

- EXERCICI 1 - NIVELL 1

La teva tasca és dissenyar i crear una taula anomenada "credit_card" que emmagatzemi detalls crucials sobre les targetes de crèdit. La nova taula ha de ser capaç d'identificar de manera única cada targeta i establir una relació adequada amb les altres dues taules ("transaction" i "company"). Després de crear la taula serà necessari que ingressis la informació del document denominat "dades_introduir_credit". Recorda mostrar el diagrama i realitzar una breu descripció d'aquest.

A continuació, creem la taula que ens demana amb els camps necessaris per enregistrar les dades de l'arxiu dades_introduir_credit. Assignarem el camp *id* com a primary key. En aquest cas, funcionarà com a clau primària perquè no hi ha valors repetits i ens serveix per identificar els clients de manera única e inequívoca.

Hem assignat els camps id, iban, pan i pin com a variables varchar ja que es tracten de dades que poden contenir caràcters, números i caràcters especials de llargades variables. En els casos que no s'ha escollit la llargada màxima del varchar (p.e el camp pin), ens hem assegurat que els valors de les dades no superen el nombre de caràcters assignats a la creació de la taula.

Això s'ha fet amb l'objectiu d'agilitzar el funcionament de la base de dades i emprar una menor quantitat de memòria.

A més, el camp *id* que posteriorment serà el que fem servir per establir una relació entre aquesta taula i la taula transaction, s'ha establert com a varchar(15 caràcters), que es com està especificat a transaction. D'aquesta manera, no hi haurà problemes al establir la relació entre taules.

```
1 CREATE TABLE credit_card (  
2     id varchar(15) PRIMARY KEY,  
3     iban varchar(50),  
4     pan varchar(255),  
5     pin varchar(4),  
6     cvv int,  
7     expiring_date varchar(20)  
8 );  
9  
10  
11  
12  
13  
14  
15  
16  
17
```

Output

#	Time	Action	Message	Duration / Fetch
1	11:25:58	CREATE TABLE credit_card (id varchar(15) PRIMARY KEY, iban varchar(50), p...	0 row(s) affected	0.047 sec

Fem córrer l'script de les dades per omplir la taula en una altra pestanya

Per poder establir una relació amb les altres taules de la base de dades, afegirem una clàusula constraint o restricció del tipus foreign key amb un nom que li assignem nosaltres (fk_transaction_credit_card). Aquesta foreign key l'assignem a la taula transaction al camp credit_card_id i referenciem a quina taula i columna apunta. És a dir, definim una relació entre les taules transaction i credit_card a través de la clau primària de credit_card (anomenada id).

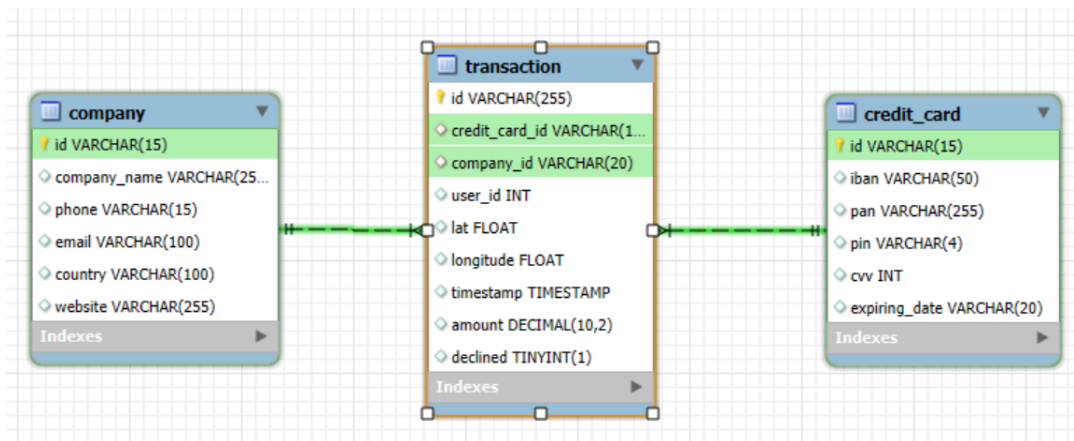
```
10 • ALTER TABLE transaction
11 ADD CONSTRAINT fk_transaction_credit_card
12 FOREIGN KEY (credit_card_id)
13 REFERENCES credit_card(id);
14
15
16
17
18
19
20
21
22
23
24
25
```

Output

#	Time	Action	Message	Duration / Fetch
✓ 276	11:28:18	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Cc...	1 row(s) affected	0.000 sec
✓ 277	11:28:18	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('Cc...	1 row(s) affected	0.000 sec
✓ 278	11:28:22	ALTER TABLE transaction ADD CONSTRAINT fk_transaction_credit_card FOREI...	587 row(s) affected Records: 587 Duplicates: 0 Warnings: 0	0.297 sec

D'aquesta manera, la nostra base de dades, ja queda actualitzada amb la nova taula (credit_card). Aquesta taula serà una taula de dimensions que emmagatzema la informació clau, sense repeticions, de les targetes de crèdit dels clients que fan transaccions. Està relacionada amb la taula transaction a través de la clau primària (id) de 1 a molts. És a dir, la taula credit_card_id tindrà la informació de les identifikacions de totes les targetes dels clients sense repetir, mentre que la taula transaction tindrà tantes vegades aquesta informació com transaccions hagin fet amb una mateixa targeta.

