Autonomous University of Zacatecas

Academic Unit of Electrical Engineering

Software Engineering Academic Program

Group: 5B - Semester: 2022-5º

Practice Number: 12

Displaying Data from Multiple Tables

Date: 01/November/2022

Professor:

Aldonso Becerra Sánchez.

Student:

Cristian Omar Alvarado Rodríguez.

Contents

Displaying Data from Multiple Tables

November 1th, 2022

1 Introduction

Relational databases store their data in multiple tables. The normal thing, in almost any query, is to require data from several tables at the same time. This is possible because the data in the tables is linked by columns that contain secondary or foreign keys that allow the data in that table to be related to data in another table. Sometimes you have to use data from more than one table. A join is used to view information from multiple tables. In this way, you can join different tables and thus see information in them.

Types of Unions:

The joints compatible with the standard are:

- Cross joints.
- Natural unions.
- USING clause.
- Full external joins (or two-sided).
- Arbitrary join conditions for outer joins.

Natural Joins: You can automatically join tables based on columns in the two tables that have corresponding names and data types. Do this using the NATURAL JOIN keywords.

Unions with USING: The USING clause can be used to specify only the columns that should be used for an equal join. Columns referenced in the USING clause must not have a qualifier (table name or alias) anywhere in the SQL statement.

Unions with the ON Clause: Use the ON clause to specify a join condition. This allows you to specify join conditions apart from any search or filter conditions in the WHERE clause.

Non-Equal Unions: An inegal join is one that contains something other than an equality operator. Although conditions such as (<= and >=) can be used, it is advisable to use BETWEEN, since it is the simplest. Remember to specify the lowest value first and the highest value last when using BETWEEN.

External Unions (OUTER JOIN): LEFT is indicated if we want all the data of the table to the left of the JOIN word to appear (remember that for Oracle the instruction has only one line). In the same way, if we want the table on the right to show all the data, RIGHT is indicated. Finally, if we want both to show all the data, FULL is used.

Cross Unions (CROSS JOIN) and Cartesian Products: When a join condition is invalid or is skipped altogether, the result is a Cartesian product, showing all combinations of rows. All rows from the first table are joined to all rows from the second table.

A cartesian product tends to generate a large number of rows, so the result is usually not useful. You should always include a valid join condition unless you have a specific need to join all rows from all tables.

Cartesian products are useful if you need to generate a large number of rows to simulate an acceptable amount of data.

Although to generate one it is only necessary to omit a join condition, there is a CROSS JOIN clause that does the same thing.

2 Practice objective

Use SQL SELECT statements for retrieving data from several tables.

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.

WRITE SELECT STATEMENTS TO ACCESS DATA FROM MORE THAN ONE TABLE USING EQUIJOINS AND NONEQUIJOINS.

1. The EMPLOYEES and DEPARTMENTS tables have two identically named columns: DEPARTMENT_ID and MANAGER_ID. Which of these statements joins these tables based only on common DEPARTMENT_ID values? (Choose all that apply.)

A) SELECT * FROM EMPLOYEES NATURAL JOIN DEPARTMENTS;

B) SELECT * FROM EMPLOYEES E NATURAL JOIN DEPARTMENTS D ON E.DEPARTMENT _ID=D.DEPARTMENT_ID;

C) SELECT * FROM EMPLOYEES NATURAL JOIN DEPARTMENTS USING (DEPARTMENT_ID);

D) None of the above.

Explanation: D: A, B and C are incorrect. A performs a pure natural join that implicitly joins the two tables on all columns with identical names, which, in this case, are DEPARTMENT_ID and MANAGER_ID.

2. The EMPLOYEES and DEPARTMENTS tables have two identically named columns: DEPARTMENT_ID and MANAGER_ID. Which statements join these tables based on both column values? (Choose all that apply.)

A) SELECT * FROM EMPLOYEES NATURAL JOIN DEPARTMENTS;

B) SELECT * FROM EMPLOYEES JOIN DEPARTMENTS USING (DEPARTMENT _ID,MANAGER_ID);

C) SELECT * FROM EMPLOYEES E JOIN DEPARTMENTS D ON E.DEPARTMENT_ID = D.DEPARTMENT_ID AND E.MANAGER_ID = D.MANAGER_ID;

D) None of the above.

Explanation: A, B, and C. These clauses demonstrate different techniques for joining the tables on the DEPARTMENT_ID and MANAGER_ID columns; That is, all queries perform the same only in a different way.

3. Which join is performed by the following query?
SELECT E.JOB_ID,J.JOB_ID FROM EMPLOYEES E JOIN JOBS J ON (E.SALARY < J.MAX_SALARY); (Choose the best answer.)

A) Equijoin.

B) Nonequijoin.

C) Cross join.

D) Outer join.

Explanation: B, The join condition is an expression based on the less than inequality operator. Therefore, this join is a nonequijoin.

4. Which of the following statements are syntactically correct? (Choose all that apply.)

A) SELECT * FROM EMPLOYEES E JOIN DEPARTMENTS D USING (DEPARTMENT_ID);

B) SELECT * FROM EMPLOYEES JOIN DEPARTMENTS D USING (D.DEPARTMENT_ID);

C) SELECT D.DEPARTMENT_ID FROM EMPLOYEES JOIN DEPARTMENTS D USING (DE-

PARTMENT_ID);

D) None of the above.

Explanation: A, This statement is correct as it demonstrates the proper use of the JOIN USING clause. values).

5. Which of the following statements are syntactically correct? (Choose all that apply.)

A) SELECT E.EMPLOYEE_ID, J.JOB_ID PREVIOUS_JOB, E.JOB_ID CURRENT_JOB FROM JOB_HISTORY J CROSS JOIN EMPLOYEES E ON (J.START_DATE = E.HIRE_DATE);

B) SELECT E.EMPLOYEE_ID, J.JOB_ID PREVIOUS_JOB, E.JOB_ID CURRENT_JOB FROM JOB_HISTORY J JOIN EMPLOYEES E ON (J.START_DATE = E.HIRE_DATE);

C) SELECT E.EMPLOYEE_ID, J.JOB_ID PREVIOUS_JOB, E.JOB_ID CURRENT_JOB FROM JOB_HISTORY J OUTER JOIN EMPLOYEES E ON (J.START_DATE = E.HIRE_DATE);

D) None of the above.

Explanation: B, This option demonstrates the correct use of the JOIN ON clause since the limitations of the clause are met.

6. Choose one correct statement regarding the following query:
SELECT * FROM EMPLOYEES E JOIN DEPARTMENTS D ON (D.DEPARTMENT_ID = E.DEPARTMENT_ID) JOIN LOCATIONS L ON (L.LOCATION_ID = D.LOCATION_ID);

A) Joining three tables is not permitted.

B) A Cartesian product is generated.

C) The JOIN...ON clause may be used for joins between multiple tables.

D) None of the above.

Explanation: C, The JOIN ON clause and the other join clauses can be used for joins between multiple tables. JOIN ON and JOIN USING clauses are more suitable for N-way table joins

JOIN A TABLE TO ITSELF USING A SELF-JOIN.

7. How many rows are returned after executing the following statement? SELECT * FROM REGIONS R1 JOIN REGIONS R2 ON (R1.REGION_ID = LENGTH(R2. REGION_NAME)/2); The REGIONS table contains the following row data. (Choose the best answer.)

A) 2

B) 3

C) 4

D) None of the above.

Explanation: B, Three rows are returned for the row with a REGION_ID value of 2, REGION_NAME is Asia, and half the length of REGION_NAME is also 2, so returning this row results in rows with values of REGION_ID of three and four and REGION_NAME values of Europe and America that are returned.

VIEW DATA THAT DOES NOT MEET A JOIN CONDITION USING OUTER JOINS.

8. Choose one correct statement regarding the following query. SELECT C.COUNTRY_ID FROM LOCATIONS L RIGHT OUTER JOIN COUNTRIES C ON (L.COUNTRY_ID = C.COUNTRY_ID) WHERE L.COUNTRY_ID is NULL;

A) No rows in the LOCATIONS table have the COUNTRY_ID values returned

B) No rows in the COUNTRIES table have the COUNTRY_ID values returned.

C) The rows returned represent the COUNTRY_ID values for all the rows in the LOCATIONS table.

D) None of the above.

Explanation: A, The right outer join fetches the COUNTRIES rows that the inner join between the LOCATIONS and COUNTRIES tables have excluded. The WHERE clause then restricts the results by eliminating the inner join results. This leaves the rows from the COUNTRIES table with which no records from the LOCATIONS table records are associated.

9. Which of the following statements are syntactically correct? (Choose all that apply.)

A) SELECT JH.JOB_ID FROM JOB_HISTORY JH RIGHT OUTER JOIN JOBS J ON JH.JOB_ID = J.JOB_ID

B) SELECT JOB_ID FROM JOB_HISTORY JH RIGHT OUTER JOIN JOBS J ON (JH.JOB_ID = J.JOB_ID)

C) SELECT JOB_HISTORY.JOB_ID FROM JOB_HISTORY OUTER JOIN JOBS ON JOB_HISTORY. JOB_ID = JOBS.JOB_ID

D) None of the above.

Explanation: A, This statement demonstrates the correct use of the RIGHT OUTER JOIN ON clause.

GENERATE A CARTESIAN PRODUCT OF TWO OR MORE TABLES.

10. If the REGIONS table, which contains 4 rows, is cross joined to the COUNTRIES table, which contains 25 rows, how many rows appear in the final results set? (Choose the best answer.)

A) 100 rows

B) 4 rows

C) 25 rows

D) None of the above.

Explanation: A, The cross join associates every four rows from the REGIONS table 25 times with the rows from the COUNTRIES table yielding a result set that contains 100 rows.

Activity 2: Propose an answer to the following issues:

a) You are required to retrieve information from multiple tables, group the results, and apply an aggregate function to them. Can a group function be used against data from multiple table sources?

R = If the rest is possible, since when the tables are joined, a table is obtained as a result and a group can be made to said table, therefore if group functions can be applied.

b) When joining two tables, there is a risk that between them they contain common column names. Does Oracle know which tables to fetch data from if such columns are present in the SELECT list?

R = If Oracle knows as long as the union between tables is done with a NATURAL JOIN or with a correct JOIN.

c) The NATURAL JOIN clause is used to join rows from two tables based on columns with common names sharing identical values. Is it possible to join two tables based on some of the shared columns and not all of them?

R = If this is possible since the USING clause or the ON clause could be used.

d) The data in two tables you wish to join is related but does not share any identically named columns. Is it possible to join tables using columns that do not share the same name?

R = If this is possible since the USING clause or the ON clause could be used.

e) You wish to divide staff into four groups named after the four regions in the REGIONS table. Is it possible to obtain a list of EMPLOYEE_ID, LAST_NAME, and REGION_NAME values for each employee by joining the EMPLOYEE_ID and REGION_ID columns in a round-robin manner?

R = It is not possible since the EMPLOYEES and REGIONS tables do not have any relationship between them, the only thing that could be done is a cross union and this would generate a Cartesian product.

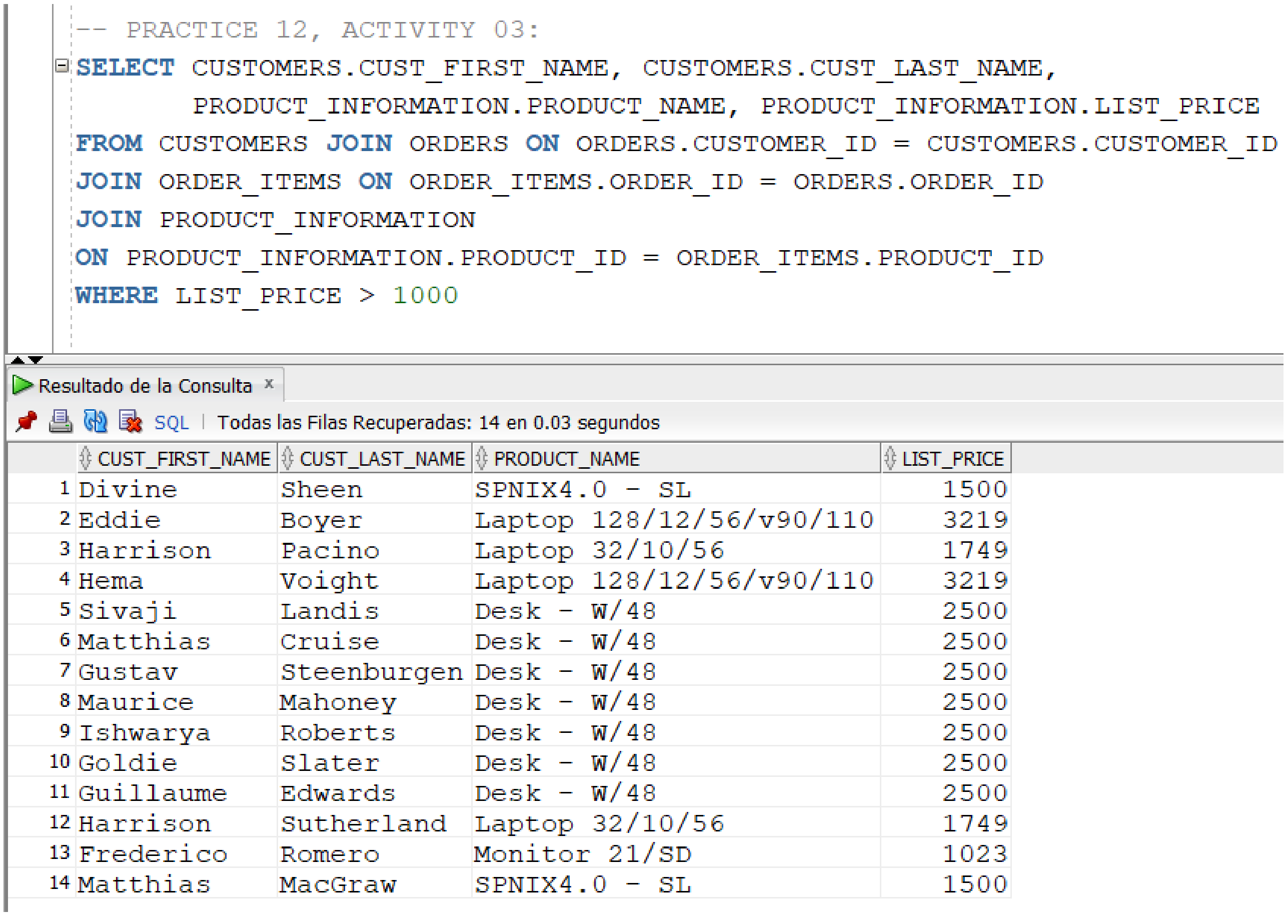
f) You are required to retrieve a list of DEPARTMENT_NAME and LAST_NAME values for all departments, including those that currently have no employees assigned to them. In such cases the string 'No Employees' should be displayed as the LAST_NAME column value. Can this be done using joins?

