ORACLE

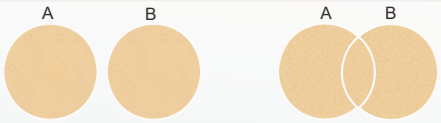
9

Using Set Operators

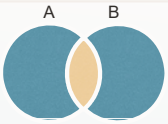
Lesson Agenda

- 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

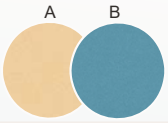
Set Operators



UNION/UNION ALL



INTERSECT



MINUS

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.
- 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`.

5

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

6

Tables Used in This Lesson

The tables used in this lesson are:

- `employees:` Provides details about all current employees
- `retired_employees:` Provides details about all past employees

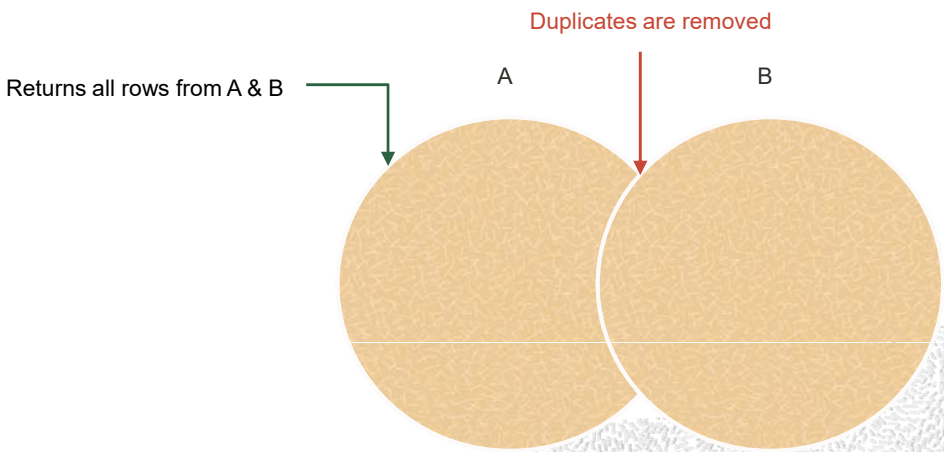
7

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

8

UNION Operator



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

Using the UNION Operator

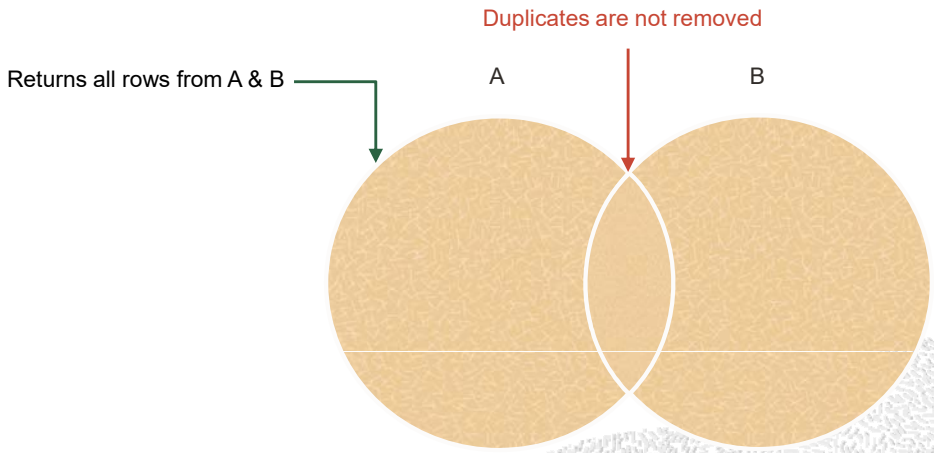
Display the job details of all the current and retired employees. Display each job only once.

```
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

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

UNION ALL Operator



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

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

...

#	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	90
8	AD_VP	80
9	AD_VP	90

28	SA_REP	NULL
29	SA_REP	80
30	SA_REP	80
31	ST_CLERK	50
32	ST_CLERK	50
33	ST_CLERK	50
34	ST_CLERK	50
35	ST_MAN	50

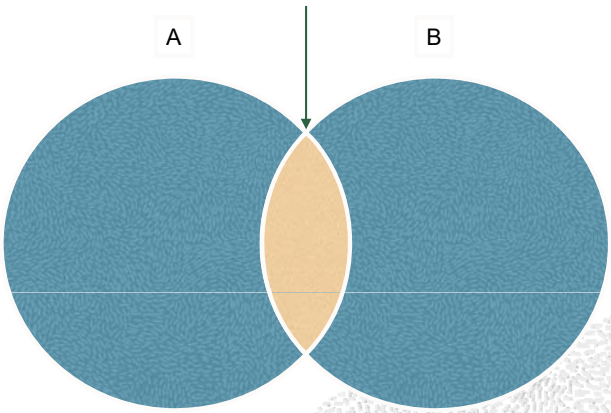
...

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

Returns all rows common to A & B



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

Using the INTERSECT Operator

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

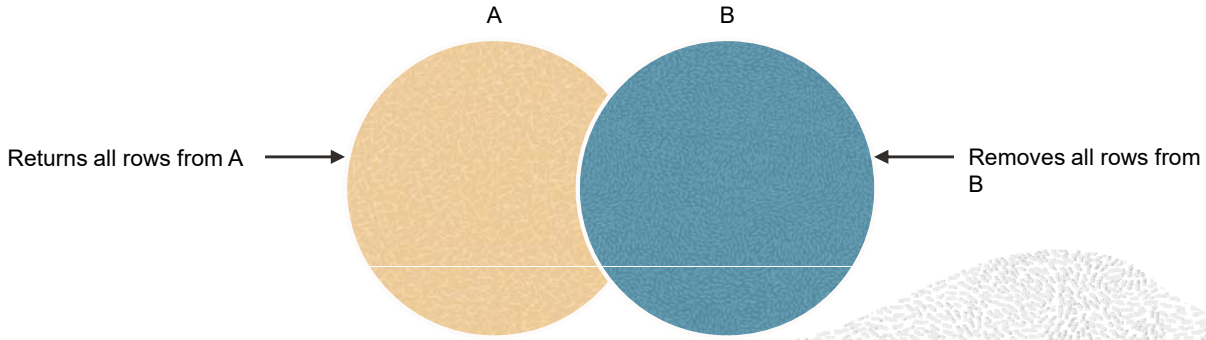
```
SELECT  manager_id,department_id
FROM   employees
INTERSECT
SELECT manager_id,department_id
FROM   retired_employees;
```

	MANAGER_ID	DEPARTMENT_ID
1	149	80

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.

Using the MINUS Operator

Display the manager IDs and Job IDs of employees whose managers have never managed retired employees in the sales department.

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

	MANAGER_ID	JOB_ID
1	100	SA_MAN
2	149	SA_REP

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 in Oracle

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;
```

