DNYANSAGAR ARTS AND COMMERCE COLLEGE, BALEWADI, PUNE – 45

Subject: 603 : Recent Trends in IT

CLASS: TYBBA(CA) VI SEM (2013 PATTERN)

Prof. Gayatri A.Amate

www.dacc.edu.in

Unit 1: Software Process And Project Metrics, Analysis Concepts And Principles

Software metrics

Software metrics is a standard of measure that contains many activities which involve some degree of measurement. It can be classified into three categories: product metrics, process metrics, and project metrics.

Key Project & Process Metric Groups

Project managers have a wide variety of metrics to choose from. We can classify the most commonly used metrics into the following groups:

Metrics for software quality

Software metrics can be classified into three categories −

Product metrics − Describes the characteristics of the product such as size, complexity, design features, performance, and quality level.

Process metrics − These characteristics can be used to improve the development and maintenance activities of the software.

Project metrics − This metrics describe the project characteristics and execution. Examples include the number of software developers, the staffing pattern over the life cycle of the software, cost, schedule, and productivity.

Some metrics belong to multiple categories. For example, the in-process quality metrics of a project are both process metrics and project metrics.

Software quality metrics can be further divided into three categories −

- Product quality metrics
- In-process quality metrics
- Maintenance quality metrics

Product Quality Metrics

This metrics include the following −

Mean Time to Failure

Defect Density

Customer Problems

Customer Satisfaction

Mean Time to Failure

Objectives of SQA Activities

The objectives of SQA activities are as follows −

In Software development (process-oriented)

- Assuring an acceptable level of confidence that the software will conform to functional technical requirements.
- Assuring an acceptable level of confidence that the software will conform to managerial scheduling and budgetary requirements.
- Initiating and managing activities for the improvement and greater efficiency of software development and SQA activities.

In Software maintenance (product-oriented)

- Assuring with an acceptable level of confidence that the software maintenance activities will conform to the functional technical requirements.
- Assuring with an acceptable level of confidence that the software maintenance activities will conform to managerial schedling and budgetary requirements.
- Initiating and managing activities to improve and increase the efficiency of software maintenance and SQA activities. This involves improving the prospects of achieving functional and managerial requirements while reducing costs.

Organizing for Quality Assurance

The quality assurance organizational framework that operates within the organizational structure includes the following participants −

Managers

- Top management executives, especially the executive directly in charge of software quality assurance
- Software development and maintenance department managers
- Software testing department managers
- Project managers and team leaders of development and maintenance projects
- Leaders of software testing teams

SQA professionals and interested practitioners −

- SQA trustees
- SQA committee members
- SQA forum members
- SQA unit team members

Only the managers and employees of the software testing department are occupied full time in the performance of SQA tasks. The others dedicate part of their time to quality issues, whether during fulfillment of their managerial functions or professional tasks, or as volunteers in others, most often a SQA committee, a SQA forum, or as SQA trustees.

Requirement analysis

In systems engineering and software engineering, requirements analysis focuses on the tasks that determine the needs or conditions to meet the new or altered product or project, taking account of the possibly conflicting requirements of the various stakeholders, analyzing, documenting, validating and managing software or system requirements.

Requirements analysis is critical to the success or failure of a systems or software project. The requirements should be documented, actionable, measurable, testable, traceable, related to identified business needs or opportunities, and defined to a level of detail sufficient for system design.

Software prototyping

It is the activity of creating prototypes of software applications, i.e., incomplete versions of the software program being developed. It is an activity that can occur in software development and is comparable to prototyping as known from other fields, such as mechanical engineering or manufacturing.

A prototype typically simulates only a few aspects of, and may be completely different from, the final product.

1.Horizontal prototype

A common term for a user interface prototype is the **horizontal prototype**. It provides a broad view of an entire system or subsystem, focusing on user interaction more than low-level system functionality, such as database access. Horizontal prototypes are useful for:

Confirmation of user interface requirements and system scope,

Demonstration version of the system to obtain buy-in from the business,

Develop preliminary estimates of development time, cost and effort.

2.Vertical prototype

A vertical prototype is an enhanced complete elaboration of a single subsystem or function. It is useful for obtaining detailed requirements for a given function, with the following benefits:

- Refinement database design,
- Obtain information on data volumes and system interface needs, for network sizing and performance engineering,
- Clarify complex requirements by drilling down to actual system functionality.

Advantages of prototyping

There are many advantages to using prototyping in software development – some tangible, some abstract.

Reduced time and costs: Prototyping can improve the quality of requirements and specifications provided to developers. Because changes cost exponentially more to implement as they are detected later in development, the early determination of *what the user really wants* can result in faster and less expensive software.[

Improved and increased user involvement: Prototyping requires user involvement and allows them to see and interact with a prototype allowing them to provide better and more complete feedback and specifications

Disadvantages of prototyping

Insufficient analysis:
User confusion of prototype and finished system:
Developer misunderstanding of user objectives
Developer attachment to prototype:
Excessive development time of the prototype.
Expense of implementing prototyping:

Unit: 2 Distributed Databases

Definition

A distributed database is a collection of multiple interconnected databases, which are spread physically across various locations that communicate via a computer network.

Features

Databases in the collection are logically interrelated with each other. Often they represent a single logical database.

Data is physically stored across multiple sites. Data in each site can be managed by a DBMS independent of the other sites.

A distributed database is not a loosely connected file system.
A distributed database incorporates transaction processing, but it is not synonymous with a transaction processing system.
Distributed Database Management System
A distributed database management system (DDBMS) is a centralized software system that manages a distributed database in a manner as if it were all stored in a single location.

Advantages of Distributed Databases

Modular Development − If the system needs to be expanded to new locations or new units, in centralized database systems, the action requires substantial efforts and disruption in the existing functioning. However, in distributed databases, the work simply requires adding new computers and local data to the new site and finally connecting them to the distributed system, with no interruption in current functions.

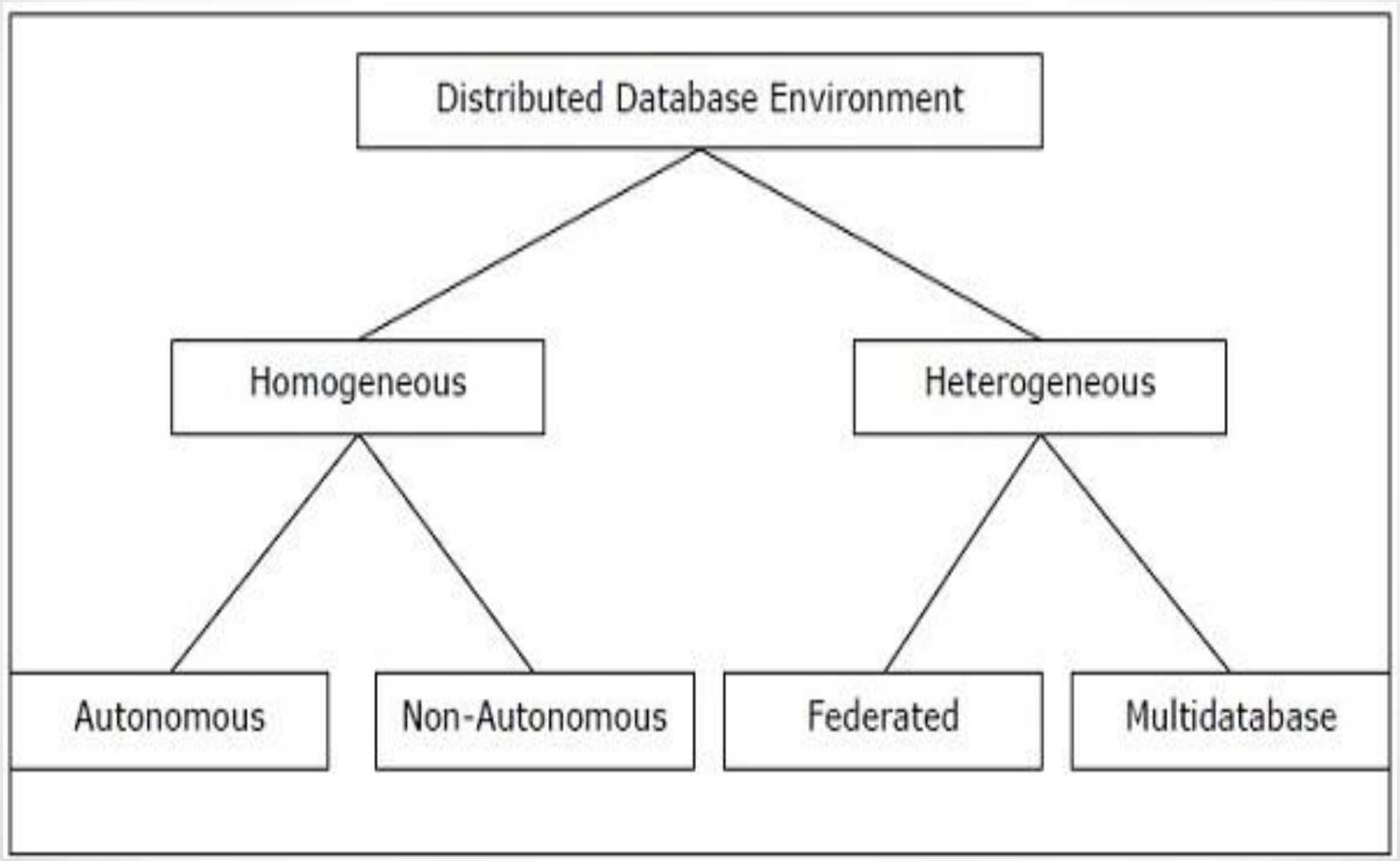
More Reliable − In case of database failures, the total system of centralized databases comes to a halt. However, in distributed systems, when a component fails, the functioning of the system continues may be at a reduced performance. Hence DDBMS is more reliable.

