LINEAR ALGEBRA AND OPTIMIZATION
30242201

COURSE OBJECTIVES

- apply matrix methods to solve systems of linear equations.
- learn algebraic structures and apply vector space concepts to analyse vector relationships.
- formulate real-life linear programming models and solve them using optimization techniques.
- develop optimization models for transportation and assignment problems to solve techniques.
- understand sequencing and queuing principles and its applications**.**

Unit I:

Linear Algebra-I: Matrix, Rank of Matrix, Echelon Form, Normal Form of Matrix, Solution of Simultaneous Equation by Elementary Transformation, Consistency of Equation, Eigen Values and Eigenvectors, Normalized Eigenvector, Cayley Hamilton Theorem and its Application to Finding Inverse of Matrix.

Unit II:

Linear Algebra-II: Introduction to Groups and Field, Vector Spaces Over the Field, Linear Dependent Vectors and Linear Independent Vectors, Linear Combination of Vectors, Linear Span of a Set of Vectors, Basis and Dimension of a Vector Space.

Unit III:

Linear Programming: Application and Model Formulation: Introduction of OR, LPP Formulation, Solution by Graphical Method, Optimal solution, Unbounded Solution, Infeasible Solution, Simplex Method, Basic Solution, Surplus and Slack Variables, Two Phase Method, Big- M method, Duality of LPP.

Unit IV:

Transportation and Assignment Problem: Mathematical Model of Transportation Problem, North-West Corner method, Least Cost Method, Vogel's Approximation Method, Test for Optimality, Dual of Transportation Method, Modi Method, Mathematical Model of Assignment Problem, Hungarian Method for Solving Assignment Problem, Travelling Salesman Problem.

Unit V: (Dynamic Content)

Sequencing & Queuing Problem: Processing n Jobs through Two Machines, Johnson's Rule of Sequencing, Processing n Jobs through Three Machines, Queuing

Theory: Characteristics of M/M/1 Queue Model, Applications of Poisson and Exponential Distributions in Estimating Arrival Rate and Service Rate.

RECOMMENDED BOOKS:

1. S. Lipschutz and M. Lipson, Linear Algebra (4th Edition), Schaum's Outline series, Mc-Graw Hill. (2009).
2. S. Boyd and L. Vandenberghe, Introduction to Applied Linear Algebra Vectors, Matrices, and Least Squares, University Printing House, Cambridge CB2 8BS, United Kingdom One Liberty Plaza, 20thFloor, New York, NY10006, USA, (2018).
3. H. A. Taha: Operations Research an Introduction, Pearson, 9th Edition (2014).
4. Sharma J K: Operations Research (Pearson, 6th Edition)
5. S Kalawathy: Operation Research (Vikas IV Edition) Natarajan- Operation Research (Pearson)
6. Singh & Kumar: Operation Research (UDH Publisher edition 2013)
7. Taha Hamdy - Operations Research - An Introduction (Prentice-Hall, 9th edition

---

COURSE OUTCOMES

After completion of this course, the students would be able to:

CO1. apply matrix techniques to solve systems of linear equations

CO2. analyze vector spaces to determining solution of the various engineering problem using algebraic structure concepts

CO3. formulate linear programming problems and obtain optimal solutions using various optimization methods

CO4. evaluate and test transportation solutions for optimality

CO5. apply sequencing and queuing techniques to solve various engineering problems.

CO-PO Mapping Matrix														
	PO1	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**	**PSO1**	**PSO2**
CO1	3	2	1	1	2					1		2	2	1
CO2	3	3	1	2	2					2		2	2	2
CO3	3	3	2	2	2	1	1	1		2		2	3	3
CO4	3	3	2	2	2	1	1		1	3	1	2	3	3
CO5	3	2	3	2	2	1	1		1	2	1	2	3	2

DATABASE MANAGEMENT SYSTEM
30242202

COURSE OBJECTIVES

- To understand the different issues involved in the design and implementation of a database system.
- To study the physical and logical database designs, database modelling, relational, hierarchical and network models.
- To understand and use data manipulation language to query, update and manage a database.

Unit I:

DBMS: Database Approach v/s Traditional File Approach, Advantages of Database System, Database Users and Administrator, Database System Environment, Application Architectures, Schemas, Instances, Data Independence, Data Models: Hierarchical Data Model, Network Data Model & Relational Data Model, Comparison between Models. Entities and Relationship Model: Entity types, Entity sets, Attributes and Keys, Relationship Types and Sets, Constraints, Design issue, E-R Diagram, Weak Entity Sets.

Unit II:

Relational Model: Structure of Relational Databases: Relation, Attribute, Domain, Tuples, Degree, Cardinality, Views, Database Relations, Properties of Relations, Attributes, Keys, Attributes of Relation, Domain Constraints, Integrity Constraints. Relational Algebra: Concepts and Operations: Select, Project, Division, Intersection, Union, Division, Rename, Join etc.

Unit III:

SQL: Purpose of SQL, Data Definition Language (DDL) Statements, Data Manipulation Language (DML) Statements Update Statements & Views in SQL, Data Control Language (DCL), Triggers. Relational Database Design: Purpose of Normalization, Data Redundancy and Update Anomalies, Functional Dependency, Process of Normalization, Various Normal Forms: 1NF, 2NF, 3NF, BCNF, Decomposition, Desirable Properties of Decomposition: Dependency Preservation, Lossless Join, Problems with Null Valued & Dangling Tuple, Multivalued Dependencies.

Unit IV:

Transaction Management: Transaction Concept, Transaction State, Concurrent Executions, Serializability: Conflict and View Serializability, Concurrency Control: Lock-Based Protocol, Recovery: Log-Based Recovery.

Unit V: (Dynamic Content)

Big Data and Distributed Databases: CAP theorem, eventual consistency, sharding. NoSQL and NewSQL Databases: Document (MongoDB), Key-Value (Redis), Column-family (Cassandra), Graph (Neo4j), Google Spanner, CockroachDB. Real-Time Data Processing and Streaming Databases: Apache Kafka, Apache Flink.

