ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені ІВАНА ФРАНКА
Факультет прикладної математики та інформатики

Бази даних та інформаційні системи

ЛАБОРАТОРНА РОБОТА №5

XML Schema XML документа

Виконав:

Ст. Прізвище Ім’я

Група ……..

2022

Тема: Вивчення поняття XML Schema XML документа.

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

Теоретичний матеріал:

Основними поняттями XML є *коректність (well formed)* та *валідність (valid):*

- **коректний** документ відповідає всім синтаксичним правилам XML. Документ, що не є коректним, не може називатись XML-документом.
- документ називається **валідним** *(valid)* або ***дійсним***, якщо він є коректним і містить посилання на граматичні правила та повністю відповідає обмеженням, вказаним у цих правилах описаних у DTD схемі або XML Schema.

Є кілька способів визначення елементів, які використовуються для представлення даних в XML документі - DTD схема, XML Schema, RELAX NG, Schematron.

Першим з розроблених методів є використання визначення типу документа **Document Type Definition** (DTD). DTD визначає елементи, які можуть з'являтися в XML документі, порядок їх відображення, те, як вони можуть бути вкладені один у одного, та інші основні деталі структури документа XML. Кожен елемент та атрибут, використовувані в документі, визначеного DTD, повинні бути описані. Кожен елемент повинен мати змістовну модель, яка визначає, які дочірні елементи чи текстові вузли дозволені, а також список допустимих атрибутів, якщо дозволені будь-які атрибути. DTD є частиною оригінальної специфікації XML і дуже схожі на SGML DTD.

Найбільш поширений, на сьогодні, метод полягає у використанні **XML Schema або XML Schema Definition (XSD)**. Схема може визначати всі структури документів, які можна помістити в DTD, а також може визначити **типи** даних і **більш складні** правила, ніж може DTD. Типи даних є контейнерами для різних видів вмісту, від тексту до цілих чисел і до дат. W3C розробив специфікацію схеми XML через кілька років після оригінальної специфікації XML.

Мова XML Schema визначена W3C у трьох частинах:

- Підручник , розташований за адресою https://www.w3.org/TR/xmlschema-0/, який дає ознайомлення з документами XML-схеми і орієнтований на швидке розуміння способів створення схем;
- Стандарт для структур документів, розміщений за адресою https://www.w3.org/TR/2004/REC-xmlschema-1-20041028/structures.html, який ілюструє, як визначити структуру XML-документів;
- Стандарт для типів даних , розташований за адресою https://www.w3.org/TR/2004/REC-xmlschema-2-20041028/datatypes.html, який визначає загальні типи даних та правила для створення нових типів.

За допомогою XML-схем є більше можливостей визначити, як виглядають дійсні документи XML. Вони мають ряд переваг перед DTD:

- **У схемах XML використовується синтаксис XML.** Іншими словами, схема XML - це документ XML. Це означає, що можна обробити схему, як і будь-який інший

документ. Наприклад, можна використати XSLT, який перетворює схему XML у веб-форму в комплекті з автоматично сформованим кодом JavaScript, який перевіряє дані під час введення. І, звичайно, підтримка просторів імен.

- **XML-схеми підтримують типи даних.** Незважаючи на те, що DTD підтримують типи даних, але вони були розроблені з погляду публікації. XML-схеми підтримують крім типів даних з DTD (такі як ідентифікатори та посилання на ідентифікатори), також цілі числа, числа з плаваючою комою, дати, час, рядки, URL-адреси та інші типи даних, корисні для обробки та перевірки даних. Це дозволяє
  - простіше описати зміст документа
  - простіше визначити обмеження щодо даних
  - простіше перевірити правильність даних
  - легше конвертувати дані між різними типами даних
- **XML-схеми розширюються.** Окрім типів даних, визначених у специфікації схеми XML, також можна створити свої власні типи та отримати нові типи даних на основі інших типів даних, використовувати повторно свою схему в інших схемах і посилатися на кілька схем в одному документі.
- **XML-схеми підтримують встановлення обмежень даних.** Наприклад, за допомогою XML-схем можна визначити, що значення будь-якого <*state*> атрибута не може бути довше двох символів або що значення будь-якого <*postal-code*> елемента повинно відповідати звичайному виразу *[0-9]{5}(-[0-9]{4})?*. З DTD не можна робити жодної з цих речей.
- **Присутня підтримка DOM (Document Object Model).**

Приклад XML-схеми

Щоб побудувати схему можна просто прослідкувати структуру XML документу та визначити кожний елемент, що зустрівся в ній. Цей метод є дуже простим, але є важливим для розуміння XML Schema.

