2020-2 데이터베이스시스템 db프로젝트

프로젝트 4 : DBMS를 활용한 프로그램 개발

2019097347 이주은

-목차-

1. 수정 사항

- '구독자 관리하다' 테이블을 없애고, 어플의 구독자에 mgr_ssn을 foreign key로 추가하여 관리자의 pk인 ssn을 가리키도록 수정하였다. 구독자와 관리자 간의 fk관계가 형성되었다. Cardinality는 1:N이고, 한 관리자가 여러 구독자들을 관리할 수 있다.(정보 열람 가능)

- '음원 관리하다' 라는 테이블이 필요하지 않다고 판단되어 삭제하였다. 모든 관리자들은 모든 음원에 바로 접근하여 등록 및 삭제가 가능하다.

- 저번 프로젝트 그림에서 음원의 attribute인 '가수'가 빠져 있어서 추가했다.

- 음원의 attribute인 '순위'를 삭제했다. 음원이 재생된 횟수로 order하여 음원 차트를 출력할 수 있기 때문에 쓸모 없는 attribute라고 판단하였기 때문이다.

- 플레이리스트에는 구독자의 pk인 ID를 가리키는 '제작자id'가 라로 있기 때문에 '제작자' 즉 제작자의 이름을 attribute로 담고 있을 필요가 없다고 판단하여 삭제하였다.

- 플레이리스트의 '생성날짜' attribute를 삭제하였다.

- 플레이리스트의 이름은 중복될 수 없다고 했던 제약사항을 없앴다. 각각의 플레이리스트는 고유번호를 가지기 때문에 이름이 중복되어도 상관없다.

- 음원 차트는 순위 10위 까지 보여준다.

2. 어플리케이션 기능 설명

```
Who are you?
0.Exit
1.Manager
2.User
```

- 0을 입력하면 프로그램이 종료된다.
- 1은 일반 어플 관리자 , 2는 어플의 사용자(구독자), 3은 super manager

$ Manager

```
1
0.delete song
1.add song
2.view user's information
3.update user's information
4.birthday event
```

 # 0 : delete song

```
0
input the title of the song
Babo
====================
1.Babo-BigBang #snum: 123463
2.Babo-Nilo #snum: 123472
====================
input the ssn of the song that you want to delete
123463
#ssn : 123463 deleted...
--------------------
```

- 삭제하려는 음원의 제목을 입력하면, 이 제목을 가진 음원들의 리스트가 보여 진다.
- 삭제하려는 음원의 snum(고유번호)를 입력하면 음원이 삭제된다.

 # 1 : add song

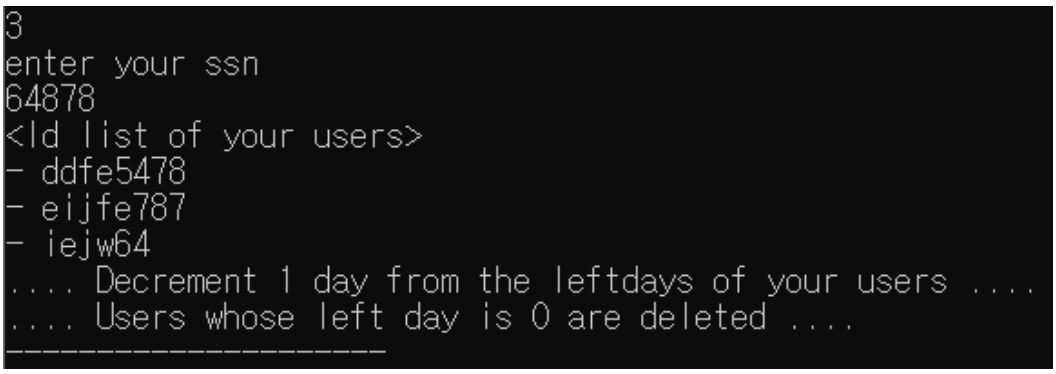
```
1
input today's date(ex>'2020-06-09')
2008-11-15
input the length(seconds) of the song(ex>180)
350
input the genre of song(ex> 'k-pop')
k-pop
input the title of song(ex> 'dynamite')
Babo
input the singer of song(ex> 'BTS')
Big-Bang
--------------------
```

- 음원을 추가하기 위해, 음원의 등록날짜와 음원의 길이, 장르, 제목, 가수를 입력한다.

