

Sprint 4

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 puguem realitzar les següents consultes.

```
3      /* CREO BB, TABLAS Y CONSTRAINTS */
4      create database sprint_4;
5      show databases;
6      use sprint_4;
7
8      create table companies (
9          company_id varchar(20) not null,
10         company_name varchar(255),
11         phone varchar(20),
12         email varchar(100),
13         country varchar(100),
14         website varchar(50),
15         primary key (company_id)
16     );
17
18     create table credit_cards (
19         id varchar(20) not null,
20         user_id int(11),
21         iban varchar(50),
22         pan varchar(50),
23         pin varchar(50),
24         cvv varchar(4),
25         track1 varchar(255),
26         track2 varchar(255),
27         expiring_date varchar(20),
28         primary key (id)
29     );
30
31     create table users (
32         id int(11) not null,
33         name varchar(255),
34         surname varchar(255),
35         phone varchar(255),
36         email varchar(255),
37         birth_date varchar(255),
38         country varchar(255),
39         city varchar(255),
40         postal_code varchar(255),
41         adress varchar(255),
42         primary key (id)
43     );
44
45     create table transactions (
46         id varchar(255) not null,
47         card_id varchar(20),
48         business_id varchar(20),
49         timestamp datetime not null default current_timestamp,
50         amount decimal(10,2),
51         declined tinyint(1),
52         product_ids varchar(50),
53         user_id int(11),
54         lat float,
55         longitude float,
56         primary key (id)
57     );
```

```

54
55 • alter table transactions
56 add constraint fk_card_id foreign key (card_id) REFERENCES credit_cards(id) on delete set null on update cascade;
57
58 • alter table transactions
59 add constraint fk_business_id foreign key (business_id) REFERENCES companies(company_id) on delete set null on update cascade;
60
61 • alter table transactions
62 add constraint fk_users_id foreign key (users_id) REFERENCES users(id) on delete set null on update cascade;
63

```

Decidí yo misma nombrar cada *constraint* porque en el *sprint* pasado tuve que borrarlos para corregir errores y me gusta saber el nombre de antemano; aun así, sé que si no lo nombro, MySQL generaría un nombre automáticamente y para encontrar el nombre tendría que buscarlo en el *output* del 'show create table'

Decidí agregar *ON DELETE* y *ON CASCADE* pensando en cómo preservar la integridad de la base de datos y evitar posibles datos huérfanos. Decidí usar *ON DELETE SET NULL* para que si un dato es borrado en la tabla padre (por ejemplo, un usuario es eliminado), las transacciones de ese usuario no sean borradas (se preserva la fila y el campo de usuario queda nulo).

