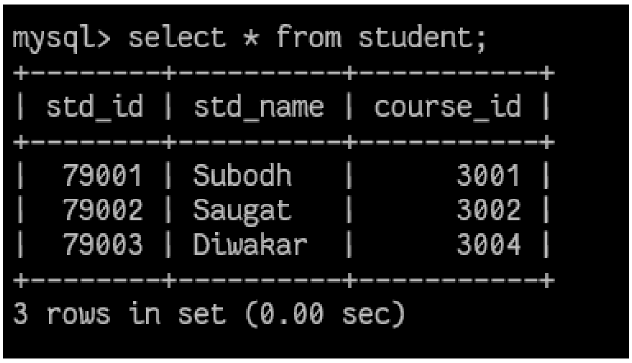
## Lab Report 3

Create two tables, Student and Courses. Perform different joins of both tables and display the output along with the SQL query.

***SQL Query:***

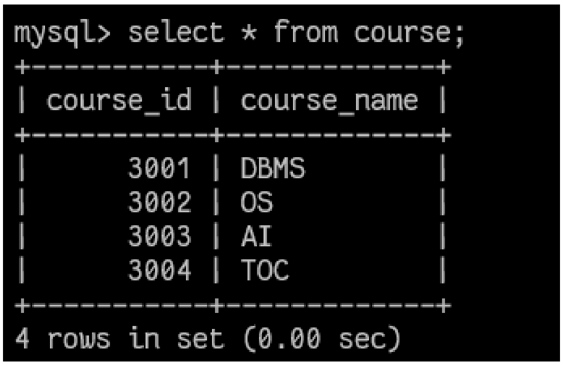
```
Create table course (
        course_id INT PRIMARY KEY,
        course_name  VARCHAR(40)
);

Create table student (
        std_id INT PRIMARY KEY,
        std_name  VARCHAR(40),
        course_id INT,
        FOREIGN KEY (course_id) REFERENCES course(course_id)
);

Insert into course (course_id, course_name)
Values
        (3001, “DBMS”),
        (3002, “OS”),
        (3003, “AI”),
        (3004, “TOC”);

Insert into students (std_id, std_name, course_id)
Values
        (79001, “Subodh”, 3001),
        (79002, “Saugat”, 3002),
        (79003, “Diwakar”, 3004);
```

*Table After Insertion of Data:*



```
mysql> select * from student;
+--------+----------+-----------+
| std_id | std_name | course_id |
+--------+----------+-----------+
|  79001 | Subodh   |      3001 |
|  79002 | Saugat   |      3002 |
|  79003 | Diwakar  |      3004 |
+--------+----------+-----------+
3 rows in set (0.00 sec)
```

***Student Table***



```
mysql> select * from course;
+-----------+-------------+
| course_id | course_name |
+-----------+-------------+
|      3001 | DBMS        |
|      3002 | OS          |
|      3003 | AI          |
|      3004 | TOC         |
+-----------+-------------+
4 rows in set (0.00 sec)
```

