Paweł DRZYMAŁA, Henryk WELFLE

Politechnika Łódzka, Instytut Mechatroniki i Systemów Informatycznych

doi:10.15199/48.2017.01.33

Zaawansowane mechanizmy wymiany danych w środowisku SQL Server z użyciem standardu XML

Streszczenie. *W pracy zaprezentowano możliwości przechowywania, przetwarzania i wymiany danych XML w środowisku SQL Server 2016, które wykorzystywane są w aplikacjach biznesowych. Coraz więcej danych XML przechowywanych jest i przetwarzanych w środowiskach bazodanowych. Z danymi XML związane są rożne technologie, do których zaliczyć można: XQuery z podzbiorem XPath oraz standard służący do definiowania struktury dokumentu XML Schema. Od wersji SQL Server 2005 zaimplementowano natywny typ danych o nazwie XML, który wykorzystuje hierarchiczną strukturę drzewa, możliwość definiowania indeksów czy zaawansowane mechanizmy przeszukiwania struktur XML. Wspiera on również standard XQuery (zapytania typu FLWOR). Metody te są zdecydowanie wydajniejsze niż przetwarzanie danych tekstowych czy binarnych.*

Abstract*. The paper presents the possibility of storing, processing and exchanging XML data in an SQL Server 2016 that are used in business applications. More and more XML data is stored and processed in database environments. With XML are related to various technologies, which include: XQuery, XPath and standard for storing XML Schema. From SQL Server 2005 implements a native data type called XML, which uses a hierarchical tree structure, the ability to define indexes or advanced search mechanisms XML structures. It also supports standard XQuery (query type FLWOR). These methods are much more efficient than processing text data or binary. (**Advanced mechanisms to exchange data in a SQL Server environment with the use standard XML**).*

Słowa kluczowe: technologie XML, bazy relacyjno-hierarchiczne, SQL Server.
Keywords: XML technology, relational-hierarchical structures, SQL Server.

Wprowadzenie

Coraz więcej danych XML przechowywanych jest i przetwarzanych w środowiskach bazodanowych. Wykonywanie zapytań do dokumentów XML jest bardzo często realizowaną operacją na bazach danych w środowiskach produkcyjnych. Z danymi XML związane są rożne technologie, do których zaliczyć można: XQuery, XPath oraz standard XML Schema. XML to najbardziej uniwersalny język znaczników przeznaczony do opisu danych pozwalający definiować dowolne dokumenty hierarchiczne. Ogólnie mówiąc mamy możliwość wykorzystania tego standardu do wymiany danych pomiędzy różnymi systemami. Stanowi on również metodę przechowywania danych konfiguracyjnych różnych urządzeń i systemów. Środowisko SQL Server 2016 posiada możliwości przechowywania, przetwarzania i wymiany danych XML, które wykorzystywane są w aplikacjach biznesowych. Od wersji SQL Server 2005 zaimplementowano natywny typ danych o nazwie XML, który wykorzystuje hierarchiczną strukturę drzewa. Pozwala również na możliwość definiowania indeksów oraz dostarcza zaawansowane mechanizmy przeszukiwania struktur XML. Wspiera on również standard XQuery w tym zapytania typu FLWOR. Metody te są zdecydowanie wydajniejsze niż przetwarzanie danych tekstowych czy binarnych.

Typ danych XML w SQL Server

Typ danych XML w systemie SQL Server, możemy stosować:

- podczas tworzenia tabel, które zawierają kolumny służące do składowania danych lub dokumentów XML w bazie relacyjnej (Rys.1), a także do definicji widoków opartych na kolumnach typu XML,

```
CREATE TABLE T1_XML
(IntCol INT PRIMARY KEY IDENTITY (1,1),
XmlCol XML);

CREATE VIEW T1_View AS
SELECT IntCol,
XmlCol.value('(/Test/@Name)[1]',
'varchar(40)') AS TName
FROM T1_XML
```

- do przetwarzania danych w procedurach przechowywanych, funkcjach użytkownika, blokach anonimowych czy skryptach w definicji zmiennych,

```
DECLARE @xmldoc XML
```

- w parametrach funkcji lub procedur składowanych służących jako dane na, których realizowane jest przetwarzanie danych typu XML,

```
CREATE PROCEDURE TEST(@xmldoc XML)
```

Przetwarzanie i przechowywanie danych XML

Przetwarzanie danych przechowywanych w bazie danych jest zbliżone do wymiany danych w usługach typu Web Services. Z aplikacji przychodzi żądanie z parametrami w postaci XML, na które baza odpowiada w postaci typu XML z odpowiednią strukturą danych.