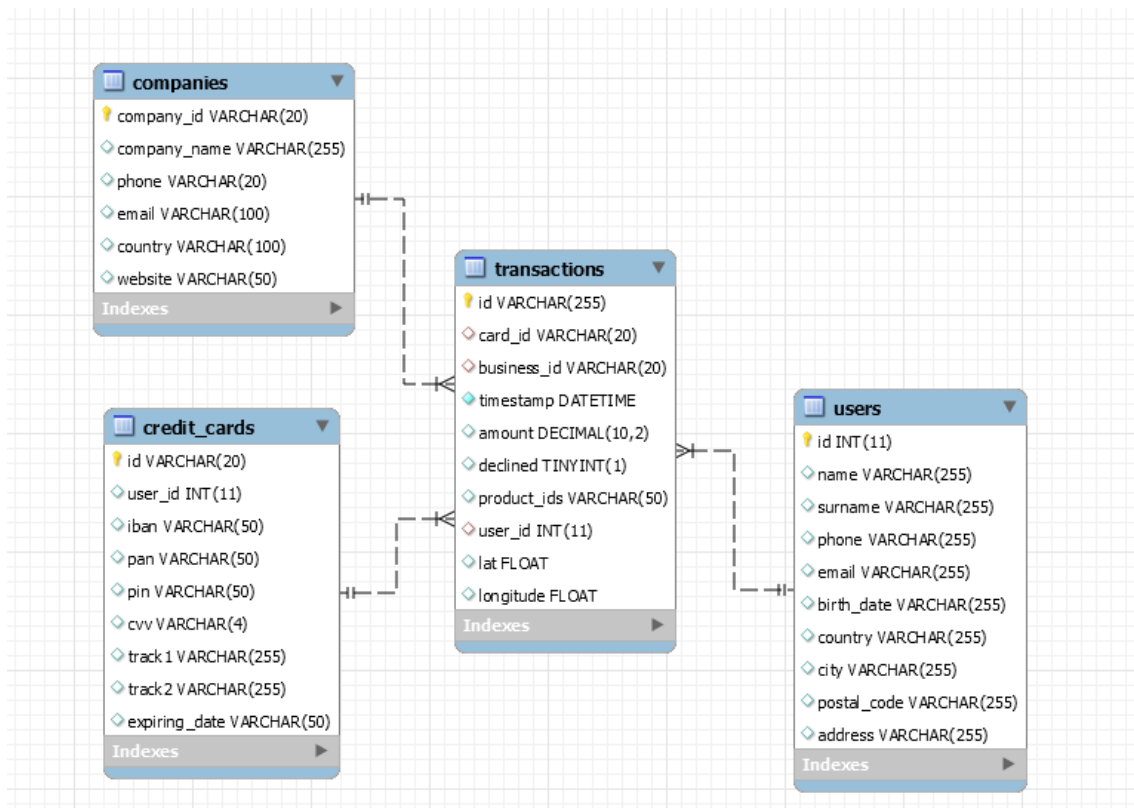
Decidí usar *ON UPDATE CASCADE* para que si un dato es actualizado en una tabla padre (por ejemplo, un usuario cambia de nombre) su nombre sea actualizado en todas las filas de transacciones que ha hecho (en la tabla hija/tabla de hechos).

```

64
65 /* INSERTO DATOS */
66
67
68 • show variables like 'secure_file_priv';
69
70 -- 'secure_file_priv', 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\'
71
72 ☐ /* Abro
73
74 C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\
75
76 y guardo los archivos csv ahí. */
77
78 • use sprint_4;
79
80 • load data infile "C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\companies.csv" into table companies
81 fields terminated by ','
82 enclosed by '"'
83 lines terminated by '\\n'
84 ignore 1 rows;
85
86 • load data infile "C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\credit_cards.csv" into table credit_cards
87 fields terminated by ','
88 enclosed by '"'
89 lines terminated by '\\n'
90 ignore 1 rows;
91
92 • load data infile "C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\users_usa.csv" into table users
93 fields terminated by ','
94 enclosed by '"'
95 lines terminated by '\\r\\n'
96 ignore 1 rows;
97
98 • load data infile "C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\users_uk.csv" into table users
99 fields terminated by ','
100 enclosed by '"'
101 lines terminated by '\\r\\n'
102 ignore 1 rows;
103
104 • load data infile "C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\users_ca.csv" into table users
105 fields terminated by ','
106 enclosed by '"'
107 lines terminated by '\\r\\n'
108 ignore 1 rows;
109
110 • load data infile "C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\transactions.csv" into table transactions
111 fields terminated by ';'
112 enclosed by '"'
113 lines terminated by '\\r\\n'
114 ignore 1 rows;
115

```

Para cada archivo tuve que revisar qué tipo de separador usaba y también tuve que probar con distintos tipos de quiebre de línea ya depende del sistema operativo en el que fue creado el archivo el archivo (Windows, Linux, Mac).



Exercici 1

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

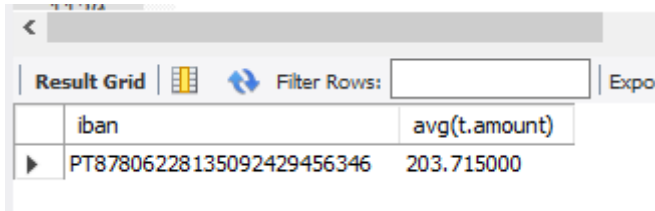
```
121 • select u.name, count(t.id)
122 from transactions as t
123 inner join users as u on u.id = t.user_id
124 group by u.name
125 having count(t.id) > 30;
126
127
```

Result Grid		
Filter Rows:		
	name	count(t.id)
▶	Lynn	39
	Ocean	52
	Hedwig	76
	Kenyon	48

Exercici 2

Mostra la mitjana d'amount per IBAN de les targetes de crèdit a la companyia Donec Ltd, utilitza almenys 2 taules.

```
150 • select cc.iban, avg(t.amount)
151 from transactions as t
152 inner join credit_cards as cc on cc.id = t.card_id
153 inner join companies as c on c.company_id = t.business_id
154 where c.company_name = 'Donec Ltd'
155 group by cc.iban;
156
```



