

8

Going Beyond Joins with Set Operations: UNION, INTERSECT, EXCEPT

We now understand how to retrieve information from multiple related tables using join operations. This chapter goes beyond joins by showing you how to use operators that combine the result set of two or more queries.

CHAPTER OBJECTIVES

The objectives for this chapter are as follows:

- Understand how to use the UNION operator.
- Understand how to use UNION ALL.
- Understand how to use the INTERSECT operator.
- Understand how to use the EXCEPT operator.

UNION Operator

The UNION operator is used to combine the results of two or more queries into a single result set that contain distinct values. By default, duplicate rows are removed. For the UNION operator to work the data types must be similar and the columns in each SELECT statement must be in the same order.

This chapter introduces a new table called the Volunteers table. The Volunteers table contains employees as well as nonemployees. Therefore, some names in the Employees table can exist in the Volunteers table and some names in the Volunteers table WILL NOT exist in the Employees table. Let's start by looking at the design of the Employees table and Volunteers table below.

Employees Table			Volunteers Table		
Column Name	Data Type	Allow Nulls	Column Name	Data Type	Allow Nulls
EmployeeID	int	<input type="checkbox"/>	VolunteerID	int	<input type="checkbox"/>
FirstName	varchar(50)	<input checked="" type="checkbox"/>	First	varchar(35)	<input type="checkbox"/>
MiddleName	varchar(50)	<input checked="" type="checkbox"/>	Last	varchar(35)	<input type="checkbox"/>
LastName	varchar(50)	<input checked="" type="checkbox"/>	MI	char(1)	<input checked="" type="checkbox"/>
Title	varchar(50)	<input type="checkbox"/>	Sex	char(1)	<input type="checkbox"/>
BirthDate	date	<input type="checkbox"/>	DOB	date	<input type="checkbox"/>
MaritalStatus	char(1)	<input type="checkbox"/>			
Gender	char(1)	<input type="checkbox"/>			
HireDate	date	<input type="checkbox"/>			
Salary	money	<input checked="" type="checkbox"/>			
VacationHours	smallint	<input type="checkbox"/>			
SickLeaveHours	smallint	<input type="checkbox"/>			
Active	char(3)	<input type="checkbox"/>			
ManagerID	int	<input checked="" type="checkbox"/>			
ModifiedDate	datetime	<input type="checkbox"/>			

Figure 1

Notice from the Employees table design and the Volunteers table design that the column names for first name, last name, middle initial, gender, and birth date are different in each table. In addition, the size of the columns is different. When we perform our UNION query this won't matter if the data types are similar as mentioned earlier.

Another important item to be aware of is the different data types on middle name. The Employees table has a middle name of Varchar (50) while the Volunteers table has a data type of char (1). This is fine since both are of text/character nature. **However, be aware that if one of the middle names was a numeric data type that would cause a problem.**

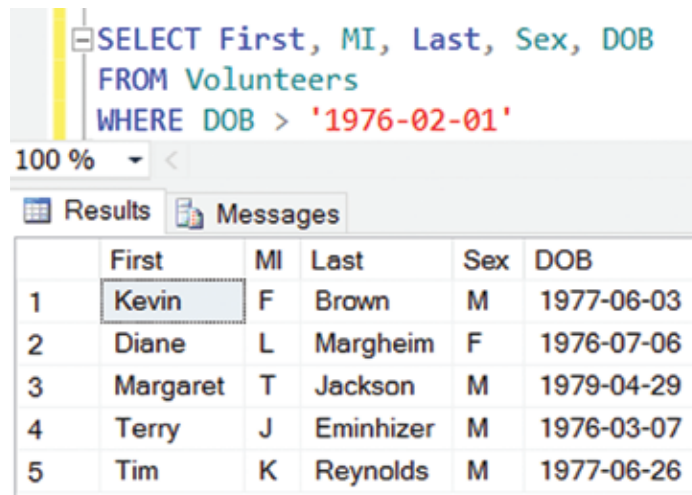
Before using the UNION operator let's look at the rows of data in each table. I am going to limit the rows of data returned from each table by using a WHERE clause to retrieve rows with birth dates greater than 02/01/1976. Figure 2 shows the SELECT statement and the resulting rows.

```
SELECT FirstName, MiddleName, LastName, Gender, BirthDate
FROM Employees
WHERE BirthDate > '1976-02-01'
```

