< Baby-Monitor project; Backend part code sample>

```
from flask import Flask, request, jsonify
from flask_sqlalchemy import SQLAlchemy
from flask_marshmallow import Marshmallow
import os
import firebase_admin
from firebase_admin import credentials
# Send to single device.
from pyfcm import FCMNotification

#initialization.. not that important
app = Flask(__name__)
basedir = os.path.abspath(os.path.dirname(__file__))
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///' + os.path.join(basedir, 'crud.sqlite')
db = SQLAlchemy(app)
ma = Marshmallow(app)

cred = credentials.Certificate("/home/babymonitor/mysite/service-account.json")
firebase_admin.initialize_app(cred)

#this is our "model". this is basically an object class and our SQL DB is made of these objects,
each row is = to one object but in our project we are only using
#one model. (1 row)
class Info(db.Model):
    # more initilzation
    id = db.Column(db.Integer, primary_key = True)
    camId = db.Column(db.Integer, unique = False)
    gender = db.Column(db.String(20), unique = False)
    age = db.Column(db.Integer, unique = False)
    isSafe = db.Column(db.Boolean, unique = False)
    isMonitoring = db.Column(db.Boolean, unique = False)

    globalOne = db.Column(db.String(40), unique = False)
    globalTwo = db.Column(db.String(40), unique = False)
    globalThree = db.Column(db.String(40), unique = False)

    fcmKey = db.Column(db.String(360), unique = False)

    #this is our constructor
    def __init__(self, camId, gender, age, isSafe, isMonitoring):
        self.camId = camId
        self.gender = gender
        self.age = age
        self.isSafe = isSafe
        self.isMonitoring = isMonitoring
        self.globalOne = "Terrace"
```

```
        self.globalTwo = "Kitchen"
        self.globalThree = "Bedroom"
        self.fcmKey = ""


#ignore this
class InfoSchema(ma.Schema):
    class Meta:
        fields = ('camId', 'gender', 'age', 'isSafe', 'isMonitoring', 'globalOne', 'globalTwo',
'globalThree', 'fcmKey')

#when we query our SQL DB, we query the info_schema
info_schema = InfoSchema()
info_schemas = InfoSchema(many=True)

#this is our FCM (Firebase cloud messaging) function
def GeneratePushNotification():
    #we get our object data
    cam = Info.query.get(1)
    # if we are currently monitoring, we send the push noti
    if cam.isMonitoring == True:
        #this is our api key
        push_service =
FCMNotification(api_key="AAAAXRfiD6U:APA91bEYkQKyql69dm1EAF6WjtV-
s0VG5HkiJb5AHxXPLzCf-jQoNtwZzsQjH8BvCD9eI7E5whl-
w9uxCCVoY6WzjtIdK7Rfz0zqap0RdilRxTTzKNlf_APcr4Heb_iYje8E7ktjbghx")

        # Your api-key can be gotten from:
https://console.firebase.google.com/project/<project-name>/settings/cloudmessaging
        #just info
        registration_id = cam.fcmKey
        message_title = "Baby Monitor Alert"
        message_body = "Baby is in DANGER!"
        result = push_service.notify_single_device(registration_id=registration_id,
message_title=message_title, message_body=message_body)
        print (result)


@app.route('/')
def base():
    return "This is main url"

#creating our initial data
@app.route('/api/create/cameraData', methods =["POST"])
def create_cameraData():
    #we are retreiving data from the posted json
    content = request.json
```

```
    camId = content["camId"]
    gender = content["gender"]
    age = content["age"]
    isSafe = content["isSafe"]
    isMonitoring = content["isMonitoring"]
    #create a new model (object) using the data we got from the json
    new_data = Info(camId, gender, age, isSafe, isMonitoring)
    #we add the data to our SQL Databse
    db.session.add(new_data)
    #we save the data
    db.session.commit()

    return "success"

#updating safe status
@app.route('/api/update/safe', methods = ["PUT"])
def update_safe_status():
    cam = Info.query.get(1)

    isSafe = request.json['isSafe']

    cam.isSafe = isSafe
    #if the parameter is not safe, we send the push notiication
    if not cam.isSafe:
        GeneratePushNotification()
    #we save the data
    db.session.commit()

    return info_schema.jsonify(cam)


@app.route('/api/update/monitor', methods = ["PUT"])
def update_monitor_status():
    cam = Info.query.get(1)

    isMonitoring = request.json['isMonitoring']

    cam.isMonitoring = isMonitoring

    db.session.commit()

    return info_schema.jsonify(cam)


# endpoint to get user detail by id
@app.route("/api/get/camera/<id>", methods=["GET"])
def camera_detail(id):
```

```
    data = Info.query.get(id)

    return info_schema.jsonify(data)

#returns isSafe status
@app.route("/api/get/status", methods =["GET"])
def get_safe_status():
    cam = Info.query.get(1)

    val = cam.isSafe
    #if not safe, we send push notification
    if not val:
        GeneratePushNotification()

    return jsonify(cam.isSafe)

#returns isMonitoring status
@app.route("/api/get/monitoring", methods =["GET"])
def get_monitor_status():
    cam = Info.query.get(1)

    return jsonify(cam.isMonitoring)

#returns global locations
@app.route("/api/get/globals", methods =["GET"])
def get_globals():
    cam = Info.query.get(1)

    return jsonify(globalOne = cam.globalOne,
                globalTwo = cam.globalTwo,
                globalThree = cam.globalThree)


#returns all the camera data but since we are using one model (object) it should return only
1
@app.route("/api/get/all", methods=["GET"])
def get_all_camera():
    all_camera = Info.query.all()
    result = info_schemas.dump(all_camera)
    return jsonify(result.data)

#update camera by id
@app.route("/api/update/camera/<id>", methods=["PUT"])
def camera_update(id):
    #get a reference to our model in SQL DB
    cam = Info.query.get(id)
    #request data from posted json
```

```
    camId = request.json['camId']
    gender = request.json['gender']
    age = request.json['age']
    isSafe = request.json['isSafe']

    #update our model with new data
    cam.camId = camId

    cam.gender = gender

    cam.age = age

    cam.isSafe = isSafe

    #save the data
    db.session.commit()

    #send push notification if the new status of isSafe is not safe.
    if not isSafe:
        GeneratePushNotification()

    return info_schema.jsonify(cam)

#gets global location of one location (parameter is the global location number)
#(.ie. id =1 == global.location1.)
@app.route("/api/update/globals/<id>", methods=["PUT"])
def global_update(id):
    cam = Info.query.get(1)

    content = request.json
    newLoc = content['global']

    if id == '1':
        cam.globalOne = newLoc

    if id == '2':
        cam.globalTwo = newLoc

    if id == '3':
        cam.globalThree = newLoc

    db.session.commit()

    return jsonify(newLoc)

#updates fcm key on our server to new one
@app.route("/api/update/fcmkey", methods=["PUT"])
```

```
def fcmKey_update():
    cam = Info.query.get(1)

    content = request.json
    newLoc = content['key']

    cam.fcmKey = newLoc

    db.session.commit()

    return "success"


if __name__ == '__main__':
    app.run(debug=True)
```

Database Team Project

Team01 : DBDBDip

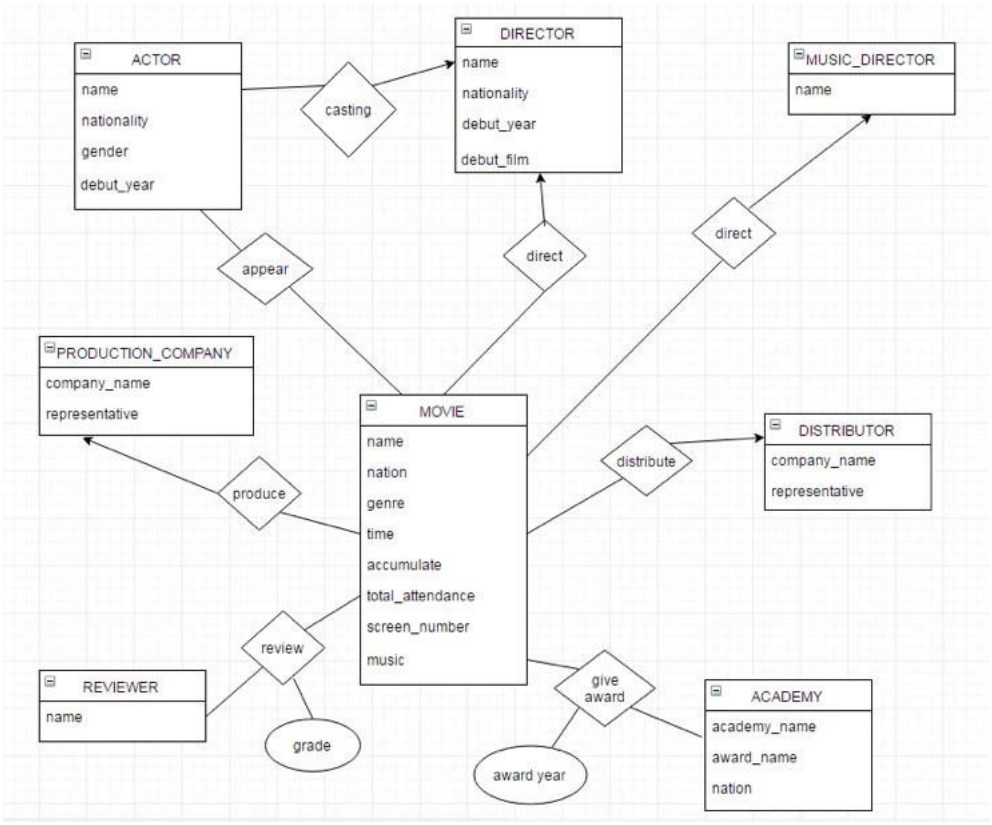
1415014 Seojin Kim

1415020 Chaeyoon Kim

1415047 Kyungmin Lee

2017.06.01

1. ER Diagram

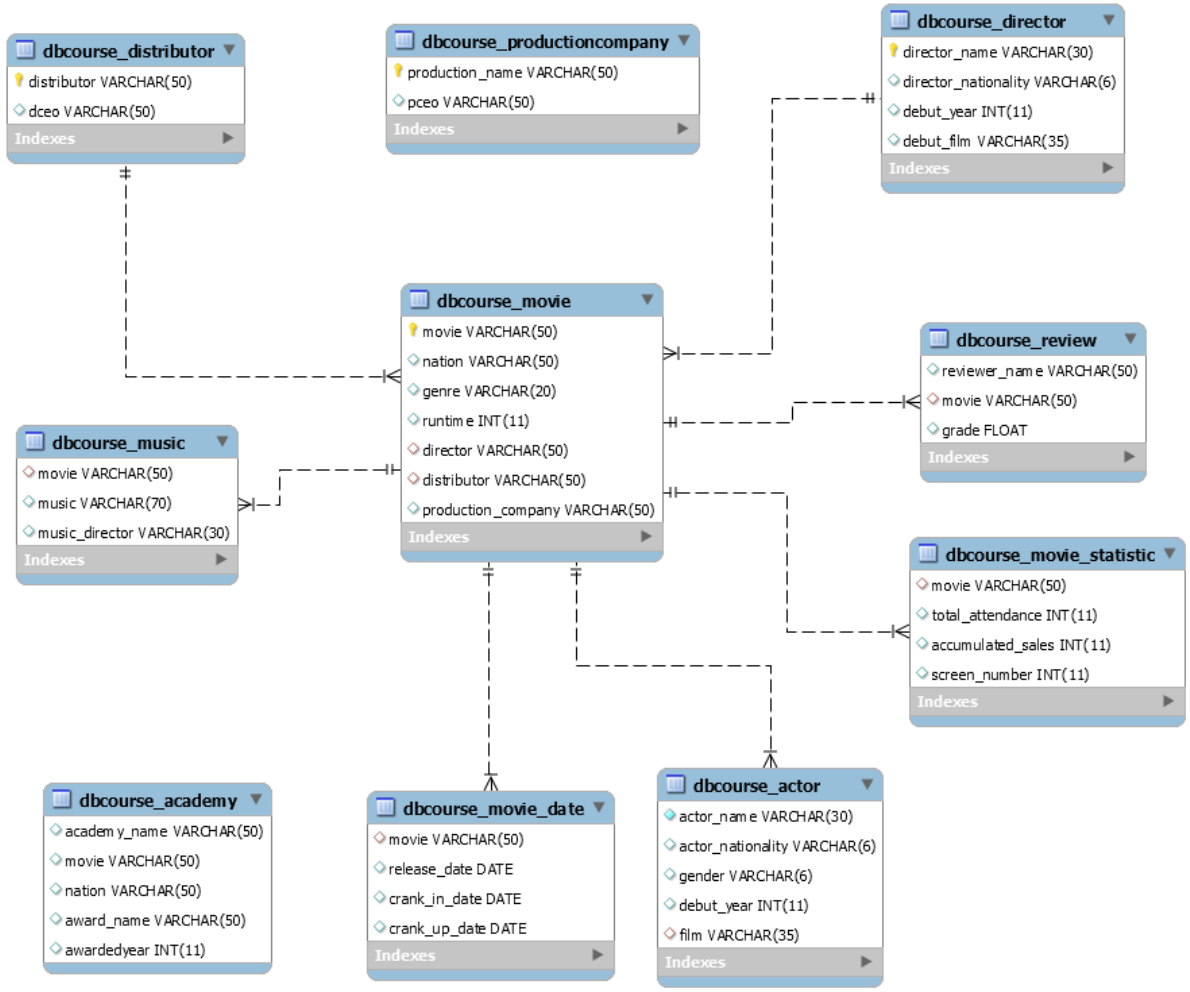


ER 다이어그램은 Entity-Relationship 다이어그램으로 개체와 관계를 통해 모델링을 하는 것을 돕는다. 이 ER다이어그램에는 '배우' '감독' '음악감독' '영화' '제작사' '배급사' '시상식' ' 평론가' 라는 개체들이 있고 이들은 속성을 가지고 서로서로 관계를 가지고 있다.

'배우'들은 각각 여러 '영화'들에 출연하고 '감독'은 '배우'들을 캐스팅한다. 또한 '감독'과 ' 음악감독'은 여러 '영화'를 감독한다. 그리고 '제작사'와 '배급사'는 각각 여러 영화들을 제작하고 배급한다.

'평론가'들은 '영화'들을 평론하고 그에 따라서 grade라는 '평점'이 속성으로 나타나게 되며 '시상식'은 '영화'에 상을 주고 그 사이 영화가 몇년도에 상을 받았는지에 대한 award_year라는 속성이 나타나게 된다.

2. Database schema diagram



이 부분에서 후에 movie table에서 distributor를 외래키로 설정하는 것은 안하기로 수정하였다.

3. Class and method

각 테이블에 대한 insert,update,delete,select문을 행할 수 있는 class를 모두 만들었다. 각 테이블은 3개의 class(GUI,Exam,DAO)로 이루어져 있으며 GUI는 MySql에서 받아온 테이블을 출력하는 창, DAO는 DB와 연결되어

4. Main class name & how to run, connection configuration

Movie_List라는 클래스에

5. 17 requirements

(1) Create 10 tables ,30columns

Create table

```
create table DBCOURSE_DISTRIBUTOR(
 distributor varchar(50),
 dceo   varchar(50),
);
```

(2) 50 records 이상 집어넣었다.

(3) PK, FK, NOT NULL 설정

처음 Database를 create할 때 각 table에 필요한 primary key, foreign key를 입력하고, 동시에 not null인 속성을 입력하였다. foreign key의 경우 그 값을 수정 및 삭제 할 때 자유로울 수 있도록 update/delete cascade를 적용하였다.

```
create table DBCOURSE_MOVIE(
movie varchar(50) not null,
.....
primary key(movie),
.....
foreign key (director) references DBCOURSE_DIRECTOR (director_name) on update cascade on delete cascade,
);
```

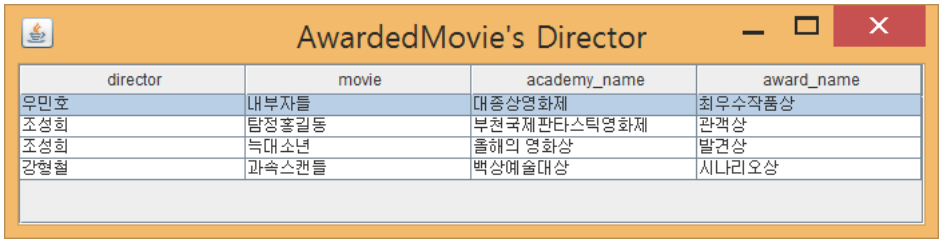
(4) Create 함수를 이용하여 view를 생성하였다.

우리팀이 만든 첫 번째 view는 DBCOURSE_MOVIETIME으로 영화테이블에서 상영시간과 영화 이름만 뽑아서 볼 수 있는 테이블이고, 두 번째는 DBCOURSE_AWARD로 상을 탄 영화 작품의 감독, 영화 이름, 영화제, 상 이름, 상을 탄 해를 보여준다.

```
94  ###create view###
95  create view DBCOURSE_MOVIETIME as select movie, runtime from DBCOURSE_MOVIE;
96  create view DBCOURSE_AWARD as select M.movie, director, academy_name, award_name, awardedyear
97  from  dbcourse_movie M, dbcourse_academy A
98  where M.movie = A.movie;
```

(5) view를 이용한 query : select *로 각 뷰를 보여주었다.

movie	runtime
Interstellar	169
Prometheus	123
The legend of 1900	123
Titanic	195
과속스캔들	108
꽃미남 연쇄 테러사건	81
내부자들	130
늑대소년	125
센과 치히로의 행방불명	126
탐정홍길동	122



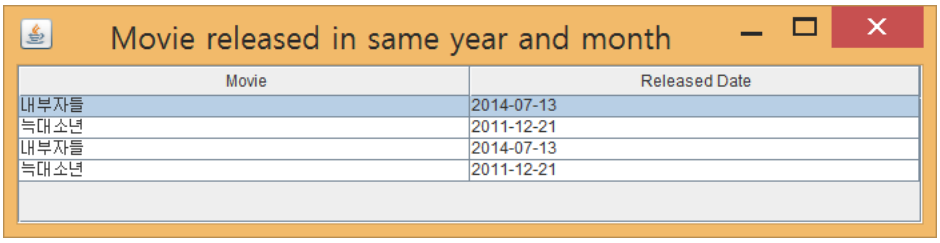
director	movie	academy_name	award_name
우민호	내부자들	대종상영화제	최우수작품상
조성희	탐정홍길동	부천국제판타스틱영화제	관객상
조성희	늑대소년	올해의 영화상	발견상
강형철	과속스캔들	백상예술대상	시나리오상

(6) Alter 함수를 이용하여 table의 속성을 변화, index를 생성하였다.

```
84 • alter table DBCOURSE_MOVIE add index index_genre(genre);
85
86 • alter table DBCOURSE_REVIEW add index index_grade(grade);
87
88 • alter table DBCOURSE_ACADEMY add index index_awardedyear(awardedyear);
89
90 • alter table DBCOURSE_MOVIE_DATE add index index_release_date(release_date);
91
92 • alter table DBCOURSE_MOVIE_STATISTIC add index index_total_attendance(total_attendance);
```

(7) Index를 사용하는 query를 포함한다.

```
select distinct a.movie, date_format(a.crank_in_date,'%Y-%m-%d')
as crank_in_date, date_format(b.crank_up_date, '%Y-%m-%d') as crank_up_date
from dbcourse_movie_date a, dbcourse_movie_date b
where (a.crank_in_date between '2011-12-01' and '2013-06-01')
and (b.crank_up_date between '2013-06-01' and '2014-12-01');
```



Movie	Released Date
내부자들	2014-07-13
늑대소년	2011-12-21
내부자들	2014-07-13
늑대소년	2011-12-21

(8) Transactions

우리 팀이 만든 database에서 트랜잭션을 적용할 수 있는 부분은 movie table에 새로운 record를 insert하려 할 때 director부분이 director 테이블과 외래키로 연결이 되어있고 director_name은 director table에서 primary key임으로 table에 없는 값을 넣으려 하면

insert/update가 이루어 지지 않는다. 따라서 director가 먼저 존재하는 record인지 확인하고 없으면 그 값을 먼저 update한 뒤에 movie table에 insert/update 할 수 있다. 하지만 구현은 하지 못하였다.

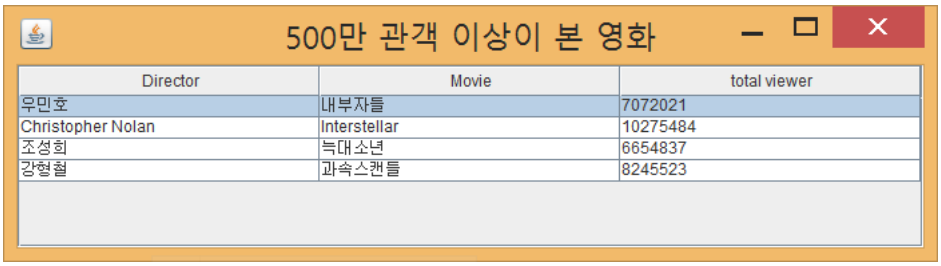
(9) Nested queries

DBCOURSE_AWARD view를 생성할 때 nested query를 이용하였다.

```
96 • create view DBCOURSE_AWARD as select M.movie, director, academy_name, award_name, awardedyear
97   from  dbcourse_movie M, dbcourse_academy A
98   where M.movie = A.movie;
```

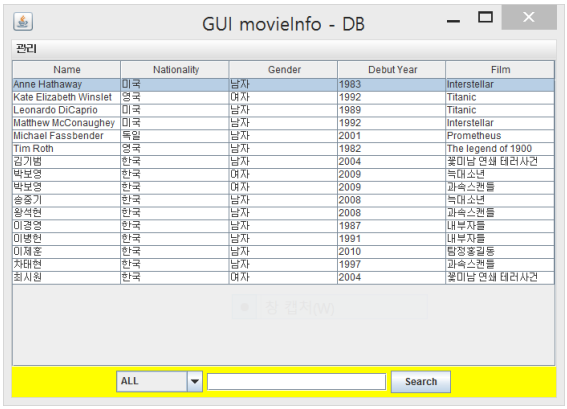
(10) Join Queries

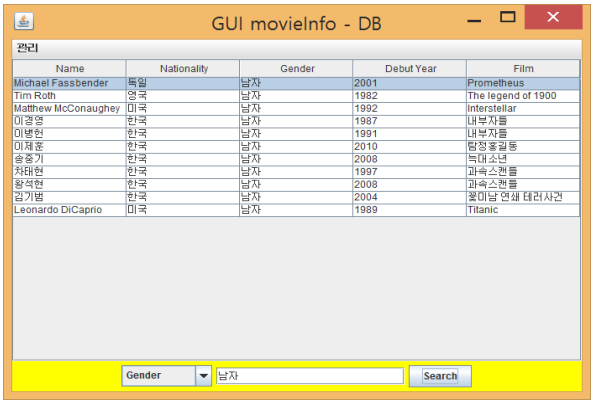
```
select director,M.movie, total_attendance as viewer
from DBCOURSE_MOVIE M ,DBCOURSE_MOVIE_STATISTIC S
where   M.movie = S.movie and total_attendance > 5000000;
```



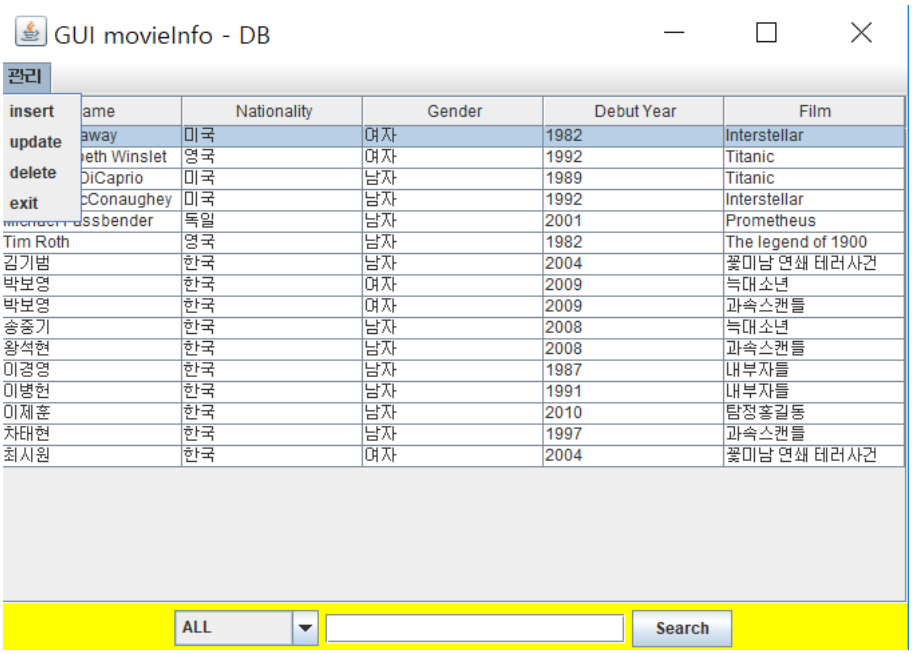
(11) Queries that are parameterized and dynamically created

각 테이블에서 특정 attribute에 대한 select 검색이 가능하며, update할 때 도 dyanamic하게 입력하여 update할 수 있다.





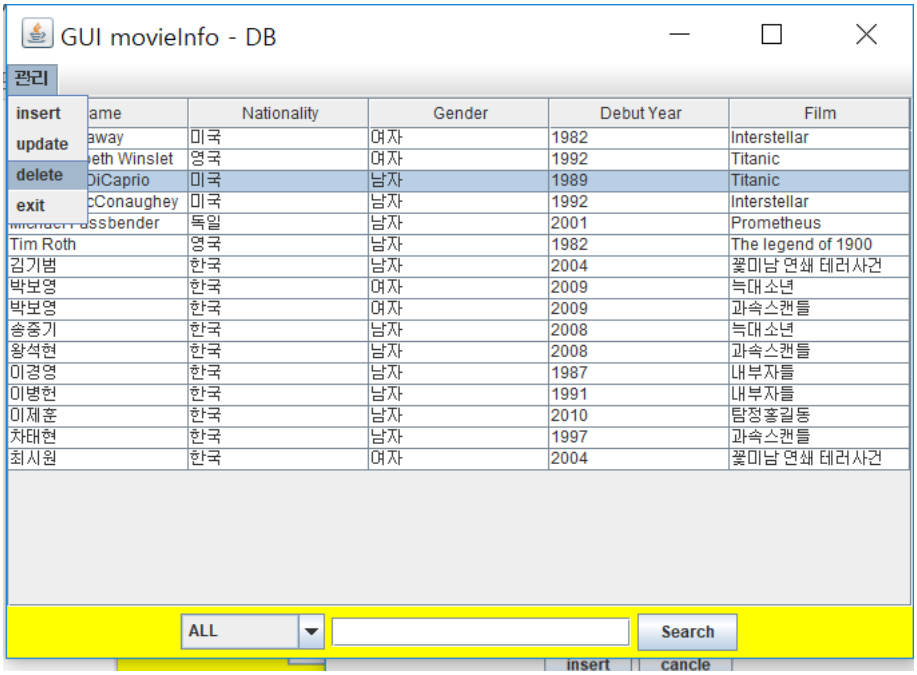
(12) user interface which is graphical or text-based



(13) insert

(14) update

(15) delete



(16) select

6. Team responsibility assignment // 누가 분담 했는지

Member name	SQL	Java code	Report	Demo video
Seojin Kim	-Insert Foreign key -Drop queries -Database Schema Diagram	-Review GUI -Academy GUI -Movie_Date GUI -Music GUI -Movie_Statistic GUI	-1, 3 -ppt making	-setup environment -record demo video
Chaeyoon Kim	-Create table -Insert records -ER Diagram	- Create main class - Director GUI -Distributor GUI -ProductionCompany GUI -Actor GUI	-2 -ppt making	
Kyungmin Lee	-Create table SQL -Create view -Create Transaction Query	-Movie GUI -main class -connect DB and Java -insert, update, delete code	-4-8, code capture - presentation	-record demo video

7. SQL scripts

(1) Create Database

```sql
create table DBCOURSE_DISTRIBUTOR(
 distributor varchar(50),
 dceo  varchar(50),
primary key(distributor)
);

CREATE TABLE DBCOURSE_DIRECTOR(
  director_name  varchar(30),
  director_nationality varchar(6),
  debut_year  int,
  debut_film  varchar(35),
  primary key(director_name) );


create table DBCOURSE_MOVIE(
   movie varchar(50) not null,
   nation varchar(50),
   genre varchar(20),
   runtime int,
   director varchar(50),
   distributor varchar(50),
   production_company varchar(50),
   primary key(movie),
   foreign key (director) references DBCOURSE_DIRECTOR (director_name) on update cascade on delete cascade
);

CREATE TABLE DBCOURSE_ACTOR(
  actor_name  varchar(30) not null,
  actor_nationality varchar(6),
  gender   varchar(6),
  debut_year  int,
  film varchar(35),
  foreign key (film) references DBCOURSE_MOVIE (movie) on update cascade on delete cascade );

```

```sql
create table DBCOURSE_ACADEMY(
 movie varchar(50),
 academy_name varchar(50),
 nation varchar(50),
 award_name varchar(50),
 awardedyear int

);

create table DBCOURSE_MUSIC(
 movie varchar(50),
 music varchar(70),
 music_director varchar(30),
 foreign key (movie) references DBCOURSE_MOVIE (movie) on update cascade on delete cascade
);

create table DBCOURSE_PRODUCTIONCOMPANY(
 production_name varchar(50),
 pceo varchar(50),
 primary key(production_name)
);

create table DBCOURSE_MOVIE_STATISTIC(
  movie varchar(50),
  total_attendance int,
  accumulated_sales int,
  screen_number int,
  foreign key (movie) references DBCOURSE_MOVIE (movie) on update cascade on delete cascade
);
```

```
65  create table DBCOURSE_REVIEW(
66      reviewer_name varchar(50),
67      movie varchar(50),
68      grade float,
69      foreign key (movie) references DBCOURSE_MOVIE (movie) on update cascade on delete cascade
70  );
71
72  create table DBCOURSE_MOVIE_DATE(
73      movie varchar(50),
74      release_date date,
75      crank_in_date date,
76      crank_up_date date,
77      foreign key (movie) references DBCOURSE_MOVIE (movie) on update cascade on delete cascade
78
79  );
```

