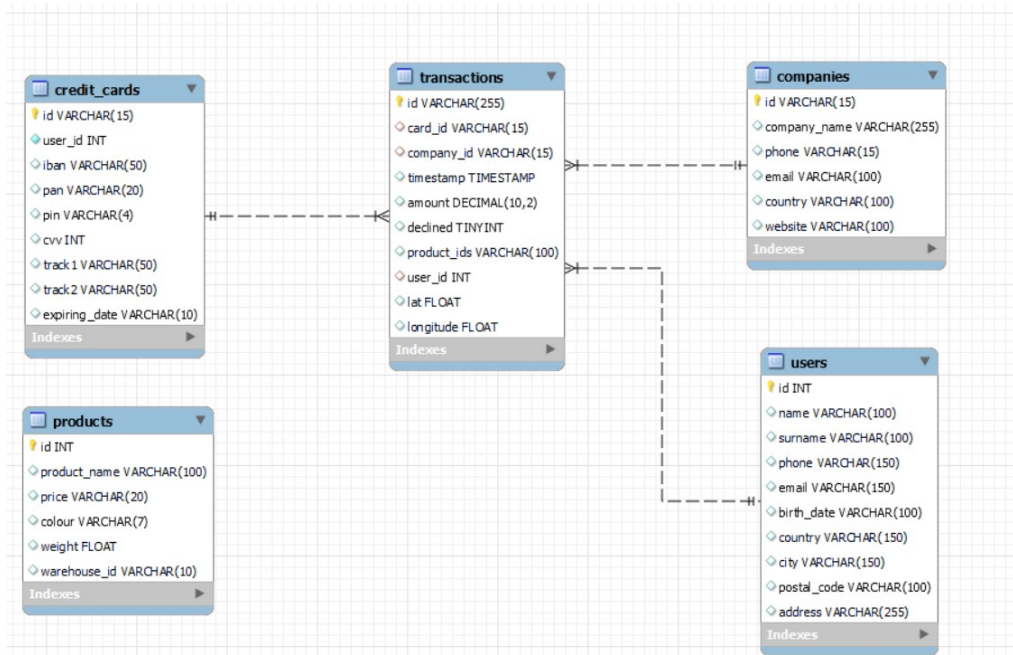


Tasca S4.01. Creació de Base de Dades

Nivell 1

Descàrrega els arxius CSV, estudia'ls i dissenya una base de dades amb un esquema d'estrella que contingui, almenys 4 taules de les quals puguis realitzar les següents consultes:



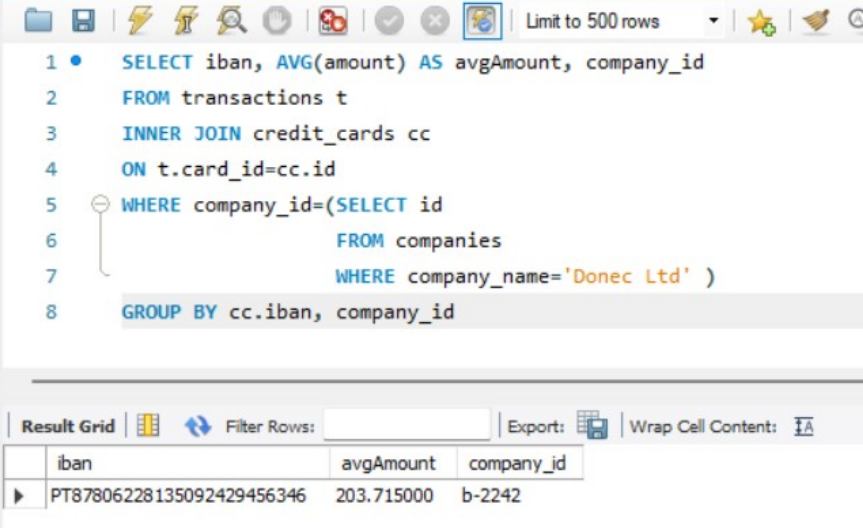
Exercici 1

Realitza una subconsulta que mostri tots els usuaris amb més de 30 transaccions utilitzant almenys 2 taules.

```
1 SELECT id, name, surname, (  
2     SELECT COUNT(*) as trans  
3     FROM transactions t2  
4     WHERE user_id=u.id ) AS numTrans  
5 FROM users u  
6 HAVING numTrans >30
```

Exercici 2

Mostra la mitjana de la suma de transaccions per IBAN de les targetes de crèdit en la companyia Donec Ltd. utilitzant almenys 2 taules.



```
1 • SELECT iban, AVG(amount) AS avgAmount, company_id
2 FROM transactions t
3 INNER JOIN credit_cards cc
4 ON t.card_id=cc.id
5 WHERE company_id=(SELECT id
6 FROM companies
7 WHERE company_name='Donec Ltd' )
8 GROUP BY cc.iban, company_id
```

