

Tasca S3.01. Manipulació de taules

Nivell 1

Exercici 1

En esta tarea, se elaboró la tabla `credit_card`. Viendo los valores escritos en el archivo `dades_introduir_credit`, se decidió que todas las columnas fueran de tipo `VARCHAR`. Una vez creada la tabla, se creó la relación con la tabla `transaction`.

The screenshot displays a database management interface with a SQL editor and a results panel.

SQL Editor:

```
19 -- Creamos la tabla credit_card
20 CREATE TABLE IF NOT EXISTS credit_card (
21     id VARCHAR(15) PRIMARY KEY UNIQUE NOT NULL,
22     iban VARCHAR(100) NOT NULL,
23     pan VARCHAR(100) NOT NULL,
24     pin VARCHAR(20) NOT NULL,
25     cvv VARCHAR(20) NOT NULL,
26     expiring_date VARCHAR(50) NOT NULL
27 );
28
29 -- Añadimos información de dades_introduir_credit
30
31 -- Confirmamos que no existe una credit cards en transaction que no este en credit_card
32 SELECT *
33 FROM transaction
34 WHERE credit_card_id NOT IN (SELECT DISTINCT(id)
35                             FROM credit_card);
36
37 -- Relación entre transaction y credit_card
38 /*ALTER TABLE transaction
39 ADD CONSTRAINT fk_credit_card
40 FOREIGN KEY (credit_card_id)
41 REFERENCES credit_card(id);*/
```

Result Grid:

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

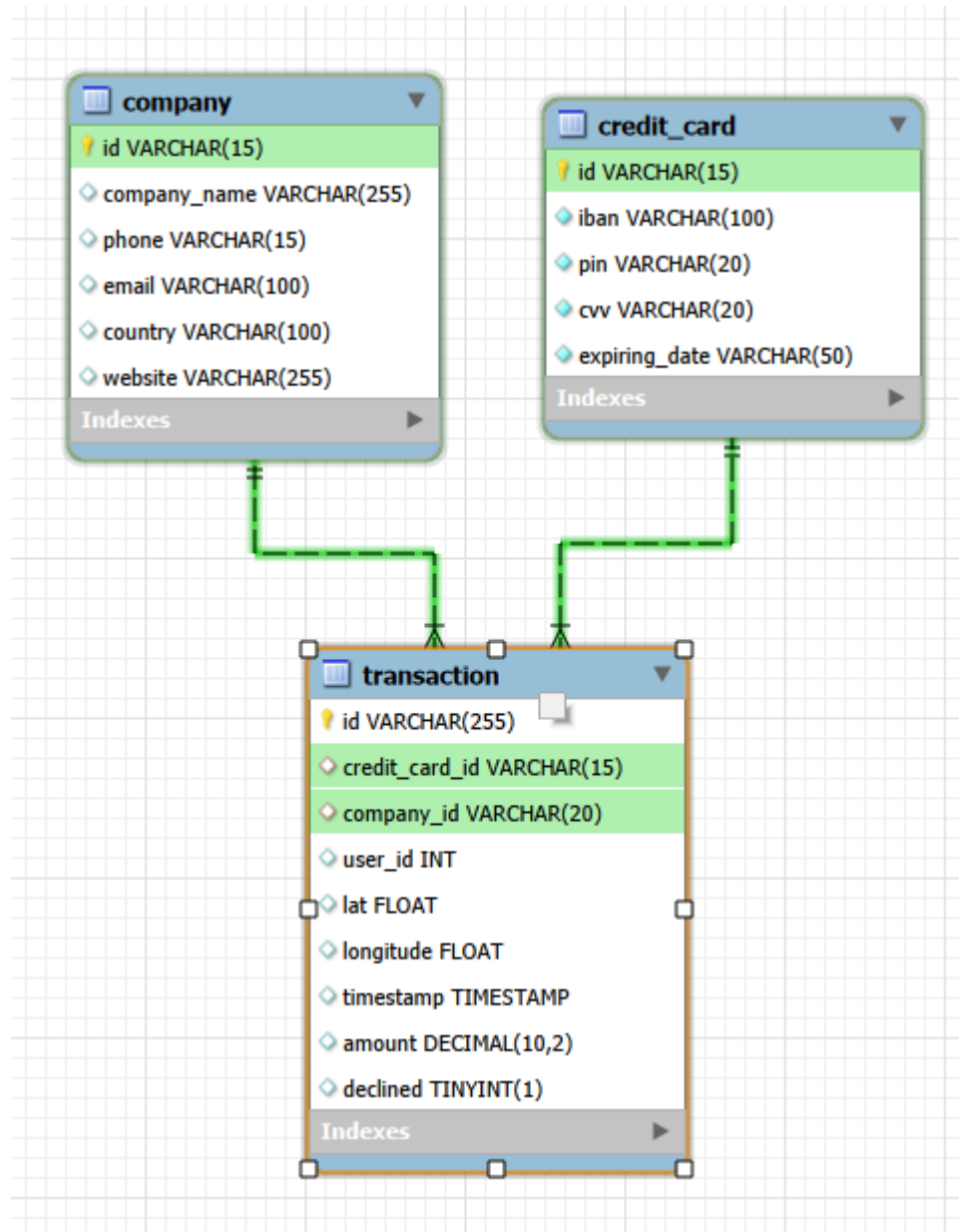
transaction 10 x [Apply] [Revert]

