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

Course RoadMap

Lesson 1: Introduction

Unit 1: Retrieving, Restricting, and Sorting Data

Unit 2: Joins, Subqueries, and Set Operators

Unit 3: DML and DDL

▷ Lesson 6: Reporting Aggregated Data Using Group Functions

▷ Lesson 7: Displaying Data from Multiple Tables Using Joins

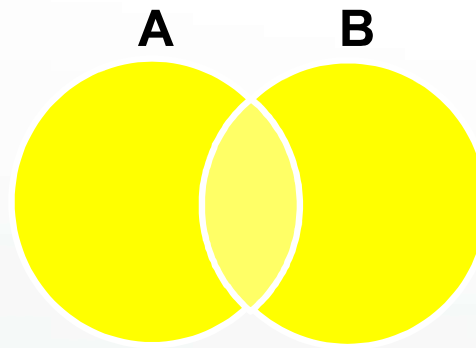▷ Lesson 8: Using Subqueries to Solve Queries
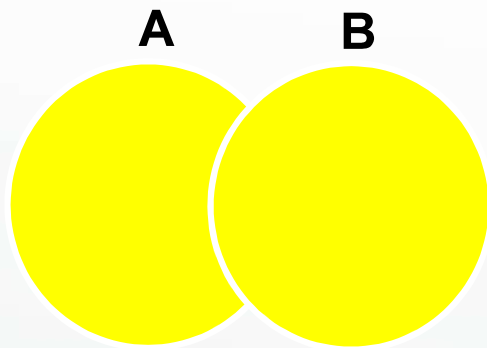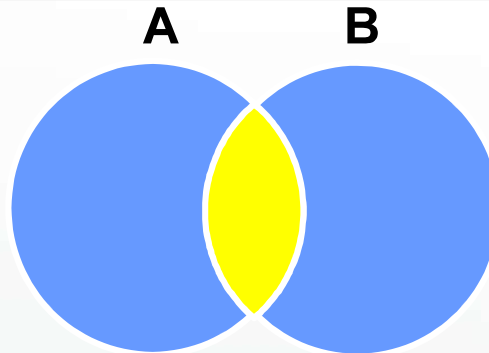
▶ **Lesson 9: Using Set Operators**

You are here!

Lesson Agenda

➢ **Set operators: Types and guidelines**

➢ Tables used in this lesson

➢ `UNION` and `UNION ALL` operator

➢ `INTERSECT` operator

➢ `MINUS` operator
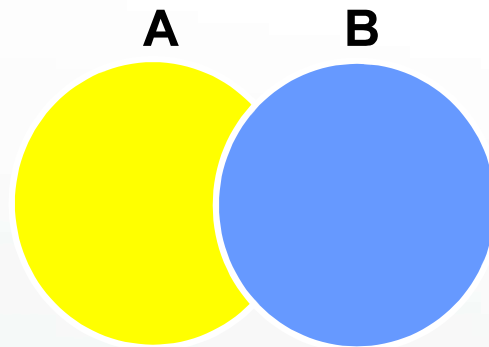
➢ Matching `SELECT` statements

➢ Using the `ORDER BY` clause in set operations

UNION/UNION ALL

INTERSECT
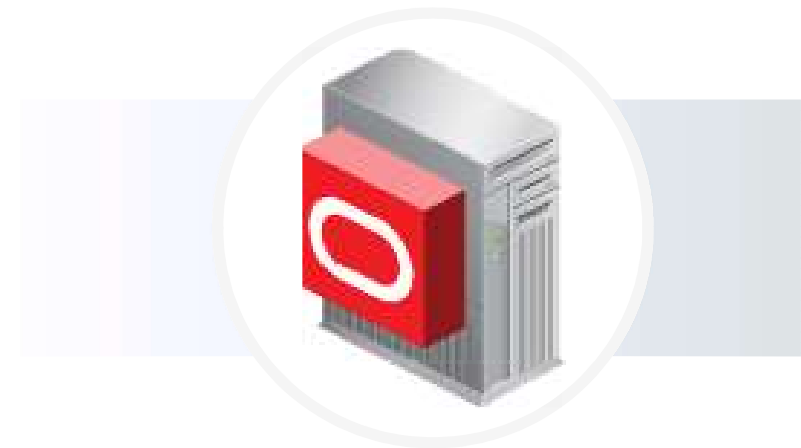
MINUS

➢ 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.