Amb la incorporació d'aquesta taula, el nostre model de dades es converteix en un model d'estrella. En el que tenim una taula de fets (transaction) i varies taules de dimensions (company i ara credit_card), que es relacionen amb la taula de fets de 1 a n.



- EXERCICI 2

El departament de Recursos Humans ha identificat un error en el número de compte de l'usuari amb ID CcU-2938. La informació que ha de mostrar-se per a aquest registre és: R323456312213576817699999. Recorda mostrar que el canvi es va realitzar.

Aquí tenim les dades del client amb targeta Ccu-2938 abans de modificar la informació del número de compte.

```

15 • SELECT *
16 FROM credit_card
17 WHERE
18 id = "CcU-2938";
19
20
21
  
```

id	iban	pan	pin	cvv	expiring_date
CcU-2938	TR301950312213576817638661	5424465566813633	3257	984	10/30/22

credit_card 1 x

Output

#	Time	Action	Message	Duration / Fetch
277	11:28:18	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ("CcU-2938", "TR301950312213576817638661", "5424465566813633", "3257", "984", "10/30/22")	1 row(s) affected	0.000 sec
278	11:28:22	ALTER TABLE transaction ADD CONSTRAINT fk_transaction_credit_card FOREIGN KEY (credit_card_id) REFERENCES credit_card (id)	587 row(s) affected Records: 587 Duplicates: 0 Warnings: 0	0.297 sec
279	12:38:02	SELECT * FROM credit_card WHERE id = "CcU-2938" LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

```

20 • UPDATE credit_card
21 SET iban = "R323456312213576817699999"
22 WHERE id = "CcU-2938";
23
24 • SELECT *
25 FROM credit_card
26 WHERE
27 id = "CcU-2938";
28
  
```

id	iban	pan	pin	cvv	expiring_date
CcU-2938	R323456312213576817699999	5424465566813633	3257	984	10/30/22

credit_card 1 x

Output

#	Time	Action	Message	Duration / Fetch
1	12:44:43	UPDATE credit_card SET iban = "R323456312213576817699999" WHERE id = "CcU-2938"	0 row(s) affected Rows matched: 1 Changed: 0 Warnings: 0	0.000 sec
2	12:44:46	SELECT * FROM credit_card WHERE id = "CcU-2938" LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

Mitjançant la funció UPDATE i SET, podem seleccionar la taula que volem modificar i a SET seleccionem el camp que volem modificar, posem el valor, i a la condició WHERE, filtrem en aquest cas per l'id de targeta en concret al que li volem fer aquest canvi.

Ara, ens assegurem de que les dades han estat correctament modificades demanant-li a la taula la informació del número de targeta.

- EXERCICI 3

En la taula "transaction" ingressa un nou usuari amb la següent informació:

id	108B1D1D-5B23-A76C-55EF-C568E49A99DD
credit_card_id	CcU-9999
company_id	b-9999
user_id	9999
lat	829.999
longitude	-117.999
amount	111.11
declined	0

```
31 # exercici 3
32 • INSERT INTO credit_card (id)
33   VALUES ("CcU-9999");
34 • INSERT INTO company (id)
35   VALUES ("b-9999");
36 • INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined)
37   VALUES( "108B1D1D-5B23-A76C-55EF-C568E49A99DD", "CcU-9999", "b-9999", 9999, 829.999, -117.999, 111.11, 0);
38 • SELECT *
39   FROM transaction
40  WHERE credit_card_id = "CcU-9999";
```

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
108B1D1D-5B23-A76C-55EF-C568E49A99DD	CcU-9999	b-9999	9999	829.999	-117.999	NULL	111.11	0
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

transaction 1 x [Apply] [Revert]

Output:

#	Time	Action	Message	Duration / Fetch
2	13:18:37	INSERT INTO credit_card (id) VALUES ("CcU-9999")	1 row(s) affected	0.015 sec
3	13:18:46	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, a...	1 row(s) affected	0.000 sec
4	13:19:32	SELECT * FROM transaction WHERE credit_card_id = "CcU-9999" LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

Com la taula de fets està directament relacionada mitjançant les foreign keys a les taules de dimensions, si afegim directament la informació d'un nou client a aquesta sense tenir la informació a les nostres taules de dimensions, ens donarà error.

Per això, és necessari crear prèviament els camps a les taules de dimensions perquè quan insertem les dades a la taula de fets, aquesta pugui relacionar les seves foreign keys correctament.

D'aquesta manera, farem INSERT INTO primerament a la taula credit_card un nou registre per l'id (id de la targeta nova) que correspondrà a VALUES ("CcU-9999"). El camp id

d'aquesta taula és el que es relaciona amb la nostra taula de fets. Si aquest id ja existís, al output ens sortiria com duplicat i no faria cap canvi (passa el mateix a la taula company).

