SPRINT3-

In this sprint, a business situation is simulated in which you have to perform various manipulations on the database tables. In turn, you will have to work with indexes and views. In this activity, you will continue working with the database that contains information about a company dedicated to selling products online. In this task, you will start working with information related to credit cards.

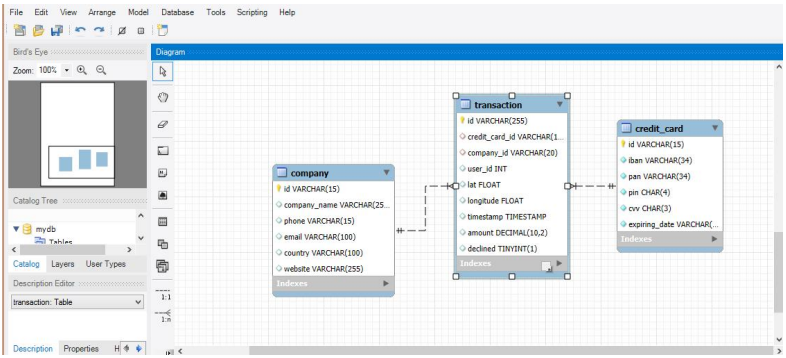
Level 1

- Exercise 1

Your task is to design and create a table called "credit_card" that stores crucial details about credit cards. The new table must be able to uniquely identify each card and establish an appropriate relationship with the other two tables ("transaction" and "company"). After creating the table, you will need to enter the information in the document called "dades_introduir_credit". Remember to show the diagram and make a brief description of it.

MY code:

```
Drop table if exists credit_card;

CREATE TABLE credit_card (

   id VARCHAR(15) PRIMARY KEY,

   iban VARCHAR(34) NOT NULL UNIQUE,

   pin CHAR(4) NOT NULL,

   cvv CHAR(3) NOT NULL,

   expiring_date varchar (8) NOT NULL

);

INSERT INTO credit_card (id, iban, pan, pin, cvv, expiring_date) VALUES (
'CcU-2938', 'TR301950312213576817638661', '5424465566813633', '3257',
'984', '10/30/22'); ....
```



- Exercise 2

The Human Resources department has identified an error in the account number of the user with ID CcU-2938. The information that should be displayed for this record is: R323456312213576817699999. Remember to show that the change was made.

#my code:

<u>##hacer cambio:</u>

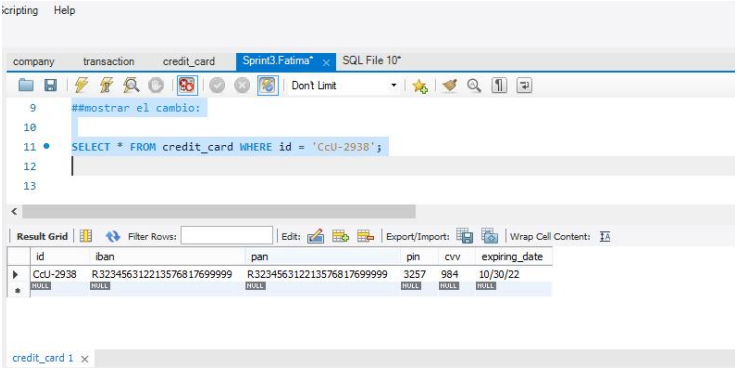
UPDATE credit_card

SET iban = 'R323456312213576817699999'

WHERE id = 'CcU-2938';

<u>##mostrar el cambio:</u>

SELECT * FROM credit_card WHERE id = 'CcU-2938';



- Exercise 3

In the "transaction" table, enter a new user with the following information:

En la taula "transaction" ingressa un nou usuari amb la següent informació:

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

#My code:
#insert data:

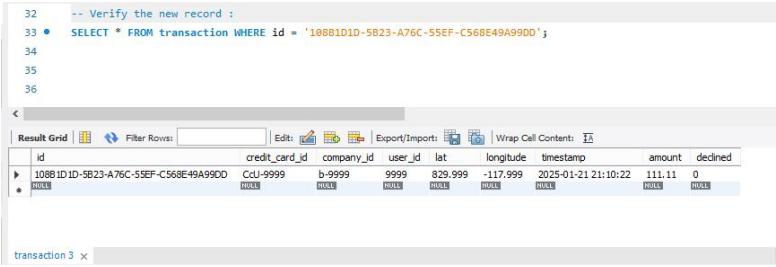
```
SELECT * FROM company WHERE id = 'b-9999';

INSERT IGNORE INTO company (id) VALUES ('b-9999');



SELECT * FROM credit_card WHERE id = 'CcU-9999';

INSERT IGNORE INTO credit_card (id) VALUES ('CcU-9999');
```

