Practice 14: Using Set Operators to Solve Problems

Jesús Manuel Juárez Pasillas

November 18, 2021

1 Introduction

The SET operators the results of two queries resulting in a single result. There are four SET operators which all have the same priority, and each one does something different from the rest.
These operators are used to carry out compound queries, with which to obtain results from two or more queries which are joined with one or more SET operators, where the final result will be what all the operators put back.

2 Development

2.1 Activity 1:

Read all the choices carefully because there might be more than one correct answer.

1. Which of these set operators will not sort the rows?

   A. INTERSECT

   B. MINUS

   C. UNION

   D. **UNION ALL**

   The UNION ALL operator does not order the result by default, but requires placing the ORDER BY clause at the end of the statement.

2. Which of these operators will remove duplicate rows from the final result?

   A. **INTERSECT**

   B. **MINUS**

   C. **UNION**

   D. UNION ALL

   Only the UNION ALL operator returns all rows regardless of repeating rows.

3. If a compound query contains both a MINUS and an INTERSECT operator, which will be applied first?

   A. The INTERSECT, because INTERSECT has higher precedence than MINUS.

   B. The MINUS, because MINUS has a higher precedence than INTERSECT.

   C. **The precedence is determined by the order in which they are specified.**

   D. It is not possible for a compound query to include both MINUS and INTERSECT.

   The Oracle server evaluates the operators from left to right if there are no parentheses indicating the priority.

4. There are four rows in the REGIONS table. Consider the following statements and choose how many rows will be returned for each: 0, 4, 8, or 16.

   A. select * from regions union select * from regions
   Returns 4 rows because it does not return duplicate rows.

   B. select * from regions union all select * from regions 8
   It returns 8 rows because if it returns duplicate rows.

   C. select * from regions minus select * from regions 0
   You don't get any columns because all the rows from the first query are in the second query.

   D. select * from regions intersect select * from regions 4
   Returns 4 rows because the rows returned by the first query are also returned by the second query.

5. Consider this compound query:
   select empno, hired from emp
   union all
   select emp_id,hired,fired from ex_emp;

   A. Because the columns EMPNO and EMP_ID have different names

   B. Because the columns EMP.HIRED and EX_EMP.HIRED are different data types

   C. **Because there are two columns in the first query and three columns in the second query**

   D. For all the reasons above

E. The query will succeed.

It gives an error because you have to have the same number of columns in the first query as in the second, in addition to the fact that the data types must also match in order and number.

6. Which line of this statement will cause it to fail?

   A. select ename, hired from current_staff

   B. **order by ename**

   C. minus

   D. select ename, hired from current staff

   E. where deptno=10

   F. order by ename;

   You can only have one ORDER BY at the end of the statement.

7. Study this statement:
   select ename from emp
   union all
   select ename from ex_emp;
   In what order will the rows be returned?

   A. The rows from each table will be grouped and within each group will be sorted on ENAME.

   B. The rows from each table will be grouped but not sorted.

   C. The rows will not be grouped but will all be sorted on ENAME.

   D. **The rows will be neither grouped nor sorted.**

   The UNION ALL operator does not order the result by default, and it also does not group the results of each query.

2.2 Activity 2:

Propose an answer to the following issues:

- How can you present several tables with similar data as one table?
  - **Using the UNION or UNION ALL operator depending on whether you want to obtain repeated rows or not. You only have to consult the similar columns of each table (obtaining the same number of columns in the same order by data type), and with this obtain the result as if it were only a table.**

- Are there performance issues with compound queries?
  - **Compound queries consist of at least two queries to perform the desired operation and depending on the number of unions required, the number of comparisons that will be made with the results of the queries will be greater or less depending on the union operator.**

2.3 Activity 3:

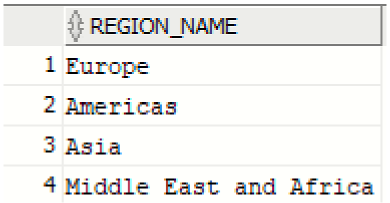
This exercise must be performed using HR schema.

a) In this exercise, you will see the effect of the set operators.

