

SPRINT S4.01 – CREACIÓN DE BASE DE DATOS**NIVEL 1**

Descargar los archivos CSV, estudiar y diseñar una base de datos con un esquema de estrella que contenga al menos 4 tablas de las que se pueda realizar algunas consultas.

Bien, hemos analizado el contenido de cada archivo y observado el encabezado en cada uno para crear las columnas de las tablas de manera apropiada.

Ahora enumeraremos en pasos la secuencia de acciones realizadas.

PASO 1: Se crea una nueva BD denominada transactionsTS4:

```
23
24   # PASO 1: Se crea una nueva BD denominada transactionsTS4:
25
26 • CREATE DATABASE transactionsTS4;
27 • USE transactionsTS4;
28
```

Output



Action Output

	#	Time	Action	Message
✓	472	13:53:39	CREATE DATABASE transactionsTS4	1 row(s) affected
✓	473	13:53:41	USE transactionsTS4	0 row(s) affected

PASO 2: Ahora se crean 4 tablas, según lo solicitado en el enunciado. Las tablas a crear serían **companies**, **credit_cards**, **users** y **transactions**.

```
18
19 • ○ CREATE TABLE companies (
20     company_id VARCHAR(10) PRIMARY KEY,
21     company_name VARCHAR(255),
22     phone VARCHAR(20),
23     email VARCHAR(60),
24     country VARCHAR(60),
25     website VARCHAR(60)
26 );
27
28 • ○ CREATE TABLE credit_cards (
29     id VARCHAR(15) PRIMARY KEY,
30     user_id VARCHAR(10),
31     iban VARCHAR(50),
32     pan VARCHAR(50),
33     pin VARCHAR(4),
34     cvv VARCHAR(3),
35     track1 VARCHAR(60),
36     track2 VARCHAR(60),
37     expiring_date VARCHAR(10)
38 );
39
```

```

40
41 • CREATE TABLE users (
42     id INT PRIMARY KEY,
43     NAME VARCHAR(60),
44     surname VARCHAR(60),
45     phone VARCHAR(15),
46     email VARCHAR(60),
47     birth_date VARCHAR(30),
48     country VARCHAR(60),
49     city VARCHAR(60),
50     postal_code VARCHAR(15),
51     address VARCHAR(250)
52 );
53
54 • CREATE TABLE transactions (
55     id VARCHAR(255) PRIMARY KEY,
56     card_id VARCHAR(15),
57     business_id VARCHAR(15),
58     TIMESTAMP VARCHAR(35),
59     amount DECIMAL(15,2),
60     declined BOOLEAN,
61     product_ids VARCHAR(25),
62     user_id INT,
63     lat VARCHAR(60),
64     longitude VARCHAR(60)
65 );

```

Output



Action Output

	#	Time	Action	Message
✓	474	13:58:44	CREATE TABLE companies (company_id VARCH...	0 row(s) affected
✓	475	13:58:47	CREATE TABLE credit_cards (id VARCHAR(15) ...	0 row(s) affected
✓	476	13:58:50	CREATE TABLE users (id INT PRIMARY KEY, ...	0 row(s) affected
✓	477	13:58:52	CREATE TABLE transactions (id VARCHAR(255) ...	0 row(s) affected

Las relaciones con las Foreign key en la tabla transactions se crearán más adelante, luego de cargar la data.

PASO 3: Ahora insertaremos la data de cada archivo en la correspondiente tabla ya creada en el paso anterior.

```

70
71 • LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/companies.csv"
72 INTO TABLE companies
73 FIELDS TERMINATED BY ','
74 ENCLOSED BY '"'
75 LINES TERMINATED BY '\r\n'
76 IGNORE 1 ROWS;
77
78
79 • LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/credit_cards.csv"
80 INTO TABLE credit_cards
81 FIELDS TERMINATED BY ','
82 ENCLOSED BY '"'
83 LINES TERMINATED BY '\n'
84 IGNORE 1 ROWS;
85
86
87 • LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/transactions.csv"
88 INTO TABLE transactions
89 FIELDS TERMINATED BY ';'
90 ENCLOSED BY '"'
91 LINES TERMINATED BY '\r\n'
92 IGNORE 1 ROWS;
93

