Oracle 11g - SQL

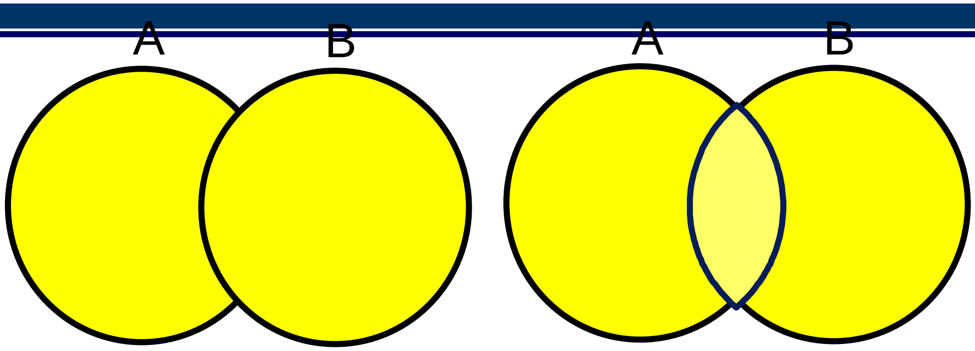
Set Operators

Objectives

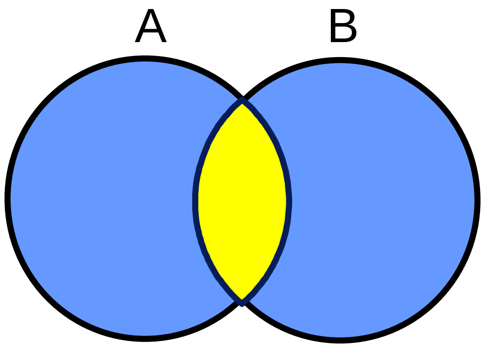
After completing this lesson, you should be able to do the following:

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

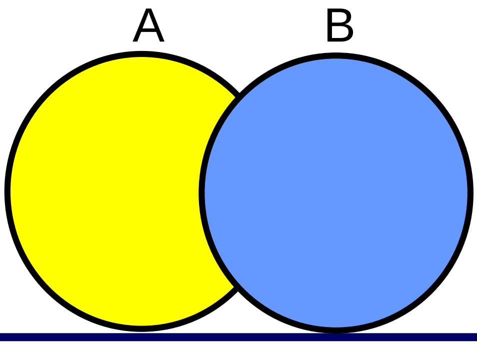
Set Operators



UNION/UNION ALL



INTERSECT



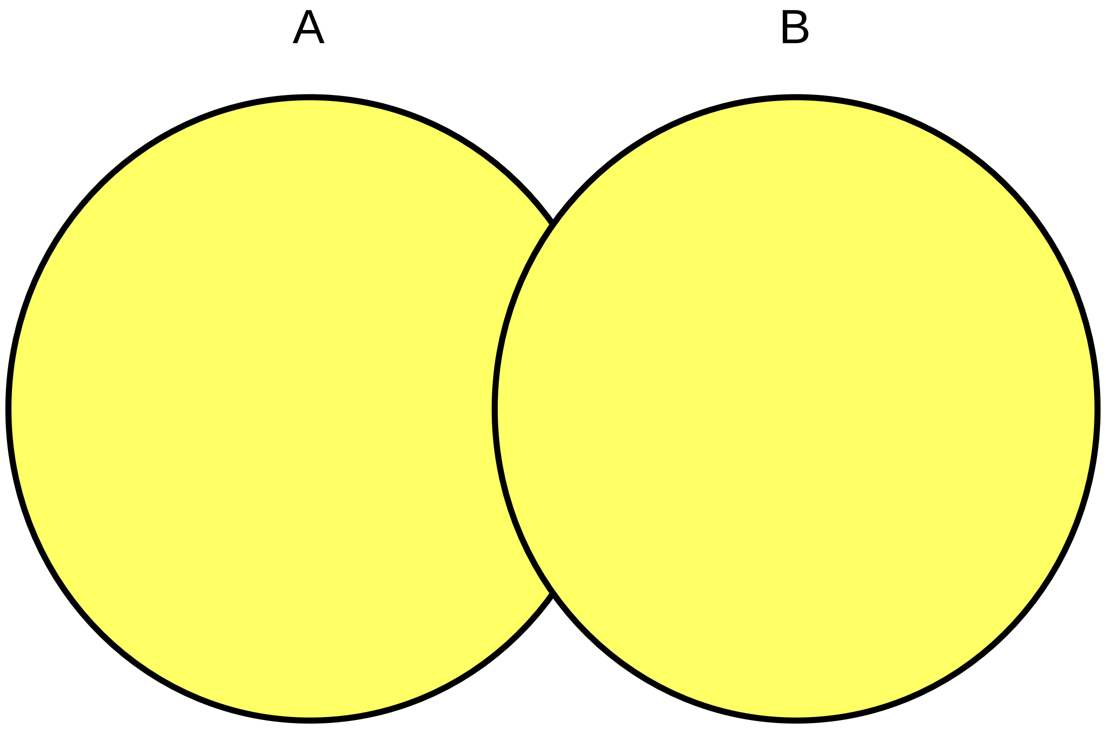
MINUS

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

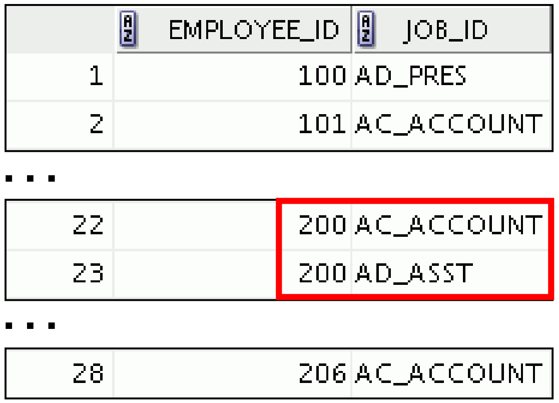


The `UNION` operator returns results from both queries after eliminating duplications.

