

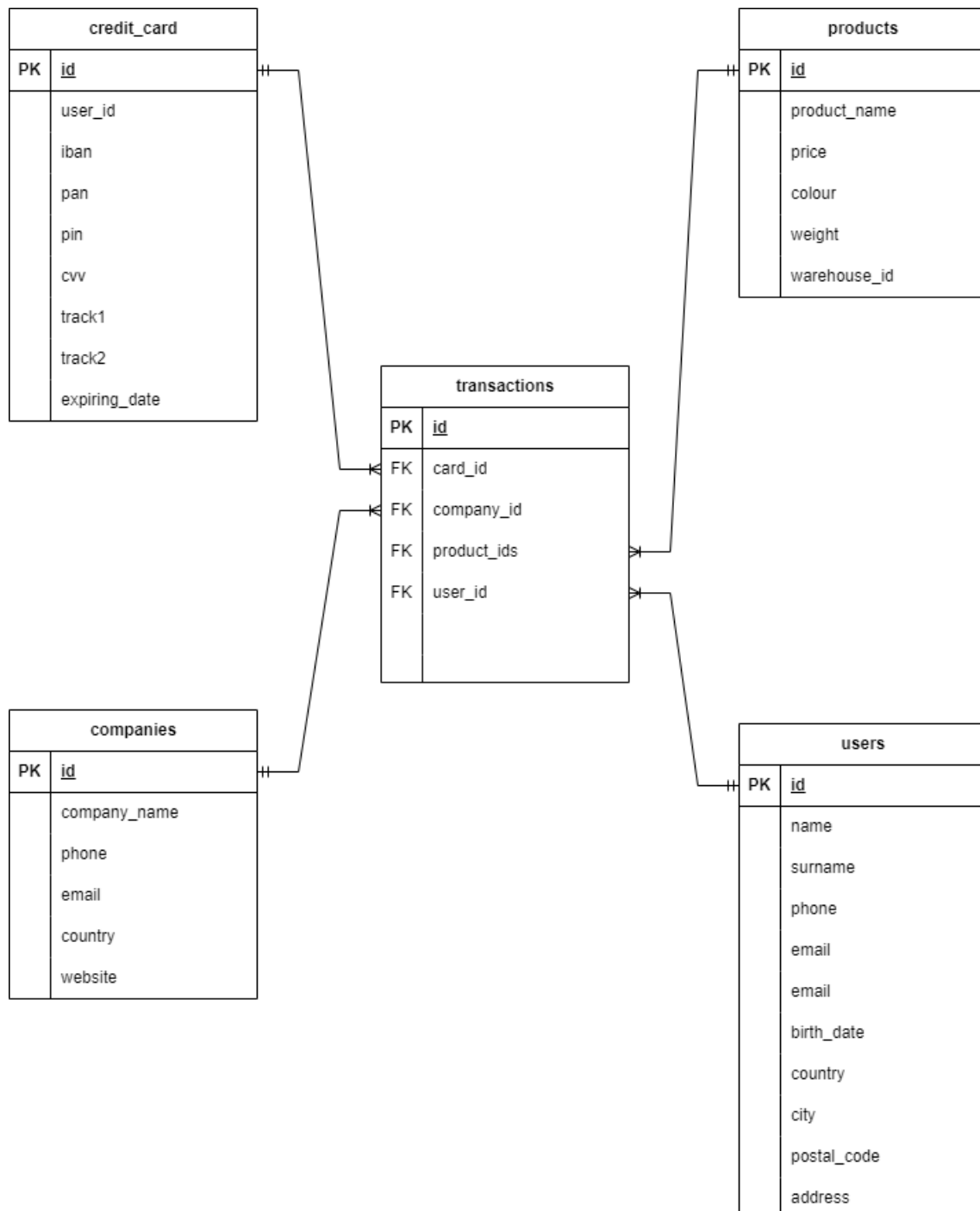
Partint d'alguns arxius CSV dissenyaràs i crearàs la teva base de dades.

