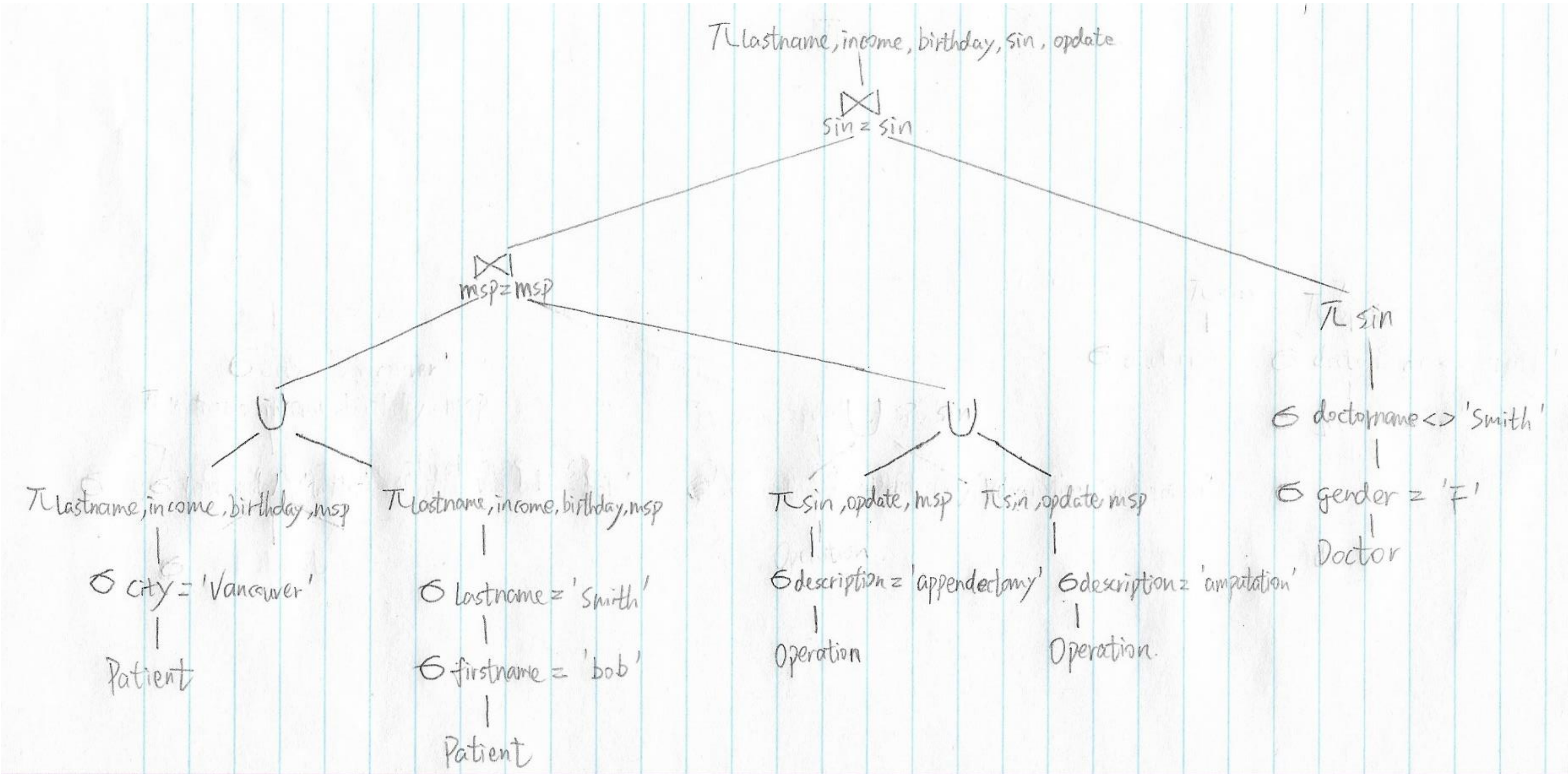
Q1




Q2

a. Since total number of countries is 5, there are approximately 10,000 records for vcountry='Canada'. Leaf nodes store 4096/ (8+16*2) ≈ 102 records (one pointer, vcountry and employee for one record). Since occupancy is 75%, each leaf node stores 102*75%≈76 records. we need to read 10,000/ 76≈132 leaf nodes. **Selection** needs 10,000+132+3-1=10,134 reads. **Projectio**n requires 9000*(16*3)/4096≈106 reads
   **Total cost= 106+10,134=10240**
b. Since there are 5 records in V (V, vcountry), there are approximately 10000 records for vcountry='Canada'. And there are 4500 of 5000 records satisfied employees > 500. So, **9000 records** satisfy requirements. 9000*(16*3) = **432,000 bytes**.
c. 200,000/25,000 = 8 (blocking factor). Selection needs 4-1+50,000/8 +4-1+30,000/8=10,006 reads.
   **Total cost=1016+10,006=11,024**
d. 200,000*25%(Vancouver)+200,000*15%(Edmonton)=**80,000 records**. 80,000*(4+16+32) = **4,160,000 bytes**. 4,160,000/4096≈1,016 blocks.
e. **100,000 records**. 100,000*(4+16+16+64+8) (size of pid, pname, vname, description and ptype respectively) = 10,800,000 bytes.