#2 : view user's information

- 관리자는 유저의 아이디를 입력하면 그 유저의 이름, 성별, 그리고 어플 이용권 잔여일을 확인할 수 있다.

#3 : update user's information



- 3번을 통해 관리자는 자신이 담당하는 구독자들의 이용권 잔여일을 1일 차감한다.
- 먼저 자신이 담당하는 유저들의 목록이 나타나고, 이 유저들의 이용권 잔여일은 1일 줄어든다. 후에, 이용권 잔여일이 0인 유저들은 이용권 기한이 끝났기 때문에 삭제된다.

 # 4 : birthday event

- 이 어플은 생일에 이용자의 남은 기간을 +10일 해주는 이벤트를 진행한다.
- 오늘 생일인 유저의 목록을 보여준다. 잔여일+=10

$user

```
Who are you?
0.Exit
1.Manager
2.User
2
enter your id please(ex> 'nnfe87' )
ddfe5478
enter your password (ex> 54f8e78 )
jf54891
0.make a new playlist
1.view your list of playlists
2.view the songs of a playlist
3.delete the playlist
4.add or delete the song of the playlist
5.search the song
6.show the melon chart
```

- 사용자는 아이디와 비밀번호를 입력하여 접속한다.
- 등록되지 않은 아이디는 접속할 수 없으며, 등록된 아이디라도 비밀번호가 틀리면 접속할 수 없다.

```
enter your id please(ex> 'nnfe87' )
ddfe5478
enter your password (ex> 54f8e78 )
efewfewefwe
wrong password!
```

 # 0 : make a new playlist

```
0
Create the title of your playlist
database
New playlist is created. You can add songs at menu
--------------------
```

- 제목을 입력하면 해당 제목의 플레이리스트가 생성된다.

 # 1 : view the list of your playlists

```
1
<Your playlists>
- Driving
- HeHe
- database
--------------------
```

- 한 유저는 여러 플레이리스트를 만들 수 있다.
- 유저의 플레이리스트 목록을 보여준다.

 # 2 : view the songs of the playlist

```
2
<Your playlists>
Playlist: Driving , Pnum : 3
Playlist: HeHe , Pnum : 4
Playlist: database , Pnum : 6
enter a Pnum of the playlist that you want to view
4
#Title : dynamite #Singer : BTS
#Title : Fire #Singer : BTS
#Title : Kill this love #Singer : BlackPink
#Title : Yeosu Bambada #Singer : Busker Busker
#Title : DNA #Singer : BTS
----------------------
```

- 먼저, 유저의 플레이리스트 목록을 보여주고 보고싶은 플레이리스트의 번호를 입력 받는다.
- 번호를 입력하면 플레이리스트의 노래목록을 보여준다.

 # 3 : delete the playlist

```
3
<Your playlists>
Playlist: Driving , Pnum : 3
Playlist: HeHe , Pnum : 4
Playlist: database , Pnum : 6
enter a Pnum of the playlist that you want to delete
6
.... Deleting ....
----------------------
```

- 유저가 소유하고 있는 플레이리스의 제목과 number가 주어진다.
- 지우고자 하는 플레이리스트의 번호를 입력하면 지울 수 있다.

 # 4 : add or delete the song of the playlist

```
4
<Your playlists>
Playlist: Driving , Pnum : 3
Playlist: HeHe , Pnum : 4
enter the Pnum of the playlist that you want to edit
3
0.add song 1.delete song
```

- 유저의 플레이리스트 목록이 보여지고, 노래를 추가하거나 삭제하고 싶은 플레이리스트의 번호를 입력하면, 노래를 추가할 것인지 삭제할 것인지 입력한다.