Better Response − If data is distributed in an efficient manner, then user requests can be met from local data itself, thus providing faster response. On the other hand, in centralized systems, all queries have to pass through the central computer for processing, which increases the response time.

Lower Communication Cost − In distributed database systems, if data is located locally where it is mostly used, then the communication costs for data manipulation can be minimized. This is not feasible in centralized systems.

Types of Distributed Databases

Distributed databases can be broadly classified into homogeneous and heterogeneous distributed database environments, each with further sub-divisions, as shown in the following illustration


Distributed Database Environment
Homogeneous
Heterogeneous
Autonomous
Non-Autonomous
Federated
Multidatabase

Homogeneous Distributed Databases

In a homogeneous distributed database, all the sites use identical DBMS and operating systems. Its properties are −

The sites use very similar software.

The sites use identical DBMS or DBMS from the same vendor.

Each site is aware of all other sites and cooperates with other sites to process user requests.

The database is accessed through a single interface as if it is a single database.

There are two types of homogeneous distributed database −

Autonomous − Each database is independent that functions on its own. They are integrated by a controlling application and use message passing to share data updates.

Non-autonomous − Data is distributed across the homogeneous nodes and a central or master DBMS co-ordinates data updates across the sites.

Heterogeneous Distributed Databases

In a heterogeneous distributed database, different sites have different operating systems, DBMS products and data models. Its properties are −

Different sites use dissimilar schemas and software.

The system may be composed of a variety of DBMSs like relational, network, hierarchical or object oriented.

Query processing is complex due to dissimilar schemas.

Transaction processing is complex due to dissimilar software.

A site may not be aware of other sites and so there is limited co-operation in processing user requests.

Types of Heterogeneous Distributed Databases

Federated − The heterogeneous database systems are independent in nature and integrated together so that they function as a single database system.

Un-federated − The database systems employ a central coordinating module through which the databases are accessed.

Distributed DBMS Architectures

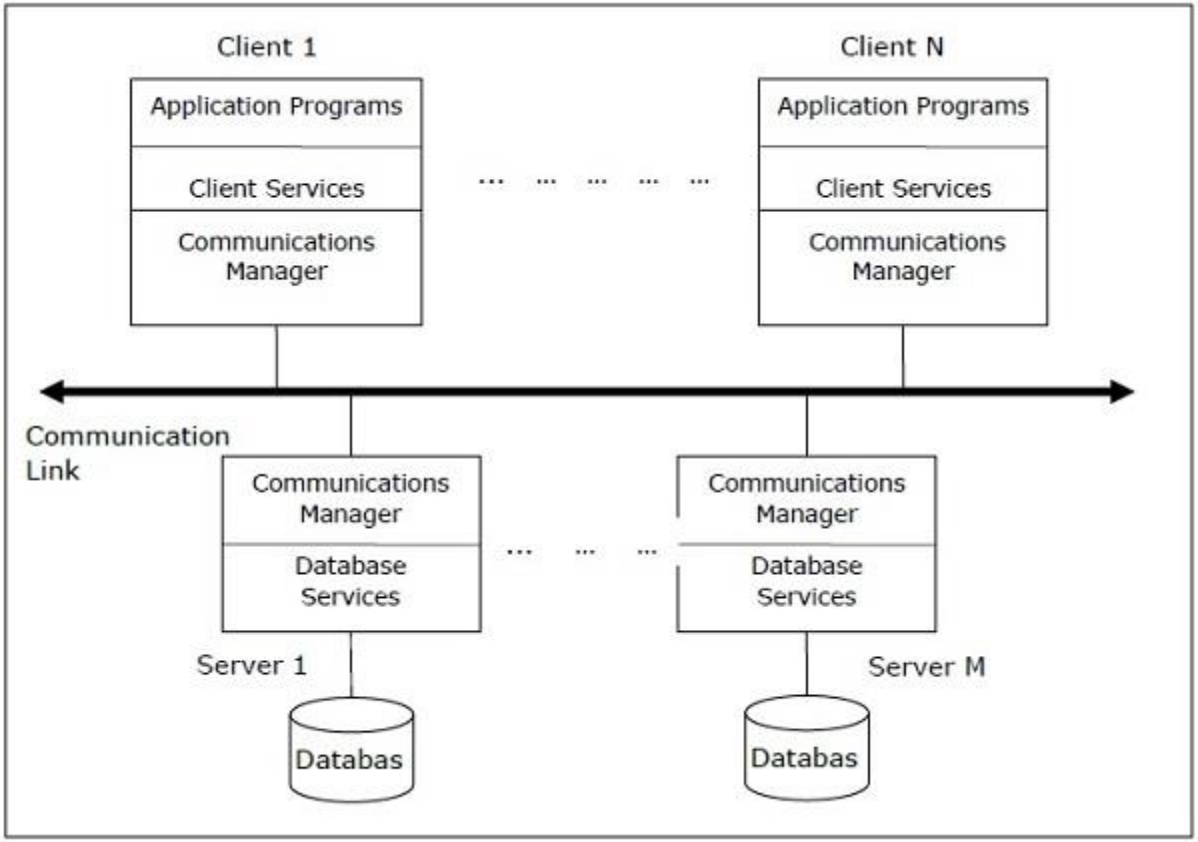
DDBMS architectures are generally developed depending on three parameters −

Distribution − It states the physical distribution of data across the different sites.

Autonomy − It indicates the distribution of control of the database system and the degree to which each constituent DBMS can operate independently.

Heterogeneity − It refers to the uniformity or dissimilarity of the data models, system components and databases.

Multiple Server Multiple Client (shown in the following diagram)

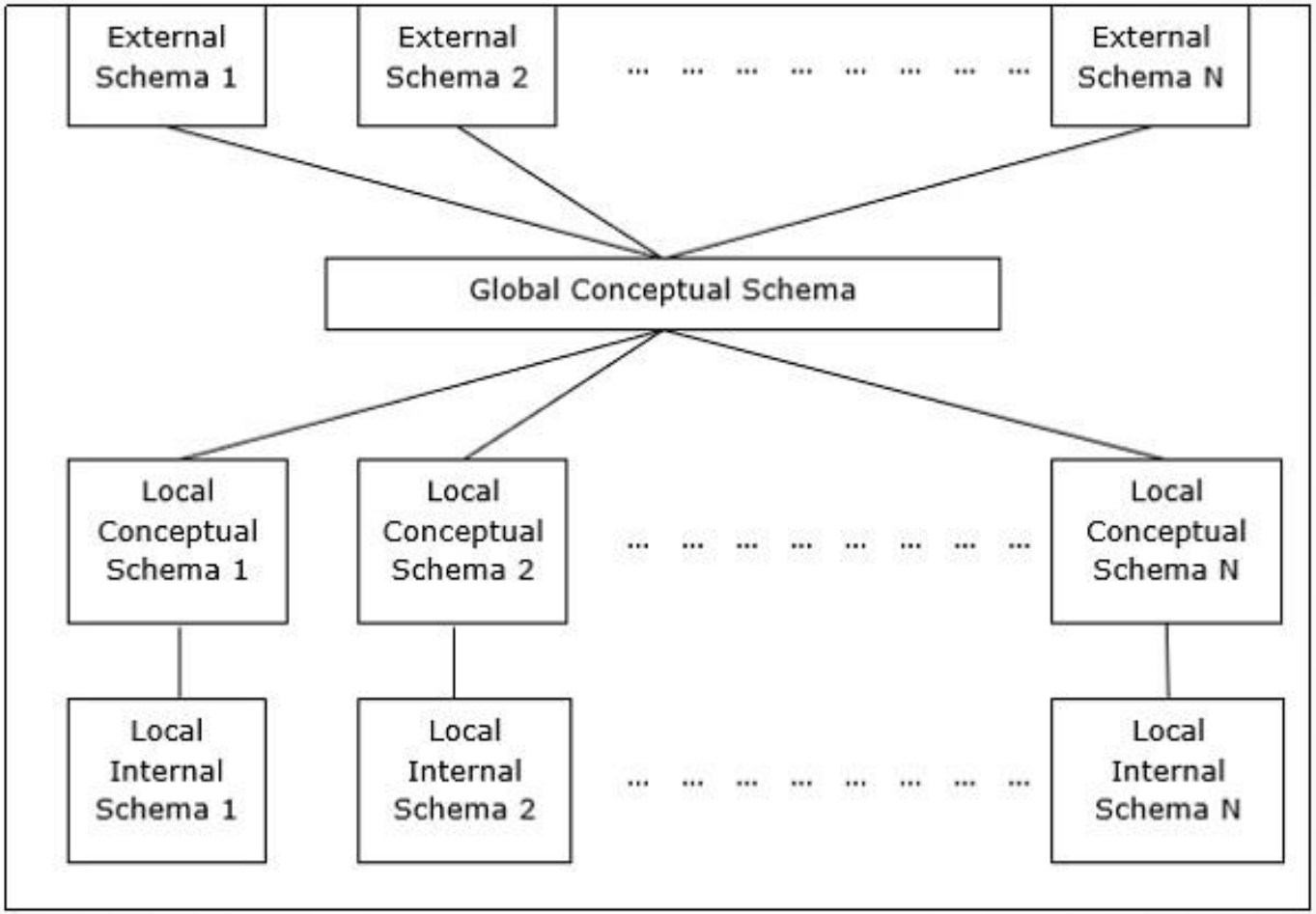


Peer- to-Peer Architecture for DDBMS

In these systems, each peer acts both as a client and a server for imparting database services. The peers share their resource with other peers and co-ordinate their activities.

