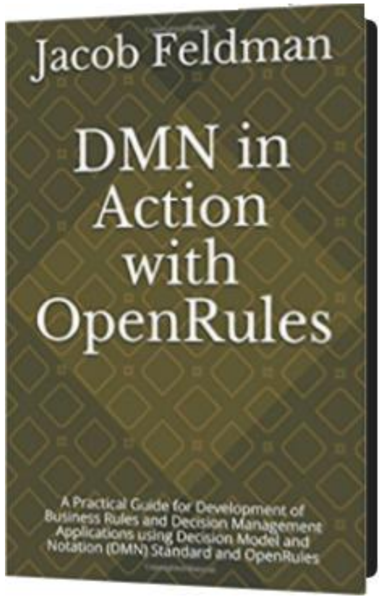
Orlando, Nov 6-10, 2017

How Business Analysts Build Executable Decision Models with DMN Standard without Programming

Presenter: Dr. Jacob Feldman
OpenRules Inc., CTO
jacobfeldman@openrules.com
www.OpenRules.com

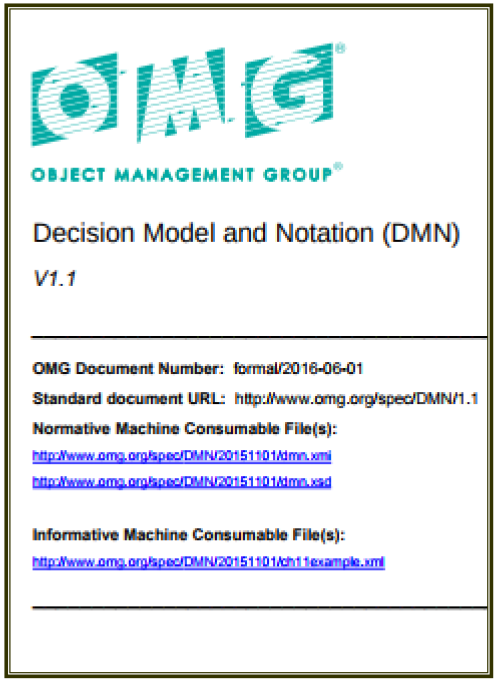

Jacob Feldman
DMN in Action with OpenRules
A Practical Guide for Development of Business Rules and Decision Management Applications using Decision Model and Notation (DMN) Standard and OpenRules

My Message: "Keep DMN Simple"

Presentation Outline:

- About practical use of the DMN Standard
- Replacing DMN programming constructs with traditional, user-friendly decision tables
- Examples of DMN-based decision models
  - with programming (CL3)
  - without programming (CL2)

DMN Standard


OBJECT MANAGEMENT GROUP®

Decision Model and Notation (DMN)

V1.1

OMG Document Number: formal/2016-06-01

Standard document URL: http://www.omg.org/spec/DMN/1.1

Normative Machine Consumable File(s):

http://www.omg.org/spec/DMN/20151101/dmn.xmi

http://www.omg.org/spec/DMN/20151101/dmn.xsd

Informative Machine Consumable File(s):

http://www.omg.org/spec/DMN/20151101/ch11example.xml

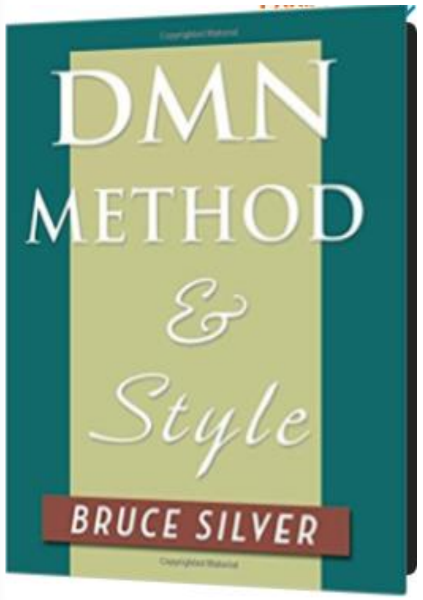
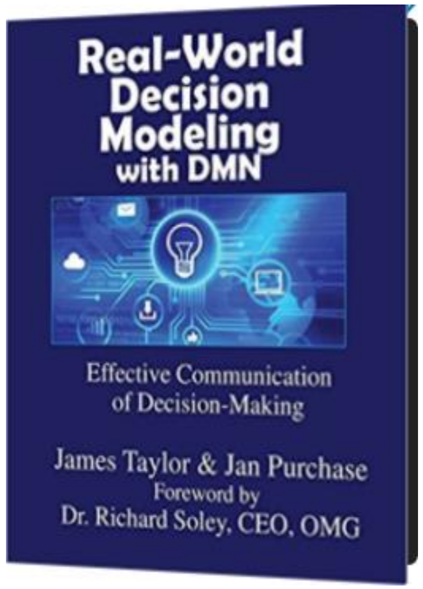
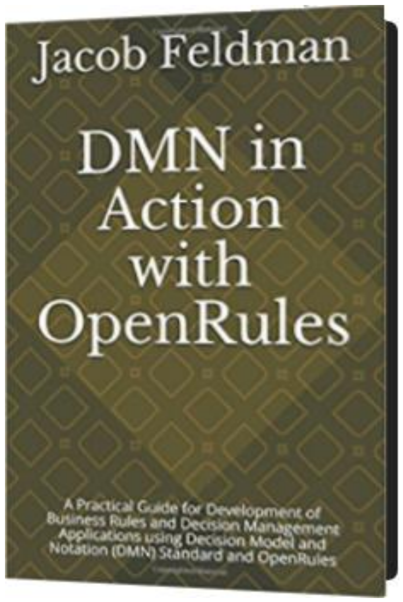
DMN stands for “**D**ecision **M**odel and **N**otation”

- Deals with Operational Business Decisioning Problems
- Oriented to Business Analysts

DMN – Decision Model and Notation

- DMN is an official OMG standard since 2014
- Specifies major Decision Modeling constructs
- Current release 1.1 supports DMN XML interchange format
- Next Release 1.2 is expected in Q1 2018

Recent DMN Books

Bruce Silver	James Taylor Jan Purchase	Jacob Feldman
		
2016	2016	2017

DMN
METHOD
&
Style
BRUCE SILVER

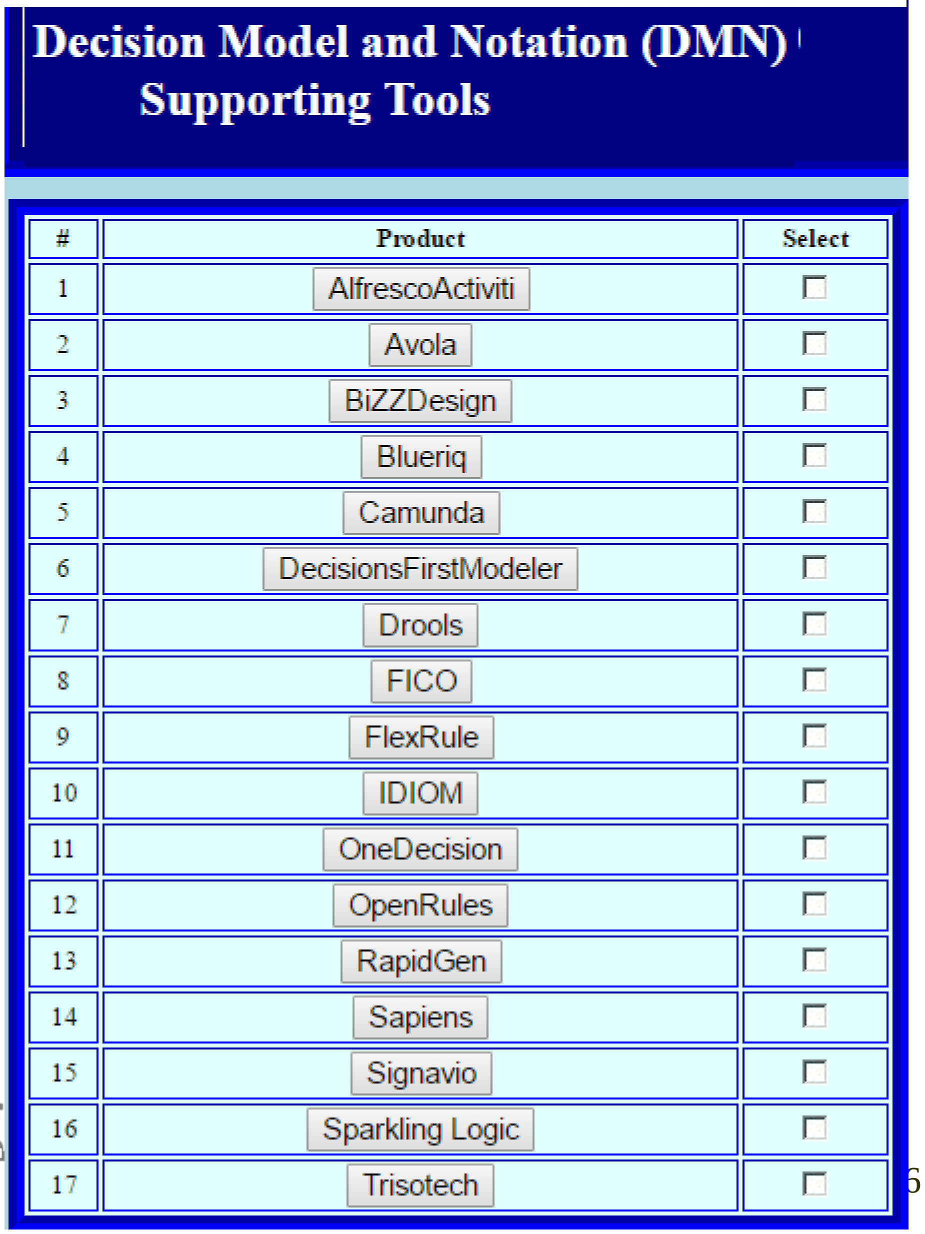
Real-World Decision Modeling with DMN
Effective Communication of Decision-Making
James Taylor & Jan Purchase
Foreword by
Dr. Richard Soley, CEO, OMG

Jacob Feldman
DMN in Action with OpenRules
A Practical Guide for Development of Business Rules and Decision Management Applications using Decision Model and Notation (DMN) Standard and OpenRules

DECISION MANAGEMENT COMMUNITY

Shared Information Sources for Decision Management Practitioners

www.DMCommunity.org



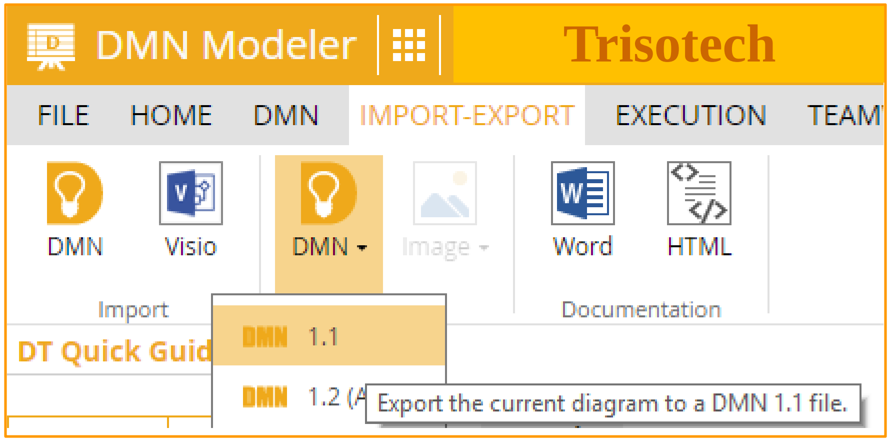
Decision Model and Notation (DMN) Supporting Tools

#	Product	Select
1	AlfrescoActiviti	☐
2	Avola	☐
3	BiZZDesign	☐
4	Blueriq	☐
5	Camunda	☐
6	DecisionsFirstModeler	☐
7	Drools	☐
8	FICO	☐
9	FlexRule	☐
10	IDIOM	☐
11	OneDecision	☐
12	OpenRules	☐
13	RapidGen	☐
14	Sapiens	☐
15	Signavio	☐
16	Sparkling Logic	☐
17	Trisotech	☐

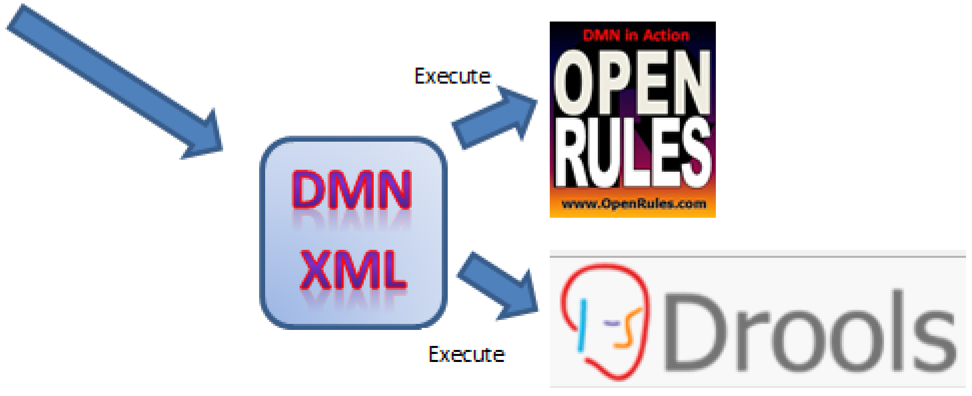
- Many vendors already announced DMN support

www.DMCommunity.org

DMN Interchange

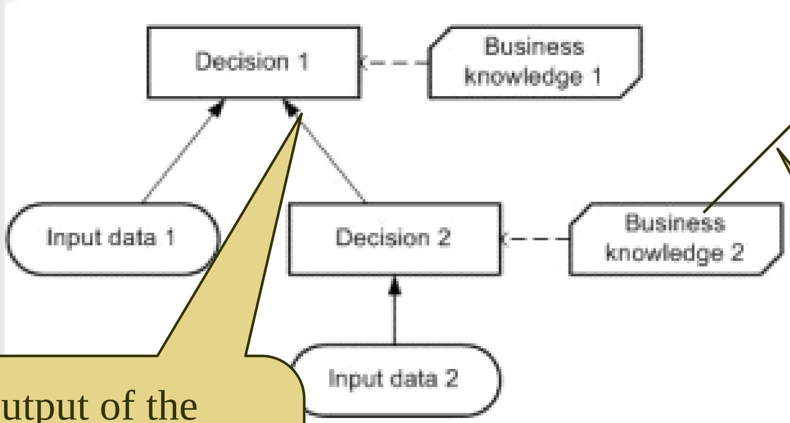


Export the current diagram to a DMN 1.1 file.