Output:

Action Output

#	Time	Action	Message	Duration / Fetch
17	20:24:57	CREATE TABLE IF NOT EXISTS credit_card (id ...	0 row(s) affected, 1 warning(s): 1050 Table 'credit...	0.016 sec
18	20:25:42	SELECT * FROM transaction WHERE credit_car...	0 row(s) returned	0.203 sec / 0.000 sec

En la siguiente figura se muestran las relaciones entre las tres tablas. Podemos observar una relación n-to-1 entre las tablas transaction y company, unidas en transaction.company_id y company.id. Esto se explica a que una misma compañía puede haber realizado diversas transacciones con la empresa dedicada a la venta de productos en línea. También observamos la relación n-to-1 entre las tablas transaction y credit_card, unidas en transaction.credit_card_id y credit_card.id. Esto se explica a que una misma tarjeta de crédito puede usarse en más de una transacción.



Exercici 2

Cambiamos el número de cuenta.

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

```
43 -- Exercici 2
44 • UPDATE credit_card
45 SET iban='TR323456312213576817699999'
46 WHERE id='CcU-2938';
47
48 -- Confirmamos el cambio
49 • SELECT *
50 FROM credit_card
51 WHERE id='CcU-2938';
52
53 -- Exercici 3
```

Below the editor is a toolbar with icons for various functions. The main area displays the 'Result Grid' with the following data:

	id	iban	pin	cvv	expiring_date
▶	CcU-2938	TR323456312213576817699999	3257	984	10/30/22
*	NULL	NULL	NULL	NULL	NULL

