UNIVERSIDAD PRIVADA DE TACNA

FACULTAD DE INGENIERIA

Escuela Profesional de Ingeniería de Sistemas

PRACTICA DE LABORATORIO: PIVOTING Y GROUPING SETS

Curso: Base de Datos II

Docente: Mag. Ing. Patrick Cuadros Quiroga

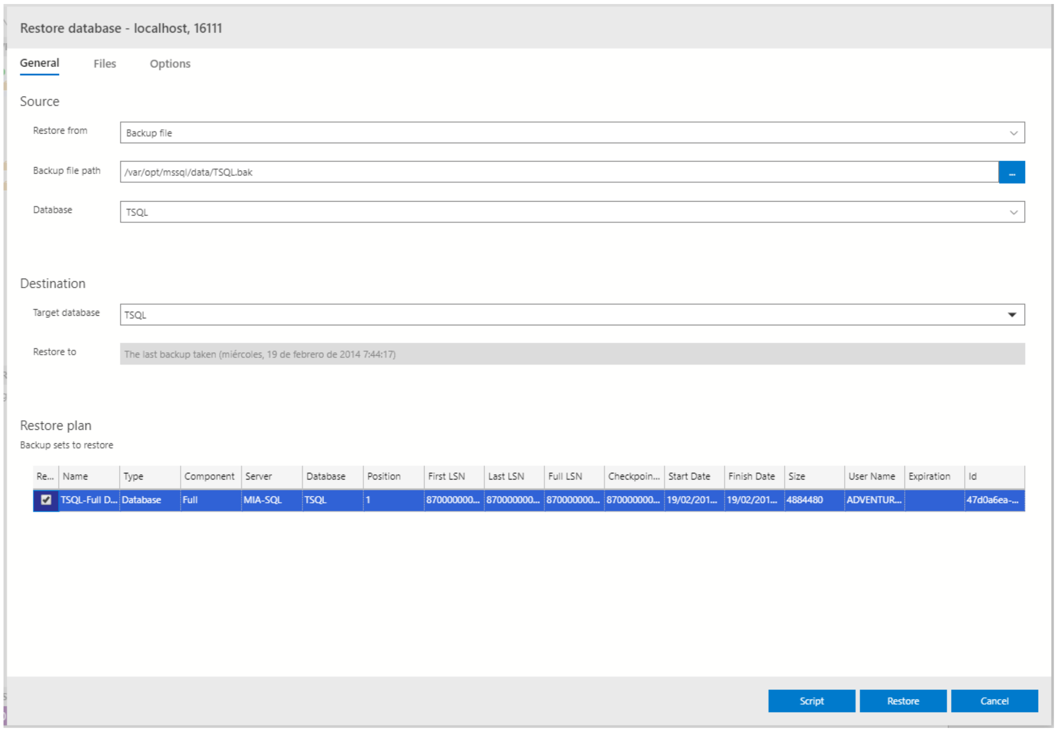
GOMEZ QUIROZ, YUMIN YHULYÑO (2015052385)

Tacna – Perú
2021

PIVOTING Y GROUPING SETS

RESTAURAR BASE DE DATOS

1. Una vez desplegado e iniciado nuestro contenedor, vamos a restaurar la base de datos TSQL.



Restore database - localhost, 16111

Source

Restore from: Backup file

Backup file path: /var/opt/mssql/data/TSQL.bak

Database: TSQL

Destination

Target database: TSQL

Restore to: The last backup taken (miércoles, 19 de febrero de 2014 7:44:17)

