

**Nivel 1**

**Ejercicio 1.** Tu tarea es diseñar y crear una tabla llamada "credit\_card" que almacene detalles cruciales sobre las tarjetas de crédito. La nueva mesa tiene que ser capaz de identificar de manera única cada tarjeta y establecer una relación adecuada con las otras dos mesas ("transaction" y "compañero"). Después de crear la tabla será necesario que ingreses la información del documento denominado "datos\_introducir\_credit". Recuerda mostrar el diagrama y realizar una breve descripción de este.

```

4 -- i "establir una relació adequada amb les altres dues taules" ("transaction" i "company"). Després de
5 -- crear la taula serà necessari que ingressis la informació del document denominat
6 -- "dades_introducir_credit". Recorda mostrar el diagrama i realitzar una breu descripció d'aquest.
7
8 • CREATE TABLE credit_card (
9     id VARCHAR(50) PRIMARY KEY, -- NOT NULL UNIQUE
10    iban VARCHAR(34) UNIQUE,
11    pan VARCHAR(19),
12    pin CHAR(6),
13    cvv CHAR(4),
14    expiring_date VARCHAR(10)
15 );
16
17 -- para modificar el formato de fecha a DATE primero

```

Output:

#	Time	Action	Message	Duration / Fetch
100105	12:40:24	INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, timestamp, amo...	1 row(s) affected	0.000 sec
100106	12:47:34	CREATE TABLE credit_card ( id VARCHAR(50) PRIMARY KEY, - NOT NULL UNIQUE iba...	0 row(s) affected	0.156 sec

Figura 1. Creación de tabla credit\_card

Tabla 1 .Estándares internacionales de longitudes de datos bancarios.

Campo	Estándar	Longitud / reglas
<b>IBAN</b>	ISO 13616	hasta 34 caracteres
<b>PAN</b>	ISO/IEC 7812	hasta 19 dígitos
<b>HIN/BIN</b>	ISO/IEC 7812	6–8 dígitos
<b>PIN</b>	ISO 9564	4–12 dígitos (general: 4 o 6)
<b>CVV/CVC</b>	PCI DSS + reglas de las marcas 3 (Visa/Mastercard), 4 (Amex)	

Campo	Estándar	Longitud / reglas
<b>Expiración</b>	PCI DSS / EMV	MM/YY
<b>BIC/SWIFT</b>	ISO 9362	8 o 11 caracteres

El código crea la tabla **credit\_card** en MySQL definiendo cada campo con su tipo y restricciones: un id como clave primaria, un iban único, y valores como PAN, PIN, CVV y fecha de expiración con longitudes basadas en estándares oficiales. La estructura está preparada para añadir relaciones con las tablas **transaction** y **company**, formando un modelo de datos organizado y consistente.

Figura 2. Agregación de columna *fecha\_actual* y conversión de fecha al formato DATE.

Se agregó columna **fecha\_actual** como tipo DATE y convirtió la fecha almacenada en texto mediante STR\_TO\_DATE. Para poder ejecutar el UPDATE incluyó un WHERE, ya que el *safe mode* de MySQL lo exige.

The screenshot shows a MySQL Workbench interface. In the top panel, there is a code editor window titled "Tarea S3.01" containing SQL code. The code includes comments like "-- cuando está activado el safe mode." and "-- Establezco relación 1:M entre credit\_card y transaction". It also contains an ALTER TABLE statement that adds a FOREIGN KEY constraint named "fk\_credit\_card" to the "transaction" table, linking the "credit\_card\_id" column to the "id" column of the "credit\_card" table. In the bottom panel, there is an "Output" window showing the results of the executed queries. It lists two rows: one for an UPDATE query changing the expiration date of 5000 credit cards, and another for the ALTER TABLE query itself, which affected 100000 records. Both queries completed successfully with 0 warnings in 0.422 seconds and 5.328 seconds respectively.

Figura 3. Creación de relación entre la tabla **transaction** y **credit\_card**

Se estableció la relación entre la tabla **transaction** y **credit\_card** agregando una clave foránea mediante ALTER TABLE. Para ello creó la restricción **fk\_credit\_card**, la cual enlaza el campo **credit\_card\_id** de *transaction* con el campo **id** de *credit\_card*. Con esta acción definió una relación de **uno a muchos**, donde una tarjeta puede tener múltiples transacciones asociadas, garantizando la integridad referencial dentro de la base de datos.

## EJERCICIO 2

El departamento de Recursos Humanos ha identificado un error en el número de cuenta asociado a la tarjeta de crédito con ID CcU-2938. La información que tiene que mostrarse para este registro es: TR323456312213576817699999. Recuerda mostrar que el cambio se realizó.

```

41 -- és: TR323456312213576817699999. Recorda mostrar que el canvi es va realitzar.
42
43 • SELECT *
44 FROM credit_card
45 WHERE id = 'CcU-2938';
46

```

The screenshot shows the MySQL Workbench interface. On the left, there's a tree view of database objects under the 'transactions' schema. In the main pane, a 'Result Grid' shows one row for the credit\_card table:

id	iban	pan	pin	cvv	expiring_date	fecha_actual
CcU-2938	TR301950312213576817638661	5424465566813633	3257	984	10/30/22	2022-10-30

Below the grid, the 'Output' section shows the execution log:

```

credit_card 1 ×
Output:
Action Output
# Time Action
1 105108 12:53:08 UPDATE credit_card SET fecha_actual = STR_TO_DATE(expiring_date, "%m/%d/%y") WHERE id = 'CcU-2938';
2 105109 06:50:03 ALTER TABLE transaction ADD CONSTRAINT fk_credit_card FOREIGN KEY(credit_card_id) REFERENCES credit_card(id);
3 105110 06:51:28 SELECT * FROM credit_card WHERE id = 'CcU-2938';

