Exp No 07: Questions:

Name:-Saeel Sakhalkar

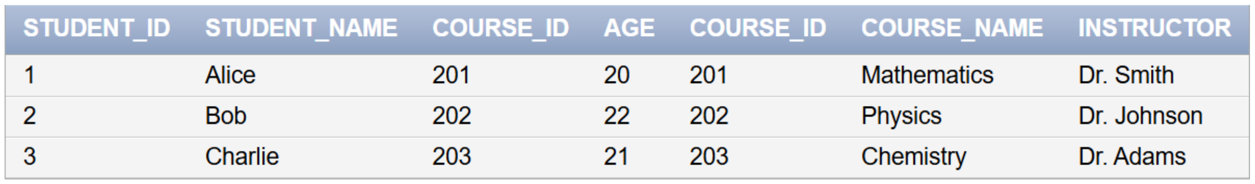
Roll No: 2439

Basic JOIN Queries (1-5)

1.**Retrieve all student details along with their course names using an INNER JOIN.**

- *Hint:* Use INNER JOIN on course_id.

SELECT * FROM STUDENTS S INNER JOIN COURSES C ON S.COURSE_ID=C.COURSE_ID

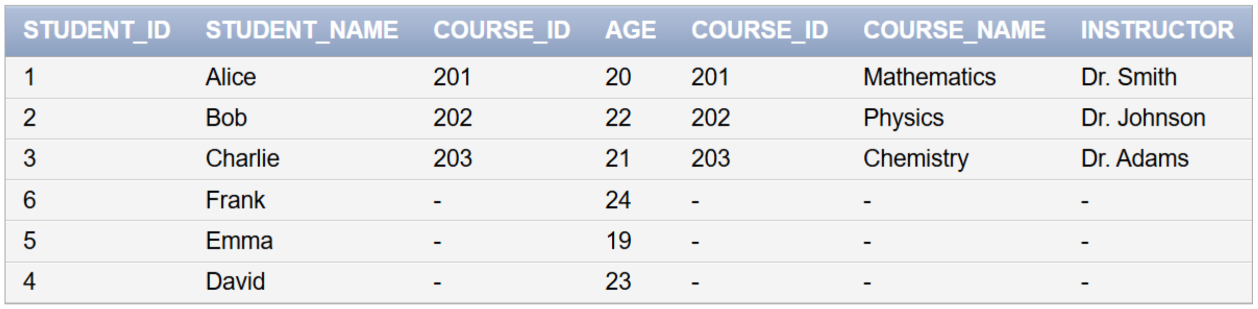


STUDENT_ID	STUDENT_NAME	COURSE_ID	AGE	COURSE_ID	COURSE_NAME	INSTRUCTOR
1	Alice	201	20	201	Mathematics	Dr. Smith
2	Bob	202	22	202	Physics	Dr. Johnson
3	Charlie	203	21	203	Chemistry	Dr. Adams

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).**

- *Hint:* Use LEFT JOIN.

SELECT * FROM STUDENTS S LEFT JOIN COURSES C ON S.COURSE_ID=C.COURSE_ID

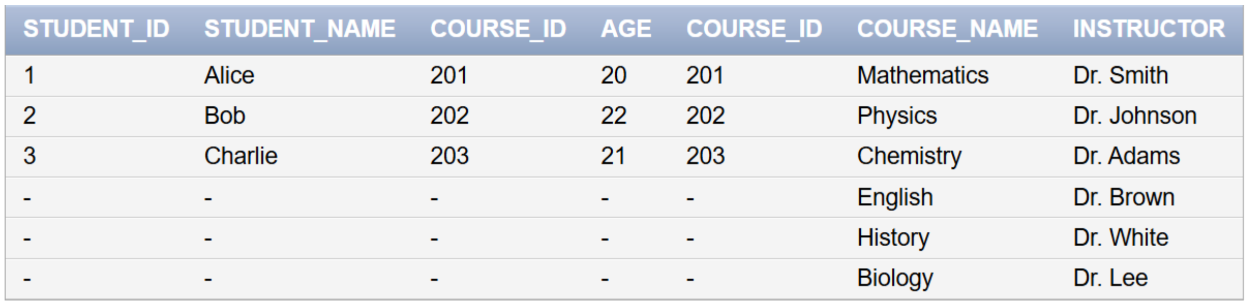


STUDENT_ID	STUDENT_NAME	COURSE_ID	AGE	COURSE_ID	COURSE_NAME	INSTRUCTOR
1	Alice	201	20	201	Mathematics	Dr. Smith
2	Bob	202	22	202	Physics	Dr. Johnson
3	Charlie	203	21	203	Chemistry	Dr. Adams
6	Frank	-	24	-	-	-
5	Emma	-	19	-	-	-
4	David	-	23	-	-	-

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

- *Hint:* Use RIGHT JOIN.

