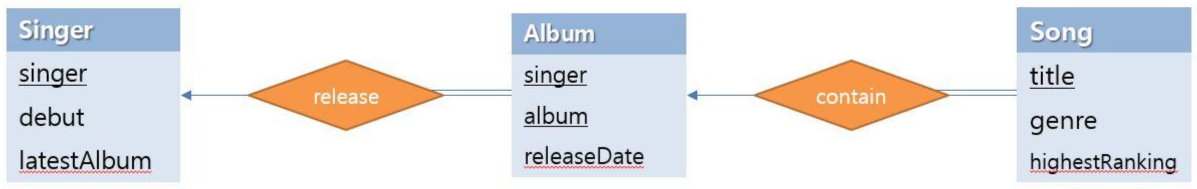
Database

project

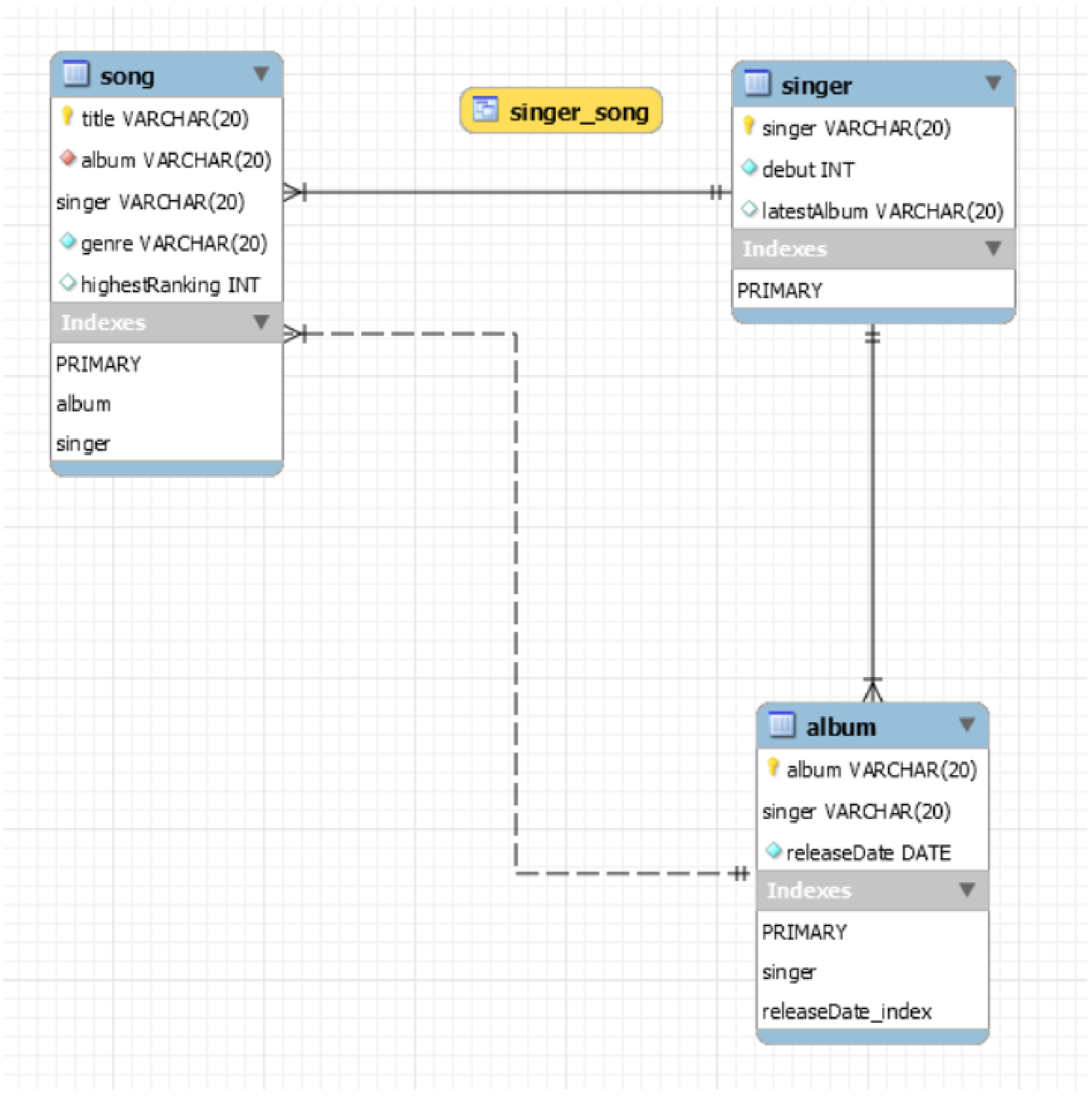
1871007

Sowon Kim

(1) ER Diagram of the database design. Draw Entity and Relations based on lecture content. (May use tools if needed. Mostly you can use just Powerpoint.)



(2) Database schema diagram. Shows tables in database as graphical tables. Details of the tables and columns and relationships are shown. May be done by drawing tool or use of programs (MS Visio, MySQL Workbench)



(3) Class and method explanation of Java codes (may use Javadoc output).

Main.java is used to execute the program. Menu occurs repeatedly until user gives an input of "6". If input value is 1, print every information about the database. Doing this, we need printAllalbum(), printAllSinger(), printAllSong() method, which is contained in other .java file.

If input value is 2, insert a new singer into singer table. To do this, we nned insertNewSinger() metod from Singer class.

If input value is 3, update the information. We can update the son's highest ranking and album's name. we use updateSongRankin() method from song.java file. Otherwise, code to update the name of album is in Main.java file. get an album name, singer name, new album name from user. And change album name to new album name.

If input value is 4, we need to delete song. We need deleteSong() method from Song class..

If input value is 5, we need to print some information that is fit into the condition. There're conditions about genre, debut year, and title. To do the search using genre, we need searchGenre() from Song class. To do the search using debut year, we use searchYear() method from Song.java file. However, we don't need other file when searching using song title. We get the title of the song, and print out the information about the song.

If input value is 6, we finish the program.

Album.java is used when we do some works on the album table. There're three attributes of the class which is column of the table.

printAllAlbum() method, gets Connection, Statement and ResultSet parameter. Using this, it print out all the information that is in the album table.

Singer.java is used when we do some works on Singer table. There're three attributes of the class which is column of Singer table.

printAllSinger() method gets Connection, Statement, and ResultSet parameter. Using this, it print out all the information that is in the Singer table.

insertNewSinger() method also gets Connection, Statement, and ResultSet parameter. It gets an input value from user about new singer, her debut year, and optionally lattestAlbum. It insert the value on the Singer table and print out the Singer table if the user wants.

Song.java is used when we do some works on Song table. There're five attributes of the Song class which is also column of Song table.

printAllSong() method gets Connection, Statement, and ResultSet parameter. Using this, it prints out all the information that is in the Song table.

updateSongRanking() method also gets Connection, Statement, and ResultSet parameter. It gets an input value from user about song title, singer name, and highest ranking. Then, update the table with these values. After doing update, it prints out all of the song if the user wants.

deleteSong() method gets Connection, Statement, and ResultSet parameter. It gets an input from user about song title, and singer. Then, delete the value using it. After the deletion, it prints all values of Song if the user wants.

searchGenre() method gets Connection, Statement, and ResultSet parameter. It gets a genre data from user, and search the song using this value. After searching data, it prints out all the result values.

searchYear() method gets Connection, Statement, and ResultSet parameter. It gets a year data from user, and search the song using this value. When searching it, it uses join and nested query. After searching data, it prints out all the result values.

(4) Provide Main class name and how to run, connection configuration instructions.

For the connection configuration, reference above Java codes (3) database connection information. If you have any kind of special environment or configuration such as using Mac, using another port number for MySQL, etc., you should note all of those things in here.

Main class is Main.java. Run Main.java to execute the program.

UserID is dbuser, passwd is dbpwd, and name of database is dbprj.

I used 'localhost:3306'

Below is the related code.

```
String userID="dbuser";
String userPW="dbpwd";
String dbName="dbprj";
String url="jdbc:mysql://localhost:3306/"+dbName+"?&serverTimezone=UTC";
myConn=DriverManager.getConnection(url,userID, userPW);
```

(5) Show with detail explanation that the above 16 requirements have been satisfied. Related parts of codes should be shown, and if user interface is involved the screen captures should also be shown.

[DB Schema]

(1) Should have at least 3 tables with each table having at least 3 columns The 3 tables should have at least 3 columns, and others may have any number of columns.

I have three tables (album, singer, song). Album has album, singer, releaseDate column, Singer has singer, debut, latestAlbum column and Song has title, album, singer, genre, highestRanking column.

```
create table Singer( #information about singer
singer varchar(20) not null, #name of singer
debut int not null, #debut year of the singer
latestAlbum varchar(20), #the latest album of the singer
primary key (singer)); #pk is singer

create table Album( #information about the Album
album varchar(20) not null, #name of album
singer varchar(20) not null, #name of the singer
releaseDate date not null, #the release date of the album
primary key (album, singer), #pk is album and singer
#singer column in this table references singer in the Singer table
foreign key (singer) references Singer (singer));

create table Song( #information about the song
title varchar(20) not null, #title of the song
album varchar(20) not null, #the album that the song is included
singer varchar(20) not null, #the singer of the song
genre varchar(20) not null, #genre of the song
#highest ranking of the song
#null value when it was never in the top 10 or don’t know the highest ranking
highestRanking int,
primary key (title), #pk is title
#if album value of the Album table changes, then the album value of Song table also changes
foreign key (album) references Album(album) on update cascade,
foreign key (singer) references Singer(singer));
```

(2) Should have at least 30 records inserted for initialization (total records for all tables) For example, A table may have 5 records, B table 10 records, and C table 15 records

Album has 11 records, Singer has 5 records, and song has 17 records.

```
#insert value in the Singer table
insert into Singer (singer, debut, latestAlbum) values
('2NE1', 2009, 'CRUSH'),
('miss A', 2010, 'Colors'),
('OH MY GIRL', 2015, 'NONSTOP'),
('Apink', 2011, 'LOOK'),
('ITZY', 2019, 'ITz ME');

#insert value in the Album table
insert into Album (album, singer, releaseDate) values
('CRUSH', '2NE1', '2014-02-27'),
('Falling In Love', '2NE1', '2013-07-08'),
('2NE1 2nd Mini Album', '2NE1', '2011-07-28'),
('Colors', 'miss A', '2015-03-30'),
('Love Alone', 'miss A', '2011-05-02'),
('LOOK', 'Apink', '2020-04-13'),
('Always', 'Apink', '2017-04-19'),
('Mr. Chu', 'Apink', '2015-02-18'),
('ITz ME', 'ITZY', '2020-03-09'),
('ITz Different', 'ITZY', '2019-02-12'),
('NONSTOP', 'OH MY GIRL', '2020-04-27');

#insert value in the Song table
insert into Song (title, album, singer, genre, highestRanking) values
('Come Back Home', 'CRUSH', '2NE1', 'dance', 2),
('Falling In Love', 'Falling In Love', '2NE1', 'dance', 1),
('hate you', '2NE1 2nd Mini Album', '2NE1', 'dance', 1),
('Ugly', '2NE1 2nd Mini Album', '2NE1', 'dance', 1),
('Love Song', 'Colors', 'miss A', 'dance', 66),
('I Caught Ya', 'Colors', 'miss A', 'R&B', 89),
('Stuck', 'Colors', 'miss A', 'R&B', null),
('Love Alone', 'Love Alone', 'miss A', 'dance', 70),
('Overwrite', 'LOOK', 'Apink', 'dance', 1),
('Be Myself', 'LOOK', 'Apink', 'dance', null),
('Always', 'Always', 'Apink', 'ballad', 30),
('Mr. Chu', 'Mr. Chu', 'Apink', 'dance', 1),
('Hush', 'Mr. Chu', 'Apink', 'dance', null),
('WANNABE', 'ITz ME', 'ITZY', 'dance', 1),
('24HRS', 'ITz ME', 'ITZY', 'ballad', 99),
('WANT IT?', 'ITz Different', 'ITZY', 'dance', 52),
('Dolphin', 'NONSTOP', 'OH MY GIRL', 'ballad', 76);
```

(3) Should include primary key in every table, and also foreign key, not null constraints should exist in some tables

Album's primary key is (album, singer). Singer's primary key is (singer). Song's primary key is (title).