	FirstName	MiddleName	LastName	Gender	BirthDate
1	Kevin	F	Brown	M	1977-06-03
2	Seriya	E	Harnpadoungsataya	M	1977-06-21
3	Tengiz	N	Kheretishvili	M	1980-05-29
4	Diane	L	Margheim	F	1976-07-06
5	Mark	L	Harrington	M	1976-05-31
6	Sairaj	L	Uddin	M	1978-01-22
7	Andreas	T	Berglund	M	1979-04-29
8	Ramesh	V	Meyyeppen	M	1978-04-14
9	Dylan	A	Miller	M	1977-03-27
10	Terry	J	Eminhizer	M	1976-03-07
11	Chris	K	Norred	M	1977-06-26
12	Janice	M	Galvin	F	1979-06-29

Figure 2

Now let's issue the same query against the Volunteers table. The result set follows in Figure 3.



The screenshot shows a SQL query window with the following text:

```
SELECT First, MI, Last, Sex, DOB
FROM Volunteers
WHERE DOB > '1976-02-01'
```

Below the query, the 'Results' tab is active, displaying a table with 5 rows and 6 columns: First, MI, Last, Sex, and DOB. The first row is highlighted.

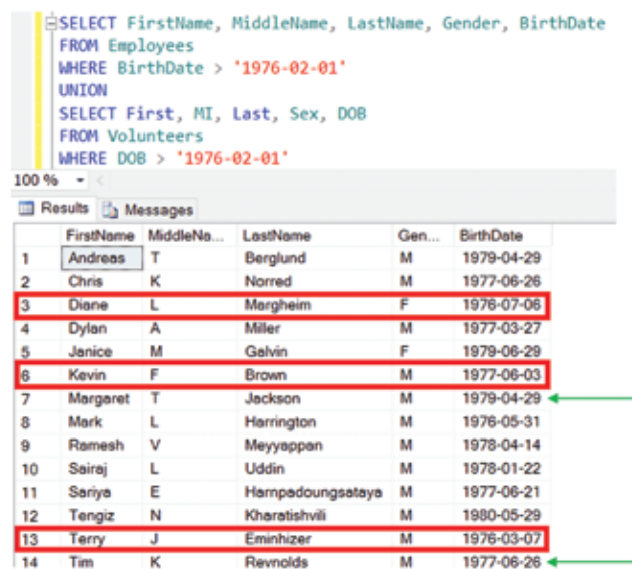
	First	MI	Last	Sex	DOB
1	Kevin	F	Brown	M	1977-06-03
2	Diane	L	Margheim	F	1976-07-06
3	Margaret	T	Jackson	M	1979-04-29
4	Terry	J	Eminhizer	M	1976-03-07
5	Tim	K	Reynolds	M	1977-06-26

Figure 3

We can see from looking at both result sets that Kevin Brown, Diane Margheim, and Terry Eminhizer are in both tables. If we were to apply both queries to a UNION operator as shown below the results would not include those names twice because the UNION operator, as mentioned earlier, automatically removes duplicates.

The following query will produce the result set shown in Figure 4.

```
SELECT FirstName, MiddleName, LastName, Gender, BirthDate
FROM Employees
WHERE BirthDate > '1976-02-01'
UNION
SELECT First, MI, Last, Sex, DOB
FROM Volunteers
WHERE DOB > '1976-02-01'
```



The screenshot shows a SQL query window with the following text:

```
SELECT FirstName, MiddleName, LastName, Gender, BirthDate
FROM Employees
WHERE BirthDate > '1976-02-01'
UNION
SELECT First, MI, Last, Sex, DOB
FROM Volunteers
WHERE DOB > '1976-02-01'
```

Below the query, the 'Results' tab is active, displaying a table with 14 rows and 6 columns: FirstName, MiddleName, LastName, Gender, and BirthDate. The results are sorted by BirthDate. Red boxes highlight rows 3, 4, 5, 6, 13, and 14, which correspond to the data in Figure 3. Green arrows point to rows 7 and 14.

