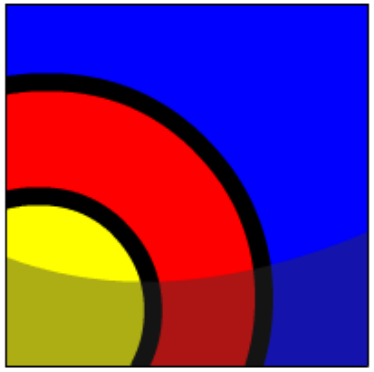
Set Operators

What Will I Learn?

In this lesson, you will learn to:



- Define and explain the purpose of Set Operators
- Use a set operator to combine multiple queries into a single query

Why Learn It?

Set operators are used to combine the results from different SELECT statements into one single result output.

Sometimes you want a single output from more than one table. If you join the tables, you only get returned the rows that match, but what if you don't want to do a join, or can't do a join because a join will give the wrong result?

This is where SET operators comes in. They can return the rows found in both statements, the rows that are in one table and not the other or the rows common to both statements

Tell Me / Show Me

In order to explain the SET operators the following two lists will be used throughout this lesson:

A = {1, 2, 3, 4, 5}

B = {4, 5, 6, 7, 8}

Or in reality: two tables, one called A and one called B.

A

A_ID
1
2
3
4
5

B

B_ID
4
5
6
7
8

Tell Me / Show Me

There are a few rules to remember when using SET operators:

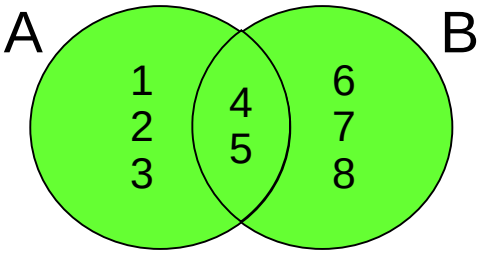
- The number of columns and the data types of the columns must be identical in all of the SELECT statements used in the query.
- The names of the columns need not be identical.
- Column names in the output are taken from the column names in the first SELECT statement. So any column aliases should be entered in the first statement as you would want to see them in the finished report.

Tell Me / Show Me

UNION

The UNION operator returns all rows from both tables, after eliminating duplicates.

```
SELECT a_id
FROM   a
UNION
SELECT b_id
FROM   b;
```



The result of listing all elements in A and B eliminating duplicates is {1, 2, 3, 4, 5, 6, 7, 8}.

If you joined A and B you would get only {4, 5}.

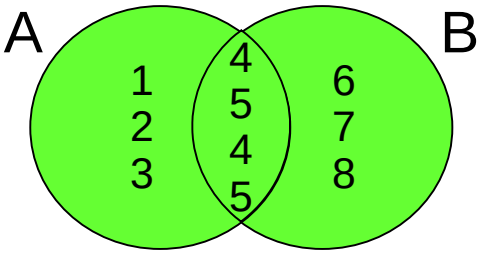
You would have to perform a full outer join to get the same list as above.

Tell Me / Show Me

UNION ALL

The UNION ALL operator returns all rows from both tables, without eliminating duplicates.

```
SELECT a_id
FROM   a
UNION  ALL
SELECT b_id
FROM   b;
```



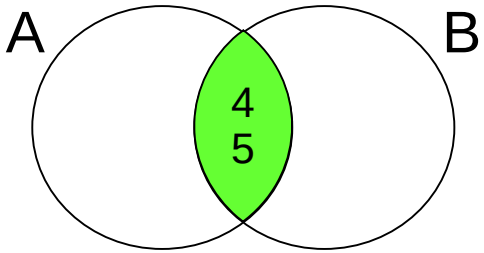
The result of listing all elements in A and B without eliminating duplicates is {1, 2, 3, 4, 5, 4, 5, 6, 7, 8}.

Tell Me / Show Me

INTERSECT

The INTERSECT operator returns all rows common to both tables.

```
SELECT a_id
FROM   a
INTERSECT
SELECT b_id
FROM   b;
```



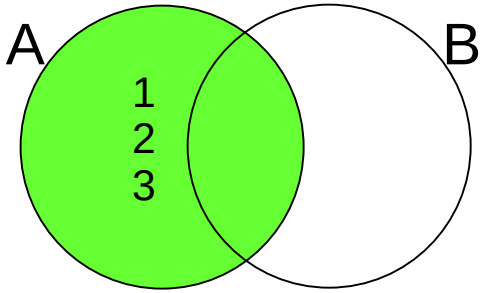
The result of listing all elements found in both A and B is {4, 5}.

