15

Using SET Operators

ORACLE

Schedule:	Timing	Topic
	30 minutes	Lecture
	20 minutes	Practice
	50 minutes	Total

Objectives

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

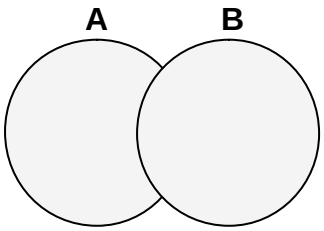
- **Describe SET operators**
- **Use a SET operator to combine multiple queries into a single query**
- **Control the order of rows returned**

ORACLE

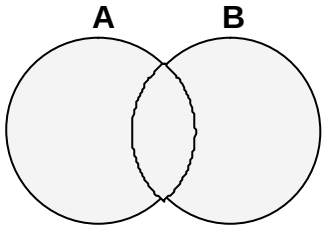
Lesson Aim

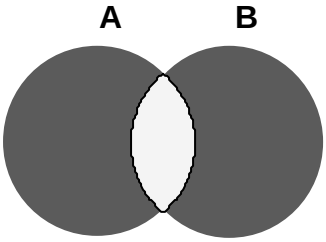
In this lesson, you learn how to write queries by using SET operators.

The SET Operators

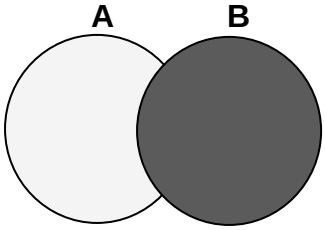


UNION/UNION ALL





INTERSECT



MINUS

ORACLE

The SET Operators

The 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	All distinct rows selected by either query
UNION ALL	All rows selected by either query, including all duplicates
INTERSECT	All distinct rows selected by both queries
MINUS	All distinct rows that are selected by the first SELECT statement and not selected in the second SELECT statement

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.

Note: In the slide, the light color (gray) in the diagram represents the query result.

Instructor Note

The INTERSECT and MINUS operators are not ANSI SQL-99 compliant. They are Oracle-specific.

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**

ORACLE

Tables Used in This Lesson

Two tables are used in this lesson. They are the EMPLOYEES table and the JOB_HISTORY table.

The EMPLOYEES table stores the employee details. For the human resource records, this table stores a unique identification number and email address for each employee. The details of the employee's job identification number, salary, and manager are also stored. Some of the employees earn a commission in addition to their salary; this information is tracked too. The company organizes the roles of employees into jobs. Some of the employees have been with the company for a long time and have switched to different jobs. This is monitored using the JOB_HISTORY table. When an employee switches jobs, the details of the start date and end date of the former job, the job identification number and department are recorded in the JOB_HISTORY table.

The structure and the data from the EMPLOYEES and the JOB_HISTORY tables are shown on the next page.

There have been instances in the company of people who have held the same position more than once during their tenure with the company. For example, consider the employee Taylor, who joined the company on 24-MAR-1998. Taylor held the job title SA_REP for the period 24-MAR-98 to 31-DEC-98 and the job title SA_MAN for the period 01-JAN-99 to 31-DEC-99. Taylor moved back into the job title of SA_REP, which is his current job title.

Similarly consider the employee Whalen, who joined the company on 17-SEP-1987. Whalen held the job title AD_ASST for the period 17-SEP-87 to 17-JUN-93 and the job title AC_ACCOUNT for the period 01-JUL-94 to 31-DEC-98. Whalen moved back into the job title of AD_ASST, which is his current job title.

Tables Used in This Lesson (continued)

```
DESC 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)
DEPARTMENT_NAME		VARCHAR2(14)

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

EMPLOYEE_ID	LAST_NAME	JOB_ID	HIRE_DATE	DEPARTMENT_ID
100	King	AD_PRES	17-JUN-87	90
101	Kochhar	AD_VP	21-SEP-89	90
102	De Haan	AD_VP	13-JAN-93	90
103	Hunold	IT_PROG	03-JAN-90	60
104	Ernst	IT_PROG	21-MAY-91	60
107	Lorentz	IT_PROG	07-FEB-99	60
124	Mourgos	ST_MAN	16-NOV-99	50
141	Rajs	ST_CLERK	17-OCT-95	50
142	Davies	ST_CLERK	29-JAN-97	50
143	Matos	ST_CLERK	15-MAR-98	50
144	Vargas	ST_CLERK	09-JUL-98	50
149	Zlotkey	SA_MAN	29-JAN-00	80
174	Abel	SA_REP	11-MAY-96	80
176	Taylor	SA_REP	24-MAR-98	80
EMPLOYEE_ID	LAST_NAME	JOB_ID	HIRE_DATE	DEPARTMENT_ID
178	Grant	SA_REP	24-MAY-99	
200	Whalen	AD_ASST	17-SEP-87	10
201	Hartstein	MK_MAN	17-FEB-96	20

▪ ▪ ▪

