HyperText Markup Language

HTML&CSS

STEP IT ACADEMY

Lesson 5

Tables

Contents

Lesson materials are attached to this PDF file. In order to get access to the materials, open the lesson in Adobe Acrobat Reader.

Tables

1. Create a Simple Table

Tables in HTML5 are used for presenting information in a structured way which is a two-dimensional table consisting of rows and columns of data cells.

Content of a cell can be anything: headers, lists, links, images, forms, and even other tables.

To create a table, use a paired tag <table> which is a container with all of the table's elements inside.

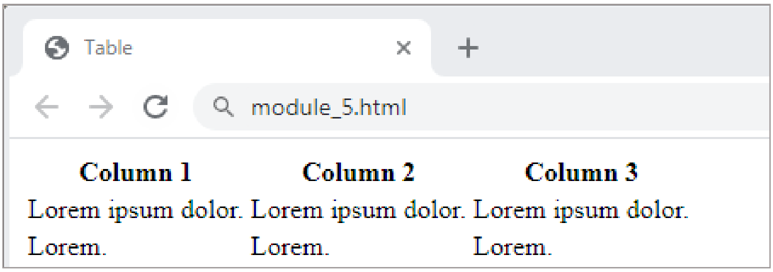
Table rows are created with the paired tag <tr> which is also a container for paired cell tags.

Table cells are divided into two types: cells for headers of a column or row (<th>) and cells for displaying table data (<td>).

Sample code for a simple table structure consisting of three rows and three columns:

```
<table>
        <tr>
            <th>Column 1</th>
            <th>Column 2</th>
            <th>Column 3</th>
        </tr>

        <tr>
            <td>Lorem ipsum dolor.</td>
            <td>Lorem ipsum dolor.</td>
            <td>Lorem ipsum dolor.</td>
        </tr>
        <tr>
            <td>Lorem.</td>
            <td>Lorem.</td>
            <td>Lorem.</td>
        </tr>
    </table>
```

This is how it is displayed in a browser:



Column 1	Column 2	Column 3
Lorem ipsum dolor.	Lorem ipsum dolor.	Lorem ipsum dolor.
Lorem.	Lorem.	Lorem.

Figure 1

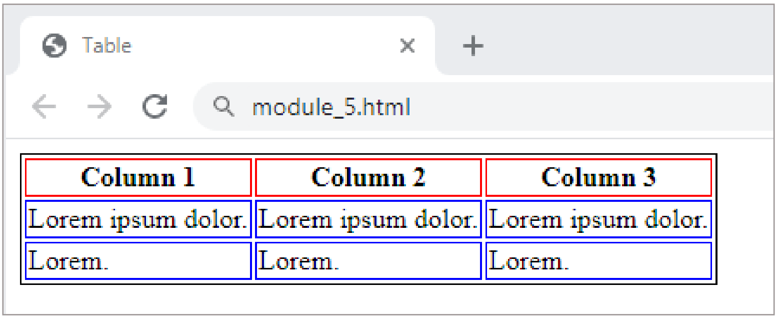
The first row cells are represented by the <th> tags. The content of these cells is visually different: the text is bold and centered in the cell.

Content of the cells in the main body of the table is aligned left in the cell.

Tables and cells have no visual borders by default. In order to make them appear, we need to use a style property border that replaced the obsolete border attribute of the <table> tag:

```
    /* outer borders of the entire table */
    table {border: 1px solid black;}

    /* cell borders of the column headers */
    th {border: 1px solid red;}

    /* cell borders of the main part */
    td {border: 1px solid blue;}
```

The table looks as follows now:



Table

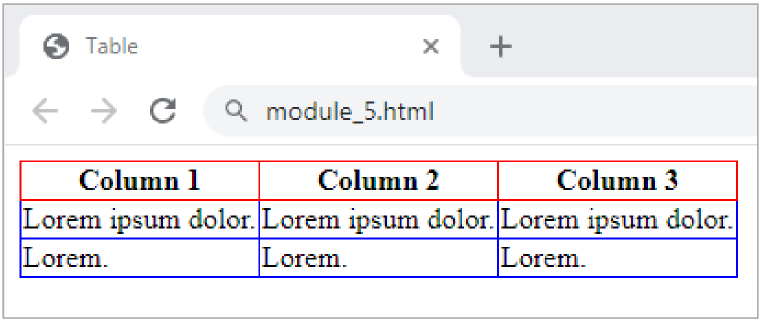
module_5.html

Column 1	Column 2	Column 3
Lorem ipsum dolor.	Lorem ipsum dolor.	Lorem ipsum dolor.
Lorem.	Lorem.	Lorem.

Figure 2

There are gaps between the table cells, and we can remove them using the style property "border-collapse: collapse;" set for the <table> tag.

```
table {
    border: 1px solid black;
    border-collapse: collapse;
}
th {border: 1px solid red;}
td {border: 1px solid blue;}
```



Table

module_5.html

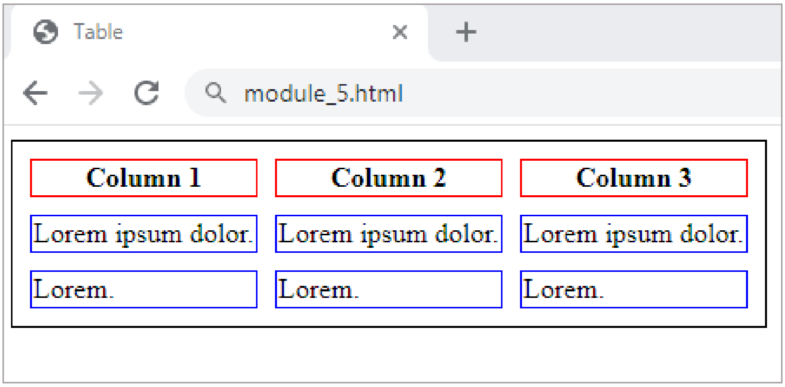
Column 1	Column 2	Column 3
Lorem ipsum dolor.	Lorem ipsum dolor.	Lorem ipsum dolor.
Lorem.	Lorem.	Lorem.

Figure 3

The Figure above shows that the border of the table is not displayed when set to "border-collapse: collapse";.

If you need to increase the space between cells, use the style property border-spacing that takes absolute or relative units of measure as a parameter. In this case the border-collapse takes a default separate value that sets spaces around the cell borders. For instance:

```
    table {
        border: 1px solid black;
        border-collapse: separate;
        border-spacing: 10px;
    }
```



Column 1	Column 2	Column 3
Lorem ipsum dolor.	Lorem ipsum dolor.	Lorem ipsum dolor.
Lorem.	Lorem.	Lorem.

