

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

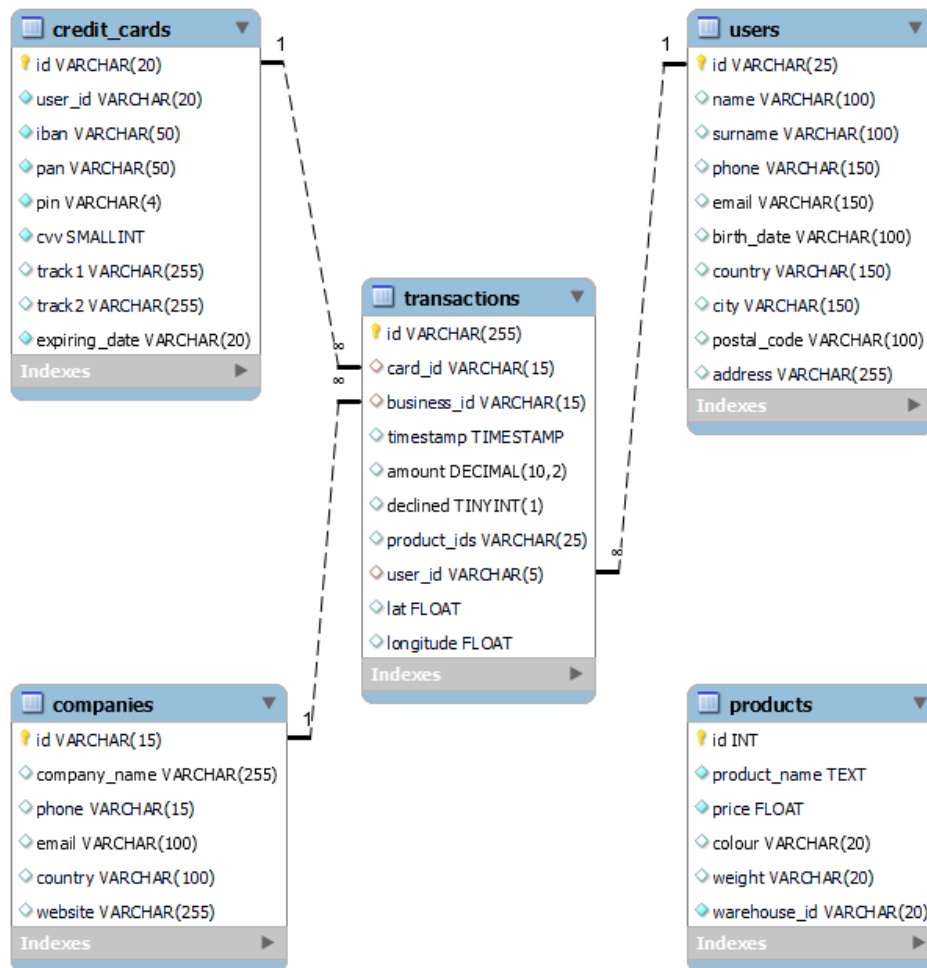
TABLA DE CONTENIDO

Nivel 1.....	2
Esquema en estrella	2
Creación de la base de datos e importación de los datos de los CSV	5
Ejercicio 1: Realiza una subconsulta que muestre a todos los usuarios con más de 30 transacciones utilizando al menos 2 tablas.....	11
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.	11
Nivel 2.....	12
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: ¿Cuántas tarjetas están activas?	12
Nivel 3.....	13
Ejercicio 1: Necesitamos conocer el número de veces que se ha vendido cada producto. 14	
Anexo: Los problemas de carga las tablas	15

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 **Transaction**, 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, cuanto dinero se ha ganado...

Esta 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.

transactions - Table	
Table Name: transactions	
Foreign Key Name	Referenced Table
card_id	transactions_s4, credit_cards
business_id	transactions_s4, companies
user_id	transactions_s4, users

Tabla Transaction:

- 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 uniríamos 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:

- 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 compañía
- 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 compañía.

Tabla Credit_cards: En esta tabla hay varios datos NOT NULL ya que son importantes a la hora de hacer compra online

- 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 alfanumérico 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.

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 float y lleva símbolo de dólar, con lo cual puede dar problema al cargar la tabla.
- 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 queries.

- id: Identificador del usuario, clave primaria.
- Name: nombre del usuario
- Surname: apellido del usuario
- Phone: número de contacto del usuario
- Email: Correo electrónico de contacto del usuario.
- birth_date : Fecha de nacimiento del usuario
- country: país de residencia del usuario
- city: Ciudad de residencia del usuario
- postal_code: Código 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 base de datos
• CREATE DATABASE IF NOT EXISTS transactions_s4;
• USE transactions_s4;

-- Creamos las tablas
#companies
• CREATE TABLE IF NOT EXISTS companies (
    id VARCHAR(15) PRIMARY KEY,
    company_name VARCHAR(255),
    phone VARCHAR(15),
    email VARCHAR(100),
    country VARCHAR(100),
    website VARCHAR(255)
);
```

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 tenían todos los campos de varchar entre comillas, he tenido que añadir “OPTIONALLY ENCLOSED BY ‘”’”.

```
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/companies.csv'
INTO TABLE companies
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS ;
```

Luego comprobamos que se han cargado los datos.

