XPATH

query language for xml

tutorialspoint

SIMPLYEASYLEARNING

www.tutorialspoint.com

https://www.facebook.com/tutorialspointindia

https://twitter.com/tutorialspoint

About the Tutorial

XPath is a query language that is used for traversing through an XML document. It is used commonly to search particular elements or attributes with matching patterns.

This tutorial explains the basics of XPath. It contains chapters discussing all the basic components of XPath with suitable examples.

Audience

This tutorials has been designed for beginners to help them understand the basic concepts related to XPath. This tutorial will give you enough understanding on XPath from where you can take yourself to higher levels of expertise.

Prerequisites

Before proceeding with this tutorial, you should have basic knowledge of XML, HTML, and JavaScript.

Disclaimer & Copyright

Table of Contents

1. XPath – Overview

Before learning XPath, we should first understand XSL which stands for **E**xtensible **S**tylesheet **L**anguage. It is similar to XML as CSS is to HTML.

Need for XSL

In case of HTML documents, tags are predefined such as table, div, span, etc. The browser knows how to add style to them and display them using CSS styles. But in case of XML documents, tags are not predefined. In order to understand and style an XML document, **World Wide Web Consortium (W3C)** developed XSL which can act as an XML-based Stylesheet Language. An XSL document specifies how a browser should render an XML document.

Following are the main parts of XSL:

- **XSLT** — used to transform XML documents into various other types of document.
- **XPath** — used to navigate XML documents.
- **XSL-FO** — used to format XML documents.

What is XPath?

XPath is an official recommendation of the World Wide Web Consortium (W3C). It defines a language to find information in an XML file. It is used to traverse elements and attributes of an XML document. XPath provides various types of expressions which can be used to enquire relevant information from the XML document.

- **Structure Definitions** — XPath defines the parts of an XML document like element, attribute, text, namespace, processing-instruction, comment, and document nodes
- **Path Expressions** — XPath provides powerful path expressions select nodes or list of nodes in XML documents.
- **Standard Functions —** XPath provides a rich library of standard functions for manipulation of string values, numeric values, date and time comparison, node and QName manipulation, sequence manipulation, Boolean values etc.
- **Major part of XSLT —** XPath is one of the major elements in XSLT standard and is must have knowledge in order to work with XSLT documents.
- **W3C recommendation —** XPath is an official recommendation of World Wide Web Consortium (W3C).

One should keep the following points in mind, while working with XPath:

- XPath is core component of XSLT standard.
- XSLT cannot work without XPath.
- XPath is basis of XQuery and XPointer.

2. XPath — Expression

An XPath expression generally defines a pattern in order to select a set of nodes. These patterns are used by XSLT to perform transformations or by XPointer for addressing purpose.

XPath specification specifies seven types of nodes which can be the output of execution of the XPath expression.

- Root
- Element
- Text
- Attribute
- Comment
- Processing Instruction
- Namespace

XPath uses a path expression to select node or a list of nodes from an XML document.

Following is the list of useful paths and expression to select any node/ list of nodes from an XML document.

Expression	Description
node-name	Select all nodes with the given name "nodename"
/	Selection starts from the root node
//	Selection starts from the current node that match the selection
.	Selects the current node
..	Selects the parent of the current node
@	Selects attributes
student	Example: Selects all nodes with the name "student"

class/student	Example: Selects all student elements that are children of class
//student	Selects all student elements no matter where they are in the document

Example

In this example, we've created a sample XML document, students.xml and its stylesheet document **students.xsl** which uses the XPath expressions under **select** attribute of various XSL tags to get the values of roll no, firstname, lastname, nickname and marks of each student node.

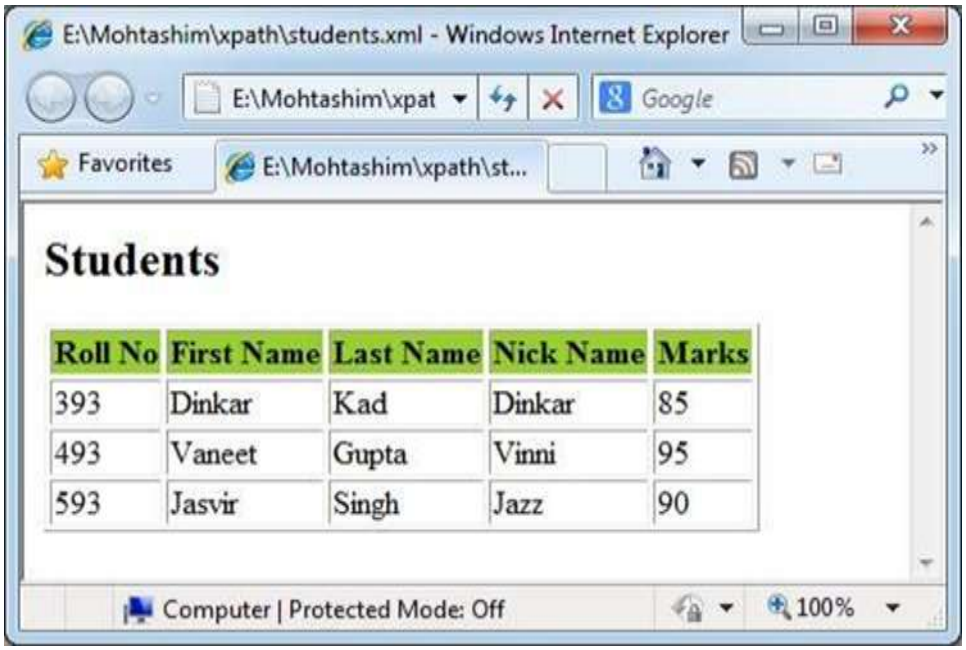
students.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="students.xsl"?>
<class>
   <student rollno="393">
      <firstname>Dinkar</firstname>
      <lastname>Kad</lastname>
      <nickname>Dinkar</nickname>
      <marks>85</marks>
   </student>
   <student rollno="493">
      <firstname>Vaneet</firstname>
      <lastname>Gupta</lastname>
      <nickname>Vinni</nickname>
      <marks>95</marks>
   </student>
   <student rollno="593">
      <firstname>Jasvir</firstname>
      <lastname>Singh</lastname>
      <nickname>Jazz</nickname>
      <marks>90</marks>
   </student>
</class>
```

students.xsl

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
  <h2>Students</h2>
  <table border="1">
    <tr bgcolor="#9acd32">
        <th>Roll No</th>
      <th>First Name</th>
      <th>Last Name</th>
        <th>Nick Name</th>
        <th>Marks</th>
    </tr>
    <xsl:for-each select="class/student">
    <tr>
        <td>
             <xsl:value-of select="@rollno"/>
        </td>
      <td><xsl:value-of select="firstname"/></td>
      <td><xsl:value-of select="lastname"/></td>
        <td><xsl:value-of select="nickname"/></td>
        <td><xsl:value-of select="marks"/></td>
    </tr>
    </xsl:for-each>
  </table>
  </body>
  </html>
</xsl:template>

</xsl:stylesheet>
```

Verify the output



3. XPath — Nodes

In this chapter, we'll see the XPath expression in details covering common types of Nodes, XPath defines and handles.

S.N.	Node Type & Description
1	**Root** Root element node of an XML Document.
2	**Element** Element node.
3	**Text** Text of an element node.
4	**Attribute** Attribute of an element node.
5	**Comment** Comment

Let us now understand the nodes in detail.

XPath Root Node

Following are the ways to get root element and do the processing afterwards.

Use Wildcard

Use **/***, wild card expression to select the root node.

```
<p><xsl:value-of select="name(/*)"/></p>
```

Use Name

Use /class, to select root node by name.

```
<p><xsl:value-of select="name(/class)"/></p>
```

Use Name with wild card

Use /class/*, select all element under root node.

```
<p><xsl:value-of select="name(/class/*)"/></p>
```

Example

In this example, we've created a sample XML document **students.xml** and its stylesheet document students.xsl which uses the XPath expressions.

Following is the sample XML used.

