

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені ІВАНА ФРАНКА

Факультет прикладної математики та інформатики

Бази даних та інформаційні системи

ЛАБОРАТОРНА РОБОТА №13

Тема: «XML схема XML документа»

Виконав:

Ст. Лук'янчук Денис

Група ПМІ-33

2025

Тема: «XML схема XML документа».

Мета роботи: Ознайомлення з конструкціями XML Schema XML документа. Отримати практичні навички її створення і використання.

Завдання лабораторної роботи:

- Опрацювати теоретичний матеріал щодо **XML Schema (XSD)**.
- Використовуючи «**Альтернативний метод**», створити **XML Schema (XSD)** для XML-документа <shop>, який містить інформацію з трьох зв'язаних таблиць.
- Обов'язково використати **обмеження (restriction)** та **грані (facets)** для валідації даних.
- Використати on-line валідатор для перевірки валідності XML документа разом з XSD Schema.
- Оформити звіт з кодом XML та XSD, а також результатами перевірки.

Хід роботи

Під час виконання цієї лабораторної роботи було виконано завдання зі створення XSD-схеми (XML Schema Definition) для XML-документа, що описує структуру магазину (<shop>). Ця структура базується на трьох таблицях: Customer, Order та Part.

```
<?xml version="1.0" encoding="UTF-8"?>
<shop>
  <customer>
    <id>1</id>
    <name>Customer_1</name>
    <email>user1@example.com</email>
    <orders>
      <orders/>
    </orders>
  </customer>
  <customer>
    <id>2</id>
    <name>Customer_2</name>
    <email>user2@example.com</email>
    <orders>
      <orders/>
    </orders>
  </customer>
  <customer>
    <id>3</id>
    <name>Customer_3</name>
    <email>user3@example.com</email>
    <orders>
      <order>
        <order_id>16</order_id>
        <date>2025-02-17T23:49:18.683291</date>
        <status>New</status>
        <quantity>4</quantity>
        <part>
          <serial>SN000064</serial>
          <part_name>Part_64</part_name>
          <price>306.80</price>
        </part>
      </order>
    </orders>
  </customer>
  <customer>
    <id>4</id>
    <name>Customer_4</name>
    <email>user4@example.com</email>
    <orders>
      <orders/>
    </orders>
  </customer>

```

Схема була створена за «Альтернативним методом», як вимагалось у завданні. Це означає, що всі елементи (прості та складні) були спочатку визначені глобально, а потім скомпоновані в кінцеву структуру за допомогою посилань ref.

Особливу увагу було приділено обмеженням (restriction) та граням (facets):

1. Типи даних: Елементом `<id>`, `<order_id>` та `<quantity>` призначено тип `xsd:positiveInteger` (ціле число > 0).
2. Грань pattern: Для `<email>` використано регулярний вираз для перевірки базової структури email-адреси. Для `<serial>` також використано pattern.
3. Грань enumeration: Для `<status>` визначено чіткий перелік допустимих значень: "Pending", "Shipped" або "Delivered".
4. Грані fractionDigits та minInclusive: Для `<price>` встановлено тип `xsd:decimal` з обмеженням у 2 знаки після коми та мінімальним значенням "0.00".

