9

Using the Set Operators

Objectives

After completing this lesson, you should be able to do the following:

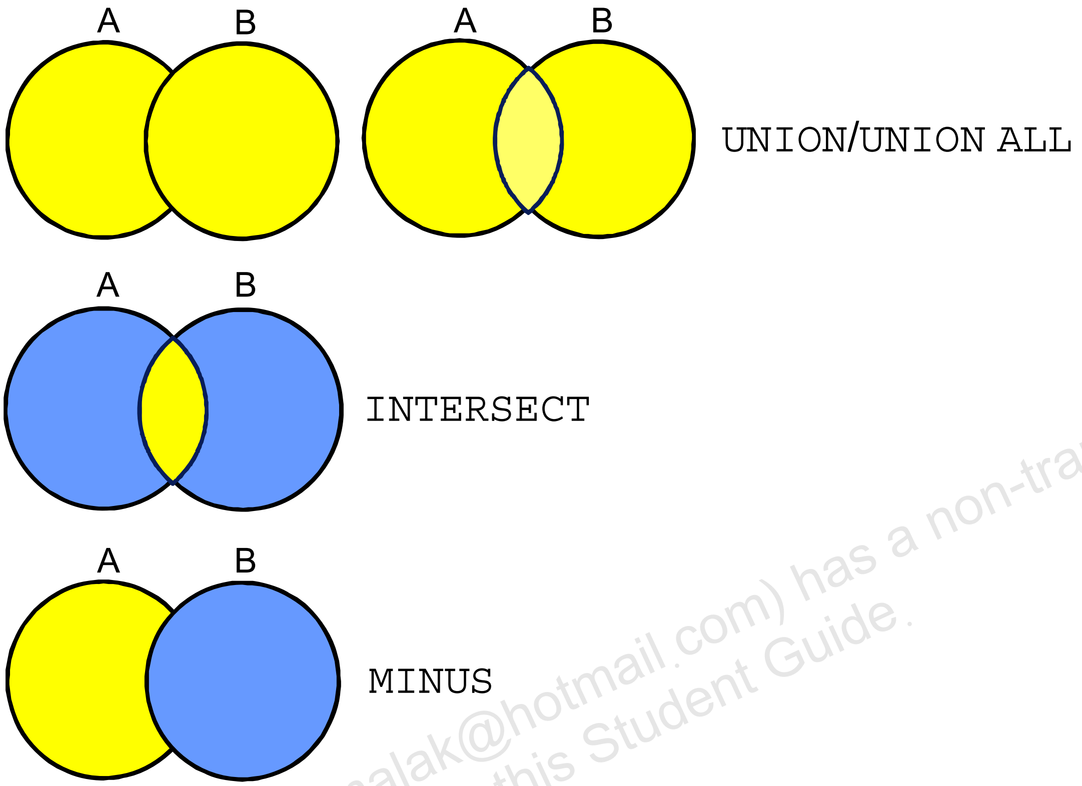
- Describe set operators
- Use a set operator to combine multiple queries into a single query
- Control the order of rows returned

In this lesson, you learn how to write queries by using set operators.

Lesson Agenda

- Set operators: Types and guidelines
- Tables used in this lesson
- `UNION` and `UNION ALL` operator
- `INTERSECT` operator
- `MINUS` operator
- Matching `SELECT` statements
- Using the `ORDER BY` clause in set operations

Set Operators



Set operators combine the results of two or more component queries into one result. Queries containing set operators are called *compound queries*.

Operator	**Returns**
UNION	Rows from both queries after eliminating duplications
UNION ALL	Rows from both queries, including all duplications
INTERSECT	Rows that are common to both queries
MINUS	Rows in the first query that are not present in the second query

All set operators have equal precedence. If a SQL statement contains multiple set operators, the Oracle server evaluates them from left (top) to right (bottom), if no parentheses explicitly specify another order. You should use parentheses to specify the order of evaluation explicitly in queries that use the INTERSECT operator with other set operators.

Set Operator Rules

- The expressions in the SELECT lists must match in number.
- The data type of each column in the subsequent query must match the data type of its corresponding column in the first query.
- Parentheses can be used to alter the sequence of execution.
- ORDER BY clause can appear only at the very end of the statement.

- The expressions in the SELECT lists of the queries must match in number and data type. Queries that use UNION, UNION ALL, INTERSECT, and MINUS operators must have the same number and data type of columns in their SELECT list. The data type of the columns in the SELECT list of the queries in the compound query may not be exactly the same. The column in the second query must be in the same data type group (such as numeric or character) as the corresponding column in the first query.
- Set operators can be used in subqueries.
- You should use parentheses to specify the order of evaluation in queries that use the INTERSECT operator with other set operators. This ensures compliance with emerging SQL standards that will give the INTERSECT operator greater precedence than the other set operators.

Oracle Server and Set Operators

- Duplicate rows are automatically eliminated except in `UNION ALL`.
- Column names from the first query appear in the result.
- The output is sorted in ascending order by default except in `UNION ALL`.

When a query uses set operators, the Oracle server eliminates duplicate rows automatically except in the case of the `UNION ALL` operator. The column names in the output are decided by the column list in the first `SELECT` statement. By default, the output is sorted in ascending order of the first column of the `SELECT` clause.

