

데이터베이스 기말 프로젝트

아이돌-팬 수익 관리

201511960 경영학과 윤민수

INDEX



01. DB데이터 소개

개체(Entity) 소개

소속사

소속사번호 (PK)
소속사이름
소속사대표이름
소속사위치

아이돌그룹

아이돌번호 (PK)
소속사번호 (FK)
아이돌이름
그룹유형 (성별)
데뷔일

멤버

멤버번호 (PK)
아이돌번호 (FK)
멤버성별
멤버이름
멤버생년월일

매니저

매니저번호 (PK)
소속사번호 (FK)
아이돌번호 (FK)
매니저이름

팬

팬번호 (PK)
성별
이름
생년월일
이메일
전화번호

콘서트

콘서트번호 (PK)
아이돌번호 (FK)
콘서트이름
콘서트장이름

프로그램

프로그램번호 (PK)
프로그램이름
방영날짜

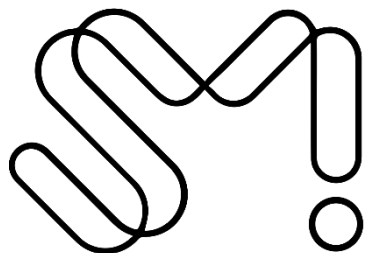
굿즈

굿즈번호 (PK)
아이돌번호 (FK)
굿즈이름
굿즈가격

01. DB데이터 소개

데이터 소개

소속사(3개)



SM
ENTERTAINMENT



01. DB데이터 소개

데이터 소개

아이돌(6그룹)

SM: 레드벨벳, EXO

JYP: 트와이스, GOT7

YG: 블랙핑크, 워너

콘서트(12개)

레드벨벳: "Red Room", "La Rouge"

EXO: "EXO PLANET #4", "EXO PLANET #5"

트와이스: "TWICELAND : The Opening", "Fantasy Park"

GOT7: "EYES ON YOU", "KEEP SPINNING"

블랙핑크: "BLACKPINK IN YOUR AREA", "PRIVATE STAGE"

워너: "EVERYWHERE TOUR", "WINNER CROSS TOUR"

※콘서트 날짜는 **Multivalued Attribute**로 설계해서 총 콘서트 개수는 31개

01. DB데이터 소개

데이터 소개

굿즈(6개)

레드벨벳: "김만봉"(35000원)

EXO: "포리디봉"(36000원)

트와이스: "캔디봉2"(36000원)

GOT7: "아가새봉"(35000원)

블랙핑크: "블망치"(30000원)

위너: "워너봉"(30000원)

프로그램(10개)

레드벨벳: "레벨업", "눈덩이프로젝트 "

EXO: "뜨거운순간엑소", "우리집옆에엑소가산다 "

트와이스: "트와이스의우아한사생활", "TWICETV5"

GOT7: "두림나이트", "갓세븐의하드캐리"

블랙핑크: "블핑하우스"

위너: "반달친구"

※프로그램은 팬의 방청 여부를 기록하기위해 여러 날짜를 방영한다고 가정하여 80개로 구성

01. DB데이터 소개

데이터 소개

※모두 임의로 가상의 데이터를 삽입함

팬(50명)

```
insert into fan(fnum,fname,sex,bd,email,pn) values
(1,"윤민수","m","1996-11-15","minsoo9615@naver.com","010-2595-6532"),
(2,"김민지","f","1994-10-24","minzko@hanmail.net","010-4985-6351"),
(3,"김지환","m","1986-06-30","ziwhan@naver.com","010-4445-7531"),
(4,"강혜정","f","1996-01-03","zeong@google.com","010-5362-7453"),
```

콘서트관람(251개)

```
insert into fanconcert(fnum,cdnum,income)values
(1,1,110000),
(1,23,110000),(1,31,130000),(1,27,110000),(1,14,120000),(1,8,110000),
(2,2,110000),(2,5,130000),(2,8,110000),(2,21,120000),(2,27,110000),
(3,8,110000),(3,17,130000),(3,21,110000),(3,25,120000),(3,30,110000),
(4,6,110000),(4,12,110000),(4,21,110000),(4,23,110000),(4,30,110000),
(5,12,110000),(5,14,110000),(5,16,110000),(5,18,110000),(5,21,110000).
```

