

N1E1

The screenshot shows a database management interface. On the left, the 'SCHEMAS' pane displays a tree view of the database structure, including tables like 'company', 'transaction', and 'credit_card'. The 'transaction' table is expanded, showing its columns: id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amount, and declined. The 'credit_card' table is also expanded, showing its columns: id, iban, pan, pin, cvv, and expiring_date. The main pane shows the execution of SQL commands. The first command is 'USE transactions;', and the second is 'CREATE TABLE IF NOT EXISTS credit_card (' followed by column definitions: id VARCHAR(15) PRIMARY KEY, iban VARCHAR(255), pan VARCHAR(15), pin CHAR(4), cvv CHAR(3), and expiring_date VARCHAR(255). The output pane shows the results of these commands, indicating that the table was created successfully. Below the main pane, there is a section for 'ALTER TABLE transaction' with the command 'ADD CONSTRAINT fk_credit_card_id FOREIGN KEY (credit_card_id) REFERENCES credit_card(id);'.

```

1 • USE transactions;
2
3 •
4 CREATE TABLE IF NOT EXISTS credit_card (
5     id VARCHAR(15) PRIMARY KEY,
6     iban VARCHAR(255),
7     pan VARCHAR(15),
8     pin CHAR(4),
9     cvv CHAR(3),
10    expiring_date VARCHAR(255)
11 )
12
13 ALTER TABLE transaction
14 ADD CONSTRAINT fk_credit_card_id FOREIGN KEY (credit_card_id) REFERENCES credit_card(id);
15

```

Table: company

Columns:

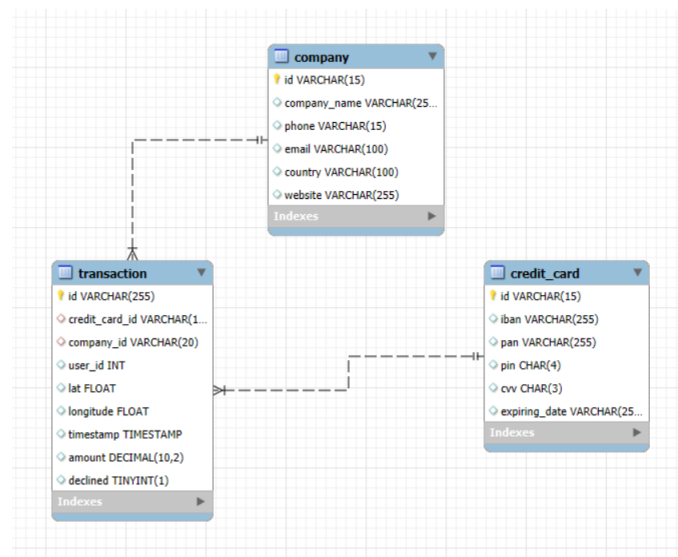
Column	DataType	PK
id	varchar(15)	PK
company_name	varchar(255)	
phone	varchar(15)	
email	varchar(100)	
country	varchar(100)	
website	varchar(255)	

Output

#	Time	Action	Message
1	11:56:55	USE transactions	0 row(s) affected
2	11:56:55	CREATE TABLE IF NOT EXISTS credit_card (0 row(s) affected

Output

#	Time	Action	Message	Duration / Fetch
278	12:12:53	INSERT INTO credit_card (id, iban, pan, pin, cvv,...	1 row(s) affected	0.016 sec
279	12:12:53	INSERT INTO credit_card (id, iban, pan, pin, cvv,...	1 row(s) affected	0.015 sec
280	12:12:53	INSERT INTO credit_card (id, iban, pan, pin, cvv,...	1 row(s) affected	0.016 sec
281	12:12:53	INSERT INTO credit_card (id, iban, pan, pin, cvv,...	1 row(s) affected	0.000 sec
282	12:12:53	INSERT INTO credit_card (id, iban, pan, pin, cvv,...	1 row(s) affected	0.016 sec
283	12:13:11	SELECT * from credit_card LIMIT 0, 50000	275 row(s) returned	0.000 sec / 0.000 sec
284	12:13:48	ALTER TABLE transaction ADD CONSTRAINT f...	587 row(s) affected Records: 587 Duplicates: 0 ...	0.156 sec



La nueva tabla se relaciona en 1:N con transaction usando credit_card_id como clave foránea, siendo la tabla credit_card 1 y transaction N. La relación tiene sentido ya que una

misma tarjeta puede hacer varias transacciones mientras que una misma transacción no es posible hacerla con distintas tarjetas.