- (노래 추가)

```
0
input the title of the song that you want to add
Babo
#Title: Babo #Singer : Nilo #Snum : 123472
#Title: Babo #Singer : Big-Bang #Snum : 123475
enter the number of the song that you want to add
123475
```

- 추가하고 싶은 노래의 제목을 입력하면 해당 제목을 가진 노래가 출력된다. 제목이 같은 노래가 있을 수도 있기 때문이다.

- 목록 중 추가하고자 하는 노래의 Snum을 입력하면 플레이리스트에 곡이 추가된다.

- (노래 삭제)

```
0.add song 1.delete song
1
#Title: Snowman #Singer : Jeongseunghwan #Snum : 123460
#Title: Aloha #Singer : Cool #Snum : 123462
#Title: Fire #Singer : BTS #Snum : 123467
#Title: Babo #Singer : Nilo #Snum : 123472
#Title: Babo #Singer : Big-Bang #Snum : 123475
enter the number of the song that you want to delete from the playlist
123462
```

- 삭제를 입력하면 플레이리스트에 있는 노래 목록이 출력된다.

- 목록 중 지우고자 하는 노래의 Snum을 입력하면 곡이 플레이리스트에서 삭제된다.

 # 5 : search the song

```
5
input the title of the song
VVV
no such song!
----------------------
```

```
5
input the title of the song
Babo
#Title: Babo #Singer : Nilo
#Title: Babo #Singer : Big-Bang
```

- 제목으로 노래를 검색할 수 있다.
- 없는 노래일 경우 없다고 출력, 있는 노래의 경우 해당 제목의 곡들을 출력한다.

 # 6 : show the melon chart

```
<melon top-10>
 1: Fire - BTS
 2: dynamite - BTS
 3: Yeosu Bambada - Busker Busker
 4: Kill this love - BlackPink
 5: Dance the night away - Twice
 6: Ballerino - Leessang
 7: Snowman - Jeongseunghwan
 8: HeartBreaker - G-dragon
 9: Aloha - Cool
10: How you like that - BlackPink
```

- 노래가 플레이 된 횟수로 음원 차트를 제공한다.
- 음원 차트는 10위까지 보여준다.

3. 코드 설명

```
public static void main(String[] args) {
    try {
        Class.forName("org.mariadb.jdbc.Driver");
        String url="jdbc:mariadb://localhost:3306/melon";
        String user = "root";
        String psw="thpine613";
        Connection con = DriverManager.getConnection(url, user, psw);
        Statement stmt = con.createStatement();
        ResultSet chk;
```

- Jdbc 드라이버와 자바를 connect한다. 이클립스의 경우 라이브러리에 jar 파일을 추가해주면 된다.

- User, password를 입력하면 mariaDB와 연결된다.

```
case 0: //delete song
    s.nextLine();//
    System.out.println("input the title of the song");
    String tit = s.nextLine();
    tit= "'"+tit+"'";
    chk=stmt.executeQuery("Select Song_num from song where Title=" +tit);
    if(!chk.next()) {
        System.out.println("no such song!");
        continue;
    }
    chk=stmt.executeQuery("Select Song_num,Title,Singer from "
            + "song where Title="+tit);
    int k=1;
    System.out.println("====================");
    while(chk.next()) {
        tit=chk.getString("Title");
        String singer = chk.getString("Singer");
        int snum=chk.getInt("Song_num");
        System.out.println(k+"."+tit+"-"+singer+" #snum: "+snum);
        k++;
    }
    System.out.println("====================");
    System.out.println("input the ssn of the song that you want to delete");
    int ssnum=s.nextInt();
    stmt.executeUpdate("Delete from include where Snum="+ssnum);
    stmt.executeUpdate("Delete from song where Song_num="+ssnum);
    System.out.println("#ssn : "+ssnum +" deleted...");
    break;
```

- 관리자 메뉴 중 0번(노래 삭제) 부분이다.