RECOMMENDED BOOKS

1. Database System Concepts, Abraham Silberschatz Henry F. Korth S. Sudarshan, McGraw-Hill 6th Edition.
2. Database Management System, Raghu Ramakrishnan Johannes Gehrke, McGraw Hill 3rd Edition.
3. Fundamentals of Database System, Elmasri & Navathe, Addison-Wesley Publishing, 5th Edition.
4. An Introduction to Database Systems, Date C. J, Addison-Wesley Publishing, 8th Edition.

COURSE OUTCOMES

After completion of the course students will be able to:

CO1. differentiate between traditional file systems and the database approach.
CO2. construct relational schemas and demonstrate the use of relational algebra operations to query and manipulate relational data.
CO3. develop SQL queries for data definition, manipulation, and control, and analyze relational schemas for normalization.
CO4. examine transaction processing concepts and analyze concurrency control and recovery mechanisms in database systems.
CO5. compare traditional RDBMS with modern distributed and NoSQL databases, and evaluate their suitability for real-time data processing scenarios.

CO-PO Mapping Matrix														
	PO1	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**	**PSO1**	**PSO2**
CO1	2	1	2	1	1							2	1	1
CO2	2	2	2	1	2	1			2		1	2	2	2
CO3	3	3	2	2	2	1		1	2	1	2	2	2	3
CO4	3	3	3	2	2	1	1	1	2	2	2	2	2	3
CO5	2	2	3	1	2	2	2	1	2	2	2	3	3	3

OPERATING SYSTEMS
30242203

COURSE OBJECTIVES

- To provide basic knowledge of computer operating system structures and functioning.
- To compare several different approaches to memory management, file management and process management.
- To understand various problems related to concurrent operations and their solutions.

Unit I

Basics of Operating System: Generations, Types, Structure, Services, System Calls, System Boot, System Programs, Protection and Security. Process Management: Process Concepts, Process States, Process Control Block, Scheduling-Criteria, Scheduling Algorithms and their Evaluation, Threads, Threading Issues.

Unit II

Process Synchronization: Background, Critical-Section Problem, Peterson's Solution, Synchronization Hardware, Semaphores, Classic Problems of Synchronization, Monitors. Deadlock: System Model, Deadlock Characterization, Deadlock Prevention, Detection and Avoidance, Recovery from Deadlock.

Unit III

Memory Management: Main Memory, Swapping, Contiguous Memory Allocation, Paging, Structure of Page Table, Segmentation, Virtual Memory, Demand Paging, Page Replacement Algorithms, Allocation of Frames, Thrashing.

Unit IV

Storage Management: Mass-Storage Structure, Overview, Disk Structure, Disk Attachment, Disk Scheduling. File System Interface: The Concept of a File, Access Methods, Directory Structure, File System Structure, Allocation Methods, Free-Space Management.

Unit V (Dynamic Content)

Containerization and Operating System-Level Virtualization, Kubernetes and Container Orchestration, Secure Boot, Trusted Execution Environments, OS-level Sandboxing and Isolation, Microkernel and Exokernel Architectures, GPU/TPU Scheduling and Driver Integration, Lightweight OS (KaiOS, Tizen, PostmarketOS).

RECOMMENDED BOOKS

1. Operating System Concepts, Silberschatz, Ninth Edition, Wiley Publication.
2. Operating Systems, internals and Design Principles, Stallings, Seventh Edition, Pearson Publication.
3. Modern Operating Systems, Tanenbaum, Fourth Edition. Pearson Publication.

COURSE OUTCOMES

After completion of the course students will be able to:

CO1. describe the fundamental concepts and services of operating systems

CO2. demonstrate solutions to process synchronization problems and analyze different approaches to deadlock detection, avoidance, and recovery.

CO3. apply memory management techniques, and evaluate page replacement algorithms.

CO4. analyze different file systems, storage structures, allocation mechanisms and free-space management methods.

CO5. evaluate the impact of modern OS-level technologies on performance and security.

CO-PO Mapping Matrix														
	PO1	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**	**PSO1**	**PSO2**
CO1	3	3	2	1	2	1			1	1		1	1	1
CO2	3	3	2	2	2	1			2	1		1	1	2
CO3	3	3	2	2	3	1		1	1	1		2	2	1
CO4	3	3	2	1	2	1	1	1	2	1		2	3	2
CO5	3	2	2	1	3	2	2	3	2	1	2	2	2	2

BUSINESS SYSTEM OPTIMIZATION TECHNIQUES
30242204

COURSE OBJECTIVES

- To understand business systems and enterprise processes, and how optimization principles are applied to improve organizational performance.
- To develop skills for system-level modeling, analysis, and decision-making for optimizing modern, technology-enabled business environments.

Unit I:

Foundations of Business Systems & Enterprise Processes: Concept and Scope of Business Systems in Engineering Organizations, Functional v/s Process-Oriented Business Structures, Components of Business Processes and Value Chains, Information Systems in Organizations: TPS, MIS, DSS, ESS, Enterprise Systems: ERP Architecture and Core Modules, CRM and SCM Systems: Coordination and Information Flow, System Development Life Cycle (SDLC), Requirements Identification and Documentation.

Unit II:

Business Modeling & Optimization Concepts: Role of Optimization in Managerial and Engineering Decision-Making, Operations Research Approach: Problem Definition, Modeling, Solution, Implementation, Types of Business Models: Deterministic, Stochastic, Static, Dynamic, Abstraction of Real-World Business Problems into Optimization Models, Objective Functions, Constraints, Feasibility, and Optimality (Interpretive Focus), Trade-Offs, Opportunity Cost, and Marginal Analysis, Sensitivity Analysis and Managerial Interpretation of Model Outcomes.

Unit III:

Networked Business Systems & Project Optimization: Business Networks: Supply, Information, and Project Networks, Project Planning Fundamentals: Activities, Events, Dependencies, Time–Cost Trade-Off and Project Crashing Concepts, Network-Based Decision Logic: Bottlenecks and Coordination, CPM and PERT: Network Construction, Time Estimation and Slack Interpretation, Critical Path Identification.

