E. Brief Statements Union

题意

给定 k 个限制，每个限制形如 $a_l\And a_{l+1}\And a_{l+2}\cdots \And a_r=x$

现询问对每个限制 i如果删除该限制是否存在长度为 n 的序列 a 满足所有其他限制。

题解

考虑枚举考虑每个位的答案，然后合并结果得到总答案。

对每个位，限制转化为两种。第一种对应区间覆盖操作，将对应区间置 1，第二种操作对应区间查询操作，查询区间是否存在 0。

首先执行所有第一种操作，对每个第二种操作再分成两类，一类为当前已经满足的，一种是当前不能满足的。

如果不存在当前不能满足的第二类操作，显然删除任意一条限制都可以找到序列满足其他约束。

如果只存在一个不能满足的第二类操作，则删除其他第二类限制显然是找不到序列满足约束的。

如果只存在多于一个不能满足的第二类操作，则删除任意一个第二类限制都是找不到序列满足约束的。

再考虑能否通过删除某个第一类操作找到需要序列满足约束。考虑每个第一类操作唯一覆盖的区间，即该区间仅被一个第一类操作覆盖。

删除这个操作能且仅能使这个区间的权值变为 0。于是问题转化为查询所有唯一覆盖区间与所有不能满足的第二类操作区间都相交的第一类操作。

上述问题可以 $O(n)$ 维护，因此总时间复杂度 $O(n\log v)$

```
const int MAXN=1e6+5,MAXK=1e6+5,MAXV=63;
int l[MAXK],r[MAXK],l2[MAXK],r2[MAXN],c[MAXN];
LL w[MAXK],s[MAXN];
bool ans[MAXK],ok[MAXK];
vector<int> vec1,vec2;
void solve(int n){
    vector<int> vec3;
    mem(ok,0);
    mem(s,0);
    for(int i:vec1){
        s[l[i]]+=MAXK+i;
        s[r[i]+1]-=MAXK+i;
        l2[i]=MAXN;
        r2[i]=0;
    }
    _rep(i,1,n){
        s[i]+=s[i-1];
        c[i]=c[i-1]+(s[i]==0);
```

```
    }
    for(int i:vec2){
        if(c[r[i]]-c[l[i]-1]==0)
        vec3.push_back(i);
    }
    if(vec3.size()==0)
    return;
    else if(vec3.size()==1){
        int t=*vec3.begin();
        for(int i:vec2){
            if(i!=t)
            ans[i]=false;
        }
    }
    else{
        for(int i:vec2)
        ans[i]=false;
    }
    int L=n,R=1;
    for(int i:vec3)
    L=min(r[i],L),R=max(l[i],R);
    _rep(i,1,n){
        if(s[i]){
            s[i]-=MAXK;
            if(s[i]<MAXK){
                l2[s[i]]=min(l2[s[i]],i);
                r2[s[i]]=max(r2[s[i]],i);
            }
        }
    }
    for(int i:vec1){
        if(l2[i]<=r2[i]){
            if(l2[i]<=L&&R<=r2[i])
            ok[i]=true;
        }
    }
    for(int i:vec1){
        if(!ok[i])
        ans[i]=false;
    }
}
int main(){
    int n=read_int(),k=read_int();
    _rep(i,1,k){
        l[i]=read_int();
        r[i]=read_int();
        w[i]=read_LL();
        ans[i]=true;
    }
    _for(i,0,MAXV){
        vec1.clear();
```

```
        vec2.clear();
        _rep(j,1,k){
            if((w[j]>>i)&1)
            vec1.push_back(j);
            else
            vec2.push_back(j);
        }
        solve(n);
    }
    _rep(i,1,k){
        if(ans[i])
        putchar('1');
        else
        putchar('0');
    }
    return 0;
}
```