

SMA 5506 Database Technology

Summer 2005

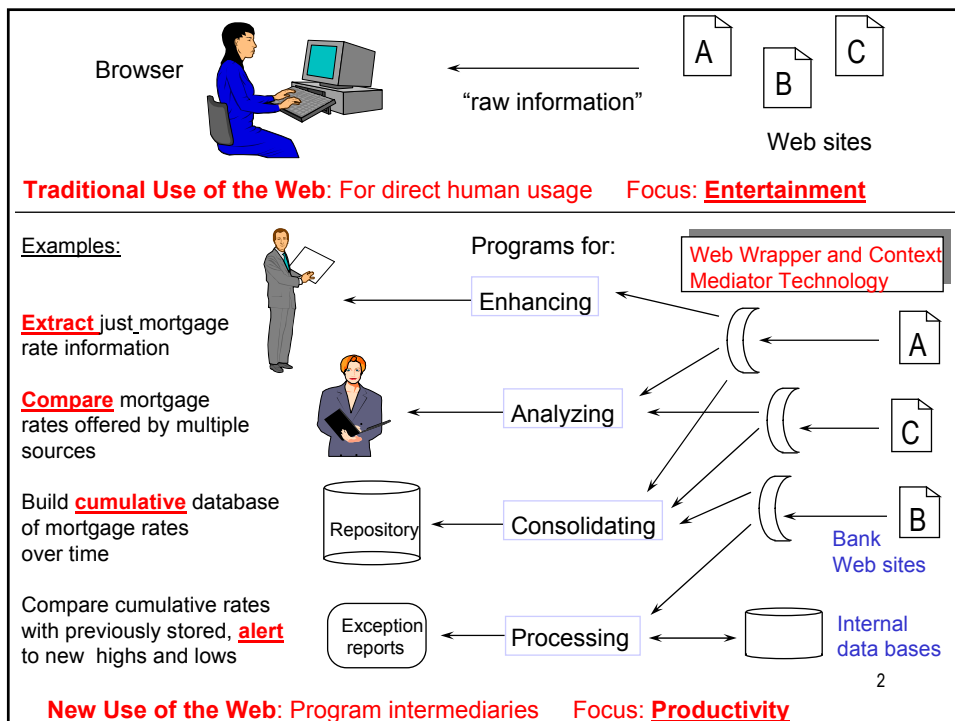
Session L16

WEB AS A DATABASE: EVOLUTION OF XML

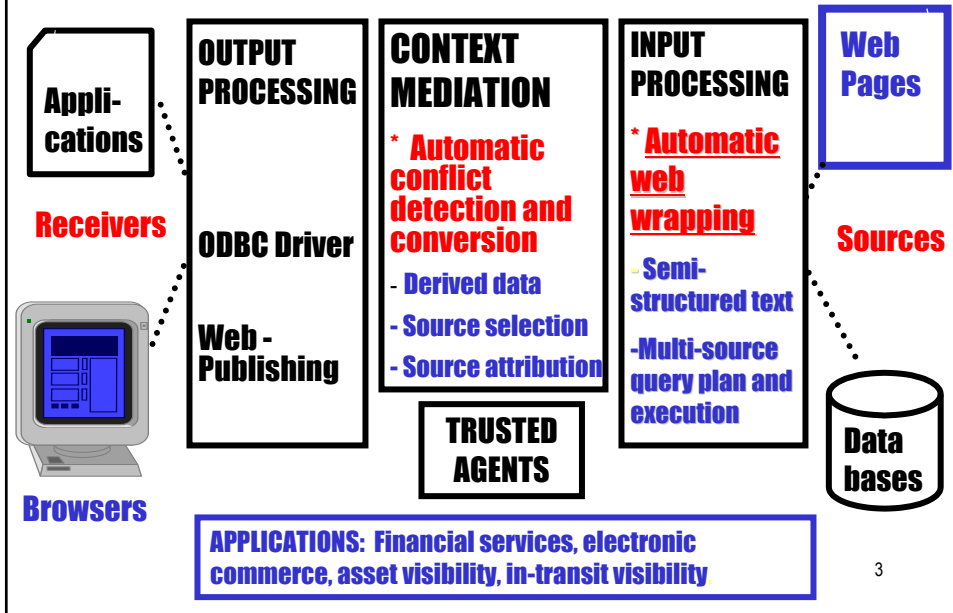
Prof. Stuart Madnick
(smadnick@mit.edu)

