Homework : sqlite3 Report

Computer Technologies In Telecommunications

William Baudrand Chaudeyrac

November 2021

1 Introduction

In one of the previous meeting, we have seen how to install sqlite3 on python, and also how to use the functions that we have seen on Mysql previously. This time, i will manipulate databases with python using sqlite3 tools.

2 Convert a .sql file to a .db file

2.1 Creating a .sql file

I started this project by creating a database. I generated a database using Mockaroo, with 5 columns : the Film name, it's genre, a Year (which was at first not accurate, an ID (here an hexadecimal code) and a rating which also was inaccurate. Then, I went on IMDB in order to modify the Year and the Rating of each film. I modified records manually, because some of the generated movies names haven't been found on the IMDB website. The result is visible on figure 1.

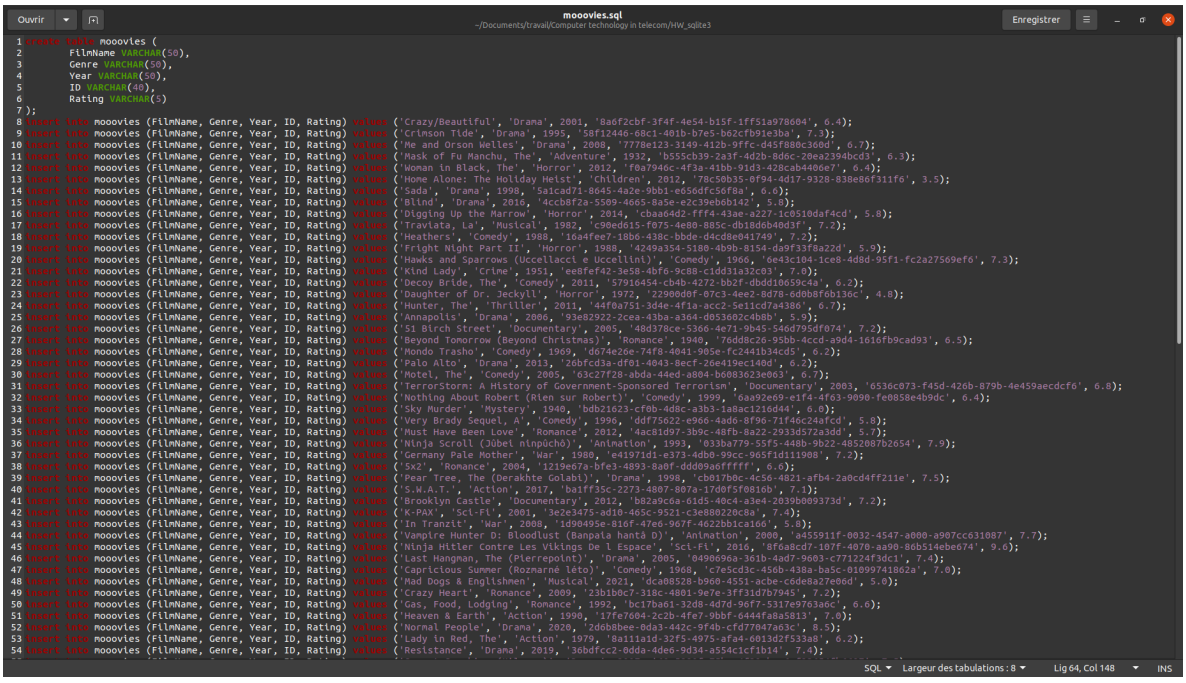

Figure 1: 100 records sql database (modified)

2.2 Create a converter in Python

First, i tried to use the sql database with sqlite3. After many failure, i decided to look at the homework task and saw that we should use a .db file, which make more sense. In the first time, i look through internet to find a .sql to .db converter. It looks like it doesn't exist. So, i searched for someone that has used a .sql database with sqlite3 and that has issues with it. Thanks to that, i found a code that i modified to fit my files and goals to be able to convert my .sql file into a .db (figure 2).
With this code, i obtained a .db file that could not bo read. So, to be able to see if my code worked, i installed DB Browser for Sqlite, that help me to take a screenshot of the database (figure 3).

```python
# the way we want to name the file
DATABASE = './mooovies.db'
DEBUG = True


app = Flask(__name__)
app.config.from_object(__name__)

def init_db():
    with closing(connect_db()) as db:
        with app.open_resource('mooovies.sql', mode='r') as f:#open the sql file and read it
            db.cursor().executescript(f.read())
        db.commit()# save the modifications into the database

def connect_db():
    return sqlite3.connect(app.config['DATABASE'])
```

