CrossTab Query with Case

A cross tab query normally aggregates data and displays it with the aggregate values as the columns.

Suppose we want to see the total quantity sold for each category of item. We could do a regular aggregate grouping by the category

Demo 01: Group by category

```
Select catg_id
, sum(quantity_ordered)    as QuantitySold
From a_oe.order_details
Join a_prd.products  using(prod_id)
Group by catg_id
;
+---------+--------------+
| catg_id | QuantitySold |
+---------+--------------+
| APL     |           48 |
| HD      |           29 |
| HW      |           62 |
| MUS     |           11 |
| PET     |          170 |
| SPG     |          201 |
+---------+--------------+
```

Demo 02: If we filter for a value of catg_id, we get an aggregate across the table and one row returned.

```
Select sum(quantity_ordered) as QuantitySold
From a_oe.order_details
Join a_prd.products  using(prod_id)
Where catg_id = 'APL'
;
+--------------+
| QuantitySold |
+--------------+
|           48 |
+--------------+
```

Demo 03: We can also use the case expression to include only the rows for APL in the total.

```
Select sum(case when catg_id = 'APL' then quantity_ordered else null end)
   as APL_QuantitySold
From a_oe.order_details
Join a_prd.products  using(prod_id)
;
+------------------+
| APL_QuantitySold |
+------------------+
|               48 |
+------------------+
```

But sometimes people want to see the output displayed in a different way with the different category names used as the headers and the quantity displayed under each header. That is called a Cross tab query.

1. Aggregates & Case for a Cross Tab

This is one way to generate a "CrossTab" query. You do have to create case expression for each column that you want returned. The first column does the sum for the appliances; if the row is for an appliance (APL), then its quantity is part of the Sum for that column. The second column does the sum for the sporting goods items.

Demo 04: We want to know how many products of each of these categories are on order.

```
Select
    sum(case when catg_id = 'APL' then quantity_ordered else null end) APL_QuantitySold
,   sum(case when catg_id = 'SPG' then quantity_ordered else null end) SPG_QuantitySold
,   sum(case when catg_id = 'HW' then quantity_ordered else null end) HW_QuantitySold
,   sum(case when catg_id = 'PET' then quantity_ordered else null end) PET_QuantitySold
From a_oe.order_details
Join a_prd.products on a_oe.order_details.prod_id= a_prd.products.prod_id
;
+------------------+------------------+-----------------+------------------+
| APL_QuantitySold | SPG_QuantitySold | HW_QuantitySold | PET_QuantitySold |
+------------------+------------------+-----------------+------------------+
|               48 |              201 |              62 |              170 |
+------------------+------------------+-----------------+------------------+
```

Demo 05: We want to know how many products of each of these categories are on EACH order.

```
Select    ord_id
,   sum(case when catg_id = 'APL' then quantity_ordered else null end) AS APL_Quant
,   sum(case when catg_id = 'SPG' then quantity_ordered else null end) AS SPG_Quant
,   sum(case when catg_id = 'HW' then quantity_ordered else null end) AS HW_Quant
,   sum(case when catg_id = 'PET' then quantity_ordered else null end) AS PET_Quant
From a_oe.order_details
Join a_prd.products on a_oe.order_details.prod_id= a_prd.products.prod_id
Group by ord_id;
+--------+-----------+-----------+----------+-----------+
| ord_id | APL_Quant | SPG_Quant | HW_Quant | PET_Quant |
+--------+-----------+-----------+----------+-----------+
|    105 |      NULL |        29 |     NULL |      NULL |
|    106 |      NULL |         1 |     NULL |      NULL |
|    107 |      NULL |      NULL |        1 |      NULL |
|    108 |      NULL |      NULL |        1 |      NULL |
|    109 |         1 |      NULL |     NULL |      NULL |
|    110 |         1 |      NULL |        1 |      NULL |
|    111 |      NULL |      NULL |     NULL |        51 |
|    112 |      NULL |      NULL |        2 |      NULL |
|    113 |      NULL |      NULL |        1 |      NULL |
|    114 |         5 |      NULL |     NULL |      NULL |
|    115 |         4 |      NULL |        7 |      NULL |
|    117 |      NULL |         1 |     NULL |         8 |
. . . rows ommitted
```

Demo 06: How many orders for each customer for each of these three months of last year?

```
set @Mnth_1 := 10;
set @Mnth_2 := 11;
set @Mnth_3 := 12;

Select    cust_id
, count(case when month(ord_date)= @Mnth_1  then 1 else null end) as "Month 1"
, count(case when month(ord_date)= @Mnth_2  then 1 else null end) as "Month 2"
, count(case when month(ord_date)= @Mnth_3  then 1 else null end) as "Month 3"
From a_oe.order_headers
Where year(ord_date)= year(curDate()) -1
```

```
Group by cust_id
Order by cust_id
;
+---------+---------+---------+---------+
| cust_id | Month 1 | Month 2 | Month 3 |
+---------+---------+---------+---------+
|  400300 |       0 |       0 |       0 |
|  401250 |       1 |       2 |       0 |
|  401890 |       0 |       1 |       0 |
|  402100 |       0 |       3 |       0 |
|  403000 |       3 |       1 |       0 |
|  403010 |       0 |       1 |       0 |
|  403050 |       1 |       0 |       0 |
|  403100 |       3 |       1 |       0 |
|  404000 |       0 |       0 |       0 |
|  404100 |       0 |       0 |       0 |
. . .
```

Demo 07: Analyze quantity of items purchased by price

```
Select
   sum(case when quoted_price between 0.01 and 25
        then quantity_ordered
        else 0 end)  as "Price 0.01-25"
,  sum(case when quoted_price between 25.01 and 100
        then quantity_ordered
        else 0 end)  as "Price 25.01-100"
,  sum(case when quoted_price between 100.01 and 250
        then quantity_ordered
        else 0 end)  as "Price 100.01- 250"
,  sum(case when quoted_price >  250
        then quantity_ordered
        else 0 end)  as "Price > 250"
,  sum(quantity_ordered) as "Tot Quant"
From a_oe.order_details
;
+---------------+-----------------+-------------------+-------------+-----------+
| Price 0.01-25 | Price 25.01-100 | Price 100.01- 250 | Price > 250 | Tot Quant |
+---------------+-----------------+-------------------+-------------+-----------+
|           240 |              83 |               142 |          56 |       521 |
+---------------+-----------------+-------------------+-------------+-----------+
```

Demo 08: A different layout for this query. The expressions in the Select and the Group By are identical.

```
Select
case
  when quoted_price between 0.01 and 25    then 'Price   0.01 -  25'
  when quoted_price between 25.01 and 100  then 'Price  25.01 - 100'
  when quoted_price between 100.01 and 250 then 'Price 100.01 - 250'
  when quoted_price >  250                 then 'Price over 250'
end as "Price Range"
,
sum(quantity_ordered) AS "Total Quantity"
From a_oe.order_details
```

```
group by case
      when quoted_price between 0.01 and 25    then 'Price   0.01 -  25'
      when quoted_price between 25.01 and 100  then 'Price  25.01 - 100'
      when quoted_price between 100.01 and 250 then 'Price 100.01 - 250'
      when quoted_price >  250                 then 'Price over 250'
      end
order by 1;
+--------------------+----------------+
| Price Range        | Total Quantity |
+--------------------+----------------+
| Price   0.01 -  25 |            240 |
| Price  25.01 - 100 |             83 |
| Price 100.01 - 250 |            142 |
| Price over 250     |             56 |
+--------------------+----------------+
```