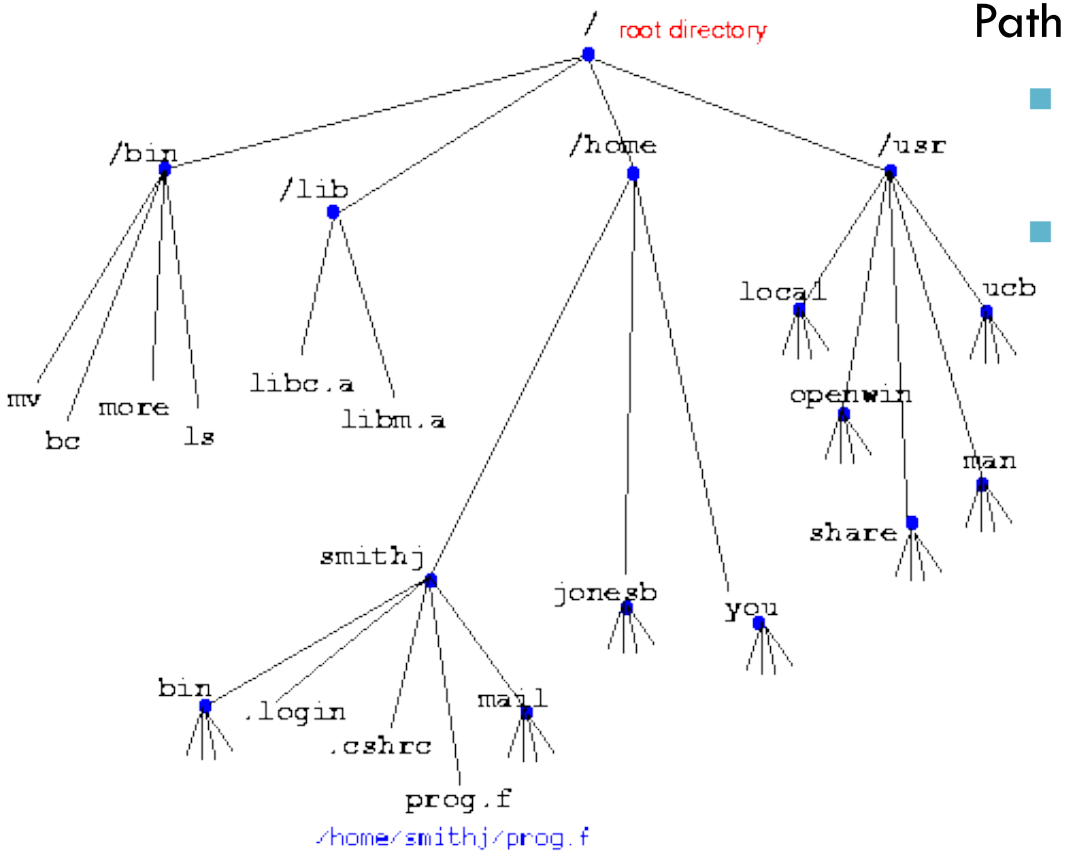
BUSINESS INTELLIGENCE LABORATORY

Data Access: Files

Two issues

- Where are my files?
  - Local file systems
  - Distributed file systems
  - Network protocols
- Which format is data in?
  - Text
    - CSV, ARFF
  - XML
  - Binary, Compressed, …

Local file system



Path of a resource

- Windows:
  - C:\Program Files\Office\sample.doc
- Linux:
  - /usr/home/r/ruggieri/sample.txt

Local file system

A logical abstraction of persistent mass memory

- hierarchical view (tree of directories and files)
- types of resources (file, directory, pipe, link, special)
- resource attributes (owner, rights, hard links)
- services (indexing, journaling)

Sample file system:

- Windows
  - NTFS, FAT32
- Linux
  - EXT2, EXT3, JFS, XFS, REISERFS, FAT32

Disk file systems [edit]

