Challenge May-2020
Pay-As-You-Go Pricing

A solution with DT5GL by Jack Jansonius – 18 July 2020

To be honest, without Dr. Bob Moore's excellent analysis, I wouldn't have started this challenge. When I looked at his solution, and in particular the conditional expressions in Excel, I immediately saw possibilities to quickly adopt his solution in my tool.

Just recently I had built in the possibility of formulas and real numbers (reals or floats), for example for the calculation of PMT in the Loan Origination - challenge of June 2017, and it looked like this:

```
Attribute: PMT   Type: Real
Derived_from_formula: (Amount*Rate/12)/(1-(1+Rate/12)**-Term)
```

The python library numpy already has a pmt function, so the above can also be done in the following way:

```
Attribute: PMT   Type: Real
Derived_from_formula: abs(pmt(Rate/12, Term, Amount))
```

With which I want to say: the formulas used within dt5gl are full Python.

It will not be surprising that Python also has conditional expressions and these are intuitively very easy to understand. Even without knowledge of this language, the next puzzle can be solved quickly:

```
>>> a=13
>>> b=11 if a==11 else 12 if a==12 else 13 if a==13 else 0
>>> b
?
```

With one peculiarity: the last else branch is mandatory.

In any case, the conditional expressions in Excel by Bob Moore can now be quickly adopted in Python.

So his formula: I =IF(E7 > Tier_3, C7, IF(B7 > Tier_3, B7 -Tier_3, 0))

becomes in dt5gl\python now:

```
Attribute: units_in_tier3
Equals: input3 if UnitsInMonthAtStartLastHour > Tier_3  \
        else input2 - Tier_3 if input2 > Tier_3         \
        else 0
```

Where I have introduced next to "Derived_from_formula: " the shorter "Equals:"; they obviously mean the same thing.

Now that Bob Moore's solution has been adopted in dt5gl (solution 1), I am also working out another solution that relies much more on decision tables than on conditional expressions.

With the question: can't we calculate the number of units consumed in one tier per tier sec (instead of depending on the number of units consumed in other tiers)?

The formal determination of the number of units consumed in Tier 2 with a single decision table looks like this (at least in dt5gl):

```
Table 2: Units in tier 2
If:                                  | 0| 1| 2| 3| 4| 5|
Start > Tier_3                       | Y| N| N| N| N| N|
Start > Tier_2                       | -| Y| Y| N| N| N|
End > Tier_3                         | -| Y| N| Y| N| N|
End > Tier_2                         | -| -| -| -| Y| N|
Then:
units_in_tier2 = Tier_3 - Tier_2     |  |  |  | X|  |  |
units_in_tier2 = Tier_3  - Start     |  | X|  |  |  |  |
units_in_tier2 = End - Tier_2        |  |  |  |  | X|  |
units_in_tier2 = End - Start         |  |  | X|  |  |  |
units_in_tier2 = 0                   | X|  |  |  |  | X|
# .......
```

where..:
Start = units in month to start hour (input 2 – input 3)
End = units in month to end hour (input 2)
Tier_2 = 20000 (= the lower limit for tier 2)
Tier_3 = 50000 (= the lower limit for tier 3)

A visualization of this problem will very quickly show the 6 possible situations in this table!

The same applies to the calculation of the number of units in tier 3:

```
Table 3: Units in tier 3
If:                                  | 0| 1| 2|
Start > Tier_3                       | Y| N| N|
End > Tier_3                         | -| Y| N|
Then:
units_in_tier3 = End - Start         | X|  |  |
units_in_tier3 = End - Tier_3        |  | X|  |
units_in_tier3 = 0                   |  |  | X|
# .......
```

Note that this table corresponds to the previously given conditional expression:

```
Attribute: units_in_tier3
Equals: input3 if UnitsInMonthAtStartLastHour > Tier_3  \
        else input2 - Tier_3 if input2 > Tier_3         \
        else 0
```

