Dnyanraj Sonawane (COTB54)

(MySQL) Practical 7

```
mysql> USE CollegeDB;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> CREATE TABLE Cust_Old (
    ->     CustID INT PRIMARY KEY,
    ->     CustName VARCHAR(100),
    ->     City VARCHAR(50)
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> CREATE TABLE Cust_New (
    ->     CustID INT PRIMARY KEY,
    ->     CustName VARCHAR(100),
    ->     City VARCHAR(50)
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> INSERT INTO Cust_Old (CustID, CustName, City) VALUES
    -> (1, 'Ramesh Sharma', 'Delhi'),
    -> (2, 'Suresh Patil', 'Mumbai'),
    -> (3, 'Anita Desai', 'Pune');
Query OK, 3 rows affected (0.00 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> INSERT INTO Cust_New (CustID, CustName, City) VALUES
    -> (2, 'Suresh Patil', 'Mumbai'),
    -> (3, 'Anita Desai', 'Pune'),
    -> (4, 'Vikram Singh', 'Delhi'),
    -> (5, 'Priya Iyer', 'Chennai'),
    -> (6, 'Rahul Mehta', 'Pune');
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql> DELIMITER $
mysql>
mysql> CREATE PROCEDURE Merge_Customers_ByCity(IN city_param VARCHAR(50))
    -> BEGIN
    ->     DECLARE v_id INT;
    ->     DECLARE v_name VARCHAR(100);
    ->     DECLARE v_city VARCHAR(50);
    ->     DECLARE done INT DEFAULT 0;
    ->
    ->     -- Cursor: fetch customers from Cust_New for the given city
    ->     DECLARE cur CURSOR FOR
    ->         SELECT CustID, CustName, City
    ->         FROM Cust_New
    ->         WHERE City = city_param;
    ->
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
    ->
    ->     OPEN cur;
    ->
    ->     read_loop: LOOP
    ->         FETCH cur INTO v_id, v_name, v_city;
    ->         IF done = 1 THEN
    ->             LEAVE read_loop;
    ->         END IF;
    ->
    ->         -- Insert only if not exists in Cust_Old
    ->         IF NOT EXISTS (SELECT 1 FROM Cust_Old WHERE CustID = v_id) THEN
    ->             INSERT INTO Cust_Old (CustID, CustName, City)
    ->             VALUES (v_id, v_name, v_city);
    ->         END IF;
    ->     END LOOP;
    ->
    ->     CLOSE cur;
    -> END $
Query OK, 0 rows affected (0.20 sec)

mysql> CALL Merge_Customers_ByCity('Pune');$
Query OK, 0 rows affected (0.01 sec)

mysql> CALL Merge_Customers_ByCity('Delhi');$
Query OK, 0 rows affected (0.00 sec)

mysql> SELECT * FROM Cust_Old;$
+--------+---------------+--------+
| CustID | CustName      | City   |
+--------+---------------+--------+
|      1 | Ramesh Sharma | Delhi  |
|      2 | Suresh Patil  | Mumbai |
|      3 | Anita Desai   | Pune   |
|      4 | Vikram Singh  | Delhi  |
|      6 | Rahul Mehta   | Pune   |
+--------+---------------+--------+
5 rows in set (0.00 sec)

mysql> CALL Merge_Customers_ByCity('Chennai');$
Query OK, 0 rows affected (0.01 sec)

mysql> SELECT * FROM Cust_Old;$
+--------+---------------+---------+
| CustID | CustName      | City    |
+--------+---------------+---------+
|      1 | Ramesh Sharma | Delhi   |
|      2 | Suresh Patil  | Mumbai  |
|      3 | Anita Desai   | Pune    |
|      4 | Vikram Singh  | Delhi   |
|      5 | Priya Iyer    | Chennai |
|      6 | Rahul Mehta   | Pune    |
+--------+---------------+---------+
6 rows in set (0.00 sec)

mysql>
```

Dnyanraj Sonawane (COTB54)

