

# Corrigé du TP 14 Base de Données n° 1

Chenevois-Jouhet-Junier

## 1 Exercice 2 Partie A

```
def lecture(fichier):  
    """Ouvre un fichier texte où chaque ligne est constituée de champs  
    séparés par des tabulations et se termine par un saut de ligne.  
    Lit le fichier ligne par ligne et collecte les différents champs dans  
    un tuple puis insère ce tuple dans une liste"""  
    f = open(fichier)  
    collecteur = []  
    for ligne in f:  
        champs = tuple(ligne.rstrip().split('\t'))  
        collecteur.append(champs)  
    f.close()  
    return collecteur
```

```
film = lecture('film.txt')  
jouer = lecture('jouer.txt')  
projection = lecture('projection.txt')  
personne = lecture('personne.txt')
```

```
"""
```

```
In [1]: film[0]  
Out[1]: ('01', '15', 'Crash', 'Drame', '1996')
```

```
In [2]: jouer[0]  
Out[2]: ('01', '05', 'Grace')
```

```
In [3]: projection[0]  
Out[3]: ('02', '05', '01-05-2002')
```

```
In [4]: personne[0]  
Out[4]: ('01', 'Kidman', 'Nicole')  
"""
```

```
def requete1():
```

```

"""Retourne la liste des personnes dont le prenom est John"""
t = []
for p in personne:
    if p[-1] == 'John':
        t.append(p)
return t

"""
In [8]: requete1()
Out[8]: [('05', 'Travolta', 'John'), ('12', 'Wayne', 'John'),
('18', 'Glen', 'John')]
"""

def requete2():
    """Retourne la liste de tous les prenom dans l'ordre alphabetique"""
    prenom = []
    for p in personne:
        prenom.append(p[-1])
    prenom.sort()
    return prenom

"""
In [14]: requete2()
Out[14]:
['Angelina', 'Bruce',... , 'Stellan']
"""

def requete3():
    """Retourne la liste de tous les prenom dans l'ordre alphabetique
sans doublons"""
    prenom = requete2()
    newprenom = [prenom[0]]
    #on utilise le fait que prenom est deja dans l'ordre alphabetique
    for p in prenom[1:]:
        if p != newprenom[-1]:
            newprenom.append(p)
    return newprenom

"""
In [2]: len(requete2())
Out[2]: 22

In [3]: len(requete3())
Out[3]: 19
"""

def requete4():
    """Retourne la liste de tous les films realises par Lars Von trier"""
    #recherche de l'identifiant de Lars Von Trier
    for p in personne:
        if p[1] == 'von Trier':

```

```

        id = p[0]
        break
    filmTrier = []
    for f in film:
        if f[1] == id:
            filmTrier.append(f[2])
    return filmTrier

"""
In [17]: requete4()
Out[17]: ['Breaking the waves', 'Dogville']
"""

```

## 2 Exercice 2 Partie B Base : cinema843.db

---

Exo 2 : Toutes les lignes avec toutes les colonnes de la table personne

```
SELECT * FROM personne
```

```

(1, Kidman, Nicole)
(2, Bettany, Paul)
(3, Watson, Emily)
(4, Skarsgard, Stellan)
... (18 de plus)

```

Exo 2 : Tous les prenom de la table personne dans l'ordre croissant des prenom

```
SELECT prenom FROM personne ORDER BY prenom ASC
```

```

(Angelina)
(Bruce)
(Clint)
(David)
... (18 de plus)

```

Exo 2 : Tous les prenom de la table personne avec suppression des doublons. Les deux requetes donnent le meme resultat.

```
SELECT DISTINCT prenom FROM personne ORDER BY prenom ASC
```

```
SELECT DISTINCT prenom FROM personne
```

```

(Nicole)
(Paul)
(Emily)