Figure 4

The table size (width and height) is determined by the table's content. The size of a particular cell depends on the size of the neighboring cells. So, the width of any cell is determined by the size of the widest cell in the current column. The height of any cell is determined by the highest content in the current row.

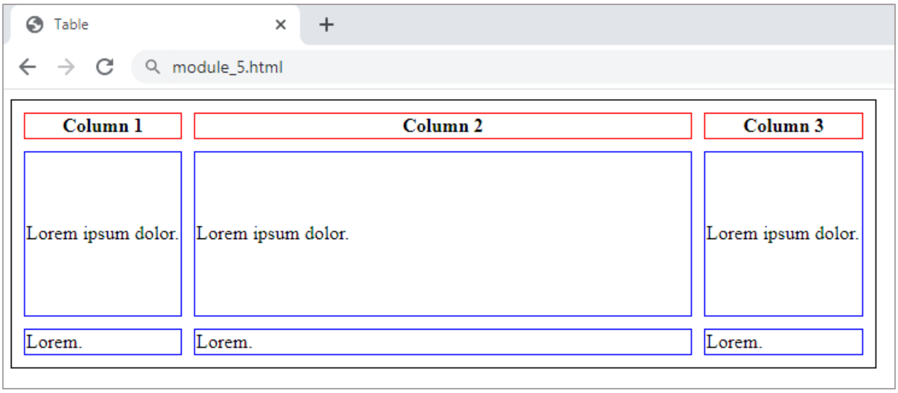
In order to set parameters to all cells, you just need to set width and height of one cell in each row and column. The style properties width and height are used now for this, they replaced the obsolete attributes width and height.

For convenient access to one cell of the table, we will write an identifier for it thus making our cell unique.

```
<table>
    <tr>
        <th>Column 1</th>
        <th>Column 2</th>
        <th>Column 3</th>
    </tr>

    <tr>
        <td>Lorem ipsum dolor.</td>
        <td id="uniq">Lorem ipsum dolor.</td>
        <td>Lorem ipsum dolor.</td>
    </tr>

    <tr>
        <td>Lorem.</td>
        <td>Lorem.</td>
        <td>Lorem.</td>
    </tr>

</table>
```

Let's access the cell in styles and set a width and height for it:

```
#uniq{
    width: 30vw;
    height: 20vh;
}
```



Column 1	Column 2	Column 3
Lorem ipsum dolor.	Lorem ipsum dolor.	Lorem ipsum dolor.
Lorem.	Lorem.	Lorem.

Figure 5

The Figure above shows that the size of one cell influences the size of all cells and the entire table.

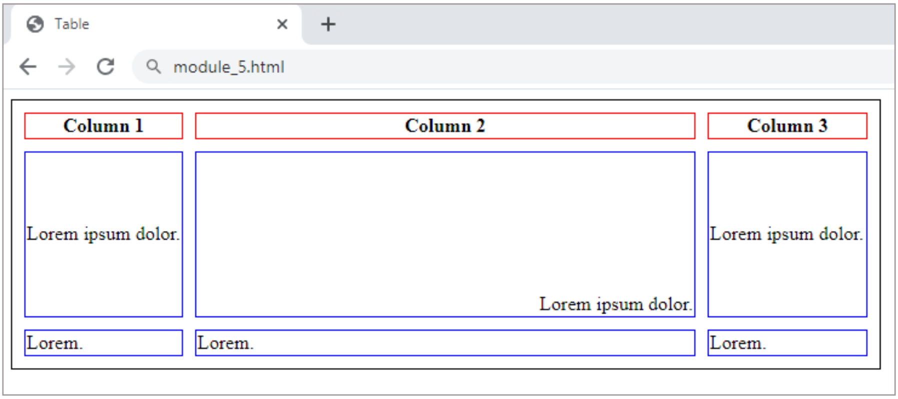
The content is aligned inside a cell with style properties that replaced the obsolete attributes align (*horizontal alignment*) and valign (*vertical alignment*):

- **text-align** is used to align text horizontally and can take the following values:
  - ▷ center — align center;
  - ▷ left — align left;
  - ▷ right — align right;
  - ▷ justify — align with the justified format;
  - ▷ initial — sets the default value;
  - ▷ inherit — inherits the value of the parent element.
- **vertical-align** aligns text and inline elements in the cell vertically and can take the following values:
  - ▷ baseline — aligns with the baseline (by default);
  - ▷ top — aligns with the top;
  - ▷ middle — is placed in the middle;

- ▹ bottom — aligns with the bottom;
- ▹ initial — sets the default value;
- ▹ inherit — inherits the value of the parent element.

Let’s align the text of our unique cell bottom right:

```
    #uniq{
        width: 30vw;
        height: 20vh;
        text-align: right;
        vertical-align: bottom;
    }
```

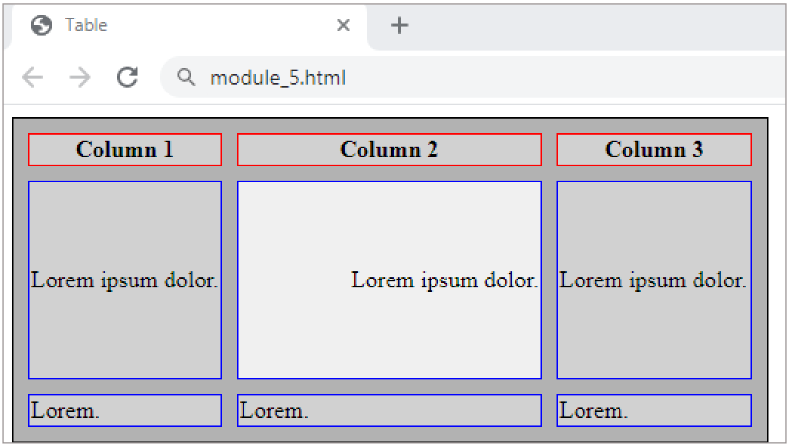


Column 1	Column 2	Column 3
Lorem ipsum dolor.	Lorem ipsum dolor.	Lorem ipsum dolor.
Lorem.	Lorem.	Lorem.