```

Output :

📄 Action Output ▼

#	Time	Action	Message
✓ 478	14:04:37	LOAD DATA INFILE "C:/ProgramData/MySQL/M...	100 row(s) affected Records: 100 Deleted: 0 Skipped: 0 Warnings: 0
✓ 479	14:04:39	LOAD DATA INFILE "C:/ProgramData/MySQL/M...	275 row(s) affected Records: 275 Deleted: 0 Skipped: 0 Warnings: 0
✓ 480	14:04:46	LOAD DATA INFILE "C:/ProgramData/MySQL/M...	587 row(s) affected Records: 587 Deleted: 0 Skipped: 0 Warnings: 0

Para el caso de los 3 archivos: **users_ca**, **users_uk** y **users_usa**; se cargarán en la misma tabla **users**, como se observa a continuación:

```

96
97 • LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_ca.csv"
98 INTO TABLE users
99 FIELDS TERMINATED BY ','
100 ENCLOSED BY '"'
101 LINES TERMINATED BY '\r\n'
102 IGNORE 1 ROWS;
103
104 • LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_uk.csv"
105 INTO TABLE users
106 FIELDS TERMINATED BY ','
107 ENCLOSED BY '"'
108 LINES TERMINATED BY '\r\n'
109 IGNORE 1 ROWS;
110
111 • LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users_usa.csv"
112 INTO TABLE users
113 FIELDS TERMINATED BY ','
114 ENCLOSED BY '"'
115 LINES TERMINATED BY '\r\n'
116 IGNORE 1 ROWS;
117

```

Output



Action Output

#	Time	Action	Message
✓ 481	14:07:25	LOAD DATA INFILE "C:/ProgramData/MySQL/M...	75 row(s) affected Records: 75 Deleted: 0 Skipped: 0 Warnings: 0
✓ 482	14:07:28	LOAD DATA INFILE "C:/ProgramData/MySQL/M...	50 row(s) affected Records: 50 Deleted: 0 Skipped: 0 Warnings: 0
✓ 483	14:07:29	LOAD DATA INFILE "C:/ProgramData/MySQL/M...	150 row(s) affected Records: 150 Deleted: 0 Skipped: 0 Warnings: 0

PASO 4: Ahora si podemos establecer las relaciones entre las tablas a través de las Foreign key.

De esta manera se habrán establecido correctamente las relaciones entre la tabla de hechos y las tablas de dimensiones.

```

121 • ALTER TABLE transactions
122     ADD FOREIGN KEY(business_id) REFERENCES companies(company_id),
123     ADD FOREIGN KEY(card_id) REFERENCES credit_cards(id),
124     ADD FOREIGN KEY(user_id) REFERENCES users(id);
125

