BS Degree in Informatics Engineering
Academic year: 2023/24 - 2nd year, 2nd term
Subject: File Structures and Databases
Second Assignment's Report: DB development and Querying

uc3m | Universidad Carlos III de Madrid

Lecturer:	**FRANCISCO JAVIER CALLE GOMEZ**		
Group:	**89**	**Lab User**	**fsdb279**
Student:	**Eduardo Alarcón Navarro**	**NIA:**	**100472175**
Student:	**Salvador Ayala Iglesias**	**NIA:**	**100495832**
Student:	**Ines Guillén Peña**	**NIA:**	**100495752**

1 Introduction

This document is the result of the work made by the team mentioned previously on the second assignment of the course on File System and Data Bases of the University Carlos III of Madrid. After reading the statement carefully and figuring out the general structure of the project and the tasks to solve, we concluded that initiating with query design would be most prudent, given its direct correlation with earlier stages. We then worked on implementing the queries, procedures, views and triggers as needed to fulfill the project requirements.
The goal of the project was to have all the structures of the statement well tested and verified, for which a rigorous design, development, and validation stage were necessary.

2 Queries

In this section, queries realized on the database will be described.
To accomplish this, the different database queries will be divided into the following sections: Relational algebra design, SQL implementation, divided into subsections to provide a better understanding, and lastly the tests carried out on the queries.

2.1 First query (Bestsellers Geographic Report)

Relational algebra

$$A \equiv \pi_{\text{username, product as product\_name, price, varietal, orderdate, country, town, (int) quantity, barCode}} \sigma_{\text{EXTRACT(YEAR FROM orderdate) = EXTRACT(YEAR FROM SYSDATE)-1}} ((l \equiv \text{lines\_anonym}) \theta_{(l.\text{barcode} = r.\text{barcode})} (r \equiv \text{References}) \theta_{r.\text{product}=p.\text{product}} (p \equiv \text{Products}))$$

$$C \equiv \pi_{\text{username, product as product\_name, price, varietal, orderdate, country, town, (int) quantity, barCode}} \sigma_{\text{EXTRACT(YEAR FROM orderdate) = EXTRACT(YEAR FROM SYSDATE)-1}} ((l \equiv \text{lines\_anonym}) \theta_{(l.\text{barcode} = r.\text{barcode})} (r \equiv \text{References}) \theta_{r.\text{product}=p.\text{product}} (p \equiv \text{Products}))$$

$$S \equiv \text{Anonym\_Data} \cup \text{Client\_Data}$$

$T \equiv \pi_{varietal,\ sum(quantity)}$ Group by $_{varietal}$ (Sales_Data)

$R \equiv \pi_{country,\ varietal,\ count(username),\ sum(quantity)\ as\ total\_units\_sold,\ count(product\_name),\ sum(quantity)/count(product\_name)\ as\ avg\_units\_per\_reference,\ row\_number,\ sum(price*quantity)\ as\ total\_revenue}$ Group by $_{varieta,\ countryl}$ (Sales_Data)

result $\leftarrow \pi_{country,\ varietal,\ candidate,\ num\_buyers,\ total\_units\_sold,\ avg\_units\_per\_reference,\ totl\_revenue}\ \sigma_{rank=1}$ order by $_{country}$ $(R\ \theta_{r.varietal=t.varietal}\ T)$

SQL Code

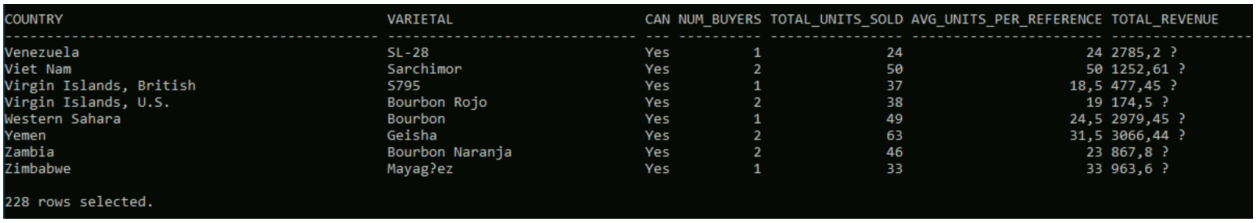
```
Unset
-- Data from each sale from both registered and anonymous clients.
WITH Anonym_Data AS (
    SELECT
        l.contact AS username,
        r.product AS product_name,
        r.price,
        p.varietal,
        l.orderdate,
        TRIM(l.dliv_country) AS country,
        l.dliv_town AS town,
        CAST(l.quantity AS INTEGER) AS quantity,
        l.barCode
    FROM
        Lines_Anonym l
    JOIN
        References r ON l.barCode = r.barCode
    JOIN
        Products p ON r.product = p.product
    WHERE
        EXTRACT(YEAR FROM orderdate) = EXTRACT(YEAR FROM SYSDATE)-1
),
-- Data from the users who registered to buy the products.
Client_Data AS (
    SELECT DISTINCT
        l.username,
        r.product AS product_name,
        r.price,
        p.varietal,
        l.orderdate,
        TRIM(l.country) AS country,
        l.town,
```

```
        CAST(l.quantity AS INTEGER) AS quantity,
        l.barCode AS barCode
    FROM
        Client_Lines l
    JOIN
        References r ON l.barCode = r.barCode
    JOIN
        Products p ON r.product = p.product
    WHERE
        EXTRACT(YEAR FROM orderdate) = EXTRACT(YEAR FROM SYSDATE)-1
),
-- Combination the data from the two sources.
SalesData AS (
    SELECT * FROM Anonym_Data
    UNION ALL
    SELECT * FROM Client_Data
),
-- Data about each varietal.
RankedVarietals AS (
    SELECT
        country,
        varietal,
        COUNT(DISTINCT username) AS num_buyers,
        SUM(quantity) AS total_units_sold,
        COUNT(DISTINCT product_name) AS num_references,
                        SUM(quantity)   /   COUNT(DISTINCT   product_name)   AS
avg_units_per_reference,
         ROW_NUMBER() OVER (PARTITION BY country ORDER BY SUM(quantity) DESC)
AS rank,
         CONCAT(SUM(price * quantity), ' €') AS total_revenue -- The € is not
displayed propperly.
    FROM
        SalesData
    GROUP BY
        country, varietal
),
-- Total amount of units sold per varietal.
TotalAmountPerVarietal AS (
    Select
        varietal,
        SUM(quantity) AS total_units_sold
```

```
    FROM
        SalesData
    GROUP BY
        varietal
)
-- The final query.
SELECT
    rv.country AS country,
    rv.varietal AS varietal,
    CASE
        WHEN rv.total_units_sold > 0.01 * tav.total_units_sold THEN 'Yes'
        ELSE 'No'
    END AS Candidate,
    rv.num_buyers AS num_buyers,
    rv.total_units_sold AS units_sold,
    rv.avg_units_per_reference AS avg_units_per_reference,
    TRIM(rv.total_revenue) AS total_revenue
FROM
    RankedVarietals rv
JOIN
    TotalAmountPerVarietal tav ON rv.varietal = tav.varietal
WHERE
    rank = 1
ORDER BY
    country;
```

The idea from this code is to select the best selling product for each country, by the number of people who have bought it in that country. As we have orders from both registered and unregistered clients, we first have to join both tables, but as the column names do not match, we have to do a double select and make a union of both datasets (the registered clients and anonymous). Once we have that information consolidated, we create a ranking for each varietal for each country and varietal. Meaning, the temporary table RankedVarietal has a row for each combination of country and varietal and the number of buyers for that specific country and varietal. We also had to obtain the total quantity of produce sold for each varietal, that is obtained in the TotalAmountPerVarietal section. Lastly, we join the information, ordering them by country and only selecting the ones that have rank = 1, which was assigned in the RankedVarietals by selecting the row number ordering them by the quantity (This will be explained in more depth in the annex).

If we run the code, we obtain the following results (the last lines only):



COUNTRY	VARIETAL	CAN	NUM_BUYERS	TOTAL_UNITS_SOLD	AVG_UNITS_PER_REFERENCE	TOTAL_REVENUE
Venezuela	SL-28	Yes	1	24	24	2785,2 ?
Viet Nam	Sarchimor	Yes	2	50	50	1252,61 ?
Virgin Islands, British	S795	Yes	1	37	18,5	477,45 ?
Virgin Islands, U.S.	Bourbon Rojo	Yes	2	38	19	174,5 ?
Western Sahara	Bourbon	Yes	1	49	24,5	2979,45 ?
Yemen	Geisha	Yes	2	63	31,5	3066,44 ?
Zambia	Bourbon Naranja	Yes	2	46	23	867,8 ?
Zimbabwe	Mayag?ez	Yes	1	33	33	963,6 ?

228 rows selected.

As we can see, all the data requested is there.

Tests

The first test we will be performing is the **number of buyers (num_buyers)**. This can be accomplished by running the search manually. We will select the country "Virgin Islands, British" as a random country to test this on.

```
Unset
-- Number of buyers of the registered users
SELECT count(distinct username)
FROM client_lines cl
JOIN References r on cl.barcode = r.barcode
JOIN Products p on r.product = p.product
WHERE
       EXTRACT(YEAR FROM orderdate) = 2023 AND
       TRIM(cl.country) = 'Virgin Islands, British' AND
       p.varietal = 'S795';

-- Number of anonymous buyers, identified by contact
SELECT count(distinct contact)
FROM lines_anonym cl
JOIN References r on cl.barcode = r.barcode
JOIN Products p on r.product = p.product
WHERE
       EXTRACT(YEAR FROM orderdate) = 2023 AND
       TRIM(cl.dliv_country) = 'Virgin Islands, British' AND
       p.varietal = 'S795';
```

```
SQL> SELECT count(distinct username)
  2  FROM client_lines cl
  3  JOIN References r on cl.barcode = r.barcode
  4  JOIN Products p on r.product = p.product
  5  WHERE
  6  EXTRACT(YEAR FROM orderdate) = 2023 AND
  7  TRIM(cl.country) = 'Virgin Islands, British' AND
  8  p.varietal = 'S795';

COUNT(DISTINCTUSERNAME)
----------------------
                     1

SQL>
SQL>
SQL> SELECT count(distinct contact)
  2  FROM lines_anonym cl
  3  JOIN References r on cl.barcode = r.barcode
  4  JOIN Products p on r.product = p.product
  5  WHERE
  6  EXTRACT(YEAR FROM orderdate) = 2023 AND
  7  TRIM(cl.dliv_country) = 'Virgin Islands, British' AND
  8  p.varietal = 'S795';

COUNT(DISTINCTCONTACT)
----------------------
                     0
```

We will continue testing if this query reflects reality for the **Units_Sold**. We will do this by computing the query for "Western Sahara" manually.

So, to check how many units have been sold, we can use these two queries:
This first one is used to check the quantity of product sold of one specific variety to one specific country. This is used as the query gives this a result, so we do the inverse search. We sum all the quantities that meet these requirements.

```
Unset
select sum(cl.quantity)
from client_lines cl, references r, products p
where cl.barcode = r.barcode AND
       r.product = p.product AND
       TRIm(cl.country) = 'Western Sahara' AND
       EXTRACT(YEAR FROM orderdate) = 2023 AND
       p.varietal = 'Bourbon';
```

But this is only useful for the purchases made by the registered clients. The anonymous clients' purchases have not been accounted for yet. To accomplish that, we have to use this other query:

```
Unset
select sum(cl.quantity)
from lines_anonym cl, references r, products p
where cl.barcode = r.barcode AND
      r.product = p.product AND
      TRIM(cl.dliv_country) = 'Western Sahara' AND
      EXTRACT(YEAR FROM orderdate) = 2023 AND
      p.varietal = 'Bourbon';
```

Which does the same selection as the last one, but for the anonymous purchases.
The result of this two queries is:

```
SQL> select sum(cl.quantity)
  2  from client_lines cl, references r, products p
  3  where cl.barcode = r.barcode AND
  4  r.product = p.product AND
  5  TRIm(cl.country) = 'Western Sahara' AND
  6  EXTRACT(YEAR FROM orderdate) = 2023 AND
  7  p.varietal = 'Bourbon';

SUM(CL.QUANTITY)
----------------
              49

SQL> select sum(cl.quantity)
  2  from lines_anonym cl, references r, products p
  3  where cl.barcode = r.barcode AND
  4  r.product = p.product AND
  5  TRIM(cl.dliv_country) = 'Western Sahara' AND
  6  EXTRACT(YEAR FROM orderdate) = 2023 AND
  7  p.varietal = 'Bourbon';

SUM(CL.QUANTITY)
----------------
```

As we can observe, the second query returns no value, because it has selected no rows, thus, the sum of nothing is nothing. But the first one does return a value, and if we check on the query, the amount for this specific country is 49.