### Nivell 1

Descàrrega els arxius CSV, estudia'ls i dissenya una base de dades amb un esquema d'estrella que contingui, almenys 4 taules de les quals puguis realitzar les següents consultes:

---

Abans de procedir amb els exercicis hem procedit a dissenyar la base de dades en esquema que se'ns sol·licita. Afegim captura del disseny d'aquesta:



A posteriori, hem procedit a crear la base de dades i malgrat que en un primer moment, hem importat les dades de cadascuna de les taules amb Data Import Wizard, procedeix a incorporar el format per carregar les dades directament des de l'ordinador amb LOAD INFILE.

```
8 # En primer lloc crearem la nova base de dades i procedirem a seleccionar-la.
9 • CREATE DATABASE ecommerce;
10 • use ecommerce;
11
12 #Un cop fet això, crearem les diferents taules i carregarem les dades d'aquestes amb Data Import Wizard.
13
14 • CREATE TABLE IF NOT EXISTS transactions (
15     id VARCHAR(50) PRIMARY KEY,
16     card_id VARCHAR(15),
17     company_id VARCHAR(15),
18     timestamp VARCHAR(30),
19     amount decimal(20,2),
20     declined boolean,
21     product_ids VARCHAR(30),
22     user_id INT,
23     lat VARCHAR(30),
24     longitude VARCHAR(30)
25 );
```

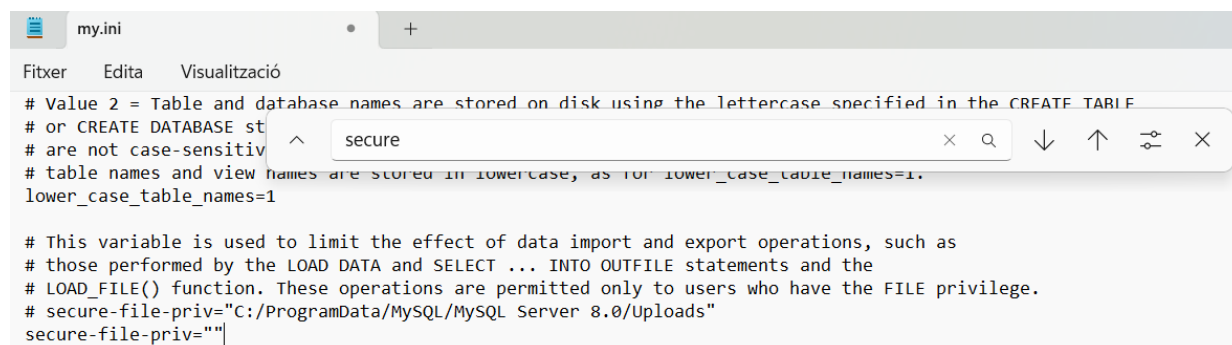
En primer lloc, hem comprovat els privilegis per accedir als arxius que volem vincular amb MySQL i per això hem utilitzat: SELECT @@secure\_file\_priv;

```
30
31 #En primer lloc, haurem de comprovar els permisos de privilegi des d'on es pot carregar l'arxiu i modificar l'arxiu my.ini
32 • SELECT @@secure_file_priv;
33
```

Com que la resposta que ens donava Workbench era la següent carpeta:

C:/ProgramData/MySQL/MySQL Server 8.0/Uploads

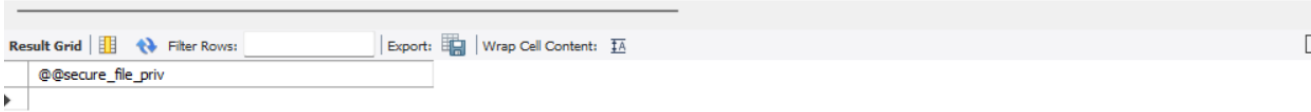
Hem modificat l'arxiu my.ini ubicat a la carpeta MySQL Server 8.0 de la següent manera:



Modificant l'accés a secure-file-priv, permetem que es puguin carregar dades des de qualsevol carpeta de l'ordinador.

