# Task: Build a custom forecast model for a set of time series

**Input**: a table of multiple time series, one per row. The table has a time column containing a vector of timestamps and the series column containing a vector of metric values

**Output**: new time & series column extrapolated to forecast n bins

**Method**: decomposition of each time series to seasonal + trend + residual (additive model)

Steps:

1. Create & render your time series set (using [make-series](https://kusto.azurewebsites.net/docs/query/make-seriesoperator.html) operator)
2. If needed split the series column to train & test parts (using [array\_slice()](https://kusto.azurewebsites.net/docs/query/arrayslicefunction.html) or [array\_split()](https://kusto.azurewebsites.net/docs/query/arraysplitfunction.html))
3. Optional filter for outliers: either using [series\_fir()](https://kusto.azurewebsites.net/docs/query/series-firfunction.html), [series\_iir()](https://kusto.azurewebsites.net/docs/query/series-iirfunction.html) or non-linear rolling median using Python [pandas.Series.rolling()](https://pandas.pydata.org/pandas-docs/stable/generated/pandas.Series.rolling.html))
4. Decompose to seasonal & non-seasonal components (using [series\_seasonal()](https://kusto.azurewebsites.net/docs/query/series-seasonalfunction.html))
5. Decompose the non-seasonal to trend & residual (using [series\_fit\_line()](https://kusto.azurewebsites.net/docs/query/series-fit-linefunction.html))
6. Extrapolate the seasonal and the trend to forecast (using [array\_concat()](https://kusto.azurewebsites.net/docs/query/arrayconcatfunction.html))
7. Create the final forecast by adding the seasonal & trend (using [series\_add()](https://kusto.azurewebsites.net/docs/query/series-addfunction.html))

Solution:

* Time-Series-Forcast-Walkthrough.csl