```
mysql> CREATE DATABASE university;
Query OK, 1 row affected (0.01 sec)

mysql> USE university;
Database changed
mysql> -- Create Students table
mysql> CREATE TABLE Students (
    ->     StudentID INT PRIMARY KEY,
    ->     Name VARCHAR(50),
    ->     Age INT,
    ->     Major VARCHAR(50)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql>
mysql> -- Create Courses table
mysql> CREATE TABLE Courses (
    ->     CourseID INT PRIMARY KEY,
    ->     CourseName VARCHAR(50),
    ->     Credits INT
    -> );
Query OK, 0 rows affected (0.08 sec)

mysql>
mysql> -- Create Enrollments table
mysql> CREATE TABLE Enrollments (
    ->     EnrollmentID INT PRIMARY KEY,
    ->     StudentID INT,
    ->     CourseID INT,
    ->     Grade CHAR(2),
    ->     FOREIGN KEY (StudentID) REFERENCES Students(StudentID),
    ->     FOREIGN KEY (CourseID) REFERENCES Courses(CourseID)
    -> );
```

```
Query OK, 0 rows affected (0.07 sec)

mysql>
mysql> -- Create Departments table
mysql> CREATE TABLE Departments (
    ->     DeptID INT PRIMARY KEY,
    ->     DeptName VARCHAR(50)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> -- Insert sample students
mysql> INSERT INTO Students (StudentID, Name, Age, Major) VALUES
    -> (1, 'Alice', 20, 'Computer Science'),
    -> (2, 'Bob', 18, 'Mathematics'),
    -> (3, 'Carol', 22, 'Computer Science'),
    -> (4, 'Dave', 16, 'Physics');
Query OK, 4 rows affected (0.02 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql>
mysql> -- Insert sample courses
mysql> INSERT INTO Courses (CourseID, CourseName, Credits) VALUES
    -> (101, 'Intro to CS', 3),
    -> (102, 'Calculus I', 4),
    -> (103, 'Physics I', 4);
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql>
mysql> -- Insert sample enrollments
mysql> INSERT INTO Enrollments (EnrollmentID, StudentID, CourseID, Grade)
VALUES
    -> (1001, 1, 101, 'A'),
    -> (1002, 2, 102, 'B'),
```

```
    -> (1003, 3, 101, 'A');
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM Students;
+-----------+-------+------+-----------------+
| StudentID | Name  | Age  | Major           |
+-----------+-------+------+-----------------+
|         1 | Alice |   20 | Computer Science |
|         2 | Bob   |   18 | Mathematics     |
|         3 | Carol |   22 | Computer Science |
|         4 | Dave  |   16 | Physics         |
+-----------+-------+------+-----------------+
4 rows in set (0.00 sec)

mysql> SELECT * FROM Courses;
+----------+------------+---------+
| CourseID | CourseName  | Credits |
+----------+------------+---------+
|      101 | Intro to CS |       3 |
|      102 | Calculus I  |       4 |
|      103 | Physics I   |       4 |
+----------+------------+---------+
3 rows in set (0.00 sec)

mysql> SELECT * FROM Enrollments;
+--------------+-----------+----------+-------+
| EnrollmentID | StudentID | CourseID | Grade |
+--------------+-----------+----------+-------+
|         1001 |         1 |      101 | A     |
|         1002 |         2 |      102 | B     |
|         1003 |         3 |      101 | A     |
+--------------+-----------+----------+-------+
3 rows in set (0.00 sec)
```

```
mysql> -- 1) Add a new column "Email" to Students
mysql> ALTER TABLE Students ADD Email VARCHAR(100);
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql>
mysql> -- 2) Update Alice’s major from 'Computer Science' to 'Data Science'
mysql> UPDATE Students SET Major = 'Data Science' WHERE StudentID = 1;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql>
mysql> -- 3) Delete students younger than 18 (this will remove Dave, age 16)
mysql> DELETE FROM Students WHERE Age < 18;
Query OK, 1 row affected (0.01 sec)

mysql> SELECT * FROM Students;
+-----------+-------+------+-----------------+-------+
| StudentID | Name  | Age  | Major           | Email |
+-----------+-------+------+-----------------+-------+
|         1 | Alice |   20 | Data Science    | NULL  |
|         2 | Bob   |   18 | Mathematics     | NULL  |
|         3 | Carol |   22 | Computer Science | NULL  |