1 Connect to your database as user HR.

2 Run a query that consult the regions table (region_name):

- **Select region_name from Regions;**

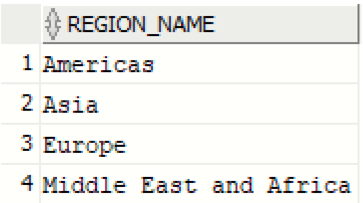


	REGION_NAME
1	Europe
2	Americas
3	Asia
4	Middle East and Africa

Figure 1: Regions table.

3 Query the Regions table twice, using UNION:

- **Select region_name from Regions
UNION
Select region_name from Regions;**

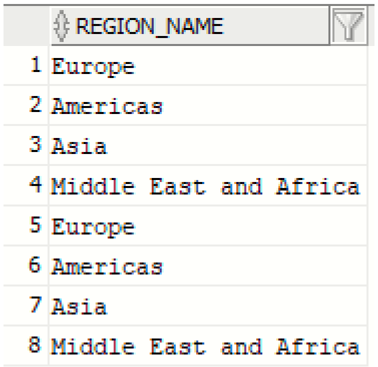


	REGION_NAME
1	Americas
2	Asia
3	Europe
4	Middle East and Africa

Figure 2: Regions table twice, using UNION.

4 This time, use UNION ALL:

- **Select region_name from Regions
UNION ALL
Select region_name from Regions;**

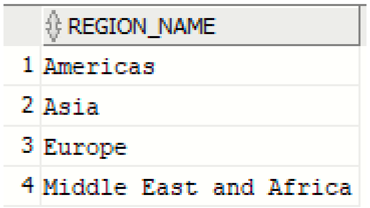


	REGION_NAME
1	Europe
2	Americas
3	Asia
4	Middle East and Africa
5	Europe
6	Americas
7	Asia
8	Middle East and Africa

Figure 3: Regions table twice, using UNION ALL.

5 An intersection will retrieve rows common to two queries:

- **Select region_name from Regions**
  INTERSECT
  Select region_name from Regions;

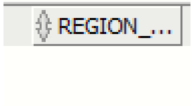


	REGION_NAME
1	Americas
2	Asia
3	Europe
4	Middle East and Africa

Figure 4: Regions table twice, using INTERSECT.

6 A MINUS will remove common rows:

- **Select region_name from Regions**
  MINUS
  Select region_name from Regions;



	REGION_...

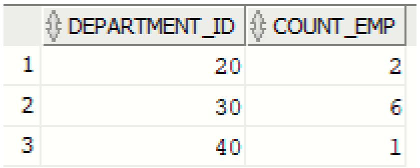
Figure 5: Regions table twice, using MINUS.

b) In this exercise, you will run more complex compound queries.

1 Connect to your database as user HR.

2 Run a simple query to count the employees in three departments (20,30,40), grouped by them:

- **Select department_id, count(employee_id) count_emp from Employees where department_id in (20,30,40) group by department_id;**



	DEPARTMENT_ID	COUNT_EMP
1	20	2
2	30	6
3	40	1

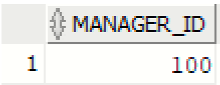
Figure 6: Count the employees in three departments (20,30,40).

3 Obtain the same result with a compound query:

- **Select department_id, count(employee_id)count_emp from Employees group by department_id**
  INTERSECT
  Select department_id, count(*) from Employees where department_id in (20,30,40) group by department_id;

4 Find out (using compound queries) if any managers manage staff in both departments 20 and 30, and exclude any managers with staff in department 40:

- **Select manager_id from Employees**
  INTERSECT
  Select manager_id from Employees where department_id = 20
  INTERSECT
  Select manager_id from Employees where department_id = 30
  INTERSECT
  Select employee_id from Employees where department_id != 40;

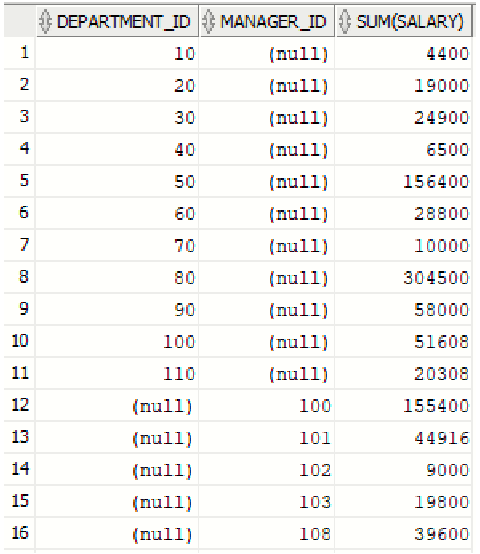


	MANAGER_ID
1	100

Figure 7: Managers staff in both departments 20 and 30

5 Use a compound query (3 sentences using two set operator) to report salaries (from employees) subtotaled by department (grouped by department_id), by manager (grouped by manager_id), and the overall total. Order the query:

-
  ```
  Select department_id,null manager_id,sum(salary)
  from Employees group by department_id
  UNION
  Select null,manager_id,sum(salary) from Employees
  group by manager_id
  UNION ALL
  Select null,null,sum(salary) from Employees;
  ```