```
DESC job_history
```

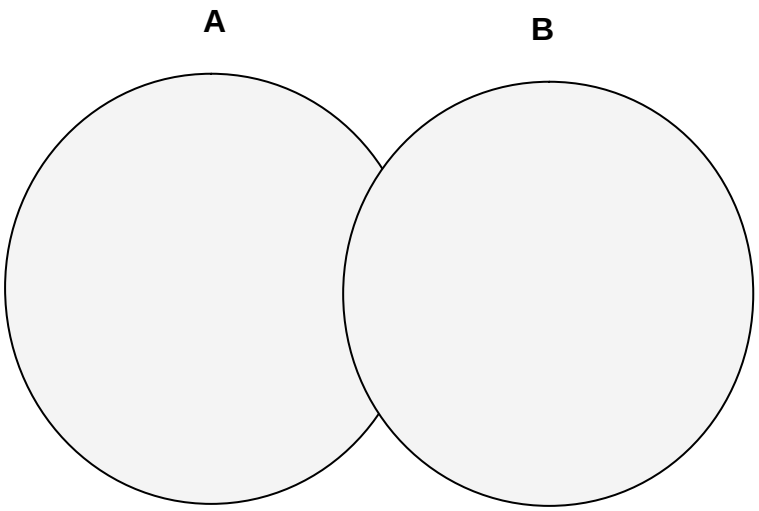
Name	Null?	Type
EMPLOYEE_ID	NOT NULL	NUMBER(6)
START_DATE	NOT NULL	DATE
END_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR2(10)
DEPARTMENT_ID		NUMBER(4)

```
SELECT * FROM job_history;
```

EMPLOYEE_ID	START_DAT	END_DATE	JOB_ID	DEPARTMENT_ID
102	13-JAN-93	24-JUL-98	IT_PROG	60
101	21-SEP-89	27-OCT-93	AC_ACCOUNT	110
101	28-OCT-93	15-MAR-97	AC_MGR	110
201	17-FEB-96	19-DEC-99	MK_REP	20
114	24-MAR-98	31-DEC-99	ST_CLERK	50
122	01-JAN-99	31-DEC-99	ST_CLERK	50
200	17-SEP-87	17-JUN-93	AD_ASST	90
176	24-MAR-98	31-DEC-98	SA_REP	80
176	01-JAN-99	31-DEC-99	SA_MAN	80
200	01-JUL-94	31-DEC-98	AC_ACCOUNT	90

10 rows selected.

The UNION Operator



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

ORACLE

The UNION Operator

The UNION operator returns all rows 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 and the datatypes of the columns being selected must be identical in all the SELECT statements used in the query. 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.
- The IN operator has a higher precedence than the UNION operator.
- By default, the output is sorted in ascending order of the first column of the SELECT clause.

Instructor Note

To illustrate the UNION SET operator, run the script demo\15_union1.sql.

Point out that the output is sorted in ascending order of the first column of the SELECT clause.

Using the UNION Operator

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

```
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

ORACLE

Using the UNION SET Operator

The UNION operator eliminates any duplicate records. If there are records that occur both in the EMPLOYEES and the JOB_HISTORY tables and are identical, the records will be displayed only once. Observe in the output shown on the slide that the record for the employee with the EMPLOYEE_ID 200 appears twice as the JOB_ID is different in each row.

