**Tutorial Bases de datos columnares (video relacionado: 5.6 Distribution styles. Contenido relacionado: Tipos de Distribution Styles en Amazon redshift)**

Actualmente trabajas en un banco popular de la ciudad, en tu primer día de trabajo te reunes con el Data Manager, quien te pide que analices qué tan ventajoso puede ser utilizar una base de datos columnar como amazon Redshift en comparación con una base de datos orientada a filas como PostgreSQL a la hora de hacer consultas de tipo analitica teniendo en cuenta la siguiente [base de datos](https://drive.google.com/file/d/1psT9IvEwCIJ-TjILeHFaAwgtMHBI2CpS/view?usp=sharing) y las siguientes restricciones:

1. La descripción de las columnas se encuentra [aquí](https://drive.google.com/file/d/130cQMDy7KvnThGh24zcX1577j8_ATinW/view?usp=sharing)
2. Nuestras variables de interés son:
3. TARGET,"Target variable (1 - client with payment difficulties: he/she had late payment more than X days on at least one of the first Y installments of the loan in our sample, 0 - all other cases)"
4. NAME\_CONTRACT\_TYPE,Identification if loan is cash or revolving
5. CODE\_GENDER,Gender of the client,
6. AMT\_CREDIT,Credit amount of the loan

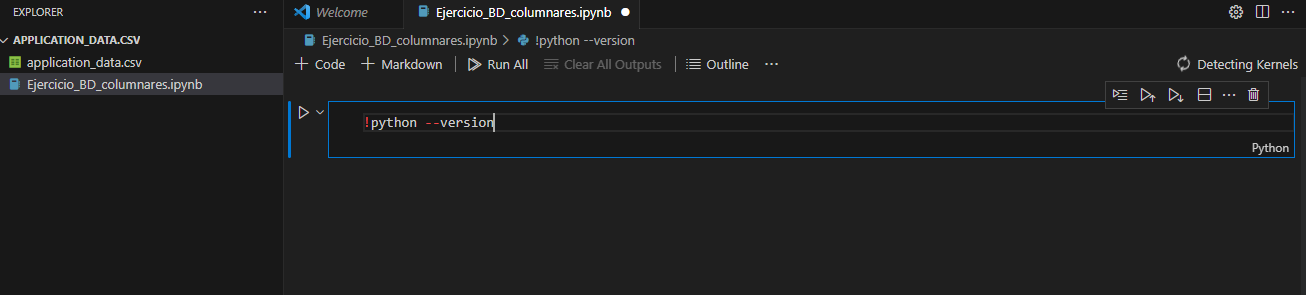
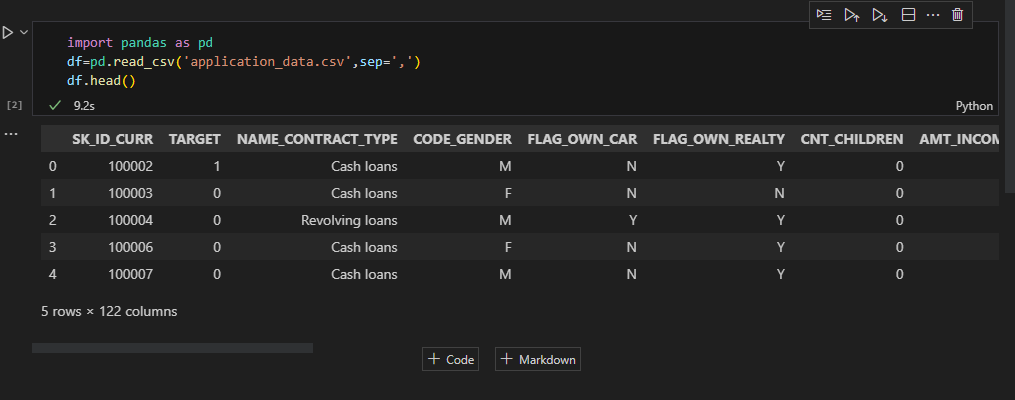
1. El objetivo será obtener agrupaciones del AMT:CREDIT de acuerdo con tipo de contrato, género y TARGET.

