FACULTY OF COMPUTER APPLICATIONS
Master of Science – Cyber Security and Cyber Law

- **Sem.** : 1
- **Subject Code** : 05CS3103
- **Subject** : Database Management System and Security
- **Course Objectives** :

1. Introduce the fundamental concepts, architecture, and models of database management systems.
2. Enable students to design efficient databases using Entity-Relationship modelling and normalization techniques.
3. Knowledge of SQL and PL/SQL for data definition, manipulation, and database programming.
4. Explain transaction management concepts for maintaining the consistency and reliability of databases.
5. Impart essential knowledge of database security mechanisms to protect data confidentiality, integrity, and availability

Prerequisites: Basic knowledge of operating and working with the computer system.

FACULTY OF COMPUTER APPLICATIONS
Master of Science – Cyber Security and Cyber Law

Unit No	Topics Covered	No of lectures required
1	**Database System Concepts and Architecture:** • Introduction to Database Management Systems • Database applications • Characteristics of database approach • Instance and schema • Data independence • Overview of database models • Database system architectures • Role of DBMS in ensuring data confidentiality, integrity, and availability	9
2	**Database Design using ER Model and Normalization:** Entity Relationship Model • Basic ER concepts • Entities, attributes, and relationships • Constraints • ER diagram symbols and examples Normalization • Introduction to normalization • First Normal Form (1NF) • Second Normal Form (2NF) • Third Normal Form (3NF) • Normalization for reducing redundancy and update anomalies	9
3	**SQL and Relational Query Processing:** • Overview of SQL • Categories of SQL commands: DDL, DML, DCL, TCL, DQL • Constraints and integrity enforcement • SQL operators • GROUP BY, HAVING, ORDER BY • Types of joins • Built-in functions • Introduction to secure SQL practices	9

	• Misuse of SQL leading to injection vulnerabilities (conceptual understanding)	
4	**Transaction Management and PL/SQL:** Transaction Management • Overview of transaction processing • Transaction states and state transition diagram • ACID properties • Transaction failures and recovery concepts PL/SQL • PL/SQL block structure • Stored procedures and functions • Triggers • Exception handling	**9**
5	**Database Security and Secure Data Management:** • Introduction to database security • Database threats and vulnerabilities • Discretionary Access Control (DAC) • Mandatory Access Control (MAC) • Role-Based Access Control (RBAC) • Authentication and authorization mechanisms • SQL injection attacks and prevention • Database auditing and logging concepts • Basics of encryption in databases	**9**

Course Outcomes:

1. Explain database concepts, architectures, and data models.
2. Design relational databases using ER modeling and normalization techniques.
3. Write and execute SQL queries for efficient data definition and manipulation.
4. Analyze transaction management and PL/SQL programs for maintaining data consistency.
5. Assess database security risks and apply appropriate security mechanisms.

Text Book:

1. Database System Concepts, Silberschatz, Korth, Sudarshan, 4th Edition, McGraw Hill Publication.
2. Fundamentals of Database Systems, Ramez Elmasri, Shamkant B. Navathe, 4th Edition, Pearson Education
3. PL/SQL User's Guide and Reference, Oracle Database 10g Release 1 (10.1), 2003.

Reference Books:

1. Database Security and Auditing, Hassan A. Afyouni, India Edition, CENGAGE Learning, 2009.
2. Database Security, Pierangela Samarati, Sabrina De Capitani di Vimercati, Springer.
3. Handbook of Database Security: Applications and Trends, Michael Gertz and Sushil Jajodia, Springer
4. NIST SP 800-53 - Security and Privacy Controls by the National Institute of Standards and Technology (NIST)

Web References:

1. https://nptel.ac.in
2. https://csrc.nist.gov

App References:

1. Oracle Live SQL
2. Learn DBMS
3. PL/SQL Tutorial

Syllabus Coverage from text /reference book & web/app reference:

Unit #	Chapter Numbers
1	Text Book 1 Chapter 1, 6; Text Book 2 Chapter 2
2	Text Book 1 Chapter 2, 7; Text Book 2 Chapter 3, 4, 10
3	Text Book 1 Chapter 4, 6, 15; Text Book 2 Chapter 8
4	Text Book 1 Chapter 1, 15; Text Book 2 Chapter 17; Text Book 3 Chapter 1, 4, 6
5	Text Book 1 Chapter 6; Text Book 2 Chapter 23

FACULTY OF COMPUTER APPLICATIONS
Master of Science – Cyber Security and Cyber Law

