Table of Contents

1. An example of an XML document 1
1.1. Experiments 2
2. Why XML is helpful 3
3. What are some of the rules for xml data? 4
3.1. The version line 4
3.2. A well-formed document needs a root element. 4
3.3. Rules for tags 4
3.4. Element content 4
3.5. Elements and subelements 5
3.6. Attributes 5
3.7. Empty elements 6
4. Some of our limitations in this discussion 6
5. Using XML in Other Applications 6
6. xml Schema 7

XML is an important collection of technologies for the exchange of data. The XML format for data is not owned by MySQL or Oracle or Microsoft and is intended to be a way to store and manipulate structured data. We will look at a part of what Oracle can do with XML.

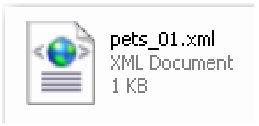
We are going to start with a demo and then follow with an explanation. Please do these steps- don't just read about them unless you already have a good grasp of XML. (XML is not the same as HTML.)

1. An example of an XML document

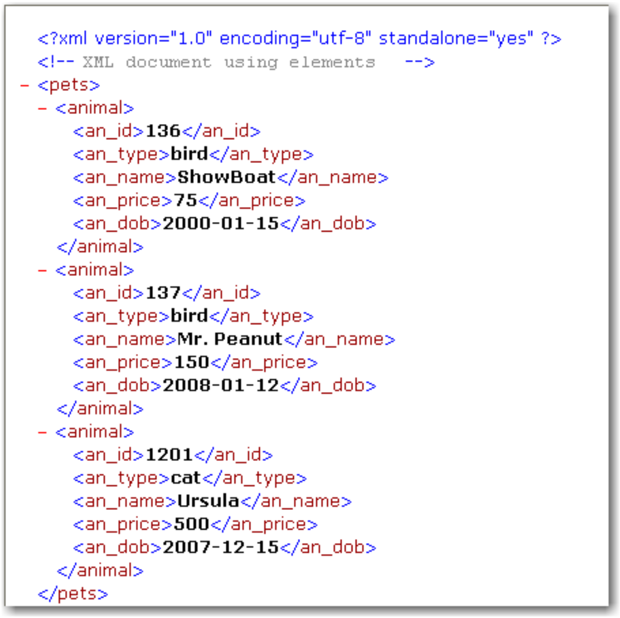
Copy the following lines and paste them into a text editor such as Notepad. Save the file with the name pets_01.xml.(pets_01.xml file is in the demos)

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<!--XML document using elements -->
<pets>
  <animal>
    <an_id>136</an_id>
    <an_type>bird</an_type>
    <an_name>ShowBoat</an_name>
    <an_price>75</an_price>
    <an_dob>2000-01-15</an_dob>
  </animal>
  <animal>
    <an_id>137</an_id>
    <an_type>bird</an_type>
    <an_name>Mr. Peanut</an_name>
    <an_price>150</an_price>
    <an_dob>2008-01-12</an_dob>
  </animal>
  <animal>
    <an_id>1201</an_id>
    <an_type>cat</an_type>
    <an_name>Ursula</an_name>
    <an_price>500</an_price>
    <an_dob>2007-12-15</an_dob>
  </animal>
</pets>
```

On a windows machine, the file icon probably looks like this



Right click this file and open it in your browser. Allowing Active X components to run, I get the following display in IE7 browser. The + and – symbols allow me to open or collapse the structure. You should get a similar display in other browsers.



