조민지 프로젝트: DBMS 프로그램 개발

MUSIC WORLD 만들기

◆ 프로젝트 1,2,3에서 설계한 DB를 기반으로 하는 어플리케이션 프로그램을 개발하였습니다.

◆ Mysql 을 이용해서 DB의 스키마를 작성했습니다. Intelli J 에서 Java 로 코딩 했습니다.

USER 테이블

NAME	SEX	SSN	ID	PSWD	UMANAGE
luffy	m	11111	luf	luf	112233
nami	w	33333	nam	nam	112233
zoro	m	44444	zor	zor	112233
usopp	m	55555	uso	uso	112233
chopper	w	66666	cho	cho	112233
NULL	NULL	NULL	NULL	NULL	NULL

PLAYLIST 테이블

PID	PNUM	POWNER
21	2	11111
23	3	33333
24	10	44444
25	0	55555
26	0	66666
NULL	NULL	NULL

MUSICCONTAIN 테이블

CPID	CMID
21	2
23	2
24	2
23	3
24	3
21	4

MUSIC 테이블

MID	MNAME	MSINGER	MLIKE	MMANAGE	MLYRIC
2	ice	icer	4	112233	iceiceice
3	iov	iover	4	112233	iovioviov
4	pho	phoer	3	112233	phophopho
5	a	aer	1	112233	aaaaaaaaaaaaa
6	b	ber	1	112233	bbbbbbbbbb
7	c	cer	1	112233	ccccccccccc
8	d	der	1	112233	ddddddddddd
9	e	eer	1	112233	eeeeeeeeee
10	f	fer	1	112233	fffffffffffff
11	g	ger	1	112233	ggggggggggg
12	h	her	1	112233	hhhhhhhhh
13	i	ier	0	112233	iiiiiii
NULL	NULL	NULL	NULL	NULL	NULL

MGENRE 테이블

MMID	MGENRE
2	h
2	k
3	k
4	k
5	k
6	i

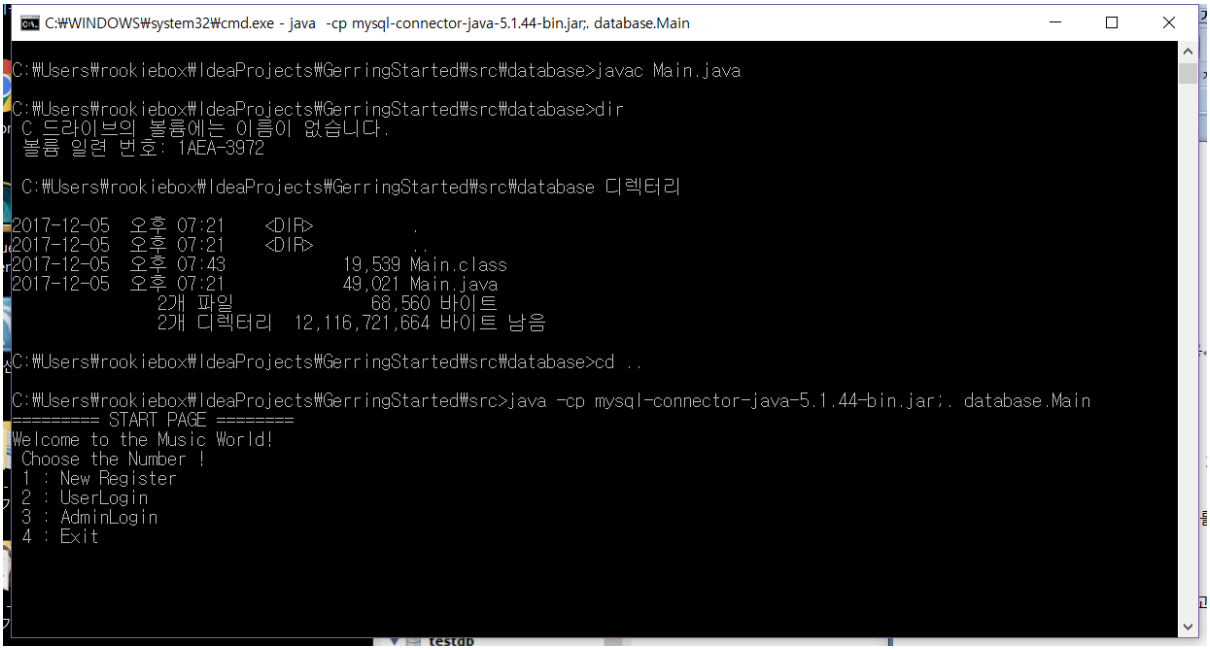
LEVEL 테이블

LNAME	LID	LMANAGE
P	11111	112233
S	33333	112233
B	44444	112233
P	55555	112233
B	66666	112233
NULL	NULL	NULL

ADMIN 테이블

ADMID	ADMPSWD	ADMSSN
admin	1234	112233
NULL	NULL	NULL

CLI 에서 Main.java 파일을 컴파일하고 실행했습니다.



C:₩Users₩rookiebox₩IdeaProjects₩GerringStarted₩src

java -cp mysql-connector-java-5.1.44-bin.jar;. database.Main

◆ 1 ===== START PAGE =====

가장 처음에 보여지는 화면입니다.

신규 회원 가입 / 사용자 로그인 / 관리자 로그인 / 프로그램을 [button]으로 선택할 수 있습니다. 올바른 값을 넣어야 합니다.

```
package database;

import ...


public class Main {

    public static void main(String[] args) throws Exception {
        boolean flag=true;
        Scanner scan = new Scanner(System.in);

        while(flag){

            int button;
            //start page select the [INT] number
            System.out.println("========= START PAGE ========");
            System.out.println("Welcome to the Music World!₩n Choose the Number ! ₩n 1 : New Register ₩n 2 : UserLogin ₩
            button = scan.nextInt(); //please int
            scan.nextLine();
```

✓ **1-1 button ==1 신규 회원가입**

사용자가 이름, 성별, 주민번호, ID, password, 설정하고 싶은 level을 입력합니다. 입력 받은 변수를 Registerinsert() 함수로 넘겨줍니다.

```java
switch(button){
    case 1:
        // Register page fill the right type value
        System.out.println("======== REGISTER PAGE ========");
        System.out.println("Please input your personal information");
        System.out.println("Enter your name : ");
        String name = scan.next();
        System.out.println("Enter your sex ( m / w ) : ");
        String sex = scan.next();
        System.out.println("Enter your ssn : ");
        int ssn = scan.nextInt();
        System.out.println("Enter your ID : ");
        String id = scan.next();
        System.out.println("Enter your password: ");
        String pass = scan.next();
        System.out.println("Which level do you want? ( Platinum: p , Gold: g , Sliver: s , Bronze: b) : ");
        String level = scan.next();
        //START PAGE : Register function
        Registerinsert(name , sex , ssn , id , pass , level);
        break;
```

Resigterinsert() 함수

```java
//Register insert this function inserts the USER not ADMIN / ADMIN is ONLY one so , I add the admin information into
public static void Registerinsert(String NAME, String SEX, int SSN, String ID, String PSWD, String LEVEL_p) {
    ResultSet rs = null;
    Connection conn = null;
    Statement stmt = null;
    int r;
    String levelname =null;
    levelname=leveler2(LEVEL_p);

    //First connection to my DB
    String driver = "com.mysql.jdbc.Driver";
    String url = "jdbc:mysql://localhost:3307/testdb?useSSL=false";
    String username = "root";
    String password = "4558";

    try {
        Class.forName(driver); //load the db
        conn = DriverManager.getConnection(url, username, password);
        int ad = 112233; // new USER is managed by ADMIN
        stmt = conn.createStatement();

        //check the ID whether duplicated
        rs = stmt.executeQuery( sql: "select * From testdb.user WHERE ID="+ID+";");
        while(rs.next()){
            System.out.println("WARNING : There are duplicate ID, please rewrite the information");
            return;
        }
        rs.close();

        //input the information in level , user , playlist DB
        stmt.executeUpdate( sql: "insert into testdb.level "+"(LNAME, LID, LMANAGE) VALUE ("+levelname+","+SSN+",
        r = stmt.executeUpdate( sql: "insert into testdb.user " + "(NAME, SEX, SSN, ID, PSWD, UMANAGE) value (" + 
        stmt.executeUpdate( sql: "insert into testdb.playlist "+"( PNUM, POWNER) VALUE ("+0+","+SSN+");");
        if (r == 1) {
            System.out.println("Register success! \n");
        } else {
            System.out.println("Register fail \n");
```

맨 처음 intellij와 mysql 을 연결합니다. String 변수에 미리 만든 스키마 DB의 주소와 로그 인을 위한 정보를 저장합니다.

Class.forName() 함수를 사용해서 지정한 DB driver 을 load 합니다.

Lever2() 함수를 이용해서 정형화한 값을 table에 삽입합니다.

```
public static String leveler2(String level){
    //System.out.println("I'm here \n"+level);
    if( level.equals("p") || level.equals("P") ) return "P";
    else if(  level.equals("g") || level.equals("G")) return "G";
    else if(  level.equals("s")|| level.equals("S")) return "S";
    else if(  level.equals("b") || level.equals("B")) return "B";
    else return null;
}
```

- ID 는 USER table 내에 유일한 값이어야 하므로 중복 체크를 해줍니다. Testdb.user 테이블에 있는 ID 칼럼과 사용자가 입력한 ID 값이 같은 게 있을 경우 SELECT 문을 사용해서 해당 행을 가져옵니다. WHERE 조건을 만족하는 행이 있을 경우 true로 판단하고 경고 창을 띄어줍니다.
- 사용자가 입력한 정보를 토대로 testdb.user, testdb.level, testdb.playlist table에 삽입합니다.

성공하면 Resiger success 를 출력하고 실패하면 Register fail 을 출력합니다.

```
======== REGISTER PAGE ========
Please input your personal information
Enter your name :
luffy
Enter your sex ( m / w ) :
m
Enter your ssn :
11111
Enter your ID :
luf
Enter your password:
luf
Which level do you want? ( Platinum: p , Gold: g , Sliver: s , Bronze: b) :
P
Register success!
```

USER table 결과값

	NAME	SEX	SSN	ID	PSWD	UMANAGE
	luffy	m	11111	luf	luf	112233
	NULL	NULL	NULL	NULL	NULL	NULL

LEVEL table 결과값

	LNAME	LID	LMANAGE
	P	11111	112233
	NULL	NULL	NULL

PLAYLIST table 결과값

	PID	PNUM	POWNER
	21	0	11111
	NULL	NULL	NULL

기능) ID 중복 체크

```
======== REGISTER PAGE ========
Please input your personal information
Enter your name :
luffy
Enter your sex ( m / w ) :
w
Enter your ssn :
22222
Enter your ID :
luf
Enter your password:
luf
Which level do you want? ( Platinum: p , Gold: g , Sliver: s , Bronze: b) :
s
WARNING : There are duplicate ID, please rewrite the information
```

✓ **1-2 button==2 사용자 로그인**

사용자의 ID 와 password 정보를 입력 받습니다.

```
case 2:
    //user login
    System.out.println("======== USER LOGIN ========");
    System.out.println("Input your ID : ");
    String loginid = scan.next();
    System.out.println("Input you password : ");
    String loginpswd = scan.next();

    //call the UserLogin function
    int check =UserLogin(loginid , loginpswd);
    // fail the login => in UserLogin function return 0;
    if(check==0)break;
    else{
        // success the login => in UserLogin function return 1; and the inter the USER PAGE
        System.out.println("login success ₩n ");
        boolean userflag= true;
```

UserLogin() 함수를 이용해서 check을 return 받습니다. Check 가 0 이면 로그 인을 실패한 것입니다. Check가 1이면 로그 인을 성공한 것입니다. 성공하면 USER PAGE로 이동합니다.

