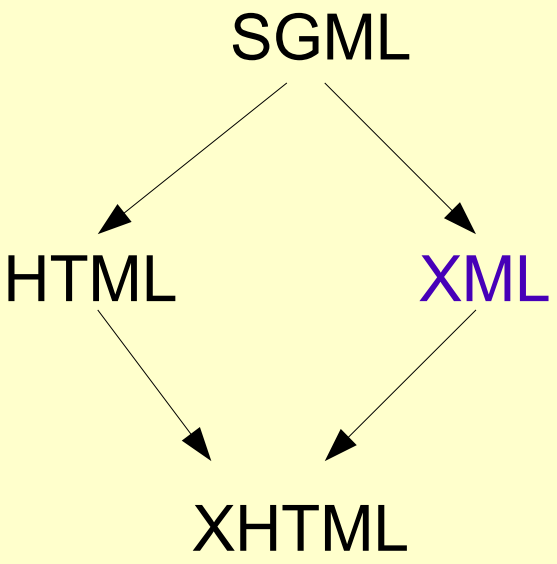
Navigačné modely

- XML databázy
  - Dátový model sú stromy
  - Navigácia po stromovej štruktúre prípadne po referenciách v uzloch
  - Vlastné jazyky Xpath, Xquery
- Objektové databázy
  - Navigácia prostredníctvom referencii v objektoch
  - Dlho sa o nich hovorí, ale doposiaľ nevznikol skutočný objektový dátový model. Spoliehajú na princípy objektového programovania a manifest.
  - OQL je modifikácia SQL pre databázy s komplexnými objektami (nie v prvej normálnej forme)
- Sieťové datázy
  - Dátový model acyklicky graf základ sú binárne 1:n vzťahy nazvané Set: Owner → Member
  - Navigácia hlavne cez sety
  - Hostiteľský programovací jazyk (Cobol, PL/1, C, …)

XML databázy



Značkovacie jazyky vyvinuté pre uchovanie štruktúrovaných textových dokumentov a ich prezentáciu na webe.
Hlavné rozdiely medzi XML a HTML sú:

- HTML má preddefinované tagy (zátvorky).
- XML zátvorky musia byť dobre spárované.
- ID/IDREF musia splňovať podmienky:
  - ID musia byť navzájom rôzne
  - IDREF/IDREFS musia byť existujúce ID

Pretože XML nemá preddefinované tagy potrebujeme nástroj na popis štruktúry dokumentov. Sú hneď tri: DTD, XSD, XDR.

Presné definície W3C: http://www.w3.org/
http://www.zvon.org/xxl/XMLTutorial/General/book.html
Podrobnejší kurz v angličtine: http://www.cs.toronto.edu/~libkin/csc2538/

Príklad

Jednoduchý XML dokument (fragment dokumentu)

```
<študent_fmfi fmfičíslo="fmfi000x">
   <meno>Jozef Novák</meno>
   <vek>21</vek>
</študent_fmfi>
```

Definícia typu dát (DTD)