```
1 • SELECT * FROM transactions_s4.companies;
```

id	company_name	phone	email	country	website
b-2222	Ac Fermentum Incorporated	06 85 56 52 33	donec.porttitor.tellus@yahoo.net	Germany	https://instagram.com/site
b-2226	Magna A Neque Industries	04 14 44 64 62	risus.donec.nibh@icloud.org	Australia	https://whatsapp.com/group/9
b-2230	Fusce Corp.	08 14 97 58 85	risus@protonmail.edu	United States	https://pinterest.com/sub/cars
b-2234	Convallis In Incorporated	06 66 57 29 50	mauris.ut@aol.couk	Germany	https://cnn.com/user/110
b-2238	Ante Iaculis Nec Foundation	08 23 04 99 53	sed.dictum.proin@outlook.ca	New Zealand	https://netflix.com/settings
b-2242	Donec Ltd	01 25 51 37 37	at.iaculis@hotmail.couk	Norway	https://nytimes.com/user/110
b-2246	Sed Nunc Ltd	02 62 64 73 48	nibh@yahoo.org	United Kingdom	https://cnn.com/one
b-2250	Amet Nulla Donec Corporation	07 15 25 14 74	mattis.integer.eu@protonmail.net	Italy	https://netflix.com/sub/cars
b-2254	Nascetur Ridiculus Mus Inc.	06 26 87 61 84	suspendisse.dui@icloud.net	United States	https://ebay.com/sub
b-2258	Vestibulum Lorem PC	02 02 87 33 40	aenean.massa.integer@aol.net	Belgium	https://pinterest.com/sub/cars
b-2262	Gravida Sagittis LLP	03 81 28 33 97	turpis.vitae@google.ca	Sweden	https://naver.com/site
b-2266	Mus Aenean Eget Foundation	06 25 15 52 43	mi.duis@hotmail.net	Sweden	https://instagram.com/group/9
b-2270	Dis Parturient Institute	05 36 29 78 74	purus@protonmail.org	Ireland	https://google.com/one
b-2274	Sed LLC	01 63 16 26 52	at@outlook.com	Belgium	https://reddit.com/fr
b-2278	Arcu LLP	06 46 04 41 45	dui@aol.ca	Norway	https://yahoo.com/sub
b-2282	Pretium Neque Corp.	07 77 48 55 28	eleifend.nec.malesuada@proton...	Australia	https://netflix.com/sub
b-2286	Fringilla LLC	08 29 15 93 57	gravida@protonmail.couk	New Zealand	https://reddit.com/user/110
b-2290	Quisque Libero LLC	01 45 48 71 11	sapien.molestie.orci@hotmail.couk	China	https://baidu.com/group/9
b-2294	Auctor Mauris Vel LLP	08 09 28 74 14	nec.tempus@icloud.couk	United States	https://instagram.com/fr
b-2298	Elit Etiam Laoreet Associates	07 69 74 17 45	ultrices@google.couk	Canada	https://yahoo.com/fr
b-2302	Nunc Interdum Incorporated	05 18 15 48 13	non@outlook.com	Germany	https://wikipedia.org/en-us
b-2306	Augue Foundation	06 88 43 15 63	mauris@yahoo.com	Germany	https://baidu.com/sub/cars
b-2310	Non Magna LLC	06 71 73 13 17	nisl.quisque.fringilla@hotmail.ca	United Kingdom	https://whatsapp.com/site

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 que los campos no estaban separados por ';' sino por ','.

```
# transactions
CREATE TABLE IF NOT EXISTS transactions (
  id VARCHAR(255) PRIMARY KEY,
  card_id VARCHAR(15) REFERENCES credit_card(id),
  business_id VARCHAR(15) REFERENCES companies(id),
  timestamp TIMESTAMP,
  amount DECIMAL(10,2),
  declined BOOLEAN,
  product_ids VARCHAR(25),
  user_id VARCHAR(5) REFERENCES users(id),
  lat FLOAT,
  longitude FLOAT
);
```

```
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/transactions.csv'
INTO TABLE transactions
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;
```

```
1 • SELECT * FROM transactions_s4.transactions;
```

Result Grid										
Filter Rows:										
	id	card_id	business_id	timestamp	amount	declined	product_ids	user_id	lat	longitude
▶	02C6201E-D90A-1859-B4EE-88D2986D3B02	CdJ-2938	b-2362	2021-08-28 23:42:24	466.92	0	71, 1, 19	92	81.9185	-12.5276
	0466A42E-47CF-8D24-FD01-C0B689713128	CdJ-4219	b-2302	2021-07-26 07:29:18	49.53	0	47, 97, 43	170	-43.9695	-117.525
	063FBA79-99EC-66FB-29F7-25726D1764A5	CdJ-2987	b-2250	2022-01-06 21:25:27	92.61	0	47, 67, 31, 5	275	-81.2227	-129.05
	0668296C-CDB9-A883-76BC-2E4C44F8C8AE	CdJ-3743	b-2618	2022-01-26 02:07:14	394.18	0	89, 83, 79	265	-34.3593	-100.556
	06CD9AA5-9B42-D684-DDDD-A5E394FEBA99	CdJ-2959	b-2346	2021-10-26 23:00:01	279.93	0	43, 31	92	33.7381	158.298
	07A46D48-31A3-7E87-65B9-0DA902AD109F	CdJ-3225	b-2386	2021-06-28 21:11:42	340.87	1	47, 23	272	38.8342	92.1905
	09DE92CE-6F27-2BB7-13B5-9385B2B388E2	CdJ-3071	b-2298	2021-05-11 20:40:06	303.05	1	67, 7	275	71.1706	10.5757
	0A476ED9-0C13-1962-F87B-D3563924B539	CdJ-4359	b-2302	2022-02-26 20:33:54	430.49	0	29, 41, 11	221	-56.4901	114.801
	0BEB80B7-9D66-1707-CE4B-9DC7E71914B5	CdJ-3141	b-2338	2022-03-04 14:54:35	288.81	1	19, 41, 29, 3	272	23.3264	-13.6037
	0C7C3A33-9947-3BC1-846D-7BE3D0D17598	CdJ-3309	b-2434	2021-04-10 20:58:41	103.44	1	89, 31	272	63.3615	-68.6667
	0CE957A6-CCAA-2B7A-6839-8A4B1B324853	CdJ-3435	b-2506	2022-02-02 07:29:36	428.69	1	83, 43, 73, 61	269	-69.3537	-10.26
	0DD2E608-5C9E-D1B3-4999-B99F43AD735A	CdJ-2959	b-2234	2021-04-17 05:30:17	252.47	1	7, 47, 17	275	9.68811	130.282

Tabla products

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