© S. Madnick, 2005 (v01)

1



MIT Sloan COntext INterchange (COIN) Project



3

Example Semi-structured Web data: Intel SEC filing

	REFERENCE	CONTACT	HELP
Interest income and other, net	215	76	
Income before taxes	3,075	1,376	
Provision for taxes	1,092	482	
Net income	\$ 1,983	\$ 894	
Earnings per common and common equivalent share	\$ 2.20	\$ 1.02	
Cash dividends declared per common share	\$ 0.05	\$ 0.04	
Weighted average common and common equivalent shares outstanding	900	880	

Net Income

4

Sample HTML (excerpt from Edgar web site)

```

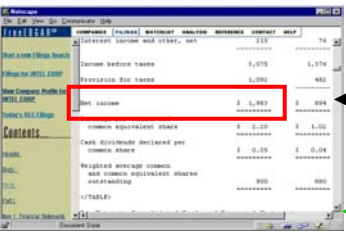
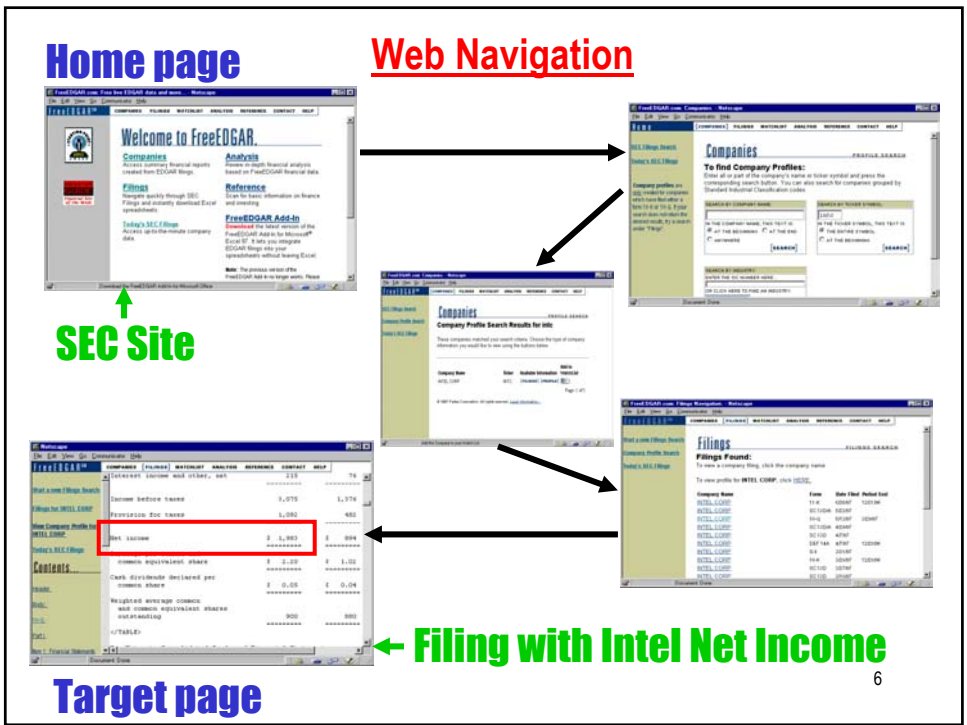
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<html>
<head>
<title>Partes&#174; FreeEDGAR: Free Real-Time SEC EDGAR Filings</title>
<BODY topmargin=18 leftmargin=6 bgcolor="#ffffff" link="#0000ee" vLINK="#551A8B" ALINK="#ff0000">
</TABLE>
<pre><font size=2>
          1998          1997          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>
</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>

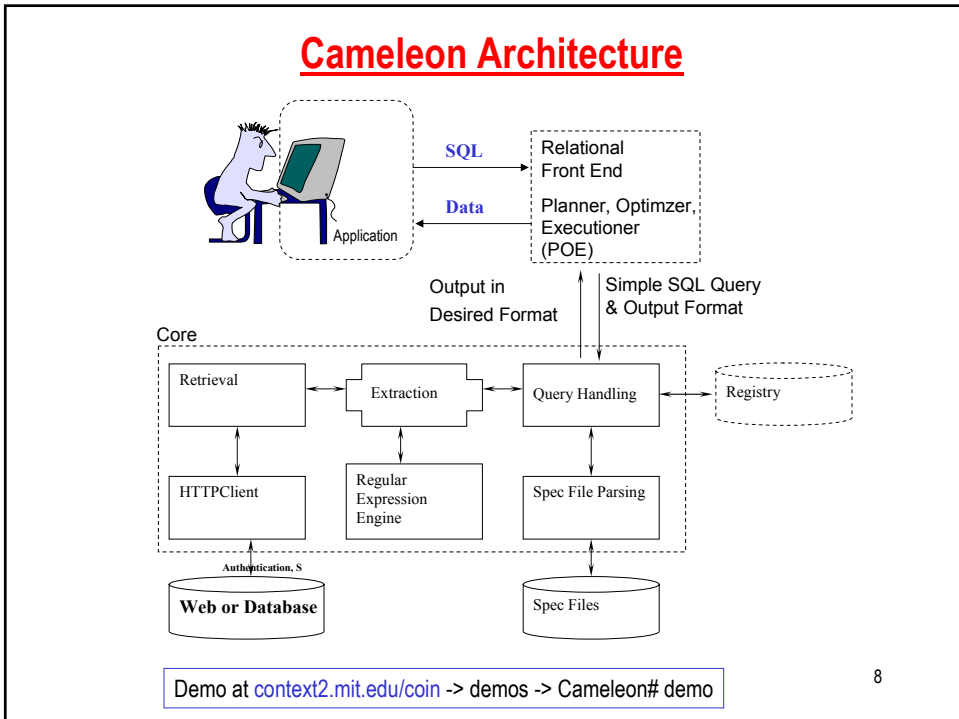
```