We will continue the testing by evaluating the column average units per reference. To check this, as the previous checks, we will select a random country, in this case "Venezuela" and "Turkmenistan" to prove this column is correct.
First, Venezuela, which has only one reference of the varietal "SL-28" because units_sold/avg_units_per_reference = 24 / 24 = 1, thus, the sum of the registered + anonymous references should be 1:

```
Unset
-- Registered users section
SELECT count(distinct cl.barcode)
FROM client_lines cl
JOIN References r on cl.barcode = r.barcode
JOIN Products p on r.product = p.product
WHERE
       EXTRACT(YEAR FROM orderdate) = 2023 AND
       TRIM(cl.country) = 'Venezuela' AND
       p.varietal = 'SL-28';


-- Anonymous section
SELECT count(distinct cl.barcode)
FROM lines_anonym cl
JOIN References r on cl.barcode = r.barcode
JOIN Products p on r.product = p.product
WHERE
       EXTRACT(YEAR FROM orderdate) = 2023 AND
       TRIM(cl.dliv_country) = 'Venezuela' AND
       p.varietal = 'SL-28';
```

```
SQL> SELECT count(distinct cl.barcode)
  2  FROM client_lines cl
  3  JOIN References r on cl.barcode = r.barcode
  4  JOIN Products p on r.product = p.product
  5  WHERE
  6  EXTRACT(YEAR FROM orderdate) = 2023 AND
  7  TRIM(cl.country) = 'Venezuela' AND
  8  p.varietal = 'SL-28';

COUNT(DISTINCTCL.BARCODE)
-------------------------
                        1

SQL> SELECT count(distinct cl.barcode)
  2  FROM lines_anonym cl
  3  JOIN References r on cl.barcode = r.barcode
  4  JOIN Products p on r.product = p.product
  5  WHERE
  6  EXTRACT(YEAR FROM orderdate) = 2023 AND
  7  TRIM(cl.dliv_country) = 'Venezuela' AND
  8  p.varietal = 'SL-28';

COUNT(DISTINCTCL.BARCODE)
-------------------------
                        0
```

Now, let's check Turkmenistan, which has 37 units sold and an average per reference of 18.5, thus the distinct number of references must be 2, across the two tables:

```
Unset
-- Registered users section
SELECT count(distinct cl.barcode)
FROM client_lines cl
JOIN References r on cl.barcode = r.barcode
JOIN Products p on r.product = p.product
WHERE
       EXTRACT(YEAR FROM orderdate) = 2023 AND
       TRIM(cl.country) = 'Turkmenistan' AND
       p.varietal = 'Bourbon';
-- Anonymous section
SELECT count(distinct cl.barcode)
FROM lines_anonym cl
JOIN References r on cl.barcode = r.barcode
JOIN Products p on r.product = p.product
WHERE
       EXTRACT(YEAR FROM orderdate) = 2023 AND
       TRIM(cl.dliv_country) = 'Turkmenistan' AND
       p.varietal = 'Bourbon';
```

```
SQL> SELECT count(distinct cl.barcode)
  2  FROM client_lines cl
  3  JOIN References r on cl.barcode = r.barcode
  4  JOIN Products p on r.product = p.product
  5  WHERE
  6  EXTRACT(YEAR FROM orderdate) = 2023 AND
  7  TRIM(cl.country) = 'Turkmenistan' AND
  8  p.varietal = 'Bourbon';

COUNT(DISTINCTCL.BARCODE)
-------------------------
                        2

SQL> SELECT count(distinct cl.barcode)
  2  FROM lines_anonym cl
  3  JOIN References r on cl.barcode = r.barcode
  4  JOIN Products p on r.product = p.product
  5  WHERE
  6  EXTRACT(YEAR FROM orderdate) = 2023 AND
  7  TRIM(cl.dliv_country) = 'Turkmenistan' AND
  8  p.varietal = 'Bourbon';

COUNT(DISTINCTCL.BARCODE)
-------------------------
                        0
```

Lastly, we will check the total_revenue column by again, selecting a country: "Finland". We can modify the previous query to obtain one that will calculate the quantity times the price of the references from that country and that varietal which should return 12.4:

```
Unset
-- Registered users section
SELECT sum(cl.price * cl.quantity)
FROM client_lines cl
JOIN References r on cl.barcode = r.barcode
JOIN Products p on r.product = p.product
WHERE
       EXTRACT(YEAR FROM orderdate) = 2023 AND
       TRIM(cl.country) = 'Finland' AND
       p.varietal = 'Bourbon';
-- Anonymous section
SELECT sum(cl.price * cl.quantity)
FROM lines_anonym cl
JOIN References r on cl.barcode = r.barcode
JOIN Products p on r.product = p.product
WHERE
       EXTRACT(YEAR FROM orderdate) = 2023 AND
       TRIM(cl.dliv_country) = 'Finland' AND
       p.varietal = 'Bourbon';
```

```
SQL> SELECT sum(cl.price * cl.quantity)
  2  FROM client_lines cl
  3  JOIN References r on cl.barcode = r.barcode
  4  JOIN Products p on r.product = p.product
  5  WHERE
  6  EXTRACT(YEAR FROM orderdate) = 2023 AND
  7  TRIM(cl.country) = 'Finland' AND
  8  p.varietal = 'Bourbon';

SUM(CL.PRICE*CL.QUANTITY)
-------------------------
                    12,4

SQL> SELECT sum(cl.price * cl.quantity)
  2  FROM lines_anonym cl
  3  JOIN References r on cl.barcode = r.barcode
  4  JOIN Products p on r.product = p.product
  5  WHERE
  6  EXTRACT(YEAR FROM orderdate) = 2023 AND
  7  TRIM(cl.dliv_country) = 'Finland' AND
  8  p.varietal = 'Bourbon';

SUM(CL.PRICE*CL.QUANTITY)
-------------------------
```

2.2 Second query (Business way of life)

Relational algebra

$A \equiv \pi_{\text{orderdate as month, contact as username, product as product_name, price, barCode}} \sigma_{\text{SYSDATE - INTERVAL '12' MONTH >= orderdate}} ((l \equiv \text{lines\_anonym}) \theta_{(l.barcode = r.barcode)} (r \equiv \text{References}))$

$R \equiv \pi_{\text{orderdate as month, username, product as product_name, price, barCode}} \sigma_{\text{SYSDATE - INTERVAL '12' MONTH >= orderdate}} ((c \equiv \text{client\_lines}) \theta_{(c.barcode = r.barcode)} (r \equiv \text{References}))$

$S \equiv A \cup R$

$Av \equiv \pi_{\text{barcode, avg(cost) as avg\_cost}} \text{Group by}_{\text{barcode}} (S)$

$RD \equiv \pi_{\text{month, product\_name, price, sum(quantity) as total\_quantity, barcode, sum(quantity * price) as total\_income, row\_number}} \text{Group by}_{\text{month, product\_name, price, barcode}} (\text{Sales\_Data})$

$\text{result} \leftarrow \pi_{\text{m barcode as best\_sold\_reference, r.units as units\_bought, total\_quantity as units\_sold, total\_income, total\_income-(avg\_cost*total\_quantity) as benefits}} \sigma_{\text{rank=1}} \text{order by}_{\text{country desc}} (RD \theta_{rd.barcode = av..barcode} Av) \rtimes_{rd.barcode=r.barcode} (R \equiv \text{Replacements})$

SQL Code

```
Unset
-- Data FROM each sale FROM both registered and anonymous clients.
WITH Anonymous AS (
    SELECT
        TO_CHAR(l.orderdate,'Month') AS MONTH,
        l.contact AS username,
        r.product AS product_name,
        r.price,
        CAST(l.quantity AS INTEGER) AS quantity,
        l.barCode
    FROM
        Lines_Anonym l
    JOIN
      References r ON l.barCode = r.barCode
    WHERE
      l.orderdate BETWEEN add_months(sysdate, -12) and add_months(sysdate, -1)
),


Registered AS (
    SELECT DISTINCT
        TO_CHAR(l.orderdate,'Month') AS MONTH,
        l.username as username,
        r.product AS product_name,
        r.price,
        CAST(l.quantity AS INTEGER) AS quantity,
        l.barCode as barCode
    FROM
        Client_Lines l
    JOIN
        References r ON l.barCode = r.barCode
    WHERE
      l.orderdate BETWEEN add_months(sysdate, -12) and add_months(sysdate, -1)
),
SalesData AS (
    SELECT * FROM Registered
    UNION ALL (select * FROM Anonymous)
),
-- Average cost per barcode.
AverageCostPerBarcode AS (
    SELECT
```

```
        barCode,
        AVG(cost) AS avg_cost
    FROM
        Supply_Lines
    GROUP BY
        barCode
),
Ranked_Data AS (
    SELECT
        MONTH,
        product_name,
        price,
        SUM(quantity) as total_quantity,
        SalesData.barCode,
        SUM(price * quantity) AS total_income,
    ROW_NUMBER() OVER (PARTITION BY MONTH ORDER BY SUM(quantity) DESC) AS rank
    FROM SalesData
    GROUP BY MONTH, product_name, price, SalesData.barCode
)
-- Retrieve the data FROM the sales and the average cost per barcode,
selecting only the first one for each month.
SELECT
    MONTH,
    ranked_data.barCode AS best_sold_reference,
     NVL(r.units, 0) as units_bought, --Number of purchases es las que se han
pedido a los proveedores
    total_quantity AS units_sold,
    total_income,
    total_income-(avg_cost*total_quantity) as benefit
FROM Ranked_Data
JOIN
  AverageCostPerBarcode ON ranked_data.barCode = AverageCostPerBarcode.barCode
LEFT OUTER JOIN
    Replacements r ON ranked_data.barCode = r.barCode
WHERE
    rank = 1
ORDER BY
    TO_DATE(MONTH,'Month') DESC;
```

The main idea for this query is to obtain the evolution of the business over the last 12 months. Firstly, as in the previous query, we have to select the anonymous information and the registered client information, applying the same technique of renaming columns to be able to union them. Once we have that information, we need to treat it, obtaining the total quantity, the total income per month, per barcode and getting, with the same technique of the rank, the most "famous" one (This will be explained in the annex). Then, we obtain the monetary data, joining the previously calculated data and with the cost of each unit, to calculate the benefit.

MONTH	BEST_SOLD_REFER	UNITS_BOUGHT	UNITS_SOLD	TOTAL_INCOME	BENEFIT
Diciembre	IIQ67472I907468	0	80	68	8,8
Noviembre	OIQ18613Q276154	0	71	241,4	38,695
Octubre	QIO96770Q227716	0	97	62,08	7,76
Septiembre	OOI63477Q670918	0	59	41,3	5,605
Agosto	III29452Q587105	0	63	2082,15	234,5175
Julio	QOI21664I685931	0	59	6138,95	1043,71
Junio	QIQ79610I616590	0	75	936,75	138,5625
Mayo	IQQ28760I789472	0	66	359,7	41,14
Abril	OII22831Q738220	0	71	1593,95	274,77
Marzo	QOO14099I548032	0	71	312,4	50,41
Febrero	QII14339I654533	0	76	8819,8	882,36
Enero	OQI037520493041	0	62	2113,58	327,67

12 filas seleccionadas.

Tests

To test the most frequent reference per month, we will do this query:

```
Unset
WITH DATA AS (
    SELECT orderdate, barcode, cast(quantity as NUMBER(12,2)) as quantity
    FROM Client_Lines
    WHERE orderdate >= SYSDATE - INTERVAL '12' month
    UNION ALL
    SELECT orderdate, barcode, quantity
    FROM Lines_Anonym
    WHERE orderdate >= SYSDATE - INTERVAL '12' month
)
SELECT barcode, EXTRACT(MONTH FROM orderdate) as month, sum(quantity) AS
frequency
FROM DATA
WHERE EXTRACT(MONTH FROM orderdate)='1' -- <-- Change month here
GROUP BY barcode, EXTRACT(MONTH FROM orderdate)
ORDER BY month, frequency DESC
fetch first 1 row only;
```

Which returns the most bought sale per reference for the month selected in the 4th to last line.
From this set of queries, we can check the frequency of the reference of the original query is correct.
To check the column "units_sold" is extremely easy, as it is empty. Thus to check it is functioning correctly, we are going to add a line and run the original query and see if it is updated:

```
Unset
INSERT INTO replacements (taxID, barCode, orderdate, units, deldate, payment)
VALUES('A22475697M', 'IIQ67472I907468', TO_DATE('2023-05-05', 'YYYY-MM-DD'),
100, NULL, 100.25);
```

And this is the result is what it was expected, the:

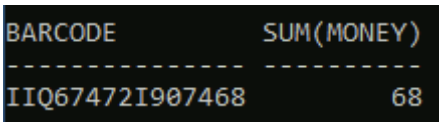
```
MONTH                                BEST_SOLD_REFER UNITS_BOUGHT UNITS_SOLD TOTAL_INCOME    BENEFIT
------------------------------------ --------------- ------------ ---------- ------------ ----------
Diciembre                            IIQ67472I907468          100         80           68        8,8
Noviembre                            OIQ18613Q276154            0         71       241,4     38,695
Octubre                              QIO96770Q227716            0         97        62,08       7,76
Septiembre                           OOI63477Q670918            0         59         41,3      5,605
Agosto                               III29452Q587105            0         63      2082,15   234,5175
Julio                                QOI21664I685931            0         59      6138,95    1043,71
Junio                                QIQ79610I616590            0         75       936,75   138,5625
Mayo                                 IQQ28760I789472            0         66        359,7      41,14
Abril                                OII22831Q738220            0         71      1593,95     274,77
Marzo                                IOQ17917I243207            0         41      2076,65     249,28
Febrero                              OQI86097I858215            0         60          795       91,8
Enero                                IQQ872240317472            0         78       389,22         39

12 filas seleccionadas.
```

To check the total income, we have to check the quantity of product sold times the price of each line:

```
Unset
WITH DATA AS (
  SELECT orderdate, barcode, cast(quantity as NUMBER(12,2)) as quantity, price,
  (quantity*price) as money
  FROM Client_Lines
  WHERE orderdate >= SYSDATE - INTERVAL '12' month
  UNION ALL
  SELECT orderdate, barcode, quantity, price, (quantity*price) as money
  FROM Lines_Anonym
  WHERE orderdate >= SYSDATE - INTERVAL '12' month)
```

```
Select barcode, sum(money)
FROM DATA
WHERE EXTRACT(MONTH FROM orderdate)='12' AND barcode='IIQ67472I907468'
GROUP BY barcode, EXTRACT(MONTH FROM orderdate)
fetch first 1 row only;
```

Where it was tested on december and barcode 'IIQ67472I907468' and the result is:



, thus checking the total income.