+-----------+-------+------+-----------------+-------+
3 rows in set (0.00 sec)

mysql> SELECT Name, Major FROM Students WHERE Age > 19;
+-------+-----------------+
| Name  | Major           |
+-------+-----------------+
| Alice | Data Science    |
| Carol | Computer Science |
+-------+-----------------+
```

```
2 rows in set (0.00 sec)

mysql> SELECT AVG(Age) AS AvgAge FROM Students;
+---------+
| AvgAge  |
+---------+
| 20.0000 |
+---------+
1 row in set (0.00 sec)

mysql> SELECT Major, COUNT(*) AS StudentCount
    -> FROM Students
    -> GROUP BY Major
    -> HAVING COUNT(*) > 1;
Empty set (0.00 sec)

mysql> SELECT * FROM Students WHERE Age > 20 AND Major = 'Computer
Science';
+-----------+-------+------+-----------------+-------+
| StudentID | Name  | Age  | Major           | Email |
+-----------+-------+------+-----------------+-------+
|         3 | Carol |   22 | Computer Science | NULL  |
+-----------+-------+------+-----------------+-------+
1 row in set (0.00 sec)

mysql> SELECT s.Name, e.Grade,
    ->        RANK() OVER (ORDER BY e.Grade DESC) AS RankInClass
    -> FROM Enrollments e
    -> JOIN Students s ON e.StudentID = s.StudentID;
+-------+-------+-------------+
| Name  | Grade | RankInClass |
+-------+-------+-------------+
| Bob   | B     |           1 |
| Alice | A     |           2 |
```

```
| Carol | A    |          2 |
+-------+-------+-------------+
3 rows in set (0.00 sec)

mysql> SELECT s.Name, c.CourseName
    -> FROM Students s
    -> INNER JOIN Enrollments e ON s.StudentID = e.StudentID
    -> INNER JOIN Courses c ON e.CourseID = c.CourseID;
+-------+-------------+
| Name  | CourseName  |
+-------+-------------+
| Alice | Intro to CS |
| Bob   | Calculus I  |
| Carol | Intro to CS |
+-------+-------------+
3 rows in set (0.00 sec)

mysql> DROP DATABASE IF EXISTS university;
Query OK, 4 rows affected (0.12 sec)

mysql> CREATE DATABASE university;
Query OK, 1 row affected (0.02 sec)

mysql> USE university;
Database changed
mysql> CREATE TABLE Students (
    ->     StudentID INT PRIMARY KEY,
    ->     Name VARCHAR(50),
    ->     Age INT,
    ->     Major VARCHAR(50)
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> CREATE TABLE Courses (
```

```
    ->     CourseID INT PRIMARY KEY,
    ->     CourseName VARCHAR(50),
    ->     Credits INT
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> CREATE TABLE Enrollments (
    ->     EnrollmentID INT PRIMARY KEY,
    ->     StudentID INT,
    ->     CourseID INT,
    ->     Grade CHAR(2),
    ->     FOREIGN KEY (StudentID) REFERENCES Students(StudentID),
    ->     FOREIGN KEY (CourseID) REFERENCES Courses(CourseID)
    -> );
Query OK, 0 rows affected (0.09 sec)

mysql> CREATE TABLE Departments (
    ->     DeptID INT PRIMARY KEY,
    ->     DeptName VARCHAR(50)
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> ALTER TABLE Students ADD Email VARCHAR(100);
Query OK, 0 rows affected (0.12 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> DROP TABLE Departments;
Query OK, 0 rows affected (0.05 sec)

mysql> -- Insert into Students
mysql> INSERT INTO Students (StudentID, Name, Age, Major, Email) VALUES
    -> (1, 'Alice', 20, 'Computer Science', 'alice@example.com'),
    -> (2, 'Bob', 18, 'Mathematics', 'bob@example.com'),
    -> (3, 'Carol', 22, 'Computer Science', 'carol@example.com'),
```

```
    -> (4, 'Dave', 16, 'Physics', 'dave@example.com');
Query OK, 4 rows affected (0.04 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql>
mysql> -- Insert into Courses
mysql> INSERT INTO Courses (CourseID, CourseName, Credits) VALUES
    -> (101, 'Intro to CS', 3),
    -> (102, 'Calculus I', 4),
    -> (103, 'Physics I', 4);
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql>
mysql> -- Insert into Enrollments
mysql> INSERT INTO Enrollments (EnrollmentID, StudentID, CourseID, Grade)
VALUES
    -> (1001, 1, 101, 'A'),
    -> (1002, 2, 102, 'B'),
    -> (1003, 3, 101, 'A');
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM Students;
+-----------+-------+------+-----------------+-------------------+