	FirstName	MiddleName	LastName	Gen...	BirthDate
1	Andreas	T	Berglund	M	1979-04-29
2	Chris	K	Norred	M	1977-06-26
3	Diane	L	Margheim	F	1976-07-06
4	Dylan	A	Miller	M	1977-03-27
5	Janice	M	Galvin	F	1979-06-29
6	Kevin	F	Brown	M	1977-06-03
7	Margaret	T	Jackson	M	1979-04-29
8	Mark	L	Harrington	M	1976-05-31
9	Ramesh	V	Meyyappan	M	1978-04-14
10	Seiraj	L	Uddin	M	1978-01-22
11	Sariya	E	Hampadounsatsaya	M	1977-06-21
12	Tengiz	N	Kharatishvili	M	1980-05-29
13	Terry	J	Eminhizer	M	1976-03-07
14	Tim	K	Reynolds	M	1977-06-26

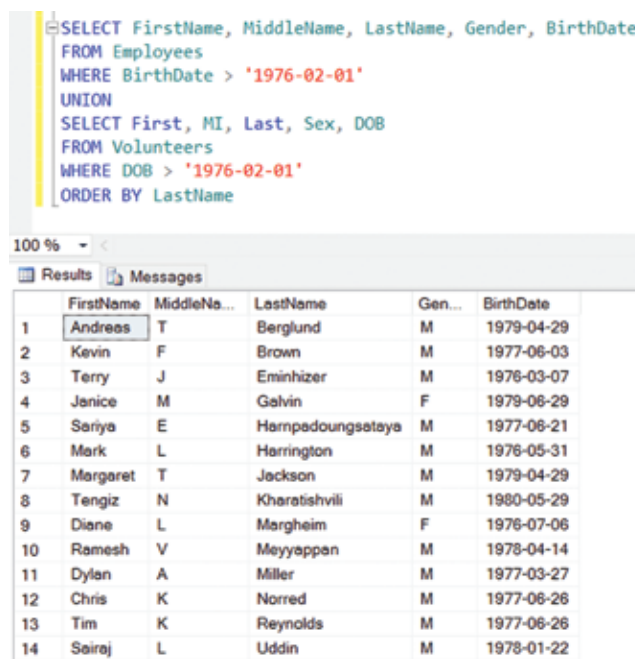
Figure 4

Notice that the result set has 14 rows. This is because it included the 12 rows from the Employees table and the 2 rows from the Volunteers table, Margaret Jackson and Tim Reynolds. As you can see, duplicates were not included.

Another important point to make is that the column names in the result set are the ones from the Employee table. The column names will always be the result of the first SELECT statement in the UNION query. So, if you use aliases for column names make sure you apply them to the first SELECT statement.

ORDER BY

It's important to learn how the ORDER BY clause works in the UNION query as it is a bit confusing. The ORDER BY is placed at the very end of the UNION query using column names associated with the first SELECT statement. See the example below in Figure 5.



```

SELECT FirstName, MiddleName, LastName, Gender, BirthDate
FROM Employees
WHERE BirthDate > '1976-02-01'
UNION
SELECT First, MI, Last, Sex, DOB
FROM Volunteers
WHERE DOB > '1976-02-01'
ORDER BY LastName

