



**Autonomous University of Zacatecas**

ACADEMIC UNIT OF ELECTRICAL ENGINEERING

Software Engineering Academic Program

*Group: 5B - Semester: 2022-5<sup>o</sup>*

**Practice Number: 09**

**Data selection and projection: single row  
functions**

DATE: 11/OCTOBER/2022

**Professor:**

Aldonso Becerra Sánchez.

**Student:**

Cristian Omar Alvarado Rodríguez.

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Practice objective</b>	<b>4</b>
<b>3</b>	<b>Developing</b>	<b>5</b>
<b>4</b>	<b>Pre-assessment</b>	<b>24</b>
<b>5</b>	<b>Conclusion</b>	<b>24</b>

# Data selection and projection: single row functions

October 11, 2022

## 1 Introduction

The DML (Data Modification Language) is one of the fundamental parts of the SQL language. It is formed by the instructions capable of modifying (add, change or delete) the data of the tables.

The set of DML statements that are executed consecutively is called a transaction. The interesting thing about transactions is that we can cancel them, since they form a logical unit of work that until they are accepted, their results will not be final.

In all DML statements, the only data returned by the system is the number of rows that have been modified by executing the statement.

The elements used to manipulate the data are the following:

- SELECT**, this statement is used to query the data.
- INSERT**, with this instruction we can insert the values in a database.
- UPDATE**, used to modify the values of one or more records.
- DELETE** is used to remove rows from a table.

**Data queries with SQL (DQL):**

DQL is short for SQL Data Query Language. The only command that belongs to this language is the versatile **SELECT** command. This command fundamentally allows:

- 1- Get data from certain columns of a table (projection).
- 2- Get records (rows) from a table according to certain criteria (selection).
- 3- Mix data from different tables (association, join).
- 4- Perform calculations on the data group data.

So far we have seen how the **SELECT** statement can be used to retrieve all or a subset of the columns from a table. But this effect affects all rows in the table, unless we specify something else in the **WHERE** clause. This is where we must propose the condition that all the rows must meet to appear in the result of the query. The complexity of the search criteria is practically unlimited, and it is possible to combine various types of operators with column functions, composing more or less complex expressions.

**ORDER BY clause:** Used to specify the sort order of the response to the query. By default the order is ascending, although a descending order can be specified. Sorting can be set on the content of columns or on expressions with columns.

## 2 Practice objective

Use SQL **SELECT** statements for retrieving data from database by means of different contexts using different Oracle functions.

### 3 Developing

**Activity 1:** Read all the choices carefully because there might be more than one correct answer. Choose all the correct answers for each question.

**Explain the reason for your answer.**

**DESCRIBE VARIOUS TYPES OF FUNCTIONS AVAILABLE IN SQL.**

**1. Which statements regarding single-row functions are true? (Choose all that apply.)**

- A) They may return more than one result.
- B) They execute once for each record processed.
- C) They may have zero or more input parameters.
- D) They must have at least one mandatory parameter.

**Explanation:** B and C, Single-row functions execute once for every record selected in a dataset and may either take no input parameters, like SYSDATE, or many input parameters.

**2. Which of these are single-row character-case conversion functions? (Choose all that apply.)**

- A) LOWER.
- B) SMALLER.
- C) INITCASE.
- D) INITCAP.

**Explanation:** A and D, The LOWER function converts the case of the input string parameter to its lowercase equivalent, while INITCAP converts the given input parameter to title case.

## USE CHARACTER, NUMBER, AND DATE FUNCTIONS IN SELECT STATEMENTS

3. What value is returned after executing the following statement: `SELECT LENGTH('How_long_is_a_piece_of_string?') FROM DUAL;` (Choose the best answer.

- A) 29.
- B) 30.
- C) 24.
- D) None of the above.

**Explanation: B**, The LENGTH function computes the number of characters in a given input string including spaces, tabs, punctuation mark, and other non printable special characters.

4. What value is returned after executing the following statement: `SELECT SUBSTR('How_long_is_a_piece_of_string?', 5,4) FROM DUAL;` (Choose the best answer.)

- A) long.
- B) \_long.
- C) string?.
- D) None of the above.

**Explanation: A**, The SUBSTR function extracts a four-character substring from the given input string starting with and including the fifth character. The characters at positions 1 to 4 are How\_. Starting with the character at position 5, the next four characters form the word “long.”

5. What value is returned after executing the following statement? `SELECT INSTR('How_long_is_a_piece_of_string?', '_ ',5,3) FROM DUAL;` (Choose the best answer.)

- A) 4.
- B) 14.
- C) 12.

D) None of the above.

**Explanation:** B, The INSTR function returns the position that the nth occurrence of the search string may be found after starting the search from a given start position. The search string is the underscore character, and the third occurrence of this character starting from position 5 in the source string occurs at position 14.

6. What value is returned after executing the following statement? `SELECT REPLACE('How long is a piece of string?', '_', '') FROM DUAL;` (Choose the best answer.)

A) How long is a piece of string?.,.

B) How long is a piece of string?.

C) Howlongisapieceofstring?.

D) None of the above.

**Explanation:** C, All occurrences of the underscore character are replaced by an empty string, which removes them from the string.

7. What value is returned after executing the following statement? `SELECT MOD(14,3) FROM DUAL;` (Choose the best answer.)

A) 3.

B) 42.

C) 2.

D) None of the above.