R = If this is possible, since a union could be made between the EMPLOYEES table and the DEPARTMENTS table using the USING clause with the id of the departments and to show the phrase without employees, the NVL function could be used.

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

- You are required to produce a report of customers who purchased products with list prices of more than $1000. The report must contain customer first and last names and the product names and their list prices. Customer information is stored in the CUSTOMERS table, which has the CUSTOMER_ID column as its primary key. The product name and list price details are stored in the PRODUCT_INFORMATION table with the PRODUCT_ID column as its primary key. Two other related tables may assist in generating the required report: the ORDERS table, which stores the CUSTOMER_ID and ORDER_ID information, and the ORDER_ITEMS table, which stores the PRODUCT_ID values associated with each ORDER_ID. There are several approaches to solving this question, **see figure 1**.

Note: Capture an image for each statement output.



```
-- PRACTICE 12, ACTIVITY 03:
SELECT CUSTOMERS.CUST_FIRST_NAME, CUSTOMERS.CUST_LAST_NAME,
       PRODUCT_INFORMATION.PRODUCT_NAME, PRODUCT_INFORMATION.LIST_PRICE
FROM CUSTOMERS JOIN ORDERS ON ORDERS.CUSTOMER_ID = CUSTOMERS.CUSTOMER_ID
JOIN ORDER_ITEMS ON ORDER_ITEMS.ORDER_ID = ORDERS.ORDER_ID
JOIN PRODUCT_INFORMATION
ON PRODUCT_INFORMATION.PRODUCT_ID = ORDER_ITEMS.PRODUCT_ID
WHERE LIST_PRICE > 1000
```

Resultado de la Consulta

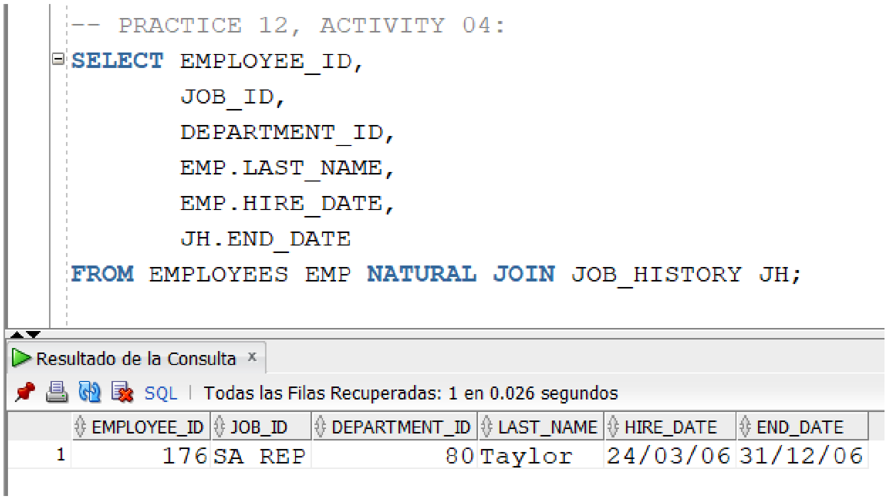
SQL | Todas las Filas Recuperadas: 14 en 0.03 segundos

	CUST_FIRST_NAME	CUST_LAST_NAME	PRODUCT_NAME	LIST_PRICE
1	Divine	Sheen	SPNIX4.0 - SL	1500
2	Eddie	Boyer	Laptop 128/12/56/v90/110	3219
3	Harrison	Pacino	Laptop 32/10/56	1749
4	Hema	Voight	Laptop 128/12/56/v90/110	3219
5	Sivaji	Landis	Desk - W/48	2500
6	Matthias	Cruise	Desk - W/48	2500
7	Gustav	Steenburgen	Desk - W/48	2500
8	Maurice	Mahoney	Desk - W/48	2500
9	Ishwarya	Roberts	Desk - W/48	2500
10	Goldie	Slater	Desk - W/48	2500
11	Guillaume	Edwards	Desk - W/48	2500
12	Harrison	Sutherland	Laptop 32/10/56	1749
13	Frederico	Romero	Monitor 21/SD	1023
14	Matthias	MacGraw	SPNIX4.0 - SL	1500

Figure 1: *Data query with multiple tables.*

Activity 4: This exercise must be performed in the HR schema.

- The JOB_HISTORY table shares three identically named columns with the EMPLOYEES table: EMPLOYEE_ID, JOB_ID, and DEPARTMENT_ID. You are required to describe the tables and fetch the EMPLOYEE_ID, JOB_ID, DEPARTMENT_ID, LAST_NAME, HIRE_DATE, and END_DATE values for all rows retrieved using a pure natural join. Alias the EMPLOYEES table as EMP and the JOB_HISTORY table as JH and use dot notation where possible, **see figure 2**.



```
-- PRACTICE 12, ACTIVITY 04:
SELECT EMPLOYEE_ID,
       JOB_ID,
       DEPARTMENT_ID,
       EMP.LAST_NAME,
       EMP.HIRE_DATE,
       JH.END_DATE
FROM EMPLOYEES EMP NATURAL JOIN JOB_HISTORY JH;
```

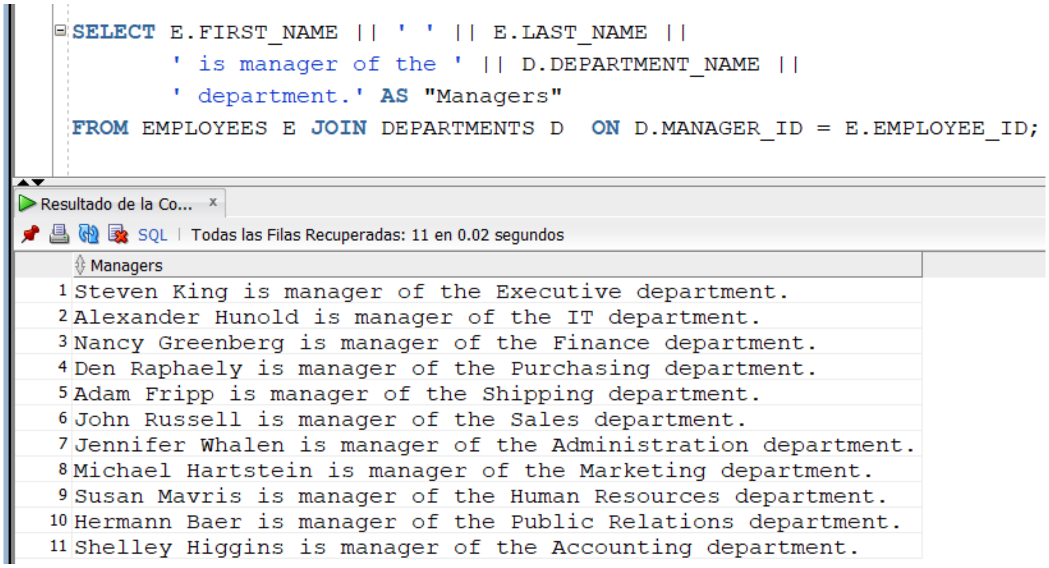
Resultado de la Consulta

SQL | Todas las Filas Recuperadas: 1 en 0.026 segundos

	EMPLOYEE_ID	JOB_ID	DEPARTMENT_ID	LAST_NAME	HIRE_DATE	END_DATE
1	176	SA_REP	80	Taylor	24/03/06	31/12/06

Figure 2: *Data query with multiple tables.*

- Each record in the DEPARTMENTS table has a MANAGER_ID column matching an EMPLOYEE_ID value in the EMPLOYEES table. You are required to produce a report with one column aliased as Managers. Each row must contain a sentence of the format FIRST_NAME LAST_NAME is manager of the DEPARTMENT_NAME department. Alias the EMPLOYEES table as E and the DEPARTMENTS table as D and use dot notation where possible, **see figure 3**.



```
SELECT E.FIRST_NAME || ' ' || E.LAST_NAME ||
       ' is manager of the ' || D.DEPARTMENT_NAME ||
       ' department.' AS "Managers"
FROM EMPLOYEES E JOIN DEPARTMENTS D  ON D.MANAGER_ID = E.EMPLOYEE_ID;
```

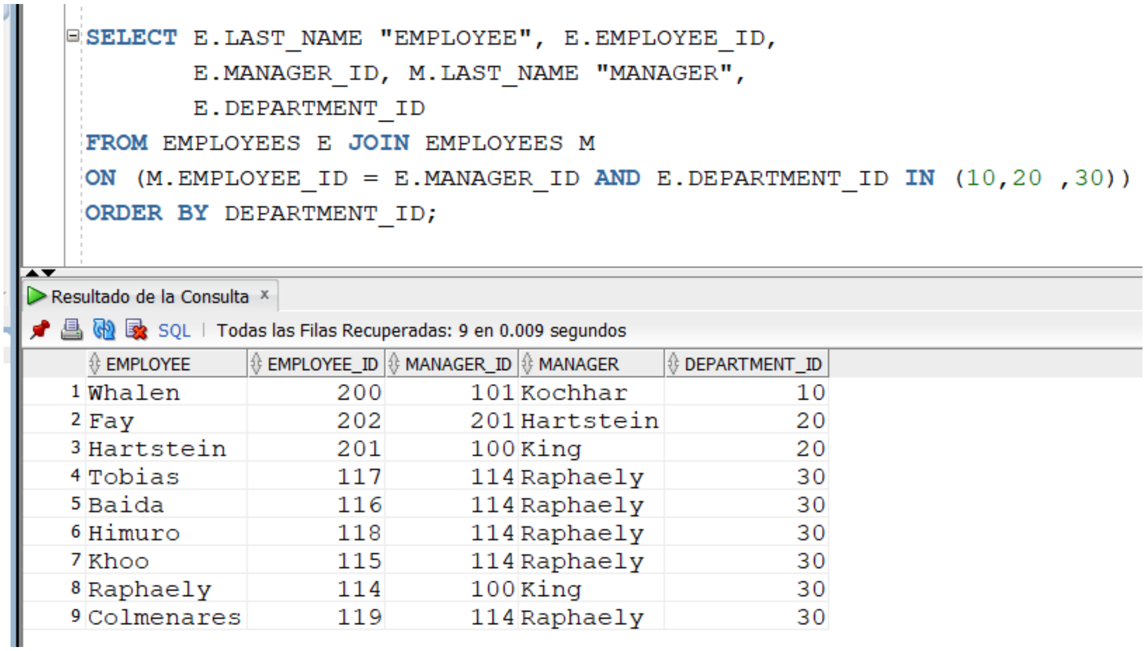
Resultado de la Co...

SQL | Todas las Filas Recuperadas: 11 en 0.02 segundos

	Managers
1	Steven King is manager of the Executive department.
2	Alexander Hunold is manager of the IT department.
3	Nancy Greenberg is manager of the Finance department.
4	Den Raphaely is manager of the Purchasing department.
5	Adam Fripp is manager of the Shipping department.
6	John Russell is manager of the Sales department.
7	Jennifer Whalen is manager of the Administration department.
8	Michael Hartstein is manager of the Marketing department.
9	Susan Mavris is manager of the Human Resources department.
10	Hermann Baer is manager of the Public Relations department.
11	Shelley Higgins is manager of the Accounting department.

