Charla Steiner, CPA

linkedin.com/in/charlasteiner

SQL Query

SQL query using SQL window functions to create a MySQL database view of sales metrics to be used in reporting. The view is easily queried from Excel and Power BI, or saved to a CSV file. SQL window functions enable reporting over various time periods.

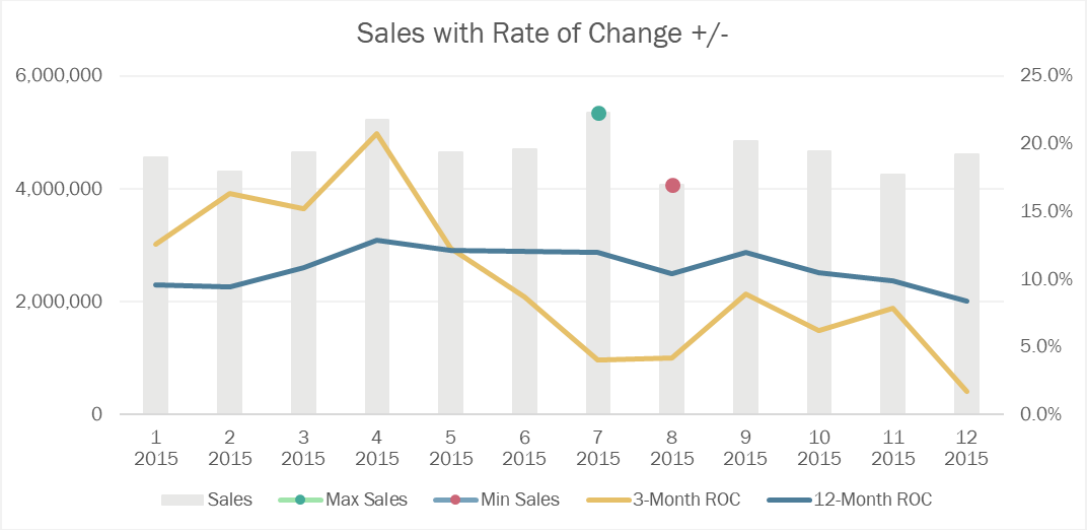
Calculated Metrics

Monthly sales over a two-year period

- Sales
- Year to date (YTD)
- Same period last year (SPLY)
- Last month (LM)
- 3-month moving average (3 MMA)
- 3-month moving total (3 MMT)
- 12-month moving average (12 MMA)
- 12-month moving total (12 MMT)
- Year over year (YOY)
- Year-over-year percent change (YOY %)
- Month over month (MOM)
- Month-over-month percent change (MOM %)
- Prior year year to date (PY YTD)
- Prior year year-to-date change (PY YTD Chg)
- Prior year year-to-date change percent (PY YTD Chg %)
- 3-month rate of change over comparable 3-month period 12 months ago (3-12 ROC)
- 12-month rate of change over comparable 12-month period 12 months ago (12-12 ROC)

Example Pivot Chart

Pivot chart created in Excel using DAX measures created in Power Pivot from the query metrics. Sliced to one year.



