

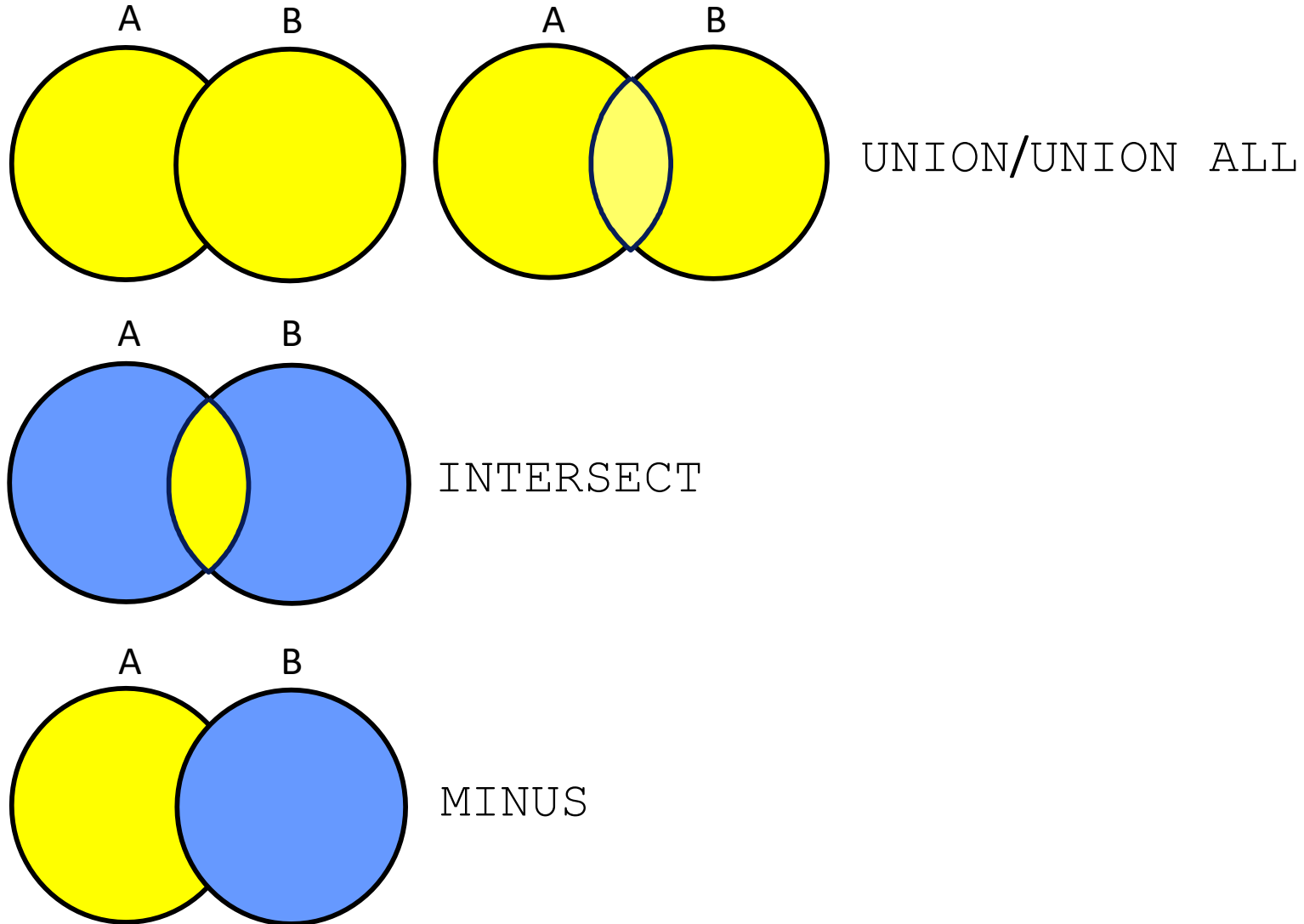
# Using the Set Operators

# 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

# Set Operators

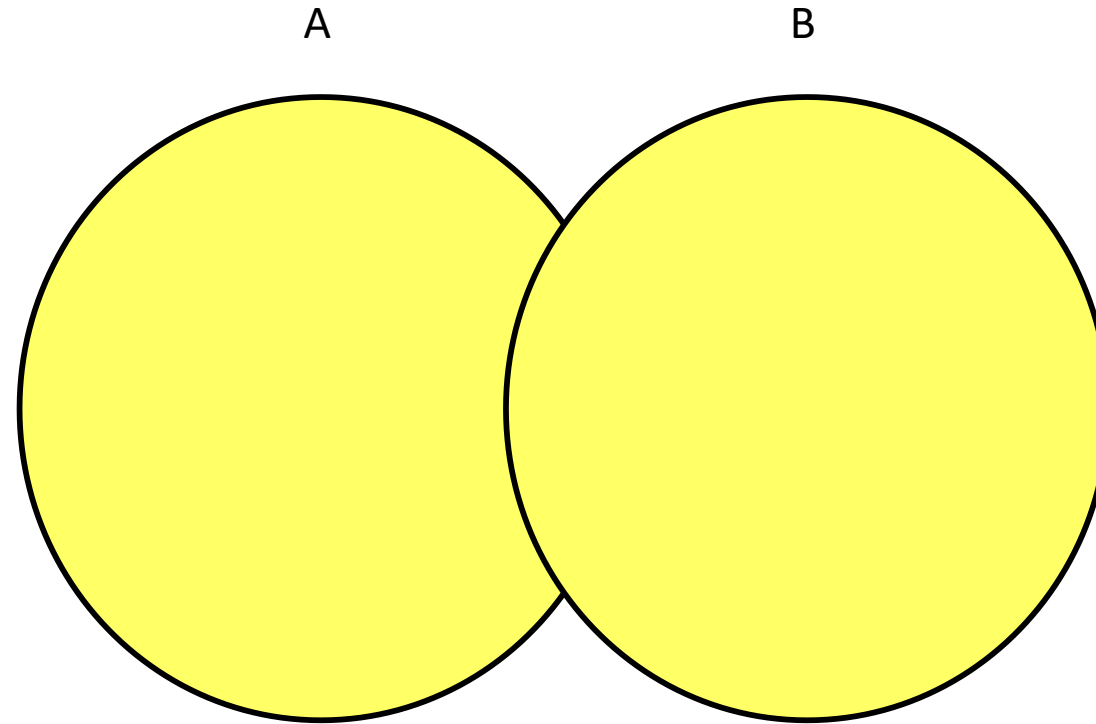


# Tables Used in This Lesson

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

