Quiz 4

Deadline	Friday, 09 April 2021 at 9:00PM
Latest Submission	Friday, 09 April 2021 at 2:57PM
Raw Mark	4.00/4.00 (100.00%)
Late Penalty	N/A
Final Mark	4.00/4.00 (100.00%)

Question 1 (1 mark)

Consider the three relations *R(id,a,b)*, *S(rid,tid,c)*, *T(id,d,e,f)* where the *id* attributes are primary keys, and *S.rid* is a foreign key referencing *R.id* and *S.tid* is a foreign key referencing *T.id*. The sets of primary keys are disjoint.

Consider also the following join on these three relations:

```
select a,b,c,d,e,f from R, S, T where R.id = S.rid and T.id = S.tid
```

Which of the following does ***not*** represent a useful join order for the above SQL statement?

(a) ○	(R ⋈ S) ⋈ T
(b) ○	R ⋈ (S ⋈ T)
(c) ○	(S ⋈ T) ⋈ R
(d) ●	S ⋈ (T ⋈ R)
(e) ○	They are all valid join orders for the query.

✔ Your response was correct.
Mark: 1.00

The keys of R and T do not overlap, so any join on them would be empty.

Question 2 (1 mark)

Under what circumstances will a block nested-loop join achieve optimal join performance, i.e.

Cost = b_{outer} + b_{inner}

Assume that we have *N* buffers, and we want to achieve optimal join performance with as few buffers as possible.

(a) ○	Nested-loop join can never achieve optimal join performance
(b) ●	If there are at most *N-2* pages in the outer table
(c) ○	If there are at most *N-2* pages in the inner table
(d) ○	If N ≥ (b_{outer} + b_{inner})
(e) ○	If N-2 ≥ (b_{outer} + b_{inner})

✔ Your response was correct.
Mark: 1.00

If the entire outer table fits in the buffer pool, and if there is one buffer for input from the inner table and one output buffer, then nested-loop join will have optimal cost.

Question 3 (1 mark)

Consider the following two relations/tables:

R(a,b,c) where r_R = 100000, c_R = 100, b_R = 1000

S(c,d,e) where r_S = 50000, c_S = 500, b_S = 100

If we do a natural join on these two tables, using a block nested loop join with 35 buffers, how many pages do we *read* in completing the join?

(a) ○	1100
(b) ○	100000
(c) ○	None of the other options is correct
(d) ○	3100
(e) ●	4100

✔ Your response was correct.
Mark: 1.00

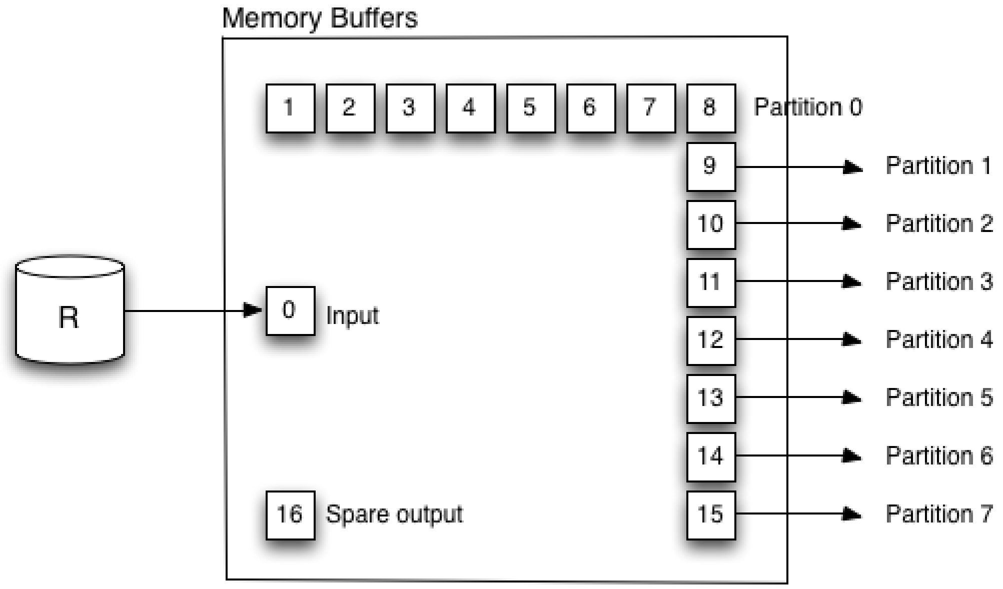
Question 4 (1 mark)

Consider two tables *R(id,x,y,z)* and *S(id,a,b,rid)* where *rid* is a foreign key referring to *R.id*. Consider also the following join on these tables:

```
select x,a from R, S where R.id = S.rid;
```

This join is implemented using the hybrid hash join algorithm with 17 memory buffers, 1 buffer used for input (#0), 8 buffers used to hold one partition in memory (#1-#8), 7 output buffers used to transfer tuples in the other partitions to disk (#9-#15), and 1 "spare" output buffer (#16) used only in the first phase (see below).

The diagram below shows the first phase of the hash join, where *R* is scanned and partitioned into 8 hash buckets. Partition 0 is stored in memory; partitions 1-7 are written to disk.



After the first phase has partitioned *R*, the join algorithm then partitions *S* using the same hash function. Any tuples that hash to partition 0 are matched against *R*'s partition 0 tuples held in memory buffers 1-8, and any resulting matches are written to disk via the spare output buffer. Other *S* tuples are written to disk-based partitions. Partitions 1-7 for *R* and *S* are then processed as for a standard hash join, using a second hash function.

Assume that:

- we have a well-behaved (uniform) hash functions for both the first and second phases
- *R* contains 3000 tuples in 60 pages; each partition of *R* requires exactly 8 pages
- *S* contains 1600 tuples in 40 pages; each partition of *S* requires exactly 5 pages
- each tuple in *S* refers to a different tuple in *R* (i.e. *S.rid* is unique)
- the result contains 1600 tuples which are written into 30 pages (i.e. they count as disk I/O)

Based on the above, compute the number of disk I/Os needed to execute this join.

(a) ○	308
(b) ○	272
(c) ●	312
(d) ○	338
(e) ○	None of the other options is correct

✔ Your response was correct.
Mark: 1.00