DATA MANAGEMENT FOR BUSINESS INTELLIGENCE

Data Access: Files

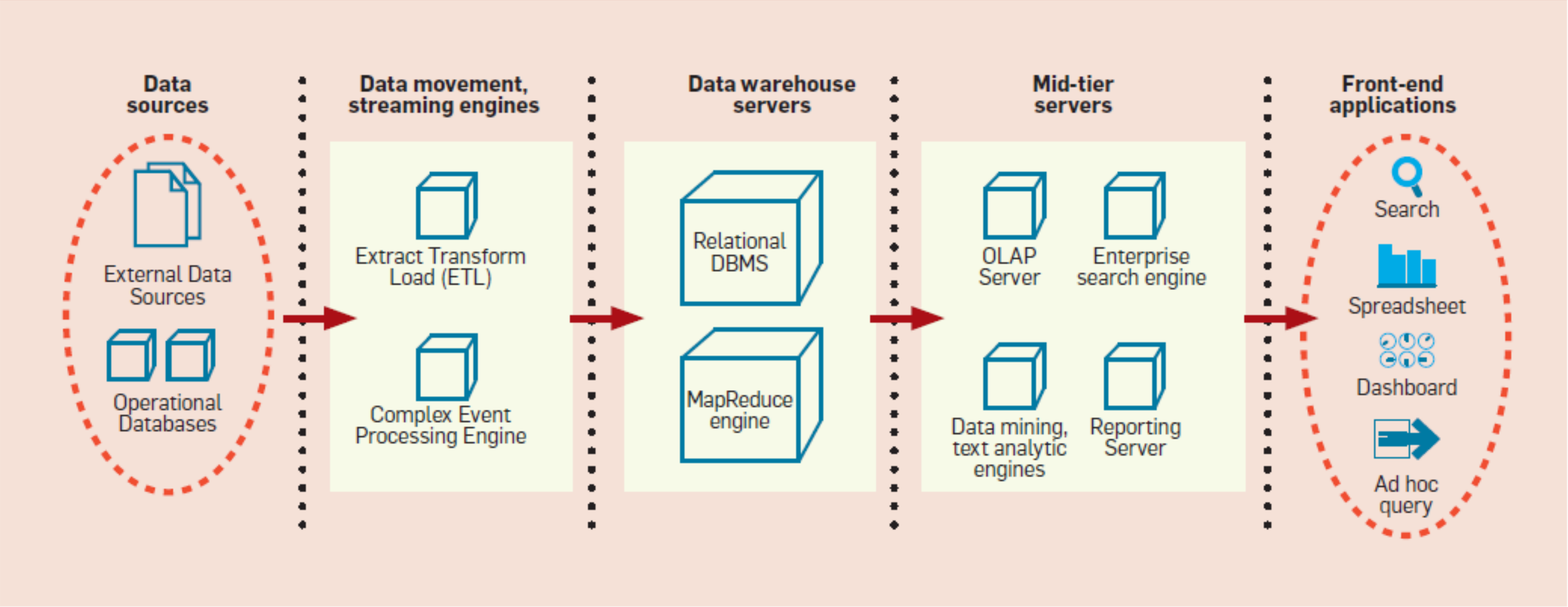
Salvatore Ruggieri

Computer Science Department, University of Pisa

Master in Big Data Analytics and Social Mining

BI Architecture

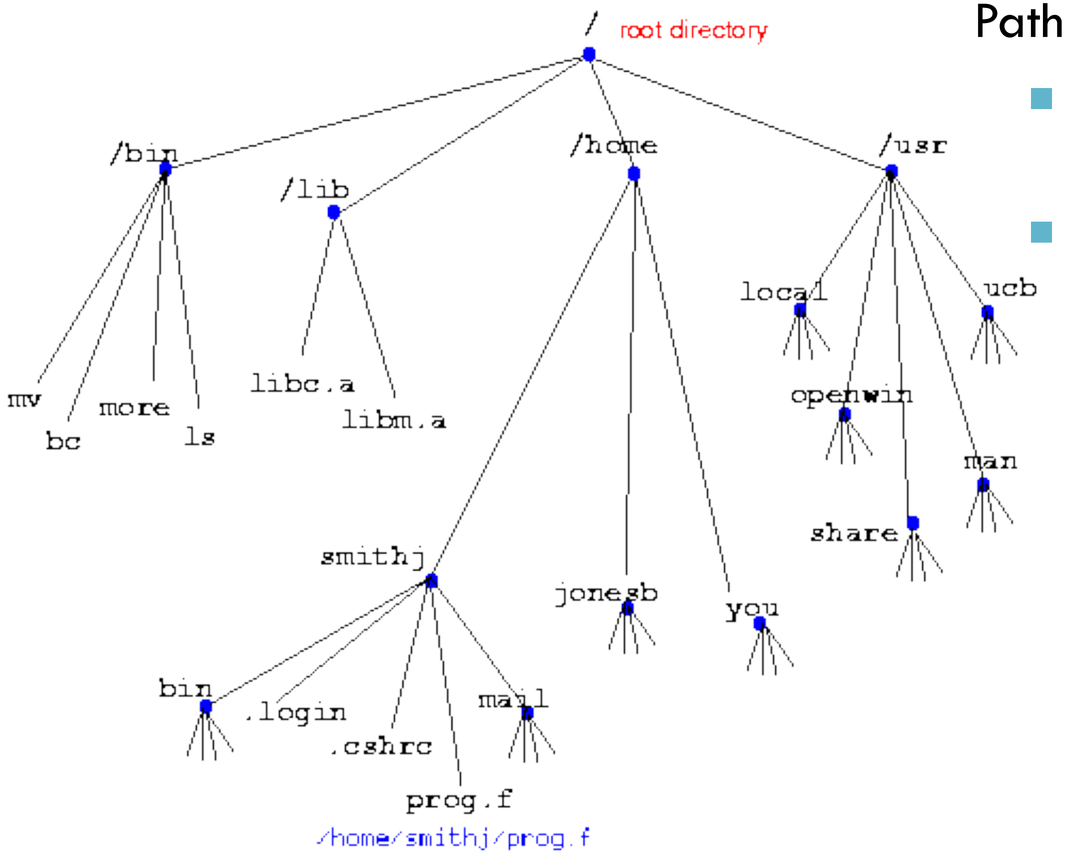
Figure 1. Typical business intelligence architecture.



