**Question 1.**

**Scenario:**  
You are developing a financial management application for a small business. The application needs to store information about transactions, including the transaction date, amount, type (credit/debit), and a description.

**Question:**  
Design the database schema for this application, considering the need for scalability and future reporting capabilities. Write C# code to create the necessary database tables and insert a sample transaction into the database.

**Expected Input and Output:**

* **Database Schema:**

sql

CREATE TABLE Transactions (

TransactionID INT PRIMARY KEY IDENTITY(1,1),

TransactionDate DATETIME NOT NULL,

Amount DECIMAL(10, 2) NOT NULL,

Type NVARCHAR(10) NOT NULL CHECK (Type IN ('Credit', 'Debit')),

Description NVARCHAR(255)

);

* **C# Code to Insert a Sample Transaction:**

using System;

using System.Data.SqlClient;

class Program

{

static void Main()

{

string connectionString = "your\_connection\_string";

using (SqlConnection connection = new SqlConnection(connectionString))

{

string query = "INSERT INTO Transactions (TransactionDate, Amount, Type, Description) " +

"VALUES (@TransactionDate, @Amount, @Type, @Description)";

SqlCommand command = new SqlCommand(query, connection);

command.Parameters.AddWithValue("@TransactionDate", DateTime.Now);

command.Parameters.AddWithValue("@Amount", 150.00);

command.Parameters.AddWithValue("@Type", "Credit");

command.Parameters.AddWithValue("@Description", "Payment received");

connection.Open();

int result = command.ExecuteNonQuery();

if (result > 0)

Console.WriteLine("Transaction inserted successfully.");

}

}

}

* **Expected Output:**

Transaction inserted successfully.

**Question 2.**

**Scenario:**  
You are tasked with designing a database for an online bookstore. The bookstore needs to store information about books, authors, customers, and orders. Additionally, you need to ensure that the database is normalized to reduce redundancy and improve data integrity.

**Question:**  
Outline the database design, including the relationships between tables (1-to-1, 1-to-many, and many-to-many). Write the SQL script to create these tables in the database.

**Expected Input and Output:**

* **Database Design:**
  + **Books Table:** Stores information about each book.
  + **Authors Table:** Stores information about authors.
  + **Customers Table:** Stores customer details.
  + **Orders Table:** Stores order information.
  + **BookAuthors Table:** (Many-to-Many relationship between Books and Authors).
* **SQL Script:**

sql

CREATE TABLE Books (

BookID INT PRIMARY KEY IDENTITY(1,1),

Title NVARCHAR(255) NOT NULL,

ISBN NVARCHAR(13) NOT NULL UNIQUE,

Price DECIMAL(10, 2) NOT NULL

);

CREATE TABLE Authors (

AuthorID INT PRIMARY KEY IDENTITY(1,1),

FirstName NVARCHAR(100) NOT NULL,

LastName NVARCHAR(100) NOT NULL

);

CREATE TABLE Customers (

CustomerID INT PRIMARY KEY IDENTITY(1,1),

FirstName NVARCHAR(100) NOT NULL,

LastName NVARCHAR(100) NOT NULL,

Email NVARCHAR(255) NOT NULL UNIQUE

);

CREATE TABLE Orders (

OrderID INT PRIMARY KEY IDENTITY(1,1),

CustomerID INT FOREIGN KEY REFERENCES Customers(CustomerID),

OrderDate DATETIME NOT NULL,

TotalAmount DECIMAL(10, 2) NOT NULL

);

CREATE TABLE BookAuthors (

BookID INT FOREIGN KEY REFERENCES Books(BookID),

AuthorID INT FOREIGN KEY REFERENCES Authors(AuthorID),

PRIMARY KEY (BookID, AuthorID)

);

* **Expected Output:**

Database tables created successfully.

**Question 3**

**Scenario:**  
You are building a module in your application that displays a list of all products in an inventory management system. The products are stored in a SQL Server database. The module should connect to the database, retrieve the product details, and display them in a console application.

