Documento: Estándar Para Las APIs De Exportación De Datos De Lotería.

API REST EXPORTACIONES LOTERÍAS V1

(JSON – XML)

Autor: Ing. Ben Gonzalez

Fecha: Julio, 2025

Versión: v1.2

Repositorios:

GitHub: https://github.com/Ben-Gonzalez-97/api-exportacion-loterias-std

GENERALIDADES:

1. Alcance del Documento:

   Esta especificación define un estándar para las APIs de exportación de datos de lotería, diseñado para ser implementado por diversos sistemas con arquitecturas heterogéneas. El objetivo es homogeneizar la forma en que los datos son expuestos, permitiendo que cada sistema elija la representación de datos (JSON o XML) que mejor se adapte a su entorno existente.
   Los implementadores deberán asegurar que su API cumpla con las estructuras de datos definidas para el formato elegido (JSON, XML o ambos) en cada uno de los endpoints especificados.

2. Especificación de Interfaz:

   - Esquema de Arquitectura: La API se adhiere a los principios de una arquitectura RESTful, utilizando métodos HTTP estándar para operaciones sobre recursos identificables. La comunicación se realiza a través de HTTPS para garantizar la seguridad. Los datos se intercambian en formato JSON o XML, según la implementación elegida por el sistema proveedor de la API.
     La flexibilidad reside en que la "API Implementada" puede exponer sus datos en JSON, XML, o ambos, siguiendo estrictamente las estructuras de datos definidas en este documento para cada formato.

   - Formatos de Datos Soportados: Esta API está diseñada para la máxima flexibilidad e interoperabilidad. Los implementadores tienen la opción de exponer sus endpoints utilizando:
     - JSON (JavaScript Object Notation): Un formato ligero y ampliamente adoptado, recomendado para nuevas implementaciones y sistemas modernos.
     - XML (Extensible Markup Language): Un formato estructurado, provisto

para asegurar la compatibilidad con sistemas legados que ya operan con estructuras basadas en XML y requieren homogeneidad en su arquitectura existente.

Si bien este estándar define y soporta tanto JSON como XML para garantizar la máxima compatibilidad, se recomienda enfáticamente el uso de JSON para todas las nuevas implementaciones. JSON ofrece un rendimiento superior y una mayor facilidad de integración con los ecosistemas de desarrollo modernos. El soporte para XML se mantiene principalmente para facilitar la transición y asegurar la interoperabilidad con sistemas legados existentes.

Importante: La elección del formato (JSON o XML) recae en el implementador de la API. Se espera que la API implementada devuelva el formato preferido por el sistema al que está sirviendo o, si lo desea, sea capaz de responder en ambos formatos si el consumidor lo indica (a través de la cabecera Accept). Para el propósito de este estándar, se definen las estructuras esperadas para ambos formatos.

- Convenciones de Nomenclatura: Para asegurar la consistencia en todas las implementaciones:
    - URLs de Endpoints: Se utilizará kebab-case para los segmentos de URL (ej., api/v1/auth/token).
    - Nombres de Propiedades (JSON) y Parámetros de Consulta: Se utilizará camelCase (ej., tokenType, expiresIn).
    - Nombres de Elementos (XML):
        - Elementos raíz y de objetos complejos: PascalCase (ej., <AuthResponse>, <ErrorResponse>).
        - Elementos anidados y de propiedades: camelCase (ej., <tokenType>, <username>).
        - Colecciones: Se representarán con un elemento contenedor (ej., <salidaConsulta>) que contendrá múltiples elementos hijos (ej., <item>).

- Codificación de Caracteres: Todas las peticiones y respuestas deben utilizar la codificación de caracteres **UTF-8**.