Unit IV:

Business Performance Optimization & Simulation-Based Analysis: Business Process Optimization Methodologies, Workflow Analysis and Throughput

Improvement, Resource Allocation Strategies under Practical Constraints, Forecasting Approaches for Operational Decision-Making, Inventory Optimization Concepts: Service Levels, Cost Trade-Offs (Non-Derivative), Introduction to Business Simulation: Monte Carlo Simulation, Discrete-Event Simulation (Conceptual Modeling), System Performance Evaluation using KPIs.

Unit V: (Dynamic Content)

Digital & Intelligent Business System Optimization: Optimization through Digital Transformation and Process Reengineering, AI-Assisted and Intelligent Decision Systems for Operational Optimization, Smart and Sustainable Business System Optimization, Integrated, Cross-Functional Optimization in Enterprise Environments, Case Studies on Data-Driven Decision Optimization in Business Systems.

RECOMMENDED BOOKS

- Goel, A., Chauhan, A., & Malik, A. K. (Eds.). (2022). Applications of Advanced Optimization Techniques in Industrial Engineering. CRC Press.
- Hamdy, A. Taha. (2019). Operations Research: An Introduction (10th ed.). Pearson Education.
- James A. O'Brien & George M. Marakas, Management Information Systems, McGraw-Hill.
- Klaus Schwab, The Fourth Industrial Revolution, Crown Publishing.
- Alok Mishra & Pankaj Mishra, Business Analytics, Wiley.

COURSE OUTCOMES

After completion of the course students would be able to:

CO1. describe the structure and functioning of business systems and enterprise processes.

CO2. analyze real-world business problems and model them using optimization concepts and decision frameworks.

CO3. apply project, process, and performance optimization techniques in networked and enterprise environments.

CO4. evaluate business performance using KPIs, forecasting, and simulation-based analysis.

CO5. analyze digital, intelligent, and data-driven approaches for optimizing contemporary business systems.

CO-PO Mapping Matrix														
	PO1	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**	**PSO1**	**PSO2**
CO1	3	2	1	1	1	1	1	1	1	1	1	2	2	1
CO2	3	3	2	1	1	1	1	1	1	1	2	2	3	2
CO3	3	3	3	2	1	1	1	1	1	1	3	1	3	2
CO4	3	3	3	3	2	1	1	1	1	1	2	1	3	2
CO5	3	3	3	2	3	2	2	2	2	2	3	3	3	3

DATA SCIENCE
30242205

COURSE OBJECTIVES

- To enable students to acquire, preprocess, analyze, and visualize data, and perform exploratory and statistical analysis for data-driven decision-making.
- To develop skills in building, evaluating, and validating predictive models using traditional techniques and modern data tools.

Unit I:

Introduction to Data Science: Scope, lifecycle, evolution, and real-life applications, Types of Data: Structured, Unstructured and Semi Structured Data, Data Formats: TXT, CSV, JSON, XLSX, Reading and writing files.

Unit II:

Data Acquisition, Pre-Processing: Data Collection Strategies, Data Pre-Processing: Data Cleaning, Data Integration and Transformation, Data Reduction, Data Discretization, Data Normalization, Outlier Detection.

Unit III:

Exploratory Data Analytics: Descriptive Statistics: Measures of central tendency: Mean, Median, Mode, Measures of dispersion: Variance, Standard Deviation, Shape of distribution: Skewness and Kurtosis, Correlation Analysis: Computing correlation coefficients between features, visualizing correlations using heatmaps, Hypothesis Testing: t-test, Chi-square test, ANOVA.

Unit IV:

Model Development, Evaluation and Visualization: Model Development: Regression Models: Simple Regression, Multiple Regression, Classification Models: Logistic Regression, Decision Tree, Random Forest. Performance Evaluation Measures: Regression and Classification, Model Validation: Train-Test split, Cross-Validation, Learning Curves, Detecting Underfitting and Overfitting, Data Visualization Techniques: Box Plot, Scatter Plot, Histogram, Bar Chart, and Pie Chart.

Unit V: (Dynamic Content)

Ethical Data Management & Tools for Analytics: Data Quality Management and Data Governance: Data Privacy, Security, and Ethical Handling, Data Science Tools:

Introduction to OpenRefine, Trifacta, KNIME, Tableau, Power BI, Google Data Studio, Airtable, Zapier, Power Automate for Data Preparation, Visualization, and Workflow Automation.

RECOMMENDED BOOKS

- McKinney, W. (2018). *Python for data analysis*. O'Reilly Media.
- Kelleher, J. D. (2018). *Data science*. MIT Press.
- Blum, A., Hopcroft, J., & Kannan, R. (2020). *Foundations of data science*. Cambridge University Press.
- Bruce, P. (2019). *Data science: What the best data scientists know*. Wiley.
- Park, A. (2020). *Data science: A beginner's guide*. Independently Published.
- Singh, N., & Huo, K. (2021). *Ace the data science interview*. Data Interview Pro.

COURSE OUTCOMES

After completion of the course students would be able to:

CO1. understand data science concepts, data types, formats, and applications.

CO2. apply data acquisition, preprocessing, normalization techniques for real datasets.

CO3. perform exploratory data analysis using descriptive statistics, correlations, and hypothesis testing methods.

CO4. develop and evaluate regression and classification models using appropriate validation techniques.

CO5. implement ethical data handling and perform workflows using modern no-code platforms.

CO-PO Mapping Matrix														
	PO1	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**	**PSO1**	**PSO2**
CO1	3	3	1	1	1	1			1	2	2	2	3	2
CO2	3	3	2	3	2	1			1	2	2	2	3	2
CO3	3	3	2	3	3	1		1	1	2	2	3	3	2
CO4	3	3	2	3	3	1	1	1	2	2	2	3	3	3
CO5	3	2	3	2	3	2	1	1	2	3	3	3	3	3

COMPETITIVE PROGRAMMING
30242208

COURSE OBJECTIVES

- To build strong foundational skills in programming logic, time–space complexity, and problem-solving techniques used in competitive programming.
- To enable students to apply data structures, algorithms, graphs, dynamic programming, and number theory concepts for solving medium to advanced-level coding problems.
- To prepare students for national and global coding contests by developing speed, accuracy, debugging skills, and consistent practice on online judge platforms.