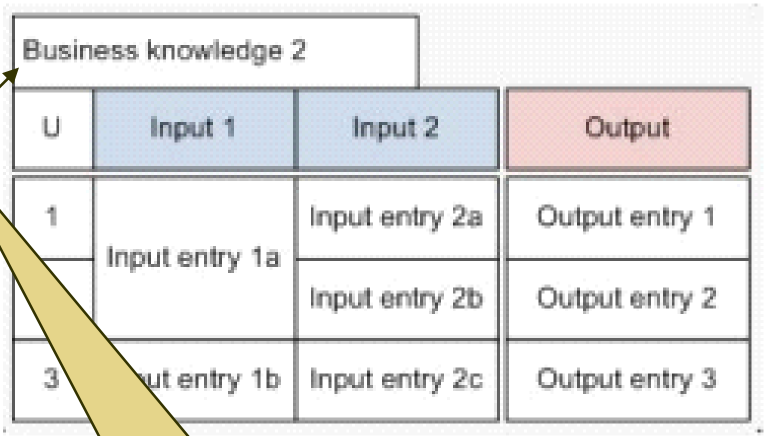


Business Analysts Like Graphical Representations:

- DMN Decision Requirement Diagrams
- DMN Decision Tables



Output of the Decision-2 is used as an input for the Decision-1



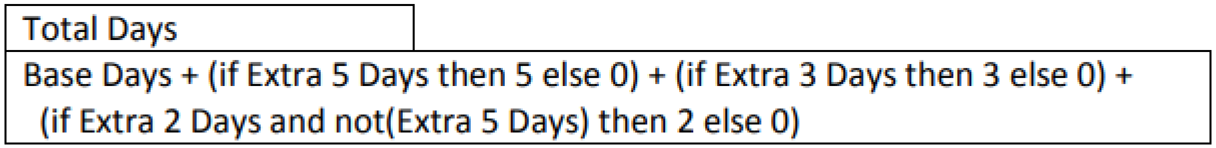
Business knowledge 2

U	Input 1	Input 2	Output
1	Input entry 1a	Input entry 2a	Output entry 1
		Input entry 2b	Output entry 2
3	ut entry 1b	Input entry 2c	Output entry 3

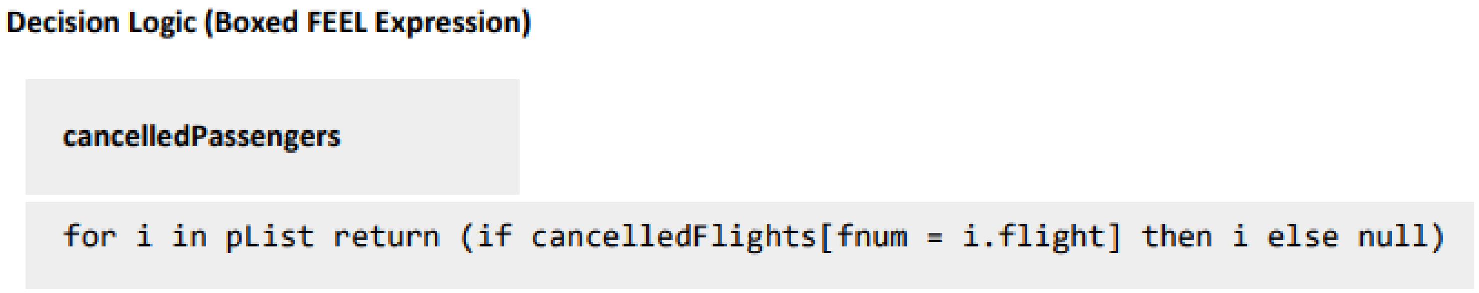
This Decision Table represents Business Knowledge-2 (decision logic)

Business Analysts Don’t Like “Programming” Constructs:

- DMN FEEL language includes:
  – If-Then-Else, Loops, Boxed Expressions, Functions with Parameters, …
- Examples:



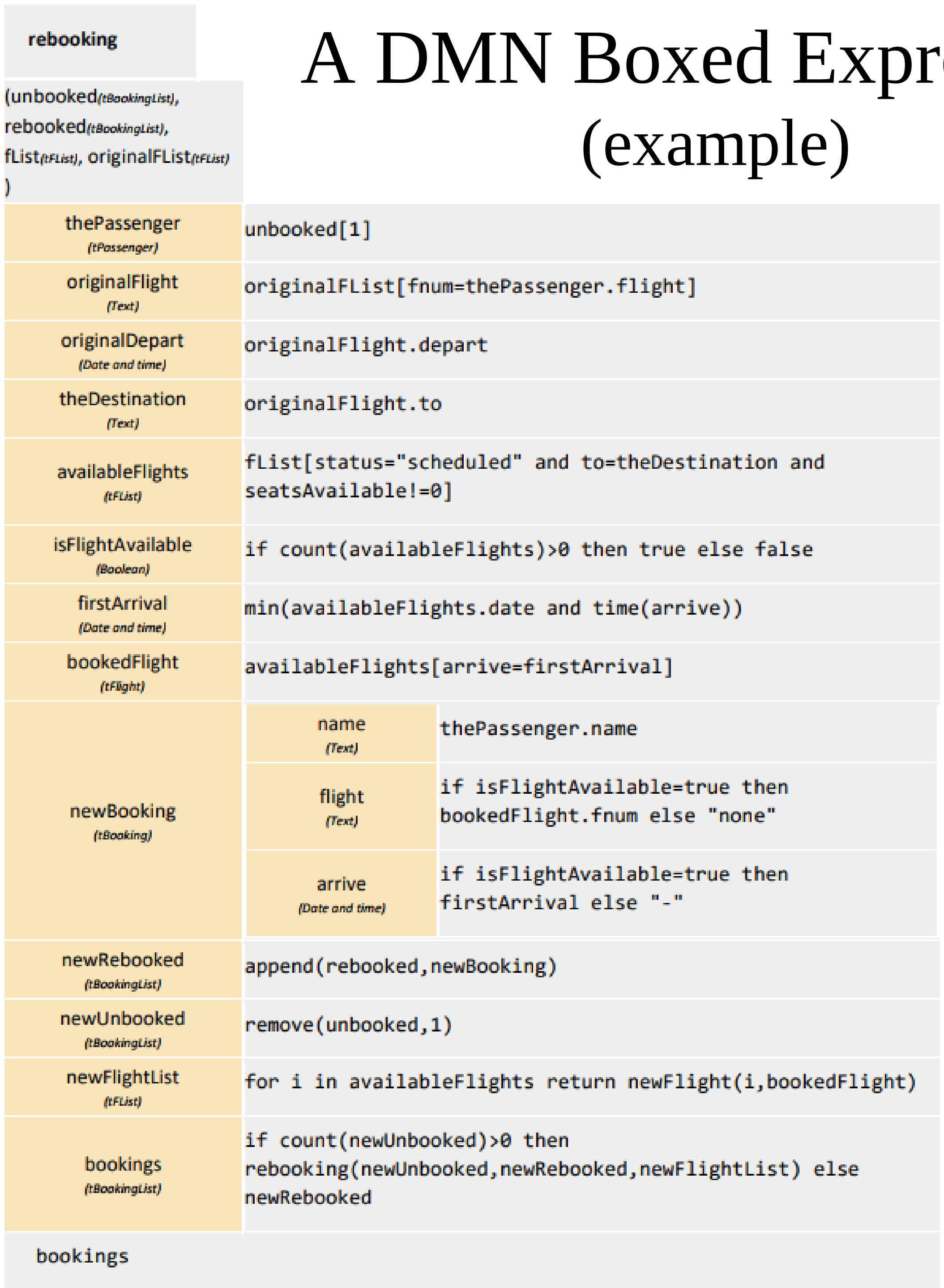
Total Days
Base Days + (if Extra 5 Days then 5 else 0) + (if Extra 3 Days then 3 else 0) + (if Extra 2 Days and not(Extra 5 Days) then 2 else 0)



Decision Logic (Boxed FEEL Expression)

cancelledPassengers
`for i in pList return (if cancelledFlights[fnum = i.flight] then i else null)`

A DMN Boxed Expression (example)



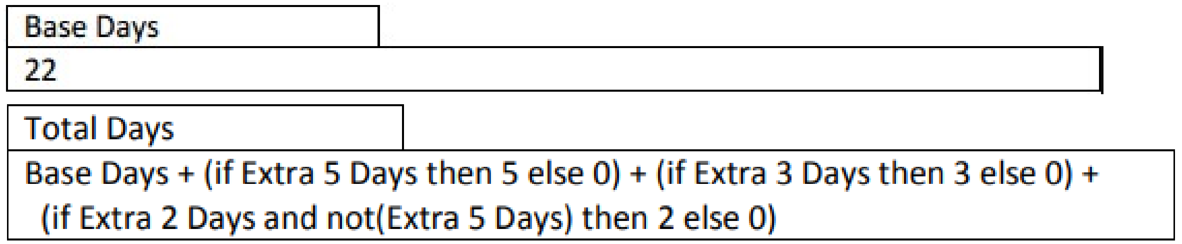
rebooking

(unbooked*(tBookingList)*, rebooked*(tBookingList)*, fList*(tFList)*, originalFList*(tFList)*)

thePassenger *(tPassenger)*		`unbooked[1]`
originalFlight *(Text)*		`originalFList[fnum=thePassenger.flight]`
originalDepart *(Date and time)*		`originalFlight.depart`
theDestination *(Text)*		`originalFlight.to`
availableFlights *(tFList)*		`fList[status="scheduled" and to=theDestination and seatsAvailable!=0]`
isFlightAvailable *(Boolean)*		`if count(availableFlights)>0 then true else false`
firstArrival *(Date and time)*		`min(availableFlights.date and time(arrive))`
bookedFlight *(tFlight)*		`availableFlights[arrive=firstArrival]`
newBooking *(tBooking)*	name *(Text)*	`thePassenger.name`
	flight *(Text)*	`if isFlightAvailable=true then bookedFlight.fnum else "none"`
	arrive *(Date and time)*	`if isFlightAvailable=true then firstArrival else "-"`
newRebooked *(tBookingList)*		`append(rebooked,newBooking)`
newUnbooked *(tBookingList)*		`remove(unbooked,1)`
newFlightList *(tFList)*		`for i in availableFlights return newFlight(i,bookedFlight)`
bookings *(tBookingList)*		`if count(newUnbooked)>0 then rebooking(newUnbooked,newRebooked,newFlightList) else newRebooked`
`bookings`		

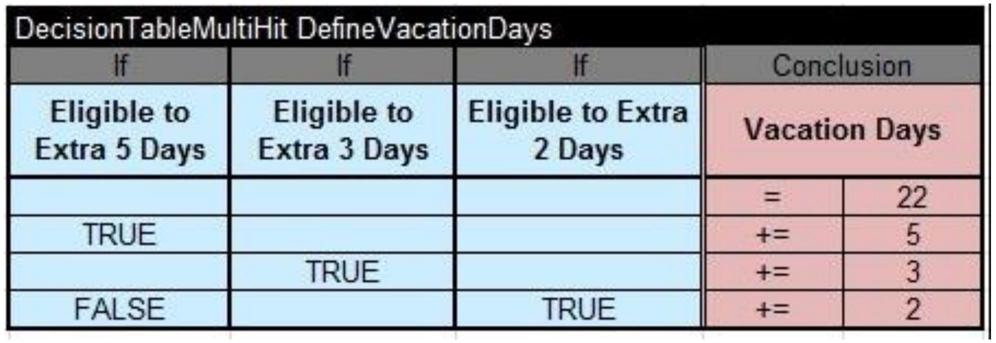
Avoiding Programming

- In many practical situations we may replace DMN programming constructs with business-oriented graphical representations, e.g.:



Base Days
22

Total Days
Base Days + (if Extra 5 Days then 5 else 0) + (if Extra 3 Days then 3 else 0) + (if Extra 2 Days and not(Extra 5 Days) then 2 else 0)



DecisionTableMultiHit DefineVacationDays

If	If	If	Conclusion	
Eligible to Extra 5 Days	Eligible to Extra 3 Days	Eligible to Extra 2 Days	Vacation Days	
			=	22
TRUE			+=	5
	TRUE		+=	3
FALSE		TRUE	+=	2

Objective

- Replacing DMN programming constructs with more traditional decision tables
- We will show DMN-based implementations of several popular decision models:
  - With and Without Programming

Decision Modeling Constructs

- Core Constructs - Conformance Level 2
  - Diagrams with Logical Connections (information requirements)
  - Decision Tables
  - Basic Expression Language (S-FEEL)
- Advanced Constructs - Conformance Level 3
  - Boxed Expressions (FEEL functions with parameters, contexts, if-then-else, for..return loops, filters, sorting, recursion, …)

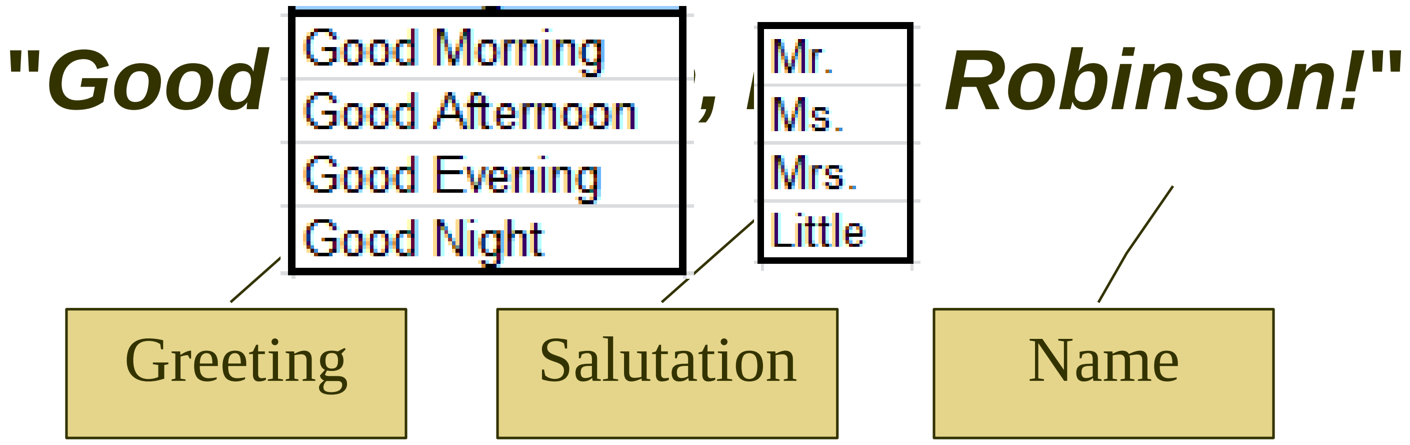
Decision Modeling with DMN

- The best way to understand DMN is to build and test real Decision Models
- We will consider several decision models:
  - Decision Hello Customer - trivial
  - Decision Vacation Days – a slightly more complex
  - Decision Flight Rebooking - complex

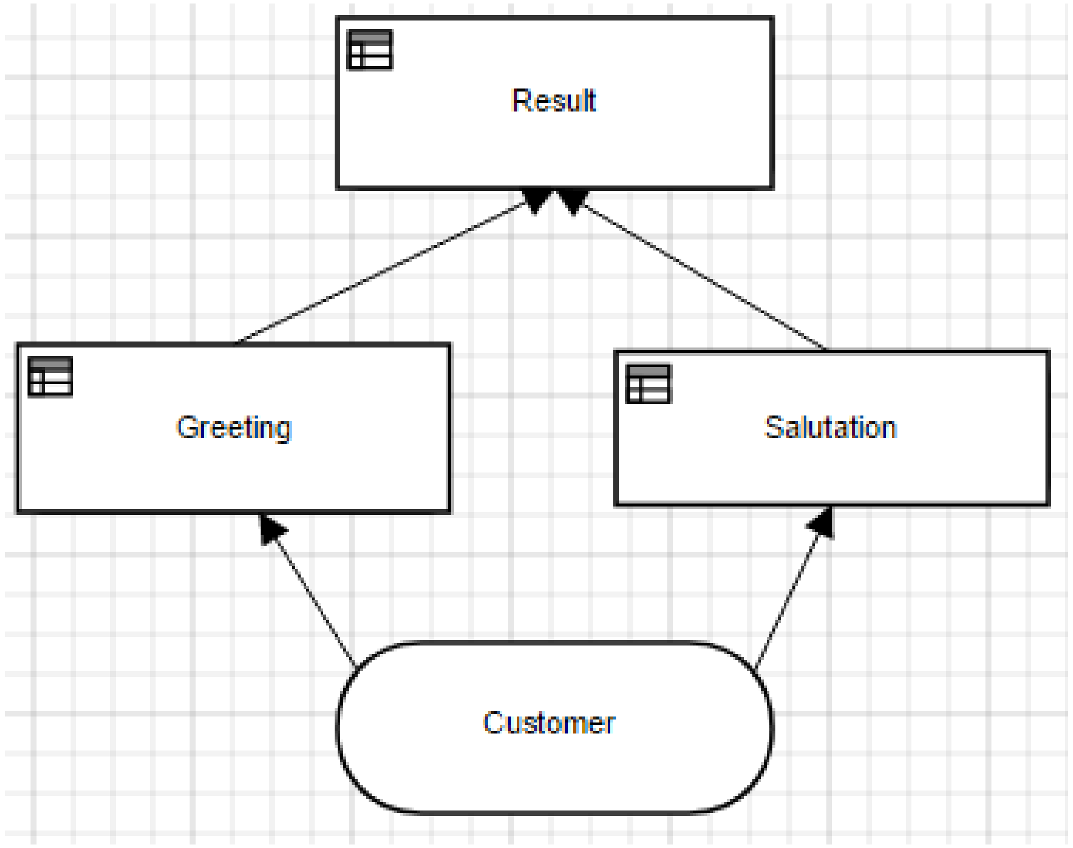
Sample Decision Model "Determine Customer Greeting"