**Explanation:** C, When 14 is divided by 3, the answer is 4 with remainder 2.

8. Assuming `SYSDATE=07-JUN-1996 12:05pm`, what value is returned after executing the following statement? `SELECT ADD_MONTHS(SYSDATE,-1) FROM DUAL;` (Choose the best answer.

- A) 07-MAY-1996 12:05pm.
- B) 06-JUN-1996 12:05pm.
- C) 07-JUL-1996 12:05pm.
- D) None of the above.

**Explanation:** A, The minus one parameter indicates to the `ADD_MONTHS` function that the date to be returned must be one month prior to the given date.

9. What value is returned after executing the following statement? Take note that 01-JAN-2009 occurs on a Thursday. (Choose the best answer.  
`SELECT NEXT_DAY('01-JAN-2009','wed') FROM DUAL;`

- A) 07-JAN-2009.
- B) 31-JAN-2009.
- C) Wednesday.
- D) None of the above.

**Explanation:** A, Since the first of January 2009 falls on a Thursday, the date of the following Wednesday is six days later.

10. Assuming `SYSDATE=30-DEC-2007`, what value is returned after executing the following statement? `SELECT TRUNC(SYSDATE,'YEAR') FROM DUAL;` (Choose the best answer.)

- A) 31-DEC-2007.
- B) 01-JAN-2008.
- C) 01-JAN-2007.



D) None of the above.

**Explanation:** C, The date TRUNC function does not perform rounding and since the degree of truncation is YEAR, the day and month components of the given date are ignored and the first day of the year it belongs to is returned.

**Activity 2:** Propose an answer to the following issues:

a) You would like to search for a character string stored in the database. The case in which it is stored is unknown and there are potentially leading and trailing spaces surrounding the string. Can such a search be performed?

**R =** If the search can be carried out since with the help of the TRIM function the blank spaces can be removed both in front and after the character string. Other functions that could help the search to be done correctly are the LOWER and UPPER functions since the format of the string to search for is unknown and with the help of these functions we can convert the string to upper or lower case.

b) You have been asked to extract the last three characters from the LAST\_NAME column in the EMPLOYEES table. Can such a query be performed without using the LENGTH function?

**R =** If a query can be made without using the LENGTH function since the SUBSTR function allows obtaining a certain number of characters from a string. Said query would be as follows:

```
SELECT SUBSTR(LAST_NAME, -3, 3) FROM EMPLOYEES ;
```

c) You would like to extract a consistent 11-character string based on the SALARY column in the EMPLOYEES table. If the SALARY value is less than 11 characters long, zeros must be added to the left of the value to yield a 11-character string. Is this possible?

**R** = If this is possible with the help of the LPAD function which pads a string from the left up to length n with a defined character. The declaration for this case would be as follows:

```
SELECT LPAD(SALARY, 11, 0) FROM EMPLOYEES;
```

d) You wish to retrieve the duration of employment in days for each employee. Is it possible to perform such a calculation?

**R** = If it is possible to carry out this calculation since it could be done by subtracting the current date (SYSDATE) minus the date of hiring of the employees.

e) You are tasked with identifying the date the end of year staff bonus will be paid. Bonuses are usually paid on the last Friday in December. Can the bonus date be computed using the NEXT\_DAY function?

**R** = If the bonus date can be calculated since the NEXT\_DAY function returns the closest date after the date (D) whose day of the week is WD. WD can be MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, SUNDAY.

f) Employees working in the IT department have moved to new offices and, although the last four digits of their phone numbers are the same, the set of the three digits 324 is changed to 326. A typical phone number of an IT staff member is 140-324-3489. You are required to provide a list of employees' names with their old and new phone numbers. Can this list be provided?

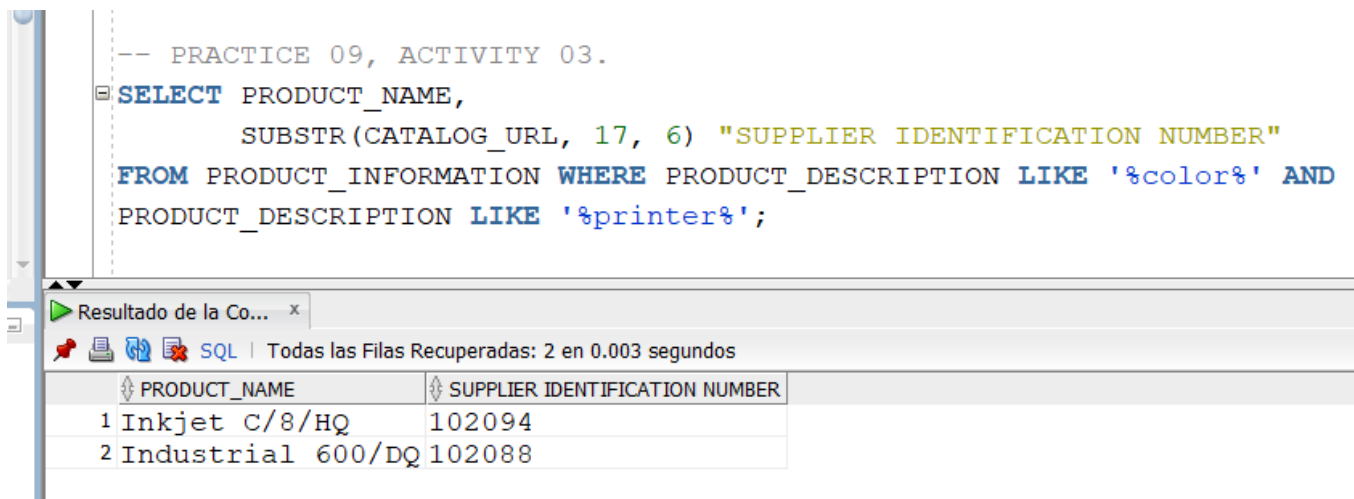
**R** = If the list of telephones can be provided since with the REPLACE function it can be done since this function replaces all the matches of a set of characters within an existing string with other specified characters.

