

# **Les données (Module 1)**

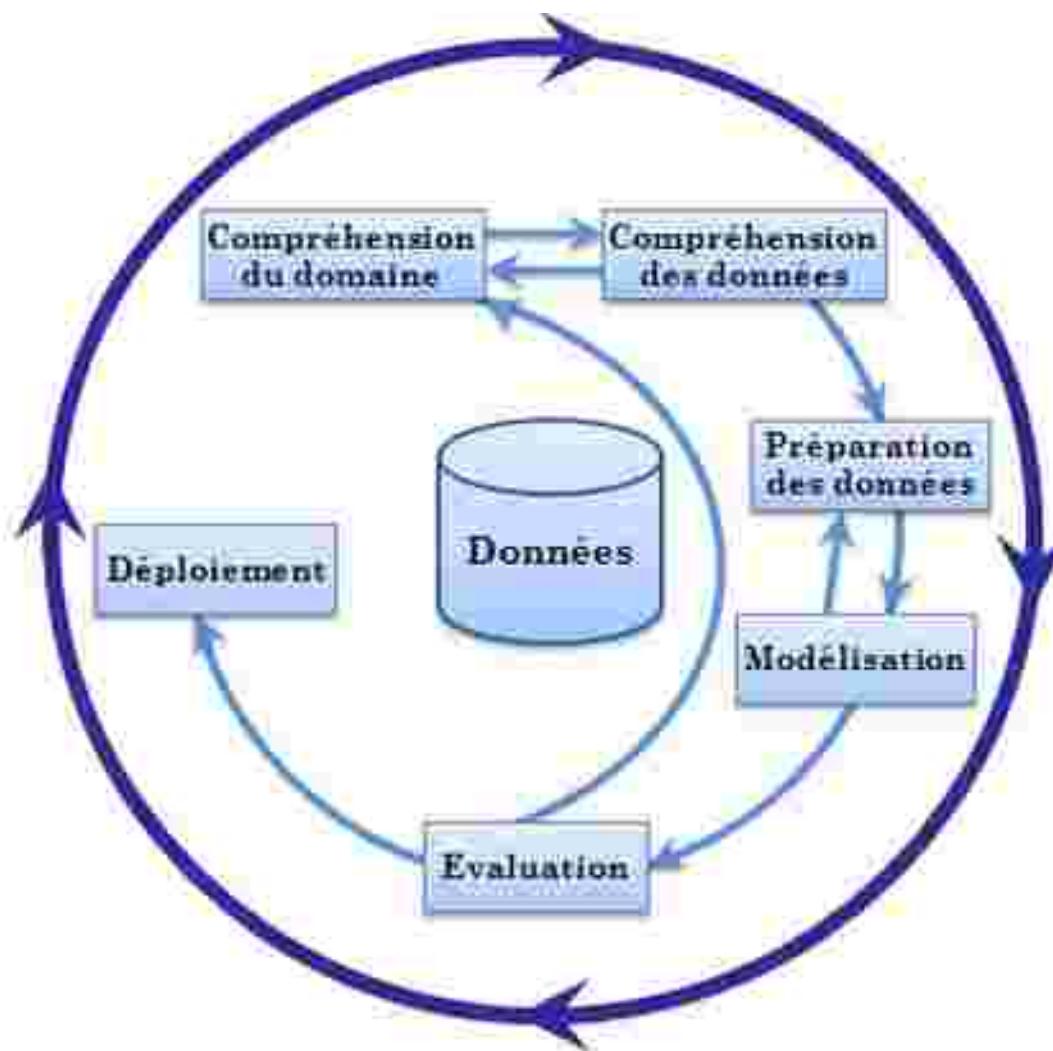
Science des données

IFT3700/IFT6758

Automne 2018

©Alain Tapp

# Modèle CRISP-DM



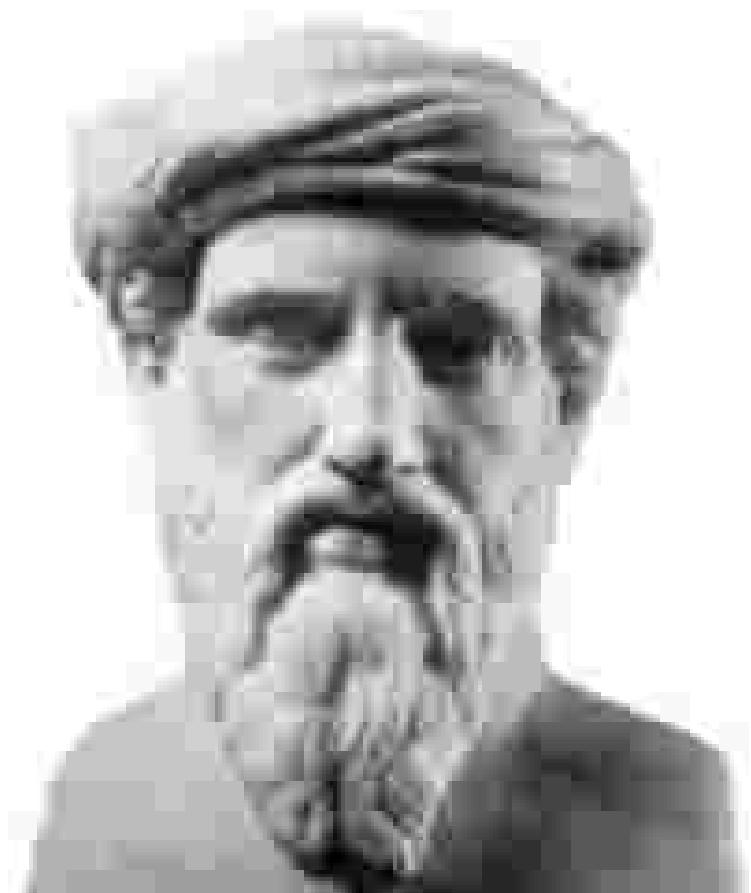
# Type de données

- Multimedia
  - Audio
  - Photo
  - vidéo
- Complexe
  - graphes, réseau, scientifique, CAD, ...
- Text
  - Non structuré
    - document, courriel, article,
  - Structuré
    - CSV, JSON
- Base de données et tableaux

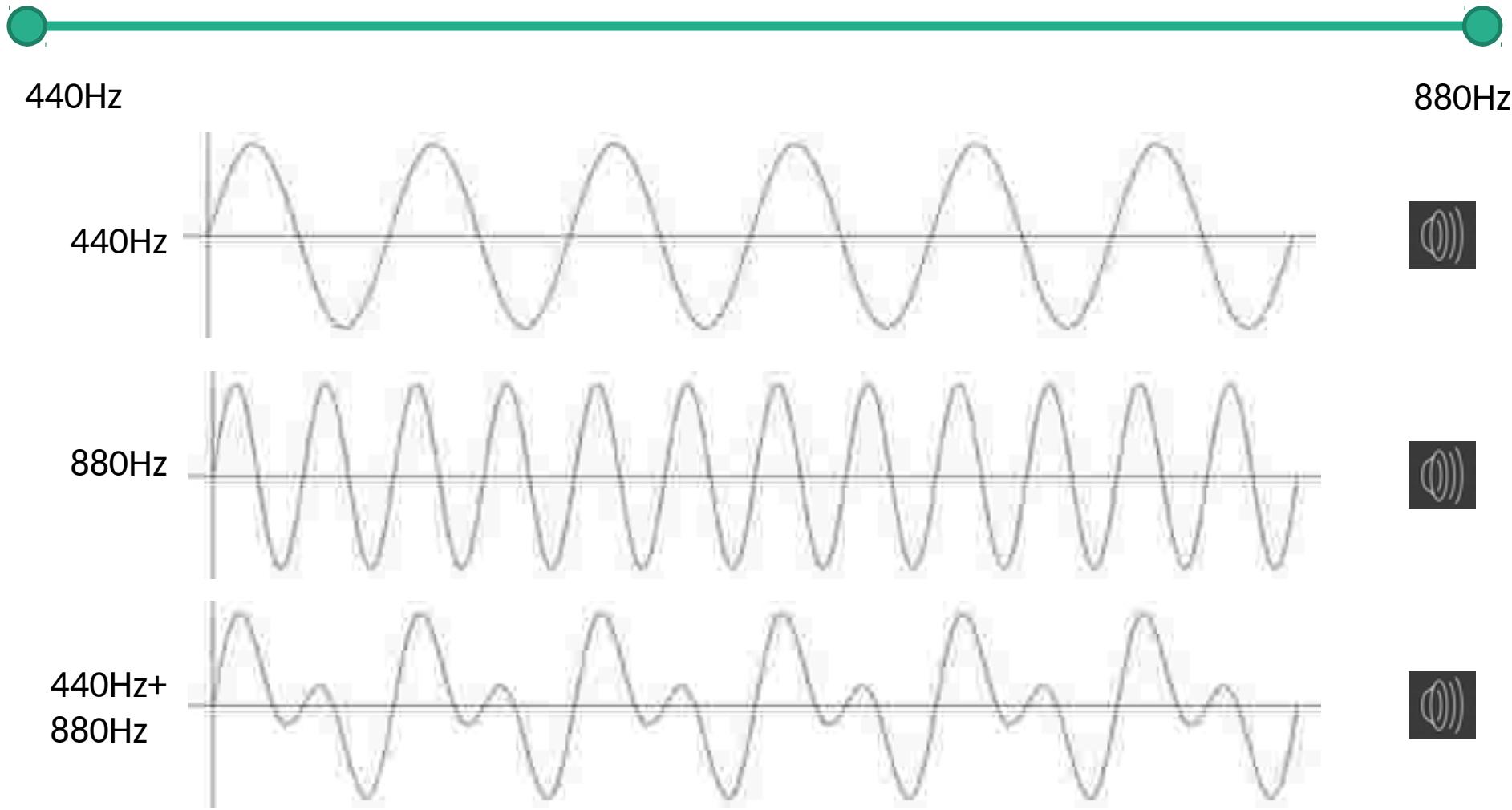
# **Audio**

# Pythagore

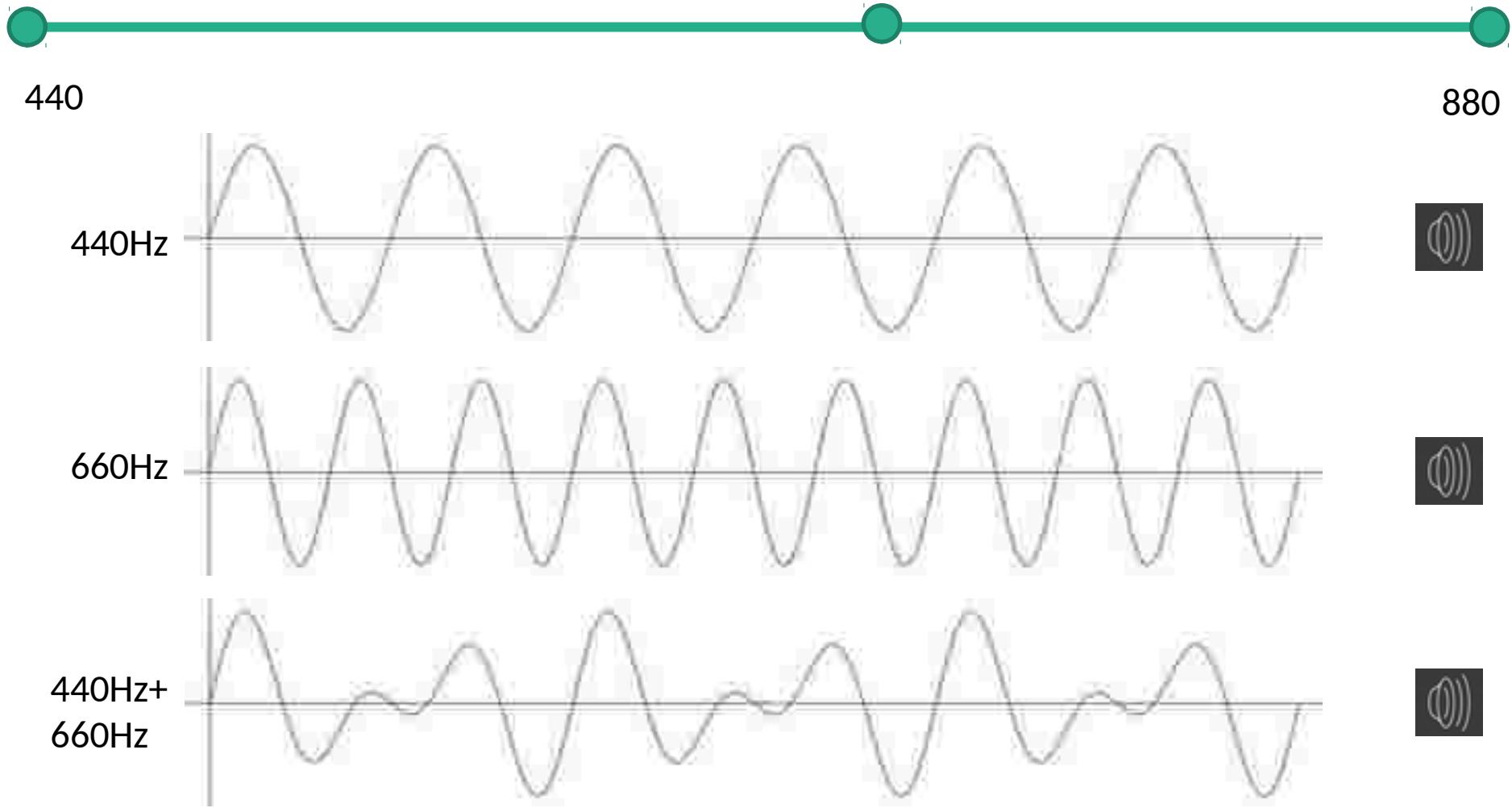
# Rationnel... ratio... fraction



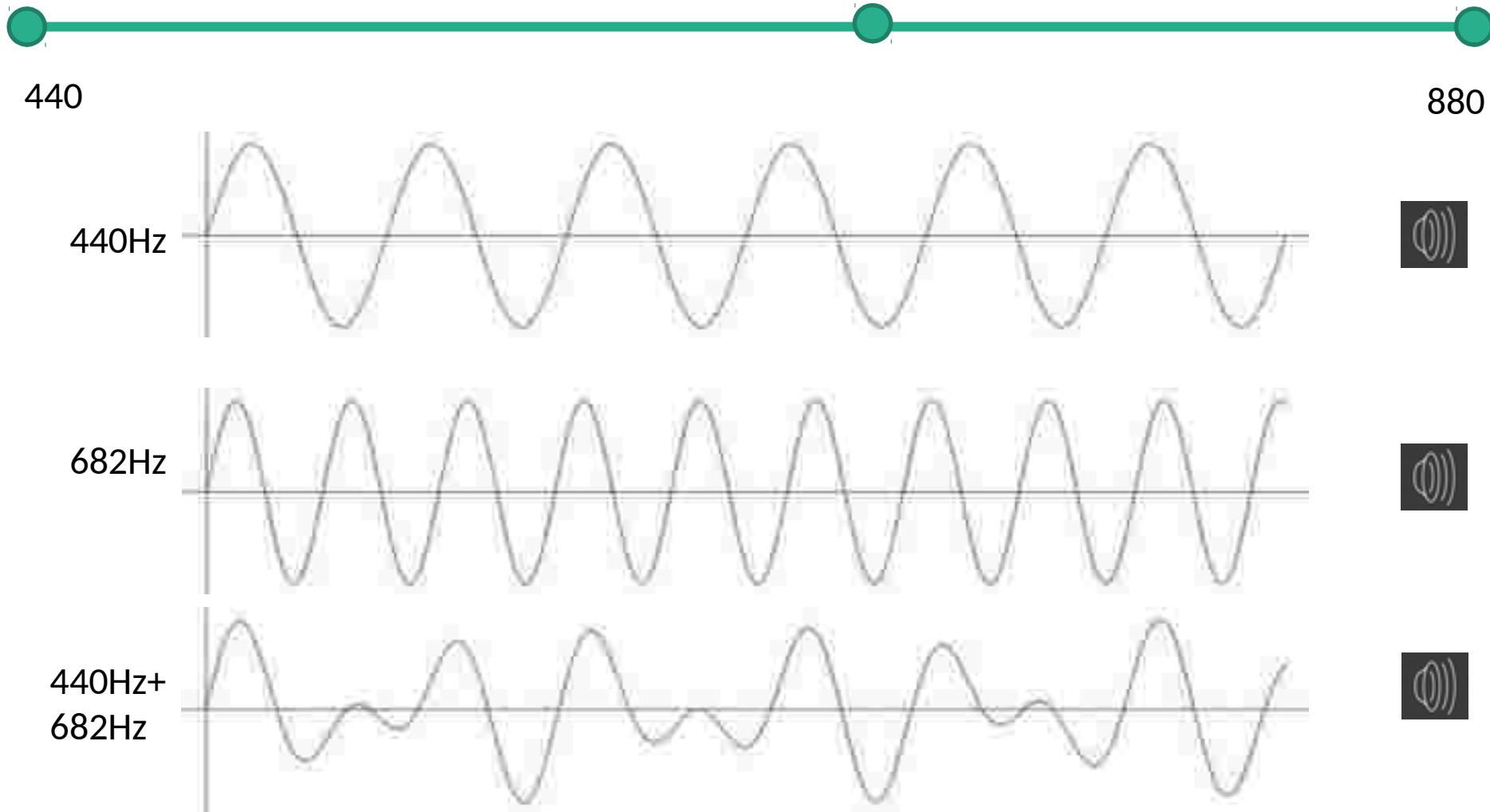
# 2/1



# 3/2



# 3.1/2



$$2/1 = 3/2 * 4/3$$



$$3/2 * 3/2 = 9/4 \dots 9/8$$



$$4/3 * 4/3 = 16/9$$



$$(3/2)^3 = 27/8 \dots 27/16$$



$$(4/3)^3 = 64/27 \dots 32/27$$



A,B,C,D,E,F,G,A

La, Si, Do, Ré, Mi, Fa, Sol, La



440

880

A,B,C,D,E,F,G,A

La, Si, Do, Ré, Mi, Fa, Sol, La

$2^{\frac{1}{12}}$   $\frac{12}{2^{12}}$   $2^{\frac{3}{12}} \frac{2}{12} \dots$   $2^{\frac{3}{12}}$   $\dots$   $2^{\frac{12}{12}} \frac{12}{2^{12}}$



440

880

La, Sib, Si, Do, Do#, Ré, Mib, Mi, Fa, Fa#, Sol, Lab , La

**1**  $2\frac{2}{12}$   $2\frac{33}{12}$   $2\frac{5}{12}$   $2\frac{7}{12}$   $2\frac{9}{12}$   $2\frac{10}{12}$  **2**



440

880

1    9/8    32/27    4/3    3/2    27/16    16/9    2

La, Si, Do, Ré, Mi, Fa, Sol, La

$$440 * \frac{9}{8} = 495$$

$$440 * \frac{2}{12} = 493.9$$



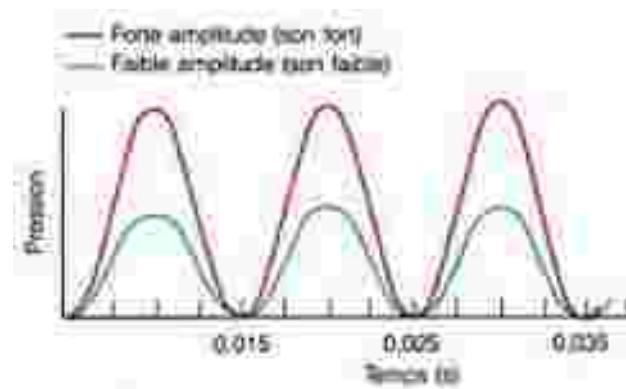
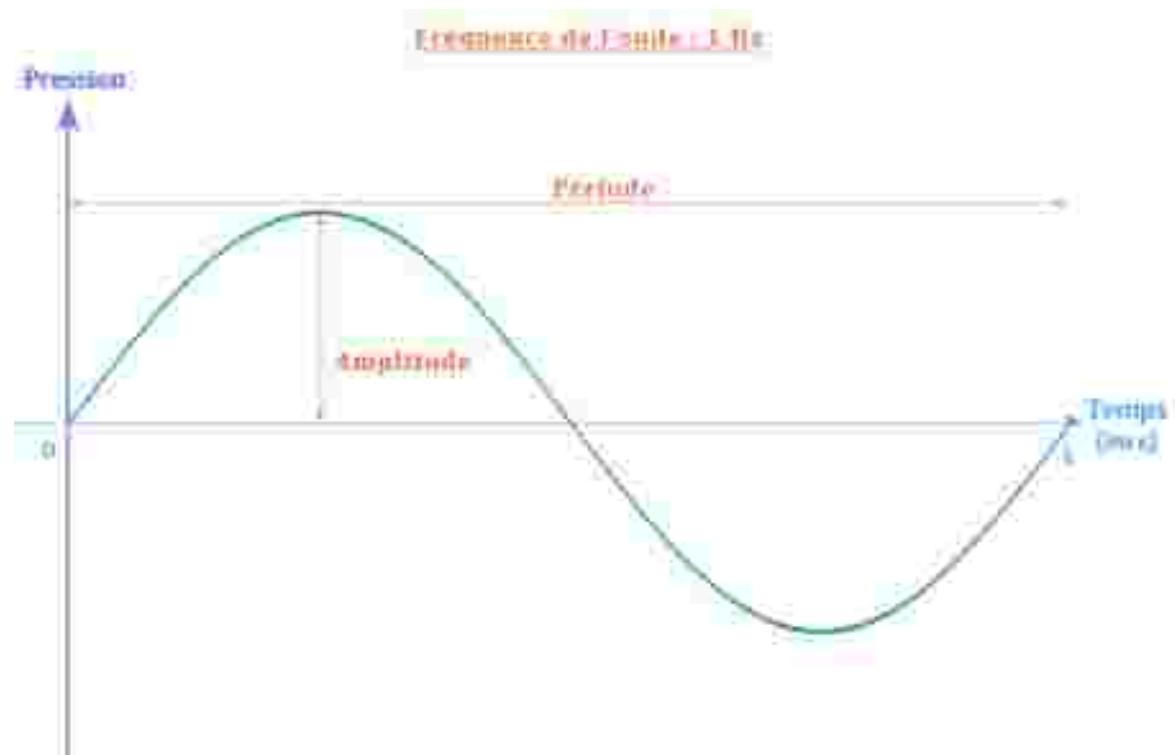
# *Das wohltemperierte Klavier*

## Le clavier bien tempéré 1722



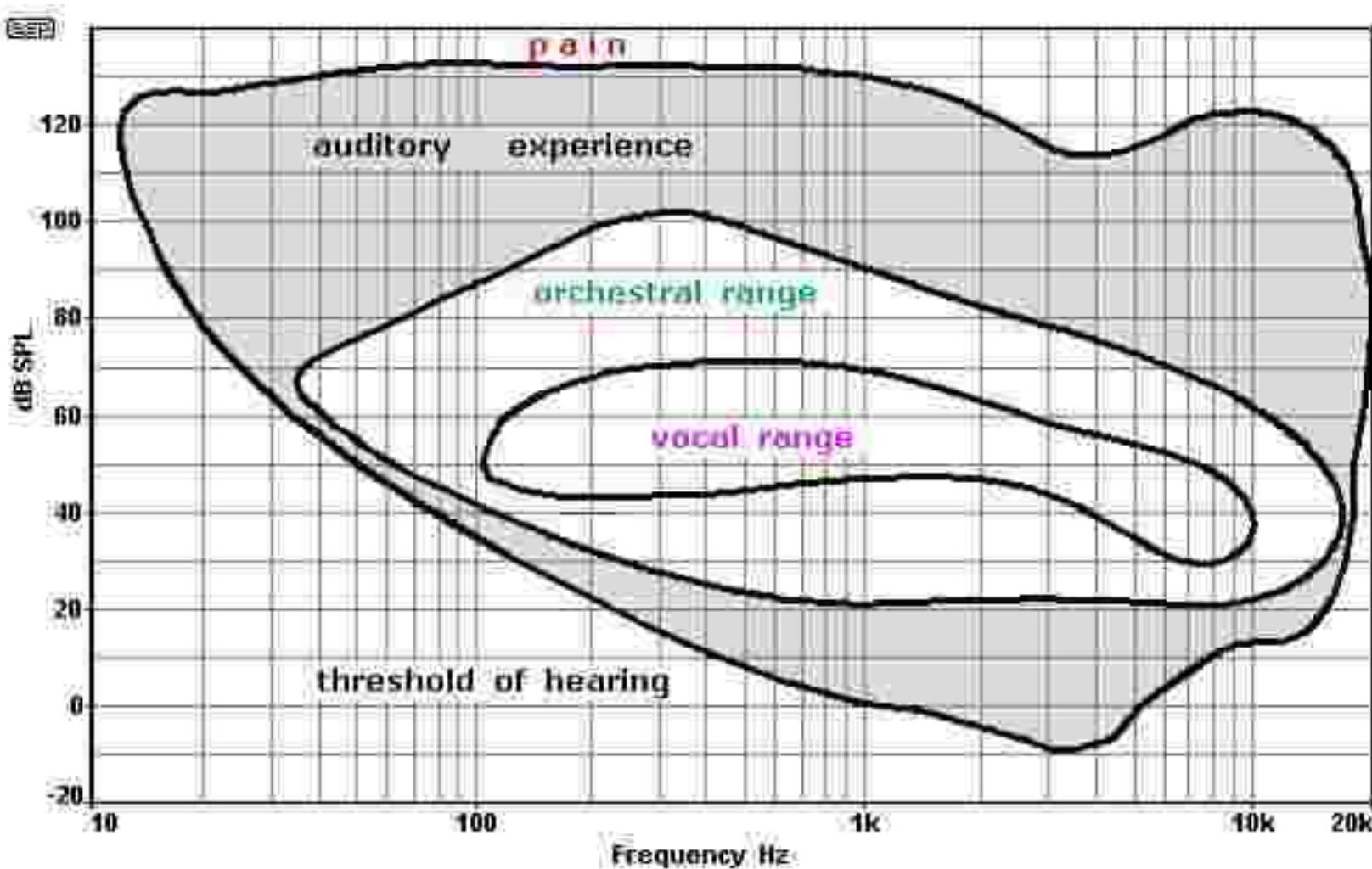
Jean-Sébastien Bach

La **fréquence** en Hz est le nombre de cycle complet par secondes.  
L'**amplitude** correspond a la force ou au volume du son.

