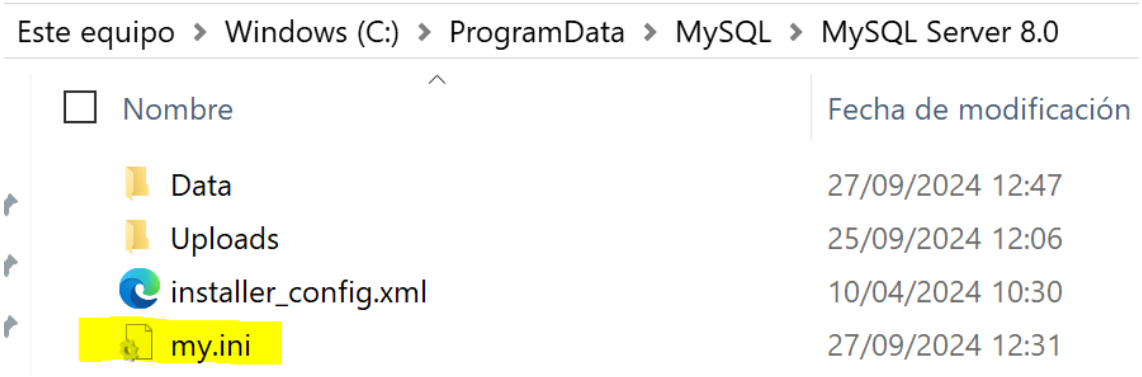
NIVEL 1

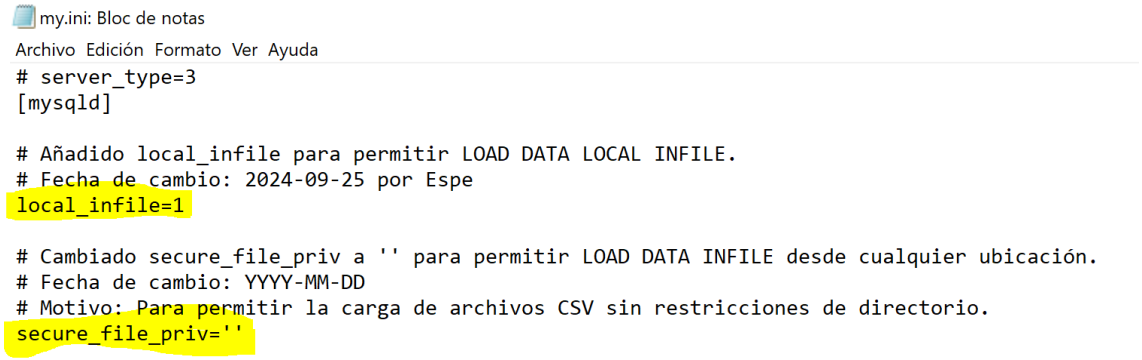
TRABAJO PREVIO

configuraciones para poder cargar los ficheros de forma local directamente con código

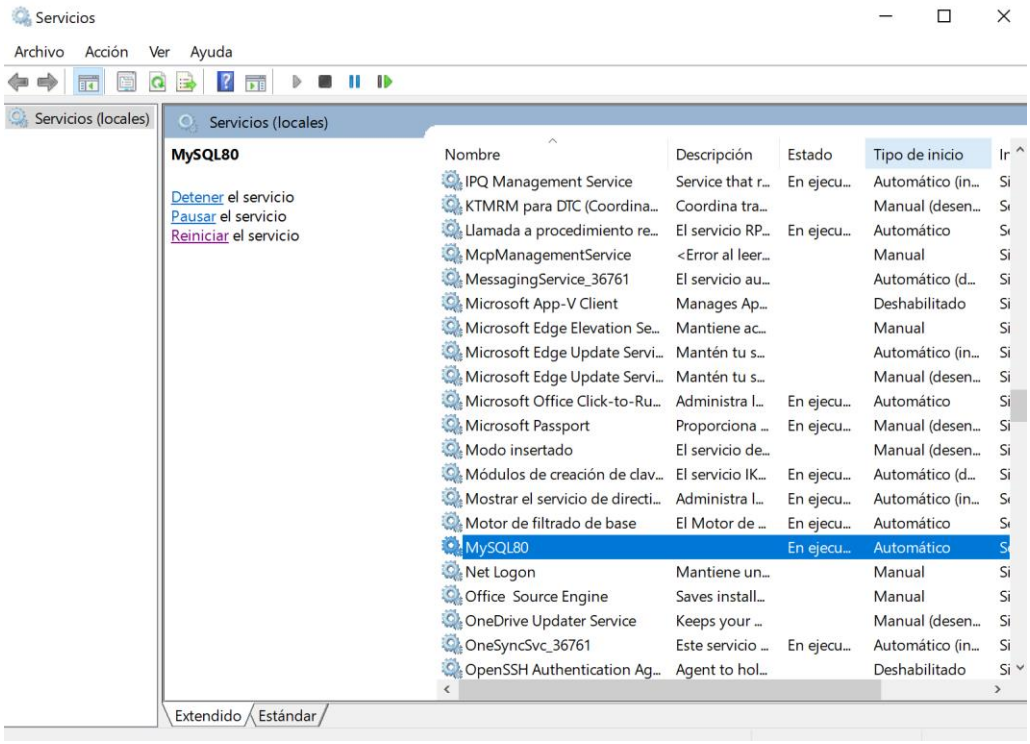
C:\ProgramData\MySQL\MySQL Server 8.0



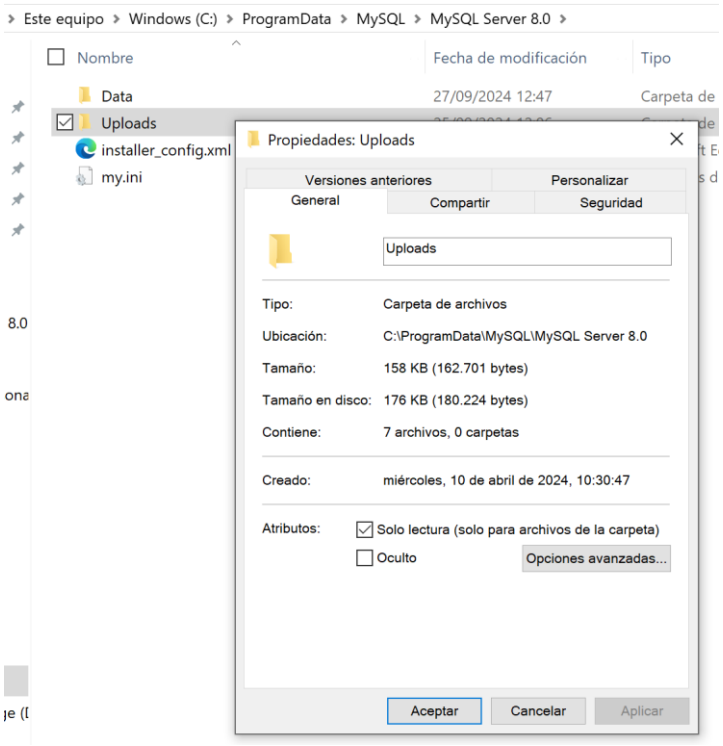
Edito el fichero 'my.ini' para que permita la carga local de datos y nos restrinja ubicaciones:



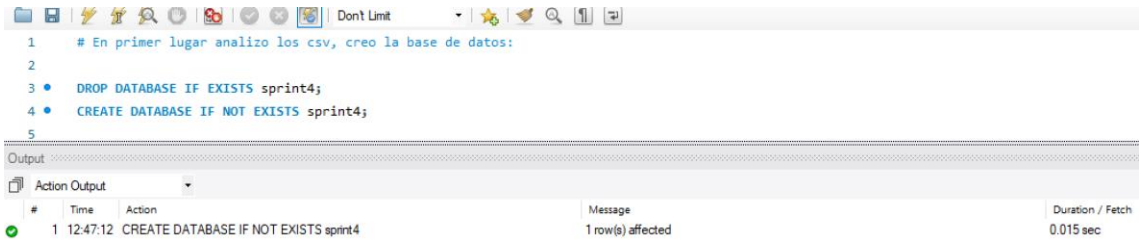
Reinicio servicio del server mysql para poder ejecutar scripts de carga:a



