SQL Server

DDL Statements

1. Create a database name “BookStoreDB” and create following tables in the database with the below mentioned columns using SQL Server.

    I. Author – AuthorId, AuthorName, DateOfBirth, State, City, Phone.
    II. Publisher – PublisherId, PublisherName, DateOfBirth, State, City, Phone.
    III. Category – CategoryId, CategoryName, Description.
    IV. Book – BookId, Title, Description, Price, ISBN, PublicationDate, Image.
    V. Order – OrderId, Date, Quantity, UnitPrice, ShippingAddress.

2. Modify the BookStoreDB and add all the required keys. Establish relationships among tables and apply following business rules.

    I. A Book can have multiple authors.
    II. An Author can write more than one book.
    III. A Book belongs to only one category.
    IV. A Book can be published by only one publishing house.
    V. An order can be placed for a single book but multiple quantities.

SELECT Statements

3. Write the appropriate SQL queries against BookStoreDB database to support following operations:

    a. Get All the books written by specific author
    b. Get all the books written by specific author and published by specific publisher belonging to “Technical” book Category
    c. Get total books published by each publisher.
    d. Get all the books for which the orders are placed.

Stored Procedures

4. Write the following stored procedure using SQL Server in BookStoreDB database to support following operations:

    a. Get All the books written by specific author
    b. Get all the books written by specific author and published by specific publisher belonging to “Technical” book Category
    c. Get total books published by each publisher.
    d. Insert a particular book
    e. Update a particular book by id
    f. Delete a particular book by id

Triggers

5. Let's assume that we have a table name "Book_History" table. If a particular book is deleted from the "Book" table, an entry with same book records to "Book_History" table must take place. Automate this process using trigger.

6. The "Book" table got an attribute "Price". Let's assume that we have a business requirement where we must ensure that the "Price" should not be less than 1. If any insert or update statement tries to make the "Price" less than 1, the SQL Server must terminate such insert or update statements. Write an appropriate trigger to implement the business requirement.

7. Create a trigger on the table "Order" and add the following functionalities. When a new order is placed, it should check whether the required quantity is available in the "Book" table. If not, it should show appropriate message and the insert statement to "Order" table should be terminated. If the quantity in book table is sufficient, it should deduct the quantity ordered from the quantity in hand in the book table and update the quantity.

Classes and Objects

Create a console application name "BookStoreApp" using C#. Write the following classes with properties. Use Automatic Properties and use suitable access modifiers for the class and properties. All the properties should have an appropriate data types.

Author – AuthorId, AuthorName, DateOfBirth, State, City, Phone.

Publisher – PublisherId, PublisherName, DateOfBirth, State, City, Phone.

Category – CategoryId, CategoryName, Description.

Book – BookId, Category, Title, Author, Publisher, Description, Price, ISBN, PublicationDate.

Collections and Generics

Use the "BookStoreApp" and create your static data store using Generic Collections. Create multiple Book objects and store using collection. Write methods to support all

the operations against the collection based on the below menu. The program should display a menu as following pattern to the user. According to user's choice the program should perform a specific operation and display appropriate output.

==========

Main Menu

==========

1. Adding new books
2. Displaying all books
3. Displaying a book by BookId
4. Updating a book by BookId
5. Deleting a book by BookId
6. Exit

Enter Your Choice (any number from 1 to 6)

Exception Handling

Modify the above BookStoreApp and implement appropriate exception handling logic. The program must handle specific exceptions using responsible handler. The program should also able to handle any exception in case specific handler is not present.

Data Validation

Modify the above BookStoreApp and use appropriate data validation to validate each and every user input. In case of invalid user input, the program should display appropriate message to help user identify his/her mistake and enter valid value.

Serialization

Modify the above BookStoreApp and write appropriate methods for the following requirements:

a. A method should create a file in the hard disk to serialize Book objects and save the objects in XML format.

b. A method should de-serialize the Book objects saved as XML data and consume in the application.

ADO.NET

1. Use the "BookStoreApp" and the "BookStoreDB" you have already created during previous assignments. Create a Data Access Layer for the application. You should call the stored procedure wherever required that you have already created during SQL Server assignments. The DAL should expose appropriate methods for the following operations:

   I. Connect to the BookStoreDB
   II. Retrieve all books and display in the application
   III. Retrieve a book by BookId
   IV. Add a new book
   V. Update an existing book
   VI. Delete an existing book
   VII. Get All the books written by specific author
   VIII. Get all the books written by specific author and published by specific publisher belonging to "Technical" book Category
   IX. Get total books published by each company.

ASP.NET MVC and Web API

Develop an application name "BookStoreApp" using ASP.NET MVC and Entity Framework. The tools to develop this application should be VS 2013 and use LocalDB (SQL Server 2012) to attach your database to the application. The application should meet the following requirements.

Phase-I

The application should have the following entities and attributes:

1. Author – AuthorId, AuthorName, DateOfBirth, State, City, Phone.
2. Publisher – PublisherId, PublisherName, DateOfBirth, State, City, Phone.
3. Category – CategoryId, CategoryName, Description.
4. Book – BookId, CategoryId, Title, AuthorId, PublisherId, Description, Price, ISBN, PublicationDate, Image.

The application should support two roles:

- User
- Administrator.

An administrator role can perform the following tasks.

1. The application should allow an administrator to
    a. Add new books,
    b. Edit and
    c. Delete existing books.
2. The view used for create and edit any entity should be same.
3. The application should have following validations. The validations should be on client side as well as server side.
    a. ISBN of the book should be valid.
    b. The date of publication for the book should not be the future date. The idea is to add custom validations.
4. The application should expose Web APIs which could be used by third party applications to get book details.

Phase-II

A user role can perform the following tasks.

1. The application should allow users to browse all the books by category.
2. The application should have functionality to search the books by title, author, and publisher. This should be done using Ajax such that only the search results section gets refreshed.
3. Users can view any single book details such as book name, category name, publisher name, author name and price.
4. While viewing book details users can add the book to their shopping cart.
5. The shopping cart should allow the user to review their cart and user can remove any item from their cart.
6. The application should ask users to login as soon as the users finally want to place an order (Use Identity).
7. If the user is not a registered user, the application should allow user to register (Use Identity).
8. After the user logged in or completed the registration process, the application should ask the shipping information from the user.
9. Finally when the user submits the order, the application should display confirmation message to the user.
10. The user should be able to add review for the book. These reviews should be displayed on book details view. Submitting review should be an Ajax action.
11. The application should expose all the operations through a Web API that can be called from any type of client applications.