华中师范大学伍伦贡联合研究院
Central China Normal University Wollongong Joint Institute

UNIVERSITY
OF WOLLONGONG
AUSTRALIA

CSIT884
Web Development

Lecture 08 – XSLT

Lecture slides are mainly taken from UOW

XSLT

- E**X**tensible **S**tylesheet **L**anguage **T**ransformation (XSLT) is an XML language for transforming XML documents
  - file extension is .xsl
  - used to transform XML file into other file formats, such as HTML
  - describes how the XML elements should be displayed

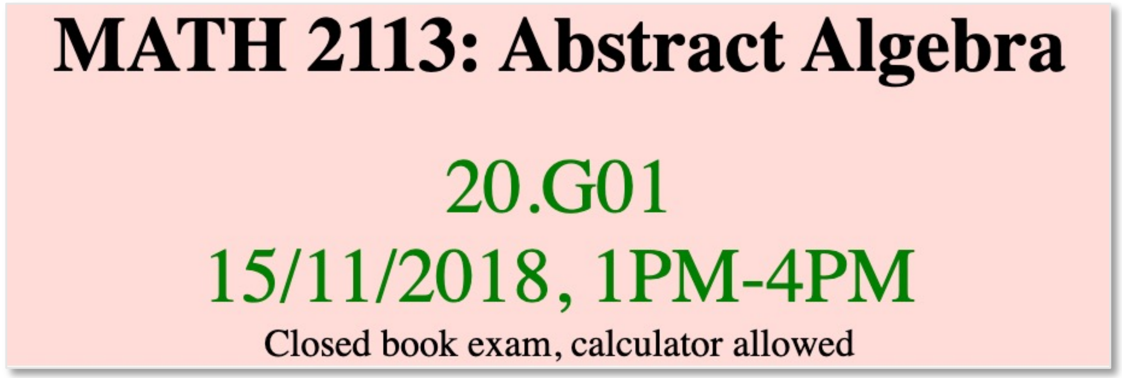
The content of the stylesheet XSL file looks like the following:

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

 ... xslt code here ...

</xsl:stylesheet>
```

Example: Exam timetable

This is an XML data representing exam information for a particular subject:

```
<exam subject="MATH 2113">
 <title>Abstract Algebra</title>
 <venue>20.G01</venue>
 <date>15/11/2018</date>
 <time>1PM-4PM</time>
 <note>Closed book exam, calculator allowed</note>
</exam>
```


MATH 2113: Abstract Algebra

20.G01

15/11/2018, 1PM-4PM

Closed book exam, calculator allowed

We would like to transform this XML content into the HTML display

Example: Exam timetable

exam.xml

```
<?xml version="1.0" ?>
<exam subject="MATH 2113">
 <title>Abstract Algebra</title>
 <venue>20.G01</venue>
 <date>15/11/2018</date>
 <time>1PM-4PM</time>
 <note>Closed book exam, calculator allowed</note>
</exam>
```

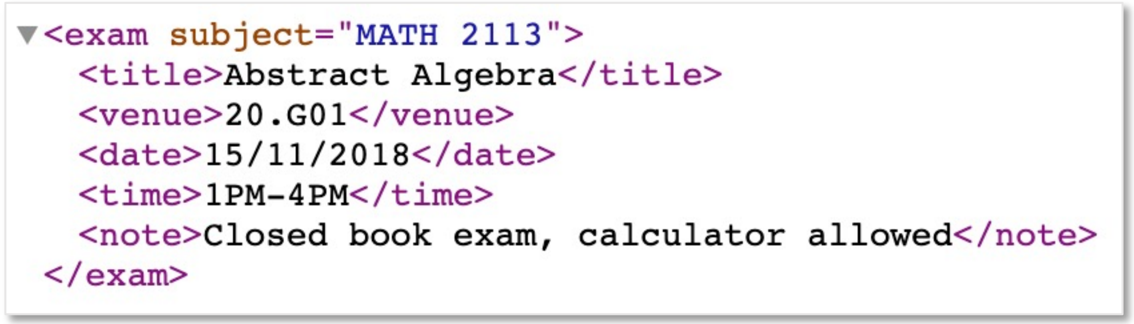
exam-with-style.xml

```
<?xml version="1.0" ?>
<?xml-stylesheet type="text/xsl" href="style-exam.xsl"?>
<exam subject="MATH 2113">
 <title>Abstract Algebra</title>
 <venue>20.G01</venue>
 <date>15/11/2018</date>
 <time>1PM-4PM</time>
 <note>Closed book exam, calculator allowed</note>
</exam>
```

These two XML files have almost the same content.
The 2nd one has a reference to a stylesheet. Let’s look at them on a browser.

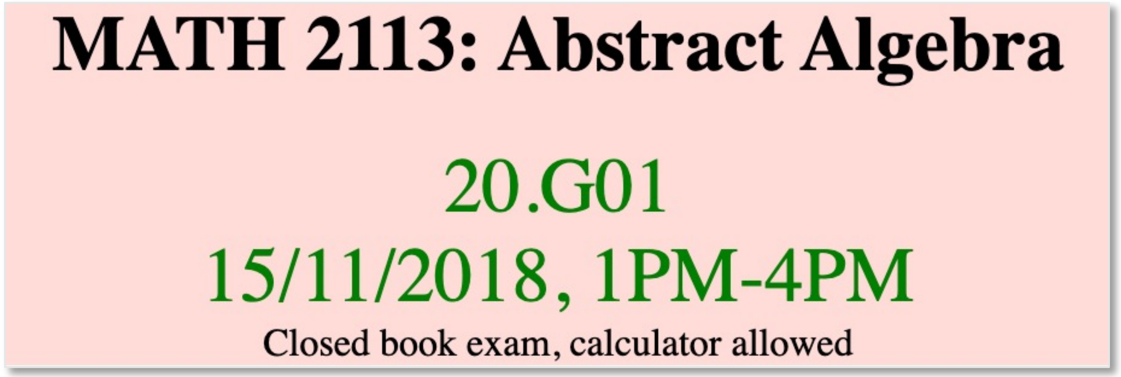
Example: Exam timetable

`exam.xml`



```
▼<exam subject="MATH 2113">
   <title>Abstract Algebra</title>
   <venue>20.G01</venue>
   <date>15/11/2018</date>
   <time>1PM-4PM</time>
   <note>Closed book exam, calculator allowed</note>
 </exam>
```

`exam-with-style.xml`



MATH 2113: Abstract Algebra

20.G01

15/11/2018, 1PM-4PM

Closed book exam, calculator allowed

On web browser, the two XML files display differently.

This is because the second XML uses a stylesheet.

```
<?xml-stylesheet type="text/xsl" href="style-exam.xsl"?>
```

Let's have a look at the **stylesheet file**

Example: Exam timetable

Let’s have a look at the stylesheet file: **`style-exam.xsl`**

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns="http://www.w3.org/1999/xhtml">
 <xsl:template match="/exam">
   <html>
     <head>
       <title>XSLT example</title>
     </head>
     <body>
        ...
     </body>
   </html>
 </xsl:template>
</xsl:stylesheet>
```

What do you see in this stylesheet file: `style-exam.xsl`?

Example: Exam timetable

Stylesheet file: **style-exam.xsl**

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0”xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns="http://www.w3.org/1999/xhtml">
 <xsl:template match="/exam">
   <html>
     <head>
       <title>XSLT example</title>
     </head>
     <body>
        ...
     </body>
   </html>
 </xsl:template>
</xsl:stylesheet>
```

This is the **root element** of the XML file

```
<?xml version="1.0" ?>
<exam subject="MATH 2113">
 <title>Abstract Algebra</title>
 <venue>20.G01</venue>
 <date>15/11/2018</date>
 <time>1PM-4PM</time>
 <note>Closed book exam, calculator allowed</note>
</exam>
```

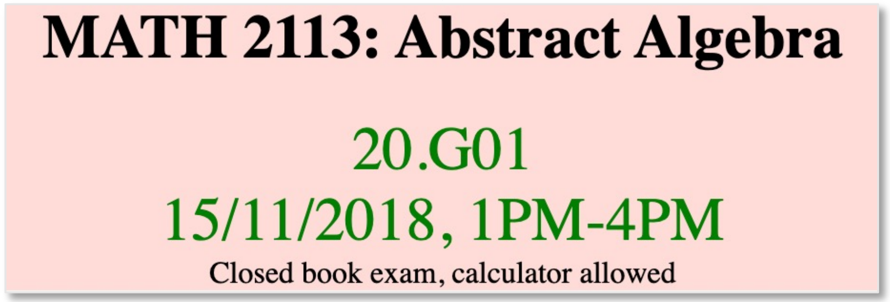
Example: Exam timetable

Stylesheet file: **`style-exam.xsl`**

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0”xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns="http://www.w3.org/1999/xhtml">
 <xsl:template match="/exam">
  <html>
    <head>
      <title>XSLT example</title>
    </head>
    <body>
       ...
    </body>
  </html>
 </xsl:template>
</xsl:stylesheet>
```

