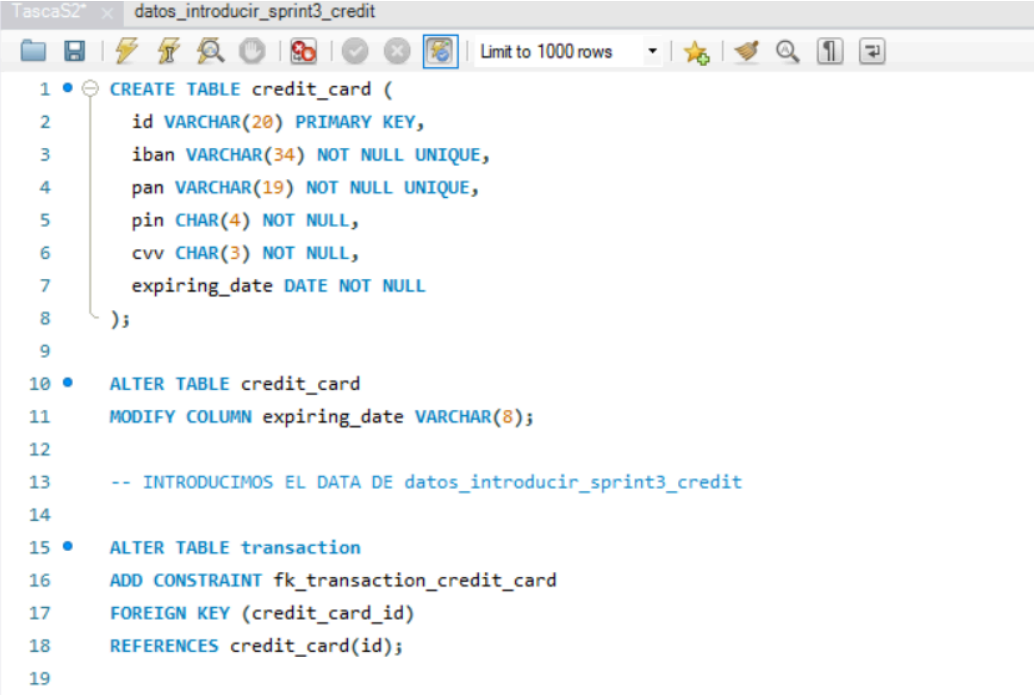


Nivell 1

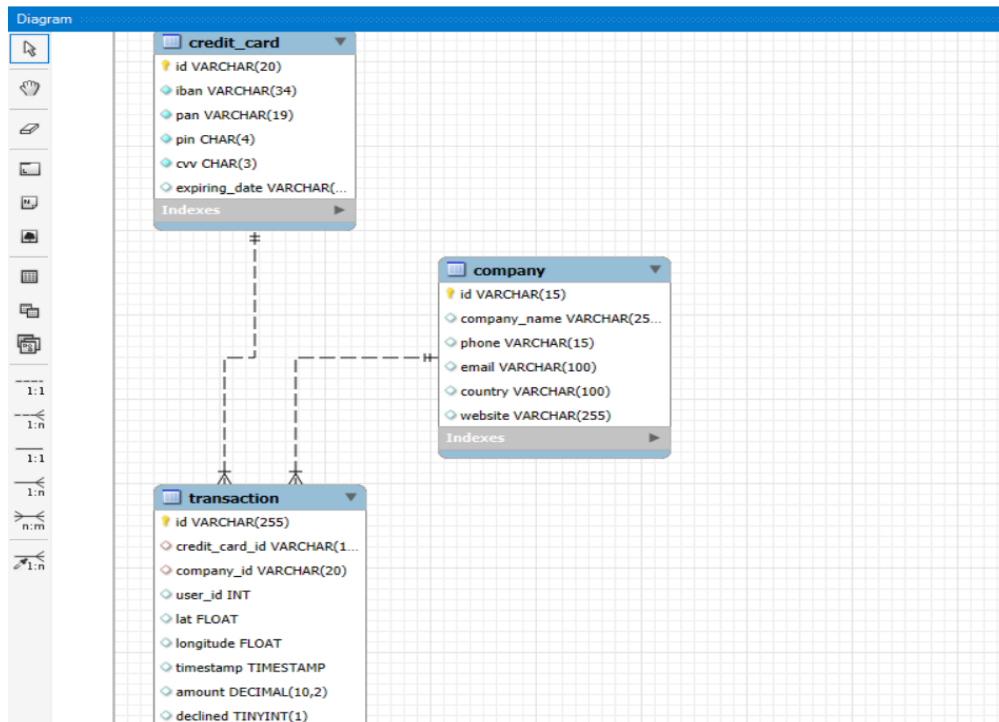
Exercici 1

Creemos la tabla `credit_card` basandonos en lo que nos pide el enunciado y tomando como referencia los “INSERT TABLE” del documento `datos_introducir_sprint3_credit` para luego añadir los datos y que “encajen”.

No me dejaba entrelazar `credit_card_id` (de transaction) con `id` (de credit) debido a que esta última tabla estaba vacía. Lo que he hecho ha sido primero introducir los datos en credit y luego ya modificar la tabla para que `credit_card_id` sea FK .



```
1 CREATE TABLE credit_card (  
2     id VARCHAR(20) PRIMARY KEY,  
3     iban VARCHAR(34) NOT NULL UNIQUE,  
4     pan VARCHAR(19) NOT NULL UNIQUE,  
5     pin CHAR(4) NOT NULL,  
6     cvv CHAR(3) NOT NULL,  
7     expiring_date DATE NOT NULL  
8 );  
9  
10 ALTER TABLE credit_card  
11     MODIFY COLUMN expiring_date VARCHAR(8);  
12  
13 -- INTRODUCIMOS EL DATA DE datos_introducir_sprint3_credit  
14  
15 ALTER TABLE transaction  
16     ADD CONSTRAINT fk_transaction_credit_card  
17     FOREIGN KEY (credit_card_id)  
18     REFERENCES credit_card(id);  
19
```



Exercici 2

Usando un Update y filtrando por el ID que nos da el enunciado modificamos el valor de iban por el que nos dan. Luego con una simple query comprobamos que se han aplicado los cambios

```

23  -- EXERCICI 2
24
25  • UPDATE credit_card
26    SET iban = 'TR323456312213576817699999'
27    WHERE id = 'CcU-2938';
28
29  • SELECT id, iban
30    FROM credit_card
31    WHERE id = 'CcU-2938';
32
  
```

Result Grid

Filter Rows:

Edit:

	id	iban
▶	CcU-2938	TR323456312213576817699999