The screenshot shows a database interface with a 'Result Grid' tab selected. The grid displays the results of the SQL query, showing the average amount for each IBAN. The interface includes a 'Filter Rows' search bar and an 'Export' button.

iban	avg(t.amount)
PT87806228135092429456346	203.715000

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.

```
344
345 • create table credit_cards_status_final as
346   select card_id,
347          sum(declined) as count_declined,
348          case
349            when sum(declined) = 3 then "Tarjeta Inactiva"
350            else "Tarjeta Activa"
351          end as card_status
352   from (
353     select *
354     from (
355       select card_id, declined, timestamp,
356              row_number() over(partition by card_id order by timestamp) as numbered_rows
357       from transactions
358     ) as temp_table
359
360     where numbered_rows <= 3
361     ) as temp_table_2
362
363   group by card_id;
```

Primero, muestro cómo se ve la tabla temporal que está en la subconsulta del FROM:

card_id	declined	timestamp	numbered_rows	count_declined	card_status
CcU-2938	0	2021-03-23 01:12:06	1	0	Tarjeta Activa
CcU-2938	0	2021-03-28 05:01:44	2	0	Tarjeta Activa
CcU-2938	0	2021-04-01 07:27:49	3	0	Tarjeta Activa
CcU-2945	1	2021-06-15 00:26:29	1	1	Tarjeta Activa
CcU-2945	0	2022-02-04 15:52:56	2	1	Tarjeta Activa
CcU-2952	1	2021-05-06 05:33:39	1	1	Tarjeta Activa
CcU-2952	0	2022-01-30 15:16:36	2	1	Tarjeta Activa
CcU-2959	0	2021-04-04 04:51:04	1	0	Tarjeta Activa
CcU-2959	0	2021-04-04 11:53:52	2	0	Tarjeta Activa
CcU-2959	0	2021-04-14 16:55:05	3	0	Tarjeta Activa
CcU-2966	1	2021-06-02 06:19:00	1	1	Tarjeta Activa
CcU-2966	0	2021-10-18 06:12:03	2	1	Tarjeta Activa
CcU-2973	1	2021-07-31 23:03:21	1	1	Tarjeta Activa
CcU-2973	0	2022-01-06 01:44:48	2	1	Tarjeta Activa
CcU-2980	0	2021-08-10 08:14:49	1	1	Tarjeta Activa
CcU-2980	1	2022-03-05 20:41:20	2	1	Tarjeta Activa
CcU-2987	1	2021-05-18 12:03:25	1	1	Tarjeta Activa
CcU-2987	0	2022-01-06 21:25:27	2	1	Tarjeta Activa
CcU-2994	0	2021-04-06 17:24:44	1	0	Tarjeta Activa
CcU-2994	0	2021-04-23 13:07:58	2	0	Tarjeta Activa
CcU-2994	0	2021-04-25 19:11:52	3	0	Tarjeta Activa

Y así se queda la tabla que únicamente muestra si las tarjetas están activas o inactivas:

1 • `SELECT * FROM sprint_4.credit_cards_status_final;`

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

	card_id	count_declined	card_status
▶	CcU-2938	0	Tarjeta Activa
	CcU-2945	1	Tarjeta Activa
	CcU-2952	1	Tarjeta Activa
	CcU-2959	0	Tarjeta Activa
	CcU-2966	1	Tarjeta Activa
	CcU-2973	1	Tarjeta Activa
	CcU-2980	1	Tarjeta Activa
	CcU-2987	1	Tarjeta Activa
	CcU-2994	0	Tarjeta Activa
	CcU-3001	1	Tarjeta Activa

Con esta tabla ya puedo realizar la consulta para responder: ¿Cuántas tarjetas están activas?

```

365 -- y ahora sí puedo contar:
366
367 • select count(card_status)
368     from credit_cards_status_final
369     where card_status like "%Tarjeta Activa%";

```

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

	count(card_status)
▶	275

```

370
371 • select count(card_status)
372     from credit_cards_status_final
373     where card_status like "%Tarjeta Inactiva%";
374
375 -- Todas están activas.
376

```

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

	count(card_status)
▶	0

En base a la instrucción, no sé si es opcional generar la *FK constraint* con credit_cards y terminar con un modelo de copo de nieve ya que habría dos tablas dimensional conectadas.