This is the 2nd installment of Maximo Formula's which is releasing as part of the Maximo 7606 release. In this version a lot of internal engine improvements as well as a slew of new baked in functions that can help folks access Maximo business data quicker than ever. For those who are new to Maximo Formula's I would strongly recommend reading the part 1 of this document that covers the basic of Maximo Formulas as well as what the version one provided.

First let's start with the **new features**.

1. **The formula expression now support handling ALN values directly inside the formula**. If you have used the version one you would see that the expression grammar did not support literal strings inside the expression. For example say you had to do an evaluation like such that for a broken status you want to put a score of 0 and 1 otherwise. Intuitively you would like to do the expression below:

    ```
    IF(status==”BROKEN”,0,1)
    ```

    Unfortunately this will not work in the version one as the processor fails to parse the quotes “”. So to do this simple expression, you would end up having to define a Maximo condition like BROKENSTATUS - :status=’BROKEN’ and then applying it to the Formula expression like below:

    ```
    IF(cond$BROKENSTATUS,0,1)
    ```

    While this will work with version 2, the engines grammar has been enhanced to recognize these string literals and hence the most intuitive way to write the expression is possible now.

    ```
    IF(status==”BROKEN”,0,1)
    ```

    Note that the grammar expects the string literals to be wrapped with double quotes (“”) and not single quotes (‘’). Just like Mbo attributes can have ALN values, there are certain functions (like **meterval**) that return ALN values (say for the characteristic meters). The function return values also can be compared with literal strings with this enhancement.

2. **The formula engine now supports lazy evaluation of the IF function**. This is a significant enhancement from performance as well as usability of the engine. If you are wondering what it really means for you, I would rather resort to an example to elaborate the enhancement. Say you want to evaluate an expression that uses the IF function as shown below:

    ```
    IF(COUNTF(“openwo”)>10,METERAVGVAL(“PRESSURE”,DURATION(0,0,10,0,
            0,0)), METERAVGVAL(“PRESSURE”,DURATION(0,0,0,10,0,0)))
    ```

As evident, this expression would evaluate the count of open work orders (say for an asset) and if that is greater than 10, it would like to get the average reading value of the PRESSURE meter for the last 10 days else it will get the average reading value of the PRESSURE meter for the last 10 hours. Now these are costly operations and you would not want them to run if the condition is not met. However the formula engine of version one would evaluate both the METERAVGVAL functions even before evaluating the IF condition. The version 2 engine does lazy evaluation of nested expressions inside the IF function. The engine would evaluate either one of the METERAVGVAL functions based on the IF condition. This helps performance of IF expressions, considering the fact that these nested expression can be deeply nested (ie can have sub expressions).

A direct implication of how this improves the usability is illustrated below. Say we wanted to evaluate an expression that calculates the replacement cost of an asset based on the priority and purchaseprice of the asset. The expression may look like below.

```
IF(isnull$priority,purchaseprice*0.8,purchaseprice/priority)
```

This statement would work as is in V2, but would have failed in the V1 because of the purchaseprice/priority expression as the engine would have evaluated that expression even in cases when the priority attribute had a null value. To make this work in the V1, we would need to add an NVL for that expression:

```
IF(isnull$priority,purchaseprice*0.8,purchaseprice/NVL(priority,1))
```

3. **All built in functions in an Maximo Formula's are now backed up by a cache**. This is another enhancement that helps improve performance. An example below is going to show what this implies.

```
IF(METERAVGVAL(“PRESSURE”,DURATION(0,0,10,0,0,0))>=10,0,IF(METERAVGVAL
(“PRESSURE”,DURATION(0,0,10,0,0,0))>5 &&
METERAVGVAL(“PRESSURE”,DURATION(0,0,10,0,0,0))<10,0.5,1))
```

This expression would compare the average meter value against a set of ranges to provide a score. But note that the function METERAVGVAL is invoked 3 times here, with exactly the same parameters. With the new enhancement, this call is cached the first time and the value is reused for the remaining calls. Note the cache expires after the expression is evaluated, so that it will not be reused for the next evaluation of this same expression.

4. We have added a bunch of **new functions** that will help writing formulas with lesser effort. We have added functions around the IBM Weather api as well as around the Maximo meters - both Asset and Locations. In the next section we would cover these new functions in detail.

5. We have added a lot more **syntax and grammar validation** of formula expression. This was only partially addressed in the V1. This helps us catch typo's at the design time rather than at the run time.

