이전 설계와 달라진 점

1. Attribute 이름들을 알기 쉽게 number, name 등으로 전부 변경하였습니다.
2. User managing, music managing 등의 테이블은 삭제하였습니다.
3. Music의 가수에 대한 테이블을 없애고 music에 singer attribute를 추가하였습니다.
4. Playlist 테이블에 primary key로 number를 추가하고, name을 primary key에서 삭제하였습니다. 대신 이름도 unique constraint으로 중복되지 않도록 하였습니다. 이에 따라 contain 테이블에서도 playlist를 참조하는 foreign key로 playlist_name을 삭제하고 playlist_number로 변경하였습니다.
5. 로그인 기능을 위해 user 테이블과 manager 테이블에 id와 pw attribute를 추가하였습니다. Id는 unique constraint으로 중복되는 id는 없도록 하였습니다.

코드 설명

시작 메뉴는

1. Manager로 로그인 2. User로 로그인 3. 회원가입

이렇게 3개의 메뉴로 시작합니다.

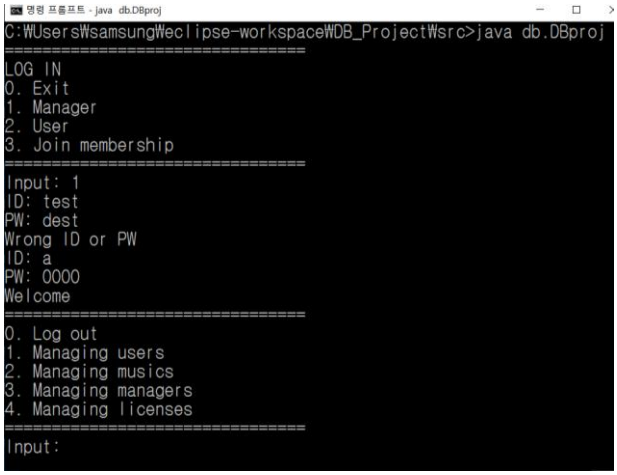
1. Manager로 로그인

Manager로 로그인을 선택하면 id와 pw를 입력하는데, manager 테이블에서 id와 pw를 조회하여 일치한다면 로그인이 성공하고, 아니면 다시 입력해야 합니다.

사용되는 sql문:

"select id from manager"

"select pw from manager where id='" + id + "'"



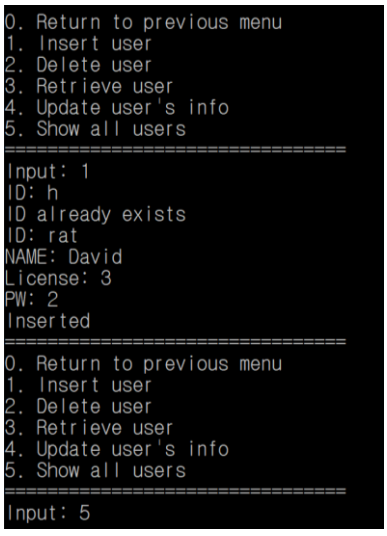
로그인을 하고 나면

1. User 관리 2. Music 관리 3. Manager 관리 4. License 관리

이렇게 4개의 메뉴가 있습니다. Manager 관리는 president라는 아이디로 접속해야만 이용 가능합니다.

User와 music, manager에는 각각 1. 삽입 2. 삭제 3. 검색 4. 정보 업데이트 5. 모든 유저 보여주기 이렇게 5개의 기능들이 존재합니다.

-insert



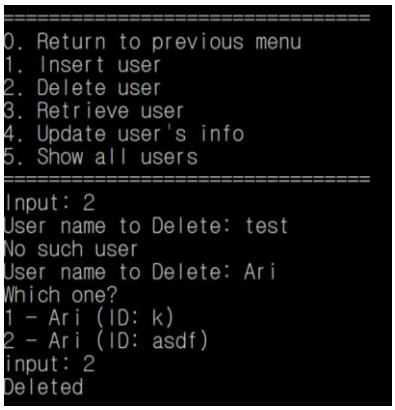
Insert는 각 테이블에 필요한 attribute를 순서대로 입력 받고 id가 중복되는 경우가 아니라면 insert합니다.

