
Comparing Multiple Datasets with the INTERSECT and EXCEPT operators

Written By: Edgewood Solutions Engineers -- 9/11/2007

Sponsor



Problem

When joining multiple datasets you have always had the ability to use the UNION and UNION ALL operator to allow you to pull a distinct result set (union) or a complete result set (union all). These are very helpful commands when you need to pull data from different tables and show the results as one unified distinct result set. On the opposite side of this it would be helpful to only show a result set where both sets of data match or only where data exists in one of the tables and not the other. This could be done with using different join types, but what other options does SQL Server offer?

Solution

With SQL Server 2005, Microsoft introduced the INTERSECT and EXCEPT operators to further extend what you could already do with the UNION and UNION ALL operators.

- INTERSECT - gives you the final result set where values in both of the tables match
- EXCEPT - gives you the final result set where data exists in the first dataset and not in the second dataset

The advantage of these commands is that it allows you to get a distinct listing across all of the columns such as the UNION and UNION ALL operators do without having to do a group by or do a comparison of every single column.

Like the UNION and UNION ALL operators the table structures need to be consistent as well as the columns need to have compatible data types.

Let's take for example we have two tables manager and customer. Both of these tables have somewhat the same structure such as the following columns:

- FirstName
- LastName
- AddressLine1
- City
- StateProvinceCode
- PostalCode

[Manager table sample data](#)

	FirstName	LastName	AddressLine1	City	StateProvinceCode	PostalCode
1	Ben	Miller	101 Candy Rd.	Redmond	WA	98052
2	Garrett	Vargas	10203 Acorn Avenue	Calgary	AB	T2P 2G8
3	Gabe	Mares	1061 Buskrik Avenue	Edmonds	WA	98020
4	Reuben	D'sa	1064 Slow Creek Road	Seattle	WA	98104
5	Gordon	Hee	108 Lakeside Court	Bellevue	WA	98004
6	Karan	Khanna	1102 Ravenwood	Seattle	WA	98104
7	François	Ajenstat	1144 Paradise Ct.	Issaquah	WA	98027
8	Sariya	Harnpadoungsataya	1185 Dallas Drive	Everett	WA	98201
9	Kirk	Koenigsbauer	1220 Bradford Way	Seattle	WA	98104
10	Kim	Ralls	1226 Shoe St.	Bothell	WA	98011
11	Michael	Raheem	1234 Seaside Way	San Francisco	CA	94109
12	Mike	Seamans	1245 Clay Road	Index	WA	98256
13	Reed	Koch	1275 West Street	Redmond	WA	98052
14	Fadi	Fakhouri	1285 Greenbrier Street	Snohomish	WA	98296
15	Paul	Singh	1343 Prospect St	Bellevue	WA	98004
16	Brenda	Diaz	1349 Steven Way	Seattle	WA	98104
17	Jack	Richins	1356 Grove Way	Monroe	WA	98272
18	John	Evans	136 Balboa Court	Seattle	WA	98104
19	Ken	Myer	1362 Somerset Place	Everett	WA	98201
20	Barbara	Moreland	137 Mazatlan	Seattle	WA	98104

Customer table sample data

	FirstName	LastName	AddressLine1	City	StateProvinceCode	PostalCode
1	Ben	Miller	101 Candy Rd.	Redmond	WA	98052
2	Garrett	Vargas	10203 Acorn Avenue	Calgary	AB	T2P 2G8
3	Gabe	Mares	1061 Buskrik Avenue	Edmonds	WA	98020
4	Reuben	D'sa	1064 Slow Creek Road	Seattle	WA	98104
5	Gordon	Hee	108 Lakeside Court	Bellevue	WA	98004
6	Karan	Khanna	1102 Ravenwood	Seattle	WA	98104
7	François	Ajenstat	1144 Paradise Ct.	Issaquah	WA	98027
8	Sariya	Harnpadoungsataya	1185 Dallas Drive	Everett	WA	98201
9	Kirk	Koenigsbauer	1220 Bradford Way	Seattle	WA	98104
10	Kim	Ralls	1226 Shoe St.	Bothell	WA	98011

