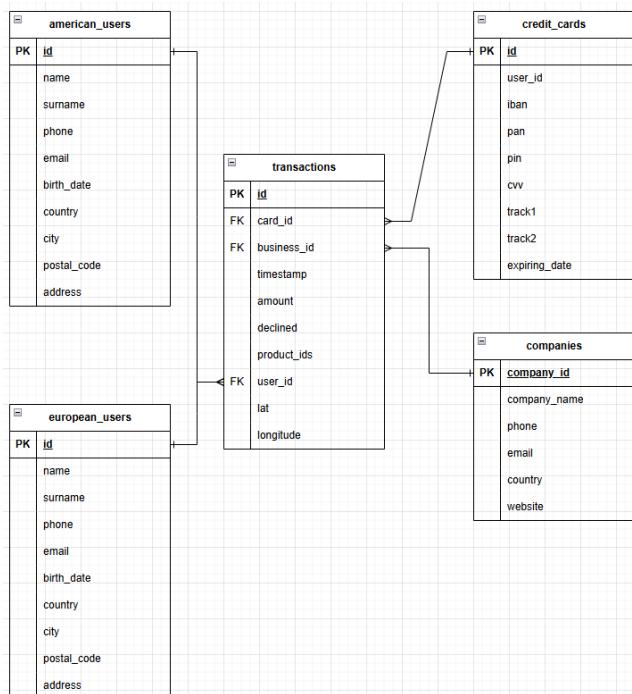


NIVEL 1

Descarga los archivos CSV, estúdialos y diseña una base de datos con un esquema de estrella que contenga, al menos 4 tablas de las que puedes realizar las siguientes consultas:

La nueva base de datos la voy a llamar “transactions_2”

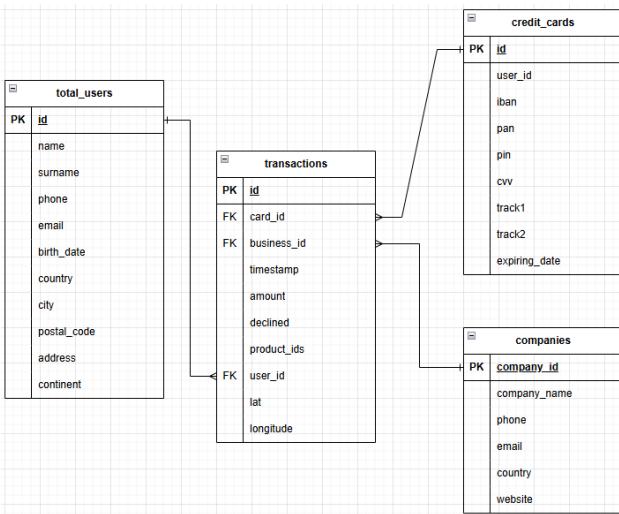
Después de estudiar las tablas haría una configuración en estrella donde la tabla de hechos seria la ““transactions”” y las de dimensiones serían las tablas “american_users”, “european_users”, “companies” y “credit_cards”. Las cardinalidades son 1:N desde las tablas de dimensiones a la de hechos. Quedando un diagrama así:



Pero vemos que las tablas “european_users” y “american_users” son exactamente iguales por lo que podría ponerlas en una única tabla pero perdiendo la granularidad del continente, por lo que para no perderla por si en un futuro la necesitase, haremos:

1. Añadiré a cada tabla un campo nuevo llamado “continent”
2. Este campo nuevo lo relleno con el valor “european” en la tabla “european_users”
3. Este campo nuevo lo relleno con el valor “american” en la tabla “american_users”
4. Creo una tabla nueva llamada “total_users” con la unión de las 2 anteriores al tener los mismos campos.

Quedándome este diagrama:



DEFINICION:

1. Creo la nueva BD llamada “transactions_2”

```
7      -- Primero creo la nueva base de datos
8 •  CREATE DATABASE IF NOT EXISTS transactions_2;
```

Output			
#	Time	Action	Message
7	13:32:17	SHOW DATABASES	9 row(s) returned 0.078 sec / 0.000 sec
8	13:34:32	CREATE DATABASE IF NOT EXISTS transactions_2	1 row(s) affected 0.047 sec

2. Compruebo que se ha creado

```
10     -- Compruebo que se ha creado correctamente
11 •  SHOW DATABASES;
```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
	Database			
▶	information_schema			
	mysql			
	new_schema			
	performance_schema			
	sakila			
	sys			
	tienda_online			
	transactions			
	transactions_2			
	world			

3. La seleccionamos para trabajar con ella

```
13     -- la seleccionamos para trabajar con ella
14 •  USE transactions_2;
```

Output			
#	Time	Action	Message
9	13:35:31	SHOW DATABASES	10 row(s) returned 0.000 sec / 0.000 sec
10	13:38:35	USE transactions_2	0 row(s) affected 0.015 sec

4. Tablas de users:

- a. Creo las tablas “american_users” y “european_users”

```

16      -- Creo la tabla american_users
17 • CREATE TABLE IF NOT EXISTS american_users (
18      id VARCHAR(10) PRIMARY KEY,
19      name VARCHAR(100),
20      surname VARCHAR(100),
21      phone VARCHAR(150),
22      email VARCHAR(150),
23      birth_date VARCHAR(100),
24      country VARCHAR(150),
25      city VARCHAR(150),
26      postal_code VARCHAR(100),
27      address VARCHAR(255)
28 );

```

Output			
	Action	Time	Message
90	DROP TABLE transactions_2.'american_users'	11:58:31	0 row(s) affected
91	CREATE TABLE IF NOT EXISTS american_users (id VARCHAR(10) PRIMARY KEY, name VARCHAR(100), surname VARCHAR(100), phone VARC...)	11:58:41	0 row(s) affected

```

30      -- Creo la tabla european_users
31 • CREATE TABLE IF NOT EXISTS european_users (
32      id VARCHAR(10) PRIMARY KEY,
33      name VARCHAR(100),
34      surname VARCHAR(100),
35      phone VARCHAR(150),
36      email VARCHAR(150),
37      birth_date VARCHAR(100),
38      country VARCHAR(150),
39      city VARCHAR(150),
40      postal_code VARCHAR(100),
41      address VARCHAR(255)
42 );
43

```

Output			
	Action	Time	Message
91	CREATE TABLE IF NOT EXISTS american_users (id VARCHAR(10) PRIMARY KEY, name VARCHAR(100), surname VARCHAR(100), phone VARC...)	11:58:41	0 row(s) affected
92	CREATE TABLE IF NOT EXISTS european_users (id VARCHAR(10) PRIMARY KEY, name VARCHAR(100), surname VARCHAR(100), phone VARC...)	11:59:40	0 row(s) affected

```

44      -- me aseguro que se han creado
45 • SHOW TABLES;

```

Result Grid		Filter Rows:
	Tables_in_transactions_2	
▶	american_users	
▶	european_users	

Output			
	Action	Time	Message
92	CREATE TABLE IF NOT EXISTS european_users (id VARCHAR(10) PRIMARY KEY, name VARCHAR(100), surname VARCHAR(100), phone VARC...)	11:59:40	0 row(s) affected
93	SHOW TABLES	12:02:22	2 row(s) returned

- b. Cargo los datos de los ficheros csv en las tablas “european_users” y “american_users”. Pero primero los tengo que poner en el directorio: C:\ProgramData\MySQL\MySQL Server 8.0\Uploads por temas de seguridad es el directorio donde deben estar los ficheros.

```

49      -- Introduzco los datos del fichero american_users.csv que me da el ejercicio
50 • LOAD DATA
51      INFILE 'C:\\\\ProgramData\\\\MySQL\\\\MySQL Server 8.0\\\\Uploads\\\\american_users.csv'
52      INTO TABLE american_users
53      FIELDS TERMINATED BY ','
54      ENCLOSED BY ''
55      LINES TERMINATED BY '\\n'
56      IGNORE 1 ROWS
57 ;

```

Output		
Action	Time	Action
93	12:02:22	SHOW TABLES
94	12:05:08	LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\american_users.csv' INTO TABLE american_users FIELDS TERMINATED BY ',' ENCLOSED BY '\"' LINES TERMINATED BY '\\n' IGNORE 1 ROWS;

```

59      -- Cuento las filas introducidas para comprobar que tiene las mismas que el csv que son 1010 sin contar la cabecera
60 •   SELECT FORMAT(COUNT(*),0) AS total_records
61     FROM american_users;