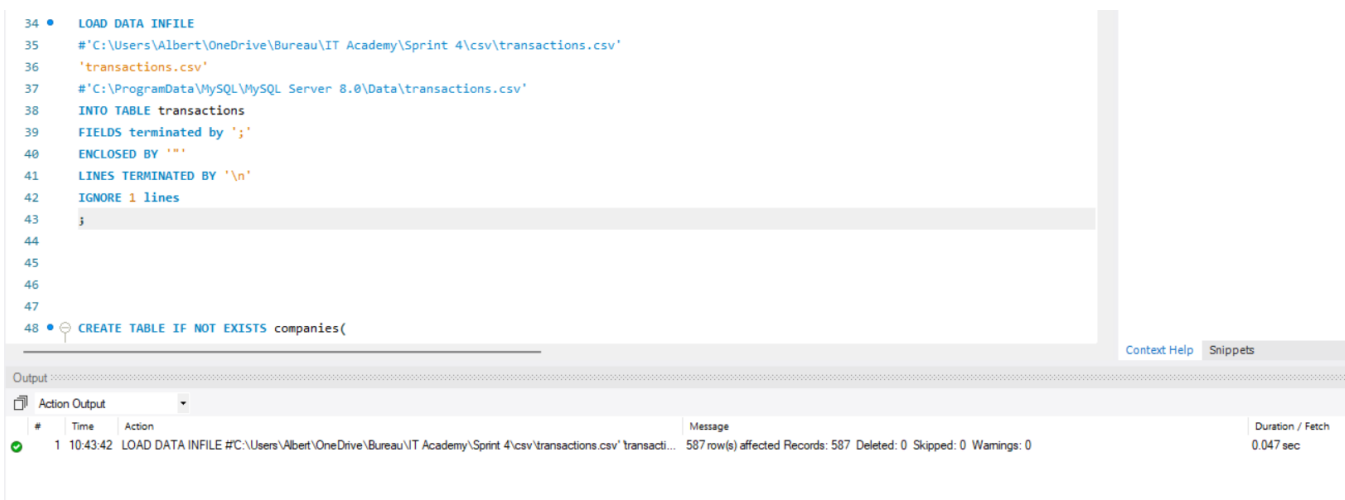
Un cop feta aquesta modificació, veiem que l'arxiu ja es pot carregar des de qualsevol carpeta de l'ordinador.

```
31 #En primer lloc, haurem de comprovar els permisos de privilegi des d'on es pot carregar l'arxiu i modificar l'arxiu my.ini
32 • SELECT @@secure_file_priv;
33
34 • LOAD DATA INFILE
35 # 'C:\Users\Albert\OneDrive\Bureau\IT Academy\Sprint 4\csv\transactions.csv'
```



Liberada aquesta possibilitat ja podem procedir a importar les dades a la nostra taula amb la següent ordre:

```
34 • LOAD DATA INFILE
35 # 'C:\Users\Albert\OneDrive\Bureau\IT Academy\Sprint 4\csv\transactions.csv'
36 'transactions.csv'
37 # 'C:\ProgramData\MySQL\MySQL Server 8.0\Data\transactions.csv'
38 INTO TABLE transactions
39 FIELDS terminated by ';'
40 ENCLOSED BY ''
41 LINES TERMINATED BY '\n'
42 IGNORE 1 lines
43 ;
```



#	Time	Action	Message	Duration / Fetch
1	10:43:42	LOAD DATA INFILE # 'C:\Users\Albert\OneDrive\Bureau\IT Academy\Sprint 4\csv\transactions.csv' transacti...	587 row(s) affected Records: 587 Deleted: 0 Skipped: 0 Warnings: 0	0.047 sec

Procedim doncs a crear les diferents taules i a carregar les dades de cadascuna d'elles.

```
45
46 • CREATE TABLE IF NOT EXISTS companies(
47     id VARCHAR(15) PRIMARY KEY,
48     company_name VARCHAR(50),
49     phone VARCHAR(15),
50     email VARCHAR(50),
51     country VARCHAR(35),
52     website VARCHAR(50)
53 );
54
55 • LOAD DATA INFILE
56 'companies.csv'
57 INTO TABLE companies
58 FIELDS terminated by ','
59 ENCLOSED BY ''
60 LINES TERMINATED BY '\n'
61 IGNORE 1 lines
62 ;
63
```

```

63
64 • CREATE TABLE IF NOT EXISTS products(
65     id INT PRIMARY KEY,
66     product_name VARCHAR(40),
67     price varchar(15),
68     colour VARCHAR(20),
69     weight decimal(10,2),
70     warehouse_id VARCHAR(15)
71 );
72
73 • LOAD DATA INFILE
74     'products.csv'
75 INTO TABLE products
76 FIELDS terminated by ','
77 ENCLOSED BY '"'
78 LINES TERMINATED BY '\n'
79 IGNORE 1 lines
80 ;
81
82 • CREATE TABLE IF NOT EXISTS credit_cards(
83     id VARCHAR(15) PRIMARY KEY,
84     user_id INT,
85     iban VARCHAR(50),
86     pan VARCHAR(30),
87     pint INT(4),
88     cvv int(3),
89     track1 VARCHAR(55),
90     track2 VARCHAR(55),
91     expiring_date VARCHAR(15)
92 );
93
94 • LOAD DATA INFILE
95     'credit_cards.csv'
96 INTO TABLE credit_cards
97 FIELDS terminated by ','
98 ENCLOSED BY '"'
99 LINES TERMINATED BY '\n'
100 IGNORE 1 lines
101 ;
102