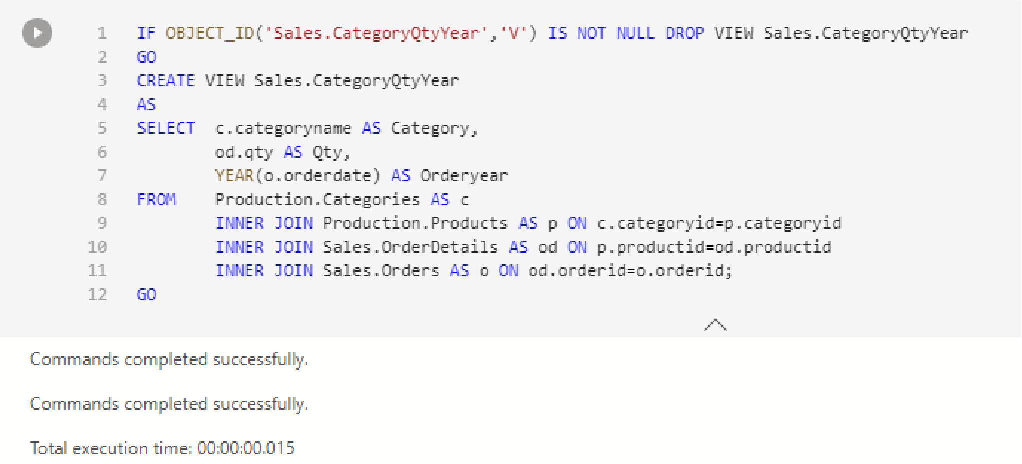
Restore plan

Backup sets to restore

Re...	Name	Type	Component	Server	Database	Position	First LSN	Last LSN	Full LSN	Checkpoin...	Start Date	Finish Date	Size	User Name	Expiration	Id
☑	TSQL-Full D...	Database	Full	MIA-SQL	TSQL	1	870000000...	870000000...	870000000...	870000000...	19/02/201...	19/02/201...	4884480	ADVENTUR...		47d0a6ea-...

PARTE 1: ESCRIBIENDO CONSULTAS CON PIVOT Y UNPIVOT

1. Cargamos el documento de la sesión y vamos a crear una vista de categorías de productos y sus cantidades por año.



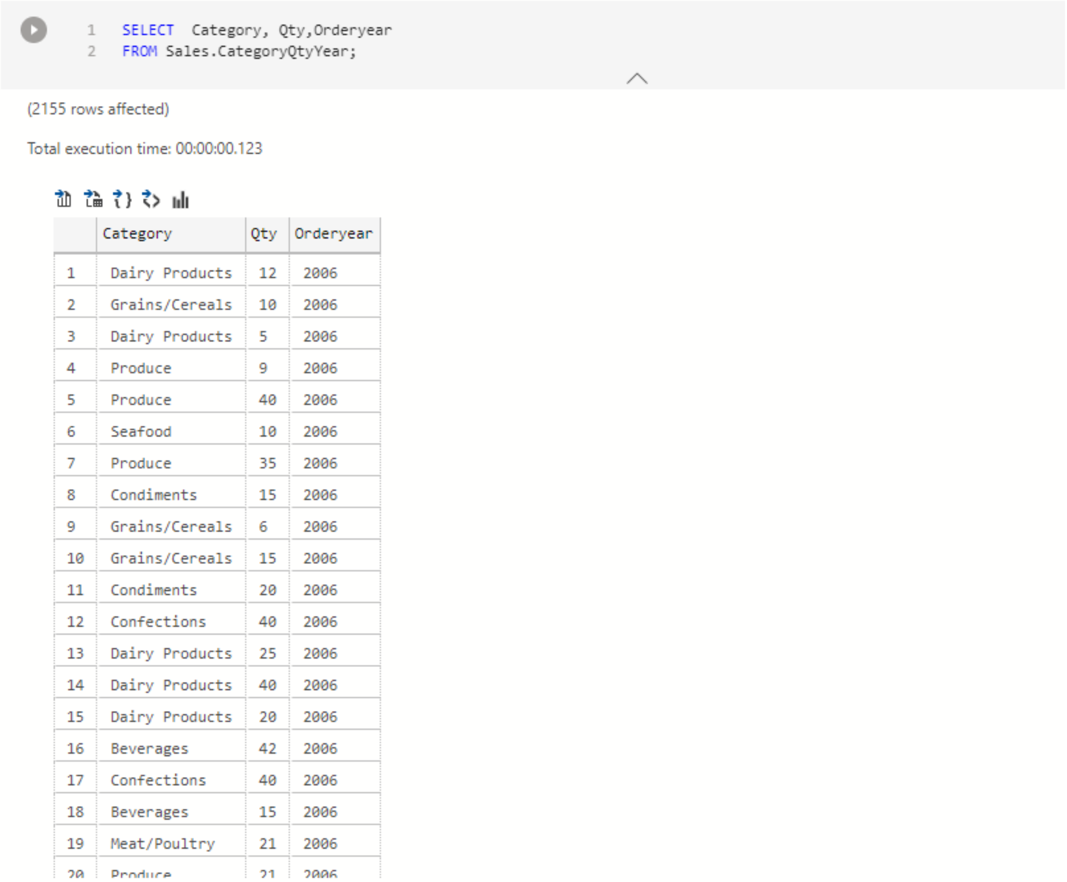
```sql
IF OBJECT_ID('Sales.CategoryQtyYear','V') IS NOT NULL DROP VIEW Sales.CategoryQtyYear
GO
CREATE VIEW Sales.CategoryQtyYear
AS
SELECT  c.categoryname AS Category,
        od.qty AS Qty,
        YEAR(o.orderdate) AS Orderyear
FROM    Production.Categories AS c
        INNER JOIN Production.Products AS p ON c.categoryid=p.categoryid
        INNER JOIN Sales.OrderDetails AS od ON p.productid=od.productid
        INNER JOIN Sales.Orders AS o ON od.orderid=o.orderid;
GO
```

```
Commands completed successfully.

Commands completed successfully.

Total execution time: 00:00:00.015
```

