WYKORZYSTANIE XML W RELACYJNYCH BAZACH DANYCH

Andrzej Ptasznik
Warszawska Wyższa Szkoła Informatyki
aptaszni@wwsi.edu.pl

1. Zagadnienia programowe

XML (ang. *Extensible Markup Language)*, w wolnym tłumaczeniu *Rozszerzalny Język Znaczników*, jest obecnie powszechnie stosowanym standardem wymiany danych. W relacyjnych bazach danych wprowadzono typ danych XML, udostępniający wiele metod działających na danych XML, oraz wprowadzono dodatkowe klauzule w poleceniu SELECT języka SQL, które umożliwiają przekształcanie wyniku zapytania do postaci dokumentu XML. W podstawie programowej język XML nie jest uwzględniony, ale ze względu na jego bardzo powszechne stosowanie warto zapoznać uczniów uzdolnionych z aspektami jego wykorzystania w kontekście relacyjnych baz danych.

Informatyka, IV etap edukacyjny, zakres podstawowy

2. Wyszukiwanie, gromadzenie, selekcjonowanie, przetwarzanie i wykorzystywanie informacji, współtworzenie zasobów w sieci, korzystanie z różnych źródeł i sposobów zdobywania informacji. Uczeń:
   1) znajduje dokumenty i informacje w udostępnianych w Internecie bazach danych (np. bibliotecznych, statystycznych, w sklepach internetowych), ocenia ich przydatność i wiarygodność i gromadzi je na potrzeby realizowanych projektów z różnych dziedzin;
4. Opracowywanie informacji za pomocą komputera, w tym rysunków, tekstów, danych liczbowych, animacji, prezentacji multimedialnych i filmów. Uczeń:
   6) tworzy bazę danych, posługuje się formularzami, porządkuje dane, wyszukuje informacje stosując filtrowanie;
   7) wykonuje podstawowe operacje modyfikowania, i wyszukiwania informacji na relacyjnej bazie danych;

Informatyka, IV etap edukacyjny, zakres rozszerzony

2. Wyszukiwanie, gromadzenie, selekcjonowanie, przetwarzanie i wykorzystywanie informacji, współtworzenie zasobów w sieci, korzystanie z różnych źródeł i sposobów zdobywania informacji. Uczeń:
   1) projektuje relacyjną bazę danych z zapewnieniem integralności danych;
   2) stosuje metody wyszukiwania i przetwarzania informacji w relacyjnej bazie danych (język SQL);
   3) tworzy aplikację bazodanową, w tym sieciową, wykorzystującą język zapytań, kwerendy, raporty; zapewnia integralność danych na poziomie pól, tabel, relacji;
   5) opisuje mechanizmy związane z bezpieczeństwem danych: szyfrowanie, klucz, certyfikat, zapora ogniowa.

Uwaga. Przedstawiony scenariusz zajęć wybiega poza zagadnienia programowe i powinien być adresowany do uczniów z lepszym przygotowaniem informatycznym i wykazujących większe zainteresowania pogłębieniem wiedzy z zakresu baz danych.

2. Temat zajęć

Wykorzystanie XML w relacyjnych bazach danych

Zajęcia są poświęcone różnym aspektom wykorzystania języka XML w relacyjnych bazach danych. Omówione zostaną:

- klauzula FOR XML w poleceniu SELECT;
- wykorzystanie metod typu XML do realizacji zapytań;
- przechowywanie danych w kolumnach typu XML;
- wykorzystanie operatora złączeń CROSS APPLY do pobierania danych z wielu dokumentów XML.

Komentarz. Jest to trudny temat, ponieważ łączy w sobie problemy przechowywania i wyszukiwania danych w relacyjnej bazie danych z przechowywaniem i wyszukiwaniem danych w dokumentach XML.

3. Cele zajęć

W wyniku realizacji tych zajęć uczeń powinien umieć:

- przekształcać wyniki zapytań SQL do postaci dokumentów XML;
- wykorzystywać metody typu XML do pozyskiwania danych;
- stosować operator złączeń CROSS APPLY w zapytaniach.

Komentarz. Cele tych zajęć wykraczają poza podstawę programową.