| StudentID | Name  | Age  | Major           | Email             |
+-----------+-------+------+-----------------+-------------------+
|         1 | Alice |   20 | Computer Science | alice@example.com |
|         2 | Bob   |   18 | Mathematics      | bob@example.com   |
|         3 | Carol |   22 | Computer Science | carol@example.com |
|         4 | Dave  |   16 | Physics          | dave@example.com  |
+-----------+-------+------+-----------------+-------------------+
4 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM Courses;
+----------+-------------+---------+
| CourseID | CourseName  | Credits |
+----------+-------------+---------+
|      101 | Intro to CS |       3 |
|      102 | Calculus I  |       4 |
|      103 | Physics I   |       4 |
+----------+-------------+---------+
3 rows in set (0.00 sec)

mysql> SELECT * FROM Enrollments;
+--------------+-----------+----------+-------+
| EnrollmentID | StudentID | CourseID | Grade |
+--------------+-----------+----------+-------+
|         1001 |         1 |      101 | A     |
|         1002 |         2 |      102 | B     |
|         1003 |         3 |      101 | A     |
+--------------+-----------+----------+-------+
3 rows in set (0.00 sec)

mysql> -- Update Alice’s major from 'Computer Science' to 'Data Science'
mysql> UPDATE Students
    -> SET Major = 'Data Science'
    -> WHERE StudentID = 1;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql>
mysql> -- Delete students younger than 18 (this will remove Dave, age 16)
mysql> DELETE FROM Students
    -> WHERE Age < 18;
Query OK, 1 row affected (0.01 sec)

mysql> SELECT * FROM Students;
```

```
+-----------+-------+------+-----------------+-------------------+
| StudentID | Name  | Age  | Major           | Email             |
+-----------+-------+------+-----------------+-------------------+
|         1 | Alice |   20 | Data Science    | alice@example.com |
|         2 | Bob   |   18 | Mathematics     | bob@example.com   |
|         3 | Carol |   22 | Computer Science | carol@example.com |
+-----------+-------+------+-----------------+-------------------+
3 rows in set (0.00 sec)

mysql> -- Show names and majors of students older than 19
mysql> SELECT Name, Major
    -> FROM Students
    -> WHERE Age > 19;
+-------+-----------------+
| Name  | Major           |
+-------+-----------------+
| Alice | Data Science    |
| Carol | Computer Science |
+-------+-----------------+
2 rows in set (0.00 sec)

mysql>
mysql> -- Show average age of students
mysql> SELECT AVG(Age) AS AvgAge
    -> FROM Students;
+---------+
| AvgAge  |
+---------+
| 20.0000 |
+---------+
1 row in set (0.00 sec)

mysql>
mysql> -- Count students in each Major (only majors with more than 5 students)
```

```
mysql> SELECT Major, COUNT(*) AS StudentCount
    -> FROM Students
    -> GROUP BY Major
    -> HAVING COUNT(*) > 5;
Empty set (0.00 sec)

mysql>
mysql> -- Students older than 20 and in Computer Science
mysql> SELECT *
    -> FROM Students
    -> WHERE Age > 20 AND Major = 'Computer Science';
+-----------+-------+------+-----------------+------------------+
| StudentID | Name  | Age  | Major           | Email            |
+-----------+-------+------+-----------------+------------------+
|         3 | Carol |   22 | Computer Science | carol@example.com |
+-----------+-------+------+-----------------+------------------+
1 row in set (0.00 sec)

mysql> SELECT s.Name, e.Grade,
    ->        RANK() OVER (ORDER BY e.Grade DESC) AS RankInClass
    -> FROM Enrollments e
    -> JOIN Students s ON e.StudentID = s.StudentID;
+-------+-------+-------------+
| Name  | Grade | RankInClass |
+-------+-------+-------------+
| Bob   | B     |           1 |
| Alice | A     |           2 |
| Carol | A     |           2 |
+-------+-------+-------------+
3 rows in set (0.00 sec)

mysql> -- ================================
mysql> -- Step 1: Create and Select Database
mysql> -- ================================
```

```
mysql> DROP DATABASE IF EXISTS UniversityDB;