Album(releaseDate), Song(album), Song(singer), Song(genre) have not null constranints.

Album(Singer) references Singer(singer), Song(album) references Album(album), and Song(singer) references Singer(singer).

```
primary key (singer));

primary key (album, singer), #pk is album and singer
#singer column in this table references singer in the Singer table
foreign key (singer) references Singer (singer));

primary key (title), #pkis title
#if album value of the Album table changes, then the album value of Song table also changes
foreign key (album) references Album(album) on update cascade,
foreign key (singer) references Singer(singer));
```

(4) Tables should be in 3 rd Normal Form (3NF)

Every non-prime attribute of table is dependent on primary key. And there is no case that a non-prime attribute is determined by another non-prime attributes.

Album:

<u>Album</u>	<u>singer</u>	releaseDate

Singer:

<u>Singer</u>	debut	latestAlbum

Song:

<u>title</u>	album	singer	genre	highestRanking

```
create table Singer( #information about singer
singer varchar(20) not null, #name of singer
debut int not null, #debut year of the singer
latestAlbum varchar(20), #the latest album of the singer
primary key (singer)); #pk is singer

create table Album( #information about the Album
album varchar(20) not null, #name of album
singer varchar(20) not null, #name of the singer
releaseDate date not null, #the release date of the album
primary key (album, singer), #pk is album and singer
#singer column in this table references singer in the Singer table
foreign key (singer) references Singer (singer));

create table Song( #information about the song
title varchar(20) not null, #title of the song
album varchar(20) not null, #the album that the song is included
singer varchar(20) not null, #the singer of the song
genre varchar(20) not null, #genre of the song
#highest ranking of the song
#null value when it was never in the top 10 or don't know the highest ranking
highestRanking int,
primary key (title), #pk is title
#if album value of the Album table changes, then the album value of Song table also changes
foreign key (album) references Album(album) on update cascade,
foreign key (singer) references Singer(singer));
```

(5) At least 1 index should be defined on the tables (The primary key(PK) columns have index automatically created, so do not create an index on a PK)

I have index that point the album table's releaseDate

```
create index releaseDate_index on Album(releaseDate);
```

(6) 1 view should be defined, and the view should be defined using at least two other tables

I created a view named singer_song using Song, Album, Singer tables.

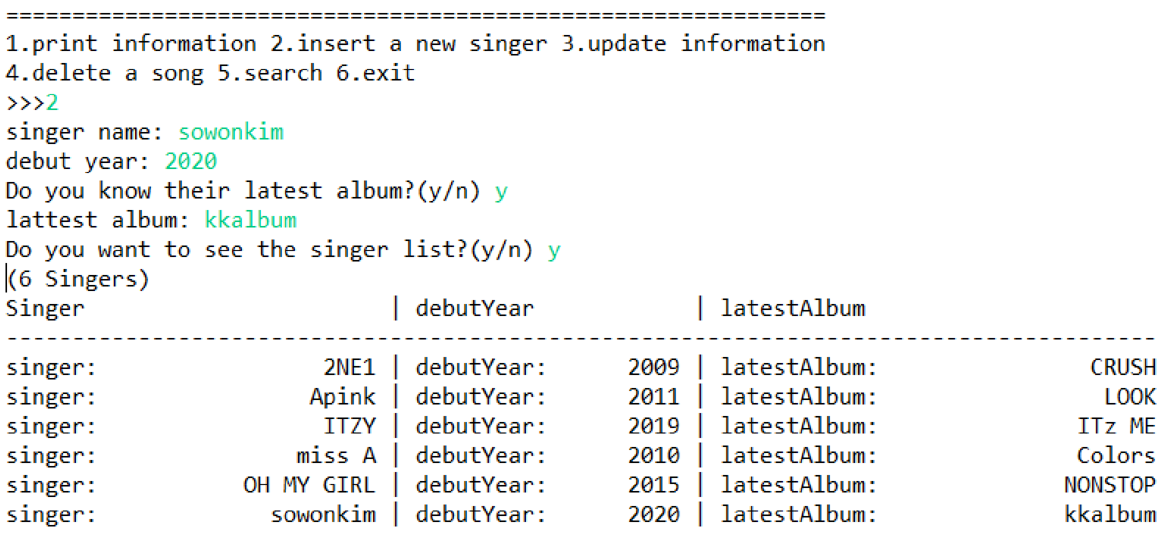
```
create view singer_song as
```

```
select Song.title, Song.singer, Album.releaseDate, Singer.debut
from Song, Album, Singer
where Song.album=Album.album and
Song.singer=Album.singer and
Album.singer=Singer.singer;
```

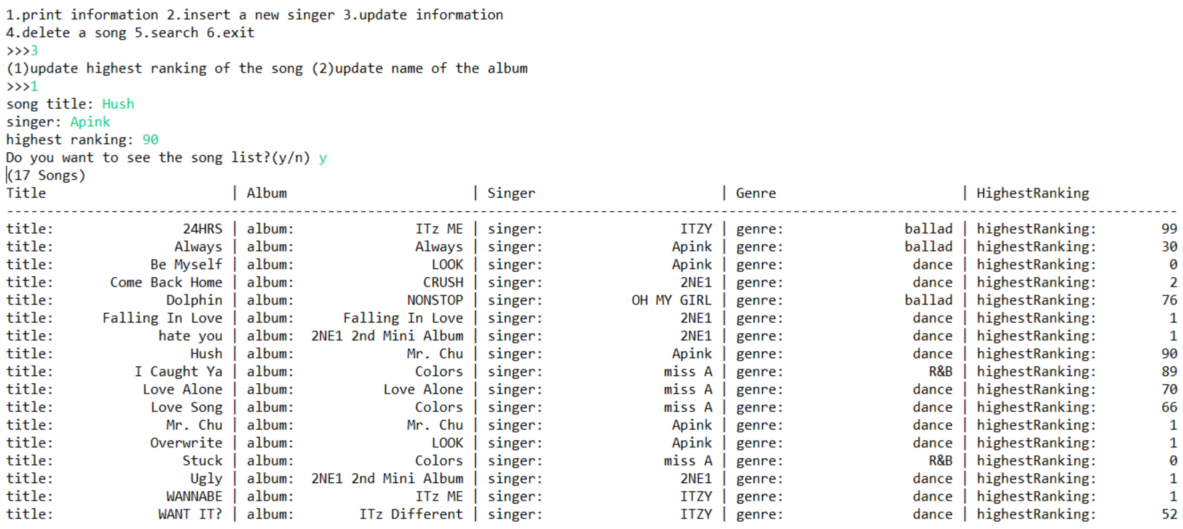
[DB Queries and Program]

(7) All queries (in 8 to 14 below) should have parameterized variables. In other words, the program asks for input value from the user and creates a query using the user input value. For example, the user may give the singer name and this value will be plugged into the SELECT query. (Hint: use PreparedStatement)

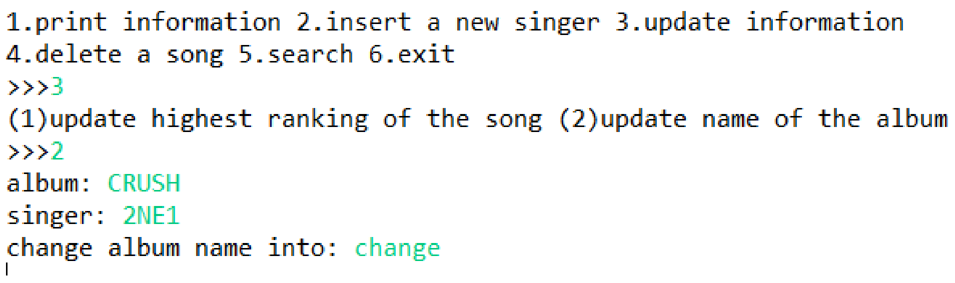
All queries have parameterized variables. Below are the screen captures that show the program asks for input value.



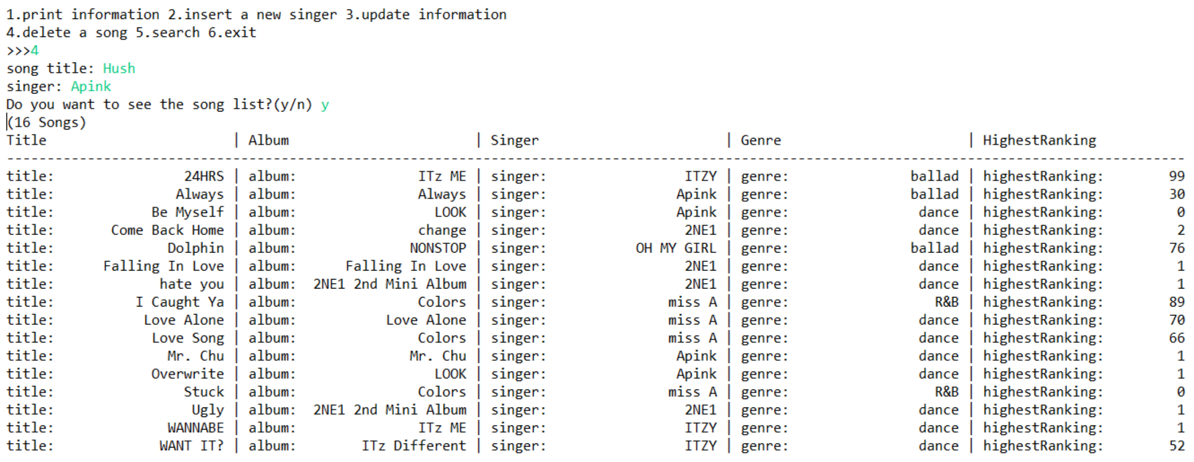
```
===============================================================
1.print information 2.insert a new singer 3.update information
4.delete a song 5.search 6.exit
>>>2
singer name: sowonkim
debut year: 2020
Do you know their latest album?(y/n) y
lattest album: kkalbum
Do you want to see the singer list?(y/n) y
(6 Singers)
Singer                         | debutYear              | latestAlbum
---------------------------------------------------------------------------------------
singer:                   2NE1 | debutYear:       2009 | latestAlbum:             CRUSH
singer:                  Apink | debutYear:       2011 | latestAlbum:              LOOK
singer:                   ITZY | debutYear:       2019 | latestAlbum:            ITz ME
singer:                 miss A | debutYear:       2010 | latestAlbum:            Colors
singer:             OH MY GIRL | debutYear:       2015 | latestAlbum:           NONSTOP
singer:               sowonkim | debutYear:       2020 | latestAlbum:           kkalbum
```