We want to do two queries:

1. Find the occurrences where a manager is a customer (intersect)
2. Find the occurrences where the manager is not a customer (except)

INTERSECT

If we want to find out which people exist in both the customer table and the manager table and get a distinct list back we can issue the following command:

```
SELECT FIRSTNAME ,
        LASTNAME ,
        ADDRESSLINE1 ,
        CITY ,
        STATEPROVINCECODE ,
        POSTALCODE
FROM    MANAGER
```

```

INTERSECT
SELECT FIRSTNAME ,
       LASTNAME ,
       ADDRESSLINE1 ,
       CITY ,
       STATEPROVINCECODE ,
       POSTALCODE
FROM   CUSTOMER

```

Here is the result set:

	FirstName	LastName	AddressLine1	City	StateProvinceCode	PostalCode
1	Ben	Miller	101 Candy Rd.	Redmond	WA	98052
2	François	Ajenstat	1144 Paradise Ct.	Issaquah	WA	98027
3	Gabe	Mares	1061 Buskrik Avenue	Edmonds	WA	98020
4	Garrett	Vargas	10203 Acorn Avenue	Calgary	AB	T2P 2G8
5	Gordon	Hee	108 Lakeside Court	Bellevue	WA	98004
6	Karan	Khanna	1102 Ravenwood	Seattle	WA	98104
7	Kim	Ralls	1226 Shoe St.	Bothell	WA	98011
8	Kirk	Koenigsbauer	1220 Bradford Way	Seattle	WA	98104
9	Reuben	D'sa	1064 Slow Creek Road	Seattle	WA	98104
10	Sariya	Harnpadoungsataya	1185 Dallas Drive	Everett	WA	98201

To do this same thing with a regular T-SQL command we would have to write the following:

```

SELECT  M.FIRSTNAME ,
        M.LASTNAME ,
        M.ADDRESSLINE1 ,
        M.CITY ,
        M.STATEPROVINCECODE ,
        M.POSTALCODE
FROM    MANAGER M
        INNER JOIN CUSTOMER C
          ON M.FIRSTNAME = C.FIRSTNAME
          AND M.LASTNAME = C.LASTNAME
          AND M.ADDRESSLINE1 = C.ADDRESSLINE1
          AND M.CITY = C.CITY
          AND M.POSTALCODE = C.POSTALCODE
GROUP BY M.FIRSTNAME , M.LASTNAME , M.ADDRESSLINE1 , M.CITY ,
        M.STATEPROVINCECODE , M.POSTALCODE

```

EXCEPT

If we want to find out which people exists in the manager table, but not in the customer table and get a distinct list back we can issue the following command:

```

SELECT FIRSTNAME ,
       LASTNAME ,
       ADDRESSLINE1 ,
       CITY ,
       STATEPROVINCECODE ,
       POSTALCODE
FROM   MANAGER
EXCEPT
SELECT FIRSTNAME ,
       LASTNAME ,
       ADDRESSLINE1 ,
       CITY ,
       STATEPROVINCECODE ,
       POSTALCODE

```

```
FROM CUSTOMER
```

Here is the result set:

	FirstName	LastName	AddressLine1	City	StateProvinceCode	PostalCode
1	Barbara	Moreland	137 Mazatlan	Seattle	WA	98104
2	Brenda	Diaz	1349 Steven Way	Seattle	WA	98104
3	Fadi	Fakhouri	1285 Greenbrier Street	Snohomish	WA	98296
4	Jack	Richins	1356 Grove Way	Monroe	WA	98272
5	John	Evans	136 Balboa Court	Seattle	WA	98104
6	Ken	Myer	1362 Somerset Place	Everett	WA	98201
7	Michael	Raheem	1234 Seaside Way	San Francisco	CA	94109
8	Mike	Seamans	1245 Clay Road	Index	WA	98256
9	Paul	Singh	1343 Prospect St	Bellevue	WA	98004
10	Reed	Koch	1275 West Street	Redmond	WA	98052

