

XI Konferencja PLOUG  
Kościelisko  
Październik 2005

# Język XQuery jako narzędzie do integracji danych – Oracle XML Data Synthesis

Tomasz Traczyk

*Politechnika Warszawska*

*e-mail: ttraczyk@ia.pw.edu.pl*

**Streszczenie:**

Referat stanowić będzie kontynuację tematyki prezentowanej na poprzedniej konferencji PLOUG. Na przykładzie nowej technologii Oracle o nazwie XML Data Synthesis (XDS) przedstawione zostaną możliwości użycia języka XQuery do integracji danych.

**Informacja o autorze:**

Autor jest adiunktem w Instytucie Automatyki i Informatyki Stosowanej na Wydziale Elektroniki i Technik Informatycznych Politechniki Warszawskiej.

## 1. Wprowadzenie

Jedną z częstych potrzeb na jakie natrafiają twórcy współczesnych systemów informacyjnych jest konieczność odpowiedniego połączenia informacji pochodzącej z różnych źródeł, zarówno wewnętrznych (zlokalizowanych w tej samej organizacji) jak i zewnętrznych.

### 1.1. Integracja danych

Integrując dane z wielu źródeł chcielibyśmy zwykle osiągnąć następujące cele:

- uzyskiwać na bieżąco najnowsze informacje, odpowiednio wybrane i połączone,
- móc łatwo zmieniać zakres uzyskiwanych informacji,
- łatwo dostosowywać się do zmian w strukturze czy formacie informacji źródłowych.

Mechanizm łączenia danych powinien zatem działać na bieżąco (*on line*) oraz charakteryzować się elastycznością.

#### 1.1.1. Metody integracji danych

Połączenie danych z różnych źródeł dziś zwykle uzyskuje się na jeden z trzech sposobów:

- Organizując bezpośrednią wymianę informacji między systemami/aplikacjami stanowiącymi źródła i odbiorców danych; wymaga to jednak z reguły ingerencji w systemy-źródła, więc w przypadku systemów zewnętrznych bywa nierealizowalne; zawsze też wiąże się ze znacznymi kosztami i ryzykiem wynikającym ze zwiększenia komplikacji systemów.
- Tworząc repozytorium, gromadzące informacje z potrzebnych źródeł, np. hurtownię danych. To podejście wymaga jednak replikowania danych, co pociąga za sobą typowe dla replikacji problemy, np. konieczność aktualizacji replik.
- Tworząc specjalizowane rozwiązania, integrujące dane dla konkretnych potrzeb; takie podejście nie jest jednak oczywiście ogólne i wymaga osobnego projektu dla każdej nowej potrzeby związanej z łączeniem danych; na ogół wiąże się to ze znacznymi kosztami i mizerną elastycznością.

Niestety, sposoby te nie są ani wygodne, ani ogólne, ani elastyczne.

#### 1.1.2. Dane ze źródeł heterogenicznych

Trudności w integrowaniu informacji szczególnie powiększa fakt, że potrzebne dane na ogół pochodzą z heterogenicznych źródeł. Pół biedy, jeśli są to jedynie różne, ale relacyjne bazy danych; użyć wówczas można bramek (*gateways*) lub uniwersalnych interfejsów (ODBC, JDBC), a wspólny relacyjny model danych pozwala na dość łatwe połączenie informacji. Ze znacznie gorszą sytuacją mamy do czynienia wówczas, gdy dane pochodzą ze źródeł o istotnie różnej strukturze, np. także z różnorodnych plików czy arkuszy kalkulacyjnych. Wówczas integracja następczo może powodować już nie tylko czysto techniczne, ale także koncepcyjne.

#### 1.1.3. Integracja danych w dobie WWW

Współcześnie źródłem wielu istotnych informacji, które muszą być uwzględniane w procesie integracji danych, jest Internet. Informacje uzyskiwane tą drogą mogą mieć różny format (HTML, XML, PDF itp.) oraz być dostępne przez różne protokoły (ftp, „czysty” http, WebDav, SOAP itp.). Szczególnie interesujące wydają się być źródła dostępne jako usługi sieciowe (*Web Services*), jak się bowiem wydaje technologia ta zdobywa coraz większe uznanie w świecie gospodarki elektronicznej typu B2B (*business-to-business*).