```

Figura 4. Identificación del error en el IBAN de la tarjeta con ID **CcU-2938**

Detectado el error en el IBAN de la tarjeta con ID **CcU-2938**, figura 4, se procedió a corregirlo mediante una sentencia UPDATE que reemplazó el valor anterior por el nuevo número proporcionado por Recursos Humanos. Después ejecutó un SELECT filtrado por el mismo ID para comprobar que el cambio se aplicó correctamente., ver figura 5. Al visualizar nuevamente el registro en la tabla *credit\_card*, pudo confirmar que el IBAN fue actualizado sin afectar el resto de los datos asociados a la tarjeta.

```

48
49 • UPDATE credit_card
50 SET iban = 'TR323456312213576817699999'
51 WHERE id = 'CcU-2938';
52
53 • SELECT *
54 FROM credit_card
55 WHERE id = 'CcU-2938';
56

```

The screenshot shows the MySQL Workbench interface. The 'Result Grid' shows the updated record:

id	iban	pan	pin	cvv	expiring_date	fecha_actual
CcU-2938	TR323456312213576817699999	5424465566813633	3257	984	10/30/22	2022-10-30

The 'Output' section shows the execution log:

```

credit_card 2 credit_card 3 ×
Output:
Action Output
# Time Action
1 105111 06:52:24 UPDATE credit_card SET iban = 'TR323456312213576817699999' WHERE id = 'CcU-2938'
2 105112 06:52:41 SELECT * FROM credit_card WHERE id = 'CcU-2938'
3 105113 06:52:41 SELECT * FROM credit_card WHERE id = 'CcU-2938'

```

Figura 5 cambio datos ya existentes en registro específico de una tabla *credit\_card* y comprobación de cambios..

### EJERCICIO 3

En la tabla "transaction" ingresa una nueva transacción con la siguiente información:

Tabla 2. Datos de transacción a ingresar.

<i>id</i>	108B1D1D-5B23-A76C-55EF-C568E49A99DD
-----------	--------------------------------------

<i>credit_card_id</i>	CcU-9999
<i>company_id</i>	b-9999
<i>user_id</i>	9999
<i>lat</i>	829.999
<i>longitude</i>	-117.999
<i>amount</i>	111.11
<i>declined</i>	0

```

b4
65 • INSERT INTO company (id)
66   VALUES ('b-9999');
67
68 • INSERT INTO credit_card (id)
69   VALUES ('CcU-9999');
70
71 -- Incerto los datos, coloco NULL en timestamp pq no me proporcionan la fecha y
72 • INSERT INTO transaction(id,credit_card_id,company_id,user_id,lat,longitude,timestamp ,amount,declined )
73   VALUES ('108B1D1D-5B23-A76C-55EF-C568E49A99DD','CcU-9999','b-9999',9999,829.999,-117.999,NULL,111.11,0);
74
75
76

```

Output					
#	Time	Action	Message	Duration / Fetch	
1	06:59:35	INSERT INTO transaction(id,credit_card_id,company_id,user_id,lat,longitude,timestamp ,amount,declined ) VALUES ('108B1D1D-5B23-A76C-55EF-C568E49A99DD','CcU-9999','b-9999',9999,829.999,-117.999,NULL,111.11,0);	Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails ('transactions'.`tr...`)	0.016 sec	
2	07:05:26	INSERT INTO credit_card (id) VALUES ('CcU-9999')	1 row(s) affected	0.031 sec	
3	07:05:39	INSERT INTO transaction(id,credit_card_id,company_id,user_id,lat,longitude,timestamp ,amount,declined ) VALUES ('108B1D1D-5B23-A76C-55EF-C568E49A99DD','CcU-9999','b-9999',9999,829.999,-117.999,NULL,111.11,0);	1 row(s) affected	0.046 sec	

Activar Windows  
Ve a Configuración para activar Windows.

Figura 6. Ingreso de nueva transacción

Se intentó insertar el registro inicialmente, pero el sistema devolvió el error 1452, lo que indicó que todavía no existían los valores correspondientes al *credit\_card\_id* o *company\_id* requeridos por la transacción. Para cumplir con las claves foráneas, primero se insertaron los datos de una tarjeta en la tabla *credit\_card* con el id 'CcU-9999' y los de una compañía en la tabla *company* con el id 'b-9999'. Una vez creados estos registros, se añadió la nueva transacción mediante *INSERT INTO transaction*, verificando que *credit\_card\_id* y *company\_id* coincidieran con los valores ya existentes. Al corregir el orden de inserción, se evitó el error de integridad referencial y se pudo registrar

correctamente la operación con todos sus campos, incluyendo coordenadas, monto e indicador de rechazo.

#### EJERCICIO 4

Desde recursos humanos te solicitan eliminar la columna "pan" de la mesa credit\_card. Recuerda mostrar el cambio realizado.

The screenshot shows the MySQL Workbench interface. In the SQL editor, the following commands were run:

```
81 • ALTER TABLE credit_card
82   DROP COLUMN pan;
83   |
84 • DESCRIBE credit_card;
```

The results grid displays the current structure of the 'credit\_card' table:

Field	Type	Null	Key	Default	Extra
id	varchar(50)	NO	PRI	NULL	
iban	varchar(34)	YES	UNI	NULL	
pin	char(6)	YES		NULL	
cvv	char(4)	YES		NULL	
expiring_date	varchar(10)	YES		NULL	
fecha_actual	date	YES		NULL	

The status bar indicates the operation was successful, with 1 row affected and 0.046 sec duration.