Text

Comments

HTML formatting





CIA Fact Book Example

The screenshot shows a web browser window displaying the CIA Fact Book entry for Singapore. The browser's address bar shows the URL: <http://www.odci.gov/cia/publications/factbook/geos/sn.html>. The page is titled "Introduction Singapore" and "Geography Singapore".

Background: Founded as a British trading colony in 1819, Singapore joined Malaysia in 1963, but withdrew two years later and became independent. It subsequently became one of the world's most prosperous countries, with strong international trading links (its port is one of the world's busiest) and with per capita GDP equal to that of the leading nations of Western Europe.

Geography:

- Location:** Southeastern Asia, islands between Malaysia and Indonesia
- Geographic coordinates:** 1 22 N, 103 48 E
- Map references:** Southeast Asia
- Area:**
 - total: 692.7 sq km
 - water: 10 sq km
 - land: 682.7 sq km
- Area - comparative:** slightly more than 3.5 times the size of Washington, DC
- Land boundaries:**

The browser's status bar at the bottom shows the URL: <http://www.odci.gov/cia/publications/factbook/geos/sn.html>. A small number "9" is visible in the bottom right corner of the browser window.

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/comeleon_sharp/comeleonspec.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>
```

11

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:

```
<b> hello </b> <i> lovely </i> <b> world </b>
<b>(.*?)</b> would match 'hello </b> <i> lovely </i> <b> world' whereas
<b>(.*?)</b> 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.

12

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*(\d|\d\d|\d\d\d)*|^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 `(\d|\d\d|\d\d\d)*?`, we match digits and dot only like:

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

13

Another Example (from Edgar web site)

```
<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>
```