Figure 3: *Data query with multiple tables.*

- There is a hierarchical relationship between employees and their managers. For each row in the EMPLOYEES table the MANAGER_ID column stores the EMPLOYEE_ID of every employee's manager. Using a self-join on the EMPLOYEES table, you are required to retrieve the employee's LAST_NAME, EMPLOYEE_ID, manager's LAST_NAME, and employee's DEPARTMENT_ID for the rows with DEPARMENT_ID values of 10, 20, or 30. Alias the EMPLOYEES table as E and the second instance of the EMPLOYEES table as M. Sort the results based on the DEPARTMENT_ID column, **see figure 4**.



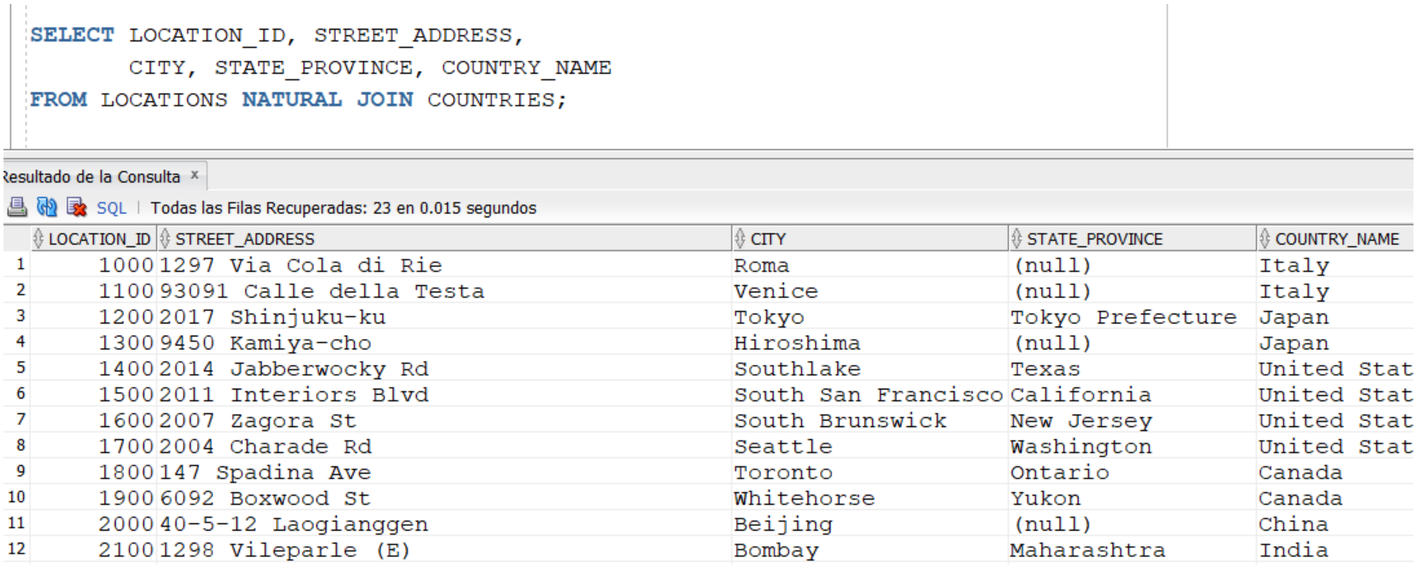
```
SELECT E.LAST_NAME "EMPLOYEE", E.EMPLOYEE_ID,
       E.MANAGER_ID, M.LAST_NAME "MANAGER",
       E.DEPARTMENT_ID
FROM EMPLOYEES E JOIN EMPLOYEES M
ON (M.EMPLOYEE_ID = E.MANAGER_ID AND E.DEPARTMENT_ID IN (10,20 ,30))
ORDER BY DEPARTMENT_ID;
```

Todas las Filas Recuperadas: 9 en 0.009 segundos

	EMPLOYEE	EMPLOYEE_ID	MANAGER_ID	MANAGER	DEPARTMENT_ID
1	Whalen	200	101	Kochhar	10
2	Fay	202	201	Hartstein	20
3	Hartstein	201	100	King	20
4	Tobias	117	114	Raphaely	30
5	Baida	116	114	Raphaely	30
6	Himuro	118	114	Raphaely	30
7	Khoo	115	114	Raphaely	30
8	Raphaely	114	100	King	30
9	Colmenares	119	114	Raphaely	30

Figure 4: *Data query with multiple tables.*

- Write a query for the HR department to produce the addresses of all the departments. Use the LOCATIONS and COUNTRIES tables. Show the location ID, street address, city, state or province, and country in the output. Use a NATURAL JOIN to produce the results, **see figure 5**.



```
SELECT LOCATION_ID, STREET_ADDRESS,
       CITY, STATE_PROVINCE, COUNTRY_NAME
FROM LOCATIONS NATURAL JOIN COUNTRIES;
```

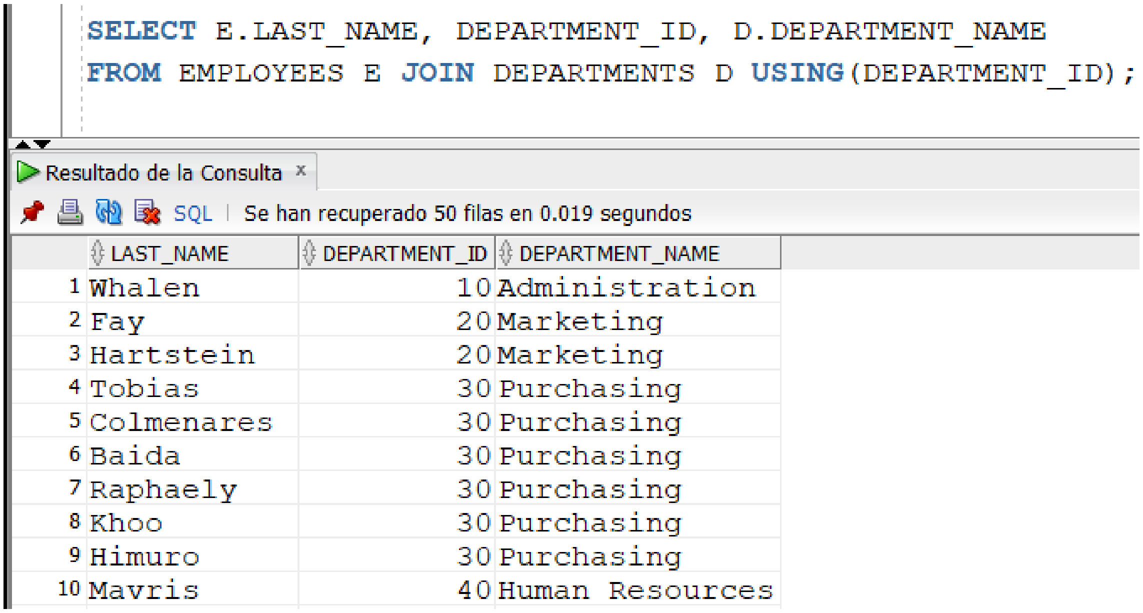
Resultado de la Consulta ×

SQL | Todas las Filas Recuperadas: 23 en 0.015 segundos

	LOCATION_ID	STREET_ADDRESS	CITY	STATE_PROVINCE	COUNTRY_NAME
1	1000	1297 Via Cola di Rie	Roma	(null)	Italy
2	1100	93091 Calle della Testa	Venice	(null)	Italy
3	1200	2017 Shinjuku-ku	Tokyo	Tokyo Prefecture	Japan
4	1300	9450 Kamiya-cho	Hiroshima	(null)	Japan
5	1400	2014 Jabberwocky Rd	Southlake	Texas	United Stat
6	1500	2011 Interiors Blvd	South San Francisco	California	United Stat
7	1600	2007 Zagora St	South Brunswick	New Jersey	United Stat
8	1700	2004 Charade Rd	Seattle	Washington	United Stat
9	1800	147 Spadina Ave	Toronto	Ontario	Canada
10	1900	6092 Boxwood St	Whitehorse	Yukon	Canada
11	2000	40-5-12 Laogianggen	Beijing	(null)	China
12	2100	1298 Vileparle (E)	Bombay	Maharashtra	India

Figure 5: *Data query with multiple tables.*

- The HR department needs a report of all employees. Write a query to display the last name, department number, and department name for all the employees, **see figure 6**.



```
SELECT E.LAST_NAME, DEPARTMENT_ID, D.DEPARTMENT_NAME
FROM EMPLOYEES E JOIN DEPARTMENTS D USING(DEPARTMENT_ID);
```

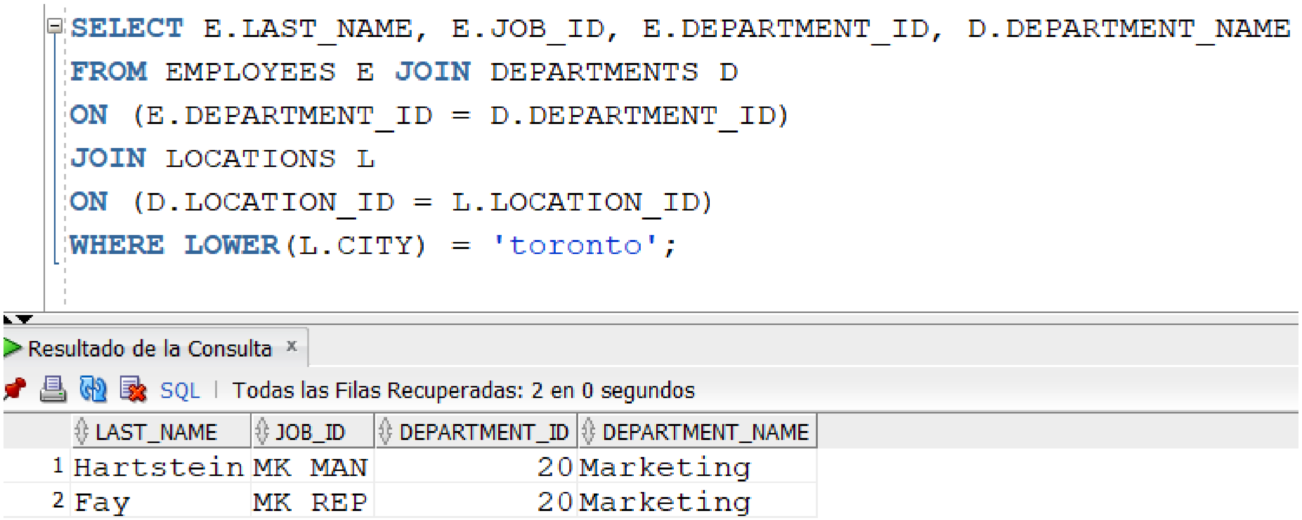
Resultado de la Consulta ×

SQL | Se han recuperado 50 filas en 0.019 segundos

	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
1	Whalen	10	Administration
2	Fay	20	Marketing
3	Hartstein	20	Marketing
4	Tobias	30	Purchasing
5	Colmenares	30	Purchasing
6	Baida	30	Purchasing
7	Raphaely	30	Purchasing
8	Khoo	30	Purchasing
9	Himuro	30	Purchasing
10	Mavris	40	Human Resources

Figure 6: *Data query with multiple tables.*

- The HR department needs a report to display the last name, job, department number, and department name for all employees who work in a city entered by a user parameter. For instance, city equals ’toronto’, **see figure 7**.



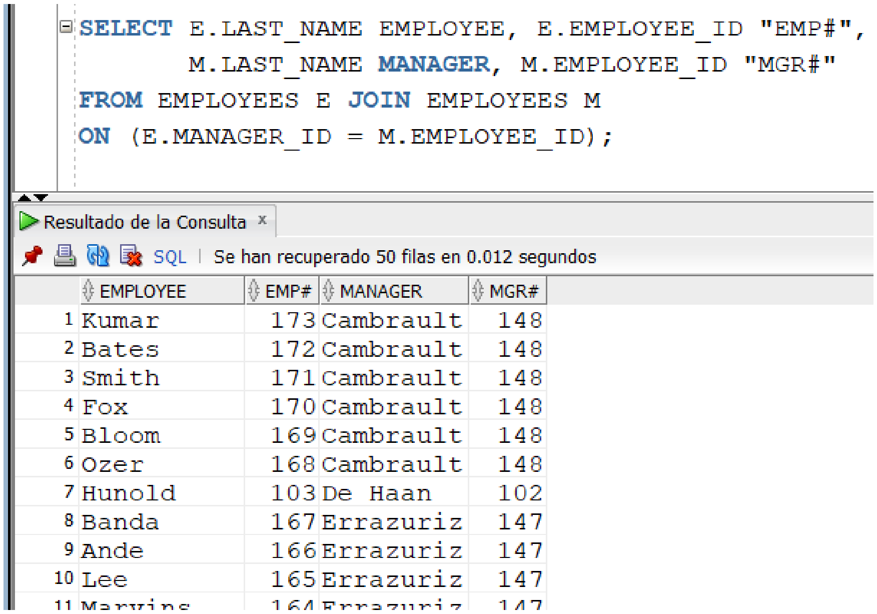
```
SELECT E.LAST_NAME, E.JOB_ID, E.DEPARTMENT_ID, D.DEPARTMENT_NAME
FROM EMPLOYEES E JOIN DEPARTMENTS D
ON (E.DEPARTMENT_ID = D.DEPARTMENT_ID)
JOIN LOCATIONS L
ON (D.LOCATION_ID = L.LOCATION_ID)
WHERE LOWER(L.CITY) = 'toronto';
```

Todas las Filas Recuperadas: 2 en 0 segundos

```
    LAST_NAME  JOB_ID  DEPARTMENT_ID DEPARTMENT_NAME
  1 Hartstein  MK_MAN             20 Marketing
  2 Fay        MK_REP             20 Marketing
```

Figure 7: *Data query with multiple tables.*

- Crete a report to display employees’ last names and employee number along with their managers’ last names and manager number. Label the columns Employee, Emp , Manager, and Mgr, respectively, **see figure 8**.



