Analyzing When to Buy and Sell Stocks Using T-SQL in SQL Server

By: [Rick Dobson](https://www.mssqltips.com/sqlserverauthor/57/rick-dobson/)

FROM: <https://www.mssqltips.com/sqlservertip/5371/analyzing-when-to-buy-and-sell-stocks-using-tsql-in-sql-server/?utm_source=dailynewsletter&utm_medium=email&utm_content=headline&utm_campaign=20180228>

Problem

Please provide examples of how to use the data science techniques of segmentation and cross validation to improve and assess the performance of technical indicators, such as the MACD line indicator, for estimating profitable buy and sell dates for stock symbols.

Solution

[A prior tip](https://www.mssqltips.com/sqlservertip/5301/mining-stock-price-time-series-with-macd-in-sql-server/) introduced the MACD (Moving Average Convergence/Divergence) technical indicators and demonstrated how one of these indicators was able to designate buy and sell dates for ten stock symbols over an approximately four-year time span. As is common for technical indicators, the indicators did not always specify buy and sell dates that lead to a profit, but the MACD line indicator did specify buy and sell dates that on average resulted in profits. The prior tip was relevant to SQL Server professionals because the demonstration was for stock price data in a SQL Server database, and the process for designating buy and sell dates was based on code in a T-SQL script.

Modern-day technical analysis tools have been growing in popularity for stock market analysis since at least the 1970's - and the MACD indicators are arguably the tools with the greatest current interest. In contrast, the emergence of data science is more recent, and its practitioners tend to be computer professionals instead of stock market technical analysts. Both technical analysis and data science offer capabilities for analyzing and predicting time-series data, such as historical price and volume data for stock symbols. This tip is an introductory exploration of interfaces between technical analysis and data science which demonstrates the application of two data science techniques for enhancing the efficacy of MACD line estimates for when to buy and sell stocks. This tip is also of more general interest because it illustrates a framework for evaluating enhancements to any predictive model based on any data stored within a SQL Server database.

The first data science technique that this tip demonstrates is segmentation. With [segmentation](https://www.analyticsvidhya.com/blog/2016/02/guide-build-predictive-models-segmentation/), a data scientist can group data to yield better predictive outcomes. For this tip, data are grouped by the duration of a stock trade, different supplemental models are applied to enhance the MACD line estimates of when to sell stocks for optimizing the profit from a trade.

A second data science technique adapted within this tip is [cross validation](https://docs.aws.amazon.com/machine-learning/latest/dg/cross-validation.html). You can cull through and/or calibrate prospective model specifications that are initially tested on one set of data and then validated on another data set. Each of two or more data sets can successively and alternatively serve as a development sample and a validation sample. This tip uses a simple adaptation of cross validation by developing models on one data set and then just validating the models on another data set.

Another special benefit delivered by this tip is the release of a refreshed version of the AllNasdaqTickerPricesfrom2014into2017 database. This database has time-series data for a subset of NASDAQ exchange stocks.

An earlier version of the database was released along with [a preceding tip](https://www.mssqltips.com/sqlservertip/5248/mining-time-series-data-by-calculating-moving-averages-with-tsql-code-in-sql-server/) that focused on arithmetic moving averages; that version of the database includes historical price and volume data for NASDAQ stocks as well as arithmetic moving averages for stock symbol close prices.

The refreshed database released with this tip includes all tables from the initial release as well as tables for

exponential moving averages for NASDAQ stock symbol close prices

three MACD indicators for NASDAQ stocks

comparison tables that allow you to confirm how to contrast different models for time-series data

A quick review of the MACD indicators

An [introductory tip on MACD indicators](https://www.mssqltips.com/sqlservertip/5301/mining-stock-price-time-series-with-macd-in-sql-server/) defined three indicators and then drilled down on how to apply one of these indicators.

The MACD line indicator, sometimes just called the MACD, is typically defined as the 12-day exponential moving average less the 26-day exponential moving average for a time-series data item, such as a stock symbol's close price on successive trading days. This indicator has a centerline value of zero. When the MACD line indicator rises from below to above its centerline value, it is a good time to buy a stock so long as the MACD line value remains above its centerline value. This is because the short-term moving average is above a longer-term moving average. In other words, prices are going up.

The signal line indicator is typically defined as a 9-day exponential moving average of the MACD line indicator. Because the signal line is a moving average based on the MACD line indicator, the signal line value trails the MACD line value. As a result, when the MACD line value falls below the signal line value, the MACD line value is beginning to fall relative to its recent values. This is sometimes interpreted as indicating near term stock price falls even though the MACD line value may be above its centerline value of zero.

The MACD histogram indicator is the MACD line value less the signal line value. MACD histogram values are often plotted as histogram bars around the MACD centerline value.

Aside from confirming how to compute all three MACD indicators, the introductory MACD tip focused on how to derive buy and sell date recommendations for a stock symbol with the MACD line indicator.

A recommended buy date is where a MACD line value moves from below its centerline value of zero to above its centerline value.

A recommended sell date is the last date after a recommended buy date where the MACD line exceeds its centerline value.

This tip revisits these rules as it looks for ways of improving the realized profit associated with buy and sell dates. Ad hoc analyses of profits resulting from buy and sell recommendations created with the introductory tip on MACD indicators suggest two supplementary rules for enhancing profits.

One new rule emerges from an understanding that the MACD line value typically declines as it approaches its centerline value of zero from above. As a result, profit enhancements may be realized by selling a stock before the MACD line crosses its centerline value.

A second new rule takes advantage of the understanding that the MACD line decelerates growth as it passes from above to below its signal line. This deceleration can be a precursor to a period of declining prices. Therefore, a sell date based on the MACD line value falling below the signal line may help to preserve profits gained since a buy date.

Empirical ad hoc testing suggests these two new rules apply best to different types of buy and sell recommendations.

The falling of the MACD line value below the signal line value works best for buy and sell recommendation dates that have relatively fewer trading days between them.

The selling of a stock slightly before the MACD line falls below its centerline value works better for buy and sell recommendation dates that have relatively more trading days between them.

Neither of these rules offer any obvious advantage to recommended buy and sell dates that are just a few days apart.

A quick review of the buy and sell recommendations from the prior tip

There are ten stocks in the base sample; these stocks were studied for profitability in the introductory MACD tip. The following script shows centerline cross-over price percent change for the ten stock symbols from that tip.

There is a separate row in the result set for each symbol from the base sample.

The number of trades column indicates the number of trades that the centerline cross-over model specified for a symbol. While each trade for each symbol is characterized by a buy date and a sell date, these dates do not show in the result set because the query aggregates results across all trades for a symbol.

The first\_last\_change\_% column returns the average percent change between the first close price (close\_first) and last close price (close\_last) across each of the trades for a symbol.

The earlier\_close\_comparison\_report table in the [AllNasdaqTickerPricesfrom2014into2017 database](https://www.mssqltips.com/tipimages2/5371_AllNasdaqTickerPricesfrom2014into2017.zip) released with this tip has a separate row for each trade for every symbol in both the base and validation samples. This table is one of two comparison report tables for this tip. This tip includes separate sections on the creation and population of each of these tables later in the tip.

The database name reminds you that all trades are for historical price and volume data from as early as the first trading day in 2014.

The data extend through November 8, 2017 or earlier if the data pull from Yahoo Finance had no historical data for that date at the time of the data extraction.

The where clause excludes from the result set stock symbols in the validation sample. This where clause restricts the results to stock symbols from the base sample. The validation sample is based on a set of stock symbols not included in the introductory MACD tip.

The group by clause returns one row per symbol from the base sample.

-- centerline cross-over price percent change for all symbols in the base sample

-- where clause excludes cross validation sample

select

symbol

,count(\*) [number of trades]

,(sum(close\_last - close\_first)/avg(close\_first) \* 100) [first\_last\_change\_%]

from [AllNasdaqTickerPricesfrom2014into2017].[dbo].[earlier\_close\_comparison\_report]

where

symbol not in ('BZUN', 'ACXM', 'MMSI', 'CORT', 'MU', 'TIL')

group by symbol

The following screen shot displays the results from the preceding query.

The ten stock symbols in the result set denote the base sample stock symbols. All model development is performed with these ten stock symbols. Many readers may be familiar with Microsoft (MSFT), Amazon (AMZN) and PriceLine (PCLN). Other symbols are for less commonly known stocks, but all symbols are for widely traded stocks that have ample volume, so they can be readily traded.