# UNION Operator



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

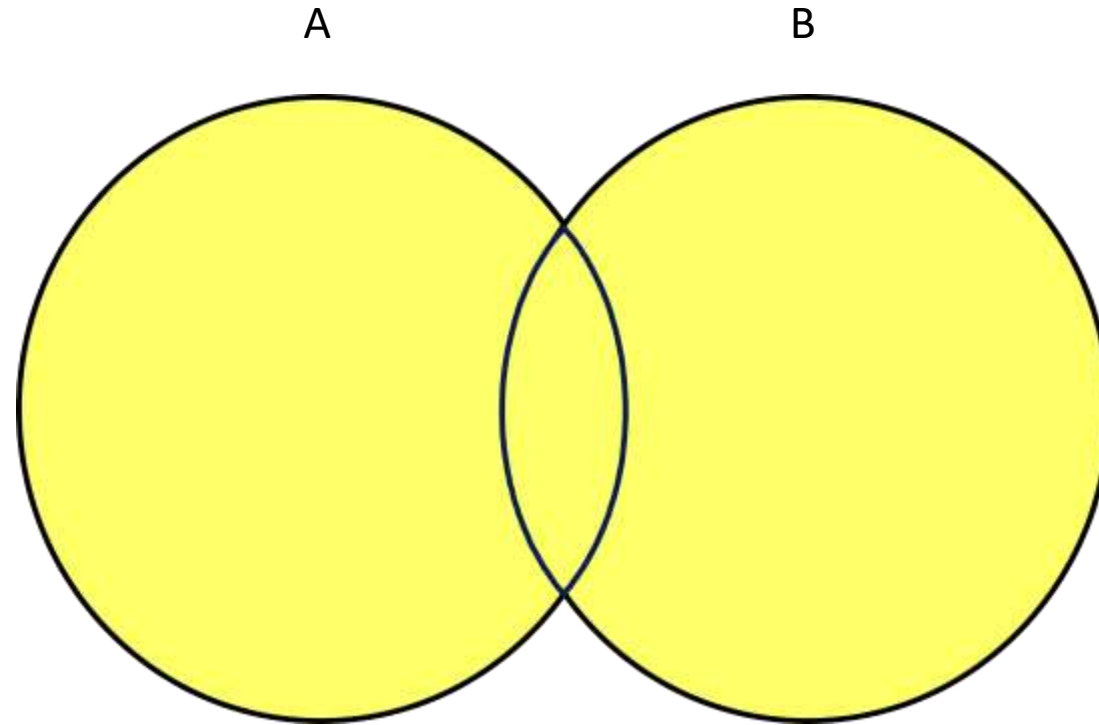
# Using the UNION Operator

Display the current and previous job details of all employees. Display each employee only

