My SQL script (committed to Github and executed in Apex):

-- Question 1:

SELECT

MIN(num\_1) AS minimum,

MAX(num\_1) AS maximum

FROM sec1103;

-- Haley Archer

-- Question 2:

SELECT

MIN(num\_1) AS minimum,

MAX(num\_1) AS maximum

FROM sec1103

WHERE row\_ID < 8;

-- Haley Archer

-- Question 3:

SELECT

COUNT(\*) AS total\_rows,

COUNT(num\_1) AS non\_null\_num1,

COUNT(\*) - COUNT(num\_1) AS null\_num1

FROM sec1106;

-- Haley Archer

-- Question 4:

SELECT COUNT(DISTINCT num\_1 || '-' || num\_2) AS distinct\_values

FROM sec1103;

-- Haley Archer

-- Question 5:

SELECT

num\_1,

num\_2,

num\_1 + num\_2 AS row\_wise\_addition,

(SELECT SUM(num\_1) FROM sec1103) + (SELECT SUM(num\_2) FROM sec1103) AS column\_wise\_addition,

COALESCE(num\_1, 0) + COALESCE(num\_2, 0) AS nulls\_treated\_as\_zero

FROM sec1103;

-- Haley Archer

-- Question 6:

SELECT

col\_1,

SUM(col\_3) AS total\_col\_3

FROM sec1202

GROUP BY col\_1;

-- Haley Archer

-- Question 7:

SELECT

col\_1,

col\_2,

SUM(col\_3) AS total\_col\_3

FROM sec1202

GROUP BY col\_1, col\_2;

-- Haley Archer

-- Question 8:

SELECT \* FROM sec1202

UNION ALL

SELECT NULL, NULL, NULL, SUM(col\_3) FROM sec1202;

-- Haley Archer

-- Question 9:

SELECT

col\_1,

SUM(col\_2) AS total\_col\_2

FROM sec1211

GROUP BY col\_1

HAVING SUM(col\_2) > 20;

-- Haley Archer

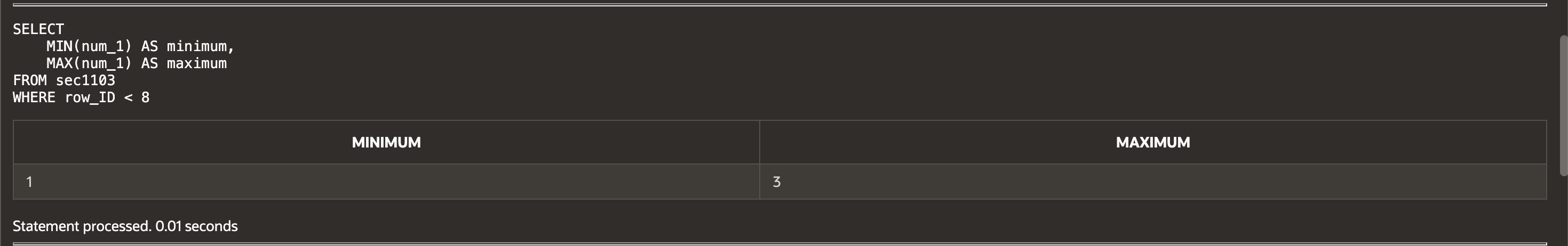
(11-3) Question 1:

Table *sec1103* contains two columns, *row\_ID* and *num\_1*. (It also contains a column named *num\_2*, but we are not going to use that column now.) Find the minimum and maximum values of the *num\_1* column. Name these values “minimum” and “maximum.”



(11-4) Question 2:

Repeat the exercise in the previous section, except this time add a *where* clause that limits the *row\_ID* column to values less than 8. Is there any change in the minimum and maximum values?



(11-6) Question 3:

In table *sec1106*, find the following information:

■ The number of rows in the table

■ The number of rows that have a non-null value in the *Num\_1* column

■ The number of rows that have a null value in the *Num\_1* column  


(11-9) Question 4:

In table *sec1103*, find the number of distinct values in the *num\_1* and *num\_2* columns, taken together.



