DAYANANDA SAGAR COLLEGE OF ENGINEERING

(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)

NAAC Accredited with A Grade

ISO 9001:2008 Certified CHOICE BASED CREDIT SYSTEM (CBCS)

SCHEME OF TEACHING AND EXAMINATION 2022-2023

Department of computer Science and Design

DATABASE MANAGEMENT SYSTEMS			
Course code	21CG43	**Credits**	04
L: T: P:		**CIE Marks**	50
Exam Hours:	03	**SEE Marks**	100
Total Hours:			

COURSE OBJECTIVES	
1:	Understand the basic concepts and the applications of database systems.
2:	Understand the relational database design principles.
3:	Master the basics of SQL and construct queries using SQL.
4:	Understand the process of Normalization, its types and Dependencies.
5:	Familiar with the basic issues of transaction processing and concurrency control.

COURSE OUTCOMES: AT THE END OF THE COURSE, STUDENTS WILL BE ABLE TO:	
CO1	Identify, analyse and define database objects, Data Modelling using Tools enforce integrity constraints and Design ER Diagrams for real world applications.
C02	Identify and Design a relation database schema.
CO3	Write and implement Structured Query Language (SQL) for database manipulation.
CO4	Define various design guidelines for relation schemas and apply functional dependencies to analyse normalization concepts.
CO5	Apply transaction processing and concurrency control techniques. Students will be learning Databases currently used in large industries.

MAPPING OF COURSE OUTCOMES TO PROGRAM OUTCOMES:															
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
CO1	-	3	3	1	3	2	-	-	-	-	-	-	1	-	-
CO2	1	1	1	1	1	-	-	-	-	-	-	-	-	-	-
CO3	1	1	1	1	1	-	-	-	-	-	-	1	1	-	-
CO4	1	1	1	1	1	-	-	-	-	-	-	-	1	-	1
C05	1	1	1	1	1	1	1	1	-	-	-	-	1	-	-

DAYANANDA SAGAR COLLEGE OF ENGINEERING

(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)

NAAC Accredited with A Grade

ISO 9001:2008 Certified CHOICE BASED CREDIT SYSTEM (CBCS)

SCHEME OF TEACHING AND EXAMINATION 2022-2023

Department of computer Science and Design

MODULE	MODULE CONTENTS	HOURS	CO's
1.	**Introduction to Databases:** Introduction, Characteristics of database approach, Advantages of using the DBMS approach, History of database applications. **Overview of Database Languages and Architectures:** Data Models, Understanding and creation of Data modelling using available tools. Schemas, and Instances. Three schema architecture and data independence, database languages, and interfaces, The Database System environment. **Conceptual Data Modelling using Entities and Relationships:** Entity types, Entity sets, attributes, roles, and structural constraints, Weak entity types, ER diagrams, Examples.	08	
2.	**Relational Model**: Relational Model Concepts, Relational Model Constraints and relational database schemas, Update operations, transactions, and dealing with constraint violations. **Relational Algebra:** Relational algebra: introduction, Selection and projection, set operations, renaming, Joins, Division. Examples of Queries in relational algebra. **Mapping Conceptual Design into a Logical Design:** Relational Database Design using ER-to-Relational mapping.	08	
3.	**SQL:**SQL data definition and data types, specifying constraints in SQL, retrieval queries in SQL, INSERT, DELETE, and UPDATE statements in SQL, Additional features of SQL. **Advances Queries:** More complex SQL retrieval queries, Specifying constraints as assertions and action triggers, Views in SQL, Schema change statements in SQL.	08	
4.	**Normalization: Database Design Theory –** Introduction to Normalization using Functional and Multivalued Dependencies: Informal design guidelines for relation schema, Functional Dependencies, Normal Forms based on Primary Keys, Second and Third Normal Forms, Boyce-Codd Normal Form, Multivalued Dependency and Fourth Normal Form, Join Dependencies and Fifth Normal Form. Examples on normal forms.	08	
5.	**Transaction management and Concurrency Control:** Transaction management: ACID properties, serializability and concurrency control, Lock based concurrency control (2PL, Deadlocks), Time stamping methods, optimistic methods, database recovery management. Introduction to NOSQL Databases. Fundamentals of MangoDB.	08	

