Prueba de Desarrollo

La prueba consiste en desarrollar un Web Service (ASMX o WCF), capaz de recibir un XML en formato *string*, el cual debe ser convertido a una entidad en donde cada nodo representa un atributo de esta entidad, asimismo debe existir la funcionalidad para validar varios campos de forma que, si el XML enviado no cumple con estos requisitos, debe devolver una respuesta reflejando que el XML fue rechazado.

Detalles

Lenguaje: C#
Formato del Web Service: ASMX o WCF
Formato de entrada: Llamada al web service usando XML en formato String
Formato de salida: JSON;
Formato de entrega: Todos los archivos tienen que venir dentro de un .zip. Este Web Service no debe que ser publicado, nosotros lo vamos a ejecutar internamente para poder validar todo el funcionamiento.

Validaciones:

- Los nodos “date_dob_d”, “date_dob_m”, “date_dob_y”, corresponden a la fecha de nacimiento, por lo tanto, uniendo estos valores y utilizando el formato “MM/DD/YYYY” se debe validar que no sea una persona menor a 18 años.
- El nodo de “employer_length_months” debe ser mayor a 12.
- El nodo “loan_amount_desired” no puede ser menor a 200 ni mayor a 500.
- El nodo “home_state” siempre debe tener largo de 2 caracteres, no puede ser menos ni mayor, de lo contrario es un XML rechazado. Además, el valor dentro del nodo no puede ser ninguno de los siguientes: CA, FL, NY, ND, GA, NC.
- El nodo “income_frequency” solo puede ser alguno de los siguientes valores: Weekly, Biweekly, Monthly, Twice. En caso contrario el XML no debe ser aceptado.
- El nodo “Income_amount” es la representación del salario, el cual está ajustado según el “income frequency” Por ejemplo si el salario es de $500 y la frecuencia es weekly, esto se tomaría como que la persona gana $500 por semana. Por ende, en el caso de que income_frequency sea weekly, Biweekly, Twice debe ser ajustado a monthly. Asimismo, se debe validar que el salario mensual solo puede estar +- 15% de $2000, de lo contrario será rechazado.

Weekly: 4 pagos al mes
Biweekly y Twice: 2 pago al mes.
Monthly: 1 vez al mes.

Respuestas esperadas.
En caso de fallar alguna de las validaciones mencionadas el Web Service debe retornar el mensaje: "El XML no cumple con el formato válido + *descripción del error*", este puede ser en formato JSON o XML.
En caso de fallar por algún error desconocido, se tiene que manejar la excepción.
En caso de ser aceptado se debe retornar el mensaje: **"El XML ID: [customer_id], fue aceptado con el nombre [name_first] [name_last]"**

Notas
* customer_id, name_first, name_last: Corresponden a un nodo dentro del xml.
* Entidad: es equivalente a una Clase

XML Base
El siguiente XML es el que se debe utilizar para la implementación, este mismo será utilizado para evaluar el funcionamiento. El número de nodos y los nombre de estos no van a variar, lo único que cambia es el valor dentro de cada nodo.

<?xml version="1.0" encoding="UTF-8"?> <tss_loan_request> <data name="sub_id">7545320</data> <data name="bank_account_type">CHECKING</data> <data name="bank_name">TRUSTONE FINANCIAL FEDERAL CR UN</data> <data name="client_url_root">lendyou.com</data> <data name="customer_id">60080928</data> <data name="employer_length_months">12</data> <data name="employer_name">walgreens</data> <data name="ext_work" /> <data name="home_city">Milwaukee</data> <data name="home_state">WI</data> <data name="home_street">4758 n 30th st</data> <data name="home_zip">53209</data> <data name="income_date1_d">19</data> <data name="income_date1_m">01</data> <data name="income_date1_y">2018</data> <data name="income_date2_d">02</data> <data name="income_date2_m">02</data> <data name="income_date2_y">2018</data> <data name="income_frequency">WEEKLY</data> <data name="income_ amount ">570</data> <data name="income_type">EMPLOYMENT</data> <data name="loan_amount_desired">500</data> <data name="military">FALSE</data> <data name="name_first">Lisa</data> <data name="name_last">Price</data> <data name="name_middle" /> <data name="name_title">MR</data>
<data name="phone_home">7981541924</data><data name="phone_cell">1541541924</data> <data name="date_dob_d">7</data><data name="date_dob_m">7</data><data name="date_dob_y">1984</data><data name="residence_length_months">12</data> <data name="residence_type">OWN</data> <data name="ssn_part_1">000</data> <data name="ssn_part_2">00</data> <data name="ssn_part_3">0000</data> <data name="state_id_number">0000406</data> <data name="state_issued_id">WI</data> <data name="bank_check_number" /> </tss_loan_request>

Prueba de SQL

¿Cuál sería el resultado de la siguiente Consulta?

```
 SELECT TOP 10 [BusinessEntityID]
       ,[PersonType]
       ,[NameStyle]
       ,[Title]
       ,[FirstName] +  ' ' + [LastName] as FullName
       ,[MiddleName]
   FROM [AdventureWorks2012].[Person].[Person]
   WHERE TITLE =  NULL
   ORDER BY [BusinessEntityID] DESC
```

¿Cuál sería el resultado de la siguiente Consulta?

```
WITH  PersonWithEMType AS (
SELECT TOP 10 [BusinessEntityID]

      ,[PersonType]
      ,[NameStyle]
      ,[Title]
      ,[FirstName] +  ' ' + [LastName] as FullName
      ,[MiddleName]
  FROM [AdventureWorks2012].[Person].[Person]
  WHERE PersonType  = 'EM'

 )


 SELECT FullName,[PhoneNumber]  FROM PersonWithEMType
 LEFT JOIN [AdventureWorks2012].[Person].[PersonPhone]
 ON PersonWithEMType.[BusinessEntityID] = [PersonPhone].[BusinessEntityID]
 WHERE [PhoneNumber] IS NULL
```

Explique el resultado de la siguiente consulta:

```
DECLARE @variable VARCHAR(10) = '01/01/2020';

SET @variable = @variable + ' 00:00 AM';

SELECT @variable AS Variable;
```

¿Cuál es la diferencia entre un INNER JOIN y un CROSS APPLY?

Si tenemos las siguientes tablas:

tabla:	**vendedor**
vendedor	nom_vendedor
V01	Pedro
V02	Luisa

tabla:	**factura_enc**	Datos del encabezado de la factura	
num_factura	cliente	total	vendedor
1	C1	915.50	V01
2	C2	3,195.00	
3	C3	1,364.00	V02

tabla:	**factura_det**	Líneas de la factura		
num_factura	cod_articulo	cantidad	precio	Total linea
1	A1	2.5	125.00	312.50
1	A2	3	201.00	603.00
2	A4	2	421.50	843.00
2	A6	4	200.50	802.00
2	A7	5	310.00	1,550.00
3	A1	3	220.00	660.00
3	A4	2	50.50	101.00
3	A6	3	201.00	603.00

1. Escriba la consulta SQL que seleccione el número de factura, cliente y el precio más grande utilizado en cada factura.

2. Escriba la consulta para seleccionar TODAS las facturas (3 facturas), mostrando el número de factura, cliente, total, vendedor y el nombre del nom_vendedor.
   NOTA: Observe que la factura 2 no tiene vendedor, pero igual se requiere que sea parte del resultado de la consulta.

3. Escriba una consulta que seleccione el número de factura, y la suma del campo “total línea”.