4. Przygotowanie uczniów

Uczniowie przystępujący do tych zajęć powinni:

- wcześniej poznać metodę rozwiązywania problemów z pomocą komputera, składającą się z sześciu etapów;
- znać podstawowe pojęcia z zakresu informatyki i baz danych;
- umieć pisać proste zapytania w języku SQL.

5. Metody pracy

W zajęciach są stosowane następujące metody pracy:

- generalnie, rozwiązywanie każdego rozważanego zagadnienia (problemu) składa się z sześciu etapów, które składają się na metodę rozwiązywania problemów z pomocą komputera; te etapy, to:
  - opis, dyskusja i zrozumienie sytuacji problemowej,
  - podanie specyfikacji problemów do rozwiązania,
  - zaprojektowanie rozwiązania,
  - implementacja (realizacja) rozwiązania w postaci polecenia w języku SQL,

  - testowanie i ewaluacja rozwiązania,
  - prezentacja sposobu otrzymania rozwiązania i samego rozwiązania;
- przygotowanie przez uczniów listy problemów związanych z realizacją zadania;
- samodzielne sporządzenie przez uczniów opisów sposobów rozwiązania poszczególnych problemów;
- zapisanie odpowiednich poleceń w języku SQL i ich testowanie;
- samodzielne testowanie poprawności zapisanych poleceń;
- prezentacja otrzymanych rozwiązań.

6. Formy pracy

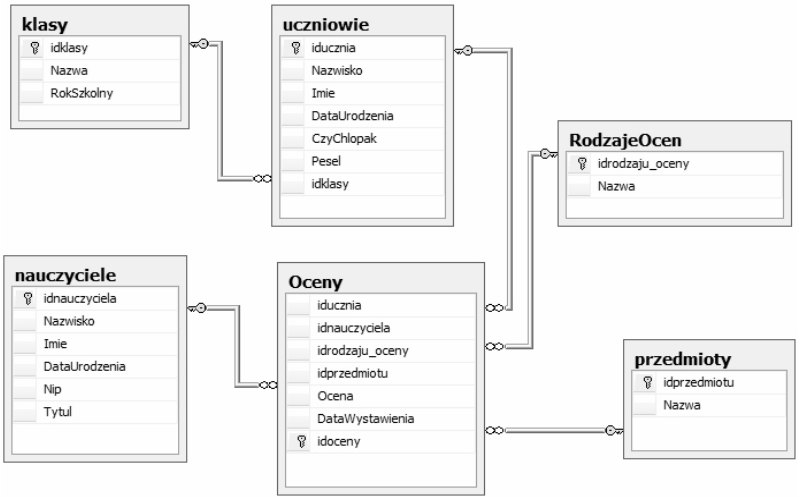
Założone cele są realizowane za pomocą następujących form pracy:

- podczas burzy mózgów prowadzonej przez całą klasę lub w grupach uczniów – ma to doprowadzić do sformułowania sposobu rozwiązania problemu;
- przygotowywanie opisów rozwiązania może odbywać się indywidualnie lub w parach uczniów;
- polecenia SQL uczniowie piszą samodzielnie;
- testowanie poleceń może odbywać się w grupach uczniów lub samodzielnie (w zależności od stopnia zaawansowania);
- końcowym efektem pracy nad danym problemem jest prezentacja napisanego polecenia oraz wyników jego działania.

Uwaga: W trakcie lekcji należy podkreślać fakt, że dokument XML jest pewną formą bazy danych.

7. Materiały pomocnicze

Do przeprowadzenia zajęć niezbędne jest zainstalowanie oprogramowania SQL Server 2008 R2 Express Edition (darmowe oprogramowanie możliwe do pobrania ze stron firmy Microsoft). Po zainstalowaniu oprogramowania serwera baz danych należy utworzyć przykładową bazę danych, a w niej zdefiniować tabele, według schematu przedstawionego na rysunku 1.



Rysunek 1. Schemat bazy danych ElektroniczmyDziennikOcen

8. Środki dydaktyczne

Uczniowie wykorzystują w czasie tych zajęć:

- materiały dotyczące baz danych, opracowane w projekcie Informatyka +;
- komputer i jego podstawowe oprogramowanie, w tym oprogramowanie MS SQL2008 R2 Express Edition oraz MS SQL Server Menagement Studio.