The second column shows the number of trades specified by the MACD line centerline model. Because each trade has both a buy date and a sell date, there are twice as many dates as there are trades for each symbol.

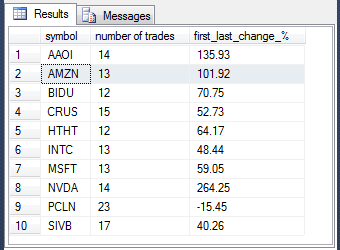
The third column returns the percent change from the buy price through to the sell price across all trades for a stock symbol.

For all except one stock symbol (PCLN), the average percent change is positive.

Three stock symbols have average percent change values of over 100 percent.

The median percent close price change is between 59.05 percent and 64.17 percent across all ten stock symbols.

Please keep in mind that these percent close price changes were achieved over the four-year span from 2014 through 2017.



The following set of four scripts shows more detail about the MSFT trades from perspectives that may help you to understand the segmentation model used for enhancing the profits from the centerline rule for specifying buy and sell dates.

The first script returns a separate row for each of the thirteen pairs of buy and sell dates for Microsoft (MSFT) stock.

The second script returns a separate row for each Microsoft stock trade that has fifty or more days between its buy date, or start date, and its sell date, or end date. Analysis of the base sample trades of fifty days or more shows them often benefitting from terminating the trade prior to reaching a sell date that matches the MACD line just before falling below its centerline value.

The third script returns a separate row for each Microsoft stock trade that is between four and forty-nine days in length. For these trades, the base sample analysis showed that profit could be improved consistently by exiting a trade after the MACD line fell below the signal line.

The fourth script is for Microsoft trades that have a duration of three days or less. These trades are characterized by a MACD line value that briefly popped above the centerline and then quickly returned below the centerline value. The empirical analysis of these base sample trades did not suggest a supplemental model that could improve their profitability.

-- summary of trades for MSFT

-- all trades for MSFT

select

start\_date

,end\_date

,symbol

,number\_of\_row\_numbers

,close\_first

,close\_last

,first\_last\_change

,[first\_last\_change\_%]

from [AllNasdaqTickerPricesfrom2014into2017].[dbo].[earlier\_close\_comparison\_report]

where symbol = 'MSFT'

-- MSFT trades of 50 days or longer

select

start\_date

,end\_date

,symbol

,number\_of\_row\_numbers

,close\_first

,close\_last

,first\_last\_change

,[first\_last\_change\_%]

from [AllNasdaqTickerPricesfrom2014into2017].[dbo].[earlier\_close\_comparison\_report]

where symbol = 'MSFT'

and number\_of\_row\_numbers >= 50

-- MSFT trades of 4 through 49 days

select

start\_date

,end\_date

,symbol

,number\_of\_row\_numbers

,close\_first

,close\_last

,first\_last\_change

,[first\_last\_change\_%]

from [AllNasdaqTickerPricesfrom2014into2017].[dbo].[earlier\_close\_comparison\_report]

where symbol = 'MSFT'

and number\_of\_row\_numbers > 3 and number\_of\_row\_numbers <= 49

-- MSFT trades of 3 days or less

select

start\_date

,end\_date

,symbol

,number\_of\_row\_numbers

,close\_first

,close\_last

,first\_last\_change

,[first\_last\_change\_%]

from [AllNasdaqTickerPricesfrom2014into2017].[dbo].[earlier\_close\_comparison\_report]

where symbol = 'MSFT'

and number\_of\_row\_numbers <= 3

The following screen shot shows the result sets from the four queries in the same order as the preceding scripts.

The top result set has a separate row for each of the 13 Microsoft trades designated by the centerline rule.

The start\_date value corresponds to the buy date for each trade, and the end\_date designates the sell date for each trade.

The symbol column is not strictly necessary in this result set because all trades are for a single symbol - MSFT. However, there are ten different symbols in the full base sample.

The number\_of\_row\_numbers column indicates the number of trading days from the buy date through the sell date for a trade. For example, the first listed trade is bought one trading day (January 31, 2014) and sold on the very next trading day (February 3, 2014). The number of calendar days is greater than two, but there are only two trading days from the buy date through to the sell date because there is no trading on weekend days (or stock market holidays either).

The close\_first and close\_last column values denote the close price on the buy and sell dates, respectively.

The first\_last\_change column is the close\_last column value less the close\_first column value. This value represents the total change in value of a single share in a stock from the buy date through to the sell date.

The first\_last\_change\_% column is the first\_last\_change column value divided by the close\_first column value. This column displays the percentage change between buy and sell dates relative to the buy price for a stock.

The second, third and fourth result sets correspond to the second, third, and fourth scripts from the preceding code listing. All columns for each of the result sets are defined in an identical way to the columns of the top result set. The only difference between the result sets is in the rows that appear in result set. Additionally, you should understand that the rows in the second, third, and fourth result sets are mutually exclusive to one another. That is, the data for a trade can exist in only one of these result sets.

The second result set is for trades that extend over fifty or more trading days.

There are five trades in this segment.

These trades all show a profit.

The profit percentage starts at a low of slightly over six percent up through slightly more than eighteen and a half percent.

The third result set is for trades that have a duration of four through forty-nine trading days.

There are also five trades in this segment.

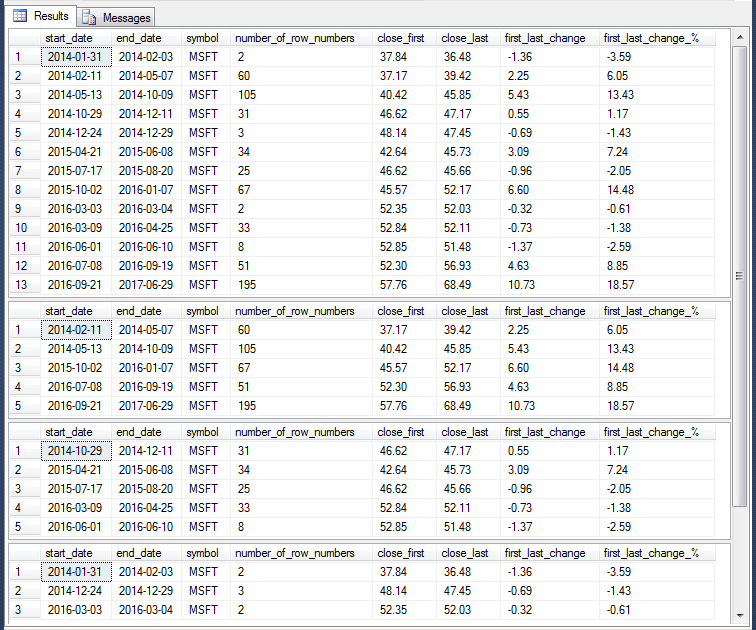
Three of the five trades have a loss percentage, and only two of the trades result in a profit.

The fourth result set is for trades that have only two or three trading days from the buy date through the sell date.

There are three trades in this segment.

All three trades result in a loss.

The losses are of relatively modest magnitude.



Creating and populating a table with trade outcomes from the earlier close model

Before assessing the success of the supplementary models, it is necessary to collect trade outcomes for analysis. The first model is named the earlier close model because it exits a trade by leaving out about the last ten percent of the trading days from the centerline cross-over model. Empirical testing confirmed that the earlier close model preserved more of the maximum gain attained by close prices for trades lasting fifty days or more than the sell date for the centerline cross-over model. Results confirming this finding are presented in the "Did the earlier close and macd\_below models improve profits?" section towards the end of this tip.

The following script can be used to generate and save the trade results for both the centerline cross-over model and the earlier close date model for a set of symbols. The trade results are archived in the earlier\_close\_comparison\_report table in the AllNasdaqTickerPricesfrom2014into2017 database. The script is available from the [close\_last\_vs\_close\_earlier\_comparer.sql](https://www.mssqltips.com/tipimages2/5371_AllNasdaqTickerPricesfrom2014into2017.zip) file, which is among the resources in the download for this tip.

Notice that the script starts with a default reference to the AllNasdaqTickerPricesfrom2014into2017 database. This reference is immediately followed by some commented code that creates a fresh copy of the earlier\_close\_comparison\_report table. The commented code is intended to be run just once for each time that you create and populate the table for a set of stock symbols.

Following the commented code for creating the earlier\_close\_comparison\_report table is a declaration statement for the @symbol local variable, which you can use to assign @symbol a value with a varchar data type having a maximum length of up to five characters. The script below assigns the string AAOI, but this value is to be changed in successive runs of the script.