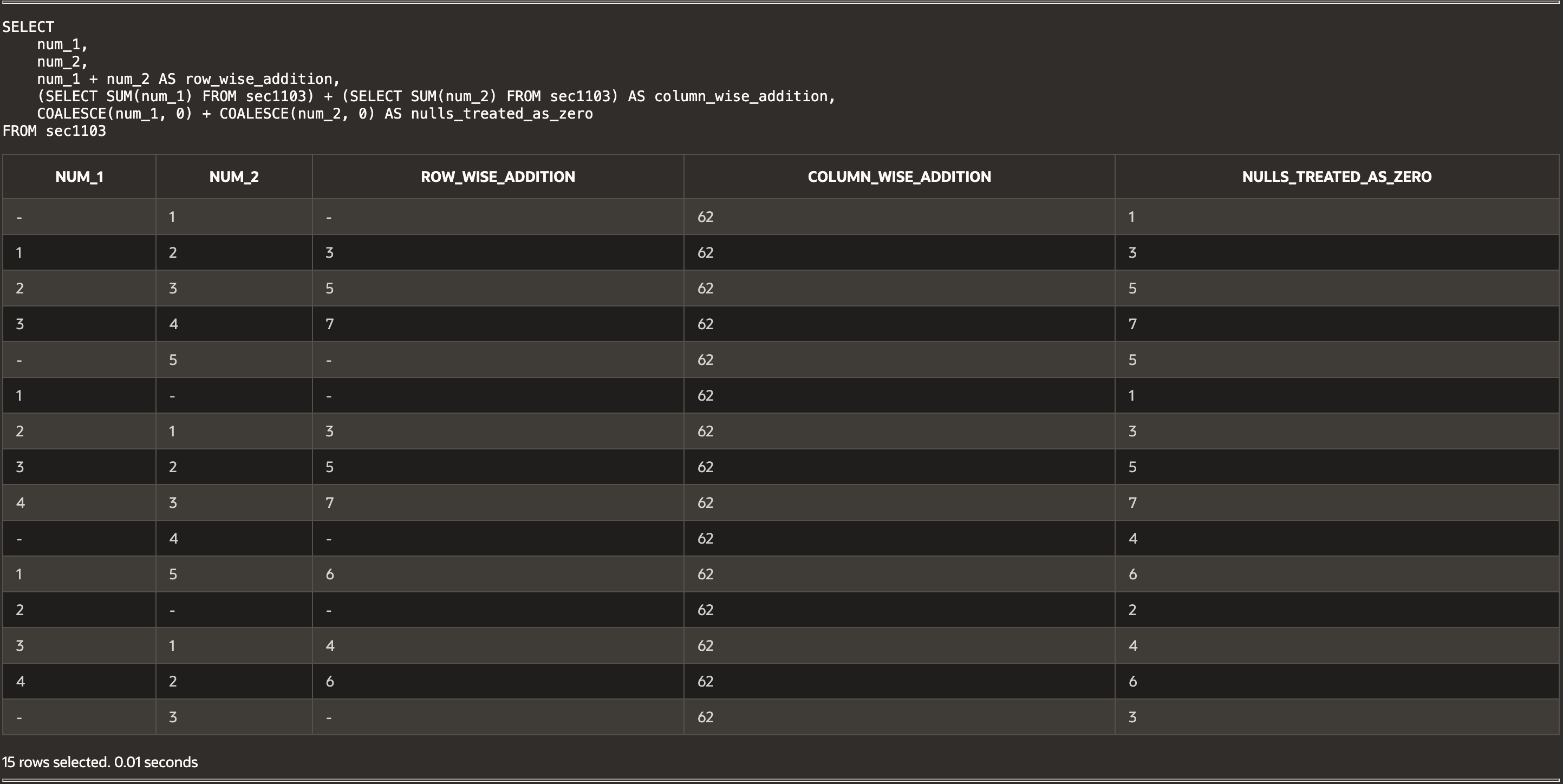
(11-11) Question 5:

In table *sec1103*, show the problem with nulls in addition and how to solve it.

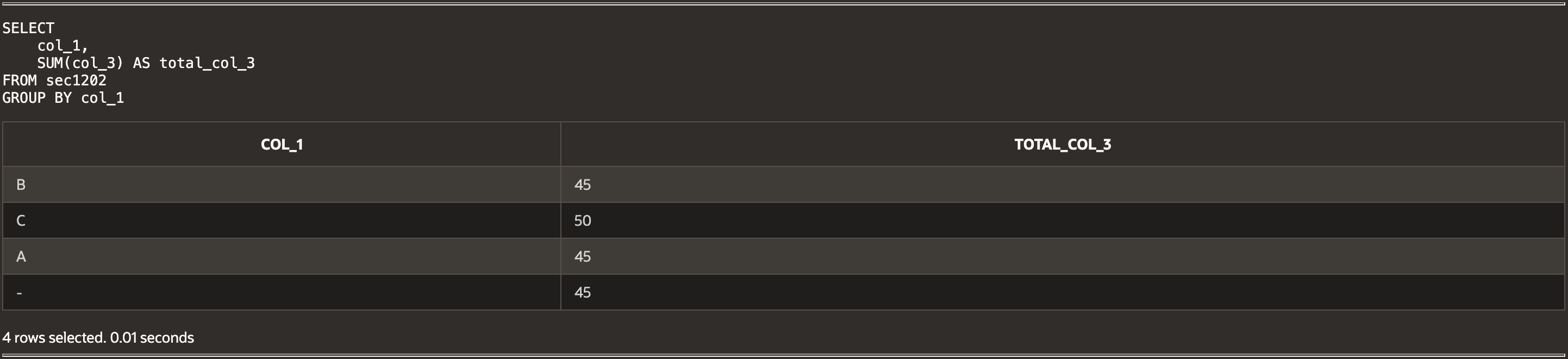
1. Add columns *num\_1* and *num\_2*, adding each row first.