- Decide how to greet a particular customer during different times of the day (think IVR)
- Test:
  - Customer: Robinson is a married woman
  - Time of the day: 14:25 pm
  - Expected decision: "*Good Afternoon, Mrs. Robinson!*"

Starting with a Decision

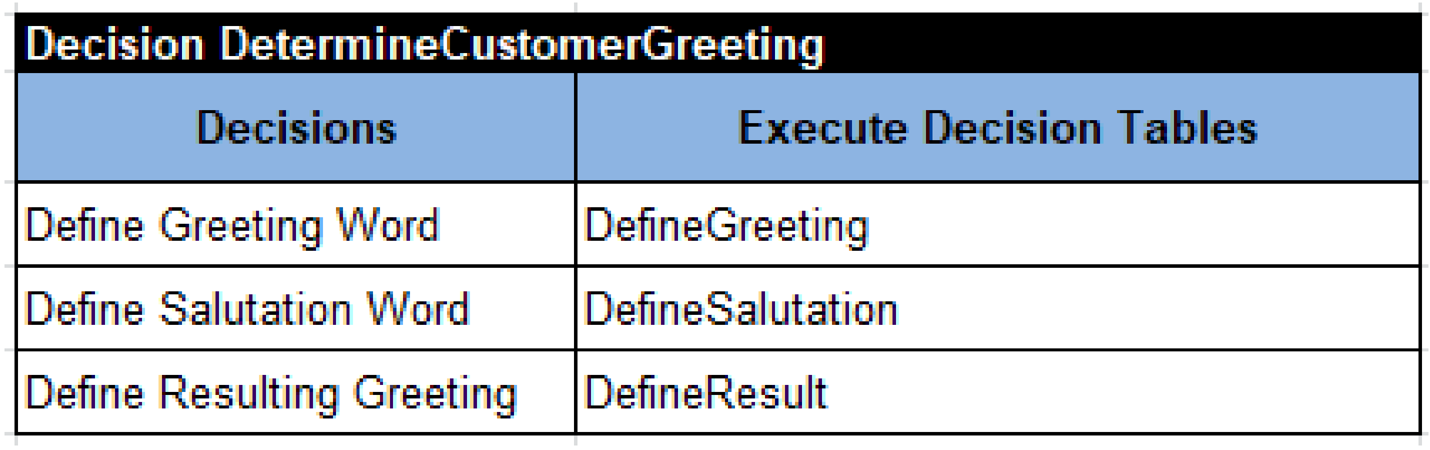


Decision Requirements Diagram



DRD as a Tabular Decision

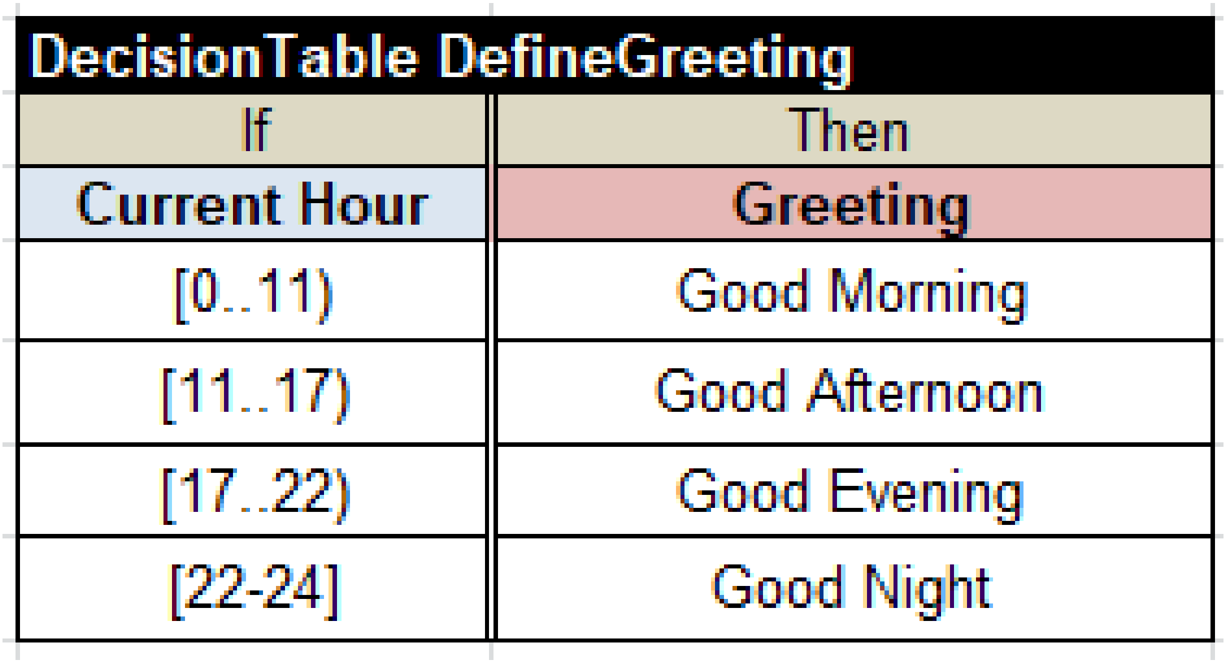
- Our DRD may be presented in OpenRules as a table:



Decision DetermineCustomerGreeting

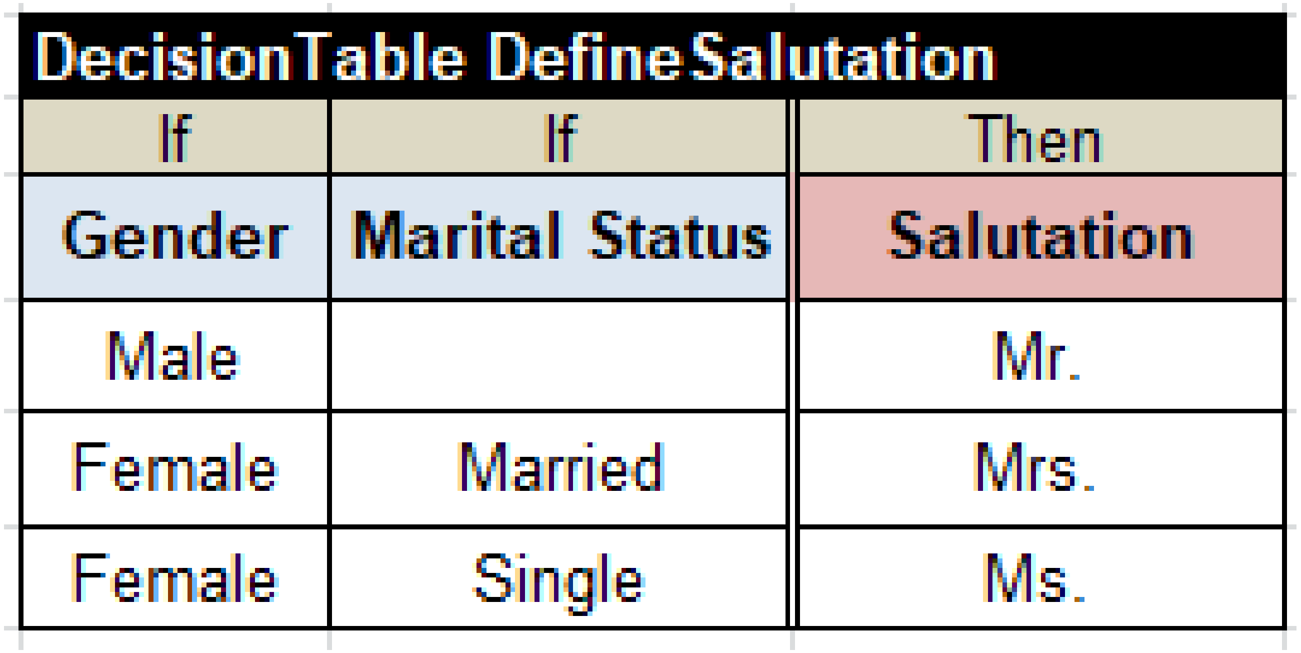
Decisions	Execute Decision Tables
Define Greeting Word	DefineGreeting
Define Salutation Word	DefineSalutation
Define Resulting Greeting	DefineResult

Decision Table “DefineGreeting”



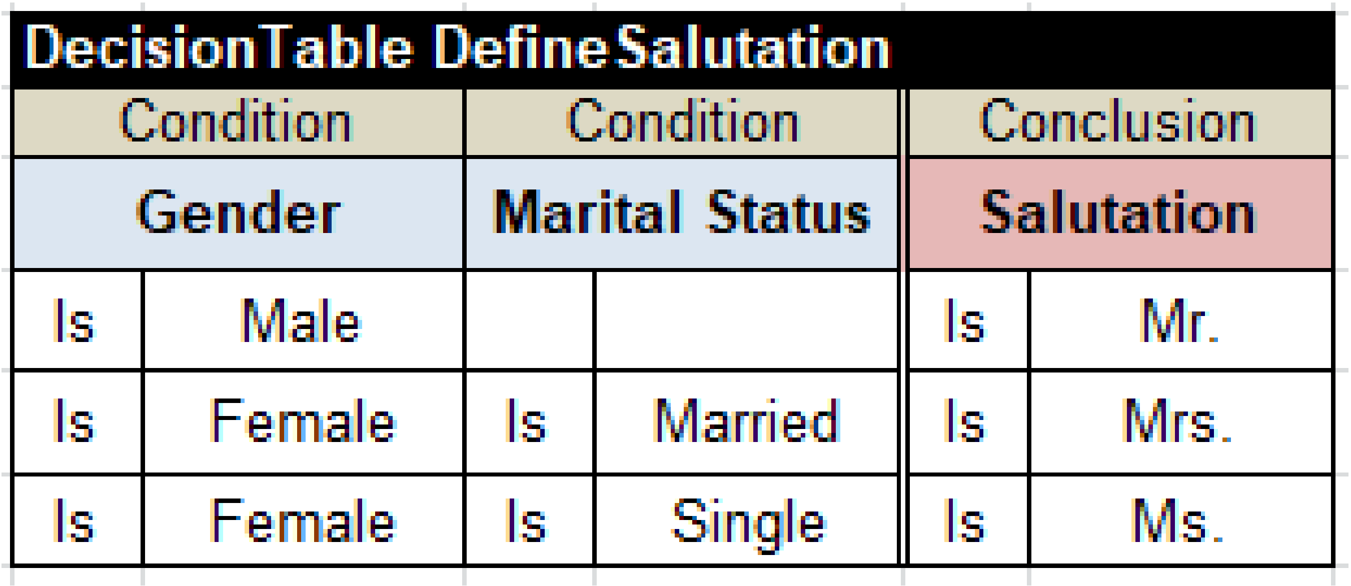
DecisionTable DefineGreeting	
If	Then
Current Hour	Greeting
[0..11)	Good Morning
[11..17)	Good Afternoon
[17..22)	Good Evening
[22-24]	Good Night

Decision Table "DefineSalutation"



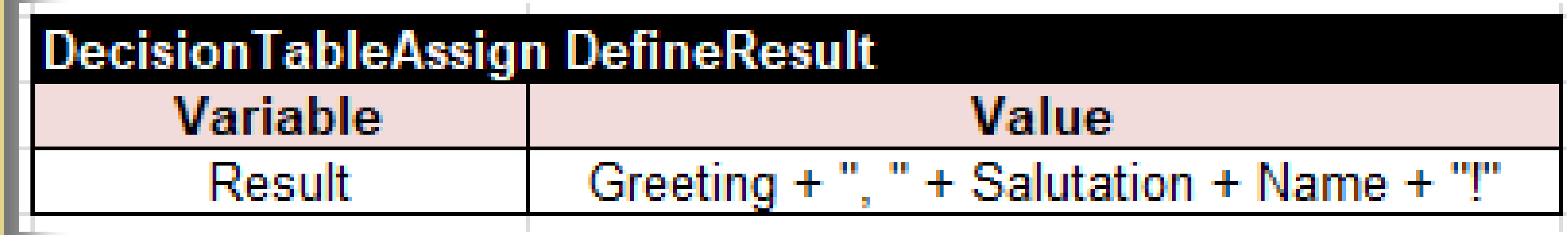
DecisionTable DefineSalutation		
If	If	Then
Gender	Marital Status	Salutation
Male		Mr.
Female	Married	Mrs.
Female	Single	Ms.

Decision Table “DefineSalutation” (alternative representation)



DecisionTable DefineSalutation					
Condition		Condition		Conclusion	
Gender		Marital Status		Salutation	
Is	Male			Is	Mr.
Is	Female	Is	Married	Is	Mrs.
Is	Female	Is	Single	Is	Ms.

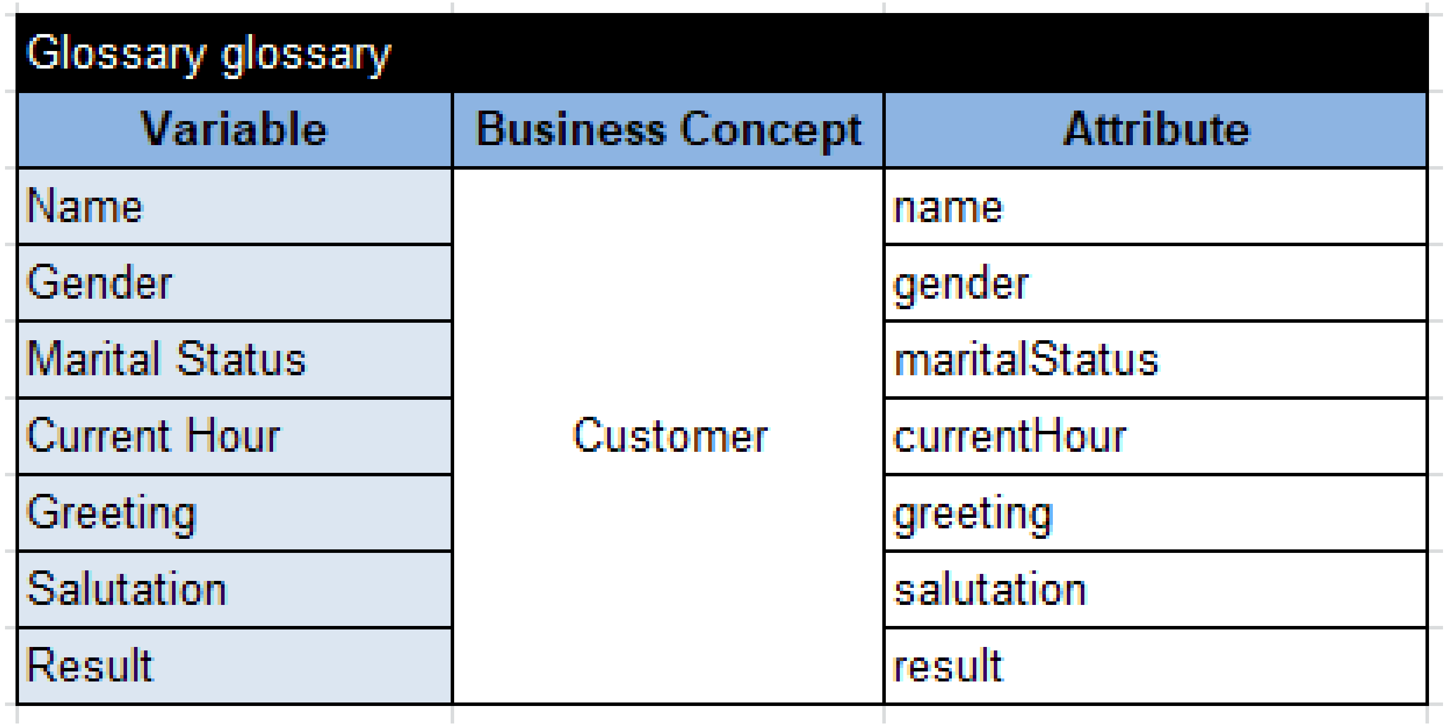
Decision Table “DefineResult”



DecisionTableAssign DefineResult	
Variable	**Value**
Result	Greeting + ", " + Salutation + Name + "!"