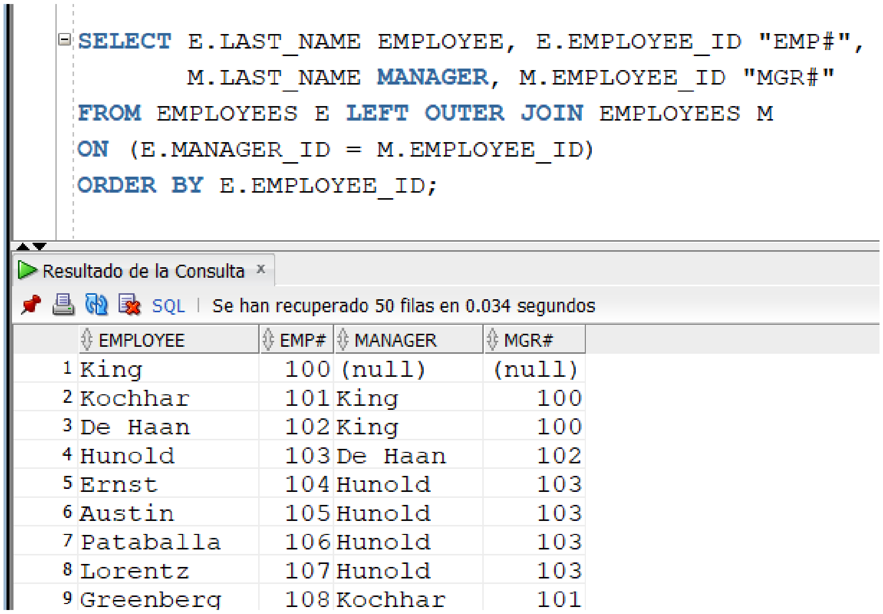
```sql
SELECT E.LAST_NAME EMPLOYEE, E.EMPLOYEE_ID "EMP#",
       M.LAST_NAME MANAGER, M.EMPLOYEE_ID "MGR#"
FROM EMPLOYEES E JOIN EMPLOYEES M
ON (E.MANAGER_ID = M.EMPLOYEE_ID);
```

Se han recuperado 50 filas en 0.012 segundos

	EMPLOYEE	EMP#	MANAGER	MGR#
1	Kumar	173	Cambrault	148
2	Bates	172	Cambrault	148
3	Smith	171	Cambrault	148
4	Fox	170	Cambrault	148
5	Bloom	169	Cambrault	148
6	Ozer	168	Cambrault	148
7	Hunold	103	De Haan	102
8	Banda	167	Errazuriz	147
9	Ande	166	Errazuriz	147
10	Lee	165	Errazuriz	147
11	Marvins	164	Errazuriz	147

Figure 8: *Data query with multiple tables.*

- Modify previous sentence to display all employees including those who have no manager. Order the results by the employee number, **see figure 9**.



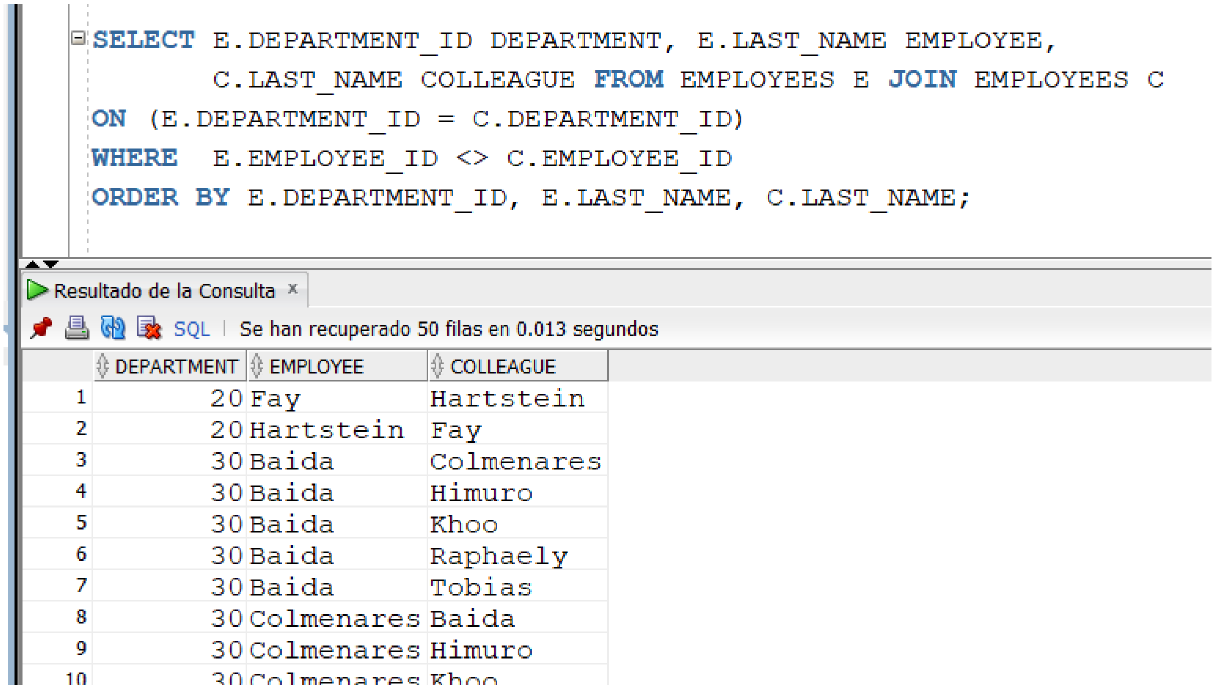
```sql
SELECT E.LAST_NAME EMPLOYEE, E.EMPLOYEE_ID "EMP#",
       M.LAST_NAME MANAGER, M.EMPLOYEE_ID "MGR#"
FROM EMPLOYEES E LEFT OUTER JOIN EMPLOYEES M
ON (E.MANAGER_ID = M.EMPLOYEE_ID)
ORDER BY E.EMPLOYEE_ID;
```

Se han recuperado 50 filas en 0.034 segundos

	EMPLOYEE	EMP#	MANAGER	MGR#
1	King	100	(null)	(null)
2	Kochhar	101	King	100
3	De Haan	102	King	100
4	Hunold	103	De Haan	102
5	Ernst	104	Hunold	103
6	Austin	105	Hunold	103
7	Pataballa	106	Hunold	103
8	Lorentz	107	Hunold	103
9	Greenberg	108	Kochhar	101

Figure 9: *Data query with multiple tables.*

- Create a report for the HR department that displays employee last names, department name for all the employees along with their colleagues. Give each column an appropriate label and order the results as you consider appropriate to the study case, **see figure 10**.



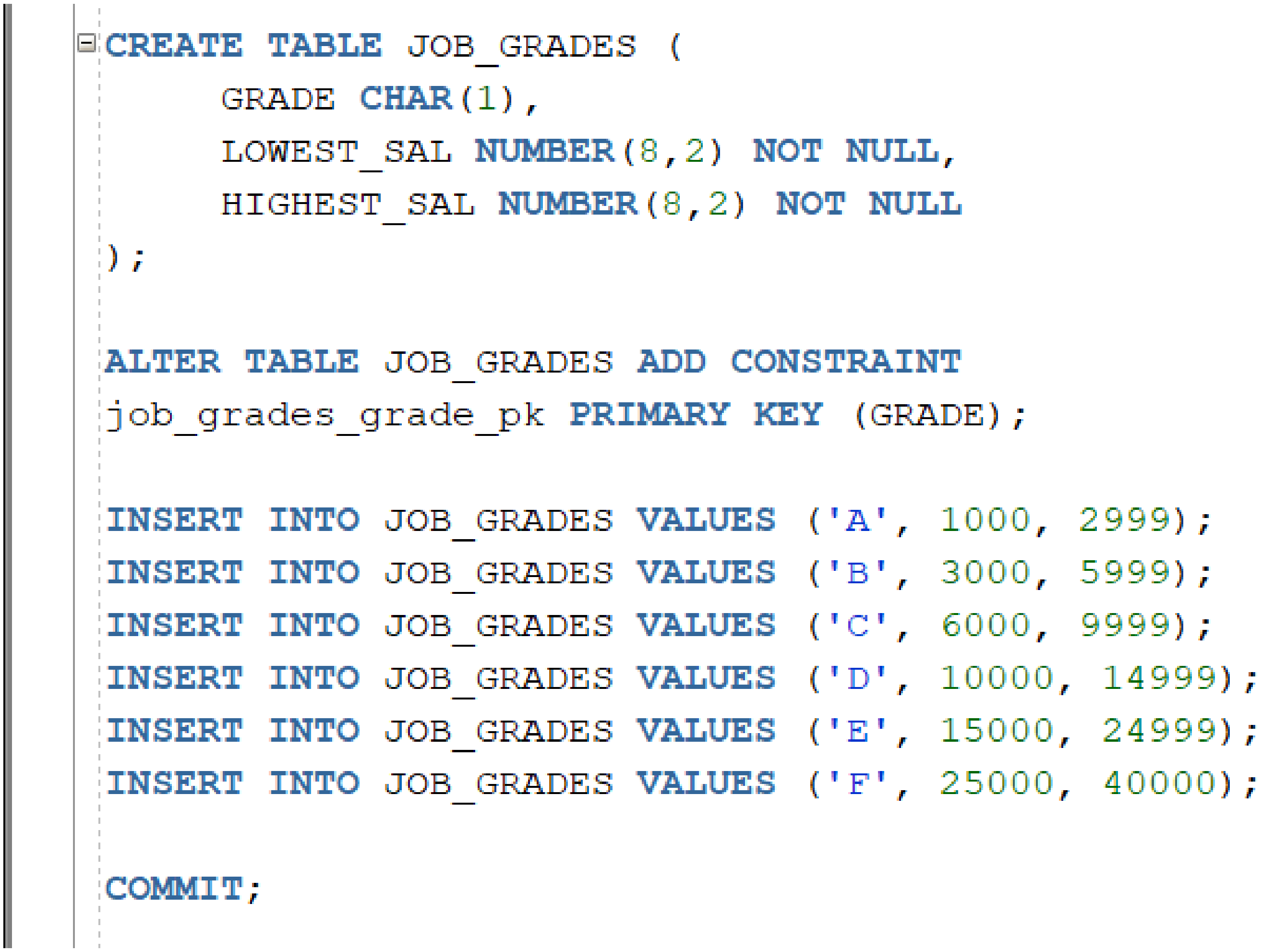
```
SELECT E.DEPARTMENT_ID DEPARTMENT, E.LAST_NAME EMPLOYEE,
       C.LAST_NAME COLLEAGUE FROM EMPLOYEES E JOIN EMPLOYEES C
ON (E.DEPARTMENT_ID = C.DEPARTMENT_ID)
WHERE  E.EMPLOYEE_ID <> C.EMPLOYEE_ID
ORDER BY E.DEPARTMENT_ID, E.LAST_NAME, C.LAST_NAME;
```

Se han recuperado 50 filas en 0.013 segundos

	DEPARTMENT	EMPLOYEE	COLLEAGUE
1	20	Fay	Hartstein
2	20	Hartstein	Fay
3	30	Baida	Colmenares
4	30	Baida	Himuro
5	30	Baida	Khoo
6	30	Baida	Raphaely
7	30	Baida	Tobias
8	30	Colmenares	Baida
9	30	Colmenares	Himuro
10	30	Colmenares	Khoo

Figure 10: *Data query with multiple tables.*

- The HR department needs a report on job grades and salaries. To familiarize yourself with the JOB_GRADES table, first show the structure of the JOB_GRADES table. Then create a query that displays the name, job, department name, salary, and grade for all employees, **see figure 11 and 12**.