The corresponding expressions in the `SELECT` lists of the component queries of a compound query must match in number and data type. If component queries select character data, the data type of the return values is determined as follows:

- If both queries select values of `CHAR` data type, of equal length, the returned values have the `CHAR` data type of that length. If the queries select values of `CHAR` with different lengths, the returned value is `VARCHAR2` with the length of the larger `CHAR` value.
- If either or both of the queries select values of `VARCHAR2` data type, the returned values have the `VARCHAR2` data type.

If component queries select numeric data, the data type of the return values is determined by numeric precedence. If all queries select values of the `NUMBER` type, the returned values have the `NUMBER` data type. In queries using set operators, the Oracle server does not perform implicit conversion across data type groups. Therefore, if the corresponding expressions of component queries resolve to both character data and numeric data, the Oracle server returns an error.

Lesson Agenda

- Set operators: Types and guidelines
- Tables used in this lesson
- `UNION` and `UNION ALL` operator
- `INTERSECT` operator
- `MINUS` operator
- Matching `SELECT` statements
- Using the `ORDER BY` clause in set operations

Tables Used in This Lesson

The tables used in this lesson are:

- `EMPLOYEES`: Provides details regarding all current employees
- `RETIRED_EMPLOYEES`: Provides details regarding all past employees

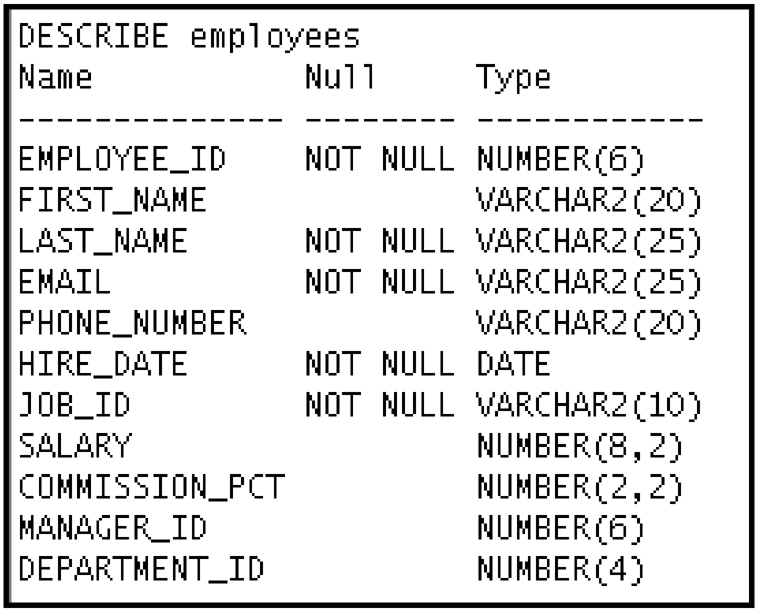
Two tables are used in this lesson: the `EMPLOYEES` table and the `RETIRED_EMPLOYEES` table.

You are already familiar with the `EMPLOYEES` table that stores employee details such as a unique identification number, email address, job identification (such as `ST_CLERK`, `SA_REP`, and so on), salary, manager, and so on.

`RETIRED_EMPLOYEES` stores the details of the employees who have left the company.

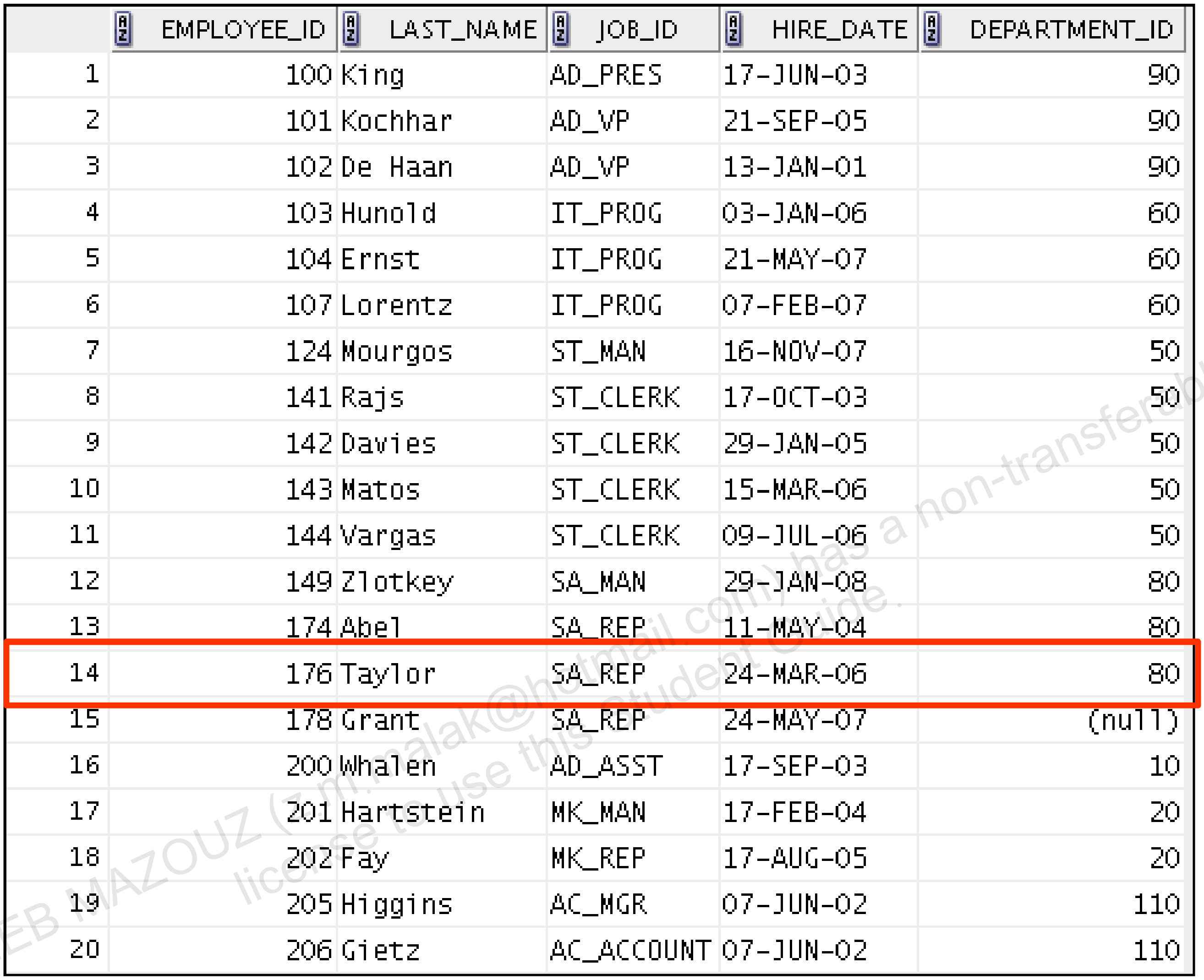
The structure and data from the `EMPLOYEES` and `RETIRED_EMPLOYEES` tables are shown on the following pages.

```
DESCRIBE employees
```