SELECT * FROM STUDENTS S RIGHT JOIN COURSES C ON S.COURSE_ID=C.COURSE_ID

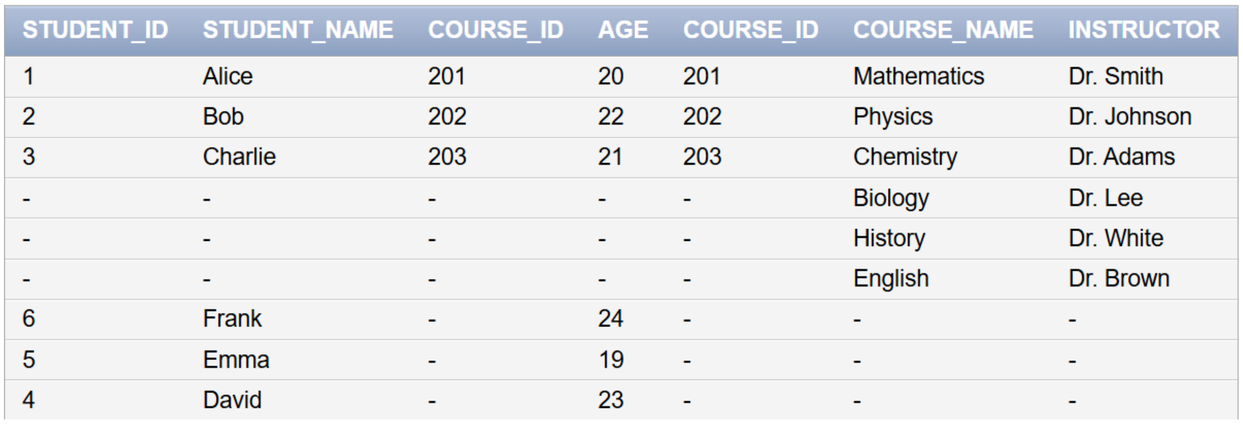


STUDENT_ID	STUDENT_NAME	COURSE_ID	AGE	COURSE_ID	COURSE_NAME	INSTRUCTOR
1	Alice	201	20	201	Mathematics	Dr. Smith
2	Bob	202	22	202	Physics	Dr. Johnson
3	Charlie	203	21	203	Chemistry	Dr. Adams
-	-	-	-	-	English	Dr. Brown
-	-	-	-	-	History	Dr. White
-	-	-	-	-	Biology	Dr. Lee

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

- *Hint:* Use FULL OUTER JOIN.

