# FIRST SEMESTER EXAMINATIONS 2019/20

# Database Development

**TIME ALLOWED : TWO Hours**

---

**INSTRUCTIONS TO CANDIDATES**

NAME OF CANDIDATE ........................................... SEAT NO ................
USUAL SIGNATURE ...........................................

READ THE FOLLOWING CAREFULLY:

1. This exam paper consists of 40 questions. Each question comprises 5 statements, for which you should select the one most appropriate answer. The questions have the same weighting.
2. The exam mark is based on the overall number of correctly answered questions. The more questions you answer correctly the higher your mark, incorrectly answered questions do not count against you.
3. Enter your name and examination number IN PENCIL on the computer answer sheet according to the instructions on that sheet.
4. When you have completed this exam paper, read the instructions on the computer answer sheet carefully and transfer your answers from the exam paper. Use a HB pencil to mark the computer answer sheet and if you change your mind be sure to erase the mark you have made. You may then mark the alternative answer.
5. At the end of the examination, be absolutely sure to hand in BOTH this exam paper AND the computer answer sheet.
6. Calculators are NOT permitted.

**THIS PAPER MUST NOT BE REMOVED FROM THE EXAMINATION ROOM**

**1.** Consider the scenario of a charter company that uses a database with a relation

Flight(flightno, seat, occupied)

to keep track of which seats are occupied. Assume that two transactions $T_1$ and $T_2$ execute on the database in the following way:

| Time | Event |
|---|---|
| 1 | Transaction $T_1$ checks which seats are free on flight 123 |
| 2 | Transaction $T_1$ sets seat 13$A$ to occupied and commits |
| 3 | $\lightning$ |
| 4 | Transaction $T_2$ checks which seats are free on flight 123 |
| 5 | Transaction $T_2$ sets seat 13$A$ to unoccupied |
| 6 | Transaction $T_2$ sets seat 27$F$ to occupied and commits |

The "$\lightning$" at time 3 indicates a power failure. Assuming the DBMS does not prevent these transactions from executing and there is no changes to the database once it is restarted, which of the ACID properties, besides perhaps Consistency, would be violated?

☐ **A.** Only Atomicity

☐ **B.** Atomicity and Isolation

☐ **C.** Only Isolation

☐ **D.** Isolation and Durability

■ **E.** No ACID property is violated

**2.** If we use Redo logging, which properties does the database need to satisfy to ensure both Atomicity and Durability:

☐ **A.** Force/Steal

☐ **B.** No Force/Steal

☐ **C.** Force/No Steal

■ **D.** No Force/No Steal

☐ **E.** None of the above; If we use Redo logging we always satisfies both Atomicity and Durability

**3.** Which of the following times is it required and sufficient to change the log when doing Undo logging?

**(I)** At the start of a transaction
**(II)** On committing a transaction
**(III)** On aborting a transaction
**(IV)** When we do a non-database operation in a transaction
**(V)** When a transaction reads an item
**(VI)** When a transaction writes an item

☐ **A.** In all the above cases
☐ **B.** Precisely in case of I, II and IV
■ **C.** Precisely in case of I, II, III and VI
☐ **D.** Precisely in case of I, II, III, V and VI
☐ **E.** In none of the above cases

**4.** Consider the following query plan and database relations:

$$\begin{tikzcd} & \pi_{B,G} \arrow[d] & \\ & \bowtie \arrow[dl] \arrow[dr] & \\ \sigma_{D=b} \arrow[d] & & \rho_{F\to A} \arrow[d] \\ R & & S \end{tikzcd}$$

Assume a database with the following relations:

R

| A | B | C | D |
|---|---|---|---|
| a | d | c | a |
| c | c | d | b |
| a | b | d | b |

S

| E | F | G |
|---|---|---|
| a | a | a |
| a | b | c |

Which of the following relations is the result of this query plan?

(I)

| B | G |
|---|---|
| b | b |

(II)

| B | G |
|---|---|
| b | a |

(III)

| B | G |
|---|---|
| d | b |
| b | a |

(IV)

| B | G |
|---|---|
| d | b |
| b | b |

(V)

| B | G |
|---|---|
| c | a |
| b | a |

☐ **A.** (I)
■ **B.** (II)
☐ **C.** (III)
☐ **D.** (IV)
☐ **E.** (V)

**5.** Consider a database that is read-only (i.e., no transaction writes any data to the database). In order to ensure serialisability, which of the following applies?