- Tipos de Datos: La API maneja los siguientes tipos de datos. Los implementadores deben mapear sus tipos de datos internos a estas representaciones estándar en JSON y XML:

Tipo de Datos	Descripción	JSON	XML (Ejemplo)
String	Cadena de caracteres Unicode.	"valor"	<elemento>valor</elemento>
Integer	Número entero sin decimales.	123	<elemento>123</elemento>
Decimal	Número con decimales (flotante de doble precisión).	123.45	<elemento>123.45</elemento>
Boolean	Valor lógico.	true / false	<elemento>true</elemento>
Date (YYYY-MM-DD)	Fecha en formato ISO 8601.	"YYYY-MM-DD"	<elemento>YYYY-MM-DD</elemento>
Array / List	Colección ordenada de elementos.	[item1, item2]	<Coleccion><item1>...</item1><item2>...</item2></Coleccion>
Object / Structure	Conjunto de pares clave-valor o elementos anidados.	{ "key": "value" }	<Objeto><key>value</key></Objeto>

- Manejo de Valores Nulos:
    - JSON: Los valores ausentes o nulos deben ser explícitamente representados con null (ej., "juego": null).
    - XML: Los valores nulos deben ser representados por un elemento vacío (ej., <juego/>) o, preferiblemente, utilizando el atributo xsi:nil="true" (ej., <juego xsi:nil="true"/>) si se utilizan esquemas XSD y se desea una mayor robustez.

DEFINICIÓN DE ENDPOINTS

En esta sección se describirán los endpoints de la API, incluyendo el método HTTP, la URL, los parámetros de consulta (si aplica), y las estructuras de peticiones y respuestas para JSON y XML.

EndPoint para la autentificación:

Mediante el método **POST**:

```
https://{{ip:puerto}}/api/v1/auth/token
```

JSON Body de la Petición:

```
{
  "username": "nombre-de-usuario",
  "password": "contraseña-secreta"
}
```

Respuesta Exitosa (200 OK)

```
{
  "tokenType": "Bearer",
  "expiresIn": 604800,
  "accessToken": "1ksUJKdfGJKuJYiw78we3gkjkkjhjkdsJHHJJHJFDDD"
}
```

XML Body de la Petición:

```
<AuthRequest>
  <username>nombre-de-usuario</username>
  <password>contraseña-secreta</password>
</AuthRequest>
```

Respuesta Exitosa (200 OK)

```
<AuthResponse>
  <tokenType>Bearer</tokenType>
  <expiresIn>604800</expiresIn>
  <accessToken>1ksUJKdfGJKuJYiw78we3gkjkkjhjkdsJHHJJHJFDDD</accessToken>
</AuthResponse>
```

Nota 0: Para las posteriores consultas se requiere un token de tipo Bearer en la cabecera de autorización.

- **Header**: Authorization
- **Value**: Bearer <<token>>

EndPoint para la lista de tipo de lotería:

Mediante el método **GET**: (Autenticación)

```
https://{{ip:puerto}}/api/v1/tipo-loteria
```

JSON Respuesta Exitosa (200 OK)

```json
{
  "tipoLoteria": [
    "triple",
    "terminal",
    "recargas",
    "multijuego",
    "juego-electronico",
    "animalito"
  ]
}
```

XML Respuesta Exitosa (200 OK)

```xml
<TiposLoteriaResponse>
  <tipoLoteria>triple</tipoLoteria>
  <tipoLoteria>terminal</tipoLoteria>
  <tipoLoteria>recargas</tipoLoteria>
  <tipoLoteria>multijuego</tipoLoteria>
  <tipoLoteria>juego-electronico</tipoLoteria>
  <tipoLoteria>animalito</tipoLoteria>
</TiposLoteriaResponse>
```

EndPoint para la lista de juegos:

Mediante el método **GET**: (Autenticación)

```
https://{{ip:puerto}}/api/v1/juegos
```

JSON Respuesta Exitosa (200 OK)

```json
{
  "juegos": [
    "activo",
    "bomba",
    "caliente",
    "granjita",
    "lotto-activo",
    "ruleta-activa",
    "uneloton",
    "zulia"
  ]
}
```

