SPRINT 4

Último repaso de SQL en el que trabajaremos la creación de bases de datos, las relaciones entre tablas, técnicas de modelado y granularidad de los datos.

Raquel Limpo Martínez

Tabla de contenido

[Nivel 1 2](#_Toc183690439)

[Esquema en estrella 2](#_Toc183690440)

[Creación de la base de datos e importación de los datos de los CSV 5](#_Toc183690441)

[Ejercicio 1: Realiza una subconsulta que muestre a todos los usuarios con más de 30 transacciones utilizando al menos 2 tablas. 11](#_Toc183690442)

[Ejercicio 2 Muestra la media de amount por IBAN de las tarjetas de crédito a la compañía Donec Ltd., utiliza por lo menos 2 tablas. 12](#_Toc183690443)

[Nivel 2 14](#_Toc183690444)

[Ejercicio 1: Crea una nueva tabla que refleje el estado de las tarjetas de crédito basado en si las últimas tres transacciones fueron declinadas y genera la siguiente consulta: ¿Cuantas tarjetas están activas? 14](#_Toc183690445)

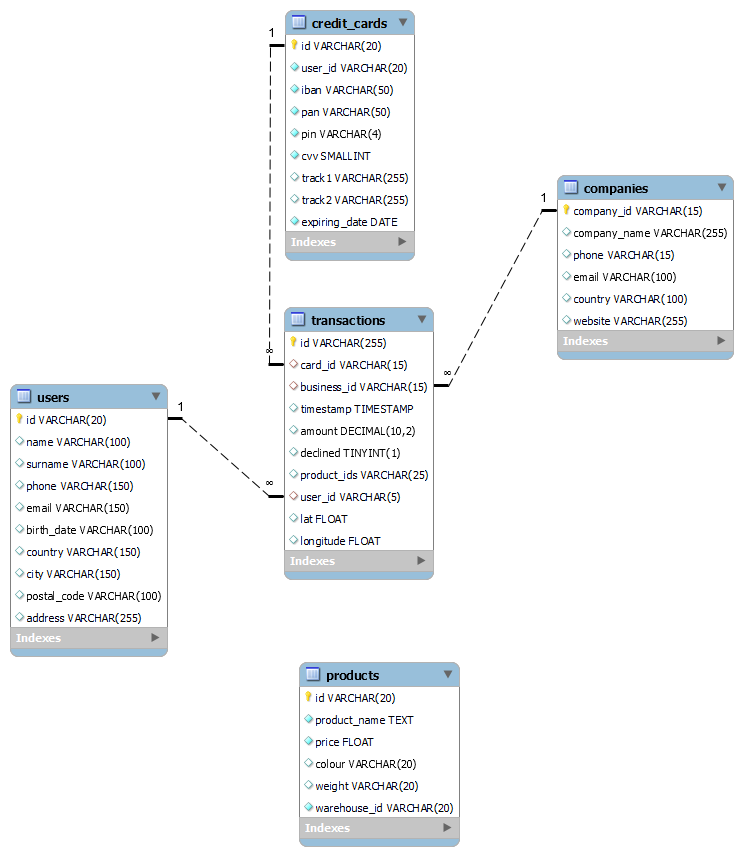
[Nivel 3 17](#_Toc183690446)

[Ejercicio 1: Necesitamos conocer el número de veces que se ha vendido cada producto. 19](#_Toc183690447)

[Anexo: Los problemas de carga las tablas 21](#_Toc183690448)

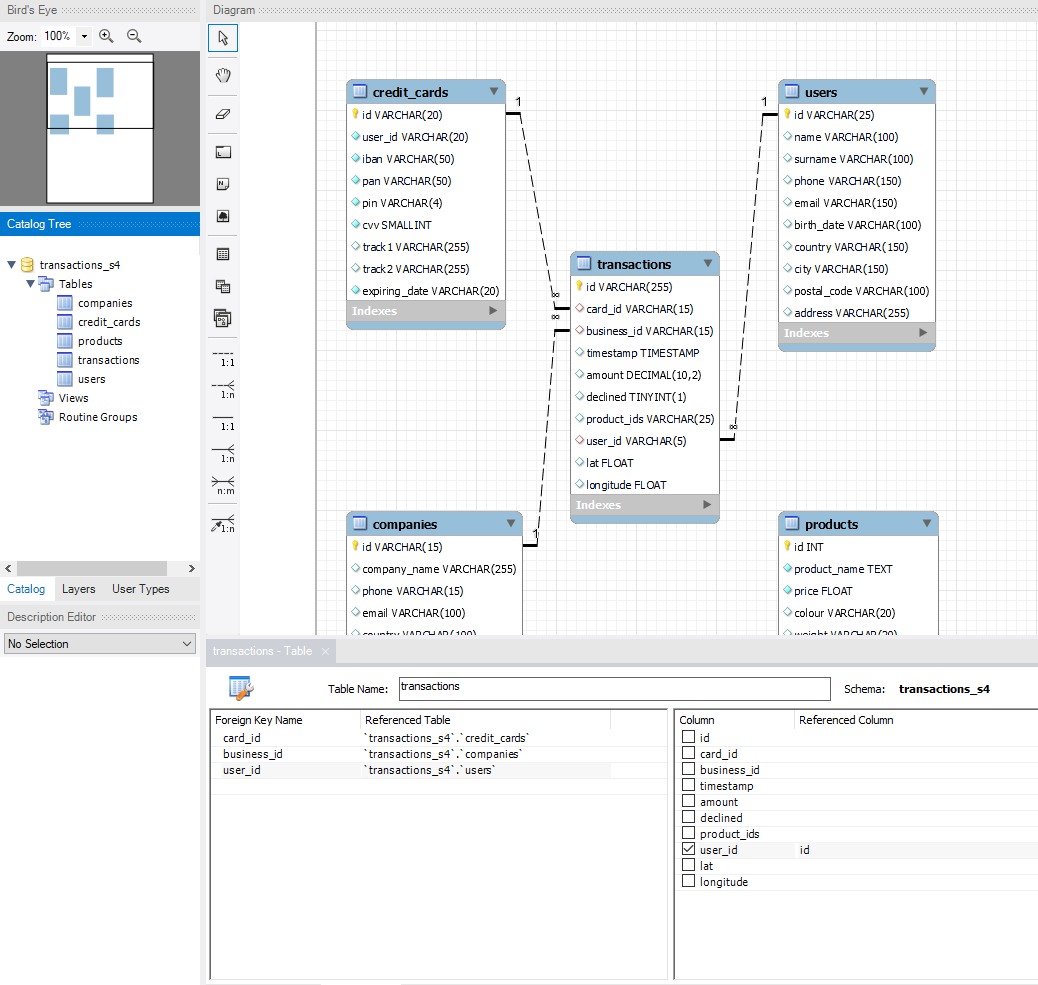
## Nivel 1

Descarga los archivos CSV, estúdialos y diseña una base de datos con un esquema en estrella que contengas al menos 4 tablas de las cuales puedas realizar las consultas de los ejercicios 1 y 2:

Esquema en estrella  


En este esquema de estrella, la tabla factual o de hechos sería **transactions**, no solo porque está conectada con todas las demás, sino que contiene los indicadores del negocio, cuanto se ha vendido , cuando a quien de parte de quien, cuánto dinero se ha ganado…

Está unida a las tablas de dimensiones: **companies, credit\_cards y users.** No me deja conectarla a **products**, porque la foreign key es muchos a muchos. Se soluciona en el nivel 3.



**Tabla transactions:**

* id : El identificador de la transacción realizada. Es la clave primaria de esta tabla Como he comentado previamente, sería ideal que este campo fuera distintivo como *transaction\_id.* Lo mismo aplica para el resto de tablas.
* card\_id : El identificador de la tarjeta de crédito con la que se ha realizado la transacción, nos une a la tabla *Credit\_card*
* business\_id : El identificador de la compañía desde la que ha realizado la transacción, nos une a la tabla *Company.*
* timestamp : La hora a la que se ha realizado la transacción
* amount : La cantidad de dinero que se ha usado en la transacción
* declined : Campo booleano, nos indica si la transacción se ha realizado (0) o se ha cancelado (1)
* product\_ids : los identificadores de los diferentes productos que se han intercambiado por dinero en la transacción. Cuando hay más de uno está separado por comas. Nos uniriamos a la tabla *Products* mediante una tabla intermedia que pondremos en el nivel 3..
* user\_id : El identificador del usuario que ha realizado la transacción. Nos une por clave foránea a las tablas *Users*
* lat y longitude: Las coordenadas desde donde se ha hecho la transacción.

**Tabla company:**

* company\_id: El identificador para cada compañía. ES la clave primaria de esta tabla.
* company\_name. El nombre de la compañía.
* phone : El número de contacto de cada compañía.
* email: el email de contacto de cada compañia
* country: el país donde esta la sede central de cada compañía. Este sería el único campo que no tendría por qué ser único de esta tabla.
* website: La página oficial de la compañia.

**Tabla credit\_cards:** En esta tabla hay varios datos NOT NULL ya que he considerado que al ser importantes a la hora de hacer compra online era necesario sí o sí que estuvieran.

* id: Es el identificador de la tarjeta de crédito. Este campo es la clave primaria,
* iban: Es el código internacional de cuenta bancaria. Es un código alfanúmerico que identifica una cuenta bancaria en cualquier lugar del mundo.
* pan: es el acrónimo de personal account number y es el número que aparece en la tarjeta de crédito.
* pin: personal identification number es tu clave para entrar en tu cuenta.
* cvv: código valor de verificación. Es un número que está detrás de la tarjeta y que se suele pedir por seguridad en transacciones online.
* track1 y track2 : Son las pistas de banda magnética. Los lectores de losTPV siempre leen las pistas 1 o 2: la información mínima necesaria para realizar una transacción se encuentra en ambas pistas. La pista 1 tiene una mayor densidad de bits, es la única que puede contener caracteres alfanuméricos, y por tanto es la única que puede contener el nombre del portador de la tarjeta.
* Expiring\_date La fecha de caducidad de la tarjeta. Al igual que el cvv se pide a menudo en transacciones online. Para cargarlo lo he hecho como VARCHARr, pero luego lo he editado a DATE

**Tabla product:**

* id : El identificador del producto, es la primary key de la tabla.
* product\_name : El nombre del producto.
* price El precio del producto, es un decimal. Lleva símbolo de dólar, con lo cual puede dar problema al cargar la tabla, pero se soluciona con un código extra como se ve más adelante.
* Colour: color en código hexadecimal del producto
* weight Peso del producto, sirve a la hora de enviar el producto y para tenerlo en cuenta al recibirlo.
* warehouse\_id : Identificador del almacén, sirve para saber dónde está cada producto (qué almacén, qué zona del almacén, qué estante…).

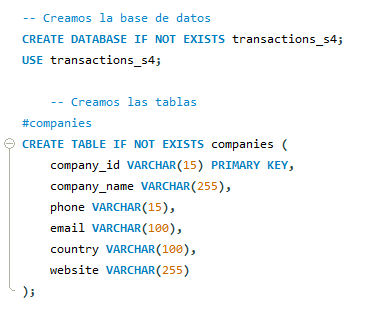
**Tablas users de Canada, Reino Unido y Estados Unidos:** todas tienen los mismos campos, así que las he unido todas en una sola tabla, para evitar repetición de código y simplificar las consultas.

* id: Identificador del usuario, clave primaria.
* Name: nombre del usuario
* Surname: apellido del usuario
* Phone: número de contacto del usuario
* Email: Correo electronico de contactodel usuario.
* birth\_date : Fecha de nacimiento del usuario
* country: país de residencia del usuario
* city: Ciudad de residencia del usuario
* postal\_code: Codigo postal del lugar de residencia del usuario
* address : dirección del lugar de residencia del usuario

### Creación de la base de datos e importación de los datos de los CSV

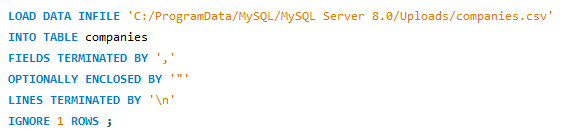
Primero creamos la base de datos, para evitar problemas con bases de datos previas, le ponemos de nombre **transactions\_s4**