And actually the calculation of the number of units in tier 1 does not deviate from this; except for the fact that this result still has to be corrected with any units still available for free use.

```
Table 1: Units in tier 1
If:                                  | 0| 1| 2|
Start > Tier_2                       | Y| N| N|
End > Tier_2                         | -| Y| N|
Then:
units_in_tier1_bc = 0                | X|  |  |
units_in_tier1_bc = Tier_2 - Start   |  | X|  |
units_in_tier1_bc = End - Start      |  |  | X|
# .......
bc=before correction with free units
```

The correction of the number of units used in tier 1 can easily be done with an expression:

```
Attribute: units_in_tier1
Equals: max(units_in_tier1_bc - FreeUnitsAtStartLastHour, 0)
```

but it can as well be done with a decision table:

```
Table 1a: correct tier1 with free units
If:                                                    | 0| 1|
units_in_tier1_bc <= FreeUnitsAtStartLastHour          | Y| N|
Then:
units_in_tier1 = 0                                     | X|  |
units_in_tier1 = units_in_tier1_bc - FreeUnitsAtStartLastHour  |  | X|
# .......
```

Solution 1: based on conditional expressions.

```
Table 0: Payments
If:                                        | 0| 1| 2| 3| 4| 5| 6|
units_in_tier3 > 0                         | Y| Y| Y| N| N| N| N|
units_in_tier2 > 0                         | Y| Y| N| Y| Y| N| N|
units_in_tier1 > 0                         | Y| N| -| Y| N| Y| N|
Then:
report is tier3+tier2+tier1                | X|  |  |  |  |  |  |
report is tier3+tier2                      |  | X|  |  |  |  |  |
report is tier3                            |  |  | X|  |  |  |  |
report is tier2+tier1                      |  |  |  | X|  |  |  |
report is tier2                            |  |  |  |  | X|  |  |
report is tier1                            |  |  |  |  |  | X|  |
report is notier                           |  |  |  |  |  |  | X|
'show derived data from input'             | X| X| X| X| X| X| X|
# .......

# with thanks to Dr. Bob Moore's excellent analysis:
# D =A7 - C7
# E =B7 - C7
# F =MAX(Free_Units-D7, 0)
# G =IF(C7 - I7 - H7 < F7, 0, C7 - I7 - H7 - F7)
# H =IF(E7 > Tier_2, C7 - I7, IF(B7 > Tier_2, B7 - I7 - Tier_2, 0))
# I =IF(E7 > Tier_3, C7, IF(B7 > Tier_3, B7 -Tier_3, 0))

Attribute: TotalUnitsAtStartLastHour     Type: Integer
Equals: input1 - input3

Attribute: UnitsInMonthAtStartLastHour   Type: Integer
Equals: input2 - input3

Attribute: FreeUnitsAtStartLastHour      Type: Integer
Equals: max(Free_Units - TotalUnitsAtStartLastHour, 0)

Attribute: units_in_tier1
Equals: 0 if input3 - units_in_tier3 - units_in_tier2 < FreeUnitsAtStartLastHour \
        else input3 - units_in_tier3 - units_in_tier2 - FreeUnitsAtStartLastHour

Attribute: units_in_tier2
Equals: input3 - units_in_tier3 if UnitsInMonthAtStartLastHour > Tier_2   \
        else input2 - units_in_tier3 - Tier_2 if input2 > Tier_2          \
        else 0

Attribute: units_in_tier3
Equals: input3 if UnitsInMonthAtStartLastHour > Tier_3  \
        else input2 - Tier_3 if input2 > Tier_3         \
        else 0

Attribute: Free_Units  Type: Integer
Equals: 10000
Attribute: Tier_2      Type: Integer
Equals: 20000
Attribute: Tier_3      Type: Integer
Equals: 50000

Attribute: input1  Type: Integer
Askable_using: "Input 1 - all units to end hour?"

Attribute: input2  Type: Integer
Askable_using: "Input 2 - units in month to end hour?"

Attribute: input3  Type: Integer
Askable_using: "Input 3 - units in last hour?"
```

```
GoalAttribute: report
Case: tier3+tier2+tier1
Print: "tier3+tier2+tier1"
Print: "Units in tier 1: %s" units_in_tier1
Print: "Units in tier 2: %s" units_in_tier2
Print: "Units in tier 3: %s" units_in_tier3

Case: tier3+tier2
Print: "tier3+tier2"
Print: "Units in tier 2: %s" units_in_tier2
Print: "Units in tier 3: %s" units_in_tier3

Case: tier3
Print: "tier3"
Print: "Units in tier 3: %s" units_in_tier3

Case: tier2+tier1
Print: "tier2+tier1"
Print: "Units in tier 1: %s" units_in_tier1
Print: "Units in tier 2: %s" units_in_tier2

Case: tier2
Print: "tier2"
Print: "Units in tier 2: %s" units_in_tier2

Case: tier1
Print: "tier1"
Print: "Units in tier 1: %s" units_in_tier1

Case: notier
Print: "notier"

GoalProposition: 'show derived data from input'
Print: "********* Derived data from input *********"
Print: "All units to start hour...........: %s" TotalUnitsAtStartLastHour
Print: "Units in month to start hour......: %s" UnitsInMonthAtStartLastHour
Print: "Free units remaining at start hour: %s" FreeUnitsAtStartLastHour
```