2. Probaremos la vista creada y nos botara 2155 datos.



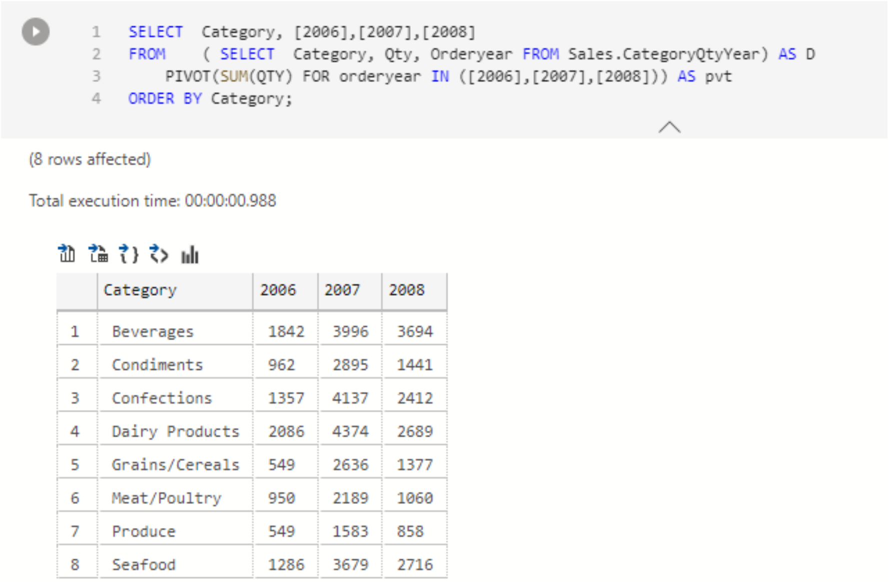
```sql
SELECT  Category, Qty,Orderyear
FROM Sales.CategoryQtyYear;
```

```
(2155 rows affected)

Total execution time: 00:00:00.123
```

	Category	Qty	Orderyear
1	Dairy Products	12	2006
2	Grains/Cereals	10	2006
3	Dairy Products	5	2006
4	Produce	9	2006
5	Produce	40	2006
6	Seafood	10	2006
7	Produce	35	2006
8	Condiments	15	2006
9	Grains/Cereals	6	2006
10	Grains/Cereals	15	2006
11	Condiments	20	2006
12	Confections	40	2006
13	Dairy Products	25	2006
14	Dairy Products	40	2006
15	Dairy Products	20	2006
16	Beverages	42	2006
17	Confections	40	2006
18	Beverages	15	2006
19	Meat/Poultry	21	2006
20	Produce	21	2006

3. Utilizaremos PIVOT entre categorías y años de órdenes.



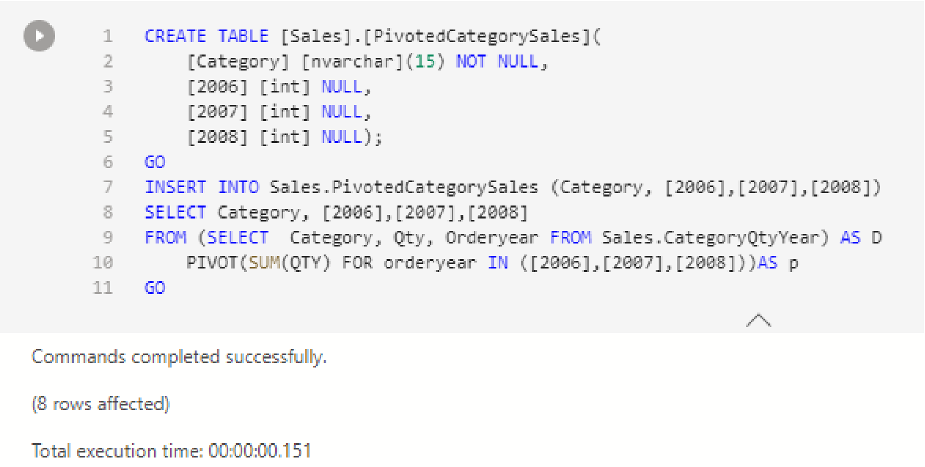
```sql
SELECT  Category, [2006],[2007],[2008]
FROM    ( SELECT  Category, Qty, Orderyear FROM Sales.CategoryQtyYear) AS D
    PIVOT(SUM(QTY) FOR orderyear IN ([2006],[2007],[2008])) AS pvt
ORDER BY Category;
```

```
(8 rows affected)

Total execution time: 00:00:00.988
```