Ось XML схема, яка відповідає DTD схемі попередньої лабораторної роботи. Але додає два обмеження: значення <state> елемента повинно бути рівно два символа, а значення <postal-code> елемента повинно відповідати регулярному виразу [0-9]{5}(-[0-9]{4})?.

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
3     <xsd:element name="address">
4       <xsd:complexType>
5         <xsd:sequence>
6           <xsd:element name="name">
7             <xsd:complexType>
8               <xsd:sequence>
9                 <xsd:element name="title" type="xsd:string" />
10                <xsd:element name="first-name" type="xsd:string" />
11                <xsd:element name="last-name" type="xsd:string" />
12              </xsd:sequence>
13            </xsd:complexType>
14          </xsd:element>
15          <xsd:element name="street" type="xsd:string" />
16          <xsd:element name="city" type="xsd:string" />
17          <xsd:element name="state">
18            <xsd:simpleType>
```

```
19              <xsd:restriction base="xsd:string">
20                <xsd:length value="2"/>
21              </xsd:restriction>
22            </xsd:simpleType>
23          </xsd:element>
24          <xsd:element name="postal-code">
25            <xsd:simpleType>
26              <xsd:restriction base="xsd:string">
27                <xsd:pattern value="[0-9]{5}(-[0-9]{4})?"/>
28              </xsd:restriction>
29            </xsd:simpleType>
30          </xsd:element>
31        </xsd:sequence>
32      </xsd:complexType>
33    </xsd:element>
34  </xsd:schema>
```

Проте, побудована за допомогою цього методу схема складного документу може бути складною для читання та підтримки.

Альтернативний метод побудови, який застосовано в наведеному нижче прикладі, заснований на початковому визначенні всіх простих елементів та атрибутів і наступному посиланні на них за допомогою атрибута ref.

Хоча схема набагато довша, ніж DTD, вона краще визначає, як виглядає дійсний документ:

```
1       <?xml version="1.0" encoding="UTF-8"?>
2       <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
3
4         <xsd:element name="title"           type="xsd:string"/>
5         <xsd:element name="first-name" type="xsd:string"/>
6         <xsd:element name="last-name"  type="xsd:string"/>
7         <xsd:element name="street"          type="xsd:string"/>
8         <xsd:element name="city"            type="xsd:string"/>
9
10        <xsd:element name="address">
11          <xsd:complexType>
12            <xsd:sequence>
13              <xsd:element ref="name"/>
14              <xsd:element ref="street"/>
15              <xsd:element ref="city"/>
16              <xsd:element ref="state"/>
17              <xsd:element ref="postal-code"/>
18            </xsd:sequence>
19          </xsd:complexType>
20        </xsd:element>
21
22        <xsd:element name="name">
23          <xsd:complexType>
24            <xsd:sequence>
25              <xsd:element ref="title" minOccurs="0"/>
26              <xsd:element ref="first-name"/>
27              <xsd:element ref="last-name"/>
28            </xsd:sequence>
29          </xsd:complexType>
30        </xsd:element>
31
```

```
32      <xsd:element name="state">
33       <xsd:simpleType>
34         <xsd:restriction base="xsd:string">
35           <xsd:length value="2"/>
36         </xsd:restriction>
37       </xsd:simpleType>
38     </xsd:element>
39
40     <xsd:element name="postal-code">
41       <xsd:simpleType>
42         <xsd:restriction base="xsd:string">
43           <xsd:pattern value="[0-9]{5}(-[0-9]{4})?"/>
44         </xsd:restriction>
45       </xsd:simpleType>
46     </xsd:element>
47   </xsd:schema>
```

Визначення елементів у схемах

Схема XML визначає кілька елементів XML з <xsd:element> елементом. Перші два елементи, <address> та <name>, складаються з інших елементів. Елемент <xsd:sequence> визначає послідовність елементів, які містяться в кожному з них:

```
1     <xsd:element name="address">
2       <xsd:complexType>
3         <xsd:sequence>
4           <xsd:element ref="name"/>
5           <xsd:element ref="street"/>
6           <xsd:element ref="city"/>
7           <xsd:element ref="state"/>
8           <xsd:element ref="postal-code"/>
9         </xsd:sequence>
10      </xsd:complexType>
11    </xsd:element>
```

Як і у версії DTD, приклад схеми XML визначає, що <address> містить <name>, <street>, <city>, <state> і <postal-code> елемент у такому ж порядку. Але схема фактично визначає новий тип даних з <xsd:complexType> елементом. Це *складний* тип, оскільки містить інші елементи.

Більшість елементів містять текст. Ви просто оголошуєте новий елемент і надаєте йому тип даних xsd:string. Це *простий* тип. Він не може містити інших елементів або атрибутів:

```
1     <xsd:element name="title"      type="xsd:string"/>
2     <xsd:element name="first-Name" type="xsd:string"/>
3     <xsd:element name="last-Name"  type="xsd:string"/>
4     <xsd:element name="street"     type="xsd:string"/>
5     <xsd:element name="city"       type="xsd:string"/>
```

Найпоширеніші вбудовані типи елементів є string, decimal, integer, boolean, date, time.

Визначення вмісту елементів у схемах

Приклад схеми визначає обмеження для вмісту двох елементів: Вміст <state> елемента повинен бути довжиною два символи, а вміст <postal-code> елемента повинен відповідати регулярному виразу [0-9]{5}(-[0-9]{4})?:

```
1       <xsd:element name="state">
2         <xsd:simpleType>
3           <xsd:restriction base="xsd:string">
4             <xsd:length value="2"/>
5           </xsd:restriction>
6         </xsd:simpleType>
7       </xsd:element>
8
9       <xsd:element name="postal-code">
10        <xsd:simpleType>
11          <xsd:restriction base="xsd:string">
12            <xsd:pattern value="[0-9]{5}(-[0-9]{4})?"/>
13          </xsd:restriction>
14        </xsd:simpleType>
15      </xsd:element>
```

Для елементів <state> та <postal-code> схема визначає нові типи даних з обмеженнями. У першому випадку використовується <xsd:length> елемент, а в другому використовується <xsd:pattern> елемент для визначення регулярного виразу, якому цей елемент повинен відповідати.

Визначення атрибута

Синтаксис для визначення атрибута: **<xsd:attribute name="xxx" type="yyy"/>**
де xxx - це ім'я атрибута, а yyy вказує тип даних атрибута.
XML-схема має багато вбудованих типів даних атрибута. Найпоширеніші типи - string, decimal, integer, boolean, date, time.

Для елемента XML з атрибутом: <city country="US"/>
відповідне визначення атрибута в схемі:

```
<xsd:element name="city"  >
   <xsd:complexType>
       <xsd:attribute name="country" type="xsd:string" />
   </xsd:complexType>
</xsd:element>
```

Але для елемента XML з атрибутом: <city country="US">**Anytown**</city>
відповідне визначення атрибута в схемі:

```
<xsd:element name="city"  >
   <xsd:complexType>
       <xsd:simpleContent>
           <xsd:extension base="xsd:string">
              <xsd:attribute name="country" type="xsd:string" />
           </xsd:extension>
       </xsd:simpleContent>
   </xsd:complexType>
</xsd:element>
```

Значення за замовчуванням і фіксовані значення для атрибутів

Атрибути можуть мати значення за замовчуванням або вказане фіксоване значення.
Значення за замовчуванням автоматично призначається атрибуту, коли інше значення не вказано.
У наступному прикладі значенням за замовчуванням є "EN":