Solution 2: based on decision tables.

```
Table 0: Payments
If:                                     | 0| 1|
units_in_all_tiers > 0                  | Y| N|
Then:
report is alltiers                      | X|  |
report is notier                        |  | X|
'show derived data from input'          | X| X|
# .......

Attribute: units_in_all_tiers
Equals: units_in_tier1 + units_in_tier2 + units_in_tier3

Attribute: TotalUnitsAtStartLastHour   Type: Integer
Equals: input1 - input3

Attribute: Start                       Type: Integer
Equals: End - input3

Attribute: FreeUnitsAtStartLastHour    Type: Integer
Equals: max(Free_Units - TotalUnitsAtStartLastHour, 0)

Table 1: Units in tier 1
If:                                  | 0| 1| 2|
Start > Tier_2                       | Y| N| N|
End > Tier_2                         | -| Y| N|
Then:
units_in_tier1_bc = 0                | X|  |  |
units_in_tier1_bc = Tier_2 - Start   |  | X|  |
units_in_tier1_bc = End - Start      |  |  | X|
# .......
bc=before correction with free units

Table 1a: correct tier1 with free units
If:                                                          | 0| 1|
units_in_tier1_bc <= FreeUnitsAtStartLastHour                | Y| N|
Then:
units_in_tier1 = 0                                           | X|  |
units_in_tier1 = units_in_tier1_bc - FreeUnitsAtStartLastHour|  | X|
# .......

# Alternative to table 1a:
# Attribute: units_in_tier1
# Equals: max(units_in_tier1_bc - FreeUnitsAtStartLastHour, 0)


Table 2: Units in tier 2
If:                                  | 0| 1| 2| 3| 4| 5|
Start > Tier_3                       | Y| N| N| N| N| N|
Start > Tier_2                       | -| Y| Y| N| N| N|
End > Tier_3                         | -| Y| N| Y| N| N|
End > Tier_2                         | -| -| -| -| Y| N|
Then:
units_in_tier2 = Tier_3 - Tier_2     |  |  |  | X|  |  |
units_in_tier2 = Tier_3  - Start     |  | X|  |  |  |  |
units_in_tier2 = End - Tier_2        |  |  |  |  | X|  |
units_in_tier2 = End - Start         |  |  | X|  |  |  |
units_in_tier2 = 0                   | X|  |  |  |  | X|
# .......
```

```
Table 3: Units in tier 3
If:                                  | 0| 1| 2|
Start > Tier_3                       | Y| N| N|
End > Tier_3                         | -| Y| N|
Then:
units_in_tier3 = End - Start         | X|  |  |
units_in_tier3 = End - Tier_3        |  | X|  |
units_in_tier3 = 0                   |  |  | X|
# .......


Attribute: Free_Units  Type: Integer
Equals: 10000
Attribute: Tier_2      Type: Integer
Equals: 20000
Attribute: Tier_3      Type: Integer
Equals: 50000

Attribute: input1      Type: Integer
Askable_using: "Input 1 - all units to end hour?"

Attribute: End         Type: Integer
Askable_using: "Input 2 - units in month to end hour?"

Attribute: input3      Type: Integer
Askable_using: "Input 3 - units in last hour?"


GoalAttribute: report
Case: alltiers
Print: "Number of units consumed in the last hour: %s" units_in_all_tiers
Print: "Units in tier 1: %s" units_in_tier1
Print: "Units in tier 2: %s" units_in_tier2
Print: "Units in tier 3: %s" units_in_tier3

Case: notier
Print: "No units used in the last hour"


GoalProposition: 'show derived data from input'
Print: "********* Derived data from input *********"
Print: "All units to start hour.....: %s" TotalUnitsAtStartLastHour
Print: "Units in month to start hour......: %s" Start
Print: "Free units remaining at start hour: %s" FreeUnitsAtStartLastHour
```