At the end of each run of the script, the trades for the stock symbol with the current value of the @symbol local variable are inserted into the earlier\_close\_comparison\_report table.

In populating the earlier\_close\_comparison\_report table for this tip, the values for the @symbol variable were successively selected from the base sample (AAOI, AMZN, BIDU, CRUS, HTHT, INTC, MSFT, NVDA, PCLN, SIVB) and the validation sample (BZUN, ACXM, MMSI, CORT, MU, TIL).

After the preliminary steps described above, the main body of the script commences. The processing steps for populating the earlier\_close\_comparison\_report table for the current value of @symbol begin by freshly creating and populating the #macd\_indicators\_with\_row\_numbers\_for\_symbol temporary table. This temporary table is, in turn, based on the macd\_indicators table, which was created initially by a prior tip on MACD indicators. With the exception of row\_number, the columns are the same as in the macd\_indicators table; the column names are: symbol, date, close, row\_number, macd, signal, and macd\_histogram.

The next significant code block freshly creates and populates the #start\_end\_dates\_b4\_odd\_drop temporary table. The code for populating this table finds dates matching MACD line value transitions from below to above the centerline and just before the MACD line moves from above to not above the centerline. For this tip, we are only interested in beginning and ending dates for when the MACD line is above the centerline.

The data from the #start\_end\_dates\_b4\_odd\_drop temporary table are successively filtered and transformed until they end up in the #start\_end\_dates\_on\_single\_rows temporary table. This temporary table has a single row with start and end dates for each block of trading days in which the MACD line is above its centerline value. The key role for the table is to provide information for documenting the trades based on the centerline line model. Critical columns in the table include the following:

start\_date and end\_date values correspond to buy and sell dates, respectively, for the trade documented on a row

symbol for the current value of @symbol

number\_of\_row\_numbers for the number of trading days from the buy date through the sell date

close\_first for the close price on the buy date

close\_last for the close price on the sell date

close\_max for the maximum close price starting on the buy date through the sell date

Next, the #for\_close\_earlier\_dates\_and\_close\_values table is freshly created and populated. While the #start\_end\_dates\_on\_single\_rows table has a single row for each trade no matter what the duration of the trade, the code for the #for\_close\_earlier\_dates\_and\_close\_values table populates its rows only for trades with a duration of fifty or more days as originally specified by the centerline model. The key role of this table is to document the sell date and close price for trades specified by the close model. The columns from the #for\_close\_earlier\_dates\_and\_close\_values table include the following:

start\_date\_with\_close\_earlier is for the start date for a trade; this column value matches start\_date column value in the #start\_end\_dates\_on\_single\_rows table

end\_date\_with\_close\_earlier is for the date that a trade would end by the centerline model (derived from the #start\_end\_dates\_on\_single\_rows table); this column, along with the start\_date\_with\_close\_earlier column, facilitates matching rows between the start\_end\_dates\_on\_single\_rows table and the #for\_close\_earlier\_dates\_and\_close\_values table

symbol is for the current value of @symbol

ceiling\_lookup\_row is for the number of trading days in the revised trade based on closing the trade prior to the sell date for the centerline model

the ceiling\_lookup\_row value is equal to the ceiling function value of 90 percent of the trading days from the centerline model for a trade

the number\_of\_row\_numbers column makes available the number of trading days from the centerline model; its value is available in this temporary table for your easy reference and to help validate the join

The next two columns provide what could arguably be called the most critical values of the #for\_close\_earlier\_dates\_and\_close\_values table

the next two columns provide what could arguably be called the most critical values of the #for\_close\_earlier\_dates\_and\_close\_values table

the close\_earlier column contains the close price when the trade closes for the earlier close model

After the #for\_close\_earlier\_dates\_and\_close\_values table is freshly created and populated, it is left joined to the #start\_end\_dates\_on\_single\_rows table. The join result set is processed further to create metrics for documenting each trade. To facilitate the proper interpretation of these metrics, the columns of the joined result in the #for\_close\_earlier\_comparison\_report table are described briefly below.

start\_date is the buy date for a trade; this date is equally valid for both the centerline and the earlier close models

end\_date is the sell date for a trade in the centerline model

close\_earlier\_date can have one of two values

when the number\_of\_row\_numbers column value is greater than or equal to fifty, it is the sell date for the trade in the earlier close model

otherwise it is null

symbol is the current value for @symbol

number\_of\_row\_numbers is the number of trading days derived from the centerline model

close\_first is the close price at the buy date

close\_last is the close price at the sell date from the centerline model

close\_earlier can have one of two values

when the number\_of\_row\_numbers column value is greater than or equal to fifty, it is the close\_earlier column value from the result set from the #for\_close\_earlier\_comparison\_report table

otherwise, it is close\_last

close\_max is the maximum close price from the buy date through the sell date based on the centerline model; this column and associated columns related to it are not a central point of interest for this tip, but some developers may find it of value to know the maximum profit potential from a trade

first\_last\_change is close\_last less close\_first; this quantity represents the profit or loss associated with the trade based on the centerline model

first\_last\_change\_% is first\_last\_change expressed as a percentage of close\_first

first\_earlier\_change is close\_earlier less close\_first; this quantity represents the profit or loss associated with the trade based on the earlier close model

first\_ealier\_change\_% is first\_earlier\_change expressed as a percentage of close\_first

first\_max\_change is close\_max less close\_first; this quantity represents the profit or loss from the centerline model if the trade was ended on the date when the profit was at its maximum value

first\_max\_change\_% is first\_max\_change expressed as a percentage of close\_first

After the #for\_close\_earlier\_comparison\_report table is freshly created and populated for the current value of @symbol, two remaining steps complete the script.

The result set for the #for\_close\_earlier\_comparison\_report table is displayed via a select statement.

Then, the #for\_close\_earlier\_comparison\_report table result set is inserted into the earlier\_close\_comparison\_report table. Completing this step for each ticker symbol in the base and validation samples builds a history of the trades that can be used to assess the efficacy of the early close model versus the centerline model.

use [AllNasdaqTickerPricesfrom2014into2017]

go

/\*

-- run once before invoking the script below

-- for which to collect symbol comparison tables

-- for a set of symbols

begin try

drop table [dbo].[earlier\_close\_comparison\_report]

end try

begin catch

print 'earlier\_close\_comparison\_report not available to drop'

end catch

create table [dbo].[earlier\_close\_comparison\_report](

[start\_date] [date] NULL,

[end\_date] [date] NULL,

[close\_earlier\_date] [date] NULL,

[symbol] [varchar](10) NULL,

[number\_of\_row\_numbers] [bigint] NULL,

[close\_first] [money] NULL,

[close\_last] [money] NULL,

[close\_earlier] [money] NULL,

[close\_max] [money] NULL,

[first\_last\_change] [money] NULL,

[first\_last\_change\_%] [money] NULL,

[first\_earlier\_change] [money] NULL,

[first\_earlier\_change\_%] [money] NULL,

[first\_max\_change] [money] NULL,

[first\_max\_change\_%] [money] NULL

) ON [PRIMARY]

go

\*/

declare @symbol varchar(5) = 'AAOI'

-- macd\_indicators extract with symbol row\_number

begin try

drop table #macd\_indicators\_with\_row\_numbers\_for\_symbol

end try

begin catch

print '#macd\_indicators\_with\_row\_numbers\_for\_symbol not available to drop'

end catch

select [symbol]

,[date]

,[close]

,row\_number() over (order by date) row\_number

,[macd]

,[signal]

,[macd\_histogram]

into #macd\_indicators\_with\_row\_numbers\_for\_symbol

from [AllNasdaqTickerPricesfrom2014into2017].[dbo].[macd\_indicators]

where

symbol = @symbol

order by [date]

--select \* from #macd\_indicators\_with\_row\_numbers\_for\_symbol order by [date]

-------------------------------------------------------------------------------------

-- processing code to derive start and end dates

-- for macd centerline cross overs

-- get start and end dates for successive, alternating macd blocks

-- before removing a trailing start block with no matching end block

begin try

drop table #start\_end\_dates\_b4\_odd\_drop

end try

begin catch

print '#start\_end\_dates\_b4\_odd\_drop not available to drop'

end catch

-- find all starts and ends for centerline cross-overs

select

\*

into #start\_end\_dates\_b4\_odd\_drop

from

(

select

[date]

,symbol

,[close]