SQL Server zapewnia wsparcie dla danych XML przy następujących elementach:

- typ danych XML nie mogą przekraczać 2GB danych. Wartości XML mogą być przechowywane natywnie w kolumnie typu XML, które muszą spełniać reguły XML Schema lub muszą być typu „well-formed" jako dokumentu poprawnie sformułowanego zgodnego ze standardem XML. Można także tworzyć indeks dla elementów kolumny XML w celu poprawy wydajności przetwarzania danych.
- możliwość wykonania zapytania XQuery z danymi XML przechowywanych w kolumnach i zmiennych typu XML (Rys.2)

```
SELECT XmlCol.query (‘
for $i In /bookstore/book
let $j := $i/title/text()
where $i/title/@lang=”eng”
return $j ‘)
FROM T1_XML
```

- funkcja OPENROWSET do zasilania tabel w dane XML (Rys.1)

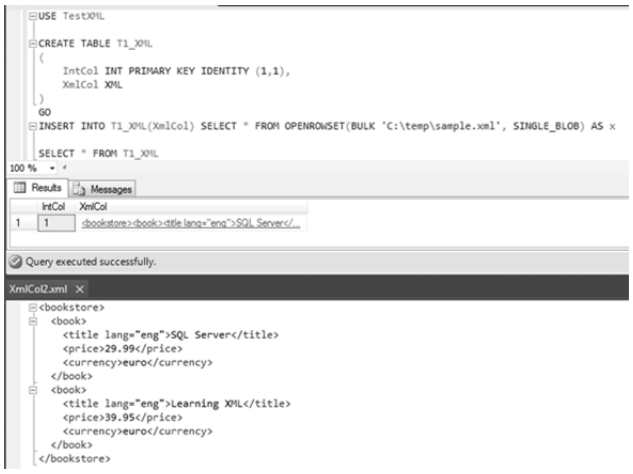
```
INSERT INTO T1_XML(XmlCol)
SELECT * FROM
OPENROWSET(BULK ‘c:\sample.xml’,
SINGLE_BLOB) AS T1;
```

- klauzula FOR XML do pobierania danych relacyjnych i przedstawiania w formacie XML. Istnieje możliwość tworzenia dokumentów XML bezpośrednio z postaci relacyjnej. Klauzula ta rozszerza składnię języka SQL (Rys.3)

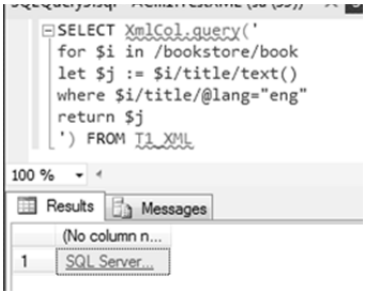
```
SELECT categoryname AS CategoryName
from northwind.dbo.Categories
FOR XML PATH
```

- funkcja OPENXML, do pobierania danych w formacie XML i zwracającej dane relacyjne (Rys. 4). Wykorzystuje się tu wbudowane metody typu XML w których wykonywana jest operacja dzielenia danych XML na dane relacyjne. Efektem końcowym może być przedstawienie danych w postaci relacyjnej lub wstawienie danych do tabeli relacyjnej. W tym celu używane są dwie systemowe procedury przechowywane: sp_xml_preparedocument, która przygotowuje dane wejściowe XML do wykorzystania przez funkcję OPENXML oraz procedura sp_xml_removedocument do czyszczenia pamięci serwera po wykonaniu odpowiednich operacji.

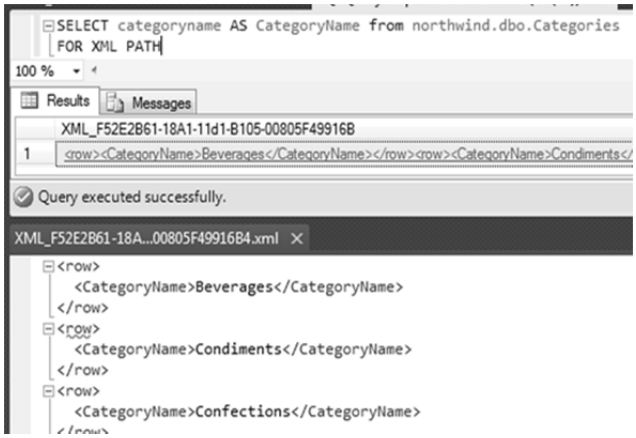
Na rys.2 przedstawiona została jedna z metod typu XML query() zwracając część dokumentu XML. Dostępne są także inne metody służące do przeszukiwania i modyfikacji danych XML. Oprócz metody query() mamy metodę do zwracania pojedynczych wartości skalarnych typów danych o nazwie value() (Rys. 7), metodę do przeszukiwania dokumentów wg. odpowiednich kryteriów – metoda exists(), czy metoda służąca do aktualizacji danych XML modify(). Należy pamiętać iż metody te obsługują przestrzenie nazw występujące w danych XML. Istnieje także możliwość wykorzystania funkcji OpenXML i wtedy także możemy wykorzystywać przestrzenie nazw.