(2) Drop Tables

```
use team01; # 김서진
# cannot delete or update a parent row, child 부터 해야한다.
# drop table 의 순서는 foreign key 와 관련이 있다
# 그래서 drop table query 는 foreign key 를 가장 많이 가지고 있고, foreign key 에 가장 많이 해당하는 MOVIE table 을 기점으로 하여
# 1. MOVIE 와 연관이 없는 테이블(ACADEMY, PRODUCTION_COMPANY),
# 2. MOVIE 의 child 에 해당하는 테이블들(ACTOR, MOVIE_DATE, MOVIE_STATISTIC, MUSIC, REVIEW)
# 3. MOVIE 본인
# 4. MOVIE 의 parent 에 해당하는 테이블들(DIRECTOR, DISTRIBUTOR)의 순서로 SQL 을 작성하였다.
drop table DBCOURSE_ACADEMY;

drop table DBCOURSE_PRODUCTIONCOMPANY;

drop table DBCOURSE_ACTOR;

drop table DBCOURSE_MOVIE_DATE;

drop table DBCOURSE_MOVIE_STATISTIC;

drop table DBCOURSE_MUSIC;

drop table DBCOURSE_REVIEW;

drop table DBCOURSE_MOVIE;

drop table DBCOURSE_DIRECTOR;

drop table DBCOURSE_DISTRIBUTOR;
```

(3) Insert

```
insert into DBCOURSE_DIRECTOR values('Christopher Nolan','영국',1998,'미행');
insert into DBCOURSE_DIRECTOR values('Ridley Scott','영국',1962,'Z Cars');
insert into DBCOURSE_DIRECTOR values('Miyajaki Hayao','일본',1978,'미래 소년 코난');
insert into DBCOURSE_DIRECTOR values('우민호','한국',2000,'누가 예수를 죽였는가?');
insert into DBCOURSE_DIRECTOR values('Giuseppe Tornatore','이탈리아',1985,'Le minoranze etniche in Sicilia');
insert into DBCOURSE_DIRECTOR values('조성희','한국',2009,'남매의집');
insert into DBCOURSE_DIRECTOR values('강형철','한국',2008,'과속스캔들');
insert into DBCOURSE_DIRECTOR values('이권','한국',2002,'겁쟁이들이 더 흉폭하다');
insert into DBCOURSE_DIRECTOR values('James Cameron','캐나다',1981,'피라나2');

insert into DBCOURSE_DISTRIBUTOR  values('showbox','유정훈');
insert into DBCOURSE_DISTRIBUTOR  values('foxkorea','오성호');
insert into DBCOURSE_DISTRIBUTOR  values('cj엔터테인먼트','김성수');
insert into DBCOURSE_DISTRIBUTOR  values('롯데엔터테인먼트','차원천');
insert into DBCOURSE_DISTRIBUTOR  values('청어람','최윤배');
```

```
insert into DBCOURSE_ACADEMY values('내부자들','대종상영화제','한국','최우수작품상',2016);
insert into DBCOURSE_ACADEMY values('Interstellar','올해의 영화상','한국','외국어영화상',2015);
insert into DBCOURSE_ACADEMY values('센과 치히로의 행방불명','아카데미 시상식','미국','장편애니메이션작품상',2003);
insert into DBCOURSE_ACADEMY values('the legend of 1900','골든글로브 시상식','미국','음악상',2000);
insert into DBCOURSE_ACADEMY values('탐정홍길동','부천국제판타스틱영화제','한국','관객상',2016);
insert into DBCOURSE_ACADEMY values('늑대소년','올해의 영화상','한국','발견상',2013);
insert into DBCOURSE_ACADEMY values('과속스캔들','백상예술대상','한국','시나리오상',2009);
insert into DBCOURSE_ACADEMY values('Titanic','골든글로브 시상식','미국','작품상감독상',1998);



insert into DBCOURSE_MOVIE values('내부자들','한국','drama',130,'우민호','showbox','내부자들');
insert into DBCOURSE_MOVIE values('The legend of 1900','이탈리아','music',123,'Giuseppe Tornatore','medusa Distribuzione','Medusa P
insert into DBCOURSE_MOVIE values('Interstellar','미국','SF',169,'Christopher Nolan','warner brothers korea','paramount Pictures')
insert into DBCOURSE_MOVIE values('센과 치히로의 행방불명','일본','fantasy',126,'Miyajaki Hayao','doho','Dentsu Inc.');
insert into DBCOURSE_MOVIE values('Prometheus','미국','thriller',123,'Ridley Scott','foxkorea','Brandywine');
insert into DBCOURSE_MOVIE values('탐정홍길동','한국','drama',122,'조성희','cj엔터테인먼트','영화사비단길');
insert into DBCOURSE_MOVIE values('늑대소년','한국','romance',125,'조성희','cj엔터테인먼트','영화사비단길');
insert into DBCOURSE_MOVIE values('과속스캔들','한국','comedy',108,'강형철','롯데엔터테인먼트','디씨지플러스');
insert into DBCOURSE_MOVIE values('꽃미남 연쇄 테러사건','한국','mystery',81,'이권','청어람','에스엠픽쳐스');
insert into DBCOURSE_MOVIE values('Titanic','미국','drama',195,'James Cameron','foxkorea','20세기 폭스 필름');
```

```
insert into DBCOURSE_ACTOR values('Michael Fassbender','독일','남자',2001,'Prometheus');
insert into DBCOURSE_ACTOR values('Tim Roth','영국','남자',1982,'The legend of 1900');
insert into DBCOURSE_ACTOR values('Matthew McConaughey','미국','남자',1992,'Interstellar');
insert into DBCOURSE_ACTOR values('Anne Hathaway','미국','여자',1982,'Interstellar');
insert into DBCOURSE_ACTOR values('이경영','한국','남자',1987,'내부자들');
insert into DBCOURSE_ACTOR values('이병헌','한국','남자',1991,'내부자들');
insert into DBCOURSE_ACTOR values('이제훈','한국','남자',2010,'탐정홍길동');
insert into DBCOURSE_ACTOR values('박보영','한국','여자',2009,'늑대소년');
insert into DBCOURSE_ACTOR values('송중기','한국','남자',2008,'늑대소년');
insert into DBCOURSE_ACTOR values('차태현','한국','남자',1997,'과속스캔들');
insert into DBCOURSE_ACTOR values('왕석현','한국','남자',2008,'과속스캔들');
insert into DBCOURSE_ACTOR values('박보영','한국','여자',2009,'과속스캔들');
insert into DBCOURSE_ACTOR values('김기범','한국','남자',2004,'꽃미남 연쇄 테러사건');
insert into DBCOURSE_ACTOR values('최시원','한국','여자',2004,'꽃미남 연쇄 테러사건');
insert into DBCOURSE_ACTOR values('Leonardo DiCaprio','미국','남자',1989,'Titanic');
insert into DBCOURSE_ACTOR values('Kate Elizabeth Winslet','영국','여자',1992,'Titanic');
```

```
insert into DBCOURSE_MUSIC values('The legend of 1900','Playing love','Ennio Morriconne');
insert into DBCOURSE_MUSIC values('내부자들','우검사','조영욱');
insert into DBCOURSE_MUSIC values('Interstellar','STAY','Hans Zimmer');
insert into DBCOURSE_MUSIC values('센과 치히로의 행방불명','언제나 몇 번이라도','Hisaishi Joe');
insert into DBCOURSE_MUSIC values('탐정홍길동',null,'김태성');
insert into DBCOURSE_MUSIC values('늑대소년','my prince','심현정');
insert into DBCOURSE_MUSIC values('과속스캔들','아마도그건','김준석');
insert into DBCOURSE_MUSIC values('꽃미남 연쇄 테러사건','wonder boy','최동훈');
insert into DBCOURSE_MUSIC values('Titanic','my heart will go on','James Horner');
```

```
insert into DBCOURSE_MOVIE_DATE values('내부자들','2015-11-19','2014-07-13','2014-11-07');
insert into DBCOURSE_MOVIE_DATE values('Titanic','1998-02-20',null,null);
insert into DBCOURSE_MOVIE_DATE values('Interstellar','2014-11-06',null,null);
insert into DBCOURSE_MOVIE_DATE values('센과 치히로의 행방불명','2002-06-27',null,null);
insert into DBCOURSE_MOVIE_DATE values('Prometheus','2012-06-06',null,null);
insert into DBCOURSE_MOVIE_DATE values('탐정홍길동','2016-05-04','2014-12-02','2015-04-16');
insert into DBCOURSE_MOVIE_DATE values('늑대소년','2012-10-31','2011-12-21','2012-04-14');
insert into DBCOURSE_MOVIE_DATE values('과속스캔들','2008-12-03','2008-07-20','2008-09-30');
insert into DBCOURSE_MOVIE_DATE values('꽃미남 연쇄 테러사건','2007-07-26',null,null);
```

```
insert into DBCOURSE_PRODUCTIONCOMPANY values('내부자들','김원국');
insert into DBCOURSE_PRODUCTIONCOMPANY values('영화사비단길','김수진');
insert into DBCOURSE_PRODUCTIONCOMPANY values('디씨지플러스','박현태');
insert into DBCOURSE_PRODUCTIONCOMPANY values('에스엠픽쳐스','김상은');
insert into DBCOURSE_PRODUCTIONCOMPANY values('20세기 폭스 필름','윌리엄 폭스');

insert into DBCOURSE_MOVIE_STATISTIC values('내부자들',7072021,5659813,1075);
insert into DBCOURSE_MOVIE_STATISTIC values('The legend of 1900',25,8,1);
insert into DBCOURSE_MOVIE_STATISTIC values('Interstellar',10275484,8229109,1342);
insert into DBCOURSE_MOVIE_STATISTIC values('센과 치히로의 행방불명',159755,125741,339);
insert into DBCOURSE_MOVIE_STATISTIC values('Prometheus',971482,860015,521);
insert into DBCOURSE_MOVIE_STATISTIC values('함정홍길동',1430666,1151439,764);
insert into DBCOURSE_MOVIE_STATISTIC values('늑대소년',6654837,4659311,706);
insert into DBCOURSE_MOVIE_STATISTIC values('과속스캔들',8245523,5394010,405);
insert into DBCOURSE_MOVIE_STATISTIC values('꽃미남 연쇄 테러사건',98284,59428,107);
insert into DBCOURSE_MOVIE_STATISTIC values('Titanic',369894,447414,272);

insert into DBCOURSE_REVIEW values('김형석','내부자들',7.5);
insert into DBCOURSE_REVIEW values('이은선','Interstellar',9.5);
insert into DBCOURSE_REVIEW values('김도훈','Prometheus',9);
insert into DBCOURSE_REVIEW values('이지혜','함정홍길동',8);
insert into DBCOURSE_REVIEW values('백은하','늑대소년',6.75);
insert into DBCOURSE_REVIEW values('허지웅','과속스캔들',7.5);
insert into DBCOURSE_REVIEW values('이동진','꽃미남 연쇄 테러사건',6);
insert into DBCOURSE_REVIEW values('이용철','Titanic',10);
```

8. JAVA CODE

(1) Movie_List :: Main 함수가 들어있고 각 테이블을 볼 수 있는 버튼이 있다.

```
import java.awt.BorderLayout;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import javax.swing.table.DefaultTableModel;

public class Movie_List extends JFrame implements MouseListener,ActionListener{

    //Panel
    JPanel pbtn1;
    JPanel pbtn2;
    JPanel pbtn3;
    //Buttons
    JButton btnActor;
    JButton btnMovie;
    JButton btnDistributor;
    JButton btnDirector;
    JButton btnAcademy;
    JButton btnMusic;
    JButton btnProduction;
    JButton btnStatistic;
    JButton btnReview;
    JButton btnDate;
    JButton btnSuccess;
    JButton btnView1;
    JButton btnView2;
    JButton IndexUse;

    public Movie_List(){
        super("MovieInfo : team01 database project");

        //Panels
        pbtn1 = new JPanel();//North
        pbtn2 = new JPanel();//Center
        pbtn3 = new JPanel();//South
        //TableShow Buttons
        btnActor = new JButton("배우");
        btnMovie = new JButton("영화");
        btnDistributor = new JButton("배급사");
        btnDirector = new JButton("감독");
        btnAcademy = new JButton("시상식");
        btnMusic = new JButton("영화 음악");
        btnProduction = new JButton("제작사");
        btnStatistic = new JButton("영화 통계량");
        btnReview = new JButton("영화 평점");
        btnDate = new JButton("영화 관련 날짜");
        //requested query buttons
        btnSuccess = new JButton("관객수 500만이상");
        btnView1 = new JButton("view1");
        btnView2 = new JButton("view2");
        IndexUse = new JButton("indexUsed");
```

```
        //Add buttons to panel
        pbtn1.add(btnActor);
        pbtn1.add(btnMovie);
        pbtn1.add(btnDistributor);
        pbtn1.add(btnDirector);
        pbtn1.add(btnAcademy);
        pbtn2.add(btnMusic);
        pbtn2.add(btnProduction);
        pbtn2.add(btnStatistic);
        pbtn2.add(btnReview);
        pbtn2.add(btnDate);
        pbtn3.add(btnSuccess);
        pbtn3.add(btnView1);
        pbtn3.add(btnView2);
        pbtn3.add(IndexUse);
        //Add panels to Frame
        add(pbtn1,BorderLayout.NORTH);
        add(pbtn2,BorderLayout.CENTER);
        add(pbtn3,BorderLayout.SOUTH);

        //Action handling
        btnActor.addActionListener(this);
        btnMovie.addActionListener(this);
        btnDistributor.addActionListener(this);
        btnDirector.addActionListener(this);
        btnAcademy.addActionListener(this);
        btnMusic.addActionListener(this);
        btnProduction.addActionListener(this);
        btnStatistic.addActionListener(this);
        btnReview.addActionListener(this);
        btnDate.addActionListener(this);
        btnSuccess.addActionListener(this);
        btnView1.addActionListener(this);
        btnView2.addActionListener(this);
        IndexUse.addActionListener(this);
        //set frame size
        setSize(700,400);
        setVisible(true);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }


    public static void main(String[] args) {
        new Movie_List();
    }//main
    @Override
    public void mouseClicked(MouseEvent e) {

    }
    @Override
    public void mouseEntered(MouseEvent e) {
        // TODO Auto-generated method stub

    }
    @Override
    public void mouseExited(MouseEvent e) {
        // TODO Auto-generated method stub
    }
    @Override
    public void mousePressed(MouseEvent e) {
        // TODO Auto-generated method stub
    }
    @Override
    public void mouseReleased(MouseEvent e) {
        // TODO Auto-generated method stub
    }
```

```java
@Override
public void actionPerformed(ActionEvent e) {
    //버튼을 클릭하면
    if(e.getSource() == btnActor ){
        new ActorJTabaleExam();
    }
    if(e.getSource() == btnMovie ){
        new MovieJTabaleExam();
    }
    if(e.getSource() == btnDistributor){
        new  DistributorJTableExam();
    }
    if(e.getSource() == btnDirector){
        new  DirectorJTableExam();
    }
    if(e.getSource() == btnAcademy){
        new AcademyJTabaleExam();
    }
    if(e.getSource() == btnMusic){
        new MusicJTabaleExam();
    }
    if(e.getSource() == btnProduction){
        new  ProductionJTableExam();
    }
    if(e.getSource() == btnStatistic){
        new StatisticJTabaleExam();
    }
    if(e.getSource() == btnReview){
        new ReviewJTabaleExam();
    }
    if(e.getSource() == btnDate){
        new DateJTabaleExam();
    }
    if(e.getSource() == btnSuccess){
        new JoinQuery();
    }
        if(e.getSource() == btnView2){
            new ViewWithNestedQuery();
            }
        if(e.getSource() == IndexUse){
            new IndexQuery();
        }

    }

}
```

-AcademyDefaultJTableDAO

```
import java.sql.Connection;

public class AcademyDefaultJTableDAO {

    Connection con;
    Statement st;
    PreparedStatement ps;
    ResultSet rs;

    public AcademyDefaultJTableDAO() {
        try {
            //load driver
            Class.forName("com.mysql.jdbc.Driver");
            //connect
            con = DriverManager.getConnection("jdbc:mysql://localhost:3306/team01?useSSL=false","team01","team01");

        } catch (ClassNotFoundException e) {
            System.out.println(e );//load fail
        } catch (SQLException e) {
            System.out.println(e);//connection fail
        }
    }//constructor


    public void dbClose() {
        try {
            if (rs != null) rs.close();
            if (st != null) st.close();
            if (ps != null) ps.close();
        } catch (Exception e) {
            System.out.println(e + "=> dbClose fail");
        }
    }//dbClose() ---
```

```
public int acdListInsert(AcademyJDailogGUI user) {
    int result = 0;
    try {
        ps = con.prepareStatement("insert into DBCOURSE_ACADEMY values(?,?,?,?,?)");
        ps.setString(1, user.Academy_Name.getText());
        ps.setString(2, user.Nation.getText());
        ps.setString(3, user.Movie.getText());
        ps.setString(4, user.Award_Name.getText());
        ps.setInt(5, Integer.parseInt(user.Year.getText()));

        result = ps.executeUpdate(); //실행 -> 저장

    } catch (SQLException e) {
        System.out.println(e + "=> userListInsert fail");
    } finally {
        dbClose();
    }

    return result;

}//acdListInsert()
```

```java
public void acdSelectAll(DefaultTableModel t_model) {
    try {
        st = con.createStatement();
        rs = st.executeQuery("select * from DBCOURSE_ACADEMY order by academy_name");

        // DefaultTableModel에 있는 기존 데이터 지우기
        for (int i = 0; i < t_model.getRowCount();) {
            t_model.removeRow(0);
        }

        while (rs.next()) {
            Object data[] = { rs.getString(1), rs.getString(2),rs.getString(3),
                    rs.getString(4), rs.getString(5) };

            t_model.addRow(data); //DefaultTableModel에 레코드 추가
        }

    } catch (SQLException e) {
        System.out.println(e + "=> userSelectAll fail");
    } finally {
        dbClose();
    }
}//acdSelectAll()
```

```java
 public int acdDelete(String Academy_Name) {
     int result = 0;
     try {
         ps = con.prepareStatement("delete from DBCOURSE_ACADEMY  where academy_name = ? ");
         ps.setString(1, Academy_Name.trim());
         result = ps.executeUpdate();

     } catch (SQLException e) {
         System.out.println(e + "=> userDelete fail");
     }finally {
         dbClose();
     }

     return result;
 }//userDelete()
```

```java
public int acdUpdate(AcademyJDailogGUI user) {
    int result = 0;
    String sql = "UPDATE DBCOURSE_ACADEMY SET nation=?, movie=?, award_name=?, awardyear=? WHERE academy_name=?";

    try {
        ps = con.prepareStatement(sql);
        // ?의 순서대로 값 넣기
        ps.setString(5, user.Academy_Name.getText());
        ps.setString(1, user.Nation.getText());
        ps.setString(2, user.Movie.getText());
        ps.setString(3, user.Award_Name.getText().trim());
        ps.setString(4, user.Year.getText());

        // 실행하기
        result = ps.executeUpdate();

    } catch (SQLException e) {
        System.out.println(e + "=> Update fail");
    } finally {
        dbClose();
    }

    return result;
}//userUpdate()
```

```
    public void acdSearch(DefaultTableModel dt, String fieldName,
            String word) {
        String sql = "SELECT * FROM DBCOURSE_ACADEMY WHERE " + fieldName.trim()
                + " LIKE '%" + word.trim() + "%'";

        try {
            st = con.createStatement();
            rs = st.executeQuery(sql);

            // DefaultTableModel에 있는 기존 데이터 지우기
            for (int i = 0; i < dt.getRowCount();) {
                dt.removeRow(0);
            }

            while (rs.next()) {
                Object data[] = { rs.getString(1), rs.getString(2),rs.getString(3),
                        rs.getInt(4), rs.getString(5) };

                dt.addRow(data);
            }

        } catch (SQLException e) {
            System.out.println(e + "=> getUserSearch fail");
        } finally {
            dbClose();
        }

    }//getSearch()

}// 클래스끝
```

-AcademyGUI

```
 2 import java.awt.BorderLayout;
14
15 public class AcademyJDailogGUI extends JDialog implements ActionListener{
16
17     JPanel pw=new JPanel(new GridLayout(5,1));
18     JPanel pc=new JPanel(new GridLayout(5,1));
19     JPanel ps=new JPanel();
20
21     JLabel lable_Academy_Name = new JLabel("Academy Name");
22     JLabel lable_Nation=new JLabel("Nation");
23     JLabel lable_Movie=new JLabel("Movie");
24     JLabel lable_Award_Name=new JLabel("Award_Name");
25     JLabel lable_Year=new JLabel("Year");
26
27
28     JTextField Academy_Name=new JTextField();
29     JTextField Nation=new JTextField();
30     JTextField Movie=new JTextField();
31     JTextField Award_Name=new JTextField();
32     JTextField Year = new JTextField();
33
34
35     JButton confirm;
36     JButton reset=new JButton("cancle");
37
38     AcademyJTabaleExam me;
39
40
41     AcademyDefaultJTableDAO dao =new AcademyDefaultJTableDAO();
42
```

```
public AcademyJDailogGUI(AcademyJTabaleExam me, String index){
    super(me,"Academy Table");
    this.me=me;
    if(index.equals("insert")){
        confirm=new JButton(index);
    }else{
        confirm=new JButton("update");

        //text박스에 선택된 레코드의 정보 넣기
        int row = me.jt.getSelectedRow();//선택된 행
        Academy_Name.setText( me.jt.getValueAt(row, 0).toString() );
        Nation.setText( me.jt.getValueAt(row, 1).toString() );
        Movie.setText( me.jt.getValueAt(row, 2).toString() );
        Award_Name.setText( me.jt.getValueAt(row, 3).toString() );
        Year.setText( me.jt.getValueAt(row, 4).toString() );

        //id text박스 비활성
        Academy_Name.setEditable(false);

    }

    //Label추가부분
    pw.add(lable_Academy_Name);//시상식이름
    pw.add(lable_Nation);//국가
    pw.add(lable_Movie);//영화
    pw.add(lable_Award_Name);//상이름
    pw.add(lable_Year);//연도

   //TextField 추가
   pc.add(Academy_Name);
   pc.add(Nation);
   pc.add(Movie);
   pc.add(Award_Name);
   pc.add(Year);

   ps.add(confirm); //update
   ps.add(reset);//cancle

   add(pw,"West");
   add(pc,"Center");
   add(ps,"South");

   setSize(500,400);
   setVisible(true);

   setDefaultCloseOperation(JDialog.DISPOSE_ON_CLOSE);

   //이벤트등록
  confirm.addActionListener(this); //insert/update event
  reset.addActionListener(this); //cancle event

}//Constructor end
//insert/update/delete
@Override
```

```java
@Override
public void actionPerformed(ActionEvent e) {
   String btnLabel =e.getActionCommand();//이벤트주체 대한 Label 가져오기

   if(btnLabel.equals("insert")){
       if(dao.acdListInsert(this) > 0 ){//가입성공
           messageBox(this , Academy_Name.getText()+"insert success");
           dispose();//JDialog 창닫기

           dao.acdSelectAll(me.dt);//모든레코드가져와서 DefaultTableModel에 올리기

           if(me.dt.getRowCount() > 0)
               me.jt.setRowSelectionInterval(0, 0);//첫번째 행 선택

       }else{//가입실패
           messageBox(this,"insert failure");
       }

   }else if(btnLabel.equals("update")){

        if( dao.acdUpdate(this) > 0){
            messageBox(this, "update complete");
            dispose();
            dao.acdSelectAll(me.dt);
            if(me.dt.getRowCount() > 0 ) me.jt.setRowSelectionInterval(0, 0);

        }else{
            messageBox(this, "Update failure");
        }


   }else if(btnLabel.equals("cancle")){
       dispose();
   }

}//actionPerformed end
   * */
  public static void messageBox(Object obj , String message){
      JOptionPane.showMessageDialog( (Component)obj , message);
  }
```

-AcademyJTableExam

```
import java.awt.Color;


public class AcademyJTabaleExam extends JFrame implements ActionListener {
    JMenu m = new JMenu("Menu");
    JMenuItem insert = new JMenuItem("insert");
    JMenuItem update = new JMenuItem("update");
    JMenuItem delete = new JMenuItem("delete");
    JMenuItem quit = new JMenuItem("exit");
    JMenuBar mb = new JMenuBar();

    String[] name = { "Acdemy Name", "Nation", "Movie","Award Name", "Year" };

    DefaultTableModel dt = new DefaultTableModel(name, 0);
    JTable jt = new JTable(dt);
    JScrollPane jsp = new JScrollPane(jt);

    /*
     * South 영역에 추가할 Componet들
     */
    JPanel p = new JPanel();
    String[] comboName = { " ALL ", " Academy Name "," Nation ", " Movie ", " Award Name ", " Year " };

    JComboBox combo = new JComboBox(comboName);
    JTextField jtf = new JTextField(20);
    JButton serach = new JButton("Search");

    AcademyDefaultJTableDAO dao = new AcademyDefaultJTableDAO();

   /**
    * 화면구성 및 이벤트등록
    */
   public AcademyJTabaleExam() {

        super("GUI AcademyInfo - DB");

        //메뉴아이템을 메뉴에 추가
        m.add(insert);
        m.add(update);
        m.add(delete);
        m.add(quit);
        //메뉴를 메뉴바에 추가
        mb.add(m);

        //윈도우에 메뉴바 세팅
        setJMenuBar(mb);

        // South영역
        p.setBackground(Color.yellow);
        p.add(combo);
        p.add(jtf);
        p.add(serach);

        add(jsp, "Center");
        add(p, "South");

        setSize(700, 500);
        setVisible(true);

        // 이벤트등록
        insert.addActionListener(this);
        update.addActionListener(this);
        delete.addActionListener(this);
        quit.addActionListener(this);
        serach.addActionListener(this);
```

```
        // 모든레코드를 검색하여 DefaultTableModle에 올리기
        dao.acdSelectAll(dt);

        //첫번행 선택.
        if (dt.getRowCount() > 0)
            jt.setRowSelectionInterval(0, 0);

    }// 생성자끝

    /**
     * 가입/수정/삭제/검색기능을 담당하는 메소드
     * */

    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == insert) {// 가입 메뉴아이템 클릭
            new AcademyJDailogGUI(this, "insert");

        } else if (e.getSource() == update) {// 수정 메뉴아이템 클릭
            new AcademyJDailogGUI(this, "update");

        } else if (e.getSource() == delete) {// 삭제 메뉴아이템 클릭
            // 현재 Jtable의 선택된 행과 열의 값을 얻어온다.
            int row = jt.getSelectedRow();
            System.out.println("선택행 : " + row);

            Object obj = jt.getValueAt(row, 0);// 행 열에 해당하는 value
            System.out.println("값 : " + obj);

            if (dao.acdDelete(obj.toString()) > 0) {
                AcademyJDailogGUI.messageBox(this, "Record Delete.");

                //리스트 갱신
                dao.acdSelectAll(dt);
                if (dt.getRowCount() > 0)
                    jt.setRowSelectionInterval(0, 0);

            }

        } else if (e.getSource() == quit) {// 종료 메뉴아이템 클릭
            System.exit(0);

        } else if (e.getSource() == serach) {// 검색 버튼 클릭
            // JComboBox에 선택된 value 가져오기
            String fieldName = combo.getSelectedItem().toString();
            System.out.println("필드명 " + fieldName);

            if (fieldName.trim().equals("ALL")) {// 전체검색
                dao.acdSelectAll(dt);
                if (dt.getRowCount() > 0)
                    jt.setRowSelectionInterval(0, 0);
            } else {
                if (jtf.getText().trim().equals("")) {
                    AcademyJDailogGUI.messageBox(this, "검색단어를 입력해주세요!");
                    jtf.requestFocus();
                } else {// 검색어를 입력했을경우
                    dao.acdSearch(dt, fieldName, jtf.getText());
                    if (dt.getRowCount() > 0)
                        jt.setRowSelectionInterval(0, 0);
                }
            }
        }

    }//actionPerformed()----------

}
```

```java
import java.sql.Connection;

public class ActorDefaultJTableDAO {

    Connection con;
    Statement st;
    PreparedStatement ps;
    ResultSet rs;

    public ActorDefaultJTableDAO() {
        try {
            //load file
            Class.forName("com.mysql.jdbc.Driver");

            con = DriverManager.getConnection("jdbc:mysql://localhost:3306/team01?useSSL=false","team01","team01");

        } catch (ClassNotFoundException e) {
            System.out.println(e + "=> load fail");
        } catch (SQLException e) {
            System.out.println(e + "=> connection fail");
        }
    }//Constructor

    /**
     * DB닫기 기능 메소드
     * */
    public void dbClose() {
        try {
            if (rs != null) rs.close();
            if (st != null) st.close();
            if (ps != null) ps.close();
        } catch (Exception e) {
            System.out.println(e + "=> dbClose fail");
        }
    }//dbClose() ---
```

```java
public int actListInsert(ActorJDailogGUI actorJDailogGUI) {
    int result = 0;
    try {
        ps = con.prepareStatement("insert into DBCOURSE_ACTOR values(?,?,?,?,?)");
        ps.setString(1, actorJDailogGUI.Actor_Name.getText());
        ps.setString(2, actorJDailogGUI.Nationality.getText());
        ps.setString(3, actorJDailogGUI.Gender.getText());
        ps.setInt(4, Integer.parseInt(actorJDailogGUI.Debut_Year.getText()));
        ps.setString(5, actorJDailogGUI.Film.getText());

        result = ps.executeUpdate(); //실행 -> 저장

    } catch (SQLException e) {
        System.out.println(e + "=> userListInsert fail");
    } finally {
        dbClose();
    }

    return result;

}//userListInsert()
```

```
public void actSelectAll(DefaultTableModel t_model) {
    try {
        st = con.createStatement();
        rs = st.executeQuery("select * from DBCOURSE_ACTOR order by Actor_Name");

        // DefaultTableModel에 있는 기존 데이터 지우기
        for (int i = 0; i < t_model.getRowCount();) {
            t_model.removeRow(0);
        }

        while (rs.next()) {
            Object data[] = { rs.getString(1), rs.getString(2),rs.getString(3),
                    rs.getInt(4), rs.getString(5) };

            t_model.addRow(data); //DefaultTableModel에 레코드 추가
        }

    } catch (SQLException e) {
        System.out.println(e + "=> userSelectAll fail");
    } finally {
        dbClose();
    }
}//userSelectAll()

public int actDelete(String Actor_Name) {
    int result = 0;
    try {
        ps = con.prepareStatement("delete from DBCOURSE_ACTOR  where Actor_Name = ? ");
        ps.setString(1, Actor_Name.trim());
        result = ps.executeUpdate();

    } catch (SQLException e) {
        System.out.println(e + "=> userDelete fail");
    }finally {
        dbClose();
    }

    return result;
}//userDelete()

public int actUpdate(ActorJDailogGUI actorJDailogGUI) {
    int result = 0;
    String sql = "UPDATE DBCOURSE_ACTOR SET actor_nationality=?, gender=?,debut_year=? , film=? WHERE actor_name=?";

    try {
        ps = con.prepareStatement(sql);
        // ?의 순서대로 값 넣기
        ps.setString(5, actorJDailogGUI.Actor_Name.getText());
        ps.setString(1, actorJDailogGUI.Nationality.getText());
        ps.setString(2, actorJDailogGUI.Gender.getText());
        ps.setInt(3, Integer.parseInt(actorJDailogGUI.Debut_Year.getText().trim()));
        ps.setString(4, actorJDailogGUI.Film.getText());

        // 실행하기
        result = ps.executeUpdate();

    } catch (SQLException e) {
        System.out.println(e + "=> Update fail");
    } finally {
        dbClose();
    }

    return result;
}//userUpdate()
```

```
    public void actSearch(DefaultTableModel dt, String fieldName,
            String word) {
        String sql = "SELECT * FROM DBCOURSE_ACTOR WHERE " + fieldName.trim()
                + " LIKE '%" + word.trim() + "%'";

        try {
            st = con.createStatement();
            rs = st.executeQuery(sql);

            // DefaultTableModel에 있는 기존 데이터 지우기
            for (int i = 0; i < dt.getRowCount();) {
                dt.removeRow(0);
            }

            while (rs.next()) {
                Object data[] = { rs.getString(1), rs.getString(2),rs.getString(3),
                        rs.getInt(4), rs.getString(5) };

                dt.addRow(data);
            }

        } catch (SQLException e) {
            System.out.println(e + "=> getUserSearch fail");
        } finally {
            dbClose();
        }

    }//getUserSearch()

}// 클래스끝
```

```
import java.awt.BorderLayout;

public class ActorJDailogGUI extends JDialog implements ActionListener{

    JPanel pw = new JPanel(new GridLayout(5,1));
    JPanel pc = new JPanel(new GridLayout(5,1));
    JPanel ps = new JPanel();

    JLabel lable_Actor_Name = new JLabel("Name");
    JLabel lable_Nationality = new JLabel("Nationality");
    JLabel lable_Gender = new JLabel("Gender");
    JLabel lable_Debut_Year = new JLabel("Debut Year");
    JLabel lable_Film = new JLabel("Film");


    JTextField Actor_Name = new JTextField();
    JTextField Nationality = new JTextField();
    JTextField Gender = new JTextField();
    JTextField Debut_Year = new JTextField();
    JTextField Film = new JTextField();


    ActorDefaultJTableDAO dao1 = new ActorDefaultJTableDAO();

    JButton confirm;
    JButton reset = new JButton("cancle");

   ActorJTabaleExam me;

   ActorDefaultJTableDAO dao =new ActorDefaultJTableDAO();
```

```
public ActorJDailogGUI(ActorJTabaleExam me, String index){
    super(me,"Actor Table");
    this.me=me;
    if(index.equals("insert")){
        confirm=new JButton(index);
    }else{
        confirm=new JButton("update");

        //text박스에 선택된 레코드의 정보 넣기
        int row = me.jt.getSelectedRow();//선택된 행
        Actor_Name.setText( me.jt.getValueAt(row, 0).toString() );
        Nationality.setText( me.jt.getValueAt(row, 1).toString() );
        Debut_Year.setText( me.jt.getValueAt(row, 3).toString() );
        Film.setText( me.jt.getValueAt(row, 4).toString() );

        //id text박스 비활성
        Actor_Name.setEditable(false);

    }


    //Label추가부분
    pw.add(lable_Actor_Name);//배우 이름
    pw.add(lable_Nationality);//배우 국적
    pw.add(lable_Gender);//배우 성별
    pw.add(lable_Debut_Year);//배우 데뷔 년도
    pw.add(lable_Film);//배우 작품

    //TextField 추가
    pc.add(Actor_Name);
    pc.add(Nationality);
    pc.add(Gender);
    pc.add(Debut_Year);
    pc.add(Film);

     ps.add(confirm); //update
     ps.add(reset);//cancle

     add(pw,"West");
     add(pc,"Center");
     add(ps,"South");

     setSize(500,350);
     setVisible(true);

     setDefaultCloseOperation(JDialog.DISPOSE_ON_CLOSE);

     //이벤트등록
     confirm.addActionListener(this); //가입/수정 이벤트등록
     reset.addActionListener(this); //취소 이벤트등록

}//End of Constructor
```

```
public void actionPerformed(ActionEvent e) {
   String btnLabel =e.getActionCommand();//get event

   if(btnLabel.equals("insert")){
       if(dao1.actListInsert(this) > 0 ){//insert complete
           messageBox(this , Actor_Name.getText()+" insert complete");
           dispose();//JDialog exit

           dao1.actSelectAll(me.dt);//get all record and insert into DefaultTableModel

           if(me.dt.getRowCount() > 0)
               me.jt.setRowSelectionInterval(0, 0);//select first row

       }else{
           messageBox(this,"insert failure");
       }

   }else if(btnLabel.equals("update")){

        if( dao1.actUpdate(this) > 0){
            messageBox(this, "Update complete");
            dispose();
            dao1.actSelectAll(me.dt);
            if(me.dt.getRowCount() > 0 ) me.jt.setRowSelectionInterval(0, 0);

        }else{
            messageBox(this, "Update Failure");
        }

   }else if(btnLabel.equals("cancle")){
       dispose();

   }


}//actionPerformed end
```

```
import java.awt.Color;


public class ActorJTabaleExam extends JFrame implements ActionListener {
    JMenu m = new JMenu("관리");
    JMenuItem insert = new JMenuItem("insert");
    JMenuItem update = new JMenuItem("update");
    JMenuItem delete = new JMenuItem("delete");
    JMenuItem quit = new JMenuItem("exit");
    JMenuBar mb = new JMenuBar();

    String[] name = { "Name", "Nationality", "Gender", "Debut Year", "Film" };

    DefaultTableModel dt = new DefaultTableModel(name, 0);
    JTable jt = new JTable(dt);
    JScrollPane jsp = new JScrollPane(jt);

    /*
     * South 영역에 추가할 Componet들
     */
    JPanel p = new JPanel();
    String[] comboName = { " ALL  ", " Actor_Name  "," Nationality  ", " Gender ", " Debut Year ", " Film " };

    JComboBox combo = new JComboBox(comboName);
    JTextField jtf = new JTextField(20);
    JButton serach = new JButton("Search");

    ActorDefaultJTableDAO dao = new ActorDefaultJTableDAO();
```

```
public ActorJTabaleExam() {

    super("GUI movieInfo - DB");

    //메뉴아이템을 메뉴에 추가
    m.add(insert);
    m.add(update);
    m.add(delete);
    m.add(quit);
    //메뉴를 메뉴바에 추가
    mb.add(m);

    //윈도우에 메뉴바 세팅
    setJMenuBar(mb);


    // South영역
    p.setBackground(Color.yellow);
    p.add(combo);
    p.add(jtf);
    p.add(serach);

    add(jsp, "Center");
    add(p, "South");

    setSize(700, 500);
    setVisible(true);

    //super.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    // 이벤트등록
    insert.addActionListener(this);
    update.addActionListener(this);
    delete.addActionListener(this);
    quit.addActionListener(this);
    serach.addActionListener(this);

    // 모든데이터를 검색하여 DefaultTableModel에 출력
    dao.actSelectAll(dt);

    //첫번행 선택.
    if (dt.getRowCount() > 0)
        jt.setRowSelectionInterval(0, 0);

}// 생성자끝
```

```
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == insert) {// 가입 메뉴아이템 클릭
        new ActorJDailogGUI(this, "insert");

    } else if (e.getSource() == update) {// 수정 메뉴아이템 클릭
        new ActorJDailogGUI(this, "update");

    } else if (e.getSource() == delete) {// 삭제 메뉴아이템 클릭
        // 현재 Jtable의 선택된 행과 열의 값을 얻어온다.
        int row = jt.getSelectedRow();
        System.out.println("선택행 : " + row);

        Object obj = jt.getValueAt(row, 0);// 행 열에 해당하는 value
        System.out.println("값 : " + obj);

        if (dao.actDelete(obj.toString()) > 0) {
            ActorJDailogGUI.messageBox(this, "Record Delete.");

            //리스트 갱신
            dao.actSelectAll(dt);
            if (dt.getRowCount() > 0)
                jt.setRowSelectionInterval(0, 0);

        } else {
            ActorJDailogGUI.messageBox(this, "failed deleting record.");
        }

    } else if (e.getSource() == quit) {// 종료 메뉴아이템 클릭
        System.exit(0);


      } else if (e.getSource() == serach) {// 검색 버튼 클릭
          // JComboBox에 선택된 value 가져오기
          String fieldName = combo.getSelectedItem().toString();
          System.out.println("필드명 " + fieldName);

          if (fieldName.trim().equals("ALL")) {// 전체검색
              dao.actSelectAll(dt);
              if (dt.getRowCount() > 0)
                  jt.setRowSelectionInterval(0, 0);
          } else {
              if (jtf.getText().trim().equals("")) {
                  ActorJDailogGUI.messageBox(this, "검색단어를 입력해주세요!");
                  jtf.requestFocus();
              } else {// 검색어를 입력했을경우
                  dao.actSearch(dt, fieldName, jtf.getText());
                  if (dt.getRowCount() > 0)
                      jt.setRowSelectionInterval(0, 0);
              }
          }
      }

  }//actionPerformed()----------

}
```

```java
import java.sql.Connection;

public class DateDefaultJTableDAO {

    Connection con;
    Statement st;
    PreparedStatement ps;
    ResultSet rs;

    public DateDefaultJTableDAO() {
        try {
            // 로드
            Class.forName("com.mysql.jdbc.Driver");

            con = DriverManager.getConnection("jdbc:mysql://localhost:3306/team01?useSSL=false","team01","team01");

        } catch (ClassNotFoundException e) {
            System.out.println(e + "=> load fail");
        } catch (SQLException e) {
            System.out.println(e + "=> connection fail");
        }
    }//생성자

    /**
     * DB닫기 기능 메소드
     * */
    public void dbClose() {
        try {
            if (rs != null) rs.close();
            if (st != null) st.close();
            if (ps != null) ps.close();
        } catch (Exception e) {
            System.out.println(e + "=> dbClose fail");
        }
    }//dbClose() ---


    public boolean getIdByCheck(String Mov) {
        boolean result = true;

        try {
            ps = con.prepareStatement("SELECT * FROM DBCOURSE_MOVIE_DATE WHERE Movie=?");
            ps.setString(1, Mov.trim());
            rs = ps.executeQuery(); //실행
            if (rs.next())
                result = false; //레코드가 존재하면 false

        } catch (SQLException e) {
            System.out.println(e + "=>  getIdByCheck fail");
        } finally {
            dbClose();
        }

        return result;

    }//getIdByCheck()

    /**
     * userlist 회원가입하는 기능 메소드
     * */
    public int dateListInsert(DateJDailogGUI user) {
        int result = 0;
        try {
            ps = con.prepareStatement("insert into DBCOURSE_MOVIE_DATE values(?,?,?,?)");
            ps.setString(1, user.Mov.getText());
            //date
            ps.setString(2, user.Release_Date.getText());
            ps.setString(3, user.Crank_In.getText());
            ps.setString(4, user.Crank_Up.getText());

            result = ps.executeUpdate(); //실행 -> 저장
```

```java
public int dateListInsert(DateJDailogGUI user) {
    int result = 0;
    try {
        ps = con.prepareStatement("insert into DBCOURSE_MOVIE_DATE values(?,?,?,?)");
        ps.setString(1, user.Mov.getText());
        //date
        ps.setString(2, user.Release_Date.getText());
        ps.setString(3, user.Crank_In.getText());
        ps.setString(4, user.Crank_Up.getText());

        result = ps.executeUpdate(); //실행 -> 저장

    } catch (SQLException e) {
        System.out.println(e + "=> userListInsert fail");
    } finally {
        dbClose();
    }

    return result;

}//userListInsert()

/**
 * userlist의 모든 레코드 조회
 * */

public void dateSelectAll(DefaultTableModel t_model) {
    try {
        st = con.createStatement();
        rs = st.executeQuery("select * from DBCOURSE_MOVIE_DATE order by movie");

        // DefaultTableModel에 있는 기존 데이터 지우기
        for (int i = 0; i < t_model.getRowCount();) {
            t_model.removeRow(0);
        }

        while (rs.next()) {
            Object data[] = { rs.getString(1), rs.getString(2),rs.getString(3),
                    rs.getInt(4) };

            t_model.addRow(data); //DefaultTableModel에 레코드 추가
        }

    } catch (SQLException e) {
        System.out.println(e + "=> userSelectAll fail");
    } finally {
        dbClose();
    }
}//userSelectAll()
```

```
public int dateDelete(String Mov) {
    int result = 0;
    try {
        ps = con.prepareStatement("delete from DBCOURSE_MOVIE_DATE  where movie = ? ");
        ps.setString(1, Mov.trim());
        result = ps.executeUpdate();

    } catch (SQLException e) {
        System.out.println(e + "=> userDelete fail");
    }finally {
        dbClose();
    }

    return result;
}//userDelete()

/**
 * ID에 해당하는 레코드 수정하기
 * */
public int dateUpdate(DateJDailogGUI user) {
    int result = 0;
    String sql = "UPDATE DBCOURSE_MOVIE_DATE SET release_date=?, crank_in_date=?, crank_up_date=? ,"+
    "director=? WHERE movie=?";

    try {
        ps = con.prepareStatement(sql);
        // ?의 순서대로 값 넣기
        ps.setString(4, user.Mov.getText());
        ps.setString(1, user.Release_Date.getText());
        ps.setString(2, user.Crank_In.getText());
        ps.setString(3, user.Crank_Up.getText().trim());

        // 실행하기
        result = ps.executeUpdate();

    } catch (SQLException e) {
        System.out.println(e + "=> Update fail");

     } finally {
        dbClose();
     }

     return result;
 }//userUpdate()

 /**
  * 검색단어에 해당하는 레코드 검색하기 (like연산자를 사용하여 _, %를 사용할때는 PreparedStatemnet안된다. 반드시
  * Statement객체를 이용함)
  * */
 public void dateSearch(DefaultTableModel dt, String fieldName,
         String word) {
     String sql = "SELECT * FROM DBCOURSE_MOVIE_DATE WHERE " + fieldName.trim()
             + " LIKE '%" + word.trim() + "%'";

     try {
         st = con.createStatement();
         rs = st.executeQuery(sql);

         // DefaultTableModel에 있는 기존 데이터 지우기
         for (int i = 0; i < dt.getRowCount();) {
             dt.removeRow(0);
         }

         while (rs.next()) {
             Object data[] = { rs.getString(1), rs.getString(2),rs.getString(3),
                     rs.getInt(4) };

             dt.addRow(data);
         }

     } catch (SQLException e) {
         System.out.println(e + "=> getUserSearch fail");
     } finally {
         dbClose();
     }
```

```java
import java.awt.BorderLayout;

public class DateJDailogGUI extends JDialog implements ActionListener{

    JPanel pw=new JPanel(new GridLayout(4,1));
    JPanel pc=new JPanel(new GridLayout(4,1));
    JPanel ps=new JPanel();

    JLabel lable_Movie = new JLabel("Movie");
    JLabel lable_Release_Date=new JLabel("Release Date");
    JLabel lable_Crank_In=new JLabel("Crank In");
    JLabel lable_Crank_Up=new JLabel("Crank Up");


    JTextField Mov=new JTextField();
    JTextField Release_Date=new JTextField();
    JTextField Crank_In=new JTextField();
    JTextField Crank_Up=new JTextField();


    JButton confirm;
    JButton reset=new JButton("cancle");

   DateJTabaleExam me;

   DateDefaultJTableDAO dao =new DateDefaultJTableDAO();


   public DateJDailogGUI(DateJTabaleExam me, String index){
       super(me,"MovieDate Table");
       this.me=me;
       if(index.equals("insert")){
           confirm=new JButton(index);
       }else{
           confirm=new JButton("update");

           //text박스에 선택된 레코드의 정보 넣기
           int row = me.jt.getSelectedRow();//선택된 행
           Mov.setText( me.jt.getValueAt(row, 0).toString() );
           Release_Date.setText( me.jt.getValueAt(row, 1).toString() );
           Crank_In.setText( me.jt.getValueAt(row, 2).toString() );
           Crank_Up.setText( me.jt.getValueAt(row, 3).toString() );

           //id text박스 비활성
           Mov.setEditable(false);

       }


       //Label추가부분
       pw.add(lable_Movie);//moviename
       pw.add(lable_Release_Date);//released date
       pw.add(lable_Crank_In);//crankin
       pw.add(lable_Crank_Up);//crankup

       //TextField 추가
       pc.add(Mov);
       pc.add(Release_Date);
       pc.add(Crank_In);
       pc.add(Crank_Up);
```

```
        ps.add(confirm); //update
        ps.add(reset);//cancle

        add(pw,"West");
        add(pc,"Center");
        add(ps,"South");

        setSize(500,350);
        setVisible(true);

        setDefaultCloseOperation(JDialog.DISPOSE_ON_CLOSE);

        //이벤트등록
        confirm.addActionListener(this); //가입/수정 이벤트등록
        reset.addActionListener(this); //취소 이벤트등록

    }//생성자끝

    /**
     * 가입/수정/삭제 기능에 대한 부분
     * */
    @Override
    public void actionPerformed(ActionEvent e) {
       String btnLabel =e.getActionCommand();//이벤트주체 대한 Label 가져오기

       if(btnLabel.equals("insert")){
           if(dao.dateListInsert(this) > 0 ){//가입성공
               messageBox(this , Mov.getText()+"님 가입축드립니다.");
               dispose();//JDialog 창닫기

               dao.dateSelectAll(me.dt);//모든레코드가져와서 DefaultTableModel에 올리기

               if(me.dt.getRowCount() > 0)
                   me.jt.setRowSelectionInterval(0, 0);//첫번째 행 선택

           }else{//가입실패
               messageBox(this,"가입되지 않았습니다.");


        }else if(btnLabel.equals("update")){

             if( dao.dateUpdate(this) > 0){
                 messageBox(this, "수정완료되었습니다.");
                 dispose();
                 dao.dateSelectAll(me.dt);
                 if(me.dt.getRowCount() > 0 ) me.jt.setRowSelectionInterval(0, 0);

             }else{
                 messageBox(this, "수정되지 않았습니다.");
             }


        }else if(btnLabel.equals("cancle")){
            dispose();
        }
    }//actionPerformed끝

    /**
     * 메시지박스 띄워주는 메소드 작성
     * */
    public static void messageBox(Object obj , String message){
        JOptionPane.showMessageDialog( (Component)obj , message);
    }

}//클래스끝
```

```java
import java.awt.Color;


public class DateJTabaleExam extends JFrame implements ActionListener {
    JMenu m = new JMenu("Menu");
    JMenuItem insert = new JMenuItem("insert");
    JMenuItem update = new JMenuItem("update");
    JMenuItem delete = new JMenuItem("delete");
    JMenuItem quit = new JMenuItem("exit");
    JMenuBar mb = new JMenuBar();

    String[] name = { "Movie", "Release Date", "Crank In", "Crank Up" };

    DefaultTableModel dt = new DefaultTableModel(name, 0);
    JTable jt = new JTable(dt);
    JScrollPane jsp = new JScrollPane(jt);

    /*
     * South 영역에 추가할 Componet들
     */
    JPanel p = new JPanel();
    String[] comboName = { " ALL ", "Movie", "Release Date", "Crank In", "Crank Up" };

    JComboBox combo = new JComboBox(comboName);
    JTextField jtf = new JTextField(20);
    JButton serach = new JButton("Search");

    DateDefaultJTableDAO dao = new DateDefaultJTableDAO();

    public DateJTabaleExam() {

        super("GUI MovieDateInfo - DB");

        //메뉴아이템을 메뉴에 추가
        m.add(insert);
        m.add(update);
        m.add(delete);
        m.add(quit);
        //메뉴를 메뉴바에 추가
        mb.add(m);

        //윈도우에 메뉴바 세팅
        setJMenuBar(mb);


        // South영역
        p.setBackground(Color.yellow);
        p.add(combo);
        p.add(jtf);
        p.add(serach);

        add(jsp, "Center");
        add(p, "South");

        setSize(700, 500);
        setVisible(true);

        // 이벤트등록
        insert.addActionListener(this);
        update.addActionListener(this);
        delete.addActionListener(this);
        quit.addActionListener(this);
        serach.addActionListener(this);

        // 모든레코드를 검색하여 DefaultTableModle에 올리기
        dao.dateSelectAll(dt);
```

```
        //첫번행 선택.
        if (dt.getRowCount() > 0)
            jt.setRowSelectionInterval(0, 0);

    }// 생성자끝

    //insert/update/delete
    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == insert) {// 가입 메뉴아이템 클릭
            new DateJDailogGUI(this, "insert");

        } else if (e.getSource() == update) {// 수정 메뉴아이템 클릭
            new DateJDailogGUI(this, "update");

        } else if (e.getSource() == delete) {// 삭제 메뉴아이템 클릭
            // 현재 Jtable의 선택된 행과 열의 값을 얻어온다.
            int row = jt.getSelectedRow();
            System.out.println("선택행 : " + row);

            Object obj = jt.getValueAt(row, 0);// 행 열에 해당하는 value
            System.out.println("값 : " + obj);

            if (dao.dateDelete(obj.toString()) > 0) {
                DateJDailogGUI.messageBox(this, "Record Delete.");

                //리스트 갱신
                dao.dateSelectAll(dt);
                if (dt.getRowCount() > 0)
                    jt.setRowSelectionInterval(0, 0);

            } else {
                DateJDailogGUI.messageBox(this, "failed deleting record.");
            }

        } else if (e.getSource() == quit) {// 종료 메뉴아이템 클릭
            System.exit(0);


          } else if (e.getSource() == serach) {// 검색 버튼 클릭
              // JComboBox에 선택된 value 가져오기
              String fieldName = combo.getSelectedItem().toString();
              System.out.println("필드명 " + fieldName);

              if (fieldName.trim().equals("ALL")) {// 전체검색
                  dao.dateSelectAll(dt);
                  if (dt.getRowCount() > 0)
                      jt.setRowSelectionInterval(0, 0);
              } else {
                  if (jtf.getText().trim().equals("")) {
                      DateJDailogGUI.messageBox(this, "검색단어를 입력해주세요!");
                      jtf.requestFocus();
                  } else {// 검색어를 입력했을경우
                      dao.dateSearch(dt, fieldName, jtf.getText());
                      if (dt.getRowCount() > 0)
                          jt.setRowSelectionInterval(0, 0);
                  }
              }
          }

      }//actionPerformed()----------

}
```

```java
import java.sql.Connection;

public class DirectorDefaultJTableDAO {

    Connection con;
    Statement st;
    PreparedStatement ps;
    ResultSet rs;

    /**
     * 로드 연결을 위한 생성자
     * @param movie_List
     * */
    public DirectorDefaultJTableDAO() {
        try {
            // 로드
            Class.forName("com.mysql.jdbc.Driver");

            con = DriverManager.getConnection("jdbc:mysql://localhost:3306/team01?useSSL=false","team01","team01");

        } catch (ClassNotFoundException e) {
            System.out.println(e + "=> load fail");
        } catch (SQLException e) {
            System.out.println(e + "=> connection fail");
        }
    }//생성자

    /**
     * DB닫기 기능 메소드
     * */
    public void dbClose() {
        try {
            if (rs != null) rs.close();
            if (st != null) st.close();
            if (ps != null) ps.close();
        } catch (Exception e) {
            System.out.println(e + "=> dbClose fail");
        }
    }//dbClose() ---

    /**
     * check if inserting data already exists
     * */
    public boolean dirCheck(String director_name) {
        boolean result = true;

        try {
            ps = con.prepareStatement("SELECT * FROM DBCOURSE_DIRECTOR WHERE director_name=?");
            ps.setString(1, director_name.trim());
            rs = ps.executeQuery(); //execute
            if (rs.next())
                result = false; //if record exists false

        } catch (SQLException e) {
            System.out.println(e + "=>  DirecCheck fail");
        } finally {
            dbClose();
        }

        return result;

    }//getDirectorCheck()

    /**
     * userlist 회원가입하는 기능 메소드
     * */
    public int dirListInsert(DirectorJDailogGUI dir) {
        int result = 0;
        try {
            ps = con.prepareStatement("insert into DBCOURSE_DIRECTOR values(?,?,?,?)");
            ps.setString(1, dir.txdirector_name.getText());
            ps.setString(2, dir.txdirector_nationality.getText());
            ps.setInt(3, Integer.parseInt(dir.txdebut_year.getText()));
            ps.setString(4, dir.txdebut_film.getText());
```

```
            result = ps.executeUpdate(); //실행 -> 저장

        } catch (SQLException e) {
            System.out.println(e + "=> userListInsert fail");
        } finally {
            dbClose();
        }

        return result;

    }//userListInsert()

    /**
     * userlist의 모든 레코드 조회
     * */
    public void dirSelectAll(DefaultTableModel t_model) {
        try {
            st = con.createStatement();
            rs = st.executeQuery("select * from DBCOURSE_DIRECTOR order by director_name");

            // DefaultTableModel에 있는 기존 데이터 지우기
            for (int i = 0; i < t_model.getRowCount();) {
                t_model.removeRow(0);
            }

            while (rs.next()) {
                Object data[] = { rs.getString(1), rs.getString(2), rs.getString(3), rs.getString(4) };

                t_model.addRow(data); //DefaultTableModel에 레코드 추가
            }

        } catch (SQLException e) {
            System.out.println(e + "=> userSelectAll fail");
        } finally {
            dbClose();
        }
    }//userSelectAll()


    public int dirDelete(String director_name) {
        int result = 0;
        try {
            ps = con.prepareStatement("delete from DBCOURSE_DIRECTOR  where director_name = ? ");
            ps.setString(1, director_name.trim());
            result = ps.executeUpdate();

        } catch (SQLException e) {
            System.out.println(e + "=> userDelete fail");
        }finally {
            dbClose();
        }

        return result;
    }//userDelete()

    /**
     * ID에 해당하는 레코드 수정하기
     * */
    public int dirUpdate(DirectorJDailogGUI user) {
        int result = 0;
        String sql = "UPDATE DBCOURSE_DIRECTOR SET director_nationality=?, debut_year=?, debut_film=? "+
        "WHERE director_name = ?";

        try {
            ps = con.prepareStatement(sql);
            // ?의 순서대로 값 넣기
            ps.setString(4, user.txdirector_name.getText());
            ps.setString(1, user.txdirector_nationality.getText());
            ps.setInt(2, Integer.parseInt(user.txdebut_year.getText()));
            ps.setString(3, user.txdebut_film.getText().trim());
            //execute
            result = ps.executeUpdate();
```

```
        } catch (SQLException e) {
            System.out.println(e + "=> userDelete fail");
        }finally {
            dbClose();
        }

        return result;
    }//userDelete()

    /**
     * ID에 해당하는 레코드 수정하기
     * */
    public int dirUpdate(DirectorJDailogGUI user) {
        int result = 0;
        String sql = "UPDATE DBCOURSE_DIRECTOR SET director_nationality=?, debut_year=?, debut_film=? "+
        "WHERE director_name = ?";

        try {
            ps = con.prepareStatement(sql);
            // ?의 순서대로 값 넣기
            ps.setString(4, user.txdirector_name.getText());
            ps.setString(1, user.txdirector_nationality.getText());
            ps.setInt(2, Integer.parseInt(user.txdebut_year.getText()));
            ps.setString(3, user.txdebut_film.getText().trim());
            //execute
            result = ps.executeUpdate();

        } catch (SQLException e) {
            System.out.println(e + "=> Update fail");
        } finally {
            dbClose();
        }

        return result;
    }//userUpdate()

      ..  ../
     public void getDirSearch(DefaultTableModel dt, String fieldName,
             String word) {
         String sql = "SELECT * FROM DBCOURSE_DIRECTOR WHERE " + fieldName.trim()
                 + " LIKE '%" + word.trim() + "%'";

         try {
             st = con.createStatement();
             rs = st.executeQuery(sql);

             // DefaultTableModel
             for (int i = 0; i < dt.getRowCount();) {
                 dt.removeRow(0);
             }

             while (rs.next()) {
                 Object data[] = { rs.getString(1), rs.getString(2), rs.getString(3), rs.getString(4) };

                 dt.addRow(data);
             }

         } catch (SQLException e) {
             System.out.println(e + "=> getUserSearch fail");
         } finally {
             dbClose();
         }

     }//getDirSearch()

}//End of Class
```

```java
import java.awt.BorderLayout;

public class DirectorJDailogGUI extends JDialog implements ActionListener{

    JPanel pw=new JPanel(new GridLayout(5,1));
    JPanel pc=new JPanel(new GridLayout(5,1));
    JPanel ps=new JPanel();

    JLabel lable_director_name = new JLabel("Director");
    JLabel lable_director_nationality=new JLabel("Nationality");
    JLabel lable_debut_year=new JLabel("Debut Year");
    JLabel lable_debut_film=new JLabel("Debut Movie");


    JTextField txdirector_name=new JTextField();
    JTextField txdirector_nationality=new JTextField();
    JTextField txdebut_year=new JTextField();
    JTextField txdebut_film=new JTextField();

    JButton confirm;
    JButton reset=new JButton("cancle");

   DirectorJTableExam me;

   DirectorDefaultJTableDAO dao =new DirectorDefaultJTableDAO();


    public DirectorJDailogGUI(DirectorJTableExam me, String index){
        super(me,"Director Table");
        this.me=me;
        if(index.equals("insert")){
            confirm=new JButton(index);
        }else{
            confirm=new JButton("update");

            //text박스에 선택된 레코드의 정보 넣기
            int row = me.jt.getSelectedRow();//선택된 행
            txdirector_name.setText( me.jt.getValueAt(row, 0).toString() );
            txdirector_nationality.setText( me.jt.getValueAt(row, 1).toString() );
            txdebut_year.setText( me.jt.getValueAt(row, 2).toString() );
            txdebut_film.setText( me.jt.getValueAt(row, 3).toString() );

            //id text박스 비활성
            txdirector_name.setEditable(false);

        }

        //Label추가부분
        pw.add(lable_director_name);//name
        pw.add(lable_director_nationality);//nationality
        pw.add(lable_debut_year);//debut year
        pw.add(lable_debut_film);//debut film

        //TextField 추가
        pc.add(txdirector_name);
        pc.add(txdirector_nationality);
        pc.add(txdebut_year);
        pc.add(txdebut_film);

        ps.add(confirm); //update
        ps.add(reset);//cancle
```

```
        add(pw,"West");
        add(pc,"Center");
        add(ps,"South");

        setSize(500,400);
        setVisible(true);

        setDefaultCloseOperation(JDialog.DISPOSE_ON_CLOSE);

        //Event
        confirm.addActionListener(this); //insert/update
        reset.addActionListener(this); //cancle
    }//생성자끝

    //Actionlistener
    @Override
    public void actionPerformed(ActionEvent e) {
       String btnLabel =e.getActionCommand();//get event

       if(btnLabel.equals("insert")){
           if(dao.dirListInsert(this) > 0 ){//Success
               messageBox(this , txdirector_name.getText()+"New director record");
               dispose();//JDialog 창닫기

               dao.dirSelectAll(me.dt);//모든레코드가져와서 DefaultTableModel에 올리기

               if(me.dt.getRowCount() > 0)
                   me.jt.setRowSelectionInterval(0, 0);//select first row

           }else{
               messageBox(this,"Insert Failure");
           }

       }else if(btnLabel.equals("update")){

            if( dao.dirUpdate(this) > 0){
                messageBox(this, "Update Complete");
                dispose();
                dao.dirSelectAll(me.dt);
                if(me.dt.getRowCount() > 0 ) me.jt.setRowSelectionInterval(0, 0);

            }else if(  dao.dirCheck(txdirector_name.getText()) ){
                messageBox(this, "Duplicate record");
            }else{
                messageBox(this, "Update Failure");
            }

        }else if(btnLabel.equals("cancle")){
            dispose();
        }

    }//actionPerformed끝

    //MessageBox Constructor
    public static void messageBox(Object obj , String message){
        JOptionPane.showMessageDialog( (Component)obj , message);
    }

}//End of Class
```

```java
import java.awt.Color;

public class DirectorJTableExam extends JFrame implements ActionListener {
    JMenu m = new JMenu("Menu");
    JMenuItem insert = new JMenuItem("insert");
    JMenuItem update = new JMenuItem("update");
    JMenuItem delete = new JMenuItem("delete");
    JMenuItem quit = new JMenuItem("exit");
    JMenuBar mb = new JMenuBar();

    String[] name = { "name", "nationality", "year", "film" };

    DefaultTableModel dt = new DefaultTableModel(name, 0);
    JTable jt = new JTable(dt);
    JScrollPane jsp = new JScrollPane(jt);

    /*
     * South 영역에 추가할 Component들
     */
    JPanel p = new JPanel();
    String[] comboName = { "  ALL  ", "director_name", "director_nationality", "debut_year", "debut_film" };

    JComboBox combo = new JComboBox(comboName);
    JTextField jtf = new JTextField(20);
    JButton serach = new JButton("Search");

    DirectorDefaultJTableDAO dao = new DirectorDefaultJTableDAO();


   public DirectorJTableExam() {

        super("GUI movieInfo - DB");

        //메뉴아이템을 메뉴에 추가
        m.add(insert);
        m.add(update);
        m.add(delete);
        m.add(quit);
        //메뉴를 메뉴바에 추가
        mb.add(m);

        //윈도우에 메뉴바 세팅
        setJMenuBar(mb);

        // South영역
        p.setBackground(Color.yellow);
        p.add(combo);
        p.add(jtf);
        p.add(serach);

        add(jsp, "Center");
        add(p, "South");

        setSize(700, 500);
        setVisible(true);

        // 이벤트등록
        insert.addActionListener(this);
        update.addActionListener(this);
        delete.addActionListener(this);
        quit.addActionListener(this);
        serach.addActionListener(this);

        // 모든레코드를 검색하여 DefaultTableModle에 올리기
        dao.dirSelectAll(dt);
```

```
        //첫번행 선택.
        if (dt.getRowCount() > 0)
            jt.setRowSelectionInterval(0, 0);

    }// 생성자끝


    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == insert) {// 가입 메뉴아이템 클릭
            new DirectorJDailogGUI(this, "insert");

        } else if (e.getSource() == update) {// 수정 메뉴아이템 클릭
            new DirectorJDailogGUI(this, "update");

        } else if (e.getSource() == delete) {// 삭제 메뉴아이템 클릭
            // 현재 Jtable의 선택된 행과 열의 값을 얻어온다.
            int row = jt.getSelectedRow();
            System.out.println("선택행 : " + row);

            Object obj = jt.getValueAt(row, 0);// 행 열에 해당하는 value
            System.out.println("값 : " + obj);

            if (dao.dirDelete(obj.toString()) > 0) {
                DirectorJDailogGUI.messageBox(this, "Record Delete.");

                //리스트 갱신
                dao.dirSelectAll(dt);
                if (dt.getRowCount() > 0)
                    jt.setRowSelectionInterval(0, 0);

            } else {
                DirectorJDailogGUI.messageBox(this, "failed deleting record.");
            }

        } else if (e.getSource() == quit) {// 종료 메뉴아이템 클릭
            System.exit(0);


        } else if (e.getSource() == serach) {// 검색 버튼 클릭
            // JComboBox에 선택된 value 가져오기
            String fieldName = combo.getSelectedItem().toString();
            System.out.println("필드명 " + fieldName);

            if (fieldName.trim().equals("ALL")) {// 전체검색
                dao.dirSelectAll(dt);
                if (dt.getRowCount() > 0)
                    jt.setRowSelectionInterval(0, 0);
            } else {
                if (jtf.getText().trim().equals("")) {
                    DirectorJDailogGUI.messageBox(this, "write in blank");
                    jtf.requestFocus();
                } else {// 검색어를 입력했을경우
                    dao.getDirSearch(dt, fieldName, jtf.getText());
                    if (dt.getRowCount() > 0)
                        jt.setRowSelectionInterval(0, 0);
                }
            }
        }

    }//actionPerformed()----------

}
```

```java
import java.sql.Connection;

public class DistributorDefaultJTableDAO {


    Connection con;
    Statement st;
    PreparedStatement ps;
    ResultSet rs;

    /**
     * 로드 연결을 위한 생성자
     * @param movie_List
     * */
    public DistributorDefaultJTableDAO() {
        try {
            // 로드
            Class.forName("com.mysql.jdbc.Driver");

            con = DriverManager.getConnection("jdbc:mysql://localhost:3306/team01?useSSL=false","team01","team01");

        } catch (ClassNotFoundException e) {
            System.out.println(e + "=> load fail");
        } catch (SQLException e) {
            System.out.println(e + "=> connection fail");
        }
    }//생성자

    /**
     * DB닫기 기능 메소드
     * */
    public void dbClose() {
        try {
            if (rs != null) rs.close();
            if (st != null) st.close();
            if (ps != null) ps.close();
        } catch (Exception e) {

            System.out.println(e + "=> dbClose fail");
        }
    }//dbClose() ---

    /**
     * 인수로 들어온 ID에 해당하는 레코드 검색하여 중복여부 체크하기 리턴값이 true =사용가능 , false = 중복임
     * */
    public boolean disCheck(String distributor) {
        boolean result = true;

        try {
            ps = con.prepareStatement("SELECT * FROM DBCOURSE_DISTRIBUTOR WHERE distributor=?");
            ps.setString(1, distributor.trim());
            rs = ps.executeQuery(); //실행
            if (rs.next())
                result = false; //레코드가 존재하면 false

        } catch (SQLException e) {
            System.out.println(e + "=>  getIdByCheck fail");
        } finally {
            dbClose();
        }

        return result;

    }//disCheck()


    public int disListInsert(DistributorJDailogGUI user) {
        int result = 0;
        try {
            ps = con.prepareStatement("insert into DBCOURSE_DISTRIBUTOR values(?,?)");
            ps.setString(1, user.txdistributor.getText());
            ps.setString(2, user.txdceo.getText());

            result = ps.executeUpdate(); //실행 -> 저장
```

```
            result = ps.executeUpdate(); //실행 -> 저장

        } catch (SQLException e) {
            System.out.println(e + "=> userListInsert fail");
        } finally {
            dbClose();
        }

        return result;

    }//disListInsert()
    //find all record from distributorDB
    public void disSelectAll(DefaultTableModel t_model) {
        try {
            st = con.createStatement();
            rs = st.executeQuery("select * from DBCOURSE_DISTRIBUTOR order by distributor");

            // DefaultTableModel에 있는 기존 데이터 지우기
            for (int i = 0; i < t_model.getRowCount();) {
                t_model.removeRow(0);
            }

            while (rs.next()) {
                Object data[] = { rs.getString(1), rs.getString(2) };

                t_model.addRow(data); //DefaultTableModel에 레코드 추가
            }

        } catch (SQLException e) {
            System.out.println(e + "=> userSelectAll fail");
        } finally {
            dbClose();
        }
    }//disSelectAll()


    public int disDelete(String distributor) {
        int result = 0;
        try {
            ps = con.prepareStatement("delete from DBCOURSE_DISTRIBUTOR  where distributor = ? ");
            ps.setString(1, distributor.trim());
            result = ps.executeUpdate();

        } catch (SQLException e) {
            System.out.println(e + "=> userDelete fail");
        }finally {
            dbClose();
        }

        return result;
    }//userDelete()

    //update DB
    public int disUpdate(DistributorJDailogGUI user) {
        int result = 0;
        String sql = "UPDATE DBCOURSE_DISTRIBUTOR SET distributor=?, dceo=?";

        try {
            ps = con.prepareStatement(sql);
            // ?의 순서대로 값 넣기
            ps.setString(1, user.txdistributor.getText());
            ps.setString(2, user.txdceo.getText());

            // 실행하기
            result = ps.executeUpdate();

        } catch (SQLException e) {
            System.out.println(e + "=> Update fail");
        } finally {
            dbClose();
        }

        return result;
    }//disUpdate()
```

```java
    /**
     * 검색단어에 해당하는 레코드 검색하기 (like연산자를 사용하여 _, %를 사용할때는 PreparedStatemnet안된다. 반드시
     * Statement객체를 이용함)
     * */
    public void disSearch(DefaultTableModel dt, String fieldName,
            String word) {
        String sql = "SELECT * FROM DBCOURSE_DISTRIBUTOR WHERE " + fieldName.trim()
                + " LIKE '%" + word.trim() + "%'";

        try {
            st = con.createStatement();
            rs = st.executeQuery(sql);

            // DefaultTableModel에 있는 기존 데이터 지우기
            for (int i = 0; i < dt.getRowCount();) {
                dt.removeRow(0);
            }

            while (rs.next()) {
                Object data[] = { rs.getString(1), rs.getString(2) };

                dt.addRow(data);
            }

        } catch (SQLException e) {
            System.out.println(e + "=> getUserSearch fail");
        } finally {
            dbClose();
        }

    }//disSearch()

}//End of Class
```

```java
import java.awt.BorderLayout;

public class DistributorJDailogGUI extends JDialog implements ActionListener{

    JPanel pw=new JPanel(new GridLayout(5,1));
    JPanel pc=new JPanel(new GridLayout(5,1));
    JPanel ps=new JPanel();

    JLabel lable_distributor = new JLabel("Distributor");
    JLabel lable_dceo=new JLabel("re");

    JTextField txdistributor=new JTextField();
    JTextField txdceo=new JTextField();

    JButton confirm;
    JButton reset=new JButton("cancle");

   DistributorJTableExam me;

   DistributorDefaultJTableDAO dao =new DistributorDefaultJTableDAO();


    public DistributorJDailogGUI(DistributorJTableExam me, String index){
        super(me,"Distributor Table");
        this.me=me;
        if(index.equals("insert")){
            confirm=new JButton(index);
        }else{
            confirm=new JButton("update");

            //textbox
            int row = me.jt.getSelectedRow();//선택된 행
            txdistributor.setText( me.jt.getValueAt(row, 0).toString() );
            txdceo.setText( me.jt.getValueAt(row, 1).toString() );

            //id text박스 비활성
            txdistributor.setEditable(false);
```

```
        }

        //Label추가부분
        pw.add(lable_distributor);//배급사
        pw.add(lable_dceo);//대표
        //TextField 추가
        pc.add(txdistributor);
        pc.add(txdceo);


        ps.add(confirm); //update
        ps.add(reset);//cancle

        add(pw,"West");
        add(pc,"Center");
        add(ps,"South");

        setSize(500,350);
        setVisible(true);

        setDefaultCloseOperation(JDialog.DISPOSE_ON_CLOSE);

        //이벤트등록
        confirm.addActionListener(this); //insert/update 이벤트등록
        reset.addActionListener(this); //취소 이벤트등록

    }//생성자끝


    @Override
    public void actionPerformed(ActionEvent e) {
       String btnLabel =e.getActionCommand();//get event lable

       if(btnLabel.equals("insert")){
           if(dao.disListInsert(this) > 0 ){//can insert
               messageBox(this , txdistributor.getText()+"Insert Complete");
               dispose();//JDialog exit

               dao.disSelectAll(me.dt);//get all records to DefaultTableModel

               if(me.dt.getRowCount() > 0)
                   me.jt.setRowSelectionInterval(0, 0);//select first row

           }else{//insert fail
               messageBox(this,"insert failure");
           }

       }else if(btnLabel.equals("update")){

            if( dao.disUpdate(this) > 0){
                messageBox(this, "Update complete");
                dispose();
                dao.disSelectAll(me.dt);
                if(me.dt.getRowCount() > 0 ) me.jt.setRowSelectionInterval(0, 0);

            }else if(  dao.disCheck(txdistributor.getText()) ){
                messageBox(this, "Update Failure : duplicate record");
            }else {
                messageBox(this, "Update Failure");
            }

       }else if(btnLabel.equals("cancle")){
           dispose();

       }


    //Message Box construct
    public static void messageBox(Object obj , String message){
        JOptionPane.showMessageDialog( (Component)obj , message);
    }

}//End of Class
```

```java
import java.awt.Color;

public class DistributorJTableExam extends JFrame implements ActionListener {
    JMenu m = new JMenu("Menu");
    JMenuItem insert = new JMenuItem("insert");
    JMenuItem update = new JMenuItem("update");
    JMenuItem delete = new JMenuItem("delete");
    JMenuItem quit = new JMenuItem("exit");
    JMenuBar mb = new JMenuBar();

    String[] name = { "Distributor_name", "representative" };

    DefaultTableModel dt = new DefaultTableModel(name, 0);
    JTable jt = new JTable(dt);
    JScrollPane jsp = new JScrollPane(jt);

    /*
     * South 영역에 추가할 Component들
     */
    JPanel p = new JPanel();
    String[] comboName = { "  ALL  ", "Distributor_name", "dceo" };

    JComboBox combo = new JComboBox(comboName);
    JTextField jtf = new JTextField(20);
    JButton serach = new JButton("Search");

    DistributorDefaultJTableDAO dao = new DistributorDefaultJTableDAO();


   public DistributorJTableExam() {

       super("GUI movieInfo - DB");

       //메뉴아이템을 메뉴에 추가
       m.add(insert);
       m.add(update);
       m.add(delete);
       m.add(quit);
       //메뉴를 메뉴바에 추가
       mb.add(m);

       //윈도우에 메뉴바 세팅
       setJMenuBar(mb);

       // South영역
       p.setBackground(Color.yellow);
       p.add(combo);
       p.add(jtf);
       p.add(serach);

       add(jsp, "Center");
       add(p, "South");

       setSize(700, 500);
       setVisible(true);

       // 이벤트등록
       insert.addActionListener(this);
       update.addActionListener(this);
       delete.addActionListener(this);
       quit.addActionListener(this);
       serach.addActionListener(this);

       // 모든레코드를 검색하여 DefaultTableModle에 올리기
       dao.disSelectAll(dt);
```

```
        //첫번행 선택.
        if (dt.getRowCount() > 0)
            jt.setRowSelectionInterval(0, 0);

    }//End of Constructor

    //insert/update/delete
    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == insert) {// 가입 메뉴아이템 클릭
            new DistributorJDailogGUI(this, "insert");

        } else if (e.getSource() == update) {// 수정 메뉴아이템 클릭
            new DistributorJDailogGUI(this, "update");

        } else if (e.getSource() == delete) {// 삭제 메뉴아이템 클릭
            // 현재 Jtable의 선택된 행과 열의 값을 얻어온다.
            int row = jt.getSelectedRow();
            System.out.println("선택행 : " + row);

            Object obj = jt.getValueAt(row, 0);// 행 열에 해당하는 value
            System.out.println("값 : " + obj);

            if (dao.disDelete(obj.toString()) > 0) {
                DistributorJDailogGUI.messageBox(this, "Record Delete.");

                //리스트 갱신
                dao.disSelectAll(dt);
                if (dt.getRowCount() > 0)
                    jt.setRowSelectionInterval(0, 0);

            } else {
                DistributorJDailogGUI.messageBox(this, "failed deleting record.");
            }

        } else if (e.getSource() == quit) {// 종료 메뉴아이템 클릭
            System.exit(0);

        } else if (e.getSource() == serach) {// 검색 버튼 클릭
            // JComboBox에 선택된 value 가져오기
            String fieldName = combo.getSelectedItem().toString();
            System.out.println("필드명 " + fieldName);

            if (fieldName.trim().equals("ALL")) {// 전체검색
                dao.disSelectAll(dt);
                if (dt.getRowCount() > 0)
                    jt.setRowSelectionInterval(0, 0);
            } else {
                if (jtf.getText().trim().equals("")) {
                    DistributorJDailogGUI.messageBox(this, "검색단어를 입력해주세요!");
                    jtf.requestFocus();
                } else {// 검색어를 입력했을경우
                    dao.disSearch(dt, fieldName, jtf.getText());
                    if (dt.getRowCount() > 0)
                        jt.setRowSelectionInterval(0, 0);
                }
            }
        }

    }//actionPerformed()----------
```

```java
import java.sql.Connection;


public class MovieDefaultJTableDAO {

    /**
     * 필요한 변수선언
     * */
    Connection con;
    Statement st;
    PreparedStatement ps;
    ResultSet rs;

    /**
     * 로드 연결을 위한 생성자
     * @param movie_List
     * */
    public MovieDefaultJTableDAO() {
        try {
            // 로드
            Class.forName("com.mysql.jdbc.Driver");

            con = DriverManager.getConnection("jdbc:mysql://localhost:3306/team01?useSSL=false","team01","team01");

        } catch (ClassNotFoundException e) {
            System.out.println(e + "=> load fail");
        } catch (SQLException e) {
            System.out.println(e + "=> connection fail");
        }
    }//생성자


    public void dbClose() {
        try {
            if (rs != null) rs.close();
            if (st != null) st.close();
            if (ps != null) ps.close();
        } catch (Exception e) {
            System.out.println(e + "=> dbClose fail");
        }
    }//dbClose() ---



    public int userListInsert(MovieJDailogGUI user) {
        int result = 0;
        try {
            ps = con.prepareStatement("insert into DBCOURSE_MOVIE values(?,?,?,?,?,?,?)");
            ps.setString(1, user.movie.getText());
            ps.setString(2, user.nation.getText());
            ps.setString(3, user.genre.getText());
            ps.setInt(4, Integer.parseInt(user.runtime.getText()));
            ps.setString(5, user.director.getText());
            ps.setString(6, user.distributor.getText());
            ps.setString(7, user.production_company.getText());

            result = ps.executeUpdate(); //실행 -> 저장

        } catch (SQLException e) {
            System.out.println(e + "=> userListInsert fail");
        } finally {
            dbClose();
        }

        return result;

    }//userListInsert()
```

```
public void SelectAllMovie(DefaultTableModel t_model) {
    try {
        st = con.createStatement();
        rs = st.executeQuery("select * from DBCOURSE_MOVIE order by movie");

        // DefaultTableModel에 있는 기존 데이터 지우기
        for (int i = 0; i < t_model.getRowCount();) {
            t_model.removeRow(0);
        }

        while (rs.next()) {
            Object data[] = { rs.getString(1), rs.getString(2),rs.getString(3),
                    rs.getInt(4), rs.getString(5), rs.getString(6), rs.getString(7) };

            t_model.addRow(data); //DefaultTableModel에 레코드 추가
        }

    } catch (SQLException e) {
        System.out.println(e + "=> movieSelectAll fail");
    } finally {
        dbClose();
    }
}//userSelectAll()

/**
 * ID에 해당하는 레코드 삭제하기
 * */
public int movieDelete(String movie) {
    int result = 0;
    try {
        ps = con.prepareStatement("delete from DBCOURSE_MOVIE  where movie = ? ");
        ps.setString(1, movie.trim());
        result = ps.executeUpdate();

    } catch (SQLException e) {
        System.out.println(e + "=> movieDelete fail");
    }finally {
        dbClose();
    }

    return result;
}//userDelete()

/**
 * ID에 해당하는 레코드 수정하기
 * */
public int movieUpdate(MovieJDailogGUI user) {
    int result = 0;
    String sql = "UPDATE DBCOURSE_MOVIE SET nation=?, genre=?, runtime=?, director=?, distributor=?, "+
    "production_company=? WHERE movie=?";

    try {

        ps = con.prepareStatement(sql);
        // ?의 순서대로 값 넣기
        ps.setString(7, user.movie.getText());
        ps.setString(1, user.nation.getText());
        ps.setString(2, user.genre.getText());
        ps.setInt(3, Integer.parseInt(user.runtime.getText().trim()));
        ps.setString(4, user.director.getText());
        ps.setString(5, user.distributor.getText());
        ps.setString(6, user.production_company.getText());

        // 실행하기
        result = ps.executeUpdate();

    } catch (SQLException e) {
        System.out.println(e + "=> Update fail");
    } finally {
            dbClose();

    }

    return result;
}//userUpdate()
```

```
    /**
     * 검색단어에 해당하는 레코드 검색하기 (like연산자를 사용하여 _, %를 사용할때는 PreparedStatemnet안된다. 반드시
     * Statement객체를 이용함)
     * */
    public void getMovieSearch(DefaultTableModel dt, String fieldName,
            String word) {
        String sql = "SELECT * FROM DBCOURSE_MOVIE WHERE " + fieldName.trim()
                + " LIKE '%" + word.trim() + "%'";

        try {
            st = con.createStatement();
            rs = st.executeQuery(sql);

            // DefaultTableModel에 있는 기존 데이터 지우기
            for (int i = 0; i < dt.getRowCount();) {
                dt.removeRow(0);
            }

            while (rs.next()) {
                Object data[] = { rs.getString(1), rs.getString(2),rs.getString(3),
                        rs.getInt(4), rs.getString(5), rs.getString(6), rs.getString(7) };

                dt.addRow(data);
            }

        } catch (SQLException e) {
            System.out.println(e + " getMovieSearch fail");
        } finally {
            dbClose();
        }

    }//movieSearch()


}//End of Class
```

```
import java.awt.BorderLayout;

public class MovieJDailogGUI extends JDialog implements ActionListener{

    JPanel pw=new JPanel(new GridLayout(7,1));
    JPanel pc=new JPanel(new GridLayout(7,1));
    JPanel ps=new JPanel();

    JLabel lable_Movie = new JLabel("Movie");
    JLabel lable_Nation=new JLabel("Nation");
    JLabel lable_Gen=new JLabel("Genre");
    JLabel lable_Run=new JLabel("Runtime");
    JLabel lable_Dir=new JLabel("Director");
    JLabel lable_Dis=new JLabel("Distributor");
    JLabel lable_Pro=new JLabel("Production");


    JTextField movie=new JTextField();
    JTextField nation=new JTextField();
    JTextField genre=new JTextField();
    JTextField runtime=new JTextField();
    JTextField director = new JTextField();
    JTextField distributor = new JTextField();
    JTextField production_company = new JTextField();


    JButton confirm;
    JButton reset=new JButton("cancle");

   MovieJTabaleExam me;

   MovieDefaultJTableDAO dao =new MovieDefaultJTableDAO();


    public MovieJDailogGUI(MovieJTabaleExam me, String index){
        super(me,"Movie Table");
        this.me=me;
```

```
        if(index.equals("insert")){
            confirm=new JButton(index);
        }else{
            confirm=new JButton("update");

            //get text from record
            int row = me.jt.getSelectedRow();//selected row
            movie.setText( me.jt.getValueAt(row, 0).toString() );
            nation.setText( me.jt.getValueAt(row, 1).toString() );
            genre.setText( me.jt.getValueAt(row, 2).toString() );
            runtime.setText( me.jt.getValueAt(row, 3).toString() );
            director.setText( me.jt.getValueAt(row, 4).toString() );
            distributor.setText( me.jt.getValueAt(row, 5).toString() );
            production_company.setText( me.jt.getValueAt(row, 6).toString() );

            //deactivate movie textbox
            movie.setEditable(false);

        }


        //Label
        pw.add(lable_Movie);//movie
        pw.add(lable_Nation);//nation
        pw.add(lable_Gen);//genre
        pw.add(lable_Run);//runtime
        pw.add(lable_Dir);//director
        pw.add(lable_Dis);//distributor
        pw.add(lable_Pro);//productioncomp


        //TextField
        pc.add(movie);
        pc.add(nation);
        pc.add(genre);
        pc.add(runtime);
        pc.add(director);
        pc.add(distributor);
        pc.add(production_company);

        ps.add(confirm); //update
        ps.add(reset);//cancle

        add(pw,"West");
        add(pc,"Center");
        add(ps,"South");

        setSize(700,450);
        setVisible(true);

        setDefaultCloseOperation(JDialog.DISPOSE_ON_CLOSE);

        //event update
        confirm.addActionListener(this); //insert/update
        reset.addActionListener(this); //cancle event


    }//Constructor

    /**
     * insert/update/delete
     * */
    @Override
    public void actionPerformed(ActionEvent e) {
       String btnLabel =e.getActionCommand();//get event label

       if(btnLabel.equals("insert")){
            if(dao.userListInsert(this) > 0 ){//complete insert
                messageBox(this , movie.getText()+"insert complete");
                dispose();//JDialog 창닫기

                dao.SelectAllMovie(me.dt);//get all record from DefaultTable

                if(me.dt.getRowCount() > 0)
                    me.jt.setRowSelectionInterval(0, 0);//get first row
```

```
            }else{//insert fail
                messageBox(this,"fail inserting.");
            }

        }else if(btnLabel.equals("update")){

             if( dao.movieUpdate(this) > 0){
                 messageBox(this, "success updating.");
                 dispose();
                 dao.SelectAllMovie(me.dt);
                 if(me.dt.getRowCount() > 0 ) me.jt.setRowSelectionInterval(0, 0);

             }else{
                 messageBox(this, "fail updating.");
             }



        }else if(btnLabel.equals("cancle")){
            dispose();

        }


    }//actionPerformed

    /**
     * Message box
     * */
    public static void messageBox(Object obj , String message){
        JOptionPane.showMessageDialog( (Component)obj , message);
    }

}//End MovieJDailogGUI class
```

```
import java.awt.Color;

public class MovieJTabaleExam extends JFrame implements ActionListener {
    JMenu m = new JMenu("관리");
    JMenuItem insert = new JMenuItem("insert");
    JMenuItem update = new JMenuItem("update");
    JMenuItem delete = new JMenuItem("delete");
    JMenuItem quit = new JMenuItem("exit");
    JMenuBar mb = new JMenuBar();

    String[] name = { "Movie", "Nation", "Genre","Runtime", "Director","Distributor","ProductionCompany"};

    DefaultTableModel dt = new DefaultTableModel(name, 0);
    JTable jt = new JTable(dt);
    JScrollPane jsp = new JScrollPane(jt);

    /*
     * South 영역에 추가할 Componet들
     */
    JPanel p = new JPanel();
    String[] comboName = { "  ALL  ", "  movie  ","  nation  ", " genre ", " runtime ", " director ",
            " distributor "," production "};

    JComboBox combo = new JComboBox(comboName);
    JTextField jtf = new JTextField(20);
    JButton serach = new JButton("Search");

    MovieDefaultJTableDAO dao = new MovieDefaultJTableDAO();

    /**
     * 화면구성 및 이벤트등록
     */
    public MovieJTabaleExam() {

        super("GUI movieInfo - DB");

        //메뉴아이템을 메뉴에 추가
```

```
        m.add(insert);
        m.add(update);
        m.add(delete);
        m.add(quit);
        //메뉴를 메뉴바에 추가
        mb.add(m);

        //윈도우에 메뉴바 세팅
        setJMenuBar(mb);


        // South영역
        p.setBackground(Color.yellow);
        p.add(combo);
        p.add(jtf);
        p.add(serach);

        add(jsp, "Center");
        add(p, "South");

        setSize(800, 500);
        setVisible(true);


        // 이벤트등록
        insert.addActionListener(this);
        update.addActionListener(this);
        delete.addActionListener(this);
        quit.addActionListener(this);
        serach.addActionListener(this);

        // 모든레코드를 검색하여 DefaultTableModle에 올리기
        dao.SelectAllMovie(dt);

        //첫번행 선택.
        if (dt.getRowCount() > 0)
            jt.setRowSelectionInterval(0, 0);


    }//Constructor


    /**
     * insert/update/delete/search method
     * */

    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == insert) {// 가입 메뉴아이템 클릭
            new MovieJDailogGUI(this, "insert");

        } else if (e.getSource() == update) {// 수정 메뉴아이템 클릭
            new MovieJDailogGUI(this, "update");

        } else if (e.getSource() == delete) {// 삭제 메뉴아이템 클릭
            // 현재 Jtable의 선택된 행과 열의 값을 얻어온다.
            int row = jt.getSelectedRow();
            System.out.println("선택행 : " + row);

            Object obj = jt.getValueAt(row, 0);// 행 열에 해당하는 value
            System.out.println("값 : " + obj);

            if (dao.movieDelete(obj.toString()) > 0) {
                MovieJDailogGUI.messageBox(this, "Record Delete.");

                //리스트 갱신
                dao.SelectAllMovie(dt);
                if (dt.getRowCount() > 0)
                    jt.setRowSelectionInterval(0, 0);

            } else {
                MovieJDailogGUI.messageBox(this, "failed deleting record.");
            }

        } else if (e.getSource() == quit) {// 종료 메뉴아이템 클릭
            System.exit(0);
```

```
        } else if (e.getSource() == serach) {// 검색 버튼 클릭
            // JComboBox에 선택된 value 가져오기
            String fieldName = combo.getSelectedItem().toString();
            System.out.println("필드명 " + fieldName);

            if (fieldName.trim().equals("ALL")) {// 전체검색
                dao.SelectAllMovie(dt);
                if (dt.getRowCount() > 0)
                    jt.setRowSelectionInterval(0, 0);
            } else {
                if (jtf.getText().trim().equals("")) {
                    MovieJDailogGUI.messageBox(this, "검색단어를 입력해주세요!");
                    jtf.requestFocus();
                } else {// 검색어를 입력했을경우
                    dao.getMovieSearch(dt, fieldName, jtf.getText());
                    if (dt.getRowCount() > 0)
                        jt.setRowSelectionInterval(0, 0);
                }
            }
        }

    }//actionPerformed()----------

}
```

```
import java.sql.Connection;

public class MusicDefaultJTableDAO {

    /**
     * 필요한 변수선언
     * */
    Connection con;
    Statement st;
    PreparedStatement ps;
    ResultSet rs;

    /**
     * 로드 연결을 위한 생성자
     * */
    public MusicDefaultJTableDAO() {
        try {
            // 로드
            Class.forName("com.mysql.jdbc.Driver");

            con = DriverManager.getConnection("jdbc:mysql://localhost:3306/team01?useSSL=false","team01","team01");

        } catch (ClassNotFoundException e) {
            System.out.println(e + "=> load fail");
        } catch (SQLException e) {
            System.out.println(e + "=> connection fail");
        }
    }//생성자

    /**
     * DB닫기 기능 메소드
     * */
    public void dbClose() {
        try {
            if (rs != null) rs.close();
            if (st != null) st.close();
            if (ps != null) ps.close();
```

```
        } catch (Exception e) {
            System.out.println(e + "=> dbClose fail");
        }
    }//dbClose() ---


    //insert/update/delete/search
    public int musicListInsert(MusicJDailogGUI user) {
        int result = 0;
        try {
            ps = con.prepareStatement("insert into DBCOURSE_MUSIC values(?,?,?)");
            ps.setString(1, user.Movie.getText());
            ps.setString(2, user.Music.getText());
            ps.setString(3, user.Music_Director.getText());


            result = ps.executeUpdate(); //실행 -> 저장

        } catch (SQLException e) {
            System.out.println(e + "=> userListInsert fail");
        } finally {
            dbClose();
        }

        return result;

    }//musicListInsert()

    //select all music
    public void musicSelectAll(DefaultTableModel t_model) {
        try {
            st = con.createStatement();
            rs = st.executeQuery("select * from DBCOURSE_MUSIC order by movie");

            // DefaultTableModel에 있는 기존 데이터 지우기
            for (int i = 0; i < t_model.getRowCount();) {
                t_model.removeRow(0);
            }

            while (rs.next()) {
                Object data[] = { rs.getString(1), rs.getString(2),rs.getString(3) };

                t_model.addRow(data); //DefaultTableModel에 레코드 추가
            }

        } catch (SQLException e) {
            System.out.println(e + "=> userSelectAll fail");
        } finally {
            dbClose();
        }
    }//userSelectAll()

    /**
     * ID에 해당하는 레코드 삭제하기
     * */
    public int musicDelete(String Movie) {
        int result = 0;
        try {
            ps = con.prepareStatement("delete from DBCOURSE_MUSIC  where movie = ? ");
            ps.setString(1, Movie.trim());
            result = ps.executeUpdate();

        } catch (SQLException e) {
            System.out.println(e + "=> userDelete fail");
        }finally {
            dbClose();
        }

        return result;
    }//musicDelete()

    /**
     * ID에 해당하는 레코드 수정하기
     * */
    public int musicUpdate(MusicJDailogGUI user) {
        int result = 0;
        String sql = "UPDATE DBCOURSE_Music SET movie=?, music=?, music_director=? WHERE movie=?";
```

```
        try {
            ps = con.prepareStatement(sql);
            // ?의 순서대로 값 넣기
            ps.setString(3, user.Movie.getText());
            ps.setString(1, user.Music.getText());
            ps.setString(2, user.Music_Director.getText());


            // 실행하기
            result = ps.executeUpdate();

        } catch (SQLException e) {
            System.out.println(e + "=> Update fail");
        } finally {
            dbClose();
        }

        return result;
    }//musicUpdate()

    /**
     * 검색단어에 해당하는 레코드 검색하기 (like연산자를 사용하여 _, %를 사용할때는 PreparedStatemnet안된다. 반드시
     * Statement객체를 이용함)
     * */
    public void musicSearch(DefaultTableModel dt, String fieldName,
            String word) {
        String sql = "SELECT * FROM DBCOURSE_MUSIC WHERE " + fieldName.trim()
                + " LIKE '%" + word.trim() + "%'";

        try {
            st = con.createStatement();
            rs = st.executeQuery(sql);

            // DefaultTableModel에 있는 기존 데이터 지우기
            for (int i = 0; i < dt.getRowCount();) {
                dt.removeRow(0);
            }


            while (rs.next()) {
                Object data[] = { rs.getString(1), rs.getString(2),rs.getString(3)};

                dt.addRow(data);
            }

        } catch (SQLException e) {
            System.out.println(e + "=> getUserSearch fail");
        } finally {
            dbClose();
        }

    }//getUserSearch()

}// 클래스끝
```

```java
import java.awt.BorderLayout;

public class MusicJDailogGUI extends JDialog implements ActionListener{

    JPanel pw=new JPanel(new GridLayout(3,1));
    JPanel pc=new JPanel(new GridLayout(3,1));
    JPanel ps=new JPanel();

    JLabel lable_Movie = new JLabel("Movie");
    JLabel lable_Music=new JLabel("Music");
    JLabel lable_Music_Director=new JLabel("Music Director");


    JTextField Movie = new JTextField();
    JTextField Music=new JTextField();
    JTextField Music_Director=new JTextField();


    JButton confirm;
    JButton reset=new JButton("cancle");

   MusicJTabaleExam me;

   MusicDefaultJTableDAO dao =new MusicDefaultJTableDAO();


    public MusicJDailogGUI(MusicJTabaleExam me, String index){
        super(me,"MovieOST Table");
        this.me=me;
        if(index.equals("insert")){
            confirm=new JButton(index);
        }else{
            confirm=new JButton("update");

            //text박스에 선택된 레코드의 정보 넣기
            int row = me.jt.getSelectedRow();//선택된 행
            Movie.setText( me.jt.getValueAt(row, 0).toString() );

          //moviename text박스 비활성
          Movie.setEditable(false);

       }

       //Label추가부분
       pw.add(lable_Movie);//moviename
       pw.add(lable_Music);//music
       pw.add(lable_Music_Director);//music director

       //TextField 추가
       pc.add(Movie);
       pc.add(Music);
       pc.add(Music_Director);


       ps.add(confirm); //update
       ps.add(reset);//cancle

       add(pw,"West");
       add(pc,"Center");
       add(ps,"South");

       setSize(500,350);
       setVisible(true);

       setDefaultCloseOperation(JDialog.DISPOSE_ON_CLOSE);

       //이벤트등록
       confirm.addActionListener(this); //가입/수정 이벤트등록
       reset.addActionListener(this); //취소 이벤트등록

   }//생성자끝
```

```
    //insert/update/delete
    @Override
    public void actionPerformed(ActionEvent e) {
       String btnLabel =e.getActionCommand();//이벤트주체 대한 Label 가져오기

       if(btnLabel.equals("insert")){
           if(dao.musicListInsert(this) > 0 ){//가입성공
               messageBox(this , Movie.getText()+" insert success");
               dispose();//JDialog 창닫기

               dao.musicSelectAll(me.dt);//모든레코드가져와서 DefaultTableModel에 올리기

               if(me.dt.getRowCount() > 0)
                   me.jt.setRowSelectionInterval(0, 0);//select first row

           }else{//insert fail
               messageBox(this,"insert Failure");
           }

       }else if(btnLabel.equals("update")){

            if( dao.musicUpdate(this) > 0){
                messageBox(this, "upadate complete");
                dispose();
                dao.musicSelectAll(me.dt);
                if(me.dt.getRowCount() > 0 ) me.jt.setRowSelectionInterval(0, 0);

            }else{
                messageBox(this, "update failure");
            }

       }else if(btnLabel.equals("cancle")){
           dispose();

       }

    }//actionPerformed끝


     /**
      * MessageBox Constructor
      * */
     public static void messageBox(Object obj , String message){
         JOptionPane.showMessageDialog( (Component)obj , message);
     }

}//End of Class
```

```
import java.awt.Color;


public class MusicJTabaleExam extends JFrame implements ActionListener {
    JMenu m = new JMenu("Menu");
    JMenuItem insert = new JMenuItem("insert");
    JMenuItem update = new JMenuItem("update");
    JMenuItem delete = new JMenuItem("delete");
    JMenuItem quit = new JMenuItem("exit");
    JMenuBar mb = new JMenuBar();

    String[] name = { "Movie", "Music", "Music_Director" };

    DefaultTableModel dt = new DefaultTableModel(name, 0);
    JTable jt = new JTable(dt);
    JScrollPane jsp = new JScrollPane(jt);

    /*
     * South 영역에 추가할 Component들
     */
    JPanel p = new JPanel();
    String[] comboName = { " ALL ", " Movie ", " Music ", " Music_Director " };

    JComboBox combo = new JComboBox(comboName);
    JTextField jtf = new JTextField(20);
    JButton serach = new JButton("Search");

    MusicDefaultJTableDAO dao = new MusicDefaultJTableDAO();