굿즈구매(104개)

```
insert into fangoods(fgnum,fnum,gnum)values
(1,1,1),(2,1,2),(3,2,1),
(4,2,1),(5,6,1),(6,8,1),(7,10,1),(8,14,1),
(9,16,1),(10,17,1),(11,19,1),(12,21,1),(13,23,1),
(14,38,1),(15,33,1),(16,31,1),(17,29,1),(18,26,1),
(23,39,1),(22,41,1),(21,43,1),(20,48,1),(19,50,1),
(24,3,2),(25,5,2),(26,6,2),(27,11,2),(28,14,2),
(29,18,2),(30,21,2),(31,23,2),(32,28,2),(33,31,2),
(34,34,2),(35,38,2),(36,42,2),(37,48,2),(38,49,2),
(39,2,3),(40,4,3),(41,5,3),(42,6,3),(43,12,3),(44,14,3),
```

프로그램방청(600개)

```
insert into fanprogram(fnum,pnum,attend)values
(1,1,'n'),
(1,2,'y'),(1,4,'n'),(1,5,'y'),(1,8,'n'),(1,10,'y'),(1,13,'n'),(1,15,'n'),(1,18,'n'),
(1,28,'n'),(1,31,'n'),(1,38,'y'),(1,41,'y'),(1,48,'n'),(1,52,'n'),(1,56,'y'),(1,68,'n'),
(2,1,'y'),(2,5,'n'),(2,8,'y'),(2,7,'n'),(2,9,'y'),(2,16,'n'),(2,21,'n'),(2,23,'n'),
(2,31,'n'),(2,34,'n'),(2,41,'y'),(2,43,'y'),(2,45,'n'),(2,51,'n'),(2,55,'y'),(2,69,'n'),
(3,1,'y'),(3,5,'n'),(3,8,'y'),(3,7,'n'),(3,9,'y'),(3,16,'n'),(3,21,'n'),(3,23,'n'),
(3,31,'n'),(3,34,'n'),(3,41,'y'),(3,43,'y'),(3,45,'n'),(3,51,'n'),(3,55,'y'),(3,69,'n'),
(4,1,'y'),(4,5,'n'),(4,8,'y'),(4,7,'n'),(4,9,'y'),(4,16,'n'),(4,21,'n'),(4,23,'n'),
```