```

```

103 • CREATE TABLE IF NOT EXISTS users(
104     id int primary key,
105     name varchar(55),
106     surname varchar(55),
107     phone varchar(30),
108     email varchar(55),
109     birth_date varchar(30),
110     country varchar(30),
111     city varchar(50),
112     postal_code varchar(25),
113     address varchar(
114         55)
115 );
116
117 • LOAD DATA INFILE
118     'users_usa.csv'
119     INTO TABLE users
120     FIELDS terminated by ','
121     ENCLOSED BY '"' -- OPTIONALLY ENCLOSED BY '"'
122     LINES TERMINATED BY '\r\n'
123     IGNORE 1 lines
124 ;
125
126 • LOAD DATA INFILE
127     'users_uk.csv'
128     INTO TABLE users
129     FIELDS terminated by ','
130     ENCLOSED BY '"'
131     LINES TERMINATED BY '\r\n'
132     IGNORE 1 lines
133 ;
134
135 • LOAD DATA INFILE
136     'users_ca.csv'
137     INTO TABLE users
138     FIELDS terminated by ','
139     ENCLOSED BY '"'
140     LINES TERMINATED BY '\r\n'
141     IGNORE 1 lines
142 ;
143

```

Un cop creades les diferents taules i volcadeu les dades de cadascuna d'elles procedim a crear les relacions entre les diferents entitats.

Primer crearem els índexs a la taula de fets, *transactions*.

```

77 #Un cop introduïdes les dades de les diferents taules, procedirem a crear les relacions entre les diferents entitats.
78 #En primer lloc, crearem els índexs de la taula de Fets.
79
80 • show create table transactions;
81 • show create table users;
82
83 • ALTER TABLE transactions
84   ADD index (card_id),
85   ADD index (company_id),
86   ADD index (user_id),
87   ADD index (product_ids);
88

```

I després afegim la vinculació amb les taules de dimensions.

```

--
89 #I a posteriori, afegirem la vinculació amb les diferents taules de dimensions.
90
91 • ALTER TABLE credit_cards
92   ADD foreign key (id) REFERENCES transactions(card_id);
93
94 • ALTER TABLE companies
95   ADD FOREIGN KEY (id) REFERENCES transactions(company_id);
96
97
98 #Al crear la vinculació amb entre la taula users i transactions ens salta un error 1452, i per això desactivarem
99 #momentàniament les claus forànies.
100
101 • SET FOREIGN_KEY_CHECKS=0;
102
103 • ALTER TABLE users
104   ADD FOREIGN KEY (id) REFERENCES transactions(user_id);
105
106 • SET FOREIGN_KEY_CHECKS=1;
107
108 # Per ara tenim un error d'incompatibilitat entre les columnes a vincular de la taula products i transactions, ho mirarem més endavant.
109
110 #ALTER TABLE products
111 #ADD FOREIGN KEY (id) REFERENCES transactions(product_ids);
112

```

En la creació de les diferents taules, hem hagut de realitzar un salt de bandera amb la instrucció *Foreign\_Key\_Checks*, i hem hagut de desactivar momentàniament les claus forànies. A més, no hem pogut crear la vinculació amb l'entitat *products* donat que hi ha un error d'incompatibilitat amb les dades dels camps a vincular. Ho mirarem més endavant.

## Exercici 1

Realitza una subconsulta que mostri tots els usuaris amb més de 30 transaccions utilitzant almenys 2 taules.