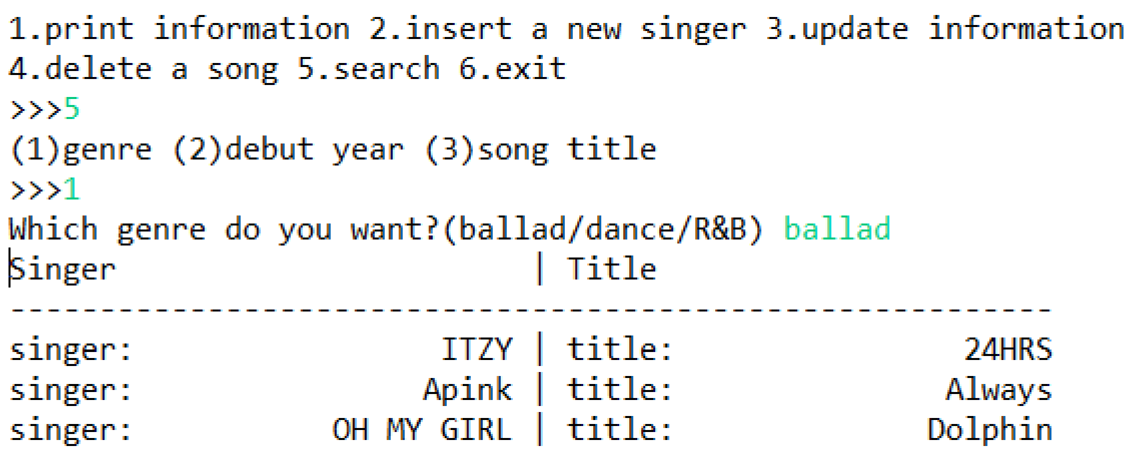
```
1.print information 2.insert a new singer 3.update information
4.delete a song 5.search 6.exit
>>>3
(1)update highest ranking of the song (2)update name of the album
>>>1
song title: Hush
singer: Apink
highest ranking: 90
Do you want to see the song list?(y/n) y
(17 Songs)
Title                        | Album                        | Singer                        | Genre                         | HighestRanking
------------------------------------------------------------------------------------------------------------------------------------------------
title:                 24HRS | album:                ITz ME | singer:                  ITZY | genre:                 ballad | highestRanking:      99
title:                Always | album:                Always | singer:                 Apink | genre:                 ballad | highestRanking:      30
title:             Be Myself | album:                  LOOK | singer:                 Apink | genre:                  dance | highestRanking:       0
title:        Come Back Home | album:                 CRUSH | singer:                  2NE1 | genre:                  dance | highestRanking:       2
title:               Dolphin | album:               NONSTOP | singer:            OH MY GIRL | genre:                 ballad | highestRanking:      76
title:       Falling In Love | album:       Falling In Love | singer:                  2NE1 | genre:                  dance | highestRanking:       1
title:              hate you | album:   2NE1 2nd Mini Album | singer:                  2NE1 | genre:                  dance | highestRanking:       1
title:                  Hush | album:               Mr. Chu | singer:                 Apink | genre:                  dance | highestRanking:      90
title:           I Caught Ya | album:                Colors | singer:                miss A | genre:                    R&B | highestRanking:      89
title:            Love Alone | album:            Love Alone | singer:                miss A | genre:                  dance | highestRanking:      70
title:             Love Song | album:                Colors | singer:                miss A | genre:                  dance | highestRanking:      66
title:               Mr. Chu | album:               Mr. Chu | singer:                 Apink | genre:                  dance | highestRanking:       1
title:             Overwrite | album:                  LOOK | singer:                 Apink | genre:                  dance | highestRanking:       1
title:                 Stuck | album:                Colors | singer:                miss A | genre:                    R&B | highestRanking:       0
title:                  Ugly | album:   2NE1 2nd Mini Album | singer:                  2NE1 | genre:                  dance | highestRanking:       1
title:               WANNABE | album:                ITz ME | singer:                  ITZY | genre:                  dance | highestRanking:       1
title:              WANT IT? | album:          ITz Different | singer:                  ITZY | genre:                  dance | highestRanking:      52
```



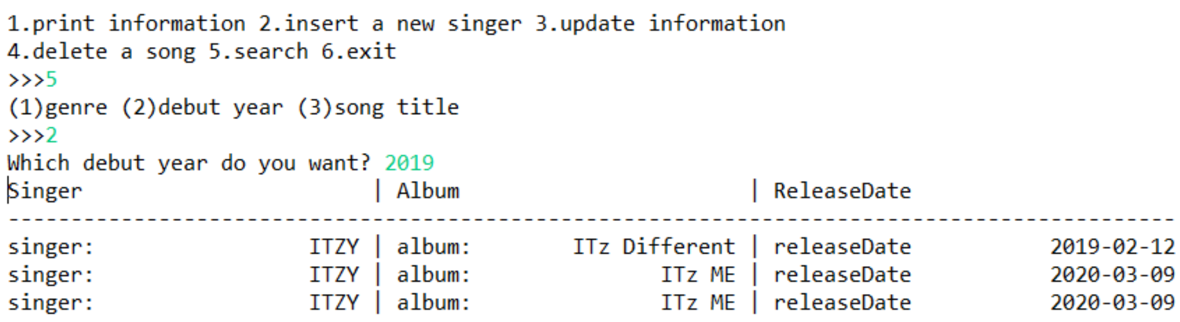
```
1.print information 2.insert a new singer 3.update information
4.delete a song 5.search 6.exit
>>>3
(1)update highest ranking of the song (2)update name of the album
>>>2
album: CRUSH
singer: 2NE1
change album name into: change
```



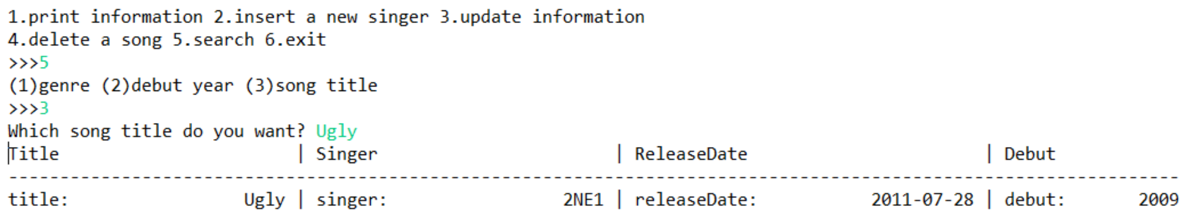
```
1.print information 2.insert a new singer 3.update information
4.delete a song 5.search 6.exit
>>>4
song title: Hush
singer: Apink
Do you want to see the song list?(y/n) y
(16 Songs)
Title                    | Album                    | Singer                   | Genre                    | HighestRanking
------------------------------------------------------------------------------------------------------------------------
title:             24HRS | album:             ITz ME | singer:             ITZY | genre:            ballad | highestRanking:       99
title:            Always | album:             Always | singer:            Apink | genre:            ballad | highestRanking:       30
title:         Be Myself | album:               LOOK | singer:            Apink | genre:             dance | highestRanking:        0
title:    Come Back Home | album:             change | singer:             2NE1 | genre:             dance | highestRanking:        2
title:           Dolphin | album:            NONSTOP | singer:       OH MY GIRL | genre:            ballad | highestRanking:       76
title:   Falling In Love | album:    Falling In Love | singer:             2NE1 | genre:             dance | highestRanking:        1
title:          hate you | album:  2NE1 2nd Mini Album | singer:           2NE1 | genre:             dance | highestRanking:        1
title:       I Caught Ya | album:             Colors | singer:           miss A | genre:               R&B | highestRanking:       89
title:        Love Alone | album:         Love Alone | singer:           miss A | genre:             dance | highestRanking:       70
title:         Love Song | album:             Colors | singer:           miss A | genre:             dance | highestRanking:       66
title:           Mr. Chu | album:            Mr. Chu | singer:            Apink | genre:             dance | highestRanking:        1
title:         Overwrite | album:               LOOK | singer:            Apink | genre:             dance | highestRanking:        1
title:             Stuck | album:             Colors | singer:           miss A | genre:               R&B | highestRanking:        0
title:              Ugly | album:  2NE1 2nd Mini Album | singer:           2NE1 | genre:             dance | highestRanking:        1
title:           WANNABE | album:             ITz ME | singer:             ITZY | genre:             dance | highestRanking:        1
title:          WANT IT? | album:      ITz Different | singer:             ITZY | genre:             dance | highestRanking:       52
```



```
1.print information 2.insert a new singer 3.update information
4.delete a song 5.search 6.exit
>>>5
(1)genre (2)debut year (3)song title
>>>1
Which genre do you want?(ballad/dance/R&B) ballad
Singer                   | Title
--------------------------------------------------
singer:             ITZY | title:            24HRS
singer:            Apink | title:           Always
singer:       OH MY GIRL | title:          Dolphin
```



```
1.print information 2.insert a new singer 3.update information
4.delete a song 5.search 6.exit
>>>5
(1)genre (2)debut year (3)song title
>>>2
Which debut year do you want? 2019
Singer                   | Album                    | ReleaseDate
-------------------------------------------------------------------------
singer:             ITZY | album:      ITz Different | releaseDate      2019-02-12
singer:             ITZY | album:             ITz ME | releaseDate      2020-03-09
singer:             ITZY | album:             ITz ME | releaseDate      2020-03-09
```



```
1.print information 2.insert a new singer 3.update information
4.delete a song 5.search 6.exit
>>>5
(1)genre (2)debut year (3)song title
>>>3
Which song title do you want? Ugly
Title                        | Singer                        | ReleaseDate                   | Debut
-------------------------------------------------------------------------------------------------------------
title:                  Ugly | singer:                  2NE1 | releaseDate:       2011-07-28 | debut:       2009
```

(8) Should have at least 1 interface (menu and user input) and query to insert into 1 table