➢ The `ORDER BY` clause can appear only at the very end of the statement.

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`.

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
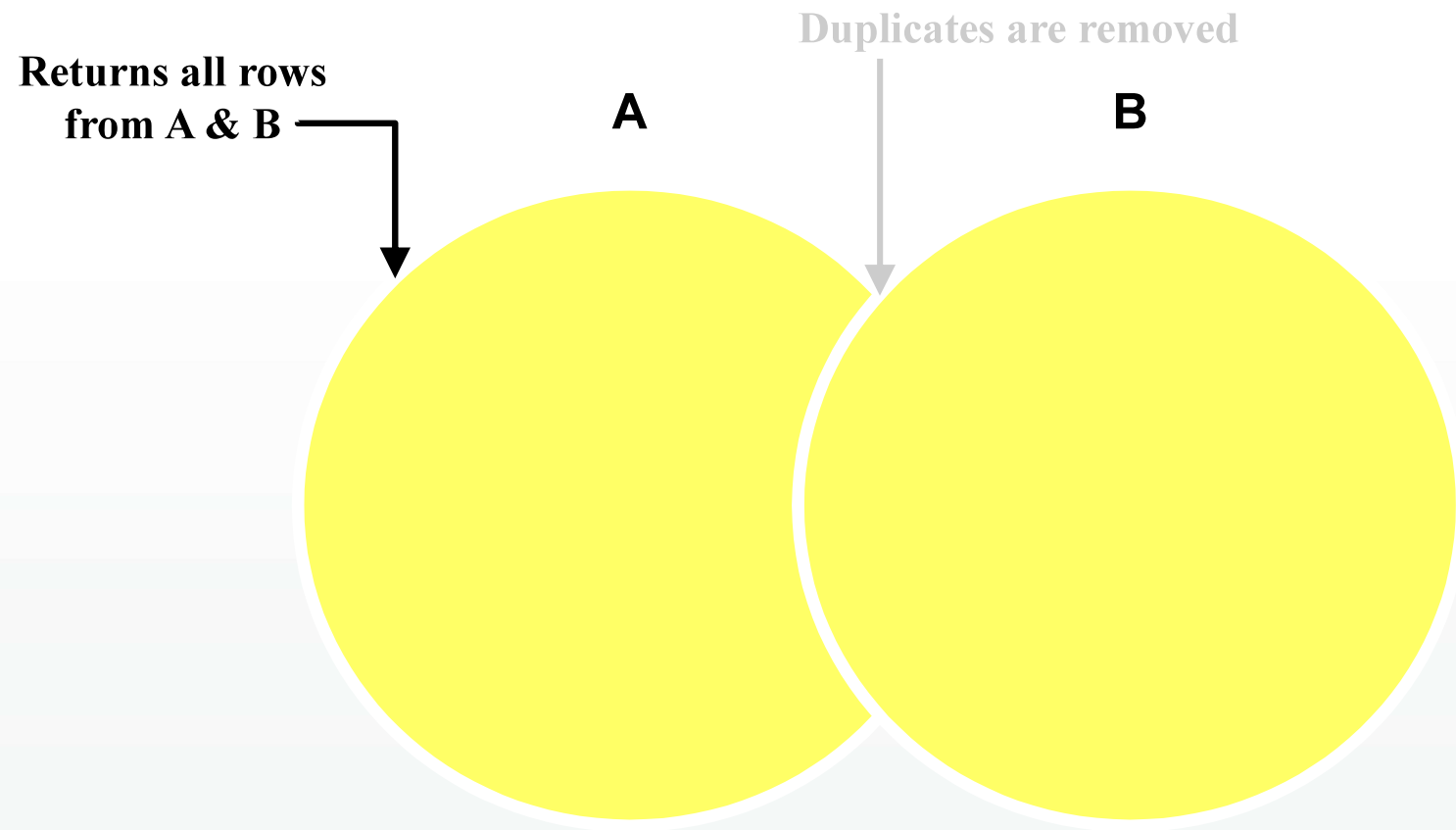
The tables used in this lesson are:

➢ `EMPLOYEES`: Provides details regarding ➢ all current employees

➢ `JOB_HISTORY`: Records the details of the start date and end date of the former job, and the job identification number and department when an employee switches jobs

➢ 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

Duplicates are removed

Returns all rows from A & B

A   B

The UNION operator returns rows from both queries after eliminating duplications.
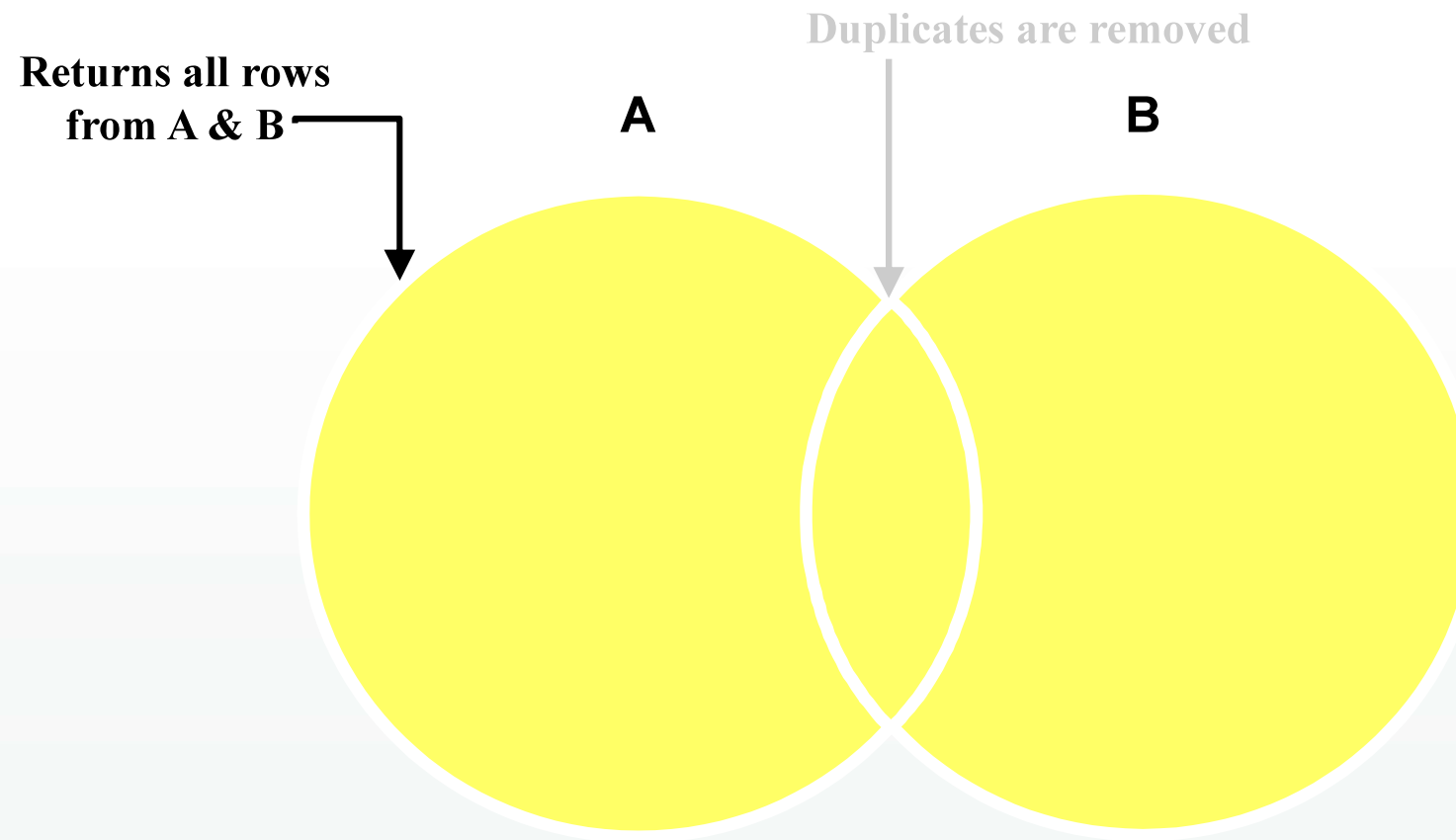
- Display the current and previous job details of all employees. Display each combination only once.

```
SELECT employee_id, job_id
FROM    employees
UNION
SELECT employee_id, job_id
FROM    job_history;
```