	Category	2006	2007	2008
1	Beverages	1842	3996	3694
2	Condiments	962	2895	1441
3	Confections	1357	4137	2412
4	Dairy Products	2086	4374	2689
5	Grains/Cereals	549	2636	1377
6	Meat/Poultry	950	2189	1060
7	Produce	549	1583	858
8	Seafood	1286	3679	2716

4. Ahora si deseamos utilizar UNPIVOT, crearemos una tabla con el resultado de la consulta con el PIVOT anterior.



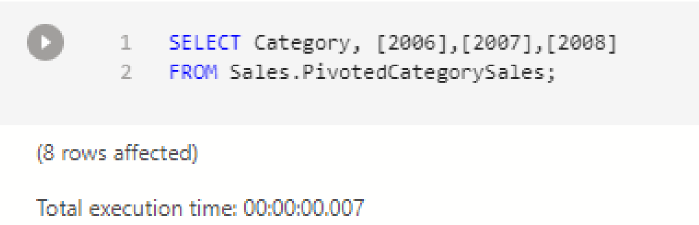
```
CREATE TABLE [Sales].[PivotedCategorySales](
    [Category] [nvarchar](15) NOT NULL,
    [2006] [int] NULL,
    [2007] [int] NULL,
    [2008] [int] NULL);
GO
INSERT INTO Sales.PivotedCategorySales (Category, [2006],[2007],[2008])
SELECT Category, [2006],[2007],[2008]
FROM (SELECT  Category, Qty, Orderyear FROM Sales.CategoryQtyYear) AS D
    PIVOT(SUM(QTY) FOR orderyear IN ([2006],[2007],[2008]))AS p
GO
```

```
Commands completed successfully.

(8 rows affected)

Total execution time: 00:00:00.151
```

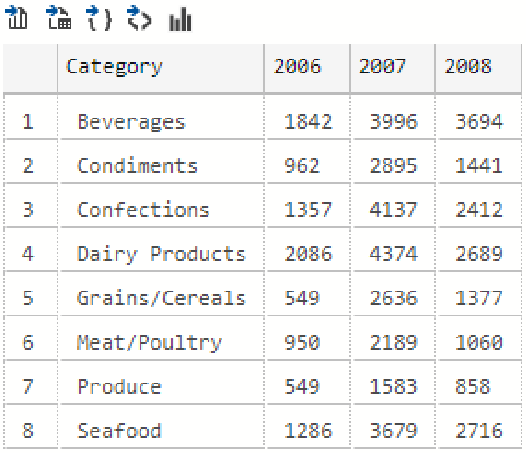
5. Probamos la tabla que se ha generado.



```
SELECT Category, [2006],[2007],[2008]
FROM Sales.PivotedCategorySales;
```

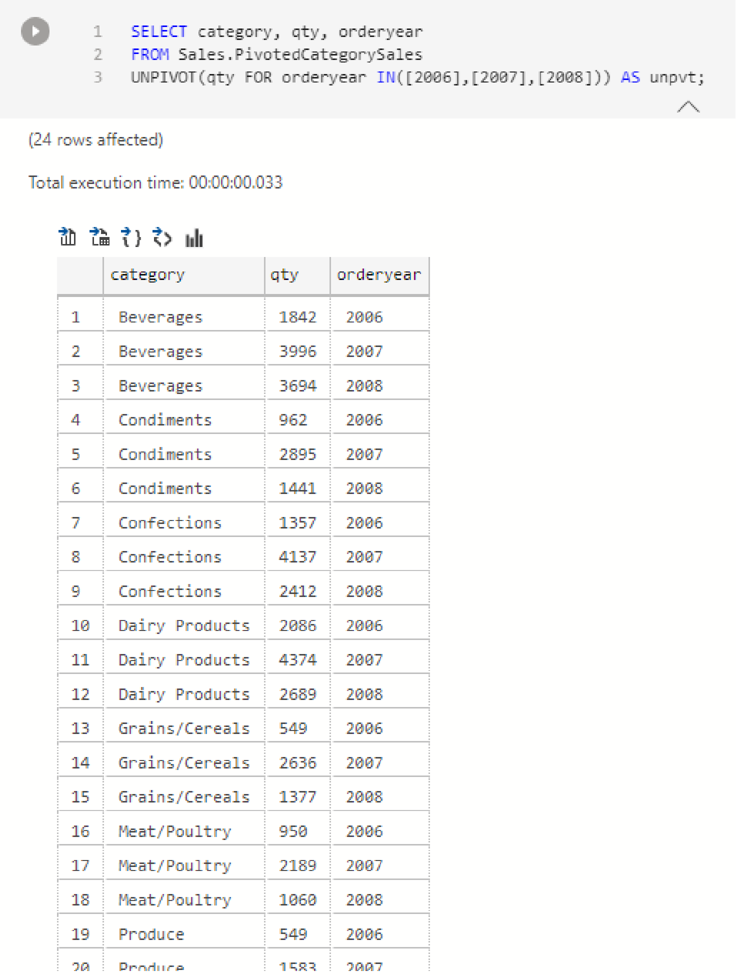
```
(8 rows affected)

Total execution time: 00:00:00.007
```