Kmentarz. Oprogramowanie MS SQL Server 2008 R2 Express Edition jest darmowe, można je pobrać ze strony http://www.microsoft.com/express/Database.

9. Przebiegu zajęć (kolejnych lekcji)

Zajęcia, których celem jest wprowadzenie do wykorzystania dokumentów XML w relacyjnych bazach danych mogą być rozłożone na kilka lekcji, niekoniecznie kolejnych. W niniejszej propozycji przyjęliśmy następujący tryb realizacji omawianego tematu:

- na początku, nauczyciel objaśnia budowę dokumentu XML i omawia podstawowe elementy składni klauzuli FOR XML wykorzystywanej w poleceniu SELECT (Lekcja 1);
- na kolejnym etapie wyjaśniane jest wykorzystywanie metod typu XML (Lekcja 2);
- na kolejnym etapie uczniowie wykonują przykłady zapytań w których wykorzystywane są jednocześnie, dane z tabel relacyjnych i dane z XML(Lekcja 3);
- rozwiązywanie każdego z zagadnień składa się z sześciu etapów, wyżej wymienionych (w Metodach pracy).

Lekcja 1. Przekształcanie wyników zapytań do postaci dokumentów XML. Czas: 60 min.

W tej części zajęć uczniowie zapoznają się z działaniem klauzuli FOR XML polecenia SELECT języka SQL, która umożliwia przekształcanie wyników zapytania do postaci dokumentu XML. Pierwsze próby wykują na bazie następującej postaci zapytania;

```
SELECT TOP 3 Nazwisko,
             Imie,
             Pesel,
             DataUrodzenia,
             CASE CzyChlopak
               WHEN 1 THEN 'Mężczyzna'
               ELSE 'Kobieta'
             END AS Płeć
FROM Uczniowie
```

Przykładowy wynik tego zapytania jest przedstawiony na rysunku 2.

Nazwisko	Imie	Pesel	DataUrodzenia	Płeć
Kotek	Kasia	92031275446	1992-03-12	Kobieta
Piesek	Jan	92051587746	1992-05-15	Mężczyzna
Lisek	Kasia	92022277654	1992-02-22	Kobieta

Rysunek 2. Przykladowa postać wyniku zapytania

W kolejnym kroku należy zademonstrować uczniom najprostszą postać przekształcenia wyniku zapytania do postaci XML. Osiągniemy to dodając do przykładowego zapytania klauzulę FOR XML RAW. Zapytanie będzie miało następującą postać;

```
SELECT TOP 3 Nazwisko,
        Imie,
        Pesel,
        DataUrodzenia,
        CASE CzyChlopak
          WHEN 1 THEN 'Mężczyzna'
          ELSE 'Kobieta'
        END AS Płeć
FROM Uczniowie
FOR XML RAW.
```

Wynik tego zapytania ma postać przedstawioną na rysunku 3.

```
<row Nazwisko="Kotek" Imie="Kasia" Pesel="92031275446" DataUrodzenia="1992-03-12" Płeć="Kobieta" />
<row Nazwisko="Piesek" Imie="Jan" Pesel="92051587746" DataUrodzenia="1992-05-15" Płeć="Mężczyzna" />
<row Nazwisko="Lisek" Imie="Kasia" Pesel="92022277654" DataUrodzenia="1992-02-22" Płeć="Kobieta" />
```

Rysunek 3. Wynik zapytania w postaci XML

Omawiając ten przykład należy zwrócić uczniom uwagę na budowę tego wyniku. Każdy wiersz zwracany przez zapytanie został przekształcony w element o nazwie **row** a kolejne kolumny są umieszczone wewnątrz elementu jako atrybuty. Prawidłowy dokument XML powinien zawierać element główny (ang. *root*) a w pokazanym przykładzie takiego elementu nie ma.