SELECT * FROM STUDENTS S FULL JOIN COURSES C ON S.COURSE_ID=C.COURSE_ID

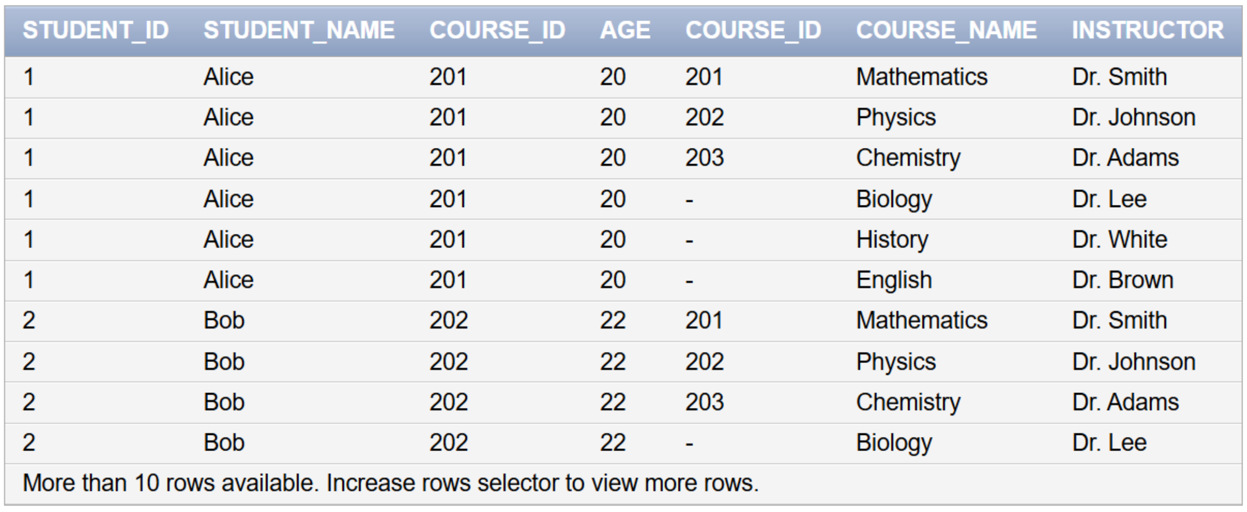


STUDENT_ID	STUDENT_NAME	COURSE_ID	AGE	COURSE_ID	COURSE_NAME	INSTRUCTOR
1	Alice	201	20	201	Mathematics	Dr. Smith
2	Bob	202	22	202	Physics	Dr. Johnson
3	Charlie	203	21	203	Chemistry	Dr. Adams
-	-	-	-	-	Biology	Dr. Lee
-	-	-	-	-	History	Dr. White
-	-	-	-	-	English	Dr. Brown
6	Frank	-	24	-	-	-
5	Emma	-	19	-	-	-
4	David	-	23	-	-	-

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

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

SELECT * FROM STUDENTS S CROSS JOIN COURSES C



STUDENT_ID	STUDENT_NAME	COURSE_ID	AGE	COURSE_ID	COURSE_NAME	INSTRUCTOR
1	Alice	201	20	201	Mathematics	Dr. Smith
1	Alice	201	20	202	Physics	Dr. Johnson
1	Alice	201	20	203	Chemistry	Dr. Adams
1	Alice	201	20	-	Biology	Dr. Lee
1	Alice	201	20	-	History	Dr. White
1	Alice	201	20	-	English	Dr. Brown
2	Bob	202	22	201	Mathematics	Dr. Smith
2	Bob	202	22	202	Physics	Dr. Johnson
2	Bob	202	22	203	Chemistry	Dr. Adams
2	Bob	202	22	-	Biology	Dr. Lee

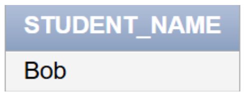
More than 10 rows available. Increase rows selector to view more rows.

JOINs with WHERE Condition (6-10)

6.**Retrieve students enrolled in 'Physics'.**

- *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';

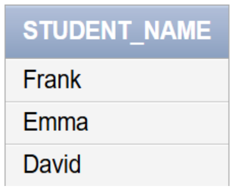


STUDENT_NAME
Bob

7.**List students who have not been assigned a course (use LEFT JOIN and check NULL values).**

- *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;

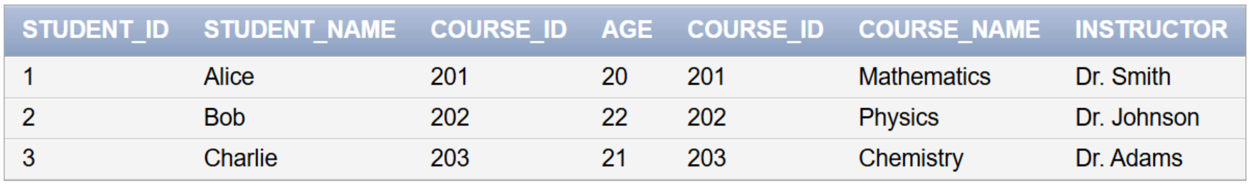


STUDENT_NAME
Frank
Emma
David

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

- *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;

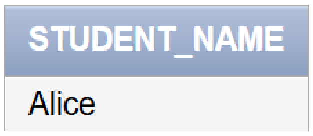


STUDENT_ID	STUDENT_NAME	COURSE_ID	AGE	COURSE_ID	COURSE_NAME	INSTRUCTOR
1	Alice	201	20	201	Mathematics	Dr. Smith
2	Bob	202	22	202	Physics	Dr. Johnson
3	Charlie	203	21	203	Chemistry	Dr. Adams

