CPS216 Data-Intensive Computing Systems - Fall 2010
Written Assignment 2 (Solutions)

Question 1
1. Completeness. To ensure there are still result generated by highly selective operation (e.g. filter, join).
2. Any graph showing the data flow is fine.

Question 2
1. 

\[
\{RSTU\} \\
\{RSU\} \{RSU\}(b) \{RST\} \{RST\}(c) \{RST\}(d) \{RUT\} \{RUT\}(a) \{RUT\}(d) \\
\{RS\} \{RS\}(b) \{RS\}(c) \{RS\}(d) \{RT\} \{RT\}(a) \{RT\}(c) \{RU\} \{RU\}(a) \{RU\}(b) \{RU\}(d) \{SU\} \{SU\} \\
\{SU\}(c) \\
R, R(a), R(b), R(c), S, S(a), S(d), T, T(b), U, U(c), U(d)
\]

2. Since we are going to select the distinct value of R.a. The order of a become interesting in this case.

\[
\{RSTU\} \{RSTU\}(a) \\
\{RSU\} \{RSU\}(a) \{RSU\}(b) \{RST\} \{RST\}(a) \{RST\}(c) \{RST\}(d) \{RUT\} \{RUT\}(a) \{RUT\}(d) \\
\{RS\} \{RS\}(a) \{RS\}(b) \{RS\}(c) \{RS\}(d) \{RT\} \{RT\}(a) \{RT\}(c) \{RU\} \{RU\}(a) \{RU\}(b) \{RU\}(d) \{SU\} \{SU\}(a) \{SU\}(c) \\
R, R(a), R(b), R(c), S, S(a), S(d), T, T(b), U, U(c), U(d)
\]

3. Since we are going to select the distinct value of T.e. The order of e become interesting in this case.

\[
\{RSTU\} \{RSTU\}(e) \\
\{RSU\} \{RSU\}(b) \{RST\} \{RST\}(c) \{RST\}(d) \{RST\}(e) \{RUT\} \{RUT\}(a) \{RUT\}(d) \{RUT\}(e) \\
\{RS\} \{RS\}(b) \{RS\}(c) \{RS\}(d) \{RT\} \{RT\}(a) \{RT\}(c) \{RT\}(e) \{RU\} \{RU\}(a) \{RU\}(b) \{RU\}(d) \{SU\} \{SU\}(a) \{SU\}(c) \\
R, R(a), R(b), R(c), S, S(a), S(d), T, T(b), T(e), U, U(c), U(d)
\]
Question 3
1. the idea is to buffer as many blocks as possible in the bucketizing process while not overflow the memory.

\[ x\left\lceil \frac{B(S)}{k} \right\rceil + k - x \leq 100 \]

\( x \) is the number of buckets we want to buffer in memory. \( k \) is the total number of buckets. Let's assume \( x=1 \), this will not change our goal to buffer as many blocks in memory as possible.

In this case, the minimum value of \( k \) is 12, which means there will be 12 buckets. The number of blocks per bucket for \( S \) is 84. The number of blocks per bucket for \( R \) is 105.

The cost will be:
- bucketizing \( S \):
  \[ B(S) + 84 \times (k-1) = 1924 \]
- bucketizing \( R \):
  \[ B(R) + 105 \times (k-1) = 2405 \]
- join:
  \[ 84 \times (k-1) + 105 \times (k-1) = 2079 \]

So, the total cost is 6408

Note that the average number of blocks per bucket you get may be 83 for \( S \) and 104 for \( R \). In that case, the total cost should be 6364. Any cost between 6408 and 6364 is fine.

2. the equation now becomes:

\[ x\left\lceil \frac{B(S)}{k} \right\rceil + k - x \leq 50 \]

The equation doesn't hold for any positive value of \( m \). So, a conventional hash join should be used here. The cost is \( 3 \times (B(R) + B(S)) = 6750 \)

Question 4
1. \( A=5, B=5 \); As the Undo logging stores the initial values, \( A \) and \( B \) will have the values of the first entry in the log.
2. \( A=15, B=5 \); During the recovery, only the uncommitted \( T2 \) will be undone.
3. \( A=10, B=10 \); Since Redo logging stores the new values and no log contained the final disk value of 10, it means that the changes of the log were not reflected on the hard disk, i.e. 10 was the initial value.
4. \( A=5, B=10 \); During the recovery, only the committed \( T1 \) will be redone.

Question 5
1. \( W=\{4, 7\}; X=\{17, 28\}; Y=5; Z=10 \)
2. \( T2 \), since it is committed after the checkpoint starts.
3. \( W=\{4, 7\}; X=\{14, 17, 28\}; Y=\{5, 15\}; Z=\{10, 20\} \); \( T1 \) and \( T2 \) will be redone.

Question 6
1. \( A=45, B=5, C=35, D=15 \)
2. \( A=10, B=5, C=35, D=5 \)
3. A=10, B=5, C=35, D=5
4. A=10, B=15, C=10, D=20
5. A=10, B=15, C=10, D=20
6. A=15, B=15, C=15, D=20

Question 7
You can get detailed comparison in Tom White's book. You should list some of the main points as below (but not limit to):
HBase: column-based, easy to scale up, data is sorted, focus on storage while expose business logic to outside; use redo logging.

RDBMS: mostly row-based, not easy to scale up, support various indexes, strict constrains (ACID, schema, etc.), has an interface SQL and execution engine. Various logging mechanisms.