List of Assignments

Suggested List of Laboratory Experiments/Assignments Assignments from all Groups (A, B, C) are compulsory	
Sr. No.	**Group A: SQL and PL/SQL**
1.	ER Modeling and Normalization: Decide a case study related to real time application in group of 2-3 students and formulate a problem statement for application to be developed. Propose a Conceptual Design using ER features using tools like ERD plus, ER Win etc. (Identifying entities, relationships between entities, attributes, keys, cardinalities, generalization, specialization etc.) Convert the ER diagram into relational tables and normalize Relational data model. Note: Student groups are required to continue same problem statement throughout all the assignments in order to design and develop an application as a part Mini Project. Further assignments will be useful for students to develop a backend for system. To design front end interface students should use the different concepts learnt in the other subjects also.
2.	SQL Queries: a. Design and Develop SQL DDL statements which demonstrate the use of SQL objects such as Table, View, Index, Sequence, Synonym, different constraints etc. b. Write at least 10 SQL queries on the suitable database application using SQL DML statements. C. Note: Instructor will design the queries which demonstrate the use of concepts like Insert, Select, Update, Delete with operators, functions, and set operator etc.
3.	SQL Queries - all types of Join, Sub-Query and View: Write at least 10 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
4.	Unnamed PL/SQL code block: Use of Control structure and Exception handling is mandatory. Suggested Problem statement: Consider Tables: 1. Borrower(Roll_no, Name, DateofIssue, NameofBook, Status)

	2. Fine(Roll_no,Date,Amt) □ Accept Roll_no and NameofBook from user. □ Check the number of days (from date of issue). □ If days are between 15 to 30 then fine amount will be Rs 5per day. □ If no. of days>30, per day fine will be Rs 50 per day and for days less than 30, Rs. 5 per day. □ After submitting the book, status will change from I to R. □ If condition of fine is true, then details will be stored into fine table. □ Also handles the exception by named exception handler or user define exception handler. OR Write a PL/SQL code block to calculate the area of a circle for a value of radius varying from 5 to 9. Store the radius and the corresponding values of calculated area in an empty table named areas, consisting of two columns, radius and area. Note: Instructor will frame the problem statement for writing PL/SQL block in line with above statement.
5.	Named PL/SQL Block: PL/SQL Stored Procedure and Stored Function. Write a Stored Procedure namely proc_Grade for the categorization of student. If marks scored by students in examination is <=1500 and marks>=990 then student will be placed in distinction category if marks scored are between 989 and 900 category is first class, if marks 899 and 825 category is Higher Second Class. Write a PL/SQL block to use procedure created with above requirement. Stud_Marks(name, total_marks) Result(Roll,Name, Class) Note: Instructor will frame the problem statement for writing stored procedure and Function in line with above statement.
6.	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_RollCall with the data available in the table O_RollCall. If the data in the first table already exist in the second table then that data should be skipped. Note: Instructor will frame the problem statement for writing PL/SQL block using all types of Cursors in line with above statement.

7	Unnamed PL/SQL code block: Use of Control structure and Exception handling is mandatory. Suggested Problem statement: Consider Tables: 1. Borrower(Roll_no, Name, DateofIssue, NameofBook, Status) 2. Fine(Roll_no,Date,Amt) □ Accept Roll_no and NameofBook from user. □ Check the number of days (from date of issue). □ If days are between 15 to 30 then fine amount will be Rs 5per day. □ If no. of days>30, per day fine will be Rs 50 per day and for days less than 30, Rs. 5 per day. □ After submitting the book, status will change from I to R. □ If condition of fine is true, then details will be stored into fine table. □ Also handles the exception by named exception handler or user define exception handler. OR Write a PL/SQL code block to calculate the area of a circle for a value of radius varying from 5 to 9. Store the radius and the corresponding values of calculated area in an empty table named areas, consisting of two columns, radius and area. Note: Instructor will frame the problem statement for writing PL/SQL block in line with above statement.
8.	Named PL/SQL Block: PL/SQL Stored Procedure and Stored Function. Write a Stored Procedure namely proc_Grade for the categorization of student. If marks scored by students in examination is <=1500 and marks>=990 then student will be placed in distinction category if marks scored are between 989 and 900 category is first class, if marks 899 and 825 category is Higher Second Class. Write a PL/SQL block to use procedure created with above requirement. Stud_Marks(name, total_marks) Result(Roll,Name, Class) Note: Instructor will frame the problem statement for writing stored procedure and Function in line with above statement.
Group B: NoSQL Databases	
1.	MongoDB Queries: Design and Develop MongoDB Queries using CRUD operations. (Use CRUD operations,

