In [2]:
```
# 라이브러리 불러오기
import numpy as np
import pandas as pd
import sys
import warnings
from datetime import datetime, timedelta
from scipy.spatial import distance
```

In [3]:
```
# 데이터 불러오기
hall=pd.read_csv('unmissing_final.csv', low_memory=False)
```

In [4]:
```
# 원본 데이터 유지를 위해 복제
df=hall.copy()
```

In [5]:
```
# 옵션 설정
pd.set_option('display.max_columns', None)
```

좌석 분리

In [29]:
```
%%time
# seat를 층, 블록, 열, 번호로 분리
floor=[]
block=[]
seat_line=[]
seat_number=[]
df['floor']=''
df['block']=''
df['seat_line']=0
df['seat_number']=0

for index, row in df.iterrows(): # BOX석과 아닐때를 구분해서 분리
    floor.append(row['seat'].split()[0]) # 층
    if row['seat'].split()[1][:3]=='BOX':
        block.append(row['seat'].split()[1][:5]) # BOX석 블록
        seat_line.append(int(row['seat'].split()[1][3])) # BOX석 열
    else:
        block.append(row['seat'].split()[1][0]) # 알파벳 블록
        seat_line.append(int(row['seat'].split()[1][3:-1])) # 알파벳 열
    seat_number.append(int(row['seat'].split()[2])) # 번호

df.floor=floor
df.block=block
df.seat_line=seat_line
df.seat_number=seat_number
```

```
CPU times: total: 29 s
Wall time: 49.8 s
```

고유한 공연 구분

In [30]:
```
df['play_date_time']=df['play_date']+' '+df['play_st_time']
```

좌석 찾기

In [32]:
```
# 좌표 데이터
seat=pd.read_csv('seat_coor.csv')
```

In [33]:
```
# 병합
df=pd.merge(df, seat, how='left', on = 'seat')
```

In [34]:
```
# 복구한 좌석을 저장할 열 생성
df['grade']=df['seat_level']
```

In [35]:
```
# x, y에 가중치를 두어 거리를 center를 기준으로 절댓값을 취해 계산하고 가장 가끼
def findmin(df, x, y, center, scale):
    min_distance=np.inf # 무한대로 초기화
    nearest_grade='' # 변수 초기화
    if scale==0: # x에 가중치
        for index, row in df.iterrows():
            d = distance.euclidean((abs(x-center)/100, y), (abs(row.x-center)/1
            if d < min_distance: # 만약 더 작을 경우(가까울 경우)
                min_distance = d # 최솟값을 변경
                nearest_grade = row.seat_level # 그때 좌석도 변경
    elif scale==1: # y에 가중치
        for index, row in df.iterrows():
            d = distance.euclidean((abs(x-center), y/100), (abs(row.x-center),
            if d < min_distance:
                min_distance = d
                nearest_grade = row.seat_level
    else: # 가중치 없음
        for index, row in df.iterrows():
            d = distance.euclidean((abs(x-center), y), (abs(row.x-center), row.
            if d < min_distance:
                min_distance = d
                nearest_grade = row.seat_level
    return nearest_grade
```

