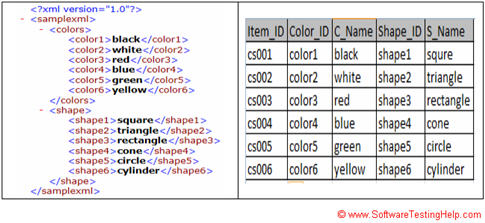
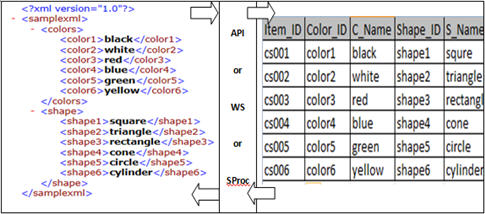
To make the XML to Database testing successful, the most crucial input is the **mapping document** that defines each element in the XML versus the columns in the database.

XML is a reliable technique to communicate messages (Data) between two applications. XML contains set of elements that have specific values. Sometimes the values may be NULL or blank.

Database stores data in the form of tables. A database contains several tables. An application can feed data into the table in a database and also the table data can be fetched by applications when required.



n some cases the upstream and downstream applications can be databases of different applications and they are communicating/transmitting data in XML format using Stored Procedures, Web services, APIs, etc. In others there may be a combination of databases and applications that are communicating data with each other.

[](http://cdn.softwaretestinghelp.com/wp-content/qa/uploads/2016/01/XML-Vs-Data-Testing-2.jpg)

First of all you take the input XML file and [validate the XML structure](http://www.xmlvalidation.com/)i.e., elements. This can be done with the help of XSD which defines the structure for the respective XML.

------------

The XSD file looks like XML and it defines the structure of XML, like element name, element type, minOccurs, maxOccurs, etc. Once the XML validation is done, export it to excel. Just drag the xml file to a new excel sheet. It will give you a popup asking how you want to open the file, just select ‘As an XML table’. The data will save into the excel file as table.

You can see data populated into the table, query the table with the particular data and fetch the record. Copy the data to the same excel file to another sheet. Now using EXACT function in excel you can easily compare the XML data vs DB data. Make sure you will compare only data not the column names.

In this way you can compare multiple record data and **can save a lot of manual effort** for comparing XML element data values vs DB column data values.

**Excel file comparison should be limited to 100MB file size**. You will encounter performance issues if you go beyond.

**Failure scenarios:**

Let’s discuss about some possible failure scenarios.

* One simple failure scenario is **incorrect mapping**. The mapping between the XML elements vs DB columns should be analyzed during the analysis or planning phase by a tester. Discuss all the mapping concerns with BA/SA to clarify doubts. Once the mapping is frozen, you can ensure that the XML elements vs DB columns values would match.
* Compare the values and if it does not match, log a defect to address the issue. There are number of possibilities for the defect raised, like Data defect – May be the **test data issue**; Code defect – May be the bug in the code which parses the data values to not map; Artifact defect – May be incorrect mapping provided by BA/SA.
* **XML format issue –** XML header or metadata or some incorrect xml tags. In this case the XML itself failed to store the data values into the database table.
* **Datatype mismatch –** Element value in the XML is having more char in length which is more than the DB column can accept. This will be a code issue and dev team has to make necessary changes in the datatype length for that column.
* **Environment failure –** Environment down or DB application down, the data flow remains incomplete.
* **Performance issue –** May be the amount of records consisting the message is huge or the load on the DB could be high to be begin with record consist is too large.
* **Middleware failure** will cause the data flow letdown from application to database.
* **Database access issue** due to which the inbound application is unable to send the data to the respective table.

| **Life Cycle stage** | **Entry Criteria** | **Activities / Responsibility** | **Exit Criteria** |
| --- | --- | --- | --- |
| Identify test data requirements | Following documents are available  Database design, UI design, requirement specification, technical architecture, data flow diagram, Use case diagrams | Understand the data requirements referencing the documents from entry criteria (QA, DT, DEV)  Test data requirements (QA, DT, DEV) - Documents all data needs for every screen showing a mapping between screen display names and corresponding XML |  |

#### 3. Build the data

| **LC stage** | **Entry Criteria** | **Activities / Responsibility** | **Exit Criteria** |
| --- | --- | --- | --- |
| Build data | Data request file | Build the data in the DB (DT)  Extract the data from the DB into the XML (DT)  Validate the XML against Schema (DT)  Share the XML file with QA (DT) |  |

#### 4. Test the data

| **LC stage** | **Entry Criteria** | **Activities / Responsibility** | **Exit Criteria** |
| --- | --- | --- | --- |
| Test the data | Data request XML file | Validate the XML against schema for completeness and correctness (QA)  Update the mapping document with test results (QA) | Test results shared with DEV, DT |

Extract, Transform, Load) tool. As the name suggests, it performs the following three operations −

* Extracts the data from your transactional system which can be an Oracle, Microsoft, or any other relational database,
* Transforms the data by performing data cleansing operations, and then
* Loads the data into the OLAP data Warehouse.

The term "**schema**" refers to the organization of data as a blueprint of how the**database** is constructed (divided into **database** tables in the case of relational**databases**).