```

```java
    /**
     * 화면구성 및 이벤트등록
     */
    public MusicJTabaleExam() {

        super("GUI MusicInfo - DB");

        //메뉴아이템을 메뉴에 추가
        m.add(insert);
        m.add(update);
        m.add(delete);
        m.add(quit);
        //메뉴를 메뉴바에 추가
        mb.add(m);

        //윈도우에 메뉴바 세팅
        setJMenuBar(mb);

        // South영역
        p.setBackground(Color.yellow);
        p.add(combo);
        p.add(jtf);
        p.add(serach);

        add(jsp, "Center");
        add(p, "South");

        setSize(700, 500);
        setVisible(true);

        // 이벤트등록
        insert.addActionListener(this);
        update.addActionListener(this);
        delete.addActionListener(this);
        quit.addActionListener(this);
        serach.addActionListener(this);


          // 모든레코드를 검색하여 DefaultTableModle에 올리기
          dao.musicSelectAll(dt);

          //첫번행 선택.
          if (dt.getRowCount() > 0)
              jt.setRowSelectionInterval(0, 0);

    }// 생성자끝

    /**
     * 가입/수정/삭제/검색기능을 담당하는 메소드
     * */

    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == insert) {// 가입 메뉴아이템 클릭
            new MusicJDailogGUI(this, "insert");

        } else if (e.getSource() == update) {// 수정 메뉴아이템 클릭
            new MusicJDailogGUI(this, "update");

        } else if (e.getSource() == delete) {// 삭제 메뉴아이템 클릭
            // 현재 Jtable의 선택된 행과 열의 값을 얻어온다.
            int row = jt.getSelectedRow();
            System.out.println("선택행 : " + row);

            Object obj = jt.getValueAt(row, 0);// 행 열에 해당하는 value
            System.out.println("값 : " + obj);

            if (dao.musicDelete(obj.toString()) > 0) {
                MusicJDailogGUI.messageBox(this, "Record Delete.");

                //리스트 갱신
                dao.musicSelectAll(dt);
                if (dt.getRowCount() > 0)
                    jt.setRowSelectionInterval(0, 0);
```

```
            } else {
                MusicJDailogGUI.messageBox(this, "failed deleting record.");
            }

        } else if (e.getSource() == quit) {// 종료 메뉴아이템 클릭
            System.exit(0);

        } else if (e.getSource() == serach) {// 검색 버튼 클릭
            // JComboBox에 선택된 value 가져오기
            String fieldName = combo.getSelectedItem().toString();
            System.out.println("필드명 " + fieldName);

            if (fieldName.trim().equals("ALL")) {// 전체검색
                dao.musicSelectAll(dt);
                if (dt.getRowCount() > 0)
                    jt.setRowSelectionInterval(0, 0);
            } else {
                if (jtf.getText().trim().equals("")) {
                    MusicJDailogGUI.messageBox(this, "검색단어를 입력해주세요!");
                    jtf.requestFocus();
                } else {// 검색어를 입력했을경우
                    dao.musicSearch(dt, fieldName, jtf.getText());
                    if (dt.getRowCount() > 0)
                        jt.setRowSelectionInterval(0, 0);
                }
            }
        }

    }//actionPerformed()----------

}
```

```
import java.sql.Connection;

public class ProductionDefaultJTableDAO {

    /**
     * 필요한 변수선언
     * */
    Connection con;
    Statement st;
    PreparedStatement ps;
    ResultSet rs;

    /**
     * 로드 연결을 위한 생성자
     * @param movie_List
     * */
    public ProductionDefaultJTableDAO() {
        try {
            // 로드
            Class.forName("com.mysql.jdbc.Driver");

            con = DriverManager.getConnection("jdbc:mysql://localhost:3306/team01?useSSL=false","team01","team01");

        } catch (ClassNotFoundException e) {
            System.out.println(e + "=> load fail");
        } catch (SQLException e) {
            System.out.println(e + "=> connection fail");
        }
    }//생성자

    /**
     * DB닫기 기능 메소드
     * */
    public void dbClose() {
        try {
            if (rs != null) rs.close();
            if (st != null) st.close();
```

```
            if (ps != null) ps.close();
        } catch (Exception e) {
            System.out.println(e + "=> dbClose fail");
        }
    }//dbClose() ---

    /**
     * 인수로 들어온 ID에 해당하는 레코드 검색하여 중복여부 체크하기 리턴값이 true =사용가능 , false = 중복임
     * */
    public boolean proCheck(String production_name) {
        boolean result = true;

        try {
            ps = con.prepareStatement("SELECT * FROM DBCOURSE_PRODUCTIONCOMPANY WHERE production_name=?");
            ps.setString(1, production_name.trim());
            rs = ps.executeQuery(); //실행
            if (rs.next())
                result = false; //레코드가 존재하면 false

        } catch (SQLException e) {
            System.out.println(e + "=>  getIdByCheck fail");
        } finally {
            dbClose();
        }

        return result;

    }//proCheck()

    public int proListInsert(ProductionJDailogGUI user) {
        int result = 0;
        try {
            ps = con.prepareStatement("insert into DBCOURSE_PRODUCTIONCOMPANY values(?,?)");
            ps.setString(1, user.txproduction_name.getText());
            ps.setString(2, user.txpceo.getText());

            result = ps.executeUpdate(); //실행 -> 저장

        } catch (SQLException e) {
            System.out.println(e + "=> userListInsert fail");
        } finally {
            dbClose();
        }

        return result;

    }//proListInsert()

    /**
     * userlist의 모든 레코드 조회
     * */
    public void proSelectAll(DefaultTableModel t_model) {
        try {
            st = con.createStatement();
            rs = st.executeQuery("select * from DBCOURSE_PRODUCTIONCOMPANY order by production_name");

            // DefaultTableModel에 있는 기존 데이터 지우기
            for (int i = 0; i < t_model.getRowCount();) {
                t_model.removeRow(0);
            }

            while (rs.next()) {
                Object data[] = { rs.getString(1), rs.getString(2) };

                t_model.addRow(data); //DefaultTableModel에 레코드 추가
            }
```

```
        } catch (SQLException e) {
            System.out.println(e + "=> userSelectAll fail");
        } finally {
            dbClose();
        }
    }//proSelectAll()

    //delete selected record
    public int proDelete(String production_name) {
        int result = 0;
        try {
            ps = con.prepareStatement("delete from DBCOURSE_PRODUCTIONCOMPANY  where production_name = ? ");
            ps.setString(1, production_name.trim());
            result = ps.executeUpdate();

        } catch (SQLException e) {
            System.out.println(e + "=> userDelete fail");
        }finally {
            dbClose();
        }

        return result;
    }//proDelete()

    //update selected record
    public int proUpdate(ProductionJDailogGUI user) {
        int result = 0;
        String sql = "UPDATE DBCOURSE_PRODUCTIONCOMPANY SET production_name=?, pceo=?";

        try {
            ps = con.prepareStatement(sql);
            // ?의 순서대로 값 넣기
            ps.setString(1, user.txproduction_name.getText());
            ps.setString(2, user.txpceo.getText());

            // 실행하기
            result = ps.executeUpdate();


        } catch (SQLException e) {
            System.out.println(e + "=> Update fail");
        } finally {
            dbClose();
        }

        return result;
    }//proUpdate()

    /**
     * 검색단어에 해당하는 레코드 검색하기 (like연산자를 사용하여 _, %를 사용할때는 PreparedStatemnet안된다. 반드시
     * Statement객체를 이용함)
     * */
    public void proSearch(DefaultTableModel dt, String fieldName,
            String word) {
        String sql = "SELECT * FROM DBCOURSE_PRODUCTIONCOMPANY WHERE " + fieldName.trim()
                + " LIKE '%" + word.trim() + "%'";

        try {
            st = con.createStatement();
            rs = st.executeQuery(sql);

            // DefaultTableModel에 있는 기존 데이터 지우기
            for (int i = 0; i < dt.getRowCount();) {
                dt.removeRow(0);
            }

            while (rs.next()) {
                Object data[] = { rs.getString(1), rs.getString(2) };

                dt.addRow(data);
            }

        } catch (SQLException e) {
            System.out.println(e + "=> getUserSearch fail");
        } finally {
            dbClose();
        }

    }//proSearch()

}//End of Class
```

```java
import java.awt.BorderLayout;


public class ProductionJDailogGUI extends JDialog implements ActionListener{

    JPanel pw=new JPanel(new GridLayout(5,1));
    JPanel pc=new JPanel(new GridLayout(5,1));
    JPanel ps=new JPanel();

    JLabel lable_production_name = new JLabel("제작사");
    JLabel lable_pceo=new JLabel("대표");

    JTextField txproduction_name=new JTextField();
    JTextField txpceo=new JTextField();

    JButton confirm;
    JButton reset=new JButton("cancle");

   ProductionJTableExam me;


   ProductionDefaultJTableDAO dao =new ProductionDefaultJTableDAO();


    public ProductionJDailogGUI(ProductionJTableExam me, String index){
        super(me,"다이어로그");
        this.me=me;
        if(index.equals("insert")){
            confirm=new JButton(index);
        }else{
            confirm=new JButton("update");

            //text박스에 선택된 레코드의 정보 넣기
            int row = me.jt.getSelectedRow();//선택된 행
            txproduction_name.setText( me.jt.getValueAt(row, 0).toString() );
            txpceo.setText( me.jt.getValueAt(row, 1).toString() );


            //productionname inactivate
            txproduction_name.setEditable(false);

        }


        //Label추가부분
        pw.add(lable_production_name);//제작사
        pw.add(lable_pceo);//대표

        //TextField 추가
        pc.add(txproduction_name);
        pc.add(txpceo);

        ps.add(confirm); //update
        ps.add(reset);//cancle

        add(pw,"West");
        add(pc,"Center");
        add(ps,"South");

        setSize(500,350);
        setVisible(true);

        setDefaultCloseOperation(JDialog.DISPOSE_ON_CLOSE);

        //이벤트등록
        confirm.addActionListener(this); //가입/수정 이벤트등록
        reset.addActionListener(this); //취소 이벤트등록

    }//end of constructor

    //insert/update/delete actionListener
    @Override
    public void actionPerformed(ActionEvent e) {
       String btnLabel =e.getActionCommand();//이벤트주체 대한 Label 가져오기