사용되는 sql문:

"select max(number) from user"

"insert into user values(" + number + ",'" + name + "',0," + license + ",'" + id + "','" + pw + "')"

-delete



Delete는 삭제할 이름을 받아서 삭제합니다. 만약 없는 이름이면 다시 입력 받고, 두 개 이상의 이름이 존재하면 옆에 id 또는 가수와 함께 다시 선택할 수 있도록 합니다.

사용되는 sql문:

"select count(number) from user where name='" + name + "'"

"select number, name, id from user where name='" + name + "'"

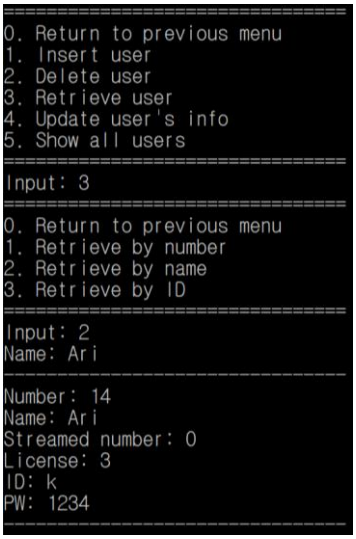
"delete from streaming where user_number=" + number);

“delete from contain where playlist_user=" + number

“delete from playlist where user_number=" + number

"delete from user where number=" + number

-retrieve



Retrieve는 이름으로 검색할지 id로 검색할지, music의 경우 음악이름, 가수, 장르로 검색할지 선택합니다. 그 다음 입력을 받아서 일치하는 정보를 모두 보여줍니다. 없다면 출력되지 않습니다.

사용되는 sql문:

"select * from user where name='" + name + "'"

"select * from user where id='" + id + "'"

-update

Update는 어떤 tuple을 업데이트할지 이름으로 입력을 받습니다. 그 다음 어떤 attribute를 업데이트 할 지와 어떤 값으로 업데이트를 할지 차례로 입력 받아 정보를 수정합니다.

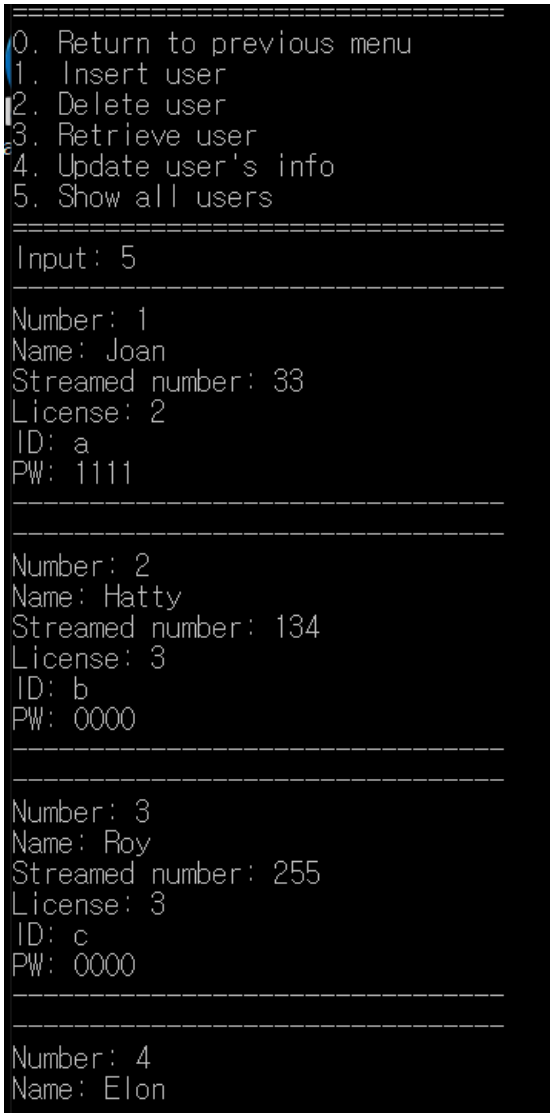
사용되는 sql문:

"select count(number) from user where name='" + name + "'"

"select number, name, id from user where name='" + name + "'"

"update user set id='" + stringValue + "' where number=" + number

-show all



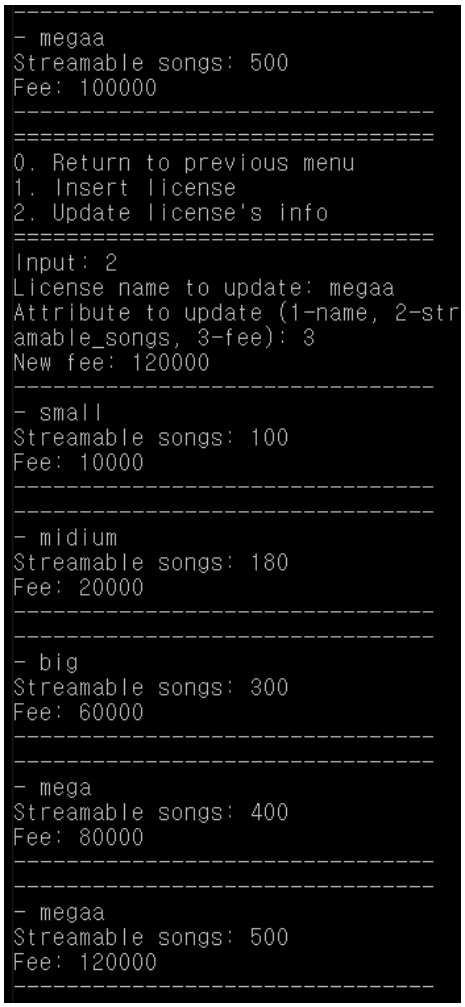
table내의 모든 tuple들의 정보를 보여줍니다.

사용되는 sql문:

"select * from user"

4. Manage license의 경우는 먼저 모든 이용권 정보를 보여주고 같은 방식으로 삽입과 업데이트만

가능합니다.



사용되는 sql문:

"select * from license"

"select max(number) from license"

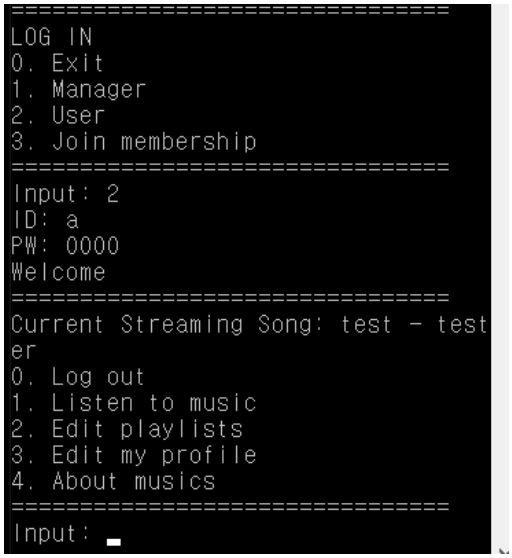
"insert into license values(" + number + ",'" + name + "'," + streamableSongs + "," + fee + ")"

"select exists(select number from license where name='" + name + "') as success"

"select number from license where name='" + name + "'"