개발 환경



JAVA와 Eclipse를 활용



02. 개발환경 및 화면구성

클래스 구성

▷ DBAccess.java

DB에 로그인

▷ SelectView.java

기능 선택 메뉴

▷ Idol_income_r.java
▷ Idol_income.java

아이돌 수익

▷ Noshow_r.java
▷ Noshow.java

불참 명단

▷ Fan_insert.java
▷ Fansearch_r.java

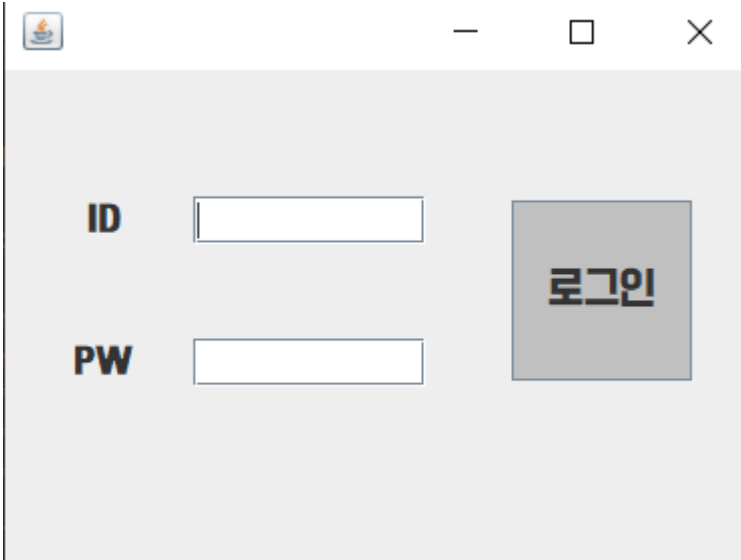
팬 조회/추가

▷ Idol_insert.java
▷ Idolsearch_r.java

아이돌 조회/추가

02. 개발환경 및 화면구성

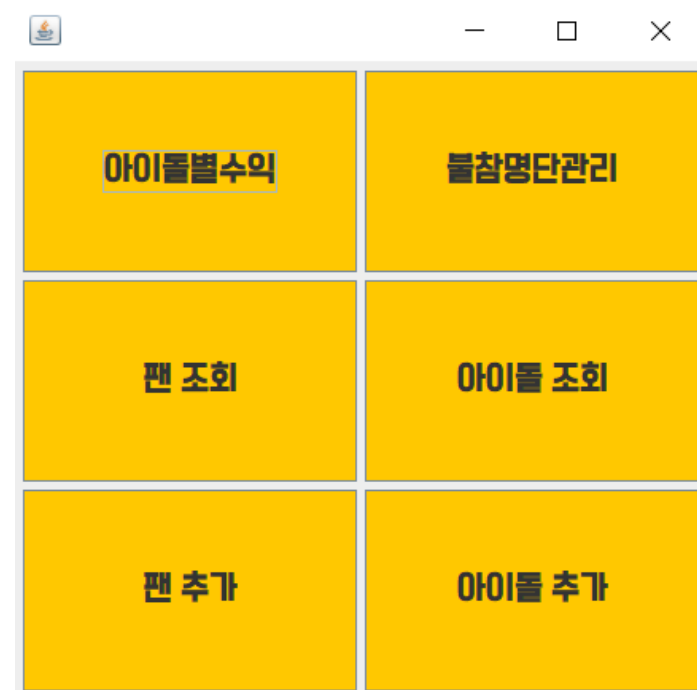
초기 화면



A mockup of a login window. It features a light gray background with a white border. On the left, there are two input fields: the top one is labeled 'ID' and the bottom one is labeled 'PW'. To the right of these fields is a gray rectangular button with the text '로그인' (Login) in white. The window has a standard title bar with a small icon on the left and minimize, maximize, and close buttons on the right.

〈로그인 화면〉

실제 DB에 접속하기위한 계정정보
(root 계정 사용)



A mockup of a main menu window. It has a white background and a standard title bar. The content is organized into a 3x2 grid of yellow rectangular buttons. The buttons contain the following text: Top-left '아이돌별수익' (Idol's Income), Top-right '블람명단관리' (Blam Name Management), Middle-left '팬 조회' (Fan Search), Middle-right '아이돌 조회' (Idol Search), Bottom-left '팬 추가' (Add Fan), and Bottom-right '아이돌 추가' (Add Idol).

〈초기 화면〉

6개의 메뉴로 구성(조회,삽입)
Grid Layout 사용

03. CASE1: 아이돌별 수익관리

아이돌별 수익관리 개요

사용자의 입력: 아이돌 이름, 수익을 측정할 기간

프로그램 내부 처리(Query):

1. 굿즈 수익에 대한 뷰를 만든다.
2. 콘서트 수익에 대한 뷰를 만든다.
3. Union 2번을 활용하여 굿즈와 콘서트를 구매한 팬의 정보를 취합한다.
4. 정보를 출력한다.
5. 뷰를 삭제한다.

프로그램 출력: 팬 이름, 전화번호, 기간 동안의 총 수익, 굿즈 구매횟수, 콘서트 관람횟수

03. CASE1: 아이돌별 수익관리

SQL

SQL : 뷰 생성

```
create view goods_i AS
select f.fnum, f.fname, f.pn, sum(g.price) as gi, count(distinct fg.fgnum) as gpc
from fan f, idol i, goods g, fangoods fg
where i.iname = ["레드벨벳"] and g.inum = i.inum and g.gnum = fg.gnum and f.fnum = fg.fnum
group by f.fnum
having count(*)>=1
order by sum(g.price);
```

————→ 굿즈

```
create view con_i AS
select f.fnum, f.fname, f.pn, sum(fc.income) as ci, count(*) as cpc
from fan f, idol i, concert c, concertdate cd, fanconcert fc
where i.iname = ["레드벨벳"] and c.inum = i.inum and c.connum = cd.connum and cd.condate > ["2017-04-15"] and fc.cdnum = cd.cdnum and fc.fnum = f.fnum
and c.connum = cd.connum
group by f.fnum
having count(*)>=1
order by sum(fc.income);
```

————→ 콘서트

[" "] : 사용자 입력부분

03. CASE1: 아이돌별 수익관리

SQL

SQL : 정보 취합 + 출력 + 뷰 삭제

```
select g.fname,g.pn,sum(g.gi + c.ci) AS income, g.gpc as gnum, c.cpc as cnum
from goods_i g , con_i c
where g.fname = c.fname and g.gi is not null and c.ci is not null
group by g.fname
union
```