- This is an example of a simple DMN FEEL expression

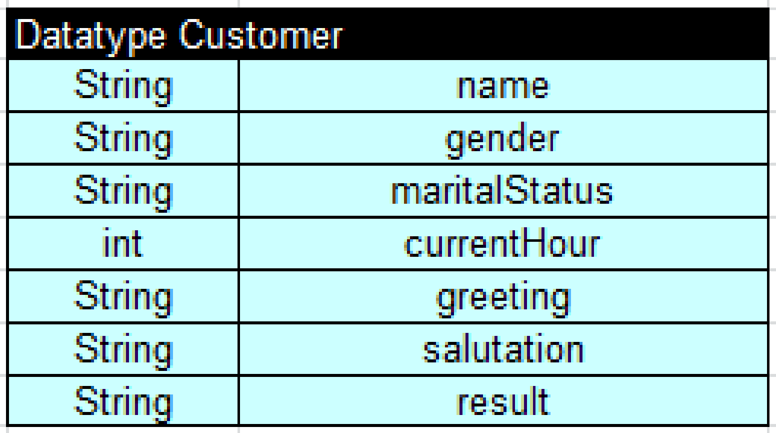
Defining Business Glossary



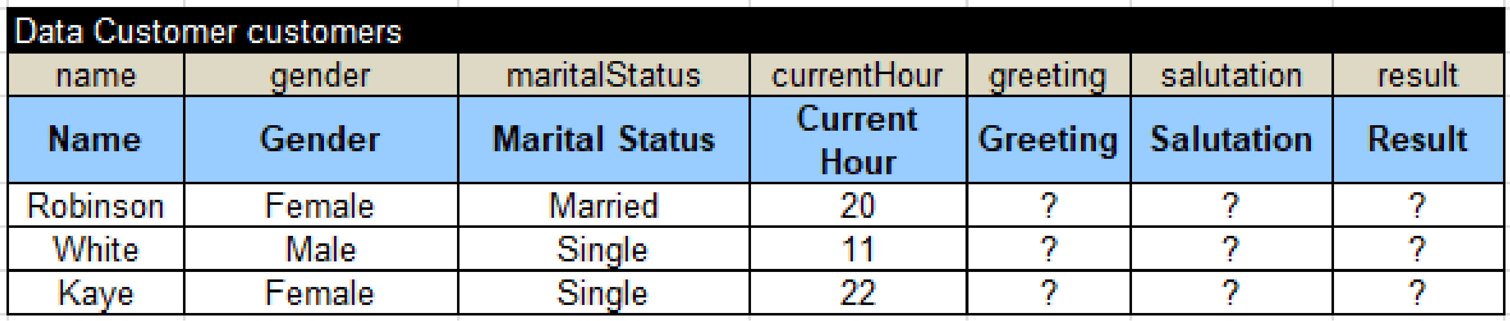
Glossary glossary

Variable	Business Concept	Attribute
Name		name
Gender		gender
Marital Status		maritalStatus
Current Hour	Customer	currentHour
Greeting		greeting
Salutation		salutation
Result		result

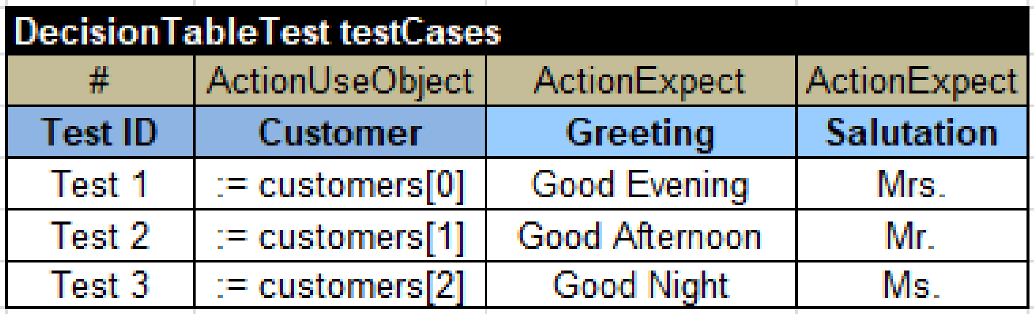
Defining Test Data (in Excel)



Datatype Customer	
String	name
String	gender
String	maritalStatus
int	currentHour
String	greeting
String	salutation
String	result

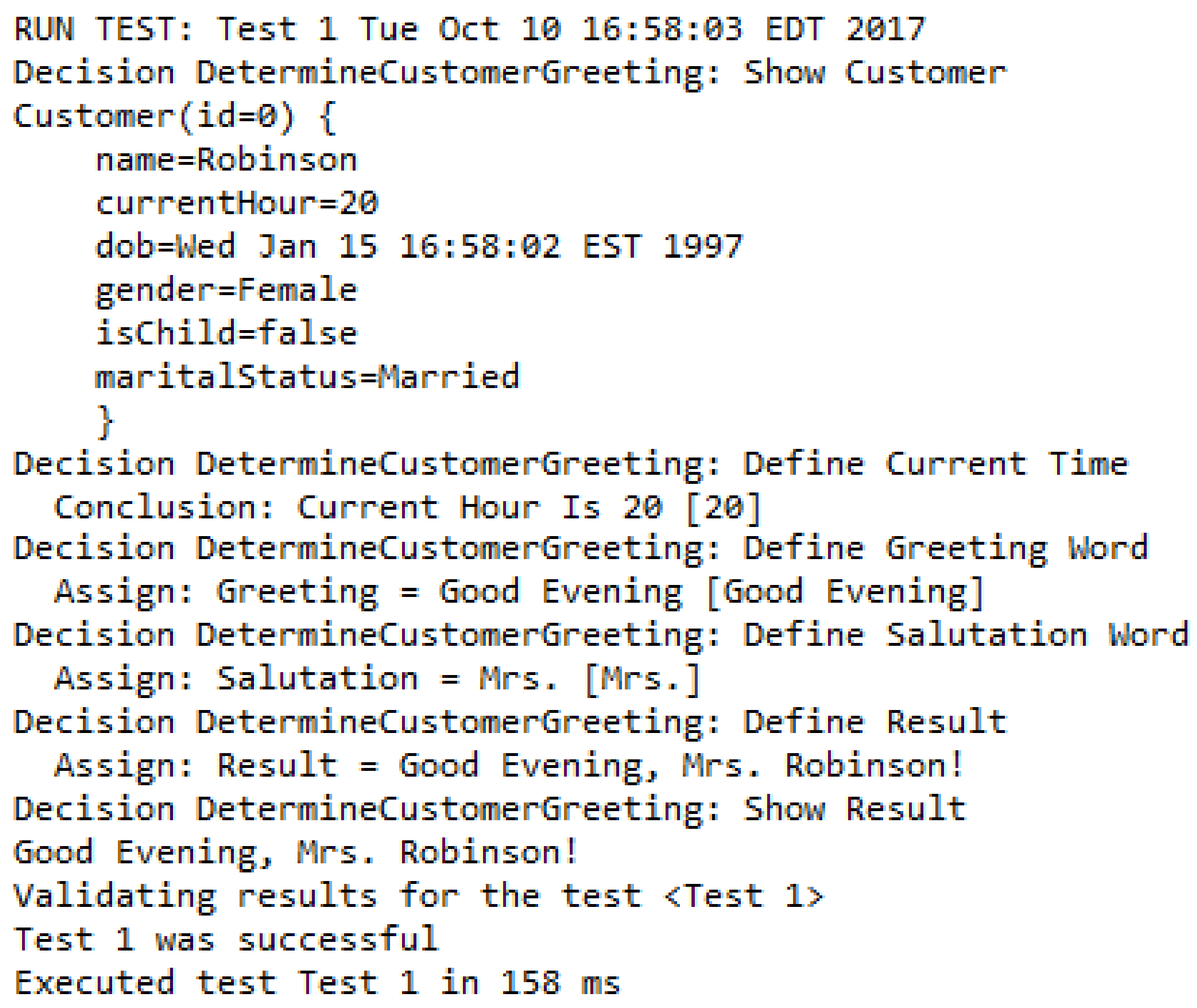


Data Customer customers						
name	gender	maritalStatus	currentHour	greeting	salutation	result
Name	**Gender**	**Marital Status**	**Current Hour**	**Greeting**	**Salutation**	**Result**
Robinson	Female	Married	20	?	?	?
White	Male	Single	11	?	?	?
Kaye	Female	Single	22	?	?	?



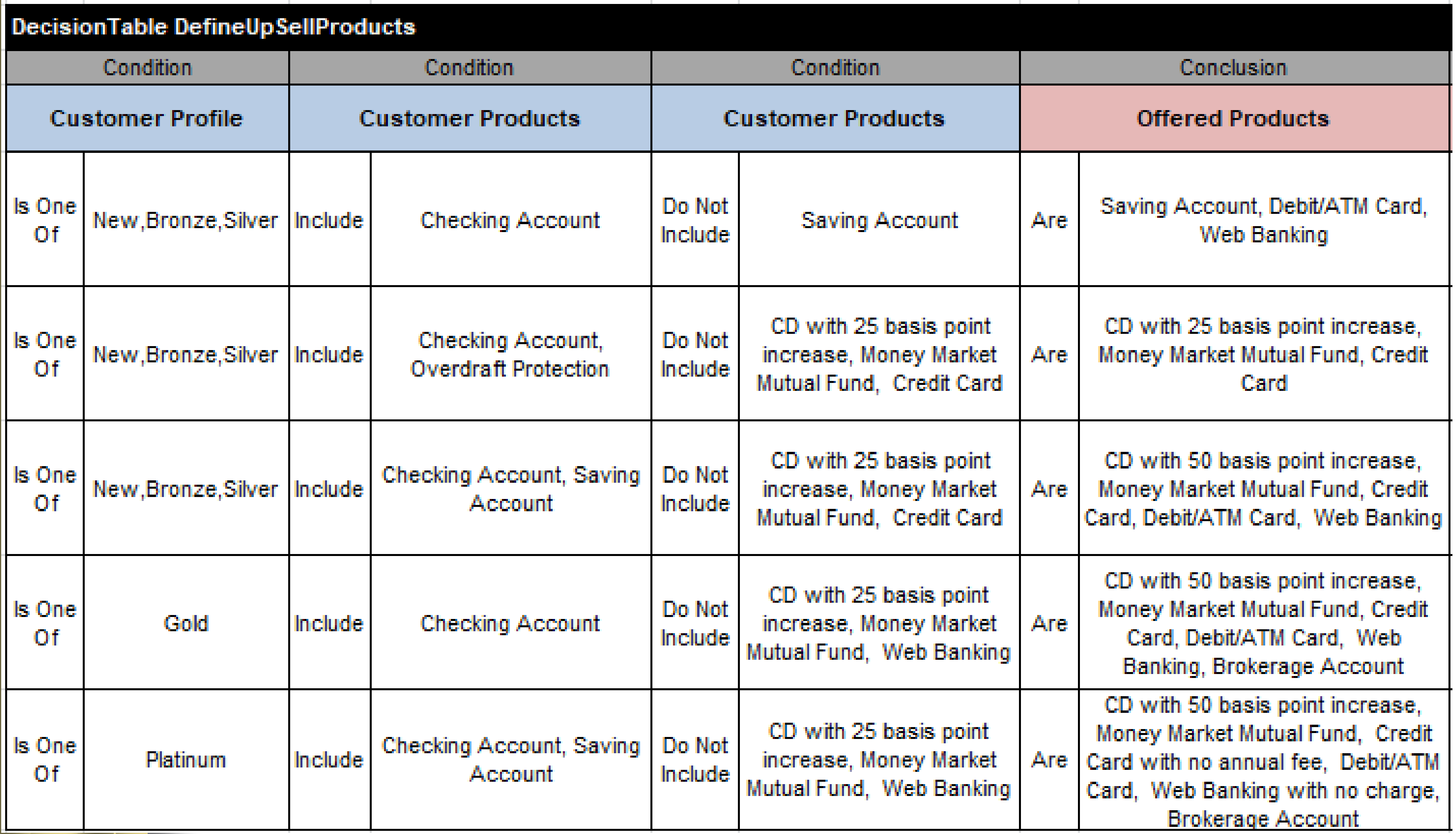
DecisionTableTest testCases			
#	ActionUseObject	ActionExpect	ActionExpect
Test ID	**Customer**	**Greeting**	**Salutation**
Test 1	:= customers[0]	Good Evening	Mrs.
Test 2	:= customers[1]	Good Afternoon	Mr.
Test 3	:= customers[2]	Good Night	Ms.

Executing Decision Model



```
RUN TEST: Test 1 Tue Oct 10 16:58:03 EDT 2017
Decision DetermineCustomerGreeting: Show Customer
Customer(id=0) {
    name=Robinson
    currentHour=20
    dob=Wed Jan 15 16:58:02 EST 1997
    gender=Female
    isChild=false
    maritalStatus=Married
    }
Decision DetermineCustomerGreeting: Define Current Time
  Conclusion: Current Hour Is 20 [20]
Decision DetermineCustomerGreeting: Define Greeting Word
  Assign: Greeting = Good Evening [Good Evening]
Decision DetermineCustomerGreeting: Define Salutation Word
  Assign: Salutation = Mrs. [Mrs.]
Decision DetermineCustomerGreeting: Define Result
  Assign: Result = Good Evening, Mrs. Robinson!
Decision DetermineCustomerGreeting: Show Result
Good Evening, Mrs. Robinson!
Validating results for the test <Test 1>
Test 1 was successful
Executed test Test 1 in 158 ms
```

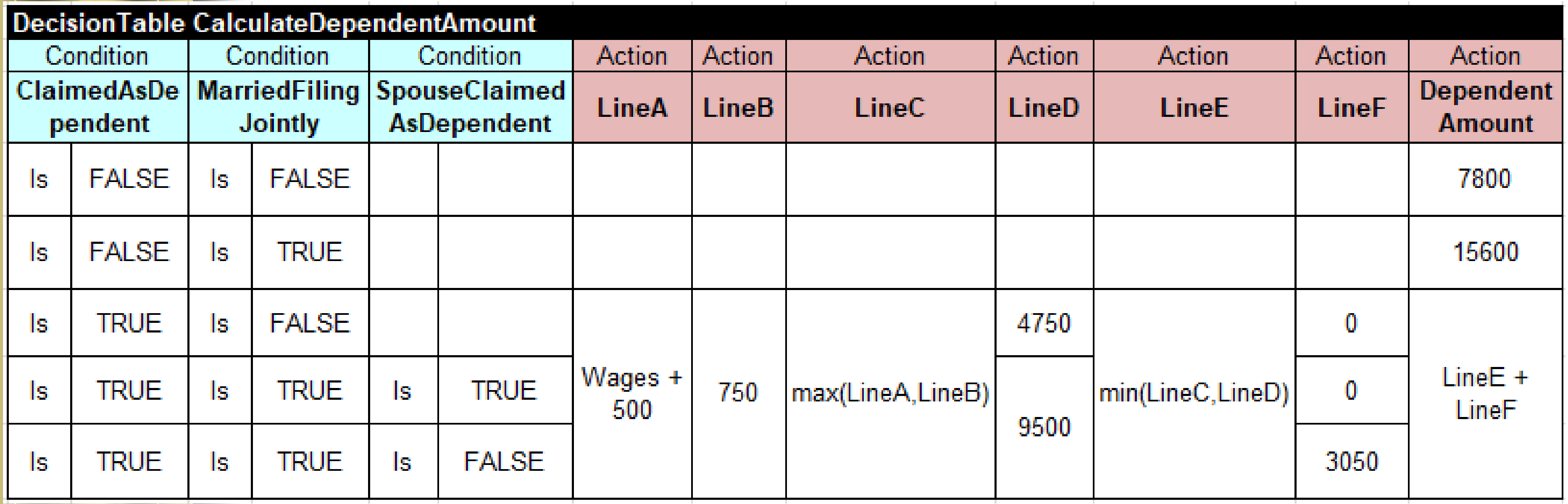
More Complex Decision Tables



DecisionTable DefineUpSellProducts							
Condition		Condition		Condition		Conclusion	
Customer Profile		Customer Products		Customer Products		Offered Products	
Is One Of	New,Bronze,Silver	Include	Checking Account	Do Not Include	Saving Account	Are	Saving Account, Debit/ATM Card, Web Banking
Is One Of	New,Bronze,Silver	Include	Checking Account, Overdraft Protection	Do Not Include	CD with 25 basis point increase, Money Market Mutual Fund, Credit Card	Are	CD with 25 basis point increase, Money Market Mutual Fund, Credit Card
Is One Of	New,Bronze,Silver	Include	Checking Account, Saving Account	Do Not Include	CD with 25 basis point increase, Money Market Mutual Fund, Credit Card	Are	CD with 50 basis point increase, Money Market Mutual Fund, Credit Card, Debit/ATM Card, Web Banking
Is One Of	Gold	Include	Checking Account	Do Not Include	CD with 25 basis point increase, Money Market Mutual Fund, Web Banking	Are	CD with 50 basis point increase, Money Market Mutual Fund, Credit Card, Debit/ATM Card, Web Banking, Brokerage Account
Is One Of	Platinum	Include	Checking Account, Saving Account	Do Not Include	CD with 25 basis point increase, Money Market Mutual Fund, Web Banking	Are	CD with 50 basis point increase, Money Market Mutual Fund, Credit Card with no annual fee, Debit/ATM Card, Web Banking with no charge, Brokerage Account

