

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені ІВАНА ФРАНКА

Факультет прикладної математики та інформатики

Бази даних та інформаційні системи

ЛАБОРАТОРНА РОБОТА №11

Тема: «Мова XML та її використання в СКБД PostgreSQL»

Виконав:

Ст. Лук'янчук Денис

Група ПМІ-33

2025

Тема: «Мова XML та її використання в СКБД PostgreSQL».

Мета роботи: Ознайомлення з конструкціями мови XML та її використанням в СКБД PostgreSQL, зокрема, зі створенням XML даних та XML документів, перетворенням таблиць реляційної бази в XML документ.

Завдання лабораторної роботи:

- Опрацювати теоретичний матеріал щодо структури XML та його використання в PostgreSQL.
- Створити XML-документ із інформацією з 3–4 зв'язаних таблиць реляційної бази даних `spare_parts_shop` без використання функцій `table_to_xml` чи `database_to_xml`, використовуючи `xmlelement`, `xmlforest`, `xmlagg` тощо.
- Використати онлайн-інструменти для форматування та перевірки синтаксису створеного XML-документа.
- Оформити звіт із зазначеними розділами.

Теоретичний матеріал

XML (Розширювана мова розмітки) — це мова, розроблена W3C, яка дозволяє створювати структуровані документи для зберігання та транспортування даних. На відміну від HTML, де теги визначають відображення, XML фокусується на самоописовості даних завдяки користувацьким тегам (наприклад, `<name>` або `<postal-code>`). Вона базується на SGML, але адаптована для інтернету.

Основні елементи XML:

- Теги: Початкові (`<tag>`) та кінцеві (`</tag>`), наприклад, `<name>Mary</name>`.
- Елементи: Вміст між тегами, включаючи вкладені елементи (наприклад, `<name><first-name>Mary</first-name></name>`).
- Атрибути: Пари ім'я="значення" у тегах (наприклад, `<city state="NC">Anytown</city>`).

Вимоги до документа:

- Має бути правильно сформований (well-formed): один кореневий елемент, коректне вкладення тегів, використання UTF-8/UTF-16.
- Коректний (valid) документ відповідає DTD або XML Schema (у цій роботі DTD не створюється).

Використання в PostgreSQL:

PostgreSQL підтримує XML через функції, такі як:

- `xmlelement`: Створює XML-елемент.

- xmlforest: Групує кілька значень у XML-елементи.
- xmlagg: Агрегує набір XML-елементів.
- xmlroot: Додає кореневий елемент.

Хід роботи

XML-документ створено на основі бази даних spare_parts_shop, яка містить таблиці Customer, Part і Order. Використано SQL-запит із функціями XML для генерації структури без автоматичних функцій table_to_xml. Процес включає:

- Вибір даних із таблиць.
- Формування XML-елементів за допомогою xmlelement і xmlforest.
- Агрегацію даних із кількох рядків за допомогою xmlagg у підзапиті.
- Перевірку синтаксису онлайн.

Код створення XML

Використано наступний виправлений SQL-запит:

| Query | Query History |
|-------|------------------------------------|
| 1 | SELECT xmlelement(name "shop", |
| 2 | xmlagg(customer_element) |
| 3 |) |
| 4 | FROM (|
| 5 | SELECT xmlelement(name "customer", |
| 6 | xmlforest(|
| 7 | c.Customer_ID AS "id", |
| 8 | c.Name AS "name", |
| 9 | c.Email AS "email" |
| 10 |), |
| 11 | xmlelement(name "orders", |
| 12 | COALESCE((|
| 13 | SELECT xmlagg(|
| 14 | xmlelement(name "order", |
| 15 | xmlforest(|
| 16 | o.Order_ID AS "order_id", |
| 17 | o.Order_Date AS "date", |
| 18 | o.Status AS "status", |
| 19 | o.Quantity AS "quantity" |
| 20 |), |
| 21 | xmlelement(name "part", |
| 22 | xmlforest(|
| 23 | p.Serial_Number AS "serial", |
| 24 | p.Name AS "part_name", |
| 25 | p.Price AS "price" |
| 26 |) |
| 27 |) |
| 28 |) |
| 29 |) |

```
30         FROM "Order" o
31         JOIN Part p ON o.Part_ID = p.Part_ID
32         WHERE o.Customer_ID = c.Customer_ID
33     ), xmlparse(content '<orders/>'))
34 )
35 ) AS customer_element
36 FROM Customer c
37 WHERE c.Customer_ID = 85
38 ) subquery;
```

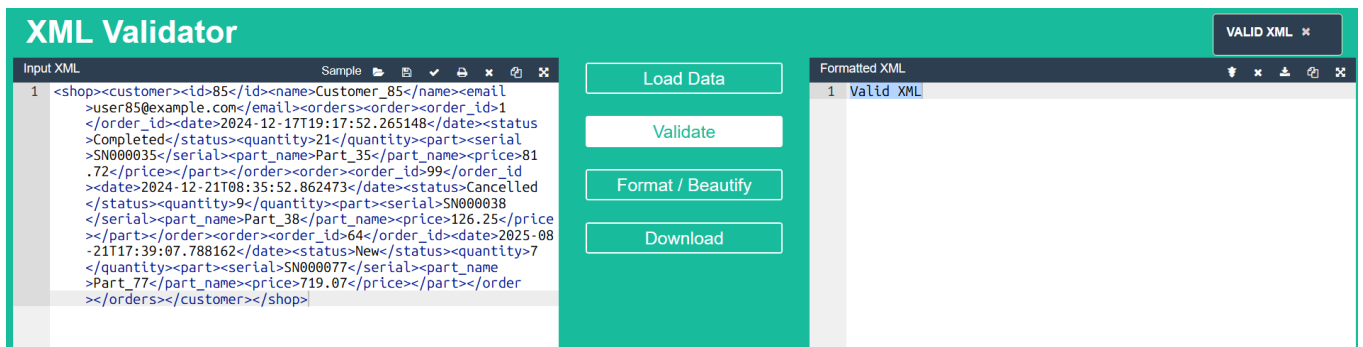
Data Output Messages Notifications

| | xmlelement xml |
|---|---|
| 1 | <shop><customer><id>85</id><name>Customer_85</name><email>user85@example.cc |