Figure 2: Script to create a new database file from a sql

mooovies

Table : mooovies

Filtrer dans n'importe quelle colonne

	FilmName	Genre	Year	ID	Rating
	Filtre	Filtre		Filtre	...
1	Crazy/Beautiful	Drama	2001	8a6f2cbf-3f4f-4e54-b15f-1ff51a978604	6.4
2	Crimson Tide	Drama	1995	58f12446-68c1-401b-b7e5-b62cfb91e3ba	7.3
3	Me and Orson Welles	Drama	2008	7778e123-3149-412b-9ffc-d45f880c360d	6.7
4	Mask of Fu Manchu, The	Adventure	1932	b555cb39-2a3f-4d2b-8d6c-20ea2394bcd3	6.3
5	Woman in Black, The	Horror	2012	f0a7946c-4f3a-41bb-91d3-428cab4406e7	6.4
6	Home Alone: The Holiday Heist	Children	2012	78c50b35-0f94-4d17-9328-838e86f311f6	3.5
7	Sada	Drama	1998	5a1cad71-8645-4a2e-9bb1-e656dfc56f8a	6.6
8	Blind	Drama	2016	4ccb8f2a-5509-4665-8a5e-e2c39eb6b142	5.8
9	Digging Up the Marrow	Horror	2014	cbaa64d2-fff4-43ae-a227-1c0510daf4cd	5.8
10	Traviata, La	Musical	1982	c90ed615-f075-4e80-885c-db18d6b40d3f	7.2
11	Heathers	Comedy	1988	16a4fee7-18b6-438c-bbde-d4cd8e041749	7.2
12	Fright Night Part II	Horror	1988	4249a354-5180-4b9b-8154-da9f33f8a22d	5.9
13	Hawks and Sparrows (Uccellacci e Uccellini)	Comedy	1966	6e43c104-1ce8-4d8d-95f1-fc2a27569ef6	7.3
14	Kind Lady	Crime	1951	ee8fef42-3e58-4bf6-9c88-c1dd31a32c03	7.0
15	Decoy Bride, The	Comedy	2011	57916454-cb4b-4272-bb2f-dbdd10659c4a	6.2
16	Daughter of Dr. Jeckyll	Horror	1972	22900d0f-07c3-4ee2-8d78-6d0b8f6b136c	4.8
17	Hunter, The	Thriller	2011	44f0a751-3d4e-4f1a-acc2-5e11cd7a4386	6.7
18	Annapolis	Drama	2006	93e82922-2cea-43ba-a364-d053602c4b8b	5.9
19	51 Birch Street	Documentary	2005	48d378ce-5366-4e71-9b45-546d795df074	7.2
20	Beyond Tomorrow (Beyond Christmas)	Romance	1940	76dd8c26-95bb-4ccd-a9d4-1616fb9cad93	6.5
21	Mondo Trasho	Comedy	1969	d674e26e-74f8-4041-905e-fc2441b34cd5	6.2
22	Palo Alto	Drama	2013	26bfcd3a-df01-4043-8ecf-26e419ec140d	6.2
23	Motel, The	Comedy	2005	63c27f28-abda-44ed-a804-b6083623e063	6.7
24	TerrorStorm: A History of Government-Spo...	Documentary	2003	6536c073-f45d-426b-879b-4e459aecdcf6	6.8
25	Nothing About Robert (Rien sur Robert)	Comedy	1999	6aa92e69-e1f4-4f63-9090-fe0858e4b9dc	6.4
26	Sky Murder	Mystery	1940	bdb21623-cf0b-4d8c-a3b3-1a8ac1216d44	6.0
27	Very Brady Sequel, A	Comedy	1996	ddf75622-e966-4ad6-8f96-71f46c24afcd	5.8
28	Must Have Been Love	Romance	2012	4ac81d97-3b9c-48fb-8a22-2933d572a3dd	5.7
29	Ninja Scroll (Jûbei ninpûchô)	Animation	1993	033ba779-55f5-448b-9b22-4852087b2654	7.9
30	Germany Pale Mother	War	1980	e41971d1-e373-4db0-99cc-965f1d111908	7.2

1 - 31 de 100

Aller à : 1

Figure 3: The database in .db, visible thanks to DB browser

3 Having fun with the Database

Let's modify the database that we have now !

3.1 Creating the database and the table

To create a database and to be able to reuse this code in future projects, it is better to use OOP with class (figure 4). This is why i created a class named DATA_BASE. the function __init__ is used to create a new object (a new database) and the Table_Creator will be use to create a table to store the data, the same way as the previous homework on Mysql.

```
class Data_Base():

    def __init__(self,DB_Name : str ):# To create the database

        self.db = sqlite3.connect(DB_Name)
        self.cur = self.db.cursor()

    def Table_creator(self, Table_name : str, column : str ):# To cleate a table where Data will be store

        self.cur.execute('CREATE table IF NOT EXISTS' + name + ' ( ' + column + ' )' )
        self.db.commit()
```

Figure 4: Object creation for the database and the table

3.2 display existing data

It is important to know what are the data inside the database to be able to see if the modifications have been applied. That is why i've created an other method to print all the data from the database (figure 5). The, i use will use this method many times in the main program such as in figure 6.

```
def Print_Data_Table(self, column : str , Table : str):#print All the Data that is inside the table


    self.cur.execute("SELECT " + column + " FROM " + Table ) #taking all the data contained in the table

    for ligne in self.cur :# displaying all the data from the table
        print("FilmName : {0}  | Genre : {1} | Year : {2} | Rating : {4} | ID : {3}".format(ligne[0],ligne[1],ligne[2],ligne[3],ligne[4]))
```

Figure 5: creation of the SELECT * From datatable

```
Test_DB = Data_Base("mooovies.db") #I create my object that contain the data from the database mooovies
print("\n Original Data \n")
Test_DB.Print_Data_Table("*","mooovies")
```

Figure 6: using the method in the main program

On figure 7 and 8, we can see that the results printed by my method are exactly the same as the one contained in the database file. It's great news, because i didn't applied for now a modification to the database.

Figure 7: Original data screen 1

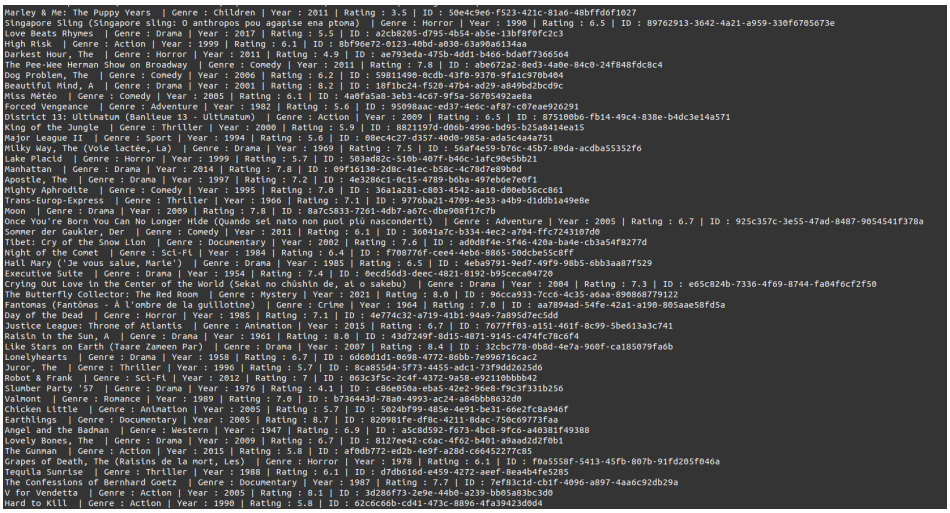

Figure 8: Original data screen 2

3.3 Adding some new data

Now, it is time to change things in the database. Lets start with adding some new data. First, I created a new method named Data_Adder, that will be able to add new movies to the list (figure 9). Then, i'm using it in the main program (figure 10). I'm adding *Prend la ciboulette et tire toi*, an amateur film made by JDG in 2016, and i made it a Drama, with a Rating of 5.0. i also gave it an ID in hexadecimal that does not correspond to an other film in the list. As we can see, it appears at the

```
def Data_Adder(self,Table : str ,Column_List : str ,data): #to add data to the Table

    Column_Number = ""# calculate the number of the column. this part of the code can be used for an
    for i in range(0,len(Column_List.split(',')) - 1 ,1) :
        Column_Number = Column_Number + ("?,")
    Column_Number = Column_Number + ("?")

    request = 'INSERT INTO ' + Table + ' ( ' + Column_List + ' ) VALUES ( ' + Column_Number + ')'#pu
    self.cur.execute(request, data)
    self.db.commit()
```