## 1.2. Rola języka XML w integracji danych

Pojawienie się i spopularyzowanie języka XML stanowi istotny krok ku ułatwieniu integracji informacji. Wiele źródeł danych albo wprost udostępnia informacje w XML – tak jest np. w przypadku usług sieciowych (*Web Services*), albo stosunkowo łatwo informacje te można przekształcić na XML – dotyczy to np. relacyjnych baz danych, arkuszy kalkulacyjnych czy – w mniejszym stopniu – źródeł w HTML. XML jest więc naturalnym kandydatem na uniwersalny język dostarczania danych, które mają podlegać integracji.

## 2. XQuery a integracja danych

Samo przedstawienie danych źródłowych we wspólnym formacie nie rozwiązuje jeszcze problemu integracji; trzeba teraz dokonać właściwego połączenia informacji. Oczywiście istnienie wspólnego języka, w którym można łatwo wyrazić dane dostarczane przez różne heterogeniczne źródła, znacznie ułatwia ten proces, ale niezbędne jest jeszcze narzędzie umożliwiające wyszukiwanie informacji, jej przekształcanie i łączenie. Najlepiej byłoby, gdyby pożądaný sposób przetwarzania dało się określać deklaratywnie za pomocą narzędzia możliwie uniwersalnego. Takie cechy mają języki zapytań; w przypadku danych wyrażonych w XML odpowiednim kandydatem jest w sposób oczywisty XQuery.

### 2.1. Przypomnienie: czym jest XML Query?

Rosnąca popularność języka XML sprawiła, że rośnie potrzeba istnienia wygodnych i silnych narzędzi, które umożliwiłyby wyszukiwanie i przetwarzanie informacji w zasobach dokumentów XML-owych. Dobrym rozwiązaniem wydaje się stworzenie odpowiedniego języka zapytań.

Język XQuery (XML Query), jest właśnie takim językiem zapytań, zaproponowanym przez *World Wide Web Consortium* [W3q05]. Zapytania języka bazują na ścieżkach XPath 2.0 [W3p05], obudowanych konstrukcjami umożliwiającymi równoczesne korzystanie z wielu źródeł danych XML-owych, wykonywanie obliczeń (szczególnie agregacji), użycie zmiennych pomocniczych, definiowanie i wywoływanie funkcji użytkownika itp.

Język XQuery jest językiem funkcyjnym, którego konstrukcje składają się z wyrażeń zwracających tzw. sekwencje, czyli ciągi węzłów oraz wartości prostych (patrz [Tra03, W3p05]). Najprostszym zapytaniem w XQuery jest po prostu wyrażenie ścieżkowe XPath (dostępna jest niemal pełna składnia ścieżek XPath 2.0). Bardziej złożone zapytania konstruuje się używając charakterystycznych dla XQuery wyrażeń FLWOR, składających się z klauzul **for**, **let**, **where**, **order by** i **return**:

- klauzula **for** przypisuje do zmiennej kolejne pozycje sekwencji zwróconej przez wyrażenie XPath;
- klauzula **let** dokonuje jednokrotnego podstawienia wyniku wyrażenia do zmiennej;
- klauzule **where** i **order by** mają znaczenie podobne jak w SQL;
- klauzula **return** określa postać wyniku, umożliwiając dokonywanie różnorodnych przekształceń.

XQuery wykorzystuje przestrzenie nazw; wprowadzono kilka predefiniowanych przestrzeni i prefiksów, np. dla typów danych XML Schema i XPath oraz funkcji wbudowanych XPath, określić można też inne przestrzenie nazw.

## 2.2. XQuery w technologii Oracle

W najnowszych wersjach podstawowych produktów Oracle pojawiły się rozbudowane mechanizmy, pozwalające korzystać z języka XQuery.

