XSLT Queries

Query 1 - List the first and last names of all Staff

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

  <xsl:template match="/">
    <StaffNames>
      <xsl:for-each select="Hotel/Staff">
        <Staff>
          <FirstName><xsl:value-of select="FirstName"/></FirstName>
          <LastName><xsl:value-of select="LastName"/></LastName>
        </Staff>
      </xsl:for-each>
    </StaffNames>
  </xsl:template>

</xsl:stylesheet>
```

Query 2 - List all RoomNo’s for a given hotel name

```
<!-- XSLT query 2: List all room details -->
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <!-- Define a parameter for the hotel name -->
  <xsl:param name="hotelName">Grand Hotel</xsl:param>

  <!-- Key to match HotelCode with HotelName -->
  <xsl:key name="hotel-by-name" match="Hotel" use="HotelName"/>

  <!-- Template to match HotelCode for the given hotel name -->
  <xsl:template match="/">
    <!-- Find the HotelCode for the given hotel name -->
    <xsl:variable name="hotelCode" select="key('hotel-by-name',
$hotelName)/HotelCode"/>
    <!-- Apply the template to select RoomNo for the matched HotelCode -->
    <xsl:apply-templates select="/Database/Room[HotelCode = $hotelCode]"/>
  </xsl:template>

  <!-- Template to select RoomNo for the matched HotelCode -->
  <xsl:template match="Room">
    <xsl:value-of select="RoomNo"/>
    <xsl:text>, </xsl:text> <!-- Add comma and space for separation -->
  </xsl:template>

</xsl:stylesheet>
```

Query 3 - List all guest first and last names

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

  <!-- Template to match all first and last names -->
  <xsl:template match="/">
    <xsl:apply-templates select="//FirstName | //LastName"/>
  </xsl:template>

  <!-- Template to output first and last names -->
  <xsl:template match="FirstName | LastName">
    <xsl:value-of select="."/>
    <xsl:text> </xsl:text> <!-- Add space for separation -->
  </xsl:template>

</xsl:stylesheet>
```

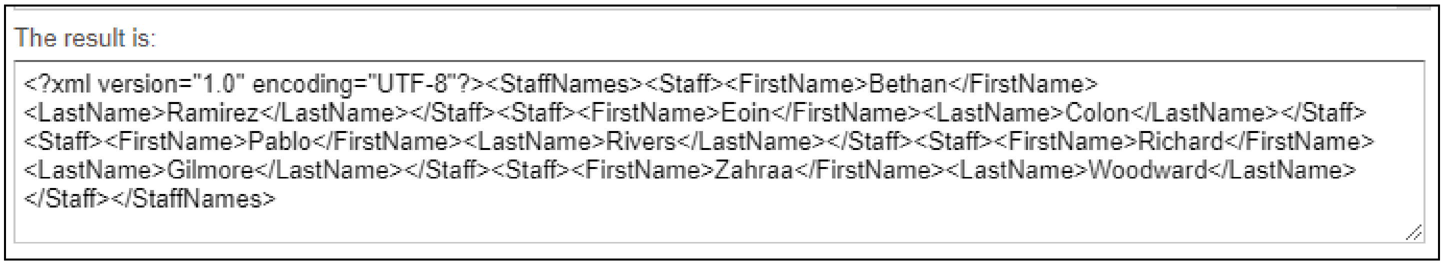
Query 4 - List all resource name and quantities

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

  <!-- Template to match all resource names and quantities -->
  <xsl:template match="/">
    <xsl:apply-templates select="//Resource"/>
  </xsl:template>

  <!-- Template to output resource names and quantities -->
  <xsl:template match="Resource">
    <xsl:value-of select="concat(Name, ' - ', Quantity)"/>
    <xsl:text>&#10;</xsl:text> <!-- Add newline for separation -->
  </xsl:template>

</xsl:stylesheet>
```

Query 5 - List all expense names for a given hotel name

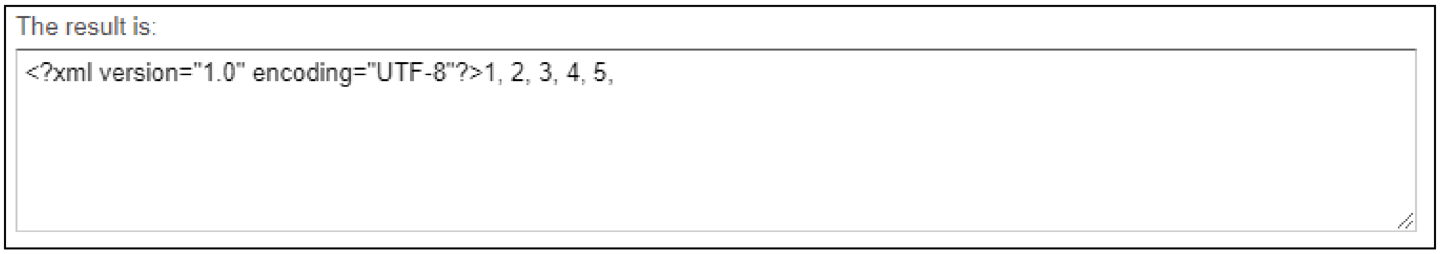
```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <!-- Parameter to specify the hotel name -->
  <xsl:param name="hotelName" select="'Grand Hotel'"/>

