JEPPIAAR
ENGINEERING COLLEGE

DEPARTMENT OF MANAGEMENT STUDIES

MBA / CF / 2025-27/ MB25102

COURSE FILE – THEORY

SUBJECT	:	INFORMATION MANAGEMENT
SUBJECT CODE	:	MB25CO4
BRANCH	:	MBA
SEMESTER	:	I
YEAR	:	I

DETAILED SYLLABUS

MB25102	Information Management	L	T	P	C
		4	0	0	4

Course Objectives:
This course equips the foundational and advanced knowledge of how information systems contribute to strategic Business decisions and operations. It introduces the key concepts of data, system design, databases, and information security and integrates recent IT advancements such as AI, IoT, blockchain, and quantum computing.

Fundamentals of Information Systems in Business: Data, Information, Information System, evolution, types based on functions and hierarchy, Enterprise and functional information systems.

Systems Analysis and Design Techniques: The work of a system analyst- SDLC-System design, AGILE Model, Waterfall Model, Spiral Model, Iterative and Incremental Model - RAD Model - Requirement analysis – Data flow diagram, relationship diagram, UML diagram, design-Implementation-Evaluation and maintenance of MIS, Database System: Overview of Database-Components-advantages and disadvantages of database; Data Warehousing and Data Mining; Business Intelligence; Artificial Intelligence; Expert System; Big Data; Cyber Safety and Security-Cryptography; RSA Model of Encryption; Data Science – Block Chain Technology; E-commerce and E-Business models; IOT – RFID.

Database Management Systems and Warehousing: DBMS, types and evolution, RDBMS, OODBMS, RODBMS, Data warehousing, Data Mart, Data mining - Association rule mining - Clustering - Pattern matching.

Integrated Systems and Information Security: Knowledge based decision support systems, Integrating social media and mobile technologies in Information system, Security, IS Vulnerability, Disaster Management, Computer Crimes, Securing the Web.

Emerging Information Technologies: Machine learning - Deep learning, Big data, Pervasive Computing, Cloud computing, Advancements in AI, IoT, Block chain, Crypto currency, Quantum computing

Managerial Applications and Strategic Use: Strategic role of IT in Business, Role of CIO and IT governance, Aligning IT with Business objectives, Legal and ethical issues in information management, Technology adoption models, IT project management, Role of information systems in Business analytics and decision science.

Weightage: Continuous Assessment: 40%, End Semester Examinations: 60%.

Assessment Methodology: Written Test I & II (60%) Assignment, Presentation, Case Study, Quiz, Simulation, Online Certification, Seminar, Mini project (40%)

References:

1. Laudon, K. C., & Laudon, J. P. (2022). Management information systems (17th ed.). Pearson.
2. Schultheis, R., & Sumner, M. Management information systems – The manager's view. Tata McGraw Hill.
3. (Note: No edition/year available; if you find one, I can update it.)
4. Panneerselvam, R. (2018). Database management systems (3rd ed.). PHI Learning.
5. Laudon, K. C., Turban, E., & Traver, C. G. (2023). E-commerce: Business, technology, society (17th ed.). Pearson.
6. Loshin, D. (2021). Big data analytics (2nd ed.). Elsevier.
7. Han, J., Kamber, M., & Pei, J. (2012). Data mining: Concepts and techniques (3rd ed.).

E-Resources:

1. NPTEL – Management Information System https://onlinecourses.nptel.ac.in
2. World Economic Forum – Reports on Emerging Tech https://www.weforum.org/

	CO description	PO Mapping	PSO1	PSO2
CO1	Demonstrate conceptual knowledge of information systems fundamentals, system analysis and design techniques, database management systems, information security, emerging technologies, and their strategic applications in business.	PO5(3)	-	2
CO2	Interpret and relate the evolution and types of information systems, system development methodologies, database concepts, the integration of integrated systems, the impact of emerging technologies, and the strategic role of IT governance to inform managerial decisions.	PO1(1) PO5(3)	1	2
CO3	Apply system analysis and design techniques, database management principles, security practices, and emerging technologies to solve business problems and support strategic decision-making across enterprise and functional information systems.	PO1(3) PO4(3)	2	3
CO4	Analyze the functional and hierarchical aspects of information systems, evaluate system design methodologies, examine the role of database management and data warehousing, and assess the strategic implications of integrated systems, emerging technologies, and IT governance for business.	PO1(3) PO3(1) PO4(2)	3	3
CO5	Evaluate the effectiveness of different information systems, system design models, database management systems, security measures, and emerging technologies (like AI, IoT, blockchain) to assess their impact on business transformation and ethical and legal compliance.	PO1(2) PO3(3) PO4(2)	3	3
CO6	Develop strategic information management plans and business solutions by synthesizing knowledge of information systems fundamentals, system design, database management, security, and emerging technologies to achieve business objectives and enhance organizational performance.	PO1(3) PO3(2) PO4(3)	3	3

CO PO MATRIX

SUBJECT : INFORMATION MANAGEMENT
SUBJECT CODE : MB25102

COURSE OBJECTIVES:

This course equips the foundational and advanced knowledge of how information systems contribute to strategic Business decisions and operations. It introduces the key concepts of data, system design, databases, and information security and integrates recent IT advancements such as AI, IoT, blockchain, and quantum computing.

	CO description	PO Mapping	PSO1	PSO2
CO1	Demonstrate conceptual knowledge of information systems fundamentals, system analysis and design techniques, database management systems, information security, emerging technologies, and their strategic applications in business.	PO5(3)	-	2
CO2	Interpret and relate the evolution and types of information systems, system development methodologies, database concepts, the integration of integrated systems, the impact of emerging technologies, and the strategic role of IT governance to inform managerial decisions.	PO1(1) PO5(3)	1	2
CO3	Apply system analysis and design techniques, database management principles, security practices, and emerging technologies to solve business problems and support strategic decision-making across enterprise and functional information systems.	PO1(3) PO4(2)	2	3
CO4	Analyze the functional and hierarchical aspects of information systems, evaluate system design methodologies, examine the role of database management and data warehousing, and assess the strategic implications of integrated systems, emerging technologies, and IT governance for business.	PO1(3) PO3(1) PO4(2)	3	3
CO5	Evaluate the effectiveness of different information systems, system design models, database management systems, security measures, and emerging technologies (like AI, IoT, blockchain) to assess their impact on business transformation and ethical and legal compliance.	PO1(3)	3	3
CO6	Develop strategic information management plans and business solutions by synthesizing knowledge of information systems fundamentals, system design, database management, security, and emerging technologies to achieve business objectives and enhance organizational performance.	PO1(3) PO3(1) PO4(2)	3	3

JEPPIAAR
ENGINEERING COLLEGE

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 119.
(ISO 9001:2008 & NBA Accredited Institution)

DEPARTMENT OF MANAGEMENT STUDIES

Name of the Subject	: INFORMATION MANAGEMENT
Semester, Year & Branch	: I, I & MBA
Subject Code	: MB25102
Name of the Staff Handling the Subject	: Dr.M.S.Nelson Nirmal Kumar
Students Batch	: 2025-2027

Signature of HOD

Date : 18-12-25

INFORMATION MANAGEMENT

UNIT 1

INTRODUCTION TO INFORMATION MANAGEMENT

UNIT I
INTRODUCTION TO INFORMATION MANAGEMENT

SYLLABUS

Data, Information, Information System, evolution, types based on functions and hierarchy, Enterprise and functional information systems.

****

INTRODUCTION:

Data

Data is defined as facts or figures, or information that's stored in or used by a computer. An example of data is information collected for a research paper. the quantities, characters, or symbols on which operations are performed by a computer, which may be stored and transmitted in the form of electrical signals and recorded on magnetic, optical, or mechanical recording media.

Information

Information is a stimulus that has meaning in some context for its receiver. When information is entered into and stored in a computer, it is generally referred to as data. After processing (such as formatting and printing), output data can again be perceived as information.

Information (shortened as info or info.) is that which informs, i.e. that from which data can be derived. At its most fundamental, information is any propagation of cause and effect within a system. Information is conveyed either as the content of a message or through direct or indirect observation of something. That which is perceived can be construed as a message in its own right, and in that sense, information is always conveyed as the content of a message. Information can be encoded into various forms for transmission and interpretation. For example, information may be encoded into signs, and transmitted via signals.

These are difficult times for all organizations of all sizes and in all sectors. On the one hand, customers have ever-increasing expectations in terms of the speed and quality of service they expect and, on the other resources are continually under pressure.

This document sets out how effective information and records management can help any organization to move forward in this challenging environment through,

- achieving cost and efficiency savings;
- making best use of information assets and
- Taking advantage of the opportunities offered by new technologies.

Intelligence

Intelligence has been defined in many different ways such as in terms of one's capacity for logic, abstract thought, understanding, self-awareness, communication, learning, emotional knowledge, memory, planning, creativity and problem solving.

Knowledge

Knowledge is a familiarity, awareness or understanding of someone or something, such as facts, information, descriptions, or skills, which is acquired through experience or education by perceiving, discovering, or learning. Knowledge can refer to a theoretical or practical understanding of a subject.

Importance:

- Learning Better
- Setting Goals As You Learn
- Learn Complex Things Faster
- Knowledge Helps You Solve Problems
- Understanding Yourself

Information Technology (IT)

Information technology (IT) is the application of computers and telecommunications equipment to store, retrieve, transmit and manipulate data, often in the context of a business or other enterprise.

Need

- Education is a lifelong process therefore anytime anywhere access to it is the need
- Information explosion is an ever increasing phenomena therefore there is need to get access to this information
- Education should meet the needs of variety of learners and therefore IT is important in meeting this need

- It is a requirement of the society that the individuals should posses technological literacy
- We need to increase access and bring down the cost of education to meet the challenges of illiteracy and poverty-IT is the answer

Importance

- access to variety of learning resources
- immediacy to information
- anytime learning
- anywhere learning
- collaborative learning
- multimedia approach to education
- authentic and up to date information
- access to online libraries
- teaching of different subjects made interesting
- educational data storage
- distance education
- access to the source of information
- Multiple communication channels-e-mail, chat, forum, blogs, etc.
- access to open courseware
- better accesses to children with disabilities
- reduces time on many routine tasks

Information system

An information system (IS) is a system composed of people and computers that processes or interprets information. The term is also sometimes used in more restricted senses to refer to only the software used to run a computerized database or to refer to only a computer system.

Importance

1. To control the creation and growth of records

Despite decades of using various non-paper storage media, the amount of paper in our

offices continues to escalate. An effective records information system addresses both creation control (limits the generation of records or copies not required to operate the

business) and records retention (a system for destroying useless records or retiring inactive records), thus stabilizing the growth of records in all formats.

2. To reduce operating costs

Recordkeeping requires administrative dollars for filing equipment, space in offices, and staffing to maintain an organized filing system (or to search for lost records when there is no organized system).It costs considerably less per linear foot of records to store inactive records in a Data Records Center versus in the office and there is an opportunity to effect some cost savings in space and equipment, and an opportunity to utilize staff more productively - just by implementing a records management program.

3. To improve efficiency and productivity

Time spent searching for missing or misfiled records are non-productive. A good records management program (e.g. a document system) can help any organization upgrade its recordkeeping systems so that information retrieval is enhanced, with corresponding improvements in office efficiency and productivity. A well designed and operated filing system with an effective index can facilitate retrieval and deliver information to users as quickly as they need it.

4. To assimilate new records management technologies

A good records management program provides an organization with the capability to assimilate new technologies and take advantage of their many benefits. Investments in new computer systems whether this is financial, business or otherwise, don't solve filing problems unless current manual recordkeeping or bookkeeping systems are analyzed (and occasionally, overhauled) before automation is applied.

5. To ensure regulatory compliance

In terms of recordkeeping requirements, China is a heavily regulated country. These laws can create major compliance problems for businesses and government agencies since they can be difficult to locate, interpret and apply. The only way an organization can be reasonably sure that it is in full compliance with laws and regulations is by operating a good management information system which takes responsibility for regulatory compliance, while working closely with the local authorities. Failure to comply with laws and regulations could result in severe fines, penalties or other legal consequences.

6. To minimize litigation risks

Business organizations implement management information systems and programs in order to reduce the risks associated with litigation and potential penalties. This can be equally true in Government agencies. For example, a consistently applied records management program can reduce the liabilities associated with document disposal by providing for their systematic, routine disposal in the normal course of business.

7. To safeguard vital information

Every organization, public or private, needs a comprehensive program for protecting its vital records and information from catastrophe or disaster, because every organization is vulnerable to loss. Operated as part of a good management information system, vital records programs preserve the integrity and confidentiality of the most important records and safeguard the vital information assets according to a "Plan" to protect the records. This is especially the case for financial information whereby ERP (Enterprise Resource Planning) systems are being deployed in large companies.

8. To support better management decision making

In today's business environment, the manager that has the relevant data first often wins, either by making the decision ahead of the competition, or by making a better, more informed decision. A good management information system can help ensure that managers and executives have the information they need when they need it.

9. To preserve the corporate memory

An organization's files, records and financial data contain its institutional memory, an irreplaceable asset that is often overlooked. Every business day, you create the records, which could become background data for future management decisions and planning.

10. To foster professionalism in running the business

A business office with files, documents and financial data askew, stacked on top of file cabinets and in boxes everywhere, creates a poor working environment. The perceptions of customers and the public, and "image" and "morale" of the staff, though hard to quantify in cost-benefit terms, may be among the best reasons to establish a good management information system.

Evolution

The first business application of computers (in the mid- 1950s) performed repetitive, high-volume, transaction-computing tasks. The computers‖ crunched numbers‖ summarizing and organizing transactions and data in the accounting, finance, and human resources areas. Such systems are generally called transaction processing systems (TPSs).

Management Information Systems (MISs): these systems access, organize, summarize and display information for supporting routine decision making in the functional areas.Office Automation Systems (OASs): such as word processing systems were developed to support office and clerical workers.

Decision Support Systems: were developed to provide computer based support for complex, non routine decision. „ End- user computing: The use or development of information systems by the principal users of the systems‘ outputs, such as analysts, managers, and other professionals.

Intelligent Support System (ISSs): Include expert systems which provide the stored knowledge of experts to non experts, and a new type of intelligent system with machine-learning capabilities that can learn from historical cases. „ Knowledge Management Systems: Support the creating, gathering, organizing, integrating and disseminating of organizational knowledge.

Data Warehousing: A data warehouse is a database designed to support DSS, ESS and other analytical and end-user activities. „ Mobile Computing: Information systems that support employees who are working with customers or business partners outside the physical boundaries of their company; can be done over wire or wireless networks.

Kinds of Information Systems

- Organizational Hierarchy
- Organizational Levels
- Information

Systems Four General

Kinds of IS

- Operational-level systems
  - Support operational managers by monitoring the day-to-day‘s elementary

activities and
transactions of the organization. e.g. TPS.

- Knowledge-level systems
  - Support knowledge and data workers in designing products, distributing information, and coping with paperwork in an organization. e.g. KWS, OAS
- Management-level systems
  - Support the monitoring, controlling, decision-making, and administrative activities of middle managers. e.g. MIS, DSS
- Strategic-level systems
  - Support long-range planning activities of senior management. e.g. ESS
- Executive Support Systems (ESS)
- Management Information Systems (MIS)
- Decision Support Systems (DSS)
- Knowledge Work Systems (KWS)
- Office Automation Systems (OAS)
- Transaction Processing Systems (TPS)

Transaction Processing Systems (TPS)

Computerized system that performs and records the daily routine transactions necessary to conduct the business; these systems serve the operational level of the organization

- TYPE: Operational-level
- INPUTS: transactions, events
- PROCESSING: updating
- OUTPUTS: detailed reports
- USERS: operations personnel, supervisors
- DECISION-MAKING: highly

structured

- EXAMPLE: payroll, accounts

payable.

Office Automation Systems (OAS)

Computer system, such as word processing, electronic mail system, and scheduling system, that is designed to increase the productivity of data workers in the office.

- TYPE: Knowledge-level
- INPUTS: documents, schedule
- PROCESSING: document management, scheduling, communication
- OUTPUTS: documents; schedules
- USERS: clerical workers
- EXAMPLE: document

imaging system

Knowledge Work Systems (KWS)

Information system that aids knowledge workers in the creation and integration of new knowledge in the organization.

- TYPE: Knowledge-level
- INPUTS: design specifications
- PROCESSING: modelling
- OUTPUTS: designs, graphics
- USERS: technical staff;

professionals EXAMPLE:
Engineering workstations

Decision Support Systems (DSS)

Information system at the management level of an organization that combines data and sophisticated analytical models or data analysis tools to support semi-structured and unstructured decision making.

- TYPE: Management-level
- INPUTS: low volume data
- PROCESSING: simulations, analysis
- OUTPUTS: decision analysis
- USERS: professionals, staff managers
- DECISION-MAKING: semi-

structured EXAMPLE: sales region
analysis

Management Information Systems (MIS)

Information system at the management level of an organization that serves the functions of planning, controlling, and decision making by providing routine summary and exception reports.

TYPE: Management-level

- INPUTS: high volume data
- PROCESSING: simple models
- OUTPUTS: summary reports
- USERS: middle managers
- DECISION-MAKING: structured to semi-structured

EXAMPLE: annual budgeting

Executive Support Systems (ESS)

Information system at the strategic level of an organization that address unstructured decision making through advanced graphics and communications.

TYPE: Strategic level

- INPUTS: aggregate data; internal and external
- PROCESSING: interactive
- OUTPUTS: projections
- USERS: senior managers
- DECISION-MAKING: highly

unstructured EXAMPLE: 5 year operating
plan

Classification of IS by Organizational Structure

- Departmental Information Systems
- Enterprise Information System
- Inter-organizational Systems
  - NYCE
  - SABRE or APOLLO

Classification of IS by Functional Area

- The accounting information system
- The finance information system
- The manufacturing (operations, production) information system
- The marketing information system
- The human resources information system.

Decision Support System (DSS)

A Decision Support System (DSS) is a computer-based information system that supports business or organizational decision-making activities.

DSSs serve the management, operations, and planning levels of an organization (usually mid and higher management) and help to make decisions, which may be rapidly changing and not easily specified in advance (Unstructured and Semi-Structured decision problems). Decision support systems can be either fully computerized, human or a combination of both.

Decision support systems generally involve non-programmed decisions. Therefore; there will be no exact report, content or format for these systems. Reports are generated on the fly.