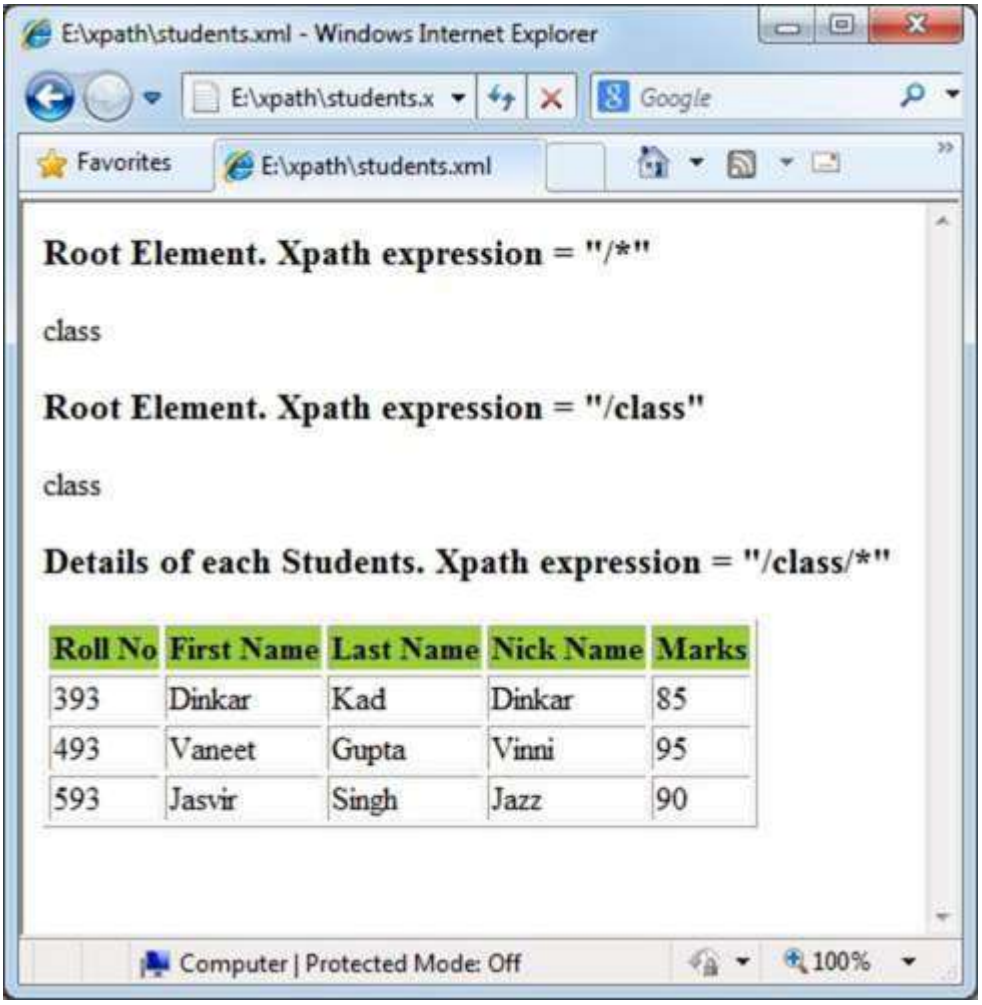
students.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="students.xsl"?>
<class>
   <student rollno="393">
      <firstname>Dinkar</firstname>
      <lastname>Kad</lastname>
      <nickname>Dinkar</nickname>
      <marks>85</marks>
   </student>
   <student rollno="493">
      <firstname>Vaneet</firstname>
      <lastname>Gupta</lastname>
      <nickname>Vinni</nickname>
      <marks>95</marks>
   </student>
   <student rollno="593">
      <firstname>Jasvir</firstname>
      <lastname>Singh</lastname>
      <nickname>Jazz</nickname>
      <marks>90</marks>
   </student>
</class>
```

students.xsl

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html>
<body>
   <h3>Root Element. Xpath expression = "/*"</h3>
   <p><xsl:value-of select="name(/*)"/></p>
   <h3>Root Element. Xpath expression = "/class"</h3>
   <p> <xsl:value-of select="name(/class)"/></p>
   <h3>Details of each Students. Xpath expression = "/class/*"</h3>
   <table border="1">
      <tr bgcolor="#9acd32">
         <th>Roll No</th>
         <th>First Name</th>
         <th>Last Name</th>
         <th>Nick Name</th>
         <th>Marks</th>
      </tr>
      <xsl:for-each select="/class/*">
      <tr>
         <td>
            <xsl:value-of select="@rollno"/>
         </td>
         <td><xsl:value-of select="firstname"/></td>
         <td><xsl:value-of select="lastname"/></td>
         <td><xsl:value-of select="nickname"/></td>
         <td><xsl:value-of select="marks"/></td>
      </tr>
      </xsl:for-each>
   </table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

Verify the output



XPath Element Node

There are multiple ways to get and handle elements.

/class/* — select all element under root node.

```
<xsl:for-each select="/class/*">
```

/class/student — select all student element under root node.

```
<xsl:for-each select="/class/student">
```

//student — select all student elements in the document.

```
<xsl:for-each select="//student">
```

Example

In this example, we've created a sample XML document **students.xml** and its stylesheet document **students.xsl** which uses the XPath expressions.

Following is the sample XML used.

students.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="students.xsl"?>
<class>
   <student rollno="393">
      <firstname>Dinkar</firstname>
      <lastname>Kad</lastname>
      <nickname>Dinkar</nickname>
      <marks>85</marks>
   </student>
   <student rollno="493">
      <firstname>Vaneet</firstname>
      <lastname>Gupta</lastname>
      <nickname>Vinni</nickname>
      <marks>95</marks>
   </student>
   <student rollno="593">
      <firstname>Jasvir</firstname>
      <lastname>Singh</lastname>
      <nickname>Jazz</nickname>
      <marks>90</marks>
   </student>
</class>
```

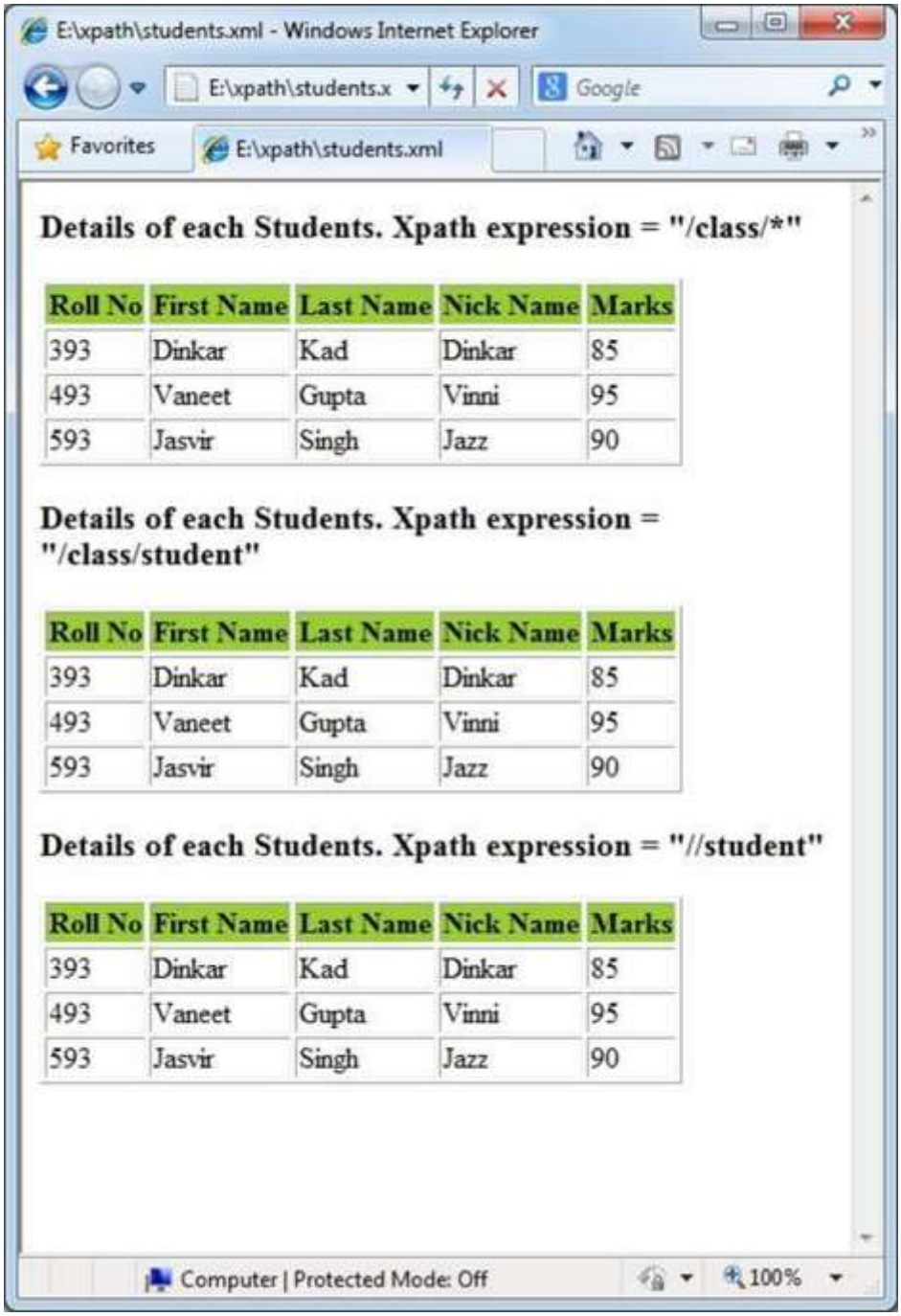
students.xsl

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html>
<body>
   <h3>Details of each Students. Xpath expression = "/class/*"</h3>
```

```
<table border="1">
   <tr bgcolor="#9acd32">
      <th>Roll No</th>
      <th>First Name</th>
      <th>Last Name</th>
      <th>Nick Name</th>
      <th>Marks</th>
   </tr>
   <xsl:for-each select="/class/*">
   <tr>
      <td>
         <xsl:value-of select="@rollno"/>
      </td>
      <td><xsl:value-of select="firstname"/></td>
      <td><xsl:value-of select="lastname"/></td>
      <td><xsl:value-of select="nickname"/></td>
      <td><xsl:value-of select="marks"/></td>
   </tr>
   </xsl:for-each>
</table>
<h3>Details of each Students. Xpath expression = "/class/student"</h3>
<table border="1">
   <tr bgcolor="#9acd32">
      <th>Roll No</th>
      <th>First Name</th>
      <th>Last Name</th>
      <th>Nick Name</th>
      <th>Marks</th>
   </tr>
   <xsl:for-each select="/class/student">
   <tr>
      <td>
         <xsl:value-of select="@rollno"/>
      </td>
      <td><xsl:value-of select="firstname"/></td>
      <td><xsl:value-of select="lastname"/></td>
      <td><xsl:value-of select="nickname"/></td>
      <td><xsl:value-of select="marks"/></td>
```

```
      </tr>
      </xsl:for-each>
   </table>
   <h3>Details of each Students. Xpath expression = "//student"</h3>
   <table border="1">
      <tr bgcolor="#9acd32">
         <th>Roll No</th>
         <th>First Name</th>
         <th>Last Name</th>
         <th>Nick Name</th>
         <th>Marks</th>
      </tr>
      <xsl:for-each select="//student">
      <tr>
         <td>
            <xsl:value-of select="@rollno"/>
         </td>
         <td><xsl:value-of select="firstname"/></td>
         <td><xsl:value-of select="lastname"/></td>
         <td><xsl:value-of select="nickname"/></td>
         <td><xsl:value-of select="marks"/></td>
      </tr>
      </xsl:for-each>
   </table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

Verify the output



XPath Text Node

Text can be easily retrieved and checked by using the name of the element.

name — get the text value of node "name".

```
<td><xsl:value-of select="firstname"/></td>
```

Text can be used to compared using operators.

marks > 85 — get the text value of node "marks" and compare with a value.

```
<xsl:if test="marks > 85">
```

Example

In this example, we've created a sample XML document **students.xml** and its stylesheet document **students.xsl** which uses the XPath expressions.