Per altra banda, farem el mateix amb l'altra taula de dimensions, la taula company. En aquest cas, aquesta es relaciona amb transaction a través de id també (fa referència al company_id).

Una vegada hem creat aquests registres a les taules de dimensions, ja podem afegir totes les dades que ens queden a la taula transaction. Per últim, comprovem que s'ha afegit seleccionant tots els camps de la taula transaction filtrant per qualsevol valor dels que hem afegit. En aquest cas hem filtrat per credit_card_id = "CcU-9999".

- EXERCICI 4

Des de recursos humans et sol·liciten eliminar la columna "pan" de la taula credit_card. Recorda mostrar el canvi realitzat.

The screenshot shows a database management interface. At the top, a SQL query is entered in a text area:

```
43 # Exercici 4
44 • SELECT *
45 FROM credit_card;
46
47
48
49
50
```

Below the query editor, a "Result Grid" displays the results of the query. The grid has columns: id, iban, pin, cvv, expiring_date, and pan. The data is as follows:

id	iban	pin	cvv	expiring_date	pan
CcU-2938	R323456312213576817699999	3257	984	10/30/22	5424465566813633
CcU-2945	DO26854763748537475216568689	9080	887	08/24/23	5142423821948828
CcU-2952	BG451VQL52710525608255	4598	438	06/29/21	4556 453 55 5287
CcU-2959	CR7242477244335841535	3583	667	02/24/23	372461377349375
CcU-2966	BG72LKTQ15627628377363	4900	130	10/29/24	448566 886747 7265

Below the result grid, an "Output" section shows the execution details:

#	Time	Action	Message	Duration / Fetch
1	13:54:03	SELECT * FROM credit_card LIMIT 0, 1000	276 row(s) returned	0.000 sec / 0.000 sec

Primer, llançem una comanda perquè ens mostri tots els camps de la taula credit_card. Observem que ens mostra el camp pan, entre d'altres.

```

43 # Exercici 4
44 • ALTER TABLE credit_card
45 DROP COLUMN pan;
46
47 • SELECT *
48 FROM credit_card;
49
50
51
52

```

id	iban	pin	cvv	expiring_date
CcU-2938	R323456312213576817699999	3257	984	10/30/22
CcU-2945	DO26854763748537475216568689	9080	887	08/24/23
CcU-2952	BG451VQL52710525608255	4598	438	06/29/21
CcU-2959	CR7242477244335841535	3583	667	02/24/23
CcU-2966	BG72LKTQ15627628377363	4900	130	10/29/24

credit_card 2 x

Output

#	Time	Action	Message	Duration / Fetch
4	13:19:32	SELECT * FROM transaction WHERE credit_card_id = "CcU-9999" LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
5	13:37:39	ALTER TABLE credit_card DROP COLUMN pan	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.078 sec
6	13:38:00	SELECT * FROM credit_card LIMIT 0, 1000	276 row(s) returned	0.000 sec / 0.000 sec

Ara, per eliminar la columna, ho fem mitjançant ALTER TABLE i DROP COLUMN. Amb el nom de la columna que volem destruir (en aquest cas pan), podem observar que desapareix.

```

49 • SHOW COLUMNS
50 FROM credit_card;
51

```

Field	Type	Null	Key	Default	Extra
id	varchar(15)	NO	PRI	NULL	
iban	varchar(50)	YES		NULL	
pin	varchar(4)	YES		NULL	
cvv	int	YES		NULL	
expiring_date	varchar(20)	YES		NULL	

Result 3 x

Output

#	Time	Action	Message	Duration / Fetch
1	11:22:31	SHOW COLUMNS FROM credit_card	5 row(s) returned	0.000 sec / 0.000 sec

Una altra manera de comprovar quines columnes tenim a la taula credit_card seria mitjançant SHOW COLUMNS. D'aquesta manera, només mostra els noms dels camps i no totes les dades i verifiquem que el camp pan s'ha eliminat correctament.

- EXERCICI 1 - NIVELL 2

Elimina de la taula transaction el registre amb ID 02C6201E-D90A-1859-B4EE-88D2986D3B02 de la base de dades.

```
51 # Nivell 2
52 # Exercici 1
53 • SELECT *
54 FROM transaction
55 WHERE id = "02C6201E-D90A-1859-B4EE-88D2986D3B02";
56
57
58
59
```

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
02C6201E-D90A-1859-B4EE-88D2986D3B02	CcU-2938	b-2362	92	81.9185	-12.5276	2021-08-28 23:42:24	466.92	0
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

transaction 1 x Apply Revert

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	10:16:07	SELECT * FROM transaction WHERE id = "02C6201E-D90A-1859-B4EE-88D2986D3B02";	1 row(s) returned	0.000 sec / 0.000 sec

Primerament, seleccionem tots els camps de la taula transaction i filtrem per l'id de la transacció que ens indica l'enunciat.

```
56 • DELETE FROM transaction
57 WHERE id = "02C6201E-D90A-1859-B4EE-88D2986D3B02";
58 • SELECT *
59 FROM transaction
60 WHERE id = "02C6201E-D90A-1859-B4EE-88D2986D3B02";
61
62
```