I have new singer insertion, and it inset value into only Singer table.

```
        //insert a new singer into singer table
        public static void insertNewSinger(Connection myConn, Statement myState,
ResultSet myResSet) throws SQLException {
                Scanner scanner=new Scanner(System.in);
                String sql="";

                //insert a new singer into the singer table
                //we are going to get the value from the user
                sql="insert into Singer(singer, debut, latestAlbum)
values(?,?,?)";
                PreparedStatement ps=myConn.prepareStatement(sql);
                Singer singer=new Singer(); //make a new object to store the
input value
                //get an input of the new singer name
                System.out.print("singer name: ");
                singer.singer=scanner.nextLine();
                //get an input of the singer's debut year
                System.out.print("debut year: ");
                singer.debut=scanner.nextInt();
                //get an input of the singer's latest album (but it's optional)
                scanner=new Scanner(System.in);
                System.out.print("Do you know their latest album?(y/n) ");
                String ans=scanner.nextLine();

                if (ans.equals("y")) {//if user wants to insert the value of the
latest album
                        System.out.print("lattest album: ");
                        singer.latestAlbum=scanner.nextLine();
                        ps.setString(1, Singer.singer);
                        ps.setInt(2, singer.debut);
                        ps.setString(3, singer.latestAlbum);
                }
                else {//if user don't want know the value of the latest album
                        //fill this with null value
                        ps.setString(1, Singer.singer);
                        ps.setInt(2, singer.debut);
                        ps.setString(3, null);
                }
                ps.executeUpdate();

                //if user wants to get the value of changed singer table
                //print all
                System.out.print("Do you want to see the singer list?(y/n) ");
```

```
		ans=scanner.nextLine();
		if (ans.equals("y")) singer.printAllSinger(myConn, myState, 
myResSet);
		System.out.println();

	}
```

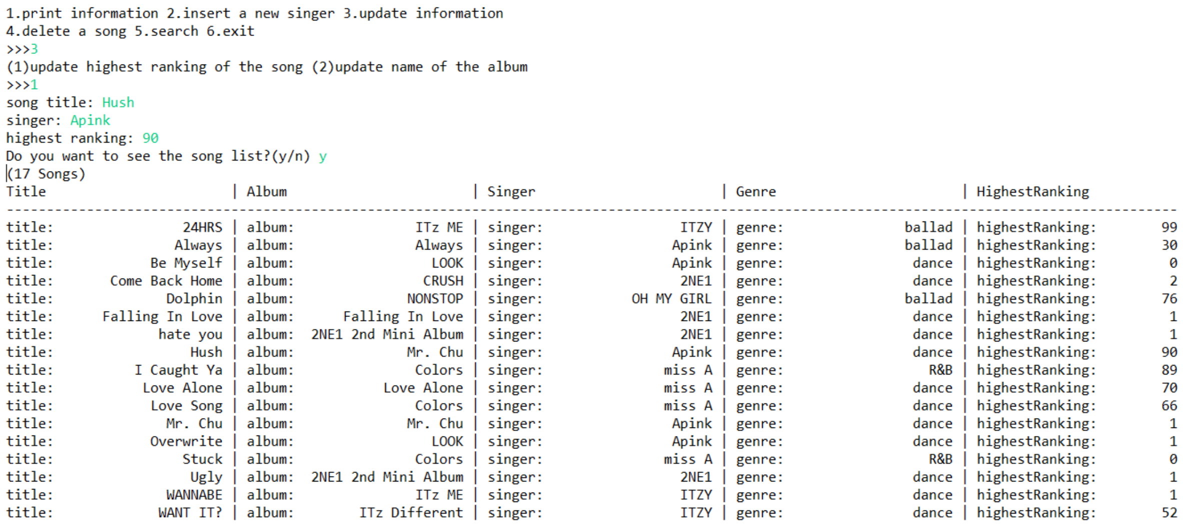
```
==============================================================
1.print information 2.insert a new singer 3.update information
4.delete a song 5.search 6.exit
>>>2
singer name: sowonkim
debut year: 2020
Do you know their latest album?(y/n) y
lattest album: kkalbum
Do you want to see the singer list?(y/n) y
(6 Singers)
Singer                      | debutYear            | latestAlbum
-------------------------------------------------------------------------------------
singer:                2NE1 | debutYear:      2009 | latestAlbum:               CRUSH
singer:               Apink | debutYear:      2011 | latestAlbum:                LOOK
singer:                ITZY | debutYear:      2019 | latestAlbum:              ITz ME
singer:              miss A | debutYear:      2010 | latestAlbum:              Colors
singer:          OH MY GIRL | debutYear:      2015 | latestAlbum:             NONSTOP
singer:            sowonkim | debutYear:      2020 | latestAlbum:             kkalbum
```

(9) Should have at least 1 interface (menu and user input) and query to update on 1 or 2 tables

Update highestRanking of the song on only Song table.

```
	//staic function that updates song's highest ranking
	public static void updateSongRanking(Connection myConn, Statement 
myState, ResultSet myResSet) throws SQLException {
		Song song=new Song();
		Scanner scanner=new Scanner(System.in);
		String sql="";
		//update the song's highest ranking with the input value
		sql="update Song set highestRanking=? where title=? and 
singer=?";
		PreparedStatement ps=myConn.prepareStatement(sql);
		//get the value of song title
		System.out.print("song title: ");
		song.title=scanner.nextLine();
		//get the value of singer
		System.out.print("singer: ");
		song.singer=scanner.nextLine();
		//get the value of highest ranking (change the table's highest 
ranking value into this)
		System.out.print("highest ranking: ");
		song.highestRanking=scanner.nextInt();
		scanner=new Scanner(System.in);
		ps.setInt(1,song.highestRanking);
		ps.setString(2, song.title);
		ps.setString(3, song.singer);
		//execute update
```

```java
               ps.executeUpdate();
               
               //if the user want to see the changed song list, print all
               System.out.print("Do you want to see the song list?(y/n) ");
               String ans=scanner.nextLine();
               if (ans.equals("y")) song.printAllSong(myConn, myState, 
myResSet);
               System.out.println();
        }
```

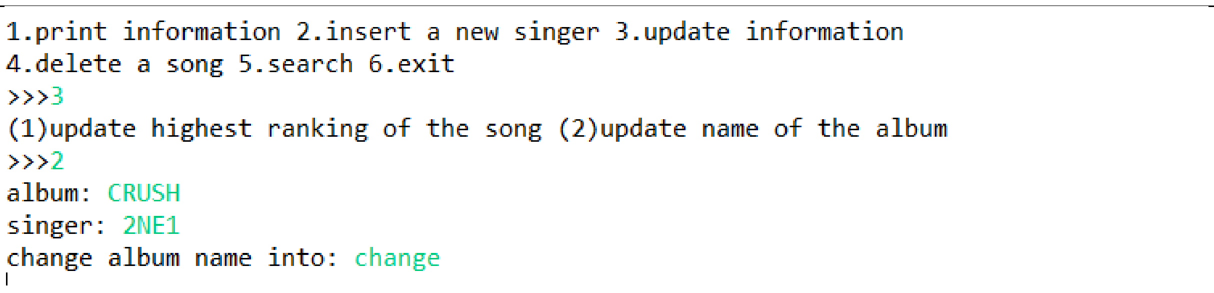


```
1.print information 2.insert a new singer 3.update information
4.delete a song 5.search 6.exit
>>>3
(1)update highest ranking of the song (2)update name of the album
>>>1
song title: Hush
singer: Apink
highest ranking: 90
Do you want to see the song list?(y/n) y
(17 Songs)
Title                   | Album                      | Singer                  | Genre                   | HighestRanking
----------------------------------------------------------------------------------------------------------------------------------
title:            24HRS | album:              ITz ME | singer:            ITZY | genre:           ballad | highestRanking:      99
title:           Always | album:              Always | singer:           Apink | genre:           ballad | highestRanking:      30
title:        Be Myself | album:                LOOK | singer:           Apink | genre:            dance | highestRanking:       0
title:   Come Back Home | album:               CRUSH | singer:            2NE1 | genre:            dance | highestRanking:       2
title:          Dolphin | album:             NONSTOP | singer:      OH MY GIRL | genre:           ballad | highestRanking:      76
title:  Falling In Love | album:     Falling In Love | singer:            2NE1 | genre:            dance | highestRanking:       1
title:         hate you | album: 2NE1 2nd Mini Album | singer:            2NE1 | genre:            dance | highestRanking:       1
title:             Hush | album:             Mr. Chu | singer:           Apink | genre:            dance | highestRanking:      90
title:      I Caught Ya | album:              Colors | singer:          miss A | genre:              R&B | highestRanking:      89
title:       Love Alone | album:          Love Alone | singer:          miss A | genre:            dance | highestRanking:      70
title:        Love Song | album:              Colors | singer:          miss A | genre:            dance | highestRanking:      66
title:          Mr. Chu | album:             Mr. Chu | singer:           Apink | genre:            dance | highestRanking:       1
title:        Overwrite | album:                LOOK | singer:           Apink | genre:            dance | highestRanking:       1
title:            Stuck | album:              Colors | singer:          miss A | genre:              R&B | highestRanking:       0
title:             Ugly | album: 2NE1 2nd Mini Album | singer:            2NE1 | genre:            dance | highestRanking:       1
title:          WANNABE | album:              ITz ME | singer:            ITZY | genre:            dance | highestRanking:       1
title:         WANT IT? | album:       ITz Different | singer:            ITZY | genre:            dance | highestRanking:      52
```

(10) One of the updates should occur on 2 tables by using transactions

Update name of album on Album table, and update the laestAlbum of singer on Singer table.

```java
                                   else if (subop==2){//update the name of album
                                          scanner=new Scanner(System.in);
                                          //get an input of album name that the user want to 
change
                                          System.out.print("album: ");
                                          album.album=scanner.nextLine();
                                          //get an input of singer name that the user want to 
change
                                          System.out.print("singer: ");
                                          album.singer=scanner.nextLine();
                                          //get an input of new name that the user want to 
change into
                                          System.out.print("change album name into: ");
                                          String newAlbumName=scanner.nextLine();
                                          try {//use transaction to update two tables at one 
time
                                                 //turn off the auto commit
                                                 myConn.setAutoCommit(false);
                                                 //update the album name at the album table
                                                 sql="update album set album=? where album=?
and singer=?";
                                                 PreparedStatement
ps=myConn.prepareStatement(sql);

                                                 ps.setString(1, newAlbumName);
                                                 ps.setString(2, album.album);
                                                 ps.setString(3, album.singer);
                                                 ps.executeUpdate();

                                                 //we need to update all the column that
contains album
                                                 //change the latest album name of the singer,
if the value is updated
                                                 sql="update singer set latestAlbum=? where
latestAlbum=? and singer=?";
                                                 ps=myConn.prepareStatement(sql);

                                                 ps.setString(1, newAlbumName);
                                                 ps.setString(2, album.album);
                                                 ps.setString(3, album.singer);
                                                 ps.executeUpdate();
                                                 //commit
                                                 myConn.commit();
                                          }
                                          catch(SQLException e) {
                                                 e.printStackTrace();
                                                 if (myConn!=null) {
                                                        try {//if commit didn't work, then
roll back

       System.out.println("Transaction is being rolled back");
                                                                  myConn.rollback();
                                                        }
                                                        catch(SQLException e1) {
                                                                  e1.printStackTrace();
                                                        }
                                                 }
                                          }
                                          finally {
                                                 //change auto commit into true
                                                 myConn.setAutoCommit(true);
                                          }
                                          System.out.println();
                                   }
```