PRACTICALS

<table>
<tr><th>Unit No</th><th>List of Practicals</th></tr>
<tr><td>1</td><td>

Task 1: Create the following tables:

a) Branch

CREATE TABLE BRANCH (BNAME VARCHAR2(18) PRIMARY KEY, CITY VARCHAR2(18) NOT NULL);

b) Customers

CREATE TABLE CUSTOMERS (CNAME VARCHAR2(19) PRIMARY KEY, CITY VARCHAR2(18) NOT NULL);

c) Deposit

CREATE TABLE DEPOSIT (ACTNO VARCHAR2(5) PRIMARY KEY, CNAME VARCHAR2(18) NOT NULL, BNAME VARCHAR2(18) NOT NULL, AMOUNT NUMBER (8,2) CHECK (AMOUNT > 0), ADATE DATE NOT NULL, CONSTRAINT fk_dep_cust FOREIGN KEY (CNAME) REFERENCES CUSTOMERS(CNAME), CONSTRAINT fk_dep_branch FOREIGN KEY (BNAME) REFERENCES BRANCH(BNAME));

d) Borrow

CREATE TABLE BORROW (LOANNO VARCHAR2(5) PRIMARY KEY, CNAME VARCHAR2(18) NOT NULL, BNAME VARCHAR2(18) NOT NULL, AMOUNT NUMBER (8,2) CHECK (AMOUNT > 0), CONSTRAINT fk_bor_cust FOREIGN KEY (CNAME) REFERENCES CUSTOMERS(CNAME), CONSTRAINT fk_bor_branch FOREIGN KEY (BNAME) REFERENCES BRANCH(BNAME));

Task 2: Insert data into all tables:

Data for the branch table:

BNAME	CITY
ANDHERI	MUMBAI
NAVRANGPURA	AHMEDABAD
MGROAD	BENGALURU
SALTLAKE	KOLKATA
TNSQUARE	CHENNAI

</td></tr>
</table>

Data for the customers table:

CNAME	CITY
AMIT	MUMBAI
NEHA	AHMEDABAD
RAHUL	BENGALURU
PRIYA	CHENNAI
SANJAY	KOLKATA

Data for the deposit table:

ACTNO	CNAME	BNAME	AMOUNT	ADATE
D101	AMIT	ANDHERI	45000.00	15-JAN-23
D102	NEHA	NAVRANGPURA	32000.50	10-FEB-23
D103	RAHUL	MGROAD	60000.00	05-MAR-23
D104	PRIYA	TNSQUARE	28000.75	20-APR-23
D105	SANJAY	SALTLAKE	75000.00	01-MAY-23

Data for the borrow table:

LOANNO	CNAME	BNAME	AMOUNT
L201	AMIT	ANDHERI	150000.00
L202	NEHA	NAVRANGPURA	200000.00
L203	RAHUL	MGROAD	175000.50
L204	PRIYA	TNSQUARE	120000.00
L205	SANJAY	SALTLAKE	250000.00

Task 3: Execute queries

1. Display all tables in the database.
2. Describe the structure of the DEPOSIT table.
3. Display the current database user.
4. Insert a new customer.
5. Insert a new branch.
6. Insert a deposit record.

	7. Update the deposit amount of D103. 8. Update the customer city from "Chennai" to "Patna." 9. Delete a deposit record where the balance is less than 30000. 10. Delete customer without deposits. 11. Create a new table for audit logging. 12. Add an email column to CUSTOMERS. 13. Modify the CITY column size in BRANCH. 14. Rename table AUDIT_LOG to SYSTEM_LOG. 15. Drop the EMAIL column from CUSTOMERS. 16. Drop the SYSTEM_LOG table. 17. Display all customers. 18. Display all branches in Mumbai. 19. Display deposits greater than 40,000. 20. List customers with their deposit amounts. 21. Display customer name and branch city. 22. Calculate the total deposit amount. 23. Calculate the average loan amount. 24. Count the customers who have both a deposit and a loan. 25. Display the customer details with the highest loan amount. 26. Create a new database user. 27. Create a savepoint and rollback. 28. Grant SELECT privilege on the DEPOSIT table. 29. Verify SELECT privilege (run as created db user). 30. Revoke SELECT privilege from the user.
2	**Task 1: Create the following tables:** **a) Students** **Column Name** | **Data Type** | **Constraints** STUD_ID | INT | PRIMARY KEY SNAME | VARCHAR(30) | NOT NULL DEPARTMENT | VARCHAR(20) | — EMAIL | VARCHAR(40) | UNIQUE **b) Courses** **Column Name** | **Data Type** | **Constraints** COURSE_ID | INT | PRIMARY KEY

