

Tarea S3.01. Manipulación de tablas

1. Nivel 1	1
1.1. Ejercicio 1	1
• Creación de la tabla credit_card	1
• Insertar datos	3
• Creación de las relaciones	5
• Diagrama ER	5
1.2. Ejercicio 2	7
• Modificar de datos	7
• Mostrar cambio realizado	7
1.3. Ejercicio 3	8
• Insertar nuevo registro en transaction	8
1.4. Ejercicio 4	14
• Eliminar columna	14
• Comprobar cambio realizado	14
2. Nivel 2	15
2.1. Ejercicio 1	15
• Eliminar registro	15
2.2. Ejercicio 2	16
• Crear la vista	16
• Mostrar resultados de la vista	17
2.3. Ejercicio 3	18
3. Nivel 3	19
3.1. Ejercicio 1	19
• Comparación con modelo anterior y definición de modificaciones necesarias por tablas	20
• Creación de tabla data_user, ingesta de datos y relación con tabla transaction..	21
• Modificaciones en tablas company y credit_card	27
• Generación del Modelo ER	28
3.2. Ejercicio 2	28
• Crear la vista	29
• Mostrar resultados de la vista	29

1. Nivel 1

1.1. Ejercicio 1

Diseñar y crear una tabla llamada "`credit_card`" que almacene detalles cruciales sobre las tarjetas de crédito. La nueva tabla debe ser capaz de identificar de forma única cada tarjeta y establecer una relación adecuada con las otras dos tablas ("`transaction`" y "`company`"). 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 del mismo.

- Creación de la tabla `credit_card`

Definición de atributos

- Identificador único de la tarjeta (PAN)
- Nombre del titular de la tarjeta
- Código IBAN único por tarjeta
- Código PIN secreto de 4 dígitos
- Número principal de la tarjeta
- Código de seguridad de la tarjeta
- Fecha de vencimiento de la tarjeta

Aspectos considerados

- Para diseñar la tabla `credit_card`, se tomaron en cuenta los datos del archivo "`datos_introducir_credit`".
- Inicialmente fue considerado el campo `nombre`, pero como no estaba presente en los datos del archivo, quedó comentado en el código.
- Se utilizó `VARCHAR` para las columnas `id`, `iban`, `pan`, `pin` y `cvv` debido a su naturaleza alfanumérica. Las longitudes fueron calculadas en función de los datos observados, y para los campos `iban`, `pan`, `cvv` y `pin`, se siguieron sugerencias basadas en estándares internacionales: 34 para `iban`, 19 para `pan`, 4 para `cvv` y 4 para `pin`.
- Inicialmente se previó que el campo `expiring_date` fuera de tipo `DATE`. Sin embargo, dado que los datos llegan en formato `MM/DD/YY` (por ejemplo, `10/30/22`), el cual no es compatible directamente con el tipo `DATE` en MySQL, se decidió declararlo inicialmente como `VARCHAR`. Luego de la ingesta de datos, se aplicará la transformación correspondiente para almacenarlos finalmente como `DATE`.
- Se definieron restricciones `UNIQUE` en `iban` y `pan` para evitar duplicados, ya que ambos son datos únicos en una tarjeta.
- Se definió la condición `NOT NULL` en todos los campos para garantizar que no haya valores vacíos.

- Creación de la tabla

```

1  -- Tarea S3.01
2  ----- Nivel 1 -----
3  -- Nivel 1. Ejercicio 1
4  -- Creación de Tabla credit_card
5  CREATE TABLE IF NOT EXISTS credit_card (
6      id VARCHAR(15) PRIMARY KEY, -- Identificador único de la tarjeta
7      -- nombre VARCHAR(100) NOT NULL, -- Nombre del titular de la tarjeta
8      iban VARCHAR(34) NOT NULL UNIQUE, -- Código IBAN único
9      pan VARCHAR(19) NOT NULL UNIQUE, -- Número PAN de la tarjeta
10     pin VARCHAR(4) NOT NULL, -- PIN de 4 dígitos
11     cvv VARCHAR(4) NOT NULL, -- CVV de 3 o 4 dígitos
12     expiring_date VARCHAR(10) NOT NULL -- Fecha de expiración
13 );
14
Output
Action Output
# Time Action Message
1 16:11:34 CREATE TABLE IF NOT EXISTS credit_card ( id VARCHAR(15) PRIMARY KEY. -- Identificador único de la tarjeta -- nombre V... 0 row(s) affected

```

- Visualización de la estructura de la tabla creada

```

13     );
14
15     -- Mostrar la estructura de la tabla credit_card
16     DESCRIBE credit_card;
17
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
Field Type Null Key Default Extra
id varchar(15) NO PRI NULL
iban varchar(34) NO UNI NULL
pan varchar(19) NO UNI NULL
pin varchar(4) NO NULL
cvv varchar(4) NO NULL
expiring_date varchar(10) NO NULL

Result 1
Output
Action Output
# Time Action Message
1 16:26:03 DESCRIBE credit_card 6 row(s) returned

```

- Insertar datos

- Ingesta de datos ejecutando código del archivo "datos_introducir_credit"

S301 Manipulación de tablas

datos_introducir_crediti

Limit to 1000 rows

259	0	259	0	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4730', 'EE541536644818872885', '373396765877539', '5988', '988', '03/08/25');	
260	0	260	0	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4737', 'AT278617748359359721', '3426 555216 37521', '9048', '509', '02/25/22');	
261	0	261	0	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4744', 'AZ5875188617480603476638322', '5256255735951122', '6209', '606', '10/12/21');	
262	0	262	0	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4751', 'PK8373046933330403264694', '3436 372241 88142', '6392', '760', '06/19/23');	
263	0	263	0	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4758', 'GB51GUVH61469185263634', '378486693428441', '5241', '571', '07/27/23');	
264	0	264	0	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4765', 'SA2888713798782221436615', '448 51353 39347 393', '1667', '153', '02/28/21');	
265	0	265	0	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4772', 'ME59832015454148127328', '455 63485 32288 611', '1201', '552', '09/21/20');	
266	0	266	0	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4779', 'FI1909231810971761', '513 92416 26288 645', '8788', '131', '08/18/23');	
267	0	267	0	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4786', 'SI51703104173167515', '557 97688 75435 755', '9002', '199', '05/15/20');	
268	0	268	0	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4793', 'HU95215627749276573565556322', '471662 767641 7624', '7216', '848', '11/09/23');	
269	0	269	0	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4800', 'SI97824334522161436', '5455 7952 5528 3322', '3745', '886', '05/23/20');	
270	0	270	0	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4807', 'LB19298318715580851625676971', '4539 4326 8269 4216', '8596', '626', '04/07/22');	
271	0	271	0	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4814', 'MR4845282437847152280636374', '374471619343357', '8790', '124', '12/19/20');	
272	0	272	0	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4821', 'LT253147505686466784', '453987 7873842836', '9000', '867', '07/15/20');	
273	0	273	0	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4828', 'BG11LMJ30149367569464', '4485252735942', '2789', '942', '09/04/24');	
274	0	274	0	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4835', 'PT34592171131762300132583', '3723 677744 22550', '1149', '680', '01/08/24');	
275	0	275	0	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4842', 'SA2156708581957118818229', '3774 636724 83250', '4655', '750', '11/11/24');	
276	0	276	0	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4849', 'SE281323487163628531121', '5223363813491514', '9992', '779', '03/21/25');	
277	0	277	0	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4856', 'TR373872558313545667124286', '3495528235713651', '9086', '974', '05/19/23');	
278	0	278	0		

Output

Action Output

#	Time	Action	Message
245	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4639', 'L8155319586365581720', '373693...	1 row(s) affected
246	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4646', 'CR72377351351001258248', '4485 8...	1 row(s) affected
247	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4653', 'DE45027134377528457318', '4716 ...	1 row(s) affected
248	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4660', 'LV89QR3CT474307668821', '49294...	1 row(s) affected
249	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4667', 'PT79551618452454886142166', '37...	1 row(s) affected
250	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4674', 'DO7186427375257585510355726...	1 row(s) affected
251	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4681', 'GE38261236401561281317', '448 5...	1 row(s) affected
252	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4688', 'IS3832557567461896771082', '3...	1 row(s) affected
253	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4695', 'GT44327784356731801238887639...	1 row(s) affected
254	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4702', 'DK7772824378841077', '34864267...	1 row(s) affected
255	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4709', 'GE45504127369320160124', '4532...	1 row(s) affected
256	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4716', 'PL75422719203842155307261926...	1 row(s) affected
257	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4723', 'FI8248793454336573', '3414 37643...	1 row(s) affected
258	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4730', 'EE541536644818872885', '3733967...	1 row(s) affected
259	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4737', 'AT278617748359359721', '3426 555...	1 row(s) affected
260	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4744', 'AZ5875188617480603476638322...	1 row(s) affected
261	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4751', 'PK8373046933330403264694', '343...	1 row(s) affected
262	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4758', 'GB51GUVH61469185263634', '378...	1 row(s) affected
263	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4765', 'SA2888713798782221436615', '448...	1 row(s) affected
264	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4772', 'ME59832015454148127328', '455 6...	1 row(s) affected
265	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4779', 'FI1909231810971761', '513 92416 ...	1 row(s) affected
266	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4786', 'SI51703104173167515', '557 97688...	1 row(s) affected
267	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4793', 'HU95215627749276573565556322...	1 row(s) affected
268	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4800', 'SI97824334522161436', '5455 7952...	1 row(s) affected
269	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4807', 'LB19298318715580851625676971'...	1 row(s) affected
270	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4814', 'MR4845282437847152280636374'...	1 row(s) affected
271	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4821', 'LT253147505686466784', '453987...	1 row(s) affected
272	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4828', 'BG11LMJ30149367569464', '44852...	1 row(s) affected
273	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4835', 'PT34592171131762300132583', '37...	1 row(s) affected
274	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4842', 'SA2156708581957118818229', '377...	1 row(s) affected
275	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4849', 'SE281323487163628531121', '522...	1 row(s) affected
276	16:14:54	INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES ('CcU-4856', 'TR373872558313545667124286', '3...	1 row(s) affected

Dado que la columna `expiring_date` contenía fechas en un formato no compatible directamente con el tipo `DATE` de MySQL y se insertaron como `VARCHAR`, se decidió realizar ajustes posteriores a la ingesta de datos para asegurar que las fechas quedaran almacenadas en el tipo de dato adecuado, garantizando así una correcta gestión en el futuro.

