# edō Transaction Validation Specification Language

Below is the 'Complex Preward Example' (items 1 through 7) from earlier. The numbers have been omitted and statement variables (A through F) are introduced for truth-functional statements. The statements may be specified using the following form: **property operator *argument(s)***, (which is extended later) where *property* is a transaction data element, *operator* is a rule or criteria and the *argument* is an array of one or more variables:

$5 Preward to Outback Steakhouse (*Not truth-functional but rather a consequent.*)

**A** Preward only valid at the locations at 315 Maple Ave E Vienna, VA 22180-4717 and 3411 S Las Vegas Blvd. Las Vegas, NV 89109 (MID restriction) **MID IN *MID1 [, MID2, …, MIDn] Specified using the IN operator where property is contained within the specified list of values.***

**B** Preward only valid for at the take-out service (TID restriction) **TID EQ *TID1 Specified using the EQ (equals) operator where property equals a particular value.***

**C** Preward only valid on the first Tuesday and last Thursday of October, November and December (Date restriction) **DATE IN *DATE1 [, DATE2, …, DATEn]***

**D** Preward only valid between 4pm and 10pm (time restriction) **TIME BETWEEN *TIME1, TIME2*** *Specified using the BETWEEN operator where property value is between two distinct values.*

**E** Minimum spend is $20 (purchase filter) **AMOUNT GE *VALUE1*** *Specified using the GE (greater than or equal to) operator where property is greater than or equal to a particular value.*

**F** But if you spend more than $50, the Preward value jumps to 20% off (tiered scenario, with percentage) **AMOUNT GT VALUE2** *Specified using the GT (greater than) operator where property is greater than a particular value.*

## The Simple Case – Considering Statements A through E

The specification of the above rules (A through E) can be arranged in a table structure:

(All of the fields are string values that are parsed at runtime and converted to objects or variables in our Validation System.)

|  |  |  |
| --- | --- | --- |
| **MID** | **IN** | ***MID1 [, MID2, …, MIDn]*** |
| **TID** | **EQ** | ***TID1*** |
| **DATE** | **IN** | ***DATE1 [, DATE2, …, DATEn]*** |
| **TIME** | **BETWEEN** | ***TIME1, TIME2*** |
| **AMOUNT** | **GE** | ***VALUE1*** |

Where the final statement, **S** (to be evaluated by the system), is understood to be the conjunction of all of the above rows: **S = A Λ B Λ C Λ D Λ E**. (Negated operators, **¬IN**, **¬EQ**, etc. are possible in this scheme as well.) To establish the Preward, the system simply determines **IF S THEN** $5 Preward to Outback Steakhouse. All of the statement variables are evaluated in our systolic-array. If any one of the variables A, B, C, etc. evaluate to False before the others then the system can determine that the whole statement, **S**, is False.

Note too that disjunctions are possible by virtue of the argument array. Thus statement **A**, is really a disjunction of smaller statements **A = (a1 V a2 V a3 … V an)** where if any of its variables are true then the whole statement, **A** is true.

The More Complex Case – Statements A through F

The multiple consequents ($5 Preward to Outback or the Preward value jumps to 20% off) make the matter more complex. So what we really have now is two statements:

**S1 = A Λ B Λ C Λ D Λ E**

**S2 = A Λ B Λ C Λ D Λ E Λ F**

And so the system must determine...

**IF S1 THEN** $5 Preward to Outback

**IF S2 THEN** 20% off Preward to Outback

Meaning that if **F** evaluates to False, **S2** is False but it's still possible that **S1** may be True. On the other hand, if **E** evaluates to False, then both **S1** and **S2** are False.

To specify the above we really need to specify multiple contexts, where a context is a relation on a set of rules to a Preward:

**C1**: **{A, B, C, D, E}**, $5 Preward to Outback

**C2**: **{A, B, C, D, E, F}**, 20% off Preward to Outback

The specification of the above rules might look like this :

|  |  |  |  |
| --- | --- | --- | --- |
| Property | Operator | Args | Contexts |
| **MID** | **IN** | ***MID1 [, MID2, …, MIDn]*** | C1, C2 |
| **TID** | **EQ** | ***TID1*** | C1, C2 |
| **DATE** | **IN** | ***DATE1 [, DATE2, …, DATEn]*** | **C1, C2** |
| **TIME** | **BETWEEN** | ***TIME1, TIME2*** | **C1, C2** |
| **AMOUNT** | **GE** | ***VALUE1*** | **C1, C2** |
| **AMOUNT** | **GT** | **VALUE2** | **C2** |

|  |  |
| --- | --- |
| ***Operators*** | ***Description*** |
| ***MATCH*** | *A regular expression match. Argument is a regular expression* |
| ***LIKE*** | *Simple match case-insensitive but no anomalies in the argument* |
| ***IN*** | *Is value in a given list? Is x in y where y=[34,56,78,95]* |
| ***BETWEEN*** | *Is value between 2 numbers* |
| ***EQ*** | *Is value equal to* |
| ***LE*** | *Is value less than or equal to* |
| ***LT*** | *Is value less than* |
| ***GE*** | *Is value greater than or equal to* |
| ***GT*** | *Is value greater than* |