```
#products
```

```
CREATE TABLE IF NOT EXISTS products (  
  id VARCHAR(20) PRIMARY KEY,  
  product_name TEXT NOT NULL ,  
  price FLOAT NOT NULL ,  
  colour VARCHAR(20),  
  weight VARCHAR(20),  
  warehouse_id VARCHAR(20) NOT NULL  
);
```

```
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/products.csv'  
INTO TABLE products  
FIELDS TERMINATED BY ','  
LINES TERMINATED BY '\n'  
IGNORE 1 ROWS  
(id, product_name, @price_raw, colour, weight, warehouse_id)  
SET price = REPLACE(@price_raw, '$', '');
```

```
1 • SELECT * FROM transactions_s4.products;
```

	id	product_name	price	colour	weight	warehouse_id
▶	1	Direwolf Stannis	161.11	#7c7c7c	1	WH-4
	2	Tarly Stark	9.24	#919191	2	WH-3
	3	duel tourney Lannister	171.13	#d8d8d8	1.5	WH-2
	4	warden south duel	71.89	#111111	3	WH-1
	5	skywalker ewok	171.22	#bdbdbd	3.2	WH-0
	6	dooku solo	136.6	#c4c4c4	0.8	WH--1
	7	north of Casterly	63.33	#b7b7b7	0.6	WH--2
	8	Winterfell	32.37	#383838	1.4	WH--3
	9	Winterfell	76.4	#b5b5b5	1.2	WH--4
	10	Karstark Dorne	119.52	#f4f4f4	2.4	WH--5
	11	Karstark Dorne	49.7	#141414	2.7	WH--6
	12	duel Direwolf	181.6	#a8a8a8	2.1	WH--7
	13	palpatine chewbacca	139.59	#2b2b2b	1	WH--8
	14	Direwolf	147.53	#c4c4c4	2	WH--9
	15	Stannis warden	194.29	#bdbdbd	1.5	WH--10
	16	the duel warden	180.91	#666666	3	WH--11
	17	skywalker ewok sith	91.89	#7c7c7c	3.2	WH--12
	18	Karstark warden	148.91	#c4c4c4	0.8	WH--13
	19	dooku solo	60.33	#3f3f3f	0.6	WH--14
	20	warden Karstark	91.96	#b5b5b5	1.4	WH--15
	21	duel Direwolf	96.9	#e2e2e2	1.2	WH--16

Tabla credit_cards

#credit_cards

```
CREATE TABLE credit_cards (
    id VARCHAR(20) PRIMARY KEY,
    user_id VARCHAR(20) NOT NULL,
    iban VARCHAR(50) NOT NULL,
    pan VARCHAR(50) NOT NULL,
    pin VARCHAR(4) NOT NULL,
    cvv SMALLINT NOT NULL,
    track1 VARCHAR(255),
    track2 VARCHAR(255),
    expiring_date VARCHAR(20) NOT NULL
);
```