TEXT BOOKS	
1.	Fundamentals of Database Systems, Ramez Elmasri and Shamkant B. Navathe, 7th Edition, 2017, Pearson.
2.	Database management systems, Ramakrishnan, and Gehrke, 3rd Edition, 2014, McGraw Hill
REFERENCE BOOKS	
1.	Silberschatz Korth and Sudharshan, Database System Concepts, 6th Edition, Mc-GrawHill, 2013.
2.	Coronel, Morris, and Rob, Database Principles Fundamentals of Design, Implementation and Management, Cengage Learning 2012.

DATABASE MANAGEMENT LABORATORY WITH MINI PROJECT			
Course code		**Credits**	
L: T: P:		**CIE Marks**	
Exam Hours:		**SEE Marks**	
Total Hours:			

COURSE OBJECTIVES	
1:	Foundation knowledge in database concepts, technology and practice to groom students into well-informed database application developers.
2:	Understand the database design, schema and relationship between the tables.
3:	Understand joins and subqueries to fetch the data from multiple tables in a single query.
4:	Strong practice in SQL programming through a variety of database problems.
5:	To Develop databases for different real world scenarios.

COURSE OUTCOMES: AT THE END OF THE COURSE, STUDENTS WILL BE ABLE TO:	
CO1	Identify the Entities, Attributes and different Constraints for the given Database requirements commonly used in day to day life in the different fields like Education, Banking, Business and all other fields where there is a need for data security, data updating, management and its maintenance.
C02	Design a Database schema and establish the relationships using foreign keys.
CO3	Create, Update and query on the database.
CO4	Demonstrate the working of different concepts of DBMS.
CO5	Implement, analyse and evaluate the mini project developed for an application in different sectors like Healthcare, Travel, Food etc.

MAPPING OF COURSE OUTCOMES TO PROGRAM OUTCOMES:

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
CO1	-	1	1	1	-	1	1	-	1	-	-	1	-	-	-
CO2	1	1	1	1	-	1	-	-	1	-	-	1	-	-	-
CO3	1	1	-	1	1	1	-	-	1	-	-	-	1	-	-
CO4	1	1	-	1	1	1	-	-	1	1	-	1	1	-	-
C05	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-

PART A: SQL PROGRAMMING

PROGRAM 1: INSURANCE DATABASE

Consider the Insurance database given below. Table names and Data types are specified.

PERSON (driver – id #: String, name: String, address: String)
CAR (Regno: String, model: String, year: int)
ACCIDENT (report-number: int, date: date, location: String)
OWNS (driver-id #: String, Regno: String)
PARTICIPATED (driver-id: String, Regno: String, report-number: int, damage-amount: int)

WRITE THE SQL QUERIES TO:

1. Create the above tables by properly specifying the primary keys and the foreign keys.
2. Enter at least five tuples for each relation.
3. Demonstrate how to add a new accident to the database.
4. Find the total number of people who owned cars that involved in accidents in 2008.
5. Find the number of accidents in which cars belonging to a specific model were involved.

PROGRAM 2: COMPANY DATABASE

Consider the schema for Company Database:
EMPLOYEE(SSN, Name, Address, Sex, Salary, SuperSSN, DNo)
DEPARTMENT(DNo, DName, MgrSSN, MgrStartDate)
DLOCATION(DNo,DLoc)
PROJECT(PNo, PName, PLocation, DNo)
WORKS_ON(SSN, PNo, Hours)

WRITE THE SQL QUERIES TO:

1. Make a list of all project numbers for projects that involve an employee whose last name is 'Scott', either as a worker or as a manager of the department that controls the project.
2. Show the resulting salaries if every employee working on the 'IoT' project is given a 10 percent raise.
3. Find the sum of the salaries of all employees of the 'Accounts' department, as well as the maximum salary, the minimum salary, and the average salary in this department
4. Retrieve the name of each employee who works on all the projects controlled by department number 5 (use NOT EXISTS operator).
5. For each department that has more than five employees, retrieve the department number and the number of its employees who are making more than Rs.6,00,000.