Consider the following example:

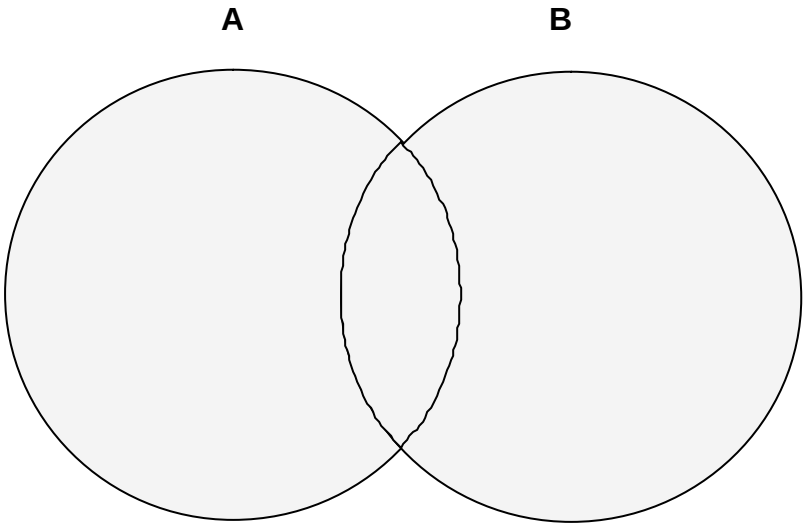
```
SELECT  employee_id, job_id, department_id
FROM    employees
UNION
SELECT  employee_id, job_id, department_id
FROM    job_history;
```

EMPLOYEE_ID	JOB_ID	DEPARTMENT_ID
...		
200	AC_ACCOUNT	90
200	AD_ASST	10
200	AD_ASST	90
...		

29 rows selected.

In the preceding output, employee 200 appears three times. Why? Notice the DEPARTMENT_ID values for employee 200. One row has a DEPARTMENT_ID of 90, another 10, and the third 90. Because of these unique combinations of job IDs and department IDs, each row for employee 200 is unique and therefore not considered a duplicate. Observe that the output is sorted in ascending order of the first column of the SELECT clause, EMPLOYEE_ID in this case.

The UNION ALL Operator



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

ORACLE

The UNION ALL Operator

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

Guidelines

- Unlike UNION, duplicate rows are not eliminated and the output is not sorted by default.
- The DISTINCT keyword cannot be used.

Note: With the exception of the above, the guidelines for UNION and UNION ALL are the same.

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.

ORACLE

The UNION ALL Operator (continued)

In the example, 30 rows are selected. The combination of the two tables totals to 30 rows. The UNION ALL operator does not eliminate duplicate rows. The duplicate rows are highlighted in the output shown in the slide. UNION returns all distinct rows selected by either query. UNION ALL returns all rows selected by either query, including all duplicates. Consider the query on the slide, now written with the UNION clause:

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

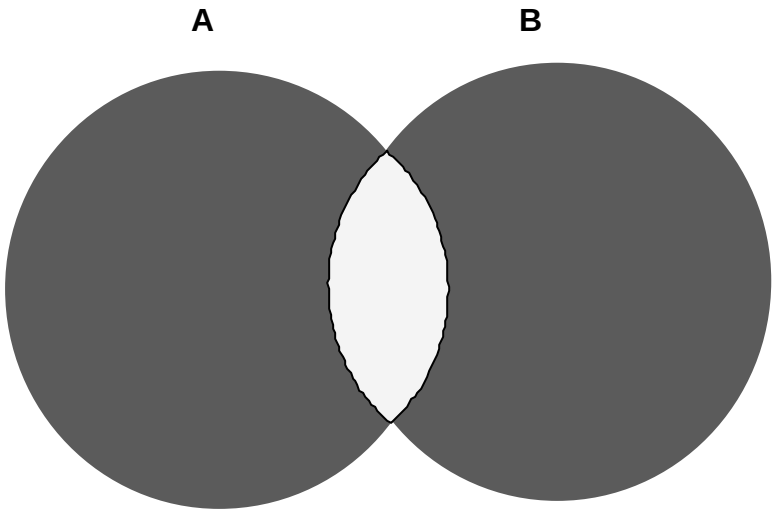
The preceding query returns 29 rows. This is because it eliminates the following row (as it is a duplicate):

EMPLOYEE_ID	JOB_ID	DEPARTMENT_ID
176	SA_REP	80

Instructor Note

Note that this is the example from page 15-8.

The INTERSECT Operator



ORACLE

The INTERSECT Operator

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

Guidelines

- The number of columns and the datatypes 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 need not be identical.
- Reversing the order of the intersected tables does not alter the result.
- INTERSECT does not ignore NULL values.

