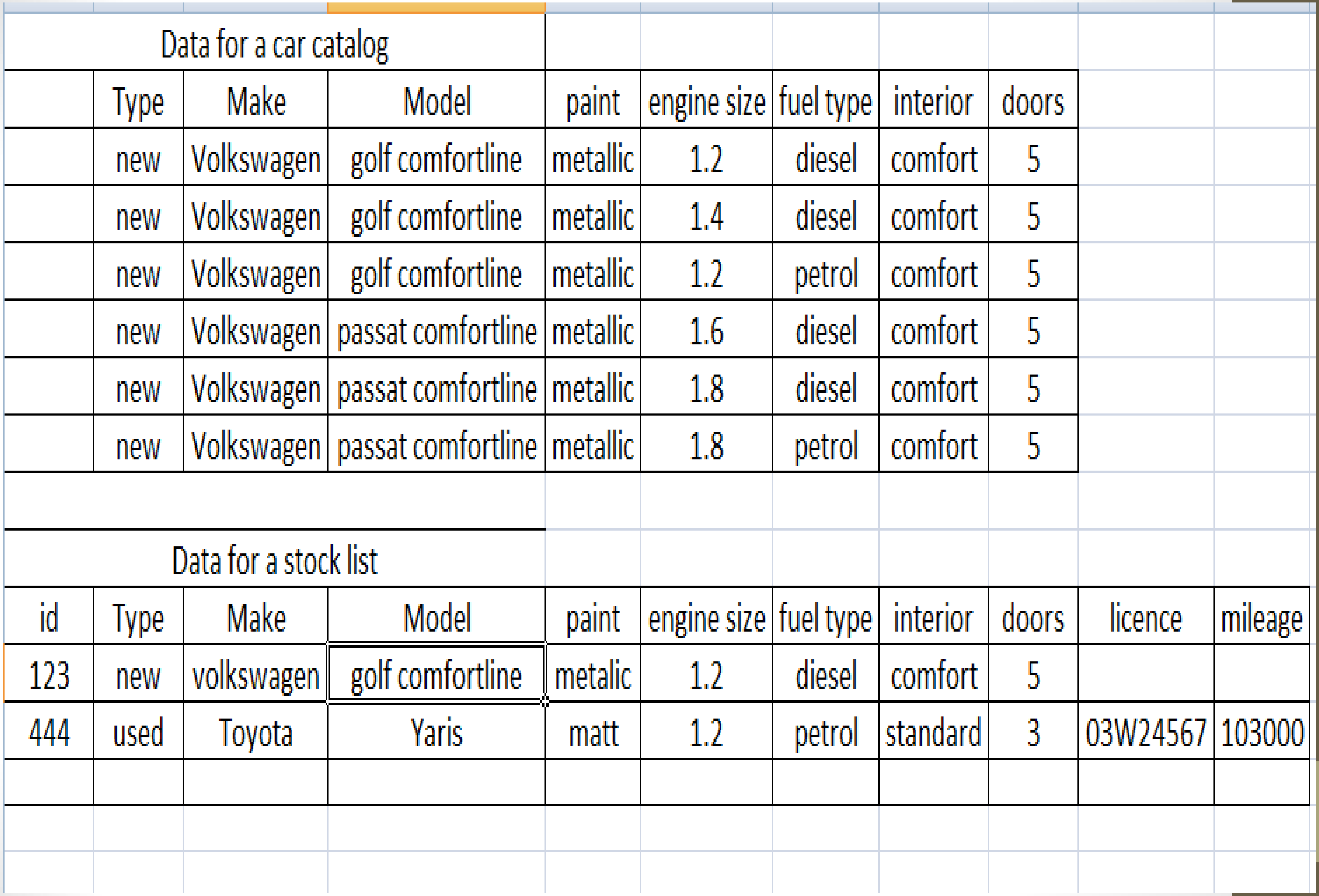


Data for a car catalog

	Type	Make	Model	paint	engine size	fuel type	interior	doors
	new	Volkswagen	golf comfortline	metallic	1.2	diesel	comfort	5
	new	Volkswagen	golf comfortline	metallic	1.4	diesel	comfort	5
	new	Volkswagen	golf comfortline	metallic	1.2	petrol	comfort	5
	new	Volkswagen	passat comfortline	metallic	1.6	diesel	comfort	5
	new	Volkswagen	passat comfortline	metallic	1.8	diesel	comfort	5
	new	Volkswagen	passat comfortline	metallic	1.8	petrol	comfort	5