Two issues

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

Local file system



Path of a resource

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

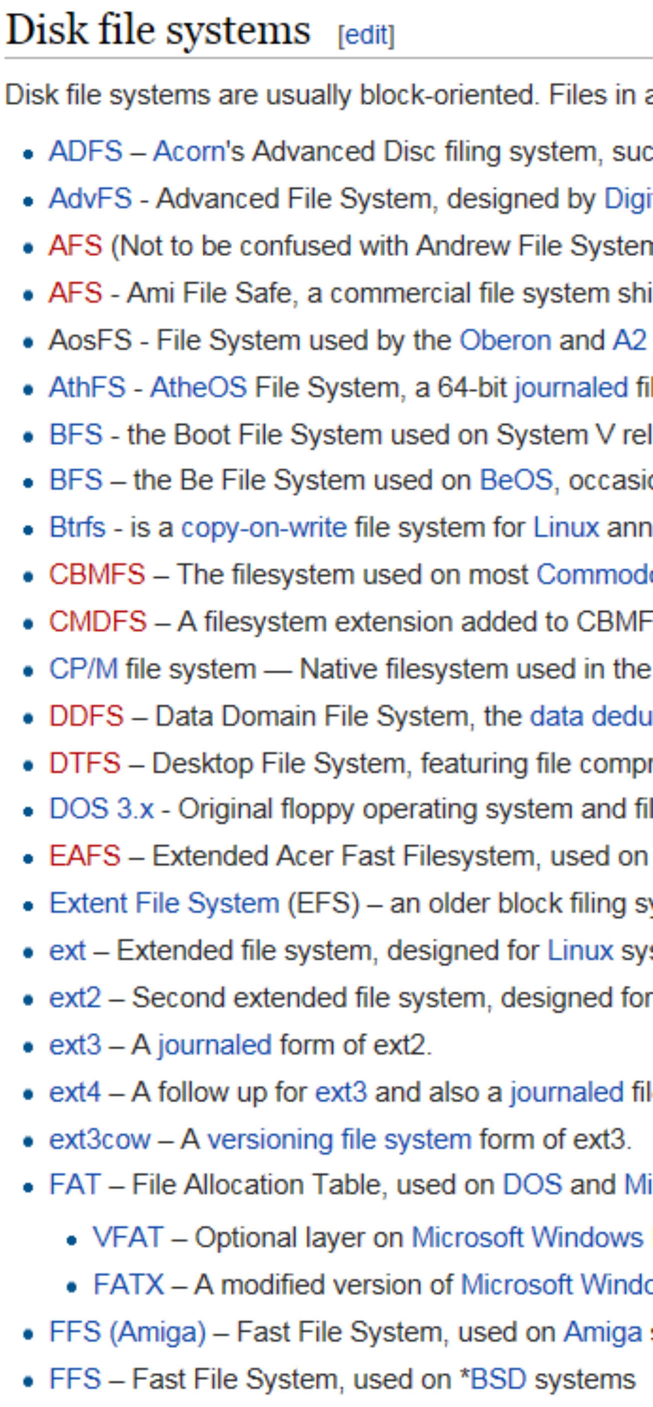
Local file system

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

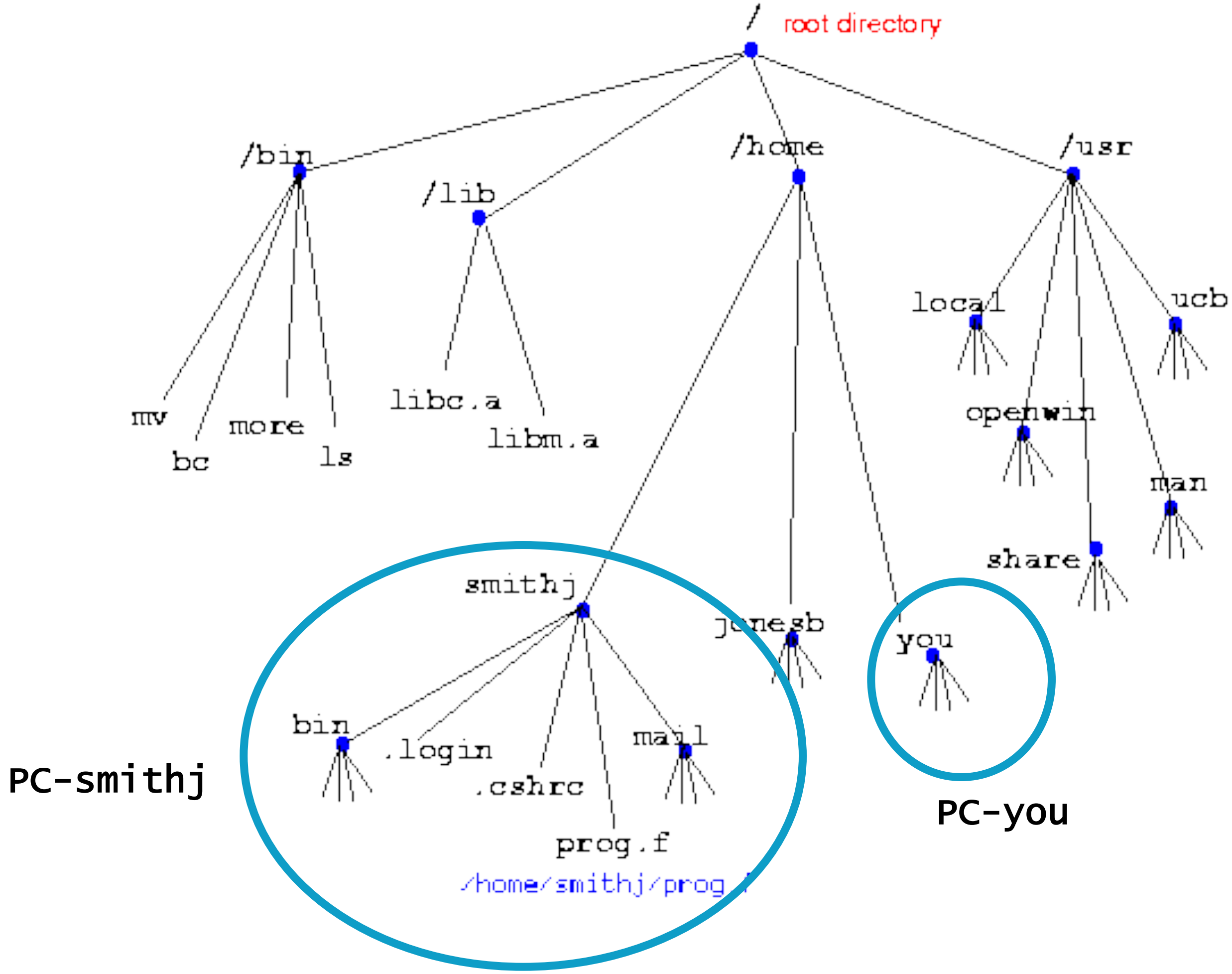


Disk file systems [edit]

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 fil
- 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 Commodo
- 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 fil
- 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 s
- FFS – Fast File System, used on *BSD systems

Distributed file system



Distributed file system

Acts as a client for a remote file access protocol

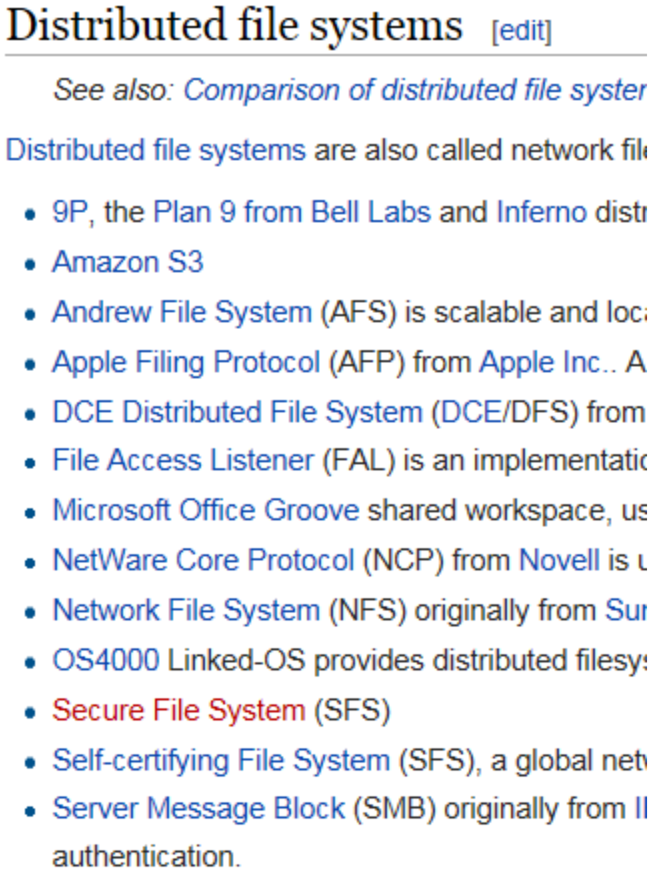
- logical abstraction of remote persistent mass memory

Sample file system:

- Samba (SMB)

  or Common Internet File System (CIFS)
- Network File System (NFS)
- Hadoop Distributed File System (HDFS)

Mount/unmount



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 filesy
- Secure File System (SFS)
- Self-certifying File System (SFS), a global net
- Server Message Block (SMB) originally from II
  authentication.

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

SCP Protocol

- Secure Copy
  - > scp data.zip user@mylinux.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 file data in?
  - Text
    - CSV, ARFF, JSON

What is a file?

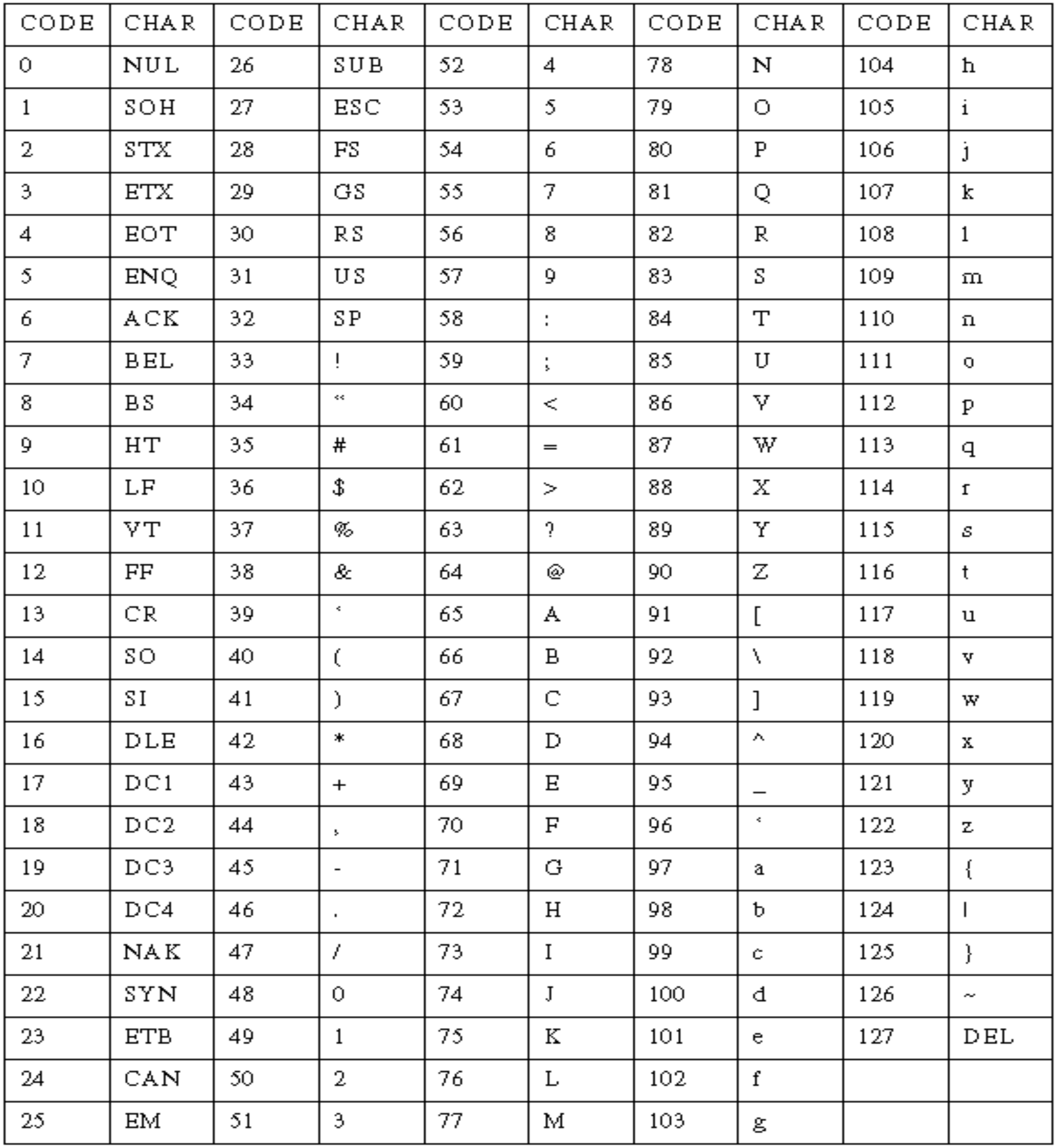
- File = sequence of bytes

67	73	65	79	10	83	10	…	

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

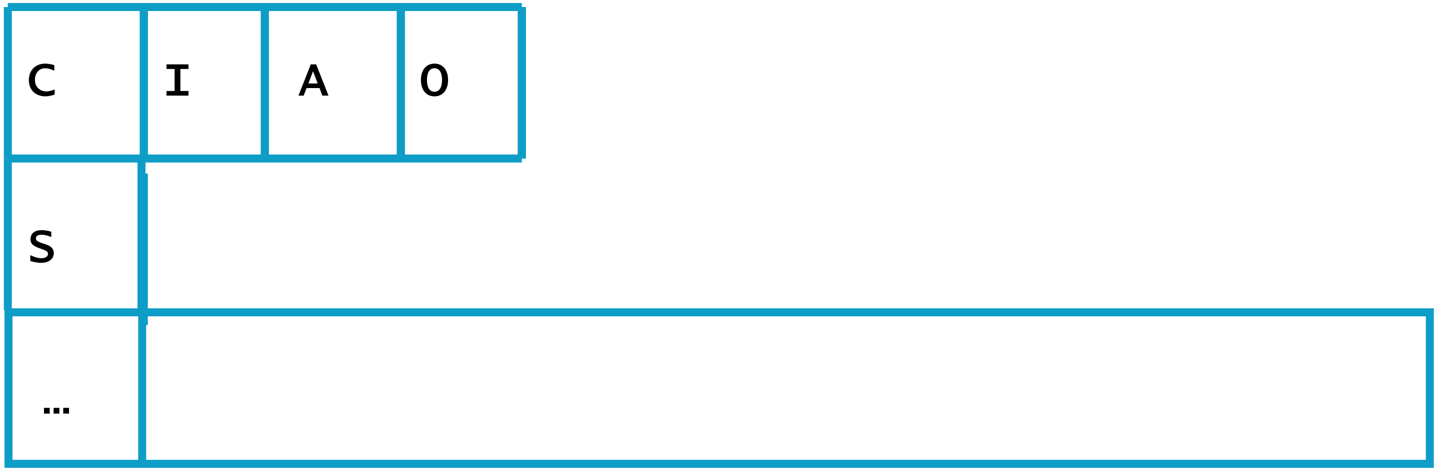
C	I	A	O	\n	S	\n	…	

Viewing text files

- By a text editor
  - Emacs, Nodepad++,TextPad, GEdit, 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**