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, CURRENT_TIMESTAMP, 111.11, 0);

#-- Verify the new record :

SELECT * FROM transaction WHERE id = '108B1D1D-5B23-A76C-55EF-C568E49A99DD';



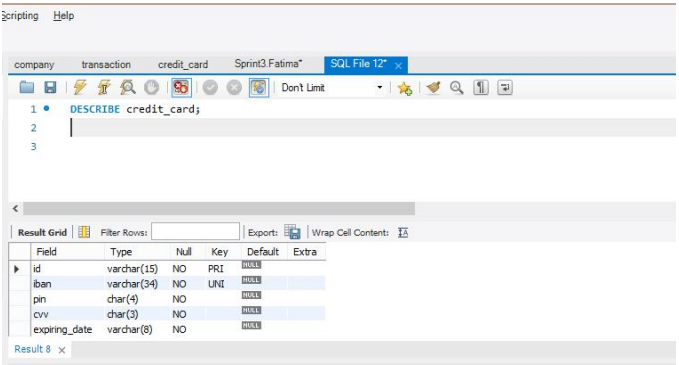
Exercise 4

Human Resources asks you to delete the "pan" column from the credit_*card table. Remember to show the change you made.

ALTER TABLE credit_card

DROP COLUMN pan;

DESCRIBE credit_card;



Level 2

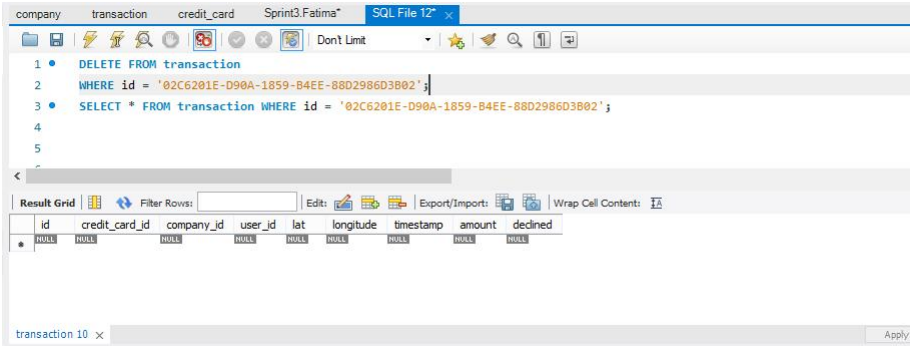
Exercise 1 Delete the record with ID 02C6201E-D90A-1859-B4EE-88D2986D3B02 from the database transaction table.

DELETE FROM transaction

WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02';

#showing code :

SELECT * FROM transaction WHERE id = '02C6201E-D90A-1859-B4EE-88D2986D3B02';

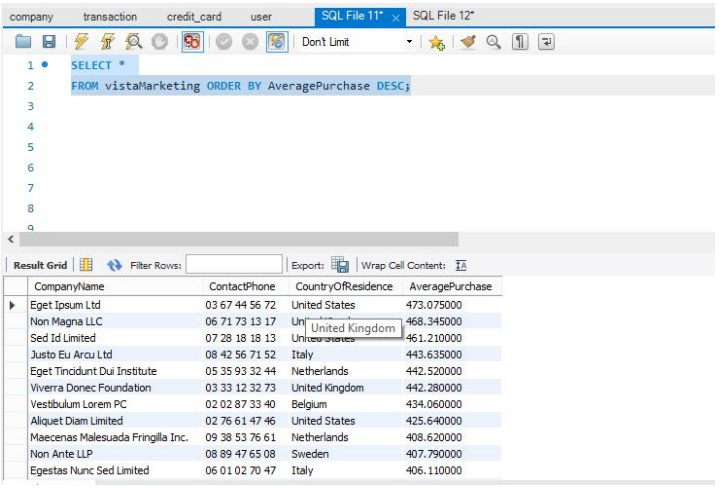


Exercise 2

The marketing department wants to have access to specific information to perform effective analysis and strategies. It has been requested to create a view that provides key details about the companies and their transactions. You will need to create a view called VistaMarketing that contains the following information: Company name. Contact phone number. Country of residence. Average purchase made by each company. Present the view created, sorting the data from highest to lowest average purchase.