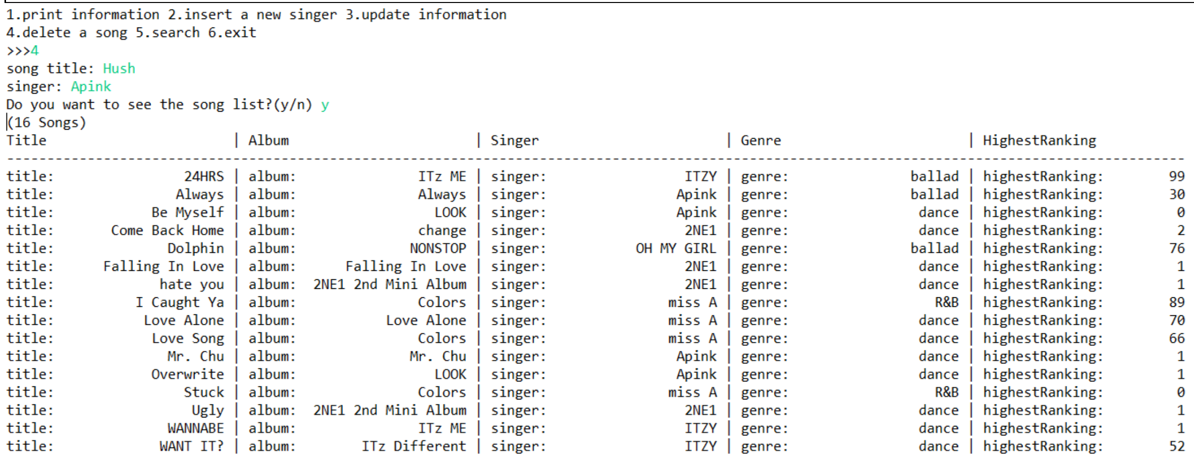


```
1.print information 2.insert a new singer 3.update information
4.delete a song 5.search 6.exit
>>>3
(1)update highest ranking of the song (2)update name of the album
>>>2
album: CRUSH
singer: 2NE1
change album name into: change
```

(11) Should have at least 1 interface (menu and user input) and queries to delete from 1 table

Delete song from Song table.

```
	//delete song
	public static void deleteSong(Connection myConn, Statement myState, 
ResultSet myResSet) throws SQLException {
		//make a song object and store the input data at it
		Song song=new Song();
		Scanner scanner=new Scanner(System.in);
		String sql="";
		
		//title that tue user want to delete
		System.out.print("song title: ");
		song.title=scanner.nextLine();
		//get an input about the singer who sings that song
		System.out.print("singer: ");
		song.singer=scanner.nextLine();
		
		//delete the song which has same value as the value of input 
value
		sql="delete from Song where title=? and singer=?";
		PreparedStatement ps=myConn.prepareStatement(sql);
		
		ps.setString(1, song.title);
		ps.setString(2, song.singer);
		ps.executeUpdate();
		
		//if the user want to see the changed song list, print all
		System.out.print("Do you want to see the song list?(y/n) ");
		String ans=scanner.nextLine();
		if (ans.equals("y")) song.printAllSong(myConn, myState, 
myResSet);
		System.out.println();
	}
```

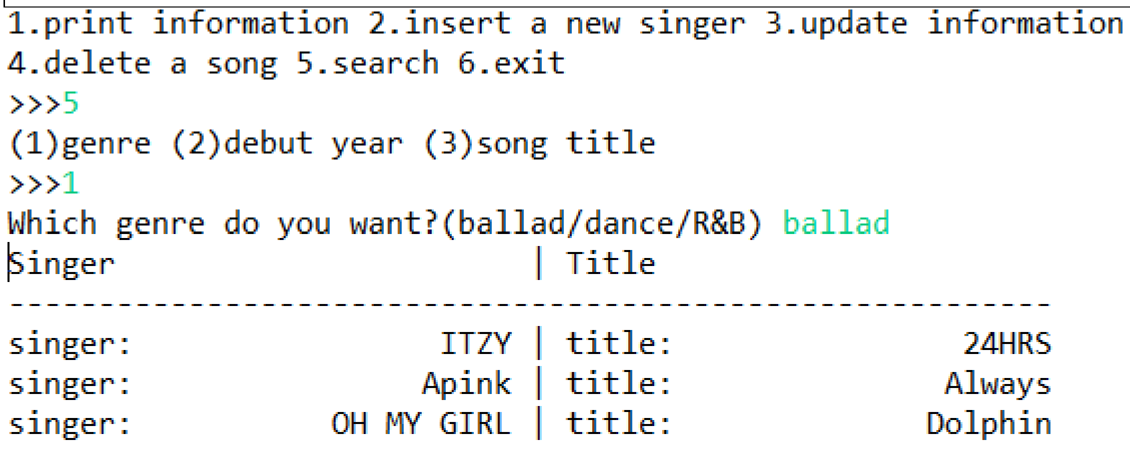


```
1.print information 2.insert a new singer 3.update information
4.delete a song 5.search 6.exit
>>>4
song title: Hush
singer: Apink
Do you want to see the song list?(y/n) y
(16 Songs)
Title                       | Album                       | Singer                       | Genre                       | HighestRanking
---------------------------------------------------------------------------------------------------------------------------------------------------

title:                24HRS | album:               ITz ME | singer:                 ITZY | genre:               ballad | highestRanking:        99
title:               Always | album:               Always | singer:                Apink | genre:               ballad | highestRanking:        30
title:            Be Myself | album:                 LOOK | singer:                Apink | genre:                dance | highestRanking:         0
title:       Come Back Home | album:               change | singer:                 2NE1 | genre:                dance | highestRanking:         2
title:              Dolphin | album:              NONSTOP | singer:           OH MY GIRL | genre:               ballad | highestRanking:        76
title:      Falling In Love | album:      Falling In Love | singer:                 2NE1 | genre:                dance | highestRanking:         1
title:             hate you | album:  2NE1 2nd Mini Album | singer:                 2NE1 | genre:                dance | highestRanking:         1
title:          I Caught Ya | album:               Colors | singer:               miss A | genre:                  R&B | highestRanking:        89
title:           Love Alone | album:           Love Alone | singer:               miss A | genre:                dance | highestRanking:        70
title:            Love Song | album:               Colors | singer:               miss A | genre:                dance | highestRanking:        66
title:              Mr. Chu | album:              Mr. Chu | singer:                Apink | genre:                dance | highestRanking:         1
title:            Overwrite | album:                 LOOK | singer:                Apink | genre:                dance | highestRanking:         1
title:                Stuck | album:               Colors | singer:               miss A | genre:                  R&B | highestRanking:         0
title:                 Ugly | album:  2NE1 2nd Mini Album | singer:                 2NE1 | genre:                dance | highestRanking:         1
title:              WANNABE | album:               ITz ME | singer:                 ITZY | genre:                dance | highestRanking:         1
title:             WANT IT? | album:        ITz Different | singer:                 ITZY | genre:                dance | highestRanking:        52
```

(12) Should have at least 1 interface (menu and user input) and queries to select from database.

Select genre from song table

```java
	//print all the songs that is 00genre
	public static void searchGenre(Connection myConn, Statement myState, 
ResultSet myResSet) throws SQLException {
		//make an object to store all the input value
		Song song=new Song();
		String sql="";
		Scanner scanner=new Scanner(System.in);
		
		//get an input value from user about which genre's song does the 
user want to get
		System.out.print("Which genre do you want?(ballad/dance/R&B) ");
		song.genre=scanner.nextLine();
		//get the value of the song which has the genre of the input 
value
		sql="select singer, title from Song where genre=?";
		PreparedStatement ps=myConn.prepareStatement(sql);
		
		ps.setString(1, song.genre);
		myResSet=ps.executeQuery();
		
		System.out.println(String.format("Singer %21s | 
Title %21s","",""));
		System.out.println(String.format("%58s", "").replace(' ', '-'));
		
		//print all the value
		while(myResSet.next()) {
			song.singer=myResSet.getString("singer");
			song.title=myResSet.getString("title");
			System.out.println(String.format("singer: %20s | 
title: %20s", song.singer, song.title));
		}
		System.out.println();
	}
```



```
1.print information 2.insert a new singer 3.update information
4.delete a song 5.search 6.exit
>>>5
(1)genre (2)debut year (3)song title
>>>1
Which genre do you want?(ballad/dance/R&B) ballad
Singer                        | Title
----------------------------------------------------------
singer:                  ITZY | title:                24HRS
singer:                 Apink | title:               Always
singer:            OH MY GIRL | title:              Dolphin
```

(13) Should have at least 1 interface (menu and user input) and queries to select using nested queries and join.

Join song and album table.

And ested query is select singer from Singer where debut=?

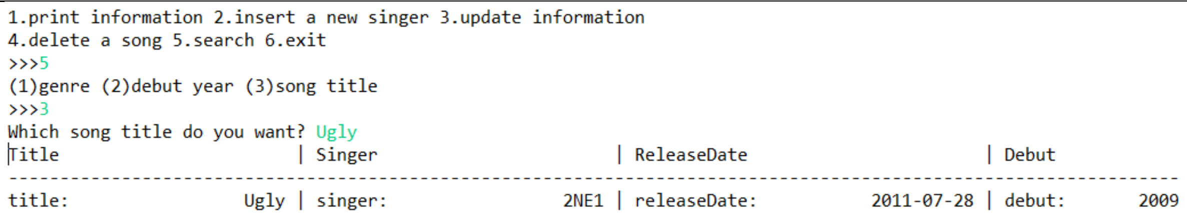
```
	//print all the album that the debut year of album's singer is equal to
the input year value
	public static void seachYear(Connection myConn, Statement myState,
ResultSet myResSet) throws SQLException {
		Scanner scanner=new Scanner(System.in);
		String sql="";
		//make a new song object and store input value here
		Song song=new Song();

		//get an input of debut year
		System.out.print("Which debut year do you want? ");
		int year=scanner.nextInt();
		//print the singer, album and release date of the album
		//if the debut year of song's singer is what the user want
		sql="select Song.singer, Song.album, Album.releaseDate " +
				"from Song, Album " +
				"where Song.album=Album.album " +
				"and Song.singer=Album.singer " +
				"and Song.singer in(select singer from Singer
where debut=?);";
		PreparedStatement ps=myConn.prepareStatement(sql);

		ps.setInt(1, year);
		myResSet=ps.executeQuery();

		System.out.println(String.format("Singer %21s | Album %21s |
ReleaseDate %21s","","",""));
		System.out.println(String.format("%93s", "").replace(' ', '-'));
		//print all the result value
		while(myResSet.next()) {
			song.singer=myResSet.getString("Song.singer");
			song.album=myResSet.getString("Song.album");
			String date=myResSet.getString("Album.releaseDate");
			System.out.println(String.format("singer: %20s |
album: %20s | releaseDate %20s", song.singer, song.album, date));
		}
		System.out.println();


	}
```

```
1.print information 2.insert a new singer 3.update information
4.delete a song 5.search 6.exit
>>>5
(1)genre (2)debut year (3)song title
>>>2
Which debut year do you want? 2019
Singer                        | Album                         | ReleaseDate
---------------------------------------------------------------------------------------------
singer:                  ITZY | album:         ITz Different | releaseDate          2019-02-12
singer:                  ITZY | album:                 ITz ME | releaseDate          2020-03-09
singer:                  ITZY | album:                 ITz ME | releaseDate          2020-03-09
```