Figura 7.- Eliminación de la columna PAN

Para atender la solicitud de Recursos Humanos, se procedió a eliminar la columna pan de la tabla *credit\_card* utilizando el comando `ALTER TABLE credit_card DROP COLUMN pan;`. Después se ejecutó `DESCRIBE credit_card` para mostrar la estructura actualizada de la tabla y confirmar que la columna había sido removida correctamente, tal y como lo muestra la figura 7. Con esto, dejó la tabla ajustada a los nuevos requerimientos sin afectar el resto de los datos existentes.

**NIVEL 2****EJERCICIO 1**

Elimina de la Tabla transaction el registro con ID 000447FE-B650-4DCF-85DE-C7ED0EE1CAAD de la base de datos.

The screenshot shows the MySQL Workbench interface. In the central query editor window, there is a SQL script with the following content:

```
-- de la base de datos.
SELECT id
FROM transaction
WHERE id = '000447FE-B650-4DCF-85DE-C7ED0EE1CAAD';
DELETE FROM transaction WHERE id = '000447FE-B650-4DCF-85DE-C7ED0EE1CAAD';
```

Below the script, a Result Grid shows a table with one row, labeled 'id' and containing the value 'NULL'. At the bottom of the interface, the Output pane displays the execution log:

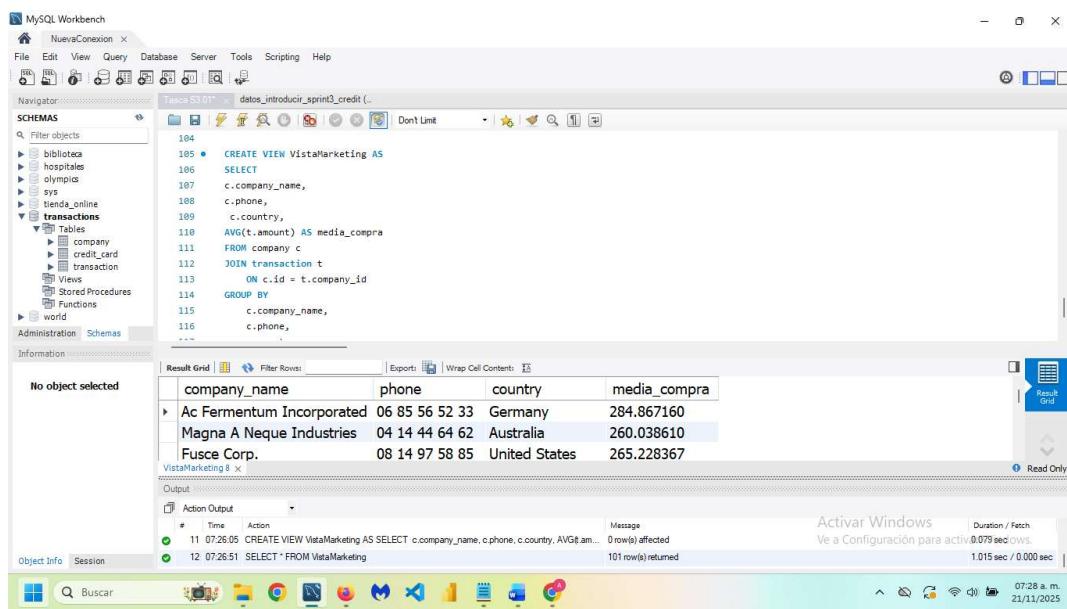
#	Time	Action	Message	Duration / Fetch
7	07:08:40	DESCRIBE credit_card	6 row(s) returned	0.016 sec / 0.000 sec
8	07:12:35	SELECT id FROM transaction WHERE id = '000447FE-B650-4DCF-85DE-C7ED0EE1CAAD'	1 row(s) returned	0.000 sec / 0.000 sec
9	07:17:55	DELETE FROM transaction WHERE id = '000447FE-B650-4DCF-85DE-C7ED0EE1CAAD'	1 row(s) affected	0.063 sec / 0.000 sec
10	07:18:44	SELECT id FROM transaction WHERE id = '000447FE-B650-4DCF-85DE-C7ED0EE1CAAD'	0 row(s) returned	0.000 sec / 0.000 sec

Figura 8. Eliminación de la Tabla transaction registro con ID 000447FE-B650-4DCF-85DE-C7ED0EE1CAAD

Se procedió a localizar el registro de la tabla transaction donde id era 000447FE-B650-4DCF-85DE-C7ED0EE1CAAD mediante una consulta SELECT, verificando así que el dato existía. Una vez confirmado, ejecutó la sentencia DELETE FROM transaction con el mismo identificador para eliminarlo de la base de datos. Finalmente se volvió a ejecutar un SELECT para comprobar que el registro ya no aparecía, ver figura 8, confirmando que la eliminación se realizó correctamente.

