RELACIONES DE EJERCICIOS DE QUERIES

Contenido

1 EJERCICIOS INTRODUCTORIOS

1. Full name of those customers belonging to the country identified by the id 52778. 2039 rows

SELECT CUST_FIRST_NAME, CUST_LAST_NAME FROM CUSTOMERS WHERE COUNTRY_ID=52778;

2. Full name and descriptions of the products whose category is not 205 (the largest category). 50 rows. Notice that both attributes have the same content.

SELECT PROD_NAME, PROD_DESC FROM PRODUCTS WHERE NOT (PROD_CATEGORY_ID = 205);

3. Full name of the customers having an email address in the server "apple.com". The mail can be written in uppercase or lowercase.1897 rows

SELECT CUST_FIRST_NAME, CUST_LAST_NAME FROM CUSTOMERS WHERE CUST_EMAIL LIKE '%@apple.com' OR CUST_EMAIL LIKE '%@APPLE.COM';

4. Full name of those customers that have not provided their marital status. 17428 rows

SELECT CUST_FIRST_NAME, CUST_LAST_NAME FROM CUSTOMERS WHERE CUST_MARITAL_STATUS IS NULL

Write the query using the IS NULL operator and using = NULL. Do they have the same behaviour?

=NULL selecciona 0 elementos.

5. Show the sales of the product 45. Show the list chronologically ordered by date and also by the customer code (ascending). 10742

SELECT * FROM SALES WHERE PROD_ID=45 ORDER BY TIME_ID ASC, CUST_ID ASC;

6. Products in the category 205. Provide their names and the proportion between the catalog price and the minimum price (provide the data in percentage, with the % at the end to indicate it).

SELECT PROD_NAME, CONCAT(ROUND(PROD_MIN_PRICE / PROD_LIST_PRICE, 4) *100, '%') PERCENTAGE FROM PRODUCTS WHERE PROD_CATEGORY_ID=205;

SELECT PROD_NAME, TO_CHAR(ROUND(PROD_MIN_PRICE / PROD_LIST_PRICE, 2) *100 || '%') PERCENTAGE FROM PRODUCTS WHERE PROD_CATEGORY_ID=205;

2 EJERCICIOS DE FUNCIONES

2.1 SOBRE FECHAS

1. List the information of all the sales made in the last century (we consider that the current century began on Jan 1st 2000). SH.V_FUN_1

```
SELECT * FROM SALES WHERE ((TO_CHAR(TIME_ID, 'YYYY')<2000) AND (TO_CHAR(TIME_ID, 'YYYY')>1899));
```

2. Information of the costs that was stored in the database at most 19 years ago. SH.V_FUN_2.

```
SELECT * FROM COSTS WHERE ABS((MONTHS_BETWEEN(SYSDATE,TIME_ID))/12)<19;
```

3. Get a list with the id of the customer and the id of the product sold and the date of the sale. Show the amount of three-years period spent from that day up to now. Name this column TRIENNIUM. SH.V_FUN_3

```
SELECT CUST_ID, PROD_ID, TIME_ID, FLOOR((MONTHS_BETWEEN(SYSDATE,TIME_ID)/36))
FROM SALES;
```

4. Show the name of all the products containing the (verbatim) tag "Phone". In the list, substitute such tag by "Smartphone". SH.V_FUN_4

```
SELECT REPLACE(PROD_NAME, 'Phone', 'Smartphone') FROM PRODUCTS;
```

5. Get a list of the customers that didn't submit their marital status. You have to inform about this situation and the name of the customer, following the schema: "The customer .....(full name of the customer) didn't provide .....(his or her) marital status. You can use the concat function (||) and also the DECODE function. The scape code for the symbol ' in Oracle is ''. SH.V_FUN_5

```
SELECT DISTINCT TO_CHAR('The customer ' ||CUST_FIRST_NAME || ' ' || CUST_LAST_NAME
|| ' didn''t provide '

|| CASE WHEN CUST_GENDER='M' THEN 'his' END

|| CASE WHEN CUST_GENDER='F' THEN 'her' END

|| CASE WHEN CUST_GENDER!='F' AND CUST_GENDER!='M' THEN 'their' END

||' marital status') "NoN submitted" FROM CUSTOMERS WHERE CUST_MARITAL_STATUS IS
NULL;
```

6. Show the information of the customers (get the first and last name, ID and year of birth) living in the cities Aachen, Aalborg or Aalen. Those customers with no year of birth provided will appear with the label "NON SUBMITTED". Notice that the NVL function must have all its parameters of the same data_type.

```
SELECT CUST_FIRST_NAME, CUST_LAST_NAME, CUST_ID,
NVL(TO_CHAR(CUST_YEAR_OF_BIRTH), 'NON SUBMITTED') "CUST_YEAR_OF_BIRTH"

FROM CUSTOMERS

WHERE CUST_CITY='Aachen' OR CUST_CITY='Aalborg' OR CUST_CITY='Aalen';
```

Can you simplify your query by using the property that these three cities begin with "Aa" (they are no more cities fulfilling this property). SH.V_FUN_6

SELECT CUST_FIRST_NAME, CUST_LAST_NAME, CUST_ID,
NVL(TO_CHAR(CUST_YEAR_OF_BIRTH), 'NON SUBMITTED') "CUST_YEAR_OF_BIRTH"

FROM CUSTOMERS

WHERE CUST_CITY LIKE 'Aa%';

7. List the information of the sales made on the second Monday of each month. Provide the ID of the customer, ID of the product and date. Take into account that the DAY token in the TO_CHAR function for the dates can provide some extra spaces because it renders a fixed size output. SH.V_FUN_7

The first week of the month are the first days if the other month did not finish on Sunday and the next week, the second Mondays will always be on Week 2, day of the week 1.