#### Database y tabla companies

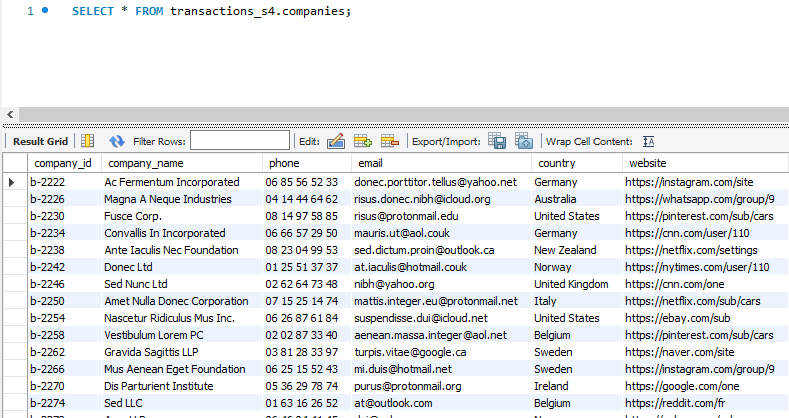


Creamos la tabla **companies** y la cargamos.

Para cargar los datos he buscado la ruta a través del código sql “SHOW VARIABLES LIKE 'secure\_file\_priv';”, este te da la ruta exacta y te permite entrar en carpetas, aunque estén ocultas, pudiendo copiar ahí los archivos. Como estos dtos no tenian todos los campos de varchar entre comillas, he tenido que añadir “OPTIONALLY ENCLOSED BY ‘”’.



Luego comprobamos que se han cargado los datos.

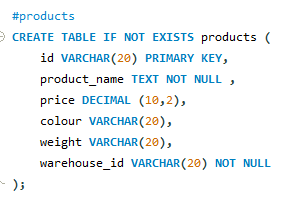


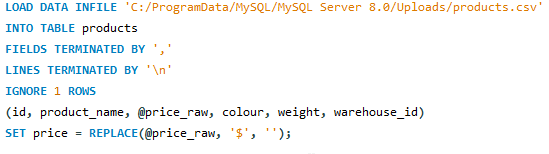
#### Tabla products

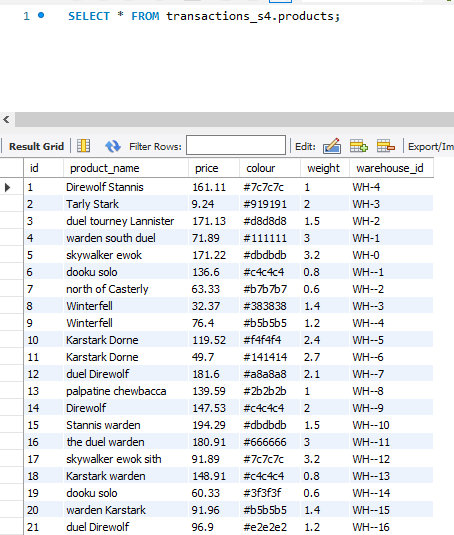
El precio estaba en dólares, así que no se cargaba correctamente, por lo que reemplacé el símbolo de dólar por un espacio usando el código:

*(id, product\_name, @price\_raw, colour, weight, warehouse\_id)*

*SET price = REPLACE(@price\_raw, '$', '');.*

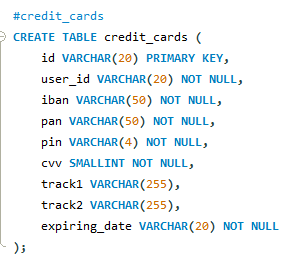


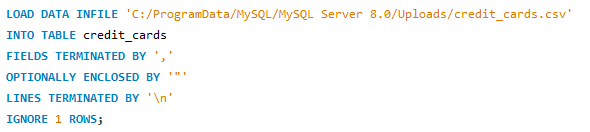


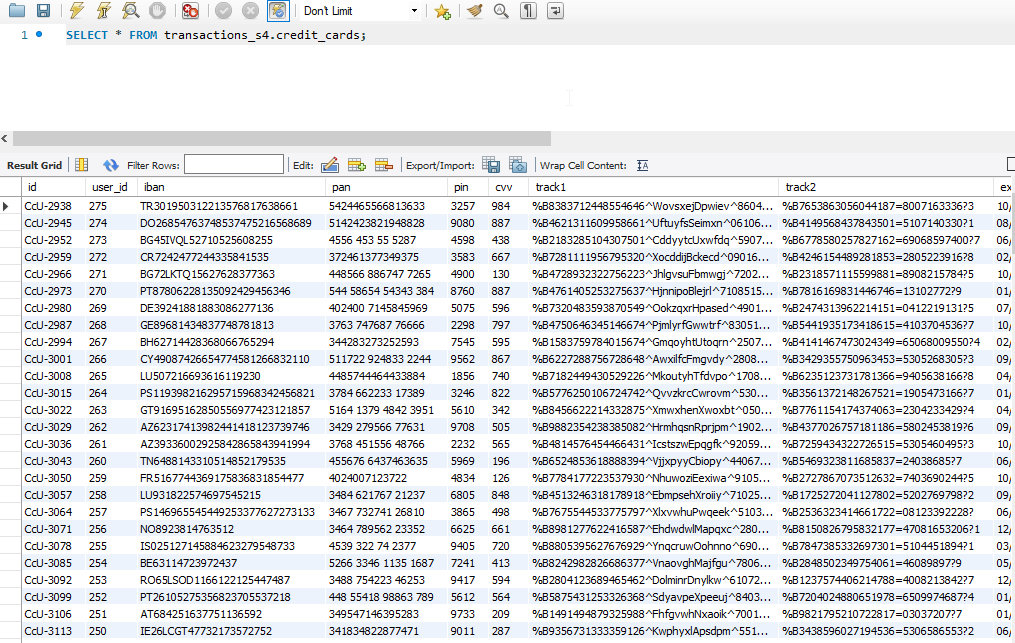


#### Tabla credit\_cards

Aquí el reto era cargar el campo expiring\_date debido al estilo de las fechas, por lo que se ha cargado como VARCHAR. También estaba el error de las comillas que aparecen y desaparecen en algunos registros, pero como vimos anteriormente, se arregla con *OPTIONALLY ENCLOSED BY '"'*