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

transaction 4 x Apply Revert

Output

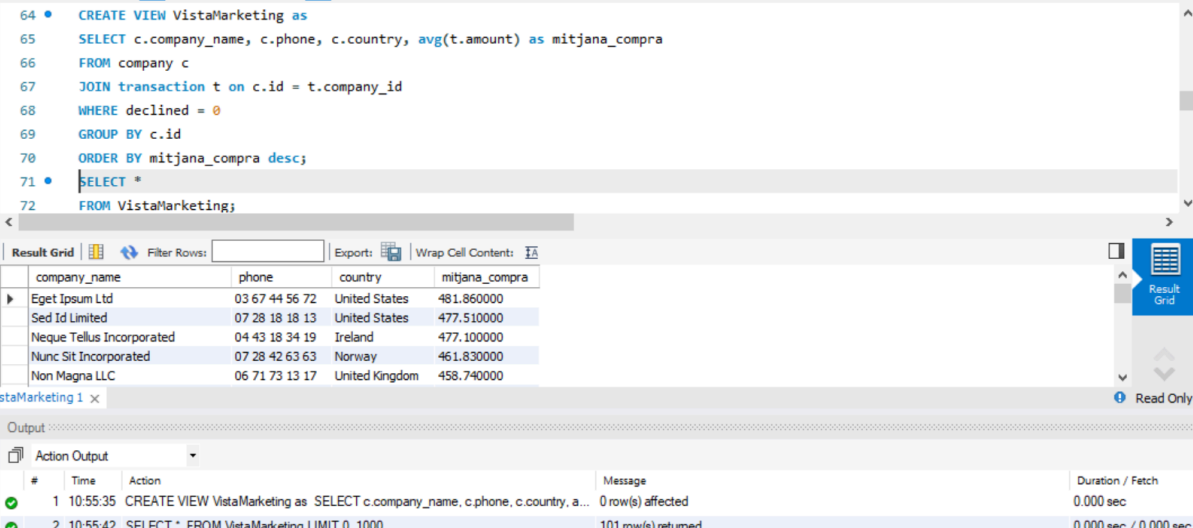
Action Output

#	Time	Action	Message	Duration / Fetch
1	10:21:20	DELETE FROM transaction WHERE id = "02C6201E-D90A-1859-B4EE-88D2986D3B02";	1 row(s) affected	0.000 sec
2	10:21:25	SELECT * FROM transaction WHERE id = "02C6201E-D90A-1859-B4EE-88D2986D3B02";	0 row(s) returned	0.000 sec / 0.000 sec

Eliminem el id indicat mitjançant la funció DELETE FROM, filtrant per l'id que volem eliminar. Després, tornem a repetir la query inicial per assegurar-nos que s'ha esborrat el registre.

- EXERCICI 2

La secció de màrqueting desitja tenir accés a informació específica per a realitzar anàlisi i estratègies efectives. S'ha sol·licitat crear una vista que proporcioni detalls clau sobre les companyies i les seves transaccions. Serà necessària que creïs una vista anomenada VistaMarketing que contingui la següent informació: Nom de la companyia. Telèfon de contacte. País de residència. Mitjana de compra realitzat per cada companyia. Presenta la vista creada, ordenant les dades de major a menor mitjana de compra.



```
64 • CREATE VIEW VistaMarketing as
65 SELECT c.company_name, c.phone, c.country, avg(t.amount) as mitjana_compra
66 FROM company c
67 JOIN transaction t on c.id = t.company_id
68 WHERE declined = 0
69 GROUP BY c.id
70 ORDER BY mitjana_compra desc;
71 • SELECT *
72 FROM VistaMarketing;
```

company_name	phone	country	mitjana_compra
Eget Ipsum Ltd	03 67 44 56 72	United States	481.860000
Sed Id Limited	07 28 18 18 13	United States	477.510000
Neque Tellus Incorporated	04 43 18 34 19	Ireland	477.100000
Nunc Sit Incorporated	07 28 42 63 63	Norway	461.830000
Non Magna LLC	06 71 73 13 17	United Kingdom	458.740000

#	Time	Action	Message	Duration / Fetch
1	10:55:35	CREATE VIEW VistaMarketing as SELECT c.company_name, c.phone, c.country, a...	0 row(s) affected	0.000 sec
2	10:55:42	SELECT * FROM VistaMarketing LIMIT 0, 1000	101 row(s) returned	0.000 sec / 0.000 sec

En aquest exercici, crearem una vista a la que anomenem VistaMarketing com ens indica l'enunciat. Seleccionem els camps que volem mostrar de les taules company i transaction (els quals hem abreviat com *c* i *t* respectivament). També farem una funció d'agregació average per trobar la mitjana del amount de transacció de cada empresa. Com tenim dades de dues taules, farem una join entre les dues mitjançant la clau primària i forana de cada taula i afegirem la condició `declined = 0`, per comptar solament les transaccions efectives. Agruparem per l'id de les empreses per tenir la mitjana de les compres associada a cada empresa i ordenarem de major a menor per la quantitat de vendes (amb àlies `mitjana_compra`).

