INF20010 / 60014 Database Systems **Week 6 Tutorial Questions**

Lost Update Concurrency
Transactions & VB.NET Transactions

Tutorial Tasks

1. What is a statement-level rollback?

   The process of undoing the effects of a single SQL statement within a transaction

2. Describe a business scenario where a transaction has more than one SQL insert, update or delete statement to take the database from one consistent state to another.

   A customer transfers money from their savings account to their smart access account. The transaction includes two updates: deducting the amount from the savings account and adding it to the smart access account.

3. Using the previous example, what would happen if only part of the transaction was executed and committed? How does this compare to the first two ACID principles

   If only one part of the transaction (for example, the withdrawal from savings) is committed without the deposit in the smart access account, the account balances would be inconsistent. This violates the ACID principle of atomicity, which requires that a transaction either completes fully or not at all, and consistency, which ensures that a transaction brings the database from one consistent state to another.

4. What is the difference between transactions that are executed serially and a transaction schedule that is serializable?

   Transactions executed serially are processed one after the other, while a serializable schedule allows transactions to be interleaved but guarantees that the outcome is the same as if the transactions had been executed serially.

5. What is meant by the term Concurrent Transactions?

   Concurrent transactions refer to multiple transactions being executed at the same time

6. Describe how a Dirty Read may occur?

   A dirty read occurs when one transaction reads data that has been modified by another transaction but not yet committed.

7. What is another name for a Dirty Read?

   Uncommitted read

Lost Update Concurrency
Transactions & VB.NET Transactions

8. Imagine that Customer 123 has a balance of $100
   Transaction A is supposed to increase customer 123's balance by $40.
   Transaction B is supposed to decrease customer 123's balance by $10.
   Both procedures run concurrently executing statements is the order shown:

Statement	Transaction A	Transaction B
1	**vbalance number :=0;**	
2	**vcustid number := 123;**	
3		**vbal number :=0;**
4		**vcustid number := 123;**
5	**Begin**	
6		**Begin**
7	**SELECT balance** **INTO vbalance** **FROM customer** **WHERE custid = pcustid;**	
8		**SELECT balance** **INTO vbal** **FROM customer** **WHERE custid = pcustid;**
9	**vbalance := vbalance + 40;**	
10	**UPDATE customer** **SET balance = vbalance** **WHERE custid = pcustid;**	
11	**Commit;**	
12	**End;**	
13		**vbal := vbal - 10;**
14		**UPDATE customer** **SET balance = vbal** **WHERE custid = pcustid;**
15		**Commit;**
16		**End;**

At the completion of Transactions A and B, customer 123's balance should be $130

a) What is customer 123's actual balance at the end of statement 12?

   140

b) What is customer 123's actual balance at the end of statement 16?

   90

c) A problem has occurred. What name do we give to this problem?

   Lost Update

Lost Update Concurrency
Transactions & VB.NET Transactions

9. This scenario is identical to the question above.
   However, statements 7 & 8 below have been modified to include a FOR UPDATE clause.

   a. What is the effect of the FOR UPDATE clause in the SELECT statement?
   The FOR UPDATE locks the selected row so that no other transaction can modify or access it until the transaction holding the lock is either committed or rolled back.
   b. List the sequence in which these statements will be executed.
   1,2,3,4,5,6,7,9,10,11,8,13,14,15,16
   c. What is customer 123's balance at the end of statement 16: 130

Statement	Procedure A	Procedure B
1	vbalance number :=0;	
2		vbalance number :=0;
3	vcustid number := 123;	
4		vcustid number := 123;
5	Begin	
6		Begin
7	SELECT balance INTO vbalance FROM customer WHERE custid = pcustid FOR UPDATE;	
8		SELECT balance INTO vbalance FROM customer WHERE custid = pcustid FOR UPDATE;
9	vbalance := vbalance + 40;	
10	UPDATE customer SET balance = vbalance WHERE custid = pcustid;	
11	Commit;	
12	End;	
13		vbalance := vbalance - 10;
14		UPDATE customer SET balance = vbalance WHERE custid = pcustid;
15		Commit;
16		End;