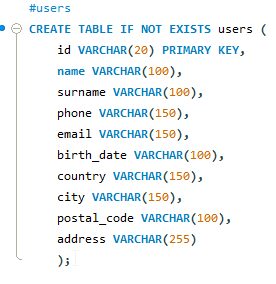


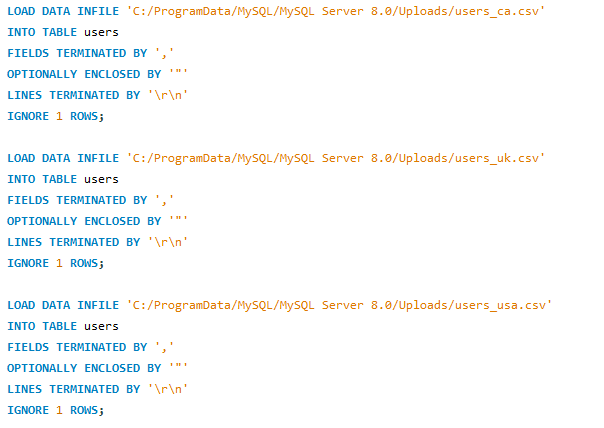


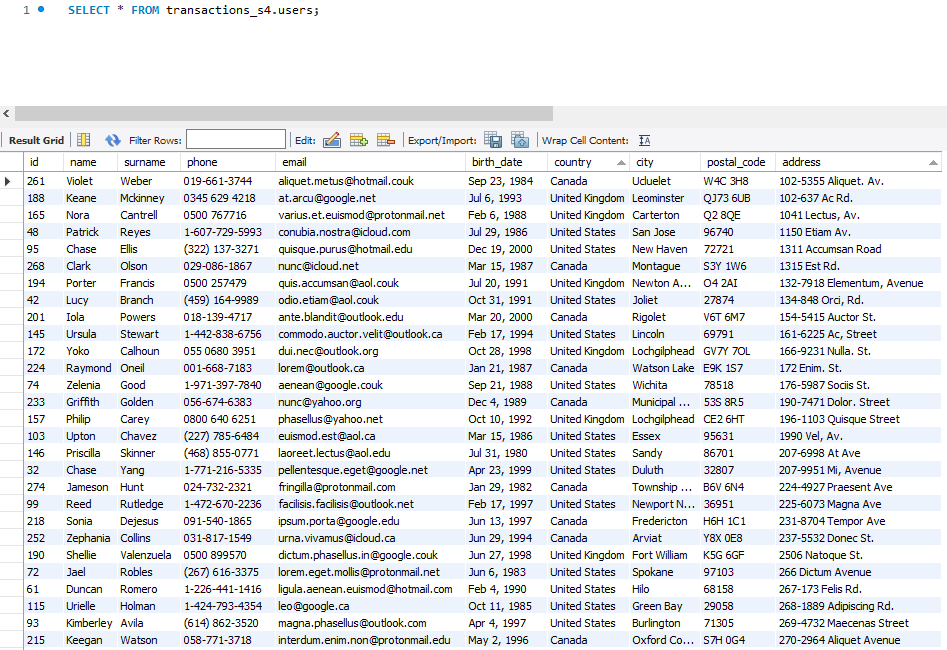


#### Tabla users:

He unido las tres tablas de users en una para evitar código repetido. En este caso el problema de carga era el salto de línea. Según el manual depende de qué documentos y cómo hubieran sido creados hacia falta poner el \r <http://download.nust.na/pub6/mysql/doc/refman/5.0/es/load-data.html#:~:text=Los%20valores%20FIELDS%20TERMINATED%20BY,BY%20'%5Cr%5Cn'%20>. “***Nota****: Si ha generado el fichero de texto en un sistema Windows , puede tener que usar****LINES TERMINATED BY '\r\n'****para leer correctamente el fichero, ya que los programas de Windows típicamente usan dos caracteres como terminadores de línea . Algunos programas como****WordPad****, pueden usar****\r****como terminador de línea al escribir ficheros. Para leer tales ficheros, use****LINES TERMINATED BY '\r'****.”*