(MySQL) Practical 8

```
mysql> USE CollegeDB;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> CREATE TABLE Library (
    ->     BookID INT PRIMARY KEY AUTO_INCREMENT,
    ->     Title VARCHAR(100),
    ->     Author VARCHAR(100),
    ->     Published_Year INT
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TABLE Library_Audit (
    ->     AuditID INT PRIMARY KEY AUTO_INCREMENT,
    ->     BookID INT,
    ->     Title VARCHAR(100),
    ->     Author VARCHAR(100),
    ->     Published_Year INT,
    ->     Operation_Type VARCHAR(20),
    ->     Operation_Time TIMESTAMP DEFAULT CURRENT_TIMESTAMP
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> DESC Library;
+----------------+--------------+------+-----+---------+----------------+
| Field          | Type         | Null | Key | Default | Extra          |
+----------------+--------------+------+-----+---------+----------------+
| BookID         | int          | NO   | PRI | NULL    | auto_increment |
| Title          | varchar(100) | YES  |     | NULL    |                |
| Author         | varchar(100) | YES  |     | NULL    |                |
| Published_Year | int          | YES  |     | NULL    |                |
+----------------+--------------+------+-----+---------+----------------+
4 rows in set (0.00 sec)

mysql> DESC Library_Audit;
+--------+-------------+------+-----+-------------------+-------------------+
| Field  | Type        | Null | Key | Default           | Extra             |
+--------+-------------+------+-----+-------------------+-------------------+
| AID    | int         | NO   | PRI | NULL              | auto_increment    |
| BID    | int         | YES  |     | NULL              |                   |
| Title  | varchar(60) | YES  |     | NULL              |                   |
| Op     | varchar(10) | YES  |     | NULL              |                   |
| OpTime | timestamp   | YES  |     | CURRENT_TIMESTAMP | DEFAULT_GENERATED |
+--------+-------------+------+-----+-------------------+-------------------+
5 rows in set (0.00 sec)

mysql> INSERT INTO Library (Title, Author, Published_Year) VALUES
    -> ('Database Systems', 'C. J. Date', 2019),
    -> ('Learning SQL', 'Alan Beaulieu', 2020),
    -> ('PL/SQL Programming', 'Steven Feuerstein', 2018),
    -> ('Wings of Fire', 'A. P. J. Abdul Kalam', 1999),
    -> ('The Guide', 'R. K. Narayan', 1958),
    -> ('Train to Pakistan', 'Khushwant Singh', 1956),
    -> ('Clean Code', 'Robert C. Martin', 2008),
    -> ('Design Patterns', 'Erich Gamma', 1994),
    -> ('Introduction to Algorithms', 'Thomas H. Cormen', 2009),
    -> ('Effective Java', 'Joshua Bloch', 2017),
    -> ('The Pragmatic Programmer', 'Andrew Hunt', 1999),
    -> ('India After Gandhi', 'Ramachandra Guha', 2007),
    -> ('The White Tiger', 'Aravind Adiga', 2008);
Query OK, 13 rows affected (0.01 sec)
Records: 13  Duplicates: 0  Warnings: 0

