



Module 3: Performing Connected Database Operations

Overview

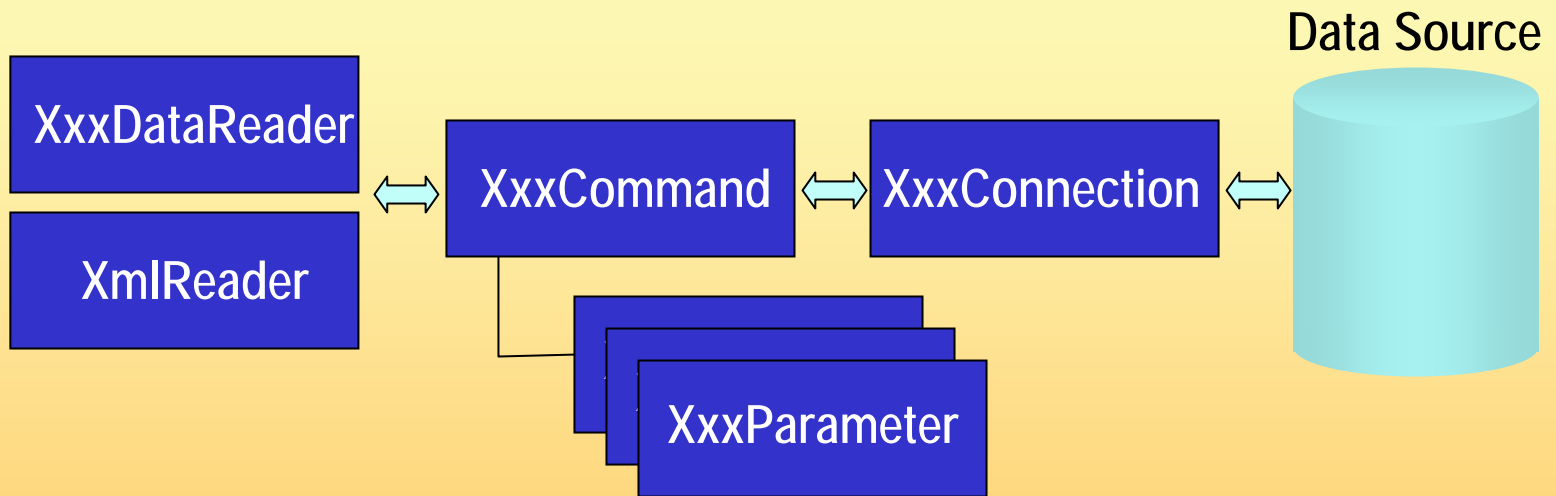
- Working in a Connected Environment
- Building Command Objects
- Executing Command Objects That Return a Single Value
- Executing Commands That Return Rows
- Executing Commands That Do Not Return Rows
- Using Transactions

Lesson: Working in a Connected Environment

- **Object Model for Connected Applications**
 - Typical scenarios for a connected environment
 - .NET Framework classes used in a connected environment application

Object Model for Connected Applications

Classes in a Connected Application



Lesson: Building Command Objects

- What Is a Command Object?
- How to Create a Stored Procedure
- How to Create a Command Object
- Demonstration: Creating a Command Object Graphically
- What Are Command Parameters?
- How to Create Parameters for a Command Object

What Is a Command Object?

- A command object is a reference to a SQL statement or stored procedure
- Properties
 - (Name), Connection, CommandType, CommandText, Parameters
- Methods
 - ExecuteScalar, ExecuteReader, ExecuteNonQuery
 - ExecuteXmlReader (SqlCommand only)

How to Create a Stored Procedure

■ Server Explorer

- On the View menu, click Server Explorer, or press Ctrl+Alt+S
- Create a data connection
- Click New Stored Procedure
- Insert SQL

■ Demonstration

- Creating a stored procedure
- Testing a stored procedure

How to Create a Command Object

- Programmatically

- Server Explorer

- On the View menu, click Server Explorer, or press Ctrl+Alt+S
- Drag stored procedure onto form or component

- Toolbox

- Use SqlConnection or OleDbConnection
- Use SqlCommand or OleDbCommand

Demonstration: Creating a Command Object Graphically



- In this Demonstration, you will see how to use a stored procedure that will return all products in the Northwind database that have not been discontinued

What Are Command Parameters?