Me aseguro de que la carpeta donde irá a buscar los ficheros por defecto (Uploads), tenga permisos de lectura (botón derecho/propiedades)



Creo la base de datos



Creo las tablas iniciales y ejecuto ca uno de los scripts:

```sql
# A continuación, creo las 6 de las 7 tablas. Transactions necesita que se creen antes users, credit_cards, companies
# para que no de error
DROP TABLE IF EXISTS companies;
CREATE TABLE companies
(
    company_id VARCHAR(15) NOT NULL PRIMARY KEY,
    company_name VARCHAR(200),
    phone VARCHAR(20),
    email VARCHAR(150),
    country VARCHAR(100),
    website VARCHAR(200)
);
DESCRIBE companies;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
company_id	varchar(15)	NO	PRI	NULL	
company_name	varchar(200)	YES		NULL	
phone	varchar(20)	YES		NULL	
email	varchar(150)	YES		NULL	
country	varchar(100)	YES		NULL	

Result 10 | Read Only

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	11:41:52	CREATE TABLE companies (company_id VARCHAR(15) NOT NULL P...	0 row(s) affected	0.032 sec
2	11:41:54	DESCRIBE companies	6 row(s) returned	0.000 sec / 0.000 sec

```sql
DROP TABLE IF EXISTS product;
CREATE TABLE product
(
    id INT NOT NULL PRIMARY KEY,
    product_name VARCHAR(200),
    price DECIMAL(10, 2), #  Haré CAST en el UPLOAD. En el CSV es un VARCHAR por el símbolo $
    colour CHAR(7),
    weight INT,
    warehouse_id VARCHAR(10)
);
DESCRIBE product;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	
product_name	varchar(200)	YES		NULL	
price	decimal(10,2)	YES		NULL	
colour	char(7)	YES		NULL	
weight	int	YES		NULL	

Result 11 | Read Only

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	11:43:45	CREATE TABLE product (id INT NOT NULL PRIMARY KEY, product_na...	0 row(s) affected	0.047 sec
2	11:43:48	DESCRIBE product	6 row(s) returned	0.000 sec / 0.000 sec

```sql
DROP TABLE IF EXISTS users_uk;
CREATE TABLE users_uk
(
    id INT NOT NULL PRIMARY KEY,
    name VARCHAR(100),
    surname VARCHAR(100),
    phone VARCHAR(20),
    email VARCHAR(100),
    birth_date VARCHAR(20),
    country VARCHAR(50),
    city VARCHAR(50),
    postal_code VARCHAR(10),
    address VARCHAR(200)
);
DESCRIBE users_uk;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	
name	varchar(100)	YES		NULL	
surname	varchar(100)	YES		NULL	
phone	varchar(20)	YES		NULL	
email	varchar(100)	YES		NULL	

Result 13 | Read Only

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	11:46:35	CREATE TABLE users_uk (id INT NOT NULL PRIMARY KEY, nam...	0 row(s) affected	0.031 sec
2	11:46:56	DESCRIBE users_uk	10 row(s) returned	0.000 sec / 0.000 sec

```sql
DROP TABLE IF EXISTS users_usa;
CREATE TABLE users_usa
(
    id INT NOT NULL PRIMARY KEY,
    name VARCHAR(100),
    surname VARCHAR(100),
    phone VARCHAR(20),
    email VARCHAR(100),
    birth_date VARCHAR(20),
    country VARCHAR(50),
    city VARCHAR(50),
    postal_code VARCHAR(10),
    address VARCHAR(200)
);
DESCRIBE users_usa;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	
name	varchar(100)	YES		NULL	
surname	varchar(100)	YES		NULL	
phone	varchar(20)	YES		NULL	
email	varchar(100)	YES		NULL	

Result 14 | Read Only

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	11:48:09	CREATE TABLE users_usa (id INT NOT NULL PRIMARY KEY, nam...	0 row(s) affected	0.031 sec
2	11:48:32	DESCRIBE users_usa	10 row(s) returned	0.016 sec / 0.000 sec

```sql
DROP TABLE IF EXISTS users_ca;
CREATE TABLE users_ca
(
    id INT NOT NULL PRIMARY KEY,
    name VARCHAR(100),
    surname VARCHAR(100),
    phone VARCHAR(20),
    email VARCHAR(100),
    birth_date VARCHAR(20),
    country VARCHAR(50),
    city VARCHAR(50),
    postal_code VARCHAR(10),
    address VARCHAR(200)
);
DESCRIBE users_ca;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	
name	varchar(100)	YES		NULL	
surname	varchar(100)	YES		NULL	
phone	varchar(20)	YES		NULL	
email	varchar(100)	YES		NULL	

Result 15 | Read Only

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	11:49:20	CREATE TABLE users_ca (id INT NOT NULL PRIMARY KEY, nam...	0 row(s) affected	0.047 sec
2	11:49:33	DESCRIBE users_ca	10 row(s) returned	0.000 sec / 0.000 sec

Verifico parámetros que me permitirán la carga con código:

```
99  SET GLOBAL local_infile = 1;
100 SHOW VARIABLES LIKE 'local_infile';  # verifica que realmente aparece activado
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Variable_name	Value
local_infile	ON

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	19:18:26	SET GLOBAL local_infile = 1	0 row(s) affected	0.000 sec
2	19:18:30	SHOW VARIABLES LIKE 'local_infile'	1 row(s) returned	0.000 sec / 0.000 sec