Unit I:

Basics of C++/Python, input–output, loops, conditionals, functions. Introduction to time–space complexity and constraints. Arrays, strings, frequency maps, simple logic problems. Basics of STL (vector, set, map, pair). Practice: 40–50 beginner problems on LeetCode/CodeChef.

Unit II:

Core Data Structures & Techniques: Stacks, queues, linked lists, heaps, hash maps. Two-pointer, sliding window, prefix sums, hashing. Binary search and sorting-based logic patterns. Practice on GFG DS Track & LeetCode/CodeChef Data Structures. Target: 30–40 DS-based problems.

Unit III:

Graphs & Dynamic Programming, Graph basics: BFS, DFS, components, cycles. Shortest paths: Dijkstra, Bellman-Ford, MST algorithms. Recursion, memoization, DP introduction. DP topics: knapsack, LIS, LCS, grid DP.Practice: 30-40 graph & DP problems on LeetCode/CodeChef.

Unit IV:

Advanced Algorithms & Number Theory, Sieve, modular arithmetic, combinatorics, modular inverse. Fast exponentiation, prime factorization. Greedy strategies and optimization techniques. Bit manipulation & bitmask DP. Practice: 30–40 advanced-level problems across platforms.

Unit V:

Contest Practice & Deployment Registration on LeetCode, CodeChef, GFG, HackerRank. Weekly contests, mock tests, debugging strategies. Editorial writing and

clean coding practices. Mini-project: CP notebook/GitHub repository. Target: 30–40 problems + 6 mock contests.

RECOMMENDED BOOKS

- Competitive Programming 4, by Steven Halim & Felix Halim (2018).
- Guide to Competitive Programming, by Antti Laaksonen (Springer, 2017).
- Programming Challenges: The Competitive Programmer's Handbook, by Laaksonen (Revised Edition 2016).
- Elements of Programming Interviews in C++, by Adnan Aziz et al. (2016).

COURSE OUTCOMES

After completion of the course students would be able to:

CO1. solve beginner-level CP problems using programming fundamentals.

CO2. apply appropriate data structures and algorithmic techniques for intermediate problems.

CO3. use graph algorithms and dynamic programming to solve complex problems.

CO4. implement advanced number theory, greedy methods, and optimization in contest environments.

CO5. participate effectively in coding contests and maintain consistent problem-solving practice.

CO-PO Mapping Matrix														
	PO1	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**	**PSO1**	**PSO2**
CO1	3	3	2	2	2	1			1	2	2	3	3	2
CO2	3	3	2	2	2	2			1	2	2	3	3	2
CO3	3	3	2	2	3	2		1	1	2	2	3	3	2
CO4	3	3	2	2	3	2	1	1	2	2	2	3	3	3
CO5	3	3	3	2	3	2	1	1	2	2	2	3	3	3

PROJECT MANAGEMENT, ECONOMICS & FINANCING
30242211

COURSE OBJECTIVES

- To provide knowledge about project attributes and planning essentials, develop project networks, make rational decisions for project completion, utilize resources effectively, and understand the basics of project finances and management.

Unit I:

Project Planning: Introduction to Project Management, Difference between Project and Production, Attributes of a Project: Time, Cost, Quality and Safety, Stakeholders of a Project, Project Life Cycle, Project Planning: Types of Project Plans and Feasibility.

Unit II:

Project Network Logic: Project Networking and Work Flows, Activity Duration and Methods of Estimating Activity Duration – One Time Estimate Three-Time Estimates, Duration Estimation Procedure, use of Bar Charts, Mile Stone Charts and Networks, Network Representation Schemes: Activity on Arrow and Activity on Node Networks (A-o-A & A-o-N), Logic Behind Developing Project Network and Simple Network Calculations, Critical Paths and Floats.

Unit III:

Decision Making through Networks: CPM, PERT & PDM: Use of Network in Decision Making: Importance of critical path, Monitoring the progress and updating the Project Plan, Use of Floats in Resource Smoothening, Introduction to Precedence Diagramming Method (PDM), Different lag and lead relations in terms of SS (Start to Start), SF (Start to Finish), Finish to Start (FS), and Finish to Finish (FF) and Composite Relations.

Unit IV:

Project Cost Control: Breakeven Analysis in Planning Stage, Direct and Indirect Cost, Slope of Direct Cost Curve, Total Project Cost and Optimum Duration, Contracting the Network for Cost Optimization, Escalation & Variation in Prices.

Unit V:

Projects Financing: Introduction to Project Financing; Role of Governments in Financing Projects, Funder and Concessionaire: Economic Multiplier Effects of Projects; Means of Financing-Public Finance and Private Finance, Granting Authority: World Bank Group, IMF,ADB, Micro and Small Enterprises Funding Scheme

(MSME), Elementary Understanding of Procurement of Infrastructure Projects through Public Private Partnership (PPP) Route, Build Operate Transfer (BOT), Build Operate Own & Transfer (BOOT); Stakeholders' Perspectives, Lifecycle of PPP Projects, Micro & Macroeconomics Concepts and its Application in Project Financing.

RECOMMENDED BOOKS

- Project Management Scheduling PERT and CPM by Dr. B.C. Punmia, K.K. Khandelawal.
- PERT & CPM Principles and Applications by L.S. Srinath, Affiliated EWP Pvt. Ltd.
- Project Planning and Control by Albert Lester, Fourth Edition Elsevier Butterworth-Heinemann.
- A Management Guide to PERT/CPM With GERT/PDM/DCPM and Other networks by Jerome D. Wiest, Ferdinand K. Levy, Prentice Hall.
- Project Management with CPM and PERT by Joseph J. Moder, Cecil R. Phillips, Van Nostrand Reinhold Company.

