Module 4

Advanced grouping and pivoting data

Module Overview

- Independent grouping using GROUPING SETS
- Generating super-aggregates using ROLLUP and CUBE
- Pivoting data

Lesson: Working with Grouping Sets

- Using with GROUPING SETS
- Example query suing GROUPING SETS
- Identifying grouped columns using GROUPING and GROUPING_ID functions

Writing Queries with Grouping Sets

- GROUPING SETS subclause builds on T-SQL GROUP BY clause
- Allows multiple groupings to be defined in same query
- Alternative to use of UNION ALL to combine multiple outputs (each with different GROUP BY) into one result set

```
SELECT <column list with aggregate(s)>
FROM <source>
GROUP BY
GROUPING SETS(
        (col1),          -- single column
        (col2, col3),    -- combination of columns
        ()               -- empty parentheses aggregates all rows
                )
```

SELECT - GROUP BY- Transact-SQL
https://docs.microsoft.com/en-us/sql/t-sql/queries/select-group-by-transact-sql

What Is the SQL GROUPING SETS Clause, and How Do You Use it?
https://learnsql.com/blog/sql-grouping-sets-clause/

SQL Server GROUPING SETS
https://www.sqlservertutorial.net/sql-server-basics/sql-server-grouping-sets/

Example query suing GROUPING SETS

```
SELECT Category, Cust, SUM(Qty) AS TotalQty
FROM Sales.CategorySales
GROUP BY
GROUPING SETS((Category),(Cust),());
```

```
Category        Cust        TotalQty
--------------- ----------- -----------
NULL            NULL        999
NULL            1           80
NULL            2           12
NULL            3           154
NULL            4           241
NULL            5           512
Beverages       NULL        513
Condiments      NULL        114
Confections     NULL        372
```

Identifying grouped columns using GROUPING and GROUPING_ID functions

- GROUPING SETS basically is a union over several results with different grouping column.
  - This means that for some rows a column **is** involved in the grouping
  - While the same column for other rows that column **is not** involved in the grouping
  - We have identified these column so far having NULL
  - That is not reliable since the underlying data column also be NULL
- The GROUPING aggregate function returns 0 or 1 based on if that column is involved in the grouping or not
- The GROUPING_ID aggregate function is like GROUPING, but can input more than one column and returns a bit-mask

```
SELECT GROUPING (Category)AS grpCat, GROUPING_ID(Category, Cust) AS grpInfo,
        Category, Cust, SUM(Qty) AS TotalQty
FROM Sales.CategorySales
GROUP BY CUBE(Category,Cust)
```

Demo GROUPING SETS

Understanding GROUPING and GROUPING_ID Functions in SQL Server
https://codingsight.com/understanding-grouping-and-grouping_id-functions-in-sql-server/

Lesson: Generating super-aggregates using ROLLUP and CUBE

- CUBE and ROLLUP
- Identifying grouped columns using GROUPING and GROUPING_ID functions

CUBE and ROLLUP

- ROLLUP provides shortcut for defining grouping sets, creates combinations assuming input columns form a hierarchy

```
SELECT Category, Cust, SUM(Qty) AS TotalQty
FROM Sales.CategorySales
GROUP BY ROLLUP(Category,Cust)
ORDER BY Category, Cust;
```

- CUBE provides shortcut for defining grouping sets given a list of columns
- All possible combinations of grouping sets created

```
SELECT Category, Cust, SUM(Qty) AS TotalQty
FROM Sales.CategorySales
GROUP BY CUBE(Category,Cust)
ORDER BY Category, Cust;
```

Group By in SQL Server with CUBE, ROLLUP and GROUPING SETS Examples
https://www.mssqltips.com/sqlservertip/6315/group-by-in-sql-server-with-cube-rollup-and-grouping-sets-examples/

SQL Server ROLLUP
https://www.sqlservertutorial.net/sql-server-basics/sql-server-rollup/

SQL Server CUBE
https://www.sqlservertutorial.net/sql-server-basics/sql-server-cube/