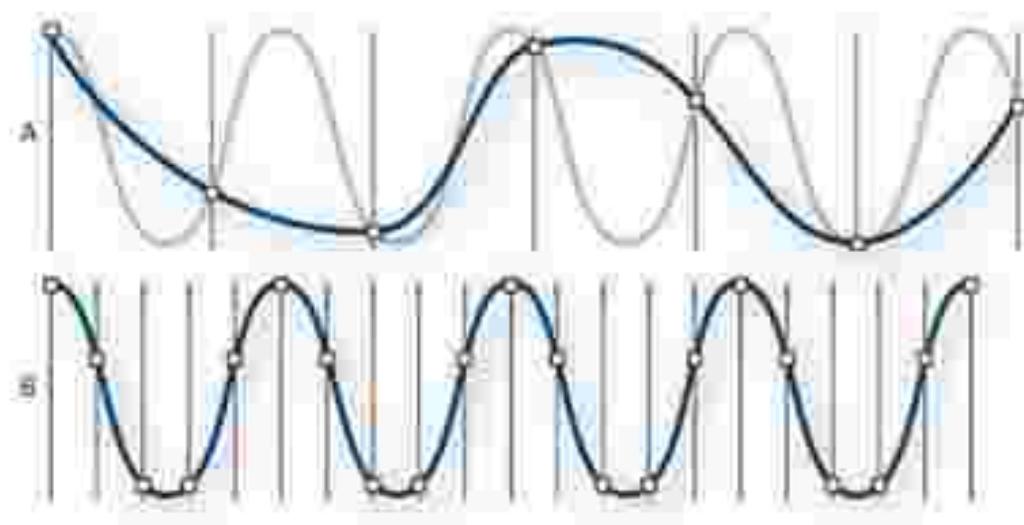
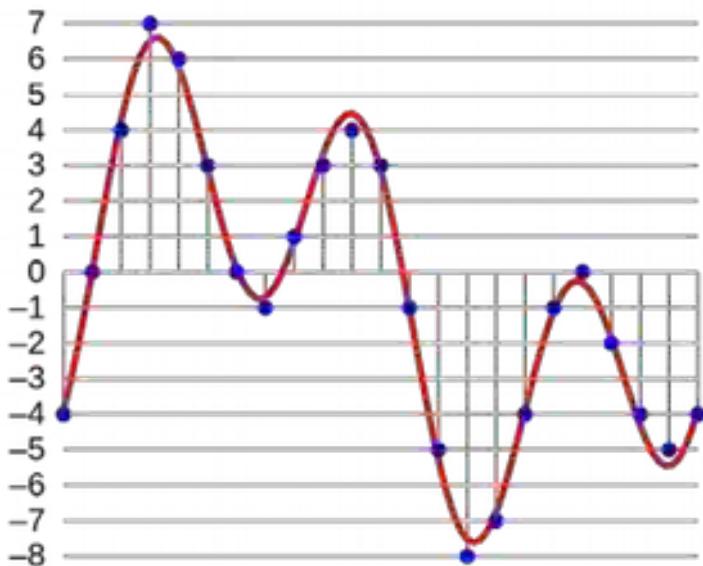


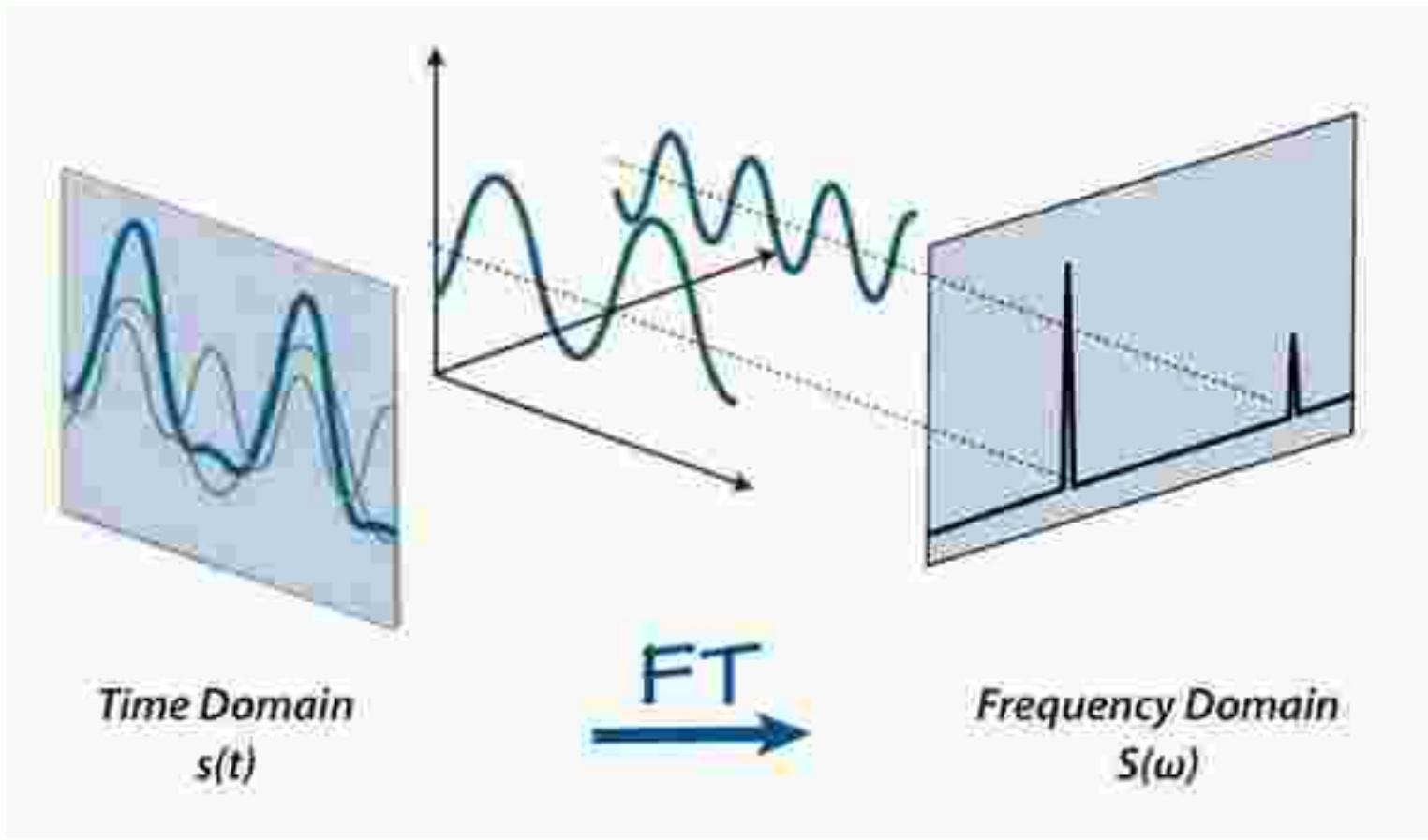
# *Typical human earing range*

Piano = 27.5 to 4186.01



Lorsqu'il est nécessaire de capturer l'audio couvrant toute la gamme de **20 à 20 000 Hz** de l'audition humaine, on utilise généralement un échantillonage à 44,1 kHz (CD), **48 kHz**. Des fréquences d'échantillonnage supérieures à environ 50 kHz à 60 kHz ne peuvent fournir plus d'informations utilisables par les auditeurs. Les premiers fabricants d'équipements audio professionnels ont choisi des taux d'échantillonnage de l'ordre de 50 kHz pour cette raison.





Dans l'espace de fréquence il est facile d'accélérer ou de ralentir en enregistrement sans modifier la texture du son.

**MP3** (*MPEG-1 Audio Layer III* ou *MPEG-2 Audio Layer III* ou *MPEG-2.5 Audio Layer III*) est très populaire.

**Opus** supporte débit constant et variable entre 6 kbit/s et 510 kbit/s, échantillonage entre 8 kHz et 48 kHz.

Une taille entre 180 KB et 15.3 MB pour un enregistrement de 4 minutes.



Original, MP3 16 kbit/s (LAME), Opus 16 kbit/s.

**Opus 16 kbit/s:**

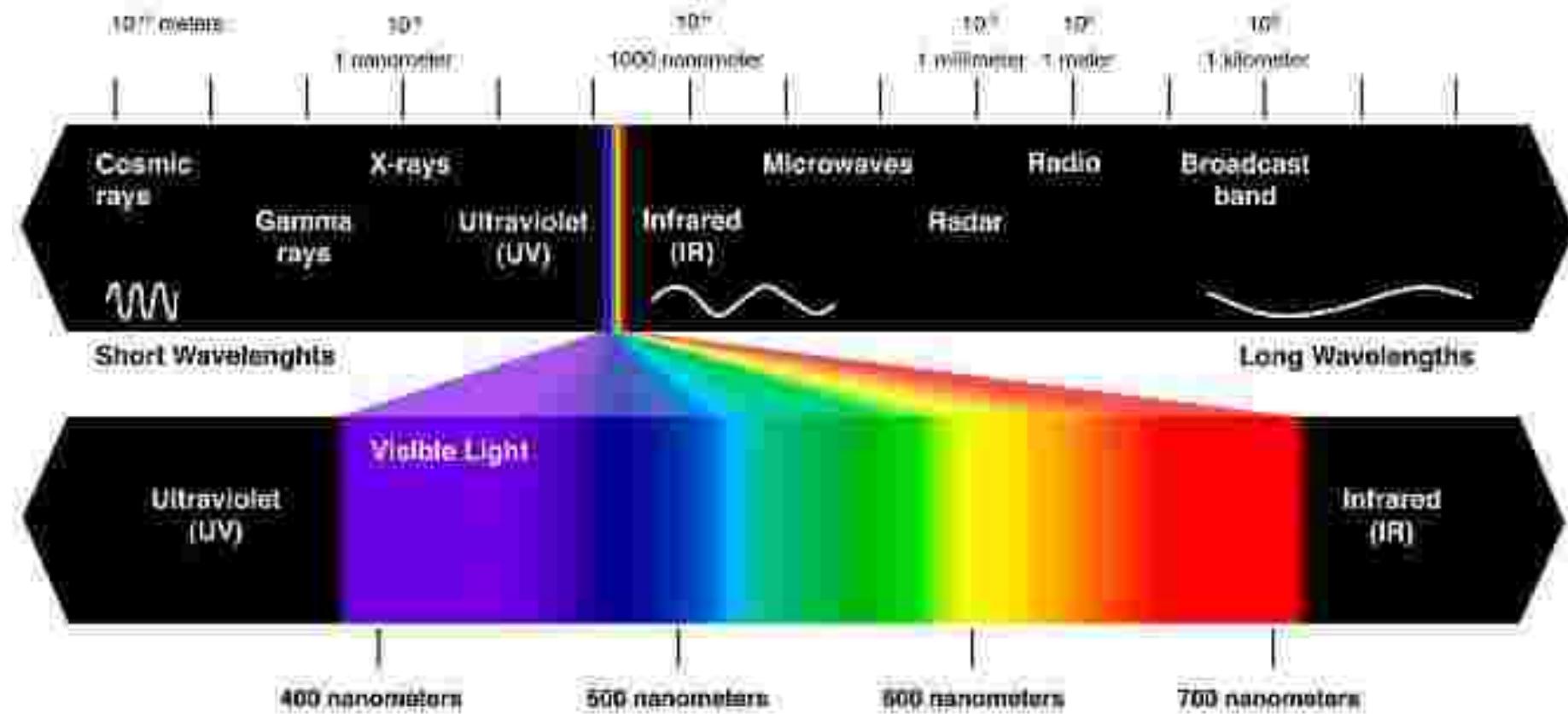
- 1 seconde ≈ 2 KB
- Chanson ≈ 0.5 MB
- Film ≈ 10 MB
- Année d'activité humaine ≈ 20 GB

Google Play Music et Spotify ont à leur disposition plus de 30M de chansons à diffuser.

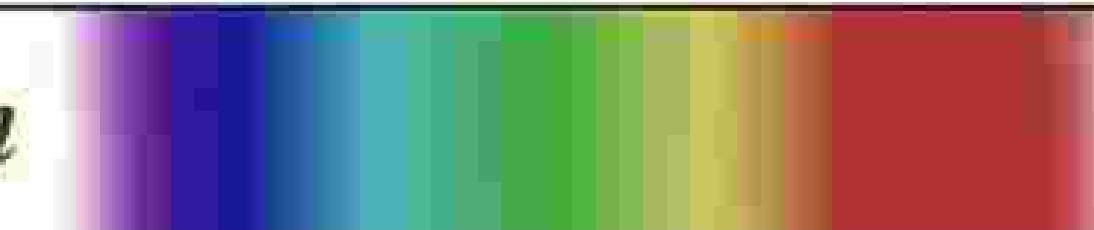
256kbps x 240 seconds x plus de 30 M ≈ proche de 2 PB.

Mais, 1G personnes avec 600 pièces musicales nécessite 40 EB de stockage.

