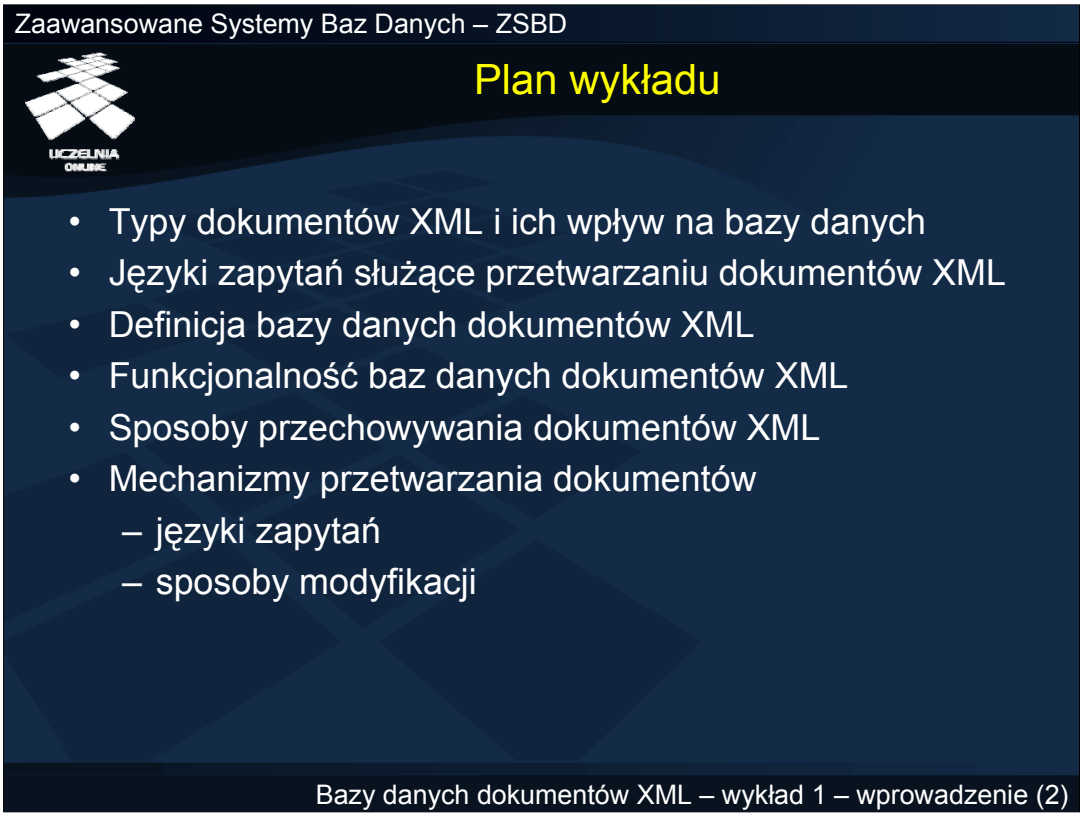
Przez ostatnich kilkanaście lat znaczenie formatu danych XML stale rośnie. Popularność XML wynika z jego prostoty i elastyczności. Te cechy spowodowały, że XML stał się popularnym i dominującym standardem wymiany danych o złożonej, zmiennej i nieokreślonej strukturze. Dzięki tym cechom XML jest z powodzeniem wykorzystywany w takich dziedzinach jak: nauka, finanse, wymiana informacji, medycyna, grafika, kartografia, multimedia itp.

Początkowo dokumenty XML przechowywanie były przy wykorzystaniu systemów plików. Rosnące znaczenie i popularność XML-a a także stale rosnąca liczba dokumentów XML i konieczność ich wydajnego przetwarzania spowodowały, że w ostatnim czasie bardzo wiele komercyjnych systemów zarządzania obiektowymi i postrelacyjnymi systemami baz danych rozszerzyło swoją funkcjonalność o mechanizmy pozwalające na przechowywanie i przetwarzanie dokumentów XML. Takie rozwiązania pozwalają w znakomity sposób integrować dane mające różny charakter, z jednej strony strukturalny (dane obiektowe i relacyjne) z drugiej strony semistrukturalne (dane w formacie XML). Oprócz adaptacji systemów obiektowych i posterelacyjnych obserwuje się także powstawanie licznych specjalizowanych rozwiązań pod postacią baz danych dokumentów XML (ang. native XML database systems). W tego typu bazach danych podstawową jednostką informacji jest dokument XML, użytkownicy przechowują w nich swoje dokumenty XML, przeglądają ich zawartość, generują na podstawie zawartości bazy danych nowe dokumenty XML.

Z punktu widzenia klasyfikacji, bazy danych dokumentów XML są przykładem semistrukturalnych baz danych.



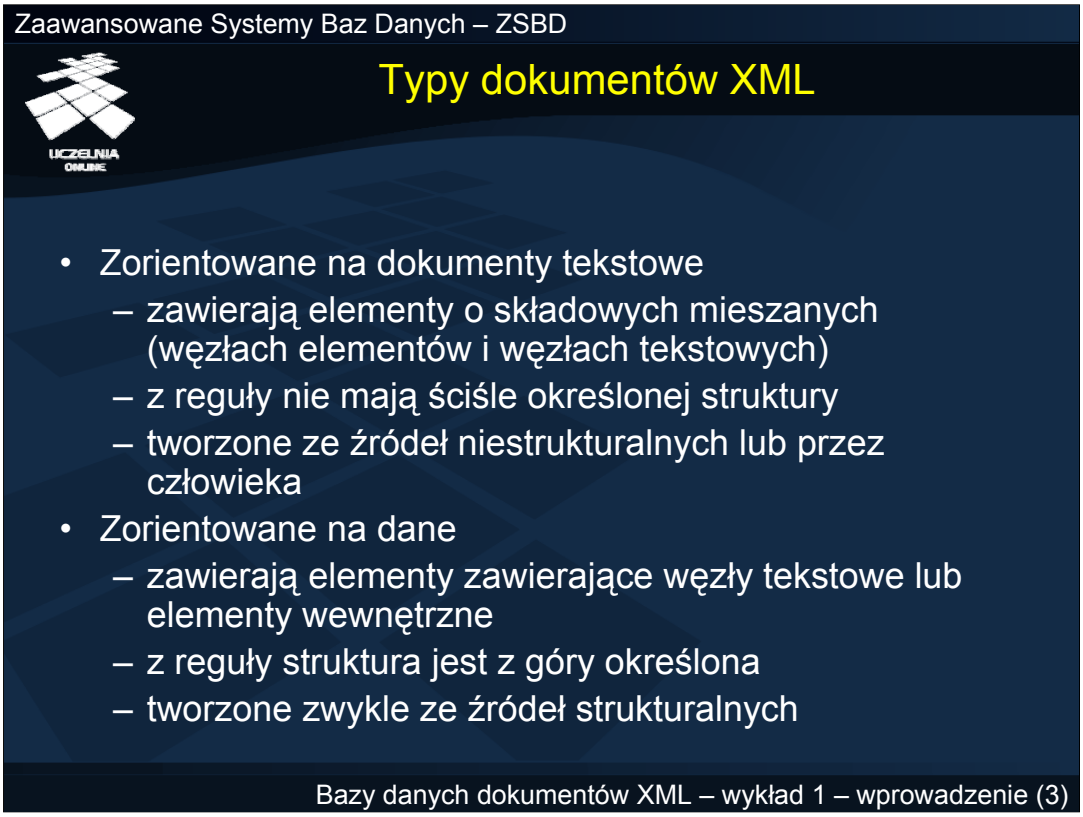
Informacje dotyczące XML-owych baz danych zostaną przedstawione w trzech kolejnych wykładach.

Wykład pierwszy będzie wprowadzeniem do tematyki XML-owych baz danych. Zostanie przedstawiona definicja, funkcjonalność oraz typy XML-owych baz danych. Dodatkowo zostaną omówione te cechy XML-owych baz danych, które wyróżniają je na tle systemów relacyjnych i obiektowych.

Zadaniem wykładu drugiego będzie przedstawienie języka XQuery jako jednego z najbardziej zaawansowanych i najbardziej popularnego z języków zapytań stosowanych w XML-owych baz danych. Omówiona zostanie jego składania oraz funkcjonalność. Wykład zostanie zilustrowany szeregiem przykładów i porównań.

Wykład trzeci głównie będzie dotyczył języków przeznaczonych do modyfikacji dokumentów XML w XML-owych bazach danych. Ze względu na to, że nie ma obecnie standardu takiego typu języka, zostanie przedstawionych kilka podstawowych propozycji stosowanych w różnych bazach danych dokumentów XML.

W dzisiejszym wykładzie omówimy typy dokumentów XML i ich wpływ na bazy danych. Omówimy języki zapytań służące przetwarzaniu dokumentów XML. Zostanie przedstawiona jedna z możliwych definicji baz danych dokumentów XML. W dalszej części wykładu zostanie mówiona funkcjonalność baz danych dokumentów XML, sposoby przechowywania i mechanizmy przetwarzania dokumentów XML.

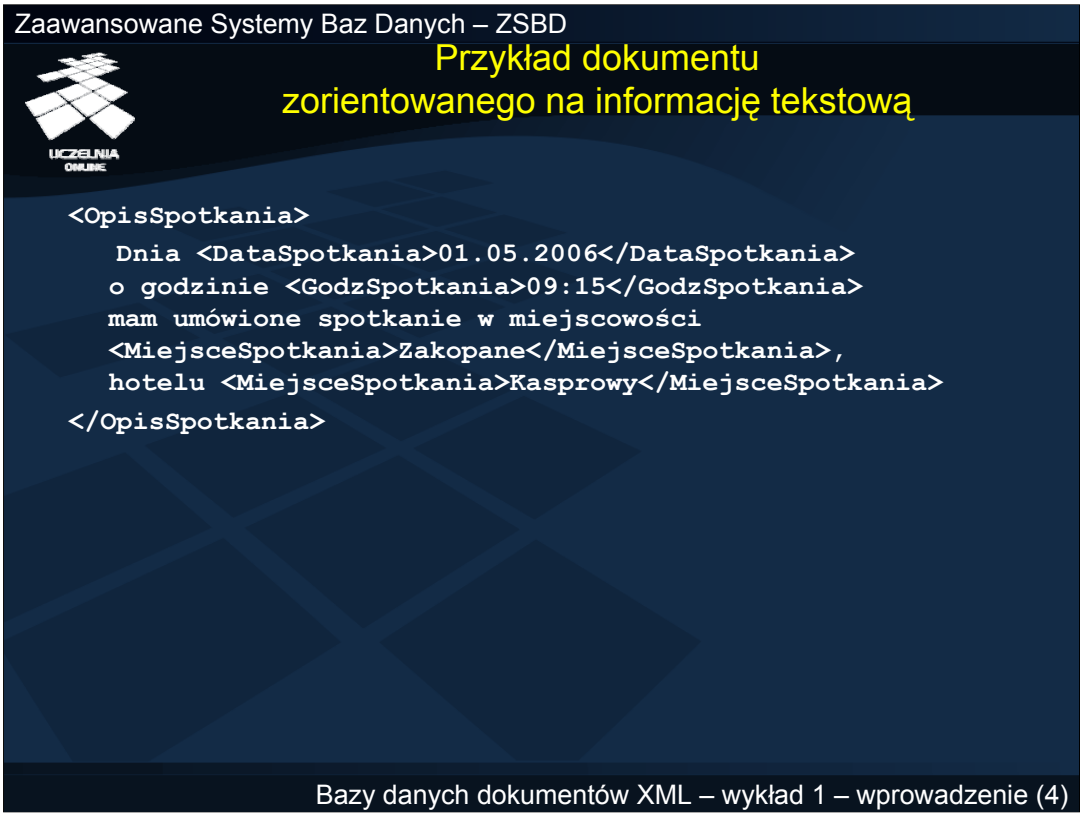


Typ, budowa i funkcjonalność bazy danych dokumentów XML uzależniona jest ściśle od typów dokumentów XML jakie będą w niej przechowywane.