W kolejnych krokach należy przedstawić modyfikacje klauzuli FOR XML, w wyniku których będzie można otrzymać różne postaci dokumentu XML. Ostateczna postać klauzuli FOR XML będzie miała następującą postać:

FOR XML RAW('Uczen'), ELEMENTS, ROOT('ListaUczniow'),

a wynik zapytania jest pokazany na rysunku 4.

```
<ListaUczniow>
  <Uczen>
    <Nazwisko>Kotek</Nazwisko>
    <Imie>Kasia</Imie>
    <Pesel>92031275446</Pesel>
    <DataUrodzenia>1992-03-12</DataUrodzenia>
    <Płeć>Kobieta</Płeć>
  </Uczen>
  <Uczen>
    <Nazwisko>Piesek</Nazwisko>
    <Imie>Jan</Imie>
    <Pesel>92051587746</Pesel>
    <DataUrodzenia>1992-05-15</DataUrodzenia>
    <Płeć>Mężczyzna</Płeć>
  </Uczen>
  <Uczen>
    <Nazwisko>Lisek</Nazwisko>
    <Imie>Kasia</Imie>
    <Pesel>92022277654</Pesel>
    <DataUrodzenia>1992-02-22</DataUrodzenia>
    <Płeć>Kobieta</Płeć>
  </Uczen>
</ListaUczniow>
```

Rysunek 4. Przykładowa postać wyniku zapytania

Doprowadzenie wyniku do postaci pokazanej na rysunku 4 można realizować sekwencyjnie, pokazując działanie poszczególnych opcji klauzuli FOR XML:

- RAW('Uczen') – zastosowanie tej opcji powoduje nadanie nazwy elementowi dokumentu XML (zamiast domyślnej nazwy row);
- ELEMENTS – zastosowanie tej opcji powoduje budowanie wyniku w postaci elementów a nie atrybutów, jak pokazano na rysunku 3;
- ROOT('ListaUczniow') – opcja powodująca dodanie do wyniku elementu głównego dokumentu XML.

W kolejnym przykładzie można pokazać przekształcenie do postaci dokumentu o budowie mieszanej (część danych w atrybutach a część danych w elementach), co możemy osiągnąć stosując w klauzuli FOR XML opcję PATH zamiast opcji RAW. Zapytanie będzie miało postać:

```
SELECT TOP 3 Nazwisko as „@Nazwisko",
             Imie as "@Imie",
             Pesel,
             DataUrodzenia,
            CASE CzyChlopak
                WHEN 1 THEN 'Mężczyzna'
                ELSE 'Kobieta'
                 END AS Płeć
FROM Uczniowie
FOR XML PATH('Uczen'), ROOT('ListaUczniow').
```

Wynik tego zapytania jest pokazany na rysunku 5. Zastosowana w zapytaniu opcja PATH klauzuli FOR XML, umożliwia umieszczanie wybranych danych w atrybutach, co osiągamy nadając nazwę zastępczą kolumny (np. **Imie** as **„@Imie")** umieszczoną w cudzysłowach i poprzedzoną znakiem @.

```
<ListaUczniow>
  <Uczen Nazwisko="Kotek" Imie="Kasia">
    <Pesel>92031275446</Pesel>
    <DataUrodzenia>1992-03-12</DataUrodzenia>
    <Płeć>Kobieta</Płeć>
  </Uczen>
  <Uczen Nazwisko="Piesek" Imie="Jan">
    <Pesel>92051587746</Pesel>
    <DataUrodzenia>1992-05-15</DataUrodzenia>
    <Płeć>Mężczyzna</Płeć>
  </Uczen>
  <Uczen Nazwisko="Lisek" Imie="Kasia">
    <Pesel>92022277654</Pesel>
    <DataUrodzenia>1992-02-22</DataUrodzenia>
    <Płeć>Kobieta</Płeć>
  </Uczen>
</ListaUczniow>
```