(14) Should have at least 1 interface (menu and user input) and queries to select from view

Select from view named singer_song.

```
                                    else if (subop==3) {//get an input of song's title, then
print information about the song
                                            scanner=new Scanner(System.in);
                                            //get an input of the song title
                                            System.out.print("Which song title do you want? ");
                                            String title=scanner.nextLine();
                                            //show the singer of the song, singer's debut year,
release date of the album which contains the song
                                            sql="select title, singer, releaseDate, debut from
singer_song where title=?";
                                            PreparedStatement ps=myConn.prepareStatement(sql);

                                            ps.setString(1, title);
                                            myResSet=ps.executeQuery();

                                            System.out.println(String.format("Title %21s |
Singer %21s | ReleaseDate %21s | Debut %11s","","","",""));
                                            System.out.println(String.format("%114s",
"").replace(' ', '-'));

                                            while(myResSet.next()) {//show all values that we get
from sql
                                                   song.title=myResSet.getString("title");
                                                   song.singer=myResSet.getString("singer");
                                                   String date=myResSet.getString("ReleaseDate");
                                                   int debut=myResSet.getInt("Debut");
                                                   System.out.println(String.format("title: %20s
| singer: %20s | releaseDate: %20s | debut: %10d",song.title, song.singer, date, debut));
                                            }
                                            System.out.println();
```



```
1.print information 2.insert a new singer 3.update information
4.delete a song 5.search 6.exit
>>>5
(1)genre (2)debut year (3)song title
>>>3
Which song title do you want? Ugly
Title                    | Singer                    | ReleaseDate                    | Debut
---------------------------------------------------------------------------------------------------
title:               Ugly | singer:               2NE1 | releaseDate:         2011-07-28 | debut:       2009
```

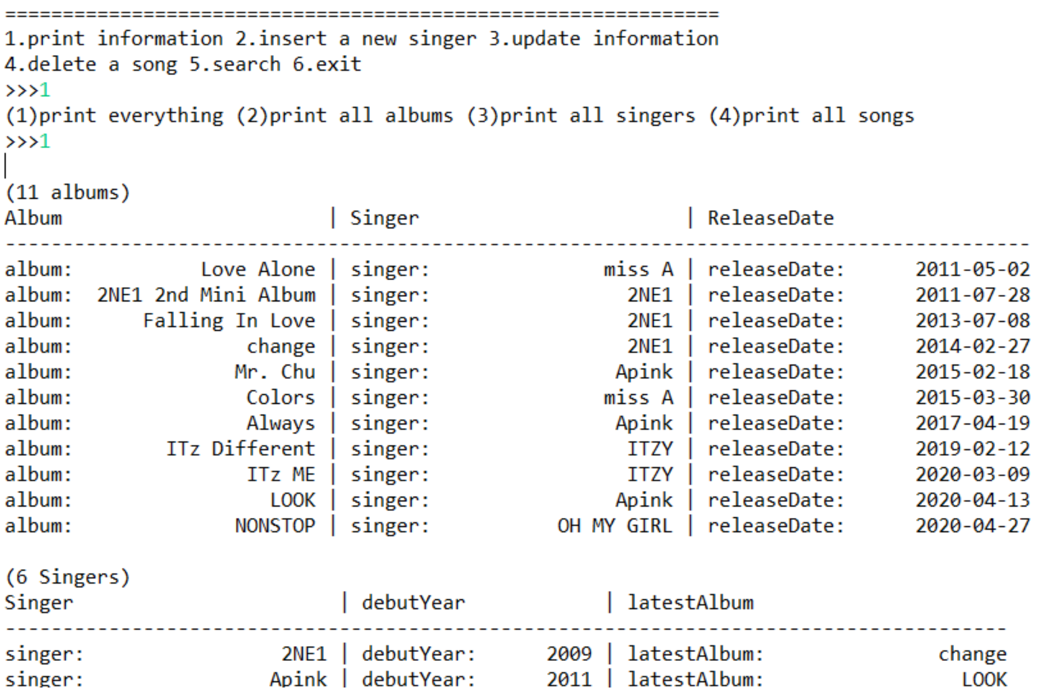
(15) Should have interface (menu) to print out contents of all tables

If a user gives an input of “1”, then program print out contents of all tables.

```
album.printAllAlbum(myConn, myState, myResSet); //static function that prints
every information about album
singer.printAllSinger(myConn, myState, myResSet); //static function that prints
information of singer
song.printAllSong(myConn, myState, myResSet); //static function that prints
information of song
```

```
	static void printAllAlbum(Connection myConn, Statement myState,
ResultSet myResSet) throws SQLException {
		String sql="";

		//get the number of album which is stored in the database
		sql="SELECT count(*) FROM album";
		myResSet=myState.executeQuery(sql);
		myResSet.next();
		int num=myResSet.getInt(1);
		//print out the number of album
		System.out.println("("+num+" albums)");
		//get all the information of the albums
		sql="SELECT * FROM album";
		myResSet=myState.executeQuery(sql);

		System.out.println(String.format("Album %21s | Singer %21s |
ReleaseDate %15s", "","",""));
		System.out.println(String.format("%89s", "").replace(' ', '-'));
		//print out the whole information about the album
		while(myResSet.next()) {
			album=myResSet.getString("album");
			singer=myResSet.getString("singer");
			releaseDate=myResSet.getString("releaseDate");

			System.out.println(String.format("album: %20s |
singer: %20s | releaseDate: %15s", album, singer, releaseDate));
		}
		System.out.println();
	}


	//static function that shows all the information of the singer
	public static void printAllSinger(Connection myConn, Statement myState,
ResultSet myResSet) throws SQLException {
		String sql="";
		//get the number of singer which is stored in the database
		sql="SELECT count(*) FROM Singer";
		myResSet=myState.executeQuery(sql);
		myResSet.next();
		int num=myResSet.getInt(1);
		//print out the number of singer
		System.out.println("("+num+" Singers)");
		//get all the value of singer
		sql="SELECT * FROM Singer";
		myResSet=myState.executeQuery(sql);

		System.out.println(String.format("Singer %21s | debutYear %10s |
latestAlbum %21s", "","",""));
		System.out.println(String.format("%87s", "").replace(' ', '-'));
		//print out the whole information of the singer table
		while(myResSet.next()) {
			singer=myResSet.getString("singer");
			debut=myResSet.getInt("debut");
			latestAlbum=myResSet.getString("latestAlbum");

			System.out.println(String.format("singer: %20s |
debutYear: %9s | latestAlbum: %20s", singer, debut, latestAlbum));
		}
```

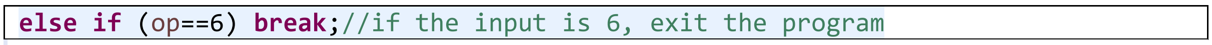
```
              System.out.println();
        }

        //static function that shows all the information of the song
        public static void printAllSong(Connection myConn, Statement myState,
ResultSet myResSet) throws SQLException {
              String sql="";

              //get the number of song which is stored in the song table
              sql="SELECT count(*) FROM Song";
              myResSet=myState.executeQuery(sql);
              myResSet.next();
              int num=myResSet.getInt(1);
              //print out number of song
              System.out.println("("+num+" Songs)");
              //get all the information from the song table
              sql="SELECT * FROM Song";
              myResSet=myState.executeQuery(sql);

              System.out.println(String.format("Title %21s | Album %21s |
Singer %21s | Genre %21s | HighestRanking %10s", "","","","",""));
              System.out.println(String.format("%146s", "").replace(' ', '-'));
              //print out the whole information of the song table
              while(myResSet.next()) {
                     title=myResSet.getString("title");
                     album=myResSet.getString("album");
                     singer=myResSet.getString("singer");
                     genre=myResSet.getString("genre");
                     highestRanking=myResSet.getInt("highestRanking");
                     System.out.println(String.format("title: %20s |
album: %20s | singer: %20s | genre: %20s | highestRanking: %9s", title, album,
singer, genre, highestRanking));
              }
              System.out.println();

        }
```



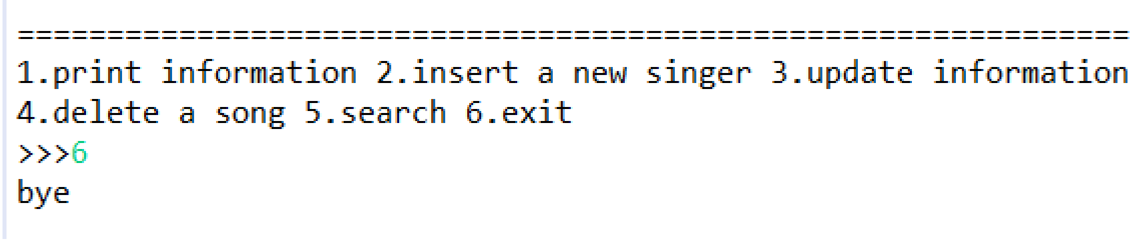
```
==============================================================
1.print information 2.insert a new singer 3.update information
4.delete a song 5.search 6.exit
>>>1
(1)print everything (2)print all albums (3)print all singers (4)print all songs
>>>1

(11 albums)
Album                       | Singer                      | ReleaseDate
------------------------------------------------------------------------------------------
album:          Love Alone | singer:              miss A | releaseDate:      2011-05-02
album:   2NE1 2nd Mini Album | singer:                2NE1 | releaseDate:      2011-07-28
album:      Falling In Love | singer:                2NE1 | releaseDate:      2013-07-08
album:               change | singer:                2NE1 | releaseDate:      2014-02-27
album:              Mr. Chu | singer:               Apink | releaseDate:      2015-02-18
album:               Colors | singer:              miss A | releaseDate:      2015-03-30
album:               Always | singer:               Apink | releaseDate:      2017-04-19
album:        ITz Different | singer:                ITZY | releaseDate:      2019-02-12
album:               ITz ME | singer:                ITZY | releaseDate:      2020-03-09
album:                 LOOK | singer:               Apink | releaseDate:      2020-04-13
album:              NONSTOP | singer:          OH MY GIRL | releaseDate:      2020-04-27

(6 Singers)
Singer                        | debutYear            | latestAlbum
----------------------------------------------------------------------------------------
singer:                  2NE1 | debutYear:      2009 | latestAlbum:                change
singer:                 Apink | debutYear:      2011 | latestAlbum:                  LOOK
```

(16) Should have interface (menu) to finish program gracefully. Otherwise menu should repeatedly appear automatically.