---

Realitzem la consulta amb una subquery com se'ns ha sol·licitat, i també afegim la mateixa consultat feta amb un Join.

```
115 #Realitza una subconsulta que mostri tots els usuaris amb més de 30 transaccions utilitzant almenys 2 taules.
116
117 #A continuació la subconsulta sol·licitada.
118 • SELECT users.id, users.name, users.surname, (SELECT count(transactions.user_id)
119 FROM transactions
120 WHERE transactions.user_id = users.id
121 ) as Num_transaccions
122 FROM users
123 GROUP BY 1, 2, 3
124 having Num_transaccions>30
125 ORDER BY 4 desc;
126
127 #Tanmateix, per obtenir les dades sol·licitades, no caldria fer una subconsulta, i la següent consulta seria més eficient:
128 • SELECT users.id, users.name, users.surname, count(transactions.user_id) AS Num_Operacions
129 FROM users
130 JOIN transactions ON users.id = transactions.user_id
131 GROUP BY 1, 2, 3
132 HAVING Num_Operacions > 30
133 ORDER BY 4 DESC;
```

Result Grid

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

Result 3 x Read Only

## Exercici 2

Mostra la mitjana de la suma de transaccions per IBAN de les targetes de crèdit en la companyia Donec Ltd. utilitzant almenys 2 taules.

---

En aquesta consulta, hem afegit dos Joins, un per poder consultar el camp *iban* a l'entitat *credit\_card* i l'altre per poder filtrar el nom de la companyia que es troba a l'entitat *companies*.

```
135 #Ex2
136 #Mostra la mitjana de la suma de transaccions per IBAN de les targetes de crèdit en la companyia Donec Ltd. utilitzant almenys 2 taules.
137
138 • SELECT companies.company_name as Companyia, credit_cards.iban, ROUND(AVG(transactions.amount),2) as Mitjana_transaccions
139 FROM transactions
140 JOIN companies ON transactions.company_id = companies.id
141 JOIN credit_cards ON transactions.card_id = credit_cards.id
142 #WHERE transactions.declined = 0
143 WHERE companies.company_name = 'Donec Ltd'
144 group by 1, 2
145 ;
146
147 #####
148 #Nivell 2
```

Result Grid

	Companyia	iban	Mitjana_transaccions
▶	Donec Ltd	PT87806228135092429456346	203.72

## Nivell 2

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:

---

Després de diferents dubtes amb la creació d'aquesta taula, he procedit a crear-la amb una funció condicional on apliquem una suma amb les operacions declinades que si ens dona un resultat de 3 vol dir que la targeta està inactiva, en el cas que aquest número sigui inferior, la targeta apareixerà com activa.

```
154 • CREATE TABLE card_state as
155     SELECT credit_cards.id
156     /* , case
157        when transactions.declined = 0 then 'Activa'
158        else 'Inactiva'
159        end as estat_targeta*/
160     , if(sum(transactions.declined) = 3, 'Inactiva', 'Activa') as estat_targeta
161     #, transactions.declined, transactions.timestamp
162     FROM transactions
163     JOIN (SELECT transactions.timestamp, row_number() Over (partition by transactions.card_id order by transactions.timestamp DESC) as Num_operacions
164           FROM transactions) as sub_operacions ON transactions.timestamp = sub_operacions.timestamp
165     JOIN credit_cards ON credit_cards.id = transactions.card_id
166     WHERE Num_operacions <=3 -- aquest és el sistema que utilitzariem si filtressim per les 3 últimes operacions, que és el que demana l'exercici
167     #WHERE Num_operacions <=1 -- en aquest cas, només filtrem per la última operació, ja que la targeta només estarà inactiva si és l'última operació la
168     GROUP BY 1;
169
```

Per tal de poder obtenir les tres últimes operacions de cada targeta, hem hagut d'afegir una subconsulta una funció *partition by* per *card\_id*; establint per tant que cada número de targeta és una partició independent. Prèviament, hem afegit un *Row\_number* que ens compta les files (operacions) que té cada targeta. A més, hem afegit que estiguin ordenades per ordre temporal descendent, de tal manera que les operacions que apareixeran primer són les més recents.