→ 굿즈 + 콘서트

```
select g.fname,g.pn,sum(g.gi) AS income, g.gpc, 0 as cnum
from goods_i g
where g.fname not in(
select fname
from con_i)
group by g.fname
union
```

→ 굿즈만

```
select c.fname,c.pn,sum(c.ci) AS income, 0 as gnum, c.cpc as cnum
from con_i c
where c.fname not in(
select fname
from goods_i)
group by c.fname
order by income desc;
```

→ 콘서트만

SQL: 뷰삭제

```
drop view con_i;
drop view goods_i;
```

03. CASE1: 아이돌별 수익관리

구현

```
state = conn.createStatement();
String sql;
sql = " create view goods_i AS\r\n" +
      " select f.fnum, f.fname, f.pn, sum(g.price) as gi, count(distinct fg.fgnum) as gpc\r\n" +
      " from fan f, idol i, goods g, fangoods fg \r\n" +
      " where i.iname = \"" + idolName + "\" and g.inum = i.inum and g.gnum = fg.gnum and f.fnum = fg.fnum \r\n" +
      " group by f.fnum\r\n" +
      " having count(*)>=1\r\n" +
      " order by sum(g.price);";
```

```
state.executeUpdate(sql);
```

```
sql =
      " create view con_i AS\r\n" +
      " select f.fnum, f.fname, f.pn, sum(fc.income) as ci, count(distinct fc.fcnum) as cpc\r\n" +
      " from fan f, idol i, concert c, concertdate cd, fanconcert fc \r\n" +
      " where i.iname = \"" + idolName + "\" and c.inum = i.inum and c.cdnum = fc.cdnum \r\n" +
      " group by f.fnum\r\n" +
      " having count(*)>=1\r\n" +
      " order by sum(fc.income);";
```

```
state.executeUpdate(sql);
```

```
        order by income desc;\r\n" ;
ResultSet rs = state.executeQuery(sql);
while(rs.next()){
    String fname = rs.getString("fname");
    String fpn = rs.getString("pn");
    String ci = rs.getString("income");
    String gn = rs.getString("gnum");
    String cn = rs.getString("cnum");
    System.out.println(fname + " " + fpn + " " + ci + " " + gn + " " + cn);
    model.addRow(new Object[] {fname,fpn,ci,gn,cn});
}
```

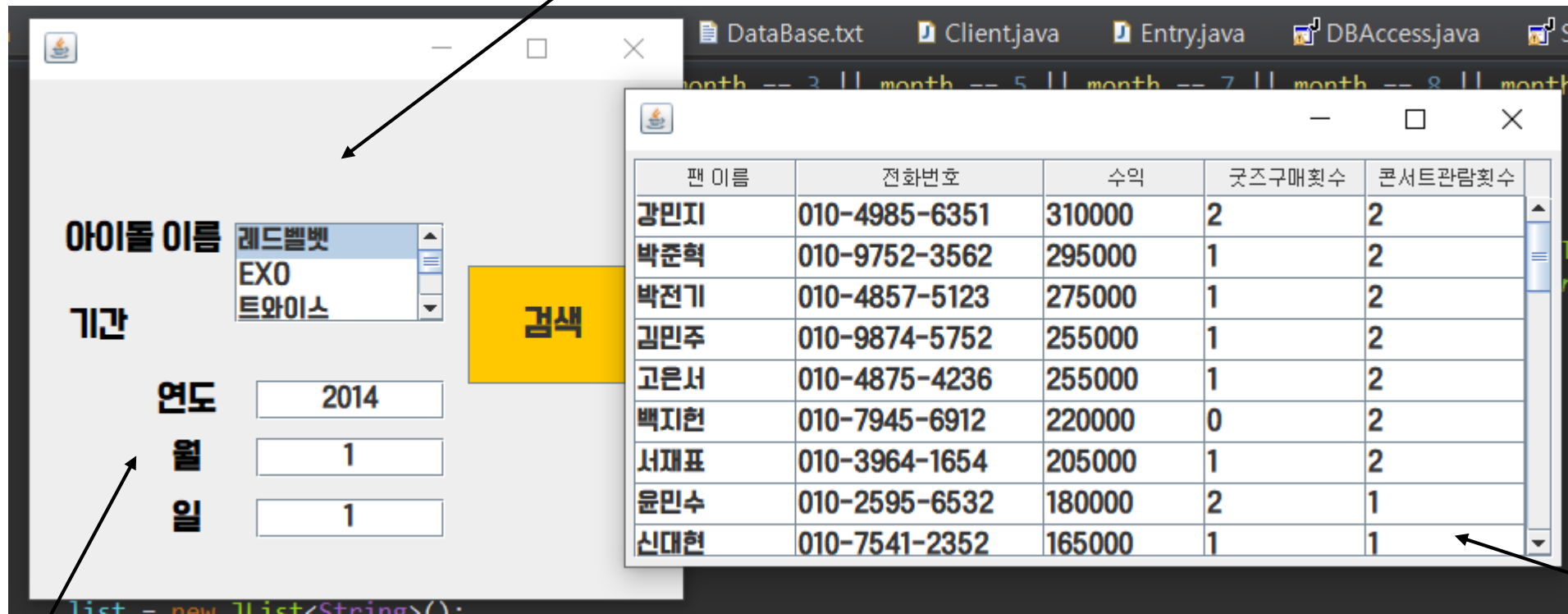
```
sql = " drop view con_i;\r\n";
state.executeUpdate(sql);
sql = " drop view goods_i;";
state.executeUpdate(sql);
```