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

EMPLOYEE_ID		JOB_ID
	100	AD_PRES
	101	AC_ACCOUNT
...		
	200	AC_ACCOUNT
	200	AD_ASST
...		
	205	AC_MGR
	206	AC_ACCOUNT

# UNION ALL Operator



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

# Using the UNION ALL Operator

Display the current and previous departments of all employees.

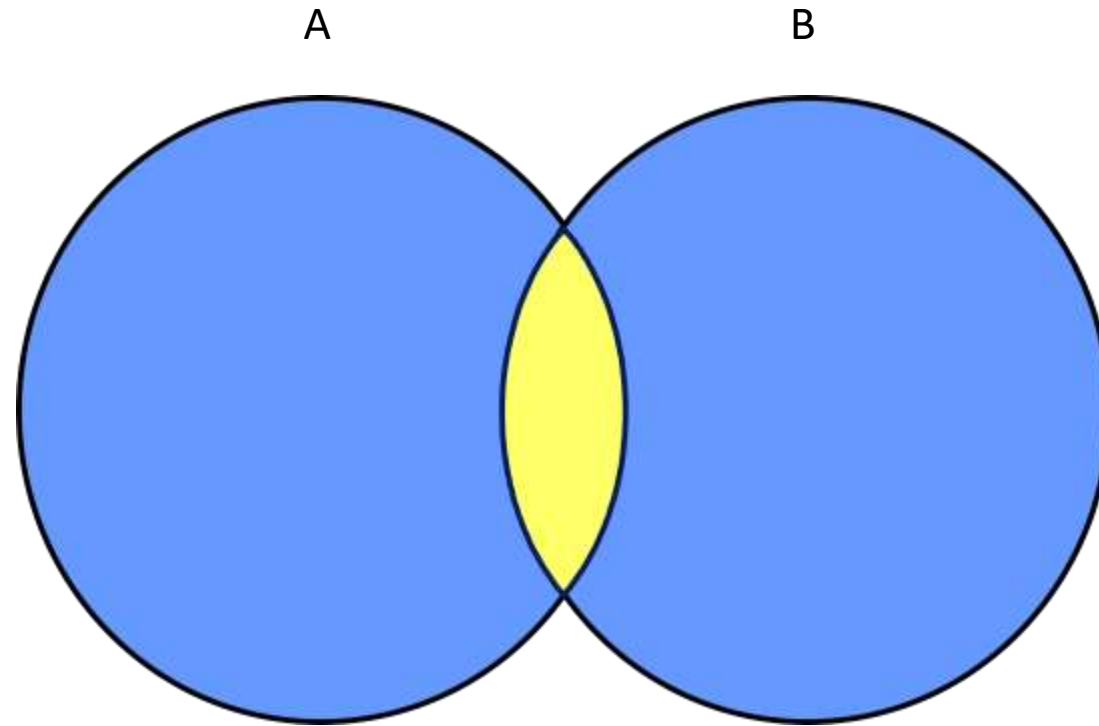
```
SELECT employee_id, job_id, department_id
FROM   employees
UNION ALL
SELECT employee_id, job_id, department_id
FROM   job_history
ORDER BY employee_id;
```

EMPLOYEE_ID	JOB_ID	DEPARTMENT_ID
100	AD_PRES	90
101	AD_VP	90
...		
200	AD_ASST	10
200	AD_ASST	90
200	AC_ACCOUNT	90
...		
205	AC_MGR	110
206	AC_ACCOUNT	110

30 rows selected.



