SMA 5506
Database Technology

Summer 2005

Session L16

WEB AS A DATABASE: EVOLUTION OF XML

Prof. Stuart Madnick
(smadnick@mit.edu)

Traditional Use of the Web: For direct human usage Focus: **Entertainment**

Examples:

Extract just mortgage rate information

Compare mortgage rates offered by multiple sources

Build **cumulative** database of mortgage rates over time

Compare cumulative rates with previously stored, **alert** to new highs and lows

Programs for:

Web Wrapper and Context Mediator Technology

Enhancing

Analyzing

Consolidating

Processing

A

C

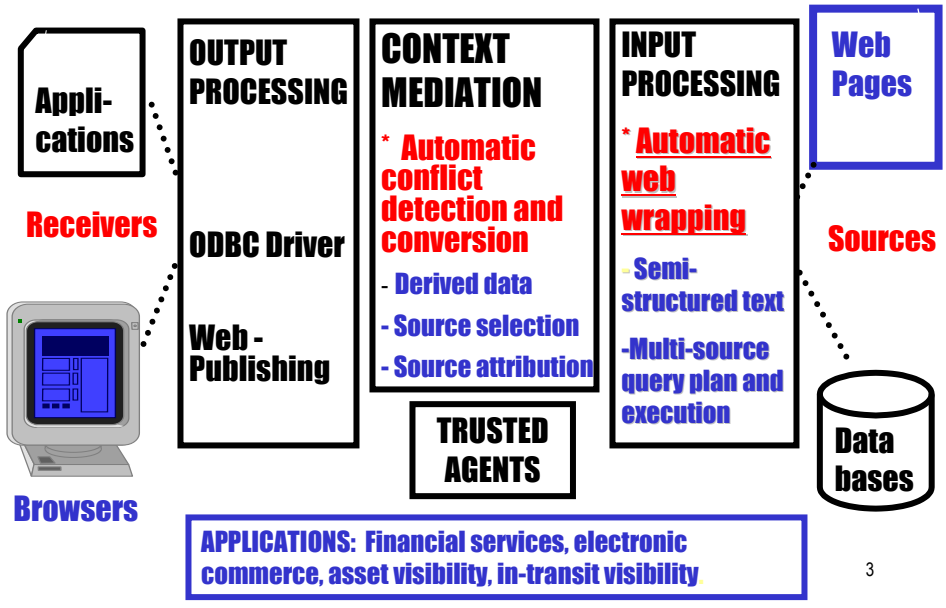
B

Bank Web sites

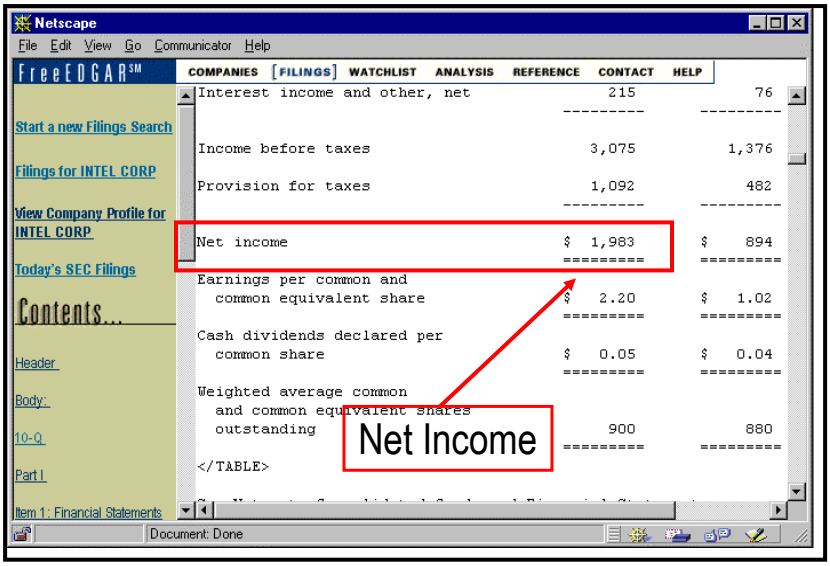
Internal data bases

New Use of the Web: Program intermediaries Focus: **Productivity**

MIT Sloan COntext INterchange (COIN) Project



Example Semi-structured Web data: Intel SEC filing



Sample HTML (excerpt from Edgar web site)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<html>
<head>
<title>Partes&#174; FreeEDGAR: Free Real-Time SEC EDGAR Filings</title>
<BODY topmargin=18 leftmargin=6 bgcolor="#ffffff" link="#0000ee" VLINK="#551A8B" ALINK="#ff0000">
</TABLE>
<pre><font size=2>
                                  1998        1997        1996
                                 -------     -------     -------
NET REVENUES                    $26,273     $25,070     $20,847
                                 -------     -------     -------
Cost of sales                    12,144       9,945       9,164
                                 -------     -------     -------
OPERATING INCOME                  8,379       9,887       7,553
                                 -------     -------     -------
NET INCOME                      $ 6,068     $ 6,945     $ 5,157
</font></pre>
<!-- end company info -->
