other words, the users are classified according to the data they can access.

Data Classification

The following two approaches can be used to classify the data −

- Data can be classified according to its sensitivity. Highly-sensitive data is classified as highly restricted and less-sensitive data is classified as less restrictive.
- Data can also be classified according to the job function. This restriction allows only specific users to view particular data. Here we restrict the users to view only that part of the data in which they are interested and are responsible for.

There are some issues in the second approach. To understand, let's have an example. Suppose you are building the data warehouse for a bank. Consider that the data being stored in the data warehouse is the transaction data for all the accounts. The question here is, who is allowed to see the transaction data. The solution lies in classifying the data according to the function.

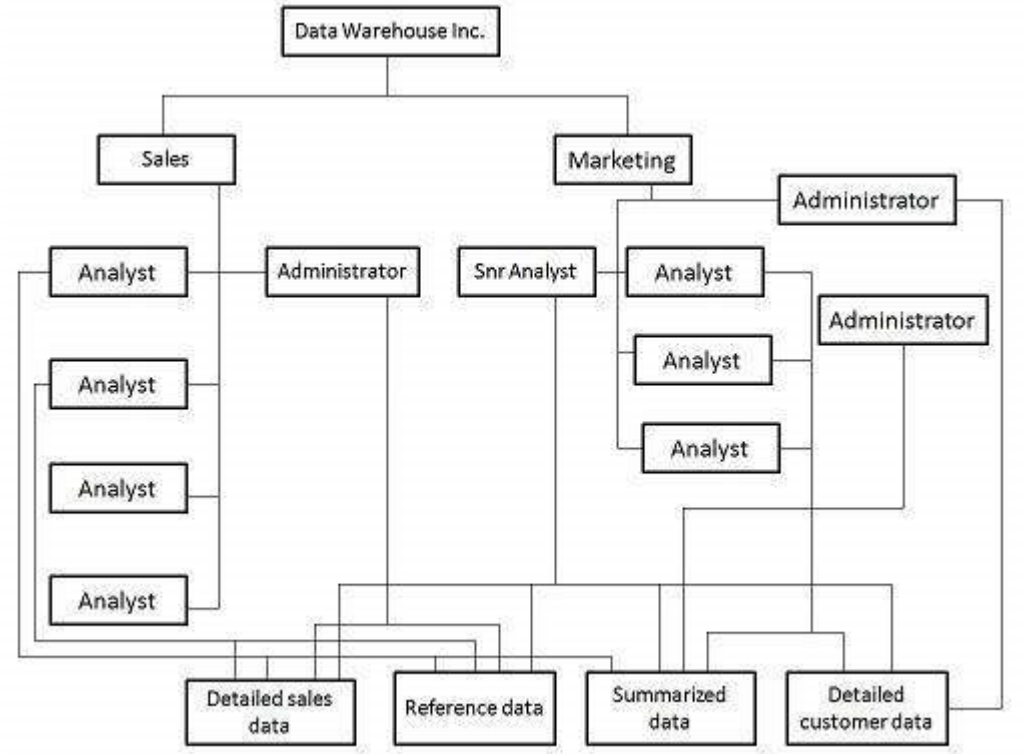
User classification

The following approaches can be used to classify the users −

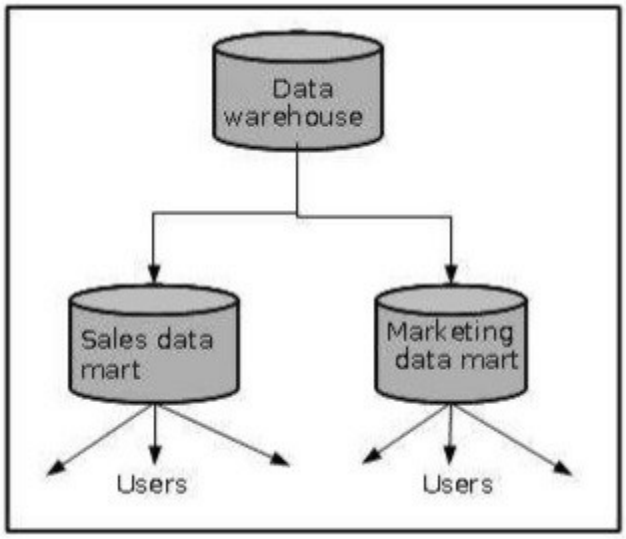
- Users can be classified as per the hierarchy of users in an organization, i.e., users can be classified by departments, sections, groups, and so on.
- Users can also be classified according to their role, with people grouped across departments based on their role.

Classification on basis of Department

Let's have an example of a data warehouse where the users are from sales and marketing department. We can have security by top-to-down company view, with access centered on the different departments. But there could be some restrictions on users at different levels. This structure is shown in the following diagram.

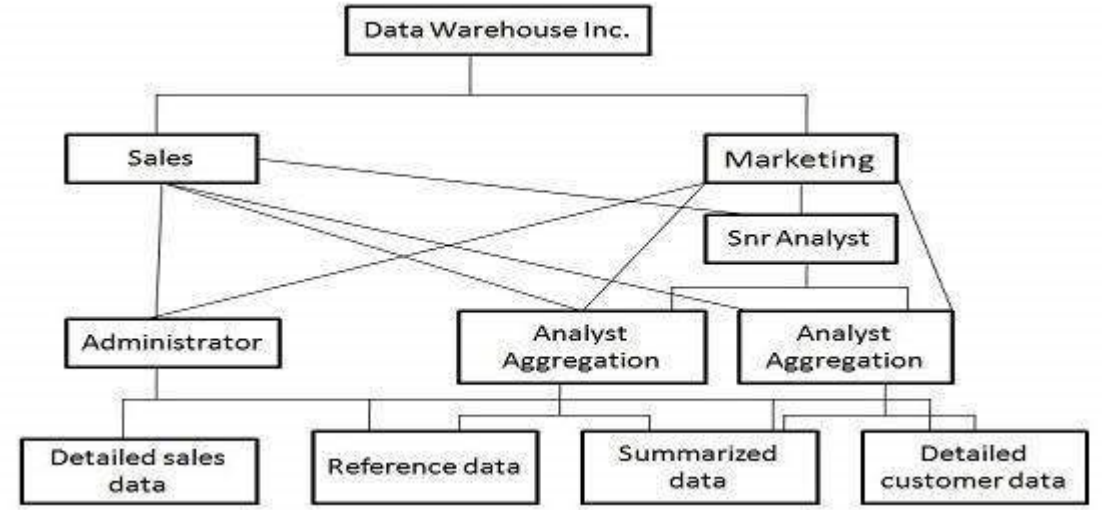


But if each department accesses different data, then we should design the security access for each department separately. This can be achieved by departmental data marts. Since these data marts are separated from the data warehouse, we can enforce separate security restrictions on each data mart. This approach is shown in the following figure.



Classification Based on Role

If the data is generally available to all the departments, then it is useful to follow the role access hierarchy. In other words, if the data is generally accessed by all the departments, then apply security restrictions as per the role of the user. The role access hierarchy is shown in the following figure.



Audit Requirements

Auditing is a subset of security, a costly activity. Auditing can cause heavy overheads on the system. To complete an audit in time, we require more hardware and therefore, it is recommended that wherever possible, auditing should be switched off. Audit requirements can be categorized as follows −

- Connections
- Disconnections
- Data access
- Data change

Note − For each of the above-mentioned categories, it is necessary to audit success, failure, or both. From the perspective of security reasons, the auditing of failures are very important. Auditing of failure is important because they can highlight unauthorized or fraudulent access.

Network Requirements

Network security is as important as other securities. We cannot ignore the network security requirement. We need to consider the following issues −

- Is it necessary to encrypt data before transferring it to the data warehouse?
- Are there restrictions on which network routes the data can take?

These restrictions need to be considered carefully. Following are the points to remember −

- The process of encryption and decryption will increase overheads. It would require more processing power and processing time.
- The cost of encryption can be high if the system is already a loaded system because the encryption is borne by the source system.

Data Movement

There exist potential security implications while moving the data. Suppose we need to transfer some restricted data as a flat file to be loaded. When the data is loaded into the data warehouse, the following questions are raised −

- Where is the flat file stored?
- Who has access to that disk space?

If we talk about the backup of these flat files, the following questions are raised −

- Do you backup encrypted or decrypted versions?
- Do these backups need to be made to special tapes that are stored separately?
- Who has access to these tapes?

Some other forms of data movement like query result sets also need to be considered. The questions raised while creating the temporary table are as follows −

- Where is that temporary table to be held?
- How do you make such table visible?

We should avoid the accidental flouting of security restrictions. If a user with access to the restricted data can generate accessible temporary tables, data can be visible to non-authorized users. We can overcome this problem by having a separate temporary area for users with access to restricted data.

Documentation

The audit and security requirements need to be properly documented. This will be treated as a part of justification. This document can contain all the information gathered from −

- Data classification
- User classification
- Network requirements
- Data movement and storage requirements
- All auditable actions

Impact of Security on Design

Security affects the application code and the development timescales. Security affects the following area −

- Application development
- Database design
- Testing

Application Development

Security affects the overall application development and it also affects the design of the important components of the data warehouse such as load manager, warehouse manager, and query manager. The load manager may require checking code to filter record and place them in different locations. More transformation rules may also be required to hide certain data. Also there may be requirements of extra metadata to handle any extra objects.

To create and maintain extra views, the warehouse manager may require extra codes to enforce security. Extra checks may have to be coded into the data warehouse to prevent it from being fooled into moving data into a location where it should not be available. The query manager requires the changes to handle any access restrictions. The query manager will need to be aware of all extra views and aggregations.

Database design

The database layout is also affected because when security measures are implemented, there is an increase in the number of views and tables. Adding security increases the size of the database and hence increases the complexity of the database design and management. It will also add complexity to the backup management and recovery plan.

Testing

Testing the data warehouse is a complex and lengthy process. Adding security to the data warehouse also affects the testing time complexity. It affects the testing in the following two ways −

- It will increase the time required for integration and system testing.
- There is added functionality to be tested which will increase the size of the testing suite.

Backup:

A data warehouse is a complex system, and it contains a huge volume of data. Therefore, it is important to back up all the data so that it becomes available for recovery in future as per requirement. In this chapter, we will discuss the issues in designing the backup strategy.

Backup Terminologies

Before proceeding further, you should know some of the backup terminologies discussed below.

- **Complete backup** − It backs up the entire database at the same time. This backup includes all the database files, control files, and journal files.
- **Partial backup** − As the name suggests, it does not create a complete backup of the database. Partial backup is very useful in large databases because they allow a strategy whereby various parts of the database are backed up in a round-robin fashion on a day-to-day basis, so that the whole database is backed up effectively once a week.
- **Cold backup** − Cold backup is taken while the database is completely shut down. In multi-instance environment, all the instances should be shut down.
- **Hot backup** − Hot backup is taken when the database engine is up and running. The requirement of hot backup varies from RDBMS to RDBMS.
- **Online backup** − It is quite similar to hot backup.

Hardware Backup

It is important to decide which hardware to use for the backup. The speed of processing the backup and restore depends on the hardware being used, how the hardware is connected, bandwidth of the network, backup software, and the speed of server's I/O system. Here we will discuss some of the hardware choices that are available and their pros and cons. These choices are as follows −

- Tape Technology
- Disk Backups

Tape Technology

The tape choice can be categorized as follows −

- Tape media
- Standalone tape drives
- Tape stackers
- Tape silos

Tape Media

There exists several varieties of tape media. Some tape media standards are listed in the table below −

Tape Media	Capacity	I/O rates
DLT	40 GB	3 MB/s
3490e	1.6 GB	3 MB/s
8 mm	14 GB	1 MB/s

Other factors that need to be considered are as follows −

- Reliability of the tape medium
- Cost of tape medium per unit
- Scalability
- Cost of upgrades to tape system
- Cost of tape medium per unit
- Shelf life of tape medium

Standalone Tape Drives

The tape drives can be connected in the following ways −

- Direct to the server
- As network available devices
- Remotely to other machine

There could be issues in connecting the tape drives to a data warehouse.

- Consider the server is a 48node MPP machine. We do not know the node to connect the tape drive and we do not know how to spread them over the server nodes to get the optimal performance with least disruption of the server and least internal I/O latency.
- Connecting the tape drive as a network available device requires the network to be up to the job of the huge data transfer rates. Make sure that sufficient bandwidth is available during the time you require it.
- Connecting the tape drives remotely also require high bandwidth.

Tape Stackers

The method of loading multiple tapes into a single tape drive is known as tape stackers. The stacker dismounts the current tape when it has finished with it and loads the next tape, hence only one tape is available at a time to be accessed. The price and the capabilities may vary, but the common ability is that they can perform unattended backups.

Tape Silos

Tape silos provide large store capacities. Tape silos can store and manage thousands of tapes. They can integrate multiple tape drives. They have the software and hardware to label and store the tapes they store. It is very common for the silo to

be connected remotely over a network or a dedicated link. We should ensure that the bandwidth of the connection is up to the job.

Disk Backups

Methods of disk backups are −

- Disk-to-disk backups
- Mirror breaking

These methods are used in the OLTP system. These methods minimize the database downtime and maximize the availability.

Disk-to-Disk Backups

Here backup is taken on the disk rather on the tape. Disk-to-disk backups are done for the following reasons −

- Speed of initial backups
- Speed of restore

Backing up the data from disk to disk is much faster than to the tape. However it is the intermediate step of backup. Later the data is backed up on the tape. The other advantage of disk-to-disk backups is that it gives you an online copy of the latest backup.

Mirror Breaking

The idea is to have disks mirrored for resilience during the working day. When backup is required, one of the mirror sets can be broken out. This technique is a variant of disk-to-disk backups.

Note − The database may need to be shutdown to guarantee consistency of the backup.

Optical Jukeboxes

Optical jukeboxes allow the data to be stored near line. This technique allows a large number of optical disks to be managed in the same way as a tape stacker or a tape silo. The drawback of this technique is that it has slow write speed than disks. But the optical media provides long-life and reliability that makes them a good choice of medium for archiving.

Software Backups

There are software tools available that help in the backup process. These software tools come as a package. These tools not only take backup, they can effectively manage and control the backup strategies. There are many software packages available in the market. Some of them are listed in the following table −

Package Name	Vendor
Networker	Legato
ADSM	IBM

Epoch	Epoch Systems
Omniback II	HP
Alexandria	Sequent

Criteria for Choosing Software Packages

The criteria for choosing the best software package are listed below −

- How scalable is the product as tape drives are added?
- Does the package have client-server option, or must it run on the database server itself?
- Will it work in cluster and MPP environments?
- What degree of parallelism is required?
- What platforms are supported by the package?
- Does the package support easy access to information about tape contents?
- Is the package database aware?
- What tape drive and tape media are supported by the package?

Tuning:

A data warehouse keeps evolving and it is unpredictable what query the user is going to post in the future. Therefore it becomes more difficult to tune a data warehouse system. In this chapter, we will discuss how to tune the different aspects of a data warehouse such as performance, data load, queries, etc.

Difficulties in Data Warehouse Tuning

Tuning a data warehouse is a difficult procedure due to following reasons −

- Data warehouse is dynamic; it never remains constant.
- It is very difficult to predict what query the user is going to post in the future.
- Business requirements change with time.
- Users and their profiles keep changing.
- The user can switch from one group to another.
- The data load on the warehouse also changes with time.

Note − It is very important to have a complete knowledge of data warehouse.

Performance Assessment

Here is a list of objective measures of performance −

- Average query response time
- Scan rates
- Time used per day query
- Memory usage per process
- I/O throughput rates

Following are the points to remember.

- It is necessary to specify the measures in service level agreement (SLA).
- It is of no use trying to tune response time, if they are already better than those required.
- It is essential to have realistic expectations while making performance assessment.
- It is also essential that the users have feasible expectations.

- To hide the complexity of the system from the user, aggregations and views should be used.
- It is also possible that the user can write a query you had not tuned for.

Data Load Tuning

Data load is a critical part of overnight processing. Nothing else can run until data load is complete. This is the entry point into the system.

Note − If there is a delay in transferring the data, or in arrival of data then the entire system is affected badly. Therefore it is very important to tune the data load first.

There are various approaches of tuning data load that are discussed below −

- The very common approach is to insert data using the **SQL Layer**. In this approach, normal checks and constraints need to be performed. When the data is inserted into the table, the code will run to check for enough space to insert the data. If sufficient space is not available, then more space may have to be allocated to these tables. These checks take time to perform and are costly to CPU.
- The second approach is to bypass all these checks and constraints and place the data directly into the preformatted blocks. These blocks are later written to the database. It is faster than the first approach, but it can work only with whole blocks of data. This can lead to some space wastage.
- The third approach is that while loading the data into the table that already contains the table, we can maintain indexes.
- The fourth approach says that to load the data in tables that already contain data, **drop the indexes & recreate them** when the data load is complete. The choice between the third and the fourth approach depends on how much data is already loaded and how many indexes need to be rebuilt.

Integrity Checks

Integrity checking highly affects the performance of the load. Following are the points to remember −

- Integrity checks need to be limited because they require heavy processing power.
- Integrity checks should be applied on the source system to avoid performance degrade of data load.

Tuning Queries

We have two kinds of queries in data warehouse −

- Fixed queries
- Ad hoc queries

Fixed Queries

Fixed queries are well defined. Following are the examples of fixed queries −

- regular reports
- Canned queries
- Common aggregations

Tuning the fixed queries in a data warehouse is same as in a relational database system. The only difference is that the amount of data to be queried may be different. It is good to store the most successful execution plan while testing fixed queries. Storing these executing plan will allow us to spot changing data size and data skew, as it will cause the execution plan to change.

Note − We cannot do more on fact table but while dealing with dimension tables or the aggregations, the usual collection of SQL tweaking, storage mechanism, and access methods can be used to tune these queries.

Ad hoc Queries

To understand ad hoc queries, it is important to know the ad hoc users of the data warehouse. For each user or group of users, you need to know the following −

- The number of users in the group
- Whether they use ad hoc queries at regular intervals of time
- Whether they use ad hoc queries frequently
- Whether they use ad hoc queries occasionally at unknown intervals.
- The maximum size of query they tend to run
- The average size of query they tend to run
- Whether they require drill-down access to the base data
- The elapsed login time per day
- The peak time of daily usage
- The number of queries they run per peak hour

Points to Note

- It is important to track the user's profiles and identify the queries that are run on a regular basis.
- It is also important that the tuning performed does not affect the performance.
- Identify similar and ad hoc queries that are frequently run.
- If these queries are identified, then the database will change and new indexes can be added for those queries.
- If these queries are identified, then new aggregations can be created specifically for those queries that would result in their efficient execution.

Testing:

Testing is very important for data warehouse systems to make them work correctly and efficiently. There are three basic levels of testing performed on a data warehouse −

- Unit testing
- Integration testing
- System testing

Unit Testing

- In unit testing, each component is separately tested.
- Each module, i.e., procedure, program, SQL Script, Unix shell is tested.
- This test is performed by the developer.

Integration Testing

- In integration testing, the various modules of the application are brought together and then tested against the number of inputs.
- It is performed to test whether the various components do well after integration.

System Testing

- In system testing, the whole data warehouse application is tested together.
- The purpose of system testing is to check whether the entire system works correctly together or not.
- System testing is performed by the testing team.

- Since the size of the whole data warehouse is very large, it is usually possible to perform minimal system testing before the test plan can be enacted.

Test Schedule

First of all, the test schedule is created in the process of developing the test plan. In this schedule, we predict the estimated time required for the testing of the entire data warehouse system.

There are different methodologies available to create a test schedule, but none of them are perfect because the data warehouse is very complex and large. Also the data warehouse system is evolving in nature. One may face the following issues while creating a test schedule −

- A simple problem may have a large size of query that can take a day or more to complete, i.e., the query does not complete in a desired time scale.
- There may be hardware failures such as losing a disk or human errors such as accidentally deleting a table or overwriting a large table.

Note − Due to the above-mentioned difficulties, it is recommended to always double the amount of time you would normally allow for testing.

Testing Backup Recovery

Testing the backup recovery strategy is extremely important. Here is the list of scenarios for which this testing is needed −

- Media failure
- Loss or damage of table space or data file
- Loss or damage of redo log file
- Loss or damage of control file
- Instance failure
- Loss or damage of archive file
- Loss or damage of table
- Failure during data failure

Testing Operational Environment

There are a number of aspects that need to be tested. These aspects are listed below.

- **Security** − A separate security document is required for security testing. This document contains a list of disallowed operations and devising tests for each.
- **Scheduler** − Scheduling software is required to control the daily operations of a data warehouse. It needs to be tested during system testing. The scheduling software requires an interface with the data warehouse, which will need the scheduler to control overnight processing and the management of aggregations.
- **Disk Configuration.** − Disk configuration also needs to be tested to identify I/O bottlenecks. The test should be performed with multiple times with different settings.
- **Management Tools.** − It is required to test all the management tools during system testing. Here is the list of tools that need to be tested.
    - Event manager
    - System manager
    - Database manager
    - Configuration manager
    - Backup recovery manager

Testing the Database

The database is tested in the following three ways −

- **Testing the database manager and monitoring tools** − To test the database manager and the monitoring tools, they should be used in the creation, running, and management of test database.
- **Testing database features** − Here is the list of features that we have to test −
    - Querying in parallel
    - Create index in parallel
    - Data load in parallel
- **Testing database performance** − Query execution plays a very important role in data warehouse performance measures. There are sets of fixed queries that need to be run regularly and they should be tested. To test ad hoc queries, one should go through the user requirement document and understand the business completely. Take time to test the most awkward queries that the business is likely to ask against different index and aggregation strategies.

Testing the Application

- All the managers should be integrated correctly and work in order to ensure that the end-to-end load, index, aggregate and queries work as per the expectations.
- Each function of each manager should work correctly
- It is also necessary to test the application over a period of time.
- Week end and month-end tasks should also be tested.

Logistic of the Test

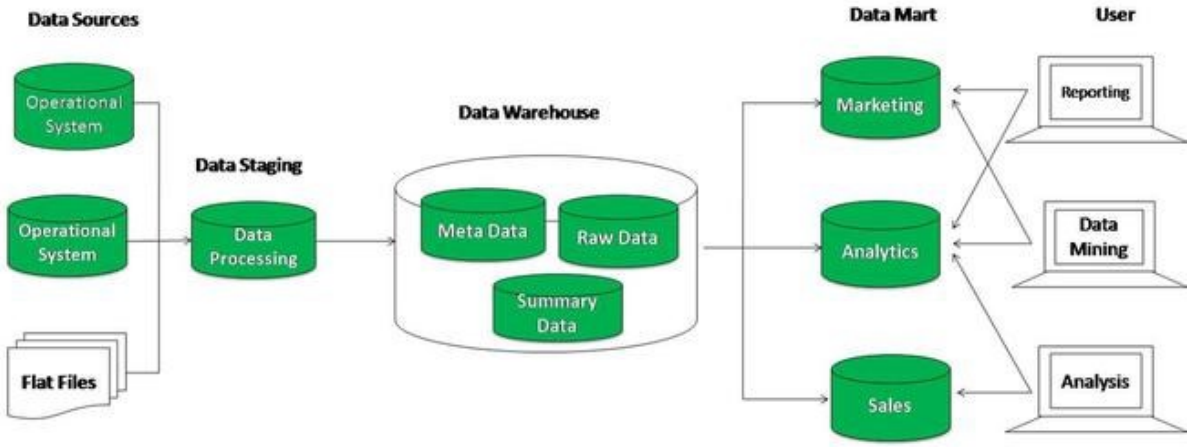
The aim of system test is to test all of the following areas −

- Scheduling software
- Day-to-day operational procedures
- Backup recovery strategy
- Management and scheduling tools
- Overnight processing
- Query performance

Note − The most important point is to test the scalability. Failure to do so will leave us a system design that does not work when the system grows.

Data Warehouse Tools and technologies

A data warehouse is a Data management system that is used for storing, reporting, and data analysis. It is the primary component of business intelligence and is also known as an enterprise data warehouse. Data Warehouses are central repositories that store data from one or more heterogeneous sources. Data warehouses are analytical tools built to support decision-making for reporting users across many departments. Data warehouse works to create a single, unified system of truth for an entire organization and store historical data about business and organization so that it could be analysed and extract insights from it.



Data Flow through Warehouse Architecture

Previously, organizations had to build lots of infrastructure for data warehousing but today, cloud computing technology has amazingly reduced the efforts as well as the cost of building data warehousing for businesses. Data warehouses and their tools are moving from physical data centres to cloud-based data warehouses. Many large organizations still operate data through the traditional way of data warehousing but clearly, the future of the data warehouse is in the cloud. The cloud-based data warehousing tools are fast, efficient, highly scalable, and available based on pay-per-use.

There are various cloud-based Data Warehousing tools available. So, it becomes difficult to select top Data Warehouse tools according to our project requirements. Following are the top 8 Data Warehousing tools:

1. Amazon Redshift:

Amazon Redshift is a cloud-based fully managed petabytes-scale data warehouse By the Amazon Company. It starts with just a few hundred gigabytes of data and scales to petabytes or more. This enables the use of data to accumulate new insights for businesses and customers. It is a relational database management system (RDBMS) therefore it is compatible with other RDBMS applications. Amazon Redshift offers quick querying capabilities over structured data by the use of SQL-based clients and business intelligence (BI) tools using standard ODBC and JDBC connections. Amazon Redshift is made around industry-standard SQL, with additional practicality to manage massive datasets and support superior analysis and reporting of these data. It helps to work quickly and easily along with data in open formats, and simply integrates with and connects to the AWS scheme. Also query and export data to and from the data lake. No alternative cloud data warehouse tool makes it straightforward to query data and writes data back to the data lake in open formats. It focuses

on simple Use and Accessibility. MySQL and alternative SQL-based systems are one in all the foremost well-liked and simply usable interfaces for database management. Redshift's easy query-based system makes platform adoption and acclimatization a light breeze. It is incredibly quick once it involves loading data and querying it for analytical and reporting functions. Redshift features a massively parallel processing (MPP) design that permits loading data at a very high speed.

2. Microsoft Azure:

Azure is a cloud computing platform that was launched by Microsoft in 2010. Microsoft Azure is a cloud computing service provider for building, testing, deploying, and managing applications and services through Microsoft-managed data centres. Azure is a public cloud computing platform that offers Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). The Azure cloud platform provides more than 200 products and cloud services such as Data Analytics, Virtual Computing, Storage, Virtual Network, Internet Traffic Manager, Web Sites, Media Services, Mobile Services, Integration, etc. Azure facilitates simple portability and genuinely compatible platform between on-premise and public Cloud. Azure provides a range of cross-connections including virtual private networks (VPNs), caches, content delivery networks (CDNs), and ExpressRoute connections to improve usability and performance. Microsoft Azure provides a secure base across physical infrastructure and operational security. Azure App offers a completely managed web hosting service that helps in building web applications, services, and Restful APIs. It offers a variety of plans to meet the requirements of any application, from small to globally scaled web applications. Running virtual machines or containers in the cloud is one of the most popular applications of Microsoft Azure.

3. Google BigQuery:

BigQuery is a serverless data warehouse that allows scalable analysis over petabytes of data. It's a Platform as a Service that supports querying with the help of ANSI SQL. It additionally has inbuilt machine learning capabilities. BigQuery was declared in 2010 and made available for use there in 2011. Google BigQuery is a cloud-based big data analytics web service to process very huge amount of read-only data sets. BigQuery is designed for analyzing data that are in billions of rows by simply employing SQL-lite syntax. BigQuery can run advanced analytical SQL-based queries beneath big sets of data. BigQuery is not developed to substitute relational databases and for easy CRUD operations and queries. It is oriented for running analytical queries. It is a hybrid system that enables the storage of information in columns; however, it takes into the NoSQL additional features, like the data type, and the nested feature. BigQuery is a better option than Redshift since we have to pay by the hour. BigQuery may also be the best solution for data scientists running ML or data mining operations since they deal with extremely large datasets. Google Cloud also offers a set of auto-scaling services that enables you to build a data lake that integrates with your existing applications, skills, and IT investments. In BigQuery, most of the time is spent on metadata/initiation, but the actual execution time is very small.

4. Snowflake:

Snowflake is a cloud computing-based data warehousing built on top of the Amazon Web Services or Microsoft Azure cloud infrastructure. The Snowflake design allows storage and computes to scale independently, thus customers can use and pay money for storage and computation individually. In Snowflake data processing is simplified: Users will do data blending, analysis, and transformations against varied forms of data structures with one

language, SQL. Snowflake offers dynamic, scalable computing power with charges primarily based strictly on usage. With Snowflake, computation and storage are fully separate, and also the storage value is that the same as storing the data on Amazon S3. AWS tried to handle this issue by introducing Redshift Spectrum, which allows querying data that exists directly on Amazon S3; however, it's not as seamless as Snowflake. With Snowflake, we can clone a table, a schema, or perhaps a database in no time and occupying no extra space. This is often because the cloned table creates pointers that point to the kept data, however, not the actual data. In alternative words, the cloned table solely has data that's completely different from its original table.

5. Micro Focus Vertica:

Micro Focus Vertica: Micro Focus Vertica is developed to use in data warehouses and other big data workloads where speed, scalability, simplicity, and openness are crucial to the success of analytics. It is a self-monitored MPP database and offers scalability and flexibility that other tools don't. It is used on commercial hardware; therefore we can scale the database as required. It is designed significantly in-database advanced analytics capabilities to improve query performance over traditional relational database systems and unverified open-source offerings. For example, Vertica is a column-oriented relational database; therefore, it might not qualify as a NoSQL database. A NoSQL database is best outlined as being a non-relational, shared-nothing, horizontally scalable database while not ACID guarantees. Vertica differs from normal RDBMS within the approach that it stores data by grouping data at once on disk by column instead of by row, Vertica reads the columns documented by the query, rather than scanning the complete table as row-oriented databases should do. Vertica offers the foremost advanced unified analytical warehouse that allows the organization to stay up with the dimensions and complexness of huge amounts of data volumes. With Vertica, businesses can perform tasks like predictive maintenance, client remembrance, economic compliance, and network optimization, and far more.

6. Amazon DynamoDB:

Amazon DynamoDB is a fully managed proprietary NoSQL data warehouse service that supports key-value and document data structures and is obtainable by Amazon.com as a part of the Amazon Web Services portfolio. DynamoDB has an identical data model and encompasses a completely different underlying implementation. A partition key value is used in DynamoDB as input to an enclosed hash function. The output from the hash function determines the partition within which the item is going to be kept. All items with identical partition key values are stored together, in sorted order by sort key value. It offers customers high availability, dependability, and progressive scalability, with no limits on dataset size or request output for a given table. DynamoDB is meant for OLTP use cases high-speed data access wherever you are operative on many records at a time. However, users even have a desire for OLAP access patterns massive, analytical queries over the complete dataset to search out common things, or a variety of orders by day, or different insights. DynamoDB is aligned with the values of Serverless applications: automatic scaling consistent with your application load, pay-per-what-you-use rating, simple to induce started with, and no servers to manage. This makes DynamoDB an awfully common selection for Serverless applications running in AWS.

7. PostgreSQL:

It is an extremely stable database management system, backed by over twenty years of community development that has contributed to its high levels of resilience, integrity, and

correctness. PostgreSQL is employed because the primary data store or data warehouse for several web, mobile, geospatial, and analytics applications. SQL Server is a database management system that is especially used for e-commerce and providing different data warehousing solutions. PostgreSQL is a sophisticated version of SQL that provides support to various functions of SQL like foreign keys, subqueries, triggers, and other user-defined varieties and functions. Postgres is a feature-rich database that can handle advanced complicated queries and big databases. MySQL is a less complicated database that is comparatively simple to line up and manage, fast, reliable, and well-understood. PostgreSQL performs well in OLTP/OLAP systems once read/write speeds are needed and intensive data analysis is required. PostgreSQL additionally works well with Business Intelligence applications however is best suited to data warehousing and data analysis applications that require quick read/write operations speed.

8. Amazon S3:

Amazon S3 is object storage engineered to store and retrieves any quantity of data from any place. It is an easy storage service that provides business-leading sturdiness, accessibility, performance, security, and nearly unlimited scalability at very low prices. AWS S3 is a key-value store, one of the foremost classes of NoSQL databases used for accumulating voluminous, mutating, unstructured, or semi-structured data. Features like metadata support, prefixes, and object tags enable users to arrange data consistent with their desires. The S3 object storage cloud service offers subscriber access to similar systems that Amazon uses to run its own websites. Amazon S3 is object storage capable of storing massive objects, up to 5TB in size. S3 allows customers to access, store and download practically any file or object that's up 5 TB in size with the biggest single upload capped at 5 gigabytes (GB). S3 is often used for storing pictures, videos, logs, and alternative varieties of files. There's no limit on the number of objects that may be stored in an S3 bucket. Every object in S3 includes a URL that might be used to download the object. S3 provides unlimited storage at a comparatively low cost than DynamoDB; however, scan operations are abundant slower than DynamoDB, although it can perform HTTP queries for the same. Amazon S3 sets the quality once it involves business cloud storage whereas simple use is not a part of that standard but, top-quality security, extreme flexibility, and total integration are.

9. Teradata:

Teradata is one of the admired Relational Database Management systems. It is appropriate for building big data warehousing applications. Teradata accomplishes this with the help of parallelism. Teradata database system is built on Massively Parallel Processing (MPP) architecture. The Teradata system primarily splits the work among its processes and runs them in parallel to reduce workload and also makes sure that the task is accomplished quickly and successfully. Teradata provides real-time, intelligent answers by processing 100% of the appropriate data, despite the volume of the query. Teradata fulfils all the requirements in terms of Integration or ETL with the capabilities of consuming, analysing and managing the data. Data in an exceeding data warehouse is organized to support analysis instead of processing real-time transactions as in online transaction processing systems (OLTP). Although it is geared towards OLAP. It's one of the most powerful data integration and analytics database solutions within the market. Teradata is employed or has been utilized in past by most business enterprises. It processes enormous amounts of data very easily. It's simple to navigate and a sensible graphical user interface helps Business users use it with

basic training and query knowledge, however, big data processing is a challenge because of its existing architectures.

