Database Management Systems
Practice Exam Questions Solutions

Question 1 FFFFFFTT

Question 2 The relation model is appropriate for structured data where consistency is import (for example banking application, managing information about customers, sales, products and many forms of enterprise data). The unstructured model is used to represent documents where data creators have full autonomy and there are no restrictions on how data is modeled. Note that structured data can be retrieved using structure queries like SQL whereas unstructured data is retrieved using search.

Question 3

1. See P. 728
2. See P. 727
3. See P. 728

Question 4

1. PDBMS can use indexing to speed up queries and maintains statistics on data that can inform optimization decisions. MR systems do not support indexing and have little, if any optimization.
2. MR support fault tolerance for queries by saving partial results persistently after every map-reduce step and can replicate computations. (Note PDBMS only support fault-tolerance for updates.)

Question 5

1.
```
select Link, count(URL)
from Page
where Link ≠ URL
group by Link
```

2. Modify reduce program:

```
reduce(String key, Iterator values):
    int result = 0:
    for each v in values:
            if key ≠ v then result++:
    if result > θ_R then
            Emit(key, AsString(result));
```

```
select Link, count(URL)
from Page
where Link ≠ URL
group by Link
having count(URL)> θ_R
```

Question 6

1. view and conflict serializable, recoverable, avoids cascading aborts, 2PL
2. view and conflict serializable, 2PL
3. view and conflict serializable, recoverable, 2PL
4. all except serial
5. view serializable, recoverable, avoids cascading aborts,

Question 7

1. Solution:
   - V
   - U, S, T (note that if doing physical logging, some actions of V may be redone before being undone)

- Yes, transaction U (specifically update 04) no longer needs to be redone.
- Redo 04: Bring Page C into memory. If pageLSN of C $<$ 04, then change value 1 to value 2 on Page C and change pageLSN to 04.
  Redo 08: Bring Page D into memory. If pageLSN of D $<$ 08, then change value 1 to value 2 on Page D and change pageLSN to 08.
  (for physical logging may redo 09 and 11 as well).
  Redo 13: Bring Page A into memory. If pageLSN of A $<$ 13, then change value 2 to value 3 on Page A and change pageLSN to 13.
  Note that no logging is done during redo as the update actions are already logged.
- Need to undo the update in 09. Bring page C into memory and change value 3 back to 2. Note that this is a new update that needs to be logged. So we add the following records to the log.
  [18,V,CLR,17,C,3,2,$\emptyset$]
  [19,V,END,18]

2. Solution:
   - Undo: V, S
   - Redo U T
   - Redo 04: Bring Page C into memory. If pageLSN of C $<$ 04, then change value 1 to value 2 on Page C and change pageLSN to 04.
     Redo 08: Bring Page D into memory. If pageLSN of D $<$ 08, then change value 1 to value 2 on Page D and change pageLSN to 08.
     Note that no logging is done during redo as the update actions are already logged.
   - Undo 13: Bring A into memory and change value 3 back to 2. The pageLSN of A is set to 13.
     Undo 11. Bring E into memory and change value 2 back to 1. The pageLSN of E is set to 11.
     Undo 09. Bring C into memory and change value 3 back to 2. The pageLSN of C is set to 09.
     Undo 03. Bring A into memory (if it is not already there) and change value 2 back to 1. The pageLSN of A is set to 03.

     Logging:
     [14,S,CLR,13,A,3,2,03]
     [15,V,CLR,11,E,2,1,09]
     [16,V,CLR,15,C,3,2,$\emptyset$]
     [17,V,END,16]
     [18,S,CLR,03,A,2,1,$\emptyset$]
     [19,S,END,18]

Question 8 1. Yes. If the effects of an update by a committed transaction T (which is older than all "loser" or aborted transactions) is still in memory at the time of a crash.

2. No, writing a set of pages is not atomic. Yes, undo only needed if effects of an uncommitted transaction can hit disk.

Question 9 S on Supplier and Supply. SIX Part. If a part tuple p_i satisfies predicate then get X on p_i to do update.

Question 10 YNNY

Question 11 Solution

1. T1 acquires shared-lock on X; When T2 asks for an exclusive lock on X, since T2 has a lower priority, it will be aborted; T3 now gets exclusive-lock on Y; When T1 also asks for an Ex-Lock on Y which is still held by T3, since T1 has higher priority, T1 will be blocked waiting; T3 now finishes write, commits and releases all the lock; T1 wakes up, acquires the lock, proceeds and finishes; T2 now can be restarted successfully.
2. In deadlock detection, transactions are allowed to wait, they are not aborted until a deadlock has been detected. (In comparison, in a prevention scheme some transactions may have been aborted prematurely, an unnecessarily.)

   T1 gets S-Lock on X; T2 blocks waiting for Ex-Lock on X; T3 gets Ex-Lock on Y; T1 blocks waiting for Ex-Lock on Y; T3 finishes, commits and releases locks; T1 wakes up, gets Ex-Lock on Y, finishes up and releases lock on X and Y; T2 now gets both Ex-Lock on X and Y, proceeds to finish. No deadlock.