## EJERCICIO 2

La sección de marketing desea tener acceso a información específica para realizar análisis y estrategias efectivas. Se ha solicitado crear una vista que proporcione detalles clave sobre las compañías y sus transacciones. Será necesaria que crees una vista llamada VistaMarketing que contenga la siguiente información: Nombre de la compañía. Teléfono de contacto. País de residencia. Media de compra realizado por cada compañía. Presenta la vista creada, ordenando los datos de mayor a menor media de compra.



```
CREATE VIEW VistaMarketing AS
SELECT
    c.company_name,
    c.phone,
    c.country,
    AVG(t.amount) AS media_compra
FROM company c
JOIN transaction t
ON c.id = t.company_id
GROUP BY
    c.company_name,
    c.phone,
```

company_name	phone	country	media_compra
Ac Fermentum Incorporated	06 85 56 52 33	Germany	284.867160
Magna A Neque Industries	04 14 44 64 62	Australia	260.038610
Fusce Corp.	08 14 97 58 85	United States	265.228367

Figura 9. Creación de vista VistaMarketing

En la figura 9 se observa que se creó la vista VistaMarketing para proporcionar al área de marketing información relevante sobre cada compañía y sus transacciones. La vista muestra el nombre de la empresa, su teléfono de contacto, el país de residencia y la media de compra obtenida a partir de sus transacciones, utilizando una combinación de las tablas company y transaction. Después de generar la vista, se ordenó los resultados de mayor a menor según la media de compra, permitiendo que el departamento de marketing visualice de inmediato qué compañías realizan mayores volúmenes de compra y facilitando así sus análisis y estrategias.

### EJERCICIO 3

Filtra la vista VistaMarketing para mostrar solo las compañías que tienen su país de residencia en "Germany"

The screenshot shows a database query interface with the following details:

Query (SQL):

```
119
120 • SELECT *
121   FROM VistaMarketing
122   ORDER BY media_compra DESC;
123
124
125
126
127 -- FILTERED 3
```

Result Grid:

company_name	phone	country	media_compra
Ac Fermentum Incorporated	06 85 56 52 33	Germany	284.91
Pretium Neque Corp.	07 77 48 55 28	Australia	275.58
Urna Convallis Associates	06 01 24 77 04	United States	273.57

Output:

#	Time	Action	Message	Duration / Fetch
100344	12:13:04	SELECT * FROM VistaMarketing WHERE country = 'Germany' ORDER BY media_com... 8 row(s) returned	8 row(s) returned	0.219 sec / 0.000 sec
100345	12:14:41	SELECT * FROM VistaMarketing ORDER BY media_compra DESC	101 row(s) returned	1.109 sec / 0.000 sec

Figura 10 Filtro de la vista VistaMarketing por país “Germany”.

Filtró la vista **VistaMarketing** para mostrar exclusivamente las compañías cuyo país de residencia es *Germany*, utilizando una sentencia SELECT con la condición WHERE country = 'Germany'. Luego ordenó los resultados de mayor a menor según la media de compra.

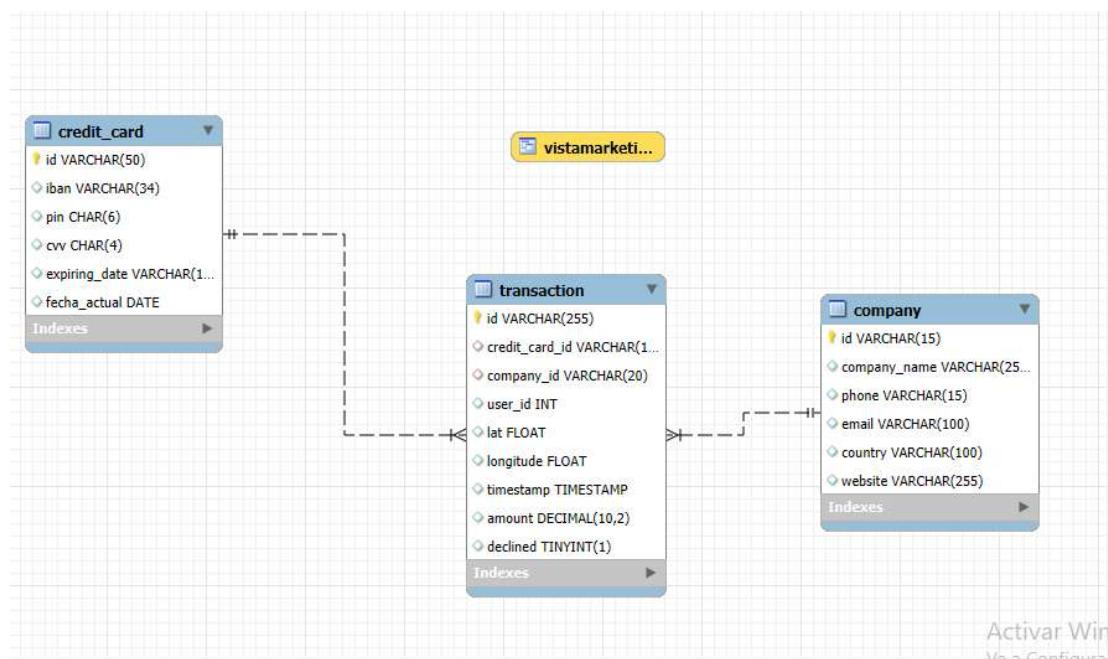


Figura 11. Diagrama resultado del nivel 2

**NIVEL 3****Ejercicio 1**

La próxima semana tendrás una nueva reunión con los gerentes de marketing. Un compañero de tu equipo realizó modificaciones en la base de datos, pero no recuerda como las realizó. Te pide que lo ayudes a dejar los comandos ejecutados para obtener el siguiente diagrama:

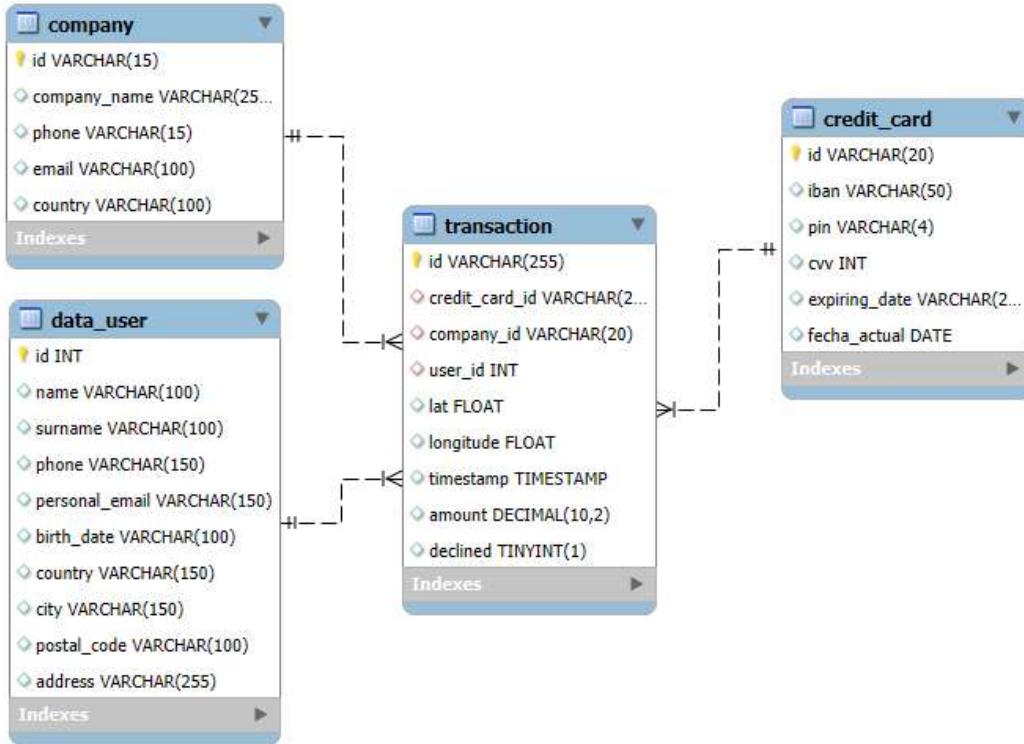


Figura 12. Diagrama ejercicio nivel 3

Para verificar los cambios realizados se verificó el diagrama actual con el diagrama resultados hechos hasta el nivel 2. Evidenciándose la creación de una nueva tabla inicialmente, data\_user

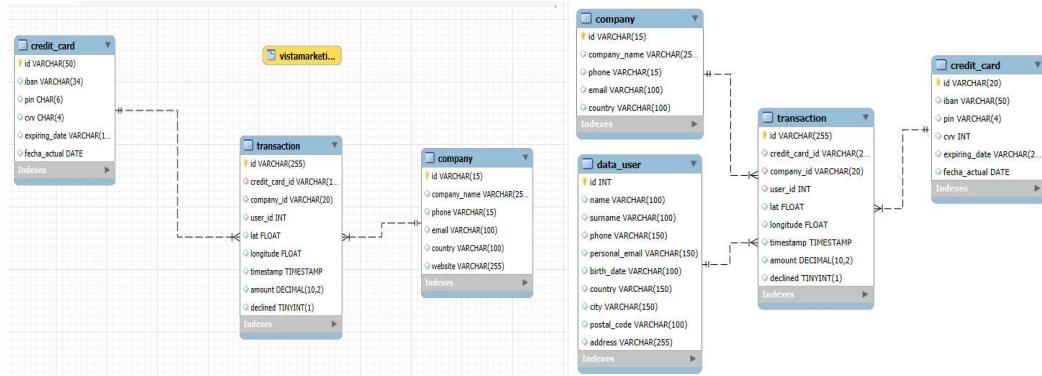


Figura 13. Diagramas antes y después de los cambios.

Se evidenciaron los siguientes cambios entre los diagramas, ver tabla 3

*Tabla 3. Cambios realizados al diagrama 1 evidenciados en diagrama 2*

Elemento	Diagrama 1	Diagrama 2	Cambio realizado
Nueva tabla <b>agregada</b>	No existe tabla data_user	Aparece tabla data_user	Se agregó una nueva entidad llamada data_user con información personal del usuario.
<b>Campos de la tabla data_user</b>	—	id, name, surname, phone, personal_email, birth_date, country, postal_code, address	Nueva estructura completa de datos del usuario.
<b>Relación transaction → data_user</b>	transaction.user_id existe pero no tenía referencia visual	transaction.user_id se conecta claramente a data_user.id	Se formalizó la relación entre transacciones y usuarios.
<b>Tabla credit_card</b>	id (VARCHAR50), iban (VARCHAR24), pin (CHAR5), cvv (CHAR5), expiring_date VARCHAR/fecha_actual DATE	id VARCHAR(20), iban VARCHAR(50), pin VARCHAR(4), cvv INT, expiring_date VARCHAR(20), fecha_actual DATE	Cambios en tipos de datos y longitudes (reducción en id, ampliación de iban, ajuste de pin y cvv).
<b>Tabla company</b>	Campos iguales pero sin orden estandarizado	Igual estructura pero organizada en nuevo orden  No aparece el campo website	Reorganización de campos, sin cambios de estructura.  Se borra el campo website
<b>Relación transaction → company</b>	Existía vínculo	Se mantiene vínculo	Se mantiene la relación
<b>Tipos de datos en transaction</b>	id VARCHAR(255), lat FLOAT, long FLOAT, timestamp TIMESTAMP, amount DECIMAL(10,2), declined TINYINT(1)	Igual, sin cambios	Sin modificaciones.
<b>Relación transaction → credit_card</b>	credit_card_id VARCHAR(?) apuntaba a credit_card.id	Se mantiene igual	Sin cambios.