# **Image et vidéo**



*Human*



400

500

600

700

*Bee*

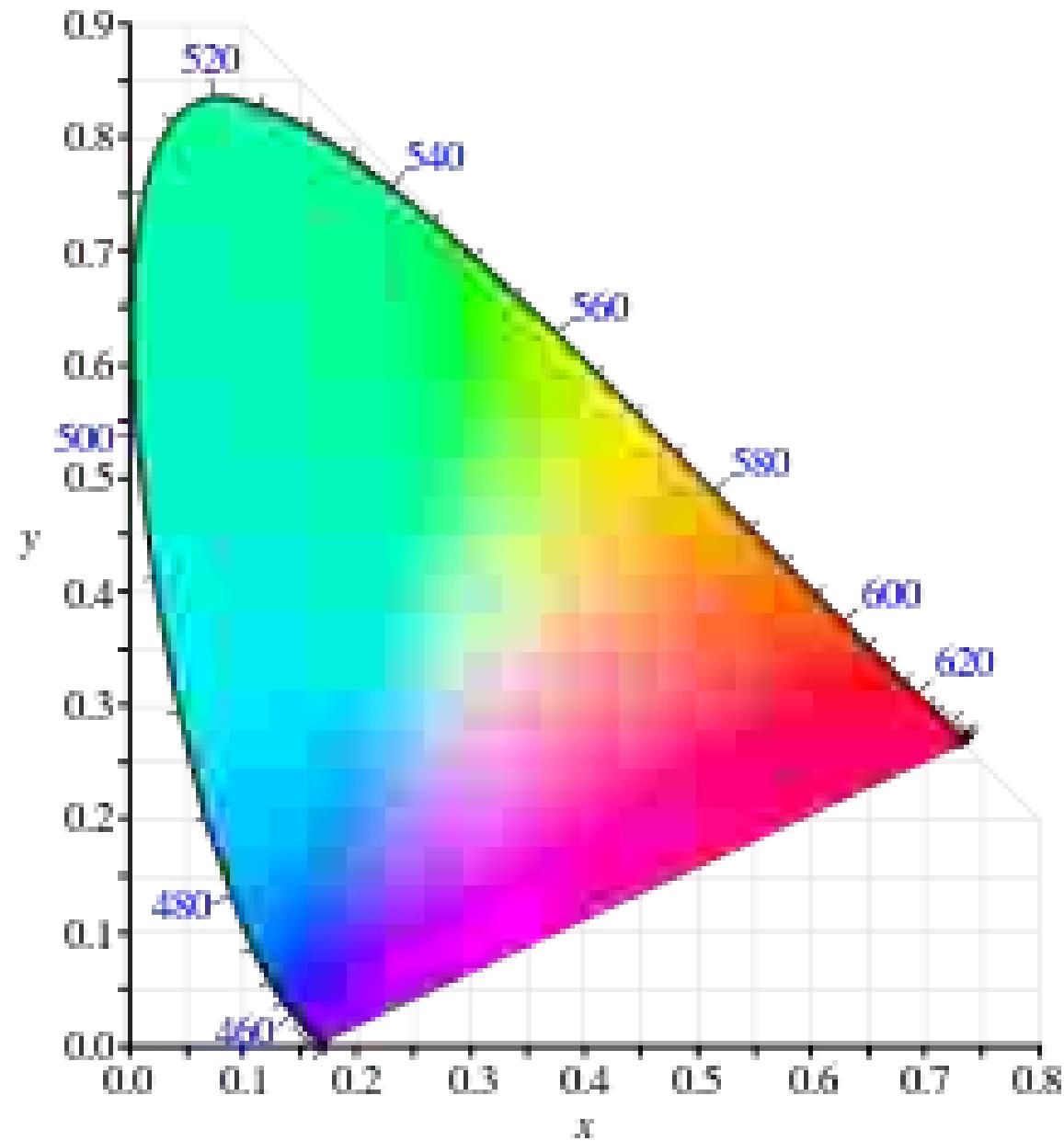
400

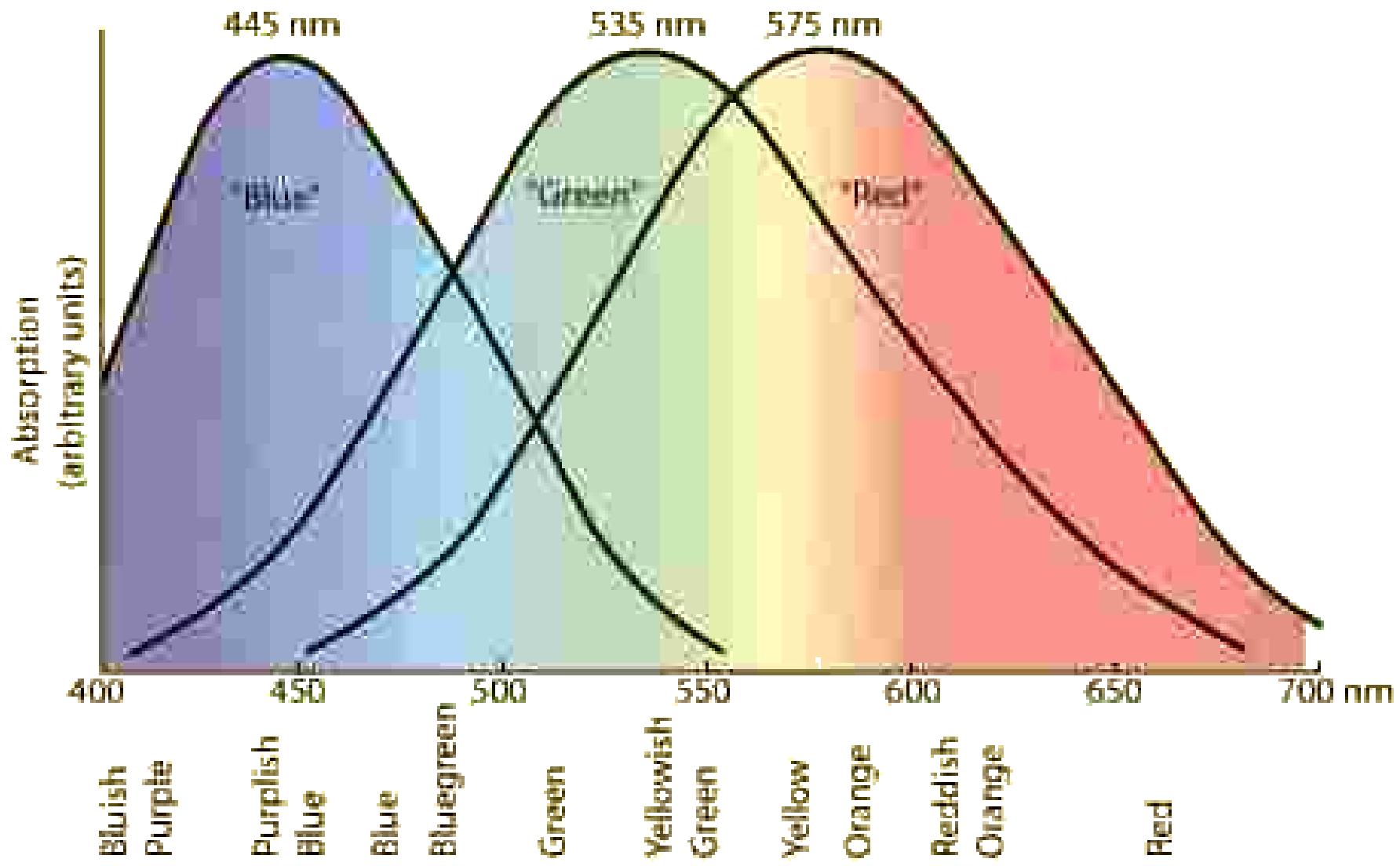
500

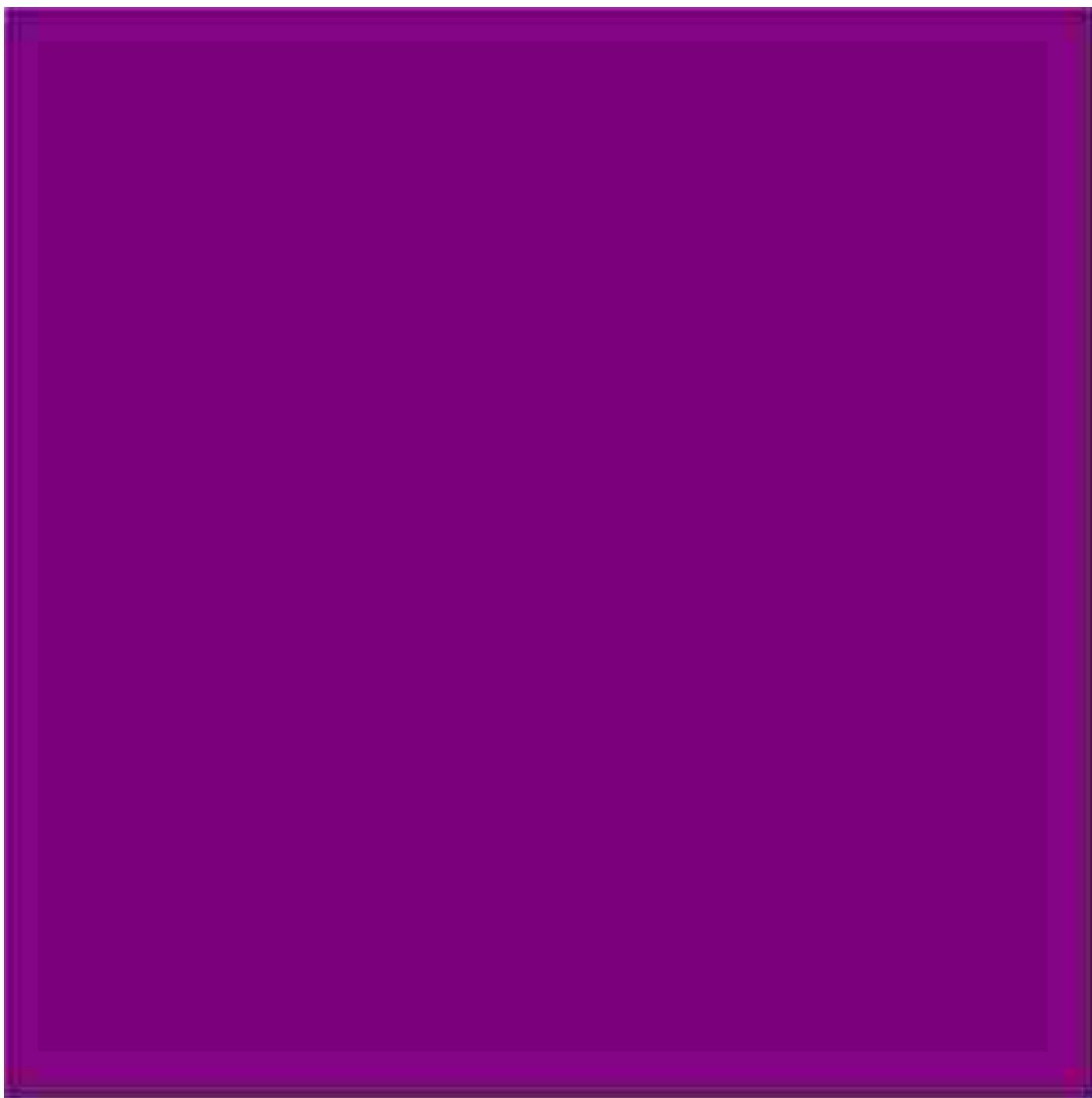
600

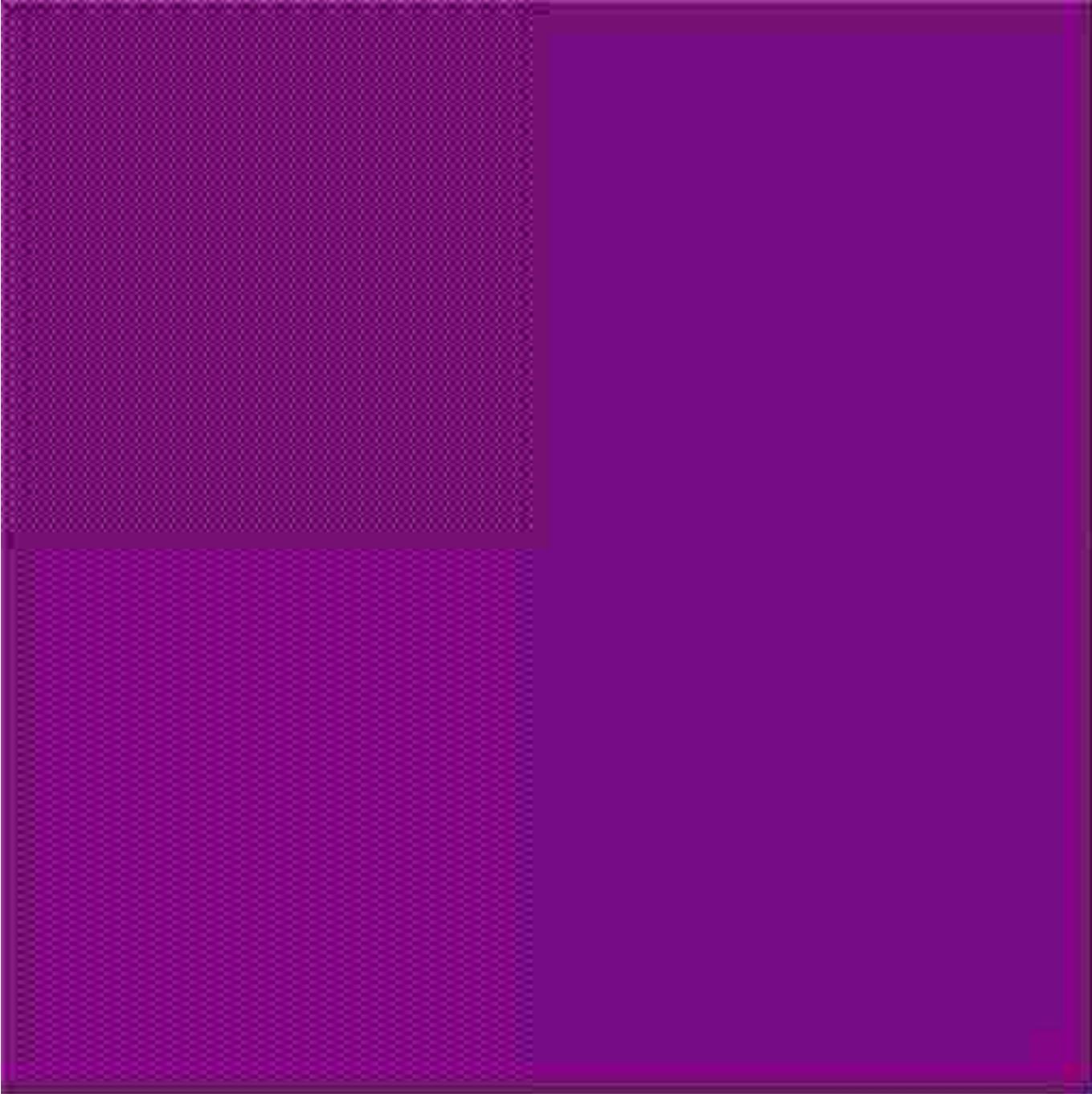
300

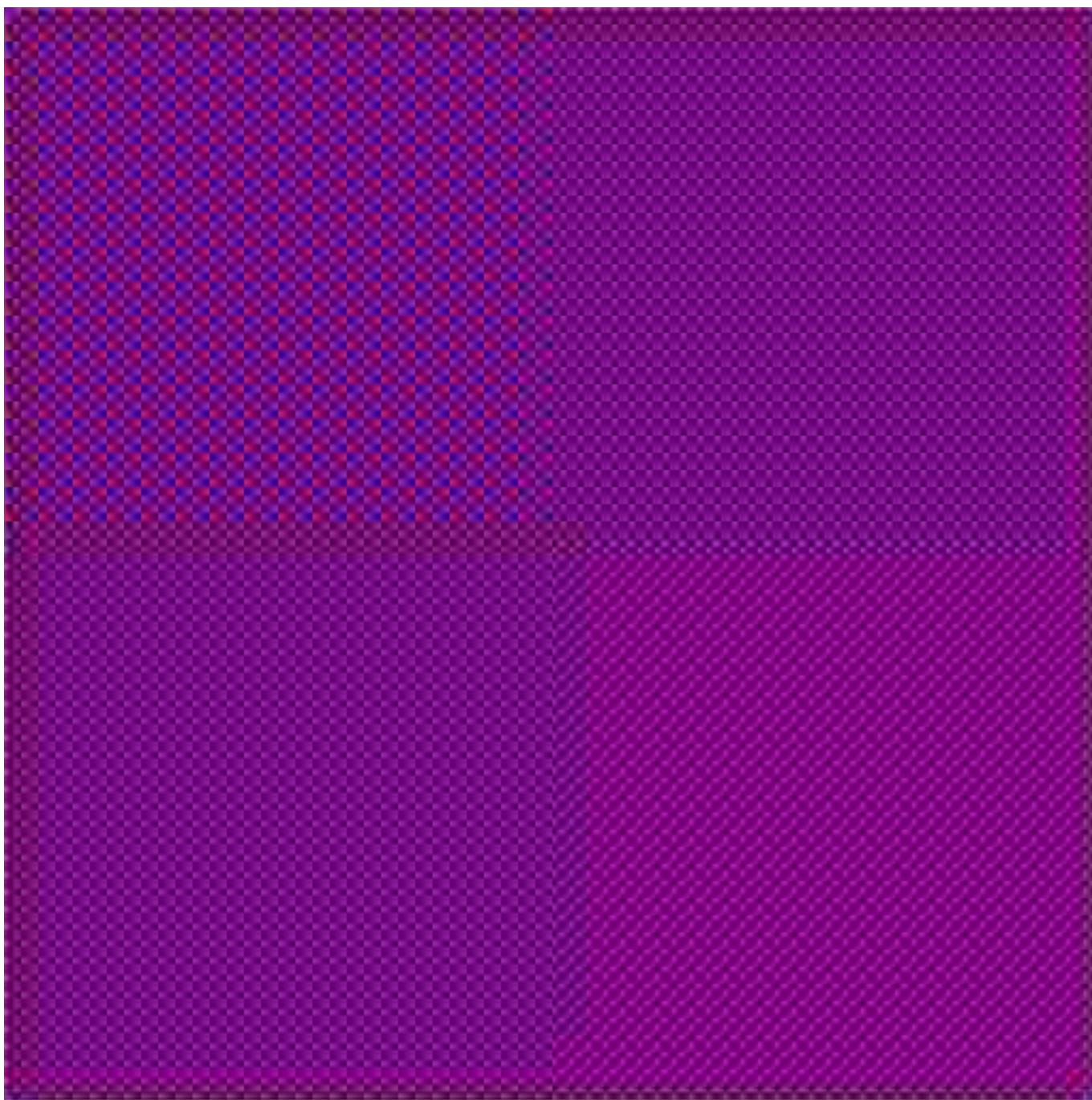
wavelength (nm)

