Profesor
M. en I.A. Erick Orlando Matla Cruz
Ayudantes
L. en C.C. Efraín Hipólito Chamú
L. en C.C. Anahí Quiroz Jiménez
L. en C.C. Karen Monserrat Zavala Correa

Práctica 09

CROSSTAB y Funciones de Ventana

1. Objetivo General

Comprender la estructura y aplicación de la función *CROSSTAB* y de las funciones de ventana para la manipulación de datos en PostgreSQL.

2. Objetivos Secundarios

- Entender cuándo y cómo pueden ser útiles las funciones de ventana.
- Entender la diferencia entre agrupaciones y funciones de ventana.
- Entender cómo utilizar la función CROSSTAB y comprender su utilidad.

3. Introducción

Cuando uno manipula bases de datos, es importante conocer herramientas que nos permiten mejorar la presentación de los resultados de nuestras consultas, esto es, obtener mejores estructuras (formatos) de salida de los datos recuperados; sin embargo, las herramientas mencionadas, no necesariamente son ajenas al SMBD ni requieren el uso o entendimiento de grandes conceptos del lenguaje SQL.

3.1 CROSSTAB

Esta función permite "trasponer" el contenido de una tabla (que puede ser el resultado de una consulta), en una tabla formada por columnas con valores únicos de tuplas. CROSSTAB opera con uno o dos parámetros:

- Parámetro 1: es la tabla que se desea trasponer o "invertir" y debe estar formada por 3 atributos que tomarán cierta "acción según su posición" y el contenido deberá estar ordenado por el primer y segundo atributo.

Pivote	Columna	valor

1. Pivote: cada que cambie de valor, esto es, cada vez que se encuentre un valor nuevo, se creará una nueva tupla en la tabla resultado (tabla transpuesta). Es el único atributo que se mantiene en su posición en la tabla resultado.
2. Columna: cada que cambia de valor, se crea una nueva columna en la tabla resultado, sin embargo, el "nombre del atributo" de salida no será el de la etiqueta generadora de la columna, excepto, si el nombre se asigna de manera explícita en la tabla resultado.

3. Valor: es el valor que se asigna a la posición pivote(n), columna(m).

- Parámetro 2: indica los posibles valores únicos que pueden tomar el atributo “columna” de la tabla parámetro 1.

A continuación, se describe su uso.

```
SELECT *
FROM CROSSTAB('SELECT pivote,columna,valor
               FROM tabla
               ORDER BY 1,2')
AS t_resultado (opivote TIPO_DATO, ocolumna1 TIPO_DATO, …, ocolumnan
TIPO_DATO);
```

Figura A.1 Sintaxis de CROSSTAB con 1 parámetro

```
SELECT *
FROM CROSSTAB('SELECT pivote,columna,valor
               FROM tabla
               ORDER BY 1,2',
              'SELECT DISTINCT(atributo)
               FROM tabla
               ORDER BY 1')
AS t_resultado (opivote TIPO_DATO,ocolumna1 TIPO_DATO,…,ocolumnan
TIPO_DATO);
```

Figura A.2 Sintaxis de CROSSTAB con 2 parámetros

Tanto en A.1 como en A.2 encontramos:

1. **SELECT ***

Palabra reservada para indicar que deseamos obtener todos (*) los atributos del resultado de la ejecución de CROSSTAB.

2. **FROM CROSSTAB**

   Palabra reservada y llamada a la función CROSSTAB para indicarle al SMBD que deseamos ejecutar dicha función.

3. **AS**

   Palabra reservada que permite crear la estructura de la tabla resultado que se obtendrá al aplicar CROSSTAB a la tabla pasada como parámetro 1.

4. **t_resultado**

   Nombre asignado a la tabla resultado que contendrá los datos traspuestos.

5. **'Consulta'**

   Estructura de consulta que genera una tabla de 3 atributos, ordenada por los primeros 2.

6. **;**

   Indica la finalización de la instrucción.

Nota: Se debe añadir la extensión 'tablefunc' para poder hacer uso de la función.