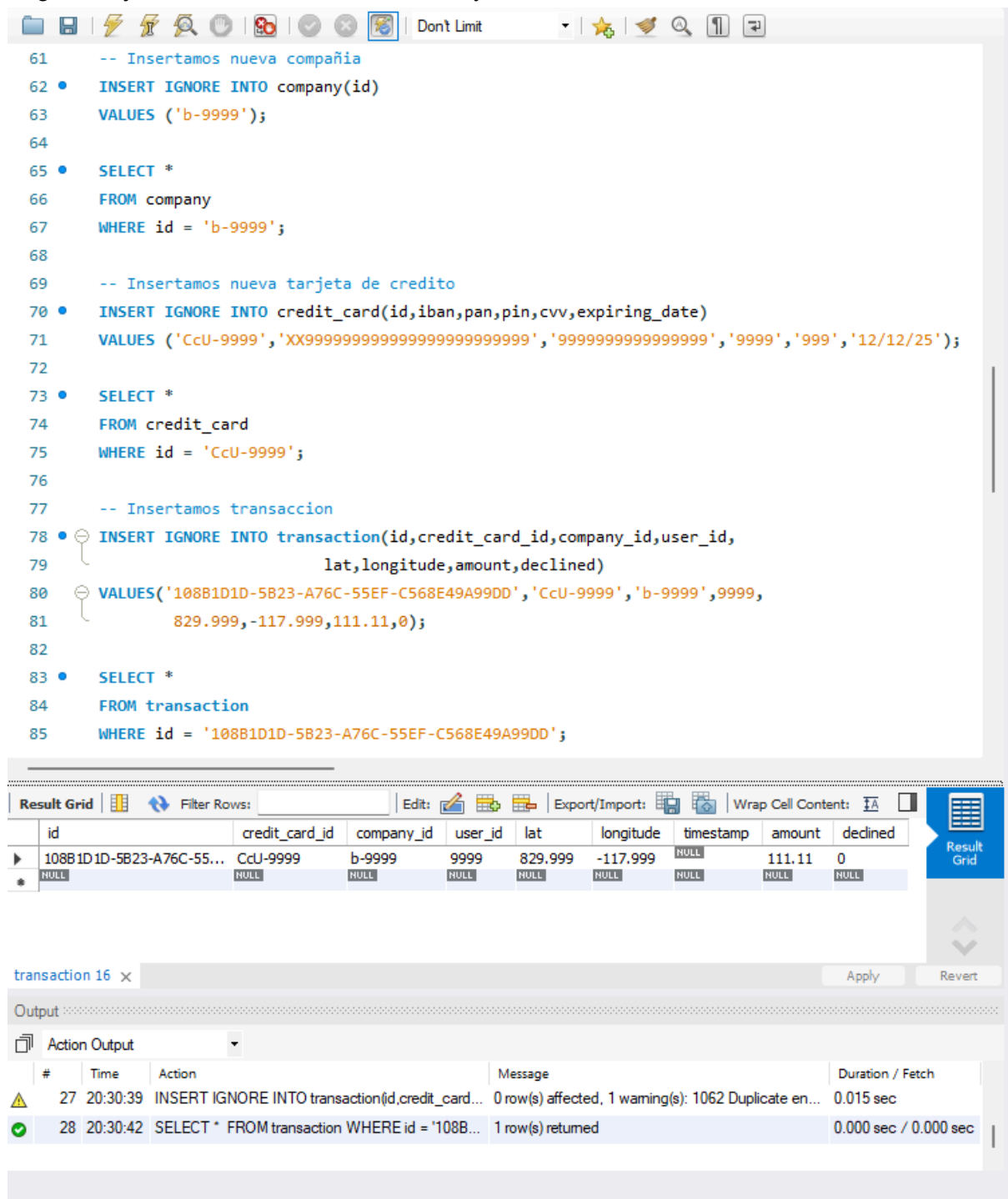
On the right side, there is a vertical toolbar with icons for 'Result Grid', 'Form Editor', 'Field Types', 'Query Stats', and 'Execution Plan'. Below the result grid, there is a tab labeled 'credit_card 11' with 'Apply' and 'Revert' buttons.

The 'Output' section shows the following actions:

#	Time	Action	Message	Duration / Fetch
✓ 19	20:29:10	UPDATE credit_card SET iban='TR32345631221...	0 row(s) affected Rows matched: 1 Changed: 0 ...	0.000 sec
✓ 20	20:29:13	SELECT * FROM credit_card WHERE id='CcU-2...	1 row(s) returned	0.000 sec / 0.000 sec

Exercici 3

Para ingresar la nueva transacción, nos dimos cuenta que teníamos que agregar una nueva empresa y una nueva tarjeta de crédito a sus respectivas tablas, debido a que existen relaciones con keys entre las tres tablas. Primero agregamos la compañía de id 'b-9999', luego la tarjeta de crédito de id 'CcU-9999' y finalmente la nueva transacción.