Verifico la ruta desde la que acepta cargar los ficheros (carpeta ‘Uploads’) para definir correctamente la ruta en la instrucción load data:

```
99  SET GLOBAL local_infile = 1;
100 SHOW VARIABLES LIKE 'local_infile';  # verifica que realmente aparece activado
101 SHOW VARIABLES LIKE 'secure_file_priv'; # muestra si existe una carpeta por def
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

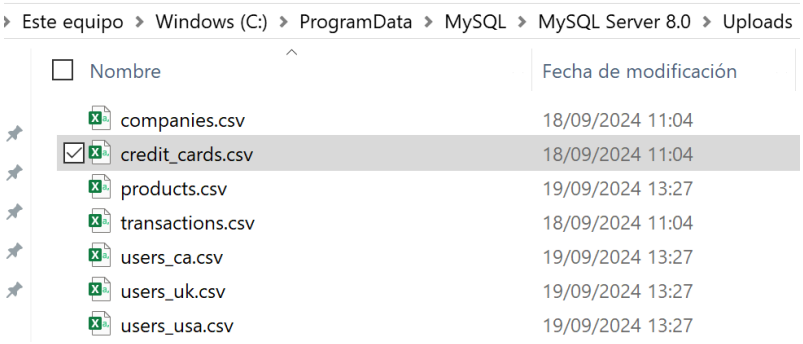
Variable_name	Value
secure_file_priv	C:\ProgramData\MySQL\MySQL Server 8.0\Uploads\

Output

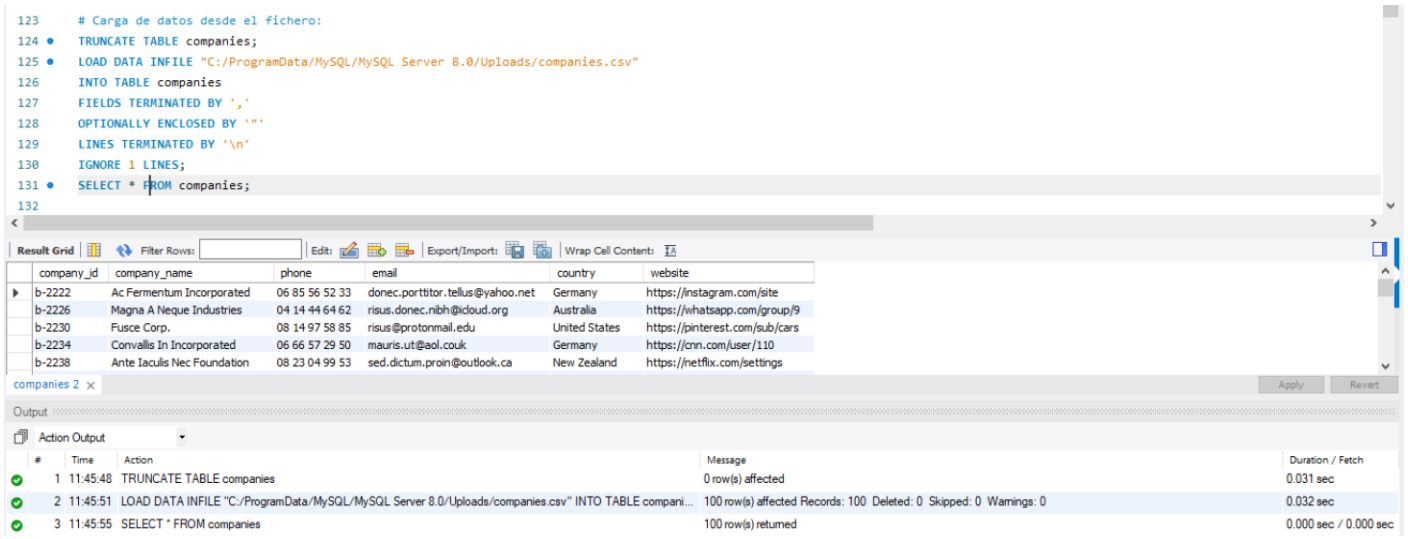
Action Output

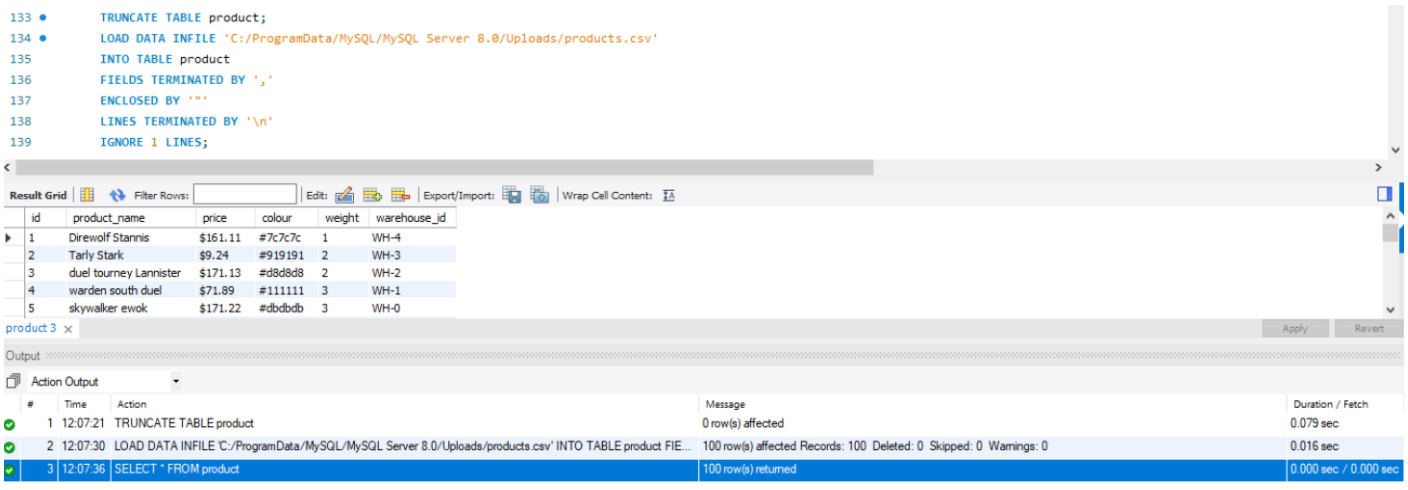
#	Time	Action	Message	Duration / Fetch
1	19:18:26	SET GLOBAL local_infile = 1	0 row(s) affected	0.000 sec
2	19:18:30	SHOW VARIABLES LIKE 'local_infile'	1 row(s) returned	0.000 sec / 0.000 sec
3	19:20:01	SHOW VARIABLES LIKE 'secure_file_priv'	1 row(s) returned	0.000 sec / 0.000 sec

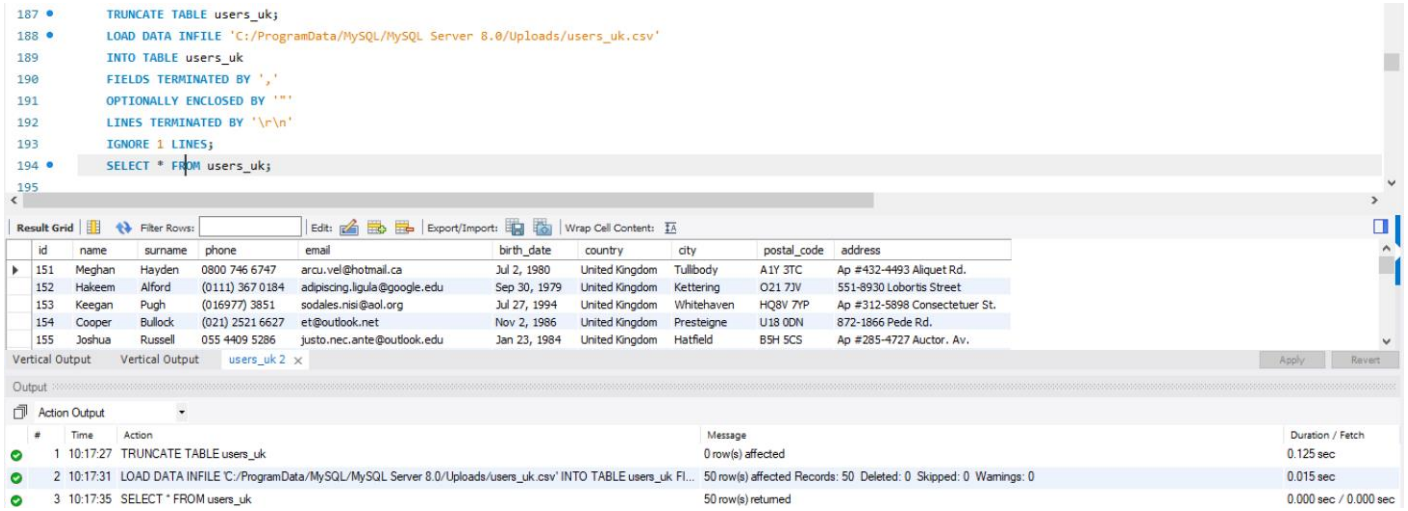
Verifico que los ficheros csv estén en la ruta especificada:



Cargo los ficheros csv de cada tabla:



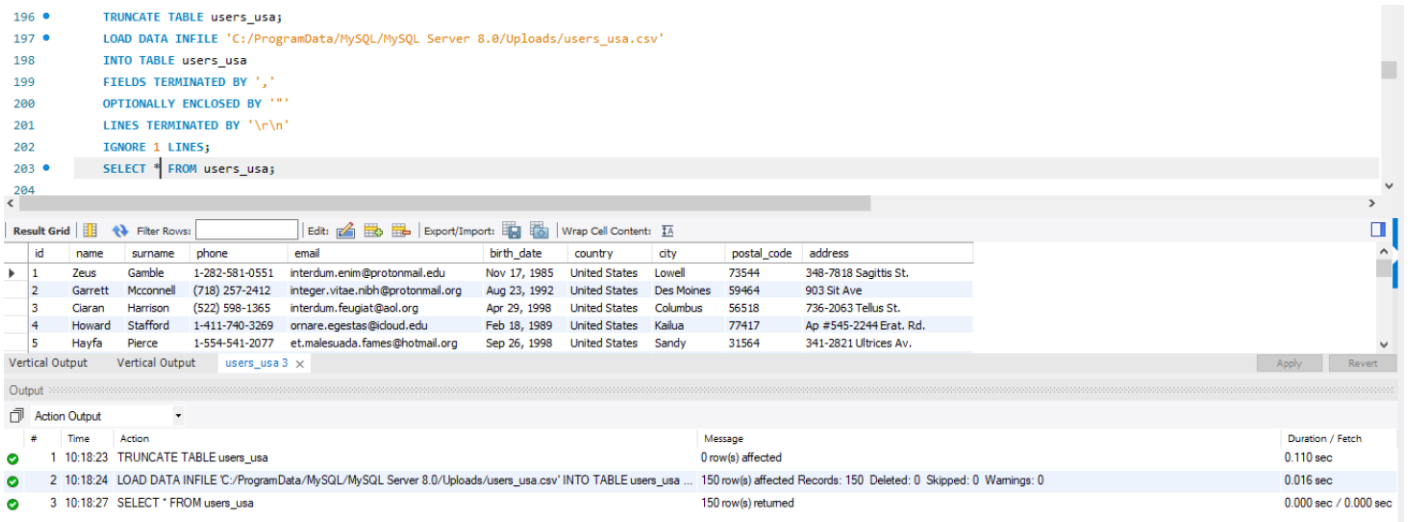




