**Decision Table (таблица решений)** — техника, помогающая наглядно изобразить комбинаторику условий из ТЗ. В тестировании таблица решений используется для того, чтобы на основе требований составить тест-кейсы. И ничего не забыть при сложных комбинациях входных условий! Ведь каждая строка или столбец таблицы → готовый тест-кейс.

Используется если нужно протестировать логику, где нельзя нажимать на кнопки Например:

* условие по которым будет проверяться является ли треугольник равнобедренным, равносторонним или неравносторонний
* Расчет скидки по сумме выкупа и кол-ва товаров

**Алгоритм работы по этой технике**:  
  
**1.** Определить все условия и выписать их. Треугольник может иметь все разные стороны, 2 стороны равны, 3 стороны равны. Треугольник существует если 1 сторона + 2 сторона >= 3 стороне

**2.** определить эквивалентные классы условий, их граничные значения и выписать это. Треугольник существует если a+b>=c. Граничные значения будут a+b<c ; a+b=c ; a+b>c.

**3.** Прописать в Decision Table все условия + Expected Result.

**4.** Прописать все комбинации условий – Rules (например если у нас 2 условия и каждое условие имеет 4 значения, то проверяем каждое значение первого условие с каждым значением второго условия)

Количество комбинаций = кол-во комбинаций 1 условия \* кол-во комбинаций 2 условия \* …  
В случае с треугольником будет Sides (все разные стороны, 2 равны, 3 равны) \* Triangle Existence (a+b>c ; a+b=c ; a+b<c) 🡪 3 \* 3 = 9 проверок  
  
**5**. Прописать в Expected Result ожидаемый результат по каждой проверке

**6.** Провести тестирование

**Как составлять таблицу**

* По горизонтали — выписываем условия, которые влияют на результат. А чуть ниже — сам результат
* По вертикали — правила: конкретная комбинация входных условий.

То есть мы указываем значения условий и результата

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Condition** | Rule 1 | Rule 2 | … | Rule N |
| Условие 1 |  |  |  |  |
| Условие 2 |  |  |  |  |
| … |  |  |  |  |
| Условие N |  |  |  |  |
| **Expected Result** |  |  |  |  |
| Result 1 |  |  |  |  |
| Result 2 |  |  |  |  |
| … |  |  |  |  |
| Result N |  |  |  |  |

При заполнении таблицы будем следовать следующим правилам.

Первая строка таблицы формируется так: возьмём число всех комбинаций (Rules) и разделим его на число вариантов первого условия. Например у нас есть 16 комбинаций. Делим на 2 условия («Да» и «Нет»). Получается 8. 8 значений «Да» и 8 значений «Нет» будет в первой строке. У нас появились 2 подмножества вариантов – подмножество вариантов «Нет» и подмножество вариантов «Да».

Вторая строка формируется следующим образом: количество значений «Да» первого условия поделим на количество вариантов второго условия («Да» и «Нет»), получим 4. 4 значения «Да» и 4 значения «Нет» будет у нас в каждом подмножестве из строки 1

Остальные строки формируются аналогично.

**Пример 1. Страховка на автомобиль (один результат)**

Я прихожу в страховую компанию и заполняю анкету, где есть 2 вопроса:

1. Есть ли 5 лет стажа вождения?
2. Была ли в авариях?

Ответить можно либо да, либо нет.

Получается 2 условия по 2 возможных варианта, итого 4 варианта пересечения условий, 4 правила. На каждое правило свой результат:

* Если у меня небольшой стаж и я часто бываю в авариях — придется заплатить по максимуму, иначе страховать такого водителя будет невыгодно.
* Если нет стажа, но нет аварий — плачу поменьше, но не сильно. Знаете как бывает — первое время катаются очень осторожно, а потом начинают думать «да я царь и бог, не попаду в аварию». И понеслось...
* Если я опытный водитель, но бываю в авариях — ценник еще чуть ниже. Ведь бывать в авариях — это нормально. Иногда ты просто стоишь на светофоре, а в тебя влетает дурак, ну что тут поделаешь? Но если аварий мало, а опыта много — это хороший знак.
* Если опытный, да еще и без аварий — меньше всего. Очень аккуратный водитель, платить скорее всего не придется!