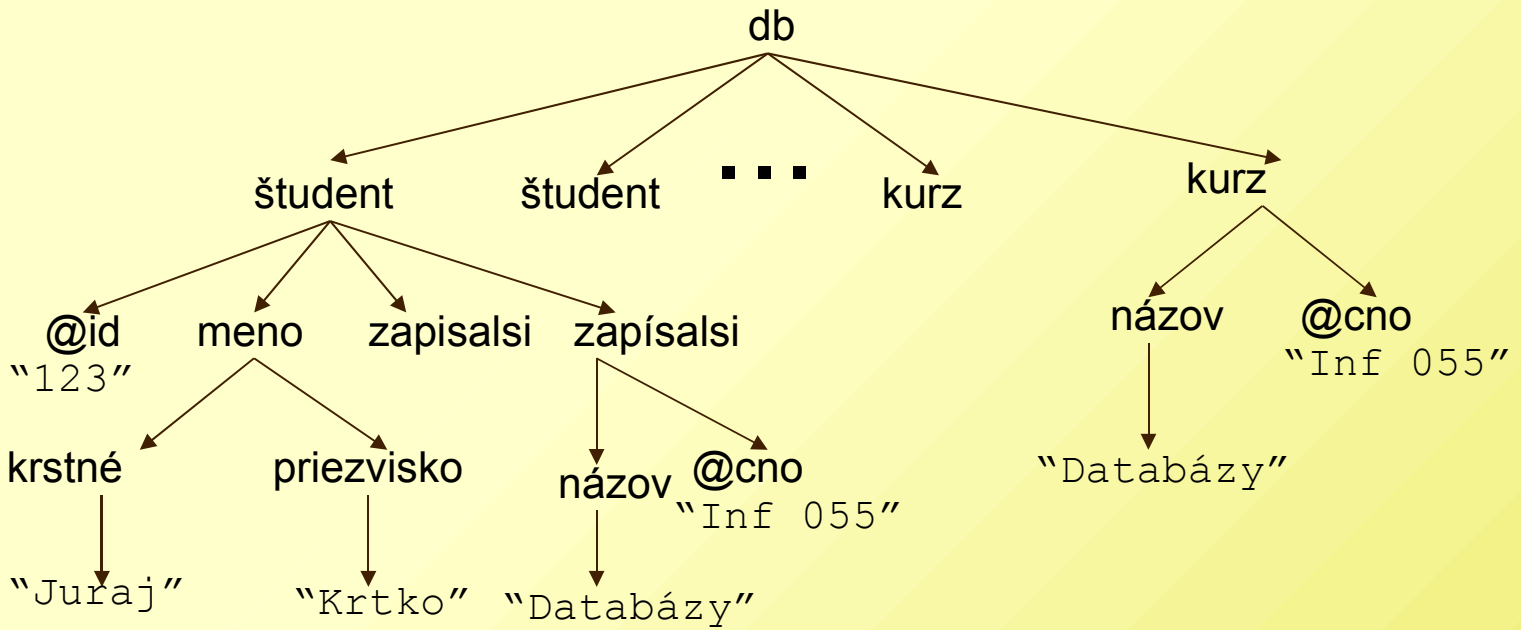
```
<!ELEMENT študent_fmfi (meno, vek)>
  <!ATTLIST študent_fmfi fmfičíslo CDATA>
  <!ELEMENT meno (#PCDATA)>
  <!ELEMENT vek (#PCDATA)>
```

XML definícia schémy (XSD)

```
<schema xmlns="http://www.w3.org/2001/XMLSchema" >
  <element name="študent_fmfi ">
   <complexType>
    <sequence>
     <element name="meno" type="string"/>
     <element name="vek" type="unsignedInt"/>
    </sequence>
    <attribute name="fmfičíslo">
     <simpleType>
      <restriction base="string">
       <pattern value="fmfi\d{3}[A-Za-z]{1}"/>
      </restriction>
     </simpleType>
    </attribute>
   </complexType>
  </element>
</schema>
```

Dátový model – labeled trees

Nevyžaduje pevnú schému – semištruktúrované dáta.
XML dáta modelujeme stromami, ktorých uzly sú označkované štrukturálnymi tagmi.



Stromy väčšinou berieme ako neusporiadané. Pri niektorých operáciach sú však usporiadané (following-sibling::)

Xpath – základné konštrukcie

Výraz	splňuje
/	koreň, alebo oddelovač medzi krokmi cesty
X	element X
*	akýkoľvek element
@X	atribút X bežného elementu ("context node")
//	bežný element a všetci jeho potomkovia
[C]	ohodnotí podmieku C pre bežný element
[N]	vyberie N-tý zodpovedajúci element
Path1 | Path2	zjednotenie ciest Path1 a Path2
contains(s1,s2)	vráti TRUE, ak reťazec s1 obsahuje reťazec s2
name()	vráti tag bežného elementu
parent::	otec bežného elementu, ak existuje
preceding-sibling::	všetci starší súrodenci bežného elementu
following-sibling::	všetci mladší súrodenci bežného elementu
descendants::	všetci potomkovia bežného elementu
self::	bežný element