■ Introduction

- SQL statements and stored procedures can have input and output parameters, and a return value
- Command parameters allow these parameters to be set and retrieved
- SqlParameter, OleDbParameter

■ Properties

- ParameterName, DbType, Size, Direction

How to Create Parameters for a Command Object

- Programmatically
- Code example

```
Dim prmName As New SqlParameter( _  
    "@CatName", SqlDbType.NChar, 15)  
prmName.Direction = ParameterDirection.Output  
cmdCountProds.Parameters.Add(prmName)
```

- Using the Visual Studio .NET graphical tools

Lesson: Executing Command Objects That Return a Single Value

- Why Return a Single Value in a Command?
- How to Execute a Command That Returns a Single Value
- How to Retrieve Output and Return Values

Why Return a Single Value in a Command?

- ADO.NET is more efficient than ADO, where a complete record set is returned
- Examples
 - Units in stock for a particular product
 - How many products?
 - COUNT, MAX, MIN, AVERAGE

How to Execute a Command That Returns a Single Value

- Call the ExecuteScalar method
 - ExecuteScalar returns a value of the type Object
 - Use CType or a cast, to convert into appropriate type
- Microsoft Visual Basic® code example:

```
cmProducts.Parameters("@ProdID").Value = 42
cnNorthwind.Open()
Dim qty As Integer = _
    CType(cmProducts.ExecuteScalar(), Integer)
cnNorthwind.Close()
```

How to Retrieve Output and Return Values

- How to get output parameters from a command

```
cmProducts.Parameters("@CatName").Value
```

- How to get the return value from a stored procedure

```
cmProducts.Parameters("@RETURN_VALUE").Value
```

- Code examples

Lesson: Executing Commands That Return Rows

- Returning Rows
- DataReader Properties and Methods
- How to Use a DataReader to Process Rows
- How to Execute Multiple SQL Statements

Returning Rows

- **DataReader**

- Read-only, forward-only, stream of rows

- **The ExecuteReader method**

- Returns a DataReader
- For example, SqlDataReader, OleDbDataReader

DataReader Properties and Methods

- Read method
 - Loads the next row
 - Returns true if a row exists, false if at end of rows
- Item property
- GetXxx methods – for example, GetString, GetInt32
- GetValues method
- IsDBNull method
- Close method

How to Use a DataReader to Process Rows

- Using a DataReader object to process a result set
- Code example

```
Dim cmProducts As New SqlCommand( _  
    "SELECT ProductName, UnitsInStock " & _  
    "FROM Products", cnNorthwind)  
cnNorthwind.Open()  
Dim rdrProducts As SqlDataReader  
rdrProducts = cmProducts.ExecuteReader()  
Do While rdrProducts.Read()  
    ListBox1.Items.Add(rdrProducts.GetString(0))  
Loop  
rdrProducts.Close()
```

How to Execute Multiple SQL Statements

- A stored procedure can contain multiple SQL statements
 - Group-related tasks
 - Encapsulate business rules
- If the stored procedure returns multiple result sets
 - Call `NextResult` to move to the next result set
- To determine how many rows were affected by the stored procedure
 - Use the `RecordsAffected` property

Lesson: Executing Commands That Do Not Return Rows

- What Are DDL and DCL Statements?
- How to Execute DDL and DCL Statements
- What Are DML Modification Statements?
- How to Execute DML Modification Statements
- Troubleshooting Data Modification

What Are DDL and DCL Statements?