This architecture generally has four levels of schemas −
Global Conceptual Schema − Depicts the global logical view of data.
Local Conceptual Schema − Depicts logical data organization at each site.
Local Internal Schema − Depicts physical data organization at each site.
External Schema − Depicts user view of data

Peer- to-Peer Architecture for DDBMS



Fragmentation

Fragmentation is the task of dividing a table into a set of smaller tables. The subsets of the table are called **fragments**. Fragmentation can be of three types: horizontal, vertical, and hybrid (combination of horizontal and vertical). Horizontal fragmentation can further be classified into two techniques: primary horizontal fragmentation and derived horizontal fragmentation.

The three fragmentation techniques are −

- Vertical fragmentation
- Horizontal fragmentation
- Hybrid fragmentation

Vertical Fragmentation

In vertical fragmentation, the fields or columns of a table are grouped into fragments. In order to maintain reconstructiveness, each fragment should contain the primary key field(s) of the table. Vertical fragmentation can be used to enforce privacy of data.

For example, let us consider that a University database keeps records of all registered students in a Student table having the following schema.

STUDENT

Reg_No	Name	Course	Address	Semester	Fees	Marks

Advantages of Fragmentation

- Since data is stored close to the site of usage, efficiency of the database system is increased.
- Local query optimization techniques are sufficient for most queries since data is locally available.

Since irrelevant data is not available at the sites, security and privacy of the database system can be maintained

Horizontal Fragmentation

Horizontal fragmentation groups the tuples of a table in accordance to values of one or more fields. Horizontal fragmentation should also confirm to the rule of reconstructiveness. Each horizontal fragment must have all columns of the original base table.

For example, in the student schema, if the details of all students of Computer Science Course needs to be maintained at the School of Computer Science, then the designer will horizontally fragment the database as follows −

```
CREATE COMP_STD AS
   SELECT * FROM STUDENT
   WHERE COURSE = "Computer Science
```

Hybrid Fragmentation

In hybrid fragmentation, a combination of horizontal and vertical fragmentation techniques are used. This is the most flexible fragmentation technique since it generates fragments with minimal extraneous information. However, reconstruction of the original table is often an expensive task.

Hybrid fragmentation can be done in two alternative ways −

- At first, generate a set of horizontal fragments; then generate vertical fragments from one or more of the horizontal fragments.
- At first, generate a set of vertical fragments; then generate horizontal fragments from one or more of the vertical fragments.

Object Relational Database

Abstract data types which are structures consisting of a number of different elements, each of which uses one of the base data types provided within the Oracle product

Object tables These are tables created within Oracl which have column values that are basedonADTs. Therefore and address ADTs described above, the table will be an object table.

Nested tables A nested table is a table within a table. It is a collection of rows, represented as a column in the main table. For each record in the main table, the nested table may contain multiple rows. This can be considered as a way of storing a one to-many relationship within one table

Varying arrays A varying array, or varray, is a collection of objects, each with the same data type. The size of the array is preset when it is created. The varying array is treated like a column in a main table. Conceptually, it is a nested table, with a preset limit on its number of rows. Varrays also then allow us to store up to a preset number of repeating values in a table. The data type for a varray is determined by the type of data to be stored.

• **Blob:** Stores any kind of data in binary format. Typically used for multimedia data such as images, audio and video.

• **Clob**: Stores string data in the database character set format. Used for large strings or documents that use the database character set exclusively. Characters in the database character set are in a fixed-width format.

• **Nclob:** Stores string data in National Character Set format. Used for large strings or documents in the National Character Set. Supports characters of varying-width format.

• **Bfile:** Is a pointer to a binary file stored outside of the database in the host operating system file system, but accessible from database tables. It is possible to have multiple large objects (including different types) per table.

Unit 3: Data Warehouse

Data warehousing is the process of constructing and using a data warehouse. A data warehouse is constructed by integrating data from multiple heterogeneous sources that support analytical reporting, structured and/or ad hoc queries, and decision making. Data warehousing involves data cleaning, data integration, and data consolidations.

Using Data Warehouse Information

There are decision support technologies that help utilize the data available in a data warehouse. These technologies help executives to use the warehouse quickly and effectively. They can gather data, analyze it, and take decisions based on the information present in the warehouse. The information gathered in a warehouse can be used in any of the following domains −

•Tuning Production Strategies − The product strategies can be well tuned by repositioning the products and managing the product portfolios by comparing the sales quarterly or yearly.

•Customer Analysis − Customer analysis is done by analyzing the customer's buying preferences, buying time, budget cycles, etc.

•Operations Analysis − Data warehousing also helps in customer relationship management, and making environmental corrections. The information also allows us to analyze business operations

To integrate heterogeneous databases, we have two approaches −

- Query-driven Approach
- Update-driven Approach

Process of Query-Driven Approach

- When a query is issued to a client side, a metadata dictionary translates the query into an appropriate form for individual heterogeneous sites involved.
- Now these queries are mapped and sent to the local query processor.
- The results from heterogeneous sites are integrated into a global answer set.

The following are the functions of data warehouse tools and utilities −

- **Data Extraction** − Involves gathering data from multiple heterogeneous sources.
- **Data Cleaning** − Involves finding and correcting the errors in data.
- **Data Transformation** − Involves converting the data from legacy format to warehouse format.
- **Data Loading** − Involves sorting, summarizing, consolidating, checking integrity, and building indices and partitions.
- **Refreshing** − Involves updating from data sources to warehouse.

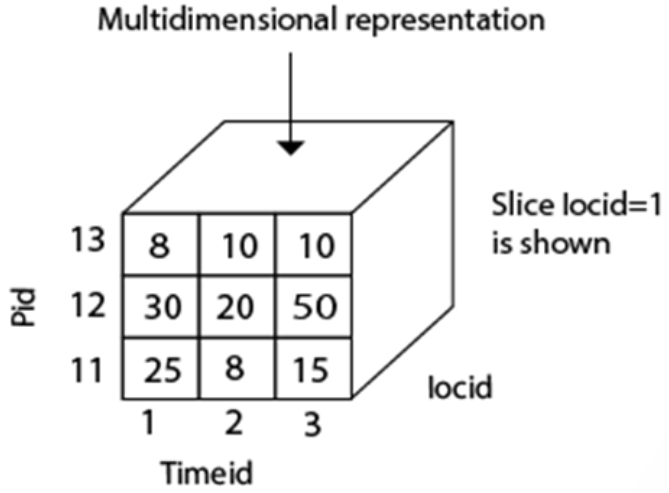
What is Multi-Dimensional Data Model?

A multidimensional model views data in the form of a data-cube. A data cube enables data to be modeled and viewed in multiple dimensions. It is defined by dimensions and facts.

A multidimensional data model is organized around a central theme, for example, sales. This theme is represented by a fact table. Facts are numerical measures. The fact table contains the names of the facts or measures of the related dimensional tables.