```

Result Grid	Filter
total_records	
▶ 1,010	

Output		
Action	Time	Action
34	18:25:06	SELECT COUNT(*) AS total_records FROM american_users
35	18:25:24	SELECT FORMAT(COUNT(),0) AS total_records FROM american_users

```

63      -- Introduzco los datos del fichero european_users.csv que me da el ejercicio
64 •   LOAD DATA
65       INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\european_users.csv'
66       INTO TABLE european_users
67       FIELDS TERMINATED BY ','
68       ENCLOSED BY '\"'
69       LINES TERMINATED BY '\\n'
70       IGNORE 1 ROWS
71 ;

```

Output		
Action	Time	Action
98	12:10:45	SELECT COUNT(*) AS total_records FROM european_users AS eu
99	12:10:49	LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\european_users.csv' INTO TABLE american_users FIELDS TERMINATED BY ',' ENCLOSED BY '\"' LINES TERMINATED BY '\\n' IGNORE 1 ROWS;

```

72      -- Cuento las filas introducidas para comprobar que tiene las mismas que el csv que son 3990 sin contar la cabecera
73 •   SELECT FORMAT(COUNT(*),0) AS total_records
74     FROM european_users AS eu;

```

Result Grid	Filter
total_records	
▶ 3,990	

Output		
Action	Time	Action
28	18:19:13	SELECT COUNT(*) AS total_records FROM european_users AS eu
29	18:19:34	SELECT FORMAT(COUNT(),0) AS total_records FROM european_users AS eu

- c. Añado el campo llamado “continent”, en ambas tablas y luego lo relleno en todos los registros de la tabla “american_users” con el dato “american” y en la tabla “european_users” con ““european”. Así no pierdo granularidad

```

87      -- añado el campo continent
88 •   ALTER TABLE american_users ADD continent VARCHAR(150);
89

```

Output		
Action	Time	Action
112	12:15:22	SELECT COUNT(*) AS total_records FROM european_users AS eu
113	12:18:31	ALTER TABLE american_users ADD continent VARCHAR(150)

```

91      -- compruebo que la columna continent esta vacia
92 •   SELECT au.*
93     FROM american_users AS au;

```

Result Grid											Edit: Export/Import: Wrap Cell Content: Fetch rows:
	id	name	surname	phone	email	birth_date	country	city	postal_code	address	continent
▶	1	Zeus	Gamble	1-282-581-0551	interdum.enim@protonmail.edu	Nov 17, 1985	United States	New York	10001	348-7818 Sagittis St.	NULL
	10	Robert	Mccarthy	(324) 746-6771	fermentum@protonmail.com	Apr 30, 1984	United States	San Jose	95101	P.O. Box 773	NULL
	100	Melodie	Mclean	1-677-221-7152	risus.varius@google.ca	Sep 15, 1989	United States	San Jose	95101	Ap #644-8492 Sagittis St.	NULL
	1006	Rbkocq	Swkmynuj	+38-186-8580	rbkocq.swkmynuj@example.com	Dec 2, 1989	United States	New York	10001	567 Swkmynuj St	NULL
	101	Sarah	Beck	(358) 691-4345	vitae.risus@aol.co.uk	Apr 9, 1983	United States	Philadelphia	19101	665-9047 In	NULL
	1011	Vosctr	Cpmnpofu	+74-726-8327	vosctr.cpmnpofu@example.com	Mar 21, 1954	United States	San Jose	95101	775 Cpmnpofu St	NULL
	1015	Ydxljl	Yhswlpqn	+84-730-3250	ydxljl.yhswlpqn@example.com	Nov 13, 2000	Canada	Hamilton	L8E 0A1	209 Yhswlpqn St	NULL
	1018	Ugddlo	Zpbkcjgo	+78-537-1497	ugddlo.zpbkcjgo@example.com	Aug 7, 1950	United States	Chicago	60601	47 Zpbkcjgo St	NULL
	102	Jasper	Landry	1-397-765-1118	consectetuer.euismod@aol.org	Apr 16, 1982	United States	Philadelphia	19101	Ap #374-7325 Sodales Rd.	NULL
	103	Upton	Chavez	(227) 785-6484	euismod.est@aol.ca	Mar 15, 1986	United States	Philadelphia	19101	1990 Vel	NULL
	1037	Igeldc	Blnegpaz	+64-495-5490	igeldc.blnegpaz@example.com	Oct 26, 2000	Canada	Hamilton	L8E 0A1	634 Blnegpaz St	NULL
	104	Martha	Barlow	(732) 326-5448	vulpulture@hotmail.net	Oct 29, 1988	United States	Los Angeles	90001	Ap #311-7103 In Avenue	NULL
	1047	Gfmrvh	Kivnsnmc	+63-719-3839	nfmdvh.kivnsnmc@example.com	Nov 26, 1960	United States	Houston	77001	40R Kivnsnmc St	NULL

Output											
Action Output											
#	Time	Action	Message						Duration / Fetch		
●	113	12:18:31	ALTER TABLE american_users ADD continent VARCHAR(150)							0.015 sec	
●	114	12:19:18	SELECT * FROM american_users AS au						1010 row(s) returned	0.000 sec / 0.015 sec	

Previamente compruebo que el índice "0" no existe

```
95      -- Cargo el valor american en la campo continent para todos los registros
96  ● UPDATE american_users
97    SET continent = "american"
98    WHERE id <> "0";
```

Output											
Action Output											
#	Time	Action	Message						Duration / Fetch		
●	115	12:21:20	SELECT au.* FROM american_users AS au						1010 row(s) returned	0.000 sec / 0.000 sec	
●	116	12:22:55	UPDATE american_users SET continent = "american" WHERE id <> "0"						1010 row(s) affected Rows matched: 1010 Changed: 1010 Warnings: 0	0.047 sec	

```
89      -- compruebo que la columna continent ya tiene el valor american y esta en todos los registros (1010)
90  ● SELECT FORMAT(COUNT(au.continent),0) AS total_records_with_american
91    FROM american_users AS au;
```

Result Grid											Edit: Export/Import: Wrap Cell Content: Fetch rows:
	total_records_with_american										
▶	1,010										

Output											
Action Output											
#	Time	Action	Message						Duration / Fetch		
●	25	18:16:52	SELECT COUNT(au.continent) AS total_records_with_american FROM american_users AS au						1 row(s) returned	0.000 sec / 0.000 sec	

```
94      -- añado el campo continent a la tabla european_users
95  ● ALTER TABLE european_users ADD continent VARCHAR(150);
```

Output											
Action Output											
#	Time	Action	Message						Duration / Fetch		
●	118	12:27:05	SELECT COUNT(au.continent) AS total_records_with_american FROM american_users AS au						1 row(s) returned	0.000 sec / 0.000 sec	
●	119	12:32:07	ALTER TABLE european_users ADD continent VARCHAR(150)						0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.016 sec	

```
97      -- compruebo que la columna continent esta vacia
98  ● SELECT eu.*
99    FROM european_users AS eu;
```

Result Grid											Edit: Export/Import: Wrap Cell Content: Fetch rows:
	id	name	surname	phone	email	birth_date	country	city	postal_code	address	continent
▶	1000	Amkjrv	Qbulrxbp	+48-258-9936	amkjrv.qbulrxbp@example.com	May 17, 1970	Germany	Stuttgart	70173	215 Qbulrxbp St	NULL
	1001	Nfvrbl	Oydaiwbg	+94-121-2522	nfvrbl.oydaiwbg@example.com	Mar 4, 1994	Germany	Cologne	50667	121 Oydaiwbg St	NULL
	1002	Ijbfdm	Jbddzhvp	+70-120-3668	ijbfdm.jbddzhvp@example.com	Sep 27, 2001	Germany	Munich	80331	412 Jbddzhvp St	NULL
	1003	Uyciig	Sfbdynmjz	+58-123-6968	uyciig.sfbdynmjz@example.com	Jan 20, 1981	Germany	Stuttgart	70173	735 Sfbdynmjz St	NULL
	1004	Yjqurq	Ojizvgqi	+77-944-2340	yjqurq.ojizvgqi@example.com	Jul 27, 1954	Germany	Munich	80331	685 Ojizvgqi St	NULL
	1005	Mnlqtu	Glofegwk	+54-801-2627	mnlqtu.glofegwk@example.com	Nov 15, 1962	Portugal	Funchal	9000-001	8 Glofegwk St	NULL
	1007	Ehrclp	Xahkzrlm	+71-862-6101	ehrclp.xahkzrlm@example.com	Nov 8, 1971	Spain	Sevilla	41001	205 Xahkzrlm St	NULL
	1008	Securp	Faofvqfy	+76-822-2041	securp.faofvqfy@example.com	Nov 13, 1957	Sweden	Stockholm	111 20	535 Faofvqfy St	NULL
	1009	Rsznah	Vfqffqaqt	+67-294-3155	rsvnah.vfqffqaqt@example.com	Nov 10, 1952	Spain	Valencia	46001	507 Vfqffqaqt St	NULL
	1010	Dvugzj	Aaesjmca	+78-476-4768	dvugzj.aaesjmca@example.com	Dec 23, 1974	United Kingdom	Birmingham	B1 1AA	822 Aaesjmca St	NULL
	1012	Dwufbr	Gsryolyv	+57-569-7693	dwufbr.gsryolyv@example.com	Aug 10, 1981	Netherlands	Amsterdam	1011	816 Gsryolyv St	NULL
	1013	Kaahoi	Gktlopru	+56-147-5969	kaahoi.gktlopru@example.com	Nov 24, 1993	Poland	Wroclaw	50-001	555 Gktlopru St	NULL
	1014	Frxdix	Onshifev	+69-709-4077	frxdix.onshifev@example.com	Mar 6, 1984	France	Nantes	44000	980 Onshifev St	NULL