```
  <?xml version="1.0" encoding="utf-8" standalone="yes" ?>
  <!-- XML document using elements   -->
- <pets>
  - <animal>
      <an_id>136</an_id>
      <an_type>bird</an_type>
      <an_name>ShowBoat</an_name>
      <an_price>75</an_price>
      <an_dob>2000-01-15</an_dob>
    </animal>
  - <animal>
      <an_id>137</an_id>
      <an_type>bird</an_type>
      <an_name>Mr. Peanut</an_name>
      <an_price>150</an_price>
      <an_dob>2008-01-12</an_dob>
    </animal>
  - <animal>
      <an_id>1201</an_id>
      <an_type>cat</an_type>
      <an_name>Ursula</an_name>
      <an_price>500</an_price>
      <an_dob>2007-12-15</an_dob>
    </animal>
  </pets>
```

Most of you are probably at least somewhat familiar with html and this looks a bit like html- but it isn't- I don't have any of the html tags that you may know except for the comment tag.

This is a simple XML document. The first thing to notice is that this is all text and except for the first line it is humanly-readable and understandable.

It looks like we are storing data about animals and animal have an id, a name, a type, a price and a date of birth. If I asked you to include data about a dog (named Blue who costs 450 with ID of 834 born on April 3 2001) you could add that data easily. Probably you would copy one set of the animal lines and edit it because you can see that this document has a structure.

- The words inside the angle brackets are called tags.
- This document includes data values that are organized/structured.
- We have three animals- the first animal is a bird named ShowBoat and the third is a cat named Ursula.
- For each beginning tag we have an ending tag which includes a /: <an_name>…</an_name>
- The tags name the data components.
- We have data values between the tags
- The tagged elements can be nested- the <an_name>is inside the <animal> and all of the animals are inside <pets> Note the ending tag for pets </pets> at the bottom.

1.1. Experiments

Do a few experiments with this file. (Really- do this- it will save you a lot of time and frustration if you try these out now.)You should be able to keep the editor open with the file and your browser open. Then make changes to the file, save it and refresh your browser. For each of these, do a single change and if it causes an error- fix the error before going to the next step.

- Try adding another animal to this file. Follow the pattern used
- Change the starting tag <an_id> for one of the rows to <an_ID> but don't change its ending tag.
- Remove one of the ending tags.
- Remove one of the an_id lines completely.
- Put a single blank space at the front of one of the tags: < an_id>
- Change one of the dob values to something that is not a valid date.

From this you can see that xml is fussier about syntax than SQL. And a lot fussier than html!

2. Why XML is helpful

There are a lot of different ways to store data in a computer system. We could use csv (comma separated values) files, fixed length record files, spreadsheets, relational database tables, and xml. XML is another format for storing data that also stores information about the structure of the data.

One of the advantages of storing data in a csv file is that the data is in text format and therefore generally readable by different computer systems. But a csv does not include metadata. You can figure out the following lines of a csv file because it stores the data we have been talking about- but the file itself does not store the metadata.

```
bird,ShowBoat,136,75,2000-01-15
bird,Mr. Peanut,137,150,2008-01-12
cat,Ursula,1201,500,2007-12-15
```

One problem with storing data in a relational database system is that the actual format of the data is proprietary and not always sharable across systems. Believe it or not, there are some people who do not have access to an Oracle database and we might want to share data with those people.

XML data is commonly used to transfer data across different computer systems. XML is all text so it does not use proprietary data formats. People who need to share data could agree on the tag names to use and the organization of the tags for the data – this set of rules for the organization of the data is called an xml schema- the xml schema is written in xml. Many industries and fields of study have set up rules/schema for XML data and can share data more easily. XML is extensible in that you can define more tags as needed.

XML dates from 1996 and is part of W3C; XML is not owned by Microsoft, by Oracle or by any company. W3C provides some standards for XML; companies generally try to meet those standards but if a standard cannot be agreed on, then a company might implement its own version of an XML feature

XML is hierarchical. With a relational table, the order of the rows cannot be logically significant; with XML data the order of the lines can be significant. With relational data, we often store data in two or more tables with relationships between them. With XML data you are more apt to have a single document with the relational structure changed into a hierarchical structure. So there is not a complete matchup between relational data and xml data.

If we wanted to store data about a client and their pets, we would create two relational tables. In XML we might use the following. The root is now <client_list>. We have three <client> elements. Each client element includes a <pets> element. The first client has two <animal> elements in their <pets> element; the second client has one animal in their <pets> element; the third client has an empty <pets> element. It is possible with the XML structure to say that ShowBoat is the first animal for client 8243 and Mr. Peanut is the second. The ordering of the subelements in an XML document can be logically determined. (client_pets_01.xml)

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<!--XML document using elements -->
<client_list>
  <client>
    <cl_id>8243</cl_id>
    <cl_name>Johnson</cl_name>
    <pets>
      <animal>
        <an_id>"136"</an_id>
        <an_type>bird</an_type>
        <an_name>ShowBoat</an_name>
        <an_price>75</an_price>
        <an_dob>2000-01-15</an_dob>
      </animal>
      <animal>
        <an_id>"137"</an_id>
        <an_type>bird</an_type>
        <an_name>Mr. Peanut</an_name>
        <an_price>150</an_price>
        <an_dob>2008-01-12</an_dob>
      </animal>
```

```
        </pets>
      </client>
      <client>
        <cl_id>3908</cl_id>
        <cl_name>Nelson</cl_name>
        <pets>
          <animal>
            <an_id>"1201"</an_id>
            <an_type>cat</an_type>
            <an_name>Ursula</an_name>
            <an_price>500</an_price>
            <an_dob>2007-12-15</an_dob>
          </animal>
        </pets>
      </client>
      <client>
        <cl_id>3775</cl_id>
        <cl_name>Miller</cl_name>
        <pets/>
      </client>
    </client_list>
```