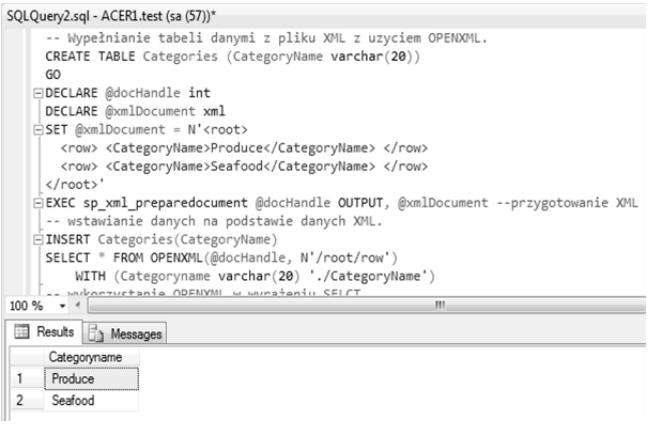
Rys.1. Skrypt tworzący tabelę z kolumna XML, zasilenie tabeli danymi oraz przedstawienie danych w postaci dokumentu XML.



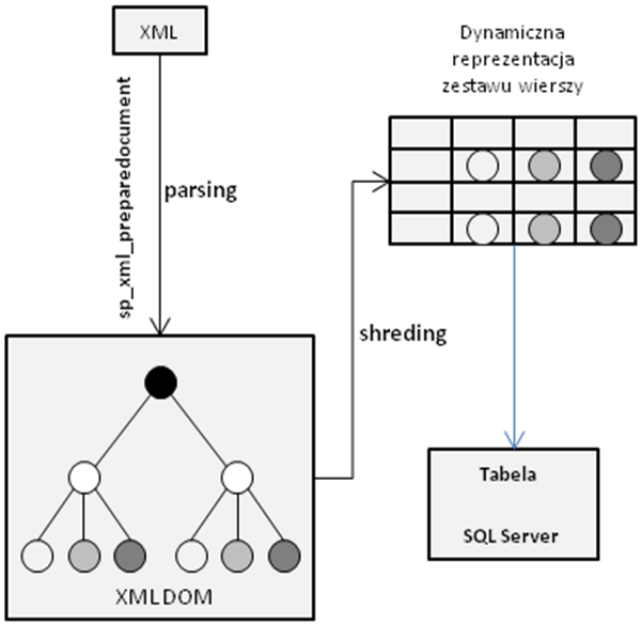
Rys.2. Skrypt wykonujący zapytanie XQuery na danych XML (FLWOR)



Rys.3. Zapytanie z klauzulą FOR XML do generacji danych XML z danych relacyjnych

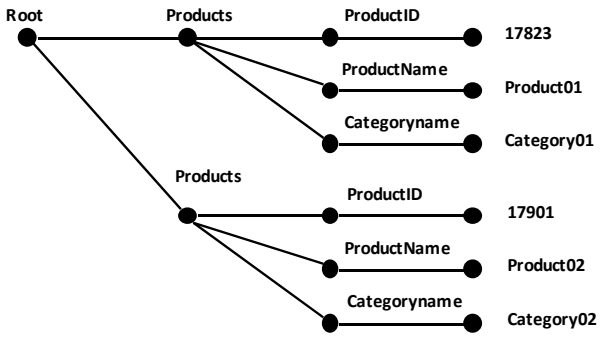


Rys.4. Wykorzystanie funkcji OPENXML do wstawiania i pobierania danych w formacie XML