C I A O

S

…

Tabular data format

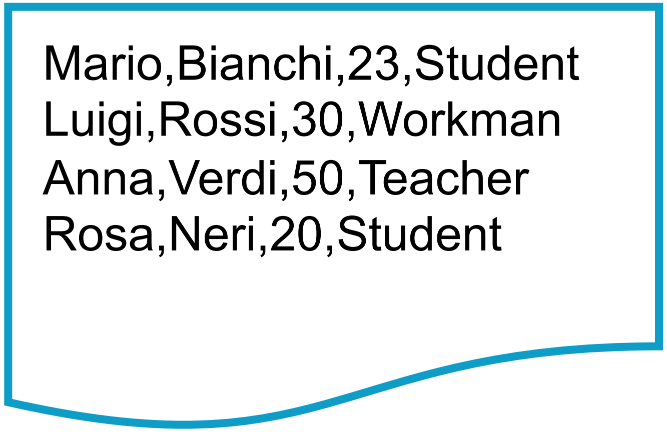
Column

Row

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

Representing tabular data in text files

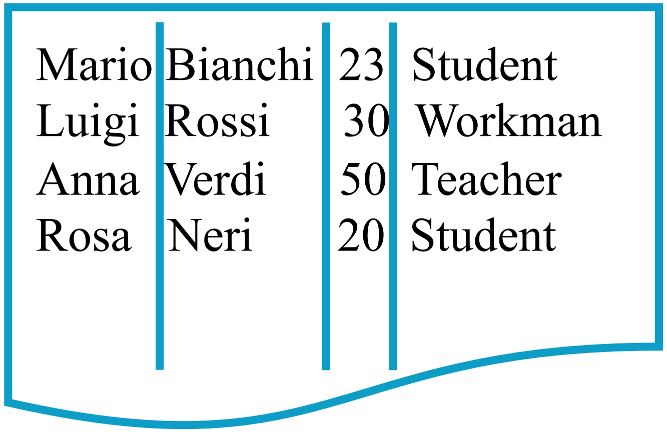
- **Comma Separated Values (CSV)**
  - A row per line
  - Column values in a line separated by a special character
  - Delimiters: comma, tab, space



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

Representing tabular data in text files

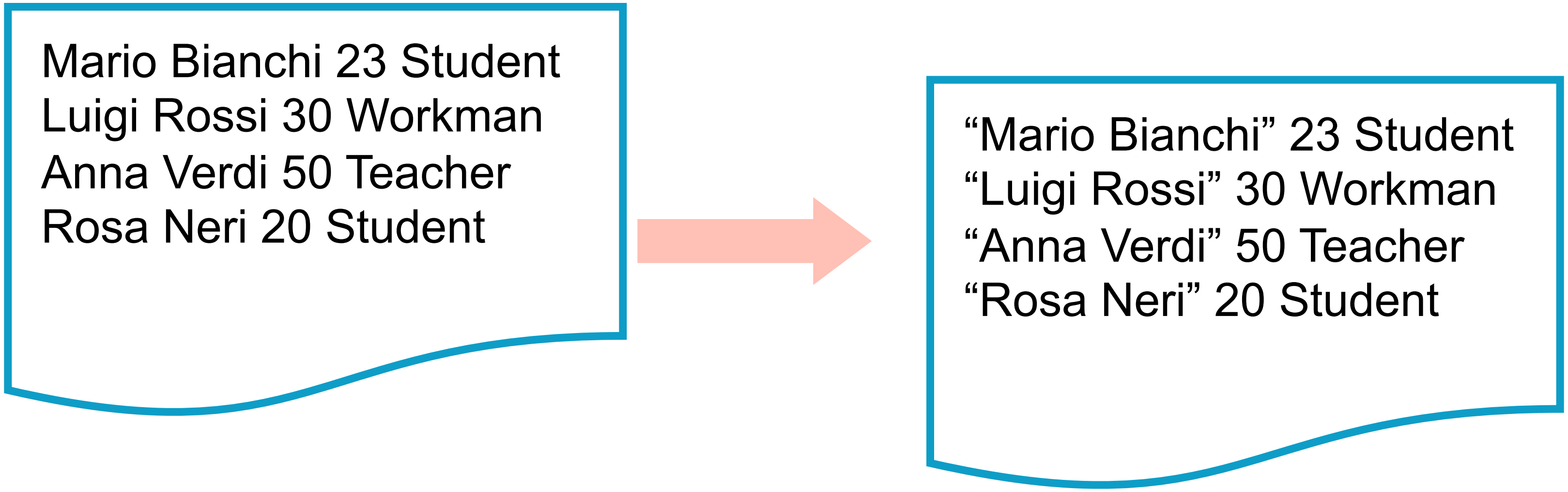
- **Fixed Length Values (FLV)**
  - A row per line
  - Column values occupy a fixed number of chars
    - Allow for random access to elements
    - Higher disk space requirements



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

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. “ … “)

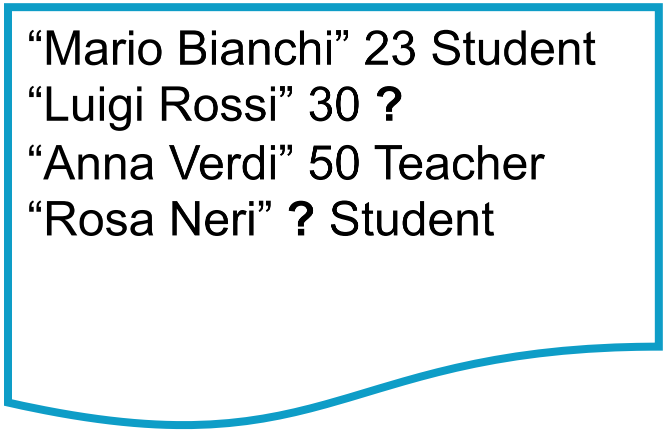


Mario Bianchi 23 Student
Luigi Rossi 30 Workman
Anna Verdi 50 Teacher
Rosa Neri 20 Student

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

Missing values

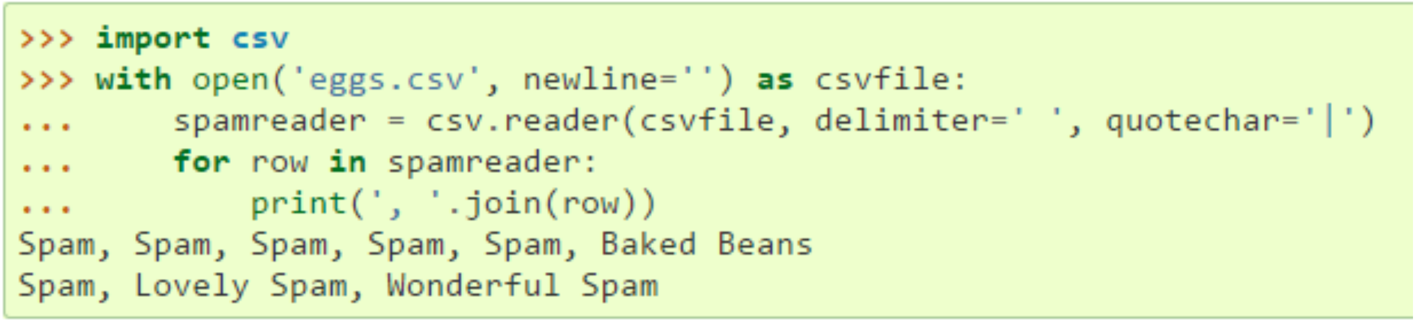
- How to represent missing values in CSV or FLV?
  - A reserved string: “?”, “null”, “”



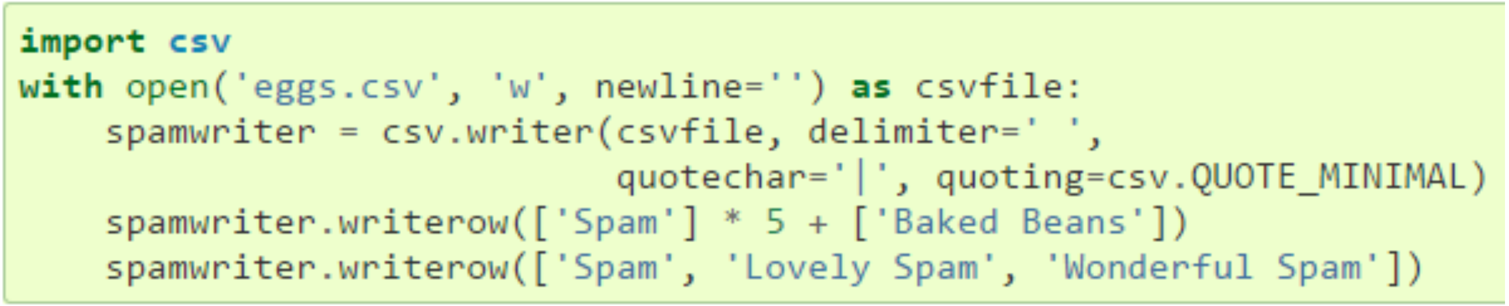
```
“Mario Bianchi” 23 Student
“Luigi Rossi” 30 ?
“Anna Verdi” 50 Teacher
“Rosa Neri” ? Student
```