***Course Table***

**1. Inner Join**

***SQL Query:***

Select s.std_id, s.std_name, c.course_name

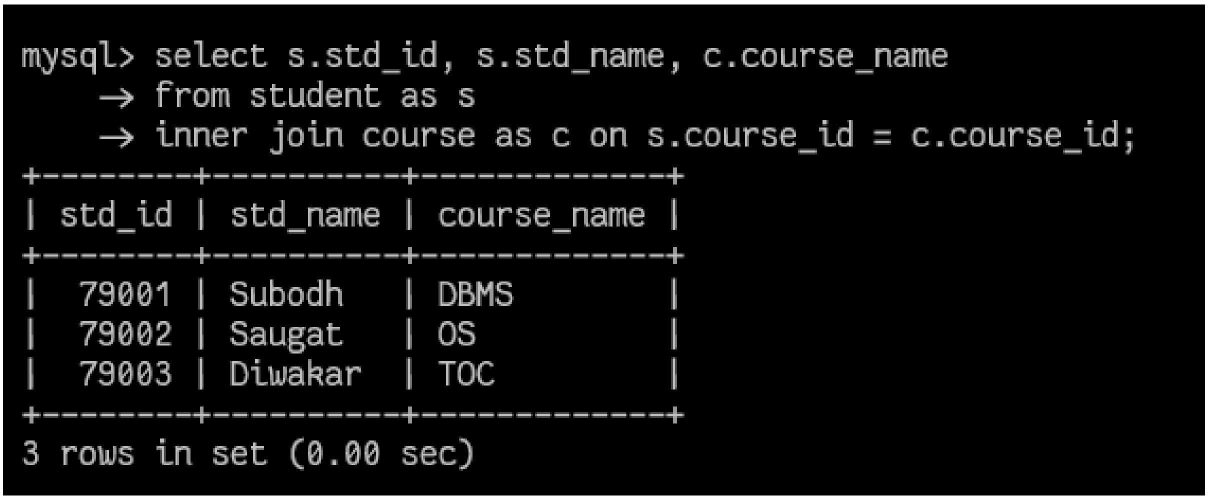
From student as s

Inner join course as c on s.course_id = c.course_id;

***NOTE*: *Here "as" is used to create an alias for a table.***

***Table after Inner Join:***

Here, only the rows where there is a match in both tables are shown.



```
mysql> select s.std_id, s.std_name, c.course_name
    -> from student as s
    -> inner join course as c on s.course_id = c.course_id;
+--------+----------+-------------+
| std_id | std_name | course_name |
+--------+----------+-------------+
|  79001 | Subodh   | DBMS        |
|  79002 | Saugat   | OS          |
|  79003 | Diwakar  | TOC         |
+--------+----------+-------------+
3 rows in set (0.00 sec)
```

### 2. Left Join

***SQL Query:***

Select s.std_id, s.std_name, c.course_name

From student as s

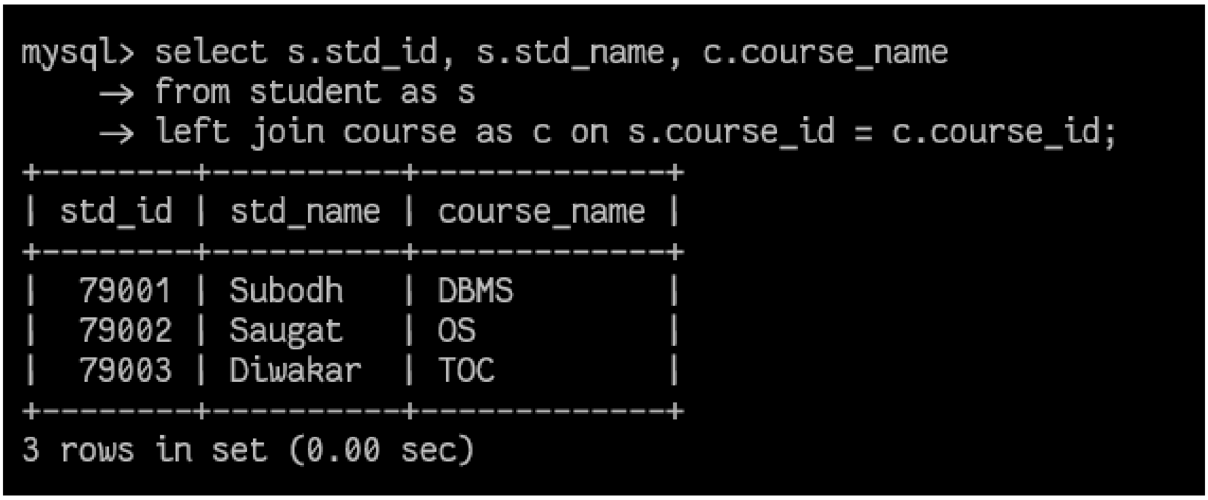
Left join course as c on s.course_id = c.course_id;

***NOTE*: *Here "as" is used to create an alias for a table.***

***Table after Left Join:***

Here, all records from the "student" table and matched records from "course" table are shown.