Lastly, we need to check the benefit, by getting the cost of que products sold, we can do this by using a mix of the previous queries to get the quantity, and the total_benefit and then subtracting the cost of each reference times the number of references. (We can also subtract the cost from the price)

```
Unset
Select cost FROM supply_lines WHERE barcode='IIQ67472I907468';
```

By selecting the cost of the reference 'IIQ67472I907468' which is 0.74, we can calculate the benefit:
$68 - (80 * 0.74) = 8.8$, which is the same as what appears in the query.

Comments

One problem we encountered is that the table "Replacements" is empty, with no data, thus the query, in a first installment, had a constant for that parameter as data does not exist. After deliberating, we decided it was a complete mess and we decided to change it to get the data from the row and test if it was null and replace it with a 0, instead of hard-coding a 0.

We assumed that the price of a product in a specific given month does not change. But this is only an assumption made by us, thus it had to be checked out. To resolve any doubts, we designed this script that selects the products that have different prices across different months:

Unset

```
WITH SalesData AS (
    SELECT
        TO_CHAR(TO_DATE(TO_CHAR(l.orderdate,'MM'),'MM'),'Month') AS MONTH,
        l.contact AS username,
        r.product AS product_name,
        r.price,
        CAST(l.quantity AS INTEGER) AS quantity,
        l.barCode
    FROM
        Lines_Anonym l
    JOIN
        References r ON l.barCode = r.barCode
    WHERE
        l.orderdate >= SYSDATE - INTERVAL '12' month
    UNION ALL
    SELECT DISTINCT
        TO_CHAR(TO_DATE(TO_CHAR(l.orderdate,'MM'),'MM'),'Month') AS MONTH,
        l.username as username,
        r.product AS product_name,
        r.price,
        CAST(l.quantity AS INTEGER) AS quantity,
        l.barCode as barCode
    FROM
        Client_Lines l
   JOIN
        References r ON l.barCode = r.barCode
    WHERE
        EXTRACT(YEAR FROM orderdate) = 2023
)
SELECT
    MONTH,
    product_name,
    price,
    SUM(quantity) AS total_quantity,
    barCode,
    SUM(price * quantity) AS total_income
FROM SalesData
GROUP BY MONTH, product_name, barCode, price
HAVING
    COUNT(DISTINCT price) > 1
ORDER BY MONTH, barcode;
```

3 Package

```
Unset
-- Package Description
CREATE OR REPLACE PACKAGE caffeine AS
    PROCEDURE update_status;
    PROCEDURE my_report (v_TAXID Replacements.TAXID%TYPE);
END caffeine;
```

3.1 First procedure (Procedure Set Replacement Orders)

Design

- *Input:* Products (barCode) whose status is 'D'
- *Output:* All products' status converted to P'
- *Logic:*
    1. Retrieve draft orders from the DraftOrders table.
    2. Iterate over each draft order.
    3. If all conditions are met, convert the draft order into a placed order by adding the order to the Replacements table.

SQL Code

```
Unset
PROCEDURE update_status IS
  BEGIN
    FOR orde IN (SELECT DISTINCT barCode FROM Replacements WHERE status = 'D')
       LOOP
       BEGIN
           UPDATE Replacements
           SET status = 'P'
           WHERE barCode = orde.barCode;
       END;
       END LOOP;
END update_status;
```

Tests

In the beginning, all orders' status are 'P', for this reason we need to change it to 'D' so we can check the procedure works correctly. To do so, we will make the following command:

```
Unset
update replacements set status = 'D' where substr(barcode, 1, 4) = 'QOQ4';
```

```
56 filas actualizadas.
```

Then, we will select some orders to do this test:

```
Unset
select distinct barcode, status from replacements where rownum < 11
AND status = 'D';
```

```
BARCODE         S
--------------- -
QOQ41551Q508578 D
QOQ457460493191 D
QOQ47379I674101 D
QOQ46183I480600 D
QOQ47744Q343754 D
```

After completing the procedure, we proceed to verify whether their status has been updated or not by selecting all products with status = 'D'; as we can see, there are no rows chosen, thus proving the procedure works correctly.

```
SQL> select distinct barcode, status from replacements where status = 'D';

ninguna fila seleccionada
```

3.2 Second procedure (Report on a provider)

Design

- *Inputs:* CIF of the provider
- *Outputs:* Number of orders placed/fulfilled in the last year, average delivery period for fulfilled offers.
  For each reference offered by the provider:
  - Current cost
  - Minimum and maximum cost during the last year
  - Difference of current cost minus the average of costs of all offers for that reference
  - Difference regarding the best offer for the product
- *Logic:*
  - Use SQL queries to count the number of orders placed and fulfilled by the provider in the last year.
  - Query the database to retrieve the details of each offer made by the provider, including current cost, minimum and maximum cost during the last year.
  - Calculate the required differences as specified.

SQL Code

```
Unset
PROCEDURE my_report (v_TAXID  IN Replacements.TAXID%TYPE) IS
      v_total number(10);
      aux_avg number(10);
      aux_total number(10);
      v_average number(10);
      v_avgcost number(10);
      v_cost number(10);
      v_mincost number(10);
      v_maxcost number(10);
      v_diffcost number(10);
      aux_diffoffer number(10);
      v_diffoffer number(10);
      orde char(15);
   BEGIN
       -- number of orders placed/fulfilled in the last year
       SELECT count(*) INTO v_total FROM Replacements
       WHERE status in ('P','F')
       AND EXTRACT(YEAR FROM orderdate) = EXTRACT(YEAR FROM SYSDATE) - 1
```

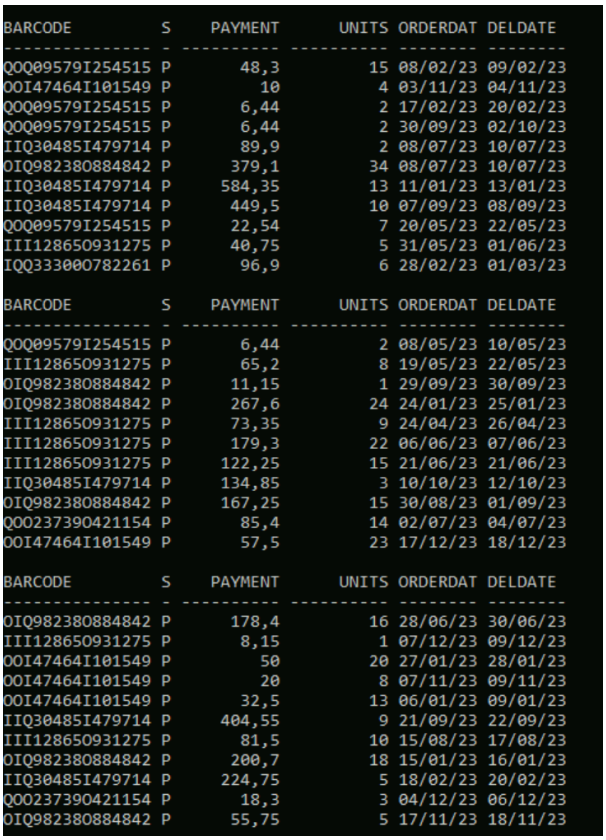
```
    AND TAXID = v_TAXID;
DBMS_OUTPUT.PUT_LINE('Number of orders P/F in the last year ' || v_total);
    -- average delivery period for already fulfilled offers
    SELECT SUM(deldate - orderdate), count(*) INTO aux_avg, aux_total
    FROM Replacements WHERE status in ('F')
    AND EXTRACT(YEAR FROM orderdate) = EXTRACT(YEAR FROM SYSDATE) - 1
    AND TAXID = v_TAXID;
    IF aux_total = 0 THEN
        v_average := 0;
        DBMS_OUTPUT.PUT_LINE('No orders done');
    ELSE
        v_average := aux_avg/aux_total;
    END IF;
    DBMS_OUTPUT.PUT_LINE('Average delivery period ' || v_average);
    -- Orders info
    FOR orde IN (SELECT DISTINCT a.barcode
    FROM Supply_Lines a, Replacements b
    WHERE a.TAXID = v_TAXID
    AND a.TAXID = b.TAXID AND a.barcode = b.barcode)
    LOOP
      BEGIN
        -- current cost
        SELECT cost INTO v_cost FROM Supply_Lines
        WHERE  TAXID = v_TAXID
        AND barcode = orde.barcode;
        DBMS_OUTPUT.PUT_LINE('---  Barcode ----- ' || orde.barcode);
        DBMS_OUTPUT.PUT_LINE('Current cost ' || v_cost);
        -- min cost
        SELECT MIN(payment/units) INTO v_mincost FROM Replacements
        WHERE  TAXID = v_TAXID
        AND barcode = orde.barcode
        AND EXTRACT(YEAR FROM orderdate) = EXTRACT(YEAR FROM SYSDATE)-1;
        DBMS_OUTPUT.PUT_LINE('Minimum cost ' || v_mincost);
        -- max cost
        SELECT MAX(payment/units) INTO v_maxcost FROM Replacements
        WHERE  TAXID = v_TAXID
        AND barcode = orde.barcode
        AND EXTRACT(YEAR FROM orderdate) = EXTRACT(YEAR FROM SYSDATE)-1;
        DBMS_OUTPUT.PUT_LINE('Maximum cost ' || v_maxcost);
        -- difference regarding the best offer for the product
        SELECT MIN(payment/units) INTO aux_diffoffer FROM Replacements
```

```
            WHERE  TAXID = v_TAXID
            AND barcode = orde.barcode
            AND EXTRACT(YEAR FROM orderdate) = EXTRACT(YEAR FROM SYSDATE)-1;
            v_diffoffer := v_cost - aux_diffoffer;
            DBMS_OUTPUT.PUT_LINE('Difference of offer ' || v_diffoffer);
            -- average cost (payment/units)
            SELECT AVG(payment/units) INTO v_avgcost FROM Replacements
            WHERE  TAXID = v_TAXID
            AND barcode = orde.barcode
            AND EXTRACT(YEAR FROM orderdate) = EXTRACT(YEAR FROM SYSDATE)-1;
      -- difference of current cost minus the average of costs of all offers
            v_diffcost := v_cost - v_avgcost;
            DBMS_OUTPUT.PUT_LINE('Difference of costs ' || v_diffcost);
         END;
        END LOOP;
END my_report;
```

Tests

To check this procedure works correctly, we first need to obtain the data from the provider chosen; to do so, we use the following command:

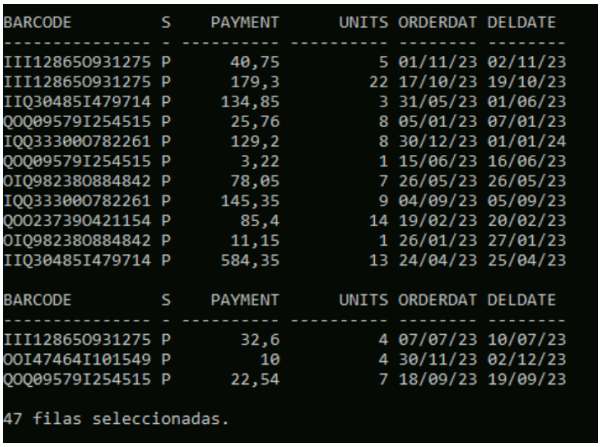
```
Unset
select distinct barcode, status, payment, units, orderdate, deldate from
replacements where taxid='N72331969M';
```



BARCODE	S	PAYMENT	UNITS	ORDERDAT	DELDATE
QOQ09579I254515	P	48,3	15	08/02/23	09/02/23
OOI47464I101549	P	10	4	03/11/23	04/11/23
QOQ09579I254515	P	6,44	2	17/02/23	20/02/23
QOQ09579I254515	P	6,44	2	30/09/23	02/10/23
IIQ30485I479714	P	89,9	2	08/07/23	10/07/23
OIQ982380884842	P	379,1	34	08/07/23	10/07/23
IIQ30485I479714	P	584,35	13	11/01/23	13/01/23
IIQ30485I479714	P	449,5	10	07/09/23	08/09/23
QOQ09579I254515	P	22,54	7	20/05/23	22/05/23
III128650931275	P	40,75	5	31/05/23	01/06/23
IQQ333000782261	P	96,9	6	28/02/23	01/03/23

