

**Q1. Create a table named students with fields:**

- stdid INT PRIMARY KEY
- stdname VARCHAR(50)
- age INT
- city VARCHAR(50)

Ans:

```
[mysql> desc students;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| stdId | int           | NO   | PRI | NULL    |       |
| stdName | varchar(50)   | NO   |     | NULL    |       |
| Age    | int           | NO   |     | NULL    |       |
| City   | varchar(50)   | NO   |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.007 sec)
```

**Q2. Insert the following records into the students table:**

stdid	stdname	age	city
1	Rohan	20	Pune
2	Meera	22	Mumbai
3	Arjun	21	Delhi
4	Kavya	23	Pune
5	Neha	22	Kolkata

Ans:

```
mysql> INSERT INTO students (stdId, stdName, Age, City) VALUES
-> (1, 'Rohan', 20, 'Pune'),
-> (2, 'Meera', 22, 'Mumbai'),
-> (3, 'Arjun', 21, 'Delhi'),
-> (4, 'Kavya', 23, 'Pune'),
-> (5, 'Neha', 22, 'Kolkata');
Query OK, 5 rows affected (0.004 sec)
Records: 5 Duplicates: 0 Warnings: 0
```

**Q3. Display all student records.**

```
mysql> select * from students;
+-----+-----+-----+-----+
| stdId | stdName | Age | City |
+-----+-----+-----+-----+
|      1 | Rohan   | 20  | Pune |
|      2 | Meera   | 22  | Mumbai |
|      3 | Arjun   | 21  | Delhi |
|      4 | Kavya   | 23  | Pune |
|      5 | Neha    | 22  | Kolkata |
+-----+-----+-----+-----+
5 rows in set (0.000 sec)
```

Ans:

**Q4. Display only the name and age of all students.**

```
[mysql> select stdName, Age from students;
+-----+-----+
| stdName | Age |
+-----+-----+
| Rohan   | 20  |
| Meera   | 22  |
| Arjun   | 21  |
| Kavya   | 23  |
| Neha    | 22  |
+-----+-----+
5 rows in set (0.000 sec)
```

Ans:

**Q5. Display students who are from Pune.**

Ans:

```
[mysql> select * from students where City = 'Pune';
+-----+-----+-----+-----+
| stdId | stdName | Age | City |
+-----+-----+-----+-----+
|      1 | Rohan   | 20  | Pune |
|      4 | Kavya   | 23  | Pune |
+-----+-----+-----+-----+
2 rows in set (0.000 sec)
```

**Q6. Display students whose age is greater than 21.**

```
mysql> select * from students where Age > 21;
+-----+-----+-----+-----+
| stdId | stdName | Age | City |
+-----+-----+-----+-----+
|      2 | Meera   | 22  | Mumbai |
|      4 | Kavya   | 23  | Pune |
|      5 | Neha    | 22  | Kolkata |
+-----+-----+-----+-----+
3 rows in set (0.000 sec)
```

Ans:

Q7. Display students in descending order of age.

```
mysql> select * from students order by age desc;
+-----+-----+-----+-----+
| stdId | stdName | Age | City |
+-----+-----+-----+-----+
|      4 | Kavya   | 23  | Pune |
|      2 | Meera   | 22  | Mumbai |
|      5 | Neha    | 22  | Kolkata |
|      3 | Arjun   | 21  | Delhi |
|      1 | Rohan   | 20  | Pune |
+-----+-----+-----+-----+
5 rows in set (0.001 sec)
```

Ans:

Q8. Count how many students belong to each city. (Use GROUP BY)

```
mysql> SELECT City, COUNT(*) AS total_students
-> FROM students
-> GROUP BY City;
+-----+-----+
| City | total_students |
+-----+-----+
| Pune | 2 |
| Mumbai | 1 |
| Delhi | 1 |
| Kolkata | 1 |
+-----+-----+
4 rows in set (0.003 sec)
```

Ans:

**Q9. Display students whose name starts with 'K'. (Use LIKE)**

**Ans**

```
[mysql> select * from students where stdName like 'K%';
+-----+-----+-----+-----+
| stdId | stdName | Age | City |
+-----+-----+-----+-----+
|      4 | Kavya   | 23  | Pune |
+-----+-----+-----+-----+
1 row in set (0.001 sec)
```

**Q10. Delete student whose stdid = 5.**

```
[mysql> delete from students where stdId = 5;
Query OK, 1 row affected (0.002 sec)
```

```
[mysql> select * from students;
+-----+-----+-----+-----+
| stdId | stdName | Age | City |
+-----+-----+-----+-----+
|      1 | Rohan   | 20  | Pune |
|      2 | Meera   | 22  | Mumbai |
|      3 | Arjun   | 21  | Delhi |
|      4 | Kavya   | 23  | Pune |
+-----+-----+-----+-----+
4 rows in set (0.000 sec)
```

**PART – 2**

**Q11. Display student name and marks of only those students who have matching IDs in both tables.**

**(Students without marks should not appear.)**

**Ans:**

```
mysql> SELECT students.stdName, marks.marks from students inner join marks on students.stdId = marks.stdId;
+-----+-----+
| stdName | marks |
+-----+-----+
| Rohan   | 88    |
| Meera   | 76    |
| Arjun   | 92    |
+-----+-----+
3 rows in set (0.001 sec)
```

**Q12. Display all students and their marks.**

**(If marks not available, show NULL.)**

```
mysql> select students.stdName, marks.marks from students left join marks on students.stdId = marks.stdId;
+-----+-----+
| stdName | marks |
+-----+-----+
| Rohan   | 88    |
| Meera   | 76    |
| Arjun   | 92    |
| Kavya   | NULL  |
+-----+-----+
4 rows in set (0.001 sec)
```

**Q13. Display all marks records along with student names.**

**(If student doesn't exist in students table, show NULL.)**

```
mysql> SELECT students.stdName, marks.marks FROM students RIGHT JOIN marks ON students.stdId = marks.stdId;
+-----+-----+
| stdName | marks |
+-----+-----+
| Rohan   | 88    |
| Meera   | 76    |
| Arjun   | 92    |
| NULL    | 67    |
+-----+-----+
4 rows in set (0.001 sec)
```

**Q14. Display all possible combinations of students and subjects.**

**(Use CROSS JOIN between students and marks table to show every pair.)**

```
[mysql> SELECT students.stdName, marks.subject FROM students CROSS JOIN marks;
```

stdName	subject
Kavya	Maths
Arjun	Maths
Meera	Maths
Rohan	Maths
Kavya	Maths
Arjun	Maths
Meera	Maths
Rohan	Maths
Kavya	Maths
Arjun	Maths
Meera	Maths
Rohan	Maths
Kavya	Maths
Arjun	Maths
Meera	Maths
Rohan	Maths

```
16 rows in set (0.000 sec)
```

**Q15. Using INNER JOIN, display students who scored more than 80.**

```
[mysql> SELECT students.stdName, marks.marks FROM students INNER JOIN marks ON students.stdId = marks.stdId WHERE marks.marks > 80; |
```

stdName	marks
Rohan	88
Arjun	92

```
2 rows in set (0.001 sec)
```