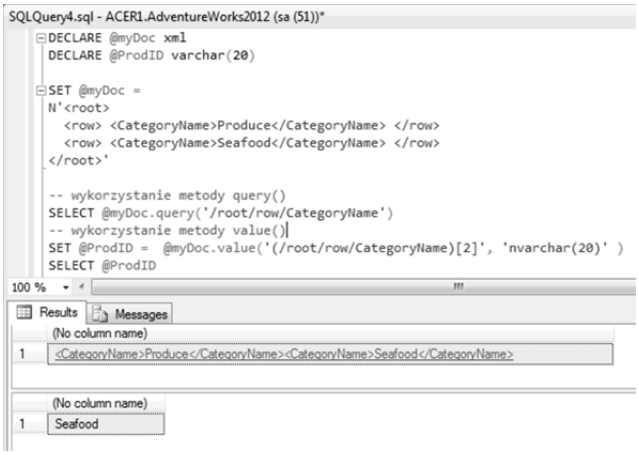


Rys.5. Przedstawienie procesu otrzymania z danych XML danych relacyjnych z wykorzystaniem funkcji OPENXML.

Na Rys. 5. i Rys. 6. przedstawiono drzewo XML otrzymane z dokumentu XML, które zostało utworzone za pomocą procedury sp_xml_preparedocument, a następnie przedstawione w postaci tabeli relacyjnej.



Rys.6. Drzewo XML dokumentu XML (przykładowe), który został utworzony za pomocą sp_xml_preparedocument.



Rys.7. Wykorzystanie metod typu XML query() i value().

Wykorzystanie indeksów na kolumnach XML

Indeksy XML mogą być tworzone na kolumnach typu XML. Indeksować można wszystkie znaczniki, wartości i atrybuty XML. Korzysta się z indeksu XML w następujących sytuacjach:
- gdy zapytania są powszechne wykonywane na kolumnach XML. Należy w takim wypadku brać pod uwagę koszty utrzymania indeksu XML podczas wstawiania i modyfikacji danych.
- gdy dane XML są stosunkowo duże, a pobierane elementy są stosunkowo małe. Zbudowanie indeksu pozwala uniknąć analizowania całych danych i przynosi korzyści dla efektywnego przetwarzania zapytań.

Indeksy XML są następujących typów:
- indeks XML typu podstawowego (Primary) (Rys. 8.)
- indeks XML typu dodatkowego (Secondary) (Rys. 9.)

Definicja tabeli zawierającej wymienione typy indeksów XML ma postać:

```
CREATE TABLE T (Col1 INT primary key,
XmlCol XML)
```

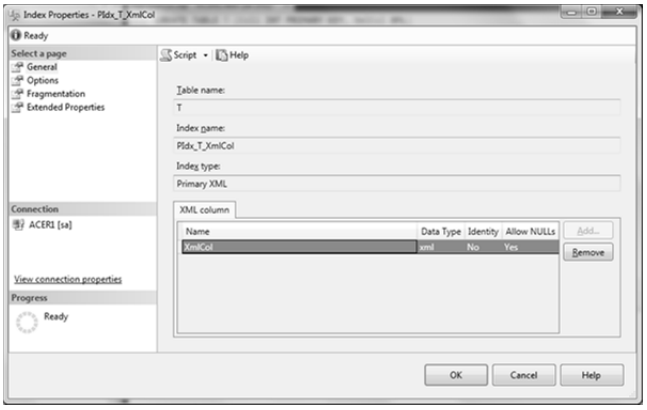
Instrukcja, w której indeks XML typu Primary o nazwie PIdx_T_XmlCol, na kolumnie XML XmlCol w tabeli T ma postać:

```
CREATE PRIMARY XML INDEX PIdx_T_XmlCol
ON T(XmlCol)
```

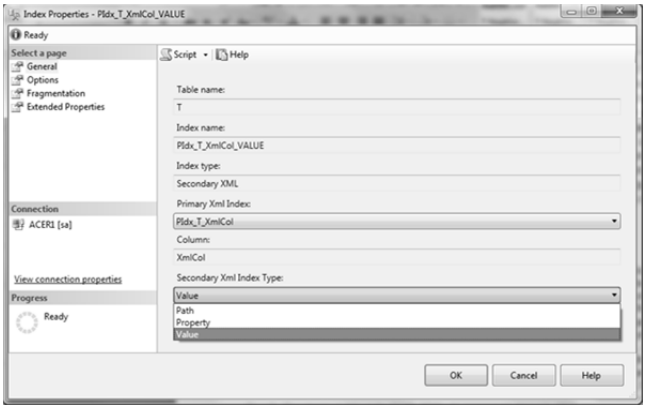
Instrukcja, w której indeks XML typu Secondary (znacznik, wartość, atrybut) ma postać:

```
CREATE XML INDEX PIdx_T_XmlCol_PATH ON
T(XmlCol)
USING XML INDEX PIdx_T_XmlCol
FOR PATH

CREATE XML INDEX PIdx_T_XmlCol_VALUE ON
T(XmlCol)
USING XML INDEX PIdx_T_XmlCol
FOR VALUE

CREATE XML INDEX PIdx_T_XmlCol_PROPERTY ON
T(XmlCol)
USING XML INDEX PIdx_T_XmlCol
FOR PROPERTY
```