	SAVE method, logical operators etc.).
2.	MongoDB - Aggregation and Indexing: Design and Develop MongoDB Queries using aggregation and indexing with suitable example using MongoDB.
3.	MongoDB - Map reduces operations: Implement Map reduces operation with suitable example using MongoDB.
4.	Database Connectivity: Write a program to implement MongoDB database connectivity with any front end language to implement Database navigation operations (add, delete, edit etc.)
Group C: Mini Project	
1.	Using the database concepts covered in Group A and Group B, develop an application with following details: 1. Follow the same problem statement decided in Assignment -1 of Group A. 2. Follow the Software Development Life cycle and other concepts learnt in Software Engineering Course throughout the implementation. 3. Develop application considering: □ Front End : Java/Perl/PHP/Python/Ruby/.net/any other language □ Backend : MongoDB/MySQL/Oracle 4. Test and validate application using Manual/Automation testing. 5. Student should develop application in group of 2-3 students and submit the Project Report which will consist of documentation related to different phases of Software Development Life Cycle: □ Title of the Project, Abstract, Introduction □ Software Requirement Specification □ Conceptual Design using ER features, Relational Model in appropriate Normalize form □ Graphical User Interface, Source Code □ Testing document □ Conclusion.

GROUP A

Assignment 1

Title: Study of open sources Relational Databases:MySQL

Objectives:

- Install MySQL server and client on different operating systems (Windows, Linux, macOS).
- Understand and work with basic features of MySQL (creating databases, tables, querying, etc.).
- Explore DBMS models, architecture, and features.
- Analyze applications of DBMS in real-world scenarios.
- Research and discuss trends & future directions in DBMS.

Outcomes: Students will be able to learn concepts of relational databases.

Hardware requirements: Any CPU with Pentium Processor or similar, 256 MB AMor more, 1 GB Hard Disk or more.

Software requirements: Operating System, MySQL.

Theory

Operating System	Steps to Install MySQL
Linux (Ubuntu / Debian-based)	1. Update package index: sudo apt update 2. Install mysql server: sudo apt install mysql-server 3. After installation, run sudo systemctl start mysql, sudo systemctl enable mysql to start at boot. 4. Secure the installation via: sudo mysql_secure_installation (set root password, remove test DB, disallow remote root login etc.). 5. Login: sudo mysql -u root -p. 6. (Optional) Setup remote access by adjusting mysqld.cnf / my.cnf to bind-address etc.
Linux (RHEL / CentOS / Fedora / yum-based)	Use MySQL's yum repository or RPM packages. Example: add MySQL YUM repo, then sudo yum install mysql-server, then same steps for starting service, securing installation.
macOS	Download native package installer (DMG) from MySQL site; run installer. Alternatively, use compressed tar archive or generic binaries. Then set up MySQL launch daemon or startup via System Preferences or using launchd. Run the "mysql_secure_installation" to configure.

Operating System	Steps to Install MySQL
Windows	Download MySQL Installer (MSI) from MySQL official site. Run the installer wizard. Choose server, client, tools (Workbench etc.). Set root password, configure port, startup options. Allow service to start automatically. Configure firewall if needed.

