Module 2

Set operators

Module Overview

- Set operators overview
- UNION
- EXCEPT and INTERSECT
- APPLY

Lesson: Set operators overview

- Operating on the result from queries

Operating on the result from queries

- The results of two input queries may be further manipulated
  - You can add more SET operators, in effect combining more than two resultsets
- Sets may be combined, compared, or operated against each other
- The resultsets must have the same number of columns
  - Implicit data type conversion applies
  - Including the rules for data type precedence
- ORDER BY not allowed in input queries, but may be used for result of set operation
- NULLs are considered equal when comparing sets

```
<SELECT query_1>
<set_operator>
<SELECT query_2>
[ORDER BY <sort_list>];
```

Set Operators in SQL Server (UNION, UNION ALL, INTERSECT, EXCEPT)
https://sql-programmers.com/set-operators-in-sql-server-union-union-all-intersect-except

Set Operators (UNION, INTERSECT and EXCEPT)
https://www.w3computing.com/sqlserver2012/set-operators-union-intersect-except/

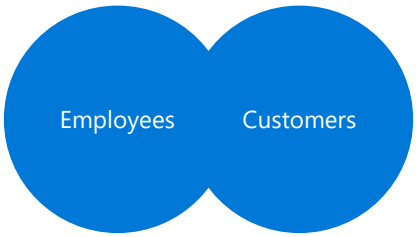
Understanding the interaction between Set Theory and Set Operators in SQL Server
https://www.sqlshack.com/understanding-the-interaction-between-set-theory-and-set-operators-in-sql-server/

Lesson: UNION

- Interactions Between Sets
- The UNION Operator
- The UNION ALL Operator

The UNION Operator

- UNION returns a result set of distinct rows combined from both input sets
- Duplicates are removed during query processing
  - This isn't free (performance)




```
-- Distinct rows from both queries
SELECT country, region, city FROM HR.Employees
UNION
SELECT country, region, city FROM Sales.Customers
```

Set Operators - UNION (Transact-SQL)
https://docs.microsoft.com/en-us/sql/t-sql/language-elements/set-operators-union-transact-sql

SQL Server UNION
https://www.sqlservertutorial.net/sql-server-basics/sql-server-union/

SQL Union overview, usage and examples
https://www.sqlshack.com/sql-union-overview-usage-and-examples/

Using the UNION ALL Operator

- UNION ALL keep duplicate rows
- You avoid he performance overhead of removing duplicates
- Consider whether to use UNION ALL or UNION when you write queries that combine large amount of rows

```
-- All rows from both queries
SELECT country, region, city FROM HR.Employees
UNION ALL
SELECT country, region, city FROM Sales.Customers
```

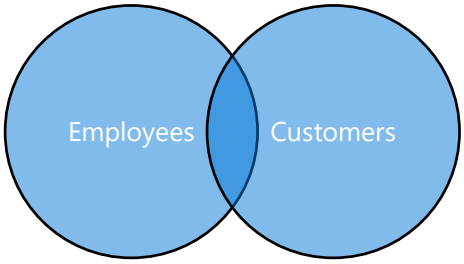
Demo UNION and UNION ALL

Lesson: EXCEPT and INTERSECT

- The INTERSECT Operator
- The EXCEPT Operator

The INTERSECT Operator

- INTERSECT returns the distinct set of rows that appear in both input result sets




```
-- Rows that exist in both results
SELECT country, region, city FROM HR.Employees
INTERSECT
SELECT country, region, city FROM Sales.Customers
```

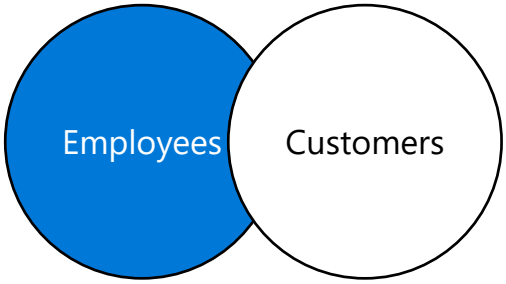
Set Operators - EXCEPT and INTERSECT (Transact-SQL)
https://docs.microsoft.com/en-us/sql/t-sql/language-elements/set-operators-except-and-intersect-transact-sql