Špecifikácia: http://www.w3.org/TR/xpath
Tútorial (český):
http://www.zvon.org/xxl/XPathTutorial/General_cze/examples.html

XQuery

- path expression
- element constructor
- FLoWeR (for, let, where, return) expression
- conditional expression (if then else)
- quantifiers (some, every)
- function definition

Špecifikácia: http://www.w3.org/TR/xquery/
Tútorialy: http://www.ipedo.com/html/xquery/xquery_tutorial/
http://www.w3schools.com/xquery/default.asp

Path expressions

Xquery podporuje path expressions z Xpath (navrhovaná verzia).

1. `//zamestnanec[meno="Fero"]/plat * 12`

Z dokumentu, čo obsahuje zamestnanncov a ich platy, vyber ročný plat zamestnanca, čo sa volá "Fero".

2. `document("zoo.xml")//kapitola[2 TO 5]//obrázok`

Nájdi všetky obrázky v kapitolách 2 až 5 v dokumente "zoo.xml".

Element constructors

Niekedy je potrebné vytvoriť nový XML element. Takéto elementy sa dajú vnoriť priamo do dotazu. Zodpovedajúca konštrukcia sa nazýva element constructor

```
<zamestnanec  idzam = {$id}>
              {$meno}
              {$zaradenie}
</zamestnanec>
```

Vytvorí element <zamestnanec>. Tento má atribút "idzam". Hodnota atribútu a obsah položiek elementu je určený premennými z iných častí dotazu.

FLWR expressions

Príkaz cyklu FOR je iteratívna konštrukcia, ktorá priraďuje premennej postupnosť hodnôt vrátenú dotazom (obvykle „by a path expression“).

Príkaz LET tiež priraďuje premenným hodnoty, na rozdiel postupnosti priradení sa uskutoční len jedno. Podobne ako to robí priraďovací príkaz v programovacom jazyku.

Klauzula WHERE obsahuje podmienky, ktoré musia spĺňovať uzly vybrané prdchádzajúcin príkazom FOR alebo LET.

Príkaz RETURN generuje výstup FLWR výrazu. Môže to byť postupnosť uzlov alebo atomických hodnôt. Príkaz RETURN sa vykoná raz pre každý uzol vrátený príkazom FOR alebo LET a splňujúci podmienky za WHERE. Výsledky tohto niekoľko násobného vykonania sú zreťazené a vrátené ako výsledok FLWR výrazu.

FLWR expressions – príklady

1.
```
FOR $b IN document("bib.xml")//book
        WHERE $b/publisher = "Morgan Kaufmann"
        AND $b/year = "1998"
        RETURN $b/title
```

Zoznam názvov kníh vydaných vydavateľstvom "Morgan Kaufmann" v roku 1998.

2.
```
<big_publishers>
    {
      FOR $p IN distinct(document("bib.xml")//publisher)
      LET $b := document("bib.xml")//book[publisher = $p]
      WHERE count($b) > 100
      RETURN $p
    }
</big_publishers>
```

Vydavatelia, ktorý vydali viac ako 100 kníh.

Podmienený výraz

Podmienený výraz vyhodnotí testovací výraz a vráti jeden z dvoch možných výsledných výrazov. Ak hodnota testovaného výrazu je true, vráti hodnotu výrazu za THEN, inak vráti hodnotu výrazu za ELSE.

```
FOR $h IN //holding
  RETURN
    <holding>
      {$h/title,
       IF ($h/@type = "Journal")
       THEN $h/editor
       ELSE $h/author
      }
    </holding>
  SORTBY (title)
```

Vypíš zoznam dvojíc < názov, vlastník > usporiadaný podľa názvu. Vlastníkom časopisu je vydavateľ a vlastníkom každého iného typu dokumentu je autor.