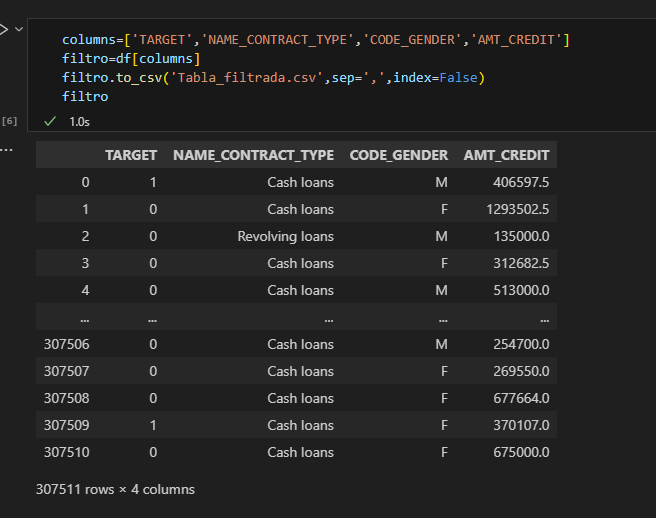
**Solución:**

1. Descargamos el dataset y lo descomprimimos**:**



1. Luego abrimos nuestro editor de código (e.g Visual studio) y leemos nuestro archivo con pandas:



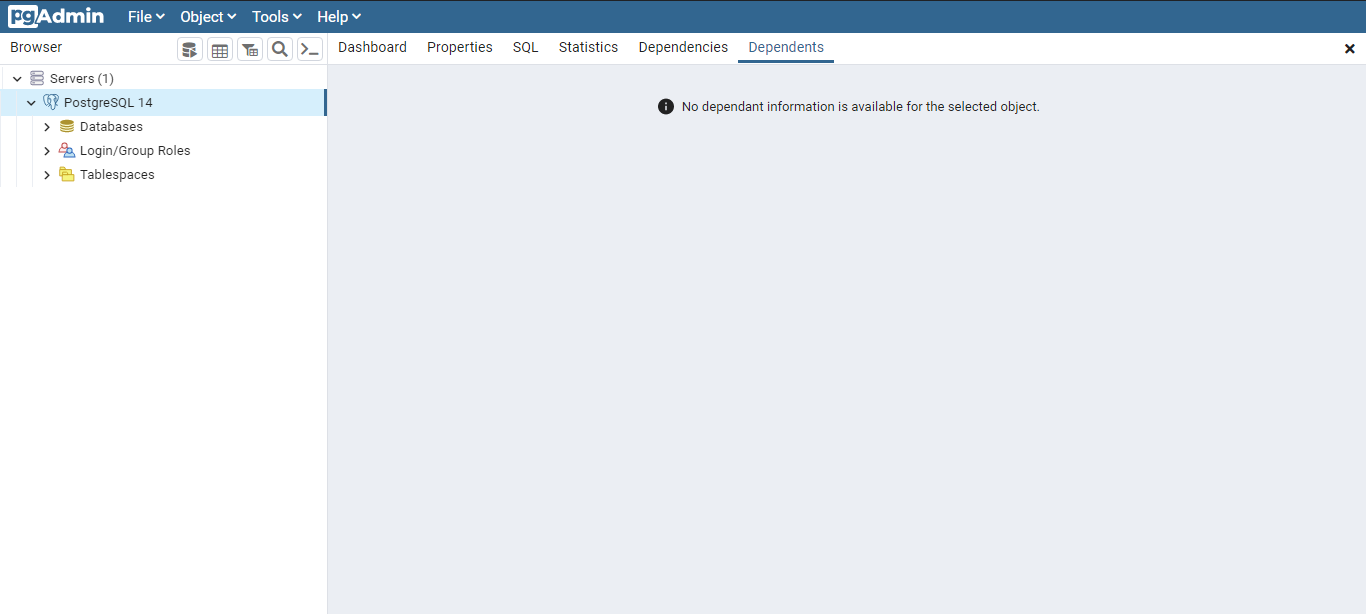
1. Hacemos el subset de las variables de interés y exportamos esta tabla como csv:

**NOTA: TE RECOMIENDO USAR SOLO 5000 filas del dataset para que no demores demasiado importando los datos en Redshift.**

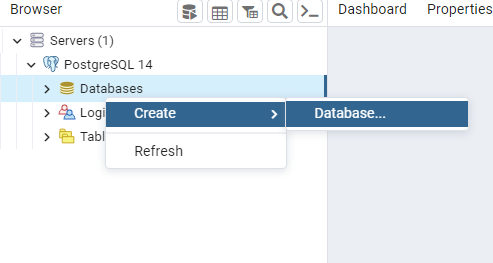
1. Ahora podemos abrir PgAdmin:



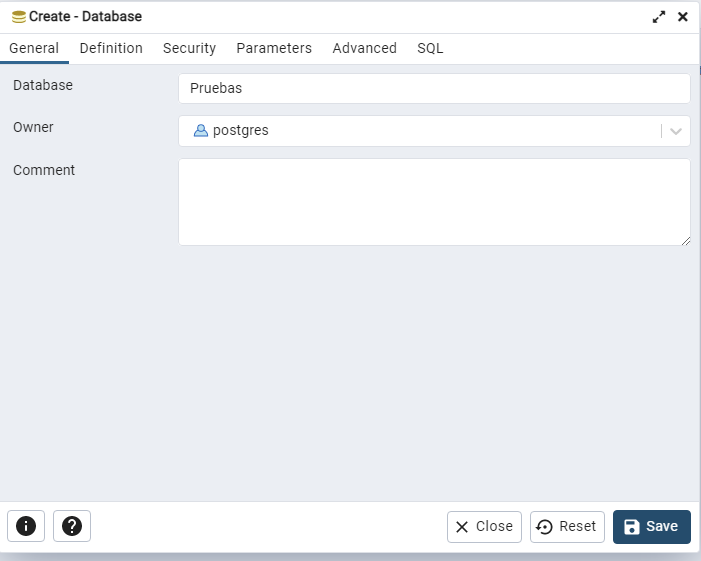
1. Dentro de pgadmin podemos crear una base de datos o utilizar alguna ya existen:



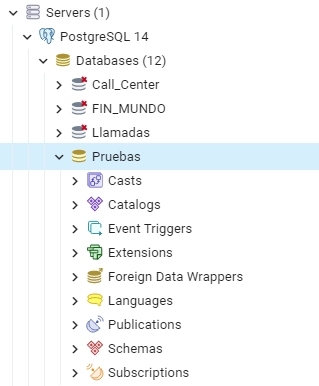
1. Creamos una base de datos llamada Pruebas dando click derecho en Databases > Crear Database:



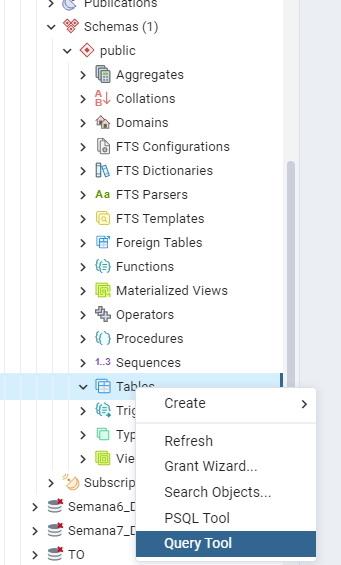
1. Podemos dejar todo por default asignamos el nombre Pruebas y damos en Save:



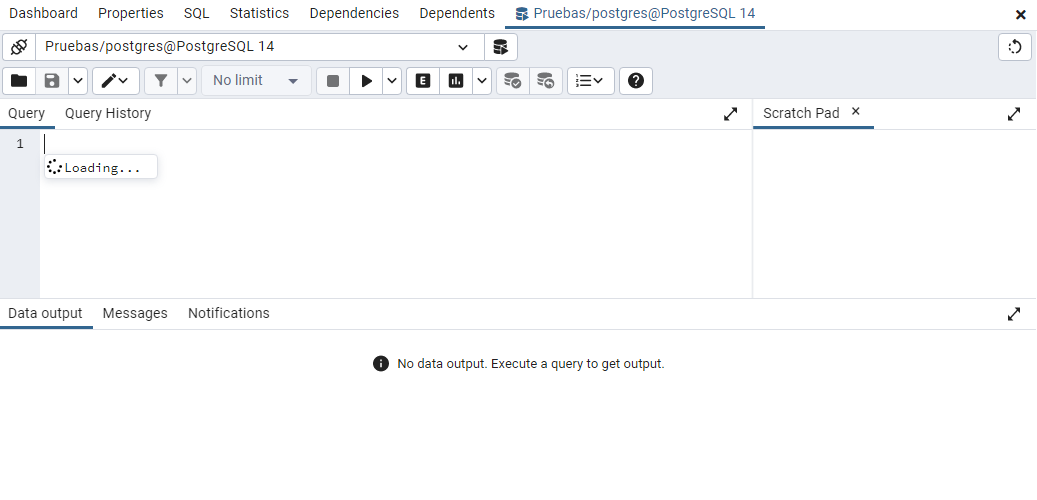
1. Ahora debería aparecer nuestra base de datos:



1. Vamos donde dice esquemas > Tablas > Click derecho y luego seleccionamos Query Tool:



1. Aparecerá una ventana donde podremos crear nuestra tabla:



1. Creamos nuestra tabla con el siguiente comando de sql:

```sql

create table fraude (

TARGET bool,

NAME\_CONTRACT\_TYPE VARCHAR(50),

CODE\_GENDER VARCHAR(5),

AMT\_CREDIT FLOAT

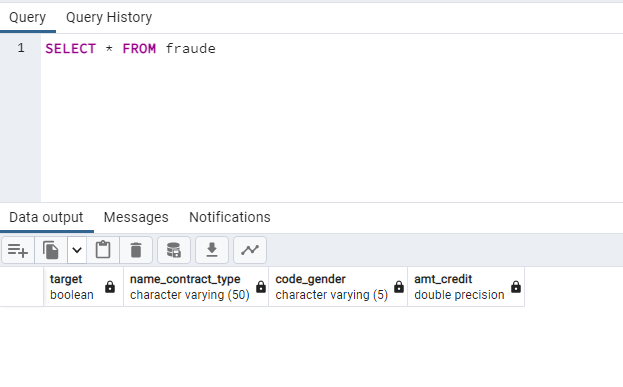
);

```

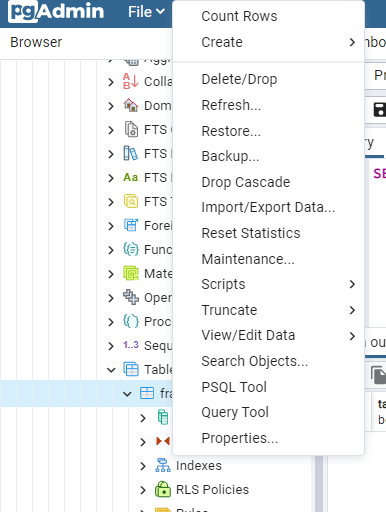
Así entonces deberíamos tener nuestra tabla creada:



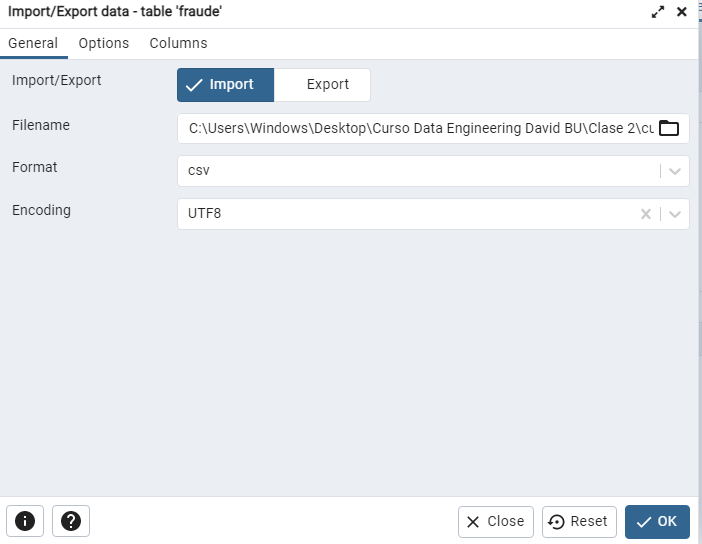
1. Ahora podemos ver nuestra tabla:



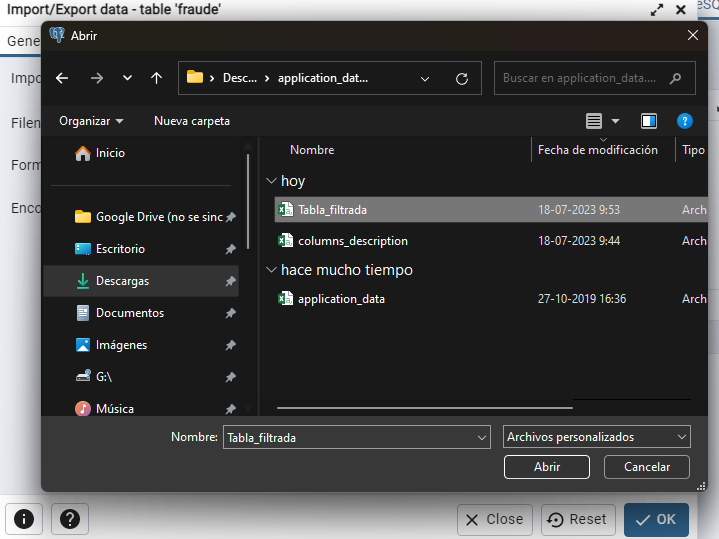
1. Procedemos a llenar la tabla con registros. Click derecho en el nombre de la tabla > Import/Export:



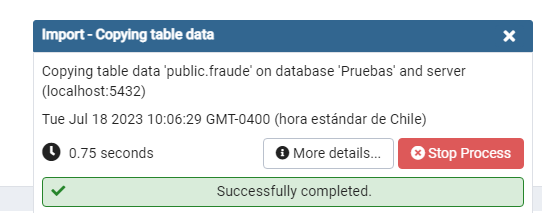
1. Aparecerá una ventana como esta:



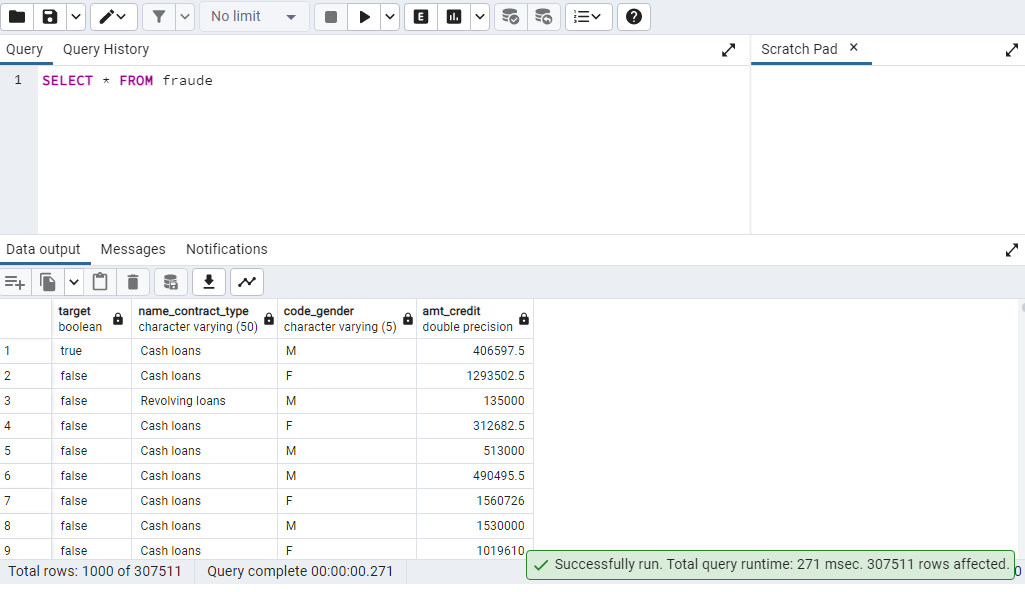
1. Seleccionamos import y elegimos nuestro archivo:



1. Damos en OK y deberá salir un mensaje como este:



1. Verificamos nuestra tabla:



1. Ahora realizamos diversas columnas utilizando funciones de agregación:

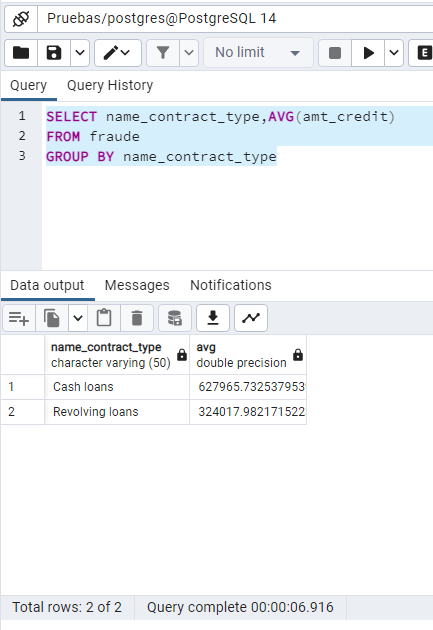
```sql

SELECT name\_contract\_type,AVG(amt\_credit)

FROM fraude

GROUP BY name\_contract\_type

```



Notarás que se demora cerca de 6.9 segundos, ten este tiempo presente para las comparaciones:

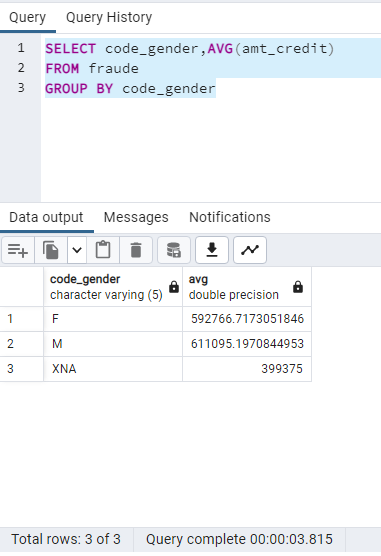
```sql

SELECT code\_gender,AVG(amt\_credit)

FROM fraude

GROUP BY code\_gender

```



En este otro caso demora cerca de 3.8 seg

**Ahora bien si hacemos algo más sofisticado como esto notarás que tarda cerca de ~20seg**

```sql

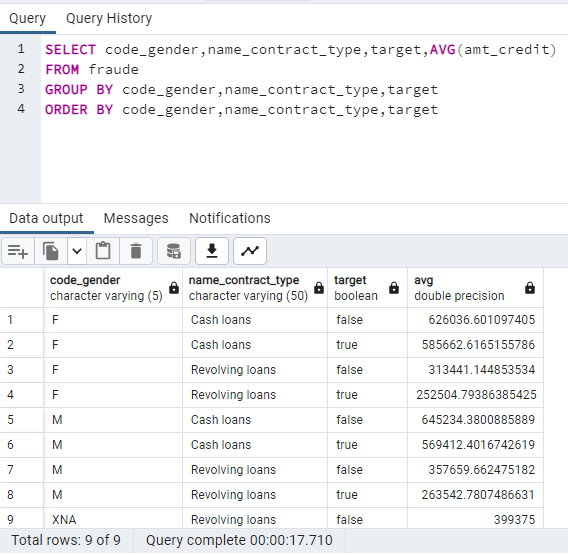
SELECT code\_gender,name\_contract\_type,target,AVG(amt\_credit)

FROM fraude

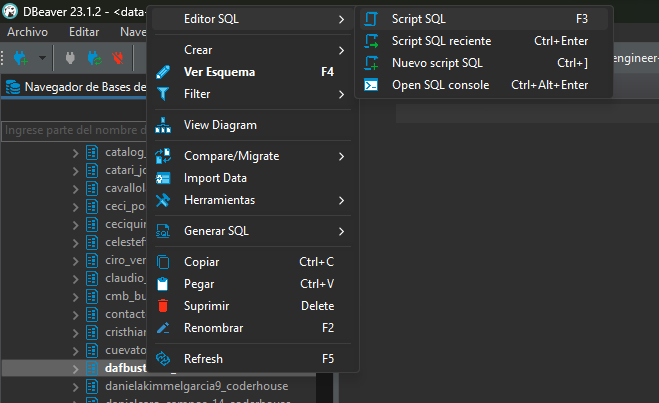
GROUP BY code\_gender,name\_contract\_type,target

ORDER BY code\_gender,name\_contract\_type,target

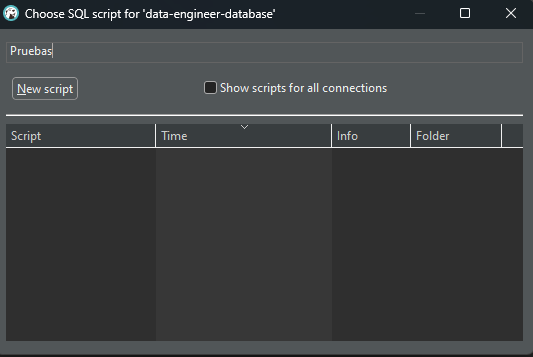
```



1. Ahora hagamos lo mismo en Amazon Redshift. abrimos Dbeaver. elegimos nuestro nombre de esquema > Click derecho > Editor de SQL > New Script



1. Ponemos un nombre a nuestro script y damos enter:



1. Creamos nuestra tabla:

```sql

create table fraude (

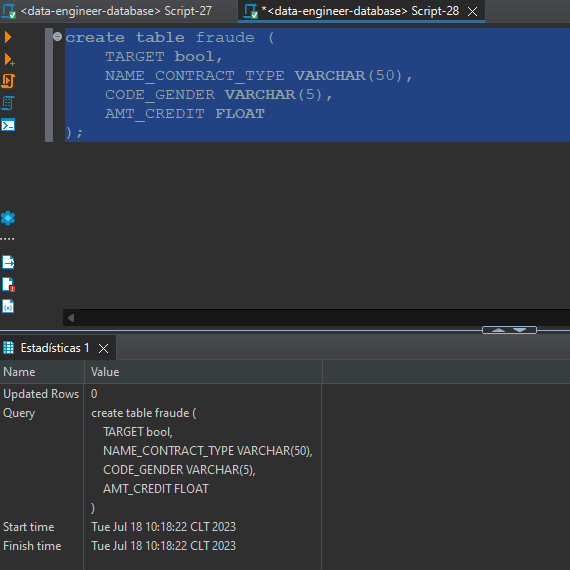
TARGET bool,

NAME\_CONTRACT\_TYPE VARCHAR(50),

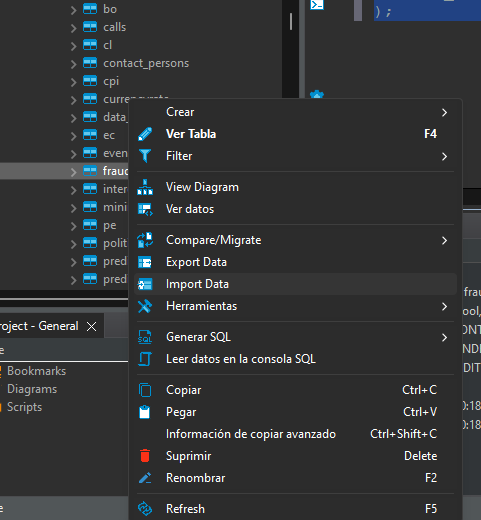
CODE\_GENDER VARCHAR(5),

AMT\_CREDIT FLOAT

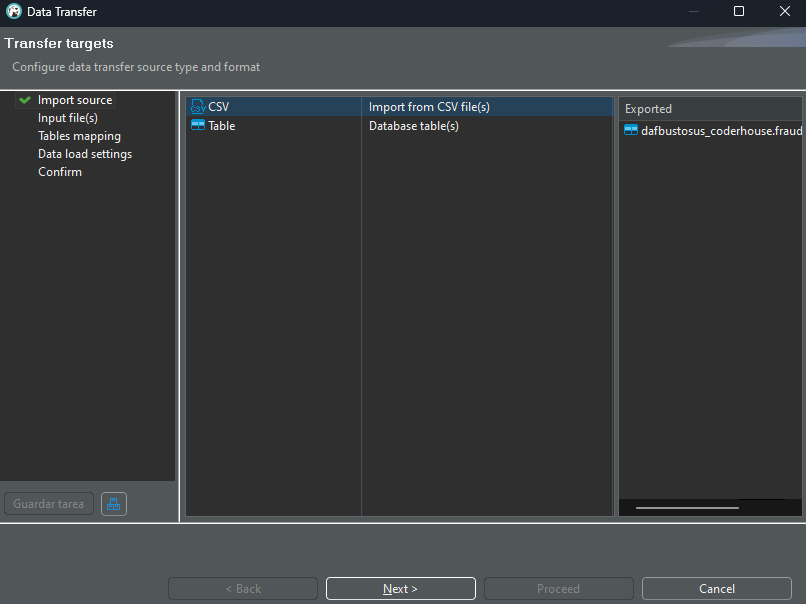
);

```

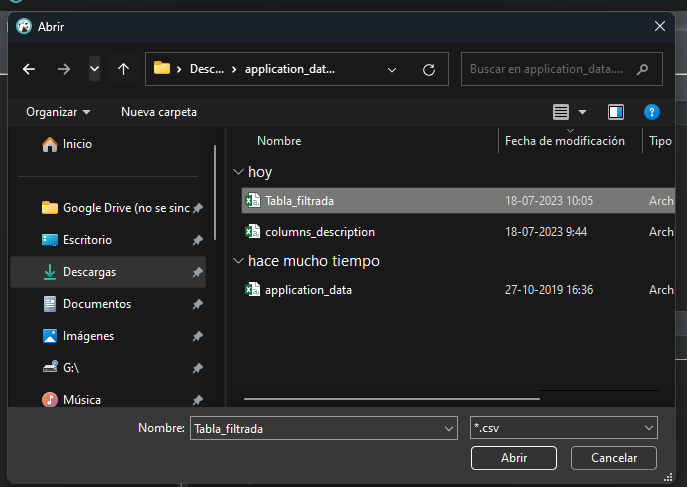
1. Ahora para insertar nuestros registros podemos elegir nuestra tabla > click derecho > Import data:

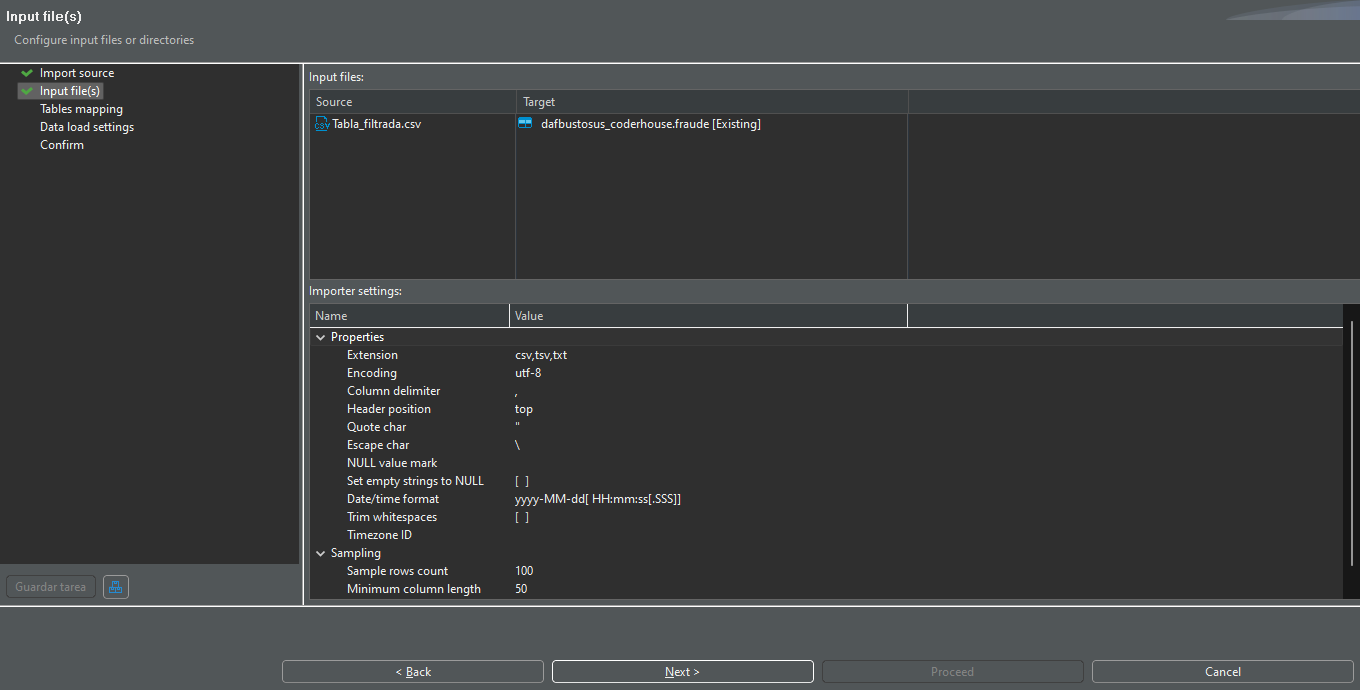


1. Aparecerá esta ventana, damos click en CSV y Next:

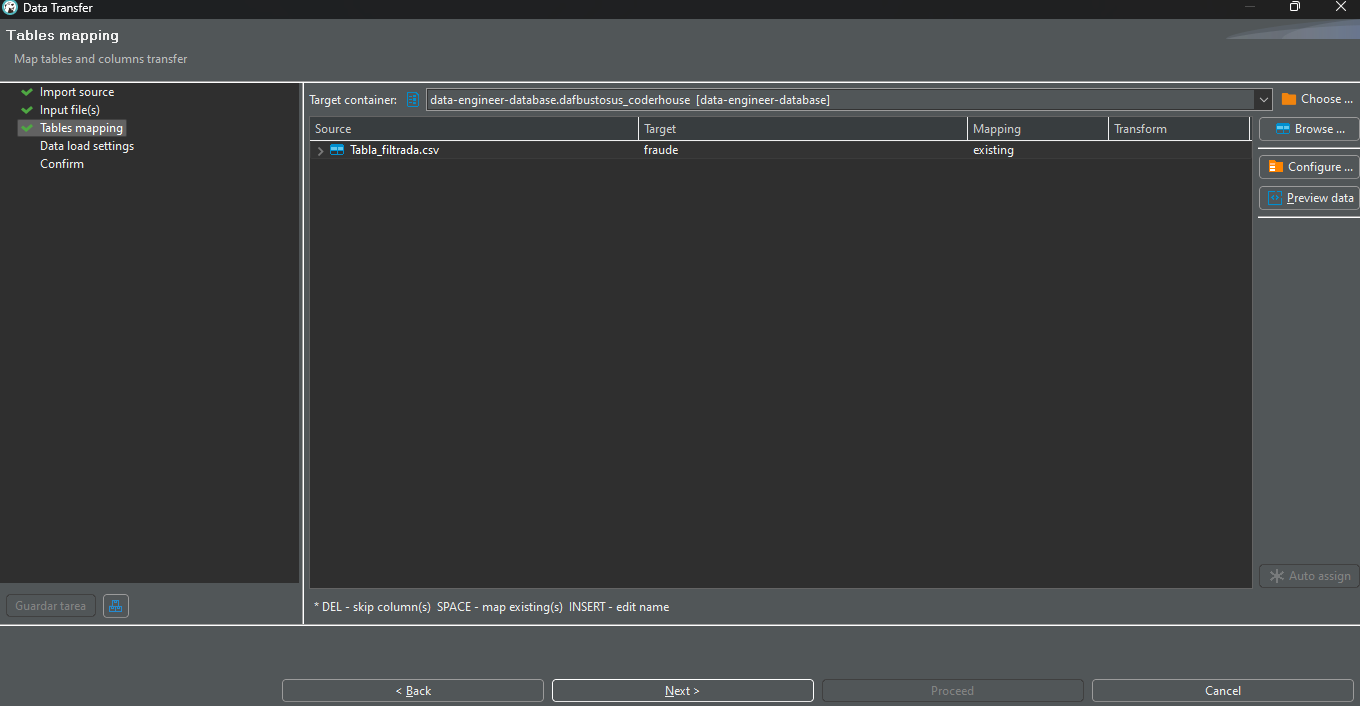


1. Luego elegimos nuestra fuente de datos:

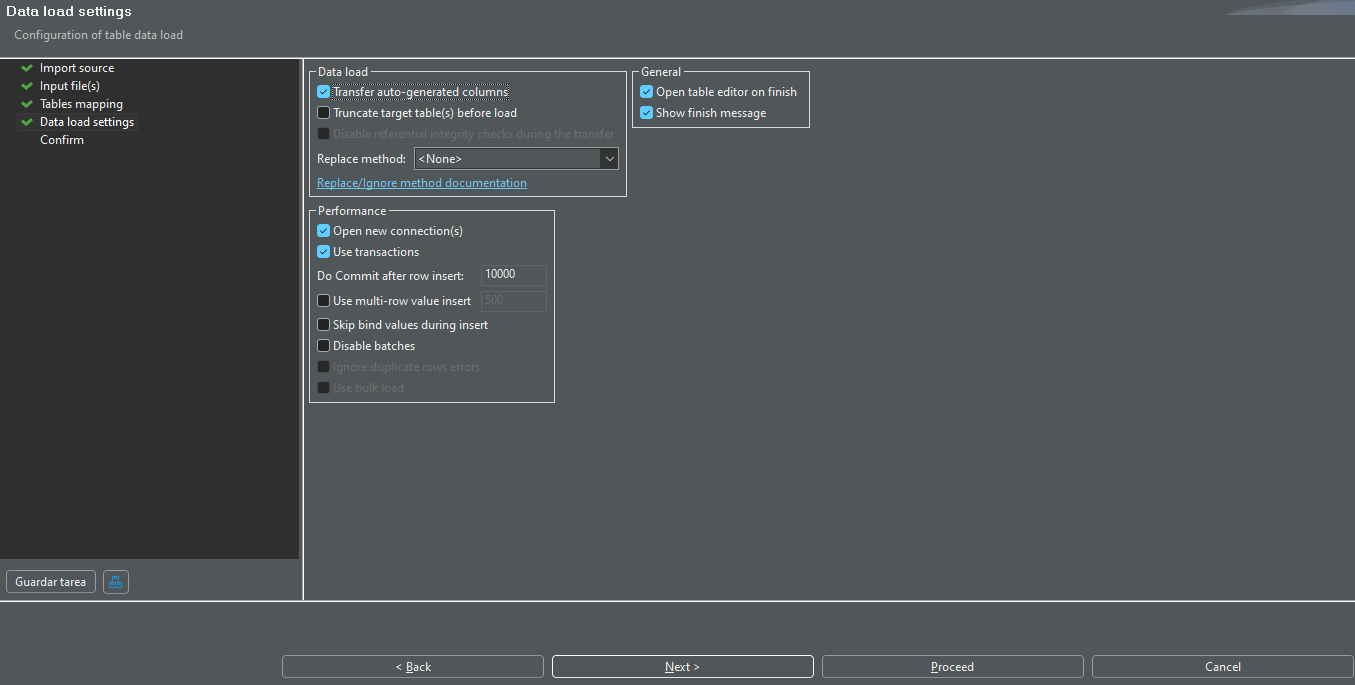