- EXERCICI 3

Filtra la vista VistaMarketing per a mostrar només les companyies que tenen el seu país de residència en "Germany"

```
74 # Exercici 3
75 • SELECT *
76 FROM VistaMarketing
77 WHERE country = "Germany";
78
79
80
81
82
```

company_name	phone	country	mitjana_compra
Aliquam PC	01 45 73 52 16	Germany	385.265000
Ac Industries	09 34 65 40 60	Germany	289.645000
Rutrum Non Inc.	02 66 31 61 09	Germany	266.900000
Nunc Interdum Incorporated	05 18 15 48 13	Germany	244.025238
Augue Foundation	06 88 43 15 63	Germany	240.800000

VistaMarketing 12 x Read Only

Output

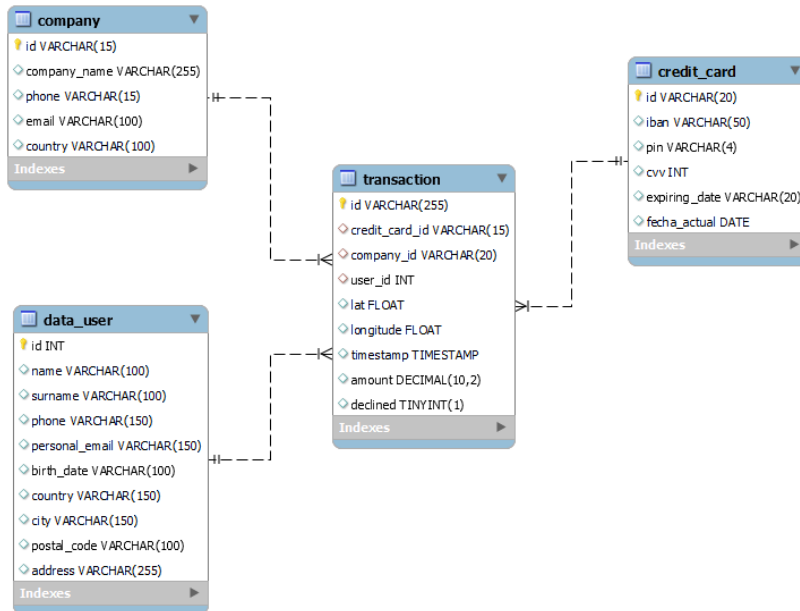
Action Output

#	Time	Action	Message	Duration / Fetch
✓ 1	10:49:15	CREATE VIEW VistaMarketing as SELECT c.company_name, c.phone, c.country, a...	0 row(s) affected	0.015 sec
✓ 2	10:49:19	SELECT * FROM VistaMarketing LIMIT 0, 1000	101 row(s) returned	0.000 sec / 0.000 sec
✓ 3	11:05:33	SELECT * FROM VistaMarketing WHERE country = "Germany" LIMIT 0, 1000	8 row(s) returned	0.016 sec / 0.000 sec

En aquest cas només cal seleccionar tots els camps de la nostra vista i afegir-li un filtre per país. En aquest cas el país és "Germany". Com abans hem ordenat la nostra view de major a menor segons la quantitat mitjana de compra, en aquest cas no cal ordenar les dades.

- EXERCICI 1 - NIVELL 3

La setmana vinent tindràs una nova reunió amb els gerents de màrqueting. Un company del teu equip va realitzar modificacions en la base de dades, però no recorda com les va realitzar. Et demana que l'ajudis a deixar els comandos executats per a obtenir el següent diagrama:



Resum de les accions fetes:

1. Modificacions a la taula credit_card

- Afegir nou camp anomenat fecha_actual amb format date.
- Canvi de variable *id* de varchar(15) a varchar(20).

2. Creació i modificacions de la taula data_user

- Creació de la taula user
- Canvi de nom de la taula user a data_user
- Cerca i eliminació de la foreign key amb orientació errònia establerta al script de creació
- Intent d'establiment de la foreign key correcta i cerca del user_id que falta a la taula user_data que sí que es troba a la taula transaction
- Inserció del user_id "9999" a la taula data_user i creació de la foreign key correcta.

1. Modificacions a la taula credit_card

```
79 # Nivell 3
80 # Exercici 1
81 • ALTER TABLE credit_card
82   ADD COLUMN fecha_actual date;
83 • SELECT *
84   FROM credit_card;
```

id	iban	pin	cvv	expiring_date	fecha_actual
CcU-2938	R323456312213576817699999	3257	984	10/30/22	NULL
CcU-2945	DO26854763748537475216568689	9080	887	08/24/23	NULL
CcU-2952	BG451VQL52710525608255	4598	438	06/29/21	NULL
CcU-2959	CR7242477244335841535	3583	667	02/24/23	NULL
CcU-2966	BG72LKTQ15627628377363	4900	130	10/29/24	NULL

credit_card 15 x

Output

#	Time	Action	Message	Duration / Fetch
1	11:21:11	ALTER TABLE credit_card ADD COLUMN fecha_actual date	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.063 sec
2	11:22:08	SELECT * FROM credit_card LIMIT 0, 1000	276 row(s) returned	0.000 sec / 0.000 sec

