1. SQL Queries:

• Design and Develop SQL DDL statements which demonstrate the use of SQL objects such as Table, View, Index, Sequence, Synonym, different constraints etc.

• Write at least 10 SQL queries on the suitable database application using SQL DML statements.

2. SQL Queries – all types of Join, Sub-Query and View:

Write at least10 SQL queries for suitable database application using SQL DML statements. Note: Instructor will design the queries which demonstrate the use of concepts
like all types of Join ,Sub-Query and View.

3. MongoDB Queries:

Design and Develop MongoDB Queries using CRUD operations. (Use CRUD operations)

4. Unnamed PL/SQLcode block: Use of Control structure and Exception handling is mandatory.

Suggested Problem statement: Consider Tables:

1. Borrower (Roll_no, Name, Date_of_Issue, Name_of_Book, Status)
2. Fine (Roll_no, Date, Amt) Accept Roll_no and Name_of_Book from user.
3. Check the number of days (from Date_of_Issue). •
4. If days are between 15 to 30 then fine amount will be Rs 5per day.
5. If no. of days>30, per day fine will be Rs 50 per day and for days less than 30, Rs. 5 per day.
6. After submitting the book, status will change from I to R.
7. If condition of fine is true, then details will be stored into fine table.
8. Also handles the exception by named exception handler or user define exception handler.

5. Cursors: (All types: Implicit, Explicit, Cursor FOR Loop, Parameterized Cursor) Write a PL/SQL block of code using parameterized Cursor that will merge the data available in the newly created table N_Roll_Call with the data available in the table O_Roll_Call. If the data in the first table already exists in the second table then that data should be skipped.

6. Database Connectivity: Write a program to implement MySQL/Oracle database connectivity with any front end language to implement Database navigation operations (add, delete, edit etc.)

7. Implement depth first search algorithm and Breadth First Search algorithm. Use an undirected graph.

8. Implement A star (A*) algorithm for any game search problem.

9. Implement Alpha-Beta Tree search for any game search problem.

10. Implement a solution for a Constraint Satisfaction Problem using Branch and Bound and Backtracking for n-queens problem or a graph coloring problem.

11. Implement Greedy search algorithm for any of the following application:

- Selection Sort
- Prim's Minimal Spanning Tree Algorithm

12. Implement Greedy search algorithm for any of the following application:

- Selection Sort
- Kruskal's Minimal Spanning Tree Algorithm

13. Implement Greedy search algorithm for any of the following application:

- Dijkstra's Minimal Spanning Tree Algorithm.