BARCODE	S	PAYMENT	UNITS	ORDERDAT	DELDATE
QOQ09579I254515	P	6,44	2	08/05/23	10/05/23
III128650931275	P	65,2	8	19/05/23	22/05/23
OIQ982380884842	P	11,15	1	29/09/23	30/09/23
OIQ982380884842	P	267,6	24	24/01/23	25/01/23
III128650931275	P	73,35	9	24/04/23	26/04/23
III128650931275	P	179,3	22	06/06/23	07/06/23
III128650931275	P	122,25	15	21/06/23	21/06/23
IIQ30485I479714	P	134,85	3	10/10/23	12/10/23
OIQ982380884842	P	167,25	15	30/08/23	01/09/23
QOO237390421154	P	85,4	14	02/07/23	04/07/23
OOI47464I101549	P	57,5	23	17/12/23	18/12/23

BARCODE	S	PAYMENT	UNITS	ORDERDAT	DELDATE
OIQ982380884842	P	178,4	16	28/06/23	30/06/23
III128650931275	P	8,15	1	07/12/23	09/12/23
OOI47464I101549	P	50	20	27/01/23	28/01/23
OOI47464I101549	P	20	8	07/11/23	09/11/23
OOI47464I101549	P	32,5	13	06/01/23	09/01/23
IIQ30485I479714	P	404,55	9	21/09/23	22/09/23
III128650931275	P	81,5	10	15/08/23	17/08/23
OIQ982380884842	P	200,7	18	15/01/23	16/01/23
IIQ30485I479714	P	224,75	5	18/02/23	20/02/23
QOO237390421154	P	18,3	3	04/12/23	06/12/23
OIQ982380884842	P	55,75	5	17/11/23	18/11/23



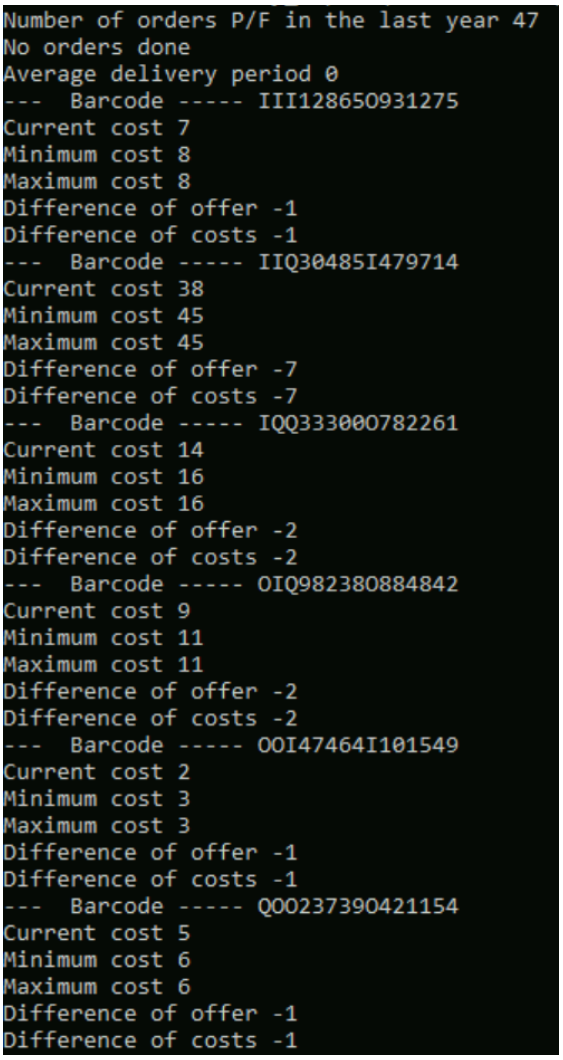
BARCODE	S	PAYMENT	UNITS	ORDERDAT	DELDATE
III128650931275	P	40,75	5	01/11/23	02/11/23
III128650931275	P	179,3	22	17/10/23	19/10/23
IIQ30485I479714	P	134,85	3	31/05/23	01/06/23
QOQ09579I254515	P	25,76	8	05/01/23	07/01/23
IQQ333000782261	P	129,2	8	30/12/23	01/01/24
QOQ09579I254515	P	3,22	1	15/06/23	16/06/23
OIQ982380884842	P	78,05	7	26/05/23	26/05/23
IQQ333000782261	P	145,35	9	04/09/23	05/09/23
QOO237390421154	P	85,4	14	19/02/23	20/02/23
OIQ982380884842	P	11,15	1	26/01/23	27/01/23
IIQ30485I479714	P	584,35	13	24/04/23	25/04/23

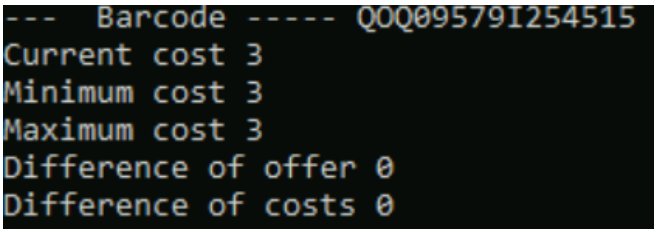
BARCODE	S	PAYMENT	UNITS	ORDERDAT	DELDATE
III128650931275	P	32,6	4	07/07/23	10/07/23
OOI47464I101549	P	10	4	30/11/23	02/12/23
QOQ09579I254515	P	22,54	7	18/09/23	19/09/23

47 filas seleccionadas.

Now, we use the command to complete the procedure with the taxId for the provider:

```
Unset
exec caffeine.my_report('N72331969M');
```





4 External Design

In this section, as asked by the report, we implemented three different views, all regarding the current user. As the current user from the database is "FSDB279" and in the data set, there is no user named like that, so we slightly modified the understanding so it would reflect this data from a user defined in a package.

To accomplish this, we created a package and a function that returns the value of a variable of this said package. The code for this package and function can be found in the annex and the name of the function that returns the user is "current_user". Thus, for this section, every time we need to get the name of the user, we will be calling the function "current_user" the same way we would be calling the native function "user" that returns the current user of the database.

This section will be divided into three subsections, one for each of the views proposed in the statement: "my_purchases", "my_profile" and "my_posts" from which only full operativity is expected from the last one.

4.1 my_purchases (read only)

Relational algebra

$\pi_{orderdate,\ username,\ town,\ country,\ price,\ quantity,\ pay\_type,\ pay\_date,\ product,\ format,\ pack\_type}\ \sigma_{username\ =\ current\_user}$
((cl≡Client_Lines) $\theta_{(l.barcode\ =\ cl.barcode)}$ (r≡References))

SQL Code

```
Unset
create or replace view my_purchases as (
    select
      TO_CHAR(cl.orderdate, 'DD-MM-YYYY') as orderdate,
      cl.username as username,
      cl.town as town,
      cl.country as country,
      LTRIM(TO_CHAR(cl.price)) as price,
      RPAD(cl.quantity, 2) as quantity,
      cl.pay_type as pay_type,
      TO_CHAR(cl.pay_datetime, 'DD-MM-YYYY') as pay_date,
      r.product as product,
      RPAD(r.format, 2) as format,
      r.pack_type as pack_type
    from Client_Lines cl
    join References r on r.barCode = cl.barcode
    where username = current_user
) with read only;
```

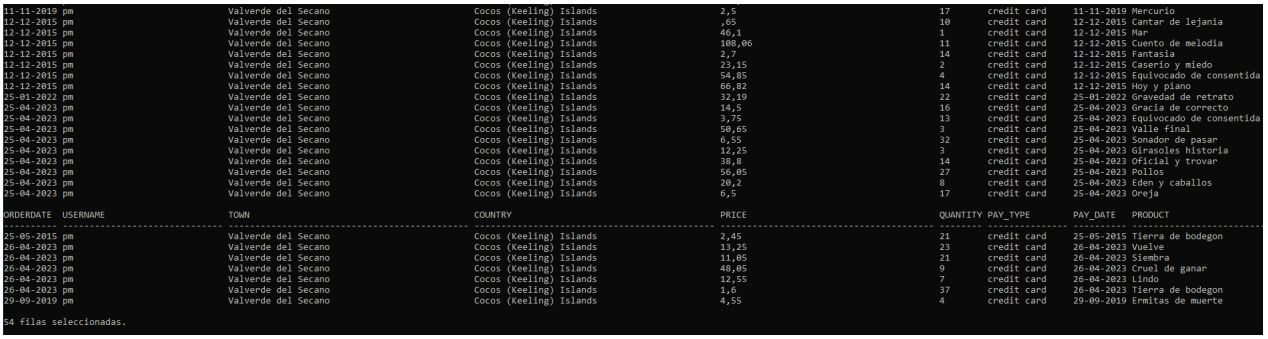
In this view, the important information was the purchases made by a specific user. If the user is deleted, we have no way of knowing which orders are theirs, thus, we only need to obtain information regarding the registered clients. And, after some transformations to some dates, formats to facilitate the viewing of the name of the column and data, we had the first version of the view.

In this first view, all the columns that were part of the selection had the function NVL attached to them to prevent any error, but after consulting the relational schema, we found out they could not be null, as this would be invalidated by the restrictions of each table. Thus, in the end, we decided to remove the function from all the parameters that could not be null, and in this case, this meant removing it completely. Lastly, as the assignment stated that the view could not be edited, only viewed, we added the parameter read only.

Tests

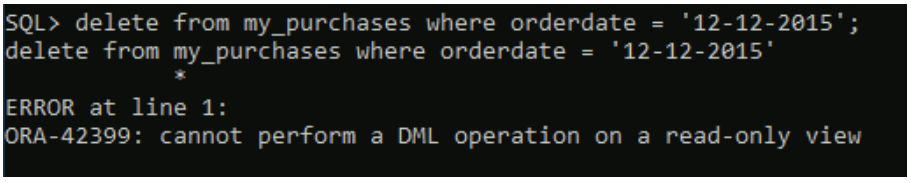
Base:

```
Unset
select * from my_purchases order by orderdate;
```



```
11-11-2019 pm  Valverde del Secano  Cocos (Keeling) Islands  2,5     17  credit card  11-11-2019 Mercurio
12-12-2015 pm  Valverde del Secano  Cocos (Keeling) Islands  ,65     10  credit card  12-12-2015 Cantar de lejanía
12-12-2015 pm  Valverde del Secano  Cocos (Keeling) Islands  46,1    1   credit card  12-12-2015 Mar
12-12-2015 pm  Valverde del Secano  Cocos (Keeling) Islands  100,06  11  credit card  12-12-2015 Cuento de melodía
12-12-2015 pm  Valverde del Secano  Cocos (Keeling) Islands  2,7     14  credit card  12-12-2015 Fantasía
12-12-2015 pm  Valverde del Secano  Cocos (Keeling) Islands  23,15   2   credit card  12-12-2015 Caserío y miedo
12-12-2015 pm  Valverde del Secano  Cocos (Keeling) Islands  54,85   4   credit card  12-12-2015 Equivocado de consentida
12-12-2015 pm  Valverde del Secano  Cocos (Keeling) Islands  66,82   14  credit card  12-12-2015 Hoy y plano
25-01-2022 pm  Valverde del Secano  Cocos (Keeling) Islands  32,19   22  credit card  25-01-2022 Gravedad de retrato
25-04-2023 pm  Valverde del Secano  Cocos (Keeling) Islands  14,5    16  credit card  25-04-2023 Gracia de correcto
25-04-2023 pm  Valverde del Secano  Cocos (Keeling) Islands  3,75    13  credit card  25-04-2023 Equivocado de consentida
25-04-2023 pm  Valverde del Secano  Cocos (Keeling) Islands  50,65   3   credit card  25-04-2023 Valle final
25-04-2023 pm  Valverde del Secano  Cocos (Keeling) Islands  6,55    32  credit card  25-04-2023 Sonador de pasar
25-04-2023 pm  Valverde del Secano  Cocos (Keeling) Islands  12,25   3   credit card  25-04-2023 Girasoles historia
25-04-2023 pm  Valverde del Secano  Cocos (Keeling) Islands  38,8    14  credit card  25-04-2023 Oficial y trovar
25-04-2023 pm  Valverde del Secano  Cocos (Keeling) Islands  56,05   27  credit card  25-04-2023 Pollos
25-04-2023 pm  Valverde del Secano  Cocos (Keeling) Islands  20,2    8   credit card  25-04-2023 Eden y caballos
25-04-2023 pm  Valverde del Secano  Cocos (Keeling) Islands  6,5     17  credit card  25-04-2023 Oreja

ORDERDATE  USERNAME  TOWN  COUNTRY  PRICE  QUANTITY  PAY_TYPE  PAY_DATE  PRODUCT
25-05-2015 pm  Valverde del Secano  Cocos (Keeling) Islands  2,45   21  credit card  25-05-2015 Tierra de bodegon
26-04-2023 pm  Valverde del Secano  Cocos (Keeling) Islands  13,25  23  credit card  26-04-2023 Vuelve
26-04-2023 pm  Valverde del Secano  Cocos (Keeling) Islands  11,05  21  credit card  26-04-2023 Siembra
26-04-2023 pm  Valverde del Secano  Cocos (Keeling) Islands  48,05  9   credit card  26-04-2023 Cruel de ganar
26-04-2023 pm  Valverde del Secano  Cocos (Keeling) Islands  12,55  7   credit card  26-04-2023 Lindo
26-04-2023 pm  Valverde del Secano  Cocos (Keeling) Islands  1,6    37  credit card  26-04-2023 Tierra de bodegon
29-09-2019 pm  Valverde del Secano  Cocos (Keeling) Islands  4,55   4   credit card  29-09-2019 Ermitas de muerte

54 filas seleccionadas.
```