- Actualizar las fechas al formato correcto (YYYY-MM-DD)

Por defecto, MySQL Workbench tiene activado el **modo seguro** para evitar modificaciones masivas sin condiciones basadas en claves primarias. Para actualizar los datos fue necesario:

- Desactivarlos temporalmente.
- Actualizar la columna `expiring_date` con la modificación del formato de la fecha.
- Volver a activar el modo seguro

The screenshot shows a MySQL Workbench window titled 'S301 Manipulación de tablas'. The SQL editor contains the following commands:

```

19 -- Modificar el formato de la fecha
20 SET SQL_SAFE_UPDATES = 0; -- Desactivando el modo seguro que impide actualizar datos sin una condición basada en una clave primaria.
21
22 UPDATE credit_card
23 SET expiring_date = STR_TO_DATE(expiring_date, '%m/%d/%y');
24
25 SET SQL_SAFE_UPDATES = 1; -- Activando nuevamente el modo seguro.
26

```

The Output window shows the execution results:

#	Time	Action	Message
1	16:53:00	SET SQL_SAFE_UPDATES = 0	0 row(s) affected
2	16:53:00	UPDATE credit_card SET expiring_date = STR_TO_DATE(expiring_date, '%m/%d/%y')	275 row(s) affected Rows matched: 275 Changed: 275 Warnings: 0
3	16:53:00	SET SQL_SAFE_UPDATES = 1	0 row(s) affected

- Actualizar el tipo de datos de `expiring_date` a `DATE`

The screenshot shows a MySQL Workbench window titled 'S301 Manipulación de tablas'. The SQL editor contains the following command:

```

25
26 Save the script to a file. tipo de dato a Date
27 ALTER TABLE credit_card
28 MODIFY COLUMN expiring_date DATE;
29

```

The Output window shows the execution results:

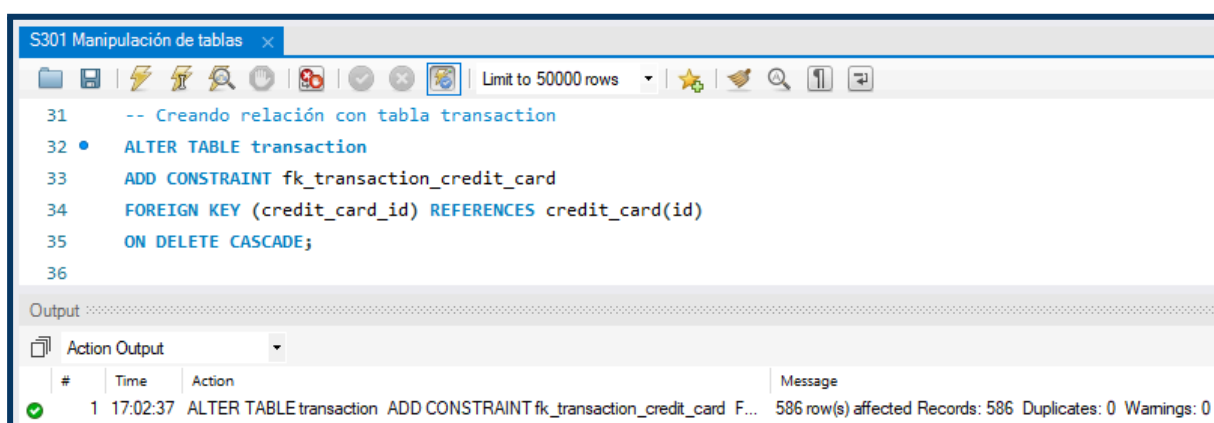
#	Time	Action	Message
1	16:57:19	ALTER TABLE credit_card MODIFY COLUMN expiring_date DATE	275 row(s) affected Records: 275 Duplicates: 0 Warnings: 0

- Creación de las relaciones

La tabla `credit_card` debe tener una relación de uno a muchos con la tabla `transaction`, ya que una tarjeta puede estar asociada a múltiples transacciones. Para lograrlo, se debe establecer el campo `credit_card_id` en la tabla `transaction` como clave foránea, vinculándolo con el campo `id` de la tabla `credit_card`.

Esta relación no se configuró inmediatamente después de crear la tabla `credit_card`, debido a que la tabla `transaction` ya contenía datos y es necesario que los identificadores de las tarjetas en las transacciones existentes correspondan a tarjetas registradas. Si se hubiera intentado establecer una relación con registros en `transaction` sin una tarjeta asociada válida en `credit_card`, se produciría un error para evitar la creación de registros "huérfanos".

Se incluye la opción `ON DELETE CASCADE` para mantener la integridad referencial. Esto elimina automáticamente las transacciones asociadas a una tarjeta eliminada, lo cual, aunque útil para evitar errores de inconsistencia y prevenir la persistencia de datos huérfanos, debe manejarse con precaución.



The screenshot shows a SQL editor window titled "S301 Manipulación de tablas". The SQL code being executed is as follows:

```

31  -- Creando relación con tabla transaction
32  • ALTER TABLE transaction
33  ADD CONSTRAINT fk_transaction_credit_card
34  FOREIGN KEY (credit_card_id) REFERENCES credit_card(id)
35  ON DELETE CASCADE;
36

```

Below the code editor is an "Output" section showing the execution results:

#	Time	Action	Message
1	17:02:37	ALTER TABLE transaction ADD CONSTRAINT fk_transaction_credit_card F...	586 row(s) affected Records: 586 Duplicates: 0 Warnings: 0

- Diagrama ER

🔗 Descripción

El diagrama representa una base de datos donde se registran transacciones financieras, vinculadas a tarjetas de crédito y compañías.

★ Tabla `credit_card`

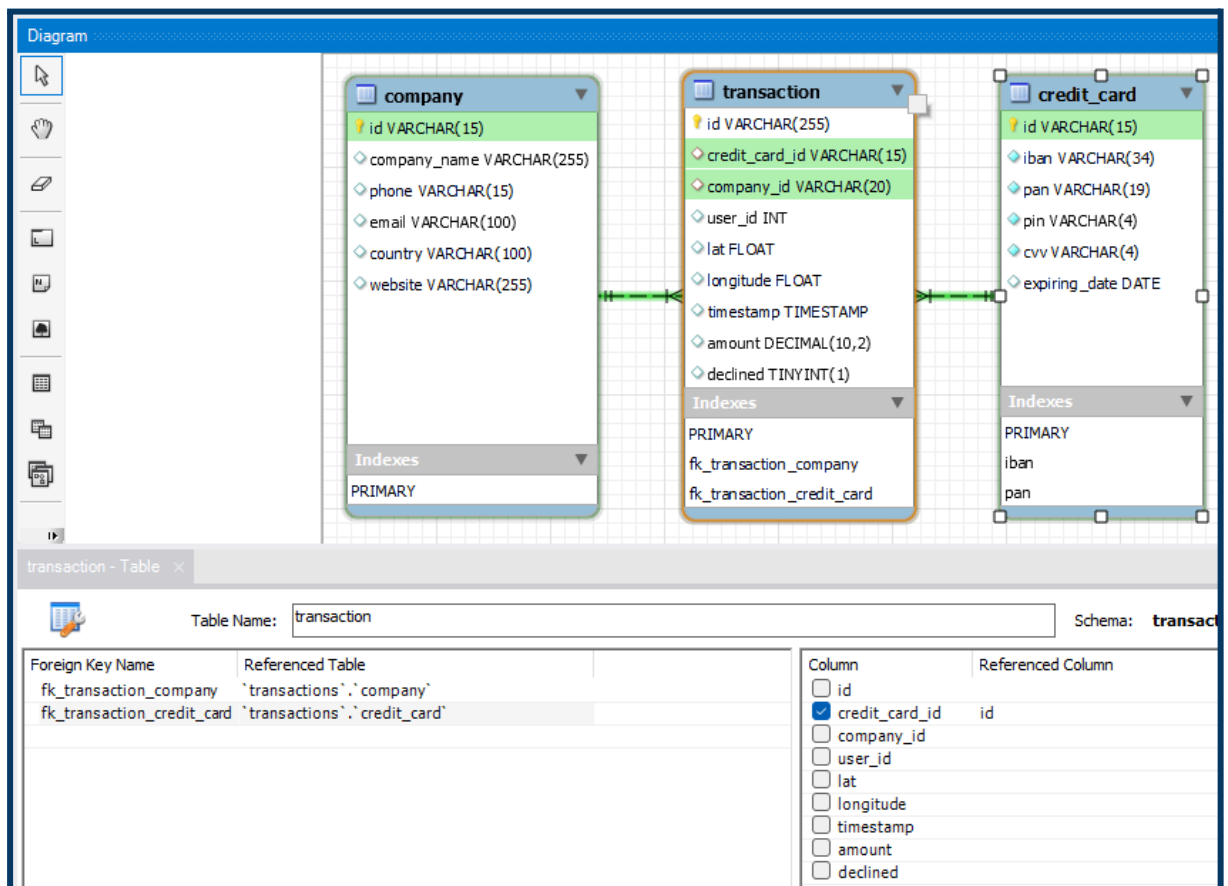
- Recoge datos relevantes de las tarjetas de crédito
- Clave **primaria**: `id`
- Una tarjeta puede estar vinculada a múltiples transacciones (relación 1:N)

★ Tabla **company**

- Recoge datos relevantes de las compañías
- Clave **primaria**: **id**
- Una compañía puede estar vinculada a múltiples transacciones (relación 1:N)

★ Tabla **transaction**

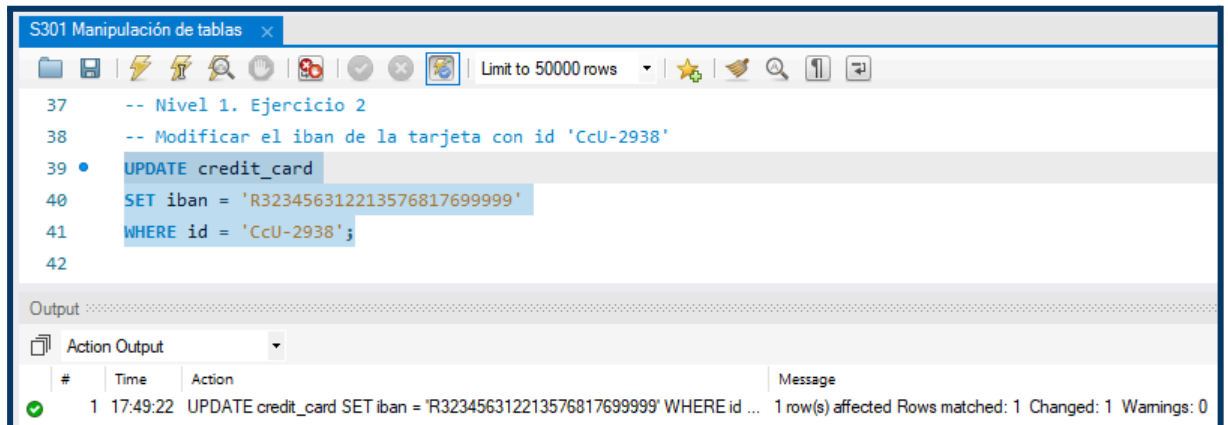
- Recoge los datos de las transacciones realizadas.
- Clave **primaria**: **id**
- Claves **foráneas**:
 - ◆ **card_id**: Hace referencia a la tarjeta de crédito utilizada en la transacción, vinculada a la clave primaria de **credit_card** (**id**)
 - ◆ **company_id**: Hace referencia a la compañía que realiza la transacción, vinculada a la clave primaria de **company** (**id**)
- Cada transacción puede estar vinculada a una única tarjeta y a una única compañía (relación 1:N).



1.2. Ejercicio 2

El departamento de Recursos Humanos ha identificado un error en el número de cuenta del usuario con ID CcU-2938. La información que debe mostrarse para este registro es: R323456312213576817699999. Recuerda mostrar que el cambio se realizó.