Result Grid

iban	avgAmount	company_id
PT87806228135092429456346	203.715000	b-2242

Nivell 2

Crea una nova taula que reflecteixi l'estat de les targetes de crèdit basat en si les últimes tres transaccions van ser declinades i genera la següent consulta:

0.1) Creació de la taula

```
1 CREATE TABLE `active_cards` (  
2   `card_id` VARCHAR(15) NOT NULL,  
3   `declined` TINYINT DEFAULT NULL,  
4   PRIMARY KEY (`card_id`),  
5   CONSTRAINT `active_cards_ibfk_1` FOREIGN KEY (`card_id`) REFERENCES `credit_cards` (`id`)  
6 ) ;
```

0.2) Funció per calcular si ha estat declinada en les últimes tres transaccions

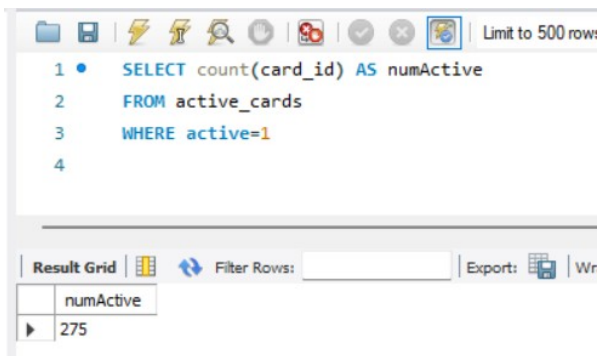
```
1 DELIMITER //  
2  
3 CREATE FUNCTION LastDeclined (idCard VARCHAR(15) )  
4 RETURNS INT  
5 READS SQL DATA  
6  
7 BEGIN  
8  
9   DECLARE SumDeclined INT;  
10  
11   SELECT SUM(declined) INTO SumDeclined FROM (  
12     SELECT t.card_id,timestamp,declined from transactions t  
13     WHERE t.card_id=idCard  
14     ORDER BY t.card_id,timestamp DESC  
15     LIMIT 3) s;  
16  
17   RETURN SumDeclined;  
18  
19 END; //  
20  
21 DELIMITER ;  
22
```

0.3) Inserció de les dades a la nova taula

```
1  
2 INSERT INTO active_cards  
3 (  
4   SELECT DISTINCT card_id, IF (lastDeclined(card_id)=3,0,1) AS active  
5   FROM transactions  
6 )
```

Exercici 1

Quantes targetes estan actives?



Hi ha 275 targetes actives

Nivell 3

Crea una taula amb la qual puguem unir les dades del nou arxiu products.csv amb la base de dades creada, tenint en compte que des de transaction tens product_ids. Genera la següent consulta:

0.1) Creació de la taula

```
1 CREATE TABLE `product_transac` (  
2     `id` int NOT NULL AUTO_INCREMENT,  
3     `product_id` int NOT NULL,  
4     `transaction_id` VARCHAR(255) DEFAULT NULL,  
5     PRIMARY KEY (`id`),  
6     CONSTRAINT `product_transac_ibfk_1` FOREIGN KEY (`product_id`) REFERENCES `products` (`id`),  
7     CONSTRAINT `product_transac_ibfk_2` FOREIGN KEY (`transaction_id`) REFERENCES `transactions` (`id`)  
8 ) ;
```