<!-- begin page footer -->
<table cellpadding=0 cellspacing=0 border=0>
<tr><td align=left valign=middle width=455 nowrap height=20>
<img src="/Global/images/black.gif" height=2 width=455></td></tr>
<tr><td align=left valign=top nowrap width=455>
<font size=1 face="helvetica,arial">
<p>Copyright &#169;1999 Partes Corporation.<br></font></td></tr></table>
<!-- end page footer -->
</td></tr></table>
</BODY>
</HTML>
```

Text

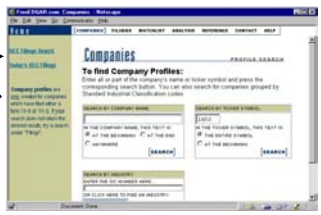
Comments

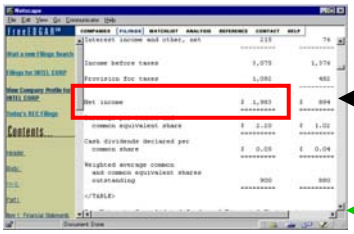
HTML formatting

Home page

Web Navigation

SEC Site

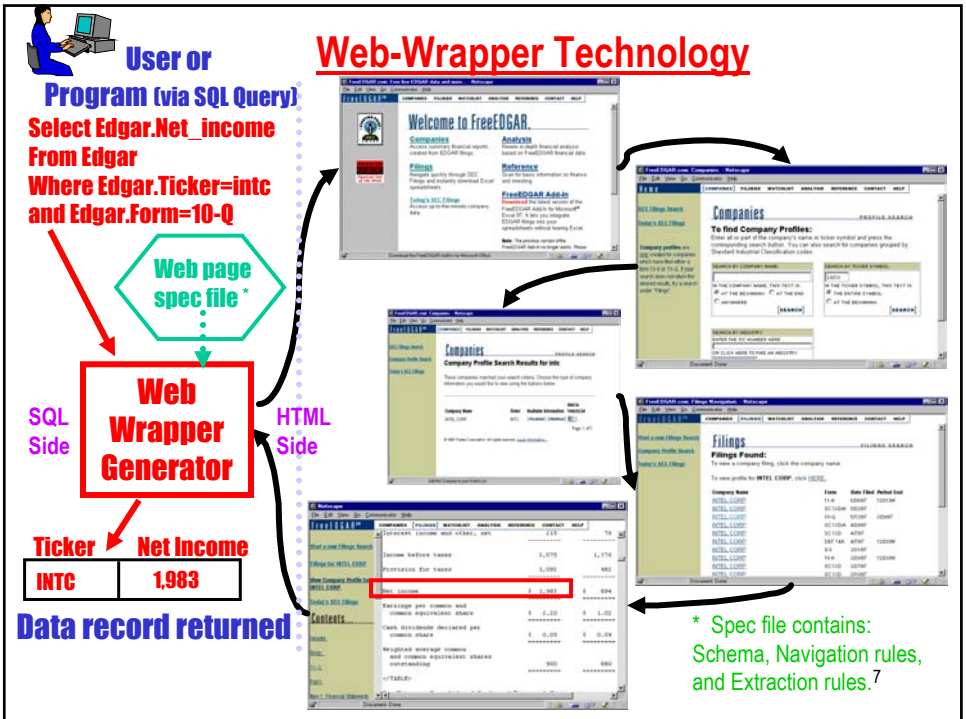




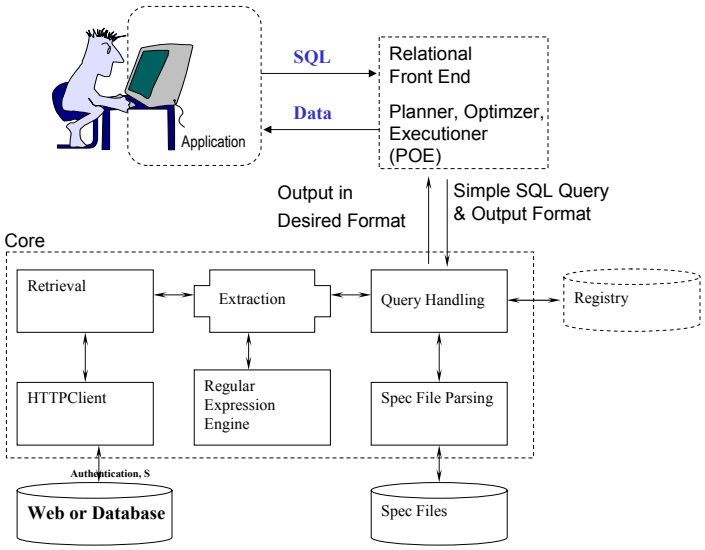
Filing with Intel Net Income

Target page

Web-Wrapper Technology

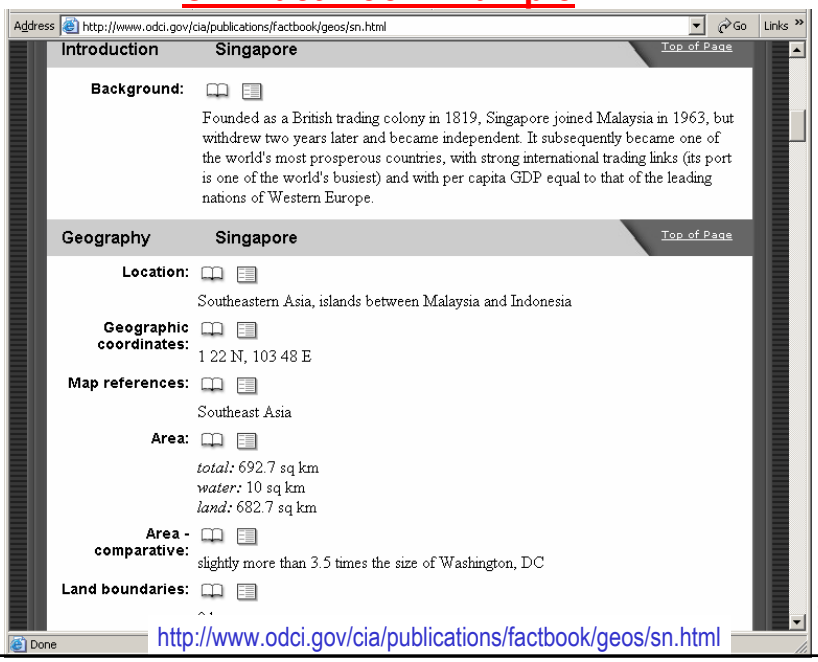


Cameleon Architecture



Demo at context2.mit.edu/coin -> demos -> Cameleon# demo

CIA Fact Book Example



http://www.odci.gov/cia/publications/factbook/geos/sn.html

CAMELEON **QUERY**:

Select capital, location, coordinates, totalarea, climate, population, GDP from cia where Country="Singapore"

CAMELEON **RESULTS**:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <DOCUMENT>
  - <ELEMENT>
      <capital>Singapore</capital>
      <location>Southeastern Asia, islands between Malaysia and
        Indonesia</location>
      <coordinates>1 22 N, 103 48 E</coordinates>
      <totalarea>692.7 sq km</totalarea>
      <climate>tropical; hot, humid, rainy; two distinct monsoon seasons -
        Northeastern monsoon from December to March and Southwestern
        monsoon from June to September; inter-monsoon - frequent afternoon
        and early evening thunderstorms</climate>
      <population>4,452,732 (July 2002 est.)</population>
      <gdp>$106.3 billion (2001 est.)</gdp>
    </ELEMENT>
  </DOCUMENT>
```