Figure 9: the method of the data adder

```
print("\n Part 1 : Adding Data \n")
Test_DB.Data_Adder("mooovies","FilmName,Year,Genre,Rating,ID",("prend la ciboulette et tire toi",2016,"Drama",5.0,"aaca0baa-0faa-4daa-aa28-aaae86faaaf6") )
Test_DB.Print_Data_Table("*","mooovies")
```

Figure 10: adding data in the main program

```
FilmName : Valmont  | Genre : Romance | Year : 1989 | Rating : 7.0 | ID : b736443d-78a0-4993-ac24-a84bbb8632d0
FilmName : Chicken Little  | Genre : Animation | Year : 2005 | Rating : 5.7 | ID : 5024bf99-485e-4e91-be31-66e2fc8a946f
FilmName : Earthlings  | Genre : Documentary | Year : 2005 | Rating : 8.7 | ID : 820981fe-df8c-4211-8dac-750c69773faa
FilmName : Angel and the Badman  | Genre : Western | Year : 1947 | Rating : 6.9 | ID : a5c8d592-f673-4bc8-9fc6-a40381f49388
FilmName : Lovely Bones, The  | Genre : Drama | Year : 2009 | Rating : 6.7 | ID : 8127ee42-c6ac-4f62-b401-a9aad2d2f0b1
FilmName : The Gunman  | Genre : Action | Year : 2015 | Rating : 5.8 | ID : af0db772-ed2b-4e9f-a28d-c66452277c85
FilmName : Grapes of Death, The (Raisins de la mort, Les)  | Genre : Horror | Year : 1978 | Rating : 6.1 | ID : f0a5558f-5413-45fb-807b-91fd205f046a
FilmName : Tequila Sunrise  | Genre : Thriller | Year : 1988 | Rating : 6.1 | ID : d7db616d-e459-4272-aeef-8ea4b4fe5285
FilmName : The Confessions of Bernhard Goetz  | Genre : Documentary | Year : 1987 | Rating : 7.7 | ID : 7ef83c1d-cb1f-4096-a897-4aa6c92db29a
FilmName : V for Vendetta  | Genre : Action | Year : 2005 | Rating : 8.1 | ID : 3d286f73-2e9e-44b0-a239-bb05a83bc3d0
FilmName : Hard to Kill  | Genre : Action | Year : 1990 | Rating : 5.8 | ID : 62c6c66b-cd41-473c-8896-4fa39423d0d4
FilmName : prend la ciboulette et tire toi  | Genre : Drama | Year : 2016 | Rating : 5.0 | ID : aaca0baa-0faa-4daa-aa28-aaae86faaaf6
```

Figure 11: adding "prend la ciboulette et tire toi"

end of the list in figure 11, so the adding is successful.

3.4 Updating some records

In this part, i will update 20 records.
But First, lets start with the creation of a method that can help me to modify data. Many methods exists in sql to modify data, but the one that i will use here is with UPTADE Table SET new data WHERE conditions for the new data (figure 12).

```
def Data_Updater(self, Table : str , New_Data : str , Modif_Condition : str):# to update 1 ty

    self.cur.execute("UPDATE " + Table + " SET " + New_Data + " WHERE " + Modif_Condition)
    self.db.commit()
```

Figure 12: creating the method Data_Updater

Then i'm gonna use this methods in few different ways in order to modify 20 records.
The first update is changing a movie's name (so it's a 1 record change). in the figure 13, we can see that i modified the name of the film *5x2* and replaced it with *La Boucherie Martin*. The modification can be seen on figure 14.

```
print("\n Part 2 : Updating Data  \n")
#first update : changing movie name (1 record change)
Test_DB.Data_Updater("mooovies","ID = '1219e67a-bfe3-4893-8a0f-ddd09a6fffff',FilmName = 'La Boucherie Martin'","FilmName = '5x2'")#
#second update : changig Genre Horror to Comedy  (10 records changed)
Test_DB.Data_Updater("mooovies","Genre = 'Comedy'","Genre = 'Horror'")# update some data
#third update : changing Rating of animations films (4 records changed)
Test_DB.Data_Updater("mooovies","Rating = 8.5","Genre = 'Animation'")# update some data
#fourth update : making all films from 2012 Fantasy films (5 records)
Test_DB.Data_Updater("mooovies","Genre = 'Fantasy'","Year = 2012")# update some data
Test_DB.Print_Data_Table("*","mooovies")
```