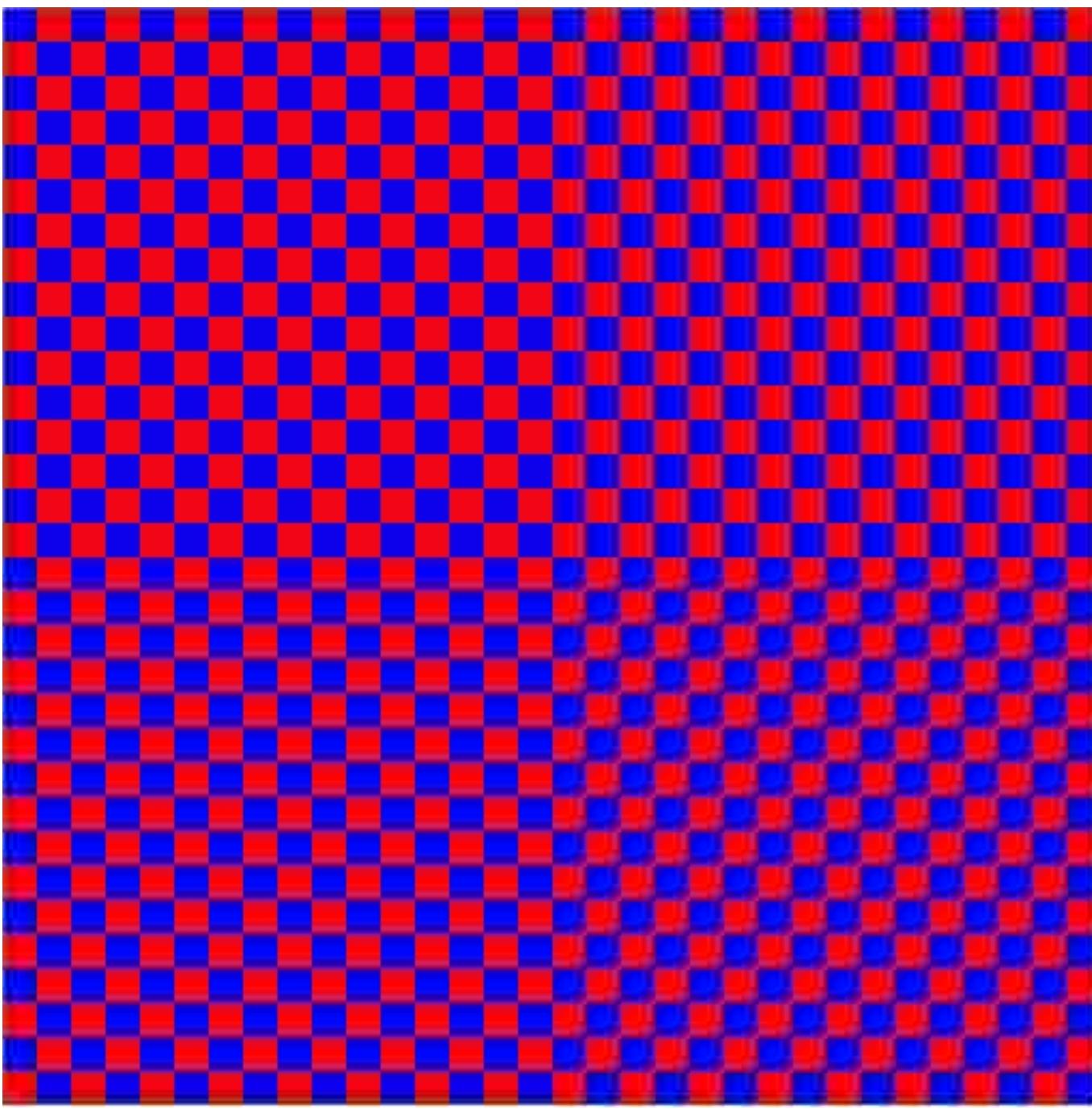


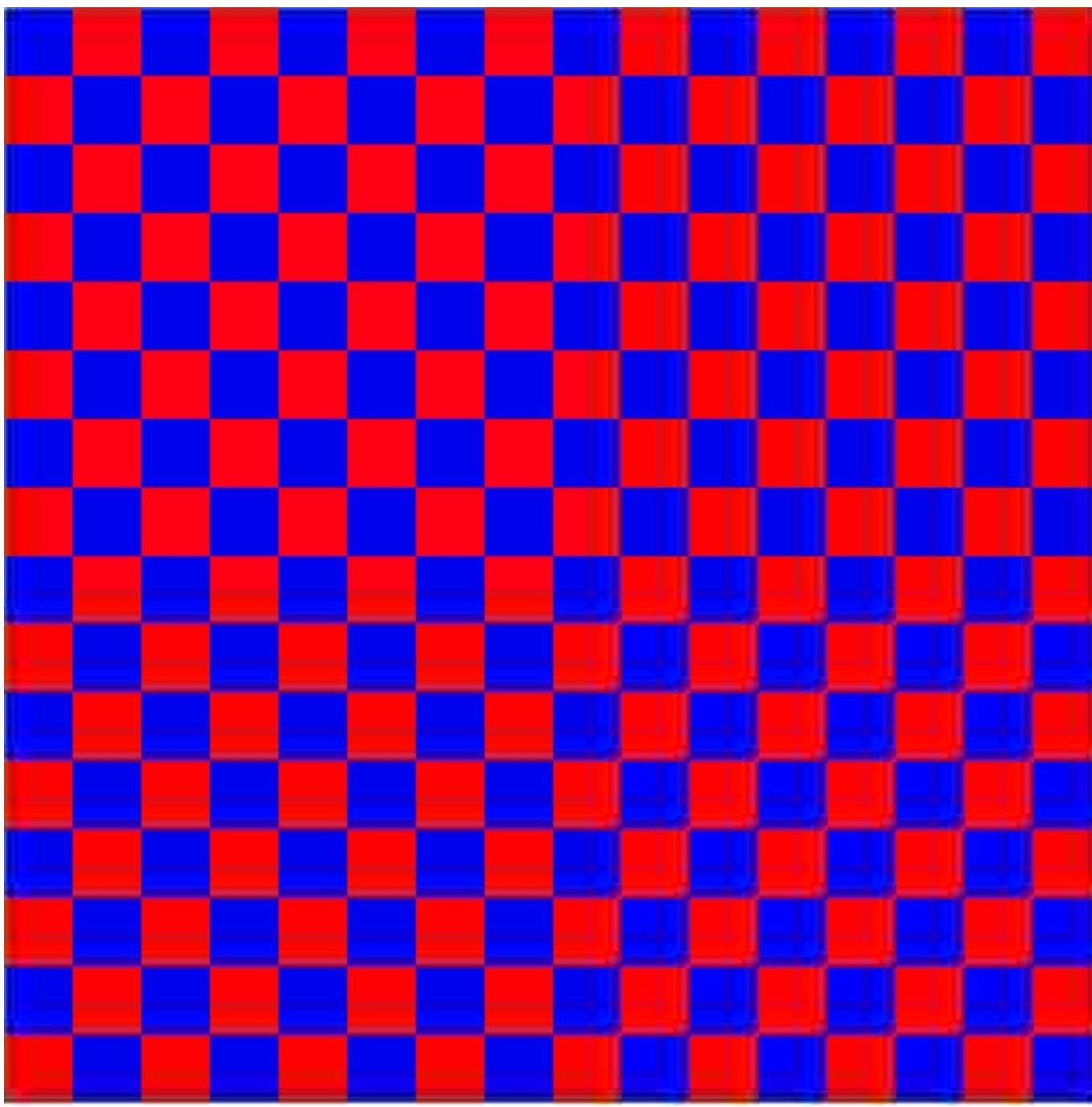


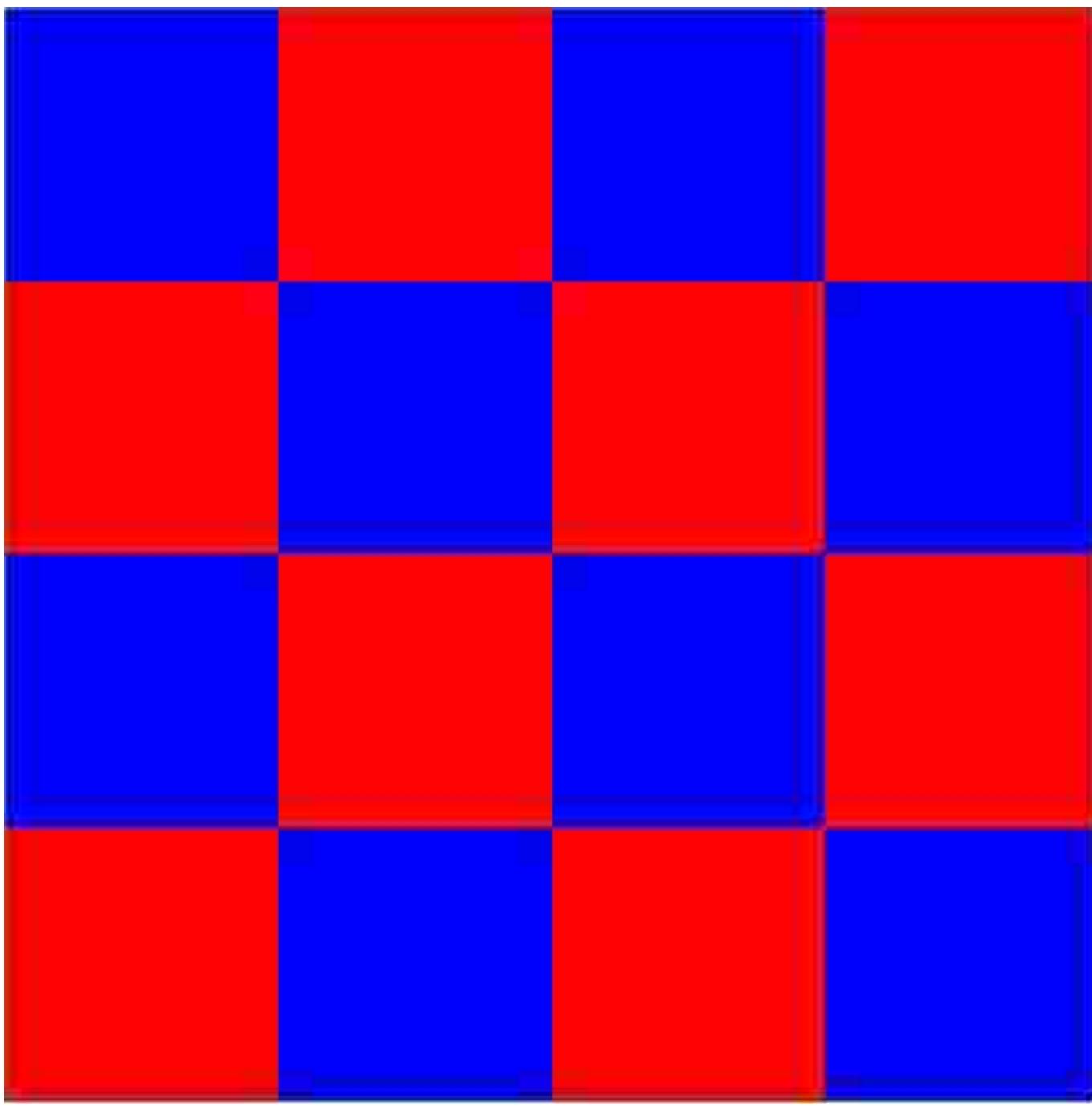


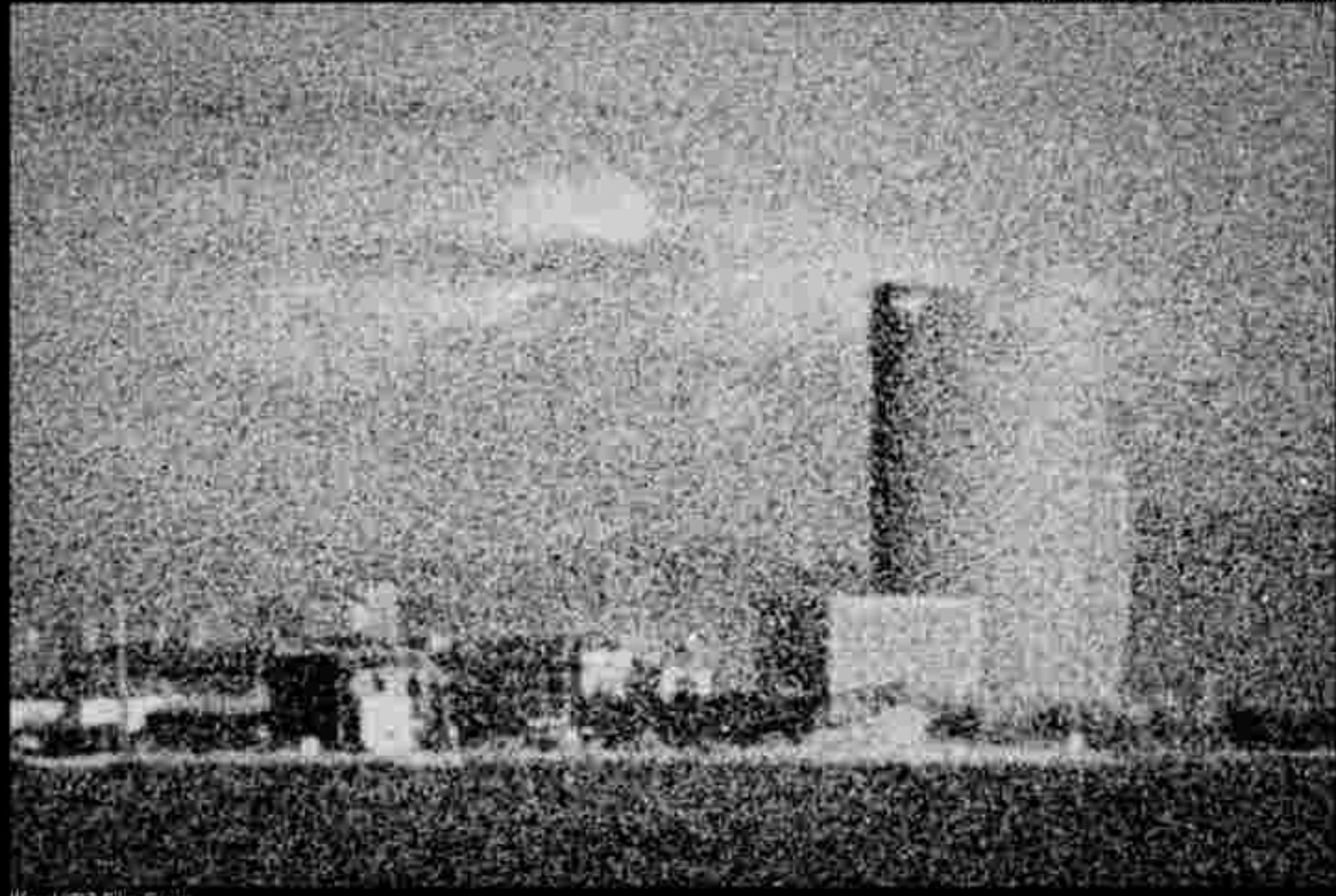


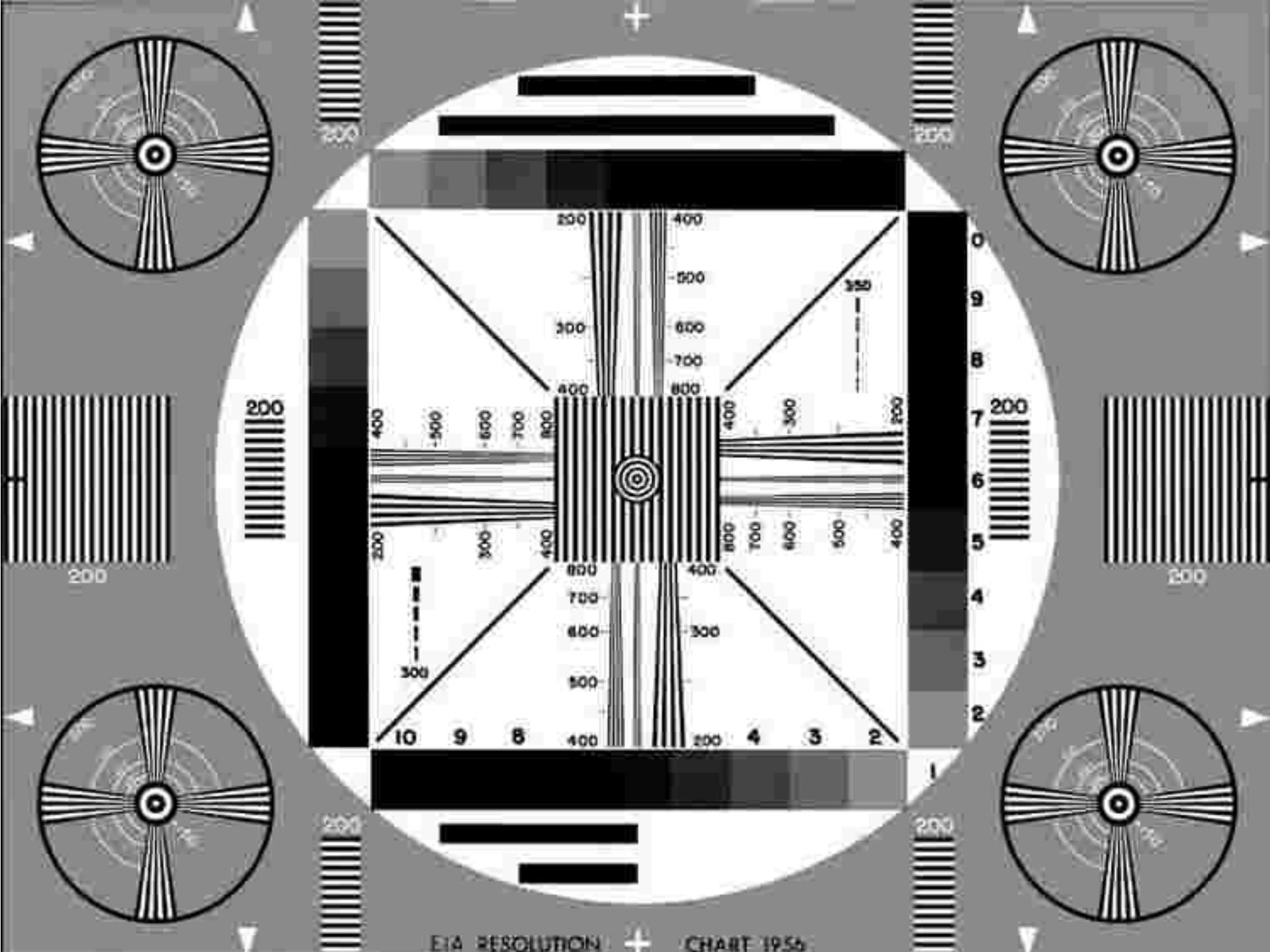










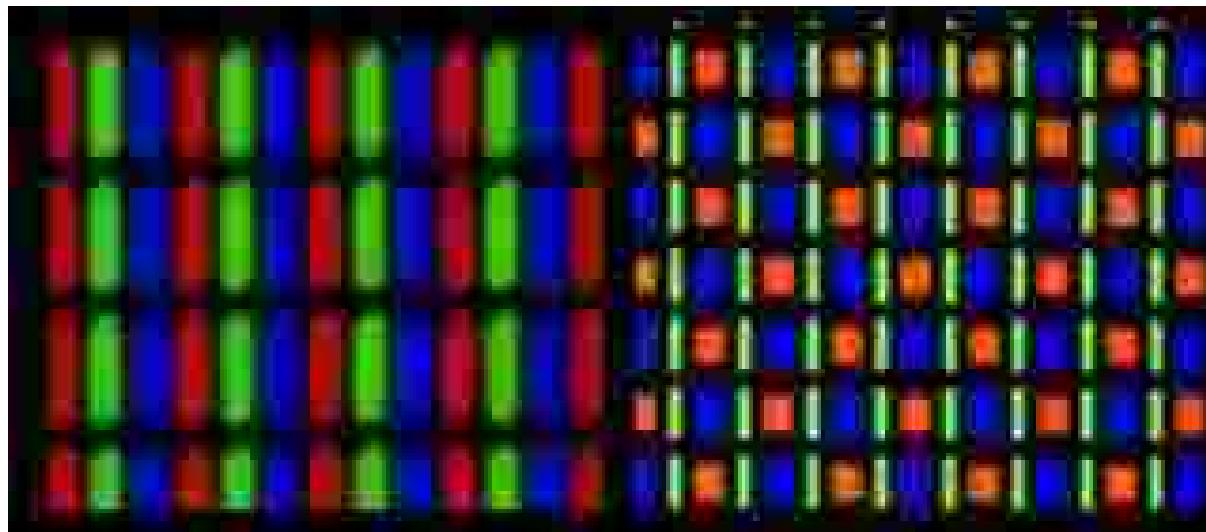


# Résolution

Résolution = taille des détails que l'on peut résoudre.

Résolution = taille d'une image en pixel.

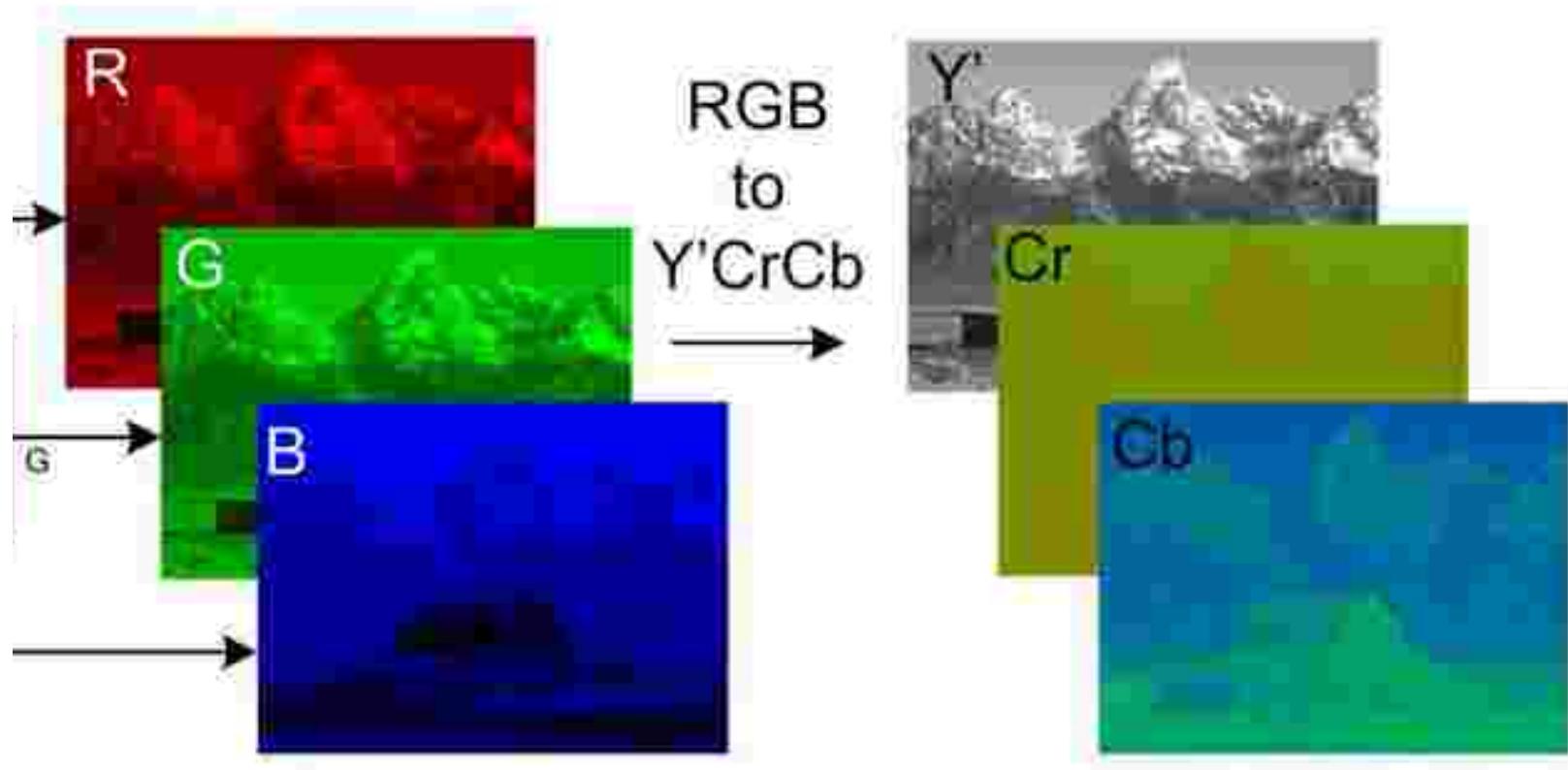
Attention, certain arrangement de pixel sur des écrans possède une résolution différente pour Y et Cb,Cr

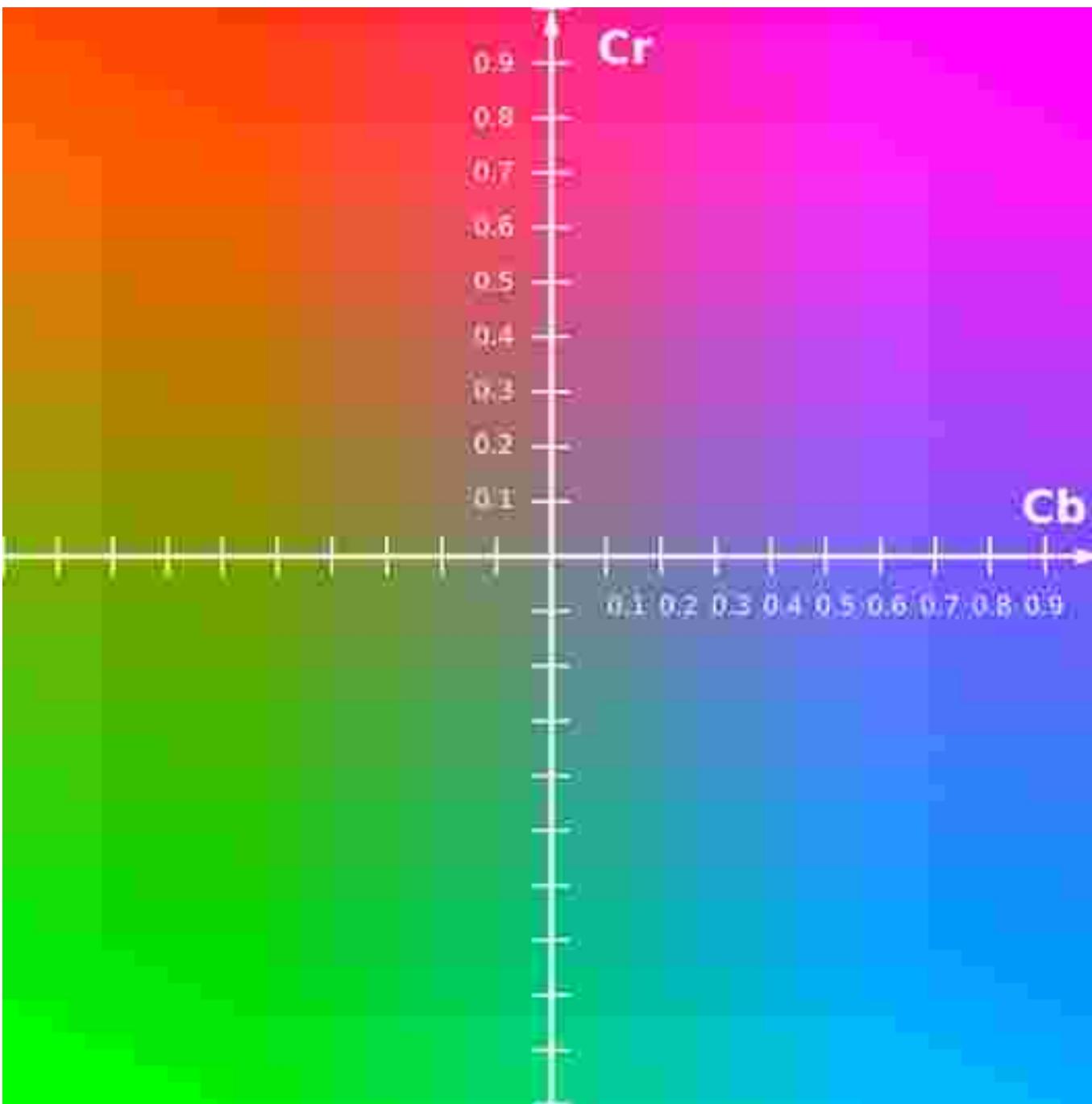




RGB

YCbCr





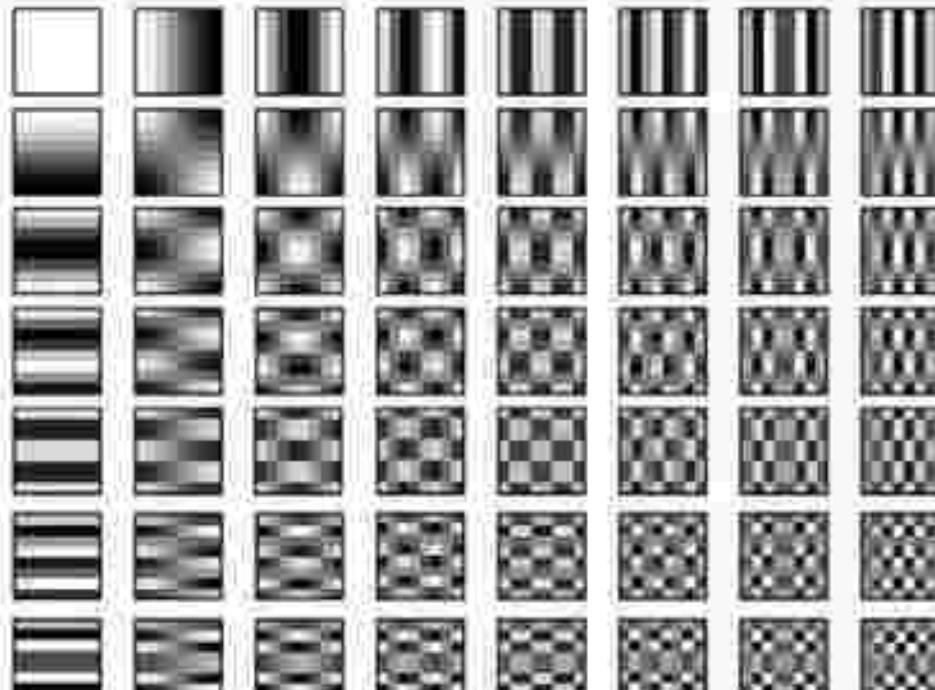
# JPEG

- Transforme l'image en YCbCr
- Résolution plus faible pour les composantes Cb et Cr
- Divise l'image en bloc
- Exprimer chaque bloc dans la base de cosinus
- Quantifier les valeurs, avec une précision plus faible pour les hautes fréquences.

# JPEG: Discrete Cosine Transform (DCT)

$$\text{basis}[i,j] = \cos\left[\pi \frac{i}{N}\left(x + \frac{1}{2}\right)\right] \times \cos\left[\pi \frac{j}{N}\left(y + \frac{1}{2}\right)\right]$$

$i = 0$



In JPEG, Apply discrete cosine transform (DCT) to each 8x8 block of image values

DCT computes projection of image onto 64 basis functions:

$\text{basis}[i,j]$

$i = 7$

$j = 0 \qquad \qquad \qquad j = 7$

DCT applied to 8x8 pixel blocks of Y' channel, 16x16 pixel blocks of Cb, Cr (assuming 4:2:0)

# Vidéo

- Image par seconde
  - 24 : premier film
  - 30 television
  - 60 : jeux, cinéma
  - 90 : réalité virtuelle
- Résolution
  - HD =  $1920 \times 1080 = 2\text{M pixels}$
  - 4K =  $3840 \times 2160 = 8\text{M pixels}$
  - 8K =  $7680 \times 4320 = 16\text{M pixels}$
- Débit
  - Très variables
- Ratio d'image
  - 4/3, 16/9
  - Anamorphique
- Immersif
  - 180, 360
  - Monoscopique ou stéréoscopique

# Vidéo

La compression vidéo utilise des techniques plus sophistiquées que la simple compression de chaque image. Les techniques modernes utilisent la grande similarité entre les images successives qui compose une séquence vidéo.

VP9 (Google, ouvert)

2160p (4k)	35-45 Mbps
1440p (2k)	10 Mbps
1080p (HD)	8,000 kbps
720p	5,000 kbps
480p	2,500 kbps
360p	1,000 kbps



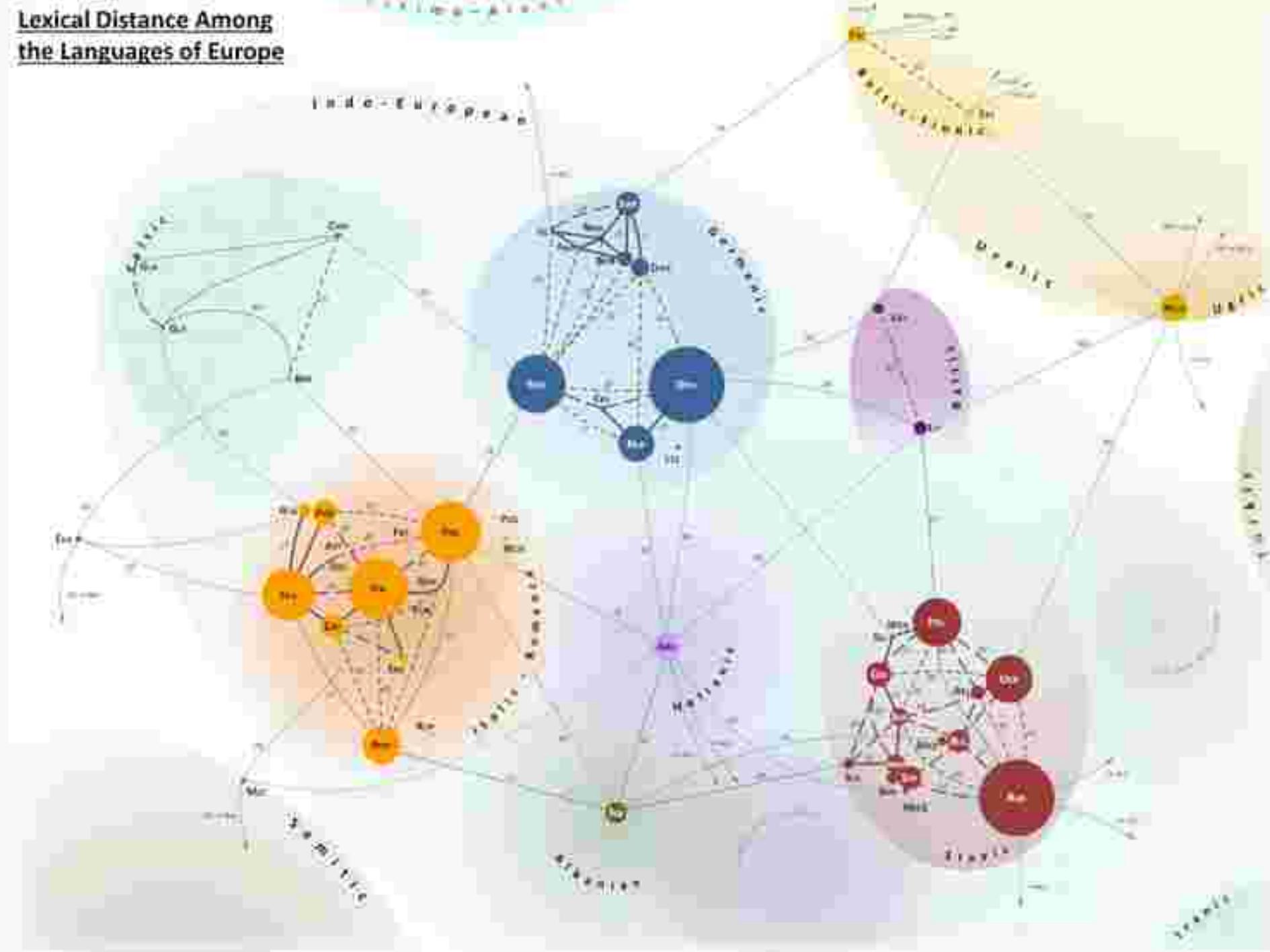


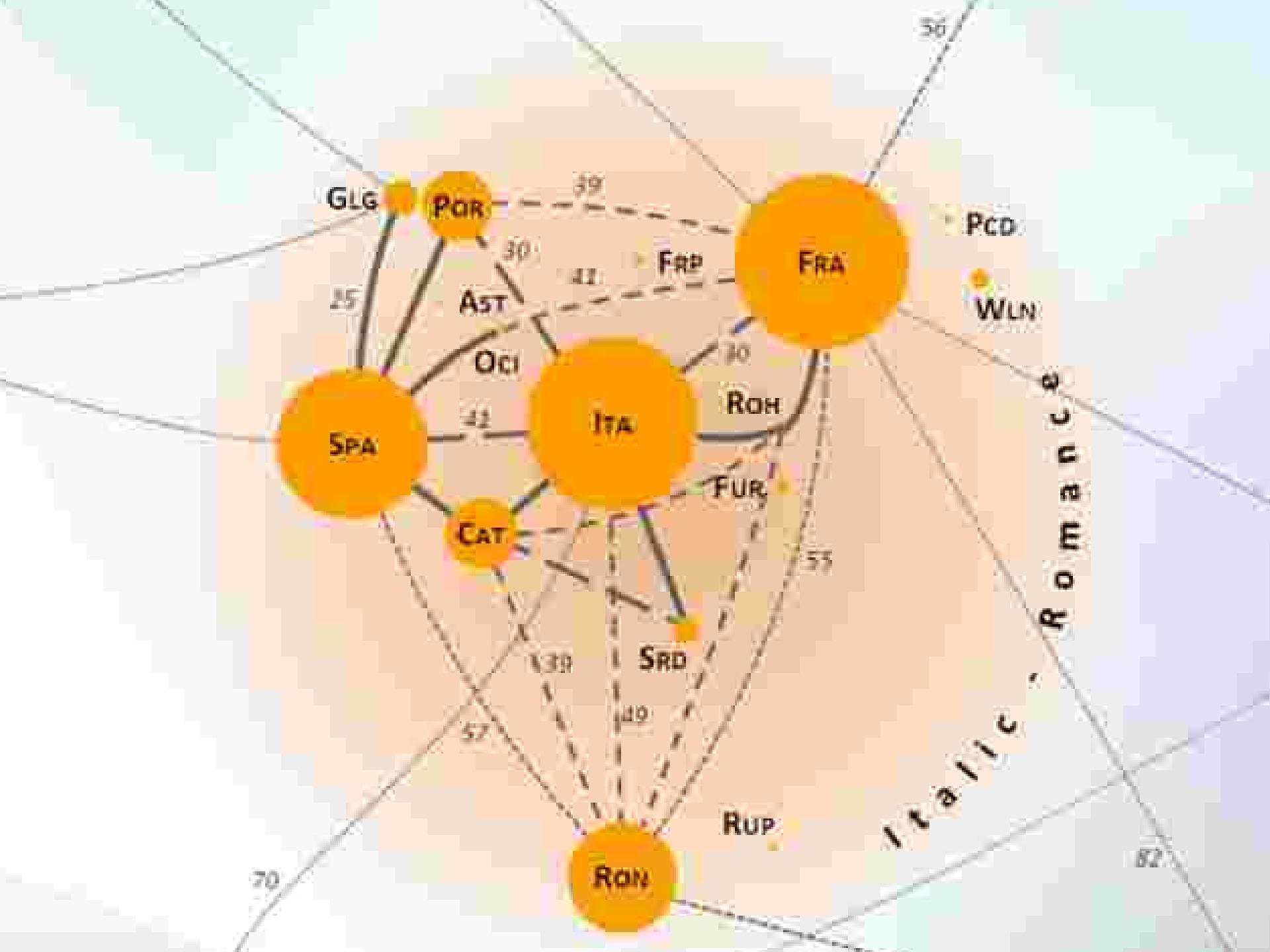
**Text**

## The INDO-EUROPEAN & URALIC LANGUAGES FAMILIES



## Lexical Distance Among the Languages of Europe





- mãe (Portuguese)
- madre (Espagnol)
- mare (Catalan)
- mère (French)
- mother (English)
- moeder (Dutch)
- mutter (German)
- Μητέρα (mitéra, Greek)
- Mama (Russian)
- eu falo (Portuguese)
- hablo (Espagnol)
- parlo (Catalan)
- io parlo (Italian)
- je parle (French)
- jeg taler (Danish)
- I speak (English)
- ik spreek (Dutch)
- ich spreche (German)



# Hiéroglyphe égyptien

	man		house, building		book, writing, abstract
	woman		town, village		small, bad, weak
	god, king		desert, foreign country		wood, tree
	force, effort		sun, light, time		<i>logogram indicator</i>
	eat, drink, speak		walk, run		<i>plural indicator</i>

- 3800 BC
- 1,000 characters
- Rébus
- Logograms



Hiéroglyphe: mayim



eau

Proto-Sinaitique



Phénicien



Grec

M μ



Paléo-hébreu

Anglais M m



Araméen



Hébreu



# The Letter Aleph

Early Hebrew



Middle Hebrew



Late Hebrew



Modern Hebrew



Greek

Roman

1

Number

# Hindi

क + आ = का	k + a = ka
क + इ = कि	k + i = ki
क + ई = की	k + ī = kī
क + उ = कु	k + u = ku
क + ऊ = कू	k + ū = kū
क + ऋ = कृ	k + ṛ = kr̥
क + ङ्ग = कृ	k + ḍ = kr̥
क + ए = के	k + e = ke
क + ऐ = कै	k + ai = kai
क + ओ = को	k + o = ko
क + औ = कौ	k + au = kau

## Korean Alphabet Chart

Consonants	Vowels									
	ㅏ (a)	ㅑ (ya)	ㅓ (o)	ㅕ (yo)	ㅗ (oh)	ㅛ (yo)	ㅜ (ow)	ㅠ (you)	ㅡ (er)	ㅣ (ee)
ㄱ(G)	가	갸	거	거	고	교	구	규	그	기
ㄴ(N)	나	냐	녀	녀	노	뇨	누	뉴	느	니
ㄷ(D)	다	댜	더	더	도	툐	두	듀	드	디
ㄹ(R/L)	라	랴	러	러	로	툐	루	류	르	리
ㅁ(M)	마	먀	머	머	모	묘	무	뮤	므	미
ㅂ(B)	바	뱌	버	벼	보	뵤	부	뷰	브	비
ㅅ(S)	사	샤	서	셔	소	쇼	수	슈	스	시
ㅇ Silent	아	야	어	여	오	요	우	유	으	이
ㅈ(J)	자	쟈	저	저	조	죠	주	쥬	즈	지
ㅊ(CH)	차	챠	처	쳐	초	쵸	추	츄	츠	치
ㅋ(K)	카	캬	커	커	코	쿄	쿠	큐	크	키
ㅌ(T)	타	탸	터	터	토	툐	투	튜	트	티
ㅍ(P)	파	파	페	페	포	표	푸	퓨	프	피
ㅎ(H)	하	핳	허	허	호	호	후	휴	흐	히

# Mandarin

Oracle Bone Script	Seal Script	Classical Script	Semi-Cursive Script	Regular Script (Traditional)	Meaning
○	日	曰	日	日	Sun
月	夕	月	月	月	Moon
山	山	山	山	山	Mountain
水	水	水	水	水	Water
雨	雨	雨	雨	雨	Rain
木	木	木	木	木	Wood
禾	禾	禾	禾	禾	Rice Plant
人	入	人	人	人	Human

# Mandarin TAXI

出租车司机

go out

rent

car

control

machine

# Japonais

hiragana

ざどーぐばいいつざまん。

katakana

ザドーグバイツザマン。

Japonais

犬は人を噛む。

トーベ・ヤンソン「たのしいムーミン一家」挿絵、タンペレ市立美術館ムーミン谷博物館、1948年 (C) Moomin Characters (TM)



「小さなバイキングビッケ」の表紙

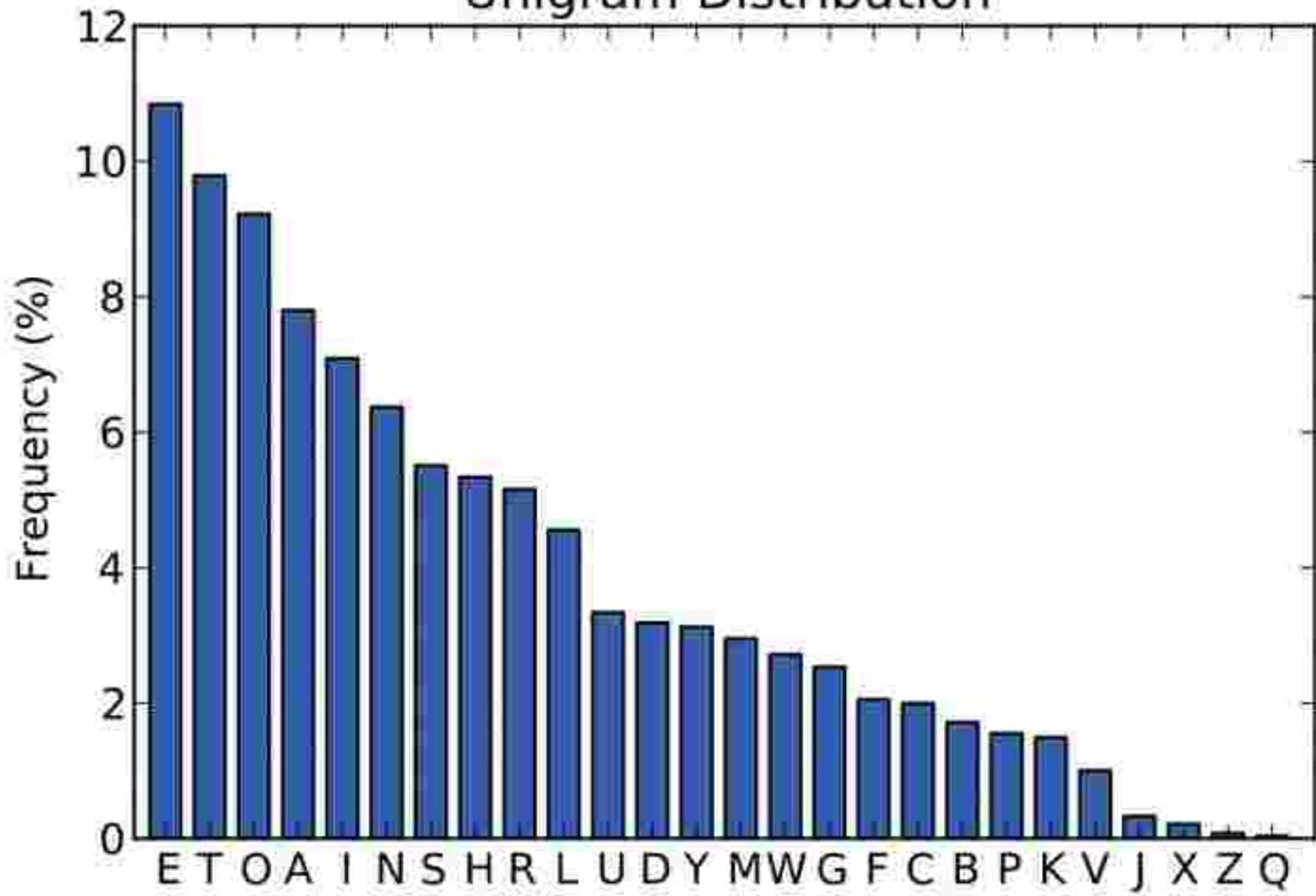


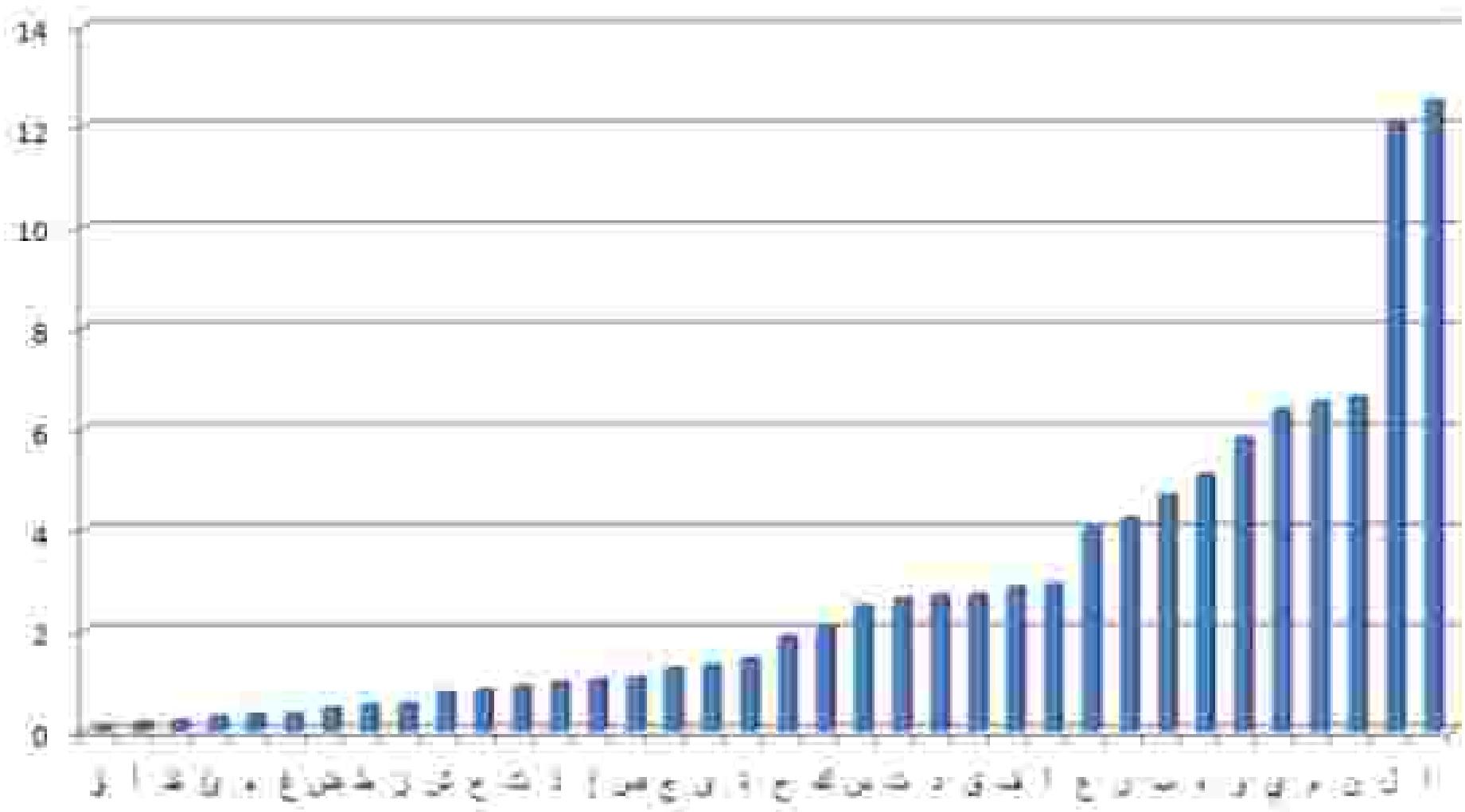
今年の大学入試センター試験で出題された、アニメの舞台となつた北欧の国を問う問題が波紋を広げている。センター側の正解では「ムーミン」はフィンランド、「小さなバイキング ビッケ」はノルウェーだが、両アニメに関係する会社は「舞台は不明」との立場。各国の大便館も発信する事態となつてゐる。

# Alphabets

Langue	Caractéristiques	Phrase
Français	consonne et voyelle	Le chien mord l'homme.
Anglais	consonne et voyelle	The dog bites the man.
Russe	consonne et voyelle	Собака кусает человека.
Coréen	Featural	개는 남자를 때린다 .
Arabe	Pas de voyelle	الكلب بعض لجن
Hindi	Consonne-voyelle	कुत्ते आदमी काटता है
Mandarin	Idée, son et mot	狗咬人了。
hiragana	syllabic	ざ ど ー ぐ ば い つ ざ ま ん .
katakana	syllabic	ザ ド ー グ バイ ツ ザ マ ソ .
Japonais	mixe	犬は人を噛む。
Hiéroglyphes	mixe	

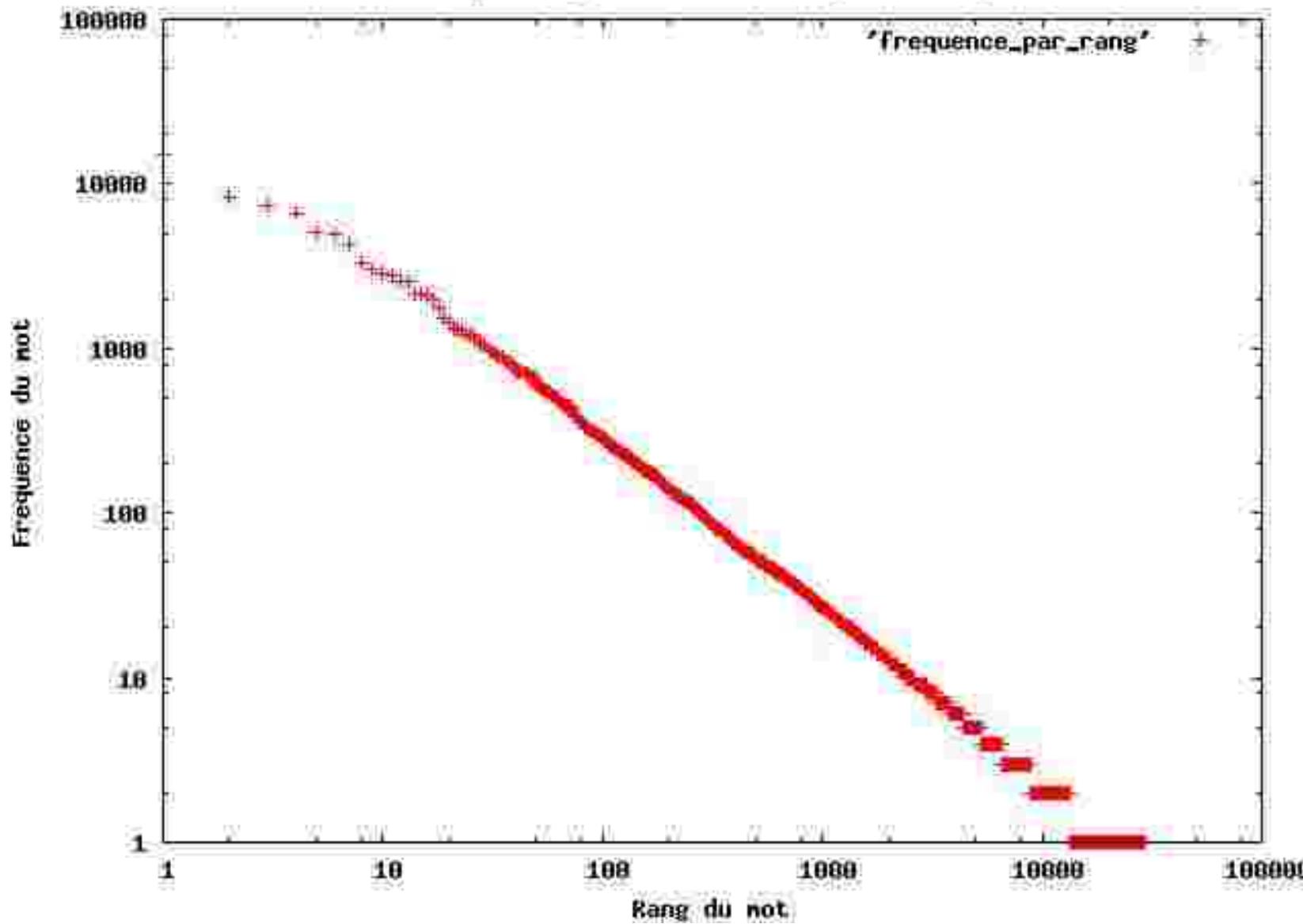
# Unigram Distribution





1 的	1826	21 時	307	41 般	208	61 子	143	81 正	114
2 我	1161	22 以	304	42 上	204	62 說	147	82 比	113
3 是	985	23 了	300	43 家	203	63 最	142	83 本	107
4 有	750	24 無	292	44 成	203	64 樣	137	84 面	106
5 不	719	25 大	286	45 作	193	65 那	137	85 下	106
6 —	692	26 都	286	46 賣	190	66 邊	136	86 從	105
7 人	597	27 廉	284	47 遇	187	67 邊	132	87 發	104
8 要	462	28 得	274	48 但	183	68 譴	132	88 同	103
9 在	459	29 年	250	49 事	179	69 心	131	89 貢	102
10 他	449	30 司	247	50 聽	176	70 只	131	90 法	102
11 這	442	31 已	240	51 後	168	71 業	130	91 屬	101
12 個	418	32 多	230	52 當	168	72 謹	129	92 果	99
13 故	391	33 好	229	53 看	168	73 定	126	93 孩	97
14 言	389	34 想	223	54 如	163	74 中	126	94 間	97
15 們	385	35 生	220	55 因	160	75 力	124	95 其	97
16 你	383	36 也	218	56 什	158	76 小	121	96 通	96
17 很	372	37 做	216	57 工	157	77 無	113	97 無	96
18 到	328	38 對	214	58 所	156	78 才	113	98 裡	96
19 來	316	39 來	213	59 出	156	79 而	116	99 重	95
20 自	313	40 去	210	60 現	156	80 然	115	100 知	95

Loi de Zipf ("Ulysse" de James Joyce (en anglais))



# Georges Perec

(édition)

## La disparition

Les éditions Nouvelles

livre



### "La disparition" de Georges Perec

"Anton Voyd n'arrivait pas à dormir. Il alluma. Son jarre marquait minuit vingt. Il poussa un profond soupir, s'assit dans son lit, s'appuyant sur son pochoir. Il prit un roman, l'ouvrit, il lut, mais il n'y saisissait qu'un imbroglio confus, il butait à tout instant sur un mot dont il ignorait la signification.

Il abandonna son roman sur son lit. Il alla à son lavabo; il mouilla son gant qu'il posa sur son front, sur son cou.

Son poils battait trop fort. Il avait chaud. Il ouvrit ses vasistas, scruta la nuit. Il faisait doux. Un bruit indistinct menait du faubourg. Un carillon, plus lourd qu'un glas, plus sonore qu'un tessin, plus profond qu'un boudon, non loin, sonna trois coups. Du canal Saint-Martin, un clapotis plaintif signalait un chaland qui passait.

Sur l'abattant du vasistas, un animal au thorax indigo, à l'aiguillon safran, ni un cafard, ni un charançon, mais plutôt un artisan, s'avancait, traînant un brin d'alfa. Il s'approcha, voulant l'aplatir d'un coup vif, mais l'animal prit son vol, disparaissant dans la nuit avant qu'il ait pu l'assassiner."

Fiche: 319 pages

Éditeur: Gallimard (16 mai 1989)

Collection: L'Imaginaire

# ASCII: American Standard Code for Information Interchange

Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	0	0		32	20	40	[Space]	64	40	100	�	96	60	140	�
1	1	1	�	33	21	41	�	65	41	101	A	97	61	141	a
2	2	2	�	34	22	42	�	66	42	102	B	98	62	142	b
3	3	3	�	35	23	43	�	67	43	103	C	99	63	143	c
4	4	4	�	36	24	44	�	68	44	104	D	100	64	144	d
5	5	5	�	37	25	45	�	69	45	105	E	101	65	145	e
6	6	6	�	38	26	46	�	70	46	106	F	102	66	146	f
7	7	7	�	39	27	47	�	71	47	107	G	103	67	147	g
8	8	10	�	40	28	50	�	72	48	110	H	104	68	148	h
9	9	11	�	41	29	51	�	73	49	111	I	105	69	149	i
10	A	12	�	42	2A	52	�	74	4A	112	J	106	6A	152	j
11	B	13	�	43	2B	53	�	75	4B	113	K	107	6B	153	k
12	C	14	�	44	2C	54	�	76	4C	114	L	108	6C	154	l
13	D	15	�	45	2D	55	�	77	4D	115	M	109	6D	155	m
14	E	16	�	46	2E	56	�	78	4E	116	N	110	6E	156	n
15	F	17	�	47	2F	57	�	79	4F	117	O	111	6F	157	o
16	10	20	�	48	30	60	�	80	50	120	P	112	70	160	p
17	11	21	�	49	31	61	�	81	51	121	Q	113	71	161	q
18	12	22	�	50	32	62	�	82	52	122	R	114	72	162	r
19	13	23	�	51	33	63	�	83	53	123	S	115	73	163	s
20	14	24	�	52	34	64	�	84	54	124	T	116	74	164	t
21	15	25	�	53	35	65	�	85	55	125	U	117	75	165	u
22	16	26	�	54	36	66	�	86	56	126	V	118	76	166	v
23	17	27	�	55	37	67	�	87	57	127	W	119	77	167	w
24	18	30	�	56	38	68	�	88	58	128	X	120	78	168	x
25	19	31	�	57	39	69	�	89	59	129	Y	121	79	169	y
26	1A	32	�	58	3A	72	�	90	5A	132	Z	122	7A	172	z
27	1B	33	�	59	3B	73	�	91	5B	133	�	123	7B	173	�
28	1C	34	�	60	3C	74	�	92	5C	134	�	124	7C	174	�
29	1D	35	�	61	3D	75	�	93	5D	135	�	125	7D	175	�
30	1E	36	�	62	3E	76	�	94	5E	136	�	126	7E	176	�
31	1F	37	�	63	3F	77	�	95	5F	137	�	127	7F	177	�

À	&Agrave;
Á	&Aacute;
Â	&Acirc;
Ã	&Atilde;
Ä	&Auml;
Å	&Aring;
Æ	&AElig;
Ç	&Ccedil;
È	&Egrave;
É	&Eacute;
Ê	&Ecirc;
Ë	&Euml;
Ì	&Igrave;
Í	&Iacute;
Î	&Icirc;
Ï	&Iuml;
Ð	&ETH;
Ñ	&Ntilde;

# HTML-ASCII

# LaTeX-ASCII

à	\`{o}	ò	\^{\prime}{o}	ó	\~{o}
á	\^{\prime}{o}	ó	\^{\prime}{o}	ó	\^{\prime}{o}
â	\.{o}	ô	\u{o}	ô	\v{o}
ã	\H{o}	õ	\t{oo}	õ	\c{o}
ä	\ddot{o}	ö	\b{o}	ö	\oe
ç	\c{c}	ç	\ae	æ	\aa
è	\`{e}	è	\`{e}	è	\`{e}
é	\^{\prime}{e}	é	\^{\prime}{e}	é	\^{\prime}{e}
ê	\.{e}	ê	\AA	ê	\o
ë	\H{e}	ë	\L	ë	\L
ì	\`{i}	ì	\`{i}	ì	\`{i}
í	\^{\prime}{i}	í	\^{\prime}{i}	í	\^{\prime}{i}
î	\.{i}	î	\AA	î	\AA
ï	\H{i}	ï	\L	ï	\L

# ASCII / ISO 8859-1 (Latin-1)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20	!	“	#	\$	%	&	‘	(	)	*	+	,	=	.	/	
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	^	_	
60	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{		}	~	
80	€	ƒ	„	…	‡	‡	^	‰	₪	₵	₵	₵	₵	₵	₵	₵
90	“	”	”	”	—	—	—	TM	®	®	®	®	®	®	®	®
A0	í	é	£	¤	™	§	“	©	ª	«	¬	¬	®	—	—	—
B0	°	±	²	³	µ	¶	·	·	·	»	¼	½	¾	½	½	½
C0	À	Á	Â	Ã	Ä	Å	Æ	È	É	Ê	Ë	Ì	Í	Î	Ï	Ï
D0	Ð	Ñ	Ò	Ó	Ô	Õ	×	Ø	Ù	Û	Ü	Ü	Ý	Þ	Þ	Þ
E0	à	á	â	ã	ä	å	æ	è	é	ê	ë	ì	í	î	ï	ï
F0	ð	ñ	ò	ó	ô	õ	÷	ø	ù	û	ü	ü	ý	þ	ý	ý

# Unicode

**Unicode** est un standard pour le codage, la représentation et le traitement cohérents du texte exprimé dans la plupart des systèmes d'écriture du monde. La norme est maintenue par le consortium Unicode et, à partir de juin 2018, la version la plus récente, Unicode 11.0, contient un répertoire de 137 439 caractères couvrant 146 scripts modernes et historiques, ainsi que plusieurs jeux de symboles et emoji.

**UTF-8** est un codage de caractères à largeur variable capable de coder tous les 1 112 064 points de code valides dans Unicode en utilisant un à quatre octets de 8 bits.

# Définition du nombre d'octets utilisés dans le codage (uniquement les séquences valides)

Caractères codés	Représentation binaire UTF-8	Premier octet valide (hexadécimal)	Signification
U+0000 à U+007F	0xxxxxx	00 à 7F	1 octet, codant 7 bits
U+0080 à U+07FF	110xxxxx 10xxxxxx	C2 à DF	2 octets, codant 11 bits
U+0800 à U+0FFF	11100000 101xxxxx 10xxxxxx	E0 (le 2 <sup>e</sup> octet est restreint de A0 à BF)	3 octets, codant 16 bits
U+1000 à U+1FFF	11100001 10xxxxxx 10xxxxxx	E1	
U+2000 à U+3FFF	1110001x 10xxxxxx 10xxxxxx	E2 à E3	
U+4000 à U+7FFF	111001xx 10xxxxxx 10xxxxxx	E4 à E7	
U+8000 à U+BFFF	111010xx 10xxxxxx 10xxxxxx	E8 à EB	
U+C000 à U+CFFF	11101100 10xxxxxx 10xxxxxx	EC	
U+D000 à U+D7FF	11101101 100xxxxx 10xxxxxx	ED (le 2 <sup>e</sup> octet est restreint de 80 à 9F)	
U+E000 à U+FFFF	1110111x 10xxxxxx 10xxxxxx	EE à EF	
U+10000 à U+1FFFF	11110000 1001xxxx 10xxxxxx 10xxxxxx	F0 (le 2 <sup>e</sup> octet est restreint de 90 à BF)	4 octets, codant 21 bits
U+20000 à U+3FFFF	11110000 101xxxxx 10xxxxxx 10xxxxxx		
U+40000 à U+7FFFF	11110001 10xxxxxx 10xxxxxx 10xxxxxx	F1	
U+80000 à U+xFFFFF	1111001x 10xxxxxx 10xxxxxx 10xxxxxx	F2 à F3	
U+100000 à U+10FFFF	11110100 1000xxxx 10xxxxxx 10xxxxxx	F4 (le 2 <sup>e</sup> octet est restreint de 80 à 8F)	

# **Données structurées**

# **Donné structuré**

- Binaire
  - Image, son, etc...
- Nominal
  - Données catégoriques
  - Bouléen
- Numérique
  - Données ordinales
  - Quasi Continu

# Donné structure : Nominal

- Ensemble fini non ordonné
- Données catégoriques:  
  {Bleu, Blanc, Rouge}  
  {Célibataire, conjoint de fait,  
  marié, veuf}
- Bouléen, mariées: {Vrai, Faux}
- Représentation OneHot ADN:  
  {A, C, G, T}=  
  {(1, 0, 0, 0), (0, 1, 0, 0), (0, 0, 1, 0), (0, 0, 0, 1)}

# Donné structure : Numérique

- Ordonné avec un ordre totale
- Borné ou non borné
- Fini ou infini
- Données ordinales
  - Âge : entier entre 0 et 150 ans
  - Niveau de scolarité :  
{Primaire, Secondaire, Collégial, Universitaire, Docteur}
- Quasi continue
  - Échelle (les ratios sont comparables)
  - Poids en kg, Distance en km
- Multidimensionnel
  - Vecteur, matrice, tenseur.
  - Image: (tenseur a 3 dimensions)  
Longueur x Largeur x (R,V,B), 3 dimensions réelles

# Tables *Spreadsheet*

Transactions - Microsoft Excel

The screenshot shows a Microsoft Excel spreadsheet titled "Transactions - Microsoft Excel". The data is organized into columns representing various financial transactions. The columns include Date, Type, Symbol, Shares, Price, Costs, Fees, Total, DistrButh, Shares, AffCurrency, Exchange, CashAffect, Name, and Comment. The data rows show purchases of Lufkin Industries (LUFK) and FuelTech Inc. (FTEK), as well as sales of LUFK and FTEK, and a cash transaction. The "AffCurrency" column consistently shows USD, indicating all transactions are in US dollars.

	Date	Type	Symbol	Shares	Price	Costs	Fees	Total	DistrButh	Shares	AffCurrency	Exchange	CashAffect	Name	Comment
3	7/3/2007	CASH_IN					0	50000			USD		TRUE		
4	4/11/2011	BUY	LUFK	100	89.41	10	-8951				USD		TRUE	Lufkin Industries	
5	6/23/2011	BUY	LUFK	200	65.43	10	-13092				USD		TRUE	Lufkin Industries	
6	9/2/2011	BUY	LUFK	150	58.82	10	-8824.5				USD		TRUE	Lufkin Industries	
7	3/14/2012	SELL	LUFK	250	78.41	10	-20002.5				USD		TRUE	Lufkin Industries	
8	7/11/2011	BUY	FTEK	200	7.19	10	-1448				USD		TRUE	FuelTech Inc.	
9	8/12/2011	BUY	FTEK	200	5.4	10	-1090				USD		TRUE	FuelTech Inc.	
10	9/2/2011	BUY	FTEK	150	5.85	10	-877.5				USD		TRUE	FuelTech Inc.	
11	5/8/2012	SELL	FTEK	500	4.65	10	-2324				USD		TRUE	FuelTech Inc.	
12	6/16/2011	BUY	GS	50	136.05	10	-6814.5				USD		TRUE	Goldmann Sachs	
13	10/3/2011	BUY	GS	50	90.08	10	-4534				USD		TRUE	Goldmann Sachs	
14	3/26/2012	SELL	GS	25	120.07	10	-3002.5				USD		TRUE	Goldmann Sachs	
15	5/12/2010	BUY	JNJ	100	58.45	10	-5855				USD		TRUE	Johnson & Johnson	

# **Tables CVS**

# ***Comma-Separated Values***

**CSV**, est un format informatique ouvert représentant des données tabulaires sous forme de valeurs séparées par des virgules. Ce format n'a jamais vraiment fait l'objet d'une spécification formelle. Toutefois, la RFC 41801 décrit la forme la plus courante et établit son type MIME « text/csv ».

## Un document CVS

- est du texte brut utilisant un jeu de caractères tel que ASCII, divers jeux de caractères Unicode (par exemple, UTF-8), EBCDIC ou Shift JIS,
- se compose d'enregistrements (généralement un enregistrement par ligne),
- avec les enregistrements divisés en champs séparés par des délimiteurs (généralement un seul caractère réservé tel que virgule, point-virgule ou tabulation; parfois le séparateur peut inclure des espaces facultatifs),
- où chaque enregistrement a la même séquence de champs.

# Table

Prenom	Nom	Email	Âge	Ville
Robert	Lepingre	bobby@exemple.com	41	Paris
Jeanne	Ducoux	jeanne@exemple.com	32	Marseille
Alain	Tapp	alain.tap@gmail.com		Montréal
Pierre	Lenfant	pierre@exemple.com	23	Rennes

## Table en format CVS

```
Prenom;Nom;Email;Âge;Ville
Robert;Lepingre;bobby@exemple.com;41;Paris
Jeanne;Ducoux;jeanne@exemple.com;32;Marseille
Alain;Tapp;alain.tap@gmail.com;;Montréal
Pierre;Lenfant;pierre@exemple.com;23;Rennes
```

# JSON

JavaScript Object Notation (JSON) est un format de données textuelles dérivé de la notation des objets du langage JavaScript. Il permet de représenter de l'information structurée comme le permet XML par exemple. Créé par Douglas Crockford entre 2002 et 2005, il est décrit par la RFC 7159 de l'IETF.

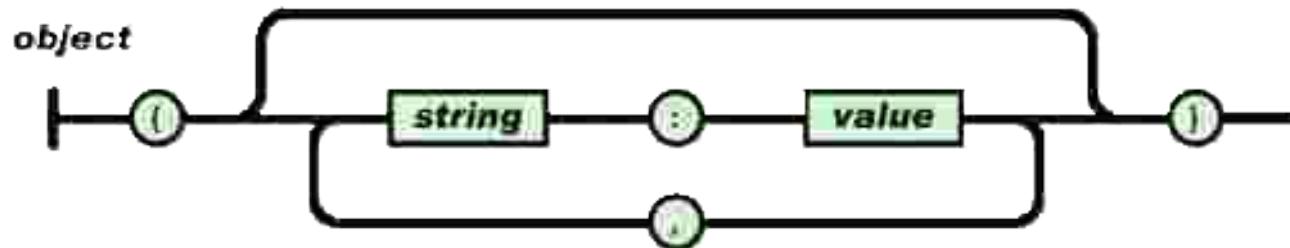
Un document JSON ne comprend que deux types d'éléments structurels :

- des ensembles de paires « nom » (alias « clé ») / « valeur » ;
- des listes ordonnées de valeurs.

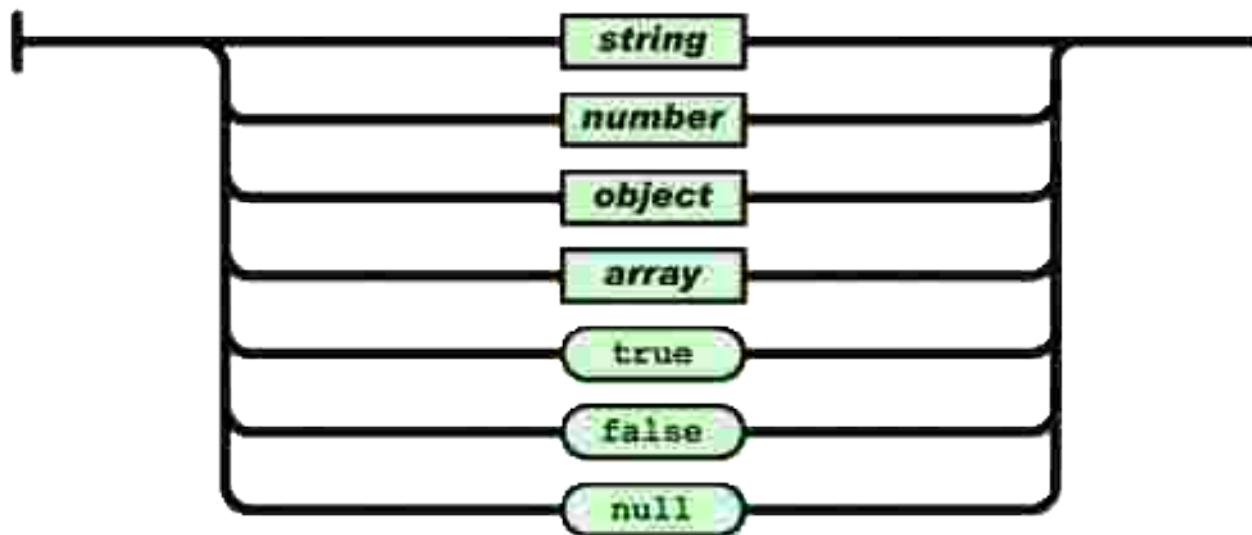
Ces mêmes éléments représentent trois types de données :

- des objets ;
- des tableaux ;
- des valeurs génériques de type tableau, objet, booléen, nombre, chaîne ou null.

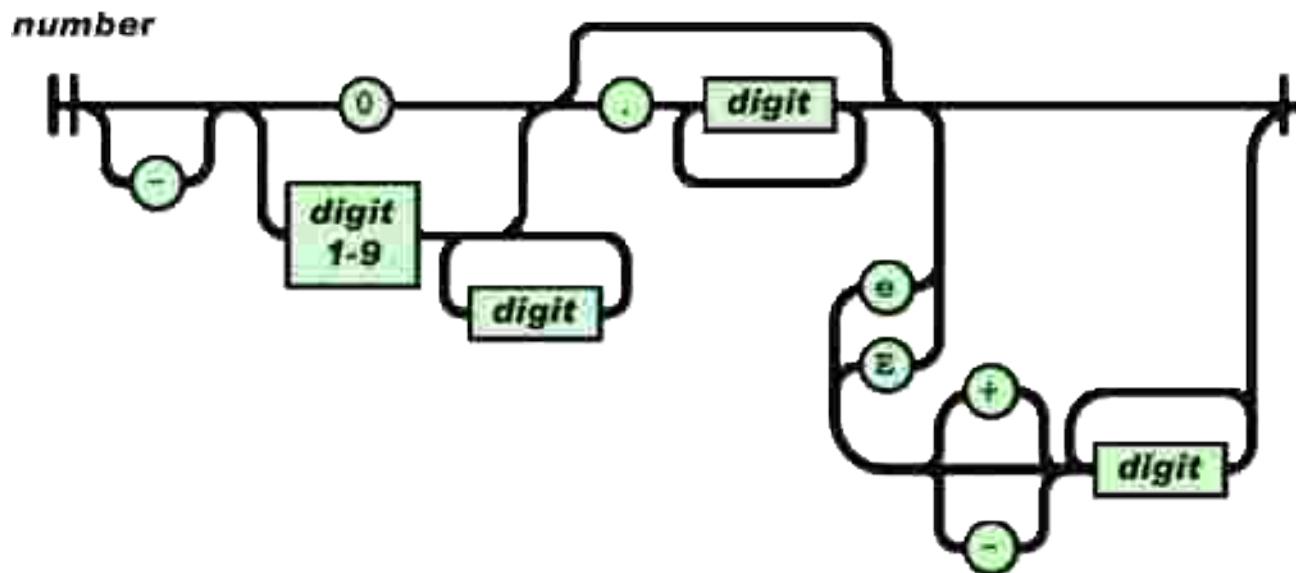
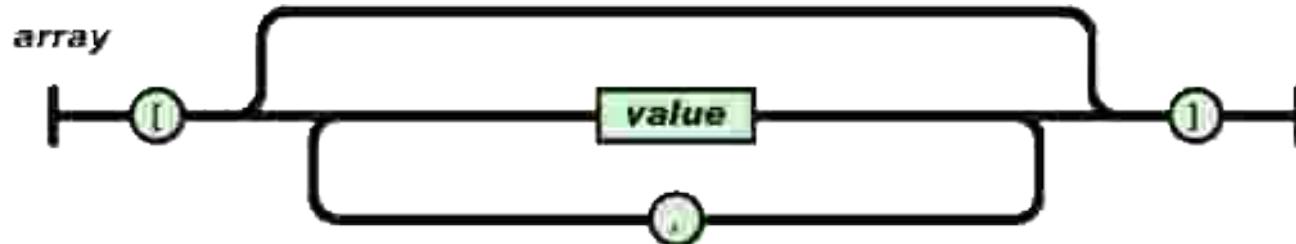
# JSON



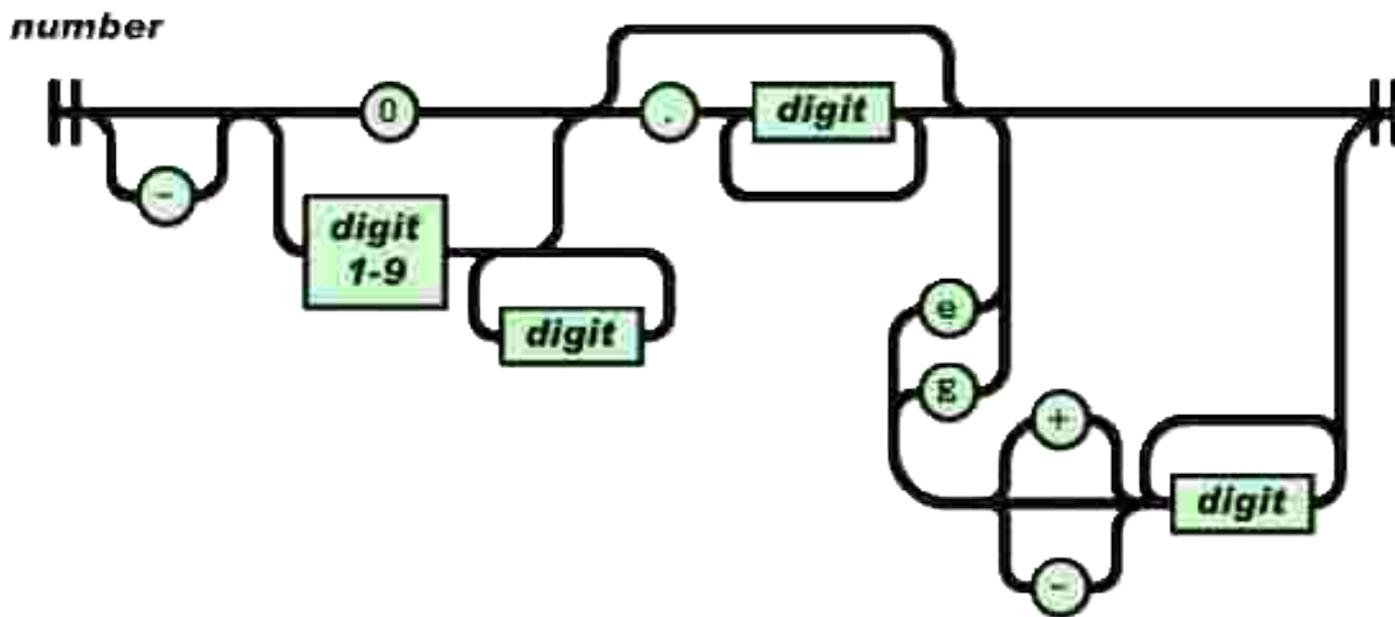
*value*



# JSON



# JSON



# JSON examples

```
{"menu": {  
    "id": "file",  
    "value": "File",  
    "popup": {  
        "menuitem": [  
            {"value": "New", "onclick":  
"CreateNewDoc()"},  
            {"value": "Open", "onclick":  
"OpenDoc()"},  
            {"value": "Close", "onclick":  
"closeDoc()"}  
        ]  
    }  
}}
```

```
{"widget": {  
    "debug": "on",  
    "window": {  
        "title": "Sample Konfabulator Widget",  
        "name": "main_window",  
        "width": 500,  
        "height": 500  
    },  
    "image": {  
        "src": "Images/Sun.png",  
        "name": "sun1",  
        "hOffset": 250,  
        "vOffset": 250,  
        "alignment": "center"  
    },  
    "text": {  
        "data": "Click Here",  
        "size": 36,  
        "style": "bold",  
        "name": "text1",  
        "hOffset": 250,  
        "vOffset": 100,  
        "alignment": "center",  
        "onMouseUp": "sun1.opacity = (sun1.opacity / 100) *  
90;"  
    }  
}}}
```

# **Collecte de données**

# Collecte de données

- Données existantes:
  - Collecter
  - Regrouper
  - Nettoyer
  - Questioner
- Crédit de données
  - Mise en place de collecteur
  - *Crowdsourcing, Amazon Mechanical Turk*

# Turc mécanique 1770



Le Turc Mécanique  
Automate qui joue aux Echecs avec un grand succès et par la force de sa propre volonté.

# Micro-travail

Amazon *Mechanical Turk* (AMT, en français « Turc mécanique d'Amazon ») est un service de micro-travail lancé par Amazon.com fin 2005. C'est une plateforme web de *crowdsourcing* qui vise à faire effectuer par des humains, contre rémunération, des tâches plus ou moins complexes. Les tâches en question doivent être dématérialisées ; il s'agit souvent d'analyser ou de produire de l'information dans des domaines où l'intelligence artificielle est encore trop peu performante, par exemple l'analyse du contenu d'images.

# Lac de donné (*Data lake*)

Un lac de données est un système ou un référentiel de données stockées dans son format naturel. Un lac de données est généralement **un magasin unique de toutes les données de l'entreprise**, y compris les copies brutes des données du système source et les données transformées utilisées pour des tâches telles que le reporting, la visualisation, l'analyse et l'apprentissage automatique. Un *Data Lake* peut inclure des données structurées issues de bases de données relationnelles (lignes et colonnes), des données semi-structurées (CSV, logs, XML, JSON), des données non structurées (emails, documents, PDF) et des données binaires (images, audio, vidéo).

Hortonworks, Google, Oracle, Microsoft, Zaloni, Teradata, Cloudera et Amazon proposent tous des solutions de lac de donné.

# Entrepôt de données

Le terme **entrepôt de données** (ou base de données décisionnelle ; *data warehouse* ou **DWH**) désigne une base de données utilisée pour collecter, ordonner, journaliser et stocker des informations provenant de bases de données opérationnelles et fournir ainsi un socle à l'aide à la décision en entreprise.

# **Nettoyage de données**

# Nettoyage de données

Le nettoyage de données est l'opération de détection et de correction (ou suppression) d'erreurs présentes sur des données stockées dans des bases de données ou dans des fichiers.

Le nettoyage de données est un des problèmes majeurs des entrepôts de données.

Les données présentes dans les bases de données peuvent avoir plusieurs types d'erreurs comme des erreurs de frappe, des informations manquantes, des imprécisions etc. La partie impropre de la donnée traitée peut être remplacée, modifiée ou supprimée. Le processus de nettoyage identifie les données erronées et les corrige automatiquement avec un programme informatique ou les propose à un humain pour qu'il effectue les modifications.

Le nettoyage de données est différent de la validation de données. La validation de données est l'étape qui consiste à vérifier et rejeter les données qui ne respectent pas certaines règles avant l'ajout en base de données, alors que le nettoyage intervient après (sur des données déjà présentes en base de données).

Les approches classiques de nettoyage utilisent les contraintes d'intégrité, les statistiques ou l'apprentissage automatique pour nettoyer les données.

# Types d'erreurs

- Erreurs de syntaxe
  - Erreurs lexicales
  - Erreurs de formatage
  - Erreurs d'irrégularité
- Erreurs sémantiques
  - Violation des contraintes d'intégrité
  - Erreurs de contradiction
  - Erreurs de duplication
  - Erreurs de donnée invalide
- Erreurs de couverture
  - Valeur manquante
  - Donnée manquante

# Erreurs de syntaxe

Une **erreur lexicale** est une divergence entre le nom de la donnée attendu et le format spécifié.

Prénom	Âge
Alice	F
Bob	M
Charlie	21

Une **erreur de formatage** est une entrée qui ne correspond pas à un modèle donné.

Nom
Doe, John
John Smith

Une **erreur d'irrégularité** se produit lorsqu'une donnée n'est pas représentée de façon régulière, en suivant le même schéma, la même suite logique d'écriture. Cela peut intervenir pour la représentation de distance par exemple en utilisant différents systèmes métriques.

Distance
25 km
23 km
20 mi

# Erreurs sémantiques

- **Violation des contraintes d'intégrité**
  - Ce sont les erreurs qui ne respectent pas les règles d'intégrité du schéma de données.
  - Ce sont souvent des règles de bon sens comme la vérification d'une valeur supérieure à 0 pour l'âge d'une personne ou encore la vérification de l'unicité d'une clé primaire dans une base de données.
- **Erreurs de contradiction**
  - Les erreurs de contradiction sont des contradictions dans les données.
  - Par exemple une erreur de contradiction peut intervenir lorsque l'âge spécifié ne correspond pas à la date de naissance.
- **Erreurs de duplication**
  - Les erreurs de duplication surviennent lorsque plusieurs occurrences de la même donnée sont stockées.
  - Ces erreurs peuvent être vues comme un cas spécifique des erreurs de contradiction.
- **Erreurs de donnée invalide**
  - Ce sont des entrées qui sont dites invalides mais qui ne sont pas détectables par la mise en place de contraintes. Ce sont des erreurs qui sont spécifiques au domaine.

# Erreurs de couverture

Prénom	Âge	Sexe	Taille
Alice	23	F	1,70
Bob	34	M	1,82
Charlie	19	M	

Valeur manquante

Prénom	Âge	Sexe	Taille
Alice		F	1,70
Bob		M	1,86
Charlie		M	1,68

Donnée manquante

# Nettoyage

- *Parsing*
- Transformation de donnée
- Renforcement des contraintes d'intégrité
- Méthode statistique
- *Crowdsourcing*

# Nettoyage: *parsing*

La méthode de *parsing* est utilisée pour la détection d'erreurs de syntaxe. Un parseur décide de l'acceptabilité de la donnée représentée par une chaîne de caractères. Il s'assure que la donnée suit la spécification.

Ce type d'approche requiert un set de données qui peut être converti en distance pour pouvoir être comparé.

Cette approche a ses limites. En effet, celle-ci se base généralement sur des expressions régulières pour déterminer la validité d'une entrée. Ce processus de vérification peut être amélioré avec des techniques d'apprentissage automatique.

- Expression régulières
- Grammaires hors contexte
- Autres

# Expressions régulières : aperçu

## NOTE PAD++

- Un caractère: .
  - Chiffre: \d
  - LF: \n
  - Space : \s
- Répétition:
  - Un ou plusieur: +
  - Aucun, un ou plusieurs: \*
- Regrouper: ( )
- Ou logique: |

. \* (ALAIN|Alain|alain) . \*

DNA\s(A|C|G|T)+

DNA ACCCATGTGCA

ACG

Alain

# Nettoyage: transformation de donnée

La transformation de donnée est une opération qui affecte plusieurs champs (collones).

Par exemple, la décomposition d'une date en plusieurs champs [année, mois et jour] ou encore la transformation d'une entrée de type booléen en entier (`false=0, true = 1`).

# Nettoyage: renforcement des contraintes d'intégrité

Les contraintes d'intégrité sont au départ utilisées pour prévenir des erreurs sur les données, cependant il se peut qu'elles ne suffisent pas et que des erreurs sur les données se manifestent au fil du temps. Dans certains cas, le meilleur moyen d'améliorer la qualité des données n'est pas de corriger les données corrompues, mais plutôt de modifier les contraintes d'intégrité car la sémantique des données ou de l'application peut avoir évolué.

Le but de cette méthode n'est pas de modifier les données de manière directe mais de trouver et modifier les contraintes d'intégrité douteuses afin qu'elles s'accordent mieux avec les données.

# Nettoyage: méthodes statistique

Malgré les recherches faites sur les contraintes d'intégrité ainsi que sur d'autres méthodes visant à améliorer la qualité des données, les bases de données peuvent encore contenir un certain nombre d'erreurs subtiles, syntaxiques ou sémantiques, qu'il est difficile voire impossible d'exprimer (et détecter) en utilisant les contraintes générales offertes dans les SGBD actuels.

L'utilisation des statistiques permet d'obtenir des corrections d'erreurs plus fiables. En identifiant de potentielles dépendances statistiques entre les paires de données similaires et en développant des algorithmes que l'on peut greffer simplement dans les SGBD standards, on peut estimer automatiquement ces dépendances. Cette méthode permet par exemple de déduire des valeurs correctes même en présence de valeurs manquantes ou corrompues.

Les méthodes statistiques peuvent être utilisées pour l'analyse des données et/ou leur correction.

# Nettoyage: méthodes statistique

Apprentissage automatique

- Supervisé ou non supervisé
- Statistique de base
- Donnée aberrante (*outliers*)
- Détection d'anomalie

# Nettoyage: *Crowdsourcing*

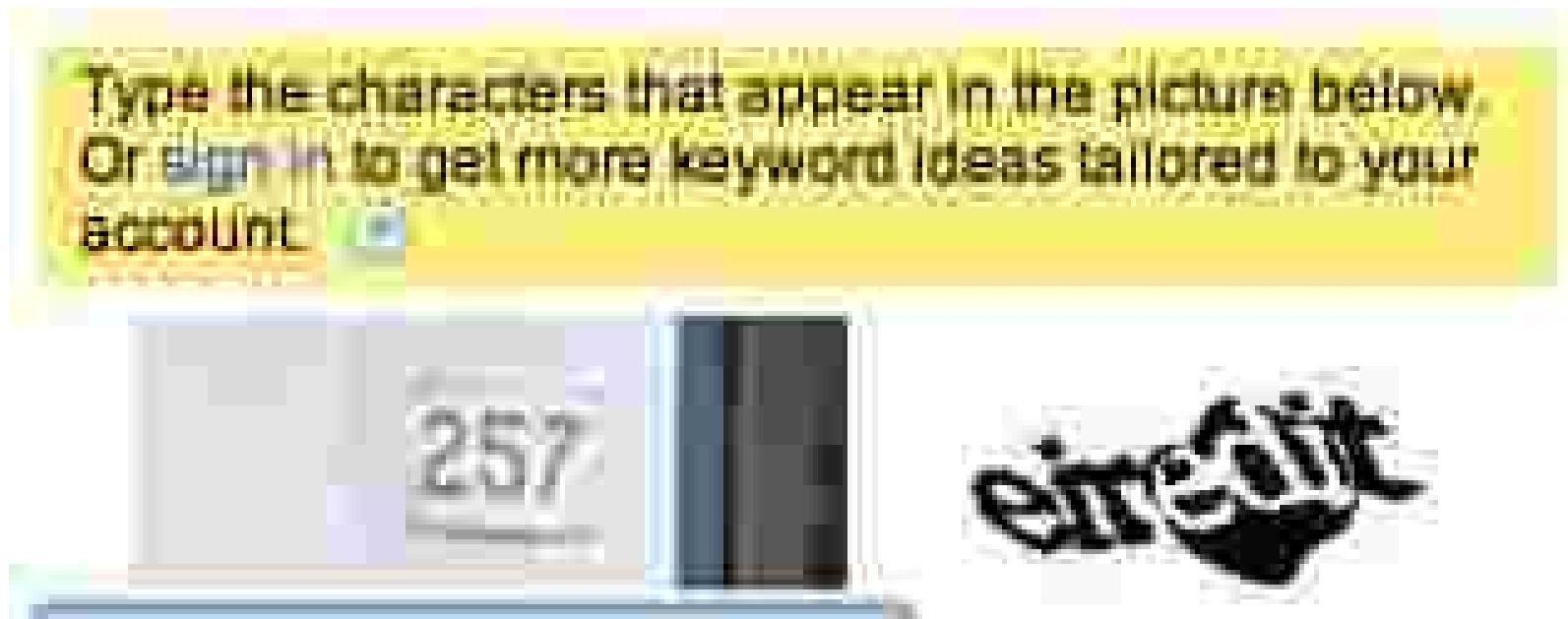
Il peut arriver que les méthodes précédentes n'aient pas assez d'éléments de preuve pour pouvoir identifier et corriger les erreurs.

Une approche basée sur le *crowdsourcing* permet d'utiliser des bases de connaissance externes qui permettent d'identifier plus d'erreurs automatiquement.

Les erreurs détectées qui ne peuvent être réparées automatiquement sont proposées à un humain qui les corrige manuellement.

# CAPTCHA

Completely Automated Public Turing test to tell Computers and Humans Apart



# **Basse de donné (relationnel)**

# Bases de données relationnelles

Une base de données relationnelle stocke des informations dans des tableaux. Chaque thème d'information est stocké dans sa propre table. Les éléments seront associés de façon appropriée à l'aide de pointeurs.

Une **entité** est quelque chose d'important pour un utilisateur devant être représenté dans une base de données.

Une entité représente un thème ou un sujet et est représentée par un **tableau**.

Les dimensions du tableau, comme une matrice, se composent de lignes (**tuples**) et de colonnes (**attributs**).

Une **clé primaire** est une (ou plusieurs) colonne d'une table utilisée pour identifier une ligne.

Une **clé étrangère** est une clé primaire d'une table placée dans une autre table (colonne). Elle est à toute fin pratique un pointeurs explicite.

File Home Insert Page Layout Formulas Data Review View

Font Size: 11 A A = = = = Date Date Conditional Formatting + or Value + Sort & Filter +

Font Color: Black Red Green Blue Yellow Orange

Font Style: Normal Bold Italic Underline

Font Weight: Normal Medium Strong

Font Size: 10pt 12pt 14pt 16pt 18pt 20pt 22pt 24pt 26pt 28pt 30pt 32pt 34pt 36pt 38pt 40pt

Font Family: Arial Calibri Cambria Times New Roman

Font Alignment: Left Center Right

Font Number: 1000 2000 3000 4000 5000 6000 7000 8000 9000 10000

Font Color: Black Red Green Blue Yellow Orange

Font Style: Normal Bold Italic Underline

Font Weight: Normal Medium Strong

Font Size: 10pt 12pt 14pt 16pt 18pt 20pt 22pt 24pt 26pt 28pt 30pt 32pt 34pt 36pt 38pt 40pt

Font Family: Arial Calibri Cambria Times New Roman

Font Alignment: Left Center Right

Font Number: 1000 2000 3000 4000 5000 6000 7000 8000 9000 10000

A2      B C D E F G H I J K L M N O

Date Type Symbol Shares Price Costs Fees Total DistribShares AtTCurrency Exchange CashAffectName Comment

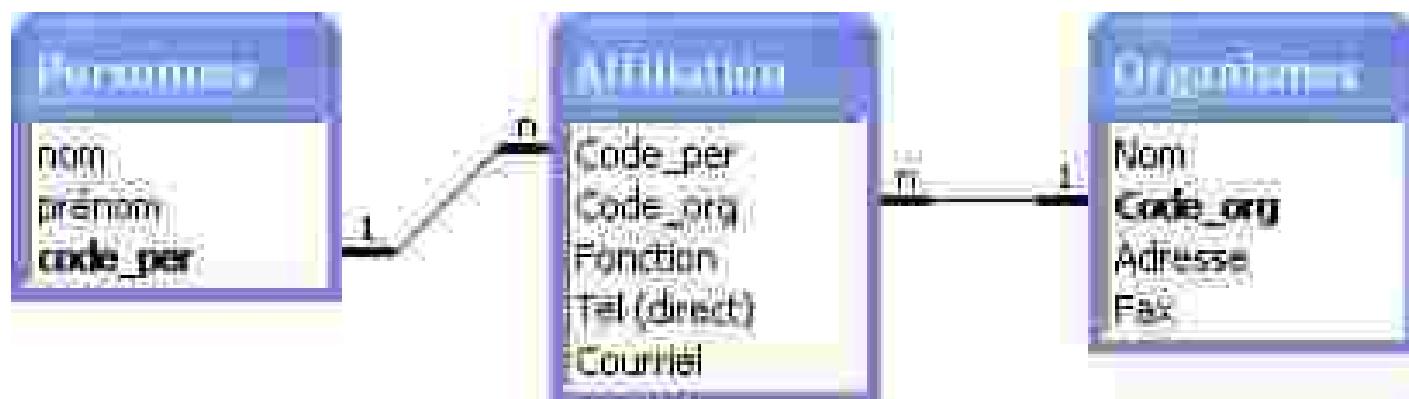
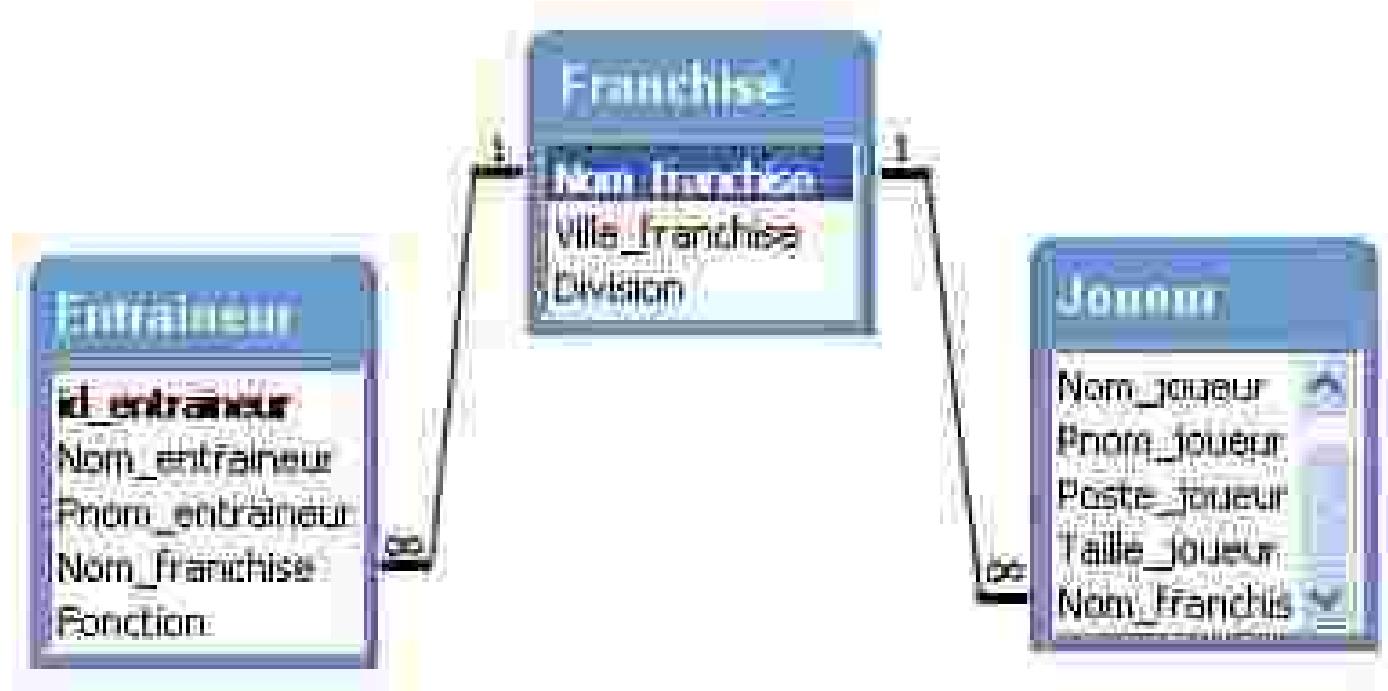
7/3/2007	CASH\_IN					0	50000		USD		TRUE	
4/11/2011 BUY	LUKX	100	89.41	10	-8951			USD			TRUE	Lufkin Industries
6/23/2011 BUY	LUKX	200	65.43	10	-13092			USD			TRUE	Lufkin Industries
9/2/2011 BUY	LUKX	150	58.83	10	-8825.5			USD			TRUE	Lufkin Industries
1/14/2012 SELL	LUKX	250	78.41	10	27831.5			USD			TRUE	Lufkin Industries
7/11/2011 BUY	FTEK	200	7.19	10	-1448			USD			TRUE	FuelTech Inc.
8/12/2011 BUY	FTEK	200	5.4	10	-1080			USD			TRUE	FuelTech Inc.
9/2/2011 BUY	FTEK	150	5.85	10	-877.5			USD			TRUE	FuelTech Inc.
5/6/2012 SELL	FTEK	500	4.65	10	1594			USD			TRUE	FuelTech Inc.
6/16/2011 BUY	GS	50	136.05	10	6814.5			USD			TRUE	Goldmann Sachs
10/3/2011 BUY	GS	50	90.08	10	-4534			USD			TRUE	Goldmann Sachs
3/26/2012 SELL	GS	25	129.07	10	3192.5			USD			TRUE	Goldmann Sachs
5/12/2010 BUY	JNJ	100	58.45	10	-5855			USD			TRUE	Johnson & Johnson

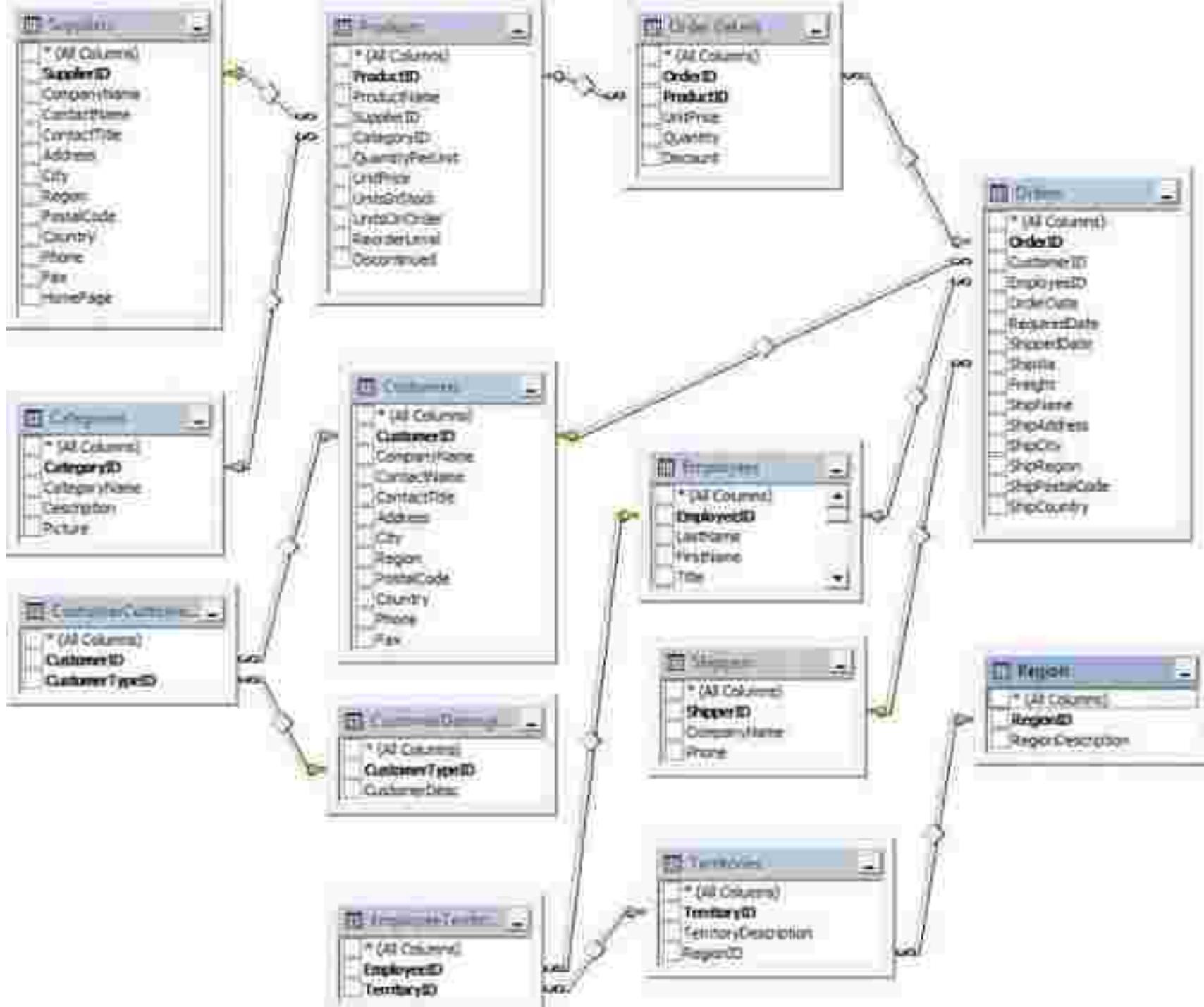
Table "album"

ID	titre	numero	titre
1	Thorgal	4	La galère noire
2	Blueberry	1	Fort Navajo
3	Lanfeust de Troy	1	L'voice du Magonamoth
4	Aldebaran	1	La catastrophe
5	Astérix	5	Le tour de gaule
6	Gaston Lagaffe	12	Le gang des gaffeurs
7	Les maîtres de l'orge	1	Charles, 1854
8	Tintin	2	Tintin en Amérique
9	Thorgal	1	Le magicienne trahie
10	Thorgal	2	L'île des mers gelées
11	Thorgal	3	Les 3 vieillards du pays d'Asan
12	Blueberry	2	Torneresse à l'ouest
13	XIII	1	Le jour du soleil noir
14	Aldebaran	2	La blonde
15	Astérix	8	Astérix chez les Bretons
16	Les maîtres de l'orge	2	Adrien, 1886
17	Thorgal	5	Au-delà des hommes
18	Tintin	15	Objectif lune
19	Tintin	16	On a marché sur la lune
20	Thorgal	6	La chute de Brèk Zarith
22	Aldebaran	3	Le groupe
23	Aldebaran	4	La créature
24	Iznogoud	5	Des astres pour Iznogoud
25	Lucky Luke	3	Dalton City
23	Gaston Lagaffe	2	Le bureau des gaffes en gros
26	Lucky Luke	11	Le cavalier blanc
27	Astérix	2	La serpe d'or
28	Astérix	7	Le combat des Chefs
29	Astérix	1	Astérix le Gaulois

Table "collection"

ID	titre	éditeur	illustrateur	auteur
1	Astérix	Dargaud	Uderzo	Goscinny
2	Gaston Lagaffe	Dupuis	Franquin	Franquin
3	Thorgal	Le Lombard	Réginald	Van Hamme
4	Blueberry	Dargaud	Giraud	Charlier
5	Lanfeust de Troy	Soleil	Tanguy	Ariston
6	Les maîtres de l'orge	Glénat	Vallée	Van Hamme
7	Tintin	Casterman	Hergé	Hergé
8	Aldebaran	Dargaud	Léo	Léo
9	Iznogoud	Dargaud	Tibary	Goscinny
10	Lucky Luke	Dargaud	Morris	Goscinny
11	XIII	Dargaud	Vance	Van Hamme





# Relations au sein de la base de données relationnelle

## 1 : M

**Idéal** de modélisation relationnelle. Devrait être la norme dans toute conception de base de données relationnelle.

## 1 : 1

Devrait être **rare** dans toute conception de base de données relationnelle.

## M : N

**Ne peut pas être implémenté** en tant que tel dans le modèle relationnel. (Les relations M: N peuvent être changées en deux relations 1: M).

**FIGURE  
3.18**

The 1:M relationship between  
**PAINTER** and **PAINTING**



**FIGURE  
3.19**

The implemented 1:M relationship between PAINTER and PAINTING

Table name: PAINTER

Primary key: PAINTER\_NUM

Foreign key: none

Database name: Ch03\_Museum

	PAINTER_NUM	PAINTER_LNAME	PAINTER_FNAME	PAINTER_INITIAL
▶	123	Ross	Gebregette	R
•	126	Bero	Jalo	G

Table name: PAINTING

Primary key: PAINTING\_NUM

Foreign key: PAINTER\_NUM

	PAINTING_NUM	PAINTING_TITLE	PAINTER_NUM
▶	1336	Dawn Thunder	123
	1339	Vanilla Roses To Nowhere	123
	1340	Tired Flounders	126
	1341	Hasty Exit	123
	1342	Plastic Paradise	126

**FIGURE  
3.20**

**The 1:M relationship between  
COURSE and CLASS**

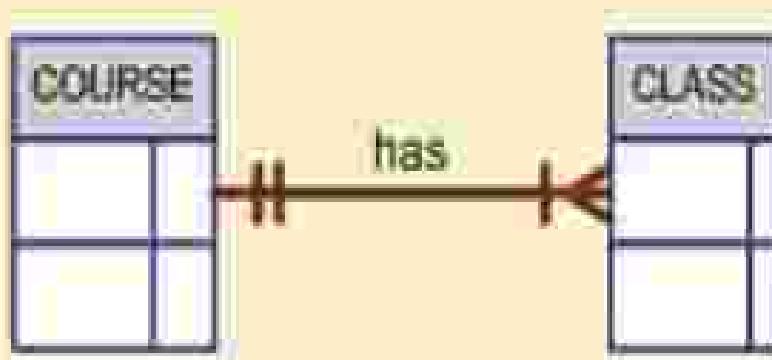


FIGURE  
3.21

## The implemented 1:M relationship between COURSE and CLASS

Table name: COURSE

Primary key: CRS\_CODE

Foreign key: none

Database name: Ch03\_TinyCollege

	CRS_CODE	DEPT_CODE	CRS_DESCRIPTION	CRS_CREDIT
▶	ACCT-211	ACCT	Accounting I	3
▶	ACCT-212	ACCT	Accounting II	3
▶	CIS-220	CIS	Intro. to Microcomputing	3
▶	CIS-420	CIS	Database Design and Implementation	4
▶	QM-261	QM	Intro. to Statistics	3
▶	QM-362	QM	Statistical Applications	4

Table name: CLASS

Primary key: CLASS\_CODE

Foreign key: CRS\_CODE

	CLASS_CODE	CRS_CODE	CLASS_SECTION	CLASS_TIME	CLASS_ROOM	PROF_NUM
▶	10012	ACCT-211	1	MWF 8:00-8:50 a.m.	BUS311	105
▶	10013	ACCT-211	2	MWF 9:00-9:50 a.m.	BUS200	105
▶	10014	ACCT-211	3	TTh 2:30-3:45 p.m.	BUS252	342
▶	10015	ACCT-212	1	MWF 10:00-10:50 a.m.	BUS311	301
▶	10016	ACCT-212	2	Tu 8:00-8:40 p.m.	BUE252	301
▶	10017	CIS-220	1	MWF 9:00-9:50 a.m.	KLR209	228
▶	10018	CIS-220	2	MWF 9:00-9:50 a.m.	KLR211	114
▶	10019	CIS-220	3	MWF 10:00-10:50 a.m.	KLR209	228
▶	10020	CIS-420	1	W 8:00-8:40 p.m.	KLR208	162
▶	10021	QM-261	1	MWF 8:00-8:50 a.m.	KLR200	114
▶	10022	QM-261	2	TTh 1:00-2:15 p.m.	KLR200	114
▶	10023	QM-362	1	MWF 11:00-11:50 a.m.	KLR200	162
▶	10024	QM-362	2	TTh 2:30-3:45 p.m.	KLR200	162

# *Structured Query Language (SQL)*

# La syntaxe ordinaire de SQL

```
SELECT [DISTINCT] expr1 [[AS] nom1], expr2 [[AS] nom2]...
FROM table1 [[AS] alias1], table2 [[AS] alias2]...
WHERE prédicat
GROUP BY expr1, expr2...
ORDER BY expr1 [ASC | DESC], expr2 [ASC | DESC]...
LIMIT nombre [OFFSET décalage]
```

nom

expr = Expression

table

alias

prédicat

# Expressions

Les expressions valides mettent en jeu des noms de champs, le mot clé « \* » (qui signifie « tous les champs »), des constantes, des fonctions et des opérateurs classiques. Il existe des fonctions logiques, mathématiques, de manipulation de chaînes, de dates et des fonctions d'agrégation.

Les fonctions d'agrégation permettent de calculer un résultat numérique (un entier ou un flottant) à partir d'un ensemble de valeurs. Les fonctions d'agrégation sont au nombre de 5 : COUNT, SUM, MIN, MAX, AVG. Les quatre dernières s'appliquent à un champ ; par exemple AVG(nom) renvoie la moyenne des valeurs contenues dans le champ nom.

# Quelques précisions sur COUNT

COUNT(\*) compte le nombre total d'enregistrements dans une table, après restriction éventuelle par un WHERE et groupe par groupe en présence d'une clause GROUP BY.

COUNT(nom) compte le nombre d'enregistrements pour lesquels le champ nom contient une valeur autre que NULL.

COUNT(DISTINCT nom) compte le nombre de valeurs distinctes autres que NULL présentes dans le champ nom.

# Prédicats

## Prédicats simples

Un prédicat simple est la comparaison de plusieurs expressions au moyen d'un opérateur logique :

WHERE expr1 = | != | < | > | <= | >= expr2

WHERE expr1 [NOT] LIKE expr2

WHERE expr1 IS [NOT] NULL

WHERE expr1 [NOT] IN (expr2, expr3...)

WHERE [NOT] EXISTS (expr1)

## Prédicats composés

Les opérateurs logiques AND et OR permettent de combiner plusieurs prédicats.

# Jointure vs produit cartésien

Avec une base de données complexe, on a souvent besoin de rassembler des données réparties dans plusieurs tables, typiquement concaténer les enregistrements provenant de deux tables, table1 et table2, et vérifiant table1.champ1 = table2.champ2 (l'un des deux champ1 ou champ2 est souvent une clé primaire dans sa table). Pour ce faire, une solution (coûteuse) consiste à former le produit cartésien des deux tables, puis à sélectionner parmi le (trop) grand nombre d'enregistrements obtenus :

```
SELECT expr1, expr2... FROM table1, table2 WHERE table1.champ1 = table2.champ2
```

Il faut préférer à ce type de requête l'utilisation d'une jointure (partie du produit cartésien formée des enregistrements vérifiant la condition). Une telle jointure est créée par les logiciels de gestion de bases de données de façon plus efficace que la sélection dans le produit cartésien. La syntaxe est la suivante :

```
SELECT expr1, expr2... FROM table1 JOIN table2 ON table1.champ1 = table2.champ2
```

# *Structured Query Language (SQL) References*

TABLE  
7.2

## SQL Data Manipulation Commands

COMMAND	FUNCTION
INSERT	Inserts row(s) into a table
SELECT	Selects attributes from rows in one or more tables or views
WHERE	Restricts the selection of rows based on a conditional expression
GROUP BY	Groups the selected rows based on one or more attributes
HAVING	Restricts the selection of grouped rows based on a condition
ORDER BY	Orders the selected rows based on one or more attributes
UPDATE	Modifies an attribute's values in one or more table's rows
DELETE	Deletes one or more rows from a table
COMMIT	Permanently saves data changes
ROLLBACK	Restores data to their original values

TABLE  
7.2

## SQL Data Manipulation Commands (continued)

COMMAND OR OPERATOR	DESCRIPTION
<b>COMPARISON OPERATORS</b>	
=, <, >, <=, >=, <>	Used in conditional expressions
<b>LOGICAL OPERATORS</b>	
AND/OR/NOT	Used in conditional expressions
<b>SPRINGER OPERATORS</b>	
BETWEEN	Checks whether an attribute value is within a range
IS NULL	Checks whether an attribute value is null
LIKE	Checks whether an attribute value matches a given string pattern
IN	Checks whether an attribute value matches any value within a value list
EXISTS	Checks whether a subquery returns any rows
DISTINCT	Limits values to unique values
<b>MEDICAL FUNCTIONS</b>	Used with SELECT to return mathematical summaries on columns
COUNT	Returns the number of rows with non-null values for a given column
MIN	Returns the minimum attribute value found in a given column
MAX	Returns the maximum attribute value found in a given column
SUM	Returns the sum of all values for a given column
AVG	Returns the average of all values for a given column

TABLE  
7.4

## Some Common SQL Data Types

DATA TYPE	FORMAT	COMMENTS
Numeric	NUMBER(L,D)	The declaration NUMBER(7,2) indicates numbers that will be stored with two decimal places and may be up to six digits long, including the sign and the decimal place. Examples: 12.32, -134.99.
	INTEGER	May be abbreviated as INT. Integers are (whole) counting numbers, so they cannot be used if you want to store numbers that require decimal places.
	SMALLINT	Like INTEGER, but limited to integer values up to six digits. If your integer values are relatively small, use SMALLINT instead of INT.
	DECIMAL(L,D)	Like the NUMBER specification, but the storage length is a <i>minimum</i> specification. That is, greater lengths are acceptable, but smaller ones are not. DECIMAL(9,2), DECIMAL(9), and DECIMAL are all acceptable.
Character	CHAR(L)	Fixed-length character data for up to 255 characters. If you store strings that are not as long as the CHAR parameter value, the remaining spaces are left unused. Therefore, if you specify CHAR(25), strings such as "Smith" and "Katzenjammer" are each stored as 25 characters. However, a U.S. area code is always three digits long, so CHAR(3) would be appropriate if you wanted to store such codes.
	VARCHAR(L) or VARCHAR2(L)	Variable-length character data. The designation VARCHAR2(25) will let you store characters up to 25 characters long. However, VARCHAR will not leave unused spaces. Oracle users may use VARCHAR2 as well as VARCHAR.
Date	DATE	Stores dates in the Julian date format.

TABLE  
7.6

## Comparison Operators

SYMBOL	DESCRIPTION
=	Equal to
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
<> or !=	Not equal to

TABLE  
7.7

## The Arithmetic Operators

SYMBOL	DESCRIPTION
+	Add
-	Subtract
*	Multiply
/	Divide
<sup>n</sup>	Raise to the power of <i>n</i> . (Some applications use $^{**}$ instead of $^n$ .)

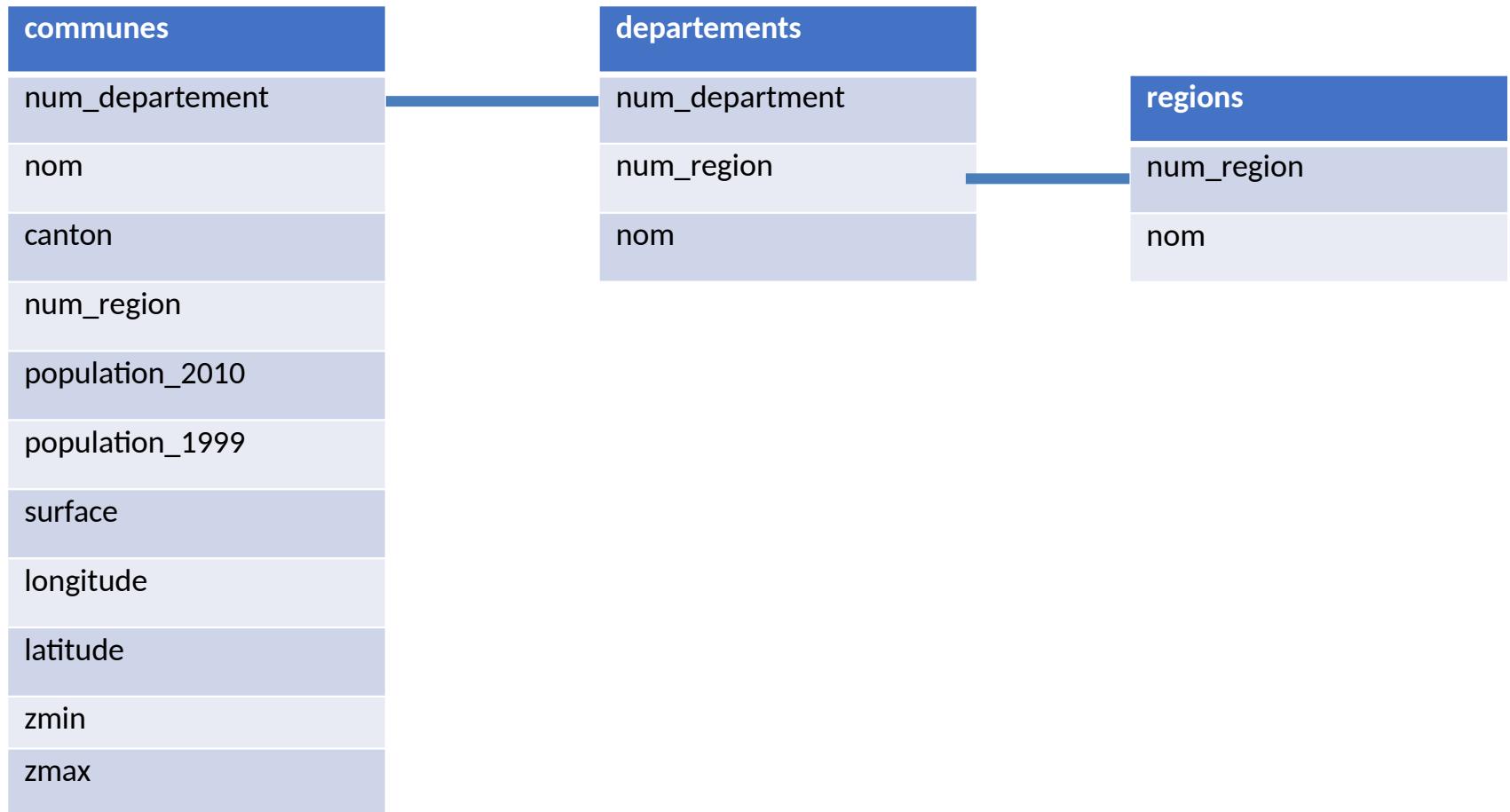
TABLE  
7.8

## Some Basic SQL Aggregate Functions

FUNCTION	OPTIONAL
COUNT	The number of rows containing non-null values
MIN	The minimum attribute value encountered in a given column
MAX	The maximum attribute value encountered in a given column
SUM	The sum of all values for a given column
AVG	The arithmetic mean (average) for a specified column

# **Exemple de BDR SQL**

# Exemple de BD



# Requêtes sans jointure

1) Combien d'habitants en France (en supposant que chacun n'a été compté qu'une fois dans les différentes communes... ) ?

```
SELECT SUM(population_2010) FROM communes;
```

2) Combien de communes en Loire-Atlantique ?

```
SELECT COUNT(*) FROM communes WHERE num_departement=44;
```

3) Quelles sont les dix plus petites communes de France, en superficie ?

```
SELECT nom, num_departement, surface FROM communes ORDER BY surface LIMIT 10;
```

4) Quelles sont les dix communes de France les plus peuplées ?

```
SELECT nom, num_departement, population_2010  
FROM communes ORDER BY population_2010 DESC LIMIT 10;
```

5) Quelles sont les douze communes de Loire-Atlantique les plus densément peuplées ?

```
SELECT nom, population_2010/surface AS densité  
FROM communes WHERE num_departement=44 ORDER BY densité DESC LIMIT 12;
```

6) Donner la liste des numéros de département, avec pour chaque numéro le nombre de communes du département. Afficher un titre explicite pour la colonne des nombres de communes.

```
SELECT num_departement, COUNT(*) AS nb_communes  
FROM communes GROUP BY num_departement;
```

7) Donner la liste des numéros de département, avec pour chaque numéro la population totale du département. Trier par population totale décroissante et limiter la liste aux départements ayant plus d'un million d'habitants.

```
SELECT num_departement, SUM(population_2010) AS population FROM communes  
GROUP BY num_departement HAVING population>1000000 ORDER BY population DESC;
```

8) Donner les communes ayant les six voyelles dans leur nom. Noter que SQL ne distingue pas majuscules/minuscules. Lister ces noms, triés par longueur croissante (la fonction LENGTH donne la longueur d'une chaîne de caractères). Y en a-t-il en Loire-Atlantique ? Dans le Maine-et-Loire ?

```
SELECT nom FROM communes WHERE nom LIKE '%a%' AND nom LIKE '%e%'  
AND nom LIKE '%i%' AND nom LIKE '%o%' AND nom LIKE '%u%' AND nom LIKE '%y%'  
ORDER BY LENGTH(nom);
```

Pour voir par exemple s'il y en a en Loire-Atlantique, ajouter AND num\_departement=44

9) Quelles sont les quinze communes pour lesquelles l'écart entre les altitudes maximale et minimale est le plus grand ?

```
SELECT nom, num_departement, zmax-zmin as écart FROM communes  
ORDER BY écart DESC LIMIT 15;
```

10) Quelles sont les dix communes de Loire-Atlantique où la population a le plus augmenté en valeurs absolues, entre 1999 et 2010 ?

```
SELECT nom, population_2010-population_1999 AS var_abs  
FROM communes WHERE num_departement=44 ORDER BY var_abs DESC LIMIT 10;
```

11) Même question en valeurs relatives. Attention ! Pour SQL, un quotient d'entiers est le quotient (entier !) de la division euclidienne. Pour forcer un calcul en flottants, on peut multiplier par 1.0 !

```
SELECT nom, 1.*population_2010/population_1999-1 AS var_rel  
FROM communes WHERE num_departement=44 ORDER BY var_rel DESC LIMIT 10;
```

# Requêtes avec jointures

- 1) Donner la liste des noms des départements des régions Bretagne et Pays de la Loire.

```
SELECT departements.nom, regions.nom  
FROM departements JOIN regions  
ON departements.num_region = regions.num_region AND  
(regions.nom = 'Bretagne' OR regions.nom = 'Pays de la Loire');
```

- 2) Donner la liste des noms de départements, avec pour chaque département le nombre de communes. Ordonner par population décroissante.

```
SELECT departements.nom, departements.num_departement, COUNT(*) as  
nb_communess  
SUM(population_2010) AS population FROM communes  
JOIN departements ON communes.num_departement =  
departements.num_department  
GROUP BY communes.num_departement ORDER BY population DESC;
```

# Requêtes avec jointures (suite)

- 3) Donner la liste des noms des régions avec la densité de population de chaque région.

```
SELECT regions.nom, SUM(population_2010)/SUM(surface) AS densité
FROM communes
JOIN departements ON communes.num_departement = departements.num_department
JOIN regions ON departements.num_region = regions.num_region
GROUP BY regions.num_region ORDER BY densité DESC;
```

- 4) Donner sans doublons la liste des noms des départements contenant une commune dont le nom commence par “Petit”.

```
SELECT DISTINCT departements.nom
FROM communes JOIN departements
ON communes.num_departement = departements.num_department
WHERE communes.nom LIKE 'petit%';
```