Various testruns

Test cases for both solutions:

All units	This month units	Last hour units	Tier 1	Tier 2	Tier 3
2,000	2,000	160	0	0	0
10,200	10,200	350	200	0	0
22,000	22,000	20,500	10,000	2,000	0
32,500	32,500	10,500	0	10,500	0
65,000	55,000	30,000	0	25,000	5,000
120,258	60,390	2,350	0	0	2,350
120,000	120,000	120,000	10,000	30,000	70,000

Solution 1:

Case 1:

```
"Input 3 - units in last hour?"
(int)> 160
"Input 2 - units in month to end hour?"
(int)> 2000
"Input 1 - all units to end hour?"
(int)> 2000

notier
********* Derived data from input *********
All units to start hour...........: 1840
Units in month to start hour......: 1840
Free units remaining at start hour: 8160
```

Case 2:

```
"Input 3 - units in last hour?"
(int)> 350
"Input 2 - units in month to end hour?"
(int)> 10200
"Input 1 - all units to end hour?"
(int)> 10200

tier1
Units in tier 1: 200
********* Derived data from input *********
All units to start hour...........: 9850
Units in month to start hour......: 9850
Free units remaining at start hour: 150
```

Case 3:

```
"Input 3 - units in last hour?"
(int)> 20500
"Input 2 - units in month to end hour?"
(int)> 22000
"Input 1 - all units to end hour?"
(int)> 22000

tier2+tier1
Units in tier 1: 10000
Units in tier 2: 2000
********* Derived data from input *********
All units to start hour...........: 1500
Units in month to start hour......: 1500
Free units remaining at start hour: 8500
```

Case 4:

```
"Input 3 - units in last hour?"
(int)> 10500
"Input 2 - units in month to end hour?"
(int)> 32500
"Input 1 - all units to end hour?"
(int)> 32500

tier2
Units in tier 2: 10500
********* Derived data from input *********
All units to start hour...........: 22000
Units in month to start hour......: 22000
Free units remaining at start hour: 0
```

Case 5:

```
"Input 3 - units in last hour?"
(int)> 30000
"Input 2 - units in month to end hour?"
(int)> 55000
"Input 1 - all units to end hour?"
(int)> 65000

tier3+tier2
Units in tier 2: 25000
Units in tier 3: 5000
********* Derived data from input *********
All units to start hour...........: 35000
Units in month to start hour......: 25000
Free units remaining at start hour: 0
```

Case 6:

```
"Input 3 - units in last hour?"
(int)> 2350
"Input 2 - units in month to end hour?"
(int)> 60390

tier3
Units in tier 3: 2350
********* Derived data from input *********
All units to start hour...........: None
Units in month to start hour......: 58040
Free units remaining at start hour: None
```

Note: input1 is not even requested (and several intermediate calculations are therefore not performed). In my opinion an important characteristic of (artificial) intelligence:
only ask for information and perform calculations that are relevant to the problem solving!