Tabular representation

Pid	Timeid	locid	Sales
11	1	1	25
11	2	1	8
11	3	1	15
12	1	1	30
12	2	1	20
12	3	1	50
13	1	1	8
13	2	1	10
13	3	1	10
11	1	2	35

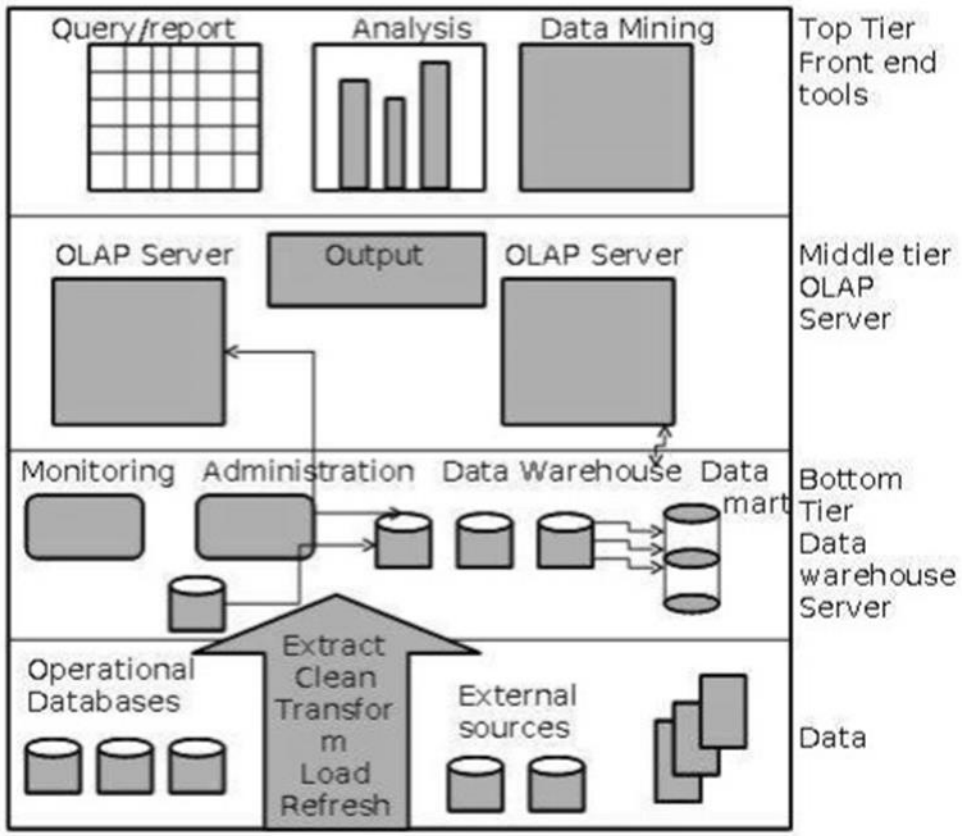


Three-Tier Data Warehouse Architecture

Generally a data warehouses adopts a three-tier architecture. Following are the three tiers of the data warehouse architecture.

- **Bottom Tier** − The bottom tier of the architecture is the data warehouse database server. It is the relational database system. We use the back end tools and utilities to feed data into the bottom tier. These back end tools and utilities perform the Extract, Clean, Load, and refresh functions.
- **Middle Tier** − In the middle tier, we have the OLAP Server that can be implemented in either of the following ways.
  - By Relational OLAP (ROLAP), which is an extended relational database management system. The ROLAP maps the operations on multidimensional data to standard relational operations..

Top-Tier – This tier is the front-end client layer. This layer holds the query tools and reporting tools, analysis tools and data mining tools.



Data Mart

Data mart contains a subset of organization-wide data. This subset of data is valuable to specific groups of an organization.

In other words, we can claim that data marts contain data specific to a particular group. For example, the marketing data mart may contain data related to items, customers, and sales. Data marts are confined to subjects.

Enterprise Warehouse

- An enterprise warehouse collects all the information and the subjects spanning an entire organization
- It provides us enterprise-wide data integration.
- The data is integrated from operational systems and external information providers.
- This information can vary from a few gigabytes to hundreds of gigabytes, terabytes or beyond.

Data cleaning

1. Fill in missing values (attribute or class value):
    - Ignore the tuple: usually done when class label is missing.
    - Use the attribute mean (or majority nominal value) to fill in the missing value.
    - Use the attribute mean (or majority nominal value) for all samples belonging to the same class.
    - Predict the missing value by using a learning algorithm: consider the attribute with the missing value as a dependent (class) variable and run a learning algorithm (usually Bayes or decision tree) to predict the missing value.
2. Identify outliers and smooth out noisy data:
    - Binning
        - Sort the attribute values and partition them into bins (see "Unsupervised discretization" below);
        - Then smooth by bin means, bin median, or bin boundaries.
    - Clustering: group values in clusters and then detect and remove outliers (automatic or manual)
    - Regression: smooth by fitting the data into regression functions.
3. Correct inconsistent data: use domain knowledge or expert decision.

Data transformation

Normalization:

- Scaling attribute values to fall within a specified range.
  - Example: to transform V in [min, max] to V' in [0,1], apply V'=(V-Min)/(Max-Min)
- Scaling by using mean and standard deviation (useful when min and max are unknown or when there are outliers): V'=(V-Mean)/StDev

Aggregation: moving up in the concept hierarchy on numeric attributes.

Generalization: moving up in the concept hierarchy on nominal attributes.

Attribute construction: replacing or adding new attributes inferred by existing attributes.

Data reduction

Reducing the number of attributes

- Data cube aggregation: applying roll-up, slice or dice operations.
- Removing irrelevant attributes: attribute selection (filtering and wrapper methods), searching the attribute space (see Lecture 5: Attribute-oriented analysis).
- Principle component analysis (numeric attributes only): searching for a lower dimensional space that can best represent the data..

Reducing the number of attribute values

- Binning (histograms): reducing the number of attributes by grouping them into intervals (bins).
- Clustering: grouping values in clusters.
- Aggregation or generalization

Reducing the number of tuples

- Sampling

Unit 4: Network Security

Cryptography

Cryptography is the science of encrypting and decrypting data. Based on complex mathematics, cryptography provides several important information security services such as authentication, confidentiality, integrity, and non-repudiation. Cryptographic protocols and applications make cryptography user-friendly and enable users to secure their data without having to carry out the complex mathematics themselves

cryptography is classified as either symmetric or asymmetric key cryptography.

Both symmetric and asymmetric key cryptography provide data confidentiality.

Asymmetric key encryption is sometimes called public key encryption. Digital signatures, one of the by-products of public key cryptography, enable the verification of authenticity, integrity, and non-repudiation.

Encryption

To encrypt a letter, a user needs to write a key underneath the plaintext. The plaintext letter is placed on the top and the key letter on the left. The cross section achieved between two letters is the plain text. It is described in the example below −

Decryption

To decrypt a letter, user takes the key letter on the left and finds cipher text letter in that row. The plain text letter is placed at the top of the column where the user can find the cipher text letter.

Difference between Substitution Cipher Technique and Transposition Cipher Technique

Both **Substitution cipher technique** and **Transposition cipher technique** are the types of Traditional cipher which are used to convert the plain text into cipher text.

Substitution Cipher Technique:

In Substitution Cipher Technique plain text characters are replaced with other characters, numbers and symbols as well as in substitution Cipher Technique, character's identity is changed while its position remains unchanged.

Transposition Cipher Technique:

Transposition Cipher Technique rearranges the position of the plain text's characters. In transposition Cipher Technique, The position of the character is changed but character's identity is not changed.

One-time pad cipher

One-time pad cipher is a type of Vignere cipher which includes the following features −

It is an unbreakable cipher.

The key is exactly same as the length of message which is encrypted.

The key is made up of random symbols.

As the name suggests, key is used one time only and never used again for any other message to be encrypted.

Due to this, encrypted message will be vulnerable to attack for a cryptanalyst. The key used for a one-time pad cipher is called pad, as it is printed on pads of paper.

Why is it Unbreakable?

The key is unbreakable owing to the following features −

The key is as long as the given message.

The key is truly random and specially auto-generated.

Key and plain text calculated as modulo 10/26/2.

Each key should be used once and destroyed by both sender and receiver.

There should be two copies of key: one with the sender and other with the receiver

AES – The Advances Encryption Standard;

The more popular and widely adopted symmetric encryption algorithm likely to be encountered nowadays is the Advanced Encryption Standard (AES). It is found at least six time faster than triple DES.

A replacement for DES was needed as its key size was too small. With increasing computing power, it was considered vulnerable against exhaustive key search attack. Triple DES was designed to overcome this drawback but it was found slow.