"update license set name='" + newName + "' where number=" + number

2. User로 로그인



유저로 로그인하게 되면

1. Listen to music 2. Edit playlists 3. Edit my profile 4. About music

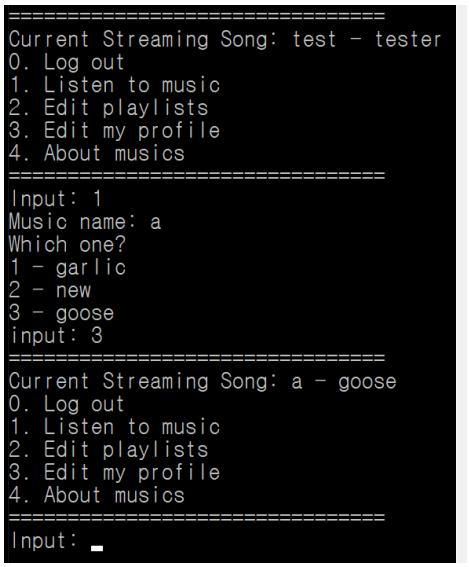
이렇게 4개의 메뉴가 있습니다.

사용되는 sql문:

"select number, id from user"

"select pw from user where id='" + id + "'"

1. Listen to music



음악듣기는 어떤 음악을 들을 지 이름으로 입력 받고, 같은 이름이 있다면 가수까지 함께 보여줘서 음악을 선택합니다. 선택된 음악이 streaming 테이블에 있던 기존 음악이 삭제되고 선택된 음악이 올라갑니다. 메뉴 위의 Current Streaming Song의 음악이 바뀝니다.

사용되는 sql문:

"select count(name) from music where name='" + name + "'"

"delete from streaming where user_number=" + userNumber

"insert into streaming values(" + userNumber + "," + musicNumber + ")"

"update user set streamed_num=streamed_num+1 where number=" + userNumber

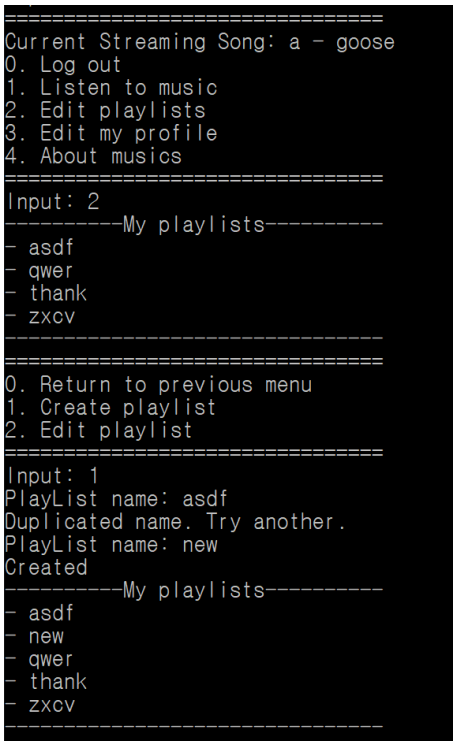
"update music set played_number=played_number+1 where number=" + musicNumber

"select m.name, m.singer from streaming as s, music as m where s.user_number="++ userNumber + " and s.music_number=m.number"

2. Playlist 편집

먼저 현재 로그인한 유저의 플레이리스트 목록을 보여줍니다. 그 다음 플레이리스트 생성과 편집을 선택할 수 있습니다.

생성은 중복되지 않는 이름을 입력 받아서 추가합니다.



사용되는 sql문:

"select name from playlist where user_number=" + userNumber + " order by name"

"select distinct m.name, m.singer from playlist as p, contain as c, music as m where p.number=' +

playlistNumber+ " and c.playlist_user=" + userNumber+"and c.playlist_number=p.number and

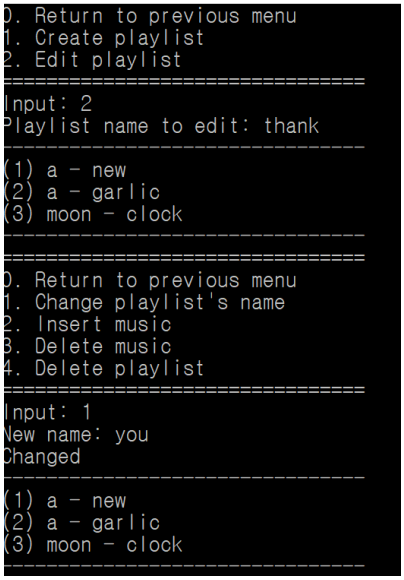
c.music_number=m.number order by m.name"

편집은 먼저 편집할 플레이리스트를 이름으로 입력 받아서 선택하고, 그 플레이리스트에 있는 음악 목록을 보여줍니다.