XML Respuesta Exitosa (200 OK)

```xml
<JuegosResponse>
  <juego>activo</juego>
  <juego>bomba</juego>
  <juego>caliente</juego>
  <juego>granjita</juego>
  <juego>lotto-activo</juego>
  <juego>ruleta-activa</juego>
  <juego>uneloton</juego>
  <juego>zulia</juego>
</JuegosResponse>
```

EndPoint para la lista de juegos por tipo de lotería:

Mediante el método **GET**: (Autenticación)

```
https://{{ip:puerto}}/api/v1/tipos-loteria/{tipo-loteria-id}/juegos
```

JSON Respuesta Exitosa (200 OK)

```
{
  "tipoLoteria": "triple",
  "juegos": [
    "trio-activo",
    "chance",
    "terminaltrio"
  ]
}
```

XML Respuesta Exitosa (200 OK)

```
<TipoLoteriaJuegosResponse>
  <tipoLoteria>triple</tipoLoteria>
  <juegos>
    <juego>trio-activo</juego>
    <juego>chance</juego>
    <juego>terminaltrio</juego>
  </juegos>
</TipoLoteriaJuegosResponse>
```

EndPoint para las exportaciones:

Mediante el método **GET**: (Autenticación)

```
https://{{ip:puerto}}/api/v1/exportaciones?entidad=entidad&moneda=moneda&fechaDesde=yyyy-mm-dd&fechaHasta=yyyy-mm-dd&tipoLoteria=tipo&juego=juego
```

Explicación de los campos a enviar para la consulta:

1. entidad= (obligatorio): filtra el tipo de consulta, si es por agencias o por distribuidores (recogedores).
   - ag: Para agencias.
   - rc: Para distribuidores.

2. moneda= (opcional): filtra la moneda a consultar, si no se coloca este parámetro, se retorna por defecto Bolívares.
   - ved: para bolívares digitales.
   - usd: para dólares estadounidenses.

3. fechaDesde = (obligatorio): fecha desde o fecha inicial de la consulta.

4. fechaHasta = (opcional): fecha hasta o fecha final de la consulta, si no se coloca este parámetro, se retorna por defecto los valores desde fechaDesde hasta la fecha anterior al momento de la consulta.

   Nota 1: Esta tiene el límite que se puede consultar de la fecha presente -1. Ejemplo: hoy es 03/07/2025, como la venta del 03 no ha pasado al histórico no va a traer esa venta, va a traer hasta la fecha del 02.

   Nota 2: fechaDesde <= fechaHasta

5. tipoLoteria = (opcional): define el tipo de lotería. Si no se coloca este parámetro, se retorna por defecto el valor de multijuego. (Los valores posibles se pueden obtener del endpoint **GET /api/v1/tipo-loteria**).

6. juego= (opcional): define un juego en específico. Si no se coloca este parámetro, se retorna por defecto todos los valores de los juegos asociados a este tipo de lotería. (Los valores posibles se pueden obtener de **GET /api/v1/juegos o GET /api/v1/tipos-loteria/{tipo-loteria-id}/juegos**).

Ejemplo: Código 200 OK (Consulta específica)

```
curl -X GET
"https://{{ip:puerto}}/api/v1/exportaciones?entidad=rc&moneda=ved&fechaDesde=2025
-07-01&fechaHasta=2025-07-03&tipoLoteria=triple&juego=trio-activo" \
-H "Authorization: Bearer lksUJKdfGJKuJYiw78we3gkjkkjhjkdsJHHJJHJFDDD"
```

JSON Respuesta:


```
{
  "metadata": {
    "entidad": "rc",
    "moneda": "ved",
    "fechaDesde": "2025-07-01",
    "fechaHasta": "2025-07-03",
    "tipoLoteria": "triple",
    "juego": "trio-activo"
  },
  "salidaConsulta": [
       {
        "nombre": "distribuidor-1",
        "venta":12865,
        "premio":4500,
        "porcentajeVenta":1801.1,
        "total":6563.9,
        "tipoLoteria":"triple",
        "juego":"trio-activo"
       },
       {
        "nombre": "distribuidor-2",
        "venta":475,
        "premio":0,
        "porcentajeVenta":66.5,
        "total":408.5,
        "tipoLoteria":"triple",
        "juego":"trio-activo"
       },
       {
        "nombre": "distribuidor-3",
        "venta":1345,
        "premio":1800,
        "porcentajeVenta":188.3,
        "total":-643.3,
        "tipoLoteria":"triple",
        "juego":"trio-activo"
       }
  ]
}
```

