MySQL Course: Beginner to Advanced

Introduction to MySQL

- **What is MySQL?**
  - Definition and importance of MySQL.
  - Use cases of MySQL in real-world applications.
  - Overview of relational databases.
- **Installing MySQL**
  - Step-by-step guide for Windows, macOS, and Linux.
  - Setting up MySQL Server and Workbench.
  - Verifying installation.
- **Basic MySQL Commands**
  - Connecting to MySQL server.
  - Understanding MySQL shell and MySQL Workbench.

Section 1: Getting Started with MySQL

1.1 Basics of SQL

- Introduction to SQL and its importance.
- Types of SQL commands: DDL, DML, DCL, TCL.
- SQL Syntax rules.

1.2 Database and Tables

- Creating a database:

```
CREATE DATABASE school;
```

- Viewing existing databases:

```
SHOW DATABASES;
```

- Selecting a database:

```
USE school;
```

- Creating tables:

```
CREATE TABLE students (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100),
    age INT,
    grade VARCHAR(10)
);
```

- Viewing table structure:

```
DESCRIBE students;
```

1.3 Basic CRUD Operations

- Inserting data:

```
INSERT INTO students (name, age, grade) VALUES ('John Doe', 15,
'10th');
```

- Reading data:

```
SELECT * FROM students;
```

- Updating data:

```
UPDATE students SET age = 16 WHERE name = 'John Doe';
```

- Deleting data:

```
DELETE FROM students WHERE name = 'John Doe';
```

Section 2: Intermediate MySQL

2.1 Data Types

- Overview of MySQL data types:
  - Numeric: INT, FLOAT, DOUBLE.
  - String: CHAR, VARCHAR, TEXT.
  - Date and Time: DATE, TIME, DATETIME, TIMESTAMP.
- Choosing the right data type for columns.

2.2 Constraints and Indexes

- Constraints:
  - NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY.
  - Example:

```
ALTER TABLE students ADD CONSTRAINT unique_name UNIQUE
(name);
```

- Indexes:
  - Creating indexes for performance optimization:

```
CREATE INDEX idx_name ON students (name);
```

  - Viewing indexes:

```
SHOW INDEX FROM students;
```

2.3 Joins

- Understanding JOIN operations:
  - INNER JOIN:

```
SELECT students.name, grades.subject
FROM students
INNER JOIN grades ON students.id = grades.student_id;
```

    - LEFT JOIN, RIGHT JOIN, FULL JOIN.
- Use cases and examples.

2.4 Aggregate Functions

- Functions: COUNT, SUM, AVG, MAX, MIN.
- Grouping data with GROUP BY:

```
SELECT grade, COUNT(*)
FROM students
GROUP BY grade;
```

- Filtering grouped data with HAVING.

Section 3: Advanced MySQL

3.1 Stored Procedures and Functions

- Creating stored procedures:

```
DELIMITER //
CREATE PROCEDURE GetStudents()
BEGIN
    SELECT * FROM students;
END //
DELIMITER ;
```

- Executing stored procedures:

```
CALL GetStudents();
```

- Creating functions:

```
CREATE FUNCTION GetGradeCount(grade VARCHAR(10))
RETURNS INT
BEGIN
    DECLARE count INT;
    SELECT COUNT(*) INTO count FROM students WHERE grade =
grade;
    RETURN count;
END;
```

3.2 Triggers

- Creating triggers:

```
CREATE TRIGGER before_insert_students
BEFORE INSERT ON students
FOR EACH ROW
SET NEW.name = UPPER(NEW.name);
```

- Use cases for triggers.

3.3 Transactions

- Understanding transactions.
- Using START TRANSACTION, COMMIT, ROLLBACK:

```
START TRANSACTION;
UPDATE students SET grade = '11th' WHERE id = 1;
ROLLBACK;
```

- Ensuring data integrity.

3.4 Advanced Query Optimization

- Using EXPLAIN to analyze queries:

```
EXPLAIN SELECT * FROM students WHERE age > 15;
```

- Optimizing query performance with indexes and query restructuring.

3.5 Advanced Data Types and JSON

- Working with JSON in MySQL:

```
CREATE TABLE json_example (
    id INT AUTO_INCREMENT PRIMARY KEY,
    data JSON
);

INSERT INTO json_example (data) VALUES ('{"name": "John",
"age": 25}');
```

- Querying JSON data:

```
SELECT data->'$.name' AS name FROM json_example;
```

Section 4: MySQL Administration

4.1 User Management

- Creating users:

```
CREATE USER 'user1'@'localhost' IDENTIFIED BY 'password';
```

- Granting privileges:

```
GRANT ALL PRIVILEGES ON school.* TO 'user1'@'localhost';
```

- Viewing and revoking privileges.

4.2 Backup and Restore

- Exporting databases with `mysqldump`:

```
mysqldump -u root -p school > school_backup.sql
```

- Restoring databases:

```
mysql -u root -p school < school_backup.sql
```

4.3 Performance Monitoring

- Monitoring with `SHOW STATUS` and `SHOW PROCESSLIST`.
- Tuning MySQL configuration for performance.

Section 5: Real-World Projects

5.1 Building an E-commerce Database

- Designing tables for products, customers, orders, etc.
- Writing queries to handle inventory and order management.

5.2 Employee Management System

- Designing a relational schema.
- Writing stored procedures for common tasks.

Conclusion

- Summary of key concepts.
- Best practices for working with MySQL.
- Resources for further learning.

Appendix

- Useful SQL commands reference.
- Troubleshooting common MySQL errors.