Spec file for CIA Fact Book (partial)

```
<?xml version="1.0" encoding="UTF-8" ?>
 <!DOCTYPE RELATION SYSTEM "http://interchange.mit.edu/cameleon_sharp/cameleonspec.dtd">
 <RELATION name="cia">
  <SOURCE URI="http://www.odci.gov/cia/publications/factbook/index.html">
       <ATTRIBUTE name="Link" type="String“ link=“true”>
          <BEGIN> <![CDATA[ <body  ]]>  </BEGIN>
          <PATTERN> <![CDATA[ <option value="([^"]*)"[^>]*>#Country#  ]]>  </PATTERN>
          <END> <![CDATA[ </[Bb][oO][dD][yY]>  ]]>  </END>
       </ATTRIBUTE>
  </SOURCE>
  <SOURCE URI="http://www.odci.gov/cia/publications/factbook/#Link#">
       <ATTRIBUTE name="totalarea" type="String">
          <BEGIN> <![CDATA[ Area:  ]]>  </BEGIN>
          <PATTERN> <![CDATA[ total:.*?</i>\s*([\0-\377]*?)\s*<  ]]>  </PATTERN>
          <END> <![CDATA[ </tr>  ]]>  </END>
        </ATTRIBUTE>
       <ATTRIBUTE name="capital" type="String">
          <BEGIN> <![CDATA[ Capital:  ]]>  </BEGIN>
          <PATTERN> <![CDATA[ <br>\s*([\0-\377]*?)\s*<  ]]>  </PATTERN>
          <END> <![CDATA[ </tr>  ]]>  </END>
       </ATTRIBUTE>
      …
  </SOURCE>
 </RELATION>
```