*	NULL	NULL

Exercici 3

Antes de introducir el nuevo elemento de transaction, hay que añadir una nueva fila de credit_card ya que en transaction, tenemos la FK credit_card_id el cual esta entrelazado con credit_card con la columna id (PK) . Por lo tanto el valor de credit_card_id tiene que existir obligatoriamente en id de credit_card.

Por lo tanto creo la fila de credit_card con el valor que se quiere añadir luego a transaction "CcU-9999".

```
33  -- EXERCICI 3
34
35  • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date)
36  VALUES ('CcU-9999', 'DUMMY_IBAN', '0000000000000000', '0000', '000', '2025-12-31');
37
38  • INSERT INTO transaction (
39      id, credit_card_id, company_id, user_id, lat, longitude, amount, declined, timestamp
40  )
41  VALUES (
42      '10881D1D-5B23-A76C-55EF-C568E49A99DD',
43      'CcU-9999',
44      'b-9999',
45      9999,
46      829.999,
47      -117.999,
48      111.11,
49      0,
50      NOW()
51  );
52
```

Output

Action Output

#	Time	Action	Message
5017	12:16:11	INSERT IGNORE INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-9999', 'DUMMY_IB...	1 row(s) affecte
5018	12:16:30	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined, timesta...	1 row(s) affecte

Exercici 4

Simplemente borramos la columna con el comando DROP COLUMN

```
55 • ALTER TABLE credit_card
56 DROP COLUMN pan;
57
58
```

Output:

#	Time	Action	Message
5018	12:16:30	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined, timesta...	1 row(s) affected
5019	12:21:30	ALTER TABLE credit_card DROP COLUMN pan	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

