# 2021

3.

a)

bfr = floor(B/R) = floor(512/250) = 2

1. 20  
2. 40  
3. 80  
4. 160

4 levels

b) sl(Age) = r/n = 10000/200 = 50  
sl(Salary) = r/n = 10000/500 = 20  
sl(DNO) = r/n = 10000/100 = 100

(sl(age=60) + sl(age=25)) \* sl(dno=5) \* (sl(salary=60000 + salary=30000)

= (50 + 50) \* 100 \* (20 + 20) = 100 \* 100 \* 40 tuples (**0.01 \* 0.01 \* 0.004**)

c)

Plan 1 (B+ Clustering index on Age): = t + 1 + r\*sl(DNO) = 4 + 0.01 \*

4.

a) n = 500

jc = (|R|\*|S|)/max(n, m) = 500 \* 30 / max(500, 10) = **30**  
js = 1/max(500, 10) = 1/500

Matching tuples: 2 attributes => R = 10 \* 2 = 20 bytes

bfr = floor(B/R) = floor(64/20) = **3**

b)

Using Clustering Index over ESSN:

bfr\_E = floor(B/R) = floor(64/(10\*3)) = 2  
b\_E = ceil(500/2) = 250 [mistake in solution]

bfr\_D = floor(64/20) = 3  
b\_D = ceil(30/3) = 10

**k** (last term in sum) **= jc/bfr\_resulting** = 30/3 = 10



b\_E + |E| \* (1 + ceil(sl(D)/f\_D)) + k = 250 + 500 \* (1+ ceil(10/3)) + 10 = 260 + 2500 = NO

Using Index-based Loop join on B+ Tree:



10 + 30 \* (3 + 1 + ???) + k(=10) = 20 + 120 + 30\*??? = **140** + 30\*???

# 2020

3.

a) selectivity = 30 000 / 90 000 \* 2 / 5 = 1/3 \* 2/5 = 0.133

b)

sl(Salary) = 1/3  
cost(Salary) = t + ceil(r\*sl(A) / f) = 3 + ceil(10 000 / 3 / 2) = 1667 + 3 = 1670

sl(DNO) = 2/5  
cost(DNO) = t + 1 + r\*sl(DNO) = 2 + 1 + 10 000 \* 2/5 = 3 + 4000 = 4003

Thus, plan to do Salary first is better.

4.

a) (q-1)\*7 + (q-1)\*7 + 7\*1 <= 64 or 14q <= 71 or q <= 5.07

q = 5

b) Since there are q-1 (5-1 = 4) index values, 280/4 = 70 leaf nodes

c)   
5600/280 = 20 employees having Salary values

sl(Salary) = 1/280

cost(Salary) = t + 1 + r\*sl(Salary) =

# 2019

3.

a)

M1: 0, 3, -> 6, 9 (2 blocks)   
M2: 4 (1 block)  
M3: 2, 8 (1 block)

b)

best: (1+1+1)/3 = 1

worst: (2+1+1)/3 = 1.33

c) SELECT \* FROM EMPLOYEE WHERE DNO >= 3 AND DNO <= 8

Worst-case in Hashed file, sequential file (ordered by DNO), Heap File

Hashed file: 1.33 \* 6 (6 values in the range, even if not in DB) = 8

Sequential file: 4 blocks => log2(4) + 1 (next contiguous block) = 3

Heap File: scan whole file => 4

Sequential < Heap < Hashed

4.

Index cost:  
bfr = floor(B/R) = floor(256/100) = 2  
b = ceil(r/bfr) = ceil(1000/2) = **500** blocks  
index entry = P + R\_DNO = 5+5 = 10  
bfr\_index m = floor(B/index entry) = floor(256/10) = 25  
b\_index = ceil(n/m) = 1 block  
Each cluster has on average 500/4 = 125 blocks  
Cost = 1 + 125 = **126** block accesses

Baseline (serial scan):  
Each DNO’s probability ¼ (4 values)  
If DNO=1, access first 125 blocks  
If DNO=2, 250, etc  
Cost = ¼ \* (125 + 250 + 375 + 500) = **312**.5 block accesses

* Use Clustering Index

5.

a) sl(Salary range) = (50000 – 35000)/45000 = 15/45 = 1/3  
sl(DNO) = 1/125  
conjunctive sl = 1/375

s(query) = 10000 \* 1/375 = 26.67

b)

Plan 1: Cost(Salary) = t + O(b/n) (or just s/bfr?) = 3 + 10000\*1/3 /5 = 3 + 666.67 = 669.67 blocks (not considering memory) => 670 + 670-100 = 1240 block accesses

Plan 2: Cost(DNO) = t + 1 + r\*sl(DNO) = 3 + 10,000\*1/125 = 83 block accesses (16 index blocks) < 100 (memory)

* Choose plan 2

# 2018

7.

a)

 + (js \* r\_E \* r\_D)/bfr\_RS

js = 1/max(1000, 10) = 0.001

D as outer Cost = 50 + 100\*ceil(50/10) + (1/1000 \* 1000 \* 10)/10 = 50 + 500 + 1 = 551

E as outer Cost = 100 + 50\*ceil(100/10) + 1 = 601

* Best is D as outer

b) Cartesian Cost = n\_E \* n\_D + (js \* r\_E \* r\_D)/bfr\_RS = 5000 + 1 = 5001

c)

Plan 1: Secondary (B+ Tree) Index Cost:  
 = b\_E + r\_E \* (x\_D + 1) + 1 = 100 + 1000 \* (2 + 1) + 1 = 3101

Plan 2: Primary Index on SSN:

= b\_D + r\_D \* (x\_E + 1) + 1 = 50 + 10 \* (2 + 1) + 1 = 81

* Best plan is Plan 2

d) b\_R + b\_S + 1 = **151**