3. What are some of the rules for xml data?

Syntax rules (simplified)

3.1. The version line

The first line of an XML document is generally

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

This is a declaration statement that says you are going to follow the syntax rules for XML 1.0.

Note that the declaration tag delimiters includes a ? character. Do not put whitespace before this statement.

3.2. A well-formed document needs a root element.

The declaration is not enough to have a well-formed document. You need a root element.
In the first document <pets > </ pets > is the root element. You could have an empty element; you can have a well formed xml document that consists of just an empty root element.

```
<?xml version="1.0" encoding="utf-8" ?>
<pets></pets>
```

<pets> is the start tag . </pets> is the end tag

3.3. Rules for tags

- Every element needs a start tag and an end tag; the end tag is identical to the start tag except that it includes a slash.
- XML element names are case specific.
- The element name cannot start with a space: <   animal> is not acceptable; the element name can end with a space <animal   > is OK. Do not put spaces inside an element name.

3.4. Element content

We can add content to the element.

```
<?xml version="1.0" encoding="utf-8" ?>
<pets>
This is simple!
</pets>
```

In our longer example the content is in the sub elements, but you can have content in any element.

3.5. Elements and subelements

XML is hierarchical in nature; XML documents have a tree structure. An XML element can contain text or another element (a child element or a sub element). If you are storing structured data, you will probably have sub elements. Here the root element name is animal and the child element names are an_name and an_type.

```
<?xml version="1.0" encoding="utf-8" ?>
<animal>
  <an_name>Jayson</an_name>
  <an_type>bird</an_type>
</animal>
```

- A well-formed document has only one root element.
- Subelements must be completely contained within their parent element.
- Indentation is helpful for a human reader. Most application programs can be told to ignore whitespace.

Suppose we have two animals. This is not a well-formed document. It does not have an enclosing root element. The error message is generally expressed as the document having multiple root elements.

```
<?xml version="1.0" encoding="utf-8" ?>
<animal>
  <an_name>Tweetie</an_name>
  <an_type>bird</an_type>
</animal>

<animal>
  <an_name>Sylvester</an_name>
  <an_type>cat</an_type>
</animal>
```

We do have a name for a piece of a document; an **XML fragment**. It is well formed with the exception that it does not have a root element.

3.6. Attributes

You can use attributes – name=value pairs – for data values. You can use single or double quotes; it is better style to be consistent in a document. You can include white space around the = operator. (pets_02.xml)

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<!--XML document using elements -->
<pets>
  <animal an_id="136">
    <an_type>bird</an_type>
    <an_name>ShowBoat</an_name>
    <an_price>75</an_price>
    <an_dob>2000-01-15</an_dob>
  </animal>
  <animal an_id="137">
    <an_type>bird</an_type>
    <an_name>Mr. Peanut</an_name>
    <an_price>150</an_price>
    <an_dob>2008-01-12</an_dob>
  </animal>
  <animal an_id="120">
    <an_type>cat</an_type>
    <an_name>Ursula</an_name>
    <an_price>500</an_price>
    <an_dob>2007-12-15</an_dob>
  </animal>
</pets>
```

- You cannot have two attributes with the same name in the same element.
- If an attribute is storing a numeric value, the value is still entered as a text value enclosed in quotes. Your application may need to cast that string to a numeric value.

There is no general agreement about which items to store as attributes and which as subelements. Often an item which is an identifying item is handled as an attribute.

There are some situations where you have to use subelements:

- Any item which may be repeated has to be handled as an element.
- Any item which is itself complex and needs to be broken down into pieces should be handled as an element.
- If the order of the values is important, then they need to be handled as elements.

If you are used to writing HTML you may be in the habit of using attributes a lot:

```
<td class="body_title" colspan="2">
```

That makes sense for display attributes but it is not as useful for XML and data. When in doubt, use elements rather than attributes for data.

3.7. Empty elements

An element does not have to contain any content

- Empty elements can be written in either of two formats:

```
<an_name></an_name>
<an_name/>
```

4. Some of our limitations in this discussion

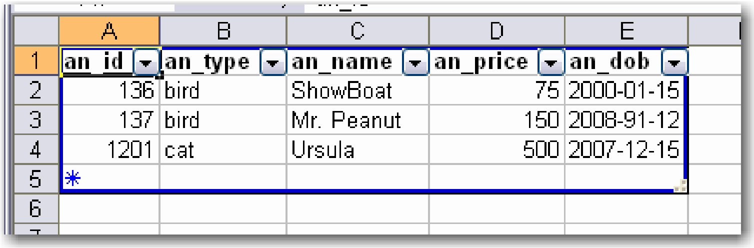
Because we are spending only one week on this topic, we cannot discuss everything about XML. Some things we will not work with:

- creating and using xml schemas
- combining xml and html
- all of the functions that Oracle provides for xml data; we will talk about some functions
- all of the features of XML Query; we will look at some XPath features

5. Using XML in Other Applications

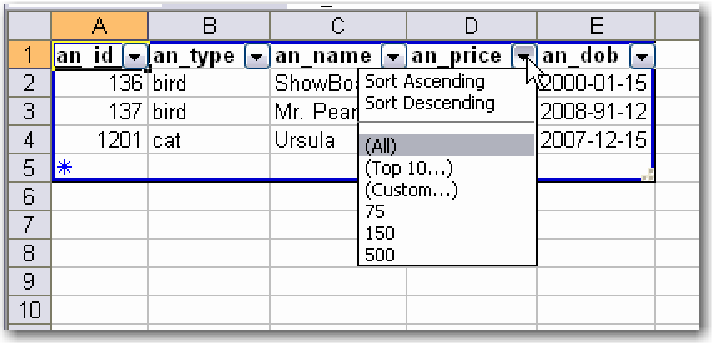
Many applications today include the ability to handle xml data.

Take one of the XML documents; open it in Excel. This uses the choice to open the XML file as an XML list.



	A	B	C	D	E
1	an_id	an_type	an_name	an_price	an_dob
2	136	bird	ShowBoat	75	2000-01-15
3	137	bird	Mr. Peanut	150	2008-91-12
4	1201	cat	Ursula	500	2007-12-15
5	*				
6					

Excel provides sorting options



	A	B	C	D	E
1	an_id	an_type	an_name	an_price	an_dob
2	136	bird	ShowBo		2000-01-15
3	137	bird	Mr. Pear		2008-91-12
4	1201	cat	Ursula		2007-12-15
5	*				

6. xml Schema

If you have a copy of Visual Studio 2008 available, create a project; add an xml file to the project and copy in your xml data. Then tell VS to generate an xml Schema. This is the generated schema for our file. You can see that it describes the file and seems to get the data types correct.

The XML Schema file can be used to check that the data that you are inserting into the XML column is valid according to data types. Oracle can be told to associate an XMLS schema with the XML data column.

The word "schema" has several different meanings- in a discussion of xml and xml schemas it refers to the file that describes the structure of the xml document.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
          xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="pets">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" name="animal">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="an_id" type="xs:unsignedShort" />
              <xs:element name="an_type" type="xs:string" />
              <xs:element name="an_name" type="xs:string" />
              <xs:element name="an_price" type="xs:unsignedShort" />
              <xs:element name="an_dob" type="xs:date" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Examples of the kinds of validation that can be specified by a schema include:

- The order in which the individual sub elements must appear
- Whether or not an element is optional
- The minimum and maximum number of times an element can be repeated.
- Data type- such as date, integer, string

More complex validation can include

- Range tests
- List tests
- Limitations on the length of a string

A column in a table, a variable, or a value returned by a function can be associated with a schema- that xml would be called typed XML. XML that is not associated with a schema is called untyped XML. If you have a typed xml object and you try to assign an xml value that does not meet the schema rules, that assignment would be rejected.