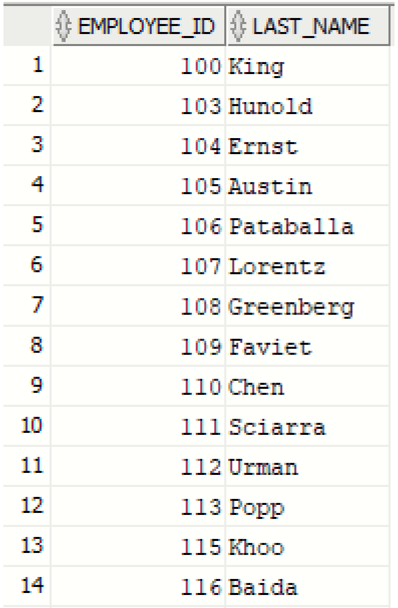
	DEPARTMENT_ID	MANAGER_ID	SUM(SALARY)
1	10	(null)	4400
2	20	(null)	19000
3	30	(null)	24900
4	40	(null)	6500
5	50	(null)	156400
6	60	(null)	28800
7	70	(null)	10000
8	80	(null)	304500
9	90	(null)	58000
10	100	(null)	51608
11	110	(null)	20308
12	(null)	100	155400
13	(null)	101	44916
14	(null)	102	9000
15	(null)	103	19800
16	(null)	108	39600

Figure 8: Salaries subtotaled by department, by manager, and the overall total.

c) Working in the HR schema, design some queries that will generate reports using the set operators. The reports required are as follows:

1 Employees have their current job (identified by JOB_ID) recorded in their EMPLOYEES row. Jobs they have held previously (but not their current job) are recorded in JOB_HISTORY. Which employees have never changed jobs? The listing should include the employees' EMPLOYEE_ID and LAST_NAME.

-
  ```
  Select employee_id, last_name from Employees
  where employee_id in
  (Select employee_id from Employees
  MINUS
  Select employee_id from Job_History) order by employee_id;
  ```

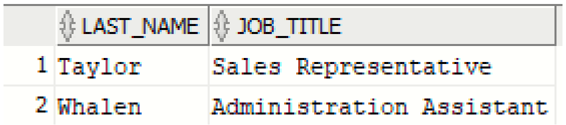


	EMPLOYEE_ID	LAST_NAME
1	100	King
2	103	Hunold
3	104	Ernst
4	105	Austin
5	106	Pataballa
6	107	Lorentz
7	108	Greenberg
8	109	Faviet
9	110	Chen
10	111	Sciarra
11	112	Urman
12	113	Popp
13	115	Khoo
14	116	Baida

Figure 9: Employees who have never changed jobs

2 Which employees were recruited into one job, then changed to a different job, but are now back in a job they held before? Again, you will need to construct a query that compares EMPLOYEES with JOB_HISTORY. The report should show the employees' names and the job titles. Job titles are stored in the table JOBS.

- **Select last_name, job_title from Employees**
  natural join Jobs natural join
  (Select employee_id,job_id from Job_History
  INTERSECT
  Select employee_id,job_id from Employees);

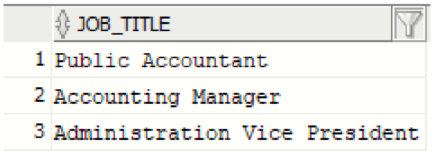


	LAST_NAME	JOB_TITLE
1	Taylor	Sales Representative
2	Whalen	Administration Assistant