Query

```
/* MySQL database: WideWorldImporters, view: vsalesmetrics
Query to create view vsalesmetrics. The view has calculations of sales
metrics based on monthly sales over a two-year period.
*/
create or replace view vSalesMetrics as
with cte_productperiod                as
    (
        /* Create date/product combinations for all dates even if no sales
for a product on a given date so that measures are calculated correctly. Date
range starts prior to the reporting period to enable calculation of metrics
for comparable periods (YoY, MoM, etc.).
        */
        select
            s.stockitemkey            ,
            d.date                    ,
            d.`cy year`      calendar_year ,
            d.`cy month num` calendar_month
        from
            date d
            cross join
                stockitem s
        where
            d.date between '2013-01-01' and '2015-12-31'
            and exists
            (
                select *
                from
                    orders o
                where
                    o.stockitemkey = s.stockitemkey
                    and o.orderdatekey between '2013-01-01' and '2015-12-31'
            )
    )
    ,
    cte_base_calculations as
    (
        /* Calculate measures that will be used directly and/or used to
create other measures.
         */
        select
            row_number()
                over(order by p.calendar_year, p.calendar_month) row_id,
            p.calendar_year order_year  ,
            p.calendar_month order_month ,
            /* Sales current and prior periods */
            round(sum(coalesce(o.totalexcludingtax,0)), 0) sales ,
            round(sum(sum(coalesce(o.totalexcludingtax, 0)))
                over(partition by p.calendar_year
                    order by p.calendar_year, p.calendar_month rows between
                unbounded preceding and current row), 0) sales_ytd ,
            round(sum(sum(o.totalexcludingtax))
                over(order by p.calendar_year, p.calendar_month rows between
                12 preceding and 12 preceding),0) same_period_last_year ,
            round(sum(sum(o.totalexcludingtax))
                over(order by p.calendar_year, p.calendar_month rows between
                1 preceding and 1 preceding),0) last_month ,
```

```
        /* Moving totals and averages */
        case
            when count(*)
                over(order by p.calendar_year, p.calendar_month rows
                between
                    2 preceding and current row) = 3
            then round(avg(sum(o.totalexcludingtax))
                over(order by p.calendar_year, p.calendar_month rows
            between
                    2 preceding and current row),0)
            else null
        end sales_3_mma , -- 3-month moving average
        case
            when count(*)
                over(order by p.calendar_year, p.calendar_month rows
            between
                    2 preceding and current row) = 3
            then round(sum(sum(o.totalexcludingtax))
                over(order by p.calendar_year, p.calendar_month rows
            between
                    2 preceding and current row),0)
            else null
        end sales_3_mmt , -- 3-month moving total
        case
            when count(*)
                over(order by p.calendar_year, p.calendar_month rows
            between
                    11 preceding and current row) = 12
            then round(avg(sum(o.totalexcludingtax))
                over(order by p.calendar_year, p.calendar_month rows
            between
                    11 preceding and current row),0)
            else null
        end sales_12_mma , -- 12-month moving average
        case
            when count(*)
                over(order by p.calendar_year, p.calendar_month rows
            between
                    11 preceding and current row) = 12
            then round(sum(sum(o.totalexcludingtax))
                over(order by p.calendar_year, p.calendar_month rows
            between
                    11 preceding and current row),0)
            else null
        end sales_12_mmt -- 12-month moving total
    from
        cte_productperiod p
        left outer join
            orders o
            on
                p.date              = o.OrderDateKey
                and p.stockitemkey = o.stockitemkey
    group by
        p.calendar_year ,
        p.calendar_month
)
,
```

```
    cte_metrics as
    (
        select
            bc.*                                                    ,
            bc.sales - coalesce(bc.same_period_last_year, null) yoy     ,
            bc.sales /nullif(bc.same_period_last_year,0) - 1    yoy_pct ,
            bc.sales - coalesce(bc.last_month, null)            mom     ,
            bc.sales /nullif(bc.last_month,0) - 1               mom_pct ,
            /* Syntax error in MySQL when using a default value with lag.
Using coalesce() instead.
             */
            coalesce(lag(sales_ytd, 12)
                over(order by order_year, order_month), 0) sales_py_ytd ,
            sales_ytd - coalesce(lag(sales_ytd, 12)
                over(order by order_year, order_month), 0) sales_py_ytd_diff
            ,
            (sales_ytd - lag(sales_ytd, 12)
                over(order by order_year, order_month)) /
                nullif(coalesce(lag(sales_ytd, 12)
                    over(order by order_year, order_month), 0), 0)
                sales_py_ytd_diff_pct ,
            (sales_3_mmt  / lag(nullif(sales_3_mmt, 0), 12)
                over(order by order_year, order_month))-1 3_12_roc , -- 3/12
            sales rate of change (pct)
            (sales_12_mmt / lag(nullif(sales_12_mmt, 0), 12)
                over(order by order_year, order_month))-1 12_12_roc -- 12/12
            sales rate of change (pct)
        from
            cte_base_calculations bc
    )
select *
from
    cte_metrics
where
    order_year in (2014, 2015) -- reporting period
```