CSV in Python

- □ https://docs.python.org/3/library/csv.html



```
>>> import csv
>>> with open('eggs.csv', newline='') as csvfile:
...     spamreader = csv.reader(csvfile, delimiter=' ', quotechar='|')
...     for row in spamreader:
...         print(', '.join(row))
Spam, Spam, Spam, Spam, Spam, Baked Beans
Spam, Lovely Spam, Wonderful Spam
```



```
import csv
with open('eggs.csv', 'w', newline='') as csvfile:
    spamwriter = csv.writer(csvfile, delimiter=' ',
                            quotechar='|', quoting=csv.QUOTE_MINIMAL)
    spamwriter.writerow(['Spam'] * 5 + ['Baked Beans'])
    spamwriter.writerow(['Spam', 'Lovely Spam', 'Wonderful Spam'])
```

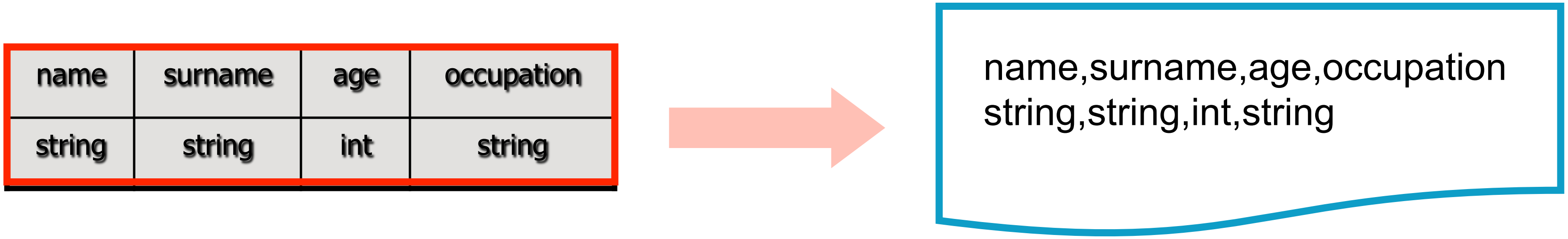
Meta-data

- 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

How to represent meta-data in text files?

- One or two rows: names and types

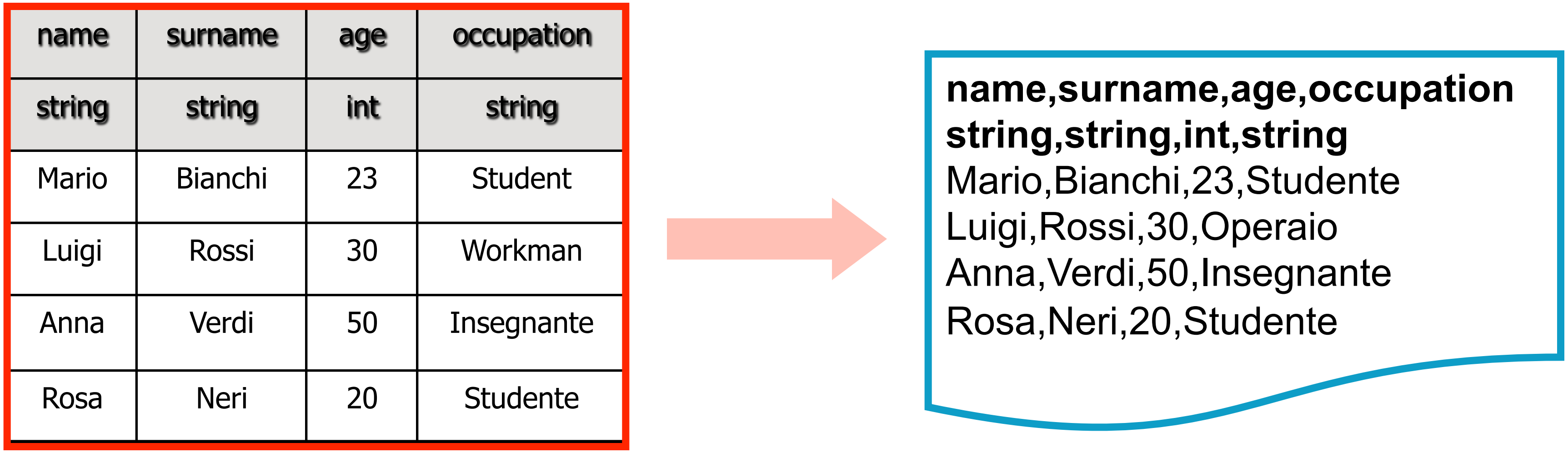


name	surname	age	occupation
string	string	int	string

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

Meta-data and data in text files

- In the same file
  - Meta-data first (header), 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

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

Two issues

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

Data interchange issue

- 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.

JSON http://www.json.org/

- Objects:
  - comma-separated list of pairs in the form
    - name : value

```
{
    "name":"John",
    "surname":"Doe",
    "age":25
}
```

- Name is a string
- Value data types:
  - strings ("John")
  - integer, real (25)

JSON

- Value data types:
  - Arrays: comma-separated list of values

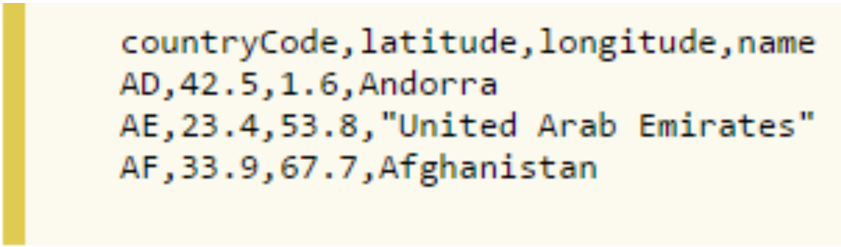
```
{
    "name":"John",
    "surname":"Doe",
    "age":25,
    "courses": ["BD", "DM", "AI"]
}
```

JSON

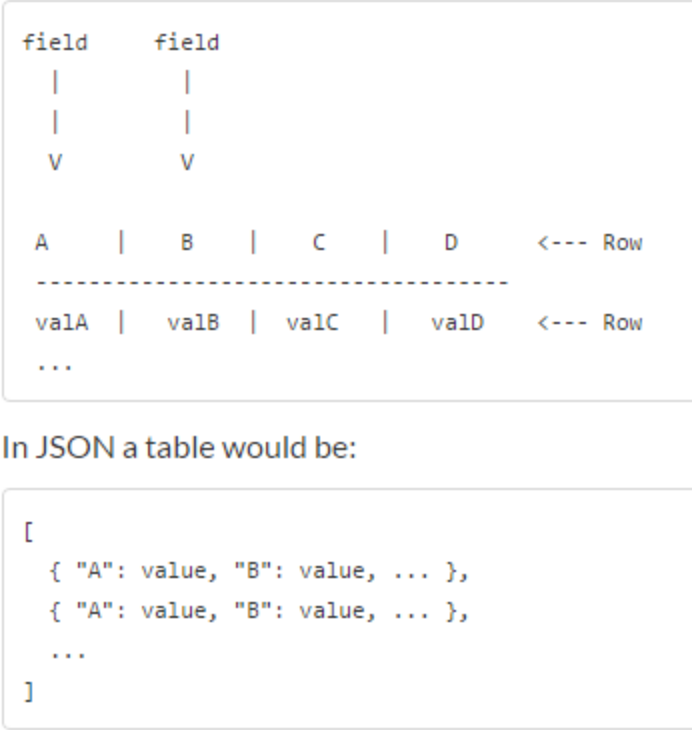
- Value data types:
  - Objects

```
{
    "name":"John",
    "surname":"Doe",
    "age":25,
    "courses": ["BD", "DM", "AI"],
    "address": {"street":"5th Av.", "city":"NY"},
    "friends": [ {"name":"Ed", "surname":"May"},
                {"name":"Al", "surname":"Black"} ]
}
```

How to map CSV in JSON?



```
countryCode,latitude,longitude,name
AD,42.5,1.6,Andorra
AE,23.4,53.8,"United Arab Emirates"
AF,33.9,67.7,Afghanistan
```

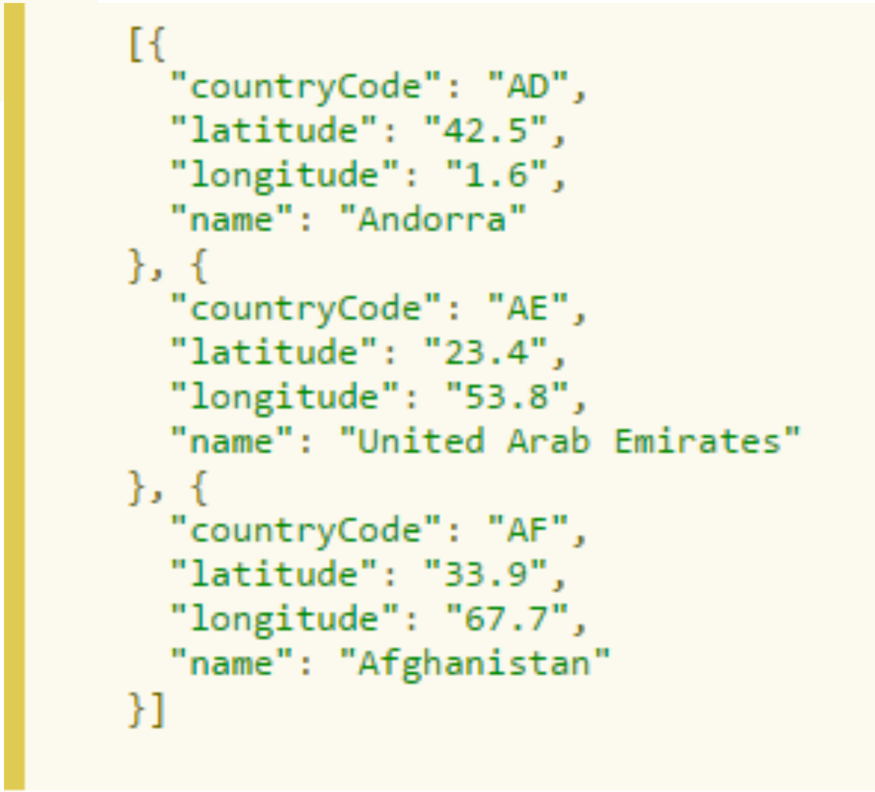


```
field     field
  |         |
  |         |
  V         V

 A     |    B    |    C    |    D      <--- Row
 ------------------------------------
 valA  |  valB   |  valC   |   valD   <--- Row
 ...
```

