Decision Table

A decision table is a precise yet compact way to present complicated logic. It has similar logic to if-else statement and associate conditions to actions in an elegant way.

Decision Tables are typically divided in 4 quadrants.

Conditions	Condition entries
Actions	Corresponding action

Condition entries can be represented as:

- Simple true/false values (akin to if-then-else)
- Numbered alternatives (akin to switch-case)
- Fuzzy logic or probabilistic representations

Action entries can be represented as:

- Simple decision tables: whether an action is to be performed (tick the actions to perform)
- Advanced decision tables: sequencing of actions to perform (number the actions in the order to perform)

For example, consider the following scenario. Each SH1 students at NJC will attain a final mark for their performance across various assessments throughout the year. Suppose a simple grading scheme is applied to these students, such that: a mark of 70 or more corresponds to an A; a mark below 50 corresponds to a Fail; any other mark corresponds to a Pass.

The decision table is set up by applying the following steps:

1. List Conditions and Condition Values

Conditions	Mark ≥ 70	Y	Y	N	N
	Mark < 50	Y	N	Y	N
Actions					

2. List Actions and Action Values

Conditions	Mark ≥ 70	Y	Y	N	N
	Mark < 50	Y	N	Y	N
Actions	A	-	X		
	Pass	-			X
	Fail	-		X	

Note that a Y means true, while N means false. X means this action is performed. A dash is added where the conditions alternatives are irrelevant or impossible.

Decision tables allow us to detect and remove redundancies within the program logic. To describe how this is done, let us look at a more complex example:

- Consider an online company that charges $5 for delivery of packages.
- If the order value is over $50 and the package is small and the customer has a promotion code, the delivery is free.
- If the order is over $50 and the package is small, the delivery charge is $1.
- if the order value is over $50 and the customer has a promotion code, the delivery charge is $1.

The decision table can be created as follows:

Conditions	Order Value > $50	Y	Y	Y	Y	N	N	N	N
	Small Package	Y	Y	N	N	Y	Y	N	N
	Promotion Code	Y	N	Y	N	Y	N	Y	N
Actions	Free Delivery	X							
	$1 Charge		X	X					
	$5 Charge				X	X	X	X	X

To find redundancies, we look at each action and then check whether the conditions are required.

Free delivery only applies if all 3 conditions are true. There are no redundancies here.

The $1 charge only applies if condition 1 if true and either condition 2 or condition 3 is true. Again, there are no redundancies here.

The $5 charge applies in all cases where condition 1 is false. The redundant conditions are thus shown by the shaded cells. We can therefore simplify the table, which is shown below.

Conditions	Order Value > $50	Y	Y	Y	Y	N
	Small Package	Y	Y	N	N	-
	Promotion Code	Y	N	Y	N	-
Actions	Free Delivery	X				
	$1 Charge		X	X		
	$5 Charge				X	X

A more systematic way to arrive at the above reduction is to always try to perform pairwise reductions first. Referring to the example above, arriving that the final, fully reduced decision table is actually the result of 3 pairwise reductions.

Let us look at the first 2 columns that comprise the redundant conditions:

Conditions	Order Value > $50	N	N	N	N
	Small Package	Y	Y	N	N
	Promotion Code	Y	N	Y	N
Actions	Free Delivery				
	$1 Charge				
	$5 Charge	X	X	X	X

We notice that for these 2 columns, only 1 condition varies: promotion code; everything else, including the action, is the same across the 2 columns. This means that promotion code is irrelevant - no matter what value it takes (i.e., Y or N), the outcome or action is the same. We may thus reduce this pair of columns:

Conditions	Order Value > $50	N	N	N
	Small Package	Y	N	N
	Promotion Code	-	Y	N
Actions	Free Delivery			
	$1 Charge			
	$5 Charge	X	X	X

Recall that the dash indicates irrelevance - i.e., we are specifying that we do not care about the promotion code condition for this column/rule.

In a similar fashion, promotion code is also redundant for the last 2 columns. Reducing those columns, we have:

Conditions	Order Value > $50	N	N
	Small Package	Y	N
	Promotion Code	-	-
Actions	Free Delivery		
	$1 Charge		
	$5 Charge	X	X

When we reach this point, we also notice that the 2 reduced rules are exactly the same except for 1 row: the small packages condition. We may thus also reduce these 2 columns:

Conditions	Order Value > $50	N
	Small Package	-
	Promotion Code	-
Actions	Free Delivery	
	$1 Charge	
	$5 Charge	X

In summary, to remove redundancies in a decision table, we perform the following:

- Detect pairs of rules which only differ in terms of the values of 1 condition (do note that the action must be the same) and merge them by replacing the redundant condition values with a "-"
- Repeat this process until such pairs no longer exist in the table
- Do note that when rules can be reduced in more than 1 way, you should test all permutations and utilise the one that result in the greatest reduction of redundancy

There are various benefits to using decision tables:

- It allows us to more easily audit the control logic

Decision Tables	**Traditional Control Structures**
● Make it easy to observe that all possible conditions of the system are accounted for ● Every possible combination of the provided conditions is given, with their corresponding action ● When conditions are omitted, it is obvious even at a glance that logic is missing	● Gaps in program logic are not easy to notice at a glance ● Sometimes, it is difficult to follow which conditions correspond to which actions

- Demands a programmer to think of all the possible combinations
- With traditional control structure, it is easy to forget about corner cases, especially when the else statement is optional
- Since logic is so important to programming, decision tables are an excellent tool for designing control logic

Exercise 1:

Customers input the amount of money they want to withdraw from their bank account into an Automatic Teller Machine (ATM). The ATM will dispense the money only if the amount is no more than the account balance and no more than the withdrawal limit on the account. The ATM will also check the amount of money held in the ATM, and offer the amount available if it is less than the amount requested. Otherwise the transaction will be cancelled.

(a) Create a decision table to show these conditions and actions.

(b) Simplify your decision table by removing redundancies.

Exercise 2:

A fashion shop gives customers a discount on purchases totalling more than $30:

- Discount of 5% with a member card
- Discount of 5% for purchases totalling more than $200
- Discount of 10% with a member card and for purchases totalling more than $200

(a) Create a decision table to show all the possible conditions and actions.

(b) Simplify your decision table by removing redundancies.

Exercise 3:

Below is the manner in which the school library will process its overdue list:

- If a book is overdue then a reminder letter would normally be sent.
- However, if the book is more than 5 days overdue, 2 further checks are made to see whether the reminder should be replaced by a warning letter:
    - If the student has had a previous warning letter the student will not only receive the warning letter but, in addition, a copy will be sent to the parents of the student.

    - If the student has more than 4 books overdue, but no previous warning letter, the reminder letter is replaced by a warning letter.

(a) Create a decision table showing all the possible outcomes and results.

(b) Simplify your decision table by removing redundancies.