The screenshot displays a SQL IDE interface with a query editor at the top and a results panel at the bottom. The query editor contains SQL code for inserting and selecting data from three tables: company, credit_card, and transaction. The results panel shows the output of the last query, which is a SELECT statement filtering for a specific transaction ID. Below the results panel, the 'Action Output' section shows the execution log of the queries.

```
61 -- Insertamos nueva compañía
62 • INSERT IGNORE INTO company(id)
63 VALUES ('b-9999');
64
65 • SELECT *
66 FROM company
67 WHERE id = 'b-9999';
68
69 -- Insertamos nueva tarjeta de credito
70 • INSERT IGNORE INTO credit_card(id,iban,pan,pin, cvv,expiring_date)
71 VALUES ('CcU-9999','XX99999999999999999999999999999999','9999999999999999','9999','999','12/12/25');
72
73 • SELECT *
74 FROM credit_card
75 WHERE id = 'CcU-9999';
76
77 -- Insertamos transaccion
78 • INSERT IGNORE INTO transaction(id,credit_card_id,company_id,user_id,
79 lat,longitude,amount,declined)
80 VALUES('108B1D1D-5B23-A76C-55EF-C568E49A99DD','CcU-9999','b-9999',9999,
81 829.999,-117.999,111.11,0);
82
83 • SELECT *
84 FROM transaction
85 WHERE id = '108B1D1D-5B23-A76C-55EF-C568E49A99DD';
```

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

transaction 16 x Apply Revert

Output

Action Output

#	Time	Action	Message	Duration / Fetch
27	20:30:39	INSERT IGNORE INTO transaction(id,credit_card...	0 row(s) affected, 1 warning(s): 1062 Duplicate en...	0.015 sec
28	20:30:42	SELECT * FROM transaction WHERE id = '108B...	1 row(s) returned	0.000 sec / 0.000 sec

Exercici 4

Eliminamos la columna 'pan' de la tabla credit_card.

The screenshot shows a database management tool interface. The top section displays SQL code for Exercise 4:

```
85 WHERE id = '108B1D1D-5B23-A76C-55EF-C568E49A99DD';
86
87 -- Exercici 4
88 • ALTER TABLE credit_card
89 DROP COLUMN pan;
90
91 • SHOW COLUMNS FROM credit_card;
92
93 -- Nivell 2
94 Exercici 1
```

Below the code, the 'Result Grid' shows the structure of the 'credit_card' table:

Field	Type	Null	Key	Default	Extra
id	varchar(15)	NO	PRI	NULL	
iban	varchar(100)	NO		NULL	
pin	varchar(20)	NO		NULL	
cvv	varchar(20)	NO		NULL	
expiring_date	varchar(50)	NO		NULL	

On the right side, there is a vertical toolbar with icons for 'Result Grid', 'Form Editor', 'Field Types', 'Query Stats', and 'Execution Plan'.

At the bottom, the 'Output' section shows the execution results for 'Result 17':

#	Time	Action	Message	Duration / Fetch
29	20:31:57	ALTER TABLE credit_card DROP COLUMN pan	Error Code: 1091. Can't DROP 'pan'; check that c...	0.016 sec
30	20:31:59	SHOW COLUMNS FROM credit_card	5 row(s) returned	0.000 sec / 0.000 sec

Nivell 2

Exercici 1

Eliminamos la transacción indicada.

The screenshot displays a database management interface with a SQL editor and a results pane.

SQL Editor:

```
97 WHERE id = '000447FE-B650-4DCF-85DE-C7ED0EE1CAAD';
98
99 • DELETE IGNORE FROM transaction
100 WHERE id = '000447FE-B650-4DCF-85DE-C7ED0EE1CAAD';
101
102 • SELECT *
103 FROM transaction
104 WHERE id = '000447FE-B650-4DCF-85DE-C7ED0EE1CAAD';
105
106
```

Result Grid:

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

Execution Log:

#	Time	Action	Message	Duration / Fetch
✓ 31	20:32:42	DELETE IGNORE FROM transaction WHERE id ...	0 row(s) affected	0.016 sec
✓ 32	20:32:48	SELECT * FROM transaction WHERE id = '00044...	0 row(s) returned	0.000 sec / 0.000 sec

Exercici 2

Elaboramos la vista que se nos pedía. Utilizamos un INNER JOIN para relacionar las tablas y para poder hacer el GROUP BY con las otras columnas, decidimos utilizar GROUP_CONCAT(DISTINCT()). Solo consideramos las transacciones donde declined=0.