Following is the sample XML used.

students.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="students.xsl"?>
<class>
   <student rollno="393">
      <firstname>Dinkar</firstname>
      <lastname>Kad</lastname>
      <nickname>Dinkar</nickname>
      <marks>85</marks>
   </student>
   <student rollno="493">
      <firstname>Vaneet</firstname>
      <lastname>Gupta</lastname>
      <nickname>Vinni</nickname>
      <marks>95</marks>
   </student>
   <student rollno="593">
      <firstname>Jasvir</firstname>
      <lastname>Singh</lastname>
      <nickname>Jazz</nickname>
      <marks>90</marks>
   </student>
</class>
```

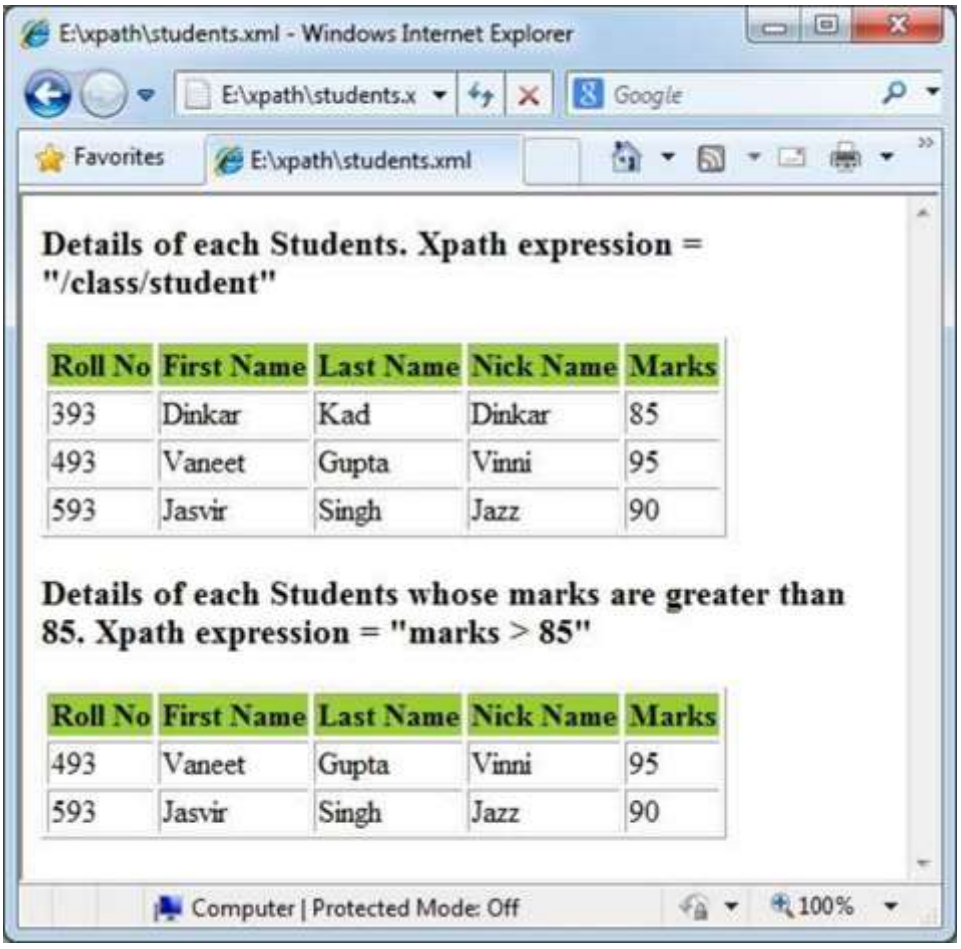
students.xsl

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

```
<xsl:template match="/">
<html>
<body>
   <h3>Details of each Students. Xpath expression = "/class/student"</h3>
   <table border="1">
      <tr bgcolor="#9acd32">
         <th>Roll No</th>
         <th>First Name</th>
         <th>Last Name</th>
         <th>Nick Name</th>
         <th>Marks</th>
      </tr>
      <xsl:for-each select="/class/student">
      <tr>
         <td>
            <xsl:value-of select="@rollno"/>
         </td>
         <td><xsl:value-of select="firstname"/></td>
         <td><xsl:value-of select="lastname"/></td>
         <td><xsl:value-of select="nickname"/></td>
         <td><xsl:value-of select="marks"/></td>
      </tr>
      </xsl:for-each>
   </table>
   <h3>Details of each Students whose marks are greater than 85. Xpath
expression = "marks > 85"</h3>
   <table border="1">
      <tr bgcolor="#9acd32">
         <th>Roll No</th>
         <th>First Name</th>
         <th>Last Name</th>
         <th>Nick Name</th>
         <th>Marks</th>
      </tr>
      <xsl:for-each select="//student">
         <xsl:if test="marks > 85">
         <tr>
            <td>
```

```
            <xsl:value-of select="@rollno"/>
         </td>
         <td><xsl:value-of select="firstname"/></td>
         <td><xsl:value-of select="lastname"/></td>
         <td><xsl:value-of select="nickname"/></td>
         <td><xsl:value-of select="marks"/></td>
      </tr>
    </xsl:if>
  </xsl:for-each>
 </table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

Verify the output



XPath Attribute Node

This attribute can be easily retrieved and checked by using the **@attribute-name** of the element.

@name - get the value of attribute "name".

```
<td><xsl:value-of select="@rollno"/></td>
```

Attribute can be used to compared using operators.

@rollno = 493 - get the text value of attribute "rollno" and compare with a value.

```
<xsl:if test="@rollno = 493">
```

Example

In this example, we've created a sample XML document **students.xml** and its stylesheet document **students.xsl** which uses the XPath expressions.

Following is the sample XML used.

students.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="students.xsl"?>
<class>
   <student rollno="393">
      <firstname>Dinkar</firstname>
      <lastname>Kad</lastname>
      <nickname>Dinkar</nickname>
      <marks>85</marks>
   </student>
   <student rollno="493">
      <firstname>Vaneet</firstname>
      <lastname>Gupta</lastname>
      <nickname>Vinni</nickname>
      <marks>95</marks>
   </student>
   <student rollno="593">
      <firstname>Jasvir</firstname>
      <lastname>Singh</lastname>
      <nickname>Jazz</nickname>
```

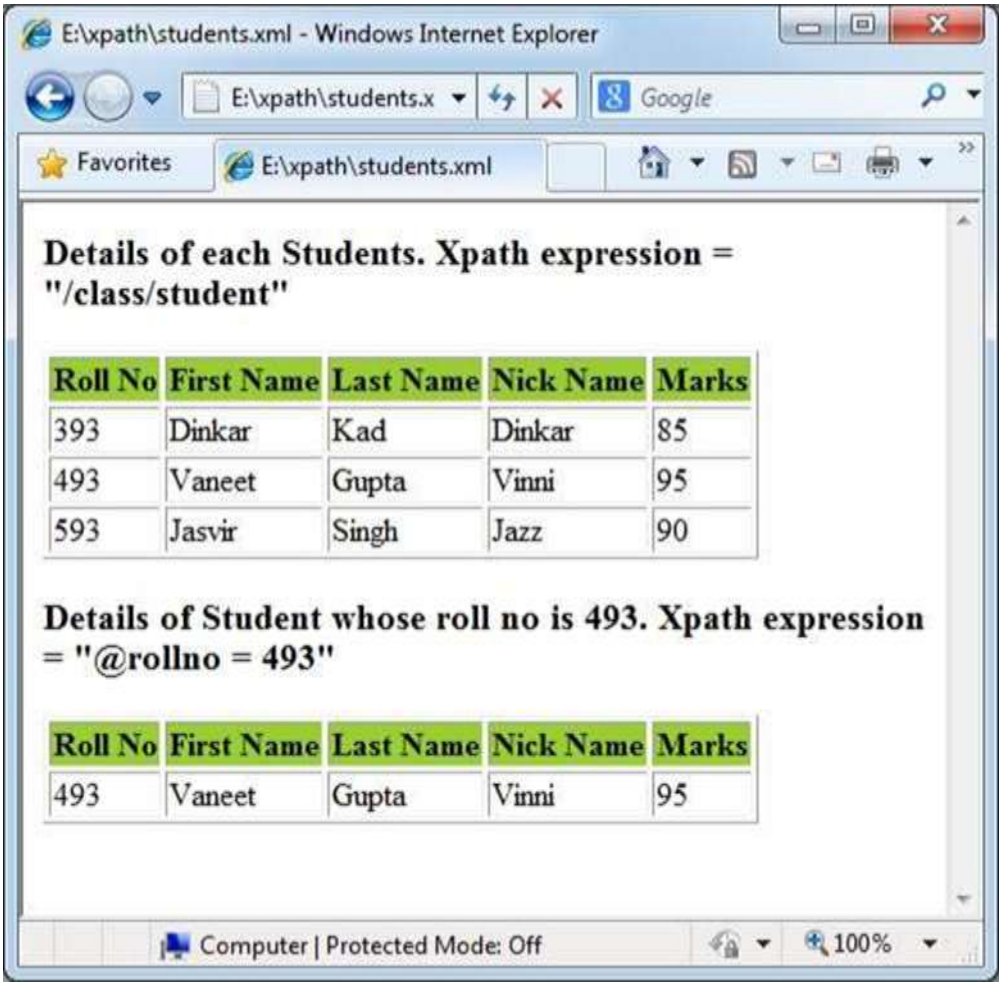
```
      <marks>90</marks>
   </student>
</class>
```

students.xsl

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html>
<body>
   <h3>Details of each Students. Xpath expression = "/class/student"</h3>
   <table border="1">
      <tr bgcolor="#9acd32">
         <th>Roll No</th>
         <th>First Name</th>
         <th>Last Name</th>
         <th>Nick Name</th>
         <th>Marks</th>
      </tr>
      <xsl:for-each select="/class/student">
      <tr>
         <td>
            <xsl:value-of select="@rollno"/>
         </td>
         <td><xsl:value-of select="firstname"/></td>
         <td><xsl:value-of select="lastname"/></td>
         <td><xsl:value-of select="nickname"/></td>
         <td><xsl:value-of select="marks"/></td>
      </tr>
      </xsl:for-each>
   </table>
   <h3>Details of Student whose roll no is 493. Xpath expression = "@rollno =
493"</h3>
   <table border="1">
      <tr bgcolor="#9acd32">
         <th>Roll No</th>
```

```
         <th>First Name</th>
         <th>Last Name</th>
         <th>Nick Name</th>
         <th>Marks</th>
      </tr>
      <xsl:for-each select="//student">
         <xsl:if test="@rollno = 493">
         <tr>
            <td>
               <xsl:value-of select="@rollno"/>
            </td>
            <td><xsl:value-of select="firstname"/></td>
            <td><xsl:value-of select="lastname"/></td>
            <td><xsl:value-of select="nickname"/></td>
            <td><xsl:value-of select="marks"/></td>
         </tr>
        </xsl:if>
      </xsl:for-each>
   </table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

Verify the output



XPath Comment Node

Comment can be easily retrieved of the element.

comment() - get the value of attribute "name".