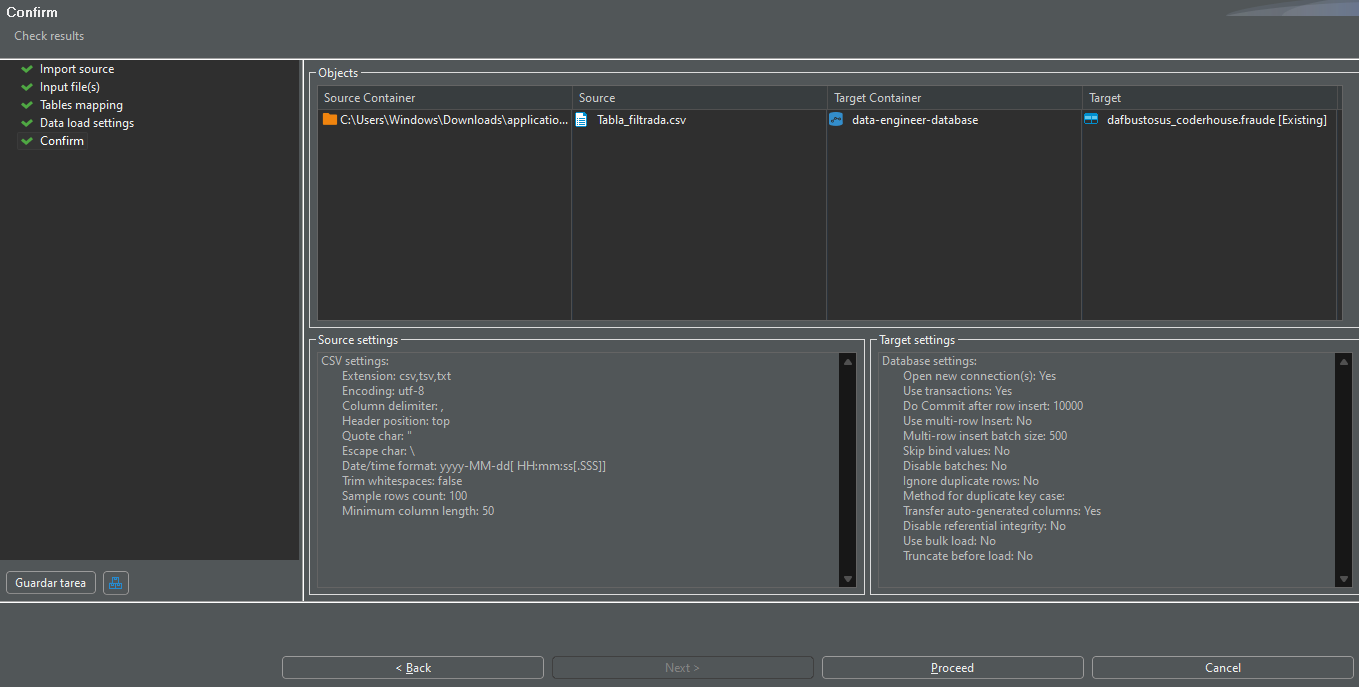
1. Elegimos la tabla de destino que en este caso se llama fraude y damos en Next:



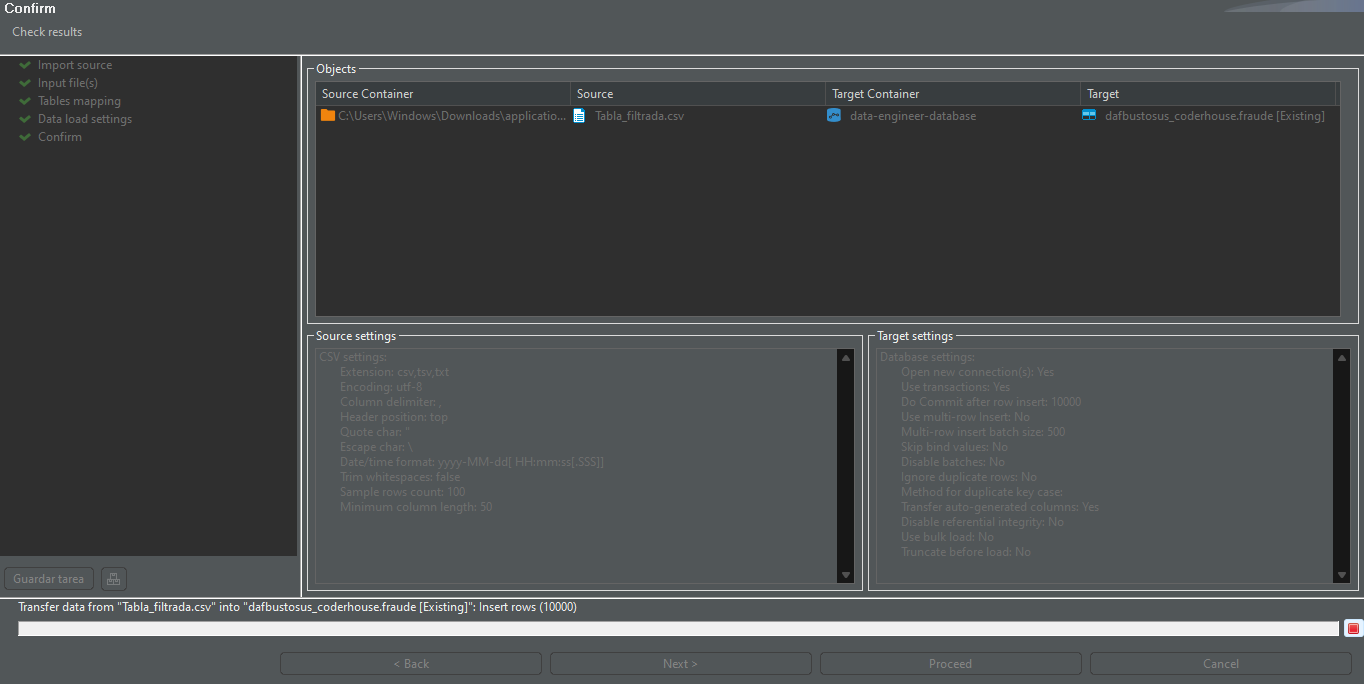
1. Dejamos la configuración por default y damos en Next:

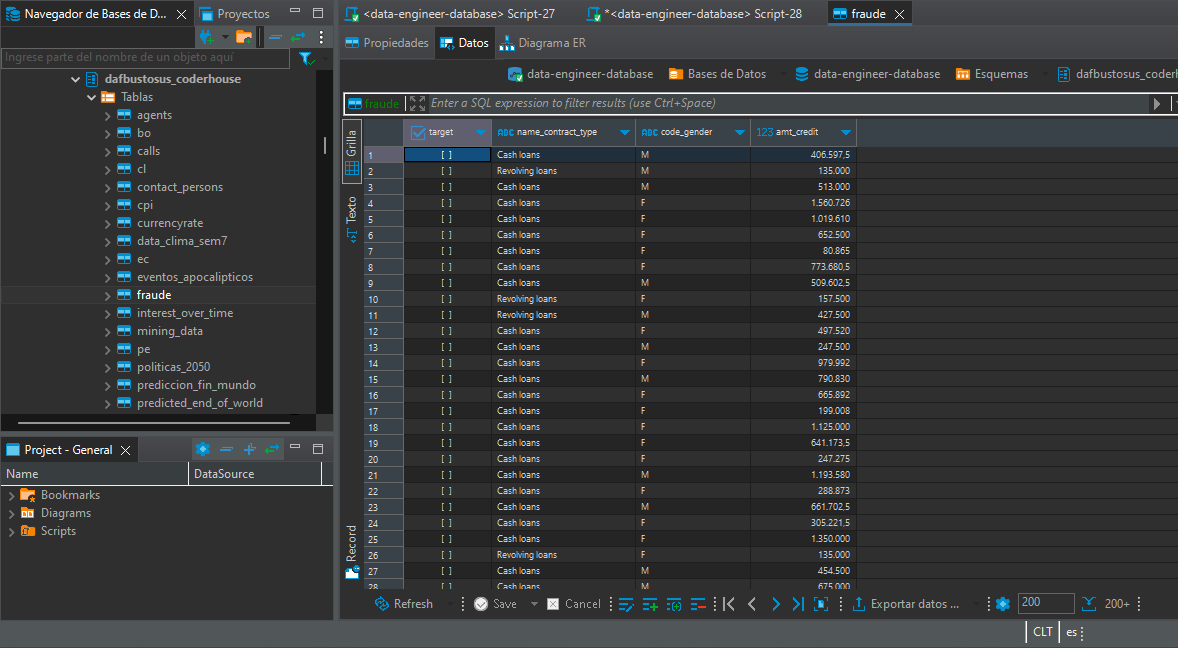


1. Finalmente damos click en Proceed y esperamos a que carguen los datos:



1. Esto probablemente tome bastante tiempo así que ten paciencia, ya que como recordarás que Amazon Redshift es una base de datos columnar por ende es bastante lenta para cargar información de transacciones.





1. Ahora hacemos las mismas consultas que habíamos hecho en la base de datos PostgreSQL y comparemos los tiempos de ejecución

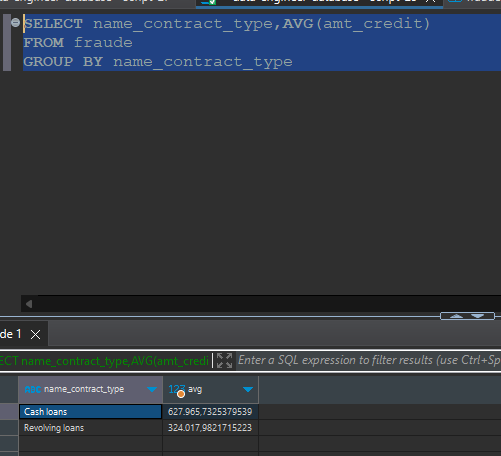
```sql

SELECT name\_contract\_type,AVG(amt\_credit)

FROM fraude

GROUP BY name\_contract\_type

```

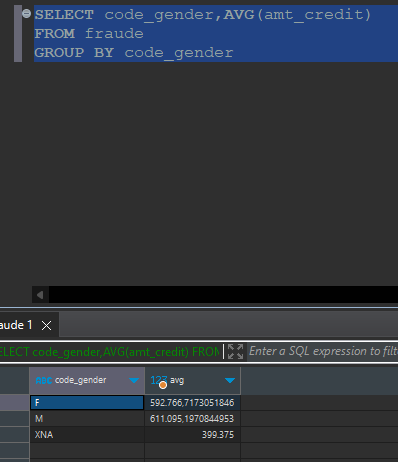


Aquí se demora 1.6 seg

```sql

SELECT code\_gender,AVG(amt\_credit)

FROM fraude

GROUP BY code\_gender

```

2.3 seg en esta

```sql

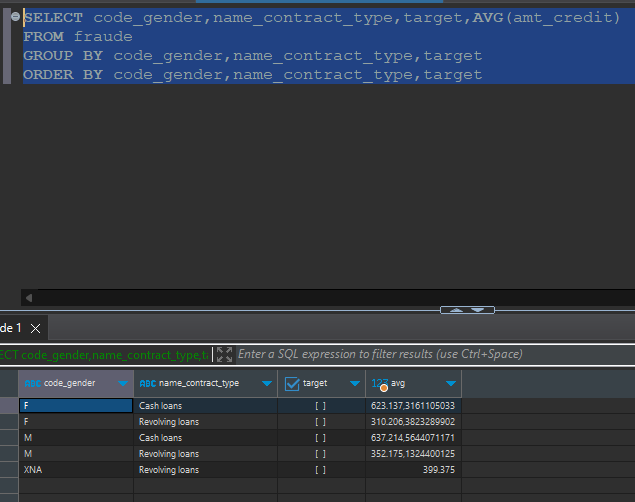
SELECT code\_gender,name\_contract\_type,target,AVG(amt\_credit)

FROM fraude

GROUP BY code\_gender,name\_contract\_type,target

ORDER BY code\_gender,name\_contract\_type,target

```



En esta última se demora solo 6.1 seg

1. En resumen tenemos esto:

|  | Amazon RD (seg) | PostgreSQL (seg) |
| --- | --- | --- |
| Consulta 1 | 1.6 | 6.9 |
| Consulta 2 | 2.3 | 3.8 |
| Consulta 3 | 6.1 | 17.7 |

**Conclusión a presentarte a tu Data Leader:** Claramente se ve que Amazon RD es mucho más rápido para consultas analiticas que involucran agrupaciones en comparación con bases de datos como PostgreSQL que son mucho mejores para sistemas transaccionales de inserción de filas.