Attributes of a DSS

- Adaptability and flexibility
- High level of Interactivity
- Ease of use
- Efficiency and effectiveness
- Complete control by decision-makers.
- Ease of development
- Extendibility
- Support for modeling and analysis
- Support for data access
- Standalone, integrated

Characteristics of a DSS

- Support for decision makers in semi structured and unstructured problems.
- Support for managers at various managerial levels, ranging from top executive to line managers.
- Support for individuals and groups. Less structured problems often requires the involvement of several individuals from different departments and organization level.
- Support for interdependent or sequential decisions.
- Support for intelligence, design, choice, and implementation.

Benefits of DSS

- Improves efficiency and speed of decision making activities
- Increases the control, competitiveness and capability of futuristic decision making of the organization
- Facilitates interpersonal communication
- Encourages learning or training
- Since it is mostly used in non-programmed decisions, it reveals new approaches and sets up new evidences for an unusual decision

Components of a DSS

Following are the components of the Decision Support System:

- Database Management System (DBMS): To solve a problem the necessary data may come from internal or external database. In an organization, internal data are generated by a system such as TPS and MIS.External data come from a variety of sources such as newspapers, online data services, databases (financial, marketing, human resources).

- Model Management system: It stores and accesses models that managers use to make decisions. Such models are used for designing manufacturing facility, analyzing the financial health of an organization. Forecasting demand of a product or service etc.

Support Tools: Support tools like online help; pull down menus, user interfaces, graphical analysis, error correction mechanism, facilitates the user interactions with the system.

Classification of DSS

There are several ways to classify DSS. Hoi Apple and Whinstone classify DSS in following:

- Text Oriented DSS: It contains textually represented information that could have a bearing on decision. It allows documents to be electronically created, revise and viewed as needed

- Database Oriented DSS: Database plays a major role here; it contains organized and highly structured data.
- Spreadsheet Oriented DSS: it contains information in spread sheets that allows create, view, modify procedural knowledge and also instruct the system to execute self-contained instructions. The most popular tool is Excel and Lotus 1-2-3.
- Solver Oriented DSS: it is based on a solver, which is an algorithm or procedure written for performing certain calculations and particular program type.
- Rules Oriented DSS: It follows certain procedures adopted as rules.
- Rules Oriented DSS: Procedures are adopted in rules oriented DSS. Export system is the example.
- Compound DSS: It is built by using two or more of the five structures

explained above Types of DSS

- Status Inquiry System: helps in taking operational management level or middle level management decisions, for example daily schedules of jobs to machines or machines to operators.
- Data Analysis System: needs comparative analysis and makes use of formula or an algorithm, for example cash flow analysis, inventory analysis etc.
- Information Analysis System: In this system data is analyzed and the information report is generated. For example, sales analysis, accounts receivable systems, market analysis etc.
- Accounting System: keep tracks of accounting and finance related information, for example, final account, accounts receivables, accounts payables etc. that keep track of the major aspects of the business.
- Model Based System: simulation models or optimization models used for decision- making used infrequently and creates general guidelines for operation or management

Executive support systems

Executive support systems are intended to be used by the senior managers directly to provide support to non-programmed decisions in strategic management.

These information are often external, unstructured and even uncertain. Exact scope and context of such information is often not known beforehand.

This information is intelligence based:

- Market intelligence
- Investment intelligence
- Technology intelligence

Following are some examples of intelligent information, which is often source of an ESS:

- External databases
- Technology reports like patent records etc.
- Technical reports from consultants
- Market reports
- Confidential information about competitors
- Speculative information like market conditions
- Government policies

Advantages of ESS

- Easy for upper level executive to use
- Ability to analyze trends
- Augmentation of managers' leadership capabilities
- Enhance personal thinking and decision making
- Contribution to strategic control flexibility
- Enhance organizational competitiveness in the market place
- Instruments of change
- Increased executive time horizons.
- Better reporting system
- Improved mental model of business executive
- Help improve consensus building and communication

- Improve office automation
- Reduce time for finding information
- Detail examination of critical success factor
- Better understanding
- Time management

Disadvantage of ESS

- Functions are limited
- Hard to quantify benefits
- Executive may encounter information overload
- System may become slow
- Difficult to keep current data
- May lead to less reliable and insecure data
- Excessive cost for small company

Knowledge Management System (KMS)

All the systems we are discussing here come under knowledge management category. A knowledge management system is not radically different from all these information systems, but it just extends the already existing systems by assimilating more information.

As we have seen data is raw facts, information is processed and/or interpreted data and knowledge is personalized information.

What is knowledge?

- personalized information
- state of knowing and understanding
- an object to be stored and manipulated
- a process of applying expertise
- a condition of access to information
- potential to influence action

Sources of knowledge of an Organization

- Intranet
- Data warehouses and knowledge repositories
- Decision support tools
- Groupware for supporting collaboration
- Networks of knowledge workers

Purpose of a KMS

- Improved performance
- Competitive advantage
- Innovation
- Sharing of knowledge
- Integration
- Continuous improvement by:
  - Driving strategy
  - Starting new lines of business
  - Solving problems faster
  - Developing professional skills
  - Recruit and retain Management

UNIT 2
SYSTEM ANALYSIS AND DESIGN

SYLLABUS

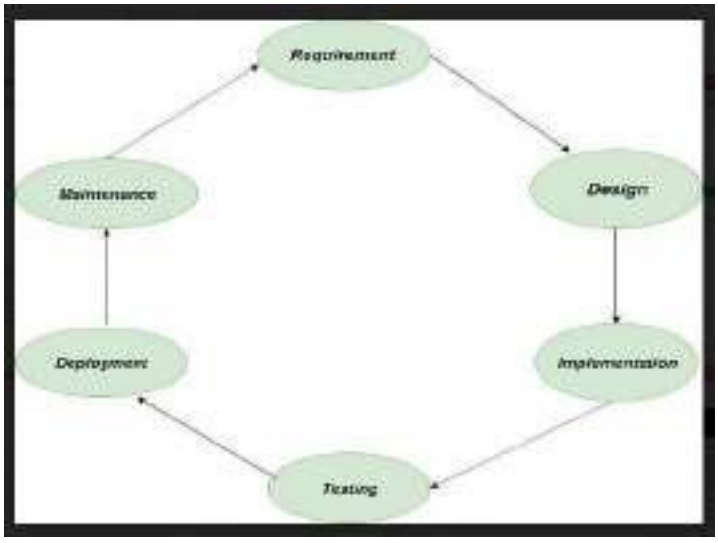
System development methodologies, Systems Analysis and Design, Data flow Diagram (DFD), Decision table, Entity Relationship (ER), Object Oriented Analysis and Design(OOAD), UML diagram.

****

The Software Development Life Cycle (SDLC)

SDLC is a process used by software development organizations to plan, design, develop, test, deploy, and maintain software applications. The SDLC is a valuable tool for organizations to use when developing software applications, as it helps to ensure that the final product is of high quality, meets the requirements**,** and is delivered on time and within budget.

The SDLC typically includes the following phases:



1. Requirements gathering and analysis: This phase involves gathering information about the software requirements from stakeholders, such as customers, end-users, and business analysts.
2. Design: In this phase, the software design is created, which includes the overall architecture of the software, data structures, and interfaces. It has two steps:

- **High-level design (HLD):** It gives the architecture of software products.
- **Low-level design (LLD):** It describes how each and every feature in the product should work and every component.

3. Implementation or coding: The design is then implemented in code, usually in several iterations, and this phase is also called as Development.
things you need to know about this phase:

- This is the longest phase in SDLC model.
- This phase consists of Front end + Middleware + Back-end.
- **In front-end:** Development of coding is done even SEO settings are done.

- **In Middleware:** They connect both the front end and back end.
- **In the back-end:** A database is created.

4. **Testing:** Before getting the software product out the door to the production environment, it's important to have your quality assurance team perform validation testing to make sure it is functioning properly and does what it's meant to do. The testing process can also help hash out any major user experience issues and security issues. The software is thoroughly tested to ensure that it meets the requirements and works correctly.

5. Deployment: After successful testing, the software is deployed to a production environment and made available to end-users.

6. Maintenance: The maintenance phase is the final stage of the SDLC. This phase includes ongoing support, bug fixes, and updates to the software.

Software development life cycle (SDLC) methodologies:

Software Development Methodologies are approaches or strategies used to organize, plan, and execute software development projects. They provide a framework for managing the development process and ensuring the project is completed on time, within budget, and with the required quality. Here are some of the most common **software development methodologies**:

1. Waterfall Model

The Waterfall model is a linear and sequential approach where each phase of the software development life cycle (SDLC) is completed before the next phase begins. It follows a strict, step-by-step process.

- Phases:
    1. Requirements gathering
    2. System design
    3. Implementation (coding)
    4. Testing
    5. Deployment
    6. Maintenance
- **Pros**: Simple and easy to understand; good for small projects with clear, stable requirements.
- **Cons**: Inflexible, as changes can be difficult and costly after the requirements phase is complete; poor for large, complex projects or ones with evolving requirements.

2. Agile Methodology

Agile is an iterative and incremental approach that promotes flexibility, collaboration, and customer feedback. Development is carried out in small, iterative cycles called **sprints**, typically lasting 1-4 weeks, and each sprint produces a usable product increment.

- Key Features:
    - Continuous delivery of working software
    - Close collaboration with stakeholders

- Ability to adapt to changing requirements

- Popular Agile Frameworks:
  - **Scrum**: Focuses on time-boxed iterations (sprints) and roles like Product Owner, Scrum Master, and Scrum Team.
  - **Kanban**: Focuses on continuous flow and improvement by visualizing the workflow and limiting work in progress (WIP).
  - **Extreme Programming (XP)**: Emphasizes technical practices like pair programming, test-driven development (TDD), and frequent releases.
- **Pros**: Highly flexible and adaptive to change; encourages collaboration and customer involvement.
- **Cons**: Can be hard to manage for large teams or complex projects; may lead to scope creep if not managed properly.

4. DevOps

DevOps is a methodology that combines software development (Dev) and IT operations (Ops) to enhance collaboration and improve the speed of software delivery. It emphasizes continuous integration, continuous delivery (CI/CD), and automated testing.

- Key Features:
  - Automation of manual tasks like testing, deployment, and infrastructure management.
  - Continuous feedback loop between development and operations teams.
  - Use of version control systems, automated build tools, and deployment pipelines.
- **Pros**: Faster development cycles, improved collaboration between teams, better product quality through continuous testing and delivery.
- **Cons**: Requires a shift in culture and organization, and can be complex to implement.

5. Spiral Model

The Spiral model is a risk-driven approach that combines elements of both the Waterfall and Prototyping models. It focuses on iterative development and allows for frequent reassessment of the project's risks.

- Phases:
  1. Planning
  2. Risk Analysis
  3. Engineering (Design and Development)
  4. Evaluation
- Key Features:
  - Emphasizes risk analysis and management.
  - Allows for iterative development and continuous refinement.
  - Each cycle involves planning, risk analysis, engineering, and evaluation.
- **Pros**: Highly adaptable and risk-focused; better suited for large and complex projects.
- **Cons**: Can be resource-intensive and may result in project delays if risk management is not effective.

7. Prototyping Model

The Prototyping model involves creating an early, simplified version of the software (prototype) to demonstrate its functionality to stakeholders. The prototype is refined based on feedback until the final product is developed.

- Key Features:
  - Develop an initial prototype, gather feedback, and refine the system incrementally.
  - Allows early visualization of the system for better user involvement.
- **Pros**: Allows for early user feedback and helps clarify requirements.
- **Cons**: The final product may end up being overly influenced by the prototype, leading to suboptimal designs.

8. RAD (Rapid Application Development)

RAD is an Agile-based methodology that focuses on rapid prototyping, user feedback, and iterative development. It typically involves a small development team and quick releases of working prototypes.

- Key Features:
  - Use of prototypes to speed up the development process.
  - Frequent user feedback and adjustments.
  - Focus on reducing development time through the use of pre-built components and tools.
- **Pros**: Fast delivery of software, high flexibility, and customer involvement.
- **Cons**: Can be difficult to manage with larger teams or complex projects; may compromise quality for speed.

9. Incremental Model

The Incremental model breaks the system into smaller, manageable parts (increments) that are developed and delivered in stages. Each increment adds more functionality to the existing system.

- Phases:
  1. Requirements gathering
  2. Design
  3. Implementation (incrementally)
  4. Testing
  5. Deployment
- Key Features:
  - Software is developed and delivered in parts.
  - Each part is built on top of the previous one, making it easier to manage.

10. V-Model (Verification and Validation)

The V-Model is an extension of the Waterfall model that emphasizes verification and

validation at each development phase. It mirrors the stages of the Waterfall model with corresponding testing activities.

Phases**:**

1. Requirements
2. System Design
3. Detailed Design
4. Coding
5. Unit Testing
6. Integration Testing
7. System Testing
8. Acceptance Testing

- Key Features:
  - Each development phase has a corresponding testing phase.
  - Emphasizes the importance of early detection of defects.
- **Pros**: Easy to manage due to its structured nature; effective for projects with clear, well-defined requirements.
- **Cons**: Inflexible, and changes in requirements can lead to significant challenges.

System Analysis and Design (SAD)

System Analysis and Design (SAD) is a structured process used to define, model, and implement a software or information system. It involves studying and understanding the current system, identifying problems, proposing solutions, and designing a new system or improving the existing one. SAD is typically done in the initial phases of the Software Development Life Cycle (SDLC) and provides a roadmap for building efficient and functional systems.

System Analysis

System analysis is the process of gathering, analyzing, and documenting the business needs and system requirements. The goal is to understand the current system (if any) and the requirements of the new system.

Key Steps in System Analysis:

- **Feasibility Study**:
  - **Technical Feasibility**: Can the technology support the new system?
  - **Economic Feasibility**: Is the project financially viable?
  - **Operational Feasibility**: Can the organization operate the system effectively?
- Requirement Gathering:
  - **Interviews**: Discussing with stakeholders to gather their needs.
  - **Surveys/Questionnaires**: Collecting data from users or customers.
  - **Document Review**: Examining existing system documentation, if applicable.

    - **Observation**: Directly observing how the current system works.
- Modeling the System:
    - **Data Flow Diagrams (DFD)**: Representing the flow of data within the system.
    - **Entity-Relationship Diagrams (ERD)**: Showing the relationships between data entities.
    - **Use Case Diagrams**: Describing how users will interact with the system.
- **Identifying Problems**: Analyzing the current system to identify inefficiencies, bottlenecks, or areas of improvement.
- **Defining System Requirements**: Documenting the system's functional and non-functional requirements.

System Design

System design is the phase where solutions to the problems identified in the analysis phase are devised. It involves creating detailed blueprints for how the system will function and how the components will work together.

Key Steps in System Design:

- High-Level Design (Architectural Design):
    - Defining the architecture of the system, such as client-server architecture, cloud-based solutions, or microservices.
    - Identifying major system components and how they interact with each other.
    - Choosing the appropriate technology stack (programming languages, databases, frameworks).
- Detailed Design:
    - **Database Design**: Defining the database schema, tables, relationships, and constraints.
    - **Interface Design**: Designing user interfaces and how users will interact with the system.
    - **System Modules**: Defining each module's functionality, inputs, and outputs.
    - **Data Flow**: Detailing how data moves between different components or users.
- Design Diagrams:
    - **Entity-Relationship Diagrams (ERD)**: Detailing the database structure.
    - **Data Flow Diagrams (DFD)**: Detailing the flow of data between processes and data stores.
    - **Use Case Diagrams**: Showing interactions between users and the system.
    - **Class Diagrams**: Describing the object-oriented design for object classes, relationships, and attributes.
    - **Sequence Diagrams**: Illustrating how objects or components interact over time.
- **User Interface Design**: Creating wireframes or prototypes for the user interface (UI), including the layout, navigation, and overall user experience.
- **Security Design**: Planning security features, such as encryption, user authentication, and data protection.
- **Performance Design**: Ensuring the system will meet performance requirements, such as speed, scalability, and reliability.

Tools Used in System Analysis and Design

1. **Unified Modeling Language (UML)**: A set of standardized diagrams (such as Use Case, Class, Sequence, and Activity diagrams) used for modeling the system during the design phase.
2. **ERD Tools**: Used to create entity-relationship diagrams that define database structure.
3. **Prototyping Tools**: Used for creating interactive mock-ups or prototypes of the user interface (e.g., Figma, Sketch).
4. **Version Control**: Tools like Git to manage and track changes in the codebase during development.

System Analysis focuses on understanding the business problems, defining the requirements, and analyzing the existing systems.

System Design takes the requirements and specifications from analysis and translates them into a blueprint for building the system.

Together, **System Analysis and Design** ensure that the software is built to meet the organization's needs, is efficient, and can evolve as business requirements change. It provides a structured approach to ensure that software development is done systematically and with minimal risks.

Data Flow Diagram (DFD)

A **Data Flow Diagram (DFD)** is a visual representation of the flow of data within a system, showing how input data is transformed into output through various processes. DFDs are used in **System Analysis and Design (SAD)** to represent the flow of information, which helps in understanding how a system works. It focuses on **data movement** rather than control flow, helping analysts identify system components that handle data and how they interact with each other.

Key Components of a Data Flow Diagram:

1. **Processes**: Represented by circles or ovals, processes show how data is transformed. A process takes inputs (data) and produces outputs (results).
   - Example: "Calculate Total Price" or "Process Payment."
2. **Data Stores**: Represented by open-ended rectangles, data stores show where data is stored in the system (e.g., databases, files, etc.).
   - Example: "Customer Database," "Orders File."
3. **External Entities**: Represented by rectangles, external entities are sources or destinations of data outside the system being modeled (e.g., users, other systems, or devices).
   - Example: "Customer," "Payment Gateway."
4. **Data Flows**: Represented by arrows, data flows show how data moves between processes, data stores, and external entities.
   - Example: "Customer Info," "Order Details."

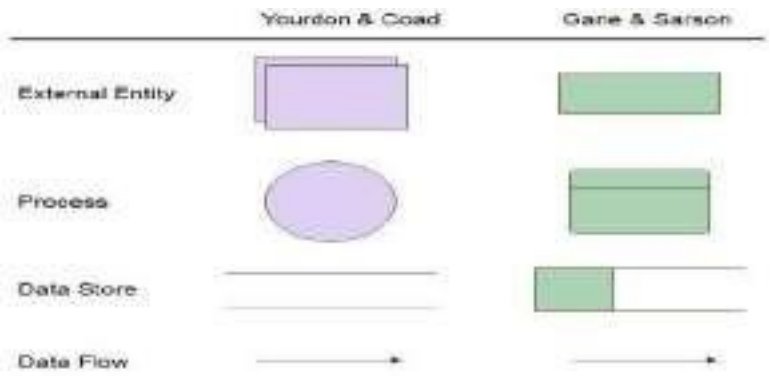
5. **Resource Flow** - A resource flow shows the flow of any physical material from its source to its destination. For this reason, they are sometimes referred to as physical

flows.