```
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/credit_cards.csv'
INTO TABLE credit_cards
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;
```

1 • SELECT * FROM transactions_s4.credit_cards;

id	user_id	iban	pan	pin	cvv	track1	track2	ex
CcU-2938	275	TR301950312213576817638661	5424465566813633	3257	984	%B8383712448554646^WovsxejDpwiev^8604...	%B7653863056044187=800716333673	10
CcU-2945	274	DO26854763748537475216568689	5142423821948828	9080	887	%B4621311609958661^UftuyfsSeimxn^06106...	%B4149568437843501=510714033071	08
CcU-2952	273	BG45IVQL52710525608255	4556 453 55 5287	4598	438	%B2183285104307501^CddytytLxwfdq^5907...	%B6778580257827162=6906859740077	06
CcU-2959	272	CR7242477244335841535	372461377349375	3583	667	%B7281111956795320^XocddjBckecd^09016...	%B4246154489281853=280522391678	02
CcU-2966	271	BG72LKTQ15627628377363	448566 886747 7265	4900	130	%B4728932322756223^JhgvsuFbmwgj^7202...	%B2318571115599881=890821578495	10
CcU-2973	270	PT87806228135092429456346	544 58654 54343 384	8760	887	%B4761405253275637^Hjnnipoblejr^7108515...	%B7816169831446746=1310277279	01
CcU-2980	269	DE39241881883086277136	402400 7145845969	5075	596	%B7320483593870549^OokzqxHpoased^4901...	%B2474313962214151=041221913125	07
CcU-2987	268	GE89681434837748781813	3763 747687 76666	2298	797	%B4750646345146674^Pjmlrfgwvtrf^83051...	%B5441935173418615=410370453677	10
CcU-2994	267	BH62714428368066765294	344283273252593	7545	595	%B1583759784015674^GmqoyhtUtoqrn^2507...	%B4141467473024349=6506800955074	02
CcU-3001	266	CY49087426654774581266832110	511722 924833 2244	9562	867	%B6227288756728648^AwxlfcFmgvdy^2808...	%B3429355750963453=530526830573	09
CcU-3008	265	LU507216693616119230	4485744464433884	1856	740	%B7182449430529226^MkoutyhTfdvpo^1708...	%B6235123731781366=940563816678	04
CcU-3015	264	PS119398216295715968342456821	3784 662233 17389	3246	822	%B5776250106724742^QvvzkrCwrovrm^530...	%B3561372148267521=190547316677	01
CcU-3022	263	GT91695162850556977423121857	5164 1379 4842 3951	5610	342	%B8456622214332875^XmwvhenXwobxt^050...	%B7761154174374063=230423342974	04
CcU-3029	262	AZ62317413982441418123739746	3429 279566 77631	9708	505	%B9882354238385082^HmhqsnRprjam^1902...	%B4377026757181186=580245381976	09
CcU-3036	261	AZ39336002925842865843941994	3768 451556 48766	2232	565	%B4814576454466431^IcstszwEpqgfk^92059...	%B7259434322726515=530546049573	10
CcU-3043	260	TN6488143310514852179535	455676 6437463635	5969	196	%B6524853618888394^VjxpyyBiopy^44067...	%B5469323811685837=2403868577	06
CcU-3050	259	FR5167744369175836831854477	4024007123722	4834	126	%B7784177223537930^NhuwoziEexiwa^9105...	%B2727867073512632=740369024475	10
CcU-3057	258	LU931822574697545215	3484 621767 21237	6805	848	%B4513246318178918^EbmipsehVroiiy^71025...	%B1725272041127802=520276979872	09
CcU-3064	257	PS146965545449253377627273133	3467 732741 26810	3865	498	%B7675544533775797^XbvwhwPvqeeq^5103...	%B2536323414661722=081233922287	06
CcU-3071	256	NO8923814763512	3464 789562 23352	6625	661	%B8981277622416587^EhdwdwMapqxc^280...	%B8150826795832177=4708165320671	12
CcU-3078	255	ISO25127145884623279548733	4539 322 74 2377	9405	720	%B8805395627676929^YnqruwOohnno^690...	%B7847385332697301=510445189471	03
CcU-3085	254	BE631147239724237	5266 3346 1135 1687	7241	413	%B8242982826686377^VnaovghMajfgu^7806...	%B2848502349754061=4608989799	05
CcU-3092	253	R06LSOD1166122125447487	3488 754223 46253	9417	594	%B2804123689465462^DolmrvDnylkw^61072...	%B1237574406214788=400821384277	12
CcU-3099	252	PT26105275356823705537218	448 55418 98863 789	5612	564	%B5875431253326368^SdyavvpeXeeuj^8403...	%B7204024880651978=650997468774	01
CcU-3106	251	AT684251637751136592	349547146395283	9733	209	%B1491494879325988^FhfgvwhNbaok^7001...	%B9821795210722817=0303720777	01
CcU-3113	250	IE26LCGT47732173572752	341834822877471	9011	287	%B9356731333359126^KwphxylApsdpm^551...	%B3438596027194536=530658655372	06

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'**.”

```
#users
CREATE TABLE IF NOT EXISTS users (
  id VARCHAR(20) PRIMARY KEY,
  name VARCHAR(100),
  surname VARCHAR(100),
  phone VARCHAR(150),
  email VARCHAR(150),
  birth_date VARCHAR(100),
  country VARCHAR(150),
  city VARCHAR(150),
  postal_code VARCHAR(100),
  address VARCHAR(255)
);

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_ca.csv'
INTO TABLE users
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\r\n'
IGNORE 1 ROWS;

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_uk.csv'
INTO TABLE users
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\r\n'
IGNORE 1 ROWS;

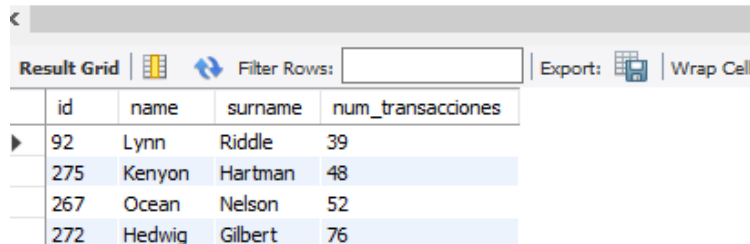
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_usa.csv'
INTO TABLE users
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\r\n'
IGNORE 1 ROWS;
```