  <!-- Template to match the expenses for the specified hotel -->
  <xsl:template match="/">
    <xsl:for-each select="//Expense[../HotelCode = //Hotel[HotelName =
$hotelName]/HotelCode]">
      <!-- Output the expense name -->
      <xsl:value-of select="Name"/>
      <xsl:text>&#10;</xsl:text> <!-- Line break -->
    </xsl:for-each>
  </xsl:template>

</xsl:stylesheet>
```


The result is:

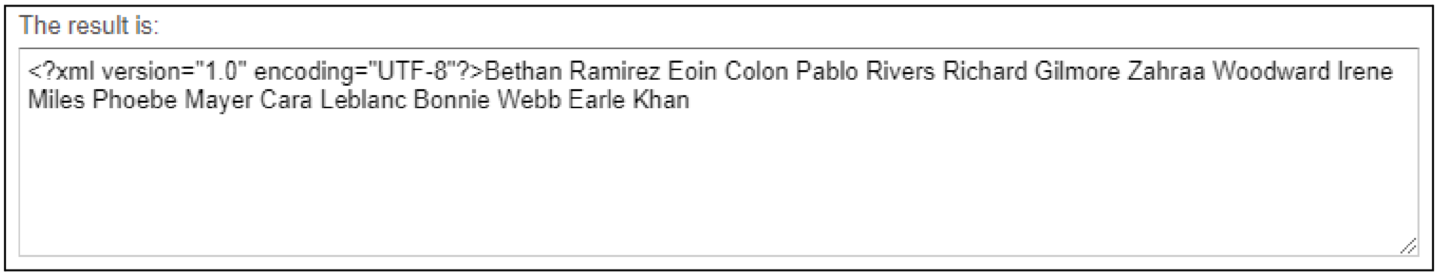
```
<?xml version="1.0" encoding="UTF-8"?><StaffNames><Staff><FirstName>Bethan</FirstName>
<LastName>Ramirez</LastName></Staff><Staff><FirstName>Eoin</FirstName><LastName>Colon</LastName></Staff>
<Staff><FirstName>Pablo</FirstName><LastName>Rivers</LastName></Staff><Staff><FirstName>Richard</FirstName>
<LastName>Gilmore</LastName></Staff><Staff><FirstName>Zahraa</FirstName><LastName>Woodward</LastName>
</Staff></StaffNames>
```

Figure 1: Query 1 results.


The result is:

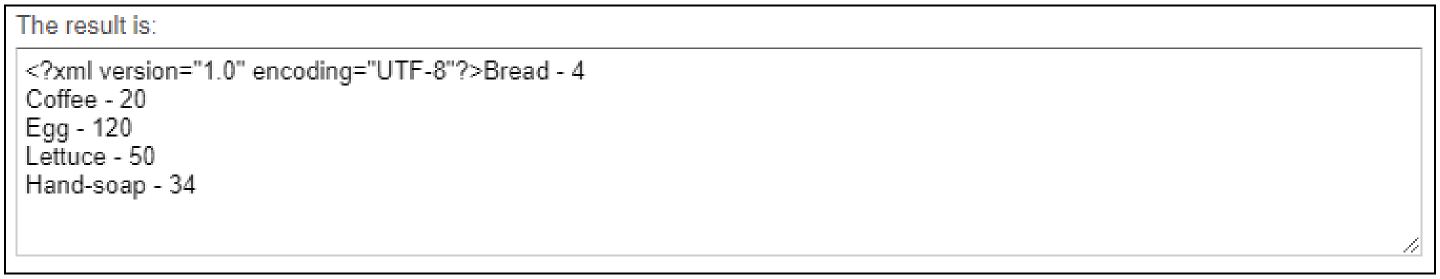
```
<?xml version="1.0" encoding="UTF-8"?>1, 2, 3, 4, 5,
```

Figure 2: Query 2 results.


The result is:

```
<?xml version="1.0" encoding="UTF-8"?>Bethan Ramirez Eoin Colon Pablo Rivers Richard Gilmore Zahraa Woodward Irene
Miles Phoebe Mayer Cara Leblanc Bonnie Webb Earle Khan
```

Figure 3: Query 3 results.


The result is:

```
<?xml version="1.0" encoding="UTF-8"?>Bread - 4
Coffee - 20
Egg - 120
Lettuce - 50
Hand-soap - 34
```

Figure 4: Query 4 results.