W tym kontekście wyróżnia się dwa typy dokumentów XML:

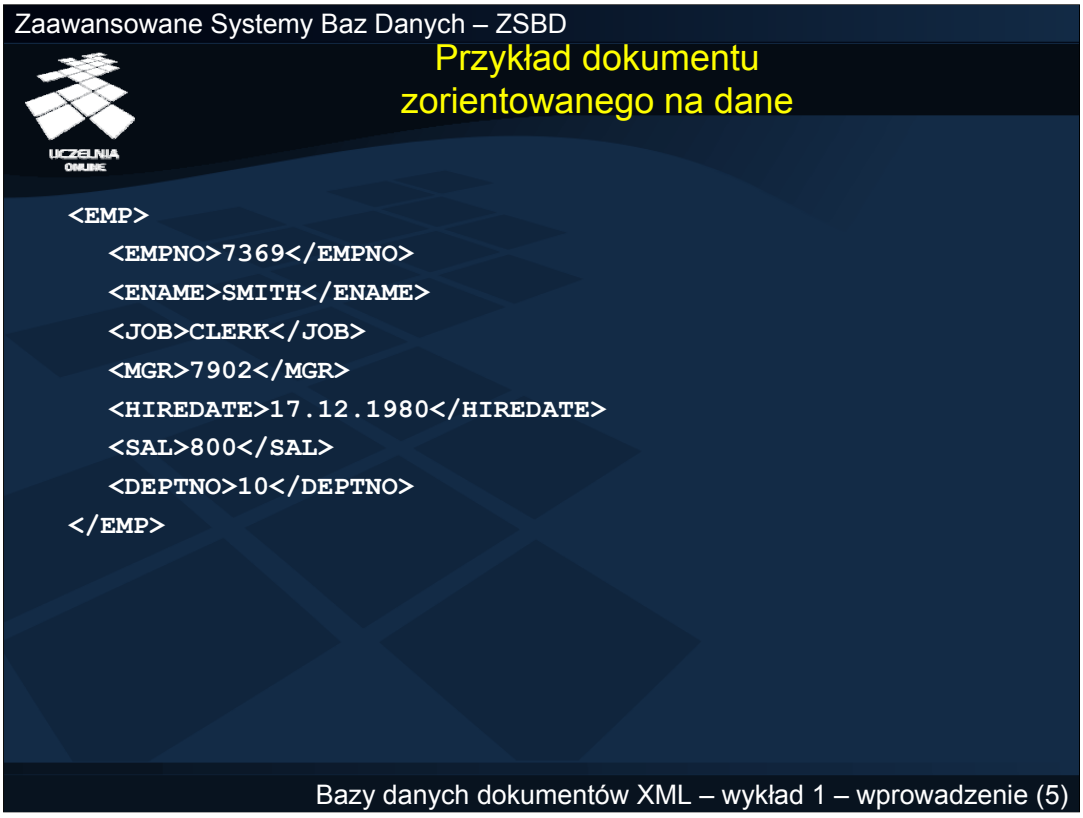
Po pierwsze, dokumenty zorientowane na dokumenty tekstowe. Tego typu dokumenty zwykle zawierają elementy o składowych mieszanych (węzły elementów i węzły tekstowe jednocześnie). Ponadto, tego typu dokumenty, z reguły, nie mają ściśle określonej struktury, nie ma narzuconego schematu, który ogranicza i definiuje strukturę dokumentu. Jest ona indywidualna dla każdego dokumentu. Często dokumenty takie są tworzone ze źródeł niestrukturalnych takich jak dokumenty tekstowe, dokumenty HTML. Mogą też być tworzone w sposób nieautomatyczny przez człowieka.

Drugim typem dokumentów XML są dokumenty zorientowane na dane. Tego typu dokumenty zawierają elementy, których składowe są ściśle określone. Są to albo węzły tekstowe albo elementy wewnętrzne. Z reguły, struktura tego typu dokumentów zorientowana jest na przechowywanie danych, jest ona też z góry określona. Struktura ta wynika najczęściej ze struktury lub możliwości źródła pochodzenia informacji zawartej w dokumencie. Ponadto, tego typu dokumenty są zwykle tworzone ze źródeł strukturalnych takich jak bazy danych obiektowe, relacyjne itp.



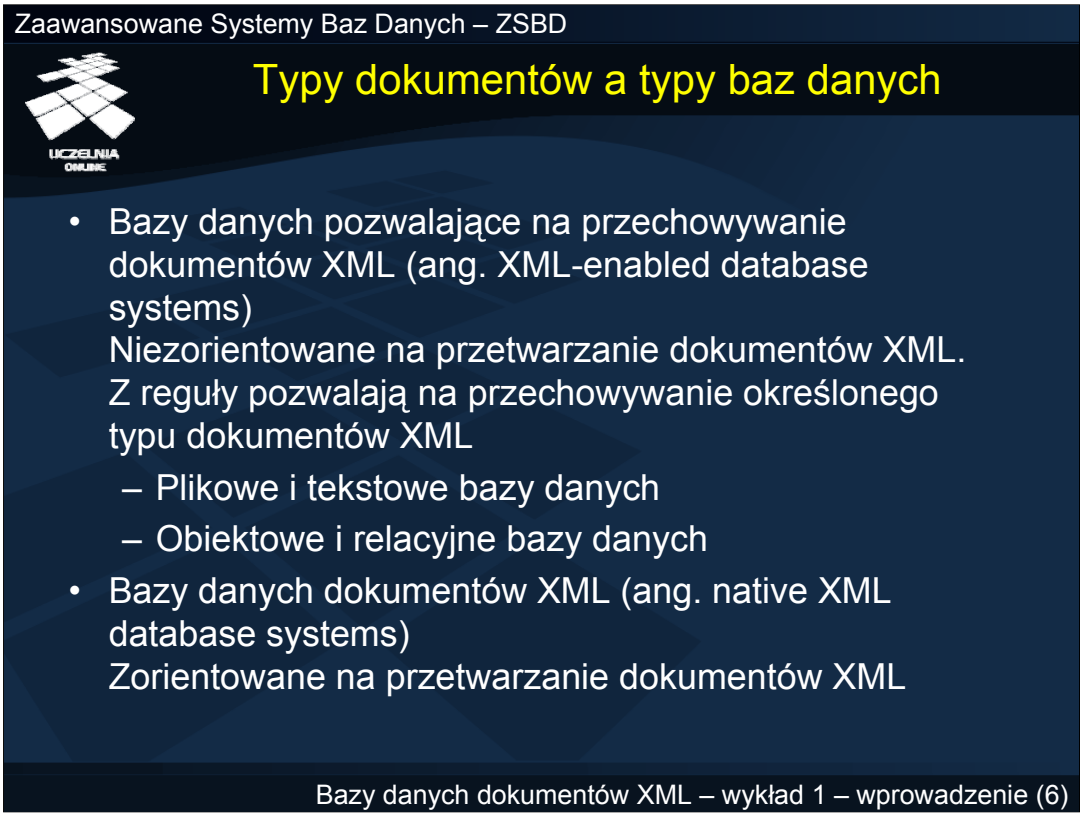
Na slajdzie przedstawiono przykładowy dokument zorientowany na informację tekstową.

Element OpisSpotkania jest typowym przykładem elementu mieszanego, zawierającego w swoim wnętrzu zarówno węzły tekstowe (przykładowo węzeł z tekstem "Dnia ") jak i węzły będące podelementami (przykładowo element DataSpotkania). Można założyć, że inne dokumenty posiadające elementy opisujące spotkanie mogą mieć różną od powyższego przykładu zawartość. Dla przykładu element OpisSpotkania może w ich przypadku mieć tylko jeden podelement MiejsceSpotkanie, może nie mieć podelementów takich jak DataSpotkania lub GodzSpotkania (bo dla przykładu nie są one jeszcze znane). Ponadto element MiejsceSpotkania może być złożony z dodatkowych podelementów, dla przykładu, Miejscowość, Ulica, KodPocztowy itp.



Na tym z kolei slajdzie przedstawiono dokument zorientowany na dane. Informacje w nim zawarte mogą pochodzić z bazy danych relacyjnej lub obiektowej. Dokument ma wyraźną bardzo ściśle określoną budowę.

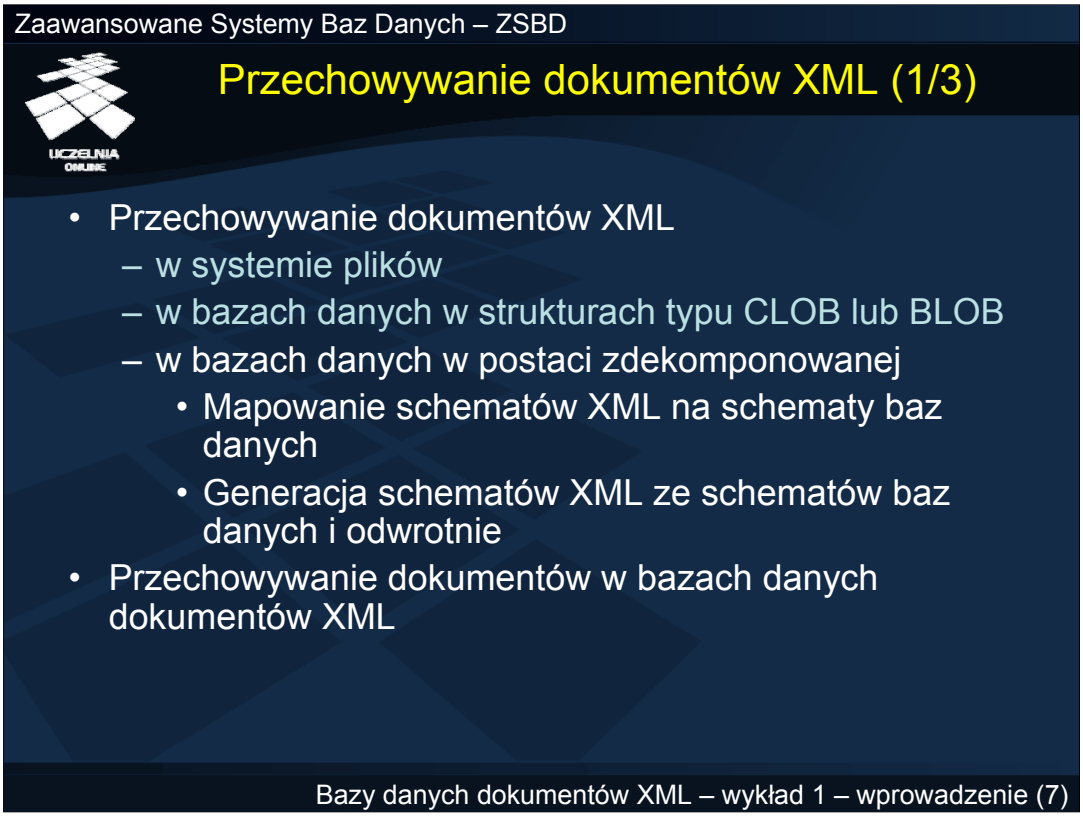
Element ROW składa się tylko z podelementów. Nie ma bezpośrednich żadnych podwęzłów tekstowych. Podelementy elementu ROW, w przypadku dokumentu ze slajdu, są elementami prostymi. Ich zawartość jest prosta, może ona pochodzić z atrybutów tabeli lub obiektu. Można założyć, że budowa innych elementów EMP będzie bardzo podobna, każdy z nich będzie się składał z podelementów prostych wśród, których znajdą się elementy takie jak EMPNO czy ENAME.



Typ wykorzystywanych dokumentów XML bardzo często ma zasadniczy wpływ na wybór bazy danych wykorzystywanej do ich przechowywania. Można wręcz powiedzieć, że możliwość przechowywania określonego typu dokumentów XML definiuje swego rodzaju klasyfikację baz danych wykorzystywanych do przechowywania dokumentów XML.

Z jednej strony mamy bazy danych, obiektowe lub obiektowo-relacyjne, które umożliwiają przechowywanie dokumentów XML. Niestety w większości przypadków jest to możliwe tylko w przypadku dokumentów XML o ściśle określonej budowie. Wynika to faktu, że poszczególne fragmenty dokumentów XML są umieszczane we wcześniej zdefiniowanych tabelach, lub obiektach posiadających ściśle określoną budowę. Jeśli takie ograniczenia występują mówimy o bazach danych pozwalających na przechowywanie dokumentów XML (ang. XML-enabled databases).

Z drugiej strony, mamy do czynienia z bazami danych, które pozwalają na przechowywanie dowolnych dokumentów XML, niezależnie od tego czy posiadają one określoną strukturę czy też nie. Jeżeli dodatkowo dokument XML-owy dla użytkownika stanowi podstawowy element przechowywany w bazie danych, i jego przetwarzanie jest realizowane głównie w oparciu o standardy związane z XML, wówczas możemy mówić o bazach danych dokumentów XML (ang. native XML database systems).



Istnieje kilka podstawowych sposobów przechowywania dokumentów XML.

Po pierwsze przy wykorzystaniu systemów plików. Składowanie dokumentów XML w systemach plików może być z powodzeniem wykorzystane w przypadkach małych zbiorów dokumentów XML. W przypadkach, gdy konieczne jest proste zarządzanie transakcjami można wykorzystać narzędzia, jakie jak CVS.

Nieco bardziej zaawansowanym sposobem jest wykorzystanie baz danych do przechowywania dokumentów XML w strukturach takich jak duże obiekty tekstowe lub binarne. Zalety takiego rozwiązania wywodzą się z funkcjonalności systemów baz danych takich jak: wielodostęp, obsługa transakcji, autoryzacja użytkowników itp. Kolejną zaletą jest możliwość wykorzystania indeksów do przeszukiwania dużych obiektów tekstowych.

