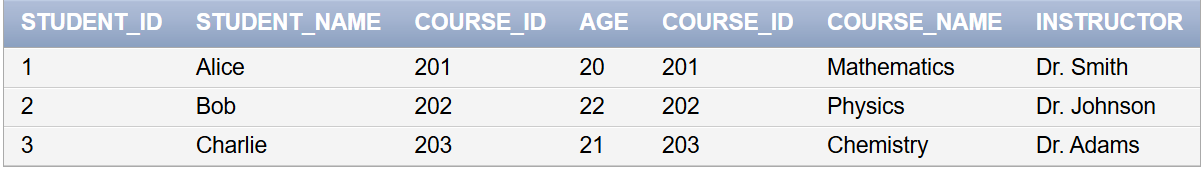
**Exp No 07: Questions:**

Name:-Saeel Sakhalkar   
Roll No: 2439   
**Basic JOIN Queries (1-5)**   
 1.**Retrieve all student details along with their course names using an INNER JOIN.**  o *Hint:* Use INNER JOIN on course\_id.

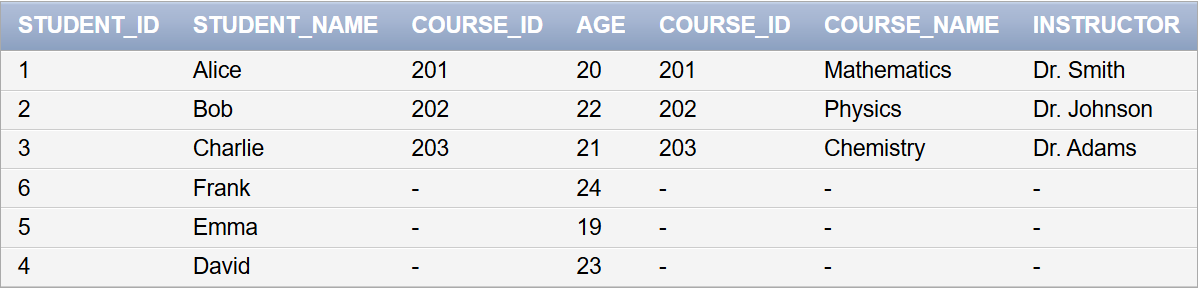
**SELECT \* FROM STUDENTS S INNER JOIN COURSES C ON S.COURSE\_ID=C.COURSE\_ID**



2.**Get a list of students and their assigned courses, ensuring that even students with no assigned**  **course appear in the result (use LEFT JOIN).**

o *Hint:* Use LEFT JOIN.

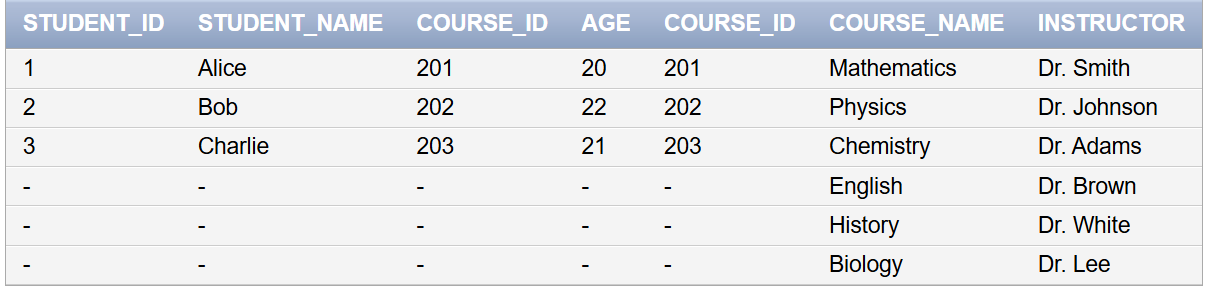
**SELECT \* FROM STUDENTS S LEFT JOIN COURSES C ON S.COURSE\_ID=C.COURSE\_ID**



3.**Retrieve all courses and the students enrolled in them, ensuring that even courses with no**  **students appear in the result (use RIGHT JOIN).**

o *Hint:* Use RIGHT JOIN.