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

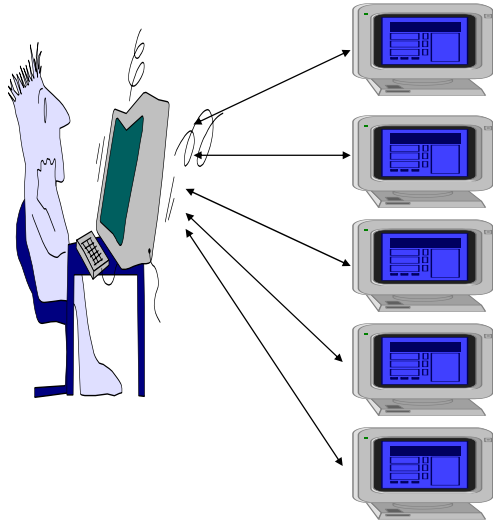
To extract 1996 Cost of Sales (<http://soursop.mit.edu/hzedgar.xml>)

```
...
<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|\d\d|\d\d\d)*|^s*(\d|\d\d|\d\d\d)*|^s*]]> </PATTERN>
  <END><![CDATA[OPERATING\s*INCOME]]></END>
</ATTRIBUTE>
```

...

14

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

Microsoft Excel - demo.xls

Stock	Ticker	Shares held	Purchase Price	Current Price	Market Value	Gain / Loss	Investment Option	Net Income
Intel Corp	INTC	15000	50	62.625	144375	654375	8-1-17	1985
Int Bus Machine	IBM	24000	66	106	252000	900000	None	795
General Motors	GM	90000	50	64	2240000	480000	8-2-27	1,796

Market Value

IBM	38%
INTC	22%
GM	41%

Real time Tibco

Merrill Lynch Research Report: General Motors

Long Term Recommendation: **ACCELERATE**

Price:	157
62 Month Price Objective:	168

Estimated (Buc):

1994	1997E	1998E	
EPS	\$7.66	\$7.57	\$8.23
P/E	7.43x	7.97x	9.03x

Opinion & Financial Data

Investment Opinion: **8-2-27**

Est. Value / Shares Outstanding (mil): 209,011

Yahoo! Finance

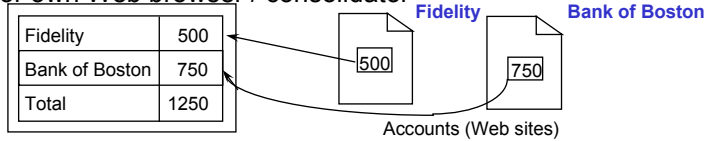
Intel Corp

Tuesday July 6, 1997

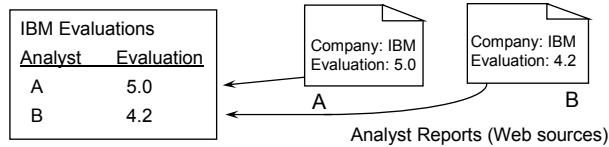
Price: **50.00** - Wall Street Journal, July 23 - Reuters - 2:12 am

Sample Applications of Semi-Structured Web Data

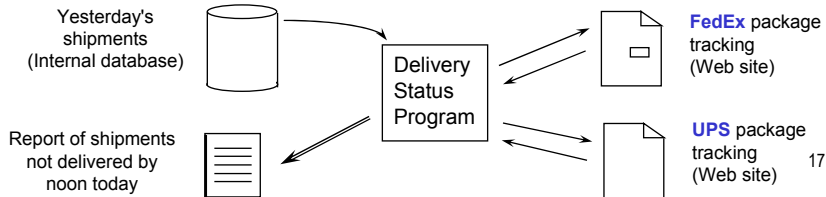
- Automate **Extraction** of data from specific Web sites into user tool, like Excel, or own Web browser / consolidator



- 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.

19

XML Does help create structured Web pages

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

20

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 Price	Our 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>
```

```
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.

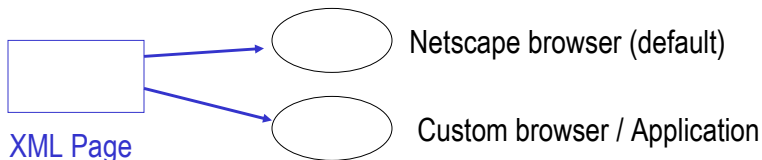
Example: CIA Factbook in XML at

<http://www.cs.washington.edu/research/xmldatasets/data/mondial/mondial-3.0.xml>

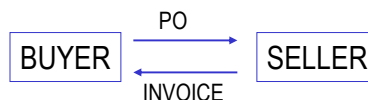
21

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)



22

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 . . .