```

Output

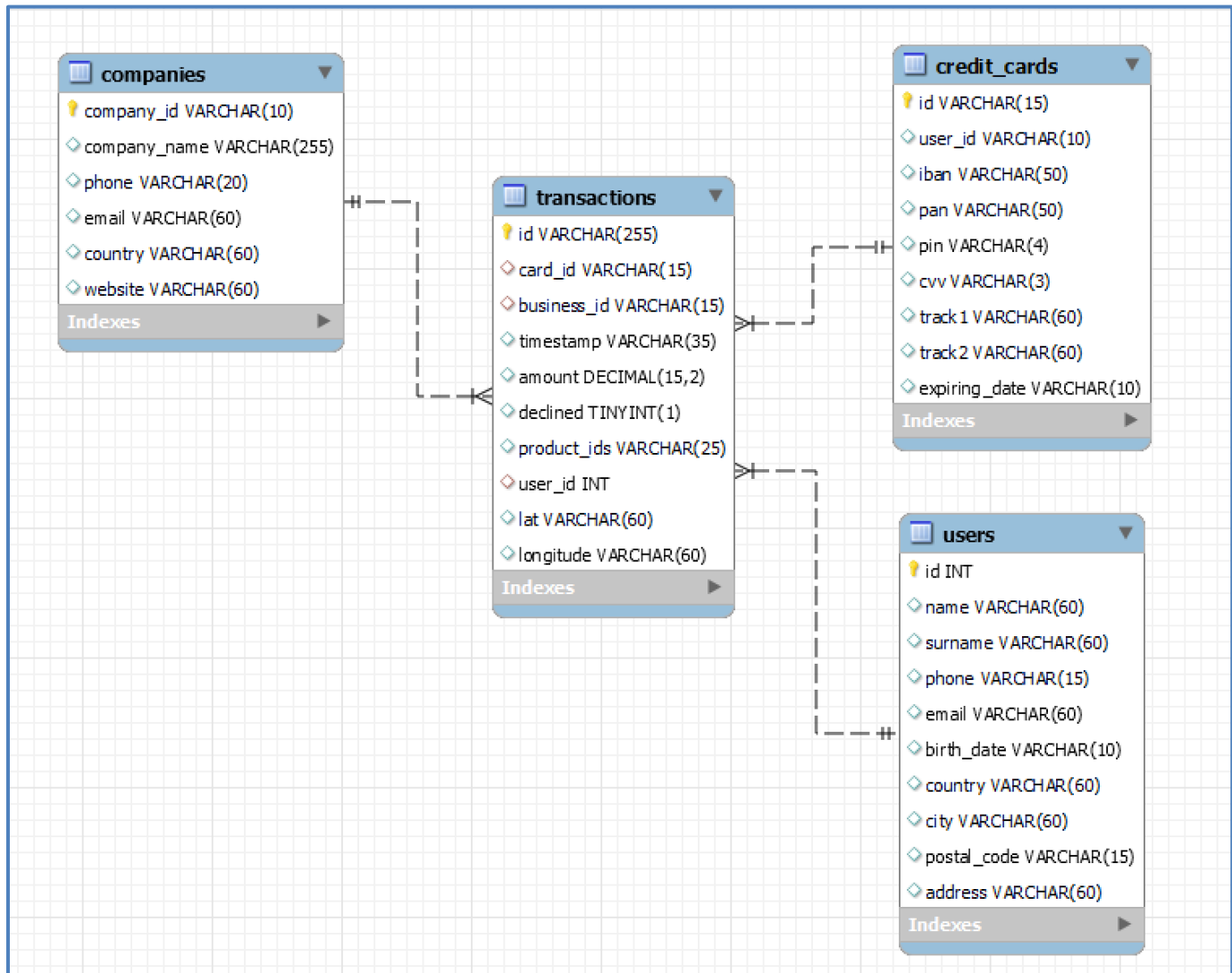


Action Output

#	Time	Action	Message
✓ 484	14:13:10	ALTER TABLE transactions ADD FOREIGN KEY(...	587 row(s) affected Records: 587 Duplicates: 0 Warnings: 0

Dados los pasos anteriores, disponemos de la Base de Datos **transactionsTS4** bien montada y lista para ejecutar las queries solicitadas.

Como resultado, el Diagrama de la BD **transactionsTS4** sería como sigue:



El esquema del diagrama de la BD tendría forma de estrella en donde la tabla **transactions** representaría la tabla de hechos, mientras que las tablas de dimensiones estarían conformadas por **companies**, **credit_cards** y **users**, siendo la relación de 1 a muchos entre estas tablas y la tabla de hechos.

Ejercicio 1.1

Realizar una subconsulta que muestre a todos los usuarios con más de 30 transacciones utilizando al menos 2 tablas.

Solucion:

Necesitaremos datos de las tablas **transactions** y **users**; es por ello que serán las 2 tablas que utilizaremos.

La solución utilizando una **subconsulta**, como solicitado, sería de la siguiente forma:

```

133 • SELECT *, (SELECT COUNT(user_id) FROM transactions
134           WHERE transactions.user_id = users.id ) AS cantidad_trans
135 FROM users
136 GROUP BY users.id
137 HAVING cantidad_trans > 30;
138

```

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

	id	NAME	surname	phone	email	birth_date	country
▶	92	Lynn	Riddle	1-387-885-4057	vitae.aliquet@outlook.edu	Sep 21, 1984	United States
	267	Ocean	Nelson	079-481-2745	aenean@yahoo.com	Dec 26, 1991	Canada
	272	Hedwig	Gilbert	064-204-8788	sem.eget@idoud.edu	Apr 16, 1991	Canada
	275	Kenyon	Hartman	082-871-7248	convallis.ante.lectus@yahoo.com	Aug 3, 1982	Canada