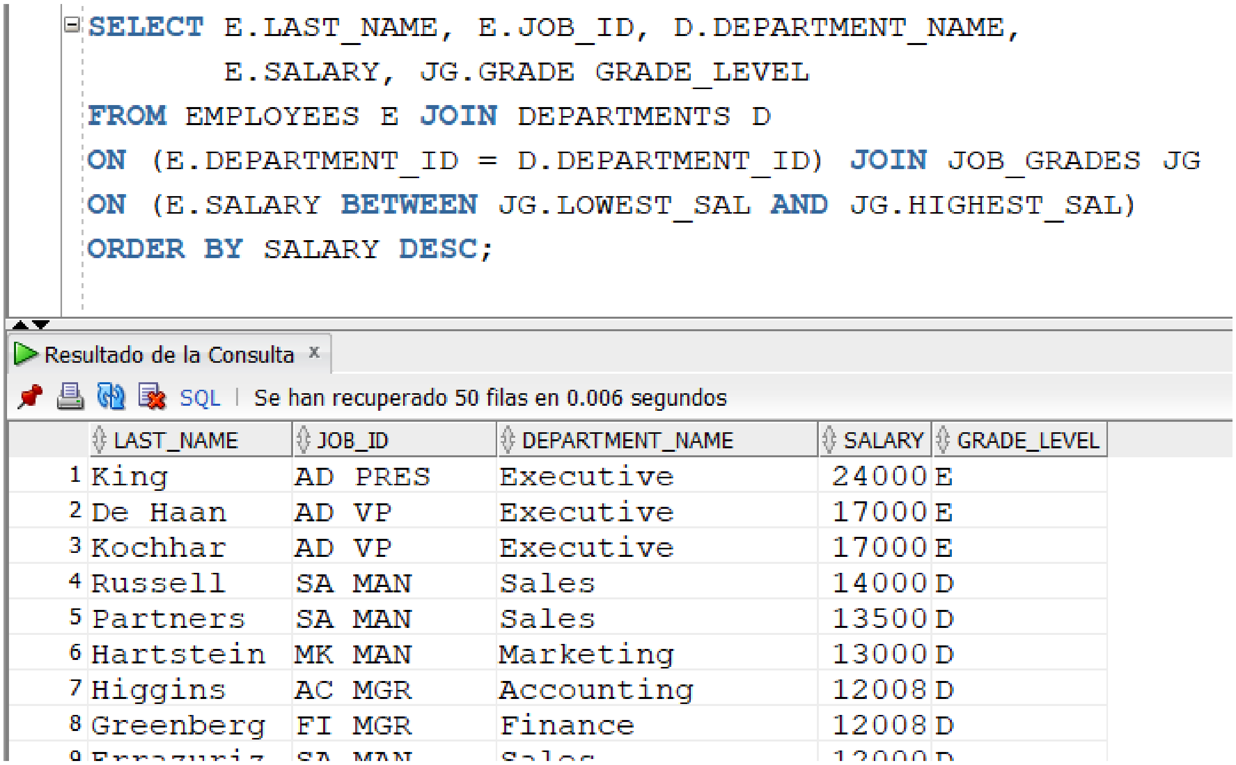
```
CREATE TABLE JOB_GRADES (
     GRADE CHAR(1),
     LOWEST_SAL NUMBER(8,2) NOT NULL,
     HIGHEST_SAL NUMBER(8,2) NOT NULL
);

ALTER TABLE JOB_GRADES ADD CONSTRAINT
job_grades_grade_pk PRIMARY KEY (GRADE);

INSERT INTO JOB_GRADES VALUES ('A', 1000, 2999);
INSERT INTO JOB_GRADES VALUES ('B', 3000, 5999);
INSERT INTO JOB_GRADES VALUES ('C', 6000, 9999);
INSERT INTO JOB_GRADES VALUES ('D', 10000, 14999);
INSERT INTO JOB_GRADES VALUES ('E', 15000, 24999);
INSERT INTO JOB_GRADES VALUES ('F', 25000, 40000);

COMMIT;
```

Figure 11: *Creating a new table: HR Schema.*



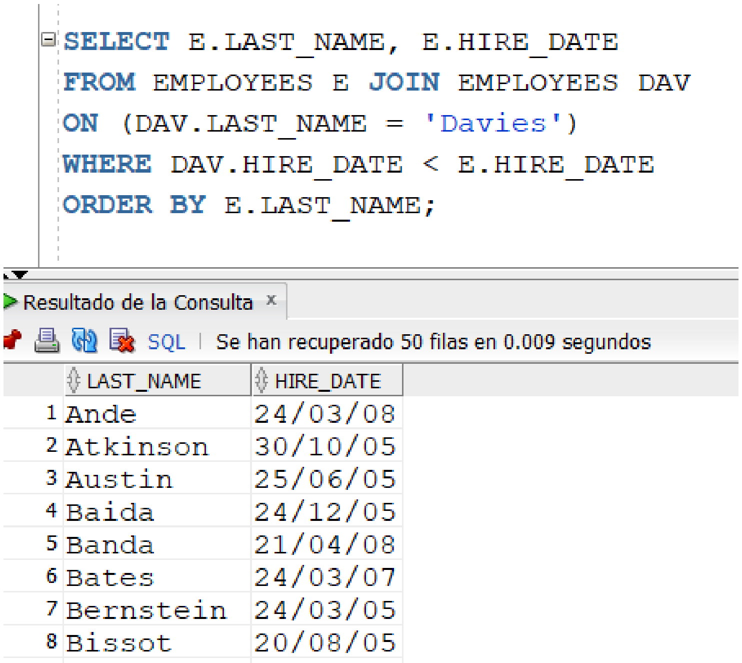
```sql
SELECT E.LAST_NAME, E.JOB_ID, D.DEPARTMENT_NAME,
       E.SALARY, JG.GRADE GRADE_LEVEL
FROM EMPLOYEES E JOIN DEPARTMENTS D
ON (E.DEPARTMENT_ID = D.DEPARTMENT_ID) JOIN JOB_GRADES JG
ON (E.SALARY BETWEEN JG.LOWEST_SAL AND JG.HIGHEST_SAL)
ORDER BY SALARY DESC;
```

Se han recuperado 50 filas en 0.006 segundos

	LAST_NAME	JOB_ID	DEPARTMENT_NAME	SALARY	GRADE_LEVEL
1	King	AD_PRES	Executive	24000	E
2	De Haan	AD_VP	Executive	17000	E
3	Kochhar	AD_VP	Executive	17000	E
4	Russell	SA_MAN	Sales	14000	D
5	Partners	SA_MAN	Sales	13500	D
6	Hartstein	MK_MAN	Marketing	13000	D
7	Higgins	AC_MGR	Accounting	12008	D
8	Greenberg	FI_MGR	Finance	12008	D
9	Errazuriz	SA_MAN	Sales	12000	D

Figure 12: *Data query with multiple tables.*

- HR department wants to create a query to display the name and hire date of any employee hired after an employee entered by a user parameter. For Instance, after ‘Davies’, **see figure 13**.



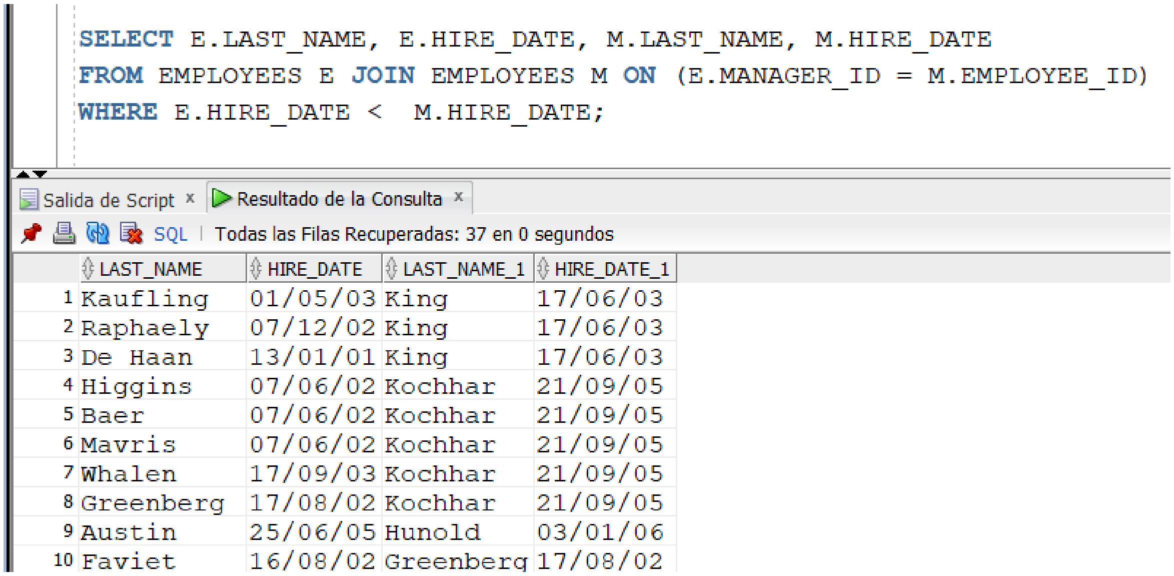
```
SELECT E.LAST_NAME, E.HIRE_DATE
FROM EMPLOYEES E JOIN EMPLOYEES DAV
ON (DAV.LAST_NAME = 'Davies')
WHERE DAV.HIRE_DATE < E.HIRE_DATE
ORDER BY E.LAST_NAME;
```

Se han recuperado 50 filas en 0.009 segundos

	LAST_NAME	HIRE_DATE
1	Ande	24/03/08
2	Atkinson	30/10/05
3	Austin	25/06/05
4	Baida	24/12/05
5	Banda	21/04/08
6	Bates	24/03/07
7	Bernstein	24/03/05
8	Bissot	20/08/05

Figure 13: *Data query with multiple tables.*

- The HR department needs to find the names and hire dates for all employees who were hired before their managers, along with their managers' names and hire dates, **see figure 14**.



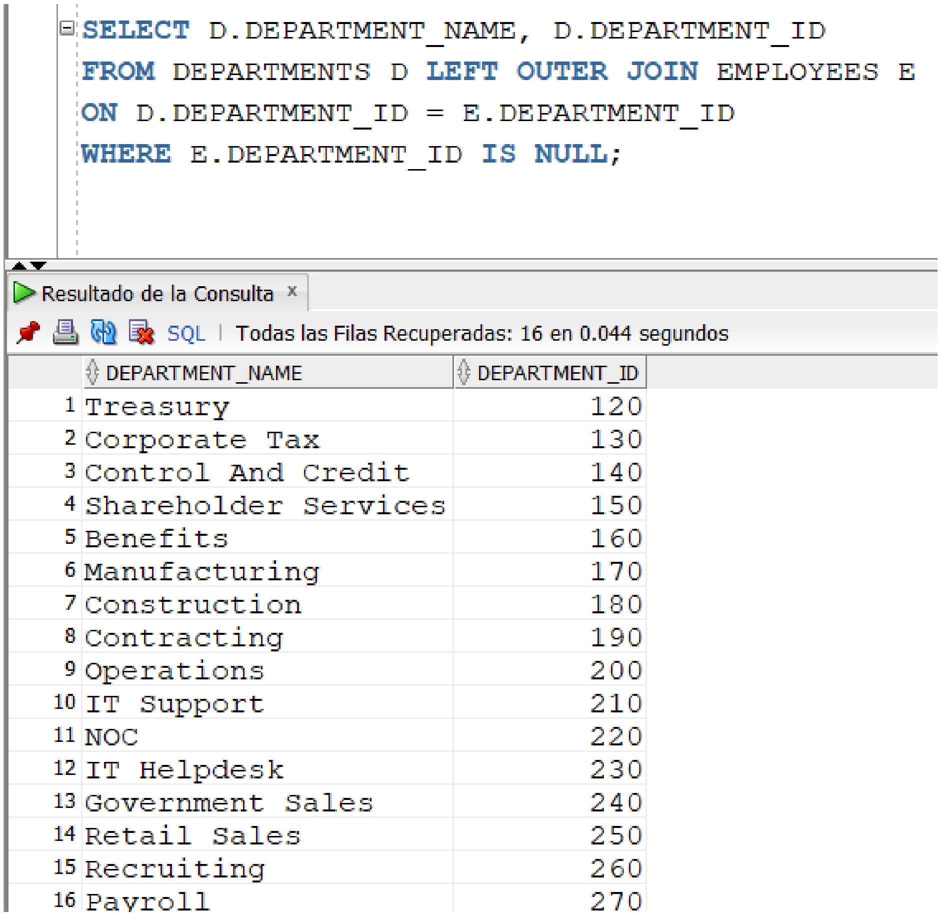
```
SELECT E.LAST_NAME, E.HIRE_DATE, M.LAST_NAME, M.HIRE_DATE
FROM EMPLOYEES E JOIN EMPLOYEES M ON (E.MANAGER_ID = M.EMPLOYEE_ID)
WHERE E.HIRE_DATE <  M.HIRE_DATE;
```

Todas las Filas Recuperadas: 37 en 0 segundos

	LAST_NAME	HIRE_DATE	LAST_NAME_1	HIRE_DATE_1
1	Kaufling	01/05/03	King	17/06/03
2	Raphaely	07/12/02	King	17/06/03
3	De Haan	13/01/01	King	17/06/03
4	Higgins	07/06/02	Kochhar	21/09/05
5	Baer	07/06/02	Kochhar	21/09/05
6	Mavris	07/06/02	Kochhar	21/09/05
7	Whalen	17/09/03	Kochhar	21/09/05
8	Greenberg	17/08/02	Kochhar	21/09/05
9	Austin	25/06/05	Hunold	03/01/06
10	Faviet	16/08/02	Greenberg	17/08/02

Figure 14: *Data query with multiple tables.*

- The DEPARTMENTS table contains details of all departments in the organization. You are required to retrieve the DEPARTMENT_NAME and DEPARTMENT_ID values for those departments to which no employees are currently assigned, **see figure 15**.