Instructor Note

To illustrate the INTERSECT SET operator, run the script demo\15_inters.sql.

Using the INTERSECT Operator

Display the employee IDs and job IDs of employees who currently have a job title that they held before beginning their tenure with the company.

```
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

ORACLE

The INTERSECT Operator (continued)

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

What will be the results if you add the DEPARTMENT_ID column to the SELECT statement from the EMPLOYEES table and add the DEPARTMENT_ID column to the SELECT statement from the JOB_HISTORY table and run this query? The results may be different because of the introduction of another column whose values may or may not be duplicates.

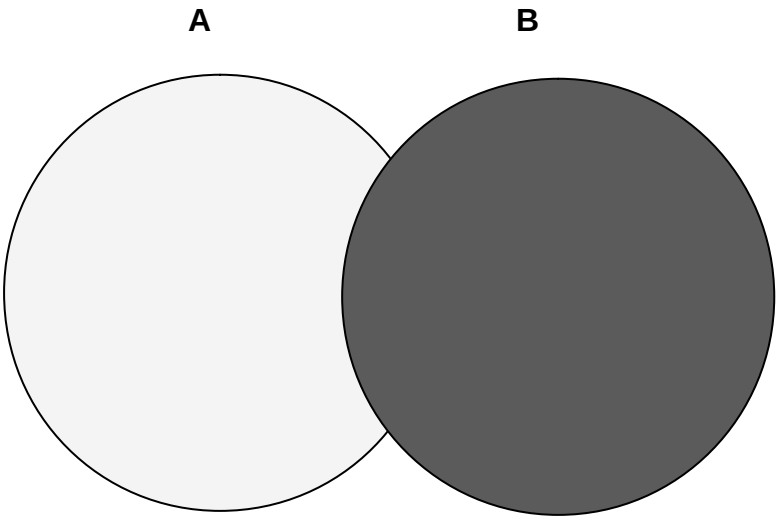
Example

```
SELECT employee_id, job_id, department_id
FROM   employees
INTERSECT
SELECT employee_id, job_id, department_id
FROM   job_history;
```

EMPLOYEE_ID	JOB_ID	DEPARTMENT_ID
176	SA_REP	80

Employee 200 is no longer part of the results because the EMPLOYEES.DEPARTMENT_ID value is different from the JOB_HISTORY.DEPARTMENT_ID value.

The MINUS Operator



The MINUS Operator

Use the MINUS operator to return rows returned by the first query that are not present in the second query (the first SELECT statement MINUS the second SELECT statement).

Guidelines

- The number of columns and the datatypes 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 need not be identical.
- All of the columns in the WHERE clause must be in the SELECT clause for the MINUS operator to work.

Instructor Note

To illustrate the MINUS operator, run the script demo\15_minus.sql.

The MINUS Operator

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

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

EMPLOYEE_ID	JOB_ID
100	AD_PRES
101	AD_VP
102	AD_VP
103	IT_PROG
...	
201	MK_MAN
202	MK_REP
205	AC_MGR
206	AC_ACCOUNT

18 rows selected.

ORACLE

The MINUS Operator (continued)

In the example in the slide, the employee IDs and Job IDs in the JOB_HISTORY 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 JOB_HISTORY table. These are the records of the employees who have not changed their jobs even once.

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

SET Operator Guidelines

- The expressions in the select lists of the queries must match in number and datatype. Queries that use UNION, UNION ALL, INTERSECT, and MINUS SET operators in their WHERE clause must have the same number and type of columns in their SELECT list. For example:

```
SELECT employee_id, department_id
FROM   employees
WHERE  (employee_id, department_id)
       IN (SELECT employee_id, department_id
           FROM   employees
           UNION
           SELECT employee_id, department_id
           FROM   job_history);
```

- The ORDER BY clause:
  - Can appear only at the very end of the statement
  - Will accept the column name, an alias, or the positional notation
- The column name or alias, if used in an ORDER BY clause, must be from the first SELECT list.
- SET operators can be used in subqueries.

Instructor Note

You might want to mention that the ORDER BY clause accepts the column name only if the column has the same name from both queries.

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`.**

ORACLE

The Oracle Server and SET Operators

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 datatype. If component queries select character data, the datatype of the return values are determined as follows:

- If both queries select values of datatype `CHAR`, the returned values have datatype `CHAR`.
- If either or both of the queries select values of datatype `VARCHAR2`, the returned values have datatype `VARCHAR2`.

Instructor Note

You might want to mention that the output is sorted in ascending order of the first column, then the second column, and so on, of the `SELECT` clause.

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.

ORACLE

Matching the SELECT Statements

As the expressions in the select lists of the queries must match in number, you can use dummy columns and the datatype conversion functions to comply with this rule. In the slide, the name `location` is given as the dummy column heading. The `TO_NUMBER` function is used in the first query to match the `NUMBER` datatype of the `LOCATION_ID` column retrieved by the second query. Similarly, the `TO_DATE` function in the second query is used to match the `DATE` datatype of the `HIRE_DATE` column retrieved by the first query.

Instructor Note

Demonstration: `demo\15_union3.sql`, `demo\15_dummy.sql`

Purpose: The demonstration `15_union3.sql` illustrates using conversion functions while matching columns in the two select lists. The demonstration `15_dummy.sql` uses dummy columns in order to match the select lists. For the `15_dummy.sql`, run the script, then uncomment the `REMARKS`, add `ORDER BY 2`, and rerun.

You might want to mention that the conversion functions in the code shown on the slide are not mandatory. The code will work fine even without the conversion functions, but it is recommended to explicitly convert values for performance benefits.

Matching the SELECT Statement

- **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.

ORACLE

Matching the SELECT Statement: Example

The EMPLOYEES and JOB_HISTORY 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 EMPLOYEE_ID, JOB_ID, and SALARY using the UNION operator, knowing that the salary exists only in the, EMPLOYEES table?

The code example in the slide matches the EMPLOYEE_ID and the JOB_ID columns in the EMPLOYEES and in the JOB_HISTORY tables. A literal value of 0 is added to the JOB_HISTORY SELECT statement to match the numeric SALARY column in the EMPLOYEES SELECT statement.

In the preceding results, each row in the output that corresponds to a record from the JOB_HISTORY table contains a 0 in the SALARY column.

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
FROM dual
UNION
SELECT 'the world to', 2
FROM dual
ORDER BY 2;
```

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

ORACLE

Controlling the Order of Rows

By default, the output is sorted in ascending order on the first column. You can use the ORDER BY clause to change this.

Using ORDER BY to Order Rows

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, an alias, or the positional notation. Without the ORDER BY clause, the code example in the slide produces the following output in the alphabetical order of the first column:

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

Note: Consider a compound query where the UNION SET operator is used more than once. In this case, the ORDER BY clause can use only positions rather than explicit expressions.

Instructor Note

To illustrate the ordering of rows with a SET operator, run the script demo\15_setord.sql. Briefly explain the COLUMN command with the NOPRINT option. Highlight the usage of the single quotes in the 'I''d like to teach' literal, in the second SELECT statement. You might want to mention that one can only use ORDER BY with a column, alias, or position of the column of the first query.

Summary

In this lesson, you should have learned how to:

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

ORACLE

Summary

- The `UNION` operator returns all rows selected by either 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 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 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 datatype.

Practice 15 Overview

This practice covers using the Oracle9*i* datetime functions.

Practice 15 Overview

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

1. List the department IDs for departments that do not contain the job ID ST_CLERK, using SET operators.

DEPARTMENT_ID
10
20
60
80
90
110
190

7 rows selected.

2. Display the country ID and the name of the countries that have no departments located in them, using SET operators.

CO	COUNTRY_NAME
DE	Germany

3. Produce a list of jobs for departments 10, 50, and 20, in that order. Display job ID and department ID, using SET operators.

JOB_ID	DEPARTMENT_ID
AD_ASST	10
ST_CLERK	50
ST_MAN	50
MK_MAN	20
MK_REP	20

4. List the employee IDs and job IDs of those employees who currently hold the job title that they held before beginning their tenure with the company.

EMPLOYEE_ID	JOB_ID
176	SA_REP
200	AD_ASST

Practice 2: (continued)

5. Write a compound query that lists the following:
   - Last names and department ID of all the employees from the `EMPLOYEES` table, regardless of whether or not they belong to any department or not
   - Department ID and department name of all the departments from the `DEPARTMENTS` table, regardless of whether or not they have employees working in them