SURNAME:	NAME:	DNI:	GRADE:
COURSE: Databases	DATE: 27/03/19	GROUP:	

Exercise 2: [LO 1,2,3,4,5] 5.5 points

Given the following database:

```
CREATE TABLE Songs (
ID INTEGER PRIMARY KEY AUTOINCREMENT,
Name TEXT,
Artist INTEGER,
FOREIGN KEY(Artist) REFERENCES Artists(ID)
);
INSERT INTO Songs (ID,Name,Artist) VALUES
(1,'AZUL',1),
(2,'NUBES',2),
(3,'ROPAJA',2),
(4,'CALORRO',2),
(5,'HIGHWAY TO HELL',3),
(6,'FLOR DE LOTO',4);

CREATE TABLE Artists (
ID INTEGER PRIMARY KEY AUTOINCREMENT,
Email TEXT UNIQUE,
Birthdate INTEGER,
Name TEXT
);
INSERT INTO Artists (ID,Email,Birthdate,Name) VALUES
(1,'s@dalma.com','1/1/1960','SERGIO DALMA'),
(2,'est@pa.com','1/2/1955','ESTOPA'),
(3,'AC@dc.com','30/04/1940','AC/DC'),
(4,'HE@es.com','2/2/1970','HEROES DEL SILENCIO');

CREATE TABLE Album (
ID INTEGER PRIMARY KEY AUTOINCREMENT,
Title TEXT
);
INSERT INTO Album (ID,Title) VALUES
(1,'MIX'),
(2,'ESTOPA'),
(3,'MIX VARIOS');
CREATE TABLE Album_Songs (
ID_ALBUM INTEGER,
ID_SONG INTEGER,
PRIMARY KEY(ID_ALBUM,ID_SONG)
);
INSERT INTO Album_Songs (ID_ALBUM,ID_SONG) VALUES
(1,1), (1,2), (2,2),
(2,3), (3,4), (3,5),
(3,6);
```

SURNAME:	NAME:	DNI:	GRADE:
COURSE: Databases	DATE: 27/03/19	GROUP:	

Write down the SQL sentences that give the following results. If the result is a table, draw it and put the results inside.**(0.5 points per sentence)**:

1. The title and number of songs of every album.
2. The name of every song and the name of the artist that performs it. Order the results alphabetically by the artist name.
3. The name of every album and the number of artists that have at least one song in it.
4. Deletes the songs of every artist that have both an "A" and an "E" in their name.
5. Increases the birthdate of the oldest artist by 10 years.
6. Creates a view with all the songs performed by 'ESTOPA'.

Write down the Java code that you need to represent this database **(1 point)**. You don't need to implement the methods.

Write down the following Java methods, part of a *DatabaseManager* class which provides you with already initialized *Connection c* object, that you can freely use. **(0.75 points per method)**:

1. Using JDBC, a *searchAlbumsBySong(Song s)* which searchs into the database all the albums that contains a given song and returns a list of them. This method should be immune to SQL injection or similar attacks causes by a not sanitized input.
2. Using JDBC, a *createAlbum(String title, List<Song> songs)* which adds a new album to the database, including all the songs passed as parameters. The database should reflect the fact that the songs are part of the album. Assume that none of the songs are already in the database. You can create a helper method if you want. This whole insertion process needs to be part of one transaction.