Data for a stock list

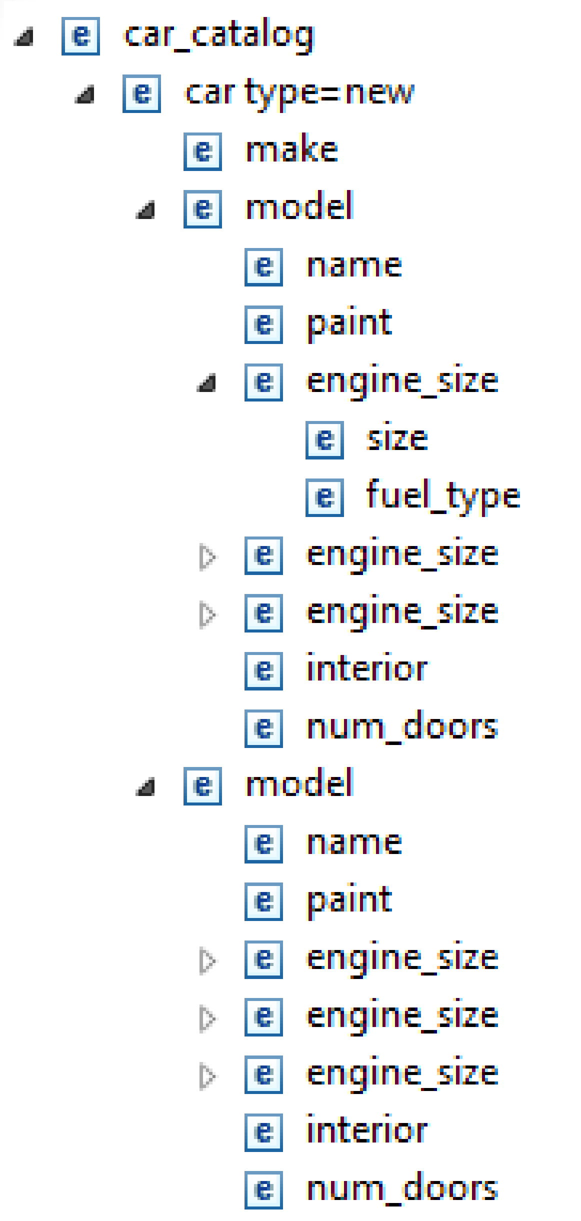
id	Type	Make	Model	paint	engine size	fuel type	interior	doors	licence	mileage
123	new	volkswagen	golf comfortline	metalic	1.2	diesel	comfort	5		
444	used	Toyota	Yaris	matt	1.2	petrol	standard	3	03W24567	103000