1040EZ Decision Table

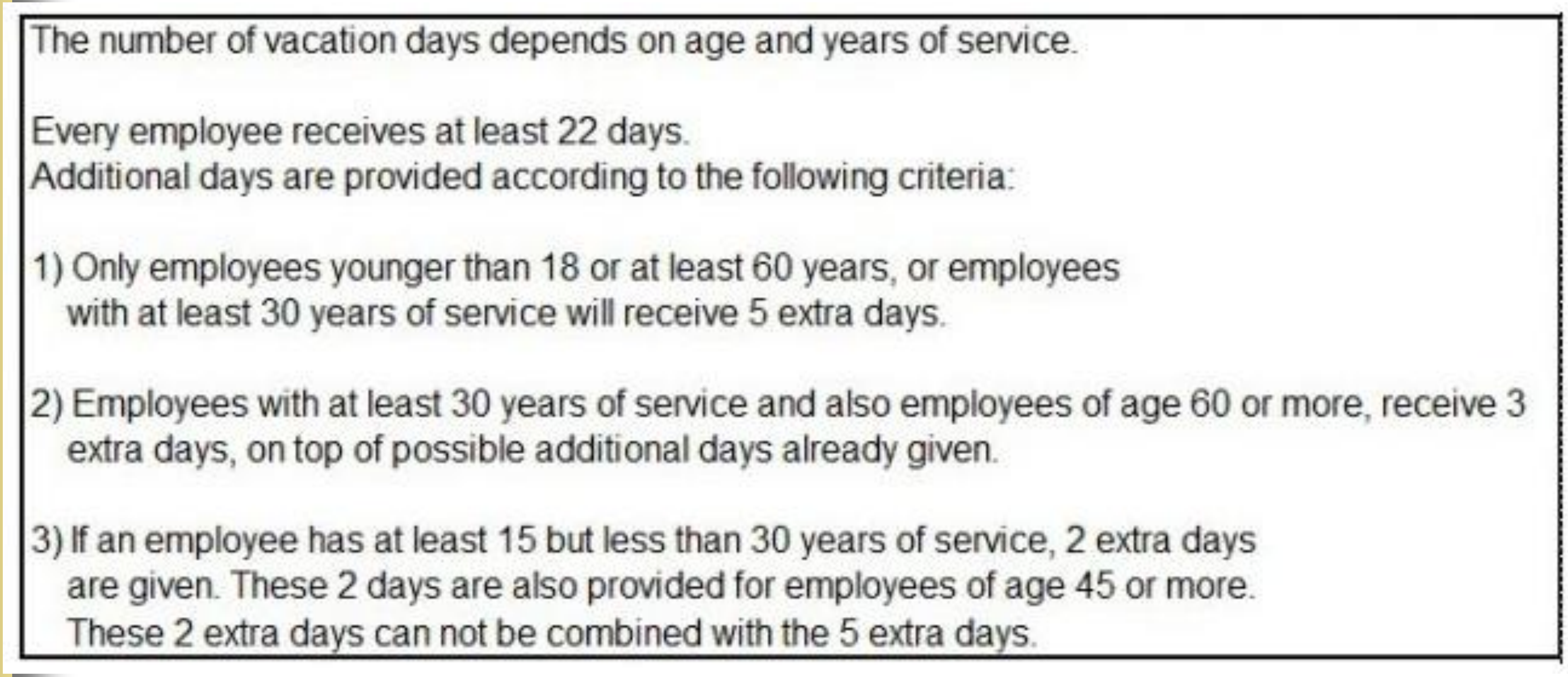
- Decision Table with Calculations



DecisionTable CalculateDependentAmount												
Condition		Condition		Condition		Action	Action	Action	Action	Action	Action	Action
ClaimedAsDependent		MarriedFilingJointly		SpouseClaimedAsDependent		LineA	LineB	LineC	LineD	LineE	LineF	Dependent Amount
Is	FALSE	Is	FALSE									7800
Is	FALSE	Is	TRUE									15600
Is	TRUE	Is	FALSE			Wages + 500	750	max(LineA,LineB)	4750	min(LineC,LineD)	0	LineE + LineF
Is	TRUE	Is	TRUE	Is	TRUE				9500		0	
Is	TRUE	Is	TRUE	Is	FALSE						3050	

Decision Model “Vacation Days”

- DMCommunity.org Challenge Jan-2016 provides 20 different solutions for this problem:



The number of vacation days depends on age and years of service.

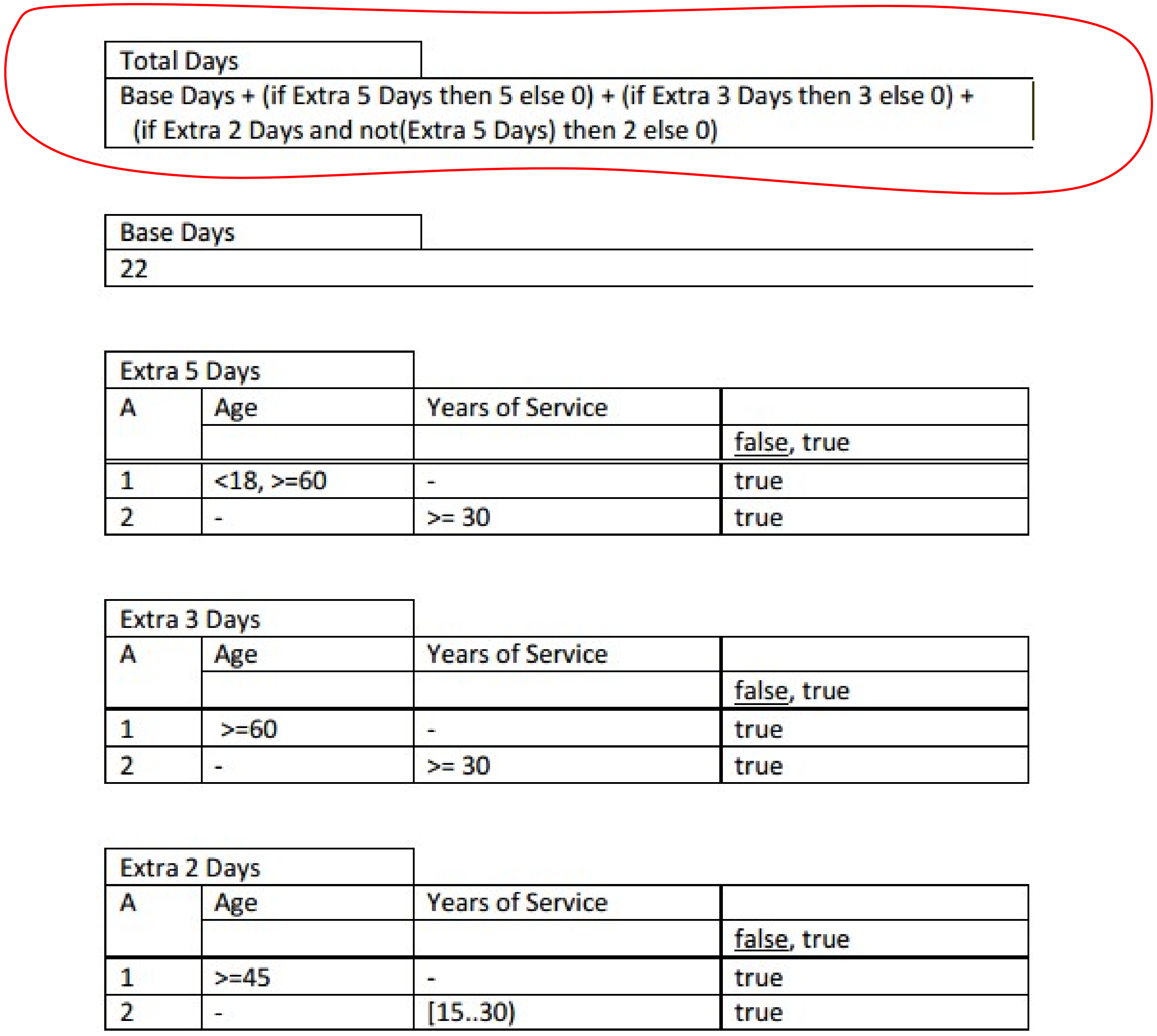
Every employee receives at least 22 days.
Additional days are provided according to the following criteria:

1) Only employees younger than 18 or at least 60 years, or employees with at least 30 years of service will receive 5 extra days.

2) Employees with at least 30 years of service and also employees of age 60 or more, receive 3 extra days, on top of possible additional days already given.

3) If an employee has at least 15 but less than 30 years of service, 2 extra days are given. These 2 days are also provided for employees of age 45 or more. These 2 extra days can not be combined with the 5 extra days.

Solution with FEEL Formula



Total Days
Base Days + (if Extra 5 Days then 5 else 0) + (if Extra 3 Days then 3 else 0) + (if Extra 2 Days and not(Extra 5 Days) then 2 else 0)

Base Days
22

Extra 5 Days

A	Age	Years of Service	
			false, true
1	<18, >=60	-	true
2	-	>= 30	true

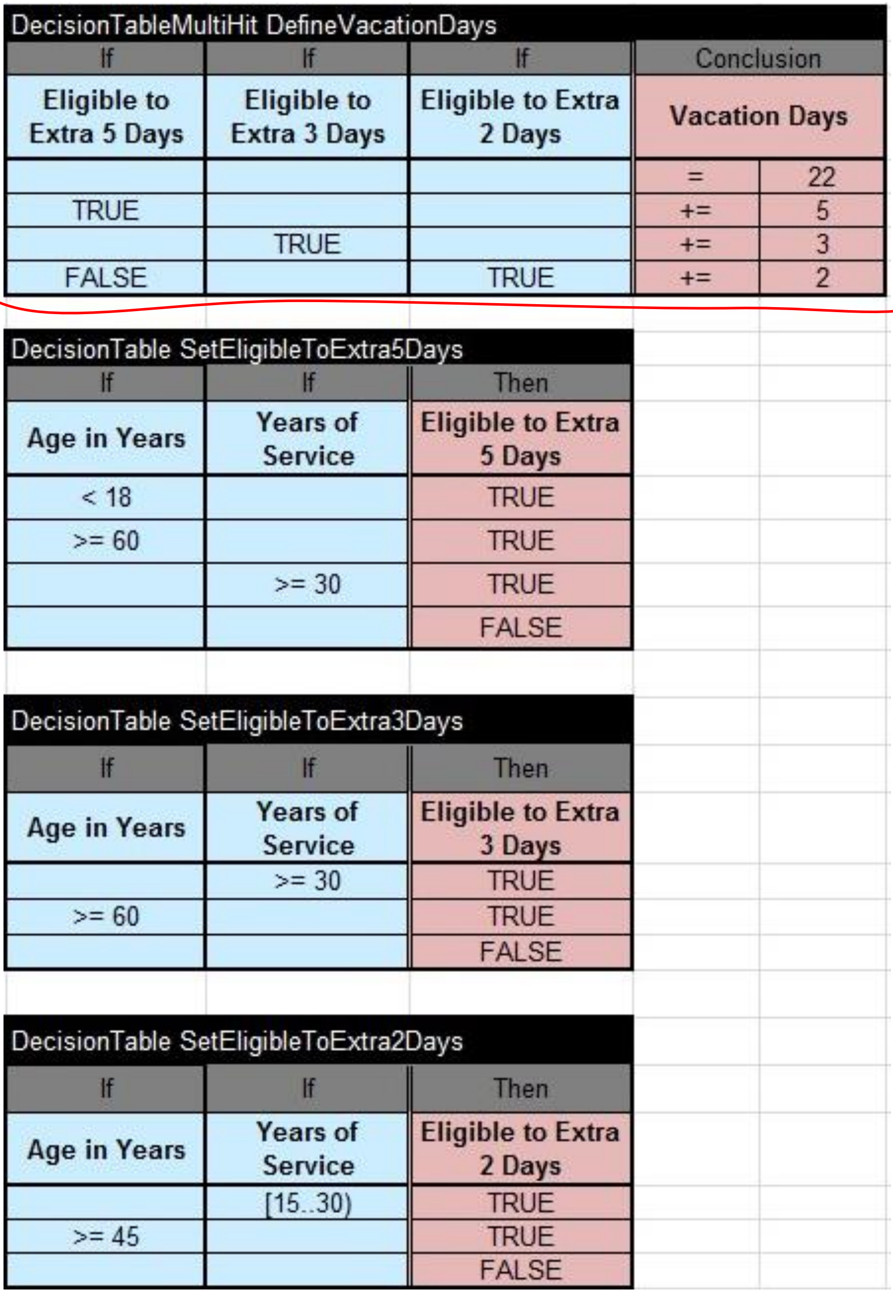
Extra 3 Days

A	Age	Years of Service	
			false, true
1	>=60	-	true
2	-	>= 30	true

Extra 2 Days

A	Age	Years of Service	
			false, true
1	>=45	-	true
2	-	[15..30)	true

Solution without FEEL Formula



DecisionTableMultiHit DefineVacationDays

If	If	If	Conclusion	
Eligible to Extra 5 Days	Eligible to Extra 3 Days	Eligible to Extra 2 Days	Vacation Days	
			=	22
TRUE			+=	5
	TRUE		+=	3
FALSE		TRUE	+=	2

DecisionTable SetEligibleToExtra5Days

If	If	Then
Age in Years	Years of Service	Eligible to Extra 5 Days
< 18		TRUE
>= 60		TRUE
	>= 30	TRUE
		FALSE