03. CASE1: 아이돌별 수익관리

구현

sql을 통해 DB에서 아이돌 목록을 가져와 JList로 출력



수익 검색을 시작할 날짜 선택
(프로그램 단계에서 날짜의 유효성 검사)

검색 버튼을 누르면 나오는 결과화면
수익이 많은 순서로 정렬
JTable 활용

04. CASE2: 팬 불참명단 관리(방청 미참여)

팬 불참명단 관리 개요+ SQL

사용자의 입력: 아이돌 이름, 불참횟수

프로그램 내부 처리(Query): 1. 지정한 SQL에 아이돌 이름과 불참 횟수를 넣는다.
2. 검색을 실행하고 정보를 출력한다.

프로그램 출력: 팬 이름, 전화번호, 불참횟수

SQL

```
select f.fname, f.pn, count(*)AS "불참횟수"
from fan f, idol i, idolfan idf, program p, appearance a, fanprogram fp
where i.iname = "WINNER" and f.fnum = idf.fnum and i.inum = idf.inum and fp.fnum = f.fnum and p.pnum = fp.pnum and p.pnum = a.pnum and a.inum = i.inum
group by f.fnum
having count(*)>=1
order by count(*) desc;
```

[] : 사용자 입력부분

04. CASE2: 팬 불참명단 관리(방청 미참여)

구현

```
try {
    state = conn.createStatement();
    String sql;
    sql = " select f.fname, f.pn, count(*)AS \"불참횟수\"\\r\\n\" +
        \"from fan f, idol i, idolfan idf, program p, appearance a, fanprogram fp\\r\\n\" +
        \"where i.iname = '\" + idolName + \"'\" and f.fnum = idf.fnum and i.inum = idf.inum and fp.fnum = f.fnum and p.pnum = fp.pnum and p.pnum = a.pnum and\" +
        \"group by f.fnum\\r\\n\" +
        \"having count(*)>=\" + nscount + \"\\r\\n\" +
        \"order by count(*) desc\";
    ResultSet rs = state.executeQuery(sql);
    while(rs.next()){
        String fname = rs.getString("f.fname");
        String fpn = rs.getString("f.pn");
        String nsc = rs.getString("불참횟수");
        System.out.println(fname + " " + fpn + " " + nsc);
        model.addRow(new Object[] {fname,fpn,nsc});
    }
}
```

결과를 한 레코드씩 받아서 JTable에 추가

04. CASE2: 팬 불참명단 관리(방청 미참여)

구현

DB에서 아이돌 목록을 가져옴

불참횟수의 내림차순으로 정렬

팬 이름	전화번호	불참 횟수
강민지	010-4985-6351	3
김승권	010-7124-6932	3
강혜정	010-5362-7453	3
박규리	010-7954-6921	3
고은서	010-4875-4236	3
나홍민	010-7951-4235	3
곽대경	010-6479-7823	3
김혜연	010-7951-4426	3
김민우	010-7512-3546	3

05. CASE3: 팬과 아이돌 데이터 조회/추가

팬, 아이돌 데이터 조회

사용자의 입력: X

프로그램 내부 처리(Query): 1. 팬, 아이돌 테이블의 모든 항목을 조회한다.