Podstawowymi wadami obu podejść są:

- konieczność przetwarzania niestrukturalnego,

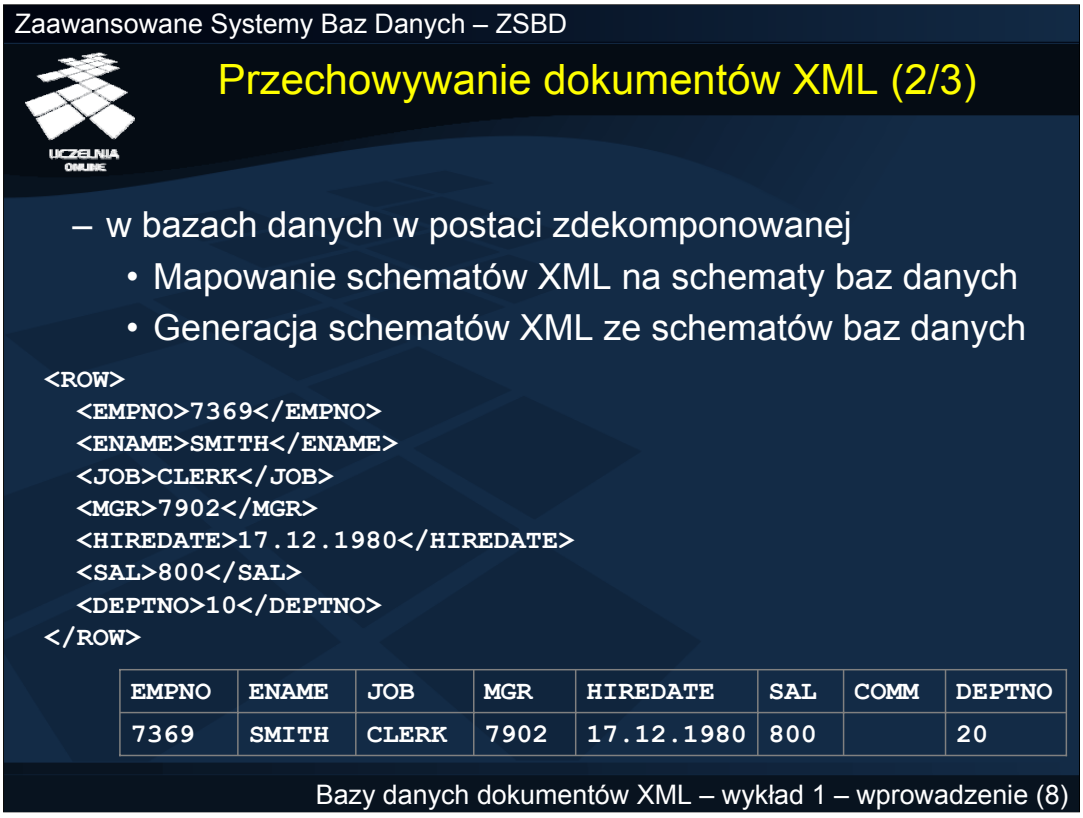
- brak rozróżnienia znaczników i ich zawartości.

Zaletą jest natomiast możliwość przechowywania dokumentów XML o dowolnej, nieokreślonej z góry strukturze.

Ponadto dokumenty XML możemy przechowywać:

- wykorzystując tabele lub obiekty składowane w obiektowo-relacyjnych bazach danych do przechowywania zdekomponowanych na składowe proste dokumentów XML, lub

- wykorzystując bazy danych dokumentów XML.



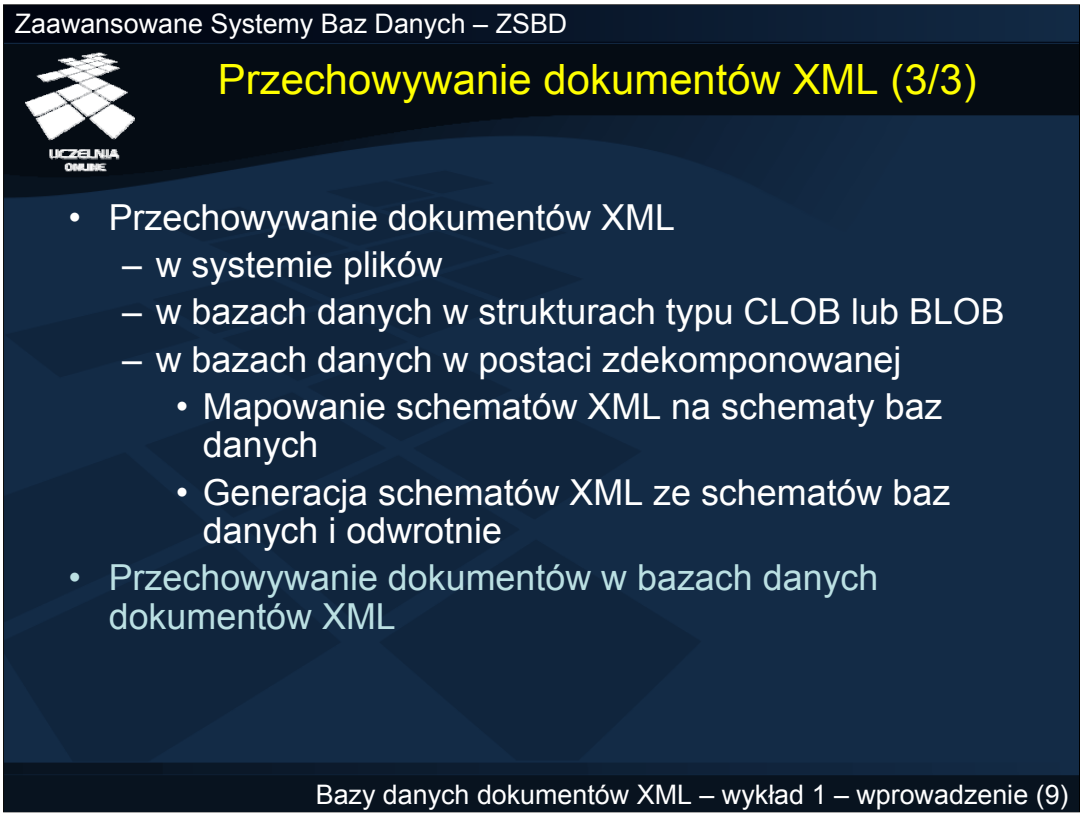
W przypadku wykorzystywania obiektowych lub relacyjnych baz danych, można przechowywać dokumenty XML w postaci zdekomponowanej. Dekompozycja polega na podziale dokumentu XML na mniejsze fragmenty, i przechowywanie ich w schemacie relacyjnym i obiektowych pozwalającym na późniejsze odtworzenie oryginalnej postaci dokumentu XML. Aby dekompozycja mogła być zrealizowana konieczna jest wcześniejsza znajomość struktury dokumentów XML i stworzenie schematów relacyjnych lub obiektowych zgodnych z tą strukturą. Przykładem dekompozycji dokumentu XML przedstawionego na slajdzie może być zawartość tabeli EMP w postaci również przedstawionej na slajdzie.

Dzięki elastyczności dokumentów XML możliwe jest mapowanie zarówno baz danych relacyjnych jak i obiektowych na postać dokumentów XML. Mapowanie takie jest stosunkowo powszechne i daje możliwość udostępniania zawartości baz danych w postaci dokumentów XML. Przykładem standardu, który zajmuje się między innymi regułami dotyczącymi mapowania baz danych relacyjnych na dokumenty jest SQL/XML.

Mapowanie odwrotne, tzn. dokumentów XML na postać relacyjną lub obiektową, jest zagadnieniem nieco bardziej złożonym aczkolwiek i w tym przypadku mamy do dyspozycji konkretne rozwiązania, które wykorzystywane są dla przykładu przy budowie generatorów schematów. Mapowanie dokumentów XML na struktury obiektowe określane jest wiązaniem danych XML (ang. *XML data binding).* Mapowanie dokumentów XML na zawartość relacji może mieć różną postać:

1. Mapowanie strukturalne – możliwe jest tylko w przypadku znajomości struktury dokumentów XML. Istniejący schemat w bazie danych pozwala na przechowywanie tylko dokumentów zgodnych z tą strukturą. Przykładami takiego mapowania są przykładowo podejścia STORED czy XORator.

2. Mapowanie niestrukturalne – w tym przypadku nie jest konieczna znajomość struktury dokumentu ani jego schematu. Istniejący schemat w bazie danych pozwala na przechowywanie dokumentów o zróżnicowanej strukturze wewnętrznej. Propozycjami takiego mapowania dokumentów XML na zawartość tabel są przykładowo podejścia: Edge, XParent czy XRel.



Składowanie dokumentów XML w postaci zdekomponowanej nie jest pozbawione wad. W dalszym ciągu:

- przetwarzamy dokumenty XML w sposób niezorientowany na XML,

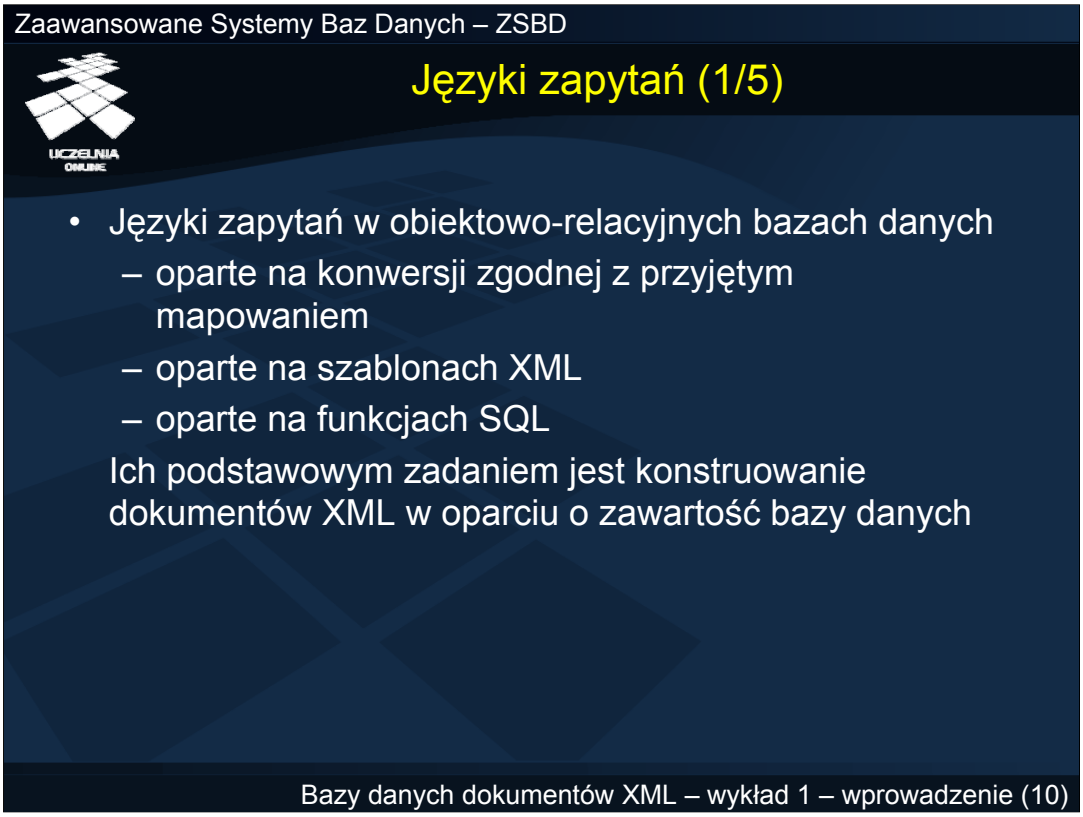
- dużym kosztem obarczona jest operacja odtworzenia dokumentu XML do jego oryginalnej postaci.

Z drugiej jednak strony:

- przetwarzanie jest w pełni strukturalne, co pozwala na wykorzystanie takich mechanizmów jak indeksy,

- występuje rozróżnienie pomiędzy schematem i strukturą dokumentu a jego zawartością.

Wykorzystanie specjalizowanych baz danych dokumentów XML stanowi rozwiązanie w większości przypadków pozbawione powyżej wymienianych wad.



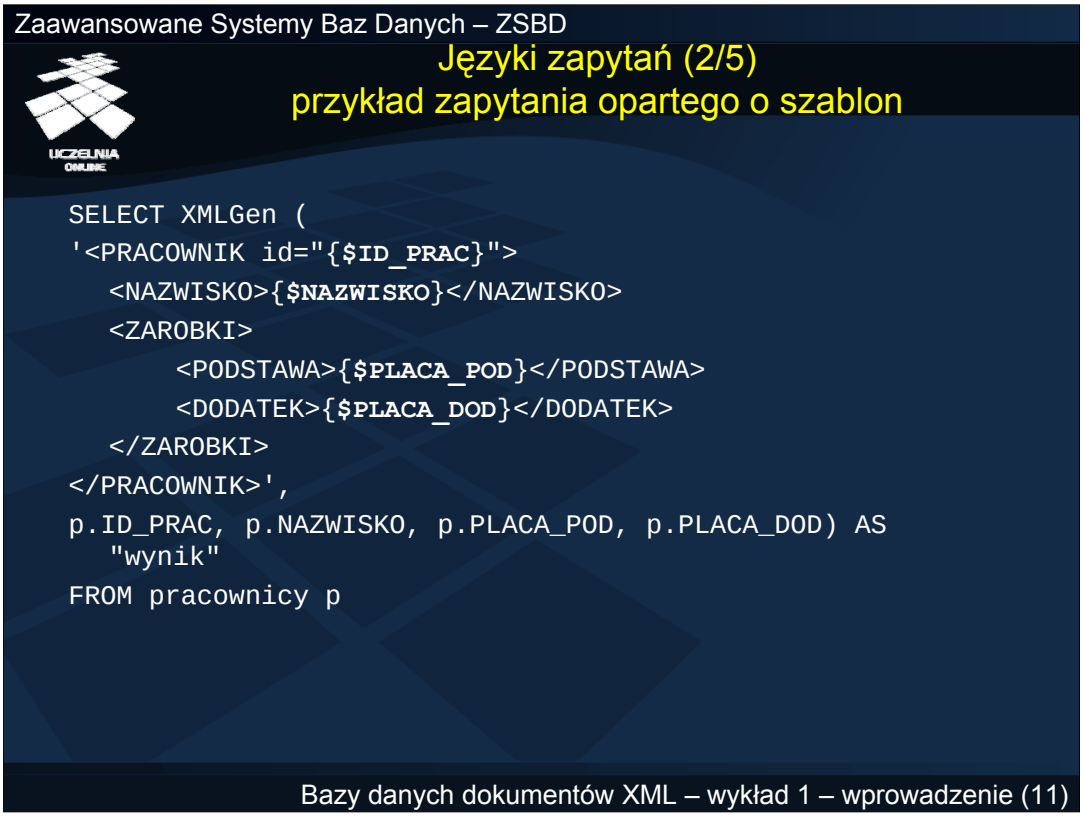
Istnieje wiele rozwiązań związanych z przeszukiwaniem, przetwarzaniem lub konstruowaniem dokumentów XML opartych na językach zapytań. Podstawowy podział pomiędzy tymi rozwiązaniami rysuje się na linii baz danych obiektowo-relacyjnych i baz danych dokumentów XML.