0.2) Creació d'un stored procedure per insertar els registres

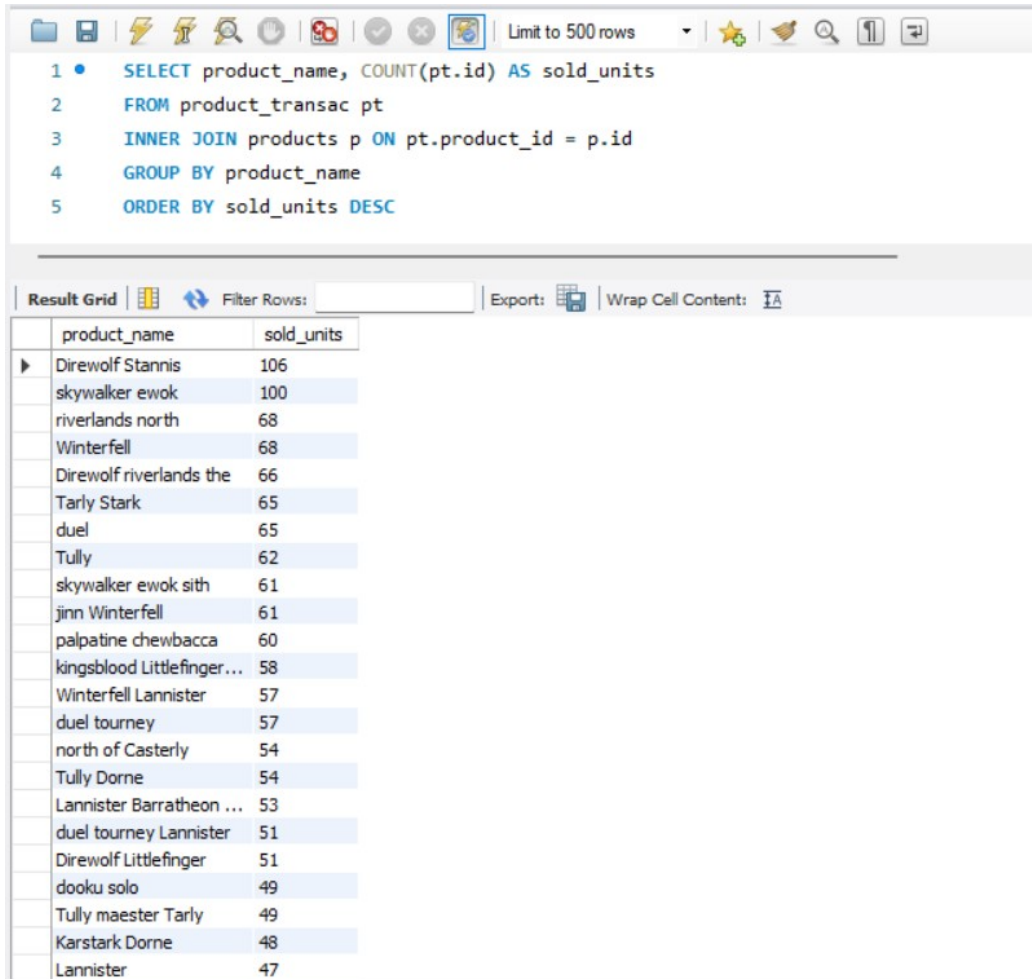
```
1 DELIMITER //  
2 CREATE PROCEDURE sp_Insert_In_Product_Trans()  
3 BEGIN  
4     DECLARE sId varchar(150);  
5     DECLARE sProduct_ids varchar(100);  
6     DECLARE inumComas INT;  
7     DECLARE i INT;  
8     DECLARE var_final INT DEFAULT 0;  
9     DECLARE cur1 CURSOR FOR SELECT id,product_ids, ((LENGTH(product_ids)) - LENGTH(REPLACE(product_ids, ','))) AS numComas  
10    FROM transactions  
11    WHERE id NOT IN (SELECT transaction_id  
12    FROM product_transac);  
13    DECLARE CONTINUE HANDLER FOR NOT FOUND SET var_final = 1;  
14  
15    OPEN cur1;  
16    read_loop: LOOP  
17    FETCH cur1 INTO sId, sProduct_ids, inumComas;  
18    IF var_final = 1 THEN  
19        LEAVE read_loop;  
20    END IF;  
21    SET i=0;  
22    getIdProd: LOOP  
23        INSERT INTO product_transac (transaction_id,product_id) VALUES (sId, SUBSTRING_INDEX(SUBSTRING_INDEX(sProduct_ids, ',', i+1),  
24        SET i = i +1;  
25        IF i > inumComas THEN  
26            LEAVE getIdProd;  
27        END IF;  
28    END LOOP getIdProd;  
29  
30    END LOOP read_loop;  
31    CLOSE cur1;  
32  
33 END //  
34  
35 DELIMITER ;  
36
```

0.3) Execució del stored procedure

```
SQL File 19*  SQL File 20*  SQL File 21*  SQL File 8*  SQL File  
1 CALL `sp_Insert_In_Product_Trans`();
```

Exercici 1

Necessitem conèixer el nombre de vegades que s'ha venut cada producte.



The screenshot shows a SQL query editor with a toolbar at the top. The query is as follows:

```
1 • SELECT product_name, COUNT(pt.id) AS sold_units
2 FROM product_transac pt
3 INNER JOIN products p ON pt.product_id = p.id
4 GROUP BY product_name
5 ORDER BY sold_units DESC
```

Below the query editor, there is a 'Result Grid' section with a 'Filter Rows' input and an 'Export' button. The result grid displays the following data:

product_name	sold_units
Direwolf Stannis	106
skywalker ewok	100
riverlands north	68
Winterfell	68
Direwolf riverlands the	66
Tarly Stark	65
duel	65
Tully	62
skywalker ewok sith	61
jinn Winterfell	61
palpatine chewbacca	60
kingsblood Littlefinger...	58
Winterfell Lannister	57
duel tourney	57
north of Casterly	54
Tully Dorne	54
Lannister Barratheon ...	53
duel tourney Lannister	51
Direwolf Littlefinger	51
dooku solo	49
Tully maester Tarly	49
Karstark Dorne	48
Lannister	47

Annexe

Diagrama final de la base de dades