```

	FirstName	MiddleName	LastName	Gen...	BirthDate
1	Andreas	T	Berglund	M	1979-04-29
2	Kevin	F	Brown	M	1977-06-03
3	Terry	J	Eminhizer	M	1976-03-07
4	Janice	M	Galvin	F	1979-06-29
5	Sariya	E	Hampadounsataya	M	1977-06-21
6	Mark	L	Harrington	M	1976-05-31
7	Margaret	T	Jackson	M	1979-04-29
8	Tengiz	N	Kharatishvili	M	1980-05-29
9	Diane	L	Margheim	F	1976-07-06
10	Ramesh	V	Meyyappen	M	1978-04-14
11	Dylan	A	Miller	M	1977-03-27
12	Chris	K	Norred	M	1977-06-26
13	Tim	K	Reynolds	M	1977-06-26
14	Sairej	L	Uddin	M	1978-01-22

Figure 5

If you tried to use a column from anything other than the first SELECT statement an error would occur as can be seen when trying to order by the column name "Last" from the Volunteers table in Figure 6.



```

SELECT FirstName, MiddleName, LastName, Gender, BirthDate
FROM Employees
WHERE BirthDate > '1976-02-01'
UNION
SELECT First, MI, Last, Sex, DOB
FROM Volunteers
WHERE DOB > '1976-02-01'
ORDER BY Last

```

100 %

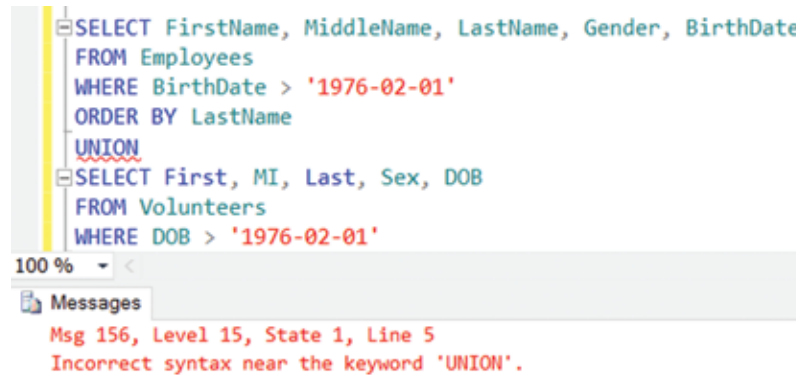
Messages

Msg 207, Level 16, State 1, Line 8
Invalid column name 'Last'.

Msg 104, Level 16, State 1, Line 8
ORDER BY items must appear in the select list if the statement contains a UNION, INTERSECT or EXCEPT operator.

Figure 6

An error would occur, also, if you tried to use the ORDER BY right after the first SELECT statement even if the column name is associated with the first SELECT statement. Figure 7 demonstrates this.



```

SELECT FirstName, MiddleName, LastName, Gender, BirthDate
FROM Employees
WHERE BirthDate > '1976-02-01'
ORDER BY LastName
UNION
SELECT First, MI, Last, Sex, DOB
FROM Volunteers
WHERE DOB > '1976-02-01'

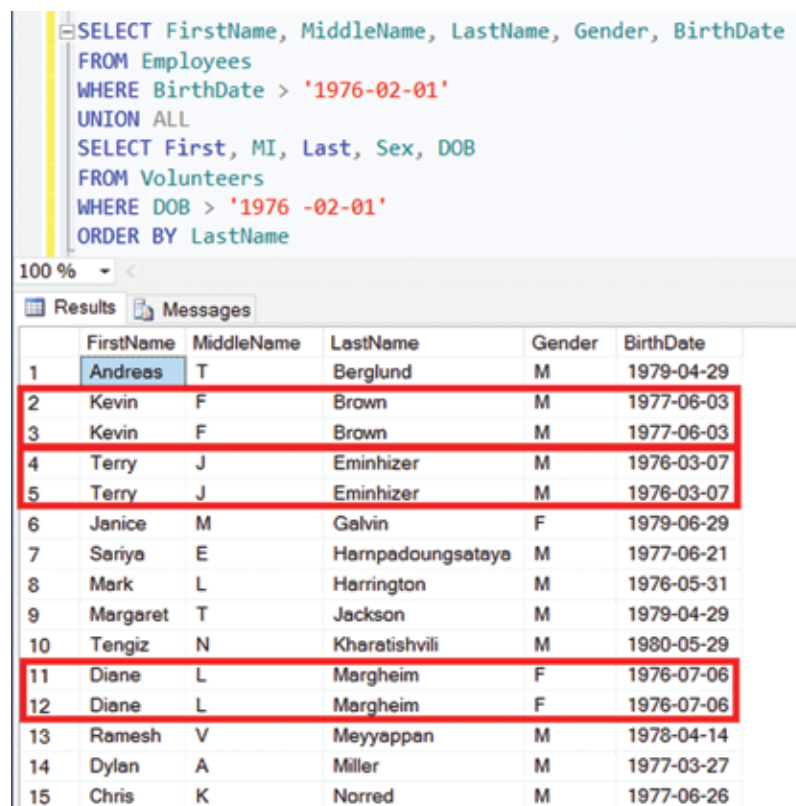
```