Query OK, 0 rows affected, 1 warning (0.08 sec)

mysql> CREATE DATABASE UniversityDB;
Query OK, 1 row affected (0.02 sec)

mysql> USE UniversityDB;
Database changed
mysql>
mysql> -- ================================
mysql> -- Step 2: Create Tables
mysql> -- ================================
mysql> CREATE TABLE Students(
    ->     StudentID INT PRIMARY KEY,
    ->     Name VARCHAR(50),
    ->     Age INT,
    ->     Major VARCHAR(50)
    -> );
Query OK, 0 rows affected (0.07 sec)

mysql>
mysql> CREATE TABLE Courses(
    ->     CourseID INT PRIMARY KEY,
    ->     CourseName VARCHAR(50),
    ->     Credits INT
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql>
mysql> CREATE TABLE Enrollments(
    ->     EnrollmentID INT PRIMARY KEY,
    ->     StudentID INT,
    ->     CourseID INT,
    ->     Grade CHAR(2),
```

```
    ->     FOREIGN KEY (StudentID) REFERENCES Students(StudentID),
    ->     FOREIGN KEY (CourseID) REFERENCES Courses(CourseID)
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql>
mysql> CREATE TABLE Departments(
    ->     DeptID INT PRIMARY KEY,
    ->     DeptName VARCHAR(50)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql>
mysql> -- ================================
mysql> -- Step 3: Alter and Drop
mysql> -- ================================
mysql> ALTER TABLE Students ADD Email VARCHAR(100);
Query OK, 0 rows affected (0.10 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> DROP TABLE Departments;
Query OK, 0 rows affected (0.03 sec)

mysql>
mysql> -- ================================
mysql> -- Step 4: Insert Data into Students
mysql> -- ================================
mysql> INSERT INTO Students (StudentID, Name, Age, Major) VALUES
    -> (1, 'Alice', 20, 'Computer Science'),
    -> (2, 'Bob', 25, 'DS'),
    -> (3, 'Millie', 17, 'EC');
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0
```

```
mysql>
mysql> -- Update Alice’s major
mysql> UPDATE Students SET Major = 'Data Science' WHERE StudentID = 1;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql>
mysql> -- Delete students younger than 18
mysql> DELETE FROM Students WHERE Age < 18;
Query OK, 1 row affected (0.01 sec)

mysql>
mysql> -- Insert more students
mysql> INSERT INTO Students (StudentID, Name, Age, Major) VALUES
    -> (3, 'Millie', 17, 'EC'),
    -> (4, 'John', 22, 'Computer Science'),
    -> (5, 'Sita', 19, 'Data Science'),
    -> (6, 'Arjun', 23, 'Computer Science'),
    -> (7, 'Priya', 21, 'Electronics'),
    -> (8, 'David', 20, 'Data Science'),
    -> (9, 'Kiran', 22, 'Data Science'),
    -> (10, 'Anjali', 23, 'Data Science'),
    -> (11, 'Rahul', 21, 'Data Science'),
    -> (12, 'Meera', 20, 'Data Science'),
    -> (21, 'Sam', 18, 'CS');
Query OK, 11 rows affected (0.01 sec)
Records: 11  Duplicates: 0  Warnings: 0

mysql>
mysql> -- ================================
mysql> -- Step 5: Insert into Courses
mysql> -- ================================
mysql> INSERT INTO Courses (CourseID, CourseName, Credits) VALUES
    -> (101, 'Database Systems', 4),
```

```
    -> (102, 'Machine Learning', 3),
    -> (103, 'Algorithms', 4),
    -> (104, 'Electronics Basics', 3),
    -> (105, 'Statistics', 3);
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0


mysql>
mysql> -- ================================
mysql> -- Step 6: Insert into Enrollments
mysql> -- ================================
mysql> INSERT INTO Enrollments (EnrollmentID, StudentID, CourseID, Grade)
VALUES
    -> (1, 1, 101, 'A'),
    -> (2, 2, 102, 'B'),
    -> (3, 3, 104, 'C'),
    -> (4, 4, 101, 'B'),
    -> (5, 5, 102, 'A'),
    -> (6, 6, 103, 'B'),
    -> (7, 7, 104, 'A'),
    -> (8, 8, 105, 'B'),
    -> (9, 9, 101, 'A'),
    -> (10, 10, 102, 'C');
Query OK, 10 rows affected (0.02 sec)
Records: 10  Duplicates: 0  Warnings: 0


mysql>
mysql> -- ================================
mysql> -- Step 7: SELECT Queries
mysql> -- ================================
mysql> SELECT Name, Major FROM Students WHERE Age > 19;
+--------+-----------------+
| Name   | Major           |
+--------+-----------------+
```