UserLogin() 함수입니다.

```java
//user login function
public static int UserLogin(String id , String pass ) throws Exception{
    Connection conn = null;
    Statement stmt = null;
    ResultSet rs = null;
    String dbpasswd = null;

    String driver = "com.mysql.jdbc.Driver";
    String url = "jdbc:mysql://localhost:3307/testdb?useSSL=false";
    String username = "root";
    String password = "4558";

    try {
        Class.forName(driver); //load the db
        conn = DriverManager.getConnection(url, username, password);
        stmt = conn.createStatement();
        //find password ,  ID == user's input id
        rs = stmt.executeQuery( sql: "select PSWD From testdb.user where ID='"+id+"';");

        if(rs.next()){ // if password exist load the PSWD in user table
            dbpasswd = rs.getString( columnLabel: "PSWD");
            // if lad PSWD is equals with user's input password return 1  / NOT equal return 0
            if (dbpasswd.equals(pass)){
                return 1;
            } else {
                System.out.println("wrong passwd please login again ₩n");
                return 0;
            }
        }else { // if user's input id is not exist return 0
            System.out.println("wrong id please login again ₩n");
            return 0;
        }
    }catch(Exception e) {
        e.printStackTrace();
    }
return 0;
```

db접근을 위한 connection과 load를 먼저 수행해줍니다.

- 우선 입력 받은 ID 정보와 testdb.user에 있는 ID칼럼이 일치하는 PSWD칼럼을 SELECT 이용해서 해당 칼럼의 값을 가져옵니다.

그 값을 [dbpasswd]변수에 저장합니다. WHERE 조건 문을 만족하는 PSWD 값이 없다면 ID 입력이 잘못됐다고 출력해줍니다. 만약 입력 받은 [pass] 정보와 [dbpasswd]가 일치하면 로그인 성공입니다. 일치하지 않으면 passwd 입력이 잘못됐다고 출력해줍니다.

Luffy 는 이미 회원가입을 했습니다. 로그 인을 성공해서 USER PAGE로 이동했습니다.

```
========= START PAGE ========
Welcome to the Music World!
 Choose the Number !
 1 : New Register
 2 : UserLogin
 3 : AdminLogin
 4 : Exit
2
======== USER LOGIN ========
Input your ID :
luf
Input you password :
luf
login success

======== USER PAGE ========
 1 : Go to the Play List
 2 : Music Store
 3 : Level Edit
 4 : Exit
```

Robin 은 회원가입을 하지 않았습니다. 로그 인을 실패하고 다시 START PAGE로 돌아왔습니다.

```
========= START PAGE ========
Welcome to the Music World!
 Choose the Number !
 1 : New Register
 2 : UserLogin
 3 : AdminLogin
 4 : Exit
2
======== USER LOGIN ========
Input your ID :
rob
Input you password :
rob
wrong id please login again

========= START PAGE ========
Welcome to the Music World!
 Choose the Number !
 1 : New Register
 2 : UserLogin
 3 : AdminLogin
 4 : Exit
```

- ✓ **1-3 button==3 관리자 로그인**

  관리자 회원가입을 따로 만들지 않았습니다.

  이유: 이 선택란이 일반 사용자에게 노출되면 안되고 이번 과제에 모델로 한 music 앱에도 이러한 서비스가 없었습니다. 제 개인적인 생각으론 관리자 정보같이 중요한 정보는 직접 DB 에 적는 게 더 적합할거 같아서 DB에 직접 넣었습니다.

```
public static void admininformation(){
    Connection conn = null;
    Statement stmt = null;
    ResultSet rs = null;
    String driver = "com.mysql.jdbc.Driver";
    String url = "jdbc:mysql://localhost:3307/testdb?useSSL=false";
    String username = "root";
    String password = "4558";
    boolean adminflag = false;

    try {
        Class.forName(driver); //load the db
        conn = DriverManager.getConnection(url, username, password);
        stmt = conn.createStatement();

        rs = stmt.executeQuery( sql: "select * From testdb.admin;");
        while(rs.next()){
            adminflag=true;
            return;
        }
        if(adminflag==false){
            stmt.executeUpdate( sql: "insert into testdb.admin (ADMID, ADMPSWD, ADMSSN) VALUE ('admin','1234',11223
        }

    }catch (ClassNotFoundException cnfe) {
        System.out.println("class not found");
    } catch (SQLException e) {
        System.out.println("sqlexception error₩n");
        e.printStackTrace();
    }
}
```

이 함수는 Main.class를 처음 실행할 때 수행되는 함수입니다.

- Testdb.admin테이블에 ADMIN 정보와 ADMPSWD정보와 ADMSSN정보를 직접 넣었습니다.

	ADMID	ADMPSWD	ADMSSN
	admin	1234	112233
	NULL	NULL	NULL

사용자 로그인과 동일하게 [adminid]와 [adminpswd]를 입력 받고 AdminLogin() 함수를 이용해서 올바른 정보인지 체크합니다. 로그 인에 성공하면 ADMIN PAGE로 이동하고 로그 인에 실패하면 START PAGE로 돌아옵니다.

```
case 3:
    //admin login
    System.out.println("======== ADMIN LOGIN ========");
    System.out.println("Input your ID : ");
    String adminid=scan.next();
    System.out.println("Input you password : ");
    String adminpswd = scan.next();
    //call the Admin login  this function is similar to UserLogin() but, they user different DB table USER , ADMIN
    int logincheck =AdminLogin(adminid , adminpswd);
    // fail the login => in AdminLogin function return 0;
    if(logincheck==0)break;
    else {
        // success the login => in AdminLogin function return 1; and the inter the ADMIN PAGE
        boolean adminflag= true;

        while (adminflag) {
            // ADMIN PAGE
            int adminbutton;
            System.out.println("======== ADMIN PAGE ========");
            System.out.println(" 1 : Music add \n 2 : Music delete \n 3 : Forced withdrawal \n 4 : Salary \n 5: Exit  ");
            adminbutton = scan.nextInt();
            scan.nextLine();
```

AdminLogin() 함수

```
//Admin login function
public static int AdminLogin(String id , String pass ) throws Exception{
    Connection conn = null;
    Statement stmt = null;
    ResultSet rs = null;
    String dbpasswd = null;

    String driver = "com.mysql.jdbc.Driver";
    String url = "jdbc:mysql://localhost:3307/testdb?useSSL=false";
    String username = "root";
    String password = "4558";

    try {
        Class.forName(driver); //load the db
        conn = DriverManager.getConnection(url, username, password);

        stmt = conn.createStatement();
        rs = stmt.executeQuery( sql: "select ADMPSWD From testdb.admin where ADMID='"+id+"';");

        if(rs.next()) {

            dbpasswd = rs.getString( columnLabel: "ADMPSWD");
            System.out.println("is it right?" + dbpasswd);

            if (dbpasswd.equals(pass)){
                System.out.println("login success \n ");
                return 1;
            }
            else {
                System.out.println("wrong passwd please login again \n");
                return 0;
            }
        }else {
            System.out.println("wrong id please login again \n");
            return 0;
        }
```

db접근을 위한 connection과 load를 먼저 수행해줍니다.

- 사용자가 입력한 id 값과 testdb.admin table에 있는 ID칼럼 값이 같으면 ADMPSWD 칼럼의 값을 가져옵니다.

그 값을 [dbpasswd]변수에 저장합니다. 만약 만족하는 값이 없으면 id를 잘못 입력한 것입니다. 사용자가 입력한 pass값과 [dbpasswd]값과 일치하는지 체크합니다. 일치하면 로그인 성공, 일치하지 않으면 잘못된 passwd 값을 입력한 것입니다.

관리자의 아이디는 admin 이고 비밀번호는 1234 입니다. 로그 인을 성공해서 ADMIN PAGE로 이동했습니다.

```
========= START PAGE ========
Welcome to the Music World!
 Choose the Number !
 1 : New Register
 2 : UserLogin
 3 : AdminLogin
 4 : Exit
3
======== ADMIN LOGIN ========
Input your ID :
admin
Input you password :
1234
login success

======== ADMIN PAGE ========
 1 : Music add
 2 : Music delete
 3 : Forced withdrawal
 4 : Salary
 5: Exit
```

로그 인을 실패했을 경우 START PAGE 로 이동

```
========= START PAGE ========
Welcome to the Music World!
 Choose the Number !
 1 : New Register
 2 : UserLogin
 3 : AdminLogin
 4 : Exit
3
======== ADMIN LOGIN ========
Input your ID :
wrong
Input you password :
wrongwrong
wrong id please login again

========= START PAGE ========
Welcome to the Music World!
 Choose the Number !
 1 : New Register
 2 : UserLogin
 3 : AdminLogin
 4 : Exit
```

✓ **1-4 button ==4 Music World 프로그램 종류**

프로그램을 종료합니다.

```
case 4:
    // Exit the music world
    flag=false;
    break;
```

◆ 2 =======ADMIN PAGE=======

관리자로 로그 인을 성공했을 때 보이는 PAGE입니다.

음악 추가 / 음악 삭제 / 사용자 강제 탈퇴 / 봉급 체크 / 관리자 페이지 나가기 기능이 있습니다. [adminbutton]으로 기능을 선택할 수 있습니다. 올바른 값을 입력해야 합니다.

```
while (adminflag) {
    // ADMIN PAGE
    int adminbutton;
    System.out.println("======== ADMIN PAGE ========");
    System.out.println(" 1 : Music add ₩n 2 : Music delete ₩n 3 : Forced withdrawal ₩n 4 : Salary ₩n 5: Exit  ");
    adminbutton = scan.nextInt();
    scan.nextLine();

    switch (adminbutton) {
        case 1:
            // Admin can add the new music
            MusicAdd();
            break;
        case 2:
            // Admin can delete the music
            MusicDelete();
            break;
        case 3:
            // Admin can delete the User withdrawal
            UserDelete();
            break;
        case 4:
            // Admin can count salary
            AdminSalary();
            break;
        case 5:
            // Exit the ADMIN PAGE
            adminflag=false;
            break;
    }
}
```

✓ **2-1 adminbutton == 1 관리자 음악 추가**

관리자가 MUSIC table에 음악을 추가할 수 있습니다. MusicAdd() 함수를 사용합니다.

```java
//admin can music add
public static void MusicAdd(){
    Scanner scan = new Scanner(System.in);
    Connection conn = null;
    Statement stmt = null;
    ResultSet rs = null;
    String driver = "com.mysql.jdbc.Driver";
    String url = "jdbc:mysql://localhost:3307/testdb?useSSL=false";
    String username = "root";
    String password = "4558";

    try {
        Class.forName(driver); //load the db
        conn = DriverManager.getConnection(url, username, password);
        stmt = conn.createStatement();
        int ad = 112233;
        String ans;
        //limit the music genre number 3
        String[] arr = new String[3];
        arr[0]=null;arr[1]=null;arr[2]=null;

        System.out.println("Do you want to check music list ? ( y / n )");
        ans = scan.next();
        //check the MUSIC LIST before the ADD
        if(ans.equals("y")){
            System.out.println("========  MUSIC LIST ========");
            System.out.println("ID  M-NAME   M-SINGER ");
            // execute the sql : to see the music information in music table
            rs = stmt.executeQuery( sql: "select * From testdb.music;");
            while(rs.next()){
                int mid = rs.getInt( columnIndex: 1);
                String mname = rs.getString( columnIndex: 2);
                String msinger = rs.getString( columnIndex: 3);
                System.out.println(mid+" ₩t"+mname+"₩t "+msinger);
            }
        }
```

db접근을 위한 connection과 load를 먼저 수행해줍니다

음악 한 개에 지정할 수 있는 장르를 3개로 한정했습니다.