I tak w wersji 10g *release 2* bazy danych Oracle wbudowano wsparcie dla zapytań w języku XQuery. Udostępniono nowe funkcje SQL/XML:

- funkcja **XMLQuery** wykonuje podane jako parametr zapytanie i zwraca w wyniku fragment dokumentu (o typie **XMLType**);
- funkcja **XMLTable** wykonuje zapytanie XQuery, a elementy wynikowej sekwencji umieszcza w wierszach wirtualnej tabeli, która może być wykorzystana w klauzuli **FROM** zapytań SQL.

W programie SQL\*Plus pojawiła się też nowa komenda **XQUERY**, umożliwiająca interaktywne wykonywanie zapytań XQuery.

Z kolei w planowaną na koniec bieżącego roku wersję serwera aplikacyjnego *Oracle Application Server* ma zostać wbudowana technologia XDS, stanowiąca przedmiot tego opracowania.

## 2.3. Zastosowanie XQuery do integracji danych

Jak wspomniano wyżej, jeśli dane podlegające integracji dają się przedstawić w postaci XML – a taka reprezentacja jest możliwa i stosunkowo łatwa dla ogromnej większości informacji, która podlega integracji – to naturalnym kandydatem na narzędzie integrujące jest język zapytań XQuery.

### 2.3.1. Cechy XQuery jako narzędzia integracji danych

Zastosowanie XQuery do integracji danych z wielu heterogenicznych źródeł jest rozwiązaniem pozwalającym uzyskać pożądane cechy:

- integrację źródeł danych w czasie rzeczywistym, bez konieczności replikowania informacji (przy założeniu że – jak to opisano dalej – dane źródłowe są dostępne *on-line*);
- efektywność, rozumianą jako dobry stosunek wydajności i jakości wyniku w stosunku do niezbędnych nakładów;
- elastyczność – dzięki deklaratywności języka zapytań mechanizmy integrujące dane są łatwe do rozszerzania oraz adaptacji do zmian danych źródłowych.

### 2.3.2. Architektura systemu integrującego dane

Jak się wydaje, najwłaściwszą lokacją dla oprogramowania integrującego dane jest warstwa pośrednia (*middle-tier*), typowo występująca we współczesnych wielowarstwowych systemach informacyjnych. Oprogramowanie działające w tej warstwie czerpać może informacje zarówno wprost z baz danych jak i z Sieci. Wyniki integracji udostępniać zaś może przez typowe protokoły wprost klientom (np. przeglądarkom HTML/XML) lub przez typowe interfejsy innym aplikacjom lokalnym lub odległym. Współdziałanie w systemie rozproszonym może odbywać się w typowych technologiach rozproszonych EJB lub CORBA albo z wykorzystaniem technologii usług sieciowych (*Web Services*).

Oczywiście narzędzie integrujące musi nie tylko wykonywać zapytania XQuery, ale także uzyskiwać dostęp do różnorodnych źródeł danych i umieć transformować je na postać XML. Nie ma bowiem większego sensu przekształcanie danych pozyskiwanych na bieżąco, np. z baz danych, na fizycznie istniejące dokumenty XML; należy raczej stworzyć wirtualne widoki XML-owe, które prezentują różnorodne dane wejściowe w postaci XML.

### 3. Technologia *XML Data Synthesis*

*Oracle XML Data Synthesis* (XDS) jest właśnie propozycją takiego narzędzia integrującego, które potrafi dostarczyć różnorodnego pochodzenia dane w formie XML i integrować je za pomocą XQuery.

Technologia ta obecnie jest dostępna w wersji testowej (*developers preview*), ale wkrótce ma stać się częścią serwera aplikacyjnego *Oracle Application Server*.