XML Respuesta:

```xml
<ExportResponse>
  <metadata>
    <entidad>rc</entidad>
    <moneda>ved</moneda>
    <fechaDesde>2025-07-01</fechaDesde>
    <fechaHasta>2025-07-03</fechaHasta>
    <tipoLoteria>triple</tipoLoteria>
    <juego>trio-activo</juego>
  </metadata>
  <salidaConsulta>
    <item>
      <nombre>distribuidor-1</nombre>
      <venta>12865</venta>
      <premio>4500</premio>
      <porcentajeVenta>1801.1</porcentajeVenta>
      <total>6563.9</total>
      <tipoLoteria>triple</tipoLoteria>
      <juego>trio-activo</juego>
    </item>
    <item>
      <nombre>distribuidor-2</nombre>
      <venta>475</venta>
      <premio>0</premio>
      <porcentajeVenta>66.5</porcentajeVenta>
      <total>408.5</total>
      <tipoLoteria>triple</tipoLoteria>
      <juego>trio-activo</juego>
    </item>
    <item>
      <nombre>distribuidor-3</nombre>
      <venta>1345</venta>
      <premio>1800</premio>
      <porcentajeVenta>188.3</porcentajeVenta>
      <total>-643.3</total>
      <tipoLoteria>triple</tipoLoteria>
      <juego>trio-activo</juego>
    </item>
  </salidaConsulta>
</ExportResponse>
```

Ejemplo: Código 200 OK (Consulta multijuego)

```
curl -X GET
"https://{{ip:puerto}}/api/v1/exportaciones?entidad=ag&moneda=usd&fechaDesde=2025
-07-01&fechaHasta=2025-07-03&tipoLoteria=multijuego&juego=zulia" \
-H "Authorization: Bearer lksUJKdfGJKuJYiw78we3gkjkkjhjkdsJHHJJHJFDDD"
```

JSON Respuesta:


```
{
  "metadata": {
    "entidad": "ag",
    "moneda": "usd",
    "fechaDesde": "2025-07-01",
    "fechaHasta": "2025-07-03",
    "tipoLoteria": "multijuego",
    "juego": null
  },
  "salidaConsulta": [
       {
        "nombre": "agencia-1",
        "venta":12865,
        "premio":4500,
        "porcentajeVenta":1801.1,
        "total":6563.9,
        "tipoLoteria":"triple",
        "juego":"trio-activo"
      },
      {
        "nombre": "agencia-2",
        "venta":475,
        "premio":0,
        "porcentajeVenta":66.5,
        "total":408.5,
        "tipoLoteria":"terminal",
        "juego":"uneloton"
      },
      {
        "nombre": "agencia-3",
        "venta":1345,
        "premio":1800,
        "porcentajeVenta":188.3,
        "total":-643.3,
        "tipoLoteria":"tripleta",
        "juego":"granjita"
      }
  ]
}
```