SELECT CUST_ID, PROD_ID, TIME_ID FROM SALES WHERE TO_CHAR(TIME_ID, 'W')=2 AND TO_CHAR(TIME_ID, 'D')=1;

2.2 JOIN I

1. Provide a list of all the sales from customers born on 1990. Provide the amount, the product id and the customer id. SH.V_JOIN_1

SELECT QUANTITY_SOLD, CUST_ID, PROD_ID FROM SALES JOIN CUSTOMERS USING (CUST_ID) WHERE CUST_YEAR_OF_BIRTH=1990;

2. List the sales made on Monday. Do not use the TO_CHAR function to get the day. Use the TIMES table where all date are categorized. Provide the same attributes that appears in the previous query. Order it by the amount of the sale. SH.V_JOIN_2

SELECT QUANTITY_SOLD, CUST_ID, PROD_ID FROM SALES JOIN TIMES USING (TIME_ID) WHERE DAY_NUMBER_IN_WEEK=1;

3. List the CUST_ID and the full name of the persons who spend more than 1500€ in a single sale. Do you believe that the DISTINCT clause is needed in this query? SH.V_JOIN_3

SELECT DISTINCT CUST_ID, CUST_FIRST_NAME, CUST_LAST_NAME FROM CUSTOMERS JOIN SALES USING (CUST_ID) WHERE AMOUNT_SOLD>1500;

DISTINCT is needed so a person who fulfils the condition twice, only appears once.

4. At what dates we have sale something using the Digital Marketing channel? (described with the Internet key word) Note that repeat information may appear. Avoid redundant information in the output. SH.V_JOIN_4

SELECT DISTINCT TIME_ID FROM SALES SA JOIN CHANNELS CH ON (SA.CHANNEL_ID=CH.CHANNEL_ID) WHERE UPPER(CHANNEL_DESC) LIKE '%INTERNET%';

SELECT DISTINCT TIME_ID FROM SALES JOIN CHANNELS USING (CHANNEL_ID) WHERE UPPER(CHANNEL_DESC) LIKE '%INTERNET%';

5. List the customer living in Singapore. Provide first name and last name. SH.V_JOIN_5

SELECT DISTINCT CUST_FIRST_NAME, CUST_LAST_NAME, CUST_ID FROM CUSTOMERS JOIN COUNTRIES USING (COUNTRY_ID) WHERE COUNTRY_NAME='Singapore';

6. List the full name of the persons born in 1990 and spending more than 500€ in one single sale. Do not remove repeated customers names.SH.V_JOIN_6

SELECT CUST_FIRST_NAME, CUST_LAST_NAME FROM CUSTOMERS JOIN SALES USING (CUST_ID) WHERE CUST_YEAR_OF_BIRTH = 1990 AND AMOUNT_SOLD>500;

7. List the name of the successful promotions, i.e. those ones that get at least one sale. Provide promotion name and id. SH.V_JOIN_7

SELECT DISTINCT PROMO_ID, PROMO_NAME FROM PROMOTIONS JOIN SALES USING (PROMO_ID);

3 EJERCICIOS DE CONJUNTOS Y UNIONES

3.1 SET

1. List all the tags that appears in the columns EDUCATION, OCCUPATION or MARITAL_STATUS (they belongs to two different tables). Provide the output in two columns, where the first one is the tag and the second one is the type (EDUCATION, OCCUPATION or MARITAL_STATUS) [SH.V_SETOP_1]

SELECT DISTINCT CUST_MARITAL_STATUS "CODE", 'MARITAL_STATUS' "TYPE" FROM CUSTOMERS

UNION

SELECT DISTINCT OCCUPATION, 'OCCUPATION' FROM SUPPLEMENTARY_DEMOGRAPHICS

UNION

SELECT DISTINCT EDUCATION, 'EDUCATION' FROM SUPPLEMENTARY_DEMOGRAPHICS;

2. List the PROD_ID of the products that haven't ever been purchased by any customer [SH.V_SETOP_2]

SELECT DISTINCT PROD_ID FROM PRODUCTS

MINUS

SELECT DISTINCT PROD_ID FROM SALES;

3. List the CUST_ID of the customers having extra information in the SUPPLEMENTARY_DEMOGRAPHICS table [SH.V_SETOP_3]

SELECT DISTINCT CUST_ID FROM CUSTOMERS

MINUS

SELECT DISTINCT CUST_ID FROM SUPPLEMENTARY_DEMOGRAPHICS;

4. List the descriptions of the products appearing exactly in the same way as a PROD_DESC, a PROD_CAT_DESC and a PROD_SUBCAT_DESC [SH.V_SETOP_4]

SELECT PROD_NAME FROM PRODUCTS

WHERE PROD_NAME=PROD_SUBCATEGORY_DESC AND
PROD_CATEGORY_DESC=PROD_SUBCATEGORY_DESC;

Can be done with intersect.

SELECT PROD_DESC FROM PRODUCTS

INTERSECT

SELECT PROD_SUBCATEGORY_DESC FROM PRODUCTS

INTERSECT

SELECT PROD_CATEGORY_DESC FROM PRODUCTS;

5. List the PROD_ID and PROD_NAME of the products having the same NAME and DESC [SH.V_SETOP_5]

SELECT PROD_ID, PROD_NAME FROM PRODUCTS

INTERSECT

SELECT PROD_ID, PROD_DESC FROM PRODUCTS;

3.2 JOIN II

SELECT DISTINCT has to be used when repeated information appears.