2. 팬은 이름 순서로, 아이돌은 데뷔일 순서로 오름차순 정렬한다.

프로그램 출력: 해당 테이블의 모든 정보

```
select fname,pn,sex,bd,email  
from fan  
order by fname;
```

〈 팬 조회〉

```
select i.iname, e.ename, i.itype, i.debutdate  
from idol i, entertainment e  
where i.eno = e.eno  
order by debutdate;
```

〈 아이돌 조회〉

05. CASE3: 팬과 아이돌 데이터 조회/추가

팬, 아이돌 데이터 조회

```
try {
    state = conn.createStatement();
    String sql;
    sql = "select fname,pn,sex,bd,email\r\n" +
        "from fan\r\n" +
        "order by fname";

    ResultSet rs = state.executeQuery(sql);
    while(rs.next()){
        String fname = rs.getString("fname");
        String fpn = rs.getString("pn");
        String sex = rs.getString("sex");
        String bd = rs.getString("bd");
        String email = rs.getString("email");
        if(sex.equals("m")) sex = "남";
        else sex = "여";
        model.addRow(new Object[] {fname,fpn,sex,bd,email});
    }
} catch (Exception e) {
```

아이돌별수익	불참명단관리
아이돌별수익	불참명단관리
팬 조회	아이돌 조회
팬 추가	

아이돌 이름	전화번호	성별	생년월일	이메일
광민지	010-4985-6351	여	1994-10-24	minzko@hanmail.net
광지현	010-2353-6346	여	1992-04-10	ziwon@hanmail.net
광지환	010-4445-7531	남	1986-06-30	ziwhan@naver.com
광혜정	010-5362-7453	여	1996-01-03	zeong@google.com
고동민	010-8271-3042	남	1996-05-12	dongmin@naver.com
고형진	010-4562-4232	남	1994-04-21	wonjin@google.com
고은서	010-4875-4236	여	1998-02-10	eunseo@hanmail.net
곽대권	010-6479-7823	남	1999-01-20	daegeum@naver.com
곽수진	010-7413-5479	여	2001-04-20	suin@google.com

< 팬 조회 >

```
try {
    state = conn.createStatement();
    String sql;
    sql = "select i.iname, e.ename, i.itype, i.debutdate\r\n" +
        "from idol i, entertainment e\r\n" +
        "where i.eno = e.eno\r\n" +
        "order by debutdate";

    ResultSet rs = state.executeQuery(sql);
    while(rs.next()){
        String iname = rs.getString("i.iname");
        String ename = rs.getString("e.ename");
        String itype = rs.getString("i.itype");
        String debutdate = rs.getString("i.debutdate");
        if(itype.equals("b")) itype = "보이그룹";
        else itype = "걸그룹";
        model.addRow(new Object[] {iname,ename,itype,debutdate});
    }
} catch (Exception e) {
```

아이돌별수익	불참명단관리
아이돌별수익	불참명단관리
팬 조회	아이돌 조회
팬 추가	

아이돌 이름	소속사	유형	데뷔일
EXO	SM엔터테인먼트	보이그룹	2012-04-08
GOT7	JYP엔터테인먼트	보이그룹	2014-01-16
레드벨벳	SM엔터테인먼트	걸그룹	2014-08-01
WINNER	YG엔터테인먼트	보이그룹	2014-08-17
트와이스	JYP엔터테인먼트	걸그룹	2015-10-20
BLACKPINK	YG엔터테인먼트	걸그룹	2016-08-08

< 아이돌 조회 >

05. CASE3: 팬과 아이돌 데이터 조회/추가

팬, 아이돌 데이터 추가

사용자의 입력: 팬,아이돌의 기본 정보

프로그램 내부 처리(Query): 1. 기존에 존재하는 팬이나 아이돌의 합계를 SQL로 읽어와 새로운 레코드에 넣을 기본키값(정수)을 배정한다.