**Activity 3:** Connect to the OE schema and complete the following tasks.

Several quotations were requested for prices on color printers. The supplier information is not available from the usual source, but you know that the supplier identification number is embedded in the CATALOG\_URL column from the PRODUCT\_INFORMATION table. You are required to retrieve the PRODUCT\_NAME and CATALOG\_URL values and to extract the supplier number from the CATALOG\_URL column for all products which have both the words COLOR and PRINTER in the PRODUCT\_DESCRIPTION column stored in any case.

**Note:** Capture an image for each statement output.

In the statement in **figure 1** we can see how to obtain the supplier number, the SUBSTR function was used, to which the CATALOG\_URL string and a 17 are passed as arguments, which indicates the starting position to extract a sub string from the CATALOG\_URL string. and as the last attribute a 6 is passed which indicates the number of characters to be extracted from the string-, that is to say that a substring of the CATALOG\_URL with the 6 characters after position 17 is going to be extracted.



```
-- PRACTICE 09, ACTIVITY 03.
SELECT PRODUCT_NAME,
       SUBSTR(CATALOG_URL, 17, 6) "SUPPLIER IDENTIFICATION NUMBER"
FROM PRODUCT_INFORMATION WHERE PRODUCT_DESCRIPTION LIKE '%color%' AND
PRODUCT_DESCRIPTION LIKE '%printer%';
```

Resultado de la Co... x

Todas las Filas Recuperadas: 2 en 0.003 segundos

	PRODUCT_NAME	SUPPLIER IDENTIFICATION NUMBER
1	Inkjet C/8/HQ	102094
2	Industrial 600/DQ	102088

Figure 1: *SELECT* statement, data projection.

**Activity 4:** This exercise must be performed using HR schema.

- Retrieve a list of all FIRST\_NAME and LAST\_NAME values from the EMPLOYEES table where FIRST\_NAME contains the character string “li.” 1. Start SQL Developer and connect to the HR schema. The data filter must compare the FIRST\_NAME column values with a pattern of characters containing all possible case combinations of the string “li.” Therefore, if the FIRST\_NAME contains the character strings “LI,” “Li,” “lI,” or “li,” that row must be retrieved. The LIKE operator is used for character matching, and four combinations can be extracted with four WHERE clauses separated by the OR keyword. However, the case conversion functions can simplify the condition, see figure 2.



```
-- PRACTICE 09, ACTIVITY 04.
SELECT FIRST_NAME, LAST_NAME
FROM EMPLOYEES WHERE LOWER(FIRST_NAME) LIKE '%li%';
```