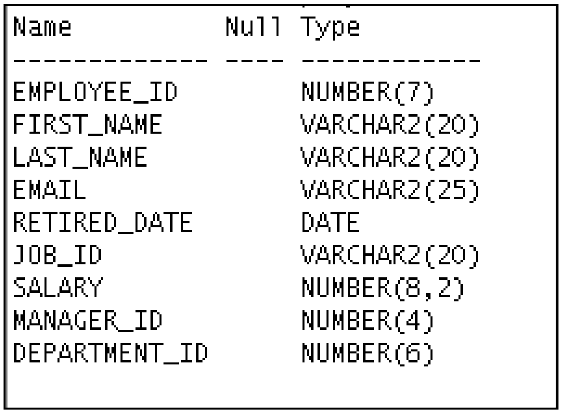
```
DESCRIBE employees
Name            Null     Type
--------------- -------- ------------
EMPLOYEE_ID     NOT NULL NUMBER(6)
FIRST_NAME               VARCHAR2(20)
LAST_NAME       NOT NULL VARCHAR2(25)
EMAIL           NOT NULL VARCHAR2(25)
PHONE_NUMBER             VARCHAR2(20)
HIRE_DATE       NOT NULL DATE
JOB_ID          NOT NULL VARCHAR2(10)
SALARY                   NUMBER(8,2)
COMMISSION_PCT           NUMBER(2,2)
MANAGER_ID               NUMBER(6)
DEPARTMENT_ID            NUMBER(4)
```

```
SELECT employee_id, last_name, job_id, hire_date, department_id
FROM employees;
```