Msg 156, Level 15, State 1, Line 5
Incorrect syntax near the keyword 'UNION'.

Figure 7

UNION ALL

The UNION ALL operator works the same way as the UNION operator except that **it includes duplicates**. Let's execute the same query as we did with the UNION operator except this time we use the UNION ALL operator instead. Let's sort by LastName so that we can see the duplicates next to each other. See Figure 8.



```

SELECT FirstName, MiddleName, LastName, Gender, BirthDate
FROM Employees
WHERE BirthDate > '1976-02-01'
UNION ALL
SELECT First, MI, Last, Sex, DOB
FROM Volunteers
WHERE DOB > '1976-02-01'
ORDER BY LastName

```

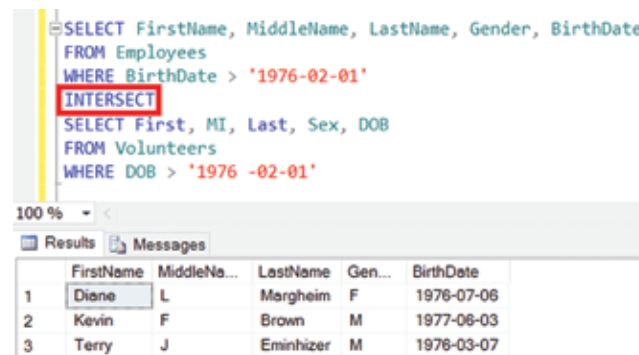
	FirstName	MiddleName	LastName	Gender	BirthDate
1	Andreas	T	Berglund	M	1979-04-29
2	Kevin	F	Brown	M	1977-06-03
3	Kevin	F	Brown	M	1977-06-03
4	Terry	J	Eminhizer	M	1976-03-07
5	Terry	J	Eminhizer	M	1976-03-07
6	Janice	M	Galvin	F	1979-06-29
7	Seriya	E	Harnpadoungsataya	M	1977-06-21
8	Mark	L	Harrington	M	1976-05-31
9	Margaret	T	Jackson	M	1979-04-29
10	Tengiz	N	Kharatishvili	M	1980-05-29
11	Diane	L	Margheim	F	1976-07-06
12	Diane	L	Margheim	F	1976-07-06
13	Ramesh	V	Meyyappan	M	1978-04-14
14	Dylan	A	Miller	M	1977-03-27
15	Chris	K	Norred	M	1977-06-26

Figure 8

The result set contains 17 rows, three more than using the UNION operator without the ALL. That's because there are three duplicates, Kevin Brown, Terry Eminhizer, and Diane Margheim.