1. List the name of the products sold by using a Tele-sale channel. [SH.V_MULJOIN_1]

SELECT DISTINCT PROD_NAME FROM SALES JOIN PRODUCTS USING (PROD_ID) JOIN CHANNELS USING(CHANNEL_ID) WHERE CHANNEL_ID=9;

2. List the name of the products bought by customers living in Middle East. [SH.V_MULJOIN_2]

SELECT DISTINCT PROD_NAME FROM SALES JOIN PRODUCTS USING (PROD_ID) JOIN CUSTOMERS USING(CUST_ID) JOIN COUNTRIES USING(COUNTRY_ID) WHERE COUNTRY_SUBREGION_ID=52796;

3. Provide the full name of the customers with a credit limit of 11000€ that have bought by using a digital marketing channel some product costing a big deal (more than 1000€). The product cost has to be checked with the PROD_LIST_PRICE attribute of the PRODUCTS table. [SH.V_MULJOIN_3]

SELECT DISTINCT CUST_FIRST_NAME, CUST_LAST_NAME FROM SALES JOIN PRODUCTS USING (PROD_ID) JOIN CUSTOMERS USING(CUST_ID) JOIN CHANNELS USING(CHANNEL_ID) WHERE CHANNEL_ID=4 AND CUST_CREDIT_LIMIT<=11000 AND PROD_LIST_PRICE>1000;

4. List the name of the products bought on a leap year by some customer with a 15000€ credit limit. The product cannot have a list-price greater than 30. [SH.V_MULJOIN_4]

Leap Year:

LAST_DAY(TO_DATE(TO_CHAR('28-02-' || TO_CHAR(SYSDATE, 'YYYY')), 'DD/MM/YYYY'))!= TO_DATE(TO_CHAR('28-02-' || TO_CHAR(SYSDATE, 'YYYY')), 'DD/MM/YYYY')

SELECT DISTINCT PROD_NAME FROM SALES JOIN PRODUCTS USING (PROD_ID) JOIN CUSTOMERS USING(CUST_ID) WHERE CUST_CREDIT_LIMIT=15000 AND PROD_LIST_PRICE>30 AND (LAST_DAY(TO_DATE(TO_CHAR('28-02-' || TO_CHAR(TIME_ID, 'YYYY')), 'DD/MM/YYYY'))!= TO_DATE(TO_CHAR('28-02-' || TO_CHAR(TIME_ID, 'YYYY')), 'DD/MM/YYYY'));

5. A non-foresight customer is the one who has to go out on Christmas eve to buy in a direct sale a camera battery. Search the year of birth of the non-foresight customers. [SH.V_MULJOIN_5]

SELECT DISTINCT CUST_YEAR_OF_BIRTH FROM SALES JOIN PRODUCTS USING (PROD_ID) JOIN CUSTOMERS USING(CUST_ID) JOIN CHANNELS USING(CHANNEL_ID) WHERE CHANNEL_CLASS_ID=12 AND PROD_SUBCATEGORY_ID=2042 AND TO_CHAR(TIME_ID, 'DD/MM')='24/12';

3.3 JOIN REFLEXIVO

1. List the full name and ID of the customers having another soul-mate in the customer table; i.e. there exists another customer living in the same city, who also was born in their year of birth, has its gender, the same income level, marital status and credit limit. Even more, both are categorized as Active or Inactive in the CUST_VALID attribute. [V_SELF_1]

SELECT DISTINCT C1.CUST_FIRST_NAME, C1.CUST_LAST_NAME, C1.CUST_ID

FROM CUSTOMERS C1 JOIN CUSTOMERS C2 ON

(C1.CUST_CITY_ID=C2.CUST_CITY_ID AND
C1.CUST_YEAR_OF_BIRTH=C2.CUST_YEAR_OF_BIRTH AND
C1.CUST_GENDER=C2.CUST_GENDER AND C1.CUST_INCOME_LEVEL=C2.CUST_INCOME_LEVEL

AND C1.CUST_MARITAL_STATUS=C2.CUST_MARITAL_STATUS AND
C1.CUST_CREDIT_LIMIT=C2.CUST_CREDIT_LIMIT AND C1.CUST_VALID=C2.CUST_VALID)

WHERE C1.CUST_ID<>C2.CUST_ID;

2. Make the same query but listing pairs of customers. [V_SELF_2]

SELECT C1.CUST_FIRST_NAME, C1.CUST_LAST_NAME, C1.CUST_ID, C2.CUST_FIRST_NAME, C2.CUST_LAST_NAME, C2.CUST_ID

FROM CUSTOMERS C1 JOIN CUSTOMERS C2 ON

(C1.CUST_CITY_ID=C2.CUST_CITY_ID AND
C1.CUST_YEAR_OF_BIRTH=C2.CUST_YEAR_OF_BIRTH AND
C1.CUST_GENDER=C2.CUST_GENDER AND C1.CUST_INCOME_LEVEL=C2.CUST_INCOME_LEVEL

AND C1.CUST_MARITAL_STATUS=C2.CUST_MARITAL_STATUS AND
C1.CUST_CREDIT_LIMIT=C2.CUST_CREDIT_LIMIT AND C1.CUST_VALID=C2.CUST_VALID)
WHERE C1.CUST_ID<C2.CUST_ID;

3. List the name and ID of the products having another one in its subcategory and so that both of them have the same list price and min price. [V_SELF_3]

SELECT DISTINCT P1.PROD_ID, P1.PROD_NAME

FROM PRODUCTS P1 JOIN PRODUCTS P2 ON

(P1.PROD_SUBCATEGORY_ID=P2.PROD_SUBCATEGORY_ID AND
P1.PROD_LIST_PRICE=P2.PROD_LIST_PRICE AND P1.PROD_MIN_PRICE=P2.PROD_MIN_PRICE)
WHERE P1.PROD_ID<>P2.PROD_ID;