W pierwszym przypadku, mamy do czynienia z językami zapytań obiektowymi lub SQL-owymi, których podstawowym zadaniem jest konstruowanie dokumentów XML w oparciu o zawartość relacyjną lub obiektową baz danych. W najprostszym przypadku dokumenty XML, które powstają w wyniku zapytań są prostym mapowaniem zawartości tabel lub obiektów na ich XML-owe reprezentacje.

Bardziej zaawansowane rozwiązania są oparte o

- szablony,

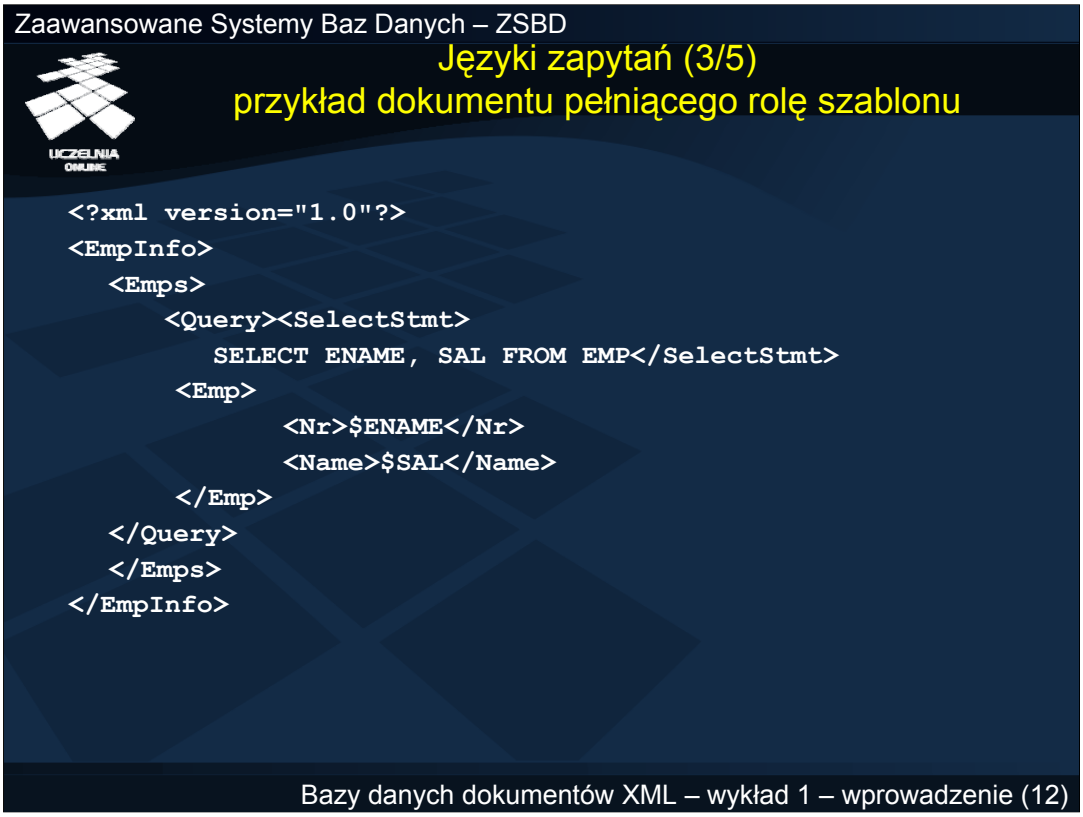
- funkcje SQL.



Zastosowanie szablonów do konstruowania dokumentów XML daje możliwość umieszczenia wyników zapytań SQL w szablonach dokumentów XML. Szablony te posiadając w odpowiednich miejscach zmienne, wymieniają je na wartości kolumn pochodzące z zapytań SQL. Przykładem takiego rozwiązania jest funkcja XMLGen standardu SQL/XML.

Na slajdzie przedstawiono przykład zapytania SQL wykorzystującego funkcję XMLGen opartą na szablonie w celu wygenerowania dokumentów XML.

W wyniku zapytania powstaną elementy PRACOWNIK posiadające atrybut id oraz podelementy NAZWISKO i ZAROBKI. Element ZAROBKI będzie elementem złożonym. Elementy te i ich zawartość zostanie wygenerowana w oparciu o zawartość tabeli pracownicy.



Innym podejściem również opartym o szablony są dokumenty XML z zatopionymi w ich wnętrzu poleceniami SQL-owymi. Po ewaluacji zapytań postać wynikowa dokumentu jest uzupełniania o obiekty XML pochodzące z zawartości bazy danych.

Na slajdzie przedstawiono przykładowy dokument, którego zawartość będzie odpowiednio modyfikowana w zależności od wyniku zawartego w nim zapytania opartego na tabeli EMP.

Komercyjnym rozwiązaniem wykorzystującym ten sposób użycia szablonów, będących dokumentami XML jest standard XSQL Pages.

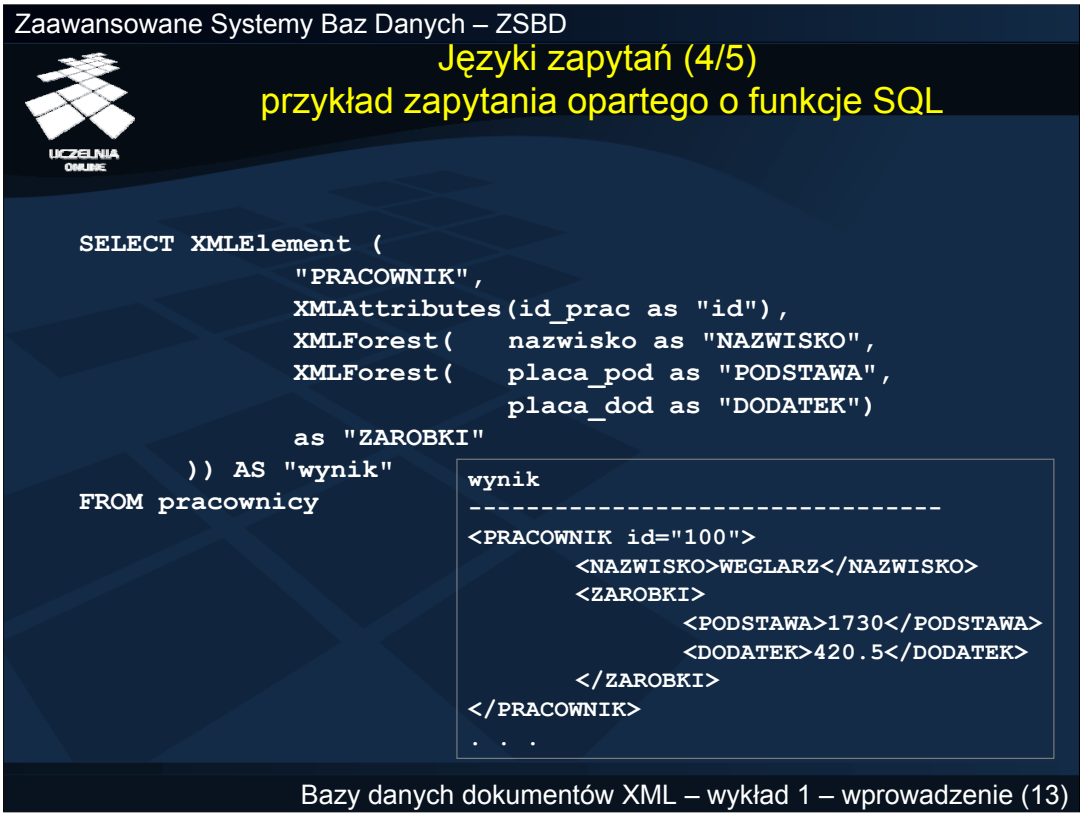
Wykorzystanie szablonów jest bardzo elastyczne. Podstawowa funkcjonalność takich rozwiązań to:

- możliwość umieszczania wyników zapytań w dowolnym miejscu dokumentów posiadających dowolnie złożoną strukturę,

- możliwość wykorzystania konstrukcji programistycznych przypominających pętle oraz instrukcje warunkowe,

- możliwość wykorzystywania zmiennych oraz funkcji SQL,

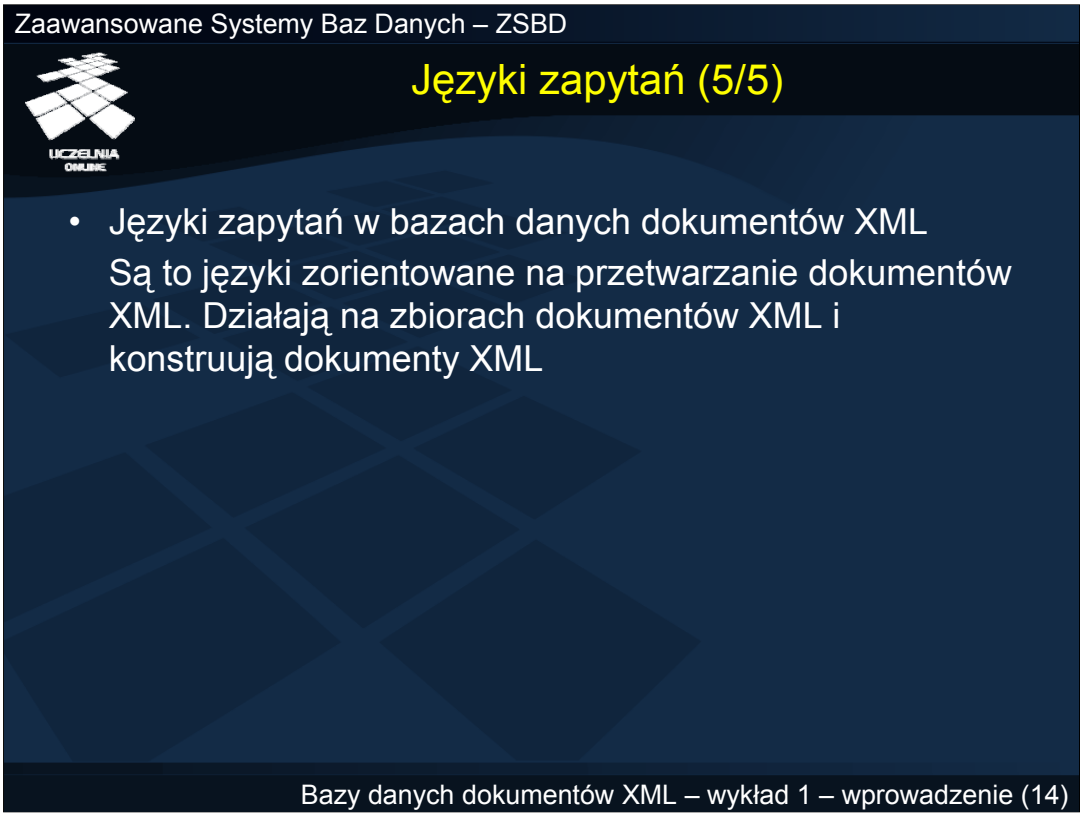
- parametryzacja zapytań SQL w oparciu o parametry HTTP.



Kolejnym sposobem na uzyskanie dokumentów XML w oparciu o zawartość bazy danych jest zastosowanie funkcji SQL. Podstawowym standardem wykorzystującym funkcje SQL do konstruowania dokumentów XML jest standard SQL/XML. Przykładowe zapytanie zostało przestawione na slajdzie.

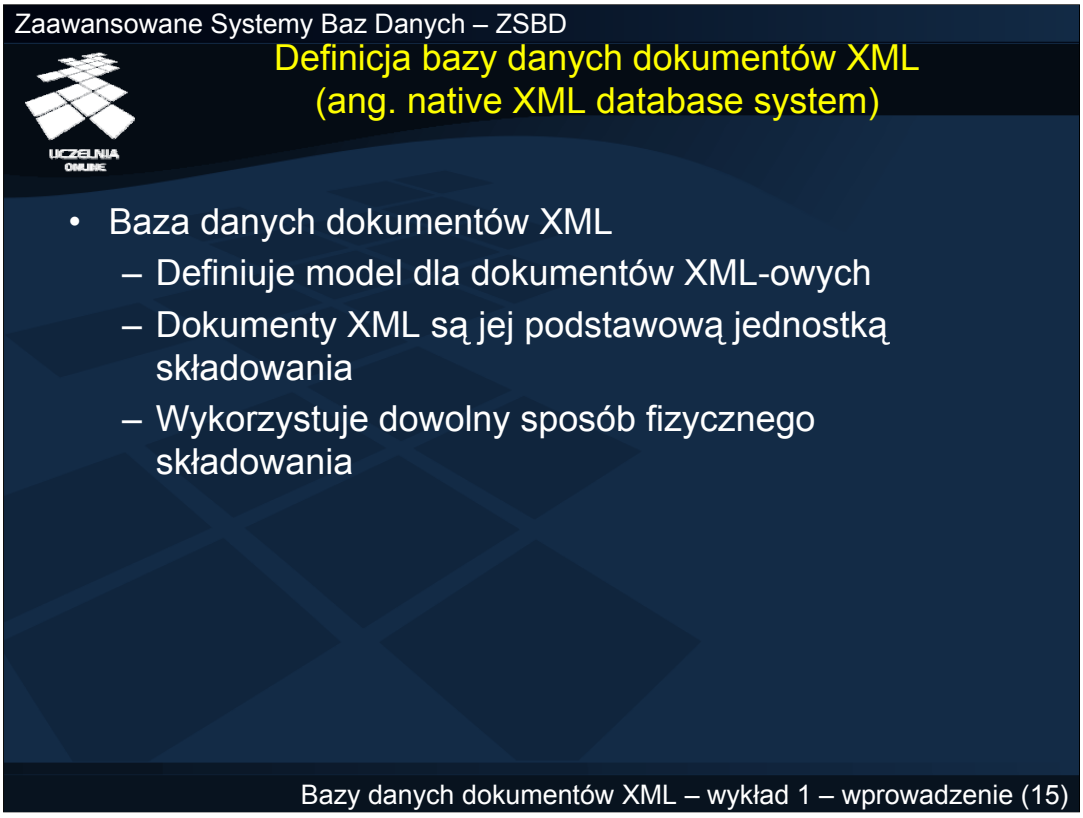
Wynikiem tego zapytania może być zbiór dokumentów w postaci zaprezentowanej na slajdzie wewnątrz ramki.

