

### EXISTING STUDENTS TABLE

STUDENT_ID	FIRST_NAME	LAST_NAME	DATE_OF_BIRTH	ENROLLMENT_DATE	SEMESTER	FEE	PROGRAM_ID	AGE	DUE_STATUS	RANKING	SUPERVISOR_ID
1	105	Hassan Ali	25-NOV-02	01-APR-24	2	57834	3	22	cleared	1	6
2	110	Nida Malik	30-JUN-03	18-MAR-24	2	52000	5	21	cleared	6	5
3	106	Aisha Khan	15-MAR-03	01-SEP-23	4	48000	1	21	cleared	2	4
4	107	Ali Raza	22-JUL-01	15-AUG-22	6	55000	2	23	cleared	3	3
5	108	Sara Ahmed	10-JAN-04	10-FEB-24	2	60000	3	20	cleared	4	2
6	109	Bilal Shah	05-DEC-02	12-JAN-23	4	45000	4	22	cleared	5	1

### EXISTING PROGRAMS TABLE

PROGRAM_ID	PROGRAM_NAME
1	1 Computer Science
2	2 Data Structures
3	3 Software Engineering
4	4 Business Administration
5	5 Information Technology

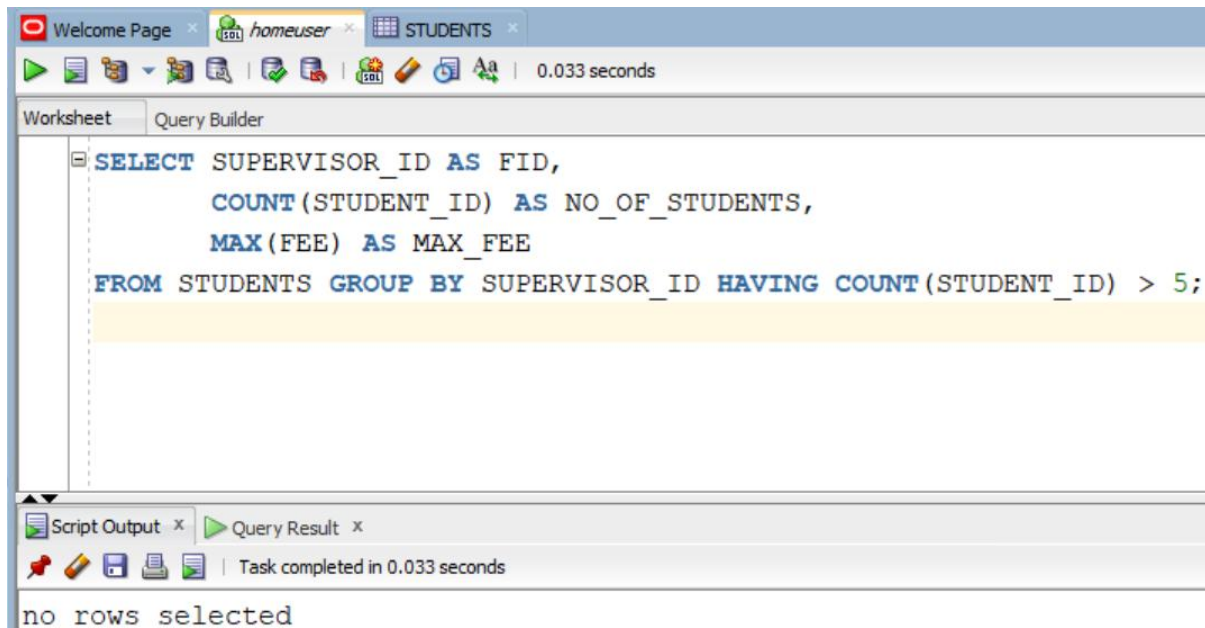
### EXISTING FACULTY TABLE

FID	FNAME	DESIGNATION	PROGRAM_ID
1	1 Dr. Ahmed Khan	Professor	1
2	2 Dr. Sara Malik	Associate Prof	2
3	3 Dr. Bilal Hussain	Assistant Prof	3
4	4 Dr. Ayesha Raza	Lecturer	4
5	5 Dr. Hamid Ali	Professor	5
6	6 Dr. Nadia Sheikh	Lecturer	1