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
3
4      <xsd:element name="id" type="xsd:positiveInteger"/>
5      <xsd:element name="name" type="xsd:string"/>
6      <xsd:element name="email">
7          <xsd:simpleType>
8              <xsd:restriction base="xsd:string">
9                  <xsd:pattern value="^[@ \t\r\n]+@[@ \t\r\n]+\.[^@ \t\r\n]+"/>
10             </xsd:restriction>
11         </xsd:simpleType>
12     </xsd:element>
13
14     <xsd:element name="order_id" type="xsd:positiveInteger"/>
15     <xsd:element name="date" type="xsd:dateTime"/>
16     <xsd:element name="status">
17         <xsd:simpleType>
18             <xsd:restriction base="xsd:string">
19                 <xsd:enumeration value="New"/>
20                 <xsd:enumeration value="Processing"/>
21                 <xsd:enumeration value="Completed"/>
22                 <xsd:enumeration value="Cancelled"/>
23             </xsd:restriction>
24         </xsd:simpleType>
25     </xsd:element>
26     <xsd:element name="quantity" type="xsd:positiveInteger"/>
27
28     <xsd:element name="serial">
29         <xsd:simpleType>
30             <xsd:restriction base="xsd:string">
31                 <xsd:pattern value="SN[0-9]{6}"/>
32             </xsd:restriction>
33         </xsd:simpleType>
34     </xsd:element>
35     <xsd:element name="part_name" type="xsd:string"/>
36     <xsd:element name="price">
37         <xsd:simpleType>
38             <xsd:restriction base="xsd:decimal">
39                 <xsd:minInclusive value="0.00"/>
40                 <xsd:fractionDigits value="2"/>
41             </xsd:restriction>
42         </xsd:simpleType>
43     </xsd:element>
```

```

44
45 <xsd:element name="part">
46   <xsd:complexType>
47     <xsd:sequence>
48       <xsd:element ref="serial"/>
49       <xsd:element ref="part_name"/>
50       <xsd:element ref="price"/>
51     </xsd:sequence>
52   </xsd:complexType>
53 </xsd:element>
54
55 <xsd:element name="order">
56   <xsd:complexType>
57     <xsd:sequence>
58       <xsd:element ref="order_id"/>
59       <xsd:element ref="date"/>
60       <xsd:element ref="status"/>
61       <xsd:element ref="quantity"/>
62       <xsd:element ref="part"/>
63     </xsd:sequence>
64   </xsd:complexType>
65 </xsd:element>
66
67 <xsd:element name="orders">
68   <xsd:complexType>
69     <xsd:choice>
70       <xsd:sequence minOccurs="0" maxOccurs="unbounded">
71         <xsd:element ref="order"/>
72       </xsd:sequence>
73       <xsd:element name="orders">
74         <xsd:complexType /> </xsd:element>
75       </xsd:choice>
76     </xsd:complexType>
77 </xsd:element>
78
79 <xsd:element name="customer">
80   <xsd:complexType>
81     <xsd:sequence>
82       <xsd:element ref="id"/>
83       <xsd:element ref="name"/>
84       <xsd:element ref="email"/>
85       <xsd:element ref="orders"/> </xsd:sequence>
86     </xsd:complexType>
87 </xsd:element>
88
89 <xsd:element name="shop">
90   <xsd:complexType>
91     <xsd:sequence>
92       <xsd:element ref="customer" minOccurs="0" maxOccurs="unbounded"/>
93     </xsd:sequence>
94   </xsd:complexType>
95 </xsd:element>
96
97 </xsd:schema>

```

Файли shop.xml та shop.xsd були завантажені в on-line XSD валідатор (в freeformatter.com). Валідатор підтвердив, що XML-документ є **коректним (valid)** відповідно до правил, визначених у XML Schema.

XML Validator – XSD (XML Schema)

Validators / XML Validator – XSD (XML Schema)

Validates the XML string/file against the specified XSD string/file. XSD files are "XML Schemas" that describe the structure of a XML document. The validator checks for well formedness first, meaning that your XML file must be parsable using a DOM/SAX parser, and only then does it validate your XML against the XML Schema. The validator will report fatal errors, non-fatal errors and warnings.

The XML document is valid. ✕

Option 1: Copy-paste your XML document here

```
<shop>
<customer>
<id>1</id>
<name>Customer_1</name>
<email>user1@example.com</email>
```

Option 2: Or upload your XML file

File encoding

Огляд...

Файл не вибрано.

UTF-8 ▼

Option 1: Copy-paste your XSD here (Optional if XSD referred in XML using schemaLocation)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <xsd:element name="id" type="xsd:positiveInteger"/>
  <xsd:element name="name" type="xsd:string"/>
```

Option 2: Or upload your XSD document

File encoding

Огляд...