Case 7:

```
"Input 3 - units in last hour?"
(int)> 120000
"Input 2 - units in month to end hour?"
(int)> 120000
"Input 1 - all units to end hour?"
(int)> 120000

tier3+tier2+tier1
Units in tier 1: 10000
Units in tier 2: 30000
Units in tier 3: 70000
********* Derived data from input *********
All units to start hour...........: 0
Units in month to start hour......: 0
Free units remaining at start hour: 10000
```

Solution 2:

Case 1:

```
"Input 2 - units in month to end hour?"
(int)> 2000
"Input 3 - units in last hour?"
(int)> 160
"Input 1 - all units to end hour?"
(int)> 2000

No units used in the last hour
********* Derived data from input *********
All units to start hour...........: 1840
Units in month to start hour......: 1840
Free units remaining at start hour: 8160
```

Case 2:

```
"Input 2 - units in month to end hour?"
(int)> 10200
"Input 3 - units in last hour?"
(int)> 350
"Input 1 - all units to end hour?"
(int)> 10200

Number of units consumed in the last hour: 200
Units in tier 1: 200
Units in tier 2: 0
Units in tier 3: 0
********* Derived data from input *********
All units to start hour...........: 9850
Units in month to start hour......: 9850
Free units remaining at start hour: 150
```

Case 3:

```
"Input 2 - units in month to end hour?"
(int)> 22000
"Input 3 - units in last hour?"
(int)> 20500
"Input 1 - all units to end hour?"
(int)> 22000

Number of units consumed in the last hour: 12000
Units in tier 1: 10000
Units in tier 2: 2000
Units in tier 3: 0
********* Derived data from input *********
All units to start hour...........: 1500
Units in month to start hour......: 1500
Free units remaining at start hour: 8500
```

Case 4:

```
"Input 2 - units in month to end hour?"
(int)> 32500
"Input 3 - units in last hour?"
(int)> 10500
"Input 1 - all units to end hour?"
(int)> 32500

Number of units consumed in the last hour: 10500
Units in tier 1: 0
Units in tier 2: 10500
Units in tier 3: 0
********* Derived data from input *********
All units to start hour...........: 22000
Units in month to start hour......: 22000
Free units remaining at start hour: 0
```

Case 5:

```
"Input 2 - units in month to end hour?"
(int)> 55000
"Input 3 - units in last hour?"
(int)> 30000
"Input 1 - all units to end hour?"
(int)> 65000

Number of units consumed in the last hour: 30000
Units in tier 1: 0
Units in tier 2: 25000
Units in tier 3: 5000
********* Derived data from input *********
All units to start hour...........: 35000
Units in month to start hour......: 25000
Free units remaining at start hour: 0
```

Case 6:

```
"Input 2 - units in month to end hour?"
(int)> 60390
"Input 3 - units in last hour?"
(int)> 2350
"Input 1 - all units to end hour?"
(int)> 120258

Number of units consumed in the last hour: 2350
Units in tier 1: 0
Units in tier 2: 0
Units in tier 3: 2350
********* Derived data from input *********
All units to start hour...........: 117908
Units in month to start hour......: 58040
Free units remaining at start hour: 0
```

Case 7:

```
"Input 2 - units in month to end hour?"
(int)> 120000
"Input 3 - units in last hour?"
(int)> 120000
"Input 1 - all units to end hour?"
(int)> 120000

Number of units consumed in the last hour: 110000
Units in tier 1: 10000
Units in tier 2: 30000
Units in tier 3: 70000
********* Derived data from input *********
All units to start hour...........: 0
Units in month to start hour......: 0
Free units remaining at start hour: 10000
```

Some alternative notations in the given solutions.