If a user gives an input of value "6", she can finish program gracefully.



```
else if (op==6) break;//if the input is 6, exit the program
```



```
==============================================================
1.print information 2.insert a new singer 3.update information
4.delete a song 5.search 6.exit
>>>6
bye
```

(6) SQL scripts (add into content of report) : this is for reviewing the codes in a nice format

createdb.sql

```
create table Singer( #information about singer
singer varchar(20) not null, #name of singer
debut int not null, #debut year of the singer
latestAlbum varchar(20), #the latest album of the singer
primary key (singer)); #pk is singer


create table Album( #information about the Album
album varchar(20) not null, #name of album
singer varchar(20) not null, #name of the singer
releaseDate date not null, #the release date of the album
primary key (album, singer), #pk is album and singer
#singer column in this table references singer in the Singer table
foreign key (singer) references Singer (singer));

create table Song( #information about the song
title varchar(20) not null, #title of the song
album varchar(20) not null, #the album that the song is included
singer varchar(20) not null, #the singer of the song
genre varchar(20) not null, #genre of the song
#highest ranking of the song
#null value when it was never in the top 10 or don't know the highest ranking
highestRanking int,
primary key (title), #pkis title
#if album value of the Album table changes, then the album value of Song table also changes
foreign key (album) references Album(album) on update cascade,
foreign key (singer) references Singer(singer));

#insert value in the Singer table
insert into Singer (singer, debut, latestAlbum) values
('2NE1', 2009, 'CRUSH'),
('miss A', 2010, 'Colors'),
('OH MY GIRL', 2015, 'NONSTOP'),
('Apink', 2011, 'LOOK'),
('ITZY', 2019, 'ITz ME');

#insert value in the Album table
insert into Album (album, singer, releaseDate) values
('CRUSH', '2NE1', '2014-02-27'),
('Falling In Love', '2NE1', '2013-07-08'),
('2NE1 2nd Mini Album', '2NE1', '2011-07-28'),
('Colors', 'miss A', '2015-03-30'),
('Love Alone', 'miss A', '2011-05-02'),
('LOOK', 'Apink', '2020-04-13'),
('Always', 'Apink', '2017-04-19'),
('Mr. Chu', 'Apink', '2015-02-18'),
('ITz ME', 'ITZY', '2020-03-09'),
('ITz Different', 'ITZY', '2019-02-12'),
('NONSTOP', 'OH MY GIRL', '2020-04-27');

#insert value in the Song table
insert into Song (title, album, singer, genre, highestRanking) values
('Come Back Home', 'CRUSH', '2NE1', 'dance', 2),
('Falling In Love', 'Falling In Love', '2NE1', 'dance', 1),
('hate you', '2NE1 2nd Mini Album', '2NE1', 'dance', 1),
('Ugly', '2NE1 2nd Mini Album', '2NE1', 'dance', 1),
('Love Song', 'Colors', 'miss A', 'dance', 66),
('I Caught Ya', 'Colors', 'miss A', 'R&B', 89),
('Stuck', 'Colors', 'miss A', 'R&B', null),
('Love Alone', 'Love Alone', 'miss A', 'dance', 70),
('Overwrite', 'LOOK', 'Apink', 'dance', 1),
('Be Myself', 'LOOK', 'Apink', 'dance', null),
('Always', 'Always', 'Apink', 'ballad', 30),
('Mr. Chu', 'Mr. Chu', 'Apink', 'dance', 1),
('Hush', 'Mr. Chu', 'Apink', 'dance', null),
('WANNABE', 'ITz ME', 'ITZY', 'dance', 1),
('24HRS', 'ITz ME', 'ITZY', 'ballad', 99),
('WANT IT?', 'ITz Different', 'ITZY', 'dance', 52),
('Dolphin', 'NONSTOP', 'OH MY GIRL', 'ballad', 76);

#make the index that point the album table's releaseDate
create index releaseDate_index on Album(releaseDate);

#make a view that select the title of the song, singer name of the song, release date of the album
and debut year of the singer
create view singer_song as
select Song.title, Song.singer, Album.releaseDate, Singer.debut
from Song, Album, Singer
where Song.album=Album.album and
Song.singer=Album.singer and
Album.singer=Singer.singer;
```

dropdb.sql

```
drop view singer_song; #view deletion
drop table Song; #song table deletion
drop table Album; #album table deletion
drop table Singer; #singer table deletion
```

(7) Java codes (add into content of report) : this is for reviewing the codes in a nice format

Main.java

```java
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Scanner;

public class Main {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Scanner scanner=new Scanner(System.in);
		//make the object of Album, Singer, Song class
		Album album=new Album();
		Singer singer=new Singer();
		Song song=new Song();

		//variables needed for connection
		String userID="dbuser";
		String userPW="dbpwd";
		String dbName="dbprj";
		String
url="jdbc:mysql://localhost:3306/"+dbName+"?&serverTimezone=UTC";

		Connection myConn=null;
		Statement myState=null;
		ResultSet myResSet=null;

		String sql="";


		try {
			//connect with jdbc
			myConn=DriverManager.getConnection(url,userID, userPW);
			myState=myConn.createStatement();

			//variables that gets an input from an user about the menu
			int op;
			while(true) {//do it repeatedly until the input value is 6
				scanner=new Scanner(System.in);
				//print the menu

	System.out.println("=====================================================
==========");
				System.out.println("1.print information 2.insert a
new singer 3.update information");
				System.out.println("4.delete a song 5.search
6.exit");
                                   System.out.print(">>>");
                                   op=scanner.nextInt();

                                   if (op==1) {//print the information
                                            //print every information, or only one of
album, singer, song tables
                                            System.out.println("(1)print everything
(2)print all albums (3)print all singers (4)print all songs");
                                            System.out.print(">>>");
                                            //get an input from user 1~4
                                            int subop=scanner.nextInt();
                                            System.out.println();
                                            if (subop==1) {//show all tables to user
                                                     album.printAllAlbum(myConn, myState,
myResSet); //static function that prints every information about album
                                                     singer.printAllSinger(myConn,
myState, myResSet); //static function that prints information of singer
                                                     song.printAllSong(myConn, myState,
myResSet); //static function that prints information of song
                                            }
                                            else if (subop==2) {//show only the
information of album

                                                     album.printAllAlbum(myConn, myState,
myResSet);

                                            }
                                            else if (subop==3) {//show only the
information of the singer

                                                     singer.printAllSinger(myConn,
myState, myResSet);

                                            }
                                            else if (subop==4) {//show only the
information of the song

                                                     song.printAllSong(myConn, myState,
myResSet);

                                            }else {//if the input is not one of 1,2,3,4
                                                     System.out.println("유효한 번호를
입력해주세요.");

                                            }

                                   }
                                   else if (op==2) {//insert new singer into singer
table
                                            singer.insertNewSinger(myConn, myState,
myResSet); //static function that insert singer name, debut year, latest
album(optionally)
                                   }
                                   else if (op==3) {//update the information
                                            //user can update the information about the
song's highest ranking or album's name
                                            System.out.println("(1)update highest
ranking of the song (2)update name of the album");
                                            System.out.print(">>>");
                                            //get an input about which information the
user want to update
                                            int subop=scanner.nextInt();
                                            if (subop==1) {//update song's highest
```

```
ranking
                                                      song.updateSongRanking(myConn, 
myState, myResSet); //get the information of song, then update the highest 
ranking
                                             }
                                             else if (subop==2){//update the name of 
album
                                                      scanner=new Scanner(System.in);
                                                      //get an input of album name that 
the user want to change
                                                      System.out.print("album: ");
                                                      album.album=scanner.nextLine();
                                                      //get an input of singer name that 
the user want to change
                                                      System.out.print("singer: ");
                                                      album.singer=scanner.nextLine();
                                                      //get an input of new name that the 
user want to change into
                                                      System.out.print("change album name 
into: ");
                                                      String 
newAlbumName=scanner.nextLine();
                                                      try {//use transaction to update two 
tables at one time
                                                               //turn off the auto commit
                                                               myConn.setAutoCommit(false);
                                                               //update the album name at 
the album table
                                                               sql="update album set album=? 
where album=? and singer=?";
                                                               PreparedStatement 
ps=myConn.prepareStatement(sql);
                                                               ps.setString(1, 
newAlbumName);
                                                               ps.setString(2, album.album);
                                                               ps.setString(3, 
album.singer);
                                                               ps.executeUpdate();
                                                               //we need to update all the 
column that contains album
                                                               //change the latest album 
name of the singer, if the value is updated
                                                               sql="update singer set 
latestAlbum=? where latestAlbum=? and singer=?";
       ps=myConn.prepareStatement(sql);
                                                               ps.setString(1, 
newAlbumName);
                                                               ps.setString(2, album.album);
                                                               ps.setString(3, 
album.singer);
                                                               ps.executeUpdate();
                                                               //commit
                                                               myConn.commit();
                                                      }
```

```
                                            catch(SQLException e) {
                                                    e.printStackTrace();
                                                    if (myConn!=null) {
                                                            try {//if commit
didn't work, then roll back

        System.out.println("Transaction is being rolled back");

        myConn.rollback();
                                                            }
                                                            catch(SQLException
e1) {

        e1.printStackTrace();
                                                            }
                                                    }
                                            }
                                            finally {
                                                    //change auto commit into
true

                                                    myConn.setAutoCommit(true);
                                            }
                                            System.out.println();
                                    }
                                    else {//if the value is not one of 1,2
                                            System.out.println("유효한 번호를
입력해주세요.");
                                    }
                            }
                            else if (op==4) {//song deletion
                                    song.deleteSong(myConn, myState,
myResSet);//static function that delete the song
                            }
                            else if (op==5) {//get an input from user, then
show the information it
                                    //genre, debut year, song title are the
three things that user can search
                                    System.out.println("(1)genre (2)debut year
(3)song title");
                                    System.out.print(">>>");
                                    //get an input about which information does
user want to get
                                    int subop=scanner.nextInt();
                                    if (subop==1) {//get an input about genre,
and show the song which is that genre
                                            song.searchGenre(myConn, myState,
myResSet); //static function that shows that genre
                                    }
                                    else if (subop==2) {//get an input about
singer's debut year, show every album of the singer who have that debut year
                                            song.seachYear(myConn, myState,
myResSet);//static function
                                    }
                                    else if (subop==3) {//get an input of
song's title, then print information about the song
                                            scanner=new Scanner(System.in);
                                            //get an input of the song title
```

```
                                                System.out.print("Which song title
do you want? ");
                                                String title=scanner.nextLine();
                                                //show the singer of the song,
singer's debut year, release date of the album which contains the song
                                                sql="select title, singer,
releaseDate, debut from singer_song where title=?";
                                                PreparedStatement
ps=myConn.prepareStatement(sql);

                                                ps.setString(1, title);
                                                myResSet=ps.executeQuery();


        System.out.println(String.format("Title %21s | Singer %21s |
ReleaseDate %21s | Debut %11s","","","",""));

        System.out.println(String.format("%114s", "").replace(' ', '-'));

                                                while(myResSet.next()) {//show all
values that we get from sql

        song.title=myResSet.getString("title");

        song.singer=myResSet.getString("singer");
                                                        String
date=myResSet.getString("ReleaseDate");
                                                        int
debut=myResSet.getInt("Debut");

        System.out.println(String.format("title: %20s | singer: %20s |
releaseDate: %20s | debut: %10d",song.title, song.singer, date, debut));
                                                }
                                                System.out.println();

                                        }
                                        else {//if input is not one of 1,2,3
                                                System.out.println("유효한 번호를
입력해주세요.");
                                        }
                                }
                                else if (op==6) break;//if the input is 6, exit
the program
                                else System.out.println("유효한 번호를
입력해주세요.");//if the input is not one of 1,2,3,4,5,6
                        }
                        System.out.println("bye");



                }catch(SQLException e) {
                        e.printStackTrace();
                }finally {//when the program end, disconnect all the connection
                        if(myResSet!=null) {
                                try {
                                        myResSet.close();
                                }catch(SQLException e) {
                                        e.printStackTrace();
                                }
                        }

                        if (myState!=null) {
                                try {
                                        myState.close();
                                }catch(SQLException e) {
                                        e.printStackTrace();
                                }
                        }

                        if (myConn!=null) {
                                try {
                                        myConn.close();
                                }catch(SQLException e) {
                                        e.printStackTrace();
                                }
                        }
                }
        }
}
```