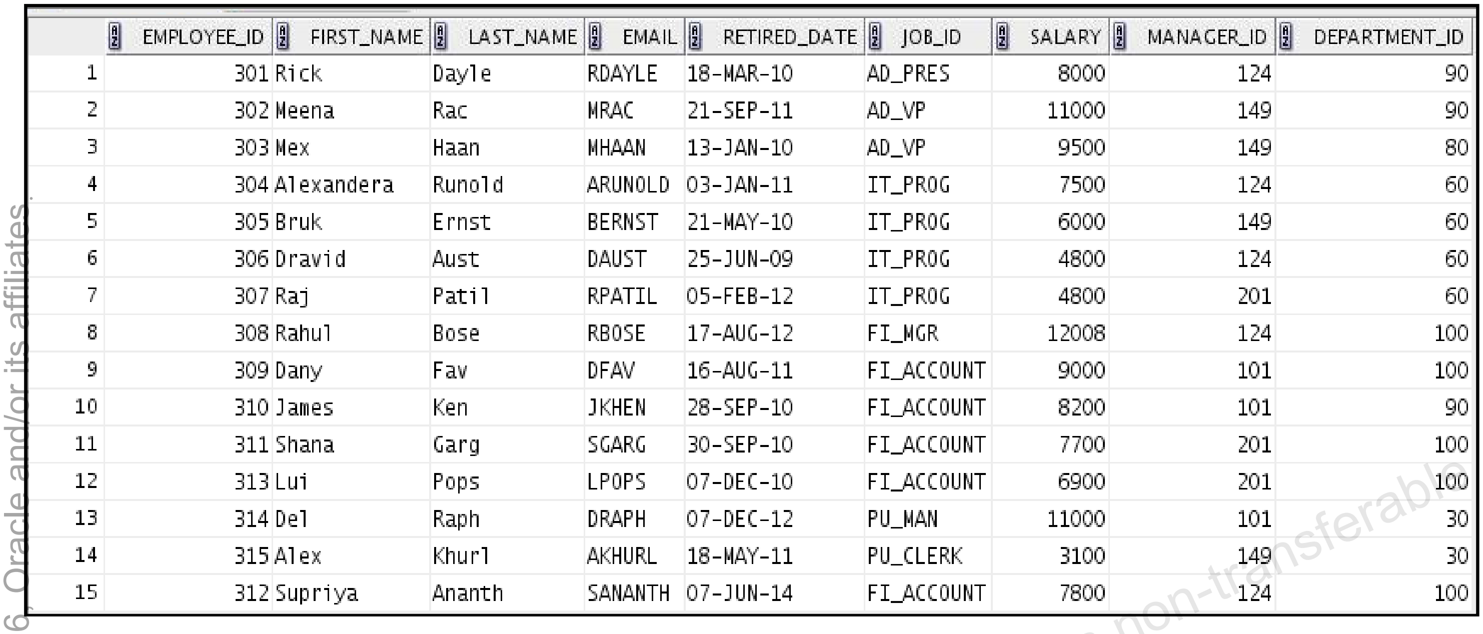
	EMPLOYEE_ID	LAST_NAME	JOB_ID	HIRE_DATE	DEPARTMENT_ID
1	100	King	AD_PRES	17-JUN-03	90
2	101	Kochhar	AD_VP	21-SEP-05	90
3	102	De Haan	AD_VP	13-JAN-01	90
4	103	Hunold	IT_PROG	03-JAN-06	60
5	104	Ernst	IT_PROG	21-MAY-07	60
6	107	Lorentz	IT_PROG	07-FEB-07	60
7	124	Mourgos	ST_MAN	16-NOV-07	50
8	141	Rajs	ST_CLERK	17-OCT-03	50
9	142	Davies	ST_CLERK	29-JAN-05	50
10	143	Matos	ST_CLERK	15-MAR-06	50
11	144	Vargas	ST_CLERK	09-JUL-06	50
12	149	Zlotkey	SA_MAN	29-JAN-08	80
13	174	Abel	SA_REP	11-MAY-04	80
14	176	Taylor	SA_REP	24-MAR-06	80
15	178	Grant	SA_REP	24-MAY-07	(null)
16	200	Whalen	AD_ASST	17-SEP-03	10
17	201	Hartstein	MK_MAN	17-FEB-04	20
18	202	Fay	MK_REP	17-AUG-05	20
19	205	Higgins	AC_MGR	07-JUN-02	110
20	206	Gietz	AC_ACCOUNT	07-JUN-02	110

```
DESCRIBE retired_employees
```



```
Name          Null Type
------------- ---- ------------
EMPLOYEE_ID        NUMBER(7)
FIRST_NAME         VARCHAR2(20)
LAST_NAME          VARCHAR2(20)
EMAIL              VARCHAR2(25)
RETIRED_DATE       DATE
JOB_ID             VARCHAR2(20)
SALARY             NUMBER(8,2)
MANAGER_ID         NUMBER(4)
DEPARTMENT_ID      NUMBER(6)
```

```
SELECT * FROM retired_employees;
```

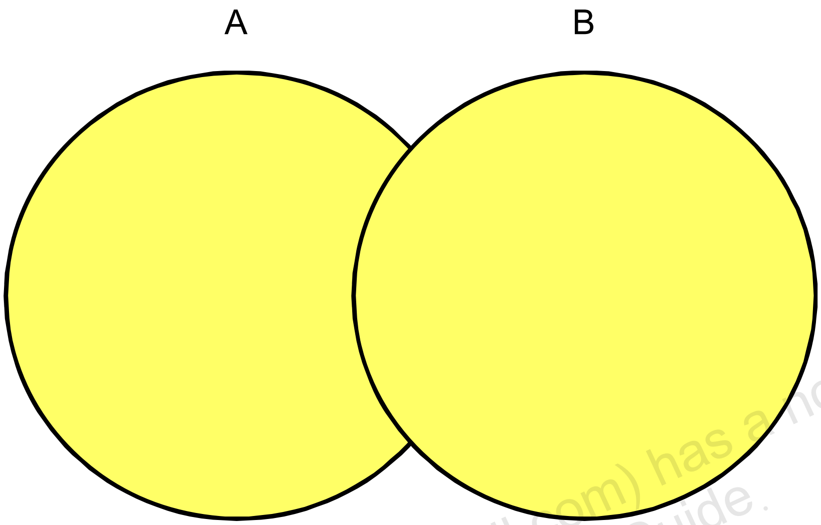


	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	RETIRED_DATE	JOB_ID	SALARY	MANAGER_ID	DEPARTMENT_ID
1	301	Rick	Dayle	RDAYLE	18-MAR-10	AD_PRES	8000	124	90
2	302	Meena	Rac	MRAC	21-SEP-11	AD_VP	11000	149	90
3	303	Mex	Haan	MHAAN	13-JAN-10	AD_VP	9500	149	80
4	304	Alexandera	Runold	ARUNOLD	03-JAN-11	IT_PROG	7500	124	60
5	305	Bruk	Ernst	BERNST	21-MAY-10	IT_PROG	6000	149	60
6	306	Dravid	Aust	DAUST	25-JUN-09	IT_PROG	4800	124	60
7	307	Raj	Patil	RPATIL	05-FEB-12	IT_PROG	4800	201	60
8	308	Rahul	Bose	RBOSE	17-AUG-12	FI_MGR	12008	124	100
9	309	Dany	Fav	DFAV	16-AUG-11	FI_ACCOUNT	9000	101	100
10	310	James	Ken	JKHEN	28-SEP-10	FI_ACCOUNT	8200	101	90
11	311	Shana	Garg	SGARG	30-SEP-10	FI_ACCOUNT	7700	201	100
12	313	Lui	Pops	LPOPS	07-DEC-10	FI_ACCOUNT	6900	201	100
13	314	Del	Raph	DRAPH	07-DEC-12	PU_MAN	11000	101	30
14	315	Alex	Khurl	AKHURL	18-MAY-11	PU_CLERK	3100	149	30
15	312	Supriya	Ananth	SANANTH	07-JUN-14	FI_ACCOUNT	7800	124	100