Figure 10: Which employees back in a job they held before

3 What jobs has any one employee held? This will be the JOB_ID for the employee's current job (in EMPLOYEES) and all previous jobs (in JOB_HISTORY). If the employee has held a job more than once, there is no need to list it more than once. Use a replacement variable to prompt for the EMPLOYEE_ID and display the job title(s). Employees 101 and 200 will be suitable employees for testing.

-
  ```
  Select job_title from Jobs natural join
  (Select job_id from Employees where
  employee_id = &&employee_id
  UNION
  Select job_id from Job_History where
  employee_id = &employee_id)
  ```



	JOB_TITLE
1	Public Accountant
2	Accounting Manager
3	Administration Vice President

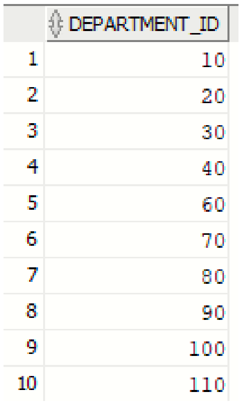
Figure 11: Employee's current job and all previous jobs

2.4 Activity 4:

In this activity you will write several queries using the set operators.

1. The HR department needs a list of department IDs for departments that do not contain the job ID ST_CLERK.

   -
     ```
     Select department_id from Employees where
     department_id is not null
     MINUS
     (Select department_id from Employees
     where job_id = 'ST_CLERK'
     UNION
     Select department_id from Job_History
     where job_id = 'ST_CLERK');
     ```

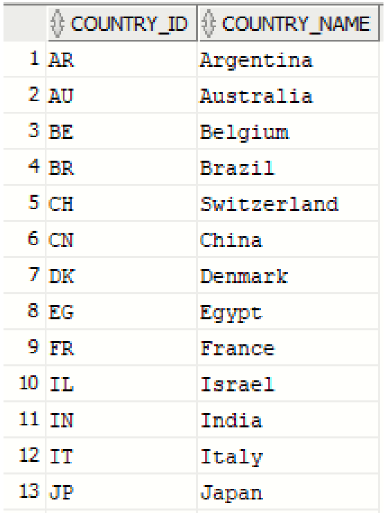


	DEPARTMENT_ID
1	10
2	20
3	30
4	40
5	60
6	70
7	80
8	90
9	100
10	110

Figure 12: Departments that do not contain the job ID ST_CLERK.

2. The HR department needs a list of countries that have no departments located in them. Display the country ID and the name of the countries.

   - **Select country_id, country_name from Countries**
MINUS
Select country_id, country_name from Countries join Locations using (country_id) join departments using (location_id);

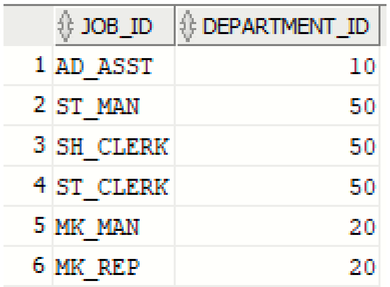


	COUNTRY_ID	COUNTRY_NAME
1	AR	Argentina
2	AU	Australia
3	BE	Belgium
4	BR	Brazil
5	CH	Switzerland
6	CN	China
7	DK	Denmark
8	EG	Egypt
9	FR	France
10	IL	Israel
11	IN	India
12	IT	Italy
13	JP	Japan

Figure 13: Countries that have no departments located in them.

3. Produce a list of jobs for departments 10, 50, and 20, in that order. Display the job ID and department ID by using the set operators.

   - **Select distinct job_id, department_id from Employees where (department_id=10)**
UNION ALL
Select distinct job_id, department_id from Employees where (department_id=50)
UNION ALL
Select distinct job_id, department_id from Employees where (department_id=20);

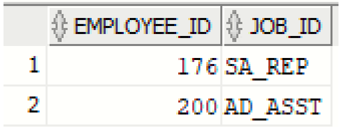


	JOB_ID	DEPARTMENT_ID
1	AD_ASST	10
2	ST_MAN	50
3	SH_CLERK	50
4	ST_CLERK	50
5	MK_MAN	20
6	MK_REP	20

Figure 14: Jobs for departments 10, 50, and 20.