XSLT – eXtensible Style Language for Transformations

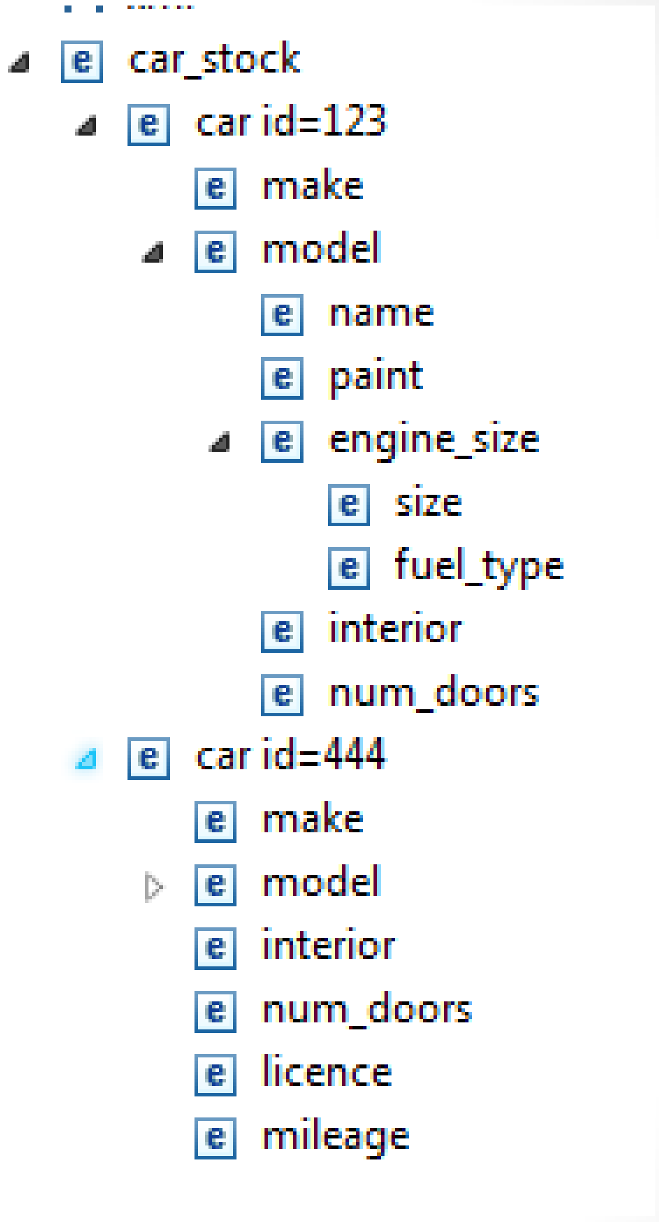
The car data can be stored in more than one way, it depends on the business use of the data, your interpretation of the description and what the client wants.
The car data shown looks similar but depending on the use we can structure it differently.

```
<car_catalog>
  <car type="new">
    <make>Volkswagen</make>
    <model>
      <name>golf comfortline</name>
      <paint>metalic</paint>
      <engine_size>
        <size>1.2</size>
        <fuel_type>diesel</fuel_type>
      </engine_size>
      <engine_size>
        <size>1.4</size>
        <fuel_type>diesel</fuel_type>
      </engine_size>
      <engine_size>
        <size>1.2</size>
        <fuel_type>petrol</fuel_type>
      </engine_size>
      <interior>comfort</interior>
      <num_doors>5</num_doors>
    </model>
    <model>
      <name>passat comfortline</name>
      <paint>metalic</paint>
      <engine_size>
        <size>1.6</size>
        <fuel_type>diesel</fuel_type>
      </engine_size>
      <engine_size>
        <size>1.8</size>
        <fuel_type>diesel</fuel_type>
      </engine_size>
      <engine_size>
        <size>1.8</size>
        <fuel_type>petrol</fuel_type>
      </engine_size>
      <interior>comfort</interior>
      <num_doors>5</num_doors>
    </model>
  </car>
</car_catalog>
```

```
<car_stock>
  <car id="123" type="new">
    <make>Volkswagen</make>
    <model>
      <name>golf comfortline</name>
      <paint>metalic</paint>
      <engine_size>
        <size>1.2</size>
        <fuel_type>diesel</fuel_type>
      </engine_size>
      <interior>comfort</interior>
      <num_doors>5</num_doors>
    </model>
  </car>
  <car id="444" type="used">
    <make>Toyota</make>
    <model>
      <name>Yaris</name>
      <paint>matt</paint>
      <engine_size>
        <size>1.2</size>
        <fuel_type>petrol</fuel_type>
      </engine_size>
    </model>
    <interior>standard</interior>
    <num_doors>3</num_doors>
    <licence>03W24567</licence>
    <mileage>103000</mileage>
  </car>
</car_stock>
```



```
car_catalog
  car type=new
    make
    model
      name
      paint
      engine_size
        size
        fuel_type
      engine_size
      engine_size
      interior
      num_doors
    model
      name
      paint
      engine_size
      engine_size
      engine_size
      interior
      num_doors
```