Rysunek 5. Wynik zapytania – dokument XML o budowie mieszanej

Kolejny przykład dotyczy budowy dokumentu XML z wykorzystaniem podzapytań. W tym przykładzie chcemy rozbudować wynikową postać dokumentu XML o informacje o dwóch ostatnich ocenach otrzymanych przez danego ucznia. Zapytanie będzie miało nastepujacą postać:

```
SELECT TOP 2 Nazwisko as „@Nazwisko, Imie as „@Imie”,
        Pesel, DataUrodzenia,
        CASE CzyChlopak
          WHEN 1 THEN ‘Mężczyzna’
          ELSE ‘Kobieta’
        END AS Płeć,
                (
                 SELECT TOP 2 Nazwa as Przedmiot,
                              Ocena,
                              DataWystawienia
                  FROM Oceny join Przedmioty ON Oceny.idprzedmiotu=Przedmioty.idprzedmiotu
                 WHERE iducznia=Uczniowie.iducznia
                 ORDER BY DataWystawienia DESC
                 FOR XML RAW(‘Ocena’),ELEMENTS, TYPE, ROOT(‘OstatnieOceny’)
                )
FROM Uczniowie
FOR XML PATH(‘Uczen’), ROOT(‘ListaUczniow’)
```

Wynik zapytania jest przedstawiony na rysunku 6. Należy omówić z uczniami nowe elementy wykorzystane w tym zapytaniu. Uczniom szczególnie zainteresowanym i uzdolnionym można zlecić wyjaśnienie znaczenia opcji TYPE wykorzystanej w zapytaniu wewnętrznym omawianego przykładu.

Rysunek 6. Przykładowy wynik zapytania

Lekcja 2. Pobieranie danych z dokumentu XML. Czas: 60 min.

Celem tej lekcji jest zapoznanie uczniów ze sposobami wykorzystania metod typu danych XML do realizacji zapytań. Typ danych XML w środowisku SQL Server udostępnia 5 metod:

- metoda value() – zwraca z dokumentu XML wskazaną wartość skalarną;
- metoda query() – zwraca wskazane elementy dokumentu XML (wynik jest typu XML);
- metoda nodes() – zwraca tabelę;

- metoda exist() – zwraca wartość logiczną informującą, czy wskazany w parametrze metody element istnieje w dokumencie XML;
- metoda modify() – modyfikuje dokument XML.

W ramach lekcji prezentowane jest działanie metod: query(), value() i nodes().

Do realizacji ćwiczeń należy przygotować przykładowy skrypt, w którym jest zadeklarowana zmienna typu XML. Przypisujemy jej dokument XML otrzymany w czasie lekcji 1 (lista uczniów z dwoma ostatnimi ocenami). Skrypt może mieć następującą postać:

```
DECLARE @dane xml='<ListaUczniow>
 <Uczen Nazwisko="Kotek" Imie="Kasia">
  <Pesel>92031275446</Pesel>
  <DataUrodzenia>1992-03-12</DataUrodzenia>
  <Płeć>Kobieta</Płeć>
  <OstatnieOceny>
   <Ocena>
    <Przedmiot>Literatura</Przedmiot>
    <Ocena>4.00</Ocena>
    <DataWystawienia>2009-02-15</DataWystawienia>
   </Ocena>
   <Ocena>
    <Przedmiot>Literatura</Przedmiot>
    <Ocena>3.00</Ocena>
    <DataWystawienia>2009-02-12</DataWystawienia>
   </Ocena>
  </OstatnieOceny>
 </Uczen>
 <Uczen Nazwisko="Piesek" Imie="Jan">
  <Pesel>92051587746</Pesel>
  <DataUrodzenia>1992-05-15</DataUrodzenia>
  <Płeć>Mężczyzna</Płeć>
  <OstatnieOceny>
   <Ocena>
    <Przedmiot>Fizyka</Przedmiot>
    <Ocena>5.00</Ocena>
    <DataWystawienia>2009-02-09</DataWystawienia>
   </Ocena>
   <Ocena>
    <Przedmiot>Informatyka</Przedmiot>
    <Ocena>4.00</Ocena>
    <DataWystawienia>2009-01-21</DataWystawienia>
   </Ocena>
  </OstatnieOceny>
 </Uczen>
</ListaUczniow>'.
```

Pierwszy przykład – wykorzystanie metody query()