4. Create a report that lists the employee IDs and job IDs of those employees who currently have a job title that is the same as their job title when they were initially hired by the company (that is, they changed jobs but have now gone back to doing their original job).

   - **Select employee_id, job_id from Employees**
     INTERSECT
     Select employee_id, job_id from Job_History;



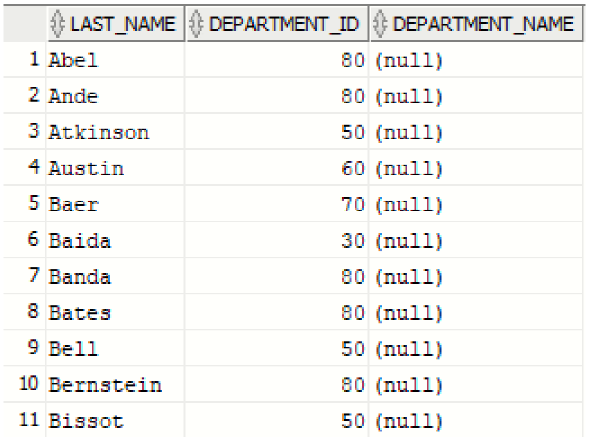
	EMPLOYEE_ID	JOB_ID
1	176	SA_REP
2	200	AD_ASST

Figure 15: Employees they changed jobs but have gone back to their original job.

5. The HR department needs a report with the following specifications:

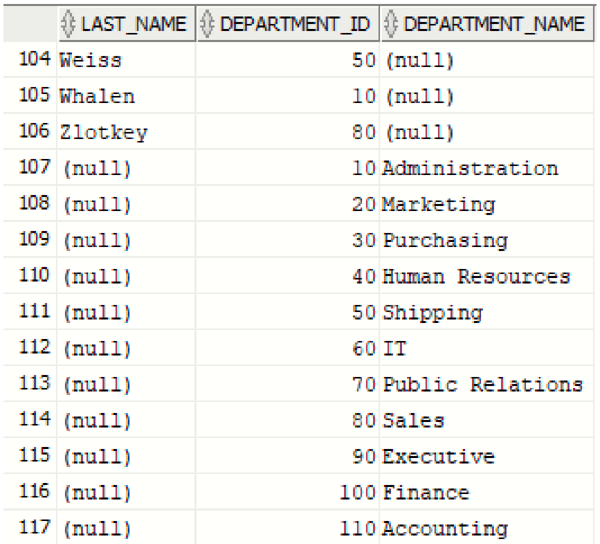
   - Last name and department ID of all employees from the EMPLOYEES table, regardless of whether or not they belong to a department
   - Department ID and department name of all departments from the DEPARTMENTS table, regardless of whether or not they have employees working in them

   - **Select last_name,department_id,to_char(null)**
     Department_name from Employees
     UNION
     Select to_char(null),department_id,department_name
     from Departments;



	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
1	Abel	80	(null)
2	Ande	80	(null)
3	Atkinson	50	(null)
4	Austin	60	(null)
5	Baer	70	(null)
6	Baida	30	(null)
7	Banda	80	(null)
8	Bates	80	(null)
9	Bell	50	(null)
10	Bernstein	80	(null)
11	Bissot	50	(null)

Figure 16: EMPLOYEES.



	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
104	Weiss	50	(null)
105	Whalen	10	(null)
106	Zlotkey	80	(null)
107	(null)	10	Administration
108	(null)	20	Marketing
109	(null)	30	Purchasing
110	(null)	40	Human Resources
111	(null)	50	Shipping
112	(null)	60	IT
113	(null)	70	Public Relations
114	(null)	80	Sales
115	(null)	90	Executive
116	(null)	100	Finance
117	(null)	110	Accounting

Figure 17: DEPARTMENTS.

3 Pre-Assessment:

- Practices pre-Assessment for Database Systems Laboratory II

Practice	Pre-Assessment
COMPLIES WITH THE REQUESTED FUNCTIONALITY	X
HAS THE CORRECT INDENTATION	X
HAS AN EASY WAY TO ACCESS THE PROVIDED FILES	X
HAS A REPORT WITH IDC FORMAT	X
REPORT INFORMATION IS FREE OF SPELLING ERRORS	X
DELIVERED IN TIME AND FORM	X
IS FULLY COMPLETED (SPECIFY THE PERCENTAGE COMPLETED)	100%

Table 1: Pre-Assessment.

4 Conclusion:

The SET operators are very important when trying to compare the results of two or more queries, in which each operator will tell us something different, therefore, it is necessary to use the one that best suits depending on what is required at the time.

It is very important to control the execution of each of these operators since, although they all have the same priority, the sentence will be executed from left to right if there are no parentheses that specify the order.