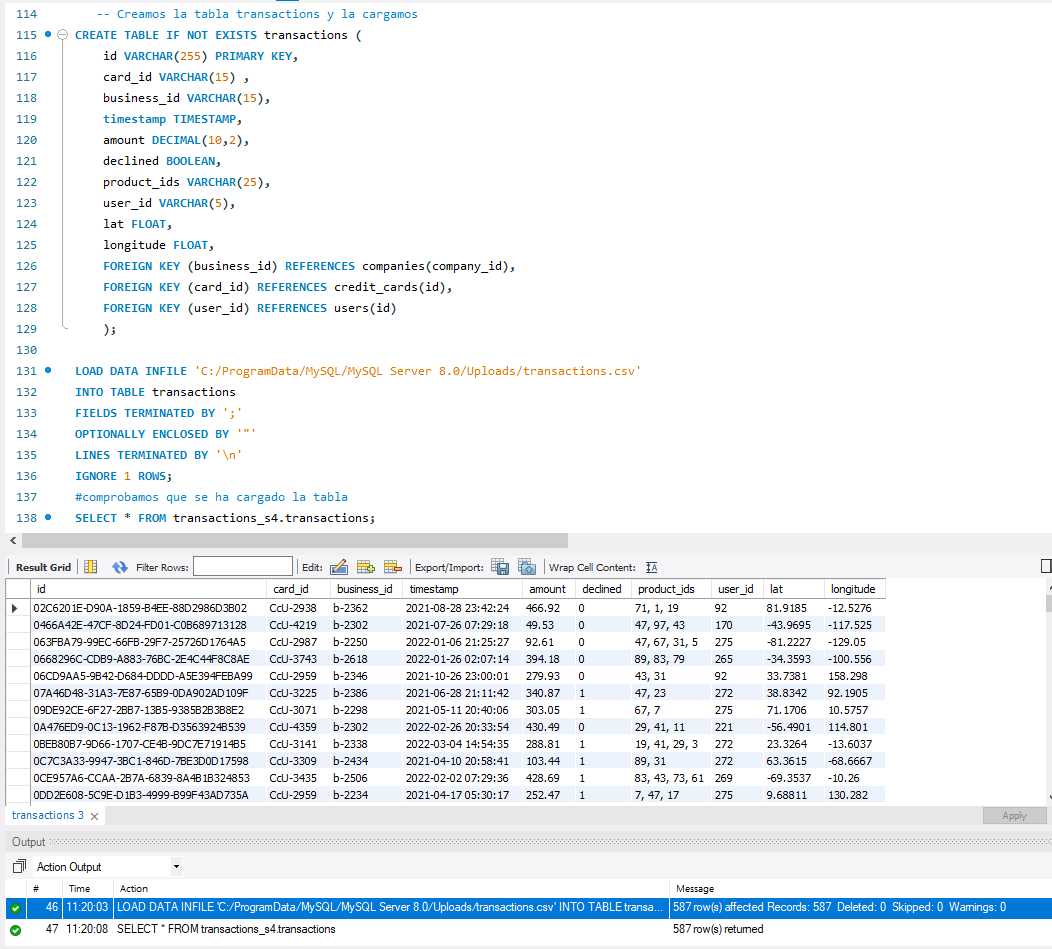




#### Tabla transactions

Repetimos el proceso con la tabla **transactions.** El reto con esta tabla fueron las foreign keys, ya que no deja hacer una relación de muchos a muchos con products\_id.

En la parte de carga, además del problema con las comillas, se nos añadía el de que los campos no estaban separados por ‘,’ sino por ‘;’ .

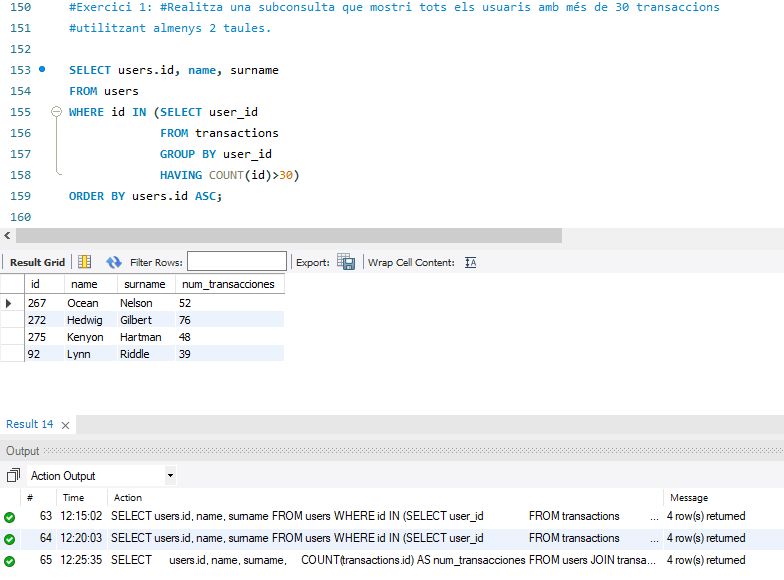


### Ejercicio 1: Realiza una subconsulta que muestre a todos los usuarios con más de 30 transacciones utilizando al menos 2 tablas.

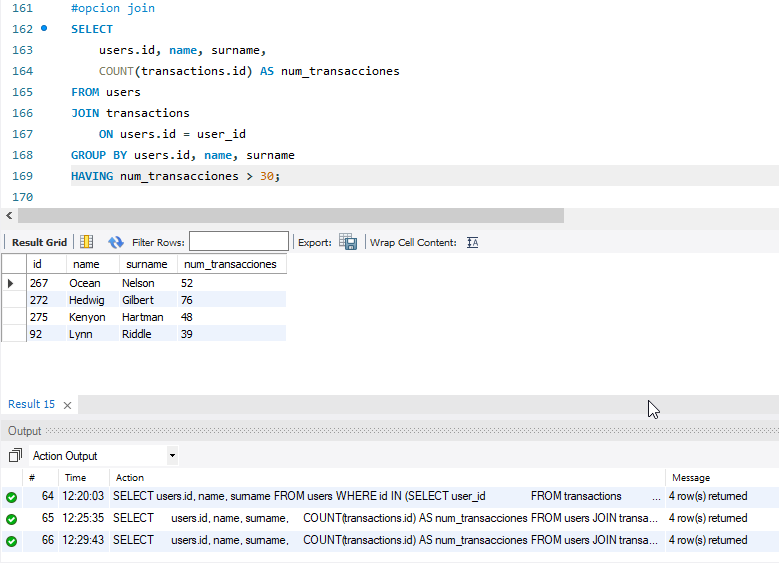
Para este ejercicio he pedido que me busque el nombre y apellidos de los usuarios y cuente la cantidad de transacciones realizadas, declinadas o no.

Nos aparecen solo 4 usuarios con más de 30 transacciones.

En un caso real debería preguntar al cliente si quería que fueran más de 30 o a partir de 30, aunque he comprobado que en este caso concreto no habría ninguna diferencia.



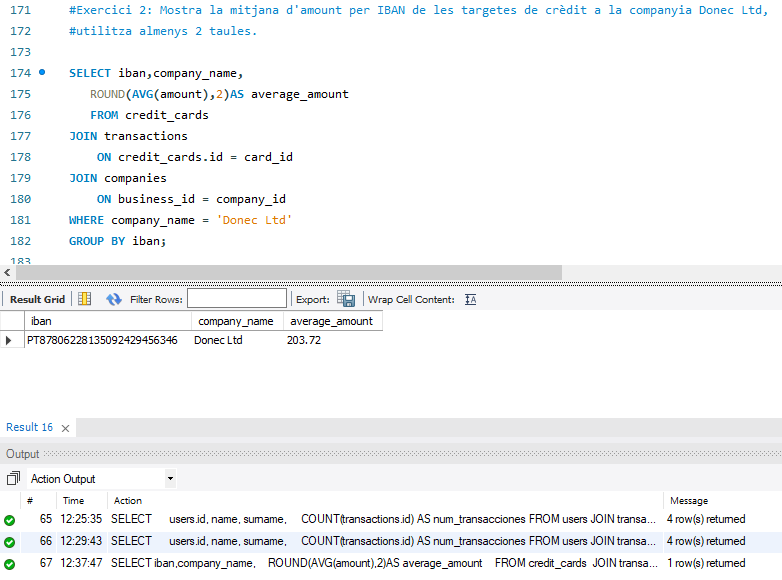
Otra manera de hacerlo sería con Join:



Ejercicio 2 Muestra la media de amount por IBAN de las tarjetas de crédito a la compañía Donec Ltd., utiliza por lo menos 2 tablas.