The 3 decision tables for solution 2 can also been combined into one table:

```
Table 1: Ranges
If:                                  | 0| 1| 2| 3| 4| 5|
Start > Tier_3                       | Y| N| N| N| N| N|
Start > Tier_2                       | -| Y| Y| N| N| N|
End > Tier_3                         | -| Y| N| Y| N| N|
End > Tier_2                         | -| -| -| -| Y| N|
Then:
units_in_tier3 = End - Start         | X|  |  |  |  |  |
units_in_tier3 = End - Tier_3        |  | X|  | X|  |  |
units_in_tier3 = 0                   |  |  | X|  | X| X|
units_in_tier2 = Tier_3 - Tier_2     |  |  |  | X|  |  |
units_in_tier2 = Tier_3 - Start      |  | X|  |  |  |  |
units_in_tier2 = End - Tier_2        |  |  |  |  | X|  |
units_in_tier2 = End - Start         |  |  | X|  |  |  |
units_in_tier2 = 0                   | X|  |  |  |  | X|
units_in_tier1_bc = Tier_2 - Start   |  |  |  | X| X|  |
units_in_tier1_bc = End - Start      |  |  |  |  |  | X|
units_in_tier1_bc = 0                | X| X| X|  |  |  |
# .......
bc=before correction with free units
```

If a fourth or even fifth tier is added to the calculation, this table becomes far too big. However, the calculations for these extra tiers can quickly be added to the 3 decision tables in solution 2!

And the decision table for Tier 2 in Solution 2:

```
Table 2: Units in tier 2
If:                                  | 0| 1| 2| 3| 4| 5|
Start > Tier_3                       | Y| N| N| N| N| N|
Start > Tier_2                       | -| Y| Y| N| N| N|
End > Tier_3                         | -| Y| N| Y| N| N|
End > Tier_2                         | -| -| -| -| Y| N|
Then:
units_in_tier2 = Tier_3 - Tier_2     |  |  |  | X|  |  |
units_in_tier2 = Tier_3  - Start     |  | X|  |  |  |  |
units_in_tier2 = End - Tier_2        |  |  |  |  | X|  |
units_in_tier2 = End - Start         |  |  | X|  |  |  |
units_in_tier2 = 0                   | X|  |  |  | X|  |
# .......
```

can also be converted to a conditional expression:

```
Attribute: units_in_tier2  Type: Integer
Equals: 0               if Start > Tier_3  or  End <= Tier_2                    \
   else Tier_3  - Start  if Start <= Tier_3 and Start > Tier_2 and End > Tier_3   \
   else End - Start      if Start <= Tier_3 and Start > Tier_2 and End <= Tier_3  \
   else Tier_3 - Tier_2  if Start <= Tier_2 and End > Tier_3                      \
   else End - Tier_2     if Start <= Tier_2 and End <= Tier_3 and End > Tier_2    \
   else 9999
```

But of course this is much more error-prone than applying a clear decision table!

Finally, the conditional expressions in solution 1 can also be converted into decision tables:

```
# Attribute: units_in_tier1
# Equals: 0 if input3 - units_in_tier3 - units_in_tier2 < FreeUnitsAtStartLastHour\
#         else input3 - units_in_tier3 - units_in_tier2 - FreeUnitsAtStartLastHour


Table 1: Units in tier 1
If:                                                                                 | 0| 1|
input3 < FreeUnitsAtStartLastHour + units_in_tier3 + units_in_tier2                 | Y| N|
Then:
units_in_tier1 = 0                                                                  | X|  |
units_in_tier1 =
        input3 - units_in_tier3 - units_in_tier2 - FreeUnitsAtStartLastHour         |  | X|
# .......


# Attribute: units_in_tier2
# Equals: input3 - units_in_tier3 if UnitsInMonthAtStartLastHour > Tier_2   \
#         else input2 - units_in_tier3 - Tier_2 if input2 > Tier_2          \
#         else 0


Table 2: Units in tier 2
If:                                                     | 0| 1| 2|
UnitsInMonthAtStartLastHour > Tier_2                    | Y| N| N|
input2 > Tier_2                                         | -| Y| N|
Then:
units_in_tier2 = input3 - units_in_tier3                | X|  |  |
units_in_tier2 = input2 - units_in_tier3 - Tier_2       |  | X|  |
units_in_tier2 = 0                                      |  |  | X|
# .......


# Attribute: units_in_tier3
# Equals: input3 if UnitsInMonthAtStartLastHour > Tier_3  \
#         else input2 - Tier_3 if input2 > Tier_3         \
#         else 0

Table 3: Units in tier 3
If:                                                     | 0| 1| 2|
UnitsInMonthAtStartLastHour > Tier_3                    | Y| N| N|
input2 > Tier_3                                         | -| Y| N|
Then:
units_in_tier3 = input2 - UnitsInMonthAtStartLastHour   | X|  |  |
units_in_tier3 = input2 - Tier_3                        |  | X|  |
units_in_tier3 = 0                                      |  |  | X|
# .......
```

Brief explanation of the functionality of DT5GL.