Standard SQL/XML, którego przykładem jest omawiane zapytanie, stał się w roku 2005 fragmentem standardu SQL. W związku z tym, można oczekiwać, że będzie on, w przyszłości, powszechnie wykorzystywany w relacyjnych bazach danych.



Innym zbiorem języków związanych z przetwarzaniem dokumentów XML w bazach danych są języki zapytań wykorzystywane w bazach danych dokumentów XML. Są to języki zorientowane na przetwarzanie dokumentów XML. Działają one na zbiorach dokumentów XML i konstruują dokumenty XML.

Języki zapytań wykorzystywane w bazach danych dokumentów XML zostaną omówione na późniejszych slajdach.



Przejdźmy teraz do zagadnień związanych bezpośrednio z bazami danych dokumentów XML.

Jedna z możliwych definicji (stworzona przez członków grupy dyskusyjnej XMLDB) mówi, że:

- Baza danych dokumentów XML definiuje model dla dokumentów XML-owych (w przeciwieństwie do danych zawartych wewnątrz dokumentów) i składuje oraz udostępnia dokumenty wg tego modelu.

- Dokumenty XML są podstawową jednostką składowania w bazach danych dokumentów XML, analogicznie do tego jaką jest krotka w bazach danych relacyjnych.

- Nie wymaga się stosowania jakiegoś określonego fizycznego modelu składowania. Bazy danych XML można, zatem, dla przykładu, budować w oparciu o bazy danych relacyjne, obiektowe, hierarchiczne, lub wykorzystywać poindeksowane, skompresowane pliki na poziomie systemu operacyjnego.

XML 1.0 nie definiuje, ani nie narzuca określonego modelu, dlatego XML-owe bazy danych definiują swój własny model. Model danych musi zawierać takie struktury XML jak elementy, atrybuty, tekst i definicję porządku. Przykładami stosowanych modeli mogą być modele wykorzystywane w XPath, DOM, SAX, XQuery. Dane zawarte w bazie danych dokumentów XML udostępniane są wg określonego modelu.

Dokument XML jest podstawową jednostką składowania – jest odpowiednikiem krotki w systemach relacyjnych. Dokument zawiera pojedynczy zbiór danych

Baza danych może wykorzystywać dowolny sposób składowania dokumentów. Dla przykładu mogą to być:

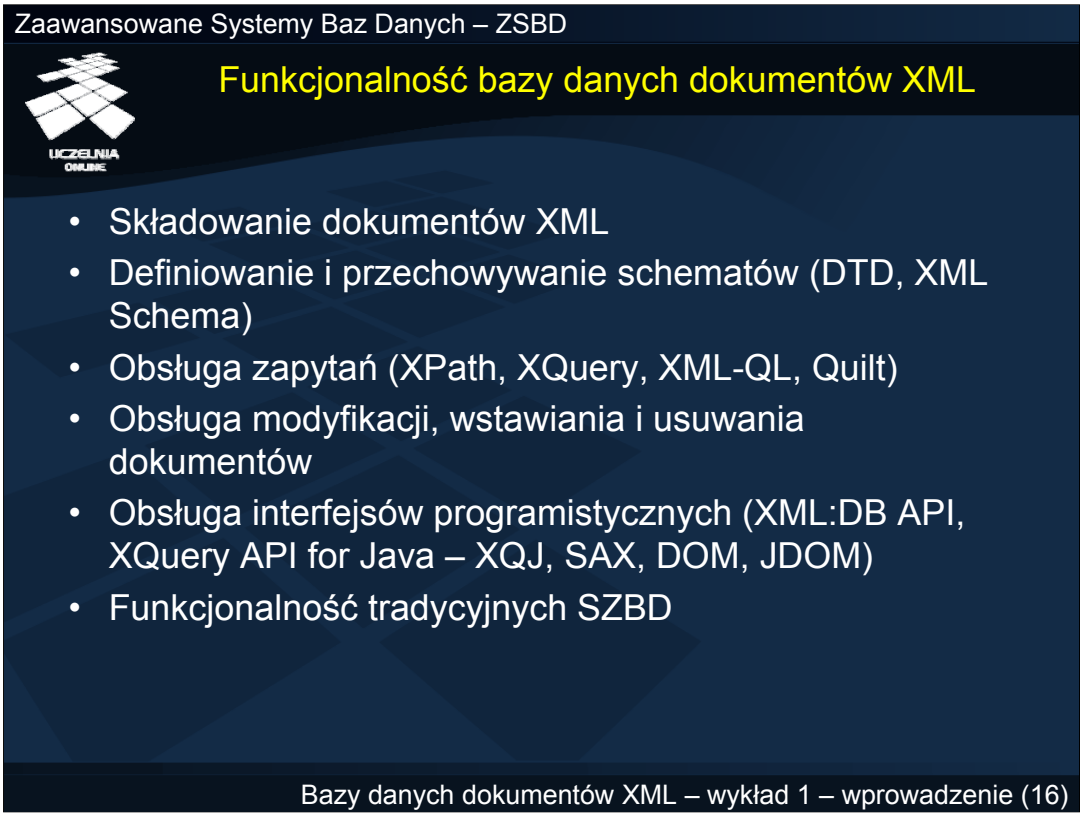
- tabele "obiektów" SAX w bazie danych relacyjnej,

- obiekty DOM w obiektowej bazie danych,

- plik binarny zoptymalizowany do modelu danych XPath,

- skompresowane i poindeksowane dokumenty XML przechowywane w systemie plików.

Stosowany sposób składowania często wpływa na możliwości przechowywania różnych typów dokumentów XML.



Zadaniem baz danych dokumentów XML jest przede wszystkim:

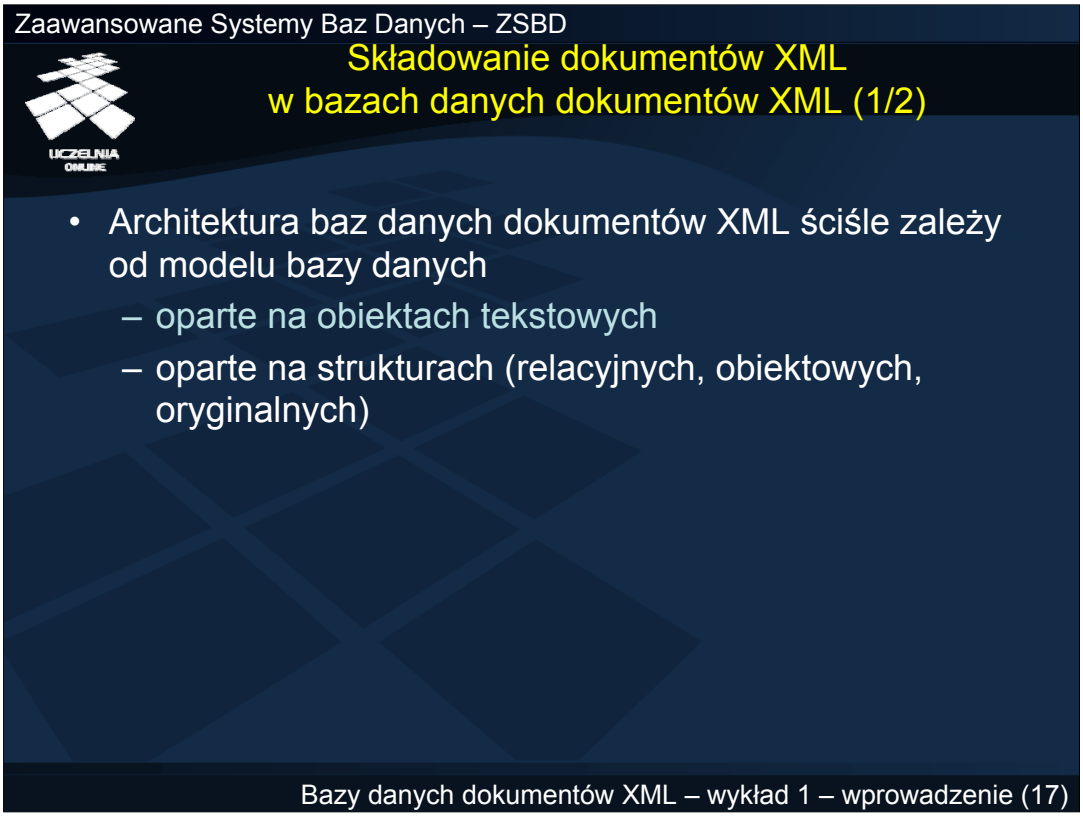
1. Umożliwienie użytkownikom przechowywania zbiorów dokumentów XML. Jednocześnie nie jest określone gdzie i w jaki sposób te dokumenty mogą być składowane.

2. Definiowanie i przechowywanie schematów dokumentów XML. Tak jak, dla przykładu, w relacyjnych systemach baz danych schemat relacji narzuca postać przechowywanych w bazie danych krotek, tak samo wymaga się od baz danych dokumentów XML możliwości definiowania schematów ograniczających postać przechowywanych dokumentów. Użytkownik powinien mieć także możliwość ich modyfikacji i odczytu.

3. Obsługa zapytań definiowanych przez użytkowników w oparciu o jeden lub wiele języków zapytań przeznaczonych do przetwarzania dokumentów XML.

4. Obsługa interfejsów programistycznych. W związku z tym, że do przetwarzania dokumentów XML jest wykorzystywanych obecnie szereg interfejsów programistycznych należy oczekiwać, że baza danych dokumentów XML będzie pozwalała na ich wykorzystanie. Daje to możliwość wykorzystania bazy danych w połączeniu z szeregiem popularnych narzędzi operujących na dokumentach XML z wykorzystaniem dla przykładu DOM API.

5. Ponadto, oczekuje się od baz danych dokumentów XML funkcjonalności tradycyjnych systemów zarządzania bazami danych. W szczególności chodzi tu o kwestie związane z wielodostępem, obsługą transakcji, mechanizmami archiwizacji i odtwarzania po awarii, importem i eksportem danych itp.



Architektura baz danych dokumentów XML ściśle zależy od modelu bazy danych. W tym kontekście możemy powiedzieć, że rozróżniamy bazy danych:

- oparte na obiektach tekstowych,

- oparte na strukturach (oparte na modelu).

Bazy danych oparte na obiektach tekstowych:

- dokumenty przechowują w całości, jako obiekty tekstowe,

- do składowania mogą wykorzystywać obiekty typu CLOB lub BLOB, systemy plików itp.,

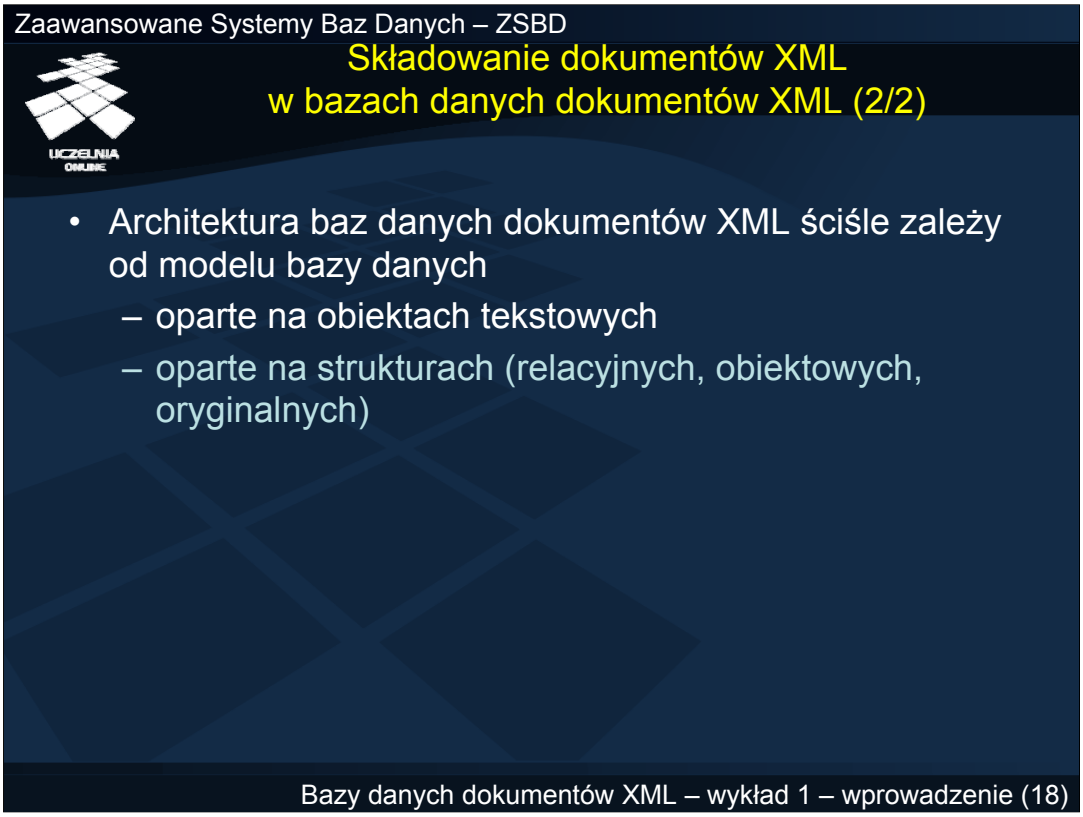
- mogą wymagać parsowania dokumentów przed ich wstawieniem do bazy danych,

- mogą wykorzystywać indeksy w celu uniknięcia parsowania, a także w celu przyspieszenia przeszukiwania; indeksy pozwalają na dostęp do dowolnego fragmentu dokumentu XML-owego,

- pobieranie dokumentów XML lub ich fragmentów nie wymaga ich rekonstrukcji – są one pobierane po prostu jako ciąg bajtów,

- pobierane dokumenty mają identyczną postać w jakiej zostały w bazie danych składowane, nie ma utraty białych znaków, komentarzy, instrukcji przetwarzania itp.,

- przykłady baz danych wykorzystujących indeksowane pliki to TextML, a wykorzystujących duże obiekty tekstowe to: Oracle, DB2 itp.