In JSON a table would be:

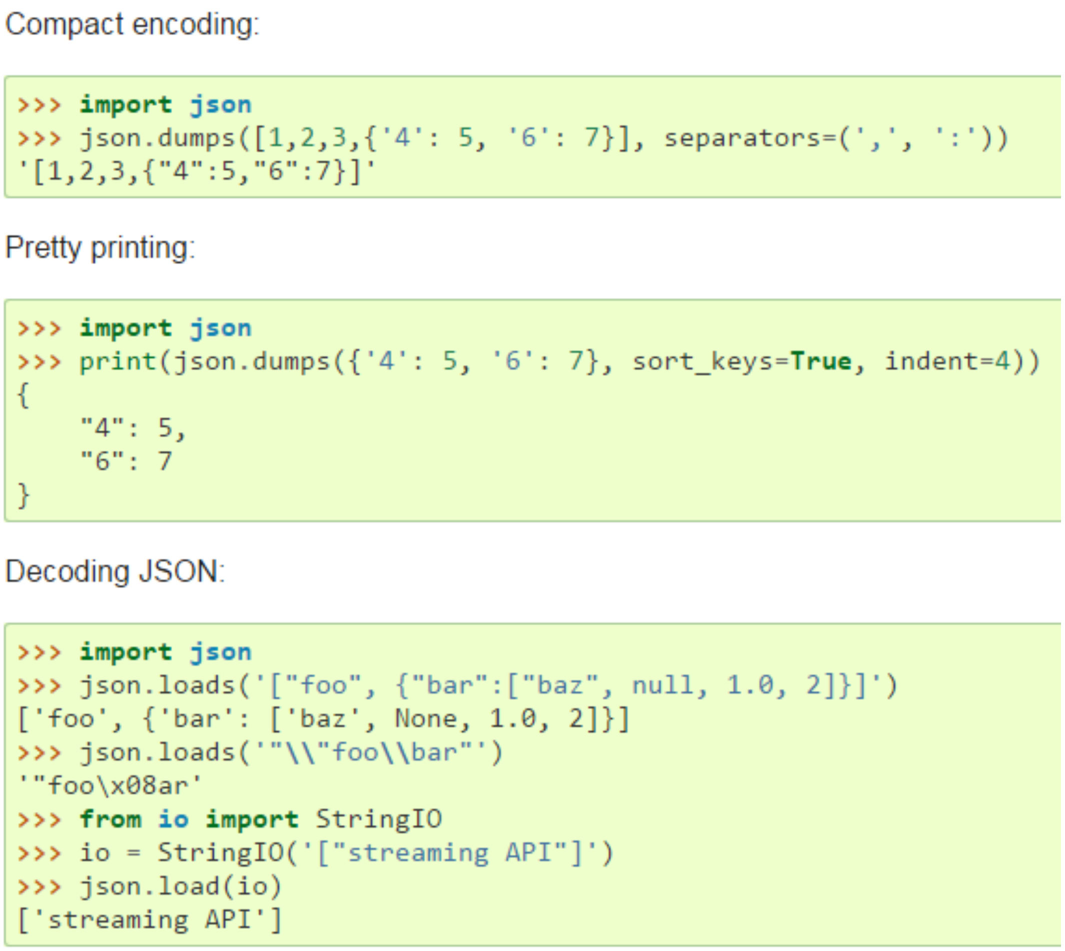
```
[
  { "A": value, "B": value, ... },
  { "A": value, "B": value, ... },
  ...
]
```



```
[{
   "countryCode": "AD",
   "latitude": "42.5",
   "longitude": "1.6",
   "name": "Andorra"
}, {
   "countryCode": "AE",
   "latitude": "23.4",
   "longitude": "53.8",
   "name": "United Arab Emirates"
}, {
   "countryCode": "AF",
   "latitude": "33.9",
   "longitude": "67.7",
   "name": "Afghanistan"
}]
```

JSON in Python

□ https://docs.python.org/3.5/library/json.html



Compact encoding:

```
>>> import json
>>> json.dumps([1,2,3,{'4': 5, '6': 7}], separators=(',', ':'))
'[1,2,3,{"4":5,"6":7}]'
```

Pretty printing:

```
>>> import json
>>> print(json.dumps({'4': 5, '6': 7}, sort_keys=True, indent=4))
{
    "4": 5,
    "6": 7
}
```

Decoding JSON:

```
>>> import json
>>> json.loads('["foo", {"bar":["baz", null, 1.0, 2]}]')
['foo', {'bar': ['baz', None, 1.0, 2]}]
>>> json.loads('"\\"foo\\bar"')
'"foo\x08ar'
>>> from io import StringIO
>>> io = StringIO('["streaming API"]')
>>> json.load(io)
['streaming API']
```

DATA MANAGEMENT FOR BUSINESS INTELLIGENCE

Data Access: Relational Data Bases

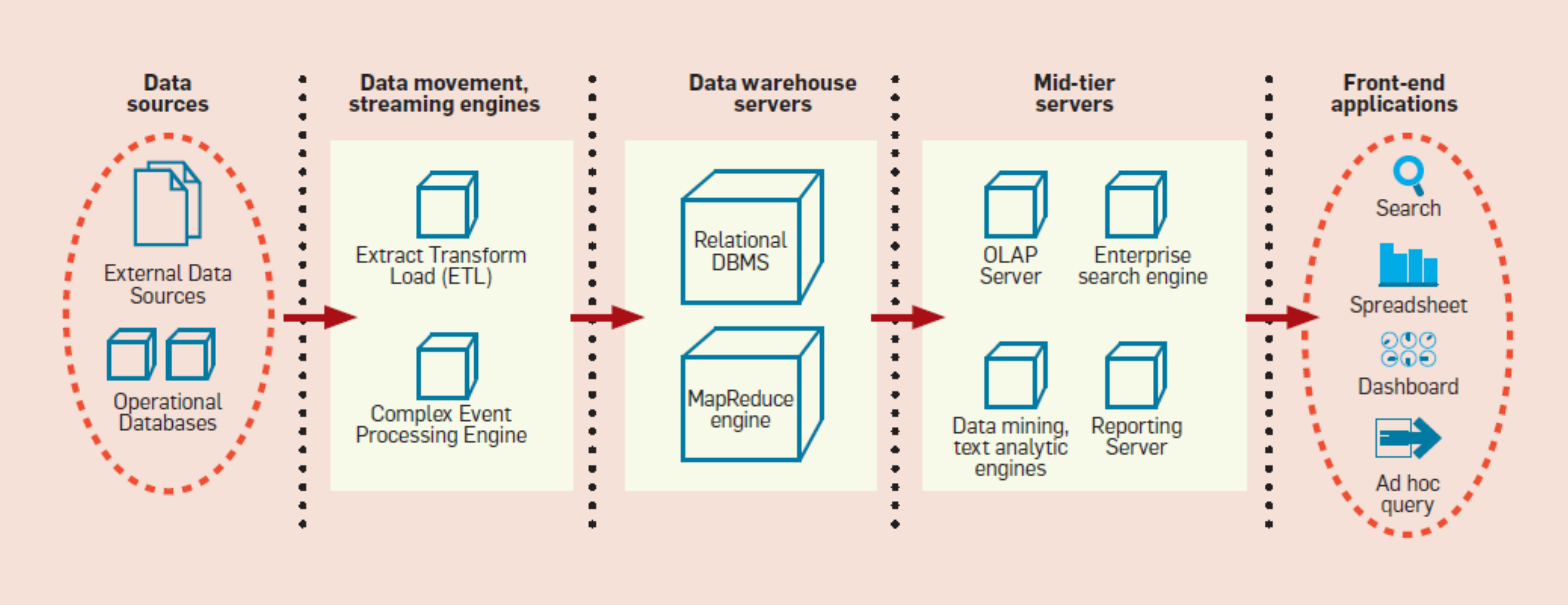
Salvatore Ruggieri

Computer Science Department, University of Pisa

Master in Big Data Analytics and Social Mining

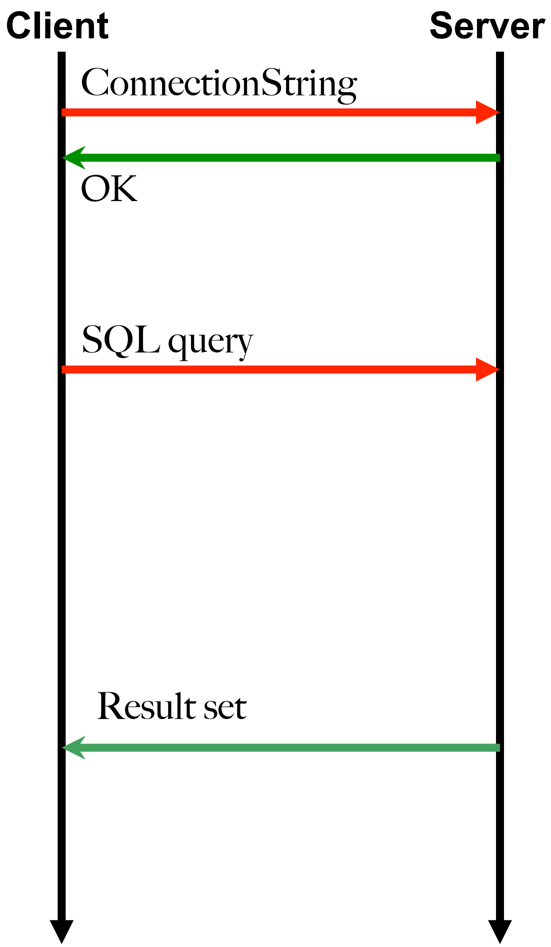
BI Architecture

Figure 1. Typical business intelligence architecture.