COURSE OUTCOMES

After completion of the course students would be able to:

CO1. know the attributes of project and its different phases.

CO2. develop the project network based on work breakdown structure and estimation of activity durations.

CO3. analyze the project network and make decide the various alternates.

CO4. evaluate the optimum cost of project for assigned deadlines.

CO5. understand the different options to arrange the finances to complete it within stipulated time.

CO-PO Mapping Matrix														
	PO1	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**	**PSO1**	**PSO2**
CO1	2	1	1	1	1	1			2	2	3	2	2	2
CO2	2	2	3	2	2	2	1	1	3	2	3	3	2	3
CO3	2	2	2	2	2	2		1	2	2	3	3	2	2
CO4	1	2	2	2	2	2	1	1	2	2	3	3	2	3
CO5	1	1	2	2	1	2	1	1	2	2	2	3	2	2

DATABASE MANAGEMENT LAB
30242206
LIST OF PROGRAMS

While creating tables and databases, ensure that each name begins with your roll number as a prefix. Example: If your roll number is 25, then every table name should start with "25_TABLE_NAME".
Additionally, follow the instructions below:

- *Mention the program name clearly.*
- *Provide a brief description of each command used to execute the query.*
- *Write all SQL commands in bold letters.*
- *Attach a screenshot of the output.*

1. Implementation of DDL Commands in SQL

- Create table (CREATE TABLE)
- Modify table structure (ALTER TABLE)
- Delete table (DROP TABLE)

2. Implementation of DML Commands in SQL

- Insert records (INSERT INTO)
- Update records (UPDATE)
- Delete records (DELETE)

3. Implementation of SQL Functions

- Number Functions
- Aggregate Functions
- Character Functions
- Conversion Functions
- Date Functions

4. Implementation of SQL Operators

- Arithmetic Operators
- Logical Operators
- Comparison Operators
- Set Operators
- Special Operators (LIKE, BETWEEN, IN)

5. Implementation of SQL Joins

- Inner Join
- Outer Join
- Natural Join

6. Grouping, Filtering & Indexing

- GROUP BY Clause
- HAVING Clause
- ORDER BY Clause
- Creating Index (CREATE INDEX)

7. Subqueries and Views

- Single-row & Multi-row Subqueries

- Creating and Managing Views (CREATE VIEW, DROP VIEW)

8. Implementation of Constraints

- Primary Key
- Foreign Key
- Unique
- Not Null
- Check
- Default

9. Database Backup & Recovery

- Export & Import Operations
- Backup Commands
- Recovery Operations

10. Transaction Control Language (TCL)

- COMMIT
- ROLLBACK
- SAVEPOINT

11. Creating Database / Table Space

- CREATE DATABASE
- Managing Storage Structure

12. User & Role Management

- Create User
- Drop User
- Grant Privileges
- Revoke Privileges

13. Trigger Implementation

- BEFORE and AFTER Triggers
- Row-level and Statement-level Triggers

14. ACID Properties & Transaction Management

- Atomicity
- Consistency
- Isolation
- Durability
- Practical Execution with SQL Queries

15. MongoDB NoSQL Operations

- CRUD Operations
- Aggregation Pipeline
- Collections & Documents
- Basic Queries in MongoDB Compass or Shell

COURSE OUTCOMES

After completion of the course students would be able to:

CO1. understand the fundamental concepts of SQL, including data types, DDL, DML, DCL, and TCL commands.

CO2. enable to design, create, and manage databases using constraints, functions, operators, and advanced SQL features.

CO3. develop skill proficiency in writing SQL queries involving joins, subqueries, grouping, indexing, triggers, and transaction control.

CO4. utilize database objects such as Tables and Views in appropriate scenarios.

CO5. develop a mini-project demonstrating end-to-end database design and implementation for a real-world application.

CO-PO Mapping Matrix														
	PO1	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**	**PSO1**	**PSO2**
CO1	2	2	1	1	1	1				1	1	2	2	2
CO2	3	2	2	2	2	1			1	1	2	2	2	2
CO3	3	3	2	2	2	2	1	1	1	1	2	3	3	2
CO4	3	3	2	2	2	2	1	1	1	1	2	3	3	3
CO5	3	3	2	2	3	2	2	2	1	2	3	3	3	3

DATA SCIENCE LAB
30242207
LIST OF PROGRAMS

1. Write a program that reads datasets from TXT, CSV, JSON, and XLSX formats, converts a CSV file into JSON and a JSON file into XLSX, writes the cleaned data into a selected folder.
2. Apply complete data preprocessing on the Adult Income Dataset and Loan Prediction Dataset, including Data Cleaning, Data Integration and Transformation, Data Reduction, Data Discretization, and Data Normalization.
3. Write a program to write a data frame given in Fig.1 to csv file and analyse a given sample dataset by finding its data distribution using histogram, column chart, box plot, scatter plots, pie chart identifying outliers through visualization, and creating histogram, bar chart, and pie chart to represent the data.

S. No.	Empid	Gender	Age	Sales	BMI	Income
1	E1	M	32	123	NORMAL	350
2	E2	M	35	114	OVERWEIGHT	450
3	E3	F	34	135	OBESITY	160
4	E4	F	42	139	UNDERWEIGHT	190
5	E5	F	26	117	NORMAL	90
6	E6	M	38	121	OBESITY	140
7	E7	F	44	133	OVERWEIGHT	125
8	E8	M	24	140	UNDERWEIGHT	95
9	E9	M	36	133	OVERWEIGHT	200
10	E10	F	40	133	OBESITY	310

Fig.1

4. Perform EDA on Credit Card Fraud Detection Dataset (open-source dataset) for analysing the data.
5. Apply Regression Model techniques to predict the future values of data on the open-source available datasets. (Housing Price Prediction Dataset, Stock Market Prices Dataset)
6. Import the Iris Flower dataset from the UCI Machine Learning Repository containing three species of iris flowers (Setosa, Versicolor, Virginica). Apply a logistic regression model for multi-class classification to categorize the iris samples based on their features.
7. Plot Accuracy and Error Metrics against number of iterations for evaluation of model performance (consider any dataset mentioned in program 2-6).
8. Plot Learning curves for model evaluation for Under-fitting and Over-fitting (consider any dataset mentioned in program 2-6).
9. Write a Python script to clean a CSV file by removing duplicates and normalizing column names, similar to OpenRefine operations.
10. Write code to automate reading a CSV file, transforming selected columns, and exporting a summary report, replicating a KNIME or Power Automate workflow.