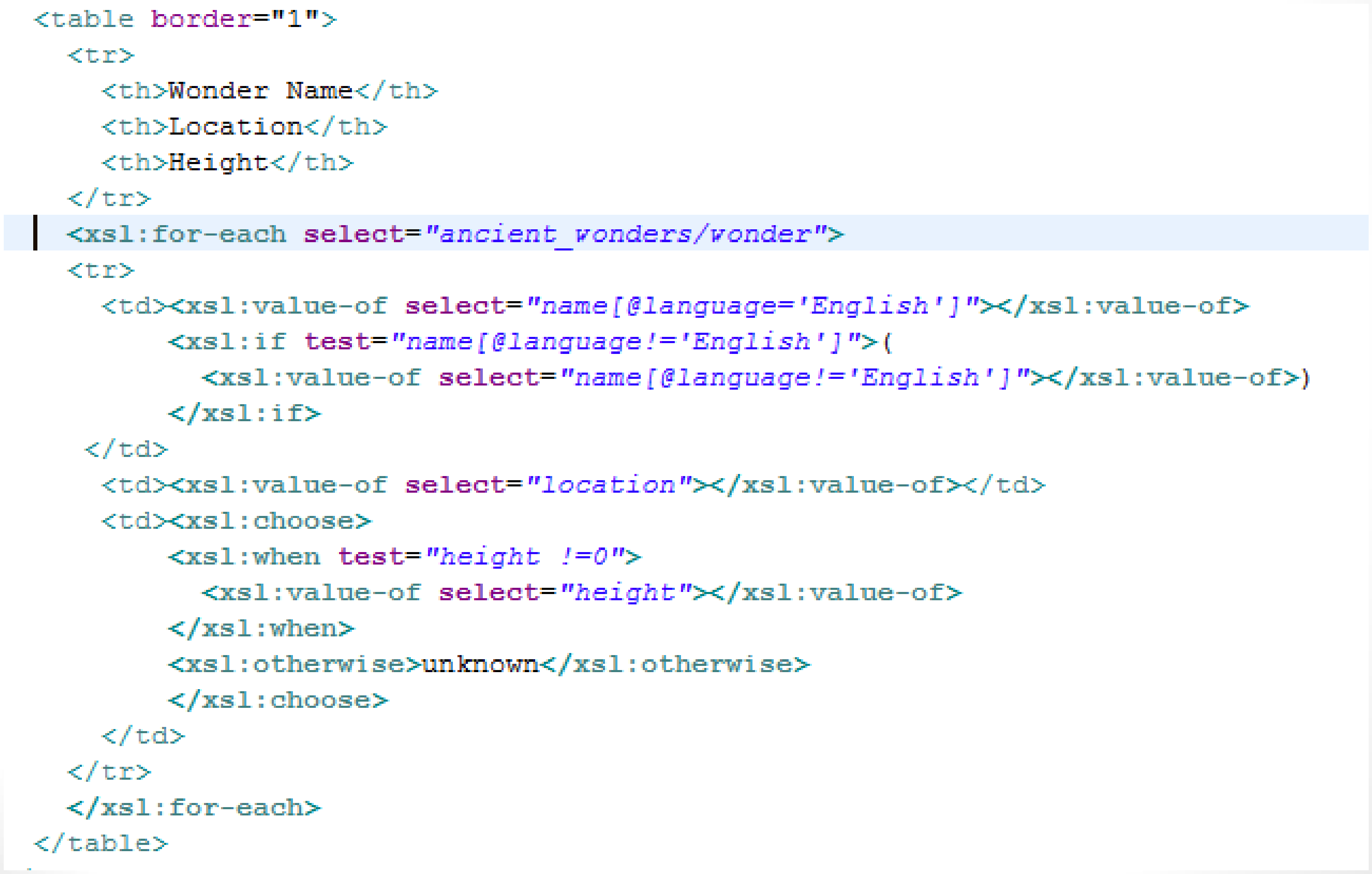
```
car_stock
  car id=123
    make
    model
      name
      paint
      engine_size
        size
        fuel_type
      interior
      num_doors
  car id=444
    make
    model
    interior
    num_doors
    licence
    mileage
```