CREATE EXTENSION tablefunc;

Ejemplos:

Teniendo como ejemplo los siguientes datos se describen algunos ejemplos para su uso:

Profesor
M. en I.A. Erick Orlando Matla Cruz
Ayudantes
L. en C.C. Efraín Hipólito Chamú
L. en C.C. Anahí Quiroz Jiménez
L. en C.C. Karen Monserrat Zavala Correa

id_product	tipo	name_product	category	total_products
1	A	Mobile	IT	20
3	A	Desktop	IT	50
5	A	Laptop	IT	30
8	A	Laptop	FG	30
2	B	Mobile	ELE	40
4	B	Desktop	ELE	60
6	B	Laptop	ELE	70
7	B	Desktop	FG	60
9	B	Laptop	IT	70

Tabla 1 Productos.

```
SELECT *
FROM CROSSTAB
(
   'SELECT name_product, category, total_products
         FROM productos
         ORDER BY 1,2'
)AS T (NAME_PRODUCT TEXT,  ELE INT, FG INT, IT INT);
```

Ejemplo 1. CROSSTAB con un parámetro

Profesor
M. en I.A. Erick Orlando Matla Cruz
Ayudantes
L. en C.C. Efraín Hipólito Chamú
L. en C.C. Anahí Quiroz Jiménez
L. en C.C. Karen Monserrat Zavala Correa

name_product	ele	fg	it
Desktop	60	60	50
Laptop	70	30	70
Mobile	40	20	

Resultado ejemplo 1.

```
SELECT *
FROM CROSSTAB
(
  'SELECT tipo,category, total_products
      FROM productos WHERE category="ELE" OR category="FG"
      ORDER BY 1',
  'SELECT DISTINCT category FROM productos order by 1'
)AS T (tipo TEXT, ELE INT, FG INT, IT INT);
```

Ejemplo 2. CROSSTAB con dos parámetros.

Profesor
M. en I.A. Erick Orlando Matla Cruz
Ayudantes
L. en C.C. Efraín Hipólito Chamú
L. en C.C. Anahí Quiroz Jiménez
L. en C.C. Karen Monserrat Zavala Correa

tipo	ele	fg	it
A		30	
B	70	60	

Resultado ejemplo 2.

3.2 Funciones de Ventana

Permiten aplicar una o múltiples funciones de agregación a un conjunto completo de tuplas o a una partición de este (subconjunto), los valores resultado de las funciones de agregación se repetirán en cada una de las tuplas del conjunto original sobre el que se está trabajando.

A continuación, se describe su uso.

```
SELECT    atributo1,…,atributon,
          Funcion_Agregación OVER(),
          Función_Agregación OVER(PARTITION BY atributo),
          Función_Agregación OVER(ORDER BY atributo)
```

Profesor
M. en I.A. Erick Orlando Matla Cruz
Ayudantes
L. en C.C. Efraín Hipólito Chamú
L. en C.C. Anahí Quiroz Jiménez
L. en C.C. Karen Monserrat Zavala Correa

En A.3 encontramos:

1. **SELECT ***

   Palabra reservada para indicar qué atributos desean obtenerse de la tabla-relación manipulada.

2. **OVER ()**

   Palabra reservada para indicar que la función de agregación que la precede debe ser aplicada a todo el conjunto de tuplas que se está manipulando.

3. **OVER (PARTITION BY *atributo*)**

   Palabra reservada para indicar que la función de agregación que la precede debe ser aplicada al subconjunto definido por los valores diferentes del atributo especificado.

4. **OVER (ORDER BY *atributo*)**

   Palabra reservada para indicar que la función de agregación que la precede debe ser aplicada al conjunto de tuplas ordenado con base en el atributo especificado.

5. **FROM**

   Palabra reservada para indicarle al SMBD la tabla-relación de la cual desean manipularse las tuplas para resolver la consulta.

6. **WHERE**

   Palabra reservada que permite indicar la condición que deben cumplir las tuplas que se desean manipular.

7. **;**

   Indica la finalización de la instrucción.

Ejemplos:

Siguiendo con el ejemplo de productos (Tabla 1), se presentan ejemplos para su uso de funciones de ventana.

Profesor
M. en I.A. Erick Orlando Matla Cruz
Ayudantes
L. en C.C. Efraín Hipólito Chamú
L. en C.C. Anahí Quiroz Jiménez
L. en C.C. Karen Monserrat Zavala Correa

category	avg
FG	45
ELE	56.6666666666666667
IT	42.5

Ejemplo 3. Promedio de total de productos por categoría.