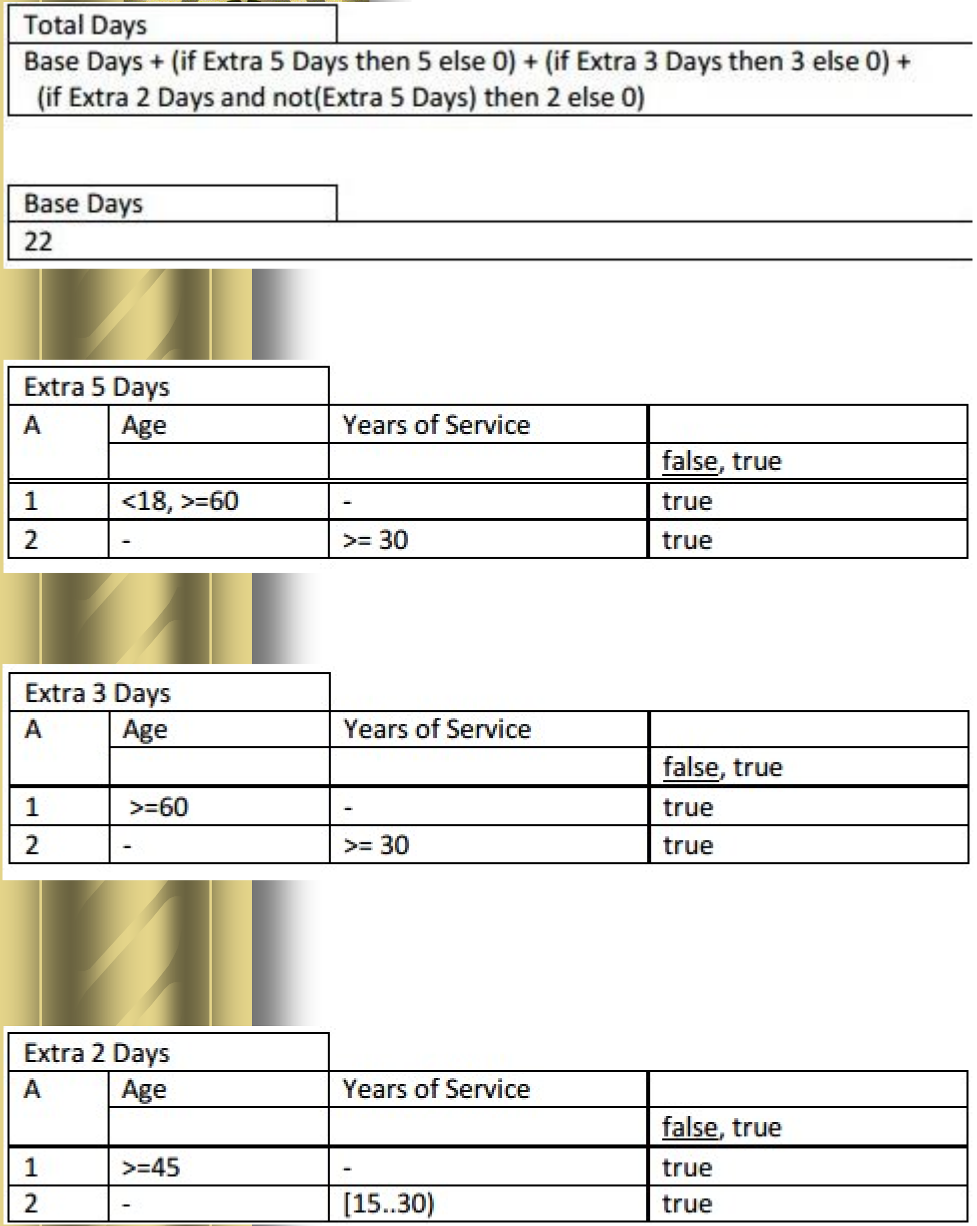
DecisionTable SetEligibleToExtra3Days

If	If	Then
Age in Years	Years of Service	Eligible to Extra 3 Days
	>= 30	TRUE
>= 60		TRUE
		FALSE

DecisionTable SetEligibleToExtra2Days

If	If	Then
Age in Years	Years of Service	Eligible to Extra 2 Days
	[15..30)	TRUE
>= 45		TRUE
		FALSE

Compare Solutions



Total Days
Base Days + (if Extra 5 Days then 5 else 0) + (if Extra 3 Days then 3 else 0) + (if Extra 2 Days and not(Extra 5 Days) then 2 else 0)

Base Days
22

Extra 5 Days

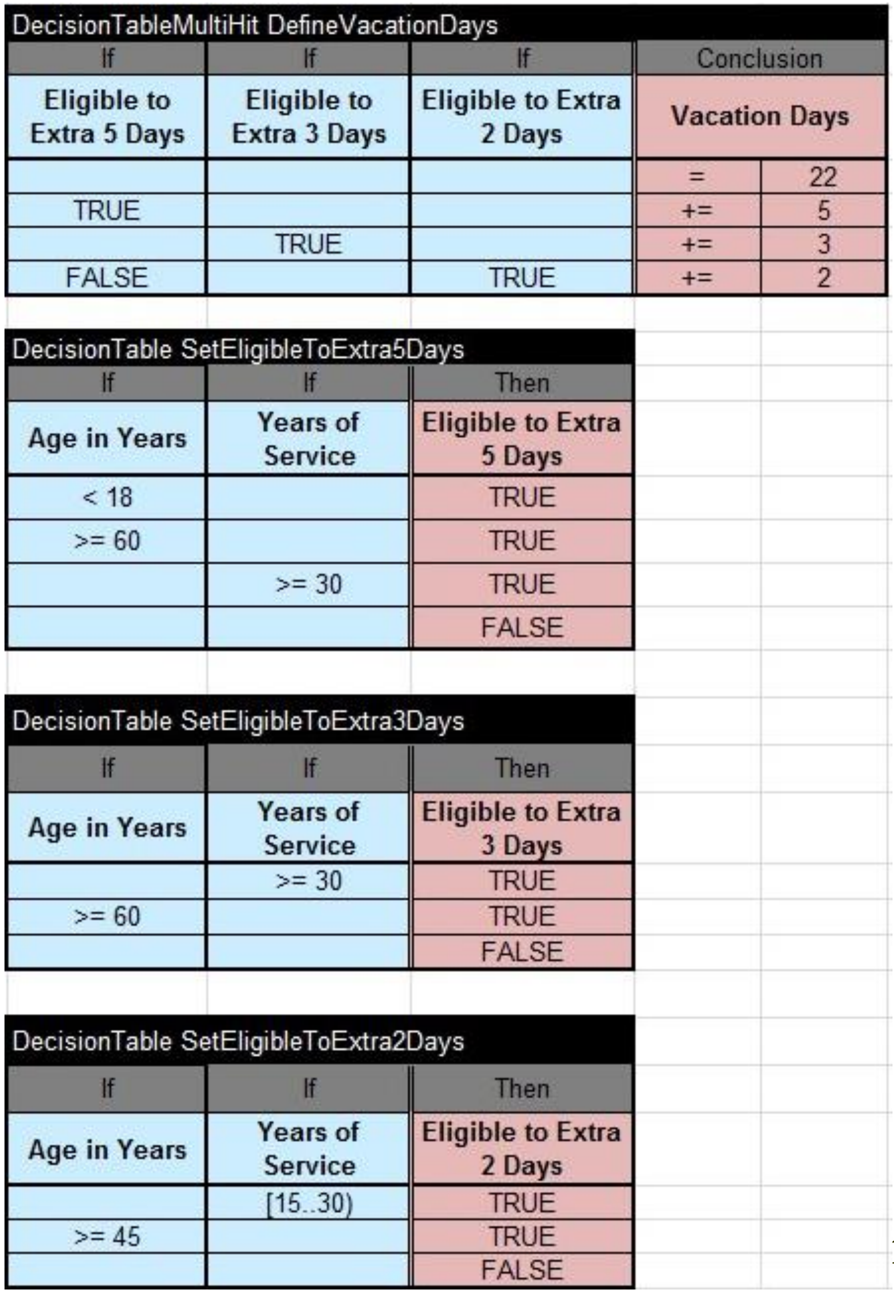
A	Age	Years of Service	
			false, true
1	<18, >=60	-	true
2	-	>= 30	true

Extra 3 Days

A	Age	Years of Service	
			false, true
1	>=60	-	true
2	-	>= 30	true

Extra 2 Days

A	Age	Years of Service	
			false, true
1	>=45	-	true
2	-	[15..30)	true



DecisionTableMultiHit DefineVacationDays

If	If	If	Conclusion	
Eligible to Extra 5 Days	Eligible to Extra 3 Days	Eligible to Extra 2 Days	Vacation Days	
			=	22
TRUE			+=	5
	TRUE		+=	3
FALSE		TRUE	+=	2

DecisionTable SetEligibleToExtra5Days

If	If	Then
Age in Years	Years of Service	Eligible to Extra 5 Days
< 18		TRUE
>= 60		TRUE
	>= 30	TRUE
		FALSE

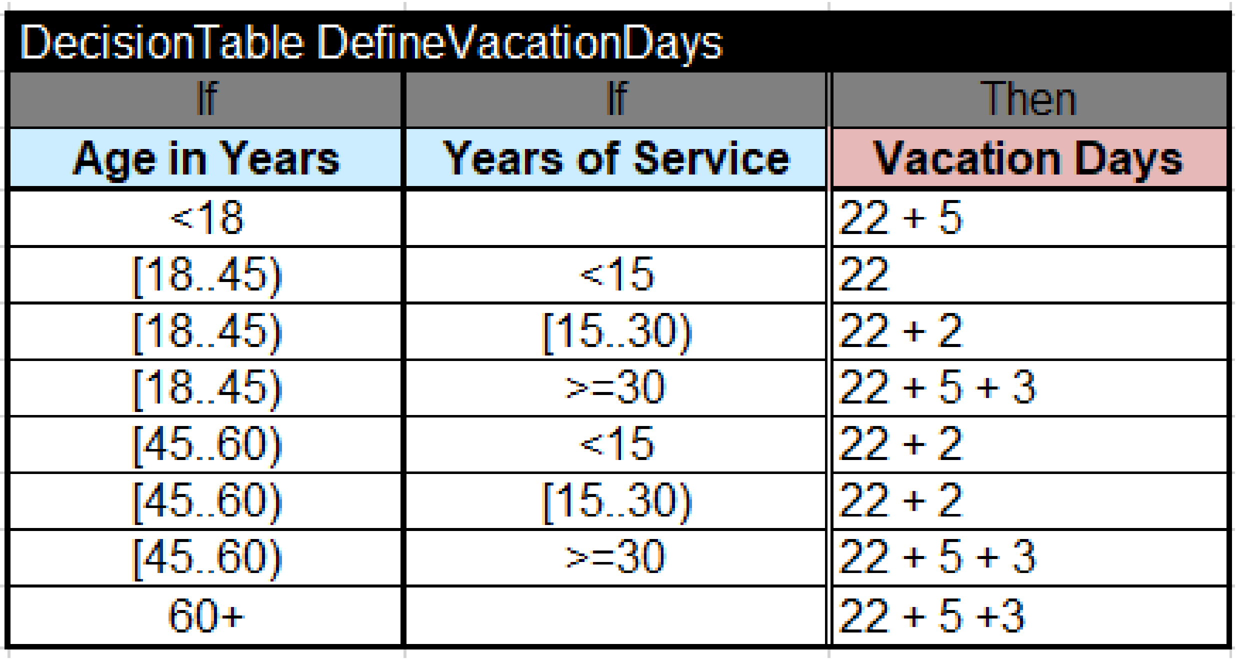
DecisionTable SetEligibleToExtra3Days

If	If	Then
Age in Years	Years of Service	Eligible to Extra 3 Days
	>= 30	TRUE
>= 60		TRUE
		FALSE

DecisionTable SetEligibleToExtra2Days

If	If	Then
Age in Years	Years of Service	Eligible to Extra 2 Days
	[15..30)	TRUE
>= 45		TRUE
		FALSE

Alternative DMN DecisionTable



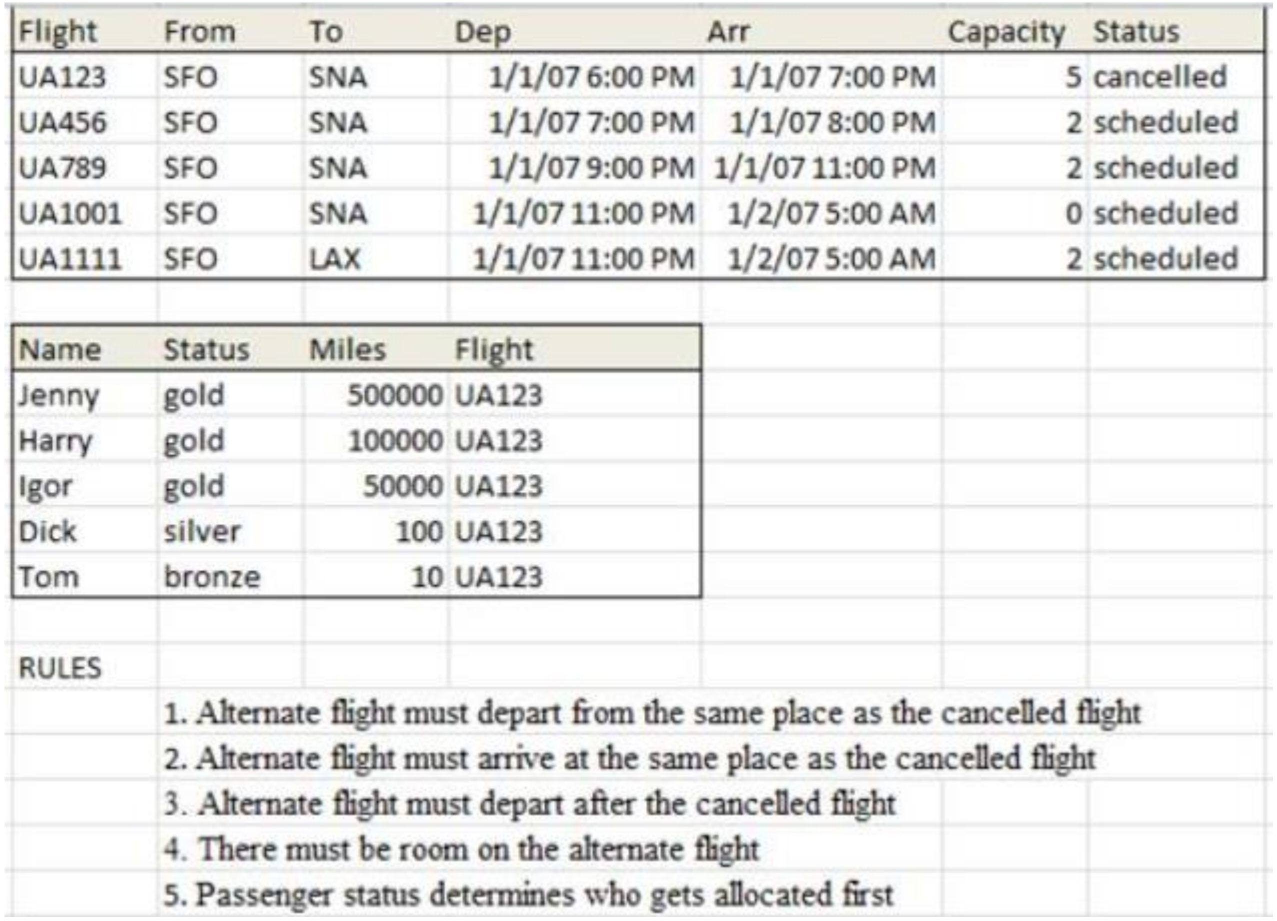
DecisionTable DefineVacationDays		
If	If	Then
Age in Years	Years of Service	Vacation Days
<18		22 + 5
[18..45)	<15	22
[18..45)	[15..30)	22 + 2
[18..45)	>=30	22 + 5 + 3
[45..60)	<15	22 + 2
[45..60)	[15..30)	22 + 2
[45..60)	>=30	22 + 5 + 3
60+		22 + 5 +3

It may look compact but:

- It's hard to recognize the plain English logic
- Difficult to change or add more rules

Decision Model "Rebooking Passengers from Cancelled Flights "

- DMCommunity.org Challenge Oct-2016:



Flight	From	To	Dep	Arr	Capacity	Status
UA123	SFO	SNA	1/1/07 6:00 PM	1/1/07 7:00 PM	5	cancelled
UA456	SFO	SNA	1/1/07 7:00 PM	1/1/07 8:00 PM	2	scheduled
UA789	SFO	SNA	1/1/07 9:00 PM	1/1/07 11:00 PM	2	scheduled
UA1001	SFO	SNA	1/1/07 11:00 PM	1/2/07 5:00 AM	0	scheduled
UA1111	SFO	LAX	1/1/07 11:00 PM	1/2/07 5:00 AM	2	scheduled

Name	Status	Miles	Flight
Jenny	gold	500000	UA123
Harry	gold	100000	UA123
Igor	gold	50000	UA123
Dick	silver	100	UA123
Tom	bronze	10	UA123

RULES

1. Alternate flight must depart from the same place as the cancelled flight
2. Alternate flight must arrive at the same place as the cancelled flight
3. Alternate flight must depart after the cancelled flight
4. There must be room on the alternate flight
5. Passenger status determines who gets allocated first

Plain English Solution

1. Sort all passengers using their GOLD, SILVER or BRONZE status. If two passengers have the same status use miles as a tiebreaker
2. Choose the first unassigned passenger from the sorted list and try to find a suitable flight for this passenger:
   – A “suitable” flight should have the same departure and arrival airports as the cancelled flight and it also should still have an available seat
   – If there are two suitable flights, choose the one with an earlier departure time
3. Do the same for the second passenger from the sorted list, then for the third passenger, etc.