N1E2

```
18 • UPDATE credit_card
19 SET iban = "R323456312213576817699999"
20 WHERE id = "CcU-2938";
21
22 • SELECT iban FROM credit_card WHERE id = "CcU-2938";
```

Result Grid

iban

R323456312213576817699999

credit_card 2

Output

Action Output

#	Time	Action	Message
1	13:13:20	UPDATE credit_card SET iban = "R323456312213...	1 row(s) affected Rows matched: 1 Changed: 1 W...
2	13:14:02	SELECT iban FROM credit_card WHERE id = "Cc...	1 row(s) returned

N1E3

```
25 -- Ejercicio 3
26 -- En la taula "transaction" ingressa un nou usuari
27 • INSERT INTO credit_card (id) VALUES ("CcU-9999"); -- Placeholder credit_card id para que no dé error insertar en transaction
28 • INSERT INTO company (id) VALUES ("b-9999"); -- Mismo placeholder
29 • INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined) VALUES ('10881D1D-5B23-A76C-55EF-C568E49A990D', 'CcU-9999',
30 • SELECT * FROM transaction WHERE credit_card_id = "CcU-9999";
```

Result Grid

id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
10881D1D-5B23-A76C-55EF-C568E49A990D	CcU-9999	b-9999	9999	829.999	-117.999	10881D1D-5B23-A76C-55EF-C568E49A990D	111.11	0

transaction 6

Output

Action Output

#	Time	Action	Message
1	13:29:56	SELECT * FROM transaction WHERE credit_card_id = "CcU-9999" LIMIT 0, 50000	0 row(s) returned
2	13:30:10	INSERT INTO credit_card (id) VALUES ("CcU-9999")	1 row(s) affected
3	13:30:10	INSERT INTO company (id) VALUES ("b-9999")	1 row(s) affected
4	13:30:10	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined) VALUES (1...	1 row(s) affected
5	13:30:24	SELECT * FROM transaction WHERE credit_card_id = "CcU-9999" LIMIT 0, 50000	1 row(s) returned

En este ejercicio he añadido una fila tanto a credit_card como a company con un solo dato (la primary key en ambas, que es la foreign key en transaction) para que al intentar añadir una fila a transaction no dé error por la falta de datos.

En un entorno real pediría esos datos a quien me los tuviera que proporcionar.

N1E4

```
35 • ALTER TABLE credit_card
36 DROP COLUMN pan;
37 • SELECT * FROM credit_card WHERE id = "CcU-9999";
```

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

id	iban	pin	cvv	expiring_date
CcU-9999	NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL	NULL

edit_card 8 x

output

Action Output

#	Time	Action	Message
1	13:39:08	ALTER TABLE credit_card DROP COLUMN pan	0 row(s) affected Records: 0 Duplicates: 0 Wamin...
2	13:39:14	SELECT * FROM credit_card WHERE id = "CcU-9...	1 row(s) returned

N2E1

```
2 • DELETE FROM transaction
3 WHERE id = "02C6201E-D90A-1859-B4EE-88D2986D3B02";
4
5 • select * from transaction where id = "02C6201E-D90A-1859-B4EE-88D2986D3B02";
```

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

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

Transaction 2 x

Input:

Action Output

#	Time	Action	Message
1	13:45:29	DELETE FROM transaction WHERE id = "02C620...	1 row(s) affected
2	13:45:33	select * from transaction where id = "02C6201E-D9...	0 row(s) returned

N2E2

```
7 -- Exercici 2
8 -- La secció de màrqueting desitja tenir accés a informació
9 • CREATE VIEW `VistaMarketing` AS
10 SELECT company_name, phone, country, avg(amount) AS Average
11 FROM company
12 JOIN transaction
13 ON (company.id = company_id)
14 GROUP BY company_name, phone, country
15 ORDER BY Average desc;
16
17 • select * from vistamarketing;
18
```