# INTERSECT Operator



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

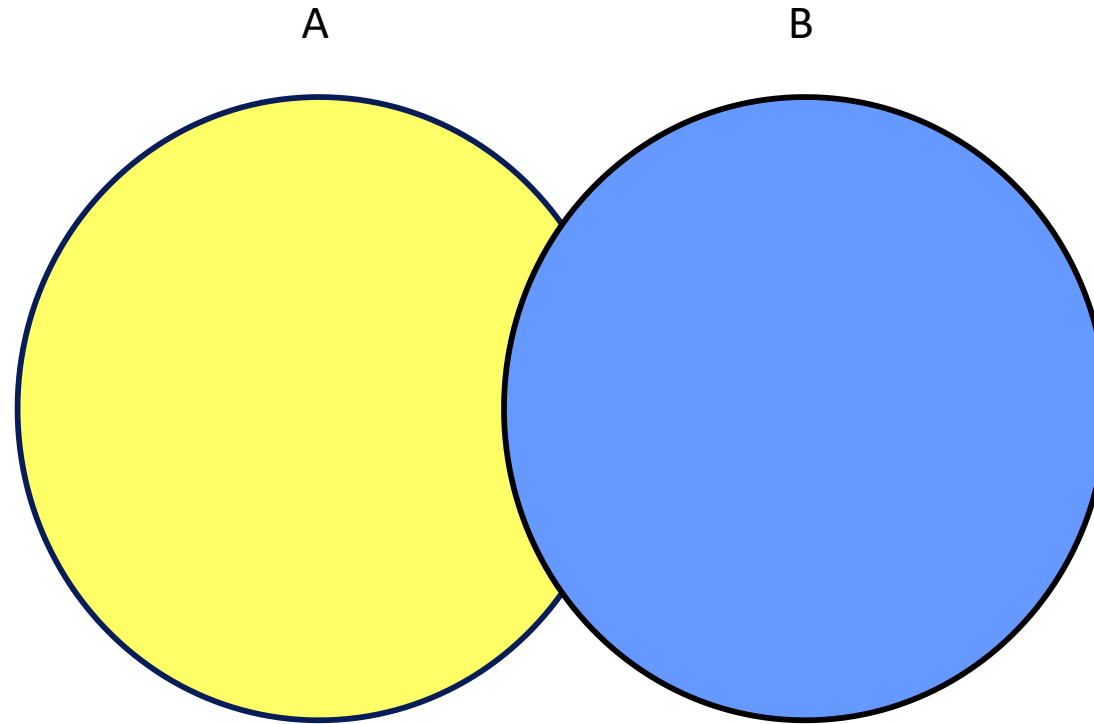
# Using the INTERSECT Operator

Display 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 (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
176	SA_REP
200	AD_ASST

# MINUS Operator



The `MINUS` operator returns rows in the first query that are not present in the second query.

# MINUS Operator

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

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

EMPLOYEE_ID	
	100
	103
	104
	107
...	
	205
	206

15 rows selected.

# Set Operator Guidelines

- The expressions in the `SELECT` lists must match in number and data type.
- Parentheses can be used to alter the sequence of execution.
- The `ORDER BY` clause:
  - Can appear only at the very end of the statement
  - Will accept the column name, aliases from the first `SELECT` statement, or the positional notation

# The 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`.

# Matching the SELECT Statements

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
10	1700	
10		17-SEP-87
20	1800	
20		17-FEB-96
...		
110	1700	
110		07-JUN-94
190	1700	
		24-MAY-99

27 rows selected.

# Matching the SELECT Statement:

## Example

Using the UNION operator, display the employee ID, job ID, and salary of all employees.

```
SELECT employee_id, job_id, salary
FROM   employees
UNION
SELECT employee_id, job_id, 0
FROM   job_history;
```

EMPLOYEE_ID	JOB_ID	SALARY
100	AD_PRES	24000
101	AC_ACCOUNT	0
101	AC_MGR	0
...		
205	AC_MGR	12000
206	AC_ACCOUNT	8300

30 rows selected.



# Controlling the Order of Rows

Produce an English sentence using two UNION operators.

```
COLUMN a_dummy NOPRINT
SELECT 'sing' AS "My dream", 3 a_dummy
FROM dual
UNION
SELECT 'I'd like to teach', 1 a_dummy
FROM dual
UNION
SELECT 'the world to', 2 a_dummy
FROM dual
ORDER BY a_dummy;
```

My dream
I'd like to teach
the world to
sing

# Summary

In this lesson, you should have learned how to:

- Use `UNION` to return all distinct rows
- Use `UNION ALL` to return all rows, including duplicates
- Use `INTERSECT` to return all rows that are shared by both queries
- Use `MINUS` to return all distinct rows that are selected by the first query but not by the second

# Practice 7: Overview

In this practice, you use the set operators to create reports:

- Using the `UNION` operator
- Using the `INTERSECTION` operator
- Using the `MINUS` operator