Benefits of Data Flow Diagrams:

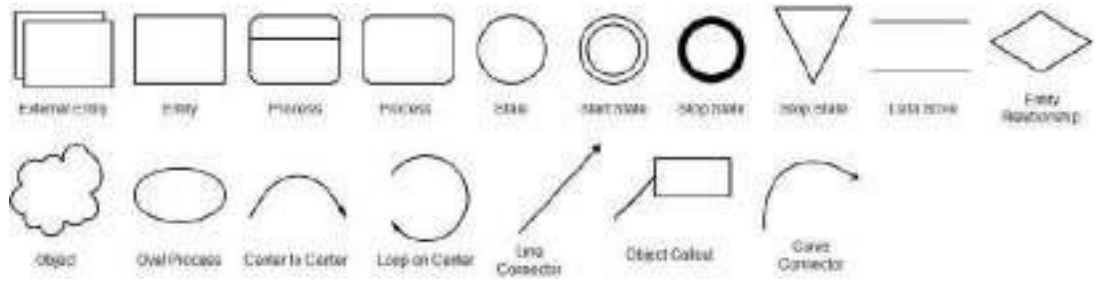
- **Clear Visualization**: DFDs provide an easy-to-understand graphical representation of data flow within a system.
- **Helps Identify Problems**: By mapping out how data moves and is processed, DFDs can help identify inefficiencies, bottlenecks, or potential areas of improvement.
- **System Understanding**: They help stakeholders (developers, business analysts, and users) understand how the system works, even without technical expertise.
- **Documentation**: DFDs serve as valuable documentation for both current and future development.

A data flow diagram has four basic elements. The elements include external entities, data stores, processes, and data flows. The elements are best represented by the two main methods of notation used in DFDs – **Yourdon & Coad**, and **Gane & Sarson**. DFD symbols vary slightly depending on methodology. Even so, the basic ideas remain the same.



1. **External entities** are represented by squares as the source or destination of data.
2. **Processes** are represented by rectangles with rounded corners.
3. **Data Flows** are referred to by arrows to denote the physical or electronic flow of data.
4. **Data Stores** are physical or electronic-like XML files denoted by open-ended rectangles.

Data Flow Diagram



How to Create a Data Flow Diagram?

To create a valid DFD, it's important to follow the 4 rules of thumb. Every single process should have at least one input and one output.

1. Each data store should have at least one data flow in and data flow out.
2. Every system's stored data has to go through a process.
3. Every process in a data flow diagram must link to another process or data store.

With the background information on DFDs and the rules of thumb, you can build your own DFD. The process entails the following five steps:

1. Identify the major inputs and outputs in your system

This step gives a macro view of your system and elucidates the broadest tasks the system should achieve. Again, the rest of the DFD is built on these elements.

2. Build a context diagram (Level 0 DFD)

You could achieve this by drawing a single process node and connecting it to related external entities. The node represents the general process information undergoes in a system from input to output.

3. Expand the context diagram into a level 1 DFD

Level 1 DFDs are more of a general overview, but they give more detail than a context diagram. Break the single process lump into detailed processes. This brings out where the information starts and what needs to happen to it.

4. Expand to level 2+ DFD

This breaks the processes down into more detailed sub-processes. Ensure you add any necessary data stores and flows at this point.

5. Ascertain the accuracy of your final DFD

Walk again through your diagram as you pay close attention to the flow of information. If it makes sense and all necessary data stores are included, then thumbs up. Other parties should find your diagram comprehensible.

Levels in Data Flow Diagrams (DFD)

Software engineering DFD (data flow diagram) can be drawn to represent the system of different levels of abstraction. Higher-level DFDs are partitioned into low levels-hacking more information and functional elements. Levels in DFD are numbered 0, 1, 2 or beyond. Here, we will see mainly 3 levels in the data flow diagram, which are: 0-level DFD, 1-level DFD, and 2- level DFD.

The following are the four levels of DFDs:

1. **Level 0 DFD**
2. **Level 1 DFD**
3. **Level 2 DFD**
4. **Level 3 DFD**

The choice of DFD level depends on the complexity of the system and the level of detail required to understand the system. Higher levels of DFD provide a broad overview of the system, while lower levels provide more detail about the system's processes, data flows, and data stores. A combination of different levels of DFD can provide a complete understanding of the system.

Level 0 Data Flow Diagram (DFD)

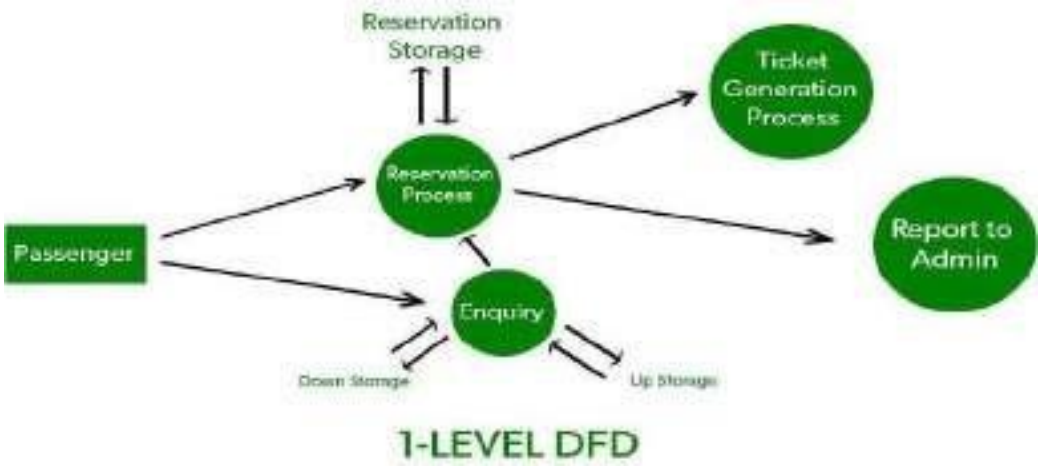
Level 0 is the highest-level Data Flow Diagram (DFD), which provides an overview of the entire system. It shows the major processes, data flows, and data stores in the system, without providing any details about the internal workings of these processes. It is also known as a context diagram. It's designed to be an abstraction view, showing the system as a single process with its relationship to external entities. It represents the entire system as a single bubble with input and output data indicated by incoming/outgoing arrows.

0-LEVEL DFD

1- Level Data Flow Diagram (DFD)

1- Level provides a more detailed view of the system by breaking down the major processes identified in the level 0 Data Flow Diagram (DFD) into sub-processes. Each sub-process is depicted as a separate process on the level 1 Data Flow Diagram (DFD). The data flows and data stores associated with each sub-process are also shown.

In 1-level Data Flow Diagram (DFD), the context diagram is decomposed into multiple bubbles/processes. In this level, we highlight the main functions of the system and breakdown the high-level process of 0-level Data Flow Diagram (DFD) into sub-processes.

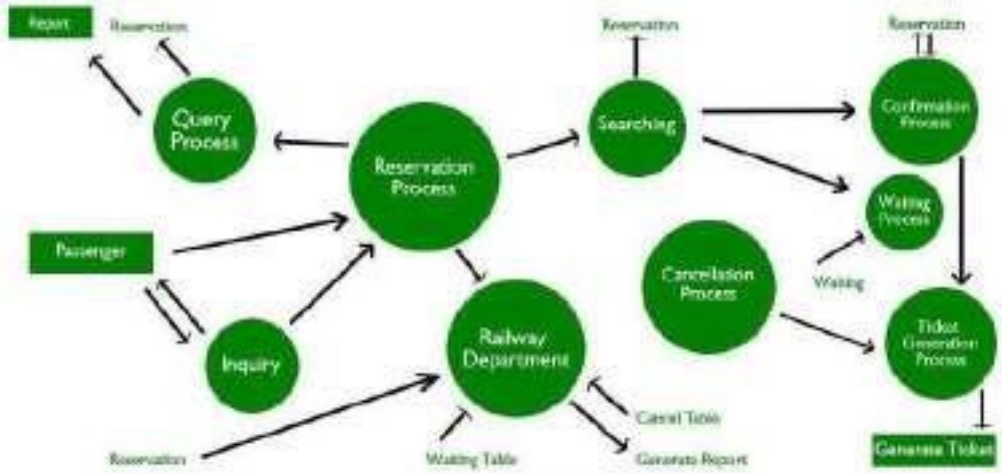


1-LEVEL DFD

2- Level Data Flow Diagram (DFD)

2- Level provides an even more detailed view of the system by breaking down the sub-processes identified in the level 1 Data Flow Diagram (DFD) into further sub-processes. Each sub-process is depicted as a separate process on the level 2 DFD. The data flows and data stores associated with each sub-process are also shown.

2- Level Data Flow Diagram (DFD) goes one step deeper into parts of 1-level DFD. It can be used to plan or record the specific/necessary detail about the system's functioning.



2-LEVEL DFD

3- Level Data Flow Diagram (DFD)

3- Level is the most detailed level of Data Flow Diagram (DFDs), which provides a detailed view of the processes, data flows, and data stores in the system. This level is typically used for complex systems, where a high level of detail is required to understand the system. Each process on the level 3 DFD is depicted with a detailed description of its input, processing, and output. The data flows and data stores associated with each process are also shown.

Advantages of using Data Flow Diagrams (DFD)

Following are the Advantage of Data Flow Diagram (DFD) :

1. **Easy to understand:** DFDs are graphical representations that are easy to understand and communicate, making them useful for non-technical stakeholders and team members.
2. **Improves system analysis:** DFDs are useful for analyzing a system's processes and data flow, which can help identify inefficiencies, redundancies, and other problems that may exist in the system.
3. **Supports system design:** DFDs can be used to design a system's architecture and structure, which can help ensure that the system is designed to meet the requirements of the stakeholders.
4. **Enables testing and verification:** DFDs can be used to identify the inputs and outputs of a system, which can help in the testing and verification of the system's functionality.
5. **Facilitates documentation:** DFDs provide a visual representation of a system, making it easier to document and maintain the system over time.

Disadvantages of using Data Flow Diagram (DFD)

Following are the Disadvantage of Data Flow Diagram (DFD):

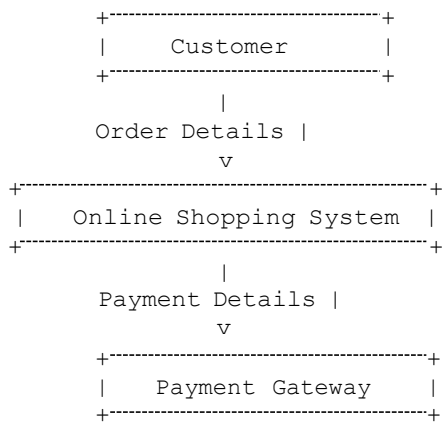
1. **Can be time-consuming:** Creating DFDs can be a time-consuming process, especially for complex systems.
2. **Limited focus:** DFDs focus primarily on the flow of data in a system, and may not capture other important aspects of the system, such as user interface design, system security, or system performance.
3. **Can be difficult to keep up-to-date:** DFDs may become out-of-date over time as the system evolves and changes.
4. **Requires technical expertise:** While DFDs are easy to understand, creating them requires a certain level of technical expertise and familiarity with the system being analyzed.

Data Flow Diagram (DFD) Examples

Example DFD for an Online Shopping Application

Let's illustrate a simple **Level 0 DFD** (Context Diagram) for an online shopping application. This diagram will represent the system as a whole, showing how it interacts with external entities (customers and payment gateways) without breaking it down into smaller processes.

Level 0 DFD (Context Diagram) for an online shopping application.



Explanation:

- **External Entity**:
  - **Customer**: Interacts with the system by placing orders and receiving order confirmations.
  - **Payment Gateway**: Handles payment processing and provides payment confirmation.
- Process:
  - **Online Shopping System**: The central process where orders are placed, payments are processed, and inventory is checked. It manages the entire workflow for an online shopping transaction.
- Data Flow:
  - **Order Details**: Data representing the items and quantities the customer wants to purchase, flowing from the customer to the system.
  - **Payment Details**: Information needed to complete the payment, flowing from the system to the payment gateway.
- External Entity:
  - **Payment Gateway**: An external entity that processes the payment and provides the payment status back to the system.

DECISION TABLE

A **decision table** is a structured way of representing and analyzing different conditions and their corresponding actions. It helps in decision-making processes where multiple conditions can lead to different actions, and it is especially useful when a system has a large number of conditions or rules to manage. Decision tables provide a systematic approach to organizing rules and can be used in various fields like software development, business rules, and automation.

Components of a Decision Table:

1. Conditions:

   Conditions represent the criteria or inputs that affect the decision. Each condition can either be true or false.

2. Actions:

   Actions are the results or outputs that occur based on the conditions. These actions are determined according to the combinations of conditions being true or false.

3. Rules:

   Rules represent the different combinations of conditions that lead to specific actions. Each rule specifies what happens when a certain set of conditions are met.

4. Condition Alternatives:

   Each condition can have two possible alternatives: **True (T)** or **False (F)**. The decision table shows all combinations of these conditions.

Structure of a Decision Table:

A typical decision table consists of:

- **Condition Stubs**: A list of conditions (one per row).
- **Action Stubs**: A list of possible actions (one per column).
- **Condition Alternatives**: Represent the possible values for each condition (True or False).
- **Action Outcomes**: Represent the actions that should be performed based on the combination of condition values.

Example of a Decision Table

Let's consider a **Simple Loan Approval System** where:

- **Condition 1**: Customer's **credit score** (Good or Bad).
- **Condition 2**: Customer's **annual income** (Above $50,000 or Below $50,000).
- **Condition 3**: Customer's **debt-to-income ratio** (High or Low).

Based on these conditions, we want to decide whether to approve or reject the loan.

Conditions:

- Credit score: Good (T) or Bad (F)
- Annual income: Above $50,000 (T) or Below $50,000 (F)
- Debt-to-income ratio: Low (T) or High (F)

Actions:

- Approve the loan
- Reject the loan

Decision Table:

Condition	**Credit Score**	**Annual Income**	**Debt-to-Income Ratio**	**Action**
Rule 1:	Good (T)	Above $50,000 (T)	Low (T)	Approve Loan
Rule 2:	Good (T)	Above $50,000 (T)	High (F)	Approve Loan
Rule 3:	Good (T)	Below $50,000 (F)	Low (T)	Reject Loan
Rule 4:	Good (T)	Below $50,000 (F)	High (F)	Reject Loan
Rule 5:	Bad (F)	Above $50,000 (T)	Low (T)	Reject Loan
Rule 6:	Bad (F)	Above $50,000 (T)	High (F)	Reject Loan
Rule 7:	Bad (F)	Below $50,000 (F)	Low (T)	Reject Loan
Condition	Credit Score	Annual Income	Debt-to-Income Ratio	Action
Rule 8:	Bad (F)	Below $50,000 (F)	High (F)	Reject Loan

Explanation:

- **Rule 1**: If the **credit score is good**, the **annual income is above $50,000**, and the **debt-**

to-income ratio is low, approve the loan.

- **Rule 2**: If the **credit score is good**, the **annual income is above $50,000**, and the **debt-to-income ratio is high**, approve the loan.
- **Rule 3**: If the **credit score is good**, the **annual income is below $50,000**, and the **debt-to-income ratio is low**, reject the loan.
- **Rule 4**: If the **credit score is good**, the **annual income is below $50,000**, and the **debt-to-income ratio is high**, reject the loan.
- **Rule 5 to Rule 8**: The remaining rules involve cases where the **credit score is bad**, leading to the loan being rejected regardless of income or debt ratio.

Types of Decision Tables

1. Limited Entry Decision Table:
    - In this table, the conditions only have a binary choice: True or False. The possible outcomes (actions) are listed accordingly.
2. Extended Decision Table:
    - This table allows for more than two possible values for each condition. For example, a customer's credit score might have categories like "Excellent," "Good," "Fair," and "Poor," rather than just "Good" or "Bad."

Advantages of Using Decision Tables:

1. **Clear Representation**: Decision tables present complex decision logic in a simple, structured format.
2. **Eliminates Ambiguity**: It helps in identifying any possible conflicts or gaps in the decision-making logic.
3. **Consistency**: Ensures that all conditions and actions are covered, reducing the risk of errors.
4. **Scalability**: They can handle multiple conditions and actions, making them suitable for complex decision-making.

Entity-Relationship (ER)

An **Entity-Relationship (ER) diagram** is a visual representation of the entities in a system and the relationships between them. It is used in database design to structure data, identify the relationships between data points, and understand how different pieces of information relate to one another within a system. ER diagrams are foundational to creating relational databases and are often used in the conceptual design phase of a database.

Components of an Entity-Relationship (ER) Diagram

1. **Entity**:
    - **Symbol**: A **rectangle**.
    - **Description**: An entity represents a real-world object or concept that has data

stored about it. Entities can be physical objects like a "Customer" or "Product," or abstract concepts like an "Order" or "Course."
    - **Example**: "Customer," "Product," "Employee."

2. Attribute:
    - **Symbol**: An **oval**.
    - **Description**: An attribute represents a property or characteristic of an entity. For example, a "Customer" entity may have attributes like "Name," "Address," or "Phone Number."
    - Types of Attributes:
        - **Simple Attribute**: Cannot be divided further (e.g., "Age").
        - **Composite Attribute**: Can be broken down into smaller parts (e.g., "Full Name" can be divided into "First Name" and "Last Name").
        - **Derived Attribute**: Its value can be derived from other attributes (e.g., "Age" derived from "Date of Birth").
        - **Multi-valued Attribute**: Can hold multiple values (e.g., "Phone Numbers" for a customer).

3. Relationship:
    - **Symbol**: A **diamond**.
    - **Description**: A relationship connects two or more entities and shows how they interact with each other. For instance, a "Customer" may "Place" an "Order."
    - Types of Relationships:
        - **One-to-One (1:1)**: One instance of an entity is related to exactly one instance of another entity (e.g., a "Person" has one "Passport").
        - One-to-Many (1

          ): One instance of an entity is related to multiple instances of another entity (e.g., a "Customer" can place many "Orders").
        - Many-to-Many (M

          ): Multiple instances of one entity are related to multiple instances of another entity (e.g., a "Student" can enroll in many "Courses," and a "Course" can have many "Students").