- Modificar de datos



The screenshot shows a database management tool window titled "S301 Manipulación de tablas". The SQL editor contains the following code:

```

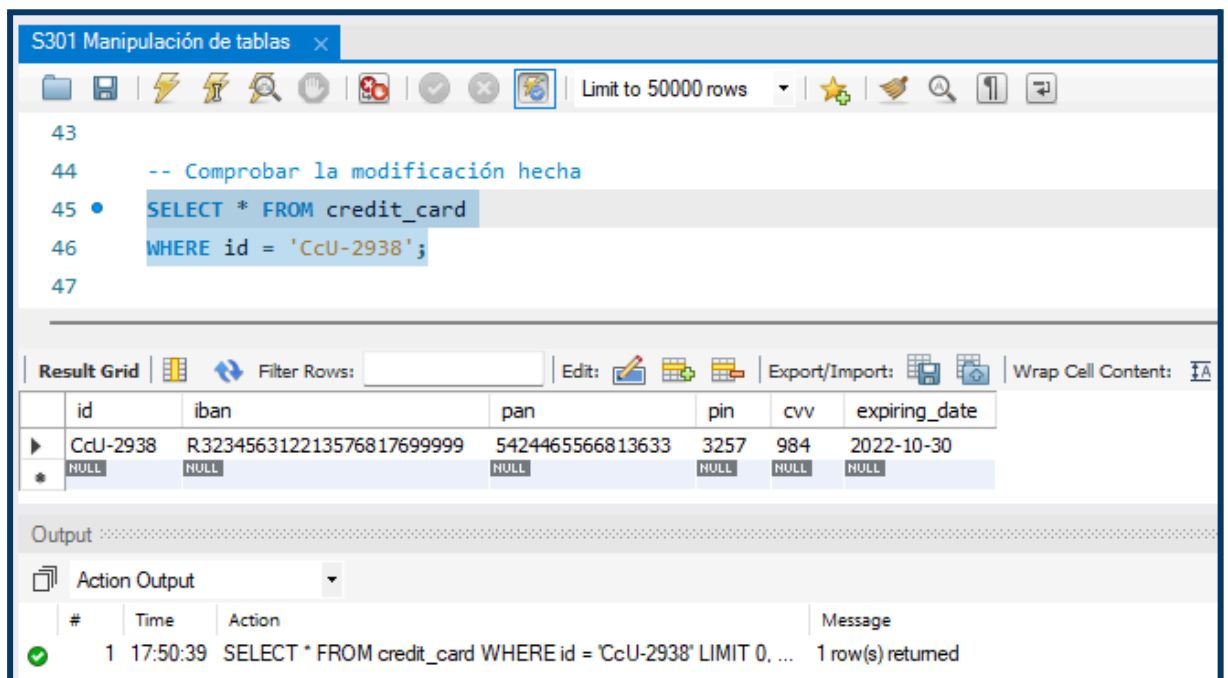
37  -- Nivel 1. Ejercicio 2
38  -- Modificar el iban de la tarjeta con id 'CcU-2938'
39  • UPDATE credit_card
40  SET iban = 'R323456312213576817699999'
41  WHERE id = 'CcU-2938';
42

```

The Output pane shows the execution results:

#	Time	Action	Message
1	17:49:22	UPDATE credit_card SET iban = 'R323456312213576817699999' WHERE id = 'CcU-2938';	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0

- Mostrar cambio realizado



The screenshot shows the same database management tool window. The SQL editor contains the following code:

```

43
44  -- Comprobar la modificación hecha
45  • SELECT * FROM credit_card
46  WHERE id = 'CcU-2938';
47

```

The Output pane shows the execution results in a Result Grid:

id	iban	pan	pin	cvv	expiring_date
CcU-2938	R323456312213576817699999	5424465566813633	3257	984	2022-10-30

The Output pane also shows the execution details:

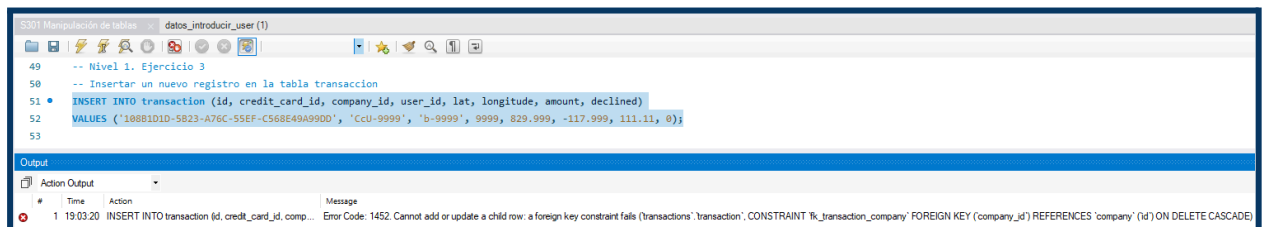
#	Time	Action	Message
1	17:50:39	SELECT * FROM credit_card WHERE id = 'CcU-2938' LIMIT 0, ...	1 row(s) returned

1.3. Ejercicio 3

En la tabla "transaction" ingresa un nuevo usuario con la siguiente información:

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

- Insertar nuevo registro en transaction



```

49 -- Nivel 1. Ejercicio 3
50 -- Insertar un nuevo registro en la tabla transaction
51 INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined)
52 VALUES ('108B1D1D-5B23-A76C-55EF-C568E49A99DD', 'CcU-9999', 'b-9999', 9999, 829.999, -117.999, 111.11, 0);
53
Output
Action Output
# Time Action Message
1 19:03:20 INSERT INTO transaction (id, credit_card_id, comp... Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails ('transactions`.`transaction`, CONSTRAINT `fk_transaction_company` FOREIGN KEY (`company_id`) REFERENCES `company` (`id`) ON DELETE CASCADE)

```

Al intentar insertar el registro en la tabla *transaction*, se produce el siguiente error :

Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails ('transactions`.`transaction`, CONSTRAINT `fk_transaction_company` FOREIGN KEY (`company_id`) REFERENCES `company` (`id`) ON DELETE CASCADE)

Este error ocurre porque el id 'b-9999' (columna *company_id*) no existe en la tabla *company* (*id*), lo cual no es compatible con la relación establecida entre ambas tablas. Dado que *company_id* es una clave foránea, el valor de esta columna en cada fila debe corresponder a un registro existente en la tabla *company*. Como la compañía con id 'b-9999' no existe en la base de datos, no se puede completar la inserción.

Con el fin exclusivo de completar este ejercicio docente, se procederá a crear la compañía en la tabla *company*, pero es importante aclarar que no sería una práctica común en un entorno productivo.

Antes de realizar la inserción con datos genéricos autodefinidos, es recomendable revisar los tipos de datos definidos en la entidad para asegurarse de que los valores sean compatibles y evitar posibles errores durante la inserción.

S301 Manipulación de tablas

VALUES ('108B1D1D-5B23-A76C-55EF-C568E49A99DD', 'CcU-999')

-- Consultar especificaciones de tabla company

DESCRIBE company;

Result Grid

Field	Type	Null	Key	Default	Extra
id	varchar(15)	NO	PRI	NULL	
company_name	varchar(255)	YES		NULL	
phone	varchar(15)	YES		NULL	
email	varchar(100)	YES		NULL	
country	varchar(100)	YES		NULL	

Result 2

Output

Action Output

#	Time	Action	Message
1	21:15:22	DESCRIBE company	5 row(s) returned

Además, es importante analizar los datos existentes en la tabla para insertar valores que respeten el formato y la estructura previamente establecidos. Esto garantiza la coherencia de la información y previene futuros conflictos en validaciones o integraciones posteriores.

S301 Manipulación de tablas

-- Consultar especificaciones de tabla company

DESCRIBE company;

-- Mostrar compañías

select * from company;

Result Grid

id	company_name	phone	email	country
b-2222	Ac Fermentum Incorporated	06 85 56 52 33	donec.porttitor.tellus@yahoo.net	Germany
b-2226	Magna A Neque Industries	04 14 44 64 62	risus.donec.nibh@icloud.org	Australia
b-2230	Fusce Corp.	08 14 97 58 85	risus@protonmail.edu	United States
b-2234	Convallis In Incorporated	06 66 57 29 50	mauris.ut@aol.couk	Germany
b-2238	Ante Iaculis Nec Foundation	08 23 04 99 53	sed.dictum.proin@outlook.ca	New Zealand
b-2242	Donec Ltd	01 25 51 37 37	at.iaculis@hotmail.couk	Norway

company 4

Output

Action Output

#	Time	Action	Message
1	21:32:49	select * from company LIMIT 0, 50000	100 row(s) returned

Al analizar la estructura de la tabla y sus registros, se determinó lo siguiente:

- **id**: El dato que se pretende insertar coincide con el tipo de datos definido (**VARCHAR**)
- **email**: Se optó por usar un formato estándar como 'company@aux.com' para asegurar que, aunque el valor no sea real, cumpla con la estructura de un correo electrónico. Esto previene inconsistencias en posibles validaciones de datos futuras y a su vez, garantiza que sea compatible con el tipo de dato **VARCHAR**
- **phone**: Se optó por utilizar el valor '99 99 99 99 99' en correspondencia con el formato de los valores específicos de esta columna. Además es compatible con el tipo de dato **VARCHAR**.
- Restantes campos: Se optó por insertar como valor 'aux', dado que todos son de tipo **VARCHAR** y no tienen un formato específico que interfiera con la estructura de la tabla. Este valor es comúnmente utilizado en ejercicios de pruebas para representar un valor genérico.

- Crear compañía “b-9999”

The screenshot shows a SQL IDE window titled "S301 Manipulación de tablas". The SQL editor contains the following code:

```

57  -- Insertar compañía con ID b-9999
58  •  INSERT INTO company (id, company_name, phone, email, country)
59      VALUES ('b-9999', 'aux', '99 99 99 99 99', 'company@aux.com', 'aux');
60

```

Below the editor, the "Output" pane shows the "Action Output" for the executed statement:

#	Time	Action	Message
1	22:10:20	INSERT INTO company (id, company_na...	1 row(s) affected

Una vez insertada la compañía, como **transaction** también está relacionada con la tabla **credit_card**, es prudente asegurarse de que exista una tarjeta de crédito con el **id** "CcU-999"; de no ser así habrá que repetir el procedimiento y crear dicha tarjeta de crédito.

S301 Manipulación de tablas

```

72  -- Comprobar si existe tarjeta de credito con id CcU-9999
73  • Select * FROM credit_card
74  WHERE id = 'CcU-9999';
75
76

```

Result Grid

	id	iban	pan	pin	cvv	expiring_date
*	NULL	NULL	NULL	NULL	NULL	NULL

Output

Action Output

#	Time	Action	Message
✓ 1	22:25:24	Select * FROM credit_card WHERE id = 'C...	0 row(s) returned

La tarjeta de crédito no existe y repetimos el procedimiento descrito anteriormente para crearla.

Se revisan los tipos de datos definidos en la entidad `credit_card`

S301 Manipulación de tablas

```

75  -- Consultar especificaciones de tabla credit_card
76  • DESCRIBE credit_card;
77
78  -- Mostrar tarjetas de crédito
79  • SELECT * FROM credit_card;

```

Result Grid

	Field	Type	Null	Key	Default	Extra
▶	id	varchar(15)	NO	PRI	NULL	
	iban	varchar(34)	NO	UNI	NULL	
	pan	varchar(19)	NO	UNI	NULL	
	pin	varchar(4)	NO		NULL	
	cvv	varchar(4)	NO		NULL	
	expiring_date	date	YES		NULL	

Result 21

Output

Action Output

#	Time	Action	Message
✓ 1	22:33:58	DESCRIBE credit_card	6 row(s) returned

Se analizan los datos de las tarjetas existentes

The screenshot shows a database management interface with the following components:

- SQL Editor:** Contains the following queries:


```

75  -- Consultar especificaciones de tabla credit_card
76  • DESCRIBE credit_card;
77
78  -- Mostrar tarjetas de crédito
79  • SELECT * FROM credit_card;
      
```
- Result Grid:** Displays the structure and data of the `credit_card` table.

	id	iban	pan	pin	cvv	expiring_date
▶	CcU-2938	R323456312213576817699999	5424465566813633	3257	984	2022-10-30
	CcU-2945	DO26854763748537475216568689	5142423821948828	9080	887	2023-08-24
	CcU-2952	BG45IVQL52710525608255	4556 453 55 5287	4598	438	2021-06-29
	CcU-2959	CR7242477244335841535	372461377349375	3583	667	2023-02-24
	CcU-2966	BG72LKTQ15627628377363	448566 886747 7265	4900	130	2024-10-29
	CcU-2973	PT87806228135092429456346	544 58654 54343 384	8760	887	2025-01-30
	CcU-2980	DE39241881883086277136	402400 7145845969	5075	596	2022-07-24
- Output:** Shows the execution of the `SELECT * FROM credit_card` query, indicating that 275 rows were returned.