```
TRUNCATE TABLE users_uk;
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_uk.csv'
INTO TABLE users_uk
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\r\n'
IGNORE 1 LINES;
SELECT * FROM users_uk;
```

id	name	surname	phone	email	birth_date	country	city	postal_code	address
151	Meghan	Hayden	0800 746 6747	arcu.vel@hotmail.ca	Jul 2, 1980	United Kingdom	Tullibody	A1Y 3TC	Ap #432-4493 Aliquet Rd.
152	Hakeem	Alford	(0111) 367 0184	adipiscing.ligula@google.edu	Sep 30, 1979	United Kingdom	Kettering	O21 7JV	551-8930 Lobortis Street
153	Keegan	Pugh	(01697) 3851	sodales.nisi@aol.org	Jul 27, 1994	United Kingdom	Whitehaven	HQ8V 7YP	Ap #312-5898 Consectetuer St.
154	Cooper	Bullock	(021) 2521 6627	et@outlook.net	Nov 2, 1986	United Kingdom	Presteigne	U18 0DN	872-1866 Pede Rd.
155	Joshua	Russell	055 4409 5286	justo.nec.ante@outlook.edu	Jan 23, 1984	United Kingdom	Hatfield	B5H 5CS	Ap #285-4727 Auctor. Av.

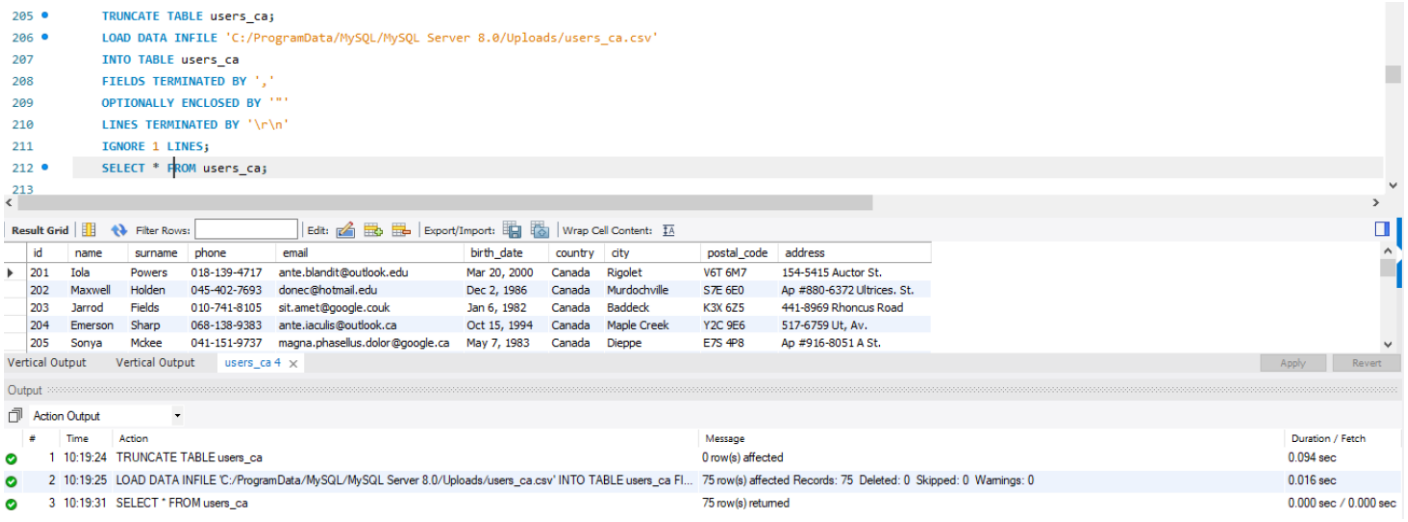
#	Time	Action	Message	Duration / Fetch
1	10:17:27	TRUNCATE TABLE users_uk	0 row(s) affected	0.125 sec
2	10:17:31	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_uk.csv' INTO TABLE users_uk Fl...	50 row(s) affected Records: 50 Deleted: 0 Skipped: 0 Warnings: 0	0.015 sec
3	10:17:35	SELECT * FROM users_uk	50 row(s) returned	0.000 sec / 0.000 sec



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

id	name	surname	phone	email	birth_date	country	city	postal_code	address
1	Zeus	Gamble	1-282-581-0551	interdum.enim@protonmail.edu	Nov 17, 1985	United States	Lowell	73544	348-7818 Sagittis St.
2	Garrett	Mcconnell	(718) 257-2412	integer.vitae.nibh@protonmail.org	Aug 23, 1992	United States	Des Moines	59464	903 Sit Ave
3	Ciaran	Harrison	(522) 598-1365	interdum.feugiat@aol.org	Apr 29, 1998	United States	Columbus	56518	736-2063 Tellus St.
4	Howard	Stafford	1-411-740-3269	ornare.egestas@icloud.edu	Feb 18, 1989	United States	Kailua	77417	Ap #545-2244 Erat. Rd.
5	Hayfa	Pierce	1-554-541-2077	et.malesuada.fames@hotmail.org	Sep 26, 1998	United States	Sandy	31564	341-2821 Ultrices Av.

#	Time	Action	Message	Duration / Fetch
1	10:18:23	TRUNCATE TABLE users_usa	0 row(s) affected	0.110 sec
2	10:18:24	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_usa.csv' INTO TABLE users_usa ...	150 row(s) affected Records: 150 Deleted: 0 Skipped: 0 Warnings: 0	0.016 sec
3	10:18:27	SELECT * FROM users_usa	150 row(s) returned	0.000 sec / 0.000 sec



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

id	name	surname	phone	email	birth_date	country	city	postal_code	address
201	Iola	Powers	018-139-4717	ante.blandit@outlook.edu	Mar 20, 2000	Canada	Rigolet	V6T 6M7	154-5415 Auctor St.
202	Maxwell	Holden	045-402-7693	donec@hotmail.edu	Dec 2, 1986	Canada	Murdochville	S7E 6E0	Ap #880-6372 Ultrices. St.
203	Jarrod	Fields	010-741-8105	sit.amet@google.couk	Jan 6, 1982	Canada	Baddeck	K3X 6Z5	441-8969 Rhoncus Road
204	Emerson	Sharp	068-138-9383	ante.iaculis@outlook.ca	Oct 15, 1994	Canada	Maple Creek	Y2C 9E6	517-6759 Ut, Av.
205	Sonya	Mckee	041-151-9737	magna.phasellus.dolor@google.ca	May 7, 1983	Canada	Dieppe	E7S 4P8	Ap #916-8051 A St.

#	Time	Action	Message	Duration / Fetch
1	10:19:24	TRUNCATE TABLE users_ca	0 row(s) affected	0.094 sec
2	10:19:25	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_ca.csv' INTO TABLE users_ca Fl...	75 row(s) affected Records: 75 Deleted: 0 Skipped: 0 Warnings: 0	0.016 sec
3	10:19:31	SELECT * FROM users_ca	75 row(s) returned	0.000 sec / 0.000 sec

Unifico las 3 tablas de ‘users’ en 1, inserto datos, los muestro y elimino posteriormente las 3 parciales:

```sql
# Unifico la 3 tablas 'users_xxx' en 1 sóla tabla 'users' para cumplir con la tercera forma normal:

# Primero creo la tabla 'users' para tener el contenedor:
    DROP TABLE IF EXISTS users;
    CREATE TABLE users
    (
        id INT NOT NULL PRIMARY KEY,
        name VARCHAR(100),
        surname VARCHAR(100),
        phone VARCHAR(20),
        email VARCHAR(100),
        birth_date VARCHAR(20),
        country VARCHAR(50),
        city VARCHAR(50),
        postal_code VARCHAR(10),
        address VARCHAR(200)
    );
    DESCRIBE users;
# Cargo los datos en 'users' desde cada una de las tablas de usuarios con INSERT INTO y UNION ALL:
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	
name	varchar(100)	YES		NULL	
surname	varchar(100)	YES		NULL	
phone	varchar(20)	YES		NULL	
email	varchar(100)	YES		NULL	

Result 7 | Read Only

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	10:55:30	CREATE TABLE users (id INT NOT NULL PRIMARY KEY, name VARCHAR(100), ...	0 row(s) affected	0.031 sec
2	10:56:02	DESCRIBE users	10 row(s) returned	0.000 sec / 0.000 sec

```sql
# Cargo los datos en 'users' desde cada una de las tablas de usuarios con INSERT INTO y UNION ALL:
    TRUNCATE TABLE users;
    INSERT INTO users
        (id, name, surname, phone, email, birth_date, country, city, postal_code, address)
    SELECT id, name, surname, phone, email, birth_date, country, city, postal_code, address
    FROM users_uk
    UNION ALL
    SELECT id, name, surname, phone, email, birth_date, country, city, postal_code, address
    FROM users_usa
    UNION ALL
    SELECT id, name, surname, phone, email, birth_date, country, city, postal_code, address
    FROM users_ca;
    SELECT * FROM users;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

id	name	surname	phone	email	birth_date	country	city	postal_code	address
1	Zeus	Gamble	1-282-581-0551	interdum.enim@protonmail.edu	Nov 17, 1985	United States	Lowell	73544	348-7818 Sagittis St.
2	Garrett	Mcconnell	(718) 257-2412	integer.vitae.nibh@protonmail.org	Aug 23, 1992	United States	Des Moines	59464	903 Sit Ave
3	Ciaran	Harrison	(522) 598-1365	interdum.feugiat@aol.org	Apr 29, 1998	United States	Columbus	56518	736-2063 Tellus St.
4	Howard	Stafford	1-411-740-3269	ornare.egestas@icloud.edu	Feb 18, 1989	United States	Kailua	77417	Ap #545-2244 Erat. Rd.
5	Hayfa	Pierce	1-554-541-2077	et.malesuada.fames@hotmail.org	Sep 26, 1998	United States	Sandy	31564	341-2821 Ultrices Av.

users 8 | Apply | Revert

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	10:58:14	TRUNCATE TABLE users	0 row(s) affected	0.078 sec
2	10:58:23	INSERT INTO users (id, name, surname, phone, email, birth_date, country, city, post...	275 row(s) affected Records: 275 Duplicates: 0 Warnings: 0	0.047 sec
3	10:58:38	SELECT * FROM users	275 row(s) returned	0.000 sec / 0.000 sec

Elimino tablas parciales de users:

```
255    #Elimino las 3 tablas parciales de users para que no molesten en el esquema E-R
256 •      DROP TABLE users_ca;
257 •      DROP TABLE users_uk;
258 •      DROP TABLE users_usa;
```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	11:14:07	DROP TABLE users_ca	0 row(s) affected	0.016 sec
2	11:14:13	DROP TABLE users_uk	0 row(s) affected	0.062 sec
3	11:14:16	DROP TABLE users_usa	0 row(s) affected	0.016 sec

Una vez creada 'Users', creo la tabla credit cards (tiene FK relacionada con 'users') por lo que no podía crearla antes:

```
32 •  DROP TABLE IF EXISTS credit_cards;
33 •  CREATE TABLE credit_cards
34    (
35        id VARCHAR(20) NOT NULL PRIMARY KEY,
36        user_id INT,
37        iban VARCHAR(34),
38        pan VARCHAR(25),
39        pin INT,
40        cvv INT,
41        track1 VARCHAR(200),
42        track2 VARCHAR(200),
43        expiring_date DATE, #  Haré CAST en el UPLOAD. El formato es  MM/DD/YY y lo pasaré a DATE
44        CONSTRAINT fk_credit_cards_users FOREIGN KEY (user_id) REFERENCES users(id)
45    );
46 •  DESCRIBE credit_cards;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
id	varchar(20)	NO	PRI	NULL	
user_id	int	YES	MUL	NULL	
iban	varchar(34)	YES		NULL	
pan	varchar(25)	YES		NULL	
pin	int	YES		NULL	

esult 12 × Read Only

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	11:45:00	CREATE TABLE credit_cards (id VARCHAR(20) NOT NULL PRIMARY...	0 row(s) affected	0.078 sec
2	11:45:24	DESCRIBE credit_cards	9 row(s) returned	0.000 sec / 0.000 sec

Cargo los datos de Credit_cards:

```
# Cargo los datos de la tabla crecit_cards
TRUNCATE TABLE credit_cards;
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/credit_cards.csv'
INTO TABLE credit_cards
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 LINES
    (@id, @user_id, @iban, @pan, @pin, @cvv, @track1, @track2, @expiring_date)
    SET
        id = @id,
        user_id = @user_id,
        iban = @iban,
        pan = @pan,
        pin = @pin,
        cvv = @cvv,
        track1 = @track1,
        track2 = @track2,
        expiring_date = STR_TO_DATE(@expiring_date, '%m/%d/%y');  # Aquí convertimos el formato MM/DD/YY a YYYY-MM-DD;
SELECT * FROM credit_cards;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

id	user_id	iban	pan	pin	cvv	track1	track2	expiring_date
CcU-2938	275	TR301950312213576817638661	5424465566813633	3257	984	%B8383712448554646^WovsxejDpwiev^8604...	%B7653863056044187=800716333673	2022-10-30
CcU-2945	274	DO2685476374853747521656868 9	5142423821948828	9080	887	%B4621311609958661^UftuyfsSeimxn^06106...	%B4149568437843501=510714033071	2023-08-24
CcU-2952	273	BG45IVQL52710525608255	4556 453 55 5287	4598	438	%B2183285104307501^CddyytcUxwfdq^5907...	%B6778580257827162=6906859740077	2021-06-29
CcU-2959	272	CR7242477244335841535	37246137734937 5	3583	667	%B7281111956795320^XocddjBckecd^09016...	%B4246154489281853=280522391678	2023-02-24
CcU-2966	271	BG72LKTQ15627628377363	448566 886747 7265	4900	130	%B4728932322756223^JhlgvsuFbmwgj^7202...	%B2318571115599881=890821578475	2024-10-29
CcU-2973	270	PT8780622813509242945 6346	544 58654 54343 384	8760	887	%B4761405253275637^HjnnipoBlejrl^7108515...	%B7816169831446746=1310277279	2025-01-30

credit_cards 17 | Apply | Revert

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	12:24:06	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/credit_cards.csv' INTO TABLE credit_c...	275 row(s) affected Records: 275 Deleted: 0 Skipped: 0 Warnings: 0	0.047 sec
2	12:24:09	SELECT * FROM credit_cards	275 row(s) returned	0.000 sec / 0.000 sec

Una vez creada ‘Users’, creo la tabla transactions (tiene FK relacionada con ‘users’) por lo que no podía crearla antes:

```
# Ejecuto script para crear Transactions con las FK correspondientes
DROP TABLE IF EXISTS transactions;
CREATE TABLE transactions
(
    id VARCHAR(100) NOT NULL PRIMARY KEY,
    card_id VARCHAR(20),
    business_id VARCHAR(20),
    timestamp DATETIME,
    amount DECIMAL(10, 2),
    declined TINYINT,
    product_ids VARCHAR(255),
    user_id INT,
    lat VARCHAR(50),
    longitude VARCHAR(50),
    CONSTRAINT fk_transactions_credit_cards FOREIGN KEY (card_id) REFERENCES credit_cards(id), # FK hacia credit_cards
    CONSTRAINT fk_transactions_companies FOREIGN KEY (business_id) REFERENCES companies(company_id), # FK hacia cimpanies
    CONSTRAINT fk_transactions_users FOREIGN KEY (user_id) REFERENCES users(id) # FK hacia users
);
DESCRIBE transactions;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
id	varchar(100)	NO	PRI	NULL	
card_id	varchar(20)	YES	MUL	NULL	
business_id	varchar(20)	YES	MUL	NULL	
timestamp	datetime	YES		NULL	
amount	decimal(10,2)	YES		NULL	

Result 19 | Read Only

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	12:30:40	CREATE TABLE transactions (id VARCHAR(100) NOT NULL PRIMARY KEY, ...	0 row(s) affected	0.094 sec
2	12:30:44	DESCRIBE transactions	10 row(s) returned	0.000 sec / 0.000 sec

Cargo los datos de transactions:

```
259     # Cargo los datos de la tabla Transactions:
260         TRUNCATE TABLE transactions;
261         LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/transactions.csv'
262         INTO TABLE transactions
263         FIELDS TERMINATED BY ';'
264         OPTIONALLY ENCLOSED BY '"'
265         LINES TERMINATED BY '\n'
266         IGNORE 1 LINES;
267         SELECT * FROM transactions;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

id	card_id	business_id	timestamp	amount	declined	product_ids	user_id	lat	longitude
02C6201E-D90A-1859-B4EE-88D2986D3B02	CcU-2938	b-2362	2021-08-28 23:42:24	466.92	0	71, 1, 19	92	81.9184589824	-12.5275561984
0466A42E-47CF-8D24-FD01-C0B689713128	CcU-4219	b-2302	2021-07-26 07:29:18	49.53	0	47, 97, 43	170	-43.9694885888	-117.5251835904
063FBA79-99EC-66FB-29F7-25726D1764A5	CcU-2987	b-2250	2022-01-06 21:25:27	92.61	0	47, 67, 31, 5	275	-81.222680576	-129.049879552
0668296C-CDB9-A883-76BC-2E4C44F8C8AE	CcU-3743	b-2618	2022-01-26 02:07:14	394.18	0	89, 83, 79	265	-34.3593055232	-100.555928064
06CD9AA5-9B42-D684-DDDD-A5E394FEBA99	CcU-2959	b-2346	2021-10-26 23:00:01	279.93	0	43, 31	92	33.7381445632	158.298210304

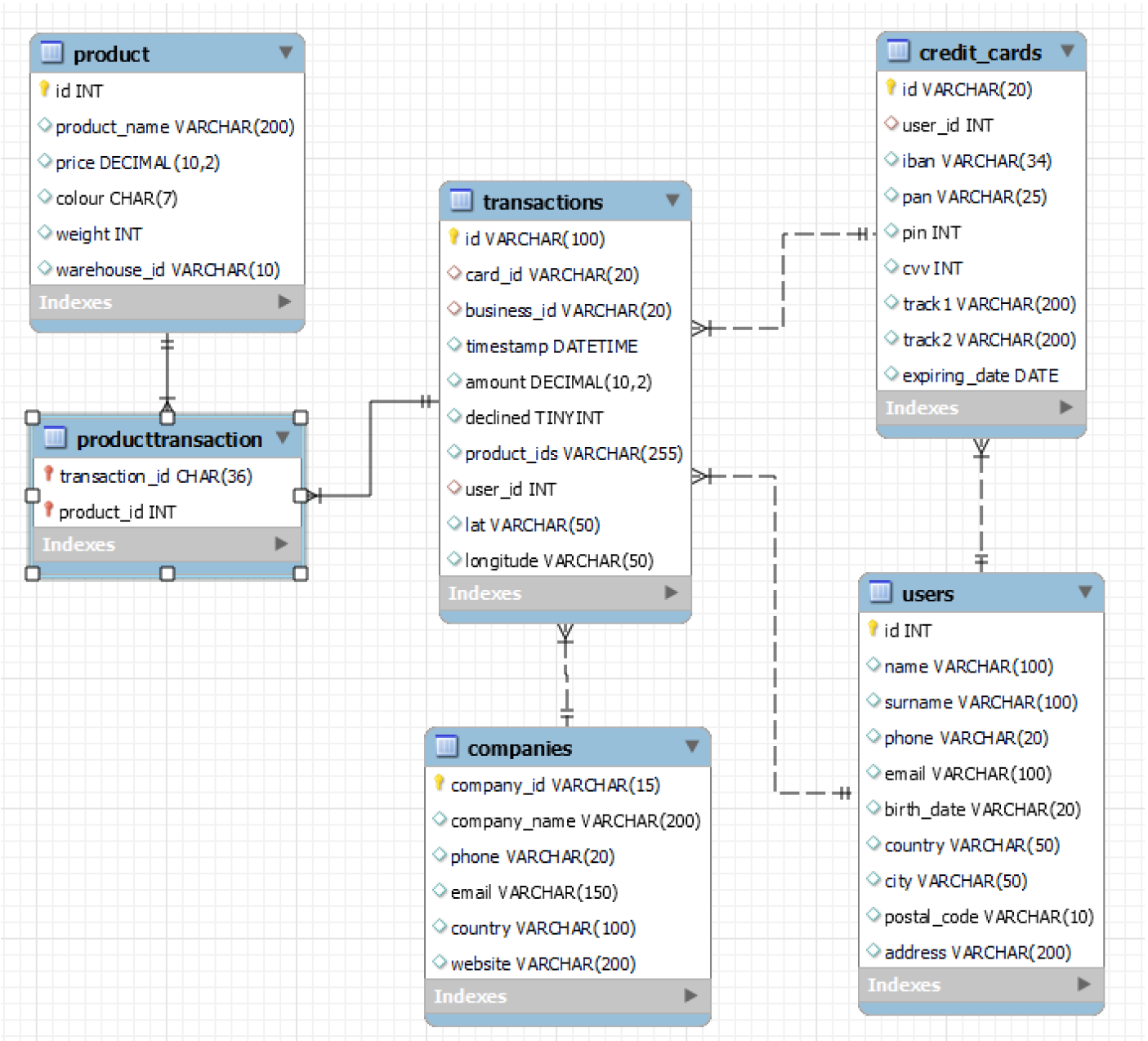
transactions 20 | Apply

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	12:32:32	LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/transacti...	587 row(s) affected Records: 587 Deleted: 0 Skipped: 0 Warnings: 0	0.250 sec
2	12:32:36	SELECT * FROM transactions	587 row(s) returned	0.000 sec / 0.000 sec

Analizo la estructura y relaciones entre las tablas:



```
/* ANÁLISIS DE LAS TABLAS Y ESQUEMA E-R

    Tabla de hechos: transactions:
        Contiene las FK que apuntarán a cada una de la PK de la tabla de dimension a la que pertenezca.
            1. credit_cards (FK card_id). Reación 1-N
            2. company (FK business_id). Relación 1-N
            3. product (FK product_ids). Relación N-N. Como el campo product_ids refleja varios ids en un solo campo,
                hay que crear una tabla intermedia ProductTransaction
            4. users_uk, users_usa, users_ca (FK user_id) Normalización: unir las 3 tablas en una sola ('users')