Bazy danych oparte na strukturach (oparte na modelu):

- składują dokumenty w formie "obiektów",

- parsują dokumenty w momencie ich wstawiania – wymagane jest to podczas procesu dekompozycji i tworzenia obiektowej reprezentacji dokumentów,

- składują dokumenty w oparciu o struktury: relacyjne, obiektowe, hierarchiczne, oryginalne itp.,

- wykorzystują indeksy przede wszystkim do przyspieszenia przeszukiwania dokumentów; wykorzystywane indeksy mogą być tradycyjnymi indeksami zależnym od wykorzystywanych struktur,

- pobieranie dokumentów wymaga ich ponownej rekonstrukcji, co wymaga znacznie większej liczby odczytów, niż ma to miejsce w przypadku baz danych opartych na obiektach tekstowych,

- z reguły wydajnie tworzą struktury oparte na dokumentach XML, dla przykładu drzewa DOM,

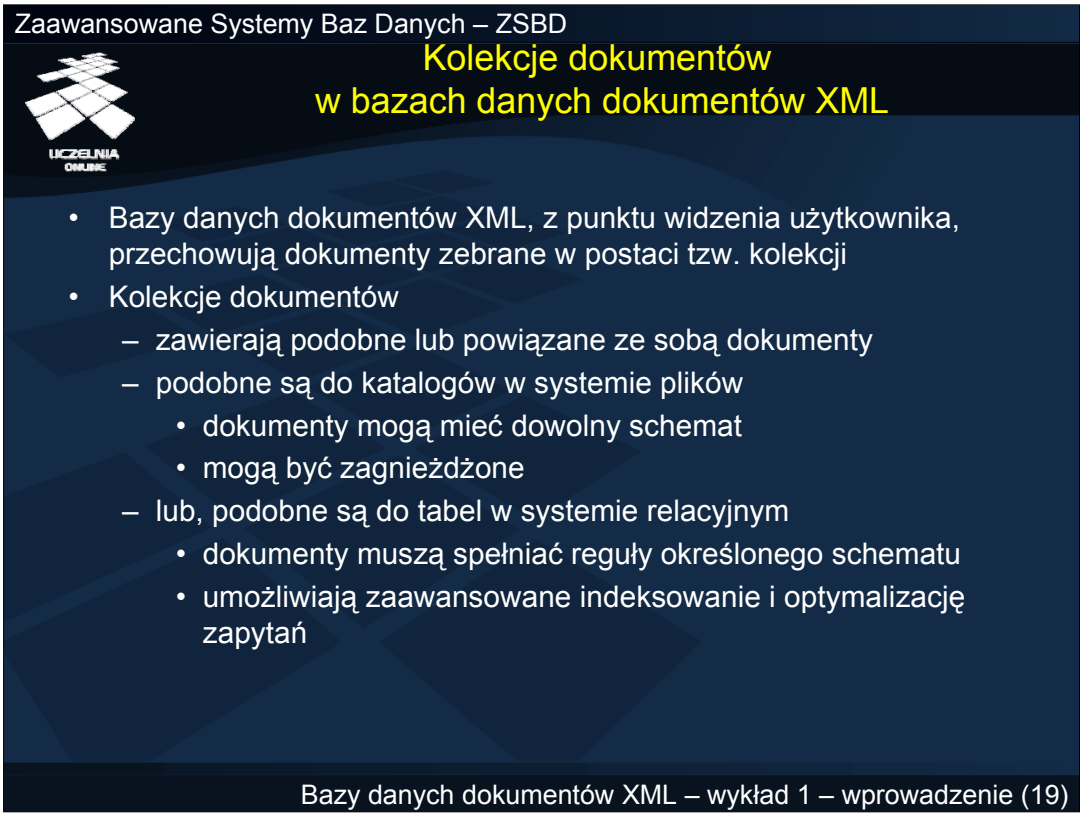
- przykładowo mogą przechowywać obiekty DOM w OORDBMS.

Przykładami baz danych opartymi na modelu i wykorzystującymi odpowiednie struktury są:

- Relacyjne: Xfinity, eXist, Sybase, DBDOM.

- Obiektowe: eXcelon, X-Hive, Ozone/Prowler, 4Suite, Birdstep.

- Oryginalne: Tamino, Xindice, Neocore, Ipedo, XStream DB, XYZFind, Infonyte, Virtuoso, Coherity, Luci, TeraText, Sekaiju, Cerisent, DOM-Safe, XDBM, i inne.



Bazy danych dokumentów XML, z punktu widzenia użytkownika, przechowują dokumenty zebrane w postaci tzw. kolekcji.

Kolekcje dokumentów zawierają podobne lub powiązane ze sobą dokumenty. Często kolekcje można przyrównać do tabel w relacyjnej bazie danych, które grupują obiekty o takim samym znaczeniu, lub do schematów, które grupują obiekty powiązane ze sobą referencjami, wykorzystaniem w jednej aplikacji itp.

Kolekcje dokumentów zawierają podobne lub powiązane ze sobą dokumenty, przykładowo mogą to być informacje o umówionych spotkaniach, o zespołach istniejących na poziomie firmy, o pracownikach, mogą to być dokumenty CV przesłane przez osoby ubiegające się o stanowisko w naszej firmie.

Kolekcje dokumentów:

- mogą być podobne do katalogów w systemie plików, w takich przypadkach:

  * dokumenty składowane wewnątrz kolekcji mogą mieć zwykle dowolną strukturę.

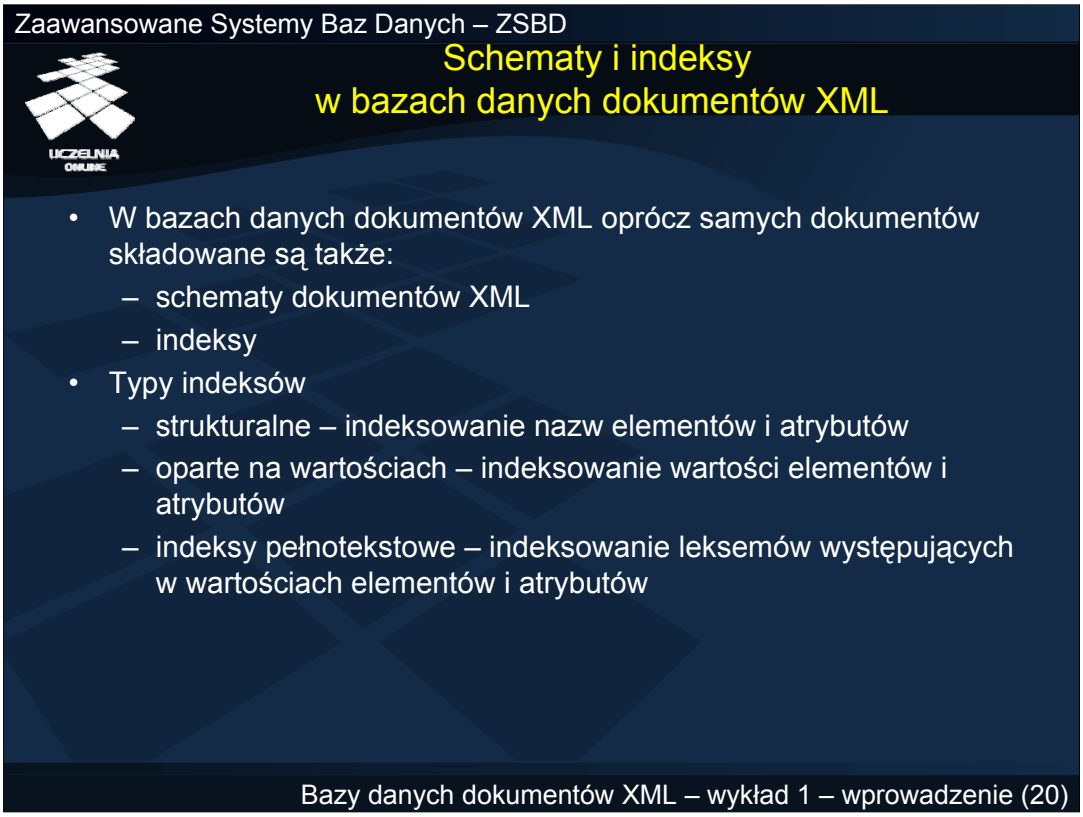
  * kolekcje mogą być wielokrotnie zagnieżdżone.

- mogą też być podobne są do tabel w systemie relacyjnym i wówczas:

  * dokumenty przechowywane w kolekcji muszą spełniać reguły określonego schematu, bardzo często przypisanego do kolekcji,

  * z reguły nie mogą być zagnieżdżane,

  * umożliwiają zaawansowane indeksowanie oraz zaawansowaną optymalizację zapytań, co jest ściśle związane z istnieniem definicji schematu.



W bazach danych dokumentów XML oprócz samych dokumentów składowane są także:

- schematy dokumentów XML,

- indeksy.

Zadaniem schematów jest przede wszystkim ograniczanie typów składowanych dokumentów oraz definiowane ograniczeń integralnościowych obowiązujących składowane dokumenty. W zależności od indywidualnych rozwiązań schematy XML mogą być składowane z punktu widzenia użytkownika albo w centralnym repozytorium, albo w ramach poszczególnych kolekcji.

Zadanie indeksów jest dwojakie. Tak jak wspomniano przy okazji omawiania sposobów składowania dokumentów XML, mogą one być wykorzystywane do parsowania składowanych dokumentów. Jednak głównym ich zadaniem, podobnie jak w przypadku relacyjnych czy obiektowych baz danych, jest zwiększenie wydajności przetwarzanych zapytań.

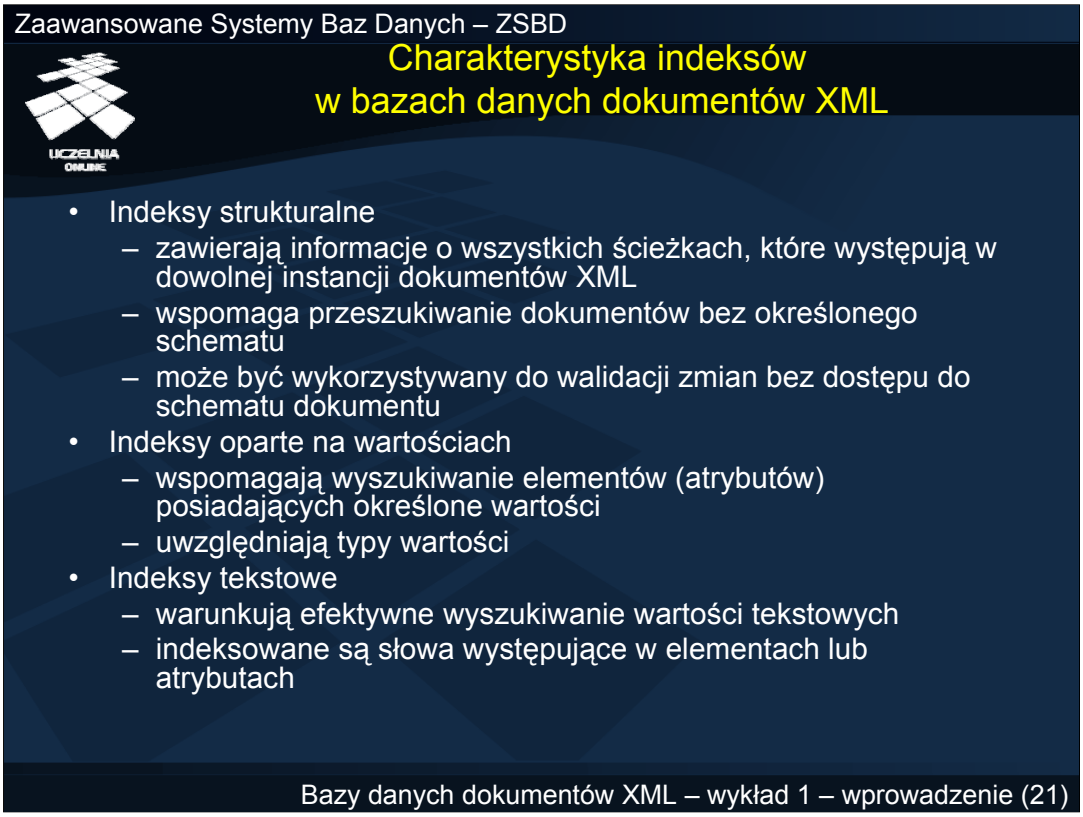
W bazach danych dokumentów XML możemy wyróżnić trzy podstawowe typy indeksów:

- indeksy strukturalne – indeksowanie nazw elementów i atrybutów, przyspiesza wyszukiwanie dokumentów posiadających określone elementy struktury,

- indeksy oparte na wartościach – indeksowanie wartości elementów i atrybutów, przyspiesza wyszukiwanie dokumentów posiadających określone wartości w określonych fragmentach dokumentu,

- indeksy pełnotekstowe – indeksowanie leksemów występujących w wartościach elementów i atrybutów, pozwala przyspieszyć wyszukiwanie dokumentów posiadających określone leksemy, często niezależnie od struktury, formy itd.