4. Make the same report but listing pairs of products. [V_SELF_4].

SELECT P1.PROD_ID, P1.PROD_NAME, P2.PROD_ID, P2.PROD_NAME

FROM PRODUCTS P1 JOIN PRODUCTS P2 ON

(P1.PROD_SUBCATEGORY_ID=P2.PROD_SUBCATEGORY_ID AND
P1.PROD_LIST_PRICE=P2.PROD_LIST_PRICE AND P1.PROD_MIN_PRICE=P2.PROD_MIN_PRICE)
WHERE P1.PROD_ID<P2.PROD_ID;

5. List the full name and ID of the customers who has bought one product that have also been bought by another soul-mate customer. [V_SELF_5]

SELECT DISTINCT C1.CUST_FIRST_NAME, C1.CUST_LAST_NAME, C1.CUST_ID

FROM CUSTOMERS C1 JOIN CUSTOMERS C2 ON

(C1.CUST_CITY_ID=C2.CUST_CITY_ID AND
C1.CUST_YEAR_OF_BIRTH=C2.CUST_YEAR_OF_BIRTH AND
C1.CUST_GENDER=C2.CUST_GENDER AND C1.CUST_INCOME_LEVEL=C2.CUST_INCOME_LEVEL

AND C1.CUST_MARITAL_STATUS=C2.CUST_MARITAL_STATUS AND
C1.CUST_CREDIT_LIMIT=C2.CUST_CREDIT_LIMIT AND C1.CUST_VALID=C2.CUST_VALID)

JOIN SALES S1 ON (C1.CUST_ID=S1.CUST_ID) JOIN SALES S2 ON (C2.CUST_ID=S2.CUST_ID)
WHERE C1.CUST_ID<>C2.CUST_ID AND S1.PROD_ID=S2.PROD_ID;

6. List the name and ID of the products successfully sold although there are other product in its same subcategory with minor price. [V_SELF_6]

SELECT DISTINCT P1.PROD_ID, P1.PROD_NAME

FROM PRODUCTS P1 JOIN PRODUCTS P2 ON
(P1.PROD_SUBCATEGORY_ID=P2.PROD_SUBCATEGORY_ID) JOIN SALES S1 ON
(P1.PROD_ID=S1.PROD_ID)

WHERE P1.PROD_ID<>P2.PROD_ID AND P1.PROD_LIST_PRICE>P2.PROD_LIST_PRICE;

4 EJERCICIOS SUBQUERIES, NEGATION Y GROUP BY

4.1 SUBQUERIES

1. Name of the products where the difference between their cost and their price is more than 600 and such that it have been sold in a sale where the amount_sold value is greater than 1780. [V_SUBQ_1]

SELECT PROD_NAME FROM PRODUCTS

WHERE PROD_ID IN

(SELECT PROD_ID FROM PRODUCTS JOIN COSTS USING (PROD_ID) JOIN SALES USING (PROD_ID) WHERE (UNIT_PRICE-UNIT_COST)>600 AND AMOUNT_SOLD>1780);

2. Full name and customer id of the customers from USA who bought some product listed in the above query and their salary is greater than 200.000. [V_SUBQ_2]

SELECT DISTINCT CUST_ID, CUST_FIRST_NAME, CUST_LAST_NAME FROM CUSTOMERS JOIN SALES USING (CUST_ID)

WHERE PROD_ID IN

(SELECT DISTINCT PROD_ID FROM SALES NATURAL JOIN COSTS JOIN PRODUCTS USING (PROD_ID) WHERE (UNIT_PRICE-UNIT_COST)>600 AND AMOUNT_SOLD>1780) AND COUNTRY_ID = 52790

AND SUBSTR(CUST_INCOME_LEVEL, 0, 1)='J' OR SUBSTR(CUST_INCOME_LEVEL, 0, 1)='K' OR SUBSTR(CUST_INCOME_LEVEL, 0, 1)='L';

3. List the name of the products bought in USA on 1998 Christmas holidays. Notice that Christmas in USA coincides with the last two weeks of the year. [V_SUBQ_3]

SELECT DISTINCT PROD_NAME

FROM PRODUCTS

WHERE PROD_ID IN (SELECT PROD_ID FROM SALES JOIN CUSTOMERS USING (CUST_ID) WHERE TO_CHAR(TIME_ID, 'YYYY') = '1998' AND TO_CHAR(TIME_ID, 'WW') IN ('52','53') AND COUNTRY_ID = 52790);

4. List the same information of the above query but translating it to Spain. In Spain, as you know, Christmas holidays also includes the first week of the next year. [V_SUBQ_4]

SELECT DISTINCT PROD_NAME

FROM PRODUCTS

WHERE PROD_ID IN

(SELECT PROD_ID

FROM SALES NATURAL JOIN TIMES JOIN CUSTOMERS USING (CUST_ID)

WHERE (CALENDAR_YEAR = '1999' AND CALENDAR_WEEK_NUMBER=1 AND COUNTRY_ID = 52778)

OR

(CALENDAR_YEAR = '1998' AND CALENDAR_WEEK_NUMBER IN (52, 53) AND COUNTRY_ID = 52778));