Disk file systems are usually block-oriented. Files in a

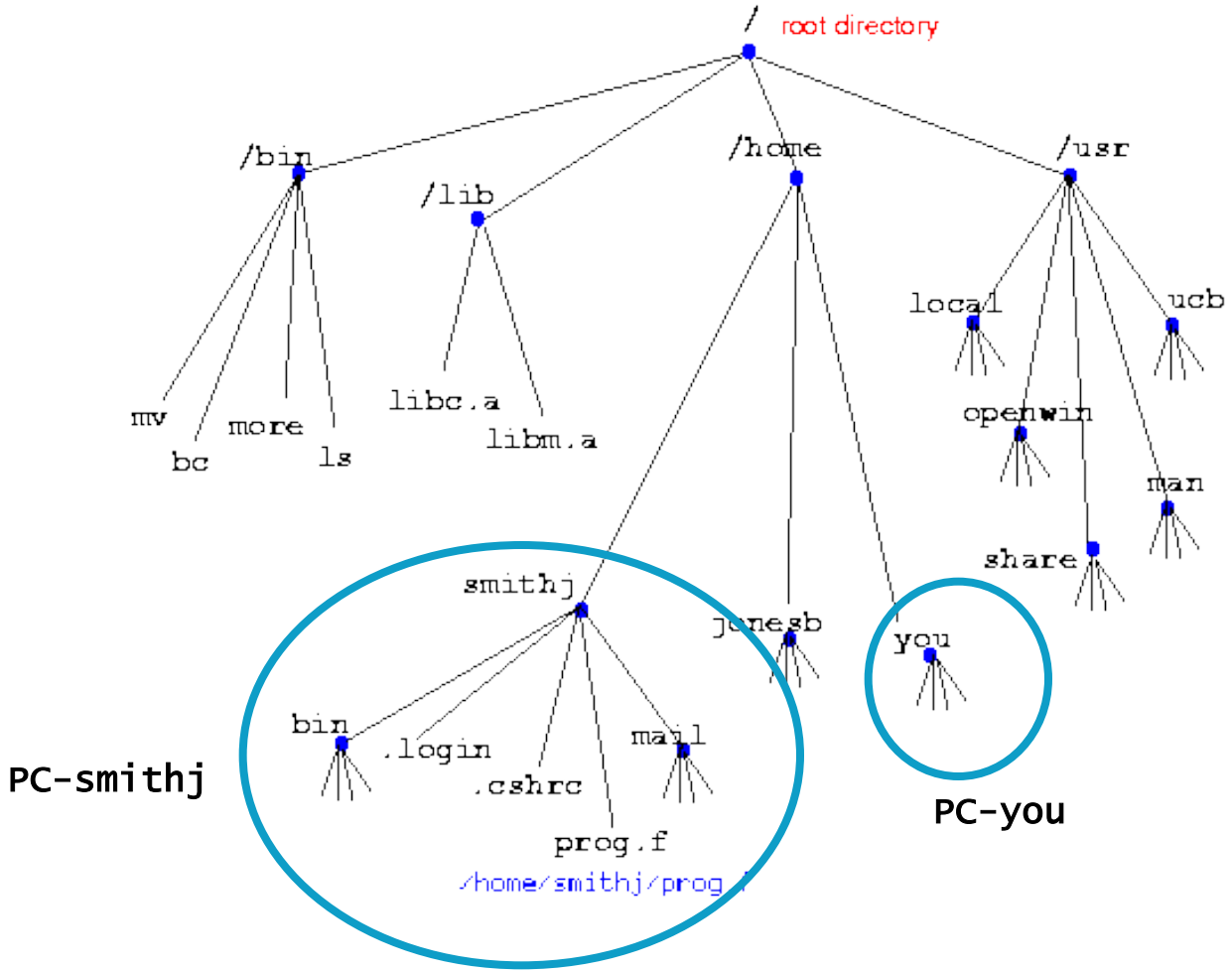
- ADFS – Acorn's Advanced Disc filing system, suc
- AdvFS - Advanced File System, designed by Digi
- AFS (Not to be confused with Andrew File Systen
- AFS - Ami File Safe, a commercial file system shi
- AosFS - File System used by the Oberon and A2
- AthFS - AtheOS File System, a 64-bit journaled fi
- BFS - the Boot File System used on System V rel
- BFS – the Be File System used on BeOS, occasio
- Btrfs - is a copy-on-write file system for Linux ann
- CBMFS – The filesystem used on most Commod
- CMDFS – A filesystem extension added to CBMF
- CP/M file system — Native filesystem used in the
- DDFS – Data Domain File System, the data dedu
- DTFS – Desktop File System, featuring file compr
- DOS 3.x - Original floppy operating system and fi
- EAFS – Extended Acer Fast Filesystem, used on
- Extent File System (EFS) – an older block filing sy
- ext – Extended file system, designed for Linux sys
- ext2 – Second extended file system, designed for
- ext3 – A journaled form of ext2.
- ext4 – A follow up for ext3 and also a journaled fil
- ext3cow – A versioning file system form of ext3.
- FAT – File Allocation Table, used on DOS and Mi
  - VFAT – Optional layer on Microsoft Windows
  - FATX – A modified version of Microsoft Windo
- FFS (Amiga) – Fast File System, used on Amiga
- FFS – Fast File System, used on *BSD systems

Local file system

Physical view

- Disk partition
  - collection of contiguous blocks on a disk
- File system driver
  - software abstracting a file system on a partition
  - Maps a file system to each partition
- Mount
  - starting a file system driver on a partition
  - Windows (start up typically is automatic:
    - at startup for NTFS and FAT partitions
    - names of partitions: A: … Z:
  - Linux
    - at startup for partitions in **/etc/fstab**
    - **> mount –t ext3 /dev/hda2 /mtn/mydisk**

Distributed file system



Distributed file system

Acts as a client for a remote file access protocol

- logical abstraction of remote persistent mass memory

Sample file system:

- Samba (SMB)

  or Common Internet File System (CIFS)
- Network File System (NFS)

Distributed file systems [edit]

See also: Comparison of distributed file syster

Distributed file systems are also called network fil

- 9P, the Plan 9 from Bell Labs and Inferno distr
- Amazon S3
- Andrew File System (AFS) is scalable and loc
- Apple Filing Protocol (AFP) from Apple Inc.. A
- DCE Distributed File System (DCE/DFS) from
- File Access Listener (FAL) is an implementatic
- Microsoft Office Groove shared workspace, us
- NetWare Core Protocol (NCP) from Novell is u
- Network File System (NFS) originally from Sur
- OS4000 Linked-OS provides distributed filesys
- Secure File System (SFS)
- Self-certifying File System (SFS), a global net
- Server Message Block (SMB) originally from I

  authentication.

Lab configuration (Windows)

- Disk H: is your home
  - beware of access rights! By default, everybody can look into it
- Disk S: is shared
  - **S:\corsi\lbi** is a shared directory with material for LBI
  - For fast access to **S:\corsi\lbi** you can:
    - create a link to desktop, or
    - map network drive **S:\corsi\lbi** as drive **Z:**

Remote address

Universal naming convention (UNC)

- Files and directories in remote server
  - **host-name****partition-name$****local-path**
- Explicitly shared resource by the remote server
  - **host-name****shared-resource**

You are using Windows

- View resources shared by other systems (including Linux)
  - > net view \\homeserver
  - from Resource explorer GUI
    - Explorer-> type \\homeserver in the address bar
- Share a resource
  - > *net share* mydirdata=C:\Data
  - … or from the properties of C:\Data
    - by selecting *Sharing*
- Mount of remote directories
  - > net use H: \\homeserver\ruggieri\LBI
  - > net use * \\homeserver\ruggieri\LBI
  - from Resource explorer GUI
    - Explorer->Tools->Map Network Drive
- Unmount
    - > net use H: /DELETE

You are using Linux

- View resources shared by other systems (including Windows)
  - > smbclient -L //homeserver -U *username*
- Mount of remote directories
  - Install cifs-utils
    - > sudo apt-get install cifs-utils
  - > mkdir *localdir*
  - > sudo mount –t cifs//homeserver/ruggieri/LBI *localdir*
    –o user=*username*,domain=FIBONACCI,file_mode=0777,dir_mode=0777
  - from Nautilius
    - Connect to server->smb://homeserver/ruggieri/LBI
- Unmount
  - > sudo umount –n *localdir*