```
<xsl:value-of select="/class/student/preceding-sibling::comment()"/>
```

Example

In this example, we've created a sample XML document **students.xml** and its stylesheet document **students.xsl** which uses the XPath expressions.

Following is the sample XML used.

students.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="students.xsl"?>
<class>
```

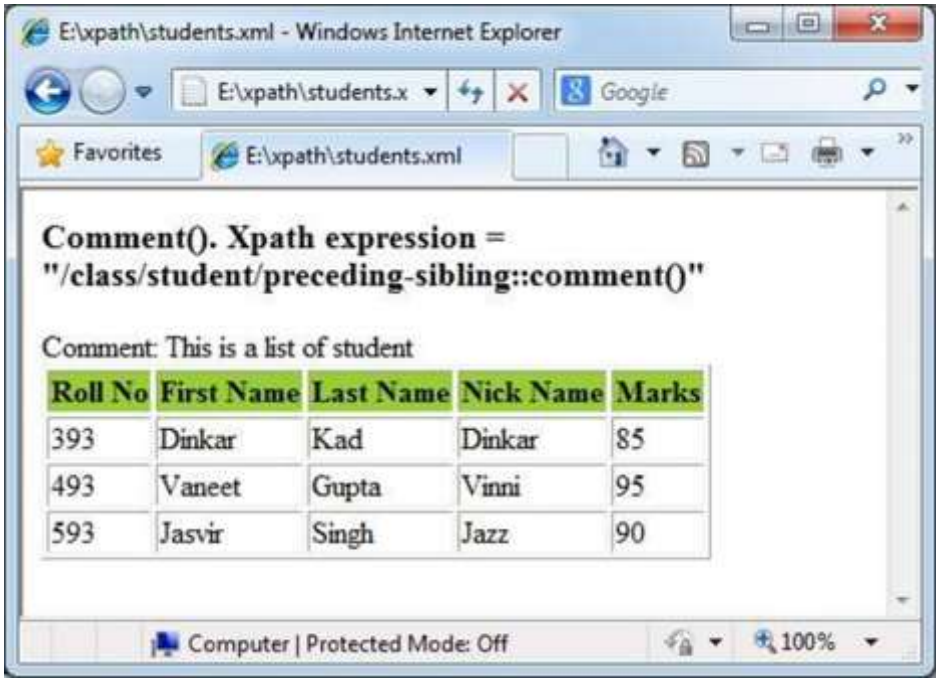
```
   <!-- Comment: This is a list of student -->
   <student rollno="393">
      <firstname>Dinkar</firstname>
      <lastname>Kad</lastname>
      <nickname>Dinkar</nickname>
      <marks>85</marks>
   </student>
   <student rollno="493">
      <firstname>Vaneet</firstname>
      <lastname>Gupta</lastname>
      <nickname>Vinni</nickname>
      <marks>95</marks>
   </student>
   <student rollno="593">
      <firstname>Jasvir</firstname>
      <lastname>Singh</lastname>
      <nickname>Jazz</nickname>
      <marks>90</marks>
   </student>
</class>
```

students.xsl

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html>
<body>
   <h3>Comment(). Xpath expression = "/class/student/preceding-
sibling::comment()"</h3>
   <xsl:value-of select="/class/student/preceding-sibling::comment()"/>

     <table border="1">
      <tr bgcolor="#9acd32">
         <th>Roll No</th>
         <th>First Name</th>
         <th>Last Name</th>
```

```
         <th>Nick Name</th>
         <th>Marks</th>
      </tr>
      <xsl:for-each select="/class/student">
      <tr>
         <td>
            <xsl:value-of select="@rollno"/>
         </td>
         <td><xsl:value-of select="firstname"/></td>
         <td><xsl:value-of select="lastname"/></td>
         <td><xsl:value-of select="nickname"/></td>
         <td><xsl:value-of select="marks"/></td>
      </tr>
      </xsl:for-each>
   </table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

Verify the output



4. XPath — Absolute Path

Location path specifies the location of node in XML document. This path can be absolute or relative. If location path starts with root node or with '/' then it is an absolute path. Following are few of the example locating the elements using absolute path.

/class/student - select student nodes within class root node.

```
<xsl:for-each select="/class/student">
```

/class/student/firstname - select firstname of a student node within class root node.

```
<p><xsl:value-of select="/class/student/firstname"/></p>
```

Example

In this example, we've created a sample XML document **students.xml** and its stylesheet document **students.xsl** which uses XPath expressions.

Following is the sample XML used.

students.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="students.xsl"?>
<class>
   <student rollno="393">
      <firstname>Dinkar</firstname>
      <lastname>Kad</lastname>
      <nickname>Dinkar</nickname>
      <marks>85</marks>
   </student>
   <student rollno="493">
      <firstname>Vaneet</firstname>
      <lastname>Gupta</lastname>
      <nickname>Vinni</nickname>
      <marks>95</marks>
   </student>
   <student rollno="593">
      <firstname>Jasvir</firstname>
      <lastname>Singh</lastname>
```

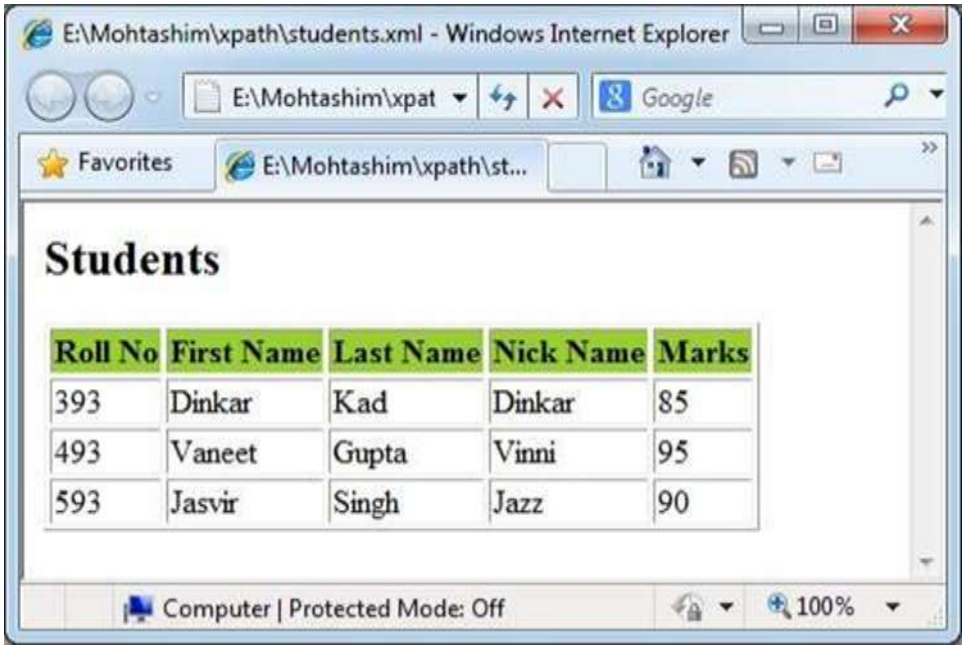
```
      <nickname>Jazz</nickname>
      <marks>90</marks>
   </student>
</class>
```

students.xsl

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/" >
<html>
<body>
   <h3>Details of each Students. </h3>
   <table border="1">
      <tr bgcolor="#9acd32">
         <th>Roll No</th>
         <th>First Name</th>
         <th>Last Name</th>
         <th>Nick Name</th>
         <th>Marks</th>
      </tr>
      <tr>
         <td>
            <xsl:value-of select="/class/student[1]/@rollno"/>
         </td>
         <td><xsl:value-of select="/class/student[1]/firstname"/></td>
         <td><xsl:value-of select="/class/student[1]/lastname"/></td>
         <td><xsl:value-of select="/class/student[1]/nickname"/></td>
         <td><xsl:value-of select="/class/student[1]/marks"/></td>
      </tr>
       <tr>
         <td>
            <xsl:value-of select="/class/student/@rollno"/>
         </td>
         <td><xsl:value-of select="/class/student[2]/firstname"/></td>
         <td><xsl:value-of select="/class/student[2]/lastname"/></td>
```

```
         <td><xsl:value-of select="/class/student[2]/nickname"/></td>

         <td><xsl:value-of select="/class/student[2]/marks"/></td>
      </tr>
       <tr>
         <td>
            <xsl:value-of select="/class/student[3]/@rollno"/>
         </td>
         <td><xsl:value-of select="/class/student[3]/firstname"/></td>
         <td><xsl:value-of select="/class/student[3]/lastname"/></td>
         <td><xsl:value-of select="/class/student[3]/nickname"/></td>
         <td><xsl:value-of select="/class/student[3]/marks"/></td>
      </tr>

   </table>

</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

Verify the output



5. XPath — Relative Path

Location path specifies the location of node in XML document. This path can be absolute or relative. If location path starts with the node that we've selected, then it is a relative path.

Following are a few examples locating the elements using relative path.

firstname — select firstname related to student nodes .

