Assignment 02

Deadline: Day 22/03/2017 @ 23:59

**[Total Marks: 8/2 = 4]**

***Database management system***

***IT-344***

**Instructions:**

* This Assignment must be submitted on Blackboard via the allocated folder.
* Email submission will not be accepted.
* You are advised to make your work clear and well-presented, marks may be reduced for poor presentation.
* You MUST show all your work.
* Late submission will result in ZERO marks being awarded.
* Identical copy from students or other resources will result in ZERO marks for all involved students.
* Convert this Assignment to PDF just before submission.

Student Details:

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **Name:**###  **CRN:**### |  | **ID:**### |
|  |  |  |

# Question One

***2 Marks***

*Learning Outcome(s):*

**Consider the given query with multiple selection conditions; by taking into account the “Query Tuning”, please answer (a) and (b).**

SELECT ESSN, LName, BDATE, Salary

FROM EMPLOYEE

WHERE Salary > 7500 OR ESSN < 205;

1. **Give a brief explanation of why the above query may not be prompting the query optimizer to use any index?**

**Answer:**

Because the query was using multiple selection conditions *which were connected by(OR)*, this would make *the index ‘useless’* as it might run through all fields on the table rather than just using a particular index.

1. **Improve the above query in order to be able to use an index.**

**Answer:**

By using ‘UNION’ instead of ‘OR’ to allow using an index (i.e. Salary or ESSN)

|  |  |
| --- | --- |
| SELECT | ESSN, LName, BDATE, Salary |
| FROM | EMPLOYEE |
| WHERE | Salary > 8000 |
| UNION |  |
| SELECT | ESSN, LName, BDATE, Salary |
| FROM | EMPLOYEE |
| WHERE | ESSN < 205; |

# Question Two

***2 Marks***

*Learning Outcome(s):*

*Instructors: State the Learning Outcome(s) that match this question*

**Test the following schedule of three transactions for serializability. Also, describe that can we create its equivalent serial schedule. If yes then how else why?**

**S1: r2(A); r1(B); w2(A); r2(B); r3(A); w1(B); w3(A); w2(B)**

**Answer:**

Because it contains cycle then above schedule is not serializable. Precedence graph for above schedule contains cycles. Hence, it is not a serializable schedule. We cannot create a serial schedule of a non-serializable schedule because it contains cycles.

# Question Three

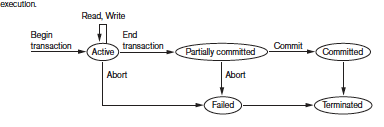
***2 Marks***

*Learning Outcome(s):*

*Instructors: State the Learning Outcome(s) that match this question*

**Draw the state transition diagram and explain the states for transaction execution.**

**Answer:**

****

* BEGIN\_TRANSACTION. This marks the beginning of transaction execution.
* READ or WRITE. These specify read or write operations on the database items that are executed as part of a transaction.
* END\_TRANSACTION. This specifies that READ and WRITE transaction operations have ended and marks the end of transaction execution. However, at this point it may be necessary to check whether the changes introduced by the transaction can be permanently applied to the database (committed) or whether the transaction must be aborted because it violates serializability or for some other reason.
* COMMIT\_TRANSACTION. This signals a successful end of the transaction so that any changes (updates) executed by the transaction can be safely committed to the database and will not be undone.
* ROLLBACK (or ABORT). This signals that the transaction has ended unsuccessfully, so that any changes or effects that the transaction may have applied to the database must be undone

# Question Four

***2 Marks***

*Learning Outcome(s):*

*Instructors: State the Learning Outcome(s) that match this question*

**Explain whether the following transactions T1 and T2 satisfy four variants of 2-Phase Locking protocol (2PL); Basic 2PL, Conservative 2PL, Strict 2PL, Rigorous 2PL. Write "Yes" or "No", motivate your answers as *why* or *why not* and be specific.**

**T2**

LOCK-S(A)

R(A)

LOCK-X(B)

UNLOCK(A)

R(B)

W(B)

COMMIT

UNLOCK(B)

**T1**

LOCK-S(A)

R(A)

LOCK-X(B)

R(A)

R(B)

B=A+B

UNLOCK(A)

W(B)

UNLOCK(B)

**(***here LOCK-S represents shared lock and LOCK-X represents exclusive lock***).**

**Answer: [** *0.25 Marks for each correct point* **]**

|  |  |  |
| --- | --- | --- |
| T1 | Basic 2PL | Yes / No: Yes  Growing and Shrinking phases are clearly mutually exclusively here. There is no unlocking during the locking phase and there is no locking during the unlocking phase. |
| Conservative 2PL | Yes / No**:** No  Conservative 2PL requires that all the locks are acquired before the transaction starts. This is not the case here and LOCK-X(B) is acquired after R(A). |
| Strict 2PL | Yes / No**:** No  Strict 2PL requires that the exclusive locks are only released after the commits or aborts. Although exclusive lock (LOCK-X(B)) is released at the end but T1 does not contain any commit or abort statement. |
| Rigorous 2PL | Yes / No**:** No  Rigorous 2PL is a variation of Strict 2PL. A transaction which is not Strict 2PL can never be a Rigorous 2PL. |
| **T2** | Basic 2PL | Yes / No**:** Yes  Growing and Shrinking phases are clearly mutually exclusively here. There is no unlocking during the locking phase and there is no locking during the unlocking phase. |
| Conservative 2PL | Yes / No**:** No  Conservative 2PL requires that all the locks are acquired before the transaction starts. This is not the case here and LOCK-X(B) is acquired after R(A). |
| Strict 2PL | Yes / No**:** Yes  Strict 2PL requires that the exclusive locks are only released after the commits or aborts. T2 contains a COMMIT operation and the exclusive lock (LOCK-X(B)) is released after it. |
| Rigorous 2PL | Yes / No**:** No  Rigorous 2PL furthers requires that all the locks (exclusive and shared) are only released after the commits or aborts. In T2, shared lock LOCK-S(A) is released before commit. |

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END OF QUESTIONS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***