XML Respuesta:

```xml
<ExportResponse>
  <metadata>
    <entidad>ag</entidad>
    <moneda>usd</moneda>
    <fechaDesde>2025-07-01</fechaDesde>
    <fechaHasta>2025-07-03</fechaHasta>
    <tipoLoteria>multijuego</tipoLoteria>
    <juego xsi:nil="true" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/> </metadata>
  <salidaConsulta>
    <item>
      <nombre>agencia-1</nombre>
      <venta>12865</venta>
      <premio>4500</premio>
      <porcentajeVenta>1801.1</porcentajeVenta>
      <total>6563.9</total>
      <tipoLoteria>triple</tipoLoteria>
      <juego>trio-activo</juego>
    </item>
    <item>
      <nombre>agencia-2</nombre>
      <venta>475</venta>
      <premio>0</premio>
      <porcentajeVenta>66.5</porcentajeVenta>
      <total>408.5</total>
      <tipoLoteria>terminal</tipoLoteria>
      <juego>uneloton</juego>
    </item>
    <item>
      <nombre>agencia-3</nombre>
      <venta>1345</venta>
      <premio>1800</premio>
      <porcentajeVenta>188.3</porcentajeVenta>
      <total>-643.3</total>
      <tipoLoteria>tripleta</tipoLoteria>
      <juego>granjita</juego>
    </item>
  </salidaConsulta>
</ExportResponse>
```

Nota 4: Tenga en cuenta que, en esta consulta, tipoLoteria podría no colocarse, ya que el valor por defecto es multijuego. Además, al momento de colocar el valor de multijuego

(manual o por defecto), el parámetro juego será desechado y se retornará null en el objeto metadata del JSON/XML de la respuesta.

Ejemplo error: Código 401 Unauthorized

JSON

```json
{
  "error": {
    "codigo": "TOKEN-INVALID",
    "mensaje": " Token de autenticación inválido o expirado."
  }
}
```

XML

```xml
<ErrorResponse>
  <codigo>TOKEN-INVALID</codigo>
  <mensaje>Token de autenticación inválido o expirado.</mensaje>
</ErrorResponse>
```

Ejemplo error: Código 400 Bad Request (Ver Nota 2)

JSON

```json
{
  "error": {
    "codigo": "INVALID-DATE-RANGE",
    "mensaje": "La fechaDesde no puede ser mayor que la fechaHasta."
  }
}
```

XML

```xml
<ErrorResponse>
  <codigo>INVALID-DATE-RANGE</codigo>
  <mensaje>La fechaDesde no puede ser mayor que la fechaHasta</mensaje>
</ErrorResponse>
```

Ejemplo error: Código 400 Bad Request (Ver Nota 1)

JSON

```
{
  "error": {
    "codigo": "FUTURE-DATE-QUERY",
    "mensaje": "No se pueden consultar datos de la fecha actual; los datos se actualizan hasta el día anterior."
  }
}
```

XML

```
<ErrorResponse>
  <codigo>FUTURE-DATE-QUERY</codigo>
  <mensaje>No se pueden consultar datos de la fecha actual; los datos se actualizan hasta el día anterior.</mensaje>
</ErrorResponse>
```

Ejemplo error: Código 401 Unauthorized

JSON

```
{
  "error": {
    "codigo": "INVALID-CREDENTIALS",
    "mensaje": "El usuario o la contraseña son incorrectos."
  }
}
```

XML

```
<ErrorResponse>
  <codigo>INVALID-CREDENTIALS</codigo>
  <mensaje>El usuario o la contraseña son incorrectos.</mensaje>
</ErrorResponse>
```

Anexos:

1. Lista de tipos de lotería, son solo referenciales estos valores:

    1. triple
    2. terminal
    3. figura
    4. animalito
    5. tripleta
    6. trio
    7. parley
    8. caballo
    9. terminalito
    10. chance animalito
    11. recargas
    12. juego-electronico
    13. multijuego
    14. bancos

2. Lista de juegos, son solo referenciales estos valores:

    1. activo
    2. bomba
    3. caliente
    4. caracas
    5. carrusel-millonario
    6. cazaloton
    7. chance
    8. deportivas -hasta-3
    9. deportivas-4-o-mas
    10. dupleta
    11. dupletard
    12. guacharito
    13. gatazo
    14. granjita
    15. granjita-plus
    16. guacharito-millonario
    17. guacharo-activo
    18. lotto-chaima
    19. lotto-activo
    20. lotto-gato
    21. lotto-internacional
    22. lotto-rey
    23. megaanimal
    24. mega-animal-40
    25. monje-millonario
    26. pal-caserito
    27. pega-5
    28. polla-express
    29. ricachona
    30. ruca-vzla
    31. ruco
    32. ruleta-activa
    33. selva-plus
    34. tachira
    35. terminalito
    36. terminaltrio
    37. tigre-millonario
    38. trioactivo
    39. tropicalactivo
    40. uneloton
    41. zamorano
    42. zulia