To do this same thing with a regular T-SQL command we would have to write the following:

```
SELECT M.FIRSTNAME ,
       M.LASTNAME ,
       M.ADDRESSLINE1 ,
       M.CITY ,
       M.STATEPROVINCECODE ,
       M.POSTALCODE
FROM   MANAGER M
WHERE  NOT EXISTS ( SELECT *
                    FROM   CUSTOMER C
                    WHERE  M.FIRSTNAME = C.FIRSTNAME
                        AND M.LASTNAME = C.LASTNAME
                        AND M.ADDRESSLINE1 = C.ADDRESSLINE1
                        AND M.CITY = C.CITY
                        AND M.POSTALCODE = C.POSTALCODE )
GROUP BY M.FIRSTNAME , M.LASTNAME , M.ADDRESSLINE1 , M.CITY ,
         M.STATEPROVINCECODE , M.POSTALCODE
```

From the two examples above we can see that using the EXCEPT and INTERSECT commands are much simpler to write then having to write the join or exists statements.

To take this a step further if we had a third table (or forth...) that listed sales reps and we wanted to find out which managers were customers, but not sales reps we could do the following.

SalesRep table sample data

	FirstName	LastName	AddressLine1	City	StateProvinceCode	PostalCode
1	Ben	Miller	101 Candy Rd.	Redmond	WA	98052
2	Garrett	Vargas	10203 Acorn Avenue	Calgary	AB	T2P 2G8
3	Gabe	Mares	1061 Buskrik Avenue	Edmonds	WA	98020
4	Reuben	D'sa	1064 Slow Creek Road	Seattle	WA	98104
5	Gordon	Hee	108 Lakeside Court	Bellevue	WA	98004

```
SELECT FIRSTNAME ,
       LASTNAME ,
```

```

        ADDRESSLINE1 ,
        CITY ,
        STATEPROVINCECODE ,
        POSTALCODE
FROM    MANAGER
INTERSECT
SELECT  FIRSTNAME ,
        LASTNAME ,
        ADDRESSLINE1 ,
        CITY ,
        STATEPROVINCECODE ,
        POSTALCODE
FROM    CUSTOMER
EXCEPT
SELECT  FIRSTNAME ,
        LASTNAME ,
        ADDRESSLINE1 ,
        CITY ,
        STATEPROVINCECODE ,
        POSTALCODE
FROM    SALESREP

```

Here is the result set:

	FirstName	LastName	AddressLine1	City	StateProvinceCode	PostalCode
1	François	Ajenstat	1144 Paradise Ct.	Issaquah	WA	98027
2	Karan	Khanna	1102 Ravenwood	Seattle	WA	98104
3	Kim	Ralls	1226 Shoe St.	Bothell	WA	98011
4	Kirk	Koenigsbauer	1220 Bradford Way	Seattle	WA	98104
5	Sariya	Harnpadoungsataya	1185 Dallas Drive	Everett	WA	98201

As you can see this is pretty simple to mix and match these statements. In addition, you could also use the UNION and UNION ALL operators to further extend your final result sets.

Next Steps

- Take a look at your existing code to see how the INTERSECT and EXCEPT operators could be used
- Keep these new operators in mind next time you need to compare different datasets with like data

Additional info from our sponsors



Best Database Management Tool

In celebration of being recognized as the best in the Database Management Tools category of the 2007 Editor's Choice Awards by SQL Server Magazine, we're going to make it even



easier to "Upgrade to the Best".

Trade in any database management tool and receive SQL Sentry Event Manager for FREE!

Upgrade to the Best

Key Features: Visual Schedule Management • Alerting/Response System • SSIS/DTS Support
Cross-platform Support • Chaining/Queuing • Schedule Performance Monitoring

Key Benefits: Lower Database Administration Costs • Improve Application Performance
Reduce Downtime • Distributed "Agent-less" Deployment • 100% .NET Based

Copyright (c) 2006-2007 [Edgewood Solutions, LLC](#) All rights reserved
[privacy statement](#) | [disclaimer](#) | [copyright](#)

Some names and products listed are the registered trademarks of their respective owners.