Result 131 x

Output

Action Output ▼

	#	Time	Action	Message
✓	486	14:25:54	SELECT *, (SELECT COUNT(user_id) FROM trans...	4 row(s) returned

La solución alternativa utilizando **JOIN** sería de la siguiente forma:

```

143
144 • SELECT u.name AS nombre, u.surname AS apellido, u.country AS pais, user_id AS ID, count(user_id) AS cantidad_trans
145 FROM transactions t
146 JOIN users u ON t.user_id = u.id
147 GROUP BY user_id
148 HAVING cantidad_trans > 30;

```

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

	nombre	apellido	pais	ID	cantidad_trans
▶	Lynn	Riddle	United States	92	39
	Ocean	Nelson	Canada	267	52
	Hedwig	Gilbert	Canada	272	76
	Kenyon	Hartman	Canada	275	48

Result 135 ×

Output

Action Output

#	Time	Action	Message
✓ 490	14:30:45	SELECT u.name AS nombre, u.surname AS apellido,...	4 row(s) returned

*Para la solución mediante el uso de **JOIN** hemos decidido cargar solo algunos campos de la tabla **users**. Mientras que para la solución con subconsulta hemos mostrado todos los campos de la tabla **users**.

Ejercicio 1.2

Mostrar la media del amount por IBAN de las tarjetas de crédito en la compañía Donec Ltd., utiliza por lo menos 2 tablas.

Solucion:

Para la siguiente query requerimos incluir una tercera tabla que contenga información del **iban**, por lo que en total utilizaremos las tablas: **transactions**, **credit_cards** y **companies**.

La media de los montos de las transacciones por **iban** por parte de la compañía **Donec Ltd** sería de 203.72 ; y se obtendría a través de los comandos mostrados a continuación.


```
160
161 • SELECT co.company_name AS compañía, cc.iban, ROUND(AVG(tr.amount),2) AS media_transacciones
162 FROM transactions AS tr
163 JOIN companies AS co ON co.company_id=tr.business_id
164 JOIN credit_cards AS cc ON cc.id = tr.card_id
165 WHERE company_name = 'Donec Ltd'
166 GROUP BY cc.iban;
167
```

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

	compañía	iban	media_transacciones
▶	Donec Ltd	PT87806228135092429456346	203.72

Result 137 x

Output

Action Output

#	Time	Action	Message
✓ 491	14:30:59	SELECT u.name AS nombre, u.suame AS apellido,...	4 row(s) returned

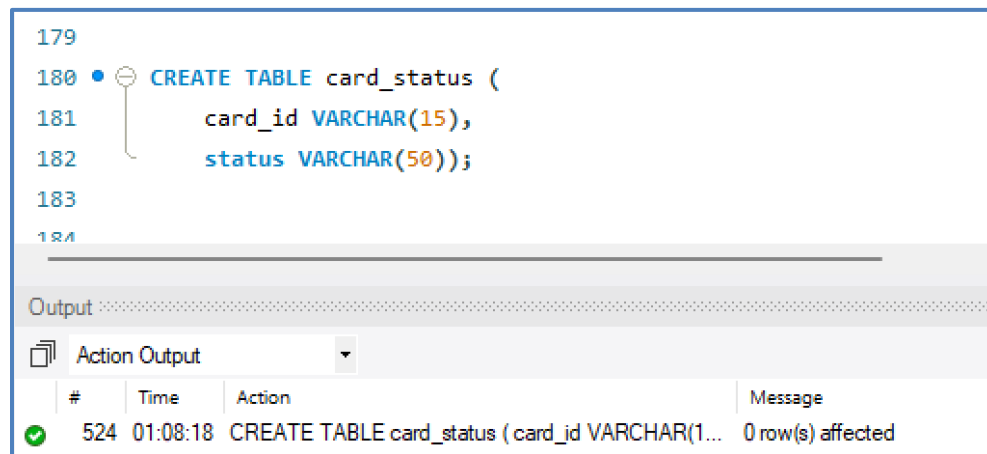
NIVEL 2

Crema una nueva tabla que refleje el estado de las tarjetas de crédito basado en si las últimas 3 transacciones fueron declinadas y genera la siguiente consulta:

¿Cuántas tarjetas están activas?