먼저 현재 music store에 있는 노래 리스트를 보고 싶냐고 묻습니다. "y"라고 하면 MUSIC LIST를 보여주고 "n"라고 하면 생략합니다.

- Testdb.music table에 있는 모든 행의 columnindex가 1,2,3인 MID, MNAME, MSINGER 값들을 변수에 저장해 출력합니다.

```
    // MUSIC ADD PAGE
    System.out.println("======== MUSIC ADD ========");
    System.out.println("Input the music information");
    System.out.println("M ID : ");
    int mid = scan.nextInt();
    System.out.println(("M NAME : "));
    String mname = scan.next();
    System.out.println("M SINGER : ");
    String msinger = scan.next();
    System.out.println("M GENRE num (max genre num is 3): ");
    int genrenum = scan.nextInt();
    for(int i=0 ;i<genrenum;i++){
        arr[i]=scan.next();
    }
    System.out.println("M Lyric : ");
    String mlyric = scan.next();
    // execute the sql : to add the music in music table
    stmt.executeUpdate( sql: "insert into testdb.music "+"(MID, MNAME, MSINGER, MLIKE, MMANAGE,MLYRIC) VALUE ("
    System.out.println("Music add success!");
    // execute the sql : to add the music genre in mgenre table
    for(int i=0;i<genrenum;i++) {
        if(arr[i] != null) {
            stmt.executeUpdate( sql: "insert into testdb.mgenre " + "(MMID, MGENRE) VALUE (" + mid + ",'" + arr
        }
    }
}catch(ClassNotFoundException cnfe){
    System.out.println("class not found");
}catch(SQLException e) {
    System.out.println("sqlexception error₩n");
    e.printStackTrace();
}
```

새로 추가할 음악에 대한 정보를 입력 받습니다.

- 입력 받은 정보대로 MUSIC table에 넣습니다.
- 음악장르는 다중에트리뷰트입니다. Musiccontain이라는 table을 따로 만들어 각 노래의 장르를 저장했습니다.

```
======== ADMIN PAGE ========
 1 : Music add
 2 : Music delete
 3 : Forced withdrawal
 4 : Salary
 5: Exit
1
Do you want to check music list ? ( y / n )
y
========  MUSIC LIST ========
ID  M-NAME   M-SINGER
1       love     lover
2       ice      icer
3       joy      joyer
4       pho      phoer
5       a        aer
6       b        ber
7       c        cer
8       d        der
9       e        eer
10      f        fer
11      g        ger
12      h        her
======== MUSIC ADD ========
Input the music information
M ID :
13
M NAME :
j
M SINGER :
jer
M GENRE num (max genre num is 3):
1
k
M Lyric :
jjjjjjj
Music add success!
======== ADMIN PAGE ========
 1 : Music add
 2 : Music delete
 3 : Forced withdrawal
 4 : Salary
 5: Exit
```

MID	MNAME	MSINGER	MLIKE	MMANAGE	MLYRIC
1	love	lover	0	112233	lovelovelove~
2	ice	icer	0	112233	iceiceice
3	joy	joyer	0	112233	joyjoyjoy
4	pho	phoer	0	112233	phophopho
5	a	aer	0	112233	aaaaaaaaaaaaa
6	b	ber	0	112233	bbbbbbbbbb
7	c	cer	0	112233	ccccccccccc
8	d	der	0	112233	ddddddddddd
9	e	eer	0	112233	eeeeeeeeee
10	f	fer	0	112233	fffffffffffff
11	g	ger	0	112233	ggggggggggg
12	h	her	0	112233	hhhhhhhhh
13	j	jer	0	112233	jjjjjjj
NULL	NULL	NULL	NULL	NULL	NULL

✓ **2-2 adminbutton == 2 관리자 음악 삭제**

관리자가 MUSIC table에 있는 음악을 삭제할 수 있습니다. MusicDelete() 함수를 사용합니다.

MusicDelete ()

```
//admin can music delete
public static void MusicDelete() {
    Scanner scan = new Scanner(System.in);
    Connection conn = null;
    Statement stmt = null;
    ResultSet rs = null;
    int[] arrpid = new int[50]; // for save the delete music mid --> find user PID
    int[] arrpnum = new int[50]; // for save the delete music mid's [PNUM]
    for(int i=0;i<50;i++){
        arrpid[i]=0;
        arrpnum[i]=0;
    }
    String driver = "com.mysql.jdbc.Driver";
    String url = "jdbc:mysql://localhost:3307/testdb?useSSL=false";
    String username = "root";
    String password = "4558";

    try {
        Class.forName(driver); //load the db
        conn = DriverManager.getConnection(url, username, password);
        stmt = conn.createStatement();
        String ans;

        System.out.println("Do you want to check music list ? ( y / n )");
        ans = scan.next();
        if (ans.equals("y")) {
            System.out.println("======== MUSIC LIST ========");
            System.out.println("ID  M-NAME   M-SINGER ");
            // execute the sql : for check the music list
            rs = stmt.executeQuery( sql: "select * From testdb.music;");
            while (rs.next()) {
                int mid = rs.getInt( columnIndex: 1);
                String mname = rs.getString( columnIndex: 2);
                String msinger = rs.getString( columnIndex: 3);
                System.out.println(mid + "₩t" + mname + "₩t " + msinger);
            }
        }
```

db접근을 위한 connection과 load를 먼저 수행해줍니다

music delete는 처리해줘야 할게 많았습니다.

먼저 현재 music store에 있는 노래 리스트를 보고 싶냐고 묻습니다. "y"라고 하면 MUSIC LIST를 보여주고 "n"라고 하면 생략합니다.

```
    System.out.println("========= MUSIC DELETE ========");
    System.out.println("Input the music MID to delete ");
    int deletemid = scan.nextInt();
    int i=0;

    //find user[PID] who have deletemid [MID] and then subtract the user's [PNUM] -1
    rs = stmt.executeQuery( sql: "select * From testdb.musiccontain,testdb.playlist WHERE CMID="+deletemid+" A
    while (rs.next()) {
        int findpid = rs.getInt( columnLabel: "PID");
        int findpnum = rs.getInt( columnLabel: "PNUM");
        arrpid[i]=findpid;
        arrpnum[i]=findpnum-1;
        i=i+1;
    }
    // execute the sql : first : update the [PNUM]
    for(int j=0;j<i;j++){
        stmt.executeUpdate( sql: "UPDATE testdb.playlist " + "SET PNUM=" + arrpnum[j] + " WHERE PID=" + arrpid[
    }
    rs.close();

    // execute the sql : second : delete the music in music db
    stmt.executeUpdate( sql: "delete from testdb.music " + "where MID=" + deletemid + ";");
    System.out.println("Music delete success!");

} catch (ClassNotFoundException cnfe) {
    System.out.println("class not found");
} catch (SQLException e) {
    System.out.println("sqlexception error₩n");
    e.printStackTrace();
}
```

먼저 삭제하고 싶은 music 의 mid 번호를 입력 받습니다.

첫 번째 처리) 관리자는 음악을 music table에서 삭제하면 사용자들이 자신의 play list 에 담아뒀던 음악도 삭제됩니다. → 사용자들이 가지고 있는 음악의 개수가 줄어듭니다 → playlist table의 PNUM 칼럼을 수정해줘야 합니다.

- Testdb.musiccontain과 testdb.playlist을 join하고 SELECT문을 사용해서 CMID=deletemid && CPID = PID를 만족하는 모든 행을 찾습니다. 그 행들의 PID과 PNUM 정보를 각각 findpid[], findpnum[]배열에 저장합니다.
- findpnum[]에 저장한 사용자의 music num의 수에서 -1 한 값으로 Update 해줍니다.

두 번째 처리) 관리자는 음악을 삭제합니다.

- Testdb.music table에 행 중 입력 받은 [deletemid]변수와 일치하는 MID 을 찾아서 삭제해줍니다.

```
======== ADMIN PAGE ========
 1 : Music add
 2 : Music delete
 3 : Forced withdrawal
 4 : Salary
 5: Exit
2
Do you want to check music list ? ( y / n )
y
======== MUSIC LIST ========
ID  M-NAME   M-SINGER
1       love     lover
2       ice      icer
3       joy      joyer
4       pho      phoer
5       a        aer
6       b        ber
7       c        cer
8       d        der
9       e        eer
10      f        fer
11      g        ger
12      h        her
13      j        jer
========= MUSIC DELETE ========
Input the music MID to delete
1
Music delete success!
======== ADMIN PAGE ========
 1 : Music add
 2 : Music delete
 3 : Forced withdrawal
 4 : Salary
 5: Exit
```

BEFORE 상태

Musiccontain

CPID	CMID
21	1
22	1
23	1
21	2
22	2
23	2
21	3
22	3
21	4
NULL	NULL

music

MID	MNAME	MSINGER	MLIKE	MMANAGE	MLYRIC
1	love	lover	0	112233	lovelovelove~
2	ice	icer	0	112233	iceiceice
3	iov	iover	0	112233	iovioviov
4	pho	phoer	0	112233	phophopho
5	a	aer	0	112233	aaaaaaaaaaaaa
6	b	ber	0	112233	bbbbbbbbbb
7	c	cer	0	112233	ccccccccccc
8	d	der	0	112233	ddddddddddd
9	e	eer	0	112233	eeeeeeeeee
10	f	fer	0	112233	fffffffffff
11	g	ger	0	112233	ggggggggggg
12	h	her	0	112233	hhhhhhhhh
13	j	jer	0	112233	jjjjjjj
NULL	NULL	NULL	NULL	NULL	NULL

playlist

PID	PNUM	POWNER
21	4	11111
22	3	22222
23	2	33333
24	0	44444
25	0	55555
26	0	66666
NULL	NULL	NULL

AFTER 상태

CPID	CMID
21	2
22	2
23	2
21	3
22	3
21	4
NULL	NULL

MID	MNAME	MSINGER	MLIKE	MMANAGE	MLYRIC
2	ice	icer	3	112233	iceiceice
3	iov	iover	2	112233	iovioviov
4	pho	phoer	1	112233	phophopho
5	a	aer	0	112233	aaaaaaaaaaaaa
6	b	ber	0	112233	bbbbbbbbbb
7	c	cer	0	112233	ccccccccccc
8	d	der	0	112233	ddddddddddd
9	e	eer	0	112233	eeeeeeeeee
10	f	fer	0	112233	fffffffffff
11	g	ger	0	112233	ggggggggggg
12	h	her	0	112233	hhhhhhhhh
13	j	jer	0	112233	jjjjjjj
NULL	NULL	NULL	NULL	NULL	NULL

PID	PNUM	POWNER
21	3	11111
22	2	22222
23	1	33333
24	0	44444
25	0	55555
26	0	66666
NULL	NULL	NULL

✓ **2-3 adminbutton == 3 관리자가 사용자를 강제로 탈퇴**

관리자가 사용자를 강제로 탈퇴시킵니다. User,level,playlist,musiccontain table 에 있던 모든 사용자 정보가 지워집니다. UserDelete() 함수를 사용합니다.