```

```
        if(btnLabel.equals("insert")){
            if(dao.proListInsert(this) > 0 ){//insert success
                messageBox(this , txproduction_name.getText()+" insert success");
                dispose();//JDialog 창닫기

                dao.proSelectAll(me.dt);//모든레코드가져와서 DefaultTableModel에 올리기

                if(me.dt.getRowCount() > 0)
                    me.jt.setRowSelectionInterval(0, 0);//select first row

            }else{//insert failure
                messageBox(this,"insert failure");
            }

        }else if(btnLabel.equals("update")){

             if( dao.proUpdate(this) > 0){
                 messageBox(this, "update complete");
                 dispose();
                 dao.proSelectAll(me.dt);
                 if(me.dt.getRowCount() > 0 ) me.jt.setRowSelectionInterval(0, 0);

             }else if(  dao.proCheck(txproduction_name.getText()) ){
                 messageBox(this, "update failure : duplicate record");
             }else{
                 messageBox(this, "update failure");
             }

        }else if(btnLabel.equals("cancle")){
            dispose();
        }


    }//actionPerformed끝


    //MessageBox Constructor
    public static void messageBox(Object obj , String message){
        JOptionPane.showMessageDialog( (Component)obj , message);
    }

}//End of Class
```

```
import java.awt.Color;

public class ProductionJTableExam extends JFrame implements ActionListener {
    JMenu m = new JMenu("관리");
    JMenuItem insert = new JMenuItem("insert");
    JMenuItem update = new JMenuItem("update");
    JMenuItem delete = new JMenuItem("delete");
    JMenuItem quit = new JMenuItem("exit");
    JMenuBar mb = new JMenuBar();

    String[] name = { "Production_name", "representative" };

    DefaultTableModel dt = new DefaultTableModel(name, 0);
    JTable jt = new JTable(dt);
    JScrollPane jsp = new JScrollPane(jt);

    /*
     * South 영역에 추가할 Componet들
     */
    JPanel p = new JPanel();
    String[] comboName = { "  ALL  ", "Production_name", "pceo" };

    JComboBox combo = new JComboBox(comboName);
    JTextField jtf = new JTextField(20);
    JButton serach = new JButton("Search");

    ProductionDefaultJTableDAO dao = new ProductionDefaultJTableDAO();

    /**
     * 화면구성 및 이벤트등록
     */
    public ProductionJTableExam() {

        super("GUI ProductionInfo - DB");

        //메뉴아이템을 메뉴에 추가
        m.add(insert);
```

```
        m.add(update);
        m.add(delete);
        m.add(quit);
        //메뉴를 메뉴바에 추가
        mb.add(m);

        //윈도우에 메뉴바 세팅
        setJMenuBar(mb);

        // South영역
        p.setBackground(Color.yellow);
        p.add(combo);
        p.add(jtf);
        p.add(serach);

        add(jsp, "Center");
        add(p, "South");

        setSize(700, 500);
        setVisible(true);


        // 이벤트등록
        insert.addActionListener(this);
        update.addActionListener(this);
        delete.addActionListener(this);
        quit.addActionListener(this);
        serach.addActionListener(this);

        // 모든레코드를 검색하여 DefaultTableModle에 올리기
        dao.proSelectAll(dt);

        //첫번행 선택.
        if (dt.getRowCount() > 0)
            jt.setRowSelectionInterval(0, 0);

    }// 생성자끝


    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == insert) {// insert메뉴아이템 클릭
            new ProductionJDailogGUI(this, "insert");

        } else if (e.getSource() == update) {//update 메뉴아이템 클릭
            new ProductionJDailogGUI(this, "update");

        } else if (e.getSource() == delete) {//delete 메뉴아이템 클릭
            // 현재 Jtable의 선택된 행과 열의 값을 얻어온다.
            int row = jt.getSelectedRow();
            System.out.println("선택행 : " + row);

            Object obj = jt.getValueAt(row, 0);// 행열에 해당하는 value
            System.out.println("값 : " + obj);

            if (dao.proDelete(obj.toString()) > 0) {
                ProductionJDailogGUI.messageBox(this, "Record Delete.");

                //리스트 갱신
                dao.proSelectAll(dt);
                if (dt.getRowCount() > 0)
                    jt.setRowSelectionInterval(0, 0);

            } else {
                ProductionJDailogGUI.messageBox(this, "failed deleting record.");
            }

        } else if (e.getSource() == quit) {// 종료 메뉴아이템 클릭
            System.exit(0);

        } else if (e.getSource() == serach) {// 검색 버튼 클릭
            // JComboBox에 선택된 value 가져오기
            String fieldName = combo.getSelectedItem().toString();
            System.out.println("필드명 " + fieldName);

            if (fieldName.trim().equals("ALL")) {// 전체검색
```

```
                dao.proSelectAll(dt);
                if (dt.getRowCount() > 0)
                    jt.setRowSelectionInterval(0, 0);
            } else {
                if (jtf.getText().trim().equals("")) {
                    ProductionJDailogGUI.messageBox(this, "검색단어를 입력해주세요!");
                    jtf.requestFocus();
                } else {// 검색어를 입력했을경우
                    dao.proSearch(dt, fieldName, jtf.getText());
                    if (dt.getRowCount() > 0)
                        jt.setRowSelectionInterval(0, 0);
                }
            }
        }

    }//actionPerformed()----------

}
```

```
2 import java.sql.Connection;
10
11 public class ReviewDefaultJTableDAO {
12     //variables for connection
13     Connection con;
14     Statement st;
15     PreparedStatement ps;
16     ResultSet rs;
17
18     public ReviewDefaultJTableDAO() {
19         try {
20             //load
21             Class.forName("com.mysql.jdbc.Driver");
22
23             con = DriverManager.getConnection("jdbc:mysql://localhost:3306/team01?useSSL=false","team01","team01");
24
25         } catch (ClassNotFoundException e) {
26             System.out.println(e + "=> load fail");
27         } catch (SQLException e) {
28             System.out.println(e + "=> connection fail");
29         }
30     }//Constructor end
31
32     /**
33      * DB닫기 기능 메소드
34      * */
35     public void dbClose() {
36         try {
37             if (rs != null) rs.close();
38             if (st != null) st.close();
39             if (ps != null) ps.close();
40         } catch (Exception e) {
41             System.out.println(e + "=> dbClose fail");
42         }
43     }//dbClose() ---
44
```

```java
    //insert review record
    public int reviewListInsert(ReviewJDailogGUI user) {
        int result = 0;
        try {
            ps = con.prepareStatement("insert into DBCOURSE_REVIEW values(?,?,?)");
            ps.setString(1, user.Reviewer_Name.getText());
            ps.setString(2, user.Mov.getText());
            ps.setFloat(3, Float.parseFloat(user.Grade.getText()));

            result = ps.executeUpdate(); //실행 -> 저장

        } catch (SQLException e) {
            System.out.println(e + "=> userListInsert fail");
        } finally {
            dbClose();
        }

        return result;

    }//reviewListInsert()

    //select all review record
    public void reviewSelectAll(DefaultTableModel t_model) {
        try {
            st = con.createStatement();
            rs = st.executeQuery("select * from DBCOURSE_REVIEW order by reviewer_name");

            // DefaultTableModel에 있는 기존 데이터 지우기
            for (int i = 0; i < t_model.getRowCount();) {
                t_model.removeRow(0);
            }

            while (rs.next()) {
                Object data[] = { rs.getString(1), rs.getString(2),rs.getString(3)};

                t_model.addRow(data); //DefaultTableModel에 레코드 추가
            }


        } catch (SQLException e) {
            System.out.println(e + "=> userSelectAll fail");
        } finally {
            dbClose();
        }
    }//reviewSelectAll()

    //delete selected record
    public int reviewDelete(String Reviewer_Name) {
        int result = 0;
        try {
            ps = con.prepareStatement("delete from DBCOURSE_REVIEW  where reviewer_name = ? ");
            ps.setString(1, Reviewer_Name.trim());
            result = ps.executeUpdate();

        } catch (SQLException e) {
            System.out.println(e + "=> userDelete fail");
        }finally {
            dbClose();
        }

        return result;
    }//reviewDelete()

    //update selected record
    public int userUpdate(ReviewJDailogGUI user) {
        int result = 0;
        String sql = "UPDATE DBCOURSE_REVIEW SET reviewer=?, grade=? WHERE movie=?";

        try {
            ps = con.prepareStatement(sql);
            // ?의 순서대로 값 넣기
            ps.setString(1, user.Reviewer_Name.getText());
            ps.setString(3, user.Mov.getText());
            ps.setFloat(2, Float.parseFloat(user.Grade.getText().trim()));
```

```
            // 실행하기
            result = ps.executeUpdate();

        } catch (SQLException e) {
            System.out.println(e + "=> Update fail");
        } finally {
            dbClose();
        }

        return result;
    }//reviewUpdate()

    /**
     * 검색단어에 해당하는 레코드 검색하기 (like연산자를 사용하여 _, %를 사용할때는 PreparedStatemnet안된다. 반드시
     * Statement객체를 이용함)
     * */
    public void reviewSearch(DefaultTableModel dt, String fieldName,
            String word) {
        String sql = "SELECT * FROM DBCOURSE_REVIEW WHERE " + fieldName.trim()
                + " LIKE '%" + word.trim() + "%'";

        try {
            st = con.createStatement();
            rs = st.executeQuery(sql);

            // DefaultTableModel에 있는 기존 데이터 지우기
            for (int i = 0; i < dt.getRowCount();) {
                dt.removeRow(0);
            }

            while (rs.next()) {
                Object data[] = { rs.getString(1), rs.getString(2),rs.getString(3) };

                dt.addRow(data);
            }


        } catch (SQLException e) {
            System.out.println(e + "=> getUserSearch fail");
        } finally {
            dbClose();
        }

    }//reviewSearch()

}//End of Class
```

```
import java.awt.BorderLayout;

public class ReviewJDailogGUI extends JDialog implements ActionListener{

    JPanel pw=new JPanel(new GridLayout(3,1));
    JPanel pc=new JPanel(new GridLayout(3,1));
    JPanel ps=new JPanel();

    JLabel lable_Reviewer_Name = new JLabel("Reviewer Name");
    JLabel lable_Movie=new JLabel("Movie");
    JLabel lable_Grade=new JLabel("Grade");


    JTextField Reviewer_Name=new JTextField();
    JTextField Mov=new JTextField();
    JTextField Grade=new JTextField();

    JButton confirm;
    JButton reset=new JButton("cancle");

   ReviewJTabaleExam me;

   ReviewDefaultJTableDAO dao =new ReviewDefaultJTableDAO();


    public ReviewJDailogGUI(ReviewJTabaleExam me, String index){
        super(me,"Revie Table");
        this.me=me;
        if(index.equals("insert")){
            confirm=new JButton(index);
        }else{
            confirm=new JButton("update");

            //text박스에 선택된 레코드의 정보 넣기
            int row = me.jt.getSelectedRow();//선택된 행
            Reviewer_Name.setText( me.jt.getValueAt(row, 0).toString() );
            Mov.setText( me.jt.getValueAt(row, 1).toString() );
```

```
            Grade.setText( me.jt.getValueAt(row, 2).toString() );

            //reviewer text박스 비활성
            Reviewer_Name.setEditable(false);

        }

        //Label추가부분
        pw.add(lable_Reviewer_Name);//평론가 이름
        pw.add(lable_Movie);//영화
        pw.add(lable_Grade);//평점

        //TextField 추가
        pc.add(Reviewer_Name);
        pc.add(Mov);
        pc.add(Grade);


        ps.add(confirm); //update
        ps.add(reset);//cancle

        add(pw,"West");
        add(pc,"Center");
        add(ps,"South");

        setSize(500,350);
        setVisible(true);

        setDefaultCloseOperation(JDialog.DISPOSE_ON_CLOSE);

        //이벤트등록
        confirm.addActionListener(this); //가입/수정 이벤트등록
        reset.addActionListener(this); //취소 이벤트등록

    }//생성자끝

    @Override
    public void actionPerformed(ActionEvent e) {
        String btnLabel =e.getActionCommand();//이벤트주체 대한 Label 가져오기

        if(btnLabel.equals("insert")){
            if(dao.reviewListInsert(this) > 0 ){
                messageBox(this , Reviewer_Name.getText()+" insert success");
                dispose();//JDialog 창닫기

                dao.reviewSelectAll(me.dt);//모든레코드가져와서 DefaultTableModel에 올리기

                if(me.dt.getRowCount() > 0)
                    me.jt.setRowSelectionInterval(0, 0);//첫번째 행 선택

            }else{//가입실패
                messageBox(this,"insert failure");
            }

        }else if(btnLabel.equals("update")){

             if( dao.userUpdate(this) > 0){
                 messageBox(this, "수정완료되었습니다.");
                 dispose();
                 dao.reviewSelectAll(me.dt);
                 if(me.dt.getRowCount() > 0 ) me.jt.setRowSelectionInterval(0, 0);

             }else{
                 messageBox(this, "수정되지 않았습니다.");
             }



        }else if(btnLabel.equals("cancle")){
            dispose();
        }



    }//actionPerformed end

    //MessageBox Constructor
    public static void messageBox(Object obj , String message){
        JOptionPane.showMessageDialog( (Component)obj , message);
    }

}//End of Class
```

```java
import java.awt.Color;

public class ReviewJTabaleExam extends JFrame implements ActionListener {
    JMenu m = new JMenu("관리");
    JMenuItem insert = new JMenuItem("insert");
    JMenuItem update = new JMenuItem("update");
    JMenuItem delete = new JMenuItem("delete");
    JMenuItem quit = new JMenuItem("exit");
    JMenuBar mb = new JMenuBar();

    String[] name = { "Reviewer Name", "Movie", "Grade" };

    DefaultTableModel dt = new DefaultTableModel(name, 0);
    JTable jt = new JTable(dt);
    JScrollPane jsp = new JScrollPane(jt);

    /*
     * South panel Components for Search
     */
    JPanel p = new JPanel();
    String[] comboName = { " ALL ", " Reviewer Name ", " Movie ", " Grade " };

    JComboBox combo = new JComboBox(comboName);
    JTextField jtf = new JTextField(20);
    JButton serach = new JButton("Search");

    ReviewDefaultJTableDAO dao = new ReviewDefaultJTableDAO();

    /**
     * 화면구성 및 이벤트등록
     */
    public ReviewJTabaleExam() {

        super("GUI reviewInfo - DB");

        //메뉴아이템을 메뉴에 추가
        m.add(insert);
        m.add(update);
        m.add(delete);
        m.add(quit);
        //메뉴를 메뉴바에 추가
        mb.add(m);

        //윈도우에 메뉴바 세팅
        setJMenuBar(mb);

        // South영역
        p.setBackground(Color.yellow);
        p.add(combo);
        p.add(jtf);
        p.add(serach);

        add(jsp, "Center");
        add(p, "South");

        setSize(700, 500);
        setVisible(true);

        // 이벤트등록
        insert.addActionListener(this);
        update.addActionListener(this);
        delete.addActionListener(this);
        quit.addActionListener(this);
        serach.addActionListener(this);

        // 모든레코드를 검색하여 DefaultTableModle에 올리기
        dao.reviewSelectAll(dt);

        //첫번행 선택.
        if (dt.getRowCount() > 0)
            jt.setRowSelectionInterval(0, 0);

    }// 생성자끝
```

```java
    //insert/update/delete
    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == insert) {// 가입 메뉴아이템 클릭
            new ReviewJDailogGUI(this, "insert");

        } else if (e.getSource() == update) {// 수정 메뉴아이템 클릭
            new ReviewJDailogGUI(this, "update");

        } else if (e.getSource() == delete) {// 삭제 메뉴아이템 클릭
            // 현재 Jtable의 선택된 행과 열의 값을 얻어온다.
            int row = jt.getSelectedRow();
            System.out.println("선택행 : " + row);

            Object obj = jt.getValueAt(row, 0);// 행 열에 해당하는 value
            System.out.println("값 : " + obj);

            if (dao.reviewDelete(obj.toString()) > 0) {
                ReviewJDailogGUI.messageBox(this, "Record Delete.");

                //리스트 갱신
                dao.reviewSelectAll(dt);
                if (dt.getRowCount() > 0)
                    jt.setRowSelectionInterval(0, 0);

            } else {
                ReviewJDailogGUI.messageBox(this, "failed deleting record.");
            }

        } else if (e.getSource() == quit) {// 종료 메뉴아이템 클릭
            System.exit(0);

        } else if (e.getSource() == serach) {// 검색 버튼 클릭
            // JComboBox에 선택된 value 가져오기
            String fieldName = combo.getSelectedItem().toString();
            System.out.println("필드명 " + fieldName);

            if (fieldName.trim().equals("ALL")) {// 전체검색
                dao.reviewSelectAll(dt);
<
```

```java
                if (dt.getRowCount() > 0)
                    jt.setRowSelectionInterval(0, 0);
            } else {
                if (jtf.getText().trim().equals("")) {
                    ReviewJDailogGUI.messageBox(this, "검색단어를 입력해주세요!");
                    jtf.requestFocus();
                } else {// 검색어를 입력했을경우
                    dao.reviewSearch(dt, fieldName, jtf.getText());
                    if (dt.getRowCount() > 0)
                        jt.setRowSelectionInterval(0, 0);
                }
            }
        }

    }//actionPerformed()----------

}
```

```java
import java.math.BigInteger;

public class StatisticDefaultJTableDAO {

    //variabale for connection
    Connection con;
    Statement st;
    PreparedStatement ps;
    ResultSet rs;

    //Constructor for loading
    public StatisticDefaultJTableDAO() {
        try {
            // 로드
            Class.forName("com.mysql.jdbc.Driver");

            con = DriverManager.getConnection("jdbc:mysql://localhost:3306/team1?useSSL=false","team1","team1");

        } catch (ClassNotFoundException e) {
            System.out.println(e + "=> load fail");
        } catch (SQLException e) {
            System.out.println(e + "=> connection fail");
        }
    }//생성자

    public void dbClose() {
        try {
            if (rs != null) rs.close();
            if (st != null) st.close();
            if (ps != null) ps.close();
        } catch (Exception e) {
            System.out.println(e + "=> dbClose fail");
        }
    }//dbClose() ---

    //insert record

   public int statisticListInsert(StatisticJDailogGUI user) {
       int result = 0;
       try {
           ps = con.prepareStatement("insert into DBCOURSE_MOVIE_STATISTIC values(?,?,?,?)");
           ps.setString(1, user.M_N.getText());
           ps.setLong(2, Integer.parseInt(user.T_A.getText()));
           ps.setLong(3, Integer.parseInt(user.A_S.getText()));
           ps.setInt(4, Integer.parseInt(user.S_N.getText()));

           result = ps.executeUpdate(); //실행 -> 저장

       } catch (SQLException e) {
           System.out.println(e + "=> userListInsert fail");
       } finally {
           dbClose();
       }

       return result;

   }//statisticListInsert()

   //select all statistic record
   public void statisticSelectAll(DefaultTableModel t_model) {
       try {
           st = con.createStatement();
           rs = st.executeQuery("select * from DBCOURSE_MOVIE_STATISTIC order by movie");

           // DefaultTableModel에 있는 기존 데이터 지우기
           for (int i = 0; i < t_model.getRowCount();) {
               t_model.removeRow(0);
           }

           while (rs.next()) {
               Object data[] = { rs.getString(1), rs.getString(2),rs.getString(3),
                       rs.getInt(4) };

               t_model.addRow(data); //DefaultTableModel에 레코드 추가
           }
```

```
        } catch (SQLException e) {
            System.out.println(e + "=> userSelectAll fail");
        } finally {
            dbClose();
        }
    }//statisticSelectAll()

    //delete selected record
    public int statisticDelete(String M_N) {
        int result = 0;
        try {
            ps = con.prepareStatement("delete from DBCOURSE_MOVIE_STATISTIC  where movie = ? ");
            ps.setString(1, M_N.trim());
            result = ps.executeUpdate();

        } catch (SQLException e) {
            System.out.println(e + "=> userDelete fail");
        }finally {
            dbClose();
        }

        return result;
    }//statisticDelete()

    /**
     * ID에 해당하는 레코드 수정하기
     * */
    public int statisticUpdate(StatisticJDailogGUI user) {
        int result = 0;
        String sql = "UPDATE DBCOURSE_MOVIE_STATISTIC SET total_attendance=?, accumulated_sales=?, screen_number=? "+
        "WHERE movie=?";

        try {
            ps = con.prepareStatement(sql);
            // ?의 순서대로 값 넣기
            ps.setString(4, user.M_N.getText());
            ps.setInt(1, Integer.parseInt(user.T_A.getText()));
            ps.setInt(2, Integer.parseInt(user.A_S.getText()));
            ps.setInt(3, Integer.parseInt(user.S_N.getText().trim()));

            // 실행하기
            result = ps.executeUpdate();

        } catch (SQLException e) {
            System.out.println(e + "=> Update fail");
        } finally {
            dbClose();
        }

        return result;
    }//statisticUpdate()

    /**
     * 검색단어에 해당하는 레코드 검색하기 (like연산자를 사용하여 _, %를 사용할때는 PreparedStatemnet안된다. 반드시
     * Statement객체를 이용함)
     * */
    public void statisticSearch(DefaultTableModel dt, String fieldName,
            String word) {
        String sql = "SELECT * FROM DBCOURSE_MOVIE_STATISTIC WHERE " + fieldName.trim()
                + " LIKE '%" + word.trim() + "%'";

        try {
            st = con.createStatement();
            rs = st.executeQuery(sql);

            // DefaultTableModel에 있는 기존 데이터 지우기
            for (int i = 0; i < dt.getRowCount();) {
                dt.removeRow(0);
            }

            while (rs.next()) {
                Object data[] = { rs.getString(1), rs.getString(2),rs.getString(3),
                        rs.getInt(4) };

                dt.addRow(data);
            }
```

```
      } catch (SQLException e) {
         System.out.println(e + "=> getUserSearch fail");
      } finally {
         dbClose();
      }

   }//statisticSearch()

}//End of Class
```

```
import java.awt.BorderLayout;

public class StatisticJDailogGUI extends JDialog implements ActionListener{

    JPanel pw=new JPanel(new GridLayout(4,1));
    JPanel pc=new JPanel(new GridLayout(4,1));
    JPanel ps=new JPanel();

    JLabel lable_Movie = new JLabel("Movie");//4개로
    JLabel lable_Total_Attendance=new JLabel("Total Attendance");
    JLabel lable_Accumulate_Sales=new JLabel("Accumulate Sales");
    JLabel lable_Screen_Number=new JLabel("Screen Number");


    JTextField M_N=new JTextField();
    JTextField T_A=new JTextField();
    JTextField A_S=new JTextField();
    JTextField S_N=new JTextField();


    JButton confirm;
    JButton reset=new JButton("cancle");

   StatisticJTabaleExam me;

   StatisticDefaultJTableDAO dao =new StatisticDefaultJTableDAO();


    public StatisticJDailogGUI(StatisticJTabaleExam me, String index){
        super(me,"Statistic Table");
        this.me=me;
        if(index.equals("insert")){
            confirm=new JButton(index);
        }else{
            confirm=new JButton("update");

            //text박스에 선택된 레코드의 정보 넣기
            int row = me.jt.getSelectedRow();//선택된 행
            M_N.setText( me.jt.getValueAt(row, 0).toString() );
            T_A.setText( me.jt.getValueAt(row, 1).toString() );
            A_S.setText( me.jt.getValueAt(row, 2).toString() );
            S_N.setText( me.jt.getValueAt(row, 3).toString() );

            //id text박스 비활성
            M_N.setEditable(false);

        }


        //Label추가부분
        pw.add(lable_Movie);//영화
        pw.add(lable_Total_Attendance);//관객수
        pw.add(lable_Accumulate_Sales);//수익
        pw.add(lable_Screen_Number);//total screenNum


        //TextField 추가
        pc.add(M_N);
        pc.add(T_A);
        pc.add(A_S);
        pc.add(S_N);


        ps.add(confirm); //update
        ps.add(reset);//cancle

        add(pw,"West");
        add(pc,"Center");
        add(ps,"South");

        setSize(500,350);
        setVisible(true);

        setDefaultCloseOperation(JDialog.DISPOSE_ON_CLOSE);
```

```
        //이벤트등록
        confirm.addActionListener(this); //가입/수정 이벤트등록
        reset.addActionListener(this); //취소 이벤트등록

    }//생성자끝

    /**
     * 가입/수정/삭제 기능에 대한 부분
     * */
    @Override
    public void actionPerformed(ActionEvent e) {
       String btnLabel =e.getActionCommand();//이벤트주체 대한 Label 가져오기

       if(btnLabel.equals("insert")){
           if(dao.statisticListInsert(this) > 0 ){//insert complete
               messageBox(this , M_N.getText()+"insert success");
               dispose();//JDialog 창닫기

               dao.statisticSelectAll(me.dt);//모든레코드가져와서 DefaultTableModel에 올리기

               if(me.dt.getRowCount() > 0)
                   me.jt.setRowSelectionInterval(0, 0);//첫번째 행 선택

           }else{//insert failure
               messageBox(this,"insert failure");
           }

       }else if(btnLabel.equals("update")){

            if( dao.statisticUpdate(this) > 0){
                messageBox(this, "update complete");
                dispose();
                dao.statisticSelectAll(me.dt);
                if(me.dt.getRowCount() > 0 ) me.jt.setRowSelectionInterval(0, 0);

            }else{
                messageBox(this, "update failure");
            `

                }


       }else if(btnLabel.equals("cancle")){
           dispose();
       }


    }//actionPerformed end

    //MessageBos Constructor
    public static void messageBox(Object obj , String message){
        JOptionPane.showMessageDialog( (Component)obj , message);
    }

}//End of Class
```

```java
import java.awt.Color;


public class StatisticJTabaleExam extends JFrame implements ActionListener {
    JMenu m = new JMenu("Menu");
    JMenuItem insert = new JMenuItem("insert");
    JMenuItem update = new JMenuItem("update");
    JMenuItem delete = new JMenuItem("delete");
    JMenuItem quit = new JMenuItem("exit");
    JMenuBar mb = new JMenuBar();

    String[] name = { "Movie Name", "Total Attendance", "Accumulate Sales(만 단위)", "Screen Number" };

    DefaultTableModel dt = new DefaultTableModel(name, 0);
    JTable jt = new JTable(dt);
    JScrollPane jsp = new JScrollPane(jt);

    /*
     * South 영역에 추가할 Componet들
     */
    JPanel p = new JPanel();
    String[] comboName = { "ALL", "Total Attendance", "Accumulate Sales", "Screen Number" };

    JComboBox combo = new JComboBox(comboName);
    JTextField jtf = new JTextField(20);
    JButton serach = new JButton("Search");

    StatisticDefaultJTableDAO dao = new StatisticDefaultJTableDAO();

    /**
     * 화면구성 및 이벤트등록
     */
    public StatisticJTabaleExam() {

        super("GUI StatisticInfo - DB");
```

```java
        //메뉴아이템을 메뉴에 추가
        m.add(insert);
        m.add(update);
        m.add(delete);
        m.add(quit);
        //메뉴를 메뉴바에 추가
        mb.add(m);

        //윈도우에 메뉴바 세팅
        setJMenuBar(mb);


        // South영역
        p.setBackground(Color.yellow);
        p.add(combo);
        p.add(jtf);
        p.add(serach);

        add(jsp, "Center");
        add(p, "South");

        setSize(700, 500);
        setVisible(true);

        // 이벤트등록
        insert.addActionListener(this);
        update.addActionListener(this);
        delete.addActionListener(this);
        quit.addActionListener(this);
        serach.addActionListener(this);

        // 모든레코드를 검색하여 DefaultTableModle에 올리기
        dao.statisticSelectAll(dt);

        //첫번행 선택.
        if (dt.getRowCount() > 0)
            jt.setRowSelectionInterval(0, 0);
```

```java
    }// 생성자끝

    /**
     * 가입/수정/삭제/검색기능을 담당하는 메소드
     * */

    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == insert) {// 가입 메뉴아이템 클릭
            new StatisticJDailogGUI(this, "insert");

        } else if (e.getSource() == update) {// 수정 메뉴아이템 클릭
            new StatisticJDailogGUI(this, "update");

        } else if (e.getSource() == delete) {// 삭제 메뉴아이템 클릭
            // 현재 Jtable의 선택된 행과 열의 값을 얻어온다.
            int row = jt.getSelectedRow();
            System.out.println("선택행 : " + row);

            Object obj = jt.getValueAt(row, 0);// 행 열에 해당하는 value
            System.out.println("값 : " + obj);

            if (dao.statisticDelete(obj.toString()) > 0) {
                StatisticJDailogGUI.messageBox(this, "Record Delete.");

                //리스트 갱신
                dao.statisticSelectAll(dt);
                if (dt.getRowCount() > 0)
                    jt.setRowSelectionInterval(0, 0);

            } else {
                StatisticJDailogGUI.messageBox(this, "failed deleting record.");
            }

        } else if (e.getSource() == quit) {// 종료 메뉴아이템 클릭
            System.exit(0);


          } else if (e.getSource() == serach) {// 검색 버튼 클릭
              // JComboBox에 선택된 value 가져오기
              String fieldName = combo.getSelectedItem().toString();
              System.out.println("필드명 " + fieldName);

              if (fieldName.trim().equals("ALL")) {// 전체검색
                  dao.statisticSelectAll(dt);
                  if (dt.getRowCount() > 0)
                      jt.setRowSelectionInterval(0, 0);
              } else {
                  if (jtf.getText().trim().equals("")) {
                      StatisticJDailogGUI.messageBox(this, "검색단어를 입력해주세요!");
                      jtf.requestFocus();
                  } else {// 검색어를 입력했을경우
                      dao.statisticSearch(dt, fieldName, jtf.getText());
                      if (dt.getRowCount() > 0)
                          jt.setRowSelectionInterval(0, 0);
                  }
              }
          }

     }//actionPerformed()----------

}
```

