

# pavanpw-19

webtable and pagination-

first capture the entire table -

```
const table=await page.locator('#productTable')
```

get total number of columns -

```
const columns= await table.locator('thead tr th')
```

get total number of rows-

```
const rows=await table.locator('tbody tr')
```

this is how we use filter to match a specific row-

```
//2) select check box for product 4 /* const machedRow= rows.filter({ has: page.locator('td'),  
hasText: 'Product 4' }) await machedRow.locator('input').check() */
```

**filter has multiple options.**

**we are looking for page locators which have td tag.**

**and we are looking for product4.**

**storing in variable.**

**then use another locator for checkbox next to product4 and check it.**

to capture a particular row-

```
const row=rows.nth(i);
```

example it will capture row one in first iteration then row two and so on.

to capture row wise data-

```
const tds=row.locator('td')
```

use this columns in the second for loop to read them.

capture column wise data-

**await tds.nth(j).textContent()**

pagination-

first get number of pages.

**const pages=await page.locator('.pagination li a')**

```
1 //
2 const {test, expect}=require('@playwright/test')
3
4 test("handling table",async ({page})⇒{
5
6     await page.goto('https://testautomationpractice.blogspot.com/');
7
8     const table=await page.locator('#productTable')
9
10    // 1) total number of rows & columns
11    const columns= await table.locator('thead tr th')
12    console.log('Number of columns:', await columns.count()) //4
13    expect(await columns.count()).toBe(4)
14
15    //get total number of rows
16    const rows=await table.locator('tbody tr')
17    console.log('Number of rows:', await rows.count()) //5
18    expect(await rows.count()).toBe(5)
```

```
1 //2) select check box for product 4
2 //filter has multiple options.
3 //we are looking for page locators which have td tag.
4 // and we are looking for product4.
5 // storing in variable.
6 // then use another locator for checkbox next to product4 and check it.
7 /* const machedRow= rows.filter({
8     has: page.locator('td'),
9     hasText: 'Product 4'
10 })
11 await machedRow.locator('input').check()
12
13 await page.waitForTimeout(5000);
14 */
```

```
1 //3) select multiple products by re-usable function
2 // await selectProduct(rows,page,'Product 1')
3 // await selectProduct(rows,page,'Product 3')
4 // await selectProduct(rows,page,'Product 5')
5
6 //await page.waitForTimeout(5000);
7
8 //4) print all product details using loop
9 /* for(let i=0;i<await rows.count();i++)
10 {
11 //to capture a particular row
12 //example it will capture row one in first iteration then row two and so on
13     const row=rows.nth(i);
14     //to capture row wise data-
15     const tds=row.locator('td')
16
17     // use this columns in the second for loop to read them.
18 //capture column wise data-
19     for(let j=0 ;j< await tds.count()-1;j++)
20     {
21         console.log(await tds.nth(j).textContent())
22     }
23 }
24 */
```

```
1 //5) read data from all the pages in the table
2
3 //get total number of pages
4 const pages=await page.locator('.pagination li a')
5 console.log('Number of pages in the table:', await pages.count())
6
7 for(let p=0 ;p< await pages.count(); p++)
8 {
9     if(p>0)
10    {
11        await pages.nth(p).click()
12    }
13    for(let i=0;i<await rows.count();i++)
14    {
15        const row=rows.nth(i);
16        const tds=row.locator('td')
17
18        for(let j=0 ;j< await tds.count()-1;j++)
19        {
20            console.log(await tds.nth(j).textContent())
21        }
22    }
23    await page.waitForTimeout(3000);
24
25 }
26
27 await page.waitForTimeout(3000)
28 })
29
30
31 async function selectProduct(rows, page, name)
32 {
33     const machedRow= rows.filter({
34         has: page.locator('td'),
35         hasText: name
36     })
37     await machedRow.locator('input').check()
38 }
```