### LAB#04 – TASK#01

Worksheet		
Query Builder		
<pre>SELECT SEMESTER, SUM(FEE) AS TOTAL_FEE, ROUND((SUM(FEE) / (SELECT SUM(FEE) FROM STUDENTS))*100, 2) AS PERCENTAGE FROM STUDENTS GROUP BY SEMESTER;</pre>		
Script Output x Query Result x		
Task completed in 0.035 seconds		
SEMESTER	TOTAL_FEE	PERCENTAGE
6	55000	17.3
2	169834	53.43
4	93000	29.26

#### LAB#04 – TASK#02



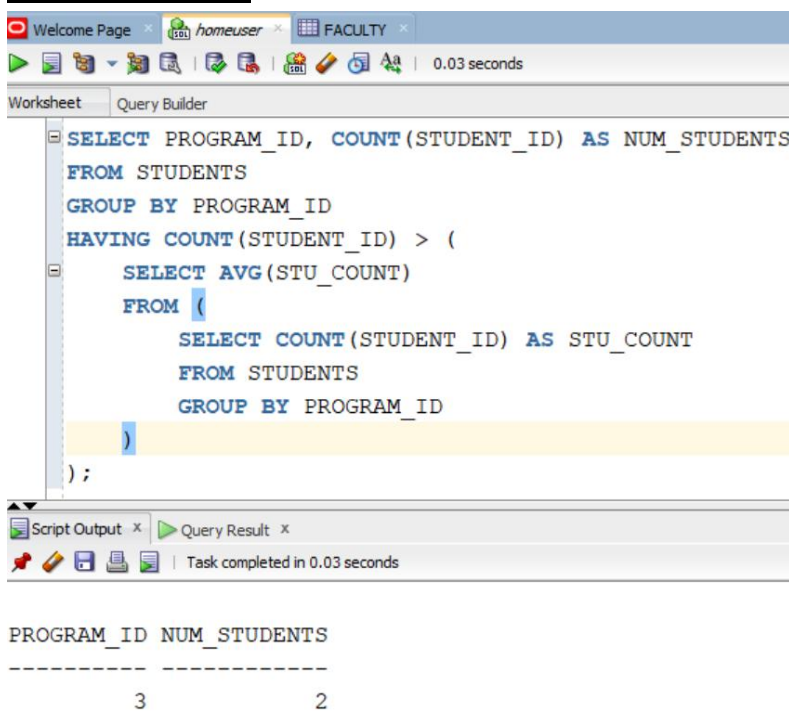
The screenshot shows the SQL Developer interface with the 'STUDENTS' table selected. The query in the Query Builder is:

```
SELECT SUPERVISOR_ID AS FID,  
       COUNT(STUDENT_ID) AS NO_OF_STUDENTS,  
       MAX(FEE) AS MAX_FEE  
FROM STUDENTS GROUP BY SUPERVISOR_ID HAVING COUNT(STUDENT_ID) > 5;
```

The Script Output pane shows the message: "no rows selected".

Since not enough students in table

#### LAB#04 – TASK#03



The screenshot shows the SQL Developer interface with the 'FACULTY' table selected. The query in the Query Builder is:

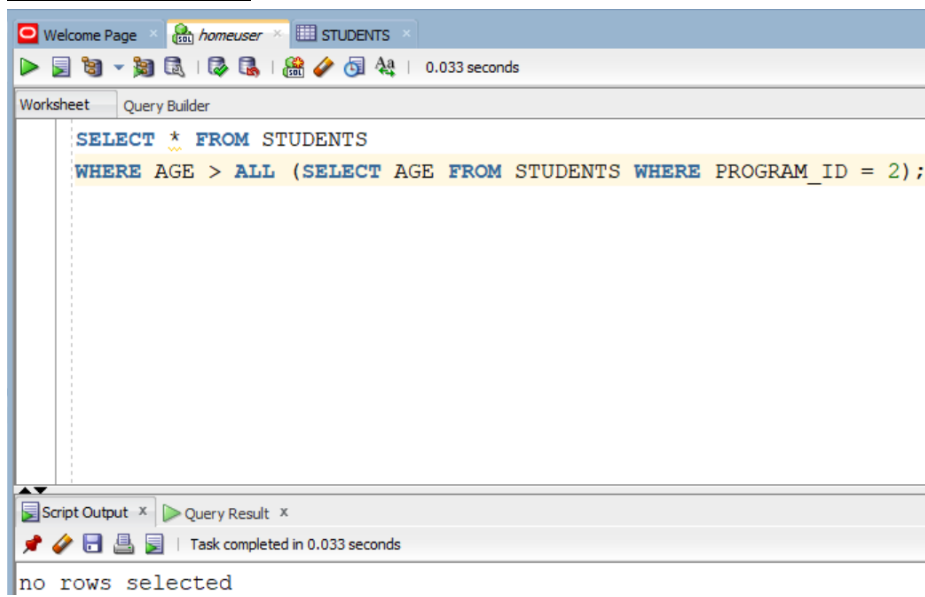
```
SELECT PROGRAM_ID, COUNT(STUDENT_ID) AS NUM_STUDENTS  
FROM STUDENTS  
GROUP BY PROGRAM_ID  
HAVING COUNT(STUDENT_ID) > (  
    SELECT AVG(STU_COUNT)  
    FROM (  
        SELECT COUNT(STUDENT_ID) AS STU_COUNT  
        FROM STUDENTS  
        GROUP BY PROGRAM_ID  
    )  
);
```

The Script Output pane shows the message: "Task completed in 0.03 seconds".

The Query Result pane displays the following table:

PROGRAM_ID	NUM_STUDENTS
3	2

#### LAB#04 – TASK#04



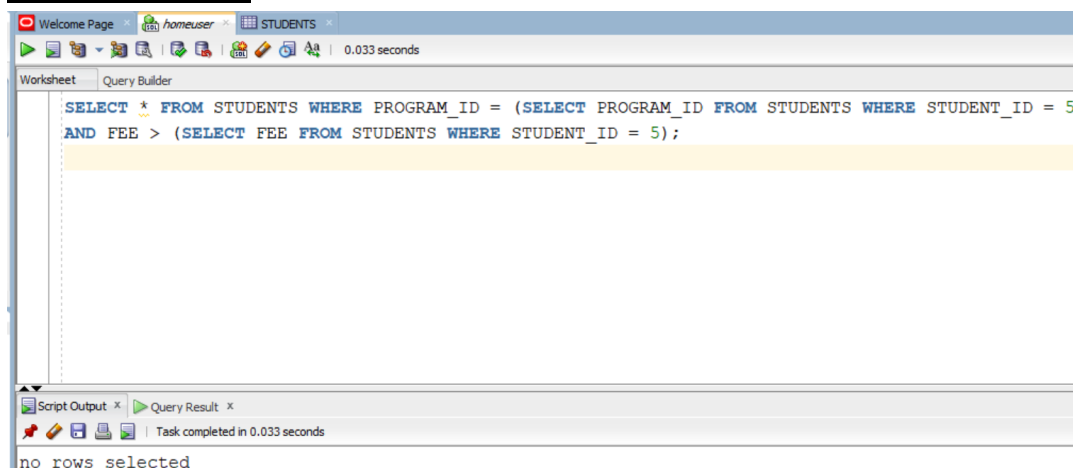
The screenshot shows a database query editor with a 'Query Builder' tab. The query is: `SELECT * FROM STUDENTS WHERE AGE > ALL (SELECT AGE FROM STUDENTS WHERE PROGRAM_ID = 2);`. The result pane at the bottom shows 'no rows selected'.

```
SELECT * FROM STUDENTS
WHERE AGE > ALL (SELECT AGE FROM STUDENTS WHERE PROGRAM_ID = 2);
```