Zadaniem jest wybranie wszystkich elementów DataUrodzenia zawartych w przykładowym dokumencie XML. W najprostszej postaci, jako parametr metody query() przekazujemy adres węzłów dokumentu XML. Wynikiem metody jest zbiór wskazanych elementów znajdujących się w dokumencie. Postać skryptu do realizacji tego zadania jest pokazana na rysunku 7.

```
DECLARE @dane XML='<ListaUczniow>...

Select @dane.query('ListaUczniow/Uczen/DataUrodzenia')
```

Rysunek 7. Postać skryptu wykorzystującego metodę query()

Wynik zapytania zawiera elementy DataUrodzenia znalezione w dokuemncie zapisanym w zmiennej @dane. W naszym przykładzie są to dwa nastepujące:

<DataUrodzenia>1992-03-12</DataUrodzenia>
<DataUrodzenia>1992-05-15</DataUrodzenia>

Omiawiając wykonanie tego przykładu należy zwrócić uwagę, że wynik metody jest fragmentem wyjściowego dokumentu XML i nie zawiera elementu głównego.

Drugi przykład – wykorzystanie metody value()

Metoda value() zwraca wskazaną wartość skalarną z dokumentu XML. Metoda wymaga dwóch parametrów: pierwszy – dokładnie adresuje wybierany element, a drugi – określa typ danych, do którego zostanie przekształcona odczytana wartość. Na rysunku 8 jest pokazana postać skryptu, który zwraca datę urodzenia zapisaną w drugim elemencie DataUrodzenia znalezionym w dokumencie XML.

```
DECLARE @dane XML='<ListaUczniow>...

Select @dane.value('(ListaUczniow/Uczen/DataUrodzenia)[2]', 'date')
```

Rysunek 8. Postać skryptu wykorzystującego metode value()

W wyniku działania skryptu pokazanego na rysunku 8 otrzymamy datę 1992-05-15. Szczególna uwagę należy zwrócić na konieczność jednoznacznego adresu wyszukiwanej wartości, ponieważ standard XML zezwala na wielokrotne wystąpienie elementu o danej nazwie w konkretnym kontekście. Dla zapewnienia jednzoznaczej adresacji stosujemy indeks określajacy, o które wystąpienie szukanego elementu chodzi. Konstrukcja (ListaUczniow/Uczen/DataUrodzenia)[2] określa drugie wystąpienie elementu DataUrodzenia zawartego w elemenci Uczen, który jest zawarty w elemencie ListaUczniow.

Trzeci przykład – wykorzystanie metody nodes()

Metoda nodes() jest szczególnie ważna, ponieważ pozwala przekształcić dokument XML do postaci tabeli, co umożliwia wykorzystanie wyniku tego przekształcenia w zapytaniach SQL. Skrypt, wykorzystujący metodę nodes, jest pokazany na rysunku 9.

```
DECLARE @dane XML='<ListaUczniow>...

SELECT k.value('@Nazwisko' , 'varchar(64)') as Nazwisko,
       k.value('@Imie' , 'varchar(64)') as Imie,
       k.value('DataUrodzenia[1]' , 'date') as DataUrodzenia,
       k.value('Pesel[1]' , 'char(11)') as Nazwisko
FROM @dane.nodes('ListaUczniow/Uczen') t(k)
```

Rysunek 9. Postać skryptu wykorzystującego metode nodes()

Parametrem metody nodes() jest adres węzła dokumentu XML (w naszym przykładzie ListaUczniow/Uczen) i zwraca tyle wierszy, ile razy wskazany element występuje w dokumencie. Ponieważ w klauzuli FROM zapytania SELECT wykorzystujemy wyrażenie budujące tabelę, to wymogiem składniowym języka SQL jest, by nadać tej tabeli nazwę. Wykorzystana w omawianym przykładzie postać nazwy zastępczej t(k) oznacza, że tabela powstała z dokumentu XML będzie miała nazwę t a kolumna będąca wynikiem działania metody nodes() będzie miała nazwę k. Kolumna wynikowa, zwracana przez metodę nodes, nie może być wykorzystana bezpośrednio lecz jedynie z użyciem metod typu XML. W przykładzie z kolumny k pobierane są wartości z atrybutów @Nazwisko i @Imie oraz dane z elementów DataUrodzenia i Pesel. Wynik przykładowego skryptu jest pokazany na rysunku 10.

