Comprendre : On organise ses données en base de données relationnelle lorsque la structure "plate" d'un tableau traditionnel ne suffit pas à faire des recherches d'informations de manière efficace. En effet, la présentation des données en tableau est intimement liée à un "point de vue", un type de requête particulier.

Par exemple, quel(s) défaut(s) voyez-vous à la table ci-dessous ?

nom	prenom	moyenne	maison	referent	dortoir	effectif
Granger	Hermione	19	Gryffondor	MacGonagall	tour	28
Diggory	Cedric	17	Poufsouffle	Chourave	logis	31
Malefoy	Drago	13	Serpentard	Rogue	donjon	25
Potter	Harry	17	Gryffondor	MacGonagall	tour	28
Weasley	Ron	16	Gryffondor	MacGonagall	tour	28

Conclusion : Il faut séparer les entités.

I Structures des bases de données

1 Vocabulaire

Le principe : une base de données est un ensemble de "tables" appelés *relations* ayant des liens entre eux.
Chaque table est composée de "lignes" appelées *enregistrements* correspondant à des n-uplets de même structure. On appelle cette structure le *schéma relationnel* de la table.

Exemples :

relation maisons

nom	referent	dortoir	effectif
Gryffondor	MacGonagall	tour	28
Serdaigle	Flitwick	tour	35
Poufsouffle	Chourave	logis	31
Serpentard	Rogue	donjon	25

relation eleves

nom	prenom	moyenne	maison
Granger	Hermione	19	Gryffondor
Diggory	Cedric	17	Poufsouffle
Malefoy	Drago	13	Serpentard
Potter	Harry	17	Gryffondor
Weasley	Ron	16	Gryffondor

Le schéma relationnel est composé de quatre "variables" appelées *attributs*

- nom, de domaine { Gryffondor, Serdaigle, Pourfsouffle, Serpentard } ou chaines de caractères
- ...
- dortoir, de domaine {tour, logis, donjon} ou chaines de caractères
- effectif de domaine $\mathbb{N}$

Cette relation est associée au schéma relationnel (nom, prenom, moyenne, maison)

En théorie, chaque enregistrement (ou valeur) doit être un n-uplet différent.

relation : table
attribut : variable, champs (field)
domaine : type des attributs, valeurs possibles
schéma relationnel : structure (liste des attributs)
enregistrement : une "ligne" de la table

2 Notion de clé primaire

Puisque, en théorie, chaque enregistrement/ligne est différent, il est possible à partir des valeurs des attributs d'identifier chaque enregistrement. Peut être a-t-on besoin de connaître les valeurs de tous les attributs, peut être qu'un seul attribut, ou deux suffisent...

Exemple : Dans la relation maisons, le nom suffit à distinguer chaque ligne. Mais le dortoir lui ne permet pas de distinguer les enregistrements de Gryffondor et Serdaigle.

Une **clé** est un ensemble d'attributs sur lequel la relation est « injective ». Tous les enregistrements sont distincts sur ces attributs. Il est très pratique (mais pas obligatoire) d'avoir un unique attribut comme clé. Tous les enregistrements prennent une valeur différente pour cet attribut, il peut donc servir à lui tout seul à les « identifier ». Parfois, il en faut plusieurs. Une clé est dite **primaire** lorsqu'elle utilise le nombre minimum d'attributs possible.
On souligne le/les attributs choisi comme clé primaire.

On choisit généralement quel(s) attribut(s) servira(ont) de clé primaire avant de créer la table (lors de la conception de la base).

Exemple : Donner une clé primaire pour chacune des relations.

Parfois un attribut n'est pas une clé dans la relation observée mais l'est dans une autre relation : on parle alors de **clé étrangère** (ce sera particulièrement utile pour la suite).

