RELATIONAL MODEL VS XML

It is important to answer the question when to use XML and when to use a purely relational model. The simplest answer is that it depends on situation. Wow, there are really big word.s Nobody can get anything meaningful from your answer. While another obvious answer is “use XML when working with XML data,” that approach is oversimplified and consequently can be wrong. In many cases, it is advantageous to use an XML data representation even for non-XML data; similarly, there are times when using an XML data representation for XML data is not the correct choice.

First, let’s see a brief comparison between relational model and XML.

	XML	Relational model
Flexibility	Flexible	Fairly rigid
Storage structure	Hierarchical	Flat (Relational table)
Retrieval	Directly	Indirectly
Self-describing	Yes	No
Ordering	Ordering inherent	Ordering not inherent
Referential integrity	Part of	Yes
Schema	Schemaless	Schema

The bulk of the influence of XML on logical and physical database design is based on fundamental properties of XML that make it different from the relational model:

- XML is self-describing.

A given document contains not only the data, but also the necessary metadata. As a result, an XML document can be searched or updated without requiring a static definition of the schema. Relational models, on the other hand, require more static schema definitions. All the rows of a table must have the same schema.

- XML is hierarchical.

A given document represents not only base information, but also information about the relationship of data items to each other in the form of the hierarchy. Relational models require all relationship information to be expressed either by primary key or foreign key relationships or by representing that information in other relations.

- XML is sequence-oriented—order.

For an XML document, the order in which data items are specified is assumed to be the order of the data in the document. There is often no other way to specify order within the document. For relational data, the order of the rows is not guaranteed unless you specify an ORDER BY clause on one or more columns.

Sometimes the nature of the data dictates the way in which you store it. For example, if the data is naturally hierarchical and self-describing, you might store it as XML data. However, other factors might influence your decision about which model to use.

Some of those factors are:

- Whether maximum flexibility of the data is needed

Relational tables are fairly rigid. For example, normalizing one table into many or denormalizing many tables into one can be very difficult. If the data design changes often, representing it as XML data is a better choice.

- Whether the data components have meaning outside a hierarchy

Data might be inherently hierarchical in nature, but the child components do not need the parents to provide value. For example, a purchase order might contain part numbers. The purchase orders with the part numbers might be best represented as XML documents. However, each part number has a part description associated with it. It might be better to include the part descriptions in a relational table, because the relationship between the part numbers and the part descriptions is logically independent of the purchase orders in which the part numbers are used.

- Whether data attributes apply to all data, or to only a small subset of the data

Some sets of data have a large number of possible attributes, but only a small number of those attributes apply to any particular data value. For example, in a

retail catalog, there are many possible data attributes, such as size, color, weight, material, style, weave, power requirements, or fuel requirements. For any given item in the catalog, only a subset of those attributes is relevant: power requirements are meaningful for a table saw, but not for a coat. This type of data is difficult to represent and search with a relational model, but relatively easy to represent and search with an XML model.

- Whether the data needs to be updated often

Currently, you can update XML data in an XML column only by replacing full documents. If you need to frequently update small fragments of very large documents for a large number of rows, it can be more efficient to store the data in non-XML columns. If, however, you are updating small documents and only a few documents at a time, storing as XML can be efficient as well.

Source: http://toyhouse.cc/profiles/blogs/relational-model-vs-xml