Connecting to a RDBMS

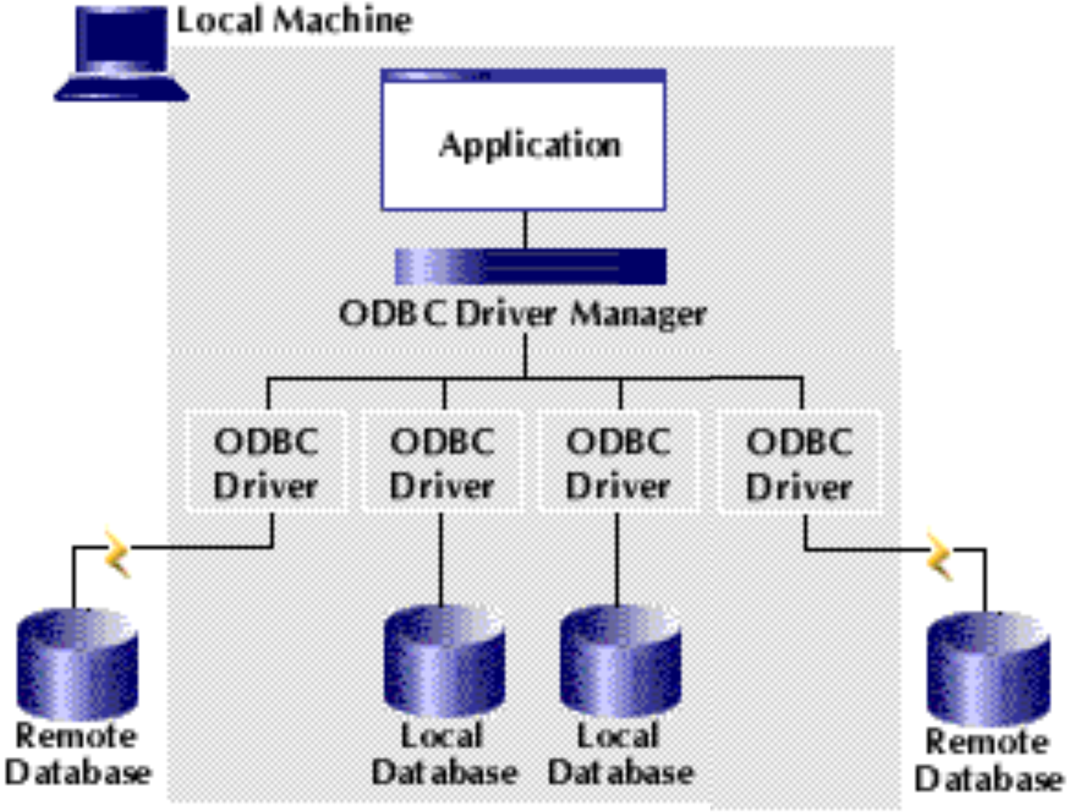
- **Connection protocol**
  - locate the RDBMS server
  - open a connection
  - user autentication
- **Querying**
  - query SQL
    - SELECT
    - UPDATE/INSERT/CREATE
  - stored procedures
  - prepared query SQL
- **Scan Result set**
  - scan row by row
  - access result meta-data



Connection Standards

- ODBC - Open DataBase Connectivity
  - Windows: odbc Linux: unixodbc, iodbc
  - Tabular Data
- JDBC
  - Java APIs for tabular data
- OLE DB (Microsoft)
  - Tabular data, XML, multi-dimensional data
- ADO (Microsoft)
  - Object-oriented API on top of OLE DB
- ADO.NET
  - Evolution of ADO in the .NET framework

ODBC Open DataBase Connectivity



ODBC Demo

- Registering an ODBC data source
- Data access
  - accessing Access data from Excel
- Linked tables
  - accessing Excel data from Access

OLE DB Demo

- Creating .udl data links
- Data access
  - accessing Access data from Excel
- Linked tables
  - accessing Excel data from Access
- OLE DB Drivers
  - By Microsoft
  - By other vendors

Python access to MySQL

```
import mysql.connector

cnx = mysql.connector.connect(user='scott',
           password='pisa', database='corsiinfo')
cursor = cnx.cursor()

query = "SELECT nome, cognome FROM studenti"

cursor.execute(query)

for (nome, cognome) in cursor:
  print(nome, " ", cognome)

cursor.close()
cnx.close()
```

Import module (driver)

Connect to DBMS

Submit query

Scan results

Close connection

DATA MANAGEMENT FOR BUSINESS INTELLIGENCE

ETL – Extract, Transform and Load

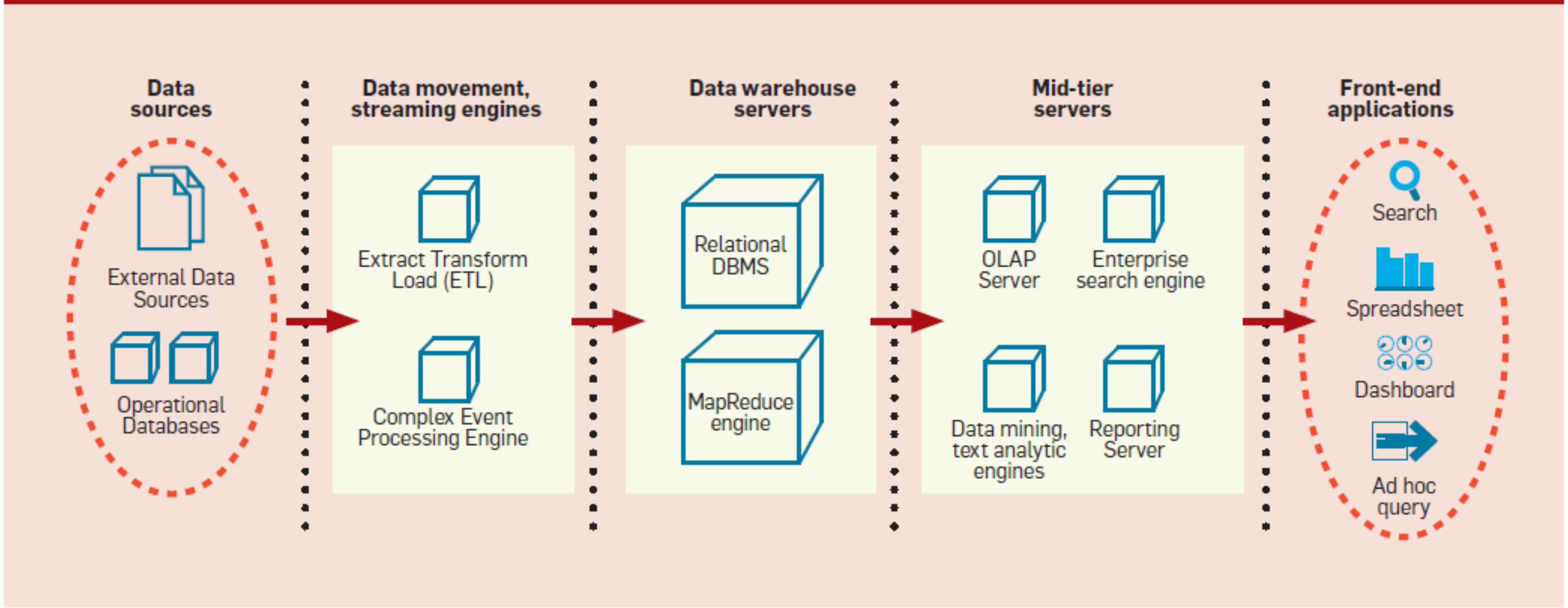
Salvatore Ruggieri

Computer Science Department, University of Pisa

Master in Big Data Analytics and Social Mining

BI Architecture

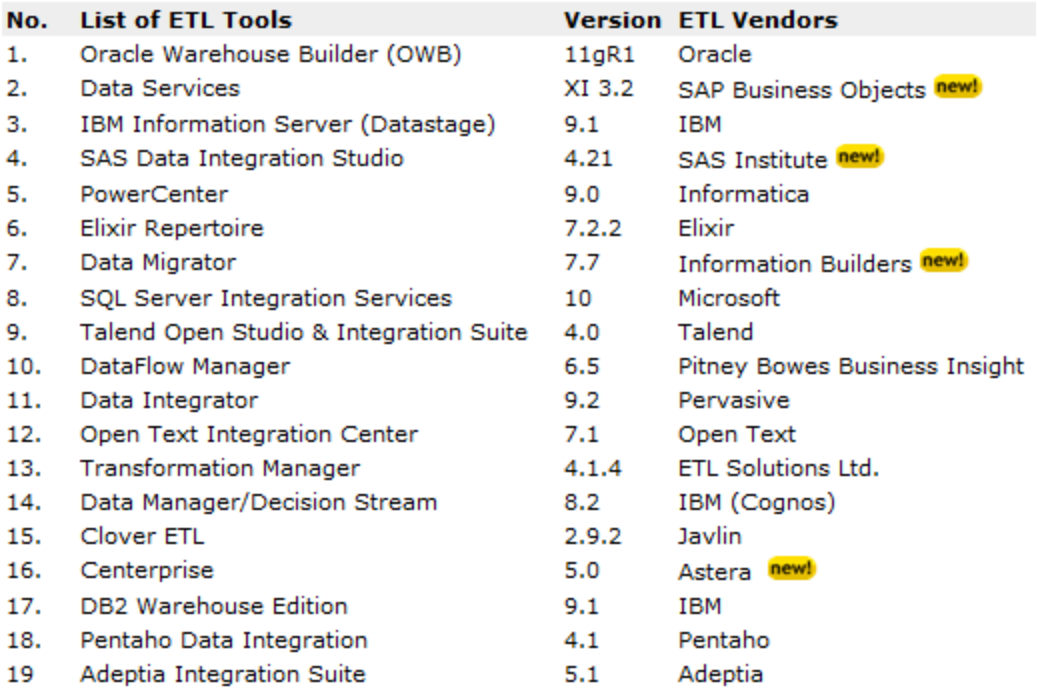
Figure 1. Typical business intelligence architecture.



Extract, Transform and Load

ETL (extract transform and load) is the process of extracting, transforming and loading data from heterogeneous sources in a data base/warehouse.

- Typically supported by (visual) tools.



