SQL – 02/18/2020

1. Query the *Name* of any student in **STUDENTS** who scored higher than  *Marks*. Order your output by the *last three characters* of each name. If two or more students both have names ending in the same last three characters (i.e.: Bobby, Robby, etc.), secondary sort them by ascending *ID*.
2. **Input Format**
3. The **STUDENTS** table is described as follows:A screenshot of a cell phone

   Description automatically generatedThe *Name* column only contains uppercase (A-Z) and lowercase (a-z) letters.
4. **Sample Input**
5. A screenshot of a cell phone

   Description automatically generated
6. **Sample Output**
7. Ashley
8. Julia
9. Belvet
10. **Explanation**
11. Only Ashley, Julia, and Belvet have *Marks* > . If you look at the last three characters of each of their names, there are no duplicates and 'ley' < 'lia' < 'vet'.

**CODE:**

**SELECT NAME FROM STUDENTS WHERE MARKS > 75 ORDER BY RIGHT(NAME,3),ID ASC ;**

**2.)** We define an employee's *total earnings* to be their monthly  worked, and the *maximum total earnings* to be the maximum total earnings for any employee in the **Employee** table. Write a query to find the *maximum total earnings* for all employees as well as the total number of employees who have maximum total earnings. Then print these values as  space-separated integers.

**Input Format**

The **Employee** table containing employee data for a company is described as follows:

A screenshot of a cell phone

Description automatically generated

where *employee\_id* is an employee's ID number, *name* is their name, *months* is the total number of months they've been working for the company, and *salary* is the their monthly salary.

**Sample Input**

A screenshot of a cell phone

Description automatically generated

**Sample Output**

69952 1

**Explanation**

The table and earnings data is depicted in the following diagram:A screenshot of a cell phone

Description automatically generated

The maximum *earnings* value is . The only employee with *earnings*  is *Kimberly*, so we print the maximum *earnings* value () and a count of the number of employees who have earned  (which is ) as two space-separated values.

**CODE:**

**SELECT (SALARY \* MONTHS) AS EARNINGS, COUNT(\*) FROM EMPLOYEE GROUP BY 1 ORDER BY EARNINGS DESC LIMIT 1;**

**3)** Query the following two values from the **STATION** table:

1. The sum of all values in *LAT\_N* rounded to a scale of  decimal places.
2. The sum of all values in *LONG\_W* rounded to a scale of  decimal places.

**Input Format**

The **STATION** table is described as follows:

A close up of a street

Description automatically generated

where *LAT\_N* is the northern latitude and *LONG\_W* is the western longitude.

**Output Format**

Your results must be in the form:

lat lon

where  is the sum of all values in *LAT\_N* and  is the sum of all values in *LONG\_W*. Both results must be rounded to a scale of  decimal places.

**CODE: SELECT ROUND(SUM(LAT\_N),2) ,ROUND(SUM(LONG\_W),2) FROM STATION ;**

**4)** Write a query identifying the *type* of each record in the **TRIANGLES** table using its three side lengths. Output one of the following statements for each record in the table:

* **Equilateral**: It's a triangle with  sides of equal length.
* **Isosceles**: It's a triangle with  sides of equal length.
* **Scalene**: It's a triangle with  sides of differing lengths.
* **Not A Triangle**: The given values of *A*, *B*, and *C* don't form a triangle.

**Input Format**

The **TRIANGLES** table is described as follows:

A screenshot of a cell phone

Description automatically generated

Each row in the table denotes the lengths of each of a triangle's three sides.

**Sample Input**

A drawing of a face

Description automatically generated

**Sample Output**

Isosceles

Equilateral

Scalene

Not A Triangle

**Explanation**

Values in the tuple  form an Isosceles triangle, because .  
Values in the tuple  form an Equilateral triangle, because . Values in the tuple  form a Scalene triangle, because .  
Values in the tuple  cannot form a triangle because the combined value of sides  and  is not larger than that of side .

**CODE:**

**select case**

**when A+B <= C or A+C <= B or B+C <= A then "Not A Triangle"**

**when A=B and B=C then "Equilateral"**

**when A=B or A=C or B=C then "Isosceles"**

**else "Scalene"**

**end as triangles\_type**

**from TRIANGLES;**