    Tablas dimensiones:
        1.  credit_cards: tarjetas de crédito utilizadas en las transacciones. Cada transacción está asociada con una tarjeta.
        2.  company: empresas donde se hacen las transacciones.
        3.  product: productos incluidos en las transacciones.
        4.  users_xxx: usuarios de las transacciones. Hay 3 tablas, una tabla por cada zona (ca, uk, usa).
            Estas tablas serían el resultado de aplicar filtros a una tabla general de 'users'. No obstante,
            dado que en el ejercicio no indica referirse a los sprints anteriores, la tabla 'users' la crearé anexando las 3 trablas
*/
```

```sql
-- ----------------------------------------------------------------------------------------------------------------
# Nivell 1
-- ----------------------------------------------------------------------------------------------------------------

# Exercici 1
# Realitza una subconsulta que mostri tots els usuaris amb més de 30 transaccions utilitzant almenys 2 taules.
    SELECT u.id, u.name, u.surname, (SELECT COUNT(t.id)
                                     FROM transactions t
                                     WHERE t.user_id = u.id) AS NumTransactions
    FROM users u
    WHERE u.id IN (SELECT t.user_id
                   FROM transactions t
                   GROUP BY t.user_id
                   HAVING COUNT(t.id) > 30)
    ORDER BY NumTransactions DESC;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

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

No lo pide el ejercicio: vesión con JOIN más sencilla

```sql
    # Esta sería más compacta y sencilla con JOIN (no lo pide el ejercicio)
    SELECT cc.IBAN, ROUND(AVG(t.amount), 2) AS AverageAamount #ROUND 2 DECIMALES. SALEN 5 POR DEFECTO
    FROM transactions t
    JOIN credit_cards cc ON t.card_id = cc.id
    JOIN companies c ON t.business_id = c.company_id
    WHERE c.company_name = 'Donec Ltd'
    GROUP BY cc.IBAN;
```

NIVEL 2

```
328     -- ########################################################################################################################
329     -- Nivell 2
330     -- ########################################################################################################################
331
332     #Crea una nova taula que reflecteixi l'estat de les targetes de crèdit basat en si les últimes tres transaccions van ser declinades i genera la següent consulta:
333     DROP TABLE IF EXISTS cc_active;
334     CREATE TABLE cc_active
335     (
336         credit_card_id VARCHAR(15) PRIMARY KEY,
337         active BOOLEAN
338     );
339     DESCRIBE cc_active;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
credit_card_id	varchar(15)	NO	PRI	NULL	
active	tinyint(1)	YES		NULL	

Result 40 | Read Only

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	15:44:46	CREATE TABLE cc_active (credit_card_id VARCHAR(15) PRIMARY KEY, active B...	0 row(s) affected	0.063 sec
2	15:45:35	DESCRIBE cc_active	2 row(s) returned	0.000 sec / 0.000 sec

OPCIÓN 1 CON ROW_NUMBER

Sólo tiene en cuenta las tarjetas con 3 o más transacciones

```
348     # SCRIPT con ROW NUMBER
349     INSERT INTO cc_active (credit_card_id, active)
350     SELECT TransNumeradas.card_id,
351                         CASE
352                             WHEN SUM(CASE
353                                         WHEN TransNumeradas.declined = 1 THEN 1
354                                         ELSE 0 END
355                                     ) = 3 THEN FALSE   # tarjeca inactiva
356                             ELSE TRUE  # Tarjeta activa
357                         END AS active    # Nombre de esta columna
358     FROM (  SELECT t.card_id, t.declined, ROW_NUMBER() OVER (PARTITION BY t.card_id ORDER BY t.timestamp DESC) AS RowNumber
359             FROM transactions t
360          )  TransNumeradas  # ordena por fecha descenciente y enumera las transacciones de cada tarjeta
361     WHERE TransNumeradas.RowNumber <= 3
362     GROUP BY TransNumeradas.card_id
363     HAVING COUNT(*) >= 3;
364     SELECT * FROM cc_active;
365
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

credit_card_id	active
CcU-2938	1
CcU-2959	1
CcU-2994	1
CcU-3393	1
CcU-3981	1

cc_active 41 | Apply | Revert

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	15:47:14	INSERT INTO cc_active (credit_card_id, active) SELECT TransNumeradas.card_id,...	9 row(s) affected Records: 9 Duplicates: 0 Warnings: 0	0.062 sec
2	15:48:02	SELECT * FROM cc_active	9 row(s) returned	0.000 sec / 0.000 sec

OPCIÓN 1 CON ROW_NUMBER

Tiene en cuenta todas las tarjetasaunque no lleguen a 3 transacciones

```sql
# SCRIPT con ROW NUMBER
    INSERT INTO cc_active (credit_card_id, active)
    SELECT TransNumeradas.card_id,
                            CASE
                                WHEN SUM(CASE
                                            WHEN TransNumeradas.declined = 1 THEN 1
                                            ELSE 0 END
                                        ) = 3 THEN FALSE   # tarjeca inactiva
                                ELSE TRUE  # Tarjeta activa
                            END AS active    # Nombre de esta columna
    FROM (  SELECT t.card_id, t.declined, ROW_NUMBER() OVER (PARTITION BY t.card_id ORDER BY t.timestamp DESC) AS RowNumber
            FROM transactions t
         )  TransNumeradas  # ordena por fecha descenciente y enumera las transacciones de cada tarjeta
    WHERE TransNumeradas.RowNumber <= 3
    GROUP BY TransNumeradas.card_id;
    # HAVING COUNT(*) >= 3; # sólo tiene en cuenta tarjetas con 3 o más transacciones
    SELECT * FROM cc_active;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

credit_card_id	active
CcU-2938	1
CcU-2945	1
CcU-2952	1
CcU-2959	1
CcU-2966	1

cc_active 42 | Apply | Revert

Output

Action Output

#	Time	Action	Message	Duration / Fetch
5	15:54:33	DROP TABLE IF EXISTS cc_active	0 row(s) affected	0.016 sec
6	15:54:35	CREATE TABLE cc_active (credit_card_id VARCHAR(15) PRIMARY KEY, active...	0 row(s) affected	0.047 sec
7	15:54:53	INSERT INTO cc_active (credit_card_id, active) SELECT TransNumeradas.card_i...	275 row(s) affected Records: 275 Duplicates: 0 Warnings: 0	0.016 sec
8	15:55:00	SELECT * FROM cc_active	275 row(s) returned	0.000 sec / 0.000 sec

OPCION 1 sin ROW_NUMBER()

```sql
# OPCION 1 SIN ROW_NUMBER:
INSERT INTO cc_active (credit_card_id, active)
SELECT t.card_id,
                CASE
                    WHEN SUM(CASE WHEN t.declined = 1 THEN 1 ELSE 0 END) = 3 THEN FALSE  -- Si hay 3 declinadas, es inactiva
                    ELSE TRUE  -- Si no, es activa
                END AS active
FROM transactions t
# la conjerá SI NO está ya incluida en la tabla cc_active
WHERE NOT EXISTS (
                    SELECT 1    # comprueba si el id de la tarjeta de transactions aparece en la tabla cc_active==> ES CIERTO QUE NO EXISTE, POR LO QUE
                    FROM cc_active ca
                    WHERE ca.credit_card_id = t.card_id
                  )
GROUP BY t.card_id
HAVING COUNT(t.id) >= 3;  # Para asegurarme de que solo tiene en cuenta inicialmente las tarjetas que tengan mínimo 3 transacciones
SELECT * FROM cc_active;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

