```
#include <iostream>
using namespace std;
const int N = 3e5 + 5;
struct node
{
    int l, r;
    int v;
} tr[4*N];
void pushdown(int u){
    if(  tr[u] ){
        {
            tr[u<<1]=;
            tr[u<<1|1]=;
        }
    }
}
void pushup(node &root, node l, node r)
{

}

void pushup(int u)
{
    pushup(tr[u], tr[u << 1], tr[u << 1 | 1]);
}

void build(int u, int l, int r)
{
    tr[u].l = l, tr[u].r = r, tr[u].v = ;
    if (l == r)
    {
                 tr[u].v =
    }
    int mid = l + r >> 1;
    build(u << 1, l, mid), build(u << 1 | 1, mid + 1, r);
    pushup(u);
}
node query(int u, int l, int r)
{
    if(l>r) {
        node re={0,0,0};
        return re;
    }
    if (tr[u].l >= l && tr[u].r <= r)
        return tr[u];
    pushdown(u);
    int mid = tr[u].l + tr[u].r >> 1;
    node ans = {0, 0, 0};
    if (mid < l)
    {
        node lef=query(u << 1|1, l, r);
        pushup(u);
        return lef;
    }
    if (mid >=r )
    {
        node rig=query(u << 1 , l, r);
        pushup(u);
        return rig;
    }
    node left = query(u << 1, l, r);
    node right = query(u << 1 | 1, l, r);
    pushup(u);
    pushup(ans, left, right);
    return ans;
}
void modify(int u, int l, int r)
{
    if (tr[u].l >= l && tr[u].r <= r)
    {
        tr[u]=;
        return;
    }
    pushdown(u);
    int mid = tr[u].l + tr[u].r >> 1;
    if (mid >= l)
        modify(u << 1, l, r);
    if (mid < r)
        modify(u << 1 | 1, l, r);
    pushup(u);
}
int main()
{
    cin >> n >> q;
    build(1, 1, n);

    while (q--)
    {
        int op, l, r;
        cin >> op >> l >> r;
        if (op == 1)
        {
            modify(1, l, r);
        }
        else
        {
            node ui = query(1, l, r);
            cout << min(ui.kt0, ui.kt1) << '\n';
        }
    }
    return 0;

}
```