The features of AES are as follows −

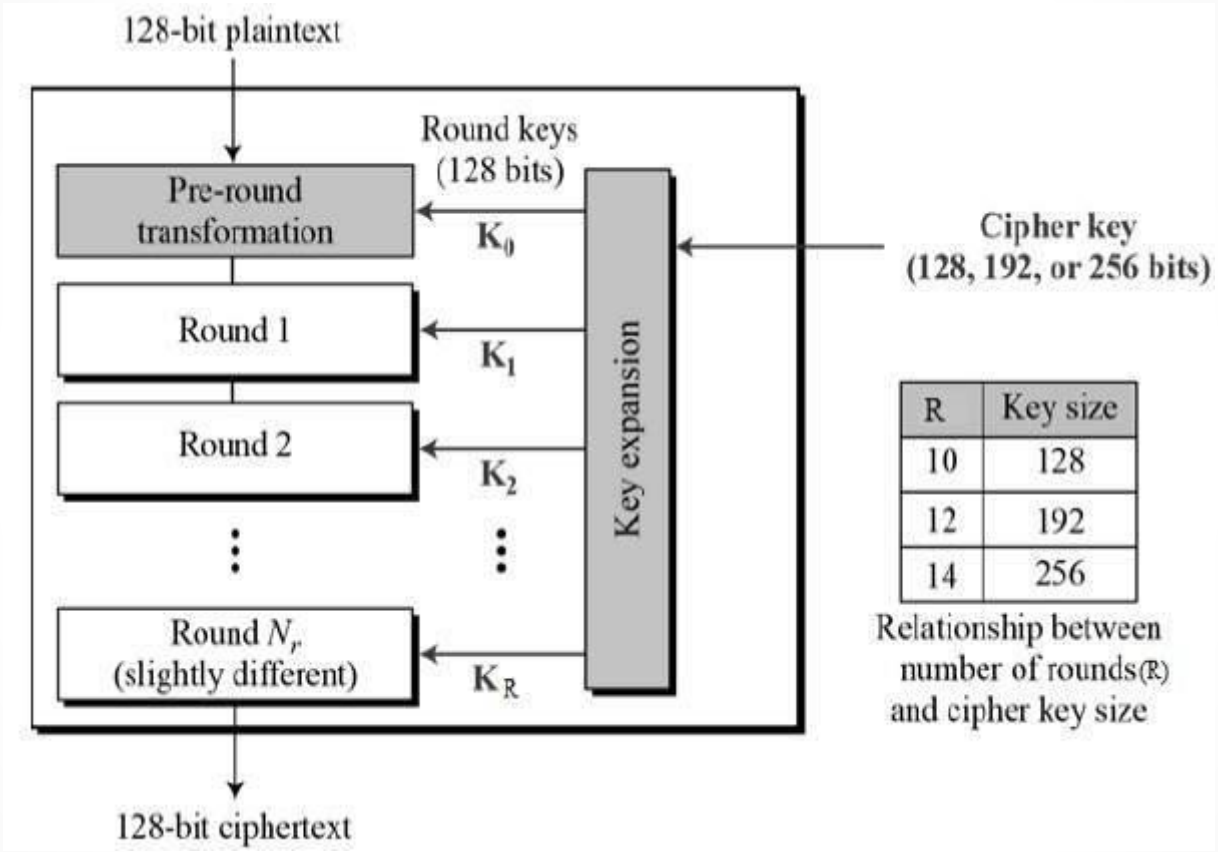
- Symmetric key symmetric block cipher
- 128-bit data, 128/192/256-bit keys
- Stronger and faster than Triple-DES
- Provide full specification and design details
- Software implementable in C and Java

Operation of AES

AES is an iterative rather than Feistel cipher. It is based on ‘substitution–permutation network’. It comprises of a series of linked operations, some of which involve replacing inputs by specific outputs (substitutions) and others involve shuffling bits around (permutations).

Interestingly, AES performs all its computations on bytes rather than bits. Hence, AES treats the 128 bits of a plaintext block as 16 bytes. These 16 bytes are arranged in four columns and four rows for processing as a matrix −

Unlike DES, the number of rounds in AES is variable and depends on the length of the key. AES uses 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys. Each of these rounds uses a different 128-bit round key, which is calculated

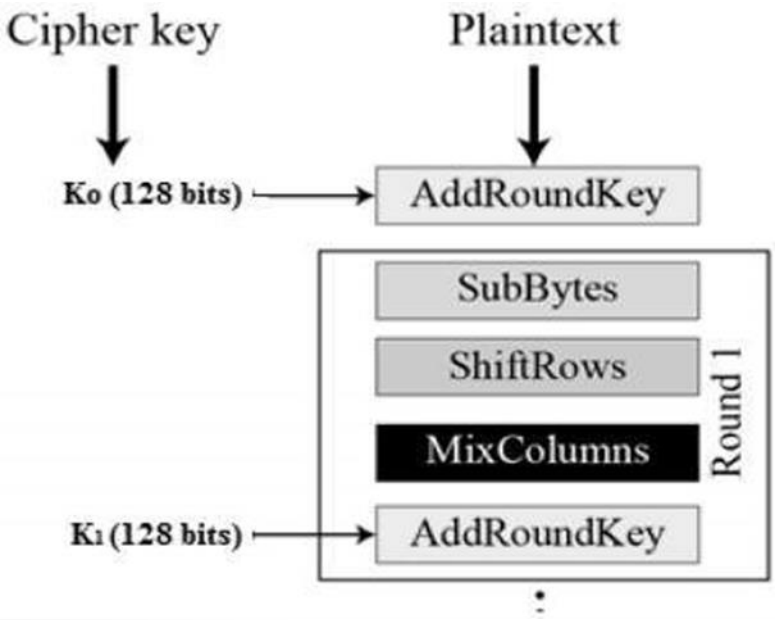


R	Key size
10	128
12	192
14	256

Relationship between number of rounds(R) and cipher key size

Encryption Process

Here, we restrict to description of a typical round of AES encryption. Each round comprise of four sub-processes. The first round process is depicted below −



Byte Substitution (SubBytes)

The 16 input bytes are substituted by looking up a fixed table (S-box) given in design. The result is in a matrix of four rows and four columns.

Shiftrows

Each of the four rows of the matrix is shifted to the left. Any entries that 'fall off' are re-inserted on the right side of row. Shift is carried out as follows −

- First row is not shifted.
- Second row is shifted one (byte) position to the left.
- Third row is shifted two positions to the left.
- Fourth row is shifted three positions to the left.
- The result is a new matrix consisting of the same 16 bytes but shifted with respect to each other.

MixColumns

Each column of four bytes is now transformed using a special mathematical function. This function takes as input the four bytes of one column and outputs four completely new bytes, which replace the original column. The result is another new matrix consisting of 16 new bytes. It should be noted that this step is not performed in the last round.

Addroundkey

The 16 bytes of the matrix are now considered as 128 bits and are XORed to the 128 bits of the round key. If this is the last round then the output is the ciphertext. Otherwise, the resulting 128 bits are interpreted as 16 bytes and we begin another similar round.

Decryption Process

The process of decryption of an AES ciphertext is similar to the encryption process in the reverse order. Each round consists of the four processes conducted in the reverse order −

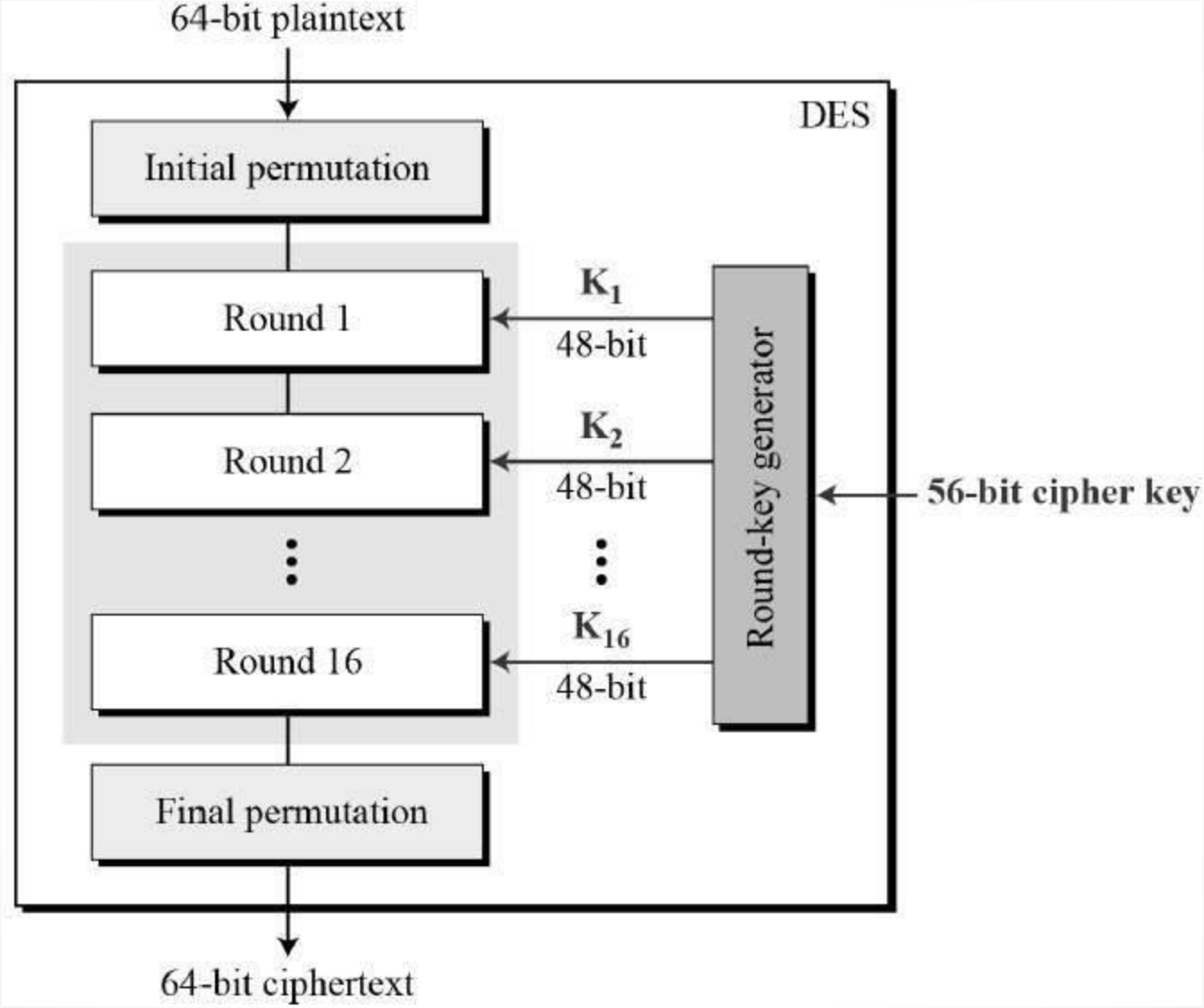
- Add round key
- Mix columns
- Shift rows
- Byte substitution

Since sub-processes in each round are in reverse manner, unlike for a Feistel Cipher, the encryption and decryption algorithms needs to be separately implemented, although they are very closely related

Data Encryption Standard (DES)

The Data Encryption Standard (DES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST).