Al analizar la estructura de la tabla y sus registros, se determinó lo siguiente:

- **id:** El dato que se pretende insertar coincide con el tipo de datos definido (**VARCHAR**)
- **iban:** Se optó por utilizar el valor '**AUX999999999**', ya que respeta el formato esperado para esta columna y es compatible con el tipo de dato **VARCHAR**.
- **pan:** Se optó por utilizar el valor '**999999999999999**', en correspondencia con el formato de los valores existentes. Además, es compatible con el tipo de dato **VARCHAR**.
- **pin:** Se optó por utilizar el valor '**9999**', siguiendo el formato de los registros actuales. También es compatible con el tipo de dato **VARCHAR**.
- **cvv:** Se optó por utilizar el valor '**999**', en línea con el formato de los datos específicos de esta columna. Asimismo, es compatible con el tipo de dato **VARCHAR**.
- **expiring_date:** Se optó por utilizar la fecha '**1900-01-01**', ya que es el formato encontrado en los valores de esta columna, lo que evita futuros conflictos en el

manejo de los mismos. Además, se eligió esta fecha porque es convencionalmente utilizada en estos casos, garantizando que no afecte las estadísticas en análisis futuros.

- Crear tarjeta “CcU-9999”

The screenshot shows a SQL IDE window titled "S301 Manipulación de tablas". The SQL editor contains the following code:

```

81  -- Insertar tarjeta con ID CcU-9999
82  • INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date)
83  VALUES ('CcU-9999', 'AUX999999999', '9999999999999999', '9999', '999', '1900-01-01');

```

The "Output" pane shows the "Action Output" for the executed statement:

#	Time	Action	Message
1	22:50:46	INSERT INTO credit_card (id, iban, pan, pi...	1 row(s) affected

- Insertar transacción

The screenshot shows a SQL IDE window titled "S301 Manipulación de tablas". The SQL editor contains the following code:

```

45
46  -- Nivel 1. Ejercicio 3
47  -- Insertar un nuevo registro en la tabla transaccion
48  • INSERT INTO transaction (id, credit_card_id, company_id, user_id, lat, longitude, amount, declined)
49  VALUES ('10881D1D-5B23-A76C-55EF-C568E49A99DD', 'CcU-9999', 'b-9999', 9999, 829.999, -117.999, 111.11, 0);
50

```

The "Output" pane shows the "Action Output" for the executed statement:

#	Time	Action	Message
1	22:52:39	INSERT INTO transaction (id, credit_card_...	1 row(s) affected

- Comprobar los valores insertados

The screenshot shows a SQL IDE window titled "S301 Manipulación de tablas". The SQL editor contains the following code:

```

74
75  -- Compruebo los datos insertados
76  • SELECT
77  t.id AS tran_id, t.credit_card_id AS tran_credit_card_id, t.company_id AS tran_company_id, t.user_id AS tran_user_id, t.lat AS tran_lat, t.longitude AS tran_longitude, t.amount AS tran_amount, t.declined AS tran_declined,
78  c.id AS comp_id, c.company_name AS comp_name, c.phone AS comp_phone, c.email AS comp_email, c.country AS comp_country,
79  cc.id AS card_id, cc.iban AS card_iban, cc.pan AS card_pan, cc.pin AS card_pin, cc.cvv AS card_cvv, cc.expiring_date AS card_expiring_date
80  FROM transaction t
81  JOIN company c ON t.company_id = c.id
82  JOIN credit_card cc ON t.credit_card_id = cc.id
83  WHERE t.id = '10881D1D-5B23-A76C-55EF-C568E49A99DD';
84

```

The "Result Grid" shows the following data:

tran_id	tran_credit_card_id	tran_company_id	tran_user_id	tran_lat	tran_longitude	tran_amount	tran_declined	comp_id	comp_name	comp_phone	comp_email	comp_country	card_id	card_iban	card_pan	card_pin	card_cvv	card_expiring_date
10881D1D-5B23-A76C-55EF-C568E49A99DD	CcU-9999	b-9999	9999	829.999	-117.999	111.11	0	b-9999	aux	99 99 99 99 99	company@aux.com	aux	CcU-9999	AUX9999999999	9999999999999999	9999	999	1900-01-01

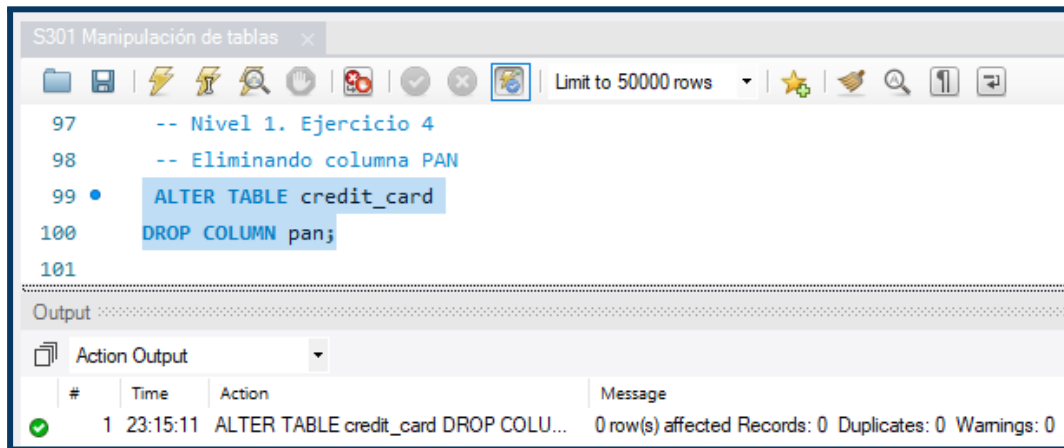
The "Output" pane shows the "Action Output" for the executed statement:

#	Time	Action	Message
1	00:17:53	SELECT t.id AS tran_id, t.credit_card_id...	1 row(s) returned

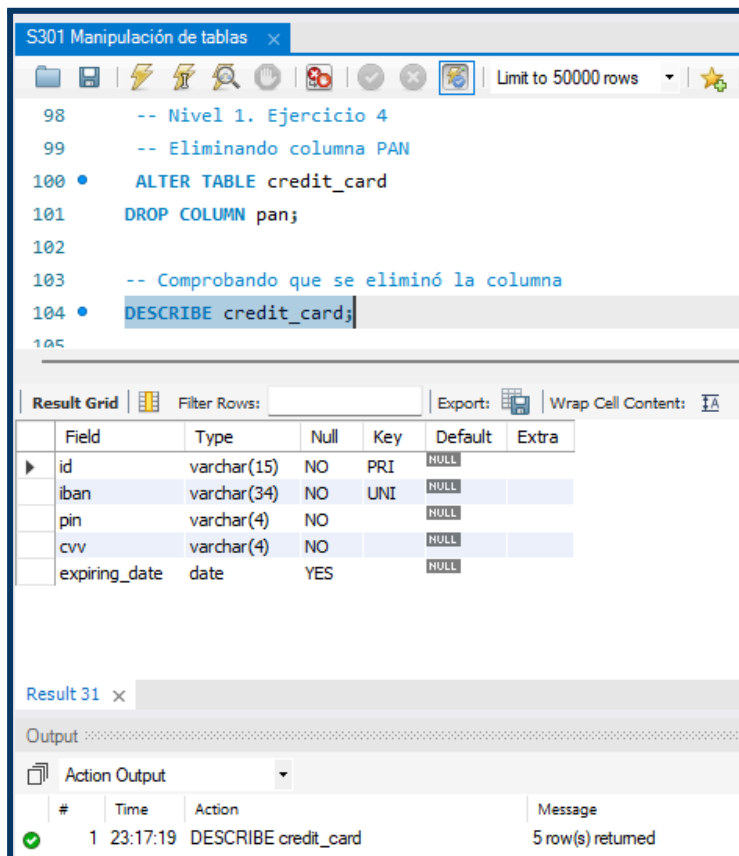
1.4. Ejercicio 4

Desde recursos humanos te solicitan eliminar la columna "pan" de la tabla credit_card. Recuerda mostrar el cambio realizado.