Finalment, per acabar amb la creació de la nova taula, hem afegit la *Primary Key* i la *Foreign Key* vinculada amb la taula *credit\_card*.

```
169
170 #Afegirem Primary Key a la nova taula i la vincularem amb la taula credit_cards
171
172 • ALTER TABLE card_state
173     ADD PRIMARY KEY (id),
174     ADD FOREIGN KEY (id) references credit_cards(id);
175
```

## Exercici 1

Quantes targetes estan actives?

---

Afegim una consulta amb un filtre a l'estat de la targeta que determini que està activa.



```

186
187 #Ex1
188 #Quantes targetes estan actives?
189 • SELECT estat_targeta, count(*) as TT_Targeta_Activa
190 FROM card_state
191 WHERE estat_targeta = 'Activa'
192 GROUP BY 1;
193
194 ##
195
196 #Nivell 3

```

estat_targeta	TT_Targeta_Activa
Activa	275

### 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:

---

Per tal de procedir amb la creació de la nova taula, hem de separar els diferents valors que hi ha al camp *product\_ids* de la taula *transaction*, i per això utilitzarem la funció *Substring\_Index*, ja que aquesta ens permet dividir els valors d'aquest camp en subcadenaes, segons el delimitador corresponent, en aquest cas la coma.

A més, hem fet una Join amb una subconsulta anomenada numbers, on hem creat una sèrie de nombres de l'1 al 4 (hem fet de l'1 al 4, perquè són el mínim i màxim de productes que hi ha al camp *product\_ids* de la taula *transactions*) utilitzant la clàusula *Union All Select*.

```

198 /*Crea una taula amb la qual puguem unir les dades del nou arxiu products.csv amb la base de dades creada,
199 tenint en compte que des de transaction tens product_ids. Genera la següent consulta:
200 */
201
202 • CREATE TABLE product_transaction as
203 SELECT id as order_id, SUBSTRING_INDEX(SUBSTRING_INDEX(product_ids, ',', numbers.n), ',', -1) AS product_id
204 FROM transactions
205 JOIN (SELECT 1 n union all select 2 union all select 3 union all select 4) numbers
206 ON CHAR_LENGTH (transactions.product_ids) - CHAR_LENGTH(REPLACE(transactions.product_ids, ',', ''))>=numbers.n-1
207 ORDER BY order_id, product_id;
208
209 # Procedim a modificar la columna product_id com a inter, afegir la clau primària i la foreign key amb la taula transactions.
210 • ALTER table product_transaction
211 modify column product_id int,
212 add primary key (order_id, product_id),
213 add foreign key (order_id) references transactions(id);
214
215 #També borrem l'index que vam afegir a la taula transactions al camp product_ids
216
217 • SHOW INDEX FROM transactions;
218
219 • ALTER TABLE transactions
220 DROP INDEX product_ids;
221

```

Altrament, un cop creada la taula, hem canviat el tipus de dada del camp *product\_id* passant-lo de Varchar a Int. El qual ens permet establir la relació amb la taula de fets, ja que el camp id de la taula *products* és int.