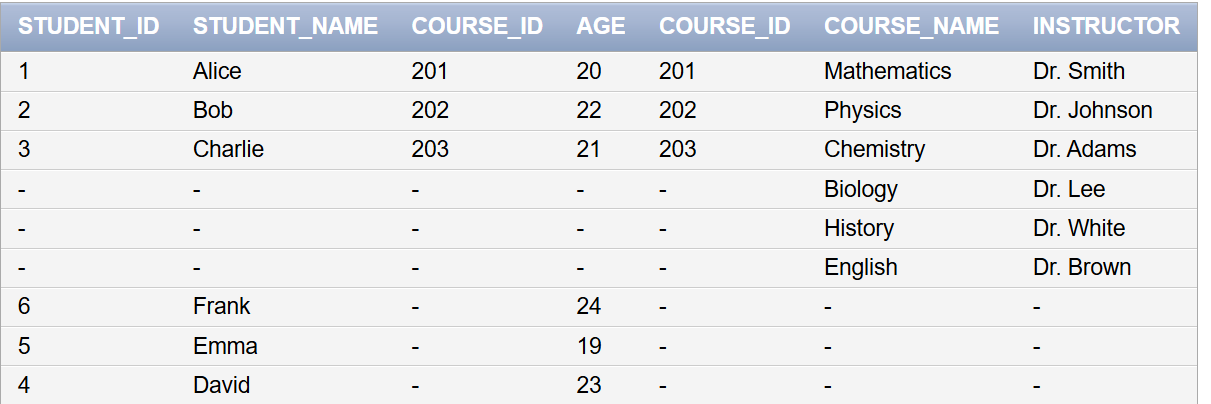
**SELECT \* FROM STUDENTS S RIGHT JOIN COURSES C ON S.COURSE\_ID=C.COURSE\_ID**



4.**List all students and courses, including those without a match in both tables (use FULL OUTER**  **JOIN).**

o *Hint:* Use FULL OUTER JOIN.

**SELECT \* FROM STUDENTS S FULL JOIN COURSES C ON S.COURSE\_ID=C.COURSE\_ID**



5.**Find all possible student-course combinations using a CROSS JOIN.**

o *Hint:* Use CROSS JOIN without an ON condition.

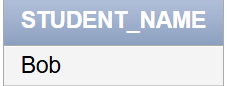
**SELECT \* FROM STUDENTS S CROSS JOIN COURSES C**



**JOINs with WHERE Condition (6-10)**   
 6.**Retrieve students enrolled in 'Physics'.**

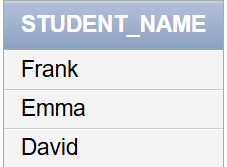
o *Hint:* Use INNER JOIN and WHERE course\_name = 'Physics'.

**SELECT STUDENT\_NAME FROM STUDENTS S INNER JOIN COURSES C ON S.COURSE\_ID=C.COURSE\_ID WHERE COURSE\_NAME='Physics';**



7.**List students who have not been assigned a course (use LEFT JOIN and check NULL values).**  o *Hint:* Use LEFT JOIN and WHERE course\_id IS NULL.

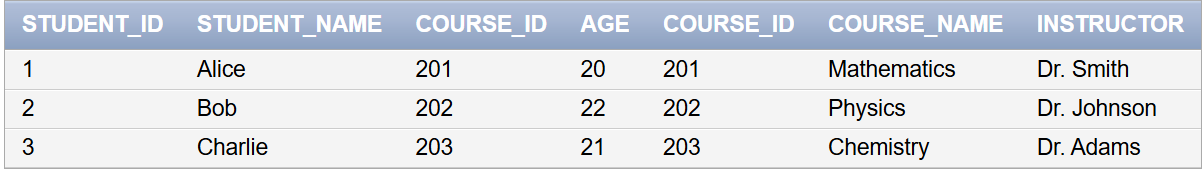
**SELECT STUDENT\_NAME FROM STUDENTS S LEFT JOIN COURSES C ON S.COURSE\_ID=C.COURSE\_ID WHERE S.COURSE\_ID IS NULL;**



8.**Retrieve courses that have at least one student enrolled (use INNER JOIN).**

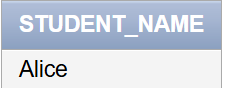
o *Hint:* Use INNER JOIN and check non-null student names.