5. How can you get all these sales in a single query? [V_SUBQ_5]

SELECT DISTINCT PROD_NAME

FROM PRODUCTS

```
WHERE PROD_ID IN (SELECT PROD_ID FROM SALES JOIN CUSTOMERS USING (CUST_ID)
WHERE TO_CHAR(TIME_ID, 'YYYY') = '1998' AND TO_CHAR(TIME_ID, 'WW') IN ('52','53') AND
COUNTRY_ID = 52790)

UNION

SELECT DISTINCT PROD_NAME

FROM PRODUCTS

WHERE PROD_ID IN

  (SELECT PROD_ID

  FROM SALES NATURAL JOIN TIMES JOIN CUSTOMERS USING (CUST_ID)

  WHERE (CALENDAR_YEAR = '1999' AND CALENDAR_WEEK_NUMBER=1 AND COUNTRY_ID =
52778)

  OR

  (CALENDAR_YEAR = '1998' AND CALENDAR_WEEK_NUMBER IN (52, 53) AND COUNTRY_ID =
52778));
```

6. List the customer full name and year of birth of the customers who have bought a product of the Hardware category [V_SUBQ_6]

```
SELECT DISTINCT CUST_FIRST_NAME, CUST_LAST_NAME, CUST_YEAR_OF_BIRTH FROM
CUSTOMERS

WHERE CUST_ID IN (SELECT CUST_ID FROM SALES JOIN PRODUCTS USING (PROD_ID) WHERE
PROD_CATEGORY_ID=202);
```

7. List the time ID of the sales where the customer was some of the previous customers. [V_SUBQ_7]

```
SELECT DISTINCT TIME_ID FROM SALES

WHERE CUST_ID IN (SELECT CUST_ID FROM SALES JOIN PRODUCTS USING (PROD_ID) WHERE
PROD_CATEGORY_ID=202);
```

8. Observe that in the previous query we are not searching the date of the product of the Hardware category sales, but the TIME ID of any sale of any of the customers in the previous exercise. Is there any difference? [V_SUBQ_8]

In the second one we are searching from the customers who have bought and in the other of all the customers.

4.2 NEGATION

1. List the name of the products which are not in the PHOTO category. [V_NEG_1]

```
SELECT PROD_NAME FROM PRODUCTS WHERE PROD_CATEGORY_ID<>204;
```

2. List the full name of all customers not living in USA. [V_NEG_2]

```
SELECT DISTINCT CUST_FIRST_NAME, CUST_LAST_NAME FROM CUSTOMERS WHERE
COUNTRY_ID <>52790;
```

3. List the name of the customers who have bought nothing belonging to the PHOTO category. Notice that if a customer bought a Camera and a printer, he has actually bought a Photo-product. [V_NEG_3]

SELECT CUST_ID, CUST_FIRST_NAME, CUST_LAST_NAME

FROM CUSTOMERS

WHERE CUST_ID NOT IN (SELECT DISTINCT CUST_ID FROM SALES WHERE CHANNEL_ID=4);

4. List the cust_id and the full name of the customers who never use an internet marketing channel. [V_NEG_4]

SELECT DISTINCT CUST_FIRST_NAME, CUST_LAST_NAME

FROM CUSTOMERS

WHERE CUST_ID NOT IN (SELECT CUST_ID FROM CHANNELS JOIN SALES USING (CHANNEL_ID) WHERE CHANNEL_ID=4);

5. A thrifty customer is the one who never spend more than 1780 in the sale amount. List the name of the countries where none of their customers are thrifty. Notice that this query hide a double negation. [V_NEG_5]

SELECT COUNTRY_NAME FROM COUNTRIES WHERE COUNTRY_ID NOT IN

(SELECT DISTINCT COUNTRY_ID FROM CUSTOMERS

WHERE CUST_ID NOT IN (SELECT CUST_ID FROM SALES JOIN CUSTOMERS USING (CUST_ID) WHERE AMOUNT_SOLD>1780));

4.3 FUNCIONES AGREGADAS

1. Calculate the number of customers stored in the database.V_AGGR_1

SELECT COUNT(CUST_ID) FROM CUSTOMERS;

55000

2. What is the total number of products?. Does it correspond with the highest ID_PRODUCTS or there exist some gap in the sequence?V_AGGR_2

SELECT COUNT(PROD_ID) FROM PRODUCTS;

81, the ones that were not sold.

3. What is the number of different values for the marital status attribute?V_AGGR_3

SELECT COUNT(DISTINCT CUST_MARITAL_STATUS) FROM CUSTOMERS;

11.

4. Calculate the average of money that the customers spend.V_AGGR_4

SELECT AVG(AMOUNT_SOLD) FROM SALES;

107.67

5. What is the average of the list price and the minimum price all the products?V_AGGR_5

SELECT AVG(PROD_MIN_PRICE), AVG(PROD_LIST_PRICE) FROM PRODUCTS;

6. What is the average, the maximum and the minimum of the list price of all the products?V_AGGR_6

SELECT MAX(PROD_LIST_PRICE), MIN(PROD_LIST_PRICE), AVG(PROD_LIST_PRICE) FROM PRODUCTS;

7. Compute the global amount of money earn by the company and the total number of single sales made during its lifetime.V_AGGR_7

SELECT SUM(AMOUNT_SOLD), COUNT(*) FROM SALES;

4.4 AGRUPACIONES (GROUP BY)

1. For each product, list its ID and the total amount of money earn with it. V_GR_1

SELECT PROD_ID, SUM(AMOUNT_SOLD) FROM SALES GROUP BY PROD_ID;

2. For each country, list the number of its citizens and the ID of the country. V_GR_2

SELECT COUNTRY_ID, COUNT(*) FROM CUSTOMERS GROUP BY COUNTRY_ID;

3. For each promotion, list its ID, the total number of sales corresponding with it, the number of different channels and the number of different customers that use the advantages of such promotion. V_GR_3