```
<p><xsl:value-of select="firstname"/></p>
```

Example

In this example, we've created a sample XML document **students.xml** and its stylesheet document **students.xsl** which uses the XPath expressions.

Following is the sample XML used.

students.XML

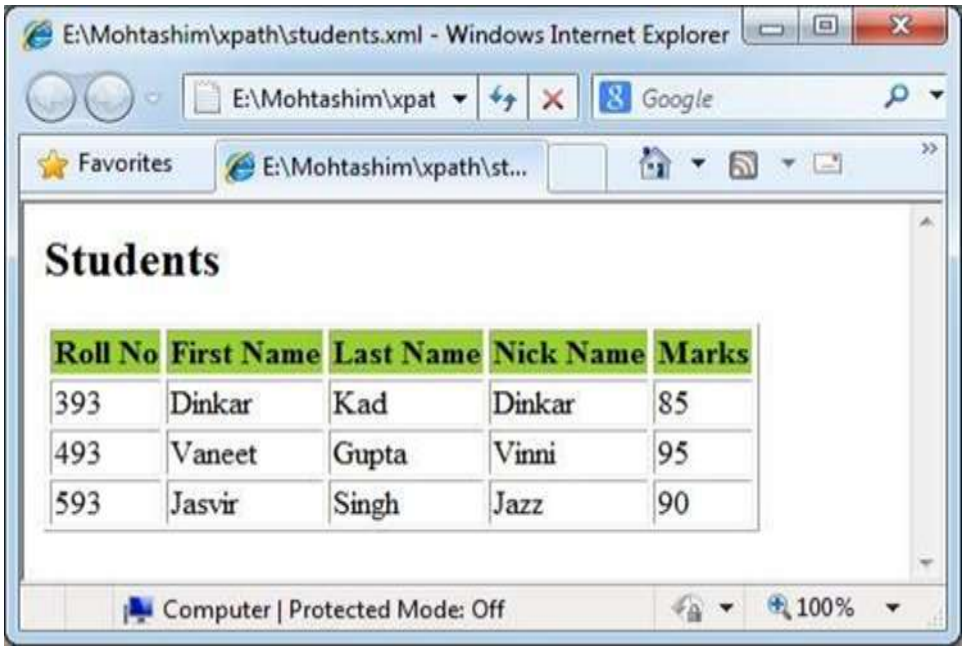
```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="students.xsl"?>
<class>
   <student rollno="393">
      <firstname>Dinkar</firstname>
      <lastname>Kad</lastname>
      <nickname>Dinkar</nickname>
      <marks>85</marks>
   </student>
   <student rollno="493">
      <firstname>Vaneet</firstname>
      <lastname>Gupta</lastname>
      <nickname>Vinni</nickname>
      <marks>95</marks>
   </student>
   <student rollno="593">
      <firstname>Jasvir</firstname>
      <lastname>Singh</lastname>
      <nickname>Jazz</nickname>
      <marks>90</marks>
```

```
   </student>
</class>
```

students.xsl

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/" >
<html>
<body>
   <h3>Details of each Students. </h3>
   <table border="1">
      <tr bgcolor="#9acd32">
         <th>Roll No</th>
         <th>First Name</th>
         <th>Last Name</th>
         <th>Nick Name</th>
         <th>Marks</th>
      </tr>
      <xsl:for-each select="/class/student">
      <tr>
         <td>
            <xsl:value-of select="@rollno"/>
         </td>
         <td><xsl:value-of select="firstname"/></td>
         <td><xsl:value-of select="lastname"/></td>
         <td><xsl:value-of select="nickname"/></td>
         <td><xsl:value-of select="marks"/></td>
      </tr>
      </xsl:for-each>
   </table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

Verify the output



6. XPath — Axes

As location path defines the location of a node using absolute or relative path, axes are used to identify elements by their relationship like **parent**, **child**, **sibling**, etc. Axes are named so because they refer to axis on which elements are lying relative to an element.

Following is the list of various Axis values.

Axis	Description
ancestor	Represents the ancestors of the current node which include the parents up to the root node.
ancestor-or-self	Represents the current node and it's ancestors.
attribute	Represents the attributes of the current node.
child	Represents the children of the current node.
descendant	Represents the descendants of the current node. Descendants include the node's children upto the leaf node(no more children).
descendant-or-self	Represents the current node and it's descendants.
following	Represents all nodes that come after the current node.
following-sibling	Represents the following siblings of the context node. Siblings are at the same level as the current node and share it's parent.
namespace	Represents the namespace of the current node.
parent	Represents the parent of the current node.
preceding	Represents all nodes that come before the current node (i.e. before it's opening tag).
self	Represents the current node.

Following are a few examples on the use of axes.

firstname — select firstname related to student nodes.

```
<p><xsl:value-of select="firstname"/></p>
<xsl:value-of select="/class/student/preceding-sibling::comment()"/>
```

Example

In this example, we've created a sample XML document **students.xml** and its stylesheet document **students.xsl** which uses the XPath expressions.

Following is the sample XML used.

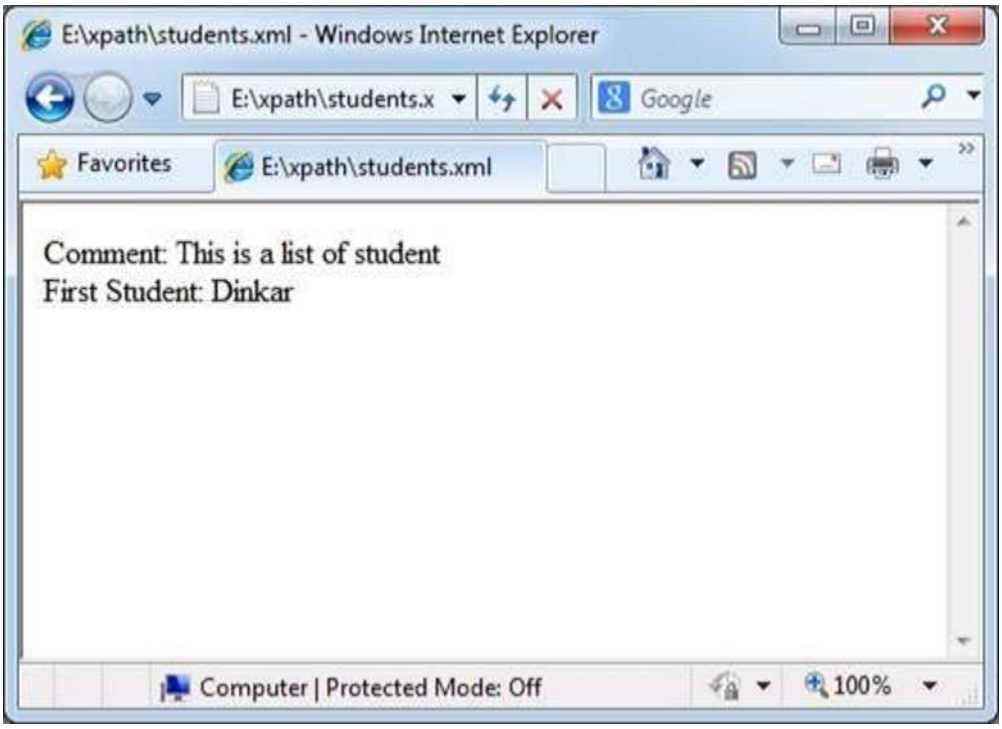
students.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="students.xsl"?>
<class>
   <!-- Comment: This is a list of student -->
   <student rollno="393">
      <firstname>Dinkar</firstname>
      <lastname>Kad</lastname>
      <nickname>Dinkar</nickname>
      <marks>85</marks>
   </student>
   <student rollno="493">
      <firstname>Vaneet</firstname>
      <lastname>Gupta</lastname>
      <nickname>Vinni</nickname>
      <marks>95</marks>
   </student>
   <student rollno="593">
      <firstname>Jasvir</firstname>
      <lastname>Singh</lastname>
      <nickname>Jazz</nickname>
      <marks>90</marks>
   </student>
</class>
```

students.xsl

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/" >
<html>
<body>
  <xsl:value-of select="/class/student/preceding-sibling::comment()"/>
  <br/>
  <xsl:text>First Student: </xsl:text>
  <xsl:value-of select="/class/student/child::firstname" />
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

Verify the output



7. XPath — Operators

In this chapter, we'll see XPath operators and functions in details covering commonly used XPath **defines** and **handles**. XPath defines Operators and functions on Nodes, String, Number and Boolean types.

Following is the list we are going to discuss about.

S.N.	Operators/Functions & Description
1	**Comparison Operators** Comparison operators to compare values.
2	**Boolean Operators** Boolean operators to check **'and'**, **'or'** & **'not'** functionalities.
3	**Number Functions/Operators** Operators/Functions on numbers.
4	**String Functions** Various string functions.
5	**Node Functions/Operators** Various functions and operators acting on nodes.

XPath Comparison Operators

XPath defines following comparison operators to be used with the XPath expressions.