Figure 6

In order to change the background of a table, row, or column, you should access the required element in the style file and set the background-color property for it.

```
table {
        border: 1px solid black;
        border-collapse: separate;
        border-spacing: 10px;
        /* background color for the table */
        background-color: #aaaaaa;
    }
    tr{
        /* background color for the row */
        background-color: #cccccc;
    }
    th {
        border: 1px solid red;
    }
    td {
        border: 1px solid blue;
    }
    #uniq{
        width: 30vw;
        height: 20vh;
        text-align: right;
        vertical-align: center;
        /* background color for the cell */
        background-color: #eeeeee;
    }
```

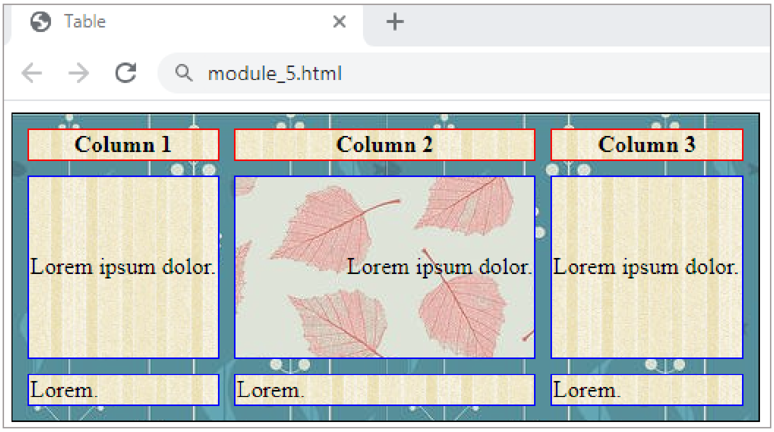


Column 1	Column 2	Column 3
Lorem ipsum dolor.	Lorem ipsum dolor.	Lorem ipsum dolor.
Lorem.	Lorem.	Lorem.

Figure 7

In order to set an image as a background for a table, row, or cell, access the required element in the style file and set the background-image property for it.

```
table {
        border: 1px solid black;
        border-collapse: separate;
        border-spacing: 10px;
        /* background image for the table.
           If the path to the image has no spaces
           or special characters, you can write
           it without quotes
        */
        background-image: url(bg_1.jpg);
    }

    tr{
        /* background image for the row */
        background-image: url(bg_2.jpg);
    }

    th {
        border: 1px solid red;
    }

    td {
        border: 1px solid blue;
    }

    #uniq{
        width: 30vw;
        height: 20vh;
        text-align: right;
        vertical-align: center;
        /* background image for the cell */
        background-image: url(bg_3.jpg);
    }
```



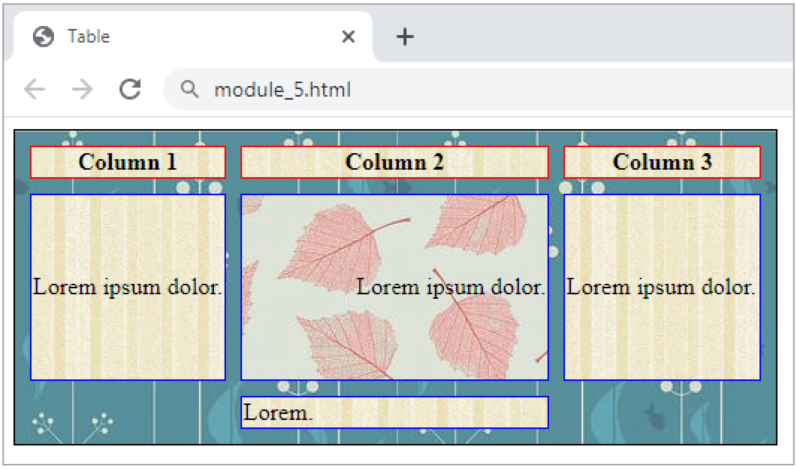
Column 1	Column 2	Column 3
Lorem ipsum dolor.	Lorem ipsum dolor.	Lorem ipsum dolor.
Lorem.	Lorem.	Lorem.

Figure 8

If a cell is empty (no content), you can hide it using the property "empty-cells:hide;" set for the table tag.

```
<table>
    <tr>
        <th>Column 1</th>
        <th>Column 2</th>
        <th>Column 3</th>
    </tr>

    <tr>
        <td>Lorem ipsum dolor.</td>
        <td id="uniq">Lorem ipsum dolor.</td>
        <td>Lorem ipsum dolor.</td>
    </tr>

    <tr>
        <td></td><!-- empty cell -->
        <td>Lorem.</td>
        <td></td><!-- empty cell -->
    </tr>
</table>
```

```
table {
    border: 1px solid black;
    border-collapse: separate;
    border-spacing: 10px;

    background-image: url(bg_1.jpg);
    /* hide empty cells */
    empty-cells: hide;
}
```



Column 1	Column 2	Column 3
Lorem ipsum dolor.	Lorem ipsum dolor.	Lorem ipsum dolor.
	Lorem.	

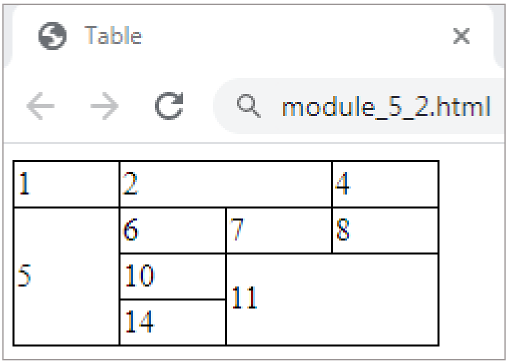
Figure 9

The Figure above shows that the cells without content are hidden in the last row.

2. Spanning cells: colspan and rowspan Attributes

Sometimes cells should be spanned in a table, for instance, in a table header for a common name of table’s columns.