독주

In [36]:
```
# 독주 공연 개수
df[df.genre=='독주'].play_date_time.nunique()
```

Out[36]:
```
41
```

In [37]:
```
%%time
center=41 # B블록과 C블록 사이 좌표
seat_none=['unknown', 'missing', 'more', 'free'] # 모르는 좌석(바꿔야함)
seat_known=['B_price', 'R_price', 'A_price', 'S_price', 'C_price', 'normal', 't
       'second_level', 'first_level'] # 아는 좌석

for num in sorted(df[df.genre=='독주'].play_date_time.unique()): # 독주인 공연
    # 필요한 변수들과 현재 공연만 test에 저장
    test=df.loc[(df.play_date_time==num), ['seat_level', 'floor', 'block', 'sea
    known_seats = test[test.seat_level.isin(seat_known)] # 아는 좌석만 따로 저장
    if known_seats.empty: # 아는 좌석이 없으면 그대로 다음 공연으로
        continue
    for index, row in test.iterrows(): # test를 순회
        if any(row['seat_level'] in s for s in seat_none):  # 모르는 좌석일 경
            x=row.x # 현재 행의 x
            y=row.y # 현재 행의 y
            if (row['floor']=='1층'): # 현재 층이 1층일 경우
                known_seat2=known_seats[(known_seats.floor=='1층')] # 아는 좌석
                if known_seat2.empty==False: # 아는 좌석중 1층이 있을 경우
                    if (row.block in ['B', 'C', 'D']): # 현재 블록이 B, C, D일
                        known_seat3=known_seat2[known_seat2.block.isin(['B', 'C
                        if known_seat3.empty==False:
                            df.loc[index, 'grade']=findmin(known_seat3, x, y, ce
                            continue
                    else:
                        known_seat3=known_seat2[known_seat2.block.isin(['A', 'E
                        if known_seat3.empty==False:
                            df.loc[index, 'grade']=findmin(known_seat3, x, y, ce
                            continue
                    # 아는 좌석이 각 블록들에 없을 경우 가중치를 두지 않고 계산
                    df.loc[index, 'grade']=findmin(known_seat2, x, y, center, 2
                    continue
            elif (row['floor']=='2층'): # 2층(위와 유사한 방식)
                known_seat2=known_seats[(known_seats.floor=='2층')]
                if known_seat2.empty==False:
                    if (row.block in ['B', 'C', 'D']):
                        known_seat3=known_seat2[known_seat2.block.isin(['B', 'C
                        if known_seat3.empty==False:
                                df.loc[index, 'grade']=findmin(known_seat3, x, y
                                continue
                    elif (row.block in ['A', 'E']):
                        known_seat3=known_seat2[known_seat2.block.isin(['A', 'E
                        if known_seat3.empty==False:
                            df.loc[index, 'grade']=findmin(known_seat3, x, y, ce
                            continue
                    else:
                        known_seat3=known_seat2[known_seat2.block.str.len()>1]
                        if known_seat3.empty==False:
                            df.loc[index, 'grade']=findmin(known_seat3, x, y, ce
                            continue
                    df.loc[index, 'grade']=findmin(known_seat2, x, y, center, 2
                    continue
            elif (row['floor']=='3층'): # 3층(위와 유사)
                known_seat2=known_seats[(known_seats.floor=='3층')]
                if known_seat2.empty==False:
                    if (row.block in ['B', 'C', 'D', 'E', 'F']):
                        known_seat3=known_seat2[known_seat2.block.isin(['B', 'C
                        if known_seat3.empty==False:
                            if (row.block in ['C', 'D','E'])&(any(known_seat3.b
                                known_seat4=known_seat3[known_seat3.block.isin(
                                if known_seat4.empty==False:
                                    df.loc[index, 'grade']=findmin(known_seat4,
                                    continue
                            elif (row.block in ['B','F'])&(any(known_seat3.blo
                                known_seat4=known_seat3[known_seat3.block.isin(
                                if known_seat4.empty==False:
                                    df.loc[index, 'grade']=findmin(known_seat4,
                                    continue
                            else:
                                df.loc[index, 'grade']=findmin(known_seat3, x,
                                continue
                    elif (row.block in ['A', 'G']):
                        known_seat3=known_seat2[known_seat2.block.isin(['A', '
                        if known_seat3.empty==False:
                            df.loc[index, 'grade']=findmin(known_seat3, x, y, c
                            continue
                    else:
                        known_seat3=known_seat2[known_seat2.block.str.len()>1]
                        if known_seat3.empty==False:
                            df.loc[index, 'grade']=findmin(known_seat3, x, y, c
                            continue
                    df.loc[index, 'grade']=findmin(known_seat2, x, y, center, 2
                    continue
            else: # 합창석
                known_seat2=known_seats[(known_seats.floor=='합창석')]
                known_seat_glee3=known_seats[(known_seats.floor=='3층')] # 1층
                known_seat_glee2=known_seats[(known_seats.floor=='2층')] # 3층
                if known_seat2.empty==False:
                    if (row.block in ['H', 'F']):
                        known_seat3=known_seat2[known_seat2.block.isin(['H', '
                        if known_seat3.empty==False:
                            df.loc[index, 'grade']=findmin(known_seat3, x, y, c
                            continue
                    else:
                        known_seat3=known_seat2[known_seat2.block=='G']
                        if known_seat3.empty==False:
                            df.loc[index, 'grade']=findmin(known_seat3, x, y, c
                            continue
                if row[36:47].empty==False:
                    df.loc[index, 'grade']=row[36:47].index[np.argmin(row[36:47
                    continue
                else:
                    if known_seat_glee3.empty==False:
                        df.loc[index, 'grade']=findmin(known_seat_glee3, x, y,
                        continue
                    elif known_seat_glee2.empty==False:
                        df.loc[index, 'grade']=findmin(known_seat_glee2, x, y,
                        continue
            df.loc[index, 'grade']=findmin(known_seats, x, y, center, 2)
            continue
```

```
CPU times: total: 1min 10s
Wall time: 2min 10s
```

독주 제외

In [38]:
```
# 독주 제외 공연 개수
df[df.genre!='독주'].play_date_time.nunique()
```

Out[38]:
```
702
```

In [39]:
```
%%time
# 위와 동일하고 center만 무대 중앙으로 변경
center=seat[seat.seat=='무대'].x.mean() # 무대 중앙
seat_none=['unknown', 'missing', 'more', 'free']
seat_known=['B_price', 'R_price', 'A_price', 'S_price', 'C_price', 'normal', 'th
         'second_level', 'first_level']

for num in sorted(df[df.genre!='독주'].play_date_time.unique()):
    test=df.loc[(df.play_date_time==num), ['seat_level', 'floor', 'block', 'sea
    known_seats = test[test.seat_level.isin(seat_known)]
    if known_seats.empty:
        continue
    for index, row in test.iterrows():
        if any(row['seat_level'] in s for s in seat_none):
            x=row.x
            y=row.y
            if (row['floor']=='1층'):
                known_seat2=known_seats[(known_seats.floor=='1층')]
                if known_seat2.empty==False:
                    if (row.block in ['B', 'C', 'D']):
                        known_seat3=known_seat2[known_seat2.block.isin(['B', 'C
                        if known_seat3.empty==False:
                            df.loc[index, 'grade']=findmin(known_seat3, x, y, ce
                            continue
                    else:
                        known_seat3=known_seat2[known_seat2.block.isin(['A', 'E
                        if known_seat3.empty==False:
                            df.loc[index, 'grade']=findmin(known_seat3, x, y, ce
                            continue
                    df.loc[index, 'grade']=findmin(known_seat2, x, y, center, 2
                    continue
            elif (row['floor']=='2층'):
                known_seat2=known_seats[(known_seats.floor=='2층')]
                if known_seat2.empty==False:
                    if (row.block in ['B', 'C', 'D']):
                        known_seat3=known_seat2[known_seat2.block.isin(['B', 'C
                        if known_seat3.empty==False:
                                df.loc[index, 'grade']=findmin(known_seat3, x, y
                                continue
                    elif (row.block in ['A', 'E']):
                        known_seat3=known_seat2[known_seat2.block.isin(['A', 'E
                        if known_seat3.empty==False:
                            df.loc[index, 'grade']=findmin(known_seat3, x, y, ce
                            continue
                    else:
                        known_seat3=known_seat2[known_seat2.block.str.len()>1]
                        if known_seat3.empty==False:
                            df.loc[index, 'grade']=findmin(known_seat3, x, y, ce
                            continue
                    df.loc[index, 'grade']=findmin(known_seat2, x, y, center, 2
                    continue
            elif (row['floor']=='3층'):
                known_seat2=known_seats[(known_seats.floor=='3층')]
                if known_seat2.empty==False:
                    if (row.block in ['B', 'C', 'D', 'E', 'F']):
                        known_seat3=known_seat2[known_seat2.block.isin(['B', 'C
                        if known_seat3.empty==False:
                            if (row.block in ['C', 'D','E'])&(any(known_seat3.b
                                known_seat4=known_seat3[known_seat3.block.isin(
                                if known_seat4.empty==False:
                                    df.loc[index, 'grade']=findmin(known_seat4,
                                continue
                            elif (row.block in ['B','F'])&(any(known_seat3.bloc
                                known_seat4=known_seat3[known_seat3.block.isin(
                                if known_seat4.empty==False:
                                    df.loc[index, 'grade']=findmin(known_seat4,
                                    continue
                            else:
                                df.loc[index, 'grade']=findmin(known_seat3, x, y
                                continue
                        elif (row.block in ['A', 'G']):
                            known_seat3=known_seat2[known_seat2.block.isin(['A', 'G
                            if known_seat3.empty==False:
                                df.loc[index, 'grade']=findmin(known_seat3, x, y, ce
                                continue
                        else:
                            known_seat3=known_seat2[known_seat2.block.str.len()>1]
                            if known_seat3.empty==False:
                                df.loc[index, 'grade']=findmin(known_seat3, x, y, ce
                                continue
                        df.loc[index, 'grade']=findmin(known_seat2, x, y, center, 2
                        continue
                else:
                    known_seat2=known_seats[(known_seats.floor=='합창석')]
                    known_seat_glee3=known_seats[(known_seats.floor=='3층')]
                    known_seat_glee2=known_seats[(known_seats.floor=='2층')]
                    if known_seat2.empty==False:
                        if (row.block in ['H', 'F']):
                            known_seat3=known_seat2[known_seat2.block.isin(['H', 'F
                            if known_seat3.empty==False:
                                df.loc[index, 'grade']=findmin(known_seat3, x, y, ce
                                continue
                        else:
                            known_seat3=known_seat2[known_seat2.block=='G']
                            if known_seat3.empty==False:
                                df.loc[index, 'grade']=findmin(known_seat3, x, y, ce
                                continue
                    if row[36:47].empty==False:
                        df.loc[index, 'grade']=row[36:47].index[np.argmin(row[36:47
                        continue
                    else:
                        if known_seat_glee3.empty==False:
                            df.loc[index, 'grade']=findmin(known_seat_glee3, x, y, c
                            continue
                        elif known_seat_glee2.empty==False:
                            df.loc[index, 'grade']=findmin(known_seat_glee2, x, y, c
                            continue
                df.loc[index, 'grade']=findmin(known_seats, x, y, center, 2)
                continue
```

```
CPU times: total: 40min 2s
Wall time: 1h 30min 3s
```

In [40]:
```
import os
```

In [41]:
```
# found_seat.csv 에 저장
df.to_csv('found_seat.csv', index=False)
```