Regular Expressions Used in Spec Files

- `*` Match 0 or more times (greedy).
- `*?` Match 0 or more times (non-greedy).
- `+` Match 1 or more times (greedy).
- `?` Match 0 or 1 time (greedy).

Greedy quantifiers such as * matches as much as possible, whereas non-greedy quantifiers stop at the minimum match. Example:

 hello lovely   world

(.*) would match ‘hello lovely   world’ whereas

(.*?) would match ‘hello’ and ‘world’

- `.` matches everything except \n
- `[\0-\377]` matches everything
- `^` matches the beginning of a string or line
- `[^ a character]` matches everything except the specified character.
  For instance [^<] matches anything but <
- `$` matches the end of a string or line
- `\s` matches a whitespace character
- `\S` matches a non-whitespace character
- `\d` matches a digit
- Expressions within parentheses are saved.

CIA Example

In source:

```
<br>
        <i>total:</i> 692.7 sq km
<br><i>water:</i> 10 sq km
```

In Spec:

```
<ATTRIBUTE name="totalarea" type="String">
        <BEGIN> <![CDATA[ Area:  ]]>  </BEGIN>
        <PATTERN> <![CDATA[ total:.*?</i>\s*([\0\377]*?)\s*<  ]]> </PATTERN>
        <END> <![CDATA[ </tr>  ]]>  </END>
<ATTRIBUTE>
```

In Result:

```
        <totalarea>692.7 sq km</totalarea>
```

What if we want to get rid of sq km in result?

Instead of matching everything using ([\0-\377]*?), we match digits and dot only like:

```
…
<PATTERN> <![CDATA[total:.*?</i>\s*([\d\.]*?)\s*sq]]> </PATTERN>
…
```


Another Example (from Edgar web site)

<pre>

	1998	1997	1996
NET REVENUES	$26,273	$25,070	$20,847
Cost of sales	12,144	9,945	9,164
OPERATING INCOME	8,379	9,887	7,553
NET INCOME	$ 6,068	$ 6,945	$ 5,157

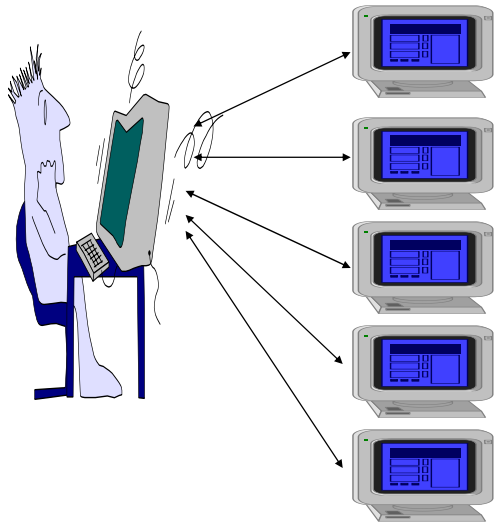
 </pre>

http://soursop.mit.edu/edgar.htm

To extract 1996 Cost of Sales (http://soursop.mit.edu/hzedgar.xml**)**

```
…
<SOURCE URI="http://soursop.mit.edu/edgar.htm">
<ATTRIBUTE name="cost_of_sales_96" type="String">
   <BEGIN> <![CDATA[Cost\s*of\s*sales ]]>  </BEGIN>
   <PATTERN> <![CDATA[\s*[\d,]*?\s+[\d,]*?\s+([\d,]*)\s+]]> </PATTERN>
   <END><![CDATA[OPERATING\s*INCOME]]></END>
</ATTRIBUTE>
…
```