4. Primary Key:
    - **Symbol**: Underlined **attribute**.
    - **Description**: A primary key is an attribute or set of attributes that uniquely identifies each instance of an entity.
    - **Example**: For a "Customer" entity, "Customer ID" might be the primary key.

5. Cardinality:
    - **Symbol**: Represented next to the relationship line (e.g., 1:1, 1

      , M

).

- **Description**: Cardinality indicates the number of instances of one entity that can be associated with instances of another entity.

Example of an ER Diagram for an Online Shopping System

Let's design a simple ER diagram for an online shopping system that includes Customers, Orders, and Products.

***Entities**:*

1. Customer:
   - Attributes: Customer_ID (Primary Key), Name, Email, Address.
2. Order:
   - Attributes: Order_ID (Primary Key), Order_Date, Total_Amount.
3. Product:
   - Attributes: Product_ID (Primary Key), Name, Price, Quantity.

Relationships:

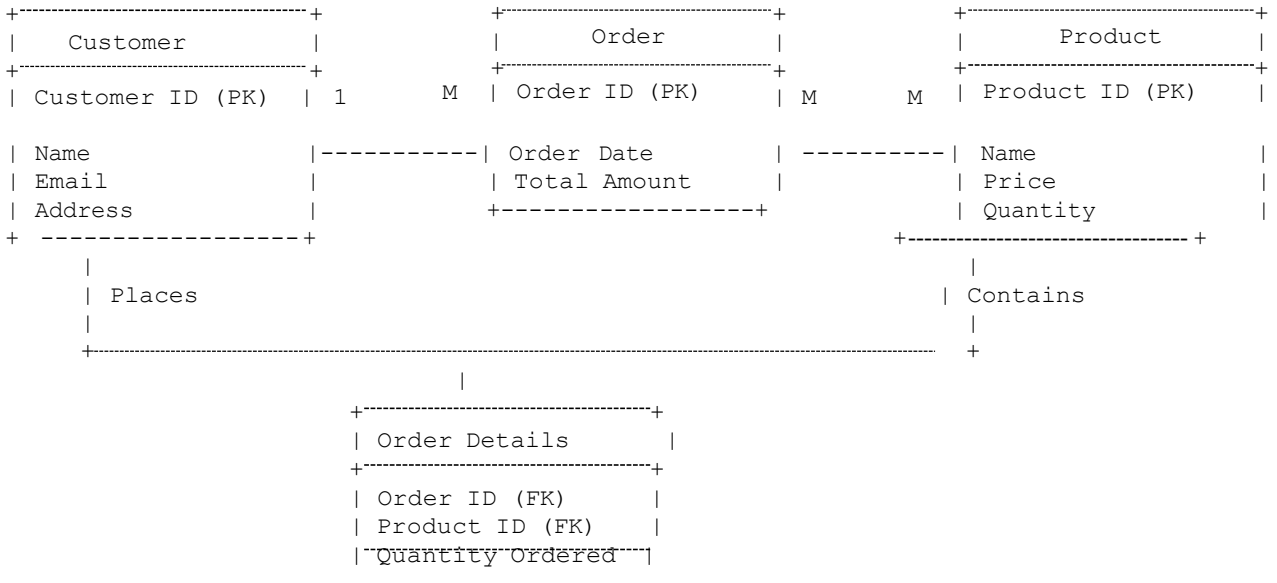
1. **Places** (between Customer and Order):
   - A customer can place many orders, but each order is placed by only one customer. This is a **One-to-Many (1**

) relationship.

2. **Contains** (between Order and Product):
   - An order can contain many products, and each product can appear in many orders. This is a **Many-to-Many (M**

) relationship, so we would introduce an associative entity (OrderDetails) to manage this.

ER Diagram Example:



Explanation:

- **Entities**:
  - **Customer** has attributes like "Customer_ID," "Name," and "Email."
  - **Order** has attributes like "Order_ID," "Order_Date," and "Total_Amount."
  - **Product** has attributes like "Product_ID," "Name," "Price," and "Quantity."
- Relationships:
  - **Places**: A **one-to-many** relationship where one **Customer** can place multiple Orders.
  - **Contains**: A **many-to-many** relationship between **Order** and **Product**, represented by the associative entity **OrderDetails**, which tracks the quantity of each product in the order.
- Cardinality:
  - The line connecting **Customer** and **Order** is labeled **1**

    , showing that each customer can place multiple orders.

  - The relationship between **Order** and **Product** is many-to-many, so it is managed by the **Order Details** table, which contains foreign keys from both **Order** and **Product**.

Object-Oriented Analysis and Design

Object-Oriented Analysis and Design (OOAD) is a structured method for analyzing and designing a system by modeling it as a group of interacting objects. It involves both the **analysis** (understanding the problem domain) and the **design** (structuring the solution domain) of the system using object-oriented principles. OOAD is a methodology based on the concepts of **objects**, **classes**, **inheritance**, **polymorphism**, **encapsulation**, and **abstraction**.

Key Concepts in Object-Oriented Analysis and Design:

1. **Object**:
   - An **object** represents a real-world entity or concept with attributes (data) and behaviors (methods). Objects can interact with one another by sending messages to each other.
   - Example: A **Car** object might have attributes like **color**, **model**, **engine type**, and behaviors like **start()**, **accelerate()**, **stop()**.
2. Class:
   - A **class** is a blueprint or template from which objects are created. It defines the properties (attributes) and behaviors (methods) common to all objects of that type.
   - Example: A **Car** class defines common attributes and methods for all car objects.
3. Encapsulation:
   - Encapsulation refers to the bundling of data (attributes) and methods (functions) that operate on the data into a single unit, known as a class. It also involves restricting direct access to some of the object's components, which is known as **information hiding**.
   - Example: The speed of a **Car** object may be an encapsulated attribute, accessible only through a **getSpeed()** method.
4. Inheritance:
   - Inheritance is the mechanism by which one class can inherit properties and behaviors (methods) from another class. It allows for code reuse and establishes a hierarchical relationship between classes.
   - Example: A **Sedan** class may inherit from the **Car** class, inheriting its attributes like **color** and **engine type** and behaviors like **accelerate()**.
5. Polymorphism:
   - Polymorphism allows objects of different classes to be treated as objects of a common superclass. It also means that different classes can provide different implementations of methods that share the same name.
   - Example: A **Car** class and a **Truck** class both have an **accelerate()** method, but each may implement it differently based on the type of vehicle.
6. Abstraction:
   - Abstraction is the concept of hiding complex implementation details and showing only the essential features of an object or system. It helps in focusing on high-level functionality and reduces complexity.
   - Example: The **Car** class might expose a high-level method like **startEngine()**, but it abstracts away the underlying details like turning the ignition key or starting the engine.

Phases of Object-Oriented Analysis and Design:

OOAD typically follows a series of steps that span both the **analysis** (problem domain) and **design** (solution domain). Below are the key phases:

1. Object-Oriented Analysis (OOA):

Understand the problem domain, identify objects, and define their relationships and behaviours.

Activities:

- **Identify Use Cases**: Analyze the functional requirements of the system. Use cases describe interactions between the system and external actors (users or other systems).
- **Identify Objects and Classes**: Determine the key objects in the system, their attributes, and the operations they need to perform. This is done by analyzing the problem space and identifying real-world entities.
- **Define Relationships**: Establish relationships between objects (e.g., inheritance, associations).
- **Create Class Diagrams**: Define classes, their attributes, and methods. Represent them in a UML (Unified Modeling Language) class diagram.

A conceptual model of the system, which is independent of how the system will be implemented.

2. Object-Oriented Design (OOD):

Define how the system will be built by creating the solution space.

Activities:

- **Refine the Design**: Design the system architecture and refine the analysis model into a detailed design. Focus on interactions between objects and how they collaborate.
- **Define Object Interfaces**: Determine the exact methods and interactions between objects. Define how objects will communicate with each other.
- **Choose Data Structures**: Design appropriate data structures for storing and manipulating data.
- **Define Class Relationships**: Specify relationships between classes (e.g., aggregation, composition).
- **Design Patterns**: Apply design patterns, such as Singleton, Observer, Factory, and Strategy, to address common design problems and ensure flexibility and reusability.

A detailed design model that can be implemented in code.

3. Implementation:

Convert the design into actual code using an object-oriented programming language (such as Java, C++, Python).

Activities:

- **Write Code**: Implement the classes, methods, and relationships as defined in the design.
- **Ensure Code Quality**: Follow coding standards, test individual components, and integrate them to ensure correct functionality.

Techniques Used in OOAD:

1. Use Case Diagram:
   - A use case diagram is a behavioural diagram that captures the functional requirements of a system from the user's perspective. It shows the system's interactions with external actors.
   - Example: In an online shopping system, actors like **Customer**, **Admin**, and **Payment Processor** might interact with use cases like **Place Order**, **Process Payment**, and **Manage Inventory**.
2. Class Diagram:
   - A class diagram shows the structure of the system by representing its classes, their attributes, methods, and the relationships between them.
   - Example: A **Customer** class may be associated with an **Order** class, indicating that customers place orders. The **Order** class would contain information about the order, such as order details and status.
3. Sequence Diagram:
   - A sequence diagram shows how objects interact with each other over time. It emphasizes the sequence of messages exchanged between objects to accomplish a specific task.
   - Example: A sequence diagram for **placing an order** might show the interactions between the **Customer**, **Shopping Cart**, **Order**, and **Payment System** objects.
4. State Diagram:
   - A state diagram models the behaviour of an object in response to various events. It shows the states an object can be in and the transitions between these states.
   - Example: An **Order** object could be in states like "Placed," "Shipped," and "Delivered," with transitions triggered by events like "Process Payment" or "Ship Order."
5. Activity Diagram:
   - An activity diagram represents the flow of control or the sequence of actions in the system. It's often used for modeling the workflow of use cases or business processes.
   - Example: An activity diagram for **checking out an order** could include activities like "Select Items," "Enter Shipping Info," "Enter Payment Info," and "Confirm Order."

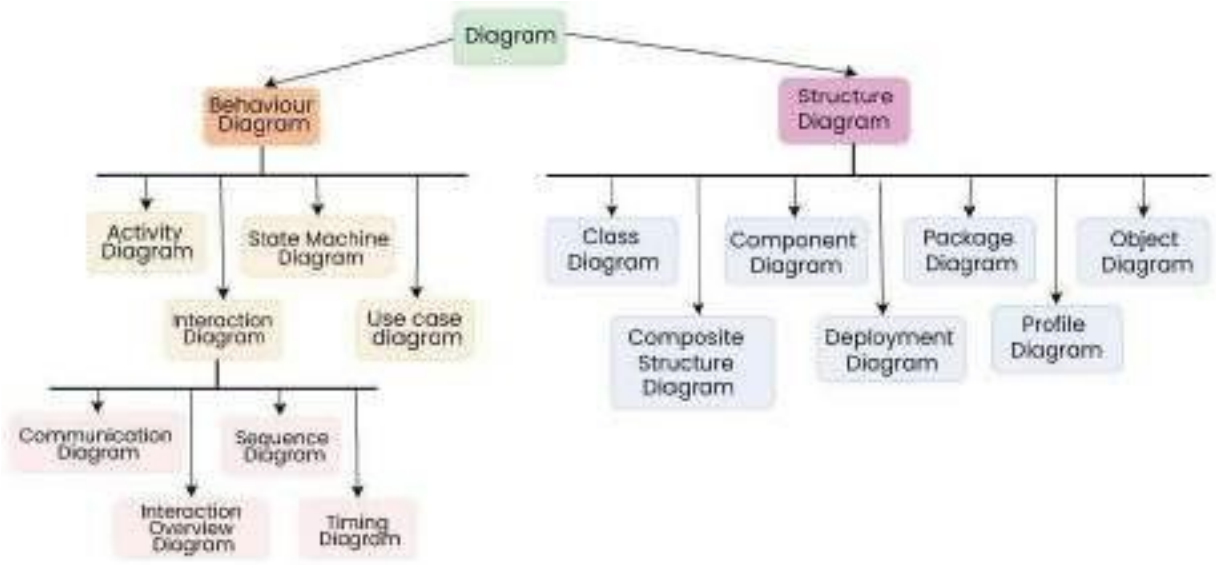
Benefits of Object-Oriented Analysis and Design:

1. **Modularity**: Objects are self-contained, which means the system is broken down into manageable components.
2. **Reusability**: Inheritance allows code to be reused, making development faster and more efficient.
3. **Maintainability**: Changes made to one object or class don't require changes to others, making the system easier to maintain.
4. **Scalability**: Object-oriented systems are easier to extend with new features or changes.
5. **Flexibility**: Polymorphism and abstraction allow the system to evolve without breaking existing code.

UML (Unified Modeling Language) Diagram

A **UML (Unified Modeling Language)** diagram is a standardized visual representation used in object-oriented modeling to describe the structure and behavior of a system. UML provides a set of diagramming conventions for capturing both the static and dynamic aspects of a system. These diagrams help software developers, system architects, and business analysts communicate complex concepts, system structure, and functionality effectively.

UML diagrams are broadly classified into **structural diagrams** and **behavioural diagrams**, each serving different purposes in system modeling.



1. Structural Diagrams

These diagrams show the **static structure** of a system, including objects, classes, and their relationships.

Represents the classes of a system, their attributes, operations (methods), and the relationships between them.

Components:

- **Classes** (rectangles divided into three sections for name, attributes, and operations).
- **Associations** (lines connecting classes).
- **Inheritance** (arrow pointing from subclass to superclass).
- **Aggregation/Composition** (representing whole-part relationships).
- **Multiplicities** (indicating how many instances of a class can be associated with another).

Example:

- A class **Customer** with attributes like **name**, **address**, and methods like **place Order(**, and a class **Order** associated with the **Customer** class.

Component Diagram

Purpose: Describes the physical components in a system and their relationships. It focuses on the system's architecture.

Components:

- **Components** (depicted by rectangles).
- **Interfaces** (depicted by small circles or lollipop symbols).
- **Dependencies** (dashed arrows indicating dependency relationships between components).

Deployment Diagram

Purpose: Shows the physical deployment of artifacts (such as software, hardware, or nodes) in a system.

Components:

- **Nodes** (rectangles representing physical hardware or virtual machines).
- **Artifacts** (depicted as document icons).
- **Associations** (lines showing communication between nodes).

Object Diagram

Purpose: Represents a snapshot of the objects in the system and their relationships at a particular moment in time.

Components:

- Objects (depicted like classes but with specific instance names and values).
- Links (depicted as solid lines connecting objects).

2. Behavioural Diagrams

These diagrams show the **dynamic aspects** of a system, focusing on interactions, state changes, and the flow of control.

Use Case Diagram

Captures the functional requirements of a system, showing the interactions between users (actors) and system use cases (functions).

Components:

- **Actors** (stick figures representing users or other systems).
- **Use Cases** (ovals representing system functions).
- **Associations** (lines connecting actors to use cases).

Example:

- A **Customer** actor interacting with a **Place Order** use case.

Sequence Diagram

Describes how objects interact in a particular sequence of events. Focuses on the order of messages exchanged between objects over time.

Components:

- **Objects** (depicted as rectangles at the top).
- **Lifelines** (dashed vertical lines showing the presence of an object over time).
- **Messages** (arrows representing interactions between objects).

Example:

- A sequence diagram showing how a **Customer** sends a message to **Shopping Cart**, which in turn interacts with **Payment Processor**.

Activity Diagram

Represents workflows and processes, showing the sequence of actions or operations.

Components:

- **Start and End nodes** (depicted as filled circles).
- **Activities** (rectangles with rounded corners).

- **Transitions** (arrows showing flow).
- **Decision nodes** (diamonds representing choices in the flow).

Example:

- An activity diagram for the **order processing** workflow.

State Machine Diagram

Describes the states an object can be in and the transitions between these states based on events.

Components:

- **States** (rounded rectangles).
- **Transitions** (arrows representing events that cause a state change).
- **Events** (triggers for state transitions).

Example:

- A state machine diagram for a **Ticket** object with states like "Available," "Reserved," and "Sold."

Communication Diagram

Similar to sequence diagrams but emphasizes the interactions between objects and the messages they exchange.

Components:

- **Objects** (represented as rectangles).
- **Messages** (labeled arrows connecting objects).

Interaction Overview Diagram

A high-level view that shows the flow of control within a system. It combines elements of activity and sequence diagrams.

Components:

- **Activity States** (depicted by rounded rectangles).
- **Control Flows** (arrows between activities).

UNIT 3

DATABASE MANAGEMENT SYSTEMS

SYLLABUS

DBMS – types and evolution, RDBMS, OODBMS, RODBMS, Data warehousing, Data Mart, Data mining.

****

DBMS- Data Base Management System

Database is collection of data which is related by some aspect. Data is collection of facts and figures which can be processed to produce information. Name of a student, age, class and her subjects can be counted as data for recording purposes.

Mostly data represents recordable facts. Data aids in producing information which is based on facts. For example, if we have data about marks obtained by all students, we can then conclude about toppers and average marks etc.

A database management system stores data, in such a way which is easier to retrieve, manipulate and helps to produce information.

Characteristics

Traditionally data was organized in file formats. DBMS was all new concepts then and all the research was done to make it to overcome all the deficiencies in traditional style of data management. Modern DBMS has the following characteristics:

- Real-world entity: Modern DBMS are more realistic and uses real world entities to design its architecture. It uses the behavior and attributes too. For example, a school database may use student as entity and their age as their attribute.
- Relation-based tables: DBMS allows entities and relations among them to form as tables. This eases the concept of data saving. A user can understand the architecture of database just by looking at table names etc.
- Isolation of data and application: A database system is entirely different than its data. Where database is said to active entity, data is said to be passive one on which the database works and organizes. DBMS also stores metadata which is data about data, to ease its own process.
- Less redundancy: DBMS follows rules of normalization, which splits a relation

when any of its attributes is having redundancy in values. Following normalization, which itself is a mathematically rich and scientific process, make the entire database to contain as less redundancy as possible.

- Consistency: DBMS always enjoy the state on consistency where the previous form of data storing applications like file processing does not guarantee this. Consistency is a state where every relation in database remains consistent. There exist methods and techniques, which can detect attempt of leaving database in inconsistent state.

- Query Language: DBMS is equipped with query language, which makes it more efficient to retrieve and manipulate data. A user can apply as many and different filtering options, as he or she wants. Traditionally it was not possible where file-processing system was used.

- ACID Properties: DBMS follows the concepts for ACID properties, which stands for Atomicity, Consistency, Isolation and Durability. These concepts are applied on transactions, which manipulate data in database. ACID properties maintains database in healthy state in multi-transactional environment and in case of failure.