Let’s create a table that will have 4 rows with 4 cells, and we are going to bring it to the following form step by step:



1	2		4
5	6	7	8
	10	11	
	14		

Figure 10

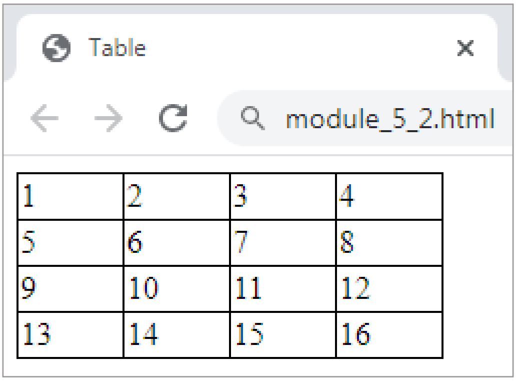
Below is the initial code of the table. Cells are numbered for convenience:

```
<table>
    <tr>
        <td>1</td>
        <td>2</td>
        <td>3</td>
        <td>4</td>
    </tr>
    <tr>
        <td>5</td>
        <td>6</td>
        <td>7</td>
        <td>8</td>
    </tr>
    <tr>
        <td>9</td>
        <td>10</td>
        <td>11</td>
        <td>12</td>
    </tr>

    <tr>
```

```
            <td>13</td>
            <td>14</td>
            <td>15</td>
            <td>16</td>
        </tr>
    </table>
```

And the style properties:

```
    table {
        border-collapse: collapse;
    }
    td {
        border: 1px solid black;
        width: 50px;
        height: 20px;
    }
```



1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

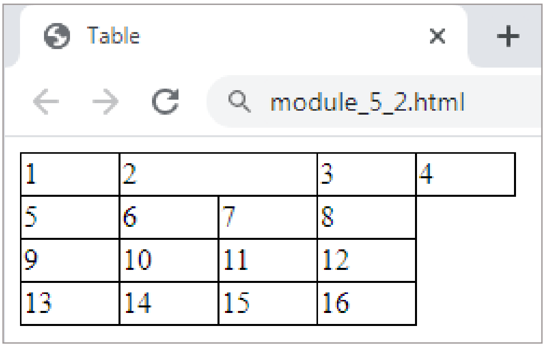
Figure 11

In order to span a cell horizontally, we need the colspan attribute. An integer equal to the number of cells spanned horizontally in one row is set as the colspan attribute.

Let's span two cells (numbered as 2 and 3) horizontally in the first row. For this write the attribute colspan="2" in cell 2.

```
        <tr>
            <td>1</td>
            <td colspan="2">2</td>
            <td>3</td>
            <td>4</td>
        </tr>
```

If we take a look at the table right now, we will see that cell 2 is spanned, and cell 4 goes beyond the table. This is because this cell became redundant and took place of the fifth cell in the column that is not in our table.

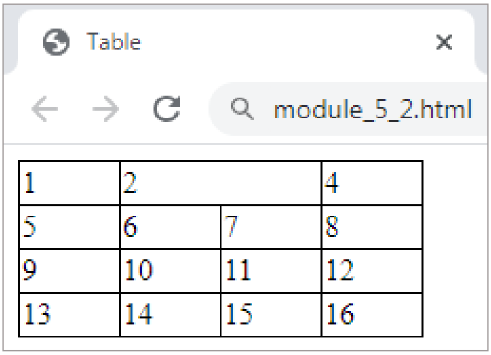


1	2		3	4
5	6	7	8	
9	10	11	12	
13	14	15	16	

Figure 12

In order to avoid this, you just need to remove or comment out the "extra" cell. Let's comment out cell 3 since we have spanned it earlier.

```
        <tr>
            <td>1</td>
            <td colspan="2">2</td>
            <!-- <td>3</td> -->
            <td>4</td>
        </tr>
```



1	2		4
5	6	7	8
9	10	11	12
13	14	15	16

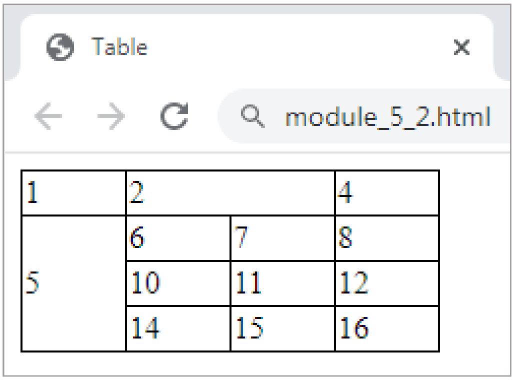
Figure 13

In order to span cells vertically, use the rowspan attribute. An integer equal to the number of cells spanned vertically in one column is set as a rowspan attribute.

Let's span vertically three cells numbered as 5, 9, and 13. For this set the attribute rowspan="3" for cell 5. And delete or comment out the cells numbered as 9 and 13.

```
<table>
    <tr>
        <td>1</td>
        <td colspan="2">2</td>
        <!-- <td>3</td> -->
        <td>4</td>
    </tr>

    <tr>
        <td rowspan="3">5</td>
        <td>6</td>
```

```
            <td>7</td>
            <td>8</td>
        </tr>

        <tr>
            <!-- <td>9</td> -->
            <td>10</td>
            <td>11</td>
            <td>12</td>
        </tr>

        <tr>
            <!-- <td>13</td> -->
            <td>14</td>
            <td>15</td>
            <td>16</td>
        </tr>
    </table>
```



1	2		4
5	6	7	8
	10	11	12
	14	15	16

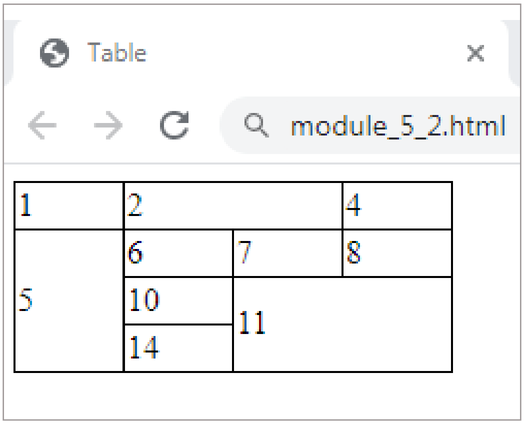
Figure 14

You can span cells vertically and horizontally by writing two attributes colspan and rowspan for one cell, the order does not matter.