INTERSECT Operator

The INTERSECT operator is used to retrieve the common rows from two or more SELECT results. If we use the same query that we used with the UNION operator but replace UNION with INTERSECT we will end up with the results shown in Figure 9. As you can see in Figure 9, Diane, Kevin, and Terry are shown in the result set since they appear in both the Employees and Volunteers tables.



```

SELECT FirstName, MiddleName, LastName, Gender, BirthDate
FROM Employees
WHERE BirthDate > '1976-02-01'
INTERSECT
SELECT First, MI, Last, Sex, DOB
FROM Volunteers
WHERE DOB > '1976 -02-01'

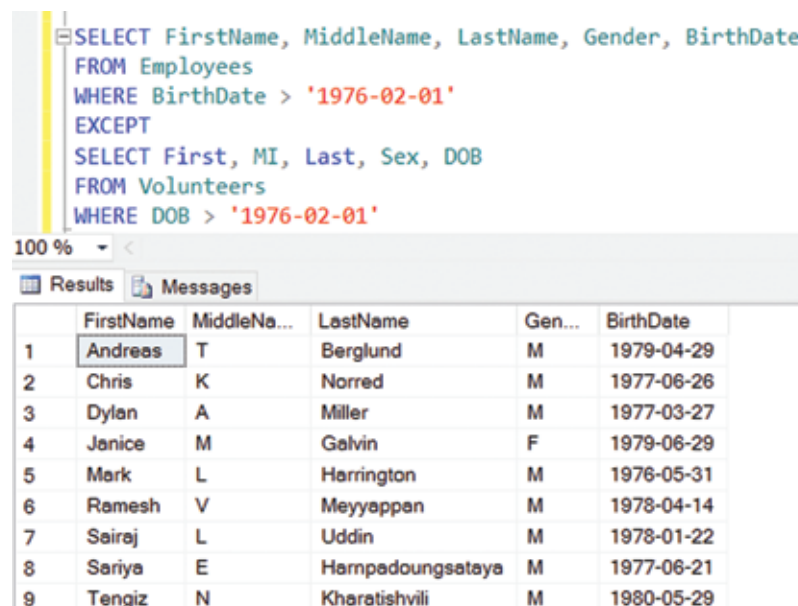
```

	FirstName	MiddleName...	LastName	Gen...	BirthDate
1	Diane	L	Margheim	F	1976-07-06
2	Kevin	F	Brown	M	1977-06-03
3	Terry	J	Eminhizer	M	1976-03-07

Figure 9

EXCEPT Operator

The EXCEPT operator is used to return all rows returned in the first query that are not found in the second query. For example, let's issue the same query we've been working with but with the EXCEPT operator. The result set is shown in Figure 10.



```

SELECT FirstName, MiddleName, LastName, Gender, BirthDate
FROM Employees
WHERE BirthDate > '1976-02-01'
EXCEPT
SELECT First, MI, Last, Sex, DOB
FROM Volunteers
WHERE DOB > '1976-02-01'

```

	FirstName	MiddleName...	LastName	Gen...	BirthDate
1	Andreas	T	Berglund	M	1979-04-29
2	Chris	K	Norred	M	1977-06-26
3	Dylan	A	Miller	M	1977-03-27
4	Janice	M	Galvin	F	1979-06-29
5	Merk	L	Harrington	M	1976-05-31
6	Ramesh	V	Meyyappen	M	1978-04-14
7	Sairej	L	Uddin	M	1978-01-22
8	Sariya	E	Harnpadoungsataya	M	1977-06-21
9	Tengiz	N	Kharatishvili	M	1980-05-29

Figure 10

The results in Figure 9 cannot be found in the Volunteer table as shown earlier in Figure 2. Let's try the reverse and make the query with the Volunteers table the first query and the one with the Employees table the second query. The results are shown in Figure 11.

```

SELECT First, MI, Last, Sex, DOB
FROM Volunteers
WHERE DOB > '1976-02-01'
EXCEPT
SELECT FirstName, MiddleName, LastName, Gender, BirthDate
FROM Employees
WHERE BirthDate > '1976-02-01'

```

	First	MI	Last	Sex	DOB
1	Margaret	T	Jackson	M	1979-04-29
2	Tim	K	Reynolds	M	1977-06-26

Figure 11

The results in Figure 11 cannot be found in the Employees table so we can see that the EXCEPT operator lists everything not found in the second query.