Se comenzó creando la data data\_user, sin embargo, inicialmente no se creó con ese nombre, sino con la estructura proporcionada en los archivos adjuntos del ejercicio para poder introducir los valores proporcionados y luego hacer los cambios sobre la tabla user, ver figura 14.

```

136 -- Como los datos proporcionados para introducir no coinciden con el nombre de
137 -- tabla del diagrama proporcionado, creará la tabla con la misma estructura de la original y
138 -- luego se realizarán los cambios para ajustarlo al diagrama
139 -- Creación tabla data_user
140 CREATE TABLE IF NOT EXISTS user (
141     id int PRIMARY KEY,
142     name VARCHAR(100),
143     surname VARCHAR(100),
144     phone VARCHAR(150),
145     email VARCHAR(150),
146     birth_date VARCHAR(100),
147     country VARCHAR(150),
148     city VARCHAR(150),
149     postal_code VARCHAR(100),
150     address VARCHAR(255)
151 );
152
153 SHOW CREATE TABLE transaction;
154
Output
Action Output
# Time Action Message Duration / Fetch
4503 06:25:24 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ('1', 'Juan', 'Perez', '1234567890', 'juan.perez@example.com', '1990-01-01', 'Argentina', 'Buenos Aires', '12345', 'Calle 123') 1 row(s) affected 0.016 sec
4504 06:25:24 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ('2', 'Maria', 'Garcia', '0987654321', 'maria.garcia@example.com', '1985-05-20', 'Argentina', 'CABA', '12345', 'Calle 456') 1 row(s) affected 0.015 sec
4505 06:25:24 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ('3', 'Pedro', 'Sanchez', '1234567890', 'pedro.sanchez@example.com', '1992-08-15', 'Argentina', 'Mendoza', '12345', 'Calle 789') 1 row(s) affected 0.000 sec
4506 06:25:24 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ('4', 'Ana', 'Rodriguez', '0987654321', 'ana.rodriguez@example.com', '1988-03-10', 'Argentina', 'CABA', '12345', 'Calle 456') 1 row(s) affected 0.016 sec
Activar Windows
Ve a Configuración para activar Windows.

```

Figura 14. Creación de tabla user

En la figura 15 podemos observar el cambio del nombre de la tabla user a data\_user y de la columna email a personal\_email

```

155 RENAME TABLE user TO data_user;
156 ALTER TABLE data_user
157 RENAME COLUMN email TO personal_email;
158
159 -- transaction + data user (muchas transacciones por usuario)
Output
Action Output
# Time Action Message Duration / Fetch
5001 06:25:30 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ('1', 'Juan', 'Perez', '1234567890', 'juan.perez@example.com', '1990-01-01', 'Argentina', 'Buenos Aires', '12345', 'Calle 123') 1 row(s) affected 0.031 sec
5002 06:25:30 INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address) VALUES ('2', 'Maria', 'Garcia', '0987654321', 'maria.garcia@example.com', '1985-05-20', 'Argentina', 'CABA', '12345', 'Calle 456') 1 row(s) affected 0.016 sec
5003 06:34:39 RENAME TABLE user TO data_user 0 row(s) affected 0.110 sec
5004 06:36:08 ALTER TABLE data_user RENAME COLUMN email TO personal_email 0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0 0.016 sec
Activar Windows
Ve a Configuración para activar Windows.

```

Figura 15.Cambio de nombre de tabla user a data\_user y de la columna email a personal\_email

En la figura 16 se puede observar la creación de la relación transaction →data\_user (muchas transacciones por usuario).

The screenshot shows the MySQL Workbench interface. In the code editor, lines 213 to 226 are visible, creating a relationship between the transaction and data\_user tables. The output window shows the execution of these statements, including the creation of the table credit\_card and the addition of constraints. A message at the bottom right of the output window says 'Vea la Configuración para activar Windows.'

```

213
214    -- transaction → data_user (muchas transacciones por usuario)
215
216 • ALTER TABLE transaction
217     ADD CONSTRAINT fk_transaction_user
218     FOREIGN KEY (user_id)
219     REFERENCES data_user(id)
220     ON DELETE CASCADE
221     ON UPDATE CASCADE;
222
223
224
225
226    -- EXERCICI 2

```

Action Output	#	Time	Action	Message	Duration / Fetch
	3	05:45:58	CREATE TABLE IF NOT EXISTS credit_card ( id VARCHAR(20) PRIMARY KEY, ... )	0 row(s) affected, 1 warning(s); 1050 Table 'credit_card' already exists	0.031 sec
	4	05:47:36	CREATE TABLE IF NOT EXISTS transaction ( id VARCHAR(255) PRIMARY KEY, ... )	0 row(s) affected, 2 warning(s); 1601 Integer display width is deprecated and will be removed in a future release	0.016 sec
	5	05:49:12	ALTER TABLE transaction ADD CONSTRAINT fk_transaction_credit_card FOREIGN KEY ( user_id ) REFERENCES data_user(id) ON DELETE CASCADE ON UPDATE CASCADE;	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	1.422 sec
	6	05:50:12	ALTER TABLE transaction ADD CONSTRAINT fk_transaction_company FOREIGN KEY ( company ) REFERENCES company(id)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	1.297 sec

Figura 16. Creación de relación transaction → data\_user (muchas transacciones por usuario)

El siguiente paso fue hacer los cambios en la tabla credit\_card estos fueron de tipo de datos y longitudes, ver figura 17.

