.NET Technology

(503112)

By Mai Van Manh

Lab 05: ADO.NET

Exercise 1: Database Connection and Querying

Create a simple program that connects to a database and executes queries:

1. Set up a database (e.g., SQL Server, SQLite) and create a table with sample data.
2. Write C# code to establish a database connection using ADO.NET.
3. Create SQL queries to retrieve data from the database (e.g., SELECT statements).
4. Execute the queries and retrieve the results using ADO.NET SqlConnection, SqlCommand, and SqlDataReader.
5. Display the queried data in the console.

Exercise 2: Data Insertion and Updating

Practice inserting and updating data in the database:

1. Create a copy of your program source code from Exercise 1 and modify it to include SQL queries for inserting new records and updating existing records in the database.
2. Implement error handling and validation to ensure data integrity during insertion and updates.

Exercise 3: Using Parameters

Learn how to use parameters in SQL queries for security and performance:

1. Create a copy of your program source code from Exercise 2 and modify your program to use parameters in SQL queries instead of directly embedding values in the query strings.
2. Demonstrate the use of parameterized queries with different data types (e.g., integers, strings, dates).

Exercise 4: Stored Procedures

Explore stored procedures in ADO.NET:

1. Create a stored procedure in your database (e.g., SQL Server) that performs a specific task, such as retrieving data or updating records.
2. Create a copy of your program source code from Exercise 3 and modify your C# program to execute the stored procedure using ADO.NET.
3. Pass parameters to the stored procedure and retrieve the results.

Exercise 5: Data Access Layer (DAL)

Create a Data Access Layer for your application:

1. Create a copy of your program source code from Exercise 4 and refactor your database access code into a separate class or set of classes that encapsulate database interactions.
2. Implement methods in the DAL for common database operations, such as selecting, inserting, updating, and deleting records.
3. Use the DAL in your application to interact with the database.

Exercise 6: Transactions

Practice implementing transactions in ADO.NET:

1. Create a scenario where multiple database operations (e.g., insert and update) need to be performed as part of a single transaction.
2. Use ADO.NET to start, commit, or rollback transactions as needed.
3. Handle exceptions and errors that may occur during transaction processing.

Exercise 7: A Complete Console Application

Apply what you've learned to write a basic student management program on the console. The program will read and write data in a database using ADO.NET. When the program runs for the first time, it will read and display the list of students read from the database (id, name, age, score, email). The program

then displays a menu for the user to select the function to perform by entering an integer corresponding to the options in the menu.

The program menu needs to have the following functions:

1. Displays a list of all students
2. add a new student. The program then asks the user to enter the student information that needs to be added one by one.
3. Search for students by any criteria in that student's attributes. The result of the search is a list of students matching the search keyword.
4. Sort the student list by a certain criterion
5. Delete a student by id

After each user's choice, the program needs to execute the request and notify the corresponding result. If an error occurs, the program needs to report the appropriate error message. After completing a request, the program continues to let the user choose other requests until the user wants to exit (for example when they enter the word exit).