**Question:**  
Write a C# program using ADO.NET to connect to the database, retrieve all products, and display them in the console. Include exception handling for database connection errors.

**Expected Input and Output:**

* **C# Code:**

using System;

using System.Data;

using System.Data.SqlClient;

class Program

{

static void Main()

{

string connectionString = "your\_connection\_string";

using (SqlConnection connection = new SqlConnection(connectionString))

{

try

{

connection.Open();

SqlCommand command = new SqlCommand("SELECT \* FROM Products", connection);

SqlDataReader reader = command.ExecuteReader();

Console.WriteLine("Product List:");

while (reader.Read())

{

Console.WriteLine($"ID: {reader["ProductID"]}, Name: {reader["ProductName"]}, Price: {reader["Price"]}");

}

}

catch (SqlException ex)

{

Console.WriteLine("Database connection error: " + ex.Message);

}

}

}

}

* **Expected Output:**

Product List:

ID: 1, Name: Laptop, Price: 1200.00

ID: 2, Name: Mouse, Price: 25.00

**Question 4.**

**Scenario:**  
You are developing an HR management system that requires access to employee records stored in a SQL Server database. You need to write a function that connects to the database, retrieves the details of an employee based on their EmployeeID, and returns these details as an object.

**Question:**  
Write a C# function using ADO.NET to connect to the database and retrieve the details of an employee. The function should take an EmployeeID as input and return an Employee object containing the employee's details.

**Expected Input and Output:**

* **C# Code:**

public class Employee

{

public int EmployeeID { get; set; }

public string FirstName { get; set; }

public string LastName { get; set; }

public string Position { get; set; }

public decimal Salary { get; set; }

}

public Employee GetEmployeeById(int employeeId)

{

string connectionString = "your\_connection\_string";

Employee employee = null;

using (SqlConnection connection = new SqlConnection(connectionString))

{

string query = "SELECT \* FROM Employees WHERE EmployeeID = @EmployeeID";

SqlCommand command = new SqlCommand(query, connection);

command.Parameters.AddWithValue("@EmployeeID", employeeId);

connection.Open();

SqlDataReader reader = command.ExecuteReader();

if (reader.Read())

{

employee = new Employee

{

EmployeeID = (int)reader["EmployeeID"],

FirstName = reader["FirstName"].ToString(),

LastName = reader["LastName"].ToString(),

Position = reader["Position"].ToString(),

Salary = (decimal)reader["Salary"]

};

}

}

return employee;

}

* **Expected Output:**

Employee employee = GetEmployeeById(1);

Console.WriteLine($"Name: {employee.FirstName} {employee.LastName}, Position: {employee.Position}, Salary: {employee.Salary}");

Name: John Doe, Position: Manager, Salary: 90000.00

**Question 5**

**Scenario:**  
You are implementing a feature in a school management system to calculate the average grade of students in a particular course. The grades are stored in a SQL Server database. Your task is to execute a SQL command that calculates the average grade for a given course and displays it.

**Question:**  
Write a C# function that connects to the database, executes a command to calculate the average grade for a specified course, and returns the result. Include error handling to manage potential issues.

**Expected Input and Output:**

* **C# Code:**

public decimal GetAverageGrade(int courseId)

{

string connectionString = "your\_connection\_string";

decimal averageGrade = 0;

using (SqlConnection connection = new SqlConnection(connectionString))

{

string query = "SELECT AVG(Grade) FROM StudentGrades WHERE CourseID = @CourseID";

SqlCommand command = new SqlCommand(query, connection);

command.Parameters.AddWithValue("@CourseID", courseId);

connection.Open();

object result = command.ExecuteScalar();

if (result != DBNull.Value)

{

averageGrade = Convert.ToDecimal(result);

}

}

return averageGrade;

}

* **Expected Output:**

decimal averageGrade = GetAverageGrade(101);

Console.WriteLine($"The average grade for course 101 is: {averageGrade}");

The average grade for course 101 is: 85.75