The built-in reasoning mechanism in DT5GL is based on goal-oriented, backward reasoning (and not data-driven, forward reasoning).

That means: every variable that is present as a condition in a decision table or is included in a formula will be:
1. retrieved from a database (e.g. Sqlite, PostgreSQL, etc.), or
2. prompted to the user (by means of askable_using), or
3. derived from other decision tables (if included in the conclusion part of those tables); or
4. derived from another formula.

At 1: I realised this at the time with the organ donation challenge; see my second contribution.

At 3: conditions in tables can also be conclusions in other tables, which are then called **condition-subtables** (as mentioned in some publications of Prof. J. Vanthienen).

Any attribute/variable that appears in a conclusion in a decision table that
- cannot be linked to a condition in another table and
- is not applied in a formula/expression
is automatically specified as a goal attribute.

This means that each program in dt5gl contains at least one decision table that then provides the goal attributes.

For Solution 1:
The variables in the conditions in this table are all derived from formulas.
The variables in these formulas are all derived from other formulas (e.g., UnitsInMonthAtStartLastHour) or requested from the user (e.g., input1, input2, input3).

For Solution 2:
The variable in the only condition in the main table is derived from a formula.
The variables in this formula are all derived from decision tables.
However, alternatively, the first variable in this formula (units_in_tier1) can also be derived from another formula, as indicated in the source.

Solution 1 is very suitable for a demonstration of the reasoning mechanism.

The first goal the reasoning mechanism tries to prove is:
report is tier3+tier2+tier1
If units_in_tier3 > 0 cannot be proved, then this goal fails.
But the next goal "report is tier3+tier2" and the next goal "report is tier3" also fail.
And so the reasoning mechanism continues with the goal:
report is tier2+tier1.

A full representation of this reasoning process can be found for both solutions in the next section.

Demo Goal-driven/Backward-chaining reasoning with condition subtables.

Solution 1:

Case 3:

```
"Input 3 - units in last hour?"
(int)> 20500
"Input 2 - units in month to end hour?"
(int)> 22000
"Input 1 - all units to end hour?"
(int)> 22000

tier2+tier1
Units in tier 1: 10000
Units in tier 2: 2000
********* Derived data from input *********
All units to start hour...........: 1500
Units in month to start hour......: 1500
Free units remaining at start hour: 8500
```

```
Prove (report is tier3+tier2+tier1)
    >> Conditions Table 0 Rule 0:
    Prove (units_in_tier3 > 0)
        "Input 3 - units in last hour?"
        (int)> 20500


        Prove (value for: UnitsInMonthAtStartLastHour)
            "Input 2 - units in month to end hour?"
            (int)> 22000


        Derived value from formula for UnitsInMonthAtStartLastHour is: 1500
        Prove (value for: Tier_3)
        Derived value from formula for Tier_3 is: 50000
    Derived value from formula for units_in_tier3 is: 0
    Failed
    >> Failed...
Failed
Prove (report is tier3+tier2)
    >> Conditions Table 0 Rule 1:
    >> Failed...
Failed
Prove (report is tier3)
    >> Conditions Table 0 Rule 2:
    >> Failed...
Failed
Prove (report is tier2+tier1)
    >> Conditions Table 0 Rule 3:
    Prove (units_in_tier2 > 0)
        Prove (value for: Tier_2)
        Derived value from formula for Tier_2 is: 20000
    Derived value from formula for units_in_tier2 is: 2000
    Succeed
    Prove (units_in_tier1 > 0)
        Prove (value for: FreeUnitsAtStartLastHour)
            Prove (value for: Free_Units)
            Derived value from formula for Free_Units is: 10000
            Prove (value for: TotalUnitsAtStartLastHour)
                "Input 1 - all units to end hour?"
                (int)> 22000


            Derived value from formula for TotalUnitsAtStartLastHour is: 1500
        Derived value from formula for FreeUnitsAtStartLastHour is: 8500
    Derived value from formula for units_in_tier1 is: 10000
    Succeed
    >> Succeed...
tier2+tier1
Units in tier 1: 10000
Units in tier 2: 2000
Succeed
Prove ('show derived data from input')
    >> Conditions Table 0 Rule 0:
    >> Failed...
    >> Conditions Table 0 Rule 1:
    >> Failed...
    >> Conditions Table 0 Rule 2:
    >> Failed...
    >> Conditions Table 0 Rule 3:
    >> Succeed...
********* Derived data from input *********
All units to start hour...........: 1500
Units in month to start hour......: 1500
Free units remaining at start hour: 8500
Succeed
```