mysql> DELIMITER $
mysql>
mysql> CREATE TRIGGER trg_Library_Update
    -> AFTER UPDATE ON Library
    -> FOR EACH ROW
    -> BEGIN
    ->     INSERT INTO Library_Audit(BID, Title, Op)
    ->     VALUES (OLD.BookID, OLD.Title, 'UPDATE');
    -> END $
Query OK, 0 rows affected (0.00 sec)

mysql> DELIMITER $
mysql>
mysql> CREATE TRIGGER trg_Library_Delete
    -> AFTER DELETE ON Library
    -> FOR EACH ROW
    -> BEGIN
    ->     INSERT INTO Library_Audit(BID, Title, Op)
    ->     VALUES (OLD.BookID, OLD.Title, 'DELETE');
    -> END $
Query OK, 0 rows affected (0.00 sec)

mysql> UPDATE Library SET Published_Year = 2005 WHERE BookID = 9;$
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> DELETE FROM Library WHERE BookID = 7;$
Query OK, 1 row affected (0.00 sec)

mysql> DELETE FROM Library WHERE BookID = 2;$
Query OK, 1 row affected (0.00 sec)

mysql> UPDATE Library SET Author = 'Chris Date' WHERE BookID = 1;$
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM Library;$
+--------+----------------------------+----------------------+----------------+
| BookID | Title                      | Author               | Published_Year |
+--------+----------------------------+----------------------+----------------+
|      1 | Database Systems           | Chris Date           |           2019 |
|      3 | PL/SQL Programming         | Steven Feuerstein    |           2018 |
|      4 | Wings of Fire              | A. P. J. Abdul Kalam |           1999 |
|      5 | The Guide                  | R. K. Narayan        |           1958 |
|      6 | Train to Pakistan          | Khushwant Singh      |           1956 |
|      8 | Design Patterns            | Erich Gamma          |           1994 |
|      9 | Introduction to Algorithms | Thomas H. Cormen     |           2005 |
|     10 | Effective Java             | Joshua Bloch         |           2017 |
|     11 | The Pragmatic Programmer   | Andrew Hunt          |           1999 |
|     12 | India After Gandhi         | Ramachandra Guha     |           2007 |
|     13 | The White Tiger            | Aravind Adiga        |           2008 |
+--------+----------------------------+----------------------+----------------+
11 rows in set (0.00 sec)

mysql> SELECT * FROM Library_Audit;$
+-----+------+----------------------------+--------+---------------------+
| AID | BID  | Title                      | Op     | OpTime              |
+-----+------+----------------------------+--------+---------------------+
|   1 |    9 | Introduction to Algorithms | UPDATE | 2025-09-18 12:21:34 |
|   2 |    7 | Clean Code                 | DELETE | 2025-09-18 12:22:04 |
|   3 |    2 | Learning SQL               | DELETE | 2025-09-18 12:22:11 |
|   4 |    1 | Database Systems           | UPDATE | 2025-09-18 12:22:22 |
+-----+------+----------------------------+--------+---------------------+
4 rows in set (0.00 sec)

mysql>
```