Результати

- Запит генерує XML-документ із кореневим елементом <shop>, що містить інформацію про клієнта, його замовлення та відповідні запчастини.
- Документ перевірено онлайн-інструментами: [XML Formatter](#) для форматування та [XML Validator](#) для синтаксичної перевірки.

```
Formatted XML
1- <shop>
2-   <customer>
3-     <id>85</id>
4-     <name>Customer_85</name>
5-     <email>user85@example.com</email>
6-     <orders>
7-       <order>
8-         <order_id>1</order_id>
9-         <date>2024-12-17T19:17:52.265148</date>
10-        <status>Completed</status>
11-        <quantity>21</quantity>
12-        <part>
13-          <serial>SN000035</serial>
14-          <part_name>Part_35</part_name>
15-          <price>81.72</price>
16-        </part>
17-      </order>
18-      <order>
19-        <order_id>99</order_id>
20-        <date>2024-12-21T08:35:52.862473</date>
21-        <status>Cancelled</status>
22-        <quantity>9</quantity>
23-        <part>
24-          <serial>SN000038</serial>
25-          <part_name>Part_38</part_name>
26-          <price>126.25</price>
27-        </part>
28-      </order>
29-      <order>
30-        <order_id>64</order_id>
31-        <date>2025-08-21T17:39:07.788162</date>
32-        <status>New</status>
33-        <quantity>7</quantity>
34-        <part>
35-          <serial>SN000077</serial>
36-          <part_name>Part_77</part_name>
37-          <price>719.07</price>
38-        </part>
39-      </order>
40-    </orders>
41-  </customer>
42- </shop>
```



Висновок:

Під час виконання лабораторної роботи було детально опрацьовано теоретичний матеріал, що стосується структури XML, його основних елементів (тегів, атрибутів, елементів) та вимог до правильно сформованих документів. Особливу увагу приділено можливостям використання XML у PostgreSQL через функції, такі як `xmlelement`, `xmlforest` і `xmlagg`. У практичній частині створено XML-документ на основі реляційних таблиць бази даних `spare_parts_shop` (Customer, Part, Order), що відображає інформацію про клієнта, його замовлення та відповідні запчастини, без використання автоматичних функцій `table_to_xml` чи `database_to_xml`. Процес завершився успішною перевіркою синтаксису за допомогою онлайн-інструментів: XML Formatter для форматування та XML Validator, де документ отримав статус "Valid XML", що підтверджує його коректність і відповідність стандартам. Робота продемонструвала практичну цінність XML для зберігання, транспортування даних та інтелектуального пошуку, а також поглибила навички інтеграції XML із реляційними базами даних у PostgreSQL, що є важливим для сучасних інформаційних технологій.

Відповіді на контрольні питання

1. Що означає аббревіатура XML?

XML – Extensible Markup Language, тобто «розширювана мова розмітки». Вона призначена для зберігання та передачі структурованих даних у текстовому форматі, який одночасно зрозумілий людині та машині.

2. Які три терміни використовуються для опису вмісту документа XML?

Три основні терміни:

1. Element (елемент) – базова структурна одиниця XML, може містити текст або інші елементи.
2. Attribute (атрибут) – додаткова інформація про елемент у вигляді пари «ім'я=значення».
3. Content (вміст) – дані всередині елемента, включаючи текст, інші елементи або порожній вміст.

3. Що означає поняття «правильно сформований» (well formed) документ?

Документ XML вважається правильно сформованим, якщо він відповідає базовим

синтаксичним правилам XML:

- Є один кореневий елемент.
- Всі теги відкриті та закриті у правильному порядку.
- Імена елементів та атрибутів допустимі.
- Атрибути правильно укладені у лапки.
- Структура XML дозволяє його парсинг стандартними XML-парсерами.

4. Що означає поняття «коректний» (valid) документ?

Документ XML вважається коректним, якщо він правильно сформований та відповідає схемі або DTD (Document Type Definition), яка визначає дозволені елементи, їх порядок, атрибути та типи даних.

- Тобто коректність перевіряє структурну відповідність правилам.

5. Для чого в PostgreSQL при створенні XML-контенту використовується функція `xmllagg`?

`xmllagg()` – агрегаційна функція, яка об'єднує кілька XML-рядків або елементів у один XML-документ.

- Наприклад, якщо потрібно зібрати всі рядки таблиці у вигляді списку `<row>...</row>` у єдиний XML-контейнер.

6. Для чого в PostgreSQL при створенні XML-контенту використовується функція `xmlforest`?

`xmlforest()` перетворює колонки таблиці у набір XML-елементів.

- Кожна колонка стає елементом з назвою колонки, а її значення – текстовим вмістом елемента.
- Це зручно для формування XML-структури «рядок → елементи» без ручного написання тегів.

7. Що таке простір імен XML і навіщо в його оголошенні вказується URI?

Простір імен (namespace) XML – це механізм для уникнення конфліктів імен елементів та атрибутів у документі, коли використовуються однакові назви, але різний зміст.

- URI (Uniform Resource Identifier) у оголошенні namespace не обов'язково веде до реального ресурсу, але служить унікальним ідентифікатором простору імен.
- Це дозволяє розрізняти елементи з однаковими іменами, які належать до різних схем.