HW12

3200105787 张云策

16.5

because is the key of r2 which means it's not repeat, so $r1 \bowtie r2$ have most 1000 size

but the E of r3 also is key,but the size of r3 is 750 which is smaller than 1000 size , so $r1 \bowtie r2 \bowtie r3$ max size is 1000

because C,E is the key of relations r2,r3, so we can build a index for C and E, then for each record in r1, we use index of C r2 for search r2, then use index of E r3 for search r3, so we will finally look up the tuple which we want.

16.6

because we could get the size from average number of tuples which join with other tuple to second relation, . In this case, for each tuple in r1, 1500/V(C,r2) = 15/11 tuples (on the average) of r2 would join with it. The intermediate relation would have 15000/11 tuples. This relation is joined with r3 to yield a result of approximately 10,227 tuples (15000/11 × 750/100 = 10227). A good strategy should join r1 and r2 first, since the intermediate relation is about the same size as r1 or r2. Then r3 is joined to this result.

16.16

because we have the index of (dept_name, building), we can easy find the tuple which have (building="watson" and dept_name="Music"), then we follow the pointer and contine find the tuple which building less than "watson",and if the budget <55000 we rejected it. then we can get the answer.