

Tarea 9 – Funciones y Procedimientos

Ejercicio 1

Declaración de un procedimiento para saber la cantidad de elementos que tiene la columna de una tabla.

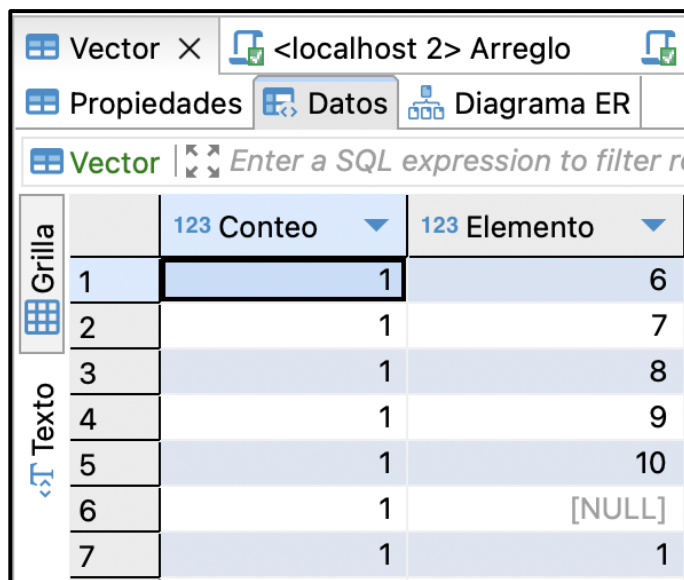
```
CREATE TABLE Vector(  
    Conteo INT,  
    Elemento INT  
);
```

```
INSERT INTO Vector(Conteo, Elemento)  
VALUES (1, 6),  
      (1, 7),  
      (1, 8),  
      (1, 9),  
      (1, 10),  
      (1, NULL),  
      (1, 1);
```

Explicación: declaramos una tabla llamada “Vector”, que contendrá 2 columnas:

- Conteo (Columna referencia).
- Elemento (Columna con datos tanto numéricos como nulos).

Insertamos solamente 1's en la columna “Conteo” y cualquier dato numérico o nulo en la columna “Elemento”.



The screenshot shows a database application window with a tab labeled 'Vector'. Below the tab are three buttons: 'Propiedades', 'Datos', and 'Diagrama ER'. The 'Datos' button is selected, and a search bar contains the text 'Enter a SQL expression to filter re'. Below the search bar is a table with two columns: '123 Conteo' and '123 Elemento'. The table has 7 rows of data. The first row is highlighted with a blue background. The first column contains the value '1' for all rows, and the second column contains the values '6', '7', '8', '9', '10', '[NULL]', and '1' respectively.

	123 Conteo	123 Elemento
1	1	6
2	1	7
3	1	8
4	1	9
5	1	10
6	1	[NULL]
7	1	1

```
DROP PROCEDURE IF EXISTS Contar_elementos;
```

```

DELIMITER //
CREATE PROCEDURE Contar_elementos(
    OUT Contador INT
)
BEGIN
    DECLARE i INT DEFAULT 0;
    DECLARE Maximo INT DEFAULT (SELECT SUM(Conteo) FROM Vector);

    Contar: LOOP
        SET i = i + 1;

        IF i > Maximo THEN
            LEAVE Contar;
        END IF;

        SET Contador = i;
    END LOOP Contar;
END //
DELIMITER ;

CALL Contar_elementos(@Total_de_elementos);
SELECT @Total_de_elementos AS Total_de_elementos;

```

Explicación:

1. Eliminamos algún procedimiento en caso de que exista.
2. Creamos el procedimiento bajo el nombre de "Contar_elementos", con una variable entera de salida ("Contador").
3. Declaramos la primera variable acumuladora "i" con un 0 de valor default.
4. Declaramos la variable "Maximo" con un valor default igual a la suma de todos los 1's que hay en la columna "Conteo" de la tabla "Vector".
5. Declaramos un ciclo llamado: "Contar", que va a sumarle 1 a la variable "i", y luego la va a asignar a la segunda variable acumuladora "Contador".
 - a. Este ciclo se va a ejecutar hasta que la suma de todas las "i's" supere el valor de la variable "Maximo".
6. Una vez que "i" supera a "Maximo", se rompe el ciclo.
7. Mandamos llamar a la función con el nombre: @Total_de_elementos.
8. Imprimimos mediante un SELECT.

123 Total_de_elementos	▼
	7

Resultado impreso con éxito, si hay 7 elementos en cada una de las columnas de la tabla.

Ejercicio 2

Declaración de una función para calcular la correlación entre 2 conjuntos de datos.

```
CREATE TABLE Valores(  
  Id INT PRIMARY KEY AUTO_INCREMENT,  
  V1 DOUBLE,  
  V2 DOUBLE  
);
```

```
INSERT INTO Valores(V1, V2)  
VALUES (6, 45),  
       (12, 47),  
       (13, 39),  
       (17, 58),  
       (22, 68),  
       (25, 76),  
       (27, 75),  
       (29, 74),  
       (30, 78),  
       (32, 81);
```

Explicación: declaramos una tabla llamada “Valores”, que contendrá las 2 columnas a utilizar; les insertamos valores numéricos.

```
CREATE TABLE Diferencias(  
  Id INT PRIMARY KEY AUTO_INCREMENT,  
  Dif_media_1 DOUBLE,  
  Dif_media_2 DOUBLE  
);
```

```
INSERT INTO Diferencias(Dif media 1, Dif media 2)  
  SELECT (V1 - AVG(V1) OVER()),  
         (V2 - AVG(V2) OVER())  
  FROM Valores;
```