☐ **A.** Both shared and exclusive locks are necessary and all transactions must be two phase locked.

☐ **B.** Shared locks are necessary and they need to be held until the end of a transaction.

☐ **C.** Shared locks are necessary but they can be released as soon as the read is complete.

■ **D.** No locks are necessary.

☐ **E.** None of the above.

**6.** What are data cubes relevant for?

■ **A.** Data warehousing

☐ **B.** Object-oriented databases

☐ **C.** Recovery

☐ **D.** Semi-structured databases

☐ **E.** None of the above.

**7.** Consider the following schedule:

| **Time** | **Transaction $T_1$** | **Transaction $T_2$** |
|---|---|---|
| 1 | | read_item($X$) |
| 2 | read_item($X$) | |
| 3 | Y := X - 1 | |
| 4 | X := 3 * X | |
| 5 | write_item($Y$) | |
| 6 | write_item($X$) | |
| 7 | | read_item($Y$) |
| 8 | | X := Y + X + 1 |
| 9 | | write_item($X$) |
| 10 | | commit |
| 11 | commit | |

What happens when we execute this schedule with initial values $X$ = 1 and $Y$ = 1? (Assume that read_item and write_item read values from and write values to a shared buffer, respectively.)

☐ **A.** The schedule does not finish, because deadlock occurs.

■ **B.** The schedule finishes execution with $X$ = 2 and $Y$ = 0 on disk.

☐ **C.** The schedule finishes execution with $X$ = 3 and $Y$ = 1 on disk.

☐ **D.** The schedule finishes execution with $X$ = 4 and $Y$ = 0 on disk.

☐ **E.** None of the above.

**8.** What is *not* an advantage of a distributed database?

☐ **A.** Performance
☐ **B.** Resilience
☐ **C.** Scalability
■ **D.** Simplicity
☐ **E.** All the above are advantages of distributed databases.

**9.** A table of a distributed database can be fragmented, but in which way?

☐ **A.** It can be horizontally fragmented but not in other ways
☐ **B.** It can be vertically fragmented but not in other ways
☐ **C.** It can be vertically or horizontally fragmented, but not both at once
■ **D.** It can be vertically or horizontally fragmented or both at once
☐ **E.** It can not be fragmented at all, fragmentation is not a concept in distributed databases

**10.** Consider the following schedule:

$$r_3(Z);\ r_2(Z);\ w_2(Z);\ r_3(Y);\ w_3(Y);\ r_1(Y);\ w_2(X);\ c_2;\ r_1(X);\ w_1(Y);\ c_3;\ c_1$$

Determine which of the following is true:

**(I)** The schedule is conflict-serialisable.
**(II)** The schedule is recoverable.
**(III)** The schedule is cascadeless.
**(IV)** The schedule is strict.

■ **A.** I and II only
☐ **B.** II and III only
☐ **C.** II, III and IV only
☐ **D.** I only
☐ **E.** All of the above

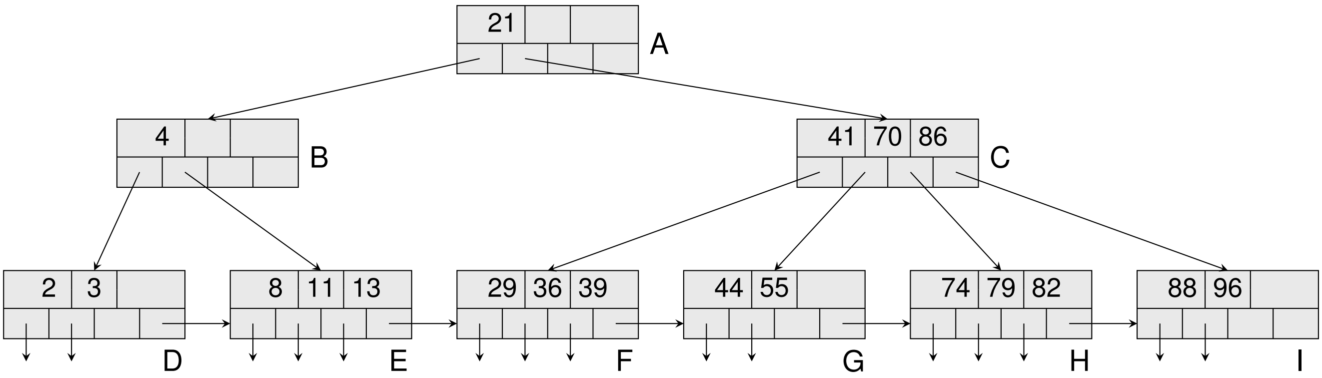


Figure 1: B+ tree for Question 11.