Buscamos la media del pago de la tarjeta de crédito y luego hacemos un doble join de credit cards con companies y transactions, después buscamos la compañía Donec y por último lo agrupamos por el Iban de la tarjeta de crédito.

La media resultante de Donec Ltd es de 203.72$



## Nivel 2

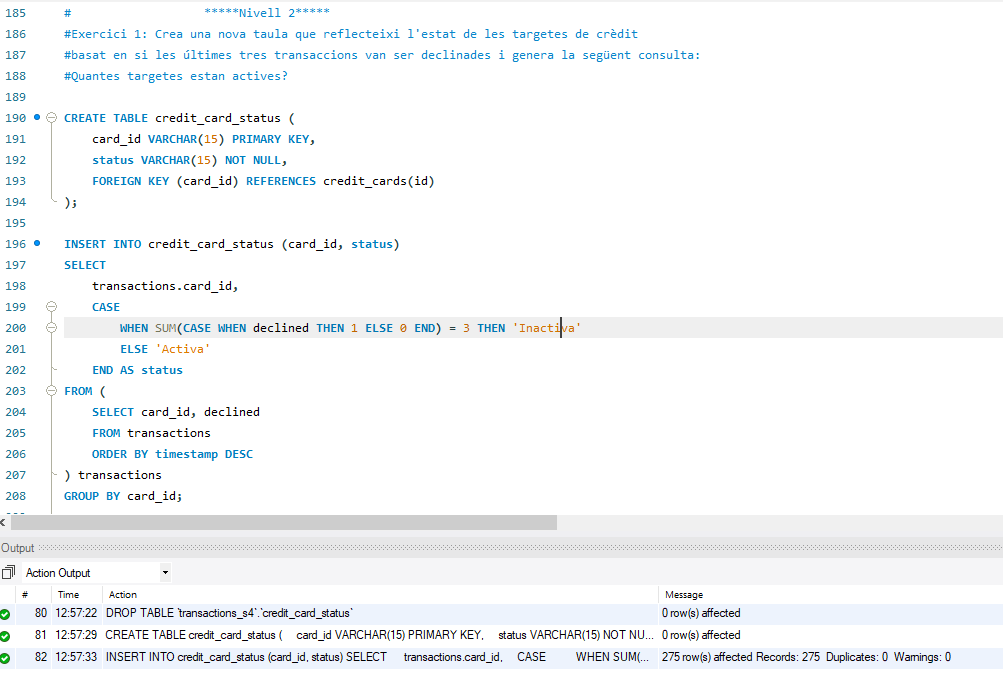
### Ejercicio 1: Crea una nueva tabla que refleje el estado de las tarjetas de crédito basado en si las últimas tres transacciones fueron declinadas y genera la siguiente consulta: ¿Cuantas tarjetas están activas?

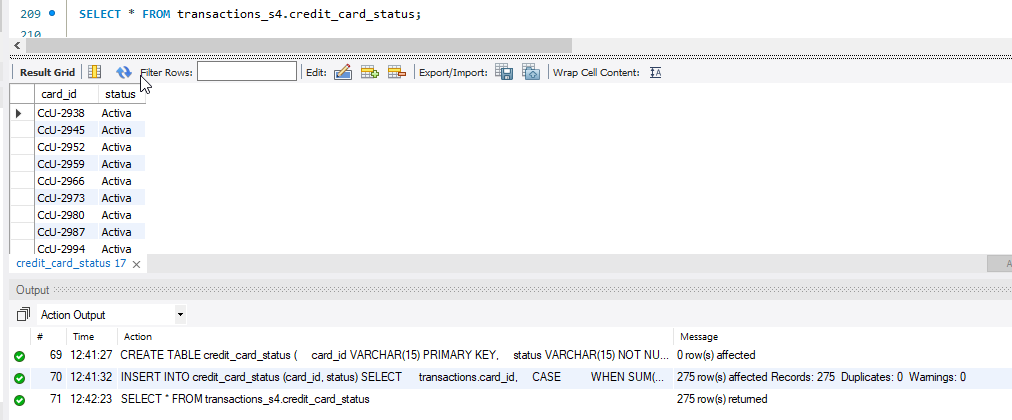
Creamos la tabla credit\_card\_status y le introducimos datos de las tablas credit\_card y transactions, de la primera el campo de id y de la segunda obtenemos su estatus, si es activa o inactiva.

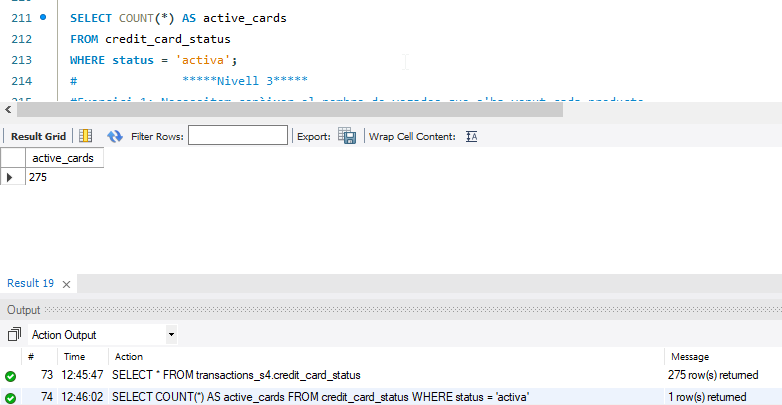
Para determinar el status usamos una subconsulta (WITH card\_transactions) para identificar las últimas tres transacciones de cada tarjeta ordenadas por timestamp. Si la transacción ha sido declinada menos de 3 veces en las últimas 3 transacciones se convierte en activa.

Después contamos cuantas tarjetas están en estado activo.