А теперь то же самое, только в виде таблички:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Правило 1** | **Правило 2** | **Правило 3** | **Правило 4** |
| **Условия** |  |  |  |  |
| Стаж 5 лет | Нет | Нет | Да | Да |
| Был в авариях? | Да | Нет | Да | Нет |
|  |  |  |  |  |
| **Результат** |  |  |  |  |
| Страховка | 200 руб | 100 руб | 50 руб | 10 руб |

**Пример 2. Интернет-магазин (множественный результат)**

Есть интернет-магазин, который предлагает:

* Скидку постоянного покупателя
* Количество вещей, которые курьер привезет бесплатно

Это такие плюшки за лояльность и повторную покупку. Как плюшки высчитываются? Есть два условия:

* Сколько клиент потратил (всего или за какой-то период времени) — бонус добавляется при достижении 100р, 500, 1000 и 5000
* Какой процент выкупа (а то, может, курьер зря мотается) — бонус получаем при достижении 5%, 30%, 50% и 80%

Если я потратила 100р и почти ничего не выкупила — скидки мне не дадут и вещей мало привезут. Если потратила больше и выкупила больше, то дадут небольшую скидочку. Ещё потрачу — станут вещей больше привозить... И так далее.

Положим требования в таблицу:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Правило 1** | **Правило 2** | **...** | **Правило N** |
| **Условия** |  |  |  |  |
| Потратил | 100 | 500 | 1000 | 5000 |
| Выкуп | 5% | 30% | 50% | 80% |
|  |  |  |  |  |
| **Плюшка** |  |  |  |  |
| Скидка | 0% | 6% | 10% | 20% |
| Кол-во вещей | 2 | 8 | 15 | 20 |

Заметьте, что условия 2, и в каждом возможны 4 варианта — это 16 возможных пересечений, 16 тестов!

Попробуем записать все условия:

16 / 4 = 4. У нас будет 4 блока первого условия

4 / 4 = 1. Второй блок условий будет повторяться единажды

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Condition** | Rule 1 | Rule 2 | Rule 3 | Rule 4 | Rule 5 | Rule 6 | Rule 7 | Rule 8 | Rule 9 | Rule 10 | Rule 11 | Rule 12 | Rule 13 | Rule 14 | Rule 15 | Rule 16 |
| Потратил | 100 | 100 | 100 | 100 | 500 | 500 | 500 | 500 | 1000 | 1000 | 1000 | 1000 | 5000 | 5000 | 5000 | 5000 |
| Выкуп % | 5 | 30 | 50 | 80 | 5 | 30 | 50 | 80 | 5 | 30 | 50 | 80 | 5 | 30 | 50 | 80 |
| **Expected Result** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Скидка % | 0 | 5 | 5 | 7 | 5 | 6 | 7 | 8 | 8 | 8 | 10 | 15 | 11 | 13 | 20 | 20 |
| Кол-во Вещей | 2 | 4 | 4 | 6 | 8 | 8 | 9 | 9 | 6 | 10 | 15 | 15 | 15 | 15 | 15 | 20 |

**Советы для упрощения таблиц**

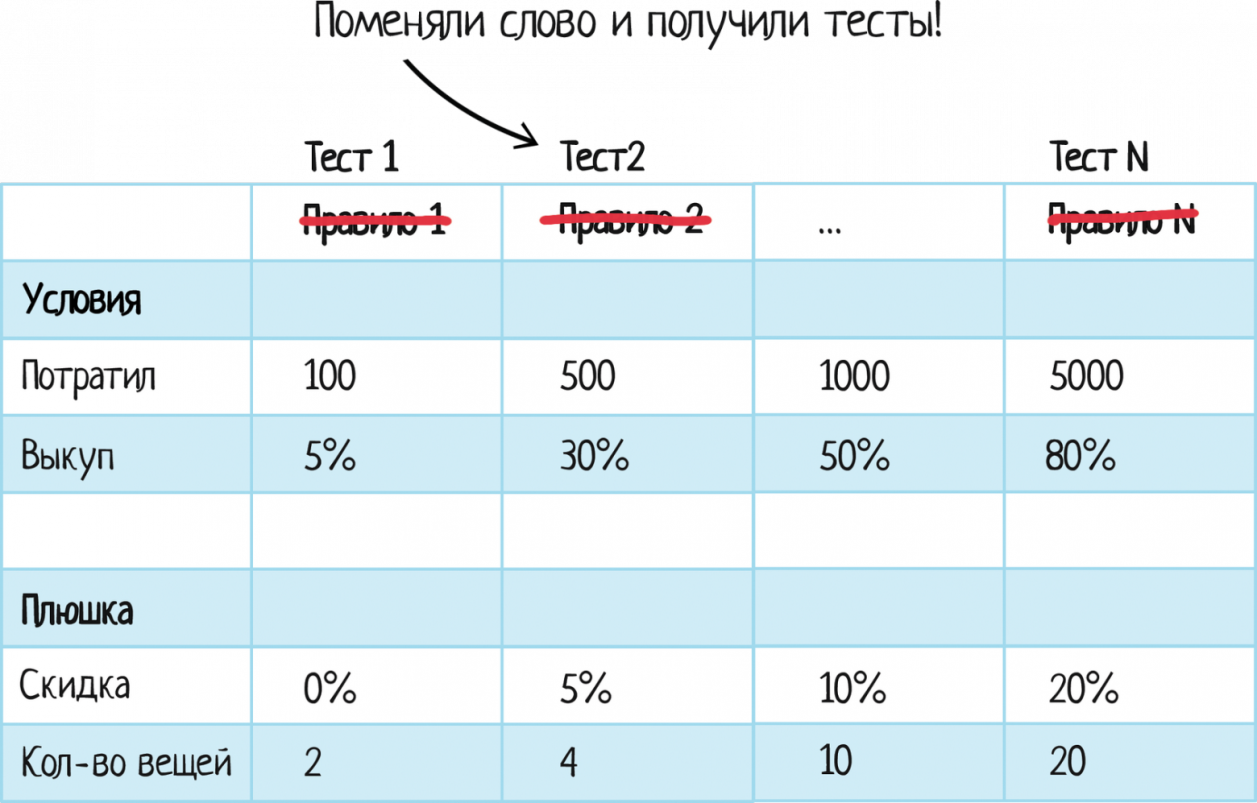
При построении и анализе таблицы принятия решений также может возникнуть ряд проблем. Вот несколько советов, как можно упростить задачу.