10. State whether each of these statements is True or False:

    a. True / False. An employee row is locked via the Select…For Update statement. No other transactions can **read** data about that employee while the lock is in place F

    b. True / False. An employee row is locked via the Select…For Update statement. No other transactions can **update** data about that employee while the lock is in place T

    c. True / False. A transaction that contains a Select …For Update clause MUST also have an Update statement that modifies the locked row(s). F

    d. True / False. A transaction that contains a Select… For Update statement does NOT have to have ANY Insert, Update or Delete SQL statements T

    e. True / False. A Select… For Update lock remains in place until the row(s) is updated F

    f. True / False. A commit statement will unlock a row(s) locked by a Select… For Update statement T

Lost Update Concurrency

Transactions & VB.NET Transactions

g. True / False. A rollback statement will unlock a row(s) locked by a Select… For Update statement T

h. True / False. An exception that causes a transaction to end will cause a rollback statement T

INF20010 / 60014 Database Systems **Week 6 Tutorial Questions**

Lost Update Concurrency
Transactions & VB.NET Transactions

Consider the following VB Code:

```
1  Imports Oracle.DataAccess.Client
2  Public Class Form1
3      Private Sub TestOracleButton_Click(ByVal sender As System.Object, ByVal e As System.E
4          TestTrans()
5      End Sub
6      Public Sub TestTrans()
7          Dim rvConn As Oracle.DataAccess.Client.OracleConnection
8          rvConn = CreateConnection()
9          Dim vOutcome As String = ""
10         Dim rvTran As Oracle.DataAccess.Client.OracleTransaction = Nothing
11         Try
12             rvConn.Open()
13             rvTran = rvConn.BeginTransaction(IsolationLevel.ReadCommitted)
14             Update_Table_1(rvConn)
15             Update_Table_2(rvConn)
16             Update_Table_3(rvConn)
17             rvTran.Commit()
18             vOutcome = ("Transaction Finished OK")
19         Catch ex As Exception
20             rvTran.Rollback()
21             vOutcome = ex.Message
22         Finally
23             rvConn.Close()
24             MessageBox.Show(vOutcome)
25         End Try
26     End Sub
27     Public Sub Update_Table_1(ByVal rvConn As Oracle.DataAccess.Client.OracleConnection)
28         Dim rvCmd As New Oracle.DataAccess.Client.OracleCommand
29         rvCmd.Connection = rvConn
30         rvCmd.CommandType = CommandType.StoredProcedure
31         rvCmd.CommandText = "SP_UPDATE_TABLE1"
32         rvCmd.ExecuteNonQuery()
33     End Sub
```

Assume that code for sub procedures **update_table_2** & **update_table_3** are similar to lines 28-34 above.

11. What is the purpose of line 10? Declares rvTran as an OracleTransaction and initializes it to Nothing to ensure it's not referencing any transaction
12. What is the purpose of line 13? Begins a new transaction with the ReadCommitted isolation level
13. What happens when VB executes line 12 & 13? Opens a connection to the Oracle database, begins a new transaction on the open connection with the ReadCommitted isolation level.
14. What happens when VB executes line 14? Update_Table_1(rvConn) is called?
15. What would happen if line 33 causes a Raise_Application_Error within Oracle? It will trigger an exception in VB.NET
16. Do you think that sp_update_table1 & sp_update_table2 & sp_update_table3 will each contain a commit statement? Why / Why not?

    No, the stored procedures (sp_update_table1, sp_update_table2, and sp_update_table3) should not contain a commit statement. The reason is that the commit is handled at the transaction level in the calling VB.NET code (rvTran.Commit())

Lost Update Concurrency
Transactions & VB.NET Transactions

17. Do you think that sp_update_table1 & sp_update_table2 & sp_update_table3 will each contain a rollback statement? Why / Why not?

    No, the stored procedures should not contain rollback statements

18. What circumstances cause a Commit to occur in this code?

    A commit occurs when the Try block executes successfully without any exceptions. Specifically, the commit happens after the Update_Table_1, Update_Table_2, and Update_Table_3 procedures are executed and no exceptions are raised

19. What circumstances cause a Rollback to occur in this code?

    A rollback occurs when the Try block executes successfully without any exceptions. Specifically, the commit happens after the Update_Table_1, Update_Table_2, and Update_Table_3 procedures are executed and no exceptions are raised

20. What is the purpose of the Finally block of code?

    The Finally block ensures that the connection (rvConn) is closed regardless of whether the transaction succeeds or fails. This is important for resource management, as it guarantees that the connection is properly closed even if an exception occurs.

Lab Tasks

Continue with Assignment work