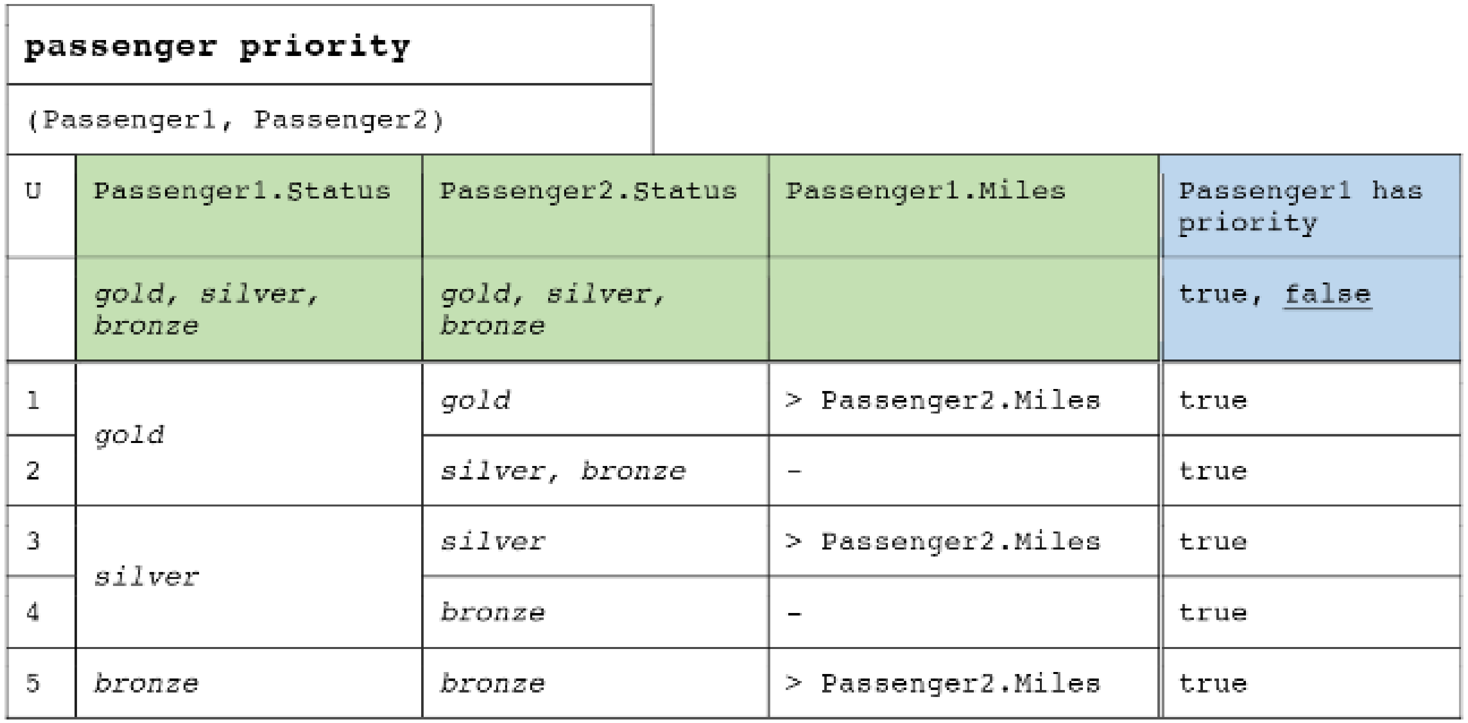
What our decision model needs to do:

- Sort lists of passengers and flights
- Use tiebreakers
- Iterate through passenger and flight lists while controlling seat availability

- No wonder this model was used by DMN experts to demonstrate the most complex DMN constructs of the Compliance Level 3

It is easy to Compare two Passengers:

Using Drools Decision Table:

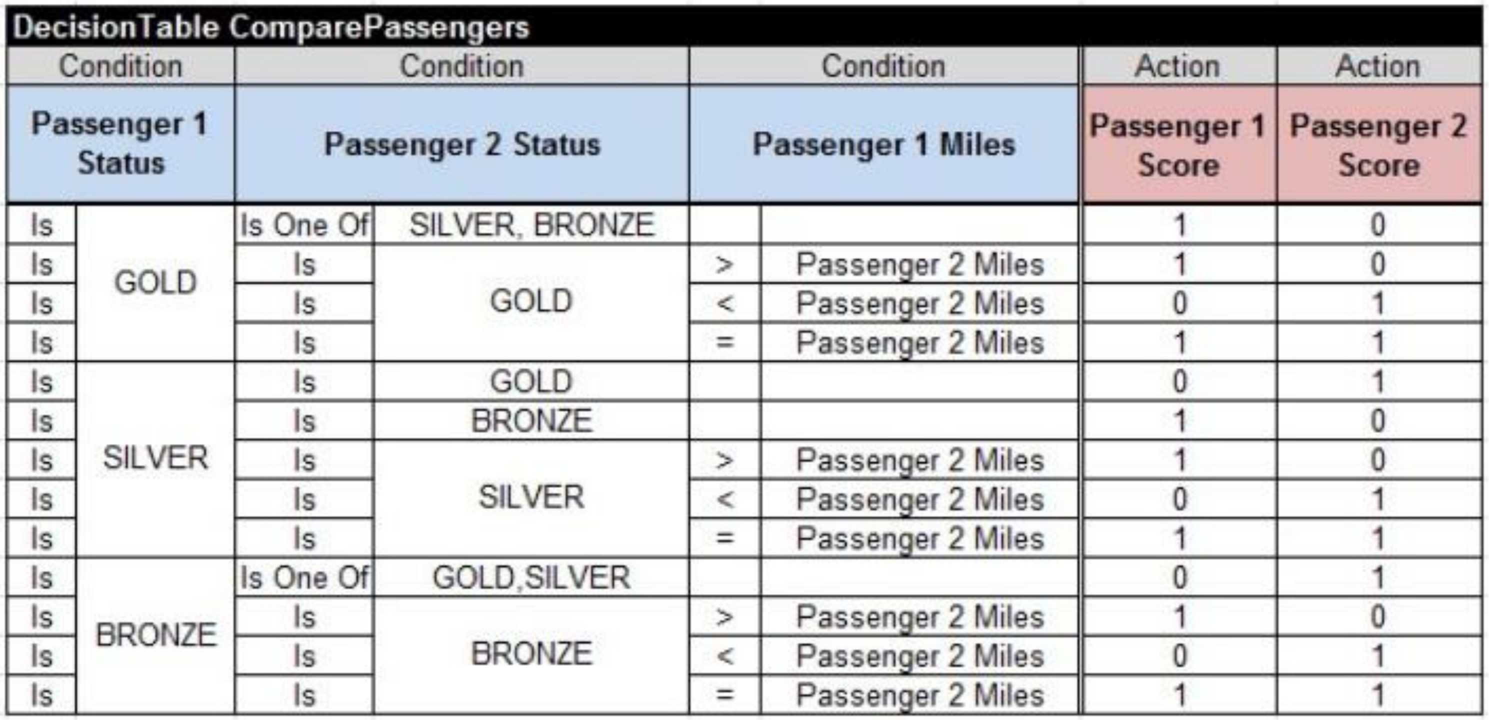


passenger priority

(Passenger1, Passenger2)

U	Passenger1.Status	Passenger2.Status	Passenger1.Miles	Passenger1 has priority
	gold, silver, bronze	*gold, silver, bronze*		true, false
1	*gold*	*gold*	> Passenger2.Miles	true
2		*silver, bronze*	-	true
3	*silver*	*silver*	> Passenger2.Miles	true
4		*bronze*	-	true
5	*bronze*	*bronze*	> Passenger2.Miles	true

The same decision table in OpenRules

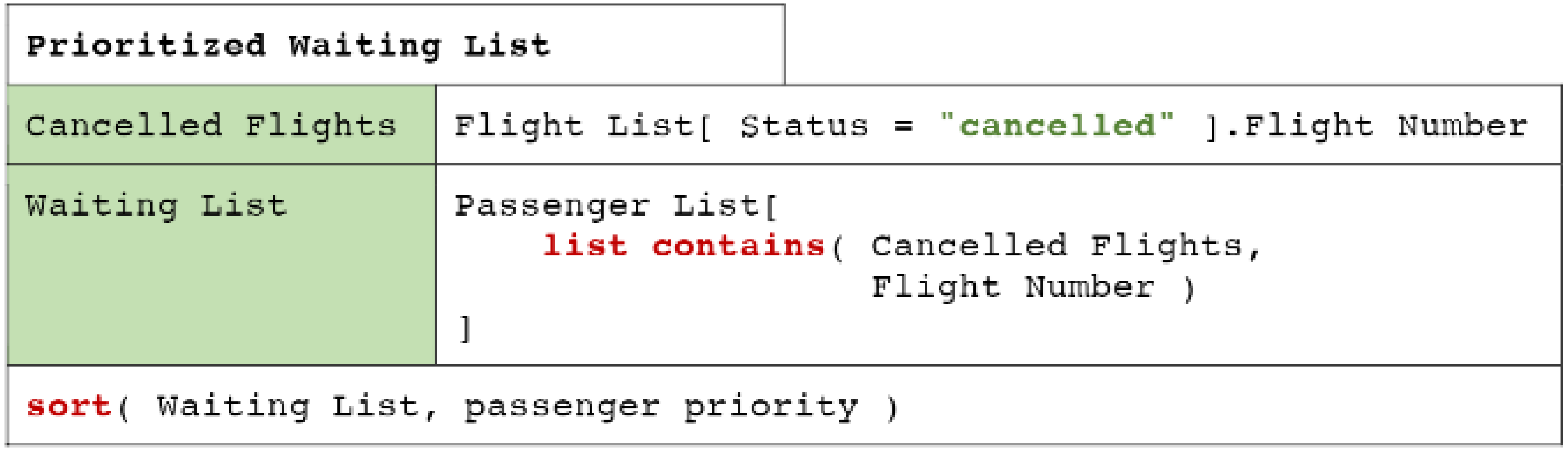


DecisionTable CompareParsengers

Condition		Condition		Condition		Action	Action
Passenger 1 Status		Passenger 2 Status		Passenger 1 Miles		Passenger 1 Score	Passenger 2 Score
Is	GOLD	Is One Of	SILVER, BRONZE			1	0
Is		Is	GOLD	>	Passenger 2 Miles	1	0
Is		Is		<	Passenger 2 Miles	0	1
Is		Is		=	Passenger 2 Miles	1	1
Is	SILVER	Is	GOLD			0	1
Is		Is	BRONZE			1	0
Is		Is	SILVER	>	Passenger 2 Miles	1	0
Is		Is		<	Passenger 2 Miles	0	1
Is		Is		=	Passenger 2 Miles	1	1
Is	BRONZE	Is One Of	GOLD,SILVER			0	1
Is		Is	BRONZE	>	Passenger 2 Miles	1	0
Is		Is		<	Passenger 2 Miles	0	1
Is		Is		=	Passenger 2 Miles	1	1

Sorting Passengers with DMN Box Context

Using Boxed Context and Sort function:



Prioritized Waiting List	
Cancelled Flights	Flight List[Status = "cancelled"].Flight Number
Waiting List	Passenger List[list contains(Cancelled Flights, Flight Number)]
sort(Waiting List, passenger priority)	

Explanations:

1.First box builds a list of Cancelled Flights

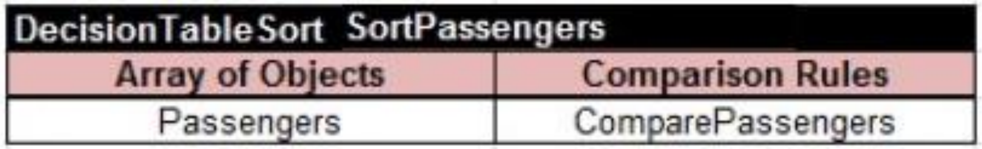
2.Second box defines a list of passengers from these flights

3.Third box call function "sort" with two parameters:

- Waiting List
- Yhe previously defined "passenger priority" to compare passengers

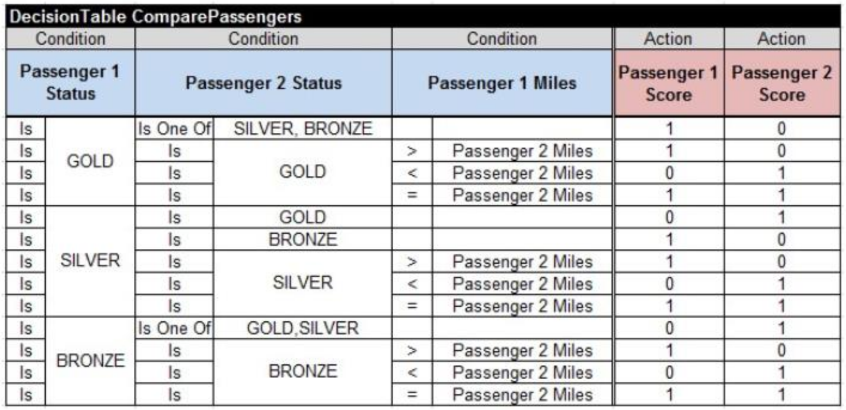
Sorting Passengers without DMN Box Context

Instead we may use the following OpenRules table:



DecisionTableSort SortPassengers	
Array of Objects	Comparison Rules
Passengers	ComparePassengers

This is a special OpenRules table of the type “**DecisionTableSort**” that naturally extends DMN decision tables.
It will sort the array “Passengers” using the previously defined decision table “ComparePassengers”:

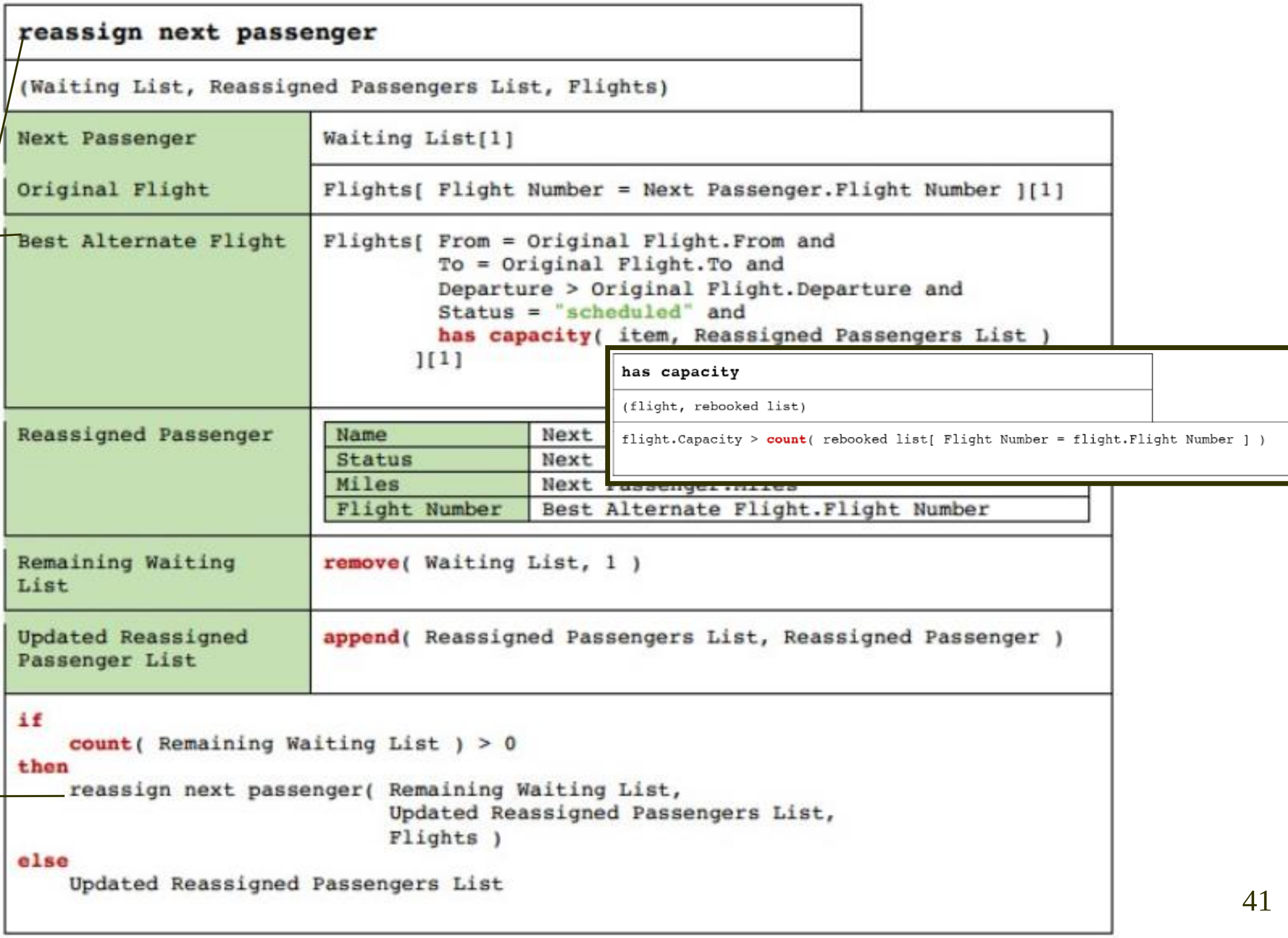


DecisionTable ComparePassengers							
Condition		Condition		Condition		Action	Action
Passenger 1 Status		Passenger 2 Status		Passenger 1 Miles		Passenger 1 Score	Passenger 2 Score
Is	GOLD	Is One Of	SILVER, BRONZE			1	0
Is		Is	GOLD	>	Passenger 2 Miles	1	0
Is		Is		<	Passenger 2 Miles	0	1
Is		Is		=	Passenger 2 Miles	1	1
Is	SILVER	Is	GOLD			0	1
Is		Is	BRONZE			1	0
Is		Is	SILVER	>	Passenger 2 Miles	1	0
Is		Is		<	Passenger 2 Miles	0	1
Is		Is		=	Passenger 2 Miles	1	1
Is	BRONZE	Is One Of	GOLD,SILVER			0	1
Is		Is	BRONZE	>	Passenger 2 Miles	1	0
Is		Is		<	Passenger 2 Miles	0	1
Is		Is		=	Passenger 2 Miles	1	1