10. Amazon RDS:

Amazon Relational Database Service is a cloud data storage service to operate and scale a relational database within the AWS Cloud. Its cost-effective and resizable hardware capability helps us to build an industry-standard relational database and manages all usual database administration tasks. Amazon RDS is a PaaS because it solely provides a platform or a group of tools to manage your database instances. AWS is IaaS; however, the RDS provided by AWS is PaaS. Amazon RDS can manage complicated and long tasks like software installation and upgrades, storage management, replication for high availability and backups for disaster recovery. We can also deploy scalable MySQL servers as per requirement in minutes with cost-effective and resizable hardware capability with the help of Amazon RDS. Amazon RDS has 3 instance classes: Standard, Memory Optimized, and Burstable Performance. These instance classes are comprised of variable combos of C.P.U., memory, storage, and networking capability and provide you with the flexibility to decide on the acceptable mix of resources for the database. In Amazon, RDS supports there are six database engines we can choose from, they are Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle info, and SQL Server.

11. IBM Db2 Warehouse:

IBM Db2 Warehouse is an elastic cloud data warehouse that provides independent scaling of data storage and computation. IBM Db2 is a data management product from IBM including Db2 relational database. It is meant to store, analyse and retrieve the data with efficiency. Extremely optimized columns data storage and in-memory process facilitate boost analytics and machine learning burden. IBM Db2 is a well-developed, completely managed Cloud SQL Database-as-a-Service solution alongside Db2 and Oracle PL/SQL compatibility. It is a Relational Database Management System (RDBMS) designed to store, analyse and retrieve the data with efficiency and is highly robust and powerful. Its Data migration processes and therefore the user interface (UI) are clean, intuitive, and simple to work for users of a variety of skill levels. IBM Db2 is now, turning into the AI database that may facilitate in power today's cognitive applications, assist to modernize AI development, and enabling management of each structured and unstructured data across the physical platform and multi-cloud environments.

12. Oracle Autonomous Warehouse:

Autonomous Data Warehouse is a cloud-based data warehousing service provided by Oracle that removes all the complexities of constructing a data warehouse, data security, and helps in developing data-driven applications. It automates the process of configuring, securing, regulating, scaling, and backing up the data in the data warehouse. It provides a new, comprehensive cloud experience for data storage that's simple, fast, and elastic. An autonomous data Warehouse is that a complete resolution that uses a converged database providing constitutional support for multi-model data and multiple workloads. It includes inbuilt self-service tools to enhance the productivity of analysts, data scientists, and developers. It independently encrypts information at rest and in motion, protects regulated information; put-ups required security reinforcements and detects threats. Additionally, customers can simply use Oracle data Safe to perform user and privilege analysis, sensitive data discovery and protection, and activity auditing. An autonomous data Warehouse makes it simple to stay data safe from outsiders and insiders. Uniquely, it's additionally ready to

incessantly alter performance standardization and auto-scaling, with no outage time, human interference. This reduces administration effort by more than 80% and allows business groups to work without facilitation from IT.

13. MariaDB:

MariaDB Server is one of the most well-liked ASCII text file relational databases. It's created by the initial developers of MySQL and absolute to keep the open-source. MariaDB includes a good choice of storage engines, as well as superior storage engines, for operating with alternative RDBMS data sources. MariaDB uses a regular and well-liked querying language. MariaDB runs on many operative systems and supports a good style of programming languages. A bit like MySQL, MariaDB conjointly uses a client/server model with a server program that files requests from client programs. As is typical of client/server computer systems, the server and therefore the client programs will be on completely different hosts. MariaDB shows an improved speed when put next to MySQL. MySQL exhibits a slower speed when put next to MariaDB. With the Memory storage engine of MariaDB, any data operating statement will be executed faster than the standard MySQL storage engine. The memory storage engine of MySQL is slower than the storage engine of MariaDB and it also supports plenty of commands along with interfaces that are more accessible to NoSQL than to SQL.

14. MarkLogic:

MarkLogic is a multi-model NoSQL database that has evolved from its XML database roots to additionally natively store JSON documents and RDF triples for its linguistics data model. It uses a distributed design that may handle many billions of documents and many terabytes of knowledge. Associate in Nursing Operational information Warehouse designed on the MarkLogic Enterprise NoSQL platform not solely improves upon the normal capabilities related to associate in Nursing ODW e.g. ingesting giant amounts of knowledge and creating it out there for question in real-time, however, additionally makes this capability out there for a wider form of data. MarkLogic provides an extremely differentiated product and provides the flexibleness for clients to vary cloud suppliers later if necessary. The planning philosophy behind the evolution of MarkLogic is that storing information is merely a part of the answer. It uses XML and JSON documents in the form of the data models and stores these documents within a transactional repository. It indexes the words and values from every one of the loaded documents, likewise because of the document structure. MarkLogic Data Hub is a set of tools that assist quickly in build an operational information hub on the MarkLogic Server. The operational data hub pattern may be a method of building information hubs that facilitate quick and a lot of agile information integration, whereas permitting period simultaneous interactive access to information.

15. Cloudera:

Cloudera Data Warehousing Platform is that the industry's 1st enterprise data cloud i.e. multi-functional analytics based on a platform that eliminates silos and speeds up the invention of data-driven insights. It applies consistent security, governance, and metadata in shared data cases. Cloudera's trendy Data Warehouse powers superior bismuth and data deposit in each on-premises deployment and as a cloud service. Business users will explore and operate on information quickly, run new reports and workloads, or access interactive dashboards while not help from the IT department. Additionally, IT will eliminate the inefficiencies of data silos by consolidating data marts into a climbable analytics platform to raised meet business desires. With its open design, information is accessed by additional

users with additional tools, together with data scientists and engineers, providing additional worth at a lower price. Solely Cloudera also offers a modern enterprise platform, tools, and skills that help us to unlock business understanding with machine learning and AI. Cloudera's trendy platform for machine learning and analytics, optimized for the cloud, enables us to build and deploy AI solutions at scale, with efficiency and firmly, anyplace we would like. Cloudera quick Forward Labs skilled guidance helps you notice your AI future, faster.

Data Warehouse - ETL & Reporting Tools

An ETL tool extracts the data from all these heterogeneous data sources, transforms the data (like applying calculations, joining fields, keys, removing incorrect data fields, etc.), and loads it into a Data Warehouse.

Extraction

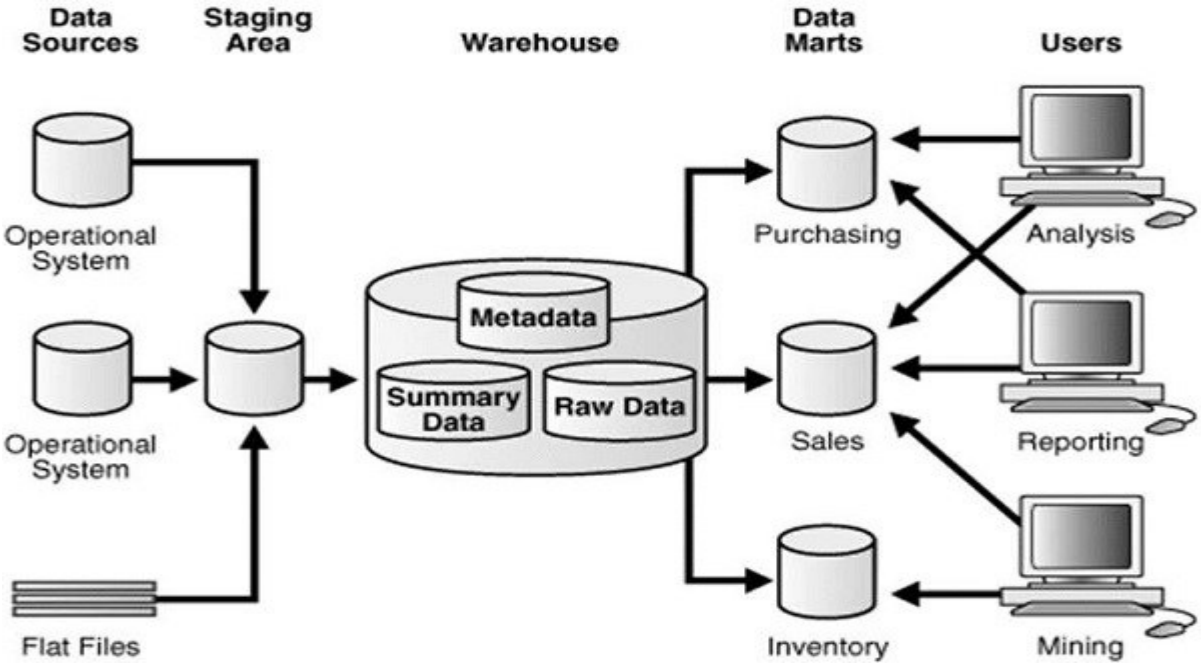
A staging area is required during the ETL load. There are various reasons why staging area is required. The source systems are only available for specific period of time to extract data. This period of time is less than the total data-load time. Therefore, staging area allows you to extract the data from the source system and keeps it in the staging area before the time slot ends.

The staging area is required when you want to get the data from multiple data sources together or if you want to join two or more systems together.

For example − You will not be able to perform an SQL Query joining two tables from two physically different databases.

The data extractions' time slot for different systems vary as per the time zone and operational hours. The data extracted from the source systems can be used in multiple Data Warehouse Systems, Operation Data Stores, etc.

ETL allows you to perform complex transformations and requires extra area to store the data.



Transform

In data transformation, you apply a set of functions on extracted data to load it into the target system. The data that does not require any transformation is known as a direct move or pass-through data.

You can apply different transformations on extracted data from the source system. For example, you can perform customized calculations. If you want sum-of-sales revenue and this is not in database, you can apply the SUM formula during transformation and load the data.
For example − If you have the first name and the last name in a table in different columns, you can use concatenate before loading.

Load

During the Load phase, data is loaded into the end-target system, and it can be a flat file or a Data Warehouse system.

BI Reporting Tool

BI (Business Intelligence) tools are used by business users to create basic, medium, and complex reports from the transactional data in data warehouse and by creating Universes using the **Information Design Tool/UDT**. Various SAP and non-SAP data sources can be used to create reports.

There are quite a few BI Reporting, Dashboard and Data Visualization Tools available in the market. Some of which are as follows −

- SAP Business Objects Web Intelligence (WebI)
- Crystal Reports
- SAP Lumira
- Dashboard Designer
- IBM Cognos
- Microsoft BI Platform
- Tableau Business Intelligence
- JasperSoft
- Oracle BI OBIEE
- Pentaho
- QlickView
- SAP BW
- SAS Business Intelligence
- Necto
- Tibco Spotfire

What is Data Mart?

A **Data Mart** is focused on a single functional area of an organization and contains a subset of data stored in a Data Warehouse. A Data Mart is a condensed version of Data Warehouse and is designed for use by a specific department, unit or set of users in an organization. E.g., Marketing, Sales, HR or finance. It is often controlled by a single department in an organization.

Data Mart usually draws data from only a few sources compared to a Data warehouse. Data marts are small in size and are more flexible compared to a Datawarehouse.

Why do we need Data Mart?

- Data Mart helps to enhance user's response time due to reduction in volume of data
- It provides easy access to frequently requested data.
- Data mart are simpler to implement when compared to corporate Datawarehouse. At the same time, the cost of implementing Data Mart is certainly lower compared with implementing a full data warehouse.

- Compared to Data Warehouse, a datamart is agile. In case of change in model, datamart can be built quicker due to a smaller size.
- A Datamart is defined by a single Subject Matter Expert. On the contrary data warehouse is defined by interdisciplinary SME from a variety of domains. Hence, Data mart is more open to change compared to Datawarehouse.
- Data is partitioned and allows very granular access control privileges.
- Data can be segmented and stored on different hardware/software platforms.

Types of Data Mart

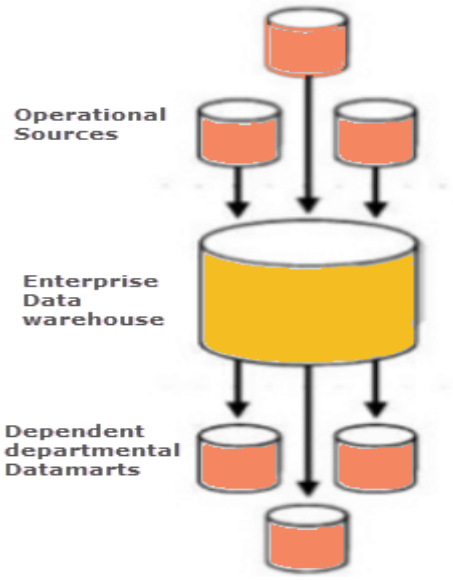
There are three main types of data mart:

1. **Dependent**: Dependent data marts are created by drawing data directly from operational, external or both sources.
2. **Independent**: Independent data mart is created without the use of a central data warehouse.
3. **Hybrid**: This type of data marts can take data from data warehouses or operational systems.

Dependent Data Mart

A dependent data mart allows sourcing organization's data from a single Data Warehouse. It is one of the data mart example which offers the benefit of centralization. If you need to develop one or more physical data marts, then you need to configure them as dependent data marts.

Dependent Data Mart in data warehouse can be built in two different ways. Either where a user can access both the data mart and data warehouse, depending on need, or where access is limited only to the data mart. The second approach is not optimal as it produces sometimes referred to as a data junkyard. In the data junkyard, all data begins with a common source, but they are scrapped, and mostly junked.

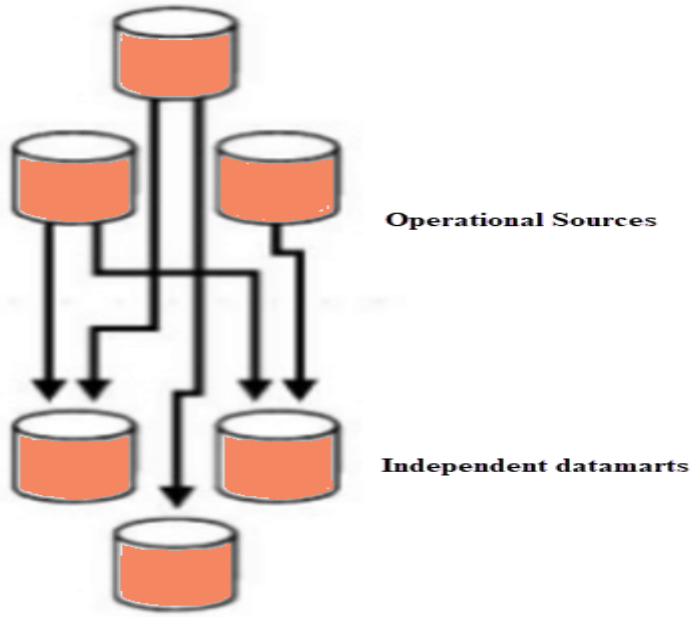


Dependent Data Mart

Independent Data Mart

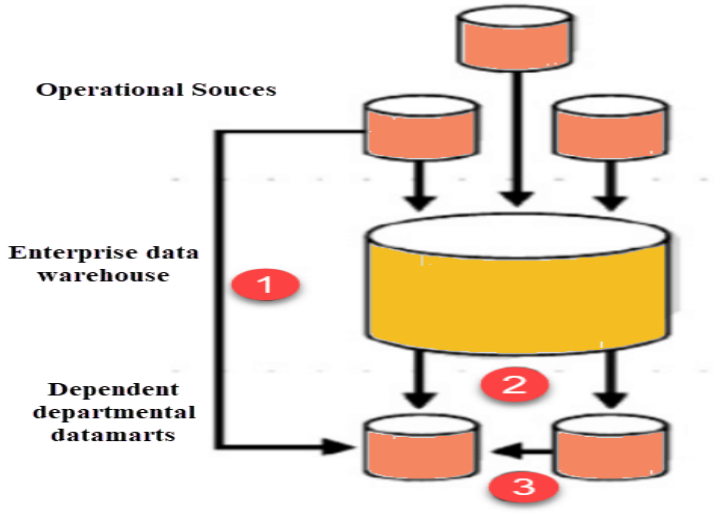
An independent data mart is created without the use of central Data warehouse. This kind of Data Mart is an ideal option for smaller groups within an organization.

An independent data mart has neither a relationship with the enterprise data warehouse nor with any other data mart. In Independent data mart, the data is input separately, and its analyses are also performed autonomously.
Implementation of independent data marts is antithetical to the motivation for building a data warehouse. First of all, you need a consistent, centralized store of enterprise data which can be analysed by multiple users with different interests who want widely varying information.



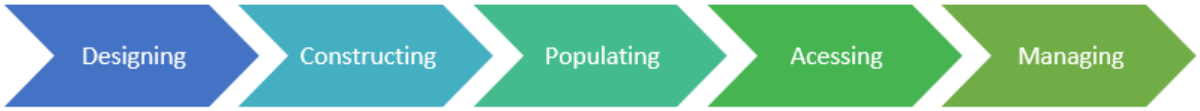
Independent Data Mart

Hybrid Data Mart:
A hybrid data mart combines input from sources apart from Data warehouse. This could be helpful when you want ad-hoc integration, like after a new group or product is added to the organization.
It is the best data mart example suited for multiple database environments and fast implementation turnaround for any organization. It also requires least data cleansing effort. Hybrid Data mart also supports large storage structures, and it is best suited for flexible for smaller data-centric applications.



Hybrid Data Mart

Steps in Implementing a Datamart



Implementing a Data Mart is a rewarding but complex procedure. Here are the detailed steps to implement a Data Mart:

Designing

Designing is the first phase of Data Mart implementation. It covers all the tasks between initiating the request for a data mart to gathering information about the requirements. Finally, we create the logical and physical Data Mart design.

The design step involves the following tasks:

- Gathering the business & technical requirements and Identifying data sources.
- Selecting the appropriate subset of data.
- Designing the logical and physical structure of the data mart.

Data could be partitioned based on following criteria:

- Date
- Business or Functional Unit
- Geography
- Any combination of above

Data could be partitioned at the application or DBMS level. Though it is recommended to partition at the Application level as it allows different data models each year with the change in business environment.

What Products and Technologies Do You Need?

A simple pen and paper would suffice. Though tools that help you create UML or ER diagram would also append meta data into your logical and physical designs.

Constructing

This is the second phase of implementation. It involves creating the physical database and the logical structures.

This step involves the following tasks:

- Implementing the physical database designed in the earlier phase. For instance, database schema objects like table, indexes, views, etc. are created.

What Products and Technologies Do You Need?

You need a relational database management system to construct a data mart. RDBMS have several features that are required for the success of a Data Mart.

- **Storage management:** An RDBMS stores and manages the data to create, add, and delete data.
- **Fast data access:** With a SQL query you can easily access data based on certain conditions/filters.
- **Data protection:** The RDBMS system also offers a way to recover from system failures such as power failures. It also allows restoring data from these backups in case of the disk fails.
- **Multiuser support:** The data management system offers concurrent access, the ability for multiple users to access and modify data without interfering or overwriting changes made by another user.
- **Security:** The RDMS system also provides a way to regulate access by users to objects and certain types of operations.

Populating:

In the third phase, data in populated in the data mart.

The populating step involves the following tasks:

- Source data to target data Mapping
- Extraction of source data
- Cleaning and transformation operations on the data
- Loading data into the data mart
- Creating and storing metadata

What Products and Technologies Do You Need?

You accomplish these population tasks using an ETL (Extract Transform Load) Tool. This tool allows you to look at the data sources, perform source-to-target mapping, extract the data, transform, cleanse it, and load it back into the data mart.

In the process, the tool also creates some metadata relating to things like where the data came from, how recent it is, what type of changes were made to the data, and what level of summarization was done.

Accessing

Accessing is a fourth step which involves putting the data to use: querying the data, creating reports, charts, and publishing them. End-users submit queries to the database and display the results of the queries

The accessing step needs to perform the following tasks:

- Set up a meta layer that translates database structures and objects names into business terms. This helps non-technical users to access the Data mart easily.
- Set up and maintain database structures.
- Set up API and interfaces if required

What Products and Technologies Do You Need?

You can access the data mart using the command line or GUI. GUI is preferred as it can easily generate graphs and is user-friendly compared to the command line.

Managing

This is the last step of Data Mart Implementation process. This step covers management tasks such as-

- Ongoing user access management.
- System optimizations and fine-tuning to achieve the enhanced performance.
- Adding and managing fresh data into the data mart.
- Planning recovery scenarios and ensure system availability in the case when the system fails.

What Products and Technologies Do You Need?

You could use the GUI or command line for data mart management.

Best practices for Implementing Data Marts

Following are the best practices that you need to follow while in the Data Mart Implementation process:

- The source of a Data Mart should be departmentally structured
- The implementation cycle of a Data Mart should be measured in short periods of time, i.e., in weeks instead of months or years.
- It is important to involve all stakeholders in planning and designing phase as the data mart implementation could be complex.

- Data Mart Hardware/Software, Networking and Implementation costs should be accurately budgeted in your plan
- Even though if the Data mart is created on the same hardware, they may need some different software to handle user queries. Additional processing power and disk storage requirements should be evaluated for fast user response
- A data mart may be on a different location from the data warehouse. That's why it is important to ensure that they have enough networking capacity to handle the Data volumes needed to transfer data to the data mart**.**
- Implementation cost should budget the time taken for Datamart loading process. Load time increases with increase in complexity of the transformations.

Advantages and Disadvantages of a Data Mart

Advantages

- Data marts contain a subset of organization-wide data. This Data is valuable to a specific group of people in an organization.
- It is cost-effective alternatives to a data warehouse, which can take high costs to build.
- Data Mart allows faster access of Data.
- Data Mart is easy to use as it is specifically designed for the needs of its users. Thus, a data mart can accelerate business processes.
- Data Marts needs less implementation time compared to Data Warehouse systems. It is faster to implement Data Mart as you only need to concentrate the only subset of the data.
- It contains historical data which enables the analyst to determine data trends.

Disadvantages

- Many a times enterprises create too many disparate and unrelated data marts without much benefit. It can become a big hurdle to maintain.
- Data Mart cannot provide company-wide data analysis as their data set is limited.

Data Warehouse Schema

What is Multidimensional schema?

Multidimensional Schema is especially designed to model data warehouse systems. The schemas are designed to address the unique needs of very large databases designed for the analytical purpose (OLAP).

Types of Data Warehouse Schema:

Following are 3 chief types of multidimensional schemas each having its unique advantages.

- Star Schema
- Snowflake Schema
- Galaxy Schema

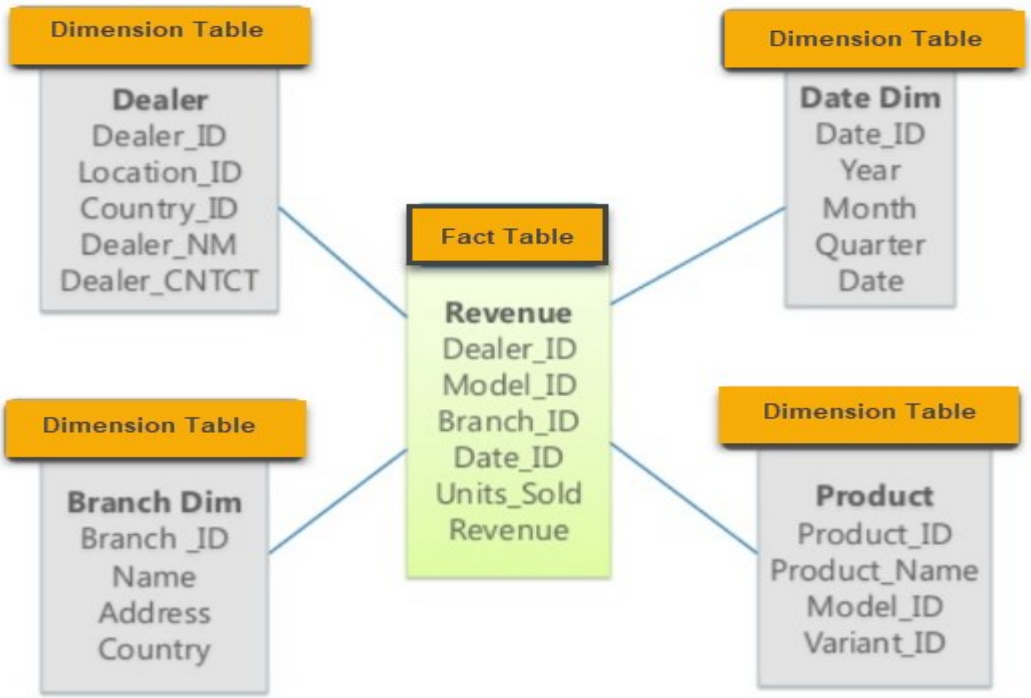
In this tutorial, we will explain Star and Snowflake schema along with Galaxy and Star Cluster Schema-

- What is a Star Schema?
- What is a Snowflake Schema?
- Star Schema Vs Snowflake Schema: Key Differences
- What is a Galaxy schema?
- What is Star Cluster Schema?

What is a Star Schema?

Star Schema in data warehouse, in which the centre of the star can have one fact table and a number of associated dimension tables. It is known as star schema as its structure resembles a star. The Star Schema data model is the simplest type of Data Warehouse schema. It is also known as Star Join Schema and is optimized for querying large data sets.

In the following Star Schema example, the fact table is at the center which contains keys to every dimension table like Dealer_ID, Model ID, Date_ID, Product_ID, Branch_ID & other attributes like Units sold and revenue.



Example of Star Schema Diagram

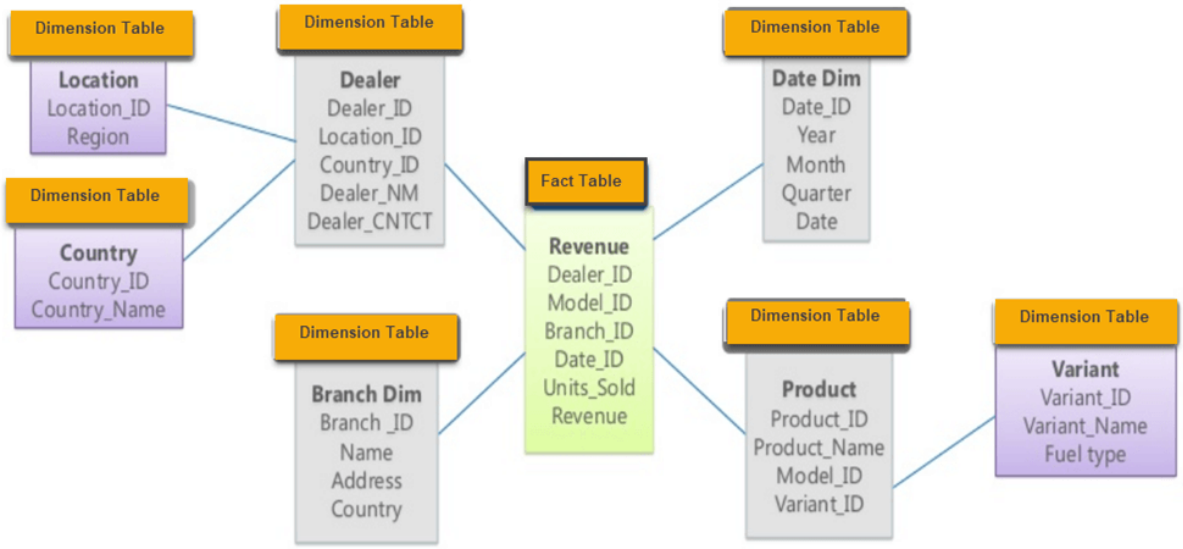
Characteristics of Star Schema:

- Every dimension in a star schema is represented with the only one-dimension table.
- The dimension table should contain the set of attributes.
- The dimension table is joined to the fact table using a foreign key
- The dimension table are not joined to each other
- Fact table would contain key and measure
- The Star schema is easy to understand and provides optimal disk usage.
- The dimension tables are not normalized. For instance, in the above figure, Country_ID does not have Country lookup table as an OLTP design would have.
- The schema is widely supported by BI Tools

What is a Snowflake Schema?

Snowflake Schema in data warehouse is a logical arrangement of tables in a multidimensional database such that the ER diagram resembles a snowflake shape. A Snowflake Schema is an extension of a Star Schema, and it adds additional dimensions. The dimension tables are normalized which splits data into additional tables.

In the following Snowflake Schema example, Country is further normalized into an individual table.



Example of Snowflake Schema

Characteristics of Snowflake Schema:

- The main benefit of the snowflake schema it uses smaller disk space.
- Easier to implement a dimension is added to the Schema
- Due to multiple tables query performance is reduced
- The primary challenge that you will face while using the snowflake Schema is that you need to perform more maintenance efforts because of the more lookup tables.

Star Schema Vs Snowflake Schema: Key Differences

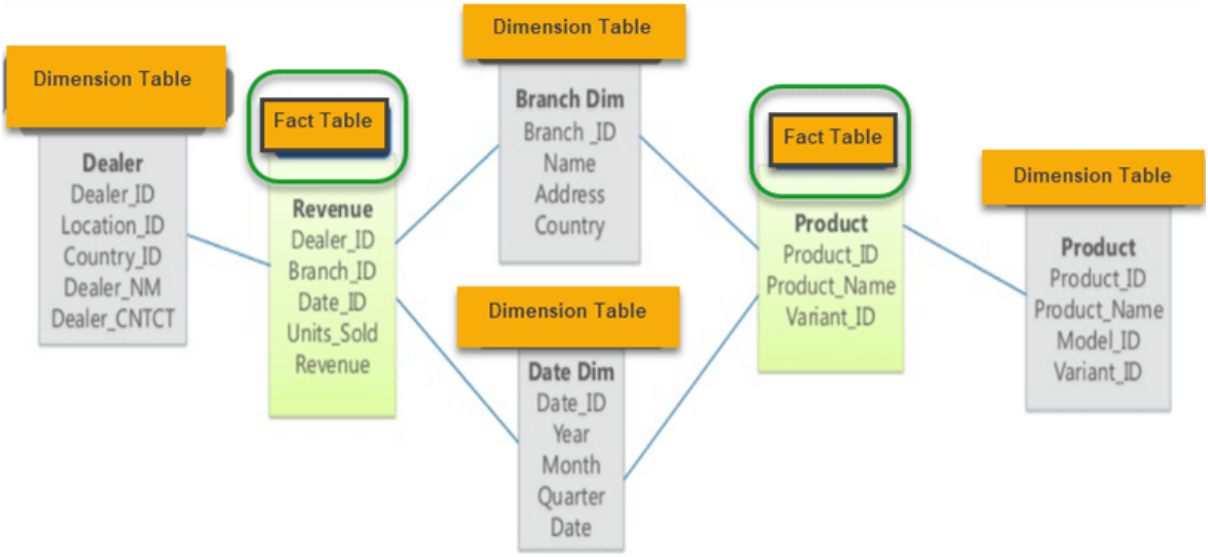
Following is a key difference between Snowflake schema vs Star schema:

Star Schema	Snowflake Schema
Hierarchies for the dimensions are stored in the dimensional table.	Hierarchies are divided into separate tables.
It contains a fact table surrounded by dimension tables.	One fact table surrounded by dimension table which are in turn surrounded by dimension table

In a star schema, only single join creates the relationship between the fact table and any dimension tables.	A snowflake schema requires many joins to fetch the data.
Simple DB Design.	Very Complex DB Design.
Denormalized Data structure and query also run faster.	Normalized Data Structure.
High level of Data redundancy	Very low-level data redundancy
Single Dimension table contains aggregated data.	Data Split into different Dimension Tables.
Cube processing is faster.	Cube processing might be slow because of the complex join.
Offers higher performing queries using Star Join Query Optimization. Tables may be connected with multiple dimensions.	The Snowflake schema is represented by centralized fact table which unlikely connected with multiple dimensions.

What is a Galaxy Schema?

A **Galaxy Schema** contains two fact table that share dimension tables between them. It is also called Fact Constellation Schema. The schema is viewed as a collection of stars hence the name Galaxy Schema.



Example of Galaxy Schema

As you can see in above example, there are two facts table

1. Revenue
2. Product.

In Galaxy schema shares dimensions are called Conformed Dimensions.

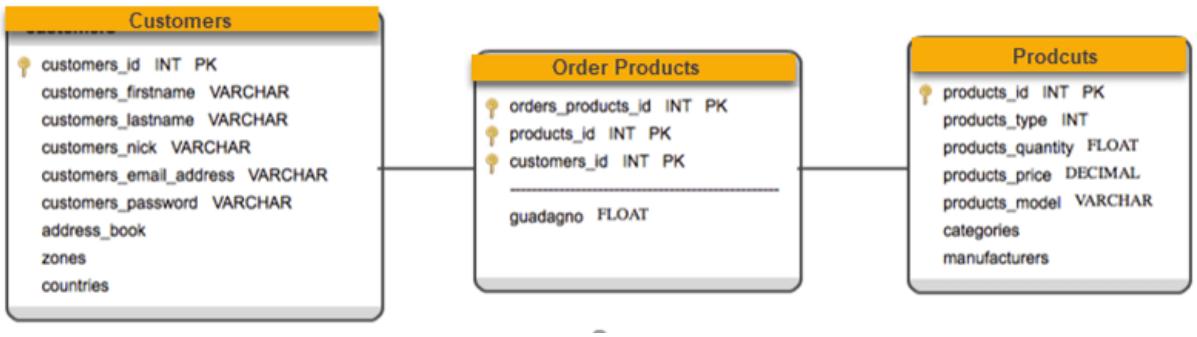
Characteristics of Galaxy Schema:

- The dimensions in this schema are separated into separate dimensions based on the various levels of hierarchy.
- For example, if geography has four levels of hierarchy like region, country, state, and city then Galaxy schema should have four dimensions.

- Moreover, it is possible to build this type of schema by splitting the one-star schema into more Star schemes.
- The dimensions are large in this schema which is needed to build based on the levels of hierarchy.
- This schema is helpful for aggregating fact tables for better understanding.

What is Star Cluster Schema?

Snowflake schema contains fully expanded hierarchies. However, this can add complexity to the Schema and requires extra joins. On the other hand, star schema contains fully collapsed hierarchies, which may lead to redundancy. So, the best solution may be a balance between these two schemas which is Star Cluster Schema design.



