Boosting SQL Query Flexibility with CROSS APPLY

CROSS APPLY is useful SQL operator that lets you look up related information for each row in a table. It is especially useful when you want to get extra details that depend on each specific row. It is especially handy when you need to perform row-by-row operations or generate additional columns dynamically, based on each row in the main table.

A great scenario for using CROSS APPLY is finding the top three highest-priced items in every order. It allows you to run a subquery for each order, efficiently retrieving only the highest-priced items. This enables dynamic filtering and precise data retrieval, highlighting the effectiveness of CROSS APPLY for managing complex data relationships.

Setup Example and Populating Example Data

The Orders Table

```
create table orders
(
  [orderid] int,
  [orderdate] date
)


insert into [dbo].[orders]
values
(1, '2024-11-01'),
(2, '2024-11-02'),
(3, '2024-11-03')
```

The Products Table

```
create table products
(
  [productid] int,
  [orderid] int,
  [productname] varchar(50),
  [price] decimal(10, 2)
);


insert into products
values
(101, 1, 'ProductA', 10.00),
(102, 1, 'ProductB', 20.00),
(103, 1, 'ProductC', 30.00),
(104, 1, 'ProductD', 40.00),
(105, 2, 'ProductE', 50.00),
(106, 2, 'ProductF', 60.00),
(107, 2, 'ProductG', 70.00),
(108, 3, 'ProductH', 80.00),
(109, 3, 'ProductI', 90.00),
(110, 3, 'ProductJ', 100.00)
```

The Query: Using CROSS APPLY to Get the Top 3 Most Expensive Products for Each Order

```
select
[order].[orderid],
[order].[orderdate],
[cross_prod].[productid],
[cross_prod].[productname],
[cross_prod].[price]
from [dbo].[orders] as [order]
cross apply
(
  select top 3
  [products].[productid],
  [products].[productname],
  [products].[price]
  from [dbo].[products] as [products]
  where [products].[orderid] = [order].[orderid]
  order by [products].[price] desc
) as [cross_prod];
```

The Output

orderid	orderdate	productid	productname	price
1	2024-11-01	104	ProductD	40.00
1	2024-11-01	103	ProductC	30.00
1	2024-11-01	102	ProductB	20.00
2	2024-11-02	107	ProductG	70.00
2	2024-11-02	106	ProductF	60.00
2	2024-11-02	105	ProductE	50.00
3	2024-11-03	110	ProductJ	100.00
3	2024-11-03	109	ProductI	90.00
3	2024-11-03	108	ProductH	80.00

Explanation

The CROSS APPLY joins each row in Orders to the top 3 products for that specific order, sorted by price in descending order.

This approach is efficient for returning related data where each subset (in this case, products for each order) has its own ranking or aggregation, which can be tricky to achieve with traditional joins alone.

Why Use CROSS APPLY?

Dynamic Row-by-Row Operations: CROSS APPLY allows you to execute a query for each row of the outer query. This is beneficial for dynamic operations where each subset has unique filtering or ordering criteria.

Retrieving Top/Bottom N Rows per Group: This is one of the most common uses of CROSS APPLY—retrieving the top or bottom N rows for each grouping, as in our example.

Handling Table-Valued Functions: It can also be used to invoke table-valued functions with parameters from the outer query, making it great for modular queries.

CROSS APPLY is efficient and expressive tool in SQL for scenarios requiring complex aggregations or subsets of data tied to each row in a primary table and often provides better performance by avoiding unnecessary joins and providing a more intuitive way to express relationships between data sets.