Pierwszy indeks na kolumnie typu XML musi być głównym indeksem XML. Korzystanie z indeksu Primary XML, obsługiwane są następujące typy indeksów typu Secondary: znacznik, wartość i atrybut. W zależności od rodzaju zapytań, indeksy te mogą pomóc w poprawie wydajności zapytań. Dodatkowymi parametrami zapewniającymi szybkość przetwarzania danych indeksowanych a co za tym idzie miejsce na dysku potrzebne do przechowywania indeksu to parametry Fill factor i Pad index, które odpowiadają za odpowiednie wypełnienia stron indeksu danymi co ważne jest w przypadku częstych operacji na indeksie tj. wstawianie, modyfikowanie danych XML.

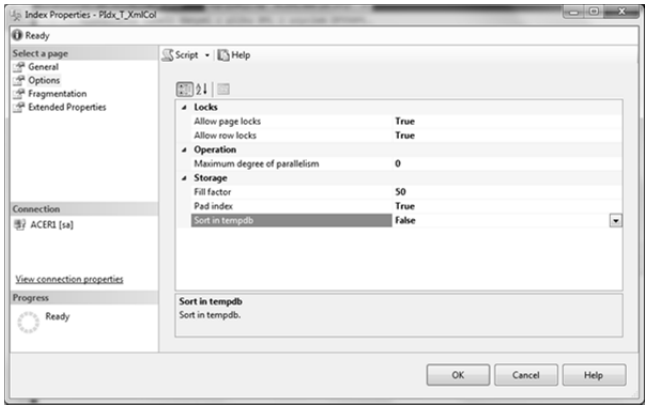


Rys.8. Definiowanie graficzne indeksu XML typu Primary.

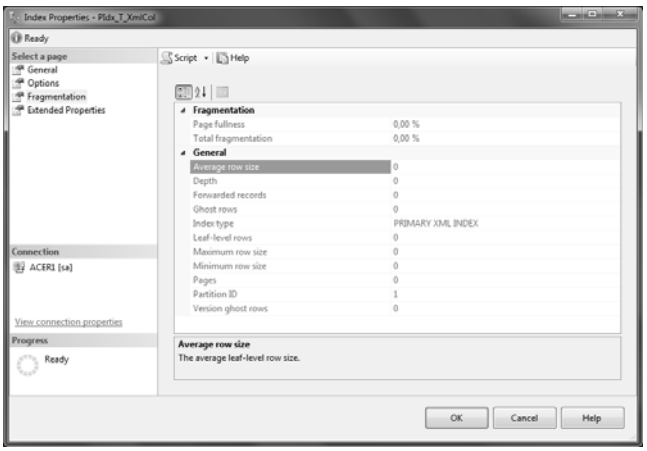


Rys.9. Definiowanie graficzne indeksu XML typu Secondary na wcześniej utworzonym indeksie typu Primary.

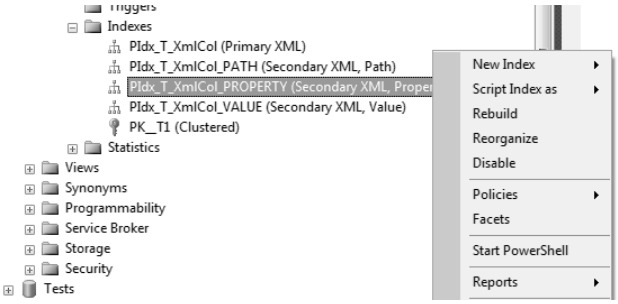
W celu poprawy wydajności przetwarzania tabel wykorzystujemy indeksy, na których w celu ich obsługi możemy wykonywać operacje takie jak Reorganize i Rebuild. Proces Rebuild powoduje skasowanie poprzedniego indeksu i ponowne jego zbudowanie. Natomiast proces Reorganize fizycznie reorganizuje punkty węzłowe liścia indeksu. Przy fragmentacji indeksu od 11% do 30% stosuje się funkcje Reorganize, a powyżej 31% funkcję Rebuild.