	EMPLOYEE_ID	JOB_ID
1	100	AD_PRES
2	101	AC_ACCOUNT

...

| 22 | 200 | AC_ACCOUNT |
| 23 | 200 | AD_ASST |

...

| 28 | 206 | AC_ACCOUNT |

Duplicates are removed

Returns all rows
from A & B

A

B

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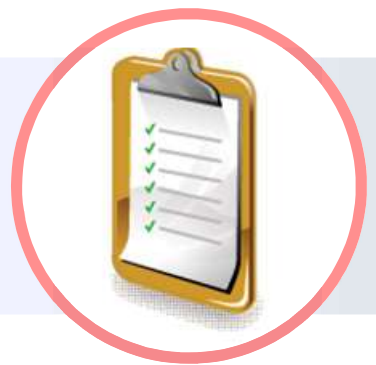queries, including all duplications.

- Display the current and previous departments of all employees.

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

	EMPLOYEE_ID	JOB_ID	DEPARTMENT_ID
1	100	AD_PRES	90
2	101	AD_VP	90

...

	EMPLOYEE_ID	JOB_ID	DEPARTMENT_ID
23	200	AD_ASST	10
24	200	AC_ACCOUNT	90
25	200	AD_ASST	90

...

	EMPLOYEE_ID	JOB_ID	DEPARTMENT_ID
30	206	AC_ACCOUNT	110

➢ 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

**Returns all rows
common to A & B**

A   B

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

- Display the employee IDs and job IDs of those employees who currently have a job title that is the same as a previous job title.

```
SELECT employee_id, job_id
FROM    employees
INTERSECT
SELECT employee_id, job_id
FROM    job_history;
```
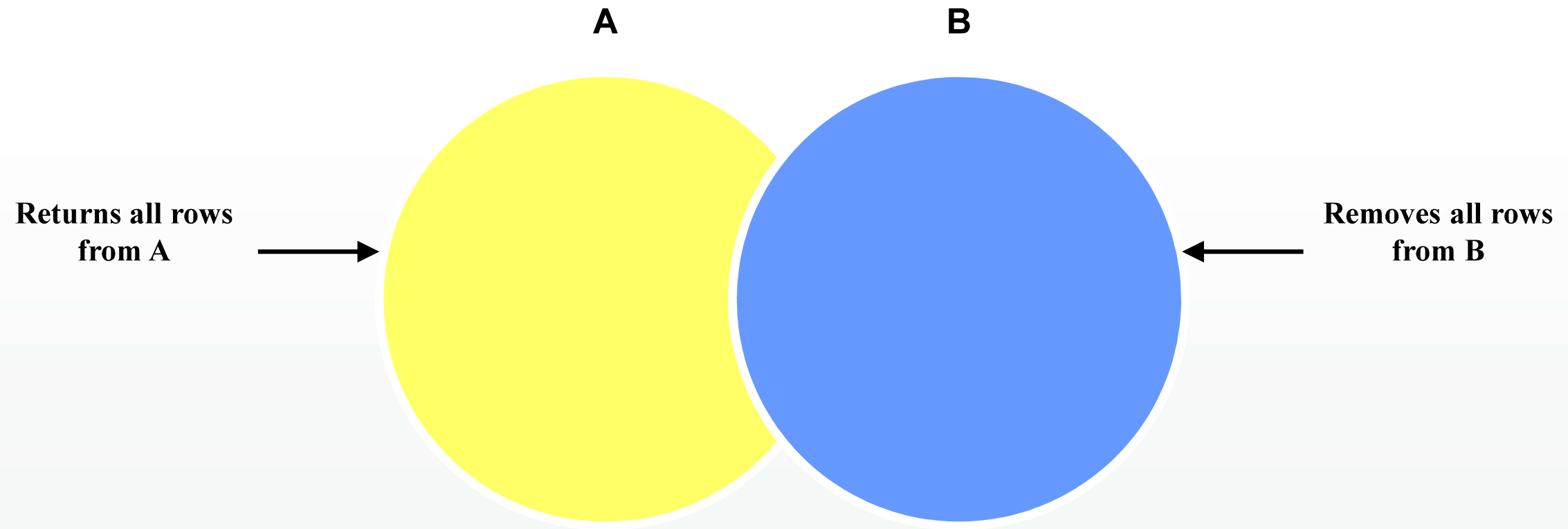
EMPLOYEE_ID	JOB_ID
1	176 SA_REP
2	200 AD_ASST