23

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">
      ...
    </Year>
  </Census>
  ...
</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>
      ...
    </Yearrec>
  </Census>
  ...
</dataroot>
  
```

24

SQL vs. XQuery

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

<u>Feature</u>	<u>SQL</u>	<u>XQuery</u>
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>

25

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>
```

26

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
```

27

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>
```

28

Results of Q7

The screenshot shows the XQuery Interpreter interface. The query in the main window is:

```

author-books.xquery  G1_mass_pop.xquery  students-from-big-states.xquery  GrowingState.xquery
for $s00 in doc("C:\HZ\Coursework\SMA\2004\XQuery\lab\data\census.xml")
//Census[Year=2000]
let $s99 := doc("C:\HZ\Coursework\SMA\2004\XQuery\lab\data\census.xml")
//Census[Year=1999 and State=$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>
    
```

The results are shown in two parts:

XQuery (XML tree view):

```

<GrowingState>
  <State>Arizona</State>
  <Cars Year="2000">3795</Cars>
  <Cars Year="1999">3606</Cars>
  <Increase>1.0524126455906821</Increase>
</GrowingState> <GrowingState>
  <State>California</State>
  <Cars Year="2000">27698</Cars>
  <Cars Year="1999">26362</Cars>
  <Increase>1.0506790076625445</Increase>
</GrowingState> <GrowingState>
  <State>Kentucky</State>
  <Cars Year="2000">2826</Cars>
  <Cars Year="1999">2661</Cars>
  <Increase>1.06200676437</Increase>
</GrowingState> <GrowingState>
  <State>Maine</State>
  <Cars Year="2000">1024</Cars>
  <Cars Year="1999">915</Cars>
  <Increase>1.11912568306</Increase>
</GrowingState> <GrowingState>
  <State>Minnesota</State>
  <Cars Year="2000">4630</Cars>
  <Cars Year="1999">4009</Cars>
  <Increase>1.15490147169</Increase>
</GrowingState> <GrowingState>
  <State>North Carolina</State>
  <Cars Year="2000">6223</Cars>
  <Cars Year="1999">5690</Cars>
  <Increase>1.09367311072</Increase>
</GrowingState> <GrowingState>
  <State>Tennessee</State>
  <Cars Year="2000">4820</Cars>
  <Cars Year="1999">4426</Cars>
  <Increase>1.08901943064</Increase>
</GrowingState> <GrowingState>
  <State>Washington</State>
  <Cars Year="2000">5116</Cars>
  <Cars Year="1999">4862</Cars>
  <Increase>1.05224187577</Increase>
</GrowingState> <GrowingState>
  <State>Wyoming</State>
  <Cars Year="2000">586</Cars>
  <Cars Year="1999">528</Cars>
  <Increase>1.10984848485</Increase>
    
```

Access (Table view):

state	c99_cars	c00_cars	increase
Arizona	3606	3795	1.05241264559
California	26362	27698	1.05067900766
Kentucky	2661	2826	1.06200676437
Maine	915	1024	1.11912568306
Minnesota	4009	4630	1.15490147169
North Carolina	5690	6223	1.09367311072
Tennessee	4426	4820	1.08901943064
Washington	4862	5116	1.05224187577
Wyoming	528	586	1.10984848485

Footer: © 2002 Fraunhofer Gesellschaft IPSI - OASYS | Line: 1, Column: 1, Pos: 0

Matching (Regular Expressions) in XQuery

```
let $sss := doc("C:\HZ\Coursework\SMA\2004\XQuery\lab\data\census.xml")
```

```
//Census[matches(./State/text(), ".*ss.*")]
```

```
for $state in distinct-values($sss/State)
```

```
return <State name="{ $state }">
```

```
{for $y in $sss/Year[./State=$state]
```

```
order by $y descending return <Year
```

```
val="{ $y }">
```

```
{ $sss/Population[ (./State=$state and ./Year=$y) ] }</Year>
```

```
</State>
```

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

Results:

```

<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 XML document, 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+expressed%2C+date%2C+rate+From+hzoanda+where+date%3D%2207%2F04%2F2004%22+and+exchanged%3D%22USD%22+and+expressed%3D%22EUR%22&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)

31

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.

32