#### 3.1. Źródła danych dla XDS

Źródła danych dla XDS mogą być różnorodne:

- dokumenty XML;
- dokumenty w formatach różnych od XML, np. arkusze kalkulacyjne, opisane za pomocą języka D3L (patrz niżej) i transformowane na bieżąco przez XDS na postać XML;
- dane w relacyjnych bazach danych dostępne przez JDBC;
- dane zwracane przez usługi sieciowe (*Web Services*);
- dane z aplikacji typu EIS, dostępne przez odpowiedni adapter JCA (*J2EE Connector Architecture*).

Wszystkie te źródła są dostępne dla zapytań XQuery przez zewnętrzne funkcje, zastępujące wbudowaną funkcję `doc()`. Źródła te i odpowiednie funkcje trzeba oczywiście zdefiniować, co robi się deklaratywnie, tworząc odpowiednie pliki konfiguracyjne XDS.

##### 3.1.1. *Data Definition Description Language* (D3L)

*Data Definition Description Language* jest częścią technologii Oracle o nazwie *Application Server Integration InterConnect*. D3L jest językiem opisu różnorodnych nie-XML-owych źródeł danych, który pozwala określić format takiego źródła i sposób jego prezentacji w XML. Sam język D3L jest także dialektem XML.

Motor D3L wbudowano w XDS; czyta on dane źródłowe i udostępnia je jako źródło XML-owe dla XQuery. Sposób transformacji opisuje się w dokumencie w języku D3L; do adresu URL tego dokumentu odwołuje się definicja źródła danych w pliku konfiguracyjnym XDS.

#### 3.2. XQuery w XDS

XDS używa standardowych mechanizmów języka XQuery do podłączenia funkcji udostępniających źródła danych. Funkcje te definiuje się w plikach konfiguracyjnych XDS (patrz niżej); każda z nazw tych funkcji musi być umieszczona w jakiejś przestrzeni nazw. Można tu użyć przestrzeni nazw zdefiniowanej przez użytkownika lub skorzystać z domyślnej przestrzeni `http://xmlns.oracle.com/ias/xds`. W obu wypadkach użytą przestrzeń trzeba zadeklarować w prologu zapytania XQuery za pomocą deklaracji **declare namespace**. Zadeklarować tam trzeba także zdefiniowane w XDS funkcje-źródła jako funkcje zewnętrzne (**declare function ... external**). Resztę zapytania definiuje się już „zwyczajnie” – w całkowicie typowy dla XQuery sposób.

Udostępniane przez XDS interfejsy programistyczne pozwalają wykonywać zapytania XQuery podawane na bieżąco (jako napisy przekazywane w parametrach odpowiednich metod) lub odwoływać się do widoków, czyli przygotowanych wcześniej zapytań XQuery, zapisanych w odpowiednim katalogu („repozytorium”) struktury definiującej aplikację XDS. Wynik zapyta-

nia ma postać sekwencji; odpowiedni typ **OXMLSequence** zdefiniowano w API XDS. W zależności od użytego interfejsu można go pobrać w całości lub element po elemencie.

### 3.3. Konfigurowanie XDS

Konfiguracja XDS sprowadza się w istocie do zdefiniowania źródeł danych i ewentualnego zapisania zapytań XQuery mających pełnić funkcję widoków.

Źródła danych dla XQuery definiuje się podając odpowiednie deklaracje w pliku konfiguracyjnym o nazwie **xds-config.xml**.

Każde ze źródeł ma nazwę, odpowiadającą nazwie funkcji przez którą będzie dostępne w XQuery. Dla źródła można także podać schemat *XML Schema*, który powinien być użyty do walidacji i typowania danych oraz optymalizacji zapytań<sup>1</sup>.

Źródła mogą mieć parametry sterujące sposobem pozyskania informacji; odpowiadają one parametrom funkcji-źródła danych w zapytaniu XQuery.

#### 3.3.1. Typy definicji źródeł

Z punktu widzenia sposobu definiowania, źródła danych dla XDS dzielą się na trzy typy:

- źródła o charakterze dokumentów,
- źródła opisywane plikami WSDL,
- widoki XQuery.