- Multiuser and Concurrent Access: DBMS support multi-user environment and allows them to access and manipulate data in parallel. Though there are restrictions on transactions when they attempt to handle same data item, but users are always unaware of them.

- Multiple views: DBMS offers multiples views for different users. A user who is in sales department will have a different view of database than a person working in production department. This enables user to have a concentrate view of database according to their requirements.

- Security: Features like multiple views offers security at some extent where users are unable to access data of other users and departments. DBMS offers methods to impose constraints while entering data into database and retrieving data at later stage. DBMS offers many different levels of security features, which enables multiple users to have different view with different features. For example, a user in sales department cannot see data of purchase department is one thing, additionally how much data of sales department he can see, can also be managed. Because DBMS is not saved on disk as traditional file system it is very hard for a thief to break the code.

DBMS Architecture

Database Management System (DBMS) architecture defines how data is stored, accessed, and managed within a database system. The architecture provides a framework that supports efficient data operations and ensures security, scalability, and reliability.

Types of DBMS Architecture

1. Single-Tier Architecture

- Simplest form of DBMS architecture.
- **Database and application** reside on the same machine.
- Suitable for small-scale applications.

Advantages:

- Easy to implement.
- Minimal overhead.

Disadvantages:

- Limited scalability.
- Poor performance for multi-user environments.

2. Two-Tier Architecture

- The **client-server model**, where:
  - The **client** handles the user interface and application logic.
  - The **server** manages the database and handles query processing.

Components:

1. **Client Tier**:
   - Sends requests to the database server.
   - Examples: Front-end applications like Microsoft Access.
2. Server Tier:
   - Executes the database query.
   - Examples: MySQL, PostgreSQL.

Advantages:

- Better scalability than single-tier.
- Easier maintenance and management.

Disadvantages:

- Limited to a single application per client.
- Increased dependency on the network connection.

3. Three-Tier Architecture

- Most widely used architecture in modern systems.
- Introduces a **middle layer** between the client and the server.

Layers:

1. **Presentation Layer (Client)**:
    - User interface to interact with the system.
    - Examples: Web browsers, GUI-based applications.
2. Application Layer (Middle Tier):
    - Processes business logic and routes requests between the client and the database server.
    - Examples: Application servers like Apache Tomcat or Node.js.
3. Database Layer (Server):
    - Responsible for data storage, query execution, and transaction management.

Advantages:

- Scalability: Supports multiple clients and applications.
- Improved security: Middle-tier acts as a mediator, ensuring secure communication.
- Flexibility: Business logic is separate from the database and user interface.

Disadvantages:

- Complex to implement.
- Higher setup and maintenance costs.

RDBMS-Relational Database Management System

In relational databases, the relationship between data files is relational. Hierarchical and network databases require the user to pass a hierarchy in order to access needed data. These databases connect to the data in different files by using common data numbers or a key field. Data in relational databases is stored in different access control tables, each having a key field that mainly identifies each row. In the relational databases are more reliable than either the hierarchical or network database structures. In relational databases, tables or files filled up with data are called relations designates a row or record, and columns are referred to as attributes or fields.

Relational databases work on each table has a key field that uniquely indicates each row, and that these key fields can be used to connect one table of data to another.

The relational database has two major reasons

1. Relational databases can be used with little or no training.
2. Database entries can be modified without specify the entire body.

Properties of Relational Tables

In the relational database we have to follow some properties which are given below.

- It's Values are Atomic
- In Each Row is alone.
- Column Values are of the same thing.
- Columns are undistinguished.
- Sequence of Rows is Insignificant.
- Each Column has a common Name.

OODBMS – Object oriented Database Management System

In this Model we have to discuss the functionality of the object oriented Programming It takes more than storage of programming language objects. Object DBMS's increase the semantics of the C++ and Java. It provides full-featured database programming capability, while containing native language compatibility. It adds the database functionality to object programming languages. This approach is the analogical of the application and database development into a constant data model and language environment. Applications require less code, use more natural data modeling, and code bases are easier to maintain. Object developers can write complete database applications with a decent amount of additional effort.

The object-oriented database derivation is the integrity of object-oriented programming language systems and consistent systems. The power of the object-oriented databases comes from the cyclical treatment of both consistent data, as found in databases, and transient data, as found in executing programs.
Object-oriented databases use small, recyclable separated of software called objects. The objects themselves are stored in the object-oriented database. Each object contains of two elements:

1. Piece of data (e.g., sound, video, text, or graphics).
2. Instructions or software programs called methods, for what to do with the data.

Disadvantage of Object-oriented databases

1. Object-oriented databases have these disadvantages.
2. Object-oriented database are more expensive to develop.
3. In the Most organizations are unwilling to abandon and convert from those databases.

The benefits to object-oriented databases are compelling. The ability to mix and match reusable objects provides incredible multimedia capability.

Query Processing

Upper levels of the data integration problem

- How to construct mappings from sources to a single mediated schema
- How queries posed over the mediated schema are reformulated over the sources

Basic Steps in Query Processing

1. Parsing and translation
2. Optimization
3. Evaluation

Evaluation

The query-execution engine takes a query-evaluation plan, executes that plan, and returns the answers to the query. A relational algebra expression may have many equivalent expressions. Each relational algebra operation can be evaluated using one of several different algorithms. Correspondingly, a relational-algebra expression can be evaluated in many ways. Annotated expression specifying detailed evaluation strategy is called an evaluation-plan.

Query Optimization

Amongst all equivalent evaluation plans choose the one with lowest cost. Cost is estimated using statistical information from the database catalog.

SQL

SQL is Structured Query Language, which is a computer language for storing, manipulating and retrieving data stored in relational database. SQL is the standard language for Relation Database System. All relational database management systems like MySQL, MS Access, and Oracle, Sybase, Informix, postgress and SQL Server use SQL as standard database language.

Why SQL?

- Allows users to access data in relational database management systems.
- Allows users to describe the data.
- Allows users to define the data in database and manipulate that data.
- Allows to embed within other languages using SQL modules, libraries & pre-compilers.
- Allows users to create and drop databases and tables.
- Allows users to create view, stored procedure, functions in a database.

Data warehouse

- Data warehouse is data management and data analysis
- Goal: is to integrate enterprise wide corporate data into a single repository from which users can easily run queries

A **Data Warehouse** is separate from DBMS, it stores a huge amount of data, which is typically collected from multiple heterogeneous sources like files, DBMS, etc. The goal is to produce statistical results that may help in decision-making. For example, a college might want to see quick different results, like how the placement of CS students has improved over the last 10 years, in terms of salaries, counts, etc.

Need for Data Warehouse

An ordinary Database can store MBs to GBs of data and that too for a specific purpose. For storing data of TB size, the storage shifted to the Data Warehouse. Besides this, a transactional database doesn't offer itself to analytics. To effectively perform analytics, an organization keeps a central Data Warehouse to closely study its business by organizing, understanding, and using its historical data for making strategic decisions and analyzing trends.

Benefits of Data Warehouse

- **Better business analytics:** Data warehouse plays an important role in every business to store and analysis of all the past data and records of the company. which can further increase the understanding or analysis of data for the company.
- **Faster Queries: The data** warehouse is designed to handle large queries that's why it runs queries faster than the database.
- **Improved data Quality:** In the data warehouse the data you gathered from different sources is being stored and analyzed it does not interfere with or add data by itself so your quality of data is maintained and if you get any issue regarding data quality then the data warehouse team will solve this.
- **Historical Insight:** The warehouse stores all your historical data which contains details about the business so that one can analyze it at any time and extract insights from it.
- **The major benefit of data warehousing are high returns on investment.**
- **Underestimation of resources for data loading**
- **Hidden problems with source systems**
- **Required data not captured**
- **Increased end-user demands**
- **Data homogenization**
- **High demand for resources**
- **Data ownership**
- **High maintenance**
- **Long-duration projects**
- **Complexity of integration**

Data Warehouse vs DBMS

Database	Data Warehouse
A common Database is based on operational or transactional processing. Each operation is an indivisible transaction.	A data Warehouse is based on analytical processing.
Generally, a Database stores current and up-to-date data which is used for daily operations.	A Data Warehouse maintains historical data over time. Historical data is the data kept over years and can used for trend analysis, make future predictions and decision support.
A database is generally application specific. **Example –** A database stores related data, such as the student details in a school.	A Data Warehouse is integrated generally at the organization level, by combining data from different databases. **Example –** A data warehouse integrates the data from one or more databases , so that analysis can be done to get results , such as the best performing school in a city.
Constructing a Database is not so expensive.	Constructing a Data Warehouse can be expensive.

Example Applications of Data Warehousing

Data Warehousing can be applied anywhere where we have a huge amount of data and we want to see statistical results that help in decision making.

- **Social Media Websites:** The social networking websites like Facebook, Twitter, Linkedin, etc. are based on analyzing large data sets. These sites gather data related to members, groups, locations, etc., and store it in a single central repository. Being a large amount of data, Data Warehouse is needed for implementing the same.
- **Banking:** Most of the banks these days use warehouses to see the spending patterns of account/cardholders. They use this to provide them with special offers, deals, etc.
- **Government:** Government uses a data warehouse to store and analyze tax payments which are used to detect tax thefts.

Features of Data Warehousing

Data warehousing is essential for modern data management, providing a strong foundation for organizations to consolidate and analyze data strategically. Its distinguishing features empower businesses with the tools to make informed decisions and extract valuable insights from their data.

- **Centralized Data Repository:** Data warehousing provides a centralized repository for all enterprise data from various sources, such as transactional databases, operational systems, and external sources. This enables organizations to have a comprehensive view of their data, which can help in making informed business decisions.
- **Data Integration:** Data warehousing integrates data from different sources into a single, unified view, which can help in eliminating data silos and reducing data inconsistencies.
- **Historical Data Storage:** Data warehousing stores historical data, which enables organizations to analyze data trends over time. This can help in identifying patterns and anomalies in the data, which can be used to improve business performance.
- **Query and Analysis:** Data warehousing provides powerful query and analysis capabilities that enable users to explore and analyze data in different ways. This can help in identifying patterns and trends, and can also help in making informed business decisions.
- **Data Transformation:** Data warehousing includes a process of data transformation, which involves cleaning, filtering, and formatting data from various sources to make it consistent and usable. This can help in improving data quality and reducing data inconsistencies.
- **Data Mining:** Data warehousing provides data mining capabilities, which enable organizations to discover hidden patterns and relationships in their data. This can help in identifying new opportunities, predicting future trends, and mitigating risks.
- **Data Security:** Data warehousing provides robust data security features, such as access controls, data encryption, and data backups, which ensure that the data is secure and protected from unauthorized access.

Advantages of Data Warehousing

- **Intelligent Decision-Making:** With centralized data in warehouses, decisions may be made more quickly and intelligently.
- **Business Intelligence:** Provides strong operational insights through business intelligence.
- **Historical Analysis:** Predictions and trend analysis are made easier by storing past data.
- **Data Quality:** Guarantees data quality and consistency for trustworthy reporting.
- **Scalability:** Capable of managing massive data volumes and expanding to meet changing requirements.
- **Effective Queries:** Fast and effective data retrieval is made possible by an optimized structure.
- **Cost reductions:** Data warehousing can result in cost savings over time by reducing data management procedures and increasing overall efficiency, even when there are setup costs initially.
- **Data security:** Data warehouses employ security protocols to safeguard confidential information, guaranteeing that only authorized personnel are granted access to certain data.

Disadvantages of Data Warehousing

- **Cost:** Building a data warehouse can be expensive, requiring significant investments in hardware, software, and personnel.
- **Complexity:** Data warehousing can be complex, and businesses may need to hire specialized personnel to manage the system.
- **Time-consuming:** Building a data warehouse can take a significant amount of time, requiring businesses to be patient and committed to the process.
- **Data integration challenges:** Data from different sources can be challenging to integrate, requiring significant effort to ensure consistency and accuracy.

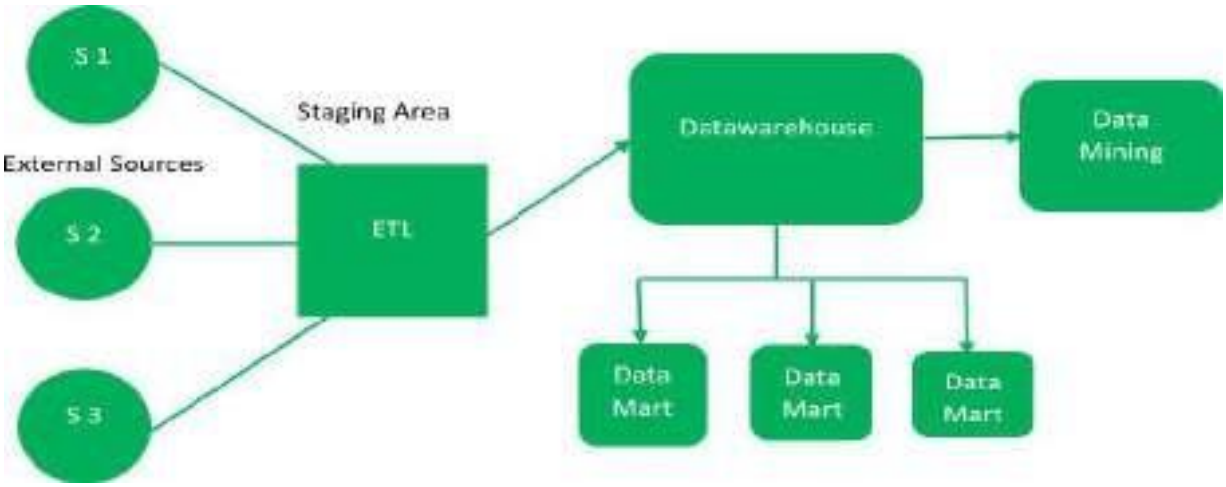
- **Data security:** Data warehousing can pose data security risks, and businesses must take measures to protect sensitive data from unauthorized access or breaches.

Data Warehouse Architecture

A D**ata-Warehouse** is a heterogeneous collection of data sources organized under a unified schema. There are 2 approaches for constructing a data warehouse: The top-down approach and the Bottom-up approach are explained below.

What is Top-Down Approach?

The initial approach developed by Bill Inmon known as the top-down approach starts with building a single source data warehouse for the whole company. Merges and processes external data through the ETL (Extract, Transform, Load) process and subsequently stores them in the data warehouse. Specialized data marts for different organizations departments, for instance, the finance department are then formed from there. The strength of this method is that it offers a clear structure for managing data, however, this method can be expensive as well as time-consuming and for that reason, it is ideal for large organizations only.



The essential components are discussed below:

1. **External Sources:** External source is a source from where data is collected irrespective of the type of data. Data can be structured, semi structured and unstructured as well.

2. **Stage Area:** Since the data, extracted from the external sources does not follow a particular format, so there is a need to validate this data to load into data-warehouse. For this purpose, it is recommended to use **ETL** tool.
   - **E(Extracted):** Data is extracted from External data source.
   - **T(Transform):** Data is transformed into the standard format.

- **L(Load):** Data is loaded into data-warehouse after transforming it into the standard format.

3. **Data-warehouse:** After cleansing of data, it is stored in the data warehouse as central repository. It actually stores the meta data and the actual data gets stored in the data marts. **Note** that data warehouse stores the data in its purest form in this top-down approach.

4. **Data Marts:** Data mart is also a part of storage component. It stores the information of a particular function of an organization which is handled by single authority. There can be as many number of data marts in an organization depending upon the functions. We can also say that data mart contains subset of the data stored in data warehouse.

5. **Data Mining:** The practice of analyzing the big data present in data warehouse is data mining. It is used to find the hidden patterns that are present in the database or in data warehouse with the help of algorithm of data mining.
   This approach is defined by **Inmon** as – data warehouse as a central repository for the complete organisation and data marts are created from it after the complete data warehouse has been created.

Advantages of Top-Down Approach

1. Since the data marts are created from the data-warehouse, provides consistent dimensional view of data marts.
2. Also, this model is considered as the strongest model for business changes. That's why, big organizations prefer to follow this approach.
3. Creating data mart from data warehouse is easy.
4. **Improved data consistency:** The top-down approach promotes data consistency by ensuring that all data marts are sourced from a common data warehouse. This ensures that all data is standardized, reducing the risk of errors and inconsistencies in reporting.
5. **Easier maintenance:** Since all data marts are sourced from a central data warehouse, it is easier to maintain and update the data in a top-down approach. Changes can be made to the data warehouse, and those changes will automatically propagate to all the data marts that rely on it.
6. **Better scalability:** The top-down approach is highly scalable, allowing organizations to add new data marts as needed without disrupting the existing infrastructure. This is particularly important for organizations that are experiencing rapid growth or have evolving business needs.
7. **Improved governance:** The top-down approach facilitates better governance by enabling centralized control of data access, security, and quality. This ensures that all data is managed consistently and that it meets the organization's standards for quality and compliance.
8. **Reduced duplication:** The top-down approach reduces data duplication by ensuring that data is stored only once in the data warehouse. This saves storage space and reduces the risk of data inconsistencies.
9. **Better reporting:** The top-down approach enables better reporting by providing a consistent view of data across all data marts. This makes it easier to create accurate and timely reports, which can improve decision-making and drive better business outcomes.
10. **Better data integration:** The top-down approach enables better data integration by ensuring that all data marts are sourced from a common data warehouse. This makes it

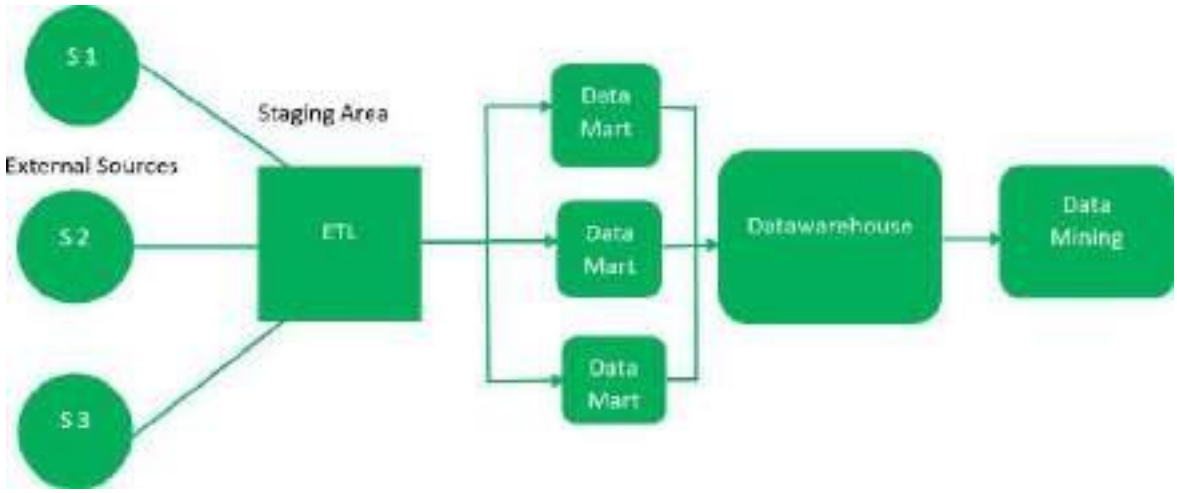
easier to integrate data from different sources and provides a more complete view of the organization's data.