- 지우려는 노래의 제목을 입력 받는다. executeQuery로 query문을 작성한다. 이때, 큐리문에 String을 넣을 때 '~' 형태로 들어가야 함을 주의한다. tit = ""+tit+"";으로 처리했다.

- Song table에서 해당 제목을 가지는 노래의 song_num을 select한다.

- 지우려는 노래의 제목을 가진 노래들을 출력하고, 지우려는 노래의 번호를 입력받으면 executeUpdate를 통해 DB에서 delete 해준다.

- 노래를 지울 때, 이 노래가 어떤 플레이리스트에 속해 있다면, include table에서 FK가 이 노래의 song_num(pk)을 가리키고 있을 것이다. 그렇기 때문에 먼저 include 테이블에서 지우려는 노래가 있는 터플들을 먼저 삭제해주고, 후에 song table에서 노래를 삭제해야 한다.

```
case 1: //add song
    s.nextLine();//
    System.out.println("input today's date(ex>'2020-06-09')");
    String date = s.nextLine();
    date="'"+date+"'";
    System.out.println("input the length(seconds) of the song(ex>180)");
    int len = s.nextInt();
    s.nextLine();//
    System.out.println("input the genre of song(ex> 'k-pop')");
    String gen = s.nextLine();
    gen="'"+gen+"'";
    System.out.println("input the title of song(ex> 'dynamite')");
    String title = s.nextLine();
    title = "'"+title+"'";
    System.out.println("input the singer of song(ex> 'BTS')");
    String singer = s.nextLine();
    singer="'"+singer+"'";
    ResultSet rs =stmt.executeQuery("select Max(Song_num) from song");
    if(rs.next()) {
        int snum=rs.getInt(1)+1;
        stmt.executeUpdate("Insert into song(Enrolldate,Length,Genre,Title,Playnum,"
                + "Song_num,Singer) values("+
                date+","+len+","+gen+","+title+","+0+","+snum+","+singer+")");
    }
    break;
```

- 관리자 메뉴 중 1(노래 추가) 메뉴이다. 노래를 추가하기 위해 필요한 정보(등록 날짜, 노래의 길이, 장르, 제목, 가수)를 입력 받는다. Song table에서 song_num(노래의 고유 번호)는 primary key이다. 그래서 null일 수 없다. 새로운 노래의 고유 번호는 기존 노래 목록 중 가장 큰 고유번호+1 로 설정되어서 table에 추가되도록 하였다.

- executeUpdate로 Insert로 새로운 터플을 추가해준다.

```
case 2: // view user's information
    s.nextLine();
    System.out.println("input the user's id");
    String id = s.nextLine();
    id="'"+id+"'";
    rs= stmt.executeQuery("Select Name,Sex,Leftdays From user Where Id="+id);
    if(!rs.next()) {
        System.out.println("no such user!");
        System.out.println("input the user's id again");
        id=s.nextLine();
        id="'"+id+"'";
    }
    rs= stmt.executeQuery("Select Name,Sex,Leftdays From user Where Id=" + id);
    while(rs.next()) {
    String name = rs.getString("Name");
    String sex = rs.getString(2);
    int days = rs.getInt(3);
    System.out.println("#Name: "+name+" Sex: "+sex+" left days: "+days);}

    break;
```

- 관리자 메뉴 중 2(유저 신상 열람) 메뉴이다.
- 정보를 알고자 하는 유저의 아이디를 입력 받으면, 큐리를 던져서 사용자의 이름, 성별, 잔여일을 select 한다.
- ResultSet에서 getString을 통해 사용자의 정보를 출력한다.

```java
case 3://update users information
    s.nextLine();
    System.out.println("enter your ssn");
    String ssn = s. nextLine();
    ssn="'"+ssn+"'";
    ResultSet rs2=stmt.executeQuery("Select Name from manager where Ssn="+ssn);
    if(!rs2.next()) {
        System.out.println("no such manager!");
        continue;
    }
    rs = stmt.executeQuery("Select Id From user where Mgr_ssn="+ssn);
    System.out.println("<Id list of your users>");
    if(rs.next()) {
        String i = rs.getString("Id");
        System.out.println("- "+ i);
        while(rs.next()) {
            i=rs.getString("Id");
            System.out.println("- "+i);
        }
    }else {
        System.out.println("--EMPTY--");
        continue;
    }
    System.out.println(".... Decrement 1 day from the leftdays of your users ....");
    System.out.println(".... Users whose left day is 0 are deleted ....");
    stmt.executeUpdate("Update user set Leftdays=Leftdays-1 where Mgr_ssn="+ssn);
    stmt.executeUpdate("Delete from user where Leftdays<=0");
    break;
```