Lesson Agenda

- Set operators: Types and guidelines
- Tables used in this lesson
- UNION and UNION ALL operator
- INTERSECT operator
- MINUS operator
- Matching SELECT statements
- Using the ORDER BY clause in set operations

UNION Operator



The UNION operator returns rows from both queries after eliminating duplications.

The UNION operator returns all rows that are selected by either query. Use the UNION operator to return all rows from multiple tables and eliminate any duplicate rows.

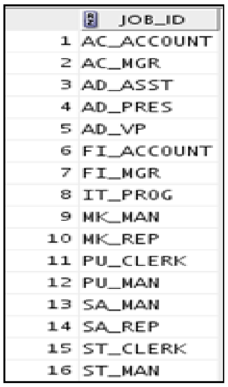
Guidelines

- The number of columns being selected must be the same.
- The data types of the columns being selected must be in the same data type group (such as numeric or character).
- The names of the columns need not be identical.
- UNION operates over all of the columns being selected.
- NULL values are not ignored during duplicate checking.
- By default, the output is sorted in ascending order of the columns of the SELECT clause.

Using the UNION Operator

Display the job details of all the current and retired employees. Display each job only once.

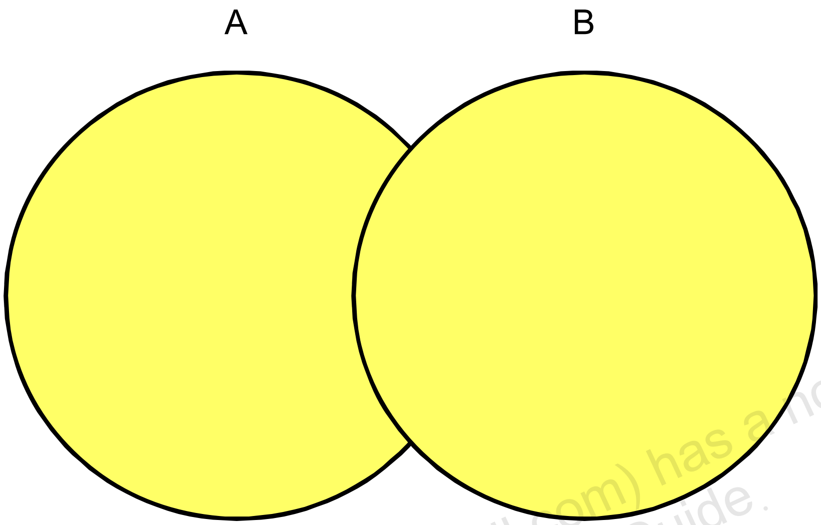
```
SELECT job_id
FROM   employees
UNION
SELECT job_id
FROM   retired_employees
```



	JOB_ID
1	AC_ACCOUNT
2	AC_MGR
3	AD_ASST
4	AD_PRES
5	AD_VP
6	FI_ACCOUNT
7	FI_MGR
8	IT_PROG
9	MK_MAN
10	MK_REP
11	PU_CLERK
12	PU_MAN
13	SA_MAN
14	SA_REP
15	ST_CLERK
16	ST_MAN

The UNION operator eliminates any duplicate records. If records that occur in both the EMPLOYEES and the RETIRED_EMPLOYEES tables are identical, the records are displayed only once.

UNION ALL Operator



The UNION ALL operator returns rows from both queries, including all duplications.

Use the UNION ALL operator to return all rows from multiple queries.

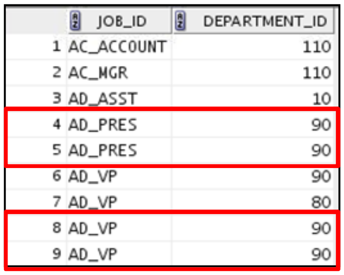
Guidelines

The guidelines for UNION and UNION ALL are the same, with the following two exceptions that pertain to UNION ALL: Unlike UNION, duplicate rows are not eliminated and the output is not sorted by default.

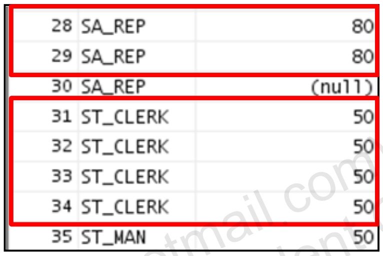
Using the `UNION ALL` Operator

Display the jobs and departments of all current and previous employees.

```
SELECT job_id, department_id
FROM   employees
UNION ALL
SELECT job_id, department_id
FROM   retired_employees
ORDER BY  job_id;
```



	JOB_ID	DEPARTMENT_ID
1	AC_ACCOUNT	110
2	AC_MGR	110
3	AD_ASST	10
4	AD_PRES	90
5	AD_PRES	90
6	AD_VP	90
7	AD_VP	80
8	AD_VP	90
9	AD_VP	90