Należy zaznaczyć, że nie wszystkie typy indeksów występują w każdej bazie danych dokumentów XML.



Indeksy strukturalne:

- skondensowana struktura indeksu zawiera informacje o wszystkich ścieżkach, które występują w dowolnej instancji określonego typu dokumentu,

- wspomaga przeszukiwanie dokumentu bez określonego schematu; w takich sytuacjach bez takiego indeksu podczas zapytania należałoby przeglądać cały dokument,

- dla dokumentów o określonym schemacie indeks ten może być wykorzystywany do walidacji zmian bez dostępu do schematu dokumentu,

- struktura indeksu oprócz faktu istnienia ścieżek, zawiera informacje o tym, które dokumenty tą ścieżkę zawierają; pozwala to znacząco przyspieszać zapytania, które dotyczą opcjonalnych fragmentów dokumentu.

Indeksy oparte na wartościach:

- wspomagają wyszukiwanie elementów (atrybutów) posiadających określone wartości,

- uwzględniają typy wartości.

Indeksy tekstowe:

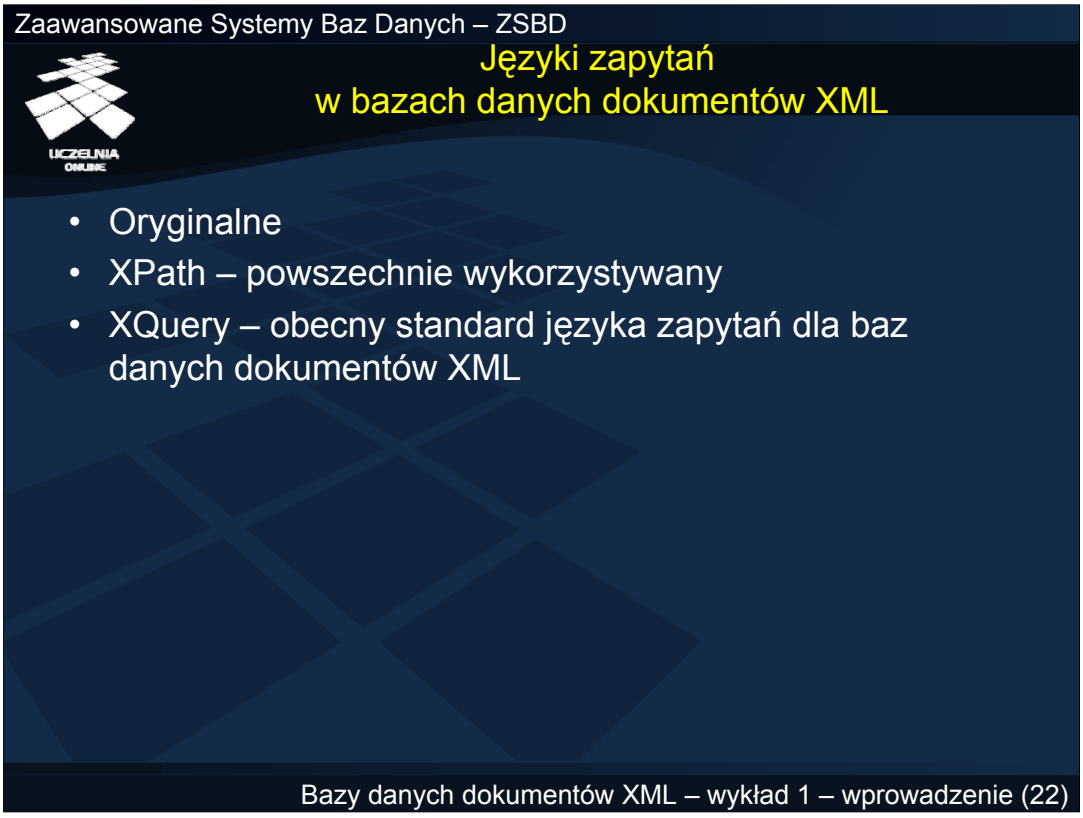
- warunkują efektywne wyszukiwanie wartości tekstowych,

- indeksowane są słowa występujące w elementach lub atrybutach,

- indeksy tekstowe nie są definiowane tylko na liściach lecz także elementach zawierających podelementy – to pozwala na wyszukiwanie obiektów lub dokumentów mających w swoim poddrzewie określone słowa,

- podczas budowy indeksu podział na słowa może odbywać się za pomocą funkcji XQuery fn:tokenize (text()),

- dane nie XML-owe (jeśli mogą być przechowywane w bazie danych dokumentów XML) są często automatycznie indeksowane tym typem indeksu.

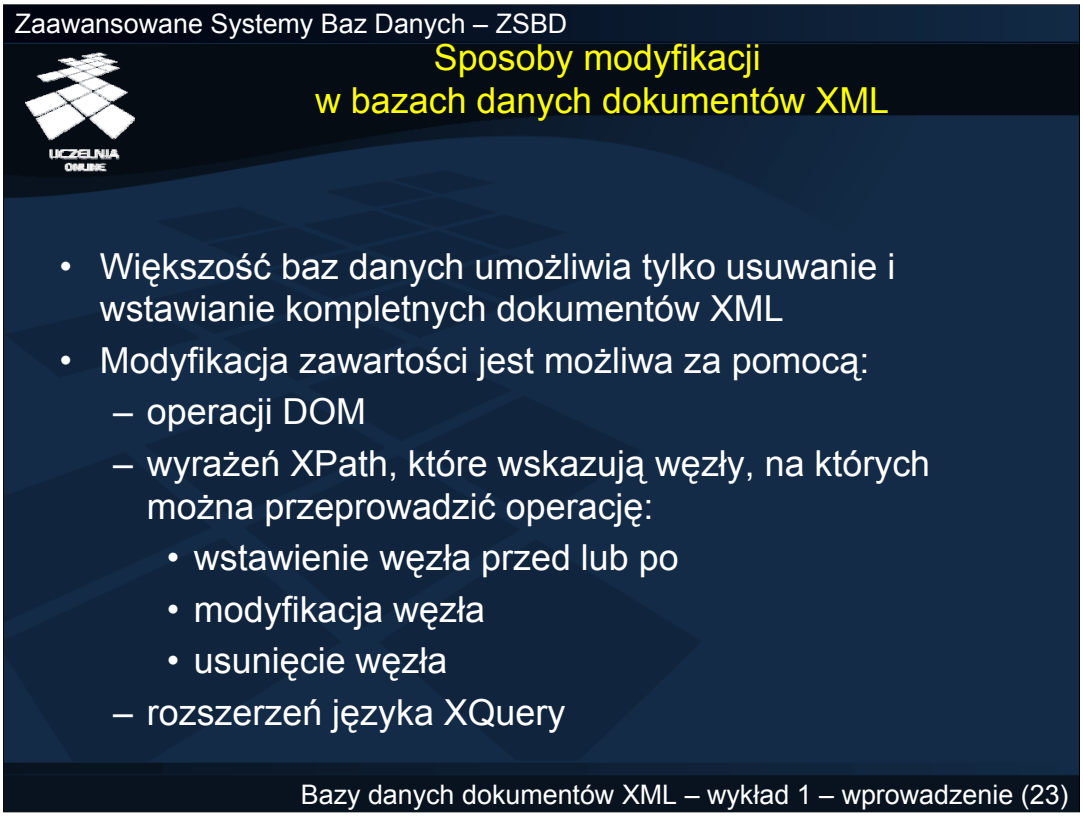


Podobnie jak swego czasu w przypadku systemów baz danych relacyjnych czy obiektowych, również w przypadku systemów baz danych dokumentów XML opracowano odpowiednie języki zapytań. Proces definiowania standardu języka zapytań do przetwarzania dokumentów XML zakończył się stosunkowo nie dawno. Standardem rekomendowanym przez organizację W3C jest język XQuery.

W roku 2005 jedynie kilka baz danych dokumentów XML pozwalało na dostęp do swoich zasobów za pomocą języka XQuery. Najbardziej rozpowszechniony w owym czasie był interfejs pozwalający na wykonywanie zapytań z użyciem wyrażeń XPath. Niektóre z komercyjnych baz danych umożliwiało wykonywanie zapytań w oparciu o własne propozycje. Przykładowo, Tamino umożliwiało wykorzystywanie języka X-Query, będący rozszerzeniem wyrażeń XPath o nowe możliwości.

W chwili obecnej XQuery to powszechnie wykorzystywany standard języka zapytań implementowany w większości baz danych dokumentów XML.

Język XQuery zostanie omówiony w drugim wykładzie poświęconym bazom danych dokumentów XML.



Duża część baz danych dokumentów XML umożliwia tylko usuwanie i wstawianie kompletnych dokumentów XML. Jest to dalekie od standardów przyjętych w bazach danych obiektowych i relacyjnych.

Bazy danych, które umożliwią modyfikację fragmentów dokumentów XML stosują następujące podejścia:

- umożliwiają wykonywanie operacji DOM na dokumentach w nich zawartych,

- umożliwiają wykorzystanie wyrażeń XPath, które wskazują węzły, na których można przeprowadzić jedną lub wiele operacji takich jak:

  * wstawienie węzła przed lub po wskazywanych przez wyrażenia XPath fragmentach,

  * modyfikacja wskazywanego węzła,

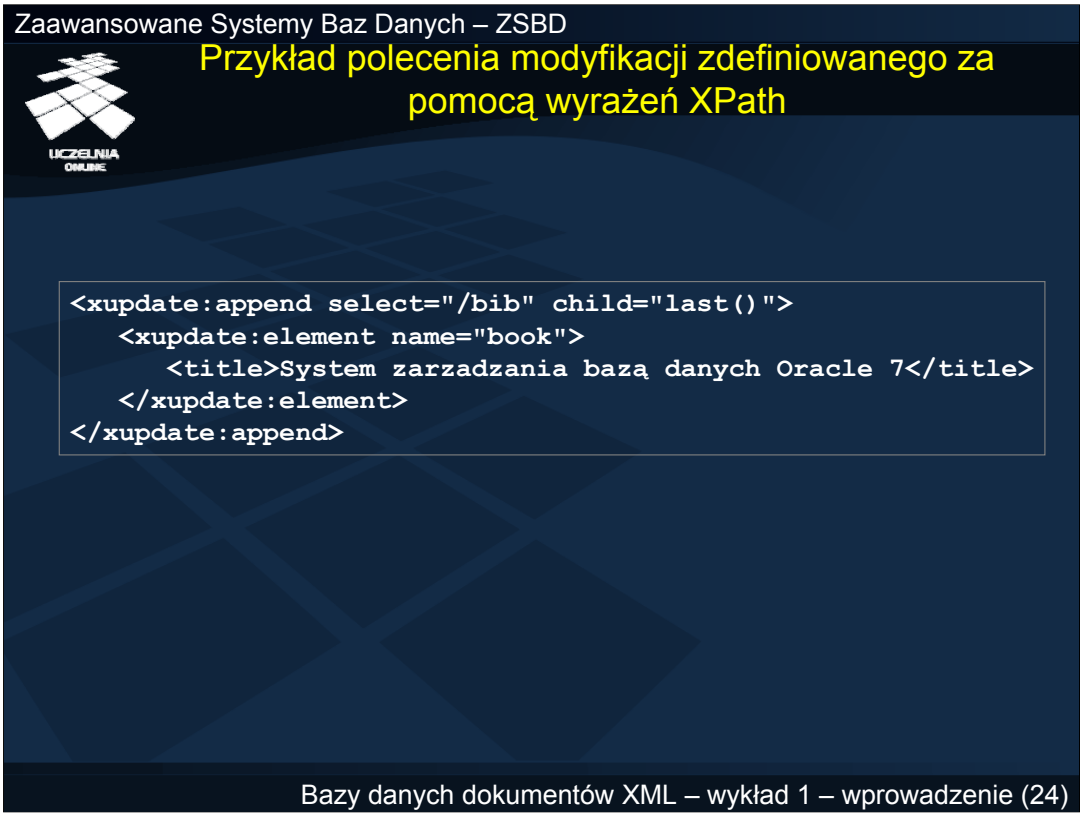
  * usunięcie wskazywanego węzła,

  * utworzenie zmiennej, której zawartość będzie identyczna ze wskazywanym węzłem,

  * zmiana nazwy znacznika wskazywanego elementu;

- wykorzystanie rozszerzeń języka XQuery.

Modyfikacja dokumentów XML za pomocą interfejsu DOM wymaga podejścia proceduralnego. Jest ono często satysfakcjonujące np. w przypadku edytorów dokumentów XML zintegrowanych z bazami danych. Nie jest to jednak podejście satysfakcjonujące użytkownika przyzwyczajonego do języków deklaratywnych takich jak SQL.