This looks like HTML code

That’s why the XML file is displayed in the browser as HTML



MATH 2113: Abstract Algebra

20.G01

15/11/2018, 1PM-4PM

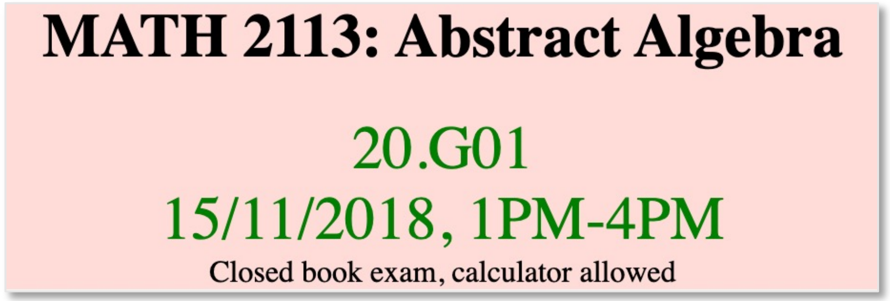
Closed book exam, calculator allowed

Example: Exam timetable

Stylesheet file: **style-exam.xsl**

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns="http://www.w3.org/1999/xhtml">
 <xsl:template match="/exam">
  <html>
    <head>
      <title>XSLT example</title>
    </head>
    <body>
       ...
    </body>
  </html>
 </xsl:template>
</xsl:stylesheet>
```

Let's have a look at this code in more detail


MATH 2113: Abstract Algebra

20.G01

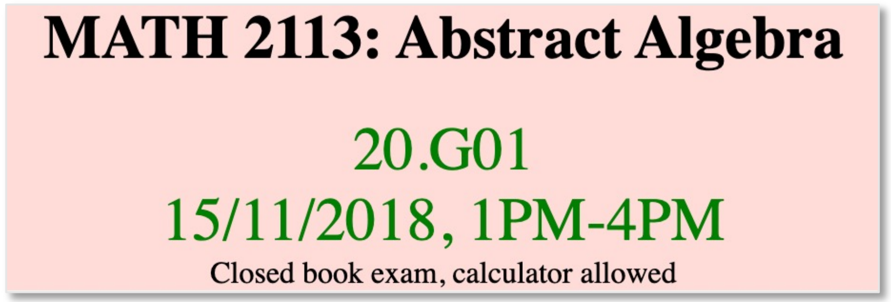
15/11/2018, 1PM-4PM

Closed book exam, calculator allowed

Example: Exam timetable

Stylesheet file: **style-exam.xsl**

```
...
<body>
 <div align="center" style="background-color:#f0371930">
   <h1>
    <xsl:value-of select="@subject" />
    <xsl:text>: </xsl:text>
    <xsl:value-of select="title" />
   </h1>
   <font size="6" color="green">
    <xsl:value-of select="venue" />
    <br />
    <xsl:value-of select="date" />
    <xsl:text>, </xsl:text>
    <xsl:value-of select="time" />
   </font>
   <br />
   <xsl:value-of select="note" />
 </div>
</body>
...
```


MATH 2113: Abstract Algebra

20.G01

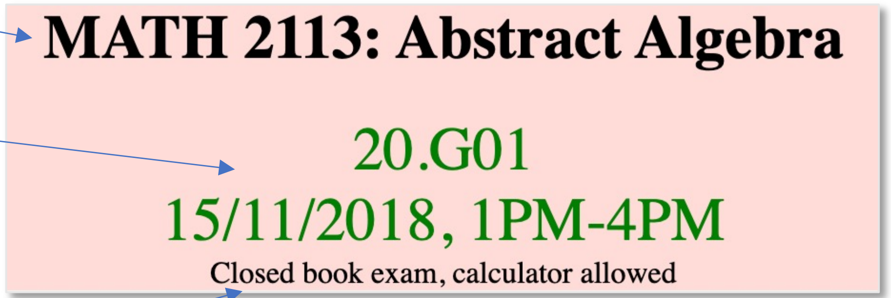
15/11/2018, 1PM-4PM

Closed book exam, calculator allowed

Example: Exam timetable

Stylesheet file: `style-exam.xsl`

```
...
<body>
 <div align="center" style="background-color:#f0371930">
   <h1>
    <xsl:value-of select="@subject" />
    <xsl:text>: </xsl:text>
    <xsl:value-of select="title" />
   </h1>
   <font size="6" color="green">
    <xsl:value-of select="venue" />
    <br />
    <xsl:value-of select="date" />
    <xsl:text>, </xsl:text>
    <xsl:value-of select="time" />
   </font>
   <br />
   <xsl:value-of select="note" />
 </div>
</body>
...
```



MATH 2113: Abstract Algebra

20.G01

15/11/2018, 1PM-4PM

Closed book exam, calculator allowed

Example: Exam timetable

Stylesheet file: **style-exam.xsl**

```
...
<xsl:value-of select="@subject" />
<xsl:text>: </xsl:text>
<xsl:value-of select="title" />

<xsl:value-of select="venue" />

<xsl:value-of select="date" />
<xsl:text>, </xsl:text>
<xsl:value-of select="time" />

<xsl:value-of select="note" />
...
```

```
<?xml version="1.0" ?>
<exam subject="MATH 2113">
 <title>Abstract Algebra</title>
 <venue>20.G01</venue>
 <date>15/11/2018</date>
 <time>1PM-4PM</time>
 <note>Closed book exam, calculator
allowed</note>
</exam>
```

```
MATH 2113
```

Get value from attribute

Example: Exam timetable

Stylesheet file: **style-exam.xsl**

```
...
<xsl:value-of select="@subject" />
<xsl:text>: </xsl:text>
<xsl:value-of select="title" />

<xsl:value-of select="venue" />

<xsl:value-of select="date" />
<xsl:text>, </xsl:text>
<xsl:value-of select="time" />

<xsl:value-of select="note" />
...
```

```
<?xml version="1.0" ?>
<exam subject="MATH 2113">
 <title>Abstract Algebra</title>
 <venue>20.G01</venue>
 <date>15/11/2018</date>
 <time>1PM-4PM</time>
 <note>Closed book exam, calculator
allowed</note>
</exam>
```

```
Abstract Algebra
20.G01
15/11/2018
1PM-4PM
Closed book exam, calculator allowed
```

Get value from element

Example: Exam timetable

Stylesheet file: **style-exam.xsl**

```
...
<xsl:value-of select="@subject" />
<xsl:text>: </xsl:text>
<xsl:value-of select="title" />

<xsl:value-of select="venue" />

 <xsl:value-of select="date" />
 <xsl:text>, </xsl:text>
 <xsl:value-of select="time" />

 <xsl:value-of select="note" />
...
```

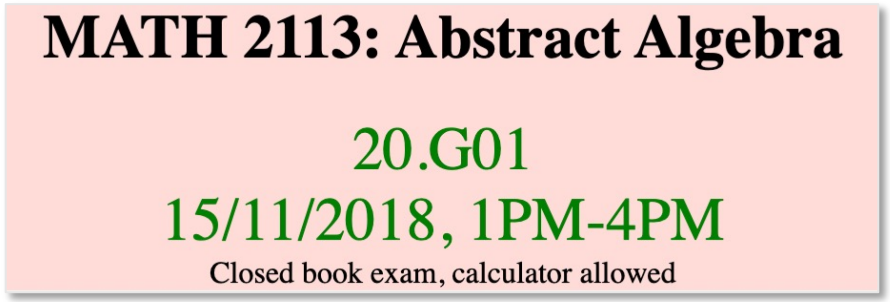
Write literal text

```
<?xml version="1.0" ?>
<exam subject="MATH 2113">
 <title>Abstract Algebra</title>
 <venue>20.G01</venue>
 <date>15/11/2018</date>
 <time>1PM-4PM</time>
 <note>Closed book exam, calculator
allowed</note>
</exam>
```

Example: Exam timetable

Stylesheet file: **style-exam.xsl**

```
...
<body>
 <div align="center" style="background-color:#f0371930">
   <h1>
    <xsl:value-of select="@subject" />  MATH 2113
    <xsl:text>: </xsl:text>
    <xsl:value-of select="title" />  Abstract Algebra
   </h1>
   <font size="6" color="green">
    <xsl:value-of select="venue" />  20.G01
    <br />
    <xsl:value-of select="date" />  15/11/2018
    <xsl:text>, </xsl:text>
    <xsl:value-of select="time" />  1PM-4PM
   </font>
   <br />
   <xsl:value-of select="note" />  Closed book exam, calculator allowed
 </div>
</body>
...
```



MATH 2113: Abstract Algebra

20.G01

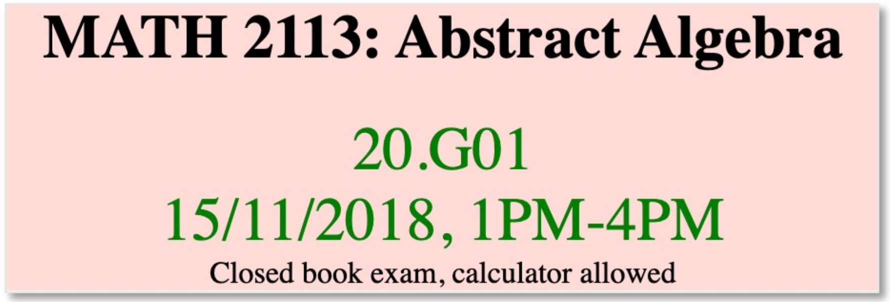
15/11/2018, 1PM-4PM

Closed book exam, calculator allowed

Example: Exam timetable

Stylesheet file: `style-exam.xsl`