Dnyanraj Sonawane (COTB54)

(MySQL) Practical 9

```
test> use mydb;
switched to db mydb
mydb> db.customers.insertOne({
...   _id: 1,
...   name: "Ramesh Sharma",
...   city: "Delhi",
...   age: 35,
...   active: true
... });
...
{ acknowledged: true, insertedId: 1 }
mydb> db.customers.insertMany([
...   { _id: 2, name: "Suresh Patil", city: "Mumbai", age: 42, active: true },
...   { _id: 3, name: "Anita Desai", city: "Pune", age: 29, active: false },
...   { _id: 4, name: "Priya Iyer", city: "Chennai", age: 31, active: true },
...   { _id: 5, name: "Rahul Mehta", city: "Pune", age: 27, active: true }
... ]);
...
{ acknowledged: true, insertedIds: { '0': 2, '1': 3, '2': 4, '3': 5 } }
mydb> db.customers.find();
...
[
  {
    _id: 1,
    name: 'Ramesh Sharma',
    city: 'Delhi',
    age: 35,
    active: true
  },
  {
    _id: 2,
    name: 'Suresh Patil',
    city: 'Mumbai',
    age: 42,
    active: true
  },
  { _id: 3, name: 'Anita Desai', city: 'Pune', age: 29, active: false },
  {
    _id: 4,
    name: 'Priya Iyer',
    city: 'Chennai',
    age: 31,
    active: true
  },
  { _id: 5, name: 'Rahul Mehta', city: 'Pune', age: 27, active: true }
]
mydb> db.customers.find({ city: "Pune" });
...
[
  { _id: 3, name: 'Anita Desai', city: 'Pune', age: 29, active: false },
  { _id: 5, name: 'Rahul Mehta', city: 'Pune', age: 27, active: true }
]
mydb> db.customers.find({}, { name: 1, city: 1, _id: 0 });
...
[
  { name: 'Ramesh Sharma', city: 'Delhi' },
  { name: 'Suresh Patil', city: 'Mumbai' },
  { name: 'Anita Desai', city: 'Pune' },
  { name: 'Priya Iyer', city: 'Chennai' },
  { name: 'Rahul Mehta', city: 'Pune' }
]
mydb> db.customers.updateOne(
...   { _id: 3 },
...   { $set: { active: true } }
... );
```

```
...
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
mydb> db.customers.updateMany(
...   { city: "Pune" },
...   { $set: { active: false } }
... );
...
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 2,
  modifiedCount: 2,
  upsertedCount: 0
}
mydb> db.customers.deleteOne({ _id: 5 });
...
{ acknowledged: true, deletedCount: 1 }
mydb> db.customers.deleteMany({ city: "Chennai" });
...
{ acknowledged: true, deletedCount: 1 }
mydb> db.customers.insertOne({
...   _id: 6,
...   name: "Vikram Singh",
...   city: "Delhi",
...   age: 40,
...   active: true
... });
...
{ acknowledged: true, insertedId: 6 }
mydb> db.customers.replaceOne(
...   { _id: 6 },  // match by id
...   {
...     _id: 6,
...     name: "Vikram Singh",
...     city: "Delhi",
...     age: 41,     // updated age
...     active: false
...   },
...   { upsert: true }   // insert if not exists
... );
...
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
mydb> db.customers.find({ city: "Pune", active: true });
...

mydb> db.customers.find({ city: "Pune", active: true });

mydb> db.customers.find({
...   $or: [{ city: "Delhi" }, { city: "Mumbai" }]
... });
...
[
  {
    _id: 1,
    name: 'Ramesh Sharma',
    city: 'Delhi',
    age: 35,
    active: true
  },
  {
    _id: 2,
    name: 'Suresh Patil',
    city: 'Mumbai',
    age: 42,
    active: true
  },
  {
    _id: 6,
    name: 'Vikram Singh',
    city: 'Delhi',
    age: 41,
    active: false
  }
]
mydb> db.customers.find({
...   age: { $not: { $gte: 40 } }
... });
...
[
  {
    _id: 1,
    name: 'Ramesh Sharma',
    city: 'Delhi',
    age: 35,
    active: true
  },
  { _id: 3, name: 'Anita Desai', city: 'Pune', age: 29, active: false }
]
mydb> db.customers.find({
...   $nor: [{ city: "Pune" }, { active: false }]
... });
...
[
  {
    _id: 1,
    name: 'Ramesh Sharma',
    city: 'Delhi',
    age: 35,
    active: true
  },
  {
    _id: 2,
    name: 'Suresh Patil',
    city: 'Mumbai',
    age: 42,
    active: true
  }
]
mydb>
```

Dnyanraj Sonawane (COTB54)