28	SA_REP	80
29	SA_REP	80
30	SA_REP	(null)
31	ST_CLERK	50
32	ST_CLERK	50
33	ST_CLERK	50
34	ST_CLERK	50
35	ST_MAN	50

…

In the example, 35 rows are selected. The combination of the two tables totals to 35 rows. The `UNION ALL` operator does not eliminate duplicate rows. `UNION` returns all distinct rows selected by either query. `UNION ALL` returns all rows selected by either query, including all duplicates. Consider the query in the slide, now written with the `UNION` clause:

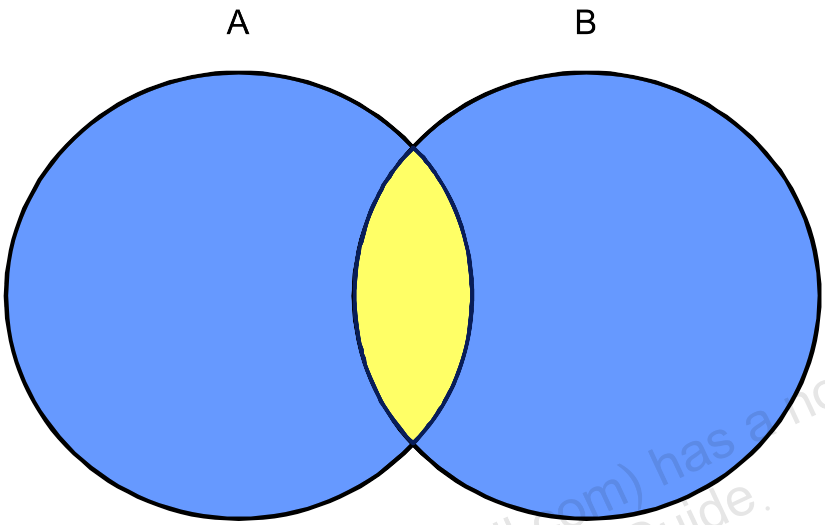
```
SELECT job_id,department_id
FROM     employees
UNION
SELECT job_id,department_id
FROM    retired_employees
ORDER BY job_id;
```

The preceding query returns 19 rows. This is because it eliminates all the duplicate rows.

Lesson Agenda

- Set operators: Types and guidelines
- Tables used in this lesson
- UNION and UNION ALL operator
- INTERSECT operator
- MINUS operator
- Matching SELECT statements
- Using ORDER BY clause in set operations

INTERSECT Operator



The INTERSECT operator returns rows that are common to both queries.

Use the INTERSECT operator to return all rows that are common to multiple queries.

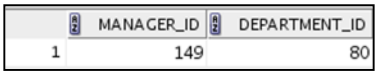
Guidelines

- The number of columns and the data types of the columns being selected by the SELECT statements in the queries must be identical in all the SELECT statements used in the query. The names of the columns, however, need not be identical.
- Reversing the order of the intersected tables does not alter the result.
- INTERSECT does not ignore NULL values.

Using the `INTERSECT` Operator

Display the common manager IDs and department IDs of current and previous employees.

```
SELECT  manager_id,department_id
FROM   employees
INTERSECT
SELECT manager_id,department_id
FROM   retired_employees
```



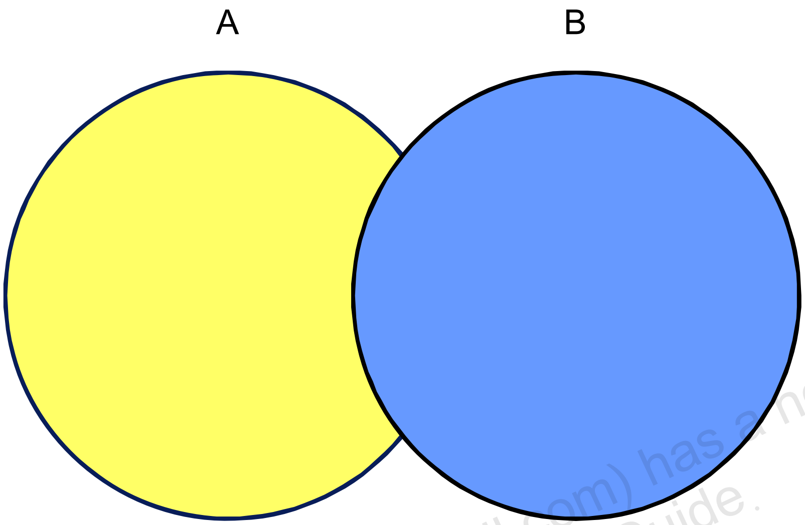
	MANAGER_ID	DEPARTMENT_ID
1	149	80

In the example in this slide, the query returns only those records that have the same values in the selected columns in both tables.

Lesson Agenda

- Set operators: Types and guidelines
- Tables used in this lesson
- UNION and UNION ALL operator
- INTERSECT operator
- MINUS operator
- Matching SELECT statements
- Using the ORDER BY clause in set operations

MINUS Operator



The MINUS operator returns all the distinct rows selected by the first query, but not present in the second query result set.

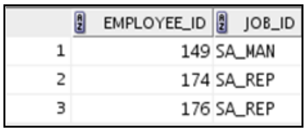
Use the MINUS operator to return all distinct rows selected by the first query, but not present in the second query result set (the first SELECT statement MINUS the second SELECT statement).