	Category	2006	2007	2008
1	Beverages	1842	3996	3694
2	Condiments	962	2895	1441
3	Confections	1357	4137	2412
4	Dairy Products	2086	4374	2689
5	Grains/Cereals	549	2636	1377
6	Meat/Poultry	950	2189	1060
7	Produce	549	1583	858
8	Seafood	1286	3679	2716

6. Utilizaremos UNPIVOT.



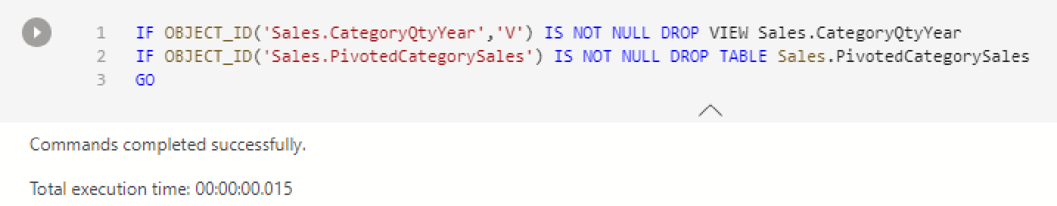
```sql
SELECT category, qty, orderyear
FROM Sales.PivotedCategorySales
UNPIVOT(qty FOR orderyear IN([2006],[2007],[2008])) AS unpvt;
```

(24 rows affected)

Total execution time: 00:00:00.033

	category	qty	orderyear
1	Beverages	1842	2006
2	Beverages	3996	2007
3	Beverages	3694	2008
4	Condiments	962	2006
5	Condiments	2895	2007
6	Condiments	1441	2008
7	Confections	1357	2006
8	Confections	4137	2007
9	Confections	2412	2008
10	Dairy Products	2086	2006
11	Dairy Products	4374	2007
12	Dairy Products	2689	2008
13	Grains/Cereals	549	2006
14	Grains/Cereals	2636	2007
15	Grains/Cereals	1377	2008
16	Meat/Poultry	950	2006
17	Meat/Poultry	2189	2007
18	Meat/Poultry	1060	2008
19	Produce	549	2006
20	Produce	1583	2007

7. Limpiaremos ahora los cambios realizados.



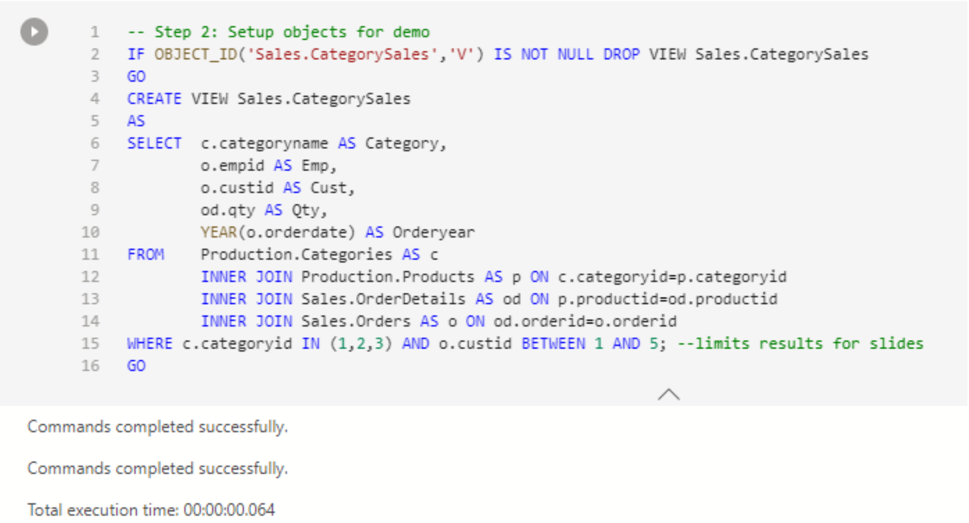
```sql
IF OBJECT_ID('Sales.CategoryQtyYear','V') IS NOT NULL DROP VIEW Sales.CategoryQtyYear
IF OBJECT_ID('Sales.PivotedCategorySales') IS NOT NULL DROP TABLE Sales.PivotedCategorySales
GO
```

Commands completed successfully.