The screenshot shows a SQL IDE interface with a script editor at the top and a result grid at the bottom. The script editor contains the following SQL code:

```
107 -- ANY_VALUE()
108 • CREATE OR REPLACE VIEW VistaMarketing AS
109 SELECT company_name AS compañía, GROUP_CONCAT(DISTINCT(phone)) AS teléfono,
110        GROUP_CONCAT(DISTINCT(country)) AS país_residencia, ROUND(AVG(amount),2) AS media_compra
111 FROM company AS c
112 INNER JOIN transaction AS t
113 ON c.id = t.company_id
114 WHERE declined=0
115 GROUP BY company_name;
116
117 • SELECT *
118 FROM VistaMarketing
119 ORDER BY media_compra DESC;
```

The result grid displays the output of the SELECT statement, showing a list of companies with their phone numbers, countries, and average purchase amounts. The columns are: compañía, teléfono, país_residencia, and media_compra. The data is sorted by media_compra in descending order.

compañía	téléfono	país_residencia	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
At Associates	09 56 61 10 65	New Zealand	272.74
Metus Vitae Associates	08 25 44 40 66	Australia	270.05
Aliquet Diam Limited	02 76 61 47 46	United States	269.29
Nec Luctus LLC	02 14 71 75 73	Norway	268.60
Neque Tellus Incorporated	04 43 18 34 19	Ireland	267.56
Cras Conditio	07 50 10 85 63	Belgium	267.28

The output section shows the execution log with the following entries:

#	Time	Action	Message	Duration / Fetch
✓ 17	11:03:25	CREATE OR REPLACE VIEW VistaMarketing AS...	0 row(s) affected	0.016 sec
✓ 18	11:03:29	SELECT * FROM VistaMarketing ORDER BY me...	101 row(s) returned	0.782 sec / 0.000 sec

Exercici 3

Filtramos la vista para mostrar solo las compañías con país 'Germany'.

The screenshot shows a database management interface with a SQL editor at the top and a results grid below. The SQL query is as follows:

```
119 ORDER BY media_compra DESC;
120
121 -- Exercici 3
122 • SELECT *
123 FROM VistaMarketing
124 WHERE país_residència = 'Germany'
125 ORDER BY media_compra DESC;
126
```

The results grid displays the following data:

compañía	teléfono	país_residència	media_compra
Ac Fermentum Incorporated	06 85 56 52 33	Germany	284.91
Nunc Interdum Incorporated	05 18 15 48 13	Germany	259.32
Convallis In Incorporated	06 66 57 29 50	Germany	257.69
Ac Industries	09 34 65 40 60	Germany	255.17
Rutrum Non Inc.	02 66 31 61 09	Germany	255.14
Auctor Mauris Corp.	05 62 87 14 41	Germany	254.68
Augue Foundation	06 88 43 15 63	Germany	253.56
Aliquam PC	01 45 73 52 16	Germany	252.96