Solution 2:

Case 3:

```
"Input 2 - units in month to end hour?"
(int)> 22000
"Input 3 - units in last hour?"
(int)> 20500
"Input 1 - all units to end hour?"
(int)> 22000

Number of units consumed in the last hour: 12000
Units in tier 1: 10000
Units in tier 2: 2000
Units in tier 3: 0
********* Derived data from input *********
All units to start hour...........: 1500
Units in month to start hour......: 1500
Free units remaining at start hour: 8500
```

```
Prove (report is alltiers)
    >> Conditions Table 0 Rule 0:
    Prove (units_in_all_tiers > 0)
        Prove (value for: units_in_tier1)
            >> Conditions Table 2 Rule 0:
            Prove (units_in_tier1_bc <= FreeUnitsAtStartLastHour)
                >> Conditions Table 1 Rule 0:
                Prove (Start > Tier_2)
                    "Input 2 - units in month to end hour?"
                    (int)> 22000


                    "Input 3 - units in last hour?"
                    (int)> 20500


                Derived value from formula for Start is: 1500
                    Prove (value for: Tier_2)
                    Derived value from formula for Tier_2 is: 20000
                Failed
                >> Failed...
                >> Conditions Table 1 Rule 1:
                >> Succeed...
                Prove (value for: FreeUnitsAtStartLastHour)
                    Prove (value for: Free_Units)
                    Derived value from formula for Free_Units is: 10000
                    Prove (value for: TotalUnitsAtStartLastHour)
                        "Input 1 - all units to end hour?"
                        (int)> 22000


                    Derived value from formula for TotalUnitsAtStartLastHour is:
                    1500
                Derived value from formula for FreeUnitsAtStartLastHour is: 8500
            Failed
            >> Failed...
            >> Conditions Table 2 Rule 1:
            >> Succeed...
        Derived value for units_in_tier1 is: 10000
        Prove (value for: units_in_tier2)
            >> Conditions Table 3 Rule 3:
                Prove (value for: Tier_3)
                Derived value from formula for Tier_3 is: 50000
            >> Failed...
            >> Conditions Table 3 Rule 1:
            >> Failed...
            >> Conditions Table 3 Rule 4:
            >> Succeed...
        Derived value for units_in_tier2 is: 2000
        Prove (value for: units_in_tier3)
            >> Conditions Table 4 Rule 0:
            >> Failed...
            >> Conditions Table 4 Rule 1:
            >> Failed...
            >> Conditions Table 4 Rule 2:
            >> Succeed...
        Derived value for units_in_tier3 is: 0
    Derived value from formula for units_in_all_tiers is: 12000
    Succeed
    >> Succeed...
Number of units consumed in the last hour: 12000
Units in tier 1: 10000
Units in tier 2: 2000
Units in tier 3: 0
Succeed
```

```
Prove ('show derived data from input')
    >> Conditions Table 0 Rule 0:
    >> Succeed...
********* Derived data from input *********
All units to start hour...........: 1500
Units in month to start hour......: 1500
Free units remaining at start hour: 8500
Succeed
```