Total execution time: 00:00:00.015

PARTE 2: TRABAJANDO CON GROUPING SETS

1. Ejecutamos la siguiente vista.



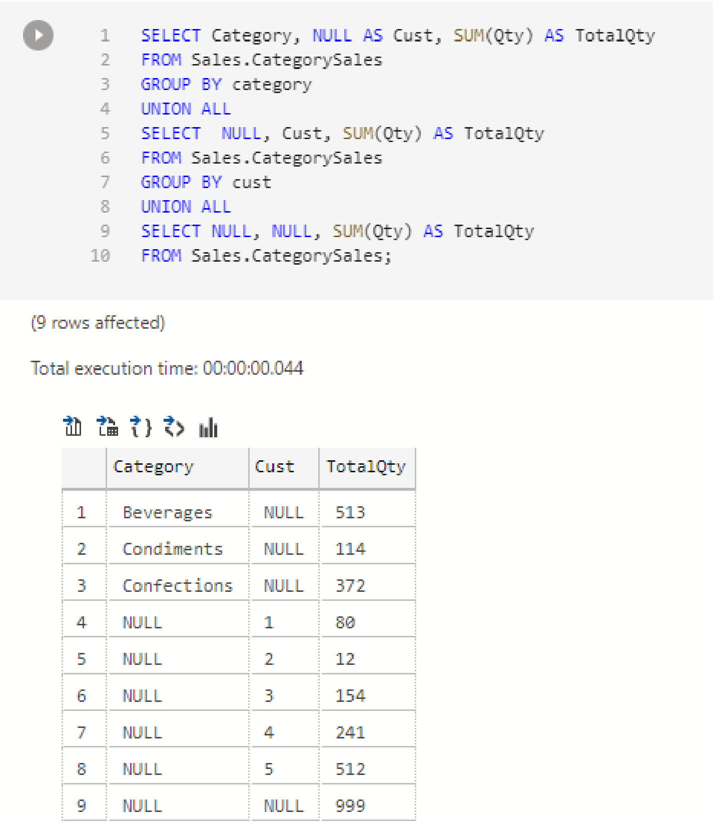
```sql
-- Step 2: Setup objects for demo
IF OBJECT_ID('Sales.CategorySales','V') IS NOT NULL DROP VIEW Sales.CategorySales
GO
CREATE VIEW Sales.CategorySales
AS
SELECT  c.categoryname AS Category,
        o.empid AS Emp,
        o.custid AS Cust,
        od.qty AS Qty,
        YEAR(o.orderdate) AS Orderyear
FROM    Production.Categories AS c
        INNER JOIN Production.Products AS p ON c.categoryid=p.categoryid
        INNER JOIN Sales.OrderDetails AS od ON p.productid=od.productid
        INNER JOIN Sales.Orders AS o ON od.orderid=o.orderid
WHERE c.categoryid IN (1,2,3) AND o.custid BETWEEN 1 AND 5; --limits results for slides
GO
```

```
Commands completed successfully.

Commands completed successfully.

Total execution time: 00:00:00.064
```

2. Elaboramos una consulta sin utilizar GROUPING SETS.



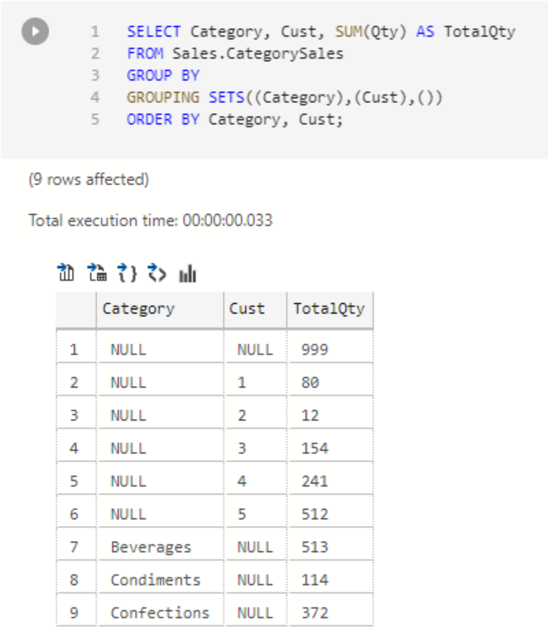
```sql
SELECT Category, NULL AS Cust, SUM(Qty) AS TotalQty
FROM Sales.CategorySales
GROUP BY category
UNION ALL
SELECT  NULL, Cust, SUM(Qty) AS TotalQty
FROM Sales.CategorySales
GROUP BY cust
UNION ALL
SELECT NULL, NULL, SUM(Qty) AS TotalQty
FROM Sales.CategorySales;
```

```
(9 rows affected)

Total execution time: 00:00:00.044
```