Output											
Action Output											
#	Time	Action	Message						Duration / Fetch		
●	119	12:32:07	ALTER TABLE european_users ADD continent VARCHAR(150)						0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.016 sec	
●	120	12:32:45	SELECT eu.* FROM european_users AS eu						3950 row(s) returned	0.000 sec / 0.016 sec	

```

101      -- Cargo el valor european en la campo continent para todos los registros
102 • UPDATE european_users
103     SET continent = "european"
104     WHERE id <> "0";

```

Output:

#	Time	Action	Message	Duration / Fetch
120	12:32:45	SELECT eu.* FROM european_users AS eu	3990 row(s) returned	0.000 sec / 0.016 sec
121	12:34:42	UPDATE european_users SET continent = "european" WHERE id <> "0"	3990 row(s) affected Rows matched: 3990 Changed: 3990 Warnings: 0	0.141 sec

```

106      -- compruebo que la columna continent ya tiene el valor european y esta en todos los registros (3990)
107 • SELECT FORMAT(COUNT(eu.continent),0) AS total_records_with_european
108     FROM european_users AS eu;

```

Result Grid | Filter Rows:

total_records_with_european
3,990

Output:

#	Time	Action	Message	Duration / Fetch
29	18:19:34	SELECT FORMAT(COUNT(),0) AS total_records FROM european_users AS eu	1 row(s) returned	0.000 sec / 0.000 sec
30	18:21:47	SELECT FORMAT(COUNT(eu.continent),0) AS total_records_with_european FROM european_users AS eu	1 row(s) returned	0.015 sec / 0.000 sec

- d. Creo una tabla nueva llamada “total_users” donde uniré todos los datos de las tablas “european_users” y “american_users”. La estructura es igual que la de las 2 anteriores.

```

115      -- Creo la tabla total_users de igual estructura que las anteriores
116 • CREATE TABLE IF NOT EXISTS total_users (
117     id VARCHAR(10) PRIMARY KEY,
118     name VARCHAR(100),
119     surname VARCHAR(100),
120     phone VARCHAR(150),
121     email VARCHAR(150),
122     birth_date VARCHAR(100),
123     country VARCHAR(150),
124     city VARCHAR(150),
125     postal_code VARCHAR(100),
126     address VARCHAR(255),
127     continent VARCHAR(150)
128 );

```

Output:

#	Time	Action	Message	Duration / Fetch
124	12:38:20	SELECT eu.* FROM european_users AS eu	3990 row(s) returned	0.000 sec / 0.000 sec
125	12:40:20	CREATE TABLE IF NOT EXISTS total_users (id VARCHAR(10) 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), continent VARCHAR(150))	0 rows affected	0.016 sec

- e. Voy a copiar todos los registros desde las tablas “european_users” y “american_users” a la tabla “total_users”. Si coincidiese algún índice entre las 2 tablas, me daría un error al copiar la segunda tabla. He revisado los csv y no hay índices coincidentes. El total de registros debe ser $1010 + 3990 = 5000$.

```

130      -- Copio todos los registros de las tablas european_users y american_users una vez comprobados que no se repiten los indices entre las 2 tablas (si se repitiese alguno daria error)
131 • INSERT INTO total_users SELECT * FROM european_users;
132 • INSERT INTO total_users SELECT * FROM american_users;

```

Output:

#	Time	Action	Message	Duration / Fetch
126	12:45:05	INSERT INTO total_users SELECT * FROM european_users	3990 row(s) affected Records: 3990 Duplicates: 0 Warnings: 0	0.094 sec
127	12:45:11	INSERT INTO total_users SELECT * FROM american_users	1010 row(s) affected Records: 1010 Duplicates: 0 Warnings: 0	0.031 sec

```

134      -- Verifico que se han introducido el total de registros  $1010 + 3990 = 5000$ 
135 • SELECT tu.continent AS continent, FORMAT(COUNT(*),0) AS users
136     FROM total_users AS tu
137     GROUP BY tu.continent;

```

Result Grid | Filter F

continent	users
american	1,010
eu	3,990

Output			
Action	Time	Action	Message
32	18:23:42	SELECT FORMAT(COUNT(eu.continent),0) AS total_records, eu.continent AS continent FROM european_users AS eu	1 row(s) returned
33	18:24:11	SELECT tu.continent AS continent, FORMAT(COUNT(*) AS users FROM total_users AS tu GROUP BY tu.continent)	2 row(s) returned

f. Creo la tabla “companies” e introduzco los datos del fichero companies.csv que me da el ejercicio

```
142      -- Creo la tabla companies
143 • CREATE TABLE IF NOT EXISTS companies (
144      company_id VARCHAR(15) PRIMARY KEY,
145      company_name VARCHAR(255),
146      phone VARCHAR(15),
147      email VARCHAR(100),
148      country VARCHAR(100),
149      website VARCHAR(255)
150 );
```

Output			
Action	Time	Action	Message
130	12:48:52	SELECT tu.continent AS continent, COUNT(*) AS users FROM total_users AS tu GROUP BY tu.continent	2 row(s) returned
131	12:58:08	CREATE TABLE IF NOT EXISTS companies (company_id VARCHAR(15) PRIMARY KEY, company_name VARCHAR(255), phone VARCHAR(15), ... 0 rows affected	0.000 sec / 0.000 sec 0.063 sec

```
152      -- Introduzco los datos del fichero companies.csv que me da el ejercicio
153 • LOAD DATA
154      INFILE 'C:\\\\ProgramData\\\\MySQL\\\\MySQL Server 8.0\\\\Uploads\\\\companies.csv'
155      INTO TABLE companies
156      FIELDS TERMINATED BY ','
157      ENCLOSED BY '\"'
158      LINES TERMINATED BY '\\n'
159      IGNORE 1 ROWS
160 ;
```

Output			
Action	Time	Action	Message
133	12:59:57	LOAD DATA LOCAL INFILE 'C:\\\\ProgramData\\\\MySQL\\\\MySQL Server 8.0\\\\Uploads\\\\companies.csv' INTO TABLE companies FIELDS TERMINATED BY ',' ENCLOSED BY '\"' LINES TERMINATED BY '\\n' IGNORE 1 ROWS;	Error Code: 3948. Loading local data is disabled; this must be enabled on both the client and server sides
134	13:00:55	LOAD DATA INFILE 'C:\\\\ProgramData\\\\MySQL\\\\MySQL Server 8.0\\\\Uploads\\\\companies.csv' INTO TABLE companies FIELDS TERMINATED BY ',' LINES TERMINATED BY '\\n' IGNORE 1 ROWS;	100 row(s) affected Records: 100 Deleted: 0 Skipped: 0 Warnings: 0 0.047 sec

```
162      -- Cuento las filas introducidas para comprobar que tiene las mismas que el csv que son 100 sin contar la cabecera
163 • SELECT COUNT(*) AS total_records
164     FROM companies;
```

total_records
100

Output			
Action	Time	Action	Message
134	13:00:55	LOAD DATA INFILE 'C:\\\\ProgramData\\\\MySQL\\\\MySQL Server 8.0\\\\Uploads\\\\companies.csv' INTO TABLE companies FIELDS TERMINATED BY ',' LINES TERMINATED BY '\\n' IGNORE 1 ROWS;	100 row(s) affected Records: 100 Deleted: 0 Skipped: 0 Warnings: 0 0.047 sec
135	13:04:15	SELECT COUNT(*) AS total_records FROM companies	1 row(s) returned 0.016 sec / 0.000 sec

g. Creo la tabla “credit_cards” e introduzco los datos del fichero credit_cards.csv que me da el ejercicio

```

167      -- Creamos la tabla credit_cards
168 • CREATE TABLE IF NOT EXISTS credit_cards (
169      id VARCHAR(20) PRIMARY KEY,
170      user_id VARCHAR(15),
171      iban VARCHAR(50),
172      pan VARCHAR(50),
173      pin VARCHAR(4),
174      cvv VARCHAR(4),
175      track1 VARCHAR(100),
176      track2 VARCHAR(100),
177      expiring_date VARCHAR(20)
178 );