SQL Server INTERSECT
https://www.sqlservertutorial.net/sql-server-basics/sql-server-intersect/

SQL intersect use in SQL Server
https://www.sqlshack.com/sql-intersect-use-in-sql-server/

The EXCEPT Operator

- EXCEPT returns only distinct rows that appear in the left set but not the right
  - The order in which sets are specified matters




```
-- Rows from Employees but not Customers
SELECT country, region, city FROM HR.Employees
EXCEPT
SELECT country, region, city FROM Sales.Customers
```

Demo EXCEPT and INTERSECT

Set Operators - EXCEPT and INTERSECT (Transact-SQL)
https://docs.microsoft.com/en-us/sql/t-sql/language-elements/set-operators-except-and-intersect-transact-sql

SQL Server EXCEPT
https://www.sqlservertutorial.net/sql-server-basics/sql-server-except/

Understanding the SQL EXCEPT statement with examples
https://www.sqlshack.com/understanding-the-sql-except-statement-with-examples/

Lesson: APPLY

- APPLY overview
- CROSS APPLY
- OUTER APPLY
- Usages for APPLY

APPLY overview

- APPLY is a table operator that can be used in the FROM clause
- Not in the SQL standard
- Postgres and Snowflake call this *lateral join*
- Operates on two input tables, referred to as left and right
- Right table may be any table expression including a derived table or a table-valued function
- For each row from the left table, the right table is applied and its result is joined to that row from the left table

```
SELECT <column_list>
FROM <left_table_source> AS <alias>
[CROSS]|[OUTER] APPLY
   <right_table_source> AS <alias>
```

FROM clause plus JOIN, APPLY, PIVOT (Transact-SQL)
https://docs.microsoft.com/en-us/sql/t-sql/queries/from-transact-sql

SQL Server CROSS APPLY and OUTER APPLY
https://www.mssqltips.com/sqlservertip/1958/sql-server-cross-apply-and-outer-apply/

The Difference between CROSS APPLY and OUTER APPLY in SQL Server
https://www.sqlshack.com/the-difference-between-cross-apply-and-outer-apply-in-sql-server/

CROSS APPLY

- CROSS APPLY applies the right table source to each row in the left table source
  - Only rows with results in ***both*** the left table source and right table source are returned
- Compare to INNER join

```
SELECT o.orderid, o.orderdate, od.productid, od.unitprice, od.qty
FROM Sales.Orders AS o
CROSS APPLY
(
SELECT productid, unitprice, qty
FROM Sales.OrderDetails AS so
WHERE so.orderid = o.orderid
) AS od
```

OUTER APPLY

- OUTER APPLY applies the right table source to each row in the left table source
  - All rows from the left table source are returned—values from the right table source are returned where they exist, otherwise NULL is returned
- Compare to LEFT OUTER JOIN

```
SELECT DISTINCT s.country AS supplier_country, c.country as customer_country
FROM Production.Suppliers AS s
OUTER APPLY
(
SELECT country
FROM Sales.Customers AS cu
WHERE cu.country = s.country
) AS c
ORDER BY supplier_country
```

Usages for APPLY

- APPLY allow a table function to use columns from the left table in the function's parameter list
  - JOIN doesn't allow for that in SQL Server
- CROSS APPLY and OUTER APPLY allow query expressions that could not appear in a JOIN to return as part of a single result set
  - For example, table-valued functions (TVFs)

```
SELECT S.supplierid, s.companyname, p.productid, p.productname, p.unitprice
FROM Production.Suppliers AS s
CROSS APPLY dbo.fn_TopProductsByShipper(s.supplierid) AS p
```

Demo APPLY

V1 Lab 2: Using Set Operators

- ***Note carefully the folder name in the lab instructions!!!***
- Exercise 1: Writing Queries That Use UNION Set Operators and UNION ALL Multi-Set Operators
- Exercise 2: Writing Queries That Use the CROSS APPLY and OUTER APPLY Operators
- Exercise 3: Writing Queries That Use the EXCEPT and INTERSECT Operators

Estimated Time: 60 minutes

V2 Lab 2: Set operators

- Ex 1. UNION, INTERSECT and EXCEPT
- Ex 2. Using APPLY

Estimated Time: 60 minutes