#### 3.3.2. Dokumenty jako źródła dla XDS

Definiowane za pomocą elementu **<document-source>** źródła o charakterze dokumentów są dostępne dla zapytań XQuery w sposób podobny do działania wbudowanej funkcji **doc()**; jednak z pewnymi przewagami w stosunku do niej:

- dokumenty mogą być buforowane;
- URL dokumentu może być na stałe zapisany w pliku konfiguracyjnym i nie musi być powtarzany w zapytaniach XQuery;
- dokumenty źródłowe mogą nie być w formacie XML, lecz być na bieżąco konwertowane na XML przez wbudowany w XDS motor D3L, wspomniany wcześniej.

Definicja źródła typu dokument zawierać może URL do tego dokumentu oraz odwołanie do pliku D3L sterującego transformacją. W przyszłych wersjach XDS należy się spodziewać udostępnienia także innych metod transformacji dokumentów źródłowych.

#### 3.3.3. Usługi sieciowe

Dane do integracji mogą być na bieżąco pozyskiwane jako wyniki wywołania usług sieciowych (*Web Services*). Usługi te są definiowane przez podanie odpowiednich plików w języku WSDL (*Web Services Definition Language*). Definicja źródła danych, zawarta w elemencie **<wsdl-source>**, określa adres URL pliku WSDL opisującego usługę, nazwy operacji, usługi i portu pozwalające zidentyfikować pożądaną akcję oraz deklaracje parametrów przekazywanych do usługi, a także określenie typu odpowiedzi i odwzorowania go na klasę języka Java.

---

<sup>1</sup> W wersji *preview* te funkcje nie są zaimplementowane.

### 3.3.4. Dostęp do danych relacyjnych

W XDS wbudowano usługę sieciową udostępniającą zawartość relacyjnych baz danych, dostępnych przez protokół JDBC. Takie źródła danych deklaruje się w pliku **xds-config.xml** podobnie jak „prawdziwe” usługi sieciowe. Natomiast definiujący konkretną usługę plik WSDL zawiera dane połączenia z bazą, treść zapytań SQL oraz określenie sposobu wiązania zmiennych związanych.

Wynik zapytania SQL jest zwracany w postaci sekwencji elementów **<ROW>**, odpowiadających wierszom, zawierających elementy o nazwach zgodnych z nazwami wynikowych kolumn; do odwzorowania wyniku zapytania SQL na strukturę XML użyto w XDS znanej biblioteki XSU (*Oracle XML SQL Utility*), stanowiącej część pakietu Oracle XDK.

### 3.3.5. WSIF

Definiując w XDS usługi sieciowe za pomocą języka WSDL można uzyskać dostęp nie tylko do „prawdziwych” usług sieciowych, czyli tych dostępnych przez protokół SOAP. Szersze możliwości uzyskano wbudowując w XDS interfejs WSIF (*Web Services Invocation Framework*), technologię z „rodziny” Apache [Ap05], pozwalającą za pomocą WDSL definiować usługi sieciowe inne niż typu SOAP.

W XDS wykorzystano WSIF do udostępniania jako źródeł danych aplikacji dostępnych za pośrednictwem klas języka Java, komponentów EJB lub przez adaptery JCA (*J2EE Connector Architecture*).

### 3.3.6. Widoki XQuery

Widoki są to wcześniej przygotowane i zapamiętane w plikach o rozszerzeniu **.xq** zapytania XQuery, operujące na własnych źródłach danych, które mogą być z kolei wykorzystane jako źródła w innych zapytaniach. Widoki te mogą być sparametryzowane za pomocą zmiennych XQuery zadeklarowanych jako zmienne zewnętrzne za pomocą deklaracji **declare variable ... external**. W czasie wykonania zapytania kolejne parametry podane funkcji udostępniającej źródło danych są przypisywane do kolejnych zmiennych zewnętrznych widoku.

Definicja tego typu źródła danych, zawarta w elemencie **<xdsview-source>**, określa nazwę pliku z zapytaniem i wskazanie na położenie katalogu – „repozytorium” takich plików; może też zawierać deklaracje parametrów widoku. Można także zadeklarować, że widok XDS będzie sam dostępny jako usługa sieciowa.