```
...
<body>
 <div align="center" style="background-color:#f0371930">
   <h1>
       MATH 2113
              :
     Abstract Algebra
   </h1>
   <font size="6" color="green">
          20.G01
    <br />
          15/11/2018
               ,
            1PM-4PM
   </font>
   <br />
    Closed book exam, calculator allowed
 </div>
</body>
...
```



MATH 2113: Abstract Algebra

20.G01
15/11/2018, 1PM-4PM
Closed book exam, calculator allowed

Summary

- Get attribute value:

```
<xsl:value-of select="@attribute-name" />
```

- Get element value:

```
<xsl:value-of select="element-name" />
```

- Literal text:

```
<xsl:text>some text here ...</xsl:text>
```

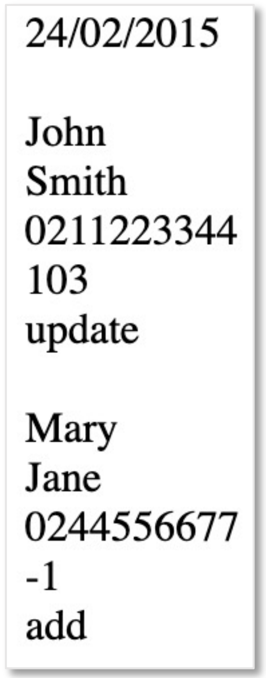
Example: Transaction v0

`transaction0.xml` uses stylesheet **`transaction-style0.xsl`**

```
<?xml version="1.0" ?>
<?xml-stylesheet type="text/xsl" href="transaction-style0.xsl"?>
<dailyTransaction date="24/02/2015">
 <person staffDbId="103" operation="update">
  <firstName>John</firstName>
  <lastName>Smith</lastName>
  <mobile>0211223344</mobile>
 </person>
 <person staffDbId="-1" operation="add">
  <firstName>Mary</firstName>
  <lastName>Jane</lastName>
  <mobile>0244556677</mobile>
 </person>
 ...
</dailyTransaction>
```

Example: Transaction v0

Have a look at the XML file `transaction0.xml` in the browser



```
24/02/2015

John
Smith
0211223344
103
update

Mary
Jane
0244556677
-1
add
```

Example: Transaction v0

Have a look at the XML stylesheet **`transaction-style0.xsl`**

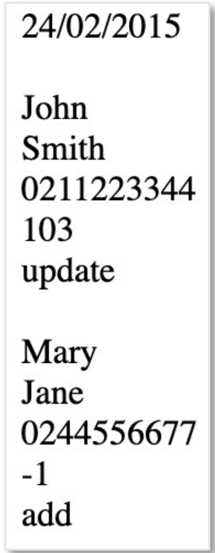
```
...
<xsl:template match="/dailyTransaction">
 <html>
  <head>
   <title>XSLT example</title>
  </head>
  <body>
  ...
  </body>
 </html>
</xsl:template>
...
```

```
<dailyTransaction date="24/02/2015">
 <person staffDbId="103" operation="update">
  <firstName>John</firstName>
  <lastName>Smith</lastName>
  <mobile>0211223344</mobile>
 </person>
 <person staffDbId="-1" operation="add">
  <firstName>Mary</firstName>
  <lastName>Jane</lastName>
  <mobile>0244556677</mobile>
 </person>
 ...
</dailyTransaction>
```

Example: Transaction v0

Have a look at the XML stylesheet **`transaction-style0.xsl`**

```
...
<body>
 <xsl:value-of select="@date" /> <br /><br />
 <xsl:for-each select="person">
  <xsl:value-of select="firstName" />
  <br />
  <xsl:value-of select="lastName" />
  <br />
  <xsl:value-of select="mobile" />
  <br />
  <xsl:value-of select="@staffDbId" />
  <br />
  <xsl:value-of select="@operation" />
  <br />
  <br />
 </xsl:for-each>
</body>
...
```



```
24/02/2015

John
Smith
0211223344
103
update

Mary
Jane
0244556677
-1
add
```

```
<dailyTransaction date="24/02/2015">
 <person staffDbId="103" operation="update">
  <firstName>John</firstName>
  <lastName>Smith</lastName>
  <mobile>0211223344</mobile>
 </person>
 <person staffDbId="-1" operation="add">
  <firstName>Mary</firstName>
  <lastName>Jane</lastName>
  <mobile>0244556677</mobile>
 </person>
 ...
</dailyTransaction>
```

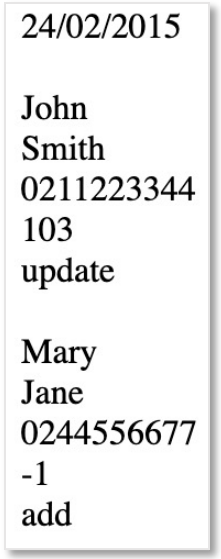
Example: Transaction v0

Have a look at the XML stylesheet **`transaction-style0.xsl`**

```
...
<body>
 <xsl:value-of select="@date" /> <br /><br />
 <xsl:for-each select="person">
  <xsl:value-of select="firstName" />
  <br />
  <xsl:value-of select="lastName" />
  <br />
  <xsl:value-of select="mobile" />
  <br />
  <xsl:value-of select="@staffDbId" />
  <br />
  <xsl:value-of select="@operation" />
  <br />
  <br />
 </xsl:for-each>
</body>
...
```

FOR loop

```
<dailyTransaction date="24/02/2015">
 <person staffDbId="103" operation="update">
  <firstName>John</firstName>
  <lastName>Smith</lastName>
  <mobile>0211223344</mobile>
 </person>
 <person staffDbId="-1" operation="add">
  <firstName>Mary</firstName>
  <lastName>Jane</lastName>
  <mobile>0244556677</mobile>
 </person>
 ...
</dailyTransaction>
```



24/02/2015

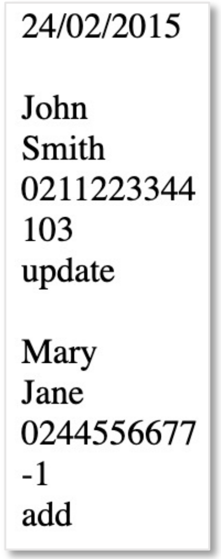
John
Smith
0211223344
103
update

Mary
Jane
0244556677
-1
add

Example: Transaction v0

Have a look at the XML stylesheet **transaction-style0.xsl**

```
...
<body>
 <xsl:value-of select="@date" /> <br /><br />
 <xsl:for-each select="person">
  <xsl:value-of select="firstName" />
  <br />
  <xsl:value-of select="lastName" />
  <br />
  <xsl:value-of select="mobile" />
  <br />
  <xsl:value-of select="@staffDbId" />
  <br />
  <xsl:value-of select="@operation" />
  <br />
  <br />
 </xsl:for-each>
</body>
...
```



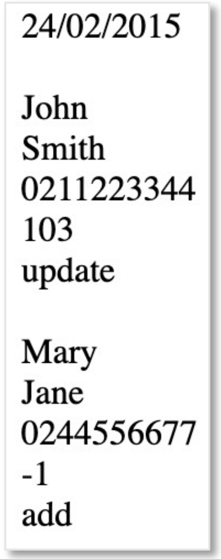
```
24/02/2015

John
Smith
0211223344
103
update

Mary
Jane
0244556677
-1
add
```

```
<dailyTransaction date="24/02/2015">
 <person staffDbId="103" operation="update">
  <firstName>John</firstName>
  <lastName>Smith</lastName>
  <mobile>0211223344</mobile>
 </person>
 <person staffDbId="-1" operation="add">
  <firstName>Mary</firstName>
  <lastName>Jane</lastName>
  <mobile>0244556677</mobile>
 </person>
 ...
</dailyTransaction>
```

Example: Transaction v0

Have a look at the XML stylesheet **`transaction-style0.xsl`**

```
...
<body>
 <xsl:value-of select="@date" /> <br /><br />
 <xsl:for-each select="person">
  <xsl:value-of select="firstName" />
  <br />
  <xsl:value-of select="lastName" />
  <br />
  <xsl:value-of select="mobile" />
  <br />
  <xsl:value-of select="@staffDbId" />
  <br />
  <xsl:value-of select="@operation" />
  <br />
  <br />
 </xsl:for-each>
</body>
...
```