COURSE_NAME	VARCHAR(30)	NOT NULL
CREDITS	INT	CHECK (CREDITS > 0)

c) Enrollments

Column Name	**Data Type**	**Constraints**
ENROLL_ID	INT	PRIMARY KEY
STUD_ID	INT	FOREIGN KEY → STUDENTS(STUD_ID)
COURSE_ID	INT	FOREIGN KEY → COURSES(COURSE_ID)
MARKS	INT	CHECK (MARKS BETWEEN 0 AND 100)

Task 2: Insert data into all tables:

a) Students

STUD_ID	**SNAME**	**DEPARTMENT**	**EMAIL**
1	Arjun	CSE	arjun@gmail.com
2	Meera	IT	meera@gmail.com
3	Karan	CSE	karan@gmail.com
4	Anita	ECE	anita@gmail.com
5	Rohit	IT	rohit@gmail.com

b) Courses

COURSE_ID	**COURSE_NAME**	**CREDITS**
201	Database Systems	4
202	Operating Systems	3
203	Computer Networks	4
204	Cyber Security	3

c) Accounts

ENROLL_ID	STUD_ID	COURSE_ID	MARKS
301	1	201	78
302	1	204	85
303	2	201	66
304	3	202	72
305	4	203	90
306	5	204	55

Task 3: Execute queries

1. Attempt to insert an enrollment record with marks exceeding the permitted range and observe the system response.
2. Attempt to insert a student record using an already existing student ID and record the outcome.
3. Attempt to insert a course record without specifying the course name and verify whether the operation succeeds.
4. Attempt to enroll a student by providing a course ID that does not exist in the COURSES table.
5. Insert a new course with an invalid credit value and verify whether the database accepts the entry.
6. Display the details of students belonging to the CSE department.
7. Display students who belong to the IT department and have a student ID greater than 2.
8. Display enrollment records where the marks obtained are greater than 80, or the course ID is 204.
9. Display enrollment records where marks fall within the range of 60 to 80.
10. Display student records whose names start with the letter ‘A’.
11. Display students who do not belong to the ECE department.
12. Display courses whose credit value is greater than or equal to

	4. 13. Display enrollment records where marks are less than 70. 14. Display the average marks obtained by students in each course. 15. Display the total number of students enrolled in each course. 16. Display course IDs where the average marks obtained are greater than 70. 17. Display the maximum marks obtained in each course. 18. Display the number of students grouped by department. 19. Display student details sorted in ascending order of student name. 20. Display enrollment records sorted in descending order of marks. 21. Display courses sorted in ascending order of credit value. 22. Display the course IDs and average marks for courses where either the average marks are greater than 75 or the number of enrolled students is more than 2. 23. Display departments where the total number of students is greater than or equal to 2, sorted in descending order of student count. 24. Display course IDs where the difference between the highest and lowest marks is greater than 15. 25. Display course IDs where the maximum marks are less than 90 but the minimum marks are greater than 50. 26. Display departments where the average marks of students are higher than the overall average marks across all enrollments. 27. Display course IDs and total enrollment count, sorted by highest enrollment first, and then by course ID in ascending order. 28. Display departments along with the number of students in each department, sorted in descending order of student count. 29. Display departments along with the average marks of students, sorted in descending order of average marks. 30. Display course IDs where the minimum marks obtained by students are greater than or equal to 60.

3

Task 1: Create the following tables:

a) Employees:

Column Name	Data Type	Constraints
EMP_ID	INT	PRIMARY KEY
EMP_NAME	VARCHAR(30)	NOT NULL
DEPT_ID	INT	—
EMAIL	VARCHAR(40)	UNIQUE
SALARY	DECIMAL(9,2)	—

b) Departments:

Column Name	Data Type	Constraints
DEPT_ID	INT	PRIMARY KEY
DEPT_NAME	VARCHAR(30)	NOT NULL
LOCATION	VARCHAR(30)	—

c) Projects:

Column Name	Data Type	Constraints
PROJ_ID	INT	PRIMARY KEY
PROJ_NAME	VARCHAR(40)	NOT NULL
DEPT_ID	INT	FOREIGN KEY → DEPARTMENTS(DEPT_ID)
BUDGET	DECIMAL(10)	—

Task 2: Insert data into all tables:

Employees:

EMP_ID	EMP_NAME	DEPT_ID	EMAIL	SALARY
1	Ravi	10	ravi@org.com	60000
2	Sneha	20	sneha@org.com	52000
3	Aman	10	aman@org.com	58000

4	Divya	30	divya@org.com	65000
5	Nikhil	NULL	nikhil@org.com	48000

Departments:

DEPT_ID	DEPT_NAME	LOCATION
10	HR	Mumbai
20	Finance	Delhi
30	IT	Bengaluru
40	Legal	Chennai

Projects:

PROJ_ID	PROJ_NAME	DEPT_ID	BUDGET
101	Payroll System	20	800000
102	Recruitment App	10	500000
103	Security Upgrade	30	1200000
104	Compliance Tool	40	300000

Task 3: Execute queries

1. Display employee names along with their department names.
2. Display all employees, including those not assigned to any department.
3. Display all departments, including those without employees.
4. Display the employee name and project name for employees working in the same department as the project.
5. Display department name and number of employees using JOIN and GROUP BY.
6. Display employees working in departments located in Mumbai.
7. Display project names and department locations using INNER JOIN.
8. Display employees and departments using LEFT JOIN.
9. Display departments and employees using RIGHT JOIN.
10. Display employee name and department name using WHERE-based join syntax.

	11. Display employees who are not assigned to any department. 12. Display all possible combinations of employees and departments. 13. Display departments that have projects assigned. 14. Display employees whose department has at least one project. 15. Display employee names in uppercase. 16. Display the length of each employee's name. 17. Display the highest salary among employees. 18. Display the average salary of employees department-wise. 19. Display the total budget allocated to each department. 20. Display the current system date. 21. Round off employee salaries to the nearest integer. 22. Display department-wise minimum and maximum salary. 23. Count the total number of employees. 24. Display employee names and their email domains using string functions. 25. Create a view that exposes only the employee name and department. 26. Write and execute a parameterized SQL query to retrieve employee details based on employee ID instead of using a hard-coded value. 27. Write a SQL query that retrieves employee details using direct user input in the WHERE clause, and then rewrite the same query using a prepared statement to prevent SQL injection. 28. Create a database user and grant a view permission only on employee name and department information. 29. Create a role for "HR_VIEWER", grant it SELECT access on a restricted employee view, assign the role to a user, and execute a query to retrieve employee details through the role. 30. Create an audit table and write an INSERT query to log employee data access, and retrieve audit records to verify logging.
4	**Programs of PL/SQL:** 1. Write a PL/SQL block to accept two numbers and display their sum using DBMS_OUTPUT. 2. Write a PL/SQL program to calculate the factorial of a given number using a loop construct. 3. Write a PL/SQL block to display the multiplication table of a given number.

	4. Write a PL/SQL program to determine whether a given number is odd or even. 5. Write a PL/SQL program to compute and display the square and cube of a given number. 6. Write a PL/SQL program to swap two numbers using a temporary variable. 7. Write a PL/SQL block to display numbers from 1 to 100 using a FOR loop. 8. Write a PL/SQL program that demonstrates the use of %TYPE and %ROWTYPE attributes with a table. 9. Write a stored procedure without parameters that updates employee salaries by a fixed percentage. 10. Write a stored procedure that increases employee salary for a given department number using an IN parameter. 11. Write a stored procedure to check whether a given employee ID exists in the EMP table using IN and OUT parameters. 12. Write a PL/SQL block to invoke the employee search procedure and display appropriate messages. 13. Write a stored procedure that inserts a new employee record into the EMP table using input parameters. 14. Write a stored function that returns the square of a given number and invoke it from a SELECT statement. 15. Write a stored function that returns the total salary paid to employees of a given department. 16. Write a stored function that returns the number of employees working in a specified department. 17. Write a row-level trigger to store deleted employee records into an audit table. 18. Write a trigger that records old and new salary values whenever an employee salary is updated. 19. Write a BEFORE INSERT trigger to restrict insertion of employee records with salary greater than ₹50,000. 20. Write a trigger to prevent INSERT, UPDATE, or DELETE operations on the EMP table on Sundays.
5	**SQL Injection:** 1. Demonstrate SQL Injection vulnerabilities and their prevention by executing insecure and secure SQL queries on a test database using XAMPP (MySQL), and to understand the role of prepared statements, access control, auditing, and secure query design in database security.

	2. Demonstrate SQL Injection vulnerabilities and their mitigation by performing authentication bypass and data extraction attacks on the SQL Injection module of DVWA deployed locally using XAMPP. 3. Identify and exploit SQL Injection flaws in OWASP Juice Shop and analyze how improper input handling and access control can lead to unauthorized database access. 4. Perform different SQL Injection techniques on bWAPP in a controlled local environment and understand the impact of insecure SQL query construction. 5. Analyze various SQL Injection attack patterns using SQLi Labs and observe how query logic manipulation leads to information disclosure.