COMP 421 Assignment 2

1.

a).

Patient

The length of a patient record: $12 + \frac{2}{3} \times 30 + \frac{2}{3} \times 30 + \frac{2}{3} \times 42 + 11 = 91\ Bytes$

The number of pages in Patient table: $91 \times 30000/(0.75 \times 4000) = \mathbf{910\ Pages}$

Doctor

The length of a doctor record: $10 + \frac{2}{3} \times 48 + 10 + \frac{2}{3} \times 33 = 74\ Bytes$

The number of pages in Doctor table: $74 \times 200/(0.75 \times 4000) = 4.93 \approx \mathbf{5\ Pages}$

Diagnosis

The length of a diagnosis record: $8 + \frac{2}{3} \times 300 + 10 + 1 + 0.8 \times 10 + 12 + 10 = 249\ Bytes$

The number of pages in Diagnosis table: $249 \times 10 \times 30000/(0.75 \times 4000) = \mathbf{24900\ Pages}$

b).

i.

Since there is no index, we have to scan the relation. Thus, I/O is **24900 Pages**.

ii.

The number of rids per data entry = 1 (since type 1 index)

The size of a data entry = 10+10 = 20 Bytes

Total leaf page data = 20*10*30000 = 6,000,000 Bytes

Thus, the number of leaf pages = 6000000/(0.75*4000) = 2000 Pages

And the number of data pages = 10*30000*1/200 = 1500 Pages

Hence, I/O = 2000*1/200+1500 = **1510 Pages**

iii.

The number of matching records = (Y-X)*10*30000/(365*2)

The number of records in one data page = (4000*0.75)/249

Hence, the number of data pages is:

$$\frac{The\ number\ of\ matching\ records}{The\ number\ of\ records\ in\ one\ data\ page} = \frac{(Y-X)\times 10\times 30000}{365\times 2} \div \frac{4000\times 0.75}{249} = \frac{2490(Y-X)}{73}\ pages$$

Since the index is clustered, we only need to access 1 leaf page.

Hence, $\boldsymbol{I/O = (1 + \frac{2490(Y-X)}{73})\ pages}$

(In worst case, we might need to access one more data page if the last record whose diagdate is between X and Y is at the end of the last data page we should have accessed. In this case, we have to read one more data page to know it's time to stop reading.)

3.

a).

First, we do:

$$\rho(t_D, \Pi_{followupdate,hcnum,practiceid}(\sigma_{diagdate\geq'2020-12-01'}(D)))$$