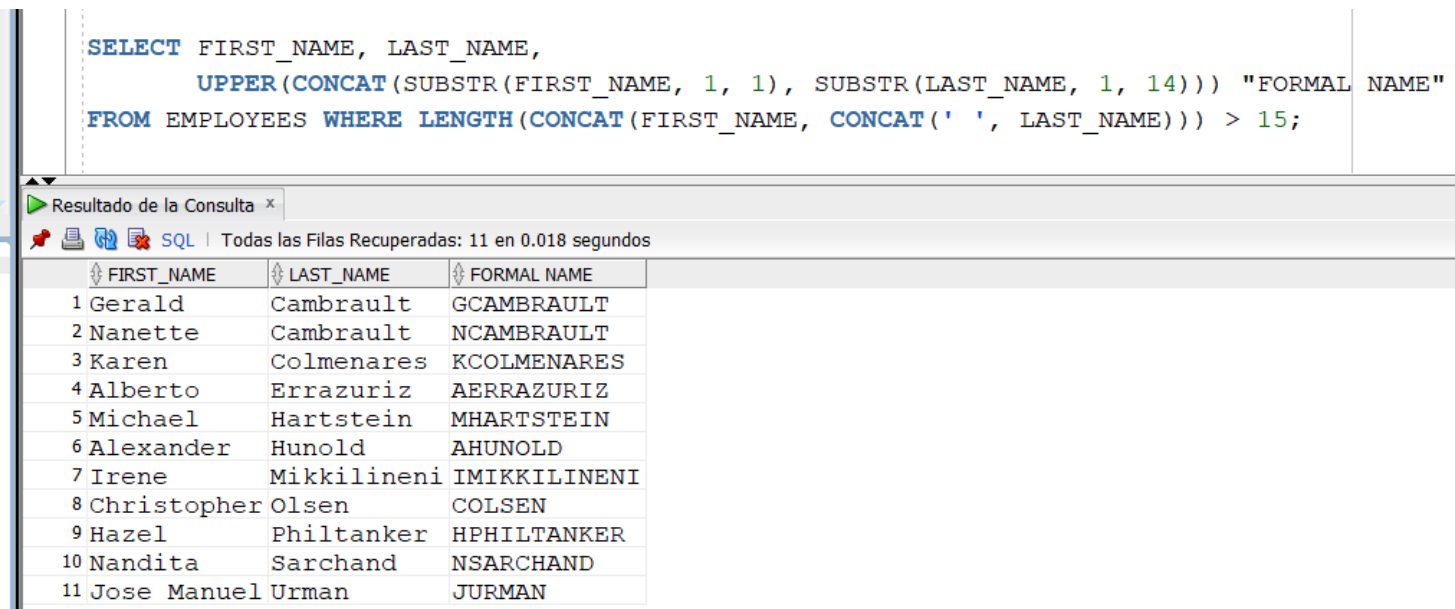
Resultado de la Co... x

SQL | Todas las Filas Recuperadas: 10 en 0 segundos

	FIRST_NAME	LAST_NAME
1	Shelli	Baida
2	Elizabeth	Bates
3	Julia	Dellinger
4	William	Gietz
5	Julia	Nayer
6	Lisa	Ozer
7	Valli	Pataballa
8	Lindsey	Smith
9	William	Smith
10	Oliver	Tuvault

Figure 2: *SELECT* statement, data projection.

- Envelope printing restricts the addressee field to 16 characters. Ideally, the addressee field contains employees' FIRST\_NAME and LAST\_NAME values separated by a single space. When the combined length of an employee's FIRST\_NAME and LAST\_NAME exceeds 15 characters, the addressee field should contain their formal name. An employee's formal name is made up of the first letter of their FIRST\_NAME and the first 14 characters of their LAST\_NAME. You are required to retrieve a list of FIRST\_NAME and LAST\_NAME values and formal names for employees where the combined length of FIRST\_NAME and LAST\_NAME exceeds 15 characters, **see figure 3**.



```

SELECT FIRST_NAME, LAST_NAME,
       UPPER(CONCAT(SUBSTR(FIRST_NAME, 1, 1), SUBSTR(LAST_NAME, 1, 14))) "FORMAL NAME"
FROM EMPLOYEES WHERE LENGTH(CONCAT(FIRST_NAME, CONCAT(' ', LAST_NAME))) > 15;

```

Resultado de la Consulta x

SQL | Todas las Filas Recuperadas: 11 en 0.018 segundos

	FIRST_NAME	LAST_NAME	FORMAL NAME
1	Gerald	Cambrault	GCAMBRAULT
2	Nanette	Cambrault	NCAMBRAULT
3	Karen	Colmenares	KCOLMENARES
4	Alberto	Errazuriz	AERRAZURIZ
5	Michael	Hartstein	MHARTSTEIN
6	Alexander	Hunold	AHUNOLD
7	Irene	Mikkilineni	IMIKKILINENI
8	Christopher	Olsen	COLSEN
9	Hazel	Philtanker	HPHILTANKER
10	Nandita	Sarchand	NSARCHAND
11	Jose Manuel	Urman	JURMAN

Figure 3: *SELECT statement, data projection.*

- You are required to obtain a list of EMPLOYEE\_ID, LAST\_NAME, and HIRE\_DATE values (add the MONTHS WORKED by the employees) for the employees who have worked more than 90 months between the date they were hired and 01-JAN-2000, **see figure 4**.

```
SELECT EMPLOYEE_ID, LAST_NAME, HIRE_DATE,
       MONTHS_BETWEEN(HIRE_DATE, '01-ENE-2000') "MONTHS WORKED"
FROM EMPLOYEES WHERE MONTHS_BETWEEN(HIRE_DATE, '01-ENE-2000') > 90;
```

Resultado de la Consulta x

SQL | Todas las Filas Recuperadas: 18 en 0.005 segundos

	EMPLOYEE_ID	LAST_NAME	HIRE_DATE	MONTHS WORKED
1	113	Popp	07/12/07	95.19354838709677419354838709677419354839
2	119	Colmenares	10/08/07	91.29032258064516129032258064516129032258
3	124	Mourgos	16/11/07	94.48387096774193548387096774193548387097
4	128	Markle	08/03/08	98.22580645161290322580645161290322580645
5	135	Gee	12/12/07	95.35483870967741935483870967741935483871
6	136	Philtanker	06/02/08	97.16129032258064516129032258064516129032
7	148	Cambrault	15/10/07	93.4516129032258064516129032258064516129
8	149	Zlotkey	29/01/08	96.90322580645161290322580645161290322581
9	155	Tuvault	23/11/07	94.70967741935483870967741935483870967742
10	164	Marvins	24/01/08	96.74193548387096774193548387096774193548
11	165	Lee	23/02/08	97.70967741935483870967741935483870967742

Figure 4: *SELECT statement, data projection.*

- You are required to display employee first names and last names joined together, the length of the employee last name, and the numeric position of the letter “a” in the employee last name for all employees whose last names end with the letter “n.” Use Oracle functions to perform the whole sentence, see figure 5.

```
SELECT FIRST_NAME, LAST_NAME, LENGTH(LAST_NAME) "LENGTH OF LAST_NAME",
       INSTR(LAST_NAME, 'a') "Position of 'a' in LAST_NAME"
FROM EMPLOYEES WHERE LAST_NAME LIKE '%n';
```

Resultado de la Consulta x

SQL | Todas las Filas Recuperadas: 19 en 0.006 segundos

	FIRST_NAME	LAST_NAME	LENGTH OF LAST_NAME	Position of 'a' in LAST_NAME
1	Mozhe	Atkinson	8	0
2	David	Austin	6	0
3	David	Bernstein	9	0
4	John	Chen	4	0
5	Lex	De Haan	7	5
6	Louise	Doran	5	4
7	Michael	Hartstein	9	2
8	Alyssa	Hutton	6	0
9	Charles	Johnson	7	0
10	Jack	Livingston	10	0
11	Jason	Mallin	6	2
12	Samuel	McCain	6	4