| Alice | Data Science |

| Bob | DS |

| John | Computer Science |

| Arjun | Computer Science |

| Priya | Electronics |

| David | Data Science |

| Kiran | Data Science |

| Anjali | Data Science |

| Rahul | Data Science |

| Meera | Data Science |

+--------+------------------+

10 rows in set (0.01 sec)

mysql> SELECT AVG(Age) AS AvgAge FROM Students;

+---------+

| AvgAge |

+---------+

| 20.8462 |

+---------+

1 row in set (0.00 sec)

mysql> SELECT Major, COUNT(*) AS StudentCount FROM Students GROUP BY Major HAVING COUNT(*) > 5;

+--------------+--------------+

| Major | StudentCount |

+--------------+--------------+

| Data Science | 7 |

+--------------+--------------+

1 row in set (0.01 sec)

mysql> SELECT * FROM Students WHERE Age > 20 AND Major = 'Computer Science';

+-----------+-------+------+------------------+-------+

| StudentID | Name | Age | Major | Email |

```
+-----------+-------+------+-----------------+-------+
|         4 | John  |   22 | Computer Science | NULL  |
|         6 | Arjun |   23 | Computer Science | NULL  |
+-----------+-------+------+-----------------+-------+
2 rows in set (0.00 sec)


mysql>
mysql> -- ==================================
mysql> -- Step 8: Window Functions
mysql> -- ==================================
mysql> SELECT s.Name, e.Grade, RANK() OVER (ORDER BY e.Grade DESC) AS
RankInClass
    -> FROM Enrollments e JOIN Students s ON e.StudentID = s.StudentID;
+--------+-------+-------------+
| Name   | Grade | RankInClass |
+--------+-------+-------------+
| Millie | C     |           1 |
| Anjali | C     |           1 |
| Bob    | B     |           3 |
| John   | B     |           3 |
| Arjun  | B     |           3 |
| David  | B     |           3 |
| Alice  | A     |           7 |
| Sita   | A     |           7 |
| Priya  | A     |           7 |
| Kiran  | A     |           7 |
+--------+-------+-------------+
10 rows in set (0.01 sec)


mysql>
mysql> SELECT s.Name, e.Grade, RANK() OVER (ORDER BY e.Grade ASC) AS
RankInClass
    -> FROM Enrollments e JOIN Students s ON e.StudentID = s.StudentID;
+--------+-------+-------------+
```

```
| Name   | Grade | RankInClass |
+--------+-------+-------------+
| Alice  | A     |           1 |
| Sita   | A     |           1 |
| Priya  | A     |           1 |
| Kiran  | A     |           1 |
| Bob    | B     |           5 |
| John   | B     |           5 |
| Arjun  | B     |           5 |
| David  | B     |           5 |
| Millie | C     |           9 |
| Anjali | C     |           9 |
+--------+-------+-------------+
10 rows in set (0.00 sec)

mysql>
mysql> -- ================================
mysql> -- Step 9: Joins
mysql> -- ================================
mysql> -- Inner Join
mysql> SELECT s.Name, c.CourseName
    -> FROM Students s
    -> INNER JOIN Enrollments e ON s.StudentID = e.StudentID
    -> INNER JOIN Courses c ON e.CourseID = c.CourseID;
+--------+-------------------+
| Name   | CourseName        |
+--------+-------------------+
| Alice  | Database Systems  |
| John   | Database Systems  |
| Kiran  | Database Systems  |
| Bob    | Machine Learning  |
| Sita   | Machine Learning  |
| Anjali | Machine Learning  |
| Arjun  | Algorithms        |
```

```
| Millie | Electronics Basics |
| Priya  | Electronics Basics |
| David  | Statistics         |
+--------+--------------------+
10 rows in set (0.00 sec)

mysql>
mysql> -- Left Join
mysql> SELECT s.Name, c.CourseName
    -> FROM Students s
    -> LEFT JOIN Enrollments e ON s.StudentID = e.StudentID
    -> LEFT JOIN Courses c ON e.CourseID = c.CourseID;
+--------+--------------------+
| Name   | CourseName         |
+--------+--------------------+
| Alice  | Database Systems   |
| Bob    | Machine Learning   |
| Millie | Electronics Basics |
| John   | Database Systems   |
| Sita   | Machine Learning   |
| Arjun  | Algorithms         |
| Priya  | Electronics Basics |
| David  | Statistics         |
| Kiran  | Database Systems   |
| Anjali | Machine Learning   |
| Rahul  | NULL               |
| Meera  | NULL               |
| Sam    | NULL               |
+--------+--------------------+
13 rows in set (0.00 sec)

mysql>
mysql> -- Cross Join
mysql> SELECT s.Name, c.CourseName
```

```
    -> FROM Students s CROSS JOIN Courses c;
+--------+--------------------+
| Name   | CourseName         |
+--------+--------------------+
| Alice  | Statistics         |
| Alice  | Electronics Basics |
| Alice  | Algorithms         |
| Alice  | Machine Learning   |
| Alice  | Database Systems   |
| Bob    | Statistics         |
| Bob    | Electronics Basics |
| Bob    | Algorithms         |
| Bob    | Machine Learning   |
| Bob    | Database Systems   |
| Millie | Statistics         |
| Millie | Electronics Basics |
| Millie | Algorithms         |
| Millie | Machine Learning   |
| Millie | Database Systems   |
| John   | Statistics         |
| John   | Electronics Basics |
| John   | Algorithms         |
| John   | Machine Learning   |
| John   | Database Systems   |
| Sita   | Statistics         |
| Sita   | Electronics Basics |
| Sita   | Algorithms         |
| Sita   | Machine Learning   |
| Sita   | Database Systems   |
| Arjun  | Statistics         |
| Arjun  | Electronics Basics |
| Arjun  | Algorithms         |
| Arjun  | Machine Learning   |
| Arjun  | Database Systems   |
```