DBMS Models (Database Models / Architectures)

Students should understand these models and compare them:

- **Hierarchical Model** — tree-structured; parent/child relationships.
- **Network Model** — allows many-to-many relationships; more flexible than hierarchical.
- **Relational Model** — data stored in tables (relations), SQL queries, strict schema etc. (this is what MySQL uses).
- **Object-Oriented Model** — data stored as objects; encapsulation; more natural for some programming languages.
- **Object-Relational Model** — relational with added object features.
- **NoSQL Models**: document stores, key-value stores, wide-column, graph databases.
- **Multi-Model DBMS** — supporting more than one model in a single system (e.g. relational + document + graph support). Reference: multi-model trends.

Applications of DBMS

Some areas where relational DBMSs like MySQL are widely used:

- Web applications (user accounts, posts, comments, e-commerce).
- Banking, finance (transaction management, accounts).
- Enterprise resource planning (ERP), inventory management.
- Healthcare systems (patient records, appointments).
- Data warehousing, reporting, business intelligence.
- Content management systems (CMS).

MySQL is popular in many open-source stacks (e.g. LAMP: Linux Apache MySQL PHP), for web hosting, online apps etc.

Future Trends of DBMS

Some of the trends and future directions in database management systems:

- Cloud-native databases, serverless databases – DBaaS (Database as a Service).
- Autonomous / self-driving databases: automatic tuning, patching, scaling.
- Multi-model DBMS (supporting relational + NoSQL + graph etc.).
- Real-time analytics, streaming data processing.
- Increasing role of AI/ML in DBMS: anomaly detection, query optimization, predictive maintenance.
- Edge / distributed databases; hybrid & multi-cloud deployments.
- Emphasis on security, privacy, data governance, regulatory compliance.

Conclusion: In this assignment, we have studied concept of relational databases, MySQL DBMS and steps to install MySQL on Ubuntu O.S.

Assignment 2

SQL Queries:

Aim:

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

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

Objective:

Understand and practice creation of SQL database objects (tables, views, indexes, sequences, synonyms) using DDL.

Learn how to enforce data integrity and constraints (primary key, foreign key, unique, check, not null, default etc.).

Get hands-on experience using DML statements (INSERT, UPDATE, DELETE, SELECT) to query and manipulate data.

Outcome:

Design a relational schema with proper constraints to ensure data integrity.

Hardware requirements: Any CPU with Pentium Processor or similar, 2 GB RAM or more, 1 GB Hard Disk or more.

Software requirements: Fedora 20/ Windows Operating System, MySQL.