* 1. Объединить несколько близких условий в одно, если это возможно по смыслу.
  2. Если условий в итоге получилось много, можно разбить их на группы, например на условия, которые выполняются до какого-то события, и на условия, которые должны выполняться после этого события. Для каждой группы условий нужно построить свою таблицу решений.
  3. Родственные комбинации следует располагать рядом, т. к. они могут приводить к одним и тем же результатам. Так легче анализировать и упрощать таблицу.
  4. Наиболее доминантные условия следует располагать вверху таблицы, потому что так проще сокращать таблицу. Например, при одном событии (скажем, при нажатии определённой кнопки) выполняется целый ряд условий. При другом событии, альтернативном первому, ничего не происходит, т. е. условия, актуальные для первого события, совсем не актуальны для второго, и комбинации второго события с этими условиями можно сразу вычеркнуть.
  5. Условия с большим количеством вариантов располагать внизу таблицы, т. к. это также упростит процесс сокращения таблицы

**Плюсы подхода**

1. **Наглядность**— таблица нагляднее текста. Можно взять таблицу и подойти к аналитику с каким-то вопросом. Или к разработчику. Им будет проще понять, о чём речь, чем если вы принесете стену текста.

2. **Нарисовал таблицу = записал тест-кейсы.** Поменял в заголовках слово «правило» на «тест-кейс», и вот они, готовые тесты! И это будут основные позитивные тесты, которые мы проводим в первую очередь.



Если неудобно, что тест приходится читать сверху вниз, а не слева направо, таблицу можно инвертировать — перевернуть:

|  |  |  |  |
| --- | --- | --- | --- |
| **Тест-кейсы** | **Условие 1:**  **Потратил** | **Условие 2:**  **Выкуп** | **Результат** |
| Кейс 1 | 100 | 5% | Do X / Do A |
| Кейс 2 | 500 | 30% | Do X / Do Y |
| Кейс 3 | 1000 | 50% | Do B / Do C |
| Кейс 4 | 5000 | 80% | Do B / Do Z |

3. **Наглядность поможет найти баги в документации**. Так как косяк формулы будет сразу бросаться в глаза.

4. **Таблица помогает взглянуть на ТЗ свежим взглядом**, по-новому. Пока мы пытаемся перекомбинировать условия, составляя таблицу, мы можем заметить пропущенный ранее баг.

**Минусы подхода**

Особых минусов нет, но таблица не нужна, если:

* Слишком простое условие — потому что «но зачем?». И так все понятно.
* Слишком много входных данных — овер дофига будет колонок. Много тестов, но мало результата, потому что тут уже нужен тест-анализ, pairwise и т.д.

Итого

Decision Table используется для описания сложных системных правил:

* Условия представляют собой входные данные.
* Действия – это ожидаемый результат.
* Колонки – тест-кейсы!

Я настоятельно рекомендую применять эту технику хотя бы однократно — к тем требованиям, которые вы уже знаете наизусть. Думаете, проверили все возможные комбинации? Нарисуйте таблицу решений и результат вас удивит!