LEARNINGS FROM SQL:

1. USING SUBSTRING COMMAND TO SORT NAMES BY LAST THREE LETTERS:

* SO, FOR THIS WE USE “SUBSTR (VARIABLE, VARIABLE\_SIZE)”.

SO, WE SORT EMPLOYEES BY LAST THREE LETTERS:

SELECT NAME FROM

POPULATION

ORDER BY SUBSTR (NAME, -3);

WE USE -3 AS WE NEED TO SORT DATA BY LAST THREE LETTERS:

IN RECENT SQL,

WE CAN USE “RIGHT(NAME,3)”.

1. USING “ROUND ()” IN MYSQL IS USED FOR ROUNDING OFF A NUMBER TO NEAREST INTEGER.

* SO, WE NEED TO ROUND OFF GIVEN DATA WE SHOW TO THE NEAREST INTEGER.

THE SQL COMMAND IS:

SELECT ROUND(AVG(POPULATION))

FROM CITY;

IT GAVE ROUND-OFF VALUE, WHICH IS 454250.

DIFFERENT QUERY: SELECT ROUND(SUM(LAT\_N),2),

ROUND(SUM(LONG\_W),2)

FROM STATION;

1. USING CASES IN SQL.

SYNTAX:

CASE  
    WHEN *condition1* THEN *result1*  
    WHEN *condition2* THEN *result2*  
    WHEN *condition3* THEN *resul3t*  
    ELSE *result*  
END;

REMEMBER: THE ELSE CONDITION IS NOT ALWAYS NECESSARY.

QUESTION.) 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.

=> SO HERE WE USE CASES TO GET OUTPUT:

THE CODE IS:

SELECT

CASE

WHEN A+B<=C OR B+C<=A OR A+C<=B THEN 'Not A Triangle'

WHEN A = B AND B = C THEN 'Equilateral'

WHEN A = B AND A <> C THEN 'Isosceles'

WHEN B = C AND B <> A THEN 'Isosceles'

WHEN C = A AND C <> B THEN 'Isosceles'

WHEN A <> B AND B <> C AND A <> C THEN 'Scalene'

END

FROM TRIANGLES;

1. DIFFERENCE BETWEEN TWO OUTPUTS:

(IMP: THIS IS NOT SET DIFFERENCE.)

QUESTION.) Query the difference between the maximum and minimum populations in **CITY**.

=> SELECT MAX(POPULATION)-MIN(POPULATION) AS DIFF

FROM CITY;

1. QUESTION: We define an employee's *total earnings* to be their monthly

(SALARY\* MONTHS) 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 2 space-separated integers.

=> SELECT MAX (MONTHS \* SALARY), COUNT(\*) FROM Employee WHERE MONTHS\* SALARY =(SELECT MAX(MONTHs \* SALARY) FROM EMPLOYEE) ;

SO, HERE WE USE SUB-QUERY , FIRST WE FOUND MAX ANNUAL SALARY AND COUNT USING THE COMMAND ON 2ND LINE.

1. Query the sum of Northern Latitudes (LAT\_N) from STATION having values greater than 38.7880 and less than 137.2345 . Truncate your answer to 4 decimal places.

=> SELECT ROUND(SUM(LAT\_N),4)

FROM STATION

WHERE LAT\_N>38.7880 AND LAT\_N<137.2345;

1. THIS QUESTION IS AN EXAMPLE OF A SUB-QUERY AND YOU SHOULD NOT FORGET IT FUTURE.

=> Query the *Western Longitude* (*LONG\_W*) for the largest *Northern Latitude* (*LAT\_N*) in **STATION** that is less than 137.2345 . Round your answer to 4 decimal places.

select round(long\_w,4) from station where lat\_n=(select max(lat\_n) from station where lat\_n<137.2345);

FOR A VALUE OF LAT\_N WE GIVE OUTPUT FOR LONG\_W.

ANOTHER WAY OF ROUNDING OFF YOUR OUTPUT IN SQL IS USING FLOOR FUNCTION.

EXAMPLE:

select country.continent,floor(avg(city.population))

from city

inner join country on city.countrycode=country.code

group by country.continent;

1. Given the CITY and COUNTRY tables, query the sum of the populations of all cities where the CONTINENT is 'Asia'.

Note: CITY.CountryCode and COUNTRY.Code are matching key columns.

=>

select sum(city.population)

from city

inner join country on city.countrycode=country.code

where country.continent='Asia';

1. In this question we learnt to calculate the Manhattan distance when coordinates of two points are given.

Consider  P1(a,b) and P2 (c,d) to be two points on a *2D* plane.

* a happens to equal the minimum value in *Northern Latitude* (*LAT\_N* in **STATION**).
* b happens to equal the minimum value in *Western Longitude* (*LONG\_W* in **STATION**).
* c happens to equal the maximum value in *Northern Latitude* (*LAT\_N* in **STATION**).
* d happens to equal the maximum value in *Western Longitude* (*LONG\_W* in **STATION**).