COURSE OUTCOMES

After completion of the course students would be able to:

CO1. apply techniques to read, convert, clean, and manage diverse data.

CO2. implement pre-processing methods to prepare datasets for analysis tasks.

CO3. perform exploratory data analysis and visualize data distributions using various plots.

CO4. implement predictive models for classification and regression to interpret the data.

CO5. create automated data workflows and preprocessing scripts inspired by no-code tools.

CO-PO Mapping Matrix														
	PO1	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**	**PSO1**	**PSO2**
CO1	3	2	1	2	1	1	1		1	1	1	2	2	2
CO2	3	3	2	3	2	1		1	2	1	1	3	3	2
CO3	3	3	2	3	2	1	1	1	2	1	1	3	3	2
CO4	3	3	3	3	2	1	1	1	2	1	3	3	3	2
CO5	3	3	2	2	3	3	1	1	3	2	2	3	3	3

Centre for Computer Science and Technology

LIST OF MACRO PROJECT-II
(30242210)

1. **Student Attendance & Timetable Management System**
   Develop a Python and DBMS-based application that manages student attendance, teacher availability, and classroom allocation. The system should generate automated timetables using constraint handling (teacher slots, room capacity) and provide reports using data visualization techniques.
2. **Multi-User File Organizer and Search Utility**
   Design an Object-Oriented Python application that stores file metadata in a database and allows users to search files quickly using efficient algorithms such as hashing or binary search. The system should also support directory organization, sorting, and duplicate-file detection.
3. **Hospital Appointment & Resource Scheduling System**
   Create a system that manages doctor availability, appointment booking, and room allocation. Apply scheduling algorithms (FCFS, Priority) and store all data in a relational database. The system should also generate daily reports for hospital administration.
4. **College Library Database & Book Recommendation Console (Non-ML)**
   Build a library management application with features such as book issue/return, fine calculation, and indexing using data structures like trees or hash tables. The database should support search by title/author and generate usage summaries using Python-based visualization.
5. **Supermarket Billing & Inventory Tracking System**
   Develop a billing system that updates stock information using DBMS transactions and triggers. Implement rule-based pricing, discount handling, and quick search operations using appropriate data structures. Provide sales reports and inventory summaries.
6. **Transport Route & Fare Management System**
   Create a system that stores bus/transport routes in a database and uses graph algorithms (BFS, DFS, shortest path) to compute optimal routes. The application should support fare calculation, route search, and generate peak-hour usage charts.
7. **File Compression & Decompression Tool**
   Design an OOP-based tool that compresses and decompresses files using algorithmic techniques such as Huffman Coding. The system should visualize compression ratio, file statistics, and allow saving compressed files in a structured directory.
8. **Hostel Room Allocation & Fee Management System**
   Implement a database-backed application for allocating hostel rooms to students based on availability and predefined rules. The system should maintain fee records, generate receipts, schedule room inspections, and produce occupancy analytics.

9. **CPU Scheduling Simulator with Performance Dashboard**
   Create an application that simulates CPU scheduling algorithms (Round Robin, SJF, Priority, FCFS). Allow users to input processes, burst times, priorities, and visualize Gantt charts along with performance measures such as waiting time and turnaround time.
10. **Sales Data Analyzer & Business Report Generator**
    Develop a Python-based dashboard that loads sales data from CSV/SQL, performs descriptive statistical analysis (totals, averages, seasonal patterns), and generates visual business reports. Include optional optimization-based revenue or cost analysis using basic OR techniques.
11. **Campus Facility Booking & Utilization Analyzer**
    Develop a Python and DBMS-based system for reserving campus facilities such as labs, seminar halls, and sports areas. The system should prevent overlapping bookings, maintain user logs, and generate utilization statistics using data aggregation. Use scheduling or queue-based algorithms to resolve slot conflicts.
12. **Personalized Study Planner & Productivity Tracker**
    Create an OOP-based application that allows students to plan study sessions, set goals, and track productivity. The system should implement algorithmic scheduling (priority-based or interval scheduling) to reorganize tasks automatically when deadlines change. Store progress data in a database and generate weekly analytical summaries.
13. **Secure Password Vault with Encryption Algorithms**
    Build a desktop tool in Python that securely stores user passwords using encryption (e.g., AES-like logic, without external AI/ML). Implement OOP design, file handling, and hashing algorithms. Provide features like password strength analysis, search, and categorized storage.
14. **Railway Platform & Train Movement Simulation**
    Design a simulator that models train arrivals, departures, and platform allocation. Integrate algorithmic concepts such as queue management, greedy scheduling, and resource allocation. The system should visualize train flow and store logs in a database.
15. **College Mess Menu Optimizer & Inventory Monitor**
    Create a Python application that maintains daily mess menus, tracks ingredient usage, and updates stock levels. Use DBMS triggers or transactions to ensure consistency. Apply simple optimization techniques to suggest cost-effective weekly menus based on available supplies.
16. **Document Version Control & Merge Utility**
    Develop a lightweight version-control tool (non-Git) that stores document revisions, detects differences, and supports basic merging operations. Use dynamic programming or string comparison algorithms. Maintain revision history in a database and display version timelines.