Let’s span cells 11, 12, 15, and 16. For this write the attributes colspan=”2” and rowspan=”2” in cell 11. And delete or comment out the rest of the cells numbered as 12, 15, and 16.

```
<table>

    <tr>
        <td>1</td>
        <td colspan="2">2</td>
        <!-- <td>3</td> -->
        <td>4</td>
    </tr>

    <tr>
        <td rowspan="3">5</td>
        <td>6</td>
        <td>7</td>
        <td>8</td>
    </tr>

    <tr>
        <!-- <td>9</td> -->
        <td>10</td>
        <td colspan="2" rowspan="2">11</td>
        <!-- <td>12</td> -->
    </tr>

    <tr>
        <!-- <td>13</td> -->
        <td>14</td>
        <!-- <td>15</td> -->
        <!-- <td>16</td> -->
    </tr>

</table>
```

This is what we have got:



1	2		4
5	6	7	8
	10	11	
	14		

Figure 15

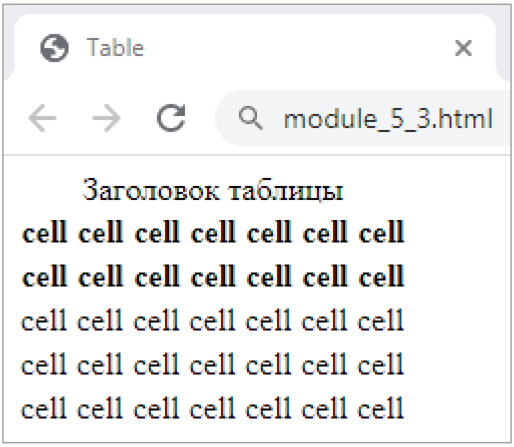
When you span cells, remember one rule: each row and each column must have at least one non-spanned cell. Otherwise, you face a situation when all row cells become "redundant," and the row will not display in the table.

3. Tags for Captions, Logical Grouping, and Logical Structuring

In order to set a table caption, use a <caption> tag. This tag immediatelly follows the opening tag <table> and is outside a row or cell of the table. Let's consider how to use this tag through an example. Create a table consisting of 5 rows and 7 columns and set its caption. Below is a snippet of code with a caption:

```
<table>
        <caption>Table caption</caption>
        <tr>
```

The table is displayed in a browser as follows:



Заголовок таблицы

cell cell cell cell cell cell cell

cell cell cell cell cell cell cell

cell cell cell cell cell cell cell

cell cell cell cell cell cell cell

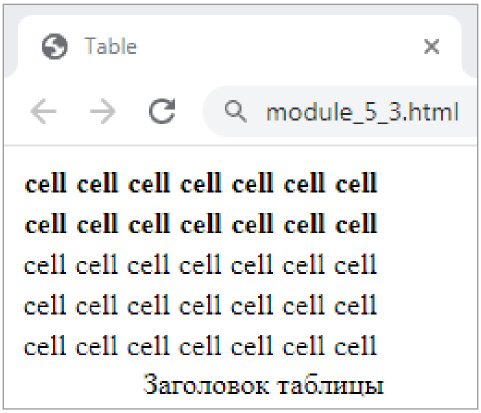
cell cell cell cell cell cell cell

Figure 16

The table's caption is at the top of the table and centered by default. We can change the position of the caption using styles. To align horizontally, use the property text-align with the values left, center, and right. To change the position of the caption above or below the table, use the caption-side property with the values top and bottom in the table. The snippet below aligns the caption right and places it under the table (Fig. 17):

```
caption{
     text-align: right;
     caption-side: bottom;
}
```

It is often required to highlight cells that have common content in a table. We have tags that combine cells into logical groups for this.



cell cell cell cell cell cell cell
cell cell cell cell cell cell cell
cell cell cell cell cell cell cell
cell cell cell cell cell cell cell
cell cell cell cell cell cell cell
Заголовок таблицы

Figure 17

A logical group of columns uses tags <colgroup> and <col> that are written inside the <table> tag, right after the <caption> tag if any. The elements <colgroup> and <col> are used for a uniform, vertical formatting of cells.

The <colgroup> tag is a container that has the <col> tags. The attribute used to specify the number of combined columns is span whose values is an integer.

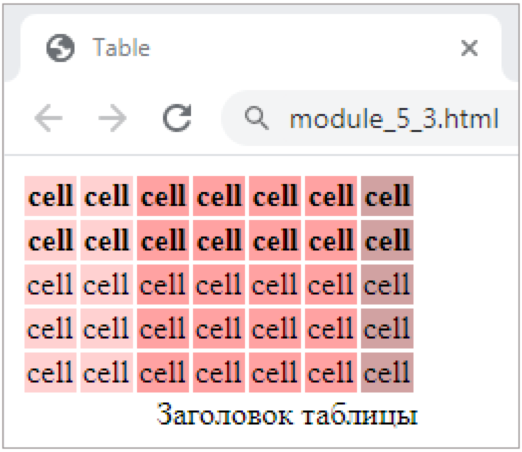
Let's consider it through our example. Span the cells of three columns as follows: the first column will have 2 columns, the second will have four columns, and the third will have one column. Use classes to set a different background for columns.

```
<table>
    <caption>Table caption</caption>
    <colgroup>
        <col span="2" class="col-1">
        <col span="4" class="col-2">
        <col class="col-3">
    </colgroup>
    <tr>
```

Write column backgrounds in the styles:

```
.col-1{
    background-color: #ffcccc;
}
.col-2{
    background-color: #ff9999;
}
.col-3{
    background-color: #cc9999;
}
```

The table looks as follows now:



cell	cell	cell	cell	cell	cell	cell
cell	**cell**	**cell**	**cell**	**cell**	**cell**	**cell**
cell	cell	cell	cell	cell	cell	cell
cell	cell	cell	cell	cell	cell	cell
cell	cell	cell	cell	cell	cell	cell

Заголовок таблицы

Figure 18

In order to logically structure table data, use the tags <thead>, <tbody>, <tfoot>. Let's consider each of them in more detail.

The <thead> tag is used as a container for table rows and applied for spanning rows and cells for column captions, thereby forming the table's header. It is written inside the <table>

tag as the first item or immediately after the tags <caption> and <colgroup> if any. You can use only one <thead> element in a table.

The <tfoot> tag is intended for a footer of the table, and it is also used as a container for rows. You can use only one <tfoot> in a table, it immediately follows the <thead> tag.

The <tbody> tag is used as a container for table rows containing cells with the main content of the table, thereby forming the "body" of the table. You can use only one <tbody> element in a table, it follows the tags <thead> and <tfoot>.