```
<xsd:attribute name="lang" type="xsd:string" default="EN"/>
```
Фіксоване значення також автоматично призначається атрибуту, і ви не можете вказати інше значення.
У наступному прикладі фіксованим значенням є "UA":
```
<xsd:attribute name="lang" type="xsd:string" fixed="UA"/>
```

Необов'язкові та необхідні атрибути
Атрибути за замовчуванням необов'язкові. Щоб вказати, що атрибут необхідний, використовується атрибут "required":
```
<xsd:attribute name="lang" type="xsd:string" use="required"/>
```

Обмеження щодо вмісту
Якщо для елемента XML або атрибута визначений тип даних, він встановлює обмеження на вміст елемента або атрибута.
Якщо елемент XML має тип "xsd:date" і містить рядок типу "Hello World", елемент вважатиметься не валідним (тобто виникне помилка при валідації).
За допомогою XML-схем також можна додавати власні обмеження до своїх елементів та атрибутів XML. Ці обмеження ще називаються гранями або границями (facets) .

Обмеження для Datatypes

enumeration	Визначає список допустимих значень
fractionDigits	Задає максимальну кількість знаків після коми. Має дорівнювати або більше нуля
length	Визначає точне число символів або елементів списку. Має дорівнювати або більше нуля
maxExclusive	Визначає верхню межу для числових значень (значення повинно бути менше цього значення)
maxInclusive	Визначає верхню межу для числових значень (значення повинно бути менше або дорівнює цьому значенню)
maxLength	Задає максимальне число символів або елементів списку. Має дорівнювати або більше нуля
minExclusive	Задає нижні межі для числових значень (значення повинно бути більше, ніж це значення)
minInclusive	Задає нижні межі для числових значень (значення повинно бути більше або дорівнює цьому значенню)
minLength	Задає мінімальну кількість символів або елементів списку. Має дорівнювати або більше нуля
pattern	Визначає точну послідовність символів, які є прийнятними
totalDigits	Визначає точне число цифр. Повинно бути більше нуля
whiteSpace	Визначає, як обробляються пропуски (наступний рядок, табуляція, пропуски і повернення на початок рядка)

Перед тим, як визначити структуру документа XML, слід заздалегідь продумати схему, як і коли б ви розробляли схему бази даних або структуру даних вашої програми. Чим більше майбутніх вимог можна передбачити, тим легше і дешевше буде їх виконати пізніше.
На даний час в PostgreSQL немає вбудованої підтримки XML Schema.

Посилання для ознайомлення з теоретичним матеріалом теми «XML Schema XML документа»:

1. Beginning XML, 5-edition © 2012 Joe Fawcett, Liam R.E. Quin, Danny Ayers
2. W3C Recommendation XML Schema Part 0: Primer Second Edition
3. XML Tutorial W3C

Перелік розділів та понять, з якими необхідно ознайомитись для виконання завдання лабораторної роботи:

1. Основні поняття XML (*коректність* та *валідність)*
2. Два способи визначення структури та елементів в XML документі.
3. Визначення структури та елементів у XML Schema.
4. Прості та складні типи елементів.
5. Визначення атрибутів.
6. Обмеження (грані) XML Schema.

Хід роботи

1. Опрацювати теоретичний матеріал.
2. Використовуючи Альтернативний метод створити XML Schema XML документа, який містить інформацію та структуру двох-трьох зв'язаних таблиць реляційної бази даних. Обов'язково використати обмеження (restriction або extension) та границі (facets)
3. Використати on-line валідатор для перевірки валідності створеного XML документа разом з XML Schema.
4. Оформити звіт про виконання лабораторної роботи, який має містити:
    - титульну сторінку;
    - тему, мету та завдання лабораторної роботи;
    - короткий перелік та опис створення XML Schema XML документа;
    - навести скріни екрану з кодом XML Schema та XML документа та результатом перевірки його валідатором.
5. Завантажити в канал "БД. Лабораторна робота" своєї команди в Teams.