Album.java

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Scanner;

public class Album {
        //column of Album table
        public static String album;
        public static String singer;
        public static String releaseDate;

        //static function that shows all the information of the album
        static void printAllAlbum(Connection myConn, Statement myState, ResultSet myResSet)
throws SQLException {
                String sql="";

                //get the number of album which is stored in the database
                sql="SELECT count(*) FROM album";
```

```
			myResSet=myState.executeQuery(sql);
			myResSet.next();
			int num=myResSet.getInt(1);
			//print out the number of album
			System.out.println("("+num+" albums)");
			//get all the information of the albums
			sql="SELECT * FROM album";
			myResSet=myState.executeQuery(sql);

			System.out.println(String.format("Album      %21s      |      Singer      %21s      |
ReleaseDate %15s", "","",""));
			System.out.println(String.format("%89s", "").replace(' ', '-'));
			//print out the whole information about the album
			while(myResSet.next()) {
				album=myResSet.getString("album");
				singer=myResSet.getString("singer");
				releaseDate=myResSet.getString("releaseDate");

				System.out.println(String.format("album:   %20s   |   singer:   %20s   |
releaseDate: %15s", album, singer, releaseDate));
			}
			System.out.println();
		}



}
```

Singer.java

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Scanner;

public class Singer {
	//column of singer table
	public static String singer;
	public static int debut;
	public static String latestAlbum;

	//static function that shows all the information of the singer
	public static void printAllSinger(Connection myConn, Statement myState, ResultSet
myResSet) throws SQLException {
		String sql="";
		//get the number of singer which is stored in the database
		sql="SELECT count(*) FROM Singer";
		myResSet=myState.executeQuery(sql);
		myResSet.next();
		int num=myResSet.getInt(1);
		//print out the number of singer
		System.out.println("("+num+" Singers)");
		//get all the value of singer
		sql="SELECT * FROM Singer";
		myResSet=myState.executeQuery(sql);

		System.out.println(String.format("Singer %21s | debutYear %10s |
latestAlbum %21s", "","",""));
		System.out.println(String.format("%87s", "").replace(' ', '-'));
		//print out the whole information of the singer table
		while(myResSet.next()) {
			singer=myResSet.getString("singer");
			debut=myResSet.getInt("debut");
			latestAlbum=myResSet.getString("latestAlbum");

			System.out.println(String.format("singer: %20s | debutYear: %9s |
latestAlbum: %20s", singer, debut, latestAlbum));
		}
		System.out.println();
	}
	//insert a new singer into singer table
	public static void insertNewSinger(Connection myConn, Statement myState, ResultSet
myResSet) throws SQLException {
		Scanner scanner=new Scanner(System.in);
		String sql="";
		//insert a new singer into the singer table
		//we are going to get the value from the user
		sql="insert into Singer(singer, debut, latestAlbum) values(?,?,?)";
		PreparedStatement ps=myConn.prepareStatement(sql);
		Singer singer=new Singer(); //make a new object to store the input value
		//get an input of the new singer name
		System.out.print("singer name: ");
		singer.singer=scanner.nextLine();
		//get an input of the singer's debut year
		System.out.print("debut year: ");
		singer.debut=scanner.nextInt();
		//get an input of the singer's latest album (but it's optional)
		scanner=new Scanner(System.in);
		System.out.print("Do you know their latest album?(y/n) ");
		String ans=scanner.nextLine();

		if (ans.equals("y")) {//if user wants to insert the value of the latest album
			System.out.print("lattest album: ");
			singer.latestAlbum=scanner.nextLine();
			ps.setString(1, Singer.singer);
			ps.setInt(2, singer.debut);
			ps.setString(3, singer.latestAlbum);
		}
		else {//if user don't want know the value of the latest album
			//fill this with null value
			ps.setString(1, Singer.singer);
			ps.setInt(2, singer.debut);
			ps.setString(3, null);
		}
		ps.executeUpdate();

		//if user wants to get the value of changed singer table
		//print all
		System.out.print("Do you want to see the singer list?(y/n) ");
		ans=scanner.nextLine();
		if (ans.equals("y")) singer.printAllSinger(myConn, myState, myResSet);
		System.out.println();

	}
```

```

}
```

Song.java

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Scanner;

public class Song {
	//column of song table
	public static String title;
	public static String album;
	public static String singer;
	public static String genre;
	public static int highestRanking;

	//static function that shows all the information of the song
	public static void printAllSong(Connection myConn, Statement myState, ResultSet 
myResSet) throws SQLException {
		String sql="";

		//get the number of song which is stored in the song table
		sql="SELECT count(*) FROM Song";
		myResSet=myState.executeQuery(sql);
		myResSet.next();
		int num=myResSet.getInt(1);
		//print out number of song
		System.out.println("("+num+" Songs)");
		//get all the information from the song table
		sql="SELECT * FROM Song";
		myResSet=myState.executeQuery(sql);

		System.out.println(String.format("Title %21s | Album %21s | Singer %21s | 
Genre %21s | HighestRanking %10s", "","","","",""));
```

```
        System.out.println(String.format("%146s", "").replace(' ', '-'));
        //print out the whole information of the song table
        while(myResSet.next()) {
            title=myResSet.getString("title");
            album=myResSet.getString("album");
            singer=myResSet.getString("singer");
            genre=myResSet.getString("genre");
            highestRanking=myResSet.getInt("highestRanking");
            System.out.println(String.format("title: %20s | album: %20s |
singer: %20s | genre: %20s | highestRanking: %9s", title, album, singer, genre, highestRanking));
        }
        System.out.println();

    }
    //staic function that updates song's highest ranking
    public static void updateSongRanking(Connection myConn, Statement myState,
ResultSet myResSet) throws SQLException {
        Song song=new Song();
        Scanner scanner=new Scanner(System.in);
        String sql="";
        //update the song's highest ranking with the input value
        sql="update Song set highestRanking=? where title=? and singer=?";
        PreparedStatement ps=myConn.prepareStatement(sql);
        //get the value of song title
        System.out.print("song title: ");
        song.title=scanner.nextLine();
        //get the value of singer
        System.out.print("singer: ");
        song.singer=scanner.nextLine();
        //get the value of highest ranking (change the table's highest ranking value into
this)
        System.out.print("highest ranking: ");
        song.highestRanking=scanner.nextInt();
        scanner=new Scanner(System.in);
        ps.setInt(1,song.highestRanking);
        ps.setString(2, song.title);
        ps.setString(3, song.singer);
        //execute update
        ps.executeUpdate();
```

```
                //if the user want to see the changed song list, print all
                System.out.print("Do you want to see the song list?(y/n) ");
                String ans=scanner.nextLine();
                if (ans.equals("y")) song.printAllSong(myConn, myState, myResSet);
                System.out.println();
        }
        //delete song
        public static void deleteSong(Connection myConn, Statement myState, ResultSet
myResSet) throws SQLException {
                //make a song object and store the input data at it
                Song song=new Song();
                Scanner scanner=new Scanner(System.in);
                String sql="";

                //title that tue user want to delete
                System.out.print("song title: ");
                song.title=scanner.nextLine();
                //get an input about the singer who sings that song
                System.out.print("singer: ");
                song.singer=scanner.nextLine();

                //delete the song which has same value as the value of input value
                sql="delete from Song where title=? and singer=?";
                PreparedStatement ps=myConn.prepareStatement(sql);

                ps.setString(1, song.title);
                ps.setString(2, song.singer);
                ps.executeUpdate();

                //if the user want to see the changed song list, print all
                System.out.print("Do you want to see the song list?(y/n) ");
                String ans=scanner.nextLine();
                if (ans.equals("y")) song.printAllSong(myConn, myState, myResSet);
                System.out.println();
        }
        //print all the songs that is 00genre
        public static void searchGenre(Connection myConn, Statement myState, ResultSet
myResSet) throws SQLException {
			//make an object to store all the input value
			Song song=new Song();
			String sql="";
			Scanner scanner=new Scanner(System.in);

			//get an input value from user about which genre's song does the user want to
get
			System.out.print("Which genre do you want?(ballad/dance/R&B) ");
			song.genre=scanner.nextLine();
			//get the value of the song which has the genre of the input value
			sql="select singer, title from Song where genre=?";
			PreparedStatement ps=myConn.prepareStatement(sql);

			ps.setString(1, song.genre);
			myResSet=ps.executeQuery();

			System.out.println(String.format("Singer %21s | Title %21s","",""));
			System.out.println(String.format("%58s", "").replace(' ', '-'));

			//print all the value
			while(myResSet.next()) {
				song.singer=myResSet.getString("singer");
				song.title=myResSet.getString("title");
				System.out.println(String.format("singer:    %20s    |    title:    %20s",
song.singer, song.title));
			}
			System.out.println();
		}
		//print all the album that the debut year of album's singer is equal to the input year
value
		public static void seachYear(Connection myConn, Statement myState, ResultSet
myResSet) throws SQLException {
			Scanner scanner=new Scanner(System.in);
			String sql="";
			//make a new song object and store input value here
			Song song=new Song();

			//get an input of debut year
			System.out.print("Which debut year do you want? ");
			int year=scanner.nextInt();
			//print the singer, album and release date of the album
			//if the debut year of song's singer is what the user want
			sql="select Song.singer, Song.album, Album.releaseDate " +
					"from Song, Album " +
					"where Song.album=Album.album " +
					"and Song.singer=Album.singer " +
					"and Song.singer in(select singer from Singer where 
debut=?);";
			PreparedStatement ps=myConn.prepareStatement(sql);

			ps.setInt(1, year);
			myResSet=ps.executeQuery();

			System.out.println(String.format("Singer %21s | Album %21s | 
ReleaseDate %21s","","",""));
			System.out.println(String.format("%93s", "").replace(' ', '-'));
			//print all the result value
			while(myResSet.next()) {
				song.singer=myResSet.getString("Song.singer");
				song.album=myResSet.getString("Song.album");
				String date=myResSet.getString("Album.releaseDate");
				System.out.println(String.format("singer: %20s | album: %20s | 
releaseDate %20s", song.singer, song.album, date));
			}
			System.out.println();



		}
}
```