```

Output			
Action	Time	Action	Duration / Fetch
135 13:04:15	SELECT COUNT(*) AS total_records FROM companies	Message 1 row(s) returned	0.016 sec / 0.000 sec
136 13:05:54	CREATE TABLE IF NOT EXISTS credit_cards (id VARCHAR(20) PRIMARY KEY, user_id VARCHAR(15), iban VARCHAR(50), ...)	Message 0 row(s) affected	0.031 sec

```

180      -- Introduzco los datos del fichero credit_cards.csv que me da el ejercicio
181 • LOAD DATA
182      INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\credit_cards.csv'
183      INTO TABLE credit_cards
184      FIELDS TERMINATED BY ','
185      ENCLOSED BY """
186      LINES TERMINATED BY '\n'
187      IGNORE 1 ROWS
188 ;

```

Output			
Action	Time	Action	Duration / Fetch
136 13:05:54	CREATE TABLE IF NOT EXISTS credit_cards (id VARCHAR(20) PRIMARY KEY, user_id VARCHAR(15), iban VARCHAR(50), ...)	Message 0 row(s) affected	0.031 sec
137 13:06:43	LOAD DATA INFILE C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\credit_cards.csv INTO TABLE credit_cards FIELDS TERMINATE...	Message 5000 row(s) affected Records: 5000 Deleted: 0 Skipped: 0 Warnings: 0	0.125 sec

```

190      -- Cuento las filas introducidas para comprobar que tiene las mismas que el csv que son 5000 sin contar la cabecera
191 • SELECT FORMAT(COUNT(*),0) AS total_records
192     FROM credit_cards;

```

Result Grid	F
total_records	5,000

Output			
Action	Time	Action	Duration / Fetch
18 18:09:59	SELECT COUNT(*) AS total_records FROM credit_cards	Message 1 row(s) returned	0.016 sec / 0.000 sec

h. Creo la tabla “transactions” e introduzco los datos del fichero transactions.csv que me da el ejercicio

```

196      -- Creamos la tabla transactions
197 • CREATE TABLE IF NOT EXISTS transactions (
198      id VARCHAR(255) PRIMARY KEY,
199      card_id VARCHAR(20),
200      business_id VARCHAR(15),
201      timestamp TIMESTAMP,
202      amount DECIMAL(10, 2),
203      declined BOOLEAN,
204      product_ids VARCHAR(50),
205      user_id VARCHAR(10),
206      lat VARCHAR(30),
207      longitude VARCHAR(30),
208      FOREIGN KEY (card_id) REFERENCES credit_cards(id),
209      FOREIGN KEY (business_id) REFERENCES companies(company_id),
210      FOREIGN KEY (user_id) REFERENCES total_users(id)
211 );

```

```

Output
Action Output
# Time Action
138 13:08:06 SELECT COUNT() AS total_records FROM credit_cards
139 13:08:37 CREATE TABLE IF NOT EXISTS transactions ( id VARCHAR(255) PRIMARY KEY, card_id VARCHAR(20), business_id VARCHAR(1... 0 row(s) affected

Duration / Fetch
0.016 sec / 0.000 sec
0.047 sec

213 -- Introduzco los datos del fichero transactions.csv que me da el ejercicio. OJO que los datos estan separados por ; en vez de por ,
214 • LOAD DATA
215     INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\transactions.csv'
216     INTO TABLE transactions
217     FIELDS TERMINATED BY ';'
218     ENCLOSED BY '\"'
219     LINES TERMINATED BY '\\n'
220     IGNORE 1 ROWS
221 ;

```

```

Output
Action Output
# Time Action
163 14:02:48 CREATE TABLE IF NOT EXISTS transactions ( id VARCHAR(255) PRIMARY KEY, card_id VARCHAR(20), business_id VARCHAR(1... 0 row(s) affected
164 14:02:53 LOAD DATA INFILE C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\transactions.csv' INTO TABLE transactions FIELDS TERMINATED... 100000 row(s) affected Records: 100000 Deleted: 0 Skipped: 0 Warnings: 0

Duration / Fetch
0.046 sec
4.563 sec


```

223 -- Cuento las filas introducidas para comprobar que tiene las mismas que el csv que son 100,000 sin contar la cabecera
 224 • SELECT FORMAT(COUNT(*),0) AS total_records
 225 FROM transactions;

Result Grid	
total_records	100,000

```

Output
Action Output
# Time Action
16 18:06:25 SELECT FORMAT(COUNT(),2) AS total_records FROM transactions

Duration / Fetch
0.015 sec / 0.000 sec

```

Elimino las tablas “american_users” y “european_users”:

232 • DROP TABLE IF EXISTS european_users;
 233 • DESCRIBE european_users;

```

Output
Action Output
# Time Action
78 18:45:25 DROP TABLE IF EXISTS european_users
79 18:47:12 DESCRIBE european_users

Message
0 row(s) affected
Error Code: 1146. Table 'transactions_2.european_users' doesn't exist

```

236 • DROP TABLE IF EXISTS american_users;
 237 • DESCRIBE american_users;

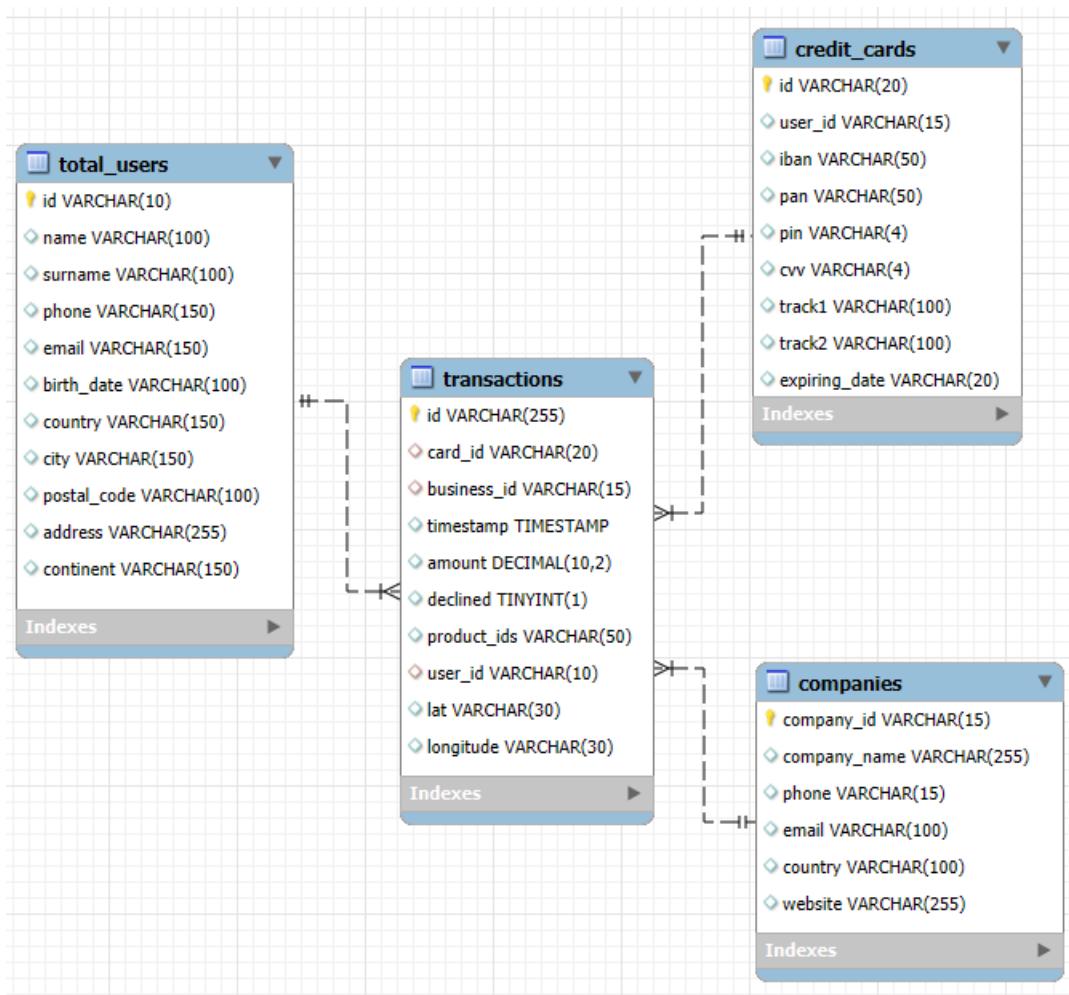
```

Output
Action Output
# Time Action
80 18:50:15 DROP TABLE IF EXISTS american_users
81 18:50:20 DESCRIBE american_users

Message
0 row(s) affected
Error Code: 1146. Table 'transactions_2.american_users' doesn't exist

```

Y llegado a este punto, tengo este modelo de BD:



Ejercicio 1

Realiza una subconsulta que muestre a todos los usuarios con más de 80 transacciones utilizando al menos 2 tablas.