no rows selected

Since no age > 23

#### LAB#04 – TASK#05



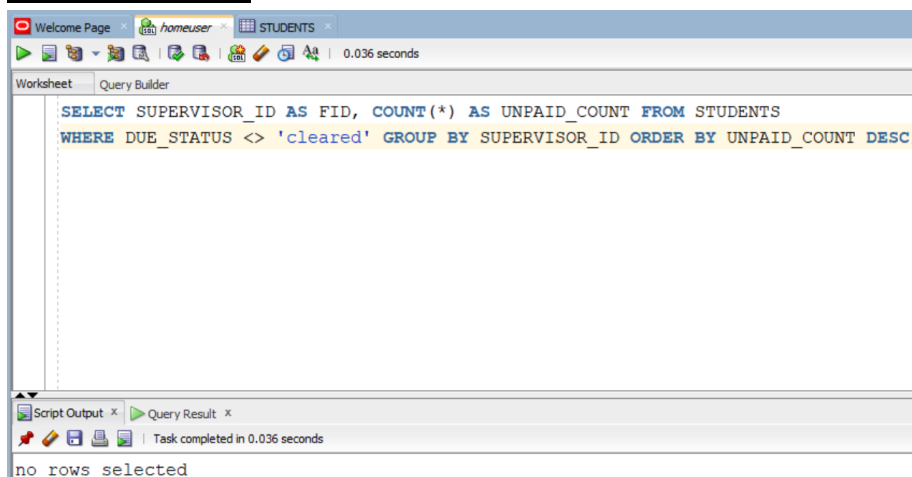
The screenshot shows a database query editor with a 'Query Builder' tab. The query is: `SELECT * FROM STUDENTS WHERE PROGRAM_ID = (SELECT PROGRAM_ID FROM STUDENTS WHERE STUDENT_ID = 5) AND FEE > (SELECT FEE FROM STUDENTS WHERE STUDENT_ID = 5);`. The result pane at the bottom shows 'no rows selected'.

```
SELECT * FROM STUDENTS WHERE PROGRAM_ID = (SELECT PROGRAM_ID FROM STUDENTS WHERE STUDENT_ID = 5)
AND FEE > (SELECT FEE FROM STUDENTS WHERE STUDENT_ID = 5);
```