Using the `UNION` Operator

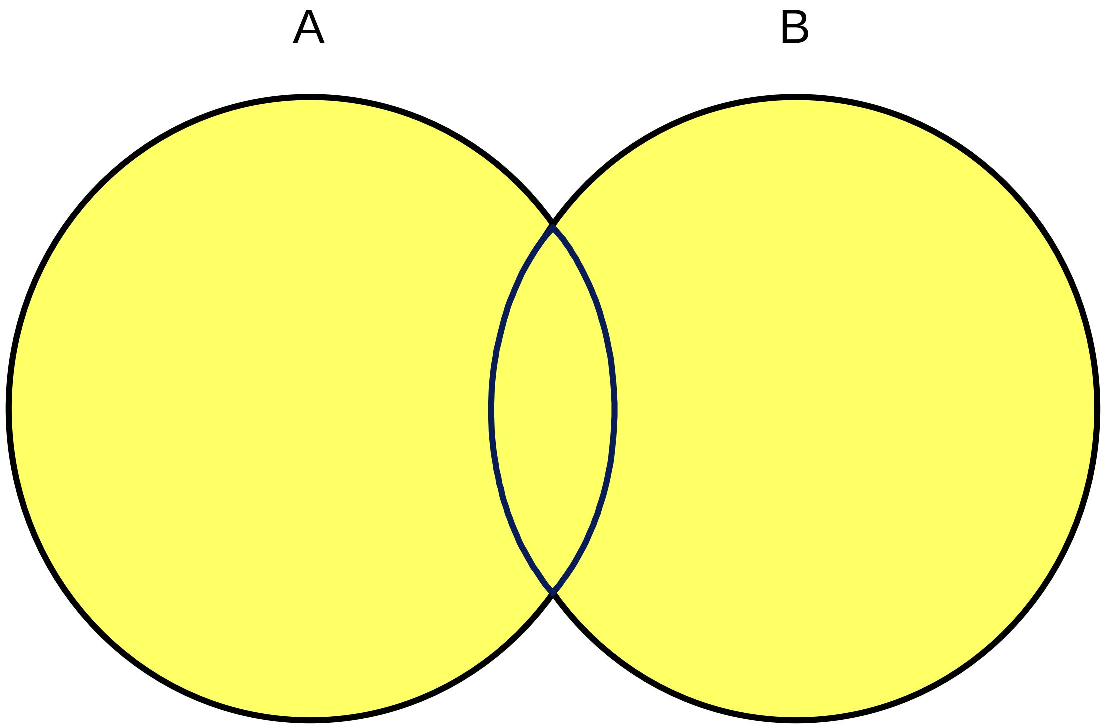
Display the current and previous job details of all employees. Display each combination only once.

```
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

UNION ALL Operator

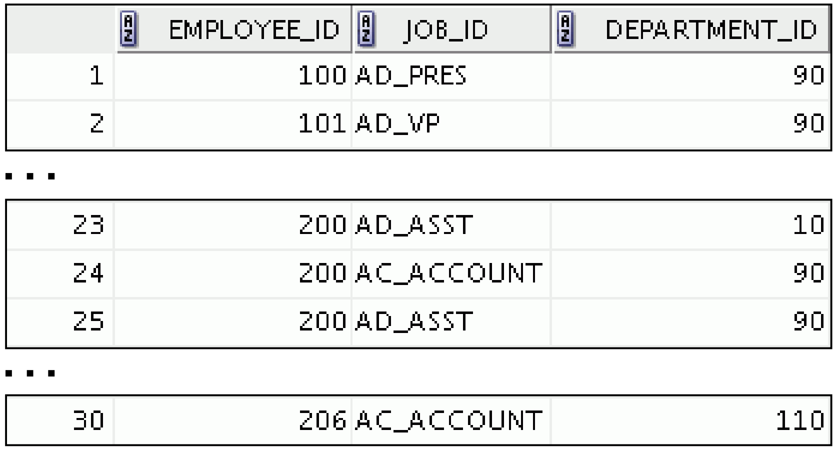


The UNION ALL operator returns results from both queries, including all duplications.

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
1	100	AD_PRES	90
2	101	AD_VP	90

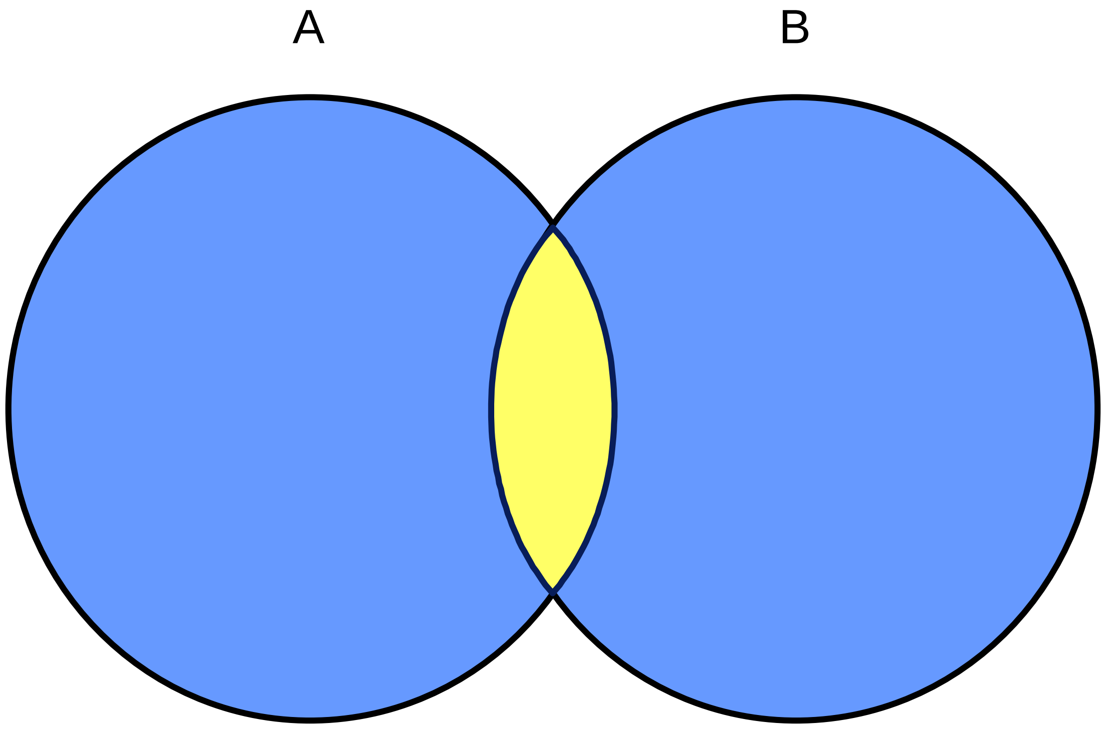
...

	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

`INTERSECT` Operator

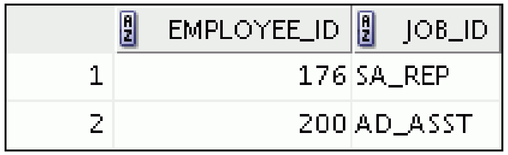


The `INTERSECT` operator returns rows that are common to both queries.

Using the INTERSECT Operator

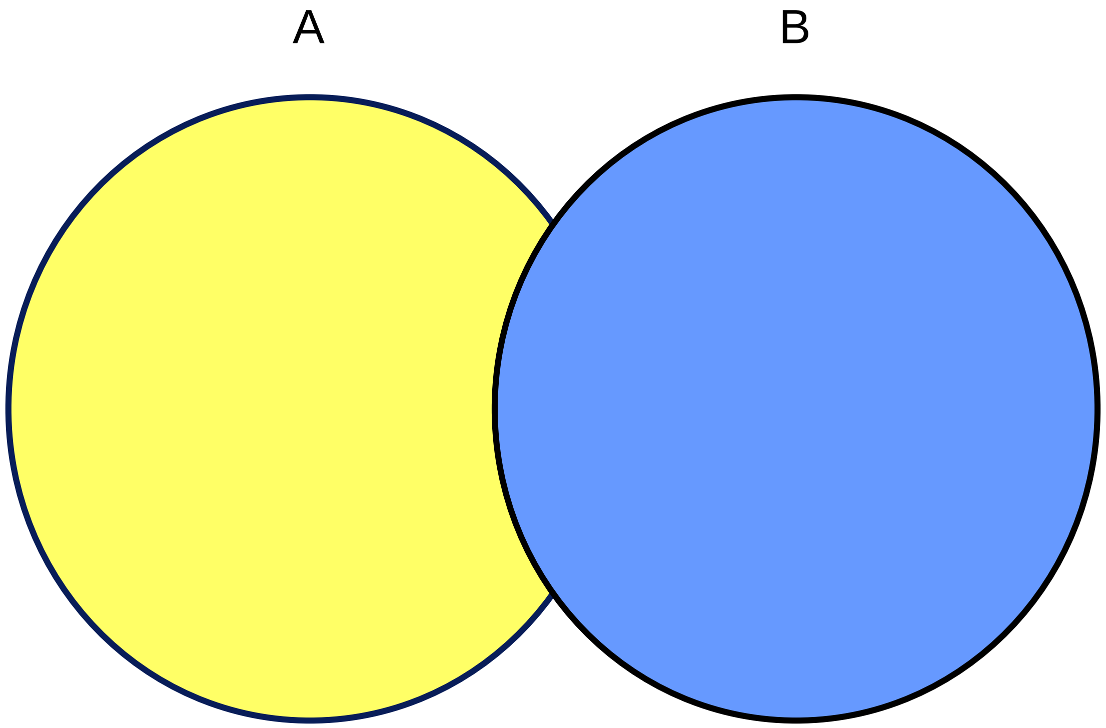
Display the employee IDs and job IDs of those employees who currently have a job title that is the same as a previous job title.