No.	List of ETL Tools	Version	ETL Vendors
1.	Oracle Warehouse Builder (OWB)	11gR1	Oracle
2.	Data Services	XI 3.2	SAP Business Objects new!
3.	IBM Information Server (Datastage)	9.1	IBM
4.	SAS Data Integration Studio	4.21	SAS Institute new!
5.	PowerCenter	9.0	Informatica
6.	Elixir Repertoire	7.2.2	Elixir
7.	Data Migrator	7.7	Information Builders new!
8.	SQL Server Integration Services	10	Microsoft
9.	Talend Open Studio & Integration Suite	4.0	Talend
10.	DataFlow Manager	6.5	Pitney Bowes Business Insight
11.	Data Integrator	9.2	Pervasive
12.	Open Text Integration Center	7.1	Open Text
13.	Transformation Manager	4.1.4	ETL Solutions Ltd.
14.	Data Manager/Decision Stream	8.2	IBM (Cognos)
15.	Clover ETL	2.9.2	Javlin
16.	Centerprise	5.0	Astera new!
17.	DB2 Warehouse Edition	9.1	IBM
18.	Pentaho Data Integration	4.1	Pentaho
19	Adeptia Integration Suite	5.1	Adeptia

ETL tasks

- Extract: access data sources
  - Local, distributed, file format, connectivity standards

- Transform: data manipulation for quality improvm
  - Selecting data
    - remove unnecessary, duplicated, corrupted, out of limits (ex., age=999) rows and columns, sampling, dimensionality reduction
  - Missing data
    - fill with default, average, filter out
  - Coding and normalizing
    - to resolve format (ex., CSV, ARFF), measurement units (ex., meters vs inches), codes (ex., person id), times and dates, min-max norm, …
  - Attribute Splitting/merging
    - of attributes (ex., address vs street+city+country)

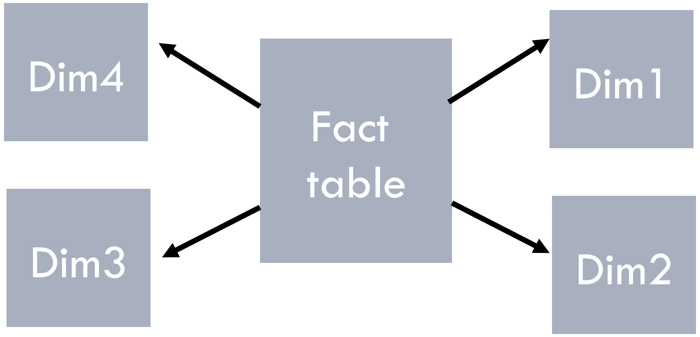
ETL tasks

- Managing surrogate key & Slowly changing dimensions
  - generation and lookup
- Aggregating data
  - At a different granularity. Ex., grain "orders" (id, qty, price) vs grain"customer" (id, no. orders, amount), discretization into bins, …
- Deriving calculated attributes
  - Ex., margin = sales – costs
- Resolving inconsistencies – record linkage
  - Ex., Dip. Informatica Via Buonarroti 2 is (?) Dip. Informatica Largo B. Pontecorvo 3
- Data merging-purging
  - from two or more sources (ex., sales database, stock database)

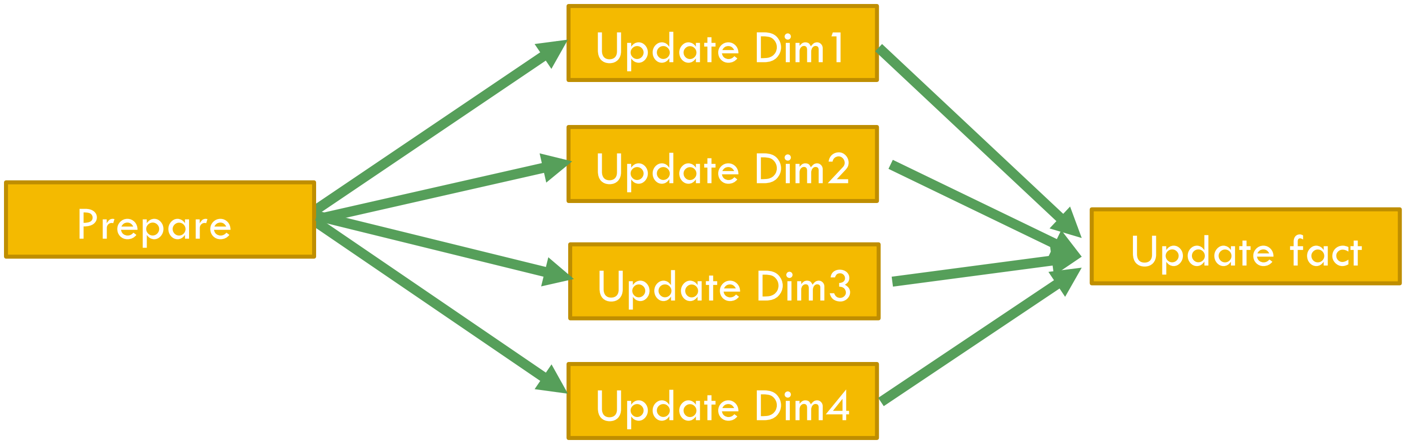
ETL tasks

- Load
  - Data staging area
    - Area containing intermediate, temporary, partially processed data
  - Types of loading:
    - Initial load (of the datawarehouse)
    - Incremental load
      - Types of updates: append, destructive merge, constructive merge
    - Full refresh

ETL process for DW

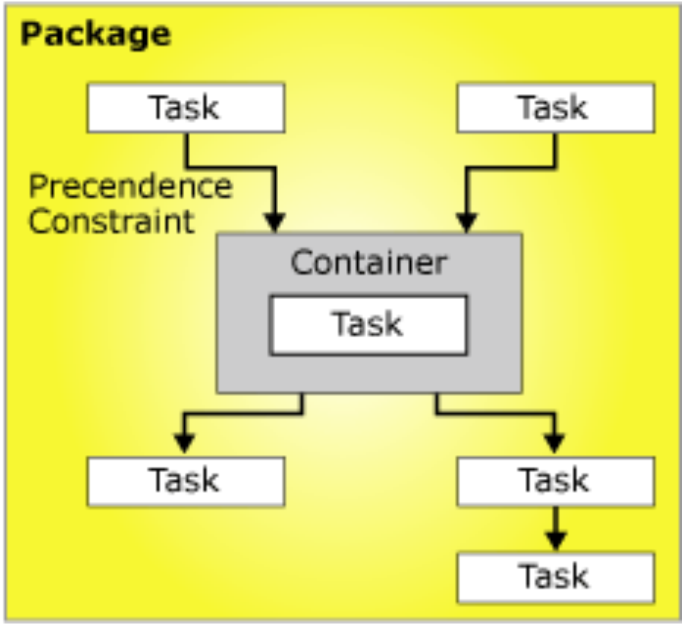


Control Flow



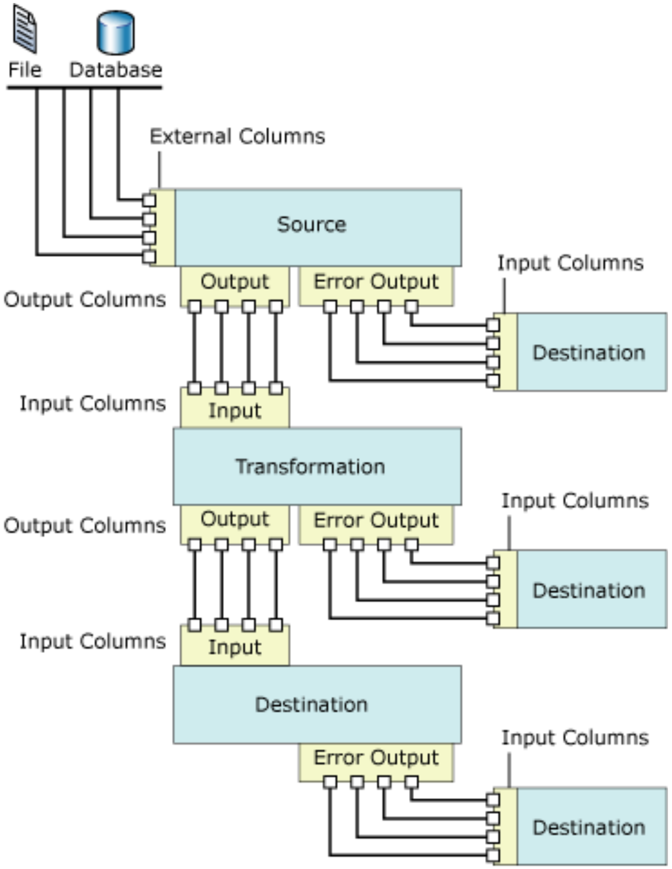
Control flow / Jobs

- Tasks & Precedence
  - Tasks
    - E.g. data flows / transformations
  - Container
    - For grouping and iteration
  - Precedence
    - Arrows connecting tasks specify precedence type

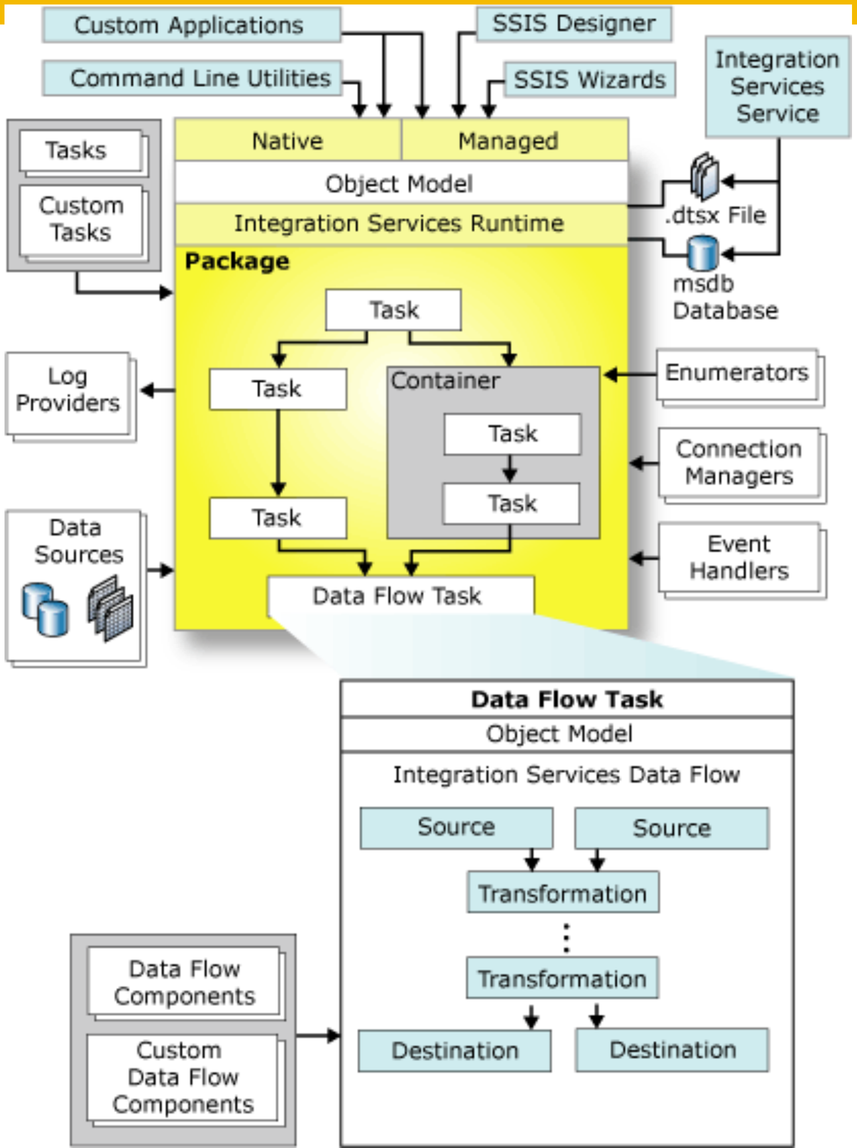


Special tasks: data flow / transformations

- Define pipelines of data flows from sources to destination
  - Data flow sources
  - Data flow transformation
  - Data destination
  - Toolbox panel for list