Explicación: declaramos una tabla llamada “Diferencias”, que contendrá la resta de la columna Vi con su respectivo promedio; les insertamos sus respectivos valores.

```
CREATE TABLE Productos(  
  Id INT PRIMARY KEY AUTO_INCREMENT,
```

```
Prod_dif_media DOUBLE  
);
```

```
INSERT INTO Productos(Prod dif media) SELECT Dif_media_1 * Dif_media_2  
FROM Diferencias;
```

Explicación: declaramos una tabla llamada “Productos”, que contendrá el producto de la diferencia de medias de la columna “V1” y la columna “V2”; les insertamos sus respectivos valores.

```
CREATE TABLE Cuadrado1(  
  Id INT PRIMARY KEY AUTO_INCREMENT,  
  Dif_media_cua1 DOUBLE  
);
```

```
INSERT INTO Cuadrado1(Dif media cua1) SELECT Dif_media_1 * Dif_media_1  
FROM Diferencias;
```

Explicación: declaramos una tabla llamada “Cuadrado1” que contendrá el producto de la diferencia de medias de la columna “V1”; le insertamos sus respectivos valores.

```
CREATE TABLE Cuadrado2(  
  Id INT PRIMARY KEY AUTO_INCREMENT,  
  Dif_media_cua2 DOUBLE  
);
```

```
INSERT INTO Cuadrado2(Dif media cua2) SELECT Dif_media_2 * Dif_media_2  
FROM Diferencias;
```

Explicación: declaramos una tabla llamada “Cuadrado2” que contendrá el producto de la diferencia de medias de la columna “V2”; le insertamos sus respectivos valores.

```
CREATE TABLE Datos AS  
SELECT V1, V2, Dif_media_1, Dif_media_2, Prod_dif_media, Dif_media_cua1,  
Dif_media_cua2  
FROM Valores
```

INNER JOIN Diferencias **ON** Diferencias.Id = Valores.Id

INNER JOIN Productos **ON** Productos.Id = Valores.Id

INNER JOIN Cuadrado1 **ON** Cuadrado1.Id = Valores.Id

INNER JOIN Cuadrado2 **ON** Cuadrado2.Id = Valores.Id;

Explicación: declaramos una tabla llamada “Datos” que contendrá todas las columnas de todas las tablas anteriormente creadas. Realizamos el anexo de las columnas mediante los JOINS de sus respectivos Id.

	123 V1	123 V2	123 Dif_media_1	123 Dif_media_2	123 Prod_dif_media	123 Dif_media_cua1	123 Dif_media_cua2
1	6	45	-15.3	-19.1	292.23	234.09	364.81
2	12	47	-9.3	-17.1	159.03	86.49	292.41
3	13	39	-8.3	-25.1	208.33	68.89	630.01
4	17	58	-4.3	-6.1	26.23	18.49	37.21
5	22	68	0.7	3.9	2.73	0.49	15.21
6	25	76	3.7	11.9	44.03	13.69	141.61
7	27	75	5.7	10.9	62.13	32.49	118.81
8	29	74	7.7	9.9	76.23	59.29	98.01
9	30	78	8.7	13.9	120.93	75.69	193.21
10	32	81	10.7	16.9	180.83	114.49	285.61

DROP FUNCTION IF EXISTS Correlacion;

DELIMITER \$\$

CREATE FUNCTION Correlacion()

RETURNS DOUBLE DETERMINISTIC

BEGIN

DECLARE SUMA1 **DOUBLE DEFAULT** (**SELECT SUM**(Prod_dif_media)
FROM Productos);

DECLARE SUMA2 **DOUBLE DEFAULT** (**SELECT SUM**(Dif_media_cua1)
FROM Cuadrado1);

DECLARE SUMA3 **DOUBLE DEFAULT** (**SELECT SUM**(Dif_media_cua2)
FROM Cuadrado2);

DECLARE r **DOUBLE**;

IF SUMA2 = 0 **OR** SUMA3 = 0 **THEN**

SET r = 0;

ELSE

SET r = SUMA1 / (**SELECT SQRT**(SUMA2 * SUMA3));

END IF;

RETURN r;

END \$\$


DELIMITER ;

SELECT Correlacion();

Explicación:

1. Eliminamos alguna función en caso de que exista.
2. Creamos la función bajo el nombre de “Correlación”.

3. Declaramos que la variable de retorno sea de origen DOUBLE (decimal) y DETERMINISTIC (su valor de retorno no cambia si se le asignan los mismos valores).
4. Declaramos la variable "SUMA1" para que tome por default la suma del producto de la diferencia de medias de la columna "V1" y la columna "V2".
5. Declaramos la variable "SUMA2" para que tome por default la suma del producto de la diferencia de medias de la columna "V1".
6. Declaramos la variable "SUMA3" para que tome por default la suma del producto de la diferencia de medias de la columna "V2".
7. Declaramos "r" como una variable de origen DOUBLE.
 - a. Si el resultado de SUMA2 = 0 y SUMA3 = 0, le asignamos a r = 0, ya que para calcular la correlación se necesita dividir entre las anteriores variables. Si el resultado de alguna de ellas es 0, la división no es posible.
 - b. Si SUMA2 <> 0 y SUMA3 <> 0, hacemos que: $r = \text{SUMA1} / (\sqrt{\text{SUMA2} * \text{SUMA3}})$;
8. Regresamos el valor de "r".
9. Imprimimos mediante un SELECT.

	123 Correlacion() ▼
1	0.9472192453

Resultado impreso con éxito. Es el mismo resultado que nos proporciona la pagina web: <https://www.statology.org/correlation-coefficient-by-hand/>.