Query the [Manhattan Distance](https://xlinux.nist.gov/dads/HTML/manhattanDistance.html) between points  and  and round it to a scale of 4 decimal places.

The formula for Manhattan distance is | a - c| + | b - d| where a and b are min lat and long and c and d are max lat and long respectively.

select

round(

abs(

min(lat\_n)- max(lat\_n)

) + abs(

min(long\_w)- max(long\_w)

), 4

)

from

station;

MY CODE:

select round(abs(min(lat\_n)-max(lat\_n))+abs(min(long\_w)-max(long\_w)),4)

from station;

THIS IS HOW WE CALCULATE MANHATTAN DISTANCE.

Your Output (stdout)

* **259.6859**

**Q) STATEMENT: THIS QUESTION WAS SIMILAR TO ABOVE ONE AND WE HAD TO CALCULATE THE EUCLIDEAN DISTANCE INSTEAD OF MANHATTAN DISTANCE.**

**=> SELECT ROUND(POWER(POWER(MIN(lat\_n) - MAX(lat\_n), POWER(MIN(long\_w) - MAX(long\_w), 2), 0.5), 4) FROM station;**

**LEETCODE QUESTION**

**{EASY LEVEL}**

**=>**

**Write an SQL query to get the names of products that have at least 100 units ordered in February 2020 and their amount.**

**Return result table in any order.**

**=> SELECT products.product\_name, sum(orders.unit) as unit**

**FROM products**

**INNER JOIN orders ON products.product\_id = orders.product\_id**

**WHERE orders.order\_date BETWEEN '2020-02-01' AND '2020-02-29'**

**GROUP BY products.product\_name**

**HAVING sum(orders.unit) >= 100;**

**Q) P(R) represents a pattern drawn by Julia in R rows. The following pattern represents P(5):**

**\***

**\* \***

**\* \* \***

**\* \* \* \***

**\* \* \* \* \***

**Write a query to print the pattern P(20).**

**=> THIS IS ONE OF THE EASIEST STILL THE MOST CONFUSING QUERY I HAVE COME ACROSS. STILL WAS ABLE TO SOLVE IT WITH SOME LEARNINGS.**

**FIRST, I WAS PRINTING PATTERNS WITH USER INPUT , BUT THE QUESTION ONLY ASKED FOR THE FIRST 20 ROWS.**

**SO, I DID ACCORDINGLY.**

**set serveroutput on;**

**Most important line highlighted in red without this we won’t get the output on the console so remember.**

**DECLARE**

**i INTEGER;**

**j INTEGER;**

**BEGIN**

**-- Loop to print the triangle pattern**

**FOR i IN 1..20 LOOP**

**FOR j IN 1..i LOOP**

**DBMS\_OUTPUT.PUT('\* ');**

**END LOOP;**

**DBMS\_OUTPUT.PUT\_LINE('');**

**END LOOP;**

**END;**

**/**

**Q) P(R) represents a pattern drawn by Julia in R rows. The following pattern represents P(5):**

**\* \* \* \* \***

**\* \* \* \***

**\* \* \***

**\* \***

**\***

**Write a query to print the pattern P(20).**

**=> Similar QUESTION JUST REVERSE THE PATTERN.**

**Remember:**

**THE KEYWORD WHICH WE USED FOR PRINTING ANYTHING**

**DBMS\_OUTPUT.PUT\_LINE(‘ ’);**

**THE (“PUT\_LINE”) KEYWORD AUTOMATICALLY TRANSFERS CONTROL TO THE NEXT LINE .SO WE SHOULD NOT USE IT INSIDE THE LOOP.**

**TO PRINT AND STAY ON THE SAME LINE:**

**USE:**

**DBMS\_OUTPUT.PUT(‘ ’);**

**=>**

**SQL> DECLARE**

**2 i INTEGER;**

**3 j INTEGER;**

**4 BEGIN**

**5 FOR i IN REVERSE 1..20 LOOP**

**6 FOR j IN 1..i LOOP**

**7 DBMS\_OUTPUT.PUT('\*');**

**8 END LOOP;**

**9 DBMS\_OUTPUT.PUT\_LINE('');**

**10 END LOOP;**

**11 END;**

**12 /**

**Q) Samantha was tasked with calculating the average monthly salaries for all employees in the EMPLOYEES table but did not realize her keyboard's 0  key was broken until after completing the calculation. She wants your help finding the difference between her miscalculation (using salaries with any zeros removed), and the actual average salary.**

**Write a query calculating the amount i.e. (actual-miscalculated) of errors (i.e.:  average monthly salaries), and round it up to the next integer.**

**=> Here we have to knowingly do what Samantha did which is replace every occurrence of 0 with blank space(‘ ’)**

**We can do this using the replace function in SQL.**

**The command to do that would be:**

**Select replace(salary,0,’’) from employees;**

**This does the job, but we have to calculate the average of actual and miscalculated and give the difference. And round to nearest integer.**

**THE SQL QUERY FOR THAT IS:**

**=>**

**sql> select ceil((avg(salary)-avg(replace(salary,0,'')))) from employees;**