Sample Application:
Providing Integrated Data and Analysis



Equity prices - TIBCO
Real-time feed

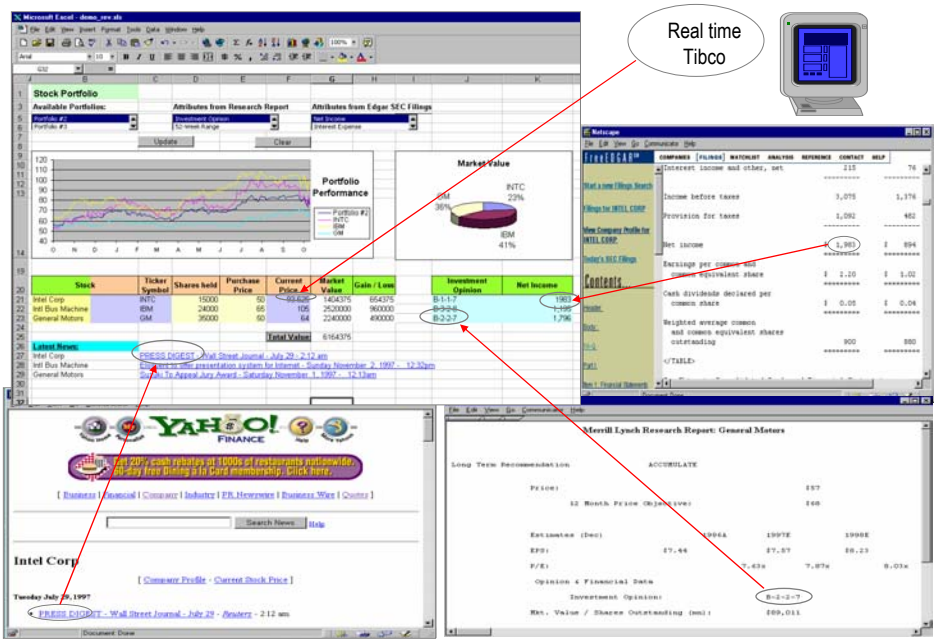
SEC Filings - EDGAR
Web based - Internet

News - Reuters, Newswire and Businesswire
Web based - Internet

Merrill Research Reports
Text based - Merrill Intranet

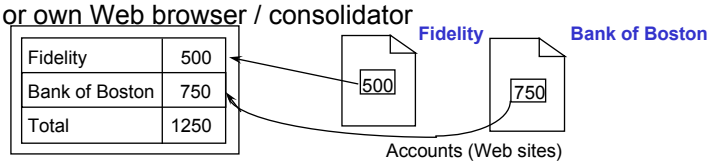
Market updates - Merrill Lynch home page
Web based - Internet

Spreadsheet Interface

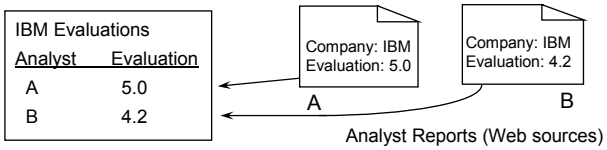


Sample Applications of Semi-Structured Web Data

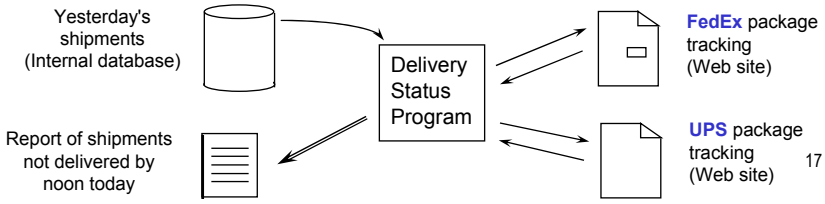
• Automate Extraction of data from specific Web sites into user tool, like Excel, or own Web browser / consolidator



• Automatically Select and Consolidate information across Web sites



• Integrate Internet / Intranet / Client Server networks for internal operations



XML – The Silver Bullet ?