La siguiente consulta presenta resultados mal planteados, se requiere mostrar cada producto con su categoría y promedio de total de productos (Ejemplo 4):

```
SELECT
      category,name_product,
      AVG (total_products)
FROM
      productos
GROUP BY category,name_product;
```

Ejemplo 4. Agrupación mal ejecutada.

Profesor
M. en I.A. Erick Orlando Matla Cruz
Ayudantes
L. en C.C. Efraín Hipólito Chamú
L. en C.C. Anahí Quiroz Jiménez
L. en C.C. Karen Monserrat Zavala Correa

category	name_product	avg
IT	Laptop	50
FG	Laptop	30
IT	Mobile	20
ELE	Laptop	70
ELE	Mobile	40
IT	Desktop	50
ELE	Desktop	60
FG	Desktop	60

Resultados Ejemplo 4.

Para resolver este conflicto es necesario usar funciones de ventana (Ejemplo 5). Donde seccionamos por category y enumeramos por el mismo atributo (notar que el orden es diferente al de id_product, usando una función para enumerar cada partición de categoría) .

Profesor
M. en I.A. Erick Orlando Matla Cruz
Ayudantes
L. en C.C. Efraín Hipólito Chamú
L. en C.C. Anahí Quiroz Jiménez
L. en C.C. Karen Monserrat Zavala Correa

```
SELECT
    id_product,name_product, category,
    ROW_NUMBER () OVER (PARTITION BY category ORDER BY
category) as orden_category,
  AVG(total_products) OVER (PARTITION BY category)
FROM
    productos;
```

Ejemplo 5. Consulta usando función de ventana

id_product	name_product	category	order_category	avg
6	Laptop	ELE	1	56.6666666666666667
4	Desktop	ELE	2	56.6666666666666667
2	Mobile	ELE	3	56.6666666666666667
7	Desktop	FG	1	45
8	Laptop	FG	2	45
9	Laptop	IT	1	42.5
3	Desktop	IT	2	42.5
5	Laptop	IT	3	42.5
1	Mobile	IT	4	42.5

Resultados de Ejemplo 5.

El siguiente ejemplo presenta la acumulación de productos por orden de aparición de id_product, Ejemplo 6.

```
SELECT
    id_product,name_product, category,total_products,
    SUM(total_products) OVER (ORDER BY id_product DESC)
FROM
    productos;
```

Ejemplo 6.

id_product	name_product	category	total_products	sum
9	Laptop	IT	70	70
8	Laptop	FG	30	100
7	Desktop	FG	60	160
6	Laptop	ELE	70	230
5	Laptop	IT	30	260
4	Desktop	ELE	60	320
3	Desktop	IT	50	370
2	Mobile	ELE	40	410
1	Mobile	IT	20	430

Resultados de Ejemplo 6.

4. Ejercicios

(NOTA: Resuelve los siguientes ejercicios en relación al proyecto que realizarás durante el curso, en dado caso que no tengas un proyecto, utiliza la base de datos "Aseguradora")

1. Realiza 2 consultas a diferentes tablas donde utilices la llamada a la función CROSSTAB con un solo parámetro.
2. Realiza 2 consultas a diferentes tablas donde utilices la llamada a la función CROSSTAB con dos parámetros.

Profesor
M. en I.A. Erick Orlando Matla Cruz
Ayudantes
L. en C.C. Efraín Hipólito Chamú
L. en C.C. Anahí Quiroz Jiménez
L. en C.C. Karen Monserrat Zavala Correa

3. Realiza 2 consultas a diferentes tablas donde utilices el operador OVER (),
4. Realiza 2 consultas donde utilices el operador OVER (PARTITION BY).
5. Realiza 2 consultas donde utilices el operador OVER (ORDER BY).

Entregables requeridos para prácticas subsecuentes, un archivo en formato SQL:

- 2 consultas a diferentes tablas y CROSSTAB con un solo parámetro.
- 2 consultas a diferentes tablas y CROSSTAB con dos parámetros.
- 2 consultas a diferentes tablas y operador OVER ().
- 2 consultas con el operador OVER (PARTITION BY).
- 2 consultas con el operador OVER (ORDER BY).