- 관리자 메뉴 중 3(유저의 정보 업데이트) 메뉴이다. 이 메뉴에서는 관리자가 담당하는 유저들의 어플 이용 잔여일에서 1일을 빼고, 만약 잔여일이 0이 되는 유저들은 삭제된다.

- 먼저 관리자의 사번을 입력 받으면, 이 관리자가 담당하는 유저들의 id를 출력한다.

- mgr_ssn이 지금 접속한 사원이 ssn과 동일한 유저들의 잔여일을 1일 빼주고, 남은 일수가 0이라면 유저 테이블에서 해당 터플들을 삭제한다.

```
case 4: //birthday event
    Calendar cal=Calendar.getInstance();
    int year = cal.get(cal.YEAR);
    int month = cal.get(cal.MONTH)+1;
    String m = ""+month;
    if(month<=9)
        m="0"+month;
    int dat=cal.get(cal.DATE);
    String d =""+dat;
    if(dat<=9)
        d="0"+dat;
    String today = m+"-"+d;
    //System.out.println(today);
    System.out.println("#EVENT : +10 days for users whose birthday is today!");
    ResultSet r = stmt.executeQuery("Select Bdate,Name From user");
    while(r.next()) {
        if(today.equals(r.getString(1).substring(5))) {
            System.out.println("<Users' name list>");
            stmt.executeUpdate("Update user set Leftdays=Leftdays+10");
            String n = r.getString("Name");
            System.out.println("- "+n);
        }
    }
    break;
```

- 관리자 메뉴 중 4(생일 이벤트) 메뉴이다.

- Calendar.getInstance( )을 통해 오늘 날짜를 구한다. 오늘 날짜를 String으로 바꾼다. 모든 유저들의 Bdate를 Select하여, equals함수를 통해 오늘 날짜와 생일이 동일한 유저들의 어플 사용 기한을 10일 늘려준다.

```
System.out.println("enter your id please(ex> 'nnfe87' )");
s.nextLine();
String uid = s.nextLine();
uid="'"+uid+"'";
chk = stmt.executeQuery("Select Id,Passwd from user where Id="+uid);
String rps;
if(!chk.next()) {
    System.out.println("! Not registered user !");
    continue;
}else {
    rps=chk.getString(2);
    //System.out.println(rps);
}
System.out.println("enter your password (ex> 54f8e78 )");
String ps = s.nextLine();

if(ps.equals(rps)==false) {
    System.out.println("wrong password!");
    continue;
}
```

- 유저는 ID와 password를 입력하여 접속한다. Id가 틀리거나, id는 맞지만 비밀번호가 틀리면 접속

할 수 없다.

```
case 0: // create a new playlist
    System.out.println("Create the title of your playlist");
    String pname = s.nextLine();
    pname="'"+pname+"'";
    ResultSet rs = stmt.executeQuery("Select Max(Pl_num) from playlist");
    int newplnum=0;
    if(rs.next()) {
        newplnum = rs.getInt(1)+1;
    }
    stmt.executeUpdate("Insert into playlist(Title,Pl_num,Creater_id) values("+pname
    newplnum+","+uid+")");
    System.out.println("New playlist is created. You can add songs at menu");
    break;
```