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

	company_name	phone	country	Average
▶	Eget Ipsum Ltd	03 67 44 56 72	United States	473.075000
	Non Magna LLC	06 71 73 13 17	United Kingdom	468.345000
	Sed Id Limited	07 28 18 18 13	United States	461.210000
	Justo Eu Arcu Ltd	08 42 56 71 52	Italy	443.635000
	Eget Tincidunt Dui Institute	05 35 93 32 44	Netherlands	442.520000

vistamarketing 2 x

Output:

Action Output

#	Time	Action	Message
✓ 1	10:27:23	CREATE VIEW `VistaMarketing` AS SELECT com...	0 row(s) affected
✓ 2	10:27:28	select * from vistamarketing LIMIT 0, 50000	101 row(s) returned

N2E3

```
17  -- Exercici 3
18  -- Filtra la vista VistaMarketing per a mostrar només les
19  • SELECT *
20  FROM vistamarketing
21  WHERE country = "Germany";
22
```

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

	company_name	phone	country	Average
▶	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 4 x

Output

Action Output

#	Time	Action	Message
✓ 1	10:30:57	SELECT * FROM vistamarketing WHERE country = ...	8 row(s) returned

N3E1

Para empezar vemos que la foreign key que ha intentado crear está mal hecha (la relación 1:N está al revés, siendo transaction el 1 en lugar de la N) así que buscamos cuál es el nombre de la regla constraint y la eliminamos.

```
18 • SELECT CONSTRAINT_NAME -- La fk esta mal creada en el código anterior, por lo q
19 FROM INFORMATION_SCHEMA.TABLE_CONSTRAINTS
20 WHERE TABLE_NAME = "data_user";
21
22 • ALTER TABLE data_user
23 DROP CONSTRAINT data_user_ibfk_1;
24
```

Output			
Action Output			
#	Time	Action	Message
1	10:59:50	SELECT CONSTRAINT_NAME FROM INFORMAT...	2 row(s) returned
2	11:01:59	ALTER TABLE data_user DROP CONSTRAINT da...	0 row(s) affected Records: 0 Duplicates: 0 Wamin...

Hecho eso creamos la nueva regla que llamaremos fk_user_id, esta vez con la relación 1:N adecuada siendo data_user el 1 y transaction la N.

Esta relación es lógica ya que una transacción solo la puede hacer un usuario, mientras que un usuario puede hacer tantas transacciones como quiera.

```
29 • ALTER TABLE transaction
30 ADD CONSTRAINT fk_user_id FOREIGN KEY (user_id) REFERENCES data_user(id);
```

Output			
Action Output			
#	Time	Action	Message
1	11:10:48	ALTER TABLE transaction ADD CONSTRAINT fk_...	586 row(s) affected Records: 586 Duplicates: 0 Wa...

También nos damos cuenta de que hay columnas distintas así que las editaremos:

```
29 • ALTER TABLE data_user -- Cambiamos el nombre de la columna email
30 CHANGE email personal_email VARCHAR(150);
```

Output			
Action Output			
#	Time	Action	Message
1	11:16:27	ALTER TABLE data_user -- Cambiamos el nombre d...	0 row(s) affected Records: 0 Duplicates: 0 Warning...

```

12 • ALTER TABLE company -- Borramos la columna website
13 DROP COLUMN website;

```

#	Time	Action	Message
1	11:18:38	ALTER TABLE company -- Borramos la columna web...	0 row(s) affected Records: 0 Duplicates: 0 Warning...