**11.** Consider the B+ tree in Figure 1. What happens if we insert 17 into this B+ tree?

☐ **A.** A new node, K, is inserted between E and F and added as a child to B. We do not change E, but insert 17 into K.

☐ **B.** A new node, K, is inserted between E and F. The value 29 and its pointer are moved from F to K, and 17 is inserted into K. Finally, K is added as a child to C, which results in splitting C into two nodes, where the new node is added as a child to A.

☐ **C.** We attach two children to E and distribute all the values in E evenly across the new children.

■ **D.** A new node, K, is inserted between E and F and added as a child to B. The value 13 and its pointer are moved from E to K, and 17 is inserted into K.

☐ **E.** None of the above.

**12.** Assume a database with schema $R(A, B)$, $S(B, C)$. If $B$ is the primary key for $B$ in $R$, what is the worst-case time required to compute $R \bowtie S$?

☐ **A.** $O(|R| \log_2 |R| \times |S| \log_2 |S|)$

☐ **B.** $O(|R| + |S|)$

■ **C.** $O(|R| \log_2 |R| + |S| \log_2 |S|)$

☐ **D.** $O(|R| \times |S|)$

☐ **E.** $O(|R| \times |S| \times \log_2(|R||S|))$

**13.** Which is *not* a type of transparency in distributed databases?

☐ **A.** Fragmentation transparency

■ **B.** Type transparency

☐ **C.** Replication transparency

☐ **D.** Naming transparency

☐ **E.** All the above are a type of transparency in distributed databases.

**14.** Consider the following schedule:

$$r_3(X);\ r_2(Y);\ w_1(X);\ w_2(Y);\ w_3(Z);\ r_3(Y)$$

This schedule is conflict-equivalent to which of the following serial schedules?

☐ **A.** $w_1(X);\ r_2(Y);\ w_2(Y);\ r_3(X);\ w_3(Z);\ r_3(Y)$

■ **B.** $r_2(Y);\ w_2(Y);\ r_3(X);\ w_3(Z);\ r_3(Y);\ w_1(X)$

☐ **C.** $r_3(X);\ w_3(Z);\ r_3(Y);\ r_2(Y);\ w_2(Y);\ w_1(X)$

☐ **D.** $r_2(Y);\ w_2(Y);\ w_1(X);\ r_3(X);\ w_3(Z);\ r_3(Y)$

☐ **E.** $r_3(X);\ w_3(Z);\ r_3(Y);\ w_1(X);\ r_2(Y);\ w_2(Y)$

**15.** Suppose we execute the following timestamp-based scheduler $S : r_1(X);\ w_1(X);\ r_2(X)$. Assume the scheduler assigns to $T_1$ the timestamp 11, and to $T_2$ the timestamp 222. Also, assume that the read and write times of all items are 0 initially.

What is the read time RT($X$) and write time WT($X$) of $X$ after executing the schedule?

☐ **A.** RT($X$) = 222 and WT($X$) = 222

☐ **B.** RT($X$) = 0 and WT($X$) = 11

☐ **C.** RT($X$) = 222 and WT($X$) = 0

■ **D.** RT($X$) = 222 and WT($X$) = 11

☐ **E.** RT($X$) = 11 and WT($X$) = 11

**16.** A query optimiser selects its best strategy for executing an input query by:

☐ **A.** Executing the initial query plan

☐ **B.** Selecting the highest cost estimate

☐ **C.** Executing the tree from the top-down

☐ **D.** Executing the first equivalent query plan it produces

■ **E.** None of the above

**17.** Let $T$ be a B+ tree of height 3. Suppose $T$ has four leaf nodes $N_1, N_2, N_3, N_4$ such that all values between 72 and 85 are contained in one of these nodes, and each of these nodes contains at least one value between 72 and 85.

How many nodes of $T$ do we have to visit *in the worst case* in order to look up all values between 72 and 85? Assume that the algorithm does not know the number of leaves that contain values between 72 and 85.