Tell Me / Show Me

MINUS

The MINUS operator returns all rows found in one table but not the other.

```
SELECT a_id
FROM   a
MINUS
SELECT b_id
FROM   b;
```



The result of listing all elements found in A but not B is {1, 2, 3}, and B MINUS A would give {6, 7, 8}.

Tell Me / Show Me

SET OPERATOR EXAMPLES

Sometimes if you are selecting rows from tables that do not have columns in common, you may have to make up columns in order to match the queries. The easiest way to do this is to include one or more NULL values in the select list. Remember to give them suitable aliases and matching datatypes.

For example:

Table A contains a location id and a department name.

Table B contains a location id and a warehouse name.

You can use the TO_CHAR(NULL) function to fill in the missing columns as shown below.

```
SELECT location_id, department_name "Department", TO_CHAR(NULL) "Warehouse"
FROM departments
UNION
SELECT location_id, TO_CHAR(NULL) "Department", warehouse_name
FROM warehouses;
```

Tell Me / Show Me

SET OPERATOR EXAMPLES

The keyword NULL can be used to match columns in a SELECT list. One NULL is included for each missing column. Furthermore, NULL is formatted to match the datatype of the column it is standing in for, so TO_CHAR, TO_DATE or TO_NUMBER functions are often used to achieve identical SELECT lists.

Tell Me / Show Me

SET OPERATOR EXAMPLES

The WAREHOUSES table description:

Object Type **TABLE** Object **WAREHOUSES**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
WAREHOUSES	LOCATION_ID	Number	-	6	0	-	✓	-	-
	WAREHOUSE_NAME	Varchar2	50	-	-	-	✓	-	-

The WAREHOUSES table data:

LOCATION_ID	WAREHOUSE_NAME
1700	London
1800	Paris
2100	Copenhagen
4000	Shanghai
3000	Atlanta

Tell Me / Show Me

SET OPERATOR EXAMPLES

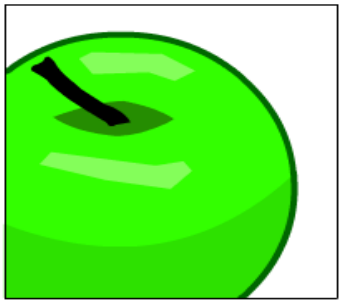
```
SELECT location_id, department_name "Department", TO_CHAR(NULL) "Warehouse"
FROM departments
UNION
SELECT location_id, TO_CHAR(NULL) "Department", warehouse_name
FROM warehouses;
```

LOCATION_ID	Department	Warehouse
1400	IT	-
1500	Shipping	-
1700	Accounting	-
1700	Administration	-
1700	Contracting	-
1700	Executive	-
1700	-	London
1800	Marketing	-
1800	-	Paris
2100	-	Copenhagen
2500	Sales	-
3000	-	Atlanta
4000	-	Shanghai

Tell Me / Show Me

Terminology

Key terms used in this lesson include:



SET operators

UNION

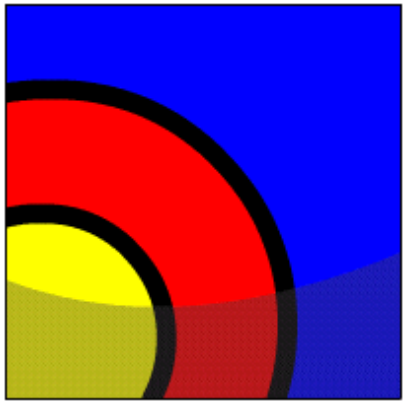
UNION ALL

INTERSECT

MINUS

TO_CHAR(null) – matching the select list

Summary



In this lesson you have learned to:

- Define and explain the purpose of Set Operators
- Use a set operator to combine multiple queries into a single query

Summary

Practice Guide

The link for the lesson practice guide can be found in the course resources in Section 0.