#include<bits/stdc++.h>

using namespace std;

struct node {

int u;

int v;

int wt;

node(int first, int second, int weight) {

u = first;

v = second;

wt = weight;

}

};

bool comp(node a, node b) {

return a.wt < b.wt;

}

int findPar(int u, vector<int> &parent) {

if(u == parent[u]) return u;

return parent[u] = findPar(parent[u], parent);

}

void unionn(int u, int v, vector<int> &parent, vector<int> &rank) {

u = findPar(u, parent);

v = findPar(v, parent);

if(rank[u] < rank[v]) {

parent[u] = v;

}

else if(rank[v] < rank[u]) {

parent[v] = u;

}

else {

parent[v] = u;

rank[u]++;

}

}

int main(){

int N,m;

cin >> N >> m;

vector<node> edges;

for(int i = 0;i<m;i++) {

int u, v, wt;

cin >> u >> v >> wt;

edges.push\_back(node(u, v, wt));

}

sort(edges.begin(), edges.end(), comp);

vector<int> parent(N);

for(int i = 0;i<N;i++)

parent[i] = i;

vector<int> rank(N, 0);

int cost = 0;

vector<pair<int,int>> mst;

for(auto it : edges) {

if(findPar(it.v, parent) != findPar(it.u, parent)) {

cost += it.wt;

mst.push\_back({it.u, it.v});

unionn(it.u, it.v, parent, rank);

}

}

cout << cost << endl;

for(auto it : mst) cout << it.first << " - " << it.second << endl;

return 0;

}