- Definition

- Automate database administration tasks

- DDL and DCL statements

- CREATE, ALTER, DROP, GRANT, DENY, REVOKE

- Code example

```
CREATE PROCEDURE dbo.SummarizeProducts AS  
CREATE TABLE ProductSummary  
( ProductName nvarchar(40),  
  CategoryName nvarchar(15) )
```

How to Execute DDL and DCL Statements

- ExecuteNonQuery method
 - Returns count of rows affected
- Code example

```
cnNorthwind.Open()  
Dim affected As Integer = _  
    cmSummarizeProducts.ExecuteNonQuery()  
cnNorthwind.Close()  
MessageBox.Show("Records affected: " & _  
    affected)
```

What Are DML Modification Statements?

- Definition
 - Modify data in the database
- DML Statements
 - INSERT, UPDATE, DELETE
- Code example

```
CREATE PROCEDURE dbo.InsertRegion
    ( @RegID int,
      @RegName nchar(50) )
AS
    INSERT INTO Region VALUES (@RegID, @RegName)
```

How to Execute DML Modification Statements

- To execute a DML statement
 - ExecuteNonQuery method
- Code example

```
cnNorthwind.Open()  
Dim affected As Integer = _  
    cmPrices.ExecuteNonQuery()  
cnNorthwind.Close()  
MessageBox.Show("Records affected: " & _  
    affected)
```

Troubleshooting Data Modification

■ Common errors

- Incorrect object names
- Server unavailability
- Data integrity issues
- Using connection before it is open
- Invalid data types

Lesson: Using Transactions

- What Is a Transaction?
- How to Manage Transactions Using SQL Statements
- How to Manage Transactions Using ADO.NET
- What Are Isolation Levels?

What Is a Transaction?

- A transaction is a set of related tasks that either succeed or fail as a unit
- Two types of transactions
 - Local transactions
 - Distributed transactions

How to Manage Transactions Using SQL Statements

- SQL transaction statements
 - BEGIN TRANS, COMMIT TRANS, ROLLBACK TRANS
- Code example

```
BEGIN TRANS
DECLARE @orderDetailsError int, @productError int
DELETE FROM "Order Details" WHERE ProductID=42
SELECT @orderDetailsError = @@ERROR
DELETE FROM Products WHERE ProductID=42
SELECT @productError = @@ERROR
IF @orderDetailsError = 0 AND @productError = 0
    COMMIT TRANS
ELSE
    ROLLBACK TRANS
```

How to Manage Transactions Using ADO.NET

- XxxConnection – for example, SqlConnection
 - BeginTransaction
- XxxTransaction – for example, SqlTransaction
 - Commit
 - Rollback
- Code examples

What Are Isolation Levels?

- Examples of concurrency problems
- Guidelines for setting the isolation level
- Code example

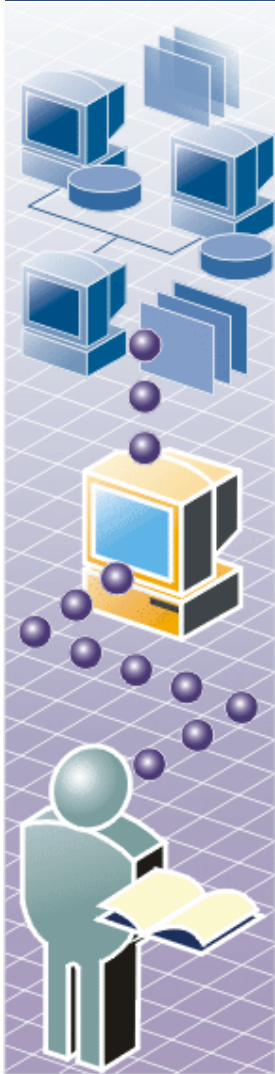
```
trans = cnNorthwind.BeginTransaction( _  
    IsolationLevel.Serializable)
```

- Support for isolation levels is dependent on which database you use

Review

- Working in a Connected Environment
- Building Command Objects
- Executing Commands That Return a Single Value
- Executing Commands That Return Rows
- Executing Commands That Do Not Return Rows
- Using Transactions

Lab 3.1: Performing Connected Database Operations



- Executing a Command Object That Returns a Single Value
- Executing a Command Object That Returns Records
- Executing a Command Object That Returns Multiple Results
- Executing a Command Object That Modifies the Database