Figure 13: Using the data updater in different ways through the main program

The second update is changing the Genre of Horror movies to Comedy. It has changed 10 records because 10 of them where at the start Horror movies. Results can be observed on figures 14 and 15.
The third update is changing Rating of animations films. Here, there are 4 animations movies, so all of them given a rating of 8.5 (figure 13). Results are visible on figures 14 and 15.
The last update was about making all films from 2012 Fantasy films. like the others, here we have 5 films from 2012, and all of them has been turned into Fantasy movies.

```
 Part 2 : Updating Data

FilmName : Crazy/Beautiful  | Genre : Drama | Year : 2001 | Rating : 6.4 | ID : 8a6f2cbf-3f4f-4e54-b15f-1ff51a978604
FilmName : Crimson Tide  | Genre : Drama | Year : 1995 | Rating : 7.3 | ID : 58f12446-68c1-401b-b7e5-b62cfb91e3ba
FilmName : Me and Orson Welles  | Genre : Drama | Year : 2008 | Rating : 6.7 | ID : 7778e123-3149-412b-9ffc-d45f880c360d
FilmName : Mask of Fu Manchu, The  | Genre : Adventure | Year : 1932 | Rating : 6.3 | ID : b555cb39-2a3f-4d2b-8d6c-20ea2394bcd3
FilmName : Woman in Black, The  | Genre : Fantasy | Year : 2012 | Rating : 6.4 | ID : f0a7946c-4f3a-41bb-91d3-428cab4405e7
FilmName : Home Alone: The Holiday Heist  | Genre : Fantasy | Year : 2012 | Rating : 3.5 | ID : 78c50b35-0f94-4d17-9328-838e86f311f6
FilmName : Sada  | Genre : Drama | Year : 1998 | Rating : 6.6 | ID : 5a1cad71-8645-4a2e-9bb1-e656dfc56f8a
FilmName : Blind  | Genre : Drama | Year : 2016 | Rating : 5.8 | ID : 4ccb8f2a-5509-4665-8a5e-e2c39eb6b142
FilmName : Digging Up the Marrow  | Genre : Comedy | Year : 2014 | Rating : 5.8 | ID : cbaa64d2-fff4-43ae-a227-1c8510daf4cd
FilmName : Traviata, La  | Genre : Musical | Year : 1982 | Rating : 7.2 | ID : c90ed615-f075-4e80-885c-db18d6b40d3f
FilmName : Heathers  | Genre : Comedy | Year : 1988 | Rating : 7.2 | ID : 16a4fee7-18b6-438c-bbde-d4cd8e041749
FilmName : Fright Night Part II  | Genre : Comedy | Year : 1988 | Rating : 5.9 | ID : 4249a354-5180-4b9b-8154-da9f33f8a22d
FilmName : Hawks and Sparrows (Uccellacci e Uccellini)  | Genre : Comedy | Year : 1966 | Rating : 7.3 | ID : 6e43c104-1ce8-4d8d-95f1-fc2a27569ef6
FilmName : Kind Lady  | Genre : Crime | Year : 1951 | Rating : 7.0 | ID : ee8fef42-3e58-4bf6-9c88-c1dd31a32c03
FilmName : Decoy Bride, The  | Genre : Comedy | Year : 2011 | Rating : 6.2 | ID : 57916454-cb4b-4272-bb2f-dbdd10659c4a
FilmName : Daughter of Dr. Jeckyll  | Genre : Comedy | Year : 1972 | Rating : 4.8 | ID : 22900d0f-07c3-4ee2-8d78-6d0b8f6b136c
FilmName : Hunter, The  | Genre : Thriller | Year : 2011 | Rating : 6.7 | ID : 44f0a751-3d4e-4f1a-acc2-5e11cd7a4386
FilmName : Annapolis  | Genre : Drama | Year : 2006 | Rating : 5.9 | ID : 93e82922-2cea-43ba-a364-d053602c4b8b
FilmName : 51 Birch Street  | Genre : Documentary | Year : 2005 | Rating : 7.2 | ID : 48d378ce-5366-4e71-9b45-546d795df074
FilmName : Beyond Tomorrow (Beyond Christmas)  | Genre : Romance | Year : 1940 | Rating : 6.5 | ID : 76dd8c26-95bb-4ccd-a9d4-1616fb9cad93
FilmName : Mondo Trasho  | Genre : Comedy | Year : 1969 | Rating : 6.2 | ID : d674e26e-74f8-4041-905e-fc2441b34cd5
FilmName : Palo Alto  | Genre : Drama | Year : 2013 | Rating : 6.2 | ID : 26bfcd3a-df01-4043-8ecf-26e419ec140d
FilmName : Motel, The  | Genre : Comedy | Year : 2005 | Rating : 6.7 | ID : 63c27f28-abda-44ed-a804-b6083623e063
FilmName : TerrorStorm: A History of Government-Sponsored Terrorism  | Genre : Documentary | Year : 2003 | Rating : 6.8 | ID : 6536c073-f45d-426b-879b-4e459aecdcf6
FilmName : Nothing About Robert (Rien sur Robert)  | Genre : Comedy | Year : 1999 | Rating : 6.4 | ID : 6aa92e69-e1f4-4f63-9090-fe0858e4b9dc
FilmName : Sky Murder  | Genre : Mystery | Year : 1940 | Rating : 6.0 | ID : bdb21623-cf0b-4d8c-a3b3-1a8ac1216d44
FilmName : Very Brady Sequel, A  | Genre : Comedy | Year : 1996 | Rating : 5.8 | ID : ddf75622-e966-4ad6-8f96-71f46c24afcd
FilmName : Must Have Been Love  | Genre : Fantasy | Year : 2012 | Rating : 5.7 | ID : 4ac81d97-3b9c-48fb-8a22-2933d572a3dd
FilmName : Ninja Scroll (Jûbei ninpûchô)  | Genre : Animation | Year : 1993 | Rating : 8.5 | ID : 033ba779-55f5-448b-9b22-4852087b2654
FilmName : Germany Pale Mother  | Genre : War | Year : 1980 | Rating : 7.2 | ID : e4197id1-e373-4db8-99cc-965f1d11908
FilmName : La Boucherie Martin  | Genre : Romance | Year : 2004 | Rating : 6.6 | ID : 1219e67a-bfe3-4893-8a0f-ddd09a6fffff
FilmName : Pear Tree, The (Derakhte Golabi)  | Genre : Drama | Year : 1998 | Rating : 7.5 | ID : cb017b0c-4c56-4821-afb4-2a0cd4ff211e
FilmName : S.W.A.T.  | Genre : Action | Year : 2017 | Rating : 7.1 | ID : ba1ff35c-2273-4807-807a-17d0f5f0816b
FilmName : Brooklyn Castle  | Genre : Fantasy | Year : 2012 | Rating : 7.2 | ID : b82a9c6a-61d5-40c4-a3e4-2039b009373d
FilmName : K-PAX  | Genre : Sci-Fi | Year : 2001 | Rating : 7.4 | ID : 3e2e3475-ad10-465c-9521-c3e880220c8a
FilmName : In Tranzit  | Genre : War | Year : 2008 | Rating : 5.8 | ID : 1d90495e-816f-47e6-967f-4622bb1ca166
FilmName : Vampire Hunter D: Bloodlust (Banpaia hantâ D)  | Genre : Animation | Year : 2000 | Rating : 8.5 | ID : a455911f-0032-4547-a000-a907cc631087
FilmName : Ninja Hitler Contre Les Vikings De l Espace  | Genre : Sci-Fi | Year : 2016 | Rating : 9.6 | ID : 8f6a8cd7-107f-4070-aa90-86b514ebe674
FilmName : Last Hangman, The (Pierrepoint)  | Genre : Drama | Year : 2005 | Rating : 7.4 | ID : 0490690a-361b-4ad7-9603-c771224f3dc1
FilmName : Capricious Summer (Rozmarné léto)  | Genre : Comedy | Year : 1968 | Rating : 7.0 | ID : c7e5cd3c-456b-438a-ba5c-01099741862a
FilmName : Mad Dogs & Englishmen  | Genre : Musical | Year : 2021 | Rating : 5.0 | ID : dca08528-b960-4551-acbe-c6de8a27e06d
FilmName : Crazy Heart  | Genre : Romance | Year : 2009 | Rating : 7.2 | ID : 23b1b0c7-318c-4801-9e7e-3ff31d7b7945
FilmName : Gas, Food, Lodging  | Genre : Romance | Year : 1992 | Rating : 6.6 | ID : bc17ba61-32d8-4d7d-96f7-5317e9763a6c
FilmName : Heaven & Earth  | Genre : Action | Year : 1990 | Rating : 7.0 | ID : 17fe7604-2c2b-4fe7-9bbf-6444fa8a5813
FilmName : Normal People  | Genre : Drama | Year : 2020 | Rating : 8.5 | ID : 2d6b8bee-0da3-442c-9f4b-cfd77047a63c
FilmName : Lady in Red, The  | Genre : Action | Year : 1979 | Rating : 6.2 | ID : 8a111a1d-32f5-4975-afa4-6013d2f533a8
FilmName : Resistance  | Genre : Drama | Year : 2019 | Rating : 7.4 | ID : 36bdfcc2-0dda-4de6-9d34-a554c1cf1b14
FilmName : Secret Sunshine (Milyang)  | Genre : Drama | Year : 2007 | Rating : 7.5 | ID : 60e5880f-75be-4f39-bca6-f23654fb6697
FilmName : Look Back in Anger  | Genre : Drama | Year : 1959 | Rating : 7.0 | ID : ed680abd-7475-43fd-a990-425baa2519fb
FilmName : Orgy of the Dead  | Genre : Comedy | Year : 1965 | Rating : 3.0 | ID : 1dac383e-4ad5-427f-9407-01b7feaca9b5
```

Figure 14: Modified database part 1

As we can see, we can choose the condition that we want and modify it the way we like. It is possible to modify 1 or more records at the same time, if they all have a common data.

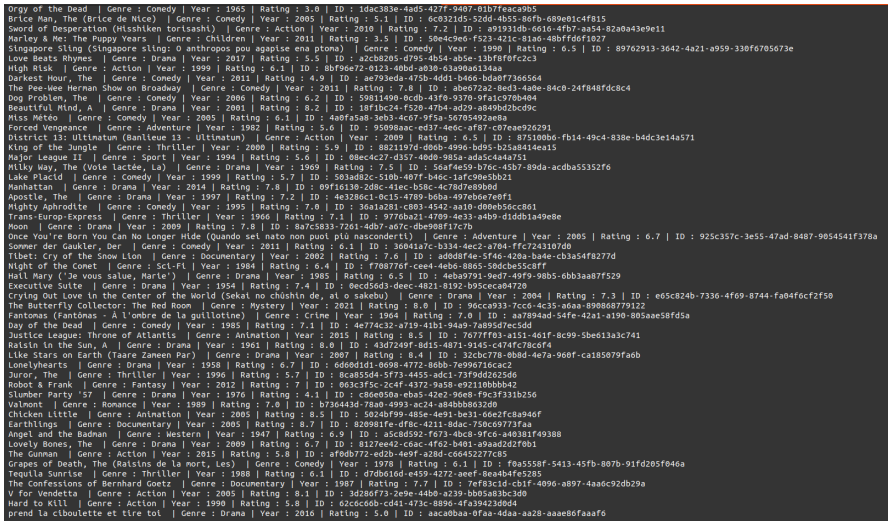

Figure 15: Modified database part 2

3.5 Deleting records

It's time to erase some data. So, we can start by creating a method to delete records with a condition. That is how the DATA_Eraser has been made (figure 16). Then, i will delete 10 records, bases on different conditions.
The first film that has been erased is the one named *the Motel*. Then, because i don't appreciate thrillers movies, i will erase all of them. This action erased 5 records. I can also erase films using their Rating as a condition. For instance, with the rating 5.9, i erased 3 records. The last record that erased has been a chosen one, because i've put the exact ID of this film, that of course cannot correspond to another record (figure 17). All those erased data can't be seen in figures 18 and 19 because of course, they have been erased. but we can see that the data list have been lightened.

```
def Data_Eraser(self, Table : str , Modif_Condition : str):

    self.cur.execute("DELETE FROM " + Table + " WHERE " + Modif_Condition)
    self.db.commit()
```

Figure 16: creating the method Data_Eraser

```
print("\n Part 3 : Deleting Data \n")
#deleting 1 film by name
Test_DB.Data_Eraser("mooovies","FilmName = 'Motel, The'")
#deleting by genre (5 movies)
Test_DB.Data_Eraser("mooovies","Genre = 'Thriller'")
#deleting by raitng (3 movies)
Test_DB.Data_Eraser("mooovies","Rating = 5.9")
#deleting by ID (1 movie)
Test_DB.Data_Eraser("mooovies","ID = '32cbc778-0b8d-4e7a-960f-ca185079fa6b'")
Test_DB.Print_Data_Table("*","mooovies")
```

Figure 17: Using the data eraser in different ways through the main program

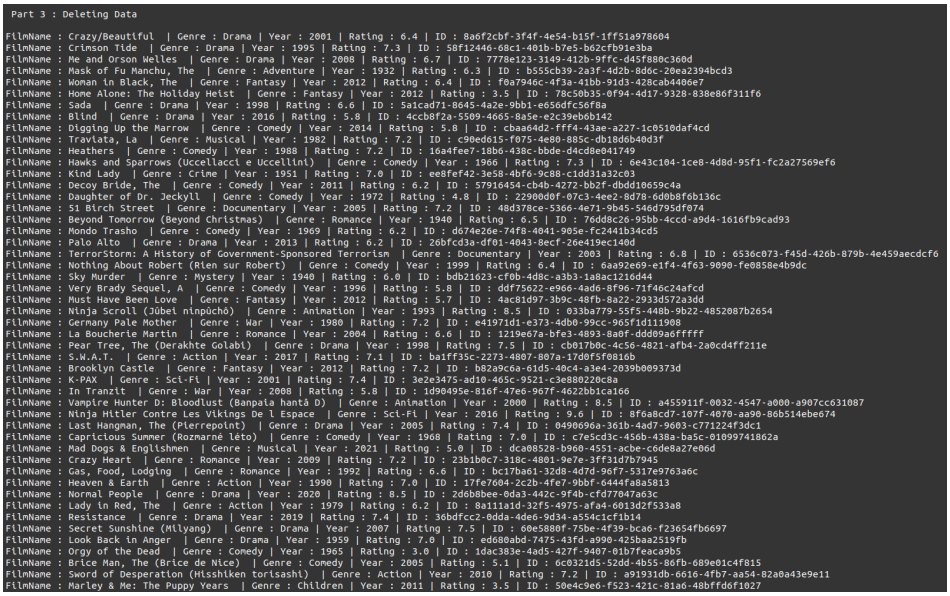

Figure 18: lightened database part 1

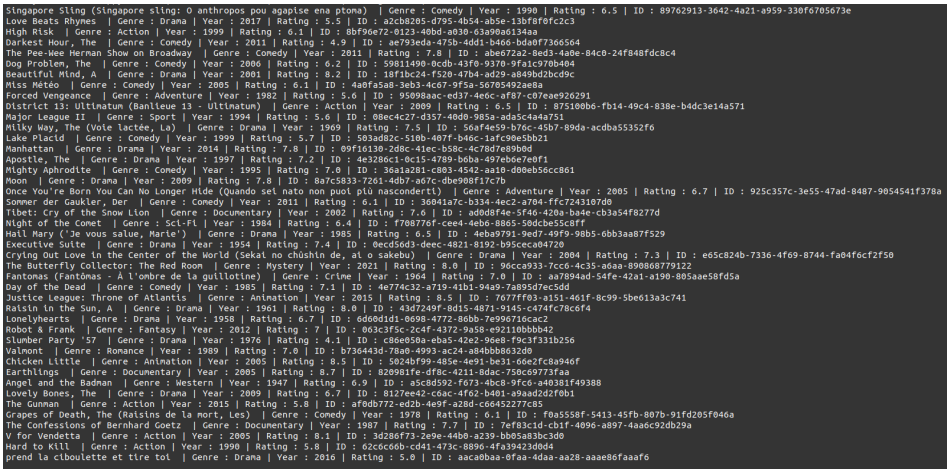

Figure 19: lightened database part 2

4 Conclusion

This Homework, i learned how to manipulate data inside a database using Python and the sqlite3 module. Moreover, i learned how to convert a .sql file to a .db file using Python. That way i will be harder to modify data that are stored in the .db file. I prefer using python rather than Mysql because i will be able to reuse my Python code unlike a Mysql coding in the shell.

About the LaTeX Language, I made some progress. Now, i can readjust images the way i like, so the final result looks cleaner. However, I'm not going to change my mind: LaTeX makes me lose time compared to other software like Word.