SMARTBRIDGE JAVA BOOTSTRAP

20MIS0346

LAKSHMI G MENON

VIT, VELLORE

ASSIGNMENT -01

1. CREATE UPDATE DELETE USING SQL COMMANDS

```
-- create a table --
CREATE TABLE students (
   ID INTEGER PRIMARY KEY,
   NAME VARCHAR(30) NOT NULL,
   GENDER CHAR(1) NOT NULL
);

-- insert some values --
INSERT INTO students VALUES (1, 'Ryan', 'M');
INSERT INTO students VALUES (2, 'Joanna', 'F');
INSERT INTO students VALUES (3, 'Riaz', 'M');
INSERT INTO students VALUES (4, 'John', 'F');
INSERT INTO students VALUES (5, 'Alice', 'M');
INSERT INTO students VALUES (6, 'Bennet', 'F');
INSERT INTO students VALUES (7, 'Bacil', 'M');
INSERT INTO students VALUES (8, 'Josh', 'F');
INSERT INTO students VALUES (9, 'Rishab', 'M');
INSERT INTO students VALUES (10, 'Gill', 'F');
INSERT INTO students VALUES (11, 'Rahul', 'M');
INSERT INTO students VALUES (12, 'Jacob', 'F');

-- updating values --
UPDATE students
SET
   NAME = 'ROHIT',
   GENDER = 'M'
WHERE
   ID = 11;
```

```
SELECT
   *
FROM
   students;
-- deleting values --
 DROP TABLE students;


SELECT
   *
FROM
   students;
```

OUTPUT:

```
ID      NAME    GENDER
1       Ryan    M
2       Joanna  F
3       Riaz    M
4       John    F
5       Alice   M
6       Bennet  F
7       Bacil   M
8       Josh    F
9       Rishab  M
10      Gill    F
11      ROHIT   M
12      Jacob   F
ERROR 1146 (42S02) at line 31: Table 'mycompiler.students' doesn't
exist

[Execution complete with exit code 1]
```

2.CREATE TABLES AND APPLY JOINS.

```
CREATE TABLE members (
   member_id INT AUTO_INCREMENT,
   name VARCHAR(100),
   PRIMARY KEY (member_id)
);
```

```
CREATE TABLE committees (
    committee_id INT AUTO_INCREMENT,
    name VARCHAR(100),
    PRIMARY KEY (committee_id)
);

INSERT INTO members(name)
VALUES('John'),('Jane'),('Mary'),('David'),('Amelia');

INSERT INTO committees(name)
VALUES('John'),('Mary'),('Amelia'),('Joe');

-- members table --
SELECT
    *
FROM
    members;

-- committees table --
SELECT
    *
FROM
    committees;

-- inner join --
SELECT
    m.member_id,
    m.name AS member,
    c.committee_id,
    c.name AS committee
FROM
```

```
    members m
        INNER JOIN
    committees c ON c.name = m.name;

-- left join --
SELECT
    m.member_id,
    m.name AS member,
    c.committee_id,
    c.name AS committee
FROM
    members m
        LEFT JOIN
    committees c USING (name);

-- right join --
SELECT
    m.member_id,
    m.name AS member,
    c.committee_id,
    c.name AS committee
FROM
    members m
        RIGHT JOIN
    committees c ON c.name = m.name;

-- cross join --
SELECT
    m.member_id,
    m.name AS member,
    c.committee_id,
```

```
    c.name AS committee
FROM
    members m
        CROSS JOIN
    committees c;
```

OUTPUT:

```
member_id      name
1       John
2       Jane
3       Mary
4       David
5       Amelia

committee_id   name
1       John
2       Mary
3       Amelia
4       Joe

member_id      member  committee_id   committee
1       John    1       John
3       Mary    2       Mary
5       Amelia  3       Amelia

member_id      member  committee_id   committee
1       John    1       John
2       Jane    NULL    NULL
3       Mary    2       Mary
4       David   NULL    NULL
5       Amelia  3       Amelia

member_id      member  committee_id   committee
1       John    1       John
3       Mary    2       Mary
5       Amelia  3       Amelia
NULL    NULL    4       Joe

member_id      member  committee_id   committee
1       John    4       Joe
1       John    3       Amelia
1       John    2       Mary
1       John    1       John
2       Jane    4       Joe
2       Jane    3       Amelia
2       Jane    2       Mary
2       Jane    1       John
3       Mary    4       Joe
3       Mary    3       Amelia
```

```
3       Mary    2       Mary
3       Mary    1       John
4       David   4       Joe
4       David   3       Amelia
4       David   2       Mary
4       David   1       John
5       Amelia  4       Joe
5       Amelia  3       Amelia
5       Amelia  2       Mary
5       Amelia  1       John

[Execution complete with exit code 0]
```

3.CREATE UPDATE DELETE USING MONGO DB.

```
db.users.insertOne({ name: "Alice", age: 25, country: "USA" })

db.users.insertMany([
  { name: "Bob", age: 30, country: "Canada" },
  { name: "Charlie", age: 28, country: "UK" }
])


-- update operation --

db.users.updateOne(
   { name: "Alice" },
   { $set: { age: 30 } } // Update operation
)
db.users.updateMany(
   { country: "USA" }, // Filter to match the document(s) to update
   { $set: { status: "active" } } // Update operation
)


-- deleting operation --

db.users.deleteOne({ name: "Bob" }) // Filter to match the document to delete


db.users.deleteMany({ status: "inactive" }) // Filter to match the documents to delete
```

OUTPUT:

```
mycompiler_mongodb> {
  acknowledged: true,
  insertedId: ObjectId("647447f25a46ce02b900c6cf")
}
mycompiler_mongodb> ... ... ... {
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("647447f35a46ce02b900c6d0"),
    '1': ObjectId("647447f35a46ce02b900c6d1")
  }
}
mycompiler_mongodb>
mycompiler_mongodb> Uncaught:


mycompiler_mongodb> ... ... ... {
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
mycompiler_mongodb> ... ... ... {
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
mycompiler_mongodb>
mycompiler_mongodb> Uncaught:


mycompiler_mongodb> { acknowledged: true, deletedCount: 1 }
mycompiler_mongodb>
mycompiler_mongodb> { acknowledged: true, deletedCount: 0 }
mycompiler_mongodb>

[Execution complete with exit code 0]
```