2. Add columns *num\_1* and *num\_2*, adding each column first.

3. Add columns *num\_1* and *num\_2*, changing all the nulls to zeros first.

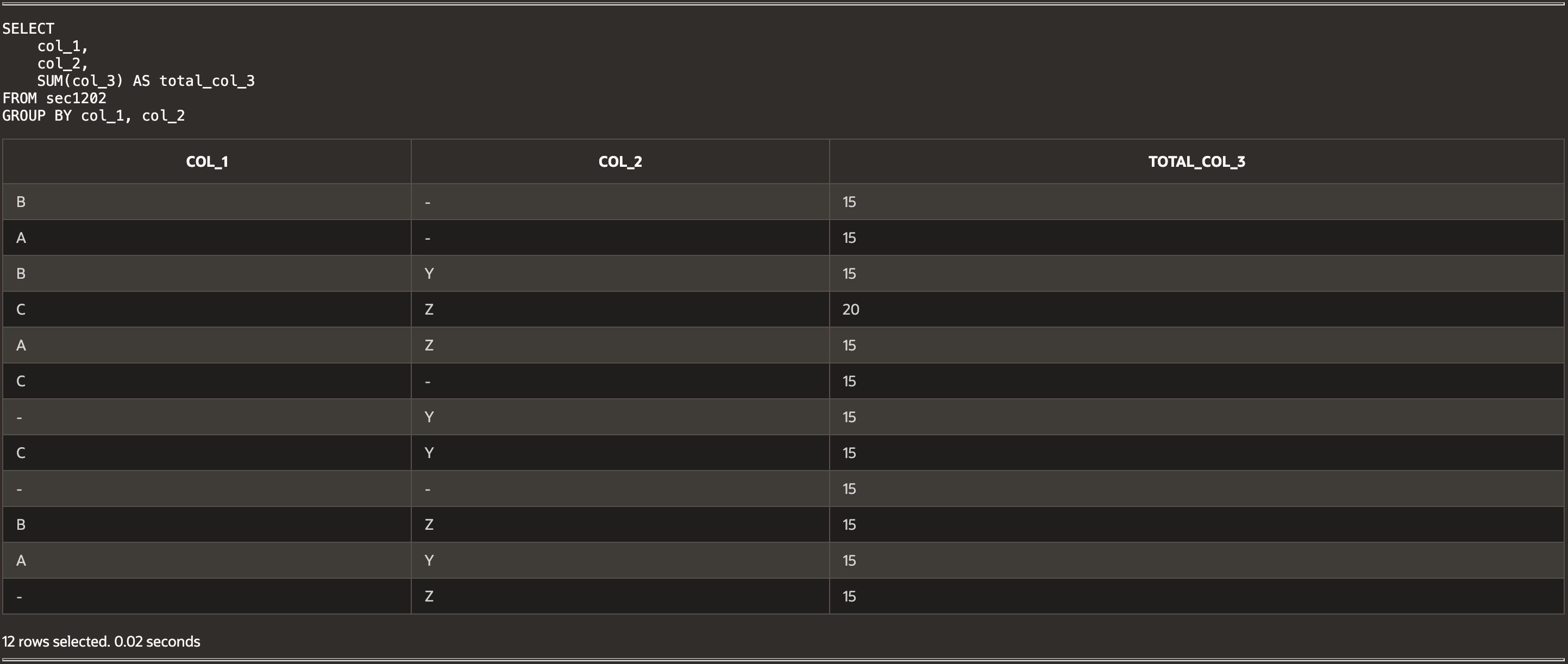


(12-2) Question 6:

Table *sec1202* has four columns: *row\_id, col\_1, col\_2,* and *col\_3*. Write a select statement that groups the rows by the value in *col\_1* and for each group determines the sum of the values in *col\_3*.  
 

(12-4) Question 7:

Use table *sec1202*. Write a select statement that groups the rows by the value in *col\_1* and *col\_2*. For each group determine the sum of the values in *col\_3*.