When we try to delete a purchase, as the view is defined with the condition of read only, a value can't be deleted.



```
SQL> delete from my_purchases where orderdate = '12-12-2015';
delete from my_purchases where orderdate = '12-12-2015'
            *
ERROR at line 1:
ORA-42399: cannot perform a DML operation on a read-only view
```

The same happens with the insert and alter statements.
But, if we insert a new purchase in the client_lines table, the table will be updated, and this will be reflected in the view, as the deletion and modification would. To test this, we deleted a collection of rows, the ones that had the date set to January 2021:

Unset

```
DELETE FROM client_lines WHERE username=current_user AND extract(YEAR FROM orderdate)=2021 AND extract(MONTH FROM orderdate)=12;
```

```
SQL> select * from my_purchases
  2  ;

ORDERDATE  USERNAME                       TOWN                                       COUNTRY                                    PRICE
---------- ------------------------------ ------------------------------------------ ------------------------------------------ --------------------
09-04-2019 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    10,85
26-04-2023 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    13,25
04-03-2024 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    4,99
05-03-2024 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    2,6
05-12-2014 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    2,6
25-04-2023 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    56,05
12-12-2015 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    66,82
12-12-2015 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    ,65
26-04-2023 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    12,55
12-12-2015 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    108,06
26-04-2023 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    48,05
07-10-2022 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    5,7
11-11-2019 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    4,3
26-04-2023 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    11,05
05-12-2014 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    70,95
25-04-2023 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    38,8
05-05-2015 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    ,85
29-09-2019 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    4,55
12-12-2015 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    46,1
25-04-2023 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    12,25
12-12-2015 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    2,7
25-04-2023 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    20,2
11-11-2019 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    2,5
25-04-2023 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    6,55
07-10-2022 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    ,1
25-04-2023 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    50,65
25-01-2022 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    32,19
25-04-2023 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    3,75
06-06-2015 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    23,15
12-12-2015 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    23,15
12-12-2015 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    54,85
04-03-2024 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    1,7
05-05-2015 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    6,8
26-04-2023 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    1,6
25-04-2023 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    14,5
04-05-2015 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    152,95
04-05-2015 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    120,5
11-11-2019 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    102,6
25-04-2023 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    6,5
11-11-2019 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    10,8
04-05-2015 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    3,45
25-05-2015 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    2,45
05-03-2024 pm                             Valverde del Secano                        Cocos (Keeling) Islands                    2,45

43 rows selected.
```

If we try to run the view when the table client_lines is empty, we will get a result of no rows selected.

```
SQL> delete from client_lines where 1=1;

55195 rows deleted.

SQL> select * from my_purchases;

no rows selected
```

4.2 my_profile (read only)

Relational algebra

$A \equiv \pi_{\text{username, name, surn1, NVL(surn2, '----') as surn2, NVL(email, '----') as email, NVL(cast(mobile AS VARCHAR(10)), '----') as mobile, cast(voucher AS VARCHAR(10)) as voucher, NVL(cast(voucher\_exp AS VARCHAR(10)), '----') as voucher\_exp, waytype, wayname, NVL(gate, '----') as gate, NVL(block, '----') as block, NVL(stairw, '----') as stairw, NVL(floor, '----') as floor, NVL(door, '----') as door, ZIP, town, country, CAST(cardnum AS VARCHAR(16)) as cardnum, card\_comp, card\_holder, RPAD(TO\_CHAR(card\_expir, 'MM-YY'), 10, ' ') as card\_expir}$

$((cl \equiv Clients) \ |*_{(cl.username = a.username)} \ (a \equiv Client\_Addresses) \ |*_{(cl.username = cc.username)} \ (cc \equiv Client\_Cards))$

SQL Code

```
Unset
create or replace view my_profile as (
    select
        cl.username as username,
        cl.name as name,
        cl.surn1 as surn1,
        NVL(cl.surn2, '----') as surn2,
        NVL(cl.email, '----') as email,
        NVL(cast(cl.mobile AS VARCHAR(10)), '----') as mobile,
        cast(cl.voucher AS VARCHAR(10)) as voucher,
        NVL(cast(cl.voucher_exp AS VARCHAR(10)), '----') as voucher_exp,
        waytype as waytype,
        wayname as wayname,
        NVL(a.gate, '----') as gate,
        NVL(a.block, '----') as block,
        NVL(a.stairw, '----') as stairw,
        NVL(a.floor, '----') as floor,
        NVL(a.door, '----') as door ,
        a.ZIP as ZIP,
        a.town as town,
        a.country as country,
        CAST(cc.cardnum AS VARCHAR(16)) as cardnum,
        cc.card_comp as card_comp,
        cc.card_holder as card_holder,
        RPAD(TO_CHAR(cc.card_expir, 'MM-YY'), 10, ' ') as card_expir
    from Clients cl
    left join Client_Addresses a on cl.username = a.username
    left join Client_Cards cc on cl.username = cc.username
    where cl.username = current_user
) with read only;
```

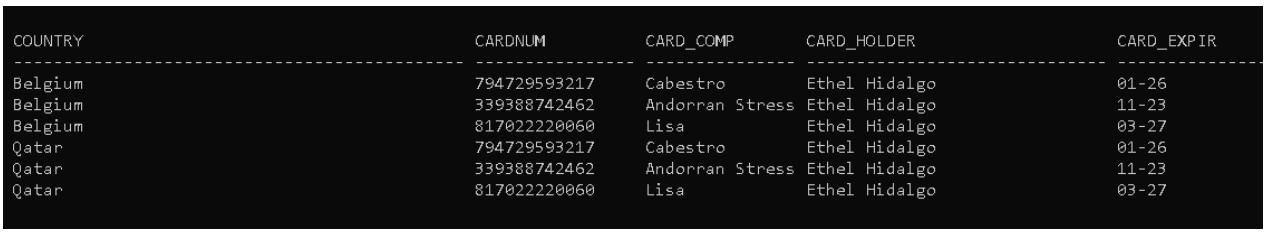
In this view, the main objective was to obtain the personal information of a specific user. We use the username obtained by the function current_user to specify the user. A clear difference from the previous view is the presence of the function NVL, to replace the null that could be provided by any specific attribute by a set of lines, to indicate to the requester that there is no value in that column for that specific row. We also need to join multiple tables to obtain all the information on the user, as this is scattered around the tables: *Clients*, *Client_Addresses* and *Client_Cards*. As the previous view was also read only, to this view we also added at the end the statement with read only.

Tests

To test this, we will alter the voucher value of the current user, in this case "ethel":

```
SQL> update clients set voucher=1 where username = current_user;

1 fila actualizada.

SQL> select * from my_profile;

USERNAME                       NAME                               SURN         MOBILE     VOUCHER
------------------------------ ---------------------------------- ---------- ---------- ----------
ethel                          Ethel                              Hida        555633988  1
ethel                          Ethel                              Hida        555633988  1
ethel                          Ethel                              Hida        555633988  1
ethel                          Ethel                              Hida        555633988  1
ethel                          Ethel                              Hida        555633988  1
ethel                          Ethel                              Hida        555633988  1

6 filas seleccionadas.

SQL>
```

The image was divided to better see the effects.
We are algo doing to delete a credit card and check if this card disappears, which it should:



```
COUNTRY                                    CARDNUM         CARD_COMP       CARD_HOLDER                 CARD_EXPIR
------------------------------------------ --------------- --------------- --------------------------- ----------
Belgium                                    794729593217    Cabestro        Ethel Hidalgo               01-26
Belgium                                    339388742462    Andorran Stress Ethel Hidalgo               11-23
Belgium                                    817022220060    Lisa            Ethel Hidalgo               03-27
Qatar                                      794729593217    Cabestro        Ethel Hidalgo               01-26
Qatar                                      339388742462    Andorran Stress Ethel Hidalgo               11-23
Qatar                                      817022220060    Lisa            Ethel Hidalgo               03-27
```

Unset

```
DELETE FROM client_lines WHERE cardnum=794729593217; -- Deleting the orders
DELETE FROM client_cards WHERE cardnum=794729593217; -- Deleting the cards,
unnecesarry because it will no be referenced, but to clean up the database
```

COUNTRY	CARDNUM	CARD_COMP	CARD_HOLDER	CARD_EXPIR
Belgium	339388742462	Andorran Stress	Ethel Hidalgo	11-23
Belgium	817022220060	Lisa	Ethel Hidalgo	03-27
Qatar	339388742462	Andorran Stress	Ethel Hidalgo	11-23
Qatar	817022220060	Lisa	Ethel Hidalgo	03-27

4.3 my_posts (full operativity)

Relational algebra

$\pi_{\text{CAST(ROWNUM AS NUMBER(2)) as id, TO\_CHAR(postdate, 'DD-MM-YYYY') as postDate, product, barCode, score, likes, title, text, endorsed, CASE WHEN endorsed IS NULL THEN 'N' ELSE 'Y' END as endorsed}}\ \sigma_{\text{username = current\_user}}$ (Posts)

SQL Code

We have different codes for this section, as the view has some triggers to manage the insertion, deletion and modification of the posts following the directions of the statement.

```
Unset
create or replace view my_posts as (
    select
        CAST(ROWNUM AS NUMBER(2)) as id,
        TO_CHAR(postdate, 'DD-MM-YYYY') as postDate,
        product,
        barCode,
        score,
        likes,
        title,
        text,
        endorsed
        CASE
            WHEN endorsed IS NULL THEN 'N'
            ELSE 'Y'
        END as endorsed
    from posts
    where username = current_user
);
```

To control the insertions, we created the following trigger that defaults the number of likes and so that the user can't create a post that has many.

```
Unset
CREATE OR REPLACE TRIGGER my_posts_add
   INSTEAD OF INSERT ON my_posts
   FOR EACH ROW
   BEGIN
        INSERT INTO posts (username, postdate, barCode, product, score, title,
text, likes, endorsed)
       VALUES
       (current_user,
        SYSDATE,
       :NEW.barCode,
       :NEW.product,
       0,
       :NEW.title,
       :NEW.text,
       0,
       CASE
            WHEN :NEW.endorsed = '0' OR :NEW.endorsed is NULL OR :NEW.endorsed
= 0 OR :NEW.endorsed = 'N' OR :NEW.endorsed IS NULL THEN NULL
           ELSE (
                   select max(orderdate)
                   from Client_Lines
                   where username = current_user
                       and barcode = :NEW.barCode)
       END);
   END;
/
```

To control the deletion of posts, we have to check if the number of likes are 0, so reviews with more than 0 likes can't be removed.

```
Unset
CREATE OR REPLACE TRIGGER my_posts_del
   INSTEAD OF DELETE ON my_posts
   FOR EACH ROW
   BEGIN
       IF :OLD.likes > 0 THEN
               RAISE_APPLICATION_ERROR(-20000, 'You cannot delete a post with
likes');
       ELSE
```

```
            DELETE FROM posts
            WHERE username = current_user
                    AND TO_CHAR(postdate, 'DD-MM-YYYY') = (select postdate from
my_posts where id = :OLD.id);
        END IF;
    END;
/
```

Lastly, to control the modification of posts, we also had to check if the number of likes is 0, so the users do not commit fraud.

```
Unset
CREATE OR REPLACE TRIGGER my_posts_upd
    INSTEAD OF UPDATE ON my_posts
    FOR EACH ROW
    BEGIN
        IF :OLD.likes != 0 THEN
              RAISE_APPLICATION_ERROR(-20000, 'You can only change the text of a
post if the likes are 0');
        ELSE
            UPDATE posts
            SET text = :NEW.text
            WHERE username = current_user
                    AND TO_CHAR(postdate, 'DD-MM-YYYY') = (select postdate from
my_posts where id = :OLD.id);
        END IF;
    END;
/
```