UserDelete()

```
//admin cna User delete
public static void UserDelete(){
    Scanner scan = new Scanner(System.in);
    Connection conn = null;
    Statement stmt = null;
    ResultSet rs = null;
    String driver = "com.mysql.jdbc.Driver";
    String url = "jdbc:mysql://localhost:3307/testdb?useSSL=false";
    String username = "root";
    String password = "4558";

    try {
        Class.forName(driver); //load the db
        conn = DriverManager.getConnection(url, username, password);
        stmt = conn.createStatement();
        String ans;

        System.out.println("Do you want to check user list ? ( y / n )");
        ans = scan.next();
        if (ans.equals("y")) {
            System.out.println("======== USER LIST ========");
            System.out.println("NAME   SSN   ID   PSWD ");
            // execute the sql : check the user table
            rs = stmt.executeQuery( sql: "select * From testdb.user;");
            while (rs.next()) {
                String name = rs.getString( columnIndex: 1);
                int ssn = rs.getInt( columnIndex: 3);
                String id = rs.getString( columnIndex: 4);
                String pswd = rs.getString( columnIndex: 5);

                System.out.println(name + "₩t" + ssn + "₩t " + id + "₩t"+pswd);
            }
            rs.close();
        }
```

db접근을 위한 connection과 load를 먼저 수행해줍니다

먼저 현재 USER table에 있는 사용자 리스트를 보고 싶냐고 묻습니다. "y"라고 하면 USER LIST를 보여주고 "n"라고 하면 생략합니다

```
    }
    //USER DELETE : get the ssn input
    System.out.println("========= USER DELETE ========");
    System.out.println("Input the user SSN to delete ");
    int dssn = scan.nextInt();

    // execute the sql : check :  user' input ssn (dssn) exist in USER table ?
    boolean existflag = false;
    rs = stmt.executeQuery( sql: "select * From testdb.user WHERE SSN="+dssn+";");
    while (rs.next()) {
        existflag=true;
    }

    if(existflag==false){
        // Not exist
        System.out.println("WRONG : You input NOT EXIST [SSN] ");
    }else if(existflag==true) {
        // right input exist
        //execute the sql : delete the level , user table if user'input ssn is correct the LID , SSN
        stmt.executeUpdate( sql: "delete from testdb.level " + "where LID=" + dssn + ";");
        stmt.executeUpdate( sql: "delete from testdb.user " + "where SSN=" + dssn + ";");
        System.out.println("Delete Success");
    }
} catch (ClassNotFoundException cnfe) {
    System.out.println("class not found");
} catch (SQLException e) {
    System.out.println("sqlexception error\n");
    e.printStackTrace();
}
```

삭제할 사용자의 SSN을 입력 받습니다.

- 입력 받은 ssn과 testdb.user table의 SSN칼럼 중 일치하는 값을 찾습니다. rs.next() 의 값이 존재해서 값이 true가 된다면 existflag를 true로 바꿔줍니다.

if조건 문을 사용해서 관리자가 입력한 ssn정보를 가진 사용자가 존재하면 delete sql을 실행하고 존재하지 않으면 WRONG 경고문을 띄어줍니다.

- 입력 받은 ssn값과 일치하는 LID 를 찾아 testdb.level에 있는 행을 삭제하고, ssn값과 일치하는 SSN을 찾아 testdb.user에 있는 행을 삭제합니다.

삭제를 성공하면 Delete Success 문을 출력합니다.

```
======== ADMIN PAGE ========
 1 : Music add
 2 : Music delete
 3 : Forced withdrawal
 4 : Salary
 5: Exit
3
Do you want to check user list ? ( y / n )
y
======== USER LIST ========
NAME   SSN   ID   PSWD
luffy   11111    luf    luf
sanfi   22222    san    san
nami    33333    nam    nam
zoro    44444    zor    zor
usopp   55555    uso    uso
chopper 66666    cho    cho
========== USER DELETE ========
Input the user SSN to delete
123123
WRONG : You input NOT EXIST [SSN]
======== ADMIN PAGE ========
 1 : Music add
 2 : Music delete
 3 : Forced withdrawal
 4 : Salary
 5: Exit
3
Do you want to check user list ? ( y / n )
n
========== USER DELETE ========
Input the user SSN to delete
22222
Delete Success
======== ADMIN PAGE ========
 1 : Music add
 2 : Music delete
 3 : Forced withdrawal
 4 : Salary
 5: Exit
```

BEFORE 상태

User

NAME	SEX	SSN	ID	PSWD	UMANAGE
luffy	m	11111	luf	luf	112233
sanfi	m	22222	san	san	112233
nami	w	33333	nam	nam	112233
zoro	m	44444	zor	zor	112233
usopp	m	55555	uso	uso	112233
chopper	w	66666	cho	cho	112233
NULL	NULL	NULL	NULL	NULL	NULL

musiccontain

CPID	CMID
21	2
22	2
23	2
21	3
22	3
21	4
NULL	NULL

playlist

PID	PNUM	POWNER
21	3	11111
22	2	22222
23	1	33333
24	0	44444
25	0	55555
26	0	66666
NULL	NULL	NULL

level

LNAME	LID	LMANAGE
P	11111	112233
G	22222	112233
S	33333	112233
B	44444	112233
P	55555	112233
B	66666	112233
NULL	NULL	NULL

AFTER 상태

NAME	SEX	SSN	ID	PSWD	UMANAGE
luffy	m	11111	luf	luf	112233
nami	w	33333	nam	nam	112233
zoro	m	44444	zor	zor	112233
usopp	m	55555	uso	uso	112233
chopper	w	66666	cho	cho	112233
NULL	NULL	NULL	NULL	NULL	NULL

CPID	CMID
21	2
23	2
21	3
21	4
NULL	NULL

PID	PNUM	POWNER
21	3	11111
23	1	33333
24	0	44444
25	0	55555
26	0	66666
NULL	NULL	NULL

LNAME	LID	LMANAGE
P	11111	112233
S	33333	112233
B	44444	112233
P	55555	112233
B	66666	112233
NULL	NULL	NULL

✓ **2-4 adminbutton==4 관리자 월급 체크**

Mucis World 사용자들의 수와 레벨 별 사용자들의 수를 파악할 수 있고 이에 따른 관리자의 월급을 계산할 수 있습니다. AdminSalary() 함수를 사용합니다.

AdminSalary ()

```
//Admin Salary
public static void AdminSalary(){
    int sum=0;
    int usernum=0;
    int[] levelbox = new int[5];
    levelbox[4]=0;levelbox[1]=0;levelbox[2]=0;levelbox[3]=0;
    Scanner scan = new Scanner(System.in);
    Connection conn = null;
    Statement stmt = null;
    ResultSet rs = null;
    String driver = "com.mysql.jdbc.Driver";
    String url = "jdbc:mysql://localhost:3307/testdb?useSSL=false";
    String username = "root";
    String password = "4558";

    try {
        Class.forName(driver); //load the db
        conn = DriverManager.getConnection(url, username, password);
        stmt = conn.createStatement();
        System.out.println("======== ADMIN SALARY ========");
        //execute the sql : load the all user's level
        rs = stmt.executeQuery( sql: "select * From testdb.level;");
        while (rs.next()) {
            String levelname = rs.getString( columnIndex: 1);
            int pay = (6-leveler(levelname))*1000;
            int levelnum = leveler(levelname);
            levelbox[levelnum]=levelbox[levelnum]+1;
            sum = sum +pay;
            usernum= usernum+1;
        }
        System.out.println("Admin salary is "+sum);
        System.out.println("Music World User number is "+usernum);
        System.out.println("Detail information ₩n Platinum : "+levelbox[1]+"₩n Gold : "+levelbox[2]+"
    } catch (ClassNotFoundException cnfe) {
        System.out.println("class not found");
```

db접근을 위한 connection과 load를 먼저 수행해줍니다

레벨 별 사용자들의 수를 계산하기 위해서

- Testdb.level 테이블 전체를 select 합니다. 첫 번째 칼럼을 불러와서 levelname 변수에 저장하고 lever()함수를 이용해서 값을 리턴 받아 분류해 월급을 계산합니다.

Lever() 함수는 String 을 입력 받아 알맞은 정수 값으로 return 해주는 함수입니다.

```
public static int leveler(String level){
    //System.out.println("I'm here \n"+level);
    if( level.equals("p") || level.equals("P") ) return 1;
    else if(  level.equals("g") || level.equals("G")) return 2;
    else if(  level.equals("s")|| level.equals("S")) return 3;
    else if(  level.equals("b") || level.equals("B")) return 4;
    else return 0;
}
```

```
======== ADMIN PAGE ========
 1 : Music add
 2 : Music delete
 3 : Forced withdrawal
 4 : Salary
 5: Exit
4
======== ADMIN SALARY ========
Admin salary is 17000
Music World User number is 5
Detail information
 Platinum : 2
 Gold : 0
 Silver : 1
 Bronze : 2
======== ADMIN PAGE ========
 1 : Music add
 2 : Music delete
 3 : Forced withdrawal
 4 : Salary
 5: Exit
```

✓ 2-5 adminbutton==5 관리자 페이지 나가기

Adminflag의 값을 false로 변경해서 ADMIN PAGE의 while() 문을 끝내고 START PAGE 로 이동합니다.

```
case 5:
    // Exit the ADMIN PAGE
    adminflag=false;
    break;
```

```
======== ADMIN PAGE ========
 1 : Music add
 2 : Music delete
 3 : Forced withdrawal
 4 : Salary
 5: Exit
5
========= START PAGE ========
Welcome to the Music World!
 Choose the Number !
 1 : New Register
 2 : UserLogin
 3 : AdminLogin
 4 : Exit
```

◆ 3 ======= USER PAGE =======

사용자가 로그 인을 성공했으면 START PAGE 에서 USER PAGE로 넘어옵니다.

USER PAGE에서는 userbutton을 이용해 원하는 기능을 선택합니다.