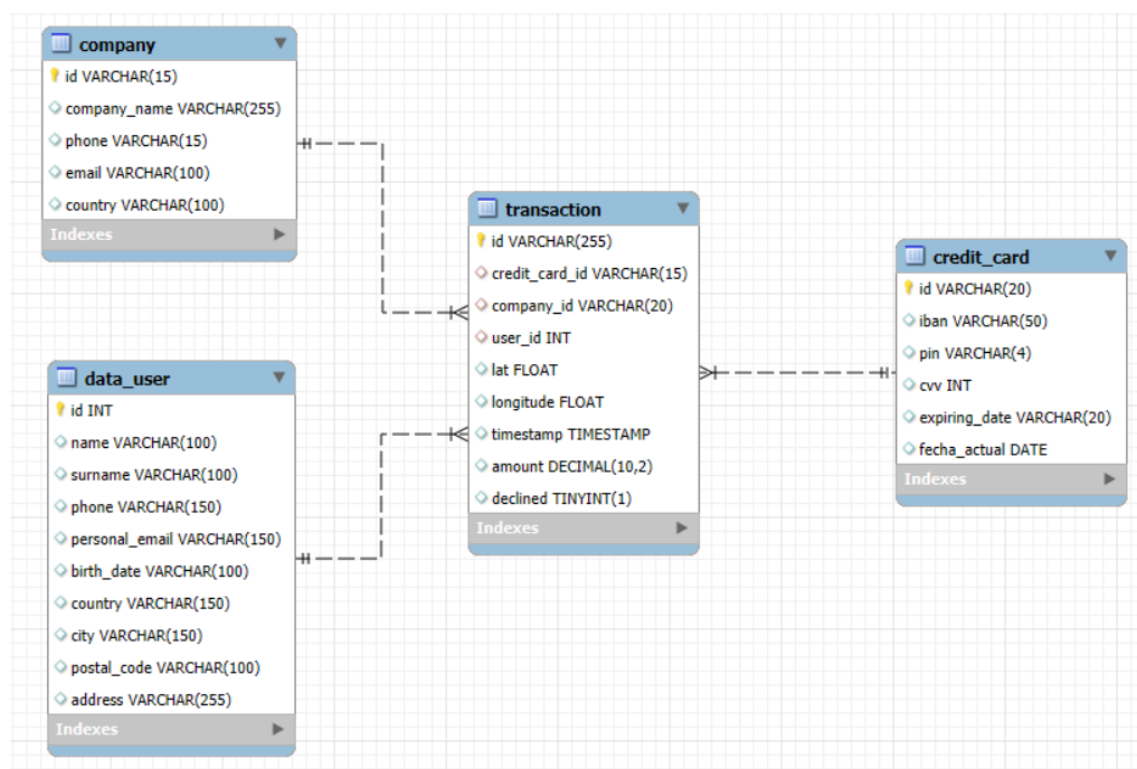

```

• ALTER TABLE credit_card -- Añadimos la columna fecha_actual
  ADD fecha_actual DATE;

```

#	Time	Action	Message
1	11:20:05	ALTER TABLE credit_card -- Añadimos la columna f...	0 row(s) affected Records: 0 Duplicates: 0 Warning...

Hecho todo lo anterior, nos quedará un esquema muy parecido al del ejercicio:



N3E2

-- Ejercicio 2

CREATE VIEW `InformeTecnico` AS

SELECT transaction.id as transaction, name, surname, iban, company_name

FROM transaction

JOIN company

ON (company_id = company.id)

JOIN credit_card

ON (credit_card_id = credit_card.id)

JOIN data_user

ON (user_id = data_user.id)

ORDER BY transaction desc;

select * from informetecnico;

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

	transaction	name	surname	iban	company
	FE96CE47-8D59-381C-4E18-E3CA3D44E8FF	Kenyon	Hartman	DO26854763748537475216568689	Magna A
	FE809ED4-2DB6-55AC-C915-929516E46468	Molly	Gilliam	SE2813123487163628531121	Nunc Inter
	FD9CBCCD-8E1E-8DA1-4606-7E3A6F3A5A65	Linus	Willis	KW9485332754781757886242955643	Nunc Inter
	FD89D51B-AE8D-77DC-E450-88083FBD3187	Hilda	Levy	LT053237077744561475	Malesuada

informetecnico 4

Output

Action Output

#	Time	Action	Message
1	11:45:29	CREATE VIEW `InformeTecnico` AS SELECT tran...	0 row(s) affected
2	11:46:12	select * from informetecnico LIMIT 0, 50000	586 row(s) returned