### 3.3.7. Buforowanie danych w XDS

Dla każdego ze źródeł danych określić można sposób buforowania danych przez XDS; odpowiednio dobierając buforowanie uzyskać można lepszą wydajność zapytań. Możliwe jest buforowanie nietrwałe, tj. ograniczone tylko do czasu trwania procesu OC4J, lub trwałe. Parametry buforowania ustala się w pliku konfiguracyjnym **xds-config.xml**.

## 3.4. Architektura i interfejsy XDS

XDS jest narzędziem napisanym w języku Java i działa jako aplikacja serwera OC4J (*Oracle Components for Java*). Dla użytkownika dostępne są następujące interfejsy narzędzia:

- API klienta w języku Java;
- komponenty EJB typu sesyjnego: bezstanowy i zachowujący stan (ten ostatni umożliwia pobieranie wyniku zapytania w wielu krokach);
- biblioteka znaczników JSP.

Widoki XQuery mogą także być udostępniane jako usługi sieciowe (*Web Services*).

## 4. Podsumowanie

Integracja danych z różnych źródeł – wewnętrznych i zewnętrznych – jest coraz częściej niezbędna do sprawnego funkcjonowania organizacji i przedsiębiorstw. Większość informacji, która podlegać może takiej integracji, da się łatwo przedstawić w postaci XML.

Jeśli dysponujemy XML-ową reprezentacją informacji źródłowej, idealnym narzędziem do wyszukiwania i scalania danych jest język zapytań XQuery. Pozwala on łączyć informację z różnych źródeł łatwo i w sposób elastyczny. Aby jednak możliwe było tworzenie działających na bieżąco serwisów integrujących dane z heterogenicznych źródeł, konieczne jest jeszcze zapewnienie mechanizmów pozwalających *on-line* udostępniać informacje o różnorodnym pochodzeniu i różnych formatach w postaci wirtualnych dokumentów XML-owych. Narzędziem zapewniającym taką funkcjonalność jest właśnie *Oracle XML Data Synthesis*.

Zastosowanie XML i XQuery do integracji danych oraz umiejscowienie serwisów scalających w warstwie pośredniej (*middle tier*) systemu informacyjnego wydaje się rozwiązaniem bardzo obiecującym, mogącym w sposób mało kosztowny i elastyczny rozwiązywać większość potrzeb dotyczących integracji informacji.

## Bibliografia

- [Tra04] Traczyk T.: Język XML Query. Materiały X Konferencji użytkowników i deweloperów Oracle. Kościelisko, październik 2004.
- [Tra03] Traczyk T.: XML – stan obecny i trendy rozwojowe. Materiały IX Konferencji użytkowników i deweloperów Oracle. Kościelisko, październik 2003.
- [KV05] Kvitka C.: XQuery: A New Way to Search. Oracle Magazine, January/February 2005, pp. 51-52.
- [JuChB05] Junnarkar N., Chanchani N., Basu J.: Aggregate Data with XQuery. Oracle Magazine, March/April 2005, pp. 45-48.
- [Gen05] Gennick J.: XQuery Flowers. Oracle Magazine, September/October 2005, pp. 65-68.
- [BaCh04] Basu J., Chanchani N.: Information Integration Within the Enterprise Using XML. XML Journal, May 2004. <http://xml.sys-con.com/read/44676.htm>
- [W3q05] World Wide Web Consortium: XQuery 1.0: An XML Query Language. W3C Working Draft, 15 September 2005. <http://www.w3.org/TR/2005/WD-xquery-20050915/>
- [W3p05] World Wide Web Consortium: XML Path Language (XPath) 2.0. W3C Working Draft, 15 September 2005. <http://www.w3.org/TR/2005/WD-xpath20-20050915/>
- [Ap05] Welcome to WSIF: Web Services Invocation Framework. <http://ws.apache.org/wsif/>