,[row\_number]

,macd

,lead(macd,1) over (order by row\_number) next\_macd

,sign(macd) sign\_of\_macd

,sign(lead(macd,1) over (order by row\_number)) sign\_of\_next\_macd

,sign(lag(macd,1) over (order by row\_number)) sign\_of\_prior\_macd

,

case

when row\_number = 3

and sign(macd) = 1 then 'start\_pos'

when sign(macd) = 1

and sign(lag(macd,1) over (order by row\_number)) = -1 then 'start\_pos'

when

sign(macd) != sign(lead(macd,1) over (order by row\_number))

and sign(macd) = 1 then 'end\_pos'

when

sign(macd) != sign(lead(macd,1) over (order by row\_number))

and sign(macd) != 1 then 'end\_not\_pos'

when row\_number = 3

and sign(macd) = -1 then 'start\_not\_pos'

when

sign(macd) != 1

and sign(macd) = sign(lead(macd,1) over (order by row\_number))

and sign(lead(macd,1) over (order by row\_number)) != 1

and sign(lag(macd,1) over (order by row\_number)) = 1 then 'start\_not\_pos'

else NULL

end macd\_pos\_not\_pos

from #macd\_indicators\_with\_row\_numbers\_for\_symbol

where

row\_number > = 3

) for\_is\_not\_start\_or\_end

where macd\_pos\_not\_pos is not null

-- delete row with start\_or\_end equal to prior\_start\_or\_end

-- gets rid of rows not continued with another matching pos or not pos\_row

begin try

drop table #start\_end\_dates\_b4\_odd\_drop\_with\_start\_or\_end

end try

begin catch

print '#start\_end\_dates\_b4\_odd\_drop\_with\_start\_or\_end not available to drop'

end catch

-- add start\_or\_end and prior\_start\_or\_end columns

-- to #start\_end\_dates\_b4\_odd\_drop\_with\_start\_or\_end

-- from #start\_end\_dates\_b4\_odd\_drop

select

\*

,

case

when left(macd\_pos\_not\_pos,5) = 'start' then 'start'

when left(macd\_pos\_not\_pos,3) = 'end' then 'end'

end start\_or\_end

,

case

when left(lag(macd\_pos\_not\_pos,1) over(order by row\_number),5) = 'start' then 'start'

when left(lag(macd\_pos\_not\_pos,1) over (order by row\_number),3) = 'end' then 'end'

end prior\_start\_or\_end

into #start\_end\_dates\_b4\_odd\_drop\_with\_start\_or\_end

from #start\_end\_dates\_b4\_odd\_drop

-- select \* from #start\_end\_dates\_b4\_odd\_drop

delete from #start\_end\_dates\_b4\_odd\_drop\_with\_start\_or\_end where start\_or\_end = prior\_start\_or\_end

-- restore #start\_end\_dates\_b4\_odd\_drop columns

-- after fix for contiguous duplicate start\_or\_end values

-- where start and end dates are the same

GO

begin try

drop table #start\_end\_dates\_b4\_odd\_drop

end try

begin catch

print '#start\_end\_dates\_b4\_odd\_drop not available to drop'

end catch

select

[date]

,symbol

,[close]

,[row\_number]

,macd

,next\_macd

,sign\_of\_macd

,sign\_of\_next\_macd

,sign\_of\_prior\_macd

,macd\_pos\_not\_pos

into #start\_end\_dates\_b4\_odd\_drop

from #start\_end\_dates\_b4\_odd\_drop\_with\_start\_or\_end

-- eliminate a start date without a matching end date

declare @date\_for\_row\_to\_delte date =

(select top 1 [date] from #start\_end\_dates\_b4\_odd\_drop order by date desc)

-- delete odd-row at end, if there is a start date without a matching end date

if (select count(\*) % 2 from #start\_end\_dates\_b4\_odd\_drop) = 1

begin

delete from #start\_end\_dates\_b4\_odd\_drop where date = @date\_for\_row\_to\_delte

end

begin try

drop table #start\_end\_dates

end try

begin catch

print '#start\_end\_dates not available to drop'

end catch

-- save cleaned start and end dates

select \* into #start\_end\_dates from #start\_end\_dates\_b4\_odd\_drop

-- select \* from #start\_end\_dates

--------------------------------------------------------------------------------------------

-- macd centerline cross-over start and end dates

-- on single rows with selected columns

begin try

drop table #start\_end\_dates\_on\_single\_rows

end try

begin catch

print '#start\_end\_dates\_on\_single\_rows not available to drop'

end catch

select

start\_date

,end\_date

,symbol

,[close]

,macd

,number\_of\_row\_numbers

,(

select

top 1 [close] close\_first

from #macd\_indicators\_with\_row\_numbers\_for\_symbol

where date >= [start\_date] and date <= [end\_date] order by [date]

) close\_first

,(

select

top 1 [close] close\_last

from #macd\_indicators\_with\_row\_numbers\_for\_symbol

where date >= [start\_date] and date <= [end\_date]

order by date desc

) close\_last

,(

select

max([close]) close\_max

from #macd\_indicators\_with\_row\_numbers\_for\_symbol

where date >= [start\_date] and date <= [end\_date]

) close\_max

into #start\_end\_dates\_on\_single\_rows

from

(

select

start\_date

,end\_date

,symbol

,[close]

,macd

,number\_of\_row\_numbers

from

(

select

macd\_pos\_not\_pos

,[date] start\_date

,lead([date],1) over (order by date) end\_date

,symbol

,[close]

,macd

,row\_number start\_row\_number

,lead(row\_number,1) over (order by date) end\_row\_number

,lead(row\_number,1) over (order by date)-row\_number + 1 number\_of\_row\_numbers

from #start\_end\_dates

where macd\_pos\_not\_pos in ('start\_pos', 'end\_pos')

--where macd\_pos\_not\_pos = 'start\_pos'

) for\_start\_date\_end\_date

where macd\_pos\_not\_pos = 'start\_pos'

) for\_close\_prices\_in\_dates

-- select \* from #start\_end\_dates\_on\_single\_rows

---------------------------------------------------------------------------------------------

-- generate new earlier close dates with matching close prices

-- before last close dates and prices for macd centerline cross-overs

-- lasting 50 days or longer

begin try

drop table #for\_close\_earlier\_dates\_and\_close\_values

end try

begin catch

print '#for\_close\_earlier\_dates\_and\_close\_values not available to drop'

end catch

select

start\_date [start\_date\_with\_close\_earlier]

,end\_date [end\_date\_with\_close\_earlier]

,symbol

,number\_of\_row\_numbers

,ceiling(number\_of\_row\_numbers\*.9) ceiling\_lookup\_row

,

(

select [close] from #macd\_indicators\_with\_row\_numbers\_for\_symbol where row\_number =

(

-- returns row\_number from #macd\_indicators\_with\_row\_numbers\_for\_symbol for new close value

select (row\_number - (number\_of\_row\_numbers - ceiling(number\_of\_row\_numbers\*.9)))

from #macd\_indicators\_with\_row\_numbers\_for\_symbol

where [date] = end\_date

)

) close\_earlier

,

(

select [date] from #macd\_indicators\_with\_row\_numbers\_for\_symbol where row\_number =

(

-- returns row\_number from #macd\_indicators\_with\_row\_numbers\_for\_symbol for new close value

select (row\_number - (number\_of\_row\_numbers - ceiling(number\_of\_row\_numbers\*.9)))

from #macd\_indicators\_with\_row\_numbers\_for\_symbol

where [date] = end\_date

)

) close\_earlier\_date

into #for\_close\_earlier\_dates\_and\_close\_values

from #start\_end\_dates\_on\_single\_rows

where number\_of\_row\_numbers >= 50

-- select \* from #for\_close\_earlier\_dates\_and\_close\_values

-- generate a report for macd centerline cross-overs and

-- compare buy/sell profits based on close\_first versus

-- close\_last, close\_earlier, and close\_max

begin try

drop table #for\_close\_earlier\_comparison\_report

end try

begin catch

print '#for\_close\_earlier\_comparison\_report not available to drop'

end catch

select

start\_date

,end\_date

,close\_earlier\_date

,symbol

,number\_of\_row\_numbers

,close\_first

,close\_last

,

case

when close\_earlier is not null then close\_earlier

else close\_last

end close\_earlier

,close\_max

,first\_last\_change

,[first\_last\_change\_%]