Identifying grouped columns using GROUPING and GROUPING_ID functions

- GROUPING SETS basically is a union over several results with different grouping column.
  - This means that for some rows a column **is** involved in the grouping
  - While the same column for other rows that column **is not** involved in the grouping
  - We have identified these column so far having NULL
  - That is not reliable since the underlying data column also be NULL
- The GROUPING aggregate function returns 0 or 1 based on if that column is involved in the grouping or not
- The GROUPING_ID aggregate function is like GROUPING, but can input more than one column and returns a bit-mask

```
SELECT GROUPING (Category)AS grpCat, GROUPING_ID(Category, Cust) AS grpInfo,
        Category, Cust, SUM(Qty) AS TotalQty
FROM Sales.CategorySales
GROUP BY CUBE(Category,Cust)
```

Understanding GROUPING and GROUPING_ID Functions in SQL Server
https://codingsight.com/understanding-grouping-and-grouping_id-functions-in-sql-server/

Lesson: Pivoting data

- What Is Pivoting?
- Elements of PIVOT
- Writing queries with UNPIVOT

What is pivoting?

- Pivoting data is rotating data from a rows-based orientation to a columns-based orientation
- Distinct values from a single column are projected across as headings for other columns—may include aggregation

Category	Qty	OrderYear
Beverages	5	2006
Beverages	9	2007
Beverages	4	2008
Dairy Products	7	2006
Dairy Products	7	2007
Dairy Products	6	2008
Meat	3	2006
Meat	4	2006
Meat	6	2007
Seafood	9	2006
Seafood	1	2007
Seafood	5	2007

Category	2006	2007	2008
Beverages	5	9	4
Dairy Products	7	7	6
Meat	7	6	NULL
Seafood	9	6	NULL

FROM - Using PIVOT and UNPIVOT
https://docs.microsoft.com/en-us/sql/t-sql/queries/from-using-pivot-and-unpivot

SQL Server PIVOT
https://www.sqlservertutorial.net/sql-server-basics/sql-server-pivot/

Elements of PIVOT

- Pivoting includes three phases:
  - Grouping determines which element gets a row in the result set
  - Spreading provides the distinct values to be pivoted across
  - Aggregation performs an aggregation function (such as SUM)

Elements of PIVOT

```
SELECT  Category, [2006],[2007],[2008]
FROM (SELECT  Category, Qty, Orderyear FROM Sales.CategoryQtyYear) AS D
PIVOT(SUM(QTY) FOR orderyear IN ([2006],[2007],[2008])) AS pvt
ORDER BY Category;
```

```
Category       2006 2007 2008
-------------- ---- ---- ----
Beverages      1842 3996 3694
Condiments     962  2895 1441
Confections    1357 4137 2412
Dairy Products 2086 4374 2689
Grains/Cereals 549  2636 1377
Meat/Poultry   950  2189 1060
Produce        549  1583 858
Seafood        1286 3679 2716
```

Writing queries with UNPIVOT

- Unpivoting data is rotating data from a columns-based orientation to a rows-based orientation
- Spreads or splits values from one source row into one or more target rows
- Each source row becomes one or more rows in result set based on number of columns being pivoted
- Unpivoting includes three elements:
  - Source columns to be unpivoted
  - Name to be assigned to new values column
  - Name to be assigned to names columns

Demo PIVOT and UNPIVOT

V1 Optional Lab 4: Pivoting and Grouping Sets

- This is an optional lab

- ***Note carefully the folder name in the lab instructions!!!***
- Exercise 1: Writing Queries That Uses PIVOT
- Exercise 2: Writing Queries That Uses UNPIVOT
- Exercise 3: Writing Queries That Uses GROUPING SETS, CUBE, and ROLLUP

Estimated Time: 45 minutes

V2 Optional Lab 4: Advanced grouping and pivoting data

- Ex 1. Performing independent aggregations
- Ex 2. Pivot the result from a query

Estimated Time: 45 minutes