no rows selected

Since not enough students in table

#### LAB#04 – TASK#06

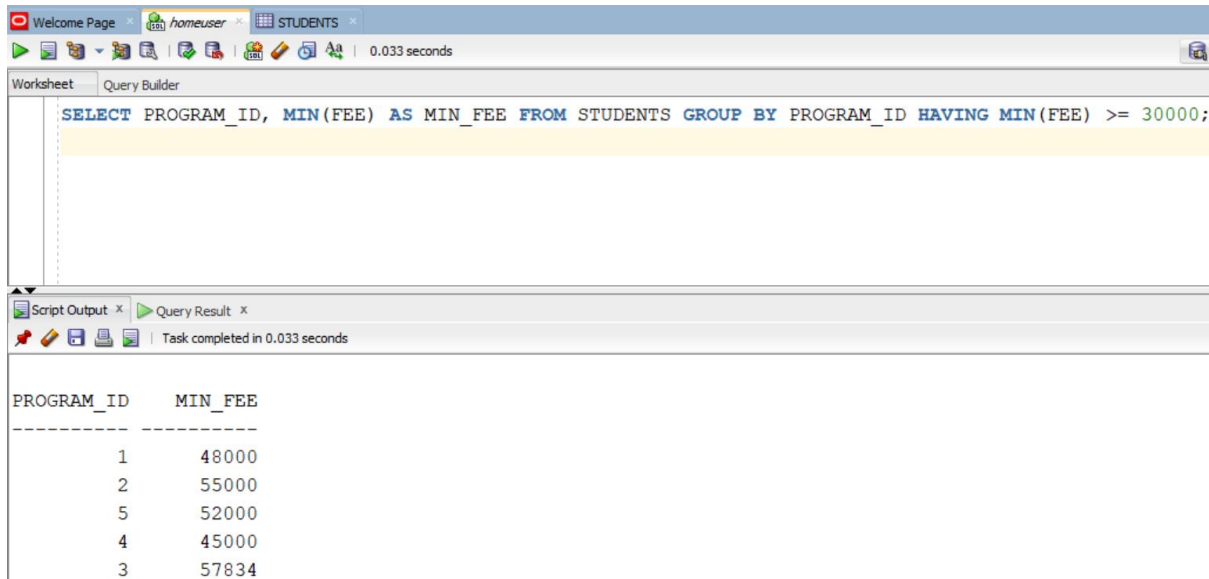


The screenshot shows a database query editor with a 'Query Builder' tab. The query is: `SELECT SUPERVISOR_ID AS FID, COUNT(*) AS UNPAID_COUNT FROM STUDENTS WHERE DUE_STATUS <> 'cleared' GROUP BY SUPERVISOR_ID ORDER BY UNPAID_COUNT DESC;`. The result pane at the bottom shows 'no rows selected'.

```
SELECT SUPERVISOR_ID AS FID, COUNT(*) AS UNPAID_COUNT FROM STUDENTS
WHERE DUE_STATUS <> 'cleared' GROUP BY SUPERVISOR_ID ORDER BY UNPAID_COUNT DESC;
```

no rows selected

### LAB#04 – TASK#07



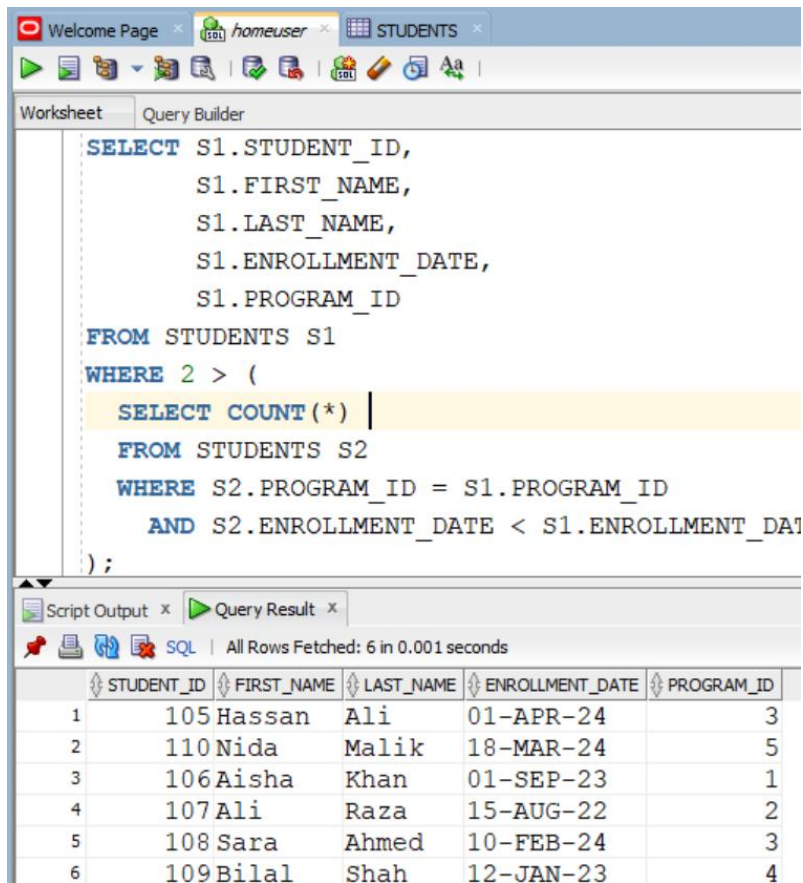
The screenshot shows the SQL Developer interface. The Query Builder pane contains the following SQL query:

```
SELECT PROGRAM_ID, MIN(FEE) AS MIN_FEE FROM STUDENTS GROUP BY PROGRAM_ID HAVING MIN(FEE) >= 30000;
```