Nazwisko	Imie	DataUrodzenia	Nazwisko
Kotek	Kasia	1992-03-12	92031275446
Piesek	Jan	1992-05-15	92051587746

Rysunek 10. Wynik zastosowania metody nodes()

Zainteresowanym uczniom można polecić wykonanie zapytania, które zwróci nazwisko i imie ucznia oraz nazwę przedmiotu i wystawioną ocenę.

Lekcja 3. Zapytania wykorzystujące dane relacyjne i XML. Czas: 60 min.

Wprowadzenie typu XML do standardu SQL spowodowało, że projektując bazę danych, w tabelach relacyjnych są umieszczane kolumny zawierające dane XML. Zapytania języka SQL muszą korzystać z różnych postaci danych. W ramach tej lekcji są przedstawiane dwa przykłady takich zapytań. Przed przystąpieniem do ćwiczenia należy przygotować odpowiednią tabelę, która będzie zawierała kolumnę typu XML. W tym celu należy wykonać następujące zapytanie:

```
SELECT Nazwisko,
       Imie,
       Pesel,
       DataUrodzenia,
       CzyChlopak,
      (
          SELECT  Nazwa as Przedmiot,
                  Ocena,
                  DataWystawienia
           FROM Oceny join Przedmioty ON Oceny.idprzedmiotu=Przedmioty.idprzedmiotu
```

```
        WHERE iducznia=Uczniowie.iducznia
        ORDER BY DataWystawienia DESC
        FOR XML RAW('Ocena'),ELEMENTS, TYPE, ROOT('ListaOcen')
    ) as OcenyUcznia
INTO UczniowieXML
FROM Uczniowie
```

Klauzula INTO spowoduje, że w bazie danych zostanie utworzona tabela o nazwie UczniowieXML, będąca wynikiem tego zapytania. Przykładowa zawartość tej tabeli jest pokazana na rysunku 11.

Nazwisko	Imie	Pesel	DataUrodzenia	CzyChlopak	OcenyUcznia
Kotek	Kasia	92031275446	1992-03-12	0	<ListaOcen><Ocena><Przedmiot>Literatura</Przedmiot><Ocena>4.00</Ocena><DataWystawienia>2...
Piesek	Jan	92051587746	1992-05-15	1	<ListaOcen><Ocena><Przedmiot>Fizyka</Przedmiot><Ocena>5.00</Ocena><DataWystawienia>200...
Lisek	Kasia	92022277654	1992-02-22	0	<ListaOcen><Ocena><Przedmiot>Fizyka</Przedmiot><Ocena>3.00</Ocena><DataWystawienia>200...
Kurka	Jola	92060288788	1992-06-02	0	<ListaOcen><Ocena><Przedmiot>Geografia</Przedmiot><Ocena>2.00</Ocena><DataWystawienia>...
Gąska	Wacek	91031199123	1991-03-11	0	<ListaOcen><Ocena><Przedmiot>Geografia</Przedmiot><Ocena>1.00</Ocena><DataWystawienia>...
Krówka	Rysio	92051577646	1992-05-15	1	<ListaOcen><Ocena><Przedmiot>Informatyka</Przedmiot><Ocena>4.00</Ocena><DataWystawienia...
Zebra	Wojtek	93031399846	1993-03-13	1	<ListaOcen><Ocena><Przedmiot>Geografia</Przedmiot><Ocena>1.00</Ocena><DataWystawienia>...

Rysunek 11. Przykładowa zawartość tabeli UczniowieXML

W kolumnie OcenyUcznia jest zapisany dokument XML w postaci, której przykład jest pokazany na rysunku 12.

```
<ListaOcen>
  <Ocena>
    <Przedmiot>Literatura</Przedmiot>
    <Ocena>4.00</Ocena>
    <DataWystawienia>2009-02-15</DataWystawienia>
  </Ocena>
  <Ocena>
    <Przedmiot>Literatura</Przedmiot>
    <Ocena>3.00</Ocena>
    <DataWystawienia>2009-02-12</DataWystawienia>
  </Ocena>
  <Ocena>
    <Przedmiot>Literatura</Przedmiot>
    <Ocena>4.00</Ocena>
    <DataWystawienia>2009-02-12</DataWystawienia>
  </Ocena>
</ListaOcen>
```