The screenshot shows the MySQL Workbench interface. Lines 173 to 184 show the modification of the credit\_card table, changing various fields from VARCHAR to different types and sizes. Line 183 shows a query to select distinct user\_ids from the transaction table. The output window shows the execution of these statements, with a message at the bottom right saying 'Activar Windows'.

```

173
174    -- 2. Ajustes de tipos de datos y longitudes
175 • ALTER TABLE credit_card
176     MODIFY id VARCHAR(20),
177     MODIFY iban VARCHAR(50),
178     MODIFY pin VARCHAR(4),
179     MODIFY cvv INT,
180     MODIFY expiring_date VARCHAR(20),
181     MODIFY fecha_actual DATE;
182
183 • SELECT DISTINCT user_id
184   FROM transaction

```

Action Output	#	Time	Action	Message	Duration / Fetch
	1	07:01:38	ALTER TABLE credit_card MODIFY id VARCHAR(20), MODIFY iban VARCHAR(50), MODIFY pin VARCHAR(4), MODIFY cvv INT, MODIFY expiring_date VARCHAR(20), MODIFY fecha_actual DATE	1 row(s) affected Records: 1 Duplicates: 0 Warnings: 0	0.156 sec

Figura 17. Modificación de longitudes y tipo de datos

Se evaluó la integridad referencial garantizar que las relaciones entre tablas sean **coherentes y válidas**, debido a que en ejercicios anteriores se realizaron modificaciones y se necesita saber si hay datos huérfanos, ver código en figura 18. En este caso se encontró que en la tabla data\_user no está el id =9999 y se procede a insertar, ya que la tabla padre (data\_user) debe tener los mismos id que en la tabla hija (transaction) y así crear la FK.

Es importante destacar que inicialmente se había presentado la situación de que se pudo crear la FK entre data\_user y transaction sin hacer la validación referencial lo que no debe suceder si están activas las FK. Es por este sentido se borra la FK creada inicialmente y

se activan las restricciones de las FK y se realiza correctamente la adición de la FK correctamente, ver figura 19.

```

182 -- Validar la integridad referencial o limpiar datos erróneos, ya que en ejercicios anteriores
183 -- se realizaron modificaciones en las tablas.
184
185 • SELECT DISTINCT user_id
186   FROM transaction
187   WHERE user_id IS NOT NULL
188   AND user_id NOT IN (SELECT id FROM data_user);
189
190 -- EXERCICI 2-L'empresa també us demana crear una vista anomenada "InformeTecnico" que

```

The screenshot shows the MySQL Workbench interface. At the top, there's a code editor with the above SQL query. Below it is a result grid showing a single row with 'user\_id' containing '9999'. The bottom part is the 'transaction 10' output pane, which displays the query 'SELECT DISTINCT user\_id FROM transaction WHERE user\_id IS NOT NULL AND user\_id NOT IN (SELECT id FROM data\_user);' along with its execution time and message '1 row(s) returned'.

Figura 18. Evaluación de integridad referencial

Evidentemente no me debió dejar crear la FK y es por eso que se evidencia que estaban desactivadas las restricciones de foreign key ya que obliga a que cada transaction.user\_id exista en data\_user.id y por tanto daría error.

```

190
191 • SELECT @@FOREIGN_KEY_CHECKS;
192 • SET FOREIGN_KEY_CHECKS = 1;
193
194 • ALTER TABLE transaction
195   ADD CONSTRAINT fk_transaction_user
196   FOREIGN KEY (user_id)
197   REFERENCES data_user(id)
198   ON DELETE CASCADE
199   ON UPDATE CASCADE;
200
201
202

```

The screenshot shows the MySQL Workbench interface. It displays three rows in the output pane corresponding to the queries in the code block. Row 1 shows the result of 'SELECT @@FOREIGN\_KEY\_CHECKS;'. Row 2 shows the result of 'SET FOREIGN\_KEY\_CHECKS = 1;'. Row 3 shows the result of the 'ALTER TABLE' command, which adds a constraint 'fk\_transaction\_user' to the 'transaction' table, referencing 'data\_user(id)' with cascade options. The output pane also includes performance metrics like duration and fetch times.

Figura 19. Activación de las restricciones de la FK y creación de la FK entre transacción y data\_user

The screenshot shows the MySQL Workbench interface. In the left pane, there is a table named 'transaction' with columns: id, credit\_card\_id, company\_id, user\_id, lat, longitude, timestamp, amount, and deleted. The 'deleted' column is defined as tinyint(1). In the main pane, a query is being run:

```

198 -- se borra de company campo website
199 alter table company
200 drop column website;
201
202

```

The output pane shows the results of the query:

#	Time	Action	Message	Duration / Fetch
2	07:46:03	selected * from company	101 row(s) returned	0.031 sec / 0.000 sec
3	07:50:49	alter table company drop column website	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.125 sec

At the bottom right, there is a message: "Activar Windows Ve a Configuración para activar Windows. 0.031 sec / 0.000 sec".

Figura 20. Drop de la columna website

Adicionalmente en los cambios se puede observar que en la tabla company no existe el campo website

The screenshot shows the MySQL Workbench interface. In the main pane, a query is being run:

```

203
204 -- Modificar el tipo de dato en tabla data_user
205 alter table data_user
206 modify personal_email varchar(150);
207

```

The output pane shows the results of the query:

#	Time	Action	Message	Duration / Fetch
1	08:12:39	alter table data_user modify personal_email varchar(150)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.093 sec

At the bottom right, there is a message: "Activar Windows Ve a Configuración para activar Windows. 0.093 sec".