- XML is (according to press reports …)
  - "HTML on steroids" ?
  - "a Rosetta Stone" ?
  - "a universal way to translate data" ?
  - "a miraculous way to" … information integration ?
  - "a silver bullet" ?

XML What is it … really?

- XML - EXtensible Markup Language
- Meta language for defining a markup language
- Based on SGML - Standard Generalized Markup Language
- Data model for syntax for structuring data
- Can define tags at will
- Can nest document structures to arbitrary levels of complexity
- Can use Document Type Definition (DTD)
- Some background sources: w3c.org/XML and XML.org

- Many other members of "family":
  - XSL, XSLT, XLL, XML-Schema, XQuery, etc.

XML Does help create structured Web pages

Feature	HTML	XML
Extensibility	Fixed set of tags	Extensible set of tags
Tag purpose	Presentation	Content
Views	Single	Multiple (XSL)
Orientation	Documents	Documents + semi-structured data
Search	Keyword	Keyword plus Field-sensitive queries

Example: HTML Compared with XML

HTML *

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<html>
<head> . . .
<BODY topmargin=18 leftmargin=6 bgcolor="#ffffff" link="#0000ee" VLINK="#551A8B" ALINK="#ff0000">
<pre><font size=2>
                    Regular   Our
                    Price     Price
      Palm Pilot V  329.00    236.00    In stock
</font></pre>
<table cellpadding=0 cellspacing=0 border=0>
<tr><td align=left valign=middle width=455 nowrap height=20>
<tr><td align=left valign=top nowrap width=455>
<font size=1 face="helvetica,arial"> . . .
</BODY>
</HTML>
```

XSL

XML

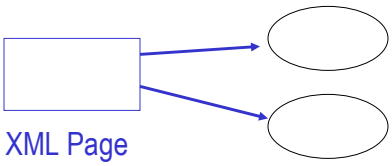
```
<ProductInfo>
   <Product> Palm Pilot V </Product>
   <RegularPrice> 329.00 </RegularPrice>
   <OurPrice> 236.00 </OurPrice>
   <InStock> yes </InStock>
</ProductInfo>
```

* The HTML is normally much messier, with lots more formatting details and <tr> and <td> tags for defining tables and tab positions.

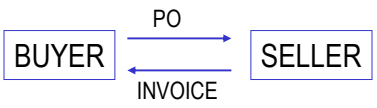
Example: **CIA Factbook in XML at**
http://www.cs.washington.edu/research/xmldatasets/data/mondial/mondial-3.0.xml

XML Why do we need it?

- W3C wanted to get out of the tag creation business
- Separate data from presentation
  - Use of a style sheet instead of "hardcoded" HTML formatting
  - Flexibility / scalability / extensibility



- Also important for Wireless Applications (WML/ XHTML)
- Human readability
- Computer processable
- Information interchange (EDI)



XML . . . Multiple Standards

- What is so great about XML standards – is that there are so many . . .
- Should tag for catalog be called "price" or "cost" ?

"The director of electronic trading at Credit Suisse First Boston and chairman of financial services XML working group is struggling with more than a dozen XML protocols … for financial trading applications." (*ComputerWorld*)

- XML is helpful for integration, but not quite a "silver bullet"
- Much more is needed for information integration
  - ➢ Context Interchange and Semantic Web research are promising areas . . .

Database as XML

RELATIONAL

```
<dataroot>
 <Census>
   <State>Alabama</State>
  <Abbr>AL</Abbr>
  <Year>1999</Year>
  <Population>4370</Population>
  <Cars>3957</Cars>
  <Drivers>3521</Drivers>
 </Census>
 <Census>
  <State>Alabama</State>
  <Abbr>AL</Abbr>
  <Year>2000</Year>
  <Population>4447</Population>
  <Cars>3960</Cars>
  <Drivers>3521</Drivers>
 </Census>
…
</dataroot>
```

NESTED/HIERARCHICAL