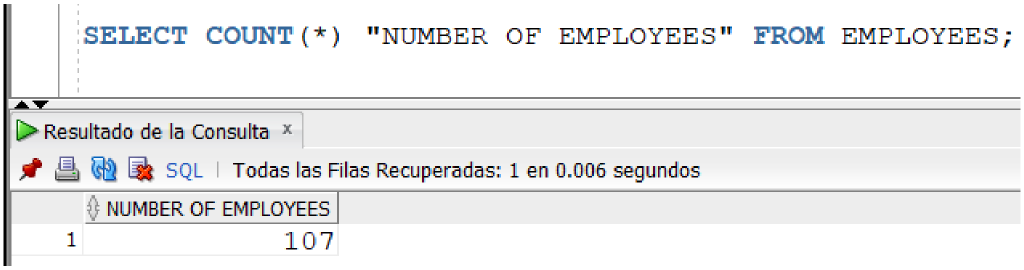
```
SELECT D.DEPARTMENT_NAME, D.DEPARTMENT_ID
FROM DEPARTMENTS D LEFT OUTER JOIN EMPLOYEES E
ON D.DEPARTMENT_ID = E.DEPARTMENT_ID
WHERE E.DEPARTMENT_ID IS NULL;
```

Todas las Filas Recuperadas: 16 en 0.044 segundos

	DEPARTMENT_NAME	DEPARTMENT_ID
1	Treasury	120
2	Corporate Tax	130
3	Control And Credit	140
4	Shareholder Services	150
5	Benefits	160
6	Manufacturing	170
7	Construction	180
8	Contracting	190
9	Operations	200
10	IT Support	210
11	NOC	220
12	IT Helpdesk	230
13	Government Sales	240
14	Retail Sales	250
15	Recruiting	260
16	Payroll	270

Figure 15: *Data query with multiple tables.*

- You are required to obtain the number of rows in the EMPLOYEES and DEPARTMENTS table as well as the number of records that would be created by a Cartesian product of these two tables. Confirm your results by explicitly counting and multiplying the number of rows present in each of these tables. In **figure 16** we can see that there are 107 employees, in **figure 17** it can be seen that there are 27 departments, therefore when performing a Cartesian product (107 x 27) a total of 2889 output records will be obtained, **see figure 18**.

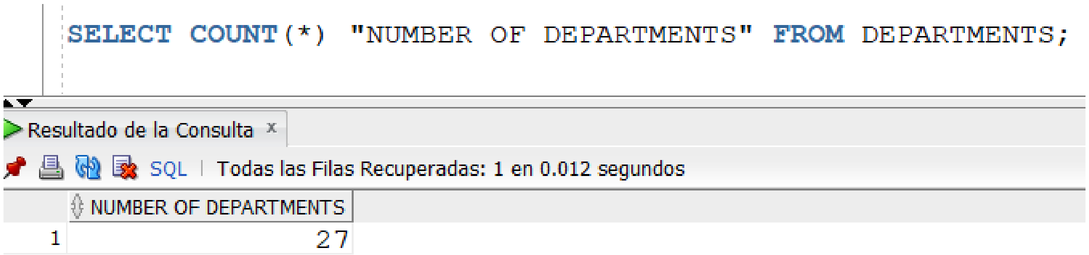


```
SELECT COUNT(*) "NUMBER OF EMPLOYEES" FROM EMPLOYEES;
```

SQL | Todas las Filas Recuperadas: 1 en 0.006 segundos

	NUMBER OF EMPLOYEES
1	107

Figure 16: *Count Records: Employees.*

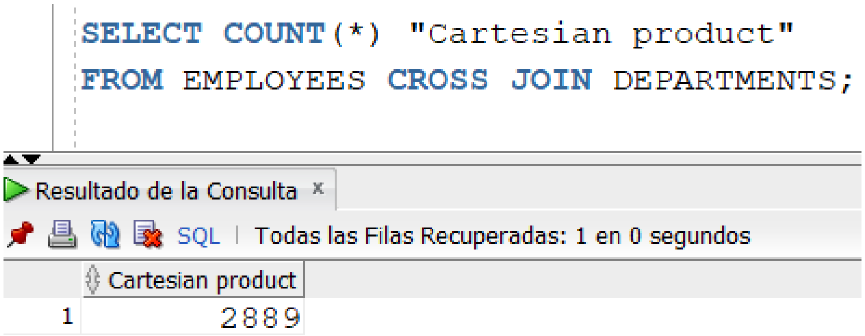


```
SELECT COUNT(*) "NUMBER OF DEPARTMENTS" FROM DEPARTMENTS;
```

SQL | Todas las Filas Recuperadas: 1 en 0.012 segundos

	NUMBER OF DEPARTMENTS
1	27

Figure 17: *Count Records: Departments.*



```
SELECT COUNT(*) "Cartesian product"
FROM EMPLOYEES CROSS JOIN DEPARTMENTS;
```

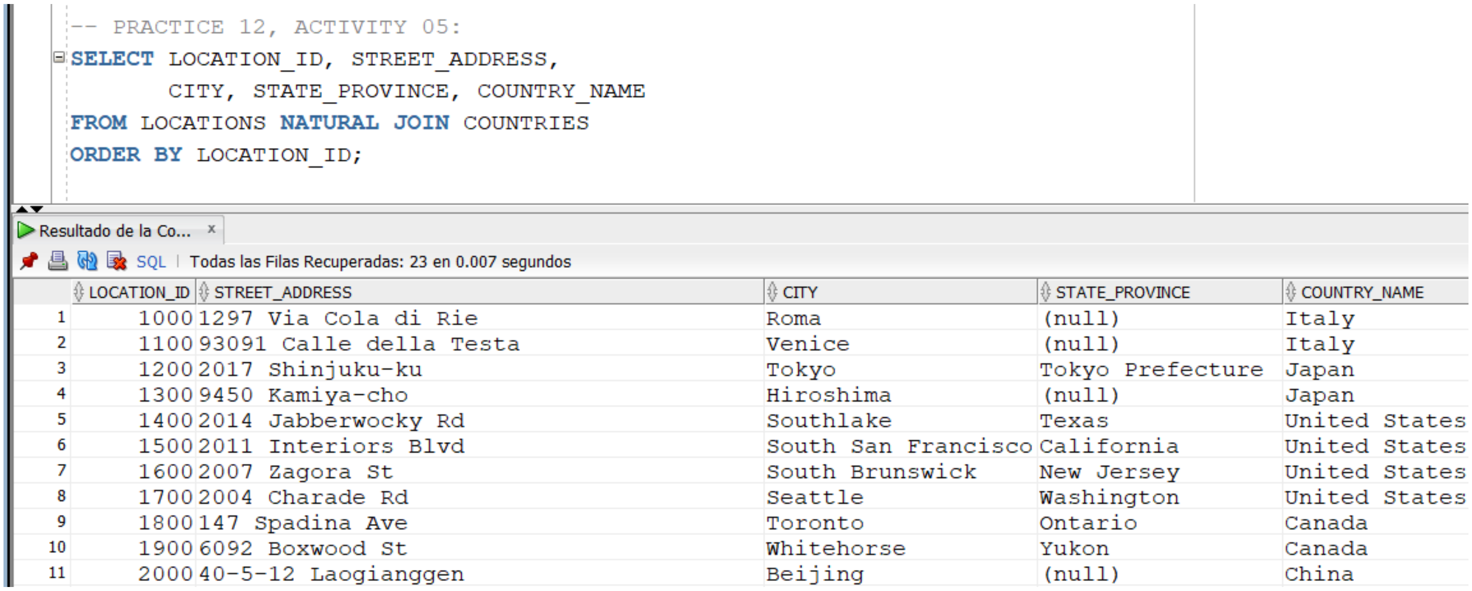
SQL | Todas las Filas Recuperadas: 1 en 0 segundos

	Cartesian product
1	2889

Figure 18: *Cartesian product between employees and departments.*

Activity 5: This practice is intended to give you experience in extracting data from more than one table using the SQL:1999–compliant joins.

1. Write a query for the HR department to produce the addresses of all the departments. Use the LOCATIONS and COUNTRIES tables. Show the location_ID, street address, city, state or province, and country in the output. Use a NATURAL JOIN to produce the results, **see figure 19**.



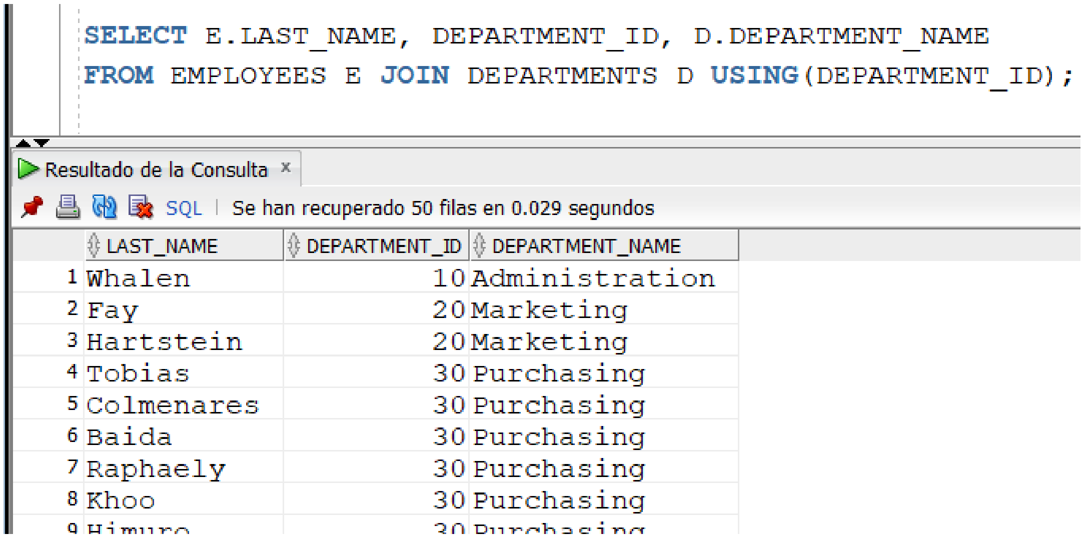
```
-- PRACTICE 12, ACTIVITY 05:
SELECT LOCATION_ID, STREET_ADDRESS,
       CITY, STATE_PROVINCE, COUNTRY_NAME
FROM LOCATIONS NATURAL JOIN COUNTRIES
ORDER BY LOCATION_ID;
```

Todas las Filas Recuperadas: 23 en 0.007 segundos

	LOCATION_ID	STREET_ADDRESS	CITY	STATE_PROVINCE	COUNTRY_NAME
1	1000	1297 Via Cola di Rie	Roma	(null)	Italy
2	1100	93091 Calle della Testa	Venice	(null)	Italy
3	1200	2017 Shinjuku-ku	Tokyo	Tokyo Prefecture	Japan
4	1300	9450 Kamiya-cho	Hiroshima	(null)	Japan
5	1400	2014 Jabberwocky Rd	Southlake	Texas	United States
6	1500	2011 Interiors Blvd	South San Francisco	California	United States
7	1600	2007 Zagora St	South Brunswick	New Jersey	United States
8	1700	2004 Charade Rd	Seattle	Washington	United States
9	1800	147 Spadina Ave	Toronto	Ontario	Canada
10	1900	6092 Boxwood St	Whitehorse	Yukon	Canada
11	2000	40-5-12 Laogianggen	Beijing	(null)	China

Figure 19: *Data query with multiple tables.*

2. The HR department needs a report of all employees. Write a query to display the last name, department number, and department name for all the employees, **see figure 20**.