Rysunek 12. Przykładowa postać dokumentu XML

Pierwszy przykład – obliczenie średniej oceny ucznia

W tym przykładzie chcemy uzyskać tabelę wynikową zawierajacą nazwisko, imię i Pesel ucznia oraz jego średnią ocen. Zwracamy uczniom uwagą na fakt, że część danych możemy pobrac bezposrednio z tabeli UczniowieXML, natomiast średnią ocenę możemy uzyskać wykonując zapytanie odwołujace się do kolumny OcenyUcznia (dane XML). Polecenie realizujace to zadanie ma następującą postać:

```
SELECT Nazwisko,
       Imie,
       Pesel ,
```

```
        (
          SELECT AVG(k.value('Ocena[1]','decimal(5,2)'))
          FROM OcenyUcznia.nodes('ListaOcen/Ocena') t(k)
        ) as Średnia
FROM UczniowieXML.
```

Problem sprowadził się do wykonania dla każdego wiersza tabeli UczniowieXML zapytania obliczającego średnią z dokumentu XML, zapisanego w kolumnie OcenyUcznia. Przykładowy wynik zapytania jest przedstawiony na rysunku 13.

Nazwisko	Imie	Pesel	Średnia
Kotek	Kasia	92031275446	3.000000
Piesek	Jan	92051587746	2.980000
Lisek	Kasia	92022277654	3.229508
Kurka	Jola	92060288788	3.030769
Gąska	Wacek	91031199123	2.857142
Krówka	Rysio	92051577646	2.781818
Zebra	Wojtek	93031399846	3.075757
Gazela	Basia	92111177446	2.932432
Sarenka	Rysio	92121278766	2.563636
Konik	Kasia	93031275446	2.686274
Ryba	Jan	93051587746	3.123076
Kura	Kasia	93022277654	3.035087

Rysunek 13. Przykładowa postać wyniku zapytania

Drugi przykład – obliczenie średniej oceny z poszczególnych przedmiotów

W tym przykładzie, do wykonania zadania jest potrzebne pobranie danych ze wszystkich dokumentów XML zapisanych w kolumnie OcenyUcznia tabeli UczniowieXML. Do realizacji zadania zastosujemy operator złączenia tabel CROSS APPLY, który dla każdego wiersza tabeli umieszczonej po lewej stronie operatora, tworzy dynamicznie tabelę zgodnie z wyrażeniem umieszczonym po prawej stronie operatora i łączy ten wiersz z każdym wierszem otrzymanym w wyniku wykonania wyrażenia. Operator złączeń CROSS APPLY umożliwi połączenie wiersza opisującego danego ucznia z jego ocenami uzyskanymi z zapytania odwołującego się do kolumny OcenyUcznia. Zapytanie będzie miało postać:

```
SELECT Przedmiot,
        AVG(Ocena) as Średnia
FROM UczniowieXML  CROSS APPLY
                        (
                          SELECT k.value('Przedmiot[1]', 'varchar(64)') as Przedmiot,
                                  k.value('Ocena[1]', 'decimal(5,2)') as Ocena
                          FROM  OcenyUcznia.nodes('ListaOcen/Ocena') as t(k)
                        ) as Oceny
GROUP BY Przedmiot.
```

W wyniku otrzymujemy tabelę zawierającą nazwę przedmiotu oraz średnią ocenę z danego przedmiotu – przykładowy wynik jest pokazany na rysunku 14.

Przedmiot	Średnia
Informatyka	2.862745
Literatura	3.125000
Fizyka	2.891891
Geografia	3.058823
Matematyka	3.024590

Rysunek 14. Przykładowy wynik zapytania wykorzystującego operator CROSS APPLY

W ramach lekcji można polecić uczniom wykonanie zmodyfikowanych zapytań, dodając dodatkowe warunki. (np. średnia ocena z poszczególnych przedmiotów uzyskanych w bieżącym roku).