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

- *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';

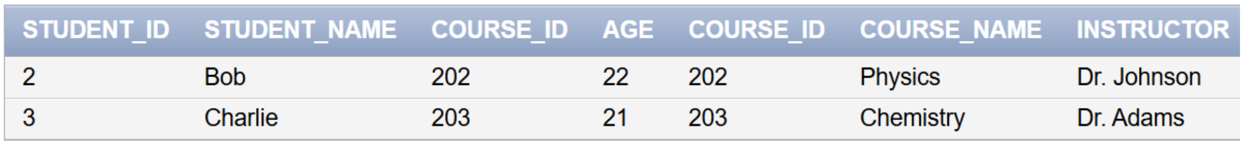


STUDENT_NAME
Alice

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;



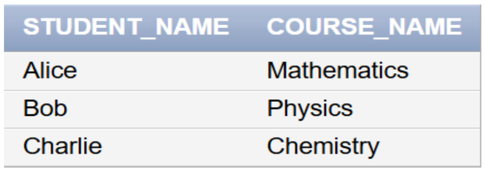
STUDENT_ID	STUDENT_NAME	COURSE_ID	AGE	COURSE_ID	COURSE_NAME	INSTRUCTOR
2	Bob	202	22	202	Physics	Dr. Johnson
3	Charlie	203	21	203	Chemistry	Dr. Adams

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.

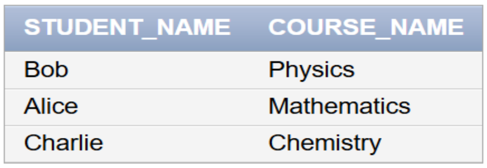
SELECT STUDENT_NAME,COURSE_NAME FROM STUDENTS S INNER JOIN COURSES C ON S.COURSE_ID=C.COURSE_ID ORDER BY S.STUDENT_NAME ASC;



STUDENT_NAME	COURSE_NAME
Alice	Mathematics
Bob	Physics
Charlie	Chemistry

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;

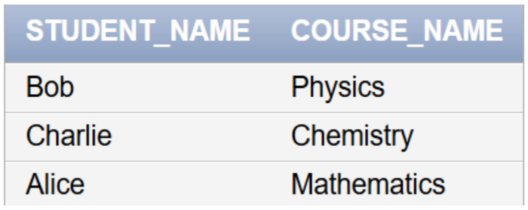


STUDENT_NAME	COURSE_NAME
Bob	Physics
Alice	Mathematics
Charlie	Chemistry

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

- *Hint:* Use ORDER BY age DESC.

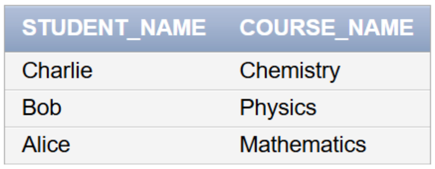
SELECT STUDENT_NAME,COURSE_NAME FROM STUDENTS S INNER JOIN COURSES C ON S.COURSE_ID=C.COURSE_ID ORDER BY AGE DESC;



STUDENT_NAME	COURSE_NAME
Bob	Physics
Charlie	Chemistry
Alice	Mathematics

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;