Figure 5: *SELECT statement, data projection.*

- Display the employee number, hire date, number of months employed, six-month review date, first Friday after hire date, and the last day of the hire month for all employees who have been employed for fewer than 150 months, see figure 6.

```

SELECT EMPLOYEE_ID, HIRE_DATE,
       MONTHS_BETWEEN(SYSDATE, HIRE_DATE) "Number Of Months Employed",
       ADD_MONTHS(HIRE_DATE, 6) "Review Date Six-Month ",
       NEXT_DAY(HIRE_DATE, 'FRIDAY') "Next Day Friday hire Date",
       LAST_DAY(HIRE_DATE) "Last Day Hire Date"
FROM EMPLOYEES WHERE MONTHS_BETWEEN(SYSDATE, HIRE_DATE) > 150;

```

Resultado de la Consulta x

SQL | Se han recuperado 50 filas en 0.003 segundos

	EMPLOYEE_ID	HIRE_DATE	Number Of Months Employed	Review Date Six-Month	Next Day Friday hire Date	Last Day Hire Date
1	100	17/06/03	231.866563620071684587813620071684587814	17/12/03	20/06/03	30/06/03
2	101	21/09/05	204.73753136200716845878136200716845878121	03/06	23/09/05	30/09/05
3	102	13/01/01	260.99559587813620071684587813620071684613	07/01	19/01/01	31/01/01
4	103	03/01/06	201.318176523297491039426523297491039427	03/07/06	06/01/06	31/01/06
5	104	21/05/07	184.73753136200716845878136200716845878121	11/07	25/05/07	31/05/07
6	105	25/06/05	207.60849910394265232974910394265232974925	12/05	01/07/05	30/06/05
7	106	05/02/06	200.2536603942652329749103942652329749105	08/06	10/02/06	28/02/06
8	107	07/02/07	188.18914426523297491039426523297491039407	08/07	09/02/07	28/02/07
9	108	17/08/02	241.86656362007168458781362007168458781417	02/03	23/08/02	31/08/02
10	109	16/08/02	241.89882168458781362007168458781362007216	02/03	23/08/02	31/08/02
11	110	28/09/05	204.51172491039426523297491039426523297528	03/06	30/09/05	30/09/05
12	111	30/09/05	204.44720878136200716845878136200716845931	03/06	07/10/05	30/09/05
13	112	07/02/06	180.18914426523297491039426523297491039407	08/06	10/02/06	28/02/06

Figure 6: *SELECT* statement, data projection.

- Compare the hire dates for all employees who started in 1997. Display the employee number and hire date. Because the version of Oracle DataBase that I have installed is different and there are no records for the year 1997, I decided to change the query to the year 2002, see figure 7.

```

SELECT EMPLOYEE_ID, HIRE_DATE FROM EMPLOYEES WHERE HIRE_DATE LIKE '%02';

```

Resultado de la Consulta x

SQL | Todas las Filas Recuperadas: 7 en 0.03 segundos

	EMPLOYEE_ID	HIRE_DATE
1	108	17/08/02
2	109	16/08/02
3	114	07/12/02
4	203	07/06/02
5	204	07/06/02
6	205	07/06/02
7	206	07/06/02

Figure 7: *SELECT* statement, data projection.



**Activity 5:** This activity provides a variety of exercises using different functions that are available for character, number, and date data types.

**Part 1:**

1. Write a query to display the system date. Label the column as Date. **Note:** If your database is remotely located in a different time zone, the output will be the date for the operating system on which the database resides, see **figure 8**.

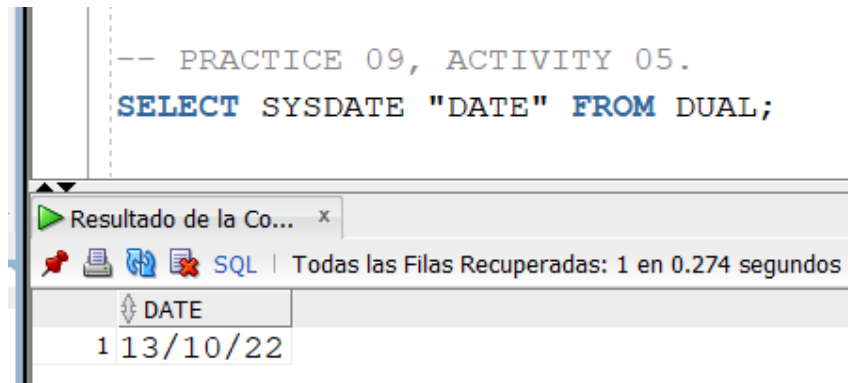
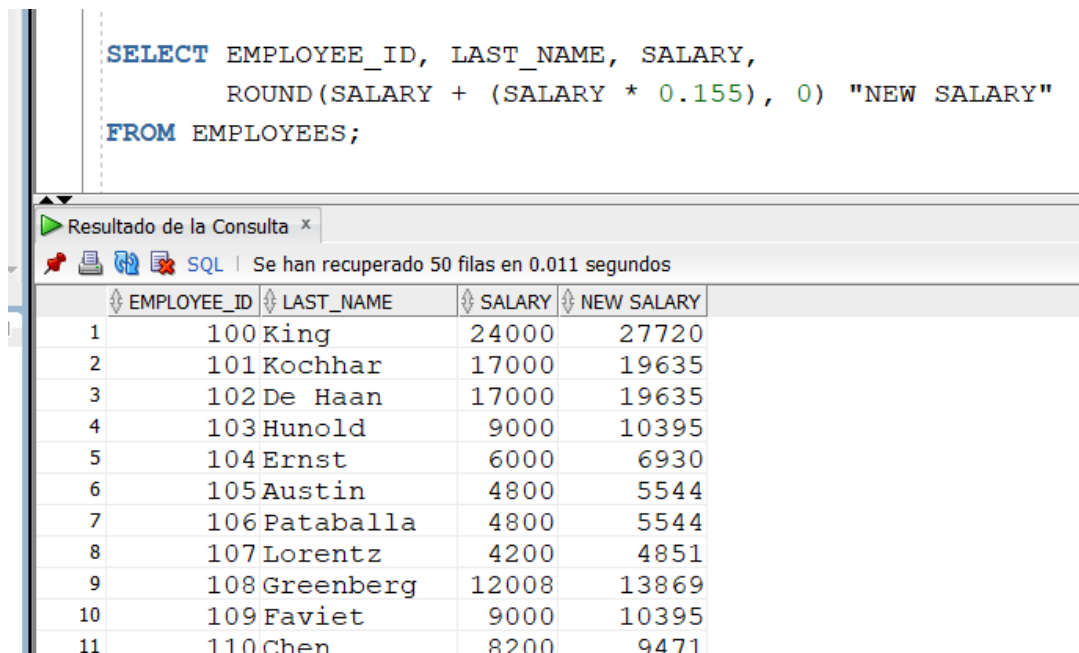