- Eliminar columna



- Comprobar cambio realizado

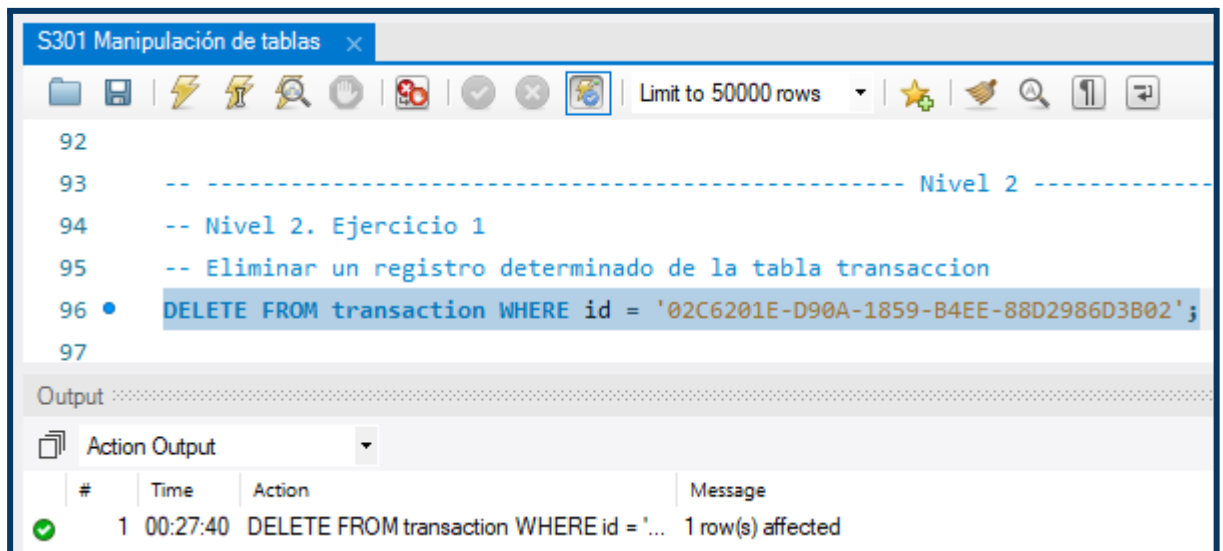


2. Nivel 2

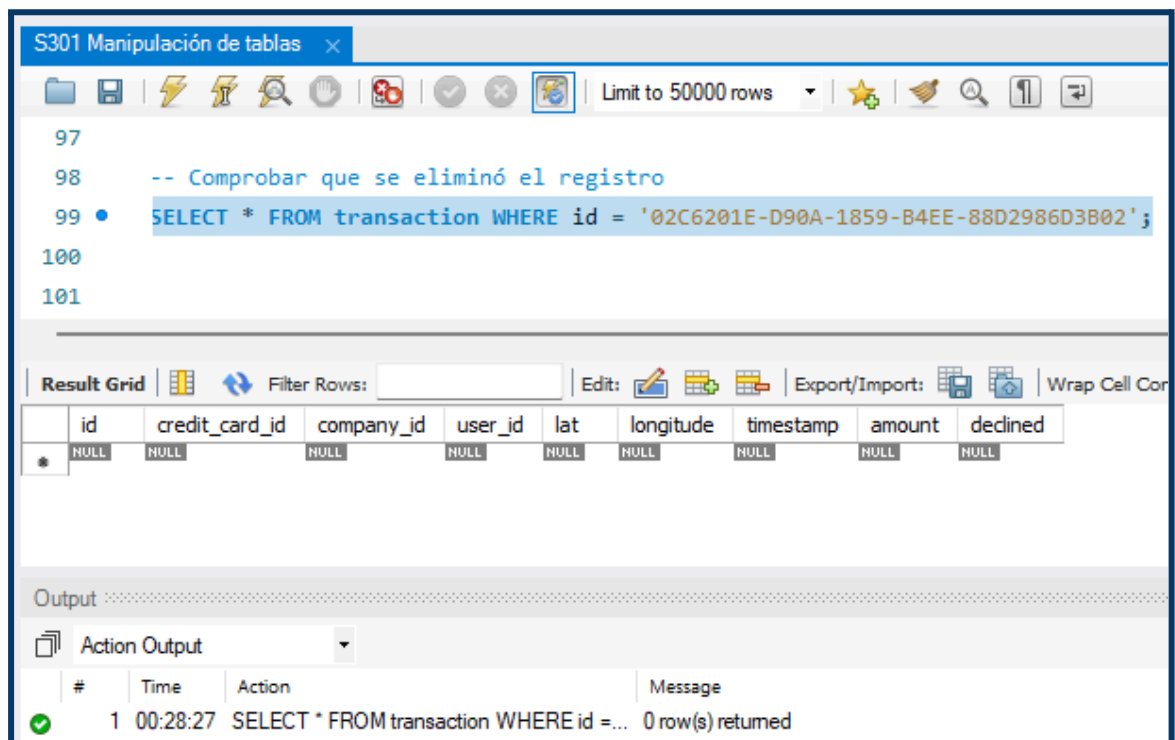
2.1. Ejercicio 1

Elimina de la tabla transacción el registro con ID
02C6201E-D90A-1859-B4EE-88D2986D3B02 de la base de datos.

- Eliminar registro



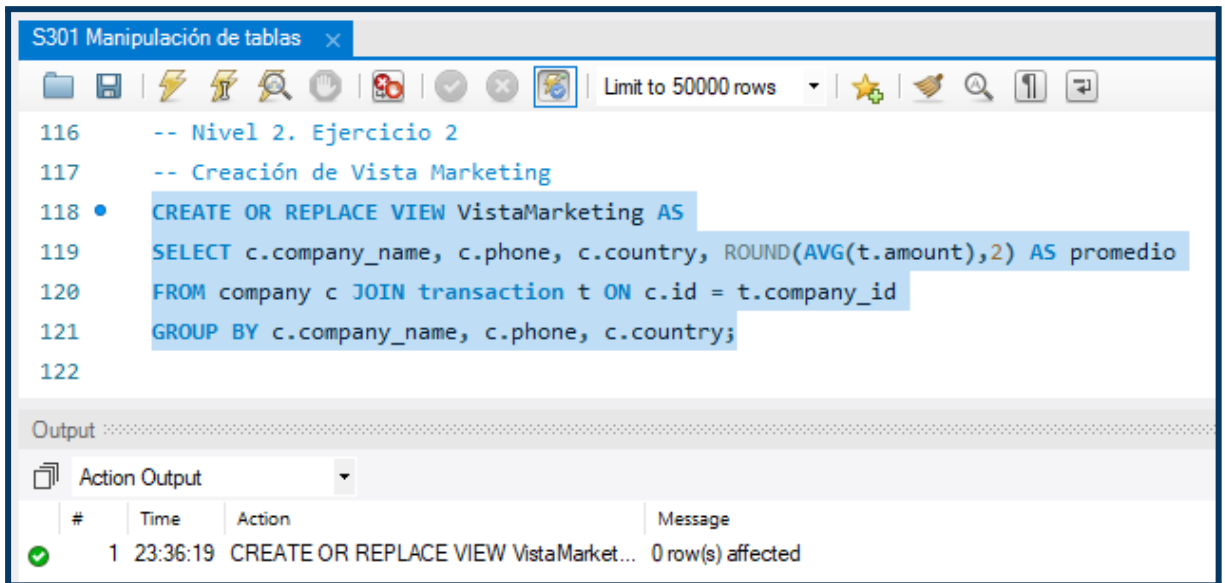
Comprobar cambio realizado



2.2. 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 promedio de compra.

- Crear la vista



```
116 -- Nivel 2. Ejercicio 2
117 -- Creación de Vista Marketing
118 • CREATE OR REPLACE VIEW VistaMarketing AS
119 SELECT c.company_name, c.phone, c.country, ROUND(AVG(t.amount),2) AS promedio
120 FROM company c JOIN transaction t ON c.id = t.company_id
121 GROUP BY c.company_name, c.phone, c.country;
122
```

Output

#	Time	Action	Message
1	23:36:19	CREATE OR REPLACE VIEW VistaMarket...	0 row(s) affected

- Mostrar resultados de la vista

El ordenamiento se realiza al consultar la vista y no al crearla porque las vistas son representaciones dinámicas de consultas, no datos estáticos. Ordenar al consultar ofrece flexibilidad, permitiendo ajustar el orden según sea necesario sin modificar la vista. Además, independientemente del orden que se defina al crearla, siempre se puede reordenar al consultar, lo cual evitaría añadir código innecesario.

The screenshot shows a database IDE window titled "S301 Manipulación de tablas". The SQL editor contains the following code:

```

117 -- Creación de Vista Marketing
118 • CREATE OR REPLACE VIEW VistaMarketing AS
119 SELECT c.company_name, c.phone, c.country, ROUND(AVG(t.amount),2) AS promedio
120 FROM company c JOIN transaction t ON c.id = t.company_id
121 GROUP BY c.company_name, c.phone, c.country;
122
123 -- Mostrar la vista
124 • SELECT * FROM VistaMarketing
125 ORDER BY promedio DESC;

```

Below the editor, the "Result Grid" shows the query results. The table has four columns: company_name, phone, country, and promedio. The results are ordered by promedio in descending order.

	company_name	phone	country	promedio
▶	Eget Ipsum Ltd	03 67 44 56 72	United States	473.08
	Non Magna LLC	06 71 73 13 17	United Kingdom	468.35
	Sed Id Limited	07 28 18 18 13	United States	461.21
	Justo Eu Arcu Ltd	08 42 56 71 52	Italy	443.64
	Eget Tincidunt Dui Institute	05 35 93 32 44	Netherlands	442.52
	Viverra Donec Foundation	03 33 12 32 73	United Kingdom	442.28
	Vestibulum Lorem PC	02 02 87 33 40	Belgium	434.06
	Aliquet Diam Limited	02 76 61 47 46	United States	425.64
	Maecenas Malesuada Fringilla Inc.	09 38 53 76 61	Netherlands	408.62

Below the result grid, the "Output" section shows the execution of the query:

#	Time	Action	Message
1	23:37:19	SELECT * FROM VistaMarketing ORDER ...	101 row(s) returned

2.3. Ejercicio 3

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

S301 Manipulación de tablas

Limit to 50000 rows

```

126
127 -- Nivel 2. Ejercicio 3
128 -- Filtrar Vista Marketing
129 • SELECT *
130 FROM VistaMarketing
131 WHERE country = 'Germany'
132 ORDER BY promedio DESC;
133

```

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

	company_name	phone	country	promedio
▶	Aliquam PC	01 45 73 52 16	Germany	385.27
	Ac Industries	09 34 65 40 60	Germany	289.65
	Rutrum Non Inc.	02 66 31 61 09	Germany	266.90
	Nunc Interdum Incorporated	05 18 15 48 13	Germany	244.03
	Augue Foundation	06 88 43 15 63	Germany	240.80
	Ac Fermentum Incorporated	06 85 56 52 33	Germany	206.47
	Auctor Mauris Corp.	05 62 87 14 41	Germany	184.31
	Convallis In Incorporated	06 66 57 29 50	Germany	156.73

VistaMarketing 36

Output

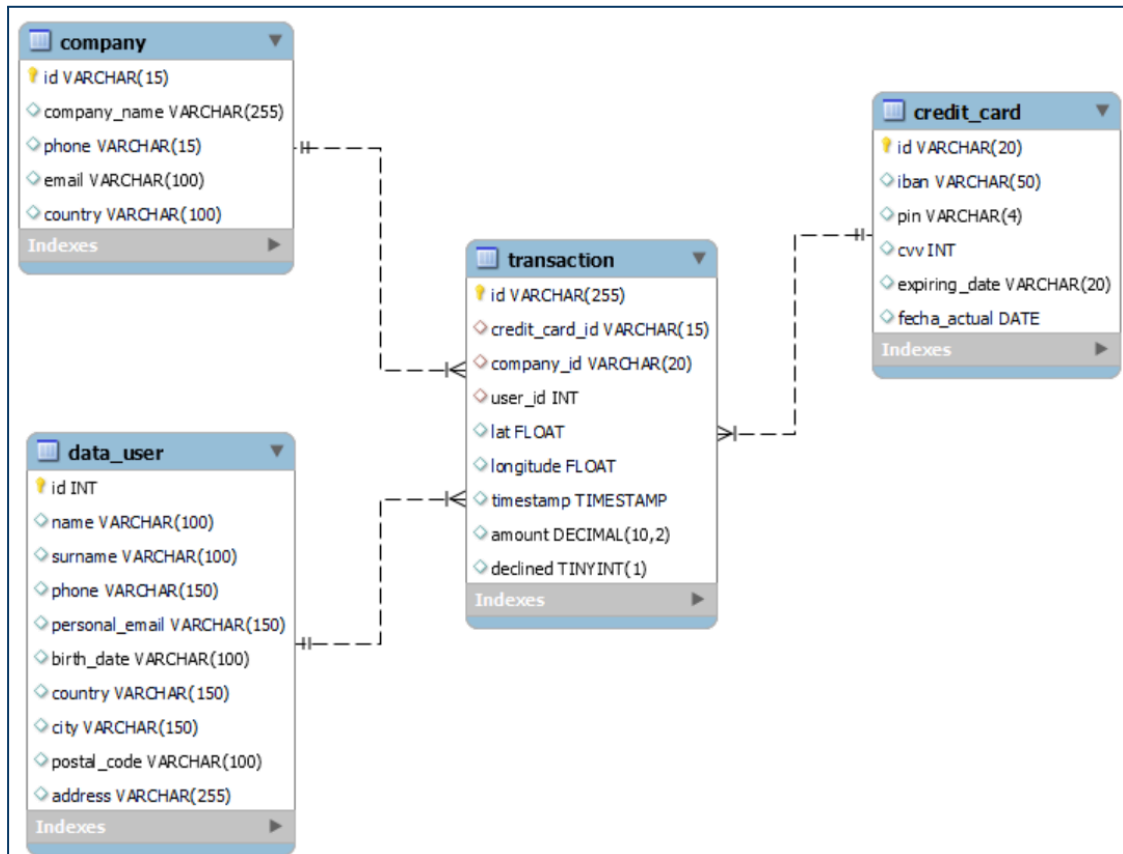
Action Output

#	Time	Action	Message
✓ 1	23:38:38	SELECT * FROM VistaMarketing WHERE...	8 row(s) returned

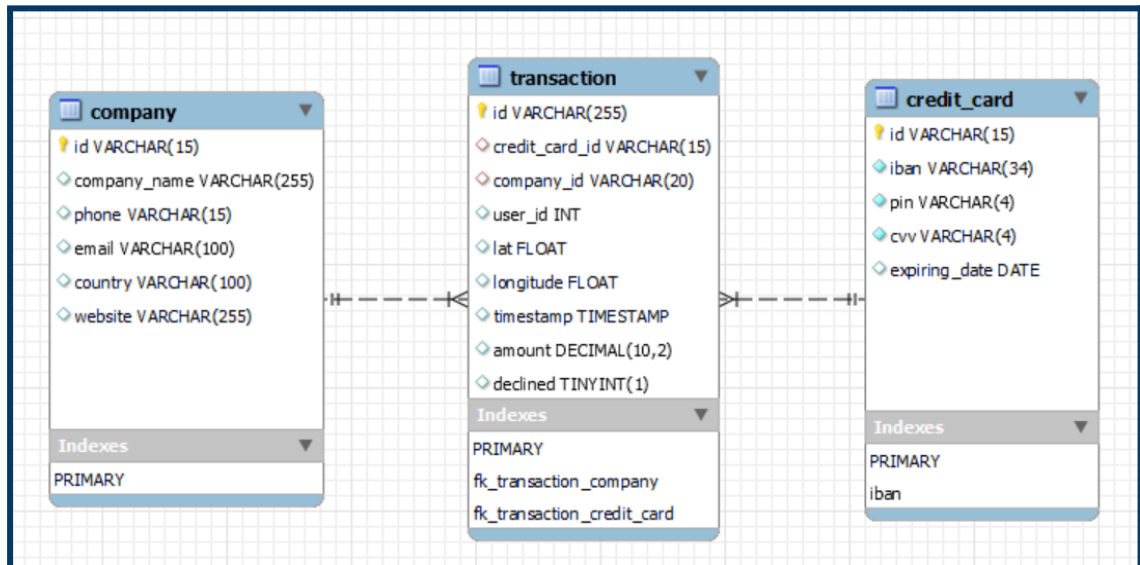
3. Nivel 3

3.1. 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 cómo las realizó. Te pide que le ayudes a dejar los comandos ejecutados para obtener el siguiente diagrama:



- Comparación con modelo anterior y definición de modificaciones necesarias por tablas



★ Tabla `data_user`:

→ Crear tabla con los siguientes atributos

- ◆ `id INT PRIMARY KEY`
- ◆ `name VARCHAR(100)`
- ◆ `surname VARCHAR(100)`
- ◆ `phone VARCHAR(150)`
- ◆ `personal_email VARCHAR(150)`
- ◆ `birth_date VARCHAR(100)`
- ◆ `country VARCHAR(150)`
- ◆ `city VARCHAR(150)`
- ◆ `postal_code VARCHAR(100)`
- ◆ `address VARCHAR(255)`

→ Ingestar data

★ Tabla `transaction`

→ Convertir `user_id` en **clave foránea**, estableciendo la relación con la tabla `user`.

★ Tabla `company`

→ Eliminar columna `website`

★ Tabla `credit_card`

→ Modificar la longitud de la columna `id`, de `VARCHAR(15)` a `VARCHAR(20)`

- Modificar la longitud de la columna `iban`, de `VARCHAR(34)` a `VARCHAR(50)`
 - Modificar la longitud de la columna `pin`, de `VARCHAR(10)` a `VARCHAR(4)`
 - Modificar el tipo de datos de la columna `cvv`, de `VARCHAR(4)` a `INT`
 - Modificar el tipo de datos de columna `expiring_date`, de `DATE` a `VARCHAR (20)`
 - Modificar de campo obligatorio a campo opcional los atributos `iban`, `pin`, `cvv`
 - Agregar columna `fecha_actual` (`DATE`)
- Creación de tabla `data_user`, ingesta de datos y relación con tabla `transaction`

En el archivo proporcionado para la creación de la tabla `data_user` se encontraron los siguientes hallazgos relevantes:

- Se define como nombre de la tabla `user` y debe ser `data_user`
- Se crea el campo `email` y debe ser `personal_email`
- Se crea un índice en la columna `user_id` de la tabla `transaction`

```
CREATE INDEX idx_user_id ON transaction(user_id);
```

- Se crea la tabla `user` con todos los atributos pertinentes y además una **clave foránea** vinculando el campo `id` de `user` con la columna `user_id` de la tabla `transaction`.

```
CREATE TABLE IF NOT EXISTS user (
  id INT PRIMARY KEY,
  ...
  FOREIGN KEY(id) REFERENCES transaction(user_id));
```

Esta instrucción es incorrecta porque la relación entre `user` y `transaction` es de uno a muchos, definiendo que un usuario puede realizar varias transacciones. En este caso, la clave foránea debe estar en `transaction` (no en `user`), para que cada transacción esté correctamente asociada a un único usuario.

A su vez la clave primaria de `user` (`id`) no puede ser declarada como clave foránea apuntando a `transaction` (`user_id`), ya que cada usuario es único y su identificador no debe depender de la existencia de transacciones.

✚ Para corregir la estructura:

- Crear tabla `user` sin la clave foránea
 - ◆ Ingestar data
- Adición de la clave foránea en la tabla `transaction`
- Cambiar el nombre del campo `email` a `personal_email`
- Cambiar el nombre de la tabla de `user` a `data_user`

Al declarar `user_id` como llave foránea, ya no es necesario crear de manera explícita un índice en este campo. El sistema de bases de datos crea automáticamente un índice en los campos que son llave foránea.

La tabla se crea inicialmente con los nombres de los campos que vienen en el fichero de entrada. Después de la ingesta de los datos, se realizarán los cambios pertinentes en el nombre del campo y de la tabla. Este orden evita posibles errores de compatibilidad durante el proceso de carga.

- Creación de la tabla `user`

The screenshot shows a database management interface with a SQL editor. The editor contains the following SQL code:

```

121  ----- Ni
122  -- Nivel 3. Ejercicio 1
123  -- Creación de la tabla
124  CREATE TABLE IF NOT EXISTS user (
125      id INT PRIMARY KEY,
126      name VARCHAR(100),
127      surname VARCHAR(100),
128      phone VARCHAR(150),
129      email VARCHAR(150),
130      birth_date VARCHAR(100),
131      country VARCHAR(150),
132      city VARCHAR(150),
133      postal_code VARCHAR(100),
134      address VARCHAR(255));
135
  
```

Below the editor, there is an 'Output' section with a dropdown menu set to 'Action Output'. It displays the execution results of the SQL command:

#	Time	Action	Message
1	00:33:51	CREATE TABLE IF NOT EXISTS user (...	0 row(s) affected

- Ingesta de datos

Código archivo “datos_introducir_user1”

S301 Manipulación de tablas estructura_datos_user datos_introducir_user (1) x

Limit to 50000 rows

```

269 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (1, 'John', 'Doe', '1234567890', 'john.doe@example.com', '1990-01-01', 'USA', 'New York', '10001');
270 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (2, 'Jane', 'Smith', '9876543210', 'jane.smith@example.com', '1985-05-15', 'USA', 'Los Angeles', '90001');
271 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (3, 'Michael', 'Johnson', '5555555555', 'michael.johnson@example.com', '1978-12-20', 'USA', 'Chicago', '60601');
272 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (4, 'Emily', 'Brown', '4444444444', 'emily.brown@example.com', '1992-03-10', 'USA', 'Houston', '77001');
273 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (5, 'David', 'Wilson', '3333333333', 'david.wilson@example.com', '1988-07-25', 'USA', 'Phoenix', '85001');
274 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (6, 'Sophia', 'Davis', '2222222222', 'sophia.davis@example.com', '1995-09-05', 'USA', 'Philadelphia', '19101');
275 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (7, 'Daniel', 'Garcia', '1111111111', 'daniel.garcia@example.com', '1980-11-18', 'USA', 'San Antonio', '78201');
276 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (8, 'Olivia', 'Martinez', '0000000000', 'olivia.martinez@example.com', '1993-04-22', 'USA', 'San Diego', '92101');
277 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (9, 'Liam', 'Hernandez', '9999999999', 'liam.hernandez@example.com', '1987-06-30', 'USA', 'Dallas', '75201');
278 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (10, 'Ava', 'Lopez', '8888888888', 'ava.lopez@example.com', '1991-02-14', 'USA', 'San Jose', '95101');
279
280 • SET foreign_key_checks = 1;
  
```

Output

Action Output

#	Time	Action	Message
✓ 256	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (1, 'John', 'Doe', '1234567890', 'john.doe@example.com', '1990-01-01', 'USA', 'New York', '10001');	1 row(s) affected
✓ 257	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (2, 'Jane', 'Smith', '9876543210', 'jane.smith@example.com', '1985-05-15', 'USA', 'Los Angeles', '90001');	1 row(s) affected
✓ 258	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (3, 'Michael', 'Johnson', '5555555555', 'michael.johnson@example.com', '1978-12-20', 'USA', 'Chicago', '60601');	1 row(s) affected
✓ 259	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (4, 'Emily', 'Brown', '4444444444', 'emily.brown@example.com', '1992-03-10', 'USA', 'Houston', '77001');	1 row(s) affected
✓ 260	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (5, 'David', 'Wilson', '3333333333', 'david.wilson@example.com', '1988-07-25', 'USA', 'Phoenix', '85001');	1 row(s) affected
✓ 261	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (6, 'Sophia', 'Davis', '2222222222', 'sophia.davis@example.com', '1995-09-05', 'USA', 'Philadelphia', '19101');	1 row(s) affected
✓ 262	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (7, 'Daniel', 'Garcia', '1111111111', 'daniel.garcia@example.com', '1980-11-18', 'USA', 'San Antonio', '78201');	1 row(s) affected
✓ 263	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (8, 'Olivia', 'Martinez', '0000000000', 'olivia.martinez@example.com', '1993-04-22', 'USA', 'San Diego', '92101');	1 row(s) affected
✓ 264	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (9, 'Liam', 'Hernandez', '9999999999', 'liam.hernandez@example.com', '1987-06-30', 'USA', 'Dallas', '75201');	1 row(s) affected
✓ 265	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (10, 'Ava', 'Lopez', '8888888888', 'ava.lopez@example.com', '1991-02-14', 'USA', 'San Jose', '95101');	1 row(s) affected
✓ 266	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (11, 'Noah', 'Gonzalez', '7777777777', 'noah.gonzalez@example.com', '1989-08-01', 'USA', 'Austin', '78701');	1 row(s) affected
✓ 267	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (12, 'Isabella', 'Perez', '6666666666', 'isabella.perez@example.com', '1994-10-12', 'USA', 'Jacksonville', '32201');	1 row(s) affected
✓ 268	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (13, 'Mason', 'Rodriguez', '5555555555', 'mason.rodriguez@example.com', '1986-04-28', 'USA', 'Fort Worth', '76101');	1 row(s) affected
✓ 269	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (14, 'Mia', 'Taylor', '4444444444', 'mia.taylor@example.com', '1996-07-03', 'USA', 'Columbus', '31901');	1 row(s) affected
✓ 270	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (15, 'Ethan', 'Anderson', '3333333333', 'ethan.anderson@example.com', '1983-09-17', 'USA', 'San Francisco', '94101');	1 row(s) affected
✓ 271	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (16, 'Charlotte', 'Thomas', '2222222222', 'charlotte.thomas@example.com', '1997-11-24', 'USA', 'Indianapolis', '46201');	1 row(s) affected
✓ 272	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (17, 'Alexander', 'White', '1111111111', 'alexander.white@example.com', '1981-03-09', 'USA', 'San Jose', '95101');	1 row(s) affected
✓ 273	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (18, 'Amelia', 'Harris', '0000000000', 'amelia.harris@example.com', '1998-06-16', 'USA', 'Portland', '97201');	1 row(s) affected
✓ 274	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (19, 'Benjamin', 'Clark', '9999999999', 'benjamin.clark@example.com', '1984-12-02', 'USA', 'Seattle', '98101');	1 row(s) affected
✓ 275	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (20, 'Harper', 'Lewis', '8888888888', 'harper.lewis@example.com', '1999-01-27', 'USA', 'Denver', '80201');	1 row(s) affected
✓ 276	00:37:33	INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code) VALUES (21, 'Lucas', 'Walker', '7777777777', 'lucas.walker@example.com', '1987-05-11', 'USA', 'Boston', '02101');	1 row(s) affected
✓ 277	00:37:33	SET foreign_key_checks = 1	0 row(s) affected