PROGRAM 3: BANKING ENTERPRISE DATABASE

Consider the following database for a banking enterprise.
BRANCH (branch-name: String, branch-city: String, assets: real)
ACCOUNTS (accno: int, branch-name: String, balance: real)
DEPOSITOR (customer-name: String, customer-street: String,
customer-city: String)
LOAN (loan-number: int, branch-name: String, amount: real)
BORROWER (customer-name: String, loan-number: int)

WRITE THE SQL QUERIES TO:

1. Create the above tables by properly specifying the primary keys and the foreign keys.

2. Enter at least five tuples for each relation.
3. Find all the customers who have at least two accounts at the *Main* branch.
4. Find all the customers who have an account at *all* the branches located in a specific city.
5. Demonstrate how you delete all account tuples at every branch located in a specific city.
6. Generate suitable reports.
7. Create suitable front end for querying and displaying the results.

PROGRAM 4: LIBRARY DATABASE

Consider the following schema for a Library Database:

BOOK(Book_id, Title, Publisher_Name, Pub_Year)
BOOK_AUTHORS(Book_id, Author_Name)
PUBLISHER(Name, Address, Phone)
BOOK_COPIES(Book_id, Programme_id, No-of_Copies)
BOOK_LENDING(Book_id, Programme_id, Card_No, Date_Out, Due_Date)
LIBRARY_PROGRAMME(Programme_id, Programme_Name, Address)

WRITE THE SQL QUERIES TO:

1. Retrieve details of all books in the library – id, title, name of publisher, authors, number of copies in each Programme, etc.
2. Get the particulars of borrowers who have borrowed more than 3 books, but from Jan 2017 to Jun 2017.
3. Delete a book in BOOK table. Update the contents of other tables to reflect this data manipulation operation.
4. Partition the BOOK table based on year of publication. Demonstrate its working with a simple query.
5. Create a view of all books and its number of copies that are currently available in the Library.

PROGRAM 5: ORDER PROCESSING DATABASE

Consider the following relations for an Order Processing database application in a company.
CUSTOMER (CUST #: int, cname: String, city: String)
ORDER (order #: int, odate: date, cust #: int, ord-Amt: int)
ITEM (item #: int, unit-price: int)
ORDER-ITEM (order #: int, item #: int, qty: int)
WAREHOUSE (warehouse #: int, city: String)
SHIPMENT (order #: int, warehouse #: int, ship-date: date)

WRITE THE SQL QUERIES TO:

1. Create the above tables by properly specifying the primary keys and the foreign keys and the foreign keys.
2. Enter at least five tuples for each relation.
3. Produce a listing: CUSTNAME, #oforders, AVG_ORDER_AMT, where the middle column is the total numbers of orders by the customer and the last column is the average order amount for that customer.
4. List the order# for orders that were shipped from all warehouses that the company has in a specific city.
5. Demonstrate how you delete item# 10 from the ITEM table and make that field *null* in the ORDER_ITEM table.

PART B: Mini project: For any problem selected, make sure that the application should have five or more tables. Indicative areas include: Organization, health care, Ecommerce etc.

- For any problem selected, make sure that the application should have **five or more** tables. Indicative areas include: Organization, health care, Ecommerce etc.
- Students can pick one experiment from the questions lot of PART A with an equal choice to all the students in a batch. For PART B, the project group (Maximum of 4 students per batch) should demonstrate the mini-project.
- Weightage of marks for PART A is 60% and for PART B is 40%.
- Change of experiment is allowed only once and Marks allotted to the procedure part to be made zero (Not allowed for Part B).
- Mini project can be done using any DBMS – for back end and any Programming language for the front end as per the choice of students.
- Mini-Project report should be submitted in the form of Hard copy, spiral binding and it should be as per the department standards and format.