Example of Star Cluster Schema

Overlapping dimensions can be found as forks in hierarchies. A fork happens when an entity acts as a parent in two different dimensional hierarchies. Fork entities then identified as classification with one-to-many relationships.

Data Warehousing Design

What is Data Warehousing?

A Data Warehousing (DW) is process for collecting and managing data from varied sources to provide meaningful business insights. A Data warehouse is typically used to connect and analyze business data from heterogeneous sources. The data warehouse is the core of the BI system which is built for data analysis and reporting.

It is a blend of technologies and components which aids the strategic use of data. It is electronic storage of a large amount of information by a business which is designed for query and analysis instead of transaction processing. It is a process of transforming data into information and making it available to users in a timely manner to make a difference.

In this Data Warehouse (DWH) tutorial, you will learn more about-

Data Warehouse Tools

The decision support database (Data Warehouse) is maintained separately from the organization's operational database. However, the data warehouse is not a product but an environment. It is an architectural construct of an information system which provides users with current and historical decision support information which is difficult to access or present in the traditional operational data store.

You many know that a 3NF-designed database for an inventory system many have tables related to each other. For example, a report on current inventory information can include more than 12 joined conditions. This can quickly slow down the response time of the query and report. A data warehouse provides a new design which can help to reduce the response time and helps to enhance the performance of queries for reports and analytics.

Data warehouse system is also known by the following name:

- Decision Support System (DSS)
- Executive Information System
- Management Information System
- Business Intelligence Solution
- Analytic Application
- Data Warehouse

History of Datawarehouse

The Datawarehouse benefits users to understand and enhance their organization's performance. The need to warehouse data evolved as computer systems became more complex and needed to handle increasing amounts of Information. However, Data Warehousing is a not a new thing.

Here are some key events in evolution of Data Warehouse-

- 1960- Dartmouth and General Mills in a joint research project, develop the terms dimensions and facts.
- 1970- A Nielsen and IRI introduces dimensional data marts for retail sales.
- 1983- Tera Data Corporation introduces a database management system which is specifically designed for decision support
- Data warehousing started in the late 1980s when IBM worker Paul Murphy and Barry Devlin developed the Business Data Warehouse.
- However, the real concept was given by Inmon Bill. He was considered as a father of data warehouse. He had written about a variety of topics for building, usage, and maintenance of the warehouse & the Corporate Information Factory.

How Datawarehouse works?

A Data Warehouse works as a central repository where information arrives from one or more data sources. Data flows into a data warehouse from the transactional system and other relational databases.

Data may be:

1. Structured
2. Semi-structured
3. Unstructured data

The data is processed, transformed, and ingested so that users can access the processed data in the Data Warehouse through Business Intelligence tools, SQL clients, and spreadsheets. A data warehouse merges information coming from different sources into one comprehensive database.

By merging all of this information in one place, an organization can analyze its customers more holistically. This helps to ensure that it has considered all the information available. Data warehousing makes data mining possible. Data mining is looking for patterns in the data that may lead to higher sales and profits.

Types of Data Warehouse

Three main types of Data Warehouses (DWH) are:

1. Enterprise Data Warehouse (EDW):

Enterprise Data Warehouse (EDW) is a centralized warehouse. It provides decision support service across the enterprise. It offers a unified approach for organizing and representing data. It also provide the ability to classify data according to the subject and give access according to those divisions.

2. Operational Data Store:

Operational Data Store, which is also called ODS, are nothing but data store required when neither Data warehouse nor OLTP systems support organizations reporting needs. In ODS, Data warehouse is refreshed in real time. Hence, it is widely preferred for routine activities like storing records of the Employees.

3. Data Mart:

A data mart is a subset of the data warehouse. It specially designed for a particular line of business, such as sales, finance, sales or finance. In an independent data mart, data can collect directly from sources.

General stages of Data Warehouse

Earlier, organizations started relatively simple use of data warehousing. However, over time, more sophisticated use of data warehousing begun.

The following are general stages of use of the data warehouse (DWH):

Offline Operational Database:

In this stage, data is just copied from an operational system to another server. In this way, loading, processing, and reporting of the copied data do not impact the operational system’s performance.

Offline Data Warehouse:

Data in the Datawarehouse is regularly updated from the Operational Database. The data in Datawarehouse is mapped and transformed to meet the Datawarehouse objectives.

Real time Data Warehouse:

In this stage, Data warehouses are updated whenever any transaction takes place in operational database. For example, Airline or railway booking system.

Integrated Data Warehouse:

In this stage, Data Warehouses are updated continuously when the operational system performs a transaction. The Datawarehouse then generates transactions which are passed back to the operational system.

Components of Data warehouse

Four components of Data Warehouses are:

Load manager: Load manager is also called the front component. It performs with all the operations associated with the extraction and load of data into the warehouse. These operations include transformations to prepare the data for entering into the Data warehouse.

Warehouse Manager: Warehouse manager performs operations associated with the management of the data in the warehouse. It performs operations like analysis of data to ensure consistency, creation of indexes and views, generation of denormalization and aggregations, transformation and merging of source data and archiving and baking-up data.

Query Manager: Query manager is also known as backend component. It performs all the operation operations related to the management of user queries. The operations of this Data warehouse components are direct queries to the appropriate tables for scheduling the execution of queries.

End-user access tools:

This is categorized into five different groups like 1. Data Reporting 2. Query Tools 3. Application development tools 4. EIS tools, 5. OLAP tools and data mining tools.

Who needs Data warehouse?

DWH (Data warehouse) is needed for all types of users like:

- Decision makers who rely on mass amount of data
- Users who use customized, complex processes to obtain information from multiple data sources.
- It is also used by the people who want simple technology to access the data
- It also essential for those people who want a systematic approach for making decisions.
- If the user wants fast performance on a huge amount of data which is a necessity for reports, grids or charts, then Data warehouse proves useful.
- Data warehouse is a first step If you want to discover 'hidden patterns' of data-flows and groupings.

What Is a Data Warehouse Used For?

Here, are most common sectors where Data warehouse is used:

Airline:

In the Airline system, it is used for operation purpose like crew assignment, analyses of route profitability, frequent flyer program promotions, etc.

Banking:

It is widely used in the banking sector to manage the resources available on desk effectively. Few banks also used for the market research, performance analysis of the product and operations.

Healthcare:

Healthcare sector also used Data warehouse to strategize and predict outcomes, generate patient's treatment reports, share data with tie-in insurance companies, medical aid services, etc.

Public sector:

In the public sector, data warehouse is used for intelligence gathering. It helps government agencies to maintain and analyze tax records, health policy records, for every individual.

Investment and Insurance sector:

In this sector, the warehouses are primarily used to analyze data patterns, customer trends, and to track market movements.

Retain chain:

In retail chains, Data warehouse is widely used for distribution and marketing. It also helps to track items, customer buying pattern, promotions and also used for determining pricing policy.

Telecommunication:

A data warehouse is used in this sector for product promotions, sales decisions and to make distribution decisions.

Hospitality Industry:

This Industry utilizes warehouse services to design as well as estimate their advertising and promotion campaigns where they want to target clients based on their feedback and travel patterns.

Steps to Implement Data Warehouse

The best way to address the business risk associated with a Datawarehouse implementation is to employ a three-prong strategy as below

1. **Enterprise strategy**: Here we identify technical including current architecture and tools. We also identify facts, dimensions, and attributes. Data mapping and transformation is also passed.
2. **Phased delivery**: Datawarehouse implementation should be phased based on subject areas. Related business entities like booking and billing should be first implemented and then integrated with each other.
3. **Iterative Prototyping**: Rather than a big bang approach to implementation, the Datawarehouse should be developed and tested iteratively.

Here, are key steps in Datawarehouse implementation along with its deliverables.

Step	Tasks	Deliverables
1	Need to define project scope	Scope Definition
2	Need to determine business needs	Logical Data Model
3	Define Operational Datastore requirements	Operational Data Store Model
4	Acquire or develop Extraction tools	Extract tools and Software
5	Define Data Warehouse Data requirements	Transition Data Model
6	Document missing data	To Do Project List
7	Maps Operational Data Store to Data Warehouse	D/W Data Integration Map
8	Develop Data Warehouse Database design	D/W Database Design
9	Extract Data from Operational Data Store	Integrated D/W Data Extracts
10	Load Data Warehouse	Initial Data Load
11	Maintain Data Warehouse	On-going Data Access and Subsequent Loads

Best practices to implement a Data Warehouse

- Decide a plan to test the consistency, accuracy, and integrity of the data.
- The data warehouse must be well integrated, well defined and time stamped.

- While designing Datawarehouse make sure you use right tool, stick to life cycle, take care about data conflicts and ready to learn you're your mistakes.
- Never replace operational systems and reports
- Don't spend too much time on extracting, cleaning and loading data.
- Ensure to involve all stakeholders including business personnel in Datawarehouse implementation process. Establish that Data warehousing is a joint/ team project. You don't want to create Data warehouse that is not useful to the end users.
- Prepare a training plan for the end users.

Why We Need Data Warehouse? Advantages & Disadvantages

Advantages of Data Warehouse (DWH):

- Data warehouse allows business users to quickly access critical data from some sources all in one place.
- Data warehouse provides consistent information on various cross-functional activities. It is also supporting ad-hoc reporting and query.
- Data Warehouse helps to integrate many sources of data to reduce stress on the production system.
- Data warehouse helps to reduce total turnaround time for analysis and reporting.
- Restructuring and Integration make it easier for the user to use for reporting and analysis.
- Data warehouse allows users to access critical data from the number of sources in a single place. Therefore, it saves user's time of retrieving data from multiple sources.
- Data warehouse stores a large amount of historical data. This helps users to analyze different time periods and trends to make future predictions.

Disadvantages of Data Warehouse:

- Not an ideal option for unstructured data.
- Creation and Implementation of Data Warehouse is surely time confusing affair.
- Data Warehouse can be outdated relatively quickly
- Difficult to make changes in data types and ranges, data source schema, indexes, and queries.
- The data warehouse may seem easy, but actually, it is too complex for the average users.
- Despite best efforts at project management, data warehousing project scope will always increase.
- Sometime warehouse users will develop different business rules.
- Organisations need to spend lots of their resources for training and Implementation purpose.

The Future of Data Warehousing

- Change in **Regulatory constrains** may limit the ability to combine source of disparate data. These disparate sources may include unstructured data which is difficult to store.
- As the **size** of the databases grows, the estimates of what constitutes a very large database continue to grow. It is complex to build and run data warehouse systems which are always increasing in size. The hardware and software resources are available today do not allow to keep a large amount of data online.

- **Multimedia data** cannot be easily manipulated as text data, whereas textual information can be retrieved by the relational software available today. This could be a research subject.

Data Warehouse Tools

There are many Data Warehousing tools are available in the market. Here, are some most prominent one:

1. MarkLogic:

MarkLogic is useful data warehousing solution that makes data integration easier and faster using an array of enterprise features. This tool helps to perform very complex search operations. It can query different types of data like documents, relationships, and metadata.
https://www.marklogic.com/product/getting-started/

2. Oracle:

Oracle is the industry-leading database. It offers a wide range of choice of data warehouse solutions for both on-premises and in the cloud. It helps to optimize customer experiences by increasing operational efficiency.
https://www.oracle.com/index.html

3. Amazon RedShift:

Amazon Redshift is Data warehouse tool. It is a simple and cost-effective tool to analyze all types of data using standard SQL and existing BI tools. It also allows running complex queries against petabytes of structured data, using the technique of query optimization.
https://aws.amazon.com/redshift/?nc2=h_m1

Here is a complete list of useful Datawarehouse Tools.

Week 12

Mining Complex Types of Data
Trends in Data Mining

Seokho Chi
Associate Professor I Ph.D.
SNU Construction Innovation Lab

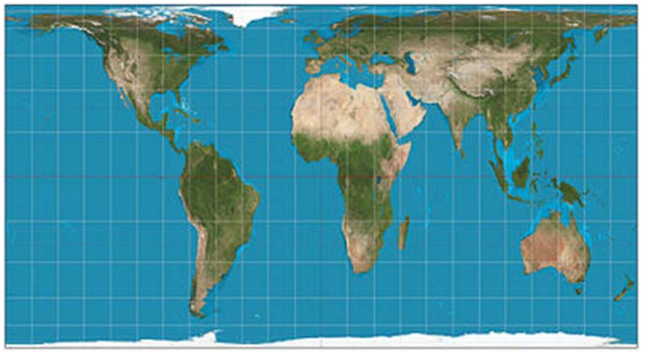
Source: Tan, Kumar, Steinback (2006)

Mining Complex Types of Data

- Mining spatial databases
- Mining multimedia databases
- Mining time-series and sequence data
- Mining text databases
- Mining the World-Wide Web
- Summary

Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Spatial Data



- Spatial data integration: a big issue
  - Structure-specific formats (raster- vs. vector-based, OO vs. relational models, different storage and indexing, etc.)
  - Vendor-specific formats (ESRI, MapInfo, Integraph, IDRISI, etc.)
  - Geo-specific formats (geographic vs. equal area projection, etc.)

Raster-based: composed of pixels
Vector-based: composed of paths (points where the paths start and end, straight or curved, border and fill, etc.)
ESRI: GIS mapping software

Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

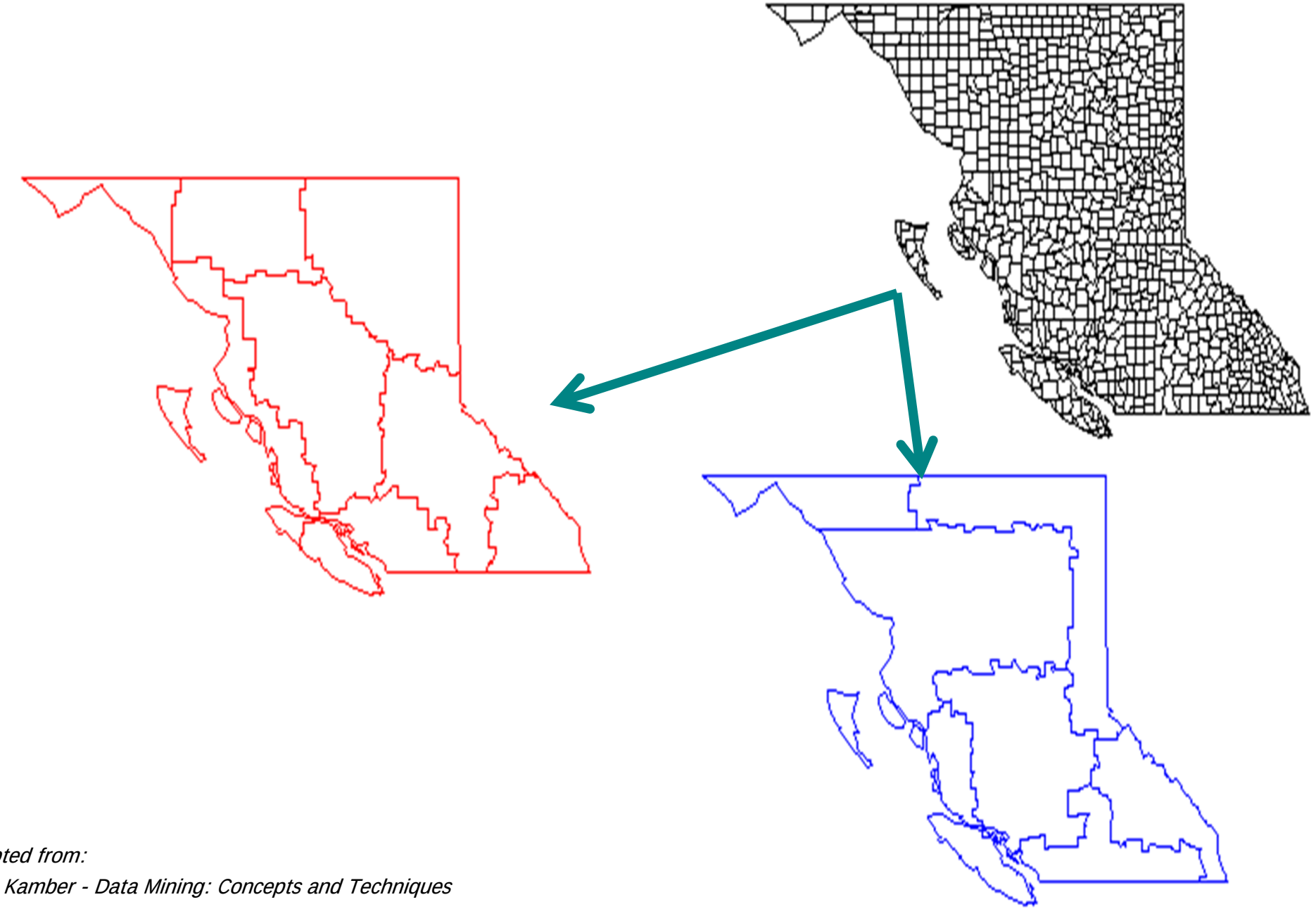
Example: British Columbia Weather Pattern Analysis

- Input
  - A map with about 3,000 weather probes scattered in B.C.
  - Daily data for temperature, precipitation, wind velocity, etc.
- Output
  - A map that reveals patterns: merged (similar) regions
- Goals
  - Interactive analysis
  - Fast response time
  - Minimizing storage space used
- Challenge
  - A merged region may contain hundreds of "primitive" regions (polygons)

Adapted from:

Han, Kamber - Data Mining: Concepts and Techniques

Dynamic Merging of Spatial Objects



Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Spatial Association Analysis

- Spatial association rule: $A \Rightarrow B$ [*s%, c%*]
  - A and B are sets of spatial or non-spatial predicates
    - Topological relations: *intersects, overlaps, disjoint,* etc.
    - Spatial orientations: *left_of, west_of, under,* etc.
    - Distance information: *close_to, within_distance,* etc.
  - *s%* is the support and *c%* is the confidence of the rule
- Examples

1) *is_a(x, large_town) ^ intersect(x, highway)* $\rightarrow$ *adjacent_to(x, water)*
[7%, 85%]

2) What kinds of objects are typically located close to golf courses?

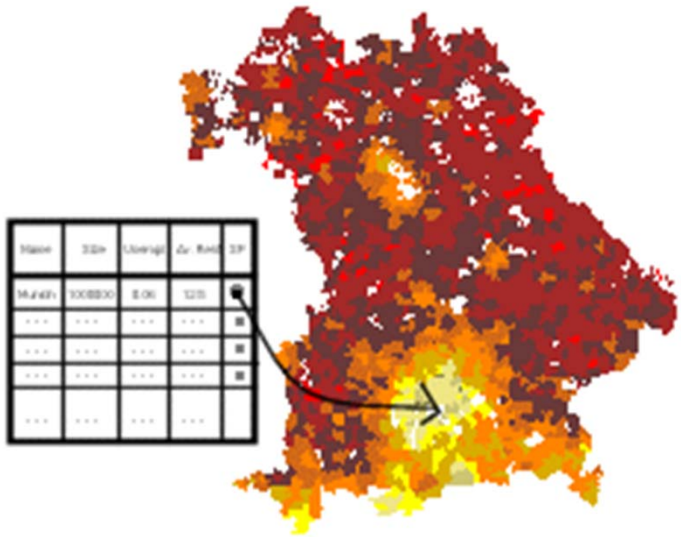
Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Spatial Classification

- Analyze spatial objects to derive classification schemes in relevance to certain spatial properties (district, highway, river, etc.)
- Employ most of the classification methods
  - Decision-tree classification, Naïve-Bayesian classifier, neural network, etc.
  - Association-based multi-dimensional classification - Example: classifying house value based on proximity to lakes, highways, mountains, etc.

Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Spatial Trend Analysis

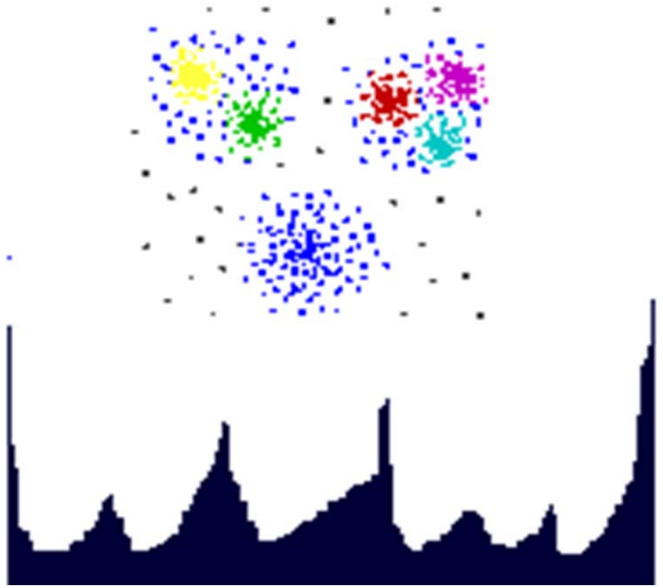


- Function
  - Detect changes and trends along a spatial dimension
  - Study the trend of non-spatial or spatial data changing with space
- Application examples
  - Observe the trend of changes of the climate or vegetation with increasing distance from an ocean
  - Crime rate or unemployment rate change with regard to city geo-distribution
  - Farm Insurance Frauds (from NPR)

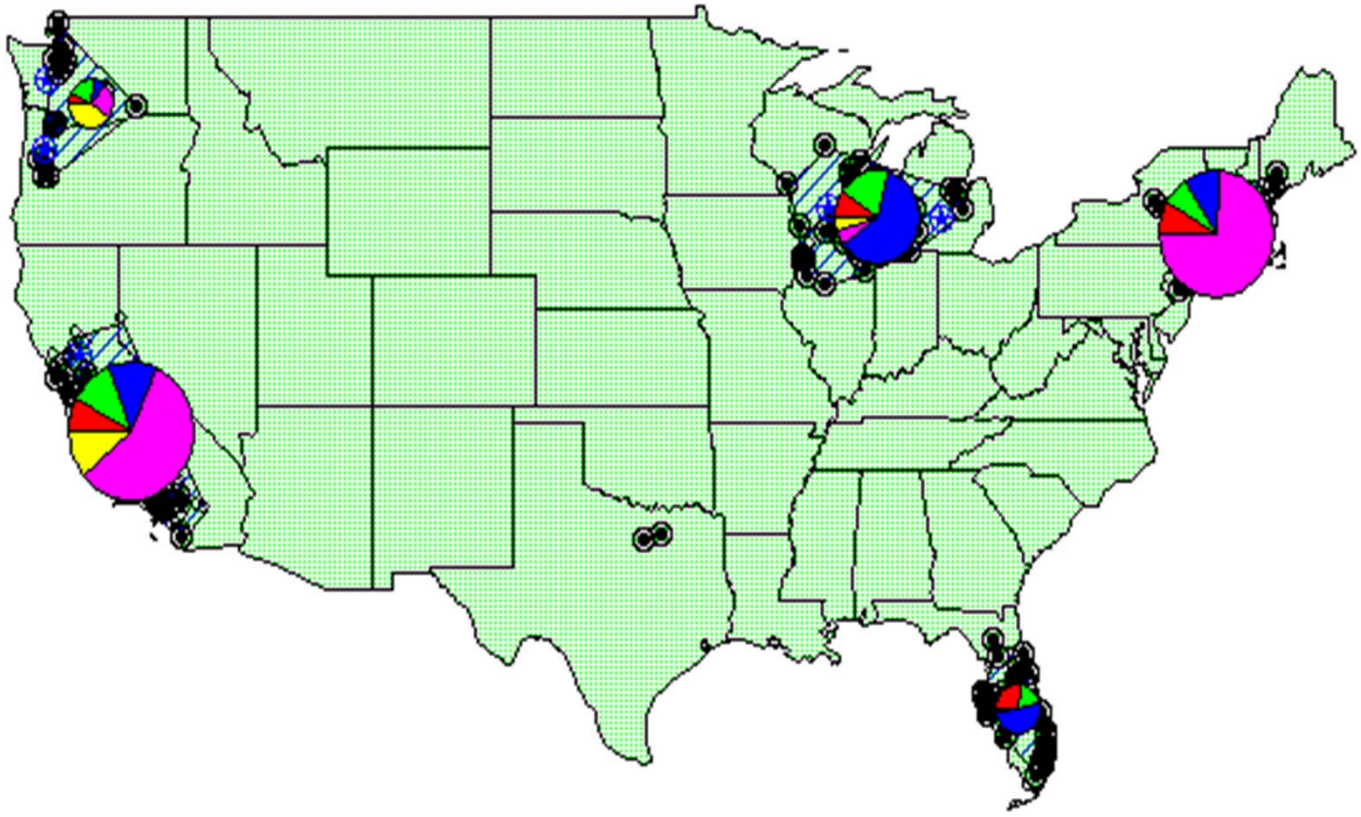
"Perpetrators falsely claim weather or insects destroyed their crops and cash in on a government-backed insurance program. Some don't bother planting at all. Others sell their harvests in secret."

Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Spatial Cluster Analysis



- Mining clusters—k-means, k-medoids, hierarchical, density-based, etc.
- Analysis of distinct features of the clusters



Mining Complex Types of Data

- Mining spatial databases
- Mining multimedia databases
- Mining time-series and sequence data
- Mining text databases
- Mining the World-Wide Web
- Summary

Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Similarity Search in Multimedia Data

- Description-based retrieval systems
  - Build indices and perform object retrieval based on image descriptions, such as keywords, captions, size, and time of creation
  - Labor-intensive if performed manually
  - Results are typically of poor quality if automated
- Content-based retrieval systems
  - Support retrieval based on the image content, such as color histogram, texture, shape, objects, and wavelet transforms

Mining Multimedia Databases

Refining or combining searches

Search for “blue sky”
(top layout grid is blue)

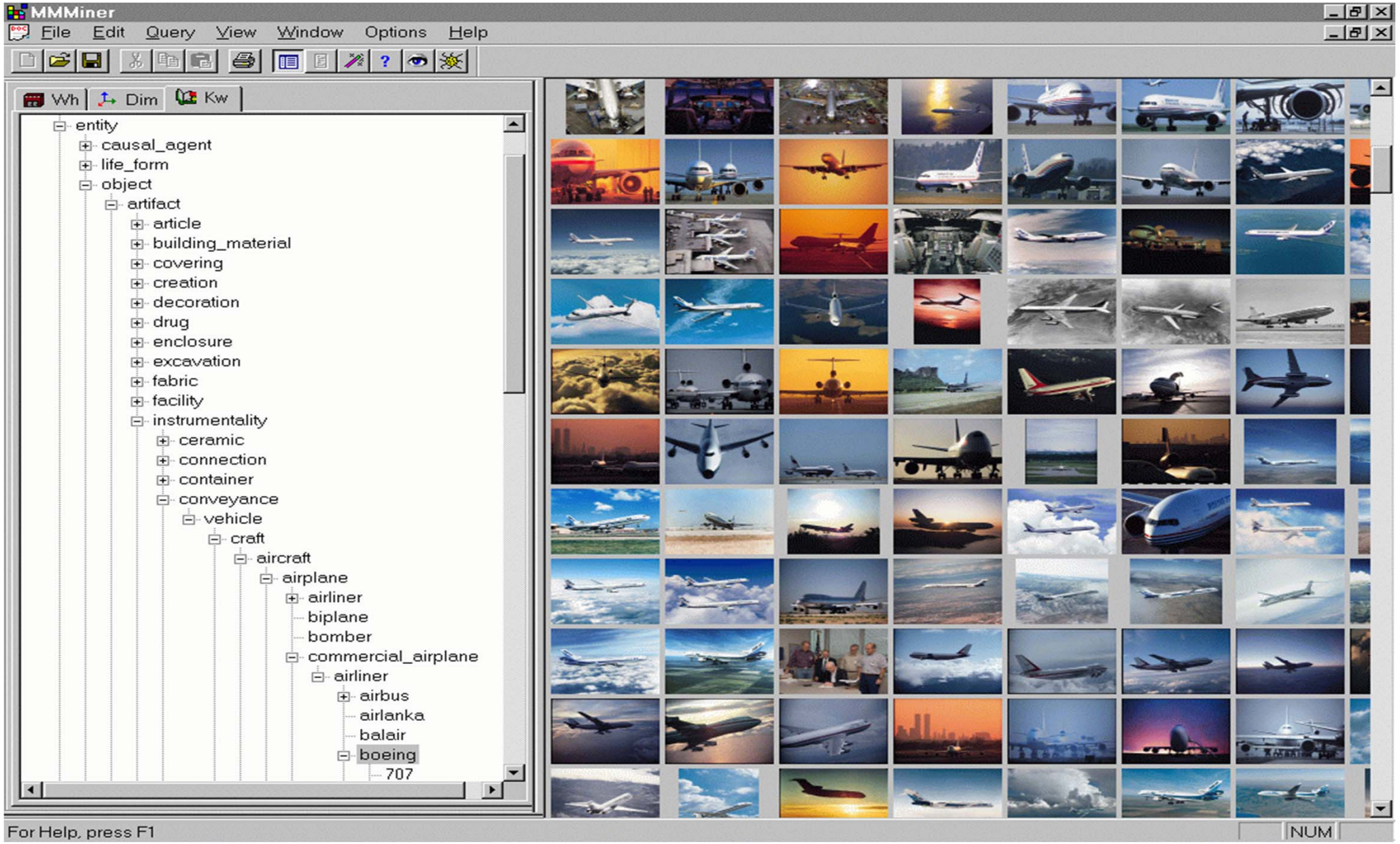
Search for “airplane in blue sky”
(top layout grid is blue and keyword = “airplane”)

Search for “blue sky and green meadows”
(top layout grid is blue and bottom is green)

Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

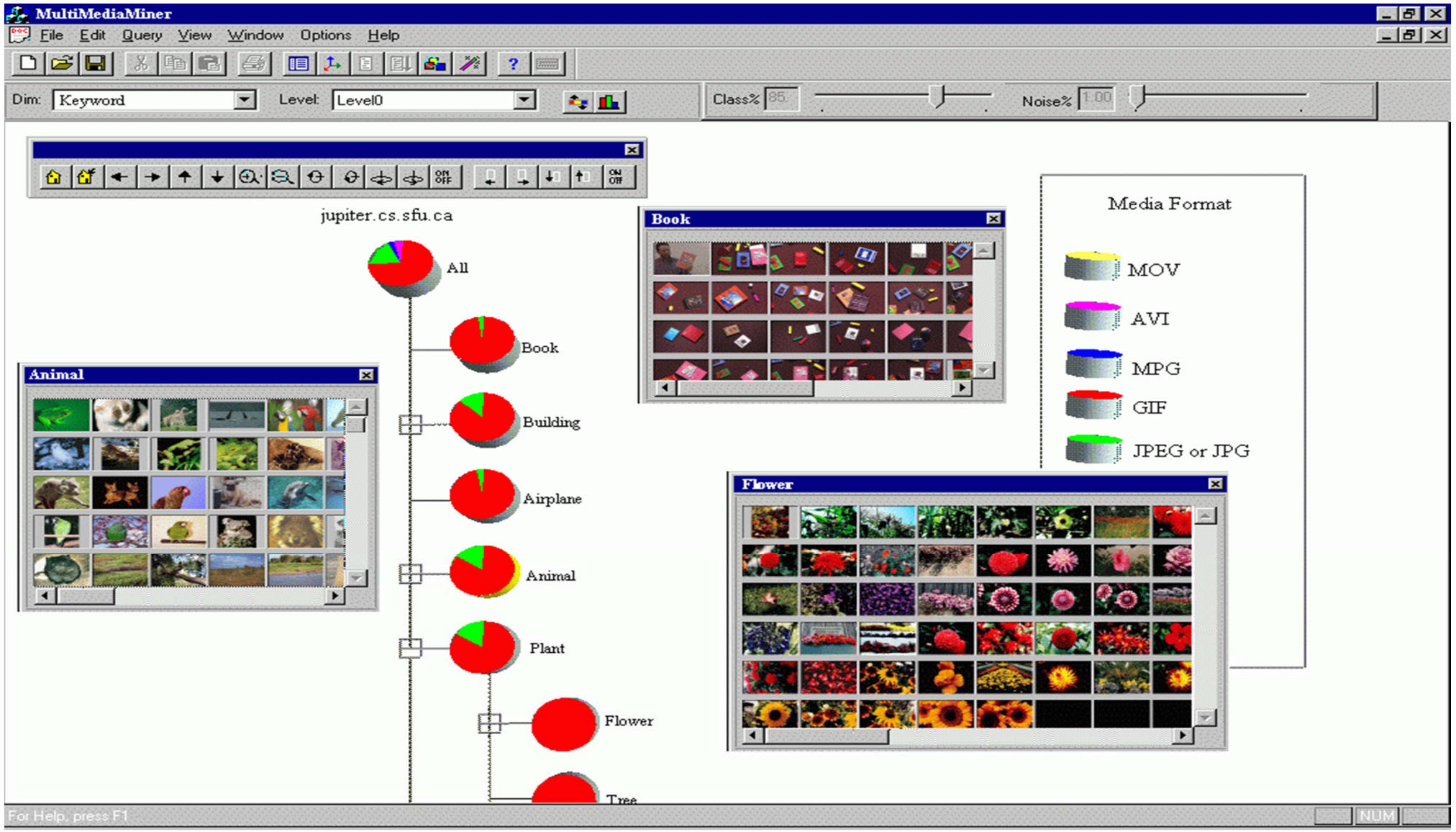
Mining Multimedia Databases in MultiMediaMiner

Thumbnails of images and video frames in the database can be browsed with MultiMediaMiner user interface.