Figure 8: *SELECT statement, data projection.*

2. The HR department needs a report to display the employee number, last name, salary, and salary increased by 15.5% (expressed as a whole number) for each employee. Label the column New Salary. Save your SQL statement in a file named lab\_9\_02.sql.

3. Run your query in the lab\_9\_02.sql file, see **figure 9**.



```
SELECT EMPLOYEE_ID, LAST_NAME, SALARY,  
       ROUND(SALARY + (SALARY * 0.155), 0) "NEW SALARY"  
FROM EMPLOYEES;
```

Resultado de la Consulta x

SQL | Se han recuperado 50 filas en 0.011 segundos

	EMPLOYEE_ID	LAST_NAME	SALARY	NEW SALARY
1	100	King	24000	27720
2	101	Kochhar	17000	19635
3	102	De Haan	17000	19635
4	103	Hunold	9000	10395
5	104	Ernst	6000	6930
6	105	Austin	4800	5544
7	106	Pataballa	4800	5544
8	107	Lorentz	4200	4851
9	108	Greenberg	12008	13869
10	109	Faviet	9000	10395
11	110	Chen	8200	9471

Figure 9: *SELECT* statement, data projection.

4. Modify your query lab\_9\_02.sql to add a column that subtracts the old salary from the new salary. Label the column Increase. Save the contents of the file as lab\_9\_04.sql. Run the revised query, see figure 10.

```

SELECT EMPLOYEE_ID, LAST_NAME, SALARY,
       ROUND(SALARY + (SALARY * 0.155), 0) "NEW SALARY",
       ROUND((SALARY + (SALARY * 0.155)) - SALARY, 0) "INCREASE"
FROM EMPLOYEES;

```

Resultado de la Consulta x

SQL | Se han recuperado 50 filas en 0.008 segundos

	EMPLOYEE_ID	LAST_NAME	SALARY	NEW SALARY	INCREASE
1	100	King	24000	27720	3720
2	101	Kochhar	17000	19635	2635
3	102	De Haan	17000	19635	2635
4	103	Hunold	9000	10395	1395
5	104	Ernst	6000	6930	930
6	105	Austin	4800	5544	744
7	106	Pataballa	4800	5544	744
8	107	Torentz	4200	4851	651

Figure 10: *SELECT* statement, data projection.

5. Write a query that displays the last name (with the first letter in uppercase and all the other letters in lowercase) and the length of the last name for all employees whose name starts with the letters “J,” “A,” or “M.” Give each column an appropriate label. Sort the results by the employees’ last names, see figure 11.

```

SELECT INITCAP(LAST_NAME) "LAST_NAME", LENGTH(LAST_NAME) "LENGTH OF LAST_NAME"
FROM EMPLOYEES WHERE LAST_NAME LIKE 'A%' OR LAST_NAME LIKE 'J%' OR LAST_NAME LIKE 'M%'
ORDER BY LAST_NAME ASC;

```

Resultado de la Consulta x

SQL | Todas las Filas Recuperadas: 16 en 0.005 segundos

	LAST_NAME	LENGTH OF LAST_NAME
1	Abel	4
2	Ande	4
3	Atkinson	8
4	Austin	6
5	Johnson	7
6	Jones	5
7	Mallin	6
8	Markle	6
9	Marlow	6
10	Marvins	7
11	Matos	5

Figure 11: *SELECT* statement, data projection.

Rewrite the query so that the user is prompted to enter a letter that the last name starts with. For example, if the user enters “H” (capitalized) when prompted for a letter, then the output should show all employees whose last name starts with the letter “H.”. Modify the query such that the case of the entered letter does not affect the output. The entered letter must be capitalized before being processed by the SELECT query, see figure 12 and 13.