```

personne		
idp	nom	prenom
1	Kidman	Nicole
2	Bettany	Paul
3	Watson	Emily
4	Skarsgard	Stellan
5	Travolta	John
6	L. Jackson	Samuel
7	Willis	Bruce
8	Irons	Jeremy
9	Spader	James
10	Hunter	Holly
11	Arquette	Rosanna
12	Wayne	John
13	von Trier	Lars
14	Tarantino	Quentin
15	Cronenberg	David
16	Mazursky	Paul
17	Jones	Grace
18	Glen	John
19	Eastwood	Clint
20	Spacey	Kevin
21	Mendes	Sam
22	Jolie	Angelina

projection		
idc	idf	jour
2	5	2002-05-01
2	5	2002-05-02
2	5	2002-05-03
2	4	1996-12-02
1	1	1996-05-07
2	7	1985-05-09
1	4	1996-08-02
4	3	1994-04-08
3	6	1990-12-02
2	2	1990-12-08
3	3	1994-11-05
4	3	1994-11-06
1	6	1980-07-05
2	4	1996-09-02
4	6	2002-08-01
3	6	1960-11-09
1	2	1988-03-12
2	8	1989-02-01
2	1	1997-05-11
2	3	1994-07-05
2	6	2002-08-01
1	3	1994-03-02
2	9	2008-12-02
1	10	2000-10-03
2	11	2004-03-02

film				
idf	idr	titre	genre	annee
1	15	Crash	Drame	1996
2	15	Faux-Semblants	Epouvante	1988
3	14	Pulp Fiction	Policier	1994
4	13	Breaking the waves	Drame	1996
5	13	Dogville	Drame	2002
6	12	Alamo	Western	1960
7	18	Dangereusement vôtre	Espionnage	1985
8	19	Chasseur blanc, coeur noir	Drame	1989
9	19	Minuit dans le jardin du bien et du mal	Policier	1998
10	21	American Beauty	Drame	1999
11	19	L'Echange	Drame	2008

jouer		
ida	idf	role
1	5	Grace
2	5	Tom Edison
3	4	Bess
4	4	Jan
5	3	Vincent Vega
6	3	Jules Winnfield
7	3	Butch Coolidge
8	2	Beverly & Elliot Mantle
9	1	James Ballard
10	1	Helen Remington
11	1	Gabrielle
4	5	Chuck
16	7	May Day
19	8	John Wilson
20	9	Jim Williams
20	10	Lester Burnham

cinema		
idc	nom	adresse
2	UGC	Part-Dieu
1	Pathé	Bellecour
3	Astoria	Cours Vitton
4	Comedia	Avenue Berthelot

Figure 1: Schéma relationnel de la table cinema843.db

(Stellan)  
... (15 de plus)

Exo 2 : Toutes les lignes de la table personne qui contiennent le prenom 'John'

```
SELECT * FROM personne WHERE prenom = 'John'
```

(5, Travolta, John)  
(12, Wayne, John)  
(18, Glen, John)

Exo 2 : Récupère l'idp de Lars von Trier

```
SELECT idp FROM personne  
WHERE nom = 'von Trier' AND prenom = 'Lars'
```

(13)

Exo 2 : Liste des titres de films réalisés par Lars von Trier

```
SELECT titre FROM film  
WHERE idr = (SELECT idp FROM personne  
WHERE personne.nom = 'von Trier'  
AND personne.prenom = 'Lars')
```

(Breaking the waves)  
(Dogville)

Exo 2 : Déterminer les titres de films dont le genre est Drame

```
SELECT titre FROM film WHERE genre = 'Drame'
```

(Crash)  
(Breaking the waves)  
(Dogville)  
... (3 de plus)

Exo 2 : Déterminer les titres des films réalisés dans les années 80

```
SELECT titre FROM film  
WHERE 1980 <= annee AND annee <= 1990
```

(Faux-Semblants)  
(Dangereusement vôtre)  
... (1 de plus)

Exo 2 : Déterminer le nombre de projections de films

```
SELECT COUNT(*) FROM projection
```

(25)