SELECT PROMO_ID, COUNT(*) "TOTAL NUMBER OF SALES", COUNT (DISTINCT CHANNEL_ID) "NUMBER OF CHANNELS", COUNT (DISTINCT CUST_ID) "NUMBER OF DIFFERENT CUSTOMERS" FROM SALES GROUP BY PROMO_ID;

4. For each promotion and channel, compute the total amount of money received with its sales. V_GR_4

SELECT PROMO_ID, CHANNEL_ID, SUM(AMOUNT_SOLD) FROM SALES GROUP BY PROMO_ID, CHANNEL_ID;

5. For each country and gender, calculate the number of customers in such category. Provide also the range of ages (minimum and maximum) for each category. V_GR_5

SELECT COUNTRY_ID, CUST_GENDER, COUNT(*), MIN(CUST_YEAR_OF_BIRTH) "Oldest", MAX(CUST_YEAR_OF_BIRTH) "Youngest" FROM CUSTOMERS GROUP BY COUNTRY_ID, CUST_GENDER ;

6. For each continent, list the number of countries and the number of subcontinents. V_GR_6

SELECT COUNTRY_REGION_ID, COUNT(*), COUNT(DISTINCT COUNTRY_SUBREGION_ID) FROM COUNTRIES GROUP BY COUNTRY_REGION_ID;

4.5 MÁS AGRUPACIONES

1. For each product, list the money earn with it. List the name of the product and the global amount of money.SH.V_MOREGR_1

SELECT PROD_NAME, SUM(NVL(AMOUNT_SOLD, 0)) MONEY_EARNED

FROM PRODUCTS OUTER LEFT JOIN SALES USING(PROD_ID)

GROUP BY PROD_NAME, PROD_ID;

2. For each country, list the number of its citizens. Show the name of the country and this number of people.SH.V_MOREGR_2

SELECT COUNTRY_NAME, COUNT(*)

FROM CUSTOMERS JOIN COUNTRIES USING (COUNTRY_ID)

GROUP BY COUNTRY_NAME;

3. For each month, make a report with the total number of sales (show the month name stored in the TIMES table).SH.V_MOREGR_3

SELECT TO_CHAR(TIME_ID, 'Month'), COUNT(*) FROM SALES GROUP BY (TO_CHAR(TIME_ID, 'Month'));

4. In the above query, add the global amount of money sold in the month.SH.V_MOREGR_4

SELECT TO_CHAR(TIME_ID, 'Month'), COUNT(*), SUM(AMOUNT_SOLD) FROM SALES GROUP BY (TO_CHAR(TIME_ID, 'Month'));

5. Change the above query to structure the information according to the day of the weeks instead of the month.SH.V_MOREGR_5

SELECT TO_CHAR(TIME_ID, 'Day'), COUNT(*), SUM(AMOUNT_SOLD) FROM SALES GROUP BY (TO_CHAR(TIME_ID, 'Day'));

6.For each promotion and product, list the number of sales. Show the name of the product and the name of the promotion.SH.V_MOREGR_6

SELECT PROMO_NAME, PROD_NAME, COUNT(*) FROM SALES JOIN PROMOTIONS USING (PROMO_ID) JOIN PRODUCTS USING (PROD_ID) GROUP BY PROMO_NAME, PROD_NAME;

4.6 INCLUSO MÁS AGRUPACIONES

1. We would like to compute the number of women older than 92 which bought each product. The age must be computed with the sysdate as a reference (do not add manually 2020) to preserve its validity in the future. We also want to show the name of the product in this report. [V_EVENMABGR_1]

SELECT PROD_NAME, COUNT(DISTINCT CUST_ID)

FROM SALES JOIN PRODUCTS USING (PROD_ID) JOIN CUSTOMERS USING (CUST_ID)

WHERE CUST_GENDER='F' AND TO_NUMBER(TO_CHAR(SYSDATE, 'YYYY'))-
CUST_YEAR_OF_BIRTH>92 GROUP BY PROD_NAME;

2. In the database we would like to check the behaviour of the products with low price. Particularly we will show for each product with a list price less than 8$ the total amount of money we earn with its sales, i.e. the net benefit. This net benefit is computed by subtracting to the amount sold value the unit cost price. Thus, for each product we compute the total of the money we get as a benefit in all the sales of such product. Show this global net benefit, the total amount of money received for this product, the number of sales for the product, the last time when such product has been sold and, finally, the product name. [V_EVENMABGR_2].

SELECT PR.PROD_NAME "NOMBREDELPRODUCTO", SUM(SA.AMOUNT_SOLD) "TOTAL MONEY", COUNT(*) "NUMBER OF SALES", MAX(SA.TIME_ID) "LAST TIME", SUM(SA.AMOUNT_SOLD-CO.UNIT_COST) "NET BENEFIT"

FROM SALES SA JOIN COSTS CO ON (SA.PROMO_ID=CO.PROMO_ID AND SA.PROD_ID=CO.PROD_ID AND SA.CHANNEL_ID=CO.CHANNEL_ID AND SA.TIME_ID=CO.TIME_ID) JOIN PRODUCTS PR ON (SA.PROD_ID=PR.PROD_ID)

WHERE PROD_LIST_PRICE <8

GROUP BY PROD_NAME;

From this report, do you deduce that all the cheap products worth to be in the catalogue?

