How To Move A Node In Nested Sets With SQL

By Roger Keays, 8 January 2012

Moving nodes in nested sets is a complex operation. There were a few solutions on Stack Overflow for moving a node under a given parent, however this doesn't let you select the position of the node amongst it's siblings.

This solution lets you move a node to any position in the tree, with just a single input parameter - the new left position (newpos) of the node.

Fundamentally there are three steps:

1. Create new space for the subtree.
2. Move the subtree into this space.
3. Remove the old space vacated by the subtree.

In psuedo-sql, it looks like this:

```
/**
 *  -- create new space for subtree
 *  UPDATEtags SETlpos = lpos + :width WHERE lpos >= :newpos
 *  UPDATEtags SETrpos = rpos + :width WHERE rpos >= :newpos
 *
 *  -- move subtree into new space
 *  UPDATE tags SET lpos = lpos + :distance, rpos = rpos + :distance
 *          WHERE lpos >= :tmppos AND rpos < :tmppos + :width
 *
 *  -- remove old space vacated by subtree
 *  UPDATEtags SETlpos = lpos - :width WHERE lpos > :oldrpos
 *  UPDATEtags SET rpos = rpos - :width WHERE rpos > :oldrpos
 */
```

The **:distance** variable is the distance between the new and old positions, the **:width** is the size of the subtree, and **:tmppos** is used to keep track of the subtree being moved during the updates. These variables are defined as:

```
// calculate position adjustment variables
int width = node.getRpos() - node.getLpos() + 1;
int distance = newpos - node.getLpos();
int tmppos = node.getLpos();

// backwards movement must account for new space
if (distance < 0) {
    distance -= width;
    tmppos += width;
}
```

Here is some sample code of how this is done with OrmLite.

```
public void move(Node node, int newpos) {
    try {
        refresh(node);

        // calculate position adjustment variables
        int width = node.getRpos() - node.getLpos() + 1;
        int distance = newpos - node.getLpos();
        int tmppos = node.getLpos();

        // backwards movement must account for new space
        if (distance < 0) {
            distance -= width;
            tmppos += width;
        }

        // create new space for subtree
        UpdateBuilder update1 = updateBuilder();
        update1.updateColumnExpression("lpos", "lpos + " + width);
        update1.where().ge("lpos", newpos);
        update(update1.prepare());

        UpdateBuilder update2 = updateBuilder();
        update2.updateColumnExpression("rpos", "rpos + " + width);
        update2.where().ge("rpos", newpos);
        update(update2.prepare());

        // move subtree into new space
```

```
            UpdateBuilder update3 = updateBuilder();
            update3.updateColumnExpression("lpos", "lpos + " + distance);
            update3.updateColumnExpression("rpos", "rpos + " + distance);
            update3.where().ge("lpos", tmppos)
                   .and().lt("rpos", tmppos + width);
            update(update3.prepare());

            // remove old space vacated by subtree
            UpdateBuilder update4 = updateBuilder();
            update4.updateColumnExpression("lpos", "lpos - " + width);
            update4.where().gt("lpos", node.getRpos());
            update(update4.prepare());

            UpdateBuilder update5 = updateBuilder();
            update5.updateColumnExpression("rpos", "rpos - " + width);
            update5.where().gt("rpos", node.getRpos());
            update(update5.prepare());

            // refresh local data
            refresh(node);
            getObjectCache().clearAll();
        } catch (SQLException e) {
            Log.e("Error moving node", e.getMessage());
        }
    }
```

Hope that helps you.

About Roger Keays

Roger Keays is an artist, an engineer, and a student of life. He has no fixed addressand has leftfootprints on 40-something different countries around the world. Roger is addicted to surfing. His other interests are music, psychology, languages, the proper use of semicolons, and finding good food.