➢ 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

A **B**

Returns all rows
from A →

← Removes all rows
from B

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

MINUS **Operator**

- Display the employee IDs of those employees who have not changed their jobs even once.

```
SELECT employee_id
FROM    employees
MINUS
SELECT employee_id
FROM    job_history;
```

	EMPLOYEE_ID
1	100
2	103
3	104

...

| 14 | 205 |
| 15 | 206 |

- Using the UNION operator, display the department ID, location, and hire date for all employees.

```
SELECT  department_id, TO_NUMBER(null)
        location, hire_date
FROM    employees
UNION
SELECT  department_id, location_id,  TO_DATE(null)
FROM    departments;
```

	DEPARTMENT_ID	LOCATION	HIRE_DATE
1	10	1700	(null)
2	10	(null)	17-SEP-87
3	20	1800	(null)

...

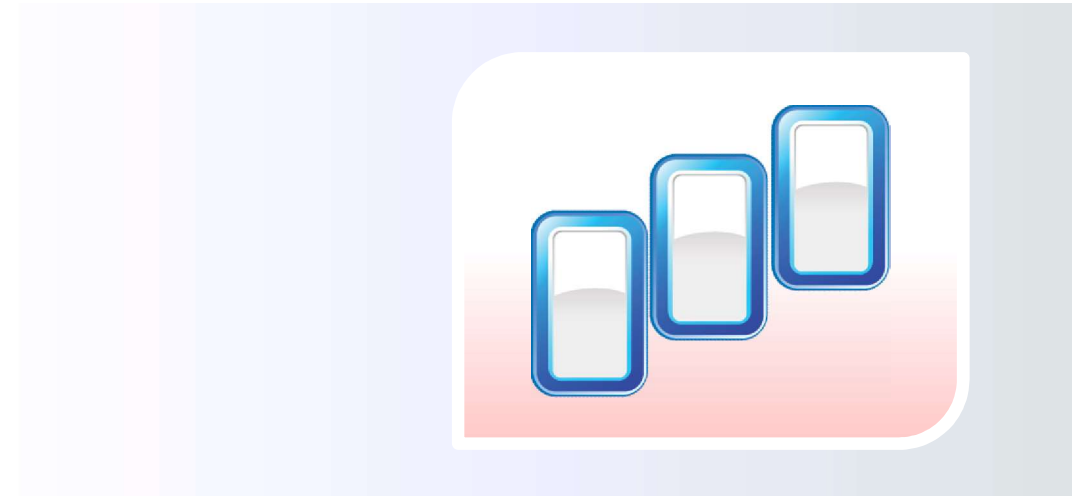| 26 | 190 | 1700 | (null) |
| 27 | (null) | (null) | 24-MAY-99 |

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**

➢ 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 ascending order.

Identify two set operator guidelines.

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

b. Parentheses cannot 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.

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

In this practice, you create reports by using:

➢ The `UNION` operator

➢ The `INTERSECT` operator

➢ The `MINUS` operator