```
1 • SELECT * FROM transactions_s4.users;
```

Result Grid										
Filter Rows:										
Edit: Export/Import: Wrap Cell Content:										
	id	name	surname	phone	email	birth_date	country	city	postal_code	address
▶	261	Violet	Weber	019-661-3744	aliquet.metus@hotmail.couk	Sep 23, 1984	Canada	Ucluelet	W4C 3H8	102-5355 Aliquet. Av.
	188	Keane	Mckinney	0345 629 4218	at.arcu@google.net	Jul 6, 1993	United Kingdom	Leominster	QJ73 6UB	102-637 Ac Rd.
	165	Nora	Cantrell	0500 767716	varius.et.euismod@protonmail.net	Feb 6, 1988	United Kingdom	Carterton	Q2 8QE	1041 Lectus, Av.
	48	Patrick	Reyes	1-607-729-5993	conubia.nostra@icloud.com	Jul 29, 1986	United States	San Jose	96740	1150 Etiam Av.
	95	Chase	Ellis	(322) 137-3271	quisque.purus@hotmail.edu	Dec 19, 2000	United States	New Haven	72721	1311 Accumsan Road
	268	Clark	Olson	029-086-1867	nunc@icloud.net	Mar 15, 1987	Canada	Montague	S3Y 1W6	1315 Est Rd.
	194	Porter	Francis	0500 257479	quis.accumsan@aol.couk	Jul 20, 1991	United Kingdom	Newton A...	O4 2AI	132-7918 Elementum, Avenue
	42	Lucy	Branch	(459) 164-9989	odio.etiam@aol.couk	Oct 31, 1991	United States	Joliet	27874	134-848 Orci, Rd.
	201	Iola	Powers	018-139-4717	ante.blandit@outlook.edu	Mar 20, 2000	Canada	Rigolet	V6T 6M7	154-5415 Auctor St.
	145	Ursula	Stewart	1-442-838-6756	commodo.auctor.velit@outlook.ca	Feb 17, 1994	United States	Lincoln	69791	161-6225 Ac, Street
	172	Yoko	Calhoun	055 0680 3951	dui.nec@outlook.org	Oct 28, 1998	United Kingdom	Lochgilphead	GV7Y 7OL	166-9231 Nulla. St.
	224	Raymond	Oneil	001-668-7183	lorem@outlook.ca	Jan 21, 1987	Canada	Watson Lake	E9K 1S7	172 Enim. St.
	74	Zelenia	Good	1-971-397-7840	aenean@google.couk	Sep 21, 1988	United States	Wichita	78518	176-5987 Sociis St.
	233	Griffith	Golden	056-674-6383	nunc@yahoo.org	Dec 4, 1989	Canada	Municipal ...	53S 8R5	190-7471 Dolor. Street
	157	Philip	Carey	0800 640 6251	phasellus@yahoo.net	Oct 10, 1992	United Kingdom	Lochgilphead	CE2 6HT	196-1103 Quisque Street
	103	Upton	Chavez	(227) 785-6484	euismod.est@aol.ca	Mar 15, 1986	United States	Essex	95631	1990 Vel, Av.
	146	Priscilla	Skinner	(468) 855-0771	laoreet.lectus@aol.edu	Jul 31, 1980	United States	Sandy	86701	207-6998 At Ave
	32	Chase	Yang	1-771-216-5335	pellentesque.eget@google.net	Apr 23, 1999	United States	Duluth	32807	207-9951 Mi, Avenue
	274	Jameson	Hunt	024-732-2321	fringilla@protonmail.com	Jan 29, 1982	Canada	Township ...	B6V 6N4	224-4927 Praesent Ave
	99	Reed	Rutledge	1-472-670-2236	facilisis.facilisis@outlook.net	Feb 17, 1997	United States	Newport N...	36951	225-6073 Magna Ave
	218	Sonia	Dejesus	091-540-1865	ipsum.porta@google.edu	Jun 13, 1997	Canada	Fredericton	H6H 1C1	231-8704 Tempor Ave
	252	Zephania	Collins	031-817-1549	urna.vivamus@icloud.ca	Jun 29, 1994	Canada	Arviat	Y8X 0E8	237-5532 Donec St.
	190	Shelle	Valenzuela	0500 899570	dictum.phasellus.in@google.couk	Jun 27, 1998	United Kingdom	Fort William	K5G 6GF	2506 Natoque St.
	72	Jael	Robles	(267) 616-3375	lorem.eget.mollis@protonmail.net	Jun 6, 1983	United States	Spokane	97103	266 Dictum Avenue
	61	Duncan	Romero	1-226-441-1416	ligula.aenean.euismod@hotmail.com	Feb 4, 1990	United States	Hilo	68158	267-173 Felis Rd.
	115	Urielle	Holman	1-424-793-4354	leo@google.ca	Oct 11, 1985	United States	Green Bay	29058	268-1889 Adipiscing Rd.
	93	Kimberley	Avila	(614) 862-3520	magna.phasellus@outlook.com	Apr 4, 1997	United States	Burlington	71305	269-4732 Maecenas Street
	215	Keegan	Watson	058-771-3718	interdum.enim.non@protonmail.edu	May 2, 1996	Canada	Oxford Co...	S7H 0G4	270-2964 Aliquet Avenue

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

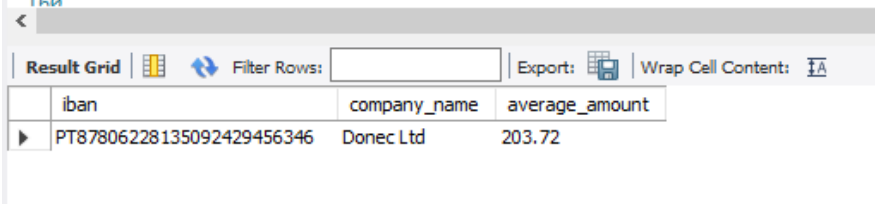
```
139 • SELECT
140     users.id, users.name, users.surname,
141     COUNT(transactions.id) AS num_transacciones
142 FROM users
143 JOIN transactions
144     ON users.id = transactions.user_id
145 GROUP BY users.id, users.name, users.surname
146 HAVING num_transacciones > 30;
147
```



	id	name	surname	num_transacciones
▶	92	Lynn	Riddle	39
	275	Kenyon	Hartman	48
	267	Ocean	Nelson	52
	272	Hedwig	Gilbert	76

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.