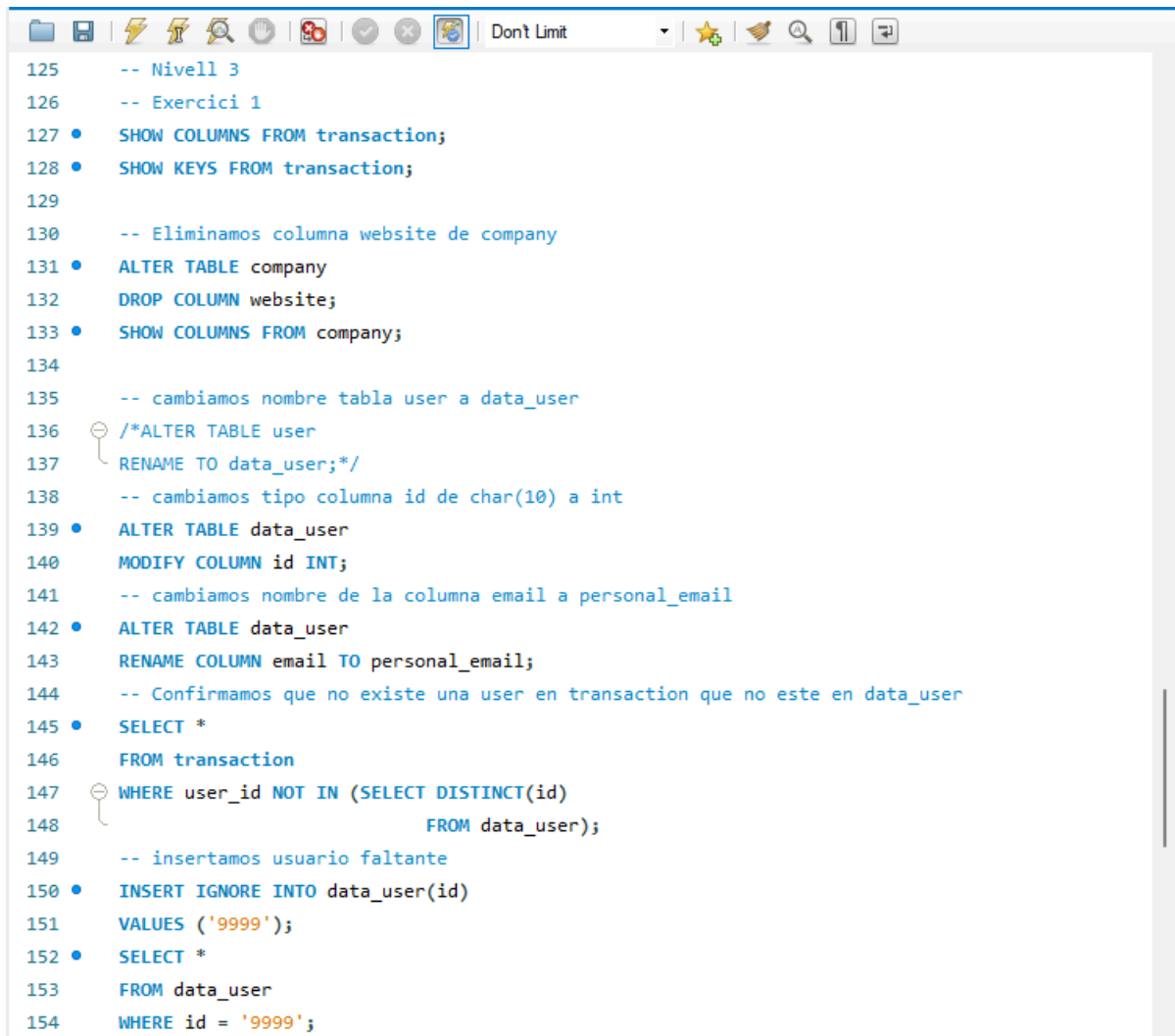
Below the results grid, the 'Output' section shows the execution log:

#	Time	Action	Message	Duration / Fetch
✓ 18	11:03:29	SELECT * FROM VistaMarketing ORDER BY me...	101 row(s) returned	0.782 sec / 0.000 sec
✓ 19	11:08:03	SELECT * FROM VistaMarketing WHERE país_f...	8 row(s) returned	0.859 sec / 0.000 sec

Nivell 3

Exercici 1

Realizamos diversos cambios para obtener el diagrama indicado.