	BENEFIT	QUANTITY	NUMBER_SALES	LAST_TIME	PROD_NAME
1	292605,65	577420,62	19403	30/12/01	DVD-R Disc with Jewel Case, 4.7 GB
2	111361,45	711741,8	14381	31/12/01	OraMusic CD-R, Pack of 10
3	32457,19	170405,76	22189	31/12/01	CD-R with Jewel Cases, pACK OF 12
4	9918,25	58372,4	7283	28/12/01	Extension Cable
5	6186,64	34547,82	4091	31/12/01	Fly Fishing

Exxtension Cable and Fly Fishing are not worth it.

3. A receipt is the information regarding those products that a customer has brought at one particular day. It includes the full name of the customer, its ID and address together with the total amount sold and the date. Besides that, we include the tax information. In Spain the IVA is the 21% of the amount we have paid. [V_EVENMABGR_3].

SELECT CUST_FIRST_NAME, CUST_LAST_NAME, CUST_ID, CUST_STREET_ADDRESS, CUST_POSTAL_CODE, CUST_CITY, TIME_ID, SUM(AMOUNT_SOLD) "PAID", (SUM(AMOUNT_SOLD)*0.21)"TAXES"

FROM CUSTOMERS JOIN SALES USING (CUST_ID)

GROUP BY CUST_FIRST_NAME, CUST_LAST_NAME, CUST_ID, CUST_STREET_ADDRESS, CUST_POSTAL_CODE, CUST_CITY, TIME_ID;

4. We would like to analyse the behaviour of the sales by regions. Thus, for each subcontinent, we will show its name, the number of countries, the number of customers, the average of the sales in such region, the total amount sold and the range of sales (the interval where all amounts are included in this region). We do not consider the customers who were born before 1950. [V_EVENMABGR_4]

SELECT COUNTRY_SUBREGION, COUNT(DISTINCT COUNTRY_NAME) "COUNTRIES", COUNT(DISTINCT CUST_ID) "CUSTOMERS", AVG(AMOUNT_SOLD), SUM (AMOUNT_SOLD), MIN(AMOUNT_SOLD), MAX(AMOUNT_SOLD)

FROM SALES JOIN CUSTOMERS USING (CUST_ID) JOIN COUNTRIES USING (COUNTRY_ID)

WHERE TO_NUMBER(TO_CHAR(TIME_ID, 'YYYY'))>=1950

GROUP BY (COUNTRY_SUBREGION);

5. Show the first and the last day where we received money by using channel number 3. Also indicate the number of sales and the average of all these sales. [V_EVENMABGR_5]

SELECT COUNT(*), AVG(AMOUNT_SOLD), MIN(TIME_ID), MAX(TIME_ID) FROM SALES WHERE CHANNEL_ID=3;

6. Can you include in this query the information of the day of the week corresponding to these first and last day? Perhaps nesting SELECTs can help you. [V_EVENMABGR_6]

SELECT COUNT(*), AVG(AMOUNT_SOLD), TO_CHAR(MIN(TIME_ID), 'DAY') "DÍA PRIMERO", MIN(TIME_ID), MAX(TIME_ID), TO_CHAR(MAX(TIME_ID), 'DAY') "DÍA ÚLTIMO" FROM SALES WHERE CHANNEL_ID=3;

O más eficiente:

SELECT CANTIDAD, MEDIA, PRIMERO, TO_CHAR(PRIMERO, 'DAY') "DÍA PRIMERO", "ÚLTIMO", TO_CHAR(ÚLTIMO, 'DAY')"DÍA ÚLTIMO"

FROM DUAL,

(SELECT COUNT(*) "CANTIDAD", AVG(AMOUNT_SOLD) "MEDIA", MIN(TIME_ID) "PRIMERO", MAX(TIME_ID) "ÚLTIMO" FROM SALES WHERE CHANNEL_ID=3);

5 EJERCICIOS HAVING Y NESTING

5.1 HAVING

1. For each promotion (show the promotion ID), list the number of sales made. In this list, only the significant promotions, i.e. those which have at least 10.000 sales. V_HAVING_1

SELECT PROMO_ID, COUNT(*) NUM_SALES

FROM PROMOTIONS JOIN SALES USING (PROMO_ID)

GROUP BY PROMO_ID

HAVING COUNT (*) >= 10000;

2. For each product (show the product ID), list the average of its amount in the sales and the total units sold. Extract only the products having less than 1200 sales. V_HAVING_2

SELECT PROD_ID, AVG(AMOUNT_SOLD), COUNT(*) TOTAL_UNITS

FROM PRODUCTS JOIN SALES USING (PROD_ID)

GROUP BY PROD_ID

HAVING COUNT (*) < 1200;

3. For each country, show its ID, its name, the number of its customer being a female and the average of its credit limit. Show only the significant countries, i.e. the ones having at least 300 women. V_HAVING_3

SELECT COUNTRY_ID, COUNTRY_NAME, AVG(CUST_CREDIT_LIMIT),COUNT(*) NUM_CUSTOMERS

FROM CUSTOMERS JOIN COUNTRIES USING (COUNTRY_ID)

WHERE CUST_GENDER='F'

```
GROUP BY COUNTRY_ID, COUNTRY_NAME

HAVING COUNT(*)>=300;
```

4. For each product, show its ID and name, the average of amounts in the sales but only for those products sold in four different channels. V_HAVING_4

```
SELECT PROD_ID, PROD_NAME, AVG(AMOUNT_SOLD)

FROM PRODUCTS JOIN SALES USING(PROD_ID)

GROUP BY PROD_ID, PROD_NAME

HAVING COUNT(DISTINCT CHANNEL_ID)=4;
```