Classification in MultiMediaMiner

MM-Characterizer, MM-Comparator, MM-Associator, MM-Classifier

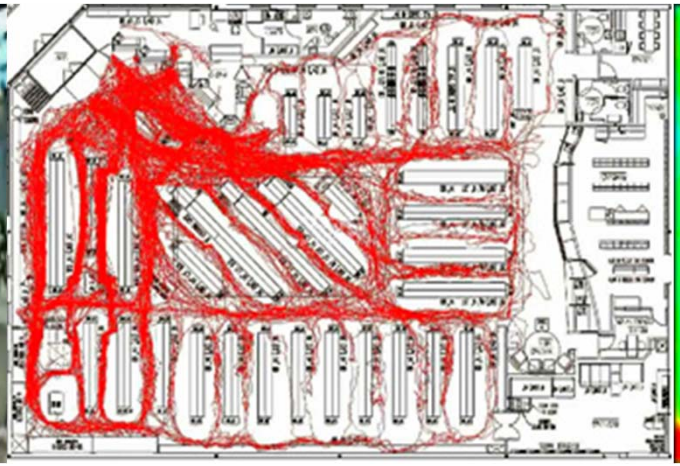


Adapted from:

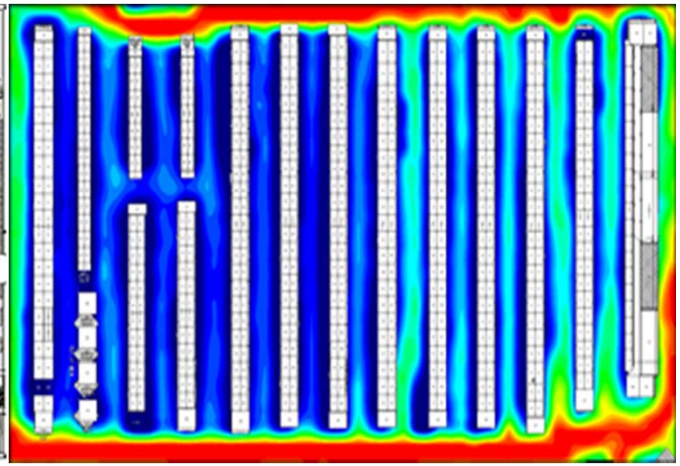
Han, Kamber - Data Mining: Concepts and Techniques

Classification in VideoMining (www.videomining.com)

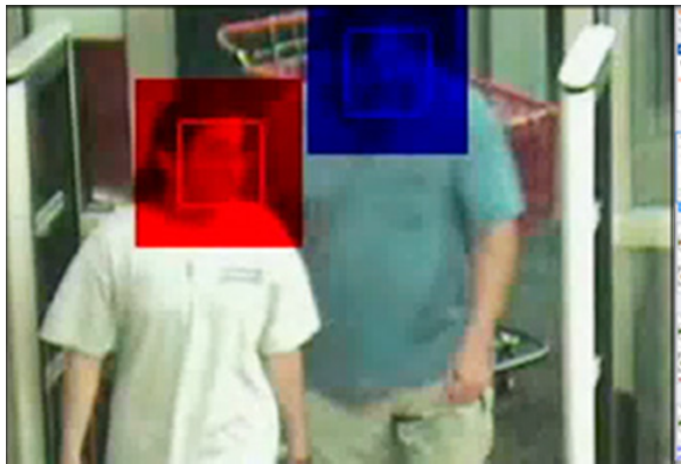
Tracking the Shopper Path



Multiple Shopping Trips



Heat Maps



Demographics Analysis

Market Analysis

Adapted from:

Han, Kamber - Data Mining: Concepts and Techniques

Mining Complex Types of Data

- Mining spatial databases
- Mining multimedia databases
- Mining time-series and sequence data
- Mining stream data
- Mining text databases
- Mining the World-Wide Web
- Summary

Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

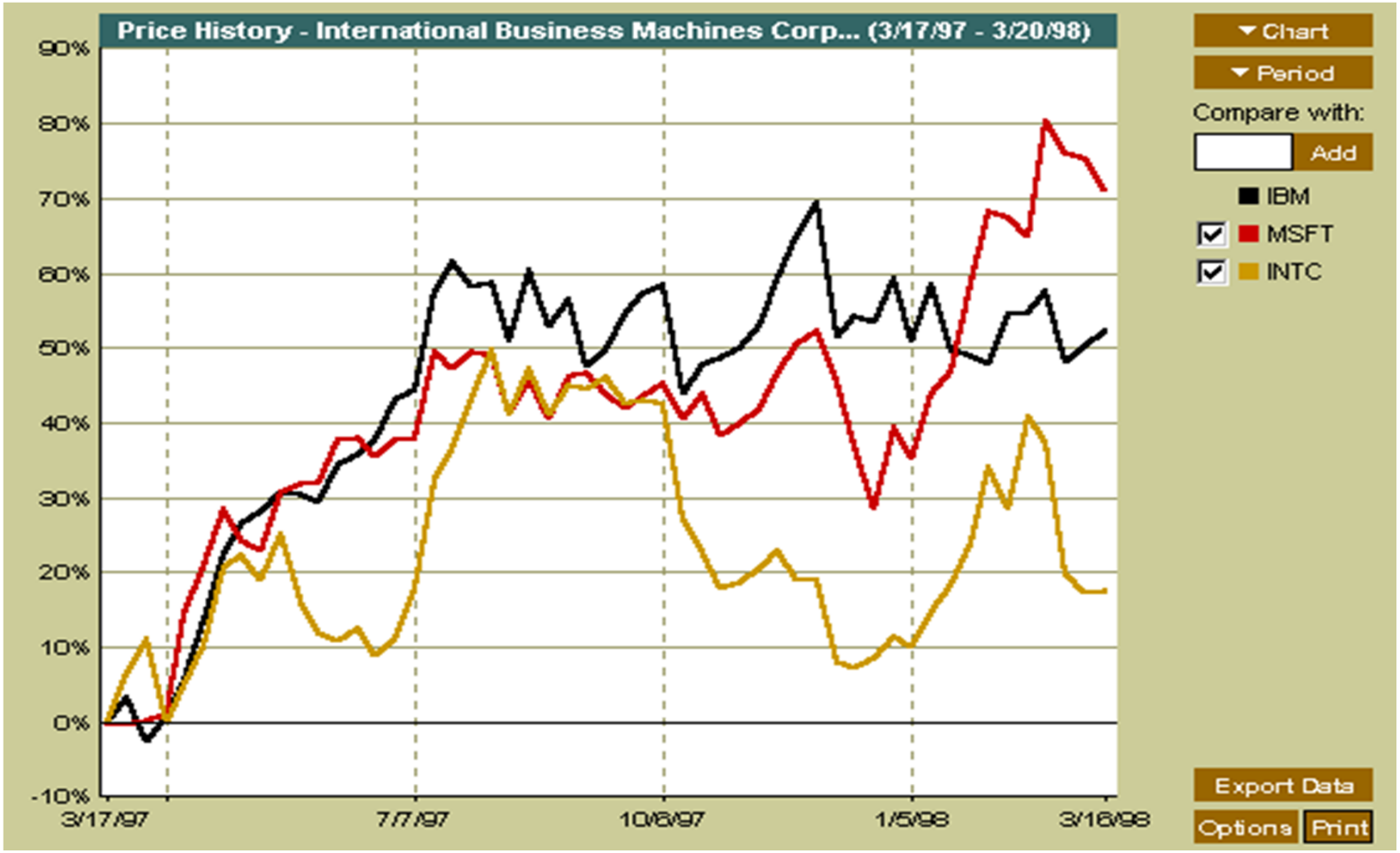
Mining Time-Series and Sequence Data

- Time-series database
  - Consists of sequences of values or events changing with time
  - Data is recorded at regular intervals
  - Characteristic time-series components
    - Trend, cycle, seasonal, irregular
- Applications
  - Financial: stock price, inflation
  - Biomedical: blood pressure
  - Meteorological: precipitation

Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Mining Time-Series and Sequence Data

Time-series plot



Adapted from:

Han, Kamber - Data Mining: Concepts and Techniques

Mining Time-Series and Sequence Data: Trend analysis

- A time series can be illustrated as a time-series graph which describes a point moving with the passage of time
- Categories of Time-Series Movements
  - Long-term or trend movements (trend curve)
  - Cyclic movements or cycle variations, e.g., business cycles
  - Seasonal movements or seasonal variations
    - i.e, almost identical patterns that a time series appears to follow during corresponding months of successive years.
  - Irregular or random movements

Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Mining Complex Types of Data

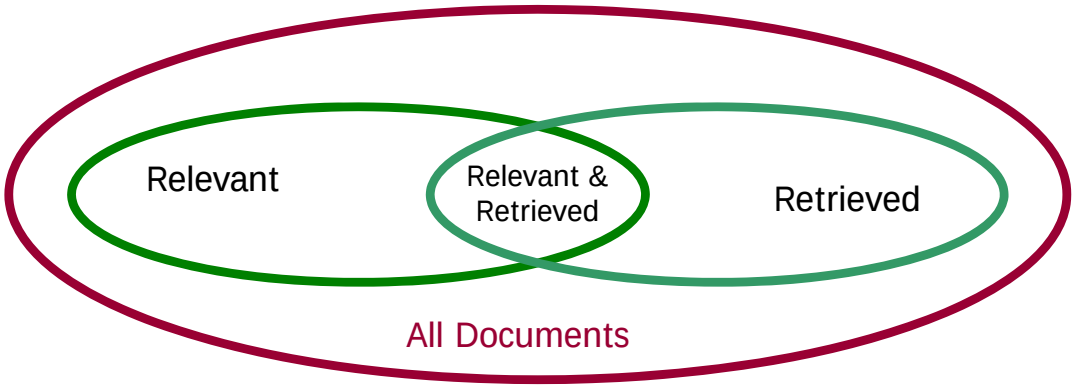
- Mining spatial databases
- Mining multimedia databases
- Mining time-series and sequence data
- Mining stream data
- Mining text databases
- Mining the World-Wide Web
- Summary

Text Databases and IR

- Text databases (document databases)
  - Large collections of documents from various sources: news articles, research papers, books, digital libraries, e-mail messages, and Web pages, library database, etc.
  - Data stored is usually *semi-structured*
  - Traditional information retrieval techniques become inadequate for the increasingly vast amounts of text data
- Information retrieval
  - A field developed in parallel with database systems
  - Information is organized into (a large number of) documents
  - Information retrieval problem: locating relevant documents based on user input, such as keywords or example documents

Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Basic Measures for Text Retrieval



- Precision: the percentage of retrieved documents that are in fact relevant to the query (i.e., "correct" responses)

$$precision = \frac{|\{Relevant\} \cap \{Retrieved\}|}{|\{Retrieved\}|}$$

- Recall: the percentage of documents that are relevant to the query and were, in fact, retrieved

$$recall = \frac{|\{Relevant\} \cap \{Retrieved\}|}{|\{Relevant\}|}$$

Adapted from:

Han, Kamber - Data Mining: Concepts and Techniques

Information Retrieval

- Basic Concepts
  - A document can be described by a set of representative keywords called index terms.
  - Different index terms have varying relevance when used to describe document contents.
  - This effect is captured through the assignment of numerical weights to each index term of a document. (e.g.: frequency, tf-idf)

Term Frequency – Inverse Document Frequency:

TF-IDF = TF x IDF

TF: Frequency of terms within the document
IDF: Inverse of the frequency of terms within the similar document group

e.g.) TF of "worker" is high within a construction document
But DF of "worker" within the construction document group is high, so IDF becomes small

****Frequent in a document + Unique in a document group → higher weight***

Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Boolean Model: Keyword-Based Retrieval

- Consider that index terms are either present or absent in a document
- The index term weights are assumed to be all binaries
- A document can be identified by a set of keywords

- Queries may use expressions of keywords
  - Car *and* repair shop, tea *or* coffee, DBMS *but not* Oracle
  - Synonymy: multiple words with the same meaning
    - e.g., elevator and lift, repair and maintenance
  - Polysemy: words that have multiple meanings
    - E.g.: get, door (paint the door vs walk through the door)

Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Keyword-Based Association Analysis

- Motivation
  - Collect sets of keywords or terms that occur frequently together and then find the association or correlation relationships among them
- Association Analysis Process
  - Preprocess the text data by parsing, stemming, removing stop words, etc.
  - Evoke association mining algorithms
    - Consider each document as a transaction
    - View a set of keywords in the document as a set of items in the transaction
  - Term level association mining

Stop list "irrelevant" : a, the, of, for, to, with
Word stem : drug, drugs, drugged

Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Mining Complex Types of Data

- Mining spatial databases
- Mining multimedia databases
- Mining time-series and sequence data
- Mining stream data
- Mining text databases
- Mining the World-Wide Web
- Summary

Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

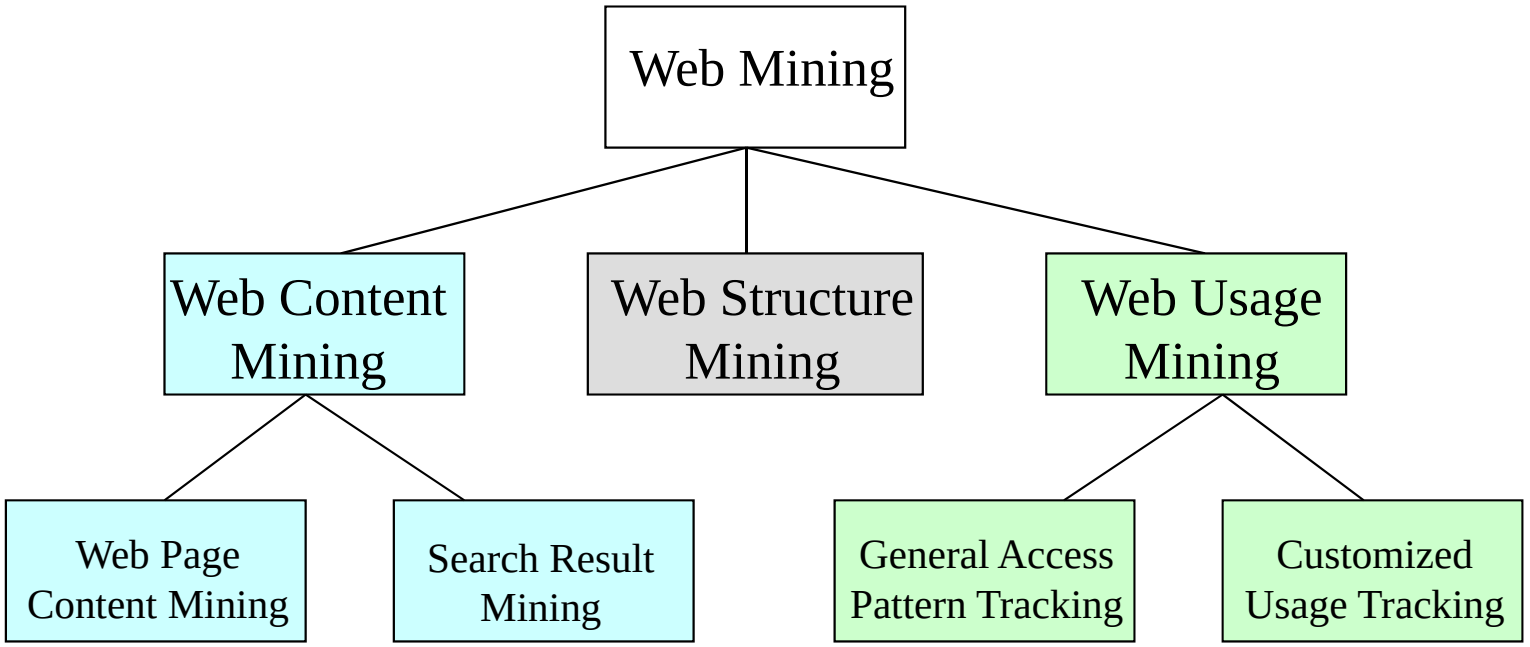
Mining the World-Wide Web

- The WWW is huge, widely distributed, global information service center for:
  - Information services: news, advertisements, consumer information, financial management, education, government, e-commerce, etc.
  - Hyper-link information
  - Access and usage information
- WWW provides rich sources for data mining
- Challenges
  - Too huge for effective data warehousing and data mining
  - Too complex and heterogeneous: no standards and structure

99% of the Web information is useless to 99% of Web users
How can we find high-quality Web pages on a specified topic?

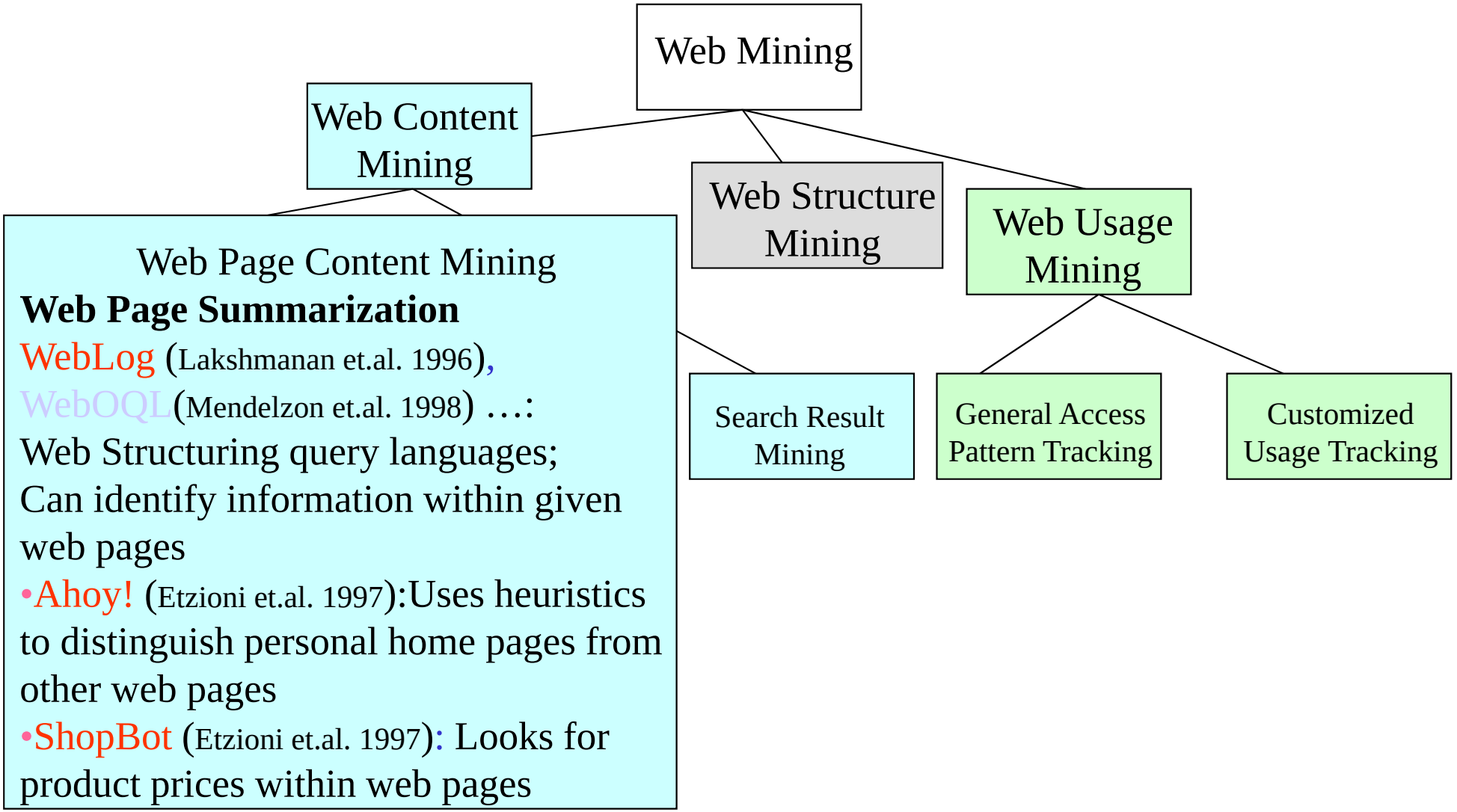
Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Web Mining Taxonomy



Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

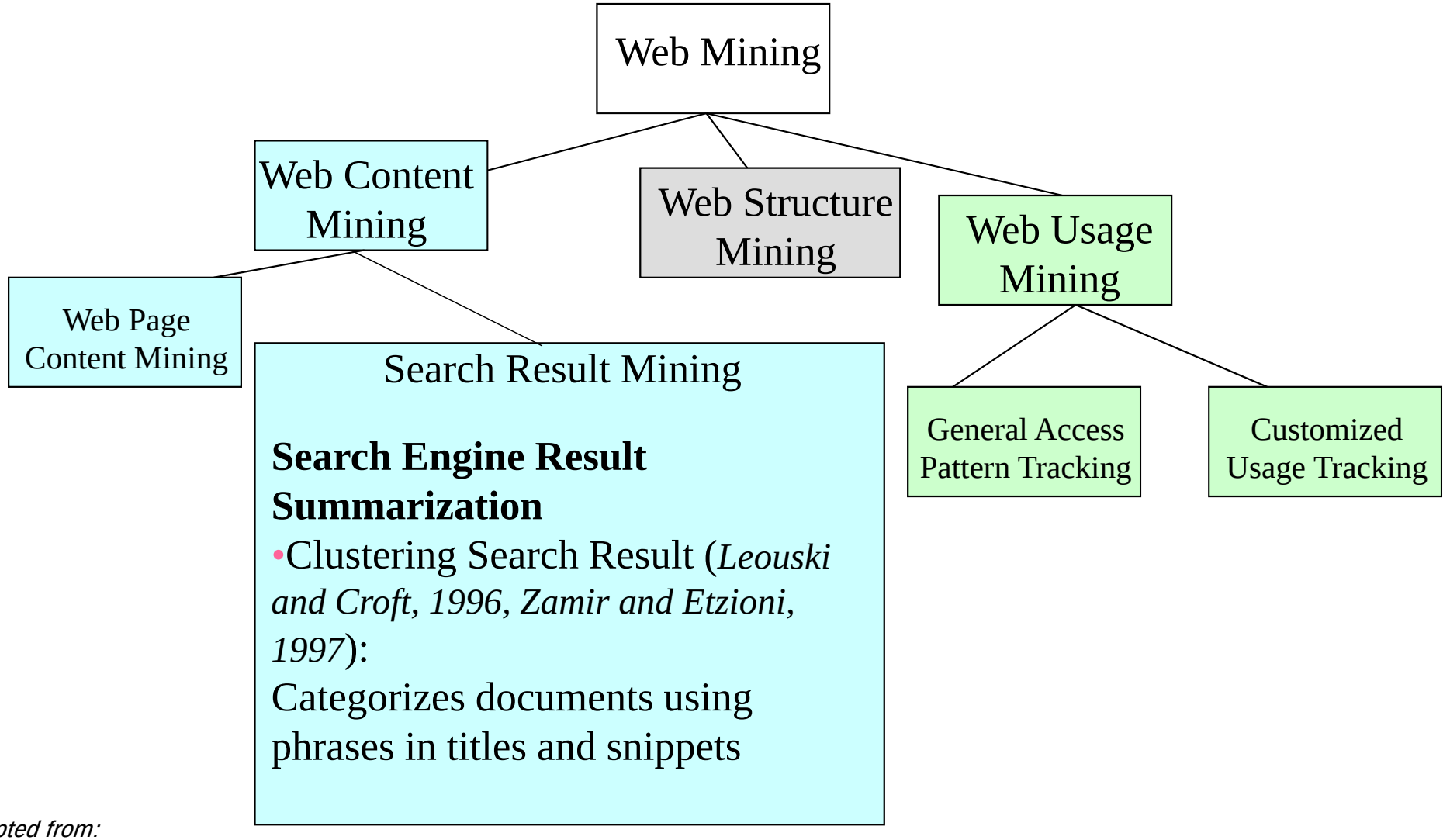
Mining the World-Wide Web



Adapted from:

Han, Kamber - Data Mining: Concepts and Techniques

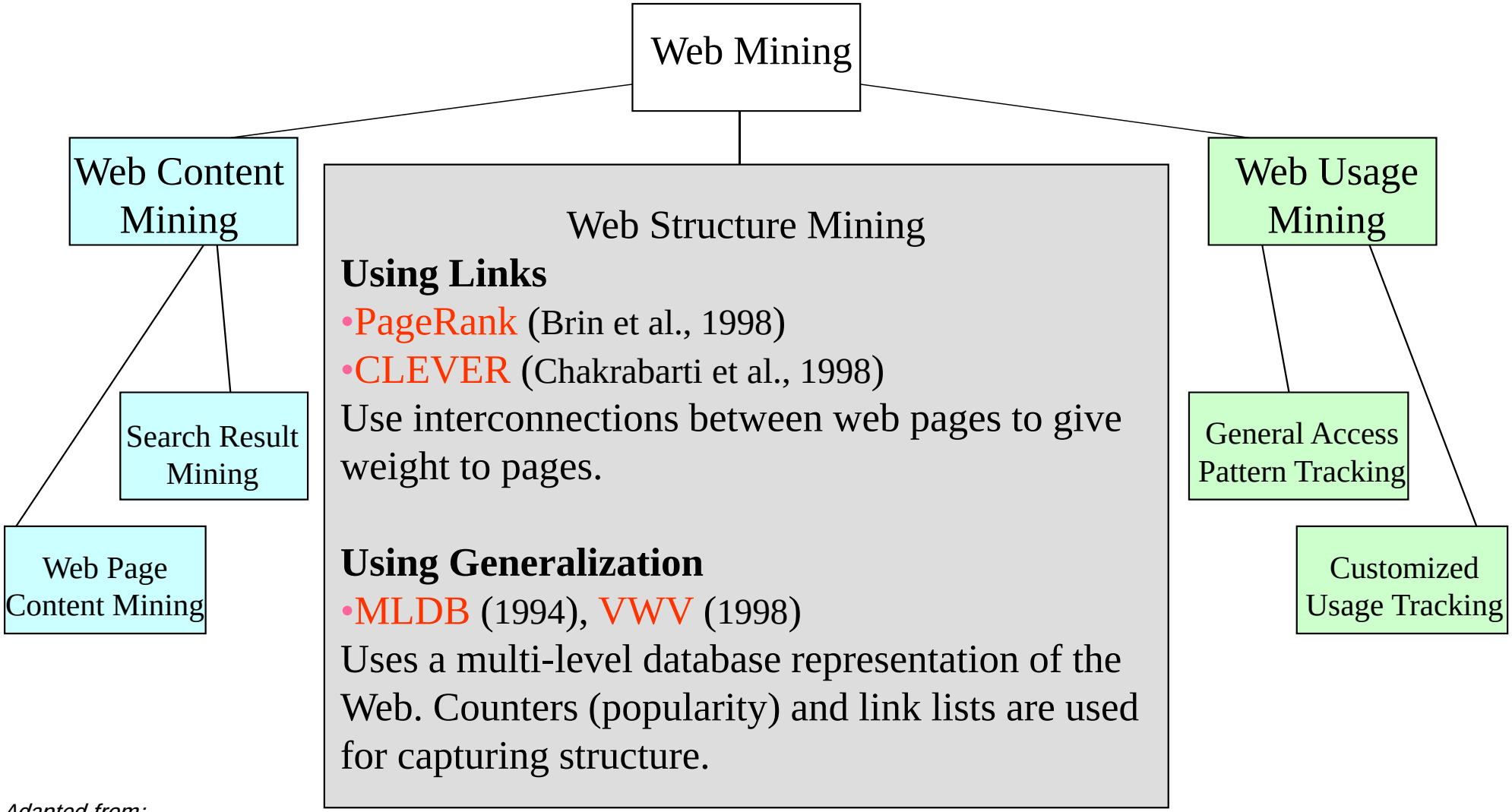
Mining the World-Wide Web



Adapted from:

Han, Kamber - Data Mining: Concepts and Techniques

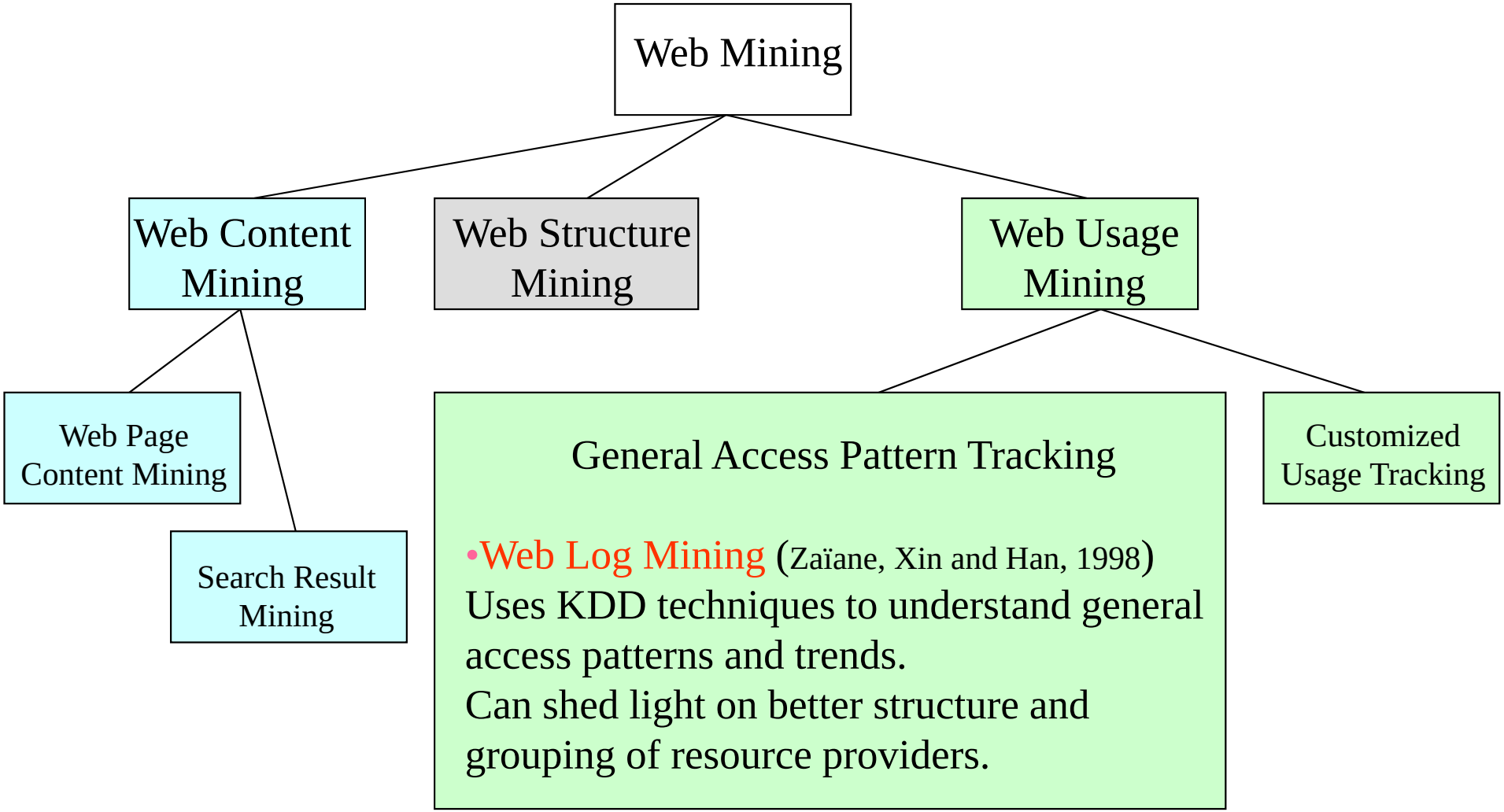
Mining the World-Wide Web



Adapted from:

Han, Kamber - Data Mining: Concepts and Techniques

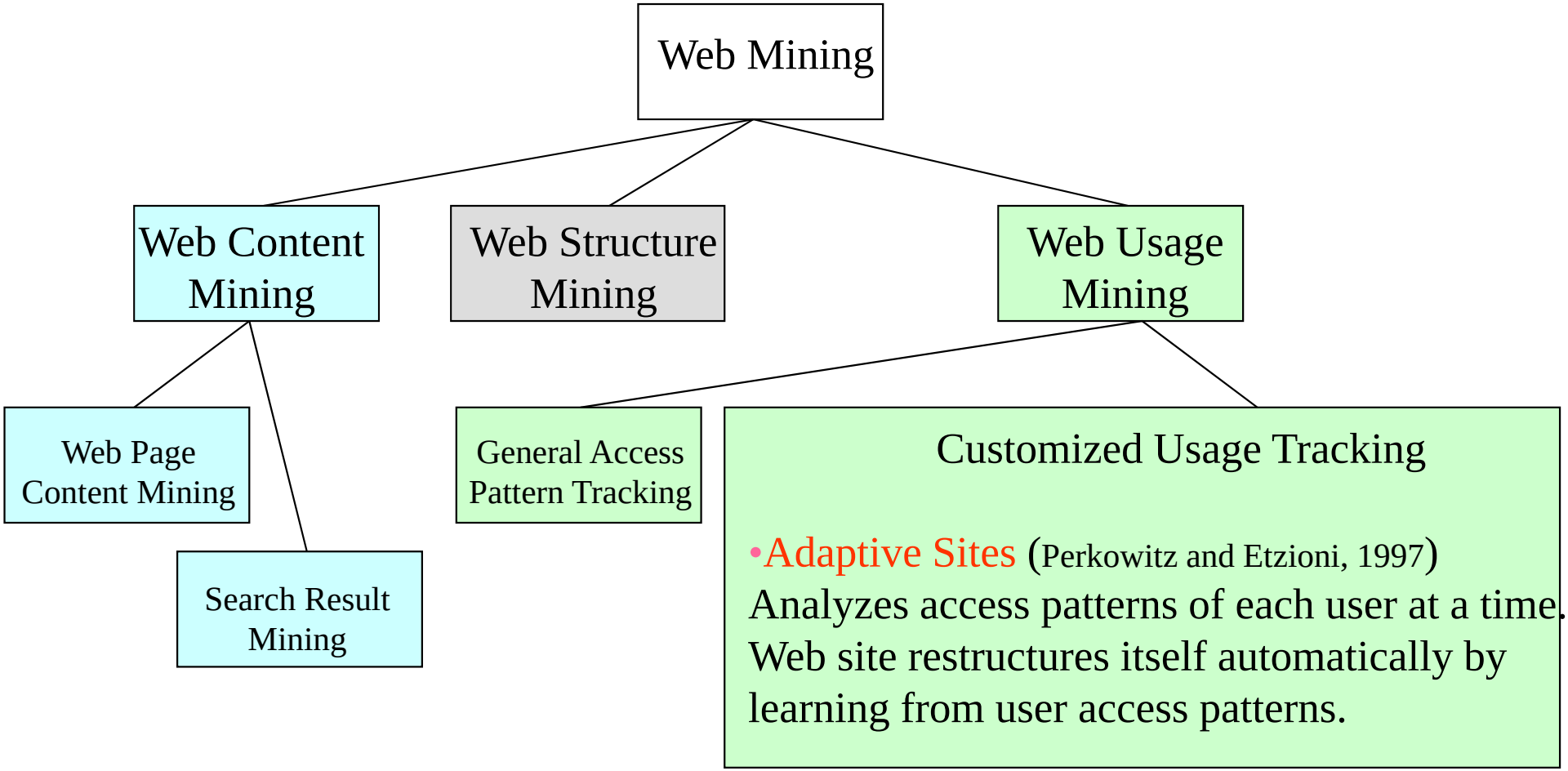
Mining the World-Wide Web



Adapted from:

Han, Kamber - Data Mining: Concepts and Techniques

Mining the World-Wide Web



Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Web Usage Mining

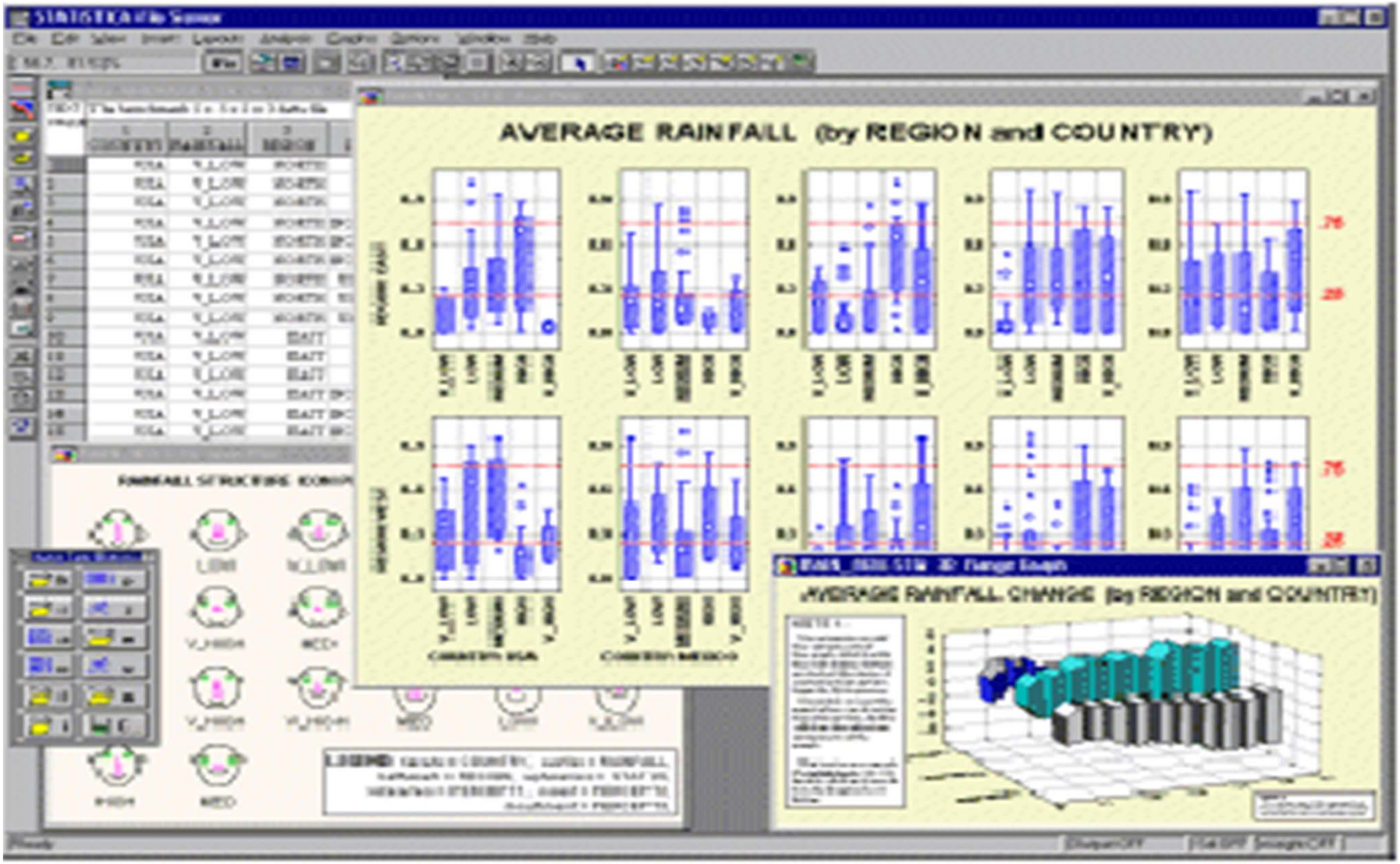
- Mining Web log records to discover user access patterns of Web pages
- Applications
  - Target potential customers for electronic commerce
  - Enhance the quality and delivery of Internet information services to the end user
  - Improve Web server system performance
  - Identify potential prime advertisement locations
- Web logs provide rich information about Web dynamics
  - Typical Web log entry includes the URL requested, the IP address from which the request originated, and a timestamp

Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Others

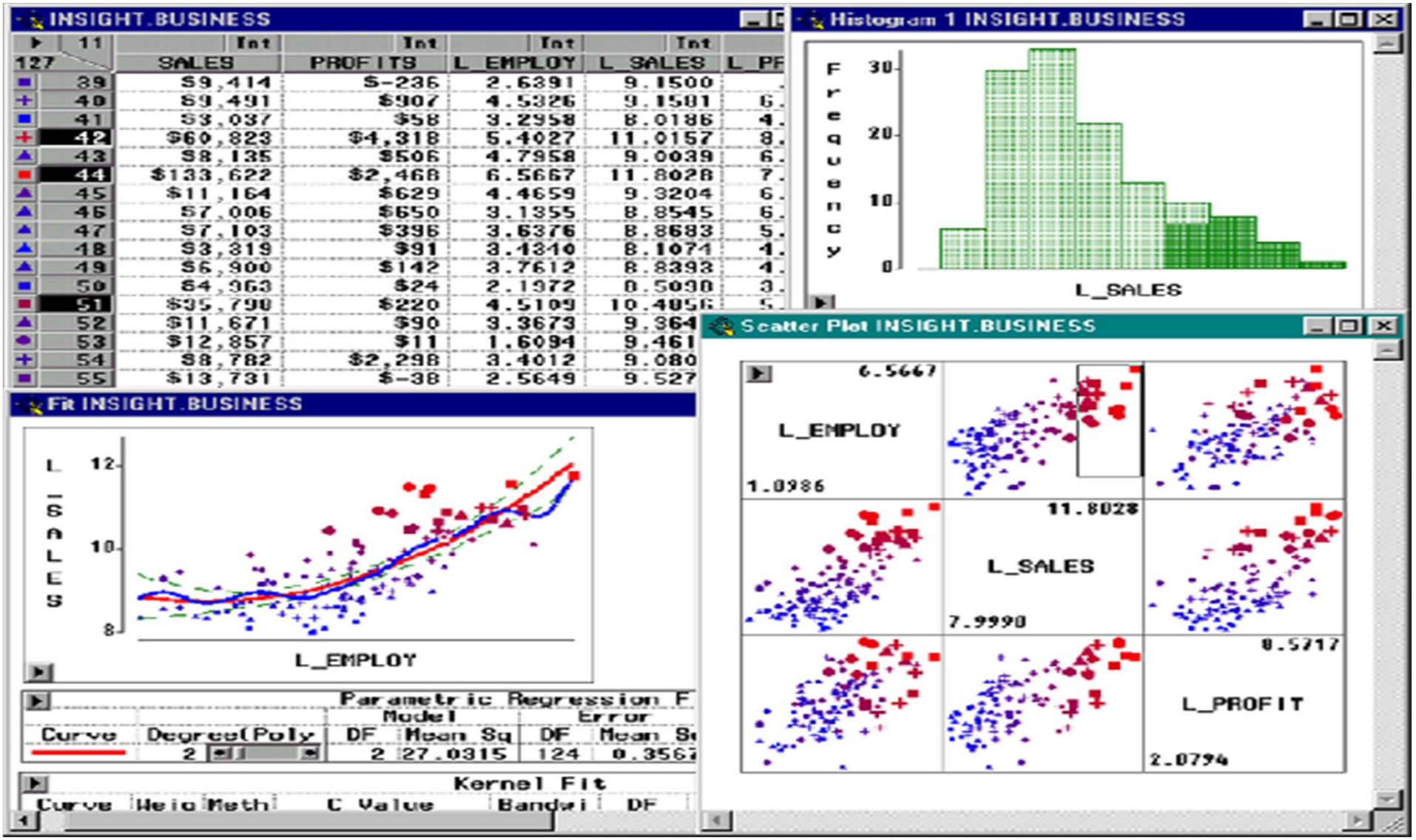
Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Boxplots from Statsoft: Multiple Variable Combinations



Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

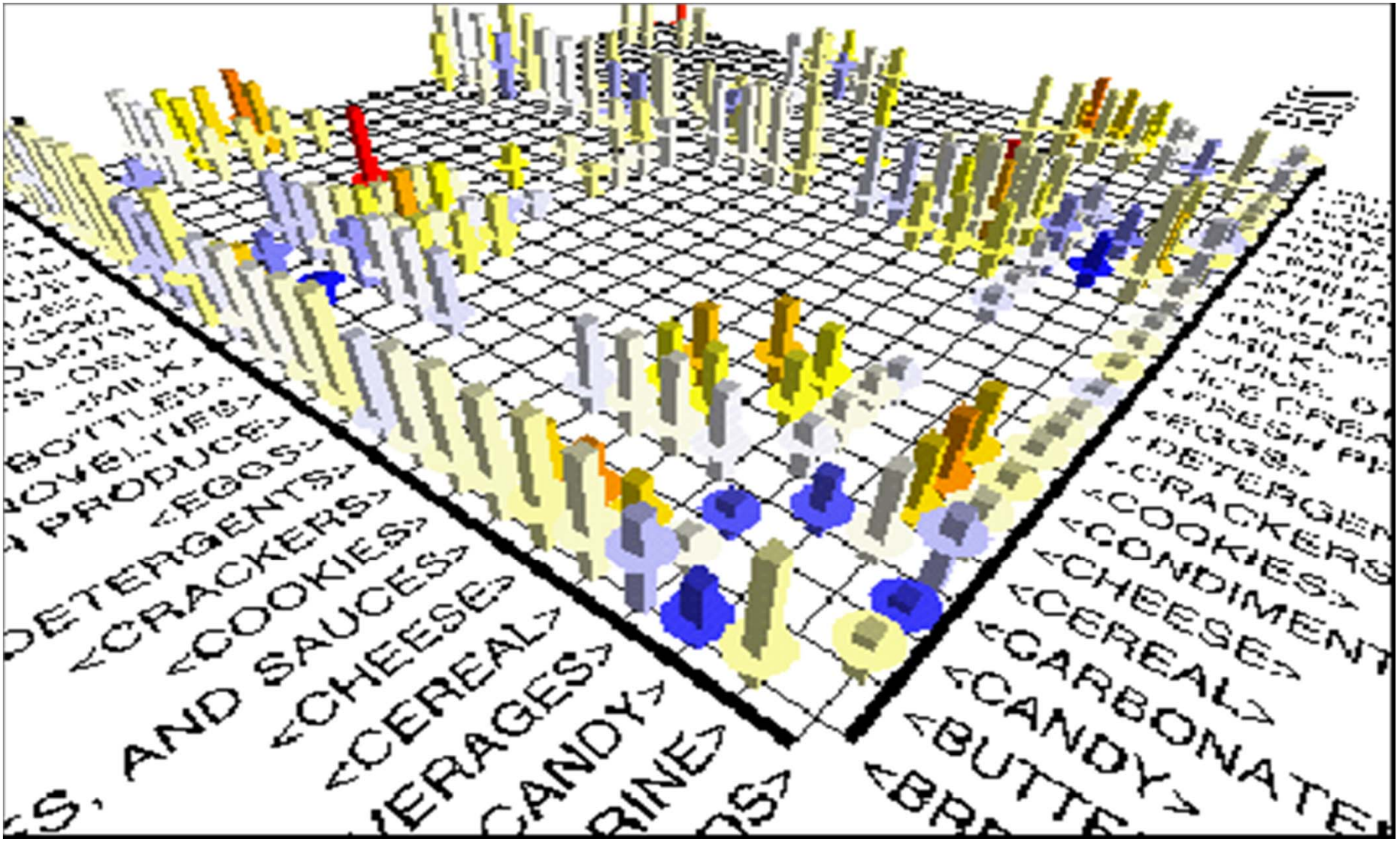
Visualization of Data Mining Results in SAS Enterprise Miner: Scatter Plots



Adapted from:

Han, Kamber - Data Mining: Concepts and Techniques

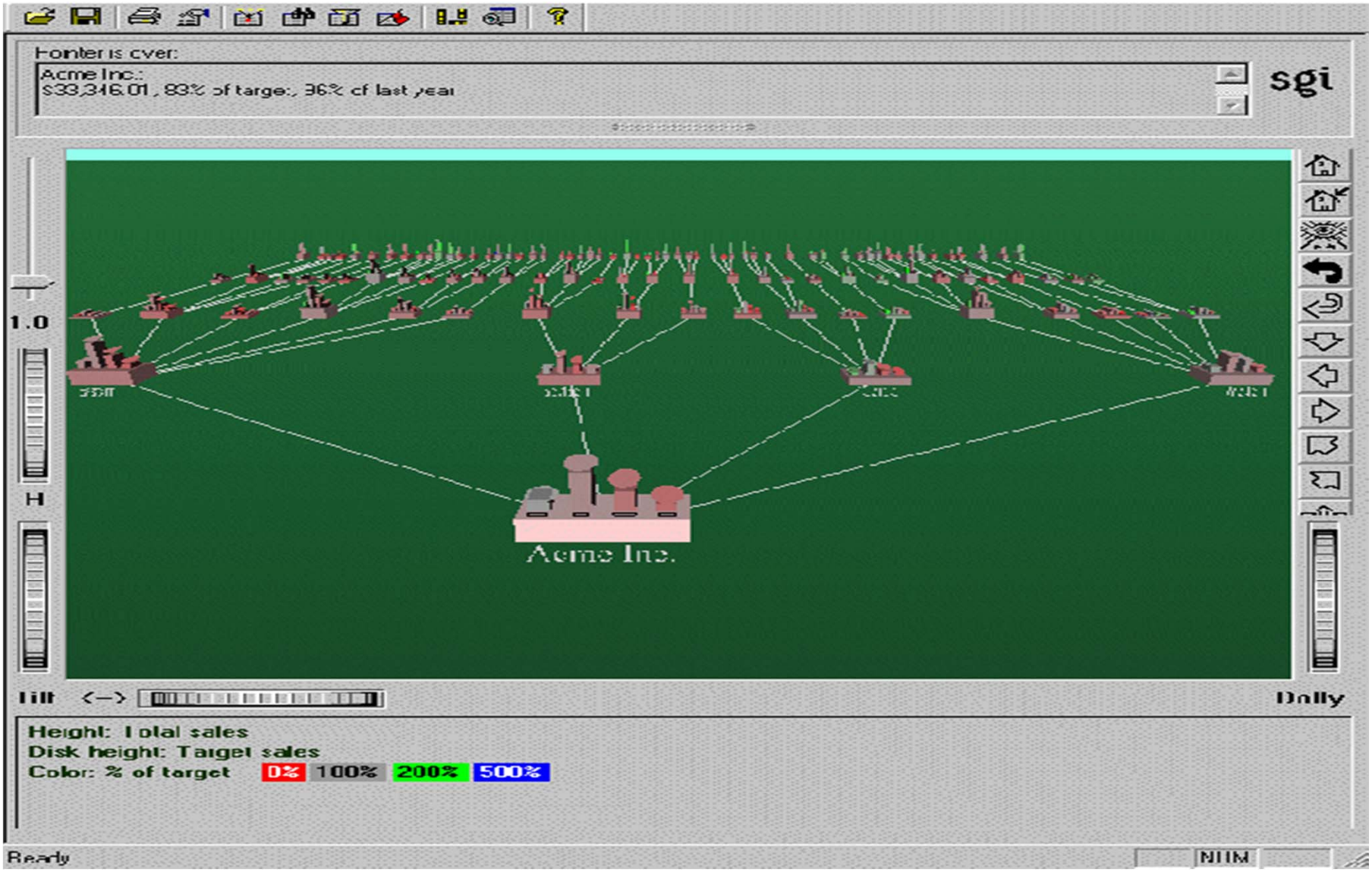
Visualization of Association Rules in SGI/MineSet 3.0



Adapted from:

Han, Kamber - Data Mining: Concepts and Techniques

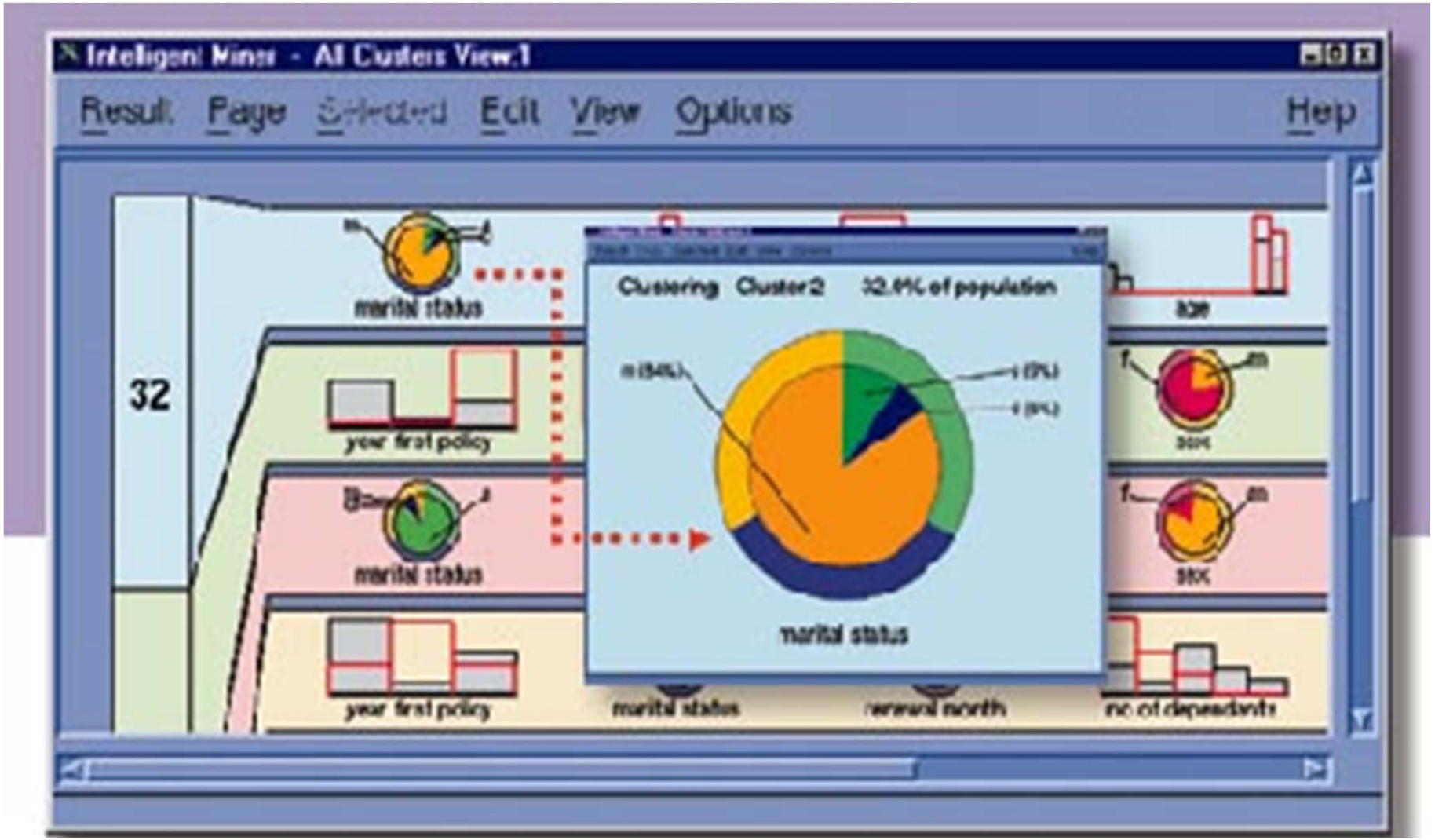
Visualization of a Decision Tree in SGI/MineSet 3.0



Adapted from:

Han, Kamber - Data Mining: Concepts and Techniques

Visualization of Cluster Grouping in IBM Intelligent Miner



Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Audio Data Mining

- Uses audio signals to indicate the patterns of data or the features of data mining results
- An interesting alternative to visual mining
- An inverse task of mining audio (such as music) databases which is to find patterns from audio data
- Visual data mining may disclose interesting patterns using graphical displays, but requires users to concentrate on watching patterns
- Instead, transform patterns into sound and music and listen to pitches, rhythms, tune, and melody in order to identify anything interesting or unusual

Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

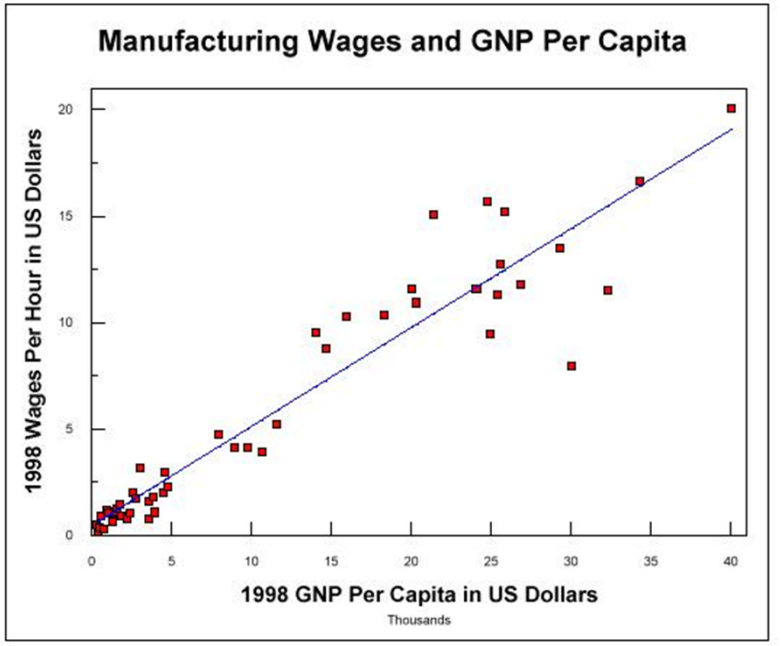
Scientific and Statistical Data Mining

- There are many well-established statistical techniques for data analysis, particularly for numeric data
  - applied extensively to data from scientific experiments and data from economics and the social sciences

- **Regression**

  - predict the value of a response (dependent) variable from one or more predictor (independent) variables where the variables are numeric

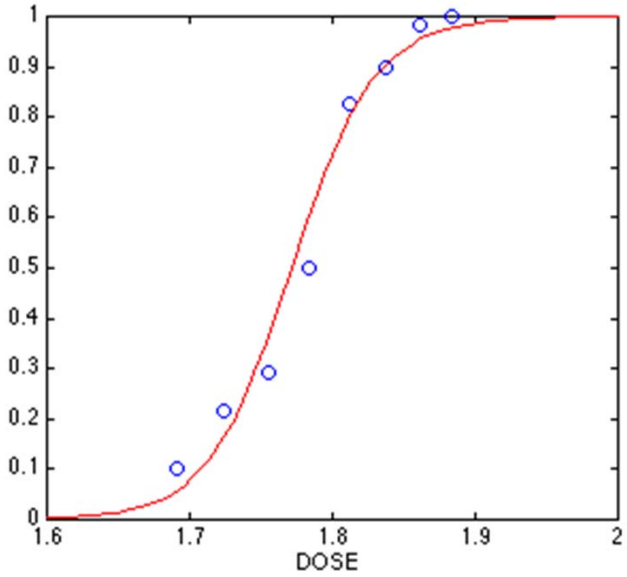
  - forms of regression: linear, multiple, weighted, polynomial, etc.



Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Scientific and Statistical Data Mining

- **Generalized linear models**
  - allow a categorical response variable (or some transformation of it) to be related to a set of predictor variables
  - similar to the modeling of a numeric response variable using linear regression
  - include logistic regression and Poisson regression



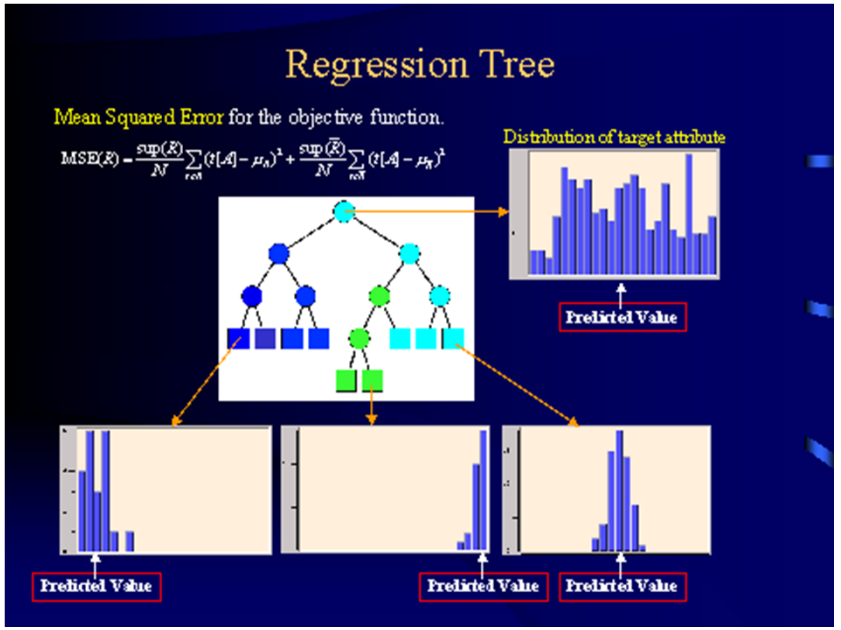
- **Mixed-effect models**
  - For analyzing grouped data, i.e. data that can be classified according to one or more grouping variables
  - Typically describe relationships between a response variable and some covariates in data grouped according to one or more factors

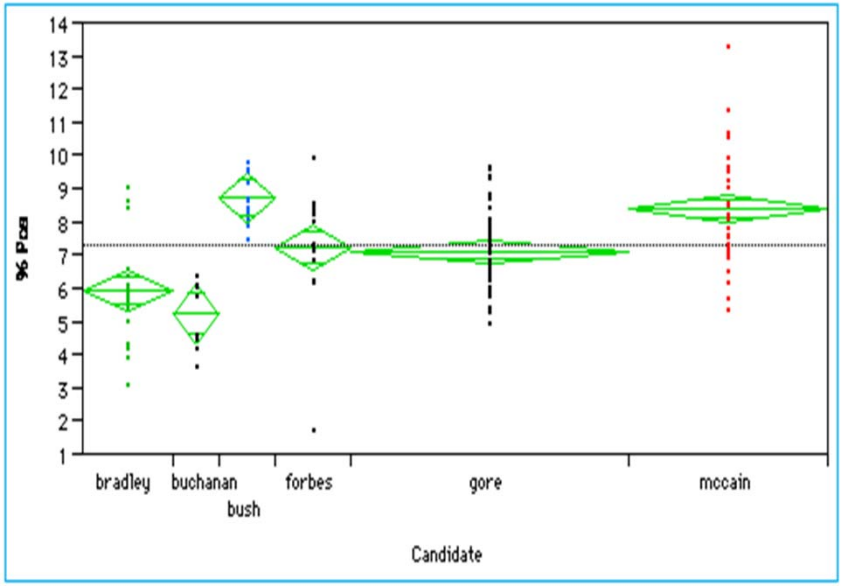
Adapted from:

Han, Kamber - Data Mining: Concepts and Techniques

Scientific and Statistical Data Mining

- **Regression trees**
  - Binary trees used for classification and prediction
  - Similar to decision trees:Tests are performed at the internal nodes
  - In a regression tree the mean of the objective attribute is computed and used as the predicted value
- **Analysis of variance**
  - Analyze experimental data for two or more populations described by a numeric response variable and one or more categorical variables (factors)

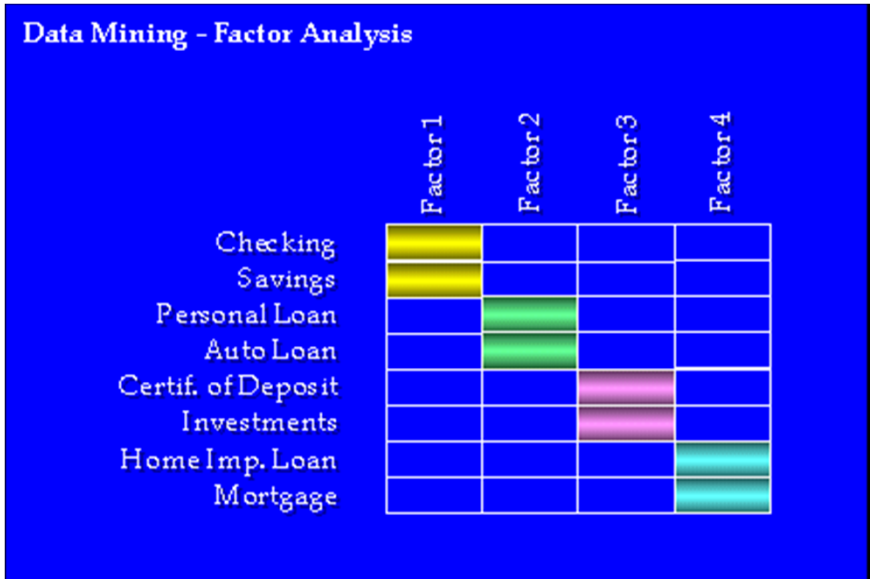


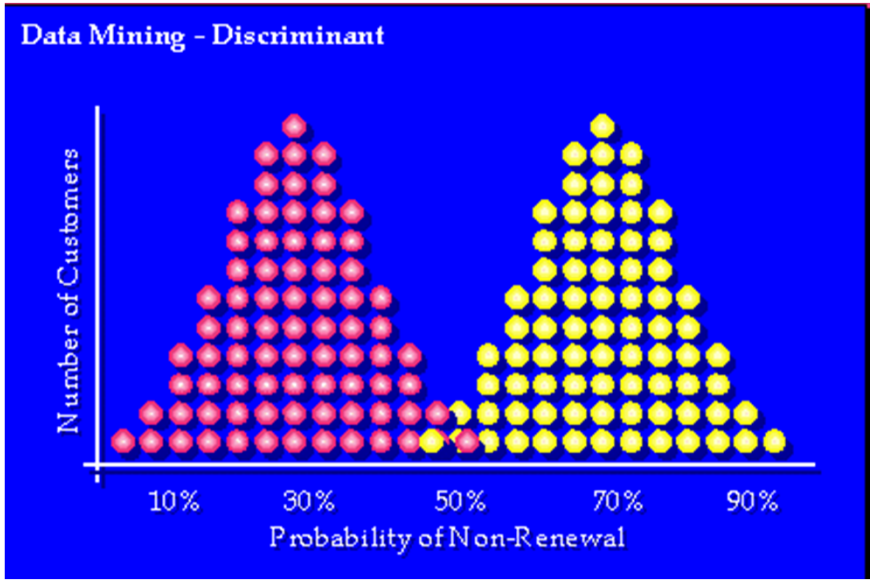


Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Scientific and Statistical Data Mining

- **Factor analysis**
  - determine which variables are combined to generate a given factor
  - e.g., for many psychiatric data, one can indirectly measure other quantities (such as test scores) that reflect the factor of interest
- **Discriminant analysis**
  - predict a categorical response variable, commonly used in social science
  - Attempts to determine several discriminant functions (linear combinations of the independent variables) that discriminate among the groups defined by the response variable





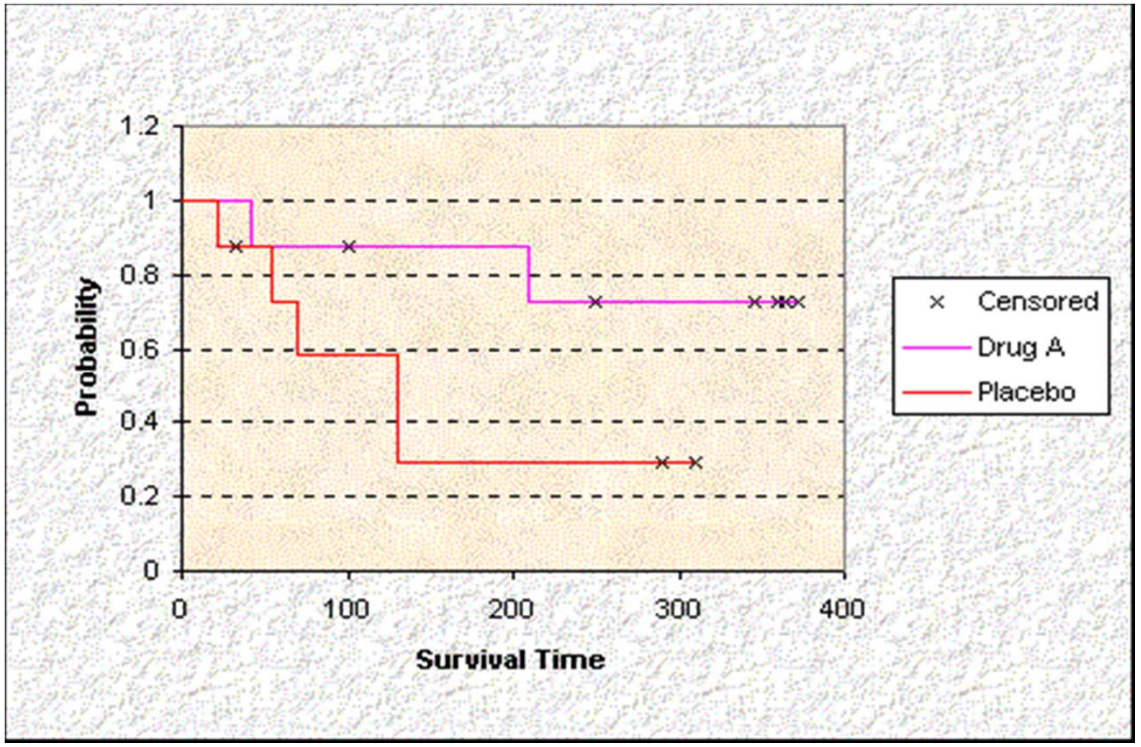
Adapted from:

Han, Kamber - Data Mining: Concepts and Techniques

Scientific and Statistical Data Mining

- **Time series**: many methods such as autoregression, ARIMA (Autoregressive integrated moving-average modeling), long memory time-series modeling
- **Quality control:** displays group summary charts

- **Survival analysis**
  - predicts the probability that a patient undergoing a medical treatment would survive at least to time *t* (life span prediction)



Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Data Mining: Merely Managers' Business or Everyone's?

Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

Social Impacts: Threat to Privacy and Data Security?

- Is data mining a threat to privacy and data security?
  - "Big Brother", "Big Banker", and "Big Business" are carefully watching you
  - Profiling information is collected every time
    - Credit card, debit card, supermarket loyalty card, or frequent flyer card, or apply for any of the above
    - You surf the Web, rent a video, fill out a contest entry form,
    - You pay for prescription drugs, or present you medical care number when visiting the doctor
  - Collection of personal data may be beneficial for companies and consumers, there is also potential for misuse
    - Medical Records, Employee Evaluations, Etc.

Protect Privacy and Data Security

- Fair information practices
  - International guidelines for data privacy protection
  - Cover aspects relating to data collection, purpose, use, quality, openness, individual participation, and accountability
  - Purpose specification and use limitation
  - Openness: Individuals have the right to know what information is collected about them, who has access to the data, and how the data are being used

공정정보규정원칙	내용
공지, 인식	데이터 수집 대상에게 데이터를 수집하는 것에 대한 공지를 해야한다.
선택, 동의	데이터 수집 대상이 자신의 데이터가 이차적으로 사용되는 것을 선택하고 동의할 수 있어야 한다.
접근, 참가	데이터 수집 대상이 수집된 데이터에 쉽게 접근할 수 있어야 한다.
보안	데이터 수집 기관은 데이터에 대한 보안을 책임져야 한다.
시행	공정정보규정원칙을 시행하기 위한 여러 법률과 규정이 필요하다.

Adapted from:

Han, Kamber - Data Mining: Concepts and Techniques

Data Mining in Construction

- Application exploration
  - development of application-specific data mining system
- Scalable data mining methods
  - Constraint-based mining: use of constraints to guide data mining systems in their search for interesting patterns
- Integration of data mining with database systems, data warehouse systems, and Web database systems
- Invisible data mining (mining as built-in function)

Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

The Future of your Discipline

"UC Berkeley's Prof. Nicholas Sitar has also noticed that some outstanding civil engineering graduates are going into jobs in areas such as data mining and risk analysis."

http://www.graduatingengineer.com/futuredisc/civil2.html

Adapted from:
Han, Kamber - Data Mining: Concepts and Techniques

DATABASE SYSTEMS GROUP

Ludwig-Maximilians-Universität München
Institut für Informatik
Lehr- und Forschungseinheit für Datenbanksysteme

LMU

Knowledge Discovery in Databases

SS 2016

Chapter 3: Frequent Itemset Mining

Lecture: Prof. Dr. Thomas Seidl

Tutorials: Julian Busch, Evgeniy Faerman,
Florian Richter, Klaus Schmid

Chapter 3: Frequent Itemset Mining

1) Introduction
   - Transaction databases, market basket data analysis
2) Mining Frequent Itemsets
   - Apriori algorithm, hash trees, FP-tree
3) Simple Association Rules
   - Basic notions, rule generation, interestingness measures
4) Further Topics
5) Extensions and Summary

What is Frequent Itemset Mining?

Frequent Itemset Mining:

Finding frequent patterns, associations, correlations, or causal structures among sets of items or objects in transaction databases, relational databases, and other information repositories.

- Given:
  - A set of items $I = \{i_1, i_2, \ldots, i_m\}$
  - A database of transactions D, where a transaction $T \subseteq I$ is a set of items
- Task 1: find all subsets of items that occur together in many transactions.
  - E.g.: 85% of transactions contain the itemset {milk, bread, butter}
- Task 2: find all rules that correlate the presence of one set of items with that of another set of items in the transaction database.
  - E.g.: 98% of people buying tires and auto accessories also get automotive service done
- Applications: Basket data analysis, cross-marketing, catalog design, loss-leader analysis, clustering, classification, recommendation systems, etc.

Example: Basket Data Analysis

- Transaction database

  D= {{butter, bread, milk, sugar};
  {butter, flour, milk, sugar};
  {butter, eggs, milk, salt};
  {eggs};
  {butter, flour, milk, salt, sugar}}

- Question of interest:
  - Which items are bought together frequently?

items	frequency
{butter}	4
{milk}	4
{butter, milk}	4
{sugar}	3
{butter, sugar}	3
{milk, sugar}	3
{butter, milk, sugar}	3
{eggs}	2
…	

- Applications
  - Improved store layout
  - Cross marketing
  - Focused attached mailings / add-on sales
  - $* \Rightarrow$ *Maintenance Agreement*
    (What the store should do to boost Maintenance Agreement sales)
  - *Home Electronics* $\Rightarrow *$ (What other products should the store stock up?)

Chapter 3: Frequent Itemset Mining

1) Introduction
   - Transaction databases, market basket data analysis
2) Mining Frequent Itemsets
   - Apriori algorithm, hash trees, FP-tree
3) Simple Association Rules
   - Basic notions, rule generation, interestingness measures
4) Further Topics
   - Hierarchical Association Rules
     - Motivation, notions, algorithms, interestingness
   - Quantitative Association Rules
     - Motivation, basic idea, partitioning numerical attributes, adaptation of apriori algorithm, interestingness
5) Extensions and Summary

Mining Frequent Itemsets: Basic Notions

- *Items* $I = \{i_1, i_2, \ldots, i_m\}$: a set of literals (denoting items)
- *Itemset* X*:* Set of items $X \subseteq I$
- *Database* D: Set of *transactions* T, each transaction is a set of items $T \subseteq I$
- Transaction T *contains* an itemset X*:* $X \subseteq T$
- The items in transactions and itemsets are sorted lexicographically:
  - itemset $X = (x_1, x_2, \ldots, x_k)$, where $x_1 \leq x_2 \leq \ldots \leq x_k$
- *Length* of an itemset: number of elements in the itemset
- *k-itemset:* itemset of length k
- The *support* of an itemset X is defined as: $support(X) = |\{T \in D | X \subseteq T\}|$
- *Frequent itemset: an itemset* X is called frequent for database D iff it is contained in more than $minSup$ many transactions: $support(X) \geq minSup$

- Goal 1: Given a database $D$and a threshold $minSup$, find all frequent itemsets $X \in Pot(I)$.

Mining Frequent Itemsets: Basic Idea

- Naïve Algorithm
  - count the frequency of all possible subsets of I in the database

  → *too expensive* since there are 2^m such itemsets for $|I| = m$ items

  cardinality of power set
- The *Apriori* principle (anti-monotonicity):

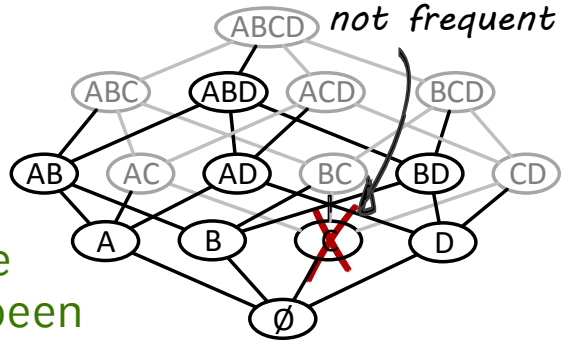
  Any non-empty subset of a frequent itemset is frequent, too!

  $A \subseteq I$ with $\text{support}(A) \geq \text{minSup} \Rightarrow \forall A' \subset A \wedge A' \neq \emptyset: \text{support}(A') \geq \text{minSup}$

  Any superset of a non-frequent itemset is non-frequent, too!

  $A \subseteq I$ with $\text{support}(A) < \text{minSup} \Rightarrow \forall A' \supset A: \text{support}(A') < \text{minSup}$

- Method based on the apriori principle
  - First count the 1-itemsets, then the 2-itemsets, then the 3-itemsets, and so on
  - When counting (k+1)-itemsets, only consider those (k+1)-itemsets where all subsets of length k have been determined as frequent in the previous step



The Apriori Algorithm

variable C_k: candidate itemsets of size k
variable L_k: frequent itemsets of size k

L_1 = {frequent items}
for ($k = 1$; L_k !=∅; k++) **do begin**
    // JOIN STEP: join L_k with itself to produce C_{k+1}
    // PRUNE STEP: discard (k+1)-itemsets from C_{k+1} that contain non-frequent k-itemsets as subsets
    C_{k+1} = candidates generated from L_k

produce candidates

    for each transaction t in database do
        Increment the count of all candidates in C_{k+1} that are contained in t
    L_{k+1} = candidates in C_{k+1} with min_support

prove candidates

return $\cup_k L_k$

Generating Candidates (Join Step)

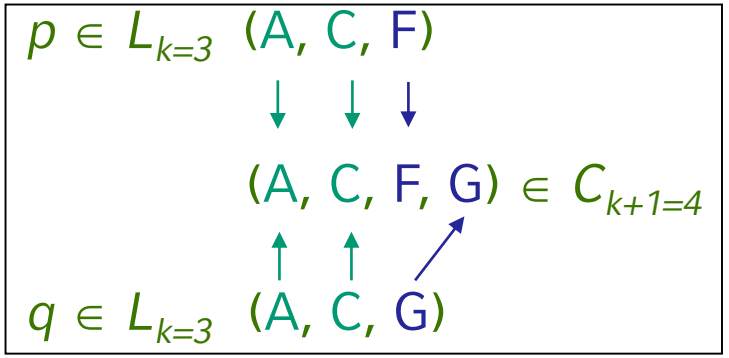
- Requirements for set of all candidate $(k+1)$-itemsets C_{k+1}
  - *Completeness*:
    Must contain all frequent $(k+1)$-itemsets (superset property $C_{k+1} \supseteq L_{k+1}$)
  - *Selectiveness*:
    Significantly smaller than the set of all $(k+1)$-subsets
  - Suppose the items are sorted by any order (e.g., lexicograph.)
- Step 1: Joining ($C_{k+1} = L_k \bowtie L_k$)
  - Consider frequent k-itemsets p and q
  - p and q are joined if they share the same first $k-1$ items

insert into C_{k+1}

select $p.i_1, p.i_2, \ldots, p.i_{k-1}, p.i_k, q.i_k$

from $L_k : p, L_k : q$

where $p.i_1 = q.i_1, \ldots, p.i_{k-1} = q.i_{k-1}, p.i_k < q.i_k$



Generating Candidates (Prune Step)

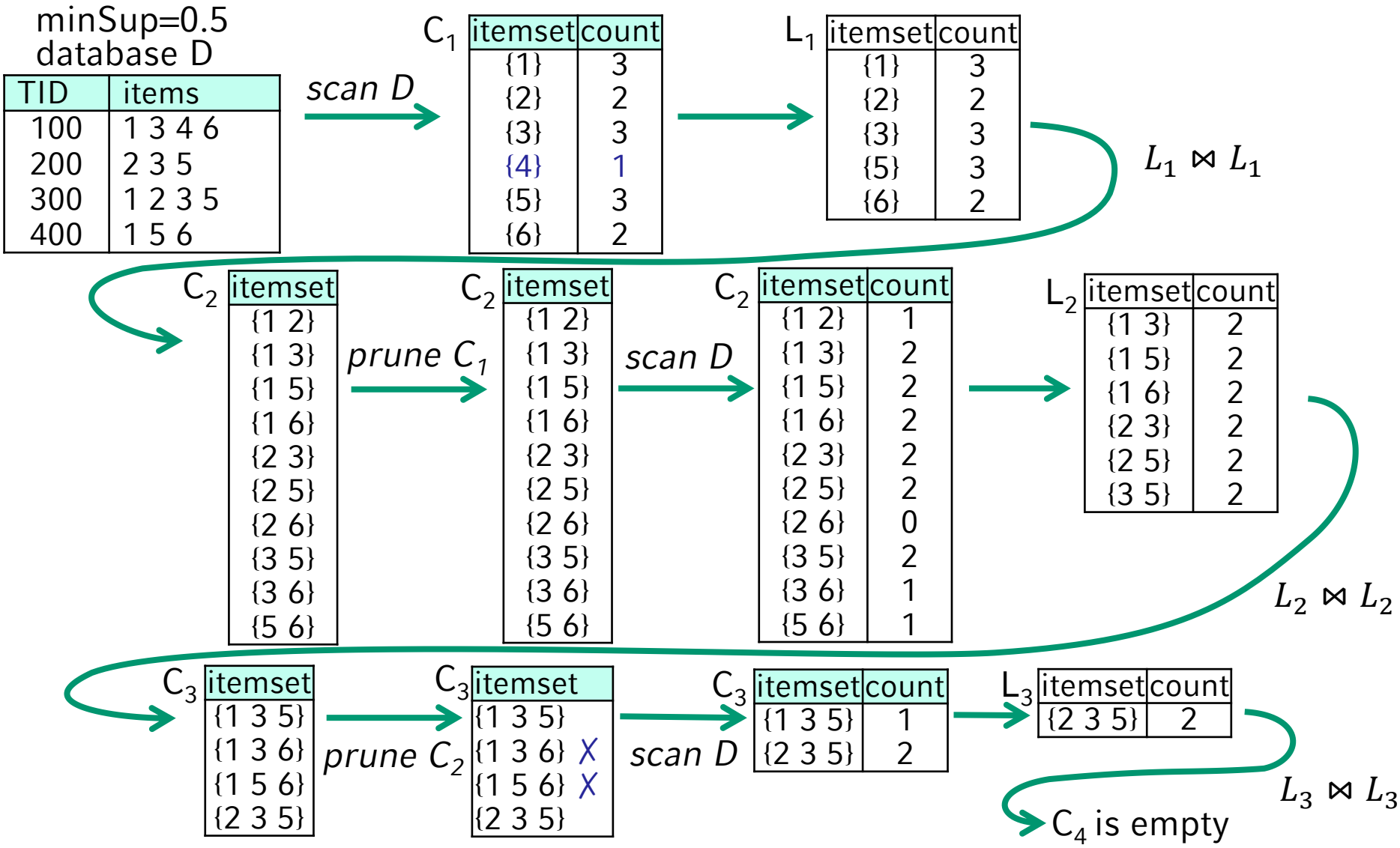
- Step 2: Pruning ($L_{k+1} = \{X \in C_{k+1} | support(X) \geq minSup\}$)
  - Naïve: Check support of every itemset in C_{k+1} ← inefficient for huge C_{k+1}
  - Instead, apply Apriori principle first: Remove candidate *(k+1)* -itemsets which contain a non-frequent *k* -subset *s*, i.e., s $\notin L_k$

```
forall itemsets c in C_{k+1} do
    forall k-subsets s of c do
        if (s is not in L_k) then delete c from C_{k+1}
```

- Example 1
  - L_3 = {(ACF), (ACG), (AFG), (AFH), (CFG)}
  - Candidates after the join step: {(ACFG), (AFGH)}
  - In the pruning step: delete (AFGH) because (FGH) $\notin L_3$, i.e., (FGH) is not a frequent 3-itemset; also (AGH) $\notin L_3$

→ C_4 = {(ACFG)} → check the support to generate L_4

Apriori Algorithm – Full Example

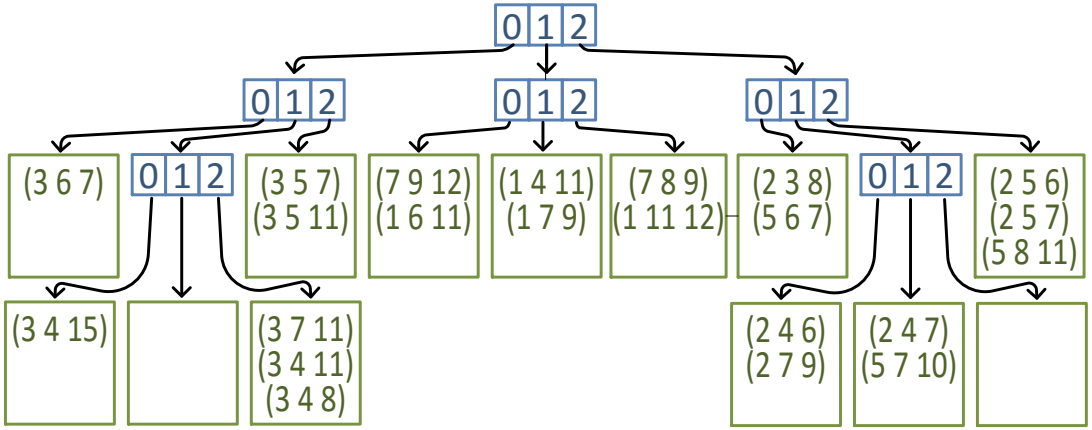


How to Count Supports of Candidates?

- Why is counting supports of candidates a problem?
  - The total number of candidates can be very huge
  - One transaction may contain many candidates
- Method: Hash-Tree
  - Candidate itemsets are stored in a hash-tree
  - Leaf nodes of hash-tree contain lists of itemsets and their support (i.e., counts)
  - Interior nodes contain hash tables
  - Subset function finds all the candidates contained in a transaction

e.g. for 3-Itemsets

$h(K) = K \bmod 3$

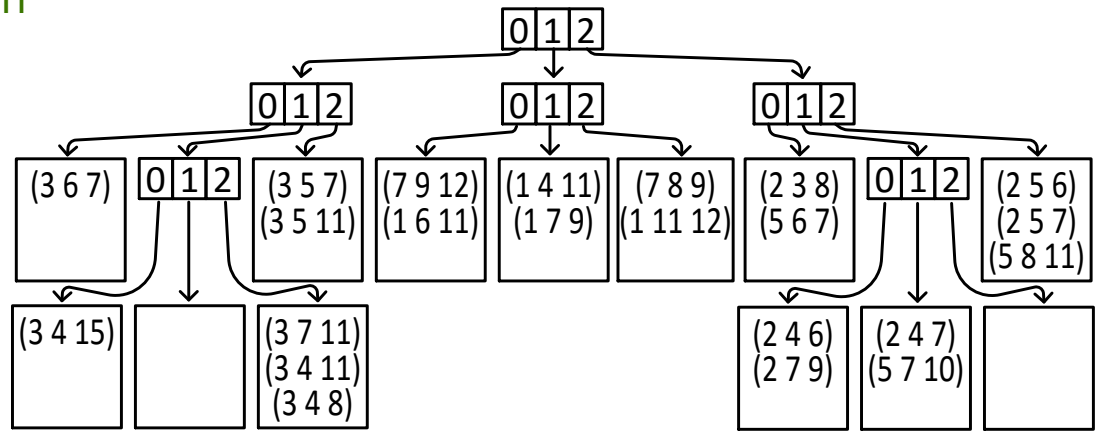


Hash-Tree – Construction

- Searching for an itemset
  - Start at the root (level 1)
  - At level *d:* apply the hash function *h* to the *d*-th item in the itemset
- Insertion of an itemset
  - search for the corresponding leaf node, and insert the itemset into that leaf
  - if an overflow occurs:
    - Transform the leaf node into an internal node
    - Distribute the entries to the new leaf nodes according to the hash function

for 3-Itemsets

$h(K) = K \bmod 3$



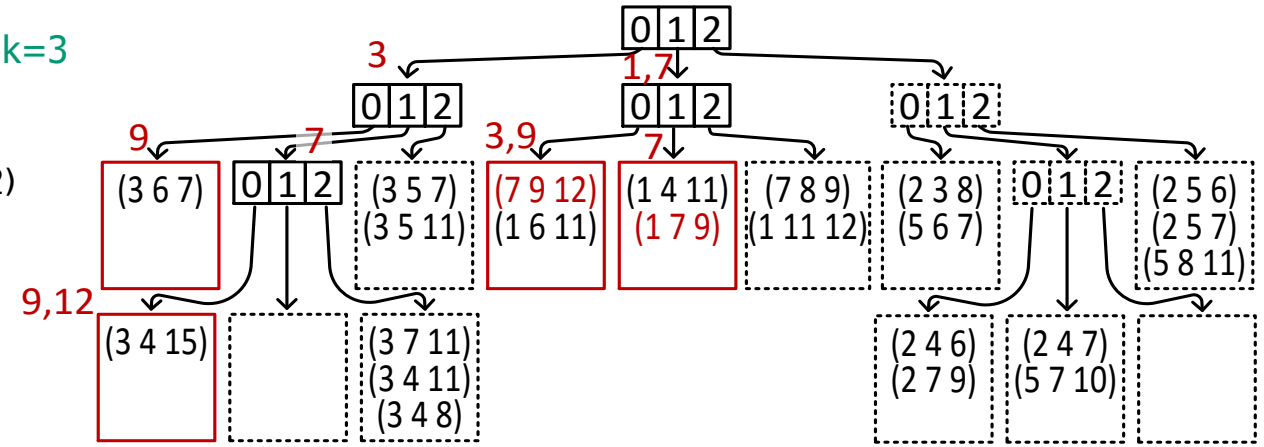
Hash-Tree – Counting

- Search all candidate itemsets contained in a transaction $T = (t_1\ t_2\ \ldots\ t_n)$ for a current itemset length of k
- At the root
  - Determine the hash values for each item $t_1\ t_2\ \ldots\ t_{n-k+1}$ in T
  - Continue the search in the resulting child nodes
- At an internal node at level d (reached after hashing of item t_i)
  - Determine the hash values and continue the search for each item t_j with $i < j \leq n - k + d$
- At a leaf node
  - Check whether the itemsets in the leaf node are contained in transaction T

in our example n=5 and k=3

$h(K) = K \bmod 3$

Transaction (1, 3, 7, 9, 12)



Tested leaf nodes

Pruned subtrees

Is Apriori Fast Enough? — Performance Bottlenecks

- The core of the Apriori algorithm:
  - Use frequent (k – 1)-itemsets to generate candidate frequent k-itemsets
  - Use database scan and pattern matching to collect counts for the candidate itemsets
- The bottleneck of *Apriori*: candidate generation
  - Huge candidate sets:
    - 10^4 frequent 1-itemsets will generate 10^7 candidate 2-itemsets
    - To discover a frequent pattern of size 100, e.g., $\{a_1, a_2, \ldots, a_{100}\}$, one needs to generate $2^{100} \approx 10^{30}$ candidates.
  - Multiple scans of database:
    - Needs n or n+1 scans, n is the length of the longest pattern

→ Is it possible to mine the complete set of frequent itemsets without candidate generation?

Mining Frequent Patterns Without Candidate Generation

- Compress a large database into a compact, *Frequent-Pattern tree* (*FP-tree*) structure
  - highly condensed, but complete for frequent pattern mining
  - avoid costly database scans

- Develop an efficient, FP-tree-based frequent pattern mining method
  - A divide-and-conquer methodology: decompose mining tasks into smaller ones
  - Avoid candidate generation: sub-database test only!

- Idea:
  - Compress database into FP-tree, retaining the itemset association information
  - Divide the compressed database into conditional databases, each associated with one frequent item and mine each such database separately.

Construct FP-tree from a Transaction DB

Steps for compressing the database into a FP-tree:

1. Scan DB once, find frequent 1-itemsets (single items)
2. Order frequent items in frequency descending order

TID	items bought
100	{*f, a, c, d, g, i, m, p*}
200	{*a, b, c, f, l, m, o*}
300	{*b, f, h, j, o*}
400	{*b, c, k, s, p*}
500	{*a, f, c, e, l, p, m, n*}

1&2

header table:

item	frequency
f	4
c	4
a	3
b	3
m	3
p	3

minSup=0.5

sort items in the order of descending support

Construct FP-tree from a Transaction DB

Steps for compressing the database into a FP-tree:

1. Scan DB once, find frequent 1-itemsets (single items)
2. Order frequent items in frequency descending order
3. Scan DB again, construct FP-tree starting with most frequent item per transaction

TID	items bought	(ordered) frequent items
100	{*f, a, c, d, g, i, m, p*}	{*f, c, a, m, p*}
200	{*a, b, c, f, l, m, o*}	{*f, c, a, b, m*}
300	{*b, f, h, j, o*}	{*f, b*}
400	{*b, c, k, s, p*}	{*c, b, p*}
500	{*a, f, c, e, l, p, m, n*}	{*f, c, a, m, p*}

for each transaction only keep its frequent items sorted in descending order of their frequencies

1&2

header table:

item	frequency
f	4
c	4
a	3
b	3
m	3
p	3

3a

for each transaction build a path in the FP-tree:

- If a path with common prefix exists: increment frequency of nodes on this path and append suffix
- Otherwise: create a new branch

Construct FP-tree from a Transaction DB

Steps for compressing the database into a FP-tree:

1. Scan DB once, find frequent 1-itemsets (single items)
2. Order frequent items in frequency descending order
3. Scan DB again, construct FP-tree starting with most frequent item per transaction

TID	items bought	(ordered) frequent items
100	{f, a, c, d, g, i, m, p}	{f, c, a, m, p}
200	{a, b, c, f, l, m, o}	{f, c, a, b, m}
300	{b, f, h, j, o}	{f, b}
400	{b, c, k, s, p}	{c, b, p}
500	{a, f, c, e, l, p, m, n}	{f, c, a, m, p}

1&2

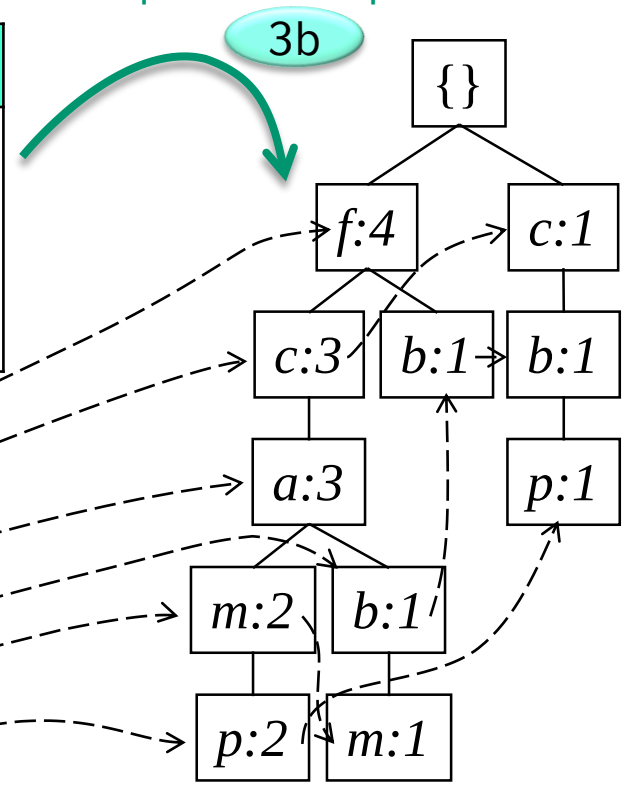
3a

3b

header table:

item	frequency	head
f	4	●
c	4	●
a	3	●
b	3	●
m	3	●
p	3	●

header table references the occurrences of the frequent items in the FP-tree



Benefits of the FP-tree Structure

- Completeness:
  - never breaks a long pattern of any transaction
  - preserves complete information for frequent pattern mining
- Compactness
  - reduce irrelevant information—infrequent items are gone
  - frequency descending ordering: more frequent items are more likely to be shared
  - never be larger than the original database (if not count node-links and counts)
  - Experiments demonstrate compression ratios over 100

Mining Frequent Patterns Using FP-tree

- General idea (divide-and-conquer)
  - Recursively grow frequent pattern path using the FP-tree
- Method
  - For each item, construct its conditional pattern-base (*prefix paths*), and then its conditional FP-tree
  - Repeat the process on each newly created conditional FP-tree …
  - …until the resulting FP-tree is empty, or it contains only one path (single path will generate all the combinations of its sub-paths, each of which is a frequent pattern)

Major Steps to Mine FP-tree

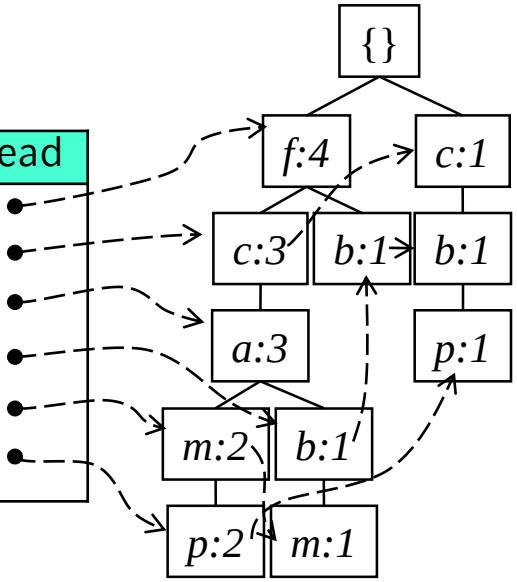
1) Construct conditional pattern base for each node in the FP-tree
2) Construct conditional FP-tree from each conditional pattern-base
3) Recursively mine conditional FP-trees and grow frequent patterns obtained so far
   - If the conditional FP-tree contains a single path, simply enumerate all the patterns

Major Steps to Mine FP-tree: Conditional Pattern Base

1) Construct conditional pattern base for each node in the FP-tree
   - Starting at the frequent header table in the FP-tree
   - Traverse FP-tree by following the link of each frequent item (dashed lines)
   - Accumulate all of transformed prefix paths of that item to form a conditional pattern base
     - For each item its prefixes are regarded as condition for it being a suffix. These prefixes form the conditional pattern base. The frequency of the prefixes can be read in the node of the item.

header table:

item	frequency	head
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	



conditional pattern base:

item	cond. pattern base
f	*{}*
c	*f:3, {}*
a	*fc:3*
b	*fca:1, f:1, c:1*
m	*fca:2, fcab:1*
p	*fcam:2, cb:1*

Properties of FP-tree for Conditional Pattern Bases

- Node-link property
  - For any frequent item a_i, all the possible frequent patterns that contain a_i can be obtained by following a_i's node-links, starting from a_i's head in the FP-tree header
- Prefix path property
  - To calculate the frequent patterns for a node a_i in a path P, only the prefix sub-path of a_i in P needs to be accumulated, and its frequency count should carry the same count as node a_i.