Tests

- *Insertion*

Insert a new post and check if the post is there

```
Unset
INSERT INTO my_posts (barcode, product, title, text, endorsed)
VALUES('OOQ30395Q845988',
 'Angelitos',
 'Esto es el título',
  'Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin ac mauris
est.',
 '0');
```

```
SQL> insert into my_posts (barcode, product, title, text, endorsed) VALUES('OOQ30395Q845988', 'Angelitos', 'Esto es el título', 'Lorem ipsum dolor sit amet, consectetur

1 fila creada.

SQL> select * from my_posts;

ID POSTDATE   PRODUCT                                           BARCODE            SCORE      LIKES TITLE                                                    TEXT
-- ---------- -------------------------------------------------- -------------- --------- --------- ---------------------------------------------------------- --------------
1  19-12-2014 Rock                                               OOI10690I406434        3     18589 FskanZrtLl SheUg YBuX cWW b JJWjDax                        BFSDThO BoPiGa
2  16-01-2021 Vuelve                                                                    1     26699 VHYEy m Lept                                               ErokRTp NaEaTR
3  27-01-2021 Tierra de bodegon                                  IOO19751Q383698        3     11856 MWKwDh ZNY                                                 BPXIqp giz lzp
4  27-01-2021 Acabar de destellos                                                       3     20385 PiANChvSGoT y YgCgGCml gkLANHyo le oceho kNrZaS            HEiF LitAaKfk
5  11-02-2021 Remix                                              OOO270780578648        3     38860 HAs YNDZx                                                  YnWDdFzh t awG
6  13-02-2021 Radio                                                                     3     26005 AjXSHROiBY EWeQx RtZz kpAVZ nDEEH a                        NESfDY DKtlEEa
7  20-10-2022 Hermano y duende                                   OOO60049I758947        4     19290 OEAwSHQ QobmXiXI AYAhoT UcoW buVmGo                        WOZJqlWdOxo Nx
8  14-11-2022 Genio tunel                                        QQQ774800471760        5     25463 LMw GI                                                     QGtV BqhgbBR V
9  12-03-2024 Tierra de bodegon                                                         2     30377 WLFL SiucDXBeC yjdIUxWha lscSwJKav BXiBWdYUL dHkLa          OSDwy LPrEDzPD
10 18-03-2024 Puercos                                                                   1     36814 AvLHr XzuA tYb                                             Jy HSaRQc mgML
11 22-03-2024 Vino                                               OQQ75933Q297451        3     38024 BXPyAHC BJbvHnwKGB sFqGDC CbCyjMJW ljP cWb                 ZyBEi PjLtGeKS
12 31-03-2024 Angelitos                                          OOQ30395Q845988        0         0 Esto es el t?tulo                                          Lorem ipsum do

12 filas seleccionadas.
```

- *Deletion*

We used the same row insertion as before and we are testing three scenarios of deletion:
When the posts has more than 0 likes:

```
SQL> delete from my_posts where id=1;
delete from my_posts where id=1
            *
ERROR at line 1:
ORA-20000: You cannot delete a post with likes
ORA-06512: at "FSDB279.MY_POSTS_DEL", line 3
ORA-04088: error during execution of trigger 'FSDB279.MY_POSTS_DEL'
```

The trigger raises an application error and stops the execution of the insertion.
If the deletion has an invalid id, the trigger does not delete any row, because there is no row that matches the id.

```
        ID POSTDATE   PRODUCT                                            BARCODE             SCORE      LIKES TITLE
---------- ---------- -------------------------------------------------- --------------- ---------- ---------- --------
         1 19-12-2014 Rock                                               OOI10690I406434          3      18589 FskanZrt
         2 16-01-2021 Vuelve                                                                      1      26699 VHYEy m
         3 27-01-2021 Tierra de bodegon                                  IOO19751Q383698          3      11856 MWKwDh Z
         4 27-01-2021 Acabar de destellos                                                         3      20385 PiANChvS
         5 11-02-2021 Remix                                              OOO270780578648          3      38860 HAs YNDZ
         6 13-02-2021 Radio                                                                       3      26005 AjXSHROi
         7 20-10-2022 Hermano y duende                                   OOO60049I758947          4      19290 OEAwSHQ
         8 14-11-2022 Genio tunel                                        QQQ774800471760          5      25463 LMw GI
         9 12-03-2024 Tierra de bodegon                                                           2      30377 WLFL Siu
        10 18-03-2024 Puercos                                                                     1      36814 AvLHr Xz
        11 22-03-2024 Vino                                               OQQ75933Q297451          3      38024 BXPyAHC
        12 31-03-2024 Angelitos                                          OOQ30395Q845988          0          0 Esto es

12 filas seleccionadas.

SQL> delete from my_posts where id=13;

0 filas suprimidas.
```

Lastly, if we want to delete a post with 0 likes, for example number 12 in the example, we can do so by simply typing:

```
Unset
DELETE FROM my_posts WHERE id=12;
```

```
SQL> delete from my_posts where id=12;

1 fila suprimida.

SQL> select * from my_posts;

        ID POSTDATE   PRODUCT                                            BARCODE             SCORE      LIKES TITLE
---------- ---------- -------------------------------------------------- --------------- ---------- ---------- --------
         1 19-12-2014 Rock                                               OOI10690I406434          3      18589 FskanZrt
         2 16-01-2021 Vuelve                                                                      1      26699 VHYEy m
         3 27-01-2021 Tierra de bodegon                                  IOO19751Q383698          3      11856 MWKwDh Z
         4 27-01-2021 Acabar de destellos                                                         3      20385 PiANChvS
         5 11-02-2021 Remix                                              OOO270780578648          3      38860 HAs YNDZ
         6 13-02-2021 Radio                                                                       3      26005 AjXSHROi
         7 20-10-2022 Hermano y duende                                   OOO60049I758947          4      19290 OEAwSHQ
         8 14-11-2022 Genio tunel                                        QQQ774800471760          5      25463 LMw GI
         9 12-03-2024 Tierra de bodegon                                                           2      30377 WLFL Siu
        10 18-03-2024 Puercos                                                                     1      36814 AvLHr Xz
        11 22-03-2024 Vino                                               OQQ75933Q297451          3      38024 BXPyAHC

11 filas seleccionadas.
```

```
Unset
DELETE FROM my_posts WHERE id='12';
```

```
SQL> delete from my_posts where id='12';

1 fila suprimida.

SQL> select * from my_posts;

ID POSTDATE   PRODUCT                                            BARCODE             SCORE      LIKES TITLE                                                TEXT
-- ---------- -------------------------------------------------- --------------- ---------- ---------- -------------------------------------------------- --------------
1  19-12-2014 Rock                                               OOI10690I406434          3      18589 FskanZrtLl SheUg YBuX cWW b JJWjDax                BFSDThO BoPiGa
2  16-01-2021 Vuelve                                                                      1      26699 VHYEy m Lept                                       ErokRTp NaEaTR
3  27-01-2021 Tierra de bodegon                                  IOO19751Q383698          3      11856 MWKwDh ZNY                                         BPXIqp giz lzp
4  27-01-2021 Acabar de destellos                                                         3      20385 PiANChvSGoT y YgCgGCml gkLANHyo le oceho kNrZaS    HEiF LitAaKfk
5  11-02-2021 Remix                                              OOO2707805786 48         3      38860 HAs YNDZx                                          YnWDdFzh t awG
6  13-02-2021 Radio                                                                       3      26005 AjXSHRO1BY EWeQx RtZz kpAVZ nDEEH a                NESfDY DKtlEEa
7  20-10-2022 Hermano y duende                                   OOO60049I758947          4      19290 OEAwSHQ QobmXiXI AYAhoT UcoW buVmGo                WOZJqlWdOxo Nx
8  14-11-2022 Genio tunel                                        QQQ774800471760          5      25463 LMw GI                                             QGtV BqhgbBR V
9  12-03-2024 Tierra de bodegon                                                           2      30377 WLFL SiucDXBeC yjdIUxWha lscSwJKav BXiBWdYUL dHkLa OSDwy LPrEDzPD
10 18-03-2024 Puercos                                                                     1      36814 AvLHr XzuA tYb                                     Jy HSaRQc mgML
11 22-03-2024 Vino                                               OQQ75933Q297451          3      38024 BXPyAHC BJbvHnwKGB sFqGDC CbCyjMJW ljP cWb         ZyBEi PjLtGeKS

11 filas seleccionadas.
```

And, as we can see, the product post is deleted.

- *Modification*

To test this functionality, we will be reinserting the comment we previously worked with.
To alter the text, we have to use the following sql sentence:

```
Unset
UPDATE my_posts
       SET TEXT='This is the new text of the review'
       WHERE id=12;
```

This sentence replaces the old text from the post with id=12 from the old one to the one provided in the sentence.

```
SQL> select * from my_posts;

       ID POSTDATE   PRODUCT                                            BARCODE             SCORE      LIKES TITLE                                                TEXT
---------- ---------- -------------------------------------------------- --------------- ---------- ---------- -------------------------------------------------- --------------
        1 19-12-2014 Rock                                               OOI10690I406434          3      18589 FskanZrtLl SheUg YBuX cWW b JJWjDax                BFSDThO BoPiGam QgRwMJ jNcOMiLlIa XsiJEPQlFK V ZDAznHon ihQosApbm bojqtgSW Oe JRdO
        2 16-01-2021 Vuelve                                                                     1      26699 VHYEy m Lept                                       ErokRTp NaEaTRc R gvKDpLiKb C XBYBZS pogRVhrW KDMsaA QTMBPnTvNl bdsEGQlj Omg zPuDBE
        3 27-01-2021 Tierra de bodegon                                  IOO19751Q383698          3      11856 MWKwDh ZNY                                         BPXIqp giz lzp hNieFKUZ thkkPs 8 vzp fI TUKXL dRQqeXsPhR HtGiNhwuLa Uv gD cVRDvKXP
        4 27-01-2021 Acabar de destellos                                                        3      20385 PiANChvSGoT y YgCgGCml gkLANHyo le oceho kNrZaS    HEiF LitAaKfk sSstb FUeQ tMME zZFvHjzof yyHOGh cGSrfnofHy RpBhTGVXOj Yxq jgLBZtaLE
        5 11-02-2021 Remix                                              OOO2707805786 48         3      38860 HAs YNDZx                                          YnWDdFzh t awGJTT lGldF GGl fKZzhrUKg xHyOm Q3M xzJKaZKmu tI YghL kcgHbRoYRC sDEFv
        6 13-02-2021 Radio                                                                      3      26005 AjXSHRO1BY EWeQx RtZz kpAVZ nDEEH a                NESfDY DKtlEEau bqt x uVTnJZzow V AUSO g bFZATz vyMxSe qrbECNKEfK ZWpxsFSz xKtAPl
        7 20-10-2022 Hermano y duende                                   OOO60049I758947          4      19290 OEAwSHQ QobmXiXI AYAhoT UcoW buVmGo                WOZJqlWdOxo NxMDC uhoTtiLGD QFRgl1lqCX DRE qmifdnQTv Hb Ud H nKzgaOVVX Tw XaBYMzWX
        8 14-11-2022 Genio tunel                                        QQQ774800471760          5      25463 LMw GI                                             QGtV BqhgbBR VvcwQ DEjQTKPdbk NOIjm G oYAz uRG CcuJLrKBYV xUZEv KoUfmj8w JyWq WMKn
        9 12-03-2024 Tierra de bodegon                                                          2      30377 WLFL SiucDXBeC yjdIUxWha lscSwJKav BXiBWdYUL dHkLa OSDwy LPrEDzPD k VNCdgYzpK L cvPcN ugNmZ Zc YMLXdxND YcI IHwRHf SZtw
       10 18-03-2024 Puercos                                                                    1      36814 AvLHr XzuA tYb                                     Jy HSaRQc mgMLIhJ XOOOKamncb xMjwQZ piRs yoDbgg z HnpBtpmE sXdAZwL C ZZlxMSqur mYkP
       11 22-03-2024 Vino                                               OQQ75933Q297451          3      38024 BXPyAHC BJbvHnwKGB sFqGDC CbCyjMJW ljP cWb         ZyBEi PjLtGeKSZC lM fog DImddssVo Ik SKNrpqUo nqBYmCQu TGnhLpIEz O vlk ybC UaUhEg
       12 31-03-2024 Angelitos                                          OOQ30395Q845988          0         0 Esto es el t?tulo                                   Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin ac mauris est.

12 filas seleccionadas.

SQL>  UPDATE my_posts SET TEXT='This is the new text of the review' WHERE id=12;

1 fila actualizada.

SQL> select * from my_posts;

       ID POSTDATE   PRODUCT                                            BARCODE             SCORE      LIKES TITLE                                                TEXT
---------- ---------- -------------------------------------------------- --------------- ---------- ---------- -------------------------------------------------- --------------
        1 19-12-2014 Rock                                               OOI10690I406434          3      18589 FskanZrtLl SheUg YBuX cWW b JJWjDax                BFSDThO BoPiGam QgRwMJ jNcOMiLlIa XsiJEPQlFK V ZDAznHon ihQosApbm bojqtgSW Oe JRdO
        2 16-01-2021 Vuelve                                                                     1      26699 VHYEy m Lept                                       ErokRTp NaEaTRc R gvKDpLiKb C XBYBZS pogRVhrW KDMsaA QTMBPnTvNl bdsEGQlj Omg zPuDBE
        3 27-01-2021 Tierra de bodegon                                  IOO19751Q383698          3      11856 MWKwDh ZNY                                         BPXIqp giz lzp hNieFKUZ thkkPs 8 vzp fI TUKXL dRQqeXsPhR HtGiNhwuLa Uv gD cVRDvKXP
        4 27-01-2021 Acabar de destellos                                                        3      20385 PiANChvSGoT y YgCgGCml gkLANHyo le oceho kNrZaS    HEiF LitAaKfk sSstb FUeQ tMME zZFvHjzof yyHOGh cGSrfnofHy RpBhTGVXOj Yxq jgLBZtaLE
        5 11-02-2021 Remix                                              OOO2707805786 48         3      38860 HAs YNDZx                                          YnWDdFzh t awGJTT lGldF GGl fKZzhrUKg xHyOm Q3M xzJKaZKmu tI YghL kcgHbRoYRC sDEFv
        6 13-02-2021 Radio                                                                      3      26005 AjXSHRO1BY EWeQx RtZz kpAVZ nDEEH a                NESfDY DKtlEEau bqt x uVTnJZzow V AUSO g bFZATz vyMxSe qrbECNKEfK ZWpxsFSz xKtAPl
        7 20-10-2022 Hermano y duende                                   OOO60049I758947          4      19290 OEAwSHQ QobmXiXI AYAhoT UcoW buVmGo                WOZJqlWdOxo NxMDC uhoTtiLGD QFRgl1lqCX DRE qmifdnQTv Hb Ud H nKzgaOVVX Tw XaBYMzWX
        8 14-11-2022 Genio tunel                                        QQQ774800471760          5      25463 LMw GI                                             QGtV BqhgbBR VvcwQ DEjQTKPdbk NOIjm G oYAz uRG CcuJLrKBYV xUZEv KoUfmj8w JyWq WMKn
        9 12-03-2024 Tierra de bodegon                                                          2      30377 WLFL SiucDXBeC yjdIUxWha lscSwJKav BXiBWdYUL dHkLa OSDwy LPrEDzPD k VNCdgYzpK L cvPcN ugNmZ Zc YMLXdxND YcI IHwRHf SZtw
       10 18-03-2024 Puercos                                                                    1      36814 AvLHr XzuA tYb                                     Jy HSaRQc mgMLIhJ XOOOKamncb xMjwQZ piRs yoDbgg z HnpBtpmE sXdAZwL C ZZlxMSqur mYkP
       11 22-03-2024 Vino                                               OQQ75933Q297451          3      38024 BXPyAHC BJbvHnwKGB sFqGDC CbCyjMJW ljP cWb         ZyBEi PjLtGeKSZC lM fog DImddssVo Ik SKNrpqUo nqBYmCQu TGnhLpIEz O vlk ybC UaUhEg
       12 31-03-2024 Angelitos                                          OOQ30395Q845988          0         0 Esto es el t?tulo                                   This is the new text of the review

12 filas seleccionadas.
```