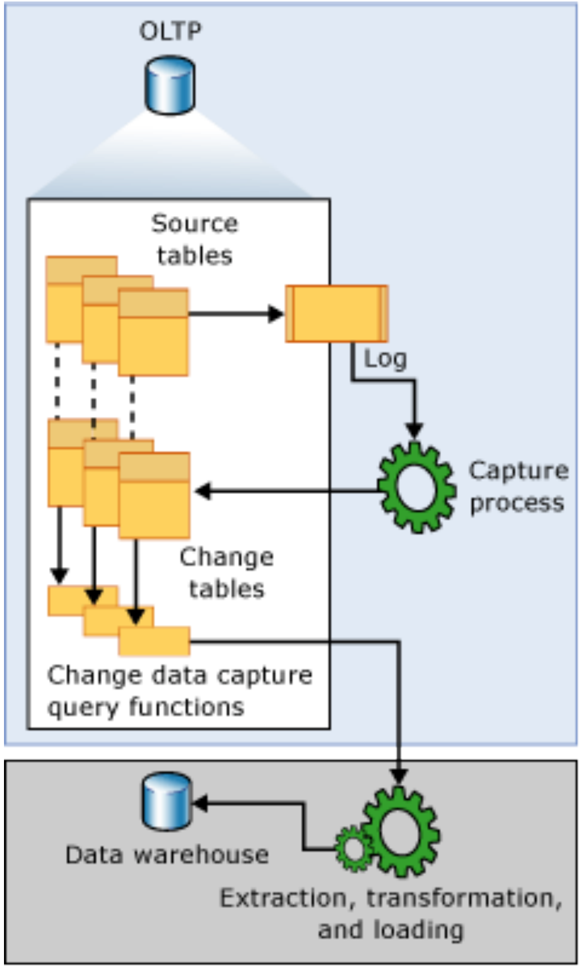
ETL projects structure



Data types

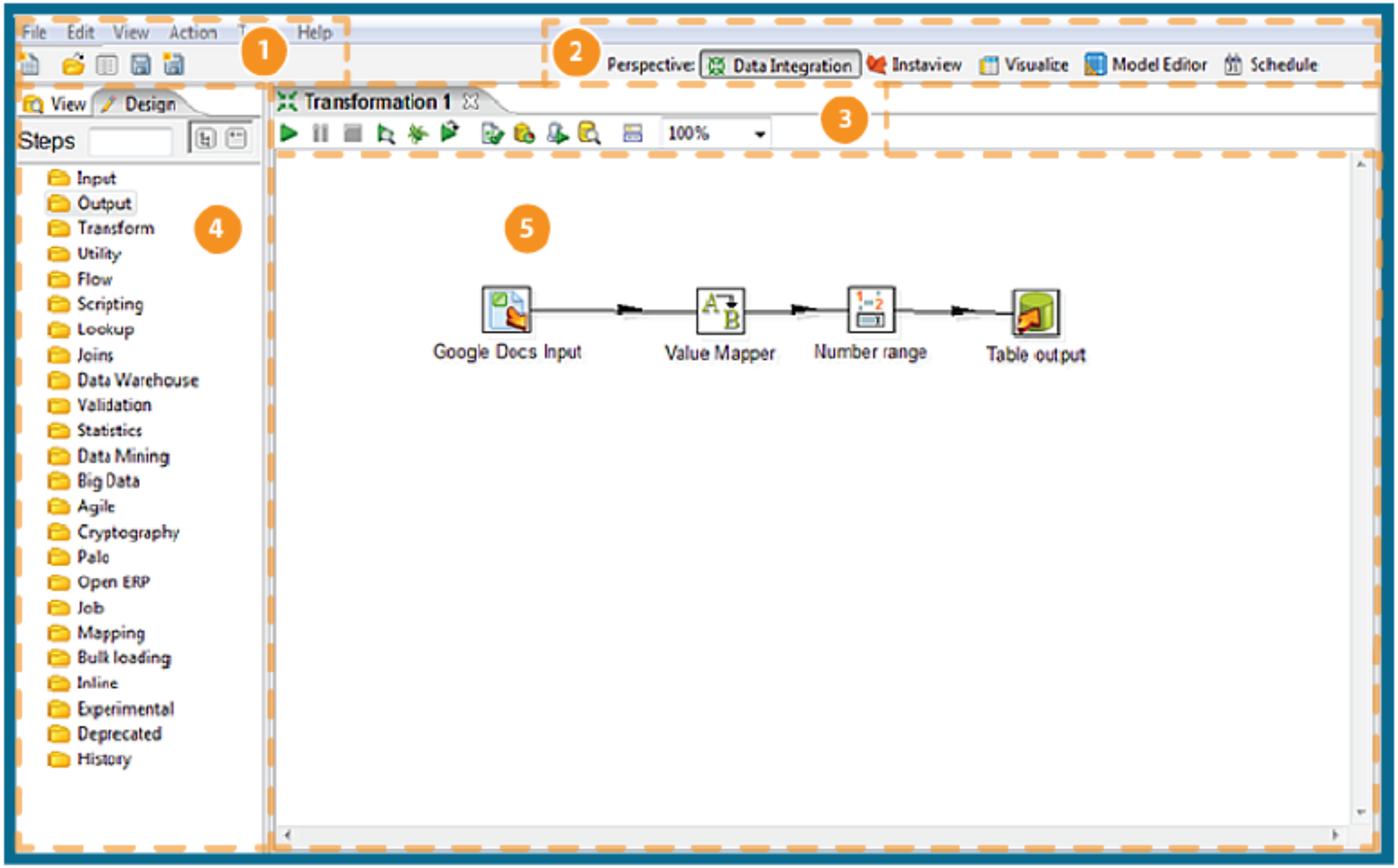
- ETL tools define a set of reference data types
- Data type from sources are mapped into ETL types
- ETL transformations work on ETL types
- ETL types are mapped to destination data types

Change data capture


OLTP
Source tables
Log
Capture process
Change tables
Change data capture query functions
Data warehouse
Extraction, transformation, and loading

Pentaho Data Integration - Demo

Tour Spoon



BUSINESS INTELLIGENCE LABORATORY

ETL Demo: Pipeline, Sampling and Surrogate Keys

Pipeline

- Consider the Foodmart sales database
- Design an ETL project for writing to a CSV file the list of products ordered descending by gain
  - Gain of a single sale is defined as (store_sales – store_cost)*unit_sales
  - Gain of a product is the sum of gains for all product sales
- Do not use views or queries! Do all work in ETL.

Pipeline

- Consider the SAKILA database
- Design an ETL project for writing to a CSV file the list of customers descending by total gain
  - Gain of a single sale is defined as (amount – rental_cost) where the rental_cost is set to 10% of the amount
  - Total Gain wrt a customer is the sum of gains for all customer rental
- Do not use views or queries! Do all work in ETL.

Stratified subsampling

- Consider the census table in the MasterBigData db
- Design an ETL project for writing to a CSV a random sampling of 30% stratified by sex
  - 30% of males plus 30% of females
- Do not use views or queries! Do all work in ETL.

BUSINESS INTELLIGENCE LABORATORY

Lab exercise on ETL: SCD

Business Informatics Degree

SCD: background

- Slowly Changing Dimensions
  - Datawarehouse dimensions members updates
  - Three types:
    - Type 1: overwrite previous value
    - Type 2: keep all previous values
    - Type 3: keep last N previous values (N ~ 1, 2, 3)
  - Each attribute of the dimension can have its own type
    - Type 1: name, surname, …
    - Type 2: address, …

SCD: input and output tables

- Database SAKILA in MySQL
- Input
  - table customer
- Output in the MAINS database
  - create a table <surname>_customer_dim
    - columns
      - surrogate_key (PK), customer_id, customer_name, address, date_start, date_end
    - with
      - surrogate_key being a surrogate key, customer_name including name and surname, address made of address-city, date_start and date_end are dates

SCD: type 1 updates

- Overwrite previous value
- Changes on the input table customer
  - On 10/3/2007
    - 231, Maria Miller, 900 Santiago de Compostela Parkway
  - On 12/3/2007
    - 231, Mary Miller, 900 Santiago de Compostela Parkway
  - Name has been corrected

SCD: type 2 updates

- Keep all previous values
- Changes on the input table customer
  - On 12/3/2007
    - 231, Maria Miller, 900 Santiago de Compostela ParkwayOn 25/9/2008
    - Maria Miller, 100 Santiago de Compostela Parkway
    - Customer has changed his address

SCD: type 2 updates

- The DW <surname>_customer_dim table looks as:

surrogate_key, customer_id, name, address, date_start, date_end

874, 231, Maria Miller, 900 Santiago de Compostela Parkway, 10/3/2007, 25/9/2008

987, 231, Maria Miller, 100 Santiago de Compostela Parkway, 25/9/2008 NULL

Today exercise

- Design an ETL project to update <surname>_customer_dim starting from customer as follows:
  - Customers in customer that are not in <surname>_customer_dim are added to it
  - Updates of customer_name are of Type 1
  - Updates of address are of Type 2