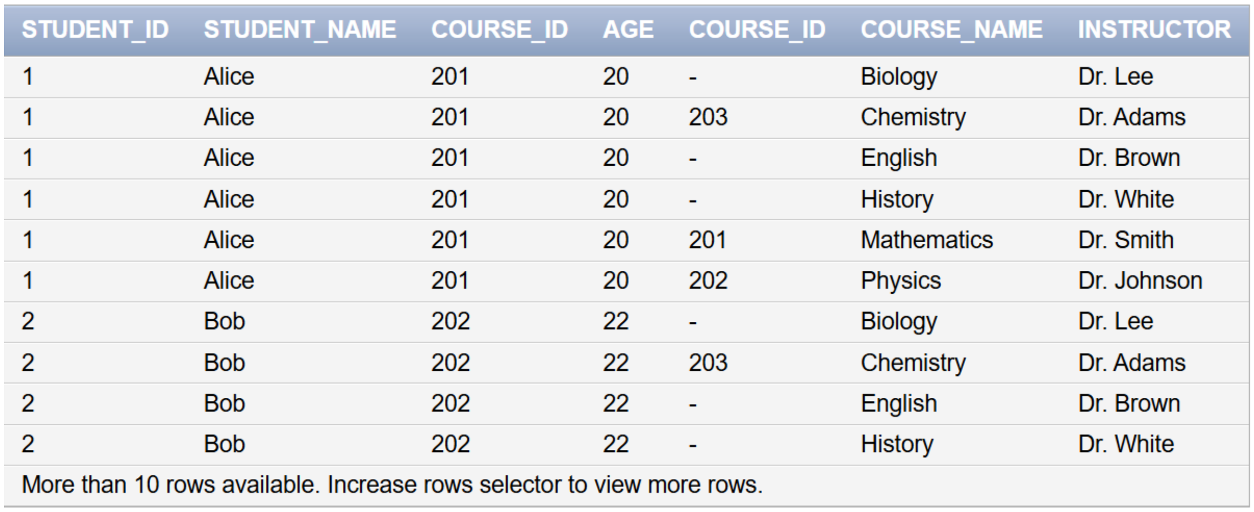


STUDENT_NAME	COURSE_NAME
Charlie	Chemistry
Bob	Physics
Alice	Mathematics

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;



STUDENT_ID	STUDENT_NAME	COURSE_ID	AGE	COURSE_ID	COURSE_NAME	INSTRUCTOR
1	Alice	201	20	-	Biology	Dr. Lee
1	Alice	201	20	203	Chemistry	Dr. Adams
1	Alice	201	20	-	English	Dr. Brown
1	Alice	201	20	-	History	Dr. White
1	Alice	201	20	201	Mathematics	Dr. Smith
1	Alice	201	20	202	Physics	Dr. Johnson
2	Bob	202	22	-	Biology	Dr. Lee
2	Bob	202	22	203	Chemistry	Dr. Adams
2	Bob	202	22	-	English	Dr. Brown
2	Bob	202	22	-	History	Dr. White

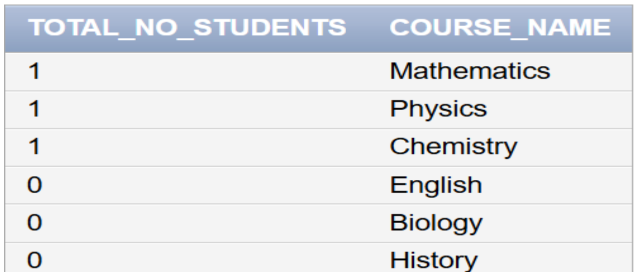
More than 10 rows available. Increase rows selector to view more rows.

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

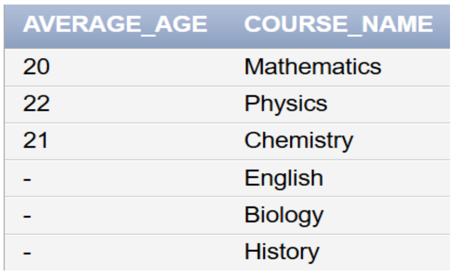


TOTAL_NO_STUDENTS	COURSE_NAME
1	Mathematics
1	Physics
1	Chemistry
0	English
0	Biology
0	History

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



AVERAGE_AGE	COURSE_NAME
20	Mathematics
22	Physics
21	Chemistry
-	English
-	Biology
-	History

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

- *Hint:* Use HAVING COUNT(student_id) > 1.

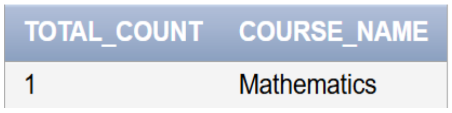
SELECT COUNT(STUDENT_ID) AS TOTAL_COUNT,COURSE_NAME FROM STUDENTS S INNER JOIN COURSES C ON S.COURSE_ID=C.COURSE_ID GROUP BY COURSE_NAME HAVING COUNT(STUDENT_ID) > 1

no data found

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;

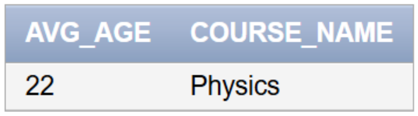


TOTAL_COUNT	COURSE_NAME
1	Mathematics

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



AVG_AGE	COURSE_NAME
22	Physics