	Category	Cust	TotalQty
1	Beverages	NULL	513
2	Condiments	NULL	114
3	Confections	NULL	372
4	NULL	1	80
5	NULL	2	12
6	NULL	3	154
7	NULL	4	241
8	NULL	5	512
9	NULL	NULL	999

3. Consultamos con GROUPING SETS.



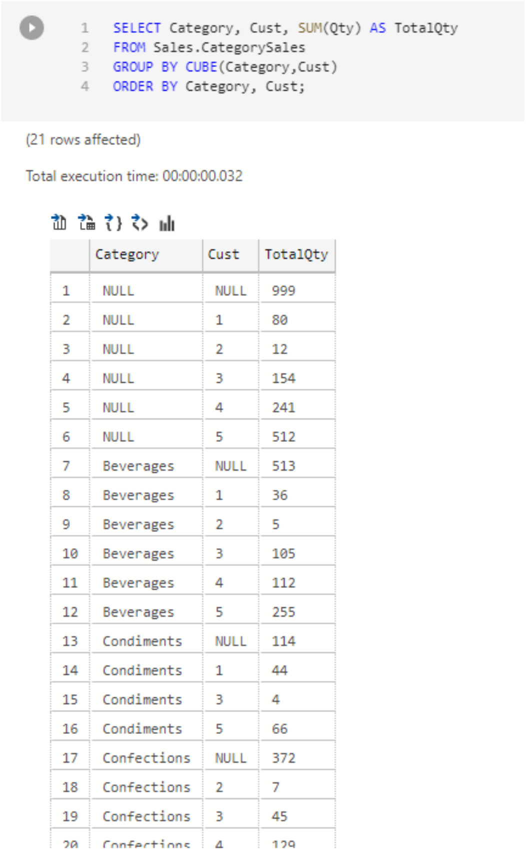
```sql
SELECT Category, Cust, SUM(Qty) AS TotalQty
FROM Sales.CategorySales
GROUP BY
GROUPING SETS((Category),(Cust),())
ORDER BY Category, Cust;
```

(9 rows affected)

Total execution time: 00:00:00.033

	Category	Cust	TotalQty
1	NULL	NULL	999
2	NULL	1	80
3	NULL	2	12
4	NULL	3	154
5	NULL	4	241
6	NULL	5	512
7	Beverages	NULL	513
8	Condiments	NULL	114
9	Confections	NULL	372

4. Consultamos con CUBE.



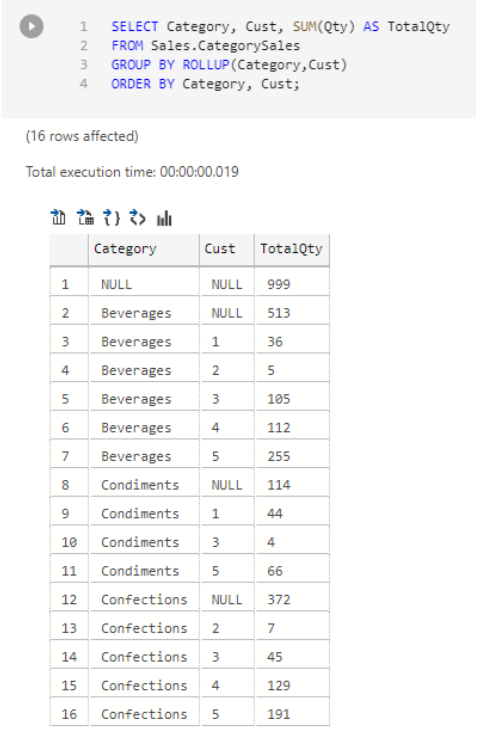
```sql
SELECT Category, Cust, SUM(Qty) AS TotalQty
FROM Sales.CategorySales
GROUP BY CUBE(Category,Cust)
ORDER BY Category, Cust;
```

(21 rows affected)

Total execution time: 00:00:00.032

	Category	Cust	TotalQty
1	NULL	NULL	999
2	NULL	1	80
3	NULL	2	12
4	NULL	3	154
5	NULL	4	241
6	NULL	5	512
7	Beverages	NULL	513
8	Beverages	1	36
9	Beverages	2	5
10	Beverages	3	105
11	Beverages	4	112
12	Beverages	5	255
13	Condiments	NULL	114
14	Condiments	1	44
15	Condiments	3	4
16	Condiments	5	66
17	Confections	NULL	372
18	Confections	2	7
19	Confections	3	45
20	Confections	4	129