17. **Smart Parking Allocation & Fee Calculation System**
    Create a parking management system that allocates slots based on vehicle type and availability. Use search and sorting algorithms to optimize slot assignment. Store entry/exit logs in a database and auto-generate fees using rule-based logic.
18. **E-Learning Quiz Generator & Performance Dashboard**
    Build a Python-OOP-based application where teachers create quizzes, students attempt them, and results are stored in a database. Use algorithmic question randomization, timer implementation, and performance analytics to generate progress charts. No ML-based recommendation is required.
19. **Multi-Threaded File Downloader & Integrity Checker**
    Develop a utility that downloads files using multi-threading to improve speed. Implement algorithms for splitting file chunks, merging, and verifying data using checksum or hashing. Provide download statistics and allow pausing/resuming.
20. **Supply Chain Mini-Simulator for Small Businesses**
    Construct a simulation system that models product flow between suppliers, warehouses, and customers. Integrate graph or queue-based algorithms to manage order processing, delivery times, and stock movement. Use Python and DBMS to store transactional data and show optimization-based cost summaries.
21. **Online Examination & Result Processing System**
    Develop a web-based exam platform in Python (Flask/Django), HTML/CSS, and DBMS with secure login, question banks, randomized papers, timers, auto-evaluation for objective questions, manual grading for subjective responses, and result analytics with downloadable reports.
22. **Employee Payroll & Leave Management System**
    Create an OOP-driven payroll system with DBMS-backed employee records, salary components, leave application workflows, approval hierarchy, and automated payslip generation. Include audit logging, role-based access, and monthly visualization of payroll cost trends.
23. **Banking Transaction & Rule-Based Fraud Monitor**
    Build a secure Python web app handling accounts, transfers, deposits, withdrawals with ACID transactions, triggers for ledger updates, and rule-based anomaly detection (velocity checks, amount thresholds). Provide HTML dashboards for users and admins with exportable statements.
24. **Event Management & Ticket Booking System**
    Implement an event portal with seat maps, booking windows, cancellation/resale logic, waitlists, and payment records. Use DBMS for inventorying seats, apply greedy or interval scheduling for sessions, and render occupancy charts with revenue summaries.
25. **E-Commerce Cart, Checkout & Order Tracking**
    Develop an online store prototype with catalog management, shopping cart, coupons, tax/shipping calculations, and order lifecycle tracking. Use DBMS transactions for stock updates, OOP services for pricing rules, and HTML order status pages with sales analytics.

26. Airline Reservation & Flight Scheduling

Create a flight booking system with route graphs, fare classes, seat allocation, and cancellation policies. Use shortest path for multi-leg itineraries, DBMS for PNR records, and HTML itineraries. Include load-factor dashboards and delay statistics.

27. Hostel Complaint & Maintenance Tracker

Build a web app where residents file issues, admins prioritize and assign tasks, and staff update status. Use DBMS for ticket history, SLA timers, notification logs, and HTML dashboards showing backlog, resolution time, and category heatmaps.

28. Resume Builder & Job Application Portal

Create a system with resume templates, HTML form builder, profile storage in DBMS, job postings, application workflows, and recruiter views with filtering. Include export to PDF, tagging, and analytics of applications per job and candidate pipeline.

29. Secure Online Voting & Poll Management

Develop a polling/voting system with one-person-one-vote enforcement, ballot design, time windows, and result tabulation. Use hashing for ballot integrity, DBMS for audit logs, and HTML live results with turnout graphs and constituency summaries.

30. Course Registration & Fee Payment

Implement course catalogs, prerequisite validation, enrollment caps, waitlists, and fee invoicing/receipts. Use DBMS constraints and triggers, OOP services for rules, and HTML student/admin portals with enrollment analytics and financial summaries.

31. Complaint Redressal & Feedback Analytics

Create a unified portal for complaints, feedback categorization, priority queues, and resolution workflows. Store threads in DBMS, implement status transitions, and present HTML dashboards with sentiment tags (rule-based), SLA breach alerts, and trend charts.

32. Auction & Bidding Platform

Build an auction site with item listings, bid increments, time-bound auctions, proxy bidding logic, and winner determination. Use DBMS for bids history, OOP modules for rules, and HTML realtime countdowns with post-auction settlement reports.

33. Food Ordering & Delivery Coordination

Develop menus, orders, kitchen queueing, delivery assignment, and status tracking. Apply queue algorithms for kitchen tasks, DBMS for order lifecycle, and HTML order pages with prep-time estimates and shift-wise sales analytics.

34. Movie Ticket Booking & Seat Allocation

Create cinema listings, showtimes, seat selection, dynamic pricing rules, and booking/cancellation workflows. Use DBMS for seat locks, OOP pricing engine, and HTML seat maps with occupancy and revenue per show.

35. Insurance Policy & Claims Management

Implement policy issuance, premium schedules, claims filing, validation rules, and settlement tracking. Use DBMS triggers for payment updates, OOP modules for claim rules, and HTML dashboards for claim ratios and outstanding liabilities.

36. Gym Membership & Workout Scheduler

Build a portal for memberships, trainer slots, workout plans, and progress logs. Use interval scheduling for trainer allocation, DBMS for workout history, and HTML progress charts with attendance and plan adherence metrics.

37. Hotel Booking & Room Service Management

Develop room inventory, booking calendars, check-in/out, housekeeping tasks, and service requests. Use DBMS for reservations and tasks, OOP modules for pricing and availability, and HTML dashboards for occupancy and service SLAs.

38. Vehicle Rental & Utilization Tracking

Create fleet inventory, booking windows, damage logs, fuel tracking, and billing. Use DBMS for rentals, OOP engines for rate calculation, and HTML utilization charts with maintenance schedules and idle-time analytics.

39. Blood Bank & Donor Registry

Implement donor profiles, blood group inventory, request matching, and expiry alerts. Use DBMS triggers for stock updates, OOP modules for compatibility rules, and HTML dashboards for supply-demand and donation event planning.

40. Sports Tournament Scheduling & Scoring

Build league/knockout scheduling, referee assignments, live scoring, and standings. Use round-robin or bracket algorithms, DBMS for fixtures/results, and HTML leaderboards with fair-play and form charts.