Nivell 2

Exercici 1

Hacemos un DELETE filtrando por el id. Luego hacemos una búsqueda con el mismo id y vemos que no sale nada. Lo que significa que hemos borrado correctamente la linea.

```
--
59 -- NIVELL 2
60 -- EXERCICI 1
61
62 • DELETE FROM transaction
63 WHERE id = '000447FE-B650-4DCF-85DE-C7ED0EE1CAAD';
64
65 • SELECT * FROM transaction
66 WHERE id = '000447FE-B650-4DCF-85DE-C7ED0EE1CAAD';
67
68
```

Result Grid								
Filter Rows: <input type="text"/>								
Edit: Export/Import: Wrap Cell Content:								
id	credit_card_id	company_id	user_id	lat	longitude	timestamp	amount	declined
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Exercici 2

Creamos la vista como si fuera una simple query que haríamos normalmente pero guardandola como vista en VistaMarketing . Podemos comprobar que en el menu de la izquierda en “Views” nos aparece la vista que hemos creado.

Luego con un simple “SELECT * FROM VistaMarketing;” nos aparece el resultado abajo.

Navigator

Schemas

Filter objects

zakila

sys

transactions

Tables

Views

vistamarketing

Stored Procedures

Functions

world

Administration

Schemas

Information

Schema: transactions

Tasca52

datos_introducir_sprint3_credit

Limit to 1000 rows

```

70
71 • CREATE OR REPLACE VIEW VistaMarketing AS
72 SELECT
73     c.company_name AS nom_empresa,
74     c.phone AS telefon,
75     c.country AS pais,
76     ROUND(AVG(t.amount), 2) AS mitjana_compra
77 FROM
78     company c
79 JOIN
80     transaction t ON c.id = t.company_id
81 GROUP BY
82     c.id, c.company_name, c.phone, c.country
83 ORDER BY
84     mitjana_compra DESC;
85

```

Field Types

#	Field	Schema	Table	Type	Character Set	Display Size	Precision	Scale
1	nom_empresa	transactions	company	VARCHAR	utf8mb4	255	35	0
2	telefon	transactions	company	VARCHAR	utf8mb4	15	14	0
3	pais	transactions	company	VARCHAR	utf8mb4	100	14	0
4	mitjana_compra	transactions	vistamarketing	DECIMAL	binary	13	4	2

VistaMarketingo 3

Output

Action Output

#	Time	Action	Message	Duration / Fetch
5024	12:38:00	CREATE OR REPLACE VIEW VistaMarketing AS SELECT	c.company_name AS nom_empresa, c...	0 row(s) affected 0.000 sec
5025	12:38:09	SELECT * FROM VistaMarketing LIMIT 0.1000	101 row(s) returned	0.281 sec / 0.000 sec

Exercici 3

Usando la vista en el FROM , filtramos por “Germany” como si de una tabla se tratara.

```

90 -- EXERCICI 3
91
92 • SELECT *
93 FROM VistaMarketing
94 WHERE pais = 'Germany';
95

```

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

	nom_empresa	telefon	pais	mitjana_compra
▶	Ac Fermentum Incorporated	06 85 56 52 33	Germany	284.87
	Nunc Interdum Incorporated	05 18 15 48 13	Germany	259.32
	Convallis In Incorporated	06 66 57 29 50	Germany	257.75
	Ac Industries	09 34 65 40 60	Germany	255.15
	Rutrum Non Inc.	02 66 31 61 09	Germany	255.14

VistaMarketingo 6

Nivell 3

Exercici 1

Creamos la tabla e introducimos los datos a partir de los scripts que nos dan. Luego entrelazamos user_id cambiar id de user con user_id de transaction.

```
97      -- NIVELL 3
98      -- EXERCICI 1
99
100
101
102 •   ALTER TABLE transaction
103      MODIFY COLUMN user_id CHAR(10);
104
105
106 •   ALTER TABLE transaction
107      ADD CONSTRAINT fk_transaction_user
108      FOREIGN KEY (user_id) REFERENCES user(id);
109
```

Podemos comprobar que nos queda un esquema tal y como nos pide el enunciado:

