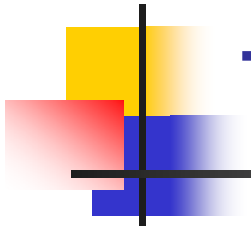
COMP4317: XML & Database
Tutorial 4: XML => Database

Week 5

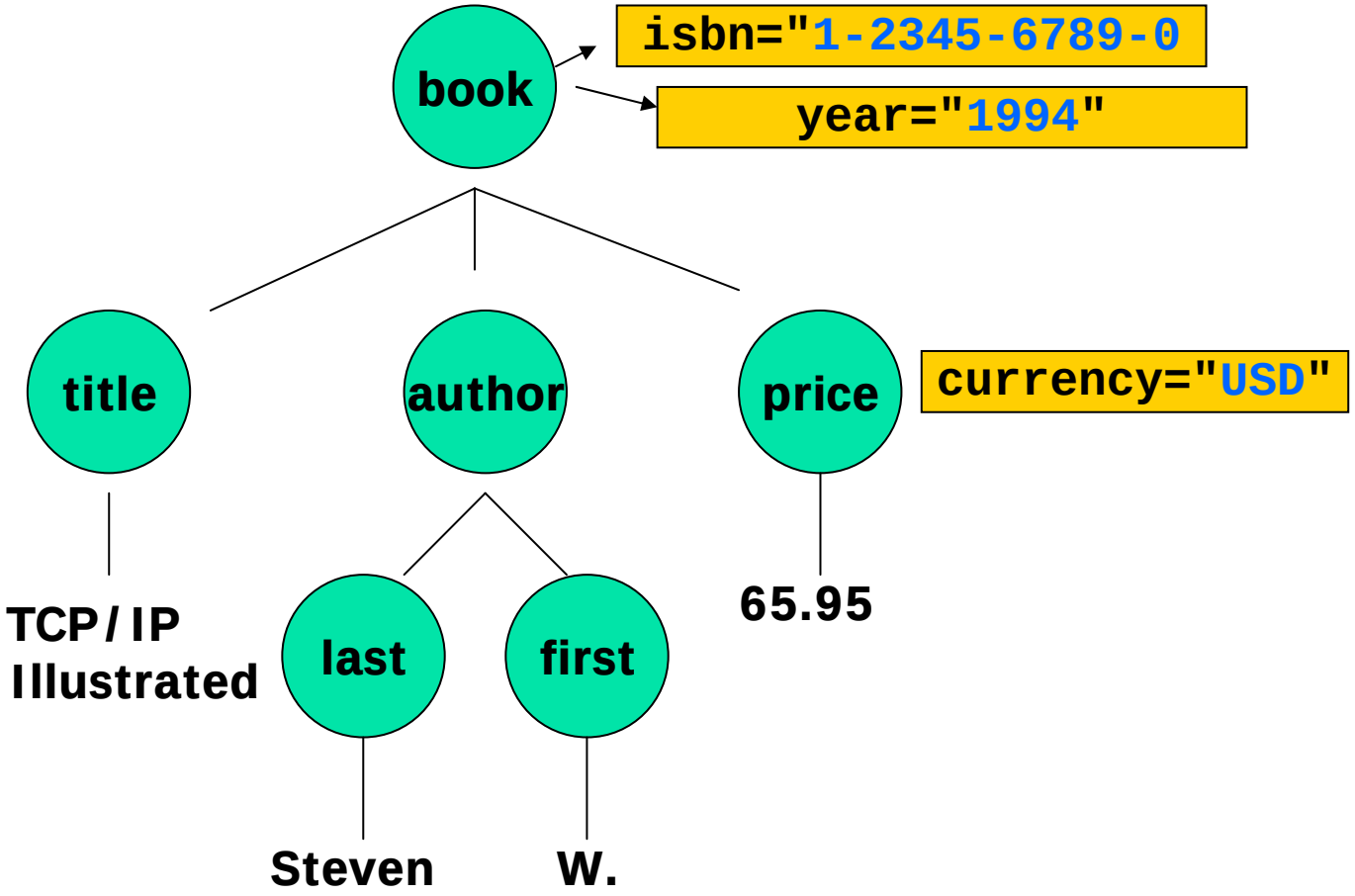
Thang Bui @ CSE.UNSW



Tree & Tables

- Two options:
  - A tree -> one table
    - tree_tbl(pre, post, tag, text, attributes)
  - A tree -> two table
    - tree_tbl(pre, post, tag, text)
    - attr_tbl(pre, attr_name, attr_value)

Tree & Tables


book
isbn="1-2345-6789-0
year="1994"
title
author
price
currency="USD"
TCP/IP
Illustrated
last
first
65.95
Steven
W.

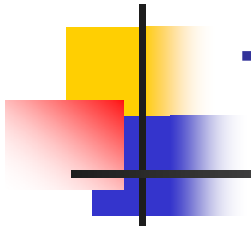
Tree & Tables

pre	post	tag	text	attrs
1	10	book	null	(<isbn,"1-2345-6789-0“>, <year,"1994“>)
2	2	title	null	null
3	1	null	ICP/IP Illustrated	null
4	7	author	null	null
5	4	last	null	null
6	3	null	Steven	null
7	6	first	null	null
8	5	null	W.	null
9	9	price	null	(<currency,"USD">)
10	8	null	65.95	null

Tree & Tables

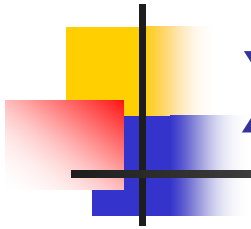
pre	post	tag	text
1	10	book	null
2	2	title	null
3	1	null	ICP/IP Illustrated
4	7	author	null
5	4	last	null
6	3	null	Steven
7	6	first	null
8	5	null	W.
9	9	price	null
10	8	null	65.95

pre	attr_name	attr_value
1	isbn	1-2345-6789-0
1	year	1994
9	currency	USD



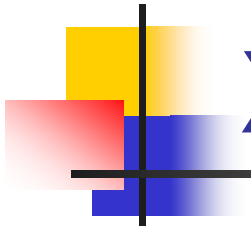
Tree -> Tables: simple

- Travel pre-order
- Travel post-order
- Print out the tree (as tables)



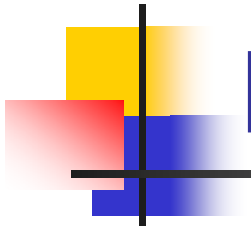
XML -> Tables

- Using DOM/SAX
  - Get the DOM tree (by DOM parser)
  - OR Build the tree from SAX
  - Travel the tree
- Using SAX
  - Build the tables on-the-fly?



XML -> Tables

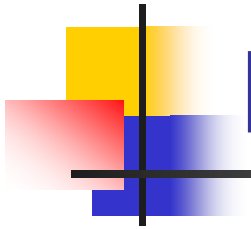
- Using DOM/SAX
  - Get the DOM tree (by DOM parser)
  - OR Build the tree from SAX
  - Travel the tree
- Using SAX
  - Build the tables on-the-fly?
    - startElement: pre-order ID
    - endElement: post-order ID
    - Assign all pre/post-order IDs before printing the table



Example

	Events	Table content

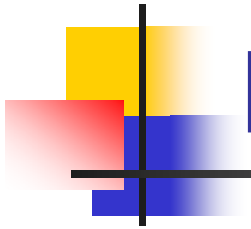
<book>
  <title>
  TCP/IP Illustrated
  </title>
  <author>
  <first>
  Stevens
  </first>
  <last>
  W.
  </last>
  </author>
</book>
	1. start: “book”	book_tbl={<pre,post,tag,text>}
		r1: <1,?,"book",null>



Example

	Events	Table content

<book>
  <title>
  TCP/IP Illustrated
  </title>
  <author>
  <first>
  Stevens
  </first>
  <last>
  W.
  </last>
  </author>
</book>
	1. start: “book”
	2. start: “title”
		book_tbl={<pre,post,tag,text>}
		r1: <1,?,"book",null>
		r2: <2,?,"title",null>

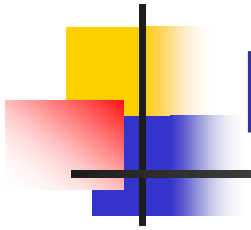


Example

	Events	Table content

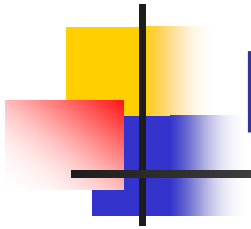
<book>
  <title>
  TCP/IP Illustrated
  </title>
  <author>
  <first>
  Stevens
  </first>
  <last>
  W.
  </last>
  </author>
</book>
	1. start: “book”
	2. start: “title”
	3. character:
		book_tbl={<pre,post,tag,text>}
		r1: <1,?,"book",null>
		r2: <2,?,"title",null>
		r3: <3,1,null,"TCP/IP...">

Example



	Events	Table content

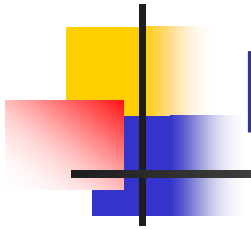
<book>
  <title>
  TCP/IP Illustrated
  </title>
  <author>
  <first>
  Stevens
  </first>
  <last>
  W.
  </last>
  </author>
</book>
	1. start: “book”
	2. start: “title”
	3. character:
	4. end: “title”
		book_tbl={<pre,post,tag,text>}
		r1: <1,?,"book",null>
		r2: <2,2,"title",null>
		r3: <3,1,null,"TCP/IP...">



Example

	Events	Table content

<book>
  <title>
  TCP/IP Illustrated
  </title>
  <author>
  <first>
  Stevens
  </first>
  <last>
  W.
  </last>
  </author>
</book>
	1. start: “book”
	2. start: “title”
	3. character:
	4. end: “title”
	5. start: “author”
		book_tbl={<pre,post,tag,text>}
		r1: <1,?,"book",null>
		r2: <2,2,"title",null>
		r3: <3,1,null,"TCP/IP...”>
		r4: <4,?,"author",null>



Example

	Events	Table content

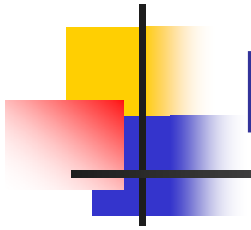
<book>
  <title>
  TCP/IP Illustrated
  </title>
  <author>
  <first>
  Stevens
  </first>
  <last>
  W.
  </last>
  </author>
</book>
	1. start: “book”
	2. start: “title”
	3. character:
	4. end: “title”
	5. start: “author”
	6. start: “first”
		book_tbl={<pre,post,tag,text>}
		r1: <1,?,"book",null>
		r2: <2,2,"title",null>
		r3: <3,1,null,"TCP/IP...”>
		r4: <4,?,"author",null>
		r6: <5,?,"first",null>

Example

	Events	Table content

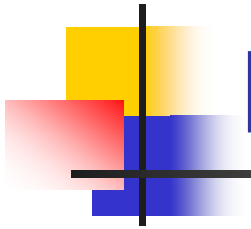
<book>
  <title>
  TCP/IP Illustrated
  </title>
  <author>
  <first>
  Stevens
  </first>
  <last>
  W.
  </last>
  </author>
</book>
	1. start: “book”
	2. start: “title”
	3. character:
	4. end: “title”
	5. start: “author”
	6. start: “first”
	7. character:
		book_tbl={<pre,post,tag,text>}
		r1: <1,?,"book",null>
		r2: <2,2,"title",null>
		r3: <3,1,null,"TCP/IP...">
		r4: <4,?,"author",null>
		r6: <5,?,"first",null>
		r7: <6,3,null,“Stevens">

Example



	Events	Table content

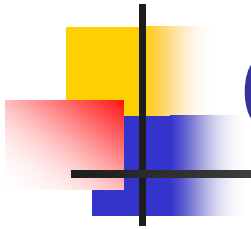
<book>
  <title>
  TCP/IP Illustrated
  </title>
  <author>
  <first>
  Stevens
  </first>
  <last>
  W.
  </last>
  </author>
</book>
	1. start: “book”
	2. start: “title”
	3. character:
	4. end: “title”
	5. start: “author”
	6. start: “first”
	7. character:
	8. end: “first”
		book_tbl={<pre,post,tag,text>}
		r1: <1,?,"book",null>
		r2: <2,2,"title",null>
		r3: <3,1,null,"TCP/IP...">
		r4: <4,?,"author",null>
		r6: <5,4,"first",null>
		r7: <6,3,null,“Stevens">



Example

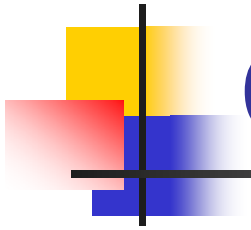
	Events	Table content

<book>
  <title>
  TCP/IP Illustrated
  </title>
  <author>
  <first>
  Stevens
  </first>
  <last>
  W.
  </last>
  </author>
</book>
	1. start: “book”
	2. start: “title”
	3. character:
	4. end: “title”
	5. start: “author”
	6. start: “first”
	7. character:
	8. end: “first”
	9. start: “last”
	10. character:
	11. end: “last”
	12. end: “author”
	13. end: “book”
		book_tbl={<pre,post,tag,text>}
		r1: <1,8,"book",null>
		r2: <2,2,"title",null>
		r3: <3,1,null,"TCP/IP...”>
		r4: <4,7,"author",null>
		r6: <5,4,"first",null>
		r7: <6,3,null,"Stevens”>
		r8: <7,6,"last",null>
		r9: <8,5,null,"W.”>



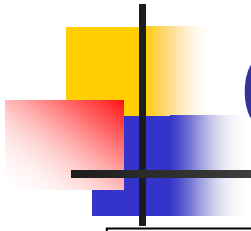
Connect to Database

- Open a connection to Database server
  - JDBC: Java Database Connectivity (for Java)
  - ODBC: Open Database Connectivity (for C++)
- Execute commands:
  - SQL statements
  - ...



Connect to MySQL

- Need MySQL Connector/J
  - /usr/local /java/jdk-1.5.0_07/jre/lib/mysql-connector-java-3.1.13-bin.jar
- or MySQL Connector/ODBC driver
  - /usr/lib/libmysqlclient.a
- Java: Database URL
  - jdbc:mysql://host_name:port/dbname



Connect to MySQL

Java code

```
import java.sql.*;
class Tester {
 public static void main() {
  Connection conn = null;
  try {
   String url = "jdbc:mysql://cs4317.srvr/student";
   Class.forName ("com.mysql.jdbc.Driver").newInstance ();
   conn = DriverManager.getConnection (url, "stu1", “pass1");
  } catch (Exception e){ }
  if (conn != null) {
   try { conn.close ();
   } catch (Exception e) {}
  }
 }
}
```

Connect to MySQL

Java code

```
import java.sql.*;
class Tester {
 public static void main() {
  Connection conn = null;
  try {
   String url = "jdbc:mysql://cs4317.srvr/studentdb1";
   Class.forName ("com.mysql.jdbc.Driver").newInstance ();
   conn = DriverManager.getConnection (url, "stu1", "stu1pass");
  } catch (Exception e){ }
  if (conn != null) {
   try { conn.close ();
   } catch (Exception e) {}
  }
 }
}
```

servername

database name

username

password

```
$export CLASSPATH=".:/usr/share/java/xercesImpl.jar:/usr/local
/java/jdk-1.5.0_07/jre/lib/mysql-connector-java-3.1.13-bin.jar"
```

Connect to MySQL

C++ code

```
#include <stdio.h>
#include <mysql/mysql.h>

void main() {
 MYSQL mysql;
 mysql_init(&mysql);
 if (!mysql_real_connect(&mysql,
        "cs4317.srvr","stu2","stu2pass","studentdb2",0,NULL,0))
  printf("Error!");
 mysql_close(&mysql);
}
```

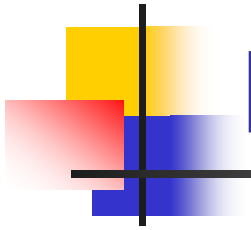
database name

servername

username

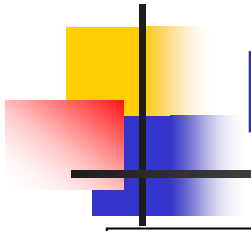
password

$g++ –lmysqlclient -lz tester.cpp



Manipulate tables in Java

- Statement
  - ResultSet executeQuery(String sql)
  - int executeUpdate(String sql)
- ResultSet
  - boolean next()
  - String getString(int columnIndex)
  - int getInt(int columnIndex)
  - ...



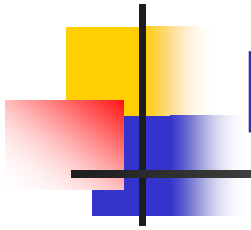
Manipulate tables in Java

```
try {
  Statement stmt = conn.createStatement();

  stmt.executeUpdate("UPDATE table1 SET column1 = \"new value\"");

  ResultSet rs = stmt.executeQuery("SELECT * FROM table1");
  int maxCols = rs.getMetaData.getColumnCount();

  while (rs.next()) {
    for (int i = 1; i <= maxCols; i++)
      System.out.print(rs.getString(i) + "\t");
    System.out.println();
  }
} catch (Exception e){ }
```

Manipulate tables in Java

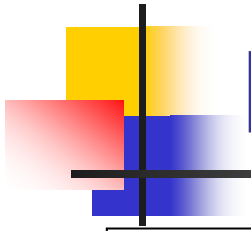
```
try {
  Statement stmt = conn.createStatement();

  stmt.executeUpdate("UPDATE table1 SET column1 = \"new value\"");

  ResultSet rs = stmt.executeQuery("SELECT * FROM table1");
  int maxCols = rs.getMetaData.getColumnCount();

  while (rs.next()) {
    for (int i = 1; i <= maxCols; i++)
      System.out.print(rs.getString(i) + "\t");
    System.out.println();
  }
} catch (Exception e){ }
```

Column index starts from 1



Manipulate tables in C++

- int **mysql_real_query**
  - (MYSQL *mysql, const char * stmt, unsigned long length)
- int **mysql_field_count**
  - (MYSQL *mysql)
- MYSQL_RES ***mysql_store_result**
  - (MYSQL *mysql)
- MYSQL_ROW **mysql_fetch_row**
  - (MYSQL_RES *result)



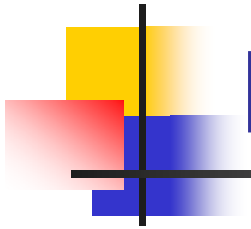
Manipulate tables in C++

```
mysql_query(&mysql,"SELECT * from table1");
MYSQL_RES * res = mysql_store_result(&mysql);

MYSQL_ROW row;
unsigned int fields_count = mysql_field_count(&mysql);

while ((row = mysql_fetch_row(res))) {
  for (int i = 0; i < fields_count; i++) {
     cout << row[i] ? row[i] : "NULL";
  }
  cout << "\n";
}
mysql_free_result(res);
```

Manipulate tables in C++

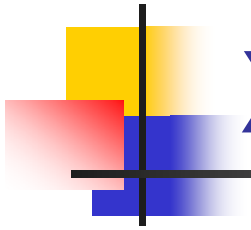
```
mysql_query(&mysql,"SELECT * from table1");
MYSQL_RES * res = mysql_store_result(&mysql);

MYSQL_ROW row;
unsigned int fields_count = mysql_field_count(&mysql);

while ((row = mysql_fetch_row(res))) {
  for (int i = 0; i < fields_count; i++) {
     cout << row[i] ? row[i] : "NULL";
  }
  cout << "\n";
}
mysql_free_result(res);
```

Column index starts from 0



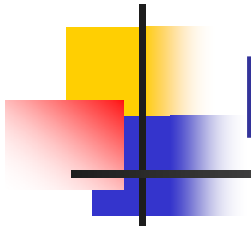
Resource for MySQL

- http://dev.mysql.com/doc/refman/4.1/en/



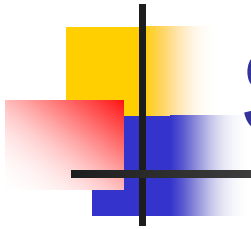
XML -> MySQL

- Fetch XML into (in-memory) tables
- Open a connection to MySQL server
- Call “CREATE TABLE” statement(s) to setup (some) table(s)
- Travel the (in-memory) tables
  - Call "INSERT" statements on the connection



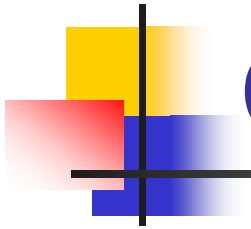
MySQL example

- $mysql -h cs4317.srvr -u stu1 -pstu1pass studentdb2
- mysql>CREATE TABLE table1(id INTEGER);
- mysql>INSERT INTO table1(id) VALUES(1);
- mysql>COMMIT;
- mysql>SELECT * FROM table1;
- mysql>DROP TABLE table1;
- mysql>exit



Special symbols in databasse

```
void character(ch[] char, int start, int length) {
    String str = new String(ch, start, length);
    str = str.trim();
    str = str.replace("&", "&amp;amp;" );
    ...
}
```



One class = one file ?

```
class A {
  class B {
  ...
  }
  class C {
  ...
  }
 ...
}
```