DES is an implementation of a Feistel Cipher. It uses 16 round Feistel structure. The block size is 64-bit. Though, key length is 64-bit, DES has an effective key length of 56 bits, since 8 of the 64 bits of the key are not used by the encryption algorithm (function as check bits only). General Structure of DES is depicted in the following illustration −

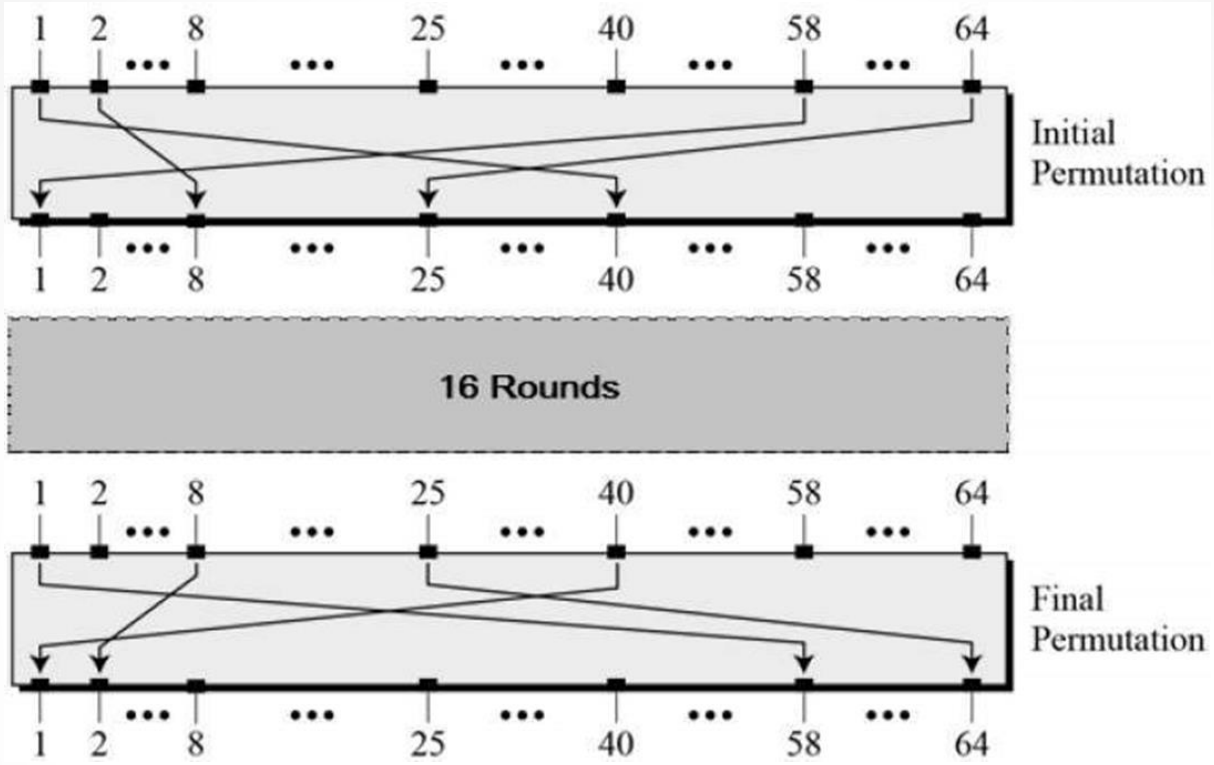

64-bit plaintext
DES
Initial permutation
Round 1
Round 2
Round 16
Final permutation
K1
48-bit
K2
48-bit
K16
48-bit
Round-key generator
56-bit cipher key
64-bit ciphertext

Since DES is based on the Feistel Cipher, all that is required to specify DES is –

- Round function
- Key schedule
- Any additional processing − Initial and final permutation

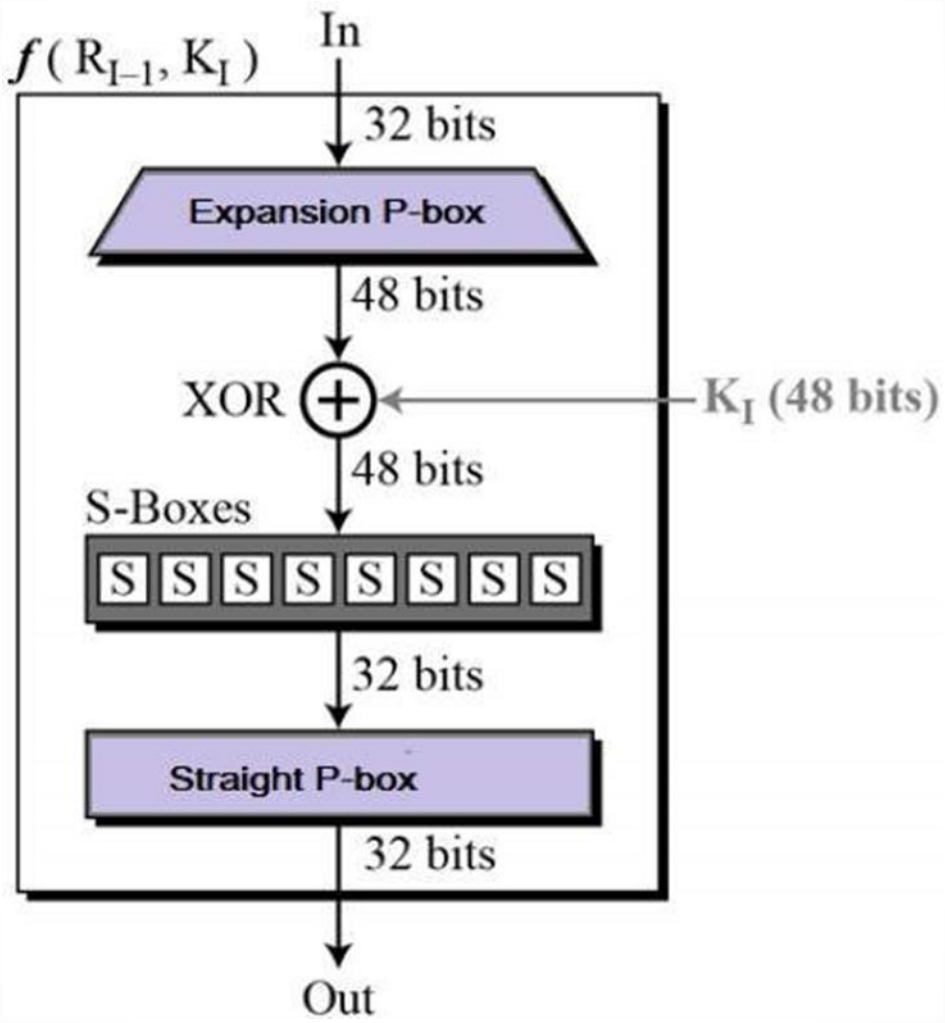
Initial and Final Permutation

The initial and final permutations are straight Permutation boxes (P-boxes) that are inverses of each other. They have no cryptography significance in DES. The initial and final permutations are shown as follows



Round Function

The heart of this cipher is the DES function, *f*. The DES function applies a 48-bit key to the rightmost 32 bits to produce a 32-bit output.


$f(R_{I-1}, K_I)$
In
32 bits
Expansion P-box
48 bits
XOR
K_I (48 bits)
48 bits
S-Boxes
S S S S S S S S
32 bits
Straight P-box
32 bits
Out