Файл не вибрано.

UTF-8 ▼

Validate XML

Висновок:

Під час виконання цієї лабораторної роботи я ознайомився з конструкціями XML Schema (XSD). Я отримав практичні навички її створення, використовуючи «Альтернативний метод» для визначення глобальних елементів та їх компонування.

Найголовніше, я навчився застосовувати обмеження (restrictions) та грані (facets) для строгої валідації даних. Використання pattern для email, enumeration для статусів замовлень та fractionDigits для цін демонструє, наскільки XSD дозволяє точніше контролювати коректність даних порівняно з DTD.

Відповіді на контрольні питання

1. Які є два способи визначення структури та елементів в XML документі?

Існує два основних способи:

1. DTD (Document Type Definition): Старіший метод, який визначає структуру, порядок елементів, атрибути та типи вмісту (наприклад, #PCDATA).
2. XML Schema (XSD): Сучасніший і набагато потужніший метод. XSD не лише визначає структуру, але й підтримує типи даних (як-от `xsd:integer`, `xsd:date`), простори імен та складні обмеження (facets).

2. Як визначаються прості та складні типи елементів?

- Простий тип (Simple Type): Визначає елемент, який містить лише текстові дані і не може мати дочірніх елементів або атрибутів. Його можна визначити або напямую (наприклад, `<xsd:element name="city" type="xsd:string"/>`), або за допомогою тегу `<xsd:simpleType>`, якщо потрібно додати обмеження (facets).
- Складний тип (Complex Type): Визначає елемент, який містить інші дочірні елементи та/або має атрибути. Він завжди визначається за допомогою тегу `<xsd:complexType>`. У середині нього зазвичай використовується `<xsd:sequence>` (для впорядкованих елементів) або `<xsd:all>` (для неупорядкованих).

3. Які особливості визначення атрибутів?

Атрибути визначаються всередині `<xsd:complexType>` (або `<xsd:extension>`) елемента, до якого вони належать, за допомогою тегу `<xsd:attribute>`.

Ключові особливості:

- Синтаксис: `<xsd:attribute name="назва" type="тип"/>`.
- Обов'язковість: За замовчуванням атрибути є необов'язковими. Щоб зробити атрибут обов'язковим, використовують `use="required"`.
- Значення: Атрибутам можна надати значення за замовчуванням (`default="значення"`) або фіксоване значення, яке не можна змінити (`fixed="значення"`).
- Елементи з текстом і атрибутом: Якщо елемент містить *і текст, і атрибути* (наприклад, `<city country="UA">Kyiv</city>`), його визначення ускладнюється і вимагає використання тегів `<xsd:simpleContent>` та `<xsd:extension>`.

4. Які є обмеження (facets) для елемента XML або атрибута?

Обмеження, або грані (facets), використовуються всередині `<xsd:restriction>` для уточнення та валідації простих типів даних. У наданому матеріалі згадані такі:

- **enumeration:** Визначає точний список допустимих значень (наприклад, "Pending", "Shipped", "Delivered").
- **pattern:** Визначає регулярний вираз, якому має відповідати значення (наприклад, для поштового індексу).
- **length:** Визначає *точну* кількість символів.
- **minLength / maxLength:** Визначають мінімальну та максимальну кількість символів.
- **minInclusive / maxInclusive:** Визначають мінімальне та максимальне допустиме числове значення (включно з самим значенням).
- **minExclusive / maxExclusive:** Визначають мінімальне та максимальне допустиме числове значення (виключаючи саме значення).
- **totalDigits:** Визначає точну загальну кількість цифр у числі.
- **fractionDigits:** Визначає максимальну кількість цифр після десяткової коми.