Disadvantages of Top-Down Approach

1. The cost, time taken in designing and its maintenance is very high.
2. **Complexity:** The top-down approach can be complex to implement and maintain, particularly for large organizations with complex data needs. The design and implementation of the data warehouse and data marts can be time-consuming and costly.
3. **Lack of flexibility:** The top-down approach may not be suitable for organizations that require a high degree of flexibility in their data reporting and analysis. Since the design of the data warehouse and data marts is pre-determined, it may not be possible to adapt to new or changing business requirements.
4. **Limited user involvement:** The top-down approach can be dominated by IT departments, which may lead to limited user involvement in the design and implementation process. This can result in data marts that do not meet the specific needs of business users.
5. **Data latency:** The top-down approach may result in data latency, particularly when data is sourced from multiple systems. This can impact the accuracy and timeliness of reporting and analysis.
6. **Data ownership:** The top-down approach can create challenges around data ownership and control. Since data is centralized in the data warehouse, it may not be clear who is responsible for maintaining and updating the data.
7. **Cost:** The top-down approach can be expensive to implement and maintain, particularly for smaller organizations that may not have the resources to invest in a large-scale data warehouse and associated data marts.
8. **Integration challenges:** The top-down approach may face challenges in integrating data from different sources, particularly when data is stored in different formats or structures. This can lead to data inconsistencies and inaccuracies.

What is Bottom-Up Approach?

Bottom up Approach is the Ralph Kimball's approach of the construction of individual data marts that lie at the center of specific business goals or functions such as marketing or sales. These data marts are extracted transformed & loaded first to provide organizations' ability to generate reports instantly. In turn, these data marts are affiliated to the more centralized and broad data warehouse system. This is a more flexible method of training, cheaper and best recommendable in smaller organizations. Nevertheless, it entails the creation of data silos and disparities, and this may not allow an organization to have a coherent perspective in its various departments.



1. First, the data is extracted from external sources (same as happens in top-down approach).

2. Then, the data go through the staging area (as explained above) and loaded into data marts instead of data-warehouse. The data marts are created first and provide reporting capability. It addresses a single business area.

3. These data marts are then integrated into data-warehouse.

This approach is given by **Kinball** as – data marts are created first and provides a thin view for analyses and data-warehouse is created after complete data marts have been created.

Advantages of Bottom-Up Approach

1. As the data marts are created first, so the reports are quickly generated.
2. We can accommodate more number of data marts here and in this way data-warehouse can be extended.
3. Also, the cost and time taken in designing this model is low comparatively.
4. **Incremental development:** The bottom-up approach supports incremental development, allowing for the creation of data marts one at a time. This allows for quick wins and incremental improvements in data reporting and analysis.
5. **User involvement:** The bottom-up approach encourages user involvement in the design and implementation process. Business users can provide feedback on the data marts and reports, helping to ensure that the data marts meet their specific needs.
6. **Flexibility:** The bottom-up approach is more flexible than the top-down approach, as it allows for the creation of data marts based on specific business needs. This approach can be particularly useful for organizations that require a high degree of flexibility in their reporting and analysis.
7. **Faster time to value:** The bottom-up approach can deliver faster time to value, as the data marts can be created more quickly than a centralized data warehouse. This can be particularly useful for smaller organizations with limited resources.
8. **Reduced risk:** The bottom-up approach reduces the risk of failure, as data marts can be tested and refined before being incorporated into a larger data warehouse. This approach can also help to identify and address potential data quality issues early in the process.
9. **Scalability:** The bottom-up approach can be scaled up over time, as new data marts can be added as needed. This approach can be particularly useful for organizations that are growing rapidly or undergoing significant change.

10. **Data ownership:** The bottom-up approach can help to clarify data ownership and control, as each data mart is typically owned and managed by a specific business unit. This can help to ensure that data is accurate and up-to-date, and that it is being used in a consistent and appropriate way across the organization.

Disadvantage of Bottom-Up Approach

1. This model is not strong as top-down approach as dimensional view of data marts is not consistent as it is in above approach.
2. **Data silos:** The bottom-up approach can lead to the creation of data silos, where different business units create their own data marts without considering the needs of other parts of the organization. This can lead to inconsistencies and redundancies in the data, as well as difficulties in integrating data across the organization.
3. **Integration challenges:** Because the bottom-up approach relies on the integration of multiple data marts, it can be more difficult to integrate data from different sources and ensure consistency across the organization. This can lead to issues with data quality and accuracy.
4. **Duplication of effort:** In a bottom-up approach, different business units may duplicate effort by creating their own data marts with similar or overlapping data. This can lead to inefficiencies and higher costs in data management.
5. **Lack of enterprise-wide view:** The bottom-up approach can result in a lack of enterprise-wide view, as data marts are typically designed to meet the needs of specific business units rather than the organization as a whole. This can make it difficult to gain a comprehensive understanding of the organization's data and business processes.
6. **Complexity:** The bottom-up approach can be more complex than the top-down approach, as it involves the integration of multiple data marts with varying levels of complexity and granularity. This can make it more difficult to manage and maintain the data warehouse over time.
7. **Risk of inconsistency:** Because the bottom-up approach allows for the creation of data marts with different structures and granularities, there is a risk of inconsistency in the data. This can make it difficult to compare data across different parts of the organization or to ensure that reports are accurate and reliable.

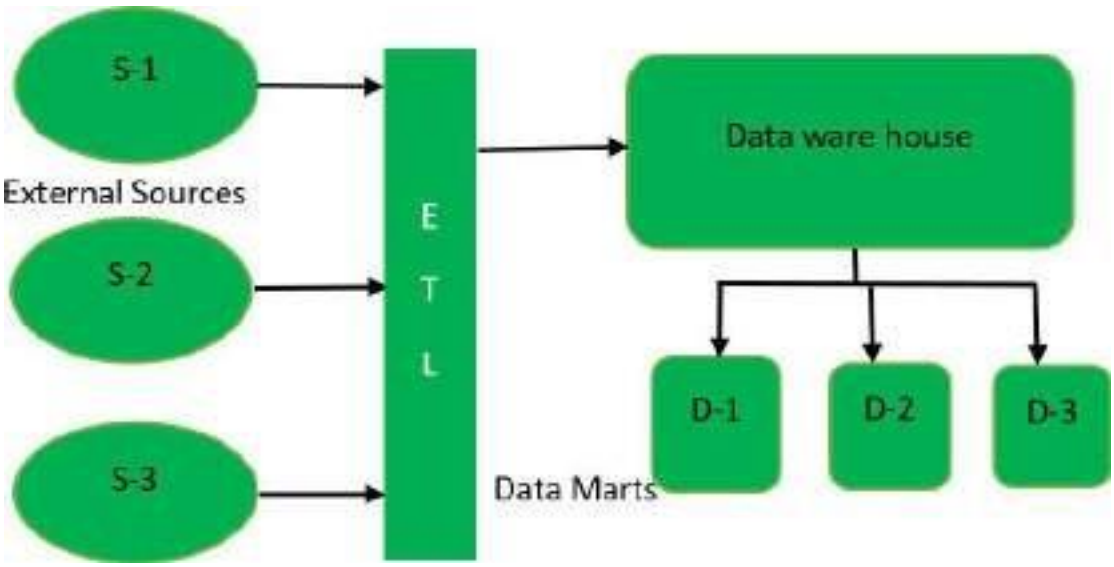
DATA MART

Data mart is such a storage component which is concerned on a specific department of an organization. It is a subset of the data stored in the data-warehouse. Data mart is focused only on particular function of an organization and it is maintained by single authority only, e.g.m finance, Marketing. Data Marts are small in size and are flexible.

Types of Data Mart:

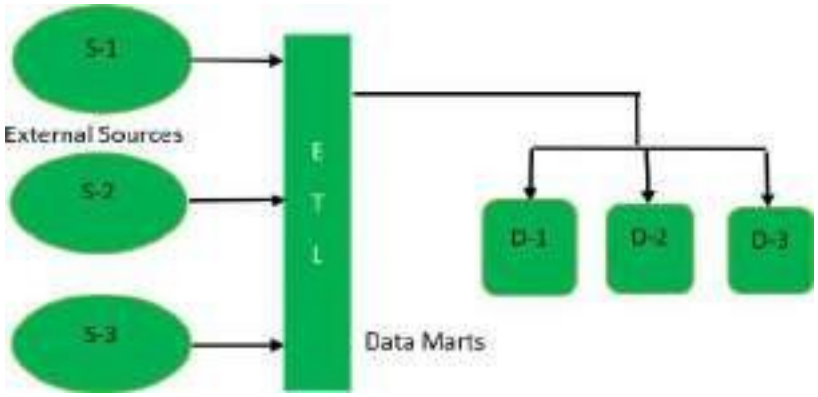
There are three types of data marts:

1. Dependent Data Mart –



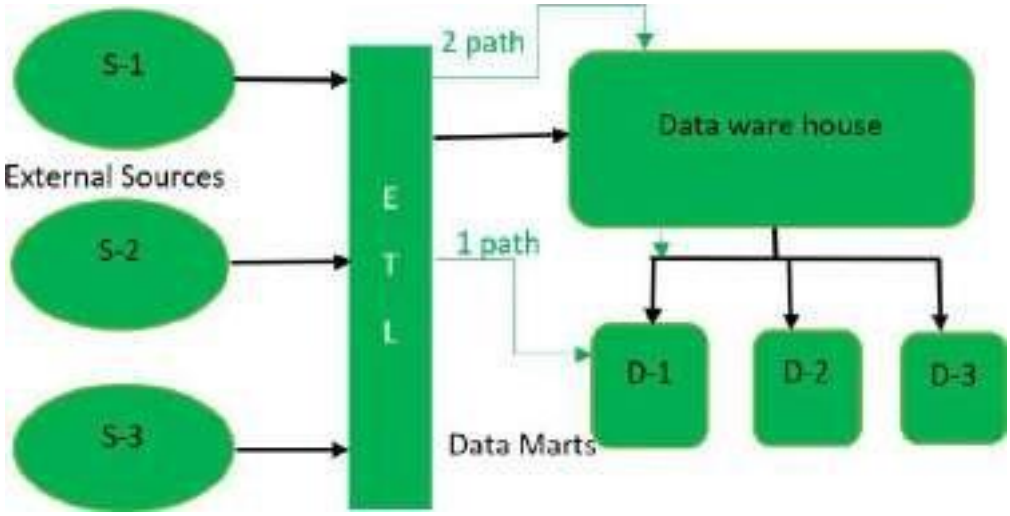
Dependent Data Mart is created by extracting the data from central repository, Data-warehouse. First data warehouse is created by extracting data (through ETL tool) from external sources and then data mart is created from data warehouse. Dependent data mart is created in top-down approach of data-warehouse architecture. This model of data mart is used by big organizations.

2. Independent Data Mart –



Independent Data Mart is created directly from external sources instead of data warehouse. First data mart is created by extracting data from external sources and then data-warehouse is created from the data present in data mart. Independent data mart is designed in bottom-up approach of data-warehouse architecture. This model of data mart is used by small organizations and is cost effective comparatively.

3. Hybrid Data Mart –



This type of Data Mart is created by extracting data from operational source or from data warehouse. 1Path reflects accessing data directly from external sources and 2Path reflects dependent data model of data mart.

Need of Data Mart:

1. Data Mart focuses only on functioning of particular department of an organization.
2. It is maintained by single authority of an organization.
3. Since, it stores the data related to specific part of an organization, data retrieval from it is very quick.
4. Designing and maintenance of data mart is found to be quite cinch as compared to data warehouse.
5. It reduces the response time of user as it stores small volume of data.
6. It is small in size due to which accessing data from it very fast.
7. This Storage unit is used by most of organizations for the smooth running of their departments.

Advantages of Data Mart:

1. Implementation of data mart needs less time as compared to implementation of data-warehouse as data mart is designed for a particular department of an organization.
2. Organizations are provided with choices to choose model of data mart depending upon cost and their business.
3. Data can be easily accessed from data mart.

4. It contains frequently accessed queries, so enable to analyze business trend.

Disadvantages of Data Mart:

1. Since it stores the data related only to specific function, so does not store huge volume of data related to each and every department of an organization like data-warehouse.
2. Creating too many data marts becomes cumbersome sometimes.

Features of data marts:

- **Subset of Data:** Data marts are designed to store a subset of data from a larger data warehouse or data lake. This allows for faster query performance since the data in the data mart is focused on a specific business unit or department.
- **Optimized for Query Performance:** Data marts are optimized for query performance, which means that they are designed to support fast queries and analysis of the data stored in the data mart.
- **Customizable:** Data marts are customizable, which means that they can be designed to meet the specific needs of a business unit or department.
- **Self-Contained:** Data marts are self-contained, which means that they have their own set of tables, indexes, and data models. This allows for easier management and maintenance of the data mart.
- **Security:** Data marts can be secured, which means that access to the data in the data mart can be controlled and restricted to specific users or groups.
- **Scalability:** Data marts can be scaled horizontally or vertically to accommodate larger volumes of data or to support more users.
- **Integration with Business Intelligence Tools:** Data marts can be integrated with business intelligence tools, such as Tableau, Power BI, or QlikView, which allows users to analyze and visualize the data stored in the data mart.
- **ETL Process:** Data marts are typically populated using an Extract, Transform, Load (ETL) process, which means that data is extracted from the larger data warehouse or data lake, transformed to meet the requirements of the data mart, and loaded into the data mart.

DATA MINING

Data mining is the process of discovering patterns and relationships in large datasets using techniques such as machine learning and statistical analysis. The goal of data mining is to extract useful information from large datasets and use it to make predictions or inform decision-making. Data mining is important because it allows organizations to uncover insights and trends in their data that would be difficult or impossible to discover manually.

Use Cases of Data Mining

Data mining has a wide range of applications and uses cases across many industries and domains. Some of the most common use cases of data mining include:

1. **Market Basket Analysis:** Market basket analysis is a common use case of data mining in the retail and e-commerce industries. It involves analyzing data on customer purchases to identify items that are frequently purchased together, and using this information to make recommendations or suggestions to customers.
2. **Fraud Detection:** Data mining is widely used in the financial industry to detect and prevent fraud. It involves analyzing data on transactions and customer behavior to identify patterns or anomalies that may indicate fraudulent activity.
3. **Customer Segmentation:** Data mining is commonly used in the marketing and advertising industries to segment customers into different groups based on their characteristics and behavior. This information can then be used to tailor marketing and advertising campaigns to specific segments of customers.
4. **Predictive Maintenance:** Data mining is increasingly used in the manufacturing and industrial sectors to predict when equipment or machinery is likely to fail or require maintenance. It involves analyzing data on the performance and usage of equipment to identify patterns that can indicate potential failures, and using this information to schedule maintenance and prevent downtime.
5. **Intrusion Detection:** Data mining is used in the cybersecurity industry to detect network intrusions and prevent cyber attacks. It involves analyzing data on network traffic and behavior to identify patterns that may indicate an attempted intrusion, and using this information to alert security teams and prevent attacks.

Data Mining Architecture

Data mining architecture refers to the overall design and structure of a data mining system. A data mining architecture typically includes several key components, which work together to perform data mining tasks and extract useful insights and information from data. Some of the key components of a typical data mining architecture include:

- **Data Sources:** Data sources are the sources of data that are used in data mining. These can include structured and unstructured data from databases, files, sensors, and other sources. Data sources provide the raw data that is used in data mining and can be processed, cleaned, and transformed to create a usable data set for analysis.
- **Data Preprocessing:** Data preprocessing is the process of preparing data for analysis. This typically involves cleaning and transforming the data to remove errors, inconsistencies, and irrelevant information, and to make it suitable for analysis. Data preprocessing is an

important step in data mining, as it ensures that the data is of high quality and is ready for analysis.

- **Data Mining Algorithms:** Data mining algorithms are the algorithms and models that are used to perform data mining. These algorithms can include supervised and unsupervised learning algorithms, such as regression, classification, and clustering, as well as more specialized algorithms for specific tasks, such as association rule mining and anomaly detection. Data mining algorithms are applied to the data to extract useful insights and information from it.
- **Data Visualization:** Data visualization is the process of presenting data and insights in a clear and effective manner, typically using charts, graphs, and other visualizations. Data visualization is an important part of data mining, as it allows data miners to communicate their findings and insights to others in a way that is easy to understand and interpret.

Types of Data Mining

There are many different types of data mining, but they can generally be grouped into three broad categories: descriptive, predictive, and prescriptive.

- **Descriptive data mining** involves summarizing and describing the characteristics of a data set. This type of data mining is often used to explore and understand the data, identify patterns and trends, and summarize the data in a meaningful way.
- **Predictive data mining** involves using data to build models that can make predictions or forecasts about future events or outcomes. This type of data mining is often used to identify and model relationships between different variables, and to make predictions about future events or outcomes based on those relationships.
- **Prescriptive data mining** involves using data and models to make recommendations or suggestions about actions or decisions. This type of data mining is often used to optimize processes, allocate resources, or make other decisions that can help organizations achieve their goals.

Benefits of Data Mining

Data mining is the process of extracting useful information and insights from large data sets. It is a powerful and flexible tool that has many benefits, including:

1. **Improved decision-making** – One of the main benefits of data mining is that it can help organizations make better decisions. By analyzing data and uncovering hidden patterns and trends, data mining can provide valuable insights and information that can be used to inform and improve decision-making.
2. **Increased efficiency and productivity** – Data mining can also help organizations increase their efficiency and productivity. By automating and streamlining the data analysis process, data mining can save time and resources, and help organizations work more effectively and efficiently.
3. **Reduced costs** – Data mining can also help organizations reduce their costs. By identifying and addressing inefficiencies and waste, data mining can help organizations save money and improve their bottom line.