**SELECT \* FROM STUDENTS S INNER JOIN COURSES C ON S.COURSE\_ID=C.COURSE\_ID WHERE S.STUDENT\_NAME IS NOT NULL;**



9.**Find all students enrolled in a course taught by 'Dr. Smith'.**

o *Hint:* Use INNER JOIN and filter with WHERE instructor = 'Dr. Smith'. **SELECT STUDENT\_NAME FROM STUDENTS S INNER JOIN COURSES C ON S.COURSE\_ID=C.COURSE\_ID WHERE C.INSTRUCTOR='Dr. Smith';**

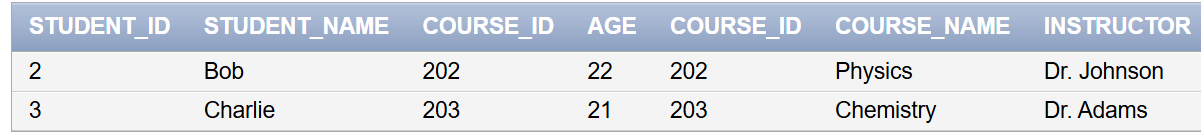


10.**Find students who are 21 years or older and have been assigned a course.**

● *Hint:* Use INNER JOIN with WHERE age >= 21.

**SELECT \* FROM STUDENTS S INNER JOIN COURSES C ON S.COURSE\_ID=C.COURSE\_ID WHERE**

**S.AGE>=21;**



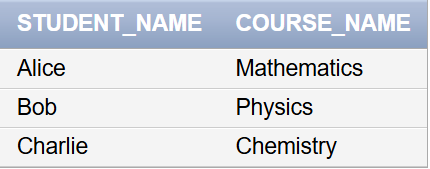
**JOINs with ORDER BY Clause (11-15)**

11.**List students and their courses, sorted by student name in ascending order.**

● *Hint:* Use ORDER BY student\_name ASC.

**SELECT STUDENT\_NAME,COURSE\_NAME FROM STUDENTS S INNER JOIN COURSES C ON**

**S.COURSE\_ID=C.COURSE\_ID ORDER BY S.STUDENT\_NAME ASC;**

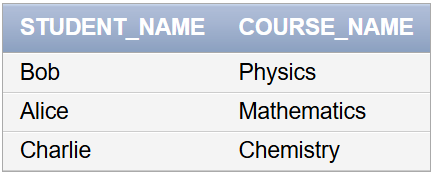


12.**List courses with their assigned students, sorted by course name in descending**

**order.**● *Hint:* Use ORDER BY course\_name DESC.

**SELECT STUDENT\_NAME,COURSE\_NAME FROM STUDENTS S INNER JOIN COURSES C ON**

**S.COURSE\_ID=C.COURSE\_ID ORDER BY COURSE\_NAME DESC;**



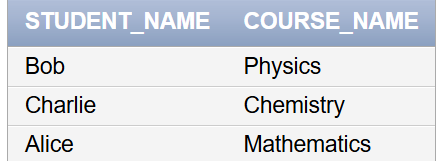
13.**Retrieve students along with their assigned courses, ordered by student age (oldest to**

**youngest).**

● *Hint:* Use ORDER BY age DESC.

**SELECT STUDENT\_NAME,COURSE\_NAME FROM STUDENTS S INNER JOIN COURSES C ON**

**S.COURSE\_ID=C.COURSE\_ID ORDER BY AGE DESC;**

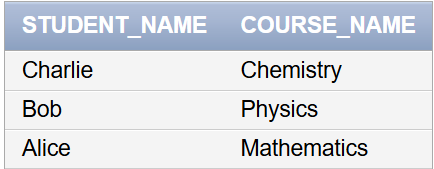


14.**List students who have a course assigned, ordered by the course instructor name.**●

*Hint:* Use ORDER BY instructor ASC.