Theory

```
-- 1. Create database/schema
CREATE DATABASE IF NOT EXISTS bookstore_db
USE bookstore_db;


-- 2. Create `Publisher` table
CREATE TABLE Publisher (
   publisher_id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
   name VARCHAR(100) NOT NULL UNIQUE,
   country VARCHAR(50),
   established_year INT
      CHECK (established_year >= 1800 AND established_year <= YEAR(CURDATE()))
) ENGINE = InnoDB;
```

```
-- 3. Create `Author` table
CREATE TABLE Author (
   author_id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
   first_name VARCHAR(50) NOT NULL,
   last_name VARCHAR(50) NOT NULL,
   birth_date DATE,
   UNIQUE (first_name, last_name, birth_date)
) ENGINE = InnoDB;

-- 4. Create `Book` table with foreign keys to Publisher & Author
CREATE TABLE Book (
   book_id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
   title VARCHAR(200) NOT NULL,
   publisher_id INT UNSIGNED NOT NULL,
   author_id INT UNSIGNED NOT NULL,
   publication_date DATE,
   price DECIMAL(10,2) DEFAULT 0.00,
   stock_quantity INT UNSIGNED DEFAULT 0,
   FOREIGN KEY (publisher_id) REFERENCES Publisher(publisher_id)
      ON DELETE CASCADE
      ON UPDATE CASCADE,
   FOREIGN KEY (author_id) REFERENCES Author(author_id)
      ON DELETE SET NULL
      ON UPDATE CASCADE
) ENGINE = InnoDB;

-- 5. Create `Customer` table
CREATE TABLE Customer (
   customer_id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
   name VARCHAR(100) NOT NULL,
   email VARCHAR(100) NOT NULL UNIQUE,
   phone VARCHAR(20),
   signup_date DATETIME DEFAULT CURRENT_TIMESTAMP,
   address VARCHAR(200) DEFAULT 'Unknown'
) ENGINE = InnoDB;

-- 6. Create `Orders` table
CREATE TABLE Orders (
   order_id BIGINT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
   customer_id INT UNSIGNED,
```

```
    order_date DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
    status ENUM('Pending','Shipped','Delivered','Cancelled') NOT NULL DEFAULT
'Pending',
    total_amount DECIMAL(12,2) NOT NULL DEFAULT 0.00,
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
        ON DELETE SET NULL
) ENGINE = InnoDB;

-- 7. Create `OrderItems` table (child of Orders, Book)
CREATE TABLE OrderItems (
    order_item_id BIGINT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    order_id BIGINT UNSIGNED NOT NULL,
    book_id INT UNSIGNED NOT NULL,
    quantity INT UNSIGNED NOT NULL DEFAULT 1,
    unit_price DECIMAL(10,2) NOT NULL,
    FOREIGN KEY (order_id) REFERENCES Orders(order_id)
        ON DELETE CASCADE,
    FOREIGN KEY (book_id) REFERENCES Book(book_id)
        ON DELETE RESTRICT
) ENGINE = InnoDB;

-- 8. Create Indexes on Book table
CREATE INDEX idx_book_title ON Book(title);
CREATE INDEX idx_book_price ON Book(price);

-- 9. Create a View for book full information
CREATE VIEW vw_book_fullinfo AS
SELECT
    b.book_id,
    b.title,
    CONCAT(a.first_name, ' ', a.last_name) AS author_name,
    p.name AS publisher_name,
    b.publication_date,
    b.price,
    b.stock_quantity
FROM Book b
JOIN Author a ON b.author_id = a.author_id
JOIN Publisher p ON b.publisher_id = p.publisher_id;

-- 10. Emulate Sequence (since MySQL lacks native SEQUENCE) using a helper table and
function
```

```
DELIMITER //

-- Helper table to simulate a sequence
CREATE TABLE sequence_table (
   seq_name VARCHAR(50) PRIMARY KEY,
   current_value BIGINT UNSIGNED NOT NULL
);

-- Initialize a sequence
INSERT INTO sequence_table(seq_name, current_value) VALUES ('order_seq', 0);

-- Function to get next sequence value
CREATE FUNCTION get_next_order_seq() RETURNS BIGINT UNSIGNED
DETERMINISTIC
BEGIN
   UPDATE sequence_table
     SET current_value = current_value + 1
     WHERE seq_name = 'order_seq';
   RETURN (SELECT current_value FROM sequence_table WHERE seq_name =
'order_seq');
END;
//
```

Conclusion: Awareness of Different SQL objects such as Table, View, Index, Sequence, and Synonym.

Assignment -3

SQL Queries - all types of Join, Sub-Query and View

Aim: SQL Queries - all types of Join, Sub-Query and View:

Write at least 10 SQL queries for suitable database application using SQL DML statements.

Objective: To understand the concept of DML statement like Insert, Select, Update, operators and set operator.

Outcome: After This students will be able to fire basic dml commands such as insert, select, update, delete with operators, functions and set operators.

Requirements:

Software Requirement : Maria DB, Fedora 20, Terminal

Hardware Requirement : Minimum 2GB Ram

Theory:

DATA MANIPULATION LANGUAGE (DML):

After the database structure is defined with DDL, database administrators and users can utilize the Data Manipulation Language to insert, retrieve and modify the data contained within it.

Types of Joins: INNER, LEFT, RIGHT, FULL, CROSS, SELF. Differences, how NULLs are handled, when rows are or aren't included.

Types of Sub-queries: single-row, multiple-row, correlated, nested, multiple-column.

Views: what they are, advantages (modularity, abstraction, reuse), limitations.

Set Operation in SQL

SQL supports few Set operations to be performed on table data. These are used to get meaningful results from data, under different special conditions.

Conclusion: Implemented all SQL DML Commands like Insert, Select, Update, Delete with operators, functions, and set operator.