ESQUEMAS XML (XSD): A continuación, se presentan todos los esquemas XML (XSD) que definen la estructura y los tipos de datos para los mensajes XML de la API. Estos esquemas pueden ser utilizados por los implementadores para validar sus peticiones y respuestas.

Nota 6: Asegúrate de reemplazar http://yourcompany.com/schemas/... con el namespace adecuado para tu organización y de que sea consistente en todos los esquemas

a) Esquema para Petición de Autenticación:

```xml
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="http://yourcompany.com/schemas/auth"
           xmlns:tns="http://yourcompany.com/schemas/auth"
           elementFormDefault="qualified">

  <xs:element name="AuthRequest">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="username" type="xs:string"/>
        <xs:element name="password" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

b) Esquema para Respuesta de Autenticación

```xml
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="http://yourcompany.com/schemas/auth"
           xmlns:tns="http://yourcompany.com/schemas/auth"
           elementFormDefault="qualified">

  <xs:element name="AuthResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="tokenType" type="xs:string"/>
        <xs:element name="expiresIn" type="xs:long"/>
        <xs:element name="accessToken" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

c) Esquema para Respuesta de Tipos de Lotería

```xml
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="http://yourcompany.com/schemas/lottery"
           xmlns:tns="http://yourcompany.com/schemas/lottery"
           elementFormDefault="qualified">

  <xs:element name="TiposLoteriaResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="tipoLoteria" type="xs:string" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

d) Esquema para Respuesta de Juegos

```xml
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="http://yourcompany.com/schemas/lottery"
           xmlns:tns="http://yourcompany.com/schemas/lottery"
           elementFormDefault="qualified">

  <xs:element name="JuegosResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="juego" type="xs:string" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

e) Esquema para Respuesta de Juegos por Tipo de Lotería

```xml
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="http://yourcompany.com/schemas/lottery"
           xmlns:tns="http://yourcompany.com/schemas/lottery"
           elementFormDefault="qualified">

  <xs:element name="TipoLoteriaJuegosResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="tipoLoteria" type="xs:string"/>
        <xs:element name="juegos">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="juego" type="xs:string" maxOccurs="unbounded"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

f) Esquema para Respuesta de Error

```xml
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="http://yourcompany.com/schemas/common"
           xmlns:tns="http://yourcompany.com/schemas/common"
           elementFormDefault="qualified">

  <xs:element name="ErrorResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="codigo" type="xs:string"/>
        <xs:element name="mensaje" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

g) Esquema para Respuesta de Exportación

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="http://yourcompany.com/schemas/export"
           xmlns:tns="http://yourcompany.com/schemas/export"
           elementFormDefault="qualified">

  <xs:element name="ExportResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="metadata">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="entidad" type="xs:string"/>
              <xs:element name="moneda" type="xs:string"/>
              <xs:element name="fechaDesde" type="xs:date"/>
              <xs:element name="fechaHasta" type="xs:date"/>
              <xs:element name="tipoLoteria" type="xs:string"/>
              <xs:element name="juego" type="xs:string" minOccurs="0"
nillable="true"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="salidaConsulta">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="item" maxOccurs="unbounded">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="nombre" type="xs:string"/>
                    <xs:element name="venta" type="xs:decimal"/>
                    <xs:element name="premio" type="xs:decimal"/>
                    <xs:element name="porcentajeVenta" type="xs:decimal"/>
                    <xs:element name="total" type="xs:decimal"/>
                    <xs:element name="tipoLoteria" type="xs:string"/>
                    <xs:element name="juego" type="xs:string"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

</xs:schema>
```