```
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

`MINUS` Operator

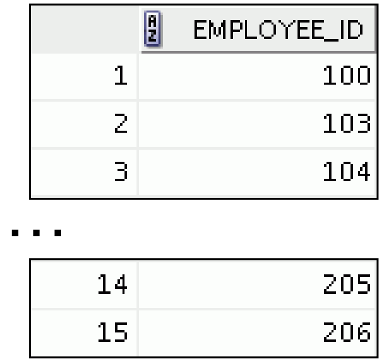


The `MINUS` operator returns rows in the first query that are not present in the second query.

MINUS Operator

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

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



	EMPLOYEE_ID
1	100
2	103
3	104
...	
14	205
15	206

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

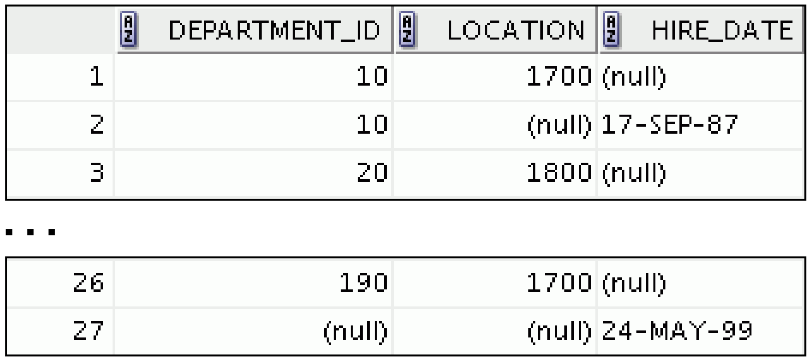
Oracle Server and Set Operators

- 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;
```

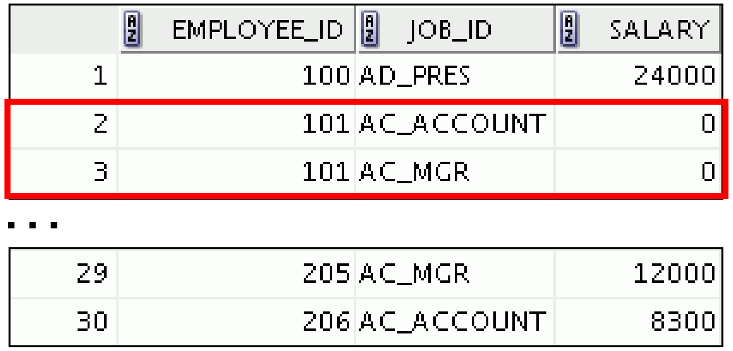


	DEPARTMENT_ID	LOCATION	HIRE_DATE
1	10	1700	(null)
2	10	(null)	17-SEP-87
3	20	1800	(null)
...			
26	190	1700	(null)
27	(null)	(null)	24-MAY-99

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
1	100	AD_PRES	24000
2	101	AC_ACCOUNT	0
3	101	AC_MGR	0
...			
29	205	AC_MGR	12000
30	206	AC_ACCOUNT	8300