플레이리스트로 이동하거나 음악 상점에 가거나 레벨을 변경할 수 있습니다.

```
while(userflag) {
    // USER PAGE
    int userbutton;
    System.out.println("======== USER PAGE ========");
    System.out.println(" 1 : Go to the Play List ₩n 2 : Music Store ₩n 3 : Level Edit ₩n 4 : Exit  ");
    userbutton = scan.nextInt();
    scan.nextLine();

    switch (userbutton) {
        case 1:
            //Go to the Play List : in here User can check their play list
            UserPlayList(loginid);
            break;
        case 2:
            // Music Store : in here User can buy the new music
            MusicStore(loginid);
            break;
        case 3:
            // Level Edit : in here User can change their level
            LevelRequest(loginid);
            break;
        case 4:
            // Exit : exit the USER PAGE return to the START PAGE
            userflag=false;
            break;
    }
}
```

✓ 3-1 userbutton == 1 사용자 플레이리스트 보기

사용자가 음악 상점에서 구입한 음악의 목록을 보여줍니다.

음악을 재생할 수 있습니다. / 음악을 삭제할 수 있습니다. / playlist 페이지를 나갑니다.

```
//User Play List
public static void UserPlayList(String loginid) {
    //print the User's music list
    System.out.println("======== MUSIC LIST ========");
    int[] arr = new int[50]; // for save the music mid in musiccontain db
    for(int j=0;j<50;j++){
        arr[j]=0;
    }// init the musiccontain save box
    int musicnum = 0; // for count the music num at list

    Scanner scan = new Scanner(System.in);
    Connection conn = null;
    Statement stmt = null;
    ResultSet rs = null;
    int updateplaylistnum=0;

    String driver = "com.mysql.jdbc.Driver";
    String url = "jdbc:mysql://localhost:3307/testdb?useSSL=false";
    String username = "root";
    String password = "4558";

    try {
        Class.forName(driver); //load the db
        conn = DriverManager.getConnection(url, username, password);
        stmt = conn.createStatement();

        int playlistid=0;
        String mlyric=null;
        // execute the sql : get the pid by use loginid
        rs = stmt.executeQuery( sql: "select PID From testdb.user,testdb.playlist WHERE ID='" + logini
        while(rs.next()) {
            playlistid = rs.getInt( columnLabel: "PID");
        }
        rs.close();
        // execute the sql : print the user's play list and save the information in arr
        rs = stmt.executeQuery( sql: "select MID,MNAME,MSINGER FROM testdb.musiccontain,testdb.music W
        while(rs.next()){
            int mid = rs.getInt( columnIndex: 1);
            arr[musicnum]=mid;
            String mname = rs.getString( columnIndex: 2);
            String msinger = rs.getString( columnIndex: 3);
            System.out.println(mid + "₩t" + mname + " ₩t" + msinger);
            musicnum=musicnum+1;
        }
        rs.close();
```

Access can be private more... (Ctrl+F1)

db접근을 위한 connection과 load를 먼저 수행해줍니다

사용자가 로그인할 때 사용한 loginid를 함수의 파라 미터로 받아옵니다.

우선 사용자의 플레이 리스트 안에 들어있는 음악들의 정보를 간단하게 보여줍니다.

- User 테이블과 playlist 테이블을 합친 후 SSN=POWNER 과 ID=loginid를 만족하는 행의 PID 칼럼을 가지고 옵니다. 그 값을 [playlistid]변수에 저장합니다.
- Music 테이블과 musiccontain 테이블을 합친 후 CMIN=MID 와 CPID=playlistid 조건을 만족하는 1,2,3칼럼을 미리 만들어놓은 arr[]에 1번columindex를 저장하고 mname에 2번, msinger에 3번을 저장해 음악정보를 출력합니다.

플레이 리스트에 있는 음악을 play 할 수 있고 음악을 삭제할 수 있고 나갈 수 있습니다.

[play button]을 이용합니다.

```java
//play the music / delete the music / exit the USER MUSIC LIST
boolean playflag = true;
int playbutton;

while (playflag) {
    System.out.println("=========================₩n 1 : Music Play ₩n 2 : Music Delete ₩n 3 : Exit");
    playbutton= scan.nextInt();
    scan.nextLine();

    switch(playbutton) {
        case 1:
```

✧ 3-1-1 playbutton == 1 사용자는 playlist에 있는 음악을 재생할 수 있습니다.
재생을 할 음악의 id를 입력 받아 [playmid]변수에 저장합니다.
추가 기능) 사용자가 입력한 id가 실제로 본인의 playlist에 있는 음악인지 체크합니다.
맨 처음 playlist를 출력했을 때 저장한 arr[]배열을 확인합니다.
배열 안에 있는 정보와 [playmid]정보가 일치하면 음악에 대한 상세한 정보를 출력해줍니다.

```java
switch(playbutton) {
    case 1:
        // music play --> show the music genre,lyric
        System.out.println("Choose the music id you want to play : ");
        int playmid = scan.nextInt();
        boolean rightinput = false;
        // check : really exist  input mid num in user play list ?
        for (int i = 0; i < musicnum; i++) {
            // already we save the music list at arr
            if (playmid == arr[i]) {
                // exist in user's list
                rightinput = true;
                // first print the music genre
                System.out.print("MUSIC GENRE : ");
                rs = stmt.executeQuery( sql: "select MGENRE From testdb.mgenre ₩HERE MMID=" + playm
                while(rs.next()) {
                    String mgenre = rs.getString( columnLabel: "MGENRE");
                    System.out.print(mgenre+" ");
                }
                rs.close();

                // second print the music lyrics
                rs = stmt.executeQuery( sql: "select MLYRIC FROM testdb.music ₩HERE MID=" + playmid
                while (rs.next()) {
                    String lyric = rs.getString( columnLabel: "MLYRIC");
                    System.out.println("₩n"+lyric);
                }
                rs.close();
            }
        }//for end
        if (rightinput == false) {
            // user input wrong music num
            System.out.println("WRONG : You input the wrong music id, please input again ");
        }
        break;
```

음악에 대한 상세한 정보인 장르와 가사를 출력해줍니다.

- Testdb.mgenre 테이블에서 MMID칼럼=playmid가 같은 행의 MGENRE를 모두 출력합니다. 장르가 한 개 이면 while문을 1번 돌고 장르가 2개이면 while문을 2번 돌기 때문에 정상적인 출력이 가능합니다.
- Testdb.music 테이블에서 MID칼럼=playmid가 같은 행의 MLYRIC칼럼을 출력합니다.

Luffy의 playlist에는 2,3,4번 노래가 있습니다. Luffy의 CPID는 21입니다.

CPID	CMID
21	2
23	2
21	3
21	4
NULL	NULL

```
======== USER PAGE ========
 1 : Go to the Play List
 2 : Music Store
 3 : Level Edit
 4 : Exit
1
======== MUSIC LIST ========
2        ice      icer
3        joy      joyer
4        pho      phoer
============================
 1 : Music Play
 2 : Music Delete
 3 : Exit
1
Choose the music id you want to play :
2
MUSIC GENRE : h k
iceiceice
============================
 1 : Music Play
 2 : Music Delete
 3 : Exit
```

Luffy 의 playlist 에는 5번 뮤직이 없습니다.

```
============================
 1 : Music Play
 2 : Music Delete
 3 : Exit
1
Choose the music id you want to play :
5
WRONG : You input the wrong music id, please input again
============================
 1 : Music Play
 2 : Music Delete
 3 : Exit
```

✧ 3-1-2 playbutton == 2 사용자는 playlist에 있는 음악을 삭제할 수 있습니다.
삭제할 음악의 id를 입력 받아 [deletemid]변수에 저장합니다.
추가 기능) 사용자가 입력한 deletmid가 playlist에 있는 음악인지 체크합니다.
Rightinput2 플래그를 사용해 if조건 문을 걸어줍니다.
맨처음 playlist를 출력했을 때 저장한 arr[]배열을 확인합니다.

```
case 2:
    //User can music delete in their play list $change the PLAYLIST 's COLUMN [PNUM]
    System.out.println("Choose the music id you want to delete : ");
    int deletemid = scan.nextInt();
    boolean rightinput2 = false;
    // check : really exist  delete mid num in user play list ?
    for (int i = 0; i < musicnum; i++) {
        // already we save the user's play list music mid
        if (deletemid == arr[i]) {
            // exist
            rightinput2 = true;
            //subtract -1 [PNUM] and change null the arr[] value to null
            musicnum=musicnum-1;
            arr[i]=0;
            // execute the sql : delete the music in user's musiccontain db
            stmt.executeUpdate( sql: "delete from testdb.musiccontain " + "where CMID=" + deletemid
            // execute the sql : get the play list [PNUM] num in level table
            rs = stmt.executeQuery( sql: "select PNUM From testdb.playlist WHERE PID=" + playlistid
            while (rs.next()) {
                updateplaylistnum = rs.getInt( columnLabel: "PNUM");
                updateplaylistnum = updateplaylistnum - 1;
            }
            // execute the sql : update the music list number
            stmt.executeUpdate( sql: "UPDATE testdb.playlist " + "SET PNUM=" + updateplaylistnum + "
            rs.close();
        }
    }
    if (rightinput2 == false) {
        // user input the wrong mid
        System.out.println("WRONG : You input the wrong music id, please input again ");
    }
    break;
```

사용자 playlist에서 음악이 삭제되면 playlist 테이블에 있는 [PNUM]의 수도 변경해 줘야 합니다. [Updateplaynum]변수에 기존에 있던 숫자를 받아 -1연산을 해서 저장해줍니다.

- Testdb.musiccontain 테이블에 있던 CMID=deletemid 조건과 CPID=playlistid 조건을 둘 다 만족하는 행을 삭제합니다.
- 기존에 있던 [PNUM]의 정보를 가져오기 위해 select문을 사용합니다. PID=playlistid 조건을 만족하는 PNUM칼럼을 testdb.playlist 테이블에서 가져옵니다.
- PID=playlistid 조건을 만족하면 -1연산을 한 [updateplaylistnum]변수를 testdb.playlist 테이블에 있는 PNUM칼럼에 UPDATE를 해줍니다.

Luffy의 play list에는 1,2,3음악이 있습니다.
Luffy의 CPID는 21번입니다. PNUM이3입니다.

CPID	CMID
21	2
23	2
21	3
21	4
NULL	NULL

PID	PNUM	POWNER
21	3	11111
23	1	33333
24	0	44444
25	0	55555
26	0	66666
NULL	NULL	NULL

```
============================
 1 : Music Play
 2 : Music Delete
 3 : Exit
2
Choose the music id you want to delete :
3
============================
 1 : Music Play
 2 : Music Delete
 3 : Exit
```

삭제 후 musiccontain 테이블이 업데이트 되었습니다.

CPID	CMID
21	2
23	2
21	4
NULL	NULL

PID	PNUM	POWNER
21	2	11111
23	1	33333
24	0	44444
25	0	55555
26	0	66666
NULL	NULL	NULL

Luffy의 playlist 에는 5번 음악이 없습니다.

```
============================
 1 : Music Play
 2 : Music Delete
 3 : Exit
2
Choose the music id you want to delete :
5
WRONG : You input the wrong music id, please input again
============================
 1 : Music Play
 2 : Music Delete
 3 : Exit
```

✧ 3-1-3 playbutton == 3 사용자는 playlist페이지에서 나갈 수 있습니다.
USER PAGE로 돌아갑니다.

```
case 3:
    //user music list exit --> go back USER PAGE
    playflag=false;
    break;
```

✓ **3-2 userbutton == 2 음악 상점 가기**

사용자는 음악상점에 가서 음악리스트를 확인하고 음악을 구입할 수 있습니다.