-IndexQuery

```
import java.sql.*;

public class IndexQuery extends JFrame {

    String[] foundS = { "Movie", "Released Date"};
    DefaultTableModel dj = new DefaultTableModel(foundS,0);
    JTable jt2 = new JTable(dj);
    JScrollPane jsp2 = new JScrollPane(jt2);

    Connection con;
    Statement st;
    PreparedStatement ps;
    ResultSet rs;

    public IndexQuery(){
        super("Movie released in same year and month");

        setSize(700, 500);
        setVisible(true);
        add(jsp2, "Center");

        try {
            // 로드
            Class.forName("com.mysql.jdbc.Driver");

            con = DriverManager.getConnection("jdbc:mysql://localhost:3306/team01?useSSL=false","team01","team01");
             getReleasedMovie(dj);
        } catch (ClassNotFoundException e) {
            System.out.println(e + "=> load fail");
        } catch (SQLException e) {
            System.out.println(e + "=> connection fail");
        }

        //첫번행 선택.
        if (dj.getRowCount() > 0)
            jt2.setRowSelectionInterval(0, 0);

    }


      public void getReleasedMovie(DefaultTableModel dt) {
          String sql = "select a.movie, date_format(a.crank_in_date,'%Y-%m-%d')"+
                  "as crank_in_date, date_format(b.crank_up_date, '%Y-%m-%d') as crank_up_date " +
                  "from dbcourse_movie_date a, dbcourse_movie_date b " +
                  "where (a.crank_in_date between '2011-12-01' and '2014-12-01') "+
                  "and (b.crank_up_date between '2011-12-01' and '2014-12-01');";
          try {
             ps = con.prepareStatement(sql);
             rs = ps.executeQuery();

             // DefaultTableModel에 있는 기존 데이터 지우기
             for (int i = 0; i < dt.getRowCount();) {
                dt.removeRow(0);
             }

             while (rs.next()) {
                Object data[] = { rs.getString(1), rs.getString(2) };

                dt.addRow(data);
             }

          } catch (SQLException e) {
             System.out.println(e + "=> getUserSearch fail");
          } finally {
              try {
                if (rs != null) rs.close();
                if (st != null) st.close();
                if (ps != null) ps.close();
              } catch (Exception e) {
                System.out.println(e + "=> dbClose fail");
              }
          }
      }

}
```

-JoinQuery

```
import java.sql.*;

public class JoinQuery extends JFrame {

    String[] foundS = { "Director", "Movie", "total viewer"};
    DefaultTableModel dj = new DefaultTableModel(foundS,0);
    JTable jt2 = new JTable(dj);
    JScrollPane jsp2 = new JScrollPane(jt2);

    Connection con;
    Statement st;
    PreparedStatement ps;
    ResultSet rs;

    public JoinQuery(){
        super("500만 관객 이상이 본 영화");

        setSize(700, 500);
        setVisible(true);
        add(jsp2, "Center");

        try {
            // 로드
            Class.forName("com.mysql.jdbc.Driver");

            con = DriverManager.getConnection("jdbc:mysql://localhost:3306/team01?useSSL=false","team01","team01");
             getSucceedMovie(dj);
        } catch (ClassNotFoundException e) {
            System.out.println(e + "=> load fail");
        } catch (SQLException e) {
            System.out.println(e + "=> connection fail");
        }

        //첫번행 선택.
        if (dj.getRowCount() > 0)
            jt2.setRowSelectionInterval(0, 0);

    }


     public void getSucceedMovie(DefaultTableModel dt) {
         String sql = "select director,M.movie, total_attendance as viewer from DBCOURSE_MOVIE M ,"
                 + "DBCOURSE_MOVIE_STATISTIC S where  M.movie = S.movie and total_attendance > 5000000;";
         try {
            ps = con.prepareStatement(sql);
            rs = ps.executeQuery();

            // DefaultTableModel에 있는 기존 데이터 지우기
            for (int i = 0; i < dt.getRowCount();) {
               dt.removeRow(0);
            }

            while (rs.next()) {
               Object data[] = { rs.getString(1), rs.getString(2),rs.getString(3) };

               dt.addRow(data);
            }

         } catch (SQLException e) {
            System.out.println(e + "=> getUserSearch fail");
         } finally {
             try {
               if (rs != null) rs.close();
               if (st != null) st.close();
               if (ps != null) ps.close();
           } catch (Exception e) {
               System.out.println(e + "=> dbClose fail");
           }
         }
      }

}
```

-ViewQuery

```
import java.sql.*;

public class ViewQuery extends JFrame {

    String[] foundS = { "movie", "runtime"};
    DefaultTableModel dj = new DefaultTableModel(foundS,0);
    JTable jt2 = new JTable(dj);
    JScrollPane jsp2 = new JScrollPane(jt2);

    Connection con;
    Statement st;
    PreparedStatement ps;
    ResultSet rs;

    public ViewQuery(){
        super("영화 상영시간");

        setSize(700, 500);
        setVisible(true);
        add(jsp2, "Center");

        try {
            // 로드
            Class.forName("com.mysql.jdbc.Driver");

            con = DriverManager.getConnection("jdbc:mysql://localhost:3306/team01?useSSL=false","team01","team01");
            getViewMovieTime(dj);
        } catch (ClassNotFoundException e) {
            System.out.println(e + "=> load fail");
        } catch (SQLException e) {
            System.out.println(e + "=> connection fail");
        }

        //첫번행 선택.
        if (dj.getRowCount() > 0)
            jt2.setRowSelectionInterval(0, 0);

    }
```

```
    public void getViewMovieTime(DefaultTableModel dt) {
        String sql = "select * from DBCOURSE_MOVIETIME;";
        try {
            ps = con.prepareStatement(sql);
            rs = ps.executeQuery();

            // DefaultTableModel에 있는 기존 데이터 지우기
            for (int i = 0; i < dt.getRowCount();) {
                dt.removeRow(0);
            }

            while (rs.next()) {
                Object data[] = { rs.getString(1), rs.getString(2) };

                dt.addRow(data);
            }

        } catch (SQLException e) {
            System.out.println(e + "=> getUserSearch fail");
        } finally {
            try {
                if (rs != null) rs.close();
                if (st != null) st.close();
                if (ps != null) ps.close();
            } catch (Exception e) {
                System.out.println(e + "=> dbClose fail");
            }
        }
    }

}
```

-ViewWithNestedQuery

```
import java.sql.*;

public class ViewWithNestedQuery extends JFrame {

    String[] foundS = { "director", "movie", "academy_name", "award_name"};
    DefaultTableModel dj = new DefaultTableModel(foundS,0);
    JTable jt2 = new JTable(dj);
    JScrollPane jsp2 = new JScrollPane(jt2);

    Connection con;
    Statement st;
    PreparedStatement ps;
    ResultSet rs;

    public ViewWithNestedQuery(){
        super("AwardedMovie's Director");

        setSize(700, 500);
        setVisible(true);
        add(jsp2, "Center");

        try {
            // 로드
            Class.forName("com.mysql.jdbc.Driver");

            con = DriverManager.getConnection("jdbc:mysql://localhost:3306/team01?useSSL=false","team01","team01");
            getViewAwardedDirector(dj);
        } catch (ClassNotFoundException e) {
            System.out.println(e + "=> load fail");
        } catch (SQLException e) {
            System.out.println(e + "=> connection fail");
        }

        //첫번행 선택.
        if (dj.getRowCount() > 0)
            jt2.setRowSelectionInterval(0, 0);

    }
```

```
     public void getViewAwardedDirector(DefaultTableModel dt) {
         String sql = "select director, movie, academy_name, award_name "+
                 "from  dbcourse_award "+
                 "where director = ANY(select director_name from dbcourse_director "
                 + "where director_nationality = '한국');";
         try {
            ps = con.prepareStatement(sql);
            rs = ps.executeQuery();

            // DefaultTableModel에 있는 기존 데이터 지우기
            for (int i = 0; i < dt.getRowCount();) {
               dt.removeRow(0);
            }

            while (rs.next()) {
               Object data[] = { rs.getString(1), rs.getString(2),rs.getString(3),rs.getString(4) };

               dt.addRow(data);
            }

         } catch (SQLException e) {
            System.out.println(e + "=> getUserSearch fail");
         } finally {
            try {
               if (rs != null) rs.close();
               if (st != null) st.close();
               if (ps != null) ps.close();
            } catch (Exception e) {
               System.out.println(e + "=> dbClose fail");
            }
         }
      }

}
```

1415014 김서진 산학실무설계_자료제출

1. Movie Trailer Web Site

1_1. README.md

README.md

==========

Movie Trailer Website Checklist

###Python Version
>*python 2.7.14*

###Notice
>*There are 3 python files, and 1 html file.*
>*media.py, entertainment_center.py, fresh_tomatoes.py, fresh_tomatoes.html*

>*<pre><code>entertainment_center.py</code></pre> will runs the application*

>*If you're using terminal to run the application, type <pre><code>python entertainment_center.py</code></pre>*

>*If you're using PyCharm to run the application, select Run from its menu and click <pre><code>Run</code></pre> and also click*
<pre><code> entertainment_center </code></pre>

>*If you're using Python IDLE to run the application, selet Run from the IDLE's menu and click <pre><code>Run Module</code></pre> from the dropdown list.*

1_2. fresh_tomatoes.html

```
<head>
    <meta charset="utf-8">
    <title>Fresh Tomatoes!</title>

    <!-- Bootstrap 3 -->
    <link rel="stylesheet"
href="https://netdna.bootstrapcdn.com/bootstrap/3.1.0/css/bootstrap.min.css">
    <link rel="stylesheet"
href="https://netdna.bootstrapcdn.com/bootstrap/3.1.0/css/bootstrap-theme.min.css">
    <script src="http://code.jquery.com/jquery-1.10.1.min.js"></script>
    <script
src="https://netdna.bootstrapcdn.com/bootstrap/3.1.0/js/bootstrap.min.js"></script>
    <style type="text/css" media="screen">
        body {
            padding-top: 80px;
        }
        #trailer .modal-dialog {
            margin-top: 200px;
            width: 640px;
            height: 480px;
        }
        .hanging-close {
            position: absolute;
            top: -12px;
            right: -12px;
            z-index: 9001;
        }
        #trailer-video {
```

```
            width: 100%;
            height: 100%;
        }
        .movie-tile {
            margin-bottom: 20px;
            padding-top: 20px;
        }
        .movie-tile:hover {
            background-color: #EEE;
            cursor: pointer;
        }
        .scale-media {
            padding-bottom: 56.25%;
            position: relative;
        }
        .scale-media iframe {
            border: none;
            height: 100%;
            position: absolute;
            width: 100%;
            left: 0;
            top: 0;
            background-color: white;
        }
    </style>
    <script type="text/javascript" charset="utf-8">
        // Pause the video when the modal is closed
        $(document).on('click', '.hanging-close, .modal-backdrop, .modal', function
(event) {
            // Remove the src so the player itself gets removed, as this is the only
            // reliable way to ensure the video stops playing in IE
            $("#trailer-video-container").empty();
        });
        // Start playing the video whenever the trailer modal is opened
        $(document).on('click', '.movie-tile', function (event) {
            var trailerYouTubeId = $(this).attr('data-trailer-youtube-id')
            var sourceUrl = 'http://www.youtube.com/embed/' + trailerYouTubeId +
'?autoplay=1&html5=1';
            $("#trailer-video-container").empty().append($("<iframe></iframe>", {
              'id': 'trailer-video',
              'type': 'text-html',
              'src': sourceUrl,
              'frameborder': 0
            }));
        });
        // Animate in the movies when the page loads
        $(document).ready(function () {
          $('.movie-tile').hide().first().show("fast", function showNext() {
            $(this).next("div").show("fast", showNext);
          });
        });
    </script>
</head>

<!DOCTYPE html>
<html lang="en">
  <body>
    <!-- Trailer Video Modal -->
    <div class="modal" id="trailer">
      <div class="modal-dialog">
        <div class="modal-content">
          <a href="#" class="hanging-close" data-dismiss="modal" aria-
hidden="true">
            <img src="https://lh5.ggpht.com/v4-
628SilF0HtHuHdu5EzxD7WRqOrrTIDi_MhEG6_qkNtUK5Wg7KPkofp_VJoF7RS2LhxwEFCO1ICHZlc-
o_=s0#w=24&h=24"/>
          </a>
          <div class="scale-media" id="trailer-video-container">
          </div>
```

```
        </div>
      </div>
    </div>

    <!-- Main Page Content -->
    <div class="container">
      <div class="navbar navbar-inverse navbar-fixed-top" role="navigation">
        <div class="container">
          <div class="navbar-header">
            <a class="navbar-brand" href="#">Fresh Tomatoes Movie Trailers</a>
          </div>
        </div>
      </div>
    </div>
    <div class="container">

<div class="col-md-6 col-lg-4 movie-tile text-center" data-trailer-youtube-
id="ebSQ7A1FCtc" data-toggle="modal" data-target="#trailer">
    <img src="http://www.impawards.com/2012/posters/les_miserables_ver3_xlg.jpg"
width="220" height="342">
    <h2>Les Miserables</h2>
</div>

<div class="col-md-6 col-lg-4 movie-tile text-center" data-trailer-youtube-
id="ZGZX5-PAwR8" data-toggle="modal" data-target="#trailer">
    <img
src="https://upload.wikimedia.org/wikipedia/en/7/75/Movie_poster_the_little_mermaid
.jpg" width="220" height="342">
    <h2>The Little Mermaid</h2>
</div>

<div class="col-md-6 col-lg-4 movie-tile text-center" data-trailer-youtube-
id="zSWdZVtXT7E" data-toggle="modal" data-target="#trailer">
    <img
src="https://upload.wikimedia.org/wikipedia/en/b/bc/Interstellar_film_poster.jpg"
width="220" height="342">
    <h2>Interstellar</h2>
</div>

<div class="col-md-6 col-lg-4 movie-tile text-center" data-trailer-youtube-
id="8hP9D6kZseM" data-toggle="modal" data-target="#trailer">
    <img src="http://www.impawards.com/2010/posters/inception.jpg" width="220"
height="342">
    <h2>Inception</h2>
</div>

<div class="col-md-6 col-lg-4 movie-tile text-center" data-trailer-youtube-
id="9UrQ4VvFO-c" data-toggle="modal" data-target="#trailer">
    <img src="https://images-na.ssl-images-
amazon.com/images/M/MV5BN2YyZjQ0NTEtNzU5MS00NGZkLTg0MTEtYzJmMWY3MWRhZjM2XkEyXkFqcGd
eQXVyMDA4NzMyOA@@._V1_SY1000_CR0,0,674,1000_AL_.jpg" width="220" height="342">
    <h2>Dunkirk</h2>
</div>

<div class="col-md-6 col-lg-4 movie-tile text-center" data-trailer-youtube-
id="wGEoA8_CJzk" data-toggle="modal" data-target="#trailer">
    <img src="http://www.impawards.com/1999/posters/legend_of_nineteen_hundred.jpg"
width="220" height="342">
    <h2>The Legend of 1900</h2>
</div>

    </div>
  </body>
</html>
```

1_3. fresh_tomatoes.py

```
import webbrowser
import os
import re

# Styles and scripting for the page
main_page_head = '''
<head>
    <meta charset="utf-8">
    <title>Fresh Tomatoes!</title>

    <!-- Bootstrap 3 -->
    <link rel="stylesheet"
href="https://netdna.bootstrapcdn.com/bootstrap/3.1.0/css/bootstrap.min.css">
    <link rel="stylesheet"
href="https://netdna.bootstrapcdn.com/bootstrap/3.1.0/css/bootstrap-theme.min.css">
    <script src="http://code.jquery.com/jquery-1.10.1.min.js"></script>
    <script
src="https://netdna.bootstrapcdn.com/bootstrap/3.1.0/js/bootstrap.min.js"></script>
    <style type="text/css" media="screen">
        body {
            padding-top: 80px;
        }
        #trailer .modal-dialog {
            margin-top: 200px;
            width: 640px;
            height: 480px;
        }
        .hanging-close {
            position: absolute;
            top: -12px;
            right: -12px;
            z-index: 9001;
        }
        #trailer-video {
            width: 100%;
            height: 100%;
        }
        .movie-tile {
            margin-bottom: 20px;
            padding-top: 20px;
        }
        .movie-tile:hover {
            background-color: #EEE;
            cursor: pointer;
        }
        .scale-media {
            padding-bottom: 56.25%;
            position: relative;
        }
        .scale-media iframe {
            border: none;
            height: 100%;
            position: absolute;
            width: 100%;
            left: 0;
            top: 0;
            background-color: white;
        }
    </style>
    <script type="text/javascript" charset="utf-8">
        // Pause the video when the modal is closed
        $(document).on('click', '.hanging-close, .modal-backdrop, .modal', function
(event) {
            // Remove the src so the player itself gets removed, as this is the only
            // reliable way to ensure the video stops playing in IE
            $("#trailer-video-container").empty();
        });
        // Start playing the video whenever the trailer modal is opened
        $(document).on('click', '.movie-tile', function (event) {
```

```
          var trailerYouTubeId = $(this).attr('data-trailer-youtube-id')
          var sourceUrl = 'http://www.youtube.com/embed/' + trailerYouTubeId +
'?autoplay=1&html5=1';
          $("#trailer-video-container").empty().append($("<iframe></iframe>", {
            'id': 'trailer-video',
            'type': 'text-html',
            'src': sourceUrl,
            'frameborder': 0
          }));
        });
        // Animate in the movies when the page loads
        $(document).ready(function () {
          $('.movie-tile').hide().first().show("fast", function showNext() {
            $(this).next("div").show("fast", showNext);
          });
        });
    </script>
</head>
'''

# The main page layout and title bar
main_page_content = '''
<!DOCTYPE html>
<html lang="en">
  <body>
    <!-- Trailer Video Modal -->
    <div class="modal" id="trailer">
      <div class="modal-dialog">
        <div class="modal-content">
          <a href="#" class="hanging-close" data-dismiss="modal" aria-
hidden="true">
            <img src="https://lh5.ggpht.com/v4-
628SilF0HtHuHdu5EzxD7WRqOrrTIDi_MhEG6_qkNtUK5Wg7KPkofp_VJoF7RS2LhxwEFCO1ICHZlc-
o_=s0#w=24&h=24"/>
          </a>
          <div class="scale-media" id="trailer-video-container">
          </div>
        </div>
      </div>
    </div>

    <!-- Main Page Content -->
    <div class="container">
      <div class="navbar navbar-inverse navbar-fixed-top" role="navigation">
        <div class="container">
          <div class="navbar-header">
            <a class="navbar-brand" href="#">Fresh Tomatoes Movie Trailers</a>
          </div>
        </div>
      </div>
    </div>
    <div class="container">
      {movie_tiles}
    </div>
  </body>
</html>
'''

# A single movie entry html template
movie_tile_content = '''
<div class="col-md-6 col-lg-4 movie-tile text-center" data-trailer-youtube-
id="{trailer_youtube_id}" data-toggle="modal" data-target="#trailer">
    <img src="{poster_image_url}" width="220" height="342">
    <h2>{movie_title}</h2>
</div>
'''


def create_movie_tiles_content(movies):
```

```
    # The HTML content for this section of the page
    content = ''
    for movie in movies:
        # Extract the youtube ID from the url
        youtube_id_match = re.search(r'(?<=v=)[^&#]+', movie.trailer_youtube_url)
        youtube_id_match = youtube_id_match or re.search(r'(?<=be/)[^&#]+',
movie.trailer_youtube_url)
        trailer_youtube_id = youtube_id_match.group(0) if youtube_id_match else
None

        # Append the tile for the movie with its content filled in
        content += movie_tile_content.format(
            movie_title=movie.title,
            poster_image_url=movie.poster_image_url,
            trailer_youtube_id=trailer_youtube_id
        )
    return content


def open_movies_page(movies):
    # Create or overwrite the output file
    output_file = open('fresh_tomatoes.html', 'w')

    # Replace the placeholder for the movie tiles with the actual dynamically
generated content
    rendered_content =
main_page_content.format(movie_tiles=create_movie_tiles_content(movies))

    # Output the file
    output_file.write(main_page_head + rendered_content)
    output_file.close()

    # open the output file in the browser
    url = os.path.abspath(output_file.name)
    webbrowser.open('file://' + url, new=2)  # open in a new tab, if possible
```

1_4. media.py

```
import webbrowser


class Movie():
    # class Movie stores movie information.
    def __init__(self, movie_title, poster_image, trailer_youtube):
        self.title = movie_title
        self.poster_image_url = poster_image
        self.trailer_youtube_url = trailer_youtube

    def show_trailer(self):
        # initialize the instance for opening trailer video
        webbrowser.open(self.trailer_youtube_url)
```

1_5. entertainment_center.py

```
import media
import fresh_tomatoes

# info of movies
# use media.Movie() to give movies their own custom data structure

les_miserables = media.Movie(
    "Les Miserables",
    "http://www.impawards.com/2012/posters/les_miserables_ver3_xlg.jpg",
    "https://www.youtube.com/watch?v=ebSQ7A1FCtc")
```

```
little_mermaid = media.Movie(
    "The Little Mermaid",
    "https://upload.wikimedia.org/wikipedia/en/7/75/"
    "Movie_poster_the_little_mermaid.jpg",
    "https://www.youtube.com/watch?v=ZGZX5-PAwR8")

inter_stellar = media.Movie(
    "Interstellar",
    "https://upload.wikimedia.org/wikipedia/en/b/bc/"
    "Interstellar_film_poster.jpg",
    "https://www.youtube.com/watch?v=zSWdZVtXT7E")

inception = media.Movie(
    "Inception",
    "http://www.impawards.com/2010/posters/inception.jpg",
    "https://www.youtube.com/watch?v=8hP9D6kZseM")

dunkirk = media.Movie(
    "Dunkirk",
    "https://images-na.ssl-images-amazon.com/images/"
    "M/MV5BN2YyZjQ0NTEtNzU5MS00NGZkLTg0MTEtYzJmMWY3"
    "MWRhZjM2XkEyXkFqcGdeQXVyMDA4NzMyOA@@._V1_SY1000_CR0,"
    "0,674,1000_AL_.jpg",
    "https://www.youtube.com/watch?v=9UrQ4VvFO-c")

legend_of_1900 = media.Movie(
    "The Legend of 1900",
    "http://www.impawards.com/1999/posters/"
    "legend_of_nineteen_hundred.jpg",
    "https://www.youtube.com/watch?v=wGEoA8_CJzk")

# the data of movies
movies = [
    les_miserables,
    little_mermaid,
    inter_stellar,
    inception,
    dunkirk,
    legend_of_1900
]

# use open_movies_page function in fresh_tomatoesca
fresh_tomatoes.open_movies_page(movies)
```

2. Build a Portfolio Site

2_1. css/style.css

```
*{
  font-family: 'Source Sans Pro', sans-serif;
  }
header{
  display: flex;
  }
div.my{
  flex: 0 0 auto;
  }
div.logo{
  align-self: center;
  margin-right: auto;
  }
img.udacity{
  height: 80px;
  width: auto;
  }
div.name{
  text-align: right;
  color: #2d3c49;
  }
div.dev{
  font-size: 60px;
  }
div.position{
  font-size: 20px;
  }
div.bar{
  width: 100%;
  height: 4px;
  background-color: #2d3c49;
  }
img.meta{
  padding-top: 20px;
  padding-bottom: 20px;
  max-width: 100%;
  height:auto;
  display: block;
  margin-left: auto;
  margin-right: auto;
  }
div.work{
  color: #7d97ad;
  font-size: 30px;
  font-weight: lighter;
  }
main{
/*  display: inline-block; */
  display: flex;
  flex-wrap: wrap;
  justify-content: space-between;
  width: 100%

  }
article{
  width: 100%
  color: #2d3c49;
  font-weight: lighter;
  text-align: center;
  flex: 1 1 auto;
  }
article.appify{  }
img.appify{
  max-width: 100%;
```

```
  }
p.appify{  }
article.sunflower{  }
img.sunflower{
  max-width: 100%;
  }
p.sunflower{  }
article.bokeh{  }
img.bokeh{
  max-width: 100%;
  }
p.bokeh{  }
```

2_2. index.html

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title> Portfolio Site </title>
  <link rel="stylesheet" href="css/style.css">
</head>

<body>
  <header>
    <div class="my logo">
      <img class="udacity" src="https://cdn.worldvectorlogo.com/logos/udacity.svg"
alt="udacity logo">
      <!-- TODO: need to resize box -->
    </div>
    <div class="my name">
      <div class="name dev">
        <h4>JANE DOETTE</h4>

      </div>
      <div class="name position">
        FRONT-END NINJA
      </div>
    </div>
  </header>
  <div class="bar">
    <!-- empty -->
  </div>
  <img class="meta" src="img/meta.png" alt="middle image">
  <div class="work">
    Featured Work
  </div>
  <main>
    <article>
      <img class="appify" src="img/appify.png" alt="appify image">
      <p class="appify"><h3>APPIFY</h3>
      <p>https://github.com/udacity/Appify/</p>
    </article>
    <article>
      <img class="sunflower" src="img/sunflower.png" alt="sunflower image">
      <p class="sunflower"><h3>SUNFLOWER</h3><p>
      <p>https://github.com/udacity/Sunflower/</p>
    </article>
    <article>
      <img class="bokeh" src="img/bokeh.png" alt="bokeh image">
      <p class="bokeh"><h3>BOKEH</h3>
      <p>https://github.com/udacity/Bokeh/</p>
    </article>
  </main>
</body>
</html>
```

3. Logs Analysis Project

3_1. README.md

Logs-Analysis

Udacity Nanodegree: Full Stack Web Developer Nanodegree Program Project03

Project Overview

The database contains newspaper articles, as well as the web server log for the site. The log has a database row for each time a reader loaded a web page. Using that information, your code will answer questions about the site's user activity.
The program you write in this project will run from the command line. It won't take any input from the user. Instead, it will connect to that database, use SQL queries to analyze the log data, and print out the answers to some questions.

Prerequisites

Programs
- Python3
- Vagrant
- VirtualBox

Notice

Directory
<Logs_Analysis> directory is for submit!!! Please check this directory when you do review.
Source code
1. There are 1 python file contains query and 1 text file for output.
   <pre><code>myqueries.py</code></pre> will run the query.

2. The size of "newsdqta.dql" was too big, so I didn't uploaded it. When needed, I'll upload it.

3_2. myqueries.py

```
#! /usr/bin/env python

import psycopg2
import sys


def get_data(query):
    '''
    Create connection between database and python file.
    '''
    try:
        db = psycopg2.connect("dbname=news")
        c = db.cursor()
        c.execute(query)
        posts = c.fetchall()
        db.close()
        return posts
    except Exception as e:
        print ("An Error has occured")


def most_popular_three_articles():
    query01 = """
            SELECT articles.title, count(*) AS PV
            FROM articles INNER JOIN log
```

```
            ON '/article/' || articles.slug = log.path
            AND log.status = '200 OK'
            GROUP BY articles.title
            ORDER BY PV DESC
            LIMIT 3;
            """
    result = get_data(query01)
    print("<The most popular three articles of all time>")

    for title, views in result:
        print ("%s - %d Views" % (title, views))


def most_popular_article_authors():
    query02 = """
            SELECT authors.name, count(*) AS PV
            FROM authors, log, articles
            WHERE '/article/' || articles.slug = log.path
            AND log.status = '200 OK'
            AND authors.id = articles.author
            GROUP BY authors.name
            ORDER BY PV
            DESC;
            """
    result = get_data(query02)
    print("<The most popular article authors of all time>")

    for author, views in result:
        print ("%s - %d Views" % (author, views))


def the_days_that_requests_lead_to_error():
    query03 = """
        SELECT date(log.time), cnt_error.error, cnt_total.total
        FROM log, (
            SELECT date(log.time) AS day, count(*) as error
            FROM log
            WHERE log.status = '404 NOT FOUND'
            GROUP BY date(log.time)
        ) AS cnt_error, (
            SELECT date(log.time) AS day, count(*) as total
            FROM log
            GROUP BY date(log.time)
        ) AS cnt_total
        WHERE cnt_error.error*100/cnt_total.total >= 1
        AND date(log.time) = cnt_total.day
        AND date(log.time) = cnt_error.day
        GROUP BY date(log.time), cnt_error.error, cnt_total.total
        ORDER BY date(log.time)
    """
    result = get_data(query03)

    for date, error, total in result:
        rate = (float)(error*100)/(float)(total)
        sys.stdout.write("%s - %.2f" % (date, rate))
        sys.stdout.write("% errors")

if __name__ == '__main__':
    most_popular_three_articles()
    print
    most_popular_article_authors()
    print
    the_days_that_requests_lead_to_error()
    print
    print
```

3_3. output.txt

<The most popular three articles of all time>
Candidate is jerk, alleges rival - 338647 Views
Bears love berries, alleges bear - 253801 Views
Bad things gone, say good people - 170098 Views

<The most popular article authors of all time>
Ursula La Multa - 507594 Views
Rudolf von Treppenwitz - 423457 Views
Anonymous Contributor - 170098 Views
Markoff Chaney - 84557 Views

<The days more than 1% of requests lead to errors>
2016-07-17 - 2.26% errors

4. Build an Item Catalog Application

4_1. README.md

Build-An-Item-Catalog

Udacity Nanodegree: Full Stack Web Developer Nanodegree Program Project04

Project Overview

You will develop an application that provides a list of items within a variety of categories as well as provide a user registration and authentication system. Registered users will have the ability to post, edit and delete their own items.

Prerequisites

Programs

- Python 2.7.12: https://www.python.org/downloads/release/python-2714/
- Vagrant: https://www.virtualbox.org/wiki/Downloads
- VirtualBox: https://www.virtualbox.org/wiki/Downloads
- VM configuration files: https://github.com/udacity/fullstack-nanodegree-vm

Notice

How to run the project?

1. Install Vagrant, VirtualBox programs by links provided above.
2. Download VM configuration files by links provided above.
3. By using terminal (or other cmd prompt), change the directory that VM configuration files are downloaded, and start vagrant.
4. Download /catalog zip file, and unzip it inside /vagrant directory.

For example: (WHEN VM configuration files are stored at Downloads/VM/ directory.)

<pre><code>
$ cd Downloads
$ cd VM
$ cd vagrant
$ vagrant up
$ vagrant ssh
$ cd /vagrant
$ cd /catalog
$ python database_setup.py </code></pre>

After this process, you'll get catalog.db file.

4. To run application.py file, type <pre><code> $ python application.py</code></pre> at the terminal.
5. After running the file, you should go to "localhost:8000" or "localhost:8000/category" through browser.

4_2. application.py

```
from flask import Flask, render_template, request, redirect
from flask import url_for, flash, jsonify
from sqlalchemy import create_engine, desc
from sqlalchemy.orm import sessionmaker
from database_setup import Base, Category, Item, User
from oauth2client.client import flow_from_clientsecrets
from oauth2client.client import FlowExchangeError
from flask import make_response
from flask import session as login_session
```

```
import httplib2
import json
import requests
import string
import random
from functools import wraps

app = Flask(__name__)

engine = create_engine('sqlite:///catalog.db')
Base.metadata.bind = engine

DBSession = sessionmaker(bind=engine)
session = DBSession()

CLIENT_ID = json.loads(
               open('client_secrets.json', 'r').read())['web']['client_id']
APPLICATION_NAME = 'CatalogItem'


def pre_login(f):
    """
    To check user is logged in.
    """
    @wraps(f)
    def login_function(*args, **kwargs):
        if 'user_id' not in login_session:
            return redirect(url_for('showLogin'))
        return f(*args, **kwargs)
    return login_function


@app.route('/login')
def showLogin():
    state = ''.join(random.choice(string.ascii_uppercase + string.digits)
                    for x in xrange(32))
    login_session['state'] = state
    # return "The current session state is %s" % login_session['state']
    return render_template('login.html', STATE=state)


@app.route('/logout')
def showLogout():
    if login_session['provider'] == 'google':
        gdisconnect()
        del login_session['gplus_id']
        del login_session['access_token']

    # reset user's information
    del login_session['username']
    del login_session['email']
    del login_session['picture']
    del login_session['user_id']
    del login_session['provider']

    return redirect(url_for('showCatalog'))


def addUser(login_session):
    """
    Function to add new User(if needed)
    """
    addUser = User(
        name=login_session['username'],
        email=login_session['email'],
        picture=login_session['picture'])
    session.add(addUser)
    session.commit()
```

```
    user = session.query(User).filter_by(email=login_session['email']).one()
    return user.id


@app.route('/gconnect', methods=['POST'])
def gconnect():
    """
    Gathers data from Google Sign In API.
    Places data inside a session variable.
    Login with google.
    """
    if request.args.get('state') != login_session['state']:
        response = make_response(json.dumps('Invalid state parameter.'), 401)
        response.headers['Content-Type'] = 'application/json'
        return response

    # access_token = request.data
    code = request.data

    try:
        oauth_flow = flow_from_clientsecrets('client_secrets.json', scope='')
        oauth_flow.redirect_uri = 'postmessage'
        credentials = oauth_flow.step2_exchange(code)
    except FlowExchangeError:
        response = make_response(json.dumps(
            'Failed to upgrade the authorization code.'), 401)
        response.headers['Content-Type'] = 'application/json'
        return response

    access_token = credentials.access_token
    url = (
        'https://www.googleapis.com/oauth2/v1/tokeninfo?access_token=%s' %
        access_token)
    h = httplib2.Http()
    result = json.loads(h.request(url, 'GET')[1])

    if result.get('error') is not None:
        response = make_response(json.dumps(result.get('error')), 500)
        response.headers['Content-Type'] = 'application/json'
        return response

    gplus_id = credentials.id_token['sub']
    if result['user_id'] != gplus_id:
        response = make_response(json.dumps(
            "Token's user ID doesn't match given user ID."), 401)
        response.headers['Content-Type'] = 'application/json'
        return response

    if result['issued_to'] != CLIENT_ID:
        response = make_response(json.dumps(
            "Token's client ID does not match app's."), 401)
        print "Token's client ID does not match app's."
        response.headers['Content-Type'] = 'application/json'
        return response

    stored_access_token = login_session.get('access_token')
    stored_gplus_id = login_session.get('gplus_id')

    if stored_access_token is not None and gplus_id == stored_gplus_id:
        response = make_response(json.dumps(
            'Current user is already connected.'), 200)
        response.headers['Content-Type'] = 'application/json'
        return response

    login_session['access_token'] = credentials.access_token
    login_session['gplus_id'] = gplus_id

    userinfo_url = "https://www.googleapis.com/oauth2/v1/userinfo"
```

```
    params = {'access_token': credentials.access_token, 'alt': 'json'}
    answer = requests.get(userinfo_url, params=params)

    data = answer.json()

    login_session['username'] = data['name']
    login_session['picture'] = data['picture']
    login_session['email'] = data['email']
    login_session['provider'] = 'google'

    user_id = getUserID(login_session['email'])
    if not user_id:
        user_id = addUser(login_session)
    login_session['user_id'] = user_id

    return "Logged in Successfully!"