Major Steps to Mine FP-tree: Conditional FP-tree

1) Construct conditional pattern base for each node in the FP-tree
2) Construct conditional FP-tree from each conditional pattern-base
   - The prefix paths of a suffix represent the conditional basis.
     →They can be regarded as transactions of a database.
   - Those prefix paths whose support ≥ minSup, induce a conditional FP-tree
   - For each pattern-base
     - Accumulate the count for each item in the base
     - Construct the FP-tree for the frequent items of the pattern base

conditional pattern base:

item	cond. pattern base
f	*{}*
c	*f:3*
a	*fc:3*
b	*fca:1, f:1, c:1*
m	*fca:2, fcab:1*
p	*fcam:2, cb:1*

e.g. item m

item	frequency
f	3
c	3
a	3
b	1 X

m-conditional FP-tree

{}|m
|
f:3
|
c:3
|
a:3

Major Steps to Mine FP-tree: Conditional FP-tree

1) Construct conditional pattern base for each node in the FP-tree
2) Construct conditional FP-tree from each conditional pattern-base

conditional pattern base:

item	cond. pattern base
f	*{}*
c	*f:3*
a	*fc:3*
b	*fca:1, f:1, c:1*
m	*fca:2, fcab:1*
p	*fcam:2, cb:1*

{}|f = {}

{}|c
|
f:3

{}|a
|
f:3
|
c:3

{}|b = {}

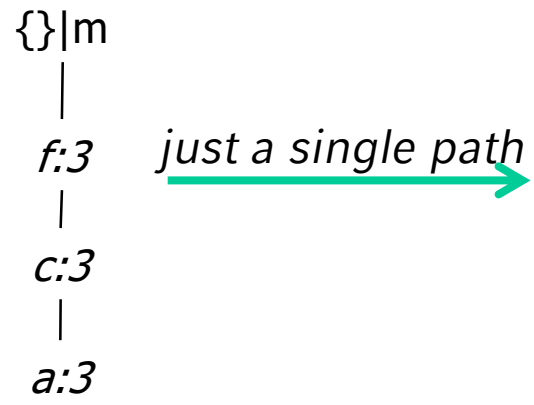
{}|m
|
f:3
|
c:3
|
a:3

{}|p
|
c:3

Major Steps to Mine FP-tree

1) Construct conditional pattern base for each node in the FP-tree
2) Construct conditional FP-tree from each conditional pattern-base
3) Recursively mine conditional FP-trees and grow frequent patterns obtained so far
   - If the conditional FP-tree contains a single path, simply enumerate all the patterns (enumerate all combinations of sub-paths)

example:
m-conditional FP-tree



All frequent patterns concerning *m*

m,
fm, cm, am,
fcm, fam, cam,
fcam

FP-tree: Full Example

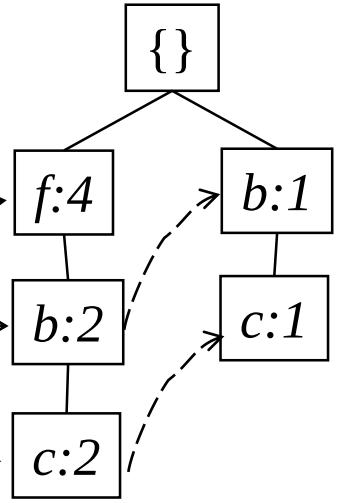
database:

TID	items bought	(ordered) frequent items
100	{*b*, *c*, *f*}	{*f*, *b*, *c*}
200	{*a*, *b*, *c*}	{*b*, *c*}
300	{*d*, *f*}	{*f*}
400	{*b*, *c*, *e*, *f*}	{*f*, *b*, *c*}
500	{*f*, *g*}	{*f*}

minSup=0.4

header table:

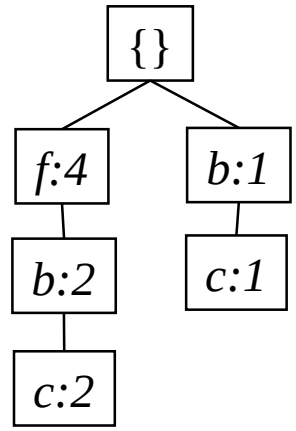
item	frequency	head
f	4	
b	3	
c	3	



conditional pattern base:

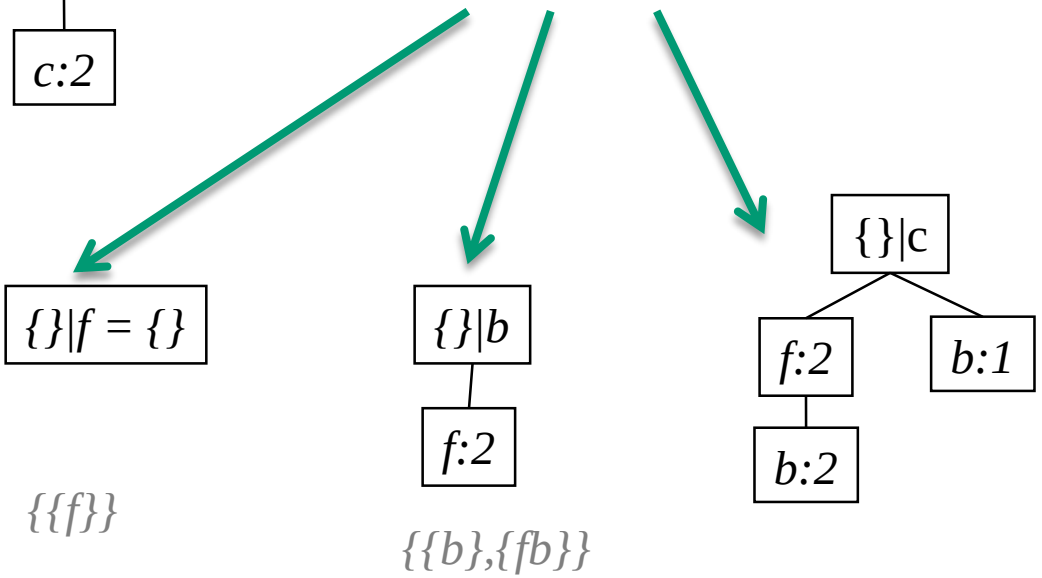
item	cond. pattern base
f	{}
b	*f:2*, {}
c	*fb:2*, *b:1*

FP-tree: Full Example



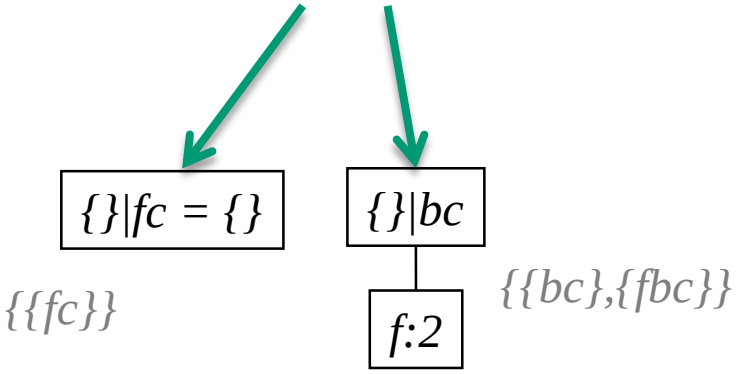
conditional pattern base 1:

item	cond. pattern base
f	*{}*
b	*f:2*
c	*fb:2, b:1*



conditional pattern base 2:

item	cond. pattern base
b	*f:2*
f	*{}*



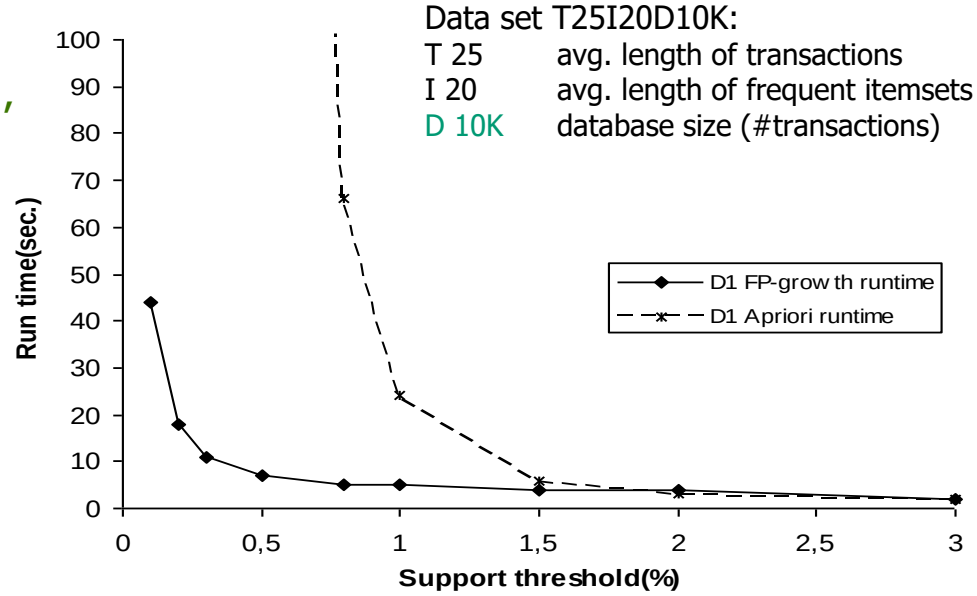
Principles of Frequent Pattern Growth

- Pattern growth property
  - Let α be a frequent itemset in DB, B be α's conditional pattern base, and β be an itemset in B. Then $\alpha \cup \beta$ is a frequent itemset in DB iff β is frequent in B.

- "*abcdef*" is a frequent pattern, if and only if
  - "*abcde*" is a frequent pattern, and
  - "*f*" is frequent in the set of transactions containing "*abcde*"

Why Is Frequent Pattern Growth Fast?

- Performance study in [Han, Pei&Yin '00] shows
  - FP-growth is an order of magnitude faster than Apriori, and is also faster than tree-projection



- Reasoning
  - No candidate generation, no candidate test
    - Apriori algorithm has to proceed breadth-first
  - Use compact data structure
  - Eliminate repeated database scan
  - Basic operation is counting and FP-tree building

Maximal or Closed Frequent Itemsets

- Big challenge: database contains potentially a huge number of frequent itemsets (especially if minSup is set too low).
  - A frequent itemset of length 100 contains 2^{100}-1 many frequent subsets
- *Closed frequent itemset*:
  An itemset X is *closed* in a data set D if there exists no proper super-itemset Y such that $support(X) = support(Y)$ in D.
  - The set of closed frequent itemsets contains complete information regarding its corresponding frequent itemsets.
- *Maximal frequent itemset*:
  An itemset X is *maximal* in a data set D if there exists no proper super-itemset Y such that $support(Y) \geq minSup$ in D.
  - The set of maximal itemsets does not contain the complete support information
  - More compact representation

Chapter 3: Frequent Itemset Mining

1) Introduction
   - Transaction databases, market basket data analysis
2) Mining Frequent Itemsets
   - Apriori algorithm, hash trees, FP-tree
3) Simple Association Rules
   - Basic notions, rule generation, interestingness measures
4) Further Topics
   - Hierarchical Association Rules
     - Motivation, notions, algorithms, interestingness
   - Quantitative Association Rules
     - Motivation, basic idea, partitioning numerical attributes, adaptation of apriori algorithm, interestingness
5) Extensions and Summary

Simple Association Rules: Introduction

- Transaction database:

  D= {{butter, bread, milk, sugar};
  {butter, flour, milk, sugar};
  {butter, eggs, milk, salt};
  {eggs};
  {butter, flour, milk, salt, sugar}}

- Frequent itemsets:

items	support
{butter}	4
{milk}	4
{butter, milk}	4
{sugar}	3
{butter, sugar}	3
{milk, sugar}	3
{butter, milk, sugar}	3

- Question of interest:
  - If milk and sugar are bought, will the customer always buy butter as well? $milk, sugar \Rightarrow butter$?
  - In this case, what would be the probability of buying butter?

Simple Association Rules: Basic Notions

- *Items* $I = \{i_1, i_2, \ldots, i_m\}$: a set of literals (denoting items)
- *Itemset* X*:* Set of items $X \subseteq I$
- *Database* D: Set of *transactions* T, each transaction is a set of items $T \subseteq I$
- Transaction T *contains* an itemset X: $X \subseteq T$
- The items in transactions and itemsets are sorted lexicographically:
  - itemset $X = (x_1, x_2, \ldots, x_k)$, where $x_1 \leq x_2 \leq \ldots \leq x_k$
- *Length* of an itemset: cardinality of the itemset (*k-itemset:* itemset of length *k)*
- The *support* of an itemset X is defined as: $support(X) = |\{T \in D | X \subseteq T\}|$
- *Frequent itemset: an itemset* X is called frequent iff $support(X) \geq minSup$

- *Association rule:* An association rule is an implication of the form $X \Rightarrow Y$ where $X, Y \subseteq I$ are two itemsets with $X \cap Y = \emptyset$.

- Note: simply enumerating all possible association rules is not reasonable!
  → What are the interesting association rules w.r.t. D?

Interestingness of Association Rules

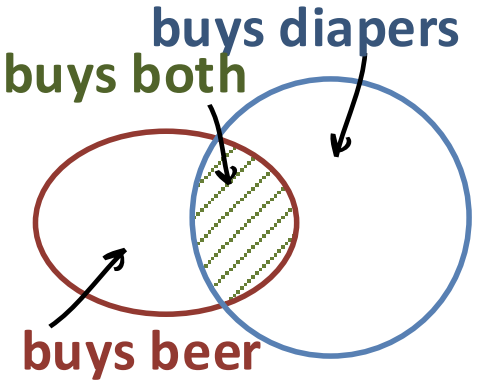
- *Interestingness of an association rule:*
  Quantify the interestingness of an association rule with respect to a transaction database D:
  - Support: frequency (probability) of the entire rule with respect to D

    $$support(X \Rightarrow Y) = P(X \cup Y) = \frac{|\{T \in D | X \cup Y \subseteq T\}|}{|D|} = support(X \cup Y)$$

    "probability that a transaction in D contains the itemset $X \cup Y$"
  - Confidence: indicates the strength of implication in the rule

    $$confidence(X \Rightarrow Y) = P(Y|X) = \frac{|\{T \in D | X \cup Y \subseteq T\}|}{|\{T \in D | X \subseteq T\}|} = \frac{support(X \cup Y)}{support(X)}$$

    "conditional probability that a transaction in D containing the itemset X also contains itemset Y"
  - Rule form: "$Body \Rightarrow Head\ [support, confidence]$"
- Association rule examples:
  - buys diapers ⇒ buys beers [0.5%, 60%]
  - major in CS ∧ takes DB ⇒ avg. grade A [1%, 75%]



Mining of Association Rules

- *Task of mining association rules:*
  Given a database D, determine all association rules having a $support \geq minSup$ and a $confidence \geq minConf$ (so-called *strong association rules*).
- Key steps of mining association rules:
  1) Find *frequent itemsets*, i.e., itemsets that have at least support $= minSup$

  e.g. Apriori, FP-growth

  2) Use the frequent itemsets to generate association rules
     - For each itemset X and every nonempty subset $Y \subset X$ generate rule $Y \Rightarrow (X - Y)$ if $minSup$ and $minConf$ are fulfilled
     - we have $2^{|X|} - 2$ many association rule candidates for each itemset X
- Example
  frequent itemsets

1-itemset	count	2-itemset	count	3-itemset	count
{A}	3	{A, B}	3	{A, B, C}	2
{B}	4	{A, C}	2		
{C}	5	{B, C}	4		

rule candidates: $A \Rightarrow B; B \Rightarrow A;\ A \Rightarrow C;\ C \Rightarrow A;\ B \Rightarrow C;\ C \Rightarrow B;$
$A, B \Rightarrow C; A, C \Rightarrow B; C, B \Rightarrow A; A \Rightarrow B, C; B \Rightarrow A, C; C \Rightarrow A, B$

Generating Rules from Frequent Itemsets

- For each frequent itemset X
  - For each nonempty subset Y of X, form a rule Y ⇒ $(X - Y)$
  - Delete those rules that do not have minimum confidence
    Note: 1) support always exceeds $minSup$
    2) the support values of the frequent itemsets suffice to calculate the confidence
- Example: $X = \{A, B, C\}$, $minConf = 60\%$
  - conf (A ⇒ B) = 3/3; ✓
  - conf (B ⇒ A) = 3/4; ✓
  - conf (A ⇒ C) = 2/3; ✓
  - conf (C ⇒ A) = 2/5; ✗
  - conf (B ⇒ C) = 4/4; ✓
  - conf (C ⇒ B) = 4/5; ✓
  - conf (A ⇒ B, C) = 2/3; ✓
  - conf (B ⇒ A, C) = 2/4; ✗
  - conf (C ⇒ A, B) = 2/5; ✗
  - conf (B, C ⇒ A) = ½ ✗
  - conf (A, C ⇒ B) = 1 ✓
  - conf (A, B ⇒ C) = 2/3 ✓

itemset	count
{A}	3
{B}	4
{C}	5
{A, B}	3
{A, C}	2
{B, C}	4
{A, B, C}	2

- Exploit anti-monotonicity for generating candidates for strong association rules!

Interestingness Measurements

- *Objective* measures
  - Two popular measurements:
  - support and
  - confidence

- *Subjective* measures [Silberschatz & Tuzhilin, KDD95]
  - A rule (pattern) is interesting if it is
  - unexpected (surprising to the user) and/or
  - actionable (the user can do something with it)

Criticism to Support and Confidence

Example 1 [Aggarwal & Yu, PODS98]

- Among 5000 students
  - 3000 play basketball (=60%)
  - 3750 eat cereal (=75%)
  - 2000 both play basket ball and eat cereal (=40%)
- Rule *play basketball* $\Rightarrow$ *eat cereal* [40%, 66.7%] is misleading because the overall percentage of students eating cereal is 75% which is higher than 66.7%
- Rule *play basketball* $\Rightarrow$ *not eat cereal* [20%, 33.3%] is far more accurate, although with lower support and confidence
- Observation: *play basketball* and *eat cereal* are *negatively correlated*

➢ Not all strong association rules are interesting and some can be misleading.
→ augment the support and confidence values with interestingness measures such as the correlation $A \Rightarrow B$ $[supp, conf, corr]$

Other Interestingness Measures: Correlation

- ***Lift*** is a simple correlation measure between two items *A* and *B:*

$$corr_{A,B} = \frac{P(A \cup B)}{P(A)P(B)} = \frac{P(B|A)}{P(B)} = \frac{conf(A \Rightarrow B)}{supp(B)}$$

! The two rules $A \Rightarrow B$ and $B \Rightarrow A$ have the same correlation coefficient.

- take both P(A) and P(B) in consideration

- $corr_{A,B} > 1$ the two items A and B are positively correlated
- $corr_{A,B} = 1$ there is no correlation between the two items A and B
- $corr_{A,B} < 1$ the two items A and B are negatively correlated

Other Interestingness Measures: Correlation

- Example 2:

X	1	1	1	1	0	0	0	0
Y	1	1	0	0	0	0	0	0
Z	0	1	1	1	1	1	1	1

- X and Y: positively correlated
- X and Z: negatively related
- support and confidence of X=>Z dominates
- but items X and Z are negatively correlated
- Items X and Y are positively correlated

rule	support	confidence	correlation
$X \Rightarrow Y$	25%	50%	2
$X \Rightarrow Z$	37.5%	75%	0.86
$Y \Rightarrow Z$	12.5%	50%	0.57

Chapter 3: Frequent Itemset Mining

1) Introduction
   - Transaction databases, market basket data analysis
2) Mining Frequent Itemsets
   - Apriori algorithm, hash trees, FP-tree
3) Simple Association Rules
   - Basic notions, rule generation, interestingness measures
4) Further Topics
   - Hierarchical Association Rules
     - Motivation, notions, algorithms, interestingness
   - Quantitative Association Rules
     - Motivation, basic idea, partitioning numerical attributes, adaptation of apriori algorithm, interestingness
5) Extensions and Summary

Hierarchical Association Rules: Motivation

- Problem of association rules in plain itemsets
  - *High minsup:* apriori finds only few rules
  - *Low minsup:* apriori finds unmanagably many rules
- Exploit item taxonomies (generalizations, *is-a* hierarchies) which exist in many applications

- New task: find all generalized association rules between generalized items → Body and Head of a rule may have items of any level of the hierarchy
- *Generalized association rule*: $X \Rightarrow Y$
  with $X, Y \subset I, X \cap Y = \emptyset$ and no item in Y is an ancestor of any item in X
  i.e., $jackets \Rightarrow clothes$ is essentially true

Hierarchical Association Rules: Motivating Example

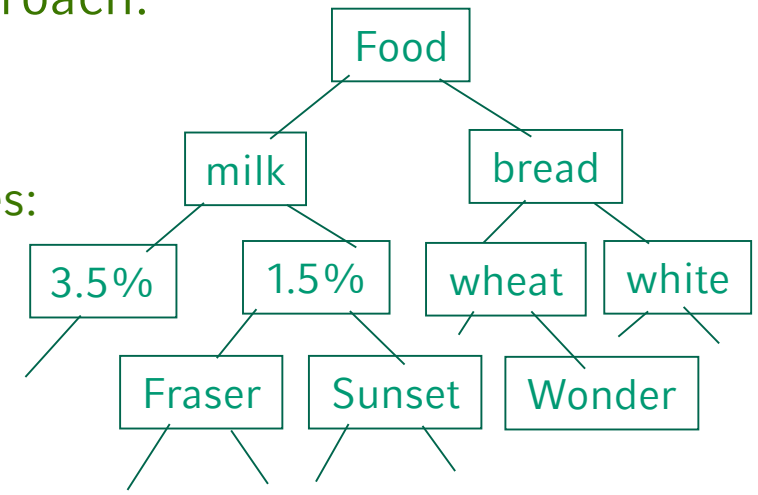
- Examples

  Jeans ⇒ boots
  jackets ⇒ boots
  } Support < minSup
  Outerwear ⇒ boots Support > minsup

- Characteristics
  - Support(“outerwear ⇒ boots”) is not necessarily equal to the sum support(“jackets ⇒ boots”) + support(“jeans ⇒ boots”)
    e.g. if a transaction with jackets, jeans and boots exists
  - Support for sets of generalizations (e.g., product groups) is higher than support for sets of individual items
    If the support of rule “outerwear ⇒ boots” exceeds minsup, then the support of rule “clothes ⇒ boots” does, too

Mining Multi-Level Associations

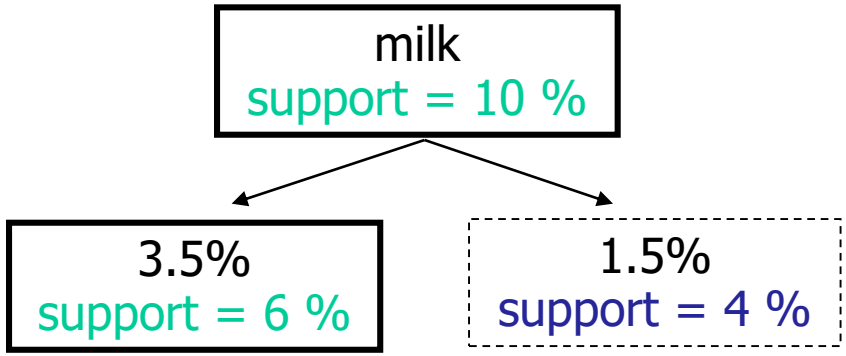
- A *top_down*, *progressive deepening* approach:
  - First find high-level strong rules:
    - *milk* ⇒ *bread* [20%, 60%].
  - Then find their lower-level "weaker" rules:
    - 1.5% *milk* ⇒ *wheat bread* [6%, 50%].



- Different min_support threshold across multi-levels lead to different algorithms:
  - adopting the same min_support across multi-levels
  - adopting reduced min_support at lower levels

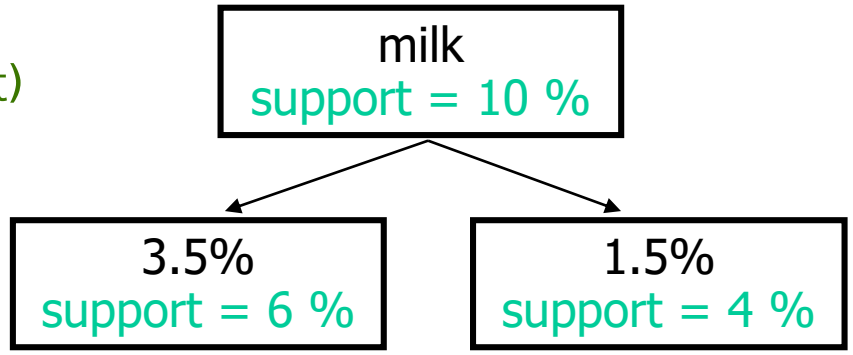
Minimum Support for Multiple Levels

- Uniform Support



  + the search procedure is simplified (monotonicity)
  + the user is required to specify only one support threshold

- Reduced Support (Variable Support)

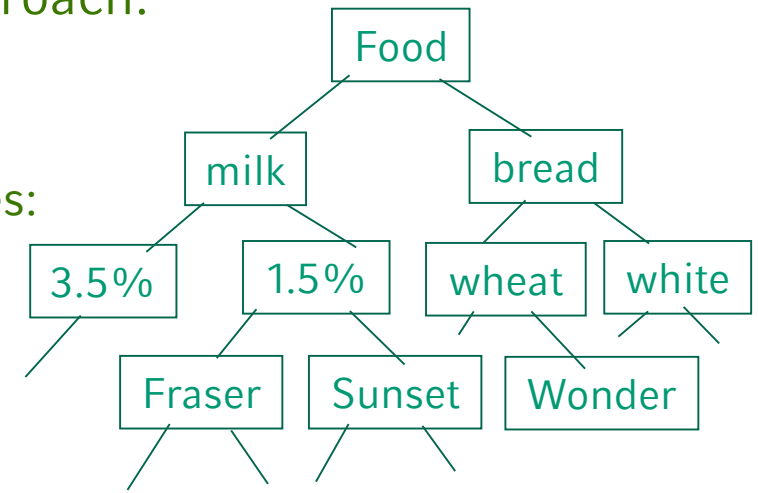


  + takes the lower frequency of items in lower levels into consideration

Multilevel Association Mining using Reduced Support

- A *top_down*, *progressive deepening* approach:
  - First find high-level strong rules:
    - *milk* ⇒ *bread* [20%, 60%].
  - Then find their lower-level "weaker" rules:
    - 1.5% *milk* ⇒ *wheat bread* [6%, 50%].

level-wise processing (breadth first)



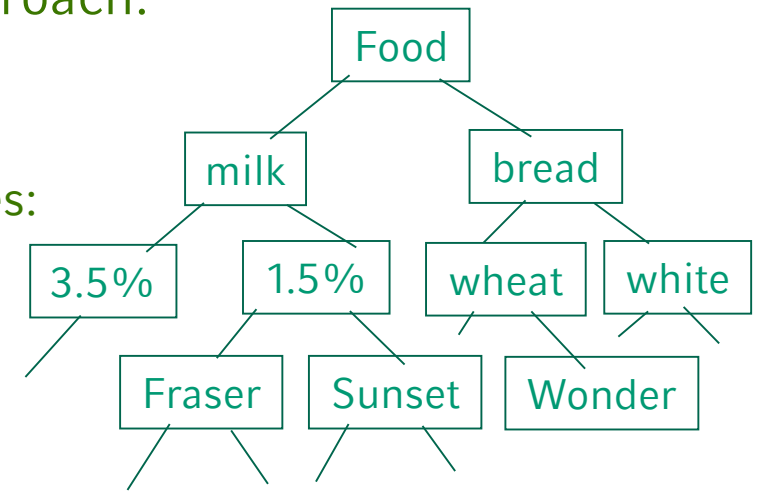
3 approaches using reduced Support:

- *Level-by-level independent method*:
  - Examine each node in the hierarchy, regardless of whether or not its parent node is found to be frequent
- *Level-cross-filtering by single item*:
  - Examine a node only if its parent node at the preceding level is frequent
- *Level-cross- filtering by k-itemset*:
  - Examine a k-itemset at a given level only if its parent k-itemset at the preceding level is frequent

Multilevel Associations: Variants

- A *top_down*, *progressive deepening* approach:
  - First find high-level strong rules:
    - *milk* ⇒ *bread* [20%, 60%].
  - Then find their lower-level "weaker" rules:
    - 1.5% *milk* ⇒ *wheat bread* [6%, 50%].

level-wise processing (breadth first)



- Variations at mining multiple-level association rules.
  - Level-crossed association rules:
    - 1.5 % *milk* ⇒ *Wonder wheat bread*
  - Association rules with multiple, alternative hierarchies:
    - 1.5 % *milk* ⇒ *Wonder bread*

Multi-level Association: Redundancy Filtering

- Some rules may be redundant due to "ancestor" relationships between items.
- Example
  - R_1: milk $\Rightarrow$ wheat bread [support = 8%, confidence = 70%]
  - R_2: 1.5% milk $\Rightarrow$ wheat bread [support = 2%, confidence = 72%]
- We say that rule 1 is an ancestor of rule 2.
- *Redundancy:*
  A rule is redundant if its support is close to the "expected" value, based on the rule's ancestor.

Interestingness of Hierarchical Association Rules: Notions

Let $X, X', Y, Y' \subseteq I$ be itemsets.

- An itemset X' is an ancestor of X iff there exist ancestors $x'_1, \ldots, x'_k$ of $x_1, \ldots, x_k \in X$ and $x_{k+1}, \ldots, x_n$ with $n = |X|$ such that
$$X' = \{x'_1, \ldots, x'_k, x_{k+1}, \ldots, x_n\}.$$
- Let X' and Y' be ancestors of X and Y. Then we call the rules $X' \Rightarrow Y'$, $X \Rightarrow Y'$, and $X' \Rightarrow Y$ *ancestors* of the rule X ⇒ Y .
- The rule X´ ⇒ Y´ is a *direct ancestor* of rule X ⇒ Y in a set of rules if:
  - Rule X´ ⇒ Y' is an ancestor of rule X ⇒ Y, and
  - There is no rule X" ⇒ Y" such that X" ⇒ Y" is an ancestor of X ⇒ Y and X´ ⇒ Y´ is an ancestor of X" ⇒ Y"
- A hierarchical association rule X ⇒ Y is called *R-interesting* if:
  - There are no direct ancestors of X ⇒ Y or
  - The actual support is larger than *R* times the expected support or
  - The actual confidence is larger than *R* times the expected confidence

Expected Support and Expected Confidence

- How to compute the expected support?
  Given the rule for X ⇒ Y and its ancestor rule X′ ⇒ Y′ the expected support of X ⇒ Y is defined as:

$$E_{Z'}[\mathrm{P}(Z)] = \frac{\mathrm{P}(z_1)}{\mathrm{P}(z_1')} \times \cdots \times \frac{\mathrm{P}(z_j)}{\mathrm{P}(z_j')} \times \mathrm{P}(Z')$$

  where $Z = X \cup Y = \{z_1, \ldots, z_n\}$, $Z' = X' \cup Y' = \{z_1', \ldots, z_j', z_{j+1}, \ldots, z_n\}$ and each $z_i' \in Z'$ is an ancestor of $z_i \in Z$

[SA'95] R. Srikant, R. Agrawal: *Mining Generalized Association Rules*. In VLDB, 1995.

Expected Support and Expected Confidence

- How to compute the expected confidence?
  Given the rule for X ⇒ Y and its ancestor rule X′ ⇒ Y′, then the expected confidence of X ⇒ Y is defined as:

$$E_{X' \Rightarrow Y'}[\mathrm{P}(Y|X)] = \frac{\mathrm{P}(y_1)}{\mathrm{P}(y_1')} \times \cdots \times \frac{\mathrm{P}(y_j)}{\mathrm{P}(y_j')} \times \mathrm{P}(Y'|X')$$

  where $Y = \{y_1, \ldots, y_n\}$ and $Y' = \{y_1', \ldots, y_j', y_{j+1}, \ldots, y_n\}$ and each $y_i' \in Y'$ is an ancestor of $y_i \in Y$

[SA'95] R. Srikant, R. Agrawal: *Mining Generalized Association Rules*. In VLDB, 1995.

Interestingness of Hierarchical Association Rules:Example

- Example
  - Let $R = 1.6$

Item	Support
clothes	20
outerwear	10
jackets	4

No	rule	support	R-interesting?
1	clothes ⇒ shoes	10	yes: no ancestors
2	outerwear ⇒ shoes	9	yes: Support > R *exp. support (wrt. rule 1) = $(1.6 \cdot (\frac{10}{20} \cdot 10)) = 8$
3	jackets ⇒ shoes	4	Not wrt. support: Support > R * exp. support (wrt. rule 1) = 3.2 Support < R * exp. support (wrt. rule 2) = 5.75 → still need to check the confidence!

Chapter 3: Frequent Itemset Mining

1) Introduction
   - Transaction databases, market basket data analysis
2) Simple Association Rules
   - Basic notions, rule generation, interestingness measures
3) Mining Frequent Itemsets
   - Apriori algorithm, hash trees, FP-tree
4) Further Topics
   - Hierarchical Association Rules
     - Motivation, notions, algorithms, interestingness
   - Multidimensional and Quantitative Association Rules
     - Motivation, basic idea, partitioning numerical attributes, adaptation of apriori algorithm, interestingness
5) Summary

Multi-Dimensional Association: Concepts

- Single-dimensional rules:
  - buys milk $\Rightarrow$ buys bread
- Multi-dimensional rules: $\geq$ 2 dimensions
  - Inter-dimension association rules (*no repeated dimensions*)
    - age between 19-25 $\wedge$ status is student $\Rightarrow$ buys coke
  - hybrid-dimension association rules (*repeated dimensions*)
    - age between 19-25 $\wedge$ buys popcorn $\Rightarrow$ buys coke

Techniques for Mining Multi-Dimensional Associations

- Search for frequent *k*-predicate set:
  - Example: {age, occupation, buys} is a 3-predicate set.
  - Techniques can be categorized by how age is treated.

1. Using static discretization of quantitative attributes
   - Quantitative attributes are statically discretized by using predefined concept hierarchies.
2. Quantitative association rules
   - Quantitative attributes are dynamically discretized into “bins”based on the distribution of the data.
3. Distance-based association rules
   - This is a dynamic discretization process that considers the distance between data points.

Quantitative Association Rules

- Up to now: associations of *boolean* attributes only
- Now: *numerical* attributes, too
- Example:
  - Original database

ID	age	marital status	# cars
1	23	single	0
2	38	married	2

  - Boolean database

ID	age: 20..29	age: 30..39	m-status: single	m-status: married	. . .
1	1	0	1	0	. . .
2	0	1	0	1	. . .

Quantitative Association Rules: Ideas

- Static discretization
  - Discretization of all attributes *before* mining the association rules
  - E.g. by using a generalization hierarchy for each attribute
  - Substitute numerical attribute values by ranges or intervals

- Dynamic discretization
  - Discretization of the attributes *during* association rule mining
  - Goal (e.g.): maximization of confidence
  - Unification of neighboring association rules to a generalized rule

Partitioning of Numerical Attributes

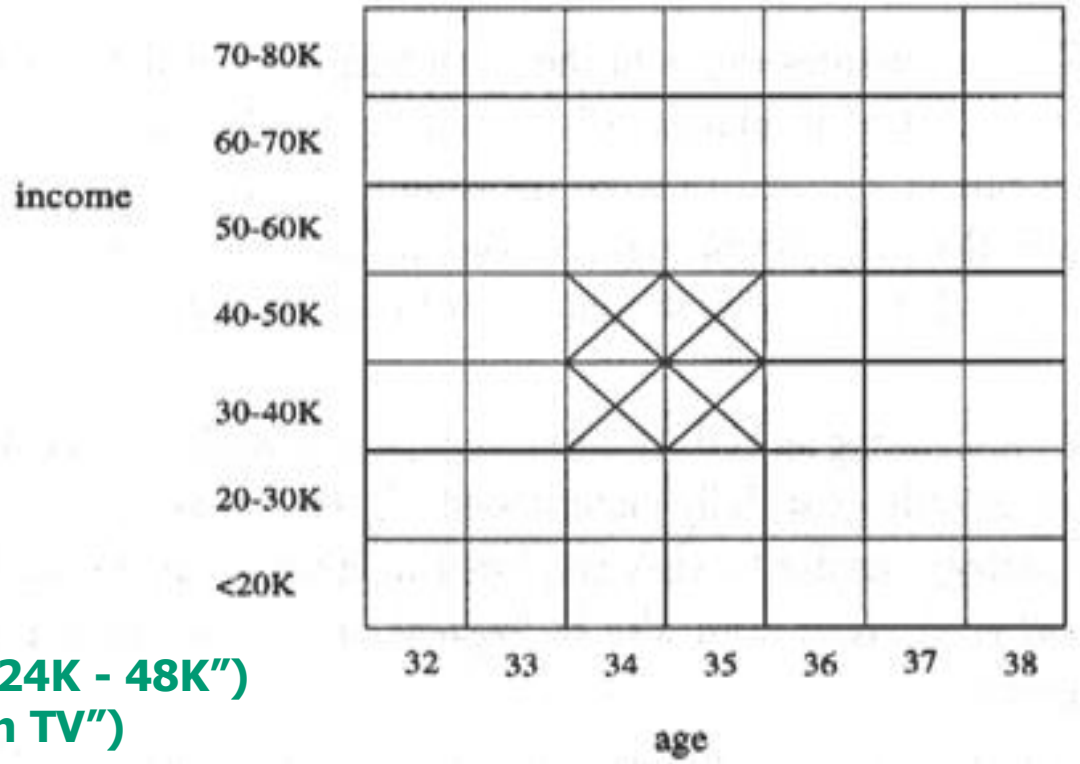
- Problem: Minimum support
  - Too many intervals → too small support for each individual interval
  - Too few intervals → too small confidence of the rules

- Solution
  - First, partition the domain into many intervals
  - Afterwards, create new intervals by merging adjacent interval

- Numeric attributes are *dynamically* discretized such that the confidence or compactness of the rules mined is maximized.