```
24/02/2015

John
Smith
0211223344
103
update

Mary
Jane
0244556677
-1
add
```

```
<dailyTransaction date="24/02/2015">
 <person staffDbId="103" operation="update">
  <firstName>John</firstName>
  <lastName>Smith</lastName>
  <mobile>0211223344</mobile>
 </person>
 <person staffDbId="-1" operation="add">
  <firstName>Mary</firstName>
  <lastName>Jane</lastName>
  <mobile>0244556677</mobile>
 </person>
 ...
</dailyTransaction>
```

Example: Transaction v1

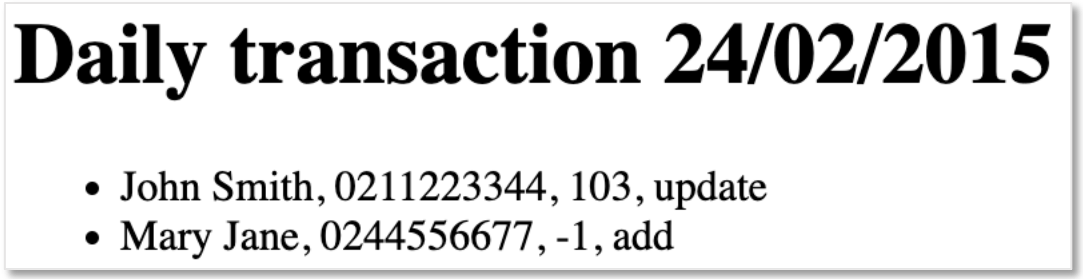
`transaction1.xml` uses stylesheet **`transaction-style1.xsl`**

```
<?xml version="1.0" ?>
<?xml-stylesheet type="text/xsl" href="transaction-style1.xsl"?>
<dailyTransaction date="24/02/2015">
 <person staffDbId="103" operation="update">
  <firstName>John</firstName>
  <lastName>Smith</lastName>
  <mobile>0211223344</mobile>
 </person>
 <person staffDbId="-1" operation="add">
  <firstName>Mary</firstName>
  <lastName>Jane</lastName>
  <mobile>0244556677</mobile>
 </person>
 ...
</dailyTransaction>
```

Example: Transaction v1

Have a look at the XML stylesheet **`transaction-style1.xsl`**

```
...
<h1>Daily transaction <xsl:value-of select="@date" /> </h1>
<ul>
 <xsl:for-each select="person">
  <li>
   <xsl:value-of select="firstName" />
   <xsl:text> </xsl:text>
   <xsl:value-of select="lastName" />
   <xsl:text>, </xsl:text>
   <xsl:value-of select="mobile" />
   <xsl:text>, </xsl:text>
   <xsl:value-of select="@staffDbId" />
   <xsl:text>, </xsl:text>
   <xsl:value-of select="@operation" />
  </li>
 </xsl:for-each>
</ul>
...
```



Daily transaction 24/02/2015

- John Smith, 0211223344, 103, update
- Mary Jane, 0244556677, -1, add

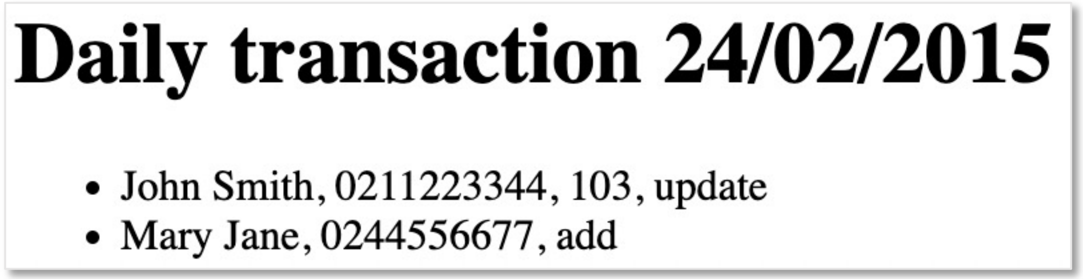
Example: Transaction v2

`transaction2.xml` uses stylesheet **`transaction-style2.xsl`**

Version 2 is exactly like version 1 except that we only display `staffDbId` if it is a positive number.

IF statement

```
…
<xsl:if test="@staffDbId &gt; 0">
 <xsl:text>, </xsl:text>
 <xsl:value-of select="@staffDbId" />
</xsl:if>
…
```



Daily transaction 24/02/2015

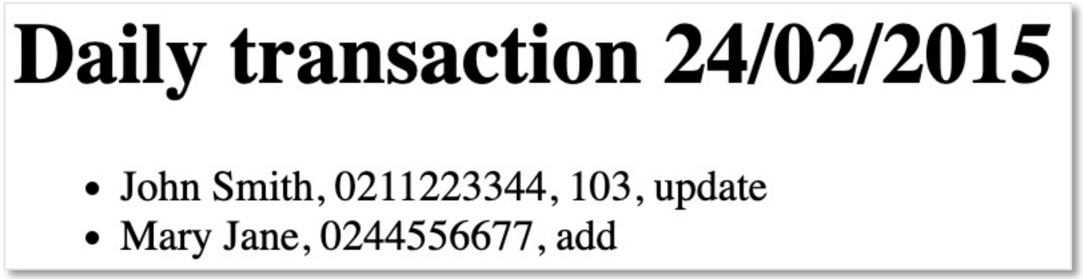
- John Smith, 0211223344, 103, update
- Mary Jane, 0244556677, add

Example: Transaction v2

```
…
<xsl:if test="@staffDbId &gt; 0">
 <xsl:text>, </xsl:text>
 <xsl:value-of select="@staffDbId" />
</xsl:if>
…
```

There is no IF-ELSE statement!

- IF statement will be applied when a specified condition is true.
- Use CHOOSE-WHEN-OTHERWISE statement to express multiple conditional tests.



Daily transaction 24/02/2015

- John Smith, 0211223344, 103, update
- Mary Jane, 0244556677, add

Conditional statement examples

Mark = 49:

```
mark = 49
```

Mark not equal to 49:

```
mark != 49
```

Student type = ‘U’:

```
type = 'U'
```

Student type not equal to ‘U’:

```
type != 'U'
```

Mark > 35:

```
mark &gt; 35
```

Mark >= 85:

```
mark &gt;= 85
```

Mark < 35:

```
mark &lt; 35
```

Mark <= 85:

```
mark &lt;= 85
```

```
…
<xsl:if test="@staffDbId &gt; 0">
 <xsl:text>, </xsl:text>
 <xsl:value-of select="@staffDbId" />
</xsl:if>
…
```

Mark NOT equal to 49:

```
not(mark = 49)
```

Student type = ‘U’ or ‘P’:

```
(type = 'U') or (type = 'P')
```

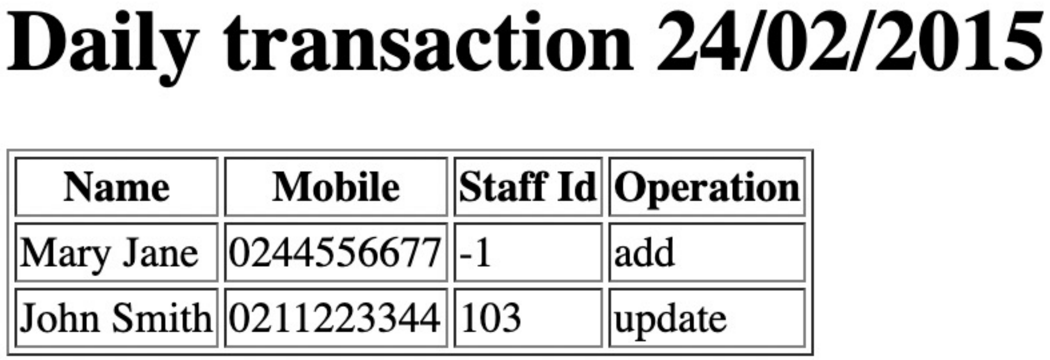
Mark >= 75 and mark < 85:

```
(mark &gt;= 75) and (mark &lt; 85)
```

Example: Transaction v3

Now look at `transaction3.xml`. It uses stylesheet **`transaction-style3.xsl`**

