Subject: Re: Traversing subtrees
Posted by DannyBoyPoker on Sat, 06 Apr 2013 01:22:34 GMT
View Forum Message <> Reply to Message

The table isn't normalized, although, okay, let's step through this..this is stuff on the plate, to get into, in general, about data. I guess I find it interesting, as stuff I am supposed to know..let's see if I can thread this needle..

You point out, that there may be other ways to handle tree structures, which, okay, sure, but that would be an implementation detail. On the other hand, a tree is a collection of nodes and edges, with rules dictating the connection among the nodes. Actually, there are numerous rules spelling out how these connections can occur. Consider, for example, if you want to enter an itinerary, such as gym, restaurant, work, restaurant, home. You can't. What is restaurant's parent? But, I can.

Like this:
1 gym
2 restaurant
3 work
4 home

And, in the 'edges' table:
Parent Child
1 2
2 3
3 2
2 4

So, here, a node can have multiple parents, as it can have multiple children.

You ask: 'Why do you think that a record with a null parent_id is not normalised?'
Well, one argument against nulls is that they don't have a well-defined interpretation.

To this, you offer that the data means exactly what it says and there's no duplication. Suppose that we want multiple roots, well, one may construe this as a single tree with a null root. Our multiple roots are children of a null parent. Suppose that I am demanding that all values and data types should have well-defined interpretations, therefore all nulls are bad. In this case, then, I have been answered. It's easy and it's normalized...I mean, well, it's easy anyways, it's a pretty straightforward design that's easily understood by everyone, in which case, no deep relational theory needed.

Beyond that, well, the topic of hierarchies has been beaten with an ugly stick, I suppose.

'My tree structure is not supposed to be the only way to represent a tree, it just happens to be the one that suited my purposes many years ago when a particular requirement arose.'

We're doing set operations. That, is what gives you mathematical control--control. It's not a matter of what is my purpose, my strategy. It's a matter of control. Here is my particular requirement: I

want to model a collection of 'things', and their relationship among these 'things'. Sets may be structured, or not. If they are structured, then they may form a tree, or other structure. Sets can be expressed within tree structures in a number of ways. How? Well, sets are groups of resources that share a common property.That property may be a container. And tree structures are largely defined by containment relationships.

There is a formal foundation, here. Database theory. Maybe I can demonstrate its practical value, and maybe I can't, but there are the formal foundations of a set-theoretic data model. All data representations are treated
as mathematical objects. Maybe it's an important approach, and maybe it isn't.