조건) 사용자의 레벨에 따라 playlist의 크기가 다릅니다.

P플레: 40개, G골드: 30개, S실버: 20개, B브론즈: 10개

만약 luffy의 레벨이 B이고 playlist에 음악이 10개가 있다면 꽉 찼다고 경고 창을 띄어줍니다.

```
//Music Store
public static void MusicStore(String loginid) {
    System.out.println("========= MUSIC STORE ========");
    Scanner scan = new Scanner(System.in);
    Connection conn = null;
    Statement stmt = null;
    ResultSet rs = null;
    String driver = "com.mysql.jdbc.Driver";
    String url = "jdbc:mysql://localhost:3307/testdb?useSSL=false";
    String username = "root";
    String password = "4558";

    try {
        Class.forName(driver); //load the db
        conn = DriverManager.getConnection(url, username, password);
        stmt = conn.createStatement();
        //print the music list
        System.out.println("ID  M-NAME   M-SINGER ");
        rs = stmt.executeQuery( sql: "select * From testdb.music;");
        while (rs.next()) {
            int mid = rs.getInt( columnIndex: 1);
            String mname = rs.getString( columnIndex: 2);
            String msinger = rs.getString( columnIndex: 3);
            System.out.println(mid + "₩t" + mname + "₩t" + msinger);
        }
        rs.close();

        boolean storeflag = true;
        System.out.println("Do you want to add the music? (y / n) ");
        String ans = scan.next();

        if (ans.equals("n")) {
            //storeflag = false;
            return;
```

db접근을 위한 connection과 load를 먼저 수행해줍니다

우선 상점에 있는 모든 음악들을 출력해줍니다.

- Testdb.music 테이블에 접근해서 모든 칼럼을 가지고 옵니다. 각 행의 1,2,3 칼럼인 MID, MNAME, MSINGER정보를 출력해줍니다.

사용자는 음악리스트를 체크하고 음악을 추가할건지에 대한 대답을 합니다. "y"라고 하면 음악을 추가하기 위한 정보를 입력 받고 "n"라고 하면 USER PAGE로 이동합니다.

```
3
======== USER PAGE ========
 1 : Go to the Play List
 2 : Music Store
 3 : Level Edit
 4 : Exit
2
========= MUSIC STORE ========
ID  M-NAME   M-SINGER
2       ice     icer
3       joy     joyer
4       pho     phoer
5       a       aer
6       b       ber
7       c       cer
8       d       der
9       e       eer
10      f       fer
11      g       ger
12      h       her
13      j       jer
Do you want to add the music? (y / n)
n
======== USER PAGE ========
 1 : Go to the Play List
 2 : Music Store
 3 : Level Edit
 4 : Exit
```

```
} else {
    // add the music in user's play list
    int playlistid=0;
    int playlistnum=0;
    int musiclikenum=0;
    int updateplaylistnum=0;
    int ssn=0;
    String levelname=null;
    int checkmid=0;

    //get the [PID] , [SSN] by use loginid
    rs = stmt.executeQuery( sql: "select PID,SSN From testdb.user,testdb.playlist WHERE ID='" + log
    while(rs.next()) {
        playlistid = rs.getInt( columnLabel: "PID");
        ssn=rs.getInt( columnLabel: "SSN");
    }
    rs.close();

    while (storeflag) {
        //check [PNUM] whether user can add the music compare with LEVEL table [LNAME]
        rs = stmt.executeQuery( sql: "select PNUM From testdb.playlist WHERE PID=" + playlistid +
        while (rs.next()) {
            //playlistnum : the number of music in User's play list
            playlistnum = rs.getInt( columnLabel: "PNUM");
        }
        rs.close();
        rs = stmt.executeQuery( sql: "select LNAME From testdb.level,testdb.playlist WHERE LID=POW
        while (rs.next()) {
            levelname = rs.getString( columnLabel: "LNAME");
        }
        rs.close();
```

"y"라고 했으면 [loginid]파라 미터를 이용해서 playlistid와 ssn정보를 얻어옵니다.

- user테이블과 playlist 테이블을 합쳐 ID=loginid 조건과 SSn=POWNER인 PID와 SSN 정보를 가져옵니다.

사용자의 [PNUM]칼럼을 확인해서 음악을 추가할 수 있는지 체크합니다.

- PID=playlistid 조건을 만족하는 PNUM 칼럼을 testdb.playlist 테이블에서 가져와 [playlistnum]변수에 저장합니다.
- Leve, playlist 테이블을 합쳐 LID=POWNER 조건과 POWNER=ssn조건을 만족하는 [LNAME]칼럼 정보를 가져옵니다. [levelname]변수에 저장합니다.

[levelname]정보를 이용해 사용자의 레벨에 따라 담을 수 있는 max음악 개수를 구해 [maxmusicnum]변수에 저장합니다.

Playlistnum 과 maxmusicnum 수를 비교해서 사용자가 음악을 추가할 수 있는지 체크합니다.

```
// maxmusicnum :  the MAX Number of music according to level
int maxmusicnum = (5 - leveler(levelname)) * 10;
if (playlistnum == maxmusicnum) {
    // User can't add the music in their play list because over the maxmusicnum
    System.out.println("WARNING : Your play list is full, please update the level first ");
    return;
} else {
    //add the music in the user's play list
```

음악을 추가할 수 있으면 추가하고 싶은 음악의 MID를 입력 받아 mid변수에 저장합니다.
추가 기능) 입력 받은 mid값이 사용자가 이미 playlist에 담은 곡인지 중복 체크합니다.

- CPID=playlistid조건과 CMID=mid조건을 만족시키는 CMID값을 testdb.musiccontain 테이블에서 찾아 옵니다.

그 값을 [checkmid]변수에 저장합니다. if조건 문을 사용해서 사용자가 입력한 mid넘버와 checkmid넘버를 비교합니다. 동일하면 dapflage를 참으로 변경하고 중복된 노래가 이미 플레이 리스트에 있다는 경고 창을 띄어줍니다.

```
} else {
    //add the music in the user's play list
    System.out.println("Input the MID number to add play list :");
    int mid = scan.nextInt();
    boolean dupflag=false;
    //duplicate music add is not allow
    rs = stmt.executeQuery( sql: "select CMID From testdb.musiccontain WHERE CPID="+playlistid+" AND CMI
    while (rs.next()){
        checkmid = rs.getInt( columnLabel: "CMID");
        if(checkmid==mid){
            dupflag=true;
            System.out.println("WARNING : There are duplicate music in your play list, ");
            break;
        }
    }
    rs.close();
```

추가 기능) 중복처리를 해준 입력 값 mid가 music store에 있는 값인지 체크합니다.

- mid 값과 testdb.music에 있는 MID칼럼과 값이 같은 행이 존재하면 checkmid에 값을 저장합니다.

존재하지 않으면 그런 음악이 없다는 경고 창을 띄어줍니다.

```
//check mid  : user input Music num exist in MUSIC db?
rs = stmt.executeQuery( sql: "select MID From testdb.music WHERE MID="+mid+";");
while (rs.next()){
    //if exist ---> come here! and change the checkmid != 0
    checkmid = rs.getInt( columnLabel: "MID");
        break;
}
if(checkmid == 0){
    // not exist ---> finish !
    dupflag=true;
    System.out.println("WARNING : There not exist music in MUSIC STORE,");
}

if(dupflag==true){
    System.out.println(" please rewrite the mid");
}else if(dupflag==false){
    //add the information in musiccontain table
```

중복처리와 올바른 값을 입력 받았으면 sql 문을 실행해서 입력 받은 mid를 user의 musiccontain table에 삽입합니다.

```
if(dupflag==true){
    System.out.println(" please rewrite the mid");
}else if(dupflag==false){
    //add the information in musiccontain table
    stmt.executeUpdate( sql: "insert into testdb.musiccontain " + "(CPID , CMID) VALUE (" + playlistid
    //for add the music like number get the information
    rs = stmt.executeQuery( sql: "select MLIKE From testdb.music WHERE MID=" + mid + ";");
    while (rs.next()) {
        musiclikenum = rs.getInt( columnLabel: "MLIKE");
        musiclikenum = musiclikenum + 1;
    }
    // update the music like number
    stmt.executeUpdate( sql: "UPDATE testdb.music " + "SET MLIKE=" + musiclikenum + " WHERE MID=" + mid
    rs.close();

    //update the play list number +1 when user add the music
    rs = stmt.executeQuery( sql: "select PNUM From testdb.playlist WHERE PID=" + playlistid + ";");
    while (rs.next()) {
        updateplaylistnum = rs.getInt( columnLabel: "PNUM");
        updateplaylistnum = updateplaylistnum + 1;
    }
    // update the music list num
    stmt.executeUpdate( sql: "UPDATE testdb.playlist " + "SET PNUM=" + updateplaylistnum + " WHERE PID=
    rs.close();

    System.out.println("Do you want to add more music ? ( y / n )");
    String moreans = scan.next();
    if (moreans.equals("n")) {
        storeflag = false;
    }
}
```

- playlistid와 mid값을 testdb.musiccontain 테이블에 삽입합니다.
- Testdb.music 테이블에 있는 MID=mid인 행의 MLIKE칼럼의 값을 가져와
- +1 연산을 한 뒤 UPDATE를 합니다.
- Testdb.playlist 테이블에 있는 PID=playlistid인 행의 PNUM칼럼의 값을 가져와
- +1 연산을 한 뒤 UPDATE를 합니다.

추가할 음악이 더 있냐고 물어보고 "y" / "n" 답변을 moreans에 저장해 if조건 문을 처리해서 반복합니다.