La primera modificació que s'ha fet a la base de dades és que s'ha afegit un nou camp anomenat `fecha_actual` amb format `date` a la taula `credit_card`. Tot i així, com no hem afegit dades a aquest camp de moment ens surt com `NULL` (està buit).

```
89 • SHOW COLUMNS
90   FROM credit_card;
```

Field	Type	Null	Key	Default	Extra
id	varchar(15)	NO	PRI	NULL	
iban	varchar(50)	YES		NULL	
pin	varchar(4)	YES		NULL	
cvv	int	YES		NULL	
expiring_date	varchar(20)	YES		NULL	
fecha_actual	date	YES		NULL	

Result 7 x

Output

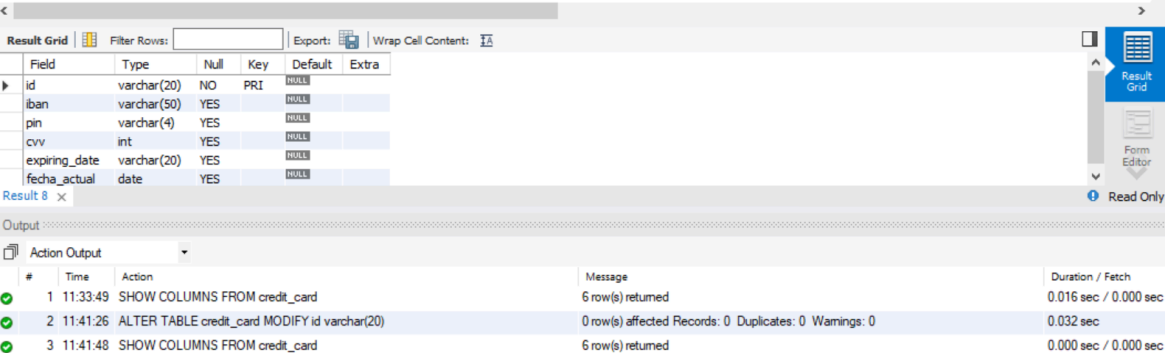
#	Time	Action	Message	Duration / Fetch
1	11:33:49	SHOW COLUMNS FROM credit_card	6 row(s) returned	0.016 sec / 0.000 sec

Observem que el tipus de variable corresponent a `id` que tenim a la taula `credit_card` no correspon amb l'esquema de l'enunciat. A l'enunciat es tracta d'un `varchar(20)` i nosaltres tenim `varchar(15)`

```

92 # cambiamos los tipos de datos de
93 • ALTER TABLE credit_card
94   MODIFY id varchar(20);
95 • SHOW COLUMNS
96 FROM credit_card;
97
98
99

```



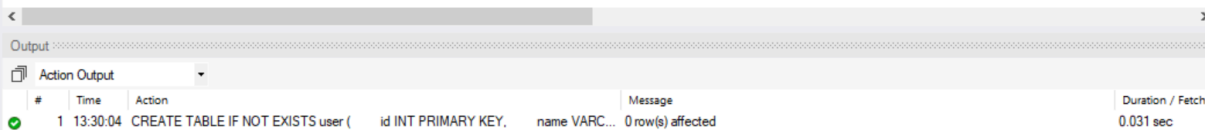
Per fer el canvi, fem servir la funció ALTER TABLE + MODIFY amb el camp que volem canviar. Després farem un SHOW COLUMNS per veure que el canvi s'ha produït correctament.

2. Creació i modificacions de la taula data_user

```

98 • CREATE INDEX idx_user_id ON transaction(user_id);
99 • CREATE TABLE IF NOT EXISTS user (
100     id INT PRIMARY KEY,
101     name VARCHAR(100),
102     surname VARCHAR(100),
103     phone VARCHAR(150),
104     email VARCHAR(150),
105     birth_date VARCHAR(100),
106     country VARCHAR(150),
107     city VARCHAR(150),
108     postal_code VARCHAR(100),
109     address VARCHAR(255),
110     FOREIGN KEY(id) REFERENCES transaction(user_id)
111 );
112

```



A continuació, copiant l'script de l'exercici anomenat estructura_datos_user.sql. Aquest script, tot i no donar error al executar-se, té un error. La manera en que s'estableix la relació entre la taula transaction i user (FOREIGN KEY(id) REFERENCES transaction(user_id)) és errònia. Aquesta relació s'ha establert del revés, de manera que es determina el camp id de la taula user com a foreign key i el camp user_id de la taula transaction com primary key. De moment però, inserim les dades.

```
265 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "262", "Brett", "Kirby"
266 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "263", "Ima", "Hendrick
267 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "264", "Keiko", "Guerra
268 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "265", "Chloe", "Keith"
269 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "266", "Aiko", "Chaney"
270 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "267", "Ocean", "Nelson
271 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "268", "Clark", "Olson"
272 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "269", "Haley", "Fitzpa
273 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "270", "Elton", "Robers
274 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "271", "Leandra", "Cher
275 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "272", "Hedwig", "Gilbe
276 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "273", "Hilary", "Fergu
277 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "274", "Jameson", "Hunt
278 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ( "275", "Kenyon", "Hartm
279
280 • SET foreign_key_checks = 1;
```