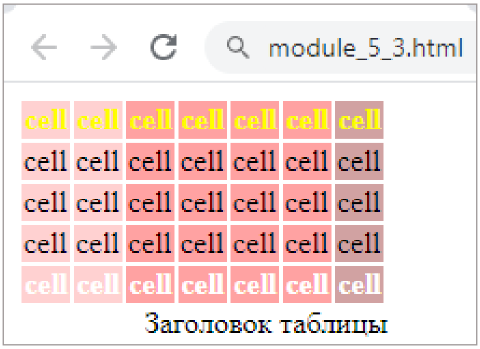
Let's consider the use of these tags in our example. The full code of the table is as follows:

```
<table>
        <caption>Table caption</caption>
        <colgroup>
            <col span="2" class="col-1">
            <col span="4" class="col-2">
            <col class="col-3">
        </colgroup>
        <thead>
            <tr>
                <th>cell</th>
                <th>cell</th>
                <th>cell</th>
                <th>cell</th>
                <th>cell</th>
                <th>cell</th>
                <th>cell</th>
            </tr>
        </thead>
        <tfoot>
            <tr>
                <th>cell</th>
                <th>cell</th>
                <th>cell</th>
                <th>cell</th>
                <th>cell</th>
                <th>cell</th>
                <th>cell</th>
            </tr>
        </tfoot>
        <tbody>
            <tr>
                <td>cell</td>
                <td>cell</td>
                <td>cell</td>
                <td>cell</td>
                <td>cell</td>
                <td>cell</td>
                <td>cell</td>
            </tr>
            <tr>
                <td>cell</td>
                <td>cell</td>
                <td>cell</td>
                <td>cell</td>
                <td>cell</td>
                <td>cell</td>
                <td>cell</td>
            </tr>
            <tr>
                <td>cell</td>
                <td>cell</td>
                <td>cell</td>
                <td>cell</td>
                <td>cell</td>
                <td>cell</td>
                <td>cell</td>
            </tr>
        </tbody>
    </table>
```

Let’s highlight the header and footer using a style for the text color:

```
thead{ color: #ffff00; }
tfoot{ color: #ffffff; }
```

We set yellow for text in the header of the table and white for the text in the footer. The Figure shows that the footer cells are at the bottom of the table.



cell	cell	cell	cell	cell	cell	cell
cell	cell	cell	cell	cell	cell	cell
cell	cell	cell	cell	cell	cell	cell
cell	cell	cell	cell	cell	cell	cell
cell	cell	cell	cell	cell	cell	cell

Заголовок таблицы

Figure 19

4. Handling Table Borders

Data structured as tables were used since the very beginning of HTML, so a table has lots of obsolete attributes with attributes that set the table borders among them. These attributes are:

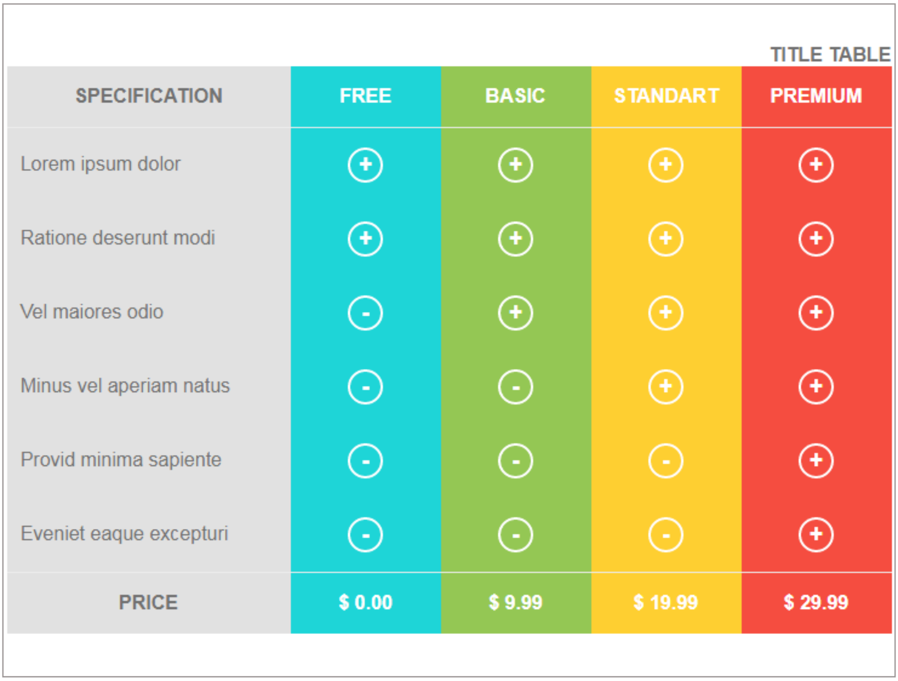
- border — to set the border’s thickness in pixels;
- bordercolor — to set border’s color;
- frame — specifies how borders around the table should be displayed;
- rules — specify where to display borders between cells.

The attributes listed above must be replaced with the style property border to set borders on all sides or the style properties border-top, border-bottom, border-left, border-right to specify a border on a specific side.

5. Practice: Create Complex Tables

Let's create as an example a complex table using all the tags we have learned.

The table will consist of 8 rows and 5 columns. Each column will represent an individual vertical column. The table will also have a header, footer, and body. In the end, our table should look like in the Figure below:



TITLE TABLE

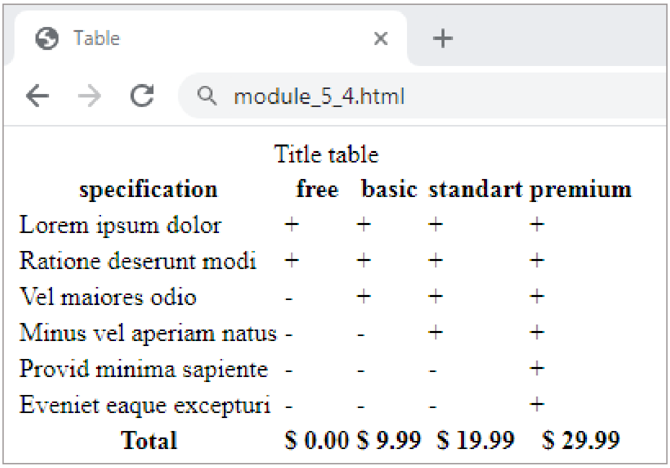
SPECIFICATION	FREE	BASIC	STANDART	PREMIUM
Lorem ipsum dolor	⊕	⊕	⊕	⊕
Ratione deserunt modi	⊕	⊕	⊕	⊕
Vel maiores odio	⊖	⊕	⊕	⊕
Minus vel aperiam natus	⊖	⊖	⊕	⊕
Provid minima sapiente	⊖	⊖	⊖	⊕
Eveniet eaque excepturi	⊖	⊖	⊖	⊕
PRICE	$ 0.00	$ 9.99	$ 19.99	$ 29.99