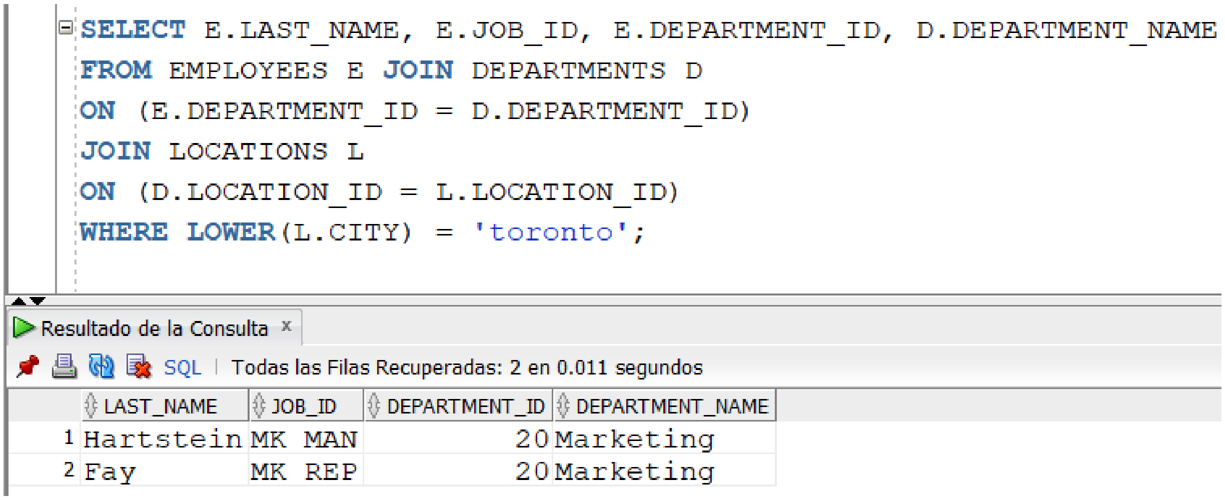
```
SELECT E.LAST_NAME, DEPARTMENT_ID, D.DEPARTMENT_NAME
FROM EMPLOYEES E JOIN DEPARTMENTS D USING(DEPARTMENT_ID);
```

Se han recuperado 50 filas en 0.029 segundos

	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
1	Whalen	10	Administration
2	Fay	20	Marketing
3	Hartstein	20	Marketing
4	Tobias	30	Purchasing
5	Colmenares	30	Purchasing
6	Baida	30	Purchasing
7	Raphaely	30	Purchasing
8	Khoo	30	Purchasing
9	Himuro	30	Purchasing

Figure 20: *Data query with multiple tables.*

3. The HR department needs a report of employees in Toronto. Display the last name, job, department number, and the department name for all employees who work in Toronto, **see figure 21**.



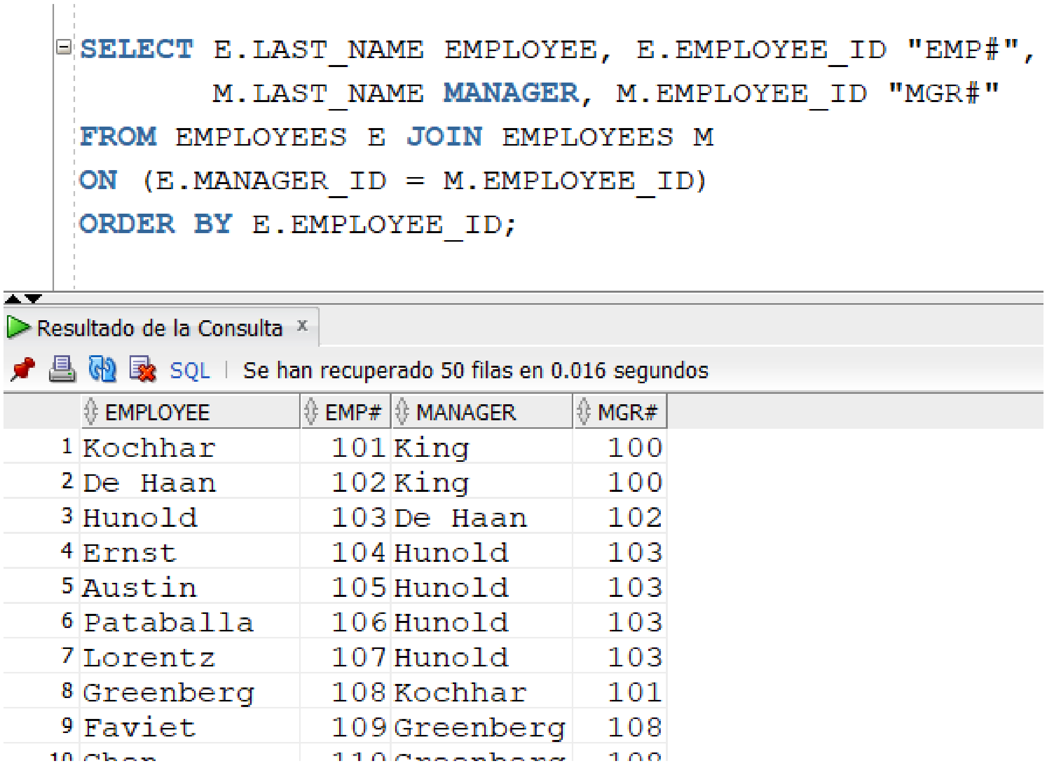
```
SELECT E.LAST_NAME, E.JOB_ID, E.DEPARTMENT_ID, D.DEPARTMENT_NAME
FROM EMPLOYEES E JOIN DEPARTMENTS D
ON (E.DEPARTMENT_ID = D.DEPARTMENT_ID)
JOIN LOCATIONS L
ON (D.LOCATION_ID = L.LOCATION_ID)
WHERE LOWER(L.CITY) = 'toronto';
```

Todas las Filas Recuperadas: 2 en 0.011 segundos

	LAST_NAME	JOB_ID	DEPARTMENT_ID	DEPARTMENT_NAME
1	Hartstein	MK_MAN	20	Marketing
2	Fay	MK_REP	20	Marketing

Figure 21: *Data query with multiple tables.*

4. Create a report to display employees' last name and employee number along with their manager's last name and manager number. Label the columns Employee, Emp#, Manager, and Mgr#, respectively. Save your SQL statement as lab_12_04.sql. Run the query, **see figure 22**.



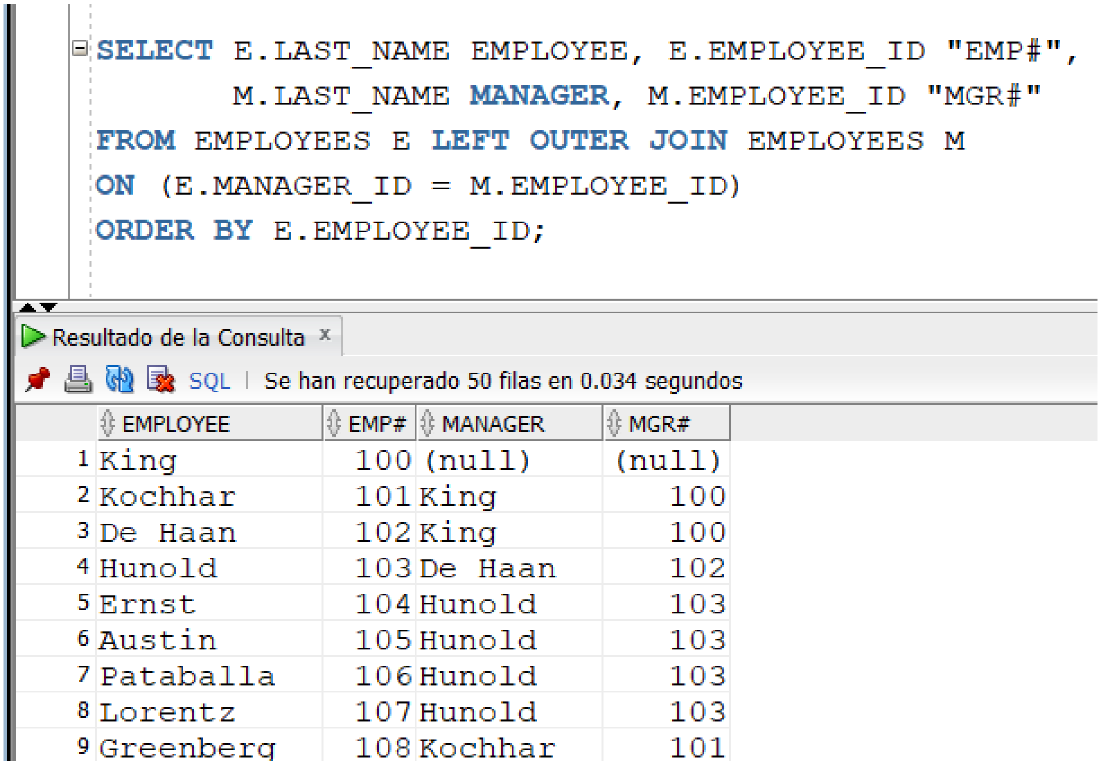
```
SELECT E.LAST_NAME EMPLOYEE, E.EMPLOYEE_ID "EMP#",
       M.LAST_NAME MANAGER, M.EMPLOYEE_ID "MGR#"
FROM EMPLOYEES E JOIN EMPLOYEES M
ON (E.MANAGER_ID = M.EMPLOYEE_ID)
ORDER BY E.EMPLOYEE_ID;
```

Se han recuperado 50 filas en 0.016 segundos

	EMPLOYEE	EMP#	MANAGER	MGR#
1	Kochhar	101	King	100
2	De Haan	102	King	100
3	Hunold	103	De Haan	102
4	Ernst	104	Hunold	103
5	Austin	105	Hunold	103
6	Pataballa	106	Hunold	103
7	Lorentz	107	Hunold	103
8	Greenberg	108	Kochhar	101
9	Faviet	109	Greenberg	108
10	Chen	110	Greenberg	108

Figure 22: *Data query with multiple tables.*

5. Modify lab_12_04.sql to display all employees including King, who has no manager. Order the results by the employee number. Save your SQL statement as lab_12_05.sql. Run the query in lab_12_05.sql, **see figure 23**.



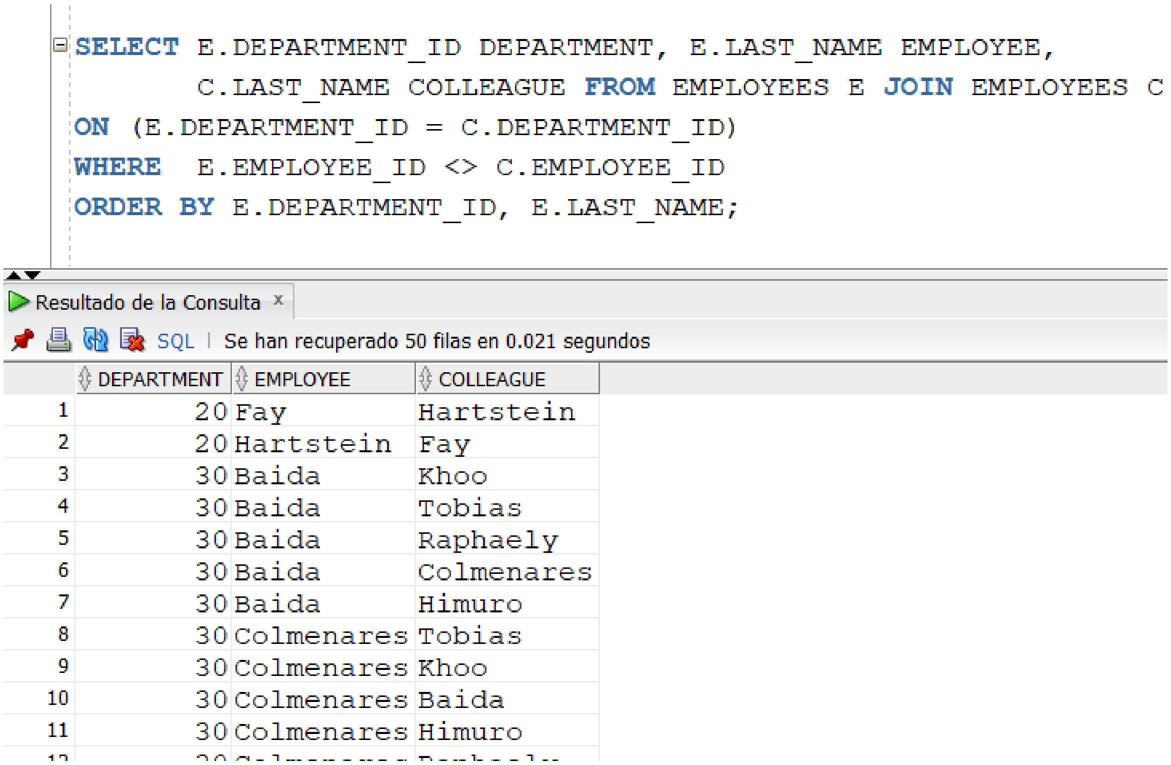
```
SELECT E.LAST_NAME EMPLOYEE, E.EMPLOYEE_ID "EMP#",
       M.LAST_NAME MANAGER, M.EMPLOYEE_ID "MGR#"
FROM EMPLOYEES E LEFT OUTER JOIN EMPLOYEES M
ON (E.MANAGER_ID = M.EMPLOYEE_ID)
ORDER BY E.EMPLOYEE_ID;
```

Se han recuperado 50 filas en 0.034 segundos

	EMPLOYEE	EMP#	MANAGER	MGR#
1	King	100	(null)	(null)
2	Kochhar	101	King	100
3	De Haan	102	King	100
4	Hunold	103	De Haan	102
5	Ernst	104	Hunold	103
6	Austin	105	Hunold	103
7	Pataballa	106	Hunold	103
8	Lorentz	107	Hunold	103
9	Greenberg	108	Kochhar	101

Figure 23: *Data query with multiple tables.*

6. Create a report for the HR department that displays employee last names, department numbers, and all the employees who work in the same department as a given employee. Give each column an appropriate label. Save the script to a file named lab_12_06.sql, **see figure 24**.