Mediante subconsultas lo voy a hacer de 2 formas, la primera con una NO correlacionada que me va a mostrar únicamente los usuarios y no el nº de transacciones de cada uno:

```

246      -- Respuesta 1.1: Usando Subconsulta NO correlacionada pero no me visualiza los transactions_number de cada user
247 •   SELECT
248     tu.id AS user_id,
249     tu.name,
250     tu.surname
251   FROM total_users AS tu
252   WHERE EXISTS (
253     SELECT t.user_id
254     FROM transactions AS t
255     WHERE tu.id = t.user_id
256     GROUP BY t.user_id
257     HAVING COUNT(*) > 80);

```

Result Grid			
	user_id	name	surname
▶	185	Molly	Gilliam
	289	Dxwgi	Hwcru
	318	Bnyr	Astuw
	454	Sfzzoh	Xgvfridxs

Output			
Action Output			
#	Time	Action	
①	81 18:50:20	DESCRIBE american_users	Message Error Code: 1146. Table 'transactions_2.american_users' doesn't exist 0.000 sec
②	82 18:54:26	SELECT tu.id AS user_id, tu.name, tu.surname FROM total_users AS tu WHERE EXISTS (SELECT t.user_id FROM transactions AS t WHERE t.user_id = tu.id) AND t.transactions_number > 80 ORDER BY transactions_number DESC;	4 row(s) returned 0.093 sec / 0.000 sec

Y la segunda con una correlacionada que me va a mostrar también, además de los usuarios, el nº de transacciones de cada uno:

```

262      -- Respuesta 1.2: Usando Subconsulta correlacionada para que me visualice tambien transactions_number
263 •   SELECT tu.id AS user_id, tu.name, tu.surname,
264      (
265          SELECT COUNT(*)
266          FROM transactions t
267          WHERE t.user_id = tu.id
268      ) AS transactions_number
269      FROM total_users AS tu
270      WHERE (
271          SELECT COUNT(*)
272          FROM transactions t
273          WHERE t.user_id = tu.id
274      ) > 80
275      ORDER BY transactions_number DESC;
276 ;

```

Result Grid				
	user_id	name	surname	transactions_number
▶	185	Molly	Gilliam	110
	289	Dxwgi	Hwcru	94
	318	Bnyr	Astuw	91
	454	Sfzzoh	Xgvfridxs	81

Output			
Action Output			
#	Time	Action	
①	82 18:54:26	SELECT tu.id AS user_id, tu.name, tu.surname FROM total_users AS tu WHERE EXISTS (SELECT t.user_id FROM transactions AS t WHERE t.user_id = tu.id) AND t.transactions_number > 80 ORDER BY transactions_number DESC;	4 row(s) returned 0.093 sec / 0.000 sec
②	83 18:56:51	SELECT tu.id AS user_id, tu.name, tu.surname, (SELECT COUNT(*) FROM transactions t WHERE t.user_id = tu.id) AS transactions_number	4 row(s) returned 0.047 sec / 0.000 sec

Y por último no usando subconsultas y usando JOIN. En la tabla “transaction” voy a contar las transacciones de cada id de usuario, pero como dice utilizar 2 tablas, hago un JOIN con la tabla “total_users” y de ahí obtengo el id, el nombre y el apellido. Los visualizo en orden descendente de número de transacciones.

```

243 •   SELECT tu.id AS user_id , tu.name AS name, tu.surname AS surname, COUNT(t.user_id) AS transactions_number
244     FROM total_users AS tu
245     JOIN transactions AS t
246     ON tu.id = t.user_id
247     GROUP BY user_id
248     HAVING transactions_number > 80
249     ORDER BY transactions_number DESC
250 ;

```

Result Grid | Filter Rows: | Export

	user_id	name	surname	transactions_number
▶	185	Molly	Gilliam	110
	289	Dxwgi	Hwcru	94
	318	Bnyr	Astuw	91
	454	Sfzzoh	Xgfridxs	81

Output

Action Output

Time Action Message Duration / Fetch

42 18:50:40 SELECT tu.id AS user_id , tu.name AS name , tu.surname AS surname , COUNT(t.user_id) AS transactions_number FROM total_users AS tu JOIN trans... 4 row(s) returned 0.109 sec / 0.000 sec

43 18:50:48 SELECT tu.id AS user_id , tu.name AS name , tu.surname AS surname , COUNT(t.user_id) AS transactions_number FROM total_users AS tu JOIN trans... 4 row(s) returned 0.109 sec / 0.000 sec

Ejercicio 2

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

Calculo el average del amount que esta en la tabla “transactions” agrupado por el iban que esta en la tabla “credit_cards” y por los nombres de compañía que están en la tabla “companies”, por eso uso las uniones de tablas. Importante ver que las medias las realiza bien a pesar de que el “amount” es del tipo VARCHAR, pero a la hora de ordenar los resultados, como son VARCHAR pues los ordena como si fuese string, por lo que tengo que usar la función CAST para cambiar de string a decimal y así ya me hace correctamente el ordenamiento, que en este caso lo pongo descendente para ver primero los iban que mas han gastado.

```

256 •   SELECT c.company_name AS company_name, cc.iban AS iban, ROUND(AVG(t.amount),2) AS average_amount
257   FROM credit_cards AS cc
258   JOIN transactions AS t
259   ON cc.id = t.card_id
260   JOIN companies AS c
261   ON t.business_id = c.company_id
262   GROUP BY company_name, iban
263   HAVING company_name = "Donec Ltd"
264   ORDER BY average_amount DESC
265 ;

```

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

	company_name	iban	average_amount
▶	Donec Ltd	XX383017813919620199366352	680.69
	Donec Ltd	XX637706357397570394973913	680.01
	Donec Ltd	XX971393971465292202312259	645.46
	Donec Ltd	XX171847116928892375969307	628.89
	Donec Ltd	XX225424638818542406223575	608.68
	Donec Ltd	XX748890729057195711766071	607.29
	Donec Ltd	TN9614563570667381893122	605.41
	Donec Ltd	XX481908034037364242591185	605.36
	Donec Ltd	XX194675519739256335753508	597.19
	Donec Ltd	XX215962766061967195493437	594.26
	Donec Ltd	XX449322320826890721001443	591.61
	Donec Ltd	XX535185492735704229474237	570.09
	Donec Ltd	CH9552373968796160224	566.38
	Donec Ltd	XX347605377125637880303131	561.80
	Donec Ltd	XX688471446697921912860304	543.42
	Donec Ltd	XX605533964582458704105956	542.00
	Donec Ltd	PL76249283566852676343404576	541.56
	Donec Ltd	XX258862585706063154381130	539.81
	Donec Ltd	XX651270526010893179119477	535.59
	Donec Ltd	CR2918135947128138635	535.11

Output			Message	Duration / Fetch
#	Time	Action		
17	11:49:09	SELECT c.company_name AS company_name, cc.iban AS iban, FORMAT(AVG(cc.amount),2) AS average_amount FROM credit_cards AS cc JOIN trans AS t ON cc.card_id = t.card_id WHERE t.timestamp > DATE_SUB(NOW(), INTERVAL 3 DAY) GROUP BY cc.card_id, cc.iban;	371 row(s) returned	1.187 sec / 0.000 sec
18	11:50:15	SELECT c.company_name AS company_name, cc.iban AS iban, ROUND(AVG(cc.amount),2) AS average_amount FROM credit_cards AS cc JOIN trans AS t ON cc.card_id = t.card_id WHERE t.timestamp > DATE_SUB(NOW(), INTERVAL 3 DAY) GROUP BY cc.card_id, cc.iban;	371 row(s) returned	1.141 sec / 0.000 sec