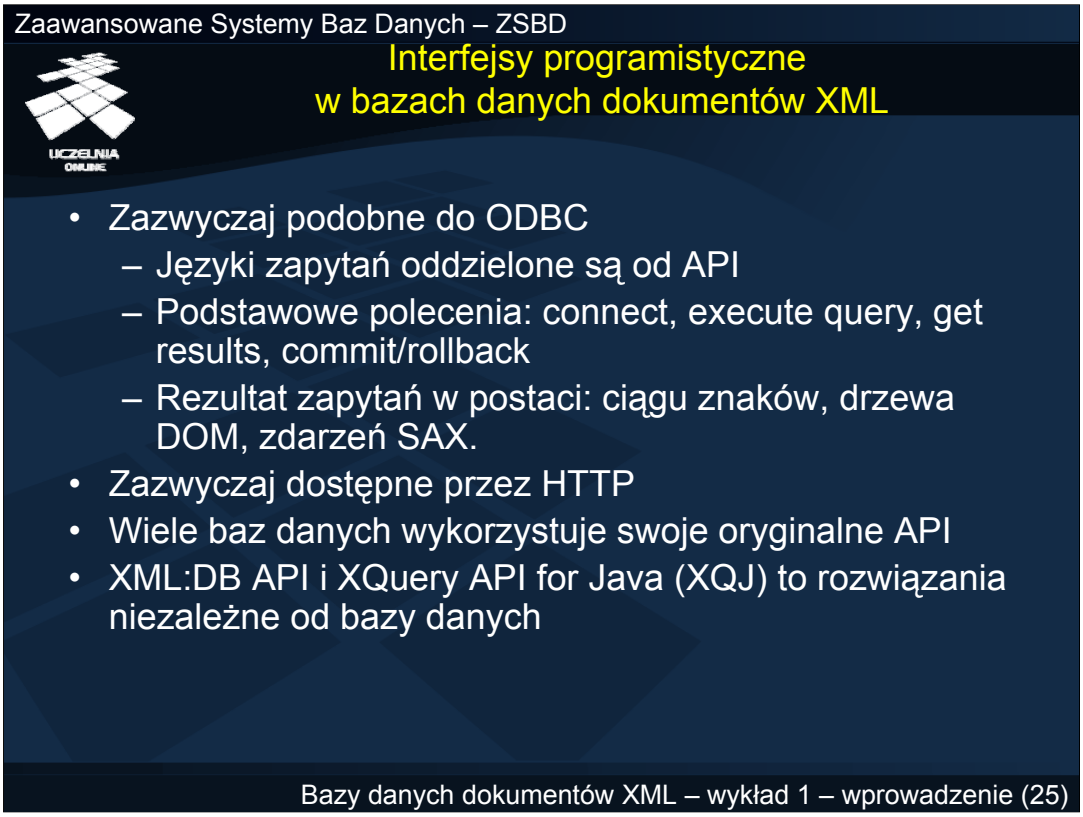
Podejście drugie oparte na wyrażeniach XPath jest w chwili obecnej najbardziej rozpowszechnione. Językiem najczęściej wykorzystywanym i pozwalającym w ten sposób definiować operacje modyfikacji jest XUpdate.

Przykładowo, polecenie na slajdzie żąda dodania elementu book, jako ostatniego w elemencie bib, będącym korzeniem dokumentu XML.

Niestety również to podejście nie jest pozbawione wad. W szczególności dotyczą one definiowania modyfikacji, które mają dotyczyć szeregu elementów.

Rozwiązaniem wydaje się być adaptacja języka zapytań XQuery do możliwości wykonywania operacji modyfikacji. Szereg komercyjnych baz danych już kilka lat temu wprowadzało tego typu rozwiązania. Należy jednak podkreślić, że, jak dotąd, nie ma wyznaczonego standardu. Nie ma również gwarancji, że przyszłe rozwiązanie dotyczące języka modyfikacji dokumentów XML będzie oparte na tym podejściu.

Językom modyfikacji dokumentów XML poświęcony został trzeci wykład dotyczący baz danych dokumentów XML.



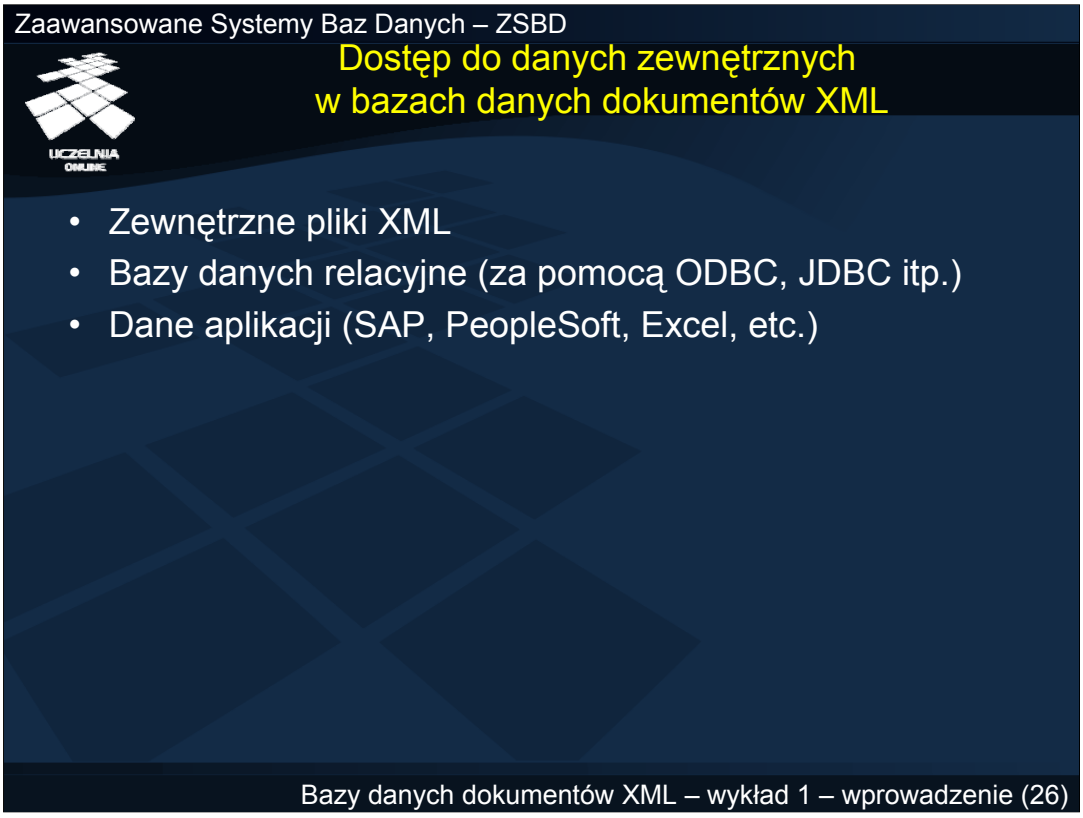
Większość z baz danych dokumentów XML udostępnia interfejsy programistyczne podobne do ODBC. Ich zadaniem jest udostępnienie programistom metod pozwalających na łączenie się z bazą danych, eksploracje metadanych takich jak dla przykładu schematy dokumentów XML lub nazwy plików lub kolekcji składowanych w bazie danych, wykonywanie poleceń i zapytań, a także pobieranie wyników. Wyniki zapytań najczęściej przyjmują postać: ciągu znaków, drzewa DOM, parsera SAX. Wiele baz danych pozwala na uzyskiwanie wyników pochodzących z wielu dokumentów znajdujących się w jednej lub wielu kolekcjach.

Większość interfejsów udostępnianych przez bazy danych dokumentów XML jest unikalnych, możliwych do zastosowanie tylko w przypadku jednej, ściśle określonej bazy danych. W roku 2004 dostępne były tylko dla interfejsy niezależne od bazy danych:

- XML:DB API – rozwijane przez XML:DB.org, która jest autorem także bazy danych dokumentów XML XMLDB oraz miała wkład w powstanie języka XUpdate,

- XQuery API for Java (XQJ) – interfejs oparty na języku JAVA i dający programistom dostęp do baz danych za pomocą zapytań XQuery. Interfejs rozwijany w ramach Sun's Java Community Process (JCP).

Ponadto wiele z baz danych dokumentów XML udostępnia interfejs pozwalający na przeglądanie dokumentów i wykonywanie zapytań za pomocą protokołu HTTP.

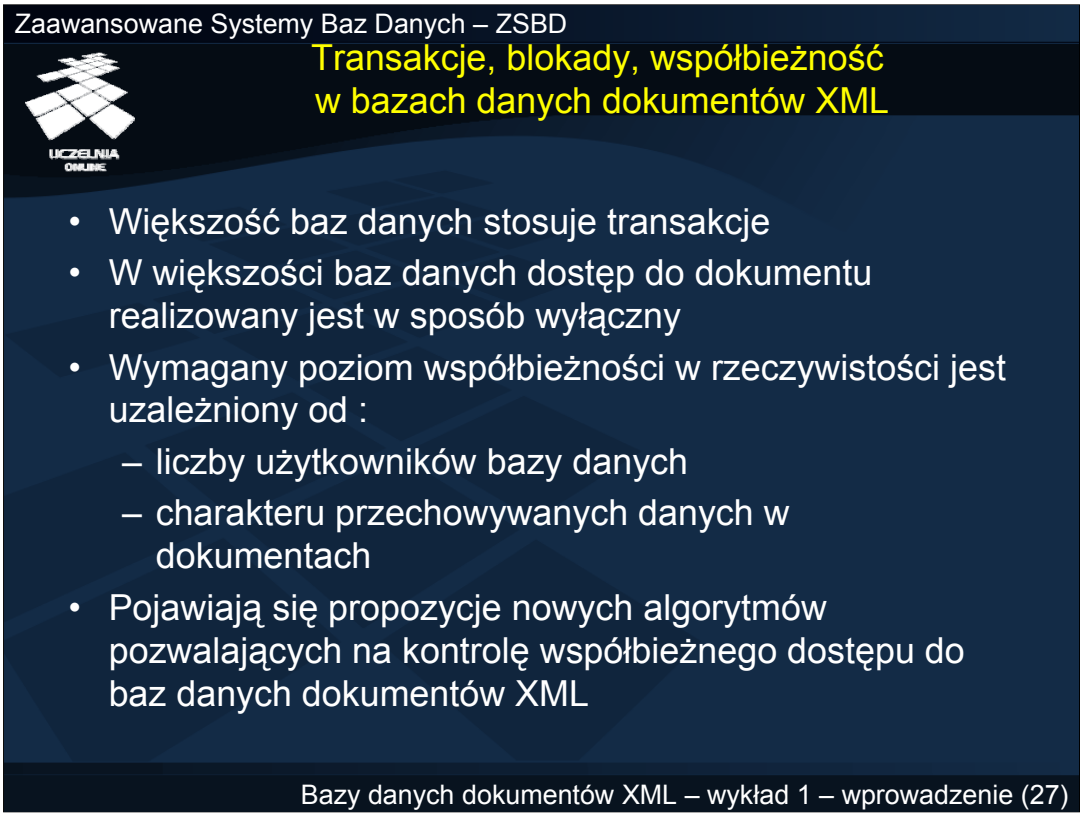


Kolejną funkcjonalnością baz danych dokumentów XML jest możliwość dostępu za ich pomocą do informacji zewnętrznych. Przykładami informacji zewnętrznych mogą być:

- pliki XML,

- źródła danych relacyjne lub obiektowe, udostępnianie w postaci wirtualnych plików XML przy wykorzystaniu odpowiedniego mapowania (dostęp za pomocą ODBC, JDBC itp.),

- dane aplikacji zewnętrznych (SAP, PeopleSoft, Excel, itp.).



Większość baz danych dokumentów XML pozwala użytkownikom na wykorzystywanie transakcji przy dostępie do dokumentów i ich modyfikacji. Dużym problemem, w chwili obecnej, jest umożliwienie współbieżnego dostępu wielu użytkownikom do pojedynczego dokumentu. W większości przypadków w bazach danych dokumentów XML dostęp i modyfikacja dokumentów wymaga założenia blokady na poziomie dokumentu.

Takie podejście ma dwa źródła. Pierwsze z nich to fakt, że bazy danych dokumentów XML początkowo miały charakter repozytoriów udostępniających głównie operacje odczytu. Drugi powód to fakt, że dokument w bazach dokumentów XML przez wielu traktowany jest na równi z krotką w systemie relacyjnym.

Wymagany poziom współbieżności w rzeczywistości jest uzależniony od:

- liczby użytkowników bazy danych,

- charakteru przechowywanych danych w dokumentach.

Dla przykładu, jeśli dokumenty zawierają rozbudowane kompozycje graficzne zdefiniowane w oparciu o XML-owy standard SVG, to może się okazać, że zawłaszczenie dokumentu przez jednego użytkownika jest całkowicie niedopuszczalne.

Dlatego też coraz częściej pojawiają się propozycje nowych algorytmów pozwalających na kontrolę współbieżnego dostępu do baz danych dokumentów XML jednocześnie zapewniając odpowiedni poziom współbieżności.

Literatura

- http://www.rpbourret.com/xml/
- http://xmldb-org.sourceforge.net/index.html
- http://www.garshol.priv.no/download/xmltools/
- http://www.oasis-open.org/cover/xmlAndDatabases.html
- *Wprowadzenie do systemów baz danych*, Ramez Elmasri, Shamkant B. Navathe, ISBN: 83-7361-716-7