Nami의 PID는 23입니다. Nami는 2번 노래를 가지고 있습니다. →중복처리

CPID	CMID
21	2
23	2
21	4
NULL	NULL

```
Input the MID number to add play list :
2
WARNING : There are duplicate music in your play list,
 please rewrite the mid
Input the MID number to add play list :
```

현재 music store 에는 1번 노래가 존재하지 않습니다. →올바른 입력값처리

MID	MNAME	MSINGER	MLIKE	MMANAGE	MLYRIC
2	ice	icer	3	112233	iceiceice
3	iov	iover	2	112233	ioviioviov
4	ohо	ohoer	1	112233	ohoohooho
5	a	aer	0	112233	aaaaaaaaaaaaa
6	b	ber	0	112233	bbbbbbbbbbb
7	c	cer	0	112233	ccccccccccc
8	d	der	0	112233	ddddddddddd
9	e	eer	0	112233	eeeeeeeeee
10	f	fer	0	112233	fffffffffffff
11	a	aer	0	112233	aaaaaaaaaaaaa
12	h	her	0	112233	hhhhhhhhh
13	i	ier	0	112233	iiiiiii
NULL	NULL	NULL	NULL	NULL	NULL

```
======== USER LOGIN ========
Input your ID :
nam
Input you password :
nam
login success

======== USER PAGE ========
 1 : Go to the Play List
 2 : Music Store
 3 : Level Edit
 4 : Exit
2
========= MUSIC STORE ========
ID  M-NAME   M-SINGER
2       ice      icer
3       joy      joyer
4       pho      phoer
5       a        aer
6       b        ber
7       c        cer
8       d        der
9       e        eer
10      f        fer
11      g        ger
12      h        her
13      j        jer
Do you want to add the music? (y / n)
y
Input the MID number to add play list :
1
WARNING : There not exist music in MUSIC STORE,
 please rewrite the mid
Input the MID number to add play list :
```

3,4번 노래의 추가는 성공했습니다. PNUM의 개수가 2증가했고, musiccontain 테이블에도 2개의 항목이 추가되었습니다.

```
Input the MID number to add play list :
3
Do you want to add more music ? ( y / n )
y
Input the MID number to add play list :
4
Do you want to add more music ? ( y / n )
n
======== USER PAGE ========
 1 : Go to the Play List
 2 : Music Store
 3 : Level Edit
 4 : Exit
```

Playlist

PID	PNUM	POWNER
21	2	11111
23	3	33333
24	0	44444
25	0	55555
26	0	66666
NULL	NULL	NULL

musiccontain

CPID	CMID
21	2
23	2
23	3
21	4
23	4
NULL	NULL

3,4번 노래의 MLIKE값이 1씩 증가했습니다.

MID	MNAME	MSINGER	MLIKE	MMANAGE	MLYRIC
2	ice	icer	3	112233	iceiceice
3	iov	iover	3	112233	iovioviov
4	pho	phoer	2	112233	phoohooho
5	a	aer	0	112233	aaaaaaaaaaaaa
6	b	ber	0	112233	bbbbbbbbbbb
7	c	cer	0	112233	ccccccccccc
8	d	der	0	112233	ddddddddddd
9	e	eer	0	112233	eeeeeeeeee
10	f	fer	0	112233	fffffffffffff
11	g	ger	0	112233	ggggggggggg
12	h	her	0	112233	hhhhhhhhh
13	i	ier	0	112233	iiiiiii
NULL	NULL	NULL	NULL	NULL	NULL

✓ **3-3 userbutton == 3 사용자 레벨 변경하기**

LevelRequest() 함수를 이용합니다.

db접근을 위한 connection과 load를 먼저 수행해줍니다

일단 사용자의 loginid정보를 이용해 현재 레벨과 SSN을 파악합니다.

- ID = loginid조건과 SSN=LID조건을 만족하는 행을 testdb.user, testdb.leve에서 찾습니다. LNAME칼럼을 userlevelname 변수에 저장하고 SSN칼럼을 ssn변수에 저장합니다.

ssn변수를 이용해서 현재 사용자가 가지고 있는 음악의 개수를 찾아 currentmusicnum변수에 저장합니다.

- POWNER=ssn조건을 만족하는 PNUM칼럼을 testdb.playlist테이블에서 찾습니다.

사용자의 현재 레벨 정보를 알려줍니다.

사용자는 레벨 upgrade와 downgrade를 선택할 수 있습니다.

사용자는 레벨 변경 페이지에서 나갈 수 있습니다.

Userlevelbutton을 사용해 사용자가 원하는 기능을 입력 받습니다.

Leveler() 함수를 사용해서 userlevelname을 알맞은 숫자로 반환 받습니다. userlevelnameNum

userlevelnameNum의 값이 1인 경우는 플레유저입니다.

플레유저는 더 이상 upgrade level을 할 수 없습니다.

플레유저는 G, S, B로 선택해서 downgrade level을 할 수 있습니다.

userlevelnameNum의 값이 2인경우는 골드유저입니다.

골드유저는 P로 upgrade level을 할 수 있습니다.

골드유저는 S, B로 선택해서 downgrade level을 할 수 있습니다.

userlevelnameNum의 값이 3인 경우는 실버유저입니다.

실버유저는 P, S로 선택해서 upgrade level을 할 수 있습니다.

실버유저는 B로 downgrade level을 할 수 있습니다.

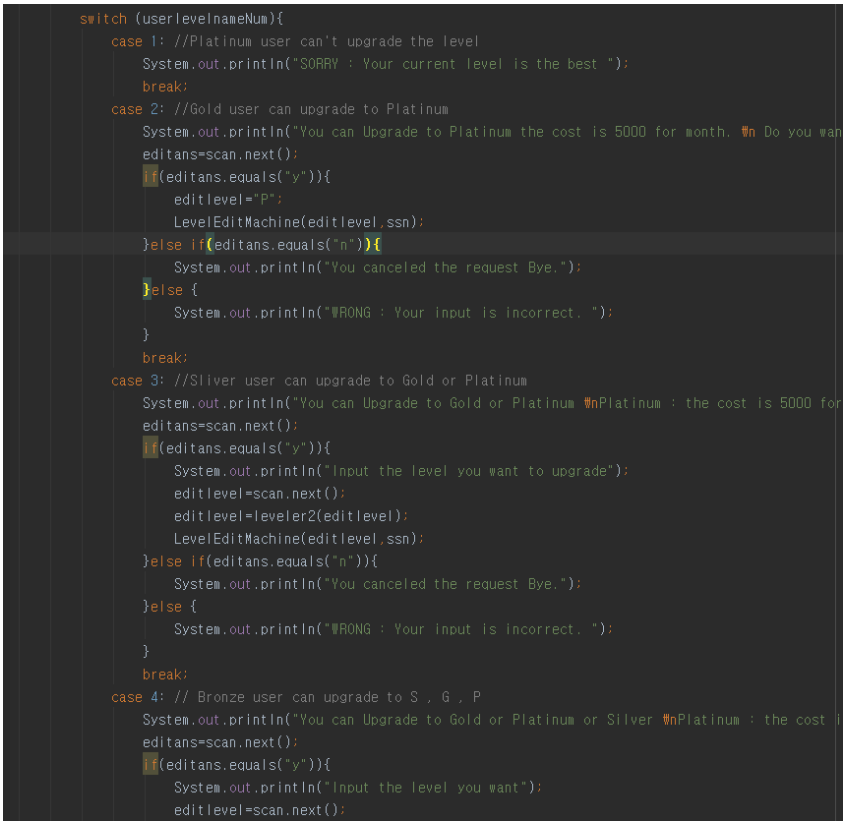
userlevelnameNum의 값이 4인 경우는 브론즈유저입니다.

브론즈유저는 S, G, P로 선택해서 upgrade level을 할 수 있습니다.

브론즈유저는 더 이상 downgrade level을 할 수 없습니다.

```java
public static void LevelRequest(String loginid){

    Scanner scan = new Scanner(System.in);
    Connection conn = null;
    Statement stmt = null;
    ResultSet rs = null;
    String userlevelname=null;
    int currentmusicnum=0;
    boolean levelflag = true;
    int levelbutton;
    int ssn=0;

    String driver = "com.mysql.jdbc.Driver";
    String url = "jdbc:mysql://localhost:3307/testdb?useSSL=false";
    String username = "root";
    String password = "4558";

    try {
        Class.forName(driver); //load the db
        conn = DriverManager.getConnection(url, username, password);
        stmt = conn.createStatement();

        while (levelflag) {
            System.out.println("======== LEVEL REQUEST ========");
            //get the user's Level name [LNAME] by use loginid
            rs = stmt.executeQuery( sql: "select * From testdb.user,testdb.level  WHERE ID='" + loginid +
            while(rs.next()) {
                userlevelname = rs.getString( columnLabel: "LNAME");
                ssn=rs.getInt( columnLabel: "SSN");
            }
            rs.close();
            // get the user's current music num [PNUM] in playlist db
            rs = stmt.executeQuery( sql: "select PNUM From testdb.playlist  WHERE POWNER="+ssn+";");
            while(rs.next()) {
                currentmusicnum = rs.getInt( columnLabel: "PNUM");
            }
            rs.close();

            System.out.println("Your current level is " + userlevelname);
            int userlevelnameNum = leveler(userlevelname);
            String editans = null;
            String editlevel= null;
            System.out.println(" 1 : Upgrade \n 2 : Downgrade \n 3 : Exit");

            levelbutton= scan.nextInt();
            scan.nextLine();

            switch(levelbutton) {
                case 1: // upgrade
```

✧ 3-3-1 userlevelbutton == 1 사용자 레벨 업그레이드 하기



```
switch (userlevelnameNum){
    case 1: //Platinum user can't upgrade the level
        System.out.println("SORRY : Your current level is the best ");
        break;
    case 2: //Gold user can upgrade to Platinum
        System.out.println("You can Upgrade to Platinum the cost is 5000 for month. ₩n Do you wan
        editans=scan.next();
        if(editans.equals("y")){
            editlevel="P";
            LevelEditMachine(editlevel,ssn);
        }else if(editans.equals("n")){
            System.out.println("You canceled the request Bye.");
        }else {
            System.out.println("₩RONG : Your input is incorrect. ");
        }
        break;
    case 3: //Sliver user can upgrade to Gold or Platinum
        System.out.println("You can Upgrade to Gold or Platinum ₩nPlatinum : the cost is 5000 for
        editans=scan.next();
        if(editans.equals("y")){
            System.out.println("Input the level you want to upgrade");
            editlevel=scan.next();
            editlevel=leveler2(editlevel);
            LevelEditMachine(editlevel,ssn);
        }else if(editans.equals("n")){
            System.out.println("You canceled the request Bye.");
        }else {
            System.out.println("₩RONG : Your input is incorrect. ");
        }
        break;
    case 4: // Bronze user can upgrade to S , G , P
        System.out.println("You can Upgrade to Gold or Platinum or Silver ₩nPlatinum : the cost i
        editans=scan.next();
        if(editans.equals("y")){
            System.out.println("Input the level you want");
            editlevel=scan.next();
```

✧ 3-3-2 levelbutton == 2 사용자 레벨 다운그레이드 하기

추가 기능)***만약 사용자의 현재 레벨이 S인상 태에서 음악이 12개 있다고 가정해 보자 이 사용자가 B로 레벨다운을 요청했을 경우 우선 음악을 먼저 삭제하라고 경고창을 띄어준다.

왜냐하면 사용자의 레벨에 따라 playlist 에 담을 수 있는 곡의 개수가 다르기 때문이다.

