

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO  
FACULTAD DE INGENIERÍA  
ESCUELA DE INGENIERÍA INFORMÁTICA

## **TALLER DE BASES DE DATOS**

**JEAN RODRÍGUEZ**

INFORME DE TALLER V

OCTUBRE, 2019

## Índice

1. Introducción	1
2. Resolución de ejercicios	2
3. Conclusiones	6

**Lista de Figuras**

1.	Primer ejercicio - resuelto . . . . .	2
2.	Primer ejercicio - error . . . . .	3
3.	Primer ejercicio - causa . . . . .	3
4.	Segundo ejercicio . . . . .	4
5.	Tercer ejercicio . . . . .	5

## **1. Introducción**

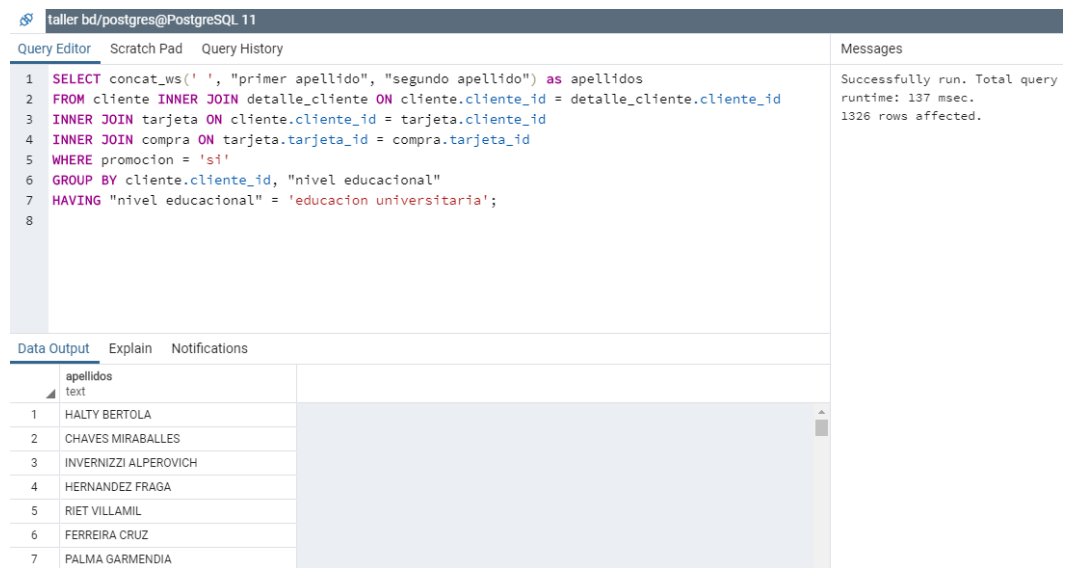
Los modelos de negocio actuales manejan cantidades enormes de datos de los clientes en cada empresa, sea grande o pequeña, lo que evidencia una clara necesidad de un software capaz de procesar las solicitudes requeridas. Cuando una entidad desea minimizar la complejidad de sus operaciones y mantener tiempos de respuesta aceptables, se debe dejar en claro un diseño simple y eficiente que sea mantenible y facilite la realización de cambios a futuro.

Con la intención de demostrar un correcto funcionamiento en la base de datos propuesta, se deja a disposición una serie de consultas para el motor PostgreSQL que entregan respuestas de manera rápida y efectiva, utilizando como base el script inicial para cargar la base de datos, y su correspondiente archivo de valores separado por comas.

## 2. Resolución de ejercicios

A continuación se presenta el conjunto de preguntas y respuestas correspondientes en el lenguaje PostgreSQL con los accesos necesarios a las tablas y sus llaves.

1.- Realice una consulta que muestre el apellido con estado de nivel educacional estudiante universitario que compre productos en promoción.



The screenshot shows a PostgreSQL query editor interface. The top bar indicates the connection is to 'taller bd/postgres@PostgreSQL 11'. The interface is divided into three main sections: 'Query Editor', 'Messages', and 'Data Output'.

**Query Editor:** Contains the following SQL query:

```
1 SELECT concat_ws(' ', "primer apellido", "segundo apellido") as apellidos
2 FROM cliente INNER JOIN detalle_cliente ON cliente.cliente_id = detalle_cliente.cliente_id
3 INNER JOIN tarjeta ON cliente.cliente_id = tarjeta.cliente_id
4 INNER JOIN compra ON tarjeta.tarjeta_id = compra.tarjeta_id
5 WHERE promocion = 'si'
6 GROUP BY cliente.cliente_id, "nivel educacional"
7 HAVING "nivel educacional" = 'educacion universitaria';
8
```

**Messages:** Displays the execution status: 'Successfully run. Total query runtime: 137 msec. 1326 rows affected.'

**Data Output:** Shows the results of the query in a table with two columns: 'apellidos' (text) and 'nivel educacional' (text). The results are as follows:

apellidos	nivel educacional
HALTY BERTOLA	educacion universitaria
CHAVES MIRABALLES	educacion universitaria
INVERNIZZI ALPEROVICH	educacion universitaria
HERNANDEZ FRAGA	educacion universitaria
RIET VILLAMIL	educacion universitaria
FERREIRA CRUZ	educacion universitaria
PALMA GARMENDIA	educacion universitaria

Figura 1: Primer ejercicio - resuelto

Como se puede apreciar, la cláusula GROUP BY utiliza el id del cliente pese a ser una clave primaria, debido a que sin ella el script no carga correctamente (ver figura 2). Para mas información, se deja ver la causa posteriormente y el enlace al sitio[1], aclarando que es un actuar aceptado para manejar las dependencias funcionales definidas en la base de datos.