credit_card_id	active
CcU-2938	1
CcU-2959	1
CcU-2994	1
CcU-3393	1
CcU-3547	1

cc_active 39 | Apply | Revert

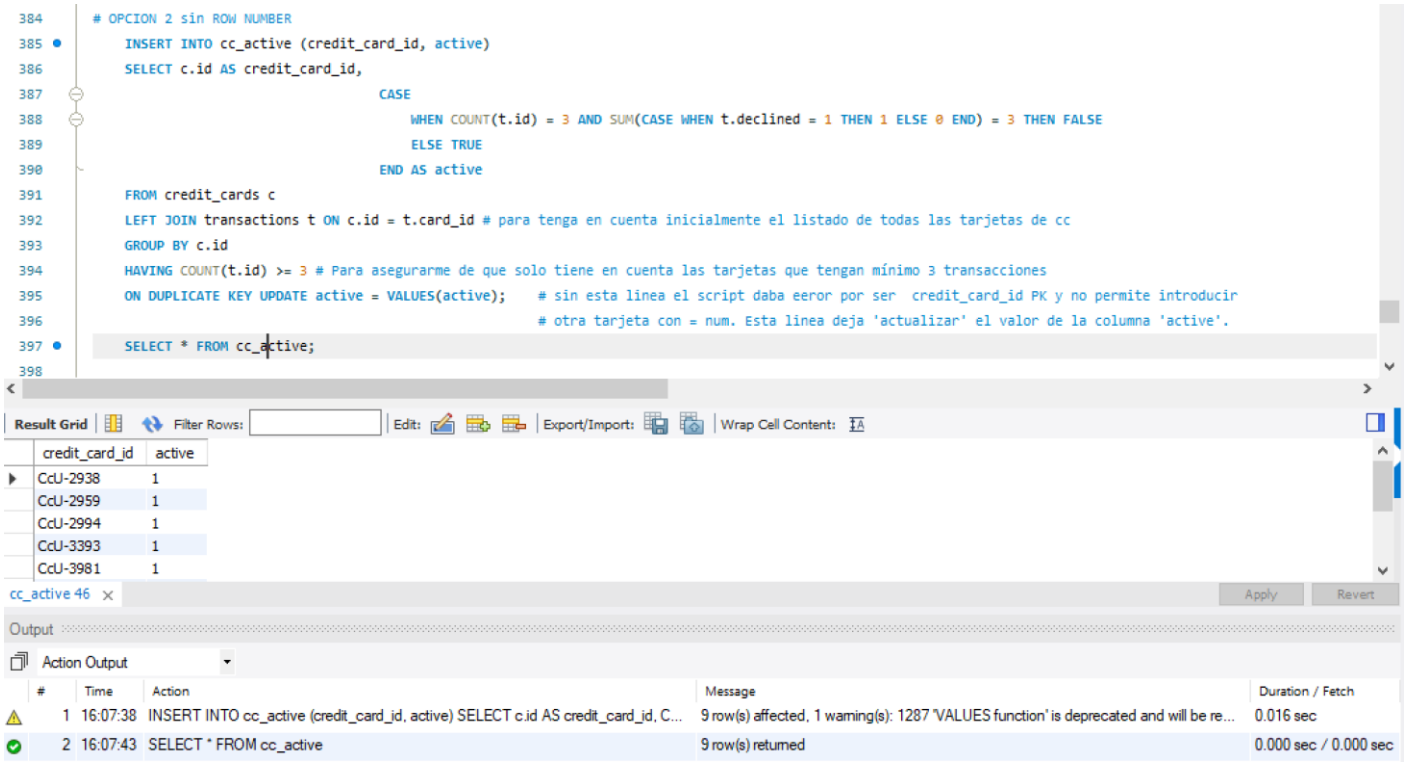
Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	15:39:31	INSERT INTO cc_active (credit_card_id, active) SELECT t.card_id, CASE WHEN ...	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.015 sec
2	15:40:08	SELECT * FROM cc_active	192 row(s) returned	0.000 sec / 0.000 sec

OPCION 2 sin ROW_NUMBER()

Sale una advertencia relacionada con futuras versiones. Puedes dejar de funcionar



NIVEL 3

```sql
-- ================================================================================================================
-- Nivell 3
-- ================================================================================================================
# Crea una taula amb la qual puguem unir les dades del nou arxiu products.csv amb la base de dades creada, tenint en compte
# que des de transaction tens product_ids. Genera la següent consulta:
    DROP TABLE IF EXISTS producttransaction;
    CREATE TABLE ProductTransaction
    (
    transaction_id CHAR(36),
    product_id INT,
    PRIMARY KEY (transaction_id, product_id),  # Clave combinada
    FOREIGN KEY (transaction_id) REFERENCES transactions(id), # FK apunta a tranzactions
    FOREIGN KEY (product_id) REFERENCES product(id) # FK apunta a product
    );
    DESCRIBE producttransaction;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
transaction_id	char(36)	NO	PRI	NULL	
product_id	int	NO	PRI	NULL	

Result 47 | Read Only

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	16:46:22	CREATE TABLE ProductTransaction (transaction_id CHAR(36), product_id...	0 row(s) affected	0.063 sec
2	16:46:25	DESCRIBE producttransaction	2 row(s) returned	0.015 sec / 0.000 sec

```sql
# Exercici 1
# Necessitem conèixer el nombre de vegades que s'ha venut cada producte.

# planteamiento de la query: hay que conseguir identificar la posición de cada id dentro de la columna products id
# No todos tendrán el mismo número de posiciones
# la coma me permite saber cuántos números hay en cada registro: 12, 3, 7
# Tendré que colocar cada id en UNA línea distinta, de forma que la misma transaction se repetira aparecerá en 3 lineas si tiene 3 product_id
# NumColumnas: posición del id en el cmapo product_ids.
# SUBSTRING_INDEX interno: consige la cadena de ids definida por NumColumnas y el externo
# SUBSTRING_INDEX externo, coge la cadena anterior y con el -1 obtiene el último numero despues de la coma, empezando por el final. Así hasta que acbe con todos los ids de ese registro
# CAST finalmente lo toma como un número (era cadena de texto)
```

```sql
    INSERT INTO producttransaction (product_id, transaction_id)
    SELECT
        CAST(SUBSTRING_INDEX(SUBSTRING_INDEX(t.product_ids, ',', ListaNum.NumColumnas), ',', -1) AS UNSIGNED) AS product_id,
        t.id AS transaction_id
    FROM transactions t
    JOIN (
            SELECT 1 AS NumColumnas UNION ALL
            SELECT 2 UNION ALL
            SELECT 3 UNION ALL
            SELECT 4 UNION ALL
            SELECT 5 UNION ALL
            SELECT 6   # he visto que el csv tienen max 4 ids. Escojo 6 por seguridad
        ) ListaNum
    ON CHAR_LENGTH(t.product_ids) - CHAR_LENGTH(REPLACE(t.product_ids, ',', '')) >= ListaNum.NumColumnas - 1;

    # muestro la tabla completa con todos los articulos de todas las transacciones
        SELECT * FROM producttransaction
        ORDER BY transaction_id;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Fetch rows:

transaction_id	product_id
02C6201E-D90A-1859-B4EE-88D2986D3B02	1
02C6201E-D90A-1859-B4EE-88D2986D3B02	19
02C6201E-D90A-1859-B4EE-88D2986D3B02	71
0466A42E-47CF-8D24-FD01-C0B689713128	43
0466A42E-47CF-8D24-FD01-C0B689713128	47

producttransaction 55 | Apply | Revert

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	08:44:08	INSERT INTO producttransaction (product_id, transaction_id) SELECT CAST(SUBSTRING_INDEX(SUBSTRING_I...	1457 row(s) affected Records: 1457 Duplicates: 0 Warnings: 0	0.156 sec
2	08:44:19	SELECT * FROM producttransaction ORDER BY transaction_id	1457 row(s) returned	0.000 sec / 0.000 sec

```sql
#Consulta para presentar los resultados con el número de ventas de cada producto
    SELECT p.id AS Reference, p.product_name AS Description, COUNT(pt.product_id) AS NumVentas
    FROM product p
    JOIN producttransaction pt ON p.id = pt.product_id
    GROUP BY p.id, p.product_name
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Reference	Description	NumVentas
23	riverlands north	68
67	Winterfell	68
79	Direwolf riverlands the	66
2	Tarly Stark	65
43	duel	65

Result 57 | Read Only

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	08:55:17	SELECT p.id AS Reference, p.product_name AS Description, COUNT(pt.product_id) AS NumVentas FROM product ...	26 row(s) returned	0.000 sec / 0.000 sec