```
<?xml version="1.0" ?>
<?xml-stylesheet type="text/xsl" href="transaction-style3.xsl"?>
<dailyTransaction date="24/02/2015">
 <person staffDbId="103" operation="update">
  <firstName>John</firstName>
  <lastName>Smith</lastName>
  <mobile>0211223344</mobile>
 </person>
 <person staffDbId="-1" operation="add">
  <firstName>Mary</firstName>
  <lastName>Jane</lastName>
  <mobile>0244556677</mobile>
 </person>
 ...
</dailyTransaction>
```



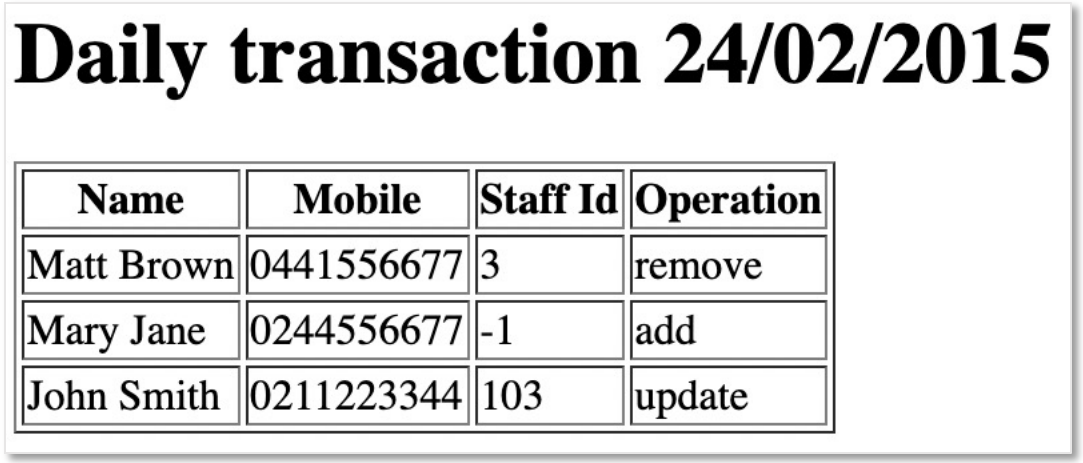
Daily transaction 24/02/2015

Name	Mobile	Staff Id	Operation
Mary Jane	0244556677	-1	add
John Smith	0211223344	103	update

Example: Transaction v3

View the source of the xml stylesheet: `transaction-style3.xsl`

We can see that it displays a table and the data is sorted.



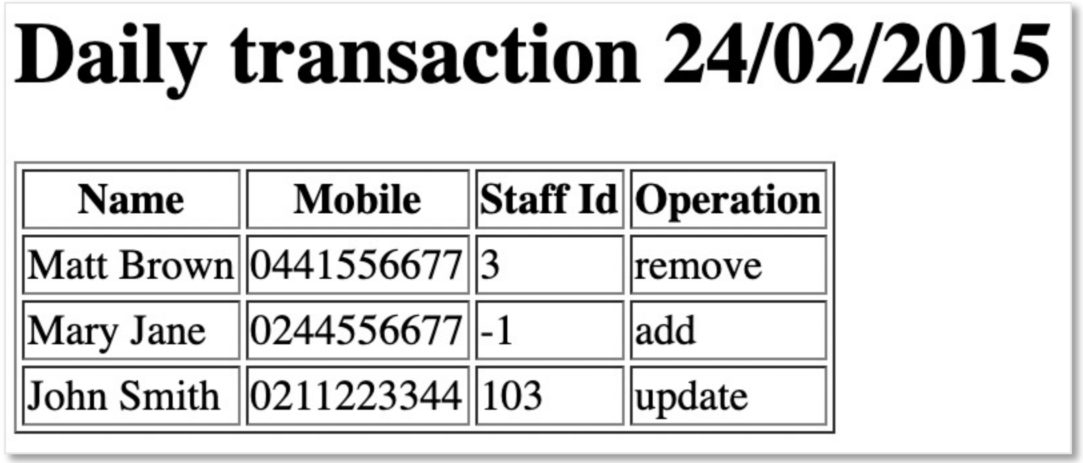
Daily transaction 24/02/2015

Name	Mobile	Staff Id	Operation
Matt Brown	0441556677	3	remove
Mary Jane	0244556677	-1	add
John Smith	0211223344	103	update

Example: Transaction v3

```
<xsl:for-each select="person">

  <xsl:sort select="lastName"/>

  ...

</xsl:for-each>
```

Sorted by **`lastName`**



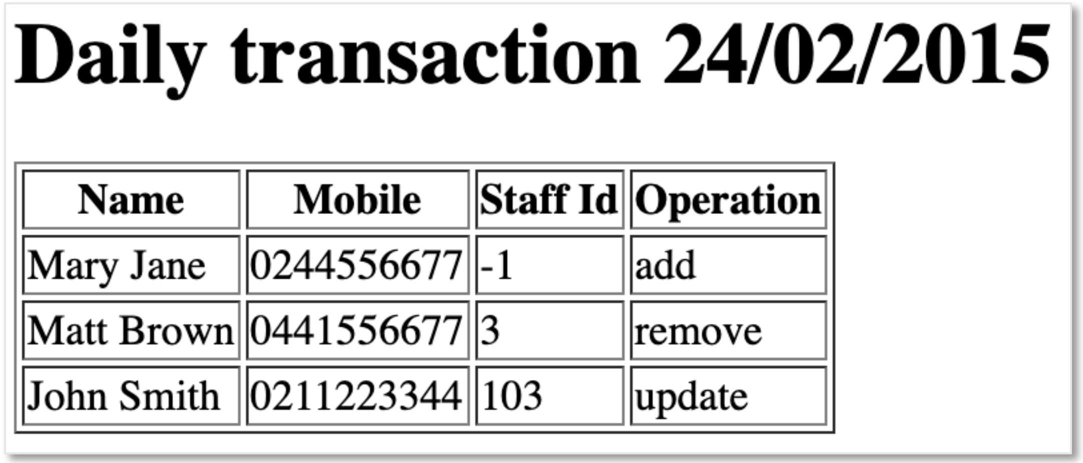
Daily transaction 24/02/2015

Name	Mobile	Staff Id	Operation
Matt Brown	0441556677	3	remove
Mary Jane	0244556677	-1	add
John Smith	0211223344	103	update

Example: Transaction v3

```
<xsl:for-each select="person">
  <xsl:sort select="@operation"/>
  ...
</xsl:for-each>
```

Sorted by **`operation`**



Daily transaction 24/02/2015

Name	Mobile	Staff Id	Operation
Mary Jane	0244556677	-1	add
Matt Brown	0441556677	3	remove
John Smith	0211223344	103	update

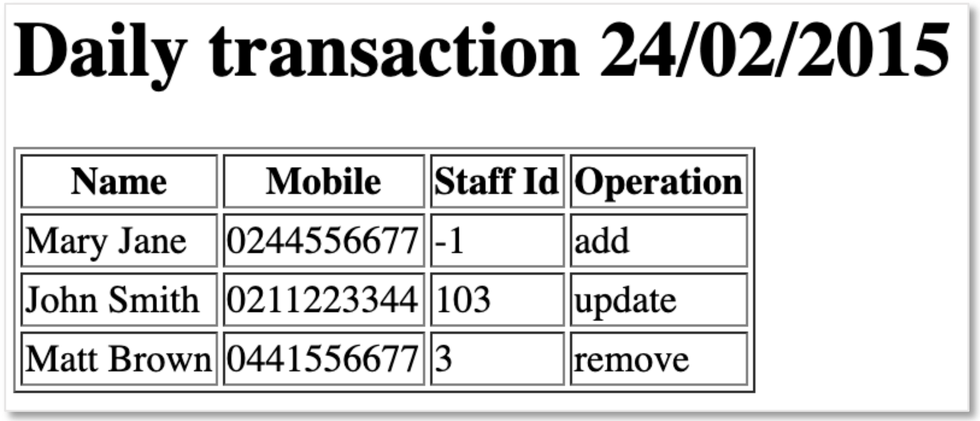
Example: Transaction v3

```
<xsl:for-each select="person">

  <xsl:sort select="@staffDbId"/>

  ...

</xsl:for-each>
```

Sorted by **`staffDbId`** as **string** data type

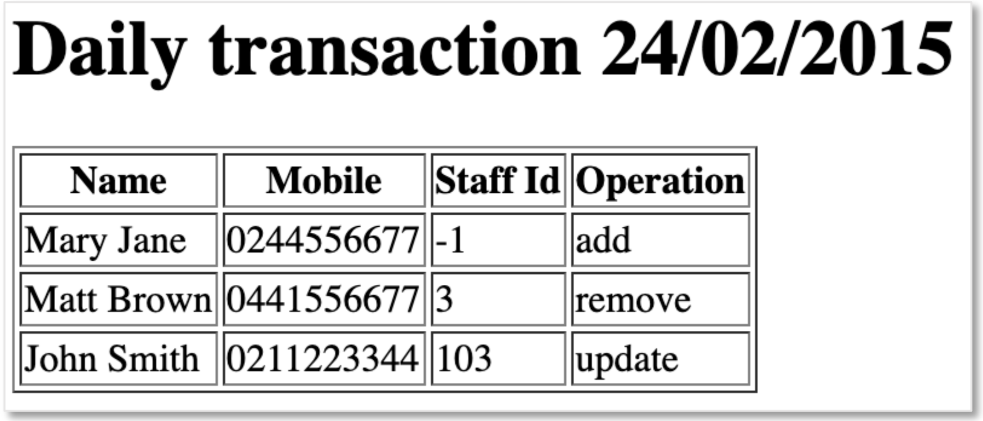
```
<xsl:for-each select="person">

  <xsl:sort select="@staffDbId" data-type="number"/>

  ...