In this image, the top selection is the post, unedited, and the second one is after the update.

Comments

As you can see from the last view, we added an id to select the posts to make it easier for the user to select the post to delete, instead of making the user select the date and manually insert it, we thought this would be much easier to visually see and prone to fewer mistakes.

5 Explicitly required Triggers

We have also divided this section into the Description + SQL code + testing.

5.1 endorsed

Description

- ➢ Table Association: Posts
- ➢ Event that Triggers: Insertion/Update of a Post
- ➢ Temporality: Before.
- ➢ Granularity: For each row.
- ➢ Condition: Depending on if the client has already bought that reference before.
- ➢ Action: This gets the last date in which the client has ordered the product they are making the post about. If they have not bought the product, the date is NULL.

SQL Code

```
Unset
CREATE OR REPLACE TRIGGER endorsed
BEFORE INSERT or UPDATE ON Posts
FOR EACH ROW
BEGIN
	DECLARE purchased DATE := NULL;
	BEGIN
		SELECT max(a.ORDERDATE) INTO purchased
		FROM Orders_Clients a, Client_Lines b
		WHERE a.username = b.username AND a.orderdate = b.orderdate
		AND b.barcode = :new.barcode AND a.username = :new.username;
		IF purchased IS NOT NULL THEN
		   :new.endorsed := purchased;
		ELSE
		   :new.endorsed := NULL;
		END IF;
	END;
END endorsed;
```

Tests

To test this, we will add a post of a previous purchased item and one that has not been purchased. In the beginning, we counted 6 rows of 'endorsed' in the Posts table.

```
SQL> select distinct endorsed from posts;

ENDORSED
--------

09/07/23
06/07/20
15/04/21
12/09/17
01/07/14

6 filas seleccionadas.
```

```
SQL> CREATE OR REPLACE TRIGGER endorsed
  2  BEFORE INSERT or UPDATE ON Posts
  3  FOR EACH ROW
  4  BEGIN
  5  DECLARE purchased DATE := NULL;
  6  BEGIN
  7  SELECT max(a.ORDERDATE) INTO purchased
  8  FROM Orders_Clients a, Client_Lines b
  9  WHERE a.username = b.username AND a.orderdate = b.orderdate
 10  AND b.barcode = :new.barcode AND a.username = :new.username;
 11  IF purchased IS NOT NULL THEN
 12     :new.endorsed := purchased;
 13  ELSE
 14     :new.endorsed := NULL;
 15  END IF;
 16  END;
 17  END endorsed;
 18  /

Disparador creado.
```

We test if our trigger works correctly; for this, after making the update, we should obtain a new endorsed value. And the same should happen when making an insert:

```
Unset
UPDATE POSTS SET TITLE='ODlZQGmpil dKWzFPTZTa' WHERE username='feliko'
AND PRODUCT='Cocinero y diablo'
AND POSTDATE=TO_DATE('2018/07/26 22:50:03','yyyy/mm/dd hh24:mi:ss');
```

```
SQL> UPDATE POSTS SET TITLE='ODlZQGmpil dKWzFPTZTa' WHERE username='feliko' AND PRODUCT='Cocinero y diablo'
  2  AND POSTDATE=TO_DATE('2018/07/26 22:50:03','yyyy/mm/dd hh24:mi:ss');

1 fila actualizada.

SQL> select endorsed from posts where username='feliko' AND  POSTDATE=TO_DATE('2018/07/26 22:50:03','yyyy/mm/dd hh24:mi:ss');

ENDORSED
--------
26/06/18
```

Unset

```
INSERT INTO POSTS (USERNAME, POSTDATE, BARCODE, PRODUCT, SCORE, TITLE, TEXT,
LIKES, ENDORSED)
VALUES ('llerena',TO_DATE('2024/04/11 16:37:14','yyyy/mm/dd hh24:mi:ss'),
'OOQ997550384919','Gracia de correcto', 0, 'Titulo', 'Texto', 0,
TO_DATE('2020/09/26 16:11:43','yyyy/mm/dd hh24:mi:ss'));
```

```
SQL> INSERT INTO POSTS (USERNAME,POSTDATE,BARCODE,PRODUCT,SCORE,TITLE,TEXT,LIKES,ENDORSED) VALUES ('llerena',TO_DATE('2024/04/11 16:37:14','yyyy/mm/dd hh24:mi:ss'),
  2  'OOQ997550384919','Gracia de correcto',0,'Titulo','Texto',0,TO_DATE('2020/09/26 16:11:43','yyyy/mm/dd hh24:mi:ss'));

1 fila creada.

SQL>
SQL> select username, barcode, endorsed from posts where postdate = TO_DATE('2024/04/11 16:37:14','yyyy/mm/dd hh24:mi:ss') and product = 'Gracia de correcto';

USERNAME                       BARCODE         ENDORSED
------------------------------ --------------- --------
llerena                        OOQ997550384919 26/09/20
```

Result: We have two new endorsed values.

```
SQL> select distinct endorsed from posts;

ENDORSED
--------

09/07/23
26/09/20
06/07/20
15/04/21
12/09/17
26/06/18
01/07/14

8 filas seleccionadas.
```

5.2 remove_client

Description

This trigger is a core component of a database, specifically in this day and age, where GDPR laws exist. To enforce these directives, we need to delete the information related to a user.

- Table Association: Clients.
- Event that Triggers: Deletion of a Client.
- Temporality: Before.
- Granularity: For each row.
- Condition: It will always execute.
- Action: In this case, the purchases the user has made, the posts and the credit cards associated with them have to be deleted. To accomplish this, we "move" all the client_lines, client_orders and post to their respective anonymous table, where there is no identification of the username.

SQL Code

```
Unset
CREATE OR REPLACE TRIGGER remove_client
BEFORE DELETE ON Clients
FOR EACH ROW
DECLARE
-- We declare some variables for ease of use during the trigger
    v_username clients.username % TYPE := :OLD.username;
    v_contact clients.email % TYPE;
    v_contact_2 clients.mobile % TYPE := NULL;
    v_surn1 clients.surn1 % TYPE := :OLD.surn1;
    v_surn2 clients.surn2 % TYPE := :OLD.surn2;
    v_name clients.name % TYPE := :OLD.name;
BEGIN
     -- Selecting the main contact as the one that is not null and assigning
the second one, if it exists to the secondary contact
    v_contact := NVL(:OLD.email, TO_CHAR(:OLD.mobile));
    IF v_contact = :OLD.email THEN
        v_contact_2 := :OLD.mobile;
    ELSE
        v_contact_2 := NULL;
    END IF;
    -- Inserting to the Anonimized orders the orders clients
    INSERT INTO Orders_Anonym
    (
```

```
        orderdate,
        contact,
        contact2,
        dliv_datetime,
        name,
        surn1,
        surn2,
        bill_waytype,
        bill_wayname,
        bill_gate,
        bill_block,
        bill_stairw,
        bill_floor,
        bill_door,
        bill_ZIP,
        bill_town,
        bill_country,
        dliv_waytype,
        dliv_wayname,
        dliv_gate,
        dliv_block,
        dliv_stairw,
        dliv_floor,
        dliv_door,
        dliv_ZIP,
        dliv_town,
        dliv_country
    )(
        SELECT
            orderdate,
            v_contact,
            v_contact_2,
            oc.dliv_datetime,
            v_name,
            v_surn1,
            v_surn2,
            ca.waytype,
            ca.wayname,
            ca.gate,
            ca.block,
            ca.stairw,
```

```
            ca.floor,
            ca.door,
            ca.ZIP,
            oc.bill_town,
            oc.bill_country,
            ca.wayname,
            ca.wayname,
            ca.gate,
            ca.block,
            ca.stairw,
            ca.floor,
            ca.door,
            ca.ZIP,
            ca.town,
            ca.country
        FROM Orders_Clients oc
        JOIN Client_Addresses ca ON
            ((oc.username = ca.username) AND
            (oc.town = ca.town) AND
            (oc.country = ca.country))
             WHERE oc.username = v_username);
    -- Inserting each of the clients lines wihtout the username and mixing the
card-number if pressent
    INSERT INTO Lines_Anonym (
        orderdate,
        contact,
        dliv_town,
        dliv_country,
        barcode,
        price,
        quantity,
        pay_type,
        pay_datetime,
        card_comp,
        card_num,
        card_holder,
        card_expir
    )
    (SELECT
        cl.orderdate,
        v_contact,
```

```
        cl.town,
        cl.country,
        cl.barCode,
        cl.barCode,
        cl.quantity,
        cl.pay_type,
        cl.pay_datetime,
        cc.card_comp,
        cc.cardnum,
        cc.card_holder,
        cc.card_expir
    FROM Client_Lines cl
    JOIN Client_Cards cc ON
        (cl.cardnum = cc.cardnum AND
        cl.username = v_username));
      -- Removing the username from the posts and inserting them into the
anonyPosts
    INSERT INTO AnonyPosts (
        postdate,
        barCode,
        product,
        score,
        title,
        text,
        likes,
        endorsed
    )
    (SELECT
        postdate,
        barcode,
        product,
        score,
        title,
        text,
        likes,
        endorsed
    FROM Posts
    WHERE username = v_username);
     -- Deleting all the rows that contain the username we deleted, from which
we copies the information.
    DELETE FROM Orders_Clients WHERE username = v_username;
```

```
    DELETE FROM Client_Lines WHERE username = v_username;
    DELETE FROM Client_Addresses WHERE username = v_username;
    DELETE FROM Posts WHERE username = v_username;
    DELETE FROM Clients WHERE username = v_username;
END remove_client;
-- We now need to add a trigger for when an insertion in the AnonymPosts so
the date dos not collide.
```

To prevent the insertion of anonymous posts with the same postdate, we created a trigger that checks the postdate and if it already exists, it adds one second to prevent the collision.