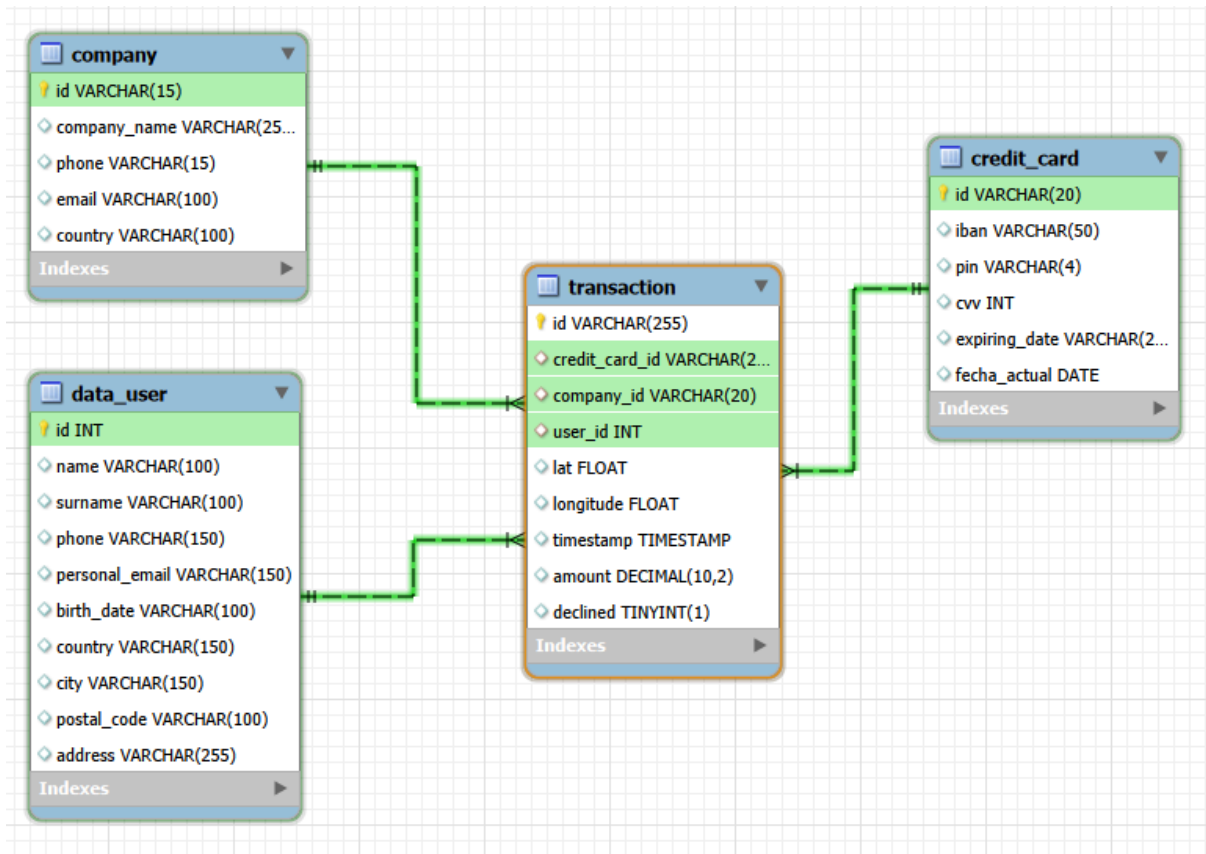
```
125 -- Nivell 3
126 -- Exercici 1
127 • SHOW COLUMNS FROM transaction;
128 • SHOW KEYS FROM transaction;
129
130 -- Eliminamos columna website de company
131 • ALTER TABLE company
132 DROP COLUMN website;
133 • SHOW COLUMNS FROM company;
134
135 -- cambiamos nombre tabla user a data_user
136 /*ALTER TABLE user
137 RENAME TO data_user;*/
138 -- cambiamos tipo columna id de char(10) a int
139 • ALTER TABLE data_user
140 MODIFY COLUMN id INT;
141 -- cambiamos nombre de la columna email a personal_email
142 • ALTER TABLE data_user
143 RENAME COLUMN email TO personal_email;
144 -- Confirmamos que no existe una user en transaction que no este en data_user
145 • SELECT *
146 FROM transaction
147 WHERE user_id NOT IN (SELECT DISTINCT(id)
148 FROM data_user);
149 -- insertamos usuario faltante
150 • INSERT IGNORE INTO data_user(id)
151 VALUES ('9999');
152 • SELECT *
153 FROM data_user
154 WHERE id = '9999';
```

```

155 -- Relación entre transaction y data_user
156 /*ALTER TABLE transaction
157 ADD CONSTRAINT fk_data_user
158 FOREIGN KEY (user_id)
159 REFERENCES data_user(id);*/
160 • SHOW COLUMNS FROM data_user;
161
162 • SHOW COLUMNS FROM credit_card;
163 -- cambiamos tipo columna iban de varchar(100) a varchar(50)
164 • ALTER TABLE credit_card
165 MODIFY COLUMN iban VARCHAR(50);
166 -- cambiamos tipo columna pin de varchar(20) a varchar(4)
167 • ALTER TABLE credit_card
168 MODIFY COLUMN pin VARCHAR(4);
169 -- cambiamos tipo columna cvv de varchar(20) a INT
170 • ALTER TABLE credit_card
171 MODIFY COLUMN cvv INT;
172 -- cambiamos tipo columna expiring_date de varchar(50) a varchar(255)
173 • ALTER TABLE credit_card
174 MODIFY COLUMN expiring_date VARCHAR(255);
175 -- añadimos columna fecha_actual
176 • ALTER TABLE credit_card
177 ADD COLUMN fecha_actual DATE;
178 -- para modificar columna id, primero eliminamos la conexión con transaction
179 /*ALTER TABLE transaction
180 DROP FOREIGN KEY fk_credit_card;*/
181 -- cambiamos columna id y credit_card_id de varchar(15) a varchar(20) en credit_card y transacti
182 • ALTER TABLE credit_card
183 MODIFY COLUMN id VARCHAR(20);
184 • ALTER TABLE transaction
185 MODIFY COLUMN credit_card_id VARCHAR(20);
186 -- Relación entre transaction y credit_card
187 /*ALTER TABLE transaction
188 ADD CONSTRAINT fk_credit_card
189 FOREIGN KEY (credit_card_id)
190 REFERENCES credit_card(id);*/
191