Next we talk about the new set of functions that we added to the library of formula functions.

Weather Functions

We have an integration inside Maximo with the various weather apis (provided by IBM) using federated resources. The Weather functions leverage those federated resources to provide certain insights. Most of the weather functions that we have provided so far either leverages the historical weather or the current weather. As discussed earlier, all these function results are cached for the duration of the expression evaluation. This is pertinent in this context as all the listed weather functions would result in some callout to the IBM Weather API's, which maybe sometimes costly for performance in certain cases. The enhancement would at least make sure that we do not call out every time the functions are used and instead leverage the existing cache.The weather functions leverage the relationship from Asset and Location to the Service Address object in Maximo. Currently these functions assume we have a relationship named SERVICEADDRESSHIERARCHY between the Maximo Object in context and the Maximo Service Address Object. This is true for Locations object. For Asset though it uses the SERVICEADDRESSLOGIC relationship to get to the service address object. These functions will return null if the service address object does not have the latitude and logitude set as the weather interface is expecting the geo location. In the next drop we will add support for making that relationship name configurable. Note that Weather is a paid api, and a Maximo installation needs to purchase the Weather api key for these function to work. The weather functions are listed below:

Functions	**Number of Params**	**Description**	**Usage**
WHAVG	2	Used for calculating the average for any weather element over a span of a number of days. Takes in 2 parameters, the weather element name and the number of days.The weather element name is simply the	IF(WHAVG("temp",10)>80,0,1)

		name of the weather attribute from the HISTWEATHER federated resource which is based on the IBM Historical Weather API. An weather element maybe the humidity (RH) or temperature (temp) or any other weather aspect that the user wants to track and driver insights. For example, WHAVG("temp",10) gives you the avg "temp" (temperature) value for last 10 days. As you can figure out, you can easily replace that with WHAVG("rh",30) - which means - give me the average humidity for last 30 days.	
WHMAX	2	Used for calculating the maximum for any weather element over a span of a number of days. Takes in 2 parameters, the weather element name and the number of days.The weather element name is simply the name of the weather attribute from the HISTWEATHER federated resource which is based on the IBM Historical Weather API. An weather element maybe the humidity (RH) or temperature (temp) or any other weather aspect that the user wants to track and driver insights. For example, WHMAX("temp",10) gives you the max "temp" (temperature) value for last 10 days. As you can figure out, you can easily replace that with WHMAX("rh",30) - which means - give me the max humidity for last 30 days.	IF(WHMAX("temp",10)>100 && WHAVG("temp",10)>80,0,1)
WHMIN	2	Same as WHMAX, with the difference that this function is used to calculate the minimum value for a weather element over a period of days.	IF(WHMIN("temp",10)>0 && WHAVG("temp",10)<32,0,1)
WCVAL	1	This function uses the current weather api to get the current value of a weather element. An example	IF(WCVAL("temp",10)>120 && WHAVG("temp",10)>1

		would be WCVAL("temp") - gives the current value for the "temp". As you can see, it uses 1 parameter - which represents the attribute name of the weather element as per the federated resource CURWEATHER.	00,0,1)
WHTCOUNT	4	This function uses the historical weather api to get the count of days a weather element has either exceeded or fallen below a given value. For example WHTCOUNT("temp", 10, 32, 0) gives you the count of times the "temp" exceeded the value 100 in last 10 days. The same can be used to say WHTCOUNT("temp", 10, 32, 0) - how many days the "temp" dropped below freezing in last 10 days.	IF(WHTCOUNT("temp", 10, 100, 1)>=9,0,IF(WHTCOUNT("temp", 10, 100, 1)>3 && WHTCOUNT("temp", 10, 100, 1)<9,0.5,1))

Asset/Locations related functions:

These functions are meter based functions, that can be used with any expression that is evaluated in the context of Asset or Locations object. The meter readings are stored in METERREADING or MEASUREMENT tables and the function figures that out based on the meter information available in Maximo meter table. As noted before, these function results are cached for the expression at evaluation time and discarded after the evaluation finishes. You can leverage the examples shown below both as Object and Attribute formulas.