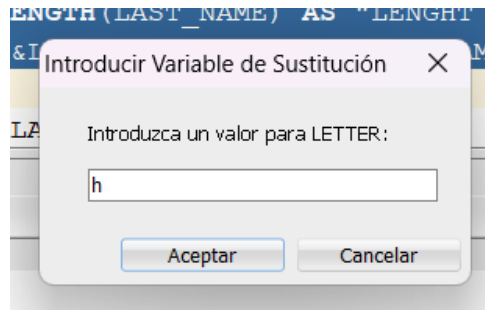


Figure 12: *Substitution variable.*

```
SELECT INITCAP(LAST_NAME) AS "LAST_NAME", LENGTH(LAST_NAME) AS "LENGHT OF LAST_NAME"
FROM EMPLOYEES WHERE LAST_NAME LIKE UPPER('&LETTER%') ORDER BY LAST_NAME ASC;
```

Resultado de la Consulta x

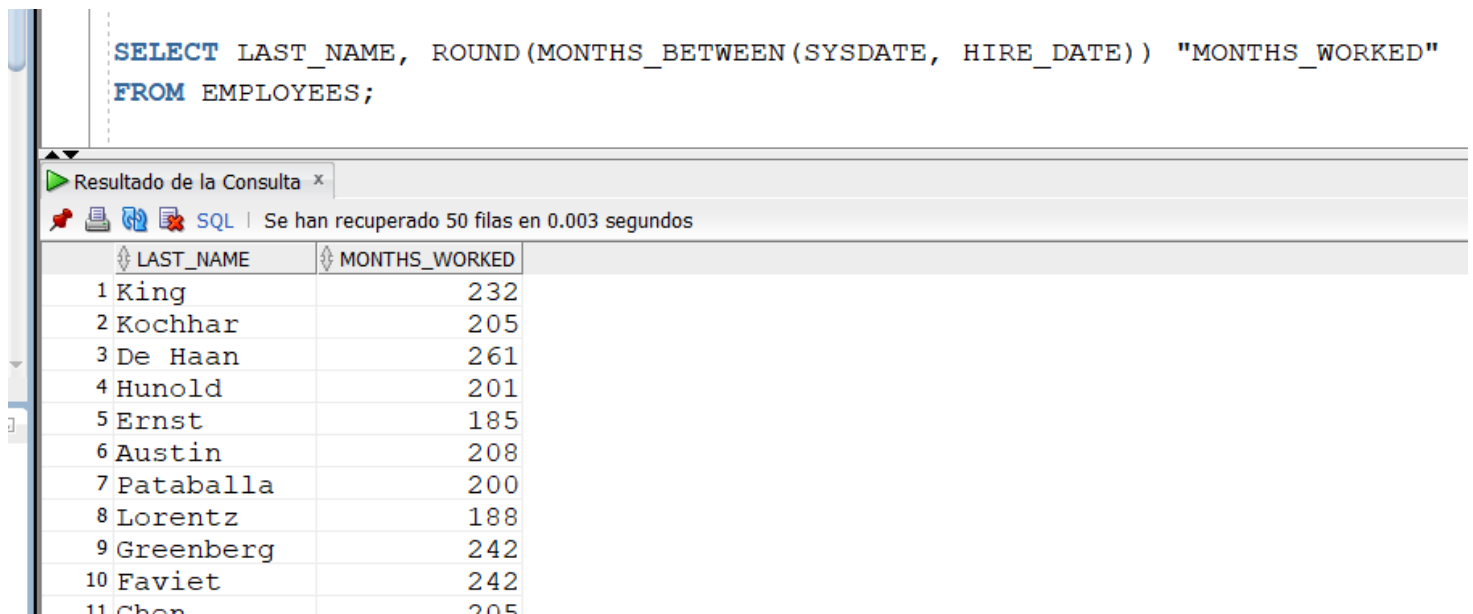
SQL | Todas las Filas Recuperadas: 6 en 0.006 segundos

	LAST_NAME	LENGHT OF LAST_NAME
1	Hall	4
2	Hartstein	9
3	Higgins	7
4	Himuro	6
5	Hunold	6
6	Hutton	6

Figure 13: *SELECT statement, data projection.*

6. The HR department wants to find the duration of employment for each employee. For each employee, display the last name and calculate the number of months between today and the date on which the employee was hired. Label the column as MONTHS\_WORKED. Order your results by the number of months employed. Round the number of months up to the closest whole number, see figure 14.

**Note:** Because this query depends on the date when it was executed, the values in the MONTHS\_WORKED column will differ for you.



```
SELECT LAST_NAME, ROUND(MONTHS_BETWEEN(SYSDATE, HIRE_DATE)) "MONTHS_WORKED"
FROM EMPLOYEES;
```

	LAST_NAME	MONTHS_WORKED
1	King	232
2	Kochhar	205
3	De Haan	261
4	Hunold	201
5	Ernst	185
6	Austin	208
7	Pataballa	200
8	Lorentz	188
9	Greenberg	242
10	Faviet	242
11	Chen	205

Figure 14: *SELECT statement, data projection.*

7. Create a query to display the last name and salary for all employees. Format the salary to be 15 characters long, left-padded with the \$ symbol. Label the column as SALARY, see figure 15.

```
SELECT LAST_NAME, LPAD(SALARY, 15, '$') "SALARY" FROM EMPLOYEES;
```

Resultado de la Consulta x

SQL | Se han recuperado 50 filas en 0.005 segundos

	LAST_NAME	SALARY
1	King	\$\$\$\$\$\$\$\$\$24000
2	Kochhar	\$\$\$\$\$\$\$\$\$17000
3	De Haan	\$\$\$\$\$\$\$\$\$17000
4	Hunold	\$\$\$\$\$\$\$\$\$9000
5	Ernst	\$\$\$\$\$\$\$\$\$6000
6	Austin	\$\$\$\$\$\$\$\$\$4800
7	Pataballa	\$\$\$\$\$\$\$\$\$4800
8	Lorentz	\$\$\$\$\$\$\$\$\$4200
9	Greenberg	\$\$\$\$\$\$\$\$\$12008
10	Faviet	\$\$\$\$\$\$\$\$\$9000

Figure 15: *SELECT* statement, data projection.