☐ **A.** 5

☐ **B.** 6

■ **C.** 7

☐ **D.** 8

☐ **E.** 9

**18.** Which of the following might happen when we execute a conflict-serialisable schedule?

**(I)** Dirty reads

**(II)** Lost updates

**(III)** Non-repeatable reads (i.e., some transaction $T_1$ reads an item $X$, another transaction $T_2$ overwrites $X$ afterwards, and finally $T_1$ reads $X$ again and thus sees a different value than the first time)

■ **A.** I only

☐ **B.** II only

☐ **C.** III only

☐ **D.** I and II only

☐ **E.** None of the above

**19.** Suppose that after a system failure the contents of the undo log are as follows:

<START $T_1$>
<START $T_2$>
<$T_2$,$Z$,14>
<$T_2$,$Y$,5>
<$T_1$,$Y$,21>
<$T_1$,$Z$,3>
<START $T_3$>
<$T_2$,$X$,9>
<COMMIT $T_2$>
<$T_1$,$X$,2>
<$T_1$,$Z$,11>

What are the values of $X$, $Y$, and $Z$ on disk after the recovery manager has processed the log file?

☐ **A.** $X = 9$, $Y = 5$, $Z = 3$

☐ **B.** $X = 2$, $Y = 21$, $Z = 11$

☐ **C.** $X = 2$, $Y = 5$, $Z = 3$

☐ **D.** $X = 2$, $Y = 21$, $Z = 14$

■ **E.** None of the above

**20.** Which ACID property ensures that there can be no cascading rollbacks?

☐ **A.** Atomicity

☐ **B.** Consistency

■ **C.** Isolation

☐ **D.** Durability

☐ **E.** None of the above. Ensuring the ACID properties does not prevent cascading rollbacks

**21.** Suppose the wait-die scheme is used for deadlock prevention. Transaction $T_1$ is older than transaction $T_2$. Which transaction(s) are aborted after running the following schedule:

$$l_1(X), r_1(X), l_2(X)$$

☐ **A.** Both transactions are aborted.

☐ **B.** $T_1$ is aborted.

■ **C.** $T_2$ is aborted.

☐ **D.** None are aborted.

☐ **E.** This schedule cannot happen, since $T_1$ has the lock on $X$ and thus this is a deadlock.

**22.** Consider the following partial transaction, where lock and unlock operations were omitted:

$$\underset{(a)}{\_\_}\ \underset{(b)}{\_\_}\ r_1(X);\ \underset{(c)}{\_\_}\ w_1(X);\ \underset{(d)}{\_\_}\ \underset{(e)}{\_\_}\ r_1(Y);\ \underset{(f)}{\_\_}\ w_1(Y);\ \underset{(g)}{\_\_}\ c_1$$

Which of the following ensure that the transaction is two-phase locked (recall that $sl_1$ requests a shared lock, $xl_1$ an exclusive lock and $u_1$ releases a lock)?

**(I)** Insert $sl_1(X)$ at (a), $xl_1(X)$ at (b), $xl_1(Y)$ at (c), $u_1(X)$ at (f), and $u_1(Y)$ at (g)

**(II)** Insert $xl_1(X)$ at (a), $u_1(X)$ at (d), $xl_1(Y)$ at (e), and $u_1(Y)$ at (g)

**(III)** Insert $xl_1(X)$ at (a), $xl_1(Y)$ at (b), $u_1(X)$ at (c), and $u_1(Y)$ at (g)

■ **A.** (I) only
☐ **B.** (II) only
☐ **C.** (III) only
☐ **D.** All of the above
☐ **E.** None of the above

**23.** If in each transaction item $X$ ($X$ does not depend on the transaction, but is the same for all transactions, even if the transaction does not use $X$ at all) is locked before all reads and writes and first unlocked after commit, the scheduler is sometimes:

☐ **A.** Not recoverable
☐ **B.** Not cascadeless
☐ **C.** Deadlocked
☐ **D.** Not serialisable
■ **E.** None of the above answers are correct

**24.** Data transfer costs may be reduced by adopting a join strategy prior to transfer. The type of join is:

☐ **A.** Inner join
☐ **B.** Outer join
■ **C.** Semijoin
☐ **D.** Natural Join
☐ **E.** Cross/Cartesian product

**25.** An organisation uses a distributed database over two sites, A and B:

- Site A holds a relation *Products*(*product_id*, *product_name*, *price*, *quantity*, *store_id*). Each value of attribute *product_name* require 20 bytes, *store_id* require 5 bytes, and each value for each of the attributes *product_id*, *price* and *quantity* require 10 bytes.
- Site B holds a relation *Stores*(*store_id*, *store_name*, *city*). Values for attribute *store_id* require 5 byte each, and values for attributes *store_name* and *city* require 20 bytes each.