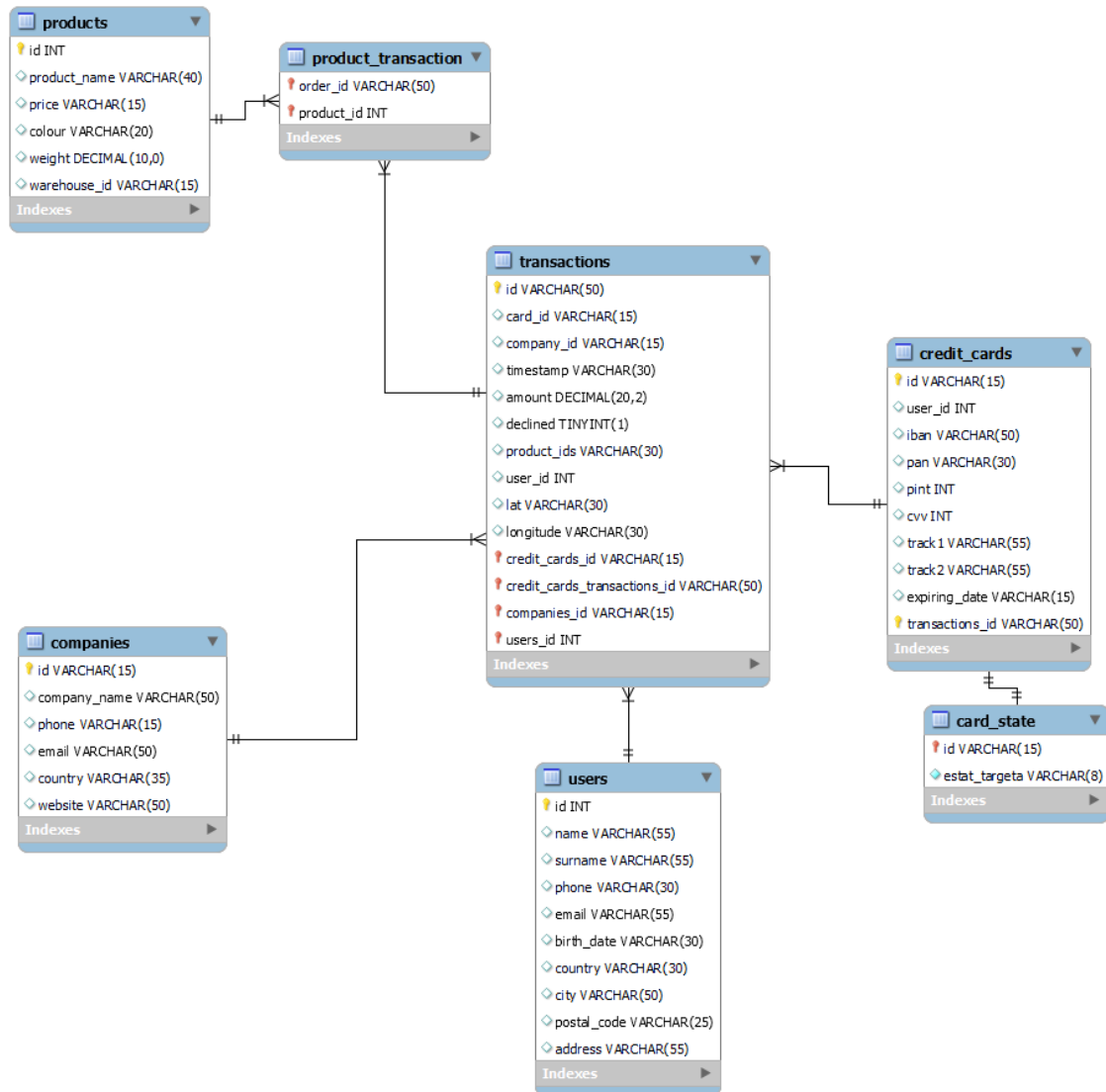
A més, també hem borrat l'index que havíem afegit a la taula *transactions* al camp *product\_ids*, doncs com ja havíem vist pel tipus de dada és incompatible; i establim la relació a la taula *products* amb la nova taula que he creat *product\_transaction*.

```

221
222      #I afegirem relacions amb altres taules que ens hem deixat fins ara
223
224 • ALTER TABLE product_transaction
225   ADD FOREIGN KEY (product_id) references products(id);
226
227

```

Així doncs, afegixo el diagrama de la base de dades perquè és vegi clarament les diferents relacions entre taules / entitats.



## Exercici 1

Necessitem conèixer el nombre de vegades que s'ha venut cada producte.

---

Per fer aquesta consulta, utilitzarem la nova taula que hem creat *product\_transaction* amb una funció *count* per producte, a més afegirem un *Join* amb la taula *product* per poder obtenir el nom del producte venut.

```

231 #Ex1
232 #Necessitem conèixer el nombre de vegades que s'ha venut cada producte.
233
234 • SELECT product_transaction.product_id as codi_producte, products.product_name as producte, count(product_transaction.product_id) as TT_Unitats
235 FROM product_transaction
236 JOIN products ON products.id = product_transaction.product_id
237 GROUP BY 1
238 ORDER BY 3 DESC;
239
240 #####
241

```

Result Grid			
Filter Rows:			
Export:   Wrap Cell Content:			
	codi_producte	producte	TT_Unitats
▶	23	riverlands north	68
	67	Winterfell	68
	79	Direwolf riverlands the	66
	2	Tarly Stark	65
	43	duel	65
	47	Tully	62
	1	Direwolf Stannis	61
	17	skywalker ewok sith	61
	97	jinn Winterfell	61
	13	nalnatine chewbacca	60

Result 1 x

Read Only