Kvantifikátory

Kvantifikátory umožňujú klásť podmienky na množiny uzlov. Existenčný kantifikátor SOME je splnený, keď aspoň jeden uzol z danej množiny splňuje podmienku. Univerzálny kvantifikátor EVERY je splnený, iba ak všetky uzly z danej množiny splňujú podmienku.

1.
```
FOR $b IN //book
WHERE SOME $p IN $b//paragraph SATISFIES
  (contains($p, "sailing") AND contains($p, "windsurfing"))
 RETURN $b/title
```

Nájdi názvy všetkých kníh, v ktorých sa aspoň v jednom odstavci vyskytujú obe slová: „sailing“ a „windsurfing“.

2.
```
FOR $b IN //book
WHERE EVERY $p IN $b//paragraph SATISFIES
   contains($p, "sailing")
RETURN $b/title
```

Nájdi názvy všetkých kníh, v korých každý odstavec obsahuje slovo „sailing“.

Definícia funkcie

Okrem základnej knižnice funkcií, ktorú poskytuje XPath, XQuery dovoľuje rozšíriť základné funnkcie o uživateľom definované funkcie – programy.

```
NAMESPACE xsd = "http://www.w3.org/2001/XMLSchema"
DEFINE FUNCTION depth($e) RETURNS xsd:integer
{
  # A leave element has depth 1
  # Otherwise, add 1 to max depth of children
  IF (empty($e/*)) THEN 1
  ELSE max(depth($e/*)) + 1
}

depth(document("partlist.xml"))
```

Vypočítaj hĺbku dokumentu "partlist.xml."

Dokument centrické použitie – XHTML

XML je použité ako nástroj na vytvorenie semi-štruktúrovaných dokumentov s rôznym obsahom, určených ľudskému čitateľovi. Príkladom dokument centrického použitia XML je XHTML.

Vzor textu v XHTML

```
<html xmlns ="http://www.w3.org/1999/xhtml">
    <head>
       <title>Sample Web Page</title>
    </head>
    <body>
     <h1>My Sample Web Page</h1>
     <p> XHTML documents must be well-formed and valid. </p>
     <img src="http://www.example.com/sample.jpg" height ="50"
      width = "25"/>
     <br />
     <br />
    </body>
</html>
```