Exo 2 : Déterminer les rôles joués par Kevin Spacey

```
SELECT role FROM jouer
  WHERE ida = (SELECT idp FROM personne
               WHERE nom = 'Spacey' AND prenom = 'Kevin')
```

(Jim Williams)

(Lester Burnham)

### 3 Exercice 3 Base : base\_triangles.db

Exo 3 : Nombre total de triangles

```
SELECT COUNT(*) FROM triangles
```

(100000)

Exo 3 : Tous les triangles dont le périmètre est 100

```
SELECT * FROM triangles WHERE ab+ac+bc=100
```

(471, 47, 50, 3)

(607, 35, 32, 33)

... (226 de plus)

Exo 3 : Tous les produits des cotes des triangles de périmètre  $\geq 100$

```
SELECT ab*ac*bc FROM triangles WHERE ab+ac+bc>=100
```

(12187500)

(50150)

... (91627 de plus)

Exo 3 : Insertion d'une ligne supplémentaire

```
INSERT INTO triangles VALUES (100001,1, 99, 1)
```

Exo 3 : Insertion d'une ligne supplémentaire

```
INSERT INTO triangles VALUES (100002,99, 1, 1)
```

Exo 3 : Tous les triangles équilatéraux

```
SELECT * FROM triangles WHERE ab = ac AND ab = bc
```

(15, 31, 31, 31)

(2045, 6, 6, 6)

... (17 de plus)

Exo 3 : Tous les triangles rectangles en A

```
SELECT * FROM triangles WHERE bc*bc=ab*ab+ac*ac
```

(0, 125, 325, 300)

(127, 70, 74, 24)

... (64 de plus)

Exo 3 : Le nombre de triangles rectangles en A

```
SELECT COUNT(*) FROM triangles WHERE bc*bc=ab*ab+ac*ac
```

(66)

Exo 3 : Le maximum des périmètres des triangles rectangles en A

```
SELECT MAX(ab+bc+ac) FROM triangles WHERE bc*bc=ab*ab+ac*ac
```

(1400)

Exo 3 : La plus petite valeur des produits  $AB.AC.BC$ , pour les triangles  $ABC$  de périmètre supérieur ou égal à 100

```
SELECT MIN(ab*bc*ac) FROM triangles WHERE ab+ac+bc>=100
```

(2450)

Exo 3 : Les longueurs correspondants au(x) triangle(s) pour le(s)quel(s) le minimum précédent est atteint

```
SELECT ab,ac,bc ,ab*ac*bc AS produit FROM triangles
WHERE ab+ac+bc>=100 AND ab*ac*bc=(
    SELECT MIN(ab*ac*bc) FROM triangles
    WHERE ab+ac+bc>=100
)
```

(50, 1, 49, 2450)

Exo 3 : Ensemble des triangles plats

```
SELECT * FROM triangles WHERE ab+ac=bc OR ab+bc=ac OR ac+bc=ab
```

(20, 16, 55, 39)

(38, 45, 97, 52)  
... (2877 de plus)

Exo 3 : Suppression des lignes insérées en 2)

```
DELETE FROM triangles WHERE idt > 100000
```

## 4 Exercice 4 Base : base\_triangles.db

```
import sqlite3
base = sqlite3.connect('base_triangles.db')
curseur = base.cursor()
res = curseur.execute("""SELECT * FROM triangles WHERE ab+bc+ac=100""")
foo = res.fetchall()
curseur.close()
base.close()
print('res=',res,end='\n\n')
print('foo=',foo,end='\n\n')
print(len(foo),end='\n\n')

"""
res= <sqlite3.Cursor object at 0xb44a09e0>

foo= [(24, 51, 12, 37), ..., (99955, 24, 72, 4)]

474
"""
```

## 5 Exercice 5 Base : communes-regions-departements.db

Exo 5 : Jointure entre les tables communes et departements

```
SELECT c.nom, d.nom
FROM communes AS c JOIN departements AS d
ON c.dep = d.id
```

(L'Abergement-Clémenciat, Ain)  
(L'Abergement-de-Varey, Ain)  
... (36703 de plus)

Exo 5 : Liste de toutes les communes avec pour chacune, son département, sa région et sa population

```
SELECT c.nom, d.nom, r.nom, c.pop
FROM communes AS c JOIN departements AS d JOIN regions AS r
ON c.dep=d.id AND d.reg=r.id
```



(L'Abergement-Clémenciat, Ain, Rhône-Alpes, 784)  
(L'Abergement-de-Varey, Ain, Rhône-Alpes, 221)  
... (36703 de plus)

Exo 5 : Liste des villes de plus de 100000 habitants, ainsi que leur population et leur région

```
SELECT  c.nom, r.nom, c.pop
FROM    communes AS c JOIN departements AS d JOIN regions AS r
ON      c.dep=d.id AND d.reg=r.id
WHERE   c.pop >= 100000
ORDER BY c.pop DESC
```

(Paris, Île-de-France, 2243833)  
(Marseille, Provence-Alpes-Côte d'Azur, 850726)  
... (39 de plus)

Exo 5 : Liste des communes (nom et population) dont le nom commence par 'Pa' et se finit par 'is'

```
SELECT  c.nom, r.nom, c.pop
FROM    communes AS c JOIN departements AS d JOIN regions AS r
ON      c.dep=d.id AND d.reg=r.id
WHERE   c.nom LIKE 'Pa%is'
```

(Pargny-les-Bois, Picardie, 136)  
(Passy-en-Valois, Picardie, 164)  
... (4 de plus)

Exo 5 : Liste des communes qui ont strictement plus de lettres dans leur nom que leur nombre d'habitants

```
SELECT  nom, pop FROM communes WHERE  LENGTH(nom )>pop
```

(Majastres, 2)  
(Saint-Martin-lès-Seyne, 18)  
... (52 de plus)

## 6 Exercice 6 Base : cinema843.db

Exo 6 : Quels sont les titres des films réalisés par Lars von Trier ?

```
SELECT film.titre FROM film
      JOIN personne
      ON  personne.idp = film.idr
      WHERE  personne.nom = 'von Trier' AND  personne.prenom = 'Lars'
```

(Breaking the waves)  
(Dogville)

Exo 6 : Quels sont les titres des films où Kevin Spacey a joué ?

```
SELECT film.titre FROM personne
  JOIN jouer ON personne.idp = jouer.ida
  JOIN film ON jouer.idf = film.idf
  WHERE personne.prenom = 'Kevin' AND personne.nom = 'Spacey'
```

(Minuit dans le jardin du bien et du mal)  
(American Beauty)

Exo 6 : Quels sont les drames que l'on a pu voir après le 1er janvier 2002 ?

```
SELECT film.titre, film.genre, projection.jour
  FROM film JOIN projection
    ON film.idf = projection.idf
  WHERE film.genre = 'Drame'
    AND projection.jour >= '2002-01-01'
```

(Dogville, Drame, 2002-05-01)  
(Dogville, Drame, 2002-05-02)  
... (2 de plus)

Exo 6 : Quels sont les noms et prénoms des personnes qui sont des acteurs ? Ordonner la liste par ordre alphabétique croissant des noms.

```
SELECT DISTINCT personne.nom, personne.prenom
  FROM personne JOIN jouer
    ON personne.idp = jouer.ida
  ORDER BY personne.nom
```

(Arquette, Rosanna)  
(Bettany, Paul)  
... (12 de plus)

Exo 6 : Quels sont les noms et prénoms des personnes qui sont des réalisateurs ?

```
SELECT DISTINCT personne.nom, personne.prenom
  FROM personne JOIN film
    ON personne.idp = film.idr
```

(Wayne, John)  
(von Trier, Lars)  
... (5 de plus)

Exo 6 : Quels sont les noms et prénoms des personnes qui sont à la fois des acteurs et des réalisateurs ?

```
SELECT DISTINCT personne.nom, personne.prenom
  FROM personne JOIN jouer
    ON personne.idp = jouer.ida
```

```

INTERSECT
SELECT DISTINCT personne.nom, personne.prenom
      FROM personne JOIN film
            ON personne.idp = film.idr

```

(Eastwood, Clint)

Exo 6 : Quels sont les noms et prénoms des acteurs qui ne sont pas des réalisateurs ?

```

SELECT DISTINCT personne.nom, personne.prenom
      FROM personne JOIN jouer
            ON personne.idp = jouer.ida
EXCEPT
SELECT DISTINCT personne.nom, personne.prenom
      FROM personne JOIN film
            ON personne.idp = film.idr

```

(Arquette, Rosanna)

(Bettany, Paul)

... (11 de plus)

Exo 6 : Noms et prénoms des réalisateurs qui ont réalisé des films policier.

```

SELECT DISTINCT personne.nom, personne.prenom
      FROM personne JOIN film
            ON personne.idp = film.idr
            WHERE genre = 'Policier'

```

(Tarantino, Quentin)

(Eastwood, Clint)

Exo 6 : Noms et prénoms des réalisateurs qui ont réalisé des films policier et des films dramatiques.

```

SELECT DISTINCT personne.nom, personne.prenom
      FROM personne JOIN film
            ON personne.idp = film.idr
            WHERE genre = 'Policier'
INTERSECT
SELECT DISTINCT personne.nom, personne.prenom
      FROM personne JOIN film
            ON personne.idp = film.idr
            WHERE genre = 'Drame'

```

(Eastwood, Clint)

Exo 6 : Liste de toutes les interprétations possibles en précisant le nom et le prénom de l'acteur ainsi que le rôle et le titre du film

```
SELECT personne.nom, personne.prenom, film.titre, jouer.role
FROM personne JOIN jouer ON personne.idp = jouer.ida
JOIN film ON jouer.idf = film.idf
ORDER BY personne.nom DESC
```

(Willis, Bruce, Pulp Fiction, Butch Coolidge)  
(Watson, Emily, Breaking the waves, Bess)  
... (14 de plus)

Exo 6 : Titres des films réalisés par David Cronenberg qui ont été projetés au cinéma UGC

```
SELECT film.titre, personne.nom, personne.prenom, cinema.nom, projection.jour
FROM personne JOIN film ON personne.idp = film.idr
JOIN projection ON film.idf = projection.idf
JOIN cinema ON projection.idc = cinema.idc
WHERE personne.nom = 'Cronenberg'
AND personne.prenom = 'David'
AND cinema.nom = 'UGC'
```

(Faux-Semblants, Cronenberg, David, UGC, 1990-12-08)  
(Crash, Cronenberg, David, UGC, 1997-05-11)

Exo 6 : Acteurs qui ont joué dans des films projetés au cinéma UGC depuis l'an 2000

```
SELECT DISTINCT personne.nom, personne.prenom
FROM personne JOIN jouer JOIN film JOIN projection JOIN cinema
ON personne.idp = jouer.ida AND jouer.idf = film.idf
AND film.idf = projection.idc AND projection.idc = cinema.idc
WHERE projection.jour >= '2000-01-01'
AND cinema.nom = 'UGC'
```

(Irons, Jeremy)

Exo 6 : Titres des films où Stellan Skarsgard a joué un role et qui ont été projetés au cinéma UGC

```
SELECT DISTINCT film.titre, personne.nom, personne.prenom
FROM personne JOIN jouer ON personne.idp = jouer.ida
JOIN film ON jouer.idf = film.idf
JOIN projection ON jouer.idf = projection.idf
JOIN cinema ON projection.idc = cinema.idc
WHERE personne.nom = 'Skarsgard'
AND personne.prenom = 'Stellan'
AND cinema.nom = 'UGC'
ORDER BY film.titre
```

(Breaking the waves, Skarsgard, Stellan)  
(Dogville, Skarsgard, Stellan)