Matching the SELECT Statement: Example in Oracle

Using the UNION operator, display the employee name, job ID, and hire date of all employees.

```
SELECT  FIRST_NAME, JOB_ID, 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-14
3	Alexandera	IT_PROG	(null)
4	Bruce	IT_PROG	21-MAY-15
5	Bruk	IT_PROG	(null)
6	Curtis	ST_CLERK	29-JAN-13
7	Dany	FI_ACCOUNT	(null)
8	Del	PU_MAN	(null)

...

Matching SELECT Statements in MySQL

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

```
SELECT location_id, department_name 'Department',
   CAST(NULL AS CHAR) 'Warehouse location'
FROM departments
UNION
SELECT location_id, CAST(NULL AS CHAR),
   state_province
FROM locations;
```

#	location_i	Department	Warehouse locatio
1	1700	Administration	NULL
2	1800	Marketing	NULL
3	1500	Shipping	NULL
4	1400	IT	NULL
5	2500	Sales	NULL
6	1700	Executive	NULL
7	1700	Accounting	NULL
8	1700	Contracting	NULL
9	1500	NULL	California
10	1800	NULL	Ontario
11	2500	NULL	Oxford
12	1400	NULL	Texas
13	1700	NULL	Washington

Matching the SELECT Statement: Example in MySQL

Using the `UNION` operator, display the employee name, job ID, and hire date of all employees.

```
SELECT  first_name, job_id, hire_date 'Hire Date'
FROM employees
UNION
SELECT first_name, job_id, CAST(NULL AS DATE)
FROM retired_employees;
```

#	first_name	job_id	Hire Date
1	Steven	AD_PRES	2011-06-17
2	Neena	AD_VP	2009-09-21
3	Lex	AD_VP	2009-01-13
4	Alexander	IT_PROG	2014-01-03
5	Bruce	IT_PROG	2015-05-21
6	Diana	IT_PROG	2015-02-07
7	Kevin	ST_MAN	2015-11-16
8	Trenna	ST_CLERK	2011-10-17
9	Curtis	ST_CLERK	2013-01-29

...

28	Rahul	FI_MGR	NULL
29	Dany	FI_ACCOUNT	NULL
30	James	FI_ACCOUNT	NULL
31	Shana	FI_ACCOUNT	NULL
32	Peter	FI_ACCOUNT	NULL
33	Lui	FI_ACCOUNT	NULL
34	Del	PU_MAN	NULL
35	Alex	PU_CLERK	NULL

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 in Oracle

- 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.

Using the ORDER BY Clause in Set Operations in Oracle: Example

Display the employee ID and job ID of all current and retired employees, sorted by job ID.

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

	EMPLOYEE_ID	JOB_ID
1	206	AC_ACCOUNT
2	205	AC_MGR
3	200	AD_ASST
4	100	AD_PRES
5	301	AD_PRES
6	101	AD_VP
7	102	AD_VP
8	302	AD_VP
9	303	AD_VP
10	309	FI_ACCOUNT
11	310	FI_ACCOUNT
12	311	FI_ACCOUNT
13	312	FI_ACCOUNT
14	313	FI_ACCOUNT

...

28	174	SA_REP
29	176	SA_REP
30	178	SA_REP
31	141	ST_CLERK
32	142	ST_CLERK
33	143	ST_CLERK
34	144	ST_CLERK
35	124	ST_MAN

Using the ORDER BY Clause with UNION in MySQL

- To use an ORDER BY clause to sort the entire UNION result, place the ORDER BY clause only once at the end of the compound query.
- The ORDER BY clause uses the columns of the first SELECT query.
- If a column to be sorted is aliased, the ORDER BY clause must use the alias rather than the column name.

Using the ORDER BY Clause with UNION: Example in MySQL

Display the employee ID and job ID of all current and retired employees, sorted by job ID.

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

#	employee_ic	job_id
1	206	AC_ACCOUNT
2	205	AC_MGR
3	200	AD_ASST
4	100	AD_PRES
5	301	AD_PRES
6	101	AD_VP
7	102	AD_VP
8	303	AD_VP
9	302	AD_VP

...

28	178	SA_REP
29	176	SA_REP
30	174	SA_REP
31	144	ST_CLERK
32	143	ST_CLERK
33	142	ST_CLERK
34	141	ST_CLERK
35	124	ST_MAN

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

29