Dáta centrické použitie – SOAP

V data centrickom modeli, sa XML ako formát pre zapamätanie a výmenu dát, ktorý je štruktúrovaný, dáta sa vyskytujú v regulárnom poradí a sú vhodné skôr pre strojové spracovanie ako na ľudské čítanie. Skutočnosť, že dáta sa prenášajú pamätajú v XML formáte je bezvýznamná, lebo existujú aj iné vhodné formáty. Príkladom data centrického použitia XML je SOAP. SOAP je protokol na výmenu informácie decentralizovanom a distribuovanom prostredí založený na XML. SOAP správa pozostáva z troch častí: z obálky (envelope), ktorá definuje rámec pre popis obsahu správy, pravidiel kódovanie pre popis aplikačných dátových typov, a konvencií pre vyvolanie vzdialených procedúr spracovania.

```
<SOAP-ENV:Envelope  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
       <SOAP-ENV:Body>
        <m:GetLastTradePrice xmlns:m="Some-URI">
         <symbol>DIS</symbol>
        </m:GetLastTradePrice>
       </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Tento spôsob nemá žiadnu výhodu oproti inému dohovorenému protokolu, okrem viery, že mu prijímateľ bude rozumieť bez ďalšieho výkladu.

XML interface k relačným databázam

V dáta centrickom modeli môžu dáta byť uložené v relačnej báze dát. Len na výstupe pre komunikáciu sú konvertované do formátu XML. Aby to fungovalo potrebujeme:

- Program na konverziu výsledkov SQL dotazu do XML
  - triviálna (raw) schéma <rows><row rowid= > … </></>
  - explicitné priradenie podobné priradeniu v embeded SQL
- Program na transformáciu XML dokumentov do relačného modelu
  - trivialne: tabuľka hrán a tabuľka (nodeid, name, value) nevýhoda je časté počítanie ciest a tranzitívnych uzáverov
  - múdrejšie transformácie ?
- Preklad Xpath a Xquery do SQL
- Nízko úrovňové problémy
  - Unicode vs. ASCII
  - blank spaces

Existuje aj pokus o normu SQL/XML:
http://www.sqlx.org/

Natívne XML databázy

- Natívna XML databáza je kolekcia XML dokumentov spojená do jednedneho stromu cez koreň.
- Používa dotazové jazyky XPath a XQuery, často sprostredkovane cez grafický uživateľský interface.
- Update cez XUpdate
- Správa databázy pomocou XML editoru.
- Doslova vzaté natívne XML databázy neexistujú, každá implementácia používa pomocné štruktúry (indexy, katalógy, slovníky, ...), ktoré obvykle už nie sú v XML.
- Produkty, ktoré deklarujú príslušnosť k tejto triede: Tamino, Timber, dbXML, … .

Hybridné systémy

- Koexistencia relačnej SQL databázy a XML databázy s Xpath a Xquery.
- Často podporuje aj vstup výstup a update relačnej databázy prostredníctvom XML formulárov.
- Odlišuje sa od XML interface k relačným databázam len tým, že XML dokumenty nie je potrebné konvertovať do relačnej db.

Väčšina súčasnych databázových systémov je hybridných, alebo poskytujú aspoň XML interface.

Oracle 9i, Oracle Text; Microsoft SQL Server 2000;
IBM DB2 XML; ale aj menej známe Caché, 602SQL Server 9.5.
Výber je náhodný, čo ma napadlo. Pravdepodobne je aj veľa iných.

Web ako jedna distribuovaná databáza

Láka predstava pozerať sa na celý web ako na jednu veľkú XML databázu.

- Trochu problém je, že chýba jednoznačný koreň (dá sa dodefinovať).
- Úplná neznalosť štruktúry znemožňuje zložitejšie dotazy.
- Dodazy iba ako na textové databázy. Vyhľadanie, konjunkcie, disjunkcie, prípadne rozdielu ($\wedge\neg$) nejakých fráz. Je tu problém porozumenia jazyku (ohýbanie slov, synonýma a hononýma, …).
- Robia to vyhľadávacie servery napr. Google, Yahoo, AltaVista, … .
- Aby sa sa to dalo robiť efektívne takýto server potrebuje nejaký adresár, kde má nainedxované dokumenty. To predpripravuje špeciálny program (crawler) a nejakú vyrovnávaciu pamäť cache, kde sú uchované najčastejšie použité dokumenty.
- Dokumenty sa hodnotia nakoľko sú relevantné k danému dotazu. Aj informačná hodnota a pôvodnosť dokumentu sa musí hodnotiť (ranking).
- Problém sú utopené referencie (zmazané a premiestnené dokumenty).
- Navyše väčšina stránok nie sú korektné XML (ani html) dokumenty.

Problémy XML databáz

- Konzistencia. Netriviálna kolekcia dokumentov je určite protirečivá.
- Normalizácia (odstranenie redundancie). Určite sa to dá urobiť, ale takéto dokumenty nebudú prehľadné. Väčšinou to budú len zmeny k iným dokumentom. Akonáhle povolíme opakovanie informácie vzniká potenciál pre inkozistenciu.
- Vzťah k logike a modely. Relačné databázy predikátový kalkul. Pokiaľ sa obmedzíme na Hornové formule, existuje najmenší model. Je asi problém vymyslieť, tak čistú a jednoznačú sémantiku pre XML databázy.
- Závislosť semištruktúrovaných dát na priestore mien (name space) a aliasoch.