- 유저의 메뉴 중 0(새로운 플레이리스트 생성) 메뉴이다.
- 플레이리스트의 제목을 입력 받는다.
- 플레이리스트 테이블의 primary key 인 Pl_num을 지정해주기 위해서 테이블의 가장 큰 고유번호를 max(Pl_num)로 select한다. 이 번호+1을 고유번호로 지정하여 테이블에 새로운 플레이리스트를 추가해준다.

```
case 1://show the playlists of an user
    rs=stmt.executeQuery("Select Title from playlist where Creater_id="+uid);
    if(rs.next()) {
        System.out.println("<Your playlists>");
        String t=rs.getString(1);
        System.out.println("- "+t);
        while(rs.next()) {
            t = rs.getString(1);
            System.out.println("- "+t);
        }
    }else {
        System.out.println("--EMPTY--");
    }
    break;
```

- 유저의 메뉴 중 1(유저의 플레이리스트 목록을 보여준다)
- 플레이리스트 테이블에서 creater_id 가 접속한 유저의 id인 터플들을 출력한다.

```
case 2: //show the songs of the playlist
    rs=stmt.executeQuery("Select Title,Pl_num from playlist where Creater_id="+uid);
    if(rs.next()) {
        System.out.println("<Your playlists>");
        String rt=rs.getString(1);
        int en = rs.getInt(2);
        System.out.println("Playlist: "+rt+" , Pnum : "+en);
        while(rs.next()) {
            rt = rs.getString(1);
            en = rs.getInt(2);
            System.out.println("Playlist: "+rt+" , Pnum : "+en);
        }
    }else {
        System.out.println("<Your playlists>");
        System.out.println("--EMPTY--");
        continue;
    }
    System.out.println("enter a Pnum of the playlist that you want to view");
    int epn = s.nextInt();
    rs=stmt.executeQuery("Select Title,Singer from song where Song_num in ( select"
            + " Snum from include where Pnum="+epn+")");
    while(rs.next()) {
        String t = rs.getString(1);
        String sg = rs.getString(2);
        System.out.println("#Title : "+t+" #Singer : "+sg);
    }
    break;
```

- 유저의 메뉴 중 2(플레이리스트의 노래 목록을 출력한다.)

- 접속한 유저의 id가 creater_id인 플레이리스트들의 제목과 고유 번호를 목록으로 출력한다. 보고 싶은 플레이리스트의 고유 번호를 입력 받는다. include테이블에서 플레이리스트의 고유번호를 pnum으로 가지는 노래들의 제목과 가수를 select한다. Nested query를 써야 한다.

```
select Title,Singer
from Song
where Song_num in ( select Song_num
                    from include
                    where Pnum= epn);
```

```
case 3: //delete the playlist
    rs=stmt.executeQuery("Select Title,Pl_num from playlist where Creater_id="+uid);
    if(rs.next()) {
        System.out.println("<Your playlists>");
        String t=rs.getString(1);
        int pn = rs.getInt(2);
        System.out.println("Playlist: "+t+" , Pnum : "+pn);
        while(rs.next()) {
            t = rs.getString(1);
            pn = rs.getInt(2);
            System.out.println("Playlist: "+t+" , Pnum : "+pn);
        }
    }else {
        System.out.println("<Your playlists>");
        System.out.println("--EMPTY--");
        continue;
    }
    System.out.println("enter a Pnum of the playlist that you want to delete");
    int delnum = s.nextInt();
    stmt.executeUpdate("Delete from include where Pnum="+delnum);
    stmt.executeUpdate("Delete from playlist where Pl_num="+delnum);
    System.out.println(".... Deleting ....");
    break;
```