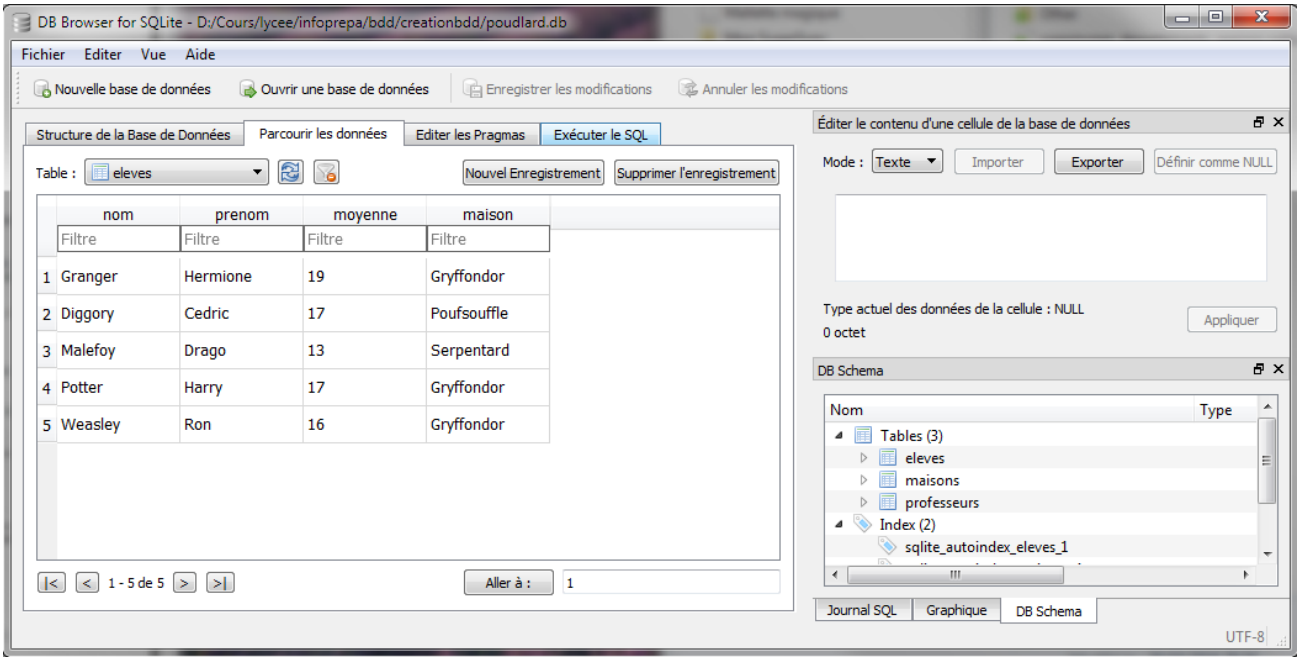
Exemple : nom est une clé primaire dans la relation maison, cet attribut est une clé étrangère dans eleves sous l'appellation maison.

Exercice

3 Créer une base de données

La conception des bases de données est hors programme.

Une fois "pensée", il faut la créer sur machine. La base de données une fois crée est juste un fichier, d'extension généralement `.db` Pour les manipuler, on utilise généralement un SGBD : Système de Gestion de Base de données. Les plus connus sont Oracle Data Base, Mysql ou SQLlite. Nous utiliserons *SQLiteDatabaseBrowserPortable* .
Ces logiciels permettent d'interagir directement avec la base de données, ou peuvent être interrogés par une autre application comme Python.



Le programme n'insiste pas sur la création de base de données mais plutôt sur leur manipulation en langage SQL. SQL est l'abréviation de Structured Query Language. C'est le langage commun à tous les logiciels de gestion de base de données.

Voir le TP1 : Création de bases de données

Par la suite (y compris en deuxième année), nous utiliserons des bases déjà créées. Néanmoins, voilà des exemples dont il serait bon de se souvenir :

Pour créer une table

```
CREATE   TABLE IF NOT EXISTS eleves (
             nom TEXT, prenom TEXT, moyenne INTEGER, maison TEXT,
             PRIMARY KEY(nom)    ) ;
```

Pour rajouter un enregistrement/ligne

```
INSERT INTO eleves (nom,prenom,moyenne,maison) VALUES ('Lovegood','Luna',15,'Serdaigle') ;
```

Pour modifier un enregistrement/ligne

```
UPDATE eleves SET moyenne=20 WHERE nom='Granger' AND prenom='Hermione' ;
```

II Opérations sur une table

Les deux opérations au programme sont

— **projection :** pour une table on décide de ne sélectionner que certains attributs. Cela se note π en algèbre relationnelle. On obtient une nouvelle table temporaire qui n'a plus le même schéma.

— **sélection :** on ne sélectionne dans une table que les enregistrements dont certains attributs correspondent à des valeurs choisies. Cela se note σ en algèbre relationnelle. On obtient une nouvelle table temporaire qui a le même schéma.

Toutes ces opérations peuvent se combiner entre elles, dans un ordre si possible optimal pour effectuer des recherches.

relation maisons

nom	referent	dortoir	effectif
Gryffondor	MacGonagall	tour	28
Serdaigle	Flitwick	tour	35
Poufsouffle	Chourave	logis	31
Serpentard	Rogue	donjon	25

Exemples : Donner les résultats de

a) $\pi_{referent,effectif}(maisons)$

b) $\sigma_{dortoir='tour'}(maisons)$

c) $\pi_{dortoir}(maisons)$

Et comment ça s'écrit en SQL ? Toutes les requêtes (sélection et projection) commencent par SELECT et se terminent par un point-virgule.

Pour effectuer la **projection** :

```
SELECT A1, .. An FROM R ;      # on sélectionne les attributs A1,...An dans la table R
SELECT * FROM R ;      # on sélectionne tous les attributs dans la table R
```

Attention, en faisant une projection, des enregistrements qui ne différaient que par leurs valeurs pour des attributs désormais non sélectionnés deviennent identiques. Dans la théorie des bases de données, ils sont « instantanément fusionnés » mais dans l'implémentation en SQL c'est à nous de préciser si on veut qu'ils soient fusionnés ou non grâce au mot clé **DISTINCT**.

Exemples :

relation maisons

nom	referent	dortoir	effectif
Gryffondor	MacGonagall	tour	28
Serdaigle	Flitwick	tour	35
Poufsouffle	Chourave	logis	31
Serpentard	Rogue	donjon	25

SELECT dortoir
FROM maisons ;

dortoir
tour
tour
logis
donjon

SELECT DISTINCT dortoir
FROM maisons ;

dortoir
tour
logis
donjon

pour renommer un attribut (ou une table), on effectue une projection en indiquant tous les attributs et en précisant avec le mot clef **AS** le nouveau nom.

Exemple : SELECT nom , referent AS prof_principal , effectif FROM eleves ;

Pour effectuer une **sélection** :

```
SELECT * FROM R WHERE A=a ;      # on sélectionne tous les enregistrements de la table R pour lesquels
l'attribue A prend la valeur a.
```

On peut effectuer des sélections composées en utilisant les opérateurs booléens OR, AND...

Exemple :

```
SELECT nom,prenom
FROM eleves
WHERE maison='Gryffondor' AND moyenne>=17 ;
```

On peut utiliser des sous requêtes et les opérateurs IN ou NOT IN

Exemple :

```
SELECT referent
FROM maisons
WHERE nom IN(   SELECT DISTINCT maison
                FROM eleves
                WHERE moyenne>=16) ;
```

III Fonctions d'agrégation

Les fonctions d'agrégation dans le langage SQL permettent d'effectuer des opérations statistiques sur un ensemble d'enregistrements.

Les fonctions d'agrégation sont des fonctions idéales pour effectuer quelques statistiques de bases sur des tables. Les principales fonctions sont les suivantes :

- **AVG()** pour calculer la moyenne sur un ensemble d'enregistrements
- **COUNT()** pour compter le nombre d'enregistrements sur une table ou une colonne distincte
- **MAX()** pour récupérer la valeur maximum d'une colonne sur un ensemble de ligne. Cela s'applique à la fois pour des données numériques ou alphanumérique
- **MIN()** pour récupérer la valeur minimum de la même manière que MAX()
- **SUM()** pour calculer la somme sur un ensemble d'enregistrement

Exemple : :
```
SELECT AVG(Moyenne)
FROM eleves;
```

relation eleves

nom	prenom	moyenne	maison
Granger	Hermione	19	Gryffondor
Diggory	Cedric	17	Poufsouffle
Malefoy	Drago	13	Serpentard
Potter	Harry	17	Gryffondor
Weasley	Ron	16	Gryffondor

Toutes ces fonctions prennent tout leur sens lorsqu'elles sont utilisée avec la commande **GROUP BY** qui permet de filtrer les données sur une ou plusieurs colonnes.

Exemple :
```
SELECT maison, AVG(moyenne)
FROM eleves
GROUP BY maison;
```

Attention, le résultat d'une fonction d'agrégat est le résultat d'une requête. Il ne peut être utilisé en cours de requête.

Exemple :
```
SELECT nom
FROM eleves
WHERE moyenne<AVG(moyenne);
# ne fonctionne pas !
```

```
SELECT nom
FROM eleves
WHERE moyenne<(  SELECT AVG(moyenne)
                 FROM eleves);
```

Exo 1 : On suppose qu'on dispose d'une table *Joueurs* (aperçu ci-dessous) qui donne le salaire annuel (en millions d'euros) et l'échéance du contrat des joueurs de football du PSG pour la saison 2020-2021 (source Sportune).

a) Donnez le schéma relationnel de la table, préciser une clé primaire.

b) Que signifie $\sigma(Salaire > 10)(Joueurs)$.
Pour la suite, donner en langage SQL les requêtes permettant d'obtenir

c) La même table, mais ordonnée du salaire le plus élevé au salaire le moins élevé.

d) Le salaire de Kylian Mbappé.

e) Les noms et prénoms des joueurs finissant leur contrat en 2021.

f) Le salaire moyen des joueurs.

g) Le nom du joueur ayant le plus gros salaire.

h) Le nombre de joueurs selon la date d'échéance de leur contrat.

i) Le nombre des joueurs gagnant moins que la moyenne.

Prenom	Nom	Salaire	Echeance
Mitchel	Bakker	1,2	2023
Juan	Bernat	3,6	2021
Colin	Dagba		2024
Ángel	Di María	13,4	2021
Abdou	Diallo	5,4	2024
Julian	Draxler	7,2	2021
Alessandro	Florenzi	3,9	2021
Idrissa	Gueye	6	2023
Ander	Herrera	7,9	2024
Mauro	Icardi	9,6	2022
Thilo	Kehrer	4,92	2023
Moise	Kean	3	2021
Presnel	Kimpembe	10,1	2024
Layvin	Kurzawa	4,5	2024
	Marquinhos	14,4	2024
Kylian	Mbappé	23	2022
Keylor	Navas	12	2023
	Neymar	48,9	2022
Leandro	Paredes	8	2023
Danilo	Pereira	1,2	2021
	Rafinha	5,2	2023
Sergio	Rico	2,6	2024
Jesé	Rodriguez	4,8	2021
Pablo	Sarabia	5,4	2024
Marco	Verratti	14,4	2024

IV Avec plusieurs tables

- entre tables **de même schéma** : On peut réaliser des unions, des intersections, des différences on obtient une « nouvelle » table.
pour les opérateurs ensemblistes UNION , INTERSECT, et EXCEPT.

Exemple de syntaxe : SELECT * FROM table1 UNION SELECT * FROM table2 ;

Exemple : Dans la base Poudlard

relation maisons

nom	referent	dortoir	effectif
Gryffondor	MacGonagall	tour	28
Serdaigle	Flitwick	tour	35
Poufsouffle	Chourave	logis	31
Serpentard	Rogue	donjon	25

relation professeurs

nom	prenom	mail
Dumbledore	Albus	a.dumbledore@ac-poudlard.uk
MacGonagall	Minerva	minervaMG@ac-poudlard.uk
Rubeus	Hagrid	hagrid@ac-poudlard.uk
Flitwick	Filius	flitwick@gmail.com
Chourave	Pomona	chourave@ac-poudlard.uk
Rogue	Severus	severus.rogue@ ac-poudlard.uk

que donne la requête suivante :
SELECT nom FROM professeurs EXCEPT SELECT referent FROM maisons ;

Exo 2 :

relation Presidents

nom	prenom	debut	fin
De Gaulle	Charles	1959	1969
Pompidou	Georges	1969	1974
Giscard d'Estaing	Valery	1974	1981
Mitterand	François	1981	1995
Chirac	Jacques	1995	2007
Sarkozy	Nicolas	2007	2012
Hollande	François	2012	2017
Macron	Emmanuel	2017	-

relation PremiersMinistres

nom	prenom	debut	fin
Debré	Michel	1959	1962
Pompidou	Georges	1962	1968
Couve de Murville	Maurice	1968	1969
Chaban-Delmas	Jacques	1969	1972
Messmer	Pierre	1972	1974
Chirac	Jacques	1974	1976
Barre	Raymond	1976	1981
Mauroy	Pierre	1981	1984
Fabius	Laurent	1984	1986
Chirac	Jacques	1986	1988
Rocard	Michel	1988	1991
Sarkozy	Nicolas	2007	2012
...	...	...	...
Castex	Jean	2020	-

Écrire une requête pour déterminer

a) les noms et prénoms de ceux qui ont été présidents de la République et premiers ministres.

b) les noms de ceux qui ont été présidents de la République mais pas premiers ministres

c) (pour réviser les agrégats) le nombre de premiers ministres prénommés Jacques

d) (pour réviser les agrégats *) le prénom le plus fréquent pour un premier ministre

- produit cartésien
Le produit cartésien de deux relations R et R' , noté $R \times R'$ est l'ensemble des possibilités d'association des valeurs de R et de R'

relation noms

Nom_Famille
Durand
Dupont

relation prenoms

Prenom
Alice
Bob

noms×prenoms

Nom_Famille	Prenom
Durand	Alice
Durand	Bob
Dupont	Alice
Dupont	Bob

Pour effectuer en SQL le produit cartésien, on effectue un SELECT de deux tables (ou plus)
SELECT * FROM table1, table2 ;

- La jointure symétrique permet de recoller deux relations de schémas différents du moment que deux de leurs attributs prennent les mêmes valeurs.

Relation Livres

titre	nomauteur
Les trois mousquetaires	Dumas
Notre Dame de Paris	Hugo

Relation Auteurs

nom	prenom
Hugo	Victor
Dumas	Alexandre Fils
Dumas	Alexandre Père

en algèbre relationnelle : Livres $\bowtie_{[\text{nomauteur=nom}]}$ Auteurs

en langage SQL :

```
SELECT *
FROM Livres
JOIN Auteurs
ON Livres.nomauteur = Auteurs.nom
```

titre	nomauteur	nom	prenom
Les trois mousquetaires	Dumas	Dumas	Alexandre père
Les trois mousquetaires	Dumas	Dumas	Alexandre fils
Notre Dame de Paris	Hugo	Hugo	Victor

Remarque 1 : Il y a une redondance avec les deux attributs sur lesquels a été faite la jointure (il faut donc effectuer une projection)

```
SELECT Livres.titre, Livres.nomauteur, Auteurs.prenom
FROM Livres
JOIN Auteurs
ON Livres.nomauteur=Auteurs.nom ;
```

Remarque 2 : Comme nom n'était pas une clé primaire dans la table auteur (ce qui est normalement le cas le plus fréquent), l'enregistrement Les trois mousquetaire a été "dupliqué" (d'où l'intérêt d'utiliser comme critère de jointure une clé étrangère).

Remarque 3 : On peut obtenir le même résultat avec un produit cartésien suivi d'une sélection mais ce n'est pas ce que recommande le programme d'Informatique pour tous.

```
SELECT Livres.titre, Livres.nomauteur, Auteurs.prenom
FROM Livres, Auteurs
WHERE Livres.nomauteur=Auteurs.nom ;
```

Remarque 4 : On peut faire une jointure sur plusieurs attributs, voire trois tables :
Exemple : Livres[nomauteur=nom, prenomauteur=prenom]Auteurs

```
SELECT Livres.titre, Livres.nomauteur, Auteurs.prenom
FROM Livres
JOIN Auteurs
ON Livres.nomauteur=Auteurs.nom AND Livres.prenomauteur=Auteurs.prenom ;
```

Exemple : Extraits Mines Ponts 2016

Pour suivre la propagation des épidémies, de nombreuses données sont recueillies par les institutions internationales comme l'O.M.S. Par exemple, pour le paludisme, on dispose de deux tables :

- la table palu recense le nombre de nouveaux cas confirmés et le nombre de décès liés au paludisme ; certaines lignes de cette table sont données en exemple (on précise que iso est un identifiant unique pour chaque pays)
- la table demographie recense la population totale de chaque pays ; certaines lignes de cette table sont données en exemple.

relation palu

nom	iso	annee	cas	deces
Bresil	BR	2009	309 316	85
Bresil	BR	2010	334 667	76
Kenya	KE	2010	898 531	26 017
Mali	ML	2011	307 035	2 128
Ouganda	UG	2010	1 581 160	8 431
...	...	...	...	...

relation demographie

pays	periode	pop
BR	2009	193 020 000
BR	2010	194 946 000
KE	2010	40 909 000
ML	2011	14 417 000
UG	2010	33 987 000
...	...	...

Écrire une requête en langage SQL qui détermine le taux d'incidence du paludisme en 2011 pour les différents pays de la table palu.

Exo 3 : On revient à la base Poudlard

relation maisons

nom	referent	dortoir	effectif
Gryffondor	MacGonagall	tour	28
Serdaigle	Flitwick	tour	35
Poufsouffle	Chourave	logis	31
Serpentard	Rogue	donjon	25

relation eleves

nom	prenom	moyenne	maison
Granger	Hermione	19	Gryffondor
Diggory	Cedric	17	Poufsouffle
Malefoy	Drago	13	Serpentard
Potter	Harry	17	Gryffondor
Weasley	Ron	16	Gryffondor

relation professeurs

nom	prenom	mail
Dumbledore	Albus	a.dumbledore@ac-poudlard.uk
MacGonagall	Minerva	minervaMG@ac-poudlard.uk
Filtwick	Filius	flitwick@gmail.com
Chourave	Pomona	chourave@ac-poudlard.uk
Rogue	Severus	severus.rogue@ ac-poudlard.uk
Hagrid	Rubeus	hagrid@ac-poudlard.uk

a) Combien d'enregistrements comptabiliserait le résultat de la requête :
SELECT * FROM maisons,eleves,professeurs ;

b) Donner une requête qui donne le nom du référent de Drago Malefoy.

c) Donner une requête qui les prénoms des professeurs responsables de plus de 30 élèves.

d) Donner une requête qui donne l'adresse du professeur de Cedric Diggory.

e) Quels sont les prénoms des professeurs responsables de maison ?

f) *pour aller plus loin* Quelles sont les maisons dont le référent a une adresse qui ne finit pas par @ac-poudlard.uk

Exo 4 : Nous allons nous intéresser à la gestion des données qui sont stockées dans la base de données nommée *Cinema*, constituée de deux tables intitulées *Films* et *Realisateurs*. Les contenus de ces tables se trouvent en Page de gauche.

La table *Films* est constituée de 6 champs :

- *id* : de type INTEGER- clé primaire ;
- *Titre* : de type TEXT ;
- *Realisateur* : de type INT (clé étrangère) ;
- *Annee* : de type INTEGER ;
- *Entrees* : de type INT ;
- *Pays* : de type TEXT.

Ce sont les 20 films ayant fait le plus d'entrées au cinéma en France.

La table *Realisateurs* est constituée de 4 champs :

- *id* : de type INTEGER – clé primaire ;
- *Nom* : de type TEXT ;
- *Prenom* : de type TEXT ;
- *Nboscars* : de type INT.

Écrire (en SQL) les requêtes permettant d'obtenir

a) la liste des titres de films par nombre d'entrées décroissantes.

b) la liste des titres de films de ce classement réalisés à partir de l'an 2000.

c) le nombre de films de ce classement par pays.

d) la liste des réalisateurs ayant le plus grand nombre d'oscars.

e) la liste des titres de films Américains ainsi que le nom et prénom de leur réalisateurs de ce classement.

f) la liste des titres des films dont le réalisateur a eu un oscar.

g) (*) le nom du réalisateur ayant le le plus grand nombre de film dans le classement.

Base de données « Cinema »

Table *Films*

id	Titre	Realisateur	Annee	Entrees	Pays
1	Titanic	1	1997	20 634 793	US
2	Bienvenue chez les Ch'tis	2	2008	20 413 165	FR
3	Blanche-Neige et les Sept Nains	4	1938	18 319 651	US
4	La Grande Vadrouille	5	1966	17 273 065	FR
5	Intouchables	3	2011	19 385 300	FR
6	Autant en emporte le Vent	6	1650	16 723 795	US
7	Il etait une fois dans l'ouest	8	1968	14 873 304	IT
8	Avatar	1	2009	14 677 888	US
9	Le Livre de la jungle	7	1967	14 696 567	US
10	Les 101 Dalmatiens	7	1961	14 660 594	US
11	Asterix et Obelix : Mission Cleopatre	9	2002	14 314 786	FR
12	Les dix commandements	10	1958	14 229 563	US
13	Ben hur	11	1960	13 853 547	US
14	Les visiteurs	12	1993	13 671 595	FR
15	Le pont de la riviere Kwai	13	1957	13 475 099	US
16	Cendrillon	14	1950	13 216 631	US
17	Les Aristochats	7	1971	12 541 369	US
18	Le petit monde de Don Camillo	15	1952	12 790 676	FR
19	Qu'est ce qu'on a fait au bon dieu	16	2014	12 237 274	FR
20	Le jour le plus long	17	1962	11 933 629	US

Table *Realisateurs*

id	Nom	Prenom	Nboscars
1	Cameron	James	3
2	Boon	Dany	0
3	Toledano	Etienne	0
4	Hand	David	0
5	Oury	Gerard	0
6	Fleming	Victor	1
7	Reitherman	Wolfgang	0
8	Leone	Sergio	0
9	Chabat	Alain	0
10	DeMille	Cecil B	2
11	Wyler	William	3
12	Poiré	Jean Marie	0
13	Lean	David	2
14	Jackson	Wilfred	0
15	Duvivier	Julien	0
16	de Chauveron	Philippe	0
17	Annakin	Ken	0