,

case

when first\_earlier\_change is not null then first\_earlier\_change

else first\_last\_change

end first\_earlier\_change

,

case

when [first\_earlier\_change\_%] is not null then [first\_earlier\_change\_%]

else [first\_last\_change\_%]

end [first\_earlier\_change\_%]

,first\_max\_change

,[first\_max\_change\_%]

into #for\_close\_earlier\_comparison\_report

from

(

select

start\_date

,end\_date

,close\_earlier\_date

,#start\_end\_dates\_on\_single\_rows.symbol

,#start\_end\_dates\_on\_single\_rows.number\_of\_row\_numbers

,close\_first

,close\_last

,close\_earlier

,close\_max

,close\_last - close\_first [first\_last\_change]

,round(((close\_last - close\_first)/(close\_first)\*100),2) [first\_last\_change\_%]

,close\_earlier - close\_first [first\_earlier\_change]

,round(((close\_earlier - close\_first)/(close\_first)\*100),2) [first\_earlier\_change\_%]

,close\_max - close\_first [first\_max\_change]

,round(((close\_max - close\_first)/(close\_first)\*100),2) [first\_max\_change\_%]

from #start\_end\_dates\_on\_single\_rows

left join #for\_close\_earlier\_dates\_and\_close\_values

on

#start\_end\_dates\_on\_single\_rows.start\_date = #for\_close\_earlier\_dates\_and\_close\_values.start\_date\_with\_close\_earlier

and

#start\_end\_dates\_on\_single\_rows.end\_date = #for\_close\_earlier\_dates\_and\_close\_values.end\_date\_with\_close\_earlier

) for\_nulls\_out

-- generate a report for macd centerline cross-overs and

-- compare buy/sell profits based on close\_first versus

-- close\_last, close\_earlier, and close\_max

select

\*

from #for\_close\_earlier\_comparison\_report

-- populate earlier\_close\_comparison\_report for current symbol

insert into earlier\_close\_comparison\_report

select

\*

from #for\_close\_earlier\_comparison\_report

Creating and populating a table with trade outcomes from the macd\_below model

The next script listing shows the code for saving a history of trades based on the model that closes a trade when the MACD descends below signal line after initially rising above its centerline value. This script is available from the [close\_last\_vs\_macd\_line\_signal\_cross-over\_comparer.sql](https://www.mssqltips.com/tipimages2/5371_AllNasdaqTickerPricesfrom2014into2017.zip) file in the download for this tip. The model built by the code described in this section is called the macd\_below model to distinguish it from the earlier close model described in the preceding section.

The code for this section is very much like that for the preceding section. Differences between the code for this section versus the preceding one include the following.

The creation and use of a new table to store the history of trades (macd\_below\_signal\_comparison\_report in this section versus earlier\_close\_comparison\_report in the preceding section).

This section additionally shows new code that captures the move of the MACD below its signal line after the MACD crosses from below to above its centerline value. It also demonstrates how to document these trades after their buy and sell dates are identified so they can be saved in the macd\_below\_signal\_comparison\_report table.

Aside from these two distinctions noted above, the code between the two sections is identical. For that reason, a large block of code for developing content for populating the macd\_below\_signal\_comparison\_report table is not shown in this section. The excluded code begins at the declare statement for the @symbol variable and ends with the code for freshly creating and populating the #start\_end\_dates\_on\_single\_rows table. Recall that this table contains buy and sell dates along with selected other trade data for the centerline cross-over model. The excluded code resides in the T-SQL script file for this section. If you feel the need to review the excluded code from the listing in this section, you can examine it from the preceding section or by opening the T-SQL script file for this section.

Here is the code to create the table for storing the history of trades based on the MACD line value descending below its signal line.

The primary difference between this table and the one from the preceding section is the table name and names of selected columns.

The main advantage of the table is that it provides a container for holding trade results from the macd\_below model that is separate and distinct from the container for the earlier close model results.

As with the earlier\_close\_comparison\_report table from the preceding section, this table can accumulate trade results for a succession of different trading symbols. The code in this section enables the recording of transfer of trade results for the same ten base-sample symbols and six validation-symbol symbols as in the preceding section.

Running the code in both sections as described generates comparable trade results by two different supplemental rules for the centerline cross-over model.

Code and selected output in the next section demonstrate how to use the results from the macd\_below\_signal\_comparison\_report and earlier\_close\_comparison\_report tables to assess the efficacy of the centerline model and its two supplemental models.

use [AllNasdaqTickerPricesfrom2014into2017]

go

-- run once before invoking the script below

-- for which to collect symbol comparison tables

-- for a set of symbols

begin try

drop table [dbo].[macd\_below\_signal\_comparison\_report]

end try

begin catch

print 'macd\_below\_signal\_comparison\_report not available to drop'

end catch

create table [dbo].[macd\_below\_signal\_comparison\_report](

[start\_date] [date] NULL,

[end\_date] [date] NULL,

[close\_macd\_below\_date] [date] NULL,

[symbol] [varchar](10) NULL,

[number\_of\_row\_numbers] [bigint] NULL,

[close\_first] [money] NULL,

[close\_last] [money] NULL,

[close\_macd\_below] [money] NULL,

[close\_max] [money] NULL,

[first\_last\_change] [money] NULL,

[first\_last\_change\_%] [money] NULL,

[first\_macd\_below\_change] [money] NULL,

[first\_macd\_below\_change\_%] [money] NULL,

[first\_max\_change] [money] NULL,

[first\_max\_change\_%] [money] NULL

) ON [PRIMARY]

go

The next block of code for review in this section starts right after the fresh creation and population of the #start\_end\_dates\_on\_single\_rows table. Recall that this table contains buy and sell dates for the stock denoted by the current value of @symbol along with close\_first, close\_last, and close\_max values based on the centerline cross-over model.

The first step after the creation and population of the #start\_end\_dates\_on\_single\_rows table is to left join a subquery named macd\_and\_signal\_vals to the table. The values of the left joined result set are saved in the #start\_end\_dates\_closes\_line\_below\_signal table. The subquery is based on the macd\_indicators table from the AllNasdaqTickerPricesfrom2014into2017 database. The join assigns all MACD and signal line values for each trading day from the first buy date through the last sell date for all trades for the symbol currently being processed. The subquery additionally adds a column named line\_below\_signal.

It is common in the technical analysis literature to require a confirmation before accepting a move as indicating a trend, such as the MACD moving below the signal line. In the script below, the second day of two consecutive trading days in which the MACD is below the signal line is the confirmation that the following script uses. The code for populating the #start\_end\_dates\_closes\_line\_below\_signal table enables a subsequent search for confirmed MACD values below signal line values by assigning a value of - 1 to each line\_below\_signal column value where the MACD value is below signal line value.

The line\_below\_signal column values are later processed to find the first confirmed trading day for each trade when the MACD value falls below the signal line. This step along with numerous other steps is implemented in the code to create and populate the #for\_macd\_below\_comparison\_report table. Another critical role of the code to create and populate the #for\_macd\_below\_comparison\_report table is to compute values for the following four fields: close\_macd\_below\_date, close\_macd\_below, first\_macd\_below\_change, and first\_macd\_below\_change\_% . Here's the definition for each of these fields. All four of these fields have conditional values.

The close\_macd\_below\_date column value is the confirmed date when the MACD line falls below the signal line for the first time in trade and the number of trading days for a trade is appropriate for the macd\_below model.

When the number\_of\_row\_numbers column value from the #start\_end\_dates\_closes\_line\_below\_signal table is greater than three and less than or equal to forty-nine, then its value is the confirmed trading date on which MACD first falls below the signal line in a trade.

Otherwise, its value is null.

The close\_macd\_below column value is the close value on trading dates when there is a non-null close\_macd\_below\_date value. Otherwise, this column value is the close value from the centerline cross-over model.

The first\_macd\_below\_change column value is the profit or loss associated with a trade.

When there is a non-null close\_macd\_below\_date value, this value is computed as the close\_macd\_below value less the close\_first value from the #start\_end\_dates\_closes\_line\_below\_signal table.

Otherwise, the first\_macd\_below\_change column value is close\_last less close\_first from the #start\_end\_dates\_closes\_line\_below\_signal table.

The first\_macd\_below\_change\_% is the first\_macd\_below\_change column value expressed as a percentage of the close\_first value from the #start\_end\_dates\_closes\_line\_below\_signal table.

These four critical macd\_below columns are used in the #for\_macd\_below\_comparison\_report table similarly to the way comparable close earlier columns are used in the #for\_close\_earlier\_comparison\_report table within the preceding section.