$$\rho(t_{Doc}, \Pi_{practiceid,name}(\sigma_{soeciality='Oncology'}(Doc)))$$

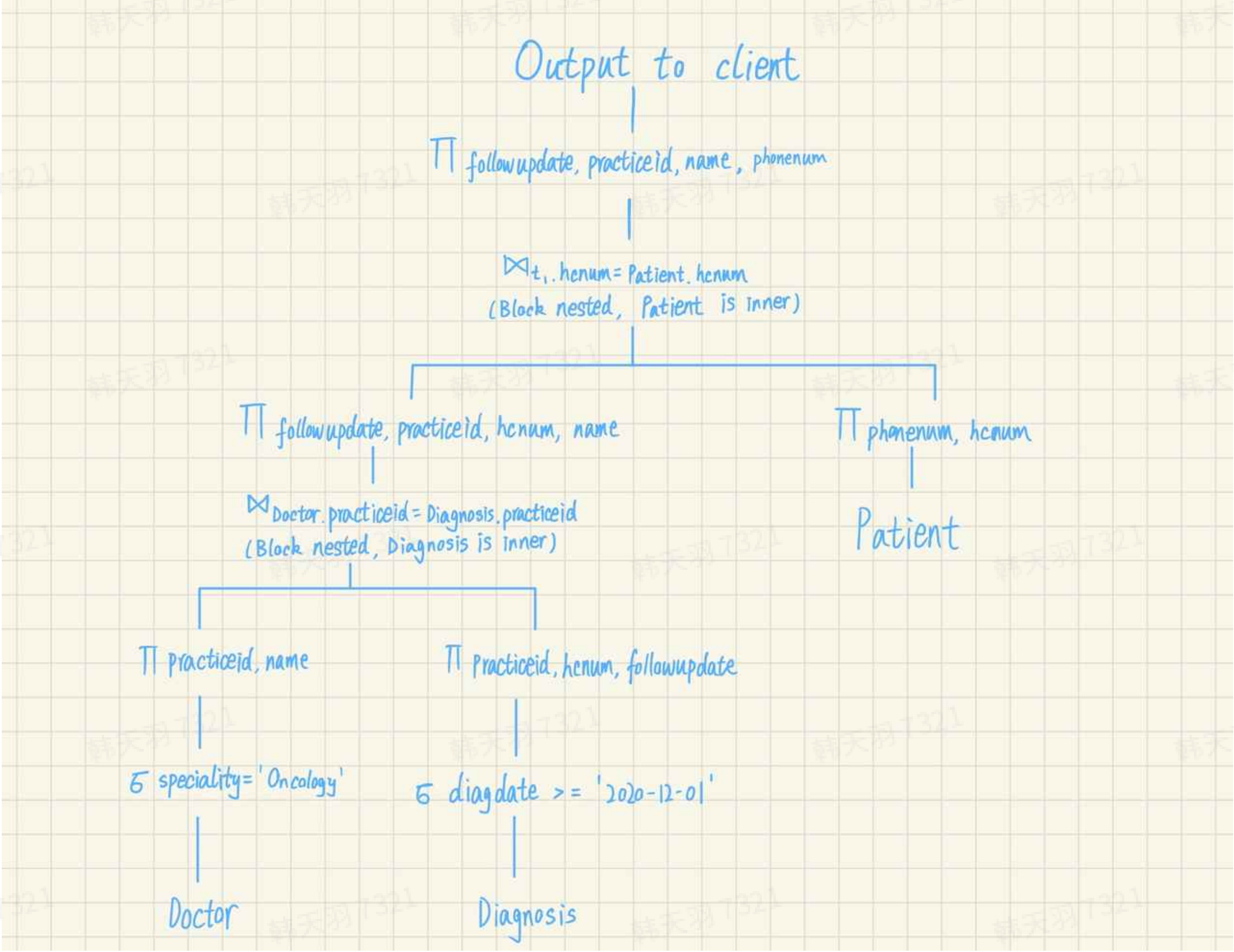
$$\rho(t_P, \Pi_{hcnum,phonenum}(P))$$

Then, we do:

$$\rho(t_1, (t_{Doc} \bowtie_{t_{Doc}.practiceid=t_D.practiceid} t_D))$$

$$\Pi_{(phonenum,followupdate,practiceid,name)}(\Pi_{(followupdate,practiceid,hcnum,name)}(t_1) \bowtie_{t_1.hcnum=t_P.hcnum} t_P)$$

b).


Output to client
Π followupdate, practiceid, name, phonenum
⋈ t1.hcnum = Patient.hcnum
(Block nested, Patient is inner)
Π followupdate, practiceid, hcnum, name
Π phonenum, hcnum
⋈ Doctor.practiceid = Diagnosis.practiceid
(Block nested, Diagnosis is inner)
Patient
Π practiceid, name
Π practiceid, hcnum, followupdate
σ speciality='Oncology'
σ diagdate >= '2020-12-01'
Doctor
Diagnosis

First, we do projection on Doctor table. Since the distribution of speciality in Doctor table is uniform, the number of matching records is:

$$\frac{1}{25} \times 200 = 8 \ records$$

Thus, the number of pages we need to store these data is:

$$(10 + \frac{2}{3} \times 48) \times 8/4000 \approx 1 \ page < 100 \ pages$$

Hence, we can store this in memory. No extra I/O needed. The I/O cost in this step is just reading the **5 pages** in Doctor table.

Thus, we can perform block nested join on Diagnosis and Doctor, with Diagnosis being inner and 1 page in the memory being outer. And the pages read in this step is: **0+1*24900=24900 Pages** since there are 24900 pages in Diagnosis table (the matching data in Diagnosis table can also fit into memory).

Next, we do the projection. Since the number of the records in output of the first join is:

$$8 \times (10 \times 30000/200) \times 30/(365 \times 2) \approx 494 \; records$$