4. **Increased customer satisfaction** – Data mining can also be used to improve customer satisfaction. By analyzing data on customer behavior and preferences, data mining can help organizations understand their customers better, and provide more personalized and relevant products and services.
5. **Improved risk management** – Data mining can also be used to improve risk management. By analyzing data on potential risks and vulnerabilities, data mining can help organizations identify and mitigate potential risks, and make more informed and strategic decisions.

UNIT 4

INTEGRATED SYSTEMS, SECURITY AND CONTROL

SYLLUBUS

Knowledge based decision support systems, Integrating social media and mobile technologies in Information system, Security, IS Vulnerability, Disaster Management, Computer Crimes, Securing the Web.

****

Knowledge-Based Decision Support Systems (KBDSS)

A Knowledge-Based Decision Support System (KBDSS) integrates human expertise, organizational knowledge, and computational tools to support complex decision-making processes.

Components of KBDSS:

- **Knowledge Base**:
  - Stores domain-specific knowledge, rules, and facts.
  - Sources: Expert systems, case studies, historical data.
- Inference Engine:
  - Processes the knowledge base to generate conclusions or recommendations.
  - Example: Logical reasoning or rule-based approaches.
- User Interface:
  - Facilitates interaction between users and the system.
- Data Management:
  - Integrates databases to retrieve necessary data for decision-making

Features of KBDSS

- **Integration of Knowledge and Data**:

  Combines domain knowledge with data analytics to enhance decision-making.

- Reasoning and Explanation:

Provides explanations for recommendations, enhancing trust in the system.

- Interactive and Adaptive:

  Learns and improves over time through feedback from users and updated data.

- Supports Complex Decisions:

  Useful in scenarios where traditional data analysis is insufficient.

Applications of KBDSS

- Medical Diagnosis**:**
  - Assists doctors by analyzing symptoms and recommending possible treatments.
  - Example: IBM Watson for Oncology.
- Financial Analysis:
  - Supports loan approvals, investment decisions, and fraud detection.
- Business Strategy:
  - Helps in strategic planning by analyzing market trends and predicting future outcomes.
- Supply Chain Management:
  - Optimizes inventory, procurement, and logistics by using past knowledge.
- Customer Support:
  - Chatbots with built-in KBDSS offer solutions based on a knowledge repository.

Advantages of KBDSS

- Improved Decision Accuracy**:**
  - Reduces human error by using expert knowledge and reasoning**.**
- Efficiency and Speed:

- Quickly processes vast amounts of data and knowledge for rapid decision-making.

- Consistency:
  - Ensures uniformity in decision-making across the organization.
- Learning Capability:
  - Adapts and evolves through machine learning and user feedback.
- Handles Complex Problems:
  - Suitable for unstructured and semi-structured decision-making scenarios

Technologies Used in KBDSS

- **Artificial Intelligence (AI)**:
  - Machine learning and neural networks for predictive analytics and reasoning.
- Natural Language Processing (NLP):
  - Allows systems to interpret and respond to user queries in natural language.
- Fuzzy Logic:
  - Handles uncertainty and imprecise data in decision-making.
- Big Data Analytics:
  - Processes large volumes of structured and unstructured data.
- Expert Systems:
  - Mimics the decision-making ability of human experts using rules and heuristics.

Examples of KBDSS in Action

- **SAP Leonardo**:
  - Combines AI, machine learning, and analytics for intelligent decision-making.

- IBM Watson:
  - Processes medical and business data to assist in diagnosis and strategic decisions.
- MYCIN:
  - An early expert system for diagnosing bacterial infections and recommending antibiotics.
- FICO Decision Management Suite:
  - Used for financial decisions, fraud detection, and credit scoring.

Integrating Social Media and Mobile Technologies in Information Systems

Introduction

The integration of social media and mobile technologies in Information Systems (IS) has transformed the way businesses and organizations interact with stakeholders. By leveraging these tools, organizations can enhance communication, data collection, and decision-making, while also improving customer engagement and operational efficiency.

Significance of Integration

- **Real-Time Communication**:
  - Social media and mobile platforms enable instant interaction with customers, employees, and other stakeholders.
  - Example: Live chats, social media comments, and push notifications.
- Enhanced Data Collection:
  - Social media generates vast amounts of user data, including preferences, feedback, and behaviors.
  - Mobile devices provide real-time geolocation and activity data.
- Customer Engagement:
  - Organizations can use social media and mobile apps to create personalized experiences for users.
- Wider Reach:
  - These technologies enable businesses to reach global audiences effectively.

Applications of Social Media and Mobile Technologies in Information Systems

1. Marketing and Branding

- Social media platforms (e.g., Facebook, Instagram, Twitter) are used for targeted advertising and brand promotion.
- Mobile apps enhance brand visibility through notifications and location-based marketing.
- Example: A retail chain uses Instagram ads and a mobile app to notify users of nearby discounts.

2. Customer Relationship Management (CRM)

- Social media provides insights into customer preferences and feedback, enabling better relationship management.
- Mobile apps offer loyalty programs, feedback collection, and personalized communication.
- Example: Starbucks Rewards App uses mobile technology to engage customers with personalized offers.

3. Data Analytics

- Social media platforms serve as sources for sentiment analysis and trend prediction.
- Mobile technologies track customer journeys and behaviors for data-driven insights.
- Example: Analyzing Twitter hashtags to predict customer sentiment about a product launch.

4. Employee Collaboration and Productivity

- Mobile apps like Slack and Microsoft Teams integrate with social media for seamless communication.
- Employees can use mobile devices for remote work and collaboration.
- Example: A multinational company uses Microsoft Teams to share updates and conduct virtual meetings.

5. Supply Chain Management

- Mobile apps track shipments in real time, while social media platforms provide updates about delays or disruptions.
- Example: FedEx integrates mobile tracking systems and social media alerts for customer transparency.

6. Public Relations and Crisis Management

- Social media is critical for addressing public concerns and maintaining a positive brand image during crises.

- Mobile alerts and notifications help manage crisis communication effectively.
- Example: Airlines using Twitter to inform passengers about flight delays.

Security in Information Systems

Information System Security ensures the protection of data, software, hardware, and networks from unauthorized access, theft, and damage.

Types of Security Measures:

1. **Authentication**:
   - Methods: Passwords, biometrics, two-factor authentication (2FA).
2. Authorization:
   - Grants access rights based on user roles.
3. Encryption:
   - Secures data in transit and at rest by converting it into unreadable formats.
   - Example: AES, RSA encryption.
4. Firewalls and Intrusion Detection Systems (IDS):
   - Monitors and controls incoming and outgoing traffic.
5. Backup and Recovery:
   - Ensures data availability during system failures.

IS Vulnerability

Vulnerabilities are weaknesses in an information system that can be exploited by threats to cause harm.

Common Vulnerabilities:

1. **Human Errors**:
   - Weak passwords, accidental data deletion.
2. Software Vulnerabilities:
   - Bugs, outdated software, and unpatched systems.
3. Physical Vulnerabilities:
   - Lack of physical security for devices and servers.
4. Network Vulnerabilities:
   - Unsecured Wi-Fi, DDoS attacks.

Mitigation Strategies:

- Regular updates and patching.
- Employee training on cybersecurity practices.

- Implementing strong network security protocols

Disaster Management in Information Systems

Disaster Management involves planning, preparing, and recovering from incidents that disrupt information systems.

Key Phases:

1. **Prevention**:
   - Risk assessment and mitigation strategies.
2. Preparedness:
   - Developing a disaster recovery plan (DRP).
   - Regularly testing backups.
3. Response:
   - Immediate steps to minimize damage, such as switching to backup systems.
4. Recovery:
   - Restoring normal operations.

Examples of Disasters:

- Natural: Earthquakes, floods, and fires.
- Human-made: Cyberattacks, hardware failure, and power outages.

Tools:

- Cloud-based backups.
- Redundant data centers.
- Incident response software.

Computer Crimes

Illegal activities involving the use of computers or networks to commit crimes.

Types of Computer Crimes:

1. **Hacking**:
   - Unauthorized access to systems or data.
2. Phishing:
   - Fraudulent attempts to obtain sensitive information through fake emails or websites.
3. Ransomware:
   - Encrypting a victim's data and demanding payment for its release.

4. Identity Theft:
    - Stealing personal information for financial or criminal purposes.
5. Distributed Denial of Service (DDoS):
    - Overloading a system to make it unavailable.

Prevention:

- Use of antivirus and antimalware software.
- Implementing strict access controls.
- Educating users about common threats.

Securing the Web

Web security involves safeguarding websites and web applications from cyberattacks.

Common Web Security Threats:

1. **Cross-Site Scripting (XSS)**:
    - Injecting malicious scripts into websites.
2. SQL Injection:
    - Exploiting vulnerabilities in database queries.
3. Man-in-the-Middle (MITM) Attacks:
    - Intercepting data during transmission.
4. Website Defacement:
    - Altering website content by hackers.

Best Practices for Web Security:

1. **SSL/TLS Encryption**:
    - Encrypts data between users and servers.
2. Web Application Firewalls (WAF):
    - Protects applications from common vulnerabilities.
3. Regular Security Audits:
    - Identifies and addresses vulnerabilities.
4. Use of Secure Coding Practices:
    - Following frameworks like OWASP.

Tools:

- Vulnerability scanners (e.g., Nessus).
- Intrusion prevention systems.

UNIT 5
EMERGING INFORMATION TECHNOLOGY

****

DEEP LEARNIG

Deep learning is a subset of machine learning that uses artificial neural networks with multiple layers (deep networks) to model and learn from complex data representations. It mimics the structure and functioning of the human brain to solve tasks like image recognition, natural language processing, and speech recognition.

What is Deep Learning?

The definition of Deep learning is that it is the branch of machine learning that is based on artificial neural network architecture. An artificial neural network or ANN uses layers of interconnected nodes called neurons that work together to process and learn from the input data. In a fully connected Deep neural network, there is an input layer and one or more hidden layers connected one after the other. Each neuron receives input from the previous layer neurons or the input layer. The output of one neuron becomes the input to other neurons in the next layer of the network, and this process continues until the final layer produces the output of the network. The layers of the neural network transform the input data through a series of nonlinear transformations, allowing the network to learn complex representations of the input data.

Key Characteristics of Deep Learning

1. **Multi-layered Structure**:
   - Neural networks with many hidden layers for feature extraction and representation.
2. Automatic Feature Extraction:
   - Eliminates the need for manual feature engineering by learning directly from raw data.
3. Large Data Requirements:
   - Performs best with vast amounts of labeled data.
4. High Computational Power:
   - Requires GPUs or TPUs for efficient training due to complex computations

Applications of Deep Learning

1. **Image Recognition**:
   - Examples: Face recognition, medical imaging (e.g., detecting tumors).

2. Natural Language Processing (NLP):
    - Examples: Sentiment analysis, machine translation (Google Translate), chatbots.
3. Speech Recognition:
    - Example: Virtual assistants like Siri, Alexa.
4. Autonomous Vehicles:
    - Used for object detection, path planning, and decision-making.
5. Healthcare:
    - Predicting diseases, drug discovery, and personalized treatment.
6. Finance:
    - Fraud detection, stock price prediction.

Advantages of Deep Learning:

1. **High accuracy:** Deep Learning algorithms can achieve state-of-the-art performance in various tasks, such as image recognition and natural language processing.
2. **Automated feature engineering:** Deep Learning algorithms can automatically discover and learn relevant features from data without the need for manual feature engineering.
3. **Scalability:** Deep Learning models can scale to handle large and complex datasets, and can learn from massive amounts of data.
4. **Flexibility:** Deep Learning models can be applied to a wide range of tasks and can handle various types of data, such as images, text, and speech.
5. **Continual improvement:** Deep Learning models can continually improve their performance as more data becomes available.

Big Data

Big Data refers to large, complex datasets that cannot be managed, processed, or analyzed using traditional tools or methods.

Characteristics (5 Vs of Big Data):

- **Volume**: Huge amount of data.
- **Variety**: Different types of data (structured, semi-structured, unstructured).
- **Velocity**: Speed of data generation and processing.
- **Veracity**: Accuracy and trustworthiness of data.
- **Value**: Transforming data into meaningful insights.

Sources of Big Data

- **Social Media**: Facebook, Twitter, Instagram data.

- **Sensor Data**: IoT devices, wearables.
- **Transaction Data**: Business transactions, e-commerce.
- **Web Logs**: Data from websites and online interactions.
- **Machine Data**: Generated by machines and applications.
- **Healthcare**: Medical records, test reports.

Types of Data in Big Data

- **Structured Data**: Organized in rows/columns, e.g., relational databases.
- **Semi-Structured Data**: JSON, XML files with some organization.
- **Unstructured Data**: Texts, videos, audio, emails, etc.

Big Data Technologies

Key tools and frameworks for managing Big Data:

1. Hadoop Ecosystem
    - **HDFS (Hadoop Distributed File System)**: Storage system.
    - **MapReduce**: Processing data in parallel.
    - **YARN**: Resource management.
    - **Hive**: SQL-like querying tool.
    - **Pig**: High-level scripting for data flow.
2. Apache Spark
    - Fast in-memory processing.
    - Supports batch, real-time, and streaming data.
3. NoSQL Databases
    - MongoDB, Cassandra, HBase for unstructured and semi-structured data.
4. Data Processing Tools
    - **Apache Kafka**: Real-time streaming.
    - **Apache Flink**: Stream and batch processing.
5. Cloud Platforms
    - AWS, Microsoft Azure, Google Cloud Platform (GCP).

Applications of Big Data

- **Business Analytics**: Analyzing sales data for trends and insights.
- **Healthcare**: Predicting diseases and patient outcomes.
- **Finance**: Fraud detection and risk management.
- **Retail**: Customer segmentation and personalized marketing.
- **Manufacturing**: Predictive maintenance of equipment.
- **Smart Cities**: Traffic and energy management.

Challenges in Big Data

- **Storage and Management**: Handling huge datasets.
- **Processing Speed**: Analyzing real-time data efficiently.
- **Data Quality**: Ensuring accurate, clean data.
- **Security and Privacy**: Protecting sensitive data.
- **Integration**: Combining data from multiple sources.

Role of Big Data in Business

- Supports decision-making with real-time insights.
- Enhances customer experience through personalized strategies.
- Optimizes operations and reduces costs.
- Improves product development using predictive analytics.

Pervasive Computing

Pervasive computing (also called **ubiquitous computing**) refers to the integration of computing capabilities into everyday objects and activities, enabling seamless communication and functionality. The goal is to make computing **invisible**, **integrated**, and **intuitive** in daily life.

Ubiquitous computing (pervasive) is a concept in software engineering and computer science where computing is made to appear everywhere and anywhere. In contrast to desktop computing, ubiquitous computing can occur using any device, in any location, and in any format. A user interacts with the computer, which can exist in many different forms, including laptop computers, tablets and terminals in everyday objects such as a fridge or a pair of glasses. The underlying technologies to support ubiquitous computing include Internet, advanced middleware, operating system, mobile code, sensors, microprocessors, new I/O and user interfaces, networks, mobile protocols, location and positioning and new materials. This new paradigm is also described as pervasive computing, ambient intelligence, or 'every ware'. Each term emphasizes slightly different aspects. When primarily concerning the objects involved, it is also known as physical computing, the Internet of Things, hap tic computing, and 'things that think'. Rather than

propose a single definition for ubiquitous computing and for these related terms, taxonomy of properties for ubiquitous computing has been proposed, from which different kinds or flavors of ubiquitous systems and applications can be described.

Ubiquitous computing touches on a wide range of research topics, including distributed computing, mobile computing, location computing, mobile networking, context-aware computing, sensor networks, human-computer interaction, and artificial intelligence.

Features of Pervasive Computing

- **Invisibility**: Computing becomes part of the environment.
- **Mobility**: Access computing resources anytime, anywhere.
- **Context Awareness**: Systems adapt to the user's environment (e.g., location, time).
- **Interconnectivity**: Devices communicate and share data seamlessly.
- **Heterogeneity**: Supports a variety of devices, sensors, and networks.

Evolution of Pervasive Computing

- **1st Generation**: Mainframes (large centralized computers).
- **2nd Generation**: Personal Computers (PCs).
- **3rd Generation**: Mobile computing (laptops, smartphones).
- **4th Generation**: Pervasive Computing (smart devices, IoT).

Components of Pervasive Computing

1. **Devices**:
    - Smartphones, wearables, smart home appliances, embedded systems.
2. Networks:
    - **Wireless Communication**: Wi-Fi, Bluetooth, 5G.
    - **IoT (Internet of Things)**: Connectivity between physical devices.
3. Sensors and Actuators:
    - Sensors collect real-time data (e.g., temperature, motion).
    - Actuators perform actions (e.g., adjusting thermostats, locking doors).
4. Software:
    - Embedded software for device operation.
    - Cloud platforms for data storage and analysis.

Applications of Pervasive Computing

1. **Smart Homes**:
   - IoT devices control lighting, security, appliances, and energy.
2. Healthcare:
   - Wearable devices (e.g., fitness trackers, smartwatches) monitor health parameters.
   - Telemedicine for remote patient monitoring.
3. Retail:
   - Personalized shopping experiences using sensors and mobile apps.
   - Smart shelves and automated billing systems.
4. Transportation:
   - Smart cars (GPS, self-driving systems).
   - Traffic monitoring and smart parking systems.
5. Workplace Automation:
   - Smart devices for seamless communication and task automation.
6. Education:
   - Smart classrooms, e-learning platforms, and virtual reality tools.
7. Urban Development:
   - Smart cities use pervasive computing for energy management, public safety, and transportation.

Advantages of Pervasive Computing

- **Convenience**: Simplifies daily activities through automation.
- **Improved Decision-Making**: Real-time data collection for accurate decisions.
- **Enhanced Communication**: Devices seamlessly interact with users and systems.
- **Efficiency**: Optimizes resources, energy, and processes.

Challenges of Pervasive Computing

- **Privacy and Security**: Protecting sensitive data across interconnected devices.

- **Interoperability**: Ensuring devices and systems work together seamlessly.
- **Energy Consumption**: Powering smart devices continuously.
- **Data Overload**: Managing and analyzing massive amounts of data.
- **Cost**: High initial investment for smart systems and infrastructure.

Role in Business and Society

- **Enhanced Customer Experience**: Businesses can provide **personalized services** using pervasive computing tools.
- **Operational Efficiency**: Smart devices streamline processes and **reduce costs**.
- **Data-Driven Decisions**: Real-time data improves decision-making.
- **Innovation**: Encourages new business models and products (e.g., smart devices, IoT- based solutions).