P: 40개

G: 30개

S: 20개

B: 10개

```
case 2: // down grade
    switch (userlevelnameNum){
        case 1: // platinum B S G
            System.out.println("You can Downgrade to Bronze or Silver of Gold₩nGold : the cost is 400
            editans=scan.next();
            if(editans.equals("y")){
                System.out.println("Input the level you want to downgrade");
                editlevel=scan.next();
                editlevel=leveler2(editlevel);
                if(DowngradeRight(currentmusicnum,editlevel)==1){
                    LevelEditMachine(editlevel,ssn);
                }else{
                    System.out.println("₩ARNING : You can't downgrade level , please delete the music
                }
            }else if(editans.equals("n")){
                System.out.println("You canceled the request Bye.");
            }else {
                System.out.println("₩RONG : Your input is incorrect. ");
            }
            break;
        case 2: // gold B S
            System.out.println("You can Downgrade to Bronze or Silver ₩nSliver : the cost is 3000 for
            editans=scan.next();
            if(editans.equals("y")){
                System.out.println("Input the level you want to downgrade");
                editlevel=scan.next();
                editlevel=leveler2(editlevel);
                if(DowngradeRight(currentmusicnum,editlevel)==1){
                    LevelEditMachine(editlevel,ssn);
                }else{
                    System.out.println("₩ARNING : You can't downgrade level, please delete the music
                }
            }else if(editans.equals("n")){
                System.out.println("You canceled the request Bye.");
            }else {
                System.out.println("₩RONG : Your input is incorrect. ");
            }
            break;
        case 3: // silver --> B
            System.out.println("You can Downgrade to Bronze the cost is 2000 for month.
            editans=scan.next();
            if(editans.equals("y")){
                editlevel="B";
                if(DowngradeRight(currentmusicnum,editlevel)==1){
                    LevelEditMachine(editlevel,ssn);
                }else{
                    System.out.println("₩ARNING : You can't downgrade level , please de
                }
            }else if(editans.equals("n")){
                System.out.println("You canceled the request Bye.");
            }else {
                System.out.println("₩RONG : Your input is incorrect. ");
            }
            break;
        case 4: // bronze --> no
            System.out.println("SORRY : Your current level is the lowest ");
            break;
    }
    break;
```

UPGAGE실험

Zoro의 PID는 24이다. Zoro의 level은 B이다. 10개의 음악을 담을 수 있다.

NAME	SEX	SSN	ID	PSWD	UMANAGE
luffy	m	11111	luf	luf	112233
nami	w	33333	nam	nam	112233
zoro	m	44444	zor	zor	112233
usopp	m	55555	uso	uso	112233
chopper	w	66666	cho	cho	112233
NULL	NULL	NULL	NULL	NULL	NULL

LNAME	LID	LMANAGE
P	11111	112233
S	33333	112233
B	44444	112233
P	55555	112233
B	66666	112233
NULL	NULL	NULL

PID	PNUM	POWNER
21	2	11111
23	3	33333
24	0	44444
25	0	55555
26	0	66666
NULL	NULL	NULL

Zoro가 음악상점에 가서 음악을 10개를 구입한다.

```
========= START PAGE ========
Welcome to the Music World!
 Choose the Number !
 1 : New Register
 2 : UserLogin
 3 : AdminLogin
 4 : Exit
2
======== USER LOGIN ========
Input your ID :
zor
Input you password :
zor
login success

======== USER PAGE ========
 1 : Go to the Play List
 2 : Music Store
 3 : Level Edit
 4 : Exit
2
========= MUSIC STORE ========
ID  M-NAME   M-SINGER
2       ice     icer
3       joy     joyer
4       pho     phoer
5       a       aer
6       b       ber
7       c       cer
8       d       der
9       e       eer
10      f       fer
11      g       ger
12      h       her
13      j       jer
Do you want to add the music? (y / n)
y
Input the MID number to add play list :
2
Do you want to add more music ? ( y / n )
y
Input the MID number to add play list :
3
Do you want to add more music ? ( y / n )
y
Input the MID number to add play list :
```

.... 2~11

결과 테이블

NAME	SEX	SSN	ID	PSWD	UMANAGE
luffy	m	11111	luf	luf	112233
nami	w	33333	nam	nam	112233
zoro	m	44444	zor	zor	112233
usopp	m	55555	uso	uso	112233
chopper	w	66666	cho	cho	112233
NULL	NULL	NULL	NULL	NULL	NULL

LNAME	LID	LMANAGE
P	11111	112233
S	33333	112233
B	44444	112233
P	55555	112233
B	66666	112233
NULL	NULL	NULL

PID	PNUM	POWNER
21	2	11111
23	3	33333
24	10	44444
25	0	55555
26	0	66666
NULL	NULL	NULL

Zoro의 PNUM가 10이 되었습니다.

이 상황에서 음악을 더 구입해보겠습니다.

```
Input the MID number to add play list :
11
Do you want to add more music ? ( y / n )
y
WARNING : Your play list is full, please update the level first
======== USER PAGE ========
 1 : Go to the Play List
 2 : Music Store
 3 : Level Edit
 4 : Exit
```

사용자의 플레이 리스트가 다 찼다는 경고 창을 받았습니다.

Zoro의 레벨을 S로 바꾸로 음악을 다시 한번 더 구입해보겠습니다.

```
======== USER PAGE ========
 1 : Go to the Play List
 2 : Music Store
 3 : Level Edit
 4 : Exit
3
======== LEVEL REQUEST ========
Your current level is B
 1 : Upgrade
 2 : Downgrade
 3 : Exit
1
You can Upgrade to Gold or Platinum or Silver
Platinum : the cost is 5000 for month.
Gold : the cost is 4000 for month.
Silver : the cost is 3000 for month.
 Do you want to upgrade your level? ( y / n )
y
Input the level you want
s
======== LEVEL REQUEST ========
Your current level is S
 1 : Upgrade
 2 : Downgrade
 3 : Exit
3
======== USER PAGE ========
 1 : Go to the Play List
 2 : Music Store
 3 : Level Edit
 4 : Exit
2
========= MUSIC STORE ========
ID  M-NAME   M-SINGER
2       ice     icer
3       joy     joyer
4       pho     phoer
5       a       aer
6       b       ber
7       c       cer
8       d       der
9       e       eer
10      f       fer
11      g       ger
12      h       her
13      j       jer
Do you want to add the music? (y / n)
y
Input the MID number to add play list :
11
WARNING : There are duplicate music in your play list,
 please rewrite the mid
Input the MID number to add play list :
12
Do you want to add more music ? ( y / n )
```

결과 테이블을 보겠습니다.

User

NAME	SEX	SSN	ID	PSWD	UMANAGE
luffy	m	11111	luf	luf	112233
nami	w	33333	nam	nam	112233
zoro	m	44444	zor	zor	112233
usopp	m	55555	uso	uso	112233
chopper	w	66666	cho	cho	112233
NULL	NULL	NULL	NULL	NULL	NULL

playlist

PID	PNUM	POWNER
21	2	11111
23	3	33333
24	11	44444
25	0	55555
26	0	66666
NULL	NULL	NULL

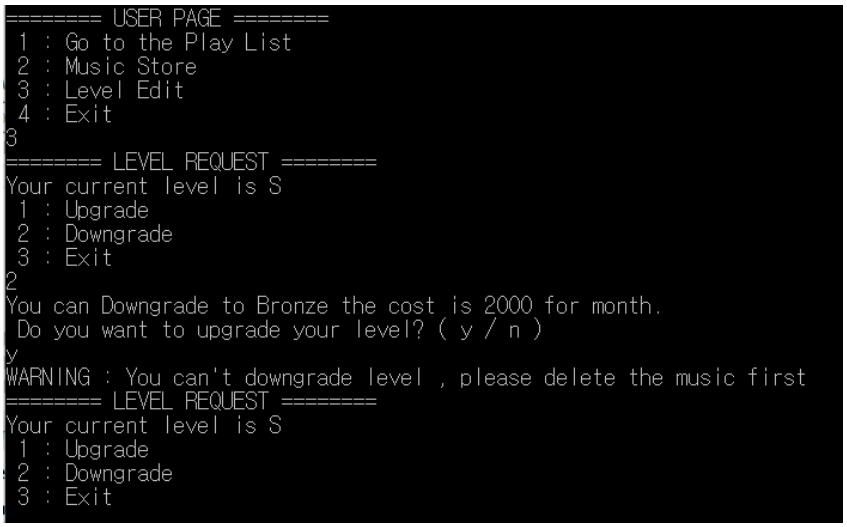
level

LNAME	LID	LMANAGE
P	11111	112233
S	33333	112233
S	44444	112233
P	55555	112233
B	66666	112233
NULL	NULL	NULL

Zoro의 레벨이 B에서 S로 변경되었고 PNUM의 수가 11이 되었습니다.

DOWNGRADE실험

Zoro가 B로 level down을 요청했으나 거부되었습니다.



경고창의 말대로 B의 playlist 사이즈는 10이므로 1개의 곡을 삭제하고 다시 요청하겠습니다.

```
======== USER PAGE ========
 1 : Go to the Play List
 2 : Music Store
 3 : Level Edit
 4 : Exit
3
======== LEVEL REQUEST ========
Your current level is S
 1 : Upgrade
 2 : Downgrade
 3 : Exit
2
You can Downgrade to Bronze the cost is 2000 for month.
 Do you want to upgrade your level? ( y / n )
y
======== LEVEL REQUEST ========
Your current level is B
 1 : Upgrade
 2 : Downgrade
 3 : Exit
```

level요청을 성공적으로 수행했습니다.
결과 table을 확인하겠습니다.

NAME	SEX	SSN	ID	PSWD	UMANAGE
luffy	m	11111	luf	luf	112233
nami	w	33333	nam	nam	112233
zoro	m	44444	zor	zor	112233
usopp	m	55555	uso	uso	112233
chopper	w	66666	cho	cho	112233
NULL	NULL	NULL	NULL	NULL	NULL

PID	PNUM	POWNER
21	2	11111
23	3	33333
24	10	44444
25	0	55555
26	0	66666
NULL	NULL	NULL

LNAME	LID	LMANAGE
P	11111	112233
S	33333	112233
B	44444	112233
P	55555	112233
B	66666	112233
NULL	NULL	NULL

Zoro의 PNUM이 10으로 셋팅되었고 레벨이 B로 바뀌었습니다.

LevelEditMachine() 함수를 정의했습니다. upgrade downgrade의 동일한 수행부분을 함수로 묶었습니다.

```
//Level Edit machine
public static void LevelEditMachine(String level ,int ssn){
    Connection conn = null;
    Statement stmt = null;
    String driver = "com.mysql.jdbc.Driver";
    String url = "jdbc:mysql://localhost:3307/testdb?useSSL=false";
    String username = "root";
    String password = "4558";

    try {
        Class.forName(driver); //load the db
        conn = DriverManager.getConnection(url, username, password);
        stmt = conn.createStatement();

        // update the level table
        stmt.executeUpdate( sql: "UPDATE testdb.level " + "SET LNAME='" + level + "' WHERE LID=" + ssn +

    }catch (ClassNotFoundException cnfe) {
        System.out.println("class not found");
    } catch (SQLException e) {
        System.out.println("sqlexception error₩n");
        e.printStackTrace();
    }
}
```

- 입력 받은 level정보를 파라 미터로 받은ssn을 이용해서 testdb.level 테이블의 LNAME칼럼의 정보를 UPDATE했습니다.

DowngradeRight() 함수를 정의했습니다. Downgrade를 요청한 사용자의 현재 playlist의 음악 개수를 파라 미터로 받아서 limitnum을 구해 조건 문으로 비교했습니다. 변경할 수 없으면 return 0 변경할 수 있으면 return 1 값을 반환받습니다.

```
// if Down level input is right use this function
public static int DowngradeRight(int currentmusicnum , String level){
    int limitnum = (5-leveler(level))*10;

    if(currentmusicnum > limitnum){
        return 0;
    }
    return 1;
}
```