NIVEL 2

Crea una nueva tabla que refleje el estado de las tarjetas de crédito basado en *si las tres últimas transacciones han sido declinadas entonces es inactivo, si al menos una no es rechazada entonces es activo*. Partiendo de esta tabla responde:

- Primero voy a hacer la consulta paso a paso utilizando los filtros de ventana hasta llegar a una visualización que me de 2 columnas, una con la id de la tarjeta y la otra con su estado. En el fichero sql están todos los pasos hasta llegar a el query final, aquí solo pongo los printados del query final, que es:

```

323      -- SOLUCION FINAL del QUERY ---
324  •  SELECT sbn.card_id,
325      CASE
326          WHEN sbn.sum_3_last = 3 THEN "inactive"
327          ELSE "active"
328      END AS state
329  •  FROM (
330      SELECT rnk.card_id, SUM(rnk.declined) AS sum_3_last
331      FROM (SELECT
332          t.card_id,
333          t.timestamp,
334          t.declined,
335          ROW_NUMBER() OVER (PARTITION BY card_id ORDER BY timestamp DESC) AS rn
336          FROM transactions AS t
337      ) AS rnk
338      WHERE rnk.rn <= 3
339      GROUP BY rnk.card_id
340      ORDER BY rnk.card_id DESC
341  ) AS sbn
342 ;

```

Result Grid | Filter Rows:

	card_id	state
▶	CcU-4856	active
	CcU-4849	active
	CcU-4842	active
	CcU-4835	active
	CcU-4828	active
	CcU-4821	active
	CcU-4814	active
	CcU-4807	active
	CcU-4800	active
	CcU-4793	active
	CcU-4786	active
	CcU-4779	active
	CcU-4772	active

Output			
Action Output	Time	Action	Message
95 12:59:45	Time	DROP TABLE `transactions_2`.'credit_state'	0 row(s) affected
96 13:02:48	Time	SELECT sbn.card_id, CASE WHEN sbn.sum_3_last = 3 THEN "inactive" ELSE "active" END AS state FROM (SELECT rnk.card_id, SUM(rnk.declined) AS sum_3...) AS sbn	5000 row(s) returned

2. Creo la estructura de la tabla con sus 2 columnas o campos. No la relaciono con ninguna otra tabla de la BD porque el ejercicio no me lo pide.

```
280      -- Creo la estructura de la tabla
281 • CREATE TABLE IF NOT EXISTS credit_state(
282     card_id VARCHAR(20) PRIMARY KEY,
283     card_state VARCHAR(10)
284 );
```

Output			
Action Output	Time	Action	Message
96 13:02:48	Time	SELECT sbn.card_id, CASE WHEN sbn.sum_3_last = 3 THEN "inactive" ELSE "active" END AS state FROM (SELECT rnk.card_id, SUM(rnk.declined) AS sum_3...) AS sbn	5000 row(s) returned
97 13:05:56	Time	CREATE TABLE IF NOT EXISTS credit_state(card_id VARCHAR(20) PRIMARY KEY, card_state VARCHAR(10))	0 row(s) affected

3. Compruebo que se ha creado correctamente y esta vacía

```
287      -- compruebo que se ha creado correctamente y esta vacia
288 •  SELECT *
289   FROM credit_state AS CS;
```

Result Grid		Filter Rows:
	card_id	card_state
*	NULL	NULL

Output			
Action Output	Time	Action	Message
97 13:05:56	Time	CREATE TABLE IF NOT EXISTS credit_state(card_id VARCHAR(20) PRIMARY KEY, card_state VARCHAR(10))	0 row(s) affected
98 13:07:31	Time	SELECT * FROM credit_state AS CS	0 row(s) returned

4. Cargo los datos en la tabla, usando la consulta final creada en el punto 1 anterior:

```
293 • INSERT INTO credit_state (
294           SELECT sbn.card_id,
295             CASE
296               WHEN sbn.sum_3_last = 3 THEN "inactive"
297               ELSE "active"
298             END AS state
299           FROM (
300             SELECT rnk.card_id, SUM(rnk.declined) AS sum_3_last
301             FROM (SELECT
302                   t.card_id,
303                   t.timestamp,
304                   t.declined,
305                   ROW_NUMBER() OVER (PARTITION BY card_id ORDER BY timestamp DESC) AS rn
306                 FROM transactions AS t
307               ) AS rnk
308               WHERE rnk.rn <= 3
309               GROUP BY rnk.card_id
310               ORDER BY rnk.card_id DESC
311             ) AS sbn
312           )
313 ;
```

Output			
Action Output	Time	Action	Message
98 13:07:31 SELECT * FROM credit_state AS CS			0 row(s) returned
99 13:11:02 INSERT INTO credit_state (SELECT sbn.card_id, CASE WHEN sbn.sum_3_last = 3 THEN "inactive" ELSE "active" END AS state FROM (SELECT s...			5000 row(s) affected Records: 5000 Duplicates: 0 Warnings: 0

5. Por último, compruebo que se han cargado los datos y hay 5.000 registros

```
315      -- compruebo que se han cargado los datos. Debe haber 5.000 filas
316 •  SELECT *
317      FROM credit_state AS CS;
```

Result Grid | Filter Rows:

card_id	card_state
CcS-4857	active
CcS-4858	active
CcS-4859	active
CcS-4860	active
CcS-4861	active
CcS-4862	active
CcS-4863	active
CcS-4864	active
CcS-4865	active
CcS-4866	active
CcS-4867	active
CcS-4868	active
CcS-4869	active

Output			
Action Output	Time	Action	Message
99 13:11:02 INSERT INTO credit_state (SELECT sbn.card_id, CASE WHEN sbn.sum_3_last = 3 THEN "inactive" ELSE "active" END AS state FROM (SELECT s...			5000 row(s) affected Records: 5000 Duplicates: 0 Warnings: 0
100 13:12:47 SELECT * FROM credit_state AS CS			5000 row(s) returned

Ejercicio 1

¿Cuántas tarjetas están activas?

```
410 •  SELECT cs.card_state, COUNT(cs.card_state) AS quantity
411      FROM credit_state AS cs
412      GROUP BY cs.card_state
413      HAVING cs.card_state = "active"
414      ;
```

Result Grid | Filter Row

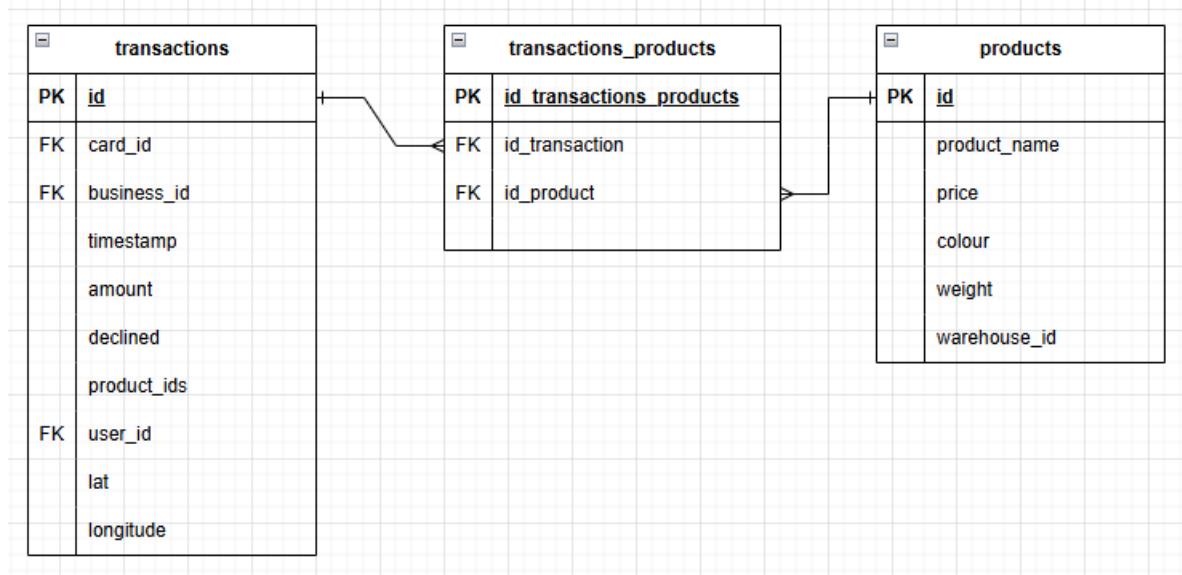
card_state	quantity
active	4995

Output			
Action Output	Time	Action	Message
100 13:12:47 SELECT * FROM credit_state AS CS			5000 row(s) returned
101 13:16:11 SELECT cs.card_state, COUNT(cs.card_state) AS quantity FROM credit_state AS cs GROUP BY cs.card_state HAVING cs.card_state = "active"			1 row(s) returned