Operator	Description
=	is equals to
!=	is not equals to
<	is less than
>	is greater than
<=	is less than or equals to

>=	is greater than or equals to

Example

This example creates a table of <student> element with its attribute roll no and its child <firstname>,<lastname><nickname> and <marks> by iterating over each student. It checks marks to be greater than 90 and then prints the student(s) details.

students.xml

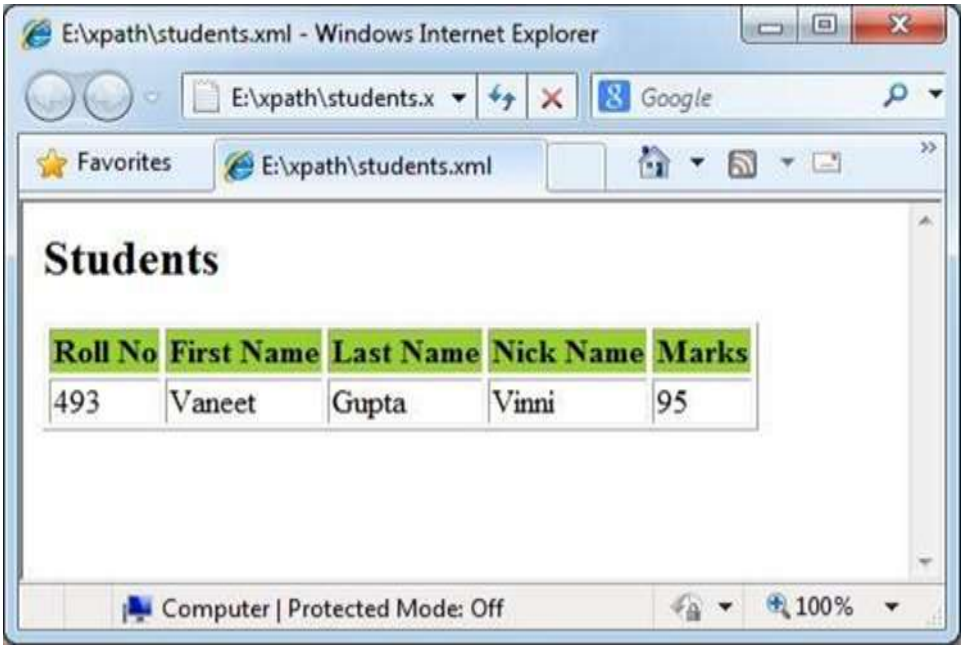
```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="students.xsl"?>
<class>
   <student rollno="393">
      <firstname>Dinkar</firstname>
      <lastname>Kad</lastname>
      <nickname>Dinkar</nickname>
      <marks>85</marks>
   </student>
   <student rollno="493">
      <firstname>Vaneet</firstname>
      <lastname>Gupta</lastname>
      <nickname>Vinni</nickname>
      <marks>95</marks>
   </student>
   <student rollno="593">
      <firstname>Jasvir</firstname>
      <lastname>Singh</lastname>
      <nickname>Jazz</nickname>
      <marks>90</marks>
   </student>
</class>
```

students.xsl

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
```

```
  <html>
  <body>
  <h2>Students</h2>
  <table border="1">
    <tr bgcolor="#9acd32">
       <th>Roll No</th>
      <th>First Name</th>
      <th>Last Name</th>
       <th>Nick Name</th>
       <th>Marks</th>
    </tr>
    <xsl:for-each select="class/student">
    <xsl:if test="marks > 90">
    <tr>
       <td>
            <xsl:value-of select="@rollno"/>
       </td>
      <td><xsl:value-of select="firstname"/></td>
      <td><xsl:value-of select="lastname"/></td>
       <td><xsl:value-of select="nickname"/></td>
       <td><xsl:value-of select="marks"/></td>
    </tr>
     </xsl:if>
    </xsl:for-each>
  </table>
  </body>
  </html>
</xsl:template>

</xsl:stylesheet>
```

Verify the output



XPath Boolean Operators

XPath defines the following Boolean operators to be used with the XPath expressions.

Operator	Description
and	both conditions to be satisfied
or	any one of the condition to be satisfied
not()	function to check condition not to be satisfied.

Example

This example creates a table of <student> element with its attribute roll no and its child <firstname>,<lastname><nickname> and <marks> by iterating over each student. It checks rollno to be either 393 or 493 and then prints the student(s) details.

students.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="students.xsl"?>
<class>
   <student rollno="393">
      <firstname>Dinkar</firstname>
```

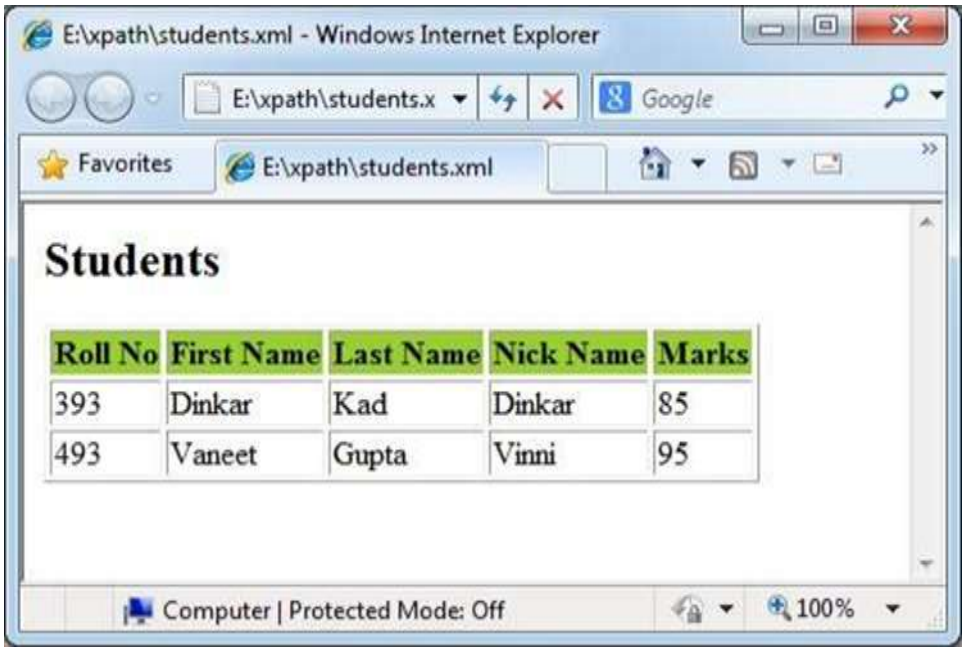
```
      <lastname>Kad</lastname>
      <nickname>Dinkar</nickname>
      <marks>85</marks>
   </student>
   <student rollno="493">
      <firstname>Vaneet</firstname>
      <lastname>Gupta</lastname>
      <nickname>Vinni</nickname>
      <marks>95</marks>
   </student>
   <student rollno="593">
      <firstname>Jasvir</firstname>
      <lastname>Singh</lastname>
      <nickname>Jazz</nickname>
      <marks>90</marks>
   </student>
</class>
```

students.xsl

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
  <h2>Students</h2>
  <table border="1">
    <tr bgcolor="#9acd32">
       <th>Roll No</th>
      <th>First Name</th>
      <th>Last Name</th>
       <th>Nick Name</th>
       <th>Marks</th>
    </tr>
    <xsl:for-each select="class/student[(@rollno = 393) or ((@rollno = 493))]">
```

```
    <tr>

       <td>
            <xsl:value-of select="@rollno"/>
       </td>
      <td><xsl:value-of select="firstname"/></td>
      <td><xsl:value-of select="lastname"/></td>
       <td><xsl:value-of select="nickname"/></td>
       <td><xsl:value-of select="marks"/></td>
    </tr>
     </xsl:for-each>
  </table>
  </body>
  </html>
</xsl:template>

</xsl:stylesheet>
```

Verify the output



XPath Number Operators / Functions

XPath defines the following operators on numbers to be used with the XPath expressions.

Operator	Description
+	used for addition operation
-	used for subtraction operation
*	used for multiplication operation
div	used for division operation
mod	used for modulo operation

XPath defines the following functions on numbers to be used with the XPath expressions.

Function	Description
ceiling()	returns the smallest integer larger than the value provided.
floor()	returns the largest integer smaller than the value provided.
round()	returns the rounded value to nearest integer.
sum()	returns the sum of two numbers.

Example

This example creates a table of <student> element with its attribute roll no and its child <firstname>,<lastname><nickname> and <marks> by iterating over each student. It calculates grades of the student and then prints the student(s) details.

students.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="students.xsl"?>
<class>
   <student rollno="393">
      <firstname>Dinkar</firstname>
      <lastname>Kad</lastname>
      <nickname>Dinkar</nickname>
      <marks>85</marks>
   </student>
```

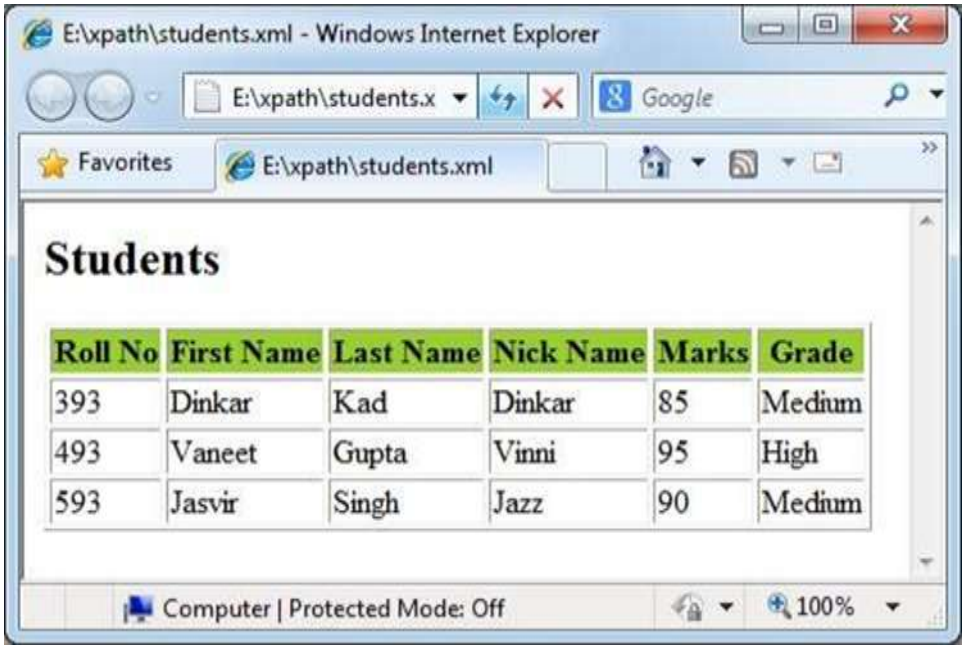
```
   <student rollno="493">
      <firstname>Vaneet</firstname>
      <lastname>Gupta</lastname>
      <nickname>Vinni</nickname>
      <marks>95</marks>
   </student>
   <student rollno="593">
      <firstname>Jasvir</firstname>
      <lastname>Singh</lastname>
      <nickname>Jazz</nickname>
      <marks>90</marks>
   </student>
</class>
```

students.xsl

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
  <h2>Students</h2>
  <table border="1">
    <tr bgcolor="#9acd32">
       <th>Roll No</th>
      <th>First Name</th>
      <th>Last Name</th>
       <th>Nick Name</th>
       <th>Marks</th>
       <th>Grade</th>
    </tr>
    <xsl:for-each select="class/student">
    <tr>
       <td>
            <xsl:value-of select="@rollno"/>
       </td>
```