After the #for\_macd\_below\_comparison\_report table is freshly created and populated for the current value of @symbol, two remaining steps complete the script.

The result set for the #for\_macd\_below\_comparison\_report table is displayed via a select statement.

Then, the #for\_macd\_below\_comparison\_report table result set is inserted into macd\_below\_signal\_comparison\_report table. As you complete this step for each ticker symbol in the base and validation samples, you will build a history of the trades that can be used to assess the efficacy of trades based on the early close model versus the centerline model.

-- create #start\_end\_dates\_closes\_line\_below\_signal as

-- left join of macd\_and\_signal\_vals query to

-- #start\_end\_dates\_on\_single\_rows temp table

-- and computed line\_below\_signal value

begin try

drop table #start\_end\_dates\_closes\_line\_below\_signal

end try

begin catch

print '#start\_end\_dates\_closes\_line\_below\_signal not available to drop'

end catch

select

#start\_end\_dates\_on\_single\_rows.[start\_date]

,#start\_end\_dates\_on\_single\_rows.end\_date

,macd\_and\_signal\_vals.date

,#start\_end\_dates\_on\_single\_rows.symbol

,#start\_end\_dates\_on\_single\_rows.[close]

,#start\_end\_dates\_on\_single\_rows.number\_of\_row\_numbers

,macd\_and\_signal\_vals.macd

,macd\_and\_signal\_vals.signal

,SIGN(macd\_and\_signal\_vals.macd - macd\_and\_signal\_vals.signal) line\_below\_signal

into #start\_end\_dates\_closes\_line\_below\_signal

from #start\_end\_dates\_on\_single\_rows

left join

(

-- macd and signal values for a symbol

select distinct

[macd\_indicators].[symbol]

,[macd\_indicators].[date]

,[macd\_indicators].[macd]

,[macd\_indicators].[signal]

from [AllNasdaqTickerPricesfrom2014into2017].[dbo].[macd\_indicators]

where

symbol = (select distinct symbol from #start\_end\_dates\_on\_single\_rows)

and signal is not null

) macd\_and\_signal\_vals

on #start\_end\_dates\_on\_single\_rows.symbol = macd\_and\_signal\_vals.symbol

where

date >= #start\_end\_dates\_on\_single\_rows.start\_date and date <= #start\_end\_dates\_on\_single\_rows.end\_date

order by #start\_end\_dates\_on\_single\_rows.start\_date

-- select \* from #start\_end\_dates\_closes\_line\_below\_signal

-- for change and change% computations for macd centerline

-- cross-overs for comparing buy/sell profits based on

-- close\_first versus

-- close\_last, close\_macd\_below, and close\_max

begin try

drop table #for\_macd\_below\_comparison\_report

end try

begin catch

print '#for\_macd\_below\_comparison\_report not available to drop'

end catch

select

for\_computed\_change\_and\_change\_percent\_fields.[start\_date]

,for\_computed\_change\_and\_change\_percent\_fields.end\_date

,for\_computed\_change\_and\_change\_percent\_fields.close\_macd\_below\_date

,for\_computed\_change\_and\_change\_percent\_fields.symbol

,for\_computed\_change\_and\_change\_percent\_fields.number\_of\_row\_numbers

,for\_computed\_change\_and\_change\_percent\_fields.close\_first

,for\_computed\_change\_and\_change\_percent\_fields.close\_last

,

case

when for\_computed\_change\_and\_change\_percent\_fields.close\_macd\_below\_date

is not null and number\_of\_row\_numbers < 50 and number\_of\_row\_numbers > 3

then for\_computed\_change\_and\_change\_percent\_fields.close\_macd\_below

else for\_computed\_change\_and\_change\_percent\_fields.close\_last

end close\_macd\_below

,for\_computed\_change\_and\_change\_percent\_fields.close\_max

,close\_last - close\_first first\_last\_change

,(close\_last - close\_first)/close\_first\*100 [first\_last\_change\_%]

,

case

when for\_computed\_change\_and\_change\_percent\_fields.close\_macd\_below\_date

is not null and number\_of\_row\_numbers < 50 and number\_of\_row\_numbers > 3

then for\_computed\_change\_and\_change\_percent\_fields.close\_macd\_below

else for\_computed\_change\_and\_change\_percent\_fields.close\_last

end

-

close\_first

first\_macd\_below\_change

,

(

case

when for\_computed\_change\_and\_change\_percent\_fields.close\_macd\_below\_date

is not null and number\_of\_row\_numbers < 50 and number\_of\_row\_numbers > 3

then for\_computed\_change\_and\_change\_percent\_fields.close\_macd\_below

else for\_computed\_change\_and\_change\_percent\_fields.close\_last

end

-

close\_first

)

/

close\_first\*100 [first\_macd\_below\_change\_%]

,close\_max - close\_first first\_max\_change

,(close\_max - close\_first)/close\_first\*100 [first\_max\_change\_%]

into #for\_macd\_below\_comparison\_report

from

(

select

#start\_end\_dates\_on\_single\_rows.[start\_date]

,#start\_end\_dates\_on\_single\_rows.end\_date

,for\_close\_macd\_below\_for\_start\_date\_end\_date\_groups.min\_date close\_macd\_below\_date

,#start\_end\_dates\_on\_single\_rows.symbol

,#start\_end\_dates\_on\_single\_rows.number\_of\_row\_numbers

,#start\_end\_dates\_on\_single\_rows.close\_first

,#start\_end\_dates\_on\_single\_rows.close\_last

,(

select [close]

from macd\_indicators

where [date] = for\_close\_macd\_below\_for\_start\_date\_end\_date\_groups.min\_date

and symbol = #start\_end\_dates\_on\_single\_rows.symbol

) close\_macd\_below

,#start\_end\_dates\_on\_single\_rows.close\_max

from #start\_end\_dates\_on\_single\_rows

left join

(

-- min date for line\_below\_signal\_indicator = -2

-- within start\_date...end\_date blocks

select

[start\_date]

,[end\_date]

,min([date]) min\_date

from

(

-- #start\_end\_dates\_closes\_line\_below\_signal with

-- computed line\_below\_signal\_indicator

select

\*

,line\_below\_signal + lag(line\_below\_signal,1) over (order by [date]) line\_below\_signal\_indicator

from #start\_end\_dates\_closes\_line\_below\_signal

group by [start\_date], end\_date, [date], symbol, [close], number\_of\_row\_numbers, macd, signal, line\_below\_signal

) for\_first\_confirmed\_line\_below\_indicator\_within\_start\_date\_end\_date

where line\_below\_signal\_indicator = -2

group by

[start\_date]

,[end\_date]

,line\_below\_signal\_indicator

) for\_close\_macd\_below\_for\_start\_date\_end\_date\_groups

on #start\_end\_dates\_on\_single\_rows.start\_date = for\_close\_macd\_below\_for\_start\_date\_end\_date\_groups.start\_date

) for\_computed\_change\_and\_change\_percent\_fields

-- generate a report for macd centerline cross-overs and

-- compare buy/sell profits based on close\_first versus

-- close\_last, close\_earlier, and close\_max

select

\*

from #for\_macd\_below\_comparison\_report

-- populate macd\_below\_signal\_comparison\_report for current symbol

insert into macd\_below\_signal\_comparison\_report

select

\*

from #for\_macd\_below\_comparison\_report

Did the earlier close and macd\_below models improve profits?

An analysis shown previously in this tip confirmed that the centerline cross-over model returned a median percent profit of between 59.05 percent and 64.17 percent across all ten stock symbols used in a prior tip. The current tip examines two alternative models for supplementing the centerline cross-over model. The earlier close model targeted trades that lasted fifty days or longer, and the macd\_below model aimed to improve trades lasting between four and forty-nine days.

Did the two alternative models provide enhanced profits for the base sample of ten stock symbols and was their effect also reflected in the validation sample of six additional stock symbols?

Furthermore, were alternative model effects demonstrated at the level of individual stock symbols as well as overall in either the set of base symbols and validation symbols?

The following script generates a result set that shows the earlier close and macd\_below model effects in both the base and validation sample. There are three parts to the script.

The first part of the script returns a result set that shows earlier close effects in both the base sample with ten stock symbols and the validation sample with six additional stock symbols.

The results are derived from the earlier\_close\_comparison\_report table. This table includes a row for each trade that reflects both the centerline cross-over model effect and the effect of the centerline cross-over model supplemented by earlier close model.