- 유저의 메뉴 중 3(플레이리스트를 삭제한다.)
- 접속한 유저의 id를 creater_id로 가지는 플레이리스트의 제목과 고유 번호를 목록으로 출력한다.
- 지우려는 플레이리스트의 고유번호를 입력 받는다. Include 테이블의 FK인 Pnum은 플레이리스트의 PK인 Pl_num을 가리키고 있기 때문에 플레이리스트를 삭제하기 위해서는 include테이블에서 먼저 지우려는 플레이리스트의 고유번호를 Pnum으로 가지는 터플들을 먼저 지워줘야 한다. 후에 플레이리스트에서 해당 플레이리스트를 삭제한다.

```
case 4: //add or delete song to playlist
    rs=stmt.executeQuery("Select Title,Pl_num from playlist where Creater_id="+uid);
    if(rs.next()) {
        System.out.println("<Your playlists>");
        String t=rs.getString(1);
        int pn = rs.getInt(2);
        System.out.println("Playlist: "+t+" , Pnum : "+pn);
        while(rs.next()) {
            t = rs.getString(1);
            pn = rs.getInt(2);
            System.out.println("Playlist: "+t+" , Pnum : "+pn);
        }
    }else {
        System.out.println("<Your playlists>");
        System.out.println("--EMPTY--");
        continue;
    }
    System.out.println("enter the Pnum of the playlist that you want to edit");
    int num = s.nextInt();
    System.out.println("0.add song 1.delete song");

    int cmdd = s.nextInt();
```

- 유저의 메뉴 중 4(플레이리스트에 노래 추가 혹은 삭제)

- 접속한 유저의 플레이리스트의 제목과 고유 번호를 먼저 출력한다.

- 편집하고 싶은 플레이리스트의 고유 번호를 입력 받고, 노래를 추가할 것인지 삭제 할 것인지 입력 받는다.

```
if(cmdd==0) { //add song
    ResultSet kk = stmt.executeQuery("select Pnum,count(*) from "
            + "include where Pnum="+num+" group by Pnum");
    if(kk.next()) {
    int pnn = kk.getInt(2);
    if(pnn >=100) {
        System.out.println("maximum number of songs is 100");
        continue;
        }
    }
    s.nextLine();
    System.out.println("input the title of the song that you want to add");
    String tit = s.nextLine();
    tit="'"+tit+"'";
    rs=stmt.executeQuery("Select Title,Singer,Song_num from song where Title="+tit);
    if(rs.next()) {
        String title = rs.getString(1);
        String singer= rs.getString(2);
        int snum= rs.getInt(3);
        System.out.println("#Title: "+title+" #Singer : "+singer +" #Snum : "+snum);
        while(rs.next()) {
            title = rs.getString(1);
            singer= rs.getString(2);
            snum=rs.getInt(3);
            System.out.println("#Title: "+title+" #Singer : "+singer+" #Snum : "+snum);
        }
    }else {
        System.out.println("no such song!");
        continue;
    }
    System.out.println("enter the number of the song that you want to add");
    int addsong = s.nextInt();
    stmt.executeUpdate("insert into include(Snum,Pnum) values("+addsong+","+num+")");
}
```

- 0 : 노래를 추가한다.

- 노래를 추가 하기 전, 먼저 count(*)를 통해 해당 플레이리스트의 곡 수를 select한다. 만약 100개 이상이라면 플레이리스트에 더 이상 노래를 추가 할 수 없다. 최대 곡 수는 100개이다.

- 100개 미만이라면 추가하려는 노래의 제목을 입력 받는다. 입력 받은 제목을 노래 제목으로 가지는 곡들의 목록을 출력한다. 추가하려는 노래의 고유번호를 목록에서 보고 입력하면 곡이 추가된다.

```
else if(cmdd==1) { //delete song
    rs=stmt.executeQuery("Select Title,Singer,Song_num from song "
            + "where Song_num In(select Snum "
            + "from include where Pnum="+num+")");
    if(rs.next()) {
        String st=rs.getString(1);
        String sg=rs.getString(2);
        int dn = rs.getInt(3);
        System.out.println("#Title: "+st+" #Singer : "+sg +" #Snum : "+dn);
        while(rs.next()) {
            st=rs.getString(1);
            sg=rs.getString(2);
            dn = rs.getInt(3);
            System.out.println("#Title: "+st+" #Singer : "+sg +" #Snum : "+dn);
        }
    }else {
        System.out.println("this playlist is empty");
        continue;
    }
    System.out.println("enter the number of the song that"
            + " you want to delete from the playlist");
    int ddn = s.nextInt();
    stmt.executeUpdate("delete from include where Pnum="+num+" AND Snum="+ddn);
}
break;
```

- 1 : 노래를 삭제한다.