편집에는 1. 플레이리스트 이름 바꾸기 2. 음악 삽입 3. 음악 삭제 4. 플레이리스트 삭제

이렇게 4개의 메뉴가 있습니다.

-change playlist's name



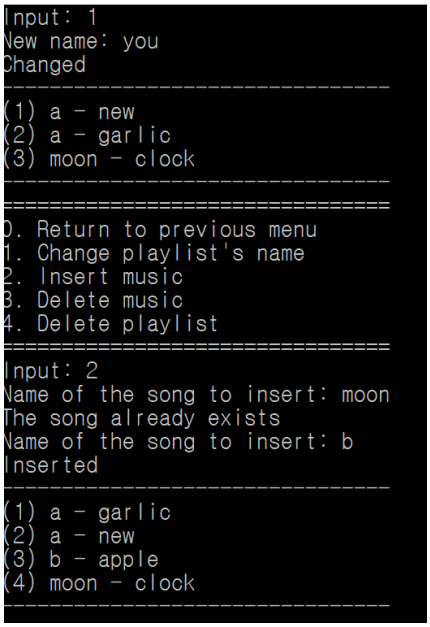
새로 변경할 이름을 입력 받아서 중복되지 않는다면 바꿔줍니다.

사용되는 sql문:

"select name from playlist where user_number=" + userNumber

"update playlist set name='" + newName + "' where number=" + playlistNumber

-insert music



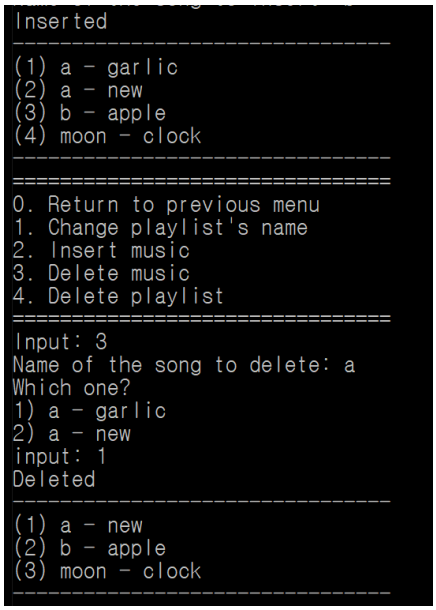
새로 추가할 음악의 이름을 입력 받아서 플레이리스트에 이미 존재하는 음악이 아니라면 추가합니다.

사용되는 sql문:

"select max(number) from playlist"

"insert into playlist values(" + userNumber + "," + number + ",'" + playlistName + "')"

-delete music



삭제할 음악을 이름으로 입력 받아서, 같은 이름의 음악이 있다면 가수로 함께 선택하여 선택된 음악을 삭제합니다.

사용되는 sql문:

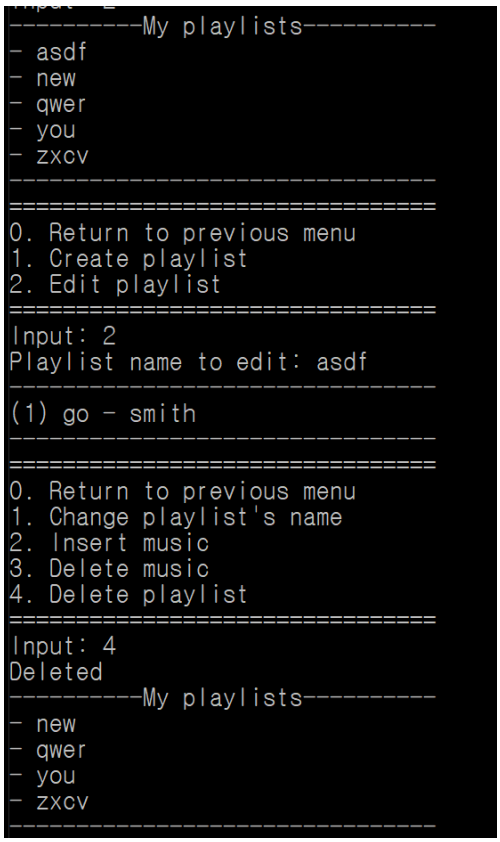
"select distinct m.name, m.singer, m.number from playlist as p, contain as c, music as m where

c.playlist_user='+userNumber + " and c.playlist_number=" + playlistNumber" and

c.music_number=m.number and m.name='" + musicName + "'"

"delete from contain where music_number=" + musicNumber

-delete playlist



선택된 플레이리스트를 삭제합니다. 이 때 contain 테이블의 음악들도 함께 삭제됩니다.

사용되는 sql문:

"delete from contain where playlist_number=" + playlistNumber+ " and playlist_user=" + userNumber

"delete from playlist where number=" + playlistNumber + " and user_number="+ userNumber

3. 프로필 편집

```
------------Profile-----------
ID: a
PW: 0000
Name: Joe
License: mega
Streamed number: 33
================================
0. Return to previous menu
1. Change name
2. Change license
3. Change password
================================
Input: 1
New name: Joan
Changed
------------Profile-----------
ID: a
PW: 0000
Name: Joan
License: mega
Streamed number: 33
```

```
================================
Input: 2
-----Currenct using license-----
License: mega
Streamable songs: 400
Fee: 80000
==============Menu==============
1. small
Streamable songs: 100
Fee: 10000
2. midium
Streamable songs: 180
Fee: 20000
3. big
Streamable songs: 300
Fee: 60000
4. mega
Streamable songs: 400
Fee: 80000
5. megaa
Streamable songs: 500
Fee: 120000
================================
Which license do you want?
Input: 2
Changed
------------Profile-----------
ID: a
PW: 0000
Name: Joan
License: midium
Streamed number: 33
```

```
------------Profile-----------
ID: a
PW: 0000
Name: Joan
License: midium
Streamed number: 33
================================
0. Return to previous menu
1. Change name
2. Change license
3. Change password
================================
Input: 3
New PW: 1111
Changed
------------Profile-----------
ID: a
PW: 1111
Name: Joan
License: midium
Streamed number: 33
```

먼저 user의 프로필을 보여줍니다. 그리고 이름, 이용권, 비밀번호 등을 바꿀 수 있습니다.

사용되는 sql문:

"select u.name, u.streamed_num, l.name, u.id, u.pw from user as u,license as l where

u.license=l.number and u.number="+ userNumber

"update user set name='" + newName + "' where number=" + userNumber

"update user set license=" + newLicense + " where number=" + userNumber

"update user set pw='" + newPW + "' where number=" + userNumber

4. 음악 정보 조회

음악은 이름, 가수, 장르로 검색할 수 있습니다. 또 played_number순으로 음악의 랭킹을 확인할 수 있습니다.