#My Code:

```
DROP VIEW IF EXISTS VistaMarketing;

CREATE VIEW  VistaMarketing AS

SELECT c.company_name AS CompanyName, c.phone AS ContactPhone,

   c.country AS CountryOfResidence,

   AVG(t.amount) AS AveragePurchase

FROM company c

JOIN transaction t ON c.id = t.company_id

WHERE  t.declined = 0

GROUP BY c.id, c.company_name, c.phone, c.country;



SELECT *

FROM vistaMarketing ORDER BY AveragePurchase DESC;
```



CompanyName	ContactPhone	CountryOfResidence	AveragePurchase
Eget Ipsum Ltd	03 67 44 56 72	United States	473.075000
Non Magna LLC	06 71 73 13 17	Un	468.345000
Sed Id Limited	07 28 18 18 13	Un	461.210000
Justo Eu Arcu Ltd	08 42 56 71 52	Italy	443.635000
Eget Tincidunt Dui Institute	05 35 93 32 44	Netherlands	442.520000
Viverra Donec Foundation	03 33 12 32 73	United Kingdom	442.280000
Vestibulum Lorem PC	02 02 87 33 40	Belgium	434.060000
Aliquet Diam Limited	02 76 61 47 46	United States	425.640000
Maecenas Malesuada Fringilla Inc.	09 38 53 76 61	Netherlands	408.620000
Non Ante LLP	08 89 47 65 08	Sweden	407.790000
Egestas Nunc Sed Limited	06 01 02 70 47	Italy	406.110000

Exercise 3

Filter the VistaMarketing view to show only the companies that have their country of residence as "Germany"

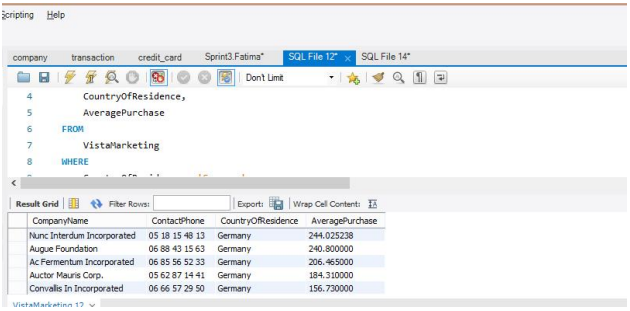
#MY Code:

SELECT CompanyName, ContactPhone, CountryOfResidence, AveragePurchase

FROM VistaMarketing

WHERE CountryOfResidence = 'Germany'

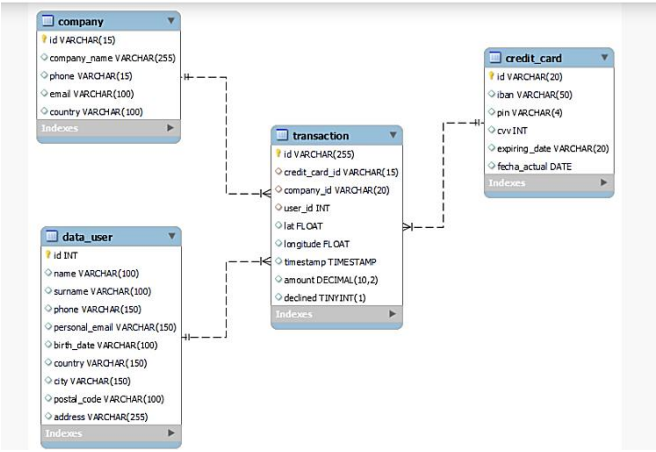
ORDER BY AveragePurchase DESC;



Level 3

Exercise 1: no me sale diagram corecto!

Next week you will have another meeting with the marketing managers. A colleague of yours made modifications to the database, but he does not remember how he did them. He asks you to help him leave the commands executed to obtain the following diagram:



!!! In this activity, you need to describe the "step by step" of the tasks performed. It is important to make descriptions simple, straightforward and easy to understand. To carry out this activity you will have to work with the files called "estructura_dades_user" and "dades_introduir_user"

Exercise 2

#Step 1:

```
SET foreign_key_checks = 0;
```

#Step2:

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

#step 3:

```
CREATE TABLE IF NOT EXISTS user (
    id INT PRIMARY KEY,
    name VARCHAR(100),
    surname VARCHAR(100),
    phone VARCHAR(150),
    email VARCHAR(150),
    birth_date VARCHAR(100),
    country VARCHAR(150),
    city VARCHAR(150),
    postal_code VARCHAR(100),
    address VARCHAR(255),
    FOREIGN KEY(id) REFERENCES transaction(user_id)
  );
```

#Step4:

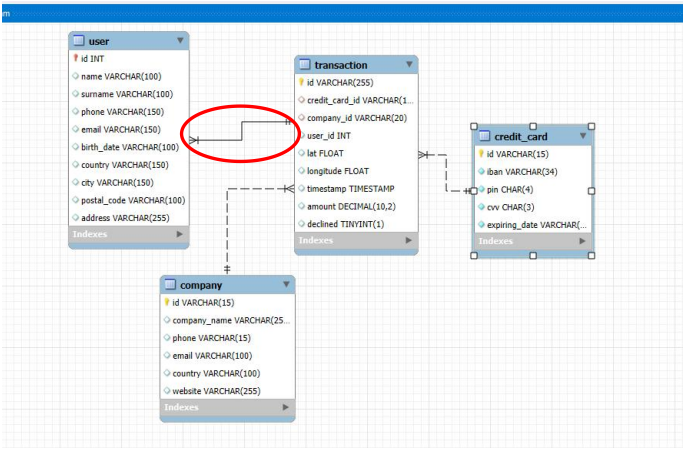
```
-- Insertamos datos de user

INSERT INTO user (id, name, surname, phone, email, birth_date, country, city,
postal_code, address) VALUES (    "1", "Zeus", "Gamble", "1-282-581-0551",
"interdum.enim@protonmail.edu", "Nov 17, 1985",    "United States",
"Lowell", "73544", "348-7818 Sagittis St.");
```

...

#Step5:

SET foreign_key_checks = 1;

#diagram: No sale de forma adecuada, porque da la relación 1:n (transaction:user)



#Solucion1? por el dato que hemos añadido en ejercicio N1-E3:

INSERT INTO user (id, name, surname, phone, email, birth_date, country, city, postal_code, address)

VALUES (9999, "Unknown", "User", "N/A", "unknown@example.com", "N/A", "Unknown", "Unknown", "00000", "Unknown");

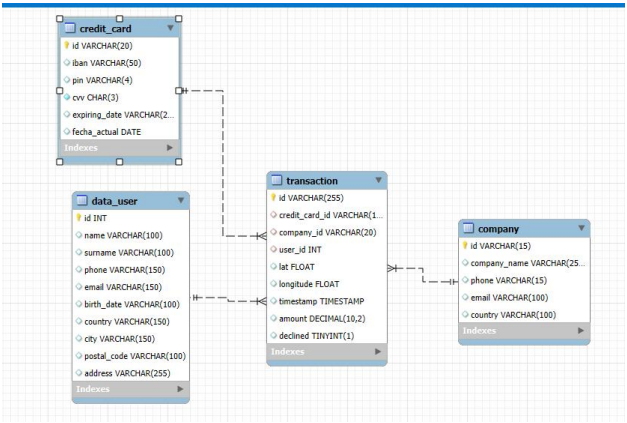
>>>Pero sigue dando lo mismo error!

#Solucion2?

#eliminar index de la tabla de transaction

```
ALTER TABLE transaction

DROP INDEX idx_user_id;

#poner FK correcto!

ALTER TABLE transaction

ADD CONSTRAINT fk_transaction_user

FOREIGN KEY (user_id) REFERENCES user(id);

#ortos cambios de tablas para conseguir como ejercicio:

#1:delete columna de website from table company:

alter table company

drop column website;

#2:add column fecha_actual Date to credit_card and modify data type:

ALTER TABLE credit_card

ADD COLUMN fecha_actual DATE;


ALTER TABLE credit_card

MODIFY COLUMN id VARCHAR(20),

MODIFY COLUMN iban VARCHAR(50),

MODIFY COLUMN pin VARCHAR(4),

MODIFY COLUMN cvv INT,

MODIFY COLUMN expiring_date VARCHAR(20);
```

RENAME TABLE user TO data_user;



Exercise 2

The company also asks you to create a view called "TechnicalReport" that contains the following information:

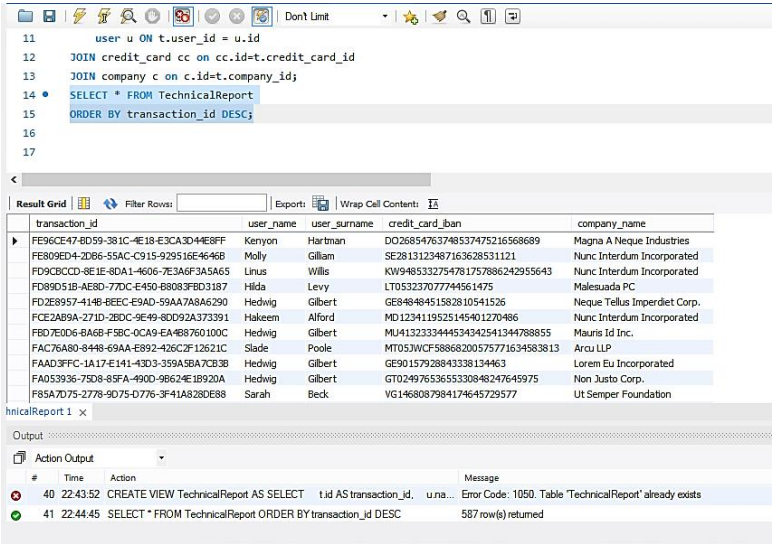
1) Transaction ID
2) User name
3) User last name
4) IBAN of the credit card used.
5) Company name of the transaction.
6) Make sure to include relevant information from both tables and use aliases to rename columns as necessary.

Display the results of the view, sorting the results in descending order based on the transaction ID variable.

CREATE VIEW TechnicalReport AS

SELECT t.id AS transaction_id,

```
    u.name AS user_name,

    u.surname AS user_surname,

    cc.iban AS credit_card_iban,

    c.company_name AS company_name

FROM  transaction t

JOIN user u ON t.user_id = u.id

JOIN credit_card cc on cc.id=t.credit_card_id

JOIN company c on c.id=t.company_id;

#show view:

SELECT * FROM TechnicalReport

ORDER BY transaction_id DESC;
```



```
11          user u ON t.user_id = u.id
12      JOIN credit_card cc on cc.id=t.credit_card_id
13      JOIN company c on c.id=t.company_id;
14 •    SELECT * FROM TechnicalReport
15      ORDER BY transaction_id DESC;
16
17
```

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

transaction_id	user_name	user_surname	credit_card_iban	company_name
FE96CE47-BD59-381C-4E18-E3CA3D44E8FF	Kenyon	Hartman	DO26854763748537475216568689	Magna A Neque Industries
FE809ED4-2DB6-55AC-C915-929516E4646B	Molly	Gilliam	SE2813123487163628531121	Nunc Interdum Incorporated
FD9CBCCD-8E1E-8DA1-4606-7E3A6F3A5A65	Linus	Willis	KW9485332754781757886242955643	Nunc Interdum Incorporated
FD89D51B-AE8D-77DC-E450-B8083FBD3187	Hilda	Levy	LT053237077744561475	Malesuada PC
FD2E8957-414B-BEEC-E9AD-59AA7A8A6290	Hedwig	Gilbert	GE8484845158281054152 6	Neque Tellus Imperdiet Corp.
FCE2AB9A-271D-2BDC-9E49-8DD92A373391	Hakeem	Alford	MD1234119525145401270486	Nunc Interdum Incorporated
FBD7E0D6-8A6B-F5BC-0CA9-EA4B8760100C	Hedwig	Gilbert	MU413233344453434254134478885 5	Mauris Id Inc.
FAC76A80-8448-69AA-E892-426C2F12621C	Slade	Poole	MT05JWCF5886820057577163458381 3	Arcu LLP
FAAD3FFC-1A17-E141-43D3-359A5BA7CB3B	Hedwig	Gilbert	GE9015792884333813446 3	Lorem Eu Incorporated
FA053936-75D8-85FA-490D-9B624E1B920A	Hedwig	Gilbert	GT0249765365533084824764597 5	Non Justo Corp.
F85A7D75-2778-9D75-D776-3F41A828DE88	Sarah	Beck	VG1468087984174645729577	Ut Semper Foundation

hnicalReport 1

Output

Action Output

#	Time	Action	Message
40	22:43:52	CREATE VIEW TechnicalReport AS SELECT t.id AS transaction_id, u.na...	Error Code: 1050. Table 'TechnicalReport' already exists
41	22:44:45	SELECT * FROM TechnicalReport ORDER BY transaction_id DESC	587 row(s) returned