Cloud Computing

Cloud computing is the delivery of computing services like servers, storage, databases, networking, software, and analytics over the Internet ("the cloud") to offer faster innovation, flexible resources, and economies of scale. Instead of owning and maintaining physical hardware, businesses rent computing resources **as needed.**

Characteristics of Cloud Computing

1. **On-Demand Self-Service**: Users can access resources anytime without human intervention.
2. **Broad Network Access**: Services are available over the internet on various devices.
3. **Resource Pooling**: Computing resources are shared among multiple users.
4. **Rapid Elasticity**: Resources can scale up or down based on demand.
5. **Measured Service**: Pay-as-you-go model based on usage.

Types of Cloud Computing Services

1. **Infrastructure as a Service (IaaS):**
    - Provides **virtualized hardware resources** like servers, storage, and networks.
    - Example: Amazon EC2, Microsoft Azure Virtual Machines.
2. Platform as a Service (PaaS):
    - Offers a platform for developing, testing, and managing applications.

- Developers can focus on code while infrastructure is managed.
- Example: Google App Engine, Heroku.

3. Software as a Service (SaaS):
   - Provides software applications over the internet on a subscription basis.
   - Example: Google Workspace, Microsoft Office 365, Salesforce.

Deployment Models of Cloud Computing

1. **Public Cloud**:
   - Resources are shared and managed by third-party cloud providers.
   - Example: AWS, Microsoft Azure, Google Cloud.
2. Private Cloud:
   - Dedicated to a single organization, either on-site or hosted.
   - Provides more control and security.
3. Hybrid Cloud:
   - Combines public and private clouds to offer flexibility.
4. Community Cloud:
   - Shared infrastructure for a specific group or industry.

Advantages of Cloud Computing

1. **Cost-Effective**: Reduces capital expenses for hardware and software.
2. **Scalability**: Quickly scales resources up or down.
3. **Accessibility**: Access data and applications from anywhere with an internet connection.
4. **Disaster Recovery**: Provides backup and recovery solutions.
5. **Improved Collaboration**: Teams can collaborate in real time using cloud applications.
6. **Security**: Leading providers offer robust security measures.

Challenges in Cloud Computing

1. **Data Privacy and Security**: Risk of unauthorized access and data breaches.
2. **Downtime**: Cloud services may experience outages.
3. **Vendor Lock-in**: Difficulty in migrating data between providers.
4. **Limited Control**: Users rely on third-party providers for infrastructure.
5. **Compliance**: Organizations must meet regulatory requirements for data.

UNIT 5

UNIT -VI
MANAGERIAL APPLICATION AND STRATEGIC USE

Applications of Cloud Computing

1. **Business**:
    - Hosting websites, data analytics, and enterprise software (ERP, CRM).
2. Education:
    - E-learning platforms, virtual classrooms, and resource sharing.
3. Healthcare:
    - Telemedicine, patient record management, and predictive diagnostics.
4. Banking and Finance:
    - Fraud detection, real-time transaction processing, and risk management.
5. Entertainment:
    - Streaming platforms (Netflix, Amazon Prime), gaming services.
6. Startups:
    - Cost-effective infrastructure to develop and scale applications.

Major Cloud Providers

- **Amazon Web Services (AWS)**: Leading cloud provider with a broad range of services.
- **Microsoft Azure**: Strong in hybrid cloud and enterprise integrations.
- **Google Cloud Platform (GCP)**: Known for AI/ML and data analytics.
- **IBM Cloud**: Focuses on AI and enterprise solutions.
- **Oracle Cloud**: Offers database and business applications.

Cloud Computing in Business

- **Cost Reduction**: Eliminates hardware purchases and maintenance costs.
- **Faster Innovation**: Reduces time-to-market for new products.
- **Global Reach**: Supports global operations with scalable infrastructure.
- **Data-Driven Decisions**: Cloud analytics enable businesses to analyze large datasets.
- **Competitive Advantage**: Enables businesses to focus on core operations while cloud providers manage IT.

Internet of Things (IoT)

A network of interconnected devices that communicate and exchange data over the internet without requiring human intervention.

Key Components

1. **Devices/Sensors**: Collect data from the environment (e.g., temperature, motion).
2. **Connectivity**: Networks like Wi-Fi, Bluetooth, Zigbee, and cellular enable data transmission.
3. **Data Processing**: Cloud or edge computing analyzes and processes the collected data.
4. **User Interface**: Applications or dashboards for users to interact with the IoT system.

Characteristics

- **Interconnectivity**: Devices are connected to share data.
- **Automation**: Enables tasks to be performed with minimal human intervention.
- **Real-time Monitoring**: Provides insights and updates instantaneously.
- **Scalability**: Supports the addition of new devices to the network.

Applications

1. **Smart Homes**: Home automation with smart thermostats, lights, and appliances.
2. **Healthcare**: Remote patient monitoring, wearable health trackers.
3. **Industrial IoT (IIoT)**: Predictive maintenance, supply chain management.
4. **Smart Cities**: Traffic management, waste management, and environmental monitoring.
5. **Agriculture**: Smart irrigation, soil monitoring, and crop management.

Challenges

- **Security**: Vulnerability to cyber-attacks and data breaches.
- **Interoperability**: Difficulty in integrating devices from different manufacturers.
- **Data Privacy**: Ensuring user data remains confidential.
- **Power Consumption**: Limited battery life of IoT devices.

Benefits

- **Efficiency**: Streamlines processes and saves resources.
- **Convenience**: Enhances user experience with automation.

- **Cost Savings**: Reduces maintenance and operational costs.
- **Improved Decision-Making**: Real-time data enables proactive decisions.

Block chain

A **block chain** is a decentralized, distributed ledger that records transactions across multiple computers in a secure, immutable, and transparent manner.

Key Characteristics

1. **Decentralized**: No central authority; the network is maintained by nodes.
2. **Immutable**: Data once written cannot be altered or deleted.
3. **Transparent**: Transactions are visible to all participants in the network.
4. **Secure**: Uses cryptographic methods to ensure data integrity and security.

Components

1. **Block**:
   - Contains transaction data, a timestamp, and a cryptographic hash of the previous block.
2. Chain:
   - A series of connected blocks linked by cryptographic hashes.
3. Nodes:
   - Devices participating in the blockchain network, maintaining copies of the ledger.
4. Consensus Mechanism:
   - Protocols to validate transactions and add them to the blockchain (e.g., Proof of Work, Proof of Stake)

Types of Block chain

1. Public Block chain: Open to everyone (e.g., Bitcoin, Ethereum).
2. Private Block chain: Controlled by a single organization.
3. Consortium Block chain: Controlled by a group of organizations.
4. Hybrid Block chain: Combines features of public and private block chains.

Applications

1. Cryptocurrencies: Bitcoin, Ethereum, and other digital currencies.
2. Supply Chain Management: Tracks product origin and movement.
3. Smart Contracts: Self-executing contracts with predefined rules.
4. Healthcare: Secure storage and sharing of medical records.
5. Voting Systems: Tamper-proof digital voting.
6. Finance: Cross-border payments, fraud prevention, and identity verification.

Advantages

- Enhanced security and privacy.
- Reduced costs through automation (e.g., smart contracts).
- Greater transparency and traceability.
- Decentralization eliminates single points of failure.

Challenges

1. Scalability: Slow transaction speeds and high costs for large networks.
2. Energy Consumption: High energy usage in consensus mechanisms like Proof of Work.
3. Regulatory Uncertainty: Lack of uniform laws governing blockchain use.
4. Interoperability: Difficulty integrating with existing systems or other blockchains.

Cryptocurrency

Cryptocurrency is a digital or virtual form of currency that uses cryptography for security. It operates on decentralized networks based on block chain technology, making it independent of central authorities like banks.

Key Features

1. **Decentralized**:
   - No central authority controls the currency.
2. Block chain Technology:
   - Ensures transparency, security, and immutability of transactions.
3. Cryptographic Security:
   - Protects transactions and controls the creation of new units.
4. Peer-to-Peer Transactions:
5. Direct exchange between users without intermediaries.Global Accessibility:
   - Can be used anywhere with an internet connection.

Popular Cryptocurrencies

1. **Bitcoin (BTC)**:
   - The first cryptocurrency, created in 2009.

2. Ethereum (ETH):
    - Known for enabling smart contracts and decentralized applications (DApps).
3. Binance Coin (BNB):
    - Used primarily on the Binance Exchange.
4. Ripple (XRP):
    - Focuses on cross-border payment solutions.
5. Litecoin (LTC):
    - A faster alternative to Bitcoin.
6. Cardano (ADA) and Solana (SOL):
    - Popular for their advanced blockchain technologies.

Applications

1. **Digital Payments**:
    - Peer-to-peer payments and online purchases.
2. Smart Contracts:
    - Self-executing contracts with predefined rules.
3. Decentralized Finance (DeFi):
    - Financial services like lending, borrowing, and trading without intermediaries.
4. Tokenization:
    - Representing physical or digital assets (e.g., NFTs).
5. Cross-Border Transactions:
    - Faster and cheaper international money transfers.

Advantages

1. **Transparency**:
    - Transactions are visible on the blockchain.
2. Security:
    - Advanced encryption protects user data.
3. Speed:
    - Faster transactions compared to traditional banking.

4. Low Fees:
    - Minimal transaction costs.
5. Decentralization:
    - No reliance on central authorities.

Challenges

1. **Volatility**:
    - Prices can fluctuate significantly in short periods.
2. Regulatory Issues:
    - Uncertain legal frameworks in many countries.
3. Security Risks:
    - Susceptible to hacking and scams.
    -
4. Scalability:
    - Limited transaction speed and capacity of many block chain networks.
5. Environmental Impact:
    - High energy consumption, especially in mining.

Quantum Computing

Quantum Computing utilizes the principles of quantum mechanics to process information. Unlike classical computers, which use bits as binary 0s and 1s, quantum computers use **quantum bits (qubits)** that can represent and process multiple states simultaneously due to superposition and entanglement.

Key Principles

1. **Superposition**:
    - A qubit can exist in multiple states (both 0 and 1) simultaneously, enhancing computational power.
2. Entanglement:
    - Qubits can be correlated in such a way that the state of one directly affects the state of another, even over large distances.

3. Quantum Interference:
    - Quantum states can interfere constructively or destructively, which is used to refine computation outcomes.

Components of a Quantum Computer

1. **Qubits**:
    - The basic units of quantum information, implemented using physical systems like photons, ions, or superconducting circuits.
2. Quantum Gates:
    - Perform operations on qubits, similar to logic gates in classical computing.
3. Quantum Processor:
    - Executes quantum operations and algorithms.
4. Control System:
    - Manages inputs and outputs for quantum operations.
5. Error Correction:
    - Addresses errors caused by quantum decoherence and noise.

Applications

1. **Cryptography**:
    - Quantum computers can potentially break current cryptographic methods (e.g., RSA) but also enable quantum-safe cryptography.
2. Optimization:
    - Solve complex problems in logistics, finance, and scheduling.
3. Drug Discovery:
    - Simulate molecular structures and interactions at quantum levels.
4. Artificial Intelligence:
    - Accelerate machine learning and pattern recognition.
5. Material Science:
    - Design new materials with desired properties by simulating atomic interactions.
6. Climate Modeling:
    - Improve simulations for weather and environmental predictions.

Benefits of Quantum Computing

1. Exponential Speed

- **Parallelism**: Quantum computers can process multiple calculations simultaneously due to superposition, enabling faster problem-solving compared to classical computers.
- **Efficiency**: Certain tasks, like factoring large numbers or searching databases, are exponentially faster with quantum algorithms (e.g., Shor's and Grover's).

2. Advanced Problem Solving

- **Optimization Problems**:
  - Efficiently solve complex problems in logistics, scheduling, and supply chain management.
- Simulation of Quantum Systems:
  - Simulate molecular interactions for drug discovery and material design, which is impractical for classical computers.
- AI and Machine Learning:
  - Accelerate training of machine learning models and improve pattern recognition.

3. Enhanced Cryptography

- **Breaking Classical Encryption**:
  - Quantum computers can break widely-used encryption methods (e.g., RSA), leading to advancements in quantum-safe cryptography.
  -
- Quantum Cryptography:
  - Enable unbreakable communication using quantum key distribution (QKD).

4. Innovation in Science and Technology

- **Drug Discovery**:
  - Quantum simulations can identify new drug molecules and interactions at atomic levels, reducing R&D time.
- Material Science:
  - Design new materials with specific properties, advancing fields like renewable energy and electronics.

5. Climate and Environmental Applications

- **Climate Modeling**:
  - More accurate simulations of weather and climate systems for better environmental predictions.
- Energy Optimization:
  - Improve energy grids, optimize renewable energy use, and develop sustainable solutions.

DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS YEAR 2025-2027 SEM : FIRST
COURSE CODE: INFORMATION MANAGEMENT

PREVIOUS YEAR QUESTION PAPERS

Reg. No. : | | | | | | | | | | | | |

Question Paper Code : 80190

M.B.A. DEGREE EXAMINATIONS, APRIL/MAY 2025

First Semester

BA 4106 – INFORMATION MANAGEMENT

(Common to: Master of Business Administration (Artificial Intelligence and Data Science))

(Regulations 2021)

Time : Three hours Maximum : 100 marks

Answer ALL questions.

PART A — (10 × 2 = 20 marks)

1. What is the difference between data and information in the context of information management?

2. What are the main types of information systems based on their functions?

3. What is a Data Flow Diagram (DFD)?

4. What is the significance of Entity Relationship (ER) diagrams in database design?

5. What distinguishes a Relational Database Management System (RDBMS) from an Object-Oriented Database Management System (OODBMS)?

6. What are the main components of data warehousing?

7. What are common vulnerabilities associated with information systems?

8. Why is disaster management important in the context of information security?

9. What role does cloud computing play in modern information management?

10. What is Quantum computing?

PART B — (5 × 13 = 65 marks)

11. (a) Discuss the evolution of information systems and their impact on business operations.

Or

(b) Discuss the role information systems play in decision-making processes and provide recommendations for organizations looking to implement or upgrade their information systems for better performance.

12. (a) Analyse various system development methodologies and their relevance in today's technology landscape.

Or

(b) Explain how the process of systems analysis and design can be applied to develop effective information systems that meet specific organizational needs and enhance operational efficiency. Discuss the significance of each phase in ensuring that the system aligns with business objectives and user requirements.

13. (a) Compare and contrast between various types of DBMS.

Or

(b) Narrate the process and application of data mining.

14. (a) Evaluate the challenges and solutions related to security and control in information management.

Or

(b) Evaluate the role of knowledge-based decision support systems in improving organizational decision-making.

15. (a) Discuss the potential of block chain technology and cryptocurrency in transforming information management practices.

Or

(b) Explain the importance of big data analytics in strategic decision-making processes.

PART C — (1 × 15 = 15 marks)

16. (a) You are leading a project to develop a new information system for a retail company. Describe the steps you would take in the system analysis and design phase. Include methodologies, tools (DFDs and ER diagrams), and considerations for user requirements.

Or

(b) A financial institution has faced multiple data breaches due to inadequate security measures. Develop a comprehensive plan to enhance the security of its information systems. Address the integration of decision support systems, risk management, and disaster recovery strategies.

Reg. No. : | | | | | | | | | | | | |

Question Paper Code : 20196

M.B.A. DEGREE EXAMINATIONS, NOVEMBER/DECEMBER 2024.

First Semester

BA 4106 – INFORMATION MANAGEMENT

(Common to: Master of Business Administration (Artificial Intelligence and Data Science))

(Regulations 2021)

Time : Three hours | Maximum : 100 marks

Answer ALL questions.

PART A — (10 × 2 = 20 marks)

1. What is data?
2. What is Information?
3. What is data flow diagram?
4. What is UML diagram?
5. What is database? Give an example.
6. What is data mart?
7. What is an information system?
8. Brief on computer crimes.
9. What is big data?
10. What is Quantum computing?

PART B — (5 × 13 = 65 marks)

11. (a) Discuss the evolution of information system.

Or

(b) Explain the various functional information systems.

12. (a) Write short notes on

(i) Decision table (6)

(ii) Entity Relationship (7)

Or

(b) Explain object oriented Analysis and Design with examples.

13. (a) Discuss the evolution of Data Base Management Systems.

Or

(b) Write short notes on

(i) Data Warehousing (7)

(ii) Data Mining (6)

14. (a) How will you integrate social media and mobile technologies in Information system?

Or

(b) Discuss the techniques to secure web. (7+6)

15. (a) Write short notes on

(i) Deep Learning (7)

(ii) Cloud Computing (6)

Or

(b) Write short notes on

(i) Artificial Intelligence (7)

(ii) Crypto Currency (6)

PART C — (1 × 15 = 15 marks)

16. (a) Choose a system of your choice and design and develop it, Draw suitable data flow diagram, decision tables and Entity Relationship.

Or

(b) Develop an information system that will defect crimes in the computer and protect the computer.

———————

Reg. No. : | | | | | | | | | | | | |

Question Paper Code : 60191

M.B.A. DEGREE EXAMINATIONS, NOVEMBER/DECEMBER 2023.

First Semester

BA 4106 — INFORMATION MANAGEMENT

(Regulations 2021)

Time : Three hours Maximum : 100 marks

Answer ALL questions.

PART A — (10 × 2 = 20 marks)

1. What is an Information System?
2. What is data?
3. What is a system?
4. What is decision table?
5. What is a data base?
6. What is data mining?
7. What are hackers?
8. What is description?
9. What is cloud computing?
10. What is big data?

PART B — (5 × 13 = 65 marks)

11. (a) What is information? What are its features?

Or

(b) Outline the features of Financial Information System.

12. (a) What is system development life cycle? Explain its phases. (3+10)

Or

(b) What is E.R. diagram? Explain the methodology of E.R. diagrams. (3+10)

13. (a) What is data warehousing? What are its characteristics? (3+10)

Or

(b) Explain the steps in database design. (13)

14. (a) Discuss the functions of information security administrater to protest the network from an authorized access. (13)

Or

(b) What is a firewall? What are faster filtering firewall and proxy – server firewall? (3+5+5)

15. (a) Write short notes on:

(i) Deep learning (6)

(ii) Pervasive computing (7)

Or

(b) Write short notes on:

(i) Advantages and disadvantages of Internet of things (7)

(ii) Crypto Currency (6)

PART C — (1 × 15 = 15 marks)

16. (a) Design develop Hospital billing system E.R. diagrams.

Or

(b) Enumerate on role of information management in any two sector.

———————