The Query Result pane displays the results of the query:

PROGRAM_ID	MIN_FEE
1	48000
2	55000
5	52000
4	45000
3	57834

### LAB#04 – TASK#08



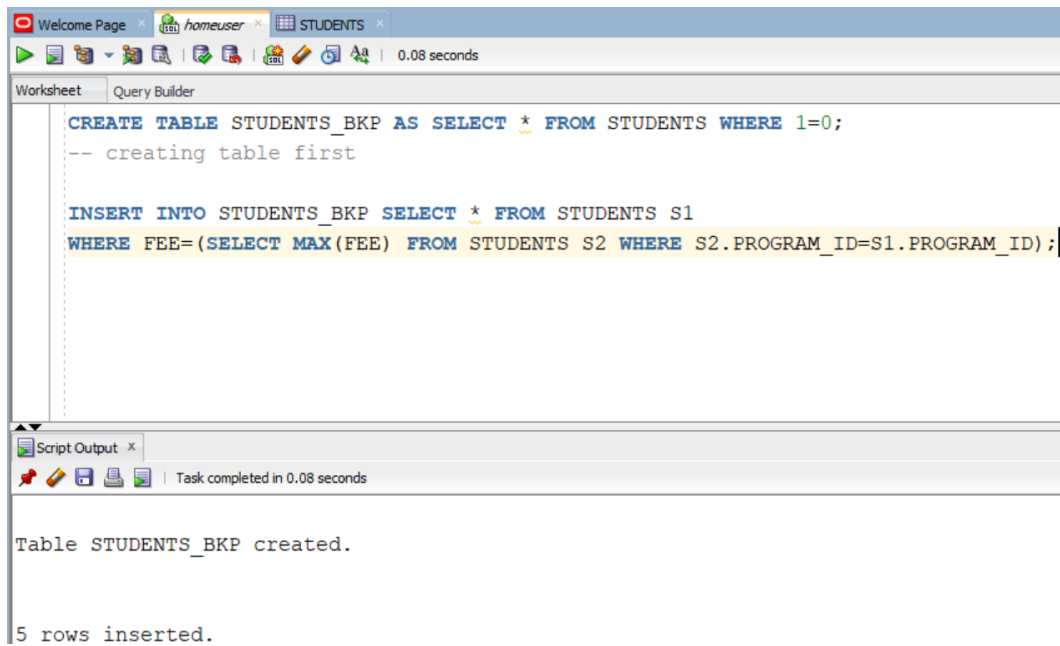
The screenshot shows the SQL Developer interface. The Query Builder pane contains the following SQL query:

```
SELECT S1.STUDENT_ID,  
       S1.FIRST_NAME,  
       S1.LAST_NAME,  
       S1.ENROLLMENT_DATE,  
       S1.PROGRAM_ID  
FROM STUDENTS S1  
WHERE 2 > (  
    SELECT COUNT(*)  
    FROM STUDENTS S2  
    WHERE S2.PROGRAM_ID = S1.PROGRAM_ID  
    AND S2.ENROLLMENT_DATE < S1.ENROLLMENT_DATE  
);
```

The Query Result pane displays the results of the query:

STUDENT_ID	FIRST_NAME	LAST_NAME	ENROLLMENT_DATE	PROGRAM_ID
1	105 Hassan	Ali	01-APR-24	3
2	110 Nida	Malik	18-MAR-24	5
3	106 Aisha	Khan	01-SEP-23	1
4	107 Ali	Raza	15-AUG-22	2
5	108 Sara	Ahmed	10-FEB-24	3
6	109 Bilal	Shah	12-JAN-23	4