</xsl:for-each>
```

Sorted by **`staffDbId`** as **number** data type



Daily transaction 24/02/2015

Name	Mobile	Staff Id	Operation
Mary Jane	0244556677	-1	add
John Smith	0211223344	103	update
Matt Brown	0441556677	3	remove

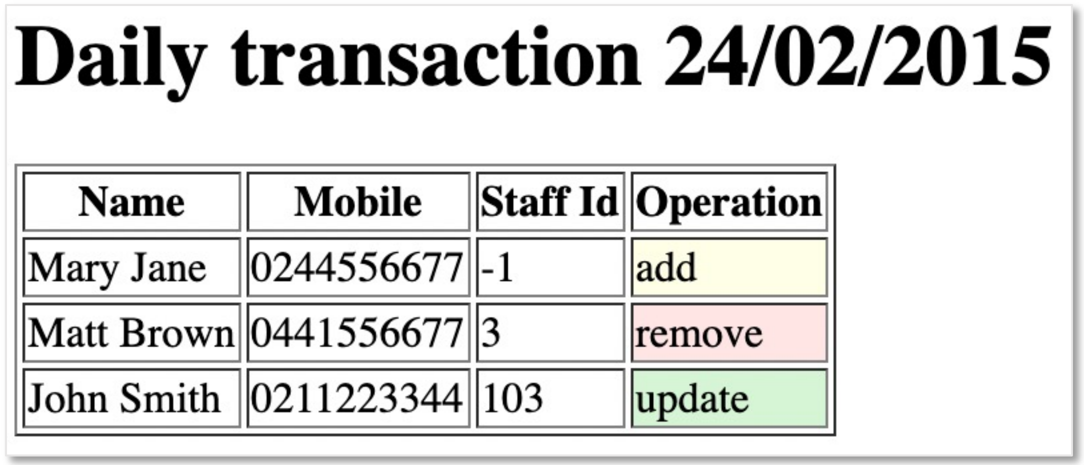


Daily transaction 24/02/2015

Name	Mobile	Staff Id	Operation
Mary Jane	0244556677	-1	add
Matt Brown	0441556677	3	remove
John Smith	0211223344	103	update

Example: Transaction v4

Now look at `transaction4.xml`.



Daily transaction 24/02/2015

Name	Mobile	Staff Id	Operation
Mary Jane	0244556677	-1	add
Matt Brown	0441556677	3	remove
John Smith	0211223344	103	update

Example: Transaction v4

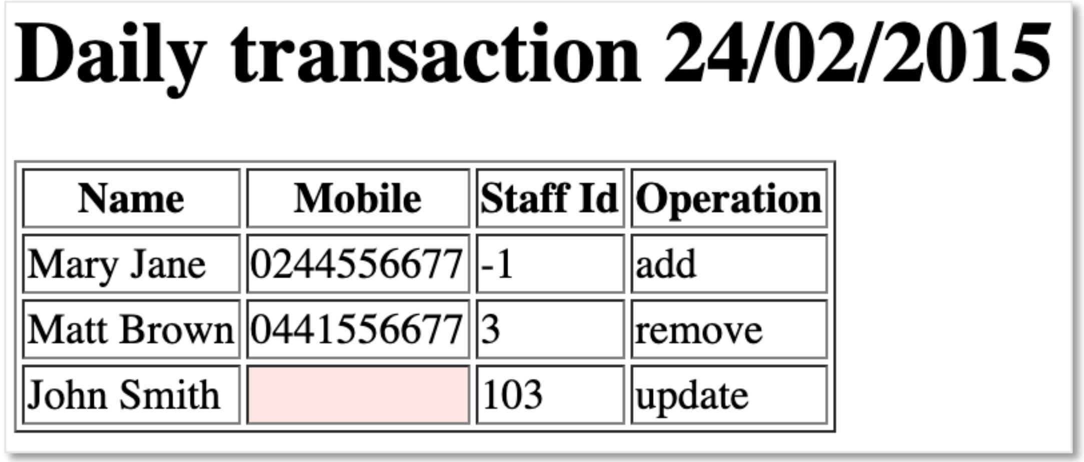
Now look at `transaction-style4.xsl`

```
 <xsl:choose>
    <xsl:when test="@operation = 'remove'">
        <td bgcolor="#ffe6e6">
            <xsl:value-of select="@operation" />
        </td>
    </xsl:when>
    <xsl:when test="@operation = 'add'">
        <td bgcolor="#ffffe6">
            <xsl:value-of select="@operation" />
        </td>
    </xsl:when>
    <xsl:otherwise>
        <td bgcolor="#d6f5d6">
            <xsl:value-of select="@operation" />
        </td>
    </xsl:otherwise>
</xsl:choose>
```

Example: Transaction v5

Now look at `transaction5.xml`.

```
…
<person staffDbId="103" operation="update">
    <firstName>John</firstName>
    <lastName>Smith</lastName>
    <mobile></mobile>
 </person>
…
```



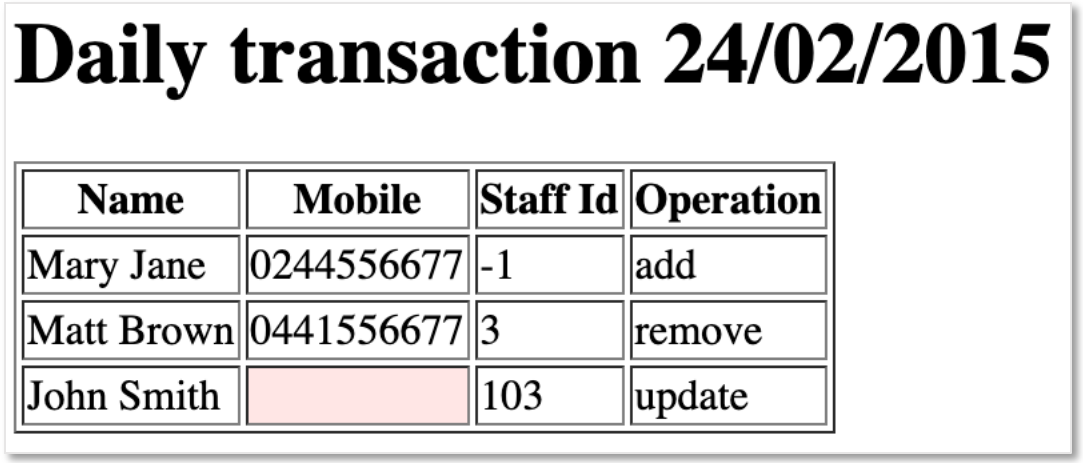
Daily transaction 24/02/2015

Name	Mobile	Staff Id	Operation
Mary Jane	0244556677	-1	add
Matt Brown	0441556677	3	remove
John Smith	🟥	103	update

Example: Transaction v5

Now look at `transaction5.xml`.