```
151 • SELECT credit_cards.iban, companies.company_name,
152     ROUND(AVG(amount), 2) AS average_amount
153 FROM credit_cards
154 JOIN transactions
155     ON credit_cards.id = transactions.card_id
156 JOIN companies
157     ON transactions.business_id = companies.id
158 WHERE companies.company_name = 'Donec Ltd'
159 GROUP BY credit_cards.iban;
160
```



	iban	company_name	average_amount
▶	PT87806228135092429456346	Donec Ltd	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: ¿Cuántas tarjetas están activas?

Crea una nueva tabla para el estado de las tarjetas.

```
CREATE TABLE credit_card_status (  
    card_id VARCHAR(15) PRIMARY KEY,  
    status VARCHAR(15) NOT NULL  
);  
  
INSERT INTO credit_card_status (card_id, status)  
SELECT  
    transactions.card_id,  
    CASE  
        WHEN SUM(CASE WHEN transactions.declined THEN 1 ELSE 0 END) = 3 THEN 'Inactiva'  
        ELSE 'Activa'  
    END AS status  
FROM (  
    SELECT card_id, declined  
    FROM transactions  
    ORDER BY timestamp DESC  
    LIMIT 3  
    ) transactions  
GROUP BY transactions.card_id;  
  
1 • SELECT * FROM transactions_s4.credit_card_status;
```



card_id	status
CcU-2959	Activa
CcU-3960	Activa
CcU-4709	Activa
NULL	NULL

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:

```
#          *****Nivell 3*****
#Exercici 1: Necessitem conèixer el nombre de vegades que s'ha venut cada producte.
CREATE TABLE bridge_products (
    transactions_id VARCHAR(100),
    products_id VARCHAR(100),
    FOREIGN KEY (transactions_id) REFERENCES transactions(id),
    FOREIGN KEY (products_id) REFERENCES products(id)
);


INSERT INTO bridge_products (transactions_id, products_id)
SELECT transactions.id AS transactions_id, products.id AS products_id
FROM transactions
JOIN products
ON FIND_IN_SET(products.id, REPLACE(transactions.product_ids, ' ', '')) > 0;