(12-7) Question 8:

Suppose you wanted to show all the data in table *sec1202* and you also wanted to show the total for *col\_3*. Could you do this with SQL?

The answer is no. The best you can do is to run two queries. One would show the data with:

*select \**

from *sec1202;*

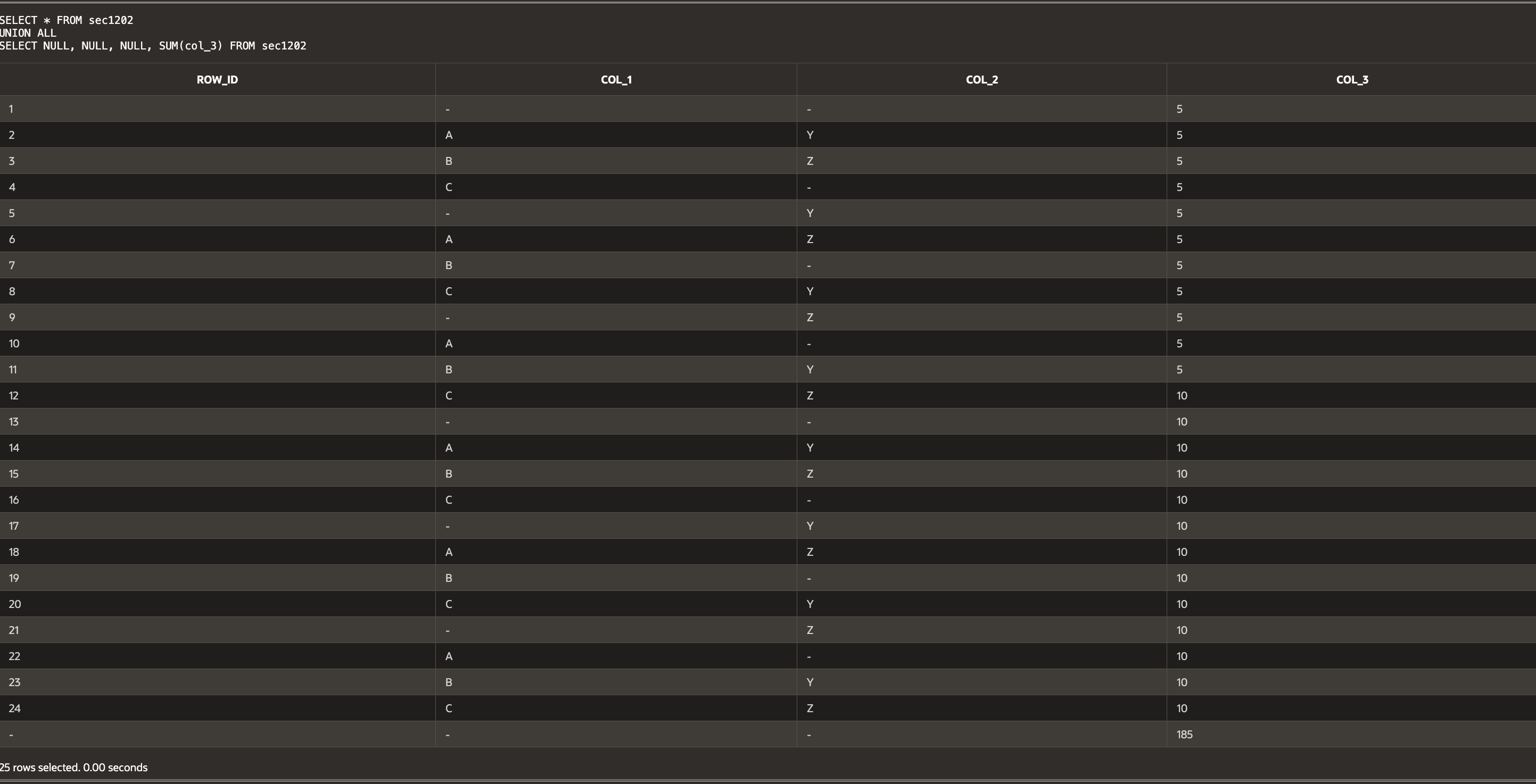
The other would show the total with:

select *sum(col\_3)* as *grand\_total*

from *sec1202;*

Then, if you were desperate, you could paste the two pieces of paper together. Or you could do the same thing in SQL using a union.

Fortunately, most SQL products give you a better way. There is usually some sort of report level to the software that will do totals and subtotals for you.



(11-11) Question 9:

Table *sec1211* has three columns: *row\_id,* *col\_1*, and *col\_2*. Group on *col\_1* and get the sum of *col\_2*. Add a having clause to show only the rows of the result table where the sum is greater than 20.