41. Music Library & Playlist Manager

Create upload/metadata management, playlists, favorites, and playback history. Use DBMS for media indexing, OOP services for playlist rules, and HTML players with usage stats and top tracks per user.

42. Travel Packages & Booking Portal

Develop itineraries, bookings, payments, cancellations, and voucher generation. Use DBMS for package and booking records, OOP modules for pricing bundles, and HTML dashboards for seasonal demand and revenue per package.

43. Alumni Network & Event Manager

Implement alumni profiles, batch groups, events, registrations, and messaging. Use DBMS for members and participation logs, and HTML dashboards for engagement analytics, donations, and mentorship mappings.

44. Donation & Fundraising Campaigns

Create campaigns, donation processing, receipt generation, and fundraising targets. Use DBMS for donor records, OOP modules for campaign rules, and HTML progress bars with donor segmentation and impact summaries.

45. Recruitment & Interview Scheduling

Develop job postings, applications, screening stages, interview slots, and evaluation forms. Use DBMS for candidate pipelines, interval scheduling for interviews, and HTML dashboards for time-to-hire and stage conversion rates.

46. Real Estate Listings & Visit Booking

Implement property catalogs, filters, visit scheduling, offer tracking, and agreements. Use DBMS for listings and leads, OOP pricing and availability rules, and HTML maps and analytics for lead sources and conversion.

47. **Conference Registration & Paper Submission**
    Build call-for-papers, submission forms, reviewer assignments, decision workflows, and registration payment. Use DBMS for papers and reviews, OOP modules for conflict-of-interest rules, and HTML acceptance stats and program builder.
48. **Loan Management & EMI Scheduler**
    Create loan products, customer onboarding, amortization schedules, payments tracking, and delinquency flags. Use DBMS triggers for installments, OOP modules for interest calculation, and HTML dashboards for NPA and recovery analytics.
49. **Disaster Relief Resource Allocation**
    Implement inventory of relief items, request intake, prioritization rules, route planning (graph), and delivery logs. Use DBMS for stock and distribution, OOP allocation engine, and HTML dashboards for coverage, deficits, and lead time.
50. **Tourism Guide & Location Finder**
    Develop attractions catalog, routes, accommodations, and itinerary builder. Use DBMS for locations and ratings, graph algorithms for shortest paths, and HTML maps with visit analytics and peak-time charts.
51. **Tourism Guide & Location Finder**
    Develop attractions catalog, routes, accommodations, and itinerary builder. Use DBMS for locations and ratings, graph algorithms for shortest paths, and HTML maps with visit analytics and peak-time charts.
52. **Retail Sales Data Analysis (End-to-End Data Science Workflow)**
    Using a real or synthetic retail sales dataset (CSV/JSON), perform the complete data science lifecycle, data acquisition, cleaning, preprocessing, exploratory data analysis, model building (regression), and evaluation. Present final insights using visual dashboards in Tableau/Power BI.
53. **Student Performance Prediction System**
    Collect or download a dataset containing student marks, attendance, and activity records. Clean and preprocess the data, analyze statistical patterns, visualize performance trends, and build regression/classification models to predict student outcomes. Validate models using cross-validation.
54. **Healthcare Data Quality & Governance Assessment**
    Select a healthcare dataset (CSV/Excel) and perform data quality checks—missing values, inconsistencies, outliers, duplicates. Create a data governance report addressing privacy, ethical issues, data security, and propose improvement strategies using OpenRefine/Trifacta.
55. **Customer Churn Classification System**
    Collect or use an open telecom/banking dataset. Clean and integrate data, handle imbalances, visualize churn patterns, compute correlations, and build classification models

(Decision Tree, Random Forest). Evaluate accuracy, precision, recall, and confusion matrix.

56. Predictive Analytics for Traffic Accident Severity

Acquire open datasets on road accidents (CSV/JSON). Perform preprocessing, correlation analysis, visualization heatmaps, and build classification models to predict accident severity. Evaluate performance using cross-validation and identify features influencing severity.

57. Online Shopping Review Quality Analysis

Collect customer review data (text + star ratings). Perform preprocessing, detect outliers in ratings, visualize sentiment distribution using bar/pie charts, compute correlation between review length and user rating, and evaluate logistic regression for positive/negative classification.

58. Banking Loan Approval Pattern Analysis

Clean and preprocess a real or synthetic loan dataset. Analyze customer behavior, perform hypothesis testing (e.g., Chi-square for approval vs. employment status), visualize risk groups, and identify key factors influencing loan approval.

59. Energy Consumption Analytics

Collect energy usage data (hourly/daily). Preprocess, integrate multiple sources if available, analyze patterns using descriptive statistics, visualize consumption trends, and build regression models to forecast future usage.

60. Agriculture Crop Yield Data Study

Use multi-year crop yield data by region. Perform normalization, reduction, calculate variance/standard deviation, correlation heatmaps, and develop a regression model to predict crop yield based on rainfall, fertilizers, and soil data.

61. Tourism Footfall Analysis

Clean and transform tourism data collected monthly/yearly. Apply histograms, pie charts, boxplots; compute correlations between season and tourist volume; and build a simple regression model for future visitor predictions.

COURSE OUTCOMES

After completion of the course students would be able to:

CO1. understand multidisciplinary technology concepts required to build integrated macro-level software projects.

CO2. apply programming, database, and web development skills to develop functional multi-module applications.

CO3. analyze user requirements to design system architecture, data flow, and module interactions effectively.

CO4. evaluate project performance, data handling, and system efficiency using testing and validation techniques.

CO5. create a complete end-to-end solution integrating C++, Python, DBMS, web technologies, and data science.

CO-PO Mapping Matrix														
	PO1	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**	**PSO1**	**PSO2**
CO1	3	2	2	2	1	1			2		1	2	3	2
CO2	3	3	3	2	2	1			2	2	2	2	3	3
CO3	3	3	3	3	2	1	1	1	2	2	2	2	3	3
CO4	2	3	3	3	2	1	1	1	2	2	2	3	3	3
CO5	3	3	3	2	3	1	1	1	3	2	3	3	3	3