(MySQL) Practical 10

```
test> use newdb;
switched to db newdb
newdb> db.customers.insertMany([
...   { _id: 1, name: "Ramesh Sharma", city: "Delhi", age: 35, active: true },
...   { _id: 2, name: "Suresh Patil", city: "Mumbai", age: 42, active: true },
...   { _id: 3, name: "Anita Desai", city: "Pune", age: 29, active: false },
...   { _id: 4, name: "Priya Iyer", city: "Chennai", age: 31, active: true },
...   { _id: 5, name: "Rahul Mehta", city: "Pune", age: 27, active: true },
...   { _id: 6, name: "Vikram Singh", city: "Delhi", age: 40, active: false }
... ]);
...
{
  acknowledged: true,
  insertedIds: { '0': 1, '1': 2, '2': 3, '3': 4, '4': 5, '5': 6 }
}
newdb> db.customers.aggregate([
...   { $group: { _id: "$city", total_customers: { $sum: 1 } } }
... ]);
...
[
  { _id: 'Delhi', total_customers: 2 },
  { _id: 'Chennai', total_customers: 1 },
  { _id: 'Pune', total_customers: 2 },
  { _id: 'Mumbai', total_customers: 1 }
]
newdb> db.customers.aggregate([
...   { $group: { _id: "$city", avg_age: { $avg: "$age" } } }
... ]);
...
[
  { _id: 'Delhi', avg_age: 37.5 },
  { _id: 'Chennai', avg_age: 31 },
  { _id: 'Pune', avg_age: 28 },
  { _id: 'Mumbai', avg_age: 42 }
]
newdb> db.customers.aggregate([
...   { $match: { active: true } },
...   { $group: { _id: "$city", active_count: { $sum: 1 } } }
... ]);
...
[
  { _id: 'Delhi', active_count: 1 },
  { _id: 'Chennai', active_count: 1 },
  { _id: 'Pune', active_count: 1 },
  { _id: 'Mumbai', active_count: 1 }
]
newdb> db.customers.createIndex({ city: 1 });
...
city_1
newdb> db.customers.getIndexes();
...
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { city: 1 }, name: 'city_1' }
]
newdb> db.customers.find({ city: "Pune" }).explain("executionStats");
...
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'newdb.customers',
    indexFilterSet: false,
    parsedQuery: { city: { '$eq': 'Pune' } },
    queryHash: '96110838',
    planCacheKey: '5B786561',
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    winningPlan: {
      stage: 'FETCH',
      inputStage: {
        stage: 'IXSCAN',
        keyPattern: { city: 1 },
        indexName: 'city_1',
        isMultiKey: false,
        multiKeyPaths: { city: [] },
        isUnique: false,
        isSparse: false,
        isPartial: false,
        indexVersion: 2,
        direction: 'forward',
        indexBounds: { city: [ '["Pune", "Pune"]' ] }
      }
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 2,
    executionTimeMillis: 0,
    totalKeysExamined: 2,
    totalDocsExamined: 2,
    executionStages: {
      stage: 'FETCH',
      nReturned: 2,
      executionTimeMillisEstimate: 0,
      works: 3,
      advanced: 2,
      needTime: 0,
      needYield: 0,
      saveState: 0,
      restoreState: 0,
      isEOF: 1,
      docsExamined: 2,
      alreadyHasObj: 0,
      inputStage: {
        stage: 'IXSCAN',
        nReturned: 2,
        executionTimeMillisEstimate: 0,
        works: 3,
        advanced: 2,
        needTime: 0,
        needYield: 0,
        saveState: 0,
        restoreState: 0,
        isEOF: 1,
        keyPattern: { city: 1 },
        indexName: 'city_1',
        isMultiKey: false,
        multiKeyPaths: { city: [] },
        isUnique: false,
        isSparse: false,
        isPartial: false,
        indexVersion: 2,
        direction: 'forward',
        indexBounds: { city: [ '["Pune", "Pune"]' ] },
        keysExamined: 2,
        seeks: 1,
        dupsTested: 0,
        dupsDropped: 0
      }
    }
  },
  command: { find: 'customers', filter: { city: 'Pune' }, '$db': 'newdb' },
  serverInfo: {
    host: '8682875bb587',
    port: 27017,
    version: '6.0.26',
    gitVersion: '0c4ec4b6005f75582ce208fc800f09f561b6c2e8'
  },
  serverParameters: {
    internalQueryFacetBufferSizeBytes: 104857600,
    internalQueryFacetMaxOutputDocSizeBytes: 104857600,
    internalLookupStageIntermediateDocumentMaxSizeBytes: 104857600,
    internalDocumentSourceGroupMaxMemoryBytes: 104857600,
    internalQueryMaxBlockingSortMemoryUsageBytes: 104857600,
    internalQueryProhibitBlockingMergeOnMongoS: 0,
    internalQueryMaxAddToSetBytes: 104857600,
    internalDocumentSourceSetWindowFieldsMaxMemoryBytes: 104857600
  },
  ok: 1
}
newdb> db.customers.createIndex({ city: 1, age: -1 });
...
city_1_age_-1
newdb> db.customers.find({ city: "Pune" }).sort({ age: -1 });
...
[
  { _id: 3, name: 'Anita Desai', city: 'Pune', age: 29, active: false },
  { _id: 5, name: 'Rahul Mehta', city: 'Pune', age: 27, active: true }
]
newdb>
```