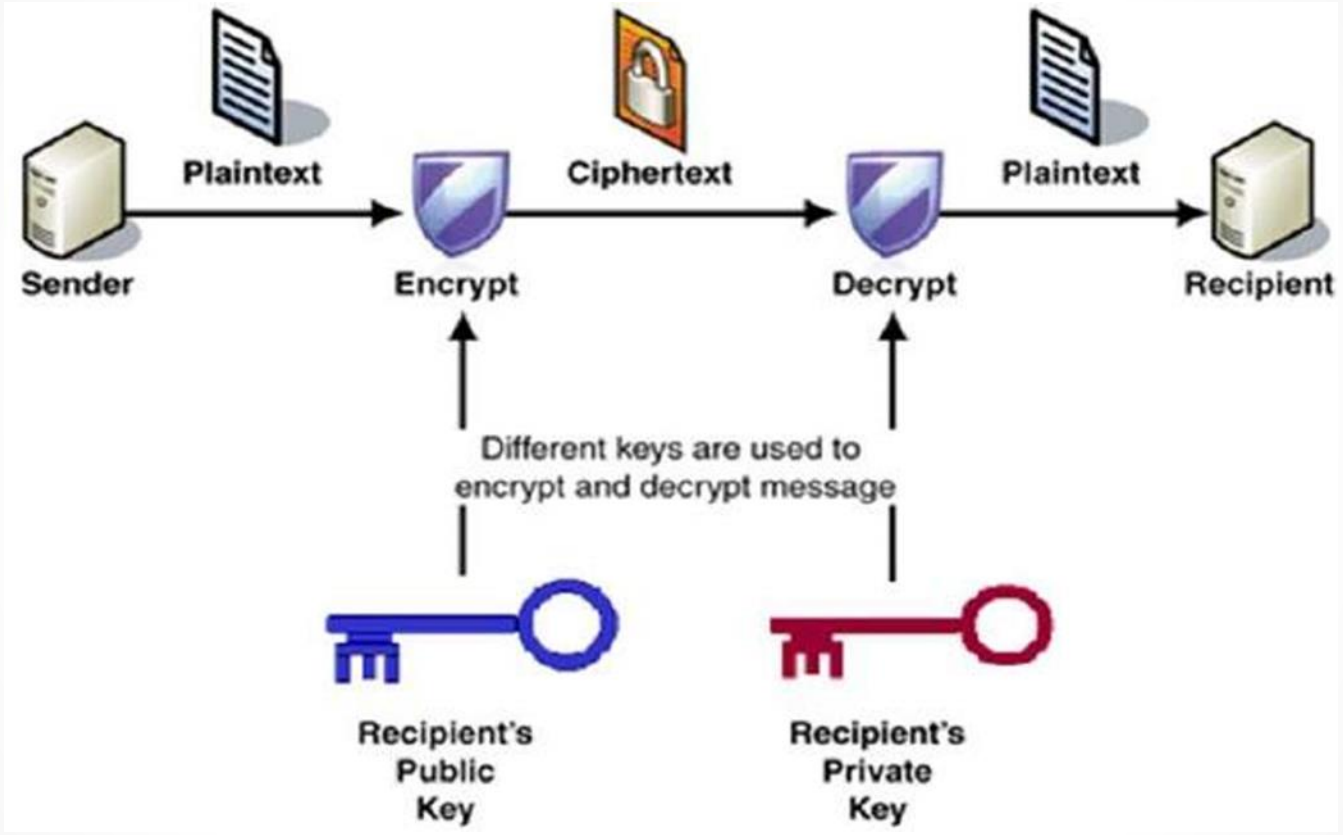
Public Key Cryptography

Unlike symmetric key cryptography, we do not find historical use of public-key cryptography. It is a relatively new concept.

Symmetric cryptography was well suited for organizations such as governments, military, and big financial corporations were involved in the classified communication.

With the spread of more unsecure computer networks in last few decades, a genuine need was felt to use cryptography at larger scale. The symmetric key was found to be non-practical due to challenges it faced for key management. This gave rise to the public key cryptosystems.

The process of encryption and decryption is depicted in the following illustration −


Plaintext
Ciphertext
Plaintext
Sender
Encrypt
Decrypt
Recipient
Different keys are used to encrypt and decrypt message
Recipient's Public Key
Recipient's Private Key

The most important properties of public key encryption scheme are −

- Different keys are used for encryption and decryption. This is a property which set this scheme different than symmetric encryption scheme.
- Each receiver possesses a unique decryption key, generally referred to as his private key.
- Receiver needs to publish an encryption key, referred to as his public key.
- Some assurance of the authenticity of a public key is needed in this scheme to avoid spoofing by adversary as the receiver. Generally, this type of cryptosystem involves trusted third party which certifies that a particular public key belongs to a specific person or entity only.

- Encryption algorithm is complex enough to prohibit attacker from deducing the plaintext from the ciphertext and the encryption (public) key.
- Though private and public keys are related mathematically, it is not be feasible to calculate the private key from the public key. In fact, intelligent part of any public-key cryptosystem is in designing a relationship between two keys.

There are three types of Public Key Encryption schemes. We discuss them in following sections −

RSA Cryptosystem

This cryptosystem is one the initial system. It remains most employed cryptosystem even today. The system was invented by three scholars **Ron Rivest, Adi Shamir,** and **Len Adleman** and hence, it is termed as RSA cryptosystem.

We will see two aspects of the RSA cryptosystem, firstly generation of key pair and secondly encryption-decryption algorithms.

Generation of RSA Key Pair

Each person or a party who desires to participate in communication using encryption needs to generate a pair of keys, namely public key and private key. The process followed in the generation of keys is described below −

- **Generate the RSA modulus (n)**
  - Select two large primes, p and q.
  - Calculate n=p*q. For strong unbreakable encryption, let n be a large number, typically a minimum of 512 bits.
- **Find Derived Number (e)**
  - Number **e** must be greater than 1 and less than (p − 1)(q − 1).

There must be no common factor for e and (p − 1)(q − 1) except for 1. In other words two numbers e and (p – 1)(q – 1) are coprime

- **Form the public key**
  - The pair of numbers (n, e) form the RSA public key and is made public.
  - Interestingly, though n is part of the public key, difficulty in factorizing a large prime number ensures that attacker cannot find in finite time the two primes (p & q) used to obtain n. This is strength of RSA.

- **Generate the private key**
  - Private Key d is calculated from p, q, and e. For given n and e, there is unique number d.
  - Number d is the inverse of e modulo (p - 1)(q – 1). This means that d is the number less than (p - 1)(q - 1) such that when multiplied by e, it is equal to 1 modulo (p - 1)(q - 1).
  - This relationship is written mathematically as follows −

$$ed = 1 \bmod (p - 1)(q - 1)$$

RSA Decryption

- The decryption process for RSA is also very straightforward. Suppose that the receiver of public-key pair (n, e) has received a ciphertext C.
- Receiver raises C to the power of his private key d. The result modulo n will be the plaintext P.

Plaintext = $C^d \bmod n$

- Returning again to our numerical example, the ciphertext C = 82 would get decrypted to number 10 using private key 29 −

Plaintext = $82^{29} \bmod 91 = 10$

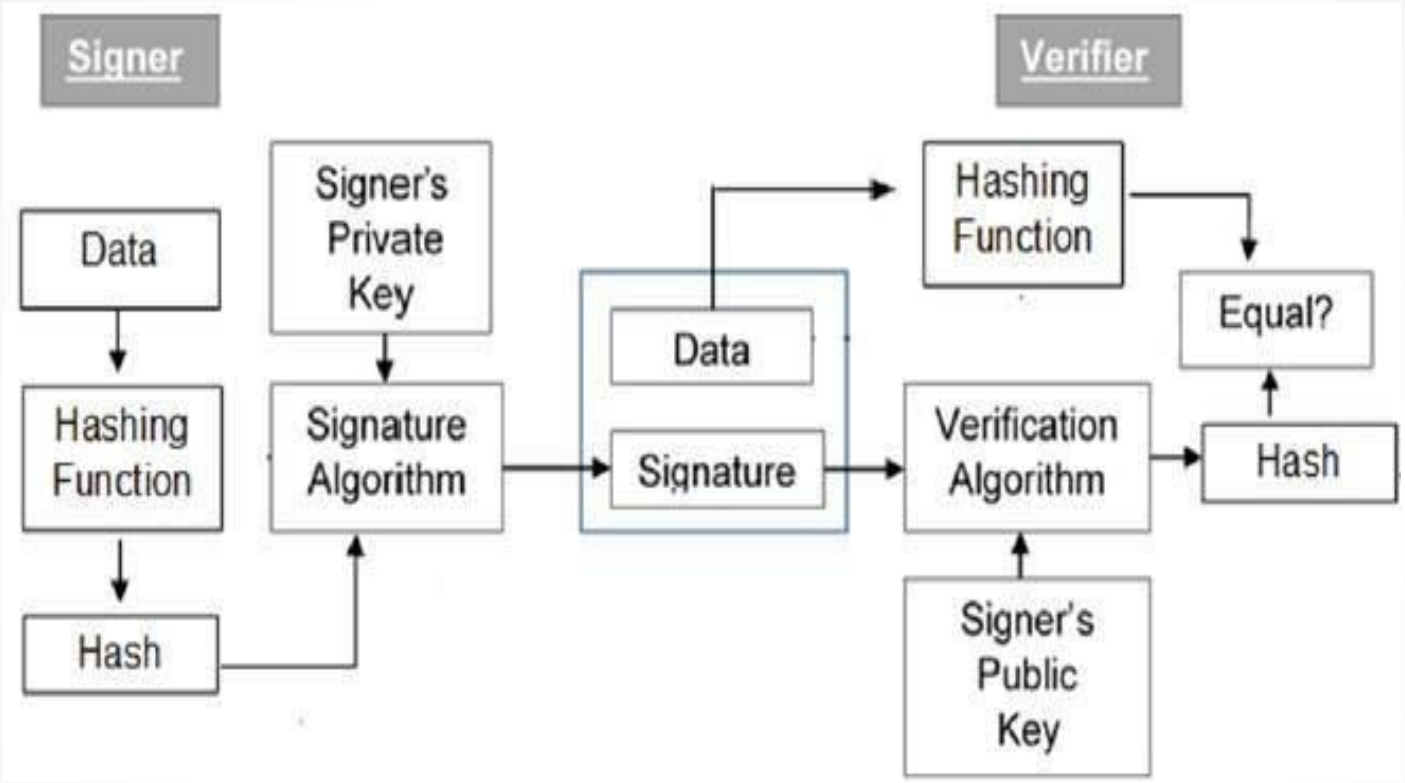
Digital signatures

Digital signatures are the public-key primitives of message authentication. In the physical world, it is common to use handwritten signatures on handwritten or typed messages. They are used to bind signatory to the message.