Rys.10. Wykorzystanie parametrów definiowania indeksu do poprawy wydajności przetwarzania danych.



Rys.11. Informacje dotyczące fragmentacji danych XML przechowywanej w tabeli z kolumną XML.



Rys.12. Możliwości operacji na indeksach XML z wykorzystaniem opcji Rebuild oraz Reorganize.

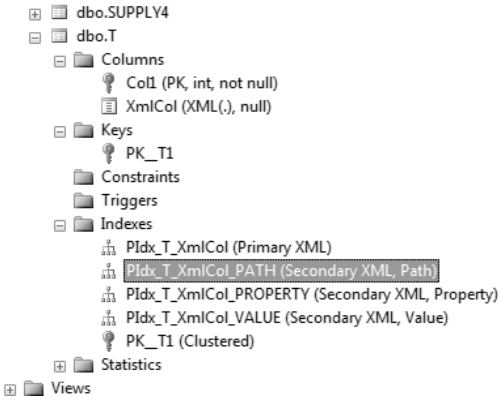
Przykład instrukcji ponownego budowania indeksu ma postać:

```
ALTER INDEX [PIdx_T_XmlCol_PATH] ON [dbo].[T]
REBUILD PARTITION = ALL
```

Przykład instrukcji do reorganizacji istniejące indeksu ma postać:

```
ALTER INDEX [PIdx_T_XmlCol_PATH] ON [dbo].[T]
REORGANIZE WITH ( LOB_COMPACTION = ON )
```

Na Rys.13. przedstawiono między innymi indeksy zbudowane na kolumnie XML i należy pamiętać aby były sprawdzane pod względem fragmentacji danych w celu ich reorganizacji lub skasowania i ponownego zbudowania. Indeksy posiadają współczynniki Fill factor i Pad index, które także należy odpowiednio ustawić przy budowie indeksu.



Rys. 13. Obiekty utworzone na tabeli T posiadającej dwie kolumny w tym jedną XML, klucz PK oraz indeksy zdefiniowane na kolumnie XML.

Podsumowanie

Artykuł prezentuje istniejące możliwości przechowywania i wykonywania zapytań do dokumentów XML w systemie SQL Server. Aktualne w zastosowaniach biznesowych często stosuje się standard XML a co zatem idzie możliwość przechowywania danych XML w bazie danych. Pozwala to na efektywne wyszukiwanie i modyfikację danych poprzez wprowadzenie indeksów.

Podejście do przetwarzania danych XML w istniejących systemach komercyjnych różni się co do sposobu przetwarzania danych i ich przechowywania a co za tym idzie optymalizacji efektywnego przetwarzania. Sam standard XML i technologie wykorzystywane w ramach standardu są rekomendowane oraz specyfikowane przez organizację W3C. Natomiast specyfikacja typu XML po stronie relacyjnej bazy danych standaryzuje ANSI SQL.

LITERATURA

[1] Books online: https://msdn.microsoft.com/pl-pl/library/ms130214(v=sql.110).aspx.
[2] Priscilla Walmsley: XQuery, 2nd Edition, O'Reilly Media 2015, ISBN 978-1-49191-510-3
[3] Benjamin Nevarez: Microsoft SQL Server 2014. Optymalizacja zapytań, Helion 2015
[4] XML - http://www.w3.org/
[5] XPath - http://www.w3.org/TR/xpath/
[6] XQuery - http://www.w3.org/XML/Query/
[7] XML Schema - http://www.w3.org/XML/Schema
[8] Korzeniewska E., Duraj A., Krawczyk A.; Detection of local changes in resistance by means of data mining algorithms, Przegląd Elektrotechniczny, 2014 R.90, nr 12, s. 229-232.

***Autorzy**: dr inż. Paweł Drzymała, Politechnika Łódzka, Instytut Mechatroniki i Systemów Informatycznych, Stefanowskiego 18/22, 90-924 Łódź, E-mail: pawel.drzymala@p.lodz.pl; dr inż. Henryk Welfle, Politechnika Łódzka, Instytut Mechatroniki i Systemów Informatycznych, Stefanowskiego 18/22, 90-924 Łódź, E-mail: henryk.welfle@p.lodz.pl.*