Dnyanraj Sonawane (COTB54)

(MySQL) Practical 11

```
test> use salesdb;
switched to db salesdb
salesdb> db.sales.insertMany([
...   { _id: 1, item: "Pen", city: "Delhi", qty: 10 },
...   { _id: 2, item: "Notebook", city: "Delhi", qty: 5 },
...   { _id: 3, item: "Pen", city: "Mumbai", qty: 15 },
...   { _id: 4, item: "Pencil", city: "Pune", qty: 20 },
...   { _id: 5, item: "Notebook", city: "Mumbai", qty: 7 },
...   { _id: 6, item: "Pen", city: "Pune", qty: 12 },
...   { _id: 7, item: "Pencil", city: "Delhi", qty: 8 }
... ]);
...
{
  acknowledged: true,
  insertedIds: { '0': 1, '1': 2, '2': 3, '3': 4, '4': 5, '5': 6, '6': 7 }
}
salesdb> var mapFunction = function() {
...     emit(this.item, this.qty);
... };
...

salesdb> var reduceFunction = function(key, values) {
...     return Array.sum(values);
... };
...

salesdb> db.sales.mapReduce(
...     mapFunction,
...     reduceFunction,
...     { out: "item_totals" }
... );
...
DeprecationWarning: Collection.mapReduce() is deprecated. Use an aggregation
instead.
See https://mongodb.com/docs/manual/core/map-reduce for details.
{ result: 'item_totals', ok: 1 }
salesdb> db.item_totals.find();
...
[
  { _id: 'Notebook', value: 12 },
  { _id: 'Pencil', value: 28 },
  { _id: 'Pen', value: 37 }
]
salesdb> var mapFunction2 = function() {
...     emit(this.city, this.qty);
... };
...
... var reduceFunction2 = function(key, values) {
...     return Array.sum(values);
... };
...
... db.sales.mapReduce(
...     mapFunction2,
...     reduceFunction2,
...     { out: "city_totals" }
... );
...
... db.city_totals.find();
...
[
  { _id: 'Delhi', value: 23 },
  { _id: 'Mumbai', value: 22 },
  { _id: 'Pune', value: 32 }
]
salesdb>
```

Dnyanraj Sonawane (COTB54)

(MySQL) Practical 12

“MySQL Setup”

```
mysql> USE CollegeDB;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> CREATE TABLE Students (
    ->     student_id INT PRIMARY KEY,
    ->     name VARCHAR(100),
    ->     department VARCHAR(50),
    ->     age INT,
    ->     active BOOLEAN
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> INSERT INTO Students (student_id, name, department, age, active) VALUES
    -> (1, 'Omkar Sonawane', 'Computer Engineering', 20, true),
    -> (2, 'Raviraj Shingare', 'Civil Engineering', 19, false),
    -> (3, 'Raj Sonawane', 'Computer Engineering', 21, true),
    -> (4, 'Mahesh Salgar', 'Mechanical Engineering', 20, true);
Query OK, 4 rows affected (0.02 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM Students;
+------------+------------------+------------------------+------+--------+
| student_id | name             | department             | age  | active |
+------------+------------------+------------------------+------+--------+
|          1 | Omkar Sonawane   | Computer Engineering   |   20 |      1 |
|          2 | Raviraj Shingare | Civil Engineering      |   19 |      0 |
|          3 | Raj Sonawane     | Computer Engineering   |   21 |      1 |
|          4 | Mahesh Salgar    | Mechanical Engineering |   20 |      1 |
+------------+------------------+------------------------+------+--------+
4 rows in set (0.00 sec)
```