8. Create a query that displays the first eight characters of the employees' last names and indicates the amounts of their salaries with asterisks. Each asterisk signifies a thousand dollars. Sort the data in descending order of salary. Label the column as EMPLOYEES\_AND\_THEIR\_SALARIES, see figure 16.

```
SELECT RPAD(LAST_NAME, 8) || ' ' ||
       RPAD(' ', SALARY/1000+1, '*') "EMPLOYEES_AND_THEIR_SALARIES"
FROM EMPLOYEES ORDER BY SALARY DESC;
```

Resultado de la Consulta x

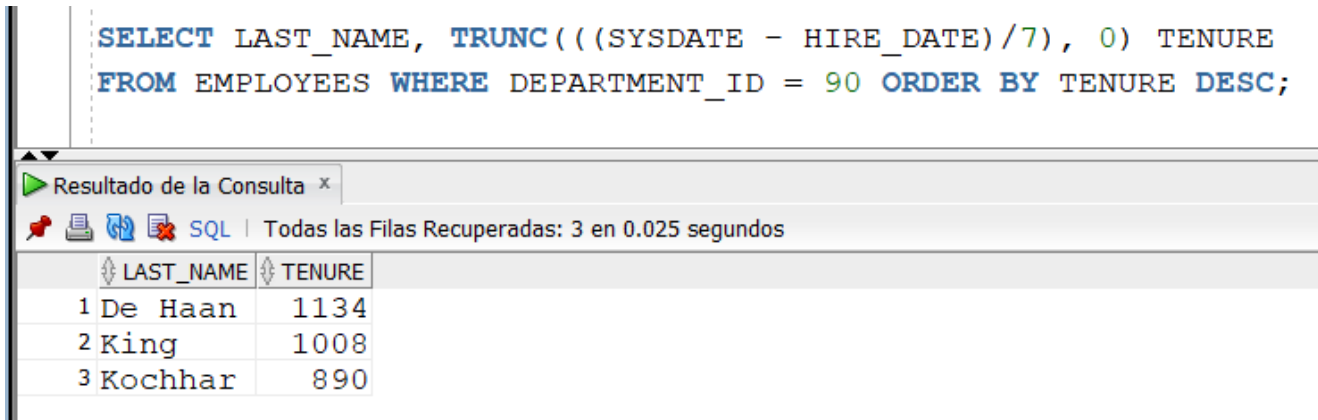
SQL | Se han recuperado 50 filas en 0.014 segundos

	EMPLOYEES_AND_THEIR_SALARIES
1	King *****
2	Kochhar *****
3	De Haan *****
4	Russell *****
5	Partners *****
6	Hartstei *****
7	Greenber *****
8	Higgins *****
9	Errazuri *****
10	Ozer *****

Figure 16: *SELECT* statement, data projection.

9. Create a query to display the last name and the number of weeks employed for all employees in department 90. Label the number of weeks column as TENURE. Truncate the number of weeks value to 0 decimal places. Show the records in descending order of the employee's tenure, **see figure 17**.

**Note:** The TENURE value will differ as it depends on the date on which you run the query.



```
SELECT LAST_NAME, TRUNC(((SYSDATE - HIRE_DATE)/7), 0) TENURE
FROM EMPLOYEES WHERE DEPARTMENT_ID = 90 ORDER BY TENURE DESC;
```

Resultado de la Consulta x

SQL | Todas las Filas Recuperadas: 3 en 0.025 segundos

	LAST_NAME	TENURE
1	De Haan	1134
2	King	1008
3	Kochhar	890

Figure 17: *SELECT* statement, data projection.

## 4 Pre-assessment

In this section you will find the Pre-assessment.

Criteria to be evaluate	Does it comply?	(%)
COMPLIES WITH THE REQUESTED FUNCTIONALITY	YES	
HAS THE CORRECT INDENTATION	YES	
HAS AN EASY WAY TO ACCESS THE PROVIDED FILES	YES	
HAS A REPORT WITH IDC FORMAT	YES	
REPORT INFORMATION IS FREE OF SPELLING ERRORS	YES	
DELIVERED IN TIME AND FORM	YES	
IS FULLY COMPLETED (SPECIFY THE PERCENTAGE COMPLETED)	YES	100%

## 5 Conclusion

The Oracle DML statements are transcendental in the handling of SQL statements at the level of both administrator and database programmer, since they allow the data manipulation of database schemes regardless of the platform used to generate it. This kind of statements can provide you data treatment mechanisms during daily programmer's days.

SQL language allows the realization of projection and selection of data to satisfy the needs of reports that may be required for a programmer, developer or end user.

This practice number 9 helped me to practice the uses of the SELECT statement for data retrieval. Finally, an important thing to mention is that the SQL language allows the projection and selection of data to satisfy the needs of reports that may be necessary for a programmer, developer or end user.