### LAB#04 – TASK#09



The screenshot shows a database query editor with a toolbar at the top. The main window is titled 'STUDENTS' and contains the following SQL code:

```
CREATE TABLE STUDENTS_BKP AS SELECT * FROM STUDENTS WHERE 1=0;
-- creating table first

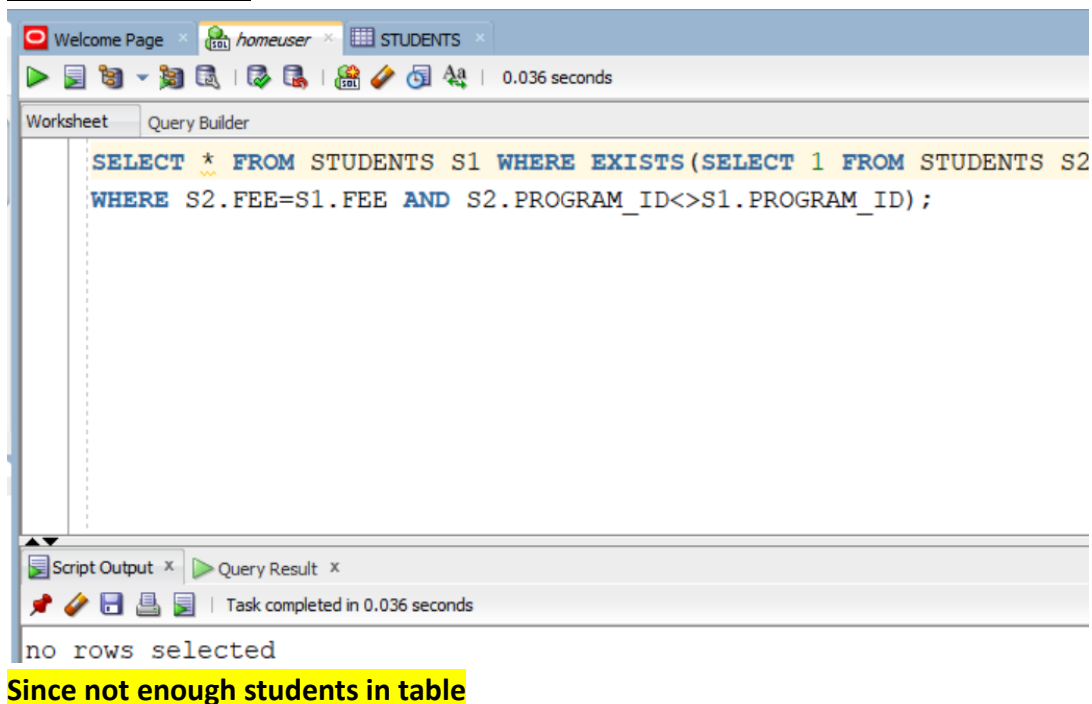
INSERT INTO STUDENTS_BKP SELECT * FROM STUDENTS S1
WHERE FEE=(SELECT MAX(FEE) FROM STUDENTS S2 WHERE S2.PROGRAM_ID=S1.PROGRAM_ID);
```

Below the code editor, the 'Script Output' tab is active, displaying the results of the execution:

```
Table STUDENTS_BKP created.

5 rows inserted.
```

### LAB#04 – TASK#10



The screenshot shows a database query editor with a toolbar at the top. The main window is titled 'STUDENTS' and contains the following SQL code:

```
SELECT * FROM STUDENTS S1 WHERE EXISTS(SELECT 1 FROM STUDENTS S2
WHERE S2.FEE=S1.FEE AND S2.PROGRAM_ID<>S1.PROGRAM_ID);
```

Below the code editor, the 'Script Output' tab is active, displaying the results of the execution:

```
no rows selected
```

Since not enough students in table