



Environmental Sensing

Ilist - Indexed List

Concepts and principles

0 – Main

- **Indexed list**

- 0 - Presentation

- **Objectives**

- Structure optimization
 - 1 - Structure understanding
 - 2 - Structure optimization
 - 3 - Size optimization
 - Integrate process
 - 4 - Building process
 - 4 - Interface tools

- **Associated tools**

- 5 - Data format
 - 5 - Exchange format

- **Extension**

- 6 - Environmental sensing
 - 6 - Sensor acquisition

0 - Tabular data

Structure

List of values :

+

Age : [12, 28, 39, 58]

List of indexes :

Name : [Paul, John, Lea, Cat]

City : [Paris, Metz, Rennes, Bollène]

....



Name	city	Age
Paul	Paris	12
John	Metz	28
Lea	Rennes	39
Cat	Bollène	58

Example : csv file, measurement, log, matrix

Why list ?

- The majority of work processes are underpinned by Sheets
- The main Open-data format is CSV
- Existing tools process data but not data structures

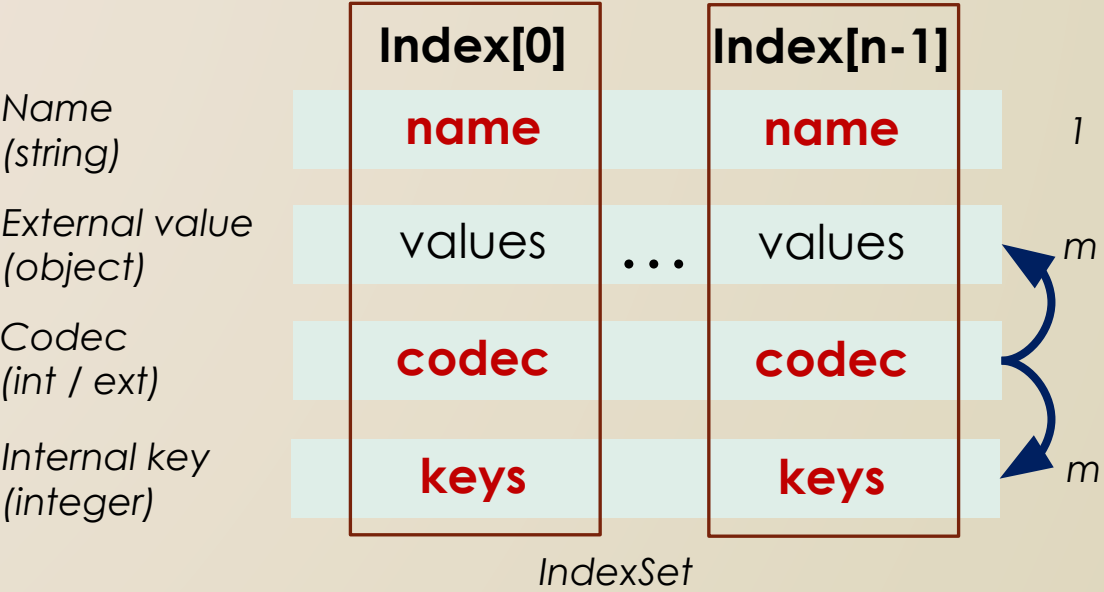
Such tool doesn't exist !

0 – Ilist structure

Red : static value n : number of indexes
Black : dynamic value m : number of values

Two levels

- **External values**
(every kind of object)
- **Internal keys**
(integer)



Example

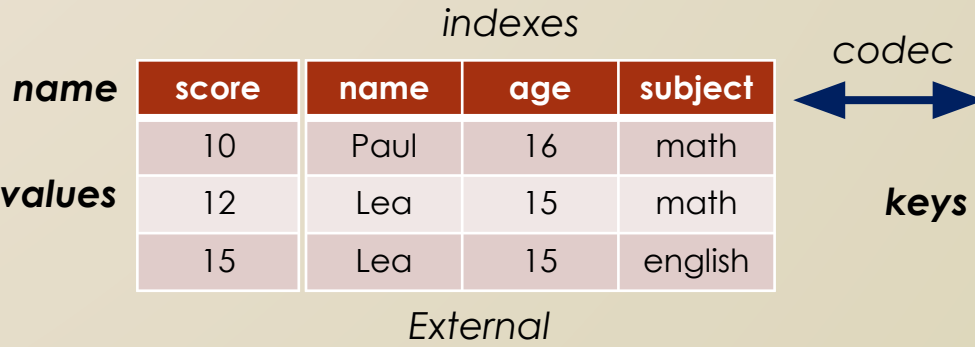
First name

[Anne, Paul, John, Paul]

[Anne, Paul, John]

[0, 1, 2, 1]

Example



Structure analysis

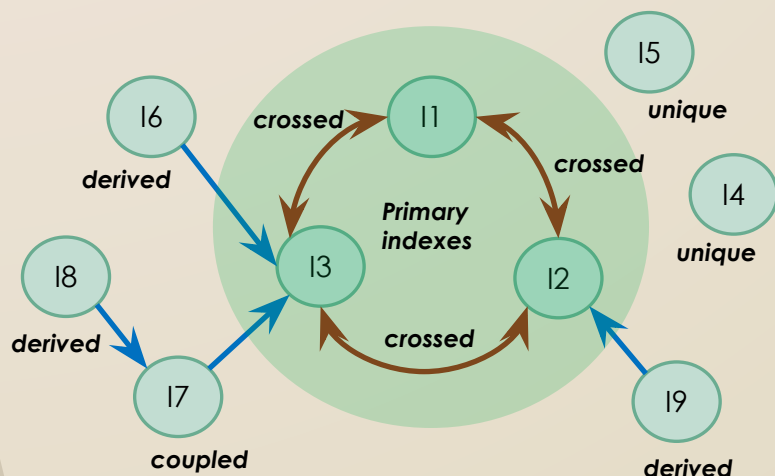
1 – Structure understanding

- **Relationship analysis**

- Index qualification
- Index relationship

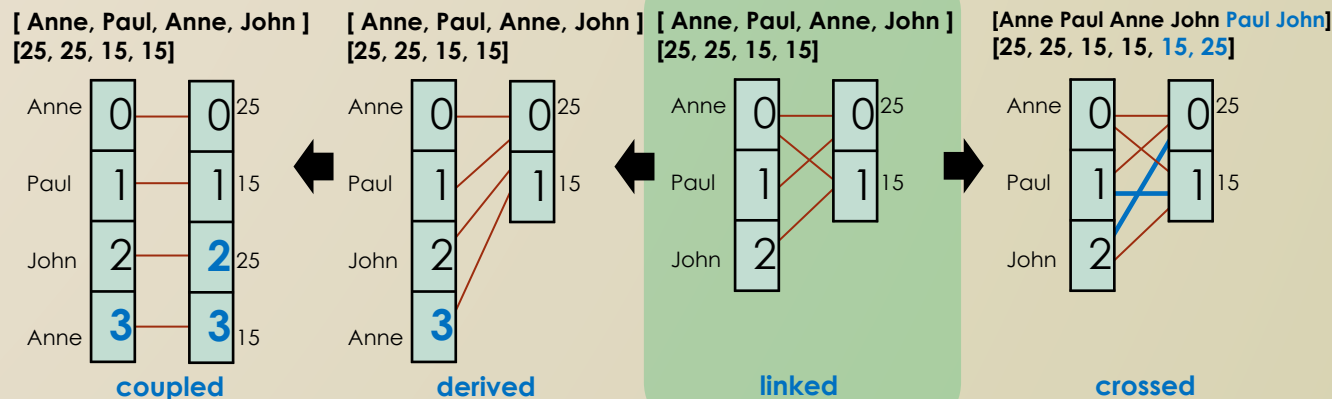
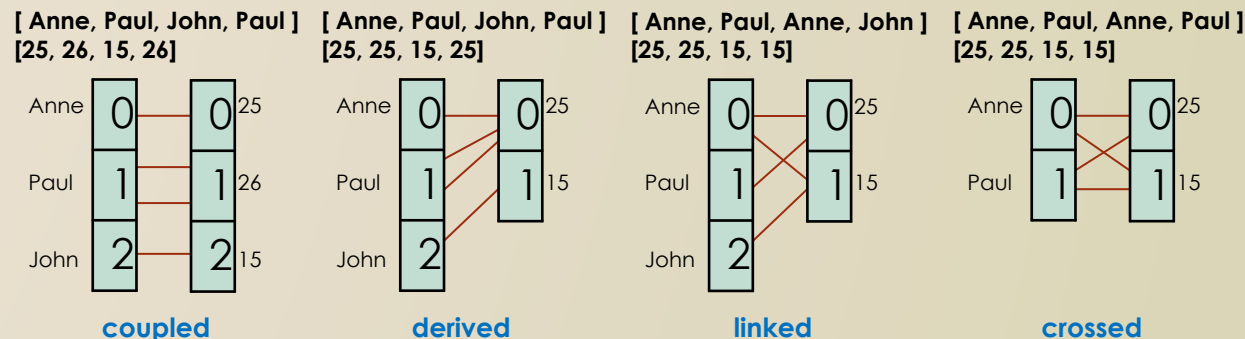
- **Data structuration**

