HW11

12.2

Consider the bank database of Figure 12.13, where the primary keys are underlined, and the following SQL query:

```
select T.branch_name
from branch T, branch S
where T.assets > S.assets and S.branch_city = "Brooklyn"
```

Write an efficient relational-algebra expression that is equivalent to this query. Justify your choice.
Answer:

Query:

$$\Pi_{T.branch\_name}((\Pi_{branch\_name,\, assets}(\rho_T(branch))) \bowtie_{T.assets > S.assets} (\Pi_{assets}(\sigma_{(branch\_city = 'Brooklyn')}(\rho_S(branch)))))$$

This expression performs the theta join on the smallest amount of data possible. It does this by restricting the right hand side operand of the join to only those branches in Brooklyn, and also eliminating the unneeded attributes from both the operands.

12.3[b]

If r_1 is the outer relation, we need $\lceil 800/(M-2) \times 1500 + 800 \rceil$ disk accesses, and $2\lceil 800/(M-2) \rceil$ disk search.

If r_2 is the outer relation, we need $\lceil 1500/(M-2) \times 800 + 1500 \rceil$ disk accesses, and $2\lceil 1500/(M-2) \rceil$ disk search.

12.10

a. The number of blocks in the main memory buffer available for sorting(M) is $(40 \times 10^6)/(4 \times 10^3) = 10^4$. The number of blocks containing records of the given relation (b_r) is $(40 \times 10^9)/(4 \times 10^3) = 10^7$. Then the cost of sorting the relation is: $(NumberofDiskSeeks \times DiskSeekCost) + (NumberofBlockTransfers \times BlockTransferTime)$. Here Disk seek cost is 5×10^{-3}seconds and block transfer time is 10^{-4} seconds $(4 \times 10^3)/(40 \times 10^6)$. The number of block transfers is independent of b_b and is equal to 25×10^6.

- **Case 1**: $b_b = 1$
  Using the equation in Section 12.4, the number of disk seeks is 5002×10^3. Therefore the cost of sorting the relation is: $(5002 \times 10^3) \times (5 \times 10^{-3}) + (25 \times 10^6) \times (10^{-4}) = 25 \times 10^3 + 2500 = 27500$ seconds.

- **Case 2**: $b_b = 100$
  The number of disk seeks is: 52×10^3. Therefore the cost of sorting the relation is: $(52 \times 10^3) \times (5 \times 10^{-3}) + (25 \times 10^6) \times (10^{-4})$ $= 260 + 2500 = 2760$ seconds.

b. $\lceil \log_{M-1}(b_r/M) \rceil$. This is independent of b_b. Substituting the values above, we get $\lceil \log_{10^4-1}(10^7/10^4) \rceil$ witch evaluates to 1.

c. Flash storage:

- **Case 1**: $b_b = 1$
  The number of disk seeks is: 5002×10^3. Therefore the cost of sorting the relation is: $(5002 \times 10^3) \times (1 \times 10^{-6}) + (25 \times 10^6) \times (10^{-4}) = 5.002 + 2500\ 2506$ seconds.

- **Case 2**: $b_b = 100$
  The number of disk seeks is: 52×10^3. Therefore the cost of sorting the relation is: $(52 \times 10^3) \times (1 \times 10^{-6}) + (25 \times 10^6) \times (10^{-4}) = 0.052 + 2500$, which is approx $= 2500$ seconds.

12.16

a. Using pipelining, output from the sorting operation on r is written to a buffer B. When B is full, the merge-join processes tuples from B, joining them with tuples from s until B is empty. At this point, the sorting operation is resumed and B is refilled. This process continues until the merge-join is complete.

b. If the sort–merge operations are run in parallel and memory is shared equally between the two, each operation will have only M/2 frames for its memory buffer. This may increase the number of runs required to merge the data.