-이름으로 검색

```
==============================
0. Return to previous menu
1. Retrieve by name
2. Retrieve by singer
3. Retrieve by genre
4. Show music ranking
==============================
Input: 1
Music name:a
------------------------------
name: a
singer: new
genre: pop
------------------------------
------------------------------
name: a
singer: garlic
genre: pop
------------------------------
------------------------------
name: a
singer: goose
genre: classic
------------------------------
```

-가수로 검색

```
==============================
0. Return to previous menu
1. Retrieve by name
2. Retrieve by singer
3. Retrieve by genre
4. Show music ranking
==============================
Input: 2
Singer:clock
------------------------------
name: moon
singer: clock
genre: ballad
------------------------------
```

-장르로 검색

```
================================
0. Return to previous menu
1. Retrieve by name
2. Retrieve by singer
3. Retrieve by genre
4. Show music ranking
================================
Input: 3
Genre :pop
--------------------------------
name: fire
singer: 2ne1
genre: pop
--------------------------------
--------------------------------
name: a
singer: new
genre: pop
--------------------------------
--------------------------------
name: a
singer: garlic
genre: pop
--------------------------------
--------------------------------
name: test
singer: tester
genre: pop
--------------------------------
```

-음원 순위 보기

```
================================
0. Return to previous menu
1. Retrieve by name
2. Retrieve by singer
3. Retrieve by genre
4. Show music ranking
================================
Input: 4
--------------------------------
1. fire - 2ne1
2. go - smith
3. a - new
4. a - garlic
5. grass - back
6. c - banana
7. test - tester
8. a - goose
9. b - apple
10. moon - clock
11. bamm - bam
--------------------------------
```

사용되는 sql문:

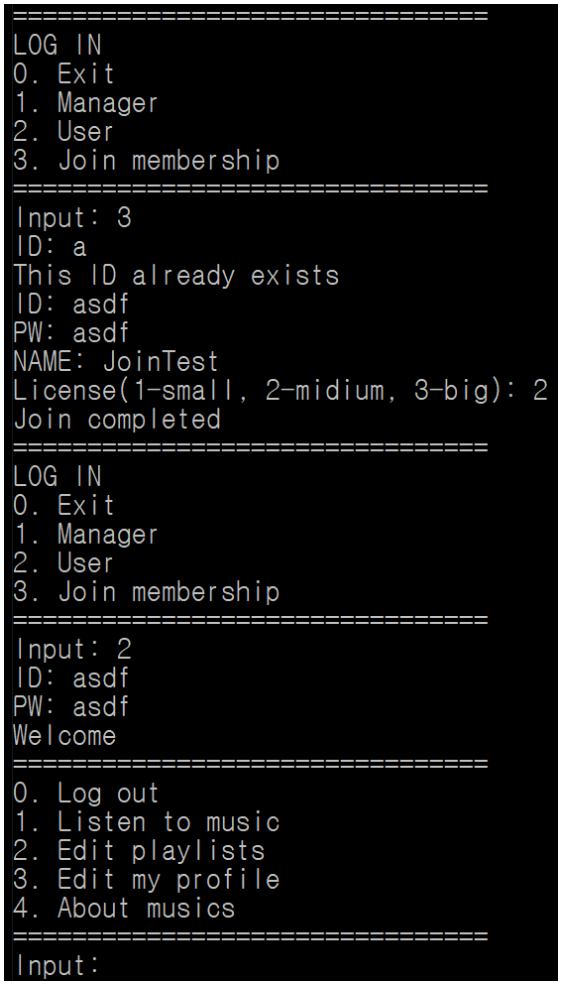
"select name, singer, genre, played_number from music where name='" + name + "' order by played_number desc"

"select exists(select name, singer, genre, played_number from music where singer='" + singer + "') as success"

"select exists(select name, singer, genre, played_number from music where genre='" + genre + "') as success"

"select name, singer from music order by played_number desc"

3. 회원 가입



새로 가입할 user의 id, pw, 이름, 이용권 정보를 차례로 입력 받아 user table에 추가해줍니다. 이때 이미 존재하는 id라면 재입력하게 하여 id가 중복되지 않도록 합니다. 가입한 아이디로 바로 user 로그인이 가능합니다.

사용되는 sql문:

"select id from user"

"select max(number) from user"

"insert into user values(" + number + ",'" + name + "',0," + license + ",'" + id + "','" + pw + "')"