Q1. Create a table named students with fields: • stdid INT PRIMARY KEY • stdname VARCHAR(50) • age INT • city VARCHAR(50)

Ans-

```
mysql> Create table student ( stdid int primary key , stdname varchar(50), age int , city varchar(50) );
Query OK, 0 rows affected (0.13 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 student 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 (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

Q3. Display all student records.

Ans-

```
mysql> select * from student;
+-------+---------+------+---------+
| 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 (0.00 sec)
```

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

Ans-

```
mysql> select stdname , age from student;
+---------+------+
| stdname | age  |
+---------+------+
| Rohan   |   20 |
| Meera   |   22 |
| Arjun   |   21 |
| Kavya   |   23 |
| Neha    |   22 |
+---------+------+
5 rows in set (0.00 sec)
```

Q5. Display students who are from Pune.

Ans-

```
mysql> select * from student where city = 'pune';
+-------+---------+------+------+
| stdid | stdname | age  | city |
+-------+---------+------+------+
|     1 | Rohan   |   20 | pune |
|     4 | Kavya   |   23 | Pune |
+-------+---------+------+------+
2 rows in set (0.01 sec)
```

Q6. Display students whose age is greater than 21.

Ans –

```
mysql> select * from student where age > 21;
+-------+---------+------+---------+
| stdid | stdname | age  | city    |
+-------+---------+------+---------+
|     2 | Meera   |   22 | Mumbai  |
|     4 | Kavya   |   23 | Pune    |
|     5 | Neha    |   22 | Kolkata |
+-------+---------+------+---------+
3 rows in set (0.00 sec)
```

Q7. Display students in descending order of age

Ans -

```
mysql> select * from student 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 (0.00 sec)
```

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

Q9. Display students whose name starts with ‘K’. (Use LIKE)\

Ans-

```
mysql> select  * from student where stdname like 'k%';
+-------+---------+------+------+
| stdid | stdname | age  | city |
+-------+---------+------+------+
|     4 | Kavya   |   23 | Pune |
+-------+---------+------+------+
1 row in set (0.00 sec)
```

Q10. Delete student whose stdid = 5.

Ans-

```
mysql> delete from student where stdid = 5;
Query OK, 1 row affected (0.01 sec)
```

PART – 2 Join Tables

Tables: Table: students stdid student_name city 1 Rohan Pune 2 Meera Mumbai 3 Arjun Delhi 4 Kavya Pune

Table: marks stdid subject marks 1 Maths 88 2 Maths 76 3 Maths 92 5 Maths 67

11. 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 a.stdname , b.marks from student as a inner join marks as b on a.stdid = b.stdid;
+---------+-------+
| stdname | marks |
+---------+-------+
| Rohan   |    88 |
| Meera   |    76 |
| Arjun   |    92 |
+---------+-------+
3 rows in set (0.00 sec)
```

LEFT JOIN

Q12. Display all students and their marks. (If marks not available, show NULL.)

Ans-

```
mysql> select a.stdname , b.marks from student as a left join marks as b on a.stdid = b.stdid;
+---------+-------+
| stdname | marks |
+---------+-------+
| Rohan   |    88 |
| Meera   |    76 |
| Arjun   |    92 |
| Kavya   |  NULL |
+---------+-------+
4 rows in set (0.00 sec)
```

RIGHT JOIN

Q13. Display all marks records along with student names. (If student doesn't exist in students table, show NULL.)

Ans-

```
mysql> select a.stdname , b.marks from student as a right join marks as b on a.stdid = b.stdid;
+---------+-------+
| stdname | marks |
+---------+-------+
| Rohan   |    88 |
| Meera   |    76 |
| Arjun   |    92 |
| NULL    |    67 |
+---------+-------+
4 rows in set (0.00 sec)
```

CROSS JOIN

Q14. Display all possible combinations of students and subjects. (Use CROSS JOIN between students and marks table to show every pair.)

```
mysql>  select a.stdname , b.subject from student as a cross join marks as b;
+---------+---------+
| 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 (0.00 sec)
```

JOIN with Filtering

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

Ans -

```
mysql> select student.stdname from student inner join marks on student.stdid = marks.stdid where marks > 80;
+---------+
| stdname |
+---------+
| Rohan   |
| Arjun   |
+---------+
2 rows in set (0.00 sec)
```