DMN Iteration Constructs

- We need to iterate through lists of passenger and flight while controlling seat availability:
- Consider two approaches:
  - Using complex DMN boxed expressions
  - Using decision tables only

Using DMN Boxed Expressions



reassign next passenger

(Waiting List, Reassigned Passengers List, Flights)

Next Passenger	Waiting List[1]
Original Flight	Flights[Flight Number = Next Passenger.Flight Number][1]
Best Alternate Flight	Flights[From = Original Flight.From and To = Original Flight.To and Departure > Original Flight.Departure and Status = "scheduled" and has capacity(item, Reassigned Passengers List)][1]
Reassigned Passenger	Name / Next Status / Next Miles / Next Passenger.Miles Flight Number / Best Alternate Flight.Flight Number
Remaining Waiting List	remove(Waiting List, 1)
Updated Reassigned Passenger List	append(Reassigned Passengers List, Reassigned Passenger)

```
if
    count( Remaining Waiting List ) > 0
then
    reassign next passenger( Remaining Waiting List,
                             Updated Reassigned Passengers List,
                             Flights )
else
    Updated Reassigned Passengers List
```

has capacity

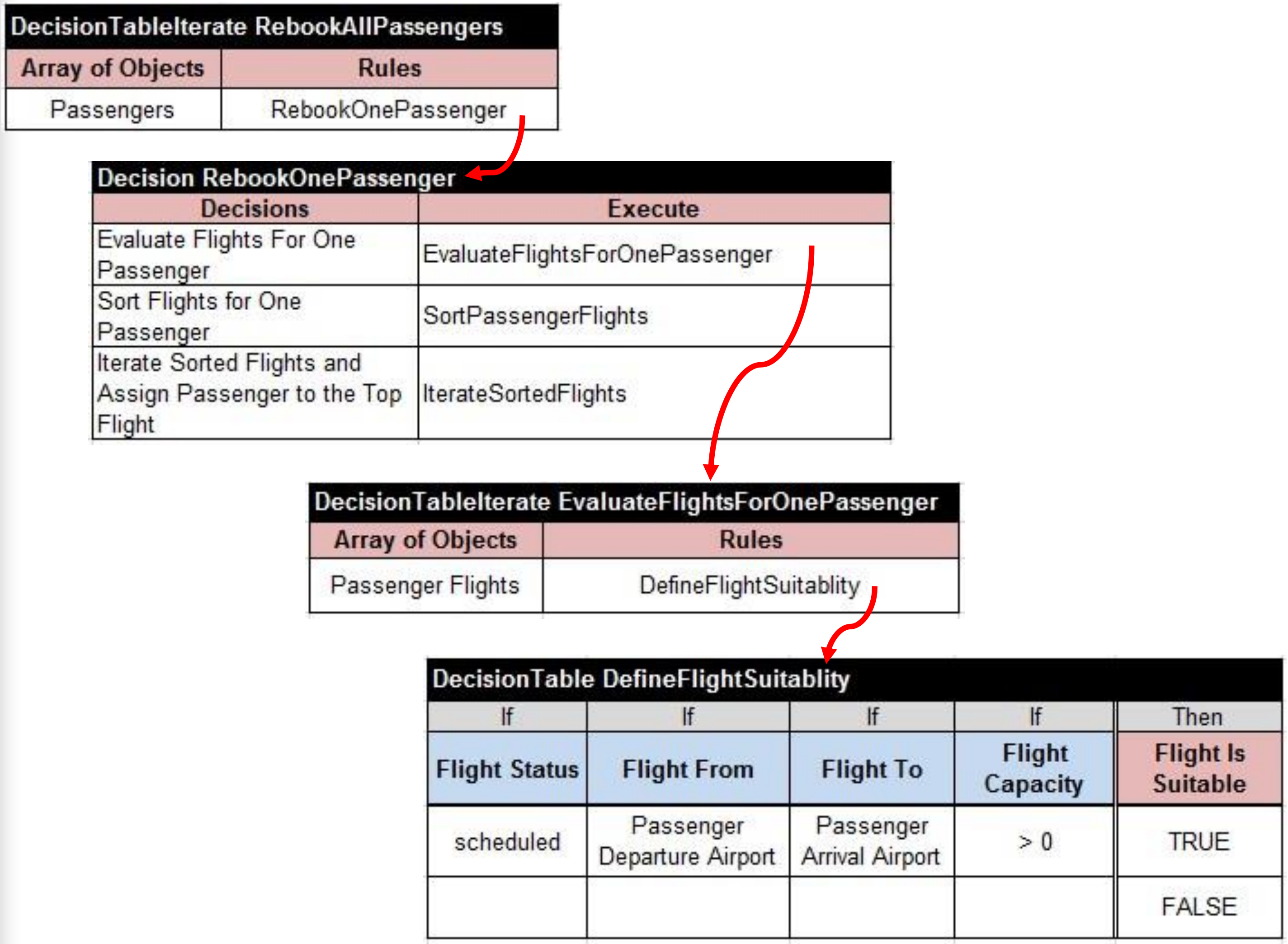
(flight, rebooked list)

```
flight.Capacity > count( rebooked list[ Flight Number = flight.Flight Number ] )
```

Not "Best" as it doesn't look for the earliest arrival

Using a recursive function call

Using Decision Tables Instead of Boxed Expressions



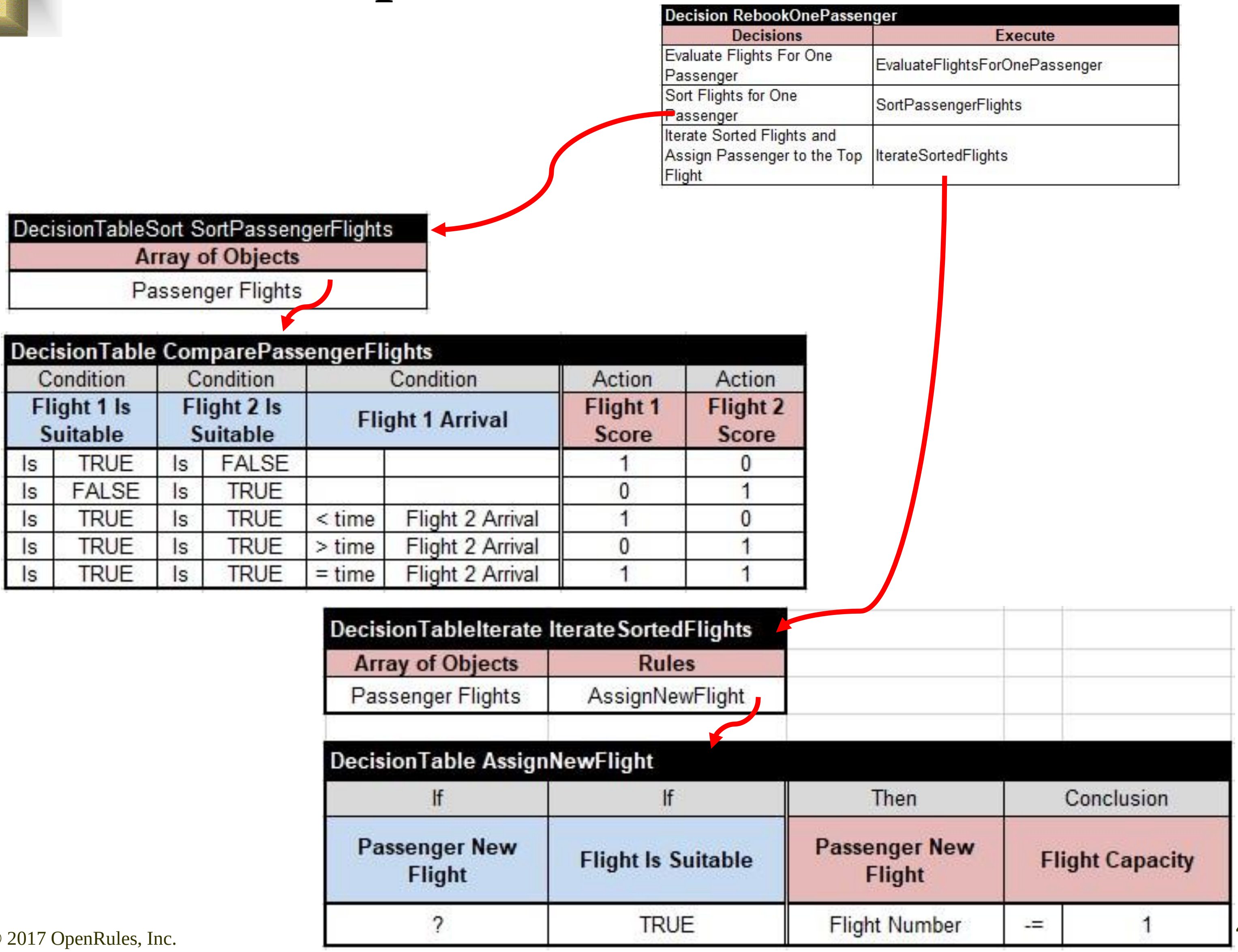
DecisionTableIterate RebookAllPassengers	
Array of Objects	**Rules**
Passengers	RebookOnePassenger

Decision RebookOnePassenger	
Decisions	**Execute**
Evaluate Flights For One Passenger	EvaluateFlightsForOnePassenger
Sort Flights for One Passenger	SortPassengerFlights
Iterate Sorted Flights and Assign Passenger to the Top Flight	IterateSortedFlights

DecisionTableIterate EvaluateFlightsForOnePassenger	
Array of Objects	**Rules**
Passenger Flights	DefineFlightSuitablity

DecisionTable DefineFlightSuitablity				
If	If	If	If	Then
Flight Status	**Flight From**	**Flight To**	**Flight Capacity**	**Flight Is Suitable**
scheduled	Passenger Departure Airport	Passenger Arrival Airport	> 0	TRUE
				FALSE

Using Decision Tables Instead of Boxed Expressions



Decision RebookOnePassenger

Decisions	Execute
Evaluate Flights For One Passenger	EvaluateFlightsForOnePassenger
Sort Flights for One Passenger	SortPassengerFlights
Iterate Sorted Flights and Assign Passenger to the Top Flight	IterateSortedFlights

DecisionTableSort SortPassengerFlights

Array of Objects
Passenger Flights

DecisionTable ComparePassengerFlights

Condition		Condition		Condition		Action	Action
Flight 1 Is Suitable		Flight 2 Is Suitable		Flight 1 Arrival		Flight 1 Score	Flight 2 Score
Is	TRUE	Is	FALSE			1	0
Is	FALSE	Is	TRUE			0	1
Is	TRUE	Is	TRUE	< time	Flight 2 Arrival	1	0
Is	TRUE	Is	TRUE	> time	Flight 2 Arrival	0	1
Is	TRUE	Is	TRUE	= time	Flight 2 Arrival	1	1

DecisionTableIterate IterateSortedFlights

Array of Objects	Rules
Passenger Flights	AssignNewFlight

DecisionTable AssignNewFlight

If	If	Then	Conclusion	
Passenger New Flight	Flight Is Suitable	Passenger New Flight	Flight Capacity	
?	TRUE	Flight Number	-=	1

Enhance Core DMN tables (not programming constructs)

- Today DMN makes emphasis on complex Boxed Expressions that belong to the "Compliance Level **3**" (CL3)
- We demonstrated that even complex decision logic including iterations and sorting can be represented by traditional decision tables that belong to the "Compliance Level **2**" (CL2)
- Hopefully, future DMN releases will add decision tables similar to the discussed ones to avoid programming

More Information

- A detailed comparison of how the iteration and sorting logic is implemented with and without programming can be found at the OpenRules Blog
- LinkedIn Articles:
  - Using Decision Tables to Sort and Iterate Over Arrays of Business Objects
  - Decision Table Properties in DMN and Beyond

Conclusion

- DMN is a serious step toward standardized and interchangeable representations of business decision logic
- Core DMN concepts allow <u>business people</u> (not programmers) to represent, test, and manage their decision models
- Even very complex business logic can be implemented <u>without programming</u>

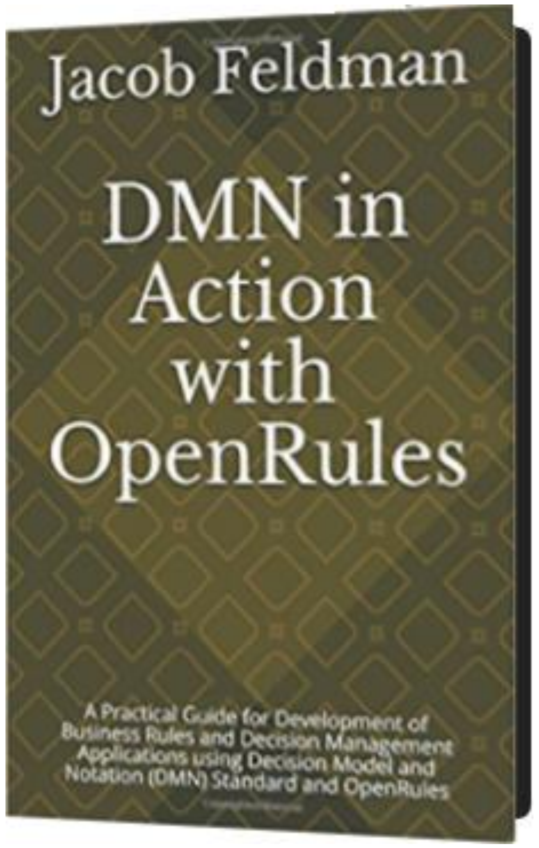
QnA

Jacob Feldman, PhD

OpenRules, Inc.

www.OpenRules.com

jacobfeldman@openrules.com


Jacob Feldman
DMN in Action with OpenRules
A Practical Guide for Development of Business Rules and Decision Management Applications using Decision Model and Notation (DMN) Standard and OpenRules

Available at Amazon.com