f. Size of single relations C, P, V, S are 80,000, 100,000, 9,000 and 10,000,000 respectively.

| Single | C | P | V | S |
|---|---|---|---|---|
| Size | 80,000 | 100,000 | 9,000 | 10,000,000 |
| Cost | 0 | 0 | 0 | 0 |

Size of relation pairs **S⋈C, S⋈P, P⋈V** are
10,000,000*80,000/200,000=4,000,000,
10,000,000*100,000/100,000=10,000,000, 100,000*9,000/50,000=18,000
respectively.

| Pairs | **S⋈C** | **S⋈P** | **P⋈V** |
|---|---|---|---|
| Size | 4,000,000 | 10,000,000 | 18,000 |
| Cost | 0 | 0 | 0 |

Size of relation Triples **S⋈C⋈P**, **S⋈P ⋈C, S⋈P⋈V and P⋈V⋈S** are
4,000,000*9,000/100,000=360,000, 360,000,
10,000,000*9,000/50,000=1,800,000 and 1,800,000 respectively.

| Triple | **S⋈C⋈P** | **S⋈P ⋈C** | **S⋈P⋈V** | **P⋈V⋈S** |
|---|---|---|---|---|
| Size | 360,000 | 360,000 | 1,800,000 | 1,800,000 |
| Cost | 4,000,000 | 10,000,000 | 10,000,000 | 18,000 |

| Final | **S⋈C⋈P⋈V** | **S⋈P ⋈C⋈V** | **S⋈P⋈V⋈C** | **P⋈V⋈S⋈C** |
|---|---|---|---|---|
| Cost | 4,360,000 | 10,360,000 | 11,800,000 | 1,818,000 |

Size of one record for P⋈V, P⋈V⋈S and P⋈V⋈S⋈C are 140 bytes,152 bytes and 192 bytes. Blocking factor for P⋈V, P⋈V⋈S and P⋈V⋈S⋈C are 29, 26 and 21.

**P⋈V⋈S⋈C** is the best plan. Cost is 1,818,000.

| | **P⋈V** | **P⋈V⋈S** | **P⋈V⋈S⋈C** |
|---|---|---|---|
| Size of records | 18,000 | 1,800,000 | 720,000 |
| Size of blocks | 621 | 69,231 | 34,286 |

g. **P⋈V**
Hash join: Size of one record in V is 48 bytes. Blocking factor is floor (4096/48) =85. All 9,000 records need 106 blocks. Size of one record in P is 108 bytes. Blocking factor is floor (4096/108) =37. All 9,000 records need 2,703 blocks.
Cost=3(B(V)+B(P)) = 8,427
Block nested loop: cost= B(V)+B(P)=2,809
Index nested loop: impossible, no relation has an index on the join attribute

h. **P⋈V⋈S**

Hash join: Size of one record in P⋈V is 140 bytes. Blocking factor is floor (4096/140) =29. All 18,000 records need 621 blocks. B(S)= 500,000.
Cost=3(B(P⋈V) +B(S)) = 1,501,863
Block nested loop: M-2=598<B(P⋈V) =621, cost=B(P⋈V) +ceiling(B(P⋈V)/(M-2)) *B(S)= 60,376
Index nested loop: Blocking factor 31

i. **P⋈V⋈S⋈C**
Hash join: Size of one record in P⋈V⋈S is 152 bytes. Blocking factor is floor (4096/152) =26. All 1,800,000 records need 69,231 blocks. Size of one record in C is 52 bytes. Blocking factor is floor (4096/52) =78. All 80,000 records need 1,026 blocks. Cost=3(B(P⋈V⋈S) + 700,297 B(C)) = 210,771
Block nested loop: M-2=598<B(C) =1,026, cost=B(C) +ceiling(B(C)/(M-2)) *B(P⋈V⋈S) = 139,488
Index nested loop: impossible, no relation has an index on C

| | **P⋈V** | **P⋈V⋈S** | **P⋈V⋈S⋈C** |
|---|---|---|---|
| Block nested loop | 2,809 | 1,000,621 | 139,488 |
| Hash join | 8,427 | 1,501,863 | 210,771 |
| Index nested loop | - | 558,621 | - |

j. The first and the second join operation are pipelining. The second join operations are accumulating tuples from the first join operation into its buffer. The first join operation can release its buffers allocated when the second join starts. However, the second join operation will not output any tuple to the third join until all tuples from first join has been received since there is not enough buffers for the third join operation.
**Total cost=2,809+558,621+139,488-621(buffers for P⋈V)= 700,297**