Quantitative Association Rules

- 2-D quantitative association rules: $A_{quan1} \wedge A_{quan2} \Rightarrow A_{cat}$
- Cluster "adjacent" association rules to form general rules using a 2-D grid.
- Example:



age(X,"30-34") $\wedge$ income(X,"24K - 48K")
$\Rightarrow$ buys(X,"high resolution TV")

Chapter 3: Frequent Itemset Mining

1) Introduction
   - Transaction databases, market basket data analysis
2) Mining Frequent Itemsets
   - Apriori algorithm, hash trees, FP-tree
3) Simple Association Rules
   - Basic notions, rule generation, interestingness measures
4) Further Topics
   - Hierarchical Association Rules
     - Motivation, notions, algorithms, interestingness
   - Quantitative Association Rules
     - Motivation, basic idea, partitioning numerical attributes, adaptation of apriori algorithm, interestingness
5) Summary

Chapter 3: Summary

- Mining frequent itemsets
  - Apriori algorithm, hash trees, FP-tree
- Simple association rules
  - support, confidence, rule generation, interestingness measures (correlation), …
- Further topics
  - Hierarchical association rules: algorithms (top-down progressive deepening), multilevel support thresholds, redundancy and R-interestingness
  - Quantitative association rules: partitioning numerical attributes, adaptation of apriori algorithm, interestingness
- Extensions: multi-dimensional association rule mining

UNIT-IV

Graph Mining, Social Network analysis and multi-relational data mining, Spatial data mining, Multimedia data mining, Text mining, Mining the world wide web(www), Data mining applications, Social impacts of data mining, Trends in data Mining.

Multi-Relational Data Mining (MRDM)

- ✓ **Multi-relational data mining (MRDM)** is a form of Data Mining operating on data stored in multiple database tables.
- ✓ MRDM is a multi-disciplinary field which deals with the Knowledge discovery from relational database which consist of number of relations.
- ✓ MRDM is required in domains where the data are highly structured.

- ✓ The multi relational data mining approach has developed as an alternative way for handling the structured data such that RDBMS. It provides mining in multiple tables directly.
- ✓ Three popular pattern finding techniques classification, clustering and association are frequently used in MRDM.
- ✓ Multi Relational Data Mining algorithms look for patterns among multiple tables (relational patterns).

MRDM APPROACHES

There are so many approaches supported by the Multi Relational Data Mining these are as below:

- ✓ **Inductive Logic Programming (ILP):** This ILP paradigm says that how the logic program will convert the patterns.
- ✓ **Multi-relational Clustering:** This technology is used to cluster the tuples in the target table in the relational database, so calculating the distance of relations in the target table is the main task in the multi-relation clustering.

- ✓ **Probabilistic Relational Models**: A probabilistic relational model (PRM) or a relational probability model is a model in which the probabilities are specified on the relations, independently of the actual individuals. Different individuals share the **probability** parameters.

Multi-relational Data Mining framework

- ✓ **Multi-relational Data Mining framework** is based on the Search for interesting patterns in the relational database,
- ✓ It is a framework which deals with gathering the data about the data (metadata) from a database and choose the best approach to get the optimal results.

MRDM Framework Architecture

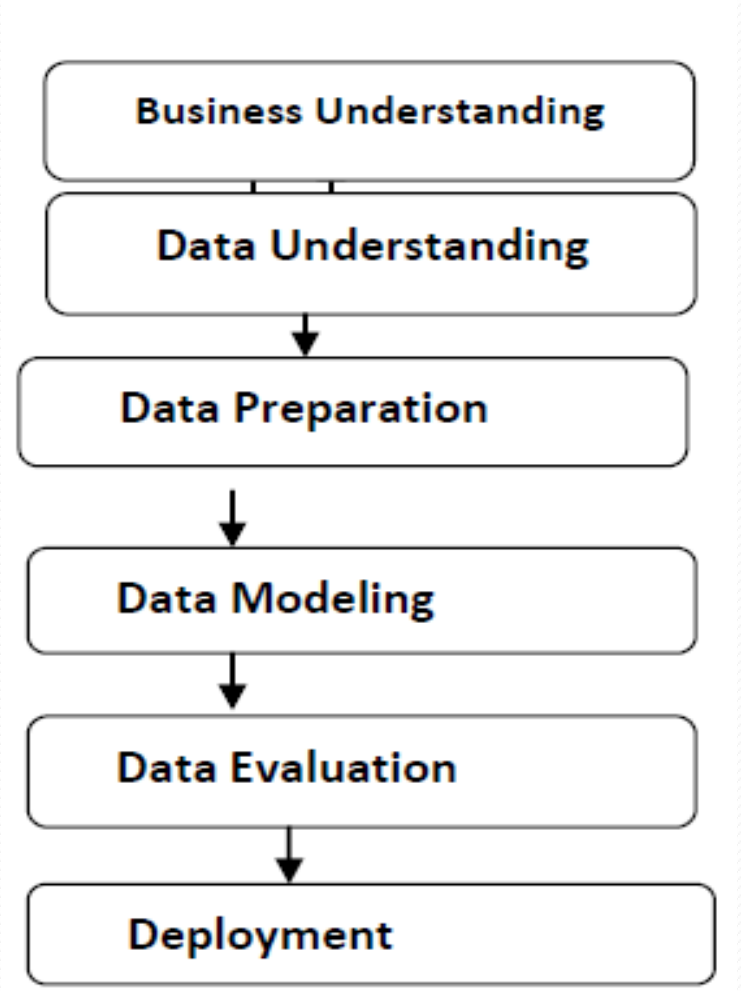


Figure: MRDM Framework Architecture

- ✓ Data understanding means gathering the metadata from the database which describes the best approach of the analysis.
- ✓ Data Preparation means transformation of the database into MRDM formats where we select the algorithms.

Spatial Data Mining

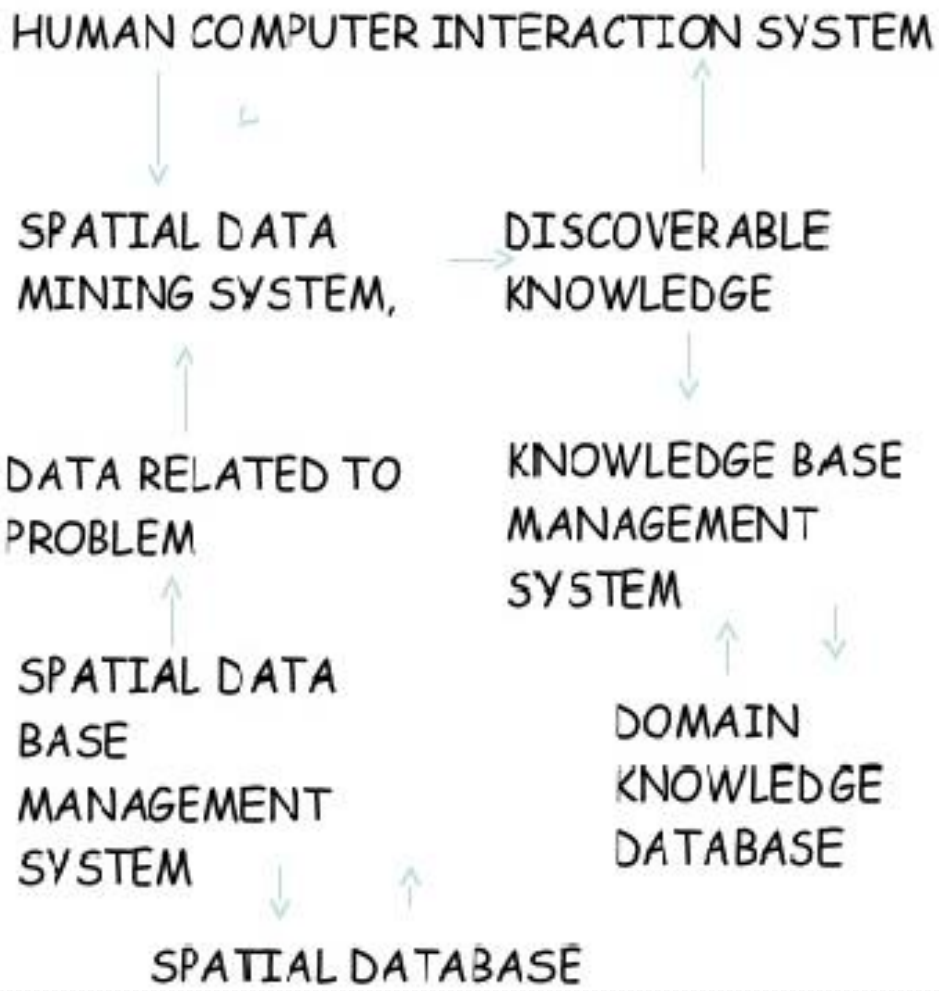
- ✓ Spatial data mining is the process of discovering interesting, useful, non-trivial patterns from large spatial datasets – E.g. Determining hotspots: unusual locations.
- ✓ **Spatial Data Mining** is the application of **Data Mining** to **spatial** models.
- ✓ In **spatial Data Mining**, analysts use geographical or **spatial** information to produce business intelligence or other results.
- ✓ This requires specific techniques and resources to get the geographical **data** into relevant and useful formats.

Spatial Data Mining Tasks

- ✓ Characteristics rule. – A spatial characteristic rule is a general description of spatial data. For example, a rule describing the general price range of houses in various geographic regions in a city is a spatial characteristic rule.
- ✓ Discriminate rule. E.g. Comparison of price ranges of different geographical area.
- ✓ Association rule-: we can associate the non spatial attribute to spatial attribute or spatial attribute to spatial attribute.
- ✓ Clustering rule-: helpful to find outlier detection which is useful to find suspicious knowledge E.g. Group crime location.

- ✓ Classification rule-: it defines whether a spatial entity belong to a particular class or how many classes will be classified. e.g. Remote sensed image based on spectrum and GIS data.
- ✓ Trend detection-A trend is a temporal pattern in some time series data. Spatial trend is defined as consider a non spatial attribute which is the neighbour of a spatial data object. Spatial trends describe a regular change of non-spatial attributes when moving away from certain start objects. e.g. economic power, is an important issue in economic geography.

Architecture of Spatial Data mining

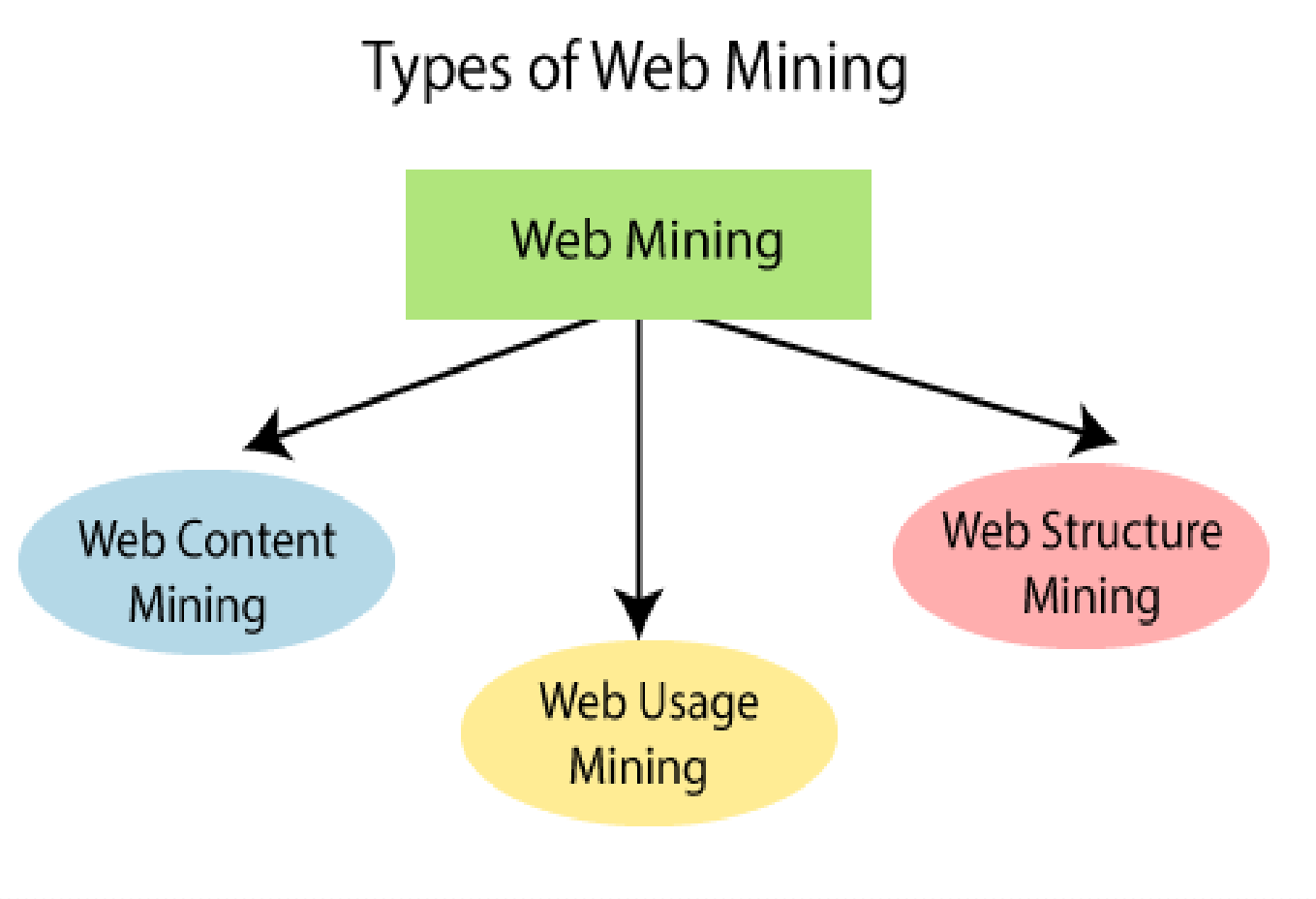


	Classical Data Mining	Spatial Data Mining
Input	Simple types Explicit relationship	Complex types Implicit relationships
Statistical Foundation	Independence of samples	Spatial autocorrelation
Output	Set-based interest measures e.g., classification accuracy	Spatial interest measures, e.g., spatial accuracy
Computational Process	Combinatorial optimization, Numerical Algorithms	Computational efficiency opportunity Spatial autocorrelation, plane-sweeping

Mining the world wide web(www)

- **Web mining** can define as the method of utilizing data mining techniques and algorithms to extract useful information directly from the web, such as Web documents and services, hyperlinks, Web content, and server logs.
- The World Wide Web contains a large amount of data that provides a rich source to data mining.
- The objective of Web mining is to look for patterns in Web data by collecting and examining data in order to gain insights.

There are three types of data mining:



1. Web Content Mining:

- ✓ Web content mining can be used to extract useful data, information, knowledge from the web page content.
- ✓ In web content mining, each web page is considered as an individual document.
- ✓ The primary task of content mining is data extraction, where structured data is extracted from unstructured websites.
- ✓ Example, if any user searches for a specific task on the search engine, then the user will get a list of suggestions.

2. Web Structured Mining:

- ✓ The web structure mining can be used to find the link structure of hyperlink.
- ✓ It is used to identify that data either link the web pages or direct link network.
- ✓ In Web Structure Mining, an individual considers the web as a directed graph, with the web pages being the vertices that are associated with hyperlinks.
- ✓ Structure and content mining methodologies are usually combined.

3. Web Usage Mining:

- ✓ Web usage mining is used to extract useful data, information, knowledge from the weblog records, and assists in recognizing the user access patterns for web pages.
- ✓ In Mining, the usage of web resources, the individual is thinking about records of requests of visitors of a website, that are often collected as web server logs.

- ✓ While the content and structure of the collection of web pages follow the intentions of the authors of the pages, the individual requests demonstrate how the consumers see these pages.
- ✓ Web usage mining may disclose relationships that were not proposed by the creator of the pages.

Application of Web Mining:

- ✓ Web mining has an extensive application because of various uses of the web. The list of some applications of web mining is given below.
- ✓ Marketing and conversion tool
- ✓ Data analysis on website and application accomplishment.
- ✓ Audience behavior analysis
- ✓ Advertising and campaign accomplishment analysis.
- ✓ Testing and analysis of a site.

Challenges in Web Mining:

- ✓ The complexity of web pages
- ✓ The web is a dynamic data source
- ✓ Diversity of client networks:
- ✓ Relevancy of data:
- ✓ The web is too broad:

Trends in Data Mining

Data mining concepts are still evolving and here are the **latest trends** that we get to see in this field −

- ✓ Application Exploration.
- ✓ Scalable and interactive data mining methods.
- ✓ Integration of data mining with database systems, data warehouse systems and web database systems.
- ✓ Standardization of data mining query language.
- ✓ Visual data mining.

- ✓ New methods for mining complex types of data.
- ✓ Biological data mining.
- ✓ Data mining and software engineering.
- ✓ Web mining.
- ✓ Distributed data mining.
- ✓ Real time data mining.
- ✓ Multi database data mining.
- ✓ Privacy protection and information security in data mining.

What is the Temporal Data Mining?

Temporal data mining defines the process of extraction of non-trivial, implicit, and potentially essential data from large sets of temporal data. Temporal data are a series of primary data types, generally numerical values, and it deals with gathering beneficial knowledge from temporal data.

The objective of temporal data mining is to find temporal patterns, unexpected trends, or several hidden relations in the higher sequential data, which is composed of a sequence of nominal symbols from the alphabet referred to as a temporal sequence and a sequence of continuous real-valued components called a time series, by utilizing a set of approaches from machine learning, statistics, and database technologies.

Temporal data mining is composed of three major works such as the description of temporal data, representation of similarity measures, and mining services.

Temporal Data Mining includes processing time series, generally sequences of data, which compute values of the same attribute at a sequence of multiple time points. Pattern matching using such information, where it is searching for specific patterns of interest, has attracted considerable interest in current years.

Temporal Data Mining can include the exploitation of efficient techniques of data storage, quick processing, and quick retrieval methods that have been advanced for temporal databases.

Temporal data mining is an individual phase in the process of knowledge discovery in temporal databases that calculate temporal patterns from or fit models too, temporal data is a temporal data mining algorithm.

Temporal data mining is concerned with the analysis of temporal data and for discovering temporal patterns and consistencies in sets of temporal information. It also allows the possibility of computer-driven, automatic exploration of the data. There are various tasks in temporal mining which are as follows −

- Data characterization and comparison
- Clustering analysis
- Classification
- Association rules
- Pattern analysis
- Prediction and trend analysis

Temporal data mining has led to a new way of interacting with a temporal database and specifying queries at a much more abstract level than say, temporal structured query language permits. It also facilities data exploration for problems that are due to multiple and multi-dimensionality.

The basic goal of temporal classification is to predict temporally related fields in a temporal database based on other fields. The problem, in general, is cast as deciding the general value of the temporal variable being predicted given the different fields, the training data in which the target variable is given for each observation, and a set of assumptions representing one’s prior knowledge of the problem. Temporal classification techniques are associated with the complex problem of density estimation.

Difference between Spatial and Temporal Data Mining

1. Spatial Data Mining :
Spatial data mining is the process of discovering interesting and previously unknown, but potentially useful patterns from spatial databases. In spatial data mining analyst use geographical or spatial information to produce business intelligence or other results. Challenges involved in spatial data mining include identifying patterns or finding objects that are relevant to research project.

2. Temporal Data Mining :
Temporal data refers to the extraction of implicit, non-trivial and potentially useful abstract information from large collection of temporal data. It is concerned with the analysis of temporal data and for finding temporal patterns and regularities in sets of temporal data tasks of temporal data mining are –

- Data Characterization and Comparison
- Cluster Analysis
- Classification
- Association rules
- Prediction and Trend Analysis
- Pattern Analysis

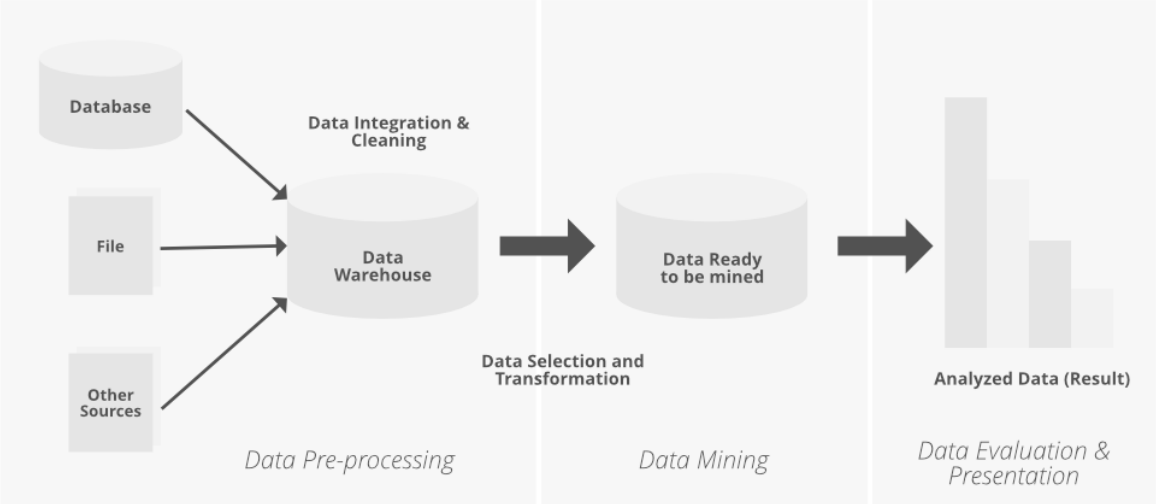
Difference between Spatial and Temporal Data Mining :

SNO.	Spatial data mining	Temporal data mining
1.	It requires space.	It requires time.
2.	Spatial mining is the extraction of knowledge/spatial relationship and interesting measures that are not explicitly stored in spatial database.	Temporal mining is the extraction of knowledge about occurrence of an event whether they follow Cyclic , Random ,Seasonal variations etc.
3.	It deals with spatial (location , Geo-referenced) data.	It deals with implicit or explicit Temporal content , from large quantities of data.
4.	Spatial databases reverses spatial objects derived by spatial data. types and spatial association among such objects.	Temporal data mining comprises the subject as well as its utilization in modification of fields.

5.	It includes finding characteristic rules, discriminant rules, association rules and evaluation rules etc.	It aims at mining new and unknown knowledge, which takes into account the temporal aspects of data.
6.	It is the method of identifying unusual and unexplored data but useful models from spatial databases.	It deals with useful knowledge from temporal data.
7.	Examples – Determining hotspots , Unusual locations.	Examples – An association rule which looks like – "Any Person who buys a car also buys steering lock". By temporal aspect this rule would be – " Any person who buys a car also buys a steering lock after that ".

Data Mining – Time-Series, Symbolic and Biological Sequences Data

Data mining refers to extracting or mining knowledge from large amounts of data. In other words, Data mining is the science, art, and technology of discovering large and complex bodies of data in order to discover useful patterns. Theoreticians and practitioners are continually seeking improved techniques to make the process more efficient, cost-effective, and accurate.
This article discusses Sequence data. Evaluation of data reached the maximum extent and may still peruse in the future. To generalize the evaluation of data we classify them as Sequence Data, Graphs, and Network Mining, another kind of data.



A sequence is an ordered list of events. Sequences data are classified based on characteristics as:

- Time-Series data (data with respect to time)
- Symbolic data (data with laps in an interval of time)
- Biological data (data related to DNA and protein)

Time-Series Data:

In this type of sequence, the data are of numeric data type recorded at a regular level. They are generated by an economic process like Stock Market analysis, Medical Observations. They are useful for studying natural phenomena.

Nowadays these times series are used for piecewise data approximations for further analysis. In this time-series data, we find a subsequence that matches the query we search.

- **Time Series Forecasting:** Forecasting is a method of making predictions based on past and present data to know what happens in the future. Trend analysis is a method of forecasting Time Series.

It is a function that generates historic patterns in time series that are used in short and long-term predictions. We can obtain various patterns in time series like cyclic movements, trend movements, seasonal movements as we see they are with respect to time or season. ARIMA, SARIMA, long memory time series modeling are some of the popular methods for such analysis

Symbolic Data:

This type of ordered set of elements or events is recorded with or without a concrete notion of time. Some symbolic sequences such as customer shopping sequences, web clickstreams are examples of symbolic data. Sequential pattern mining is mainly used for symbolic sequence

Constraint-based pattern matching is one of the best ways to interact with user-defined data. Apriori is an Algorithm used for this type of analysis Below is an example of a symbolic date where we see customers c1 and c2 are purchasing products at different time intervals

Tid	Time	Cid	Event(purchase products)
t1	11:45:30	c1	wheat, rice, fruit
t2	11:36:50	c2	rice, fruit
t1	12:00:01	c1	juice, rice
t2	01:00:34	c2	sugar, milk

Biological Data:

They are made of DNA and protein sequences. They are very long and complicated but have some hidden meaning. These types of data are used for the sequence of nucleotides or amino acids. These analyses are used for aligning, indexes, analyze biological sequence and play a crucial role in bioinformatics and modern biology. Substitution trees are used to find the probabilities of amino acids and probabilities of intersections. BLAST-Basic Local Alignment Search Tool is the most effective tool for biological sequence.

KDD Process in Data Mining

In the context of computer science, “Data Mining” can be referred to as knowledge mining from data, knowledge extraction, data/pattern analysis, data archaeology, and data dredging. Data Mining also known as Knowledge Discovery in Databases, refers to the nontrivial extraction of implicit, previously unknown and potentially useful information from data stored in databases.

The need of data mining is to extract useful information from large datasets and use it to make predictions or better decision-making. Nowadays, data mining is used in almost all places where a large amount of data is stored and processed.

For examples: Banking sector, Market Basket Analysis, Network Intrusion Detection.

KDD Process

KDD (Knowledge Discovery in Databases) is a process that involves the extraction of useful, previously unknown, and potentially valuable information from large datasets. The KDD process is an iterative process and it requires multiple iterations of the above steps to extract accurate knowledge from the data.The following steps are included in KDD process:

Data Cleaning

Data cleaning is defined as removal of noisy and irrelevant data from collection.

1. Cleaning in case of **Missing values**.
2. Cleaning **noisy** data, where noise is a random or variance error.
3. Cleaning with **Data discrepancy detection** and **Data transformation tools**.

Data Integration

Data integration is defined as heterogeneous data from multiple sources combined in a common source(DataWarehouse). Data integration using **Data Migration tools, Data Synchronization tools and ETL**(Extract-Load-Transformation) process.

Data Selection

Data selection is defined as the process where data relevant to the analysis is decided and retrieved from the data collection. For this we can use **Neural network, Decision Trees, Naive bayes, Clustering**, and **Regression** methods.

Data Transformation

Data Transformation is defined as the process of transforming data into appropriate form required by mining procedure. Data Transformation is a two step process:

1. **Data Mapping**: Assigning elements from source base to destination to capture transformations.
2. **Code generation**: Creation of the actual transformation program.

Data Mining

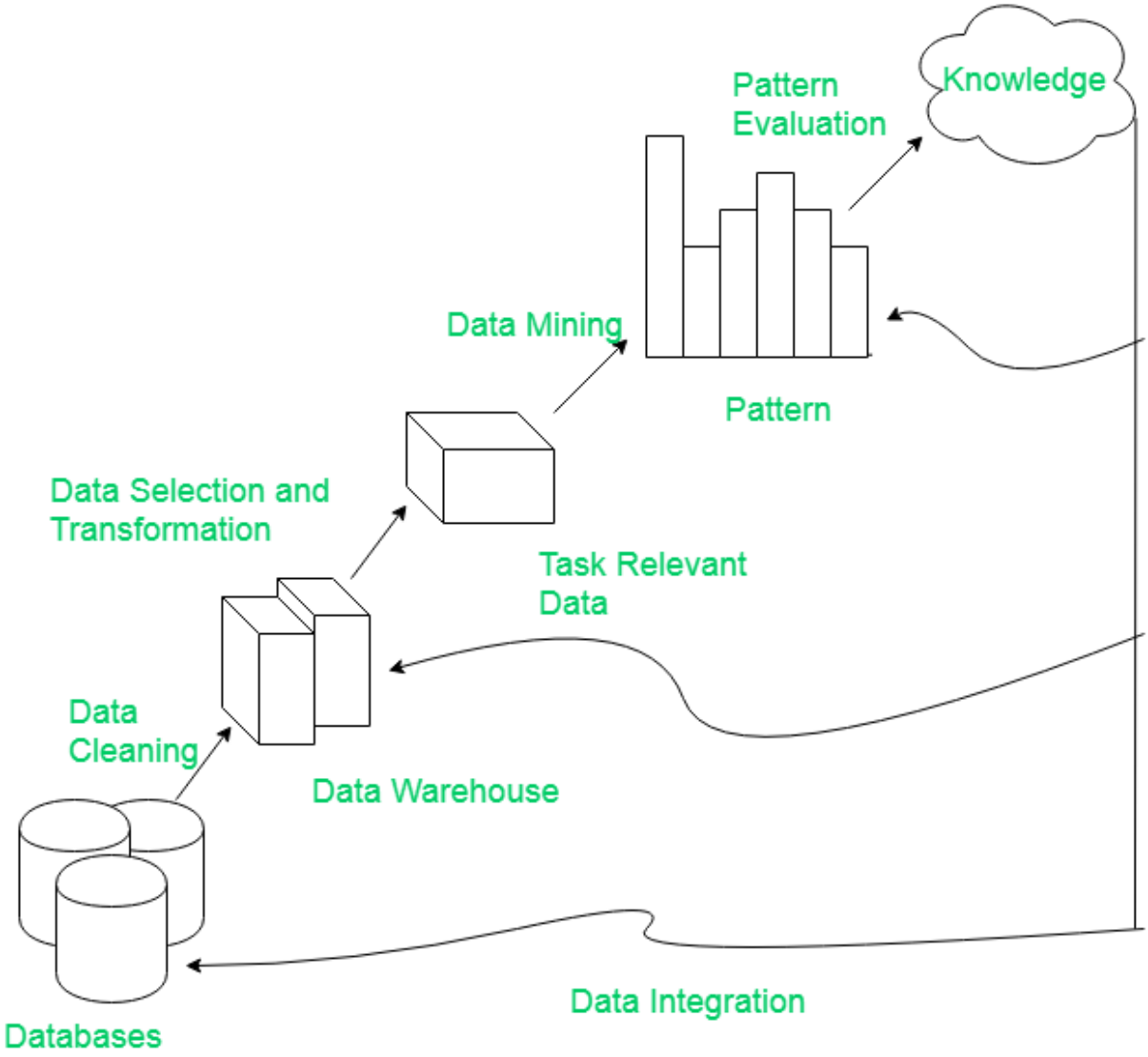
Data mining is defined as techniques that are applied to extract patterns potentially useful. It transforms task relevant data into **patterns, and d**ecides purpose of model using **classification** or **characterization**.

Pattern Evaluation

Pattern Evaluation is defined as identifying strictly increasing patterns representing knowledge based on given measures. It find **interestingness score** of each pattern, and uses **summarization** and **Visualization** to make data understandable by user.

Knowledge Representation

This involves presenting the results in a way that is meaningful and can be used to make decisions.



Note: KDD is an **iterative process** where evaluation measures can be enhanced, mining can be refined, new data can be integrated and transformed in order to get different and more appropriate results.**Preprocessing of databases** consists of **Data cleaning** and **Data Integration**.

Advantages of KDD

1. **Improves decision-making:** KDD provides valuable insights and knowledge that can help organizations make better decisions.
2. **Increased efficiency:** KDD automates repetitive and time-consuming tasks and makes the data ready for analysis, which saves time and money.

3. **Better customer service:** KDD helps organizations gain a better understanding of their customers' needs and preferences, which can help them provide better customer service.
4. **Fraud detection:** KDD can be used to detect fraudulent activities by identifying patterns and anomalies in the data that may indicate fraud.
5. **Predictive modeling:** KDD can be used to build predictive models that can forecast future trends and patterns.

Disadvantages of KDD

1. **Privacy concerns:** KDD can raise privacy concerns as it involves collecting and analyzing large amounts of data, which can include sensitive information about individuals.
2. **Complexity:** KDD can be a complex process that requires specialized skills and knowledge to implement and interpret the results.
3. **Unintended consequences:** KDD can lead to unintended consequences, such as bias or discrimination, if the data or models are not properly understood or used.
4. **Data Quality:** KDD process heavily depends on the quality of data, if data is not accurate or consistent, the results can be misleading
5. **High cost:** KDD can be an expensive process, requiring significant investments in hardware, software, and personnel.
6. **Overfitting:** KDD process can lead to overfitting, which is a common problem in machine learning where a model learns the detail and noise in the training data to the extent that it negatively impacts the performance of the model on new unseen data.

Difference between KDD and Data Mining

Parameter	**KDD**	**Data Mining**
Definition	KDD refers to a process of identifying valid, novel, potentially useful, and ultimately understandable patterns and relationships in data.	Data Mining refers to a process of extracting useful and valuable information or patterns from large data sets.
Objective	To find useful knowledge from data.	To extract useful information from data.
Techniques Used	Data cleaning, data integration, data selection, data transformation, data mining, pattern evaluation, and knowledge representation and visualization.	Association rules, classification, clustering, regression, decision trees, neural networks, and dimensionality reduction.

Parameter	KDD	Data Mining
Output	Structured information, such as rules and models, that can be used to make decisions or predictions.	Patterns, associations, or insights that can be used to improve decision-making or understanding.
Focus	Focus is on the discovery of useful knowledge, rather than simply finding patterns in data.	Data mining focus is on the discovery of patterns or relationships in data.
Role of domain expertise	Domain expertise is important in KDD, as it helps in defining the goals of the process, choosing appropriate data, and interpreting the results.	Domain expertise is less critical in data mining, as the algorithms are designed to identify patterns without relying on prior knowledge.