5. For each product, list its name together with the total amount of money earned with it (amount sold minus its cost). Pick out only those products that have been brought by 5000 different customers (at least) and whose global amount in the sales is greater than 300.000 V_HAVING_5

```
SELECT PROD_NAME, PROD_ID, SUM(AMOUNT_SOLD - UNIT_COST) AMOUNT_EARNED

FROM PRODUCTS JOIN SALES USING (PROD_ID)  JOIN COSTS USING (PROMO_ID, PROD_ID, CHANNEL_ID, TIME_ID)

GROUP BY PROD_NAME, PROD_ID

HAVING COUNT(DISTINCT CUST_ID)>=5000 AND SUM(AMOUNT_SOLD)>300000;
```

6. For each country and gender, show the country ID and the number of customers in such groups. List the information only for those groups where all its customers were born on 1925 or later. V_HAVING_6

```
SELECT COUNTRY_ID, CUST_GENDER, COUNT(*)

FROM CUSTOMERS

GROUP BY COUNTRY_ID, CUST_GENDER

HAVING MIN(CUST_YEAR_OF_BIRTH)>=1925;
```

7. List the full name of the persons who spend more than 16.000€ in a daily sale. A daily sale is consisting of all purchases belonging to the same customer in the same day. V_HAVING_7

```
SELECT TIME_ID, CUST_ID, CUST_FIRST_NAME, CUST_LAST_NAME, SUM(AMOUNT_SOLD)

FROM SALES NATURAL JOIN CUSTOMERS

GROUP BY TIME_ID, CUST_ID, CUST_FIRST_NAME, CUST_LAST_NAME

HAVING SUM(AMOUNT_SOLD)>16000;
```

5.2 NESTING

1. List the customers who had spent the most amount of money in a single sale. Do not show repeated rows. SH.V_NEST_1

```
SELECT DISTINCT CUST_ID, CUST_FIRST_NAME, CUST_LAST_NAME
```

```
FROM SALES NATURAL JOIN CUSTOMERS

WHERE AMOUNT_SOLD=(SELECT MAX(AMOUNT_SOLD) FROM SALES);
```

2. List the same information of customers who had spent in a single sale more amount than the average of single sales in all the history. Do not show repeated rows. SH.V_NEST_2

```
SELECT DISTINCT CUST_ID, CUST_FIRST_NAME, CUST_LAST_NAME

FROM SALES NATURAL JOIN CUSTOMERS

WHERE AMOUNT_SOLD>(SELECT AVG(AMOUNT_SOLD) FROM SALES);
```

3. List the same info for those customers who had spent in a single sale the maximum amount of money of all sales made in the same day on which he brought such item. Thus, the customer is the best customer in some specific day, since he spent the maximum on one day. Do not show repeated rows. SH.V_NEST_3

```
SELECT DISTINCT CUST_ID, CUST_FIRST_NAME, CUST_LAST_NAME

FROM SALES NATURAL JOIN CUSTOMERS JOIN (SELECT TIME_ID, MAX(AMOUNT_SOLD) MAX_AMOUNT FROM SALES GROUP BY TIME_ID) USING (TIME_ID)

WHERE AMOUNT_SOLD=MAX_AMOUNT;
```

4. List a similar query to the first one but, instead of "in a single sale", we look for the amount in all the sales that the customer did. SH.V_NEST_4

```
SELECT DISTINCT CUST_ID, CUST_FIRST_NAME, CUST_LAST_NAME

FROM SALES NATURAL JOIN CUSTOMERS

GROUP BY CUST_ID, CUST_FIRST_NAME, CUST_LAST_NAME

HAVING SUM(AMOUNT_SOLD)=

(SELECT MAX(SUM(AMOUNT_SOLD)) FROM SALES GROUP BY CUST_ID);
```

5. List the PROD_ID of the products that have a better chance of making a sale to the men. Hint: these products seem to be those ones that have been sold in a smaller number to the men (do not consider the equality). SH.V_NEST_5

```
SELECT PROD_ID

FROM

(

SELECT PROD_ID, COUNT(*) FEM_SALES

FROM PRODUCTS NATURAL JOIN SALES NATURAL JOIN CUSTOMERS

WHERE CUST_GENDER='F'

GROUP BY PROD_ID, CUST_GENDER

) JOIN (

SELECT PROD_ID, COUNT(*) MASC_SALES
```

```
FROM PRODUCTS NATURAL JOIN SALES NATURAL JOIN CUSTOMERS
WHERE CUST_GENDER='M'
GROUP BY PROD_ID, CUST_GENDER
) USING(PROD_ID)
WHERE MASC_SALES<FEM_SALES;
```

6. List the ID and the list price of the top-ten products which have a highest list price.

To get the position of a row in each table you can use the function ROWNUM. If you put this function is a SELECT the rows are labelled with its position. But observe that if you ORDER the result by the ROWNUM values, it seems to be unordered. The top-ten cannot be listed in a flat query, but you can first order the result by using the PRICE and use this table as a nested query in a from and then, in the outer query, you can only show the first ten rows by filtering the ROWNUM <= 10 in the second (outer) query.

SH.V_NEST_6

```
SELECT PROD_ID, PROD_LIST_PRICE
FROM (SELECT PROD_ID, PROD_LIST_PRICE FROM PRODUCTS ORDER BY PROD_LIST_PRICE DESC)
WHERE ROWNUM<=10;
```

7. Read again the exercise 5 of the "Even More About Groups" list of exercise. Can you include in this query the information of the day of the week corresponding to these first and last day? [SH.V_NEST_7]

```
SELECT NUMBER_SALES, AVERAGE_SALES, TO_CHAR(FIRST_DAY, 'DAY') FIRST_DAY_NAME, FIRST_DAY, TO_CHAR(LAST_DAY, 'DAY') LAST_DAY_NAME, LAST_DAY
```