Figura 21. Modificación de longitud de dato en la tabla data\_user del campo personal\_email

En la tabla 21 podemos observar el cambio de la longitud de dato de personal\_email

## EJERCICIO 2.

La empresa también os pide crear una vista llamada "InformeTecnico" que contenga la siguiente información:

- ID de la transacción
- Nombre del usuario/aria
- Apellido del usuario/aria
- IBAN de la tarjeta de crédito usada.
- Nombre de la compañía de la transacción realizada.

Aseguraos de incluir información relevante de las tablas que conoceréis y utilizáis alias para cambiar de nombre columnas según haga falta.

Muestra los resultados de la vista, ordena los resultados de forma descendente en función de la variable ID de transacción.

Se creó la vista y se llamó para mostrar la tabla con los resultados. Ver figura 22 y 23.

The screenshot shows the MySQL Workbench interface. In the top pane, a SQL editor window displays the creation of a view named 'InformeTecnico'. The code is as follows:

```
221 • CREATE VIEW InformeTecnico AS
222     SELECT
223         t.id AS transaction_id,
224         u.name AS user_name,
225         u.surname AS user_surname,
226         c.iban AS credit_card_ibан,
227         comp.company_name AS company_name
228     FROM transaction t
229     JOIN data_user u
230         ON t.user_id = u.id
231     JOIN credit_card c
232         ON t.credit_card_id = c.id
233     JOIN company comp
234         ON t.company_id = comp.id
235     WHERE declined = 0;
236
```

In the bottom pane, the 'Output' tab shows the execution results. It includes a table titled 'InformeTecnico 37' with the following data:

	transaction_id	user_name	user_surname	credit_card_iban	company_name
▶	00043A49-2949-494B-A5D...	Gxnmjn	Fjycossj	XX6917990232547131536...	Eget Tincidunt Duis Institute
	00045D68-ED2E-4F2F-818...	Wolake	Ukynumly	XX7152198092627919352...	Neque Tellus Imperdiet C...
	000481C3-1C26-4FEF-83A...	Umhwoi	Ubmdxvmz	XX7850228297131408807...	Fusce Corp.
	00051A49-9CBE-4268-807...	Qpfyfa	Mtnpdfq	XX6346265645368759343...	Mus Aenean Eget Founda...

Figura 22. Creación de la vista "InformeTecnico"

The screenshot shows the MySQL Workbench interface. In the top pane, a SQL editor window displays a SELECT query from the 'InformeTecnico' view. The code is as follows:

```
234     ON t.company_id = comp.id
235     WHERE declined = 0;
236
237 •     SELECT *
238     FROM InformeTecnico;
239
240
241
```

In the bottom pane, the 'Output' tab shows the execution results. It includes a table titled 'InformeTecnico 37' with the following data:

	transaction_id	user_name	user_surname	credit_card_iban	company_name
▶	00043A49-2949-494B-A5D...	Gxnmjn	Fjycossj	XX6917990232547131536...	Eget Tincidunt Duis Institute
	00045D68-ED2E-4F2F-818...	Wolake	Ukynumly	XX7152198092627919352...	Neque Tellus Imperdiet C...
	000481C3-1C26-4FEF-83A...	Umhwoi	Ubmdxvmz	XX7850228297131408807...	Fusce Corp.
	00051A49-9CBE-4268-807...	Qpfyfa	Mtnpdfq	XX6346265645368759343...	Mus Aenean Eget Founda...

Figura 23. Muestra de "InformeTecnico"

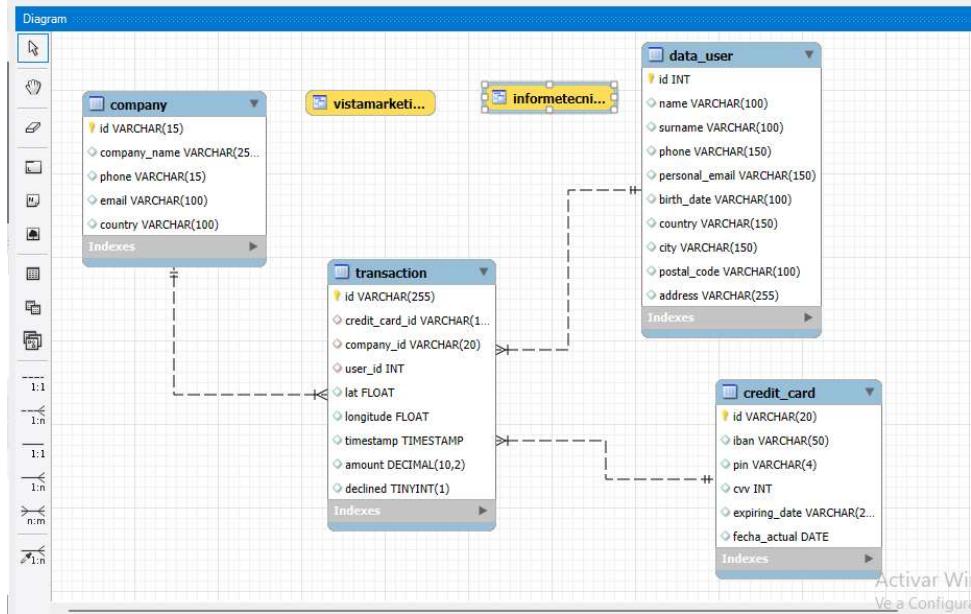


Figura 24. Diagrama Nivel 3