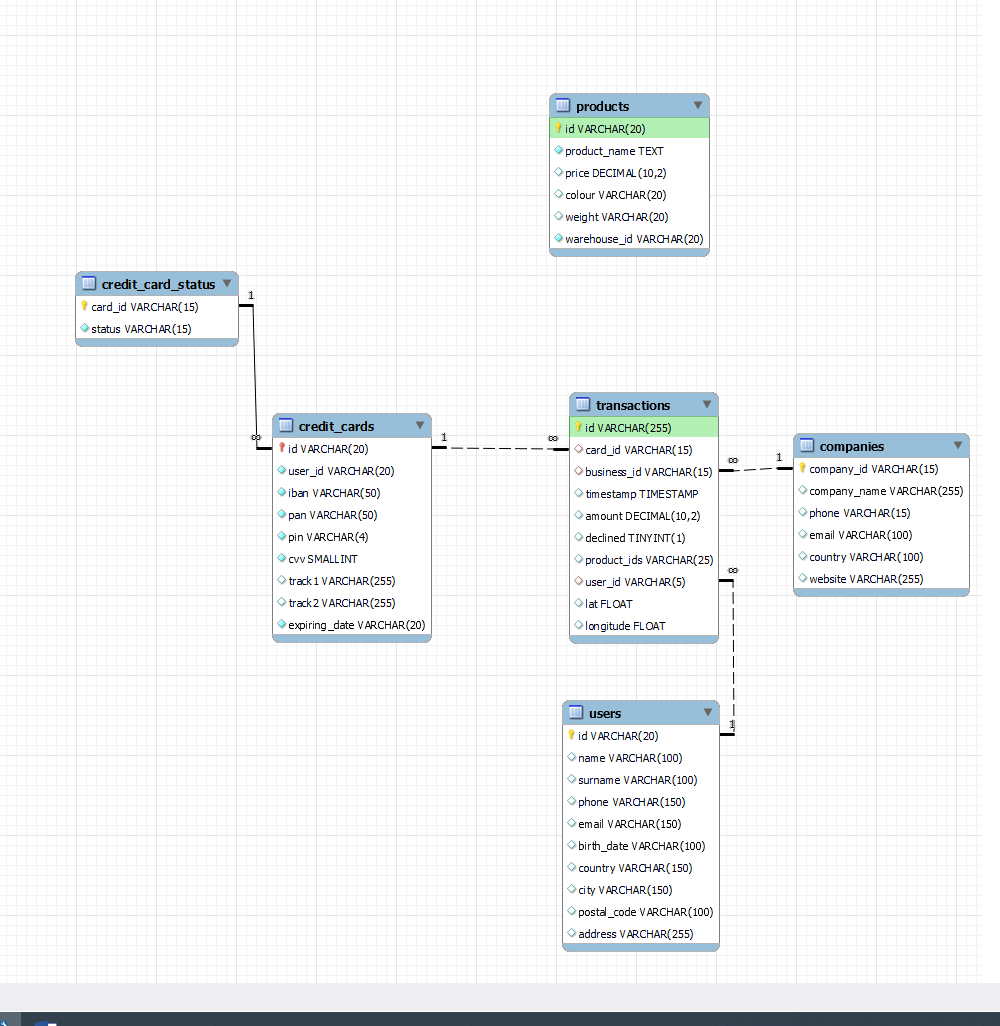
De 275 resultados no aparece ninguna tarjeta inactiva.







El modelo relacional nos quedaría así, convirtiéndose el modelo estrella en uno de copo de nieve. La tabla credit\_card status queda unida a credit\_cards mediante la clave foránea de card\_id.



## Nivel 3

Crea una tabla con la que podamos unir los datos del nuevo archivo products.csv con la base de datos creada, teniendo en cuenta que desde transaction tienes product\_ids. Genera la siguiente consulta:

Creamos la tabla bridge\_products con los campos transactions\_id y products\_id conectando así la tabla products con transactions.

Luego insertamos en la tabla bridge\_products los id de transacciones y productos para todas las transacciones, eliminando los espacios en blanco del campo products\_ids mediante la función FIND\_IN\_SET.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Comprobamos que los datos se han cargado correctamente en la tabla.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

### Ejercicio 1: Necesitamos conocer el número de veces que se ha vendido cada producto.

Como se nos pide el número de productos vendidos, usamos el declined para eliminar aquellas transacciones que no se han llevado finalmente a cabo.

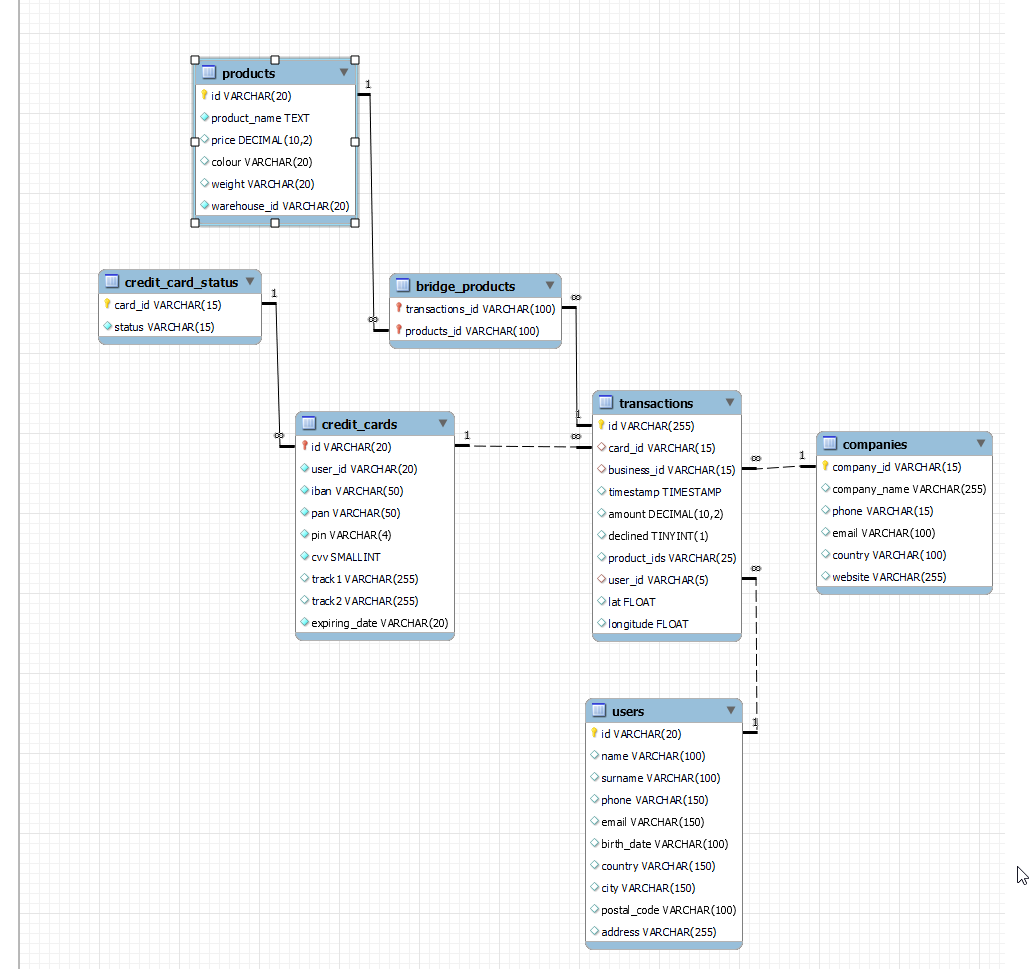
Nos da de resultado 26 columnas con el numero de ventas de cada producto-siendo el más vendido Riverlands North.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente



Por fin podemos unir products a transactions mediante la tabla bridge\_products. Esta se une a products mediante la clave foránea de products\_id.



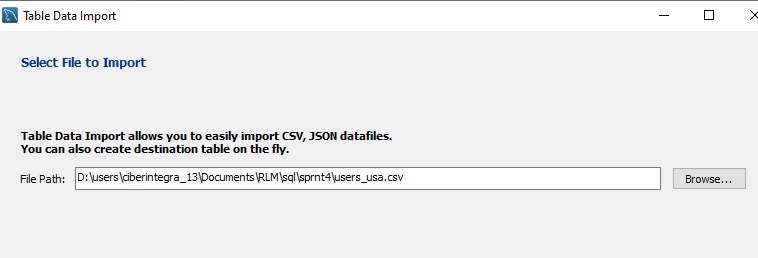
## Anexo: Los problemas de carga las tablas

### 

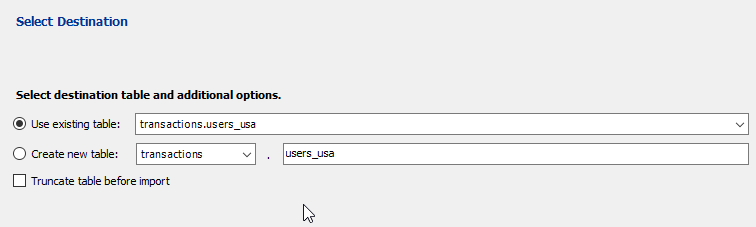
### Originalmente iba a cargarlo así, però me aparecía el error *"Error Code: 1290 - The MySQL server is running with the --secure-file-priv option so it cannot execute this statement."* Y las maneras de corregirlo son difíciles en el ordenador del cibernàrium, ya que tengo que entrar en zonas seguras y no tengo los permisos, así que voy a ir avanzando, usando el import wizard. Luego desde casa lo volveré a intentar de otra manera.

### 

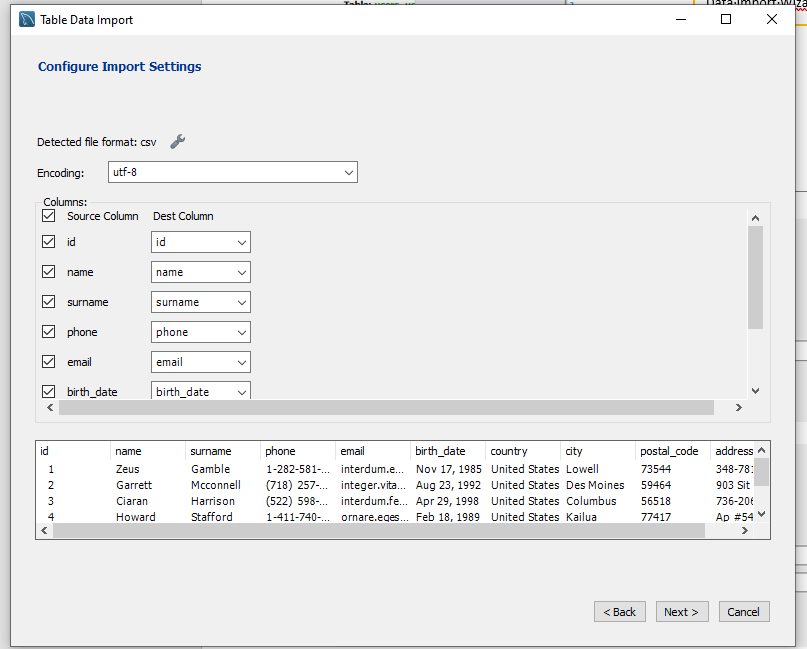
Con el botón derecho sobre la table elegida aparece una ventana de la que elegimos Table Data Import Wizard.

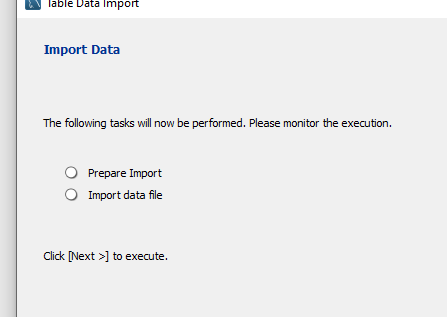
Nos abre el Wizard y le decimos de donde sacamos el archivo.****

Le damos a Next y elegimos destino



Le volvemos a dar a Next





Como corregirlo desde casa:

Hay dos maneras, una abriendo el my.ini y dejando las comillas vacías (dónde esta resaltado).Una captura de pantalla de una computadora

Descripción generada automáticamente

La segunda, llevar los archivos csv a este directorio C:/ProgramData/MySQL/MySQL Server 8.0/Uploads , que es la que voy a hacer ahora.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Para cualquiera de ellas hay que volver a iniciar el mysql.