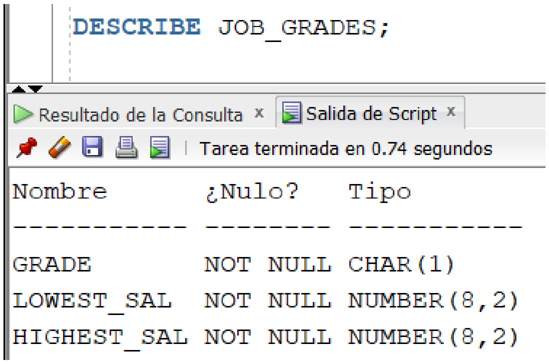
```
SELECT E.DEPARTMENT_ID DEPARTMENT, E.LAST_NAME EMPLOYEE,
       C.LAST_NAME COLLEAGUE FROM EMPLOYEES E JOIN EMPLOYEES C
ON (E.DEPARTMENT_ID = C.DEPARTMENT_ID)
WHERE  E.EMPLOYEE_ID <> C.EMPLOYEE_ID
ORDER BY E.DEPARTMENT_ID, E.LAST_NAME;
```

Se han recuperado 50 filas en 0.021 segundos

	DEPARTMENT	EMPLOYEE	COLLEAGUE
1	20	Fay	Hartstein
2	20	Hartstein	Fay
3	30	Baida	Khoo
4	30	Baida	Tobias
5	30	Baida	Raphaely
6	30	Baida	Colmenares
7	30	Baida	Himuro
8	30	Colmenares	Tobias
9	30	Colmenares	Khoo
10	30	Colmenares	Baida
11	30	Colmenares	Himuro

Figure 24: *Data query with multiple tables.*

7. The HR department needs a report on job grades and salaries. To familiarize yourself with the JOB_GRADES table, first show the structure of the JOB_GRADEStable. Then create a query that displays the name, job, department name, salary, and grade for all employees, **see figure 25 and 26**.

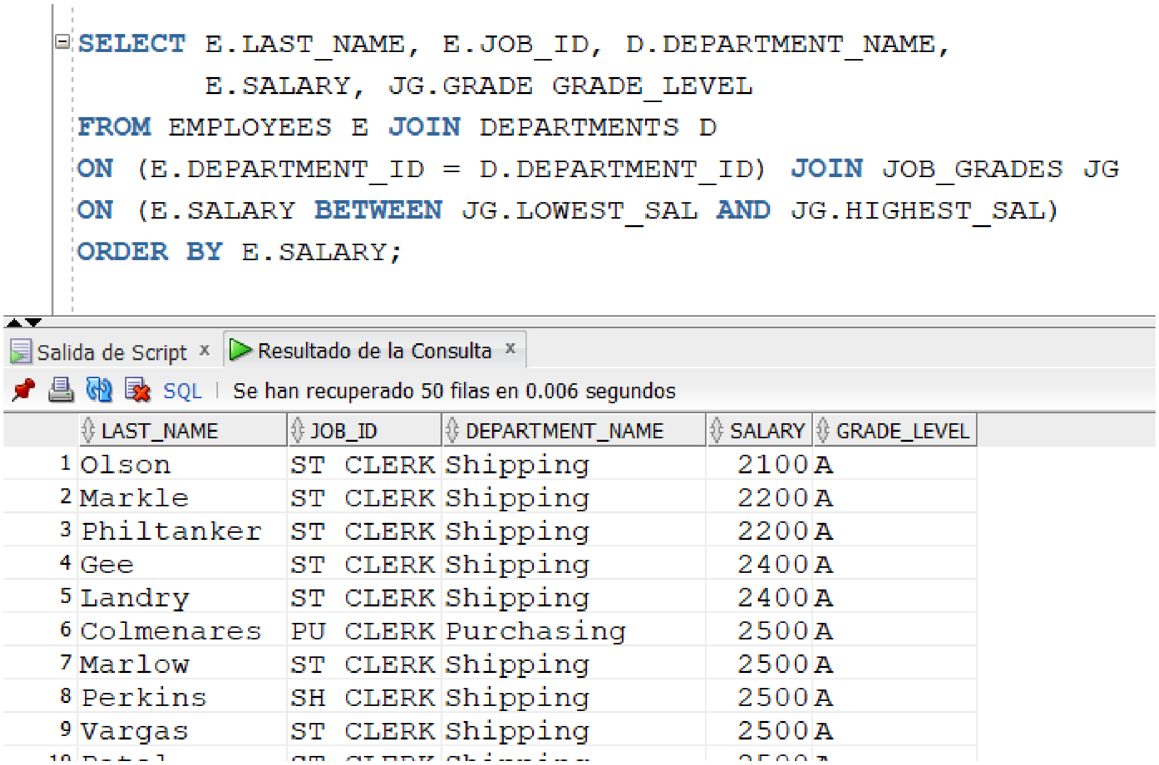


```
DESCRIBE JOB_GRADES;
```

Tarea terminada en 0.74 segundos

```
Nombre      ¿Nulo?   Tipo
----------- -------- -----------
GRADE       NOT NULL CHAR(1)
LOWEST_SAL  NOT NULL NUMBER(8,2)
HIGHEST_SAL NOT NULL NUMBER(8,2)
```

Figure 25: *Structure of the JOB_GRADES table.*



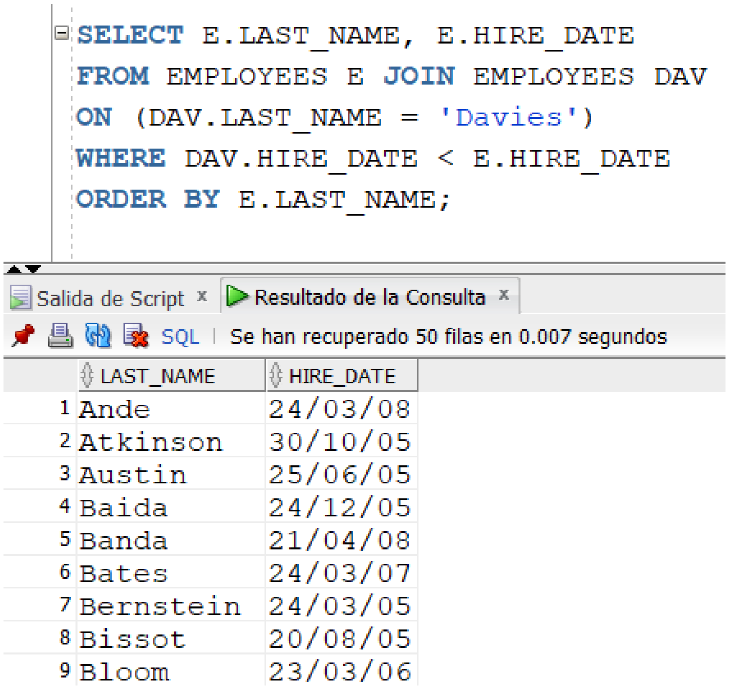
```
SELECT E.LAST_NAME, E.JOB_ID, D.DEPARTMENT_NAME,
       E.SALARY, JG.GRADE GRADE_LEVEL
FROM EMPLOYEES E JOIN DEPARTMENTS D
ON (E.DEPARTMENT_ID = D.DEPARTMENT_ID) JOIN JOB_GRADES JG
ON (E.SALARY BETWEEN JG.LOWEST_SAL AND JG.HIGHEST_SAL)
ORDER BY E.SALARY;
```

Se han recuperado 50 filas en 0.006 segundos

	LAST_NAME	JOB_ID	DEPARTMENT_NAME	SALARY	GRADE_LEVEL
1	Olson	ST CLERK	Shipping	2100	A
2	Markle	ST CLERK	Shipping	2200	A
3	Philtanker	ST CLERK	Shipping	2200	A
4	Gee	ST CLERK	Shipping	2400	A
5	Landry	ST CLERK	Shipping	2400	A
6	Colmenares	PU CLERK	Purchasing	2500	A
7	Marlow	ST CLERK	Shipping	2500	A
8	Perkins	SH CLERK	Shipping	2500	A
9	Vargas	ST CLERK	Shipping	2500	A
10	Patel	ST CLERK	Shipping	2500	A

Figure 26: *Data query with multiple tables.*

8. The HR department wants to determine the names of all the employees who were hired after Davies. Create a query to display the name and hire date of any employee hired after employee Davies, **see figure 27**.



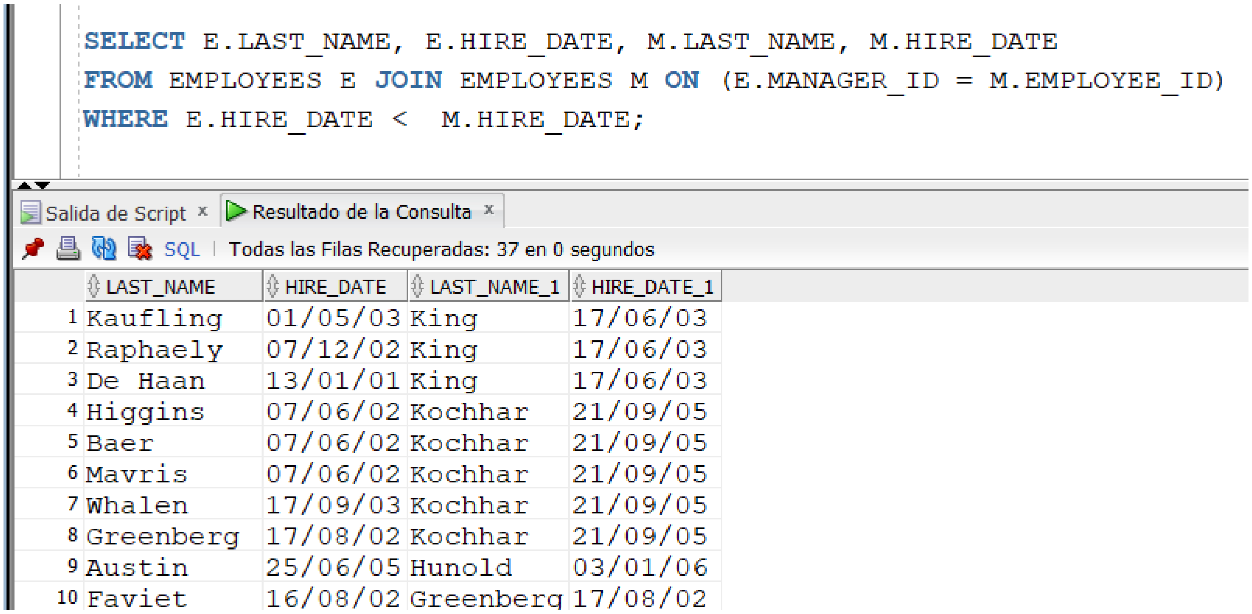
```
SELECT E.LAST_NAME, E.HIRE_DATE
FROM EMPLOYEES E JOIN EMPLOYEES DAV
ON (DAV.LAST_NAME = 'Davies')
WHERE DAV.HIRE_DATE < E.HIRE_DATE
ORDER BY E.LAST_NAME;
```

Se han recuperado 50 filas en 0.007 segundos

	LAST_NAME	HIRE_DATE
1	Ande	24/03/08
2	Atkinson	30/10/05
3	Austin	25/06/05
4	Baida	24/12/05
5	Banda	21/04/08
6	Bates	24/03/07
7	Bernstein	24/03/05
8	Bissot	20/08/05
9	Bloom	23/03/06

Figure 27: *Data query with multiple tables.*

9. The HR department needs to find the names and hire dates of all the employees who were hired before their managers, along with their managers' names and hire dates. Save the script to a file named lab_12_09.sql, **see figure 28**.



```
SELECT E.LAST_NAME, E.HIRE_DATE, M.LAST_NAME, M.HIRE_DATE
FROM EMPLOYEES E JOIN EMPLOYEES M ON (E.MANAGER_ID = M.EMPLOYEE_ID)
WHERE E.HIRE_DATE <  M.HIRE_DATE;
```

Todas las Filas Recuperadas: 37 en 0 segundos

	LAST_NAME	HIRE_DATE	LAST_NAME_1	HIRE_DATE_1
1	Kaufling	01/05/03	King	17/06/03
2	Raphaely	07/12/02	King	17/06/03
3	De Haan	13/01/01	King	17/06/03
4	Higgins	07/06/02	Kochhar	21/09/05
5	Baer	07/06/02	Kochhar	21/09/05
6	Mavris	07/06/02	Kochhar	21/09/05
7	Whalen	17/09/03	Kochhar	21/09/05
8	Greenberg	17/08/02	Kochhar	21/09/05
9	Austin	25/06/05	Hunold	03/01/06
10	Faviet	16/08/02	Greenberg	17/08/02

Figure 28: *Data query with multiple tables.*

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 projection of the data within a relational database is stored in the table in the form of rows and columns. Projections are the first items identified during query execution. They are the selected columns within a table for which a query has been designed. Projections are mentioned in the first part of the SQL query, that is, the SELECT statement. After identifying the projections within the query frame, the next step would be to identify the rows that are relevant to the query. Filters are mentioned within the WHERE clause of the query and will identify the rows to be included in the results, the latter is called a selection.

This practice number 12 helped me practice the uses of the SELECT statement for data retrieval and projection. Finally, something important to mention is that the SQL language allows the projection and selection of data to meet the reporting needs that a programmer, developer or end user may need.