Results are shown separately for the base sample and validation sample. A where clause enables this capability with a not in operator or an in operator. Results are pulled for the base sample with the not in operator and for the validation sample with the in operator.

A union operator concatenates the results set from two separate select statements - one select statement pulls results for the base sample and a second select statement pulls results for the validation sample.

Within each select statement in the first part, a where clause filters for trades that are fifty days or longer. In this way, the first part restricts its result set to trades for which the earlier close model was applied.

The second part of the script has a similar design, but this part displays results for the macd\_below model. This second part of the script is almost identical to the first part except for two issues.

First, the from clause specifies the macd\_below\_signal\_comparison\_report table as the source for its result set.

Second, the where clause specifying duration of a trade targets trades lasting more than three days and less than fifty days.

The third part of the script has four, instead of just two, separate select statements that have their results concatenated by union operators.

Two of the four select statements designate the earlier\_close\_comparison\_report table as their source.

The two remaining select statements designate the macd\_below\_signal\_comparison\_report table as their source.

Within these two pair of select statements, the query design is similar to the queries for showing the earlier close and macd\_below model effects, except for the where clause about the duration of the trade. Both pair of select statements target trades with a duration of three days or less.

-- earlier close and macd\_below model effects

-- compared for base sample versus validation sample

-- earlier close effect size in base sample versus validation sample

select

'base sample' [sample name]

,count(\*) [number of trades]

,(sum(close\_last - close\_first)/avg(close\_first) \* 100) [centerline crossover]

,(sum(close\_earlier - close\_first)/avg(close\_first) \* 100) [centerline crossover after earlier close change fix]

,((sum(close\_earlier - close\_first) - sum(close\_last - close\_first))/avg(close\_first) \* 100) [total earlier close change effect fix size]

,(((sum(close\_earlier - close\_first) - sum(close\_last - close\_first))/avg(close\_first) \* 100))/count(\*) [average earlier close change effect fix size]

from [AllNasdaqTickerPricesfrom2014into2017].[dbo].[earlier\_close\_comparison\_report]

where

number\_of\_row\_numbers >= 50

and

symbol not in ('BZUN', 'ACXM', 'MMSI', 'CORT', 'MU', 'TIL')

union

select

'validation sample' [sample name]

,count(\*) [number of trades]

,(sum(close\_last - close\_first)/avg(close\_first) \* 100) [centerline crossover]

,(sum(close\_earlier - close\_first)/avg(close\_first) \* 100) [centerline crossover after earlier close change fix]

,((sum(close\_earlier - close\_first) - sum(close\_last - close\_first))/avg(close\_first) \* 100) [total earlier close change effect fix size]

,(((sum(close\_earlier - close\_first) - sum(close\_last - close\_first))/avg(close\_first) \* 100))/count(\*) [average earlier close change effect fix size]

from [AllNasdaqTickerPricesfrom2014into2017].[dbo].[earlier\_close\_comparison\_report]

where

number\_of\_row\_numbers >= 50

and

symbol in ('BZUN', 'ACXM', 'MMSI', 'CORT', 'MU', 'TIL')

-- macd\_below effect size in base sample versus validation sample

select

'base sample' [sample name]

,count(\*) [number of trades]

,(sum(close\_last - close\_first)/avg(close\_first) \* 100) [centerline crossover]

,(sum(close\_macd\_below - close\_first)/avg(close\_first) \* 100) [centerline crossover after macd\_below change fix]

,((sum(close\_macd\_below - close\_first) - sum(close\_last - close\_first))/avg(close\_first) \* 100) [total macd\_below change effect fix size]

,(((sum(close\_macd\_below - close\_first) - sum(close\_last - close\_first))/avg(close\_first) \* 100))/count(\*) [average macd\_below change effect fix size]

from [AllNasdaqTickerPricesfrom2014into2017].[dbo].[macd\_below\_signal\_comparison\_report]

where

number\_of\_row\_numbers < 50 and number\_of\_row\_numbers > 3

and

number\_of\_row\_numbers > 3 and number\_of\_row\_numbers < 50

and

symbol not in ('BZUN', 'ACXM', 'MMSI', 'CORT', 'MU', 'TIL')

union

select

'validation sample' [sample name]

,count(\*) [number of trades]

,(sum(close\_last - close\_first)/avg(close\_first) \* 100) [centerline crossover]

,(sum(close\_macd\_below - close\_first)/avg(close\_first) \* 100) [centerline crossover after macd\_below change fix]

,((sum(close\_macd\_below - close\_first) - sum(close\_last - close\_first))/avg(close\_first) \* 100) [total macd\_below change effect fix size]

,(((sum(close\_macd\_below - close\_first) - sum(close\_last - close\_first))/avg(close\_first) \* 100))/count(\*) [average macd\_below change effect fix size]

from [AllNasdaqTickerPricesfrom2014into2017].[dbo].[macd\_below\_signal\_comparison\_report]

where

number\_of\_row\_numbers < 50 and number\_of\_row\_numbers > 3

and

number\_of\_row\_numbers > 3 and number\_of\_row\_numbers < 50

and

symbol in ('BZUN', 'ACXM', 'MMSI', 'CORT', 'MU', 'TIL')

-- no fix effect size in base sample versus validation sample

select

'base sample' [sample name]

,count(\*) [number of trades]

,(sum(close\_last - close\_first)/avg(close\_first) \* 100) [centerline crossover]

,(sum(close\_earlier - close\_first)/avg(close\_first) \* 100) [centerline crossover after no fix change fix]

,((sum(close\_earlier - close\_first) - sum(close\_last - close\_first))/avg(close\_first) \* 100) [total no fix change effect fix size]

,(((sum(close\_earlier - close\_first) - sum(close\_last - close\_first))/avg(close\_first) \* 100))/count(\*) [average no fix change effect fix size]

from [AllNasdaqTickerPricesfrom2014into2017].[dbo].[earlier\_close\_comparison\_report]

where

number\_of\_row\_numbers <= 3

and

symbol not in ('BZUN', 'ACXM', 'MMSI', 'CORT', 'MU', 'TIL')

union

select

'base sample' [sample name]

,count(\*) [number of trades]

,(sum(close\_last - close\_first)/avg(close\_first) \* 100) [centerline crossover]

,(sum(close\_macd\_below - close\_first)/avg(close\_first) \* 100) [centerline crossover after no fix change fix]

,((sum(close\_macd\_below - close\_first) - sum(close\_last - close\_first))/avg(close\_first) \* 100) [total no fix change effect fix size]

,(((sum(close\_macd\_below - close\_first) - sum(close\_last - close\_first))/avg(close\_first) \* 100))/count(\*) [average no fix change effect fix size]

from [AllNasdaqTickerPricesfrom2014into2017].[dbo].[macd\_below\_signal\_comparison\_report]

where

number\_of\_row\_numbers <= 3

and

symbol not in ('BZUN', 'ACXM', 'MMSI', 'CORT', 'MU', 'TIL')

union

select

'validation sample' [sample name]

,count(\*) [number of trades]

,(sum(close\_last - close\_first)/avg(close\_first) \* 100) [centerline crossover]

,(sum(close\_earlier - close\_first)/avg(close\_first) \* 100) [centerline crossover after no fix change fix]

,((sum(close\_earlier - close\_first) - sum(close\_last - close\_first))/avg(close\_first) \* 100) [total no fix change effect fix size]

,(((sum(close\_earlier - close\_first) - sum(close\_last - close\_first))/avg(close\_first) \* 100))/count(\*) [average no fix change effect fix size]

from [AllNasdaqTickerPricesfrom2014into2017].[dbo].[earlier\_close\_comparison\_report]

where

number\_of\_row\_numbers <= 3

and

symbol in ('BZUN', 'ACXM', 'MMSI', 'CORT', 'MU', 'TIL')

union

select

'validation sample' [sample name]

,count(\*) [number of trades]

,(sum(close\_last - close\_first)/avg(close\_first) \* 100) [centerline crossover]

,(sum(close\_macd\_below - close\_first)/avg(close\_first) \* 100) [centerline crossover after no fix change fix]

,((sum(close\_macd\_below - close\_first) - sum(close\_last - close\_first))/avg(close\_first) \* 100) [total no fix change effect fix size]

,(((sum(close\_macd\_below - close\_first) - sum(close\_last - close\_first))/avg(close\_first) \* 100))/count(\*) [average no fix change effect fix size]

from [AllNasdaqTickerPricesfrom2014into2017].[dbo].[macd\_below\_signal\_comparison\_report]

where

number\_of\_row\_numbers <= 3

and

symbol in ('BZUN', 'ACXM', 'MMSI', 'CORT', 'MU', 'TIL')

The following screen shot shows the result sets from the preceding script.