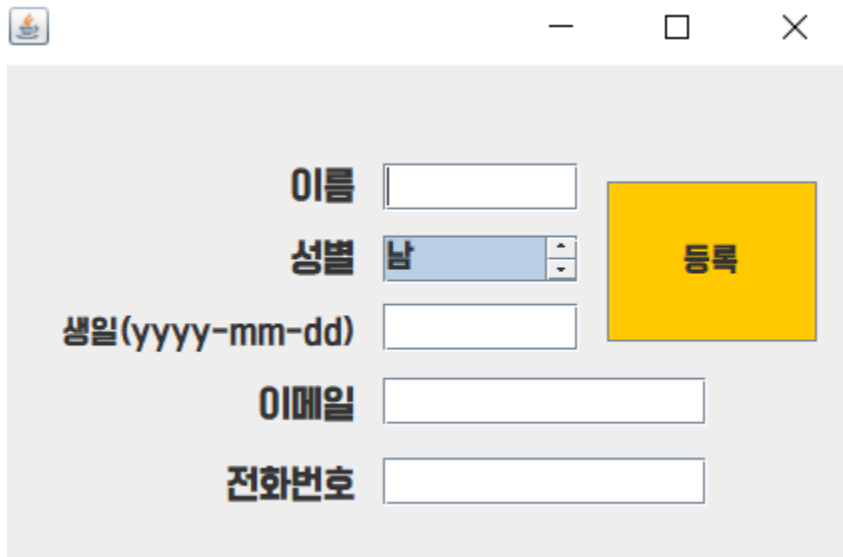
2. 아이돌 추가의 경우 소속사의 정보를 SQL로 읽어와 화면에 출력한다.

3. 입력한 정보를 바탕으로 insert into를 통해 DB에 저장한다.

프로그램 출력: X

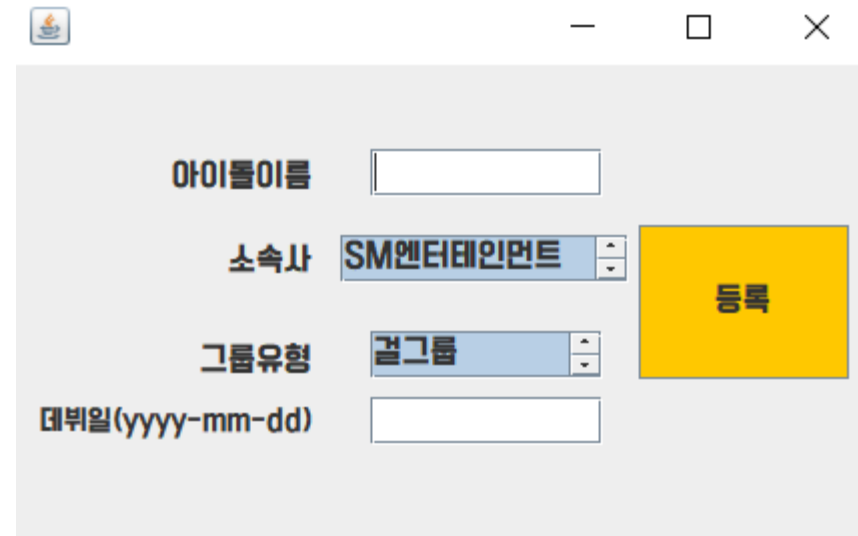
05. CASE3: 팬과 아이돌 데이터 조회/추가

팬, 아이돌 데이터 추가



A screenshot of a web application window titled "팬 추가" (Add Fan). The window has a standard Windows-style title bar with minimize, maximize, and close buttons. The form contains the following fields: "이름" (Name) with a text input, "성별" (Gender) with a dropdown menu showing "남" (Male), "생일(yyyy-mm-dd)" (Birthday) with a text input, "이메일" (Email) with a text input, and "전화번호" (Phone Number) with a text input. A large yellow button labeled "등록" (Register) is positioned to the right of the form fields.

< 팬 추가 >



A screenshot of a web application window titled "아이돌 추가" (Add Idol). The window has a standard Windows-style title bar with minimize, maximize, and close buttons. The form contains the following fields: "아이돌이름" (Idol Name) with a text input, "소속사" (Agency) with a dropdown menu showing "SM엔터테인먼트" (SM Entertainment), "그룹유형" (Group Type) with a dropdown menu showing "걸그룹" (Girl Group), and "데뷔일(yyyy-mm-dd)" (Debut Date) with a text input. A large yellow button labeled "등록" (Register) is positioned to the right of the form fields.

< 아이돌 추가 >

시연