Output

#	Time	Action	Message	Duration / Fetch
276	13:32:32	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal...	1 row(s) affected	0.015 sec
277	13:32:32	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal...	1 row(s) affected	0.016 sec
278	13:32:32	SET foreign_key_checks = 1	0 row(s) affected	0.000 sec

Ara, en una nova pestanya del workbench fem correr l'script preparat com `datos_introducir_user.sql`, que conté les dades per omplir la última taula creada: `data_user`.
Ho fem en una nova finestra per limitar l'arxiu sql `sprint3` a les queries demanades a l'enunciat.

```
102 • ALTER TABLE user
103 RENAME TO data_user;
104
```

Output

#	Time	Action	Message	Duration / Fetch
280	11:38:36	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal...	1 row(s) affected	0.015 sec
281	11:38:36	SET foreign_key_checks = 1	0 row(s) affected	0.000 sec
282	11:45:03	ALTER TABLE user RENAME TO data_user	0 row(s) affected	0.047 sec

Perquè la taula sigui com a l'esquema de l'enunciat, cal fer-li un rename a `data_user`.

SCHEMAS

Filter objects

- credit_card
- data_user
 - Columns
 - Indexes
 - Foreign Keys
 - data_user_ibfk_1
 - Triggers
 - transaction
- Views

Administration Schemas

Information

Foreign Key: data_user_ibfk_1

Definition:

Target transaction (id → user_id)

On Update RESTRICT

On Delete RESTRICT

```
117
118 • ALTER TABLE data_user
119 DROP FOREIGN KEY data_user_ibfk_1;
120
121
122
123
124
125
126
127
128
129
130
131
132
```

Output

#	Time	Action	Message	Duration / Fetch
1	13:41:39	ALTER TABLE data_user DROP FOREIGN KEY data_user_ibfk_1	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.015 sec

Abans, hem comentat que la relació que s'havia establert era errònia, per tant haurem d'eliminar la foreign key de l'script abans d'afegir-ne la correcta. Hem navegat dins d'schemas per conèixer el nom de la clau forana i l'hem eliminat.

```
120 • ALTER TABLE transaction
121 ADD CONSTRAINT fk_transaction_user
122 FOREIGN KEY (user_id) REFERENCES data_user(id);
123
124
125
126
127
128
129
130
131
132
133
134
```

Output

#	Time	Action	Message	Duration / Fetch
1	13:41:39	ALTER TABLE data_user DROP FOREIGN KEY data_user_ibfk_1	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.015 sec
2	13:44:13	ALTER TABLE transaction ADD CONSTRAINT fk_transaction_user FOREIGN K...	Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails ...	0.125 sec

Quan volem establir la correcta orientació de la relació mitjançant una constraint per establir-la (fk_transaction_user), ens diu que no podem afegir la clau forana. Això es deu a que segurament hi hagi files a la taula transaction amb valors de user_id que encara no existeixen a la taula que hem creat (data_user.id).

```
119 • SELECT DISTINCT user_id
120 FROM transaction
121 WHERE user_id NOT IN (
122 SELECT id FROM data_user);
```

Result Grid

user_id
9999

transaction 16 x Read Only

Output

#	Time	Action	Message	Duration / Fetch
1	12:36:06	ALTER TABLE transaction ADD CONSTRAINT fk_transaction_user FOREIGN KEY (...	Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails (tra...	0.078 sec
2	12:36:08	SELECT DISTINCT user_id FROM transaction WHERE user_id NOT IN (SELECT id...	1 row(s) returned	0.000 sec / 0.000 sec

Per trobar aquest usuari que es troba en la taula transaction però no en la taula data_user farem el següent:

Primerament, fer un SELECT DISTINCT per agafar només els user_id únics de la taula transaction. Filtrarem a través d'una subquery per aquells user_id que no es trobin en la taula data_user i ens surt només un resultat: el usuari 9999 (que l'hem afegit manualment abans).

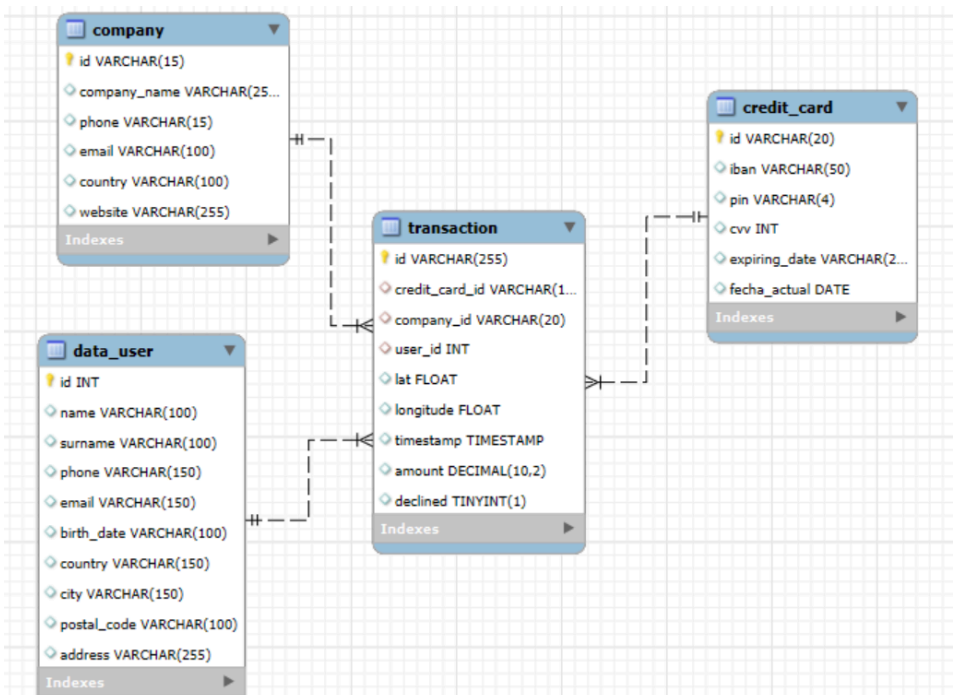
```

125 • INSERT INTO data_user (id)
126     VALUES (9999);
127 • ALTER TABLE transaction
128     ADD CONSTRAINT fk_transaction_user
129     FOREIGN KEY (user_id) REFERENCES data_user(id);
130
131
132
133
134
135
136
137