1 • SELECT * FROM transactions_s4.bridge_products;
```

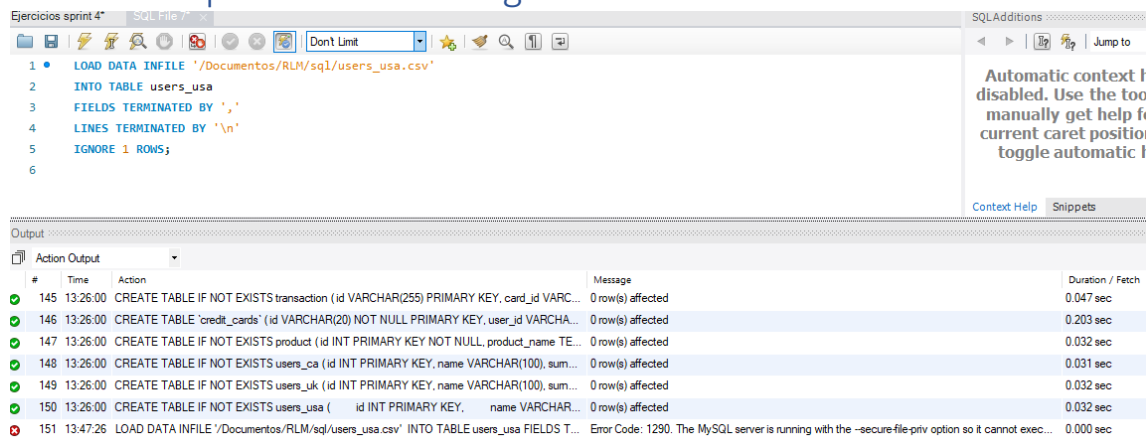
transactions_id	products_id
02C6201E-D90A-1859-B4EE-88D2986D3B02	71
02C6201E-D90A-1859-B4EE-88D2986D3B02	19
02C6201E-D90A-1859-B4EE-88D2986D3B02	1
0466A42E-47CF-8D24-FD01-C0B689713128	97
0466A42E-47CF-8D24-FD01-C0B689713128	47
0466A42E-47CF-8D24-FD01-C0B689713128	43
063FBA79-99EC-66FB-29F7-25726D1764A5	67
063FBA79-99EC-66FB-29F7-25726D1764A5	5
063FBA79-99EC-66FB-29F7-25726D1764A5	47
063FBA79-99EC-66FB-29F7-25726D1764A5	31
0668296C-CDB9-A883-76BC-2E4C44F8C8AE	89
0668296C-CDB9-A883-76BC-2E4C44F8C8AE	83
0668296C-CDB9-A883-76BC-2E4C44F8C8AE	79
06CD9AA5-9B42-D684-DDDD-A5E394FEBA99	43
06CD9AA5-9B42-D684-DDDD-A5E394FEBA99	31
07A46D48-31A3-7E87-65B9-0DA902AD109F	47

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

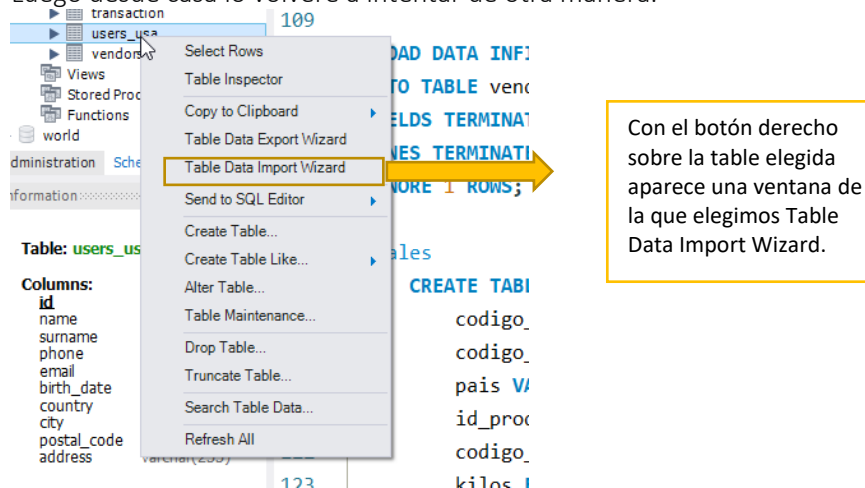
```
202 • SELECT products_id, COUNT(*) AS num_ventas
203 FROM bridge_products
204 GROUP BY products_id;
205
```

<		
Result Grid		
Filter Rows: <input type="text"/>		
Export:  Wrap Cell		
	products_id	num_ventas
▶	1	61
	11	48
	13	60
	17	61
	19	49
	2	65
	23	68
	29	49
	3	51
	31	47
	37	51
	41	53
	43	65
	47	62
	5	49
	53	58
	59	45
	61	57
	67	68
	7	54
	71	54
	73	47
	79	66
	83	57
	89	51
	97	61

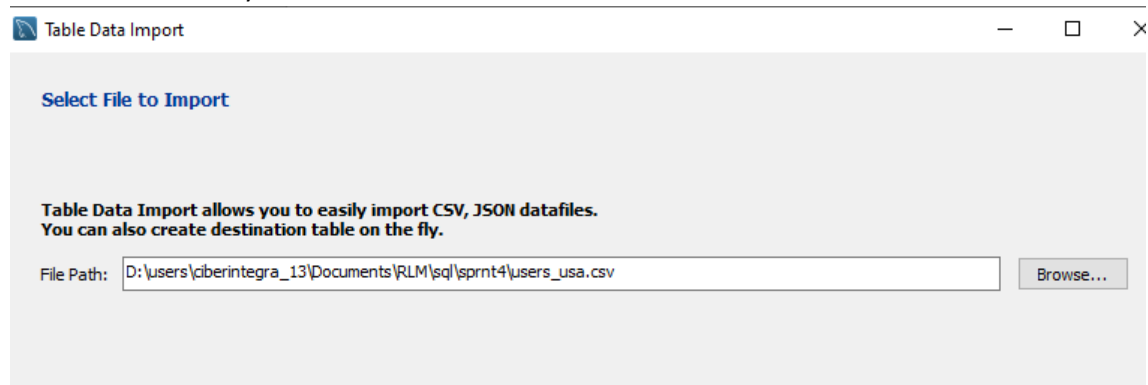
Anexo: Los problemas de carga las tablas



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.



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



Le damos a Next y elegimos destino

Select Destination

Select destination table and additional options.

☒ Use existing table: transactions.users_usa

☐ Create new table: transactions . users_usa

☐ Truncate table before import

Le volvemos a dar a Next

Table Data Import

Configure Import Settings

Detected file format: csv

Encoding: utf-8

Columns:

Source Column	Dest Column
<input checked="" type="checkbox"/> id	id
<input checked="" type="checkbox"/> name	name