Figure 20

Let's create *index.html* and write the following code in it:

```
    <table>
        <caption>Title table</caption>
        <colgroup>
            <col class="col-1">
            <col class="col-2">
            <col class="col-3">
            <col class="col-4">
            <col class="col-5">
        </colgroup>
        <thead>
            <tr>
                <th>specification</th>
                <th>free</th>
                <th>basic</th>
                <th>standard</th>
                <th>premium</th>
            </tr>
        </thead>
        <tfoot>
            <tr>
                <th>Total</th>
                <th>$ 0.00</th>
                <th>$ 9.99</th>
                <th>$ 19.99</th>
                <th>$ 29.99</th>
            </tr>
        </tfoot>
        <tbody>
            <tr>
                <td>Lorem ipsum dolor</td>
                <td>+</td>
                <td>+</td>
                <td>+</td>
                <td>+</td>
            </tr>
        <tr>
            <td>Ratione deserunt modi</td>
            <td>+</td>
            <td>+</td>
            <td>+</td>
            <td>+</td>
        </tr>
        <tr>
            <td>Vel maiores odio</td>
            <td>-</td>
            <td>+</td>
            <td>+</td>
            <td>+</td>
        </tr>
        <tr>
            <td>Minus vel aperiam natus</td>
            <td>-</td>
            <td>-</td>
            <td>+</td>
            <td>+</td>
        </tr>
        <tr>
            <td>Provid minima sapiente</td>
            <td>-</td>
            <td>-</td>
            <td>-</td>
            <td>+</td>
        </tr>
        <tr>
            <td>Eveniet eaque excepturi</td>
            <td>-</td>
            <td>-</td>
            <td>-</td>
            <td>+</td>
        </tr>
    </tbody>
</table>
```

The table looks as follows in a browser:



Title table

specification	free	basic	standart	premium
Lorem ipsum dolor	+	+	+	+
Ratione deserunt modi	+	+	+	+
Vel maiores odio	-	+	+	+
Minus vel aperiam natus	-	-	+	+
Provid minima sapiente	-	-	-	+
Eveniet eaque excepturi	-	-	-	+
Total	**$ 0.00**	**$ 9.99**	**$ 19.99**	**$ 29.99**

Figure 21

Let’s get down to styles:

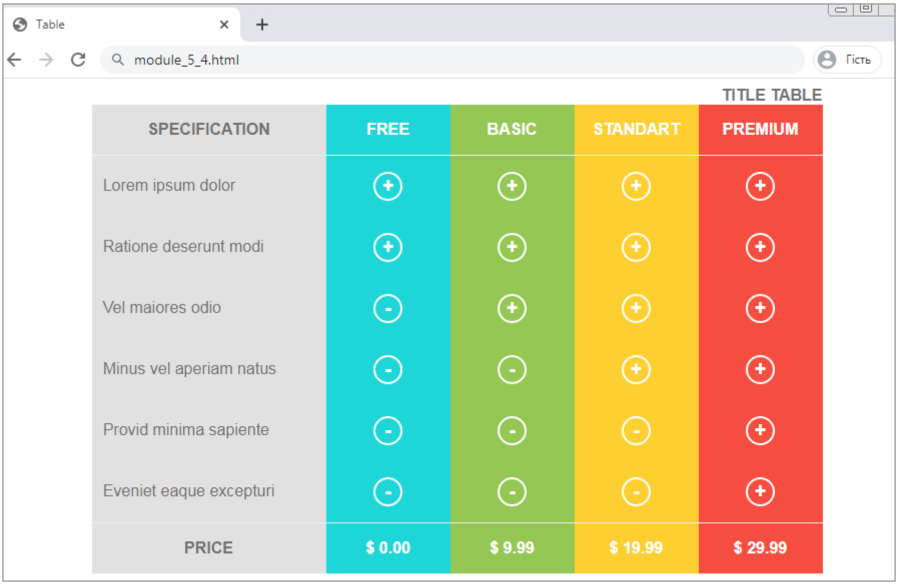
```
table{
    /* set the table's width in % of the body width */
    width: 80%;
    /*
      to prevent the table from extra shrinking or
      stretching when resizing the browser window, write
      the minimum and maximum width in pixels for them
    */
    min-width: 650px;
    max-width: 800px;
    /* place the table in the middle, horizontally */
    margin: 0 auto;
    /* remove margins between cells */
    border-collapse: collapse;
    /* set a system font for the text */
    font-family: Arial, Helvetica, sans-serif;
}
```

```
caption{
    /* align the caption right, horizontally */
    text-align: right;
    /* transform text to uppercase */
    text-transform: uppercase;
    /* set the color of the table caption */
    color: #666666;
    /* set the font weight */
    font-weight: 700;
}
/*
  transform column headers in the header and footer
  to uppercase and set them to white */
th{
    text-transform: uppercase;
    color: #ffffff;
}
/*
   set a shade of gray for the first cell of column headers
   in the header and footer */
tr th:first-child{
    color: #666666;
}
/*
  center horizontally all the cells of the table
  and set padding at the top and bottom for them */
td, th{
    text-align: center;
    padding: 1em 0;
}
/*
  align left the first cell of each row, set gray
  and padding left for them */
tr td:first-child{
    text-align: left;
    color: #666666;
    padding-left: 0.7em;
}
```

```css
/*
  the content of the main cells - span elements -
  should be designed as follows: */
span{
    color: #ffffff;
    font-weight: 700;
    font-size: 1.2em;
    display: inline-block;
    border: 2px solid  #ffffff;
    border-radius: 50%;
    width: 25px;
    height: 25px;
    line-height: 25px;
}

/*
  set a background color and width for each column into
  which we have split our table; four columns will have
  the same width, and the width of the first column
  will be calculated automatically */
.col-1{
    background-color: #dddddd;
    width: auto;
}
.col-2{
    background-color: #17d0d3;
    width: 17%;
}
.col-3{
    background-color: #8ac149;
    width: 17%;
}
.col-4{
    background-color: #feca28;
    width: 17%;
}
```

```
.col-5{
    background-color: #f44236;
    width: 17%;
}

thead{
    /* set the bottom border for the table's header */
    border-bottom: 1px solid #ebebeb;
}

tfoot{
    /* set the top border for the table's footer */
    border-top: 1px solid #ebebeb;
}
```

In the end, our table looks as follows:



TITLE TABLE

SPECIFICATION	FREE	BASIC	STANDART	PREMIUM
Lorem ipsum dolor	+	+	+	+
Ratione deserunt modi	+	+	+	+
Vel maiores odio	-	+	+	+
Minus vel aperiam natus	-	-	+	+
Provid minima sapiente	-	-	-	+
Eveniet eaque excepturi	-	-	-	+
PRICE	$ 0.00	$ 9.99	$ 19.99	$ 29.99