**SELECT STUDENT\_NAME,COURSE\_NAME FROM STUDENTS S INNER JOIN COURSES C ON**

**S.COURSE\_ID=C.COURSE\_ID ORDER BY INSTRUCTOR ASC;**



15.**Find all student-course combinations from the CROSS JOIN, sorted by student name first and**

**then by course name.**

● *Hint:* Use ORDER BY student\_name, course\_name.

**SELECT \* FROM STUDENTS S CROSS JOIN COURSES C ORDER BY**

**S.STUDENT\_NAME,C.COURSE\_NAME;**

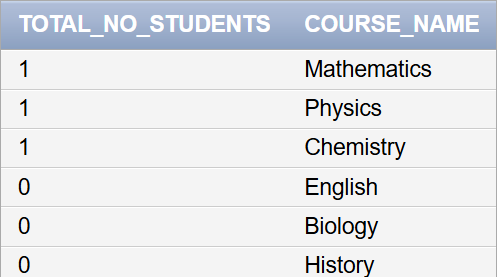


**JOINs with GROUP BY and HAVING (16-20)**

16.**Find the total number of students enrolled in each course (use GROUP BY).**

● *Hint:* Use COUNT(student\_id) GROUP BY course\_id.

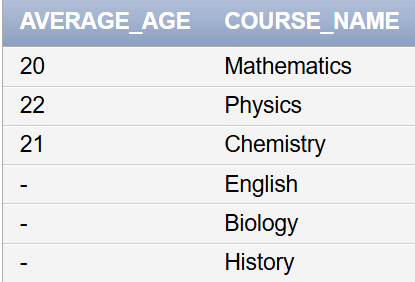
**SELECT COUNT(student\_id) AS TOTAL\_NO\_sTUDENTS,COURSE\_NAME FROM STUDENTS S RIGHT JOIN COURSES C ON S.COURSE\_ID=C.COURSE\_ID GROUP BY COURSE\_NAME**



17.**Find the average age of students enrolled in each course (use GROUP BY).**

● *Hint:* Use AVG(age) GROUP BY course\_id.

**SELECT AVG(AGE) AS AVERAGE\_AGE,COURSE\_NAME FROM STUDENTS S RIGHT JOIN COURSES C ON S.COURSE\_ID=C.COURSE\_ID GROUP BY COURSE\_NAME**



18.**Show only those courses where more than one student is enrolled.**

● *Hint:* Use HAVING COUNT(student\_id) > 1.

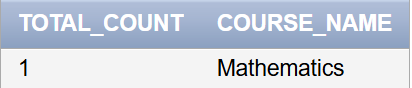
**SELECT COUNT(STUDENT\_ID) AS TOTAL\_COUNT,COURSE\_NAME FROM STUDENTS S INNER JOIN COURSES C ON S.COURSE\_ID=C.COURSE\_ID GROUP BY COURSE\_NAME HAVING COUNT(STUDENT\_ID) > 1**



19.**Find the course with the highest number of students enrolled.**

● *Hint:* Use GROUP BY and ORDER BY COUNT(student\_id) DESC LIMIT 1.

**SELECT TOTAL\_COUNT, COURSE\_NAME FROM (SELECT COUNT(S.STUDENT\_ID) AS**   
**TOTAL\_COUNT, C.COURSE\_NAME FROM STUDENTS S INNER JOIN COURSES C ON S.COURSE\_ID = C.COURSE\_ID GROUP BY C.COURSE\_NAME ORDER BY TOTAL\_COUNT DESC) WHERE**   
**ROWNUM = 1;**



20.**Find the average age of students per course, but only for courses where the average age is**  **greater than 21.**

● *Hint:* Use HAVING AVG(age) > 21.

**SELECT AVG(AGE) AS AVG\_AGE,COURSE\_NAME FROM STUDENTS S INNER JOIN COURSES C ON S.COURSE\_ID=C.COURSE\_ID GROUP BY COURSE\_NAME HAVING AVG(AGE)> 21**