5. Consultamos con ROLLUP.



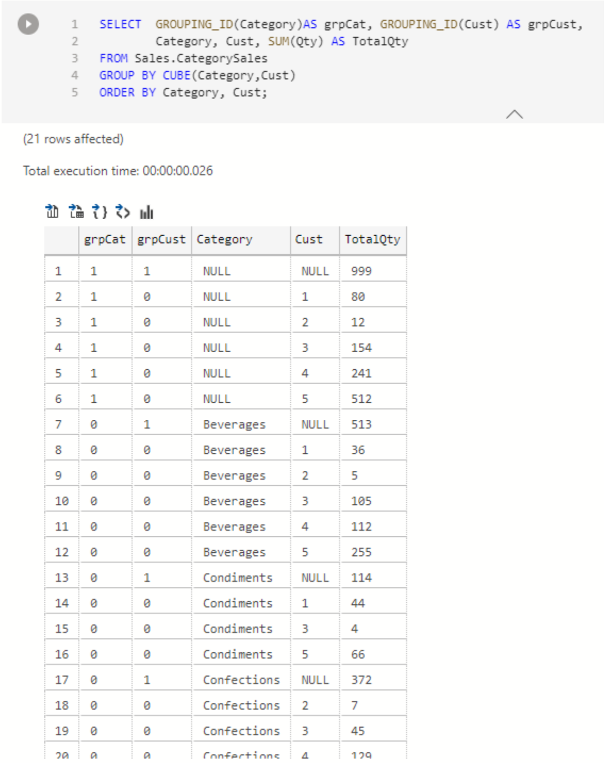
```sql
SELECT Category, Cust, SUM(Qty) AS TotalQty
FROM Sales.CategorySales
GROUP BY ROLLUP(Category,Cust)
ORDER BY Category, Cust;
```

(16 rows affected)

Total execution time: 00:00:00.019

	Category	Cust	TotalQty
1	NULL	NULL	999
2	Beverages	NULL	513
3	Beverages	1	36
4	Beverages	2	5
5	Beverages	3	105
6	Beverages	4	112
7	Beverages	5	255
8	Condiments	NULL	114
9	Condiments	1	44
10	Condiments	3	4
11	Condiments	5	66
12	Confections	NULL	372
13	Confections	2	7
14	Confections	3	45
15	Confections	4	129
16	Confections	5	191

6. Utilizaremos Grouping_ID.



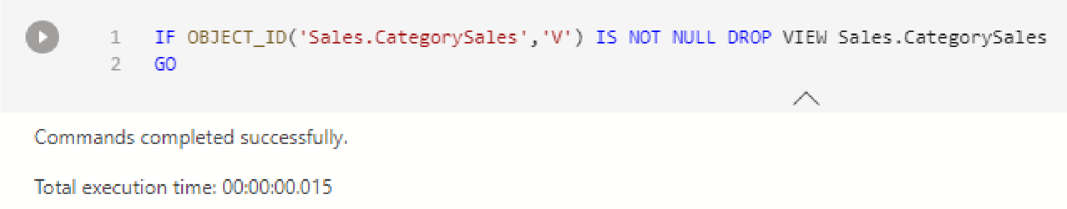
```sql
SELECT  GROUPING_ID(Category)AS grpCat, GROUPING_ID(Cust) AS grpCust,
        Category, Cust, SUM(Qty) AS TotalQty
FROM Sales.CategorySales
GROUP BY CUBE(Category,Cust)
ORDER BY Category, Cust;
```

(21 rows affected)

Total execution time: 00:00:00.026

	grpCat	grpCust	Category	Cust	TotalQty
1	1	1	NULL	NULL	999
2	1	0	NULL	1	80
3	1	0	NULL	2	12
4	1	0	NULL	3	154
5	1	0	NULL	4	241
6	1	0	NULL	5	512
7	0	1	Beverages	NULL	513
8	0	0	Beverages	1	36
9	0	0	Beverages	2	5
10	0	0	Beverages	3	105
11	0	0	Beverages	4	112
12	0	0	Beverages	5	255
13	0	1	Condiments	NULL	114
14	0	0	Condiments	1	44
15	0	0	Condiments	3	4
16	0	0	Condiments	5	66
17	0	1	Confections	NULL	372
18	0	0	Confections	2	7
19	0	0	Confections	3	45
20	0	0	Confections	4	129

7. Finalmente limpiamos los cambios.



```
IF OBJECT_ID('Sales.CategorySales','V') IS NOT NULL DROP VIEW Sales.CategorySales
GO
```

```
Commands completed successfully.

Total execution time: 00:00:00.015
```