Use of UNION Operator

Let's discuss some examples where the UNION operator could be valuable. Let's say the manager would like to see a list of employees and the gender spelled out as part of the results. Figure 12 shows us the SQL statement and a subset of the results.

```

SELECT FirstName, MiddleName, LastName, Gender, 'FEMALE' Type
FROM Employees
WHERE Gender = 'F'
UNION
SELECT FirstName, MiddleName, LastName, Gender, 'MALE'
FROM Employees
WHERE Gender = 'M'
ORDER BY Gender

```

	FirstName	MiddleName	LastName	Gender	Type
10	Gigi	N	Matthew	F	FEMALE
11	Gladys	L	Hee	F	FEMALE
12	Jee	B	Pak	F	FEMALE
13	Janaina	Barreiro Gambaro	Bueno	F	FEMALE
14	Janet	L	Smith	F	FEMALE
15	Janice	M	Galvin	F	FEMALE
16	Jean	E	O'Brien	F	FEMALE
17	Jill	A	Williams	F	FEMALE
18	Jillion	NULL	Carson	F	FEMALE
19	Jo	L	Berry	F	FEMALE
20	Karen	A	Berg	F	FEMALE
21	Kim	T	Rolls	F	FEMALE
22	Laura	F	Norman	F	FEMALE
23	Linda	C	Mitchell	F	FEMALE
24	Linda	P	Meisner	F	FEMALE
25	Lori	K	Penor	F	FEMALE
26	Lynn	N	Tenflies	F	FEMALE

Figure 12

For a more difficult example, let's say a manager wants to hand out bonuses but he wants different percentages paid to employees based on their salaries. The table below shows the percentages he wants to use on the ranges of salaries.

Salary Range	Percent
Greater than 150,000	10%
Between 100,000 and 149,000	15%
Between 50,000 and 99,999	20%
Less than 50,000	25%

To retrieve this information a UNION operator would be of great value. The query is shown below with a subset of the results shown in Figure 13.

```
SELECT FirstName, MiddleName, LastName, Title, Salary, '10% BONUS' Percentage,
Salary * .10 BONUS
FROM Employees
WHERE Salary > 150000
UNION
SELECT FirstName, MiddleName, LastName, Title, Salary, '15% BONUS', Salary * .15
BONUS
FROM Employees
WHERE Salary BETWEEN 100000 AND 149999
UNION
SELECT FirstName, MiddleName, LastName, Title, Salary, '20% BONUS', Salary * .20
BONUS
FROM Employees
WHERE Salary BETWEEN 50000 AND 99999
UNION
SELECT FirstName, MiddleName, LastName, Title, Salary, '25% BONUS', Salary * .25
BONUS
FROM Employees
WHERE Salary < 50000
ORDER BY Percentage
```

The results are shown in Figure 13. You can see the results successfully display each employee's first, middle and last name, along with their title, salary, the percentage of the bonus that is applied, and the resulting dollar amount for the bonus.


```

SELECT FirstName, MiddleName, LastName, Title, Salary, '10% BONUS' Percentage, Salary * .10 BONUS
FROM Employees
WHERE Salary > 150000
UNION
SELECT FirstName, MiddleName, LastName, Title, Salary, '15% BONUS', Salary * .15 BONUS
FROM Employees
WHERE Salary BETWEEN 100000 AND 149999
UNION
SELECT FirstName, MiddleName, LastName, Title, Salary, '20% BONUS', Salary * .20 BONUS
FROM Employees
WHERE Salary BETWEEN 50000 AND 99999
UNION
SELECT FirstName, MiddleName, LastName, Title, Salary, '25% BONUS', Salary * .25 BONUS
FROM Employees
WHERE Salary < 50000
ORDER BY Percentage