Figure 22

6. Table Layout: What It Is and Why not Use It in the Modern Standard?

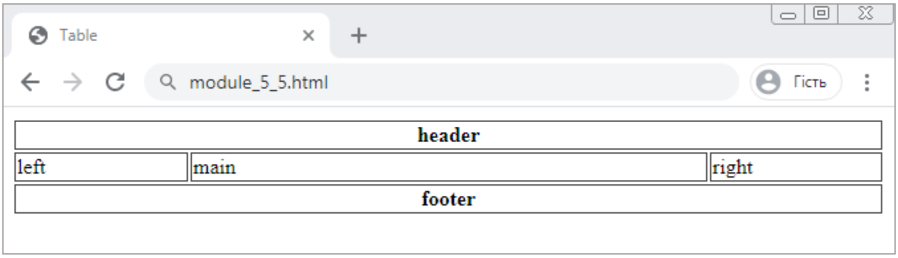
A table layout is a way to create a layout of a web page where the structural basis is a table. In this case, the <body> of the table has one child element <table> whose cells define the header, footer, and body of a web page.

Let's consider a small example using table layout. Create an html page with the following content:

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport"
          content="width=device-width, initial-scale=1.0">
    <title>Table</title>
    <style>
        table{
            width: 100%;
        }
        th,td{
            border: 1px solid #333;
        }
    </style>
</head>

<body>
    <table>
        <thead>
            <tr>
                <th colspan="3">header</th>
            </tr>
        </thead>
        <tfoot>
```

```
            <tr>
                <th colspan="3">footer</th>
            </tr>
        </tfoot>
        <tbody>
            <tr>
                <td width="20%">left</td>
                <td width="60%">main</td>
                <td width="20%">right</td>
            </tr>
        </tbody>
    </table>
</body>

</html>
```

The result is a web page like in the Figure below. The main structural elements of the page are cells with the relevant content.



```
header
left | main | right
footer
```

Figure 23

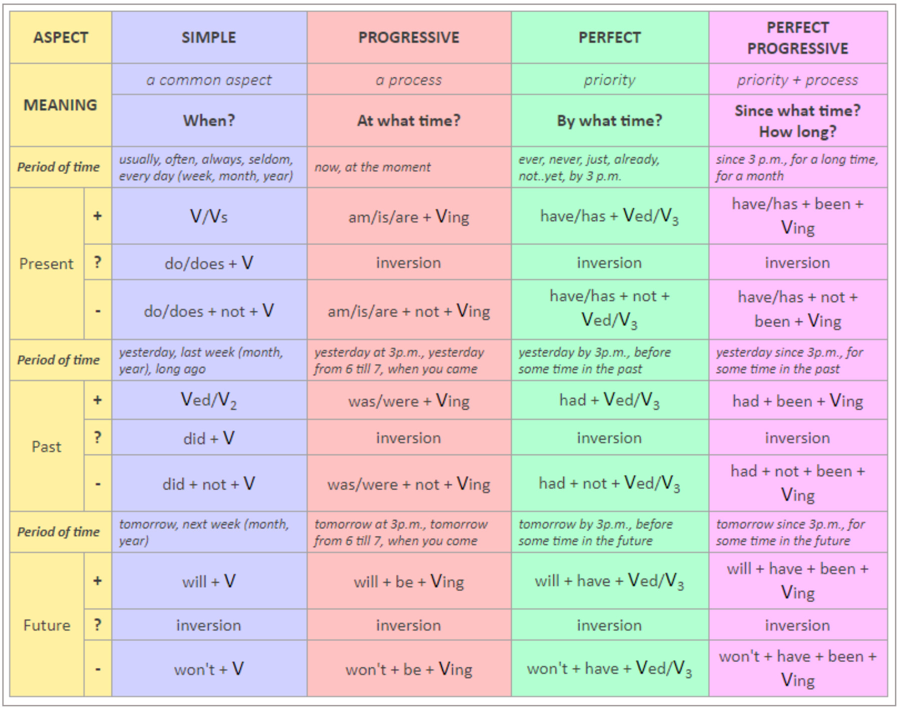
Table layout was widely used back then when the CSS standard did not exist yet, and tables were the only way to arrange elements on a page in a certain order. But in this case, the main principle of the hypertext markup language was violated: the visual display should not depend on the logical elements of the page structure. It means that in the case of table layout, the <table> tag was used not as intended.

With the advent of CSS it became possible to separate the logical structure of a document and its visual display, and a block layout appeared as an alternative to table layout.

The current standard HTML5 involves the use of tables only to display table data: comparison, timetables and schedules, statistics, and so on.

Homework

Create a layout of this table:



ASPECT		SIMPLE	PROGRESSIVE	PERFECT	PERFECT PROGRESSIVE
MEANING		*a common aspect*	*a process*	*priority*	*priority + process*
		When?	**At what time?**	**By what time?**	**Since what time? How long?**
Period of time		*usually, often, always, seldom, every day (week, month, year)*	*now, at the moment*	*ever, never, just, already, not..yet, by 3 p.m.*	*since 3 p.m., for a long time, for a month*
Present	+	V/Vs	am/is/are + Ving	have/has + Ved/V_3	have/has + been + Ving
	?	do/does + V	inversion	inversion	inversion
	-	do/does + not + V	am/is/are + not + Ving	have/has + not + Ved/V_3	have/has + not + been + Ving
Period of time		*yesterday, last week (month, year), long ago*	*yesterday at 3p.m., yesterday from 6 till 7, when you came*	*yesterday by 3p.m., before some time in the past*	*yesterday since 3p.m., for some time in the past*
Past	+	Ved/V_2	was/were + Ving	had + Ved/V_3	had + been + Ving
	?	did + V	inversion	inversion	inversion
	-	did + not + V	was/were + not + Ving	had + not + Ved/V_3	had + not + been + Ving
Period of time		*tomorrow, next week (month, year)*	*tomorrow at 3p.m., tomorrow from 6 till 7, when you come*	*tomorrow by 3p.m., before some time in the future*	*tomorrow since 3p.m., for some time in the future*
Future	+	will + V	will + be + Ving	will + have + Ved/V_3	will + have + been + Ving
	?	inversion	inversion	inversion	inversion
	-	won't + V	won't + be + Ving	won't + have + Ved/V_3	won't + have + been + Ving

Figure 24

Lesson 5.
Tables