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:

La idea es crear una tabla intermedia al tener una relación de N:M entre transacciones y productos, así que voy a crear la siguiente estructura con una tabla intermedia llamada “transactions_products”:



1. Creo la estructura de la tabla “products”

```
432      -- Creo la tabla products
433 • CREATE TABLE IF NOT EXISTS products (
434     id INT PRIMARY KEY,
435     product_name VARCHAR(100),
436     price VARCHAR(10),
437     colour VARCHAR(15),
438     weight VARCHAR(10),
439     warehouse_id VARCHAR(10)
440 );
```

Output		Action Output	Time	Action	Message	Duration / Fetch
•	10	11:04:57	DROP TABLE `transactions_2`;	products'	0 row(s) affected	0.031 sec
•	11	11:05:09	CREATE TABLE IF NOT EXISTS products (id INT PRIMARY KEY, product_name VARCHAR(100), price VARCHAR(10), ...	0 row(s) affected	0.031 sec

2. Cargo los datos de los productos del fichero products.csv en la estructura creada anteriormente

```

443      -- Introduzco los datos del fichero products.csv que me da el ejercicio
444  • LOAD DATA
445      INFILE 'C:\\\\ProgramData\\\\MySQL\\\\MySQL Server 8.0\\\\Uploads\\\\products.csv'
446      INTO TABLE products
447      FIELDS TERMINATED BY ','
448      ENCLOSED BY '\"'
449      LINES TERMINATED BY '\\n'
450      IGNORE 1 ROWS
451  ;

```

Output		Action Output	Time	Action	Message	Duration / Fetch
#	Time	Action				
12	11:06:00	SELECT * FROM products			0 row(s) returned	0.000 sec / 0.000 sec
13	11:08:15	LOAD DATA INFILE 'C:\\\\ProgramData\\\\MySQL\\\\MySQL Server 8.0\\\\Uploads\\\\products.csv' INTO TABLE products FIELDS TERMINATED BY ',' ...			100 row(s) affected Records: 100 Deleted: 0 Skipped: 0 Warnings: 0	0.047 sec

3. Compruebo que se han cargado correctamente

```

454  • SELECT *
455      FROM products;

```

Result Grid		Filter Rows:	Edit:	Export/Import		
	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	#dbdbdb	3.2	WH-0
	6	dooku solo	\$136.60	#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	\$76.40	#b5b5b5	1.2	WH-4
	10	Karstark Dorne	\$119.52	#f4f4f4	2.4	WH-5
	11	Karstark Dorne	\$49.70	#141414	2.7	WH-6
	12	duel Direwolf	\$181.60	#a8a8a8	2.1	WH-7
	13	palpatine chewbacca	\$139.59	#2b2b2b	1	WH-8
	14	Direwulf	\$147.53	#r4r4r4	?	WH--9

Output		Action Output	Time	Action	Message	Duration / Fetch
#	Time	Action				
14	11:09:22	SELECT * FROM transactions_products			Error Code: 1146. Table 'transactions_products' doesn't exist	0.016 sec
15	11:09:35	SELECT * FROM products			100 row(s) returned	0.000 sec / 0.000 sec

4. Creo la tabla intermedia entre “transactions” y “products” que llamo “transactions_products”

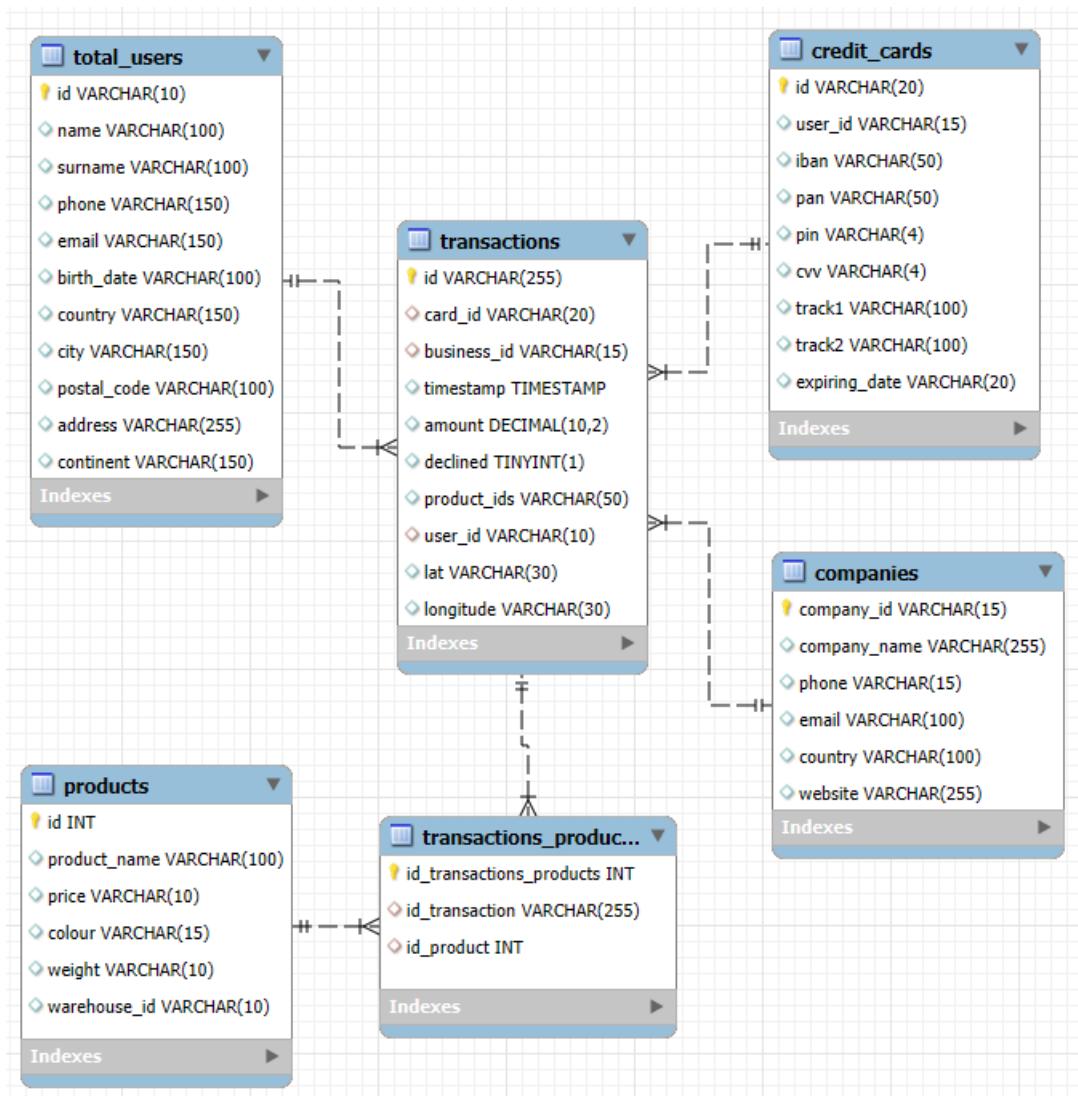
```

458      -- Creo la tabla intermedia para la relacion N:M llamada transactions_products
459  • CREATE TABLE IF NOT EXISTS transactions_products (
460      id_transactions_products INT AUTO_INCREMENT PRIMARY KEY,
461      id_transaction VARCHAR(255),
462      id_product INT,
463      FOREIGN KEY(id_product) REFERENCES products(id),
464      FOREIGN KEY(id_transaction) REFERENCES transactions(id)
465  );

```

Output		Action Output	Time	Action	Message	Duration / Fetch
#	Time	Action				
15	11:09:35	SELECT * FROM products			100 row(s) returned	0.000 sec / 0.000 sec
16	11:11:05	CREATE TABLE IF NOT EXISTS transactions_products (id_transactions_products INT AUTO_INCREMENT PRIMARY KEY,id_transaction VARCH...)			0 row(s) affected	0.047 sec

5. El modelo que tengo ahora sería este:



6. Ahora tengo que cargar los datos en la tabla “transactions_products” pero para eso tengo que resolver el problema de que en el campo product_ids de la tabla “transactions” tengo varios productos, es decir:

Tengo los id de productos almacenados así en la tabla “transactions”:

id (transaction_id) product_ids

T1 75, 73, 98

T2 3

Y el objetivo es almacenarlos así en la tabla intermedia llamada “transactions_products”:

Id_transaction Id_product

T1 75

T1 73

Id_transaction Id_product

T1	98
T2	3

Voy a utilizar la función de mesa JSON. Para ello voy a tomar el texto product_ids de cada transacción, conviértelo en un array JSON, y generar una tabla temporal (j) con una columna product_id que contiene cada valor del array como una fila. Los pasos del ejemplo anterior serán:

(inicio) 75, 73, 98 -> (json) ["75","73","98"] -> (filas):

75
73
98

(inicio) 3 -> (json) ["3"] -> (filas):

3

1. Compruebo que la tabla “transactions_products” no tiene ningún registro

```
468 •  SELECT *  
469   FROM transactions_products;
```

id_transactions_products	id_transaction	id_product
NULL	NULL	NULL

Output

#	Time	Action	Message	Duration / Fetch
8	20:22:13	SELECT * FROM transactions_products	0 row(s) returned	0.000 sec / 0.000 sec
9	20:23:07	SELECT * FROM transactions_products	0 row(s) returned	0.000 sec / 0.000 sec

2. Ejecuto la transformación a JSON y el llenado de la tabla “transactions_products”

```
473 -- pasar el campo product_ids de la tabla transactions a almacenado como JSON  
474 • INSERT INTO transactions_products (id_transaction, id_product) -- inserta los 2 campos en la tabla `transactions_products`  
475   SELECT  
476     t.id AS id_transaction, -- el indice de la tabla transactions sera el mismo que pondre en la tabla `transactions_products` en el campo id_transaction  
477     CAST(j.id_product AS UNSIGNED) AS id_product -- convierte el valor formato texto en formato INT sin signo, eliminando los espacios y asiga alias id_product  
478   FROM transactions AS t  
479   JOIN JSON_TABLE( -- genera una fila por cada valor  
480     CONCAT([",", REPLACE(t.product_ids, ',', '",')], "") -- pongo cada valor de product_ids en formato de 75,73, 98 a '["75", "73", "98"]'  
481   ),  
482   '$[*]' COLUMNS (id_product VARCHAR(10) PATH '$') -- Recorre todos los elementos JSON y crea una fila por cada uno  
483   ) AS j;
```

Output

#	Time	Action	Message	Duration / Fetch
9	20:23:07	SELECT * FROM transactions_products	0 row(s) returned	0.000 sec / 0.000 sec
10	20:25:25	INSERT INTO transactions_products (id_transaction, id_product) -- inserta los 2 campos en la tabla `transactions_products` SELECT t.id AS id_tran... 253391 row(s) affected Records: 253391 Duplicates: 0 Warnings: 0	253391 row(s) affected Records: 253391 Duplicates: 0 Warnings: 0	6.188 sec

3. Compruebo que la tabla “transactions_products” Ahora ya tiene registros

```
489 •  SELECT *
490   FROM transactions_products;
...
```

Result Grid | Filter Rows: | Edit: | Export/Import: |

	id_transactions_products	id_transaction	id_product
▶	1	00043A49-2949-494B-A5DD-A5BAE3BB19DD	16
2		00043A49-2949-494B-A5DD-A5BAE3BB19DD	26
3		00043A49-2949-494B-A5DD-A5BAE3BB19DD	97
4		00043A49-2949-494B-A5DD-A5BAE3BB19DD	87
5		000447FE-B650-4DCF-85DE-C7ED0EE1CAAD	66
6		000447FE-B650-4DCF-85DE-C7ED0EE1CAAD	69
7		000447FE-B650-4DCF-85DE-C7ED0EE1CAAD	87
8		00045D6F-FD2F-4F2F-8186-CFF074D875D0	30

Output | Action Output |

#	Time	Action	Message	Duration / Fetch
10	20:25:29	INSERT INTO transactions_products (id_transaction, id_product) -- inserta los 2 campos en la tabla 'transactions_products' SELECT 1.id AS id_tran...	253391 row(s) affected Records: 253391 Duplicates: 0 Warnings: 0	6.189 sec
11	20:27:27	SELECT * FROM transactions_products	253391 row(s) returned	0.015 sec / 0.110 sec

4. Voy a comprobar un índice en la tabla "transactions" se ha trasformado bien en una fila por valor en la tabla "transactions_products"

```
493   -- Compruebo que se ha trasformado bien un indice concreto de la tabla "transactions"
494 •  SELECT *
495   FROM transactions
496   WHERE id = "29322BF7-84F4-4A6C-9FFC-C60B6BBD8DCD";
...
```

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

	id	card_id	business_id	timestamp	amount	declined	product_ids	user_id	lat	longitude
▶	29322BF7-84F4-4A6C-9FFC-C60B6BBD8DCD	Ccs-5036	b-2222	2022-05-19 04:50:50	678.61	0	6, 12, 100, 22, 23	455	51.60907736775525	19.160038756363225
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Output | Action Output |

#	Time	Action	Message	Duration / Fetch
15	20:30:12	SELECT * FROM transactions_products WHERE id_transaction = "29322BF7-84F4-4A6C-9FFC-C60B6BBD8DCD"	5 row(s) returned	0.000 sec / 0.000 sec
16	20:34:24	SELECT * FROM transactions WHERE id = "29322BF7-84F4-4A6C-9FFC-C60B6BBD8DCD"	1 row(s) returned	0.000 sec / 0.000 sec

```
498   -- Compruebo que se ha trasformado bien un indice concreto de la tabla "transactions" en la "transactions_products"
499 •  SELECT *
500   FROM transactions_products
501   WHERE id_transaction = "29322BF7-84F4-4A6C-9FFC-C60B6BBD8DCD";
```

Result Grid | Filter Rows: | Edit: | Export/Import: |

	id_transactions_products	id_transaction	id_product
▶	41105	29322BF7-84F4-4A6C-9FFC-C60B6BBD8DCD	6
41106		29322BF7-84F4-4A6C-9FFC-C60B6BBD8DCD	12
41107		29322BF7-84F4-4A6C-9FFC-C60B6BBD8DCD	100
41108		29322BF7-84F4-4A6C-9FFC-C60B6BBD8DCD	22
41109		29322BF7-84F4-4A6C-9FFC-C60B6BBD8DCD	23
*	NULL	NULL	NULL

Output | Action Output |

#	Time	Action	Message	Duration / Fetch
16	20:34:24	SELECT * FROM transactions WHERE id = "29322BF7-84F4-4A6C-9FFC-C60B6BBD8DCD"	1 row(s) returned	0.000 sec / 0.000 sec
17	20:36:06	SELECT * FROM transactions_products WHERE id_transaction = "29322BF7-84F4-4A6C-9FFC-C60B6BBD8DCD"	5 row(s) returned	0.000 sec / 0.000 sec

Ejercicio 1

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

Cuento las unidades vendidas de cada producto que hay en la tabla intermedia “transactions_products” y las ordeno en orden descendente de los productos más vendidos.

Uso FORMAT para poner el delimitador de miles

```
509      -- Contar las unidades vendidas de cada producto
510 •  SELECT p.product_name AS product_name, tp.id_product AS product_id, FORMAT(COUNT(tp.id_product),0) AS units_sold
511     FROM transactions_products AS tp
512     JOIN products AS p
513     ON tp.id_product = p.id
514     GROUP BY product_name, product_id
515     ORDER BY units_sold DESC
516 ;
```

Result Grid | Filter Rows:

	product_name	product_id	units_sold
▶	riverlands the duel	52	2,654
	Tully maester Tarly	29	2,635
	duel Direwolf	21	2,609
	the duel warden	16	2,608
	mustafar jinn	66	2,601
	sith Jade	87	2,598
	duel warden	33	2,597
	rock Danlys in	48	2,507

Output : Action Output

#	Time	Action	Message	Duration / Fetch
30	20:51:31	SELECT p.product_name AS product_name, tp.id_product AS product_id, FORMAT(COUNT(tp.id_product),0) AS units_sold FROM transactions_products AS tp JOIN products AS p ON tp.id_product = p.id GROUP BY product_name, product_id ORDER BY units_sold DESC;	100 row(s) returned	0.531 sec / 0.000 sec
31	20:52:57	SELECT p.product_name AS product_name, tp.id_product AS product_id, FORMAT(COUNT(tp.id_product),0) AS units_sold FROM transactions_products AS tp JOIN products AS p ON tp.id_product = p.id GROUP BY product_name, product_id ORDER BY units_sold DESC;	100 row(s) returned	0.453 sec / 0.000 sec