<input checked="" type="checkbox"/> surname	surname
<input checked="" type="checkbox"/> phone	phone
<input checked="" type="checkbox"/> email	email
<input checked="" type="checkbox"/> birth_date	birth_date

id	name	surname	phone	email	birth_date	country	city	postal_code	address
1	Zeus	Gamble	1-282-581-...	interdum.e...	Nov 17, 1985	United States	Lowell	73544	348-78:
2	Garrett	Mcconnell	(718) 257-...	integer.vita...	Aug 23, 1992	United States	Des Moines	59464	903 Sit
3	Ciaran	Harrison	(522) 598-...	interdum.fe...	Apr 29, 1998	United States	Columbus	56518	736-206
4	Howard	Stafford	1-411-740-...	ornare.eques...	Feb 18, 1989	United States	Kailua	77417	Ap #54

< Back Next > Cancel

Table Data Import

Import Data

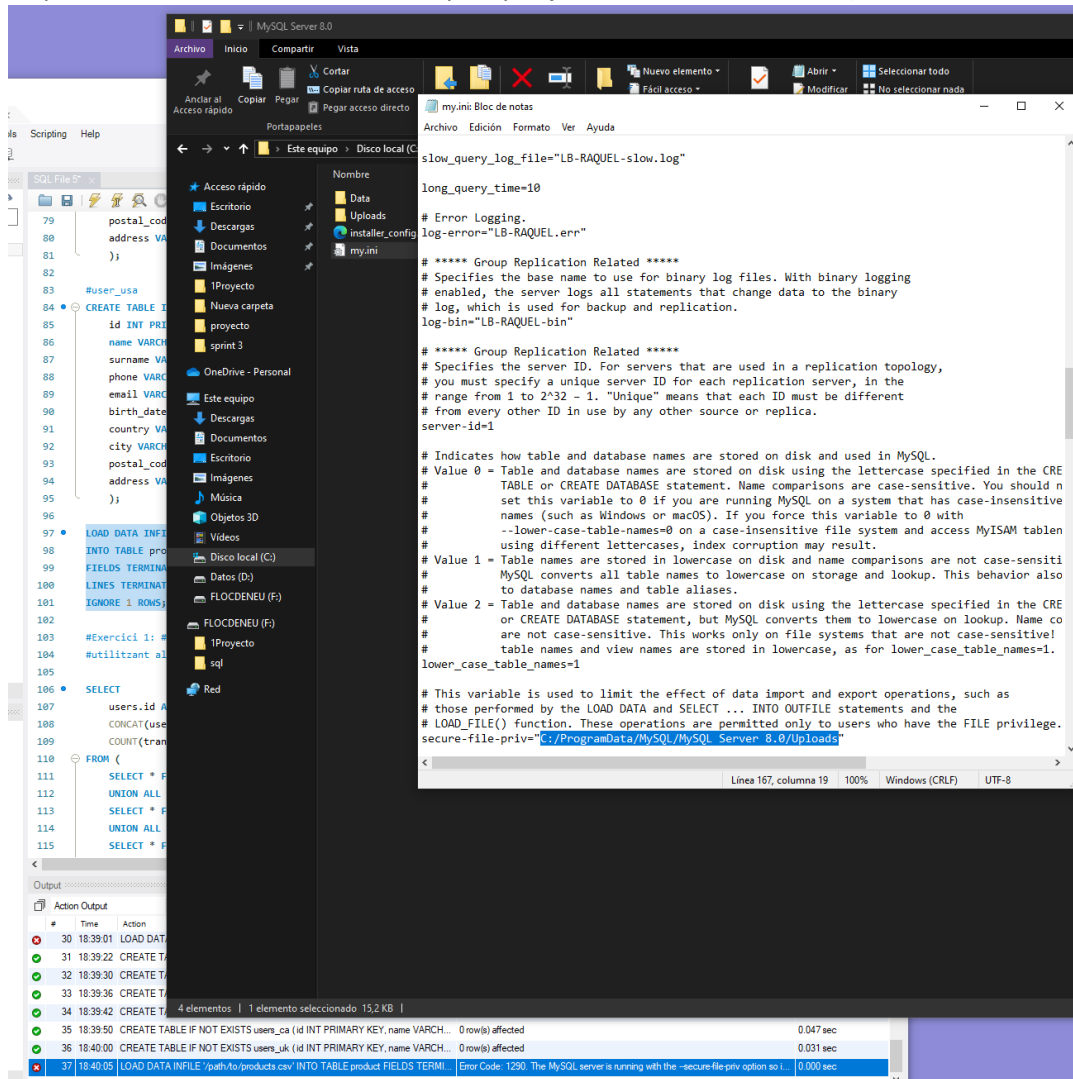
The following tasks will now be performed. Please monitor the execution.

☐ Prepare Import
☐ Import data file

Click [Next >] to execute.

Como corregirlo desde casa:

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



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.

Nombre	Fecha de modificación	Tipo	Tamaño
companies.csv	20/11/2024 10:37	Archivo de valores...	11 KB
credit_cards.csv	20/11/2024 10:37	Archivo de valores...	41 KB
products (1).csv	25/11/2024 11:14	Archivo de valores...	5 KB
transactions.csv	20/11/2024 10:37	Archivo de valores...	72 KB
users_ca.csv	20/11/2024 10:38	Archivo de valores...	9 KB
users_uk.csv	20/11/2024 10:38	Archivo de valores...	7 KB
users_usa.csv	20/11/2024 10:38	Archivo de valores...	19 KB

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

Bueno, a pesar de que ahora funciona, cada tabla da un error y hay que corregir cada una. Products, da problemas a causa del símbolo del dólar y hay que añadir SET price = REPLACE(@price_raw, '\$', '');

Companies , transactions y credit_cards, estan separadas por ; en vez de , y tienen problemas con la falta de comillas en algunos registros, mientras que users,