```

Tras los cambios indicados, obtuvimos el siguiente diagrama.



Exercici 2

Creemos la vista indicada, usando múltiples INNER JOIN para unir las tablas. Por nuestra interpretación del enunciado, decidimos también agregar las columnas país de la compañía, país del usuario, cantidad de la transacción y si fue rechazada o no.

The screenshot shows a SQL IDE interface. The top pane contains the SQL code for creating and querying a view. The bottom pane shows the execution results, including a table of data and an action log.

```
195 -- ANY_VALUE()
196 • CREATE OR REPLACE VIEW InformeTecnico AS
197   SELECT t.id AS ID_transaccion, name AS nombre_usuario, surname AS apellido_usuario,
198          iban, company_name AS compania, c.country AS pais_compania,
199          d.country AS pais_usuario, amount AS cantidad , declined AS rechazado
200   FROM transaction AS t
201   INNER JOIN data_user AS d
202   ON t.user_id = d.id
203   INNER JOIN credit_card AS cc
204   ON t.credit_card_id = cc.id
205   INNER JOIN company AS c
206   ON t.company_id = c.id;
207
208 • SELECT *
209   FROM InformeTecnico
210   ORDER BY ID_transaccion DESC;
```

Result Grid

ID_transaccion	nombre_usuario	apellido_usuario	iban	compania	pais_compania	pais_usuario	cantidad	rechaza
FFFD31D6-9495...	Bmrqli	Tprvvmrc	XX79...	Turpis Co...	Netherlands	United Kin...	74.54	0
FFFCF76D-ECF...	Dfrled	Vilqcjdl	XX63...	Amet Null...	Italy	Netherlands	148.91	0
FFFC9E8D-27C...	Securp	Faofvqfy	XX16...	Nunc Inte...	Germany	Sweden	234.22	0
FFFB270D-F53A...	Ggzjpa	Uirzjulh	XX39...	Viverra D...	United Kingdom	Portugal	349.13	0
FFF9E3CE-234E...	Yshimq	Zpsjsleed	XX88...	Convallis I...	Germany	Germany	247.39	0

Informetecnico 12 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
20	11:09:16	CREATE OR REPLACE VIEW InformeTecnico A...	0 row(s) affected	0.063 sec
21	11:09:18	SELECT * FROM InformeTecnico ORDER BY ID...	100000 row(s) returned	1.531 sec / 0.125 sec

Revisión peer-to-peer

Revisado por Rubén Serra

- Al hacer la vista, tener en cuenta que puede ser importante filtrar por declined.
- En los ejercicios de la vista, poner el ORDER BY fuera de la creación de la vista.
- Cuando se haga el cambio de tipo en diversas columnas de una tabla, hacer todo al mismo tiempo en un MODIFY, para mayor eficiencia.