Solucion:

Paso 1 : se pasa a crear una nueva tabla denominada **card_status**. La tabla contendrá 2 campos utiles para nuestro proposito, los cuales denominados **card_id** y **status**.



```
179
180 • CREATE TABLE card_status (
181     card_id VARCHAR(15),
182     status VARCHAR(50));
183
184
```

Output

Action Output

#	Time	Action	Message
✓ 524	01:08:18	CREATE TABLE card_status (card_id VARCHAR(1...	0 row(s) affected

Paso 2 : Ya creada la tabla, ahora pasamos a rellenarla. Cabe destacar la creación de una tabla temporal (CTE: Common Table Expression) denominada **transacciones_tarjeta2** que contiene como columnas algunas extraídas y generadas de la tabla **transactions**.

Además, con el uso del operador **CASE** y las condiciones que le siguen se calculará el estado de la tarjeta. En donde si la suma de declined en las 3 últimas transacciones es de 2 o menos se considera como "**tarjeta activa**". En cualquier otro caso se considera como "**tarjeta inactiva**".

El filtro para considerar únicamente las 3 últimas transacciones de cada tarjeta sería el siguiente:
WHERE row_transaction <= 3

```

194
195 • INSERT INTO card_status (card_id, status)
196 WITH transacciones_tarjeta2 AS (
197     SELECT card_id,
198         timestamp,
199         declined,
200         ROW_NUMBER() OVER (PARTITION BY card_id ORDER BY timestamp DESC) AS row_transaction
201 FROM transactions)
202 SELECT card_id,
203 CASE
204     WHEN SUM(declined) <= 2 THEN 'tarjeta activa'
205     ELSE 'tarjeta inactiva'
206 END AS estado_tarjeta
207 FROM transacciones_tarjeta2
208 WHERE row_transaction <= 3
209 GROUP BY card_id;
210
211

```

Output

Action Output

#	Time	Action	Message
✓ 526	01:18:32	INSERT INTO card_status (card_id, status) WITH tr...	275 row(s) affected Records: 275 Duplicates: 0 Warnings: 0

Paso 3 : Verificamos la data de la tabla **card_status** recién creada:

```

215
216 • SELECT card_id, status AS estado_tarjeta
217 FROM card_status;
218

```

Result Grid Filter Rows: Export: Wrap Cell Content:

card_id	estado_tarjeta
CcU-2938	tarjeta activa
CcU-2945	tarjeta activa
CcU-2952	tarjeta activa
CcU-2959	tarjeta activa
CcU-2966	tarjeta activa
CcU-2973	tarjeta activa
CcU-2980	tarjeta activa

card_status 161 x

Output

Action Output

#	Time	Action	Message
✓ 537	01:22:23	SELECT card_id, status AS estado_tarjeta FROM ca...	275 row(s) returned

Ejercicio 2.1

¿Cuántas tarjetas están activas?

Solución: Sería un simple conteo de todos los registros de la tabla `card_status` utilizando el

filtro `WHERE status = 'tarjeta activa';`

The screenshot shows a SQL query execution interface. The query is as follows:

```
223 • SELECT COUNT(*) AS 'tarjetas activas'
224 FROM card_status
225 WHERE status = 'tarjeta activa';
226
```

Below the query, the 'Result Grid' is displayed with the following data:

tarjetas activas
275

At the bottom, the 'Action Output' section shows the execution details:

#	Time	Action	Message
✓ 539	01:26:25	SELECT COUNT(*) AS 'tarjetas activas' FROM card...	1 row(s) returned

*** Modificaciones de la última actualización ***

- 1) En la tabla **transactions** ya no se han creado las relaciones con las foreign key a la hora de crear la tabla. En su lugar se ha creado la tabla **transactions** sin las **foreign key**; luego de cargar los datos recién se han establecido las relaciones con las **foreign key** para así evitar el uso de la activación / desactivación de las **Foreign Key**.
- 2) Se ha explicado el paso a paso de cada acción relevante, a diferencia de la versión anterior, en la que se había omitido en gran medida.
- 3) Se ha incluido la resolución del ejercicio del Nivel 2.