Functions	**Number of Params**	**Description**	**Usage**
METERVAL	1	Used for calculating the last reading for a meter - Asset or Location. This can be used to read any kind of meter - Characteristic/ Gauge or Continuous. An example usage could be METERVAL(“OILCOLOR”). This will return a stirng value as OILCOLOR is a characteristic	IF(METERVAL(“OILCOLOR”)==”CLEAR”,1,0)

		meter.	
METERAVGVAL	2	Used for calculating the average meter value over a span of time. This is only applicable to Gauge meters. An example usage could be METERAVGVAL(“O-PRESSUR”,DURATION(0,0,30,0,0,0) - which gives you the average reading of the O_PRESSUR meter over a span of the last 30 days	
METERMAXVAL	2	Used for calculating the maximum meter value over a span of time. This is only applicable to numeric meters..	An example usage could be METERMAXVAL(“O-PRESSUR”,DURATION(1,0,0,0,0,0) - which gives you the maximum reading of the O_PRESSUR meter over a span of the last 1 year
METERMINVAL	2	Used for calculating the maximum meter value over a span of time. This is only applicable to numeric meters.	An example usage could be METERMINVAL(“O-PRESSUR”,DURATION(1,0,0,0,0,0)) - which gives you the minimum reading of the O_PRESSUR meter over a span of the last 1 year.
METEREXCEED COUNT	3	This is in similar lines with the WHTCOUNT where you need to calculate the number of times the meter readings exceeded the target value. This function only caters to the “exceed” aspect and does not handle the “falling below a certain value” aspect. That would be covered in the next drop.	An example usage is shown below: METEREXCEEDCOUNT(“O-PRESSUR”,DURATION(1,0,0,0,0,0),0.9) which counts the meter readings that are above 0.9 in the last 1 year.
SPECATTRVAL	1	This one can be used to calculate the class spec attribute value for that asset.Although this does not	SPECATTRVAL(“specatrname”). This will figure out if the spec

		qualify as a meter function, its still related to assets and locations and hence we included it in this section.This works for locations and asset objects only.	attribute value is a numeric one or a alphanumeric one and return a string value or a numeric value accordingly.

Statistical Functions:

We added a set of statistical functions - SUMF, COUNTF, AVGF, MAXF, MINF, that should replace the ones we had in V1. In V1 we supported an expression like count$relation, max$relation$attr - which works but would not support a date range. To do that we had introduced another variant of that like FMAX$relation$attr$timelineattr(DURATION(...)). All these are still supported, but with the grammar enhancement to support literal strings we have simpler ways to do the same. As noted above, all these function values are cached for the duration of expression calculation. The table below showcases those:

Functions	**Number of Params**	**Description**	**Usage**
SUMF	4	Used for calculating the sum of a related attribute over a period of time.The usage is similar: ,SUMF(“relation”,”attr_to_sum”,”timeline_attr”,DURATION(..)).	A simple example would be: SUMF(”openwo” “estmatcost”,”statusdate”,DURATION(0,0,10,0,0,0)). As you can guess, this formula is based on the Asset object and we are trying to sum up the estmatcost for all open workorders (openwo relationship) for last 10 days. Setting the duration to -1 will do the sum for all open workorders

			for an Asset: SUMF(”openwo” “estmatcost”,”statusdate”,-1).
MAXF	4	Used for calculating the max of a related attribute over a period of time.The usage is similar: ,MAXF(“relation”,”attr_to_sum”,”timeline_attr”,DURATION(..)).	You can guess the sample from above SUMF example.
MINF	4	Used for calculating the min of a related attribute over a period of time.The usage is similar: ,MINF(“relation”,”attr_to_sum”,”timeline_attr”,DURATION(..)).	Same as SUMF example.
AVGF	4	Used for calculating the avg of a related attribute over a period of time.The usage is similar: ,MINF(“relation”,”attr_to_sum”,”timeline_attr”,DURATION(..)).	Same as SUMF example.
COUNTF	3	Used for calculating the count of a related mboset over a period of time.The usage is similar: ,AVGF(“relation”,”timeline_attr”,DURATION(..)).	COUNTF(”openwo”, ”statusdate”,DURATION(0,0,10,0,0,0)).

All these functions that we covered, can be used in any expression - whether its for Attribute or Object formula. For example, you can define a non-persistent attribute for an object - say Asset, and associate a formula expression like AVGF(“openwo”,“estcost”,”statusdate”,”openwo”) to that attribute. Now you can see that attribute value getting set to the average estmatcost value for all open workorders for that Asset. Same goes for the weather functions.