Note: The number of columns must be the same and the data types of the columns being selected by the SELECT statements in the queries must belong to the same data type group in all the SELECT statements used in the query. The names of the columns, however, need not be identical.

Using the `MINUS` Operator

Display the employee IDs and Job IDs of those employees who works in the sales department.

```
SELECT employee_id, job_id
FROM employees
WHERE department_id = 80
MINUS
SELECT employee_id, job_id
FROM retired_employees
WHERE department_id = 90;
```



	EMPLOYEE_ID	JOB_ID
1	149	SA_MAN
2	174	SA_REP
3	176	SA_REP

In the example in the slide, the employee IDs in the RETIRED_EMLOYEES table are subtracted from those in the EMPLOYEES table. The results set displays the employees remaining after the subtraction; they are represented by rows that exist in the EMPLOYEES table, but do not exist in the RETIRED_EMPLOYEES table. These are the records of the employees who work in the sales department.

Lesson Agenda

- Set operators: Types and guidelines
- Tables used in this lesson
- `UNION` and `UNION ALL` operator
- `INTERSECT` operator
- `MINUS` operator
- Matching `SELECT` statements
- Using `ORDER BY` clause in set operations

Matching SELECT Statements

You must match the data type (using the TO_CHAR function or any other conversion functions) when columns do not exist in one or the other table.

```
SELECT location_id, department_name "Department",
   TO_CHAR(NULL) "Warehouse location"
FROM departments
UNION
SELECT location_id, TO_CHAR(NULL) "Department",
   state_province
FROM locations;
```

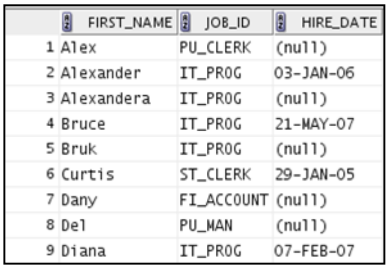
Because the expressions in the SELECT lists of the queries must match in number, you can use the dummy columns and the data type conversion functions to comply with this rule. To match the column list explicitly, you can insert NULL columns at the missing positions so as to match the count and data type of selected columns in each SELECT statement. In the slide, the name Warehouse location is given as the dummy column heading. The TO_CHAR function is used in the first query to match the VARCHAR2 data type of the state_province column that is retrieved by the second query. Similarly, the TO_CHAR function in the second query is used to match the VARCHAR2 data type of the department_name column that is retrieved by the first query.

. . .

Matching the SELECT Statement: Example

Using the UNION operator, display the employee name, department_id, and location_id of all employees.

```
SELECT  FIRST_NAME, JOB_ID, TO_DATE(hire_date)"HIRE_DATE"
FROM employees
UNION
SELECT FIRST_NAME, JOB_ID, TO_DATE(NULL)"HIRE_DATE"
FROM retired_employees;
```



	FIRST_NAME	JOB_ID	HIRE_DATE
1	Alex	PU_CLERK	(null)
2	Alexander	IT_PROG	03-JAN-06
3	Alexandera	IT_PROG	(null)
4	Bruce	IT_PROG	21-MAY-07
5	Bruk	IT_PROG	(null)
6	Curtis	ST_CLERK	29-JAN-05
7	Dany	FI_ACCOUNT	(null)
8	Del	PU_MAN	(null)
9	Diana	IT_PROG	07-FEB-07

…

The EMPLOYEES and RETIRED_EMPLOYEES tables have several columns in common (for example, EMPLOYEE_ID, JOB_ID, and DEPARTMENT_ID). But what if you want the query to display the FIRST_NAME, JOB_ID, and HIRE_DATE using the UNION operator, knowing that the HIRE_DATE exists only in the EMPLOYEES table?

The code example in the slide matches the FIRST_NAME and JOB_ID columns in the EMPLOYEES and RETIRED_EMPLOYEES tables. NULL is added to the RETIRED_EMPLOYEES SELECT statement to match the HIRE_DATE column in the EMPLOYEES SELECT statement.

In the results shown in the slide, each row in the output that corresponds to a record from the RETIRED_EMPLOYEES table contains a NULL in the HIRE_DATE column.

Lesson Agenda

- Set operators: Types and guidelines
- Tables used in this lesson
- `UNION` and `UNION ALL` operator
- `INTERSECT` operator
- `MINUS` operator
- Matching `SELECT` statements
- Using the `ORDER BY` clause in set operations

Using the ORDER BY Clause in Set Operations

- The ORDER BY clause can appear only once at the end of the compound query.
- Component queries cannot have individual ORDER BY clauses.
- The ORDER BY clause recognizes only the columns of the first SELECT query.
- By default, the first column of the first SELECT query is used to sort the output in an ascending order.

The ORDER BY clause can be used only once in a compound query. If used, the ORDER BY clause must be placed at the end of the query. The ORDER BY clause accepts the column name or an alias. By default, the output is sorted in ascending order in the first column of the first SELECT query.

Note: The ORDER BY clause does not recognize the column names of the second SELECT query. To avoid confusion over column names, it is a common practice to ORDER BY column positions.

For example, in the following statement, the output will be shown in ascending order of job_id.

```
SELECT employee_id, job_id,salary
FROM   employees
UNION
SELECT employee_id, job_id,0
FROM   retired_employees
ORDER BY 2;
```

If you omit ORDER BY, by default, the output will be sorted in ascending order of employee_id. You cannot use the columns from the second query to sort the output.