“After Executing the Java Program”

```
mysql> SELECT * FROM Students;
+------------+--------------------+----------------------+------+--------+
| student_id | name               | department           | age  | active |
+------------+--------------------+----------------------+------+--------+
|          1 | Omkar Sonawane     | Computer Engineering |   20 |      1 |
|          2 | Raviraj Shingare   | Civil Engineering    |   19 |      1 |
|          3 | Raj Sonawane       | Computer Engineering |   21 |      1 |
|          5 | Sudarshan Sonawane | ENTC Engineering     |   23 |      1 |
+------------+--------------------+----------------------+------+--------+
4 rows in set (0.00 sec)

mysql>
```

“Java Program”

```
import java.sql.*;
import java.util.Scanner;

public class MySQL_DBNavigation {

    public static void main(String[] args) {
        String url = "jdbc:mysql://localhost:3306/CollegeDB"; // Database URL
        String user = "root"; // MySQL username
        String password = "mysql123"; // MySQL password

        try (Connection con = DriverManager.getConnection(url, user, password);
             Scanner sc = new Scanner(System.in)) {

            int choice;

            do {
                System.out.println("\n--- CollegeDB CRUD Menu ---");
                System.out.println("1. Add Student");
                System.out.println("2. View Students");
                System.out.println("3. Edit Student");
                System.out.println("4. Delete Student");
                System.out.println("5. Exit");
                System.out.print("Enter your choice: ");
                choice = sc.nextInt();
                sc.nextLine(); // consume newline

                switch (choice) {
                    case 1:
                        System.out.print("Enter Student ID: ");
                        int id = sc.nextInt(); sc.nextLine();
                        System.out.print("Enter Name: ");
                        String name = sc.nextLine();
                        System.out.print("Enter Department: ");
                        String dept = sc.nextLine();
                        System.out.print("Enter Age: ");
                        int age = sc.nextInt(); sc.nextLine();
                        System.out.print("Is Active (true/false): ");
                        boolean active = sc.nextBoolean(); sc.nextLine();

                        String insertSQL = "INSERT INTO Students (student_id,
name, department, age, active) VALUES (?, ?, ?, ?, ?)";
                        try (PreparedStatement ps =
con.prepareStatement(insertSQL)) {
                            ps.setInt(1, id);
                            ps.setString(2, name);
                            ps.setString(3, dept);
                            ps.setInt(4, age);
                            ps.setBoolean(5, active);
                            ps.executeUpdate();
                            System.out.println("Student added successfully!");
                        }
                        break;

                    case 2:
                        String selectSQL = "SELECT * FROM Students";
                        try (Statement stmt = con.createStatement();
                             ResultSet rs = stmt.executeQuery(selectSQL)) {

                            System.out.println("\nAll Students:");
                            while (rs.next()) {
                                System.out.println(rs.getInt("student_id") + " | "
+
                                        rs.getString("name") + " | " +
                                        rs.getString("department") + " | " +
                                        rs.getInt("age") + " | " +
                                        rs.getBoolean("active"));
                            }
                        }
                        break;
                    case 3:
                        System.out.print("Enter Student ID to edit: ");
                        int editId = sc.nextInt(); sc.nextLine();
                        System.out.print("Enter new Name: ");
                        String newName = sc.nextLine();
                        System.out.print("Enter new Department: ");
                        String newDept = sc.nextLine();
                        System.out.print("Enter new Age: ");
                        int newAge = sc.nextInt(); sc.nextLine();
                        System.out.print("Is Active (true/false): ");
                        boolean newActive = sc.nextBoolean(); sc.nextLine();

                        String updateSQL = "UPDATE Students SET name=?, 
department=?, age=?, active=? WHERE student_id=?";
                        try (PreparedStatement ps = 
con.prepareStatement(updateSQL)) {
                            ps.setString(1, newName);
                            ps.setString(2, newDept);
                            ps.setInt(3, newAge);
                            ps.setBoolean(4, newActive);
                            ps.setInt(5, editId);
                            ps.executeUpdate();
                            System.out.println("Student updated successfully!");
                        }
                        break;

                    case 4:
                        System.out.print("Enter Student ID to delete: ");
                        int delId = sc.nextInt(); sc.nextLine();
                        String deleteSQL = "DELETE FROM Students WHERE 
student_id=?";
                        try (PreparedStatement ps = 
con.prepareStatement(deleteSQL)) {
                            ps.setInt(1, delId);
                            ps.executeUpdate();
                            System.out.println("Student deleted successfully!");
                        }
                        break;

                    case 5:
                        System.out.println("Exiting...");
                        break;

                    default:
                        System.out.println("Invalid choice!");
                }
            } while (choice != 5);

        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

Output:

```
$ javac -cp "lib/*" MySQL_DBNavigation.java
$ java -cp ".:lib/*" MySQL_DBNavigation

--- CollegeDB CRUD Menu ---
1. Add Student
2. View Students
3. Edit Student
4. Delete Student
5. Exit
Enter your choice: 2

All Students:
1 | Omkar Sonawane | Computer Engineering | 20 | true
2 | Raviraj Shingare | Civil Engineering | 19 | false
3 | Raj Sonawane | Computer Engineering | 21 | true
4 | Mahesh Salgar | Mechanical Engineering | 20 | true

--- CollegeDB CRUD Menu ---
1. Add Student
2. View Students
3. Edit Student
4. Delete Student
5. Exit
Enter your choice: 1
Enter Student ID: 5
Enter Name: Sudarshan Sonawane
Enter Department: ENTC Engineering
Enter Age: 23
Is Active (true/false): true
Student added successfully!

--- CollegeDB CRUD Menu ---
1. Add Student
2. View Students
3. Edit Student
4. Delete Student
5. Exit
Enter your choice: 2

All Students:
1 | Omkar Sonawane | Computer Engineering | 20 | true
2 | Raviraj Shingare | Civil Engineering | 19 | false
3 | Raj Sonawane | Computer Engineering | 21 | true
4 | Mahesh Salgar | Mechanical Engineering | 20 | true
5 | Sudarshan Sonawane | ENTC Engineering | 23 | true

--- CollegeDB CRUD Menu ---
1. Add Student
2. View Students
3. Edit Student
4. Delete Student
5. Exit
Enter your choice: 3
Enter Student ID to edit: 2
Enter new Name: Raviraj Shingare
Enter new Department: Civil Engineering
Enter new Age: 19
Is Active (true/false): true
Student updated successfully!

--- CollegeDB CRUD Menu ---
1. Add Student
2. View Students
3. Edit Student
4. Delete Student
5. Exit
Enter your choice: 2

All Students:
1 | Omkar Sonawane | Computer Engineering | 20 | true
```

```
2 | Raviraj Shingare | Civil Engineering | 19 | true
3 | Raj Sonawane | Computer Engineering | 21 | true
4 | Mahesh Salgar | Mechanical Engineering | 20 | true
5 | Sudarshan Sonawane | ENTC Engineering | 23 | true

--- CollegeDB CRUD Menu ---
1. Add Student
2. View Students
3. Edit Student
4. Delete Student
5. Exit
Enter your choice: 4
Enter Student ID to delete: 4
Student deleted successfully!

--- CollegeDB CRUD Menu ---
1. Add Student
2. View Students
3. Edit Student
4. Delete Student
5. Exit
Enter your choice: 2

All Students:
1 | Omkar Sonawane | Computer Engineering | 20 | true
2 | Raviraj Shingare | Civil Engineering | 19 | true
3 | Raj Sonawane | Computer Engineering | 21 | true
5 | Sudarshan Sonawane | ENTC Engineering | 23 | true

--- CollegeDB CRUD Menu ---
1. Add Student
2. View Students
3. Edit Student
4. Delete Student
5. Exit
Enter your choice: 5
Exiting...
```