@app.route('/gdisconnect')
def gdisconnect():
    """
    Logout from google.
    """
    access_token = login_session.get('access_token')
    if access_token is None:
        response = make_response(json.dumps(
            "Current user not connected."), 401)
        response.headers['Content-Type'] = 'application/json'
        return response

    url = 'https://accounts.google.com/o/oauth2/revoke?token=%s' % access_token
    h = httplib2.Http()
    result = h.request(url, 'GET')[0]

    if result['status'] != '200':
        # reset login information
        del login_session['gplus_id']
        del login_session['username']
        del login_session['email']
        del login_session['picture']

        response = make_response(json.dumps(
            "Failed to revoke token for given user."), 400)
        response.headers['Content-Type'] = 'application/json'
        return response


def getUserID(email):
    try:
        user = session.query(User).filter_by(email=email).one()
        return user.id
    except:
        return None


def getUserInfo(user_id):
    user = session.query(User).filter_by(id=user_id).one()
    return user


@app.route('/')
@app.route('/catalog/')
def showCatalog():
    categories = session.query(Category).all()
    items = session.query(Item).order_by(desc(Item.id))

    if 'username' not in login_session:
        return render_template(
```

```
            'public_catalog.html', categories=categories, items=items)
    else:
        return render_template(
            'private_catalog.html', categories=categories, items=items)


@app.route('/category/new', methods=['GET', 'POST'])
@pre_login
def newCategory():
    pre_categories = session.query(Category).all()
    if request.method == 'POST':
        if 'user_id' not in login_session and 'email' in login_session:
            login_session['user_id'] = getUserId(login_session['email'])
        newCategory = Category(
            categories=request.form['categories'],
            user_id=login_session['user_id'])
        session.add(newCategory)
        session.commit()
        flash("New Category %s successfully created!" % newCategory.categories)
        return redirect(url_for('showCatalog'))
    else:
        return render_template(
            'newCategory.html', pre_categories=pre_categories)


@app.route(
    '/category/<string:category_categories>/edit', methods=['GET', 'POST'])
@pre_login
def editCategory(category_categories):
    editedCategory = session.query(Category).filter_by(
        categories=category_categories).one()
    if editedCategory.user_id != login_session['user_id']:
        return render_template('notallowed.html')
    if request.method == 'POST':
        if request.form['categories']:
            editedCategory.categories = request.form['categories']
            flash(
                'Category %s successfully edited!' % editedCategory.categories)
            return redirect(url_for('showCatalog'))
    else:
        return render_template(
            'editCategory.html',
            category=editedCategory,
            category_categories=category_categories)


@app.route(
    '/category/<string:category_categories>/delete', methods=['GET', 'POST'])
@pre_login
def deleteCategory(category_categories):
    categoryToDelete = session.query(Category).filter_by(
        categories=category_categories).one()
    if categoryToDelete.user_id != login_session['user_id']:
        return render_template('notallowed.html')
    if request.method == 'POST':
        session.delete(categoryToDelete)
        session.commit()
        flash(
            'Category %s successfully deleted!' % categoryToDelete.categories)
        return redirect(url_for(
            'showCatalog',
            category_categories=category_categories))
    else:
        return render_template(
            'deleteCategory.html',
            category=categoryToDelete,
            category_categories=category_categories)
```

```
@app.route('/category/<string:category_categories>/')
@app.route('/category/<string:category_categories>/items')
def showCategorywithItem(category_categories):
    """
    Show particular category's all items.
    """
    category = session.query(Category).filter_by(
        categories=category_categories).one()
    items = session.query(Item).filter_by(
        category_categories=category_categories).all()
    return render_template(
        'showItemwithCategory.html',
        category=category,
        items=items)


@app.route('/category/item/new', methods=['GET', 'POST'])
@pre_login
def newItem():
    pre_items = session.query(Item).all()
    categories = session.query(Category).all()
    if request.method == 'POST':
        newItem = Item(
            name=request.form['name'],
            description=request.form['description'],
            category_categories=request.form['category'])
        session.add(newItem)
        session.commit()
        flash("New Item %s successfully created!" % newItem.name)
        return redirect(url_for('showCatalog'))
    else:
        return render_template(
            'newItem.html',
            categories=categories,
            pre_items=pre_items)


@app.route(
    '/category/<string:category_categories>/item/<string:item_name>')
@pre_login
def itemDetail(category_categories, item_name):
    category = session.query(Category).filter_by(
        categories=category_categories).one()
    item = session.query(Item).filter_by(name=item_name).one()
    return render_template('itemDetail.html', category=category, item=item)


@app.route(
    '/category/<string:category_categories>/item/<string:item_name>/edit',
    methods=['GET', 'POST'])
@pre_login
def editItem(category_categories, item_name):
    categories = session.query(Category).all()
    editedItem = session.query(Item).filter_by(name=item_name).one()
    if editedItem.user_id != login_session['user_id']:
        return render_template('notallowed.html')

    if request.method == 'POST':
        if request.form['name']:
            editedItem.name = request.form['name']
        if request.form['description']:
            editedItem.description = request.form['description']
        if request.form['category']:
            editedItem.category_categories = request.form['category']
        session.add(editedItem)
        session.commit()
        flash("Item %s successfully edited!" % editedItem.name)
```

```
        return redirect(url_for('showCatalog'))
    else:
        return render_template(
            'editItem.html', item=editedItem, categories=categories)


@app.route(
    '/category/<string:category_categories>/item/<string:item_name>/delete',
    methods=['GET', 'POST'])
@pre_login
def deleteItem(category_categories, item_name):
    category = session.query(Category).filter_by(
        categories=category_categories).one()
    itemToDelete = session.query(Item).filter_by(name=item_name).one()
    if itemToDelete.user_id != login_session['user_id']:
        return render_template('notallowed.html')
    if request.method == 'POST':
        session.delete(itemToDelete)
        session.commit()
        flash("Item %s successfully deleted!" % itemToDelete.name)
        return redirect(url_for('showCatalog'))
    else:
        return render_template(
            'deleteItem.html', item=itemToDelete, category=category)


@app.route('/categories.json')
def showCategoriesJSON():
    categories = session.query(Category).all()
    return jsonify(Categories=[c.serialize for c in categories])


@app.route('/items.json')
def showItemsJSON():
    items = session.query(Item).all()
    return jsonify(Items=[i.serialize for i in items])


if __name__ == '__main__':
    app.secret_key = 'super_secret_key'
    app.debug = True
    app.run(host='0.0.0.0', port=8000)
```

4_3. database_setup.py

```
import os
import sys

from sqlalchemy import Column, ForeignKey, Integer, String
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import relationship
from sqlalchemy import create_engine

Base = declarative_base()


class User(Base):
    __tablename__ = 'user'

    id = Column(Integer, primary_key=True)
    name = Column(String(250), nullable=False)
    email = Column(String(250), nullable=False)
    picture = Column(String(250))


class Category(Base):
```

```
    __tablename__ = 'category'

    id = Column(Integer, primary_key=True)
    categories = Column(String(250), nullable=False)
    user_id = Column(Integer, ForeignKey('user.id'))
    user = relationship(User)

    @property
    def serialize(self):
        return {
            'id': self.id,
            'name': self.categories
            }


class Item(Base):
    __tablename__ = 'item'

    name = Column(String(80), nullable=False)
    id = Column(Integer, primary_key=True)
    description = Column(String(250))

    category_categories = Column(String(250), ForeignKey('category.categories'))
    category = relationship(Category)
    user_id = Column(Integer, ForeignKey('user.id'))
    user = relationship(User)

    @property
    def serialize(self):
        """Return object data in easily serializeable format"""
        return {
            'id': self.id,
            'name': self.name,
            'description': self.description,
            'category': self.category_categories
        }


engine = create_engine('sqlite:///catalog.db')
Base.metadata.create_all(engine)
```

4_4. static/style.css

```
body
{ font-family: : sans-serif;}
h2, h3, h4
{ color: #1a979e;}

.bar {
  background: #a0a0a0;
  color: #ffffff;
  padding: 5px;
}
.login {
  background: #cbeff4;
  color: #ffffff;
  padding: 3px;
}
.logout {
  background: #cbeff4;
  color: #ffffff;
  padding: 3px;
}

.inform {
  color: #1a979e;
}
```

```
.flash {
  background-color: #e6f7f6;
  color: #f943d2;
}

.alert{
  background-color: #991b1b
  color: #ffffff;
}

.jsoninfo{
  color:#1a979e;
}

.itemdetails {
  color:#1a979e;
}
```

4_5. templates/deleteCategory.html

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <link rel = 'stylesheet' type = 'text/css' href='{{url_for('static', filename =
'style.css')}}'>
</head>

<body>
  <div class = "bar">
  <a href='{{url_for('showCatalog')}}'> <h1> Catalog App </h1> </a>
  </div>

<div class = "inform">
<h2> Are you sure you want to delete {{category.categories}}? </h2>
</div>

<form action = "{{url_for('deleteCategory', category_categories =
category_categories)}}" method = 'POST'>

<div>
<input type = 'submit' value = 'Delete'>
</div>

<a href = '{{url_for('showCatalog')}}'> Cancel </a>

</form>

</body>
</html>
```

4_6. templates/deleteItem.html

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <link rel = 'stylesheet' type = 'text/css' href='{{url_for('static', filename =
'style.css')}}'>
</head>


<body>
  <div class = "bar">
```

```
  <a href='{{url_for('showCatalog')}}'> <h1> Catalog App </h1> </a>
  </p>
  <div class = "inform">
    <h2> Delete Item {{item.name}} </h2>
  </p>
<div> Are you sure you want to delete {{item.name}}? </p>

<form action = "{{url_for('deleteItem', category_categories = category.categories,
item_name = item.name)}}" method = 'POST'>

<div>
<input type = 'submit' value = 'Delete'>
</p>

<a href = '{{url_for('showCatalog')}}'> Cancel </a>

</form>

</body>
</html>
```

4_7. templates/editCategory.html

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <link rel = 'stylesheet' type = 'text/css' href='{{url_for('static', filename =
'style.css')}}'>
</head>

<body>
  <div class = "bar">
  <a href='{{url_for('showCatalog')}}'> <h1> Catalog App </h1> </a>
  </div>

<div class = "inform">
<h2> Edit Category </h2>
</div>

<form action = "{{url_for('editCategory', category_categories =
category.categories)}}" method = 'POST'>

  <div> Name: </div>
    <input type = 'text' size = '30' name = 'categories' placeholder =
'{{category.categories}}'>

<input type = 'submit' value = 'Edit'>

<a href = '{{url_for('showCatalog')}}'> Cancel </a>

</form>

</body>
</html>
```

4_8. templates/editItem.html

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <link rel = 'stylesheet' type = 'text/css' href='{{url_for('static', filename =
'style.css')}}'>
</head>
```

```
<body>
<div class = "bar">
<a href='{{url_for('showCatalog')}}'> <h1> Catalog App </h1> </a>
</div>
<div class = "inform">
<h2> Edit Item {{item.name}} </h2>
</div>

<form method = 'POST'>
  <div> <b> Name: </b> </div>
    <input type = 'text' size = '30' name = 'name' placeholder = '{{item.name}}'>
  <div> <b> Description: </b> </div>
    <input type = 'text' size = '30' name = 'description' placeholder =
'{{item.description}}'>

  <div> <b> Category: </b> </div>
    <select name = "category">
    {% for c in categories %}
      {% if item and item.category_categories == c.categories %}
      <option name = "{{c.categories}}" value = "{{c.categories}}" selected>
{{c.categories}} </option>
      {% else %}
      <option name = "{{c.categories}}" value = "{{c.categories}}">
{{c.categories}} </option>
      {% endif %}
    {% endfor %}
    </select>
</br></br>
<input type = 'submit' value = 'Edit'>
<a href = '{{url_for('showCatalog')}}'> Cancel </a>

</form>

</body>
</html>
```

4_9. templates/itemDetail.html

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <link rel = 'stylesheet' type = 'text/css' href='{{url_for('static', filename =
'style.css')}}'>
</head>

<body>
  <div class = "bar">
  <a href='{{url_for('showCatalog')}}'> <h1> Catalog App </h1> </a>
  </div>
<div class = "inform">
<h2> {{item.name}} Info </h2>
</div>

<div class="itemdetails">
  <b>Name:</b>
</div>
{{item.name}}
<div class="itemdetails">
  <b>Category:</b>
</div>
  {{item.category_categories}}
<div class="itemdetails">
  <b>Description:</b>
</div>
  {{item.description}}
```

```
<div>
  <a href = '{{url_for('editItem', category_categories = item.category_categories,
item_name = item.name)}}'> edit</a>
   /
  <a href = '{{url_for('deleteItem', category_categories =
item.category_categories, item_name = item.name)}}'>delete </a>
</div>
```

4_10. templates/login.html

```
<html>
<!-- Development -->
<httpCookies httpOnlyCookies="true" requireSSL="false" />
<!-- Production -->
<!--<httpCookies domain=".domain.com" httpOnlyCookies="true" requireSSL="true" />--
>
<head>
  <meta charset="utf-8">
  <link rel = 'stylesheet' type = 'text/css' href='{{url_for('static', filename =
'style.css')}}'>
  <!--LOAD PRE-REQUISITES FOR GOOGLE SIGN IN -->
  <script src="//ajax.googleapis.com/ajax/libs/jquery/1.8.2/jquery.min.js">
  </script>
<script src="//apis.google.com/js/platform.js?onload=start" async defer> </script>
<!-- END PRE-REQUISITES FOR GOOGLE SIGN IN -->
</head>
<body>
  <div class = "bar">
  <a href='{{url_for('showCatalog')}}'> <h1> Catalog App </h1> </a>
  </div>

<div class = "show google">
  <h2>Login by Google</h2>
</div>
  <div id="signinButton">
    <span class="g-signin"
      data-scope="openid email"
      data-clientid="352063219145-
0f6dt243ee88i7mssotfgood2uosklcm.apps.googleusercontent.com"
      data-redirecturi="postmessage"
      data-accesstype="offline"
      data-cookiepolicy="single_host_origin"
      data-callback="signInCallback"
      data-approvalprompt="force">
    </span>
  </div>
</br>
  <div>
  <a href = '{{url_for('showCatalog')}}'> Cancel </a>
</div>

  <script>
  function signInCallback(authResult){
    if (authResult['code']) {
      //Hide the sign-in button now that the user is authorized,
      $('#signinButton').attr('style', 'display: none');

      //Send the one-time-use code to the server, if the server responds, write a
'login successful' message to the web page and then redirect back to the main
restaurants page
      $.ajax({
        type: 'POST',
        url: '/gconnect?state={{STATE}}',
        processData: false,
        data: authResult['code'],
        contentType: 'application/octet-stream; charset=utf-8',
```

```
        success: function(result) {
          if (result) {
            $('#result').html('Login Successful!</br>' + result +
'</br>Redirecting...')
            setTimeout(function() {
              window.location.href = "/catalog";
            }, 4000);
          } else if (authResult['error']) {
            console.log('There was an error: ' + authResult['error']);
          } else {
            $('#result').html('Failed to make a server-side call. Check your
configuration and console.');
          }
        }
      });
    }
  }
  </script>

</body>
</html>
```

4_11. templates/newCategory.html

```
<html>
<head>
    <link rel = 'stylesheet' type = 'text/css' href='{{url_for('static', filename =
'style.css')}}'>
  </head>
<body>
  <div class = "bar">
  <a href='{{url_for('showCatalog')}}'> <h1> Catalog App </h1> </a>
  </div>

<form action = "{{url_for('newCategory')}}" method = 'POST'>
  <div> <h2> Add New Category : </h2></div>
    <input type = 'text' size = '40' name = 'categories'>
    <input type = 'submit' value = 'Create'>

    <a href = '{{url_for('showCatalog')}}'>Cancel </a>


  <div> <h2> Preexistence Categories: </h2> </div>
    {% for c in pre_categories %}

    <div>
       {{c.categories}}
    </div>

  {% endfor %}

</form>

</body>
</html>
```

4_12. newItem.html

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <link rel = 'stylesheet' type = 'text/css' href='{{url_for('static', filename =
'style.css')}}'>
</head>
```

```
<body>
  <div class = "bar">
  <a href='{{url_for('showCatalog')}}'> <h1> Catalog App </h1> </a>
  </div>
  <div class = "inform">
    <h2> New Item </h2>
  </div>

<form action = "{{url_for('newItem', category_name = category_name)}}" method =
'POST'>
  <div> <h3> Name: </h3> </div>
    <input type = 'text' size = '30' name = 'name'>

  <div> <h3> Description: </h3> </div>
    <input type = 'text' size = '30' name = 'description'>

  <div> <h3> Category: </h3> </div>
  <select name = "category">
  {% for c in categories %}
    {% if item and item.category_categories == c.categories %}
    <option name = "{{c.categories}}" value = "{{c.categories}}" selected>
{{c.categories}} </option>
    {% else %}
    <option name = "{{c.categories}}" value = "{{c.categories}}"> {{c.categories}}
</option>
    {% endif %}
    {% endfor %}
  </select>

<div>
  <input type = 'submit' value = 'Submit'>
</div>

</form>
<div>
<h2> Pre Items </h2>
  {% for p in pre_items %}
<div> {{p.name}} </div>
  {% endfor %}
</div>
</body>
</html>
```

4_13. notallowed.html

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <link rel = 'stylesheet' type = 'text/css' href='{{url_for('static', filename =
'style.css')}}'>
</head>

<body>
  <div class = "alert">
    You are Not Allowed!
  </div>
</body>
</html>
```

4_14. pre_newcategory.html

```
<html>
<body>
```

```
<h1> Choose Category </h1>

<p>
  {% with messages = get_flashed_messages() %}
    {% if messages %}
    <ul>
      {% for message in messages %}
      <li><strong>{{message}}</strong></li>
      {% endfor %}
    </ul>
    {% endif %}
    {% endwith %}
  </p>

<form action = "{{url_for('newCategory')}}" method = 'POST'>

  <p> Categories: </p>
    <input type = 'text' size = '40' categories = 'categories'>

    <input type = 'submit' categories = 'categories' value = 'Create'>

  <p>
    <a href = '{{url_for('showCatalog')}}'> Cancel </a>
  </p>

</form>

</body>
</html>
```

4_15. private_catalog.html

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <link rel = 'stylesheet' type = 'text/css' href='{{url_for('static', filename =
'style.css')}}'>
</head>

<body>
<div class="bar">
<h1> Catalog App </h1>
</div>

<div class = "logout">
  {% if 'username' in session %}
    <a href="{{url_for('showLogout')}}"> Logout: {{session['email']}} </a>
  {% endif %}
</div>

<div class="flash">
  {% with messages = get_flashed_messages() %}
    {% if messages %}
    <ul>
      {% for message in messages %}
      <li><strong>{{message}}</strong></li>
      {% endfor %}
    </ul>
    {% endif %}
    {% endwith %}
</div>
<div>
<div class = "inform"> <h2> Categories </h2>
  <a href = '{{url_for('newCategory')}}'> Add Category</a> </br></br>
</div>
```

```
{% for c in categories %}
<div>
  <b> {{c.categories}} </b> (
  <a href = '{{url_for('showCategorywithItem', category_categories =
c.categories)}}'>view</a>
   /
  <a href = '{{url_for('editCategory', category_categories =
c.categories)}}'>edit</a>
   /
  <a href = '{{url_for('deleteCategory', category_categories =
c.categories)}}'>delete</a>) </br></br>
</div>

{% endfor %}
</div>

<div class = "inform"> <h2> Latest Items </h2>
  <a href = '{{url_for('newItem', category_categories = category_categories)}}'>
Add Item</a> </br></br>
</div>
<div class = "latest items">
{% for i in items %}
 {{i.id}}. {{i.name}} ({{i.category_categories}}) </br>
{% endfor %}
</div>
</div>
</br></br></br>
<div class="jsoninfo"> <b> Check JSON </b> </div>
<a href="{{url_for('showCategoriesJSON')}}">Categories</a> |
<a href="{{url_for('showItemsJSON')}}">Items</a>
</body>
</html>
```

4_16. public_catalog.html

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <link rel = 'stylesheet' type = 'text/css' href='{{url_for('static', filename =
'style.css')}}'>
</head>

<body>
<div class="bar">
<h1> Catalog App </h1>
</div>
<div class = "login" style="color:white">
{%if 'username' not in session %}
       <a href="{{url_for('showLogin')}}"> <b>Login</b> </a>

{% endif %}
</div>

<div class = "inform"> <h2> Categories </h2>
</div>
{% for c in categories %}
<div>
  <b> {{c.categories}} </b> </br>

</div>

{% endfor %}
</div>

<div class = "inform"> <h2> Latest Items </h2> </div>
<div class = "latest items">
```

```
{% for i in items %}
 {{i.id}}. {{i.name}} ({{i.category_categories}}) </br>
{% endfor %}
</div>
</div>
</br></br></br>
<div class="jsoninfo"> <b> Check JSON </b> </div>
<a href="{{url_for('showCategoriesJSON')}}">Categories</a> |
<a href="{{url_for('showItemsJSON')}}">Items</a>
</body>
</html>
```

4_17. showItemwithCategory.html

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <link rel = 'stylesheet' type = 'text/css' href='{{url_for('static', filename =
'style.css')}}'>
</head>

<body>
  <div class = "bar">
  <a href='{{url_for('showCatalog')}}'> <h1> Catalog App </h1> </a>
  </div>
  <div class = "inform">
    <h2> {{category.categories}}'s Items </h2>
  </div>
<div>
  {% for i in items %}
  <b>{{i.name}}</b> (<a href='{{url_for('itemDetail', category_categories =
category.categories, item_name = i.name)}}'>view</a>)</br>
  {% endfor %}
</div>
```

5. Neighborhood Map

5_1. README.md

Neighborhood-Map

Udacity Nanodegree: Full Stack Web Developer Nanodegree Program Project05

Project Overview

You will develop a single page application featuring a map of your neighborhood or a neighborhood you would like to visit. You will then add functionality to this map including highlighted locations, third-party data about those locations and various ways to browse the content.

Prerequisites

Programs
- Javascript
- HTML
- CSS

Notice

How to run the project?
1. Download Neighborhood_Map.zip file.
2. Unzip Neighborhood_Map.zip file and open it.
3. Open <code> index.html </code> file.

5_2. index.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Palace Map</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel = "stylesheet" href = "css/style.css">
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
crossorigin="anonymous">
  </head>

  <body>
    <div class="container">
      <div class="options-box">
        <h1>Find Your </br> Favorite Palace </br> in <font color =
"#3c75d8">Seoul</font> </h1>
      <hr>
      <div class = "palace-list">
        <b> Palace List: </b>
      </div>

      <div class = "location-list" data-bind = "foreach: locationList">
        <span class = "location-list-item" data-bind = "text: title, click:
show"></span> <br>
        </span>
      </div>

      <div class="info-box">
        <hr>
```

```
        Powered by Foursquare API </div>
      </div>
      <div id="map" class = "map-canvas"></div>
    </div>

    <script src="js/lib/knockout.min.js" type="text/javascript"></script>
    <script src="js/lib/jquery.min.js" type="text/javascript"></script>
    <script src="js/app.js" type="text/javascript"></script>       <script
src="js/data.js" type="text/javascript"></script>
    <script async defer
        src=
        "https://maps.googleapis.com/maps/api/js?libraries=geometry,drawing&key=AIz
aSyCTDkc_oQ7_zK36yuWdZ6yDehIO4LRPOFQ&v=3&callback=initMap" onerror =
"googleMapsErrorHandling()">
    </script>
  </body>
</html>
```

5_3. css/style.css

```
html,
body {
  font-family: Arial, sans-serif;
  height: 100%;
  margin: 0;
  padding: 0;
}
.container {
  height: 100%;
  width: 30%;
  position: absolute;
}
input {
  font-size: 12px;
}
h1 {
  color: #525454;
  font-size: 22px;
  margin: 0 0 10px 0;
  text-align: center;
}

hr {
  background: #D0D7D9;
  height: 1px;
  margin: 20px 0 20px 0;
  border: none;
}
.map-canvas {
  bottom:0px;
  height: 100%;
  width: 70%;
  left: 362px;
  position: absolute;
  right: 0px;
}
.info-box {
  color: #f94877;
  position: absolute;
  bottom: 0px;
  text-align: center;
}
.options-box {
  background: #fff;
  border: 1px solid #999;
  border-radius: 3px;
  height: 100%;
```

```
  line-height: 35px;
  padding: 10px 10px 30px 10px;
  text-align: left;
  width: 340px;
}
.text {
  font-size: 12px;
}

.list-group-item {
    margin: 0;
    color: black;
    border-left: solid;
    border-bottom: solid;
}

.list-group-item:focus {
    color: #fff;
    background-color: rgb(34, 34, 34);
    border-color: rgb(34, 34, 34);
}

.list-group-item:hover {
    color: #f5f5f5;
    background-color: #555;
}
.palace-list {
  margin-top: 10px;
  margin-right: 5px;
  padding-left: 5px;
  text-align: left;
  font-size: 25px;
/*  background-color: #b8c5db; */
/*  color: #3c75d8; */
}

.form-text {
  width: 66%;
  margin-right: 2px;
/*  margin-top: .25rem; */
/*  margin-bottom: 30px; */

}

.form-button {
  width: 31%;
  margin-left: 3px;
  background-color: #3c75d8;
  color: #ffffff;
  margin-top: .25rem;
}

.list-box {
  margin-bottom: 20px;
}
.location-list {
  margin-left: 20px;
}

.location-list-item {
  padding: 5px;
  margin: 0px;
  border-radius: 6px;
  color: #3c75d8;
}
```

5_4. js/app.js

```
'use strict';

var map;
var infoWindow;
var windowContent;
var marker;

// google maps init
function initMap() {
//Set center, zoom and style
  map = new google.maps.Map(document.getElementById('map'), {
    center: {lat: 37.566535, lng: 126.9779692},
    zoom: 14,
    styles: styles,
    mapTypeControl: false
  });

    infoWindow = new google.maps.InfoWindow();

    //console.log(new ViewModel());
    ko.applyBindings(new ViewModel());
    // ko.applyBindings({locationList00 : [
    //   {
    //     title00: 'GyeonBokGung Palace',
    //     location: {lat: 37.579617, lng: 126.977041},
    //     fourID: '4b68220ef964a52087682be3'
    //   }
    // ]});

} //end of initMap

var LocationMarker = function(data) {
    // Style the markers a bit. This will be our listing marker icon.
    var defaultIcon = makeMarkerIcon('f9b1d2');
    // Create a "highlighted location" marker color for when the user
    // mouses over the marker.
    var highlightedIcon = makeMarkerIcon('f20772');

    var clientID = 'IEFTGLJWGSSXVXUOTNQS2IR5MZQH3JH0GBXCWSKNEBXHCMLJ';
    var clientSecret = 'LNGMY5D55PNKNSXBQFCGOFT31AO4KYIEACZRLO1QD2S123RB';

    var self = this;
    this.title = data.title;
    this.position = data.location;
    this.street = '',
    this.city = '';
    this.visible = ko.observable(true);
    // JSON request of foursquare data
    var fourURL = 'https://api.foursquare.com/v2/venues/search?ll=' +
this.position.lat + ',' + this.position.lng + '&client_id=' + clientID +
'&client_secret=' + clientSecret + '&v=20160118' + '&query=' + this.title;

    $.getJSON(fourURL).done(function(data) {
                var results = data.response.venues[0];
        self.street = results.location.formattedAddress[0] ?
results.location.formattedAddress[0]: 'N/A';
        self.city = results.location.formattedAddress[1] ?
results.location.formattedAddress[1]: 'N/A';
    }).fail(function() {
        alert('Error on Foursquare!');
    });

    // Create marker
    this.marker = new google.maps.Marker({
        position: this.position,
        title: this.title,
```

```
        icon: defaultIcon
    });

    self.filterMarkers = ko.computed(function () {
        // set marker
        if(self.visible() === true) {
            self.marker.setMap(map);
        } else {
            self.marker.setMap(null);
        }
    });

//    console.log(showInfoWindow);

    // create click listener!!!!
    this.show = function(location) {
      google.maps.event.trigger(self.marker, 'click');
    };

    // Create an onclick even to open an indowindow at each marker
    this.marker.addListener('click', function() {
        showInfoWindow(this, self.street, self.city, infoWindow);
        map.panTo(this.getPosition());
    });
    // ref https://developers.google.com/maps/documentation/javascript/events?hl=ko

    // Event listener: change marker's color
    this.marker.addListener('click', function() {
        this.map.setZoom(16);
        this.setIcon(highlightedIcon);
    });

    this.marker.addListener('mouseover', function() {
        this.setIcon(highlightedIcon);
    });

    this.marker.addListener('mouseout', function() {
        this.setIcon(defaultIcon);
    });
};


function showInfoWindow(marker, street, city, iWindow) {
    // Check to make sure the infowindow is not already opened on this marker.
    if (iWindow.marker != marker) {
        iWindow.marker = marker;
        iWindow.setContent('');

        // Make sure the marker property is cleared if the infowindow is closed.
        iWindow.addListener('closeclick', function() {
            iWindow.marker = null;
        });

        // Not going to use Google Street view, try to find solution

        var streetViewService = new google.maps.StreetViewService();
        var radius = 50;

        windowContent = '<b><font color="#3c75d8" size = "3px">' + marker.title +
'</font></b>' + '<hr>' + '<p>' + street + '<br>' + city + '<br>' + '</p>';

//        infoWindow.setContent(windowContent);

        var getLocalWindow = function (data) {
          iWindow.setContent(windowContent);
        };

        //set information window with content
```

```
        iWindow.setContent(windowContent);

        // open marker with information at correct place
        iWindow.open(map, marker);
    }
}

// This function takes in a COLOR, and then creates a new marker
// icon of that color. The icon will be 21 px wide by 34 high, have an origin
// of 0, 0 and be anchored at 10, 34).
// Reference course's marker code
function makeMarkerIcon(markerColor) {
    var markerImage = new google.maps.MarkerImage(
        'http://chart.googleapis.com/chart?chst=d_map_spin&chld=1.15|0|' +
markerColor +
        '|40|_|%E2%80%A2',
        new google.maps.Size(21, 34),
        new google.maps.Point(0, 0),
        new google.maps.Point(10, 34),
        new google.maps.Size(21, 34));
    return markerImage;
}

// Set View Model
var ViewModel = function() {
    var self = this;
//    this.searchItem = ko.observable('');
    this.markerList = ko.observableArray([]);

    self.locationList = ko.observableArray(locationList);
    // add location markers for each location
    locationList.forEach(function(location) {
        self.markerList.push(new LocationMarker(location));
    });

    self.palaceList = ko.observableArray([]);
    self.query = ko.observable('');
    self.queryResult = ko.observable('');

    // locationList viewed on map
    console.log(locationList);
    this.locationList = ko.computed(function() {
        self.markerList().forEach(function(location) {
            location.visible(true);
        });
        return self.markerList();
    }, self);
};

// error handling
function googleMapsErrorHandling() {
    alert('Error on Google Maps!');
}
```

5_5. js/data.js

```
// hardcoded data
var locationList = [
  {
    title: 'GyeongBokGung Palace(경복궁)',
    location: {lat: 37.579617, lng: 126.977041},
    fourID: '4b68220ef964a52087682be3'
  },{
    title: 'ChangGyeongGung Palace(창경궁)',
    location: {lat: 37.5802443, lng: 126.9946543},
    fourID: '5a26351f178a2a17a8f92f3f'
```

```
  },{
    title: 'ChangDeokGung Palace and Huwon(창덕궁 후원)',
    location: {lat: 37.5791505, lng: 126.9909628},
    fourID: '4b6dacd9f964a52051852ce3'
  },{
    title: 'DeokSuGung Palace(덕수궁)',
    location: {lat: 37.5649333, lng: 126.976676},
    fourID: '4b27480ef964a5209d8524e3'
  },{
    title: 'JongMyo(종묘)',
    location: {lat: 37.574583, lng: 126.994143},
    fourID: '5352319d498e79f7447fbed1'
  },{
    title: 'GwangHwaMun Square(광화문 광장)',
    location: {lat: 37.5722389, lng: 126.9769386},
    fourID: '4c00991f8c1076b0957b2071'
  }
];

// data of google Map styles
var styles = [
  {
    featureType: 'water',
    stylers: [
      { color: '#96b7f2' }
    ]
  },{
    featureType: 'administrative',
    elementType: 'labels.text.stroke',
    stylers: [
      { color: '#ffffff' },
      { weight: 6 }
    ]
  },{
    featureType: 'administrative',
    elementType: 'labels.text.fill',
    stylers: [
      { color: '#3c75d8' }
    ]
  },{
    featureType: 'road.highway',
    elementType: 'geometry.stroke',
    stylers: [
      { color: '#fff9c4' },
      { lightness: -40 }
    ]
  },{
    featureType: 'transit.station',
    stylers: [
      { weight: 9 },
      { hue: '#f2372e' }
    ]
  },{
    featureType: 'road.highway',
    elementType: 'labels.icon',
    stylers: [

      { visibility: 'off' }
    ]
  },{
    featureType: 'water',
    elementType: 'labels.text.stroke',
    stylers: [
      { lightness: 100 }
    ]
  },{
    featureType: 'water',
```

```
    elementType: 'labels.text.fill',
    stylers: [
      { lightness: -100 }
    ]
  },{
    featureType: 'poi',
    elementType: 'geometry',
    stylers: [
      { visibility: 'on' },
      { color: '#f0e4d3' }
    ]
  },{
    featureType: 'road.highway',
    elementType: 'geometry.fill',
    stylers: [
      { color: '#efe9e4' },
      { lightness: -25 }
    ]
  }
];
```