- 플레이리스트의 노래들의 제목과 고유 번호를 목록으로 출력한다. 이 중 지우려고 하는 노래의 고유번호를 입력하면 include테이블에서 터플을 삭제한다. where문에서 Snum과 Pnum을 모두 명시해야 한다.

```
case 5: //search the song
    System.out.println("input the title of the song");
    String tit = s.nextLine();
    tit="'"+tit+"'";
    rs=stmt.executeQuery("Select Title,Singer from song where Title="+tit);
    if(rs.next()) {
        String title = rs.getString(1);
        String singer= rs.getString(2);
        System.out.println("#Title: "+title+" #Singer : "+singer);
        while(rs.next()) {
            title = rs.getString(1);
            singer= rs.getString(2);
            System.out.println("#Title: "+title+" #Singer : "+singer);
        }
    }else {
        System.out.println("no such song!");
    }
    stmt.executeUpdate("Update song set Playnum=Playnum+1 where Title="+tit);
    break;
```

- 유저의 메뉴 중 5 (노래를 검색한다.)

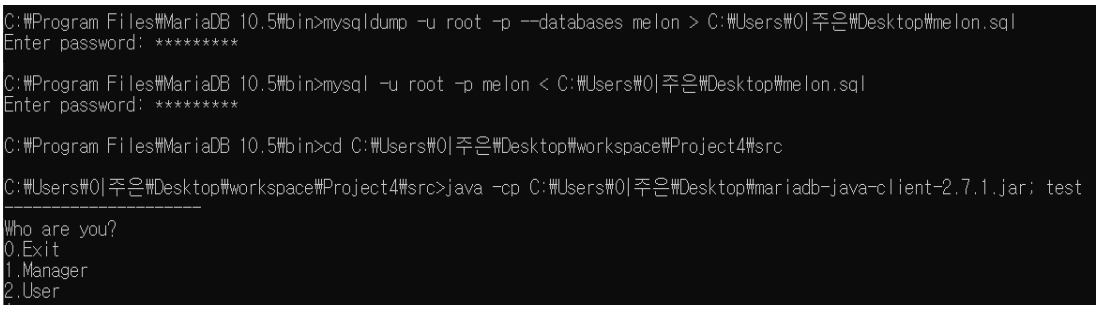
- 검색하고 싶은 노래의 제목을 입력 받는다. Song table에서 해당 노래 제목을 가진 곡들을 select하여 목록을 출력한다. 노래를 검색하면 그 노래를 듣는 것으로 간주하여 노래의 플레이 된 횟수에 1을 더해준다. Attribute의 value값을 변경하는 것은 Update~ set으로 구현한다. Playnum = Playnum+1 에서 왼쪽에는 바뀌는 값, 오른쪽은 바뀌기 전의 값이다.

```
case 6: // top-10 chart
    System.out.println("<melon top-10>");
    rs=stmt.executeQuery("Select Title,Singer from song order by Playnum desc");
    int rank=1;
    while(rs.next()) {
        String title = rs.getString(1);
        String singer =rs.getString(2);
        System.out.printf("%3d",rank);
        System.out.println(": "+title+" - "+singer);
        rank++;
        if(rank==11)
            break;
    }
    break;
```

- 유저의 메뉴 중 6 (top-10 음원 차트 출력)

- 각 노래들은 플레이 된 횟수를 저장한다. order by Planum desc을 통해, 내림 차순으로 플레이 된 횟수를 정렬한 ResultSet을 결과로 얻어서 노래 제목과 가수를 출력한다.

4. 실행 방법(Cmd)



- .sql 파일에 export를 해준다.

- dump파일을 import하여 database를 불러온다. 이때 내용은 없지만 비어 있고 같은 이름을 가진 database 틀을 만들어 놓아야 에러가 나지 않는다.

- 자바 어플리케이션 프로그램을 실행한다.