Quiz

Identify two set operator guidelines.

a. The expressions in the `SELECT` lists must match in number.

b. Parentheses may not be used to alter the sequence of execution.

c. The data type of each column in the second query must match the data type of its corresponding column in the first query.

d. The `ORDER BY` clause can be used only once in a compound query, unless a `UNION ALL` operator is used.

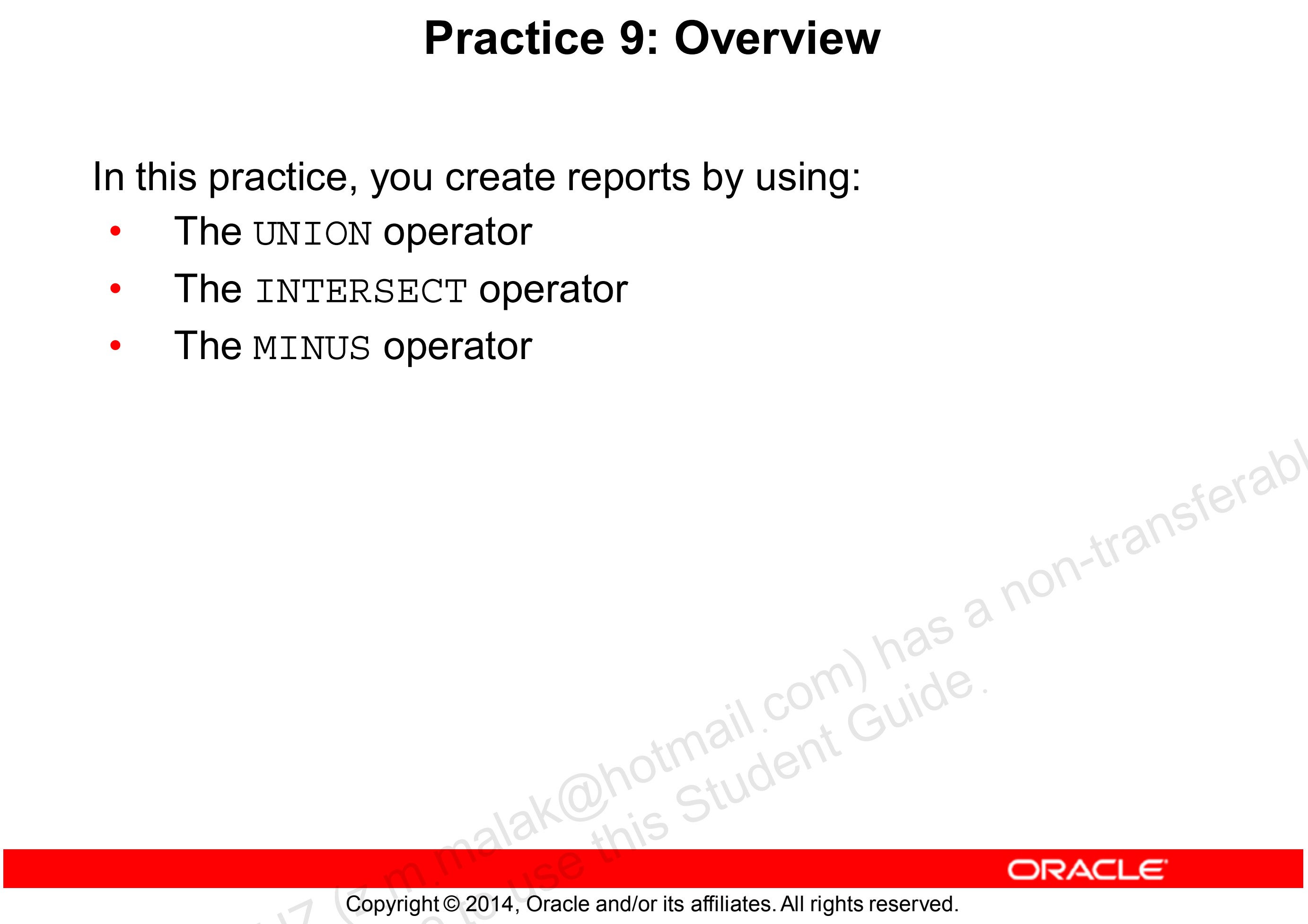
Answer: a, c

Summary

In this lesson, you should have learned how to use:

- UNION to return all distinct rows
- UNION ALL to return all rows, including duplicates
- INTERSECT to return all rows that are shared by both queries
- MINUS to return all distinct rows that are selected by the first query, but not by the second
- ORDER BY only at the very end of the statement

- The UNION operator returns all the distinct rows selected by each query in the compound query. Use the UNION operator to return all rows from multiple tables and eliminate any duplicate rows.
- Use the UNION ALL operator to return all rows from multiple queries. Unlike the case with the UNION operator, duplicate rows are not eliminated and the output is not sorted by default.
- Use the INTERSECT operator to return all rows that are common to multiple queries.
- Use the MINUS operator to return rows returned by the first query that are not present in the second query.
- Remember to use the ORDER BY clause only at the very end of the compound statement.
- Make sure that the corresponding expressions in the SELECT lists match in number and data type.



Practice 9: Overview

In this practice, you create reports by using:

- The `UNION` operator
- The `INTERSECT` operator
- The `MINUS` operator

In this practice, you write queries using the set operators.