Unset

```
CREATE OR REPLACE TRIGGER postdate_anonym_order
BEFORE INSERT ON AnonyPosts
FOR EACH ROW
DECLARE
    count_time NUMBER := 0;
    v_time AnonyPosts.postdate % TYPE;
BEGIN
    v_time := :NEW.postdate;
    -- Check if the postdate already exists in the table
     SELECT count(postdate) into count_time FROM AnonyPosts where postdate =
v_time;
    WHILE count_time > 0 LOOP
    -- Add 1 second to the postdate until it becomes unique
        v_time := v_time + INTERVAL '1' SECOND;
         SELECT count(postdate) into count_time FROM AnonyPosts where postdate
= v_time;
    END LOOP;
    -- Update the postdate with the unique timestamp
    :NEW.postdate := v_time;
    -- Display the updated timestamp (optional)
     DBMS_OUTPUT.PUT_LINE('Updated timestamp: ' || TO_CHAR(v_time, 'YYYY-MM-DD
HH24:MI:SS'));
END;
```

Tests

To check this second trigger is working we created this test, where we insert two posts with the same postdate, thus the trigger should add a second. And when selecting the postdate, the time difference should be 1 second.

```
SQL> insert into AnonyPosts (
  2      postdate,
  3      barCode,
  4      product,
  5      score,
  6      title,
  7      text,
  8      likes,
  9      endorsed
 10  ) VALUES (
 11      TO_DATE('01-01-2020', 'DD-MM-YYYY'),
 12      'OQI927570715165',
 13      'Radio',
 14      3,
 15      'Titulo',
 16      'Texto',
 17      0,
 18      Null
 19  );
Updated timestamp: 2020-01-01 00:00:00

1 fila creada.

SQL> CREATE OR REPLACE TRIGGER postdate_anonym_order
  2  BEFORE INSERT ON AnonyPosts
  3  FOR EACH ROW
  4  DECLARE
  5      count_time NUMBER := 0;
  6      v_time AnonyPosts.postdate % TYPE;
  7  BEGIN
  8      v_time := :NEW.postdate;
  9      DBMS_OUTPUT.PUT_LINE('Entro en el trigger');
 10      -- Check if the postdate already exists in the table
 11      SELECT count(postdate) into count_time FROM AnonyPosts where postdate = v_time;
 12      WHILE count_time > 0 LOOP
 13          -- Add 0.1 seconds to the postdate until it becomes unique
 14          v_time := v_time + INTERVAL '1' SECOND;
 15              DBMS_OUTPUT.PUT_LINE('Updated timestamp: ' || TO_CHAR(v_time, 'YYYY-MM-DD HH24:MI:SS'));
 16          SELECT count(postdate) into count_time FROM AnonyPosts where postdate = v_time;
 17      END LOOP;
 18      -- Update the postdate with the unique timestamp
 19      :NEW.postdate := v_time;
 20      -- Display the updated timestamp (optional)
 21      DBMS_OUTPUT.PUT_LINE('Updated timestamp: ' || TO_CHAR(v_time, 'YYYY-MM-DD HH24:MI:SS'));
 22  END;
 23  /

Disparador creado.

SQL> show errors;
No hay errores.
SQL> insert into AnonyPosts (
  2      postdate,
  3      barCode,
  4      product,
  5      score,
  6      title,
  7      text,
  8      likes,
  9      endorsed
 10  ) VALUES (
 11      TO_DATE('01-01-2020', 'DD-MM-YYYY'),
 12      'OQI927570715165',
 13      'Radio',
 14      3,
 15      'Titulo',
 16      'Texto',
 17      0,
 18      Null
 19  );
Entro en el trigger
Updated timestamp: 2020-01-01 00:00:01
Updated timestamp: 2020-01-01 00:00:01

1 fila creada.
```

Result:

```
SQL> select TO_CHAR(postdate, 'YYYY-MM-DD HH24:MI:SS') from AnonyPosts;

TO_CHAR(POSTDATE,'Y
-------------------
2020-01-01 00:00:00
2020-01-01 00:00:01
```

The two Anonymous posts have different postdates even though the insertion has the same date.

5.3 anonymous purchase

Description

This PL/SQL code is simpler than what we initially wanted to do. We wanted to insert the row and then the trigger would be of the type after insert and it would delete the row, but we encountered an error of mutating table. Thus, we decided to change the trigger to a before insert and raise an application error and stop the execution completely.

- Table Association: Lines_Anonym.
- Event that Triggers: Insertion of a line.
- Temporality: Before.
- Granularity: For each row.
- Condition: If the credit card is present in the table Client_Cards
- Action: It raises an application error if the credit card being added is already inside the client_cards table.

SQL Code

```
Unset
CREATE OR REPLACE TRIGGER prev_registered_credit_card
BEFORE INSERT ON Lines_Anonym
FOR EACH ROW
DECLARE v_count_orders NUMBER;
BEGIN
   SELECT count('x') INTO v_count_orders FROM Client_Cards WHERE cardnum =
:new.card_num;
   IF v_count_orders > 0 THEN
     RAISE_APPLICATION_ERROR(-20002, 'You can not add a credit card to an
anonymous purchase that is already registered');
    ELSE NULL;
    END IF;
END;
```

Tests

To check the validity of this trigger, we must try and insert an order with the same credit card. To accomplish this, we will try and run the following query, that inserts an order that :

```
Unset
INSERT INTO Lines_Anonym (
    orderdate,
    contact,
    dliv_town,
    dliv_country,
    barcode,
    price,
    quantity,
    pay_type,
    pay_datetime,
    card_comp,
    card_num,
    card_holder,
    card_expir
) VALUES
    (SYSDATE,
    'eee@eee.com',
    'Madrid',
    'Madrid',
    'OQI927570715165',
    5,
    5,
    'CREDIT CARD',
    TO_DATE('01/03/22', 'DD/MM/YY'),
    'Pai Pai',
    532143944287,
    'Inmaculada Gomez',
    TO_DATE('01-03-25', 'DD-MM-YY'));
```

And the result of this query should be the trigger warning us that this action could not be performed and thus, it will not be inserted. (Which in hindsight means it has been inserted and then deleted. We can see it has not been inserted by checking the count of the rows:

```
SQL> select count('x') from lines_anonym;

COUNT('X')
----------
      3537

SQL> INSERT INTO Lines_Anonym VALUES
  2      (SYSDATE,
  3      'eee@eee.com',
  4      'Madrid',
  5      'Madrid',
  6      'OQI927570715165',
  7      5,
  8      5,
  9      'CREDIT CARD',
 10      TO_DATE('01/03/22', 'DD/MM/YY'),
 11      'Pai Pai',
 12      532143944287,
 13      'Inmaculada Gomez',
 14      TO_DATE('01-03-25', 'DD-MM-YY'));
    'Madrid',
             *
ERROR at line 4:
ORA-20002: You can not add a credit card to an anonymous purchase that is already registered
ORA-06512: at "FSDB279.PREV_REGISTERED_CREDIT_CARD", line 8
ORA-04088: error during execution of trigger 'FSDB279.PREV_REGISTERED_CREDIT_CARD'


SQL> select count('x') from lines_anonym;

COUNT('X')
----------
      3537

SQL>
```

5.4 UPDATE stocks

Description

- Table Association: Lines_Anonym and Client_Lines.
- Event that Triggers: Insertion of a Client.
- Temporality: After.
- Granularity: For each row.
- Condition: Depending on the remaining stock of that reference.
- Action: This first one gets the order quantity and checks if the quantity of that reference is higher than what is asked for. If it is, it subtracts the demanded amount from the stock. But if the quantity is less, we change the order to the remaining quantity and change the stock to 0. Then, we check if there is not already a replacement order. If there isn't one, we place one.

SQL Code

```
Unset
CREATE OR REPLACE TRIGGER UPDATE_stocks_anonym
AFTER INSERT ON Lines_Anonym
FOR each ROW
DECLARE
    v_stock references.cur_stock % TYPE;
    v_min_stock references.min_stock % TYPE;
    v_max_stock references.max_stock % TYPE;
    v_replacement_order_count NUMBER;
    v_supply_lines_count NUMBER;
    v_supplier providers.taxID % TYPE;
    v_supplier_cost supply_lines.cost % TYPE;
BEGIN
    SELECT cur_stock, min_stock
    INTO v_stock, v_min_stock
    FROM references
    WHERE barcode = :new.barcode;
    IF v_stock - :new.quantity > v_min_stock then
        UPDATE references
            SET cur_stock = v_stock - :new.quantity
            WHERE barcode = :new.barcode;
    ELSIF v_stock < :new.quantity then
        UPDATE references
            SET cur_stock = 0
            WHERE barcode = :new.barCode;
        UPDATE Lines_Anonym
            SET quantity = v_stock
            WHERE barCode = :new.barcode AND
                  contact = :new.contact AND
                  orderdate = :new.orderdate;
        DBMS_OUTPUT.PUT_LINE('Stock is less than the order quantity');
        SELECT count('x')
            INTO v_replacement_order_count
        FROM replacements
            WHERE barCode = :new.barCode AND status = 'D';
        IF v_replacement_order_count = 0 THEN
            SELECT taxID, min(cost)
                INTO v_supplier, v_supplier_cost
            FROM supply_lines
                WHERE barcode = :new.barcode
                GROUP BY taxID
```

```
                HAVING cost=min(cost);
                 INSERT INTO replacements (orderdate, barCode, taxID, status,
units, payment)
               VALUES (SYSDATE, :new.barcode, v_supplier, 'D', v_max_stock/2,
v_max_stock/2 * v_supplier_cost);
             END IF;

        SELECT count('x')
            INTO v_supply_lines_count
            FROM supply_lines
            WHERE barCode = :new.barCode;

        IF v_supply_lines_count = 0 THEN
            INSERT INTO supply_lines (barCode) VALUES (:new.barcode);
        END IF;
    END IF;
END;
```

In this second trigger, we do the exact same thing but for the table of registered clients.

```
Unset
CREATE OR REPLACE TRIGGER UPDATE_stocks_reg
AFTER INSERT ON Client_Lines
FOR each ROW
DECLARE
    v_stock references.cur_stock % TYPE;
    v_min_stock references.min_stock % TYPE;
    v_max_stock references.max_stock % TYPE;
    v_replacement_order_count NUMBER;
    v_supplier providers.taxID % TYPE;
    v_supplier_cost supply_lines.cost % TYPE;
BEGIN
    SELECT cur_stock, min_stock
    INTO v_stock, v_min_stock
    FROM references
    WHERE barcode = :new.barcode;
    IF v_stock - :new.quantity > v_min_stock then
        UPDATE references
            SET cur_stock = v_stock - :new.quantity
            WHERE barcode = :new.barcode;
```

```
    ELSIF v_stock < :new.quantity then
        UPDATE references
            SET cur_stock = 0
            WHERE barcode = :new.barCode;
        UPDATE Client_Lines
            SET quantity = v_stock
            WHERE barCode = :new.barcode AND
                  username = :new.username AND
                  orderdate = :new.orderdate;
        DBMS_OUTPUT.PUT_LINE('Stock is less than the order quantity');
        SELECT count('x')
            INTO v_replacement_order_count
        FROM replacements
            WHERE barCode = :new.barCode AND status = 'D';
        IF v_replacement_order_count = 0 THEN
            SELECT taxID, min(cost)
                INTO v_supplier, v_supplier_cost
            FROM supply_lines
                WHERE barcode = :new.barcode
                GROUP BY taxID
                HAVING cost=min(cost);
                 INSERT INTO replacements (orderdate, barCode, taxID, status,
units, payment)
               VALUES (SYSDATE, :new.barcode, v_supplier, 'D', v_max_stock/2,
v_max_stock/2 * v_supplier_cost);
             END IF;

        SELECT count('x')
            INTO v_supply_lines_count
            FROM supply_lines
            WHERE barCode = :new.barCode;

        IF v_supply_lines_count = 0 THEN
            INSERT INTO supply_lines (barCode) VALUES (:new.barcode);
        END IF;
    END IF;
END;
```

Tests

We have modified the tables slightly so the test we wanted to run could be performed faster. We changed the maximum order quantity from a NUMBER(2) to a NUMBER(10) to allow us to do a purchase order of 9999 units. The objective was checking that there is an insertion in the table replacements with a supplier. Even though during class the contrary was discussed, that this trigger should leave the provider empty, the taxID, as the implementation of the table has as the primary key the attribute taxID, it has to be filled.

```
Unset
DELETE FROM client_lines WHERE orderdate=TO_DATE('2014-03-09 07:17:49',
'YYYY-MM-DD HH24:MI:SS') AND barcode='QIQ43968I541183';
INSERT INTO CLIENT_LINES VALUES(
    TO_DATE('2014-03-09 07:17:49', 'YYYY-MM-DD HH24:MI:SS'),
    'narvaez',
    'Valverde de la Alameda',
    'Spain',
    'QIQ43968I541183',
    2.8,
    9999,
    'CREDIT CARD',
    TO_DATE('06/09/23', 'DD/MM/YY'),
    904411953814
);
```

From this insertion, we should expect the stock to be 0 and there to be a new row in the table "replacements":

```
SQL> DELETE FROM client_lines WHERE orderdate=TO_DATE('2014-03-09 07:17:49', 'YYYY-MM-DD HH24:MI:SS')
  2   AND barcode='QIQ43968I541183';

0 filas suprimidas.

SQL> INSERT INTO CLIENT_LINES VALUES(
  2      TO_DATE('2014-03-09 07:17:49', 'YYYY-MM-DD HH24:MI:SS'),
  3      'narvaez',
  4      'Valverde de la Alameda',
  5      'Spain',
  6      'QIQ43968I541183',
  7      2.8,
  8      9999,
  9      'CREDIT CARD',
 10      TO_DATE('06/09/23', 'DD/MM/YY'),
 11      904411953814
 12  );

1 fila creada.

SQL> select count(*) from replacements
  2  ;

  COUNT(*)
----------
         1
```

As we can see, there is a new row (the table was empty before, as per the creation script leaves it empty)

6 Concluding Remarks

Regarding the time taken to complete this practice was a bit long. We started working on it during the spring break. Adding up all the hours spent, we will have worked on this for more than 60 + I + S. In the final weeks, as there were other assignments due the same day, the time was considerably the resource that was lacking the most.

It is true, we have learnt alot from doing this project, by having the freedom of tinkering around with the database and having the freedom of the implementation, we could test different paradigms.

Improvements for further editions: try to not use AulaVirtual. For example, what does ctrl+c do? It sometimes stops the query but other times, it closes the sqlplus interrogator. The naming of the tables with respect to the provided code could be fixed. If this is not feasible, the raw document could be sent to the students so they could edit it to fix the column names.