Priya	Statistics
Priya	Electronics Basics
Priya	Algorithms
Priya	Machine Learning
Priya	Database Systems
David	Statistics
David	Electronics Basics
David	Algorithms
David	Machine Learning
David	Database Systems
Kiran	Statistics
Kiran	Electronics Basics
Kiran	Algorithms
Kiran	Machine Learning
Kiran	Database Systems
Anjali	Statistics
Anjali	Electronics Basics
Anjali	Algorithms
Anjali	Machine Learning
Anjali	Database Systems
Rahul	Statistics
Rahul	Electronics Basics
Rahul	Algorithms
Rahul	Machine Learning
Rahul	Database Systems
Meera	Statistics
Meera	Electronics Basics
Meera	Algorithms
Meera	Machine Learning
Meera	Database Systems
Sam	Statistics
Sam	Electronics Basics
Sam	Algorithms
Sam	Machine Learning

```
| Sam    | Database Systems   |
+--------+--------------------+
65 rows in set (0.00 sec)

mysql>
mysql> -- Self Join
mysql> SELECT s1.Name AS Student1, s2.Name AS Student2
    -> FROM Students s1 JOIN Students s2
    -> ON s1.Major = s2.Major AND s1.StudentID < s2.StudentID;
+----------+----------+
| Student1 | Student2 |
+----------+----------+
| Alice    | Sita     |
| Alice    | David    |
| Alice    | Kiran    |
| Alice    | Anjali   |
| Alice    | Rahul    |
| Alice    | Meera    |
| John     | Arjun    |
| Sita     | David    |
| Sita     | Kiran    |
| Sita     | Anjali   |
| Sita     | Rahul    |
| Sita     | Meera    |
| David    | Kiran    |
| David    | Anjali   |
| David    | Rahul    |
| David    | Meera    |
| Kiran    | Anjali   |
| Kiran    | Rahul    |
| Kiran    | Meera    |
| Anjali   | Rahul    |
| Anjali   | Meera    |
| Rahul    | Meera    |
```

```
+----------+----------+
22 rows in set (0.00 sec)

mysql>
mysql> -- ==================================
mysql> -- Step 10: Group Concatenation
mysql> -- ==================================
mysql> SELECT Major, GROUP_CONCAT(Name SEPARATOR ', ') AS Students
    -> FROM Students
    -> GROUP BY Major;
+-----------------+-----------------------------------------------+
| Major           | Students                                      |
+-----------------+-----------------------------------------------+
| Computer Science | John, Arjun                                   |
| CS              | Sam                                           |
| Data Science    | Alice, Sita, David, Kiran, Anjali, Rahul, Meera |
| DS              | Bob                                           |
| EC              | Millie                                        |
| Electronics     | Priya                                         |
+-----------------+-----------------------------------------------+
6 rows in set (0.00 sec)

mysql>
mysql> -- ==================================
mysql> -- Step 11: Subqueries
mysql> -- ==================================
mysql> SELECT Name FROM Students WHERE Age > (SELECT AVG(Age) FROM
Students);
+--------+
| Name   |
+--------+
| Bob    |
| John   |
| Arjun  |
```

| Priya |

| Kiran |

| Anjali |

| Rahul |

+--------+

7 rows in set (0.00 sec)

mysql>

mysql> SELECT Name FROM Students s

    -> WHERE EXISTS (SELECT * FROM Enrollments e WHERE e.StudentID = s.StudentID AND e.Grade = 'A');

+-------+

| Name |

+-------+

| Alice |

| Sita |

| Priya |

| Kiran |

+-------+

4 rows in set (0.00 sec)

mysql>

mysql> SELECT Major, AvgAge

    -> FROM (SELECT Major, AVG(Age) AS AvgAge FROM Students GROUP BY Major) t;