```

	FirstName	MiddleName	LastName	Title	Salary	Percentage	BONUS
1	James	R	Hamilton	Vice President of Production	186382.56	10% BONUS	18638.256000
2	Amy	E	Alberts	European Sales Manager	113749.98	15% BONUS	17062.497000
3	Brian	S	Welcker	Vice President of Sales	107100.02	15% BONUS	16065.003000
4	Diane	L	Margheim	Research and Development Engineer	102000.04	15% BONUS	15300.006000
5	Dylan	A	Miller	Research and Development Manager	107100.06	15% BONUS	16065.009000
6	Gigi	N	Matthew	Research and Development Engineer	102000.04	15% BONUS	15300.006000
7	Jean	E	O'Brien	Information Services Manager	126000.07	15% BONUS	18900.010500
8	Matthias	T	Berndt	Shipping and Receiving Clerk	105000.06	15% BONUS	15750.009000
9	Wendy	Beth	Kahn	Finance Manager	107999.93	15% BONUS	16199.989500
10	Barbara	C	Moreland	Accountant	65999.98	20% BONUS	13199.996000
11	David	J	Liu	Accounts Manager	73695.05	20% BONUS	14739.010000
12	Erin	M	Hagens	Buyer	56999.91	20% BONUS	11399.982000
13	François	P	Ajenstat	Database Administrator	57119.94	20% BONUS	11423.988000
14	Fukiko	J	Ogisu	Buyer	54263.91	20% BONUS	10852.782000
15	Gail	A	Erickson	Design Engineer	66299.98	20% BONUS	13259.996000

Figure 13

Summary

This chapter introduced you to set operations to show you how to provide more power to your queries. The UNION operator was covered which allows you to further group data distinctly, whereas the UNION ALL operator is used to include duplicates. You were also introduced to the INTERSECT and EXCEPT operators. You saw how the INTERSECT operator can be used to return common rows among different tables and how the EXCEPT operator can be used to return all rows returned in the first query that are not found in the second query.

Exercises

1. Use the UNION operator to find distinct names from the Members and Volunteers table.
2. Use the UNION operator to find distinct names from the Members and Employee table.
3. Use the UNION ALL operator to find all names from the Members and Volunteers table and ORDER BY Gender.
4. Use the INTERSECT operator to find common rows among the Members and Volunteers tables.
5. Use the INTERSECT operator to find common rows among the Members and Employee tables.
6. Use the EXCEPT operator to find rows in the Members table but not in the Volunteers table.
7. Use the EXCEPT operator to find rows in the Volunteers table but not in the Members table.
8. Use the UNION operator to provide a status on salaries of "High" for anything greater than \$100,000, "Low" for anything less than \$50,000, and "Medium" for anything in between.
9. Use the UNION operator to assign the following categories to Group Name in the Departments table.
 - Executive General and Administration = "Top Management"
 - Inventory Management = "Middle Management"
 - Sales and Marketing = "Middle Management"
 - Research and Development = "Middle Management"
 - Quality Assurance = "Specialists"
 - Manufacturing = "Laborers"
 - Call the column name = "Category"
10. Use the Employee table to display first name, last name, and number of vacation hours and order by vacation hours. Apply the following category name to vacation hours.
 - Greater than 75 = Excessive
 - Less than 25 = Low
 - Everything else = Standard
11. Use the UNION operator to group employees based on the generation, in which they were born in. Categorize the generations according to the following:
 - 1925–1942 = "Silent Generation"
 - 1943–1960 = "Baby Boomers"
 - 1961–1984 = "Generation X"
 - 1985–2000 = "Millennials"
 - Label the column "Generations"
12. Use the UNION operator to label employees with the states they come from spelled out. Do this for Washington, Massachusetts, and California. For other states, list "Other". (Note: This will require a join between the Employee and Address tables.)
13. Use the INTERSECT operator to find any gym members that are also employees.
14. Use the UNION operator to display employee dependents' full names and replace NULL middle names with "NONE." Use the UNION operator to do the replacement.
15. Display all employee's names and their marital status. Using the UNION operator, replace the "M" with "Married" and the "S" with "Single."