```
<xsl:choose>
 <xsl:when test="mobile = ''">
   <td bgcolor="#ffe6e6"> </td>
 </xsl:when>
 <xsl:otherwise>
   <td> <xsl:value-of select="mobile" /> </td>
 </xsl:otherwise>
</xsl:choose>
```

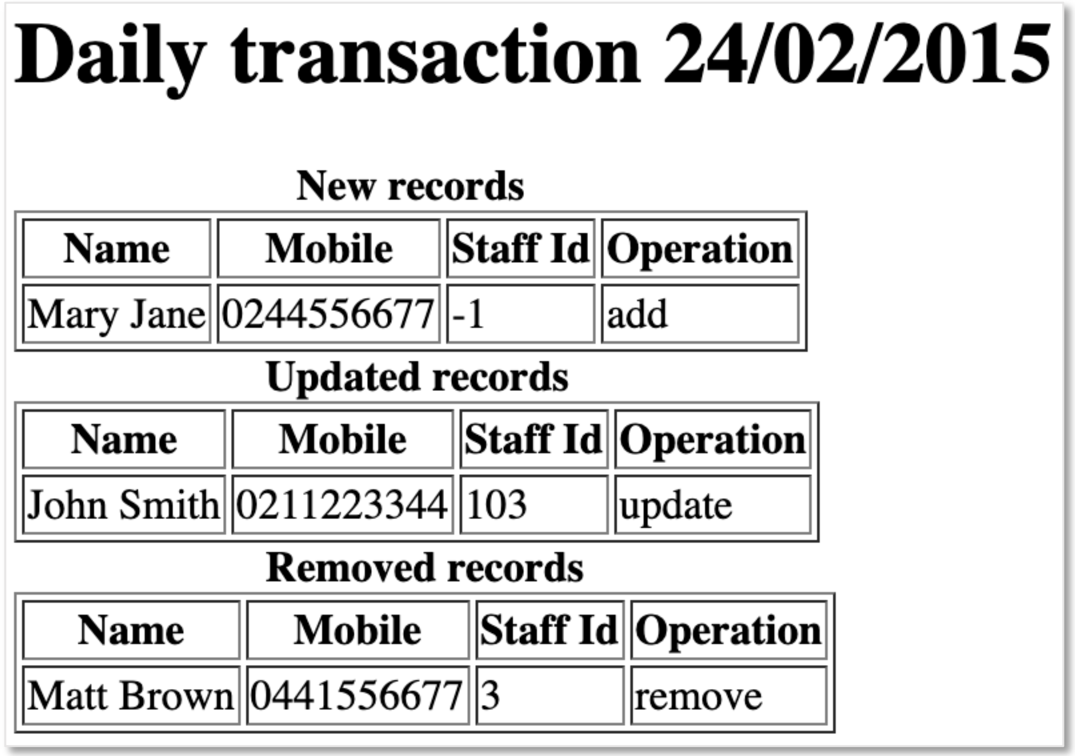


Daily transaction 24/02/2015

Name	Mobile	Staff Id	Operation
Mary Jane	0244556677	-1	add
Matt Brown	0441556677	3	remove
John Smith		103	update

Example: Transaction v6

Now look at `transaction6.xml`.



Daily transaction 24/02/2015

New records

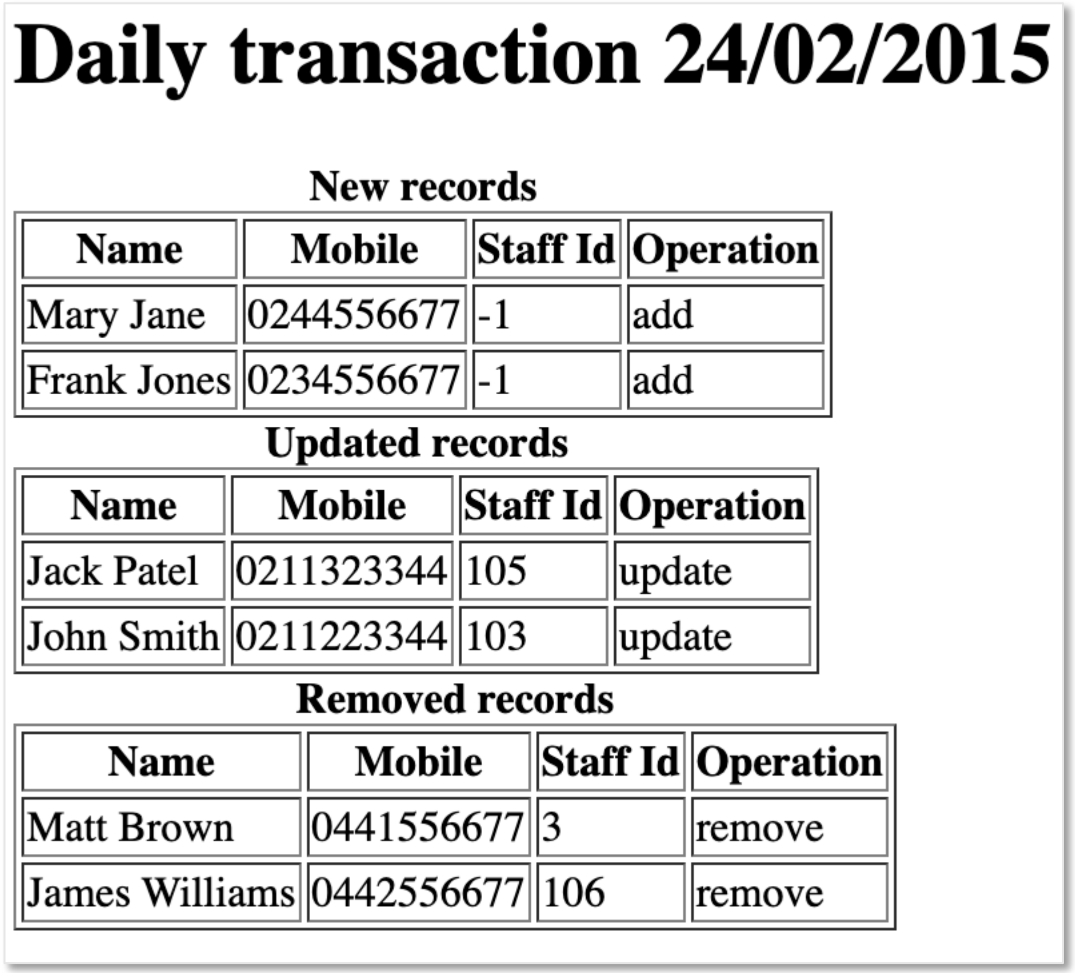
Name	Mobile	Staff Id	Operation
Mary Jane	0244556677	-1	add

Updated records

Name	Mobile	Staff Id	Operation
John Smith	0211223344	103	update

Removed records

Name	Mobile	Staff Id	Operation
Matt Brown	0441556677	3	remove



Daily transaction 24/02/2015

New records

Name	Mobile	Staff Id	Operation
Mary Jane	0244556677	-1	add
Frank Jones	0234556677	-1	add

Updated records

Name	Mobile	Staff Id	Operation
Jack Patel	0211323344	105	update
John Smith	0211223344	103	update

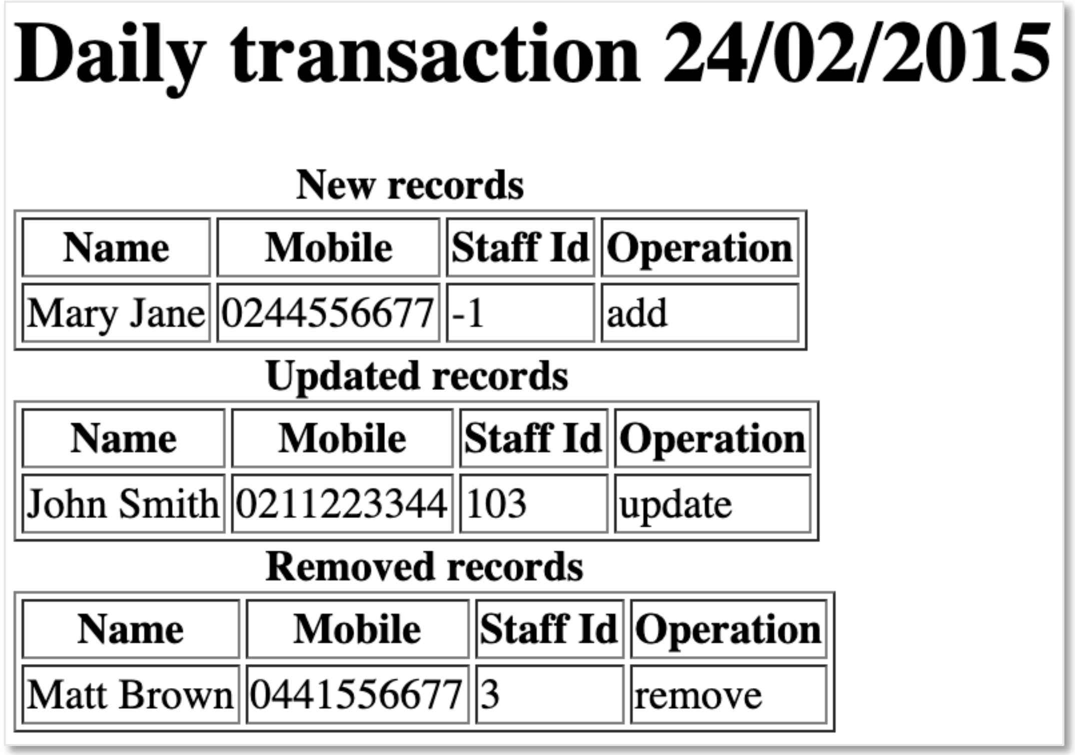
Removed records

Name	Mobile	Staff Id	Operation
Matt Brown	0441556677	3	remove
James Williams	0442556677	106	remove

Example: Transaction v6

Now look at `transaction-style6.xsl.`