+------------------+---------+

| Major | AvgAge |

+------------------+---------+

| Data Science | 20.7143 |

| DS | 25.0000 |

| EC | 17.0000 |

| Computer Science | 22.5000 |

| Electronics | 21.0000 |

| CS | 18.0000 |

```
+-----------------+---------+
6 rows in set (0.00 sec)

mysql>
mysql> -- ================================
mysql> -- Step 12: Set Operations
mysql> -- ================================
mysql> SELECT Name FROM Students
    -> UNION
    -> SELECT CourseName FROM Courses;
+--------------------+
| Name               |
+--------------------+
| Alice              |
| Bob                |
| Millie             |
| John               |
| Sita               |
| Arjun              |
| Priya              |
| David              |
| Kiran              |
| Anjali             |
| Rahul              |
| Meera              |
| Sam                |
| Database Systems   |
| Machine Learning   |
| Algorithms         |
| Electronics Basics |
| Statistics         |
+--------------------+
18 rows in set (0.00 sec)
```

```
mysql>
mysql> SELECT StudentID FROM Enrollments
    -> INTERSECT
    -> SELECT StudentID FROM Students;
+-----------+
| StudentID |
+-----------+
|         1 |
|         2 |
|         3 |
|         4 |
|         5 |
|         6 |
|         7 |
|         8 |
|         9 |
|        10 |
+-----------+
10 rows in set (0.00 sec)

mysql>
mysql> SELECT StudentID FROM Students
    -> EXCEPT
    -> SELECT StudentID FROM Enrollments;
+-----------+
| StudentID |
+-----------+
|        11 |
|        12 |
|        21 |
+-----------+
3 rows in set (0.00 sec)

mysql>
```

```
mysql> -- ================================
mysql> -- Step 13: Constraints
mysql> -- ================================
mysql> ALTER TABLE Students ADD CONSTRAINT AgeCheck CHECK (Age >= 17);
Query OK, 13 rows affected (0.08 sec)
Records: 13  Duplicates: 0  Warnings: 0

mysql> ALTER TABLE Students DROP CONSTRAINT AgeCheck;
Query OK, 0 rows affected (0.01 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql>
mysql> -- ================================
mysql> -- Step 14: Transactions
mysql> -- ================================
mysql> BEGIN;
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO Enrollments VALUES (101, 1, 101, 'A');
Query OK, 1 row affected (0.00 sec)

mysql> UPDATE Students SET Major = 'AI' WHERE StudentID = 1;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> COMMIT;
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> -- ================================
mysql> -- Step 15: Indexing
mysql> -- ================================
mysql> CREATE INDEX idx_student_major ON Students(Major);
Query OK, 0 rows affected (0.05 sec)
```

```
Records: 0  Duplicates: 0  Warnings: 0

mysql>
mysql> SELECT * FROM Students WHERE Major = 'Data Science';
+-----------+--------+------+--------------+-------+
| StudentID | Name   | Age  | Major        | Email |
+-----------+--------+------+--------------+-------+
|         5 | Sita   |   19 | Data Science | NULL  |
|         8 | David  |   20 | Data Science | NULL  |
|         9 | Kiran  |   22 | Data Science | NULL  |
|        10 | Anjali |   23 | Data Science | NULL  |
|        11 | Rahul  |   21 | Data Science | NULL  |
|        12 | Meera  |   20 | Data Science | NULL  |
+-----------+--------+------+--------------+-------+
6 rows in set (0.00 sec)

mysql>
mysql> -- ================================
mysql> -- Step 16: Sorting & Limiting
mysql> -- ================================
mysql> SELECT Name, Age FROM Students ORDER BY Age DESC;
+--------+------+
| Name   | Age  |
+--------+------+
| Bob    |   25 |
| Arjun  |   23 |
| Anjali |   23 |
| John   |   22 |
| Kiran  |   22 |
| Priya  |   21 |
| Rahul  |   21 |
| Alice  |   20 |
| David  |   20 |
| Meera  |   20 |
```

| Sita | 19 |

| Sam | 18 |

| Millie | 17 |

+--------+------+

13 rows in set (0.00 sec)

mysql> SELECT Name, Age FROM Students ORDER BY Age DESC, Name ASC;

+--------+------+

| Name | Age |

+--------+------+

| Bob | 25 |

| Anjali | 23 |

| Arjun | 23 |

| John | 22 |

| Kiran | 22 |

| Priya | 21 |

| Rahul | 21 |

| Alice | 20 |

| David | 20 |

| Meera | 20 |

| Sita | 19 |

| Sam | 18 |

| Millie | 17 |

+--------+------+

13 rows in set (0.00 sec)

mysql> SELECT * FROM Students LIMIT 5;

+-----------+--------+------+-----------------+-------+

| StudentID | Name | Age | Major | Email |

+-----------+--------+------+-----------------+-------+

| 1 | Alice | 20 | AI | NULL |

| 2 | Bob | 25 | DS | NULL |

| 3 | Millie | 17 | EC | NULL |

| 4 | John | 22 | Computer Science | NULL |

```
|        5 | Sita   |   19 | Data Science     | NULL  |
+-----------+--------+------+------------------+-------+
5 rows in set (0.00 sec)

mysql>
mysql> -- ================================
mysql> -- Step 17: Common Table Expression (CTE)
mysql> -- ================================
mysql> WITH AvgAge AS (SELECT AVG(Age) AS AgeValue FROM Students)
    -> SELECT * FROM Students WHERE Age > (SELECT AgeValue FROM AvgAge);
+-----------+--------+------+------------------+-------+
| StudentID | Name   | Age  | Major            | Email |
+-----------+--------+------+------------------+-------+
|         2 | Bob    |   25 | DS               | NULL  |
|         4 | John   |   22 | Computer Science | NULL  |
|         6 | Arjun  |   23 | Computer Science | NULL  |
|         7 | Priya  |   21 | Electronics      | NULL  |
|         9 | Kiran  |   22 | Data Science     | NULL  |
|        10 | Anjali |   23 | Data Science     | NULL  |
|        11 | Rahul  |   21 | Data Science     | NULL  |
+-----------+--------+------+------------------+-------+
7 rows in set (0.00 sec)

mysql>
```