Similarly, a digital signature is a technique that binds a person/entity to the digital data. This binding can be independently verified by receiver as well as any third party.

Digital signature is a cryptographic value that is calculated from the data and a secret key known only by the signer.

In real world, the receiver of message needs assurance that the message belongs to the sender and he should not be able to repudiate the origination of that message. This requirement is very crucial in business applications, since likelihood of a dispute over exchanged data is very high.


Signer
Verifier
Data
Signer's Private Key
Hashing Function
Signature Algorithm
Hash
Data
Signature
Hashing Function
Equal?
Verification Algorithm
Hash
Signer's Public Key

- **Data Integrity** − In case an attacker has access to the data and modifies it, the digital signature verification at receiver end fails. The hash of modified data and the output provided by the verification algorithm will not match. Hence, receiver can safely deny the message assuming that data integrity has been breached.
- **Non-repudiation** − Since it is assumed that only the signer has the knowledge of the signature key, he can only create unique signature on a given data. Thus the receiver can present data and the digital signature to a third party as evidence if any dispute arises in the future.

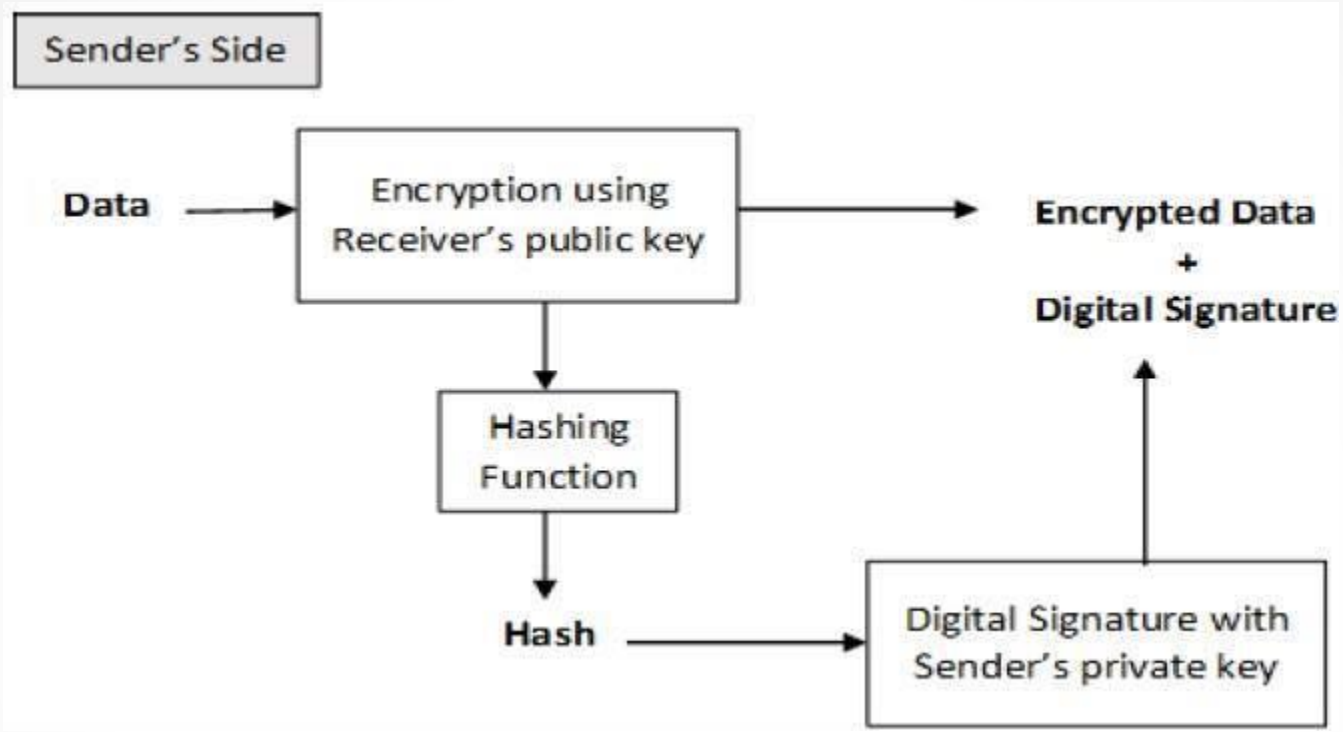
By adding public-key encryption to digital signature scheme, we can create a cryptosystem that can provide the four essential elements of security namely − Privacy, Authentication, Integrity, and Non-repudiation.

Encryption with Digital Signature

In many digital communications, it is desirable to exchange an encrypted messages than plaintext to achieve confidentiality. In public key encryption scheme, a public (encryption) key of sender is available in open domain, and hence anyone can spoof his identity and send any encrypted message to the receiver.

This makes it essential for users employing PKC for encryption to seek digital signatures along with encrypted data to be assured of message authentication and non-repudiation.

This can archived by combining digital signatures with encryption scheme. Let us briefly discuss how to achieve this requirement. There are **two possibilities, sign-then-encrypt** and **encrypt-then-sign**.


Sender's Side
Data
Encryption using
Receiver's public key
Encrypted Data
+
Digital Signature
Hashing
Function
Hash
Digital Signature with
Sender's private key

Message digests

The receiver after receiving the encrypted data and signature on it, first verifies the signature using sender's public key. After ensuring the validity of the signature, he then retrieves the data through decryption using his private key.

A message digest is a cryptographic hash function containing a string of digits created by a one-way hashing formula.

Message digests are designed to protect the integrity of a piece of data or media to detect changes and alterations to any part of a message. They are a type of cryptography utilizing hash values that can warn the copyright owner of any modifications applied to their work.

Unit 5 :Computing and Informatics

What is cloud computing, in simple terms?

Cloud computing is the delivery of on-demand computing services -- from applications to storage and processing power -- typically over the internet and on a pay-as-you-go basis.

What is the history of cloud computing?

Cloud computing as a term has been around since the early 2000s, but the concept of computing-as-a-service has been around for much, much longer -- as far back as the 1960s, when computer bureaus would allow companies to rent time on a mainframe, rather than have to buy one themselves.

These 'time-sharing' services were largely overtaken by the rise of the PC which made owning a computer much more affordable, and then in turn by the rise of corporate data centers where companies would store vast amounts of data.

But the concept of renting access to computing power has resurfaced again and again -- in the application service providers, utility computing, and grid computing of the late 1990s and early 2000s. This was followed by cloud computing, which really took hold with the emergence of software as a service and hyperscale cloud computing providers such as Amazon Web Services.

Benefits of Mobile Cloud Computing

- Extending battery lifetime
- Improving data storage capacity and processing power
- Improving reliability

Security Issues in Mobile cloud Computing

Cloud computing as opposed to standard computing has several issues which can cause reluctance or fear in the user base. Some of these issues include concerns about privacy and data ownership and security.

Solution to Security issues in Mobile Cloud computing

Individuals and enterprises take advantage of the benefits for storing large amount of data or applications on a cloud. However, issues in terms of their integrity, authentication, and digital rights must be taken care of

1) Integrity: Every mobile cloud user must ensure the integrity of their information stored on the cloud. Every access they make must me authenticated and verified. Different approaches in preserving integrity for one's information that is stored on the cloud is being proposed.

2) Authentication: Different authentication mechanisms have been presented and proposed using cloud computing to secure the data access suitable for mobile environments. Some uses the open standards and even supports the integration of various authentication methods. For example, the use of access or log-in IDs, passwords or PINS, authentication requests, etc.

2) Authentication: Different authentication mechanisms have been presented and proposed using cloud computing to secure the data access suitable for mobile environments. Some uses the open standards and even supports the integration of various authentication methods. For example, the use of access or log-in IDs, passwords or PINS, authentication requests, etc.

3) Digital rights management: Illegal distribution and piracy of digital contents such as video, image, audio and e-book, programs becomes more and more popular. Some solutions to protect these contents from illegal access are implemented such as provision of encryption and decryption keys to access these contents. A coding or decoding

Learning Resources:		
1	Reference Books	1. Roger S. Pressman, Software Engineering , McGraw Hill(1997). 2. Database System Concepts by Korth, Silberschatz, Sudarshan - McGraw Hill 3. Oracle 8i – The Complete Reference, by Kevin Loney, Geroge Koch - Tata McGraw Hill 4. Jiawei Micheline Kamber, “Data Mining Concepts and Techniques”,Morgan Kauf Mann Publishers. 5. William Stallings, "Network Security Essentials", Prentice-Hall. 6. Artificial Intelligence by Elaine Rich, Kevin Knight, TMH, 2nd Edition.
2	Websites	• https://www.geeksforgeeks.org/cryptography-introduction • https://www.tutorialspoint.com/data_mining/dm_quick_guide.html • https://www.geeksforgeeks.org/software-engineering