Figura 2: Primer ejercicio - error

When GROUP BY is present, or any aggregate functions are present, it is not valid for the SELECT list expressions to refer to ungrouped columns except within aggregate functions or when the ungrouped column is functionally dependent on the grouped columns, since there would otherwise be more than one possible value to return for an ungrouped column. A functional dependency exists if the grouped columns (or a subset thereof) are the primary key of the table containing the ungrouped column.

Figura 3: Primer ejercicio - causa

2.- Realice una consulta que indique la cantidad de soltero con cupo máximo sobre \$300.000



The screenshot shows a PostgreSQL query editor interface. The title bar indicates the connection is to 'taller bd/postgres@PostgreSQL 11'. The 'Query Editor' tab is active, displaying a SQL query. The query counts the number of single clients (soltero) whose maximum credit limit is greater than 300,000. The query is as follows:

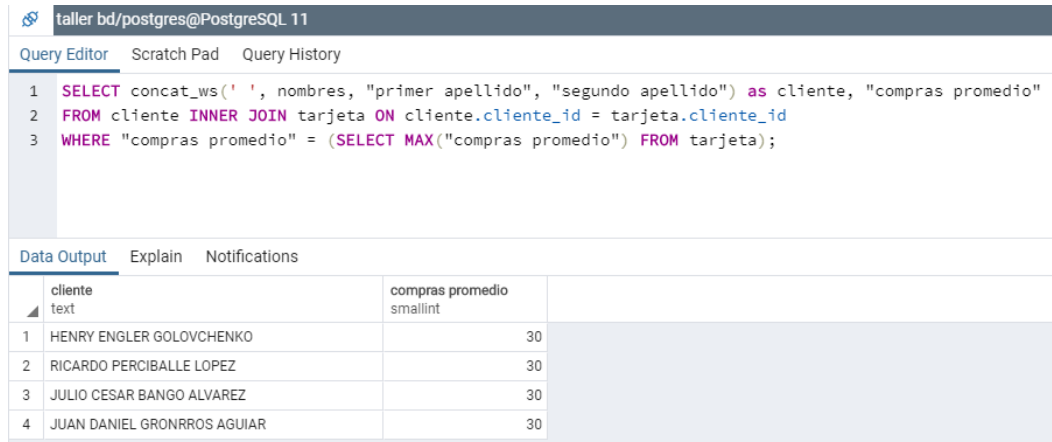
```
1 SELECT COUNT(*) AS total
2 FROM cliente INNER JOIN detalle_cliente ON cliente.cliente_id = detalle_cliente.cliente_id
3 INNER JOIN tarjeta ON cliente.cliente_id = tarjeta.tarjeta_id
4 WHERE "cupo maximo" > 300000
5 GROUP BY "estado civil"
6 HAVING "estado civil" = 'soltero';
7
```

Below the query editor, the 'Data Output' tab is active, showing the results of the query. The results are displayed in a table with two columns: 'total' and 'bigint'. The first row shows a count of 2315 for the 'soltero' status.

	total bigint
1	2315

Figura 4: Segundo ejercicio

3.- Realice una consulta que muestre en nombres de clientes que más ha comprado en el año.



The screenshot shows a PostgreSQL query editor interface. At the top, the title bar reads "taller bd/postgres@PostgreSQL 11". Below it are tabs for "Query Editor", "Scratch Pad", and "Query History". The "Query Editor" tab is active and contains the following SQL query:

```
1 SELECT concat_ws(' ', nombres, "primer apellido", "segundo apellido") as cliente, "compras promedio"
2 FROM cliente INNER JOIN tarjeta ON cliente.cliente_id = tarjeta.cliente_id
3 WHERE "compras promedio" = (SELECT MAX("compras promedio") FROM tarjeta);
```

Below the query editor, there are tabs for "Data Output", "Explain", and "Notifications". The "Data Output" tab is active and displays the results of the query in a table format:

	cliente text	compras promedio smallint
1	HENRY ENGLER GOLOVCHENKO	30
2	RICARDO PERCIBALLE LOPEZ	30
3	JULIO CESAR BANGO ALVAREZ	30
4	JUAN DANIEL GRONRROS AGUIAR	30

Figura 5: Tercer ejercicio



### **3. Conclusiones**

Del presente informe se obtuvo en detalle el proceso de organización necesario para conformar la base de datos propuesta para el cliente ficticio, de tal manera que facilita la construcción del software que debe utilizar las variables necesarias para que el negocio funcione de manera exitosa, dejándose expuesta una interpretación clara del modelo definido para entregar tiempos de respuesta breves en futuras consultas.

Además, se obtuvo resultados satisfactorios a la hora de utilizar funciones de agregación y subqueries, lo que en algunos casos facilita la obtención de datos para el usuario.

## Referencias

- [1] The PostgreSQL Global Development Group. <https://www.postgresql.org/docs/current/sql-select.html#SQL-GROUPBY>.