- Canonical format
- Convergence



Canonical structure

Index relationship

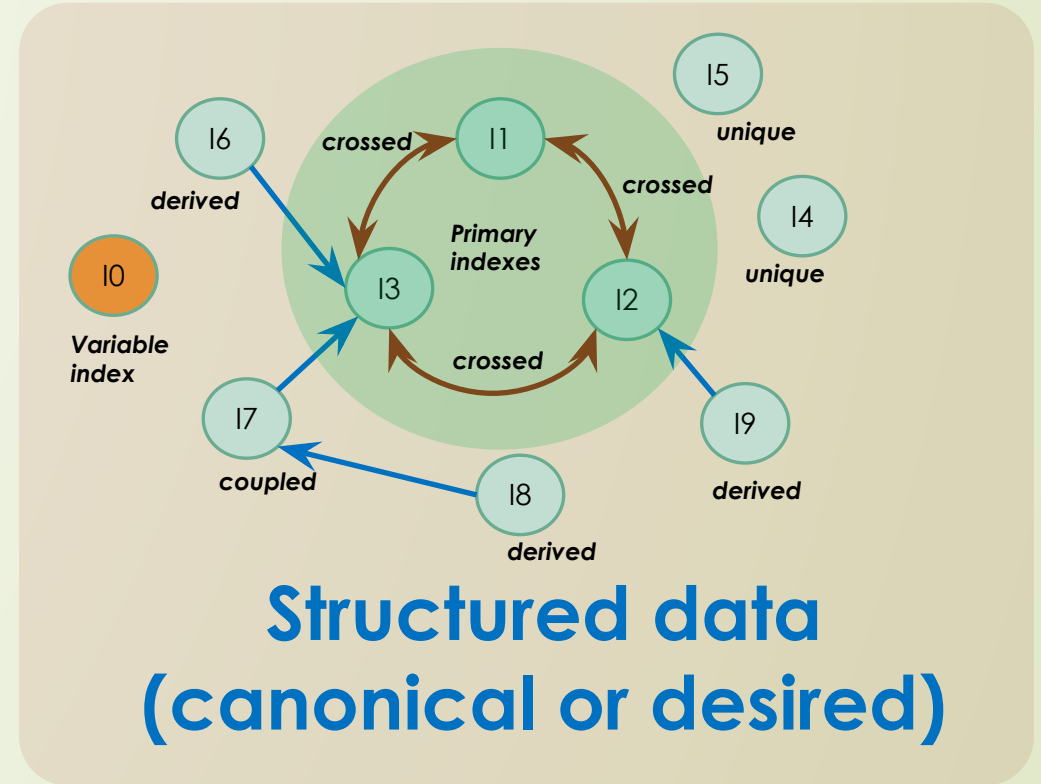
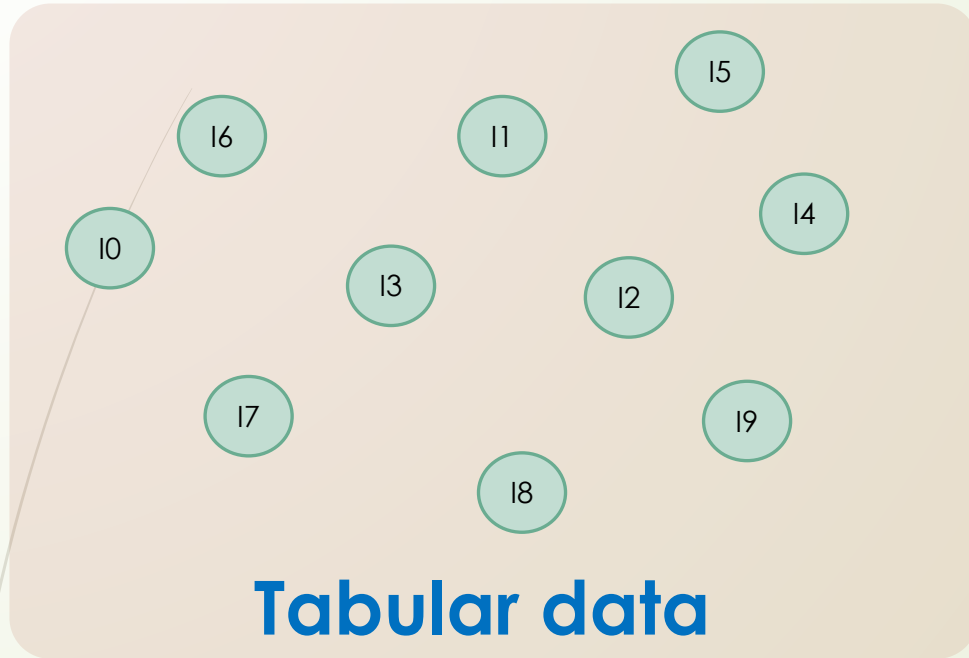


Codec extension

Value extension

Convergence

2 – Structure optimization



- **Optimization**

- minimization of additional data to achieve canonical structure

- **Consistency**

- enforce compliance with the conceptual data model (e.g. cardinality)
- identification of additional data to achieve the desired structure

3 – Size optimization

- **Canonical structure**

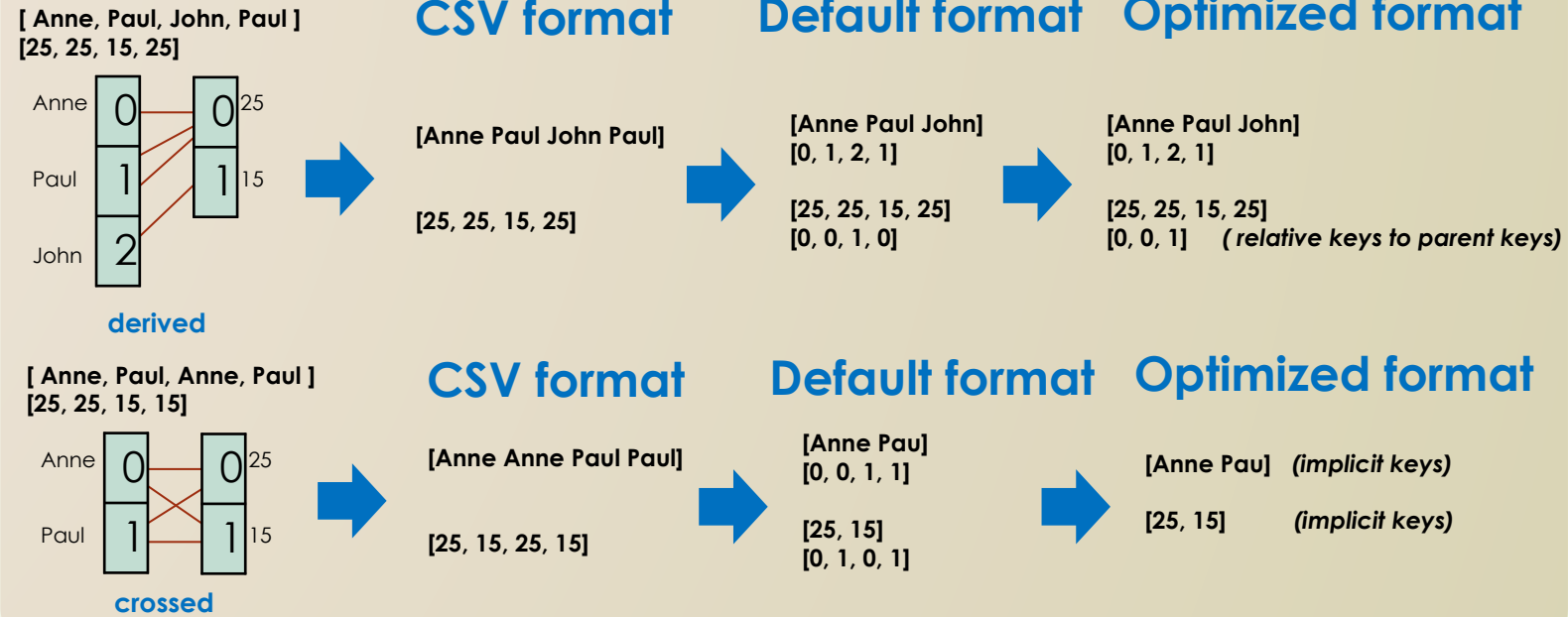
- Minimal structure

- **Minimal size**

- No multiple value
- Keys optimization

- **Exchange format**

- Text : JSON format
- Binary : CBOR (RFC 8949)



Example : Open-data - french charging point (EVSE)

7.5 Mo – 11 000 rows – 49 columns

Analysis :

Indexes : 1 coupled, 6 derived, 1 crossed, 41 linked

Canonical format : 1 crossed, 48 derived

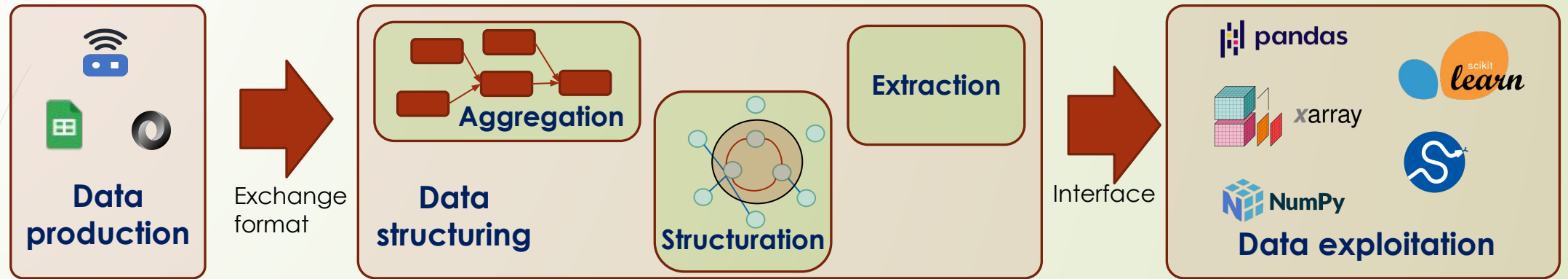
File size :

Default : 3.7 Mo

Optimized : 2.5 Mo

CBOR optimized : **1.7 Mo (gain : 77% !)**

4 – Integrate process



- **Data production interface**

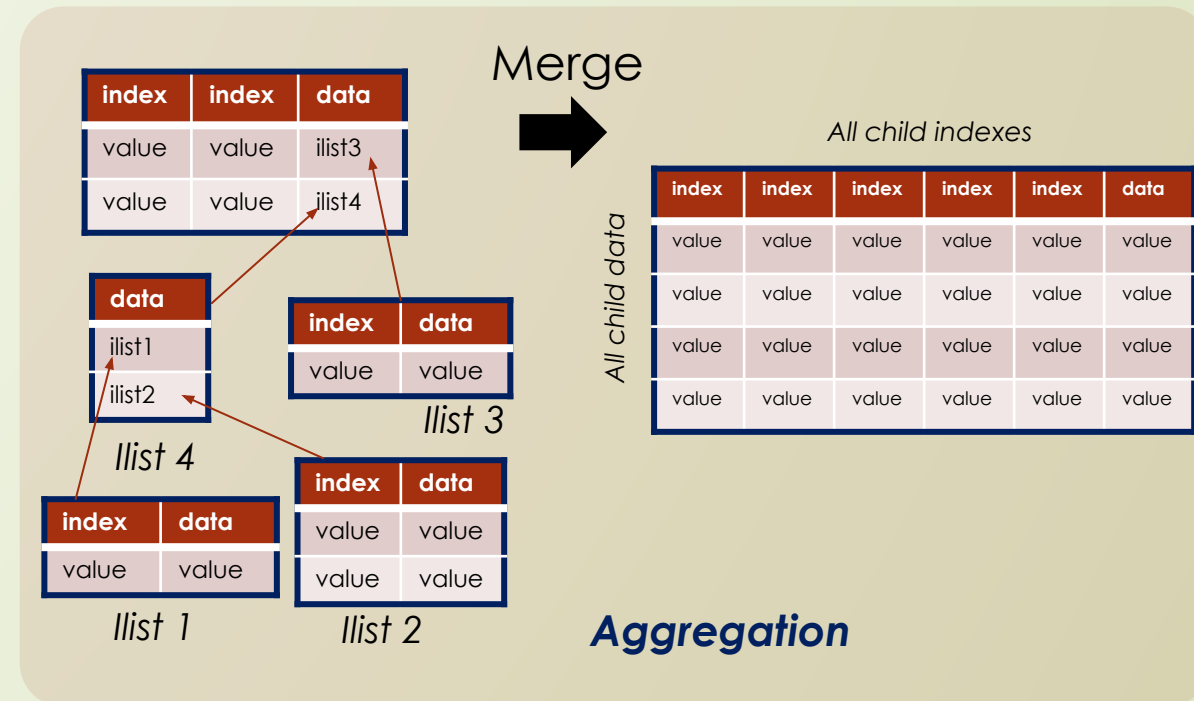
- Exchange format (Json, Bluetooth, CSV)

- **Aggregation / merge functions**

- Adapted to projects / organizations
- Add information without altering

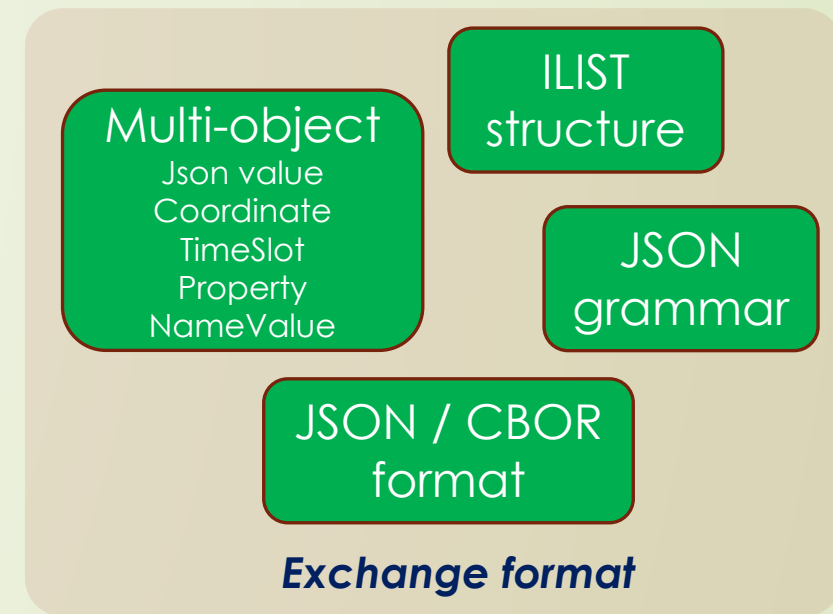
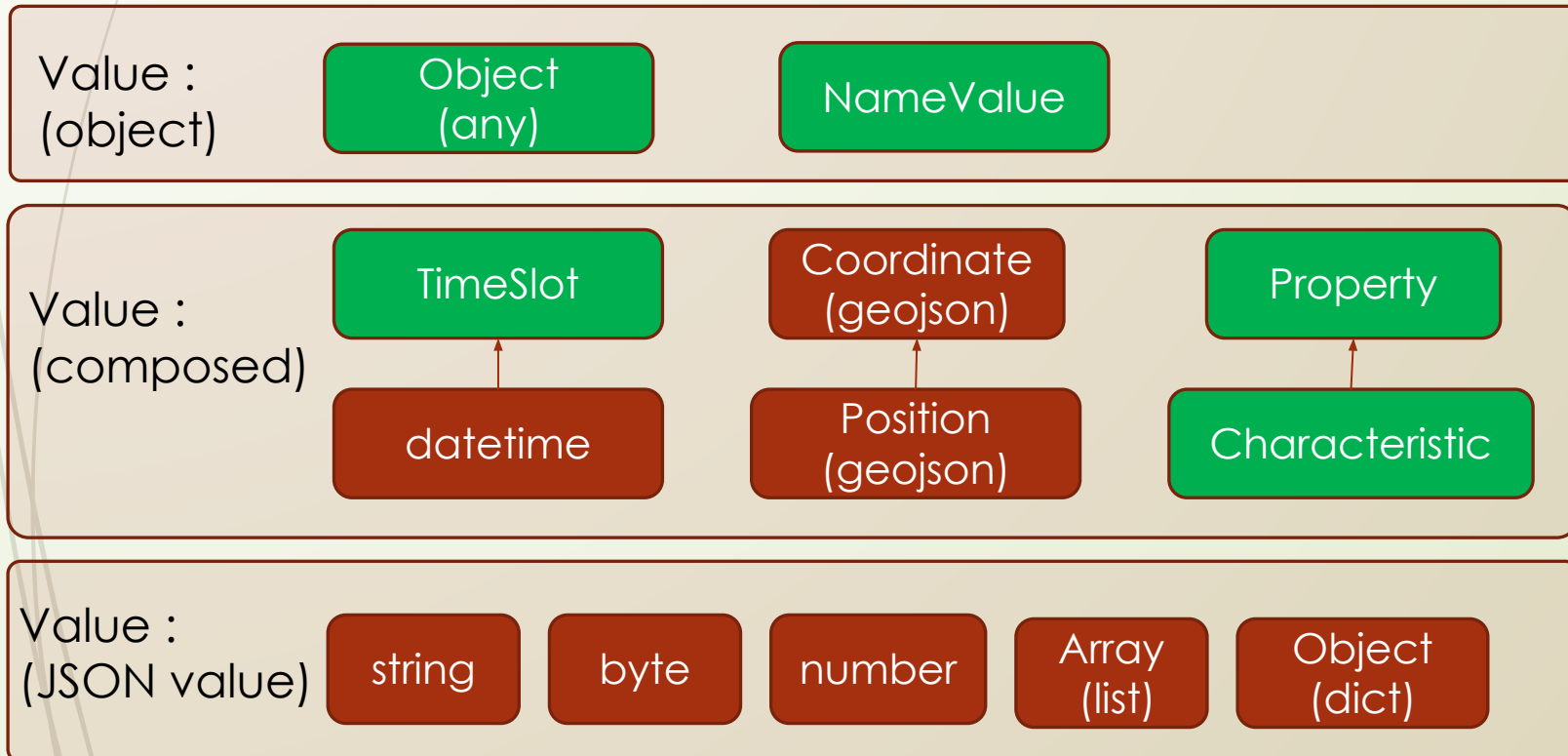
- **Export to analysis tools**

- Canonical structure compatibility



5 – Data format

- **Large set of objects**
 - New format (timeslot, property)
- **JSON representation**
 - Exchange format




NameValue

{ 'Paris' : [2,4, 48,9] }

Object

{ 'object name': object value }

TimeSlot

 [[[d1,d2]], [[d3,d4]]]

Property

{ 'char': 'PM10', 'unit': 'kg/m3', ... }

(Char -> i. e. BLE characteristic)

6 – Ilist extension

- **Observation**

- Ilist specialization with three main indexes :
 - Datation index (Timeslot), Location index (coordinate), Property
- Conformance with ISO 19156 : Observation & Measurement

- **Sensor acquisition**

- Integration of Bluetooth Environmental Sensing Profile (extension in 2021)
- Reduced exchange format for micro-controllers

- **Open-data**

- Tool to define data structuring (e.g. tabular data)
- Consistency measurement tool (e.g. check data model cardinalities)



Appendix

Concepts and principles

Detailed presentation

1 - Index analysis

2 - Matrix generation

3 - Aggregation

4 – Format, storage

0 – Terminology

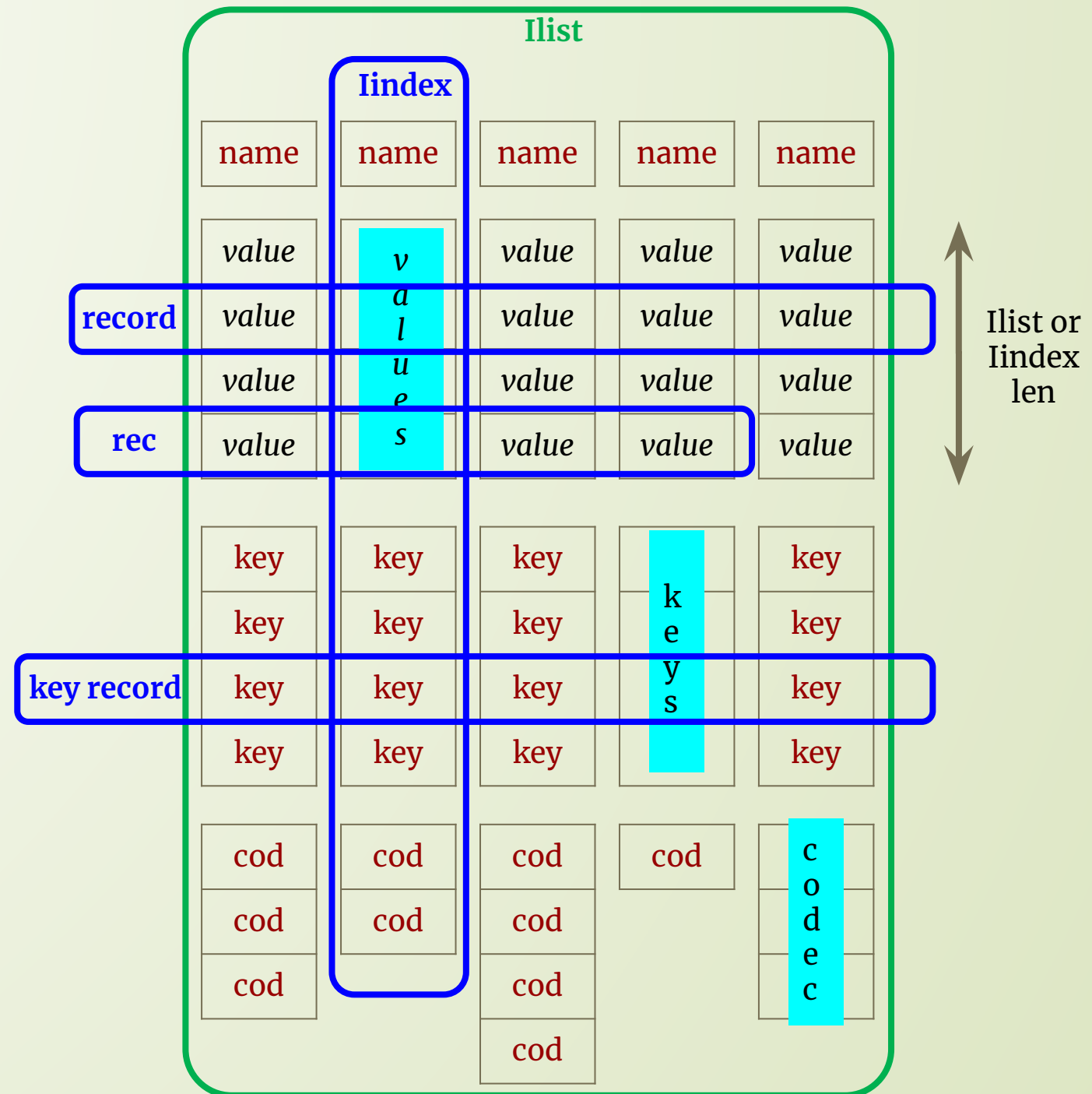
Ilist

- list of Iindex

Iindex

- name
- keys
- codec

Italic: dynamic value



1 - index categories

Values Length (number of values)	[Anne, Paul, Anne]	[Anne, Anne, Anne]	[Anne, Paul, Anne]
	3	3	3
Codec (row)	<div>0</div> <div>1</div> <div>2</div>	<div>0</div>	<div>0</div> <div>1</div>
Type codec	full	unique	default
Property	Rate : 1 Disttomax : 0	Rate : 0 Disttomin : 0	Rate : 0 Disttomin : 0
Representation	Codec : [Anne, Paul, Anne] Keys: implicit ([0,1,2])	Codec : [Anne] Keys: implicit ([0, 0, 0])	Codec : [Anne, Paul] Keys: [0, 1, 0]

Definition :

Default codec :
list of different values
Full codec :
list of values

Indicators :

M = len(values)
m = len(set(values))
x = len(codec)

Rate : $(M - x) / (M - m)$
Dist to min : $x - m$
Dist to max : $M - x$

Codec typology

$M = 0$	null
$m = M = x > 1$	complete
$x = 1$	unique
$m < M = x$	full
$x = m < M$	default
$m < x < M$	mixed

A codec defines the correspondence between values and keys (e.g.) :

- 1 : Anne
- 0 : Paul
- 2 : John

A codec may not be bijective (e.g.) :

- 0 : Anne
- 1 : Paul
- 2 : Anne

1 - properties

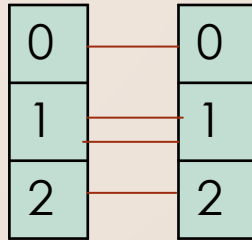
- Any index have default codec and full codec
- Default codec are the shortest codec, full are the longest codec
- index with raw number is the “root lindex”, this lindex is complete

1 - Relationship categories

Values

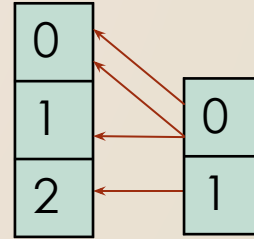
A [Anne, Paul, John, Paul] [Anne, Paul, Anne, Lea] [Anne, Paul, Anne, Lea] [Anne, Anne, Anne,
B [25, 26, 15, 26] [25, 25, 25, 12] [25, 25, 12, 12] Paul, Paul, Paul]
[25, 12, 25, 12, 25, 12]

Codec (row)

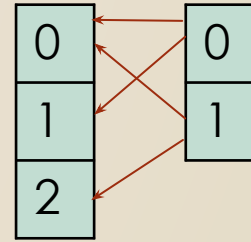


A

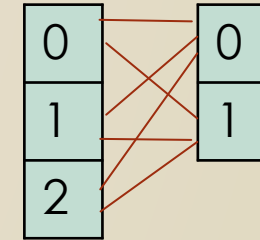
B



B derived from A
A derive B



B linked from A
A link B



A

B

Type

coupled

derived
(asymmetrical)

linked
(asymmetrical)

crossed

Property

Rate : 0
Disttomin : 0
diff : 0

Rate : 0
Disttomin : 0
0 < diff < min

0 < Rate < 1
min < dist < Max
0 ≤ diff ≤ min

Rate : 1
Disttymax : 0
0 ≤ diff < min

Keys

B Implicit
(equal Keys A)

Relative
to keys A

Absolute

Implicit
(Matrix order)

Codec : [25, 26, 15, 35]
Keys: implicit

Codec : [25, 12]
Keys: relative
[0,0,1]

Codec : [25, 12]
Keys: absolute
[0,0,1,1]

Codec : [25, 12]
Keys: implicit
([0,1,0,1,0,1])

Indicators :

$\max = x_A * x_B$
 $\min = \max(x_A, x_B)$
 $\text{diff} = \text{abs}(x_A - x_B)$
 $\text{dist} = \text{len}(\text{index}(v1, v2))$

Rate: $(\text{dist} - \min) / (\max - \min)$
Dist to min : $\text{dist} - \min$
Dist to coup: $2 * \text{dist} - 2 * \min + \text{diff}$
Dist to max : $\max - \text{dist}$

Rules :

B derived from A
 $\text{dist} = x_A$ and $\text{dist} > x_B$
A derived from B
 $\text{dist} = x_B$ and $\text{dist} > x_A$
A and B coupled
 $\text{dist} = x_B = x_A$
A and B crossed
 $\text{dist} = x_B * x_A$

Relative derived keys :

Length:

- $\text{length}(\text{parent.codec})$

Values:

- $\text{Keyder}(\text{parent.key}(i)) = \text{key}(i)$

1 - relationship properties

- Type and Indicators are independent of Values (order or value) and dependant of Codec and Keys
- If one index is complete, all the indexes are derived or coupled from it
- If one index is unique, it is derived from all other indexes
- If A is derived (coupled) from B and B is derived (coupled) from C, A is derived (coupled) from C and $\text{diff}(A,C) = \text{diff}(A,B) + \text{diff}(B,C)$
- If A and B are coupled, all the relationships with other indexes are identical
- If A and B are crossed and C is derived (coupled) from A: B and C are crossed
- If A and B are crossed: $x_A * x_B \leq M_A$
- **Keys can be deduced with coupled relationship**
 - A and B are coupled $\Rightarrow \text{keys}(B) = \text{keys}(A)$
- **Keys can be reduced with derived relationship**
 - B is derived from A $\Rightarrow \text{len}(\text{keys}(B)) = \text{len}(\text{codec}(A))$

1 – Relationship adjustment

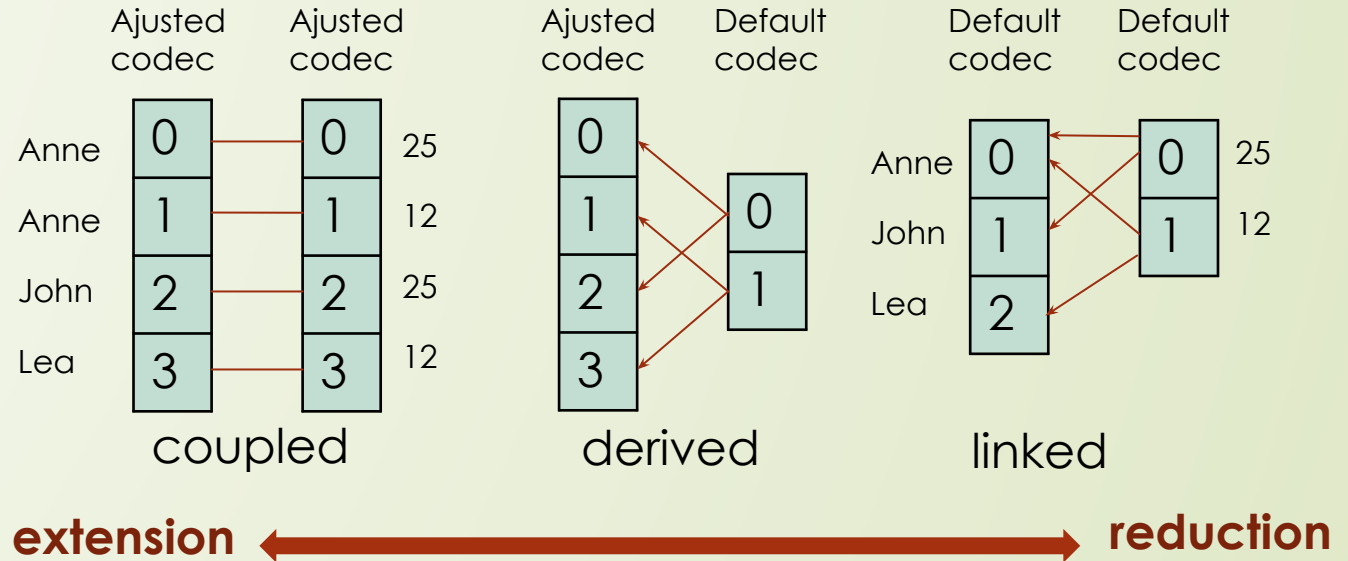
- **Codec reduction / extension**

- Codec changed
- Values unchanged

Reduction is useful to minimize codec size

Extension is useful to increase values readability (like csv data)

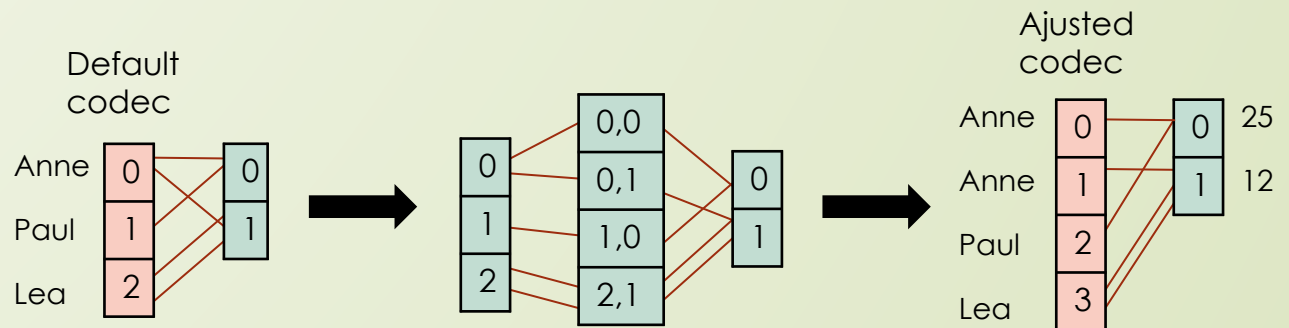
[Anne, Anne, John, Lea]
[25, 12, 25, 12]



- **Codec adjustment**

- Codec is adjusted to the other codec
- Other index is derived or coupled to the adjusted index
- If A is derived from B and if B is adjusted to C, A is still derived from B

Keys can be deduced from keys parent



1 – Relationship adjustment

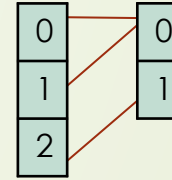
- **Values reduction / extension**

- Codec unchanged
- Values changed

**Extension is useful to
generate matrix**

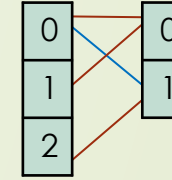
**Reduction is useful to
increase codec readability**

[Anne,Paul,Lea]
[25, 25, 12]



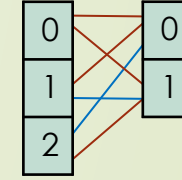
derived

[Anne,Paul,**Anne**,Lea]
[25, 25, **12**, 12]



linked

[Anne,Paul,**Lea**,Anne,**Paul**,Lea]
[25, 25, **25**, 12, **12**, 12]



crossed

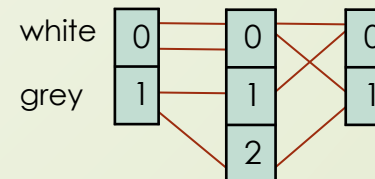
reduction ← → **extension**

- **Propagation**

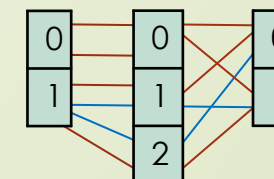
- Values reduction / extension can be propagated to derived or coupled indexes

**Extension can't be
propagated to crossed or
linked Indexes.**

[White, Grey, White, Grey]
[Anne, Paul, Anne, Lea]
[25, 25, 12, 12]



[White, Grey, **Grey**, White, **Grey**, Grey]
[Anne, Paul, **Lea**, Anne, **Paul**, Lea]
[25, 25, **25**, 12, **12**, 12]



2 – llist

- **llist structure**

- An llist object is a set of lindex with the same length
- llist have a hidden lindex : **root lindex**
- lindex are not ordered

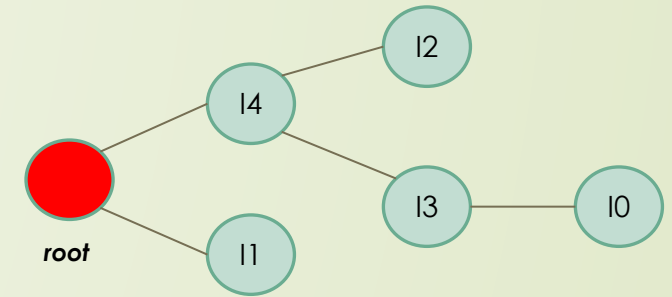
- **llist partition**

- A **partition** is a set of lindex where each root value is associated to a single combination of lindex key (coordinate structure / multi dimensional array)
- Three categories of lindex are associated with a partition :
 - **Primary lindex** : included in the partition
 - **Secondary lindex** : child lindex of primary or secondary lindex
 - **Variable lindex** : parent lindex of primary lindex + parent or child lindex of variable lindex
- The **dimension** of an llist is the maximal number of lindex in a partition

2 – Ilist properties

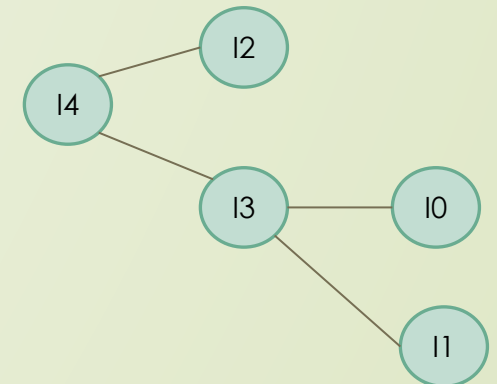
- **Ilist derived tree**

- Each index is derived (coupled) from at least one index (the root index)
- An Ilist can be represented by a hierarchical tree structure
 - Nodes are index
 - Root is root index
 - Parent node is the deriving index with minimal diff



- **Ilist rate tree**

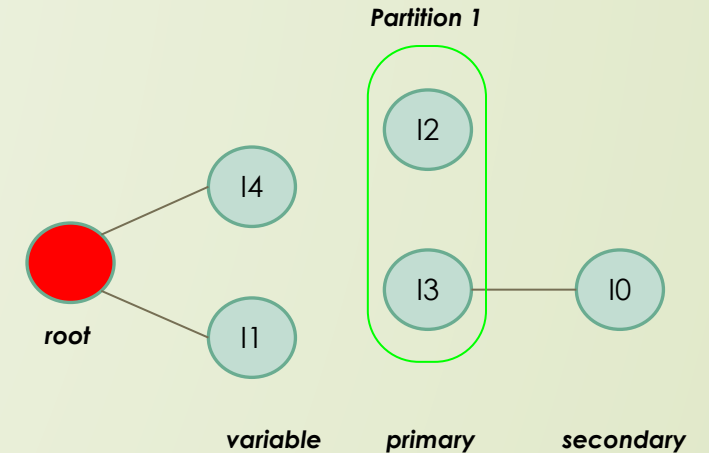
- The rate is a distance between two index
- An Ilist can be represented by a hierarchical tree structure
 - Nodes are index
 - Root is root index
 - Parent node is the index with minimal (or maximal) rate



2 – llist properties

- **llist partition**

- A set of index is a partition if :
 - Each index is crossed with each other
 - The product of index codec length is equal to the length of the llist
 - The set is 'consistent' (each record is unique)
- An llist have at least one implicit partition (the root partition)



- **Properties**

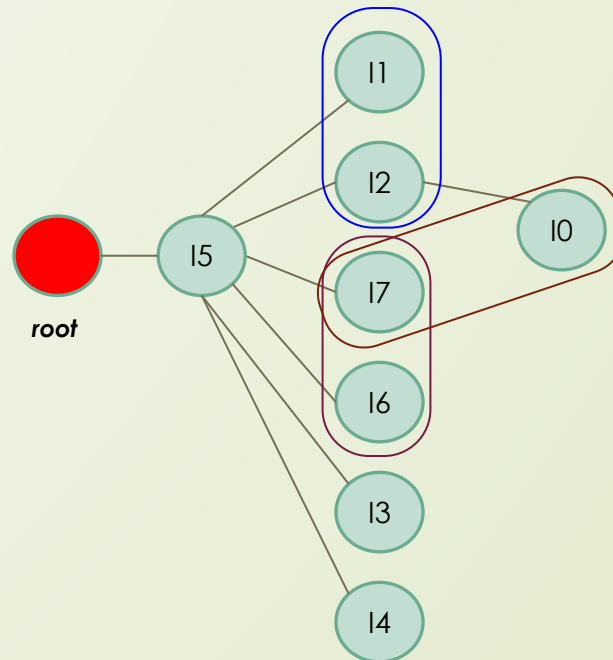
- A multi dimensional array is associated to each partition
- Keys data may be implicit for primary indexes
- Dimension can be reduced by codec extension
- Dimension can be increased by values extension
- In a root partition, all the index are secondary, the dimension is 0 (the primary index is the root index)

Example

["plants", ["fruit", "fruit", "fruit", "fruit", "vegetable", "vegetable", "vegetable", "vegetable"]]
["quantity", ["1 kg", "10 kg", "1 kg", "10 kg", "1 kg", "10 kg", "1 kg", "10 kg"]]
["product", ["apple", "apple", "orange", "orange", "peppers", "peppers", "banana", "banana"]]
["price", [1, 10, 2, 20, 1.5, 15, 1, 1.5]]
["group", ["fruit 1", "fruit 10", "fruit 1", "veget", "veget", "veget", "veget", "veget"]]
["id", [1001, 1002, 1003, 1004, 1005, 1006, 1007, 1008]]
["supplier", ["sup1", "sup1", "sup1", "sup2", "sup2", "sup2", "sup2", "sup1"]]
["location", ["fr", "gb", "es", "ch", "gb", "fr", "es", "ch"]]

Derived tree :

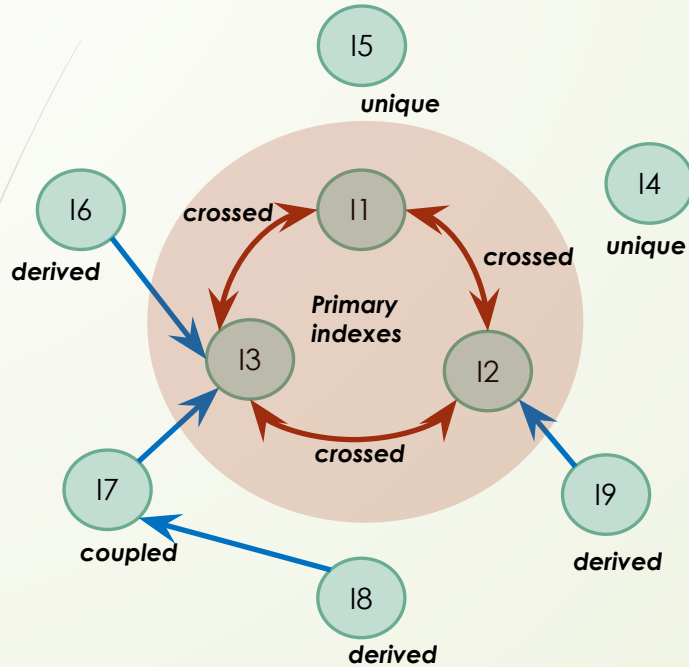
-1: root (8)
 5 : id (8)
 1 : quantity (2)
 2 : product (4)
 0 : plants (2)
 3 : price (6)
 4 : group (3)
 6 : supplier (2)
 7 : location (4)



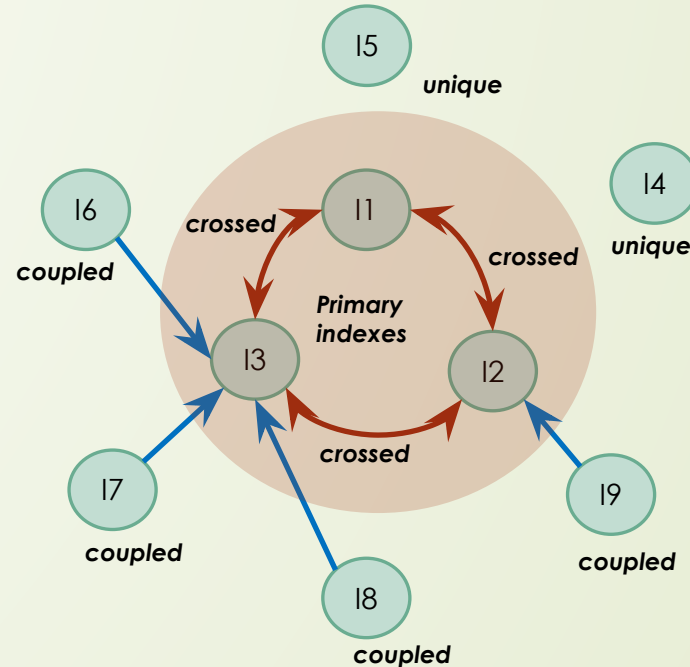
Partition :

[(0, 7), (6, 7), (1, 2)]

2 – Primary Structure



Canonical structure
(default codec)



Complete indexed structure
(adjusted codec and values)

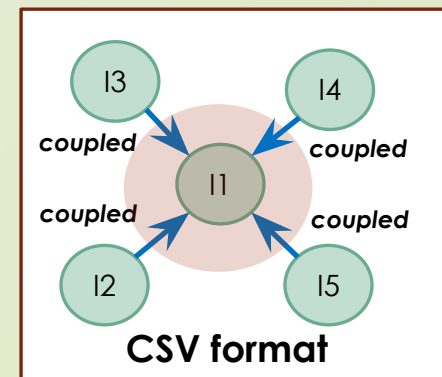
In a canonical format, Keys are:

- Implicit for Primary, Unique and Coupled indexes
- Relative for Derived indexes

In a complete indexed format, Keys are all implicit

- **Properties**

- Each Ilist has a **canonical structure** (at least one primary index)
- **Complete data** is obtained by crossing all the primary indexes (values extension)
- Complete Ilist can be transformed in **Matrix** (coupled secondary indexes)
- **CSV format** is a canonical structure with one primary index and any coupled indexes, all indexes have full codec



2 - Example

3 columns are linked

- Full name
- Course
- Examen

3 columns are derived

- First name
- Last name
- Group

1 column is coupled

- Surname

1 column is unique

- Year

ratio

- Name – Course : 37,5 %
- Name – Examen : 62,5 %
- Course – Examen : 83,7 %

l1ist

first name	last name	full name	surname	group	course	year	examen	score
Anne	White	Anne White	skyler	gr1	math	2021	t1	11
Anne	White	Anne White	skyler	gr1	math	2021	t2	13
Anne	White	Anne White	skyler	gr1	math	2021	t3	15
Anne	White	Anne White	skyler	gr1	english	2021	t2	10
Anne	White	Anne White	skyler	gr1	english	2021	t3	12
Philippe	White	Philippe White	heisenberg	gr2	math	2021	t1	15
Philippe	White	Philippe White	heisenberg	gr2	english	2021	t2	8
Camille	Red	Camille Red	saul	gr3	software	2021	t3	17
Camille	Red	Camille Red	saul	gr3	software	2021	t2	18
Camille	Red	Camille Red	saul	gr3	english	2021	t1	2
Camille	Red	Camille Red	saul	gr3	english	2021	t2	4
Philippe	Black	Philippe Black	gus	gr3	software	2021	t3	18
Philippe	Black	Philippe Black	gus	gr3	english	2021	t1	6

37% almost derived or coupled

83% almost crossed

coupled

derived

unique

2 – Structuration process

- **Objectives**

- Data understanding
- Inconsistent data identification
- Size reduction
- Transfer to analysis tools (e.g. Pandas, Xarray)

- **Analysis**

- **Index characterization**

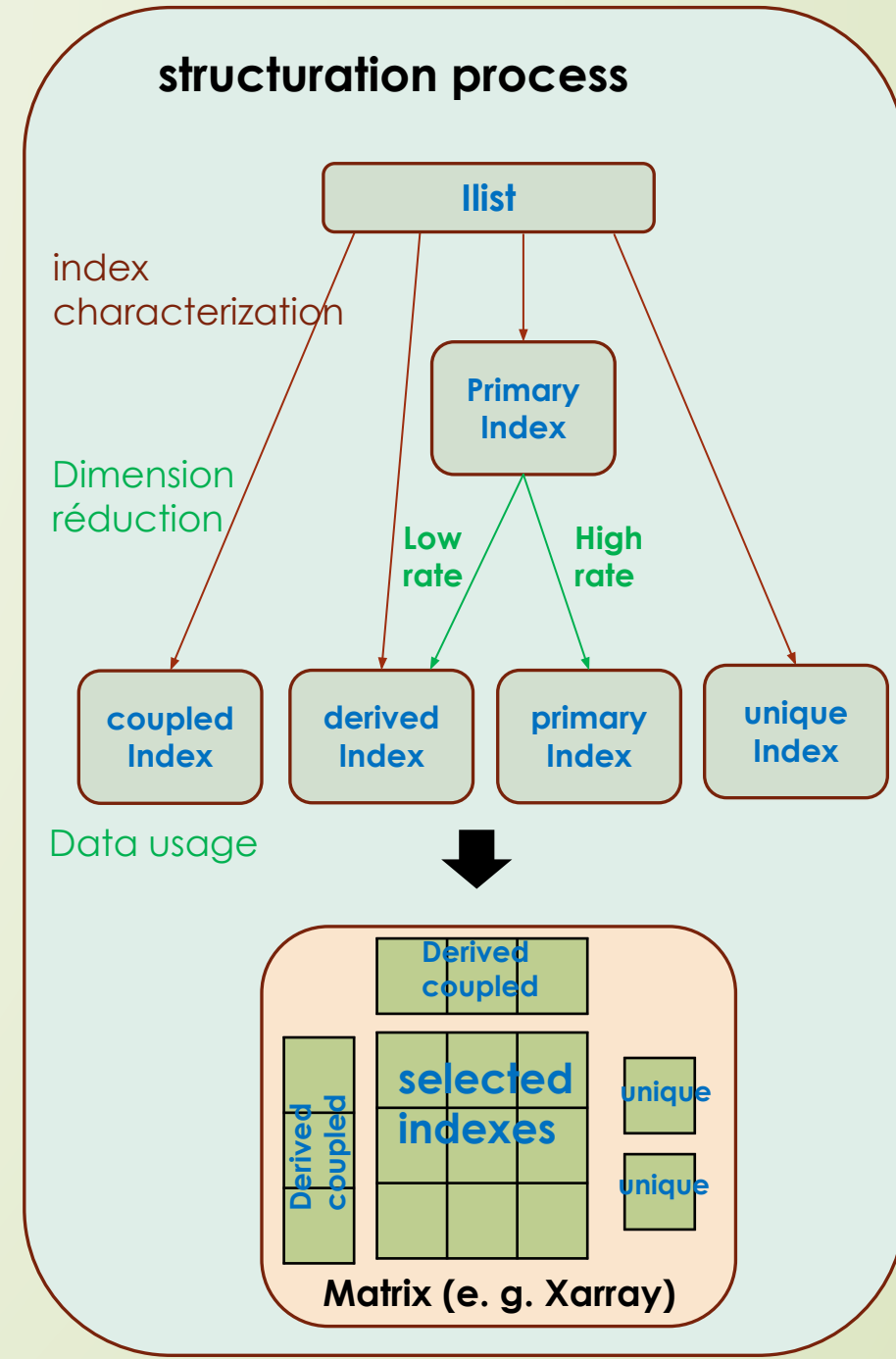
- Identification of primary indexes
 - Association of secondary indexes to primary indexes

- **Linked indexes analysis**

- Low rate (i.e. $< 0,1$) = almost derived index
 - > transform to derived index (codec extension)
 - > or values correction
 - High rate (i.e. $> 0,9$) = almost crossed index
 - > transform to crossed index
 - > or values correction

- **Data usage**

- **Dimension reduction (if necessary)**
 - Primary index merging (rather low rate)
 - **Export**
 - Matrix generation
 - Storage



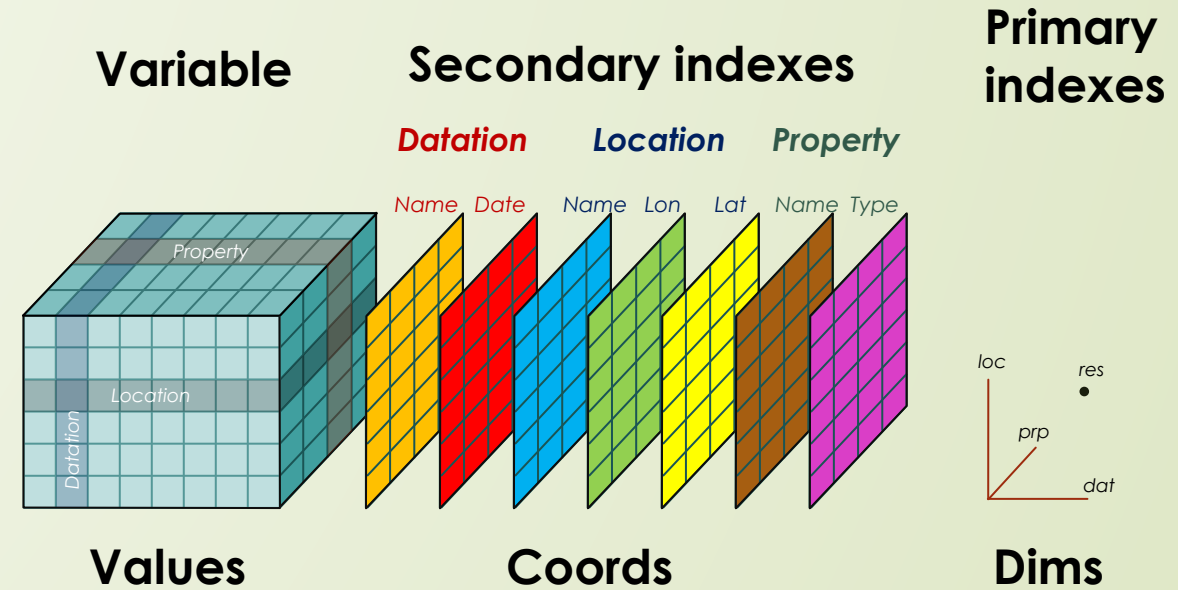
2 - Example : Xarray – mapping

> Xarray

- Values : data matrix(ex. numpy ndarray)
- Coords : list of indexes: (dims, data, attrs)
- Dims : names of dimensions
- Attrs : attribute dictionary (data or coord)
- Name

> Ilist Mapping

- Dims : Primary indexes
- Values : Variable values
- Coords : Secondary indexes
- Attrs : Unique indexes
- Name : Ilist name



2 - Example

to_xarray function :

- Primary crossed (values extension)
- Secondary coupled (full codec)

completed

first name	last name	full name	surname	group	course	year	examen	score
Anne	White	Anne White	skyler	gr1	english	2021	t1	-
Anne	White	Anne White	skyler	gr1	english	2021	t2	10
Anne	White	Anne White	skyler	gr1	english	2021	t3	12
Anne	White	Anne White	skyler	gr1	math	2021	t1	11
Anne	White	Anne White	skyler	gr1	math	2021	t2	13
Anne	White	Anne White	skyler	gr1	math	2021	t3	15
Anne	White	Anne White	skyler	gr1	software	2021	t1	-
Anne	White	Anne White	skyler	gr1	software	2021	t2	-
Anne	White	Anne White	skyler	gr1	software	2021	t3	-

derived

coupled

↑ unique

```
In [367]: cours.to_xarray(axes=cours.axesmin)
Out[367]:
<xarray.DataArray 'Ilist' (full name: 4, course: 3, examen: 3)>
array([[[ '?', '10', '12'],
        ['11', '13', '15'],
        ['?', '?', '?']],

       [['2', '4', '?'],
        ['?', '?', '?'],
        ['?', '18', '17']],

       [['6', '?', '?'],
        ['?', '?', '?'],
        ['?', '?', '18']],

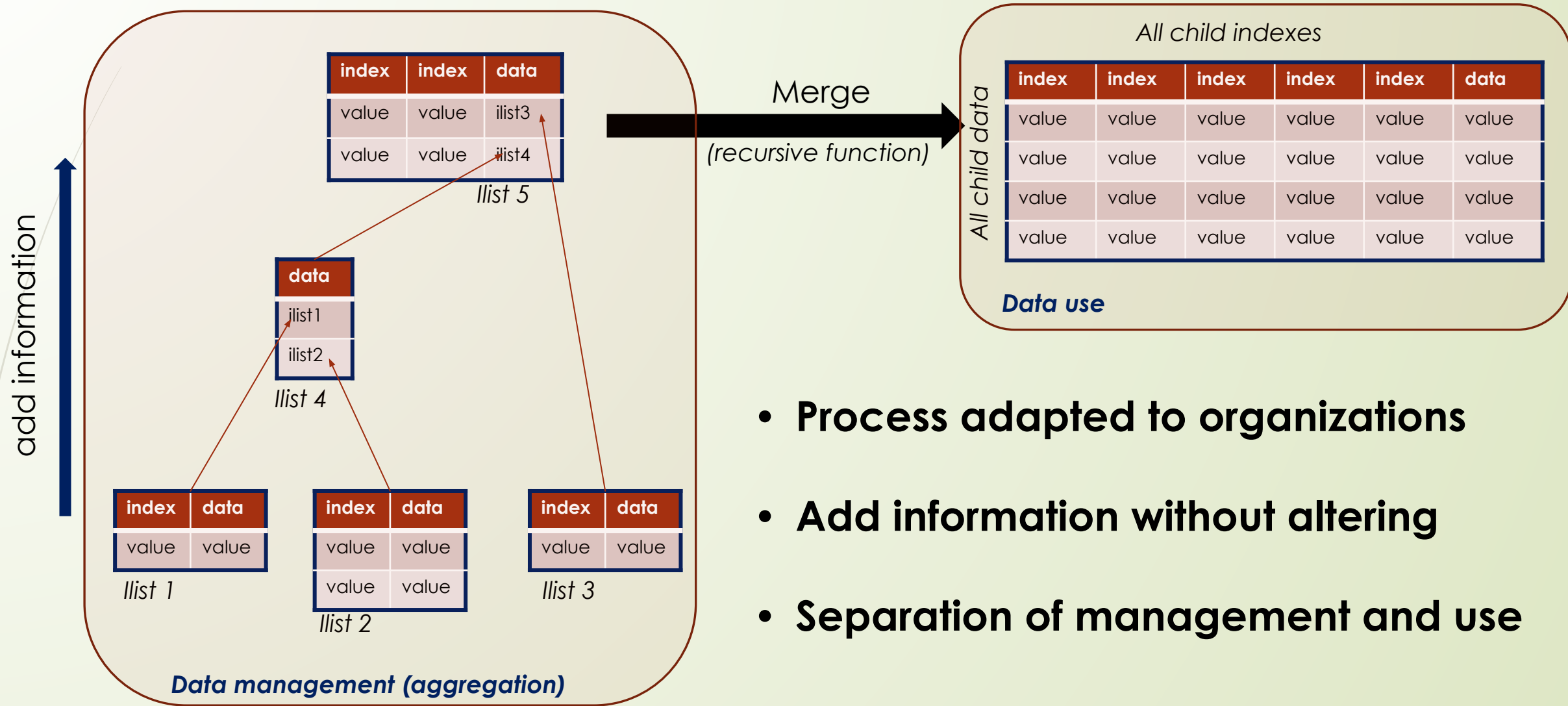
       [['?', '8', '?'],
        ['15', '?', '?'],
        ['?', '?', '?']], dtype='<U2')

```

```
Coordinates:
  i>first name  (full name) <U8 'Anne' 'Camille' 'Philippe' 'Philippe'
  last name    (full name) <U5 'White' 'Red' 'Black' 'White'
  * full name   (full name) <U14 'Anne White' ... 'Philippe White'
  surname      (full name) <U10 'gus' 'heisenberg' 'saul' 'skyler'
  group        (full name) <U3 'gr1' 'gr3' 'gr3' 'gr2'
  * course      (course) <U8 'english' 'math' 'software'
  * examen      (examen) <U2 't1' 't2' 't3'

```

3 - Building process



3 - Example

aw

IndexSet			Data
course	year	examen	score
math	2021	t1	11
math	2021	t2	13
math	2021	t3	15
english	2021	t2	10
english	2021	t3	12

pw

course	year	examen	score
math	2021	t1	15
english	2021	t2	8

cr

course	year	examen	score
software	2021	t3	17
software	2021	t2	18
english	2021	t1	2
english	2021	t2	4

pb

course	year	examen	score
software	2021	t3	18
english	2021	t1	6



total

first name	last name	full name	surname	group	file
Anne	White	Anne White	skyler	gr1	aw
Philippe	White	Philippe White	heisenberg	gr2	pw
Camille	Red	Camille Red	saul	gr3	cr
Philippe	Black	Philippe Black	gus	gr3	pb



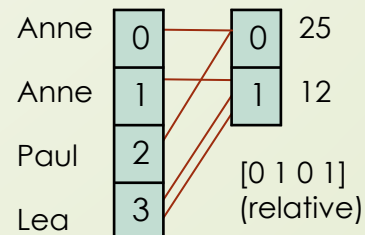
total.merge()

first name	last name	full name	surname	group	course	year	examen	score
Anne	White	Anne White	skyler	gr1	math	2021	t1	11
Anne	White	Anne White	skyler	gr1	math	2021	t2	13
Anne	White	Anne White	skyler	gr1	math	2021	t3	15
Anne	White	Anne White	skyler	gr1	english	2021	t2	10
Anne	White	Anne White	skyler	gr1	english	2021	t3	12
Philippe	White	Philippe White	heisenberg	gr2	math	2021	t1	15
Philippe	White	Philippe White	heisenberg	gr2	english	2021	t2	8
Camille	Red	Camille Red	saul	gr3	software	2021	t3	17
Camille	Red	Camille Red	saul	gr3	software	2021	t2	18
Camille	Red	Camille Red	saul	gr3	english	2021	t1	2
Camille	Red	Camille Red	saul	gr3	english	2021	t2	4
Philippe	Black	Philippe Black	gus	gr3	software	2021	t3	18
Philippe	Black	Philippe Black	gus	gr3	english	2021	t1	6

4 – index Representation

- **Codec representation**
 - List of values (or dict key/value)
- **Keys representation**
 - Absolute : List of integer (index of codec value)
 - Relative : List of integer (index of other keys)
 - Implicit : Automatic list (i.e. with full codec)
- **index /variable Formats**
 - Simple format (full codec)
 - Simple format (default codec)
 - Complete format (codec + keys)
 - Coupled format (codec + parent)
 - Derived periodic format (codec + parent)
 - Derived format (codec + parent + keys)
 - Keys = index of parent keys

[Anne, Anne, Lea, Paul, Lea]
[12, 25, 12, 25, 12]



Json Example

['Anne', 'Anne', 'John', 'Paul']

[2, 2, 3] → ['John', 'John', 'Paul']

[2, 3] → ['John', 'John', 'Paul']

[0, 1, 2, 3] ← if full codec

['Anne', 'Anne', 'John', 'Paul', 'John']

['Anne', 'John', 'Paul']

[['Anne', 'John', 'Paul'], [0, 0, 1, 2, 1]]

[['Anne', 'John', 'Paul'], parent]

[['Anne', 'John', 'Paul'], parent, [0, 0, 1, 2, 1]]

Derived index : [[25, 12], parent, [0, 1, 0, 1]]

(derived)

Parent index : [Anne, Anne, Paul, Lea], [0, 1, 3, 2, 3]]

(complete)

4 – Ilist representation

Mode 'full'

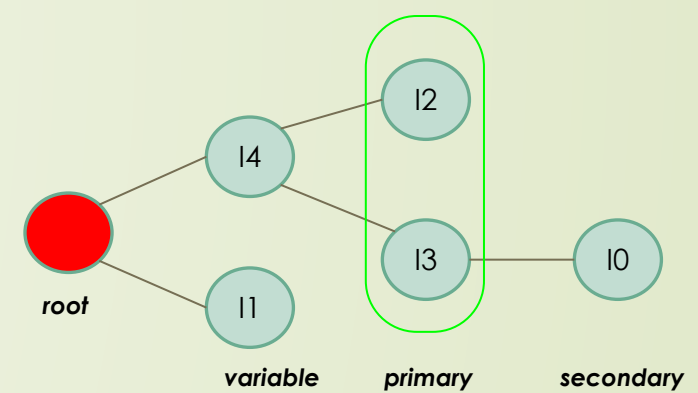
- **All index (simple format)**
 - Codec : full
 - Parent : implicit
 - Keys : implicit

Mode 'default'

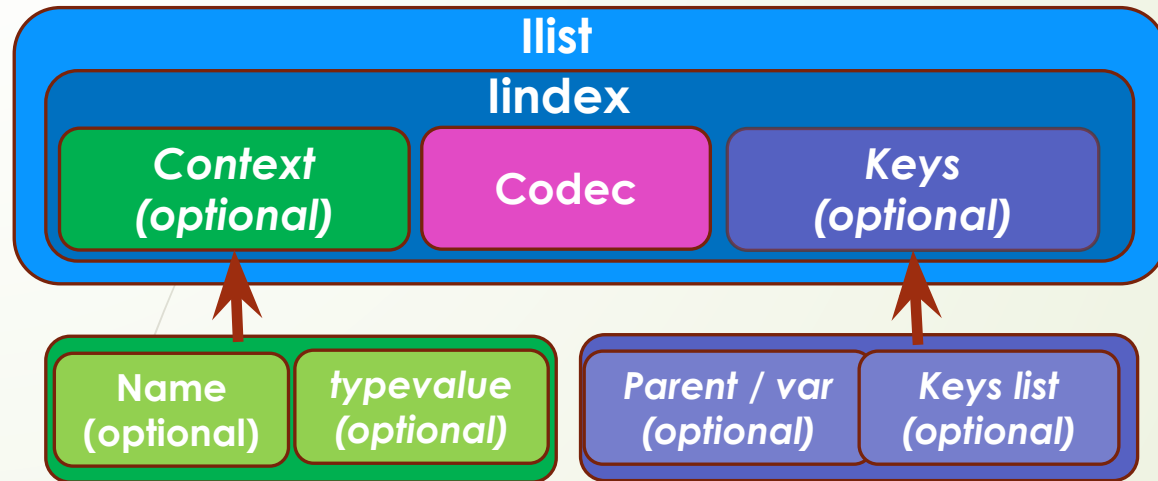
- **All index (complete format)**
 - Codec : default
 - Parent : implicit
 - Keys : absolute

Mode 'optimize'

- **Primary index (simple format)**
 - Codec : default
 - Parent : implicit
 - Keys : implicit
- **Variable and secondary index**
 - Codec : default
 - Parent : explicit (implicit if root)
 - Keys :
 - Unique : implicit
 - Coupled : implicit
 - Derived : relative
 - Root derived : absolute



4 – JSON Representation



Ilist: JSON Array
index: JSON Array (or JSON value if only one value and not a value Array)

Context: JSON Object (or JSON string if only one value)
Codec: JSON Array
Keys: JSON Array

Name, Typevalue: string
keys list: JSON Array
Parent / variable: integer (-1 for variable)

Format

Text (JSON text), Binary (CBOR)

Example Index : Name : 'team1'

Values : ['Anne', 'Anne', 'John', 'Paul', 'John']

- **Simple format (without name)**
['Anne', 'Anne', 'John', 'Paul', 'John']
-> Full codec (e.g. csv format)
- **Simple format (with name)**
['team1', ['Anne', 'John', 'Paul']]
-> Default codec (e.g. crossed index)
- **Complete format (with name)**
['team1', ['Anne', 'John', 'Paul'], [0,0,1,2,1]]
-> Default codec, name, absolute keys
- **Coupled format (with name)**
['team1', ['Anne', 'John', 'Paul', 'John'], 2]
-> Adjusted codec, parent id
- **Derived format (with name)**
['team1', ['Anne', 'John', 'Paul'], [2, [0,1,2,1]]]
-> Default codec, parent id, relative keys
- **Unique format**
['team1', ['Anne']] (with name) **'Anne'** (without name)
-> Default codec (= full codec)

4 – list size and indicators

- **Maximal size (size without coding) = $nv * sv$**
 - nv : number of values (name included)
 - sv : mean value size = maximal size / nv
- **Minimal size (size with null coding size) = $nc * sv$**
 - nc : number of different values (name included)
- **Indexed size = $(nv - nc) * sc + nc * sv$**
 - sc : mean coding value size = $(size - nc * sv) / (nv - nc)$
- **Indicators**
 - **$UL = nc / nv$ (unicity level) [0, 1] (data quality - gain maxi)**
 - UL = minimal size / maximal size
 - $1 - UL$ = gain maxi = $(\text{maximal size} - \text{minimal size}) / \text{maximal size}$
 - $UL = 0$ (empty data), $UL = 1$ (unoptimisable data)
 - **$OL = sc / sv$ (object lightness) [0, 1] (data coding - gain ratio)**
 - OL = $(\text{indexed size} - \text{minimal size}) / (\text{maximal size} - \text{minimal size})$
 - $1 - OL$ = Gain / Gain maxi
 - $OL = 0$ (maxi coding), $OL = 1$ (no coding)
 - **Gain = $(1 - UL) * (1 - OL)$**
 - Gain = $(\text{maximal size} - \text{indexed size}) / \text{maximal size}$
 - Gain maxi = $1 - UL$
 - Gain = 0 (no coding), Gain = 1 (empty data / maxi coding)

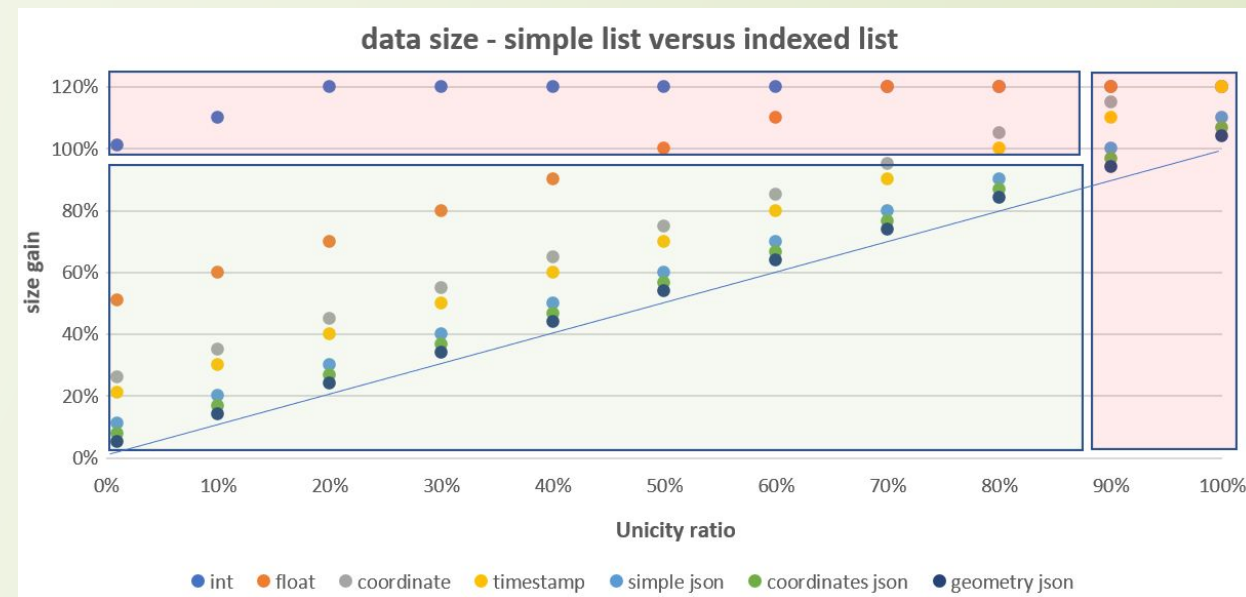
- **Example**
 - Different data:
 - $nc = nv$
 - **$UL = 1$** ($OL = 1$)
 - Identical data :
 - Lines (l), Cols (c)
 - $nv = l * c$
 - $nc = c$
 - **$UL = 1 / l$**
 - Matrix data ($n \times n$):
 - Lines (l) = $n * n$
 - Cols = 3
 - $nv = 3 * n * n$
 - $nc = 2 * n + n * n$
 - **$UL = (2+n) / 3*n$
= 0.33 (if $n \gg 2$)**

4 – list size and indicators

Example - simple list

- **Duplicate value**
 - [val, val, ..., val, val]
 - list size : n, val size : m
- **Coded list** (default)
 - [val, [0, 0, ..., 0, 0]
- **Indicators**
 - $sv = m, nv = n$
 - $sc = 2n/(n-1), nc = 1$
 - $UL = 1/n, OL = 2n / m*(n-1)$
 - $Gain = (nm-m-2n) / m*n$
 - $n \gg 1$:
 - **$UL = 1/n, OL = 2/m, Gain = 1 - 2/m$**

Object lightness	m	OL
int	2	1,00
float, int32	4	0,50
coordinate	8	0,25
string(10) (eg. timestamp)	10	0,20
simple json element (eg key/value)	20	0,10
structured json element (eg coordinates)	30	0,07
complex json element (eg geometry)	50	0,04



Example : Open-data - french charging point (EVSE)

7.5 Mo – 11 000 rows – 49 columns

Analysis :

Indexes : 1 coupled, 6 derived, 1 crossed, 41 linked

Canonical format : 1 crossed, 48 derived

File size :

Default : 3.7 Mo

Optimized : 2.5 Mo

CBOR optimized : **1.7 Mo (gain : 77% !)**