XSLT – eXtensible Style Language for Transformations

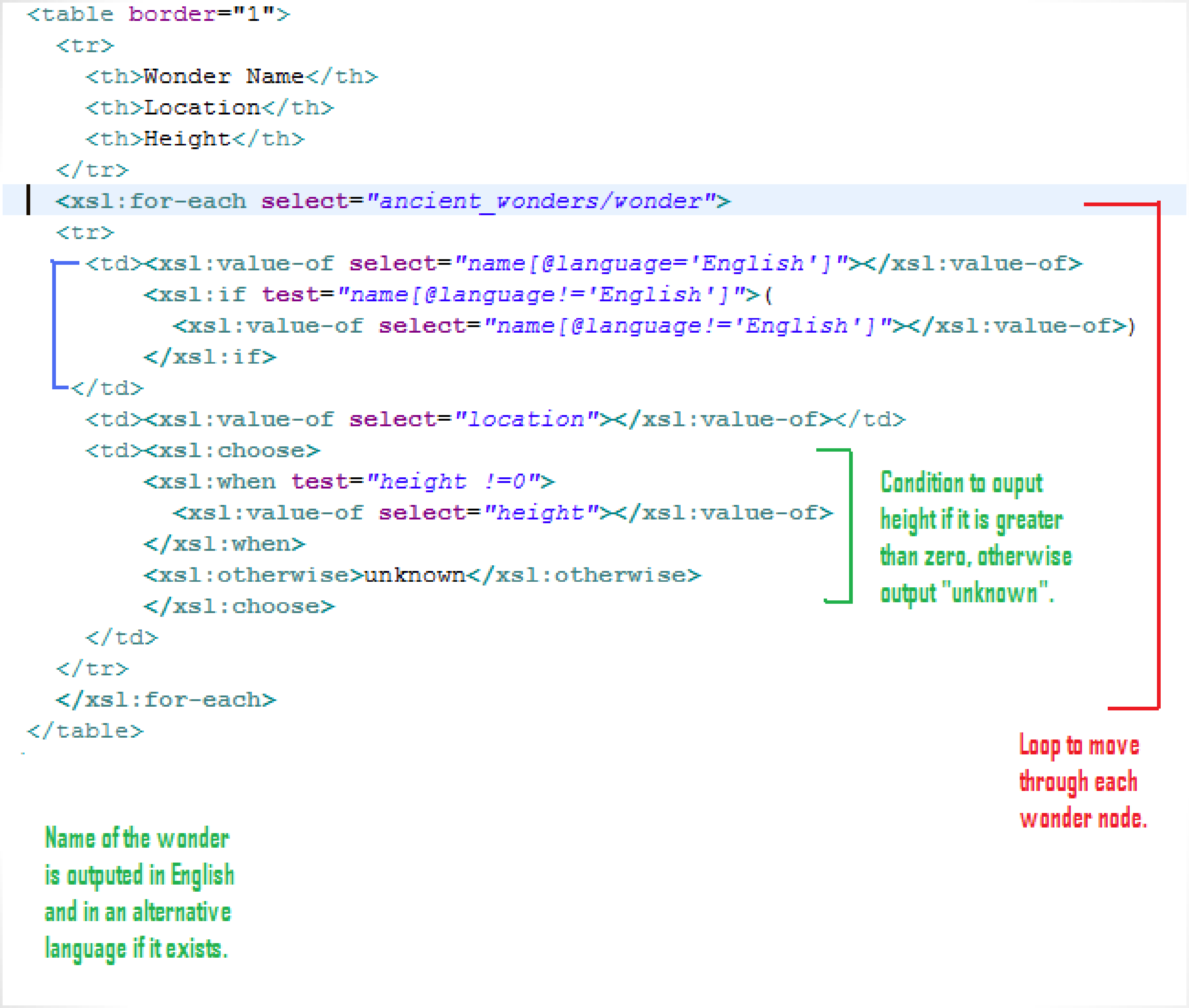
Rules covered so far:

<xsl:value-of>
<xsl:for-each>
<xsl:if>
<xsl:choose>
<xsl:when>
<xsl:otherwise>

XSLT – eXtensible Style Language for Transformations



```
<table border="1">
  <tr>
    <th>Wonder Name</th>
    <th>Location</th>
    <th>Height</th>
  </tr>
  <xsl:for-each select="ancient_wonders/wonder">
  <tr>
    <td><xsl:value-of select="name[@language='English']"></xsl:value-of>
        <xsl:if test="name[@language!='English']">(
          <xsl:value-of select="name[@language!='English']"></xsl:value-of>)
        </xsl:if>
     </td>
    <td><xsl:value-of select="location"></xsl:value-of></td>
    <td><xsl:choose>
        <xsl:when test="height !=0">
          <xsl:value-of select="height"></xsl:value-of>
        </xsl:when>
        <xsl:otherwise>unknown</xsl:otherwise>
        </xsl:choose>
    </td>
  </tr>
  </xsl:for-each>
</table>
```



```
<table border="1">
  <tr>
    <th>Wonder Name</th>
    <th>Location</th>
    <th>Height</th>
  </tr>
  <xsl:for-each select="ancient_wonders/wonder">
  <tr>
    <td><xsl:value-of select="name[@language='English']"></xsl:value-of>
        <xsl:if test="name[@language!='English']">(
          <xsl:value-of select="name[@language!='English']"></xsl:value-of>)
        </xsl:if>
   </td>
    <td><xsl:value-of select="location"></xsl:value-of></td>
    <td><xsl:choose>
        <xsl:when test="height !=0">
          <xsl:value-of select="height"></xsl:value-of>
        </xsl:when>
        <xsl:otherwise>unknown</xsl:otherwise>
        </xsl:choose>
    </td>
  </tr>
  </xsl:for-each>
</table>
```

Condition to ouput height if it is greater than zero, otherwise output "unknown".

Loop to move through each wonder node.

Name of the wonder is outputed in English and in an alternative language if it exists.

XSLT – eXtensible Style Language for Transformations

```
<?xml version="1.0" encoding=“UTF-8"?>
<CATALOG>
  <CD>
    <TITLE>Empire Burlesque</TITLE>
    <ARTIST gender=“male”>Bob Dylan</ARTIST>
    <COUNTRY type=“abbrev”>USA</COUNTRY>
    <COUNTRY type=“full”>United States of America</COUNTRY>
    <COMPANY>Columbia</COMPANY>
    <PRICE>10.90</PRICE>
    <YEAR>1985</YEAR>
  </CD>
  <CD>
    <TITLE>Hide your heart</TITLE>
    <ARTIST gender=“female”>Bonnie Tyler</ARTIST>
    <COUNTRY type=“abbrev”>UK</COUNTRY>
    <COMPANY>CBS Records</COMPANY>
    <PRICE>9.90</PRICE>
    <YEAR>1988</YEAR>
  </CD>
</CATALOG>
```

XSLT – eXtensible Style Language for Transformations

Write the xsl instructions to output the following:

1. All of the details of the first CD element.
2. Just the title and the artist of the first CD element.
3. Just the artist of the first CD element that has a female artist.
4. All of the CD titles and artists.
5. All of the CD titles and artists and country full name.
6. All of the CD details, if the country full name is not there output the abbreviated name instead.