```
<xsl:for-each select="person[@operation='add']">
 <xsl:sort select="lastName"/>
  <tr>
   <td>
    <xsl:value-of select="firstName" />
    <xsl:text> </xsl:text>
    <xsl:value-of select="lastName" />
   </td>
   <td>
    <xsl:value-of select="mobile" />
   </td>
   <td>
    <xsl:value-of select="@staffDbId" />
   </td>
   <td>
    <xsl:value-of select="@operation" />
   </td>
  </tr>
</xsl:for-each>
```



Daily transaction 24/02/2015

New records

Name	Mobile	Staff Id	Operation
Mary Jane	0244556677	-1	add

Updated records

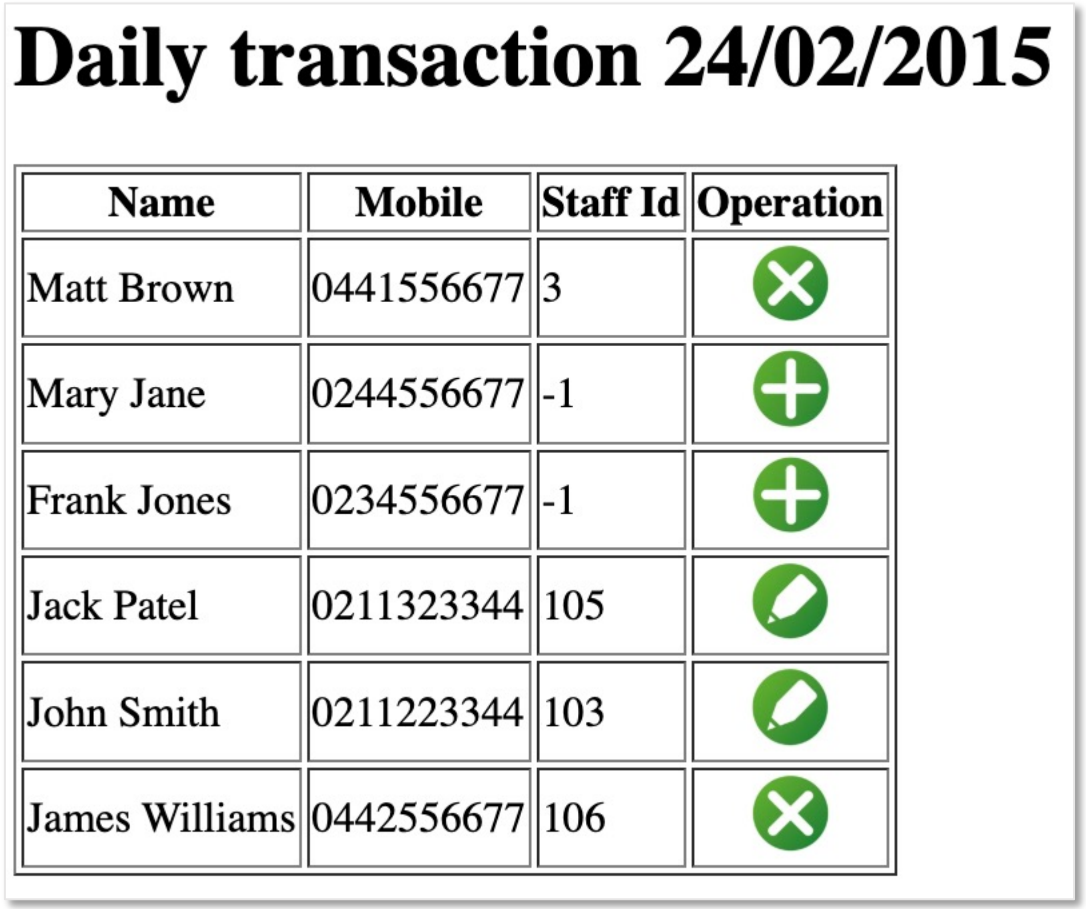
Name	Mobile	Staff Id	Operation
John Smith	0211223344	103	update

Removed records

Name	Mobile	Staff Id	Operation
Matt Brown	0441556677	3	remove

Example: Transaction v7

Now look at `transaction7.xml.`



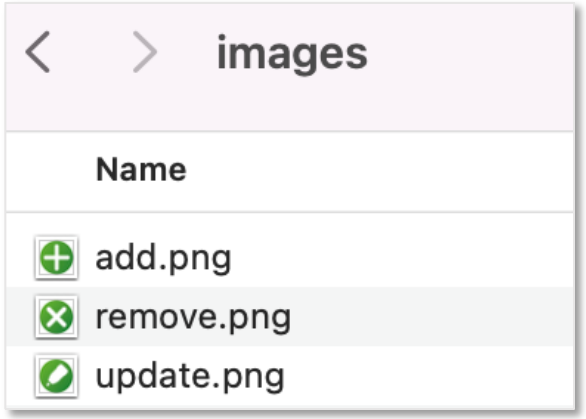
Daily transaction 24/02/2015

Name	Mobile	Staff Id	Operation
Matt Brown	0441556677	3	✖
Mary Jane	0244556677	-1	➕
Frank Jones	0234556677	-1	➕
Jack Patel	0211323344	105	✎
John Smith	0211223344	103	✎
James Williams	0442556677	106	✖

Example: Transaction v7

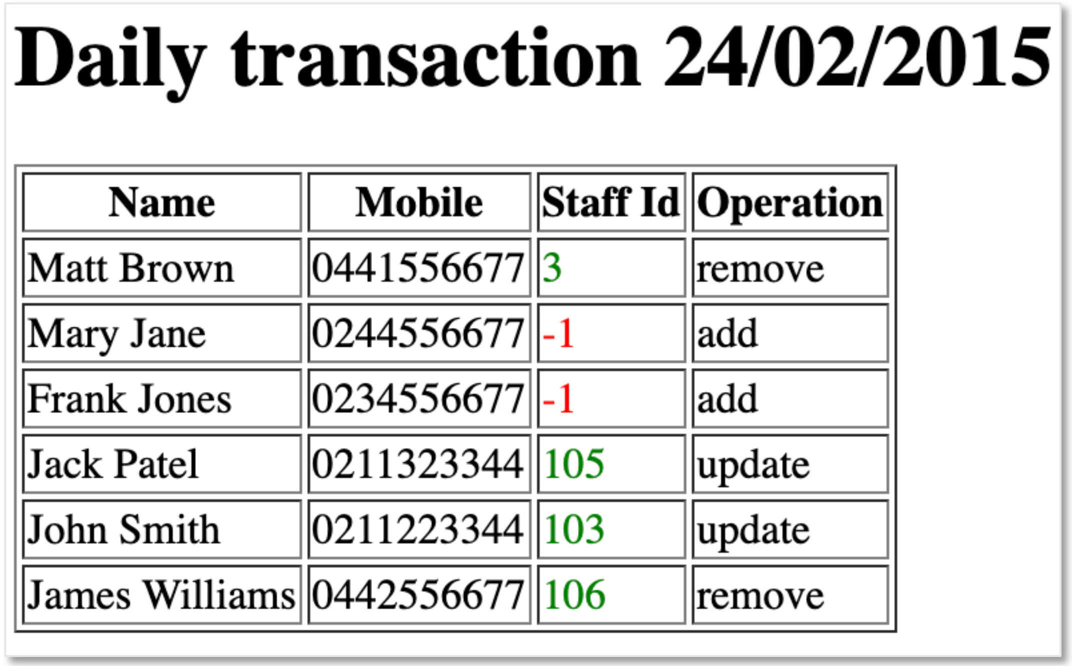
Now look at `transaction-style7.xsl`.

```
...
<td>
    <xsl:value-of select="@staffDbId" />
</td>
<td align="center">
    <img>
        <xsl:attribute name="src">
            <xsl:text>images/</xsl:text>
            <xsl:value-of select="@operation"/>
            <xsl:text>.png</xsl:text>
        </xsl:attribute>
        <xsl:attribute name="width">
            <xsl:text>30px</xsl:text>
        </xsl:attribute>
    </img>
</td>
...
```



Example: Transaction v8

Now look at `transaction8.xml`.



Daily transaction 24/02/2015

Name	Mobile	Staff Id	Operation
Matt Brown	0441556677	3	remove
Mary Jane	0244556677	-1	add
Frank Jones	0234556677	-1	add
Jack Patel	0211323344	105	update
John Smith	0211223344	103	update
James Williams	0442556677	106	remove

Example: Transaction v8

Now look at `transaction-style8.xsl.`

```
...
<td>
    <xsl:choose>
        <xsl:when test="@staffDbId &lt; 0">
            <span style="color:red">
                <xsl:value-of select="@staffDbId" />
            </span>
        </xsl:when>
        <xsl:otherwise>
            <span style="color:green">
                <xsl:value-of select="@staffDbId" />
            </span>
        </xsl:otherwise>
    </xsl:choose>
</td>
...
```

Summary

Get attribute value:

```
<xsl:value-of select="@attribute-name" />
```

Get element value:

```
<xsl:value-of select="element-name" />
```

Literal text:

```
<xsl:text>some text here ...</xsl:text>
```

Summary

FOR loop:

```
<xsl:for-each select="element-name">
 ...
</xsl:for-each>
```

FOR loop with sort:

```
<xsl:for-each select="element-name">
  <xsl:sort select="field-to-be-sorted"/>
  ...
</xsl:for-each>
```

FOR loop with filter:

```
<xsl:for-each select="element-name[filter-condition]">
  ...
</xsl:for-each>
```

Summary

IF statement: (note that there is no IF-ELSE)

```
<xsl:if test="the-if-condition">
  ...
</xsl:if>
```

Summary

CHOOSE-WHEN-OTHERWISE statement:

```
<xsl:choose>
 <xsl:when test="condition1">
  ...
 </xsl:when>
 <xsl:when test="condition2">
  ...
 </xsl:when>
 ...

 <xsl:otherwise>
  ...
 </xsl:otherwise>
</xsl:choose>
```

Summary

Mark = 49:

```
mark = 49
```

Mark not equal to 49:

```
mark != 49
```

Student type = ‘U’:

```
type = 'U'
```

Student type not equal to ‘U’:

```
type != 'U'
```

Mark > 35:

```
mark &gt; 35
```

Mark >= 85:

```
mark &gt;= 85
```

Mark < 35:

```
mark &lt; 35
```

Mark <= 85:

```
mark &lt;= 85
```

Mark NOT equal to 49:

```
not(mark = 49)
```

Student type = ‘U’ or ‘P’:

```
(type = 'U') or (type = 'P')
```

Mark >= 75 and mark < 85:

```
(mark &gt;= 75) and (mark &lt; 85)
```

References

- https://www.w3schools.com/xml/xsl_intro.asp
- https://developer.mozilla.org/en-US/docs/Web/XSLT