Assume the following:

- $|\pi_{\text{store\_id}}(\text{Stores})| = 1000$
- $|\pi_{\text{store\_id}}(\sigma_{\text{city='Liverpool'}}(\text{Stores}))| = 10$
- $|\text{Products} \ltimes \sigma_{\text{city='Liverpool'}}(\text{Stores})| = 10000$
- $|\sigma_{\text{city='Liverpool'}}(\text{Stores}) \ltimes \text{Products}| = 10$
- $|\text{Products}| = 100000$

To execute the query $\pi_{\text{product\_name,quantity,store\_id}}(\text{Products} \bowtie \sigma_{\text{city='Liverpool'}}(\text{Stores}))$ at site B, how many bytes have to be transferred between A and B at a minimum?

☐ **A.** $5 \cdot 10 + (20 + 5 + 10) \cdot 1000 = 35050$ bytes

■ **B.** $5 \cdot 10 + (20 + 5 + 10) \cdot 10000 = 350050$ bytes

☐ **C.** $5 \cdot 1000 + (20 + 5 + 10) \cdot 10000 = 355000$ bytes

☐ **D.** $5 \cdot 10000 + (20 + 5 + 10) \cdot 10000 = 400000$ bytes

☐ **E.** $(20 + 5 + 10) \cdot 100000 = 3500000$ bytes

| transaction_id | customer_id | item_bought |
|---|---|---|
| 1 | Anna | Apple, Kiwi, Strawberry |
| 2 | Brian | Apple, Strawberry |
| 3 | Chloe | Grape, Pineapple |
| 4 | Anna | Apple, Pineapple |
| 5 | Emma | Kiwi, Pineapple, Strawberry |
| 6 | Fred | Cherry, Strawberry |
| 7 | Brian | Kiwi |
| 8 | Anna | Strawberry |

Table 1: Table of customer transactions. Each transaction is a record of items bought.

**26.** Table 1 contains customer transactions. Each transaction is a record of items bought. What is the *support* with respect to the *transactions* for $J = \{Kiwi, Strawberry\}$?

☐ **A.** $\frac{1}{8}$

■ **B.** $\frac{2}{8}$

☐ **C.** $\frac{3}{8}$

☐ **D.** $\frac{4}{8}$

☐ **E.** Some other number

**27.** With reference to Table 1, what is the *support* with respect to the *customers* for:

$$J = \{Kiwi, Strawberry\}$$

☐ **A.** $\frac{0}{5}$

☐ **B.** $\frac{1}{5}$

☐ **C.** $\frac{2}{5}$

■ **D.** $\frac{3}{5}$

☐ **E.** Some other number

**28.** With reference to Table 1, find an association rule (AR) with a confidence of 100% with respect to customers for Apple, Strawberry $\rightarrow$ ? (i.e., what item is represented by ?).

☐ **A.** Cherry

☐ **B.** Grape

■ **C.** Kiwi

☐ **D.** Pineapple

☐ **E.** None, there is no 100% confidence.

**29.** With reference to Table 1 on the facing page, what are the values for *confidence with respect to transactions* (CT) and *confidence with respect to customers* (CC) for association rule Apple, Pineapple $\rightarrow$ Strawberry?

■ **A.** CT = 0 and CC = 1

☐ **B.** CT = $\frac{1}{2}$ and CC = $\frac{2}{3}$

☐ **C.** CT = 0 and CC = $\frac{2}{3}$

☐ **D.** CT = $\frac{1}{2}$ and CC = 1

☐ **E.** None of the above are correct

**30.** Given the following XQuery, what is the equivalent SQL statement?

```
FOR $v IN $doc//phonebook
WHERE $v/number = 01234567890
RETURN $v/name
```

☐ **A.** `SELECT phonebook FROM name WHERE number = 01234567890`

☐ **B.** `SELECT name, number FROM book WHERE number = 01234567890`

■ **C.** `SELECT name FROM phonebook WHERE number = 01234567890`

☐ **D.** `SELECT phonebook FROM number WHERE number = 01234567890`

☐ **E.** `SELECT number FROM phonebook WHERE name = 01234567890`