The first two rows are for comparisons of the centerline model versus the earlier close model in the base sample and validation sample rows. The third column shows the centerline model profit without any supplemental model effect. The fourth column shows the centerline profit supplemented by the earlier close model. The fifth column shows the total size of the earlier close model effect, and the sixth column shows the size of the earlier close model effect on a per trade basis.

The earlier close model improves the performance of the centerline model by about $6.40 per share per trade in both base and validation samples. This is an informative outcome that confirms the earlier close model profit enhancement is just about the same for two mutually exclusive sets of stock symbols.

Because there are more stocks and more trades in the base sample, the total profit enhancement is greater in the base sample than for the validation sample.

For trades lasting 50 days or longer by centerline cross-over model, the earlier close model adds about thirty percent more profit to the centerline model profits.

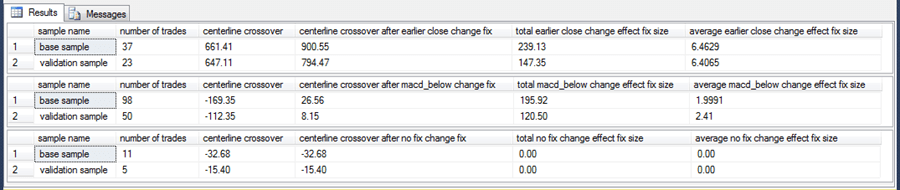
The second two rows highlight the improvement in trade performance for the centerline model when it is supplemented by the macd\_below model.

The trades for these two rows result in losses for the centerline model. However, supplementing the centerline model with the macd\_below model converts the losses into gains.

The average size of the improvement is nearly $2.00 per share per trade in the base sample for the macd\_below model. The validation sample shows an even larger improvement for the stock symbols tracked by it.

The third pair of rows is for the trades that do not attempt to supplement the centerline model with either the earlier close model or the macd\_below model. Recall that these are very short trades that last just two or three days. The third and fourth columns show the same type of trade performance outcomes (a relatively small loss) in the base and validation samples.

Across both the earlier close and macd\_below models, the profit added relative to the centerline model grows by slightly over fifty percent so that the total profit per share grows from $1091.09 to $1673.49! These profits are based on buying and selling just a single stock share, but it is common for stock share transactions to be for many shares. In any event, this phenomenal outcome on a percentage basis confirms the value of applying basic data science techniques to traditional technical analysis methods for finding profitable trades.

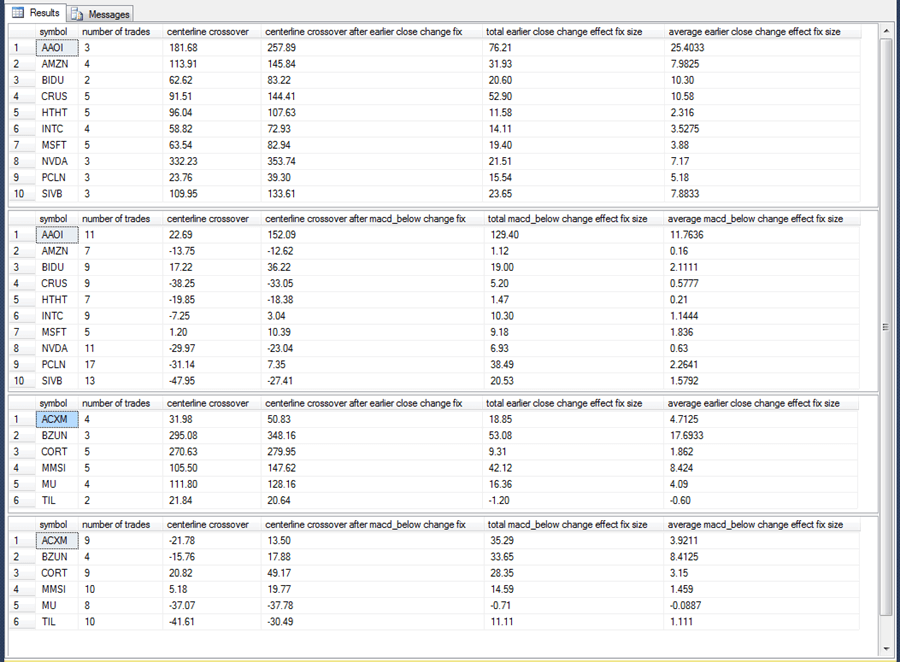


The next screen shot shows the enhancement effects from the earlier close model and the macd\_below model for individual stock symbols; the code to generate these results along with prior results are in the [macd comparer report analysis.sql](https://www.mssqltips.com/tipimages2/5371_AllNasdaqTickerPricesfrom2014into2017.zip) file within the download for this tip. As you can see, there are four results sets.

The first two result sets are for the base sample with the first one being for the earlier close model and the second one being for the macd\_below model.

The last two result sets report outcomes for the stock symbols in the validation sample. The third result set is for the earlier close model, and the final result set is for the macd\_below model.

With exception of the MU symbol (for Micron Technology, Inc.) for the macd\_below model, the average size of the supplemental model effect is greater than zero for all stock symbols in both the base and validation samples! In other words, both supplemental models (earlier close and macd\_below) were broadly successful at improving trade performance over the centerline model.



Next Steps

To start testing and adapting the scripts presented in this tip, you will need to download the backup file for the [AllNasdaqTickerPricesfrom2014into2017 database](https://www.mssqltips.com/tipimages2/5371_AllNasdaqTickerPricesfrom2014into2017.zip). For your convenience in testing and adapting code from this tip and prior related tips ( [here](https://www.mssqltips.com/sqlservertip/5231/import-end-of-day-prices-for-all-nasdaq-stocks-into-sql-server/), [here](https://www.mssqltips.com/sqlservertip/5248/mining-time-series-data-by-calculating-moving-averages-with-tsql-code-in-sql-server/), [here](https://www.mssqltips.com/sqlservertip/5290/mining-time-series-with-exponential-moving-averages-in-sql-server/), [here](https://www.mssqltips.com/sqlservertip/5301/mining-stock-price-time-series-with-macd-in-sql-server/) ), the download associated with the current tip includes a refreshed copy of the backup file for the AllNasdaqTickerPricesfrom2014into2017 database. In order to speed download times and reduce storage requirements, the refreshed copy includes a subset of the symbols from prior tips, such as the macd\_indicators table. The key tables generated from this tip, including the earlier\_close\_comparison\_report table and the macd\_below\_signal\_comparison\_report table have results for all symbols used in the tip. The download additionally includes key script files used to generate temporary and permanent tables referenced in this tip.

After restoring the AllNasdaqTickerPricesfrom2014into2017 database backup file on the computer from which you run SQL Server, you can download the script files.

Next, start examining the content in the earlier\_close\_comparison\_report table and the macd\_below\_signal\_comparison\_report table from the database. These two tables summarize results from the two supplemental models. Make sure that you understand the content in the tables. Consider making copies of the earlier\_close\_comparison\_report and macd\_below\_signal\_comparison\_report tables before running the two main script files ([close\_last\_vs\_close\_earlier\_comparer.sql and close\_last\_vs\_macd\_line\_signal\_cross-over\_comparer.sql)](https://www.mssqltips.com/tipimages2/5371_AllNasdaqTickerPricesfrom2014into2017.zip" \t "_blank). This is because the two main script files modify the comparison report tables in the database in ways that can introduce duplicate records if you do not back them up first and create a fresh empty version of the tables for documenting trades. Recall that you can create a fresh empty copy of the comparison report tables with commented code in each of the main script files.

You may care to try running the code from the two main script files with different stock symbols than the 16 referenced in this tip. The AllNasdaqTickerPricesfrom2014into2017 database released with this tip also includes data for 16 stock symbols not reported on in the tip that you can use for additional testing.

Finally, I close by indicating that this tip is not recommending any particular stocks, including those for which results are reported in this tip. The stocks reported on in this tip were selected because I had been watching them as part of my normal investing due diligence. At the time that I submitted this tip to MSSQLTips.com, my family members held positions in a subset of these stocks.

Last Update: 2018-02-28