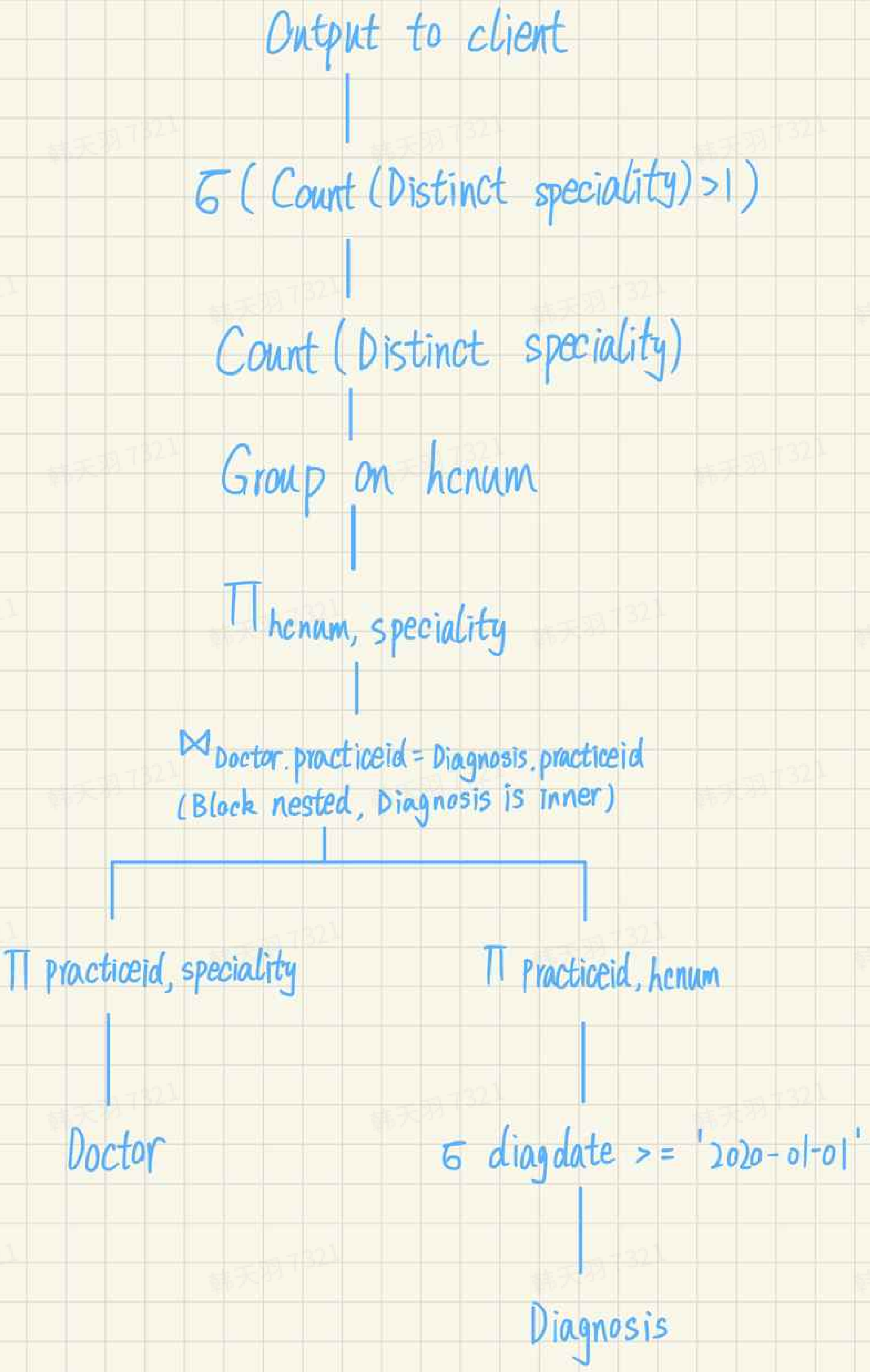
Thus, the number of pages we need to store these data is:

$$(0.8 \times 10 + 12 + 10 + \frac{2}{3} \times 48) \times 494/4000 \approx 8 \; pages < 100 \; pages$$

Hence, we can store this in memory and pipeline it into the next step, which is another block nested join with Patient, with Patient being inner. Hence, the total pages read in this step is: **0+1*910=910 Pages** since there are 910 pages in Patient table.

Hence, the total cost of this execution plan is **5+24900+910 = 25815 Pages**

4.


Output to client
σ (Count (Distinct speciality) >1)
Count (Distinct speciality)
Group on hcnum
Π hcnum, speciality
⋈ Doctor.practiceid = Diagnosis.practiceid
(Block nested, Diagnosis is inner)
Π practiceid, speciality
Π Practiceid, hcnum
Doctor
σ diagdate >= '2020-01-01'
Diagnosis

First, we do projection on Diagnosis table. Since the distribution of speciality in Doctor table is uniform, the number of matching records is:

$$30000 \times 10 \times 365/(365 \times 2) = 150,000\ records$$

Thus, the number of pages we need to store these data is:

$$(12 + 10) \times 150000/4000 = 825\ pages > 100\ pages$$

Thus, we need to write this to disk. Hence, I/O in this step is 24900+825 = **25725 pages**

Second, we do the block nested join. Since Doctor as the outer table only has 5 pages, it can fit into memory. The I/O for this is: $5 + \lceil 5/100 \rceil \times 825 = \mathbf{830\ pages}$

Next, we do the projection on the output of this join. The number of pages we need to store these data is:

$$(12 + \frac{2}{3} \times 33) \times 150000/4000 = 1275\ pages > 100\ pages$$

Thus, we need to write this to disk. Hence, I/O in this step is **1275 pages**

Finally, for sorting, since we have 1275 pages, we need Pass 0(1275/100=13 runs) and Pass 1 (for merging). But we can remove the cost of the last pass of sort by pipelining the output of Pass 1 to the aggregation. Optimized cost would be: $2 \times 1275 + 1275 = \mathbf{3825\ pages}$

Hence, the total cost is **25725+830+1275+3825 = 31655 pages**