```
      <td><xsl:value-of select="firstname"/></td>

      <td><xsl:value-of select="lastname"/></td>
       <td><xsl:value-of select="nickname"/></td>
       <td><xsl:value-of select="marks"/></td>
       <td>
        <xsl:choose>
           <xsl:when test="marks div 90 > 1">
              High
           </xsl:when>
           <xsl:when test="marks div 80 > 1">
              Medium
           </xsl:when>
           <xsl:otherwise>
              Low
           </xsl:otherwise>
      </xsl:choose>

       </td>
    </tr>
    </xsl:for-each>
  </table>
  </body>
  </html>
</xsl:template>

</xsl:stylesheet>
```

Verify the output



XPath String Functions

The following is a list of XPath String functions:

S.N.	Function & Description
1	**starts-with(string1, string2)** Returns true when first string starts with the second string.
2	**contains(string1, string2)** Returns true when the first string contains the second string.
3	**substring(string, offset, length?)** Returns a section of the string. The section starts at offset up to the length provided.
4	**substring-before(string1, string2)** Returns the part of string1 up before the first occurrence of string2.
5	**substring-after(string1, string2)** Returns the part of string1 after the first occurrence of string2.
6	**string-length(string)** Returns the length of string in terms of characters.

7	**normalize-space(string)** Trims the leading and trailing space from string.
8	**translate(string1, string2, string3)** Returns string1 after any matching characters in string2 have been replaced by the characters in string3.
9	**concat(string1, string2, ...)** Concatenates all strings.
10	**format-number(number1, string1, string2)** Returns a formatted version of number1 after applying string1 as a format string. string2 is an optional locale string.

Example

This example creates a table of <student> element with their names and length of names, by iterating over each student. It calculates length of the student name after concatenating firstname and lastname and then prints the student(s) details.

students.xml

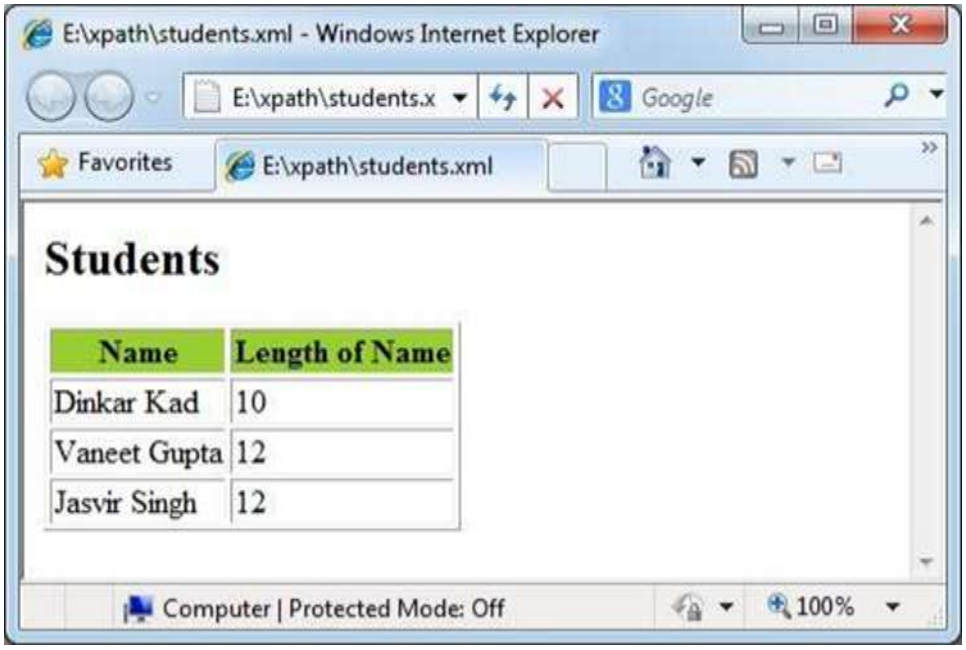
```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="students.xsl"?>
<class>
   <student rollno="393">
      <firstname>Dinkar</firstname>
      <lastname>Kad</lastname>
      <nickname>Dinkar</nickname>
      <marks>85</marks>
   </student>
   <student rollno="493">
      <firstname>Vaneet</firstname>
      <lastname>Gupta</lastname>
      <nickname>Vinni</nickname>
      <marks>95</marks>
   </student>
   <student rollno="593">
      <firstname>Jasvir</firstname>
```

```
      <lastname>Singh</lastname>
      <nickname>Jazz</nickname>
      <marks>90</marks>
   </student>
</class>
```

students.xsl

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
  <h2>Students</h2>
  <table border="1">
    <tr bgcolor="#9acd32">
      <th>Name</th>
       <th>Length of Name</th>
    </tr>
    <xsl:for-each select="class/student">
    <tr>
      <td><xsl:value-of select="concat(firstname,' ',lastname)"/></td>
      <td><xsl:value-of select="string-length(concat(firstname,' ',lastname))"/></td>

    </tr>
    </xsl:for-each>
  </table>
  </body>
  </html>
</xsl:template>

</xsl:stylesheet>
```

Verify the output



XPath Node Functions

XPath defines the following operators on nodes to be used with the XPath expressions.

Operator	Description
/	used to select node under a specific node.
//	used to select node from root node
[...]	used to check node value
|	used for union of two node sets

XPath defines the following functions on nodes to be used with the XPath expressions.

Function	Description
comment()	selects nodes which are comments.
node()	selects all kinds of nodes.
processing-instruction()	selects nodes which are processing instruction.

text()	selects a text node.
name()	provides the name of the node.
position()	provides the position of the node.
last()	selects the last node relative to current node;

Example

This example creates a table of <student> element with their details, by iterating over each student. It calculates the position of the student node then prints the student(s) details along with serial no.

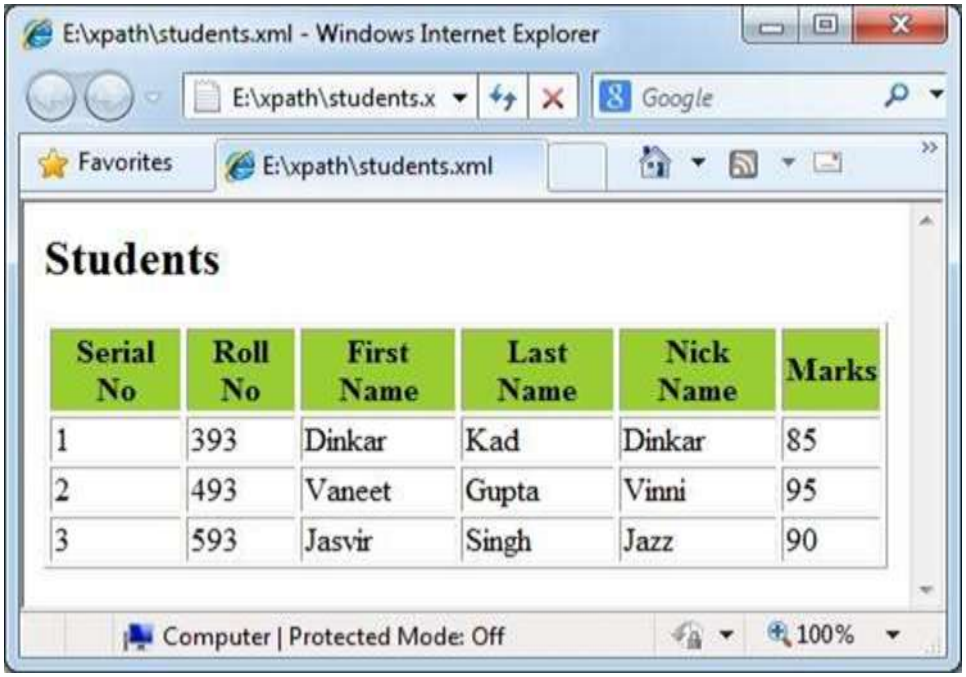
students.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="students.xsl"?>
<class>
   <student rollno="393">
      <firstname>Dinkar</firstname>
      <lastname>Kad</lastname>
      <nickname>Dinkar</nickname>
      <marks>85</marks>
   </student>
   <student rollno="493">
      <firstname>Vaneet</firstname>
      <lastname>Gupta</lastname>
      <nickname>Vinni</nickname>
      <marks>95</marks>
   </student>
   <student rollno="593">
      <firstname>Jasvir</firstname>
      <lastname>Singh</lastname>
      <nickname>Jazz</nickname>
      <marks>90</marks>
   </student>
</class>
```

students.xsl

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
  <h2>Students</h2>
  <table border="1">
    <tr bgcolor="#9acd32">
        <th>Serial No</th>
        <th>Roll No</th>
      <th>First Name</th>
      <th>Last Name</th>
        <th>Nick Name</th>
        <th>Marks</th>
    </tr>
    <xsl:for-each select="class/student">
    <tr>
        <td><xsl:value-of select="position()"/></td>
        <td><xsl:value-of select="@rollno"/></td>
      <td><xsl:value-of select="firstname"/></td>
      <td><xsl:value-of select="lastname"/></td>
        <td><xsl:value-of select="nickname"/></td>
        <td><xsl:value-of select="marks"/></td>
    </tr>
    </xsl:for-each>
  </table>
  </body>
  </html>
</xsl:template>
</xsl:stylesheet>
```

Verify the output


E:\xpath\students.xml - Windows Internet Explorer
E:\xpath\students.x
Google
Favorites
E:\xpath\students.xml
Students
Serial No
Roll No
First Name
Last Name
Nick Name
Marks
1
393
Dinkar
Kad
Dinkar
85
2
493
Vaneet
Gupta
Vinni
95
3
593
Jasvir
Singh
Jazz
90
Computer | Protected Mode: Off
100%

8. XPath — Wildcard

XPath defines the following wildcards on nodes to be used with the XPath expressions.