Here, the result is similar to the Inner join as the "course_id" values in the "student" table have matching entries in "course" table.



```
mysql> select s.std_id, s.std_name, c.course_name
    -> from student as s
    -> left join course as c on s.course_id = c.course_id;
+--------+----------+-------------+
| std_id | std_name | course_name |
+--------+----------+-------------+
|  79001 | Subodh   | DBMS        |
|  79002 | Saugat   | OS          |
|  79003 | Diwakar  | TOC         |
+--------+----------+-------------+
3 rows in set (0.00 sec)
```

### 3. Right Join

***SQL Query:***

Select s.std_id, s.std_name, c.course_name

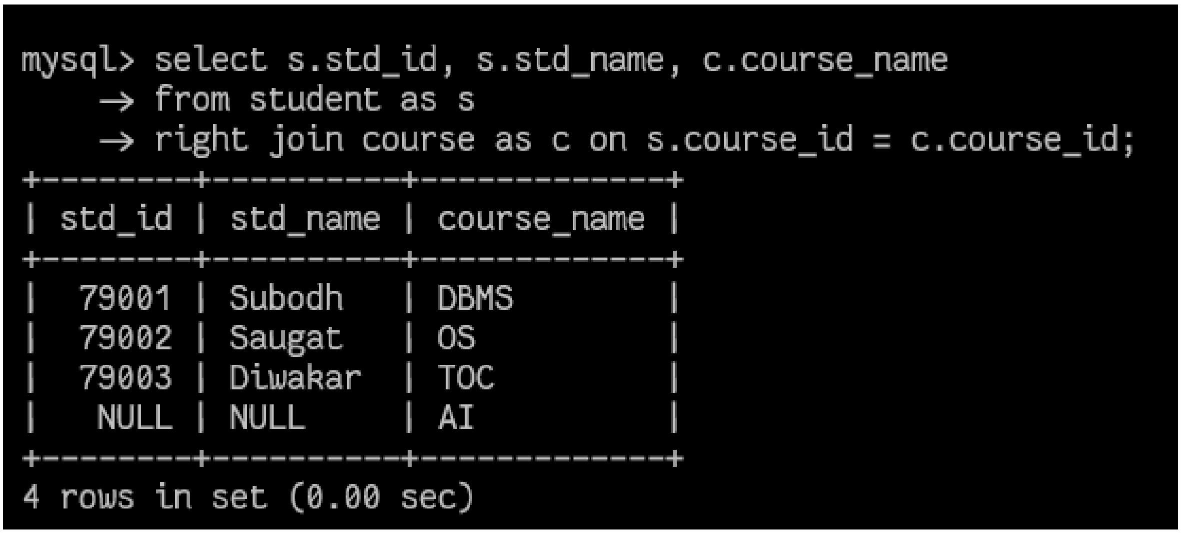
From student as s

Right join course as c on s.course_id = c.course_id;

***Table after Right Join:***

Here, all records from the “course” table and matched records from “student” table are shown.

Here, “AI” has no corresponding student, so the columns “std_id” and “std_name” are NULL.



```
mysql> select s.std_id, s.std_name, c.course_name
    -> from student as s
    -> right join course as c on s.course_id = c.course_id;
+--------+----------+-------------+
| std_id | std_name | course_name |
+--------+----------+-------------+
|  79001 | Subodh   | DBMS        |
|  79002 | Saugat   | OS          |
|  79003 | Diwakar  | TOC         |
|   NULL | NULL     | AI          |
+--------+----------+-------------+
4 rows in set (0.00 sec)
```