- Adición de la clave foránea en la tabla **transaction**

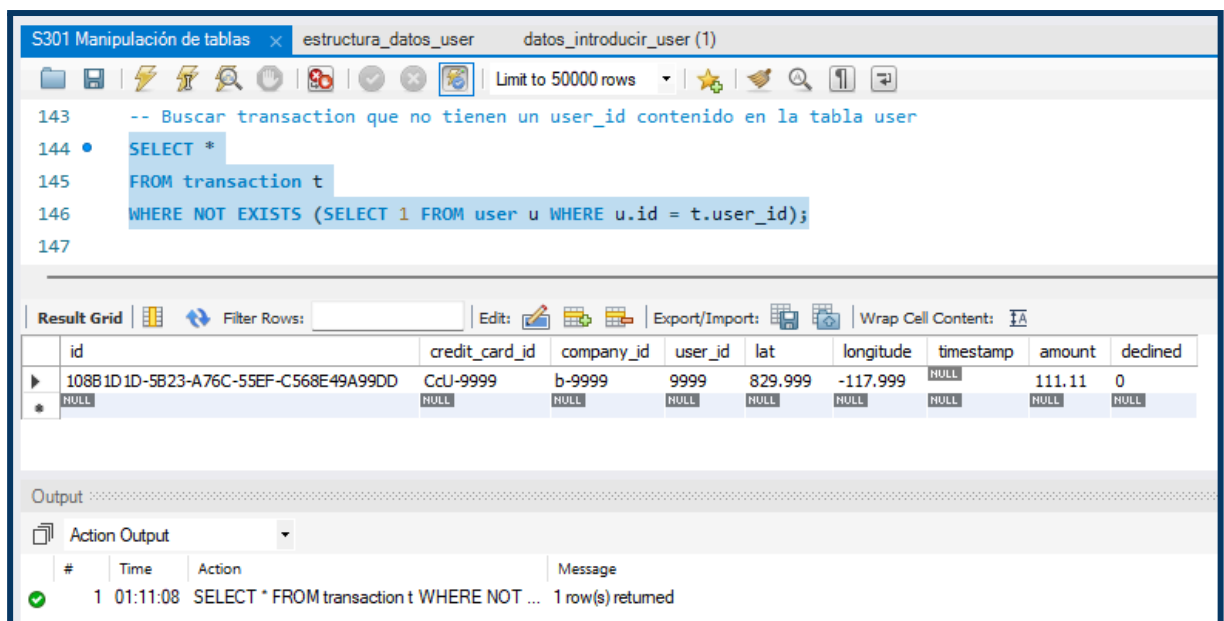


Al intentar agregar una clave foránea en la tabla **transaction** que haga referencia al **id** de la tabla **user**, da el siguiente error:

Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails (transactions.#sql-1bbc_1d, CONSTRAINT fk_transaction_user FOREIGN KEY (user_id) REFERENCES user (id) ON DELETE CASCADE) 0.031 sec

Esto se debe a que hay registros en la tabla **transaction** que tienen un valor en la columna **user_id** que no existe en la columna **id** de la tabla **user**.

Para resolverlo, lo primero sería comprobar qué valores están generando este problema.



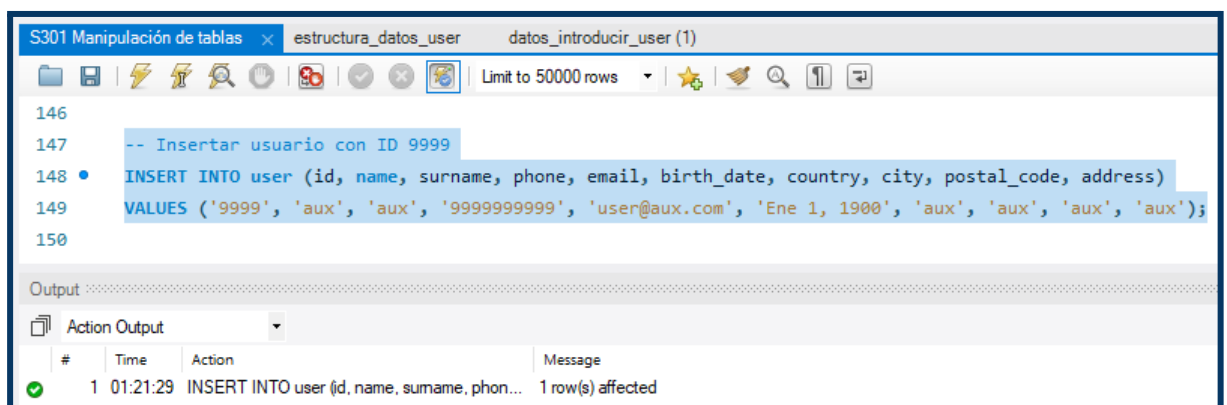
Se puede verificar que se trata de una única transacción, correspondiente a la insertada previamente en el ejercicio 3 del nivel 1.

Para mantener la coherencia con lo realizado en ese ejercicio, se procederá a insertar el usuario. Sin embargo, es importante recalcar que en un entorno productivo, esta probablemente no sería la solución más adecuada.

Al revisar la estructura de la tabla recientemente creada, se observa que, con excepción del campo `id` (definido como `INT`), el resto de los campos son de tipo `VARCHAR`. En este contexto se determinó lo siguiente:

- `id`: El dato que se pretende insertar coincide con el tipo de datos definido (`INT`)
- `email`: Se optó por usar el formato estándar como 'user@aux.com' anteriormente comentado.
- `birth_date`: Se optó por utilizar la fecha 'Ene 1, 1900', ya que es el formato encontrado en los valores de esta columna, lo que evita futuros conflictos en el manejo de los mismos. Además, se eligió esta fecha porque es convencionalmente utilizada en estos casos, garantizando que no afecte las estadísticas en análisis futuros.
- `phone`: Se optó por utilizar el valor '9999999999'. Aunque no se encontró un formato único en los valores de esta columna, se eligió este valor de 10 dígitos, con el cual se garantiza que no corresponda a un valor real y, a su vez, mantenga cierta concordancia con el resto. Además, es compatible con el tipo de dato `VARCHAR`.
- Restantes campos: Se optó por insertar como valor 'aux'.

- Crear usuario “9999”



The screenshot shows a SQL IDE window with the title 'S301 Manipulación de tablas'. The active tab is 'estructura_datos_user', and the sub-tab is 'datos_introducir_user (1)'. The SQL editor contains the following code:

```
146
147 -- Insertar usuario con ID 9999
148 • INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address)
149   VALUES ('9999', 'aux', 'aux', '9999999999', 'user@aux.com', 'Ene 1, 1900', 'aux', 'aux', 'aux', 'aux');
150
```

Below the editor, the 'Output' pane shows the 'Action Output' for the executed statement:

#	Time	Action	Message
1	01:21:29	INSERT INTO user (id, name, surname, phon...	1 row(s) affected

- Crea clave foránea en la tabla `transaction`

Se retoma la creación de la clave foránea, y esta vez se realiza con éxito.

The screenshot shows a SQL IDE window with the title 'S301 Manipulación de tablas'. The active tab is 'estructura_datos_user', and the sub-tab is 'datos_introducir_user (1)'. The SQL editor contains the following code:

```

137  -- Crear relación con tabla transaction
138  • ALTER TABLE transaction
139  ADD CONSTRAINT fk_transaction_user
140  FOREIGN KEY (user_id) REFERENCES user(id)
141  ON DELETE CASCADE;
142

```

Below the editor is the 'Output' section, which shows the 'Action Output' for the executed command:

#	Time	Action	Message
1	01:23:40	ALTER TABLE transaction ADD CONSTRAI...	587 row(s) affected Records: 587 Duplicates: 0 Warnings: 0

- Cambiar el nombre del campo `email` a `personal_email`

The screenshot shows a SQL IDE window with the title 'S301 Manipulación de tablas'. The active tab is 'estructura_datos_user', and the sub-tab is 'datos_introducir_user (1)'. The SQL editor contains the following code:

```

151
152  -- Modificar nombre del campo email
153  • ALTER TABLE user
154  RENAME COLUMN email TO personal_email;
155

```

Below the editor is the 'Output' section, which shows the 'Action Output' for the executed command:

#	Time	Action	Message
1	01:36:57	ALTER TABLE user RENAME COLUMN ema...	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

- Cambiar el nombre de la tabla `user` a `data_user`

The screenshot shows a SQL IDE window with the title 'S301 Manipulación de tablas'. The active tab is 'estructura_datos_user', and the sub-tab is 'datos_introducir_user (1)'. The SQL editor contains the following code:

```

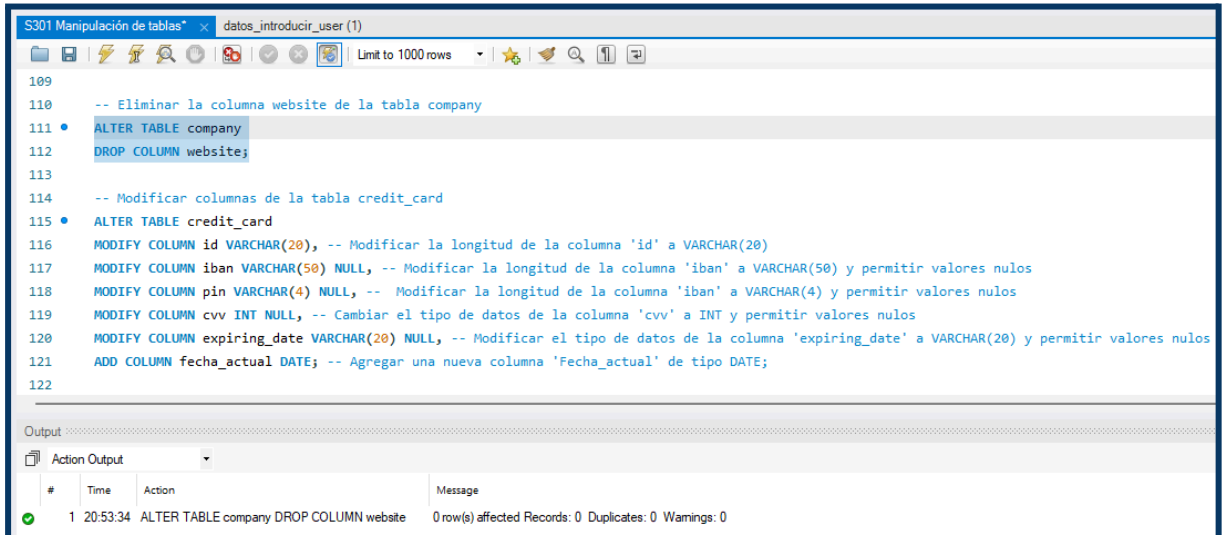
157  -- Modificar nombre de tabla user
158  • RENAME TABLE user TO data_user;
159

```

Below the editor is the 'Output' section, which shows the 'Action Output' for the executed command:

#	Time	Action	Message
1	01:40:26	RENAME TABLE user TO data_user	0 row(s) affected

- Modificaciones en tablas **company** y **credit_card**
 - Eliminar la columna website de la tabla **company**



```

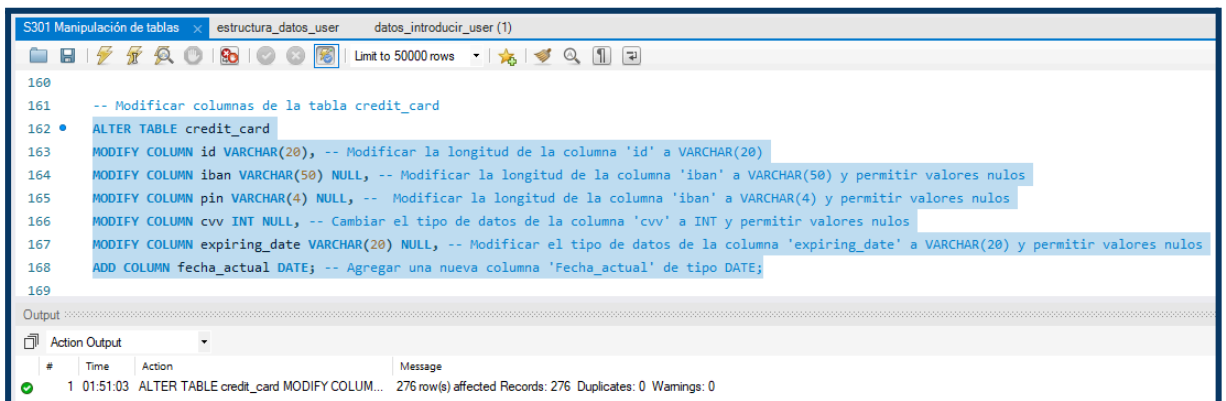
109
110 -- Eliminar la columna website de la tabla company
111 • ALTER TABLE company
112 DROP COLUMN website;
113
114 -- Modificar columnas de la tabla credit_card
115 • ALTER TABLE credit_card
116 MODIFY COLUMN id VARCHAR(20), -- Modificar la longitud de la columna 'id' a VARCHAR(20)
117 MODIFY COLUMN iban VARCHAR(50) NULL, -- Modificar la longitud de la columna 'iban' a VARCHAR(50) y permitir valores nulos
118 MODIFY COLUMN pin VARCHAR(4) NULL, -- Modificar la longitud de la columna 'iban' a VARCHAR(4) y permitir valores nulos
119 MODIFY COLUMN cvv INT NULL, -- Cambiar el tipo de datos de la columna 'cvv' a INT y permitir valores nulos
120 MODIFY COLUMN expiring_date VARCHAR(20) NULL, -- Modificar el tipo de datos de la columna 'expiring_date' a VARCHAR(20) y permitir valores nulos
121 ADD COLUMN fecha_actual DATE; -- Agregar una nueva columna 'Fecha_actual' de tipo DATE;
122

```

Output

#	Time	Action	Message
1	20:53:34	ALTER TABLE company DROP COLUMN website	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

- Modificar campos de la tabla **credit_card**



```

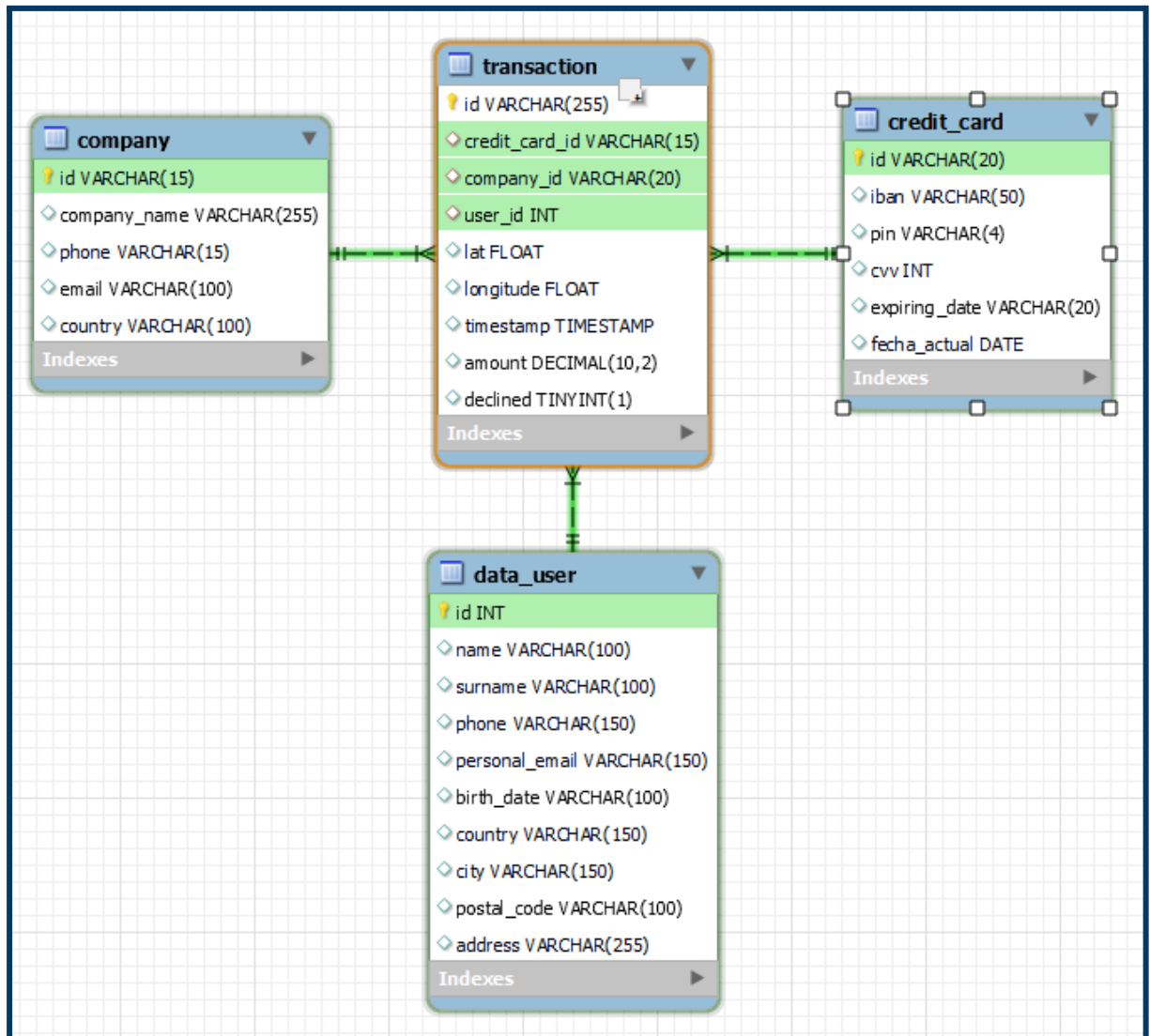
160
161 -- Modificar columnas de la tabla credit_card
162 • ALTER TABLE credit_card
163 MODIFY COLUMN id VARCHAR(20), -- Modificar la longitud de la columna 'id' a VARCHAR(20)
164 MODIFY COLUMN iban VARCHAR(50) NULL, -- Modificar la longitud de la columna 'iban' a VARCHAR(50) y permitir valores nulos
165 MODIFY COLUMN pin VARCHAR(4) NULL, -- Modificar la longitud de la columna 'iban' a VARCHAR(4) y permitir valores nulos
166 MODIFY COLUMN cvv INT NULL, -- Cambiar el tipo de datos de la columna 'cvv' a INT y permitir valores nulos
167 MODIFY COLUMN expiring_date VARCHAR(20) NULL, -- Modificar el tipo de datos de la columna 'expiring_date' a VARCHAR(20) y permitir valores nulos
168 ADD COLUMN fecha_actual DATE; -- Agregar una nueva columna 'Fecha_actual' de tipo DATE;
169

```

Output

#	Time	Action	Message
1	01:51:03	ALTER TABLE credit_card MODIFY COLUMN...	276 row(s) affected Records: 276 Duplicates: 0 Warnings: 0

- Generación del Modelo ER



3.2. Ejercicio 2

La empresa también te solicita crear una vista llamada "InformeTecnico" que contenga la siguiente información:

- ID de la transacción
- Nombre del usuario/a
- Apellido del usuario/a
- IBAN de la tarjeta de crédito usada.
- Nombre de la compañía de la transacción realizada.
- Asegúrate de incluir información relevante de ambas tablas y utiliza alias para cambiar de nombre columnas según sea necesario.
- Muestra los resultados de la vista, ordena los resultados de forma descendente en función de la variable ID de transacción.

- Crear la vista

The screenshot shows the SQL Developer interface with a script titled 'S301 Manipulación de tablas'. The script contains the following SQL code:

```

170
171 -- Nivel 3. Ejercicio 2
172 -- Creación de la vista InformeTecnico
173 • CREATE OR REPLACE VIEW InformeTecnico AS
174 SELECT t.id AS Id_transaccion, u.name AS Nombre, u.surname AS Apellido, cc.iban AS IBAN, c.company_name AS Compannia, t.amount AS Monto
175 FROM transaction t
176 JOIN data_user u ON t.user_id = u.id
177 JOIN credit_card cc ON t.credit_card_id = cc.id
178 JOIN company c ON t.company_id = c.id;
179

```

The 'Output' pane at the bottom shows the execution results:

#	Time	Action	Message
1	02:06:15	CREATE OR REPLACE VIEW InformeTecnico...	0 row(s) affected

- Mostrar resultados de la vista

The screenshot shows the SQL Developer interface with the same script as before, but now including a query to view the results of the 'InformeTecnico' view:

```

172 -- Creación de la vista InformeTecnico
173 • CREATE OR REPLACE VIEW InformeTecnico AS
174 SELECT t.id AS Id_transaccion, u.name AS Nombre, u.surname AS Apellido, cc.iban AS IBAN, c.company_name AS Compannia, t.amount AS Monto
175 FROM transaction t
176 JOIN data_user u ON t.user_id = u.id
177 JOIN credit_card cc ON t.credit_card_id = cc.id
178 JOIN company c ON t.company_id = c.id;
179
180 • SELECT * FROM InformeTecnico
181 ORDER BY Id_transaccion DESC;

```

The 'Result Grid' pane at the bottom displays the query results in a table format:

	Id_transaccion	Nombre	Apellido	IBAN	Compannia	Monto
▶	FE96CE47-BD59-381C-4E18-E3CA3D44E8FF	Kenyon	Hartman	DO26854763748537475216568689	Magna A Neque Industries	480.13
	FE809ED4-2DB6-55AC-C915-929516E46468	Molly	Gilliam	SE2813123487163628531121	Nunc Interdum Incorporated	219.83
	FD9CBCCD-8E1E-8DA1-4606-7E3A6F3A5A65	Linus	Willis	KW9485332754781757886242955643	Nunc Interdum Incorporated	42.32
	FD89D51B-AE8D-77DC-E450-B8083FBD3187	Hilda	Levy	LT053237077744561475	Malesuada PC	200.72
	FD2E8957-414B-BEEC-E9AD-59AA7A8A6290	Hedwig	Gilbert	GE84848451582810541526	Neque Tellus Imperdiet Corp.	78.29
	FCE2AB9A-271D-2BDC-9E49-8DD92A373391	Hakeem	Alford	MD1234119525145401270486	Nunc Interdum Incorporated	335.56
	FBD7E0D6-8A6B-F5BC-0CA9-EA4B8760100C	Hedwig	Gilbert	MU4132333444534342541344788855	Mauris Id Inc.	207.09
	FAC76A80-8448-69AA-E892-426C2F12621C	Slade	Poole	MT05JWCF58868200575771634583813	Arcu LLP	304.95
	FAAD3FFC-1A17-E141-43D3-359A5BA7CB38	Hedwig	Gilbert	GE90157928843338134463	Lorem Eu Incorporated	149.84

The 'Output' pane at the bottom shows the execution results:

#	Time	Action	Message
1	02:07:23	SELECT * FROM InformeTecnico ORDER BY B...	587 row(s) returned