Wildcard	Description
*	used to match any node.
.	used to match the current node in context.
@*	used to match any attribute
node()	used to match node of any type

Example

This example creates a table of <student> element with their details, by iterating over each student.

students.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="students.xsl"?>
<class>
   <student rollno="393">
      <firstname>Dinkar</firstname>
      <lastname>Kad</lastname>
      <nickname>Dinkar</nickname>
      <marks>85</marks>
   </student>
   <student rollno="493">
      <firstname>Vaneet</firstname>
      <lastname>Gupta</lastname>
      <nickname>Vinni</nickname>
      <marks>95</marks>
   </student>
   <student rollno="593">
      <firstname>Jasvir</firstname>
      <lastname>Singh</lastname>
```

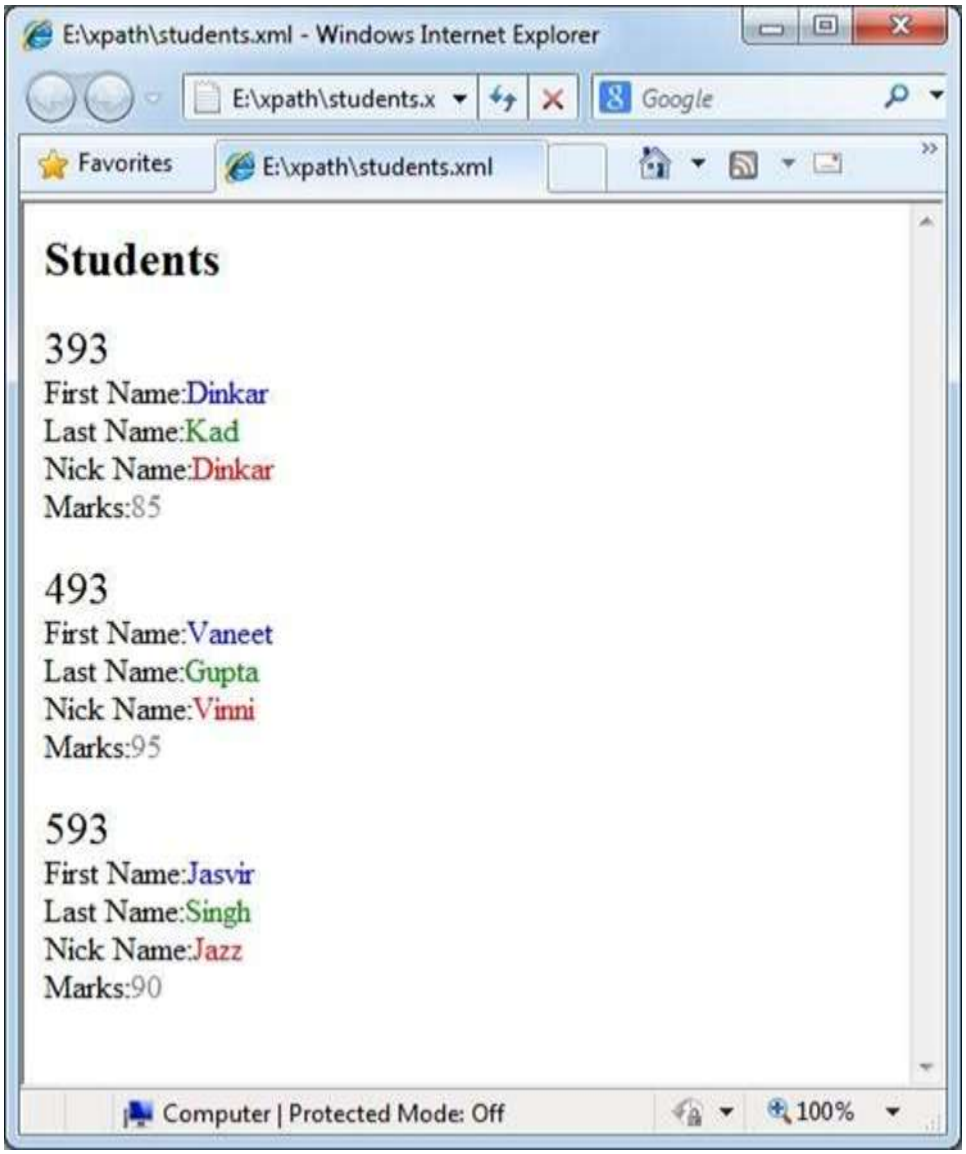
```
      <nickname>Jazz</nickname>
      <marks>90</marks>
   </student>
</class>
```

students.xsl

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
  <html>
  <body>
  <h2>Students</h2>
  <xsl:apply-templates select="class/*" />
  </body>
  </html>
</xsl:template>

<xsl:template match="class/*">
  <xsl:apply-templates select="@rollno" />
  <xsl:apply-templates select="firstname" />
  <xsl:apply-templates select="lastname" />
  <xsl:apply-templates select="nickname" />
  <xsl:apply-templates select="marks" />
<br />
</xsl:template>

<xsl:template match="@rollno">
<span style="font-size=22px;">
<xsl:value-of select="." />
</span>
<br />
</xsl:template>

<xsl:template match="firstname">
First Name:<span style="color:blue;">
```

```
<xsl:value-of select="." />
</span>
<br />
</xsl:template>

<xsl:template match="lastname">
Last Name:<span style="color:green;">
<xsl:value-of select="." />
</span>
<br />
</xsl:template>

<xsl:template match="nickname">
Nick Name:<span style="color:red;">
<xsl:value-of select="." />
</span>
<br />
</xsl:template>

<xsl:template match="marks">
Marks:<span style="color:gray;">
<xsl:value-of select="." />
</span>
<br />
</xsl:template>

</xsl:stylesheet>
```

Verify the output



9. XPath — Predicate

Predicate refers to the XPath expression written in square brackets. It refers to restrict the selected nodes in a node set for some condition. For example,

Predicate	Description
/class/student[1]	Select first student element which is child of the class element.
/class/student[last()]	Select last student element which is child of the class element.
/class/student[@rolllno=493]	Select student element with roll no 493.
/class/student[marks>85]	Select student element with marks > 85.

Example

This example creates a table of <student> element with their details, by iterating over each student. It calculates the position of the student node and then prints the student(s) details along with serial no.

students.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="students.xsl"?>
<class>
   <student rollno="393">
      <firstname>Dinkar</firstname>
      <lastname>Kad</lastname>
      <nickname>Dinkar</nickname>
      <marks>85</marks>
   </student>
   <student rollno="493">
      <firstname>Vaneet</firstname>
      <lastname>Gupta</lastname>
      <nickname>Vinni</nickname>
      <marks>95</marks>
   </student>
```

```
   <student rollno="593">
      <firstname>Jasvir</firstname>
      <lastname>Singh</lastname>
      <nickname>Jazz</nickname>
      <marks>90</marks>
   </student>
</class>
```

students.xsl

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
  <h2>Students</h2>
  <table border="1">
    <tr bgcolor="#9acd32">
       <th>Roll No</th>
      <th>First Name</th>
      <th>Last Name</th>
       <th>Nick Name</th>
       <th>Marks</th>
    </tr>
    <xsl:for-each select="/class/student[1]">
    <tr>
       <td><xsl:value-of select="@rollno"/></td>
      <td><xsl:value-of select="firstname"/></td>
      <td><xsl:value-of select="lastname"/></td>
       <td><xsl:value-of select="nickname"/></td>
       <td><xsl:value-of select="marks"/></td>
    </tr>
    </xsl:for-each>
     <xsl:for-each select="/class/student[last()]">
    <tr>
       <td><xsl:value-of select="@rollno"/></td>
```

```
      <td><xsl:value-of select="firstname"/></td>
      <td><xsl:value-of select="lastname"/></td>
       <td><xsl:value-of select="nickname"/></td>
       <td><xsl:value-of select="marks"/></td>
    </tr>
    </xsl:for-each>

     <xsl:for-each select="/class/student[@rollno=493]">
    <tr>
       <td><xsl:value-of select="@rollno"/></td>
      <td><xsl:value-of select="firstname"/></td>
      <td><xsl:value-of select="lastname"/></td>
       <td><xsl:value-of select="nickname"/></td>
       <td><xsl:value-of select="marks"/></td>
    </tr>
    </xsl:for-each>

     <xsl:for-each select="/class/student[marks > 85]">
    <tr>
       <td><xsl:value-of select="@rollno"/></td>
      <td><xsl:value-of select="firstname"/></td>
      <td><xsl:value-of select="lastname"/></td>
       <td><xsl:value-of select="nickname"/></td>
       <td><xsl:value-of select="marks"/></td>
    </tr>
    </xsl:for-each>


  </table>

  </body>
  </html>
</xsl:template>
</xsl:stylesheet>
```

Verify the output