```

Output				
Action Output				
#	Time	Action	Message	Duration / Fetch
2	12:36:08	SELECT DISTINCT user_id FROM transaction WHERE user_id NOT IN (SELECT ...	1 row(s) returned	0.000 sec / 0.000 sec
3	12:39:24	INSERT INTO data_user (id) VALUES (9999)	1 row(s) affected	0.031 sec
4	12:39:24	ALTER TABLE transaction ADD CONSTRAINT fk_transaction_user FOREIGN KE...	587 row(s) affected Records: 587 Duplicates: 0 Warnings: 0	0.219 sec

Ara que ja sabem de quin usuari es tracta, podem afegir-lo a la taula data_user. Una vegada afegit, ja podem establir sense problemes la nova foreign key.



Per últim, mitjançant l'eina reverse engineer de mysql workbench, podem comprovar que el nostre esquema ja correspon amb el del enunciat.

- Exercici 2

L'empresa també et sol·licita crear una vista anomenada "InformeTecnico" que contingui la següent informació:

- ID de la transacció
- Nom de l'usuari/ària
- Cognom de l'usuari/ària
- IBAN de la targeta de crèdit usada.
- Nom de la companyia de la transacció realitzada.

Assegura't d'incloure informació rellevant de totes dues taules i utilitza àlies per a canviar de nom columnes segons sigui necessari.

Mostra els resultats de la vista, ordena els resultats de manera descendent en funció de la variable ID de transaction.

```
139 • CREATE VIEW InformeTecnico as
140 SELECT t.id as id_transaccio, du.name as nom_usuari, du.surname as cognom_usuari, cc.iban as iban_associat, c.company_name as nom_empresa_t
141 FROM credit_card cc
142 JOIN transaction t on t.credit_card_id = cc.id
143 JOIN data_user du on du.id = t.user_id
144 JOIN company c on c.id = t.company_id
145 ORDER BY id_transaccio desc;
146 • SELECT *
147 FROM InformeTecnico;
```

id_transaccio	nom_usuari	cognom_usuari	iban_associat	nom_empresa_transaccio
FE96CE47-8D59-381C-4E18-E3CA3D44E8FF	Kenyon	Hartman	DO26854763748537475216568689	Magna A Neque Industries
FE809ED4-2D86-55AC-C915-929516E4646B	Molly	Gilliam	SE2813123487163628531121	Nunc Interdum Incorporated
FD9CBCCD-8E1E-8DA1-4606-7E3A6F3A5A65	Linus	Willis	KW9485332754781757886242955643	Nunc Interdum Incorporated
FD89D51B-AE8D-77DC-E450-B8083FBD3187	Hilda	Levy	LT053237077744561475	Malesuada PC
FD2E8957-414B-BEEC-E9AD-59AA7A8A6290	Hedwig	Gilbert	GE84848451582810541526	Neque Tellus Imperdiet Corp.

InformeTecnico 18 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	12:48:15	SELECT * FROM InformeTecnico LIMIT 0, 1000	587 row(s) returned	0.015 sec / 0.000 sec

En aquest exercici hem creat la vista InformeTecnico seleccionant diversos camps de les 4 taules de la database. Com fem referència a les diverses taules, les hem abreviat per ocupar menys espai i fer el codi més llegible (p.e *data_user* s'ha abreviat com *du*). Tot i així s'han assignat àlies a tots els camps per fer més llegible el resultat de la view InformeTecnico.

Hem fet una triple JOIN ja que ens demana informació de les 4 taules. La primera join junta les taules transaction i credit_card mitjançant la foreign key de transaction *credit_card_id*.

La segona join junta data_user amb transaction mitjançant la clau forana *user_id* de transaction.

La tercera join junta company amb transaction mitjançant la clau forana *company_id* de transaction. Per últim, ordenem els id_transacció de manera descendent com ens demana l'enunciat i li demanem que ens mostri tots els camps de la nostra nova view.

Ens surten un total de 587 transaccions.