```
<dataroot>
 <Census>
  <State>Alabama</State>
  <Abbr>AL</Abbr>
  <Year val="1999">
    <Population>4370</Population>
    <Cars>3957</Cars>
    <Drivers>3521</Drivers>
  </Year>
  <Year val="2000">
    <Population>4447</Population>
    <Cars>3960</Cars>
    <Drivers>3521</Drivers>
  </Year>
 </Census>
 <Census>
  <State>Alaska</State>
  <Abbr>AK</Abbr>
  <Year val="1999">
. . .
</dataroot>
```

NESTED/HIERARCHICAL

```
<dataroot>
 <Census>
   <State>Alabama</State>
   <Abbr>AL</Abbr>
   <Yearrec>
   <Year>1999</Year>
     <Population>4370</Population>
     <Cars>3957</Cars>
     <Drivers>3521</Drivers>
    </Yearrec>
  <Yearrec>
   <Year>2000</Year>
     <Population>4447</Population>
     <Cars>3960</Cars>
     <Drivers>3521</Drivers>
  </Yearrec>
 </Census>
 <Census>
  <State>Alaska</State>
  <Abbr>AK</Abbr>
  <Yearrec>
    <Year>1999</Year>
. . .
</dataroot>
```

SQL vs. XQuery

- **Q: How can you query an XMLdocument database? A: XQuery**

Feature	SQL	XQuery
Language	Declarative	Functional / Procedural
Data model	Relational	XML document: Sequence based, can be relational with arbitrary nesting, hierarchical, etc.
Functions	Built-ins only	Built-ins & user defined
Optimization	Established	Still being researched

Syntax: F-L-W-O-R: For, Let, Where, Order by, Return

XML Primer at http://www.softwareag.com/xml/tools/xquery_primer.pdf

Tutorial slides at http://www.brics.dk/~amoeller/XML/querying/index.html

XQuery by Example (Q1)

Census (relational)

State	Abbr	Year	Population	Cars	Drivers
Alabama	AL	1999	4370	3957	3446
Alabama	AL	2000	4447	3960	3521
…					
Wyoming	WY	2000	494	586	371

Census.xml

```
<?xml version="1.0"
encoding="UTF-8"?>
<dataroot>
  <Census>
    <State>Alabama</State>
    <Abbr>AL</Abbr>
    <Year>1999</Year>
    <Population>4370</Population>
    <Cars>3957</Cars>
    <Drivers>3521</Drivers>
  </Census>
 <Census>
    <State>Alabama</State>
    <Abbr>AL</Abbr>
    <Year>2000</Year>
    <Population>4447</Population>
    <Cars>3960</Cars>
    <Drivers>3521</Drivers>
  </Census>
…
<dataroot>
```

- Q1: population of Massachusetts
  - In SQL:

```
Select Year, Population
From Census
Where State="Massachusetts";
```

  - In XQuery

```
for $s in doc("census.xml")
//Census[State = "Massachusetts"]
return <Mass_pop> {$s/Year, $s/Population} </Mass_pop>
```

XQuery by Example (Q6)

- **Q6: Names of all students from states with more than 9 million people in 2000**
  - In SQL:

```
select student.name
from  student, census
where student.home = census.state
          and  census.year = 2000
          and  census.population > 9000
```

  - In XQuery

```
for $s in doc("census.xml")//Census[Year=2000 and
Population>9000]
let $t := doc("student.xml")/*/Student[Home=$s/State]
return $t/Name
```


XQuery by Example (Q7)

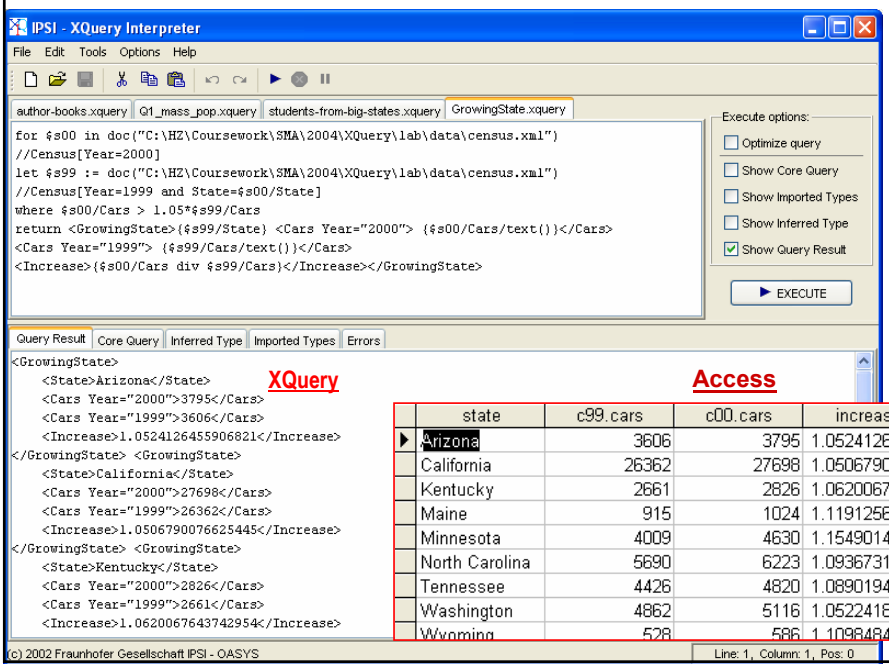
- **Q7. Names of all states where number of cars increased by over 5% between 1999 and 2000**
  - In SQL:

```
Select c00.state, c99.cars, c00.cars, c00.cars/c99.cars as
increase
From census c00, census c99
Where c00.year=2000 and c99.year=1999
And c00.cars > 1.05 * c99.cars
And c00.state = c99.state;
```

  - In XQuery

```
for $s00 in doc("census.xml")//Census[Year=2000]
let $s99 := doc("census.xml")//Census[Year=1999 and
State=$s00/State]
where $s00/Cars > 1.05*$s99/Cars
return <GrowingState>{$s99/State} <Cars Year="2000">
{$s00/Cars/text()}</Cars>
<Cars Year="1999"> {$s99/Cars/text()}</Cars>
<Increase>{$s00/Cars div $s99/Cars}</Increase></GrowingState>
```

Results of Q7



Matching (Regular Expressions) in XQuery

```
let $ss := doc("C:\HZ\Coursework\SMA\2004\XQuery\lab\data\census.xml")
//Census[matches(./State/text(), ".*ss.*")]
for $state in distinct-values($ss/State)
return <State name="{$state}">
{for $y in $ss/Year[../State=$state]
  order by $y descending  return <Year
val="{$y}">
{$ss/Population[(../State=$state and ../Year=$y)]}</Year>}
</State>
```

- Finds all states that have "ss" anywhere in their name …

Resutls:

```
<State name="Massachusetts">
  <Year val="2000">
    <Population>6349</Population>
  </Year>
  <Year val="1999">
    <Population>6175</Population>
  </Year>
  <Year val="1998">
    <Population>6144</Population>
  </Year>
</State> <State name="Mississippi">
  <Year val="2000">
    <Population>2845</Population>
  </Year>
  <Year val="1999">
    <Population>2769</Population>
  </Year>
  <Year val="1998">
    <Population>2751</Population>
  </Year>
  . . .
```

Cameleon and XQuery

- Can use Cameleon web wrapping to dynamically convert HTML web site to appear to be an XMLdocument, then use XQuery to query that document
- Benefits can perform "Joins" between XML and HTML documents.
- Example: Query exchange rate between USD and EUR on 7/4/04:

```
let $d :=
doc("http://interchange.mit.edu/cameleon_sharp/camserv.aspx?query=Select+exchanged%2C+expr
essed%2C+date%2C+rate+From+hzoanda+where+date%3D%2207%2F04%2F2004%22+and+exc
hanged%3D%22USD%22+and+expressed%3D%22EUR%22&amp;format=xml")
return $d//ELEMENT/*
```

Results:

```
<exchanged>USD</exchanged> <expressed>EUR</expressed> <date>07/04/2004</date>
<rate>0.81261</rate>
```

- Of course, this can also be done using Cameleon directly using SQL. (which allows joins among HTML, XML, and relation databases)

Summary

- **Tim Berners-Lee**, W3C Director:
  - "The Web is quickly becoming the world's fastest growing repository of data"
- In past: Primarily processed by humans.
- In future, must be processible by programs (agents for humans)
- Tools, such as MIT's Automatic Web Wrapper and W3C's XML and XQuery, are providing these capabilities.

Work reported herein has been supported, in part, by the USA Advanced Research Projects Agency, Banco Santander Central Hispano, Citibank, Fleet Bank, FirstLogic, Merrill Lynch, PricewaterhouseCoopers, MIT Total Data Quality Management Program, MIT Center for eBusiness, Singapore-MIT Alliance, Suruga Bank, and USAF/Rome Laboratory.