LBI Working directory

- ~ruggieri/LBI in Linux
  - contains data and materials to be shared
- Create a symbolic link in your Linux home
  - ln –s ~ruggieri/LBI LBIdir
  - use WinSCP -> Open Terminal
- Now LBIdir is accessible both from Linux & Win
  - in Windows as Z:\LBIdir
- Another way (works only for Windows)
  - Create a shortcut LBIdir to \\homeserver\ruggieri\LBI

Network protocols

- Files accessed through explicit request/reply
- A local copy has to be made before accessing data
- Resource naming:
  - Uniform Resource Locator (URL)
    - scheme://user:password@host:port/path
    - http://bob:bye@www.host.it:80/home/idx.html
    - scheme = protocol name (http, https, ftp, file, jdbc, …)
    - port = TCP/IP port number

HTTP Protocol

- HyperText Transfer Protocol
  - URL: http://user:pwd@www.di.unipi.it
  - State-less connections
  - Crypted variant: Secure HTTP (HTTPs)
- Windows clients
  - Any browser
  - > wget
    - GNU http://www.gnu.org/software/wget/
    - W3C http://www.w3.org/Library
- Linux clients
  - Any browser
  - > wget

FTP Protocol

- File Transfer Protocol
  - URL: ftp://user:pwd@ftp.apa.unip.it/myfile
  - State-less connections
  - Commands: get / put / mget
  - Crypted variant: Secure FTP (SFTP)
- Windows clients
  - FTP: > ftp or any browser
  - SFTP:
    - PuTTY ttp://www.chiark.greenend.org.uk/~sgtatham/putty
    - SSH Secure Shell http://www.ssh.com
- Linux clients
  - FTP: > ftp > sftp > gftp (GUI)

SCP Protocol

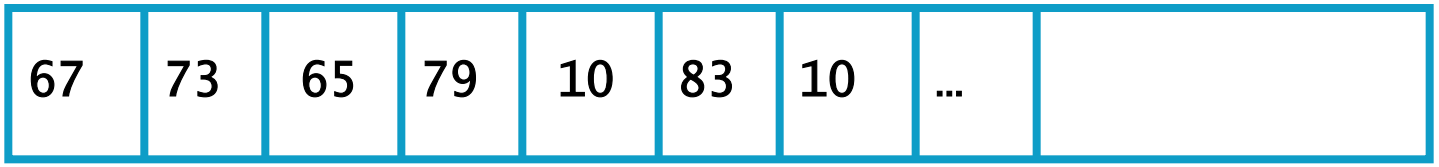
- Secure Copy
  - > scp data.zip user@alice.cli.di.unip.it:datacopy.zip
  - File copy from/to a remote account
  - File paths must be known in advance
- Client
  - command line:
    - **> scp/pscp > scp2**
  - Windows GUI
    - WinSCP http://winscp.sourceforge.net
    - SSH Secure Shell
  - Linux GUI
    - SCP: default

Two issues

- Where are my files?
  - Local file systems
  - Distributed file systems
  - Network protocols
- Which format is data in?
  - Text
    - CSV, ARFF
  - XML
  - Binary, Compressed, …

What is a file?

- File = sequence of bytes



How bytes are mapped to chars?

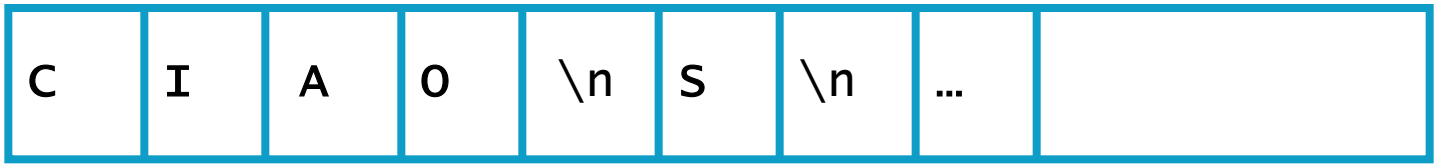
- Character set = alphabet of characters
- Coding bytes by means of a character set
  - ASCII, EBCDIC (1 byte per char)
  - UNICODE (1/2/4 bytes per char)

American Standard Code for Information Interchange

CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR
0	NUL	26	SUB	52	4	78	N	104	h
1	SOH	27	ESC	53	5	79	O	105	i
2	STX	28	FS	54	6	80	P	106	j
3	ETX	29	GS	55	7	81	Q	107	k
4	EOT	30	RS	56	8	82	R	108	l
5	ENQ	31	US	57	9	83	S	109	m
6	ACK	32	SP	58	:	84	T	110	n
7	BEL	33	!	59	;	85	U	111	o
8	BS	34	"	60	<	86	V	112	p
9	HT	35	#	61	=	87	W	113	q
10	LF	36	$	62	>	88	X	114	r
11	VT	37	%	63	?	89	Y	115	s
12	FF	38	&	64	@	90	Z	116	t
13	CR	39	'	65	A	91	[	117	u
14	SO	40	(	66	B	92	\	118	v
15	SI	41	)	67	C	93	]	119	w
16	DLE	42	*	68	D	94	^	120	x
17	DC1	43	+	69	E	95	_	121	y
18	DC2	44	,	70	F	96	`	122	z
19	DC3	45	-	71	G	97	a	123	{
20	DC4	46	.	72	H	98	b	124	|
21	NAK	47	/	73	I	99	c	125	}
22	SYN	48	0	74	J	100	d	126	~
23	ETB	49	1	75	K	101	e	127	DEL
24	CAN	50	2	76	L	102	f		
25	EM	51	3	77	M	103	g		

Text file = file+character set

- Text file = sequence di characters

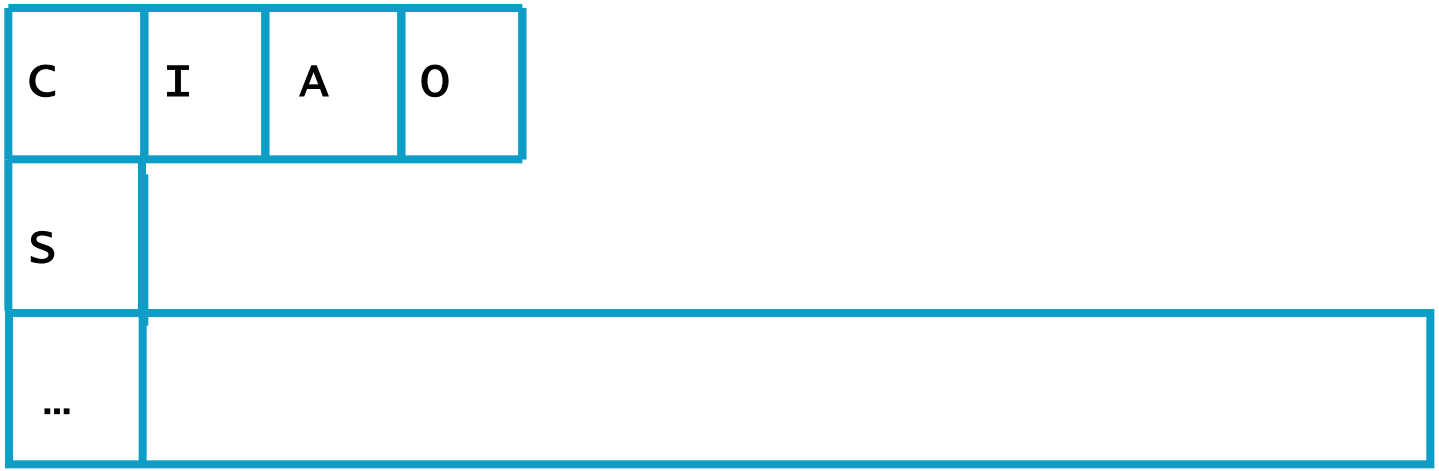


Viewing text files

- By a text editor
  - Emacs, Nodepad++,TextPad, UltraEdit, Vi, etc.
- "Carriage return" character
  - Start a new line
  - Coding
    - Unix: 1 char ASCII(0A) ('\n' in Java)
    - Windows: 2 chars ASCII(0D 0A) ("\r\n" in Java)
    - Mac: 1 char ASCII(0D) ('\r' in Java)
  - Conversions
    - > **dos2unix**
    - > **unix2dos**

Text file = file+character set

- Text file = sequence di **lines**



Tabular data format

Column

Row

Mario	Bianchi	23	Student
Luigi	Rossi	30	Workman
Anna	Verdi	50	Teacher
Rosa	Neri	20	Student

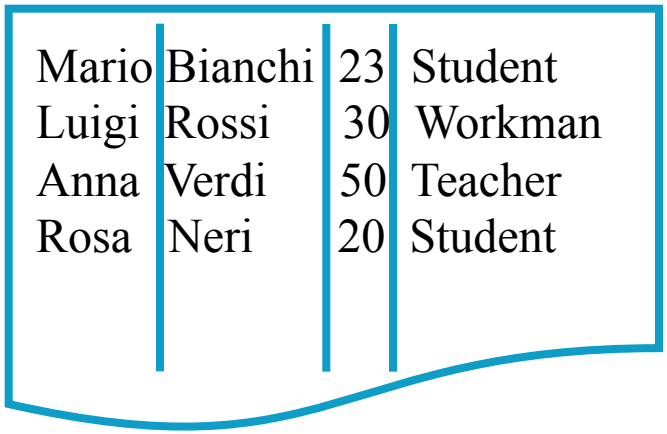
Representing tabular data in text files

- **Comma Separated Values (CSV)**
  - A row per line
  - Column values in a line separated by a special character
  - Delimiters: comma, tab, space

```
Mario,Bianchi,23,Student
Luigi,Rossi,30,Workman
Anna,Verdi,50,Teacher
Rosa,Neri,20,Student
```

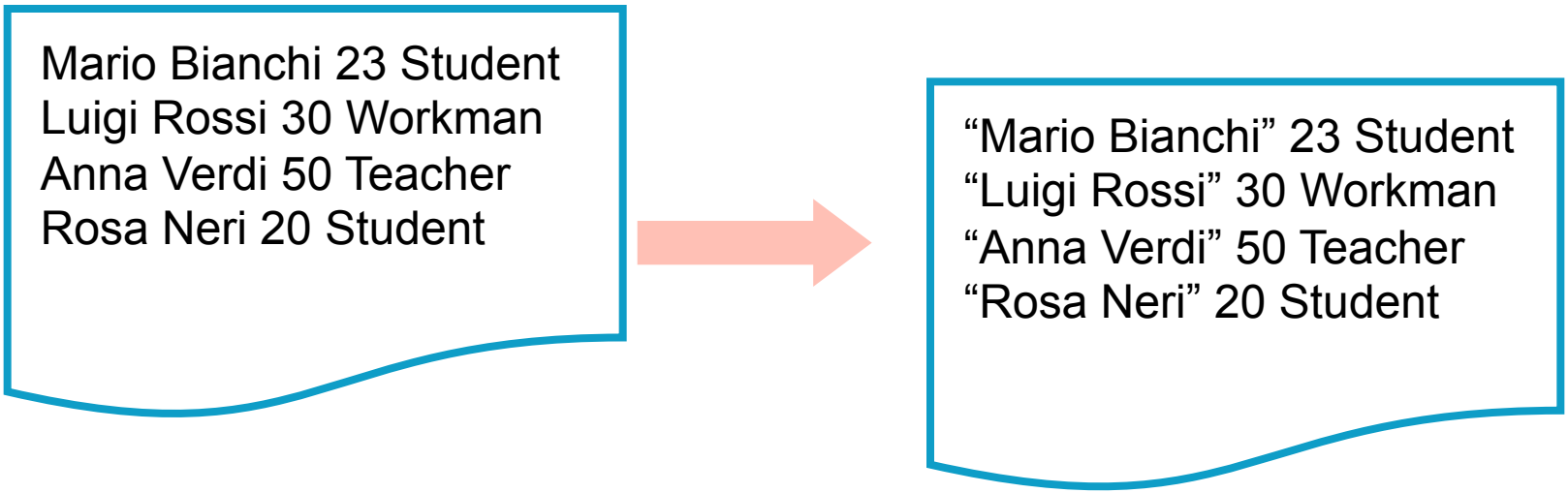
Representing tabular data in text files

- **Fixed Length Values (FLV)**
  - A row per line
  - Column values occupy a fixed number of chars
    - Allow for random access to elements
    - Higher disk space requirements



Quoting

- What happens in CSV if a delimiter is part of a value?
  - Format error
- Solution: quoting
  - Special delimiters for start and end of a value (ex. " … ")



Missing values

- How to represent missing values in CSV or FLV?
  - A reserved string: “?”, “null”, “”

“Mario Bianchi” 23 Student
“Luigi Rossi” 30 **?**
“Anna Verdi” 50 Teacher
“Rosa Neri” **?** Student

Meta-data

- Describe properties of data
  - Table name, column name, column type

name	surname	age	occupation
string	string	int	string
Mario	Bianchi	23	Student
Luigi	Rossi	30	Workman
Anna	Verdi	50	Teacher
Rosa	Neri	20	Student

Meta-data: ARFF data types

- ARFF (Attribute-Relation File Format)
  - real / integer/ numeric
    - they are synonyms and cover numeric types
  - String
    - covers strings of any length
  - { name-1, …, name-n }
    - enumerated type
    - covers an enumeration of values
    - Ex., {high, medium, low} {Play, Don't Play}
  - date "yyyy-MM-dd HH:mm:ss"
    - date and time
    - Ex., "2001-04-03 12:12:12"

How to represent meta-data in text files?

- Two rows: names and types

name	surname	age	occupation
string	string	int	string


```
name,surname,age,occupation
string,string,int,string
```

How to represent meta-data in text files?

- n rows, with two columns: name and type

name	surname	age	occupation
string	string	int	string

name	type
name	string
surname	string
age	int
occupation	string

```
name,string
surname,string
age,int
occupation,string
```

Meta-data and data in text files

- Two distinct files
  - Eg., C4.5 format with .names and .data

name	surname	age	occupation
string	string	int	string
Mario	Bianchi	23	Student
Luigi	Rossi	30	Workman
Anna	Verdi	50	Teacher
Rosa	Neri	20	Student


```
name,string
surname,string
age,int
occupation,string
```

```
Mario,Bianchi,23,Student
Luigi,Rossi,30,Workman
Anna,Verdi,50,Teacher
Rosa,Neri,20,Student
```

Meta-data and data in text files

- In the same file
  - Meta-data first, then data

name	surname	age	occupation
string	string	int	string
Mario	Bianchi	23	Student
Luigi	Rossi	30	Workman
Anna	Verdi	50	Insegnante
Rosa	Neri	20	Studente

nome,cognome,eta',professione
string,string,int,string
Mario,Bianchi,23,Studente
Luigi,Rossi,30,Operaio
Anna,Verdi,50,Insegnante
Rosa,Neri,20,Studente

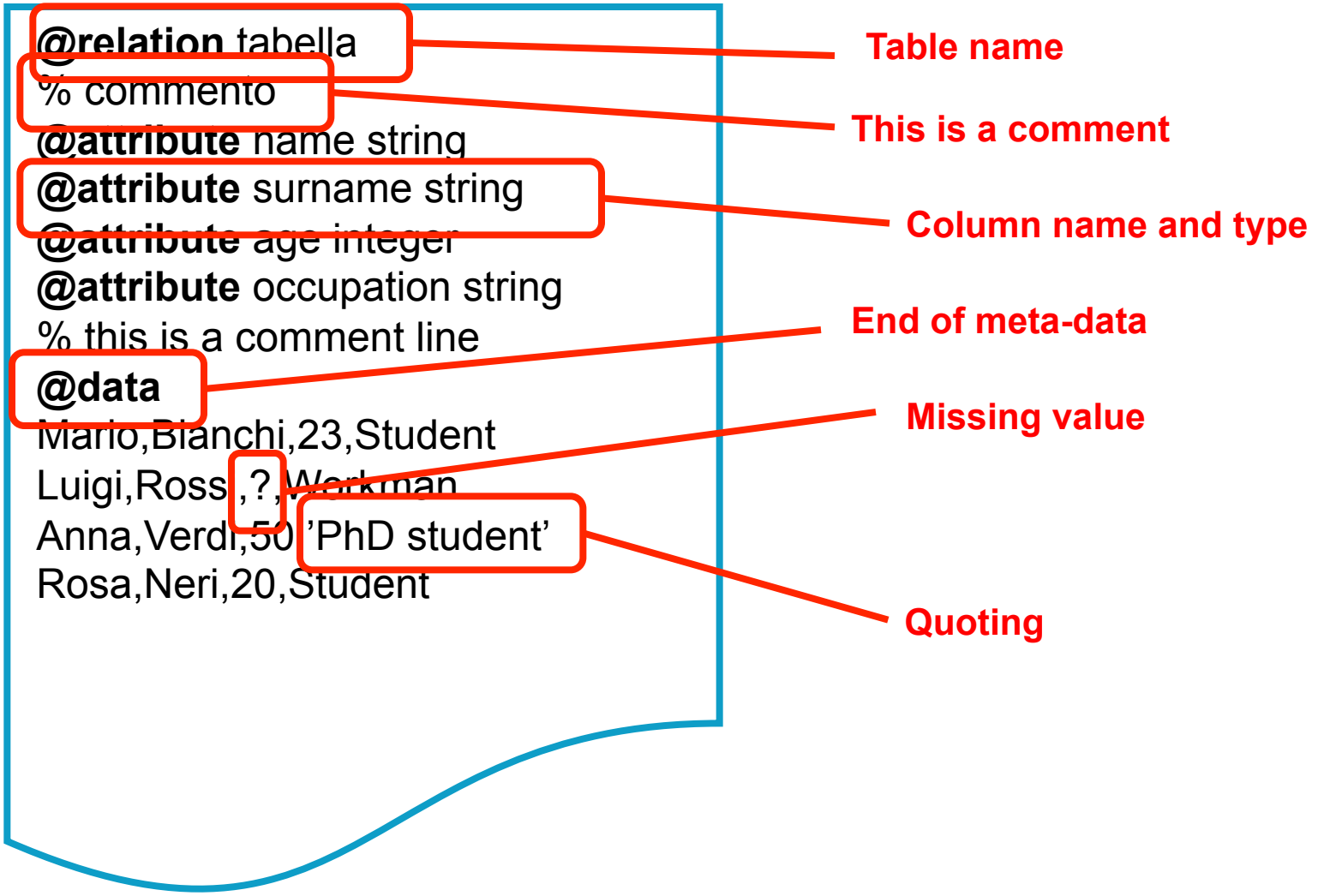
Meta-data and data in text files

- In the same file
  - Meta-data first, then data
  - A delimiter line may be required

nome	cognome	eta'	professione
string	string	int	string
Mario	Bianchi	23	Studente
Luigi	Rossi	30	Operaio
Anna	Verdi	50	Teacher
Rosa	Neri	20	Student


```
name,string
surname,string
age,int
occupation,string
@data
Mario,Bianchi,23,Student
Luigi,Rossi,30,Workman
Anna,Verdi,50,Teacher
Rosa,Neri,20,Student
```

Weka ARFF format

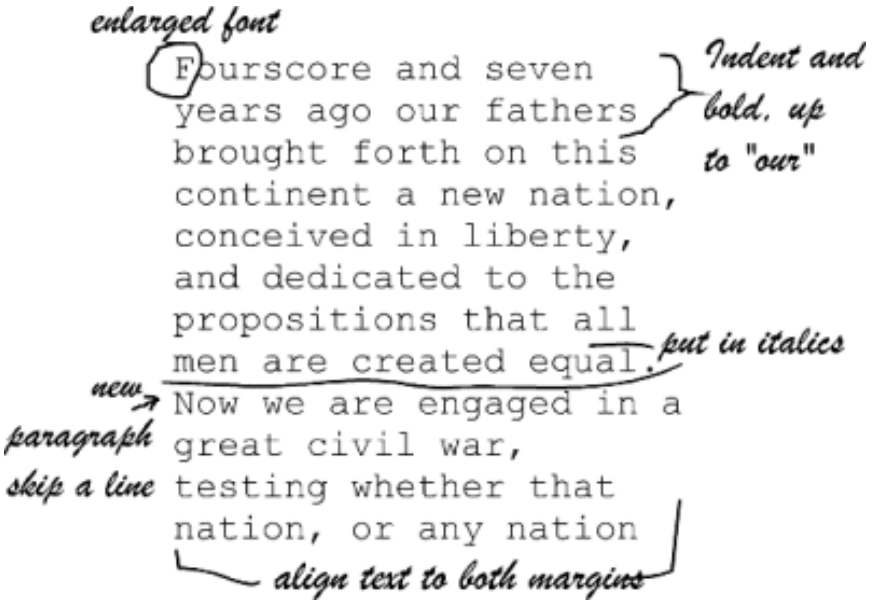


Two issues

- Where are my files?
  - Local file systems
  - Distributed file systems
  - Network protocols
- Which format is data in?
  - Text
    - CSV, ARFF
  - XML
  - Binary, Compressed, …

Data representation in XML

- XML = eXtensible Markup Language
- **XML** allows for the definition of markup languages that represent structured data
  - Markup: marking, tagging, highlighting the meaning of a data element



Why using markup languages?

- Problem: data interchange between applications
  - Proprietary data format do not allow for easy interchange
    - CSV with different delimiters, or column orders
    - Similar limitations of FLV, ARFF, binary data, etc.
- Solution:
  - definition of an interchange format…
  - … marking data elements with their meaning …
  - … so that any other party can easily interpret them.

XML by example


```
<?xml version="1.0" encoding="UTF-8"?>
<Music>
    <CD number="1" >
     <song track="1">
          <artist>Iron Maiden</artist>
          <album>Killers</album>
          <year>1980</year>
          <title>The Ides of March</title>
          <length>1:55</length>
     </song>
     <!– this is a comment -->
     <song track="4">
          <artist>Iron Maiden</artist>
          <album>Powerslave</album>
          <title>Another Life</title>
          <length>3:12</length>
     </song>
    </CD>
    …
</Music>
```

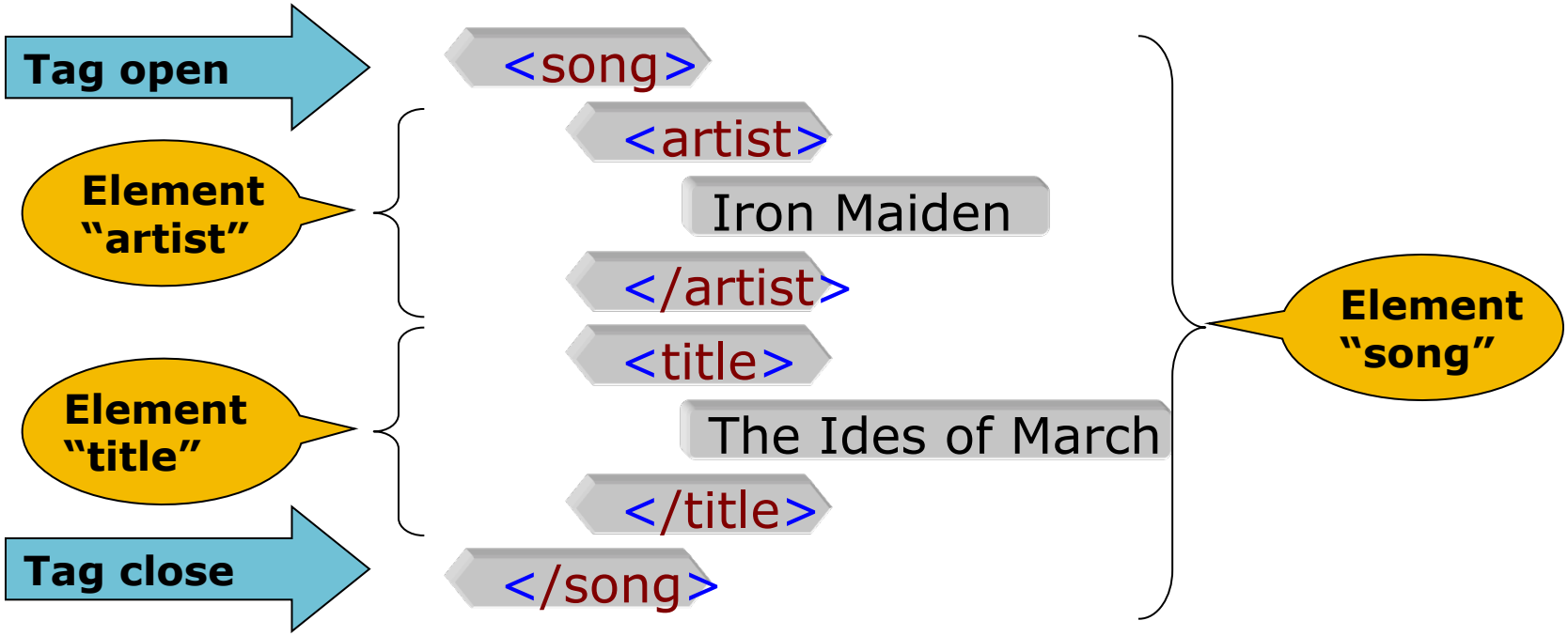
Prologue: XML declaration


```
<?xml version="1.0" encoding="UTF-8"?>
```

- Mandatory at the beginning of the document
- Attributes:
  - *version*: (mandatory) XML version of the document.
  - *encoding*: (optional) character encoding (default: UTF-8)
  - *standalone*: (optional) if set to yes then the document does not refer to external documents (default: no)

Elements

- An element is a piece of data, delimited by and identified by a tag name.



Elements

- Tag open syntax :

  <name attributes>

  - ***name*** is the name of the element.
  - ***attributes*** is an *optional* list of attribute-values

- Tag close syntax:

  </name>

  - ***name*** is the name of the element

- Elements with no content:

  <name attributes />

- There exists one and only one root element

Attributes

- They allow for specifying properties of elements using the syntax attribute = "value"

<name attribute="value">

  - <CD number="1" >
  - Attributes appear in the tag open
    - Order is not relevant
    - The "attribute or inner element?" dilemma

Text

- Reserved chars: '>', '<' and '&'
  - Meta-characters for reserved chars
    - **> < & amp;**
  - Character entities: 'à'
    - **à**
- CDATA sections
  - Bunch of textual data
    - <!CDATA[here any text with no XML meaning]]>

XML, what else …

- … we will not see in detail:
  - Document Type Definition and XML Schema
    - grammars of a class of XML documents
  - Namespaces
    - reuse of tag names in different context
  - Tag reference and hyperlinks
  - Query languages and API
  - XPath, XQuery, DOM, SAX
  - Usage in WWW:
    - Document transformation and XSLT
    - Style sheets and CSS

Tabular data, again

name	surname	age	occupation
string	string	int	string
Mario	Bianchi	23	Student
Luigi	Rossi	30	Workman
Anna	Verdi	?	Teacher
Rosa	Neri	20	Student

How to represent tabular data in XML?

- Format "Row"
  - an element <row> for every row, with an attribute for every non-missing column value

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <row name="Mario" surname="Bianchi" age="23" ocpt="Student" />
  <row name="Luigi" surname="Rossi" age="30" ocpt="Workman" />
  <row name="Anna" surname="Verdi" ocpt="Teacher" />
  <row name="Mario" surname="Bianchi" age="23" ocpt="Student" />
</root>
```

How to represent tabular data in XML?

- Format "Elements"
  - an element <row> with an inner element for every non-missing column value

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
    <row>
        <name>Mario</name>
        <surname>Bianchi</surname>
        <age>23</age>
        <ocpt>Studente</ocpt>
    </row>
    <row>
        <name>Luigi</name>
        <surname> Rossi </surname>
        <age>30</age>
        <ocpt> Operaio </ocpt>
    </row>
</root>
```

How to represent meta-data in XML?

- An element <schema> with an inner element <attribute> for every column

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
   <schema>
        <attribute name="name" type="string"/>
        <attribute name="surname" type="string"/>
        <attribute name="age" type="int"/>
        <attribute name="ocpt" type="string"/>
    </schema>
  <row name="Mario" surname="Bianchi" age="23" ocpt="Student" />
  <row name="Luigi" surname="Rossi" age="30" ocpt="Workman" />
  <row name="Anna" surname="Verdi" ocpt="Teacher" />
  <row name="Mario" surname="Bianchi" age="23" ocpt="Student" />
</root>
```

ARFF+XML = XRFF

- eXtensible attribute-Relation File Format
- XML version of ARFF
  - with additional column data types

```
- <dataset name="iris" version="3.5.3">
  - <header>
    - <attributes>
        <attribute name="sepallength" type="numeric" class="no" />
        <attribute name="sepalwidth" type="numeric" class="no" />
        <attribute name="petallength" type="numeric" class="no" />
        <attribute name="petalwidth" type="numeric" class="no" />
      - <attribute class="yes" name="class" type="nominal">
        - <labels>
            <label>Iris-setosa</label>
            <label>Iris-versicolor</label>
            <label>Iris-virginica</label>
          </labels>
        </attribute>
      </attributes>
    </header>
  - <body>
    - <instances>
      - <instance type="normal">
          <value missing="no">5.1</value>
          <value missing="no">3.5</value>
          <value missing="no">1.4</value>
          <value missing="no">0.2</value>
          <value missing="no">Iris-setosa</value>
```

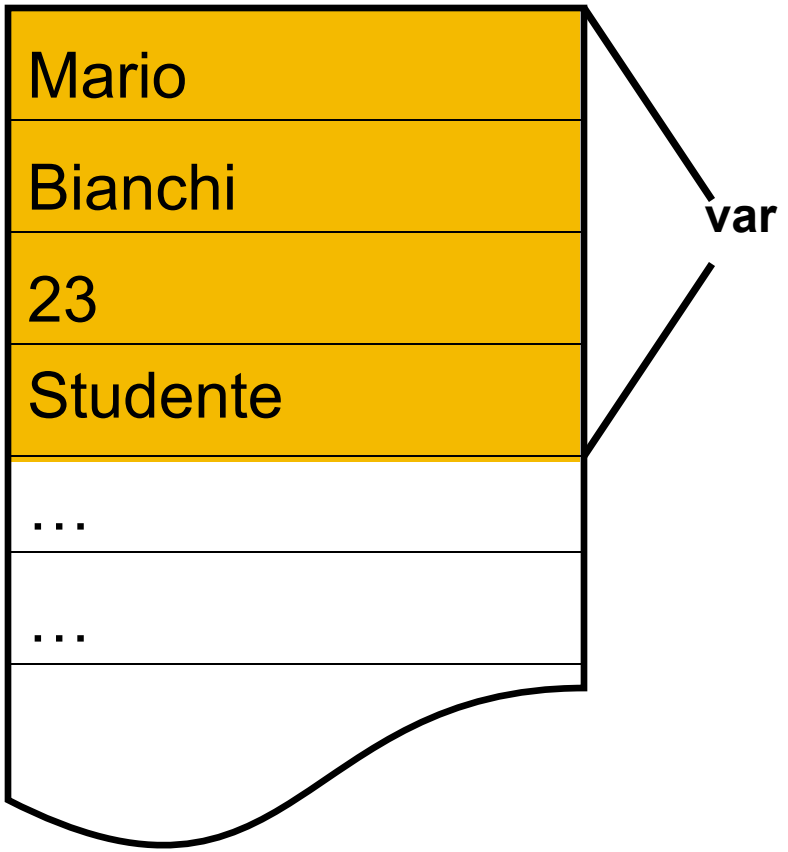
Two issues

- Where are my files?
  - Local file systems
  - Distributed file systems
  - Network protocols
- Which format is file data in?
  - Text
    - CSV, ARFF
  - XML
  - Binary

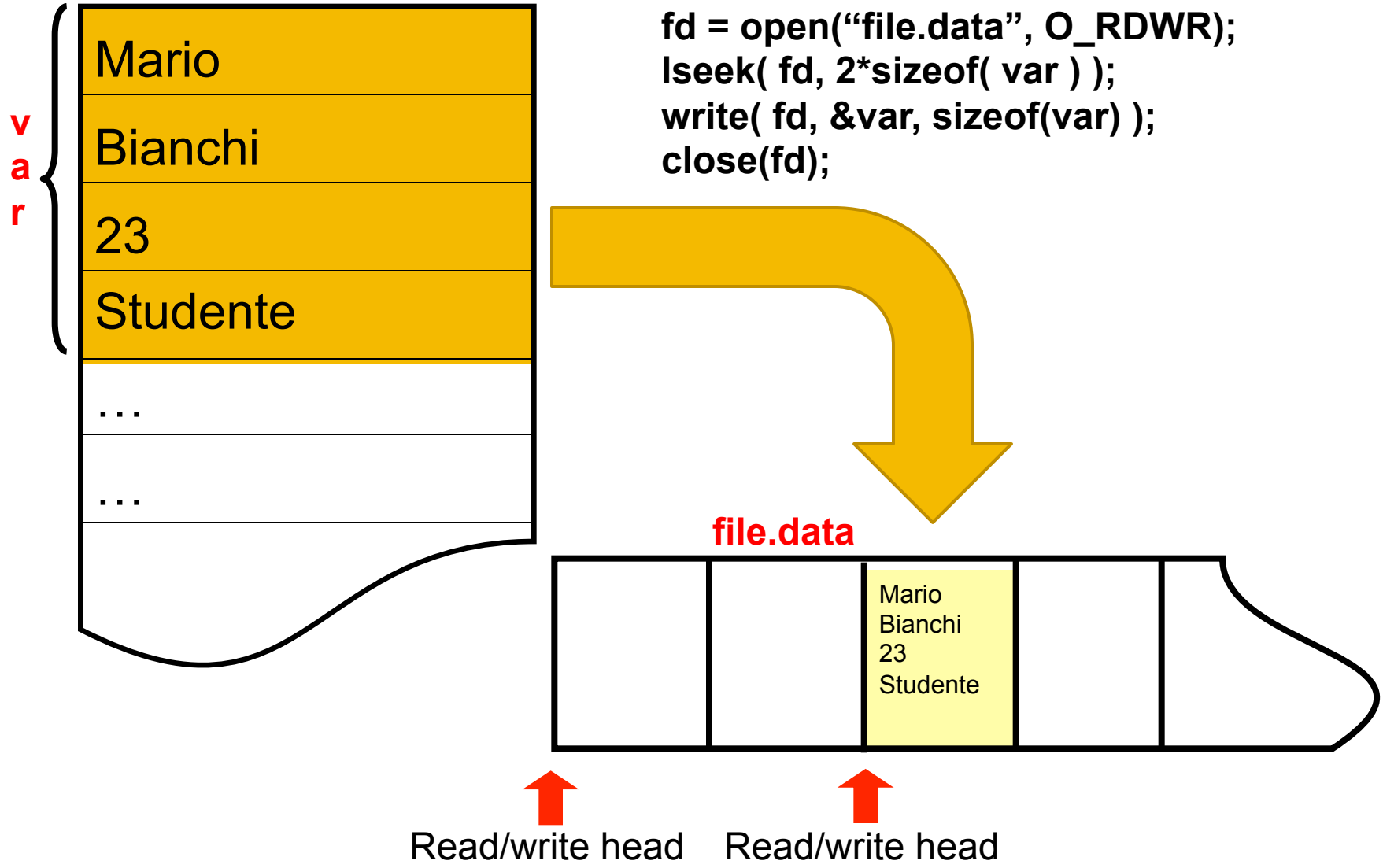
Binary files: from RAM …


```
// C struct
struct row{
    char name[20];
    char surname[20];
    int age;
    char prof[30];
} var;

// RAM occupied
int space = sizeof( var );
```



… to files, and back



Binary files: coding

- Binary coding of a char
  - character set ASCII/UNICODE
  - E.g., 'a' is coded in ASCII with one byte 01000001

- Binary coding of integers, e.g., 1027
  - Assume sizeof(int) = 4 bytes
  - Big endian (1234)
    - 00000000 00000000 00000100 00000011
  - Little endian (4321)
    - 00000011 00000100 00000000 00000000

Binary files: coding

- Binary coding of floating point numbers
  - Standard IEEE
- Binary coding of data structures
  - *struct:* sequence of the struct members
  - *array*: sequence of array elements
  - *trees, queues, indexes, tables, data bases*: … serialization of the data structure members.

Question: which format to choose ?

- Consider a table with two columns customerID (of type int) and amount (of type double), with sizeof(int) = 4 , sizeof(double) = 8
- Assume to represent table data in CSV, FLV, XML and binary formats. Which one produces the largest file? Which one produces the smallest one?
- What is the answer for a table with only one column customerName (of type string)?